

Contents

- *Chapter Wise all Topics*
- *6th batch semester Final Ques & Solved!*
- *5th batch semester Final Ques & Solved!*
- *7th Batch Mid Ques & Solved!*
- *6th Batch Mid Ques & Solved!*
- *5th Batch Mid Ques & Solved!*

[Assalaamu 'Alaikum.

Here, I have tried to solve the Previous Year's Question and included all the Important Topics of DLD. I am not a regular student. So, all the answers may not be 100% correct. If I made a mistake, I request you to see it in forgiveness's eyes and give me an opportunity to correct them.

Thank you!]

-Zahid Hasan, CSE 6th Batch, University of Barishal.

Chapter Wise Topis (Collected from Mehedi)

Chapter 1

- Decimal System, Binary System
- Electrical representation of 1.0 signal

Sample ques:

- Why do computers use binary numbers/systems? ***

Chapter-2

- Conversion(Binary, Octal, Decimal, Hexa, BCD)
- Parity bit
- How even/odd parity bit method works? ***

Chapter-3

- Basic gate(OR, AND, NOT)
- Combinational/ Universal gate(NAND, NOR)
- Propagation delay
- De morgan's Theorems

- Prove NAND and NOR is Universal Gate. ***
- Explain the Universality of logic gate ***
- Explain Demorgan's Theorem. ***

Chapter-4

- K-map. ***
- Combinational logic circuit.***
- Exclusive-OR, NOR. ***
- Parity generator and checker. ****

Chapter 5.****

- Gate latch(NAND, NOR)
- clocked signals and Flip-flop
- clock master-slave j-k Flip Flop
- T-flip-flop

Sample ques:

- Explain NAND latch circuit can be used to store a bit and S-R FF Store a bit
- Draw circuit diagrams and truth tables of J-K, S-R-T, DFF
- Limitation of a JK FF and how it can be eliminated// race around condition
- What is flip flop? Limitation of S-R-FF. Explain how the limitation can be resolved.

Chapter-6

- Binary add, sub
- 2's complement system(add, sub)
- Parallel binary adder(Advantage and limitation) ***
- Design a full adder(using gate, and half adder)***
- Complete parallel adder with registers.***
- IC parallel adder. ***
- ALU IC. ***
- Carry look, Ahead Adder.***

Sample ques:

- Design a Full adder/ Draw a Full adder using two half adder
- limitation of a parallel adder. Using ICs draw a 12-bit/16-bit parallel Adder
- Draw parallel Adder/ Subtraction using 2's complement system. Explain the steps to subtract 4 from -3. Consider that your designs circuit receives only passive input

Chapter 7

- Asynchronous Ripple Counter
- Synchronous parallel counter(up/down)
- Synchronous MOD-16/14 counter
- Counters with $MOD < 2^n$
- State transition diagram of MOD 6/7/10 counter
- MOD-10 (Decode) counter
- MOD-10 (Decode) counter

Sample ques:

- Advantages of Synchronous over Asynchronous.***

Chapter-8

Term (Fanout, Propagation delay, Noise Immunity. Power requirements, IC packages, etc)

TTL (NAND, NOR)

CMOS(NAND, NOR)

Sample ques:

- Draw and explain TTL(NOR, NAND) or CMOS(NOR, NAND) gate.
- Properties of IC

Chapter 9

Encoder, Decoder, Multiplexer, De Multiplexer

Question Solved

6th batch(Final)

1.

a) Implement the Boolean function: [3]

$$F = xy + x'y' + y's$$

- i. With AND, OR, and inverter gates,
- ii. With NOR and inverter gates,
- iii. With NAND and inverter gates

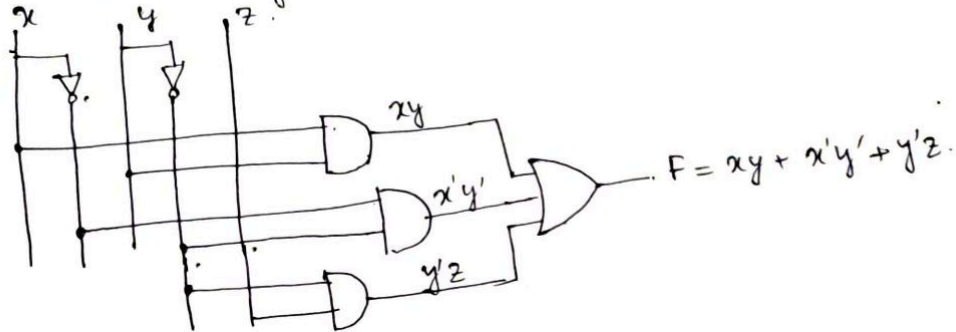
Sol:

For, i & ii

1. Given function,

$$F = xy + x'y' + y'z$$

(a) Realization using AND, OR and INVERTER Gate:-



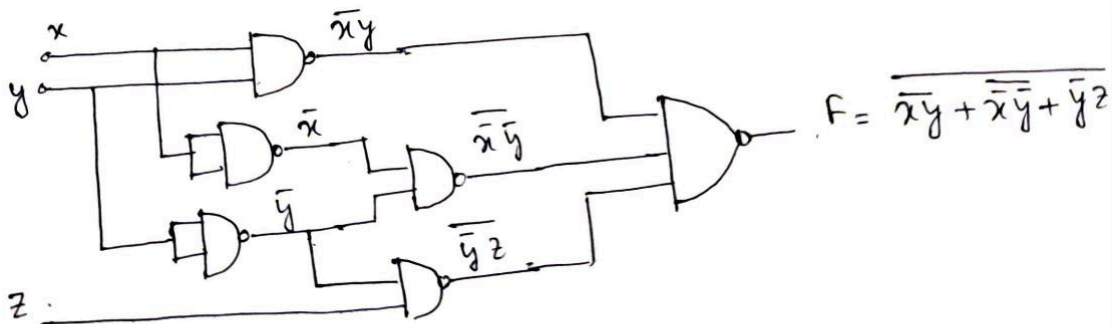
(b) NAND Gate realization:-

$$F = xy + \bar{x}\bar{y} + \bar{y}z = \overline{\overline{xy + \bar{x}\bar{y} + \bar{y}z}}$$

[If we complement two times we get the same function]

$$= \overline{\overline{xy + \bar{x}\bar{y}} \cdot \overline{\bar{y}z}} \quad [\text{using De-Morgan's theorem}]$$

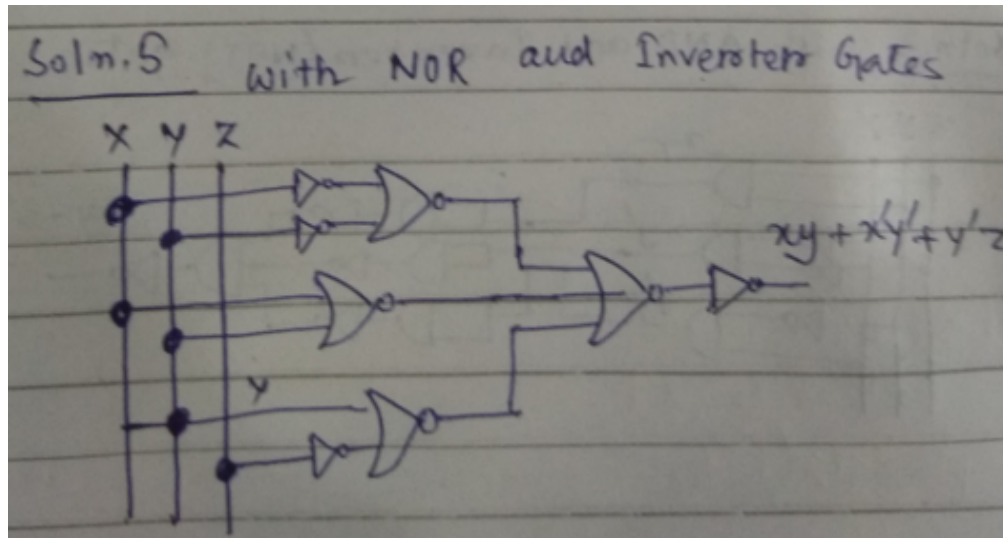
$$= \overline{\overline{xy} \cdot \overline{\bar{x}\bar{y}} \cdot \overline{\bar{y}z}}$$



Note: Uporer last e output e (+) er jaygay (-) hobe.

Sol:

iii)



b) Show that a positive logic NAND gate is a negative logic NOR gate and vice versa. [3]

Sol:

The output of a NAND gate is given by

$$Y = \overline{AB}$$

It is the complement of the product.

The output of a NOR gate is given by

$$Y = \overline{A+B}$$

Writing the truth table

A	B	$Y = \overline{AB}$	$Y = \overline{A+B}$
0	0	1	1
0	1	1	0
1	0	1	0
1	1	0	0

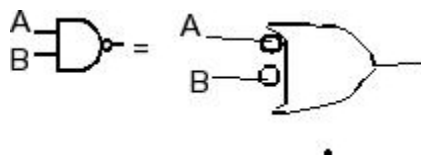
The output of a NOR gate is logically TRUE (HIGH) when input A **&** input B are **NOT** TRUE(HIGH.)

The output of a NAND gate is logically TRUE (HIGH) when input A **or** input B **or** both are not TRUE

Similarly

The output of a NOR gate is logically NOT TRUE (LOW)when input A **OR** input B **or** both are TRUE (HIGH.)

The output of a NAND gate is logically NOT TRUE (LOW) when input A **&** input B are TRUE



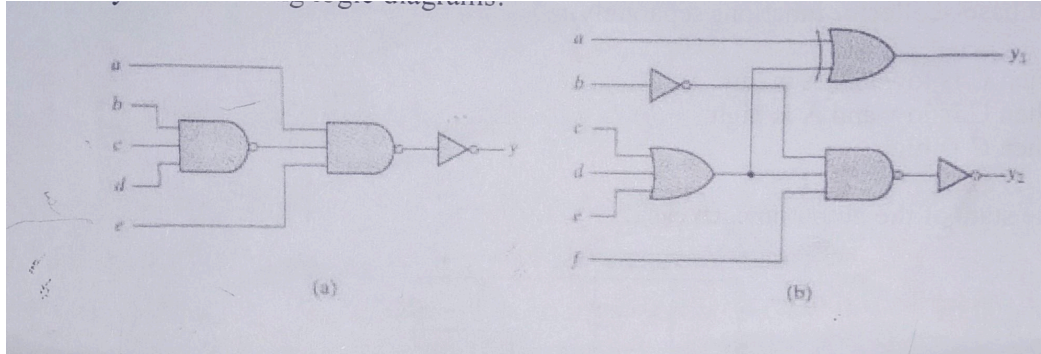
Output of gate 1 (NAND Gate) is $Y = \overline{AB}$

$$Y = A' + B'$$

Output of gate 2 (Bubbled OR gate) is $Y = A' + B'$

(Proved)

c) Write Boolean expressions and construct the truth tables describing the outputs of the circuits described by the following logic diagrams: [6]



Sol:

a)

Boolean expression is : $a.\overline{(b.c.d)}.e$

When the inputs b,c,d is given as the input for the NAND gate the output will be $\overline{(b.c.d)}$

It is given as the input for next NAND gate along with two other inputs a,e the output of this gate is $\overline{a.\overline{(b.c.d)}.e}$. This output is given as the input for the next NOT gate which complements the input that is $a.\overline{(b.c.d)}.e$

So, the output $y = a.\overline{(b.c.d)}.e$

a	b	c	d	e	$\overline{(b.c.d)}$	$\overline{a.\overline{(b.c.d)}.e}$	$y = a.\overline{(b.c.d)}.e$
0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	1	0	0	1
0	0	1	0	0	0	0	1
0	0	1	0	1	0	0	1
0	0	1	1	0	0	0	1
0	0	1	1	1	0	0	1
0	1	0	0	0	0	0	1
0	1	0	0	1	0	0	1
0	1	0	1	0	0	0	1
0	1	0	1	1	0	0	1
0	1	1	0	0	0	0	1
0	1	1	0	1	0	0	1
0	1	1	1	0	1	0	1
0	1	1	1	1	1	0	1
1	0	0	0	0	0	0	1

1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	1
1	0	0	1	1	0	0	1
1	0	1	0	0	0	0	1
1	0	1	0	1	0	0	1
1	0	1	1	0	0	0	1
1	0	1	1	1	0	0	1
1	1	0	0	0	0	0	1
1	1	0	0	1	0	0	1
1	1	0	1	0	0	0	1
1	1	0	1	1	0	0	1
1	1	1	0	0	0	0	1
1	1	1	0	1	0	0	1
1	1	1	1	0	1	0	1
1	1	1	1	1	1	1	0

b)

$$y1 = a \oplus (c + d + e)$$

$$y2 = \bar{b} \cdot (c + d + e) \cdot f$$

In this circuit

the inputs c,d,e is given to the OR gate and the output is $(c + d + e)$

the output of OR gate is given to the XOR gate along with input a to obtain

$$y1 = a \oplus (c + d + e)$$

the output of OR gate is given to the NAND gate along with complement of input b and

input f .the output of NAND gate is $\overline{b.(c + d + e).f}$

this output is given as the input of NOT gate to get y2 .NOT gate complements the input

$$y2 = \overline{b.(c + d + e).f}$$

a	b	c	d	e	f	$\bar{b}$	$(c + d + e)$	$\overline{b.(c + d + e).f}$	$y1 = a \oplus (c + d + e)$	$y2 = \overline{b.(c + d + e).f}$
0	0	0	0	0	0	1	0	1	0	0
0	0	0	0	0	1	1	0	1	0	0
0	0	0	0	1	0	1	1	1	1	0
0	0	0	0	1	1	1	1	0	1	1
0	0	0	1	0	0	1	1	1	1	0
0	0	0	1	0	1	1	1	0	1	1
0	0	0	1	1	0	1	1	1	1	0
0	0	0	1	1	1	1	1	0	1	1
0	0	1	0	0	0	1	1	1	1	0
0	0	1	0	0	1	1	1	0	1	1

0	0	1	0	1	0	1	1	1	1	1	0
0	0	1	0	1	1	1	1	0	1	1	
0	0	1	1	0	0	1	1	1	1	0	
0	0	1	1	0	1	1	1	0	1	1	
0	0	1	1	1	0	1	1	1	1	0	
0	0	1	1	1	1	1	1	0	1	1	
0	1	0	0	0	0	0	0	1	0	0	
0	1	0	0	0	1	0	0	1	0	0	
0	1	0	0	1	0	0	1	1	1	0	
0	1	0	0	1	1	0	1	1	1	0	
0	1	0	1	0	0	0	1	1	1	0	
0	1	0	1	0	1	0	1	1	1	0	
0	1	0	1	1	0	0	1	1	1	0	
0	1	0	1	1	1	0	1	1	1	0	
0	1	1	0	0	0	0	1	1	1	0	
0	1	1	0	0	1	0	1	1	1	0	
0	1	1	0	1	0	0	1	1	1	0	
0	1	1	0	1	1	0	1	1	1	0	

0	1	1	1	0	0	0	1	1	1	0
0	1	1	1	0	1	0	1	1	1	0
0	1	1	1	1	0	0	1	1	1	0
0	1	1	1	1	1	0	1	1	1	0
1	0	0	0	0	0	0	0	1	1	0
1	0	0	0	0	1	0	0	1	1	0
1	0	0	0	1	0	0	1	1	0	0
1	0	0	0	1	1	0	1	1	0	0
1	0	0	1	0	0	0	1	1	0	0
1	0	0	1	0	1	0	1	1	0	0
1	0	0	1	1	0	0	1	1	0	0
1	0	0	1	1	1	0	1	1	0	0
1	0	1	0	0	0	0	1	1	0	0
1	0	1	0	0	1	0	1	1	0	0
1	0	1	0	1	0	0	1	1	0	0
1	0	1	0	1	1	0	1	1	0	0
1	0	1	1	0	0	0	1	1	0	0
1	0	1	1	0	1	0	1	1	0	0

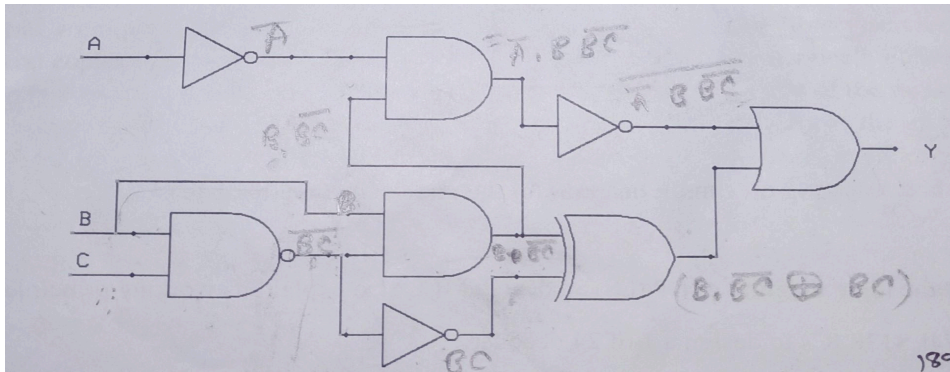
1	0	1	1	1	0	0	1	1	0	0
1	0	1	1	1	1	0	1	1	0	0
1	1	0	0	0	0	0	0	1	1	0
1	1	0	0	0	1	0	0	1	1	0
1	1	0	0	1	0	0	1	1	0	0
1	1	0	0	1	1	0	1	1	0	0
1	1	0	1	0	0	0	1	1	0	0
1	1	0	1	0	1	0	1	1	0	0
1	1	0	1	1	0	0	1	1	0	0
1	1	0	1	1	1	0	1	1	0	0
1	1	1	0	0	0	0	1	1	0	0
1	1	1	0	0	1	0	1	1	0	0
1	1	1	0	1	0	0	1	1	0	0
1	1	1	0	1	1	0	1	1	0	0
1	1	1	1	0	0	0	1	1	0	0
1	1	1	1	0	1	0	1	1	0	0
1	1	1	1	1	0	0	1	1	0	0
1	1	1	1	1	1	0	1	1	0	0

