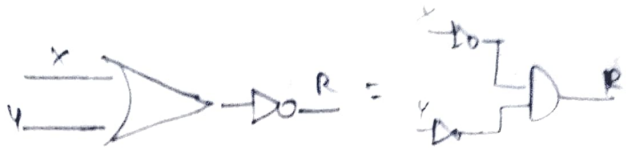
Sub Code: CS6105 Regno: 2019503579

Sub name: Digital Fundamentals and Computer Organization Name: Hemanth. N.

12.) De-Morgan's Law:

(I) $(x+y)' = x'.y'$



To prove this theorem, we consider complimentary laws

$x+x'=1$, $x.x'=0$. — (1)

Let us assume, $P = x+y$ ∴ P, x, y are logical variables.

By complementary law, $\boxed{P+P'=1 \text{ and } P.P'=0.}$ — (2)

If we prove (2) as true, De-morgan's Theorem is proved.

TO Prove: $P+P'=1$ (ie) $(x+y)+\bar{x}\bar{y}=1$

$P.P'=0$ (ie) $(x+y).\bar{x}\bar{y}=0$ } — (3)

$P+P'=1$: $(x+y)+\bar{x}\bar{y}=1$

Proof: $(x+y)+\bar{x}\bar{y} = (\bar{x}+y)+\bar{x}).(x+y)+y$ [$x+y+z=(x+y).(x+z)$]

$= [(x+\bar{x})+y][x+(y+\bar{y})]$ [$x+x'=1$]

$= (1+y).(x+1)$ [$1+x=1$]

$= 1.1$

$= \boxed{1}$

$P+P'=0$: $(x+y).x'y'=0$

Proof: $x'y'.(x+y) = x'y'x + x'y'y$ [$xy+xz=x.y+x.z$]

$= x\,x'y' + x'yy'$ ($x.x'=0$)

$= 0(y) + x'(0) = \boxed{0}$

Date: 03/01/2021 Pg no: 1 Signature:

$\therefore (x+y)' = x'y'$

(II) $(x \cdot y)' = x' + y'$

To Prove this, we consider complimentary Laws,

$x + x' = 1, \quad x \cdot x' = 0.$

Assume, $P = x + y \Rightarrow P \cdot P' = 0$ & $P + P' = 1$

i.e, $xy + (x'+y') = 1$ and $xy(x'+y') = 0$ — (1)

$xy + (x'+y')$

$= (xy + y') + x' \quad \because (x+y = y+x)$

$= (x'+y'+x)(x'+y'+y)$

Since, $(x+y)(x+z) = x + yz$

$= (x + x'y')(x + y + y')$

$= (1+y')(x+1)$

$= 1.1 = 1$

$\therefore xy + (x'+y') = 1$

$xy \cdot (x'+y') = 0$

$= xy \cdot (x'+y')$

$= xy\bar{x} + xy\bar{y}$

$(\because x(y+z) = xy + xz)$

$= xx'y + xyy'$

$= 0.y + x.0$

$= 0 + 0 = 0 \quad (x \cdot x' = 0)$

$\therefore xy \cdot (x'+y') = 0$

$\therefore (xy)' = x' + y'$

By Truth Table (I)

x	y	$(x+y)'$	$(x'.y')$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

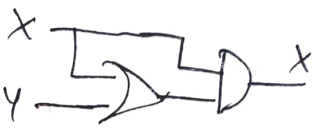
By Truth Table (II)

x	y	$(x.y)'$	$x'+y'$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

Thus, De-Morgan's Thm is Proved.

Absorption Law:

$X + XY = X$ (and) $X(X+Y) = X$



$X \cdot XY = X$

$= X \cdot 1 + X \cdot Y$ (Identity Law)

$= X(1+Y)$ (Distributive Law)

$= X \cdot 1$ (Boundness Law)

$= X$ (Identity Law)

Truth Table

X	Y	XY	X+XY
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

$\therefore X \cdot XY = X$

$X \cdot (X+Y) = X$

$\downarrow$

$X \cdot X + XY = XX + XY$ (Idempotent)

$= X(1+Y)$ ($1+Y = 1$)

$= X \cdot 1$

$= X$ (Identity Law)

$\therefore X(X+Y) = X$

Hence Proved

Subcode: CS6105

Subname: Digital Fundamentals and computer Organization

Reg no: 2019503519

Name: Hemanth.M

11.) $X = 1010110$; $Y = 1001011$

(a) $X - Y = X + (-Y)$

$Y = 1001011$; Y' = 2's complement of Y

$\Rightarrow$ 1's comp of

1's comp of Y is 0110100

$\therefore$ Y's 2's complement is 0110101

$\therefore X + (-Y) =$ $^{0}\,^{0}\,^{0}$
$1\,0\,1\,0\,1\,1\,0$
$(+)\ 0\,1\,1\,0\,1\,0\,1$
$0\,0\,0\,1\,0\,1\,1$

$\equiv (86)_{10}$
$\equiv (-75)_{10}$
$= (11)_{10}$

(b) $Y - X$

$X = 1010110$; 1's comp of X is 0101001

$\therefore$ 2's comp of X is 0101010

$\therefore Y - X = Y + (-X) =$ $1\,0\,0\,1\,0\,1\,1$
$0\,1\,0\,1\,0\,1\,0$
$1\,1\,1\,0\,1\,0\,1$

$\equiv (75)_{10}$
$\equiv (86)_{10}$
$= (11)_{10}$

$\therefore$ $X - Y = 0001011$

$Y - X = 1110101$

Date: 03/07/2021

Pg no: 4

Signature:

Subcode : CS6105

Subname : Digital Fundamentals and computer organization

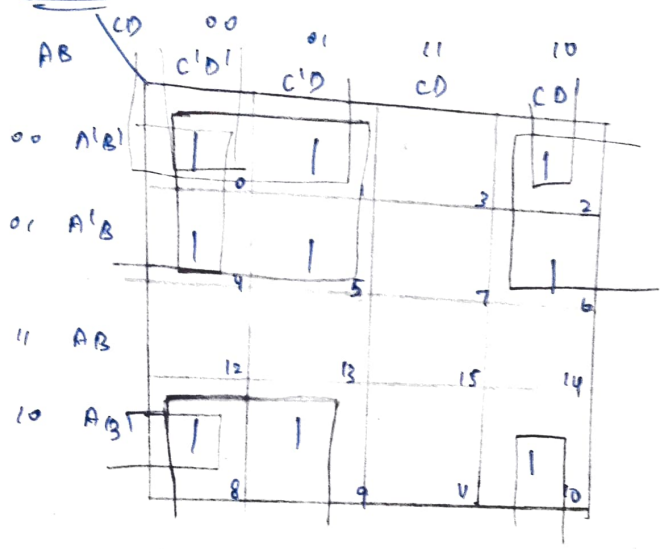
Reg No: 2019503519

Name : Hemanth.N.

