Introduction

10
G. Venkata Hanuman
Asst. Prof

Introduction to Ic technology :-

It is an electronic assembly maintain such a way that all components of the circuit are fabricated on a single container called chip (or) I.C.

* the technique to Increase no. of devices per chip is called level of Integration.

It mainly consists of

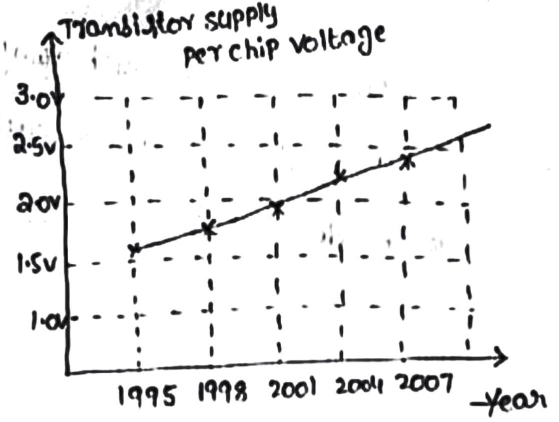
1. SSI → Small Scale Integration
2. MSI → medium scale Integration
3. LSI → large scale Integration
4. VLSI → very large scale Integration
5. ULSI → Ultra large scale Integration

Type of Ic technology	Device	Year	Applications
SSI	1-100	1960	Gates, op-Amps
MSI	100-1K	1965	Registers, filters
LSI	1K-10K	1970	A/D, D/A, microprocessors
VLSI	>10k	1975	memories, DSP kits

Advantages of IC's :-

* Low power consumption
* High speed of operation
* Small in size
* Reduce cost
* High reliability
* Eaisy replacement

Moore states that the transistor count double for every 18 months.



metal oxide semiconductor (mos transistor) :-

Principle :- A voltage is applied to the gate terminal, it controls the current in a conducting channel between source and drain.

Ex :- In NMOS, the majority carriers are electrons.

A +ve voltage applied on gate w.r.to substrate it enhances no. of electrons in the channel and Increases the conductivity of channel.

For Gate voltage is lessthan threshold voltage value., the channel is cut-off, thus causing a very low drain-to-source current.

Threshold voltage (V_t):- The voltage at which a mos device begins to conduct (turn ON).

Key representations

▥	→ metal
▨	→ Poly silicon
⠿	→ oxide

▨ → n-diffusion

▧ → P-diffusion

^ ^ ^ → p-substrate

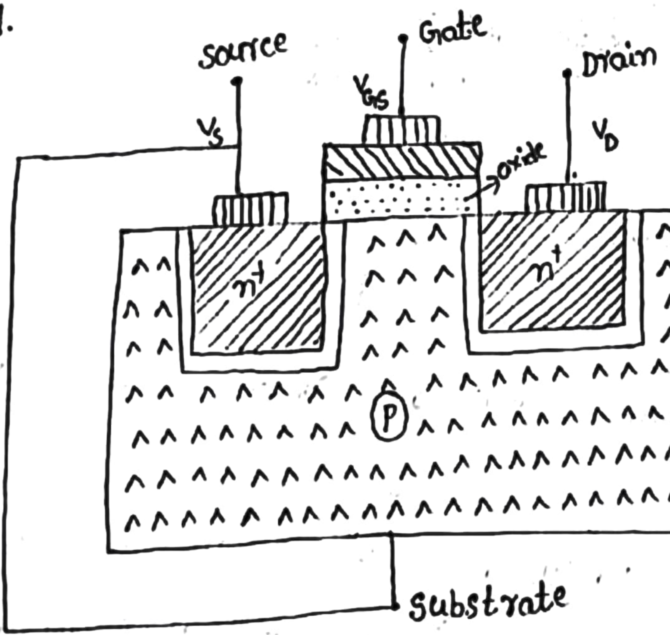
⁄ ⁄ ⁄ → n-substrate

□ → Depletion

Nmos transistor :-

1. Nmos enhancement mode transistor :-

It consists of moderately doped p-type silicon substrate into which two heavily doped n^+ regions, namely source and drain are diffused.



→ Between source and drain there is a narrow region of p-type substrate called channel which is covered at the top with SiO_2 (oxide) layer

→ over the SiO_2 layer, Poly silicon gate is deposited over the regions between drain and source.

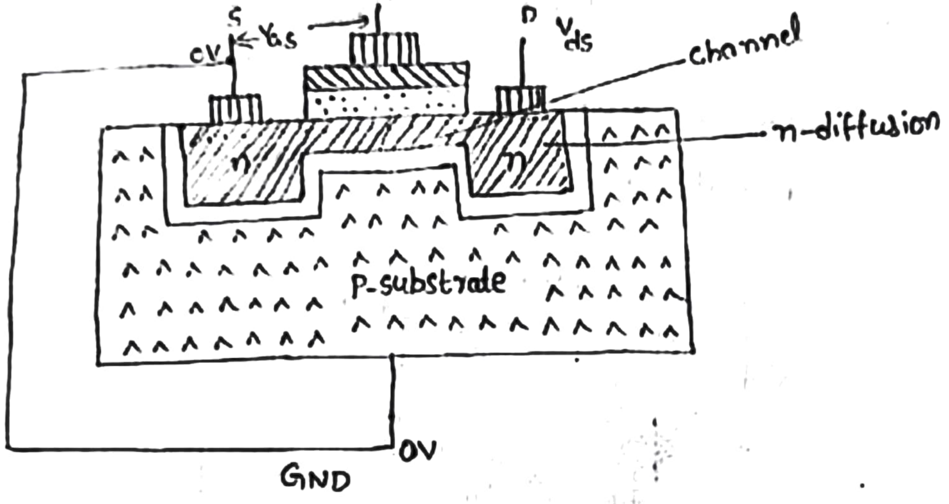
→ when gate is not biased, $V_D = V_S = V_{GS} = 0$, channel is not established hence it is not conducting.

→ when gate is connected to +ve voltage w.r.to source, then electric field is established between gate and substrate.

operation of the transistor in enhancement mode, mainly 3 different conditions are received

condition -1 :- $V_{ds} = 0$

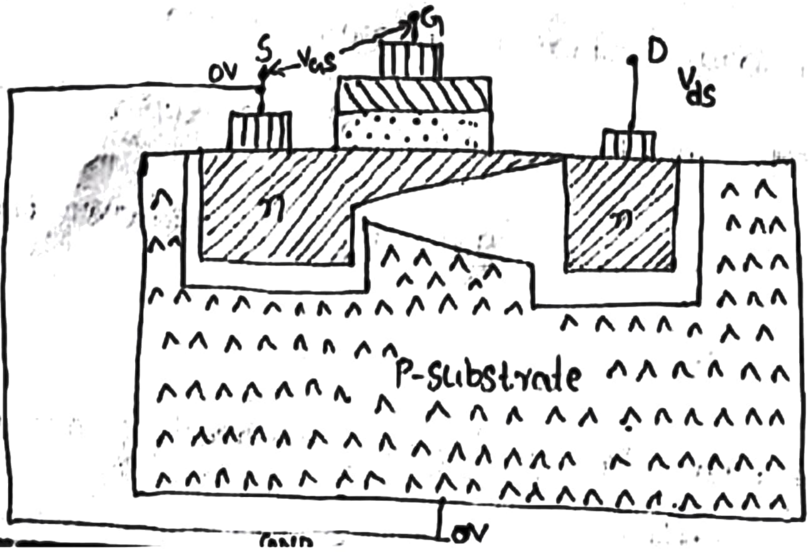
In this condition, a channel between drain and source is established but no current is flowing between source and drain i.e $V_{ds} = 0$



condition -2 :- $V_{ds} < V_{gs} - V_t$

when V_{ds} is applied between drain and source, current starts flowing through channel.

* The resistance of channel varies along its length this results in IR drop along the channel. The voltage is maximum at source end.



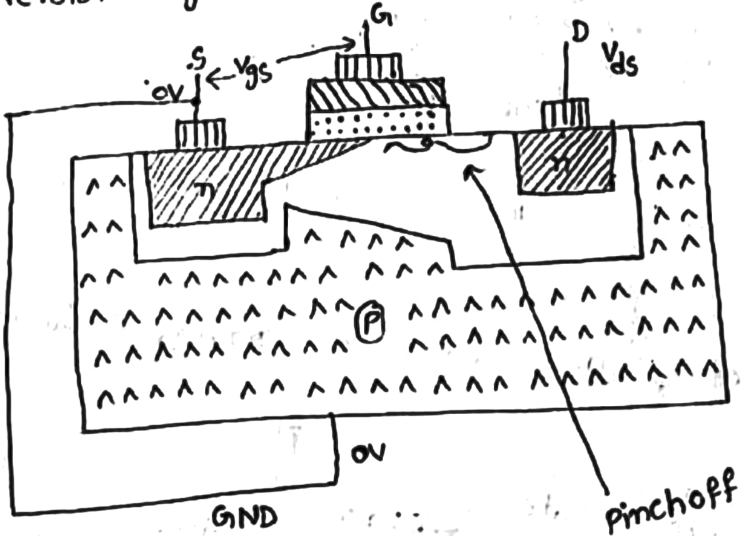
* The effective gate voltage $V_g = V_{gs} - V_t$ as no current flows when $V_{gs} < V_t$ As long as $(V_{gs} - V_t) \geq V_{ds}$, the channel will be Inverted at drain end.

* In this condition, device is in non-saturated region of operation.

condition-3:- $V_{ds} > V_{gs} - V_t$

If V_{ds} is Increased to greater than $(V_{gs}-V_t)$, in such condition IR drop = $V_{gs}-V_t$ takes place over less than whole length of channel.

* so, at drain terminal Insufficient electric field available to give rise to an Inversion layer create a channel. The channel is Pinchoff.



* In this condition, diffusion current completes the path from source to drain.

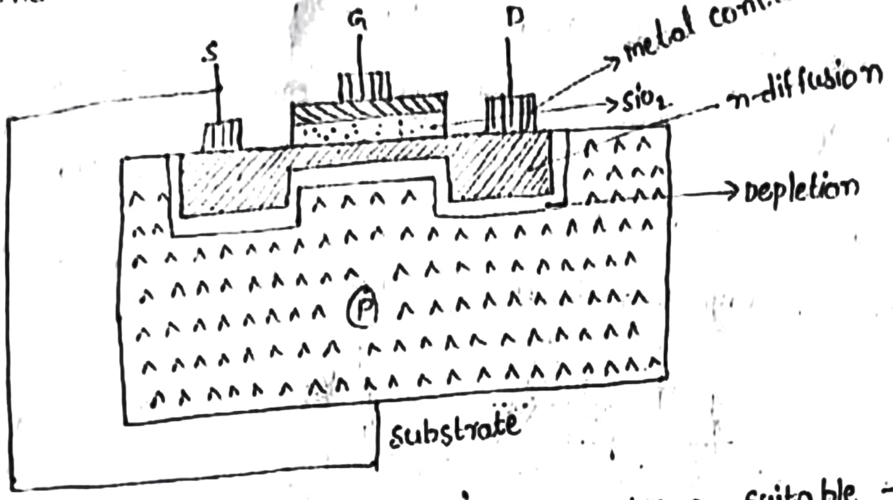
* The device operated in saturation region. The device behaves a con source for Increase of V_{ds} above $V_{ds} = V_{gs} - V_t$.

Nmos depletion mode :-

when transistor operated in depletion mode, even when $V_{gs}=0$

* Applying -ve voltage to the gate, current flows in a n-type transistor and conduction takes place.

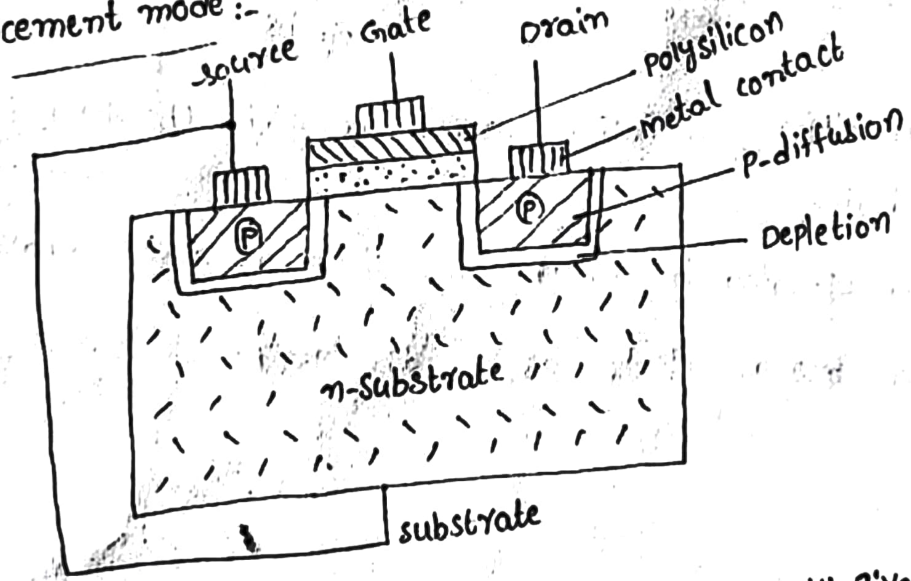
* Similarly, +ve voltage to the gate, current flows in a p-type transistor and conduction takes place.



* The channel width can be controlled by applying suitable -ve voltage to the Gate terminal.

* variation in gate voltage allow control of current between source & drain.

pmos enhancement mode :-



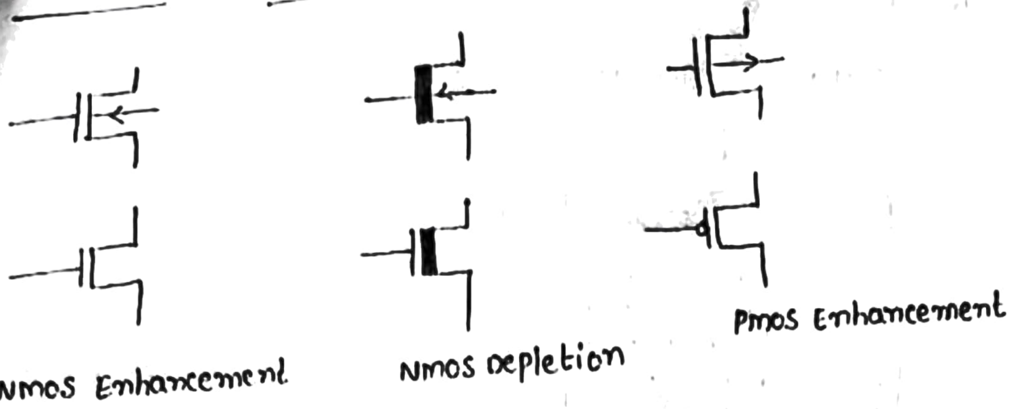
* Applying -ve voltage between gate and source will give rise to formation of p-type channel between source and drain.

* If drain is made -ve w.r.to source current can flow through chann

* Here the current is carried by holes.

* The hole mobility (μ_p) compared to electron mobility (μ_n) is 2.5 times less.

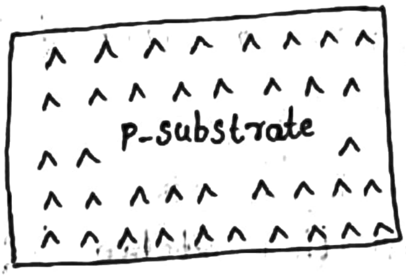
Commonly used symbols for Nmos and Pmos :-



Nmos Fabrication process :-

Step1 :- wafer preparation :-

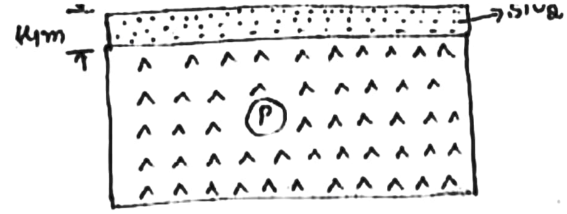
Let us consider a single peace of silicon material and it is moderately doped with P-substrate. The wafer diameter is The P-substrate is boran. 75 to 150mm and thickness is 0.4mm.



Step2 :- oxidation:-

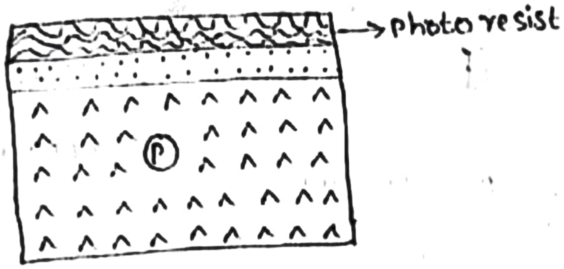
To Protect the surface, a layer of silicon dioxide (SiO_2) is grown over the surface. The thickness of SiO_2 is typically 1µm.

* This layer act as a barrier to dopants during processing.

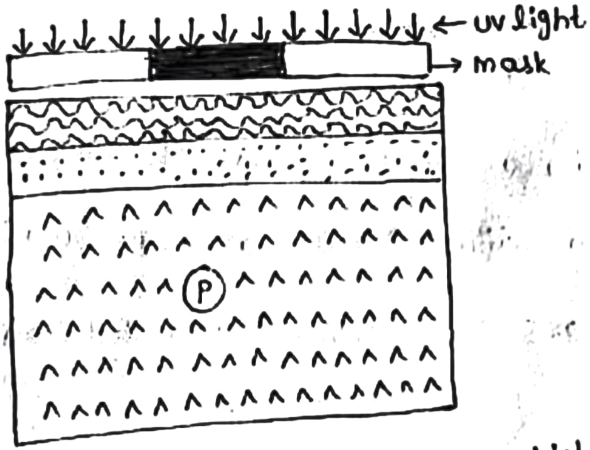


Step 3 :- Photo resist :-

The surface is now covered with photo resist which is deposited on wafer.

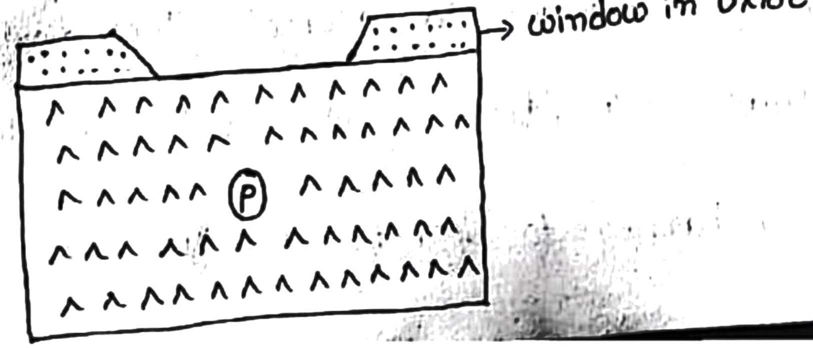


Step 4 :- masking :- The Photo resist layer is exposed to UV light through a mask.



The mask corresponds to regions into which diffusion is to take place with transistor channels.

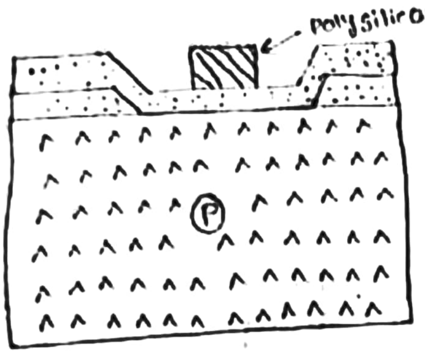
Step 5 :- Etching :- The regions are then etched together so that the wafer surface is exposed in window defined by mask.



Step 6 :- Poly silicon (or) Poly crystallization

Remaining Photoresist is removed and a thin layer of SiO_2 is grown over the entire surface and poly silicon is deposited on top of this to form the gate structure.

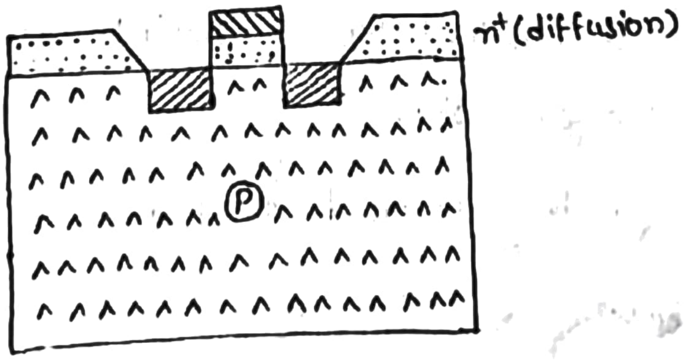
The Polysilicon layer consists of heavily doped Polysilicon deposited by chemical vapour deposition (CVD).



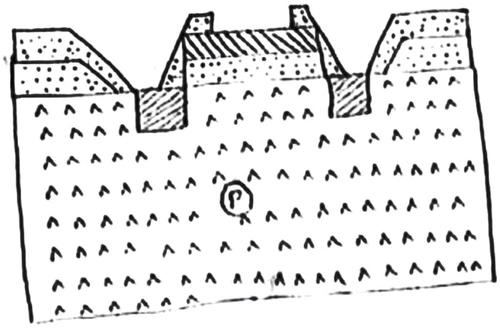
Step 7 :- n^+ diffusion :-

The thin oxide is removed to expose areas into which n-type Impurities are to be diffused from source and drain.

* Diffusion is achieved by heating the wafer to high temperature and passing gas containing the desired n-type Impurity over the surface

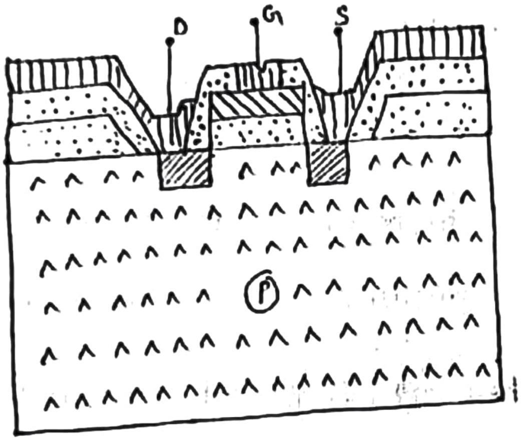


Step 8 contact cuts (holes) :- Thick Silicon dioxide (SiO_2) is grown and then masked with photo resist and then etched to expose specific areas of poly silicon gate, drain and source areas where connections are made.



Step 9 :- metallization:-

The metal (Al) is deposited over its surface to a thickness typically of 1μm. This metal layer is then masked and etched to form the required Interconnection pattern.



CMOS Fabrication process :-

CMOS can be obtained by Integrating both the NMOS and PMOS devices on the same chip.

The CMOS can be fabricated using different processes such as

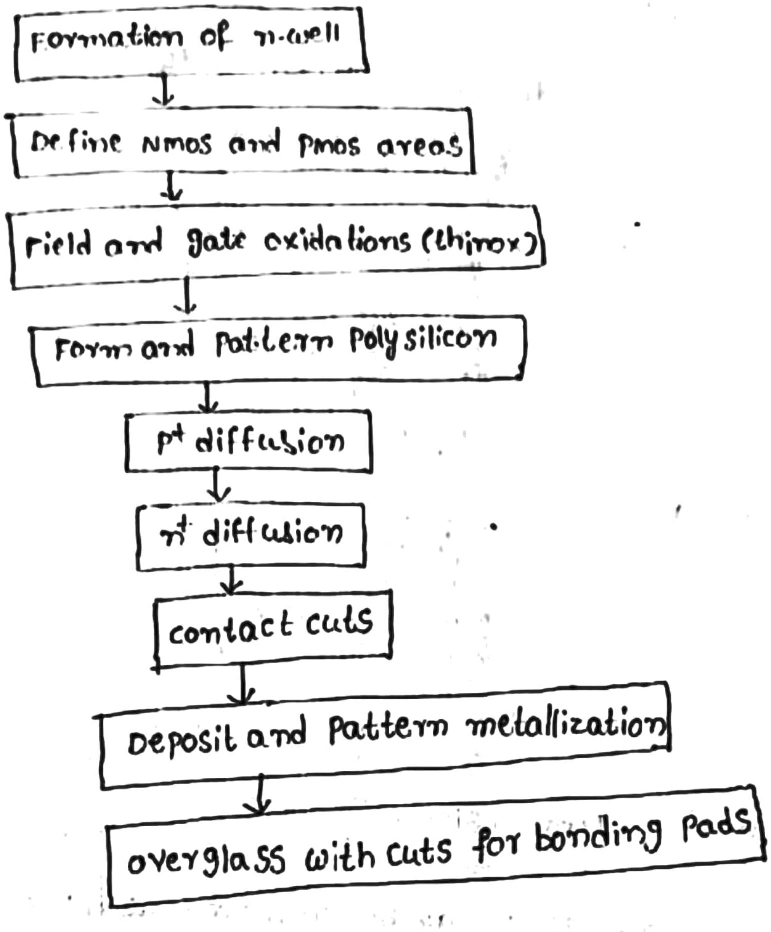
i) N-well Process

ii) P-well process

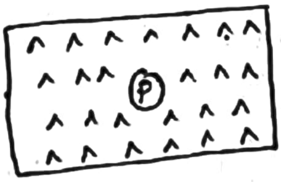
iii) Twin tub process

1. N-well cmos fabrication :-

majoy steps :-

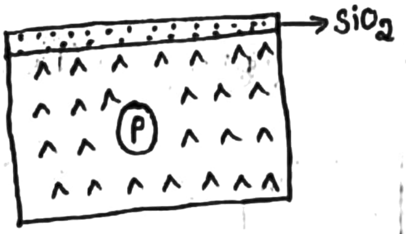

Formation of n-well
Define Nmos and Pmos areas
Field and gate oxidations (thinox)
Form and Pattern Polysilicon
p^+ diffusion
n^+ diffusion
contact cuts
Deposit and Pattern metallization
overglass with cuts for bonding Pads

Step1 :- consider a P-type substrate

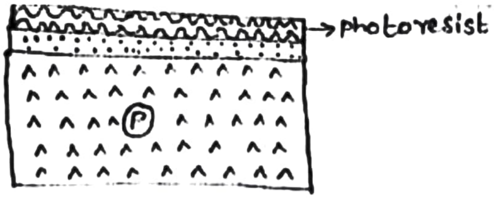


Step2 :- oxidation

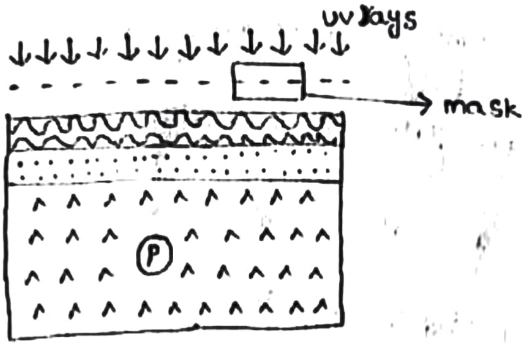
The entire surface is coated with SiO_2 layer



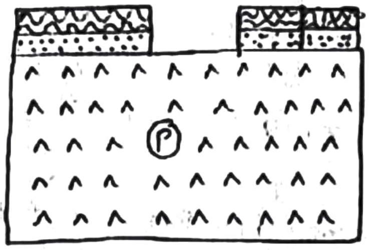
Step3 :- photoresist :- Now, photoresist is covered entire surface.



