

Subject Name (Gujarati)

4321102 -- Summer 2023

Semester 1 Study Material

Detailed Solutions and Explanations

પ્રશ્ન 1(અ) [3 ગુણ]

બુલિયન એલજીબ્રા માટેના ડે-મોર્ગનના નિયમ સમજાવો

જવાબ

ડે-મોર્ગનના નિયમમાં બે કાયદા છે જે AND, OR અને NOT ક્રિયાઓ વચ્ચેના સંબંધને દર્શાવે છે:

કાયદો 1: સરવાળાના પૂરકની કિમત પૂરકના ગુણાકાર બરાબર હોય છે $\overline{A + B} = \overline{A} \cdot \overline{B}$

કાયદો 2: ગુણાકારના પૂરકની કિમત પૂરકના સરવાળા બરાબર હોય છે $A \cdot \overline{B} = \overline{A} + B$

Table 1: ડે-મોર્ગનના નિયમની ચકાસણી

A	B	A+B	$\overline{A + B}$	\overline{A}	\overline{B}	$\overline{A} \cdot \overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

સ્મૃતિસહાય: "OR પર NOT થાય AND, AND પર NOT થાય OR"

પ્રશ્ન 1(બ) [4 ગુણ]

નીચેના ડિસિમલ નંબરને બાયનરી અને ઓક્ટલ નંબરમાં ફેરવો (i) 215 (ii) 59

જવાબ

બાયનરી રૂપાંતર:

215 માટે:

- 2 વડે ભાગ કરો: $215/2 = 107$ શેષ 1
- $107/2 = 53$ શેષ 1
- $53/2 = 26$ શેષ 1
- $26/2 = 13$ શેષ 0
- $13/2 = 6$ શેષ 1
- $6/2 = 3$ શેષ 0
- $3/2 = 1$ શેષ 1
- $1/2 = 0$ શેષ 1
- તેથી, $(215)_{10} = (11010111)_2$

59 માટે:

- 2 વડે ભાગ કરો: $59/2 = 29$ શેષ 1
- $29/2 = 14$ શેષ 1
- $14/2 = 7$ શેષ 0
- $7/2 = 3$ શેષ 1
- $3/2 = 1$ શેષ 1
- $1/2 = 0$ શેષ 1
- તેથી, $(59)_{10} = (111011)_2$

ઓક્ટલ રૂપાંતર:

215 માટે:

- 8 વડે ભાગ કરો: $215/8 = 26$ શેષ 7
- $26/8 = 3$ શેષ 2
- $3/8 = 0$ શેષ 3
- તેથી, $(215)_{10} = (327)_8$

59 માટે:

- 8 વડે ભાગ કરો: $59/8 = 7$ શેષ 3

- $7/8 = 0$ શેષ 7
- તેથી, $(59)_{10} = (73)_8$

Table 2: સંખ્યા રૂપાંતર સારાંશ

ડેસિમલ	બાયનરી	ઓક્ટલ
215	11010111	327
59	111011	73

સ્મૃતિસહાય: "આધાર વડે ભાગો, શેષ નીચેથી ઉપર વાંચો"

પ્રશ્ન 1(ક)(I) [2 ગુણ]

ડેસિમલ, બાયનરી, ઓક્ટલ અને હેક્ઝાડેસિમલ નંબર સિસ્ટમનો બેઝ લખો

જવાબ

Table 3: સંખ્યા પદ્ધતિના આધાર

સંખ્યા પદ્ધતિ	આધાર
ડેસિમલ	10
બાયનરી	2
ઓક્ટલ	8
હેક્ઝાડેસિમલ	16

સ્મૃતિસહાય: "ડ-બા-ઓ-હે: 10-2-8-16"

પ્રશ્ન 1(ક)(II) [2 ગુણ]

$$(147)_{10} = (\underline{\hspace{2cm}})_2 = (\underline{\hspace{2cm}})_{16}$$

જવાબ

ડેસિમલથી બાયનરી રૂપાંતર:

- $147/2 = 73$ શેષ 1
- $73/2 = 36$ શેષ 1
- $36/2 = 18$ શેષ 0
- $18/2 = 9$ શેષ 0
- $9/2 = 4$ શેષ 1
- $4/2 = 2$ શેષ 0
- $2/2 = 1$ શેષ 0
- $1/2 = 0$ શેષ 1
- તેથી, $(147)_{10} = (10010011)_2$

ડેસિમલથી હેક્ઝાડેસિમલ રૂપાંતર:

- બાયનરી અંકોને 4ના સમૂહમાં વિભાજિત કરો: 1001 0011
- દરેક સમૂહને હેક્સમાં રૂપાંતરિત કરો: $1001 = 9, 0011 = 3$
- તેથી, $(147)_{10} = (93)_{16}$

Table 4: રૂપાંતર પરિણામ

ડેસિમલ	બાયનરી	હેક્ઝાડેસિમલ
147	10010011	93

સ્મૃતિસહાય: "હેક્સ માટે જમણોથી 4ના સમૂહમાં વિભાજિત કરો"

પ્રશ્ન 1(ક)(III) [3 ગુણ]

નીચેના બાયનરી કોડનું ગ્રે કોડમાં રૂપાંતર કરો (i) 1011 (ii) 1110

જવાબ

બાયનરીથી ગ્રે કોડ રૂપાંતર પ્રક્રિયા:

- ગ્રે કોડનો MSB (દાબી બાજુનો બિટ) બાયનરી કોડના MSB જેવો જ હોય છે
- ગ્રે કોડના અન્ય બિટ્સ બાયનરી કોડના આસપાસના બિટ્સને XOR કરીને મેળવવામાં આવે છે

1011 માટે:

- ગ્રે કોડનો MSB = બાયનરી કોડનો MSB = 1
- બીજો બિટ = 1 XOR 0 = 1
- ત્રીજો બિટ = 0 XOR 1 = 1
- ચોથો બિટ = 1 XOR 1 = 0
- તેથી, $(1011)_2 = (1110)_a$

1110 માટે:

- ગ્રે કોડનો MSB = બાયનરી કોડનો MSB = 1
- બીજો બિટ = 1 XOR 1 = 0
- ત્રીજો બિટ = 1 XOR 1 = 0
- ચોથો બિટ = 1 XOR 0 = 1
- તેથી, $(1110)_2 = (1001)_a$

Table 5: બાયનરીથી ગ્રે કોડ રૂપાંતર

બાયનરી	રૂપાંતર પદ્ધતિ	ગ્રે કોડ
1011	$1, 1 \oplus 0 = 1, 0 \oplus 1 = 1, 1 \oplus 1 = 0$	1110
1110	$1, 1 \oplus 1 = 0, 1 \oplus 1 = 0, 1 \oplus 0 = 1$	1001

સ્મૃતિસહાય: "પહેલો રાખો, બાકીના XOR કરો"

પ્રશ્ન 1(ક) [વૈકલ્પિક પ્રશ્ન] (I) [2 ગુણ]

BCD અને ASCII નું ફૂલફૂર્મ લખો

જવાબ

Table 6: BCD અને ASCII નું પૂર્ણ નામ

સંક્ષિપ્ત રૂપ	પૂર્ણ નામ
BCD	Binary Coded Decimal
ASCII	American Standard Code for Information Interchange

સ્મૃતિસહાય: "બાયનરી કોડેડ ડેસિમલ, અમેરિકન સ્ટાન્ડર્ડ કોડ ફોર ઇન્ફોર્મેશન ઇન્ટરચેન્જ"

પ્રશ્ન 1(ક) [વૈકલ્પિક પ્રશ્ન] (II) [2 ગુણ]

નીચેના બાયનરી નંબરના 1'સ અને 2'સ કોમ્પિલમેન્ટ શોધો (i) 1010 (ii) 1011

જવાબ

1's કોમ્પિલમેન્ટ: બધા બિટ્સ ઉલટાવો (0 ને 1 અને 1 ને 0 માં બદલો) 2's કોમ્પિલમેન્ટ: 1's કોમ્પિલમેન્ટ લો અને 1 ઉમેરો

1010 માટે:

- 1's કોમ્પિલમેન્ટ: 0101
- 2's કોમ્પિલમેન્ટ: $0101 + 1 = 0110$

1011 માટે:

- 1's કોમ્પિલમેન્ટ: 0100
- 2's કોમ્પિલમેન્ટ: $0100 + 1 = 0101$

Table 7: કોમ્પિલમેન્ટ પરિણામો

બાયનરી	1's કોમ્પિલમેન્ટ	2's કોમ્પિલમેન્ટ
1010	0101	0110
1011	0100	0101

સ્મૃતિસહાય: "1's માટે બધા બિટ ઉલટાવો, 2's માટે એક ઉમેરો"

પ્રશ્ન 1(ક) [વૈકલ્પિક પ્રશ્ન] (III) [3 ગુણ]

2's કોમ્પલેન્ટ મેથડથી બાદબાકી કરો (i) $(110110)_2 - -(101010)_2$

જવાબ

2's કોમ્પલેન્ટ પદ્ધતિથી બાદબાકી માટે:

1. બાદ થનાર સંખ્યાનો 2's કોમ્પલેન્ટ શોધો
2. તેને મૂળ સંખ્યામાં ઉમેરો
3. બિટ વિડુથની બહારના કેરીને છોડી દો

બાદબાકી: $(110110)_2 - -(101010)_2$

પગલું 1: 101010 નો 2's કોમ્પલેન્ટ શોધો

- 101010 નો 1's કોમ્પલેન્ટ = 010101
- $2's$ કોમ્પલેન્ટ = $010101 + 1 = 010110$

પગલું 2: $110110 + 010110$ ઉમેરો

$$\begin{array}{r} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ + & 0 & 1 & 0 & 1 & 1 & 0 \\ \hline & 0 & 0 & 1 & 1 & 0 & 0 \end{array}$$