2

a) Design a logic circuit with inputs P, Q, R so that output S is HIGH whenever P is '1' or whenever Q=R=0. [3]

Sol: Not found, if anyone can solve it, please send me.

b) Simply the logic circuit shown below: [2]



Sol: That's a basic circuit. No need to solve it. Try yourself!

c) Design a Full Adder.[5]

Sol:

Inputs			Outputs	
A	B	C _{in}	Sum	Carry
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

For Sum

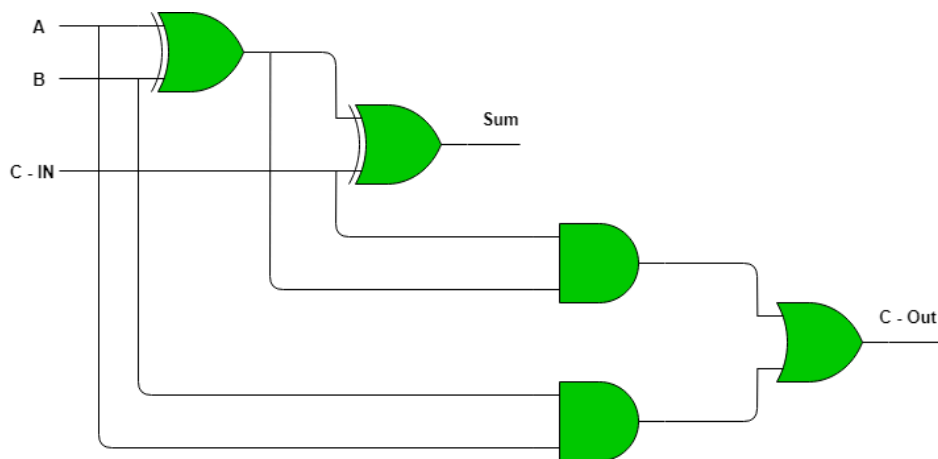
	BC			
A	00	01	11	10
0		1		1
1	1		1	

$$\begin{aligned}
 \therefore \text{Sum} &= \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC \\
 &= \bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC) \\
 &= \bar{A}(B \oplus C) + A(\overline{B \oplus C}) \quad (\because \bar{x}y + x\bar{y} = x \oplus y) \\
 &= A \oplus B \oplus C
 \end{aligned}$$

For carry

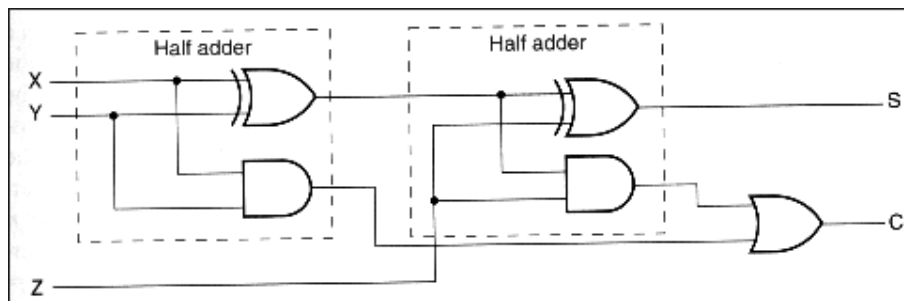
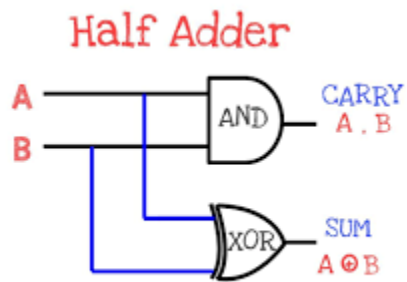
	BC			
A	00	01	11	10
0			1	
1		1	1	1

$$\text{Carry} = AB + BC + AC$$



d) Draw a Full Adder using two half adders. [2]

Sol:



3

a) Explain how an S-R FlipFlop store a bit.[3]

Sol:

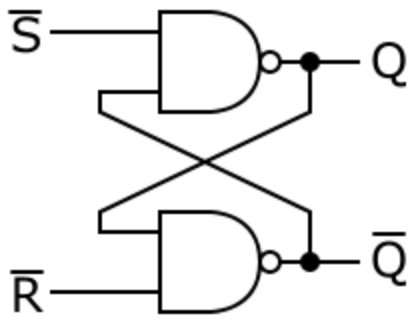


Fig: Diagram of S-R FlipFlop

The SR flip flop is a 1-bit memory bistable device having two inputs, i.e., SET and RESET. The SET input 'S' set the device or produce the output 1, and the RESET input 'R' reset the device or produce the output 0. The SET and RESET inputs are labeled as S and R, respectively.

The SR flip flop stands for "Set-Reset" flip flop. The reset input is used to get back the flip flop to its original state from the current state with an output 'Q'. This output depends on the set and reset conditions, which is either at the logic level "0" or "1".

A single flip flop stores one bit that is 0 or 1. The excitation table of flip flop tells us what input is to be applied for the desired output. Flip flop is used to store single bit. Excitation table is given as below:

Output S R

1 1 0

0 0 1

To store 1 in flip flop, S=1 and R=0 (It means high input is applied to S and low is applied to R).

To store 0 in flip flop, S=0 and R=1 (It means low input is applied to S and high is applied to R).

To keep output unchanged, $S=0$ and $R=0$ (It means low input is applied to S and R).

Hence, by the need what is to be stored in the flip flop, input is applied to the flip flop.

b) What is the limitation of S-R FlipFlop? Explain how the limitation can be resolved. [4]

Sol:

The limitation with a S-R flip-flop using NOR and NAND gate is the invalid state. This problem can be overcome by using a stable SR flip-flop that can change outputs when certain invalid states are met, regardless of the condition of either the Set or the Reset inputs.

we know it is the basic rule that Q and Q' are complement of each other.

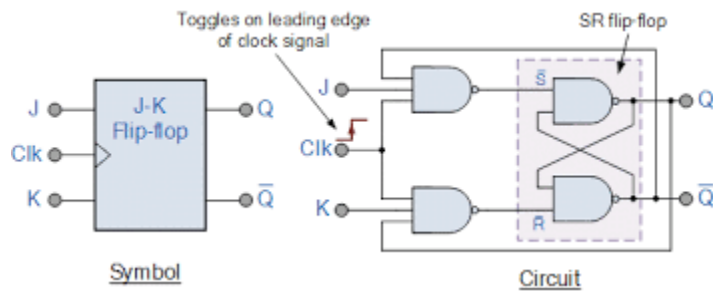
- If $Q=0$ then $Q'=1$.
- if $Q=1$ then $Q'=0$.

S-R Flip-Flop		
S	R	Q(n+1)
0	0	Q(n)
0	1	0
1	0	1
1	1	Indeterminate

When the S and R inputs of an SR flipflop is at logical 1 and then the input is changed to any other condition, then the **output** becomes unpredictable. In that condition ($Q = Q'$) which is against the rule. Thus to overcome these problem of S-R flip-flop, the **JK Filp Flop was designed** , To prevent this invalid condition a clock circuit is introduced. The JK flip flop has four possible input combinations because of the addition of the clocked input.

The j-k Flip flop has four possible input combinations because of the addition of the clocked input. The four inputs are “logic 1”, “logic 0”, “No change” and “Toggle”.

The diagram is given below:



c) Define Race Around condition in J-K FlipFlop. How can you overcome the problem? Explain with an appropriate figure and waveforms. [5]

Sol:

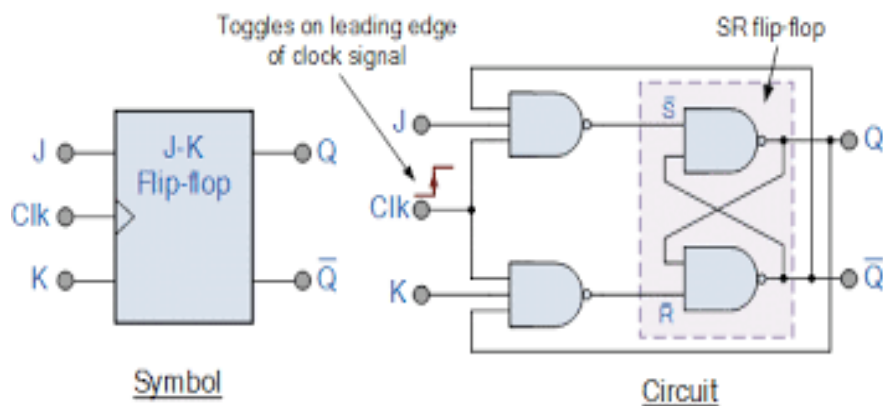
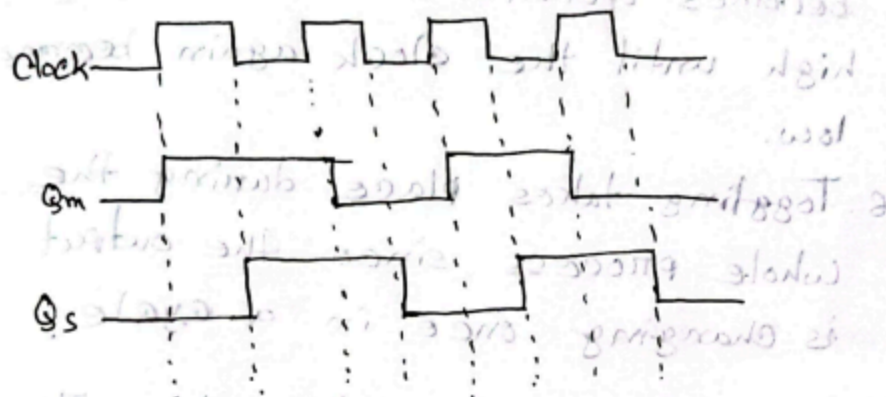


Fig: J-K Flip flop

⇒ For JK flip flop, if $J = K = 1$ and if $clk = 1$ for a long period of time then the output will toggle as long as clk is high, which makes the output of the flip flop unstable or uncertain. This problem is called race around condition in JK flip-flop.

This problem can be avoided by ensuring that the clock input is at logic "1" only for a very short time. This introduced the concept of Master slave JK flip flop.

Timing Diagram of a master slave flip flop:



1. When the clock pulse is high the output of master is high and remains high till the clock is low because the state is stored.
2. Now the output of master becomes low when the clock pulse becomes high again and remains low until the clock becomes high again.
3. Thus toggling takes place for a clock cycle.

4. When the clock pulse is high, the master is operational but not the slave thus the output of the slave remains low till the clock remains high.

Scanned with CamScanner

5. When the clock is low, the slave becomes operational and remains high until the clock again becomes low.

6. Toggling takes place during the whole process since the output is changing once in a cycle.

This makes the master-slave JK FF a synchronous device as it only passes data with the timing of the clock signal.

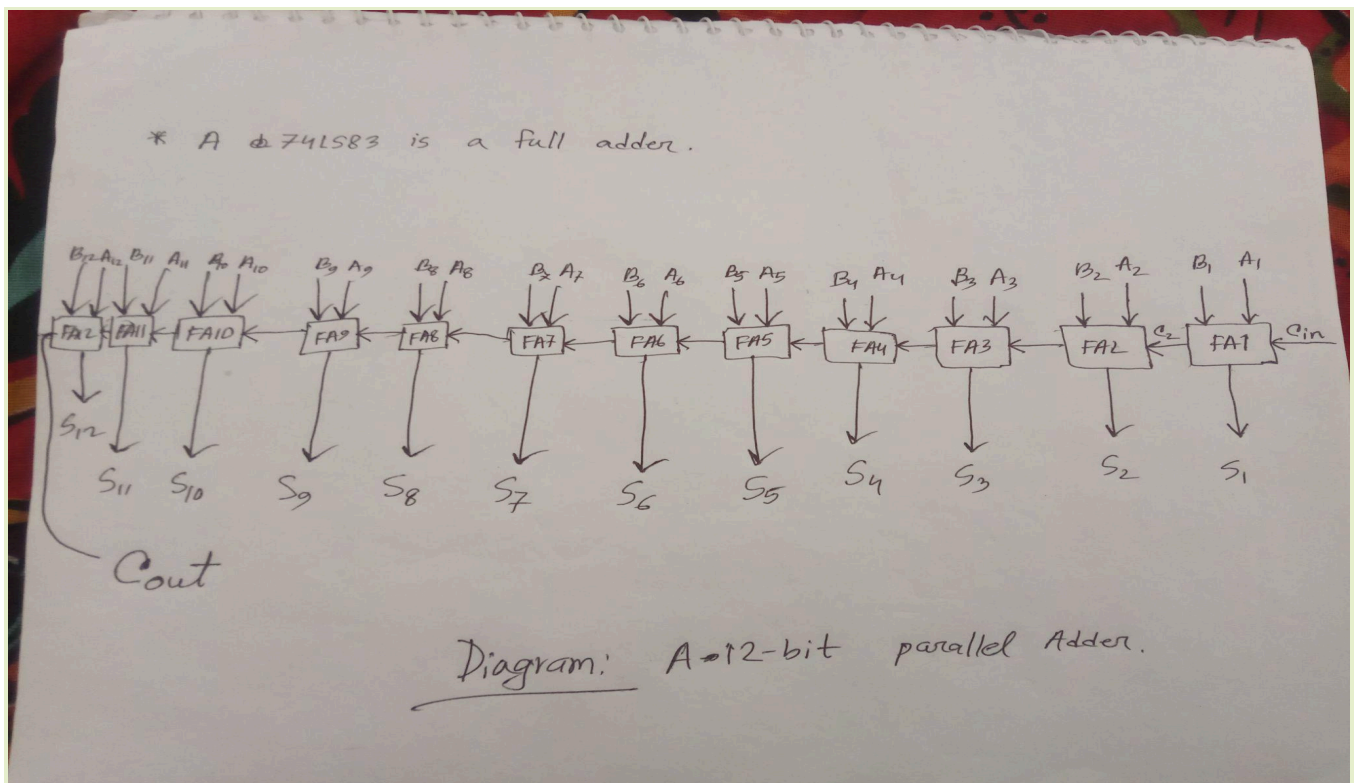
4

a) What is the limitation of parallel adder? Using 74LS83 ICs, draw a 12-bit parallel Adder.[3]

Sol:

Limitation of a parallel adder:

1. Each adder has to wait for the carry which is to be generated from the previous adder in the chain
2. The propagation delay is found to increase with the increase in the number of bits to be added.



b) Explain how 2's complement system can facilitate arithmetic operations in digital computing.[3]

Sol:

⇒ Two's complement is a mathematical operation on binary numbers and is an example of a radix complement. It is used in computer science as a method of signed number representation. The binary complement of a number is created by reversing all bits and adding 1. The carry from the high order position is eliminated. And when the Most Significant Bit is a one, the number is signed as negative. Compared to other systems for representing signed numbers, two's complement has the advantage that the fundamental arithmetic operations of addition,

subtraction and multiplication are identical to those for unsigned binary numbers. This property makes the system simpler to implement, especially for higher-precision arithmetic. Unlike one's complement system, two's complement has no representation for negative zero and thus does not suffer from its associated difficulties.

Also, complements are used in digital circuits and it is faster to subtract by adding complements than by performing true subtraction. And also the computer performs multiplication and division by doing addition and subtraction repeatedly. Thus 2's complement system can facilitate arithmetic operations.

Or,

You can also write this answer, if you want:

Advantages:

Age of Programmability:

The digital systems can be used for different applications by simply changing the program without additional changes in hardware.

Reduction in cost of hardware

The cost of hardware gets reduced by the use of digital components and this has been possible due to advances in IC technology. With ICs the number of components that can be placed in a given area of Silicon is increased which helps in cost reduction

High Speed

Digital processing of data ensures a high speed of operation which is possible due to advances in Digital Signal Processing.

High-Reliability

Digital systems are highly reliable one of the reasons for that is the use of error correction codes.

Design is Easy

The design of digital systems which require the use of Boolean algebra and other digital techniques is easier compared to analog designing.

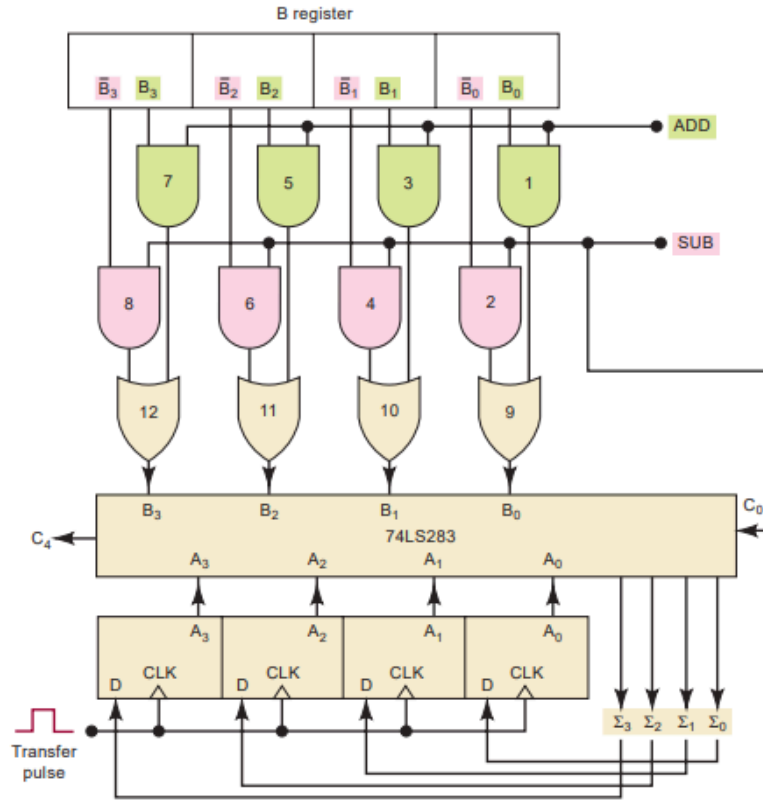
Results can be Reproduced Easily

Since the output of digital systems unlike analog systems is independent of temperature, noise, humidity, and other characteristics of components the reproducibility of results is higher in digital systems than in analog systems.

c) Draw a parallel adder/subtractor using the 2's-complement system. Using your designed circuit, explain the steps to subtract 4 from -3. Consider that your designed circuit receives only positive input.[6]

Sol:

FIGURE 6-14 Parallel adder/subtractor using the 2's-complement system.