Step4 :- masking The photo resist is exposed to UV rays through the N-well mask.

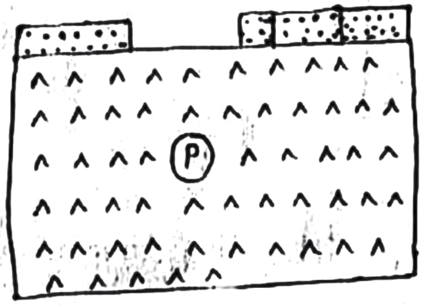


Step5 :- Etching

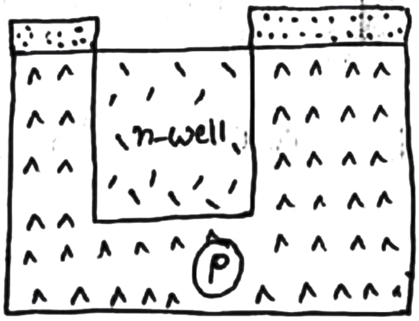


Step6 :- Removal of Photoresist :-

photo resist can be removed using of hydroflouric acid. then the entire layer of photoresist is stripped off.

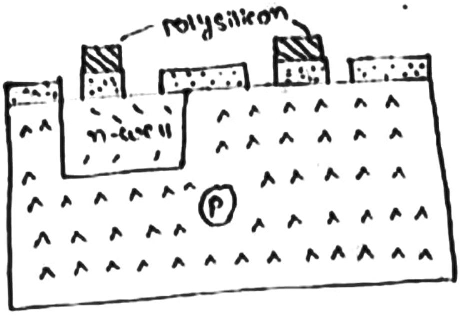


Step7 :- Formation of N-well :- By using diffusion process N-well is formed



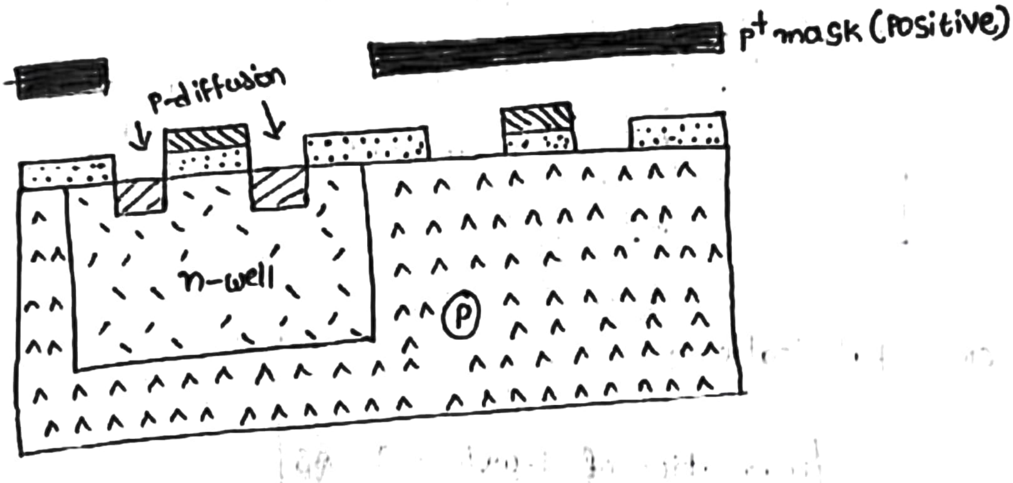
Step 8 :- Pattern the Poly silicon layer :-

Pattern the Poly sillicon layer which is deposited after thinoxide



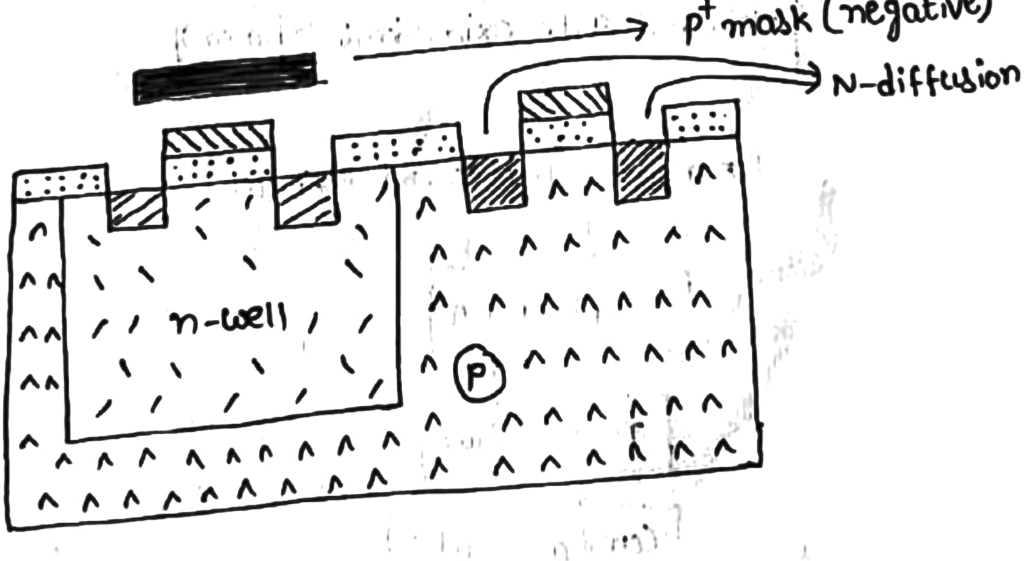
Step 9 :- p^+ diffusion and masking

P-diffusion regions are diffused to form the terimals of the pmos.

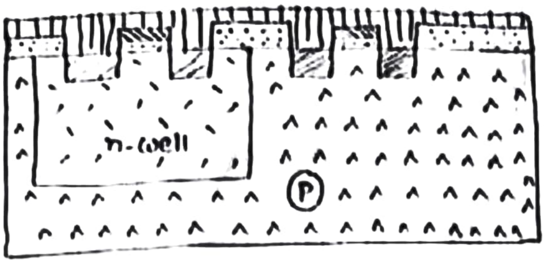


Step 10 :- masking and n^+ diffusion :-

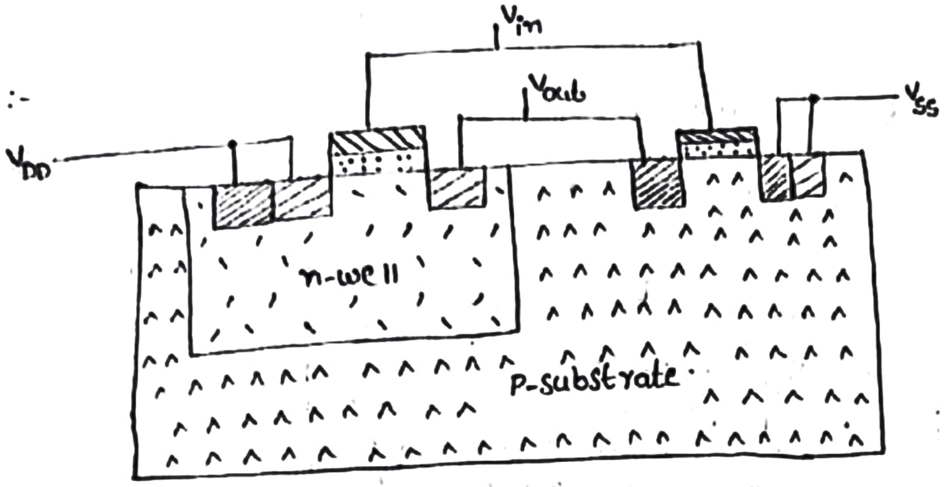
To form n-transistor, n-diffusion regions are used.



step 11 :- metallization



step 12 :-



2. P-well CMOS fabrication :-

Formation of P-well regions

↓

Define NMOS and PMOS areas

↓

Field and gate oxidations (thinox)

↓

Form and Pattern Polysilicon

↓

p^+ diffusion

↓

n^+ diffusion

↓

Contact cuts

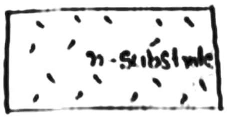
↓

Deposit and Pattern metallization

↓

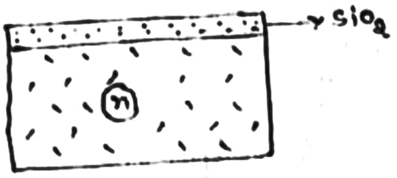
over glass with cuts for bonding Pads

Step1 :- consider N-type substrate



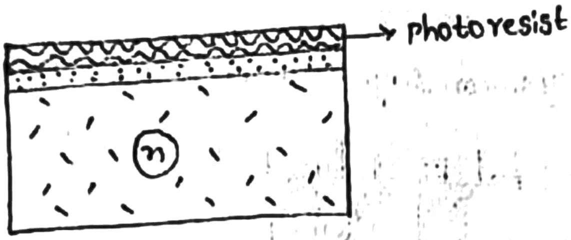
Step2 :- oxidation

The entire surface is coated with SiO_2 layer

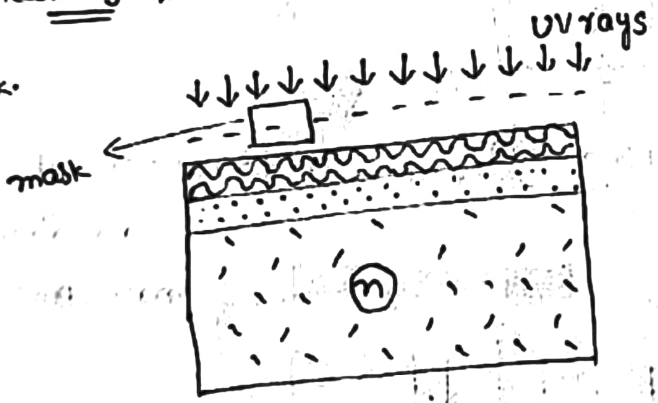


Step3 :- Photoresist

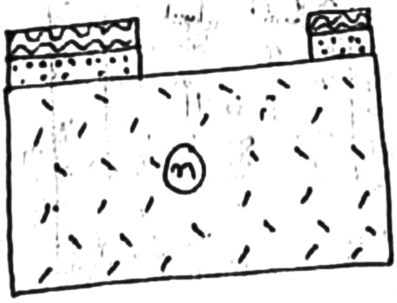
Now the entire surface covered with a photoresist



Step4:- masking :- The photoresist is exposed to UV rays through the p-well mask.

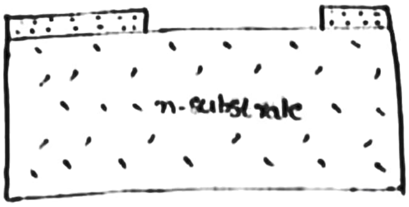


Step 5 :- Etching

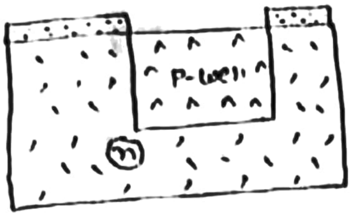


Step6 Removal of photoresist :-

The photo resist can be removed using of hydroflouric acid. Then the entire layer of photoresist is stripped off.

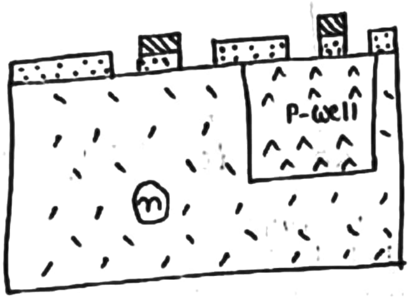


step7 :- formation of P-well :- By using diffusion Process P-well is formed

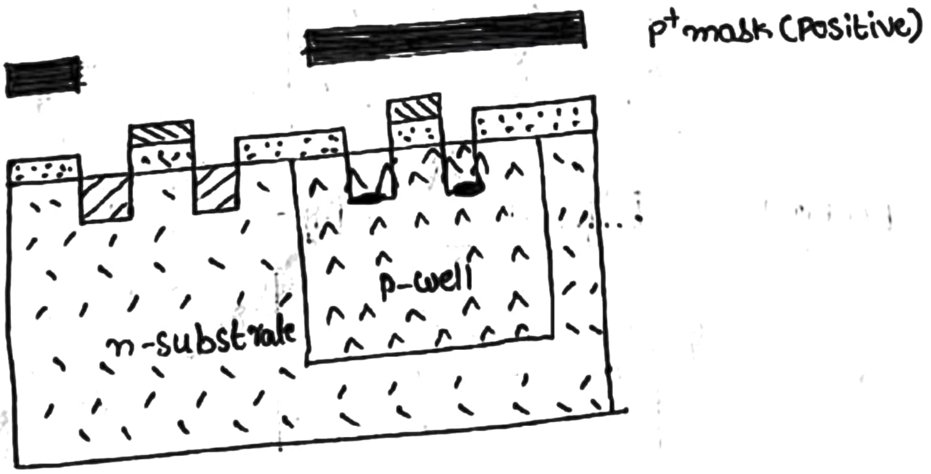


step8 :- Pattern the Polysilicon layer :-

The Polysilicon layer which is deposited after thin oxide.

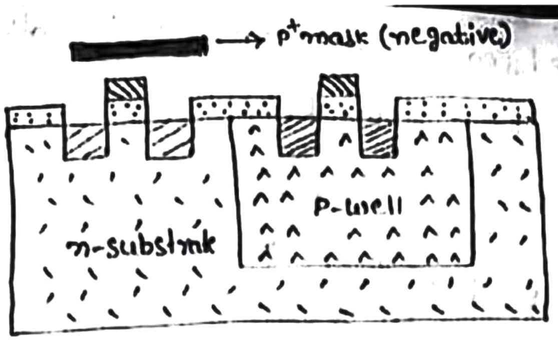


step9 :- p+ diffusion and masking :- To form P-transistor, P-diffusion regio are diffused.

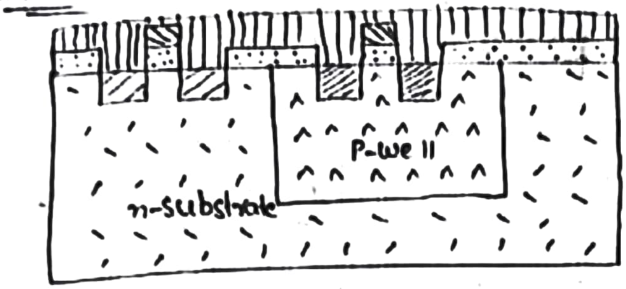


step 10 :- masking and n+ diffusion:-

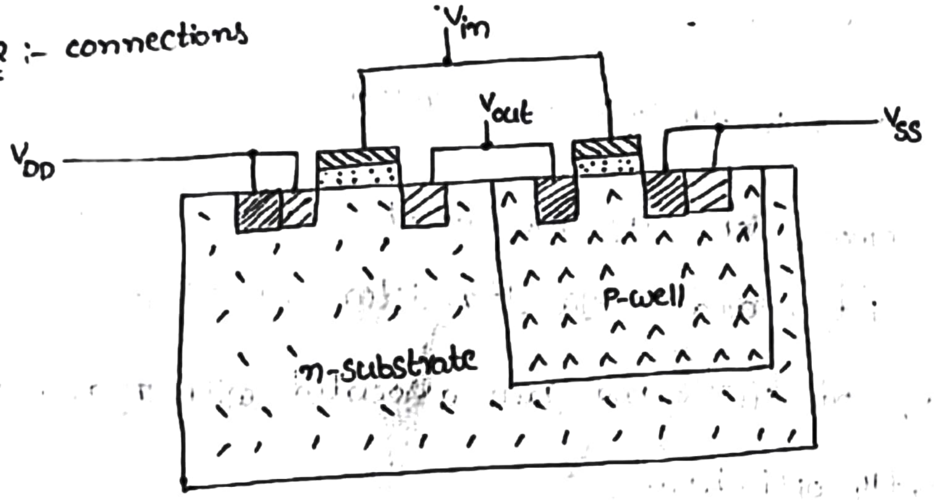
To form n-transistor, n-diffusion regions are used.



step 11 :- metallization



step 12 :- connections



Twin tub Process :-

→ A twin-tub Process is a logical extension of P-well and n-well approaches.

→ The Process starts with a substrate of high resistivity n-type material and then both P-well and n-well regions are created.

→ with this Process it is Possible to Preserve the Performance of n-transistors without compromising P-transistors

→ Doping can be controlled easily and latchup is also achieved as the twin-tub process allows separate optimization of P and n transistors.

An Inverter arrangement of twin-tub as shown in fig.

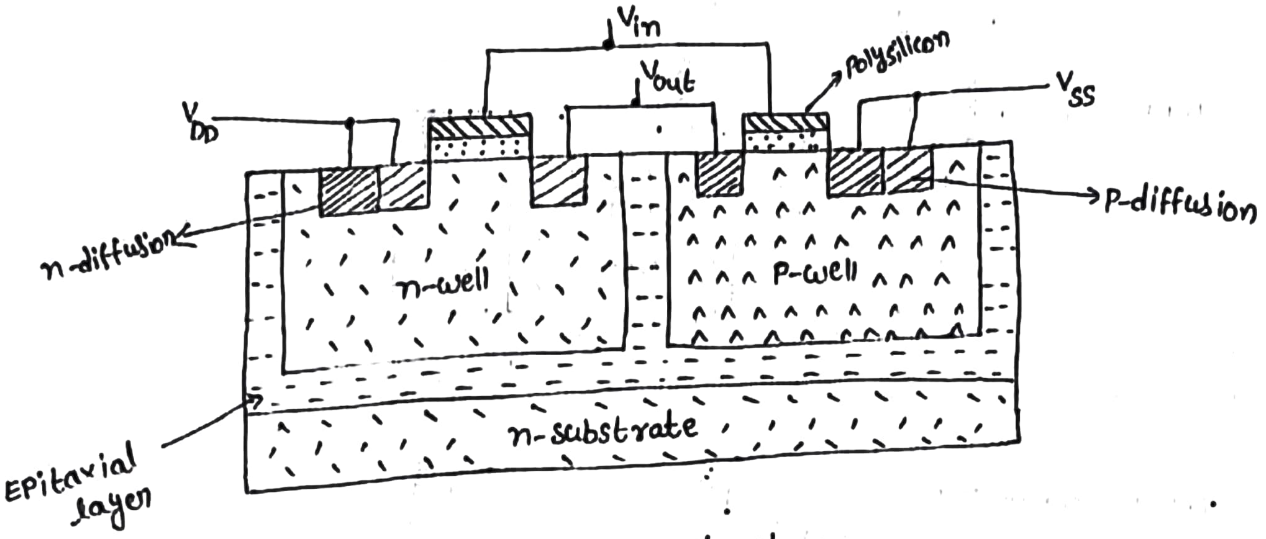


Fig:- Twin-tub Inverter structure.

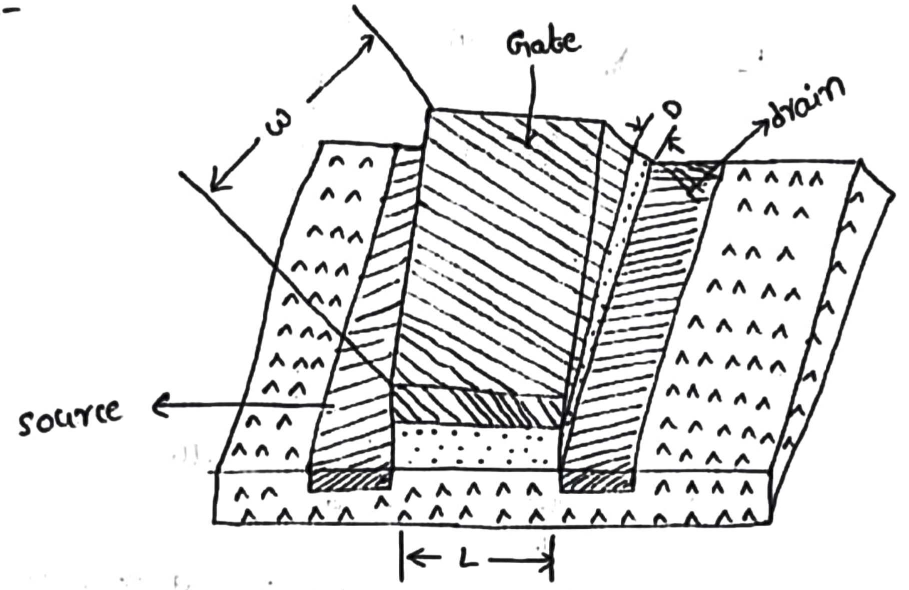
→ The twin-tub cmos technology provides the basic for separate optimization of P-type and n-type transistor making it possible for threshold voltage, body effect and gain associated with n and P-devices to be Independently optimized.

Comparisions between cmos and Bipolar technologies

cmos technolgy	Bipolar technology
* Low static power dissipation	* High static power dissipation
* High Input Impedance (Low drive current)	* Low Input Impedance (High drive current)
* High Noise margin	* Low Noise margin
* High packaging density	* Low packaging density
* Low transconductance (g_m)	* High transconductance
* output current produced is low	* output current produced is high
* Bidirectional capability	* unidirectional capability

Basic electrical properties of MOS circuits

Drain-to-source current (I_{ds}) versus drain-to-source voltage (V_{ds}) relationships :-



mos transistors are Voltage controlled device. A voltage on the gate terminal Induces a charge in the channel that exists between source and drain. The charge then move from source to drain under the Influence of electric field generated by voltage V_{ds} applied between drain and source.

The charge Induced is dependent on the gate to source Voltage V_{GS} current I_{DS} is dependent on both V_{GS} and V_{DS}.

The drain-to source current I_{ds} is given by

$$I_{ds} = \frac{\text{charge Induced in channel } (Q_c)}{\text{Electron transit time } (\tau)} = \frac{Q_c}{\tau} \quad \text{—①}$$

$$\text{Electron transit time, } \tau_{sd} = \frac{\text{Length of channel}}{\text{velocity}} = \frac{L}{V} \quad \text{—②}$$

velocity $V = \mu E_{ds}$ —③

where, μ = mobility of holes (or) electrons

E_{ds} = Electric field (drain to source)

But $E_{ds} = \frac{V_{ds}}{L}$ —④

substitute eq ④ in ③

$$V = \mu \cdot \frac{V_{ds}}{L} \quad —⑤$$

substitute eq ⑤ in ②

$$\tau_{sd} = \frac{L}{\frac{\mu V_{ds}}{L}} = \frac{L^2}{\mu V_{ds}} \quad —⑥$$

8· The electron and hole mobilities at room temperature

$\mu_n = 650\ cm^2/V\text{-}sec$, $\mu_p = 240\ cm^2/V\text{-}sec$

Non-saturated region :-

* The Average length along the channel is considered as $\frac{V_{ds}}{2}$

* Effective gate voltage $V_g = V_{gs} - V_t$

where V_t = threshold voltage is needed to Invert the charge under the gate and to establish the channel.