પગલું 3: પરિણામ $001100 = (12)_{10}$

Table 8: બાદબાકી પ્રક્રિયા

પગલું	ક્રિયા	પરિણામ
1	101010 નો 2's કોમ્પલેન્ટ	010110
2	$110110 + 010110$ ઉમેરો	001100
3	અંતિમ પરિણામ (ડિસિમલ)	12

સ્મૃતિસહાય: "બાદનારનો કોમ્પલેન્ટ લો, ઉમેરો, કેરી ભૂલી જાઓ"

પ્રશ્ન 2(અ) [3 ગુણ]

NAND ગેટનો જ ઉપયોગ કરી AND, OR અને NOT ગેટની લોજિક સર્કિટ બનાવો

જવાબ

AND ગેટ NAND ગેટથી:

- AND ગેટ = NAND ગેટ પછી NOT ગેટ (NAND ગેટ)

OR ગેટ NAND ગેટથી:

- OR ગેટ = બંને ઇનપુટને NOT (NAND ગેટ) લાગુ કરો, પછી તે પરિણામોને NAND કરો

NOT ગેટ NAND ગેટથી:

- NOT ગેટ = NAND ગેટ જેમાં બંને ઇનપુટ જોડાયેલા હોય

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph TD
    subgraph "NOT"
        A1[A] --{-{-}}--> NAND1[NAND]
        A1 --{-{-}}--> nota[NOT A]
    end

    subgraph "AND"
        B1[B] --{-{-}}--> NAND2[NAND]
        C1[C] --{-{-}}--> NAND2
        NAND2 --{-{-}}--> NAND3[NAND]
        NAND2 --{-{-}}--> andResult[B AND C]
        NAND3 --{-{-}}--> andResult
    end
```

```

end

subgraph "OR"
D1[D] {-{-}{}} NAND4([NAND])
D1 {-{-}{}} NAND4
E1[E] {-{-}{}} NAND5([NAND])
E1 {-{-}{}} NAND5
NAND4 {-{-}{}} NAND6([NAND])
NAND5 {-{-}{}} NAND6
NAND6 {-{-}{}} orResult[D OR E]
end
{Highlighting}
{Shaded}

```

સ્મૃતિસહાય: "NOT માટે એક NAND, AND માટે બે NAND, OR માટે ત્રણ NAND"

પ્રશ્ન 2(બ) [4 ગુણ]

નીચેના લોજિક ગેટનો લોજિક સિમ્બોલ, ટ્રથ ટેબલ અને સમીકરણ લખો/દોરો (i) XOR ગેટ (ii) OR ગેટ

જવાબ

XOR ગેટ:
લોજિક સિમ્બોલ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
A[A] {-{-}{}} XOR([] )
B[B] {-{-}{}} XOR
XOR {-{-}{}} Y[Y]
{Highlighting}
{Shaded}

```

ટ્રથ ટેબલ:

A	B	Y(A)
0	0	0
0	1	1
1	0	1
1	1	0

બુલિયન સમીકરણ: $Y = A = A'B + AB'$

OR ગેટ:

લોજિક સિમ્બોલ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    A[A] --{-{-}{}} OR --{[1]} --> Y[Y]
    B[B] --{-{-}{}} OR --{[1]} --> Y
    OR --{-{-}{}} Y[Y]
{Highlighting}
{Shaded}
```

ટુથ ટેબલ:

A	B	$Y(A+B)$
0	0	0
0	1	1
1	0	1
1	1	1

બુલિયન સમીકરણ: $Y = A+B$

સ્મૃતિસહાય: "XOR: એક્સક્લુસિવ OR - એક અથવા બીજું પણ બંને નહીં; OR: એક અથવા બીજું અથવા બંને"

પ્રશ્ન 2(ક)(િ) [૩ ગુણ]

બુલિયન સમીકરણ $Y = A + B[AC + (B + C)\bar{D}]$ ને algebraic મેથડથી સરળ બનાવો

જવાબ

પગલાંવાર સરળીકરણ:

$$Y = A + B[AC + (B + C)\bar{D}]$$

$$Y = A + B[AC + BD + C\bar{D}]$$

$$Y = A + BAC +$$

$$BB\bar{D} + BC\bar{D}$$

$$Y = A + BAC + BD + BC\bar{D}$$
 (કારણ કે $BB = B$)

Absorption law ($X + XY = X$) લાગુ કરો: $Y = A + AC + BD + BC\bar{D}$ (કારણ કે $A + BAC = A + AC$)

$$Y = A + BD + BC\bar{D}$$
 (કારણ કે $A + AC = A$)

$$Y = A +$$

$$B(D + C\bar{D})$$

$$Y = A + BD + BC\bar{D}$$

$$Y = A + BD(1 + C\bar{D})$$

$$Y = A + BD$$
 (કારણ કે $1 + C\bar{D} =$

1)

અંતિમ સમીકરણ: $Y = A + BD$

Table 9: સરળીકરણ પગલાં

પગલું	સમીકરણ	લાગુ પડેલ નિયમ
1	$A + B[AC + (B + C)\bar{D}]$	મૂળ
2	$A + B[AC + BD + C\bar{D}]$	વિતરણ
3	$A + BAC + BBD + BC\bar{D}$	વિતરણ
4	$A + BAC + BD + BC\bar{D}$	આઇડેમ્પોટન્ (BB = B)
5	$A + AC + BD + BC\bar{D}$	અવશોધણ
6	$A + BD + BC\bar{D}$	અવશોધણ (A+AC=A)
7	$A + B(D + C\bar{D})$	ફેક્ટરિંગ
8	$A + BD$	પૂરક નિયમ

સ્મૃતિસહાય: "આઇડેપ્લોટન્સ, અવશોષણ, અને પૂરક પેર્ટ માટે હંમેશા તપાસો"

પ્રશ્ન 2(ક) (II) [4 ગુણ]

બુલિયન સમીકરણ $F(A,B,C) = \sum m(0, 2, 3, 4, 5, 6)$ ને Karnaugh Map ની મદદથી સરળ બનાવો

જવાબ

$F(A,B,C) = \sum m(0, 2, 3, 4, 5, 6)$ માટે K-map બનાવો:

K-map:

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

1 ની ગ્રૂપિંગ કરો:

- ગ્રૂપ 1: $m(0,4)$ - $A'B'C'$ સાથે સંબંધિત
- ગ્રૂપ 2: $m(2,6)$ - $B'C$ સાથે સંબંધિત
- ગ્રૂપ 3: $m(4,5)$ - AB' સાથે સંબંધિત

સરળ સમીકરણ: $F(A,B,C) = B'C + A'B'C' + AB'$

વધુ સરળ કરીએ: $F(A,B,C) = B'C + B'C'(A' + A)$ $F(A,B,C) = B'C + B'C'$ $F(A,B,C) = B'(C + C')$ $F(A,B,C) = B'$

અંતિમ સમીકરણ: $F(A,B,C) = B'$

આકૃતિ: K-map ગ્રૂપિંગ

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

↑-----↑ ↑-----↑

Group 1 Group 2

↑-----↑

Group 3

સ્મૃતિસહાય: "2ની પાવરમાં આસપાસના 1 ને ગ્રૂપ કરો"

પ્રશ્ન 2 [વૈકલ્પિક પ્રશ્ન] (અ) [3 ગુણ]

NOR ગેટનો જ ઉપયોગ કરી AND, OR અને NOT ગેટની લોજિક સર્કિટ બનાવો

જવાબ

NOT ગેટ NOR ગેટથી:

- NOT ગેટ = NOR ગેટ જેમાં બંને ઇનપુટ જોડાયેલા હોય

AND ગેટ NOR ગેટથી:

- AND ગેટ = બંને ઇનપુટને NOT (NOR ગેટ) લાગુ કરો, પછી તે પરિણામોને ફરીથી NOR કરો

OR ગેટ NOR ગેટથી:

- OR ગેટ = NOR ગેટ પછી NOT ગેટ (NOR ગેટ)

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph TD
    subgraph "NOT"
        A1[A] --{-{-}{}}-- NOR1([NOR])
        A1 --{-{-}{}}-- NOR1
        NOR1 --{-{-}{}}-- notA[NOT A]
    end
```

```

subgraph "AND"
    B1[B] {-{-}{}} NOR2([NOR])
    B1 {-{-}{}} NOR2
    C1[C] {-{-}{}} NOR3([NOR])
    C1 {-{-}{}} NOR3
    NOR2 {-{-}{}} NOR4([NOR])
    NOR3 {-{-}{}} NOR4
    NOR4 {-{-}{}} andResult[B AND C]
end

subgraph "OR"
    D1[D] {-{-}{}} NOR5([NOR])
    E1[E] {-{-}{}} NOR5
    NOR5 {-{-}{}} NOR6([NOR])
    NOR5 {-{-}{}} NOR6
    NOR6 {-{-}{}} orResult[D OR E]
end

{Highlighting}
{Shaded}

```

સ્મૃતિસાહય: “NOT માટે એક NOR, દરેકને NOT કરીને NOR કરો AND માટે, બે વાર NOR કરો OR માટે”

પ્રશ્ન 2 [વૈકલ્પિક પ્રશ્ન] (બ) [4 ગુણ]

નીચેના લોજિક ગેટનો લોજિક સિમ્બોલ, ટુથ ટેબલ અને સમીકરણ લખો/દરો (i) NOR ગેટ (ii) AND ગેટ

