

# Subject Name (Gujarati)

4353206 -- Summer 2025

Semester 1 Study Material

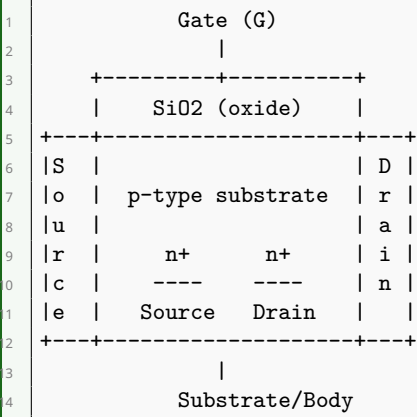
Detailed Solutions and Explanations

## પ્રશ્ન 1(અ) [3 ગુણ]

n-ચેનલ MOSFET ના ભૌતિક બંધારણનું સાફ લેબલવાળું આકૃતિ દોરો.

જવાબ

આકૃતિ:



મુખ્ય ઘટકો:

- સોર્સ: n+ ડોપ્ડ વિસ્તાર જે ઇલેક્ટ્રોન પૂરો પાડે છે
- ડ્રેઇન: n+ ડોપ્ડ વિસ્તાર જે ઇલેક્ટ્રોન એકત્ર કરે છે
- ગેટ: ચેનલને નિયંત્રિત કરતું મેટલ ઇલેક્ટ્રોડ
- ઓક્સાઇડ: SiO2 ઇન્સ્યુલેટિંગ સ્તર
- સબસ્ટ્રેટ: p-ટાઇપ સિલિકોન બોડી

મેમરી ટ્રીક

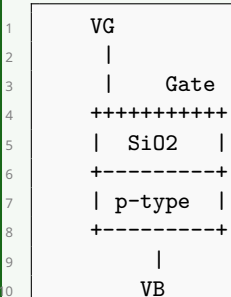
“SOGD - સોર્સ, ઓક્સાઇડ, ગેટ, ડ્રેઇન”

## પ્રશ્ન 1(બ) [4 ગુણ]

એક્સટર્નલ બાયાસ હેઠળ MOS ના ડિપ્લીશન અને ઇન્વર્શનનું એનર્જી બેન્ડ ડાયાગ્રામ MOS બાયોસિંગ ડાયાગ્રામ સાથે દોરો. ઇન્વર્શન રીજનને વિગતવાર સમજાવો.

જવાબ

MOS બાયોસિંગ સર્કિટ:



એનર્જી બેન્ડ ડાયાગ્રામ:

બાયાસ સ્થિતિ	એનર્જી બેન્ડ વર્તન
ડિપ્લીશન	બેન્ડ ઉપરની તરફ વળે છે, હોલ્સ ખતમ થાય છે
ઇન્વર્શન	મજબૂત બેન્ડ બેન્ડિંગ, ઇલેક્ટ્રોન ચેનલ બને છે

ઇન્વર્શન રીજન વિગતો:

- મજબૂત ઇન્વર્શન:  $V_G > V_T$  (થ્રેશોલ્ડ વોલ્ટેજ)
- ઇલેક્ટ્રોન ચેનલ: Si-SiO<sub>2</sub> ઇન્ટરફેસ પર બને છે
- ચેનલ કન્ડક્ટિવિટી: ગેટ વોલ્ટેજ સાથે વધે છે
- થ્રેશોલ્ડ શરત: સરફેસ પોટેન્શિયલ =  $2\phi_F$

મેમરી ટ્રીક

"DIVE - ડિપ્લીશન, ઇન્વર્શન, વોલ્ટેજ, ઇલેક્ટ્રોન્સ"

## પ્રશ્ન 1(ક) [7 ગુણ]

MOSFET ની I-V લાક્ષણિકતા સમજાવો.

જવાબ

I-V લાક્ષણિકતા વિસ્તારો:

વિસ્તાર	શરત	ડ્રેઇન કરંટ
કટઓફ	$V_{GS} < V_T$	$I_D \approx 0$
લિનિયર	$V_{GS} > V_T, V_{DS} < V_{GS} - V_T$	$I_D = \mu_n C_{ox} (W/L) [(V_{GS} - V_T)V_{DS} - V_{DS}^2/2]$
સેચ્યુરેશન	$V_{GS} > V_T, V_{DS} \geq V_{GS} - V_T$	$I_D = (\mu_n C_{ox}/2)(W/L)(V_{GS} - V_T)^2$

લાક્ષણિકતા વક્ર:



મુખ્ય પેરામીટર્સ:

- $\mu_n$ : ઇલેક્ટ્રોન મોબિલિટી
- $C_{ox}$ : ગેટ ઓક્સાઇડ કેપેસિટન્સ
- $W/L$ : પહોળાઈ અને લંબાઈનો રેશિયો
- $V_T$ : થ્રેશોલ્ડ વોલ્ટેજ

ઓપરેટિંગ મોડ્સ:

- એન્હાન્સમેન્ટ: પોઝિટિવ  $V_{GS}$  સાથે ચેનલ બને છે
- સ્ક્વેર લો: સેચ્યુરેશન વિસ્તાર ચતુર્ભુજ સંબંધ અનુસરે છે

મેમરી ટ્રીક

"CLS - કટઓફ, લિનિયર, સેચ્યુરેશન"

## પ્રશ્ન 1(ક) OR [7 ગુણ]

સ્કેલિંગ વ્યાખ્યાયિત કરો. સ્કેલિંગની જરૂરિયાત સમજાવો. સ્કેલિંગની નકારાત્મક અસરોની સૂચિ બનાવો અને સમજાવો.

## જવાબ

વ્યાખ્યા: સ્કેલિંગ એટલે પર્ફોર્મન્સ અને ડેન્સિટી સુધારવા માટે MOSFET ના પરિમાણોમાં વ્યવસ્થિત ઘટાડો. સ્કેલિંગની જરૂરિયાત:

ફાયદો	વર્ણન
વધુ ડેન્સિટી	ચિપ વિસ્તાર દીઠ વધુ ટ્રાન્ઝિસ્ટર
ઝડપી સ્પીડ	ઘટેલી ગેટ ડીલે
ઓછી પાવર	ઘટેલી સ્વિચિંગ એનર્જી
કોસ્ટ રિડક્શન	વેફર દીઠ વધુ ચિપ્સ

સ્કેલિંગ પ્રકારો:

પ્રકાર	ગેટ લંબાઈ	સપ્લાય વોલ્ટેજ	ઓક્સાઇડ જાડાઈ
કોન્સ્ટન્ટ વોલ્ટેજ	□□	સ્થિર	□□
કોન્સ્ટન્ટ ફીલ્ડ	□□	□□	□□

નકારાત્મક અસરો:

- શોર્ટ ચેનલ અસરો: થ્રેશોલ્ડ વોલ્ટેજ રોલ-ઓફ
- હોટ કેરિયર અસરો: ડિવાઇસ ડીગ્રેડેશન
- ગેટ લીકેજ: વધેલો ટનલિંગ કરંટ
- પ્રોસેસ વેરિએશન્સ: મેન્યુફેક્ચરિંગ પડકારો
- પાવર ડેન્સિટી: હોટ ડિસિપેશન સમસ્યાઓ

## મેમરી ટ્રીક

“SHGPP - શોર્ટ ચેનલ, હોટ કેરિયર, ગેટ લીકેજ, પ્રોસેસ, પાવર”

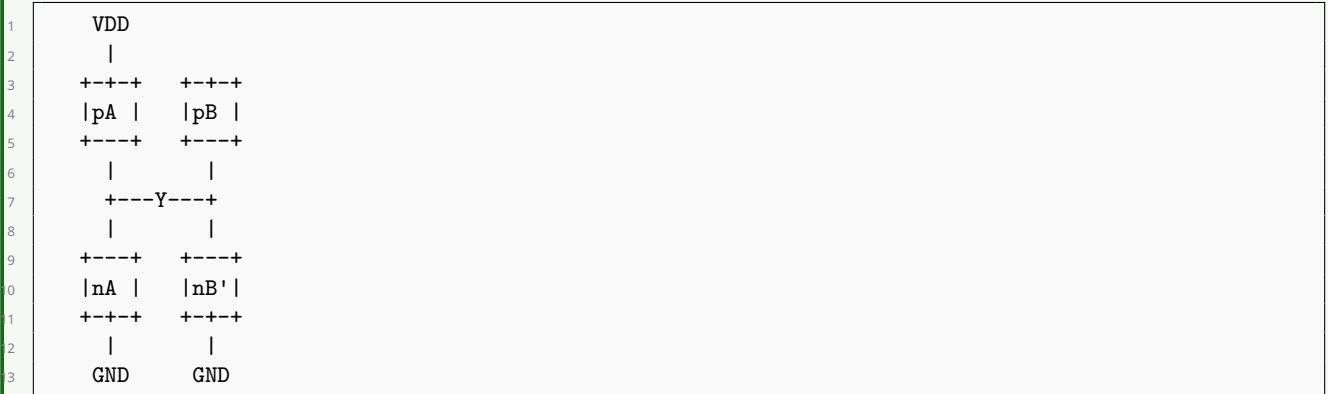
## પ્રશ્ન 2(અ) [3 ગુણ]

CMOS નો ઉપયોગ કરીને  $Y' = (AB' + A'B)$  અમલમાં મૂકો.