The charge per unit area = $E_g \, \varepsilon_0 \, \varepsilon_{ins}$ —⑦

where, ε_0 → permittivity of free space $(8.854 \times 10^{-14}\ F/cm)$

ε_{ins} → Relative permitivity of Insulation between gate and channel [Ex $SiO_2 = 4$]

E_g → Average electric field gate to channel consider $E_g = \frac{\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right]}{D}$

where $D \rightarrow$ oxide thickness

$$Q_c = \frac{\varepsilon_0 \varepsilon_{ins} \omega L}{D} \left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] \quad \text{—— ⑧}$$

substitute eq ⑧ in eq ①,

$$I_{ds} = \frac{\frac{\varepsilon_0 \varepsilon_{ins} \omega \cancel{L}}{D}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right]}{\frac{L^{\cancel{2}}}{\mu V_{ds}}} = \frac{\varepsilon_0 \varepsilon_{ins} \omega \mu}{L D}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds}$$

$$= \frac{\varepsilon_0 \varepsilon_{ins} \mu}{D} \cdot \frac{\omega}{L}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds} \quad \text{—— ⑨}$$

$$= \frac{K\omega}{L}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds} \quad \text{—— ⑩} \quad \left[\because K = \frac{\varepsilon_0 \varepsilon_{ins} \mu}{D}\right]$$

$$I_{ds} = \beta\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds} \quad \text{—— ⑪} \quad \left[\because \beta = \frac{K\omega}{L}\right]$$

Gate/channel capacitance, $C_g = \frac{\varepsilon_0 \varepsilon_{ins} \omega L}{D}$ [parallel plates] —— ⑫

substitute eq ⑫ in ⑨,

$$I_{ds} = \frac{C_g \mu}{L^2}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds} \quad \text{—— ⑬}$$

consider $C_g = C_0 \omega L$ —— ⑭

substitute eq ⑭ in eq ⑬,

$$I_{ds} = \frac{C_0 \omega \cancel{L} \mu}{L^{\cancel{2}}}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds}$$

$$\boxed{I_{ds} = \frac{C_0 \omega \mu}{L}\left[(V_{gs}-V_t) - \frac{V_{ds}}{2}\right] V_{ds}} \quad \text{—— ⑮}$$

For saturated region:-

In this region, consider $V_{ds} = V_{gs} - V_t$.

eq (13), becomes

$$I_{ds} = \frac{k\omega}{l}\left[(V_{gs}-V_t) - \frac{(V_{gs}-V_t)}{2}\right](V_{gs}-V_t) \qquad [\because V_{ds} = V_{gs}-V_t]$$

—— (16)

$$= \beta\left[(V_{gs}-V_t)^2 - \frac{(V_{gs}-V_t)^2}{2}\right] \qquad [\because \beta = \frac{k\omega}{L}]$$

$$= \beta\left[\frac{(V_{gs}-V_t)^2}{2}\right] \text{ —— (17)}$$

Now, gate per channel capacitance $C_g = \frac{\varepsilon_0 \varepsilon_{ins} \omega L}{D}$

eq (16), becomes

$$I_{ds} = \frac{C_g \mu}{2L^2}\left[(V_{gs}-V_t)^2\right]$$

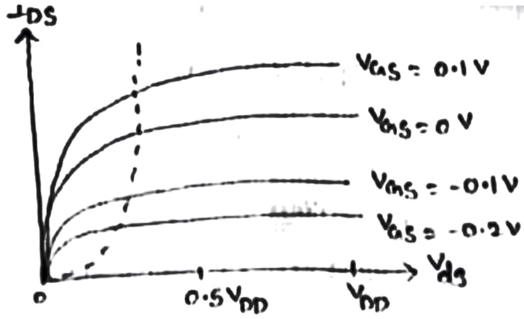
consider $C_g = C_0 \omega L$

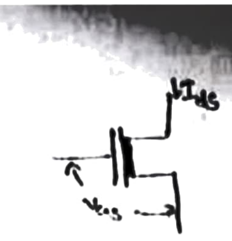
$$\Rightarrow I_{ds} = \frac{C_0 \omega L \mu}{2L^2}\left[(V_{gs}-V_t)^2\right]$$

$$\Rightarrow \boxed{I_{ds} = \frac{C_0 \omega \mu}{2L}\left[(V_{gs}-V_t)^2\right]} \text{ —— (18)}$$

The current in 3 regions

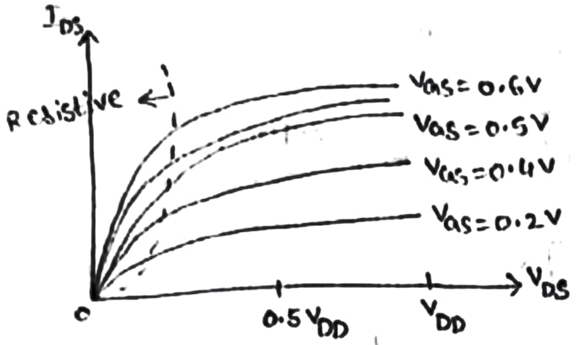
$$I_{ds} = \begin{cases} 0 & ; \; V_{gs} < V_t \rightarrow \text{cut-off} \\ \beta(-V_{gs}-V_t-\frac{V_{ds}}{2})V_{ds} & ; \; V_{ds} < V_{dsat} \rightarrow \text{linear} \\ \frac{\beta}{2}\left[V_{gs}-V_t\right]^2 & ; \; V_{ds} > V_{dsat} \rightarrow \text{saturation} \end{cases}$$

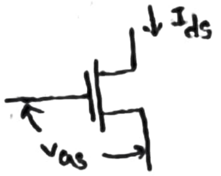




symbol

a) Depletion mode device





symbol

b) Enhancement mode device

MOS transistor threshold voltage (V_t) :-

→ For switching an enhancement mode mos transistor from OFF to ON state, applying sufficient gate voltage to neutralize these charges and enable the underlying silicon to undergo an inversion due to electric field from the gate.

→ For switching an depletion mode nmos transistor from ON to OFF state consists in applying enough voltage to the gate to add to the stored charge and Invert the 'n' Implant region to p-region.

The Threshold voltage (V_t) is given by

$$V_t = \phi_{ms} \frac{Q_B - Q_{SS}}{C_0} + 2\phi_{fN} \quad — ①$$

Q_B → charge per unit area in the depletion layer

Q_{SS} → charge density at Si : SiO$_2$ Interface

C_0 → capacitance per unit gate area

ϕ_{ms} → work function difference between gate and silicon

ϕ_{fN} → Fermi level potential between inverted surface and bulk silicon

To evaluate V_t, each term is determined as follows

$$Q_B = \sqrt{2\varepsilon_0 \varepsilon_{si}\, qN\,(2\phi_{fN} + V_{SB})} \quad \text{coulomb}/m^2 \quad — ②$$

$$\phi_{fN} = \frac{KT}{q} \ln\left(\frac{n}{n_i}\right) \text{volts} \quad — ③$$

$Q_{SS} = (1.5 \text{ to } 8) \times 10^{-8}$ coulomb/m^2 depending on crystal orientation

where,

V_{SB} → substrate bias voltage (-ve w.r.to nMOS, +ve for pMOS)

q → 1.6×10^{-19} coulomb

N → Impurity concentration in substrate (N_A (or) N_D)

ε_{si} → Relative permitivity of silicon (≅11.7)

n_i → Intrinsic electron concentration ($1.6 \times 10^{10}/cm^3$ at 300°K)

K → Boltzmann's constant (1.4×10^{-23} Joule/°K)

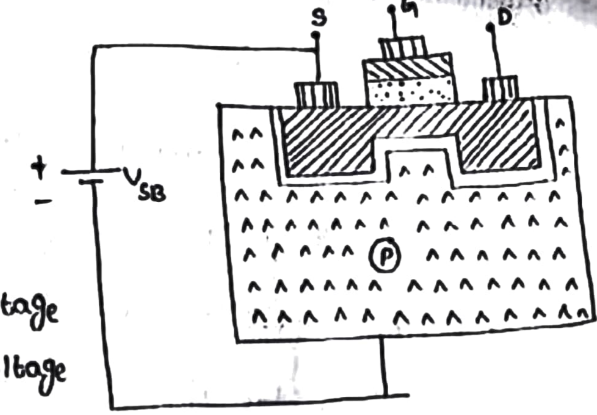
Alternative expression for V_t in terms of V_{SB} is as follows

$$V_t = V_t(0) + \left(\frac{D}{\varepsilon_0 \varepsilon_{ins}}\right) \sqrt{2\varepsilon_0 \varepsilon_{si}\, QN}\, \sqrt{V_{SB}}$$

where $V_t(0)$ → Threshold voltage for $V_{SB} = 0$

D → oxide thickness

Body effect :-



Increasing V_{SB} causes the channel to be depleted of charge carriers and thus the threshold voltage is raised.

The variation of threshold voltage due to source to substrate voltage is referred as "Body effect".

The relation ship between threshold voltage and substrate bias voltage is given by

$$\Delta V_t = \gamma (V_{SB})^{1/2}$$

where γ is constant it depends on substrate doping, so that more lightly doped the substrate, smaller will be body effect.

MOS transistor transconductance (g_m) and output conductance (g_{ds}) :-

Transconductance (g_m) :-

It is defined as the ratio of change in output current to the change in Input voltage by taking output voltage is constant.

$$g_m = \left.\frac{\delta I_{ds}}{\delta V_{gs}}\right|_{V_{ds} = \text{constant}} \quad \text{—①}$$

we know that,

$$I_{ds} = \frac{Q_c}{\tau_{sd}} \Rightarrow \delta I_{ds} = \frac{\delta Q_c}{\tau_{sd}} \quad \text{—②}$$

$$\text{But } \tau_{sd} = \frac{L^2}{\mu V_{ds}} \quad \text{—③}$$

substitute eq③ in eq②

$$\delta I_{ds} = \frac{\delta Q_t \mu V_{ds}}{L^2} \quad — ④$$

we know that, change in charge $Q = CV$

$$\Rightarrow \delta Q_c = C_g \delta V_{gs} \quad — ⑤$$

substitute eq⑤ in eq④,

$$\delta I_{ds} = \frac{C_g \delta V_{gs} \mu V_{ds}}{L^2} \quad — ⑥$$

$$\text{Transconductance, } g_m = \frac{\delta I_{ds}}{\delta V_{gs}} = \frac{C_g \mu V_{ds}}{L^2} \quad — ⑦$$

under saturation region, $V_{ds} = V_{gs} - V_t$

$\Rightarrow$ eq⑦ becomes,

$$g_m = \frac{C_g \mu (V_{gs} - V_t)}{L^2} \quad — ⑧$$

we know that $C_g = \frac{\varepsilon_0 \varepsilon_{ins} \omega L}{D}$ — ⑨

substitute eq⑨ in eq⑧,

$$g_m = \frac{\varepsilon_0 \varepsilon_{ins} \omega \cancel{L} \mu}{D L^2} (V_{gs} - V_t)$$

$$g_m = \frac{\varepsilon_0 \varepsilon_{ins} \omega \mu}{D L} [V_{gs} - V_t] \quad — ⑩$$

$$= \frac{\varepsilon_0 \varepsilon_{ins} \mu}{D} \cdot \frac{\omega}{L} [(V_{gs} - V_t)]$$

$$= \frac{K \omega}{L} [V_{gs} - V_t] \quad — ⑪ \qquad \left[\because K = \frac{\varepsilon_0 \varepsilon_{ins} \mu}{D}\right]$$

$$g_m = \beta [V_{gs} - V_t] \quad — ⑫ \qquad \left[\because \beta = \frac{K\omega}{L}\right]$$

output conductance (g_{ds}) :-

$$g_{ds} = \frac{\delta I_{ds}}{\delta V_{gs}}$$

$$= \lambda \cdot I_{ds} \doteq \left(\frac{1}{L}\right)^2$$

From above equation, $\lambda \propto \frac{1}{L}$ and $I_{ds} \propto \frac{1}{L}$

mos transistor figure of merit (ω_o) :-

The figure of merit is a quantity used to characterized the performance of a device related to other devices of same time.

It is defined as the ratio of transconductance to the gate-to-channel capacitance

$$\omega_o = \frac{g_m}{C_g} = \frac{\frac{\cancel{C_g}\mu(V_{gs}-V_t)}{L^2}}{\cancel{C_g}} \qquad \left[\because g_m = \frac{C_g\mu(V_{gs}-V_t)}{L^2}\right]$$

$$= \frac{\mu V_{ds}}{L^2} \quad -① \quad [\because \text{In saturation region, } V_{ds} = V_{gs}-V_t]$$

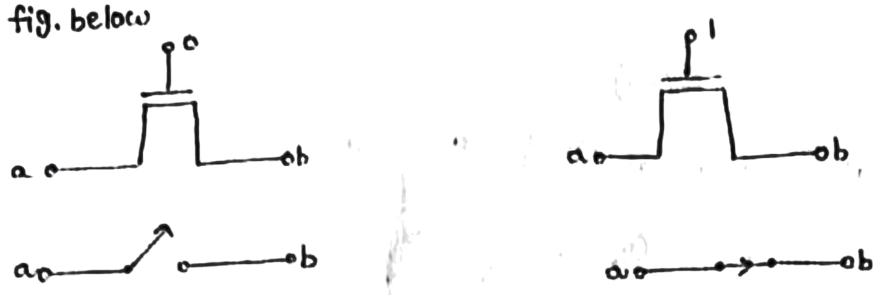
$$\omega_o = \frac{1}{\tau_{sd}} \quad -② \quad \left[\because \tau_{sd} = \frac{L^2}{\mu V_{ds}}\right]$$

From eq ①, switching speed depends on
i) carrier mobility
ii) Inversly on square of channel len
iii) gate Voltage

A high speed switching circuits, high g_m as possible.

Pass transistor :-

The transistor can be used as an ON-OFF switch as shown in fig. below



a) open switch

b) closed switch

Fig :- mos transistor as a switch

* The switch is turned off by setting $V_{gs} = 0$, the channel disappears and a small amount of leakage current flows at the drain end.

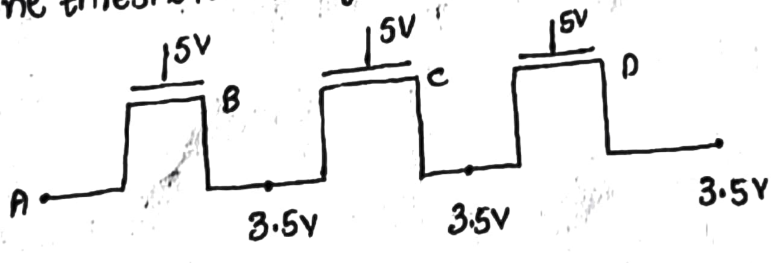
* consider $V_{gs} = V_{DD}$, the switch is turned on, the current flows through it.

* If the gate and drain are pass transistor are both HIGH, the source will rise to the lower of two potentials.

$S =$ Lower of $(V_{DD}, (V_{DD} - V_{Th}))$

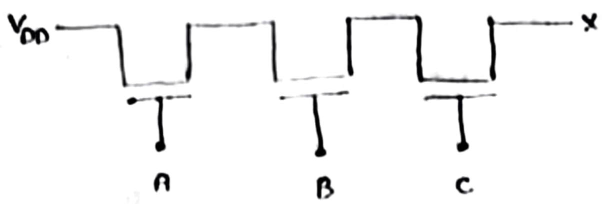
$S =$ ~~to~~ $V_{DD} - V_{Th}$

* If the gate and drain are both at V_{DD} the source can only rise to one threshold voltage below through gate.



$V_{TH} =$ ~~3.5~~ $1.5V$ $\left[\because V_{DD} - V_{Th} = 5V - 3.5V\right]$ $= 1.5V$

A number of transislors can be used as switches in series in switching logic arrays. An AND array as shown in fig.

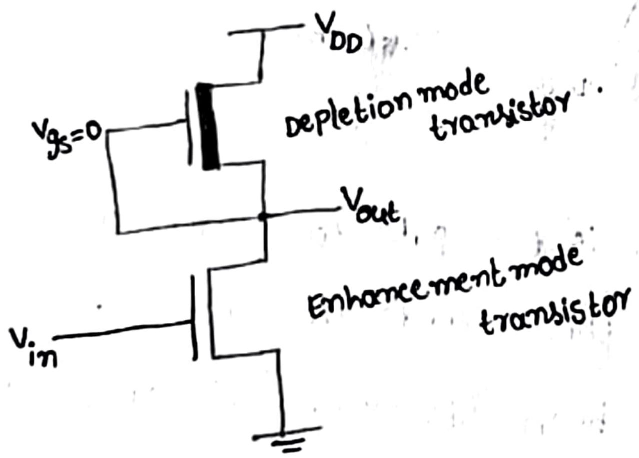


$X = ABC$ [Logic 1 = $V_{DD} - V_t$]

Nmos Inverter :-

The basic Inverter circuit requires a transistor with source connected to ground and a load resistor of some sort connected from the drain to positive supply rail V_{DD}. The output is taken from the drain and the Input applied between gate and ground.

The NMOS Inverter circuit as shown in fig.



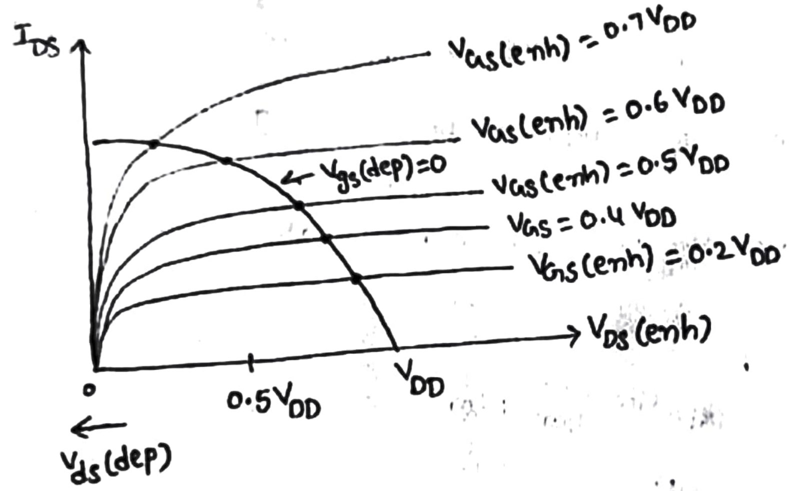
In this configuration, with no current drawn from the output, the current I_{DS} for both transistors are equal.

For the depletion mode transistor, gate is connected to the source so it is always ON and only the characteristic curve $V_{GS}=0$ is relevant.

In this configuration the depletion mode device is called the pull-up (P.U) and enhancement mode device is called the pull down (P.D) transistor.

The transfer characteristic of such an Inverter can be obtained by superimposing $V_{GS}=0$ depletion mode characteristic curve on the family of curves for the enhancement mode device with the condition the maximum voltage across the enhancement mode device corresponds to minimum voltage across the depletion mode device.

The transfer characteristics of NMOS Inverter as shown in fig.



Now the Input Voltage V_{in} is the gate-to-source voltage V_{GS} for the pull down transistor. As this exceeds the threshold voltage of the pull down transistor, current begins to flow. This causes the output voltage V_{out} to decrease and further Increase in V_{in} causes pull down transistor to come out of saturation and become resistive.

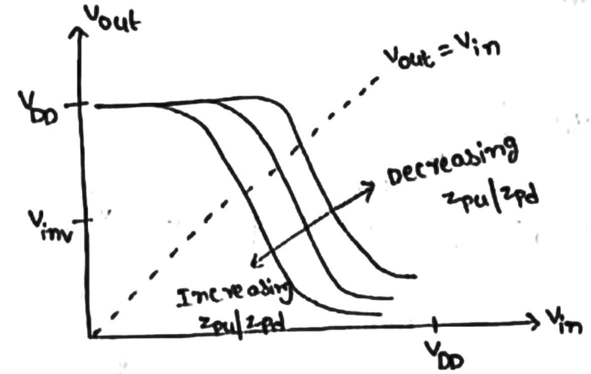


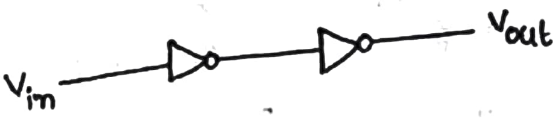
Fig:- Nmos Inverter transfer characteristics

The point on the transfer characteristic at which $V_{out} = V_{in}$ is denoted as V_{inv}.

Determination of pull-up to pull-down ratio (Z_{pu}/Z_{pd}) for an Nmos Inverter driven by another Nmos Inverter :-

consider two Inverters are arranged in a cascaded manner. These two are operate in a saturation region.

The depletion mode transistor for which $V_{gs}=0$, so maximum drain current flows i.e all devices are ON.



For equal margins around the Inverter threshold, we said $V_{inv} = 0.5\,V_{DD}$ at this point both the transistors are in saturation.

So, $V_{in} = V_{out} = V_{inversion}$

we know that,

$$I_{ds} = \frac{kw}{L}\left[\frac{(V_{gs}-V_t)^2}{2}\right] \text{ —① }$$

[∵ saturation regio

In depletion mode,

$$I_{ds} = K\cdot\frac{W_{pu}}{L_{pu}}\left[\frac{(V_{gs}-V_{td})^2}{2}\right]$$

$$= K\cdot\frac{W_{pu}}{L_{pu}}\left[\frac{(-V_{td})^2}{2}\right] \text{ --- } ② \qquad [\because V_{gs}=0V]$$

For enhancement mode,

$$I_{ds} = K\cdot\frac{W_{pd}}{L_{pd}}\left[\frac{(V_{inv}-V_t)^2}{2}\right] \text{ --- } ③ \qquad [\because V_{gs}=V_{inv}]$$

equating eq ② & ③,

$$\cancel{K}\,\frac{W_{pu}}{L_{pu}}\left[\frac{(-V_{td})^2}{2}\right] = \cancel{K}\,\frac{W_{pd}}{L_{pd}}\left[\frac{(V_{inv}-V_t)^2}{2}\right]$$

consider $Z_{pu}=\frac{L_{pu}}{W_{pu}}$, $Z_{pd}=\frac{L_{pd}}{L_{pd}}$

The Aspect ratio (Z) is defined as the ratio of length to the width.

$$\frac{1}{Z_{pu}}\left[\frac{(-V_{td})^2}{\cancel{2}}\right] = \frac{1}{Z_{pd}}\left[\frac{(V_{inv}-V_t)^2}{\cancel{2}}\right]$$

$$\Rightarrow (V_{inv}-V_t)^2 = \frac{Z_{pd}}{Z_{pu}}\left[-V_{td}\right]^2$$

$$V_{inv}-V_t = \frac{-V_{td}}{\sqrt{\frac{Z_{pu}}{Z_{pd}}}} \text{ --- } ④$$

Substitute typical values of

$V_t = 0.2\,V_{DD}$, $V_{td} = -0.6\,V_{DD}$, $V_{inv} = 0.5\,V_{DD}$ in eq ④

$$0.5\, V_{DD} - 0.2\, V_{DD} = \frac{0.6\, V_{DD}}{\sqrt{\frac{Z_{pu}}{Z_{pd}}}}$$

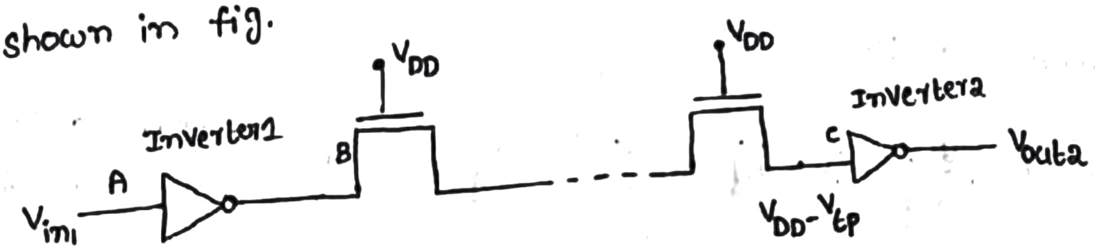
$$\sqrt{\frac{Z_{pu}}{Z_{pd}}} = \frac{0.6\, \cancel{V_{DD}}}{0.3\, \cancel{V_{DD}}}$$

$$\frac{Z_{pu}}{Z_{pd}} = \left(\frac{2}{1}\right)^2 = \frac{4}{1} \Rightarrow \boxed{\frac{Z_{pu}}{Z_{pd}} = \frac{4}{1}}$$

$$= 4:1$$

Determination of Z_{pu}/Z_{pd} ratio for an Nmos Inverter driven through one (or) more pass transistors :-

Some times the Input to an Inverter may come from the output of an Inverter but after passing through one (or) more Nmos transistors that are used as pass transistors. The arrangement as shown in fig.



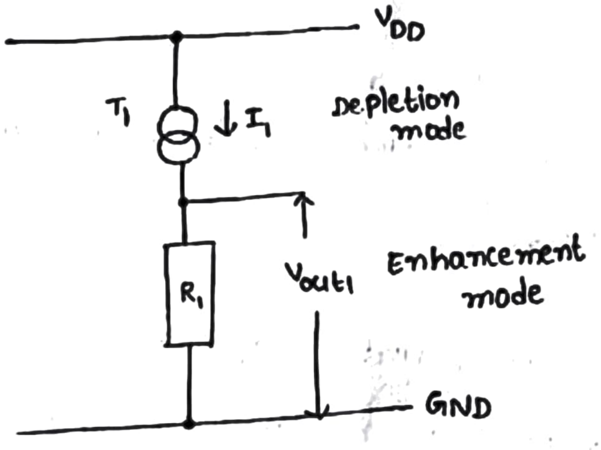
when A in is at 0 volts, B is at V_{DD}, but the voltage into Inverter 2 at Point C has got reduced from V_{DD} by threshold voltage of series pass transistor. There is a reduction in voltage by V_{tp} where V_{tp} is the threshold voltage.

Hence Input voltage to Inverter 2 is

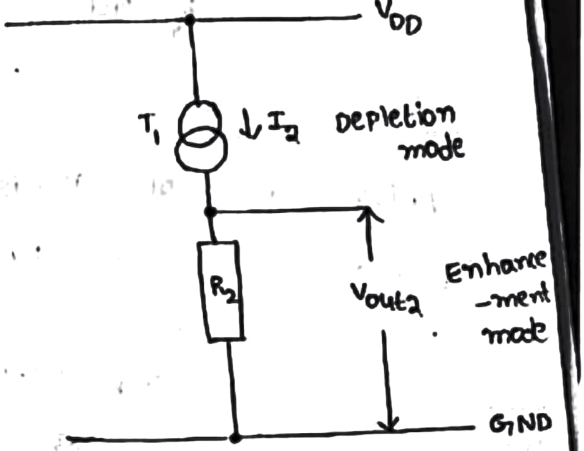
$$V_{in2} = V_{DD} - V_{tp}$$

when Input to Inverter 1 is V_{DD}, its Pull down transistor conducting with a low voltage across it. At the same time, the Pull-up transistor T_1 is in saturation and represented as a current source.