જવાબ

NOR ગેટ:
લોજિક સિમ્બોલ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
    A[A] {-{-}{}} NOR([1 with bubble])
    B[B] {-{-}{}} NOR
    NOR {-{-}{}} Y[Y]
{Highlighting}
{Shaded}

```

ટુથ ટેબલ:

A	B	$Y(A+B)'$
0	0	1
0	1	0
1	0	0
1	1	0

બુલિયન સમીકરણ: $Y = (A+B)' = A'B'$

AND ગેટ:

લોજિક સિમ્બોલ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    A[A] --{-{-}{}}--> AND1[AND([\\&])]
    B[B] --{-{-}{}}--> AND1
    AND1 --{-{-}{}}--> Y[Y]
{Highlighting}
{Shaded}
```

ટુથ ટેબલ:

A	B	$Y (A \cdot B)$
0	0	0
0	1	0
1	0	0
1	1	1

બુલિયન સમીકરણ: $Y = A \cdot B$

સ્મૃતિસહાય: "NOR: NOT OR - ન તો એક કે ન તો બીજું; AND: બંને 1 હોવા જ જોઈએ"

પ્રશ્ન 2 [વૈકલ્પિક પ્રશ્ન] (ક) [૭ ગુણ]

ઉપરની લોજિક સર્કિટ માટે બુલિયન સમીકરણ લખો. આ સમીકરણને સરળ બનાવો અને આ સરળ સમીકરણની લોજિક સર્કિટ AND-OR-Invert મેથડથી દોરો

જવાબ

પગલું 1: સર્કિટમાંથી બુલિયન સમીકરણ લખો: $Q = (A + B) \cdot (B + C \cdot ((B + C)'))$

$$Q = (A + B) \cdot (B + C \cdot (B' \cdot C'))$$

$$Q = (A + B) \cdot (B + C \cdot B' \cdot C')$$

પગલું 2: સમીકરણને સરળ બનાવો:

- નોંધ કરો કે $C \cdot C' = 0$
- તેથી, $C \cdot B' \cdot C' = 0$
- એટલે

$$Q = (A + B) \cdot (B + 0) = (A + B) \cdot$$

$$B = A \cdot B + B \cdot B = A \cdot B +$$

$$B = B +$$

$$A \cdot B = B(1 + A) = B$$

પગલું 3: અંતિમ સરળ સમીકરણ: $Q = B$

પગલું 4: AND-OR-Invert દ્વારા $Q = B$ નું અમલીકરણ:

- આ ફક્ત ઇનપુટ B થી આઉટપુટ Q સુધીનો એક તાર છે

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    B[B] --{-{-}{}}--> Q[Q]
{Highlighting}
{Shaded}
```

Table 10: સરળીકરણ પગલાં

પગલું	સમીકરણ	સરળીકરણ
1	$(A + B) \cdot (B + C \cdot ((B + C)'))$	મૂળ સમીકરણ

2	$(A + B) \cdot (B + C \cdot B' \cdot C')$	ડી મોર્ગનનો નિયમ લાગુ કરવો
3	$(A + B) \cdot (B + 0)$	$C \cdot C' = 0$
4	$(A + B) \cdot B$	સરળીકરણ
5	$A \cdot B + B \cdot B$	વિતરણ ગુણધર્મ
6	$A \cdot B + B$	આઇડેમ્પોટન્ટ ગુણધર્મ ($B \cdot B = B$)
7	$B(1 + A)$	ફેક્ટરિંગ
8	B	$1 + A = 1$

સ્મૃતિસહાય: "જ્યારે પૂરક ચલ ગુણાકાર કરે, તેઓ શૂન્ય થાય"

પ્રશ્ન 3(અ) [3 ગુણ]

કોમ્બીનેશનલ સર્કિટની વ્યાખ્યા લખો. કોમ્બીનેશનલ સર્કિટના બે ઉદાહરણ લખો

જવાબ

કોમ્બીનેશનલ સર્કિટ: એક ડિજિટલ સર્કિટ જેનું આઉટપુટ માત્ર વર્તમાન ઇનપુટ મૂલ્યો પર આધારિત હોય છે અને અગાઉના ઇનપુટ અથવા સ્થિતિઓ પર નહીં. કોમ્બીનેશનલ સર્કિટમાં કોઈ મેમરી અથવા ફીડબેક હોતા નથી.

મુખ્ય લક્ષણો:

- આઉટપુટ ફક્ત વર્તમાન ઇનપુટ પર આધારિત હોય છે
- કોઈ મેમરી એલિમેન્ટ નથી
- કોઈ ફીડબેક પાથ નથી

કોમ્બીનેશનલ સર્કિટના ઉદાહરણો:

- માલ્ટિપ્લિક્સર (MUX)
- ડિકોડર
- એડર/સબટ્રેક્ટર
- એનકોડર
- કમ્પ્યુટર

Table 11: કોમ્બીનેશનલ vs સિક્વેન્શિયલ સર્કિટ

લક્ષણ	કોમ્બીનેશનલ સર્કિટ	સિક્વેન્શિયલ સર્કિટ
મેમરી	ના	હા
ફીડબેક	ના	સામાન્ય રીતે
આઉટપુટ આધારિત	માત્ર વર્તમાન ઇનપુટ	વર્તમાન અને અગાઉના ઇનપુટ
ઉદાહરણો	માલ્ટિપ્લિક્સર, એડર	ફિલેપ-ફ્લોપ, કાઉન્ટર

સ્મૃતિસહાય: "કોમ્બીનેશનલ સર્કિટ: વર્તમાન આવે, વર્તમાન જાય - કોઈ યાદ નહીં!"

પ્રશ્ન 3(બ) [4 ગુણ]

લોજિક સર્કિટ અને ટ્રૂથ ટેબલની મદદથી હાફ એડર સમજાવો

જવાબ

હાફ એડર: એક કોમ્બીનેશનલ સર્કિટ જે બે બાયનરી અંકો ઉમેરે છે અને સમ અને કેરી આઉટપુટ ઉત્પત્ત કરે છે.

લોજિક સર્કિટ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    A[A] {-{-}{}} XOR{[]}
    B[B] {-{-}{}} XOR
    XOR {-{-}{}} S[Sum]
    A {-{-}{}} AND{[\&]}
    B {-{-}{}} AND
    AND {-{-}{}} C[Carry]
{Highlighting}
{Shaded}
```

ટુથ ટેબલ:

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

બુલિયન સમીકરણાઃ

- $\text{Sum} = A \oplus B$

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

- $\text{Carry} = A \cdot B$

મર્યાદાઓ:

- ત્રણ બાયનરી અંકો ઉમેરી શકતા નથી

- અગાઉના તબક્કામાંથી કેરી ઇનપુટ સમાવી શકતા નથી

સ્મૃતિસહાય: "XOR સમને માટે, AND કેરીને માટે"

પ્રશ્ન 3(ક)(I) [3 ગુણ]

મલિટિપ્લેક્સર ટૂકમાં સમજાવો

જવાબ

મલિટિપ્લેક્સર (MUX): એક કોમ્બીનેશનલ સર્કિટ જે સિલેક્ટ લાઇનના આધારે અનેક ઇનપુટ સિગનલ્સમાંથી એકને પસંદ કરે છે અને તેને એક આઉટપુટ લાઇન પર મોકલે છે.

મુખ્ય લક્ષણો:

- ડિજિટલ સ્વિચ તરીકે કાર્ય કરે છે
- $2^n, n, 1$
- સિલેક્ટ લાઇન નક્કી કરે છે કે ક્યું ઇનપુટ આઉટપુટથી જોડાયેલું છે

સામાન્ય મલિટિપ્લેક્સર:

- 2:1 MUX (1 સિલેક્ટ લાઇન)
- 4:1 MUX (2 સિલેક્ટ લાઇન)
- 8:1 MUX (3 સિલેક્ટ લાઇન)

