

# Subject Name (Gujarati)

4321102 -- Summer 2024

Semester 1 Study Material

Detailed Solutions and Explanations

## પ્રશ્ન 1(અ) [3 માર્ક્સ]

કન્વર્ટ કરો:  $(110101)_2 = (\underline{\hspace{1cm}})_{10} = (\underline{\hspace{1cm}})_8 = (\underline{\hspace{1cm}})_{16}$

જવાબ

સ્ટેપ-બાય-સ્ટેપ કન્વર્ઝન  $(110101)_2$  :

બાઇનરી $(110101)_2$	ડેસિમલ	ઓક્ટલ	હેક્ઝાડેસિમલ
$1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$	$32+16+0+4+0+1 = 53$	$6 \times 8^1 + 5 \times 8^0 = 48 + 5 = 53$	$3 \times 16^1 + 5 \times 16^0 = 48 + 5 = 35$
$(110101)_2$	$(53)_{10}$	$(65)_8$	$(35)_{16}$

મેમરી ટ્રીક

“બાઇનરી ડિજિટ આઉટ ડિયર” (BDOH) બાઇનરી કન્વર્ઝન માટે.

## પ્રશ્ન 1(બ) [4 માર્ક્સ]

\*\*કરો: (i)  $(11101101)_2 + (10101000)_2$  (ii)  $(11011)_2 * (1010)_2$  \*\*

જવાબ

બાઇનરી સરવાળા અને ગુણાકાર માટે ટેબલ:

(i) બાઇનરી સરવાળો	(ii) બાઇનરી ગુણાકાર
$11101101$	$11011$
$+ 10101000$	$\times 1010$
-----	-----
$110010101$	$00000$
	$11011$
	$00000$
	$11011$
	-----
	$11101110$

ડેસિમલ વેરિફિકેશન:

- 1.  $(11101101)_2 = 237, (10101000)_2 = 168, = 405 = (110010101)_2$
- 1.  $(11011)_2 = 27, (1010)_2 = 10, = 270 = (11101110)_2$

મેમરી ટ્રીક

સરવાળા માટે “કેરી અપ મેક્સ સમ” અને ગુણાકાર માટે “શિફ્ટ લેફ્ટ એડ પ્રોડક્ટ”.

## પ્રશ્ન 1(ક) [7 માર્ક્સ]

(i) કન્વર્ટ કરો:  $(48)_{10} = (\underline{\hspace{1cm}})_2 = (\underline{\hspace{1cm}})_8 = (\underline{\hspace{1cm}})_{16}$

પ્રશ્ન 1(ક) અથવા [7 માર્ક્સ]

કોડ્સ સમજાવો: ASCII, BCD, Gray

જવાબ

સામાન્ય ડિજિટલ કોડ્સનું ટેબલ:

કોડ	વર્ણન	ઉદાહરણ
ASCII (American Standard Code for Information Interchange)	128 કેરેક્ટર્સને રજૂ કરતો 7-બિટ કોડ જેમાં આલ્ફાબેટ્સ, નંબર્સ અને સ્પેશિયલ સિમ્બોલ્સ શામેલ છે	A = 65 (1000001) <sub>2</sub>
BCD (Binary Coded Decimal)	દરેક ડેસિમલ અંક (0-9) ને 4 બિટ્સનો ઉપયોગ કરીને રજૂ કરે છે	42 = 0100 0010
Gray Code	બાઇનરી કોડ જેમાં આસપાસના નંબરો માત્ર એક બિટથી અલગ પડે છે	(0,1,3,2) = (00,01,11,10)

ડાયાગ્રામ: ત્રે કોડ જનરેશન:

```

flowchart LR
    A[" "] -- "{--}" --> B[" "]
    B -- "{--}" --> C["MSB "]
    C -- "XOR" --> D[" : 0011"]
    D -- "{--}" --> E[" : 0010"]
    
```

મેમરી ટ્રીક

“ઓલવેઝ બાઇનરી જનરેટર્સ” - દરેક કોડનો પ્રથમ અક્ષર (ASCII, BCD, Gray).

પ્રશ્ન 2(અ) [3 માર્ક્સ]

બુલિયન બીજગણિતનો ઉપયોગ કરીને સરળ બનાવો:  $Y = A B + A' B + A' B' + A B'$

જવાબ

સ્ટેપ-બાય-સ્ટેપ સરળીકરણ:

સ્ટેપ	એક્સપ્રેશન	બુલિયન નિયમ
$Y = A B + A' B + A' B' + A B'$	પ્રારંભિક એક્સપ્રેશન	-
$Y = A(B + B') + A'(B + B')$	ફેક્ટરિંગ	ડિસ્ટ્રિબ્યુટિવ લૉ
$Y = A(1) + A'(1)$	કોમ્પ્લિમેન્ટ લૉ	$B + B' = 1$
$Y = A + A'$	સરળીકરણ	-
$Y = 1$	કોમ્પ્લિમેન્ટ લૉ	$A + A' = 1$

મેમરી ટ્રીક

બુલિયન સરળીકરણ સ્ટેપ્સ માટે “ફેક્ટર, સિમ્પ્લિફાય, ફિનિશ”.