The equivalent circuits of Inverters 1 & Inverter 2 as shown in fig.



a) Inverter 1 with Input = V_{DD}



b) Inverter 2 with Input $= V_{DD} - V_{tp}$

pull down transistor,

$$I_{ds} = K \frac{W_{pd1}}{L_{pd1}} \left[(V_{DD} - V_t) - \frac{V_{ds}}{2} \right] V_{ds_1}$$

[∵ Non saturation region $V_{gs} = V_{DD}$]

$$R_1 = \frac{V_{DS1}}{I_{DS}}$$

$$= \frac{\cancel{V_{DS1}}}{K \cdot \frac{W_{pd1}}{L_{pd1}} \left[(V_{DD} - V_t) - \frac{V_{ds}}{2} \right] \cancel{V_{ds_1}}}$$

$$= \frac{1}{K} \cdot \frac{L_{pd1}}{W_{pd1}} \frac{1}{\left[(V_{DD} - V_t) - \frac{V_{ds}}{2} \right]}$$

V_{ds} is small and hence it can be neglected

$$R_1 = \frac{1}{K} Z_{pd1} \frac{1}{(V_{DD} - V_t)} \quad \text{—} ①$$

For pull up transistor,

$$I_1 = I_{ds} = K \frac{W_{pu1}}{L_{pu1}} \left[\frac{(V_{gs} - V_{td})^2}{2} \right]$$

consider $V_{gs} = 0V$

$$I_1 = K \cdot \frac{W_{pu1}}{L_{pu1}} \left[\frac{(-V_{td})^2}{2} \right]$$

The output of Inverter 1, $V_{out1} = I_1 R_1$

$$V_{out1} = K \cdot \frac{W_{pu1}}{L_{pu1}} \left[\frac{(-V_{td})^2}{2} \right] \cdot \frac{1}{\cancel{K}} Z_{pd1} \frac{1}{(V_{DD} - V_t)}$$

$$= \frac{1}{Z_{pu1}} \left[\frac{(-V_{td})^2}{2} \right] \cdot Z_{pd1} \frac{1}{(V_{DD} - V_t)}$$

$$= \frac{Z_{pd1}}{Z_{pu1}} \cdot \frac{(-V_{td})^2}{2(V_{DD} - V_t)} \quad — ②$$

consider Inverter 2 with Input as $V_{DD} - V_{tp}$

similarly, $R_2 = \frac{1}{K} Z_{pd2} \left[\frac{1}{(V_{DD} - V_{tp}) - V_t} \right]$ — ③

$$I_2 = K \cdot \frac{1}{Z_{pu2}} \left[\frac{(-V_{td})^2}{2} \right] \quad — ④$$

The output of Inverter 2, $V_{out2} = I_2 R_2$

$$V_{out2} = K \cdot \frac{1}{Z_{pu2}} \left[\frac{(-V_{td})^2}{2} \right] \cdot \frac{1}{\cancel{K}} Z_{pd2} \left[\frac{1}{(V_{DD} - V_{tp}) - V_t} \right]$$

$$= \frac{Z_{pd2}}{Z_{pu2}} \cdot \frac{1}{(V_{DD} - V_{tp}) - V_t} \cdot \frac{(-V_{td})^2}{2} \quad — ⑤$$

The output of Inverter 2 is same as the Inverter 1.

$$V_{out.1} = V_{out\,2}$$

$$I_1 R_1 = I_2 R_2$$

$$\frac{Z_{pd1}}{Z_{pu1}} \cdot \frac{(-V_{td})^2}{2(V_{DD}-V_t)} = \frac{Z_{pd2}}{Z_{pu2}} \cdot \frac{1}{(V_{DD}-V_{tp})-V_t} \cdot \frac{(-V_{td})^2}{2}$$

$$\frac{Z_{pd1}}{Z_{pu1}} \cdot \frac{\cancel{(-V_{td})^2}}{\cancel{2}} \cdot \frac{1}{V_{DD}-V_t} = \frac{Z_{pd2}}{Z_{pu2}} \cdot \frac{1}{(V_{DD}-V_{tp})-V_t} \cdot \frac{\cancel{(-V_{td})^2}}{\cancel{2}}$$

$$\frac{Z_{pu2}}{Z_{pd2}} = \frac{Z_{pu1}}{Z_{pd1}} \cdot \frac{V_{DD}-V_t}{(V_{DD}-V_{tp})-V_t} \quad \text{—⑥}$$

substitute $V_t = 0.2V_{DD}$, $V_{tp} = 0.3V_{DD}$ in eq ⑥,

$$\frac{Z_{pu2}}{Z_{pd2}} = \frac{Z_{pu1}}{Z_{pd1}} \cdot \frac{V_{DD}-0.2V_{DD}}{(V_{DD}-0.3V_{DD})-0.2V_{DD}}$$

$$= \frac{Z_{pu1}}{Z_{pd1}}\left[\frac{0.8\cancel{V_{DD}}}{0.5\cancel{V_{DD}}}\right] \cong 2 \cdot \frac{Z_{pu1}}{Z_{pd1}} \qquad \left[\because \frac{0.8}{0.5} \approx \frac{1}{2}\right]$$

$$= \frac{Z_{pu1}}{Z_{pd1}}\left[\frac{8}{5}\right] \qquad \cong 2 \times \frac{4}{1} = \frac{8}{1}$$

$$= \frac{4}{1}\left[\frac{8}{5}\right] \qquad \left[\because \frac{Z_{pu1}}{Z_{pd1}} = \frac{4}{1}\right]$$

$$\boxed{\frac{Z_{pu2}}{Z_{pd2}} \simeq \frac{8}{1}}$$

Therefore an Inverter driven through one (or) more pass transistors has a Z_{pu}/Z_{pd} is greater than 8/1.

Alternative forms of pull-up :-

1. Load resistance (R_L) :-

Logic '0' is given at V_{in}, no current flows from V_{DD} to V_{SS}.

If logic '1' is given at V_{in}, there is a current flows from V_{DD} to V_{SS}.

This arrangement is not used because of the large space requirements of resistor produced in silicon substrate.

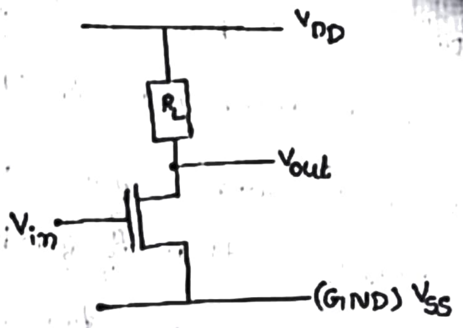


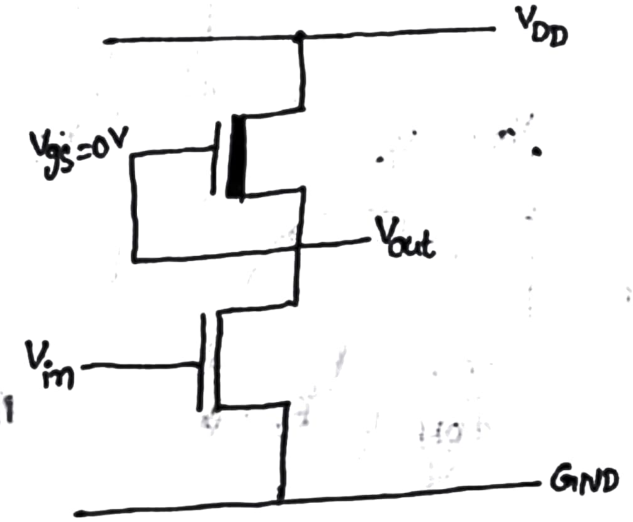
Fig :- Resistor pull-up

2. NMOS depletion mode transistor pull-up :-

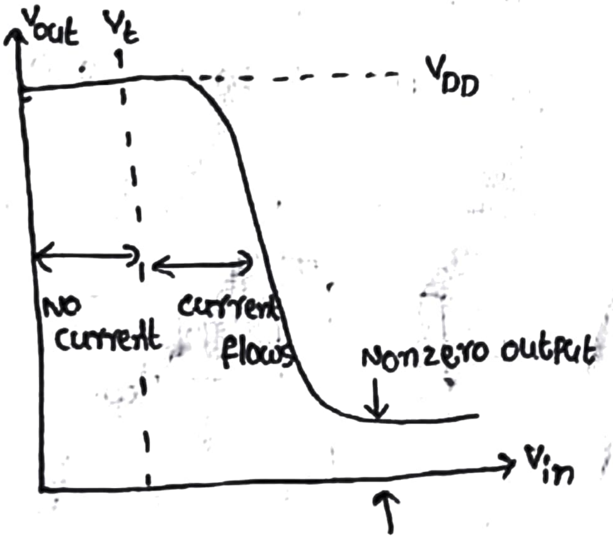
a) Dissipation is high since rail to rail current flows when V_{in} = Logical 1

b) Switching of output from 1 to 0 begins when V_{in} exceeds V_t of pull down device.

c) when switching the output from 1 to 0, ~~be~~ the pull up device is non-saturated initially, this process presents lower resistance through which to charge capacitive loads.



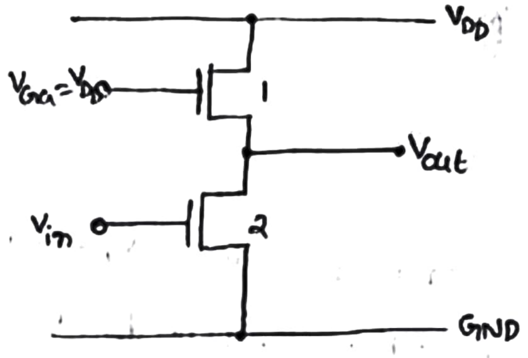
a) circuit



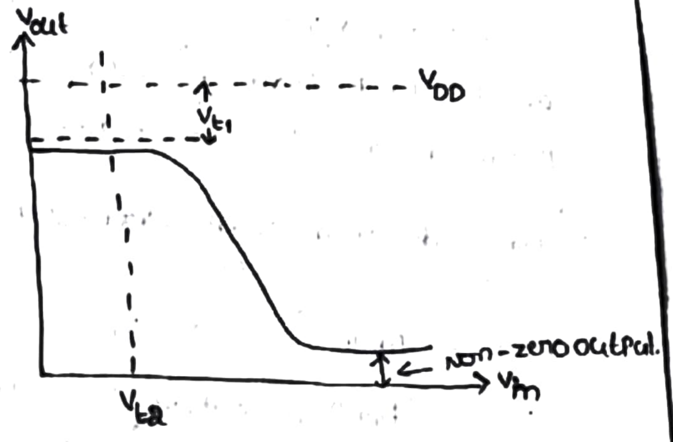
b) Transfer characteristics

3. NMOS enhancement mode pull-up :-

* Dissipation is high since current flows when V_{in} = logical 1
* V_{out} can never reach V_{DD} (Logical 1) if $V_{GG} = V_{DD}$ as is normally the case
* V_{GG} may be derived from a switching source, for example, one phase of a clock, so that dissipation can be greatly reduced. If V_{GG} is higher than V_{DD}, an extra rail is required



a) circuit

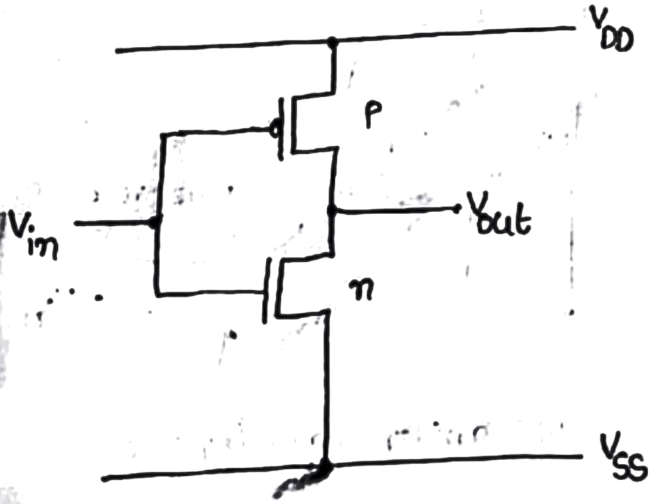


b) transfer characteristics

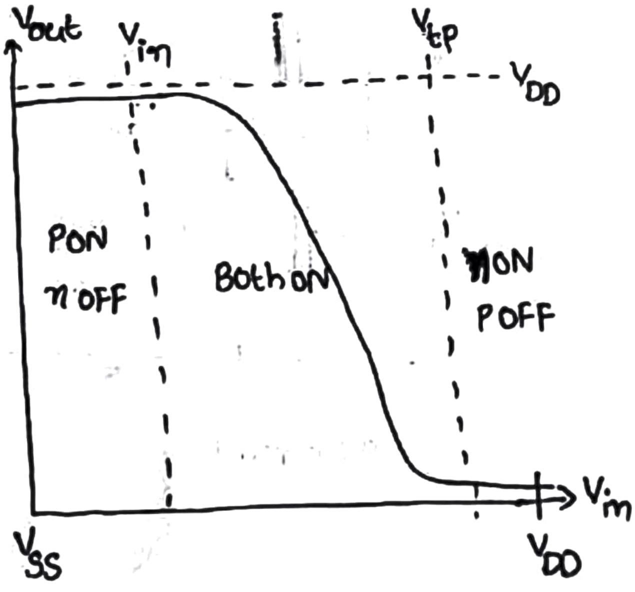
4. complementary transistor pull-up :-

→ No ~~output~~ current flow either for logical 1 (or) logical 0 Inputs.

→ For similar dimension devices, the n-channel device is faster than p-channel device.



a) circuit



b) Transfer characteristics

CMOS Inverter :-

The CMOS Inverter can be designed by using PMOS and NMOS transistors.

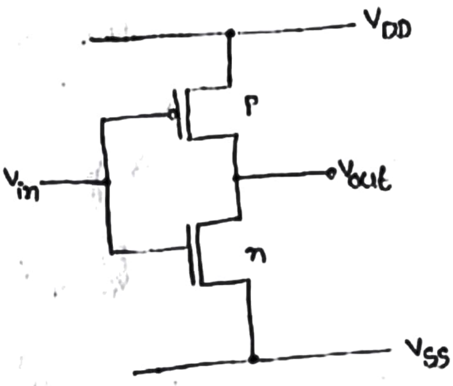


Fig :- CMOS Inverter

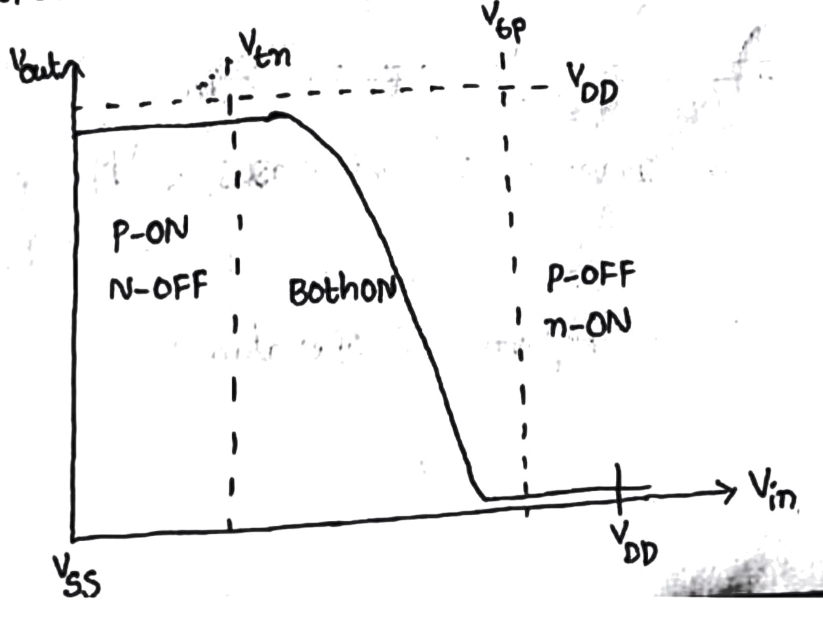
The operation of the circuit as follows:

→ when the Input voltage $V_{in}=0$, the gate of the p-channel transistor is at V_{DD} below the source Potential, i.e $V_{GS}=V_{DD}$. This turns on this transistor, capacitor charged upto V_{DD}.

→ ~~as~~ The transistor turned off, No current flows through n-channel transistor since $V_{GS}=0$.

→ If the Input voltage raised to threshold voltage level of the n-channel transistor and then to V_{DD}, n-channel transistor will conduct and p-channel transistor gets turned off, discharging the load capacitance to ground potential.

The transfer characteristics as shown in fig. below :



The current / voltage relationships for the mos transistor in saturation region is given by,

$$I_{ds} = k\frac{w}{L}\left[\frac{(V_{gs}-V_t)^2}{2}\right] \quad \text{—}①$$

where $k = \frac{\varepsilon_0 \varepsilon_{ins} \mu}{D}$

$$I_{ds} = \frac{\beta}{2}(V_{gs}-V_t)^2 \text{ —}② \left[\because \beta = \frac{kw}{L}\right]$$

'β' applicable to both NMOS and PMOS transistors,

$$\beta_p = \frac{\varepsilon_0 \varepsilon_{ins} \mu_p}{D}\frac{w_p}{L_p} \quad \text{—}③$$

$$\beta_n = \frac{\varepsilon_0 \varepsilon_{ins} \mu_n}{D}\frac{w_n}{L_n} \quad \text{—}④$$

where w_p, w_n, L_p, L_n are Pand n-transistor dimensions.

μ_p, μ_n are hole and electron mobilities.

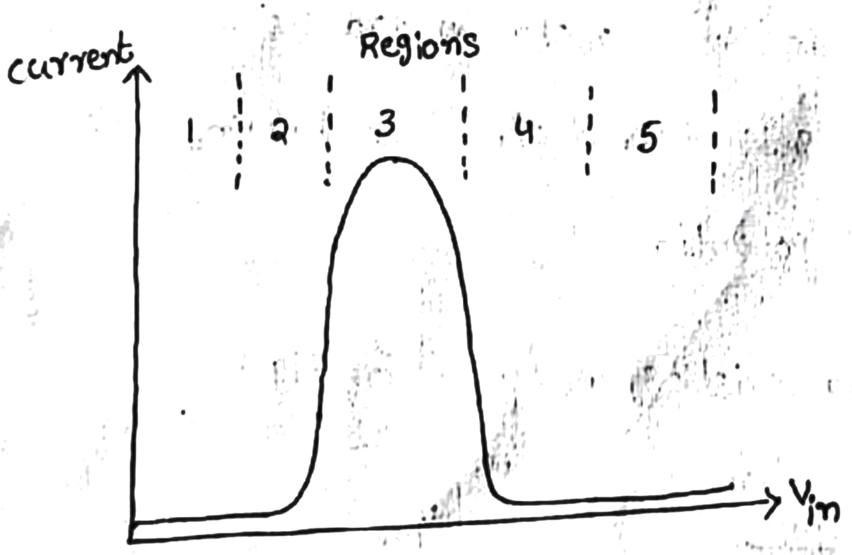


Fig:- cmos Inverter current versus V_{in}

It has Five distinct regions of operations.

Region 1 :-

V_{in} = Logic 0 = 0V

PMOS is ON and NMOS is OFF. Hence no current flows through the Inverter circuit and output directly connected to V_{DD}.

Region 5 :- V_{in} = Logic 1 = V_{DD}

N-transistor ON and P-transistor OFF. In this condition again no current flows through the circuit.

Region 2 :- $V_{in} > V_t$ (The Input Voltage is Increased above the threshold voltage)

N-transistor conducts and has a very large difference between drain and Source and is in saturation.

P-transistor also conducts, with only small voltage difference between drain and source it operates in unsaturated resistive region.

Region 4 :-

conditions are same as in region 2 but with the roles of P-and n-transistors reversed. That is, P-transistor has a large voltage across it while the n-transistor has a small voltage across it. The drain current in both the regions 2 and 4 is Small.

egion-3 :-

Large current flows in region 3. In this region, both the devices are in saturation.

Since, the two transistors are in series, the current through them is same, we can write

$$I_{dsp} = -I_{dsn}$$

$$I_{ds} = \frac{k\omega}{L} \frac{[V_{gs}-V_t]^2}{2} \quad \text{(saturation)}$$

$$I_{ds} = k\frac{\omega}{L} \left[(V_{gs}-V_t)V_{ds} - \frac{V_{ds}^2}{2}\right] \quad \text{[resistive)}$$

w.k.T $$I_{ds} = \frac{k\omega}{L} \frac{[V_{gs}-V_t]^2}{2}$$

$$I_{dsp} = \frac{k\omega_p}{L_p} \frac{(V_{gs}-V_t)^2}{2}$$

$$= \frac{k\omega_p}{L_p} \frac{(V_{in}-V_{DD}-V_{tp})^2}{2} \qquad \left[\because V_{gs} = V_{in}-V_{DD} \quad V_t = V_{tp}\right]$$

$$I_{dsn} = \frac{k\omega_n}{L_n} \left(\frac{V_{in}-V_{tn}}{2}\right)^2$$

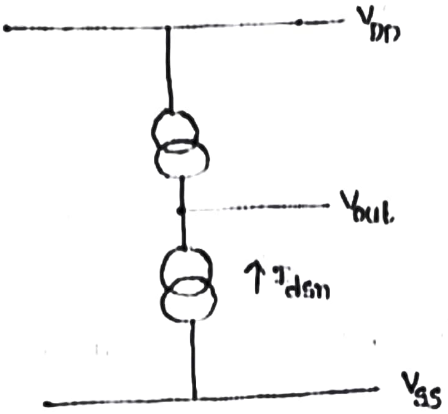
$$I_{dsp} = \frac{\beta_p}{2}(V_{in}-V_{DD}-V_{tp})^2 \quad \text{— ①}$$

$$I_{dsn} = \frac{\beta_n}{2}(V_{in}-V_{tn})^2 \quad \text{— ②}$$

Equating ① & ② we get

$$V_{in} = \frac{V_{DD}+V_{tp}+V_{tn}\left(\frac{\beta_n}{\beta_p}\right)^{1/2}}{1+(\beta_n/\beta_p)^{1/2}} \quad \text{— ③}$$

In region 3, both transistors are in saturation. they act as current source, the equivalent circuit, in this region as shown in fig.



This region 3 is very unstable and change over from one logic level to other.

If $V_{in} = -V_{tp}$, $\beta_p = \beta_n$, equation reduces to $V_{in} = 0.5\,V_{DD}$

At this point two 'β' factors will be equal.

$\beta_n = \beta_p$, the device geometries should satisfy

$$\frac{\mu_n W_n}{L_n} = \frac{\mu_p W_p}{L_p}$$

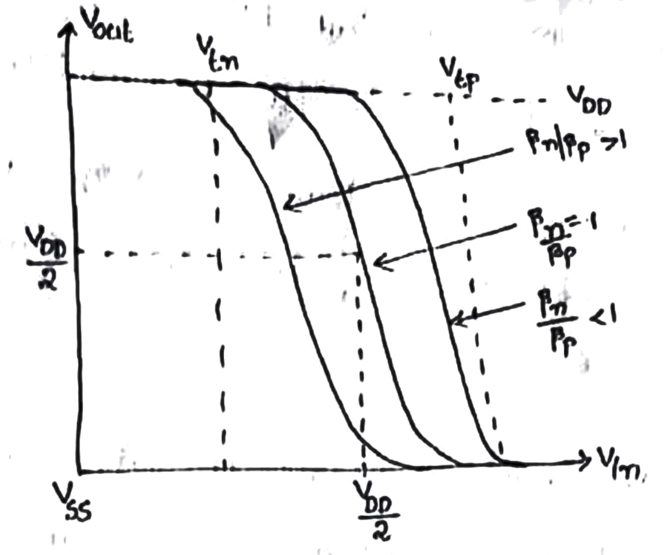
The mobilities of electrons and holes are unequal.

It is necessary that the width to length ratio (W/L) of P-device be two to three times that of n-device i.e

$$\frac{W_p}{L_p} = 2 \cdot \frac{W_n}{L_n}$$

By keeping minimum size geometries for both p-and n-devices, effect 'β' ratio is minimized.

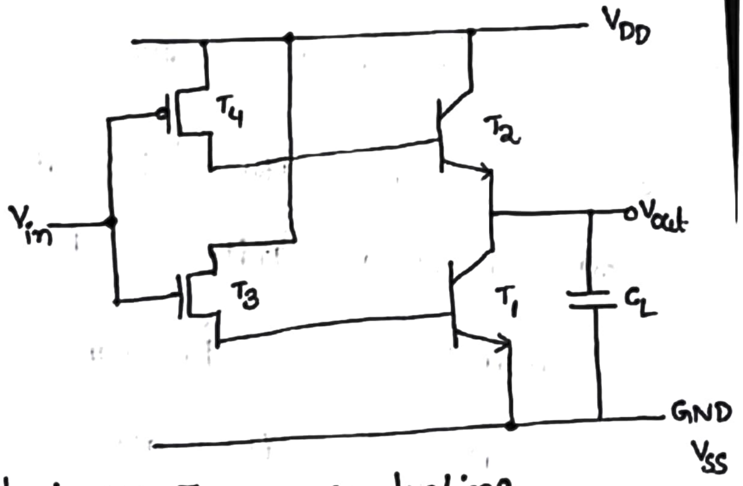
The transfer characteristics as shown in fig.



Bicmos Inverter:-

The simple Bicmos Inverter as shown in fig. below

The Inverter circuit consists of two bipolar transistors T_1, T_2 and mos devices are enhancement mode.



operation

* $V_{in} = 0$ i.e Logic 0, T_3 is off which keeps T_1 non-conducting. T_4 is ON and supplies base current to T_2 which conducts and act as a current source to charge the load C_L toward +5v (V_{DD}). The output V_{out} goes +5V less the base to emitter drop V_{BE} of T_2.

* V_{in} = Logic 1 (+5V), T_4 is OFF so that T_2 will be non-conducting. T_3 is ON and supplies current to base of T_1 which conducts

and act as a current sink to the load C_L which discharges through it to '0' volts (GND). The V_{out} falls to 0 volts plus the saturation voltage $V_{CE(sat)}$ between collector and emitter of T_1.

* charging and discharging of the load C_L is very fast because transistors T_1, T_2 present low Impedances when turned on into saturation.

→ The output logic levels will approximate the rail voltages since $V_{CE(sat)}$ is quite small and V_{BE} equals 0.7 volts approximately.

→ The Inverter offers a low output Impedance and a high Input Impedance. It occupies a relatively small area but has a high current drive capability.

→ There is a constant D.C Path between the rails through T_3 and which allows a significant static current flow whenever V_{in} = Logic

There is another Problem, that there is no discharge path for current from the base of either npn transistor when it is turned off. This effects the speed of action of the circuit.

→ The problem of the D.C Path ~~between~~ through T_1, T_3 is eliminated in an Improved Inverter circuit. as shown in fig.