## જવાબ

લોજિક વિશ્લેષણ:  $Y' = (AB' + A'B) = A \oplus B (XOR)$

CMOS અમલીકરણ:



ટ્રુથ ટેબલ:

A	B	AB'	A'B	Y'
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

## મેમરી ટ્રીક

“XOR ને કોમ્પ્લિમેન્ટરી સ્વિચિંગ જોઈએ”

### પ્રશ્ન 2(બ) [4 ગુણ]

એન્હાન્સમેન્ટ લોડ ઇન્વર્ટરને તેના સર્કિટ ડાયાગ્રામ સાથે સમજાવો.

#### જવાબ

##### સર્કિટ ડાયાગ્રામ:

```

1  VDD
2  |
3  +---o VG2
4  |
5  +---+ Enhancement
6  |ME | Load
7  +---+
8  |
9  +---o Vout
10 |
11 +---+
12 |MD | Driver
13 +---+
14 |
15 GND
16 |
17 +---o Vin
    
```

##### કન્ફિગરેશન:

ઘટક	પ્રકાર	કનેક્શન
લોડ (ME)	એન્હાન્સમેન્ટ NMOS	ગેટ VDD સાથે જોડાયેલું
ડ્રાઇવર (MD)	એન્હાન્સમેન્ટ NMOS	ગેટ ઇનપુટ છે

##### ઓપરેશન:

- લોડ ટ્રાન્ઝિસ્ટર: એક્ટિવ લોડ રેજિસ્ટર તરીકે કામ કરે છે
- હાઇ આઉટપુટ: લોડ ટ્રાન્ઝિસ્ટરના  $V_T$  દ્વારા મર્યાદિત
- લો આઉટપુટ: ડ્રાઇવરની તાકાત પર આધાર રાખે છે
- ગેરફાયદો: થ્રેશોલ્ડ ડ્રોપ કારણે ખરાબ  $VOH$

##### ટ્રાન્સફર લાક્ષણિકતાઓ:

- $VOH$ :  $VDD - V_T$  (બગડેલો હાઇ લેવલ)
- $VOL$ : ગ્રાઉન્ડ પોટેન્શિયલની નજીક
- નોઇઝ માર્જિન: થ્રેશોલ્ડ લોસ કારણે ઘટેલો

## મેમરી ટ્રીક

“ELI - એન્હાન્સમેન્ટ લોડ ઇન્વર્ટરમાં થ્રેશોલ્ડ સમસ્યા”

### પ્રશ્ન 2(ક) [7 ગુણ]

ઇન્વર્ટરની વોલ્ટેજ ટ્રાન્સફર કેરેક્ટરિસ્ટિક સમજાવો.

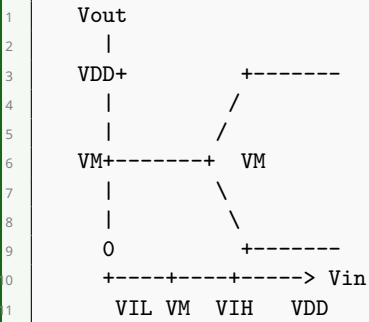
#### જવાબ

##### VTC પેરામીટર્સ:

પેરામીટર	વર્ણન	આદર્શ કિંમત
$VOH$	આઉટપુટ હાઇ વોલ્ટેજ	$VDD$

VOL	આઉટપુટ લો વોલ્ટેજ	0V
VIH	ઇનપુટ હાઇ વોલ્ટેજ	VDD/2
VIL	ઇનપુટ લો વોલ્ટેજ	VDD/2
VM	સ્વિચિંગ થ્રેશોલ્ડ	VDD/2

#### VTC વક્ર:



#### નોઇઝ માર્જિન-સ:

- NMH = VOH - VIH (હાઇ નોઇઝ માર્જિન)
- NML = VIL - VOL (લો નોઇઝ માર્જિન)

#### વિસ્તારો:

- વિસ્તાર 1: ઇનપુટ લો, આઉટપુટ હાઇ
- વિસ્તાર 2: ટ્રાન્ઝિશન વિસ્તાર
- વિસ્તાર 3: ઇનપુટ હાઇ, આઉટપુટ લો

#### ગુણવત્તા મેટ્રિક્સ:

- તીક્ષ્ણ ટ્રાન્ઝિશન: બહેતર નોઇઝ ઇમ્યુનિટી
- સિમેટ્રિક સ્વિચિંગ: VM = VDD/2
- કુલ સ્વિંગ: VOH = VDD, VOL = 0

#### મેમરી ટ્રીક

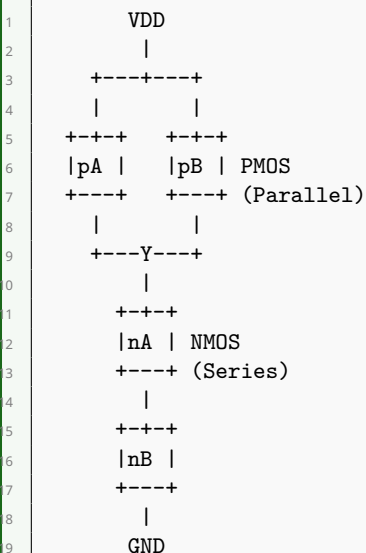
“VTC દર્શાવે છે VOH - VOL, ઇનપુટ થ્રેશોલ્ડ, લાક્ષણિકતા, બધું”

## પ્રશ્ન 2(અ) OR [3 ગુણ]

CMOS નો ઉપયોગ કરીને NAND2 ગેટ સમજાવો.

#### જવાબ

#### CMOS NAND2 સર્કિટ:



ટ્રુથ ટેબલ:

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

ઓપરેશન:

- PMOS નેટવર્ક: પેરેલલ કનેક્શન (પુલ-અપ)
- NMOS નેટવર્ક: સીરીઝ કનેક્શન (પુલ-ડાઉન)
- આઉટપુટ લો: ફક્ત જ્યારે બંને ઇનપુટ હાઇ

મેમરી ટ્રીક

“NAND - નોટ AND, પેરેલલ PMOS, સીરીઝ NMOS”

## પ્રશ્ન 2(બ) OR [4 ગુણ]

રેજિસ્ટિવ લોડ ઇન્વર્ટર સર્કિટના ઓપરેટિંગ મોડ અને VTC સમજાવો.

જવાબ

સર્કિટ કન્ફિગરેશન:

```
1 VDD
2 |
3 R (Load Resistor)
4 |
5 +---o Vout
6 |
7 +-+--+
8 |MN | NMOS Driver
9 +---+
10 |
11 GND
12 |
13 +---o Vin
```

ઓપરેટિંગ મોડ્સ:

ઇનપુટ સ્થિતિ	NMOS સ્થિતિ	આઉટપુટ
Vin = 0	OFF	VOH = VDD
Vin = VDD	ON	VOL = R·ID/(R+RDS)

VTC લાક્ષણિકતાઓ:

- VOH: ઉત્તમ (VDD)
- VOL: R અને RDS રેશિયો પર આધાર રાખે છે
- પાવર વપરાશ: ઇનપુટ હાઇ હોય ત્યારે સ્ટેટિક કરંટ
- ટ્રાન્ઝિશન: રેજિસ્ટિવ લોડ કારણે ધીમું

ડિઝાઇન ટ્રેડ-ઓફ્સ:

- મોટો R: બહેતર VOL, ધીમું સ્વિચિંગ
- નાનો R: ઝડપી સ્વિચિંગ, વધુ પાવર
- એરિયા: રેજિસ્ટર નોંધપાત્ર જગ્યા લે છે

મેમરી ટ્રીક

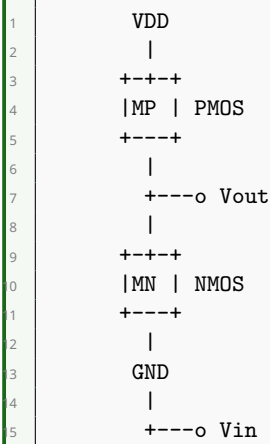
“RLI - રેજિસ્ટિવ લોડમાં અનિવાર્ય પાવર વપરાશ”

## પ્રશ્ન 2(ક) OR [7 ગુણ]

CMOS ઇન્વર્ટર દોરો અને VTC સાથે તેની કામગીરી સમજાવો.