13.) $F(A,B,C,D) = \Sigma(0,1,2,4,5,6,8,9,10)$

Sum terms given, fill in values. (1) value.

K-Map:



$= m_0 m_1 m_4 m_5 + m_2 m_0 m_8 m_{10} + m_0 m_4 m_2 m_6 + m_0 m_1 m_8 m_9$

Groups: (4) $A'C'$, $B'C'$, $A'D'$, $B'D'$

Simplifying, $A'C' + B'C' + A'D' + B'D'$

$= A'(C'+D') + B'(C'+D')$

$= \boxed{(A'+B') \cdot (C'+D')}$ → Product of sums form

Date : 03/07/2021

Pg No: 5

Signature:

Sub code: CS6105

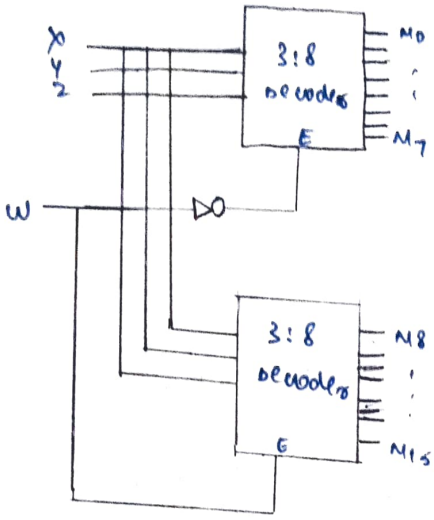
Sub name: Digital Fundamentals and computer Organizations

Reg no: 2019503519

Name: Hemanth I N.

(15.) 4:16 decoder :

[2 3:8 decoders used]



{2 3:8 decoder used to construct a 4:16 decoder}

M0 to M15 are the outputs. x, y, z are inputs

∴ [24 outputs possible for 4 i/ps]

(i) When w is equal to 0, top decoder is enabled, and bottom 3:8 decoder is disabled.

(ii) Top decoder minterms 0 to 7 i.e. 0000 to 0111

(iii) Bottom decoder o/ps → 0.

(iv) when w=1, bottom decoder is enabled and top decoder is disabled

(v) Now, top decoder o/ps → 0.

(vi) Bottom decoder minterms : 1000 to 1111 (8 to 15)

Use:

Decoder is basically a combination circuit that converts binary information from n input lines to max of 2^n unique o/p lines.

Date: 03/07/2021

Pg no: 6

Signature:

Sub code: CD6105

Sub name: Digital Fundamentals and computer organization

Reg no: 2019503519

Name: Hemanth I N.

(ie) When x, y, z, w are 0 $\Rightarrow M_0 = 1$, others 0

$x, y, w = 0;\ z = 1 \Rightarrow M_1 = 1$, others 0

$w, x, z = 0,\ y = 1 \Rightarrow M_2 = 1$, others 0

$w = 0,\ x = 1, y = 1, z = 1 \Rightarrow M_7 = 1$, others 0

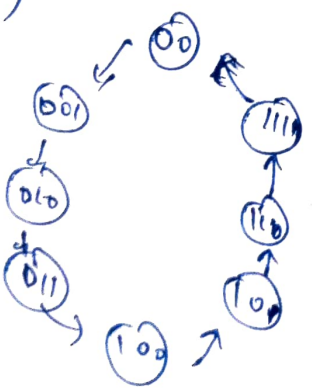
$w = 1,\ x = 0, y = 0, z = 0 \Rightarrow M_8 = 1$, others 0

$w = 1,\ x, y, z = 1 \Rightarrow M_{15} = 1$, others = 0

These are truth table values, this 16 outputs possible for 4 inputs.

[4:16 decoder is constructed using two 3:8 decoders]

18.)



Present			Next			FF i/ps		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A_2}	T_{A_1}	T_{A_0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	0	1
1	1	0	0	0	1	1	1	1
1	1	1	0	0	0	1	1	1

① State table ② FF eqn KMap ③ simplify ④ Circuit Drawing using FF & combinational circuit

Date: 03/07/2021

Pg no: 7

Sig nature:

First state diagram is drawn, then state table is obtained,

T FF truth Table

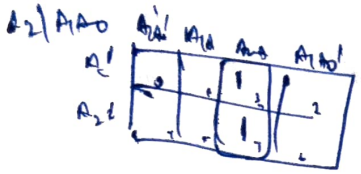
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Excitation Table

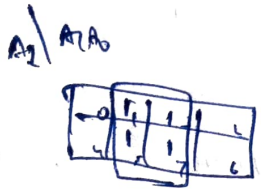
Char Table

Q_n	T	Q_{n+1}
0	0	0 (Qn)
0	1	1
1	0	1
1	1	0

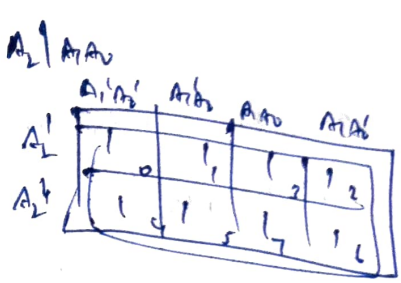
∴ Now, the prev state table is to be simplified,



