

DIGITAL ELECTRONICS

INTERVIEW QUESTIONS



Digital Electronics Question & Answers

Digital Electronics (Q&A)

Binary Number Systems

Q1) Define: (a) bit (b) nibble (c) byte (d) word

Answer:

- a. Bit: Binary digit (Either logic-1 or logic-0)
- b. Nibble: 4-bits together is called a nibble
- c. Byte: 8 bits or 2 nibbles
- d. Word: 16 bits or 2 bytes

Q2) What is weighted code? Give example.

Answer:

The weighted code will have a fixed weight for each position. For example, in normal binary system, the decimal equivalent can be obtained by multiplying the position value with position weight and adding them together.

Q3) Give an example for Non-weighted code?

Answer:

Unlike weighted code, non-weighted codes will not have any weights. For example, Excess-3 code and Gray code. So the numbers that are represented using non-weighted code cannot be directly converted to decimal equivalents.

Q4) What is the key feature of Excess-3 code?

Answer:

Self-complementing: The 9's complement of an excess 3 number can be obtained simply by replacing its 1's with 0's and 0's with 1's.

Q5) In how many different ways can number 5 be represented using 2-4-2-1 code?

Answer:

2-4-2-1 represents the weights corresponding to bit positions.

So the two possible ways are: 1011, 0101

Q6) What are all the BCD numbers that can be uniquely represented in 2-4-2-1 weighted code?

Answer:

0, 1, 8 and 9 (Only these 4 numbers will have unique representations).

Q7) How many unused combinations are there in the representation of BCD numbers using 7-4-2-1 weighted scheme? What are they?

Answer:

As BCD numbers range from 0 to 9, there are 5 unused combinations in 7-4-2-1 code. They are: 1011, 1100, 1101, 1110 and 1111.

Q8) What is the condition for a weighted code to be self-complementary?

Answer:

A weighted code is self-complementary if the sum of the weights equals to 9.

E.g.: 2-4-2-1 code. Sum of the weights = $2+4+2+1 = 9$

Q9) Convert the binary number 011101010001 to octal and hexadecimal?

Answer:

(a) Binary: 011 101 010 001

Octal: 3 5 2 1

(b) Binary: 0111 0101 0001

Hex: 7 5 1

Q10) Formulate a simple procedure for converting base-3 numbers to base-9?

Answer:

Consider $(X_{n-1} X_{n-2} \dots X_3 X_2 X_1 X_0)$ a n bit base 3 number.

The corresponding decimal equivalent is given by,

$$\begin{aligned} & 3^{n-1} X_{n-1} + 3^{n-2} X_{n-2} \dots + 3^3 X_3 + 3^2 X_2 + 3^1 X_1 + 3^0 X_0 \\ & = 9^{(n-2)/2} (3 X_{n-1} + X_{n-2}) + \dots + 9^1 (3 X_3 + X_2) + 9^0 (3 X_1 + X_0) \end{aligned}$$

So take every two digits of base-3 number from LSB side, find their decimal equivalent, it will be the corresponding base-9 digit. (Similar to the procedure of converting a binary number to octal or hex)

Q11) Convert (211101222211122)₃ to base-9?

Answer:

Base-3:(2 11 10 12 22 21 11 22)₃

Base-9: 2 4 3 5 8 7 4 8

Q12) A number N has 'n' digits in a r-radix number system. What is its (r-1)'s complement and r's complement?

Answer:

$(r^n - 1) - N$, $(r^n - N)$

Q13) What is the 9's complement of the BCD number 752?

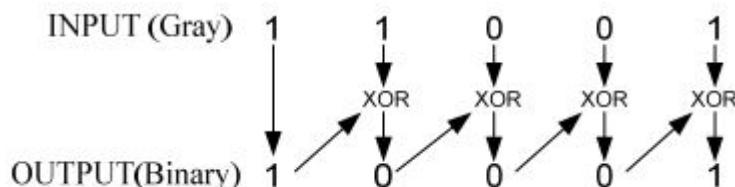
Answer:

$$999 - 752 = 247$$

Q14) Convert the Gray code number 11001 to binary code?

Answer:

Conversion from gray to binary: Retain the MSB as it is. XOR the current input bit with the previous output bit to get the new output bit. In this case, given gray code number is 11001



So, the required binary number is 10001.

Q15) Give the procedure for converting a binary number to Gray code?

Answer:

Binary to Gray code conversion: Retain MSB. XOR Current bit of Binary input with the previous bit of Binary input to get new bit of Gray code Output.

Q16) Represent 45 in the number systems (a) binary (b) BCD (c) Excess-3 (d) Gray code

Answer:

(a) 101101

(b) To get BCD: Represent each digit separately in binary 0100 0101

(c) Excess-3: Add 3 to each digit and then represent them separately in binary 0111 1000

(d) Gray code: First convert to Binary and use the procedure shown in Q15: 111011

Q17) Give the range of the numbers that can be represented using n bits in 2's complement method?

Answer:

$$-(2^{n-1}) \text{ to } +(2^{n-1} - 1)$$

Q18) What is the minimum number of bits required to represent the numbers in the range of -5 to 23 using 2's complement method?

Answer:

$$(2^{n-1} - 1) > 23 \Rightarrow 2^{n-1} > 24 \Rightarrow 2^{n-1} = 32 \Rightarrow n-1 = 5 \Rightarrow n = 6 \text{ bits}$$

Q19) What will be the result if all the zeros are retained from LSB side until 1 is seen, including that 1, and complement all the following bits of a binary number?

Answer:

2's compliment of the binary number

E.g.: Consider 10010, Its 2's compliment is given by 01110

Q20) In a particular design which uses 5 bits for integral part and 7 bits for fractional part, the result of some operation is 7B8 hex. Find the corresponding decimal equivalent?

Answer:

78B in hex = 01111.0111000 (5 integral bits and 7 fractional bits)

$$= 15.4375$$

Q21) Convert 0.95 to its binary equivalent?

Answer:

$$0.95 \times 2 = 1.90 \text{ ---- 1}$$

$$0.90 \times 2 = 1.80 \text{ ---- 1}$$

$$0.80 \times 2 = 1.60 \text{ ---- 1}$$

$$0.60 \times 2 = 1.20 \text{ ---- 1}$$

$$0.20 \times 2 = 0.40 \text{ ---- 0}$$

$$0.40 \times 2 = 0.80 \text{ ---- 0}$$

$$0.80 \times 2 = 1.60 \text{ ---- 1}$$

$$\text{So, } 0.95 = 0.111100110011001100\dots$$

Q22) AB16 – 3E16 = ?

Answer:

$$\begin{aligned}AB_{16} - 3E_{16} &= (171) - (62) \\&= (109) = 6D_{16}\end{aligned}$$

Q23) Solve for x: $(70)_8 + (122)_6 = (211)x$?

Answer:

$$2x^2 + x + 1 = 56 + 50 = 106 \Rightarrow x(2x+1) = 105 = 7 \times 15 \Rightarrow x = 7$$

Q24) Solve for X in $(135)_{12} = (X)_8 + (78)_9$?

Answer:

$$\begin{aligned}(78)_9 &= 63 + 8 = (71)_{10} \\(135)_{12} &= 144 + 36 + 5 = (181)_{10} \\(71)_{10} - (181)_{10} &= (110)_{10} = (156)_8 \\ \text{So, } X &= 156\end{aligned}$$

Q25) Explain the detailed procedure for BCD addition?

Answer:

BCD addition is similar to any binary addition. But if the result is above 9, to get valid BCD result, we need to add 6 to the result.

Q26) Perform BCD addition: $(1001) + (0110)$?

Answer:

$$\begin{array}{r}1001 \\0110 \\ \hline 1111 \text{ (>9)} \\6,0110 \\ \hline 10101\end{array}$$

So, the result is $0001\ 0101 = (15)$

Basic Gates and Boolean Algebra

Q1) Which gates are called universal gates? Why?

Answer:

NAND and NOR gates are called universal gates. Because any other logical gate like AND, OR, NOT, XOR, XNOR etc. or any other Boolean function can be implemented only with NAND or NOR gates.

Q2) How many minterms or maxterms will be there for n-inputs?

Answer:

For n inputs, possible minterms/maxterms = 2^n . For example, for 2 inputs the possible 4 minterms are $A'B'$, $A'B$, AB' , AB and maxterms are $A+B$, $A'+B$, $A+B'$, $A'+B'$.

Q3) Give the minterm and maxterms corresponding to 6 and 15 numbers (4-inputs)?

Answer:

(a) $6 = (0110)_2$ Minterm = $A'BCD'$, maxterm = $A+B'+C'+D$ (b) $15 = (1111)_2$ Minterm = $ABCD$, maxterm = $A'+B'+C'+D'$

Q4) In how many ways can a NAND gate be converted into an inverter? Show all the possibilities?

Answer:

A NAND gate can be converted into an inverter by using any of the following two methods:



Q5) How many number of 2 input AND gates are required to generate N I/P AND gate?

Answer:

N-1. For example to implement a 4 input AND gate we need three 2-input AND gates.

Q6) State De-Morgan's Laws?

Answer:

$$(A+B+C+D\dots)' = A'.B'.C'.D' \dots$$

$$(ABCD\dots\dots)' = A' + B' + C' + D' \dots$$

Q7) (a) If it is given that A & B will not be 1 at the same time, what will be the equivalent logical gate for an XOR gate?

(b) If any of the inputs of an XOR gate are inverted, XOR gate will work as ---- ?

Answer:

(a) OR Gate.

We can conclude this from truth table. Also from Boolean algebra as shown here :

As $A=B=1$, can not occur, $AB = 0$ always.

$$A \text{ XOR } B = AB' + A'B = A(AB)' + B(AB)' = A(0)' + B(0)' = A + B$$

(b) XNOR Gate.

$$A \text{ XOR } B = AB' + A'B$$

$$\text{Using this, } A' \text{ XOR } B = A \text{XOR } B' = A'B' + AB = A \text{ XNOR } B$$

Q8) State the Shannon's expansion theorem for representing a Boolean function by its co-factors?

Answer:

According to Shannon's expansion theorem any Boolean function $F(A,B,C,D,\dots)$ can be represented as $F = A F_A + A' F_{A'}$, where the cofactors F_A and $F_{A'}$ are given as, $F_A = F(1,B,C,D,\dots)$ and $F_{A'} = F(0,B,C,D,\dots)$

Q9) Write the cofactors F_A and $F_{A'}$ for $F(A,B,C,D) = ABD + BCD' + A'B'C'$?

Answer:

$$F_A = BD + BCD' \text{ and } F_{A'} = BCD' + B'C'$$

Q10) How many unique Boolean functions can exist for 'n' number of inputs?

Answer:

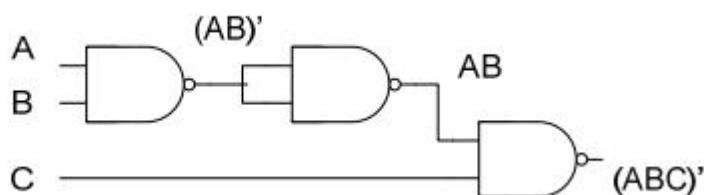
$$F_A = BD + BCD' \text{ and } F_{A'} = BCD' + B'C'$$

Q11) Mention the logical gates for which the 3 input implementation can not be obtained from two 2 input gates? How do you implement them?

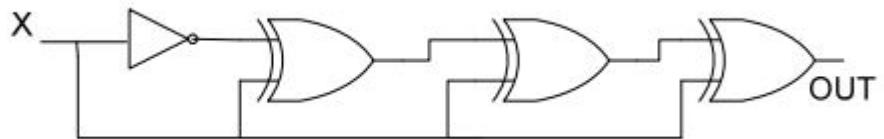
Answer:

$2^2 \times 2^n$ For n inputs, the possible number of minterms are, $k = 2^n$.

Any boolean function is combination of minterms. So all possible Boolean functions are ${}^k C_0 + {}^k C_1 + {}^k C_2 + {}^k C_3 + \dots + {}^k C_k = (1+1)^k = 2^k = 2^{2^n}$



Q12) What is OUT in the circuit shown below?



Answer:

First XOR gate output = $X \text{ XOR } X' = 1$

Second XOR output = $1 \text{ XOR } X = X'$

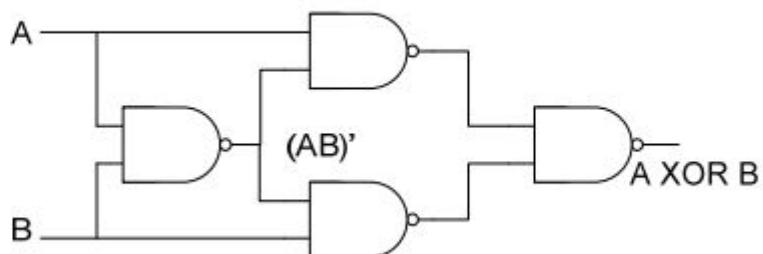
Third XOR gate output = OUT = $X' \text{ XOR } X = 1$

OUT = 1 irrespective of X

Q13) Give implementation of XOR using minimum number of NAND gates?

Answer:

$$A \text{ XOR } B = A'B + AB' = A(AB)' + B(AB)'$$



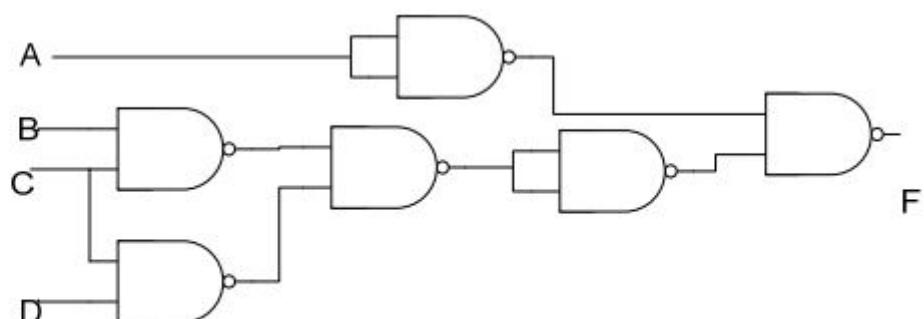
Answer:

6

A = Switch B=Sensor1 C=Sensor2 D=Sensor3 Pressed or sensor activated = 1

F=Shutdown=1

If you use K-Map and simplify, you will get $F = A + BC + CD$. The implementation of the same is shown below.



You can always use the truth table and then use Boolean simplification to get the result.

Q15) Majority function is the one which gives 1 if the input has more 1s than 0s. Show the truth table and give the AOI for 3-input majority function?

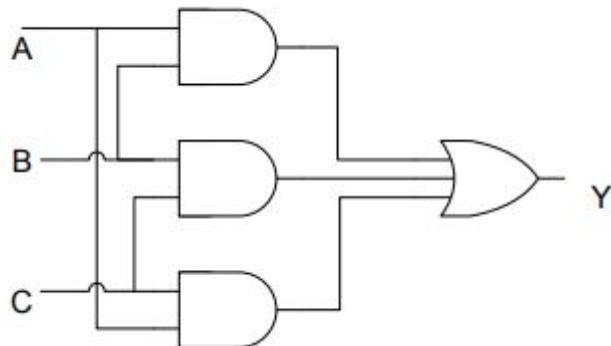
Answer:

Truth table for 3-input majority function is shown below:

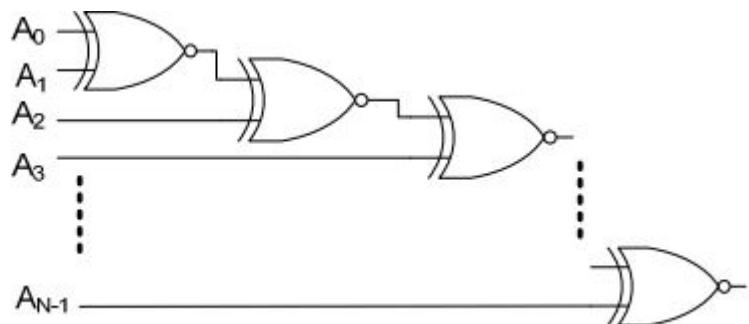
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$$Y = AB + BC + AC$$

The AND-OR Implementation (AOI) of Y is shown below:



Q16) N number of XNOR gates are connected as shown below. How does this circuit work? Explain?

**Answer:**

This circuit will work in two different ways based on N-value. (a) N is odd (b) N is even

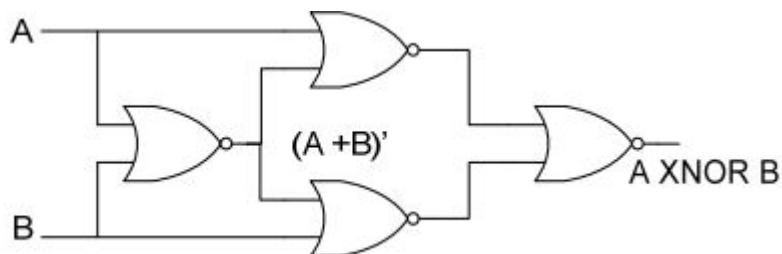
(a) If N is odd, there will be even number of XNOR gates in the circuit. Take an example of N=3, So there are 2 XNOR Gates. The two bubbles will get cancelled and this works as XOR. Same works for any odd N. So if N is odd it works as XOR Gate.

(b) If N is even, the circuit works as XNOR Gate. (Apply the same logic). One extra bubble will be there to make XOR to XNOR. You may verify the same for N=4.

Q17) Show the implementation of XNOR gate using minimum number of NOR Gates?

Answer:

Very much similar to Answer 13.



Q18) Explain parity generation and its significance?

Answer:

Parity generation adds an extra bit to the data which indicates the parity of input data. Parity generation is of two types: Even-parity and odd-parity generation. Even parity generator gives 1 if the input has odd number of 1's so that overall number of 1's will be even. Similarly odd parity generator gives 1 if the input has even number of 1's.

In data transmission systems the transmission channel itself is a source of data error. Hence the need to determine the validity of transmitted and received data. The validity of data is essential in applications where data is transmitted over long distances. Invalid data is a corruption risk. Parity generation helps in checking the validity of the data. Parity generation and validation is a scheme to provide single bit error detection capabilities.

Q19) Which logical gates can be used as parity generators?

Answer:

XOR gate can be used as even parity generator and XNOR can be used as odd parity generator.

Q20) What is the parity of (i) 10111001 (ii) 11001010

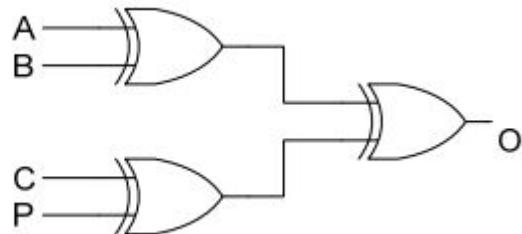
Answer:

- (i) ODD as the number of ones = 5, odd number
- (ii) EVEN, number of 1s =4,even

Q21) Give a circuit for 4-bit even parity checker? And explain the same how can it be reused for parity generation?

Answer:

The following circuit shows a parity checker for 4 inputs. A, B and C is the actual data. Whereas P is even parity bit generated at the transmitter. $P = A \text{ xor } B \text{ xor } C$. So A, B, C and P together will have even parity always. If all the bit sequences are received properly, O should be zero always. O=1 indicates that some error has occurred during transmission.

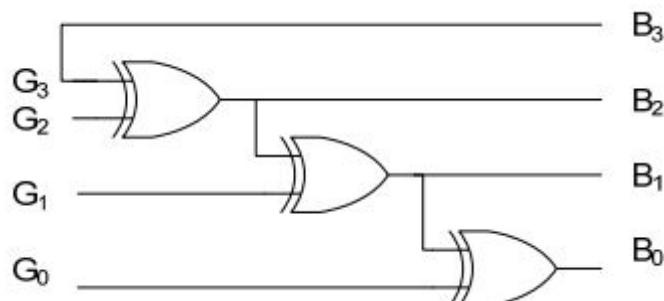


The same circuit can be used for parity generation by putting $P = 0$. If $P=0$, the same circuit works as 3-input even parity generator.

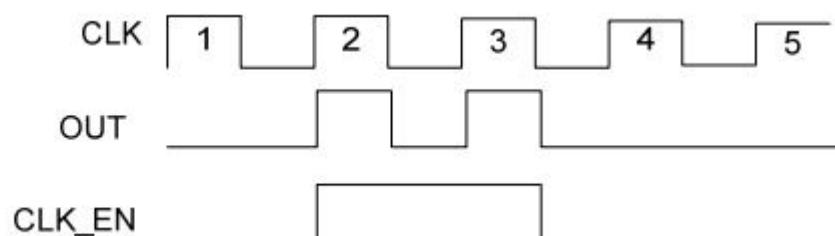
Q22) Design a combinational circuit using XOR gates that converts a 4-bit gray code number to a 4-bit binary number?

Answer:

The detailed procedure with an example for converting gray code to binary is shown in chapter 1. The same concept is shown with the XOR gates here.

**Answer:**

The input clock, the OUT that is needed and the corresponding CLK_EN are shown in the following diagram:



Q24) Which logical gate can be used to find out whether the two single bit inputs are equal or not?

Answer:

XNOR gate.

If we observe the truth tables, XNOR gate gives 1 if both the inputs are same.

Similarly XOR gives 1 if both the signals are different.

Q25) What is the difference between NAND gate and negative AND gate?

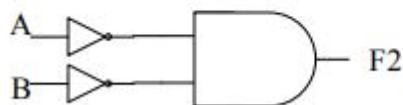
Answer:

(a) NAND gate :



$$F1 = (AB)' = A' + B'$$

(b) Negative AND gate:



$$F2 = A'.B' = (A + B)' \text{ (Equivalent of NOR Gate)}$$

Q26) How to obtain the dual of a Boolean equation?

Answer:

Dual : Replacing AND (NAND) with OR(NOR) and OR (NOR) with AND(NAND) in a given boolean equation gives the dual.

Answer:

- a) iv
- b) iii
- c) ii
- d) i

K-Maps

Q1) Define: SOP from and POS form?

Answer:

SOP: Sum Of Products : OR of all ANDs Eg: $F(A,B,C) = A + BC$

POS: Product Of sums: AND of all ORs Eg: $F(A,B,C) = (A'+B')(A'+C')$

Q2) When is a SOP/POS form is called standard or canonical?

Answer:

A SOP is called standard if each term is a minterm. Similarly a POS is called standard if each term is a maxterm.

Q3) Write the POS from for a 3-input XNOR gate? Is it canonical?

Answer:

$$Y = A \text{ XNOR } B \text{ XNOR } C$$

A	B	C	Y
0	0	0	1
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	0

$$Y = (A'B'C + A'BC' + AB'C' + ABC)'$$

So Y is POS form is $(A+B+C') (A+B'+C) (A'+B+C) (A'+B'+C')$

As each term in the POS form is maxterm, this is canonical.

Q4) Which form is suitable for designing logic circuits using

- (a) Only NAND gates
- (b) Only NOR gates

Answer:

- (a) SOP form
- (b) POS form

Q5) In which order are the bits arranged while drawing K-Maps?

Answer:

Hamming order (Gray code)

Q6) Why do we write 00 01 11 10 in that order while Drawing K-maps?

Answer:

In K-Map, the Boolean simplification is done by grouping the adjacent cells that have 1. To get the simplified expression, the adjacent cells must have 1 bit change. So gray code is used.

Q7) How many cells will a n-input variable have in K-Map?

Answer:

2^n . E.g.: 3 variables, 8 cells. Similarly..4 variables 16 cells.

Q8) How many dimensions (without projections) are there for n karnaugh map ($n > 2$)?

Answer:

Ceiling ($\log_2 n$)

Q9) What do you mean by don't care condition?

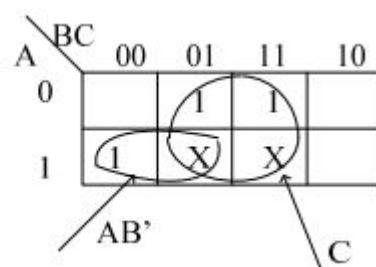
Answer:

The don't care condition set accommodates input patterns that never occur or outputs that will not be observed.

Q10) $Y = A'C + AC'B'$ and you are given that $A=C=1$ will never occur. Simplify Y?

Answer:

$Y = A'C + AC'B'$ and the output will be don't care for $A = C = 1$. So the K-map will be as follows:



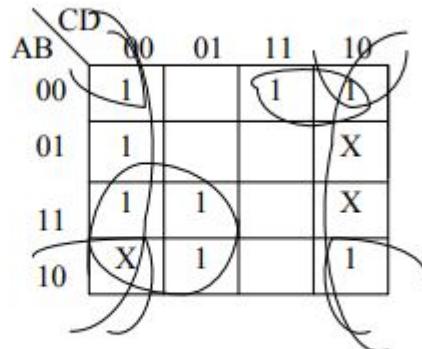
Thus the simplified expression for Y is $AB' + C$

Q11) $Y = \sum (0, 2, 3, 4, 9, 10, 12, 13)$

$= d(6, 8, 14)$. Simplify using KMap. Mention Prime Implicants & Essential Prime Implicants?

Answer:

The K-Map is :



The prime implicants are : D' , $B'D'$, AC' , $A'B'C$

The essential prime implicants are : D' , AC' , $A'B'C$

The simplified expression is : $Y = D' + AC' + A'B'C$

Q12) $Y = F(A,B,C,D) = \sum (0,1,4,5,7,9,12)$. Express the same using \prod ?

$= d(6,8,14)$.Simplify using KMap. Mention Prime Implicants & Essential Prime Implicants?

Answer:

$$Y = \prod (2,3,6,8,10,11,13,14,15)$$

Q13) If $F(A,B,C,D,E) = B'E$, how many terms will be there in the standard or canonical SOP representation of F?

Answer:

$$8 \text{ terms, } F = B'E (A + A') (C+C') (D+D')$$

Q14) In a 6 variable K-map, how many literals will the grouping of 4 adjacent cells give in the term?

Answer:

$$6 - \log_2 4 = 6 - 2 = 4$$

Q15) Generalization of Q13: The grouping of k adjacent cells, in a N variable K-Map will lead to a term of ---- literals?

Answer:

In 3 variable map, grouping all 8 cells will give zero literals in the term as it is logical 1 always. Similarly, in 4 variable map the same grouping will give 1 literals, in 5 a variable map it is 2 and so on..

So the literals in the term = $N - \log_2 k$

Q16) If the number of variables are more,(>5) , which method is suitable for Boolean simplification?

Answer:

Q-M Method

Q17) In the simplification of a Boolean function, $F = \sum (0,1,2,6,7,8,9,10,14,15)$ using QM method the following table is obtained: Q1) Define: SOP from and POS form?

	0	1	2	6	7	8	9	10	14	15
BC				X	X				X	X
CD'			X	X				X	X	
B'C'	X	X				X	X			
B'D'	X		X			X		X		

- (a) Identify the essential prime Implicants?
- (b) Find out the simplified expression for F in SOP form?

Answer:

Checkout for the columns which has only one entry (X), that term must be included in the simplified expression. So, that term will be essential.

- (a) So the essential prime implicants are: BC and B'C'
- (b) The simplified expression $F = BC + B'C' + CD' = BC + B'C' + B'D'$

Q17) In the simplification of a Boolean function, $F = \sum (0,1,2,6,7,8,9,10,14,15)$ using QM method the following table is obtained: Q1) Define: SOP from and POS form?

Answer:

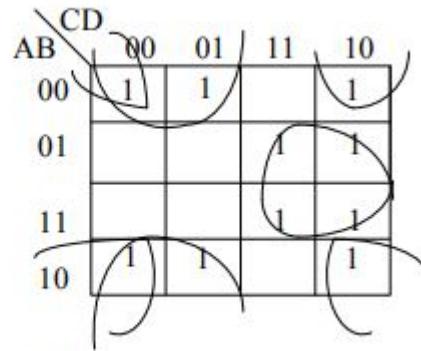
Checkout for the columns which has only one entry (X), that term must be included in the simplified expression. So, that term will be essential.

- (a) So the essential prime implicants are: BC and B'C'
- (b) The simplified expression $F = BC + B'C' + CD' = BC + B'C' + B'D'$

Q18) Use K-Map to simplify $F = \sum (0,1,2,6,7,8,9,10,14,15)$ in SOP form? Cross check the essential prime Implicants that are obtained in Q17.

Answer:

The K-Map is shown below:



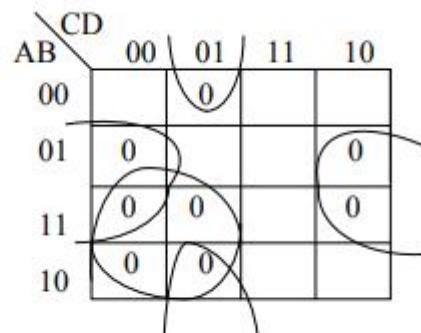
The EPI = BC and $B'C'$

$$F = BC + B'C' + CD' = BC + B'C' + B'D'$$

Q19) Simplify the Boolean function $Y = \sum (0, 2, 3, 5, 7, 10, 11, 15)$ in POS form using KMap?

Answer:

To simplify the Boolean function in POS form we need to map for 0s and then take the compliment of that function to get Y in POS form.



$$Y = (AC' + BD' + B'C'D')' = (A'+C)(B'+D)(B+C+D')$$

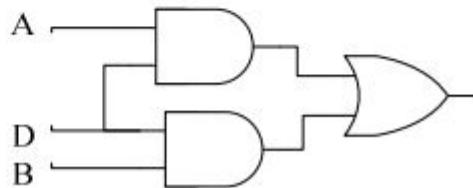
Q20) Give the AND-OR implementation of a circuit, using minimum gates, that gives HIGH when the input is BCD equivalent of 5, 7 or 9 and LOW otherwise. ?