নেক্সট অংশ (মানে, Using your designed circuit, explain the steps to subtract 4 from -3. Consider that your designed circuit receives only positive input) অনেকটা নিচে দেয়া স্যাম্পল এর মতো (বই থেকে পেলাম)। ভিডিও দেখলে লিংক এড করে দিব ইন শা আল্লাহ। আর কেও সলভ করতে পারলে আমাকে অনুগ্রহ করে সেন্ড করবেন।

Sample sol for next part:

6-4 SUBTRACTION IN THE 2's-COMPLEMENT SYSTEM

The subtraction operation using the 2's-complement system actually involves the operation of addition and is really no different from the various cases for addition considered in Section 6-3. When subtracting one binary number

CHAPTER 6/DIGITAL ARITHMETIC: OPERATIONS AND CIRCUITS

(the **subtrahend**) from another binary number (the **minuend**), use the following procedure:

1. *Negate the subtrahend.* This will change the subtrahend to its equivalent value of opposite sign.
2. *Add this to the minuend.* The result of this addition will represent the *difference* between the subtrahend and the minuend.

Once again, as in all 2's-complement arithmetic operations, it is necessary that both numbers have the same number of bits in their representations.

Let us consider the case where +4 is to be subtracted from +9.

$$\begin{array}{ll}\text{minuend (+9)} \rightarrow & 01001 \\ \text{subtrahend (+4)} \rightarrow & 00100\end{array}$$

Negate the subtrahend to produce 11100, which represents -4 . Now add this to the minuend.

$$\begin{array}{rcl} & 01001 & (+9) \\ + & 11100 & (-4) \\ \hline 1 & 00101 & (+5) \\ \uparrow & & \text{Disregard, so the result is } 00101 = +5. \end{array}$$

When the subtrahend is changed to its 2's complement, it actually becomes -4 , so that we are *adding* -4 and $+9$, which is the same as subtracting $+4$ from $+9$. This is the same as case II of Section 6-3. Any subtraction operation, then, actually becomes one of addition when the 2's-complement system is used. This feature of the 2's-complement system has made it the most widely used of the methods available because it allows addition and subtraction to be performed by the same circuitry.

Here's another example showing $+9$ subtracted from -4 :

$$\begin{array}{rcl} & 11100 & (-4) \\ - & 01001 & (+9) \\ \hline \end{array}$$

Negate the subtrahend $(+9)$ to produce 10111 (-9) and add this to the minuend (-4) .

Negate the subtrahend (+9) to produce 10111 (−9) and add this to the minuend (−4).

$$\begin{array}{r}
 11100 \quad (-4) \\
 + 10111 \quad (-9) \\
 \hline
 1\ 10011 \quad (-13) \\
 \uparrow \text{Disregard}
 \end{array}$$

The reader should verify the results of using the above procedure for the following subtractions: (a) $+9 - (-4)$; (b) $-9 - (+4)$; (c) $-9 - (-4)$; (d) $+4 - (-4)$. Remember that when the result has a sign bit of 1, it is negative and in 2's-complement form.

a) Explain the ‘working, principle of synchronous up/down MOD-16 counter with an appropriate diagram. [7]

Sol:

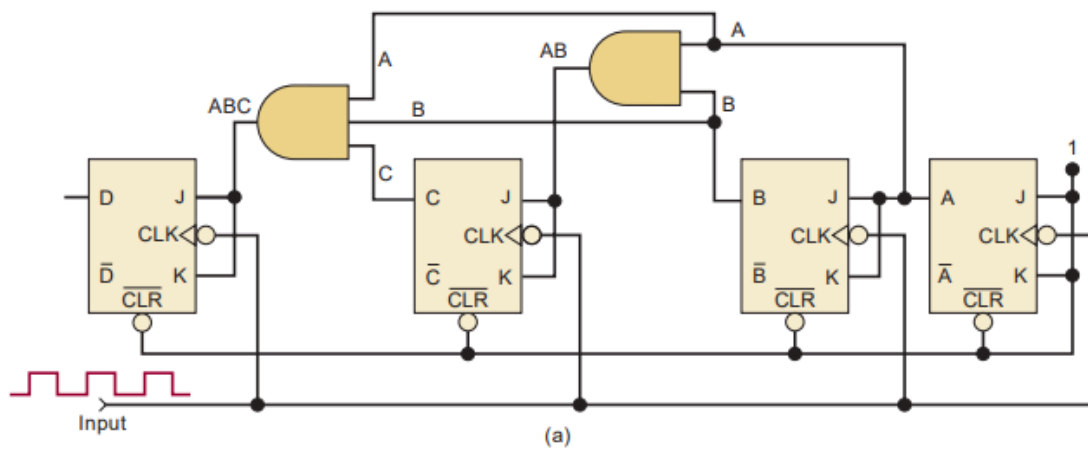


fig: 7.5; synchronous up counter

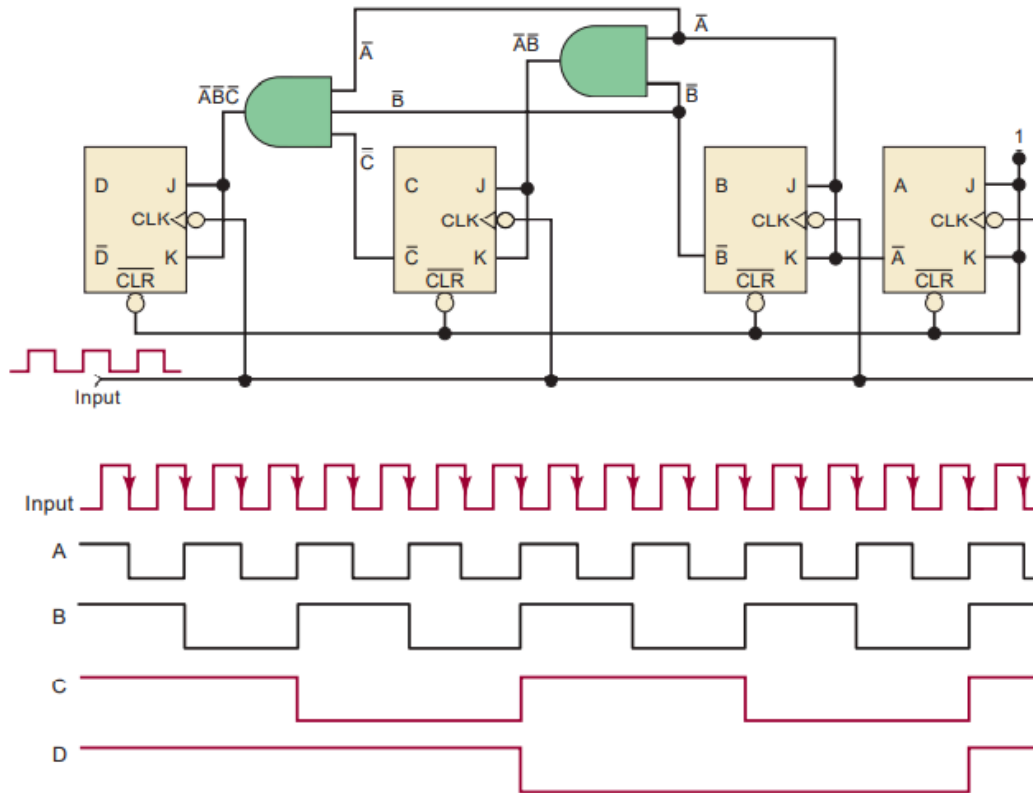


FIGURE 7-10 Synchronous, MOD-16, down counter and output waveforms.

A synchronous down counter is constructed in a similar manner except that we use the inverted FF outputs to control the higher-order J, K inputs. Comparing the synchronous, MOD-16, down counter in Figure 7-10 with the up counter in Figure 7-5 shows that we need only to substitute the corresponding inverted FF output in place of the A, B, and C outputs. For a down count sequence, the LSB FF (A) still needs to toggle with each NGT of the clock input signal. Flip-flop B must change states on the next NGT of the clock when A = 0. Flip-flop C changes states when A = B = 0 and flip-flop D changes states when A = B = C = 0. This circuit configuration will produce the count sequence: 15, 14, 13, 12, . . . , 3, 2, 1, 0, 15, 14, and so on, as shown in the timing diagram.

b) Use two 74293 ICs to design a mod 50 counter.[5]

***Sol:** Not found, if anyone can solve it, please send me.*

6

a) Show the circuit of a four-input NAND and NOR gates using CMOS transistors. [4]

***Sol:** Not found, if anyone can solve it, please send me.*

b) Construct an exclusive-OR circuit with two inverters and two transmission gates. [4]

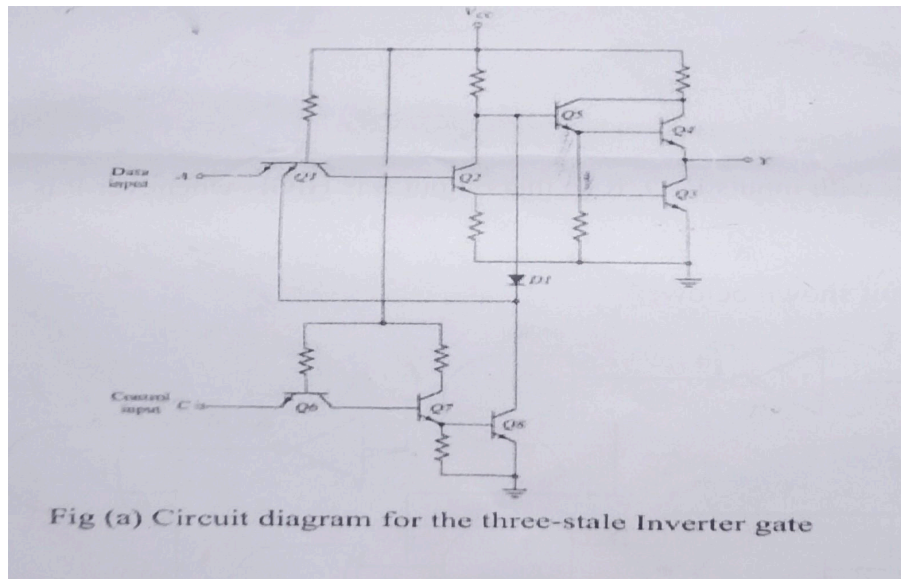
***Sol:** Not found, if anyone can solve it, please send me.*

c) For the following conditions, list the transistors that are off and the transistors that are conducting in the three-state TTL gate of Fig (a). (for Q1 and Q6. it is necessary to list the states in the base-emitter and base-collector junctions separately): [4]

- i) When C is low and A is low.
- ii) When C is low and A is high.
- iii) When C is high.

***Sol:** Not found, if anyone can solve it, please send me.*

What is the state of the output in each case?



Sol: Not found, if anyone can solve it, please send me.

7

a) Draw the logic diagram of a 74LS138 decoder IC. Also, explain its working principle.[5]

Sol:

FIGURE 9-3 (a) Logic diagram for the 74ALS138 decoder; (b) truth table; (c) logic symbol.

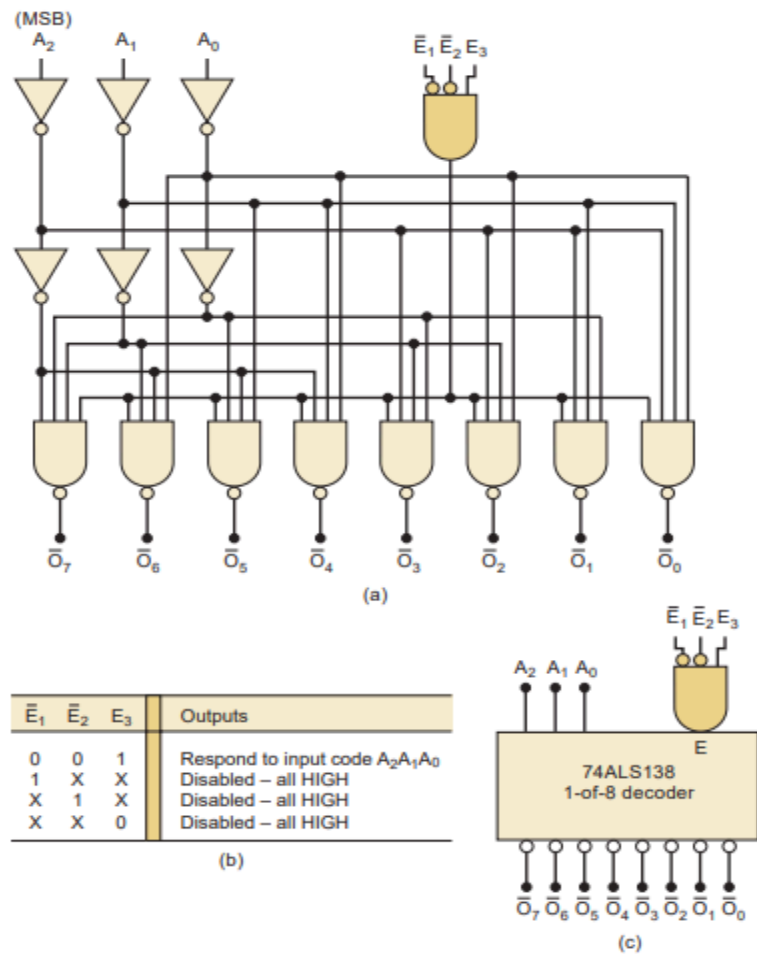


Figure 9-3(a) shows the logic diagram for the 74ALS138 decoder. By examining this diagram carefully, we can determine exactly how this decoder functions. First, notice that it has NAND gate outputs, so its outputs are active-LOW. Another indication is the labeling of the outputs as $\bar{O}_7$, and so on; the overbar indicates active-LOW outputs. The input code is applied at A_2 , A_1 , and A_0 , where A_2 is the MSB. With three inputs and eight outputs, this is a 3-to-8 decoder or, equivalently, a 1-of-8 decoder. Inputs $\bar{E}_1$, $\bar{E}_2$, and E_3 are separate enable inputs that are combined in the AND gate. In order to enable the output NAND gates to respond to the input code at A_2 , A_1 , and A_0 , this AND gate output must be HIGH. This will occur only when $\bar{E}_1$ and $\bar{E}_2$ are LOW and E_3 is HIGH. In other words, $\bar{E}_1$ and $\bar{E}_2$ are active-LOW, E_3 is active-HIGH, and all three must be in their active states to activate the decoder outputs. If one or more of the enable inputs is in its inactive state, the AND output will be LOW, which will force all NAND outputs to their inactive HIGH state regardless of the input code. This operation is summarized in the truth table in Figure 9-3(b). Recall that x represents the don't-care condition.

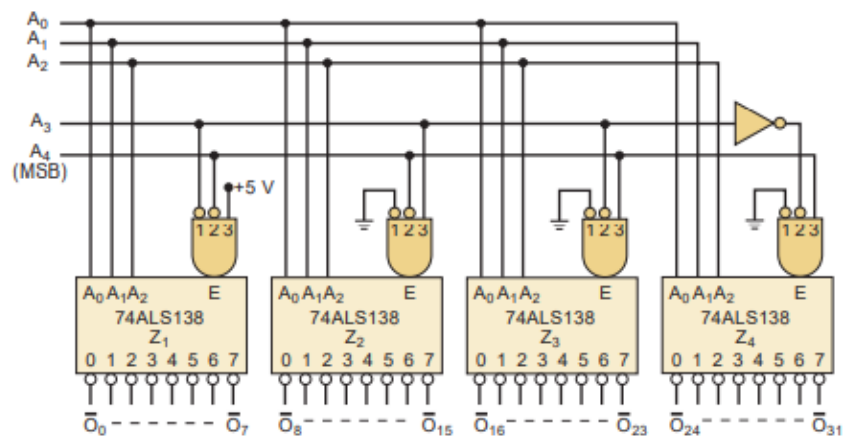
The logic symbol for the 74ALS138 is shown in Figure 9-3(c). Note how the active-LOW outputs are represented and how the enable inputs are represented. Even though the enable AND gate is shown as external to the decoder block, it is part of the IC's internal circuitry. The 74HC138 is the high-speed CMOS version of this decoder

b) Use 74LS138 ICs to design a 1-of-24 decoder.[4]

Sol:

(অনেক খুঁজলাম, শেষে বইতে 1-of-32 decoder ই পেলাম। 1-of-24 এর জন্য এভাবেই হবে, জাস্ট ১ টা সার্কিট কমে দিলেই হয়ে যাবে মেরি)

FIGURE 9-4 Four 74ALS138s forming a 1-of-32 decoder.



c) How Liquid Crystal Display (LCD) work? [3]

Sol:

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly, instead using a backlight or reflector to produce images in color or monochrome

or,

You probably use items containing an LCD (liquid crystal display) every day. They are all around us -- in [laptop computers](#), [digital clocks](#) and [watches](#), [microwave ovens](#), [CD players](#) and many other electronic devices. LCDs are common because they offer some real advantages over other display technologies. They are thinner and lighter and draw much less power than [cathode ray tubes](#) (CRTs), for example.