પ્રશ્ન 2(બ) [4 માર્ક્સ]

K-મેપનો ઉપયોગ કરીને નીચેના બુલિયન ફંક્શન ને સરળ બનાવો:  $f(A,B,C,D) = \sum m(0,3,4,6,8,11,12)$

જવાબ

K-મેપ સોલ્યુશન:

AB

CD	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	0	1	0	0
10	0	0	1	0

ગ્રુપિંગ:

- ગ્રુપ 1:  $m(0,8) = A'C'D'$
- ગ્રુપ 2:  $m(4,12) = BD'$
- ગ્રુપ 3:  $m(3,11) = CD$
- ગ્રુપ 4:  $m(6) = A'B'CD'$

સરળ કરેલ એક્સપ્રેશન:  $f(A,B,C,D) = A'C'D' + BD' + CD + A'B'CD'$

### મેમરી ટ્રીક

K-મેપ ગ્રુપિંગ સ્ટ્રેટેજી માટે ``ગ્રુપ પાવર્સ ઓફ ટુ''.

## પ્રશ્ન 2(ક) [7 માર્ક્સ]

NOR ગેટને સ્વચ્છ આકૃતિઓ સાથે યુનિવર્સલ ગેટ તરીકે સમજાવો.

જવાબ

NOR એક યુનિવર્સલ ગેટ:

ફંક્શન	NOR નો ઉપયોગ કરી ઇમ્પ્લિમેન્ટેશન	ટ્રુથ ટેબલ
NOT ગેટ		A
		0
		1
AND ગેટ		A B
		0 0
		0 1
		1 0
		1 1
OR ગેટ		A B
		0 0
		0 1
		1 0
		1 1

ડાયાગ્રામ: NOR ઇમ્પ્લિમેન્ટેશન:

```

flowchart TD
    A["NOT: A {-1{-} A"}]
    B["AND: A {-{-}1{-}{-}|}
        | 1{{-}{-} A•B}
        B {-{-}1{-}{-}|"}]
    C["OR: A {-{-}1{-}{-}|}
        | |{{-}{-} A+B}
        B {-{-}1{-}{-}|"}]

```

### મેમરી ટ્રીક

NOR ગેટ ઇમ્પ્લિમેન્ટેશન માટે ``NOT AND OR, NOR કરે મોર''.

## પ્રશ્ન 2(અ) અથવા [3 માર્ક્સ]

બુલિયન સમીકરણ માટે લોજિક સર્કિટ દોરો:  $Y = (A + B') \cdot (A' + B') \cdot (B + C)$

## જવાબ

### લોજિક સર્કિટ ઇમ્પ્લિમેન્ટેશન:

```

flowchart TD
    A["A"] --{-{-} D["OR"]}
    B["B"] --{-{-} D
    D --{-{-} G["AND"]}
    A1["A"] --{-{-} E["OR"]}
    B1["B"] --{-{-} E
    E --{-{-} G
    B2["B"] --{-{-} F["OR"]}
    C["C"] --{-{-} F
    F --{-{-} G
    G --{-{-} Y["Y"]}
    
```

### ટ્રુથ ટેબલ વેરિફિકેશન:

- ટર્મ 1:  $(A + B')$
- ટર્મ 2:  $(A' + B')$
- ટર્મ 3:  $(B + C)$
- આઉટપુટ:  $Y = \text{Term1} \cdot \text{Term2} \cdot \text{Term3}$

## મેમરી ટ્રીક

જટિલ એક્સપ્રેશન માટે ``દરેક ટર્મ અલગથી``.

## પ્રશ્ન 2(બ) અથવા [4 માર્ક્સ]

ડી-મોર્ગન્સના પ્રમેય લખો અને તેને સાબિત કરો.

## જવાબ

### ડી-મોર્ગન્સ પ્રમેય અને પ્રૂફ:

પ્રમેય	સ્ટેટમેન્ટ	ટ્રુથ ટેબલ દ્વારા પ્રૂફ
પ્રમેય 1	$(A \cdot B)' = A' + B'$	A B
		0 0
		0 1
		1 0
		1 1
પ્રમેય 2	$(A + B)' = A' \cdot B'$	A B
		0 0
		0 1
		1 0
		1 1

### ડાયાગ્રામ: ડી-મોર્ગન્સ લો વિઝ્યુલાઇઝેશન:

```

flowchart TB
    A["(A \cdot B)"] --{-{-} B["AND OR"]
    B --{-{-} D["OR AND"]
    C["(A + B)"] --{-{-} D["OR AND"]
    D --{-{-} Y["Y"]
    
```

## મેમરી ટ્રીક

ડી-મોર્ગન્સ લો લાગુ કરવા માટે ``બાર તોડો, ઓપરેશન બદલો, ઇનપુટ ઇન્વર્ટ કરો``.