Answer:

As the input is a BCD number, the output will be don't care for the input combinations, 10, 11, 12, 13, 14 and 15. So the K-Map will be as shown below:

CD \ AB	00	01	11	10
00				
01		1	1	
11	X	X	X	X
10		1	X	X

The simplified Boolean function = $Y = AD + BD$



Combinational logic

Q1) (a) Show the AOI implementation of a 2:1 Mux?

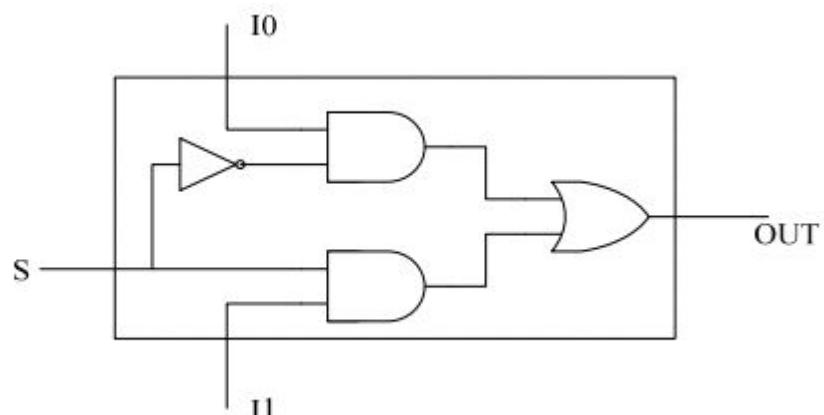
(b) Convert this to 2-input NAND implementation?

Answer:

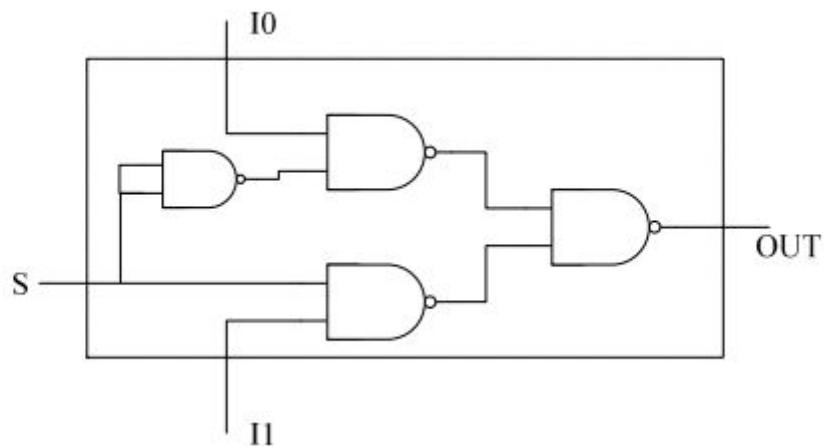
(a) For a 2:1 mux, whose inputs are I0,I1 and select line S, the out put is given by the following boolean expression:

Out = $S' I0 + S I1$

The AND-OR Implementation (AOI) is shown below:



(b) The NAND gate implementation is:

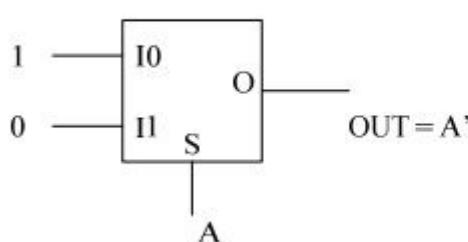


Q2) Design the following gates using minimum number of 2:1 Muxes?

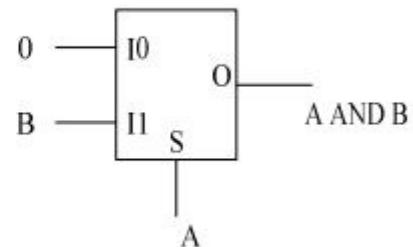
- (a) NOT (b) AND (c) OR (d) XOR

Answer:

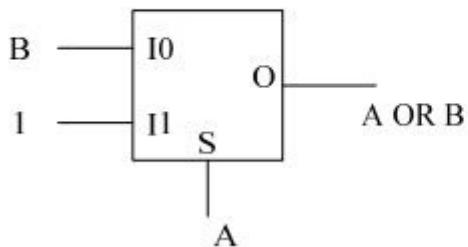
(a) NOT Gate using a 2:1 Mux:



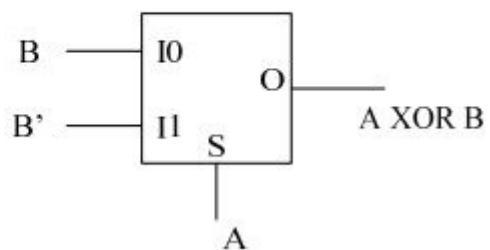
(b) AND gate



(c) OR gate



(d) XOR gate

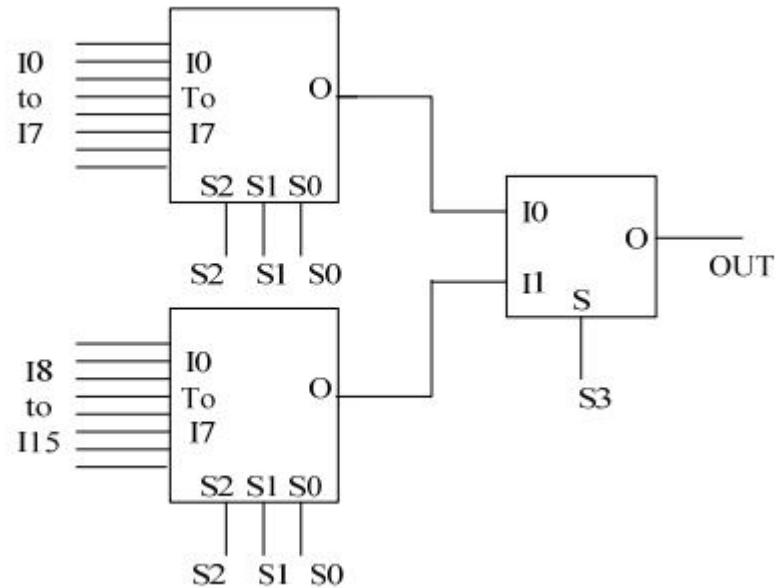


To get B' , we need an extra 2:1 Mux, as inverter. Similarly NAND, NOR, XNOR gates can also be implemented. All these implementations need 2 2:1 Muxes (similarly to XOR gate).

Q3) Construct a 16:1 Mux with two 8:1 Mux and one 2:1 Mux.

Answer:

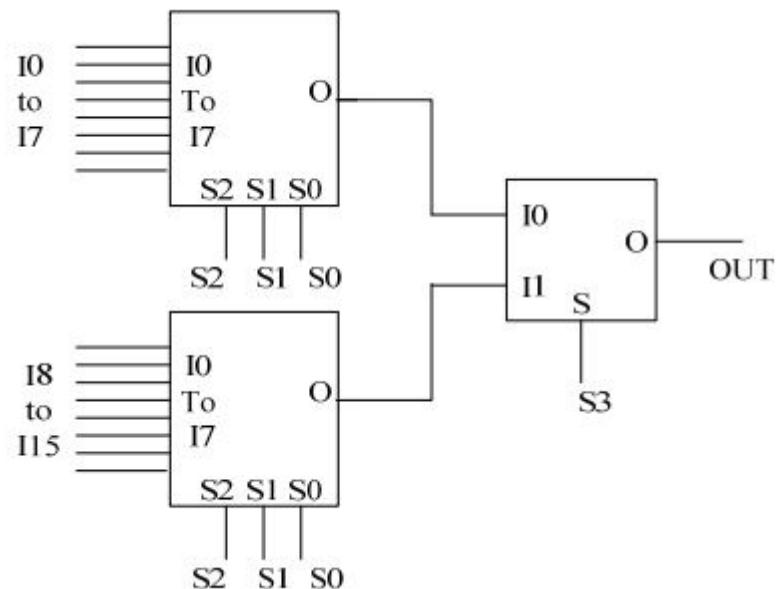
$$Y = A'B'C + A'BC' + AB'C' + ABC = A \text{ XOR } B \text{ XOR } C$$



Q4) Find out the simplified expression for Y in terms of A, B and C?

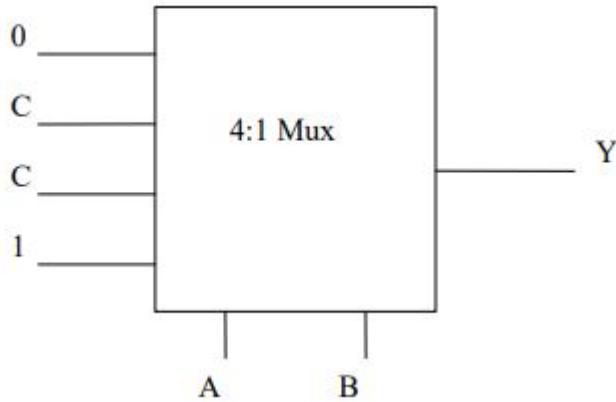
Answer:

$$Y = A'B'C + A'BC' + AB'C' + ABC = A \text{ XOR } B \text{ XOR } C$$



Q5) Design a circuit for 3-input majority function using a 4:1 Mux?

Answer:



The majority function gives 1 if the input has more number of 1s than zeros. The truth table is shown in Chapter 3. If $A=B=0$, the output will be zero irrespective of C. If $A=B=1$, output is 1 irrespective of C. But if $A=1, B=0$ or $A=0, B=1$, as we can not decide the majority without knowing C. So $I_0 = 0$, $I_1 = I_2 = C$ and $I_3 = 1$. The implementation is shown above. If we simplify the expression it gives,

$$Y = AB + BC + AC$$

- Q6) (a) Expand the Boolean function $f(x,y,z) = x'z' + xy + xz$ in terms of x?
 (b) Implement f using a 2:1 Mux and external gates?

Answer:

Shannon's expansion theorem is highly useful in implementing a Boolean function using Multiplexer.

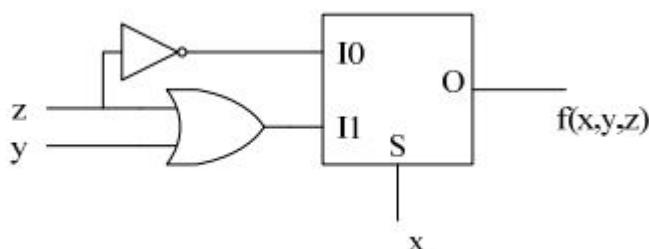
(a) To get $f(x,y,z) = x'z' + xy + xz$ in terms of x, first use Shannon's expansion theorem to get the following co-factors. The cofactors are:

$$f_x = f(1,y,z) = y + z$$

$$f_{x'} = f(0,y,z) = z'$$

$$\text{So, } f(x,y,z) = x f_x + x' f_{x'}$$

(b) Now we can use x as select line and implement $f(x,y,z)$ using 2:1 mux



- Q7) There is a single telephone which needs to transmit the data from 8 different users to the receiving end. Give a design which can accomplish this task?

Answer:

We need a 8:1 Mux at the input side and 1:8 Demux at the receiver side. We may need an 8 bit counter which runs at the same clock speed on both the sides to select one of the

user.

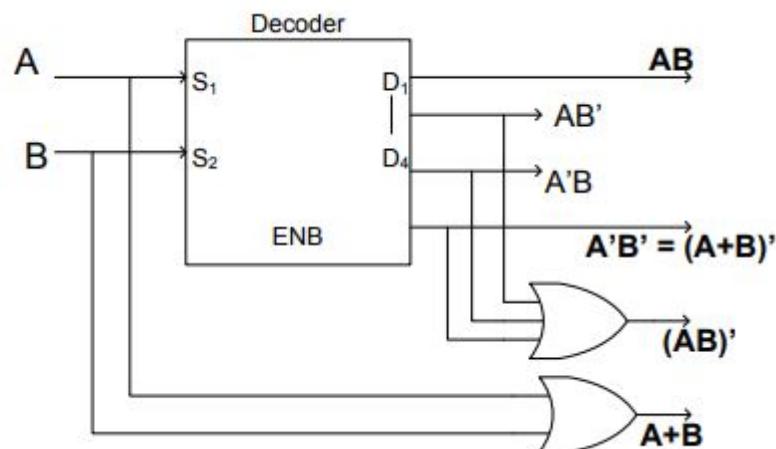
Q8) You are given a 2:4 decoder, a 2 input OR gate and a 3 input OR gate. Using these components design a system which takes A & B as inputs and generates the following four outputs: AB, (AB)', A+B, (A+B)'.

Answer:

A 2:4 decoder will have 4 O/Ps which are the minterms of the 2 inputs : AB, AB', A'B, A'B'. Out of the four outputs that are needed, AB and (A+B)' = A'B' are directly available. Whereas A+B can be obtained using the extra 2 input OR gate (which is given). So only O/P that is needed is (AB)'.

$$(AB)' = A' + B' = A'(B+B') + B'(A+A') = A'B + AB' + A'B'$$

So use 3-input OR gate to obtain (AB)'. The whole design is shown below.



Q9) Give the truth table for (a) half-adder and (b) half-subtractor?

Answer:

(a) Truth table for Half-adder:

A	B	Cout	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

(b) Truth table for Half-Subtractor:

A	B	Bout	Diff
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Q10) Design a circuit for half-subtractor using basic gates?

Answer:

From the above truth table (A9(b)), we can derive the following equations for borrow and difference:

$$\text{Borrow} = \text{Bout} = A'B$$

$$\text{Difference} = \text{Diff} = A \text{ XOR } B$$

The same equations can be shown using basic gates. (XOR, NOT, AND)

Q11) Design "OR" gate using HA's?

Answer:

The HA equations are, $\text{Cout} = AB$ and $\text{Sum} = A \text{ XOR } B = AB' + A'B$

$$\text{Sum XOR Cout} = \text{Sum}' \text{ Cout} + \text{Cout}' \text{ Sum}$$

$$= (AB + A'B') AB + (A' + B') (A'B + AB')$$

$$= AB + A'B + AB' = A + B$$

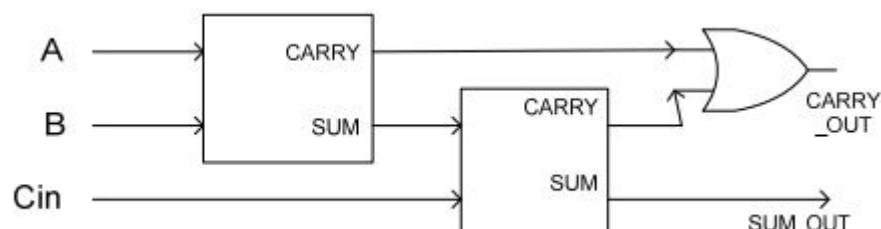
So to get OR gate we need two HA. The Sum and Cout of first HA are given as inputs to second HA. The Sum of second HA gives the A OR B.

Q12) Design a full adder using half-adders and minimum number of external gates?

Answer:

$$\text{Sum} = A \text{ XOR } B \text{ XOR } C \text{ and } \text{Carry} = AB + BC + AC$$

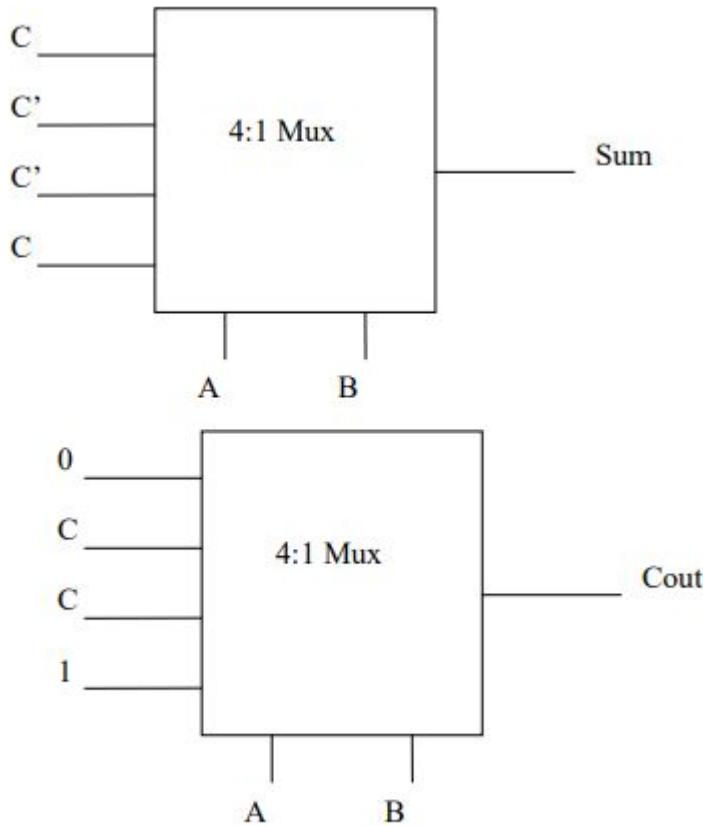
Full adder from 2 HA and one OR gate:



Q13) Implement a full adder using two 4:1 Muxes?

Answer:

$$\text{Sum} = A \text{ XOR } B \text{ XOR } C \text{ and } \text{Carry} = AB + BC + AC$$



Q14) A full adder can be implemented using basic gates in many ways. Show the efficient implementation among them, which needs minimal hardware?

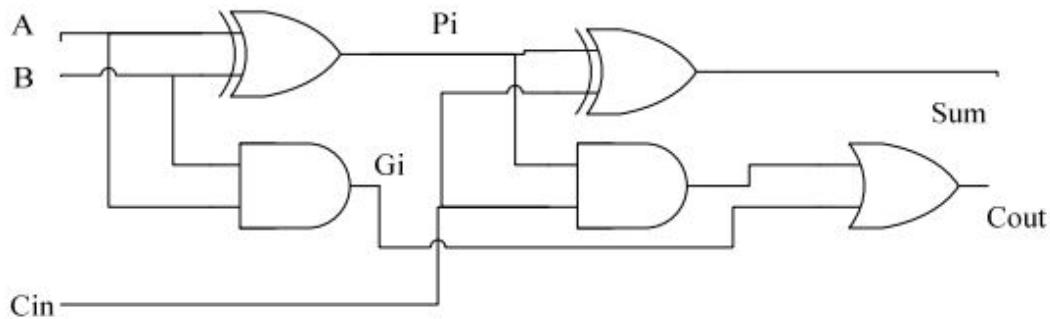
Answer:

The suitable equations are:

$$\text{Sum} = (A \text{ XOR } B) \text{ XOR } C$$

$$\text{Cout} = AB + (A \text{ XOR } B)C$$

The implementation is as follows:



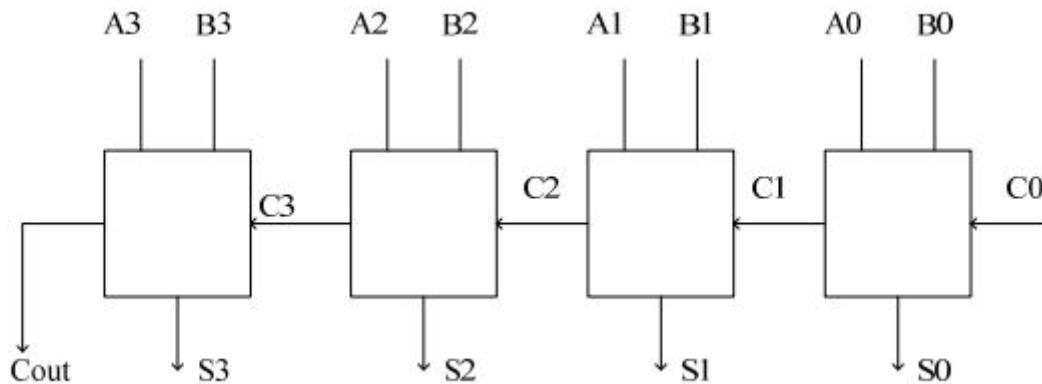
Q15) Implement a circuit for adding two 4-bit numbers using

- (a) Ripple carry adder
- (b) Carry Look Ahead (CLA) adder

Answer:

Two possible implementations of an adder are (a) Ripple carry adder and (b) Carry Look Ahead adder.

(a) The ripple carry adder for adding two 4 bit numbers, A and B is shown below:



S3-S0 indicates the final result and Cout is the final carry.

(b) Carry Look Ahead (CLA)

From the figure that is shown (A14), we can derive the following intermediate equations:

$$P_i = A_i \text{ XOR } B_i \text{ and } G_i = A_i \text{ AND } B_i \quad (1)$$

Now, Sum = P_i

XOR C_i

$$\text{and } C_{i+1} = G_i + P_i C_i \quad (2)$$

Using this equation for carry, we can generate the following algorithm for carry look ahead:

C_0 = Input carry

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1(G_0 + P_0 C_0) \text{ and so on.} \quad (3)$$

By using these recursive equations, we can look a head and decide the carrier, unlike in case of ripple carry adder.

So the three steps involved in CLA adder are:

Step1: Generation of P_i and G_i from A_i and B_i using Eq(1)

Step2: Generation of carrys using Eq(3)

Step3: Finally getting the Sum from P_i and C_i using Eq(2)

Q16) Compare the two implementations of Q15.

Answer:

In ripple carry adder, the carry propagates from first adder to last. As it has to pass through all the adders, the delay in getting the final output is considerably high. Where as it is hard-ware efficient. The scheme for CLA is explained in the previous question. The major advantage of CLA is faster output. But it needs more hardware.

Q17) If each XOR gate has a propagation delay of 10ns and AND/OR gate has a delay of 5ns each (irrespective of number of inputs), what is the total propagation delay in adding two 4 bit numbers in case of

- (a) Normal full adder
- (b) Carry Look Ahead adder.

Answer:

To perform the delay calculations, use the circuits that are shown in above answers.

(a) For full adder the best implementation is shown in A14.

XOR gate delay = 10ns and AND/OR gate delay = 5ns

The delay for each adder = $10 + 10 + 5 = 25\text{ns}$.

For adding 4-bits, we need 4 such adders, so overall delay = 100ns

(b) For CLA, as explained in A15, we need 3 main steps:

Step1: Generation of P_i, G_i : The XOR gate delay = 10ns

Step2: Generation of carry: And-Or stage = $5\text{ns} + 5\text{ns} = 10\text{ns}$

Step3: Sum generation: One more XOR operation = 10ns

So the overall delay = 30ns

Q18) Explain how a full adder can be used as majority function?

Answer:

The carry out of a full adder is equivalent to Majority function.

Majority function, $Y = \text{Cout} = AB + BC + AC$

Q19) Give the truth table of full subtractor? Design the same using full adder?

Answer:

The truth table for full subtractor is shown below:

The truth table for full subtractor is shown below:

A	B	C	Bout	Diff
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$\text{Diff} = A \text{ XOR } B \text{ XOR } C \text{ and } \text{Bout} = BC + A'(B \text{ XOR } C)$$

To implement full subtractor from full adder we need to XOR one of the inputs with 1 and then give to full adder as shown in the figure.



Q20) There is a sixteen bit adder with ripple carry. Which of the following gives minimum delay for the output? (Fastest output)

$$F_0 + F_1, FF + FF, FF + F_1$$

Answer:

$F_0 + F_1$ generates 4 carries, $FF+FF$ and $FF+F_1$ generates 8 carries. Whichever generates less number of carries, that one will give the fastest output. So F_0+F_1 gives fastest output.

The truth table for full subtractor is shown below:

A	B	C	Bout	Diff
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$\text{Diff} = A \text{ XOR } B \text{ XOR } C \text{ and } \text{Bout} = BC + A'(B \text{ XOR } C)$$

To implement full subtractor from full adder we need to XOR one of the inputs with 1 and then give to full adder as shown in the figure.



Q21) What is overflow? Under what conditions will it occur?

Answer:

Case1: Unsigned Numbers:

In N-bits, we can represent numbers from 0 to $(2^N) - 1$. Suppose if we are adding 2 N bit unsigned numbers and if the result is greater than $(2^N) - 1$, overflow will occur. To detect this, check whether the MSB addition (Nth bit) + Carry generated from (N-1) bit addition is generating any carry or not. If there is carry out, there is overflow.

Case2: Signed Numbers:

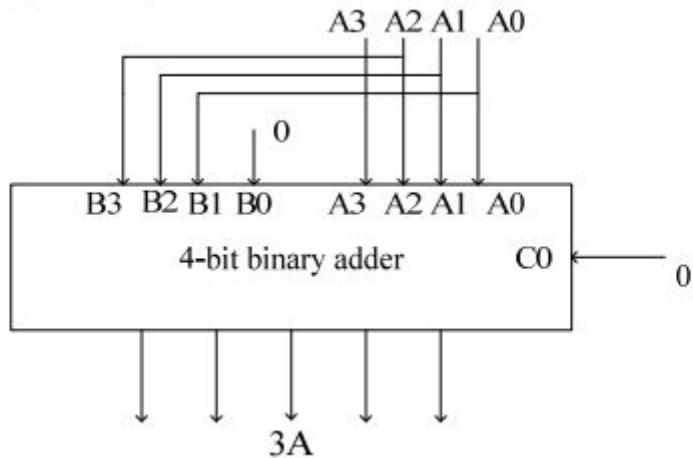
In N-bits, we can represent numbers from $-(2^{(N-1)})$ to $(2^{(N-1)}) - 1$. Suppose if we are adding 2 N bit signed numbers and if the result is not in the above range, overflow will occur. To detect overflow in this case:

- The sum of two positive numbers is negative;
- The sum of two negative numbers is positive;
- Subtracting a positive number from a negative one yields a positive result; or
- Subtracting a negative number from a positive one yields a negative result.

Q22) Using a 4-bit binary adder, design a circuit which multiplies the input by 3?

Answer:

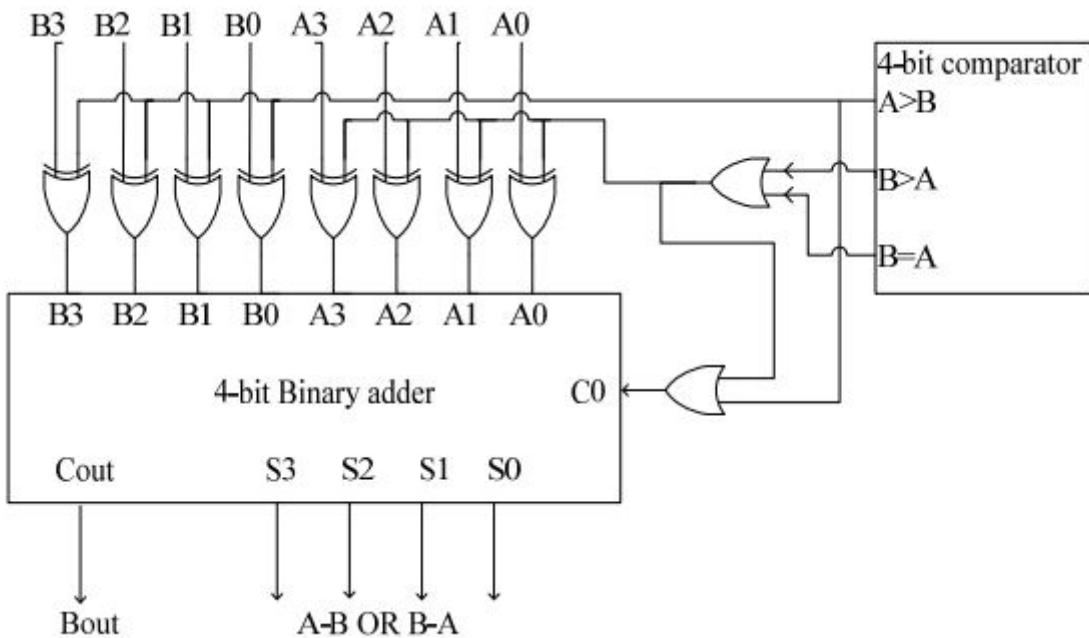
$Y = 3A = 2A + A$. The 4-bit binary adder which is shown in A15 (a) can be used as a block box here. As $2A$ can be obtained by simple right shift operation, one binary adder is sufficient for the complete design.



Q23) Design a subtractor unit using a 4-bit comparator, 4-bit binary adder and some external gates, which performs $A-B$ if $A>B$ and else $B-A$. A and B are two 4-bit binary numbers.

Answer:

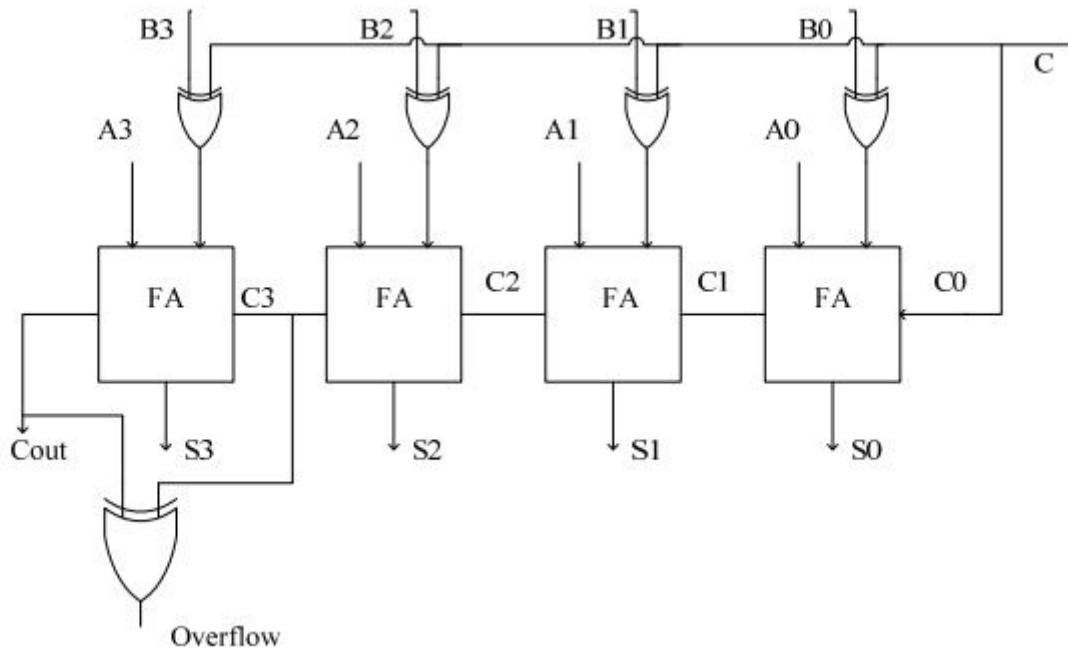
Note that $A-B = A + (-B)$. That is, to subtract B from A, just find the 2's of B and add that to A. If $A > B$, the comparator gives 1 at $A>B$ and zero at rest of the outputs. That 1 is used as one of the inputs to the XOR gate, to find the 2's compliment of B. Similarly, in case of $A < B$ or $A = B$, we need to find 2's compliment of A. The complete design is shown below:



Q24) Design an adder/subtractor unit using 4-bit binary adder and some external gates, which gives out $A+B$ if $C=0$ and $A-B$ if $C=1$. Also provide an indicator for checking the overflow?