But just what are these things called liquid crystals? The name "liquid crystal" sounds like a contradiction. We think of a crystal as a solid material like quartz, usually as hard as rock, and a liquid is obviously different. How could any material combine the two?

We learned in school that there are three common states of matter: solid, liquid or gaseous. Solids act the way they do because their molecules always maintain their orientation and stay in the same position with respect to one another. The molecules in liquids are just the opposite: They can change their orientation and move anywhere in the liquid. But there are some substances that can exist in an odd state that is sort of like a liquid and sort of like a solid. When they are in this state, their molecules tend to maintain their orientation, like the molecules in a solid, but also move around to different positions, like the molecules in a liquid. This means that liquid crystals are neither a solid nor a liquid. That's how they ended up with their seemingly contradictory name.

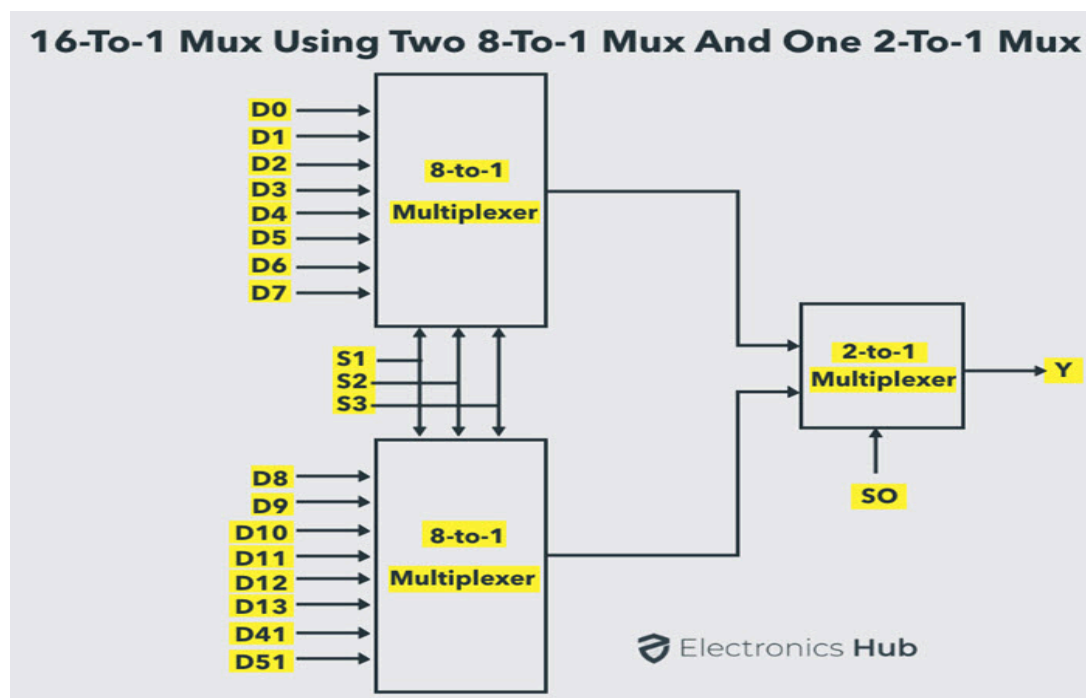
So, do liquid crystals act like solids or liquids or something else? It turns out that liquid crystals are closer to a liquid state than a solid. It takes a fair amount of heat to change a suitable substance from a solid into a liquid crystal, and it only takes a little more heat to turn that same liquid crystal into a real liquid. This explains why liquid crystals are very sensitive to temperature and why they are used to make [thermometers](#) and [mood rings](#). It

also explains why a [laptop computer](#) display may act funny in cold weather or during a hot day at the beach.

8

a) Design a 16-input multiplexer using 74151 ICs.[4]

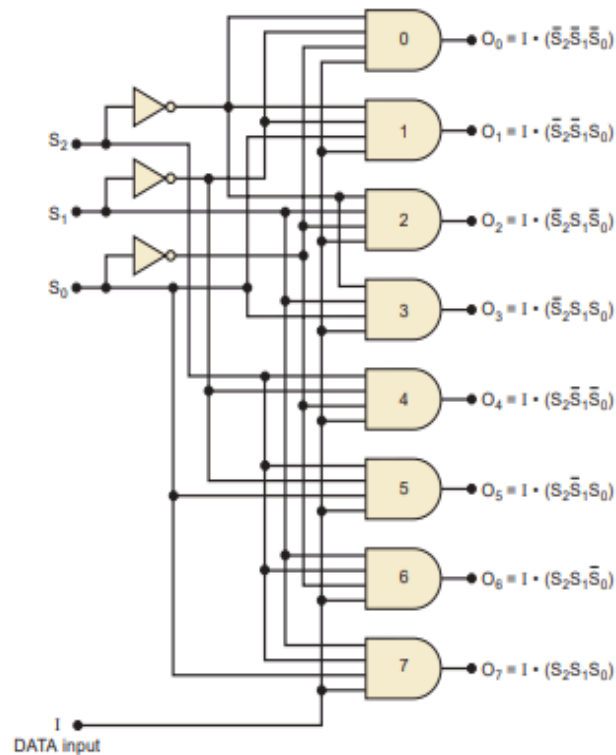
Sol:



b) With an appropriate diagram explain the working principle of a 1-line to 8-line demultiplexer.[5]

Sol:

FIGURE 9-29 A 1-line-to-8-line demultiplexer.



SELECT code			OUTPUTS							
S ₂	S ₁	S ₀	O ₇	O ₆	O ₅	O ₄	O ₃	O ₂	O ₁	O ₀
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Note: I is the data input

FIGURE 9-30 (a) The

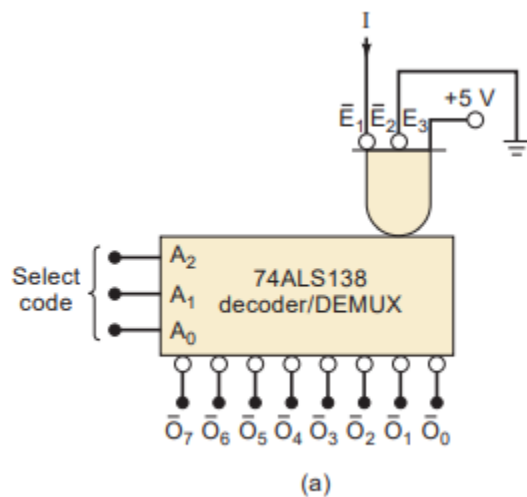
Data input

Figure 9-29 shows the logic diagram for a demultiplexer that distributes one input line to eight output lines. The single data input line I is connected to all eight AND gates, but only one of these gates will be enabled by the SELECT input lines. For example, with only AND gate 0 will be enabled, and data input I will appear at output . Other SELECT codes cause input I to reach the other outputs. The truth table summarizes the operation. The demultiplexer circuit of Figure 9-29 is very similar to the 3-line-to-8- line decoder circuit in Figure 9-2 except that a fourth input (I) has been added to each gate. It was pointed out earlier that many IC decoders have an ENABLE input, which is an extra input added to the decoder gates. This type of decoder chip can therefore be used as a demultiplexer, with the

binary code inputs (e.g., A, B, C in Figure 9-2) serving as the SELECT inputs and the ENABLE input serving as the data input I. For this reason, IC manufacturers often call this type of device a decoder/demultiplexer, and it can be used for either function.

c) How can you use a 74LS138 IC as a DEMUX? [3]

Sol:



We saw that 74ALS138 how can be used as a demultiplexer. The enable input is used as the data input I, while the other two enable inputs are held in their active states. The inputs are used as the select code. To illustrate the operation, let's assume that the select inputs are 000. With this input code, the only output that can be activated is while all other outputs are HIGH. will go LOW only if goes LOW and will be HIGH if goes HIGH. In other words, will follow the signal on (i.e., the data input, I) while all other outputs stay HIGH. In a similar manner, a different select code applied to A2A1A0 will cause the corresponding output to follow the data input, I.

5th batch(Final)

1.

a. What are the advantages and disadvantages of Digital Technology?[3]

Sol:

PROS & CONS OF DIGITAL TECHNOLOGY

PROS	CONS
1. Connectivity.	1. Data Security.
2. Communication Speed and versatile working.	2. Social Isolation.
3. Digital Devices are Portable.	3. Work Overload.
4. Quality of Stored Information is Preserved.	4. Diminishing Job Opportunity.
5. Learning Re-Defined.	5. Digital Technology Creates Addiction Towards it.
6. Entertainment	6. Manipulation of Digital Media.
7. Transportation.	7. Plagiarism and Copyright.

HONESTPROSCONS.COM

b. Convert 35.65(10) to Binary, Octal, and Hex number system. [4]

Or,

Convert (249.63)₁₀ to BCD, Octal, Binary, and Hex [10]

Sol:

For videos: <https://www.youtube.com/watch?v=8KklbjP7HRA>

$(35.65)_{10}$ to binary

$$\begin{array}{r} 2 \overline{) 10} \\ \underline{0} \\ 0 \end{array}$$

$$\begin{array}{r} 2 \overline{) 35} \\ \underline{2 \times 17} \quad 1 \\ 2 \overline{) 17} \quad 1 \\ \underline{2 \times 8} \quad 1 \\ 2 \overline{) 9} \quad 0 \\ \underline{2 \times 4} \quad 0 \\ 2 \overline{) 1} \quad 0 \\ \underline{2 \times 0} \quad 1 \end{array}$$

MSB

$$\begin{array}{r} .65 \\ \times 2 \\ \hline 1.30 \\ \times 2 \\ \hline 0.60 \\ \times 2 \\ \hline 1.20 \\ \times 2 \\ \hline 0.40 \\ \times 2 \\ \hline 0.80 \\ \times 2 \\ \hline 1.60 \end{array}$$

$\therefore (35.65)_{10} = (100011.101001 \dots)_2$

$(35.65)_{10}$ to octal

$$\begin{array}{r} 8 \overline{) 35} \\ \underline{8 \times 4} \quad 3 \\ 8 \overline{) 7} \quad 0 \end{array}$$

$$\begin{array}{r} .65 \\ \times 8 \\ \hline 5.20 \\ \times 8 \\ \hline 1.60 \\ \times 8 \\ \hline 4.80 \\ \times 8 \\ \hline 6.40 \end{array}$$

$(35.65)_{10} = (43.5146 \dots)_8$

$(35.65)_{10}$ to hexadecimal

$$\begin{array}{r}
 16 \overline{) 35} \\
 \underline{16 2} \\
 16 \overline{) 0} \\
 2
 \end{array}
 \begin{array}{l}
 \uparrow \\
 2
 \end{array}$$

$$\begin{array}{r}
 .65 \\
 \times 16 \\
 \hline
 10.40 \\
 \times 16 \\
 \hline
 6.40 \\
 \times 16 \\
 \hline
 6.40 \\
 \times 16 \\
 \hline
 6.40
 \end{array}$$

$$\begin{aligned}
 \therefore (35.65)_{10} &= (23.10666)_8 \\
 &= (23.A666\dots)_8
 \end{aligned}$$

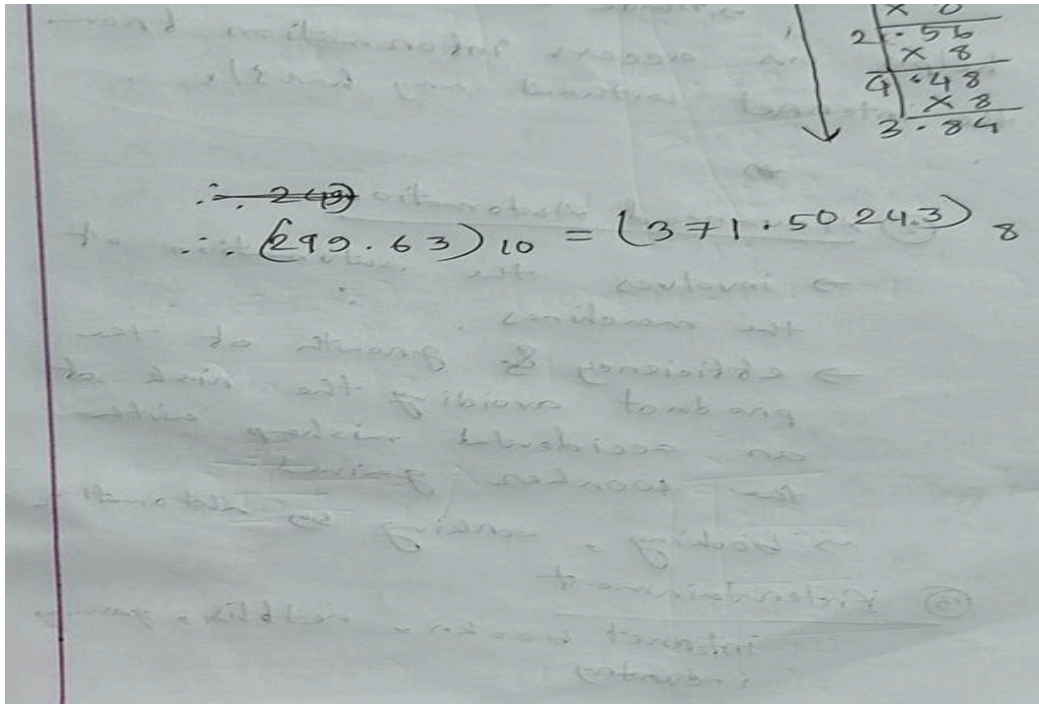
$(249.63)_{10}$ to BCD

$$\begin{aligned}
 2 &\rightarrow 0010 \\
 4 &\rightarrow 0100 \\
 9 &\rightarrow 1001 \\
 6 &\rightarrow 0110 \\
 3 &\rightarrow 0011
 \end{aligned}$$

$$\therefore (249.63)_{10} = (0010\ 0100\ 1001.0110\ 0011)_{BCD}$$

$$\begin{array}{r}
 8421 \\
 \hline
 0010 \\
 0100 \\
 1001
 \end{array}$$

$$\begin{array}{r}
 0110 \\
 0011
 \end{array}$$



Extra:

For anything to decimal: <https://www.youtube.com/watch?v=f4vFad1QA2M>

c.Explain parity bit method for error correction. [2]

Sol:

From Taifa Apu's Note:

Error Detection:

* 3rd parity bit error, (where, $n = 0, 1, \dots, n$)

Let, data bits = 4
 then, $2^3 \geq 4 + 3 + 1$
 so, parity bit $k = 3$

D_7	D_6	D_5	D_4	D_3	P_2	P_3
7	6	5	4	3	2	1

$2^0 = 1$
 $2^1 = 2$
 $2^2 = 4$
 $2^3 = 8$

$P_3 \rightarrow D_3 D_5 D_7$
 $P_2 \rightarrow D_3 D_6 D_7$
 $P_1 \rightarrow D_5 D_6 D_7$

Let, Data = 1011

then, 7-bit hamming code $\Rightarrow$

7	6	5	4	3	2	1
1	0	1	P_4	1	P_2	P_1

$\therefore P_1 = D_3 D_5 D_7 = 111$
 $\therefore$ Parity bit will be 1 (for even parity)

$\therefore P_2 = 101$; Parity bit $P_2 = 0$
 and $P_3 = 101$, Parity bit $P_3 = 0$
 $\therefore (P_1 P_2 P_3) = 000$
 So, Sender will send data as: 1010101

Let, $P_4 P_2 P_1$

0	0	0
1	0	0
2	0	1
3	0	1
4	1	0
5	1	0
6	1	1
7	1	1

$P_1 \Rightarrow 1, 3, 5, 7$
 $P_2 \Rightarrow 2, 3, 6, 7$
 $P_4 \Rightarrow 4, 5, 6, 7$

But, if the receiver gets data: 1110101

Receiver knows, $P_1 = D_3 D_5 D_7 = 111$, so, $P_1 = 1$
 $P_2 = D_3 D_6 D_7 = 111$, so, $P_2 = 1$
 $P_3 = D_5 D_6 D_7 = 111$, so, $P_3 = 1$

$\therefore (P_1 P_2 P_3) = 111$

so, here must be some errors. As P_2 and P_4 should be have even number of 1s.

Error Correcting:

Ex: If the 7-bit hamming code word received by a receiver is 1011011. Assuming the even parity state whether the received code word is correct or not. If wrong locate the bit having error.

$\Rightarrow$ Given,

D_7	D_6	D_5	P_4	D_3	P_2	P_1
1	0	1	1	0	1	1

$P_4 = D_5 D_6 D_7 = 1101 \therefore P_4 = 0$
 $P_2 = D_3 D_6 D_7 = 1001 \therefore P_2 = 0$
 $P_1 = D_3 D_5 D_7 = 1011 \therefore P_1 = 0$

So, $P_4 P_2 P_1 = (101)_5 = (5)_{10}$
 So, the 5th bit (D_5) is having error
 and D_5 should be = 0
 $\therefore$ Correct code is = (1001011)

Or you can follow that answer:

>> Explain Parity Bit Method for Error Detection

Error Detection in Computer Networks is a method to detect errors in the data introduced during transmission. Parity Check uses a parity bit to perform error detection.

Parity bit means nothing but an additional bit added to the data at the transmitter before transmitting the data. Before adding the parity bit, number of 1's or zeros is calculated in the data. Based on this calculation of data an extra bit is added to the

actual information / data. The addition of parity bit to the data will result in the change of data string size.