મૂળભૂત રચનાઃ

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph TD
    I0[IO] --> MUX([MUX])
    I1[I1] --> MUX
    In[...] --> MUX
    I2n{-1}[I2\~n{-1}] --> MUX
    S[Select Lines] --> MUX
    MUX --> Y[Output Y]
{Highlighting}
{Shaded}
```

ઉપયોગો:

- ડેટા રાઉટિંગ
- ડેટા પસંદગી
- પેરેલાથી સીરિયલ રૂપાંતર
- બુલિયન ફંક્શનનું અમલીકરણ

સ્મૃતિસહાય: "ઘણા ઇન, સિલેક્શન પસંદ કરે, એક આઉટ"

પ્રશ્ન 3(ક)(II) [4 ગુણ]

8:1 મલિટિપ્લેક્સર ડિજાઇન કરો. તેનું ટુથ ટેબલ લખો અને લોજિક સર્કિટ દોરો

જવાબ

8:1 માલ્ટિપ્લેક્સર ડિજાઇન:

- 8 ડેટા ઈન્પુટ ($I_0 I_7$)
- 3 સિલેક્ટ લાઇન (S_2, S_1, S_0)
- 1 આઉટપુટ (Y)

ટુથ ટેબલ:

S₂	S₁	S₀	આઉટપુટ Y
0	0	0	I ₀
0	0	1	I ₁
0	1	0	I ₂
0	1	1	I ₃
1	0	0	I ₄
1	0	1	I ₅
1	1	0	I ₆
1	1	1	I ₇

બુલિયન સમીકરણ: $Y = S'_2 \cdot S'_1 \cdot S'_0 \cdot I_0 + S'_2 \cdot S'_1 \cdot S_0 \cdot I_1 + S'_2 \cdot S_1 \cdot S'_0 \cdot I_2 + S'_2 \cdot S_1 \cdot S_0 \cdot I_3 + S_2 \cdot S'_1 \cdot S'_0 \cdot I_4 + S_2 \cdot S'_1 \cdot S_0 \cdot I_5 + S_2 \cdot S_1 \cdot S'_0 \cdot I_6 + S_2 \cdot S_1 \cdot S_0 \cdot I_7$

લોજિક સર્કિટ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph TD
    I0[IO] {"-{-}{}} AND0([\\&])
    I1[I1] {"-{-}{}} AND1([\\&])
    I2[I2] {"-{-}{}} AND2([\\&])
    I3[I3] {"-{-}{}} AND3([\\&])
    I4[I4] {"-{-}{}} AND4([\\&])
    I5[I5] {"-{-}{}} AND5([\\&])
    I6[I6] {"-{-}{}} AND6([\\&])
    I7[I7] {"-{-}{}} AND7([\\&])

    S0n["S0{"}] {"-}{-}{}} AND0
    S1n["S1{"}] {"-}{-}{}} AND0
    S2n["S2{"}] {"-}{-}{}} AND0

    S0["S0"] {"-{-}{}} AND1
    S1n {"-{-}{}} AND1
    S2n {"-{-}{}} AND1

    S0n {"-{-}{}} AND2
    S1["S1"] {"-{-}{}} AND2
    S2n {"-{-}{}} AND2

    S0 {"-{-}{}} AND3
    S1 {"-{-}{}} AND3
    S2n {"-{-}{}} AND3

    S0n {"-{-}{}} AND4
    S1n {"-{-}{}} AND4
    S2["S2"] {"-{-}{}} AND4

    S0 {"-{-}{}} AND5
    S1n {"-{-}{}} AND5
    S2 {"-{-}{}} AND5

    S0n {"-{-}{}} AND6
    S1 {"-{-}{}} AND6
    S2 {"-{-}{}} AND6

    S0 {"-{-}{}} AND7
    S1 {"-{-}{}} AND7
    S2 {"-{-}{}} AND7

    AND0 {"-{-}{}} OR([1])
    AND1 {"-{-}{}} OR
    AND2 {"-{-}{}} OR
    AND3 {"-{-}{}} OR
    AND4 {"-{-}{}} OR
    AND5 {"-{-}{}} OR
    AND6 {"-{-}{}} OR
    AND7 {"-{-}{}} OR

    OR {"-{-}{}} Y[Y]
{Highlighting}
{Shaded}
```

સ્મૃતિસહાય: “આઠ ઇનપુટ, ત્રણ સિલેક્ટ, ડિકોડ કરો અને આઉટપુટ મેળવવા OR કરો”

પ્રશ્ન 3 [વૈકલ્પિક પ્રશ્ન] (અ) [3 ગુણ]

4-bit બાયનરી પેરેલલ એડરનો બ્લોક ડાયાગ્રામ દોરો

જવાબ

4-bit બાયનરી પેરેલલ એડર: બે 4-bit બાયનરી નંબર ઉમેરતી અને 4-bit સરવાળો અને એક કેરી આઉટપુટ ઉત્પત્ત કરતી સર્કિટ.

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    AO[AO] {--{-}{}} FAO[FA]
    BO[BO] {--{-}{}} FAO
    Cin[Cin=0] {--{-}{}} FAO
    FAO {--{-}{}} S0[S0]

    A1[A1] {--{-}{}} FA1[FA]
    B1[B1] {--{-}{}} FA1
    FAO {--{-}C1{--}{-}{}} FA1
    FA1 {--{-}{}} S1[S1]

    A2[A2] {--{-}{}} FA2[FA]
    B2[B2] {--{-}{}} FA2
    FA1 {--{-}C2{--}{-}{}} FA2
    FA2 {--{-}{}} S2[S2]

    A3[A3] {--{-}{}} FA3[FA]
    B3[B3] {--{-}{}} FA3
    FA2 {--{-}C3{--}{-}{}} FA3
    FA3 {--{-}{}} S3[S3]

    FA3 {--{-}C4{--}{-}{}} Cout[Cout]
{Highlighting}
{Shaded}
```

ઘટકો:

- ચાર કૂલ એડર (FA) કેસ્કેડમાં જોડાયેલા
- દરેક FA સંબંધિત બિટ્સ અને અગાઉના તબક્કાની કેરી ઉમરે છે
- પ્રારંભિક કેરી-ઇન (Cin) સામાન્ય રીતે 0 હોય છે

સ્મૃતિસહાય: “ચાર FA જોડાયેલા, કેરીઓ વર્ચ્યેથી પસાર થાય છે”

પ્રશ્ન 3 [વૈકલ્પિક પ્રશ્ન] (બ) [4 ગુણ]

લોજિક સર્કિટ અને ટૂથ ટેબલની મદદથી કૂલ એડર સમજાવો

જવાબ

કૂલ એડર: એક કોમ્પ્બીનેશનલ સર્કિટ જે ત્રણ બાયનરી અંક (બે ઇનપુટ અને એક કેરી-ઇન) ઉમરે છે અને સરવાળો અને કેરી આઉટપુટ ઉત્પત્ત કરે છે.

લોજિક સર્કિટ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    A[A] {--{-}{}} XOR1([])
```

```

B[B] {-{-}{} XOR1}
XOR1 {-{-}{} XOR2([ ])}
Cin[Cin] {-{-}{} XOR2}
XOR2 {-{-}{} Sum[Sum]}

A {-{-}{} AND1([ \& ])}
B {-{-}{} AND1}
XOR1 {-{-}{} AND2([ \& ])}
Cin {-{-}{} AND2}
AND1 {-{-}{} OR([ 1 ])}
AND2 {-{-}{} OR}
OR {-{-}{} Cout[Carry out]}
{Highlighting}
{Shaded}

```

ટ્રૂથ ટેબલ:

A	B	Cin	Sum	Cout
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

બુલિયન સમીકરણ:

- $\text{Sum} = A \oplus B \oplus \text{Cin}$
- $\text{Cout} = A \cdot B + (A) \cdot \text{Cin}$

સ્મૃતિસહાય: "ત્રણોય XOR કરો સમ માટે, ANDsને OR કરો કેરી માટે"

પ્રશ્ન 3 [વૈકલ્પિક પ્રશ્ન] (ક) (I) [૩ ગુણ]

લોજિક સર્કિટ અને ટ્રૂથ ટેબલની મદદથી 4:1 મલ્ટિપ્લિક્સર સમજાવો

જવાબ

4:1 મલ્ટિપ્લિક્સર: એક ડિજિટલ સ્વિચ જે બે સિલેક્ટ લાઇન્સના આધારે ચાર ઇનપુટ લાઇન્સમાંથી એકને પસંદ કરે છે અને તેને આઉટપુટથી જોડે છે.

લોજિક સર્કિટ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph TD
    I0[I0] {-{-}{} AND0([ \& ])}
    I1[I1] {-{-}{} AND1([ \& ])}
    I2[I2] {-{-}{} AND2([ \& ])}
    I3[I3] {-{-}{} AND3([ \& ])}

    S0n["S0{"] {-}{}{-}{} AND0}
    S1n["S1{"] {-}{}{-}{} AND0

    S0["S0"] {-{-}{} AND1}
    S1n {-{-}{} AND1}

    S0n {-{-}{} AND2}
    S1["S1"] {-{-}{} AND2}

```

```

S0 {-{-}{}} AND3}
S1 {-{-}{}} AND3}

AND0 {-{-}{}} OR([1])
AND1 {-{-}{}} OR}
AND2 {-{-}{}} OR}
AND3 {-{-}{}} OR}

OR {-{-}{}} Y[Y]
{Highlighting}
{Shaded}

```

ટુથ ટેબલ:

S1	S0	આઉટપુટ Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3

બુલિયન સમીકરણ: $Y = S1' \cdot S0' \cdot I0 + S1' \cdot S0 \cdot I1 + S1 \cdot S0' \cdot I2 + S1 \cdot S0 \cdot I3$
સ્મૃતિસહાય: "બે સિલેક્ટ લાઇન ચાર ઇનપુટમાંથી એક પસંદ કરે છે"

પ્રશ્ન 3 [વૈકલ્પિક પ્રશ્ન] (ક) (II) [4 ગુણ]

બે 4:1 માલિટિપ્લિકેશનો ઉપયોગ કરીને 8:1 માલિટિપ્લિકેશર ડિજાઇન કરો.

જવાબ

ડિજાઇન અભિગમ: 8:1 MUX બનાવવા માટે બે 4:1 MUX અને એક 2:1 MUX વાપરો.
1. પ્રથમ 4:1 MUX ઇનપુટ I0-I3 સંબાને છે, સિલેક્ટ લાઇન S0,S1નો ઉપયોગ કરીને
2. બીજો 4:1 MUX ઇનપુટ I4-I7 સંબાને છે, સિલેક્ટ લાઇન S0,S1નો ઉપયોગ કરીને
3. 2:1 MUX બે 4:1 MUXના આઉટપુટ વચ્ચે S2નો ઉપયોગ કરીને પસંદગી કરે છે

બ્લોક ડાયાગ્રામ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph TD
I0[IO] {-{-}{}} MUX1([4:1 MUX])
I1[I1] {-{-}{}} MUX1
I2[I2] {-{-}{}} MUX1
I3[I3] {-{-}{}} MUX1

I4[I4] {-{-}{}} MUX2([4:1 MUX])
I5[I5] {-{-}{}} MUX2
I6[I6] {-{-}{}} MUX2
I7[I7] {-{-}{}} MUX2

S0[S0] {-{-}{}} MUX1
S1[S1] {-{-}{}} MUX1
S0 {-{-}{}} MUX2
S1 {-{-}{}} MUX2

MUX1 {-{-}{}} MUX3([2:1 MUX])
MUX2 {-{-}{}} MUX3
S2[S2] {-{-}{}} MUX3

MUX3 {-{-}{}} Y[Y]
{Highlighting}

```

{Shaded}

ટૂથ ટેબલ:

S2	S1	S0	આઉટપુટ Y
0	0	0	I0
0	0	1	I1
0	1	0	I2
0	1	1	I3
1	0	0	I4
1	0	1	I5
1	1	0	I6
1	1	1	I7

સ્મૃતિસહાય: "S0,S1 દરેક 4:1 MUXમાંથી પસંદ કરે છે, S2 તેમની વચ્ચે પસંદ કરે છે"

પ્રશ્ન 4(અ) [3 ગુણ]

સિકવન્સીયલ સર્કિટની વ્યાખ્યા લખો. તેના બે ઉદાહરણ લખો

જવાબ

સિકવન્સીયલ સર્કિટ: એક ડિજિટલ સર્કિટ જેનું આઉટપુટ માત્ર વર્તમાન ઇનપુટ પર જ નહીં પણ ઇનપુટના ભૂતકાળના ક્રમ (ઇતિહાસ/અગાઉની સ્થિતિ) પર પણ આધારિત હોય છે.