પ્રશ્ન 2(ક) અથવા [7 માર્ક્સ]

સિમ્બોલ, ટ્રુથ ટેબલ અને સમીકરણની મદદથી તમામ લોજિક ગેટ્સ સમજાવો.

જવાબ				
લોજિક ગેટ્સ સમરી:				
ગેટ	સિમ્બોલ	ટ્રુથ ટેબલ	સમીકરણ	વર્ણન
AND		A B	Y	$Y = A \cdot B$
		0 0	0	
		0 1	0	
		1 0	0	
		1 1	1	
OR		A B	Y	$Y = A + B$
		0 0	0	
		0 1	1	
		1 0	1	
		1 1	1	
NOT		A	Y	$Y = A'$
		0	1	
		1	0	
NAND		A B	Y	$Y = (A \cdot B)'$
		0 0	1	
		0 1	1	
		1 0	1	
		1 1	0	
NOR		A B	Y	$Y = (A + B)'$
		0 0	1	
		0 1	0	
		1 0	0	
		1 1	0	
XOR		A B	Y	$Y = A \oplus B$
		0 0	0	
		0 1	1	
		1 0	1	
		1 1	0	
XNOR		A B	Y	$Y = (A \oplus B)'$
		0 0	1	
		0 1	0	
		1 0	0	
		1 1	1	

મેમરી ટ્રીક

“All Operations Need Necessary eXecution” (દરેક ગેટનો પહેલો અક્ષર - AND, OR, NOT, NAND, NOR, XOR).

પ્રશ્ન 3(અ) [3 માર્ક્સ]

સંક્ષિપ્તમાં 4:2 એન્કોડર સમજાવો.

જવાબ		
4-to-2 એન્કોડર ઓવરવ્યુ:		
ફંક્શન	વર્ણન	ટ્રુથ ટેબલ
4:2 એન્કોડર	4 ઇનપુટ લાઇન-સને 2 આઉટપુટ લાઇન-સમાં કન્વર્ટ કરે છે	$I_0 I_1 I_2 I_3$
	એક સમયે માત્ર એક જ ઇનપુટ એક્ટિવ	1 0 0 0

ઇનપુટ પોઝિશન બાઇનરીમાં એન્કોડેડ

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

ડાયાગ્રામ: 4:2 એન્કોડર:

```
flowchart TD
    I0["I_{0}"] --> E["4:2"]
    I1["I_{1}"] --> E
    I2["I_{2}"] --> E
    I3["I_{3}"] --> E
    E --> Y1["Y_{1}"]
    E --> Y0["Y_{0}"]
```

મેમરી ટ્રીક

એન્કોડર ફંક્શન માટે ``ઇનપુટ પોઝિશન ક્રિએટ્સ આઉટપુટ``.

### પ્રશ્ન 3(બ) [4 માર્ક્સ]

કુલ એડર બ્લોક્સનો ઉપયોગ કરીને 4-બિટ પેરેલલ એડરને સમજાવો.

જવાબ

4-બિટ પેરેલલ એડર:

કોમ્પોનન્ટ	ફંક્શન
કુલ એડર	3 બિટ્સ (A, B, Carry-in) ને એડ કરે છે અને Sum અને Carry-out આપે છે
પેરેલલ એડર	4 કુલ એડર્સને કેરી પ્રોપેગેશન સાથે જોડે છે

ડાયાગ્રામ: 4-બિટ પેરેલલ એડર:

```
flowchart LR
    A0["A_{0}"] --> FA0["FA"]
    B0["B_{0}"] --> FA0
    C0["C_{0=0}"] --> FA0
    FA0 --> S0["S_{0}"]
    FA0 --> C1["C_{1}"]
    C1 --> FA1["FA"]
    A1["A_{1}"] --> FA1
    B1["B_{1}"] --> FA1
    FA1 --> S1["S_{1}"]
    FA1 --> C2["C_{2}"]
    C2 --> FA2["FA"]
    A2["A_{2}"] --> FA2
    B2["B_{2}"] --> FA2
    FA2 --> S2["S_{2}"]
    FA2 --> C3["C_{3}"]
    C3 --> FA3["FA"]
    A3["A_{3}"] --> FA3
    B3["B_{3}"] --> FA3
    FA3 --> S3["S_{3}"]
    FA3 --> C4["C_{4}"]
```

મેમરી ટ્રીક

પેરેલલ એડરમાં કેરી પ્રોપેગેશન માટે ``કેરી ઓલવેઝ પાસેસ રાઇટ``.