જવાબ

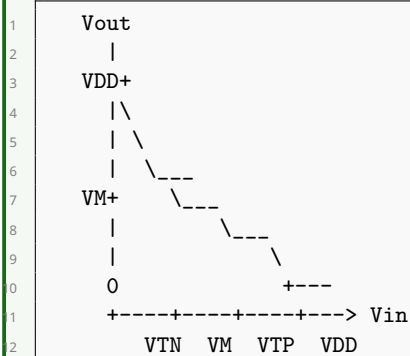
CMOS ઇન્વર્ટર સર્કિટ:



ઓપરેશન વિસ્તારો:

Vin રેન્જ	PMOS	NMOS	Vout	વિસ્તાર
0 થી VTN	ON	OFF	VDD	1
**VTN થી VDD-	VTP	**	ON	ON
**VDD-	VTP	થી VDD**	OFF	ON

VTC વિશ્લેષણ:



મુખ્ય લક્ષણો:

- ઝીરો સ્ટેટિક પાવર: કોઈ DC કરંટ પાથ નથી
- કુલ સ્વિંગ:  $VOH = VDD$ ,  $VOL = 0V$
- હાઇ નોઇઝ માર્જિન્સ:  $NMH = NML \approx 0.4VDD$
- તીક્ષ્ણ ટ્રાન્ઝિશન: ટ્રાન્ઝિશન વિસ્તારમાં હાઇ ગેઇન

ડિઝાઇન વિચારણાઓ:

- $\square$  રેશિયો: સિમેટ્રિક સ્વિચિંગ માટે  $\square N/\square P$
- થ્રેશોલ્ડ મેચિંગ:  $VTN \approx |VTP|$

મેમરી ટ્રીક

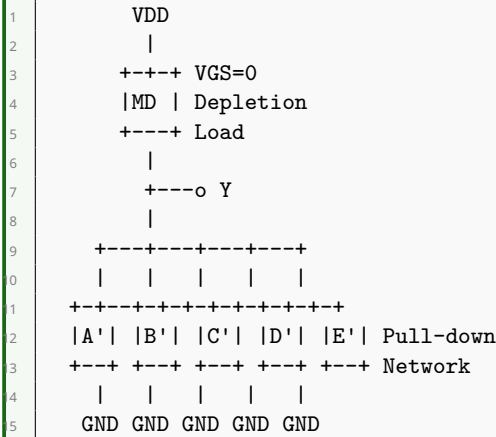
“CMOS માં ઝીરો સ્ટેટિક પાવર અને કુલ સ્વિંગ”

## પ્રશ્ન 3(અ) [3 ગુણ]

ડિપ્લીશન લોડનો ઉપયોગ કરીને  $Y = (A\square + B\square)C\square + D\square + E\square$  અમલમાં મૂકો.

## જવાબ

લોજિક સરળીકરણ:  $Y = (A + B)C + D + E = AC + BC + D + E$   
ડિપ્લીશન લોડ અમલીકરણ:



પુલ-ડાઉન નેટવર્ક:

- સીરીઝ:  $AC$  પાથ અને  $BC$  પાથ
- પેરેલલ: બધા પાથ પેરેલલમાં જોડાયેલા
- અમલીકરણ: યોગ્ય ટ્રાન્ઝિસ્ટર સાઇઝિંગ જરૂરી

## મેમરી ટ્રીક

“ડિપ્લીશન લોડ પેરેલલ પુલ-ડાઉન પાથ સાથે”

## પ્રશ્ન 3(બ) [4 ગુણ]

FPGA પર ટૂંકી નોંધ લખો.

## જવાબ

FPGA વ્યાખ્યા: ફીલ્ડ પ્રોગ્રામેબલ ગેટ એરે - રીકન્ફિગરેબલ ઇન્ટિગ્રેટેડ સર્કિટ.  
આર્કિટેક્ચર ઘટકો:

ઘટક	કાર્ય
CLB	કન્ફિગરેબલ લોજિક બ્લોક
IOB	ઇનપુટ/આઉટપુટ બ્લોક
ઇન્ટરકનેક્ટ	રાઉટિંગ રિસોર્સ
સ્વિચ મેટ્રિક્સ	કનેક્શન પોઇન્ટ્સ

પ્રોગ્રામિંગ ટેકનોલોજીઝ:

- SRAM-આધારિત: વોલેટાઇલ, ઝડપી રીકન્ફિગરેશન
- એન્ટીફ્યુઝ: નોન-વોલેટાઇલ, એક વખતનું પ્રોગ્રામેબલ
- ફ્લેશ-આધારિત: નોન-વોલેટાઇલ, રીપ્રોગ્રામેબલ

એપ્લિકેશન્સ:

- પ્રોટોટાઇપિંગ: ડિજિટલ સિસ્ટમ ડેવલપમેન્ટ
- DSP: સિગ્નલ પ્રોસેસિંગ એપ્લિકેશન્સ
- કંટ્રોલ સિસ્ટમ્સ: ઇન્ડસ્ટ્રિયલ ઓટોમેશન
- કોમ્યુનિકેશન્સ: પ્રોટોકોલ અમલીકરણ

ASIC સામે ફાયદા:

- લવચીકતા: રીકન્ફિગરેબલ ડિઝાઇન
- ટાઇમ-ટુ-માર્કેટ: ઝડપી વિકાસ
- કોસ્ટ: નાના વોલ્યુમ માટે ઓછો
- જોખમ: ઘટેલો ડિઝાઇન જોખમ



## મેમરી ટ્રીક

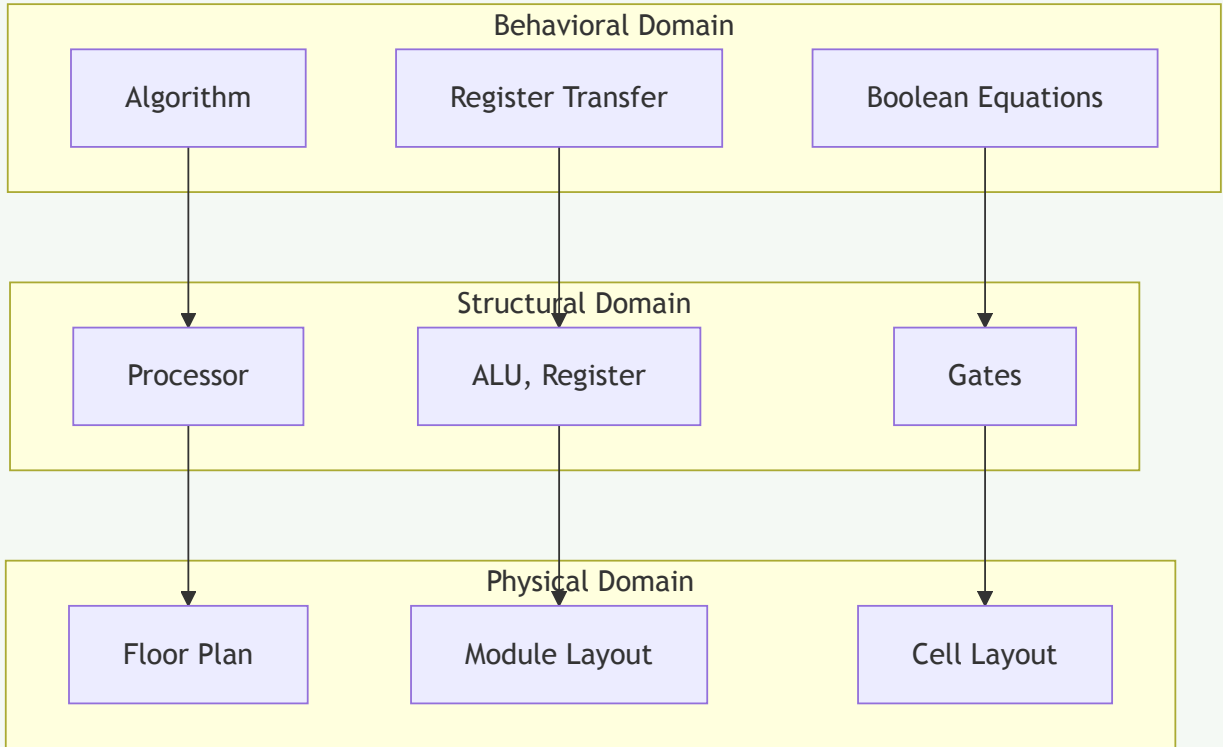
"FPGA - લવચીક પ્રોગ્રામિંગ ફાયદા આપે છે"

## પ્રશ્ન 3(ક) [7 ગુણ]

Y ચાર્ટ ડિઝાઇન ફલો દોરો અને સમજાવો.