This means if we have an 8 bit data, then after adding a parity bit to the data binary string it will become a 9 bit binary data string.

Parity check is also called as “Vertical Redundancy Check (VRC)”.

There is two types of parity bits in error detection, they are

- Even parity
- Odd parity

Even Parity

- If the data has even number of 1's, the parity bit is 0. Ex: data is 10000001 -> parity bit 0
- Odd number of 1's, the parity bit is 1. Ex: data is 10010001 -> parity bit 1

Odd Parity

- If the data has odd number of 1's, the parity bit is 0. Ex: data is 10011101 -> parity bit 0
- Even number of 1's, the parity bit is 1. Ex: data is 10010101 -> parity bit 1

NOTE: The counting of data bits will include the parity bit also.

Evabe mone rakhte paro: Even এ 1 even হলে 0; Odd এ 1 odd হলে 0

Even -even=0 && Odd-odd=0

The circuit which adds a parity bit to the data at transmitter is called “Parity generator”. The parity bits are transmitted and they are checked at the receiver. If the parity bits sent

at the transmitter and the parity bits received at receiver are not equal then an error is detected. The circuit which checks the parity at receiver is called “Parity checker”.

Messages with even parity and odd parity

3 bit data			Message with even parity		Message with odd parity	
A	B	C	Message	Parity	Message	Parity
0	0	0	000	0	000	1
0	0	1	001	1	001	0
0	1	0	010	1	010	0
0	1	1	011	0	011	1
1	0	0	100	1	100	0
1	0	1	101	0	101	1
1	1	0	110	0	110	1
1	1	1	111	1	111	0

Extra videos: <https://www.youtube.com/watch?v=U7-h2hyM1Dc>

Extra source:

https://www.tutorialspoint.com/computer_logical_organization/error_codes.htm

d. Consider a bit string, 1011. Using a parity bit generator and a parity bit checker circuit, show how even parity bit method works in a communication system. [3]

Sol: *Not found, if anyone can solve it, please send me.*

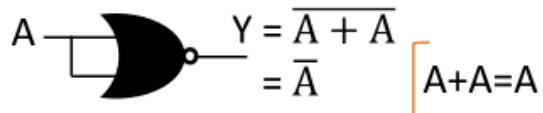
2

a. What do you understand by Universality of logic gate? Show that the NOR gate is a Universal gate. [4]

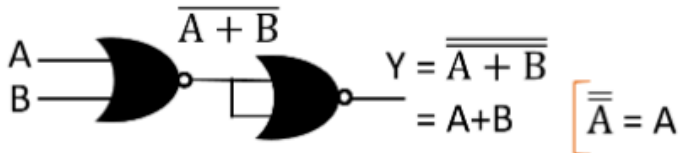
Sol:

A universal gate is a gate which can implement any Boolean function without need to use any other gate type. NOR & NAND Gate is called Universal Gates because all the other gates (NOT, AND, OR and NOR) can be created by using this gate.

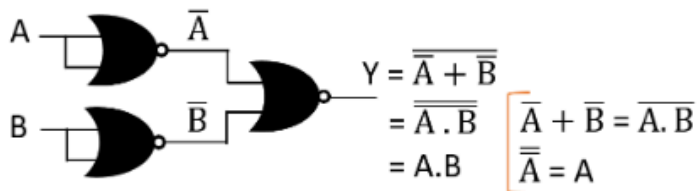
Implementation of NOT gate using only NOR gate:



Implementation of OR gate using only NOR gate:



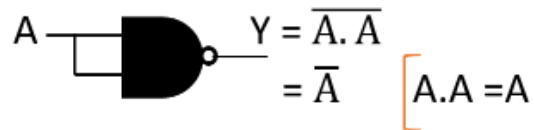
Implementation of AND gate using only NOR gate:



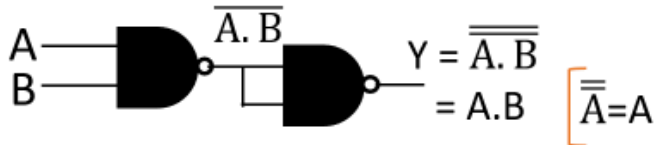
As we can see from the above discussion, All three basic gates can be implemented by only NOR Gate. Again we know that any gate or any circuit can be implemented by three basic gates. Since NOR Gate can implement any gate or any circuit with three basic gates (AND, OR, NOT), NOR Gate is called a universal Gate.

Extra. NAND gate universal;

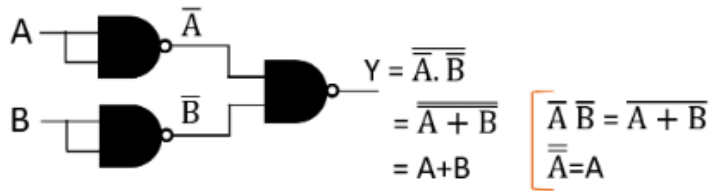
Implementation of NOT gate using only NAND gate



Implementation of AND gate using only NAND gate:



Implementation of OR gate using only NAND gate:



As we can see from the above discussion, All three basic gates can be implemented by only NAND Gate. Again we know that any gate or any circuit can be implemented by three basic gates. Since NAND Gate can implement any gate or any circuit with three basic gates (AND, OR, NOT), NAND Gate is called a universal Gate.

b) Prove DeMorgan's Theorem.[2]

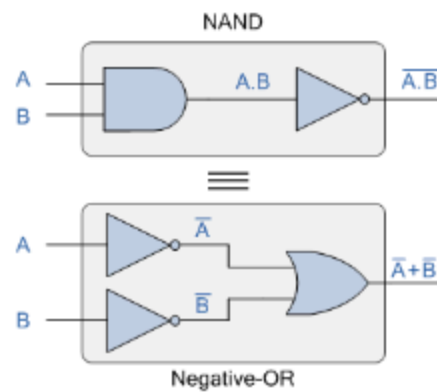
Sol:

Verifying DeMorgan's First Theorem using Truth Table

Inputs		Truth Table Outputs For Each Term				
B	A	A.B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A+B}$
0	0	0	1	1	1	1
0	1	0	1	0	1	1
1	0	0	1	1	0	1
1	1	1	0	0	0	0

We can also show that $\overline{A.B} = \overline{A+B}$ using logic gates as shown.

DeMorgan's First Law Implementation using Logic Gates



DeMorgan's Second Theorem

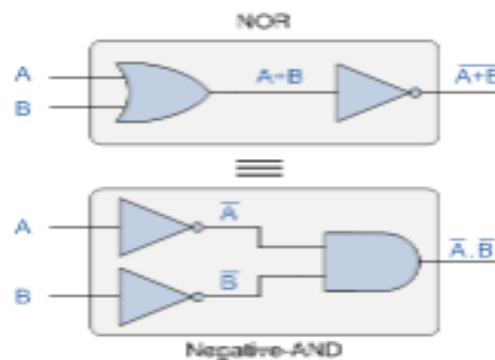
DeMorgan's Second theorem proves that when two (or more) input variables are OR'ed and negated, they are equivalent to the AND of the complements of the individual variables. Thus the equivalent of the NOR function is a negative-AND function proving that $\overline{A+B} = \overline{A}.\overline{B}$, and again we can show operation this using the following truth table.

Verifying DeMorgan's Second Theorem using Truth Table

Inputs		Truth Table Outputs For Each Term				
B	A	$A+B$	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	0	1	1	1	1
0	1	1	0	0	1	0
1	0	1	0	1	0	0
1	1	1	0	0	0	0

We can also show that $\overline{A+B} = \overline{A}.\overline{B}$ using the following logic gates example.

DeMorgan's Second Law Implementation using Logic Gates



c) What are different methods to write a logical expression from truth table?
Explain with an example.[3]

Sol:

SOP:- The two different methods are "Sum of Products" and "Product of Sums".

SOP:- The methods of logic-circuit simplification and design that we will study require the logic expression to be in a "SOP" form. Examples:

1. $ABC + \bar{A}BC$

2. $\bar{A}B + CD + EF + GH + HI$

Each of these sum-of-products expressions consists of two or more AND terms that are ORed together. Each AND term consists of one or more variable individually appearing in either complemented or uncomplemented form.

Example:-

A	B	C	Output
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

For instance, in fourth row in truth table for our two out of three logic systems, where

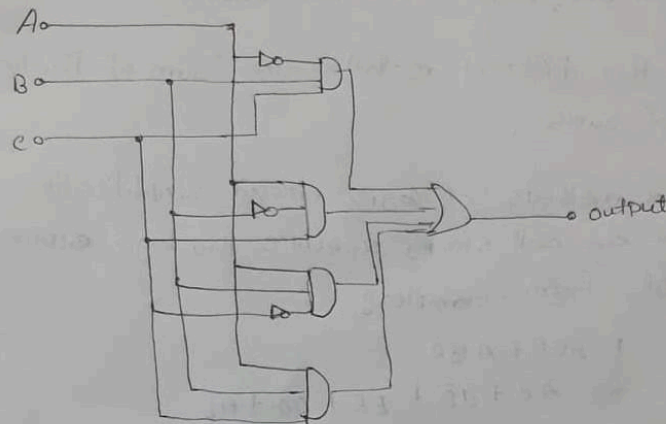
$A=0, B=1$ & $C=1$, the product term is, $\bar{A}BC$. Since the term would be a value of 1, iff $A=0, B=1, C=1$

The other rows of truth table have an output value of 1, so those rows also need Boolean product expressions to represent them:

$$\bar{A}BC, \bar{A}\bar{B}C, A\bar{B}\bar{C}, ABC$$

Finally we join the four boolean expressions together by addition, to create a single boolean expression describing the truth table, output is, $\bar{A}BC + \bar{A}\bar{B}C + A\bar{B}\bar{C} + ABC$

Logic circuit
Truth table :-



Pos:- Another form of logical expression called Product of Sums.
It consists of two or more OR terms that are ANDed together.
Each OR term consists of one or more variables in complemented or uncomplemented form. Such as, $(A+B+c)(A+c)$

Ex:-

A	B	C	output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

For instance, $A=0, B=0$ & $C=0$, the sum term would be $(A+B+c)$, since that term would have output 0 iff $A=0, B=0$ & $C=0$. $\therefore (\bar{A}\bar{B}\bar{c})$

The last sum term must be written as $(A+B+c)$ because, $A=1, B=1$ & $C=1$ produce output = 0.

The POS expression, is multiplicative combination of these two sum terms. $\text{output} = (A+B+c)(\bar{A}\bar{B}\bar{c})$

Logic circuit



d) Explain the K-Map method to simplify a logic circuit. [3]

Sol:

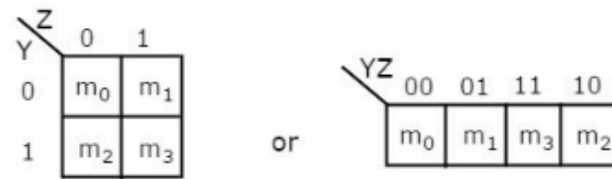
The K-map method of solving the logical expressions is referred to as the graphical technique of simplifying Boolean expressions

we have simplified the Boolean functions using Boolean postulates and theorems. It is a time consuming process and we have to re-write the simplified expressions after each step.

To overcome this difficulty, Karnaugh introduced a method for simplification of Boolean functions in an easy way. This method is known as Karnaugh map method or K-map method. It is a graphical method, which consists of 2^n cells for 'n' variables. The adjacent cells are differed only in single bit position.

2 Variable K-Map

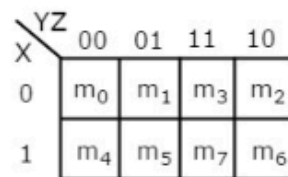
The number of cells in 2 variable K-map is four, since the number of variables is two. The following figure shows **2 variable K-Map**.



- There is only one possibility of grouping 4 adjacent min terms.
- The possible combinations of grouping 2 adjacent min terms are $\{(m_0, m_1), (m_2, m_3), (m_0, m_2) \text{ and } (m_1, m_3)\}$.

3 Variable K-Map

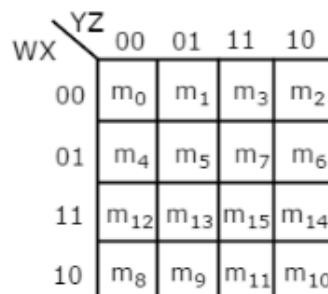
The number of cells in 3 variable K-map is eight, since the number of variables is three. The following figure shows **3 variable K-Map**.



- There is only one possibility of grouping 8 adjacent min terms.

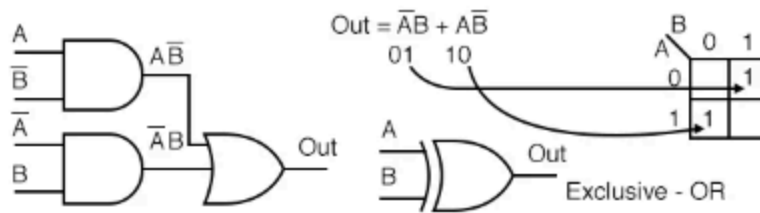
4 Variable K-Map

The number of cells in 4 variable K-map is sixteen, since the number of variables is four. The following figure shows **4 variable K-Map**.



- There is only one possibility of grouping 16 adjacent min terms.
- Let R_1, R_2, R_3 and R_4 represents the min terms of first row, second row, third row and fourth row respectively. Similarly, C_1, C_2, C_3 and C_4 represents the min terms of first column, second column, third column and fourth column respectively. The possible combinations of grouping 8 adjacent min terms are $\{(R_1, R_2), (R_2, R_3), (R_3, R_4), (R_4, R_1), (C_1, C_2), (C_2, C_3), (C_3, C_4), (C_4, C_1)\}$.
- If $w=0$, then 4 variable K-map becomes 3 variable K-map.

Example: Simplify the logic diagram below.



Solution:

- Write the Boolean expression for the original logic diagram shown above
- Transfer the product terms to the Karnaugh map.
- It is not possible to form groups.
- No simplification is possible; leave it as it is.

No logic simplification is possible for the above diagram. This sometimes happens. Neither the methods of Karnaugh maps nor Boolean algebra can simplify this logic further.

We show an Exclusive-OR schematic symbol above; however, this is not a logical simplification. It just makes a schematic diagram look nicer.

Since it is not possible to simplify the Exclusive-OR logic and it is widely used, it is provided by manufacturers as a basic integrated circuit (7486).

3.

a) Design a logic circuit with inputs P, Q, R so that output S is HIGH whenever P is '0' or whenever Q=R=1. [3]

Sol:

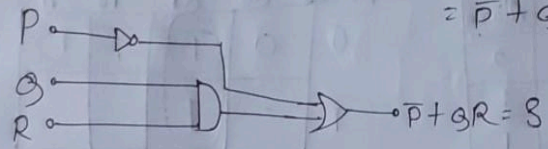
SI:-

P	Q	R	S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$P=0$

$Q=R=1$

$$\begin{aligned}
 & \therefore (\bar{P}\bar{Q}\bar{R}) + (\bar{P}\bar{Q}R) + (\bar{P}Q\bar{R}) + (\bar{P}QR) + (PQR) \\
 & \Rightarrow \bar{P}\bar{Q}(\bar{R}+R) + \bar{P}Q(\bar{R}+R) + PQR \\
 & \Rightarrow \bar{P}\bar{Q}(1) + \bar{P}Q(1) + PQR \\
 & \Rightarrow \bar{P}\bar{Q} + \bar{P}Q + PQR \\
 & \Rightarrow \bar{P}(\bar{Q}+Q) + PQR \\
 & \Rightarrow \bar{P}\bar{Q} + Q(\bar{P}+PR) \\
 & \Rightarrow \bar{P}\bar{Q} + Q(\bar{P}+R) \\
 & \Rightarrow \bar{P}\bar{Q} + \bar{P}Q + QR \\
 & \Rightarrow \bar{P}(\bar{Q}+Q) + QR \\
 & = \bar{P} + QR
 \end{aligned}$$

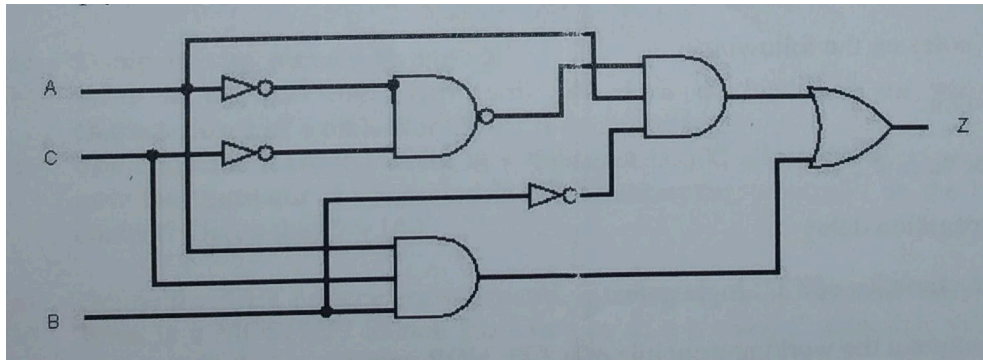


SOP
solved by
K-map

	$\bar{Q}\bar{R}$	$\bar{Q}R$	QR	$Q\bar{R}$
$\bar{P}$	1	1	1	1
P			1	1

$$\therefore \bar{P}\bar{Q} + QR = S$$

b) Simply the logic circuit shown below: [2]



Sol: Simple logic Circuit. Try yourself!

c) Design a Full Adder [4]

Sol:

Inputs			Outputs	
A	B	C _{in}	Sum	Carry
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