Answer:

This is very much similar to the above design. The extra feature is the indication of overflow, which we can get from XOR of the carries C2 and C3.



Overflow = 1 indicates that overflow has occurred. (Refer to A21) And the other logic for finding the 2's compliment is exactly same as A24.

Q25) Use the above designed circuit as block box and give a scheme for finding the absolute value of a 4-bit number?

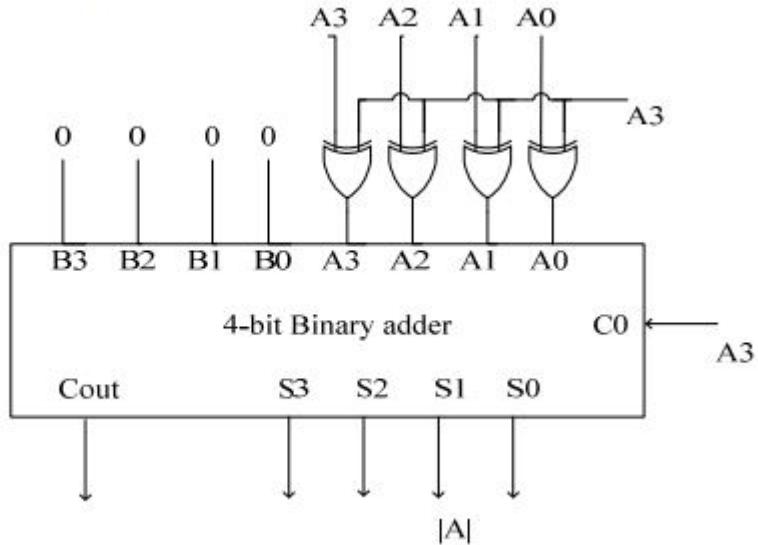
Answer:

Absolute value of a number is defined as

$$|A| = A \text{ if } A > 0,$$

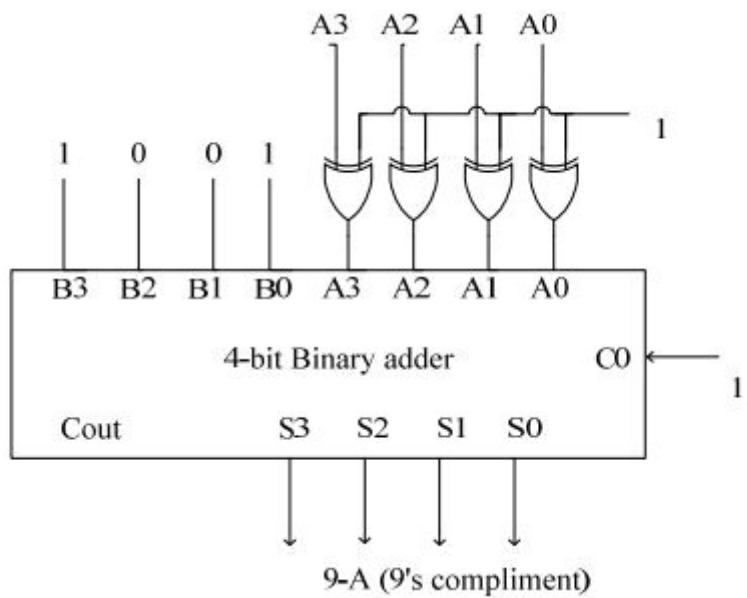
$$= -A \text{ otherwise}$$

We can use the above designed adder/subtractor unit to accomplish this task. The MSB of A, which will be 1 if A is negative, can be used as C.



Q26) Design a circuit that generates 9's compliment of a BCD digit using binary adder?

Answer:



9's compliment of a BCD number d is given by $9-d$. That is just find the 2's compliment of d and add that to 9. Cout is needed. Just the S3-S0 of the binary adder, shown in the above figure, gives the required result.

Q27) Give the circuit that adds two BCD numbers and gives out a BCD number?

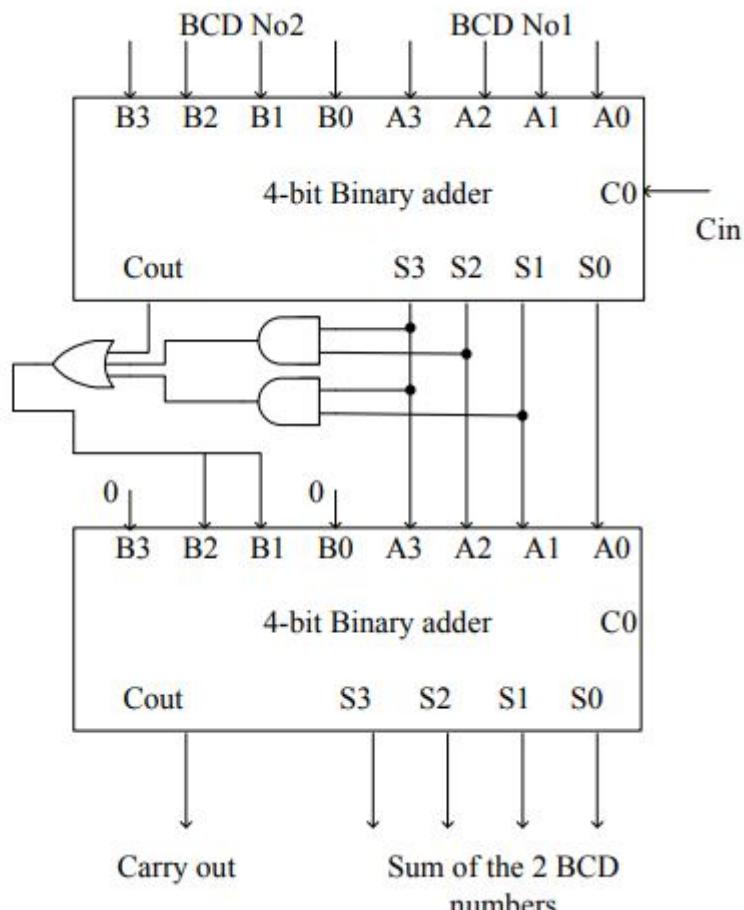
Answer:

The BCD addition is explained in Chapter1. If the result is above 9, it is needed to add 6 to obtain the result in BCD number system. So we need two 4-bit binary adders: One is just to add the two BCD numbers. The second one is for adding 6 or 0 to the result. Extra

combination logic is needed to identify the overflow. The condition for detecting the overflow can be derived as,

$$K = Cout + S_3 S_2 + S_3 S_1$$

$K = 1$ indicates the overflow and addition of 6 is needed. The complete design is shown below:



Q28) How will you count the number of 1's that are present in a given 3-bit input using full adder?

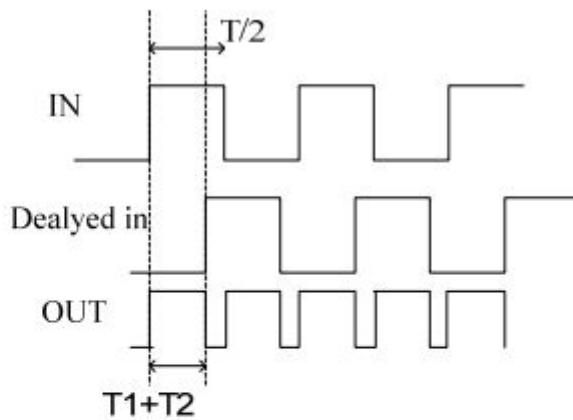
Answer:

The binary number that is formed from the Carry out as MSB and Sum as LSB, gives the number of 1s of input. The same thing is illustrated in the following table. The sixth column shows Cout-Sum together where as the last column shows the actually number of 1s in the input. Note that both are exactly same.

Answer:

(a) The circuit works as frequency doubler. That is, it gives double the frequency at the output. But the duty cycle depends upon the delay of the gates.

(b)



If you observe the output, it will be on for $(T_1 + T_2)$ duration. And the total period is $T/2$. (As output frequency = $2 * \text{input frequency}$).

$$\text{So duty cycle} = (T_1 + T_2) / (T/2) = 2(T_1 + T_2) / T \%$$

(c) To get 50% duty cycle, $T_1 + T_2$ should be equal to $T/4$. So you can use two inverters each with a delay of $T/8$.

Q30) Give the truth table for 4:2 priority encoder in which the LSB(D0) has the highest priority and MSB(D3) has the lowest priority.

Answer:

Encoder functionality is opposite of a decoder. The output of an encoder corresponds to the binary code of the input. There is a chance that, in the input stream, more than one 1 may present. In that case, to avoid clash, we need to provide the priority to any one of the bits. Here the truth table for priority encoder which gives, highest priority to its LSB, is shown:

D3	D2	D1	D0	Input		Output	
				x	y	x	y
0	0	0	0	X	X	X	X
X	X	X	1	0	0	0	0
X	X	1	0	0	1	0	1
X	1	0	0	1	0	1	0
1	0	0	0	1	1	1	1

Q31) You have three delay elements D1, D2, D3 that delay a clock by 25%, 50% and 75% respectively. Design a frequency doubling ($f_{out} = 2 * f_{in}$) circuit that uses these delay elements along with any combinational logic.

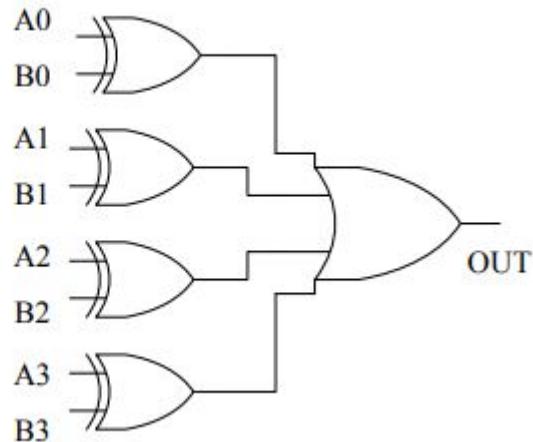
Answer:

The simple design using XOR gate is shown below. (Similar to A29)

Q32) Give a combinational circuit which checks out whether two 4-bit input signals are same or not?

Answer:

For finding out whether two signals are equal or not, the best logical gate is XOR. The design is shown below. OUT =1 implies that the two binary numbers are not equal.



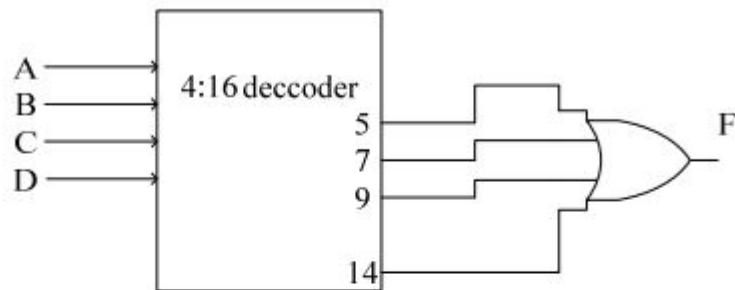
Q33) Using a 4:16 decoder and minimum number of external gates implement the following Boolean functions:

- (a) $F(A,B,C,D) = \sum (5,7,9,14)$
- (b) $G(A,B,C,D) = \sum (0,1,2,3,4,6,7,8,9,10,11,14,15)$

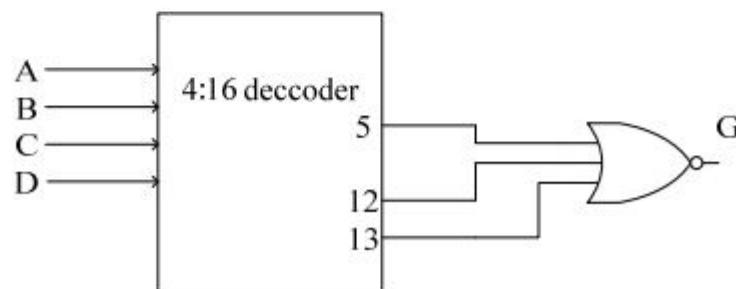
Answer:

The decoder gives all the possible minterms at the output. We need to just OR all the corresponding minterms to get a particular Boolean functions. If a Boolean function has more minterms which are 1, then take all the minterms which are zero and use NOR gate. Here in case (a) we can directly use OR gate. But case (b), NOR gate is to be used to get the optimal solution.

(a)



(b)



Introduction to flip-flops

Q1) Mention two basic applications of flip-flops?

Answer:

The major applications of flip-flops are:

- Data storage
- Data transfer
- Counting and
- Frequency division

Q2) What is the difference between a LATCH and a Flip flop?

Answer:

The differences between a LATCH and a FLIP-FLOP are:

- Latch is a level sensitive device where as flip-flop is edge sensitive
- Latch is sensitive to glitches on enable pin and flip-flop is immune to glitches
- Latches take less gates(also less power) than flip-flops
- Latches are faster than flip-flops.

Q3) What is transparent latch?

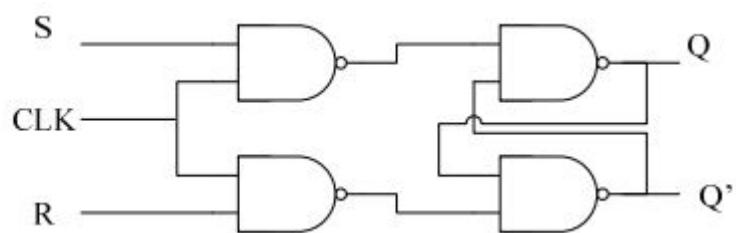
Answer:

D-Latch is called transparent Latch. As it transfers the data as it is to the output on enable.

Q4) Implement S-R Latch with control input using 2-input NAND gates?

Answer:

S-R Latch with clock using 2-input NAND gates is shown below:



Q5) Which input combinations are not allowed in (a) NAND based (b) NOR based S-R Latch? Explain.

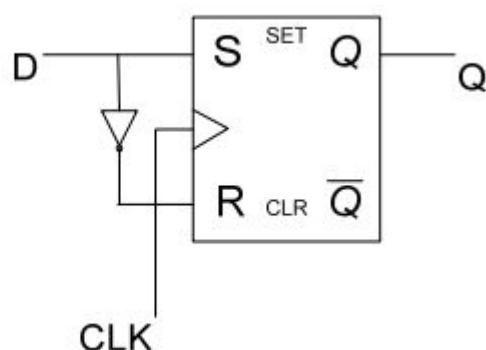
Answer:

- (a) S=R=0
- (b) S=R=1

Q6) How to convert S-R Latch to transparent latch?

Answer:

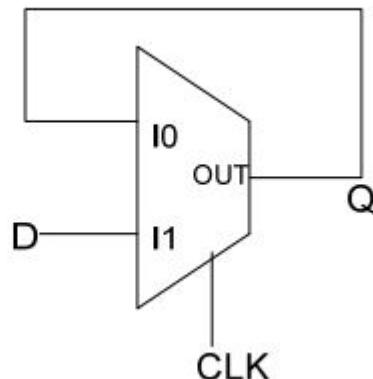
Transparent latch from S-R Latch:



Q7) Design a D-latch using 2:1 Mux.

Answer:

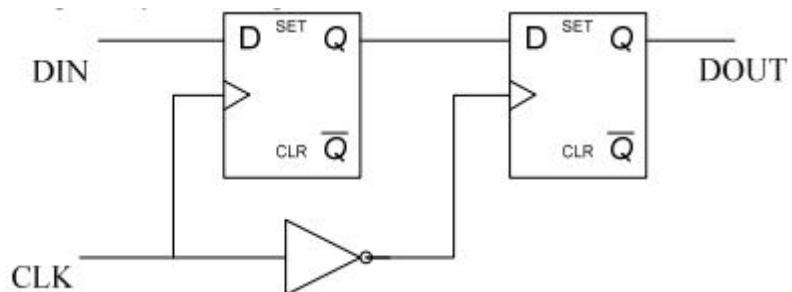
D-Latch using 2:1 Mux:



Q8) Design a master-slave D-Flip flop using D-Latch?

Answer:

2 D-latches and an inverter are needed to get the master-slave configuration. The major advantage of master-slave configuration is avoiding the race-around condition as the input changes only at the edges.



Q9) What is race-around condition? Explain it in case of J-K Latch and solution to avoid that?

Answer:

The race around condition means: the output oscillating between 0s & 1s. This problem will occur in Latches especially if the clock is high for long time. In case of J-K Latch, $J=K=1$ gives $Q(t+1) = Q(t)'$. Consider the case when clock is high for long time and $J=K=1$. Then the output oscillates between 0 & 1 continuously as long as the clock is high. To avoid this, use Master-Slave configuration which latches the input only at clock edges. So in that case, irrespective of the duration of clock high, output will be just compliment of previous output. There won't be any oscillation as such.

Q10) We have a circular wheel with half painted black and the other half painted white. There are 2 censors mounted 45 degree apart at the surface of this wheel(not touching the wheel) which give a "1" for black and "0" for white passing under them. Design a

circuit, using DFF to detect which way the wheel is moving. Can not assume any fixed position for start.

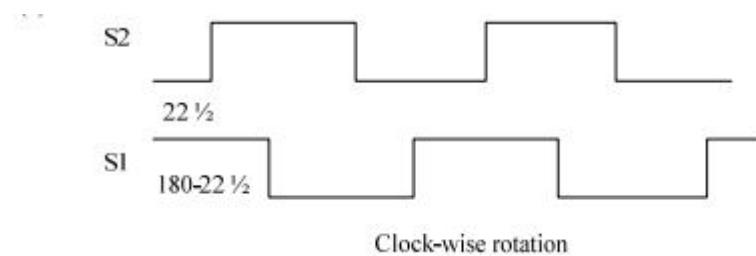
Answer:

The sensor will give 1 for Black and 0 for white. First we will draw the outputs of the sensors assuming some position of the wheel. Assume that the initial position of the wheel is as shown in the figure with respect to the sensors. The output waveforms of S1 and S2 will be as follows. Both clock wise and counter clock wise wave forms are shown here. It is clear from the waveforms that there is an initial delay of $22\frac{1}{2}$ degrees between the two waveforms (assuming the two sensors are at the same distance from the partition).

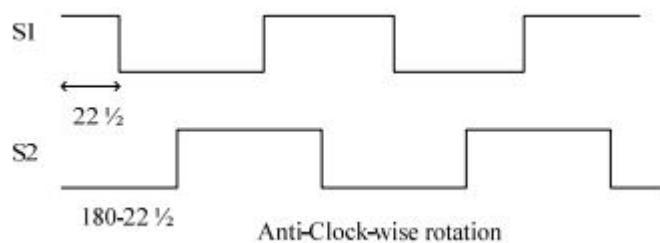
Initially $S_2 = 0$ and $S_1 = 1$ because of their initial positions (this is just our assumption). Once the wheel started moving in anti-clock wise direction, S_1 will go to 0 from 1 after $22\frac{1}{2}$ degrees. And then gets complemented once in 180 degrees. In this case, S_2 will go to 0 to 1 after $(180 - 22\frac{1}{2})$ degrees and then complements once in 180 degrees.

Similarly we can analyze for clock-wise case also. The waveforms for both the cases are shown below.

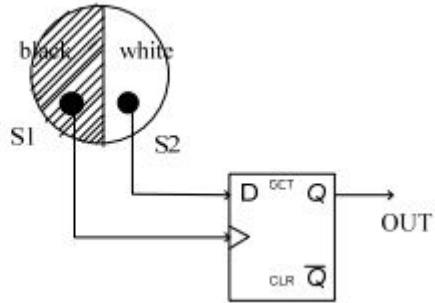
(a) For Clock-wise case the waveforms are:



(b) For anti-clock wise, the waveforms are:



These two wave forms are connected to DFF as shown in the following figure. That is S1 to the clock and S2 to the D input.



If you observe the above waveforms, for clock-wise direction, whenever there is a rising edge of S1, S2 is always 0. For anti-clock wise direction, at the rising edge of S1, S2 is 1 always.

That is OUT = 1 in the above circuit indicates the clock-wise rotation and OUT = 0 indicates anti-clockwise rotation.

Q11) Give the characteristic table and characteristic equation for J-K Flip-flop?

Answer:

The characteristic table for J-K flip flop is:

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Q'(t)

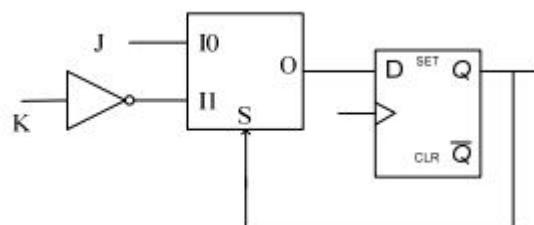
To get the characteristic equation, we can use K-Map with 3 inputs, J,K and Q(t) and obtain,

$$Q(t+1) = J Q(t)' + K' Q(t)$$

Q12) Construct a J-K flip flop using a DFF, 2:1 Mux and an-inverter?

Answer:

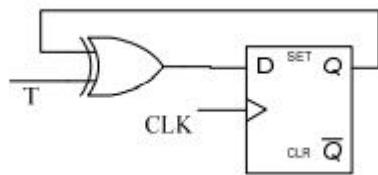
The complete design is shown here. The catch here is to use Q as select line. You can observe the cofactors of Q(t+1) with respect to J,K and Q(t). Using J or K as the select line with 2:1 mux will not do.



Q13) Implement T-flip flop from D-flip flop?

Answer:

T-flip flop using DFF:

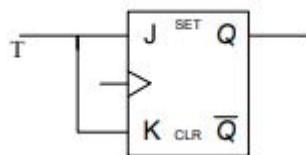


Q14) Show how to convert J-K flip flop into (a) T-flip flop (b) D-flip flop?

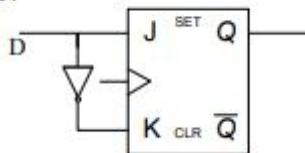
Answer:

T-flip flop using DFF:

(a) TFF:



(b) DFF:



Q15) What is excitation table?

Answer:

During the design process we usually know the transition from present state to the next state and wish to find the flip-flop input conditions that will cause the required transition. For this reason we will need a table that lists the required inputs for a given change of state. Such a list is called the excitation table.

Q16) Write the excitation table for T-flip flop?

Answer:

The excitation table for T flip flop is shown below.

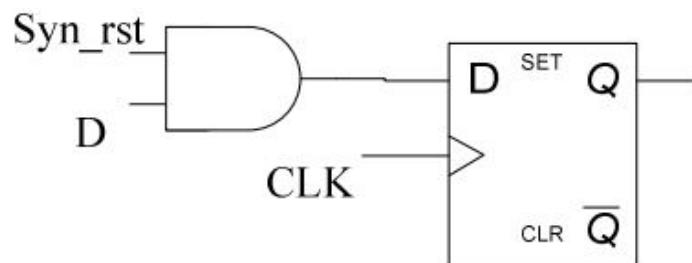
$Q(t)$	$Q(t+1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

(adsbygoogle = window.adsbygoogle || []).push({});

Q17) Draw the circuit for a D flip flop with Synchronous Reset?

Answer:

Synchronous reset will clear output only at clock edge unlike asynchronous reset. At clock edge, if $\text{syn_rst} = 0$, output will be 0 otherwise output will be D. So we just need an AND gate before the DFF as shown in the figure.



Q18) Which flip-flop can be used as delay switch?

Answer:

As DFF transmits the data as it is to the output, it can be used to provide one clock delay.

Q19) Which flip-flop can be used as toggle switch?

Answer:

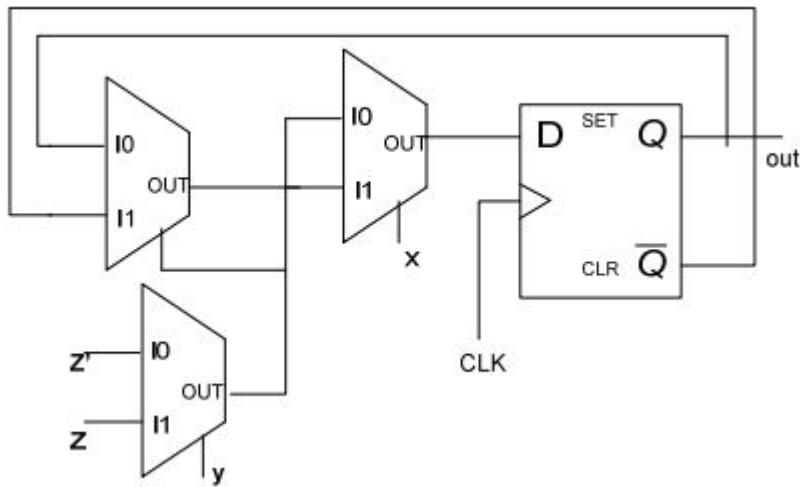
In TFF, if $T=1$, output just toggles between 1 and 0. So TFF can be used as toggle switch.

Q20) Using DFFs and minimum no. of 2×1 Mux, implement the following XYZ flip-flop.

Answer:

From the given characteristic table, it is clear that if $X=0$, $Q(t+1) = Y \text{ XNOR } Z$. If $X=1$ and if $(Y \text{ XNOR } Z) = Q(t)$, else $Q(t)'$. So we need two 2:1 mux to generate $Y \text{ XNOR } Z$. One to select $Q(t)$ and $Q(t)'$ and one more to select between $X=0$ case and $X=1$ case.

Total we need 4 2:1 mux. The design is shown here except the generation of Z' .



Q21) A AB flip flop has 4 operations: clear to 0, no change, compliment and set to 1, when inputs A and B are 00, 01, 10 and 11 respectively. Derive the characteristic equation?

Answer:

The characteristic table is shown below:

A	B	$Q(t+1)$
0	0	0
0	1	$Q(t)$
1	0	$Q(t)'$
1	1	1

The characteristic equation can be derived as,

$$Q(t+1) = A Q(t)' + B Q(t)$$

Q22) Give the excitation table for the AB flip flop described in Q20?

Answer:

The excitation table for AB flip flop is,

$Q(t)$	$Q(t+1)$	A	B
0	0	0	X
0	1	1	X
1	0	X	0
1	1	X	1

Q23) Show how the AB flip-flop can be converted to a D flip-flop?

Answer:

To get D from flip-flop from AB flop, just connect A=B=D.

We can prove this from the characteristic equation ,

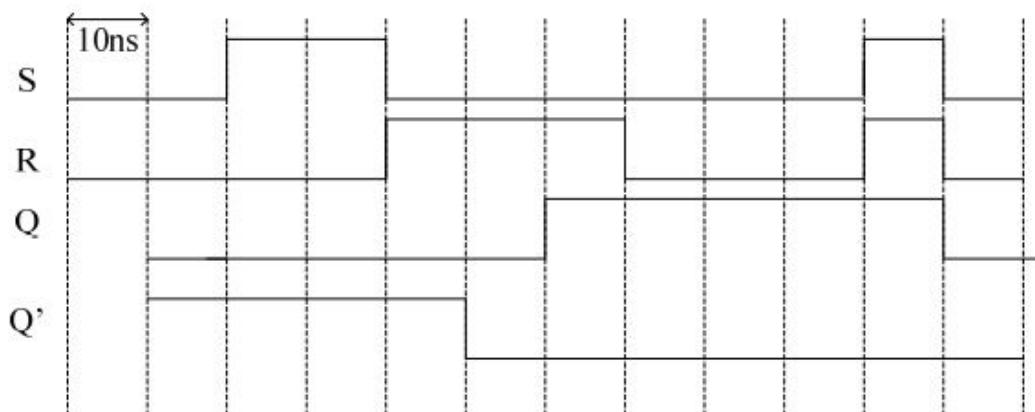
$$Q(t+1) = D \quad Q(t)' + D \quad Q(t) = D \quad (1) = D$$

Q24) Draw the output waveforms Q and Q' for a NOR based S-R Latch for the S and R waveforms shown in the following figure if each NOR gate has a delay of 10ns.

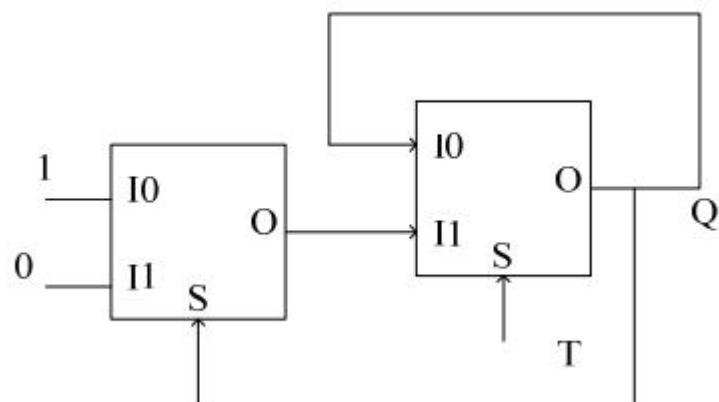
Answer:

For NOR based S-R Latch, there is no change in output for S=R=0, 0 for S=1,R=0 and 1 for S=0 and R=1. The waveforms are drawn using this. (Delay of NOR gate = 10ns).