$T_{A_2} = A_1 A_0$

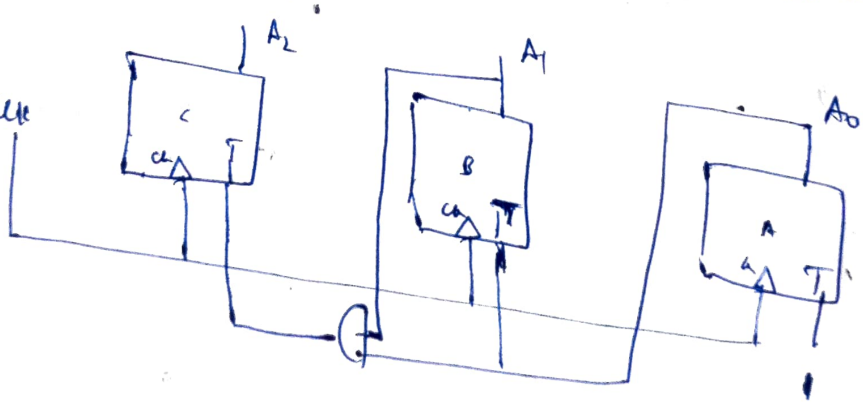


$T_{A_1} = A_0$



$T_{A_0} = 1$

After obtaining the FF eqns, draw the seq circuit usg FFs and combinational circuits.



Seq circuit diagram of Jel circuit usg Bin counter

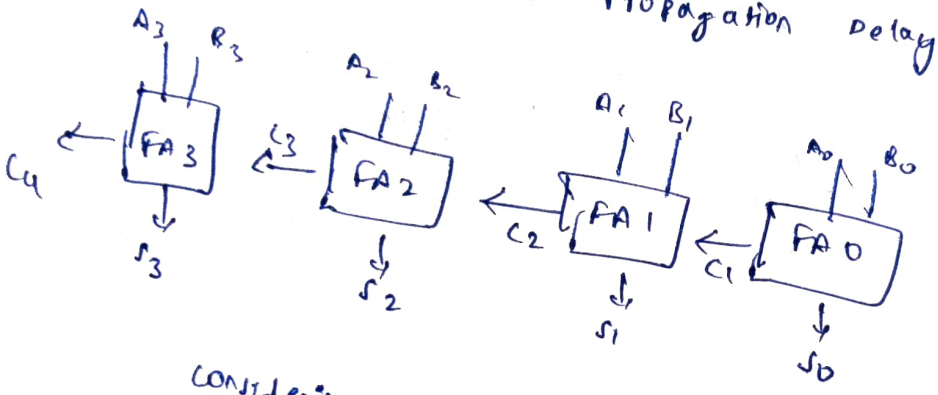
Sub code : CS6105
Sub Name : Digital Fundamentals and Computer Organization

Reg No: 2019503519
Name : Hemanth. N.

14.) Carry Look Ahead Adder:

In ripple carry adder, for each adder block, 2 bits that are added are available instantly. But carry is from previous block, waits. So, it is not possible to generate sum and carry until previous block is known.

The ith block waits for o/p from (i-1)th block. So there is a lag in time called Propagation Delay.



Considering above S_3 is produced by FA 3 as soon as i/p is given C_4 is not available until C_3 is known is given on.

A look ahead reduces the propagation delay by introducing more complex hardware. In this one, the ripple carry is transformed such that carry logic over fixed groups of bits of adder is reduced to 2 logic level.

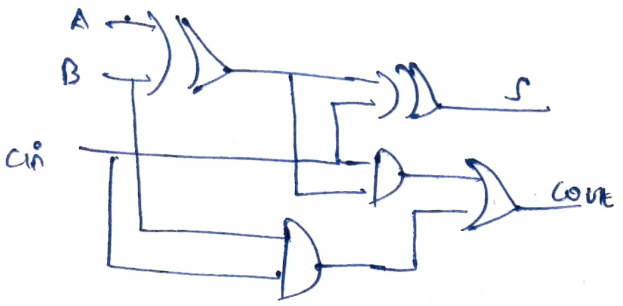
Date: 03/07/2021
Pg No: 9
Signature:

Sub Code: CS6105

Sub Name: Digital Fundamentals and Computer Organization

Reg No: 2019503519

Name: Hemanth.N.



A	B	C	C_{i+1}	condition
0	0	0	0	
0	0	1	0	No carry
0	1	0	0	
0	1	1	1	generate

No carry

A	B	C	C_{i+1}	condition
1	0	0	0	
1	0	1	1	Propagate
1	1	0	1	
1	1	1	1	carry Generate

From circuit, we define carry generate and carry Propagate.

$G_i = A_i B_i$ & $P_i = A_i \oplus B_i$

G_i gives carry when A_i, B_i are regardless of carry P_i is associated with propagation of carry.

$\Rightarrow$ Carry $C_{i+1} = G_i + P_i C_{in}$

$C_1 = G_0 + P_0 C_{in}$

$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_{in}$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{in}$

$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 (C_{in})$

From above eqns we find that carry at any stage of circuit is independent of carry from previous one, this can be implemented with one level of AND two gates

Date: 03/07/2021

Pg No: 10

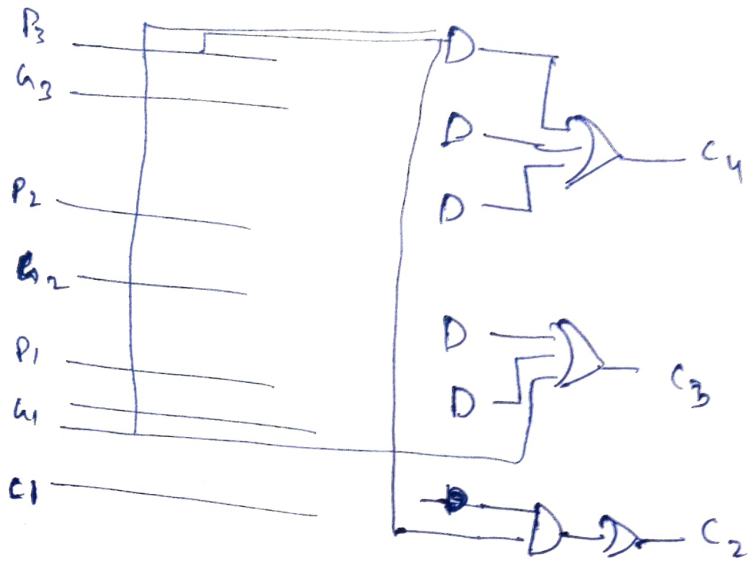
Signature:

Sub code: CS6105

Sub Name: Digital Fundamentals and Computer Organization

Reg No: 2019503579

Name: Hemanth, N.