For sum

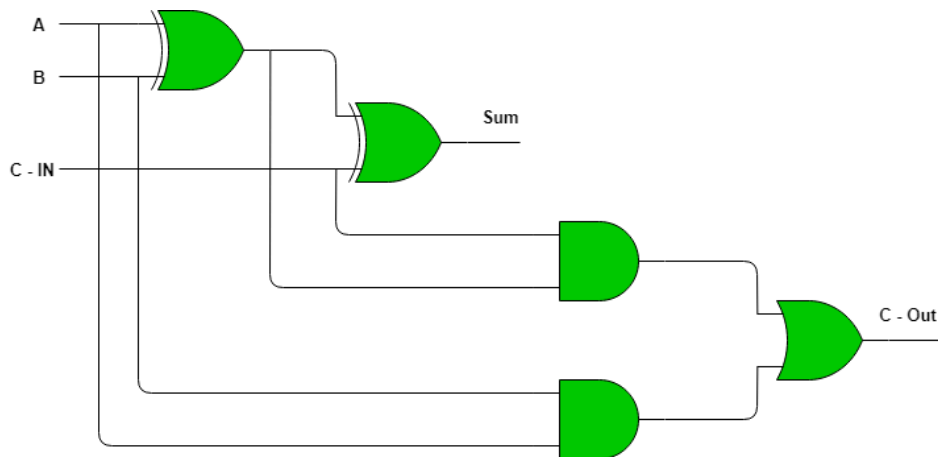
A \ BC	00	01	11	10
0		1		1
1	1		1	

$$\begin{aligned}
 \therefore \text{sum} &= \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC \\
 &= \bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC) \\
 &= \bar{A}(B \oplus C) + A(\overline{B \oplus C}) \quad (\because \bar{x}y + x\bar{y} = x \oplus y) \\
 &= A \oplus B \oplus C
 \end{aligned}$$

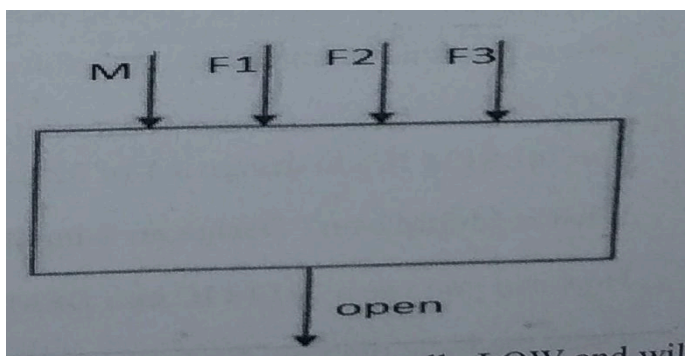
For carry

A \ BC	00	01	11	10
0			1	
1		1	1	1

$$\text{Carry} = AB + BC + AC$$



d)



Design a logic circuit that controls the elevator door in a three-story building. The circuit has four inputs. M is a logic signal that indicates when the elevator is moving (M=1) or stopped (M=0). F₁, F₂ and F₃ are floor indicator signals that are normally LOW, and they go HIGH only when the elevator is positioned at the level of that particular floor. For example, when the elevator is lined up level with the second floor, F₂=1 and F₁=F₃=0. The circuit output is the OPEN signal, which is normally LOW and will go to HIGH when the elevator door is to be opened. Find the simplified design and draw the circuit diagram of the elevator circuit. Because the elevator cannot be lined up with more than one floor at a time, the cases when more than one floor inputs are high can be treated as don't care condition. When the elevator reaches at one floor and stops, the door opens.[3]

Sol:

The answer provided below has been developed in a clear step-by-step manner.

Step: 1

Solution:

Here is the truth table:

Input				Output
M	F ₁	F ₂	F ₃	Open
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	X
0	1	0	0	1
0	1	0	1	X
0	1	1	0	X
0	1	1	1	X
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	X
1	1	0	0	0
1	1	0	1	X
1	1	1	0	X
1	1	1	1	x

Step: 2

Here is the Karnaugh map:

K-map for Open:

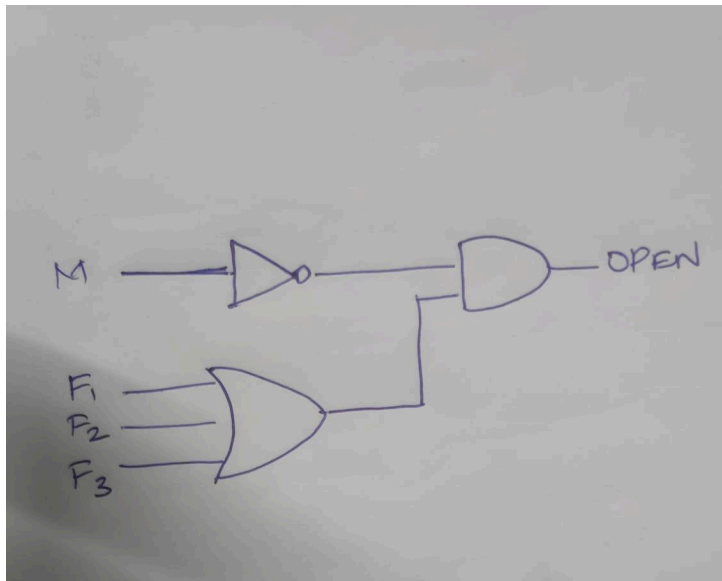
$MF_1 \backslash F_2F_3$	00	01	11	10
00		1	x	1
01	1	x	x	x
11		x	x	x
10			x	

$$\text{Open} = M_1 F_1 + M_1 F_2 + M_1 F_3$$

$$\text{Therefore, Open} = M_1 (F_1 + F_2 + F_3)$$

Step: 3

Here is the logic circuit diagram for the given problem:



Answer:

Therefore, the logic circuit is designed that controls the elevator door in the three storied building with the given inputs and requirements.

a) What is a Flip Flop? Briefly explain the function of a JK-Flip Flop with a suitable circuit diagram. [3]

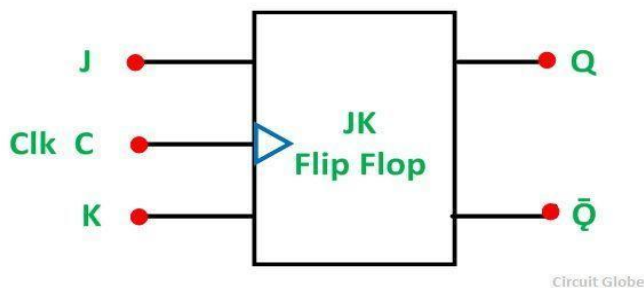
Sol:

A flip-flop is an electronic circuit with two stable states that can be used to store binary data.

The **JK Flip Flop** is the most widely used flip flop. It is considered to be a universal flip-flop circuit. The sequential operation of the JK Flip Flop is the same as for the RS flip-flop with the same **SET** and **RESET** input.

The difference is that the JK Flip Flop does not have the invalid input states of the RS Latch (when S and R are both 1)

The basic **symbol** of the JK Flip Flop is shown below:

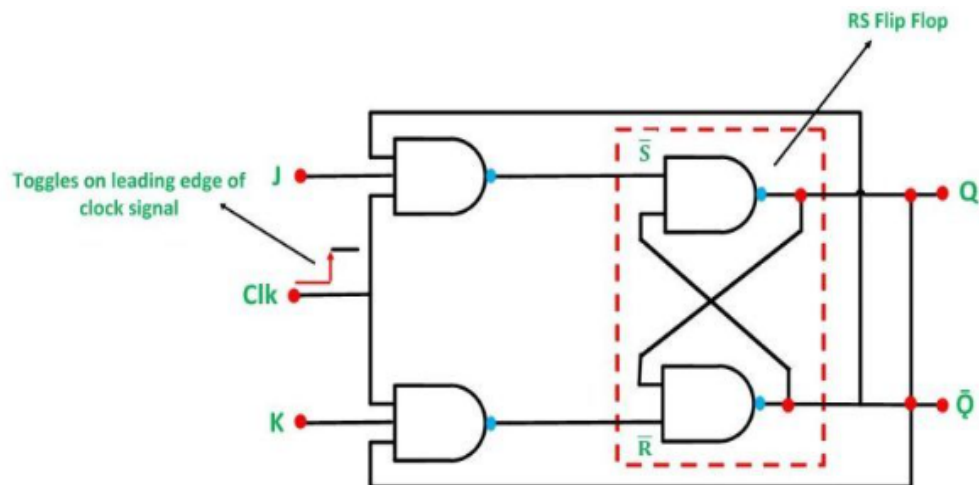


The j-k Flip flop has four possible input combinations because of the addition of the clocked input. The four inputs are “logic 1”, “logic 0”, “No change” and “Toggle”.

Truth Table:

Clock	J	K	Q_{n+1}	State
0	x	x	Q_n	
1	0	0	Q_n	Hold
1	0	1	0	Reset
1	1	1	1	Set
1	1	1	$\bar{Q}_n$	Toggle

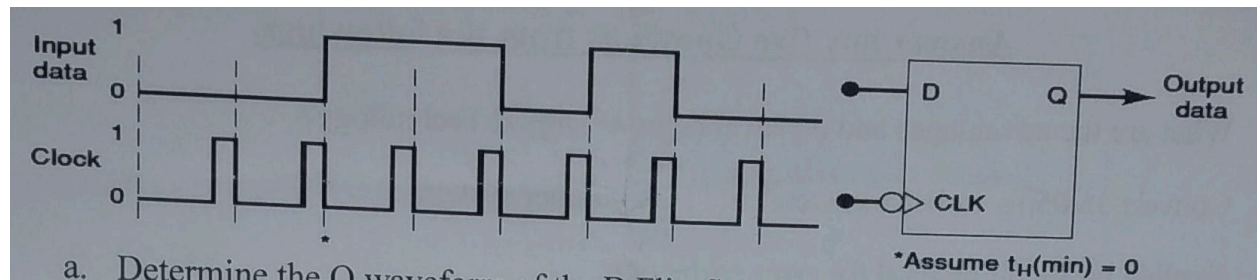
Diagram:



b) What is the limitation of S-R FlipFlop? Explain how the limitation can be resolved. [3]

Sol: See 6th batch final 3(b)

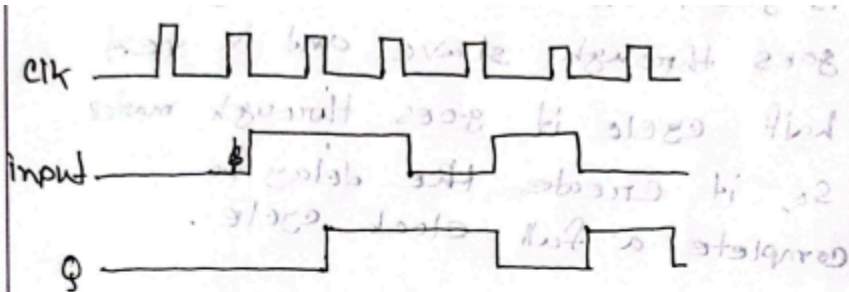
c. A, D FlipFlop is sometimes used to delay a binary waveform so that the binary information appears at the output a certain amount of time after it appears at the D input. [6]



Sol:

a) Determine the Q waveform of the D Flip-Flop.

$\Rightarrow$ Since the FF has $t_H = 0$, the FF will respond to the value present on the D input just prior to the NGT of the clock.



b) How can a delay of two clock periods can be obtained?

$\Rightarrow$ Connect Q to the D input of a second FF and connect the clock signal to the second FF. The output of the second FF will be delayed by 2 clock periods from the input data.

O/P of d flip flop depends on its ~~pre~~ previous state, hence there is a time delay of one cycle. Flip Flop has generally master slave combination. In one clock cycle either master is active or slave (both are opposite triggered). So when even a input

Scanned with CamScanner

is given in first half of cycle it goes through slave and in next half cycle it goes through master. So, it creates the delay to complete a full clock cycle.

5

a) What is the limitation of a parallel adder? Using 74LS83 ICs, draw a 12-bit parallel Adder. [3]

Sol: See 6th batch final 4(a)

b) Explain how 2's complement system can facilitate arithmetic operations in digital computing. [3]

***Sol:** See 6th batch final 4(b)*

c) Draw a parallel adder/subtractor using 2's-complement system. Using your designed circuit, explain steps to subtract 4 from -3. Consider that your designed circuit receives only positive input.[6]

***Sol:** See 6th batch final 4(c)*

6.

a) What is Counter? What are the differences between Synchronous and Ripple counter?[2]

Sol:

Counters are sequential circuits that count the number of pulses that can be either in binary code or BCD form

Differences

S.NO Synchronous Counter

Asynchronous Counter/Ripple counter

1. In synchronous counter, all flip flops are triggered with same clock simultaneously.

In asynchronous counter, different flip flops are triggered with different clock, not simultaneously.

- | | | |
|----|--|--|
| 2. | Synchronous Counter is faster than asynchronous counter in operation. | Asynchronous Counter is slower than synchronous counter in operation. |
| 3. | Synchronous Counter does not produce any decoding errors. | Asynchronous Counter produces decoding error. |
| 4. | Synchronous Counter is also called Parallel Counter. | Asynchronous Counter is also called Serial Counter. |
| 5. | Synchronous Counter designing as well implementation are complex due to increasing the number of states. | Asynchronous Counter designing as well as implementation is very easy. |
| 6. | Synchronous Counter will operate in any desired count sequence. | Asynchronous Counter will operate only in fixed count sequence (UP/DOWN). |
| 7. | Synchronous Counter examples are: Ring counter , Johnson counter . | Asynchronous Counter examples are: Ripple UP counter, Ripple DOWN counter. |

- In asynchronous counter, there is high propagation delay.

Sol:

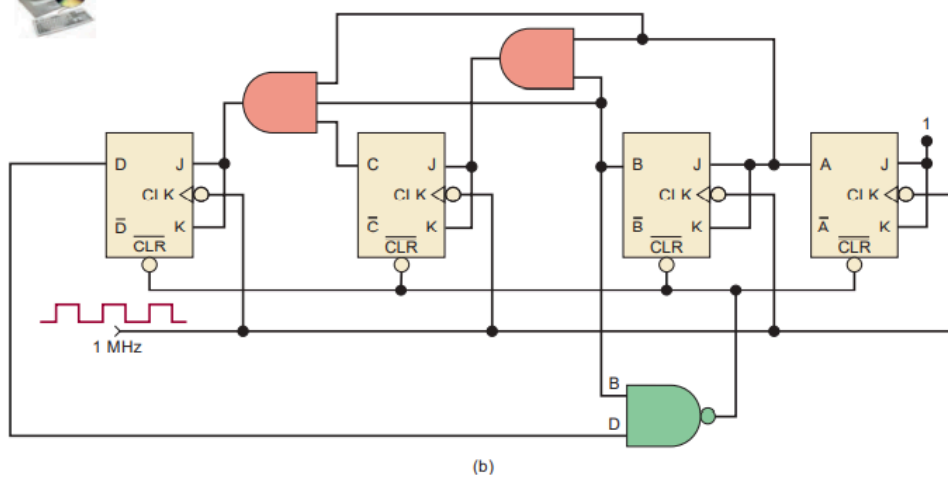


FIGURE 7-8 (a) MOD-14 ripple counter; (b) MOD-10 (decade) ripple counter.

c) Show how to wire the 74293 IC as a MOD-6 counter.[2]

Sol: Not found, if anyone can solve it, please send me.

d) Explain the operation of a Synchronous up/down MOD-8 counter with an appropriate diagram. [4]

Sol:

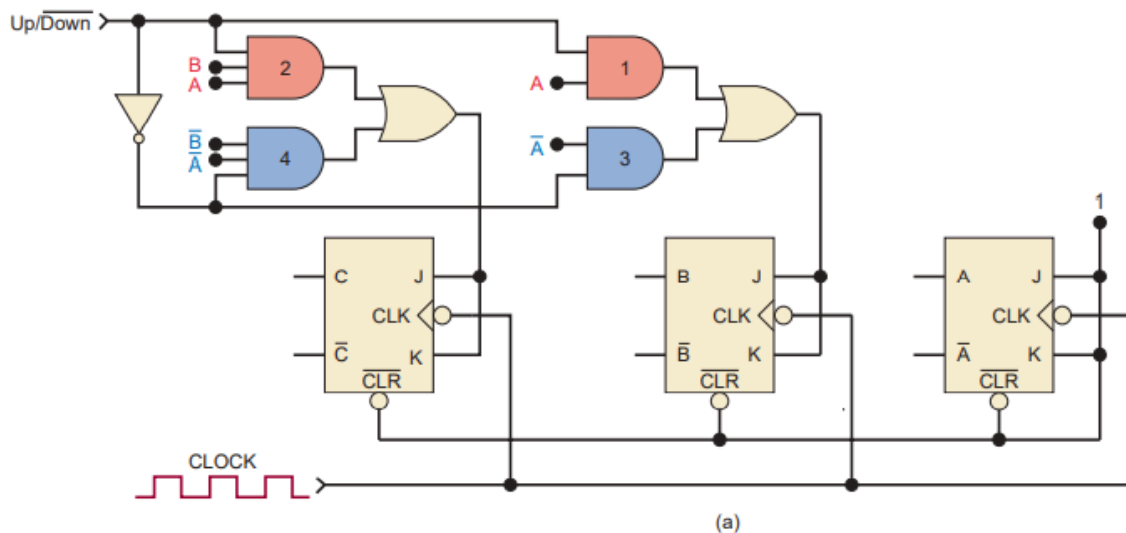


Figure 7-11(a) shows how to form a parallel up/down counter. The control input controls whether the normal FF outputs or the inverted FF outputs are fed to the J and K inputs of the successive FFs. When is held HIGH, AND gates 1 and 2 are enabled while AND gates 3 and 4 are disabled (note the inverter). This allows the A and B outputs through gates 1 and 2 to the J and K inputs of FFs B and C. When is held LOW, AND gates 1 and 2 are disabled while AND gates 3 and 4 are enabled. This allows the inverted A and B outputs through gates 3 and 4 into the J and K inputs of FFs B and C

7.

a) Write short notes on the followings:[4]

I) Fan-Out

II) Noise Immunity

III) Propagation delay

IV) Characteristics of TTL logic gate.