મુખ્ય લક્ષણો:

- મેમરી એલિમેન્ટ્સ (ફિલિપ-ફ્લોપ) ધરાવે છે
- આઉટપુટ વર્તમાન ઇનપુટ અને અગાઉની સ્થિતિઓ બંને પર આધારિત છે
- સામાન્ય રીતે ફીડબેક પાથ સમાવે છે
- સિંકોનાઇઝેશન માટે કલોક સિથલની જરૂર પડે છે (સિંકોનસ સર્કિટ માટે)

સિકવન્સીયલ સર્કિટના ઉદાહરણો:

- ફિલિપ-ફ્લોપ (SR, JK, D, T)
- રજિસ્ટર (શિક્ફટ રજિસ્ટર)
- કાઉન્ટર (બાયનરી, ડિકેડ, રિંગ કાઉન્ટર)
- સ્ટેટ મશીન
- મેમરી યુનિટ

Table 12: સિકવન્સીયલ vs કોમ્બીનેશનલ સર્કિટ

લક્ષણ	સિકવન્સીયલ સર્કિટ	કોમ્બીનેશનલ સર્કિટ
મેમરી	હા	ના
ફીડબેક	સામાન્ય રીતે	ના
આઉટપુટ આધારિત	વર્તમાન & અગાઉના ઇનપુટ	માત્ર વર્તમાન ઇનપુટ
કલોક જરૂરી	સામાન્ય રીતે	ના
ઉદાહરણો	ફિલિપ-ફ્લોપ, કાઉન્ટર	માલ્ટિપ્લેક્સર, એડર

સ્મૃતિસહાય: "સિકવન્સીયલ ઇતિહાસ યાદ રાખે છે, કોમ્બીનેશનલ માત્ર વર્તમાન જાણે છે"

પ્રશ્ન 4(બ) [4 ગુણ]

ડિકેડ કાઉન્ટર ડિઝાઇન કરો

જવાબ

ડિકેડ કાઉન્ટર: એક સિકવન્સીયલ સર્કિટ જે 0 થી 9 (ડિસિમલ) સુધી ગણે છે અને પછી 0 પર રીસેટ થાય છે.

JK ફિલિપ-ફ્લોપનો ઉપયોગ કરી ડિઝાઇન:

- 4 બિટ બાયનરી નંબર રજૂ કરવા માટે 4 JK ફિલિપ-ફ્લોપ (Q3,Q2,Q1,Q0) જરૂરી છે
- 0000 થી 1001 (0-9 ડિસિમલ) સુધી ગણે છે પછી રીસેટ થાય છે

સ્ટેટ ટેબલ:

વર્તમાન સ્થિતિ	આગામી સ્થિતિ
0 (0000)	1 (0001)
1 (0001)	2 (0010)
2 (0010)	3 (0011)
3 (0011)	4 (0100)
4 (0100)	5 (0101)
5 (0101)	6 (0110)
6 (0110)	7 (0111)
7 (0111)	8 (1000)
8 (1000)	9 (1001)
9 (1001)	0 (0000)

અલોક ડાયાગ્રામ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
    CLK[Clock] {-{-}{}} FF0[JK0]
    CLK {-{-}{}} FF1[JK1]
    CLK {-{-}{}} FF2[JK2]
    CLK {-{-}{}} FF3[JK3]

    AND((\&)) {-{-}{}} R[Reset]
    Q1 {-{-}{}} AND
    Q3 {-{-}{}} AND
    R {-{-}{}} FF0
    R {-{-}{}} FF1
    R {-{-}{}} FF2
    R {-{-}{}} FF3

    FF0 {-{-}Q0{-}{-}{-}{}} Q0[Q0]
    FF1 {-{-}Q1{-}{-}{-}{}} Q1[Q1]
    FF2 {-{-}Q2{-}{-}{-}{}} Q2[Q2]
    FF3 {-{-}Q3{-}{-}{-}{}} Q3[Q3]

    FF0 {-{-}Q0{-}{-}{-}{}} FF1
    FF1 {-{-}Q1{-}{-}{-}{}} FF2
    FF2 {-{-}Q2{-}{-}{-}{}} FF3
{Highlighting}
{Shaded}

```

J-K ઇનપુટ સમીકરણ:

- $J_0 = K_0 = 1$ (દેરેક કલોક પર ટોગલ)
- $J_1 = K_1 = Q_0$
- $J_2 = K_2 = Q_1 \cdot Q_0$
- $J_3 = K_3 = Q_2 \cdot Q_1 \cdot Q_0$

રીસેટ સ્થિતિ: જ્યારે $Q_3 \cdot Q_1 = 1$ (સ્થિતિ 1010), બધા ક્લિપ-ફ્લોપ રીસેટ કરો

સ્મૃતિસહાય: "BCD ગણો, 9 પછી રીસેટ"

પ્રશ્ન 4(ક)ા(I) [3 ગુણ]

NOR ગેટની મદદથી S-R ક્લિપ-ફ્લોપ સમજાવો. તેનો લોજિક સિમ્બોલ દોરો અને ટૂથ ટેબલ લખો.

જવાબ

NOR ગેટથી S-R ક્લિપ-ફ્લોપ: બે કોસ-કપદ NOR ગેટમાંથી બનેલું એક મૂળભૂત ક્લિપ-ફ્લોપ જે એક બિટની માહિતી સંગ્રહિત કરી શકે છે. લોજિક સર્કિટ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
    S[S] {-{-}{}} NOR1([1])
    NOR2 {-{-}Q{}{-}{-}{}} NOR1

    R[R] {-{-}{}} NOR2([1])
    NOR1 {-{-}Q{-}{-}{-}{}} NOR2

    NOR1 {-{-}Q{-}{-}{-}{}} Q[Q]
    NOR2 {-{-}Q{}{-}{-}{-}{}} Qn[Q{}]

{Highlighting}
{Shaded}

```

લોજિક સિમ્બોલ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
    S[S] {-{-}{}} SR[SR]
    R[R] {-{-}{}} SR
    SR {-{-}Q{-}{-}{-}{}} Q[Q]
    SR {-{-}Q{}{-}{-}{-}{}} Qn[Q{}]

{Highlighting}
{Shaded}

```

ટુથ ટેબલ:

S	R	Q (આગામી)	Q' (આગામી)	ઓપરેશન
0	0	Q (અગાઉની)	Q' (અગાઉની)	મેમરી (કોઈ ફેરફાર નહીં)
0	1	0	1	રીસેટ
1	0	1	0	સેટ
1	1	0	0	અમાન્ય (ટાળો)

સ્મૃતિસહાય: "S થી 1 સેટ થાય, R થી 0 રીસેટ થાય, બંને એકસાથે અમાન્ય સ્થિતિ આપે"

પ્રશ્ન 4(ક)(II) [4 ગુણ]

NAND ગેટની મદદથી S-R ફ્લિપ-ફ્લોપ સમજાવો. S-R ફ્લિપ-ફ્લોપની મર્યાદા લખો

જવાબ

NAND ગેટથી S-R ફ્લિપ-ફ્લોપ: બે કોસ-કપડા NAND ગેટમાંથી બનેલું એક મૂળભૂત ફ્લિપ-ફ્લોપ.

લોજિક સર્કિટ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
    S[S] {-{-}{}} NAND1([\&])
    NAND2 {-{-}Q{-}{-}{-}{}} NAND1

    R[R] {-{-}{}} NAND2([\&])
    NAND1 {-{-}Q{}{-}{-}{-}{}} NAND2

    NAND1 {-{-}Q{}{-}{-}{-}{}} Qn[Q{}]
    NAND2 {-{-}Q{-}{-}{-}{}} Q[Q]

{Highlighting}

```

{Shaded}

દૂધ ટેબલ:

S	R	Q (આગામી)	Q' (આગામી)	ઓપરેશન
1	1	Q (અગાઉની)	Q' (અગાઉની)	મેમરી (કોઈ ફેરફાર નહીં)
1	0	1	0	સેટ
0	1	0	1	રીસેટ
0	0	1	1	અમાન્ય (ટાળો)

SR ફિલ્પ-ફ્લોપની મર્યાદાઓ:

1. અમાન્ય સ્થિતિ: જ્યારે $S=1, R=1$ (NOR માટે) અથવા $S=0, R=0$ (NAND માટે), આઉટપુટ અનિશ્ચિત રહે છે
2. રેસ કન્ડિશન: જ્યારે ઇનપુટ એકસાથે બદલાય છે, ત્યારે અંતિમ સ્થિતિ અનિશ્ચિત હોઈ શકે છે
3. કલોકિંગ મેન્ઝિઝમ નથી: અન્ય ડિજિટલ ઘટકો સાથે સિંકોનાઇઝ થઈ શક્તું નથી
4. એજ-ટ્રિગર નથી: ટ્રાક પલ્સને વિશ્વસનીય રીતે પ્રતિક્રિયા આપી શક્તું નથી
5. અનિયુનિય ટોગલિંગ: નોઇજ કે જિલ્બને પ્રતિક્રિયા આપી શકે છે

Table 13: NAND vs NOR SR ફિલ્પ-ફ્લોપ

લક્ષણ	NAND SR ફિલ્પ-ફ્લોપ	NOR SR ફિલ્પ-ફ્લોપ
સર્કિય ઇનપુટ	લો (0)	હાઇ (1)
નિયુનિય ઇનપુટ	હાઇ (1)	લો (0)
અમાન્ય સ્થિતિ	$S=0, R=0$	$S=1, R=1$

સ્મૃતિસહાય: "NAND: ઇનપુટ એક્ટિવ-લો, NOR: ઇનપુટ એક્ટિવ-હાઇ; બંનેમાં એક અમાન્ય સ્થિતિ છે"

પ્રશ્ન 4 [વૈકલ્પિક પ્રશ્ન] (અ) [3 ગુણ]

ફિલ્પ-ફ્લોપની વ્યાખ્યા લખો. ફિલ્પ-ફ્લોપના પ્રકાર લખો

જવાબ

ફિલ્પ-ફ્લોપ: એક મૂળભૂત સિકવન-સીથલ ડિજિટલ સર્કિટ જે એક બિટની માહિતી સંગ્રહિત કરી શકે છે અને વે સ્થાયી સ્થિતિઓ (0 અથવા 1) ધરાવે છે. તે ડિજિટલ સિસ્ટમમાં મૂળભૂત મેમરી એલિમેન્ટ તરીકે કામ કરે છે.

મુખ્ય લક્ષણો:

- બાયસ્ટેબલ મલ્ટિવાયબ્રેટર (વે સ્થાયી સ્થિતિઓ)
- જાળાં સુધી બદલવાનો નિર્દેશ ન અપાય ત્યાં સુધી પોતાની સ્થિતિ અનિશ્ચિત સમય સુધી જાળવી રાખી શકે છે
- રજિસ્ટર, કાઉન્ટર અને મેમરી સર્કિટ માટે મૂળભૂત બિલ્ડિંગ બ્લોક બને છે
- કલોક સિશ્વલ (સિંકોનસ) અથવા લેવલ ચેન્જ (એસિંકોનસ) દ્વારા ટ્રિગર થઈ શકે છે

ફિલ્પ-ફ્લોપના પ્રકાર:

ફિલ્પ-ફ્લોપ પ્રકાર	વર્ણન
SR (સેટ-રીસેટ)	સૌથી મૂળભૂત ફિલ્પ-ફ્લોપ જેમાં સેટ અને રીસેટ ઇનપુટ હોય છે
JK	SR ફિલ્પ-ફ્લોપની સુધારેલી આવૃત્તિ જે અમાન્ય સ્થિતિ દૂર કરે છે
D (ડેટા)	ઇનપુટ D પરસો મૂલ્ય સંગ્રહિત કરે છે, ડેટા સ્ટોરેજ માટે વપરાય છે
T (ટોગલ)	ટ્રિગર થાય સ્થિતિ બદલે છે, કાઉન્ટર માટે ઉપયોગી
માસ્ટર-સ્લેવ	રેસ કન્ડિશન અટકાવતું બે-તબક્કાનું ફિલ્પ-ફ્લોપ

સ્મૃતિસહાય: "એક સિંગલ સ્ટોરેજ: SR, JK, D, T"

પ્રશ્ન 4 [વૈકલ્પિક પ્રશ્ન] (બ) [4 ગુણ]

3-bit રિંગ કાઉન્ટર ડિઝાઇન કરો

જવાબ

રિંગ કાઉન્ટર: એક સર્ક્યુલર શિફ્ટ રજિસ્ટર જેમાં ફક્ત એક બિટ સેટ (1) હોય છે અને બાકી બધા રીસેટ (0) હોય છે. એકમાત્ર સેટ બિટ કલોક થતાં રજિસ્ટરમાં "ફરે" છે.

D ફિલ્પ-ફ્લોપનો ઉપયોગ કરી ડિજાઇન:

- 3-bit કાઉન્ટર માટે 3 D ફિલ્પ-ફ્લોપ જરૂરી છે
- પ્રારંભિક સ્થિતિ: 100, પછી 010, 001, અને પાછા 100 પર જાય છે

સ્ટેટ ટેબલ:

વર્તમાન સ્થિતિ	આગામી સ્થિતિ
100	010
010	001
001	100

બ્લોક ડાયાગ્રામ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    CLK[Clock] {-{-}{}} FF0[DO]
    CLK {-{-}{}} FF1[D1]
    CLK {-{-}{}} FF2[D2]

    FF0 {-{-}Q0{-}{-}{-}{}} Q0[Q0]
    FF1 {-{-}Q1{-}{-}{-}{}} Q1[Q1]
    FF2 {-{-}Q2{-}{-}{-}{}} Q2[Q2]

    Q2 {-{-}{}} FF0
    Q0 {-{-}{}} FF1
    Q1 {-{-}{}} FF2

    PRESET[Preset/Clear] {-.{-}{}} FF0
    PRESET {-.{-}{}} FF1
    PRESET {-.{-}{}} FF2

{Highlighting}
{Shaded}
```

D ઇનપુટ સમીકરણ:

- $D_0 = Q_2$
- $D_1 = Q_0$
- $D_2 = Q_1$

પ્રારંભિક સ્થિતિ સેટિંગ: FF0ને 1 પર પ્રીસેટ કરો, FF1 અને FF2ને 0 પર કિલ્યર કરો
સ્મૃતિસહાય: "એક હોટ બિટ વર્તુળમાં ફેરે છે"

પ્રશ્ન 4 [વૈકલ્પિક પ્રશ્ન] (ક)(િ) [૩ ગુણ]

લોજિક સિમ્બોલ અને ટુથ ટેબલની મદદથી J-K ફિલ્પ-ફ્લોપ સમજાવો

જવાબ

J-K ફિલ્પ-ફ્લોપ: SR ફિલ્પ-ફ્લોપની સુધારેલી આવૃત્તિ જે અમાન્ય સ્થિતિ દૂર કરે છે અને બધા ઇનપુટ સંયોજનોમાં સચોટ વર્તન દર્શાવે છે.
લોજિક સિમ્બોલ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
    J[J] {-{-}{}} JK[JK]
    K[K] {-{-}{}} JK
    CLK[Clock] {-{-}{}} JK
    JK {-{-}Q{-}{-}{-}{}} Q[Q]
    JK {-{-}Q{}{-}{-}{-}{}} Qn[Q{}]
```

{Highlighting}
{Shaded}

ટ્રૂથ ટેબલ:

J	K	Q (આગામી)	ઓપરેશન
0	0	Q (આગામી)	કોઈ ફેરફાર નહીં
0	1	0	રીસેટ
1	0	1	સેટ
1	1	Q' (આગામી)	ટોગલ

મુખ્ય લક્ષણો:

- જ્યારે
 $J=K=1$, ફિલિપ-ફલોપ ટોગલ થાય છે (વિપરીત સ્થિતિમાં જાય છે)
- SR ફિલિપ-ફલોપ જેવી કોઈ અમાન્ય સ્થિતિ નથી
- બધા ઓપરેશન કરી શકે છે: સેટ, રીસેટ, હોલ્ડ, ટોગલ

સ્મૃતિસહાય: "J સેટ કરે, K રીસેટ કરે, બંને ટોગલ કરે, કોઈ નહીં યાદ રાખે"

પ્રશ્ન 4 [વૈકલ્પિક પ્રશ્ન] (ક)(II) [4 ગુણ]

J-K ફિલિપ-ફલોપનો ઉપયોગ કરી D ફિલિપ-ફલોપ અને T ફિલિપ-ફલોપની લોજિક સક્રિયા દોરો

જવાબ

JK ફિલિપ-ફલોપનો ઉપયોગ કરી D ફિલિપ-ફલોપ:

JKને D ફિલિપ-ફલોપમાં બદલવા માટે:

- D ઇનપુટને J સાથે જોડો
- D' (NOT D)ને K સાથે જોડો

લોજિક સક્રિયા:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
D[D] {-{-}{}} J[J]
D {-{-}{}} NOT[NOT]
NOT {-{-}{}} K[K]
J {-{-}{}} JK[JK Flip{-}flop]
K {-{-}{}} JK
CLK[Clock] {-{-}{}} JK
JK {-{-}Q{-}{-}{-}{-}{-} Q[Q]}
JK {-{-}Q{-}{-}{-}{-}{-} Qn[Q{}]}}

{Highlighting}
{Shaded}
```

JK ફિલિપ-ફલોપનો ઉપયોગ કરી T ફિલિપ-ફલોપ:

JKને T ફિલિપ-ફલોપમાં બદલવા માટે:

- T ઇનપુટને J અને K બંને સાથે જોડો

લોજિક સક્રિયા:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph LR
T[T] {-{-}{}} J[J]
T {-{-}{}} K[K]
J {-{-}{}} JK[JK Flip{-}flop]
K {-{-}{}} JK
CLK[Clock] {-{-}{}} JK
```

$JK \{-\{-\}Q\{-\}\{-\}\{\} Q [Q]\}$
 $JK \{-\{-\}Q\{\}-\{-\}\{-\}\{\} Qn [Q\{\}] \}$
{Highlighting}
{Shaded}