### જવાબ

Y-ચાર્ટ ડાયાગ્રામ:



ડિઝાઇન ડોમેઇન-સ:

ડોમેઇન	લેવલ	વર્ણન
બિહેવિયરલ	એલ્ગોરિધમ $\rightarrow RT \rightarrow$	સિસ્ટમ શું કરે છે
સ્ટ્રક્ચરલ	પ્રોસેસર $\rightarrow ALU \rightarrow$	સિસ્ટમ કેવી રીતે બનાવેલ છે
ફિઝિકલ	ફ્લોર પ્લાન $\rightarrow \rightarrow$	ભૌતિક અમલીકરણ

ડિઝાઇન ફલો પ્રક્રિયા:

- ટોપ-ડાઉન: બિહેવિયરલથી શરૂ કરી ફિઝિકલ તરફ જાયો
- બોટમ-અપ: ઘટકોથી ઉપરની તરફ બનાવો
- મિક્સ્ડ એપ્રોચ: બંનેની સંયુક્ત પદ્ધતિ

એબ્સ્ટ્રેક્શન લેવલ્સ:

- સિસ્ટમ લેવલ: સૌથી વધુ એબ્સ્ટ્રેક્શન
- RT લેવલ: રજિસ્ટર ટ્રાન્સફર ઓપરેશન્સ
- ગેટ લેવલ: બુલિયન લોજિક અમલીકરણ
- લેઆઉટ લેવલ: ભૌતિક જ્યોમેટ્રી

ડિઝાઇન વેરિફિકેશન:

- હોરિઝોન્ટલ: સમાન લેવલે ડોમેઇન-સ વચ્ચે
- વર્ટિકલ: સમાન ડોમેઇનમાં લેવલ્સ વચ્ચે

## મેમરી ટ્રીક

"Y-ચાર્ટ: બિહેવિયરલ, સ્ટ્રક્ચરલ, ફિઝિકલ - BSP ડોમેઇન-સ"

### પ્રશ્ન 3(અ) OR [3 ગુણ]

ડિપ્લીશન લોડનો ઉપયોગ કરીને NOR2 ગેટ સમજાવો.

જવાબ

ડિપ્લીશન લોડ NOR2 સર્કિટ:

```

1      VDD
2      |
3      +---+ VGS=0
4      |MD | Depletion
5      +---+ Load
6      |
7      +---o Y
8      |
9      +---+---+
10     |         |
11     +---+   +---+
12     |nA |   |nB | NMOS
13     +---+   +---+ (Parallel)
14     |         |
15     GND      GND

```

ટ્રુથ ટેબલ:

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

ઓપરેશન:

- બંને ઇનપુટ લો: બંને NMOS OFF, Y = VDD
- કોઈપણ ઇનપુટ હાઇ: સંબંધિત NMOS ON, Y = VOL
- લોડ ટ્રાન્ઝિસ્ટર: પુલ-અપ કરંટ પૂરો પાડે છે

મેમરી ટ્રીક

“ડિપ્લીશન સાથે NOR - પેરેલલ NMOS પુલ-ડાઉન”

### પ્રશ્ન 3(બ) OR [4 ગુણ]

કુલ કસ્ટમ અને સેમી કસ્ટમ ડિઝાઇન શૈલીઓની તુલના કરો.

જવાબ

તુલના ટેબલ:

પેરામીટર	કુલ કસ્ટમ	સેમી કસ્ટમ
ડિઝાઇન ટાઇમ	લાંબો (6-18 મહિના)	ટૂંકો (2-6 મહિના)
પર્ફોર્મન્સ	શ્રેષ્ઠ	સારું
એરિયા	લઘુત્તમ	મધ્યમ
પાવર	ઓપ્ટિમાઇઝ્ડ	સ્વીકાર્ય
કોસ્ટ	હાઇ NRE	લોઅર NRE
લવચીકતા	મહત્તમ	મર્યાદિત
જોખમ	વધુ	ઓછું

#### કુલ કસ્ટમ લાક્ષણિકતાઓ:

- દરેક ટ્રાન્ઝિસ્ટર: મેન્યુઅલી ડિઝાઇન અને પ્લેસ કરેલું
- લેઆઉટ ઓપ્ટિમાઇઝેશન: મહત્તમ ડેન્સિટી હાંસલ
- એપ્લિકેશન્સ: હાઇ-વોલ્યુમ, પર્ફોર્મન્સ-ક્રિટિકલ

#### સેમી કસ્ટમ પ્રકારો:

- ગેટ એરે: પૂર્વ-વ્યાખ્યાયિત ટ્રાન્ઝિસ્ટર એરે
- સ્ટેન્ડર્ડ સેલ: પૂર્વ-ડિઝાઇન કરેલા સેલ્સની લાઇબ્રેરી
- FPGA: ફીલ્ડ પ્રોગ્રામેબલ લોજિક

#### ડિઝાઇન ફ્લો તુલના:

- કુલ કસ્ટમ: સ્પેસિફિકેશન → → →
- સેમી કસ્ટમ: સ્પેસિફિકેશન → HDL → → &

#### મેમરી ટ્રીક

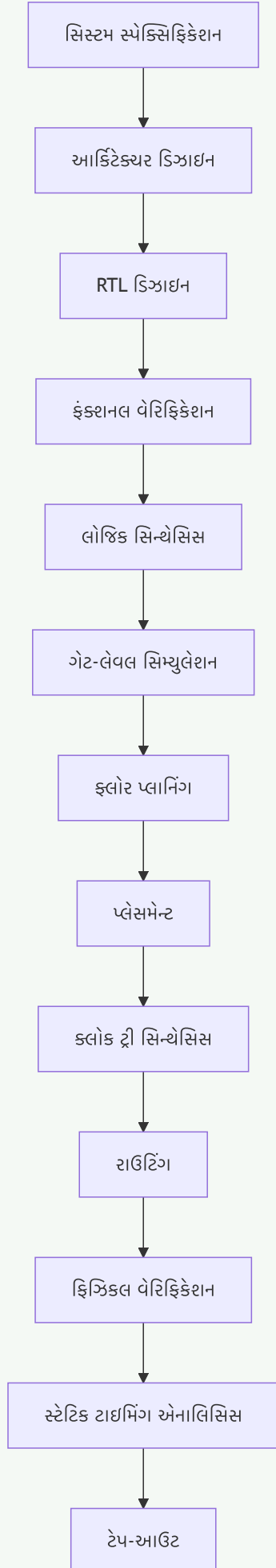
“કુલ કસ્ટમ - મહત્તમ નિયંત્રણ, સેમી કસ્ટમ - સ્પીડ સમજૂતી”

### પ્રશ્ન 3(ક) OR [7 ગુણ]

ASIC ડિઝાઇન ફ્લો વિગતવાર દોરો અને સમજાવો.