Assume that initially Q = 0 and so Q' = 1.



Q25) Design a TFF using only 2:1 Muxes?

Answer:

Q26) In C-N (Change-No change) flip flop, there won't be any change in output as long as N is 0, irrespective of C. If N=1, then if C = 0 output will change to zero else if C =1 output will be the compliment of previous output.

- (a) Write the characteristic table ?
(b) Design this using J-K flip-flop?

Answer:

(a) The characteristic table for C-N flip flop is,

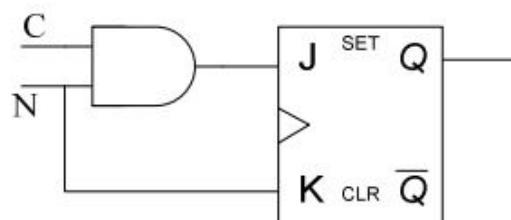
C	N	Q(t+1)
0	0	Q(t)
0	1	0
1	0	Q(t)
1	1	Q'(t)

The characteristic equation can be derived as, $Q(t+1) = (CN) Q(t)' + N' Q(t)$

(b) Comparing the characteristic equation of C-N flip flop with that of J-K flip

(b) Comparing the characteristic equation of C-N flip flop with that of J-K flip flop ($Q(t+1) = J Q(t)' + K' Q(t)$), we will get,

The J-input = CN and K-input = N



Finite State Machines – Synchronous Sequential design

Q1) Give the State Machine for Serial 2's complementer. Draw the complete design for the same using DFF?

Answer:

State Machine for Serial 2's complementer:

Logic: Starting from LSB, retain all the bits till first one has occurred including the first one and then complement all the following bits.

State Definition:

State a : No one has occurred

State b : After first one has occurred

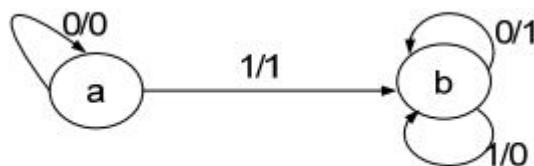
State Diagram:

(a) The characteristic table for C-N flip flop is,

C	N	Q(t+1)
0	0	Q(t)
0	1	0
1	0	Q(t)
1	1	Q'(t)

The characteristic equation can be derived as, $Q(t+1) = (CN)' Q(t) + N' Q(t)$

(b) Comparing the characteristic equation of C-N flip flop with that of J-K flip



Design using DFF:

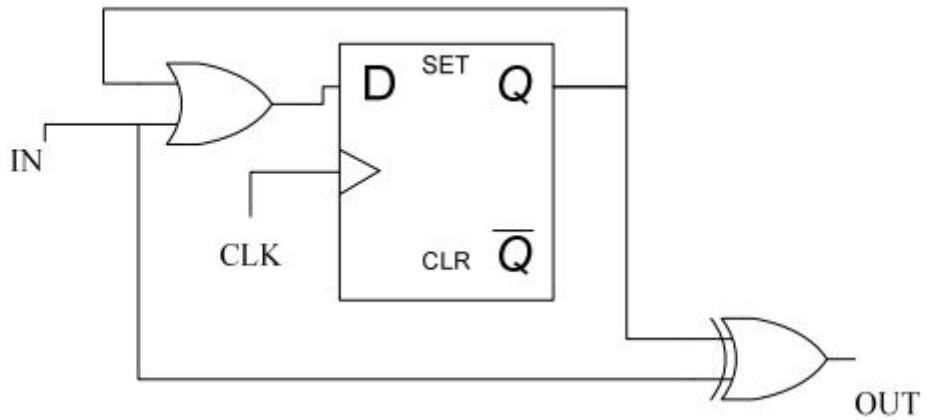
PS	INPUT	NS	OUTPUT
0(a)	0	0(a)	0
0(a)	1	1(b)	1
1(b)	0	1(b)	1
1(b)	1	1(b)	0

(a) The characteristic table for C-N flip flop is,

C	N	Q(t+1)
0	0	Q(t)
0	1	0
1	0	Q(t)
1	1	Q'(t)

The characteristic equation can be derived as, $Q(t+1) = (CN)' Q(t) + N' Q(t)$

(b) Comparing the characteristic equation of C-N flip flop with that of J-K flip



Answer:

(a) INPUT Sequence : 0 0 1 0 1 0 1 1 0 1 1 0 1 1 0 1 1 1

OUTPUT Sequence : 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1 0 1

(b) Note that the OUTPUT is 1 if the input (that has arrived till now) contains odd number of 1's. So the FSM shows the functionality of serial odd parity indicator.

(c) The state table is as follows:

PS	INPUT	NS	OUTPUT
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

The state table shows that $NS = OUTPUT = PS \oplus INPUT$

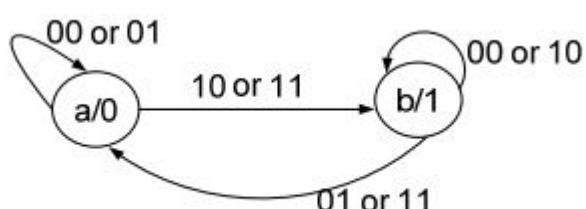
This is same as the characteristic equation of T- Flip flop.

Q3) Give the state transition diagram for J-K flip flop?

Answer:

State transition diagram for J-K Flip-flop:

The first bit of the two bits is J and the other is for K.



Answer:

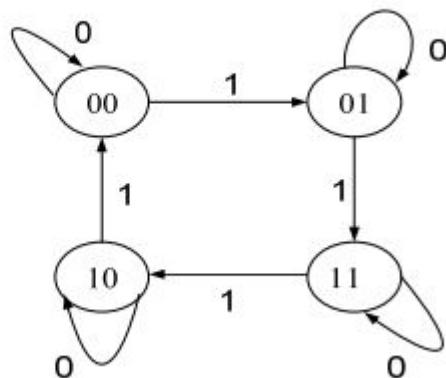
$$A(t+1) = x'A + xB$$

$$B(t+1) = xA' + x'B$$

(a) State Table:

Present State		x	Next State	
A	B		A(t+1)	B(t+1)
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	1
1	0	0	1	0
1	0	1	0	0
1	1	0	1	1
1	1	1	1	0

(b) State Diagram:

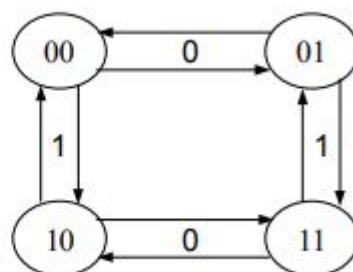


(c) Functionality:

This circuit works as enable based gray code counter. If $X=1$, the transition takes place in gray code counter and if $X=0$, there won't be any change in the present state.

Answer:

One of the possible FSMs is shown below:



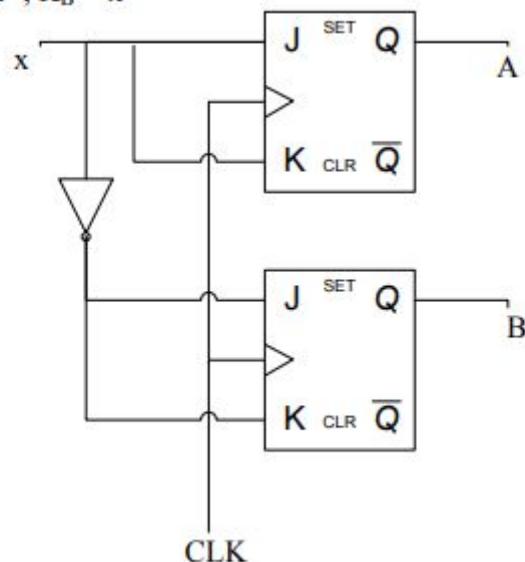
State table and flip flop inputs:

Present State		INPUT	Next State		Flip Flop inputs			
A	B	x	A	B	J _A	K _A	J _B	K _B
0	0	0	0	1	0	X	1	X
0	0	1	1	0	1	X	0	X
0	1	0	0	0	0	X	X	1
0	1	1	1	1	1	X	X	0
1	0	0	1	1	X	0	1	X
1	0	1	0	0	X	1	0	X
1	1	0	1	0	X	0	X	1
1	1	1	0	1	X	1	X	0

Design using J-K Flip flop:

The simplification using K-Map gives,

$$\begin{aligned}J_A &= x, K_A = x, \\J_B &= x', K_B = x'\end{aligned}$$



Q6) What is Moore model & Mealy model? Explain.

Answer:

A state machine consists of set of states, initial state, input symbols and transition function that maps input symbols and current state to next state.

Mealy machine: machines having outputs associated with transition of input and the current state. So Mealy machines give you outputs instantly, that is immediately upon receiving input, but the output is not held after that clock cycle.

Moore machine: machines having outputs associated with only states, that is the outputs are the properties of states themselves. The output will vary only after the states are varied. So the changes in the output will be only during clock transitions and the outputs are held until you go to some other state.

Advantages and Disadvantages: In Mealy as the output variable is a function of both input and state, changes of state of the state variables will be delayed with respect to changes of signal level in the input variables, there are possibilities of glitches appearing in the output variables. Moore overcomes glitches as output is dependent on only states and not the input signal level.

Q7) Give the State Machine for detecting the sequence “1010” from a serially coming data.

Answer:

State Machine for detecting “1010”:

Logic: Check for the bit pattern 1010. The end “10” has to be reused for next pattern.

State Definition:

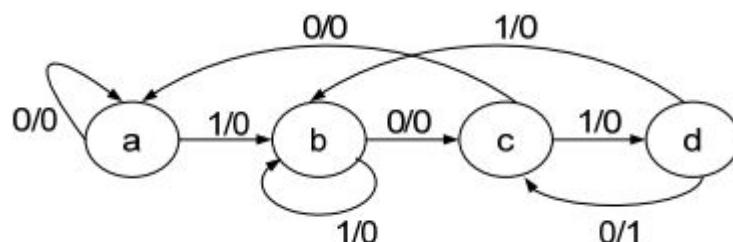
State a : No 1 detected state

State b : One 1 detected state

State c : 10 detected state

State d : 101 detected state

State Diagram:



Q8) Repeat Q7 for Non-overlapping case.

Answer:

State Machine for detecting “1010”:

Logic: Check for the bit pattern 1010. The end “10” can't be reused. So after detection of one pattern, just go to initial state. (Here it is State a).

State Definition:

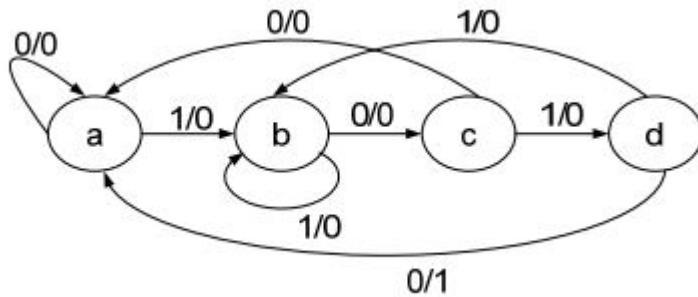
State a : No 1 detected state

State b : One 1 detected state

State c : 10 detected state

State d : 101 detected state

State Diagram:



Q9) Draw the state diagram to output a "1" for one cycle if the sequence "0110" shows up and the leading 0s cannot be used in more than one sequence.

Answer:

State Machine for detecting “0110”:

Logic: Check for the bit pattern 0110. The end “0” can’t be reused. So after detection of one pattern, just go to initial state. (Here it is State a). That is non overlapping case.

State Definition:

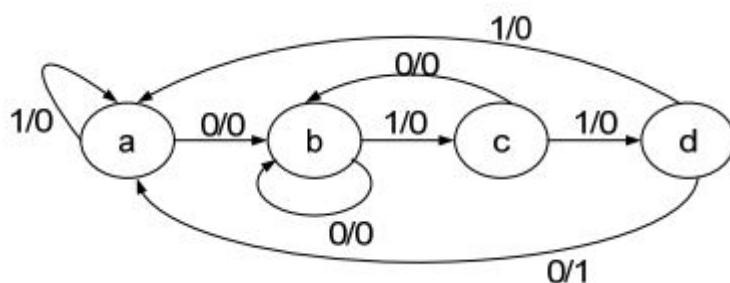
State a : No 0 detected state

State b : atleast One 0 detected state

State c : 01 detected state

State d : 011 detected state

State Diagram:



Q10) Describe a finite state machine that will detect three consecutive coin-tosses (of one coin) that result in heads.

Answer:

State Machine for detecting “111”:

Logic: If we represent Head with logic 1 and tail with logic 0, Checking for 3 consecutive heads is nothing but pattern matching for “111” (overlapping)

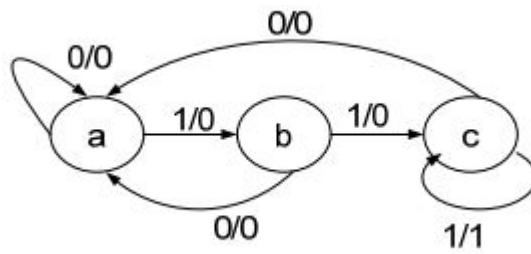
State Definition:

State a : No 1 detected state

State b : One 1 detected state

State c : More than Two 1's detected state

State Diagram:



Q11) Reduce the following state table:

Answer:

To remove a state, we need to have another state with the same next state values and output values. If we observe the given state table, state g has all entries same as those of state a. So state g can be replaced with a everywhere. Once g is replaced with a, all the entries of f are same as those of c. Thus, replacing f with c, makes state d same as state b. So with all these changes the reduced state table is shown below:

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
a	a	b	0	0
b	e	c	0	1
c	a	b	0	1
e	a	c	0	0

Q12) Starting from state a, write the next state and output values for the input sequence: 01010101001 for both Original and reduced state table.

Answer:

(a) For Original:

The input sequence: 0 1 0 1 0 1 0 1 0 0 1
 Next state : a a b e f g b e a a a b
 Output sequence : 0 0 0 0 0 0 0 0 0 0 0

(b) For Reduced:

The input sequence: 0 1 0 1 0 1 0 1 0 0 1
 Next state : a a b e c a b e a a a b
 Output sequence : 0 0 0 0 0 0 0 0 0 0 0

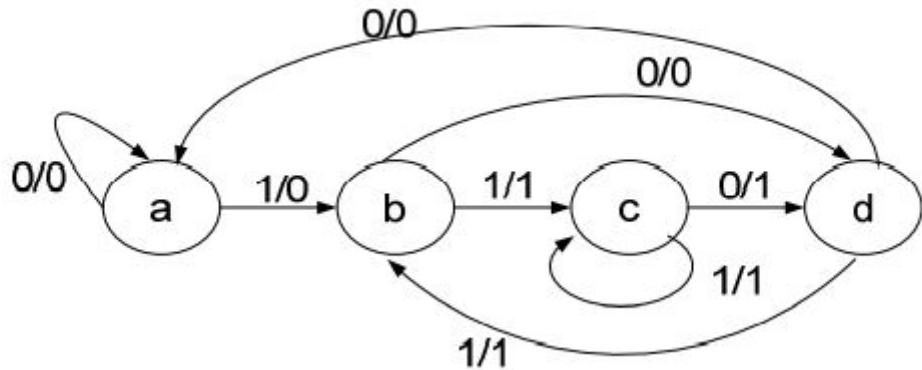
Answer:

State Machine for detecting more than one 1 in last 3 samples:

Logic: Check for the patterns 011, 101, 110 or 111. These 4 patterns have more than one

1.

State Definition: State a : No 1 detected state, continuous 0s State b : One 1 detected state, “01” or “1” detected state State c : Atleast two 1s detected state, “011” or “111” detected state State d : “010” or “001” detected state



Answer:

State Machine for detecting “101” in last 4 samples:

Logic: Possible patterns are: 0101,1101,1010,1011 (overlapping)

State Definition:

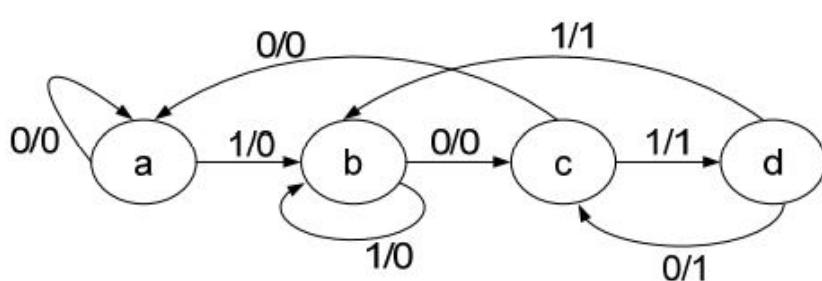
State a : Continuous 0s

State b : Atleast one 1, that is “01” or “111....”

State c : 010 detected state

State d : 0101 or 1101 detected state

State Diagram:



Answer:

State Machine for detecting “101” in last 4 samples:

Logic: Check for the patterns 101 or 010 in last 3 samples.

State Definition:

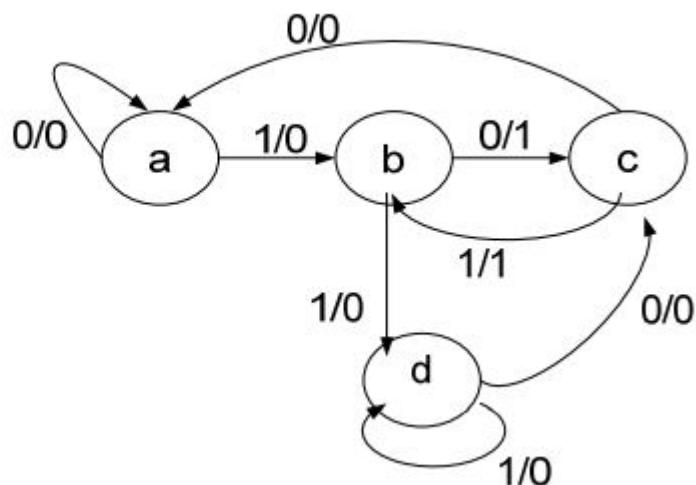
State a : State a : No 1 detected state, continuous 0s

State b : One 1 detected state, “01” or “1” detected state

State c : "10" detected state

State d : continuous 1's detected state

State Diagram:



Answer:

State Machine for eliminating short length pulses:

Logic: The 1 after two successive 0's will be made 0. Similarly the zero after two successive 1's will be made 1. If you are continuous 1s state, minimum 2 0's zeros are needed to switch to continuous 0s state and vice versa.

State Definition:

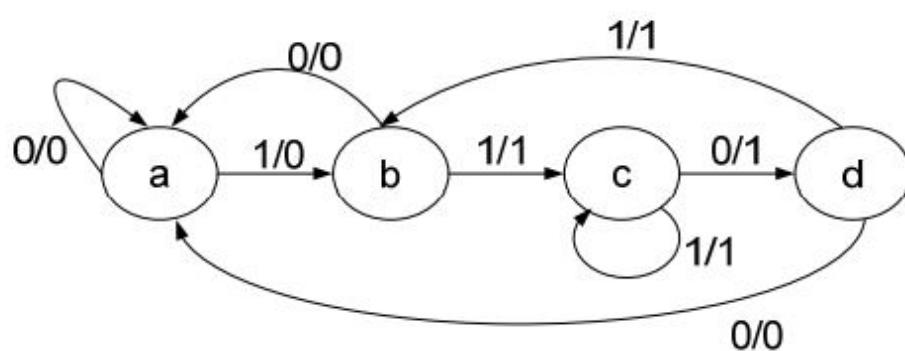
State a : Continuous 0s

State b : One 1 in between 0s

State c : Continuous 1s

State d : One 0 in between 1s

State Diagram:



Q17) What is one-hot method? List the advantages and disadvantages?

Answer:

One-Hot encoding of FSM uses one flip-flop per state. Only one flip-flop is allowed to have 'logic 1' at anytime. For example, for a five state FSM, the states are "00001", "00010", "00100", "01000", "10000". All other states are illegal. One-Hot encoding trades combinational logic for flip-flops. One hot reduces the next state and output logic complexity. Its good for 'flip-flop' rich implementation technologies. Because the combinational logic is reduced, the length of the critical path can be reduced resulting in a faster FSM. Speed increase is more significant for larger finite state machines. The disadvantage is we end up in using more number of flops.

Q18) Show the state assignment using Johnson's method for a FSM of 6 states?

Answer:

Johnson's method: 000,001,011,111,110,100

Q19) How many flip flops are needed to design a FSM having N states if the state assignment is done by using

- (a) Binary
- (b) Gray
- (c) One hot
- (d) Johnson

Answer:

- (a) $\log_2 N$
- (b) $\log_2 N$
- (c) N
- (d) $N/2$

Answer:

State Machine for identifying whether the 1's and 0's are even or odd:

Logic: The only 4 possibilities are even-even, even-odd, odd-even, odd-odd. So initial state will be even-even as no 1 or no 0. Now if 1 comes it will be even 0's odd 1. Similarly if 0 comes it will be odd 0's even 1's. So the state transition will take place accordingly.

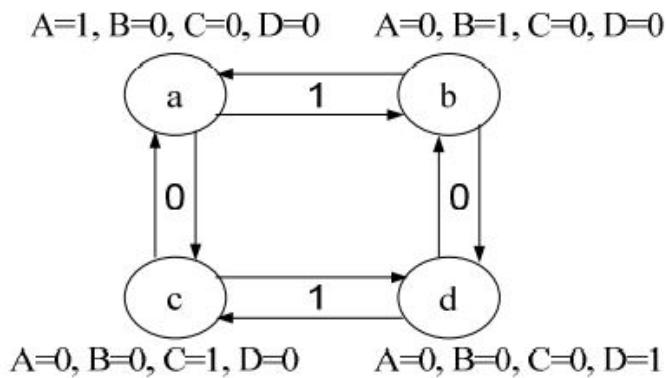
State Definition:

- State a : Even 0's Even 1's
- State b : Even 0's Odd 1's
- State c : Odd 0's Even 1's
- State d : Odd 0's Odd 1's

Outputs:

- In state a, outputs are: A = 1, B = 0, C = 0, D=0.
- In state b, outputs are: A = 0, B = 1, C = 0, D=0.
- In state c, outputs are: A = 0, B = 0, C = 1, D=0.
- In state d, outputs are: A = 0, B = 0, C = 0, D=1.

State Diagram:



Q21) For the above problem, which method is more suitable for state assignment?

Answer:

One hot method.

a : 1000, b : 0100, c : 0010, d : 0001

These values are nothing but the four outputs that are needed. So it reduces the output logic complexity.

Q22) Draw the state diagram for a circuit that outputs a "1" if the aggregate serial binary input is divisible by 5. For instance, if the input stream is 1, 0, 1, we output a "1" (since 101 is 5). If we then get a "0", the aggregate total is 10, so we output another "1" (and so on).

<u>Input</u>	<u>Sequence</u>	<u>Value</u>	<u>Output</u>
1	1	1	0
0	10	2	0
1	101	5	1
0	1010	10	1
1	10101	21	0

Answer:

State Machine to detect whether the serial binary number is divisible by 5 or not:

Logic: From the given example we can notice that the data is coming from MSB side. And the possible reminders are 0,1,2,3,4. So we need to have five states each for value and output is made 1 if we reach state 'a', reminder 0 state.

State Definition:

State a : Reminder 0

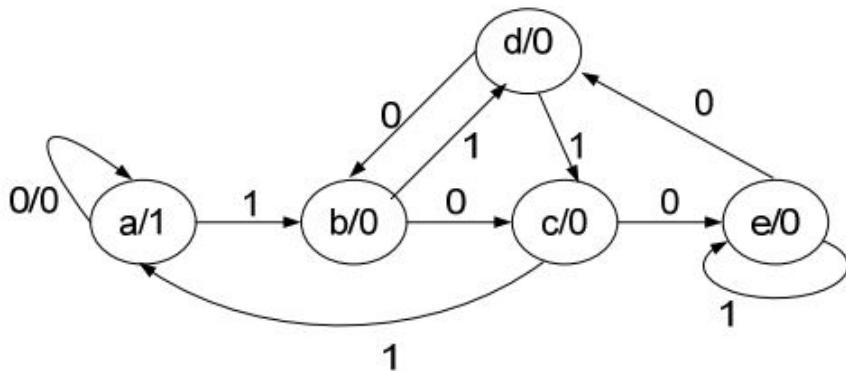
State b : Reminder 1

State c : Reminder 2

State d : Reminder 3

State e : Reminder 4

State Diagram:



Q23) Draw the FSM for checking whether the two inputs P and Q have same value for the last three continuous samples?

Answer:

State Machine to check whether the two inputs have same value for last 3 samples:

Logic: As there are two inputs, at each state we will have four possible transitions based on the two input combinations. If $P=Q=1$ or $P=Q=0$ go to next state, otherwise go back to initial state.

State Definition:

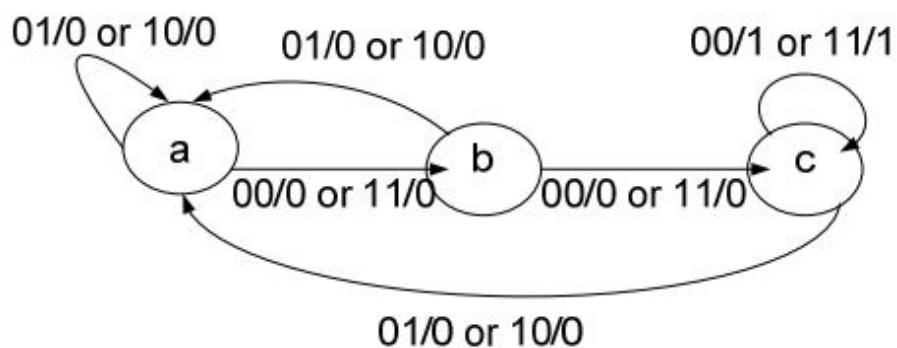
State a : P is not equal to Q

State b : $P=Q$ for last 1 sample

State c : $P=Q$ for Atleast last 2 samples

State Diagram:

Out of the two bits that are shown on the arrows, first 1 is for input A and second one is for input B.



Q24) If the number of 1s in inputs X and Y since reset is multiple of 4, the output is 1 and 0 otherwise. Give the FSM required for designing of the sequential circuit?

Answer:

State Machine with two inputs- Number of 1's together multiples of 4:

Logic: As there are two inputs, at each state we will have four possible transitions based on the two input combinations. Just count the number of 1's in X and Y together and check for the remainder if that number is divided by 4. The possible reminders are 0,1,2,3. The output will be 1 if the remainder is 0.

State Definition:

State a : Reminder 0

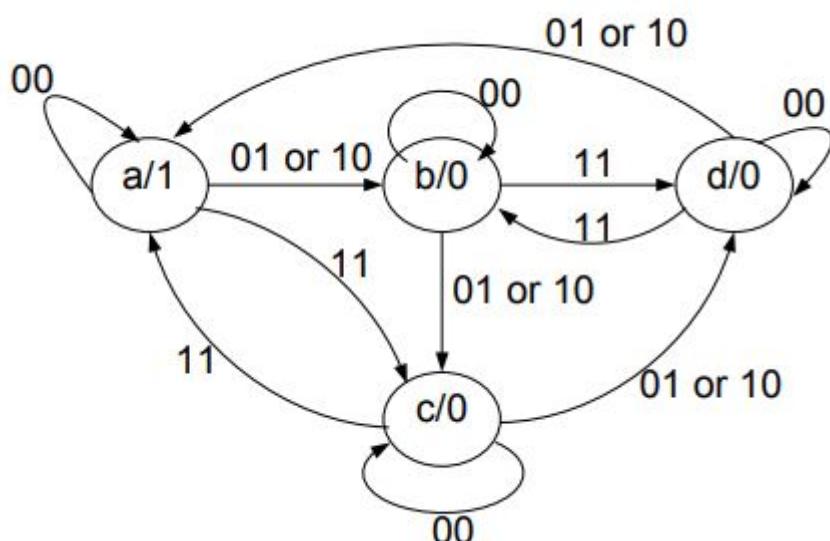
State b : Reminder 1

State c : Reminder 2

State d : Reminder 3

State Diagram:

Out of the two bits that are shown on the arrows, first 1 is for input A and second one is for input B.



Answer:

State Machine with two inputs:

Logic: As there are two inputs, at each state we will have four possible transitions based on the two input combinations. The output will be one for the first time, if A has same value in the last two clock ticks. After that, output can be made 1 either by condition A or condition B. S for initial transitions B will be don't care. But once output is 1, that if you are in state d or e, as long as B is 1, the transition will be between d and e only and the output is also 1.