પ્રશ્ન 3(ક) [7 માર્ક્સ]

ટ્રુથ ટેબલ, સમીકરણ અને સર્કિટ ડાયાગ્રામ સાથે 8:1 મલ્ટિપ્લેક્સરનું વર્ણન કરો.

જવાબ

8:1 મલ્ટિપ્લેક્સર:

કોમ્પોનન્ટ	વર્ણન	ફંક્શન
8:1 MUX	8 ઇનપુટ્સ, 3 સિલેક્ટ લાઇન્સ, 1 આઉટપુટ વાળો ડેટા સિલેક્ટર	સિલેક્ટ લાઇન્સના આધારે 8 ઇનપુટ્સમાંથી એક પસંદ કરે છે

ટ્રુથ ટેબલ:

સિલેક્ટ લાઇન્સ	આઉટપુટ
$S_2S_1S_0$	Y
0 0 0	D <sub>0</sub>
0 0 1	D <sub>1</sub>
0 1 0	D <sub>2</sub>
0 1 1	D <sub>3</sub>
1 0 0	D <sub>4</sub>
1 0 1	D <sub>5</sub>
1 1 0	D <sub>6</sub>
1 1 1	D <sub>7</sub>

બુલિયન સમીકરણ:

$$Y = S_2' \cdot S_1' \cdot S_0' \cdot D_0 + S_2' \cdot S_1' \cdot S_0 \cdot D_1 + S_2' \cdot S_1 \cdot S_0' \cdot D_2 + S_2' \cdot S_1 \cdot S_0 \cdot D_3 + S_2 \cdot S_1' \cdot S_0' \cdot D_4 + S_2 \cdot S_1' \cdot S_0 \cdot D_5 + S_2 \cdot S_1 \cdot S_0' \cdot D_6 + S_2 \cdot S_1 \cdot S_0 \cdot D_7$$

ડાયાગ્રામ: 8:1 MUX:

```

flowchart TD
    D0["D_{0}"] -- {-}{-} MUX["8:1 MUX"]
    D1["D_{1}"] -- {-}{-} MUX
    D2["D_{2}"] -- {-}{-} MUX
    D3["D_{3}"] -- {-}{-} MUX
    D4["D_{4}"] -- {-}{-} MUX
    D5["D_{5}"] -- {-}{-} MUX
    D6["D_{6}"] -- {-}{-} MUX
    D7["D_{7}"] -- {-}{-} MUX
    S0["S_{0}"] -- {-}{-} MUX
    S1["S_{1}"] -- {-}{-} MUX
    S2["S_{2}"] -- {-}{-} MUX
    MUX -- {-}{-} Y["Y"]
                    
```

મેમરી ટ્રીક

મલ્ટિપ્લેક્સર ઓપરેશન માટે ``સિલેક્ટ ડિસાઇડ્સ ડેટા આઉટપુટ``.

પ્રશ્ન 3(અ) અથવા [3 માર્ક્સ]

હાફ સબટ્રેક્ટરની લોજિક સર્કિટ દોરો અને તેનું કાર્ય સમજાવો.

જવાબ

હાફ સબટ્રેક્ટર:

ફંક્શન	વર્ણન	ટ્રુથ ટેબલ
હાફ સબટ્રેક્ટર	બે બિટ્સને બાદ કરે છે અને ડિફરન્સ અને બોરો આપે છે	<div>A B</div> <div>0 0</div>

0 1  
1 0  
1 1

### લોજિક સર્કિટ:

flowchart TD

A["A"] --{-{-} XOR[""]}

B["B"] --{-{-} XOR}

XOR --{-{-} D["D = A"]}

A1["A{"} {-}{-} AND["•"]}

B1["B{"} {-}{-} AND}

AND --{-{-} Bout["Bout = A•B"]}

### સમીકરણો:

- ડિફરન્સ (D) =  $A \oplus B$
- બોરો આઉટ (Bout) =  $A' \cdot B$

### મેમરી ટ્રીક

હાફ સબટ્રેક્ટર ઓપરેશન માટે "ડિફરન્સ બિટ્સ બોરો".

### પ્રશ્ન 3(બ) અથવા [4 માર્ક્સ]

ટ્રુથ ટેબલ અને સર્કિટ ડાયાગ્રામ સાથે 3:8 ડીકોડર સમજાવો.