#### જવાબ

ASIC ડિઝાઇન ફ્લો:



### ડિઝાઇન સ્ટેજો:

સ્ટેજ	વર્ણન	ટૂલ્સ/પદ્ધતિઓ
RTL ડિઝાઇન	હાર્ડવેર વર્ણન	વેરિલોગ/VHDL
સિન્થેસિસ	RTL ને ગેટ્સમાં કન્વર્ટ	લોજિક સિન્થેસિસ ટૂલ્સ
ફ્લોર પ્લાનિંગ	ચિપ એરિયા વિતરણ	ફ્લોર પ્લાનિંગ ટૂલ્સ
પ્લેસમેન્ટ	ગેટ્સ/બ્લોક્સ સ્થાન	પ્લેસમેન્ટ એલ્ગોરિધમ
રાઉટિંગ	પ્લેસ કરેલા એલિમેન્ટ્સ જોડો	રાઉટિંગ એલ્ગોરિધમ

### વેરિફિકેશન સ્ટેપ્સ:

- ફંક્શનલ: RTL સિમ્યુલેશન અને વેરિફિકેશન
- ગેટ-લેવલ: પોસ્ટ-સિન્થેસિસ સિમ્યુલેશન
- ફિઝિકલ: DRC, LVS, એન્ટેના ચેકસ
- ટાઇમિંગ: સેટઅપ/હોલ્ડ વાયોલેશન માટે STA

### ડિઝાઇન કન્સ્ટ્રેઇન્ટ્સ:

- ટાઇમિંગ: કલોક ફ્રીક્વન્સી જરૂરિયાતો
- એરિયા: સિલિકોન એરિયા મર્યાદાઓ
- પાવર: પાવર વપરાશ લક્ષ્યો
- ટેસ્ટ: ટેસ્ટેબિલિટી માટે ડિઝાઇન

### સાઇન-ઓફ ચેકસ:

- DRC: ડિઝાઇન રૂલ ચેક
- LVS: લેઆઉટ વર્સીસ સ્કીમેટિક
- STA: સ્ટેટિક ટાઇમિંગ એનાલિસિસ
- પાવર: પાવર ઇન્ટેગ્રિટી એનાલિસિસ

### મેમરી ટ્રીક

“ASIC ફ્લો: RTL → → →”

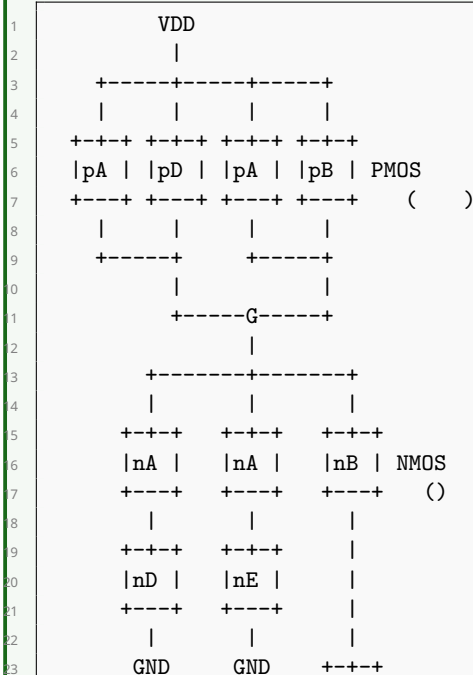
### પ્રશ્ન 4(અ) [3 ગુણ]

CMOS સાથે લોજિક ફંક્શન  $G = (A(D+E)+BC)$  અમલમાં મૂકો

### જવાબ

લોજિક વિશ્લેષણ:  $G = (A(D+E)+BC) = (AD+AE+BC)$

CMOS અમલીકરણ:





નેટવર્ક કન્ફિગરેશન:

- PMOS: કોમ્પ્લિમેન્ટનું સીરીઝ અમલીકરણ
- NMOS: મૂળ ફંક્શનનું પેરેલલ અમલીકરણ

### મેમરી ટ્રીક

“કોમ્પ્લેક્સ CMOS - PMOS સીરીઝ, NMOS પેરેલલ”

## પ્રશ્ન 4(બ) [4 ગુણ]

3 બિટ પેરિટી ચેકર માટે વેરિલોગ કોડ લખો.

### જવાબ

વેરિલોગ કોડ:

```

1 module parity_checker_3bit(
2     input [2:0] data_in,
3     output parity_even,
4     output parity_odd
5 );
6
7 //
8 assign parity_even = ^data_in;
9
10 //
11 assign parity_odd = ~(^data_in);
12
13 //
14 /*
15 assign parity_even = data_in[0] ^ data_in[1] ^ data_in[2];
16 assign parity_odd = ~(data_in[0] ^ data_in[1] ^ data_in[2]);
17 */
18
19 endmodule

```

ટ્રુથ ટેબલ:

ઇનપુટ [2:0]	1 નું સંખ્યા	ઇવન પેરિટી	ઓડ પેરિટી
000	0	0	1
001	1	1	0
010	1	1	0
011	2	0	1
100	1	1	0
101	2	0	1
110	2	0	1
111	3	1	0

મુખ્ય લક્ષણો:

- XOR રિડક્શન:  $\sim data\_in$  ઇવન પેરિટી આપે છે
- કોમ્પ્લિમેન્ટ:  $\sim(\sim data\_in)$  ઓડ પેરિટી આપે છે

### મેમરી ટ્રીક

“પેરિટી ચેક: બધા બિટ્સનું XOR”

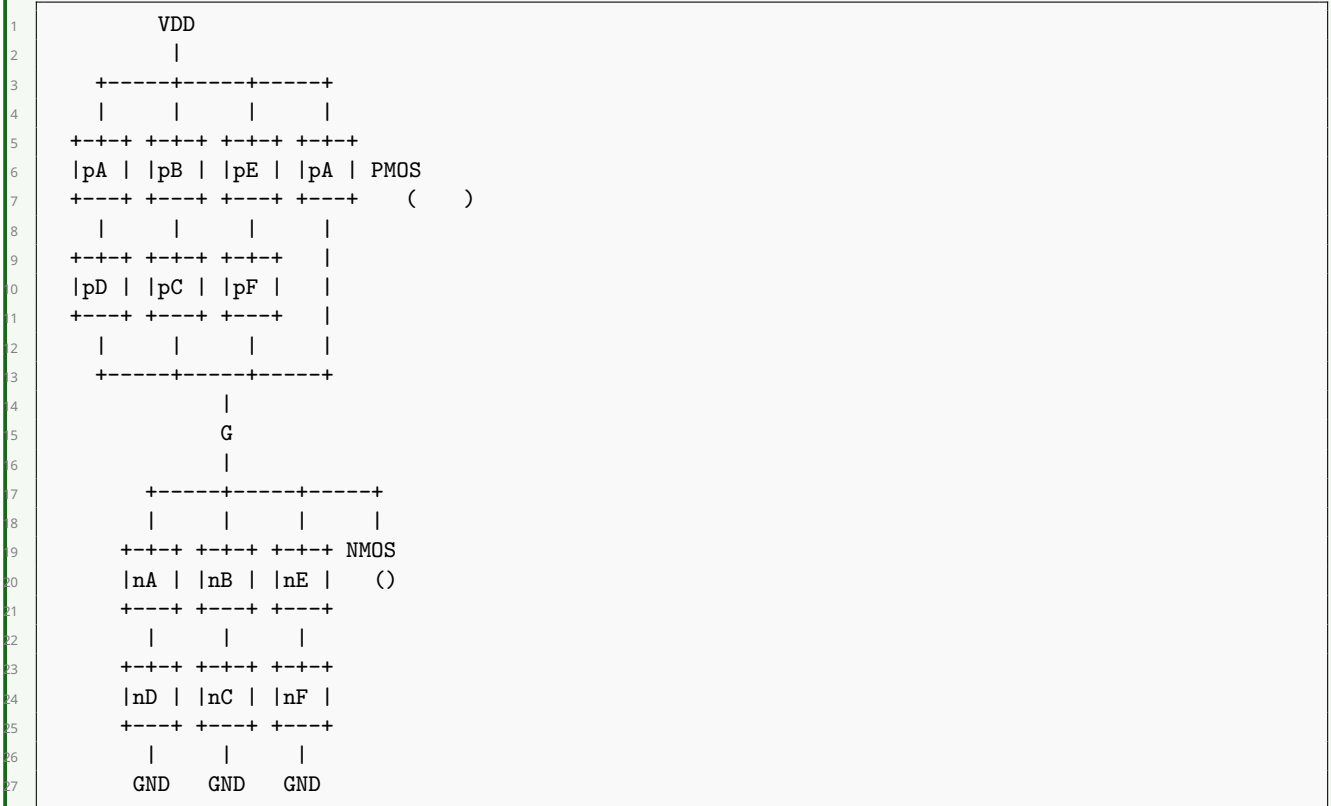
### પ્રશ્ન 4(ક) [7 ગુણ]

અમલીકરણ: 1)  $G = (AD + BC + EF)$  CMOS નો ઉપયોગ કરીને [3 ગુણ] 2)  $Y' = (ABCD + EF(G+H) + J)$  CMOS નો ઉપયોગ કરીને [4 ગુણ]

#### જવાબ

ભાગ 1:  $G = (AD + BC + EF)$  [3 ગુણ]

CMOS સર્કિટ:



ભાગ 2:  $Y' = (ABCD + EF(G+H) + J)$  [4 ગુણ]

આ એક કોમ્પ્લેક્સ ફંક્શન છે જેને બહુવિધ સ્ટેજની જરૂર છે:

સ્ટેજ 1:  $(G+H)$  અમલીકરણ સ્ટેજ 2:  $EF(G+H)$  અમલીકરણ સ્ટેજ 3: બધા ટર્મ્સ કોમ્બાઇન આવા કોમ્પ્લેક્સ ફંક્શન માટે ટ્રાન્સમિશન ગેટ્સ અને બહુવિધ સ્ટેજનો ઉપયોગ વધુ પ્રેક્ટિકલ છે.