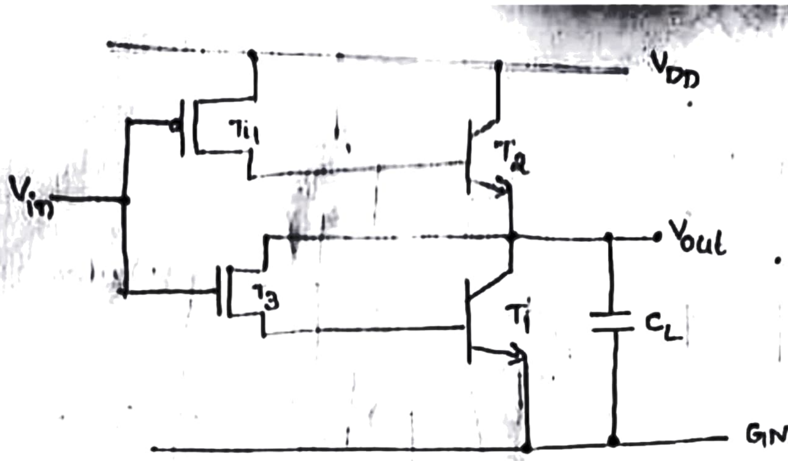
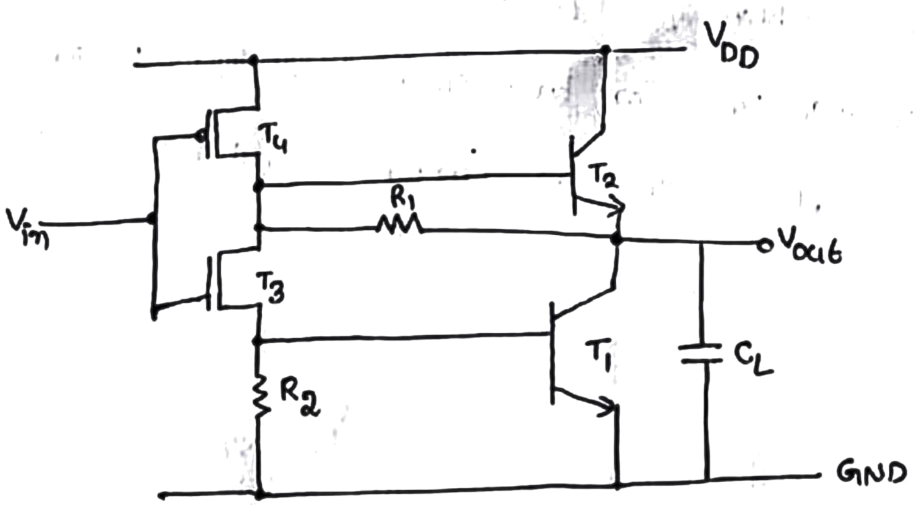


Fig :- Bicmos Inverter with no static current flow

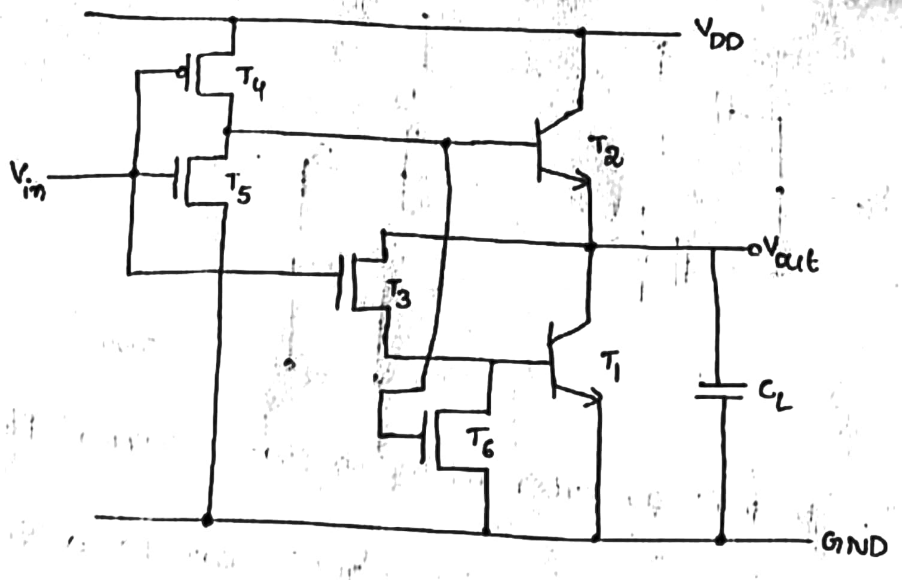
→ Further Improvement in Inverter circuit can be achieved using resistors as shown in fig.



The resistors provide an Improved swing of output voltage when either bipolar transistor is off.

They also provide discharge paths for the base currents during turn-off.

An Improved Bicmos Inverter using mos transistors for base current discharge as shown in fig.



To Turn ON transistors T_5, T_6 when T_2, T_1 respectively are being turned off. That is when T_2 is to be turned off, T_5 gets turned on and provides discharge path for base current of T_2. Bicmos Inverters are more suitable where high load current sinking & sourcing is required.

Latch-up in CMOS circuits :-

→ A problem which is Inherent in the P-well, n-well Process is due to large no. of Junctions, which are formed in these structures, consequent presence of Parasitic transistors and diodes.

→ Latch-up is, a condition in which Parasitic components give rise to the establishment of low resistance conducting Path between V_{DD}, V_{SS}.

→ Latch-up may be Induced by glitches on supply rails (or) Incident radiations.

consider, Parasitic components associated with the P-well structure,

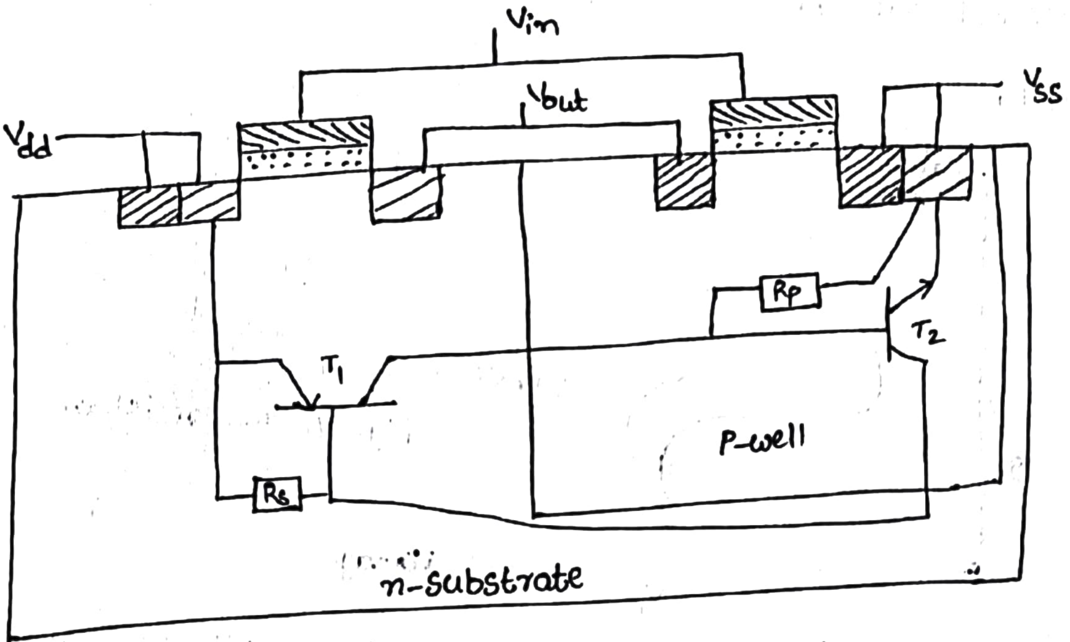


Fig :- Latch up effect in P-well structure

→ There are, in effect, 2 transistors, 2-resistances which form a path between V_{DD}, V_{SS}.

→ If sufficient substrate current flows to generate enough voltage across R_S to turn on, transistor T_1, this will then draw current through R_P

→ If voltage developed is sufficient, T_2 also turn ON, establishing a self sustaining low resistance path between supply rails.

→ If 2 current gains of 2 transistors $\beta_1 \times \beta_2 > 1$, Latch up may occur.

→ Equivalent circuit as

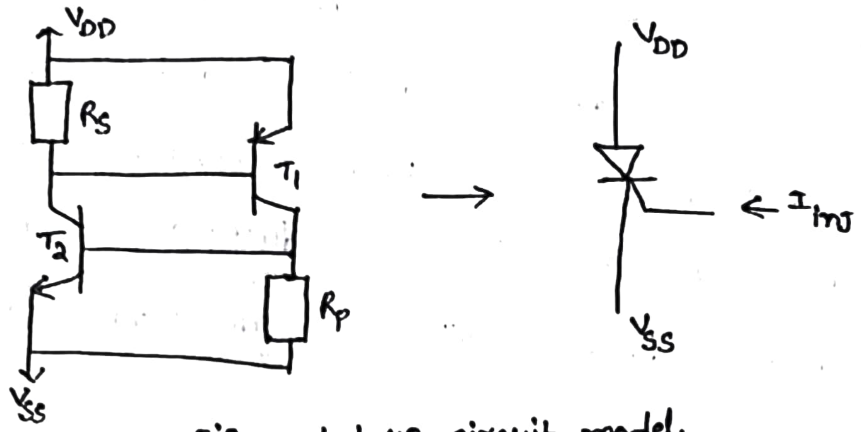
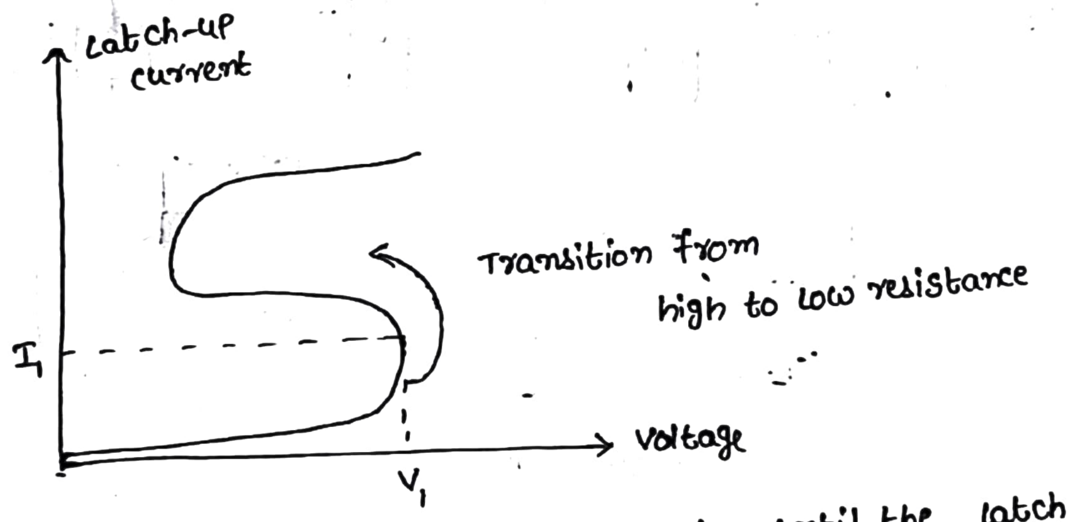


Fig:- Latch up circuit model.

→ with no Injected current, Parasitic transistors exhibit high resistance, but sufficient substrate current flow will cause switching to low resistance.



→ once Latched up, this condition maintain until the latch-up current drops below I_1.

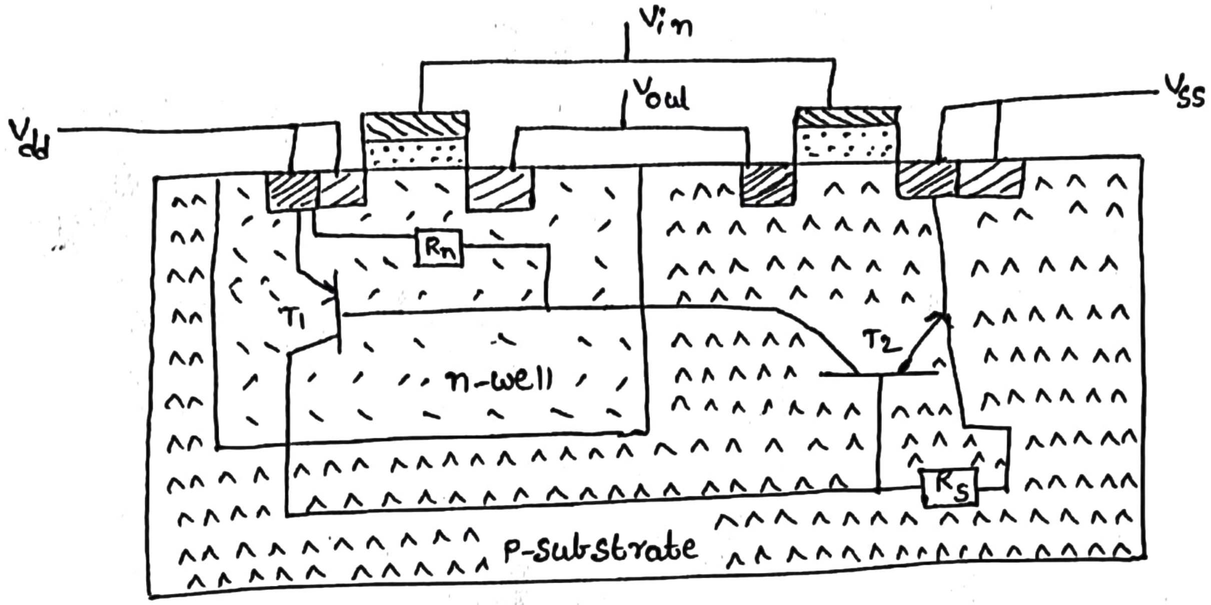
Remedies for Latch-up Problem Include,

* Increase in substrate doping levels with a consequent drop in value of R_S.

* Reducing R_P by control of fabrication parameters and by ensuring low contact resistance to V_{SS}.

* Introducing guard rings

Latch-up in N-well fabrication as shown in fig.



MOS and Bi-CMOS circuit design Processes

(8)

G. Venkata Hanuman
Asst. Prof

MOS Layers :-

MOS circuits are formed on 4-basic Layers i.e. n-diffusion, P-diffusion, polysilicon, metal.

→ Each layer is Isolated from one another by thick (or) thin Insulating layers (SiO_2).

→ A transistor is formed by Polysilicon and thinox regions cross one another.

→ The basic mos transistor properties can be modified by the use of an Implant with in the thinox region and this is used in nmos circuits to produce depletion mode transistors.

Stick diagrams :-

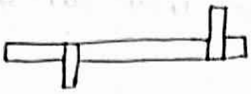
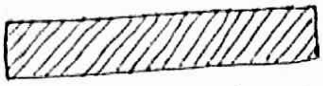
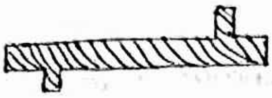
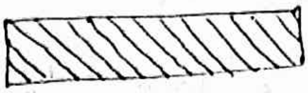
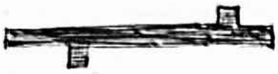
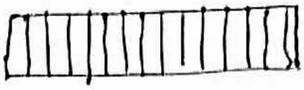
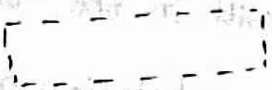
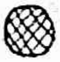
Stick diagram is used to convey layer Information through the use of color code (or) monochrome encoding schemes.

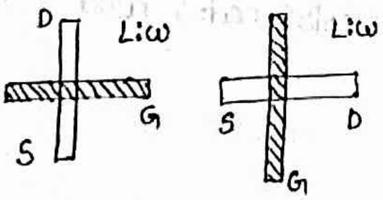
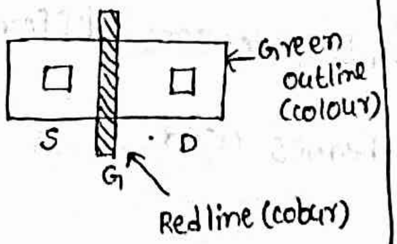
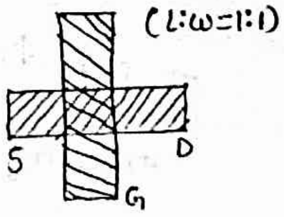
→ For different processes different color codes and different encoding schemes used.

The encoding schemes are

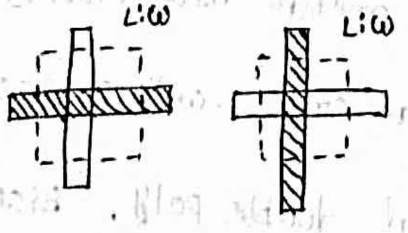
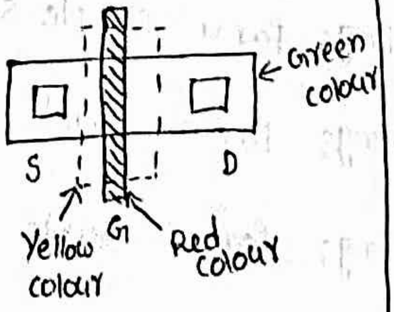
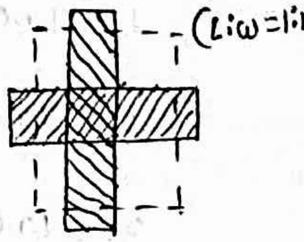
1. Encodings for a simple single metal nmos process
2. Encodings for a Double metal cmos P-well Process
3. Encodings for a double metal double poly, Bicmos n-well Proce

1. Encodings for a simple single metal Nmos process :-

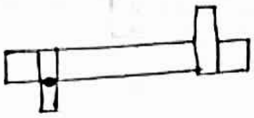
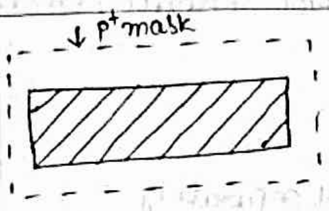
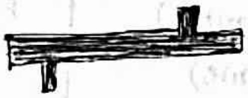
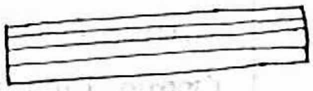
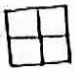
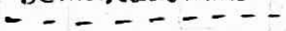
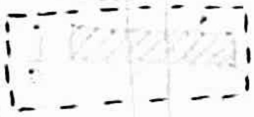
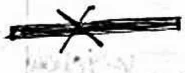
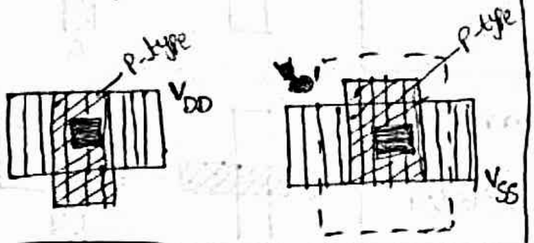
color	Stick encoding	Layers	mask Layout Encoding	CIF Layer (caltech Intermediate form)
	monochrome		monochrome	
Green		n-diffusion (n+ active thinox)	 Thinox = n-diff + transistor channels	ND
Red		Polysilicon		NP
Blue		metal 1		NM
Black	●	contact cut	■	NC
Gray	Not applicable	overglass		NG
Nmos only Yellow		Implant		NI
Nmos only Brown		Burried contact		NB

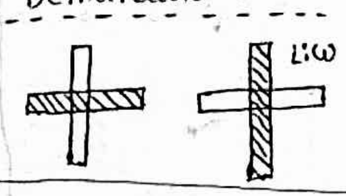
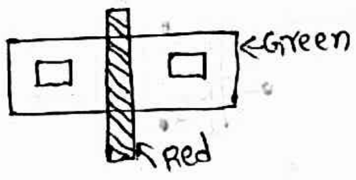
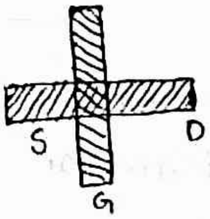
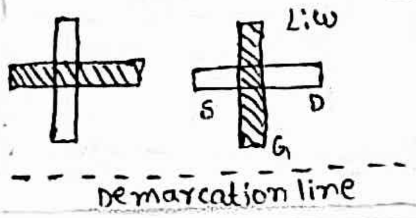
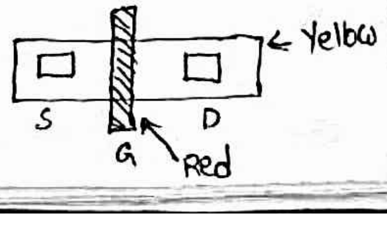
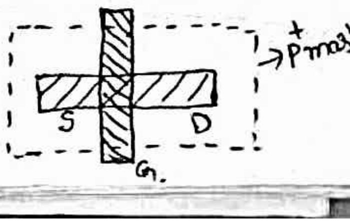
Feature	Feature (Stick) (monochrome)	Feature (symbol) (monochrome)	Feature (mask) (monochrome)
n-type enhancement mode transistor			

Transistor length to width ratio L:w

n-type Depletion mode transistor Nmos only			

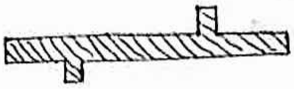
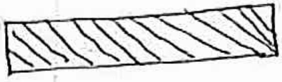
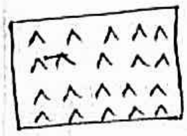
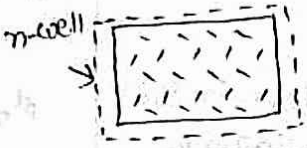
2. Double metal CMOS P-well process

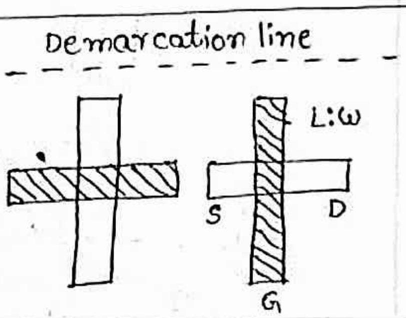
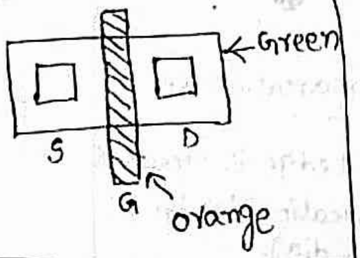
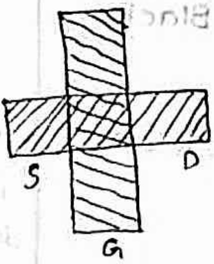
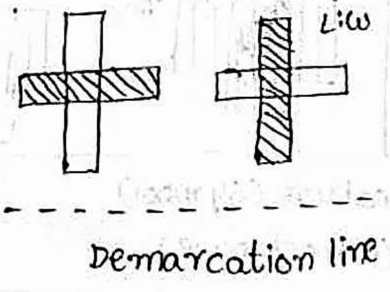
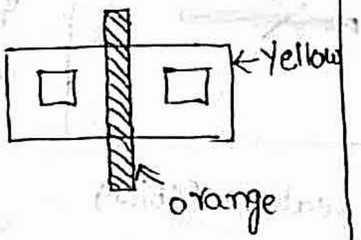
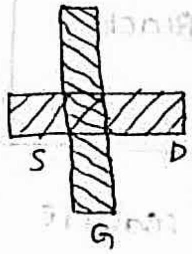
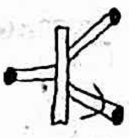
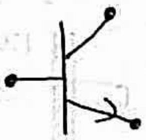
color	Stick encoding	Layers	mask layout encoding	CIF Layer
Green	Monochrome Encoding As above	[n-diffusion n+ active Thinox]	Thinox = n-diff + P-diff + Transistor channels Encoding as above	CAA (or) CNA
Red		Poly silicon		CPF
Blue		metal 1		CMF
Black		contact cut		CC
Gray		over glass		COG
Green in p+(mask) Yellow (stick)		p+-diffusion (p+ active)		CAA (or) CPA
Yellow	Not shown in stick diagram	p+ mask		CPP
Dark blue (or) purple		Metal 2		CMS
Black		VIA		CVA
Brown	Demarcation line  P-well edge is shown as demarcation line in stick diag.	P-well		CPW
Black		V_{DD} (or) V_{SS} contact		CC

Feature	Feature (stick) (monochrome)	Feature (symbol) (monochrome)	Feature (mask) (monochrome)
n-type enhancement mode transistor	Demarcation line 		
p-type enhancement mode transistor	 Demarcation line		

Note :- P-type transistor are placed above and n-type transistors below the demarcation line.

3. Additional encodings for a double metal double Poly , Bicmos n-well Process

color	Stick encoding	Layers	mask layout encoding	CIF Layer
	monochrome		monochrome	
orange		Polysilicon 2		CPS
Pink	Not separately encoded	P-base of Bipolar npn transistor		CBA ~~Not appli -cable~~
Pale Green	Not separately encoded	Burried collector of Bipolar npn transistor		CCA

Feature	Feature (stick) (monochrome)	Feature (symbol) (monochrome)	Feature (mask) (monochrome)
n-type enhancement Poly. 2 transistor			
P-type enhancement Poly. 2 transistor			
npn bipolar transistor			

Design rules and Layout :-

The design rules are the effective Interface between the circuit/system designer and the fabrication engineer.

circuit designers in general want tighter, smaller layouts for Improved performance and decreased silicon area.

on other hand, the process designer wants design rules that result in a controllable and reproducible process.