#### જવાબ

#### 3:8 ડીકોડર:

ફંક્શન	વર્ણન	ટ્રુથ ટેબલ (આંશિક)
3:8 ડીકોડર	3-બિટ બાઇનરી ઇનપુટને 8 આઉટપુટ લાઇન્સમાં કન્વર્ટ કરે છે એક સમયે માત્ર એક જ આઉટપુટ એક્ટિવ	$A_2 A_1 A_0$ 0 0 0 0 0 1 ... 1 1 1



### સર્કિટ ડાયાગ્રામ:

flowchart TD

```
A0["A_{0}"] --{-}{-} Dec["3:8"]
A1["A_{1}"] --{-}{-} Dec
A2["A_{2}"] --{-}{-} Dec
Dec --{-}{-} Y0["Y_{0}"]
Dec --{-}{-} Y1["Y_{1}"]
Dec --{-}{-} Y2["Y_{2}"]
Dec --{-}{-} Y3["Y_{3}"]
Dec --{-}{-} Y4["Y_{4}"]
Dec --{-}{-} Y5["Y_{5}"]
Dec --{-}{-} Y6["Y_{6}"]
Dec --{-}{-} Y7["Y_{7}"]
```

### સમીકરણો:

- $Y_0 = A'_2 \cdot A'_1 \cdot A'_0$
- $Y_1 = A'_2 \cdot A'_1 \cdot A_0$
- ...
- $Y_7 = A_2 \cdot A_1 \cdot A_0$

### મેમરી ટ્રીક

ડીકોડર ઓપરેશન માટે ``બાઇનરી ઇનપુટ એક્ટિવેટ્સ આઉટપુટ``.

### પ્રશ્ન 3(ક) અથવા [7 માર્ક્સ]

ટુથ ટેબલ, સમીકરણ અને સર્કિટ ડાયાગ્રામ સાથે ગ્રે થી બાઇનરી કોડ કન્વર્ટર સમજાવો.

#### જવાબ

#### ગ્રે ટુ બાઇનરી કન્વર્ટર:

ફંક્શન	વર્ણન	ટેબલ: ગ્રે ટુ બાઇનરી
ગ્રે ટુ બાઇનરી	ગ્રે કોડને બાઇનરી કોડમાં કન્વર્ટ કરે છે	ગ્રે
	બાઇનરીનો MSB ગ્રેના MSBને સમાન	0000
	દરેક બાઇનરી બિટ, હાલના ગ્રે બિટ	0001
	અને અગાઉના બાઇનરી બિટનો XOR	
	છે	0011
		0010
		0110
		...

### સર્કિટ ડાયાગ્રામ:

```
flowchart LR
    G3["G3"] --{-}{-} B3["B3"]
    G3 --{-}{-} XOR1[""]
    G2["G2"] --{-}{-} XOR1
    XOR1 --{-}{-} B2["B2"]
    XOR1 --{-}{-} XOR2[""]
    G1["G1"] --{-}{-} XOR2
    XOR2 --{-}{-} B1["B1"]
    XOR2 --{-}{-} XOR3[""]
    G0["G0"] --{-}{-} XOR3
    XOR3 --{-}{-} B0["B0"]
```

### સમીકરણો:

- $B_3 = G_3$
- $B_2 = G_3 \oplus G_2$
- $B_1 = B_2 \oplus G_1$
- $B_0 = B_1 \oplus G_0$

### મેમરી ટ્રીક

ગ્રે ટુ બાઇનરી કન્વર્ઝન માટે "MSB સ્ટેઝ, રેસ્ટ XOR".

## પ્રશ્ન 4(અ) [3 માર્ક્સ]

ટ્રુથ ટેબલ અને સર્કિટ ડાયાગ્રામ સાથે D ફ્લિપ ફ્લોપ સમજાવો.

### જવાબ

#### D ફ્લિપ-ફ્લોપ:

ફંક્શન	વર્ણન	ટ્રુથ ટેબલ
D ફ્લિપ-ફ્લોપ	ડેટા/ડિલે ફ્લિપ-ફ્લોપ ક્લોક એજ પર Q, D ને ફ્લોલો કરે છે	CLK □ □

### સર્કિટ ડાયાગ્રામ:

```
flowchart LR
    D["D"] --{-}{-} FF["D"]
    CLK[" "] --{-}{-} FF
    FF --{-}{-} Q["Q"]
    FF --{-}{-} Qnot["Q'"]
```

### કેરેક્ટરિસ્ટિક સમીકરણ:

- $Q(\text{next}) = D$

### મેમરી ટ્રીક

D ફ્લિપ-ફ્લોપ ઓપરેશન માટે "ડેટા ડિલેઝ વન ક્લોક".

## પ્રશ્ન 4(બ) [4 માર્ક્સ]

માસ્ટર સ્લેવ JK ફ્લિપ ફ્લોપનું કાર્ય સમજાવો.