Sol:

i)Fan Out

The fan-out is defined as the maximum number of inputs (load) that can be connected to the output of a gate without degrading the normal operation.

Fan In(Extra)

The fan-in is defined as the maximum number of inputs that a logic gate can accept.

ii)Noise Immunity

Noise immunity is the ability of a device or system to suppress external noise and to function – without deteriorating in performance – in the presence of noise.

iii)Propagation Delay

Propagation Delay is the amount of time it takes for a signal to travel from a source to a destination.

IV) Characteristics of TTL logic gate.

Fan In, and Fan-Out

Propagation delay

Power dissipation

Noise Margin

Advantages:

- › Power dissipation is less compared to DTL and RTL
- › Less expensive
- › Noise Margin and Fan out are better

Disadvantages:

- › It cannot be used in high performance processors
- › It is not used in high end electronic devices

Applications:

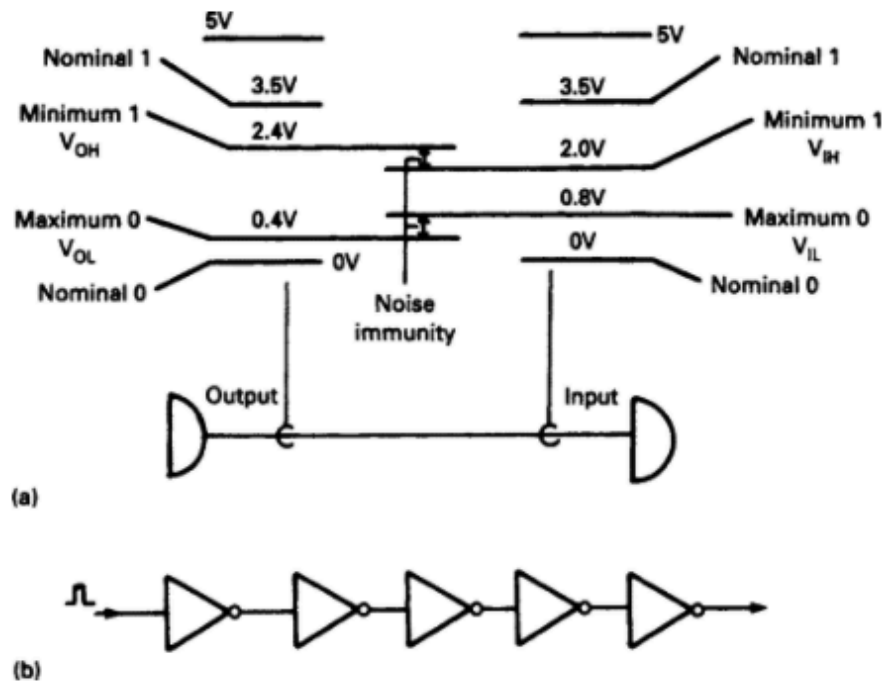
- › It is used in the processors in computers
- › It is used in controller circuits
- › Used in remote and light controller
- › Used in microprocessor and microcontrollers

What is called noise immunity of a logic circuit? [4]

Sol:

The noise immunity of a logic circuit refers to the circuit's ability to tolerate noise by causing spurious (undesirable) charges/signals in the output voltage

Electrical interference may cause 1 signals to appear as 0 signals and vice versa. The ability of a gate to reject noise is called its **noise immunity**. Defining noise immunity is more complex than it might at first appear, but the method usually adopted is that shown on *Figure 14.12(a)*. The voltages given are those for a TTL gate which has a nominal 1 V of 4.5 V and a nominal 0 V of 0 V.



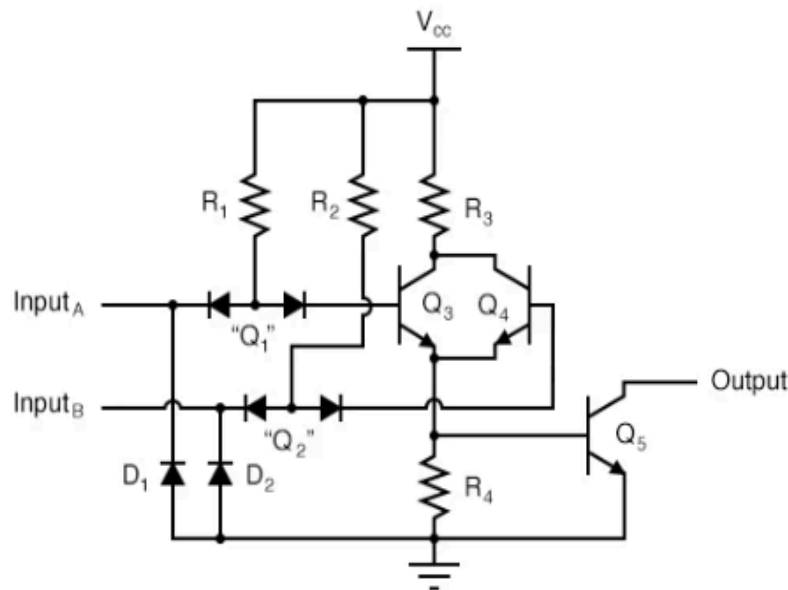
FEEDBACK

Source: [Noise Immunity - an overview | ScienceDirect Topics](#)

b) Design and explain the working principle of a TTL NOR gate[4]

Sol:

Transistors Q1 and Q2 are both arranged in the same manner that we've seen for transistor Q1 in all the other TTL circuits. Rather than functioning as amplifiers, Q1 and Q2 are both being used as two-diode "steering" networks. We may replace Q1 and Q2 with diode sets to help illustrate:



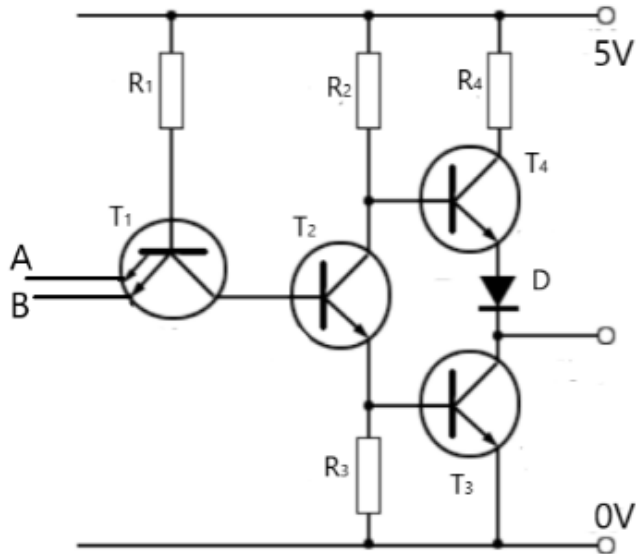
If input A is left floating (or connected to V_{CC}), current will go through the base of transistor Q_3 , saturating it. If input A is grounded, that current is diverted away from Q_3 's base through the left steering diode of "Q1," thus forcing Q_3 into cutoff. The same can be said for input B and transistor Q_4 : the logic level of input B determines Q_4 's conduction: either saturated or cutoff.

Notice how transistors Q_3 and Q_4 are paralleled at their collector and emitter terminals. In essence, these two transistors are acting as paralleled switches, allowing current through resistors R_3 and R_4 according to the logic levels of inputs A and B. If *any* input is at a "high" (1) level, then at least one of the two transistors (Q_3 and/or Q_4) will be saturated, allowing current through resistors R_3 and R_4 , and turning on the final output transistor Q_5 for a "low" (0) logic level output. The only way the output of this circuit can ever assume a "high" (1) state is if *both* Q_3 and Q_4 are cut off, which means *both* inputs would have to be grounded, or "low" (0).

Extra:

##Design and explain the working principle of a TTL NAND gate

Sol:



The above diagram is the circuit diagram of a TTL NAND gate.

From the diagram, we shall explain the working. Now, as seen, the transistor T_1 has two emitters to allow two inputs into the transistor. Now, as connected the base voltage will be at 5V. if both inputs are logic 1 (usually means about 5V too), the potential difference across base and emitter would be zero or nearly. Hence, no current will flow and the transistor is turned off. So, the collector voltage would also be equal to about 5V.

Hence, this potential can drive current through the emitter of the transistor T_2 . This then will allow the collector voltage of the transistor T_2 to fall.

Now due to the current flowing through the emitter, there would be a voltage drop across the resistor R_3 . The

desired voltage drop would be about $0.7V$. As seen, this is the input of the transistor T_3 . Hence, the transistor is turned on. Due to saturation, the collector voltage will fall to about $0.2V$ which is a logic 0. A

For the transistor T_4 , observe that the emitter voltage is made up of the entire voltage of the transistor T_3 plus the voltage drop across the diode D about $0.7V$. Hence the emitter potential would be $0.7 + 0.2 = 0.9V$.

Now the base voltage of the transistor T_4 , would be the voltage across the base-emitter of T_3 and the voltage of the entire transistor (i.e.) voltage across emitter-collector. This would also be equal to about $0.9V$. Hence

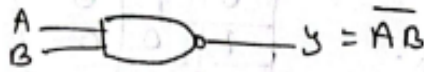
the emitter voltage and the collector voltage are equal. So the transistor T_4 will be turned off too. So the output

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

c) Draw the circuit diagram of a CMOS NAND gate. [2]

Sol:

gate:



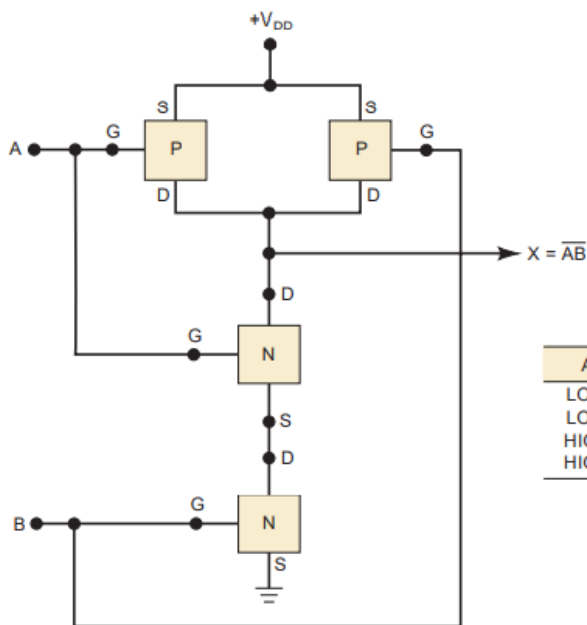
A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

for AND(.) operation;

P-MOS = Parallel

N-MOS = series

A CMOS NAND gate formed by adding a parallel P-channel MOSFET and a series N-channel MOSFET to the basic INVERTER.



##Extra: CMOS NOR gate? Or the working principle of a CMOS NOR gate?

Sol:

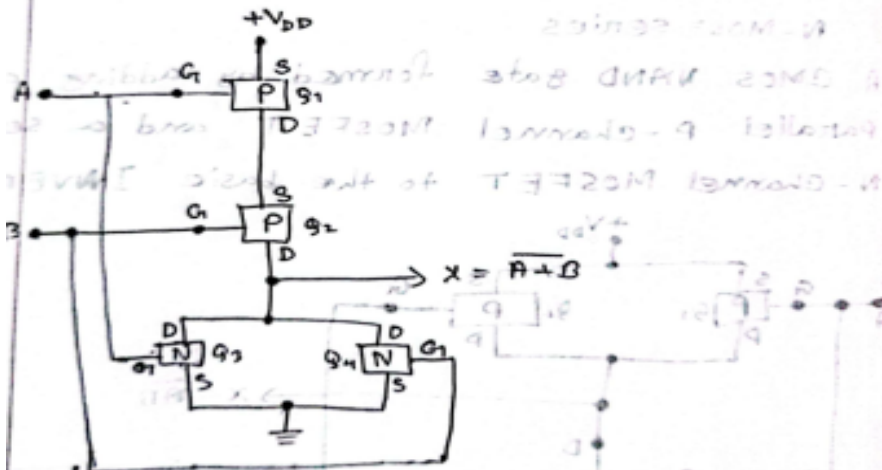
CMOS NOR gate:



For OR (+) operation:

PMOS = Series
NMOS = Parallel

A	B	A+B
0	0	1
0	1	1
1	0	1
1	1	0



Input		P-MOS		N-MOS		Output
A	B	Q1	Q2	Q3	Q4	$A+B$
0	0	1	1	0	0	1
0	1	1	0	0	1	1
1	0	0	1	1	0	1
1	1	0	0	1	1	0

For a better picture:

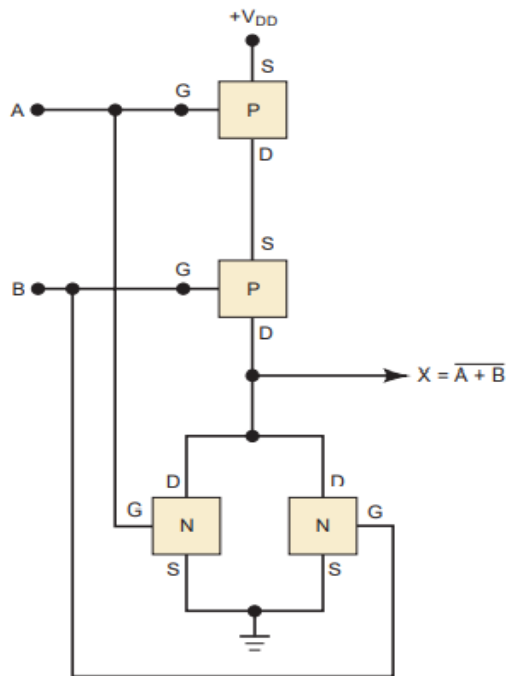


fig: CMOS NOR gate

d) A certain TTL IC output is rated at $I_{OH}(\text{max}) = 800 \mu\text{A}$ and $I_{OL} = 48 \text{ mA}$. Express the IC's fan-out in terms of unit loads. [consider 1 UL = $40 \mu\text{A}$ in the HIGH state and 1.6 mA in the LOW state] [2]

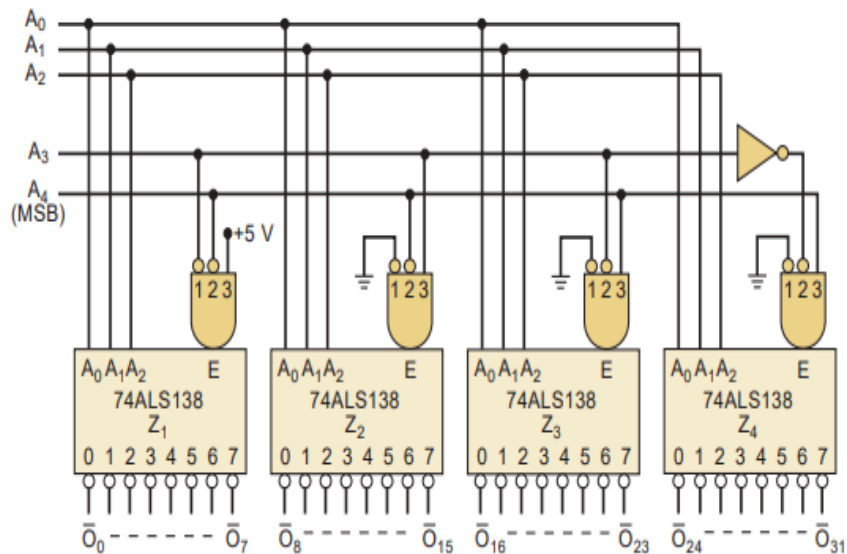
Sol: Not found, if anyone can solve it, please send me.