#### મેમરી ટ્રીક

“કોમ્પ્લેક્સ ફંક્શન-સને સ્ટેજ્સ અમલીકરણ જોઈએ”

### પ્રશ્ન 4(અ) OR [3 ગુણ]

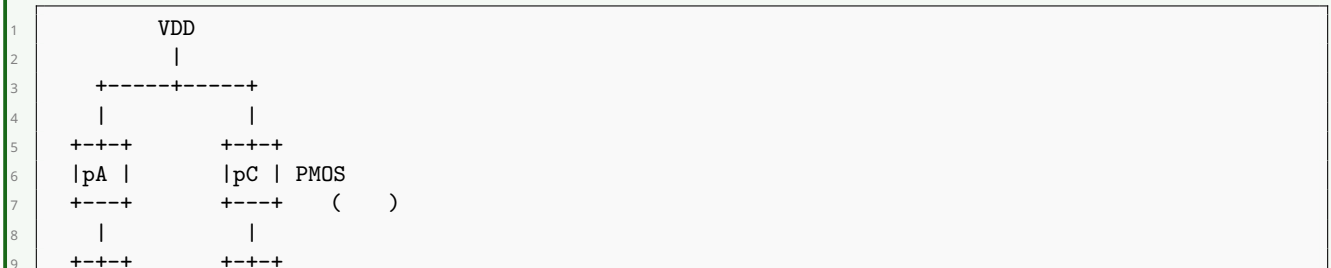
ઉદાહરણ સાથે AOI લોજિક સમજાવો.

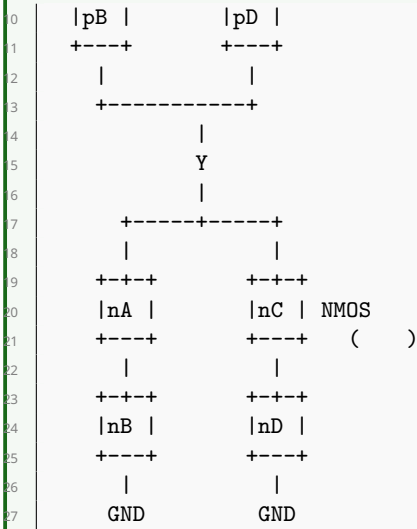
#### જવાબ

AOI વ્યાખ્યા: AND-OR-Invert લોજિક આ પ્રકારના ફંક્શન-સ અમલીકરણ કરે છે:  $Y = (AB + CD + \dots)$

ઉદાહરણ:  $Y = (AB + CD)$

AOI અમલીકરણ:





#### ફાયદા:

- સિંગલ સ્ટેજ: ડાયરેક્ટ અમલીકરણ
- ઝડપી: બહુવિધ લેવલ્સ દ્વારા પ્રોપેગેશન નહીં
- એરિયા એફિશિયન્ટ: અલગ ગેટ્સ કરતાં ઓછા ટ્રાન્ઝિસ્ટર

#### એપ્લિકેશન્સ:

- કોમ્પ્લેક્સ ગેટ્સ: મલ્ટિ-ઇનપુટ ફંક્શન્સ
- સ્પીડ-ક્રિટિકલ પાથ: ઘટેલી ડીલે

#### મેમરી ટ્રીક

“AOI - AND-OR-Invert એક સ્ટેજમાં”

### પ્રશ્ન 4(બ) OR [4 ગુણ]

4-બિટ સીરિયલ IN પેરેલલ આઉટ શિફ્ટ રજિસ્ટર માટે વેરિલોગ કોડ લખો.

#### જવાબ

##### વેરિલોગ કોડ:

```

1 module sipo_4bit(
2     input clk,
3     input reset,
4     input serial_in,
5     output reg [3:0] parallel_out
6 );
7
8 always @(posedge clk or posedge reset) begin
9     if (reset) begin
10         parallel_out <= 4'b0000;
11     end else begin
12         //          LSB
13         parallel_out <= {parallel_out[2:0], serial_in};
14     end
15 end
16
17 endmodule

```

##### ટેસ્ટબેન્ચ ઉદાહરણ:

```

1 module tb_sipo_4bit;
2     reg clk, reset, serial_in;
3     wire [3:0] parallel_out;
4
5     sipo_4bit dut(.clk(clk), .reset(reset),
6                 .serial_in(serial_in),

```



```

7         .parallel_out(parallel_out));
8
9     initial begin
10         clk = 0;
11         forever #5 clk = ~clk;
12     end
13
14     initial begin
15         reset = 1; serial_in = 0;
16         #10 reset = 0;
17         #10 serial_in = 1; // LSB
18         #10 serial_in = 0;
19         #10 serial_in = 1;
20         #10 serial_in = 1; // MSB
21         #20 $finish;
22     end
23 endmodule

```

ઓપરેશન ટાઇમલાઇન:

ક્લોક	Serial_in	Parallel_out
1	1	0001
2	0	0010
3	1	0101
4	1	1011

### મેમરી ટ્રીક

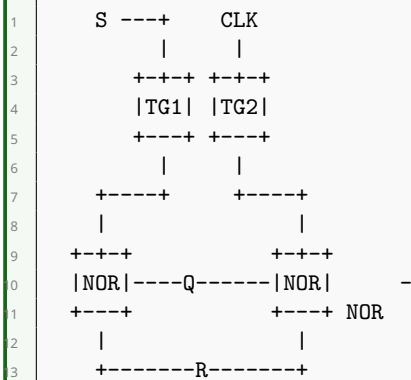
“SIPO - સીરિયલ ઇન, પેરેલલ આઉટ શિફ્ટ લેફ્ટ સાથે”

### પ્રશ્ન 4(ક) OR [7 ગુણ]

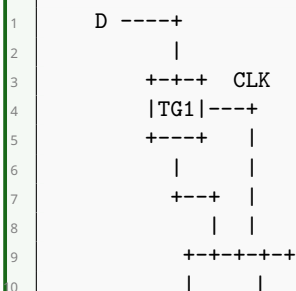
CMOS નો ઉપયોગ કરીને ક્લોક્ડ NOR2 SR લેચ અને D-લેચ અમલીકરણ કરો.

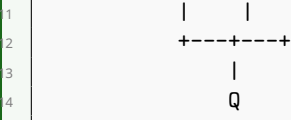
#### જવાબ

##### ક્લોક્ડ NOR2 SR લેચ:

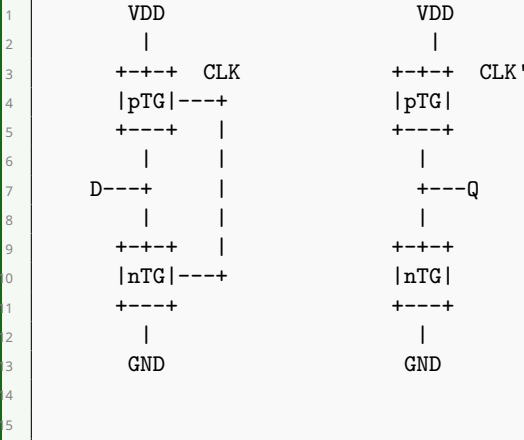


##### D-લેચ અમલીકરણ:





### CMOS D-લેચ સર્કિટ:



### ઓપરેશન:

- CLK = 1: માસ્ટર ટ્રાન્સપેરન્ટ, સ્લેવ હોલ્ડ
- CLK = 0: માસ્ટર હોલ્ડ, સ્લેવ ટ્રાન્સપેરન્ટ
- ડેટા ટ્રાન્સફર: કલોક એજ પર