Lambda-based design rules :-

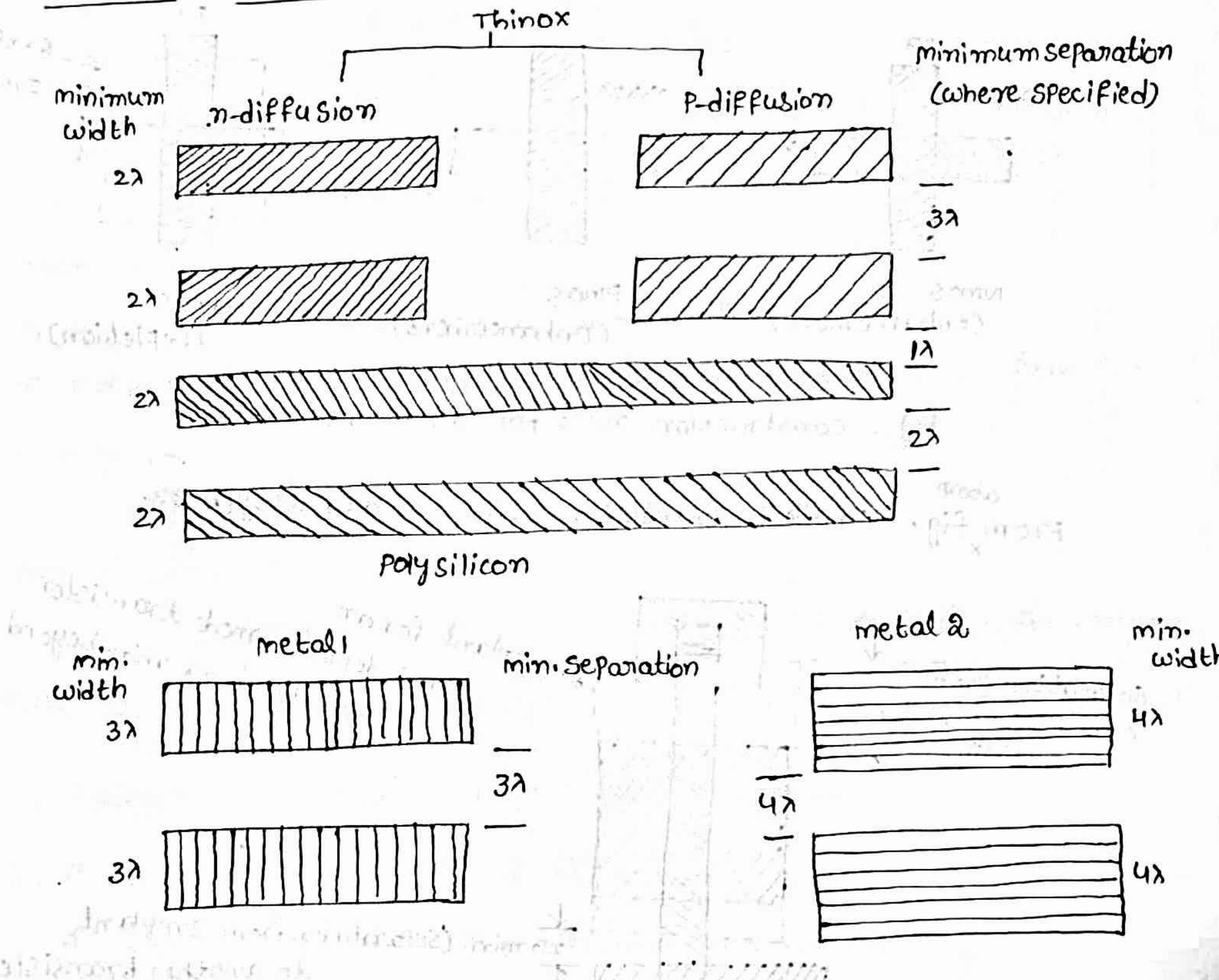


Fig :- Design rules for wires (nmos and cmos)

From above fig,

- → minimum n^+-diffusion and p^+-diffusion width is 2λ
- → minimum spacing between two n^+-diffusion and p^+-diffusion is 3λ
- → For metal 1, minimum width should be 3λ and minimum separation from another metal 1 wire is 3λ.
- → For metal 2, minimum width should be 4λ and minimum separation from another metal 2 wire is 4λ.
- → For Polysilicon wire the minimum width is 2λ and minimum separation from another polysilicon wire is 2λ. This separation is also called as Poly-Poly separation.

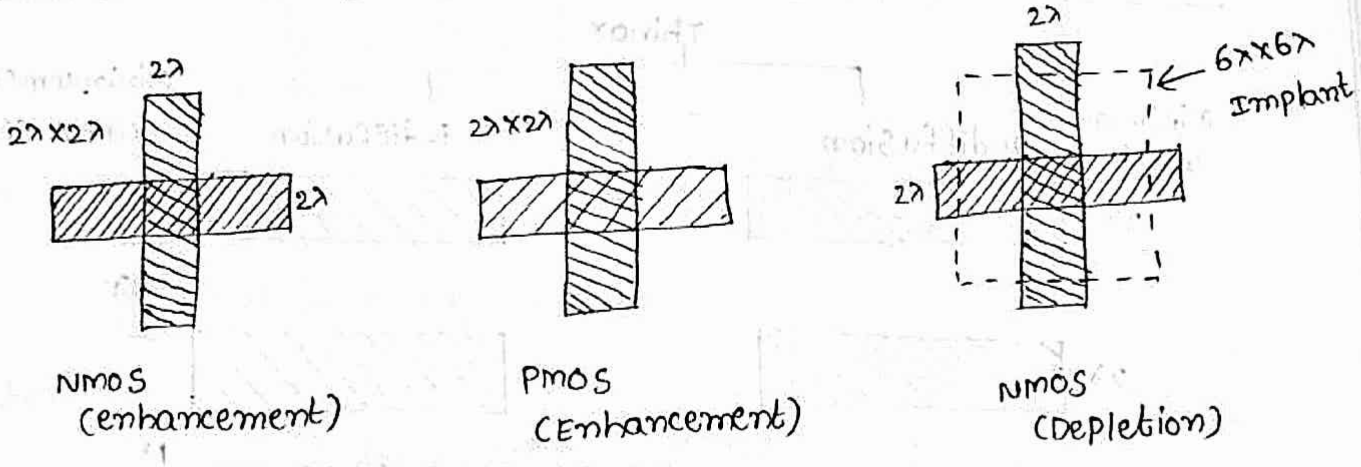
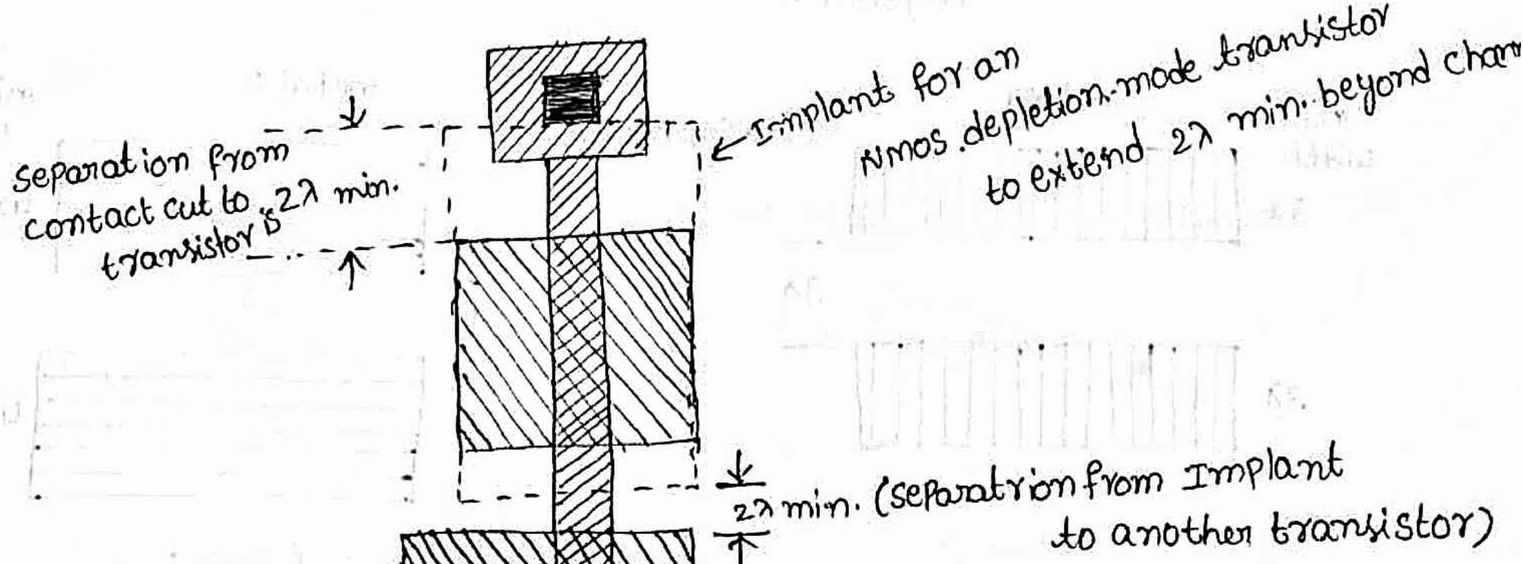


Fig :- construction rules for transistors

From above fig. smallest transistor width 2λ and length 2λ.



separation from contact cut to transistor is 2λ min.
Implant for an NMOS depletion mode transistor to extend 2λ min. beyond chan
2λ min. (separation from Implant to another transistor)

Fig :- Layout of NMOS depletion mode

From above figure,

→ Separation from contact cut to transistor is 2λ

→ Implant for an NMOS depletion mode transistor has extend 2λ from the channel in all directions.

→ separation from Implant to another transistor is 2λ.

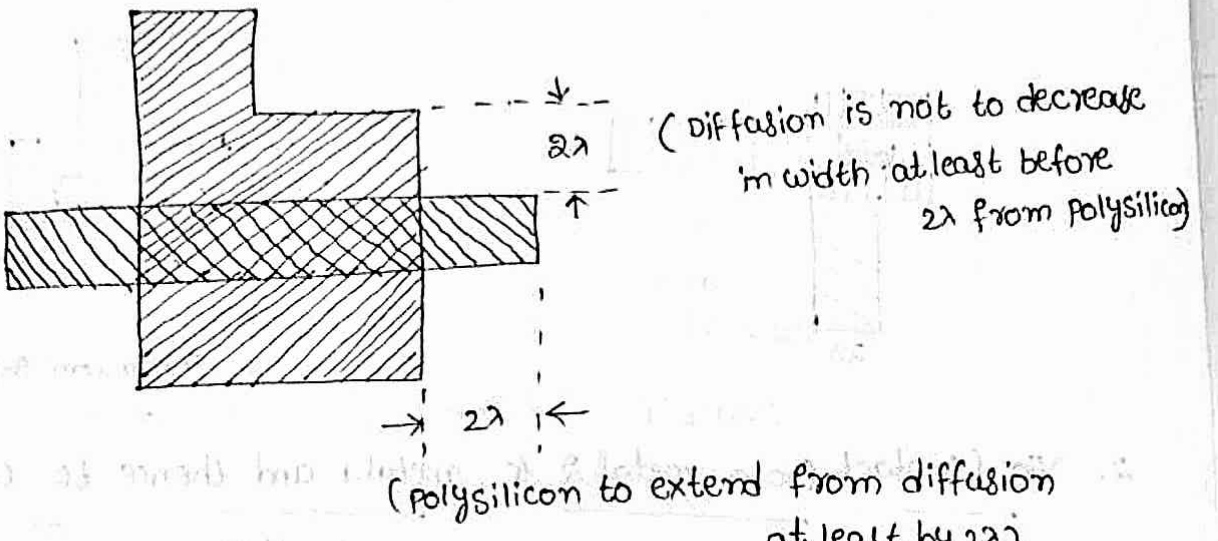

2λ
(Diffusion is not to decrease in width atleast before 2λ from Polysilicon)
2λ
(Polysilicon to extend from diffusion at least by 2λ)

Fig :- Transistor Lay out rules

From above fig,

→ Poly Silicon to extend from diffusion at least by 2λ

→ Diffusion should not decrease in width at least before 2λ from the Polysilicon.

Contact cuts :-

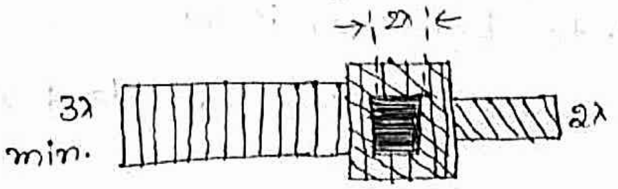
when making contacts between Polysilicon and diffusion in nmos circuits it should be recognized that there are 3 Possible approache

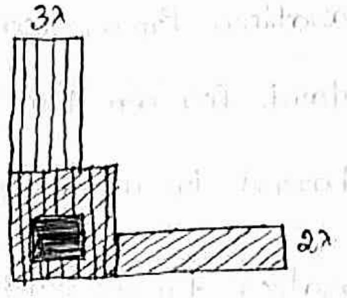
1. Polysilicon to metal and then metal to diffusion

2. Burried contact (Poly to diffusion)

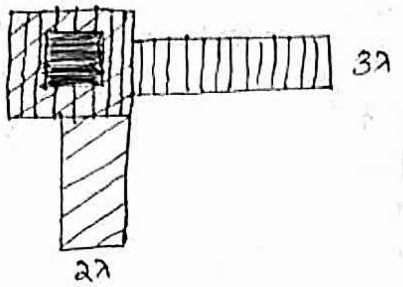
3. Butting contact (Poly to diffusion using metal)

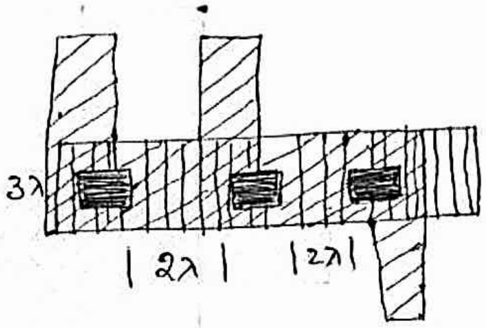
1. metal 1 to Polysilicon (or) to diffusion :-





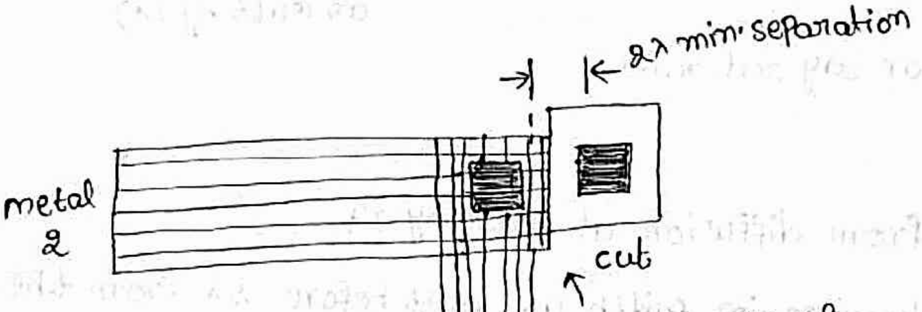
2λ x 2λ cut centered on 4λx4λ superimposed areas of layers to be joined in all cases.





minimum separation multiple cuts

2. Via (contact from metal 2 to metal 1 and thence to other layers)



4λ x4λ area of overlap with 2λx2λ via at center

Via and cut used to connect metal 2 to diffusion →

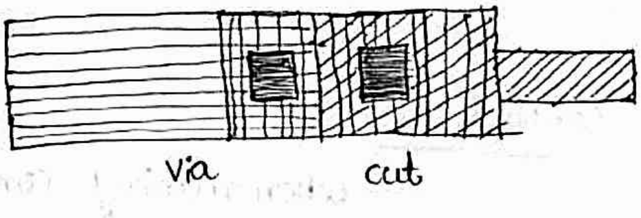


Fig :- Design rules for wires (Nmos and cmos) (or) contacts (nmos & cmos)

2μm double metal, Double poly. CMOS/BiCMOS Rules :-

colors used for CMOS,

n-well : Brown

P-well : Brown

Poly 1 : Red

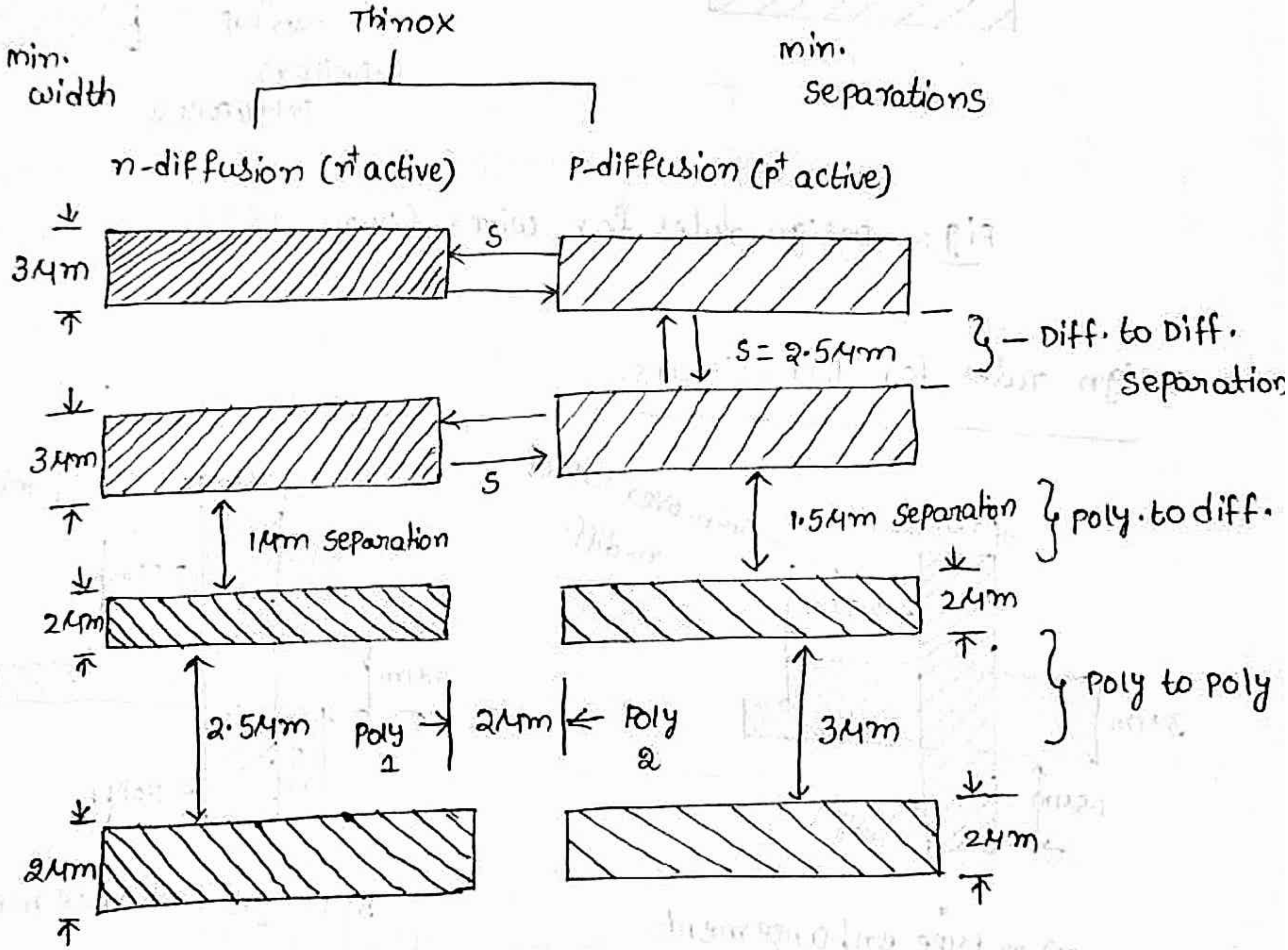
Poly 2 : Green

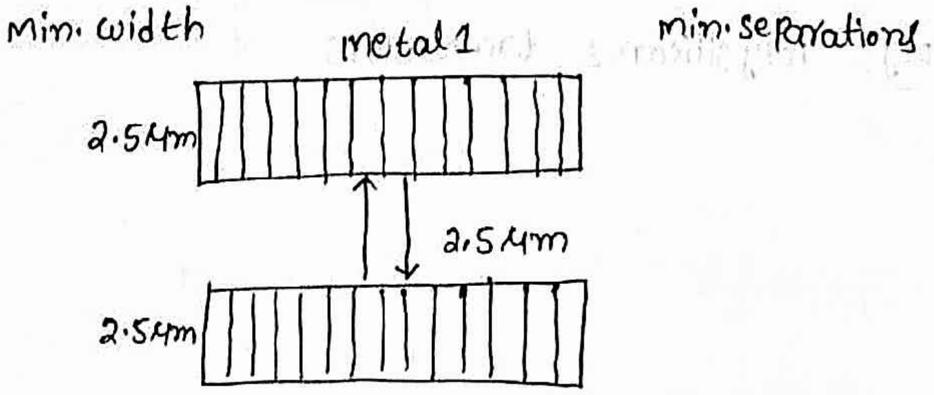
P-diffusion : Yellow

For BiCMOS,

Burried n^+ subcollector – Pale green

P-base – Pink





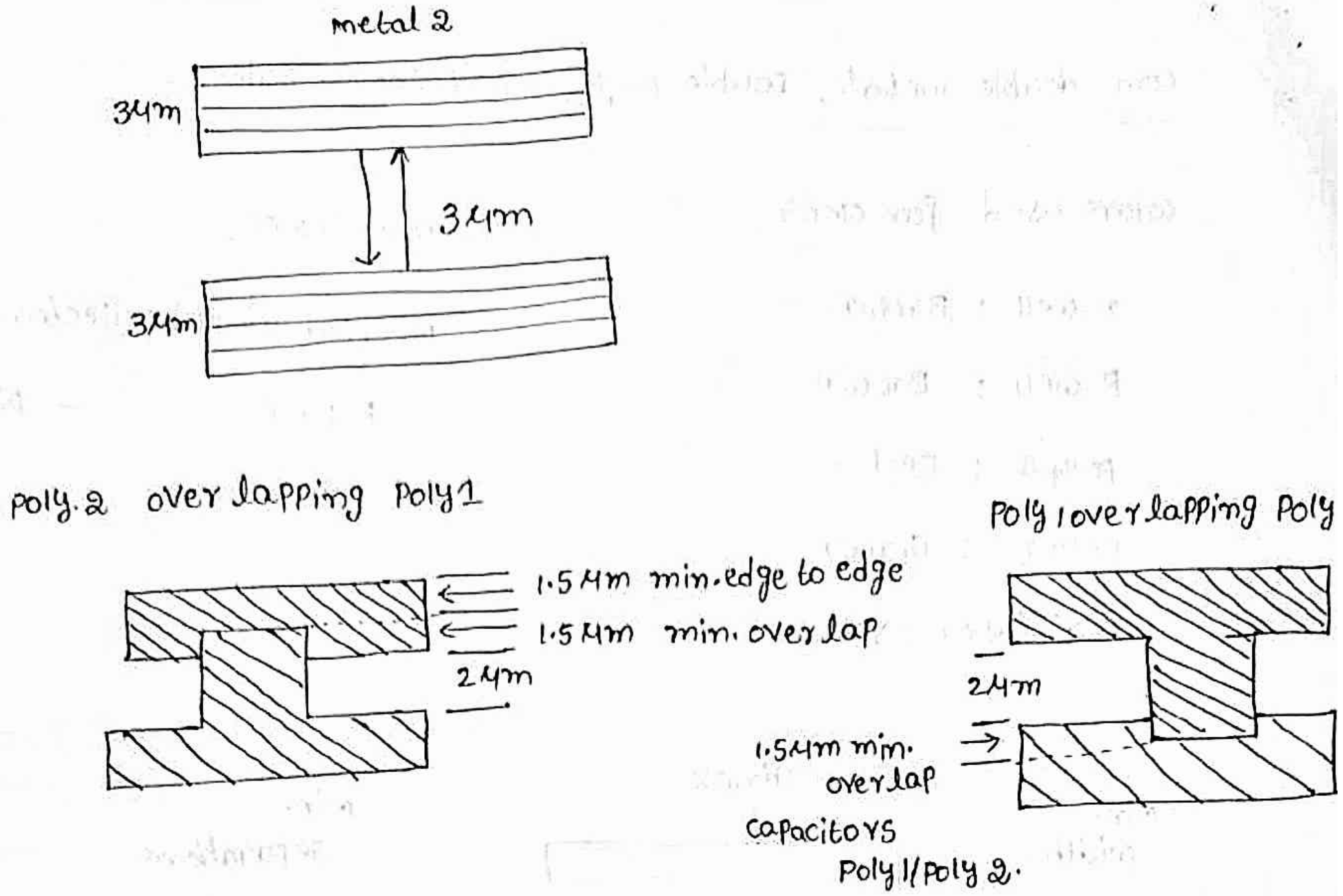
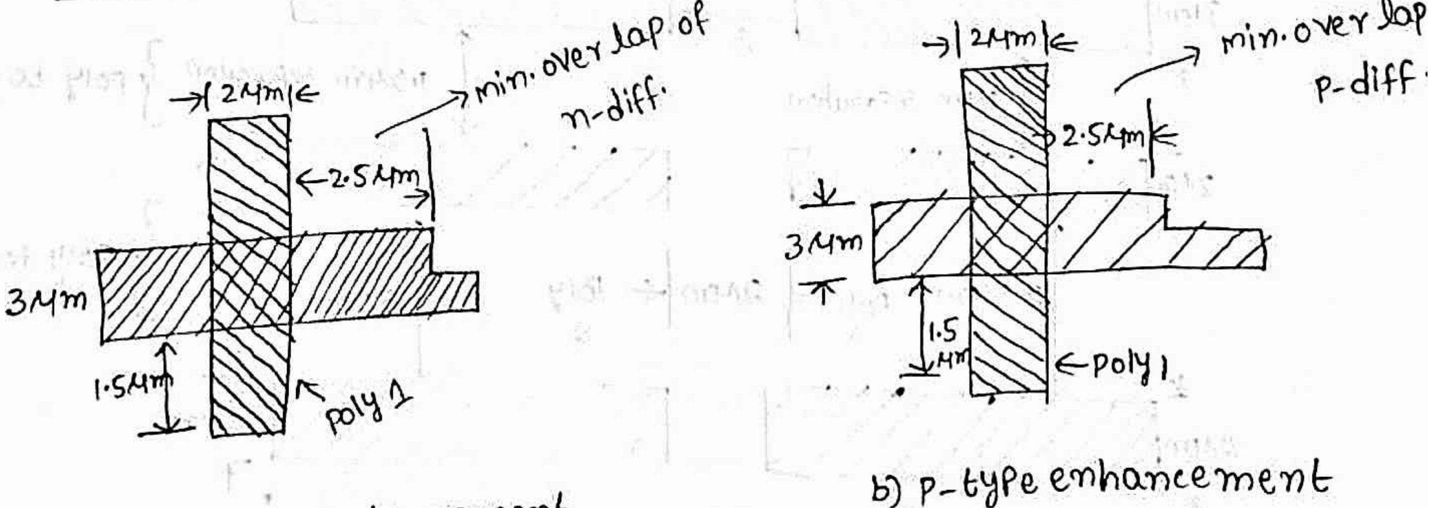


Fig:- Design rules for wires (2μm CMOS)