8

a) Use 74LS138 ICs to design a 1 of 32 decoder. [4]

Sol:

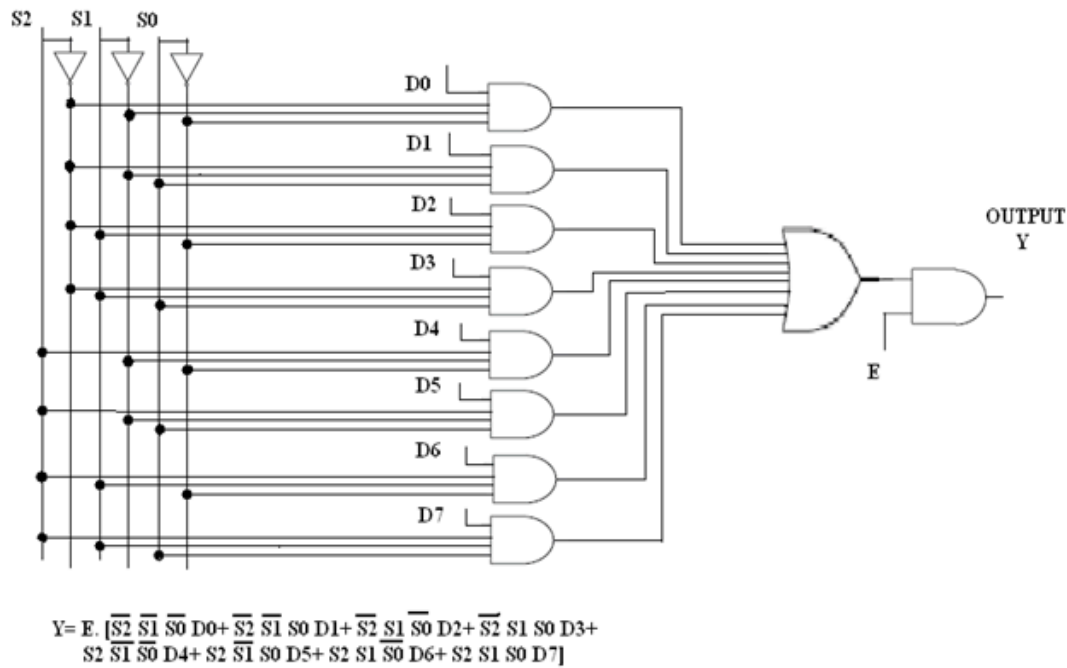
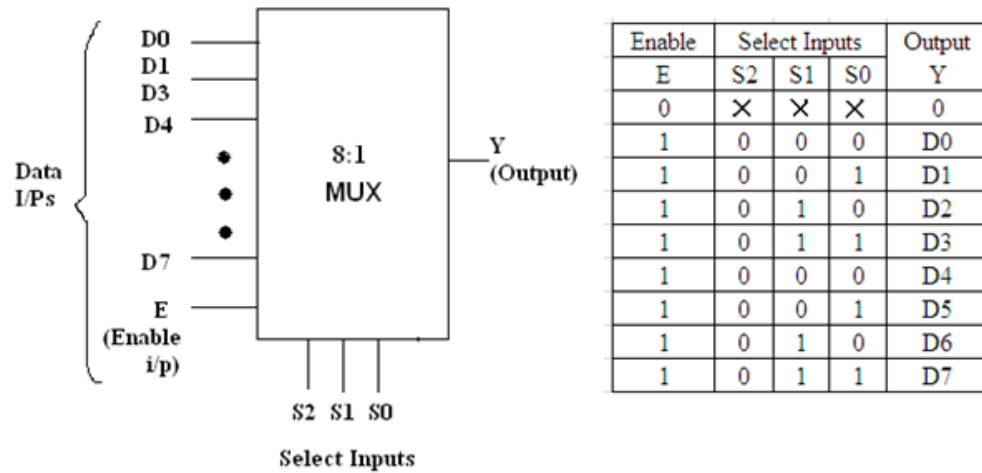
FIGURE 9-4 Four 74ALS138s forming a 1-of-32 decoder.



b) What is Multiplexer? Design an 8-input MUX and explain its operation. [4]

Sol:

Multiplexer means many into one. A multiplexer is a circuit used to select and route any one of the several input signals to a single output. A simple example of a non-electronic circuit of a multiplexer is a single-pole multi-position switch



c) How can you use a 74138 IC as a DEMUX.[2]

Sol:

Demultiplexer 74138 -

- IC 74138 is a 1:8 demultiplexer IC
- Also it is a IC of 3:8 decoder.

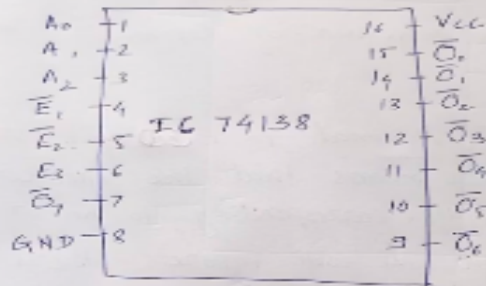


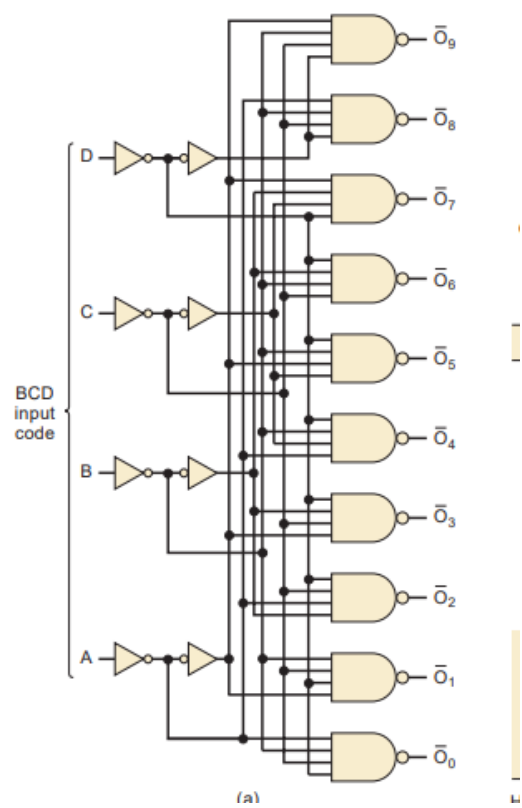
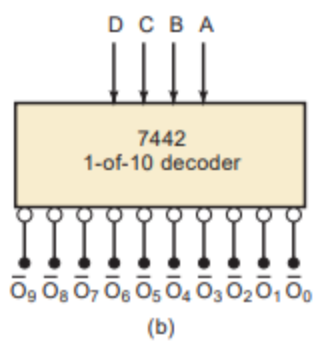
fig Pin diagram of IC 74138

- A_0 to A_2 are select lines
- $\overline{E}_1$, $\overline{E}_2$ are enable active low inputs & E_3 high enable input.
- We have to make $\overline{E}_1 = \overline{E}_2 = 0$ and $E_3 = 1$ in enable the IC.
- $\overline{O}_0$ to $\overline{O}_7$ are active low outputs.

d) Draw the logic diagram for the 7442 BCD to decimal decoder. [2]

Sol:

Each output goes LOW only when its corresponding BCD input is applied. For example, will go LOW only when inputs ; will go LOW only when . For input combinations that are invalid for BCD, none of the outputs will be activated. This decoder can also be referred to as a 4-to-10 decoder or a 1-of-10 decoder



5th batch(Mid)

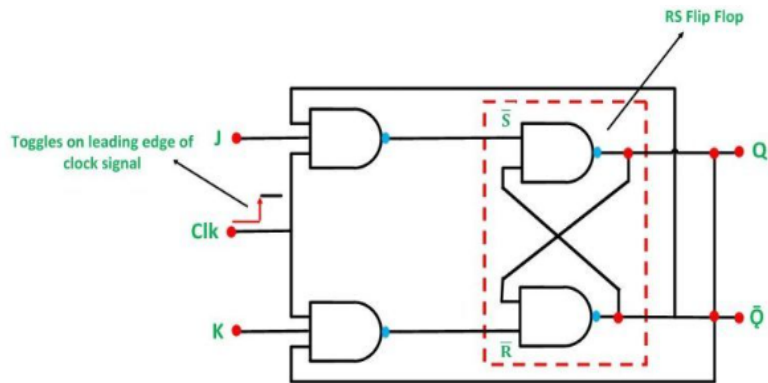
Q. Explain how a NAND latch circuit can be used to store a bit
• Draw the circuit diagrams and truth tables of JK FF (Flip Flop), T- FF, and D- FF.[5]

Sol:

The NAND gate latch or simply latch is a basic FF. The two NAND gates are cross-coupled • The inputs are set and clear (reset) • The inputs are active low, that is, the output will change when the input is pulsed low. y to end up in the $Q = 0$ state.

JK FF (Flip Flop)

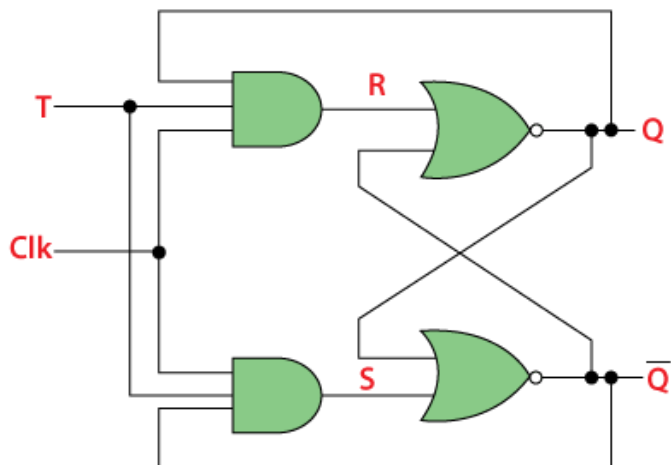
Clock	J	K	Q_{n+1}	State
0	x	x	Q_n	
1	0	0	Q_n	Hold
1	0	1	0	Reset
1	1	1	1	Set
1	1	1	$\bar{Q}_n$	Toggle



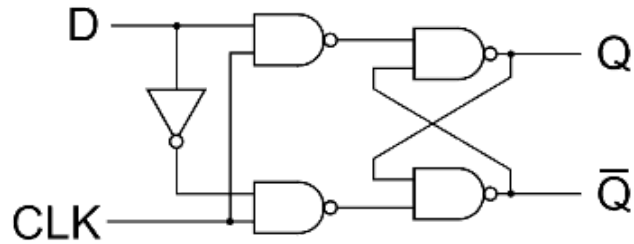
T- FF:

Truth Table of T Flip Flop

	Previous		Next	
T	Q	Q'	Q	Q'
0	0	1	0	1
0	1	0	1	0
1	0	1	1	0
1	1	0	0	1



D-FF:



Truth Table:

Clock	D	Q	Q'
↓ » 0	0	0	1
↑ » 1	0	0	1
↓ » 0	1	0	1
↑ » 1	1	1	0

Better inform.

Link: <https://www.electronicsforu.com/technology-trends/learn-electronics/flip-flop-rs-jk-t-d>

Q. What is the limitation of a JK FF? Draw a circuit diagram and it's truth table, which will eliminate its limitation.[5]

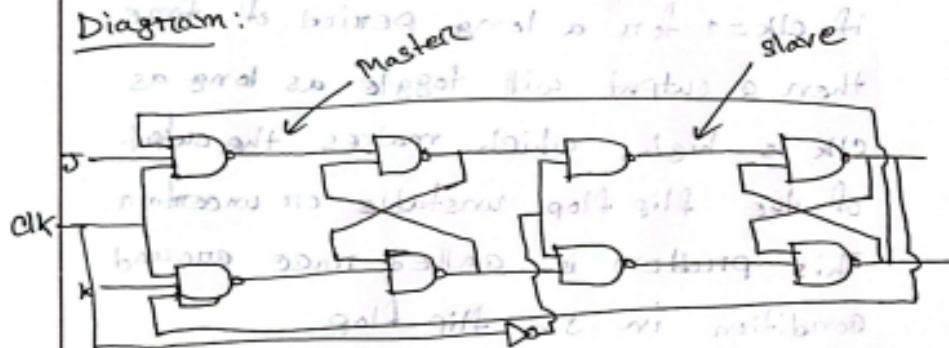
Sol:

⇒ In the J-K Flip-Flop when both J and K inputs are at logic "1" at the same time and the clock input is pulsed HIGH, the circuit toggle from its SET state to a RESET or visa versa. When both the terminals are HIGH the JK flip-flop acts as a T type toggle flip-flop.

JK flip-flop has a drawback of timing problem known as "RACE". The condition of RACE arises if the output Q changes its state before the timing pulse of the clock input has time to go in OFF state. Thus a much-improved flip-flop named Master-Slave JK Flip Flop was developed. This eliminates all the timing problems by using two SR flip-flop connected

in series. One is for the "MASTER" circuit which triggers on the leading edge of the clock pulse. The other is called the "SLAVE" circuit which triggers when the clock pulse is at the falling edge.

Diagram:



Truth Table:

clk	Inputs		Outputs		Remark
	S	R	Q_n	$\bar{Q}_n$	
X	0	0	Q_n	$\bar{Q}_n$	No change
$\square$ (1)	0	0	Q_n	$\bar{Q}_n$	No change
$\square$ (1)	0	1	0	1	Reset
$\square$ (1)	1	0	1	0	Set
$\square$ (1)	1	1	$\bar{Q}_n$	Q_n	Toggle

3. Design a Full Adder. [5]

Sol: See 6th batch final 2(c)

4. Considering a 74LS283 IC design a circuit that will perform subtraction operation. Using your designed circuit subtracts 4 from -3. Show that you have got the desired result from the process. Consider that your desired circuit receives only positive input.[5]

Sol: See 6th batch final 4(c)

6th batch(Mid)

Q. What are the advantages and disadvantages of Digital Systems?[3]

Sol: See 5th batch final 1(a)

Q. Convert 249.63(10) to BCD, octal, binary, and hex. [10]

Sol: See 5th batch final 1(b)

Q. Define the universality of the logic gate and prove it. [10]

Sol: See 5th batch final 2(a)

Q. Write short notes on the parity bit method.[7]

Sol: See 5th batch final 1(c)

Q. Write down short notes on the followings:

- i) Characteristics of TTL logic gate
- ii) Fan-Out
- iii) Noise Immunity
- iV) Propagation delay

Sol: See 5th (final 7(a))

Q. Design and explain the working principle of TTL NAND gate[2]

Sol: See 5th final 7(b)

Q. Draw the circuit diagram of a CMOS NOR gate. [1]

Sol: See 5th final 7(c)

Q. Draw the logic diagram and truth table of 7442 IC.[2]

Sol: *Not found, if anyone can solve it, please send me.*

Q. Draw the block diagram of a 16-input multiplexer using two 74151 IC.[2]

Sol: See 6th batch 8(a)

Q. Explain how you would use a 74LS138 IC as a DEMUX[1]

Sol: See 6th batch 8(c)

7th batch(Mid)

Q. Draw a 12-bit Parallel Adder with appropriate ICs.[2]

Sol: See 5th batch final 5(a)

Q. Compare TTL and CMOS devices' properties. .[2]

Sol:

1. CMOS components are generally **more expensive** when compared to the TTL components. But on system-level, CMOS chips are less expensive as these are smaller in size as compared to the TTL chips.
2. There are **propagation delays** present in both. On average, the propagation delays of TTL are usually 10nS whereas the propagation delays for the CMOS lies between 20 to 50 nS
3. CMOS has **longer rise and fall times** thus digital signals are simpler and less expensive with the CMOS chips.
4. There is a substantial difference in the voltage level range for both. For TTL it is 4.75 V to 5.25 V while for CMOS it ranges between 0 to $1/3 V_{DD}$ at a low level and $2/3 V_{DD}$ to V_{DD} at high levels.
5. CMOS technology is **more economical** and preferred more as compared to the TTL logic.
6. The current requirements of the CMOS are low and thus **power consumption is limited**. Therefore it is easier for the circuits to be designed with the best power management.
7. The **electromagnetic disruptions** CMOS components are more sensitive as compared to the TTL components
8. CMOS has one other advantage over the TTL that it has allowed **lower noise** during the transmission
9. The number of standard loads that could be connected to the output of the gate under the normal operation that is the fan-out is 10 for TTL whereas it is 50 for the CMOS.
10. The number of standard inputs that can be connected to the gate is the fan in, which is approximately 12-14 for the TTL and for the CMOS it is 10 only.
11. CMOS circuits have better **noise immunity** than the TTL circuits

Q. Draw a TTL NOR Gate.[2]

Sol: See 5th batch final 7(b)

Q. Explain the working principle of a CMOS NOR gate.[4]

Sol: See 5th batch final 7(c)

Q. Draw the circuit diagrams and truth tables of JK-FF, T- FF, and D- FF. [3]

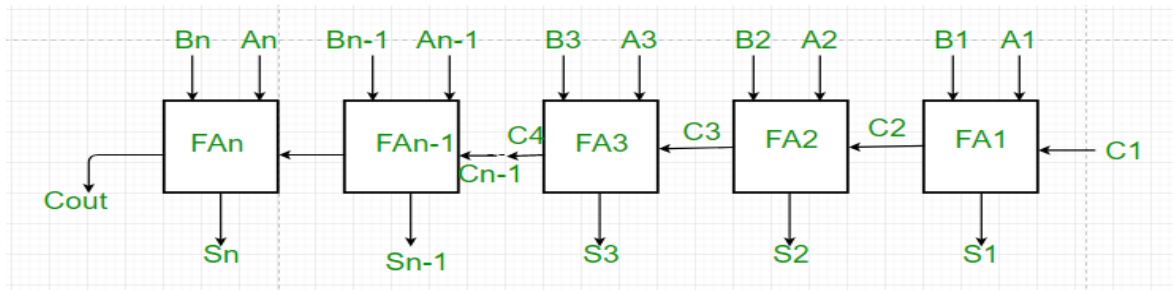
Sol: See 5th batch Mid

Q. What is the limitation of a JK FF? Draw a circuit diagram, its truth table, and timing diagram which will eliminate its limitation.[4]

Sol: See 5th batch Mid ques.

Q. Draw a complete Parallel Adder with a diagram.[2]

Sol:



Q. What is the limitation of parallel adder? How can you resolve the limitation?[1]

Sol: See 5th batch Mid ques.

Extra Ques:

- Why do computers use binary numbers/systems?

Sol:

- Binary isn't really a language. It has no grammar. Binary is simply a counting system, just like the familiar decimal counting system that we all use routinely. The binary counting system uses only two digits: 0 and 1, whereas the octal counting system uses 8, the decimal counting system uses 10, and the hexadecimal system uses 16.

- In many, but not all or even most cases, it makes sense to use binary notation because it maps very closely to the electronics that make up the computer, which are binary in nature. The binary aspect means that they input/read as well as output/write one of only two states, and we designate those states as 0 and 1 for convenience as a way to talk about them, perform arithmetic using them, etc. Some aspect of the computing hardware, while still binary in nature, are not necessarily quantities *per se* but rather just a combination of states, such as a register which contents describe the state of some peripheral chip, or even the FLAG/STATUS register of the CPU. Representing those kinds of binary data is almost misleading in any other radix, since the individual bits (binary digits) are concealed, which therefore conceals the information that they represent. Mostly, though, we look at data as quantitative numbers where the bits are ordered to give them weighted values that allow us to assign order to the binary words.

GOOD LUCK!!