## જવાબ

માસ્ટર-સ્લેવ JK ફ્લિપ-ફ્લોપ:

કોમ્પોનન્ટ	ઓપરેશન	ટ્રુથ ટેબલ
માસ્ટર	CLK = 1 હોય ત્યારે ઇનપુટ્સને સેમ્પલ કરે છે	J K
સ્લેવ	CLK = 0 હોય ત્યારે માસ્ટર આઉટપુટને ટ્રાન્સફર કરે છે	0 0 0 1 1 0 1 1

ડાયાગ્રામ: માસ્ટર-સ્લેવ JK:

flowchart LR

```

J["J"] --{-{-} Master[" JK"]}
K["K"] --{-{-} Master}
CLK[" "] --{-{-} Master}
CLK{ --{-} Slave[" JK"]}
Master --{-{-} Slave}
Slave --{-{-} Q["Q"]}
Slave --{-{-} Q["Q"]}

```

કાર્યપદ્ધતિ:

- માસ્ટર સ્ટેજ: કલોક હાઇ હોય ત્યારે ઇનપુટ કેપ્ચર કરે છે
- સ્લેવ સ્ટેજ: કલોક લો હોય ત્યારે આઉટપુટ અપડેટ કરે છે
- રેસ કન્ડિશન અટકાવે છે ઇનપુટ કેપ્ચર અને આઉટપુટ અપડેટને અલગ કરીને

## મેમરી ટ્રીક

માસ્ટર-સ્લેવ ઓપરેશન માટે “માસ્ટર સેમ્પલ્સ, સ્લેવ ટ્રાન્સફર્સ”.

## પ્રશ્ન 4(ક) [7 માર્ક્સ]

બ્લોક ડાયાગ્રામની મદદથી શિફ્ટ રજિસ્ટર્સનું વર્ગીકરણ કરો અને તેમાંના કોઈપણ એકને વિગતવાર સમજાવો.

## જવાબ

શિફ્ટ રજિસ્ટર વર્ગીકરણ:

પ્રકાર	વર્ણન	ફંક્શન
SISO	સિરિયલ ઇન સિરિયલ આઉટ	ડેટા સિરિયલી, બિટ દર બિટ, એન્ટર થાય છે અને એક્ઝિટ થાય છે
SIPO	સિરિયલ ઇન પેરેલલ આઉટ	ડેટા સિરિયલી એન્ટર થાય છે, પેરેલલમાં એક્ઝિટ થાય છે
PISO	પેરેલલ ઇન સિરિયલ આઉટ	ડેટા પેરેલલમાં એન્ટર થાય છે, સિરિયલી એક્ઝિટ થાય છે
PIPO	પેરેલલ ઇન પેરેલલ આઉટ	ડેટા પેરેલલમાં એન્ટર થાય છે અને પેરેલલમાં એક્ઝિટ થાય છે

### SIPO શિફ્ટ રજિસ્ટર વિગતવાર:

flowchart LR

```

Din[" "] --{-{-} FF1["FF_{1}"]}
FF1 --{-{-} FF2["FF_{2}"]}
FF2 --{-{-} FF3["FF_{3}"]}
FF3 --{-{-} FF4["FF_{4}"]}
CLK[" "] --{-{-} FF1}
CLK --{-{-} FF2}
CLK --{-{-} FF3}
CLK --{-{-} FF4}
FF1 --{-{-} Q0["Q_{0}"]}
FF2 --{-{-} Q1["Q_{1}"]}
FF3 --{-{-} Q2["Q_{2}"]}
FF4 --{-{-} Q3["Q_{3}"]}

```

### SIPO શિફ્ટ રજિસ્ટરનું કાર્ય:

- સિરિયલ ડેટા ડેટા-ઇન પિન પર, પ્રતિ કલોક સાયકલ એક બિટ, પ્રવેશે છે
- દરેક ફ્લિપ-ફ્લોપ કલોક પલ્સ પર તેની સામગ્રીને આગળના ફ્લિપ-ફ્લોપમાં પાસ કરે છે
- 4 કલોક સાયકલ્સ પછી, 4-બિટ ડેટા બધા ફ્લિપ-ફ્લોપ્સમાં સ્ટોર થાય છે
- પેરેલલ આઉટપુટ Q0-Q3 પરથી એક સાથે ઉપલબ્ધ થાય છે

### SIPO માટે ટાઇમિંગ ડાયાગ્રામ:

```

Clock  \_ | \_ | \_ | \_ | \_
Data    \_ \_ \_ | \_ \_ \_ | \_
Q0       \_ \_ \_ | \_ \_ \_ | \_
Q1       \_ \_ \_ \_ | \_ \_ \_ |
Q2       \_ \_ \_ \_ \_ | \_ \_ \_
Q3       \_ \_ \_ \_ \_ \_ | \_ \_

```

### મેમરી ટ્રીક

SIPO ઓપરેશન માટે "સિરિયલ ઇનપુટ્સ પેરેલલ આઉટપુટ્સ".

### પ્રશ્ન 4(અ) અથવા [3 માર્ક્સ]

ટ્રુથ ટેબલ અને સર્કિટ ડાયાગ્રામ સાથે SR ફ્લિપ ફ્લોપ સમજાવો.