Carry Look Ahead Generator (Circuit diagram)

The equations are derived.

C_1 C_2 C_3 C_4 ✓

→ Logic behind the same is also explained clearly

Date: 03/07/2021

Pg No: 11

Signature:

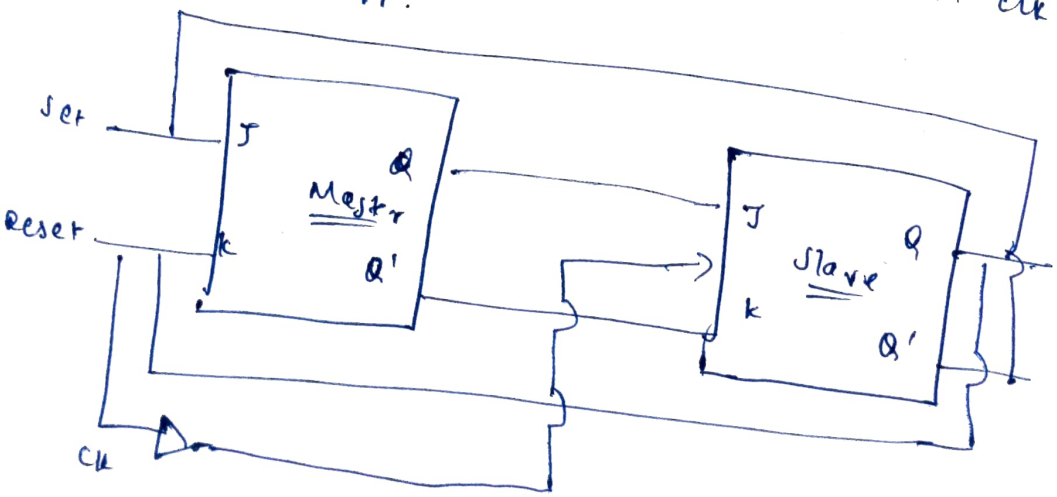
16.) Master Slave FF:

This type of FF can be designed with 2 JK FF's by connecting in series. One of these FFs, one FFs works as master as well as FF works as a slave.

Master ~~slave~~ FF connected to i/ps of slave FF.

Here slave FFs o/ps can be connected to i/ps of master FF.

The inverter connection can be done in such a way that where inverted CLK pulse, can be ~~connected~~. connected to slave FF. CLK pulse is 0 for master FF, then CLK pulse will be 1 for a slave FF. Illy CLK pulse is 1 for master FF, then CLK pulse will be 0 for slave FF.



Sub code: CS6105

Sub name: Digital Fundamentals and computer organisation

Reg no: 2019503519

Name: Hemanth.N

Master slave FF:

Master slave ff can be designed:

When CLK Pulse goes to high which means 1, then slave can be separated, inputs J & K may change condition & system

The slave FF can be is detached until the CLK pulse goes to low which means to 0. whenever CLK pulse goes back to low-state, then data can be transmitted from Master to slave.

Master FF triggered at a pos level, slave → neg level

∴ Master FF responds first

(1) J=0, K=1 → Master FF 'Q' goes to K of slave Q

Clk forces the slave ff to Reset, ∴ slave copies Master

(2) J=1, K=0 Master Q goes to J of slave, Clk, neg tran

Set slave FF & copies master

(3) J=1, K=1, toggles, over CLK is positive transition, then slave toggles over CLK neg transition

(4) J,K=0, FF immobilized & Q remains unmovable.

Timing diagram

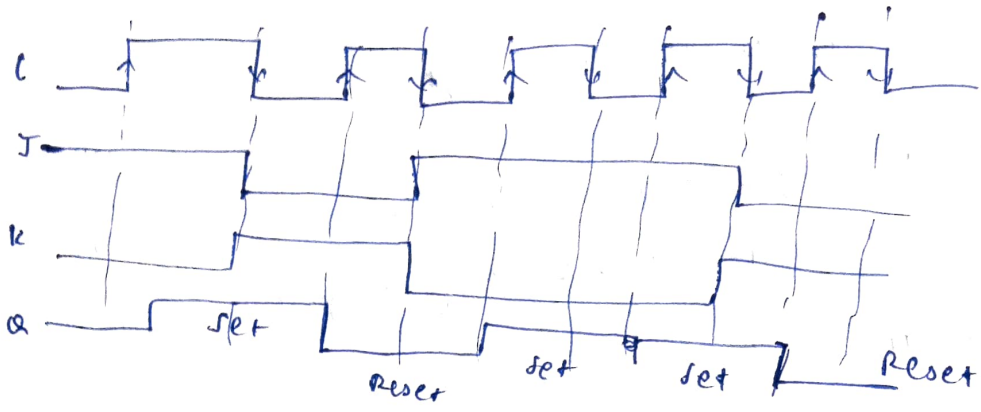
Toggling takes place for clk cycle

Sub Code: CS6105

Sub name: Digital Fundamentals and computer organization

Reg no: 2019503579

Name: Hemanth, N.



Timing Diagram of Master Slave FF

(3) When clk pulse is 1, Master is set, not slave, ∴ Slave o/p is 0, clk is 1.

(4) clk is low, slave turns into operational & remains "1" until clk again turn to '0'.