State Definition:

State a : Initial state

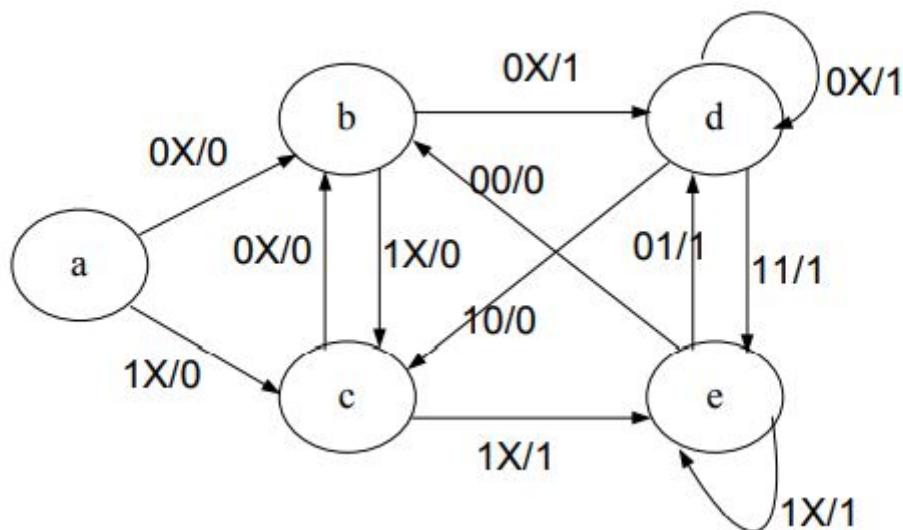
State b : A is 0 once (B can be either 1 or 0)

State c : A is 1 once (B can be either 1 or 0)

State d : A has same value for last two samples and it is equal to logic 0

State e : A has same value for last two samples and it is equal to logic 1

State Diagram: Out of the two bits that are shown on the arrows, first 1 is for inp



Answer:

State Machine with two inputs:

a

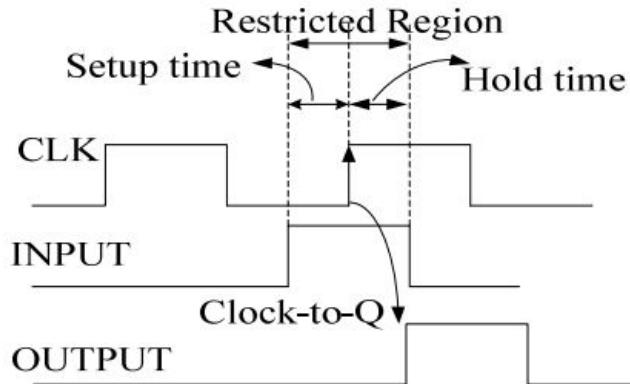
Setup Time and Hold Time

Q1) Define (a) setup time (b) hold time (c) clock to Q delay.

Answer:

- (a) Setup time: Setup time is the minimum amount of time the data signal should be held steady before the clock event so that the data is reliably sampled by the clock.
- (b) Hold time: The hold time is the minimum amount of time the data signal should be held steady after the clock event so that the data is reliably sampled by the clock.

- (c) Clock to Q delay: The clock to Q delay is the amount of the propagation time required for the data signal to reach the output (Q) of the flip flop after the clock event.



Q2) Which of the following flip flops can work at maximum frequency?

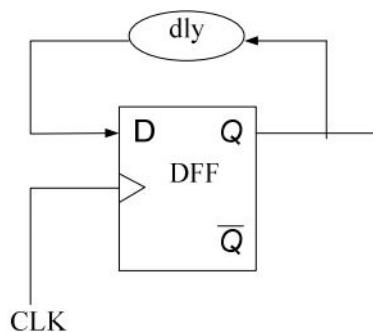
	FF1	FF2	FF3
Clock to Q delay(ns)	5	6	8
Setup time(ns)	3	4	2
Hold time(ns)	2	1	1

Answer:

For a single flip flop, lesser the clock-to-Q delay, more the operating frequency. However, the maximum frequency of operation may be limited by the configuration in which the flip flop is connected. This will be clear in the later parts of the chapter.

Among the 3 flops, the first one, FF1 has less clock to Q delay. So it can operate at maximum frequency which is given by $1/5\text{ns} = 200\text{MHz}$

Q3) Derive the maximum frequency of operation for the following circuit in terms of T_{cq} , T_{su} and T_h of the flip flop?

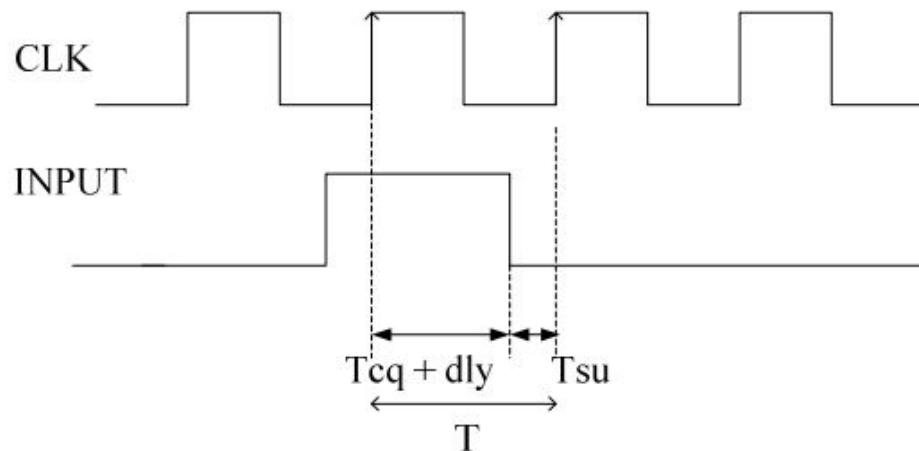


Answer:

After the posedge of the clock, the output will change after a delay of T_{cq} . The input of the flop will change after further delay of "dly". It should be available before the T_{su} of the

flop.

So the $T \geq T_{cq} + T_{su} + dly$. The same thing is illustrated in the following waveform.



Q4) For the above configuration with $dly = 0$, which of the flip flops that are shown in Q2, can be used if the available clock period is (a) 5ns (b) 8ns (c) 15ns

Answer:

For FF1, $T_{su} + T_{cq} = 3 + 5 = 8\text{ns}$

For FF2, $T_{su} + T_{cq} = 6 + 4 = 10\text{ns}$

For FF3, $T_{su} + T_{cq} = 8 + 2 = 10\text{ns}$

As $dly = 0$, $T_{su} + T_{cq} \leq T$

(a) $T = 5\text{ns}$, None of the flip flops has $T_{su} + T_{cq} \leq T$, so no one can be used.

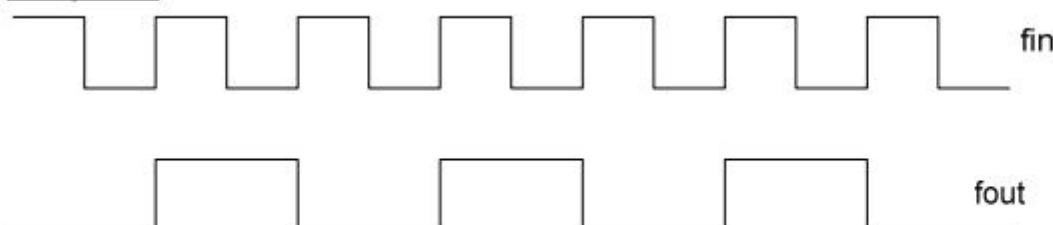
(b) $T = 8\text{ns}$, FF1 can be used

(c) $T = 15\text{ns}$, Anyone can be used

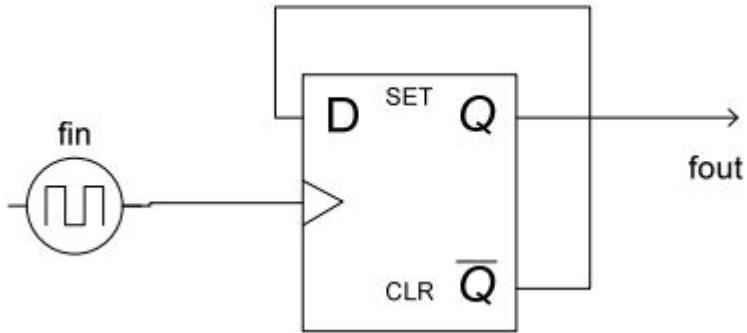
Q5) Design a circuit for clock frequency divided by 2 using DFF. Given the following information, find the maximum clock frequency that the circuit can handle? $T_{\text{setup}} = 6\text{ns}$, $T_{\text{hold}} = 2\text{ns}$ and $T_{\text{propagation}} = 10\text{ns}$

Answer:

Waveforms:



Design:



Using the same equation , $T \geq T_{cq} + T_{su}$, $T \geq 6 + 10$. So $T \geq 16\text{ns}$.

The maximum clock frequency = $1/16\text{ns} = 62.5\text{MHz}$

Q6) Is there any hold violation in the above circuit? When will the hold violation occur in a given circuit and how can it be solved in circuit level? Describe in detail.

Answer:

There are no hold violations in the above circuit. If the hold time is greater than the propagation delay then there will be hold violation for the above circuit. In that case, buffers (even number of inverters) will be used in the feedback path in order to delay the signal in reaching back to the input.

Answer:

- (a) $\text{Thold} \leq T_{cq} + \text{dly}$. But here, $2\text{ns} > 1.5 + 0.5 = 1.7\text{ns}$. So there is a hold violation in the above circuit.
- (b) $\text{dly} \geq \text{Thold} - T_{cq} = 2 - 1.5 = 0.5\text{ns}$
- (c) The delay of the clock buffer will not effect the maximum frequency of operation of the circuit.

Q8) What is clock skew? Explain.

Answer:

Clock-skew: Clock skew is a phenomenon in synchronous circuits in which the clock signal (sent from the clock circuit) arrives at different components at different times. This is typically due to two causes:

1. The first is a material flaw, which causes a signal to travel faster or slower than expected.
2. The second is distance: if the signal has to travel the entire length of a circuit, it will likely (depending on the circuit's size) arrive at different parts of the circuit at different times.

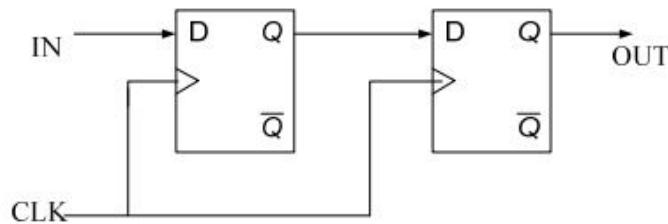
Skew is only meaningful between adjacent pairs of registers, not between any pair of registers in a clock domain.

Q9) Can hold time be negative? Explain.

Answer:

Yes, Hold time of a flip flop can be negative. Most of the modern flip flops will have either 0 or negative hold time. Assume $Thold = -2\text{ns}$, there should not be any transitions in the input before 2ns of the clock event.

Q10) Among the flip flops that are shown in Q2, which combination can give maximum frequency of operation for the following circuit?



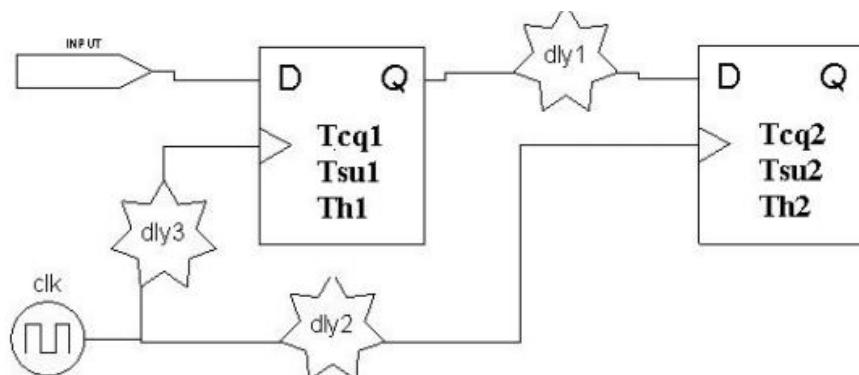
Answer:

For the given circuit, $T \geq T_{cq1} + T_{su2}$.

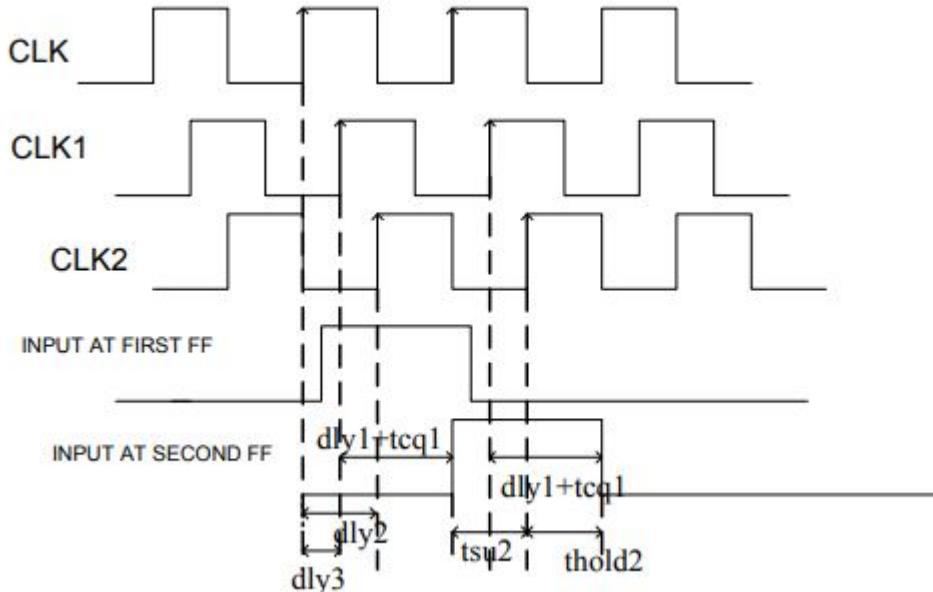
To get maximum frequency T should be less. So we should select the first flop with less clock to Q delay and second flip flop with less setup time. So FF1 and FF3 give the maximum frequency and it is equal to $1/7\text{ns} = 142.8\text{MHz}$

Q11) The following digital circuit shows two flops with a logic delay (dly1) in between and two clock buffer delays (dly2, dly3). Derive the conditions in terms of (dly1,dly2,dly3) to fix setup and hold timing violations at the input of second FF?

T_{cq} – Clock to Q delay, T_{su} -- Setup time and Th – hold time.



Answer:



The above waveforms show the CLK, CLK1 and CLK2. The input waveform at FF1 is assumed and the input of FF2 is shown accordingly with all the given delays and clock-to-Q delays. From the waveforms it is clear that, to avoid setup time violation,
 $T \geq (T_{su2} + T_{cq1} + dly1 - \text{delta})$ where $\text{delta} = dly2 - dly3$ (assuming +ve skew) From this equation we can get maximum freq of operation. To avoid hold time violation,
 $Th2 \leq T_{cq1} + dly1 - \text{delta}$

These two equations can be used as generalized equations to solve setup time/hold time problems. This works only for synch circuits. If one clock works at pos edge and other is negative edge we need to derive one more set of equations. That also we will at later section.

Answer:

(a) There is a hold time violation in the circuit, because of the feedback. T_{cq2} + AND gate delay is less than $thold2$. To avoid this , we need to use even number of inverters(buffers). Here we need to use 2 inverters each with a delay of 1ns. Then the hold time value exactly meets.

(b) In this diagram,

$$dly3 = 0$$

$$dly2 = 2\text{ns}$$

$$\text{so, delta} = 2\text{ns}$$

$$tsu2 = 3\text{ns}, tcq1 = 2\text{ns}, dly1 = 5\text{ns}$$

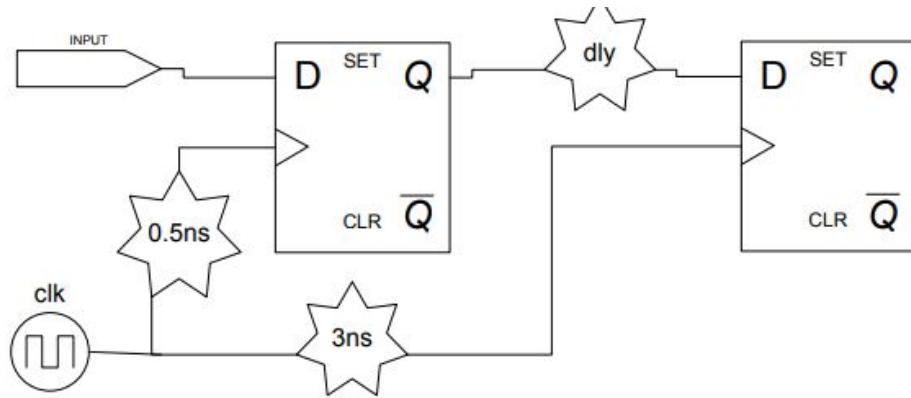
Putting all these values in Eq(1) ,

$$T \geq T_{cq1} + dly1 + Tsu2 - \text{delta}$$

$$\text{so, } T \geq 2 + 5 + 3 - 2, T \geq 8\text{ns}, f \leq 1/8$$

Max freq of operation is **125MHz**

Q13)



If both the flip flops have same clock to Q delay of 2.5 ns , setup time of 2 ns and a hold time of 1 ns , what is the maximum frequency of operation for the circuit shown in the above figure?

Answer:

$$T_{cq1} = T_{cq2} = 2.5\text{ ns}$$

$$T_{su1} = T_{su2} = 2\text{ ns}$$

$$T_{hold1} = T_{hold2} = 1\text{ ns}$$

$$\text{delta} = \text{clock_skew} = 3 - 0.5 = 2.5\text{ ns}$$

Equation for hold-violation is,

$$T_{hold} \leq \text{dly} + T_{cq1} - \text{delta} \quad (\text{Eq}(2))$$

$$1 \leq \text{dly} + 2.5 - 2.5$$

$$\text{So } \text{dly} \geq 1\text{ ns}$$

To obtain maximum freq, fixing it to lowest possible value, so $\text{dly} = 1\text{ ns}$

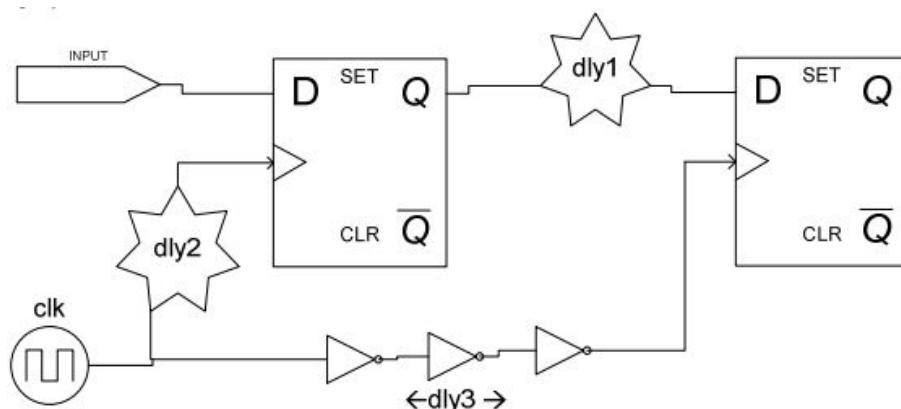
Using this value and Eqauation (1) of setup time, we can obtain max freq.

$$T \geq T_{cq1} + \text{dly} + T_{su2} - \text{delta}$$

$$T \geq 2.5 + 1 + 2 - 2.5$$

$$\text{So } T \geq 3\text{ ns} \text{ Max freq of operation} = 1/3\text{ ns} = 333.33 \text{ MHz}$$

Q14)



Repeat Q1 for the above circuit? Assume Tcq1 – Clock to Q delay, Tsu1 -- Setup time and Th1 – hold time for first FF and similarly Tcq2,Tsu2,Th2 for second FF.

Answer:

Setup time:

$$(T/2) + \text{delta} \geq \text{Tcq1} + \text{dly1} + \text{Tsu2}$$

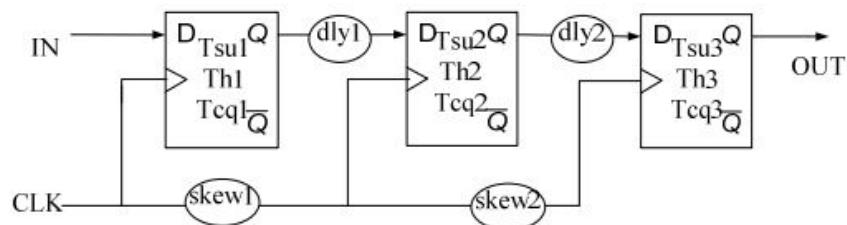
Hold time:

$$\text{Th2} \leq \text{delta} + \text{Tcq1} + \text{dly1}$$

where delta = dly3 - dly2, assuming positive skew and T is clock period.

Note: The procedure is same as that of Q11. Just draw the waveforms with proper delays, you will get above equations.

Q15) What is the maximum frequency of operation for the following configuration?



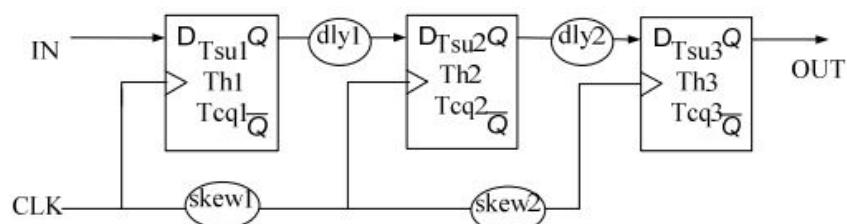
Answer:

For FF1 and FF2, $T_1 \geq (\text{Tsu2} + \text{Tcq1} + \text{dly1} - \text{skew1})$

For FF2 and FF3, $T_2 \geq (\text{Tsu3} + \text{Tcq2} + \text{dly2} - \text{skew2})$

$$T \geq \text{MAX } (T_1, T_2)$$

Q16) What is metastability? When/why it will occur? Explain how to avoid this?



Answer:

Metastable state: A unknown state in between the two known logical states is called as Metastable state.

Reason for occurrence: This will happen if the output node capacitance is not allowed to charge/discharge fully to the required logical levels. In case of flip flops, if the input changes in the restricted region, that is if there is a setup time or hold time violations, metastability will occur.

Way to avoid: To avoid this, a series of FFs is used (normally 2 or 3) which will remove the intermediate states. The extra flip flop is called the synchronizer.

Q17) For the circuit in Q11, two identical flip flops with the following data were used:
 $T_{su} = 2\text{ns}$, $T_h = -3\text{ns}$ and $T_{cq} = 5\text{ns}$. Which combination of $dly1$ and $dly2$ from the following table will give maximum frequency of operation without any violations? Given:
 $dly3 = 0$.

dly1(ns)	dly2(ns)
1	7
6	2
2	8

Answer:

Given: $T_{su} = 2\text{ns}$, $T_h = -3\text{ns}$ and $T_{cq} = 5\text{ns}$.

If hold time is negative and if its absolute value is less than T_{su} , only the setup violation equation without any modification will work. But if absolute value of hold time is more than setup time, we need to replace the setup time in the equation with hold time. The modified equation is shown below:

$$\begin{aligned} T &\geq T_{cq1} + dly1 + \max(T_{su2}, |T_h2|) \\ T &\geq 5 + dly1 + 3 \\ T &\geq 8 + dly1 \end{aligned}$$

To get maximum frequency of operation, the minimum possible $dly1 = 1\text{ns}$.

So, $T \geq 9\text{ns}$

$$F_{max} = 1/9\text{ns} = 111\text{MHz}$$

Counters and Shift Registers

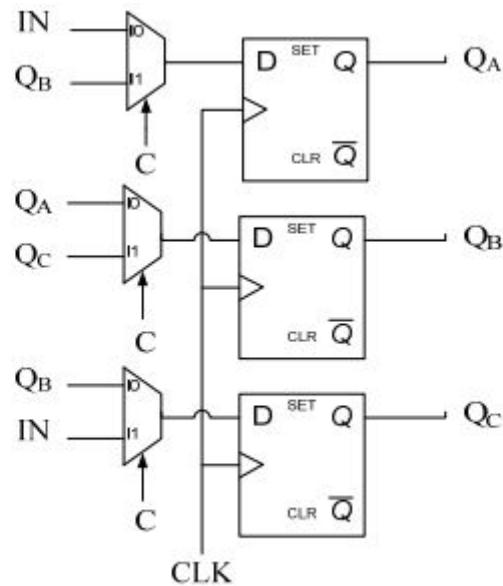
Q1) Design a 3-bit shift register using 2:1 Mux and D Flip Flops which shifts right if the control input, $C = 0$ and shifts left if $C = 1$?

Answer:

The shift register is shown below.

If C = 0, the circuit shifts from IN QA \rightarrow QB \rightarrow QC and

If C = 1, the circuit shifts from IN QC \rightarrow QB \rightarrow QA



Answer:

$$D_{next} = Q_0 \oplus Q_2 \oplus Q_3$$

(a)

Q0	Q1	Q2	Q3
1	0	0	0
1	1	0	0
1	1	1	0
0	1	1	1
0	0	1	1
0	0	0	1
1	0	0	0

(b) Q0 = 0, Q1 = 0, Q2 = 1, Q3 = 1

Q3) What is the difference between a ripple counter and a synchronous counter?

Answer:

Ripple counter is asynchronous. This means all flip flop outputs will not change at the same time. The output of one flop works as clock to the next flip flop. The state changes consequently "ripple through" the flip flops, requiring a time proportional to the length of the counter.

Where as synchronous counters will have same clock for all the flip flops. All flip flops will change the state at the same time. Design of synchronous counters is easy but needs more hardware.

Although the asynchronous counter is easier to implement, it is more "dangerous" than the synchronous counter. In a complex system, there are many state changes on

each clock edge, and some IC's (integrated circuits) respond faster than others. If an external event is allowed to affect a system whenever it occurs, a small percentage of the time it will occur near a clock transition, after some IC's have responded, but before others have. This intermingling of transitions often causes erroneous operations. What is worse, these problems are difficult to test for and difficult to foresee because of the random time difference between the events.

Q4) To count from 0 to N-1, how many flip-flops are needed?

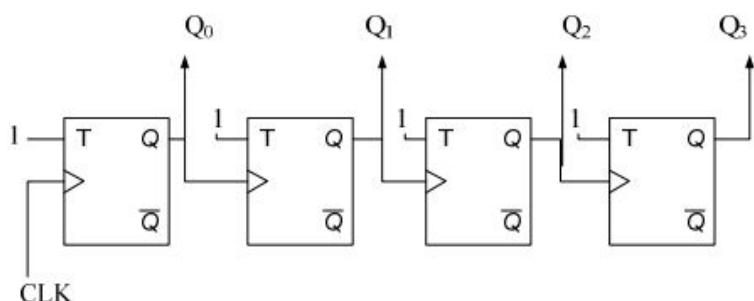
Answer:

$$\log_2 N$$

Q5) Design a 4-bit binary counter using TFFs?

Answer:

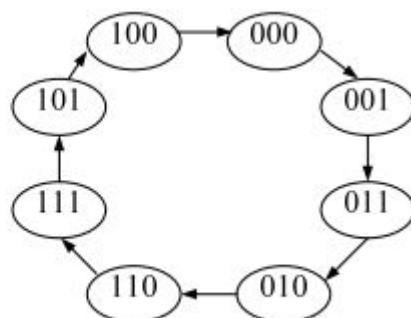
In the problem it is not clearly mentioned whether



Q6) Give the FSM for a 3-bit gray code counter?

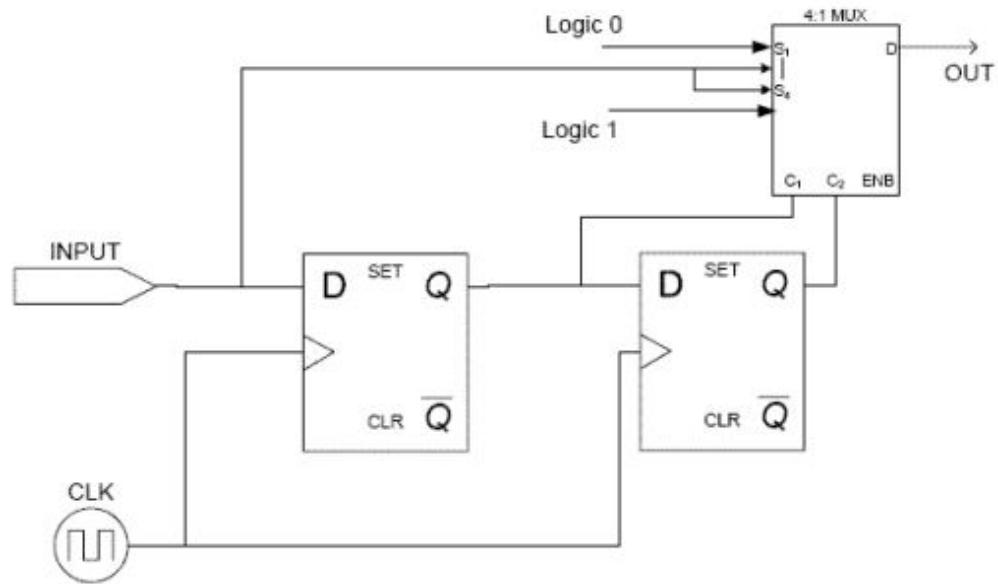
Answer:

The FSM for 3-bit gray counter is shown below. You can notice the single bit change from one state to another state.

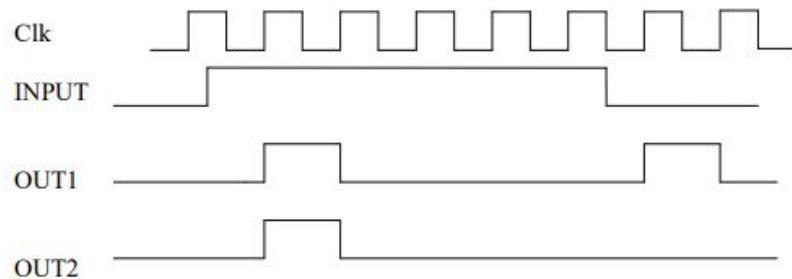


Answer:

The FSM for 3-bit gray counter is shown below. You can notice the single bit change from one state to another state.