ટુથ ટેબલ:
D ફ્લિપ-ફ્લોપ:

D	Q (આગામી)	ઓપરેશન
0	0	રીસેટ
1	1	સેટ

T ફ્લિપ-ફ્લોપ:

T	Q (આગામી)	ઓપરેશન
0	Q (આગાઉની)	કોઈ ફેરફાર નહીં
1	Q' (આગાઉની)	ટોગલ

સ્મૃતિસહાય: "D સીધું અનુસરે, T સાચું હોય ત્યારે ટોગલ થાય"

પ્રશ્ન 5(અ) [3 ગુણ]

RAM અને ROM-ની સરખામણી કરો

જવાબ

RAM (Random Access Memory) vs ROM (Read-Only Memory):

Table 14: RAM vs ROM સરખામણી

લક્ષણ	RAM	ROM
પૂર્ણ નામ	Random Access Memory	Read-Only Memory
ડેટા નિભાવણી	અસ્થાયી (પાવર બંધ થતાં ડેટા ગુમાવે)	સ્થાયી (પાવર વિના પણ ડેટા જળવાય)
વાંચન/લેખન ક્ષમતા	વાંચન અને લેખન બંને	મુખ્યત્વે માત્ર વાંચન (PROM, EPROM, EEPROM સિવાય)
ગતિ	વધુ ઝડપી	ધીમી
બિટ દીઠ ખર્ચ	વધુ	ઓછો
ઉપયોગો	અસ્થાયી ડેટા સ્ટોરેજ, સર્કિય પ્રોગ્રામ એક્ઝિક્યુશન	બૂટ સ્ટ્રોચનાઓ, ફર્મવેર, કાયમી ડેટા
પ્રકારો	SRAM, DRAM	Mask ROM, PROM, EPROM, EEPROM, Flash
સેલ જટિલતા	વધુ જટિલ	સરળ

સ્મૃતિસહાય: "RAM વાંચે અને સુધારે (પણ ભૂલી જાય), ROM શાટાઉન પર યાદ રાખે (પણ નિશ્ચિત)"

પ્રશ્ન 5(બ) [4 ગુણ]

સિરિયલ ઇન સિરિયલ આઉટ શિફ્ટ રજીસ્ટર સમજાવો

જવાબ

સિરિયલ ઇન સિરિયલ આઉટ (SISO) શિફ્ટ રજીસ્ટર: એક સિકવન્સીયલ સર્કિટ જે ઇનપુટ અને આઉટપુટ બંને પર ડેટાને એક સમયે એક બિટ શિફ્ટ કરે છે.

કાર્યપદ્ધતિ:

- ડેટા સિરિયલી એક બિટ એક વખતે દાખલ થાય છે
- દરેક કલોક પલ્સ પર દરેક બિટ રજિસ્ટરમાંથી શિકૃ થાય છે
- ડેટા સિરિયલી એક બિટ એક વખતે બહાર નીકળે છે
- પ્રથમ-ઇન, પ્રથમ-આઉટ કાર્યપદ્ધતિ

બ્લોક ડાયાગ્રામ:

Mermaid Diagram (Code)

```

{Shaded}
{Highlighting} []
graph LR
    SI[Serial In] {--> FF0[FF0[D0]]}
    CLK[Clock] {--> FF0}
    CLK {--> FF1}
    CLK {--> FF2}
    CLK {--> FF3}

    FF0 {--> Q0{Q0[-]{-}{-}} FF1[D1]}
    FF1 {--> Q1{Q1[-]{-}{-}} FF2[D2]}
    FF2 {--> Q2{Q2[-]{-}{-}} FF3[D3]}
    FF3 {--> Q3{Q3[-]{-}{-}} S0[Serial Out]}

{Highlighting}
{Shaded}

```

“1011” શિકૃ કરવા માટે ટાઇમિંગ ડાયાગ્રામ:

CLK	_ _ _ _ _
SI	_ _ -----
Q0	----- -----
Q1	----- -----
Q2	----- -----
S0	-----

ઉપયોગો:

- ડિજિટલ સિસ્ટમ વચ્ચે ડેટા ટ્રાન્સમિશન
- સિરિયલ-થી-સિરિયલ ડેટા રૂપાંતર
- સમય વિલંબ સર્કિટ
- સિગ્નલ ફિલ્ટરિંગ

સ્મૃતિસહાય: “બિટ્સ લાઇનમાં પ્રવેશે, શ્રેણીમાં આગળ વધે, ક્રમમાં બહાર નીકળે”

પ્રશ્ન 5(ક) [7 ગુણ]

લોજિક ફેબ્રિલિઝ પર ટૂક નોંધ લખો

જવાબ

લોજિક ફેબ્રિલિઝ: સમાન ઇલેક્ટ્રિકલ લક્ષણો, ફેબ્રિકેશન ટેકનોલોજી અને લોજિક અમલીકરણ સાથેના ડિજિટલ ઇન્ટિગ્રેટેડ સર્કિટના સમૂહો.

મુખ્ય લોજિક ફેબ્રિલિઝ:

1. TTL (ટ્રાન્ઝિસ્ટર-ટ્રાન્ઝિસ્ટર લોજિક):
 - બાયપોલર જંકશન ટ્રાન્ઝિસ્ટર પર આધારિત
 - સ્ટાન્ડર્ડ સિરીઝ: 7400
 - સખ્તાય વોલ્ટેજ: 5V
 - મધ્યમ ઝડપ અને પાવર વપરાશ
 - ઊંચી નોઇજ ઇમ્પ્યુનિટી
 - વેરિયન્ટ: સ્ટાન્ડર્ડ TTL, લો-પાવર TTL (74L), શોટ્કી TTL (74S), એડવાન્ડ શોટ્કી (74AS)
2. CMOS (કોમ્પ્લિમેન્ટરી મેટલ-ઓક્સાઈડ-સેમિકરન્ડર):
 - MOSFETs (P-ટાઇપ અને N-ટાઇપ) પર આધારિત
 - સ્ટાન્ડર્ડ સિરીઝ: 4000, 74C00
 - વ્યાપક સખ્તાય વોલ્ટેજ રેન્જ (3-15V)
 - ખૂબ ઓછો પાવર વપરાશ

- ઊંચી નોઇજ ઇમ્પુનિટી
- સ્ટેટિક ઇલેક્ટ્રિસ્ટી પ્રત્યે સંવેદનશીલ
- એડવાન્સ્ડ વેરિયન્ટ: HC, HCT, AC, ACT, AHC, AHCT સિરીઝ

3. ECL (ઇમિટર-ક્પલ લોજિક):

- ઇમિટર-ક્પલ ટ્રાન્ઝિસ્ટર સાથેના ડિક્રોનિશાયલ એમિલફાયર પર આધારિત
- અત્યંત ઊંચી જડપ (સૌથી જડપી લોજિક ફેબ્રિલી)
- ઊંચો પાવર વપરાશ
- ઓછી નોઇજ ઇમ્પુનિટી
- નેગટિવ સપ્લાય વોલ્ટેજ
- ઊંચી જડપ-પી એપ્લિકેશનમાં વપરાય છે

લોજિક ફેબ્રિલીના મુખ્ય પેરામીટર:

પેરામીટર	વર્ણન
કેન-ઇન	ગેટ સ્વીકારી શકે તેવા ઇનપુટની મહત્તમ સંખ્યા
કેન-આઉટ	એક આઉટપુટ દ્વારા ફ્રાઇવ થઈ શકતા ગેટની મહત્તમ સંખ્યા
નોઇજ માર્જિન	અનિરણીય ઇલેક્ટ્રિકલ નોઇજ/સિશાલ સહન કરવાની ક્ષમતા
પ્રોપેગેશન ડિલે	ઇનપુટ ચે-જ અને તેના તરત પછીના આઉટપુટ ચે-જ વચ્ચેનો સમય વિલંબ
પાવર ડિસિપેશન	ગેટ દ્વારા વપરાતી પાવરની માત્રા
ફિગર ઓફ મેરિટ	જડપ અને પાવરનો ગુણાકાર (ઓછું હોવું વધુ સાર)

સરખામણી કોષ્ટક:

પેરામીટર	TTL	CMOS	ECL
જડપ	મધ્યમ	ઓછી થી ઊંચી	ખૂબ ઊંચી
પાવર વપરાશ	મધ્યમ	ખૂબ ઓછો	ઊંચો
નોઇજ ઇમ્પુનિટી	ઊંચી	ખૂબ ઊંચી	ઓછી
કેન-આઉટ	10	50+	25
સપ્લાય વોલ્ટેજ	5V	3-15V	-5.2V
ઇનપુટ/આઉટપુટ લેવલ	0.8V/2.0V	30%/70% of VDD	-1.75V/-0.9V

સ્મૃતિસહાય: "TTL ટ્રાન્ઝિસ્ટર ટેકનોલોજી, CMOS કર્ટ ઓછો વાપરે છે, ECL એક્સટ્રીમ જડપે કામ કરે છે"

પ્રશ્ન 5 [વૈકલ્પિક પ્રશ્ન] (અ) [૩ ગુણ]

SRAM અને DRAM-ની સરખામણી કરો

જવાબ

SRAM (સ્ટેટિક RAM) vs DRAM (ડાયનેમિક RAM):