(5) Toggling takes place throughout the entire procedure & while the o/p is altering one time within a cycle.

∴ Synchronous ~~opp~~.

Master circuit → Activates over leading edge of clk pulse
Slave ff → falling edge → Activated

Date: 03/07/2021

Pg no: 14

Signature:

Sub Code: CS6105

Sub Name: Digital Fundamentals and Computer Organization.

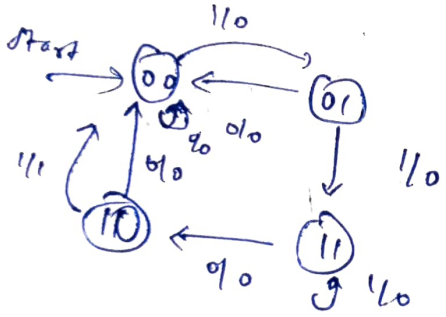
Reg No: 2019503519

Name: Hemanth.N.

(9.) Design of a synchronous sequential circuit starts from a set of specifications and culminates in a logic diagram or a list of Boolean functions from which a logic diagram can be obtained.

A synchronous seq circuit is made up of FFs and combinational gates.

(1) Derive the state diagram.



(2) Convert to state table

Present			Next		
Q_1	Q_2	X	Q_1	Q_2	F
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	1	0	0
			1	1	1

(3) Choose a FF and include their excitation table in state transition table. Say we choose D FF.

State Table ⇒

D_1	D_2
0	0
0	1
0	0
1	1
0	0
0	0
1	0
1	1

for above transition table

Date: 03/07/2021

Pg No: 15

Signature:

Sub Code: CS6105

Sub name: Digital Fundamentals and computer organization

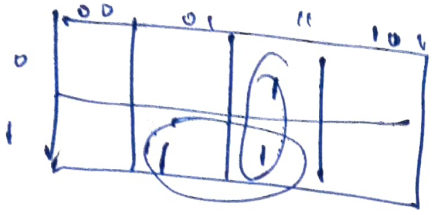
Reg No: 2019503519

Name: Hemanth.N.

④ Minimize the func. for FFs

~~Q_1Q_2X~~

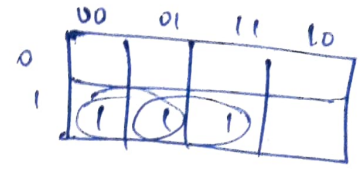
$X \mid Q_1Q_2$



$D_1 = Q_2X + Q_1Q_2$

$D_1 = Q_2(X + Q_1)$

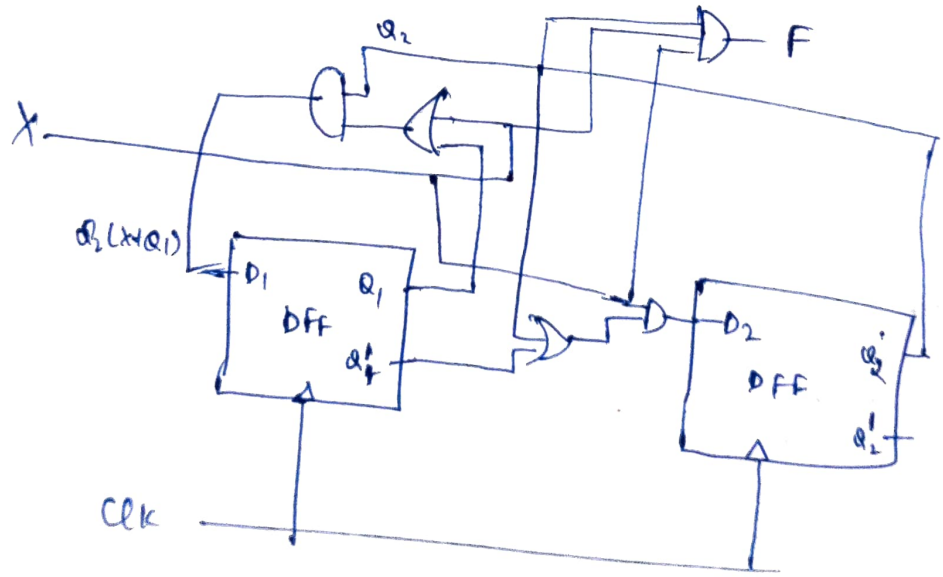
$X \mid Q_1Q_2$



$D_2 = Q_1'X + Q_2X$

$D_2 = (Q_1' + Q_2)X$

⑤ Use simplified functions obtained to D_1 D_2 represent and Draw the final circuit.



Circuit

Synchronous Sequential circuit Designed

Signature:

Sub Code: CS6105

Sub Name: Digital Fundamentals and computer organization

Reg NO: 2019503519

Name: Hemanth, N.

PART-C

23) Magnitude comparator:

It is a combinational circuit that compares two numbers to find whether 2 nos are equal or) either one is greater or smaller than the other one.

Explanation for 2 bit Magnitude comparator

A_1	A_0	B_1	B_0	$A < B$	$A = B$	$A > B$
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

Date: 03/07/2021

Pg NO: 17

Signature:

Sub Code: CS6105

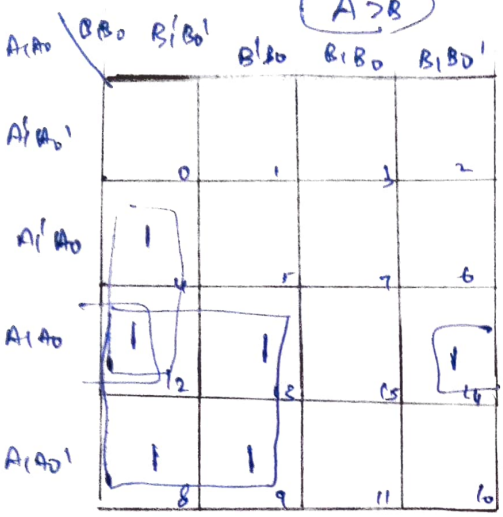
Sub Name: Digital Fundamentals and Computer Organization