### જવાબ

#### SR ફ્લિપ-ફ્લોપ:

ફંક્શન	વર્ણન	ટ્રુથ ટેબલ
SR ફ્લિપ-ફ્લોપ	સેટ-રિસેટ ફ્લિપ-ફ્લોપ	S R
	બેઝિક મેમોરી એલિમેન્ટ	0 0
		0 1
		1 0
		1 1

#### સર્કિટ ડાયાગ્રામ:

flowchart LR

```

S["S"] --{-{-} NOR1["1"]}
QN["Q{"] --{-{-} NOR1}
NOR1 --{-{-} Q["Q"]}
R["R"] --{-{-} NOR2["1"]}
Q --{-{-} NOR2}
NOR2 --{-{-} QN}

```

### મેમરી ટ્રીક

SR ફ્લિપ-ફ્લોપ ઓપરેશન માટે "સેટ ટુ 1, રિસેટ ટુ 0".

### પ્રશ્ન 4(બ) અથવા [4 માર્ક્સ]

ટ્રુથ ટેબલ અને સર્કિટ ડાયાગ્રામ સાથે JK ફ્લિપ ફ્લોપ સમજાવો.

#### જવાબ

JK ફ્લિપ-ફ્લોપ:

ફંક્શન	વર્ણન	ટ્રુથ ટેબલ
JK ફ્લિપ-ફ્લોપ	ઇમ્યુલેટ SR ફ્લિપ-ફ્લોપ અમાન્ય કન્ડિશન હલ કરે છે	J K 0 0 0 1 1 0 1 1

સર્કિટ ડાયાગ્રામ:

```

flowchart LR
    J["J"] -- "-{-}" --> AND1["•"]
    Qn["Q'"] -- "-{-}" --> AND1
    AND1 -- "AND1" --> K["K"]
    K -- "-{-}" --> AND2["•"]
    Q["Q"] -- "-{-}" --> AND2
    AND2 -- "AND2" --> OR["OR"]
    OR -- "-{-}" --> FF["D FF"]
    CLK["CLK"] -- "-{-}" --> FF
    FF -- "-{-}" --> Q
    FF -- "-{-}" --> Qn
    
```

કેરેક્ટરિસ્ટિક સમીકરણ:

$$Q(\text{next}) = J \cdot Q' + K' \cdot Q$$

### મેમરી ટ્રીક

JK ફ્લિપ-ફ્લોપ સ્ટેટ્સ માટે "જમ્પ-કીપ-ટોગલ" (J=1  
K=0: 1  
પર જમ્પ,  
J=0  
K=0: સ્ટેટ જાળવવો,  
J=1  
K=1: ટોગલ).

### પ્રશ્ન 4(ક) અથવા [7 માર્ક્સ]

ટ્રુથ ટેબલ અને સર્કિટ ડાયાગ્રામ સાથે 4-બિટ અસિંક્રોનસ અપ કાઉન્ટરનું વર્ણન કરો.

#### જવાબ

4-બિટ અસિંક્રોનસ અપ કાઉન્ટર:

ફંક્શન	વર્ણન	કાઉન્ટ સિક્વન્સ
અસિંક્રોનસ કાઉન્ટર	રિપલ કાઉન્ટર પણ કહેવાય છે ક્લોક માત્ર પહેલા FF ને ડ્રાઇવ કરે છે	0000 → 0001 → 0010 → 0011 0100 → 0101 → 0110 → 0111

દરેક FF અગાઉના FF આઉટપુટ દ્વારા ટ્રિગર થાય છે

1000 → 1001 → 1010 → 1011

1100 → 1101 → 1110 → 1111

#### સર્કિટ ડાયાગ્રામ:

flowchart LR

```

CLK[" "] --> JK1["JK FF_{0}"]
JK1 --> J1["J=1"]
J1 --> K1["K=1"]
K1 --> JK1
JK1 --> Q0["Q_{0}"]
Q0 --> JK2["JK FF_{1}"]
JK2 --> J2["J=1"]
J2 --> K2["K=1"]
K2 --> JK2
JK2 --> Q1["Q_{1}"]
Q1 --> JK3["JK FF_{2}"]
JK3 --> J3["J=1"]
J3 --> K3["K=1"]
K3 --> JK3
JK3 --> Q2["Q_{2}"]
Q2 --> JK4["JK FF_{3}"]
JK4 --> J4["J=1"]
J4 --> K4["K=1"]
K4 --> JK4
JK4 --> Q3["Q_{3}"]

```

#### કાર્યપદ્ધતિ:

- પહેલો FF દરેક કલોક પલ્સ પર ટોગલ થાય છે
- બીજો FF જ્યારે પહેલો FF 1 થી 0 પર જાય છે ત્યારે ટોગલ થાય છે
- ત્રીજો FF જ્યારે બીજો FF 1 થી 0 પર જાય છે ત્યારે ટોગલ થાય છે
- ચોથો FF જ્યારે ત્રીજો FF 1 થી 0 પર જાય છે ત્યારે ટોગલ થાય છે

#### મેમરી ટ્રીક

અસિંક્રોનસ કાઉન્ટર ઓપરેશન માટે "રિપલ કેરીઝ પ્રોપેગેશન ડિલે".