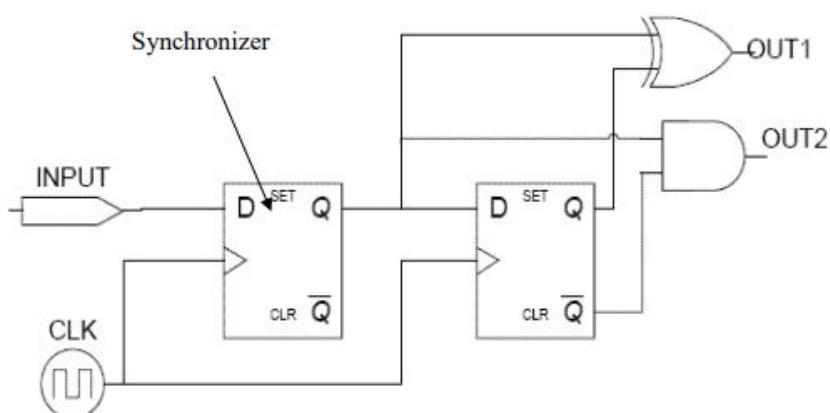


Q8) Obtain OUT1 & OUT2 from INPUT shown below? (Hint: You need a synchronizer to align INPUT with clock)

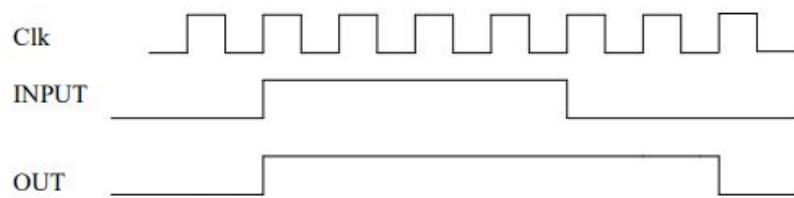


Answer:

Shift register based: The synchronizer (the first flip flop) aligns the INPUT with clock. The second flip flop delays the input by one clock. Draw the waveforms of output of first and second flip flops and then try to get the relationship between those waveforms and OUT1, OUT2. It gives the complete solution as shown below.

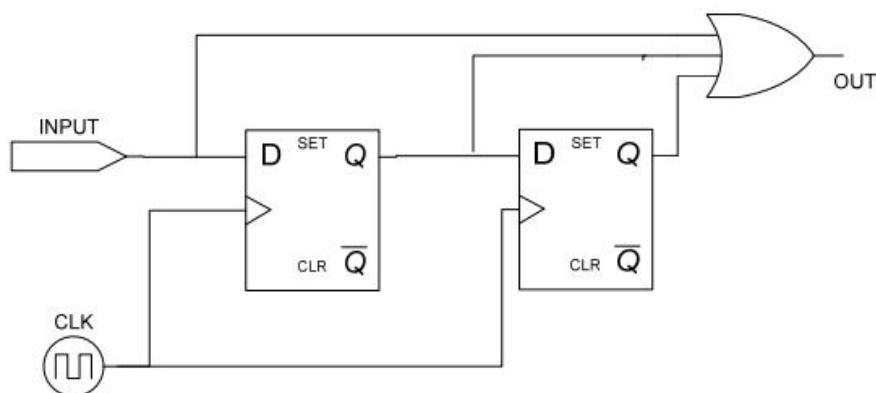


Q9) Give the circuit to extend the falling edge of the input by 2 clock pulses. The waveforms are shown in the following figure.



Answer:

Shift register based: Assumed atleast 3 clock gaps between next falling edge. Shift register of width 2 is needed.

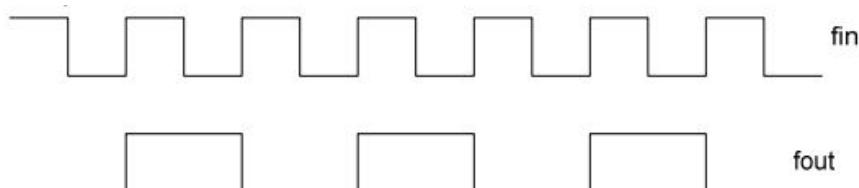


Q10) Design a frequency divide-by-2 circuit using DFF and external gates which gives (a) 50% duty cycle (b) 25% duty cycle?

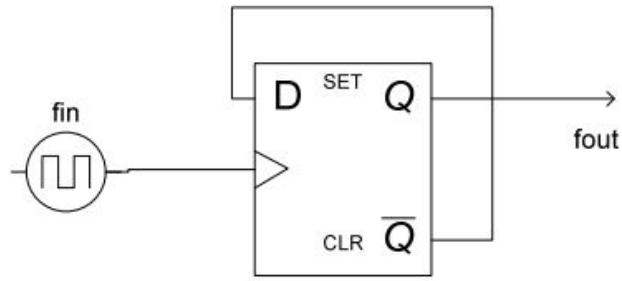
Answer:

(a) 50% duty cycle:

Waveforms:



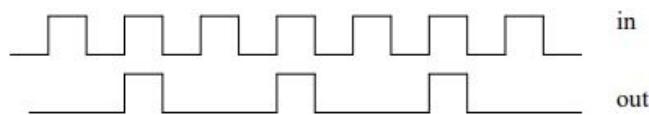
Design:



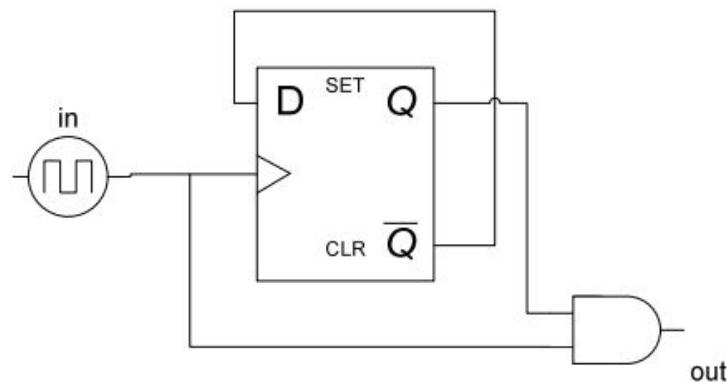
(b) 25% duty cycle:

The above circuit gives 50% duty cycle. To get 25%, we need to use an extra AND gate, which takes fin and fout as the inputs.

Waveforms:



Design:



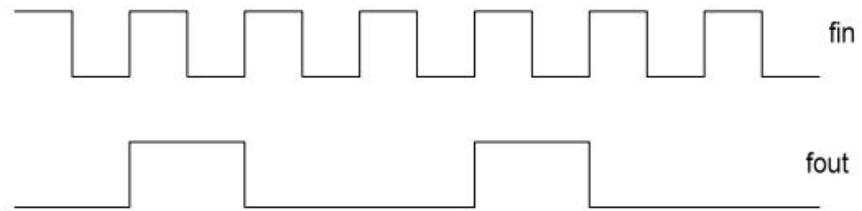
Answer:

Same as A10(b)

Q12) Design frequency divide-by-3 circuit using DFFs and external gates which gives a duty cycle of $1/3^{\text{rd}}$?

Answer:

Waveforms:



Design:

In the above problem, if you observe the waveforms, they are synchronous. So we can use FSM to design the circuit. If you observe the waveform clearly, output is 100,100,100 and so on. Assume 3 states: a,b & c. Initial state is a and output is 1 in this state. The state transition a \rightarrow b \rightarrow c \rightarrow a. Output is 1 only for state a. The state table is shown below:

PS NS O/P

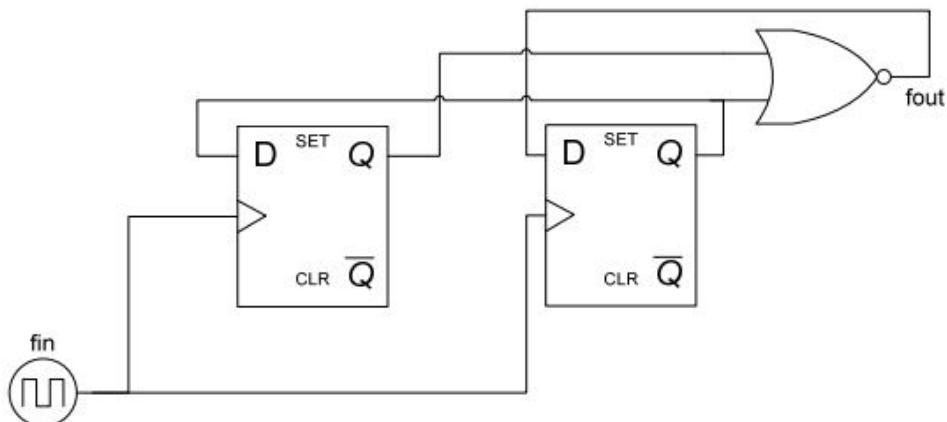
a b 1

b c 0

c a 0

With State assignment: 00-a,01-b and 10-c, we can obtain the following next state equations:

$$A(t+1) = B, B(t+1) = A \text{ NOR } B, \text{OUT} = A \text{ NOR } B$$

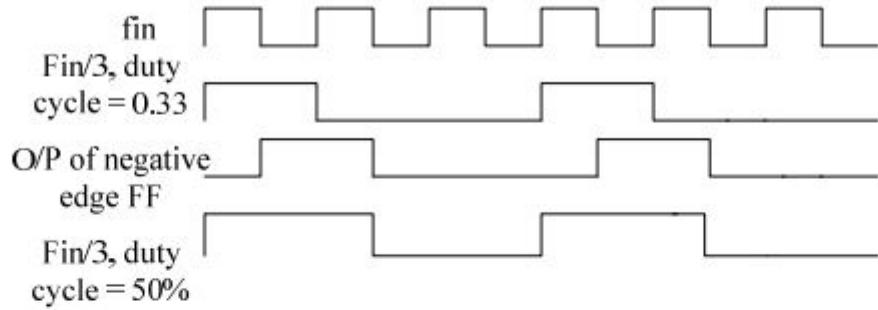


Q13) How do you change the above design to get (a) 66.67% duty cycle (b) 50% duty cycle?

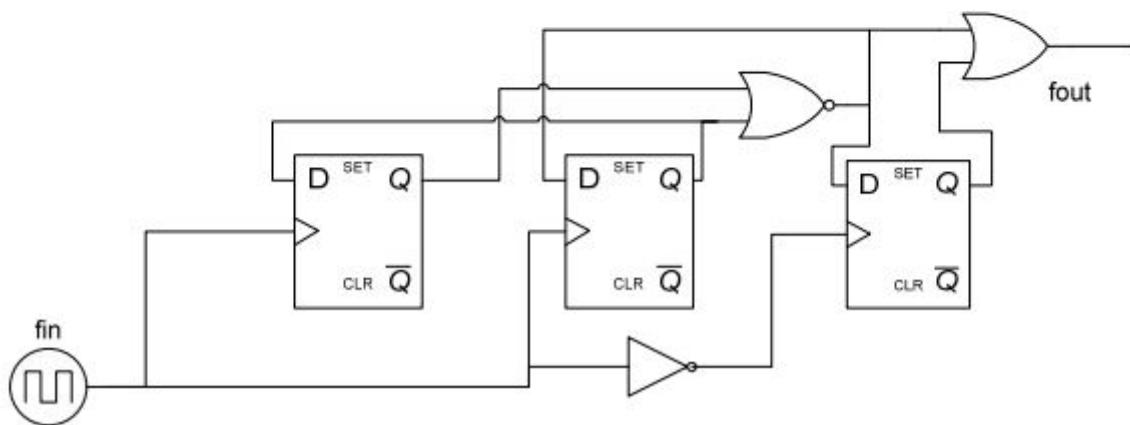
Answer:

(a) Replacing the NOR gate in the above circuit with NAND gate gives a duty cycle of 2/3. This is a 66.67% duty cycle. By observing the waveforms, we can notice that an extra flop that works at the negative edge of the 50% duty cycle is needed. The complete solution is shown below.

(i) Waveforms:



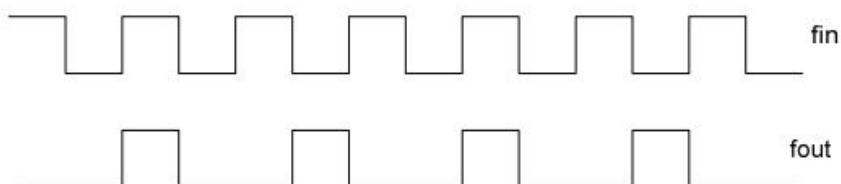
(ii) Design:



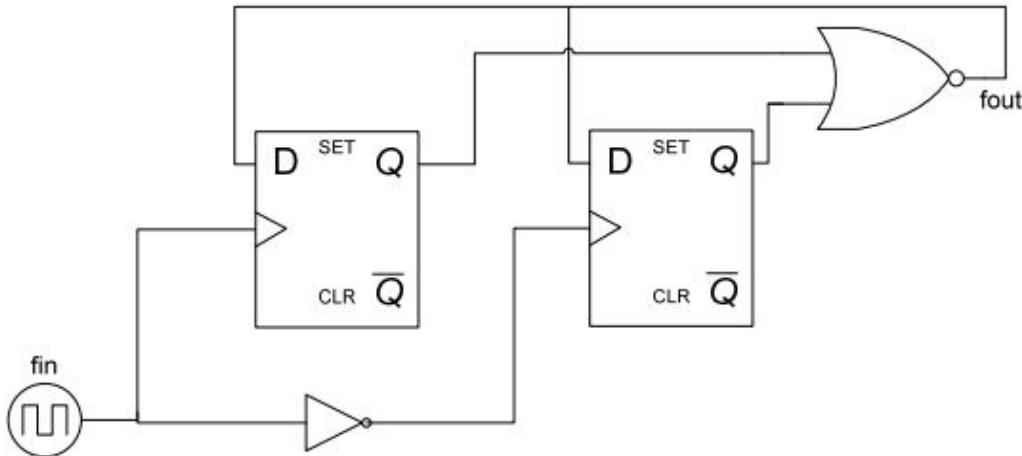
Q14) Design the Digital Circuit which gives $f_{out} = (2/3) f_{in}$

Answer:

Waveforms:



Design:



Note: The clue to get the solution is: There is a transition at the falling edge of clock. So the clock to the second flop is inverted one. The waveforms shown in the above figure, **fout** has a duty cycle of $1/3^{\text{rd}}$. To get $2/3^{\text{rd}}$ duty cycle, replace NOR gate with NAND gate in the above design.

Answer:

From the given circuit we can derive the following next state equations,
 $A(t+1) = A'$ and $B(t+1) = A \text{ XOR } B' = AB + A'B'$

Taking initial values of $A=B=0$ and drawing the waveforms with respect to the clock using above equations, we can observe that

$$\text{OUT} = \text{freq_Clock} / 4 \text{ with duty cycle} = 50\%$$

Q16) Here is an interesting design question. There is a room which has two doors one to enter and another to leave. There is a sensor in the corridor at the entrance and also at the exit. There is a bulb in the room which should turn off when there is no one inside the room. So imagine a black box with the inputs as the outputs of sensors. What should the black box be?

Answer:

The block box can be an up-down counter, where the “count_up_enable” is connected to the sensor at the entrance and “count_down_enable” to the sensor at the exit. That is if there is no one in the room, the counter’s output will be zero. Whenever this happens make the bulb “OFF”.

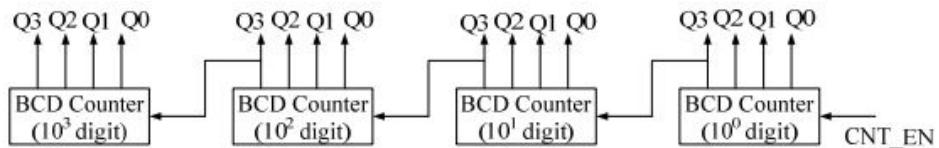
For 200 people, we need 8 bit counter.

So The O/P of entrance sensor will be used as enable for UP count and the other sensor at exit will be used for DOWN count, whenever the counter’s O/P is 0, we can make the BULB OFF, Otherwise ON.

Q17) Design a BCD counter which counts from 0 to 9999, using BCD decade counter as black box?

Answer:

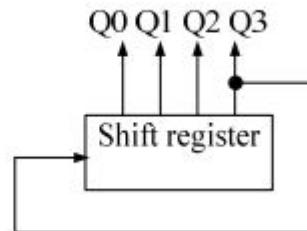
Each BCD counter counts from 0-9.



Q18) What is ring counter? Implement a ring counter using shift register?

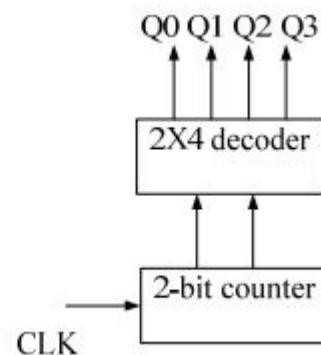
Answer:

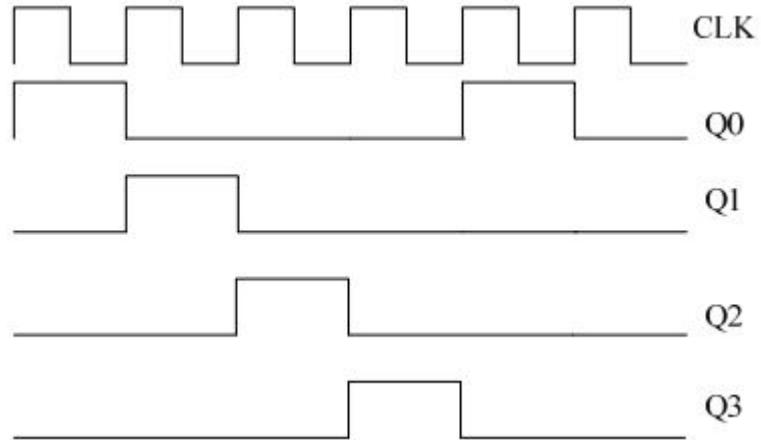
Ring counter: A ring counter is a circular shift register with only one flip-flop being set at any particular time, all others are cleared. This single bit is shifted from one flip flop to the next to produce the sequence of timing signals.



The above circuit shows the ring counter. The initial value of the shift register has to be 1000.

Q19) Ring counter implementation using 2-bit counter and 2:4 decoder is shown in the following diagram. Draw the output time signals, Q0, Q1, Q2 and Q3 with respect to the clock.

**Answer:**

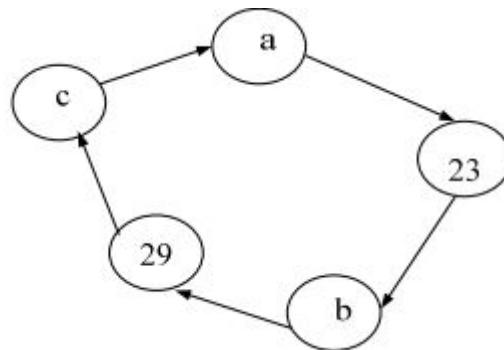


Q20) To generate 8 timing signals using a ring counter (similar to the circuit that is shown in Q19), mention the required size of decoder and the size of the counter?

Answer:

3-bit counter and 3:8 decoder

Q21) The following FSM shows the zero circulating ring counter. Predict the values of the missing states?



Answer:

In “0” passing ring counter, at any time only one flip flop will be set to 0 others will be 1. The given state values are 23 and 29. The binary representations are 10111 and 11101 respectively. So the states are : 01111, 10111, 11011, 11101, 11110. The decimal values are : 15,23,27,29,30.

a = 15, b = 27 and c=30

Q22) What are the unused states in a 3-bit Johnson counter?

Answer:

The states of 3-bit Johnson counter are: 000,100,110,111,011,001. So the unused states are 010 and 101

Q23) What is the length of counting sequence for a Johnson counter with N flip-flops?

Answer:

2N

Q24) How many unused states will be there in a Johnson's counter with N flip-flops?

Answer:

For N-flops, the total possible states = 2^N .

The number of states of a Johnson counter = 2N

So, the number of unused states = $2^N - 2N$

Q25) What is the output frequency of a 4-bit binary counter for an input clock of 160 MHz?

Answer:

The output of last flip flop of a 4-bit counter is equal to the input clock/16.

So output frequency = $160\text{MHz}/16 = 10\text{MHz}$

Q26) If each flip flop has a clock-to-Q delay of 10ns, how much time will it take for output to change to its next state in case of (a) 4-bit Ripple Counter (b) 4-bit Synchronous counter?

Answer:

(a) $10 + 10 + 10 + 10 = 40\text{ns}$

(b) 10ns

Q27) How fast can a 11 stage ripple counter be clocked, assuming worst case clock to Q delay of 40ns (of each stage) and extra gate delays of 60ns?

Answer:

Minimum time period of the clock = $11 \times 40 + 60 = 440+60 = 500\text{ ns}$ So maximum clock frequency = $1/500 = 2\text{ MHz}$

Q28) Design a counter using DFF that counts in the sequence: 0,4,2,7,0,4,2,7,0,4.....?

Answer:

The present state and next state values are shown in the table and the complete design is shown in the following diagram.

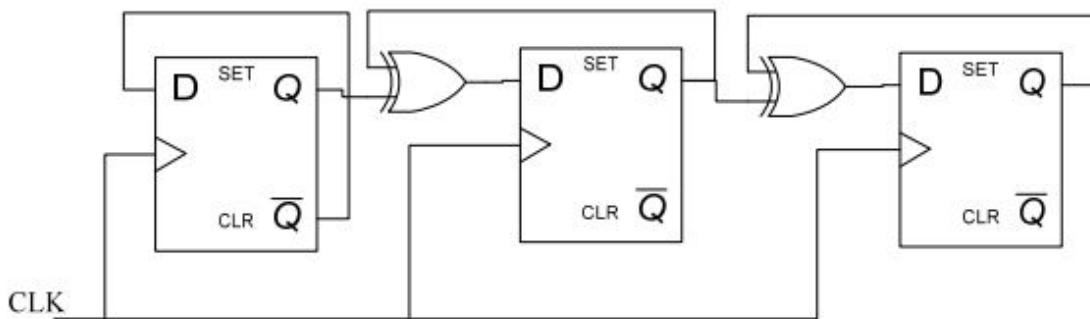
Present state			Next state		
A	B	C	A	B	C
0	0	0	1	0	0
1	0	0	0	1	0
0	1	0	1	1	1
1	1	1	0	0	0

Using this the following next state equations can be derived.

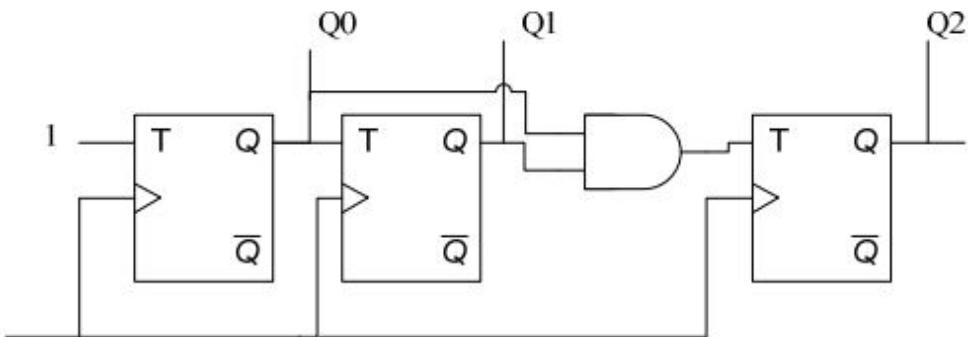
$$A(t+1) = A'$$

$$B(t+1) = A \text{ XOR } B$$

$$C(t+1) = B \text{ XOR } C$$



Q29) What is the functionality of the following circuit:



Answer:

Let us name the 3 flip flops as A,B and C

$$Q_0 = A'$$

$$Q_1 = A \text{ XOR } B$$

$$Q_2 = (AB) \text{ XOR } C$$

Starting with $A=B=C=0$, the next states can be obtained as:

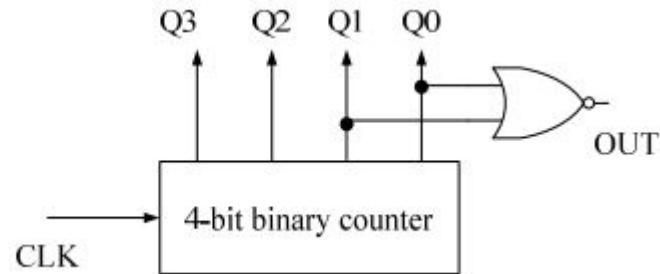
$$000 \rightarrow 001 \rightarrow 010 \rightarrow 011 \rightarrow 100 \rightarrow 101 \rightarrow 110 \rightarrow 111$$

So the circuit works as 3-bit binary counter.

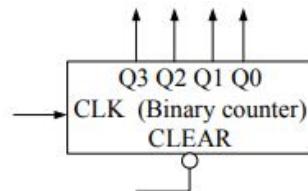
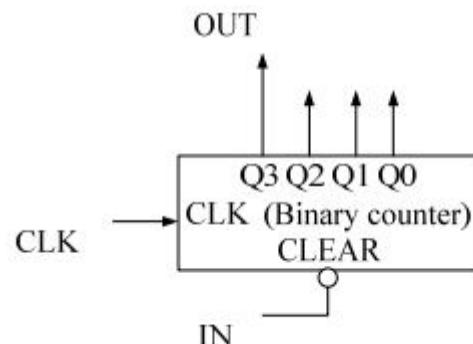
Q30) Using external gates and 4-bit counter, design a circuit which gives '1' if the number of clocks are multiples of 4.

Answer:

The output should be asserted 1 if the number of clocks is multiple of 4, that is 0, 4 , 8 and 12. The K-Map simplification gives, $OUT = Q1'Q0' = (Q1 + Q0)'$



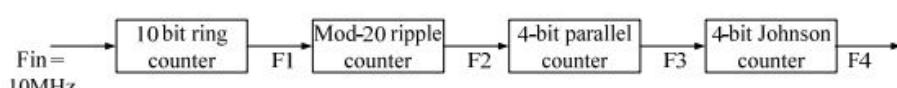
Q31) Design a sequential circuit that produces a logic 1 at the output when the input has been 1 for eight or more consecutive clock pulses using a counter(shown below) and minimum number of basic gates.

**Answer:**

Limitation of the design:

As the design used only one counter, the maximum count is 15.

Q32) What are the frequencies F1, F2, F3 and F4 after each stage if an input clock of 10MHz is applied?

**Answer:**

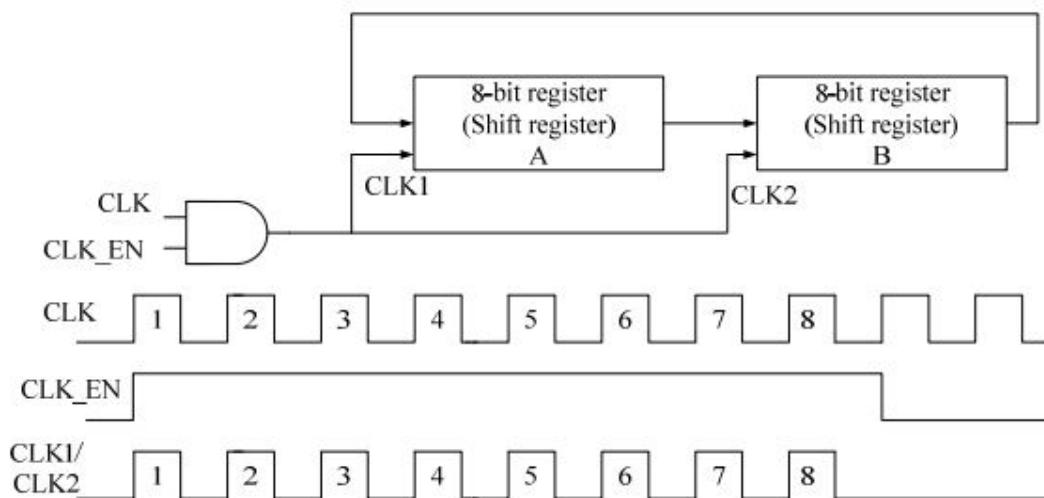
N-bit ring counter gives $1/N$ times the input frequency at the output. Johnson's output is $1/2N$ times the input. Where as the counters output will be $1/(2^N)$.

$$\begin{aligned}
 F1 &= 10\text{MHz} / 10 = 1\text{MHz} \\
 F2 &= 1\text{MHz} / 20 = 50\text{KHz} \\
 F3 &= 50\text{KHz} / 16 = 3.125\text{KHz} \\
 F4 &= 3.125\text{KHz} / 8 = 390.625\text{Hz}
 \end{aligned}$$

Q33) How to swap the contents of two 8-bit registers without using a third register.

Answer:

The complete design using shift registers is shown in the following figure. The main clock is gated with the clock enable so that A and B will be shifted just 8 clocks. After 8 clocks A and B will have their contents swapped.



Fault Analysis and Hazards

Q1) What are stuck-at problems? Explain the reason for their occurrence?

Answer:

A fault in a manufactured circuit causing a node to be stuck at a logical value of 1 (stuck-at-1) or a logic value of 0 (stuck-at-0), independent of the input to the circuit. If any rail during the layout gets connected to either VDD or GND permanently, it will lead to these stuck at problems.

Q2) How many number of stuck at problems are possible for a 2 input AND Gate? Which of those faults are not testable?

Answer:

Total possible faults are 6. (3 nodes, 2faults for each node) By single fault model, the test patterns that are needed are: 01,10 and 11 The stuck-at-0 problems at any of the inputs and stuck-at-0 problem at the output can not be distinguishable.