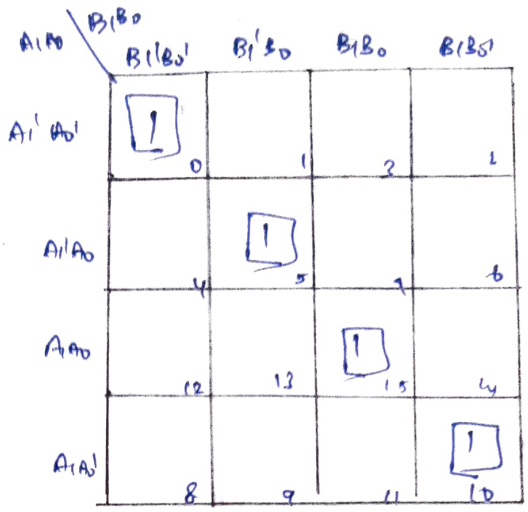
Reg No: 2019503519

Name: Hemanth I N.

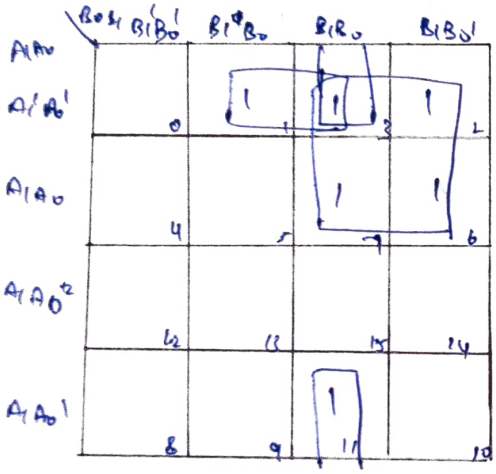
Deriving expressions for $A=B$, $A<B$ and $A>B$.

$A>B$





$A>B$



$A<B$

$A=B$: $A_1' A_0' B_0' B_1' +$
$A_1' A_0 B_1' B_0 +$
$A_1 A_0' B_0 B_1' + A_1 A_0 B_0 B_1$

$A>B$: $A_0 B_1' B_0' + A_1 A_0 B_0' + A_1 B_1'$

$A<B$: $A_1' A_0' B_0 + A_1' B_1 + A_0' B_1 B_0$

From k-Map simplifications, we obtain this.

Final expressions for comparator is derived, these are now designed.

Date: 03/07/2021

Pg No: 18

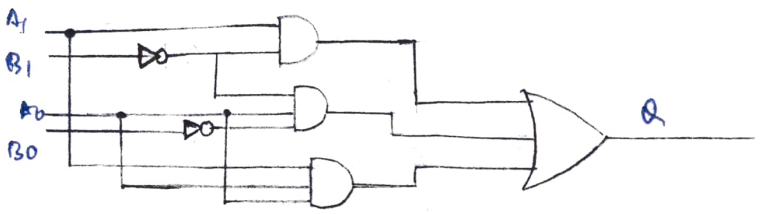
Signature:

Sub Code: CS6105

Sub name: Digital Fundaments and computer Organization

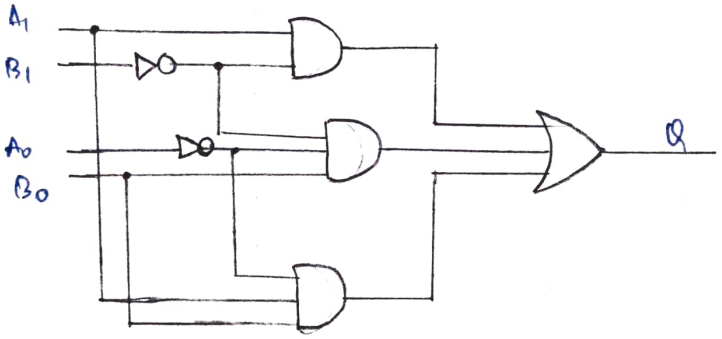
Reg no: 2019503519

name: Hemanthin.



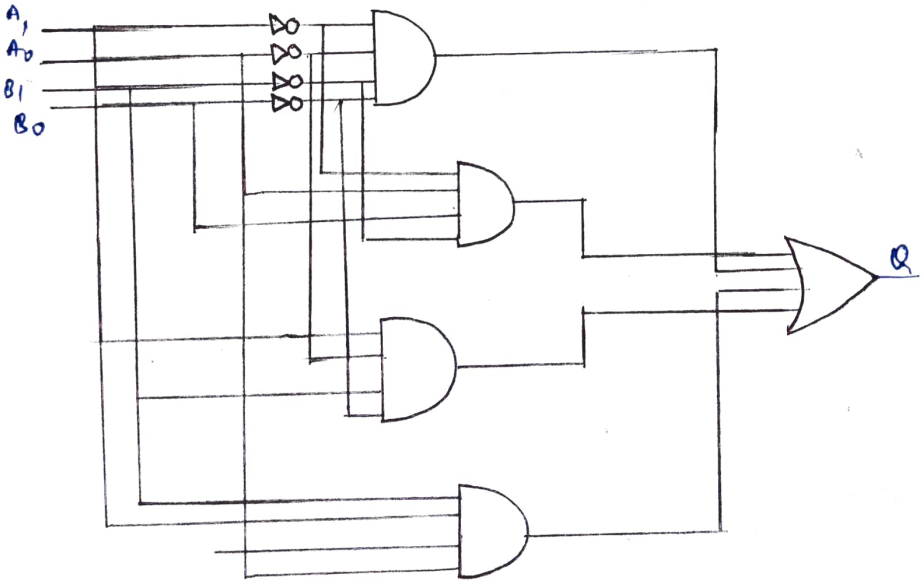
Greater than comparator

$Q = A_1 B_1' + A_0 B_0' B_1' + A_1 A_0 B_0'$



Less than comparator

$Q = B_1 A_1' + A_1' A_0' B_0 + A_0' B_0 B_1$



Equal to comparator

$Q = A_1' A_0' B_1' B_0' + A_1' B_1' A_0 B_0 + A_1 A_0' B_1 B_0' + A_0 A_1 B_0 B_1$

Circuit Diagrams for comparator.