### SR લેચ માટે ટ્રુથ ટેબલ:

S	R	CLK	Q	Q'
0	0	1	હોલ્ડ	હોલ્ડ
0	1	1	0	1
1	0	1	1	0
1	1	1	અવૈધ	અવૈધ

### મેમરી ટ્રીક

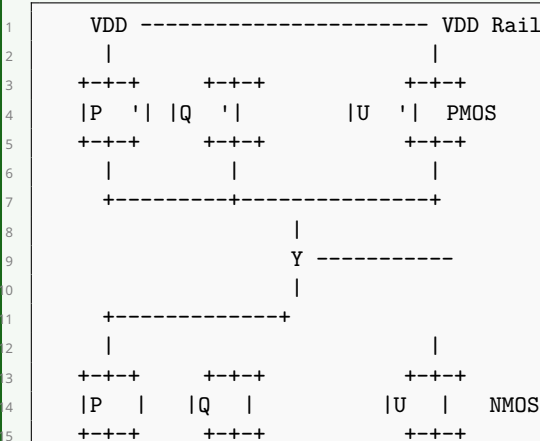
“કલોકડ લેચ ટાઇમિંગ કંટ્રોલ માટે ટ્રાન્સમિશન ગેટ ઉપયોગ કરે છે”

### પ્રશ્ન 5(અ) [3 ગુણ]

ચુલર પાથ એપ્રોચને ધ્યાનમાં લેતા CMOS નો ઉપયોગ કરીને  $Y = (PQ + U)'$  માટે સ્ટિક ડાયાગ્રામ દોરો.

### જવાબ

લોજિક વિશ્લેષણ:  $Y = (PQ + U)'$  માટે જરૂરી છે PMOS:  $(PQ)' \cdot U' = (P' + Q') \cdot U'$  NMOS:  $PQ + U$   
સ્ટિક ડાયાગ્રામ:



```

6      |           |           |
7      +-----+           |
8      |           |           |
9      GND -----+-----+--- GND Rail
10
11 :
12 - : P-diffusion (PMOS)
13 - : N-diffusion (NMOS)
14 - : Polysilicon  ( )
15 - :

```

ચુલર પાથ:

1. PMOS:  $P' \rightarrow Q'(), U'$
1. NMOS:  $P \rightarrow Q(), U$
1. ઓપ્ટિમલ રાઉટિંગ: ક્રોસઓવર મિનિમાઇઝ કરે છે

લેઆઉટ વિચારણાઓ:

- ડિફ્યુઝન બ્લેક્સ: બહેતર પર્ફોર્મન્સ માટે મિનિમાઇઝ
- કોન્ટેક્ટ પ્લેસમેન્ટ: યોગ્ય VDD/GND કનેક્શન્સ
- મેટલ રાઉટિંગ: DRC વાયોલેશન ટાળો

### મેમરી ટ્રીક

“સ્ટિક ડાયાગ્રામ ચુલર પાથ ઓપ્ટિમાઇઝેશન સાથે ફિઝિકલ લેઆઉટ દર્શાવે છે”

## પ્રશ્ન 5(બ) [4 ગુણ]

વેરિલોગનો ઉપયોગ કરીને  $8 \times 1$

જવાબ

વેરિલોગ કોડ:

```

1 module mux_8x1(
2     input [7:0] data_in,    // 8
3     input [2:0] select,     // 3-
4     output reg data_out    //
5 );
6
7 always @(*) begin
8     case (select)
9         3'b000: data_out = data_in[0];
10        3'b001: data_out = data_in[1];
11        3'b010: data_out = data_in[2];
12        3'b011: data_out = data_in[3];
13        3'b100: data_out = data_in[4];
14        3'b101: data_out = data_in[5];
15        3'b110: data_out = data_in[6];
16        3'b111: data_out = data_in[7];
17        default: data_out = 1'b0;
18    endcase
19 end
20
21 endmodule

```

વૈકલ્પિક અમલીકરણ:

```

1 module mux_8x1_dataflow(
2     input [7:0] data_in,
3     input [2:0] select,
4     output data_out
5 );
6
7 assign data_out = data_in[select];
8
9 endmodule

```

ટ્રુથ ટેબલ:

Select[2:0]	આઉટપુટ
000	data_in[0]
001	data_in[1]
010	data_in[2]
011	data_in[3]
100	data_in[4]
101	data_in[5]
110	data_in[6]
111	data_in[7]

ટેસ્ટબેન્ચ:

```

1 module tb_mux_8x1;
2     reg [7:0] data_in;
3     reg [2:0] select;
4     wire data_out;
5
6     mux_8x1 dut(.data_in(data_in), .select(select), .data_out(data_out));
7
8     initial begin
9         data_in = 8'b10110100;
10        for (int i = 0; i < 8; i++) begin
11            select = i;
12            #10;
13            $display("Select=%d, Output=%b", select, data_out);
14        end
15    end
16 endmodule

```

મેમરી ટ્રીક

“MUX સિલેક્ટ લાઇન-સના આધારે ઘણા ઇનપુટ્સમાંથી એક પસંદ કરે છે”

## પ્રશ્ન 5(ક) [7 ગુણ]

વેરિલોગમાં બિહેવિયરલ મોડેલિંગ સ્ટાઇલનો ઉપયોગ કરીને કુલ એડર અમલમાં મૂકો.

જવાબ

વેરિલોગ કોડ:

```

1 module full_adder_behavioral(
2     input A,
3     input B,
4     input Cin,
5     output reg Sum,
6     output reg Cout
7 );
8
9 //          always
10 always @(*) begin
11     case ({A, B, Cin})
12         3'b000: begin Sum = 1'b0; Cout = 1'b0; end
13         3'b001: begin Sum = 1'b1; Cout = 1'b0; end
14         3'b010: begin Sum = 1'b1; Cout = 1'b0; end
15         3'b011: begin Sum = 1'b0; Cout = 1'b1; end
16         3'b100: begin Sum = 1'b1; Cout = 1'b0; end
17         3'b101: begin Sum = 1'b0; Cout = 1'b1; end
18         3'b110: begin Sum = 1'b0; Cout = 1'b1; end

```

```

9      3'b111: begin Sum = 1'b1; Cout = 1'b1; end
10      default: begin Sum = 1'b0; Cout = 1'b0; end
11  endcase
12 end
13
14 endmodule

```

વૈકલ્પિક બિહેવિયરલ સ્ટાઇલ:

```

1 module full_adder_behavioral_alt(
2     input A, B, Cin,
3     output reg Sum, Cout
4 );
5
6 always @(*) begin
7     {Cout, Sum} = A + B + Cin;
8 end
9
10 endmodule

```

ટ્રુથ 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

ટેસ્ટબેન્ચ:

```

1 module tb_full_adder;
2     reg A, B, Cin;
3     wire Sum, Cout;
4
5     full_adder_behavioral dut(.A(A), .B(B), .Cin(Cin),
6                               .Sum(Sum), .Cout(Cout));
7
8     initial begin
9         $monitor("A=%b
10         B=%b Cin=%b | Sum=%b Cout=%b",
11                 A, B, Cin, Sum, Cout);
12
13         {A, B, Cin} = 3'b000; #10;
14         {A, B, Cin} = 3'b001; #10;
15         {A, B, Cin} = 3'b010; #10;
16         {A, B, Cin} = 3'b011; #10;
17         {A, B, Cin} = 3'b100; #10;
18         {A, B, Cin} = 3'b101; #10;
19         {A, B, Cin} = 3'b110; #10;
20         {A, B, Cin} = 3'b111; #10;
21
22         $finish;
23     end
24 endmodule

```

બિહેવિયરલ લક્ષણો:

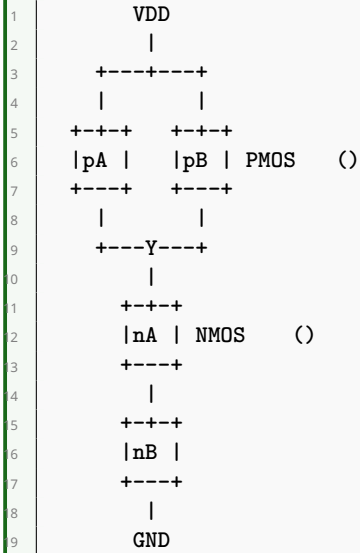
- Always બ્લોક: સ્ટ્રક્ચર નહીં, બિહેવિયર વર્ણવે છે
- Case સ્ટેટમેન્ટ: ટ્રુથ ટેબલ અમલીકરણ
- ઓટોમેટિક સિન્થેસિસ: ટૂલ્સ ઓપ્ટિમાઇઝડ સર્કિટ જનરેટ કરે છે