Table 15: SRAM vs DRAM સરખામણી

લક્ષણ	SRAM	DRAM
પૂર્ણ નામ	Static Random Access Memory	Dynamic Random Access Memory
સેલ સ્ટ્રક્ચર	6 ટ્રાન્ઝિસ્ટર (ફ્લાપ-ફ્લોપ)	1 ટ્રાન્ઝિસ્ટર + 1 કેપેસિટર
સ્ટોરેજ એલિમેન્ટ	ફ્લાપ-ફ્લોપ	કેપેસિટર
રિફેશિંગ	જરૂરી નથી	સમયાંતરે જરૂરી (ms)
જડપ	વધુ જડપી (એક્સેસ ટાઇમ: 10-30ns)	ધીમી (એક્સેસ ટાઇમ: 60-100ns)
ડેન્સિટી	ઓછી (મોટો સેલ સાઇઝ)	ઊંચી (નાનો સેલ સાઇઝ)
બિટ દીઠ ખર્ચ	વધુ	ઓછો
પાવર વપરાશ	વધુ	ઓછો
ઉપયોગો	કેશ મેમરી, બફર	મુખ્ય મેમરી (RAM)
ડેટા નિભાવણી	જ્યાં સુધી પાવર સપ્લાય થાય	થોડી મિલિસેક્ન્ડ, રિફેશની જરૂર

સ્મૃતિસહાય: "સ્ટેટિક સ્થિર રહે છે છ ટ્રાન્ઝિસ્ટર સાથે, ડાયનેમિક ડ્રેઇન થાય અને નિયમિત રિફેશ જોઈએ"

પ્રશ્ન 5 [વૈકલ્પિક પ્રશ્ન] (બ) [4 ગુણ]

8:3 એનકોડર સમજાવો

જવાબ

8:3 એનકોડર: એક કોમ્બીનેશનલ સર્કિટ જે 8 ઇનપુટ લાઇન્સને 3 આઉટપુટ લાઇન્સમાં રૂપાંતરિત કરે છે, મૂળભૂત રીતે સર્કિય ઇનપુટ લાઇન્સને તેની બાયનરી પોઝિશનમાં રૂપાંતરિત કરે છે.

કાર્યપદ્ધતિ:

- 8 ઇનપુટ લાઇન (I₀I₇)3(Y₂, Y₁, Y₀)
- એક સમયે માત્ર એક ઇનપુટ સર્કિય હોય છે
- આઉટપુટ સર્કિય ઇનપુટના સ્થાનને દર્શાવતો બાયનરી કોડ છે

લોજિક સર્કિટ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph TD
    I1[I1] {-{-}{}} ORO([1])
    I3[I3] {-{-}{}} ORO
    I5[I5] {-{-}{}} ORO
    I7[I7] {-{-}{}} ORO
    ORO {-{-}{}} Y0[Y0]

    I2[I2] {-{-}{}} OR1([1])
    I3[I3] {-{-}{}} OR1
    I6[I6] {-{-}{}} OR1
    I7[I7] {-{-}{}} OR1
    OR1 {-{-}{}} Y1[Y1]

    I4[I4] {-{-}{}} OR2([1])
    I5[I5] {-{-}{}} OR2
    I6[I6] {-{-}{}} OR2
    I7[I7] {-{-}{}} OR2
    OR2 {-{-}{}} Y2[Y2]

{Highlighting}
{Shaded}
```

ટુથ ટેબલ:

ઇનપુટ	આઉટપુટ
I ₇ I ₆ I ₅ I ₄ I ₃ I ₂ I ₁ I ₀	Y ₂ Y ₁ Y ₀
0 0 0 0 0 0 1	0 0 0
0 0 0 0 0 1 0	0 0 1
0 0 0 0 1 0 0	0 1 0
0 0 0 1 0 0 0	0 1 1
0 0 1 0 0 0 0	1 0 0
0 1 0 0 0 0 0	1 0 1
1 0 0 0 0 0 0	1 1 0
1 0 0 0 0 0 0	1 1 1

બુલિયન સમીકરણ:

- Y₀ = I₁ + I₃ + I₅ + I₇
- Y₁ = I₂ + I₃ + I₆ + I₇
- Y₂ = I₄ + I₅ + I₆ + I₇

ઉપયોગો:

- પ્રાયોરિટી એનકોડર
- કીલ્ડૉક એનકોડર
- એડ્યુક્શન ડિકોડર
- ડેટા સિલેક્ટર

સ્મૃતિસહાય: "આઈ ઇનપુટ તેમના સ્થાન ત્રણ બિટમાં બને"

પ્રશ્ન 5 [વૈકલ્પિક પ્રશ્ન] (ક) [૭ ગુણ]

લોજિક ફેબ્રિલિઝ માટે નીચેની વ્યાખ્યાઓ લખો (i) ફેન-ઇન (ii) ફેન-આઉટ (iii) નોઇઝ માર્જિન (iv) પ્રોપેશન ડિલે (v) પાવર ડિસિપેશન

જવાબ

લોજિક ફેબ્રિલિઝના મુખ્ય પેરામીટર:

1. ફેન-ઇન:

- વ્યાખ્યા: લોજિક ગેટ સ્વીકારી શકે તેવા ઇનપુટની મહત્વમ સંખ્યા
- મહત્વ: લોજિક અમલીકરણની જટિલતા નિર્ધારિત કરે છે
- સામાન્ય મૂલ્યો: મોટાભાગની ફેબ્રિલિઝ માટે 2-8
- ઉદાહરણ: 4 ઇનપુટ ધરાવતા AND ગેટનો ફેન-ઇન 4 છે

2. ફેન-આઉટ:

- વ્યાખ્યા: એક ગેટ આઉટપુટ દ્વારા વિશ્વસનીય રીતે ડ્રાઇવ થઈ શકતા સમાન ગેટની મહત્વમ સંખ્યા
- મહત્વ: લોડિંગ ક્ષમતા અને સિસ્ટમ વિસ્તરણક્ષમતા નિર્ધારિત કરે છે
- ગણતરી: આઉટપુટ કરેટ ક્ષમતા અને ઇનપુટ કરેટ જરૂરિયાતો પર આધારિત
- સામાન્ય મૂલ્યો: TTL: 10, CMOS: 50+, ECL: 25

3. નોઇઝ માર્જિન:

- વ્યાખ્યા: અનિયાનીય ઇલેક્ટ્રોલોગિક નોઇઝ/સિગ્નલને સાહન કરવાની સર્કિટની ક્ષમતાનું માપ
- મહત્વ: નોઇઝી વાતાવરણમાં વિશ્વસનીય કાર્યપદ્ધતિ સુનિશ્ચિત કરે છે
- ગણતરી: મિનિમમ હાઇ આઉટપુટ વોલટેજ અને મેક્સિમમ હાઇ ઇનપુટ વોલટેજ વર્ચેનો તફાવત
- સામાન્ય મૂલ્યો: TTL: 0.4V, CMOS: 1.5V-2.25V, ECL: 0.2V

4. પ્રોપેશન ડિલે:

- વ્યાખ્યા: ઇનપુટ ચેન્જ અને તેના તરત પછીના આઉટપુટ ચેન્જ વર્ચેનો સમય વિલંબ
- મહત્વ: મહત્વમ ઓપરેટિંગ ફિક્વન્સી અને ઝડપ નિર્ધારિત કરે છે
- માપન: ઇનપુટ ટ્રાન્ઝિશનના 50% થી આઉટપુટ ટ્રાન્ઝિશનના 50% સુધીનો સમય
- સામાન્ય મૂલ્યો: TTL: 10ns, CMOS: 5-100ns, ECL: 1-2ns

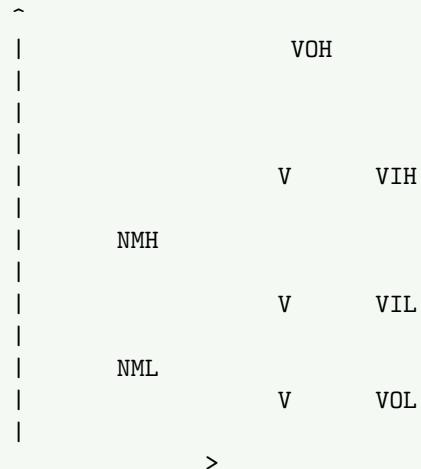
5. પાવર ડિસિપેશન:

- વ્યાખ્યા: લોજિક ગેટ દ્વારા વપરાતી પાવરની માત્રા
- મહત્વ: હીટ જનરેશન, પાવર સપ્લાય જરૂરિયાતો, બેટરી લાઇફને અસર કરે છે
- ગણતરી: સપ્લાય વોલટેજ અને કરેટ ડ્રોનો ગુણાકાર
- સામાન્ય મૂલ્યો: TTL: 10mW, CMOS: 0.1mW (સ્ટેટિક), ECL: 25mW

Table 16: લોજિક ફેબ્રિલી સરખામણી

પેરામીટર	TTL	CMOS	ECL
ફેન-ઇન	3-8	2-અમયાર્દિત	2-4
ફેન-આઉટ	10	50+	25
નોઇઝ માર્જિન	0.4V	1.5V-2.25V	0.2V
પ્રોપેશન ડિલે	10ns	5-100ns	1-2ns
પાવર ડિસિપેશન	10mW	0.1mW (સ્ટેટિક)	25mW

આફ્ટિસ: નોઇજ માર્જિન અને સ્વિચિંગ થ્રેશોલ્ડ



સંબંધો:

- NMH (નોઇજ માર્જિન હાઇ) = VOH(min) - VIH(min)
- NML (નોઇજ માર્જિન લો) = VIL(max) - VOL(max)
- ફિગર ઓફ મેરિટ = પાવર $\times ()$

સ્મૃતિસહાય: "પાંચ ફેક્ટર: ફેન-ઇન ઇનપુટ ગણે, ફેન-આઉટ ગેટ ચલાવે, નોઇજ માર્જિન દખલ સામે લડે, પ્રોપેશન ડિલે ઝડપ માપે, પાવર ડિસિપેશન ગરમી ઉત્પત્ત કરે"