### પ્રશ્ન 5(અ) [3 માર્ક્સ]

નીચેની લોજિક ફેમિલીઝની તુલના કરો: TTL, CMOS, ECL

#### જવાબ

લોજિક ફેમિલીઝ કમ્પેરિઝન:

પેરામીટર	TTL	CMOS	ECL
ટેકનોલોજી	બાયપોલર ટ્રાન્ઝિસ્ટર્સ	MOSFETs	બાયપોલર ટ્રાન્ઝિસ્ટર્સ
પાવર કન્ઝમ્પશન	મધ્યમ	ખૂબ ઓછો	ઉચ્ચ
સ્પીડ	મધ્યમ	નીચી-મધ્યમ	ખૂબ ઉચ્ચ
નોઇઝ ઇમ્યુનિટી	મધ્યમ	ઉચ્ચ	નીચી
ફેન-આઉટ	10	50+	25
સપ્લાય વોલ્ટેજ	5V	3-15V	-5.2V

#### મેમરી ટ્રીક

લોજિક ફેમિલીઝની તુલના માટે "ટેકનોલોજી કન્ટ્રોલ્સ મેની ઇલેક્ટ્રિકલ કેરેક્ટરિસ્ટિક્સ".

### પ્રશ્ન 5(બ) [4 માર્ક્સ]

કોમ્પ્લેમેન્ટરી અને સિક્વેન્શિયલ લોજિક સર્કિટ્સની સરખામણી કરો.



પ્રશ્ન 5(અ) અથવા [3 માર્ક્સ]

ડિજિટલ ICના ઇ-વેસ્ટ મેનેજમેન્ટના પગલાં અને જરૂરિયાતનું વર્ણન કરો.

જવાબ		
ડિજિટલ ICs માટે ઇ-વેસ્ટ મેનેજમેન્ટ:		
સ્ટેપ	વર્ણન	મહત્વ
કલેક્શન	ઇલેક્ટ્રોનિક વેસ્ટનું અલગ કલેક્શન	અયોગ્ય ડિસ્પોઝલને રોકે છે
સેગ્રેગેશન	ICsને અન્ય કોમ્પોનન્ટ્સથી અલગ કરવું	ટાર્ગેટેડ રિસાયક્લિંગ શક્ય બનાવે છે
ડિસમેન્ટલિંગ	હાનિકારક ભાગોને દૂર કરવા	પર્યાવરણીય નુકસાન ઘટાડે છે
રિકવરી	મૂલ્યવાન મટીરિયલ્સ (ગોલ્ડ, સિલિકોન) એક્સટ્રેક્ટ કરવા	સંસાધનો બચાવે છે
સેફ ડિસ્પોઝલ	નોન-રિસાયકલેબલ પાર્ટ્સનો યોગ્ય નિકાલ	પ્રદૂષણ અટકાવે છે
ઇ-વેસ્ટ મેનેજમેન્ટની જરૂરિયાત:		
<ul style="list-style-type: none"><li>હાનિકારક મટીરિયલ્સ: ICs લેડ, મર્ક્યુરી, કેડમિયમ ધરાવે છે</li><li>રિસોર્સ કન્ઝર્વેશન: કિંમતી ધાતુઓ અને દુર્લભ સામગ્રી પુનઃપ્રાપ્ત કરે છે</li><li>પર્યાવરણ સંરક્ષણ: જમીન અને પાણીના પ્રદૂષણને રોકે છે</li><li>હેલ્થ સેફ્ટી: ઝેરી પદાર્થોના સંપર્કને ઘટાડે છે</li></ul>		

મેમરી ટ્રીક
ઇ-વેસ્ટ મેનેજમેન્ટ સ્ટેપ્સ માટે "કલેક્શન સ્ટાર્ટ્સ ડિસમેન્ટલિંગ રિકવરી સેફલી".

પ્રશ્ન 5(બ) અથવા [4 માર્ક્સ]

સર્કિટ ડાયાગ્રામ સાથે રીંગ કાઉન્ટરનું કામ સમજાવો.

જવાબ		
રીંગ કાઉન્ટર:		
ફંક્શન	વર્ણન	કાઉન્ટ સિક્વન્સ
રીંગ કાઉન્ટર	સિંગલ 1 સાથે સર્ક્યુલર શિફ્ટ રજિસ્ટર કોઈપણ સમયે માત્ર એક જ ફ્લિપ-ફ્લોપ સેટ થયેલ હોય છે N સ્ટેટ્સ માટે N ફ્લિપ-ફ્લોપ્સ	1000 → 0100 → 0010 → 0001 → 1000