Q3) Define : (a) Test Pattern/Test set (b) ATPG

Answer:

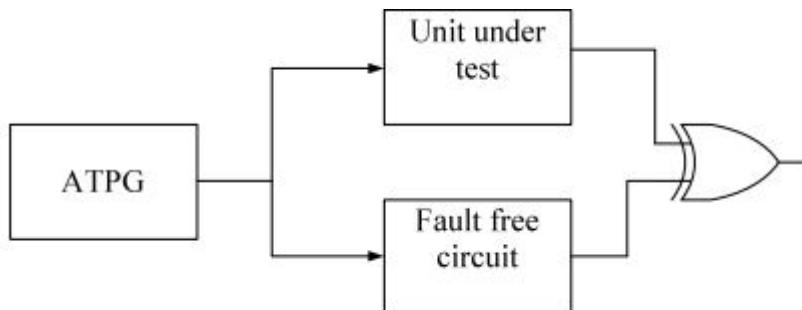
(a) Test pattern/set : The set of all input combinations that is needed to find out all the stuck-at faults of a digital circuit. Eg: Test set for 2-input AND gate: { 01, 10,11}

(b) ATPG: ATPG, or Automatic test pattern generation is an electronic design automation tool that generates the complete test set to distinguish between the correct circuit behavior and the faulty circuit behavior caused by a particular fault.

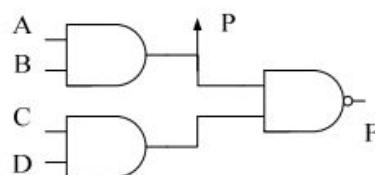
Q4) Explain the procedure for detecting a specific fault in a given circuit?

Answer:

Assume that there is only fault in the given circuit. This is called single fault model. Now apply the input combination such that the correct and faulty circuits would give different outputs.



Q5) To detect the Stuck at Zero problem at marked point 'P' in the following diagram, which of the input combinations can be used?

**Answer:**

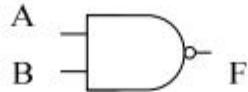
We need to select the pattern such that none of the inputs at AB,C& D should give 0 at any of the inputs of NAND gate. So the possible pattern is: A = B = C = D = 1 So we need to apply 1111 at the input, if it is correct circuit we will get, 0 at the output and if there is stuck-at-0 problem at P, we will get 1 at the output.

Q6) Give the test patterns that are needed to verify all the stuck at problems of a (a) 2-input NAND Gate (b) 3-input NAND Gate

Answer:

Single Fault method is used here.

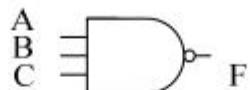
(a) 2-input NAND gate:



Node	For Stuck-at-0	For Stuck-at-1
A	11	01
B	11	10
F	00 or 10 or 10	11

So complete test set = { 10 , 01 , 11 }

(b) 3-input NAND gate:



Node	For Stuck-at-0	For Stuck-at-1
A	111	011
B	111	101
C	111	110
F	Atleast one input is 0	111

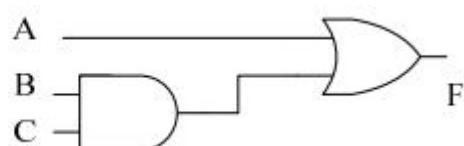
So complete test set = { 011,101,110,111 }

Q7) How many minimum number of test vectors are needed to verify all the stuck at problems of a N-input logical gate?

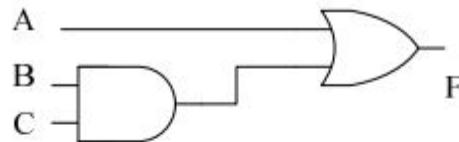
Answer:

N+1

Q8) Give the complete test set for the following circuit:



Answer:



Node	For Stuck-at-0	For Stuck-at-1
A	100	000
B	011	001
C	011	010
F	1xx	000 or 010 or 001

So complete test set = { 000, 001, 010, 011, 100}

Answer:

- (a) $F = (C+D)' = C'D'$
- (b) $F = AB$

Q10) What do you mean by “Path Sensitized Tests” in testing of logic circuits?

Answer:

If the number of nodes is more in a given circuit, it is very difficult to derive the test pattern by using single fault model (That is analyzing at each node). The other method called “Path sensitized” can be used to make the test pattern generation more efficient. In this method, all the paths from input to the output will be identified and the input will be applied such that the output will be dependant only on one particular path. And this path is called sensitized.

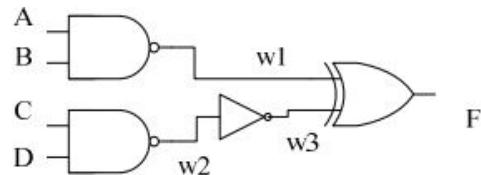
Answer:

- To make F dependant of the highlighted path, the output of XOR gate has to be 1. This can be achieved from $A = 0, B = 1$ or $A = 1, B = 0$.
- To make the output of OR gate to make dependant only on one input, the other input has to be 0. This can be achieved either $B = 0$ or $C = 0$;
- But C can not be fixed to 0. So B has to be 0. That implies $A = 1$
- To make NOR gate output only C dependant, D has to be 0.

So to make the given path sensitized, the input pattern that is needed is,

$A = 1, B = 0, C = 1$ or 0 and $D = 0$

Q12) Which input pattern can be applied to make the path from input C to the output F (via NOR gate – OR gate – AND gate), sensitized, in the following diagram?



Answer:

- (a) 0100 at the input, makes the path A-w1-F sensitized.
So it can be used to detect the following single stuck-at problems:

Stuck-at-1 at A or Stuck-at-0 at w_1 or Stuck-at-0 at F'

- (b) Path sensitized method is used here.

To make F dependant only on w_3 , $w_1 = 0$. So A=B=1

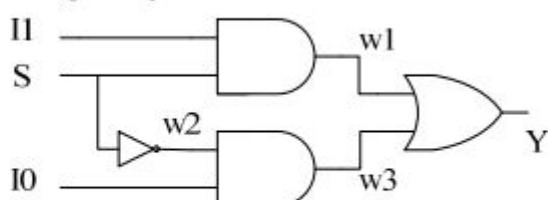
Now for identifying the Stuck-at-0 fault at w_3 , we need to apply input pattern such that we will get 1 at w_3 (C=D=1). So the required pattern is 1111.

For identifying the Stuck-at-1 fault, either C or D has to be 0. So any of the following patterns can be used : 1100, 1101 or 1110

Answer:

- (a) AOI of a 2:1 Mux:

$$Y = S' I_0 + S I_1$$



- (b) All possible paths from inputs to Y:

Path1 : $I_1 - w_1 - Y$,

Path2 : $S - w_1 - Y$,

Path3: $S - w_2 - w_3 - Y$,

Path4: $I_0 - w_3 - Y$

- (c) Test vectors needed for each path:

Path	Test vector needed
Path1 : $I_1 - w_1 - Y$	01x, 11x
Path2 : $S - w_1 - Y$	11x, 100
Path3: $S - w_2 - w_3 - Y$	011, x01
Path4: $I_0 - w_3 - Y$	x01, x00

x – indicates don't care

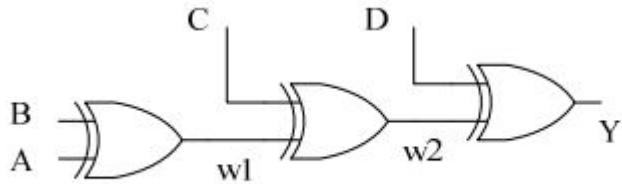
So the complete test set is { 100, 011, 11x, x01, x00 }

Q14) (a) Give the circuit for a 4-bit parity generator?

(b) Derive the minimal test set that can detect all stuck-at-faults?

Answer:

- (a) The 4-bit even parity generator is shown in the following diagram:



(b) Path sensitized test:

Path	Test vector needed
Path1 :D – Y	0001, 0000
Path2 : C – w ₂ – Y	0010, 0000
Path3: B – w ₁ – w ₂ – Y	0100, 0000
Path4: A – w ₁ – w ₂ – Y	1000, 0000

So the complete test set is {0000, 0001, 0010, 01000, 1000}

Q15) What is D-notation?

Answer:

In test of logic circuits, normally the logic levels are represented with D. This is called D-Notation. If Logic-0 is represented with D, logic 1 will be D' and vice versa.

Q16) What are the two hazards that can be there in a combinational circuit?

Answer:

A Static Hazard is defined when a single variable change at the input causes a momentary change in another variable [the output]. A Dynamic Hazard occurs when a change in the input causes multiple changes in the output [i.e. from 1 to 0 and back to 1]. In either case of a Static or Dynamic hazard the product produced is an unanticipated glitch [the hazard]. The resulting glitches in the circuit may or may not induce additional problems, other than increased issues due to switching noise.

There are two types of Static hazards: the high output transitions to a low and back high [a low going glitch]. Or the low output transitions to a high [1] and back low [0] [a high going glitch]. There are also two types of Dynamic hazards: the 0 output transitions to a 1 back to 0 and then 1 again. Or the 1 output transitions to a 0 back to 1 and then 0 again.

Answer:

- (a) ii
- (b) i
- (c) iii

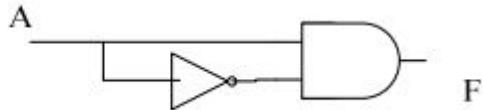
Q18) Give the characteristics of Static-zero Hazard?

Answer:

Static-zero Hazard's characteristics: Two parallel paths for x , one inverted and reconverge at an AND gate.

$$F = A \cdot A'$$

Any circuit with a static-0 hazard must reduce to the equivalent circuit of the following figure:



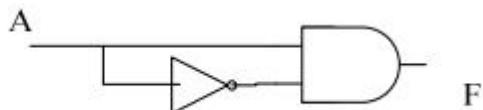
Q19) Show the equivalent circuit for Static-one Hazard?

Answer:

Static-one Hazard's characteristics: Two parallel paths for x , one inverted and reconverge at an OR gate.

$$F = A + A'$$

Any circuit with a static-1 hazard must reduce to the equivalent circuit of the following figure:



Q20) How many single variable change static-0 hazards the Boolean function, $G = AB + A' C + B'C'D$ has?

Answer:

$$G = AB + A' C + B'C'D$$

If $B = C = 0$, $G = A+A'$ (Static-0 hazard in A)

If $A = 1, C = 0, D = 1$, $G = B+B'$ (Static-0 hazard in B)

If $A = 0, D = 1$, $G = C+C'$ (Static-0 hazard in C)

Q21) How to avoid static hazards in a given circuit(single variable change hazards)?

Answer:

To avoid the static hazards, one of the possible ways is delay matching. Suppose in the circuits shown above (A18 & A19), we can provide buffer whose delay is equal to that of NOT gate. But it becomes very difficult to match the delays exactly.

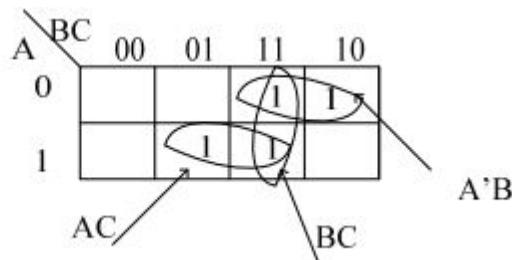
If Static Hazards are removed from the design, Dynamic Hazards will not occur. A Karnaugh map [K-map] is the easiest way to eliminate a Static Hazard or glitches. A Kmap for each combinatorial logic function which has an output should be used.

Redundant prime implicants should be added to the K-Map (circuit), which will guarantee that all single-bit input changes are covered. Multi-level functions will be reduced to "two-level" functions, and analyzed by the K-map approach. The procedure for designing a static-hazard-free network is a straightforward application. The key is to place the function in such a form that the transient output function guarantees that every set of adjacent 1's in the K-map are covered by a term, and that no terms contain both a variable and its complement. The former condition eliminates 1-hazards and the latter eliminates 0-hazards.

Answer:

(b)

The boolean expression, $Y = AB + A'C + BC$ can be further reduced to $Y = A'C + AB$. But the second expression will have hazards. So the redundant term BC is added. You can observe the K-Map of the same:



Q23) Find a hazard-free minimum cost implementation of the function:

$$F(A,B,C,D) = \sum (0,4,11,13,15) + d(2,3,5,10)$$

Answer:

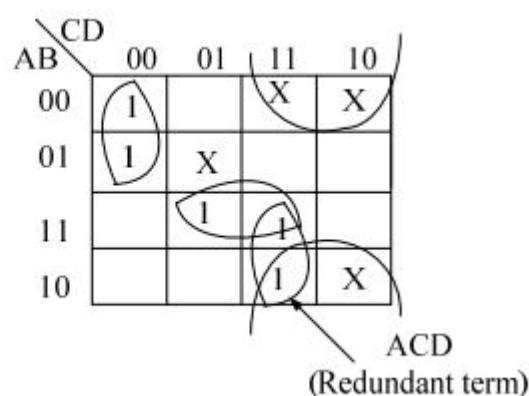
$$F(A,B,C,D) = \sum (0,4,11,13,15) + d(2,3,5,10)$$

From the K-Map shown below, the simplified expression for F is,

$$F = ABD + A'C'D' + B'C$$

But to make it Hazard free, we need to add the redundant term, ACD to this.

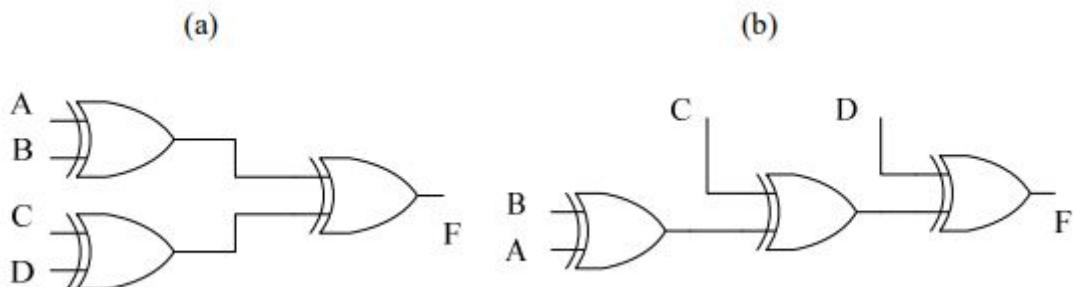
$$F = ABD + A'C'D' + ACD \text{ (Note that } B'C \text{ is removed from the equation)}$$



Q24) Design 4-input XOR gate using 2 input XOR gates in all possible ways? Discuss the advantages and disadvantages of each?

Answer:

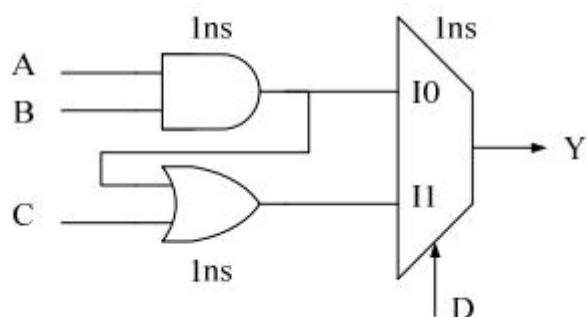
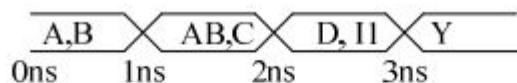
The two possible implementations are shown below:



If we compare both the implementations, in implementation (a), the delays from the inputs to the output, F are uniform. So there is no possibility of glitches. Where as in the implementation (b), the delays are not balanced properly. (a) is hazard free and the better implementation when compared to (b).

Answer:

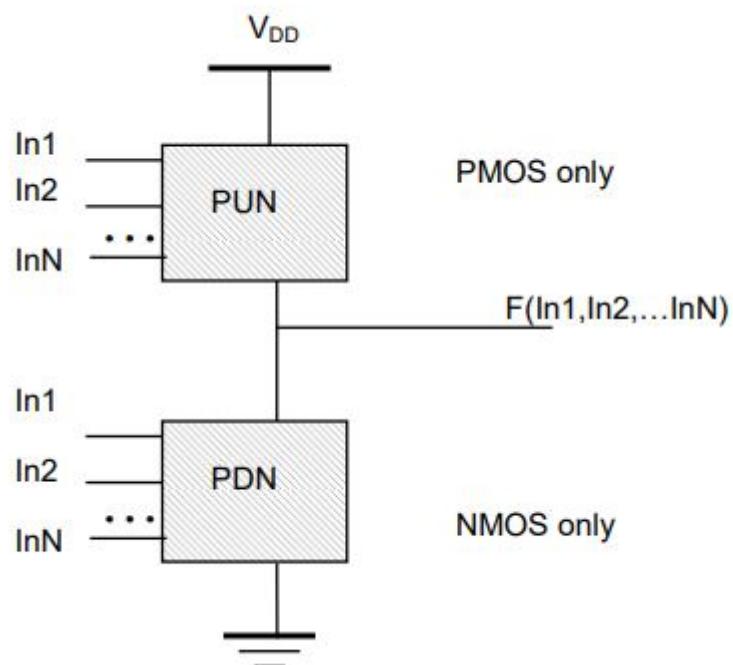
$$Y = AB + CD$$



Q1) What do you mean by CMOS technology? Explain with a block diagram.

Answer:

CMOS(Complementary MOS) circuits consist of both types of MOS devices interconnected to form logic functions as shown in the following block diagram. The PUN(Pull up network) will charge the output node in case of Logic-1 and the PDN(Pull down network) will discharge by connecting the output node to ground, in this way the output is connected either to VDD or GND continuously. PUN and PDN are dual logic networks. CMOS take advantage of the fact that both n-channel and p-channel devices be fabricated on the same substrate.



Q2) What are the advantages and disadvantages of CMOS logic?

Answer:

The CMOS logic has two important advantages:

Advantages:

- No direct path in steady state between power and ground, so no static power dissipation(except for small power dissipation due to leakage currents)
- Full logic levels (The VTC exhibits a full output voltage swing between 0 and V_{DD} , and that VTC is usually very sharp)
- High noise margins (Good noise immunity)
- Extremely high input resistance; nearly zero steady-state input current
- Low output impedance. Always a path to V_{dd} or Gnd from output node, in steady state;
- CMOS provides a greater packing density.

Disadvantages:

- CMOS processing is more complex

- Latch up problem
- Slower compared to TTL
- Higher cost

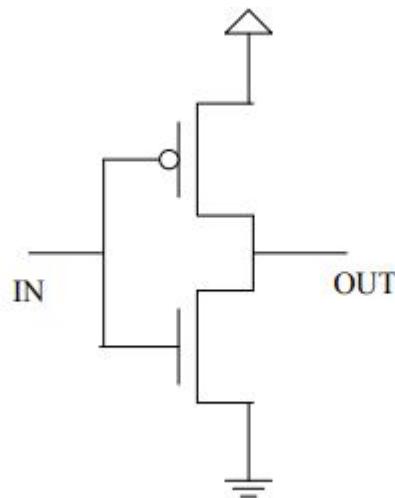
Answer:

(b)

Q4) Show the circuit for CMOS inverter and explain the basic operation?

Answer:

CMOS inverter:

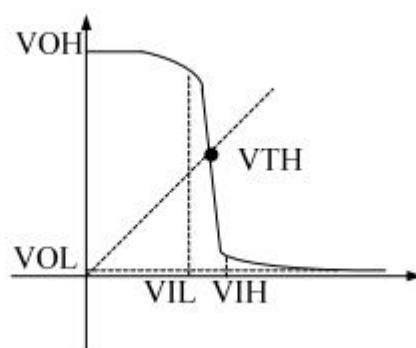


Basic operation: When input goes from 0 to 1, the PMOS will be off and the NMOS will be on. This makes the OUT to get connected with GND and goes to 0. Similarly when input is 0, the NMOS will be OFF and PMOS turns ON making the output logic to VDD. We will get full logic levels at the output.

Q5) Draw the VTC of a CMOS inverter?

Answer:

Voltage Transfer Characteristics (VTC) of a CMOS inverter:



Q6) Arrange the following in the decreasing order of voltage levels:
VOH,VOL,VIH,VIL.

Answer:

VOH > VIH > VIL > VOL (Refer to VTC shown in A5)

Q7) Define: (a) Fan-out (b) Noise-margin

Answer:

(a) **Fan_out:** The fan-out of a gate specifies the number of standard loads that can be connected to the output of the gate without degrading its normal operation. The fan-out is calculated from the amount of current available in the output of a gate and the amount of current needed in each input of a gate.

$$\text{Fan_out} = \text{Min} (I_{OH}/I_{IH}, I_{OL}/I_{IL})$$

(b) **Noise margin:** Noise margin is the maximum noise voltage that can be added to an input signal of a digital circuit that does not cause an undesirable change in the circuit output.

$$\text{Noise margin} = \text{Min} (V_{OH}-V_{IH}, V_{IL}-V_{OL})$$

Q8) Find the noise margin: VOH = 4V, VIH = 3V, VOL = 1V and VIL = 1.5V

Answer:

Given: VOH = 4V, VIH = 3V, VOL = 1V and VIL = 1.5V

$$VOH - VIH = 4 - 3 = 1$$

$$VIL - VOL = 1.5 - 1 = 0.5$$

$$\text{Noise Margin} = 0.5$$

Q9) Find out fan_out and propagation delay of a logical gate for which the following specifications are given: VCC = 5V, ICCH = 1mA, ICCL = 2mA, IOH = 1mA, IOL = 20mA, IIH = 0.05mA and IIL = 2mA

Answer:

(a) **Fan_out :**

IOH = 1mA, IOL = 20mA, IIH = 0.05mA and IIL = 2mA

$$IOH/IIH = 1/0.05 = 20$$

$$IOL/IIL = 20/2 = 10$$

$$\text{So, Fan_out} = 10$$

(b) **Power dissipation :**

VCC = 5V, ICCH = 1mA, ICCL = 2mA

$$Icc(\text{avg}) = (ICCH + ICCL)/2 = 1.5\text{mA}$$

$$\text{Power dissipation} = VCC * Icc(\text{avg}) = 5 * 1.5 = 7.5 \text{ mW}$$

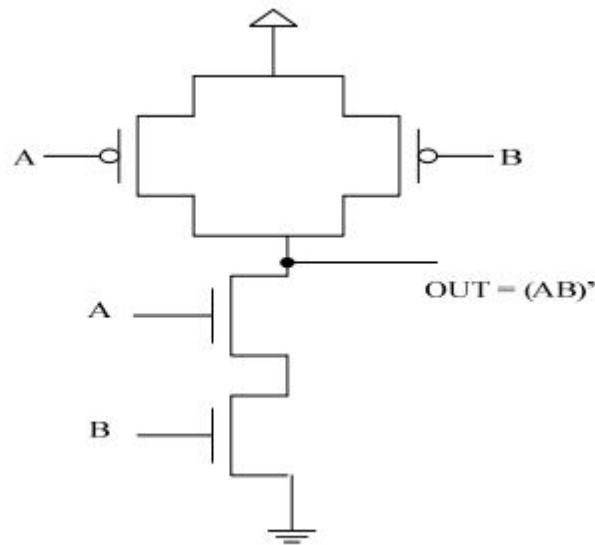
Answer:

(b)

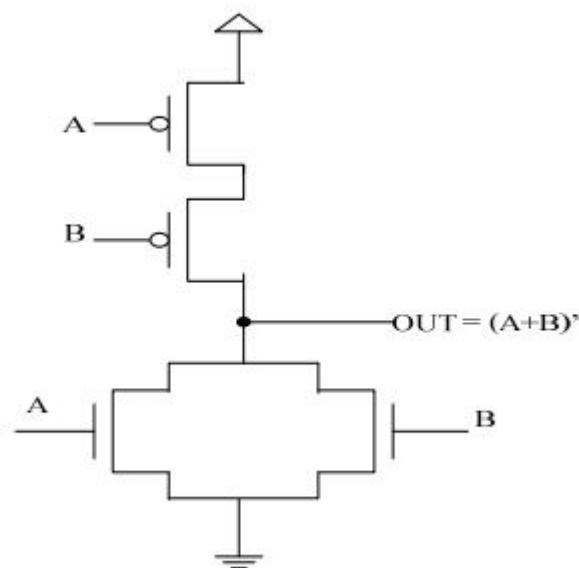
Q11) Draw the CMOS implementation of NAND and NOR gates.

Answer:

(a) 2-input NAND gate:



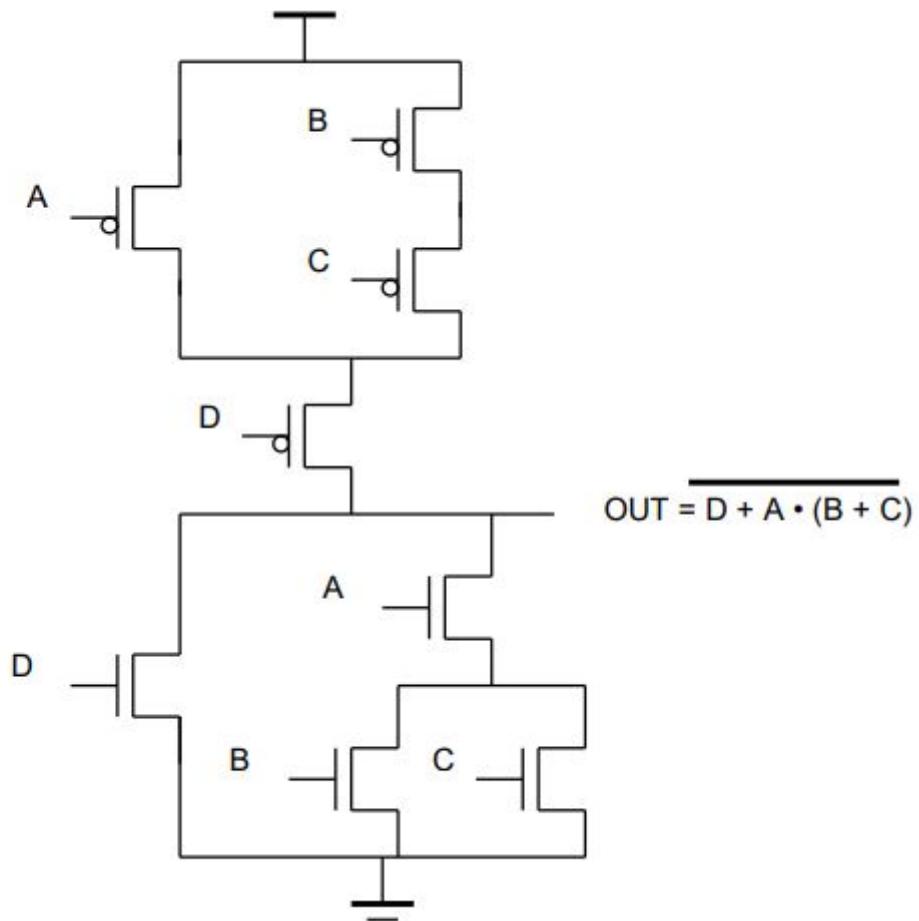
(b) 2-input NOR gate:



Answer:

Deriving the Pull-up network hierarchically identifying sub nets as shown in the following figure:

The complete circuit and the output boolean function is shown below:

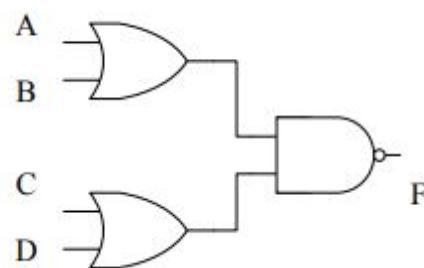


Q13) How many minimum number of MOS transistors are required to implement the Boolean function, $Y(A,B,C) = AB + A'C + BC$ using CMOS implementation assuming the inputs and their complements are available?

Answer:

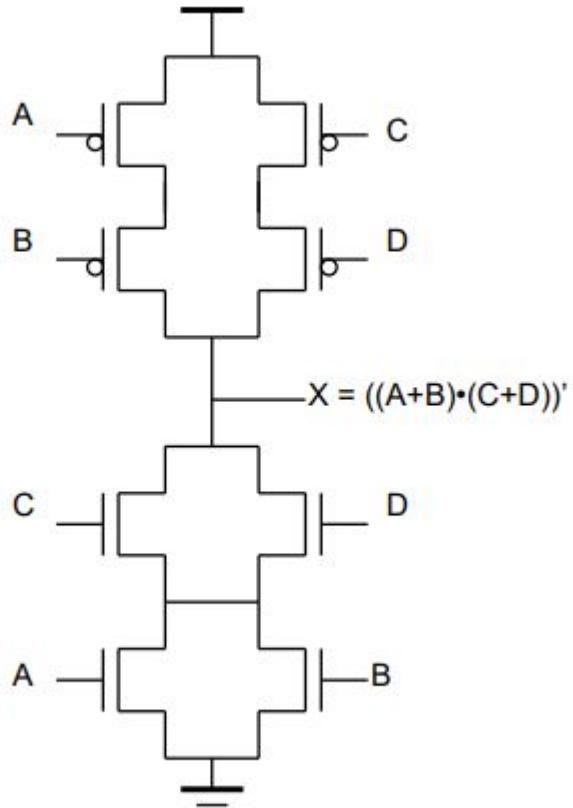
Deriving the Pull-up network hierarchically identifying sub nets as shown in the following

Q14) Draw the CMOS implementation for the following circuit:



Answer:

$$F = ((A+B) \cdot (C+D))'$$

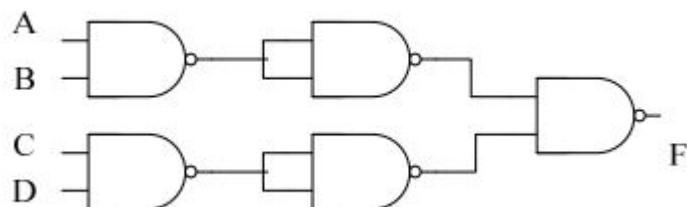


Q15)

- (a) Design a 4-input NAND gate using only 2-input NAND gates?
- (b) What are the minimum number of transistors that are needed to draw the gate level equivalent of the same?

Answer:

- (a) 4-input NAND gate from 2-input NAND gates:



- (b) Number of NMOS = Number of PMOS transistors = 4 (So total, 8)

Q16) How many minimum number of MOS transistors are required to implement a Full Adder using CMOS technology?