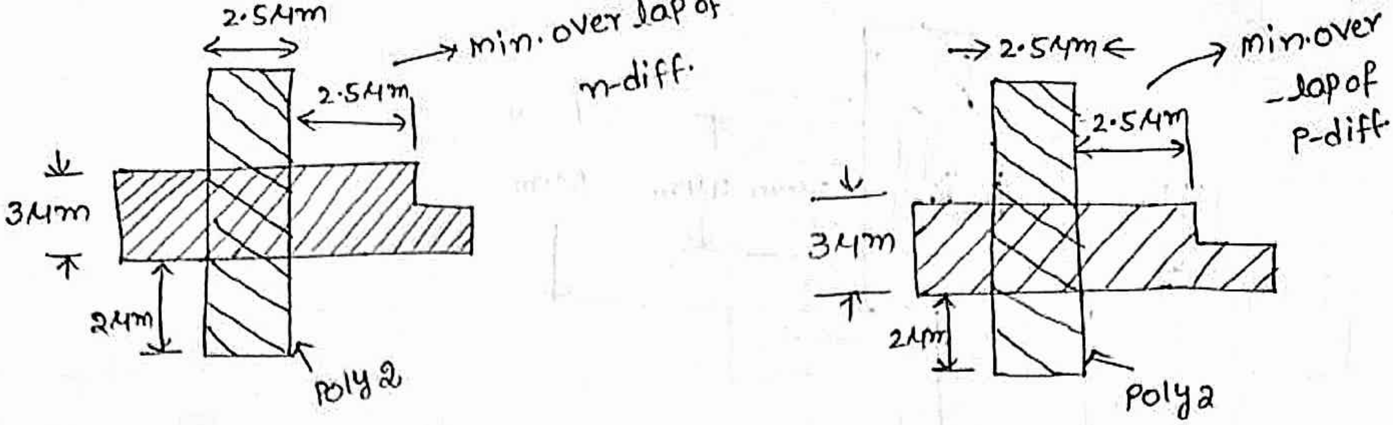
Design rules for transistors:-



a) n-type enhancement

b) P-type enhancement

Fig:- Polysilicon 1 transistors

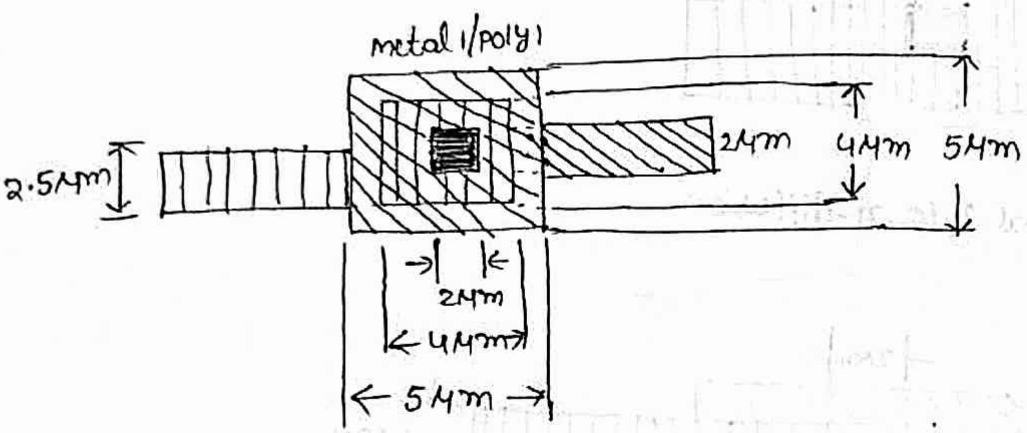


a) n-type enhancement

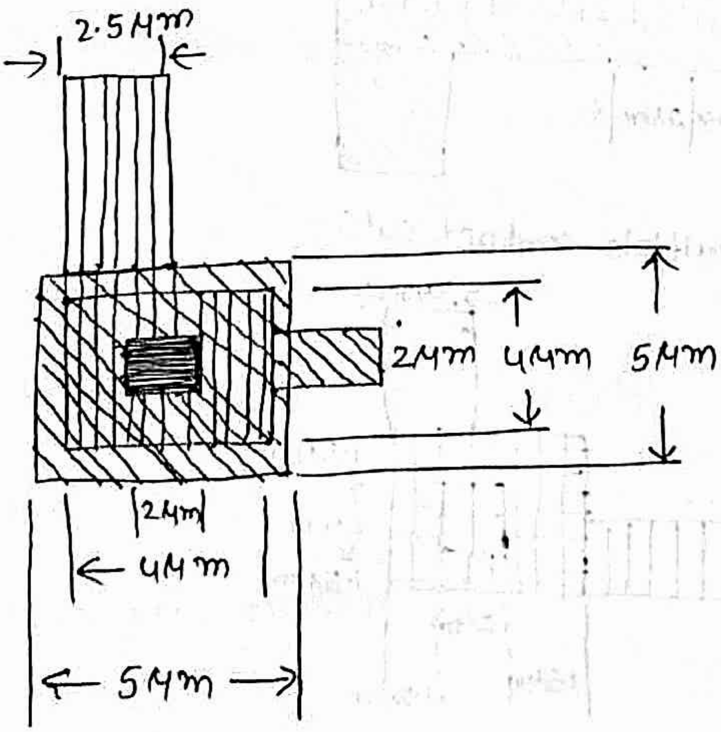
b) p-type enhancement

Fig :- Polysilicon 2 transistors

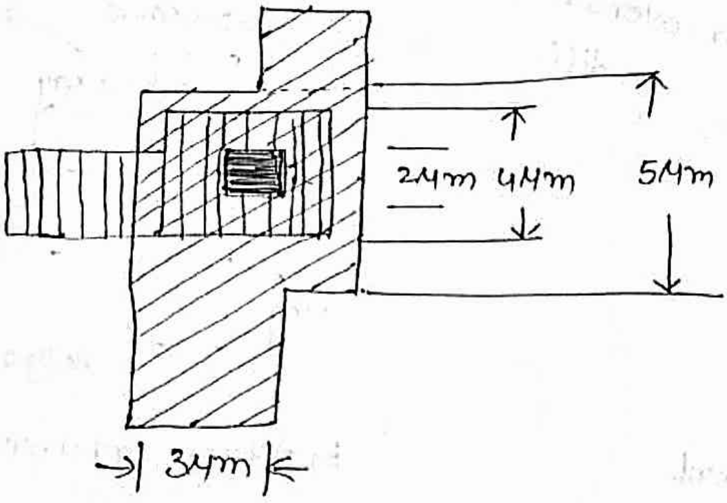
Design rules for contact cuts :-



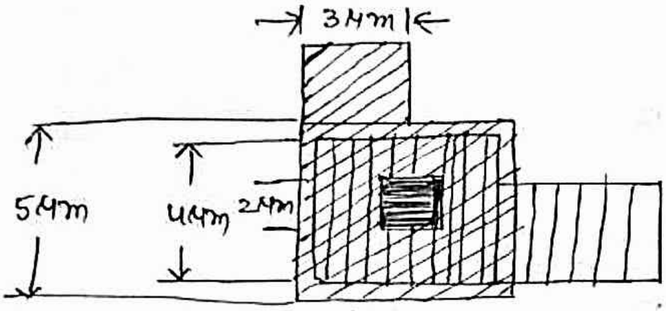
a) metal 1 / poly 1



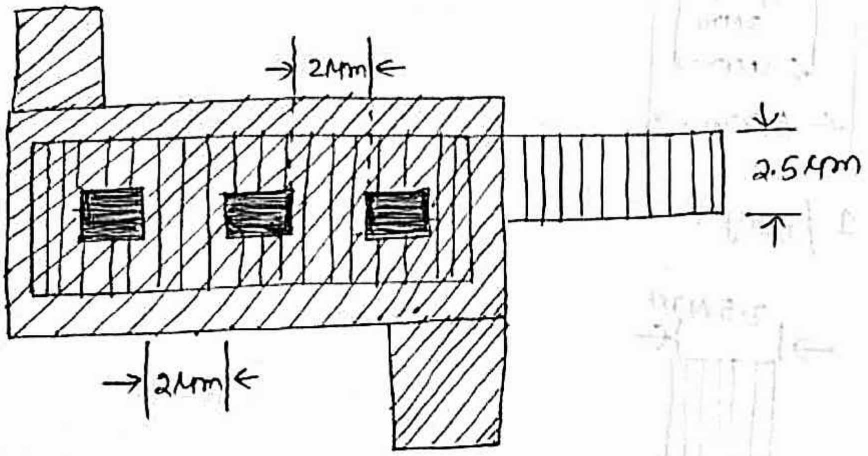
b) metal 1 to poly 2.



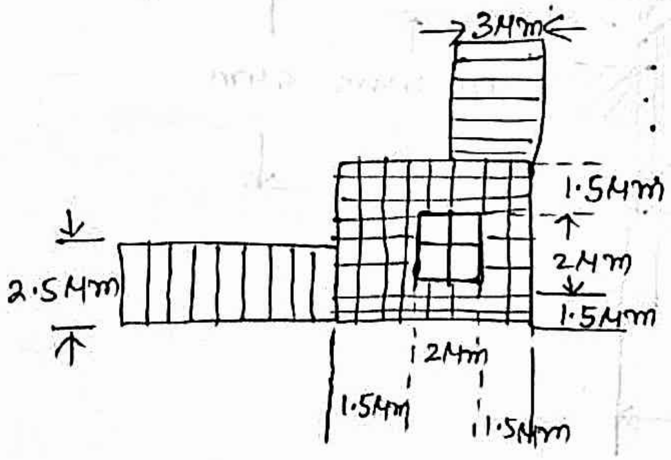
c) metal 1/ p^+ active



d) metal 1 to n-diffusion



e) multiple contact cuts



f) via metal1/metal 2

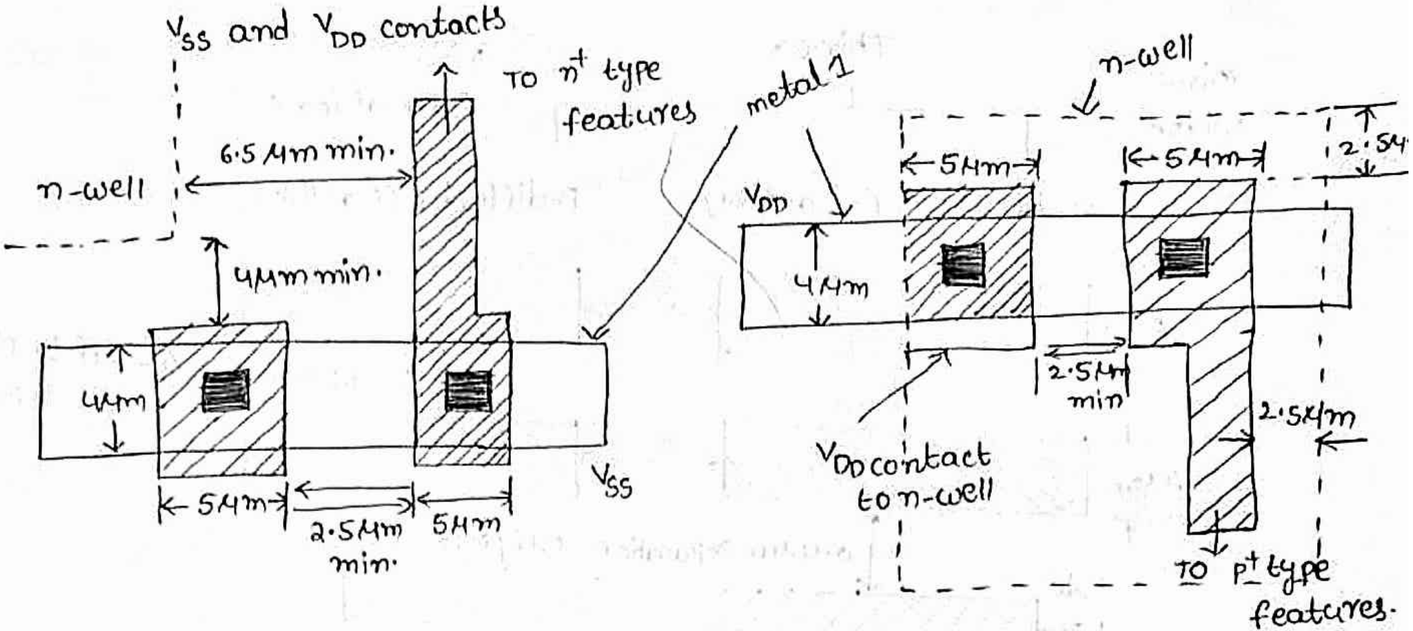
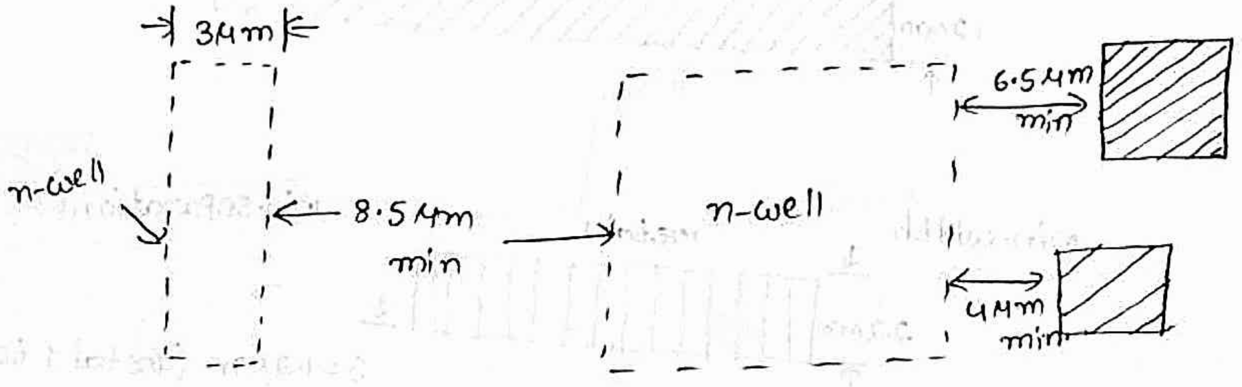


Fig :- Rules for n-well and V_{DD}, V_{SS} contacts



n-well spacings and width.

Fig :- Rules for n-well

1.2 μm Double metal, Single poly. CMOS rules :-

As fabrication technology, Improves so the feature size reduces and a separate set of micron based design rules must accompany each new feature size.

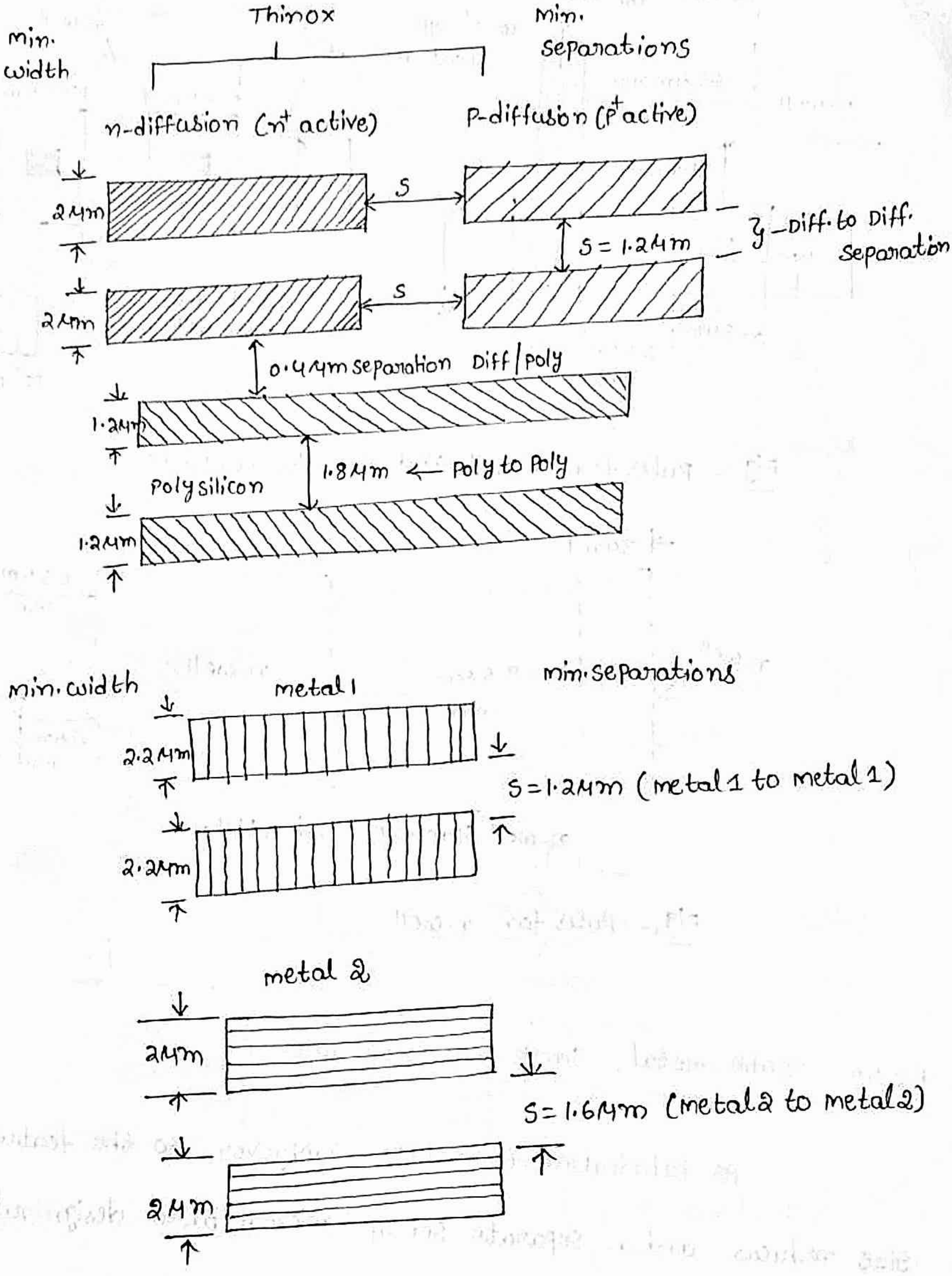
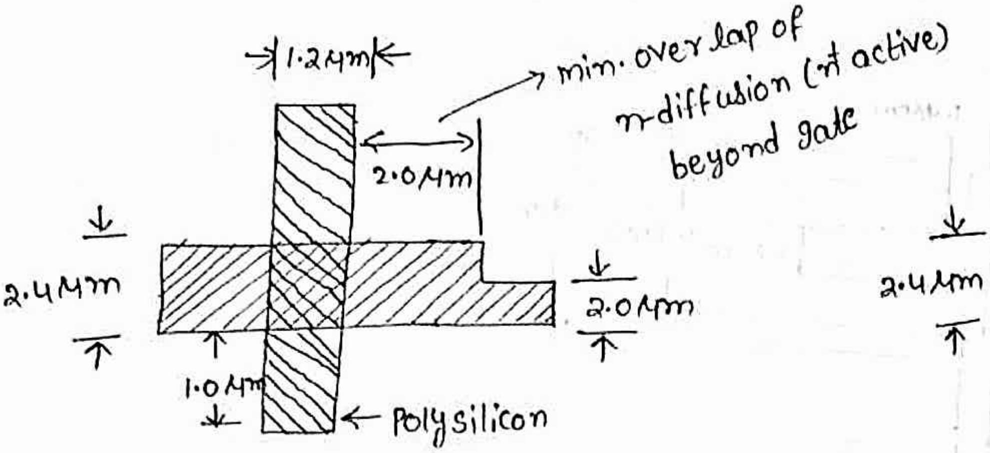
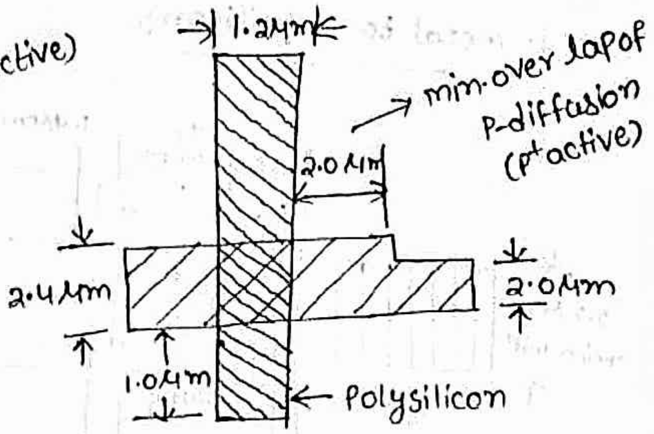


Fig :- Design rules for wires (Inter connects)

Design rules for transistors :-



a) n-type enhancement



b) P-type enhancement

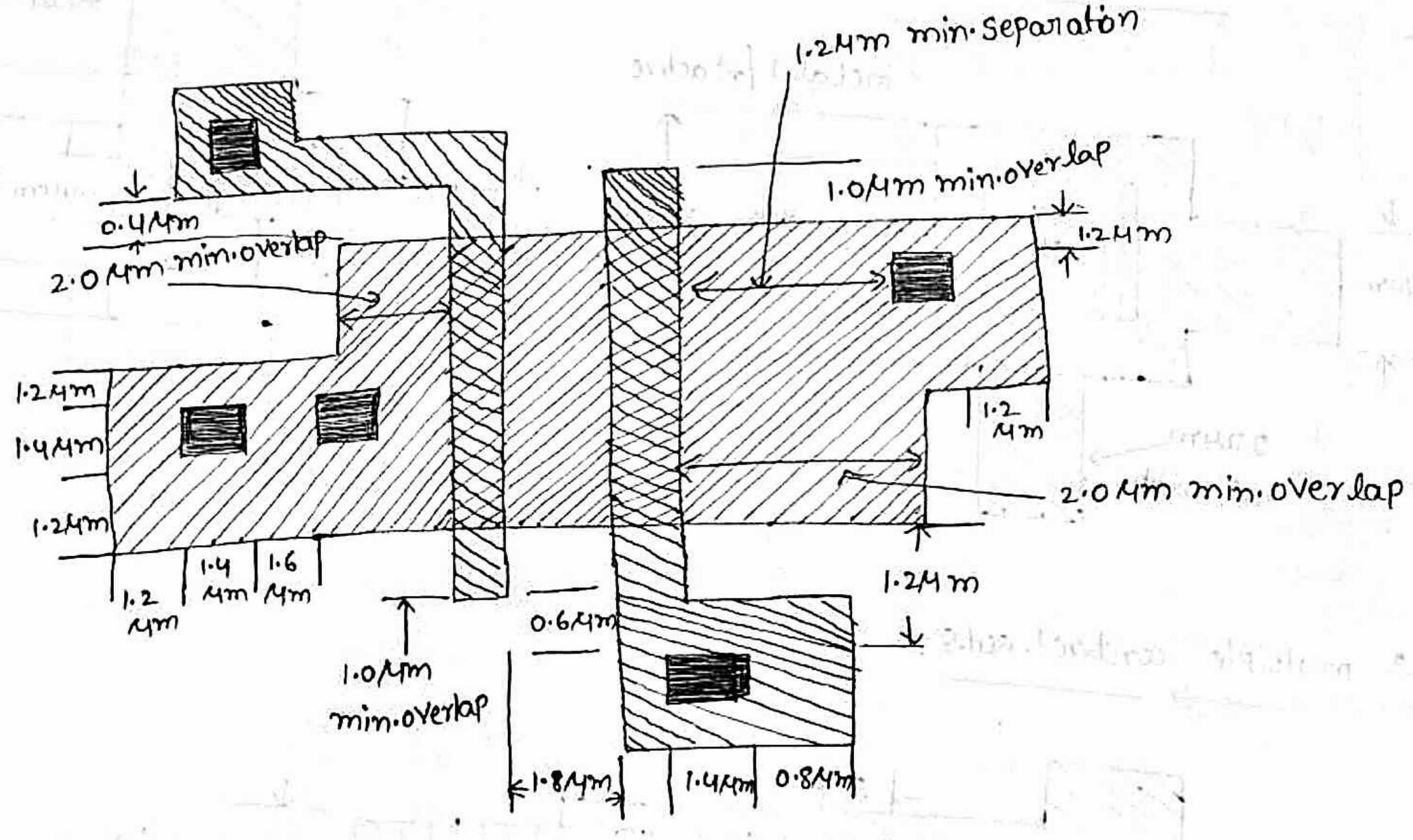
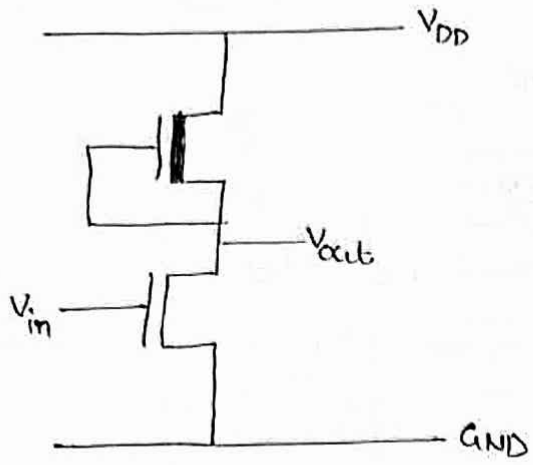
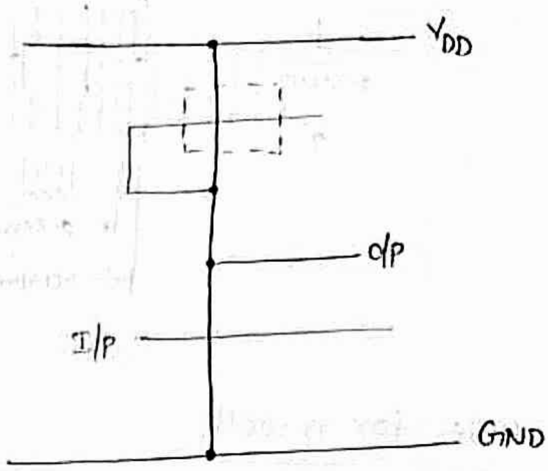


Fig :- All devices shown in n-type. The same rules for apply for P-type

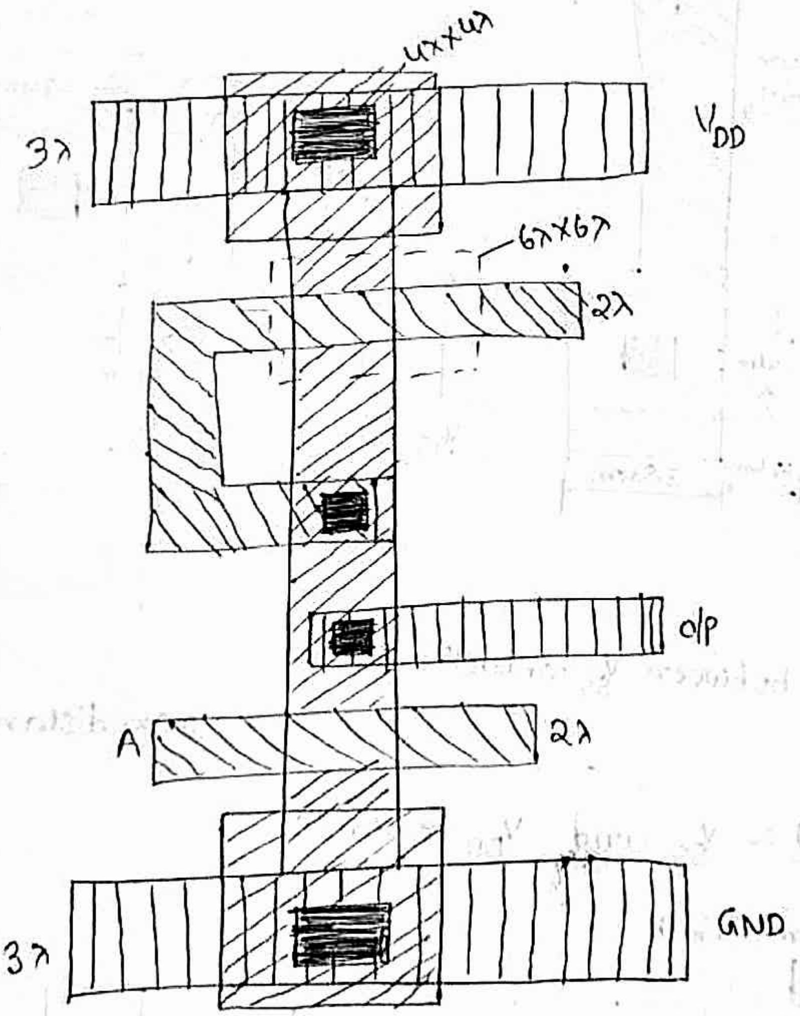
1. Draw the stick diagram and Layout diagram of NMOS Inverter.