Pg no: 19

Signature:

Date:

Working:

Comparator compares each bit in one number with each corresponding bit in another number.

A : A_1A_0 and B : B_1B_0 are 2 numbers

Output logic of comparator A=B :

$A_1=B_1$ and $A_0=B_0$ => Numbers are equal.

Output is 1 if $A_1=B_1$ & $A_0=B_0$

Output logic of comparator A>B :

$A_1=1$ and $B_1=0$, => A > B (1)

If $A_1=1$, $B_1=1$,

Checks A_0, B_0 , $A_0=1$, $B_0=0$ => A > B (1)

Output logic of comparator A<B :

$A_1=0$ & $B_1=1$ => A < B (1)

If , $A_1=1$, $B_1=1$ => check

A_0, B_0 ; $A_0=0$, $B_0=1$ => A < B (1)

Thus, working of magnitude comparator is detailed discussed.

Sub code : CS6105

Sub name : Digital Fundamentals and computer organisation

Reg NO : 2019503519

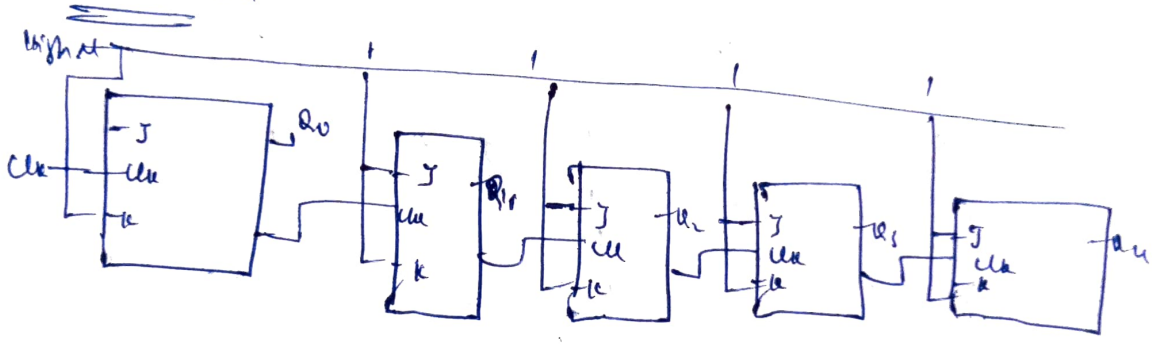
Name : Hemanth, N.

24.) Ripple counter of n bit is a counter that can count upto 2^n states and also known as mod n counters.

It is a Asynchronous counter because of the way clock pulse ripple its way.

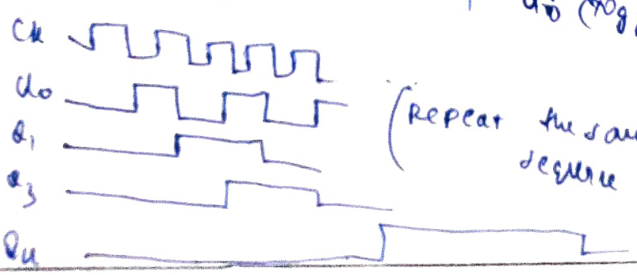
FFs are in toggle. Clk pulses is applied at LSB, FFs are in toggle mode.

JK FF is used



Q_0 is LSB & will toggle for every Clk pulse

Clk	J	K	Q_0
↓	0	0	Q_0 (no change)
↓	0	1	0
↓	1	0	1
↓	1	1	$\bar{Q}_0$ (toggle)



(Repeat the same sequence)

} (Sample Clk pulse diagram)

Date : 03/07/2021

Pg no : 21

Signature :

Sub code: CS6105

Reg No: 2019502579

Sub name: Digital Fundamentals and computer organisation

Name: Hemanth.N

Binary count sequence

LSB, toggle for every clk pulse

Counter state	Q_4	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1
16	1	0	0	0	0
17	1	0	0	0	1
18	1	0	0	1	0
19	1	0	0	1	1
20	1	0	1	0	0
21	1	0	1	0	1
22	1	0	1	1	0
23	1	0	1	1	1
24	1	1	0	0	0
25	1	1	0	0	1
26	1	1	0	1	0
27	1	1	0	1	1
28	1	1	1	0	0
29	1	1	1	0	1
30	1	1	1	1	0
31	1	1	1	1	1

Date: 24/7/2021

Pg No: 23

Signature:

Sub code : CS6105

Sub name : Digital Fundamentals and computer organization

Reg NO : 2019503579

Name : Hemanthin.

PART-A

(1) $(B41F)_{16}$ to decimal

$(11 \times 16^3) + (4 \times 16^2) + (1 \times 16) + (15 \times 16^0)$

$= 45056 + 1024 + 16 + 15$

$= (46111)_{10}$

$\therefore \boxed{(B41F)_{16} = (46111)_{10}}$

(2) $xy + x'z + yz + z'$

$xy + (x'z + z') + yz$ (Absorption Law) $AB + A' = B + A'$

$xy + x' + z' + yz$

$(xy + z') + (yz + z')$ (Absorption Law)

$= x'y + y + z'$ (Idempotent Law)

$= \boxed{x' + y + z'}$

Date : 03/07/2021

Signature :

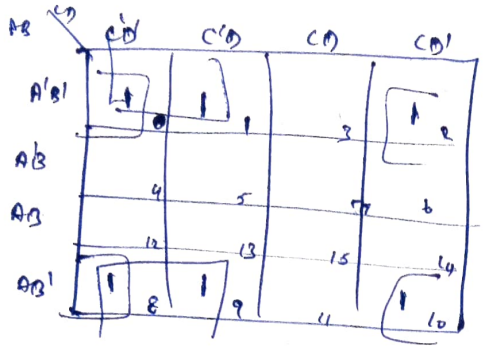
Reg NO: 2019503519

Name: Hemanth. N.