Answer:

$$S = A \text{ XOR } B \text{ XOR } C \text{ and } Cout = AB + BC + AC = AB + (A+B)C$$

S can be rewritten as, $S = ABC + (A+B+C) Cout'$

For Cout, NMOS = PMOS = 6 (Total 12)

For S, NMOS = PMOS = 8 (Total 16)

So for one bit full adder implementation, minimum number of transistors that are required = 28

Q17) Let A & B be two inputs of the NAND gate. Say signal A arrives at the NAND gate later than signal B. To optimize delay, of the two series NMOS inputs A & B, which one would you place near the output?

Answer:

The late coming signals are to be placed closer to the output node ie A should go to the NMOS that is closer to the output.

Reason is, by the time A comes, B would have turned on the bottom transistor and discharged the intermediate node between the 2 series NMOS. So by the time A comes, it can discharge the output node very quickly.

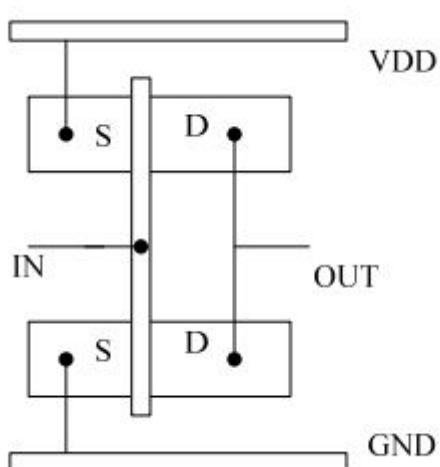
Answer:

(d)

Q19) Draw the stick diagram of CMOS inverter?

Answer:

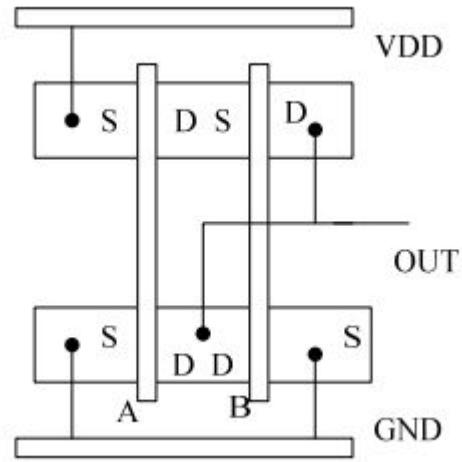
Stick diagram for CMOS inverter



Q20) Draw the stick diagram of NOR gate. Optimize it.

Answer:

Stick diagram for NOR gate



Q21) What do you mean by pass transistor logic?

Answer:

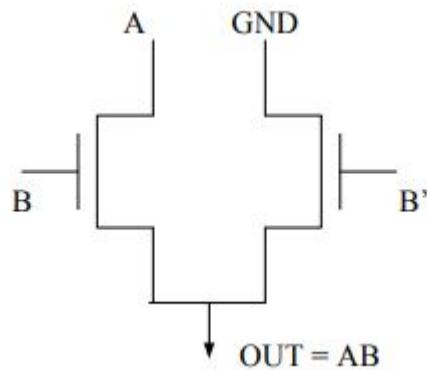
Pass transistor logic:

- A pass transistor is a MOSFET in which an input is applied not only to the gate but also to the drain
- Unlike static CMOS, there is no need for any static power supplies
- More advantageous in terms of number of transistors if the inputs and their complements are available
- Disadvantage is : Degraded logic level as NMOS passes weak logic-1

Q22) Show the implementation of AND gate using pass transistor logic?

Answer:

The AND gate using pass transistor logic is shown below:



Answer:

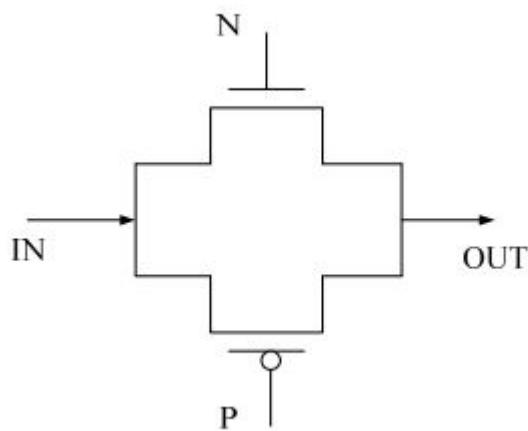
- (a) 2:1 Mux
- (b) $OUT = S I_1 + S' I_0$

- (c) Pass transistor logic
- (d) Degraded logic 1. To avoid we need to use both NMOS and PMOS together (That is transmission gate)

Q24) Show the circuit of Transmission gate and explain the functionality?

Answer:

Transmission gate consists of one n-channel and one p-channel MOS transistor connected in parallel. The same thing is shown in the following diagram. When N is at VDD and P is at ground, both transistors conduct and there is a closed path between IN and OUT.



Q25) Why don't we use just one NMOS or PMOS in a transmission gate?

Answer:

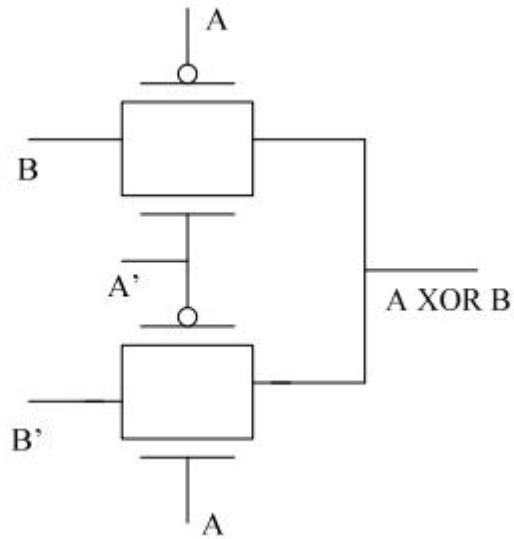
Using only an NMOS will result in an poor 1. Assume the gate voltage on NMOS is 5V. If we connect Drain to 5V, and the source is initially at 0, NMOS will turn on as long as $V_{gs} > V_{th}$, this means, once the source reaches 4.3V (Assuming $V_{th}=0.7$), the NMOS will turn off and there will be no more increase in source voltage. Similarly the opposite happens with PMOS, it doesn't give us a clean 0, but it can give a full 5V. So we use a combination of both NMOS and PMOS so that our signal doesn't get degraded by V_{th} on either side of VDD and GND.

Q26) Design a Transmission Gate based XOR. Now, how do you convert it to XNOR? (Without inverting the output)

Answer:

XOR Using TG:

If we observe the truth table of XOR, if A is 1, output is B' and if A is 0 output is B. Using this, we can implement the following circuit.

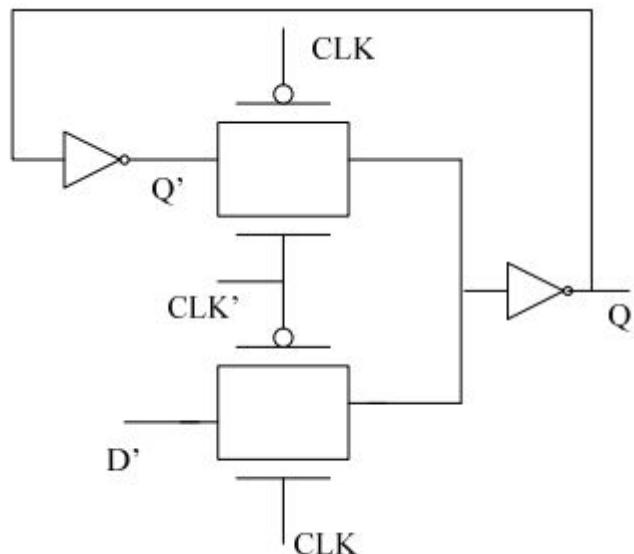


To get XNOR, just connect B directly to bottom TG and B' to the upper TG.

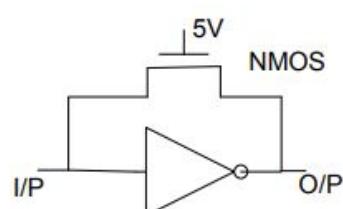
Q27) Draw a Transmission Gate-based D-Latch.

Answer:

D-Latch using TG:



Q28) The output and input of a static CMOS inverter are connected as shown in the above figure. What is the output voltage?



Answer:

The NMOS transmits the same voltage from drain to source, as long as its value is less than 4V. So in the given diagram the output and input of the inverter are same. If we observe the VTC of an inverter, $V_{out} = V_{in}$ at $V_{th} = V_{DD}/2 = 2.5V$. So if there is no noise floor, the output will settle to 2.5V (This is theoretical analysis). However practically the circuit will oscillate.

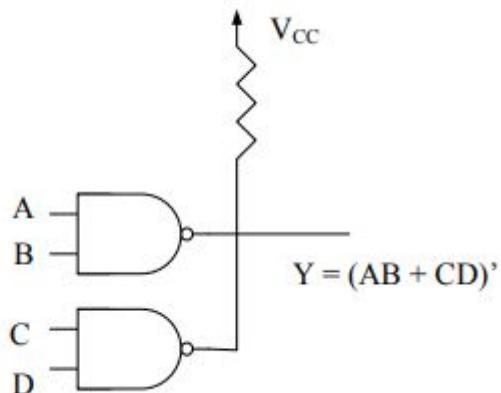
Q29) What are the applications of open-collector gate?

Answer:

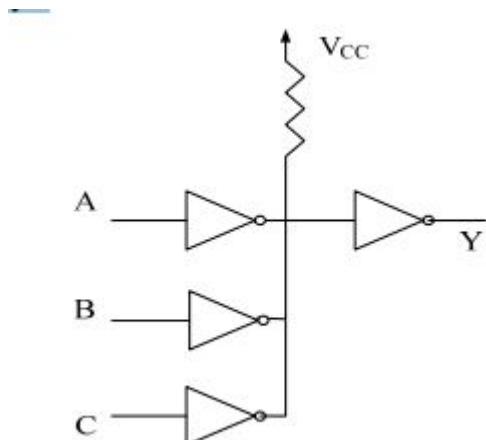
The 3 major applications of open-collector gate are:

- Driving lamp or relay
- Performing wired logic
- Construction of common bus system

Q30) Using open-collector NAND gate, implement $Y = (AB + CD)'$?

Answer:

Q31) Four open collector gates are connected as shown in following circuit. What is the functionality?



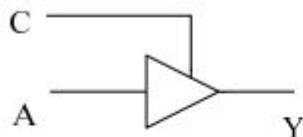
Answer:

It forms a common bus system. We can transmit one of the inputs A,B,C to the output Y by making the other inputs 0. Suppose if A=B=0, Y is C.

Q32) Describe (a) Three-state buffer gate (b) Three-state inverter gate

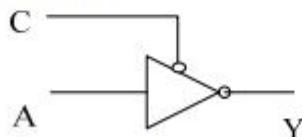
Answer:

(a) Three-state buffer:



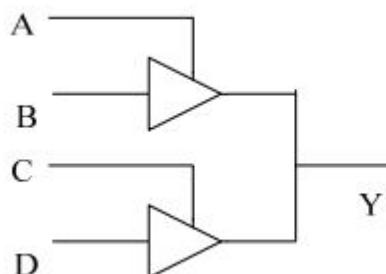
If C = 1 , Y = A and if C = 0, Y is high impedance

(b) Three-state inverter:



If C = 0 , Y = A' and if C = 1, Y is high impedance

Q33) Two Three-state buffers are shown below. The output of a 4 bit binary counter is connected to the 4 inputs A, B, C and D such that the MSB is connected to D and LSB to A. For how many counter states, the circuit is sure to produce proper output(0 or 1)?

**Answer:**

If A = 1, Y = B or If C = 1 Y = D otherwise Y is high impedance. So all combinations with A = 1, C = 0 or A = 0, C = 1 or A = 1, C = 1 Except the four combinations : 0000,0010,1000 and 1010, The rest all combinations can give either logic-1 or logic-0 at the output.

Q34) What is Latch Up? How do you avoid Latch Up?

Answer:

Latch up: In fabricating CMOS ICs, parasitic bipolar transistors are formed as byproducts of CMOS processing. These parasitic pnp and npn bipolar transistors form a SCR(Silicon

controlled Rectifier) with positive feedback and virtually short circuit the power rail to ground. The generation of such a low-impedance path in CMOS chips between the power rail and the ground rail is defined as Latch-up.

Guidelines to avoid Latchup:

- Layout n- and p-channel transistors such that all NMOS transistors are placed near close to VSS and all PMOS transistors are placed close to VDD rails. Also maintain sufficient spacings between NMOS and PMOS transistors.
- Use p+ guard rings connected to ground around NMOS transistors and n+ guard rings connected to VDD around PMOS transistors.
- Use minimum area p-wells

Note: Here very basic solution is given. The concept of SCR and guard rings is not given here. It is advisable to read about Latchup.

Memories, FIFO and Programmable devices

Q1) What is Volatile memory? Give an example.

Answer:

Any type of memory that requires power in order to store information is called volatile memory. RAM is volatile whereas ROM is non-volatile. That means ROM can store data without power also.

Q2) Differentiate between RAM and ROM?

Answer:

Differences between RAM and ROM:

1. ROM: Read Only Memory. RAM : Random Access Memory
2. ROM has no write operation. RAM has both read and write operations
3. ROMs are non-volatile and RAMs are volatile.

Q3) How many address and data lines will be there for a memory of size, 1K X 8?

Answer:

1K = 2¹⁰ , Number of address lines = 10

Number of data lines = 8

Q4) How many number of 16X8 size memories are needed to obtain a memory of size 256X16?

Answer:

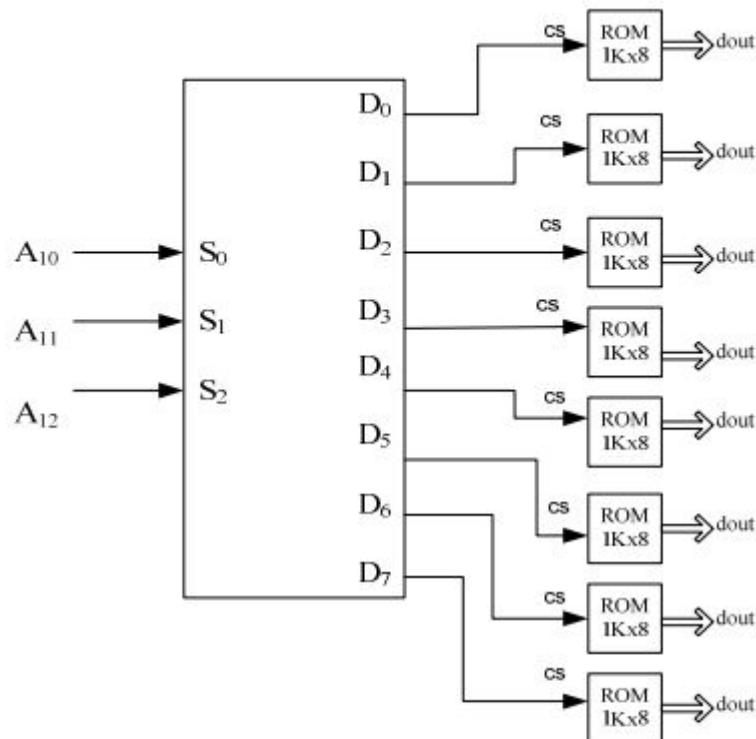
$256/16 = 16$, 16X8 memories are sufficient to get a memory of size 256 X 8. But to get 256 X 16, we need twice of that.

So, the required number of 16 X 8 memories = $16 * 2 = 32$

Q5) Design a memory of size 8KX8 using a 3:8 decoder and the minimum number of ROMs of size 1KX8 shown in the following diagram. (cs chip select, active high). Also show the complete address map.

Answer:

We need 8, 1KX8 memories. For 8KX8 memory has 13 address (A12-A0) lines. For each 1KX8 memory, there will 10 address lines. So we can connect the A0-A9 address lines directly to these 10 address lines. And the remaining, that is from A10,A11,A12 can be used as select lines for the decoder and the decoders outputs will be connected to “cs” of the ROMs. The complete design is shown below:



The corresponding address map is shown in the following table:

Address range in hex	ROM that will be selected
0000 – 03FF	ROM1
0400 – 07FF	ROM2
0800 – 0BFF	ROM3
0CFF – 0FFF	ROM4
1000 – 13FF	ROM5
1400 – 17FF	ROM6
1800 – 1BFF	ROM7
1CFF – 1FFF	ROM8

Q6) Using DFF design a binary cell, which can perform read/write operations based on enable, r/w ? Also provide memory enable, mem_en?

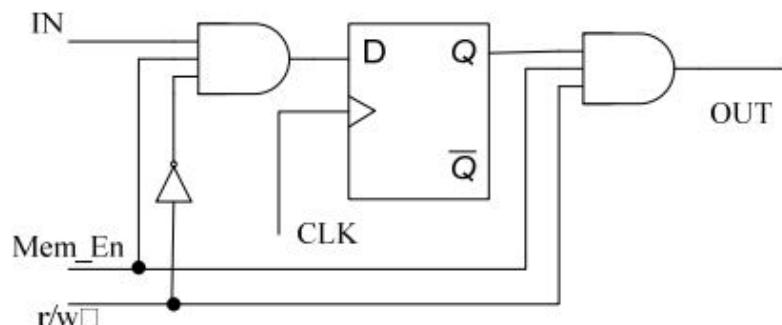
Answer:

Mem_En	r/w	Operation
0	X	Output = 0
1	0	Write
1	1	Read

Using the truth table shown in above table, the following equations can be derived:

$$Din = IN \text{ AND } Mem_En \text{ AND } (r/w)' \text{ and}$$

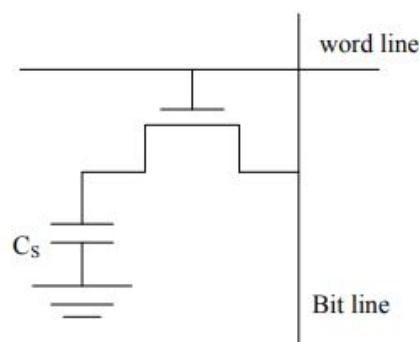
$$OUT = Q \text{ AND } Mem_En \text{ AND } (r/w)$$



Q7) Give the basic circuit for DRAM?

Answer:

One-transistor Dynamic RAM cell is as follows:



The circuit shows the access transistor, NMOS transistor and the storage capacitance (typically 30-50 fF). Logic-1 at word line makes the MOS transistor conductive. Then the bit line capacitance (nearly 30 times more than CS) comes in parallel with the CS. This allows the charge sharing between the two capacitors. (A filled capacitor equals to a logical one while an "empty" capacitor equals to a logical zero.)

Q8) Draw the circuit for SRAM?

Answer:

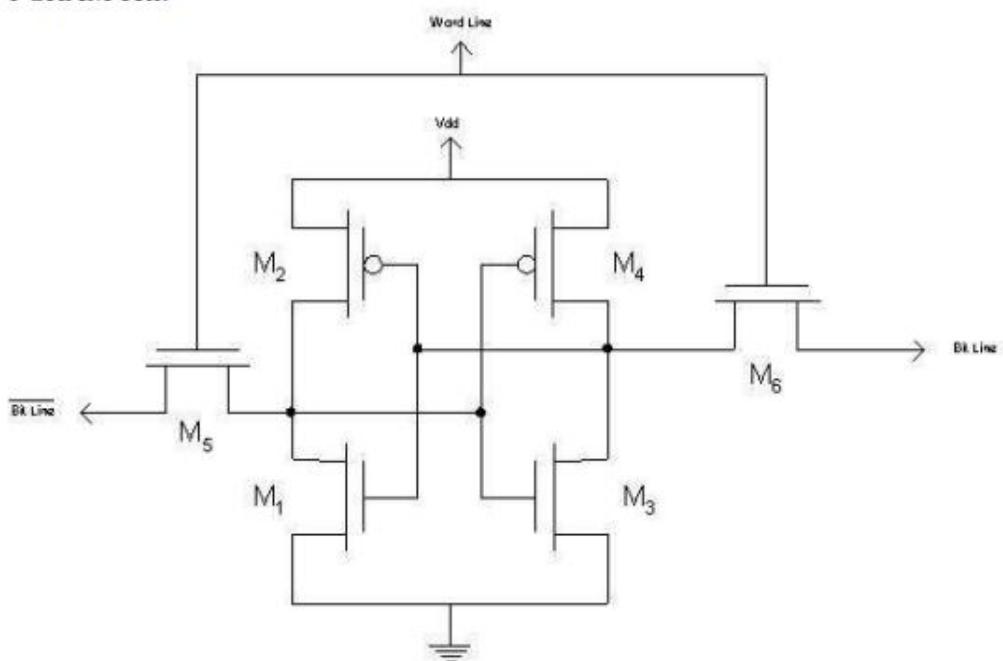
A single SRAM memory cell is shown in the below diagram. As can be noted, six total transistors are required for our design. Two NMOS and two PMOS transistors are used to construct a simple latch to store the data, plus two more pass NMOS transistors are controlled by Word Line to pass Bit Line and Bit Line Bar into the cell.

Write and Read operations are performed by executing a sequence of actions that are controlled by the outside circuit.

Write Operation: A Write operation is performed by first charging the Bit Line and Bit Line Bar with values that are desired to be stored in the memory cell. Setting the Word Line high performs the actual write operation, and the new data is latched into the circuit.

Read Operation: A Read operation is initiated by pre-charging both Bit Line and Bit Line Bar to logic 1. Word Line is set high to close NMOS pass transistors to put the contents stored in the cell on the Bit Line and Bit Line Bar

6-T SRAM cell:



Q9) Define memory access time?

Answer:

The time that is required for the data to be available at the memory output after receiving the new address at the input is called memory access time. It is a measure of a memory devices operating speed.

Q10) Which is faster: SRAM or DRAM?

Answer:

SRAM is faster than DRAM

Q11) What are the advantages and disadvantages of DRAM when compared to SRAM?

Answer:

DRAM has 3 main advantages over SRAM:

1. DRAM memory cell (1 transistor and capacitor) is simple and smaller than SRAM (6 transistors). DRAM has more density (more cells per chip). The larger memories are always made of DRAMs only. (Main memory)
2. DRAM is cheaper than SRAM.
3. DRAM dissipates lesser power.

Disadvantages:

1. DRAM is slower than SRAM. Where speed is critical, SRAM will be used. Eg: Cache memory
2. DRAM requires periodic refreshing.
3. SRAM is compatible with CMOS technology whereas DRAM is not.

Q12) Explain read-refresh operation in case of DRAM?

Answer:

In DRAMs, capacitors are used for storing the information. The capacitor discharges after some time. So periodic refreshing is needed. Also, whenever the cell is read, the value will be written back to the cell. This is called read-refresh operation.

Answer:

- (a) Size of memory = 16×1 (Each binary cell is of width 1)
- (b) 1010
- (c) Row4, Column1

Q14) What is dual data ROM?

Answer:

DDR RAM or double-data-rate RAM is a type of memory integrated circuits used in computers. It achieves greater bandwidth than ordinary RAM by transferring data on both the rising and falling edges of the clock signal. This effectively nearly doubles the transfer rate without increasing the frequency of the front side bus. Thus a 100 MHz DDR system has an effective clock rate of 200 MHz when compared to equivalent SDR RAM.

Q15) Expand the following: (a) PLD (b) PLA (c) PAL (d) FPGA

Answer:

- (a) PLD : Programmable Logic Devices
- (b) PLA : Programmable Logic Array
- (c) PAL : Programmable Array Logic
- (d) FPGA: Filed Programmable Gate Array

Q16) What is the difference between PLA and PAL?

Answer:

In PLA both AND and OR arrays are programmable whereas PAL has programmable AND array and a hardwired OR array. When number of functions to be realized is low, PLA is costly. For those cases, PAL is much cheaper.

Answer:

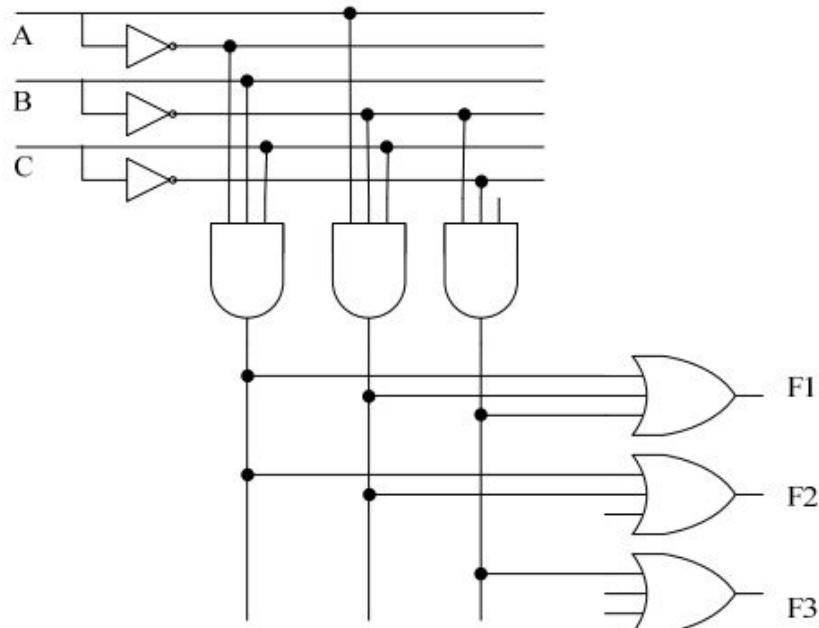
- (a) ii
- (b) iii
- (c) i

Answer:

$$F_1 = A'BC + AB'C + B'C'$$

$$F_2 = A'BC + AB'C$$

$$F_3 = B'C'$$



Q19) What is FIFO? Explain the significance?

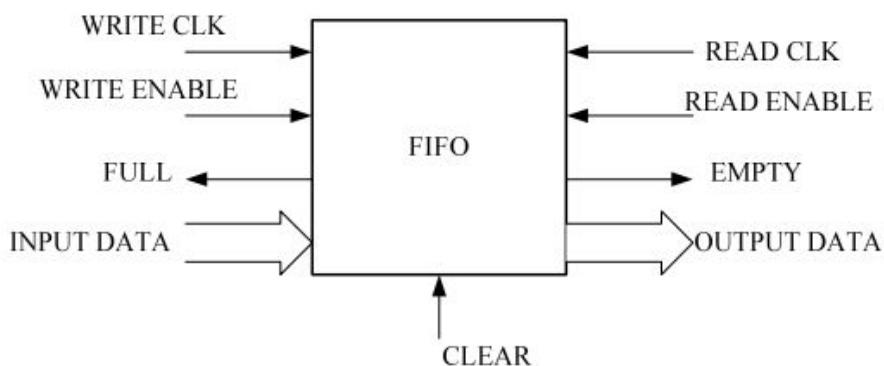
Answer:

FIFO (First In First Out) is a special type of storage memory where the first data bit written into the memory is the first to be read out. Putting in another way, FIFO is a storage method that retrieves the data stored for the longest time. The FIFO memory is used when two systems of differing data rates must communicate. Data can be entered into a FIFO register at one end and taken out at the other end at another rate.

Q20) Show the basic block diagram of FIFO and explain the basic signals or connections of a FIFO?

Answer:

A FIFO with all the necessary signal lines is shown in the following block diagram:



Q21) It is required to connect a Master, which generates data @ 200 Mega Samples/sec to a Slave which can receive the data @ 10 Mega Samples/Sec. If the data lasts in 10Micro Sec, what is the optimal size of FIFO to be used to avoid loose of data?

Answer:

Duration of the data = 10 Microsec

Input Data rate (Master) = 200 Mega samples

Output Data rate (Slave) = 10 Mega samples

$$\begin{aligned}\text{Depth of FIFO} &= (\text{Output rate} - \text{Input rate}) * \text{Duration} \\ &= (200-10) * 10 = 1900\end{aligned}$$

Q22) In a particular system, the sender sends data at the rate of 80 words / 100 clocks and the receiver can consume at the rate of 8 words / 10 clocks. Calculate the depth of FIFO so that no data is dropped under following assumptions:

- There is no feedback or handshake mechanism.
- Occurrence of data in that time period is guaranteed but exact place in those clock cycles is indeterminate.

Answer:

In the problem it is given that, out of 100 clocks the sender sends 80 words. The 80 words can occur in any of the 100 clocks. The worst case will be all 80 words coming continuously. So for 10 clocks, the sender sends 10 words where as the receiver can receive only 8 words. So we need to store 16 words in 100 cycles.

Now if we look at the process for long time, the worst case is: During first 100 clocks the sender is idle for 20 clocks and sends the data in the last 80 clocks. In the immediate 100 clocks, data is sent during first 80 clocks. In this case we need to store 32 words.

So the minimum size of FIFO required is 32 words.

Answer:

- This can be solved by taking an example.
- Let the frequency of clk_B be 100MHz.
- That implies, frequency of clk_A = clk_B/4 = 25MHz
- Period of en_B = $(1/25M) * 100 = 4 \mu s$
-

As duty cycle of en_B is 25%, it will be high for a duration of 1 μs . That means B receives the data for 1 μs and will be idle for 3 μs . Where as A sends the data every 0.04 μs . So in 4 μs it can transmit 100 words. And B receives 25 only, so we need to store the rest of 75 words in the FIFO. So the minimum size of FIFO required is 75 words.

Q24) What is the difference between Synchronous and Asynchronous FIFOs?

Answer:

A FIFO where writes to, and read from the FIFO buffer are conducted in the same clock domain is a Synchronous FIFO. For asynchronous FIFO two different and asynchronous clocks would be used. In Synchronous FIFO, as the read and write pointers will be incremented with the same clock it is easy to compare them and enable the FULL and EMPTY signals accordingly. Whereas for asynchronous FIFO this is slightly complicated and involves extra logic for the generation of FULL and EMPTY signals as the read and write pointers are getting incremented with two different clocks.