## પ્રશ્ન 5(અ) OR [3 ગુણ]

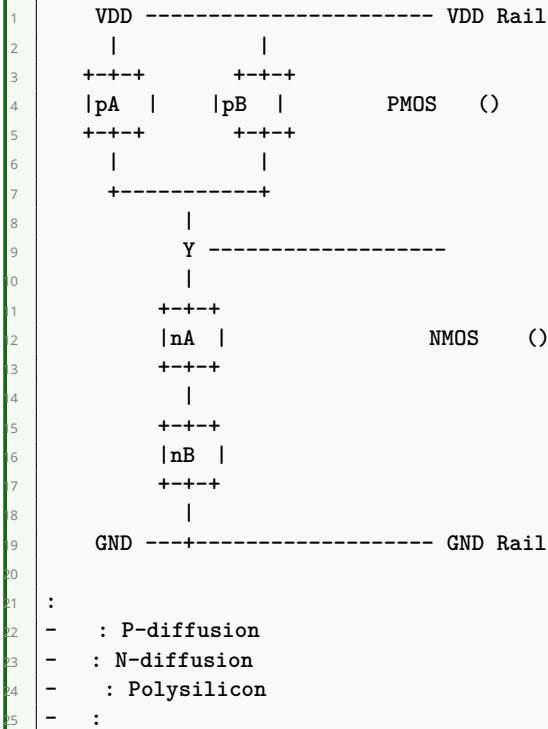
NOR2 ગેટ CMOS સર્કિટને તેના સ્ટિક ડાયાગ્રામ સાથે અમલમાં મૂકો.

### જવાબ

#### CMOS NOR2 સર્કિટ:



#### સ્ટિક ડાયાગ્રામ:



#### લેઆઉટ નિયમો:

- PMOS: પુલ-અપ માટે પેરેલલ કનેક્શન
- NMOS: પુલ-ડાઉન માટે સીરીઝ કનેક્શન
- કોન્ટેક્ટ્સ: યોગ્ય VDD/GND કનેક્શન્સ
- સ્પેસિંગ: લઘુત્તમ ડિઝાઇન નિયમો પૂરા કરવા

## પ્રશ્ન 5(બ) OR [4 ગુણ]

વેરિલોગનો ઉપયોગ કરીને 4 બિટ અપ કાઉન્ટર અમલમાં મૂકો

## જવાબ

વેરિલોગ કોડ:

```

1 module counter_4bit_up(
2     input clk,
3     input reset,
4     input enable,
5     output reg [3:0] count
6 );
7
8 always @(posedge clk or posedge reset) begin
9     if (reset) begin
10        count <= 4'b0000;
11    end else if (enable) begin
12        if (count == 4'b1111) begin
13            count <= 4'b0000; //
14        end else begin
15            count <= count + 1;
16        end
17    end
18    // enable
19 end
20
21 endmodule

```

ઓવરફ્લો સાથે વિસ્તૃત વર્ઝન:

```

1 module counter_4bit_enhanced(
2     input clk,
3     input reset,
4     input enable,
5     output reg [3:0] count,
6     output overflow
7 );
8
9 always @(posedge clk or posedge reset) begin
10    if (reset) begin
11        count <= 4'b0000;
12    end else if (enable) begin
13        count <= count + 1; //
14    end
15 end
16
17 assign overflow = (count == 4'b1111) & enable;
18
19 endmodule

```

કાઉન્ટ સિક્વન્સ:

ક્લોક	Count[3:0]	દર્શાવ
1	0000	0
2	0001	1
3	0010	2
...	...	...
15	1110	14
16	1111	15
17	0000	0 (રોલઓવર)

### ટેસ્ટબેન્ચ:

```
1 module tb_counter_4bit;
2   reg clk, reset, enable;
3   wire [3:0] count;
4
5   counter_4bit_up dut(.clk(clk), .reset(reset),
6                       .enable(enable), .count(count));
7
8   //
9   initial begin
10    clk = 0;
11    forever #5 clk = ~clk;
12  end
13
14  //
15  initial begin
16    reset = 1; enable = 0;
17    #10 reset = 0; enable = 1;
18    #200 enable = 0; //
19    #20 enable = 1; //
20    #100 $finish;
21  end
22
23  //
24  always @(posedge clk) begin
25    $display("Time=%t Count=%d", $time, count);
26  end
27 endmodule
```

### મેમરી ટ્રીક

“અપ કાઉન્ટર: ઇનેબલ હોય ત્યારે દરેક કલોક પર વધારો”

### પ્રશ્ન 5(ક) OR [7 ગુણ]

વેરિલોગમાં બિહેવિયરલ મોડેલિંગ સ્ટાઇલનો ઉપયોગ કરીને 3:8 ડીકોડર અમલમાં મૂકો.

### જવાબ

#### વેરિલોગ કોડ:

```
1 module decoder_3x8_behavioral(
2   input [2:0] address, // 3-
3   input enable, //
4   output reg [7:0] decode_out // 8-
5 );
6
7 always @(*) begin
8   if (enable) begin
9     case (address)
10      3'b000: decode_out = 8'b00000001; // Y0
11      3'b001: decode_out = 8'b00000010; // Y1
12      3'b010: decode_out = 8'b00000100; // Y2
13      3'b011: decode_out = 8'b00001000; // Y3
14      3'b100: decode_out = 8'b00010000; // Y4
15      3'b101: decode_out = 8'b00100000; // Y5
16      3'b110: decode_out = 8'b01000000; // Y6
17      3'b111: decode_out = 8'b10000000; // Y7
18      default: decode_out = 8'b00000000;
19    endcase
20  end else begin
21    decode_out = 8'b00000000; //
22  end
23 end
```



```

23 end
24
25 endmodule

```

### વૈકલ્પિક અમલીકરણ:

```

1 module decoder_3x8_shift(
2     input [2:0] address,
3     input enable,
4     output [7:0] decode_out
5 );
6
7 assign decode_out = enable ? (8'b00000001 << address) : 8'b00000000;
8
9 endmodule

```

### ટ્રુથ ટેબલ:

Enable	Address[2:0]	decode_out[7:0]
0	XXX	00000000
1	000	00000001
1	001	00000010
1	010	00000100
1	011	00001000
1	100	00010000
1	101	00100000
1	110	01000000
1	111	10000000

### ટેસ્ટબેન્ચ:

```
1 module tb_decoder_3x8;
2     reg [2:0] address;
3     reg enable;
4     wire [7:0] decode_out;
5
6     decoder_3x8_behavioral dut(.address(address), .enable(enable),
7                               .decode_out(decode_out));
8
9     initial begin
10         $monitor("Enable=%b Address=%b | Output=%b",
11                 enable, address, decode_out);
12
13         // Enable = 0
14         enable = 0;
15         for (int i = 0; i < 8; i++) begin
16             address = i;
17             #10;
18         end
19
20         // Enable = 1
21         enable = 1;
22         for (int i = 0; i < 8; i++) begin
23             address = i;
24             #10;
25         end
26
27         $finish;
28     end
29 endmodule
```

### એપ્લિકેશન્સ:

- મેમોરી એડ્રેસિંગ: 8 મેમોરી લોકેશનમાંથી એક પસંદ કરો
- ડિવાઇસ સિલેક્શન: 8 પેરિફેરલ ડિવાઇસમાંથી એક ઇનેબલ કરો
- ડીમલ્ટિપ્લેક્સિંગ: સિંગલ ઇનપુટને પસંદ કરેલા આઉટપુટ પર રાઉટ કરો

### ડિઝાઇન લક્ષણો:

- વન-હોટ એન્કોડિંગ: એક સમયે ફક્ત એક આઉટપુટ હાઇ
- ઇનેબલ કંટ્રોલ: ગ્લોબલ ઇનેબલ/ડિસેબલ ફંક્શનાલિટી
- કુલ ડીકોડિંગ: બધા શક્ય ઇનપુટ કોમ્બિનેશન હેન્ડલ

### મેમરી ટ્રીક

“3:8 ડીકોડર - 3 ઇનપુટ્સ 8 આઉટપુટમાંથી 1 પસંદ કરે છે”