Sub Code: CSB105

Sub name: Digital Fundamentals and computer organization

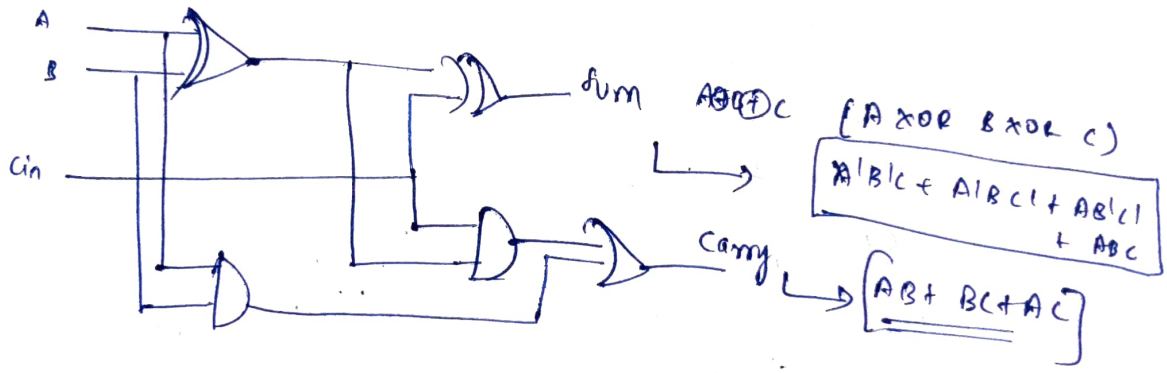
③ $F(A,B,C,D) = \sum(0,1,2,8,9,10)$



$B'C' + B'D'$

$= B'(C' + D')$

4.) Full Adder:



Sum: $A \oplus B \oplus C$ (A XOR B XOR C)

$A'B'C + A'BC' + AB'C' + ABC$

Carry: $AB + BC + AC$

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

$= A'(B \oplus C) + A(B \oplus C)' = A \oplus B \oplus C$

A	B	C	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Carry: $(A \oplus B)C + AB = (A'B + AB')C + AB = A'BC + AB'C + AB$

$= A'BC + AB'C + AB(1+C) = A'BC + ABC + AB'C + AB$

$= BC + AB'C + AB(1+C) = BC + AB + A(C(B+B))$

$= AB + BC + AC$

Date: 03/07/2021

Pg NO: 24

Signature:

Sub code: CS6105

Sub Name: Digital Fundamentals and Computer Organisation

Reg No: 2019503519

Name: Hemanth.N.

5.) Priority Encoder:

Circuit that compresses multiple Bin i/ps into smaller no of o/ps.

If 2^n inputs are given, n is no of o/ps

It has one priority value.

It o/ps the highest order i/p first.

4 to 2 Priority Encoder $I_3(3), \ldots I_1(0)$

I_3 has high priority ⇒ It is output first.

6.a) (i) Given the circuit diag we have to develop the state table and state Diagram.

(ii) Find out the i/ps and o/ps equations from given diag

(iii) Generate next state table, using FF eqns

(iv) Now Draw state Diagram.

(Some FF have asynchronous i/p's used to force FF to particular state)

Input that sets FF to 1 is called Preset

I/p that clears FF is called Clear

& Row clear Lgns are found.

m FF, n inputs ⇒ 2^{m+n} Rows in state table

Date: 03/07/2021

Pg no: 25

Signature:

Sub code: CS6105

Sub name: Digital Fundamentals and Computer Organisation

Reg No: 2019503519

Name: Hemanth. N.

7.) → Ring counter is a typical application of shift register.

only change is o/p of last FF is connected to first FF in Ring counter.

→ count no of loops

→ The no of states in Ring counter is no of FFs used.

→ To generate timing diagram with combination of ~~signals~~ shift register and decoder, is possible.

→ Each FF is in state 1, once every four clk cycles, and produces one of four timing signals. o/p of last FF is linked to I/P of first FF.

8.) Gray coding: only one bit changes b/w current state & next state.

Pros:

→ Gray coding is used in glitch free circuits, Async FIFO design, high speed decoding circuit

→ Gray coding overcomes problems in binary coding like transition to dead, unwanted.

Cons:

control logic is quite complex.

no of FF is no of FF in binary state machine

Date: 03/07/2021

Pg no: 26

Signature:

Sub Code : CS6105
Sub Name : Digital Fundamentals and Computer Organisation

Reg No : 2019503519
Name : HEMANTH.N.

One Hot Encoding :

It takes one FF for each state.

Pros :

Fast and control logic is simple

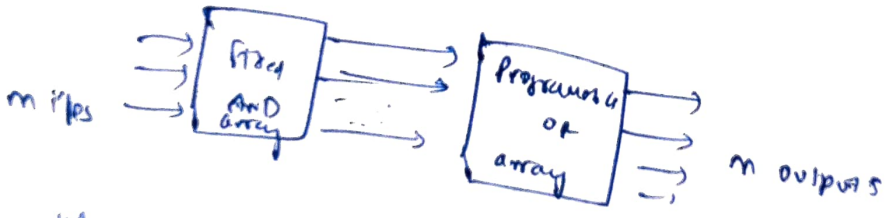
Used in FPGA.

Cons :

More Hardware is required.

9.) Yes, ROM are Programable and they are called as Programable Read only memory.

It is programable logic device that has fixed AND array & programable OR array



When bit PROM is created, all bits read as 1. Bits to be changed to 0 is etched & burned into a chip.

Signature:

Sub code : CS6105

Reg No : 2019503519

Sub Name : Digital Fundamentals and comp organisation

Name : Hemanth. N.

(b.)

- Fetch instruction
- Decode information
- Perform ALU operation
- Access Memory
- Update Register File
- Update the program counter

All the answers in this answer Booklet has been in my own handwriting. No body helped me in writing these answers.

03/07/2021

Date : 03/07/2021

Pg no : 28

Signature :