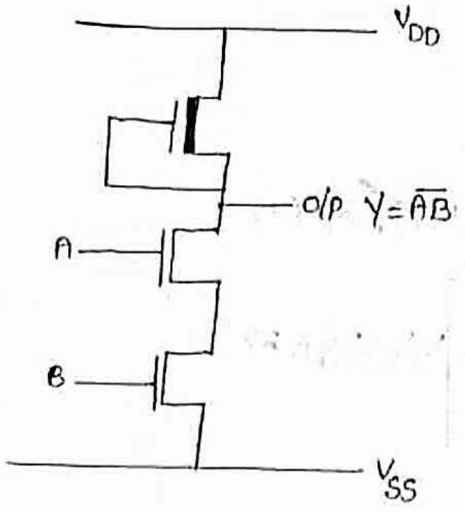
stick diagram



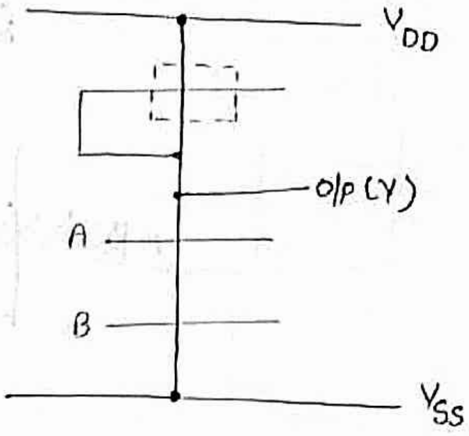
Layout :-



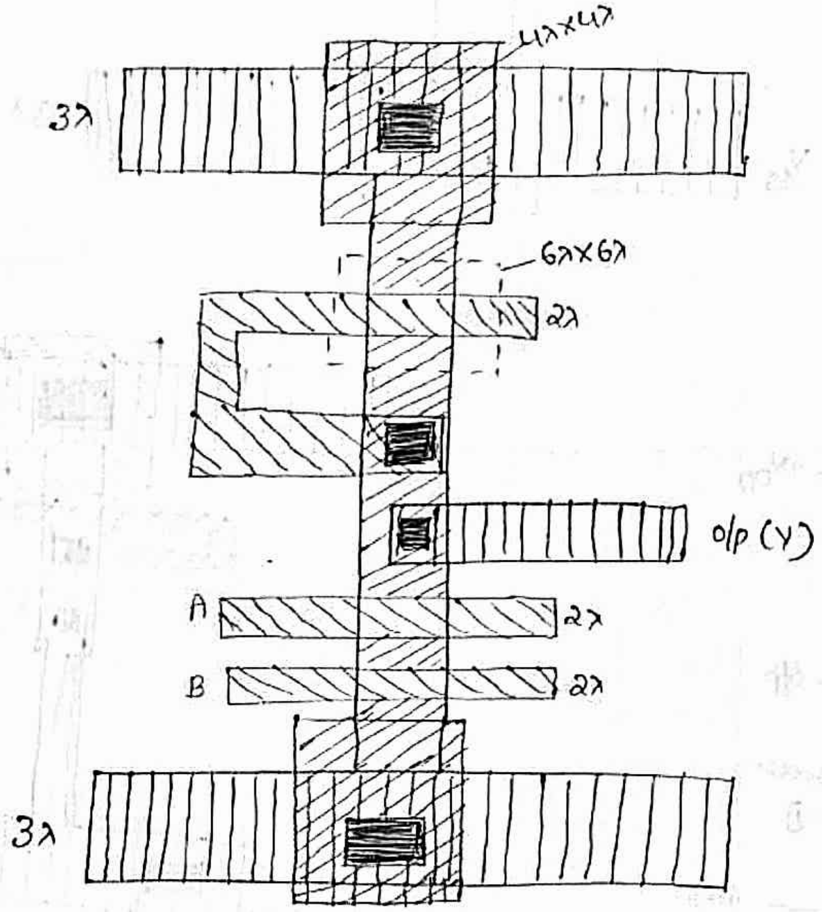
2. Draw the Layout diagram of 2-Input NMOS NAND gate. (10)



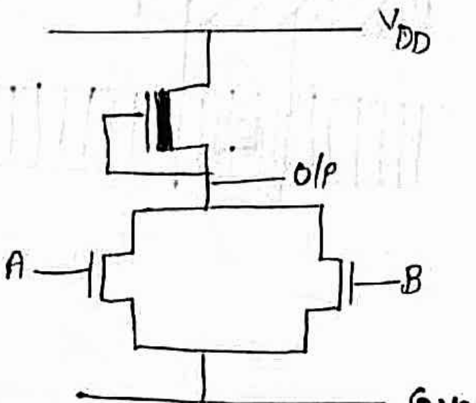
stick diagram

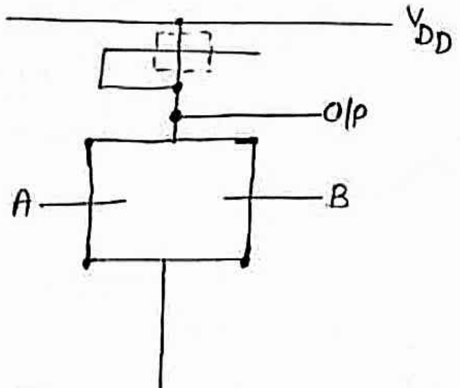


Layout :-

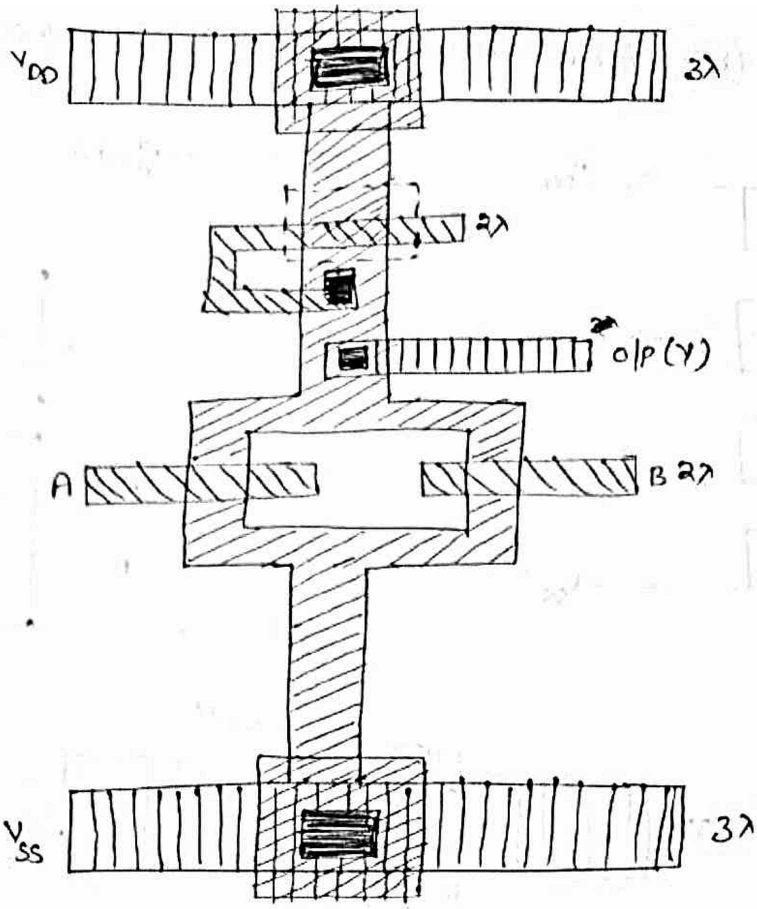


3. Draw the circuit & Layout diagram of Two Input NOR gate using nmos Logic.



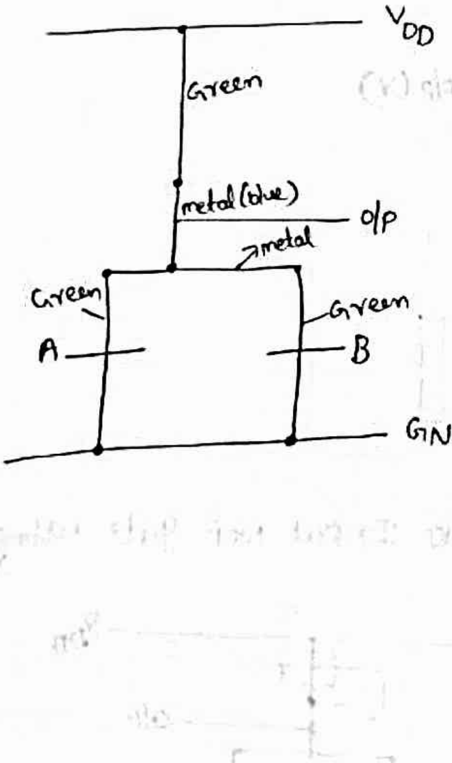


Layout

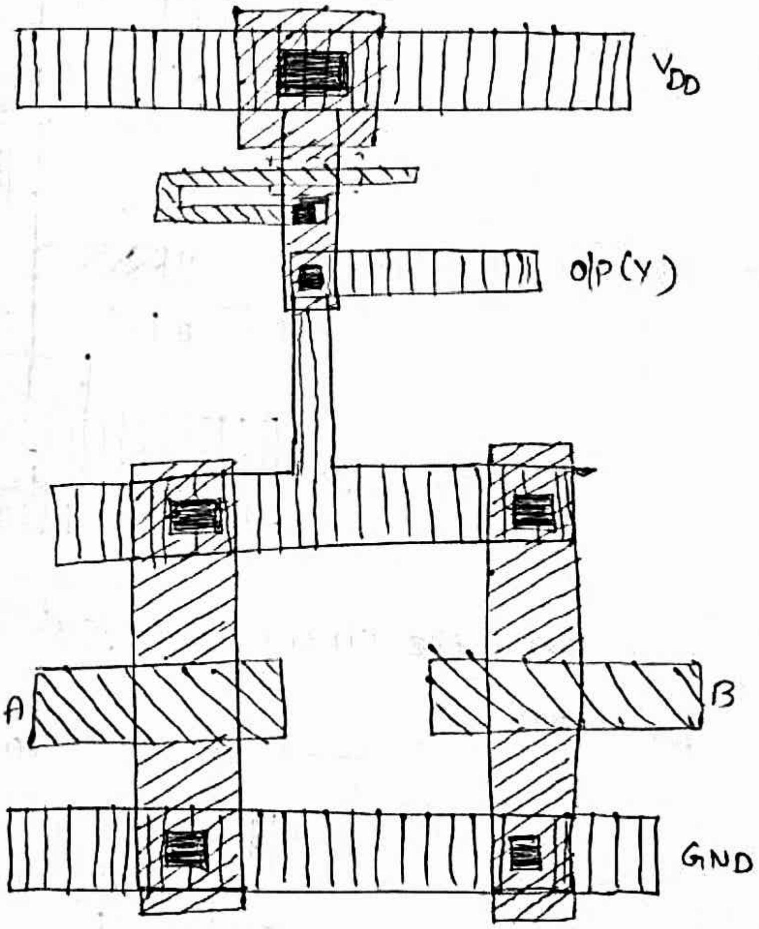


(or)

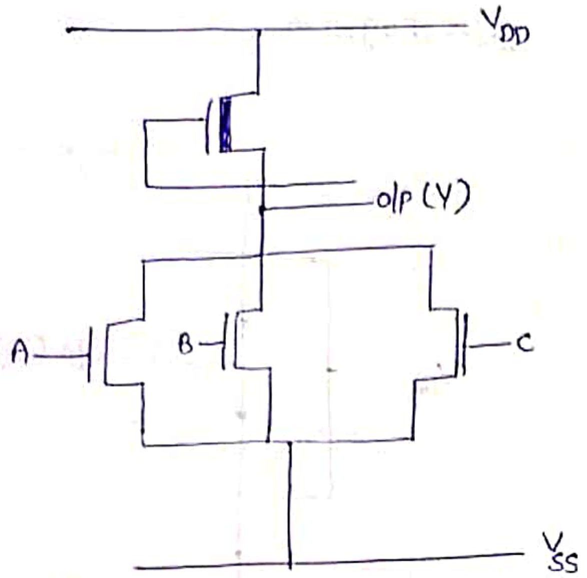
stick diag



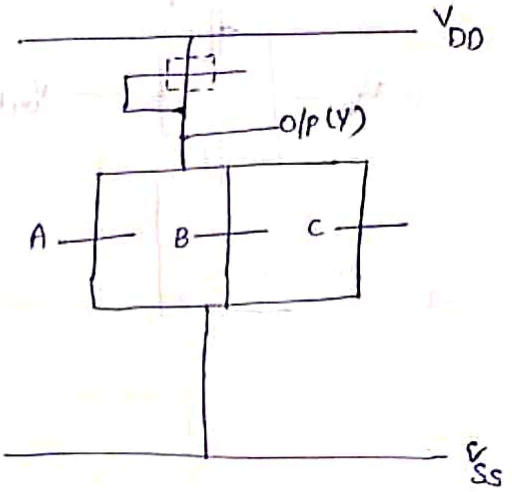
Layout



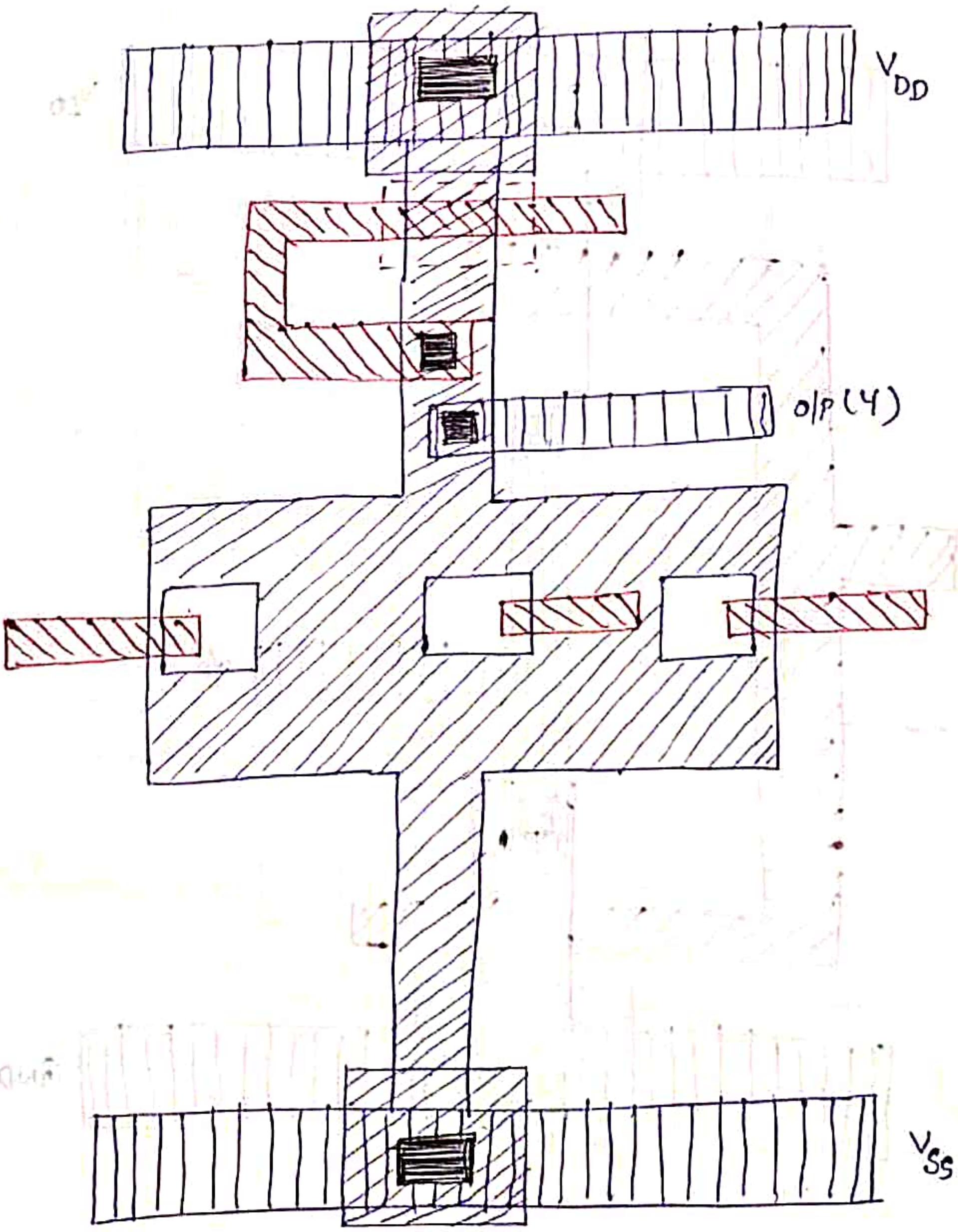
4. Draw the Layout & stick diagram of 3-i/p NOR gate.



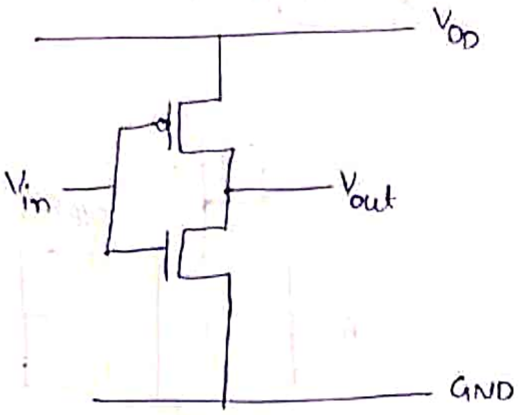
stick diagram



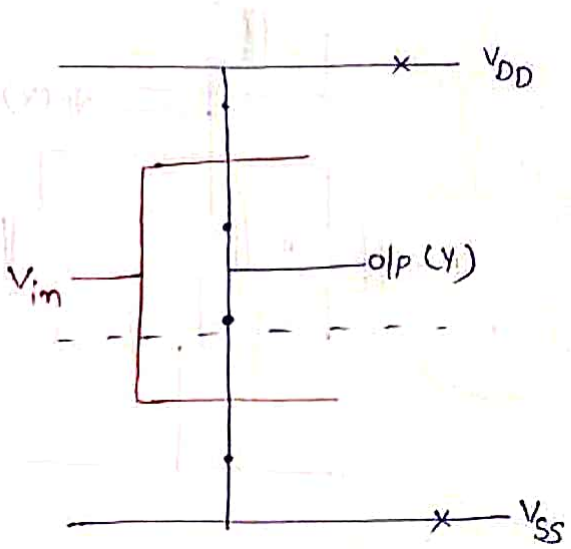
Layout



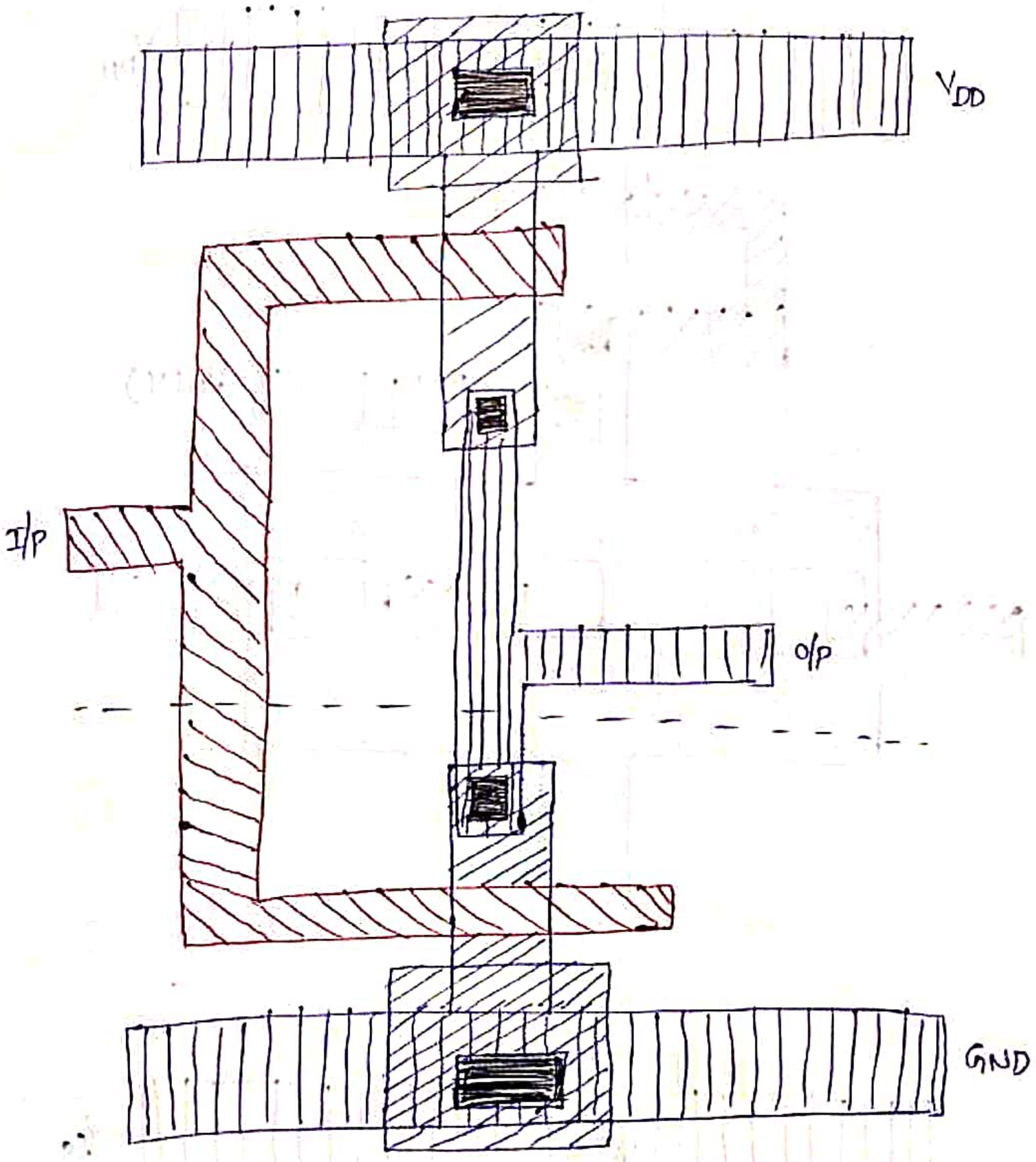
5. Draw the stick and layout diagrams of CMOS Inverter.



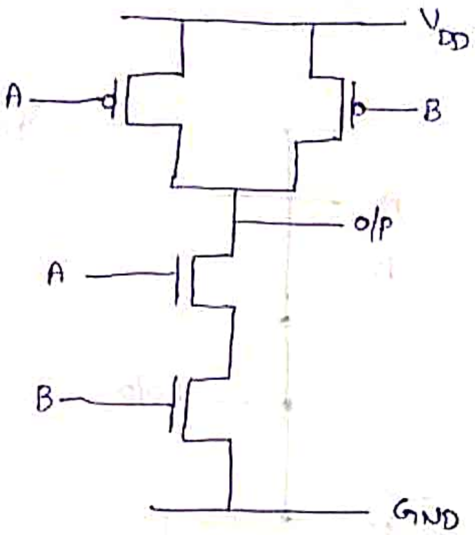
stick diagram



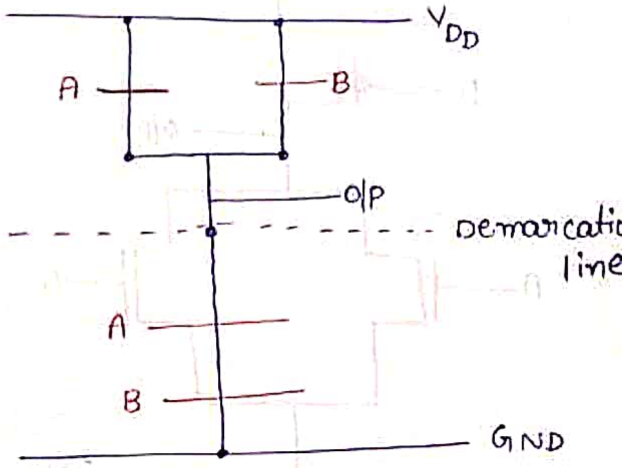
Layout



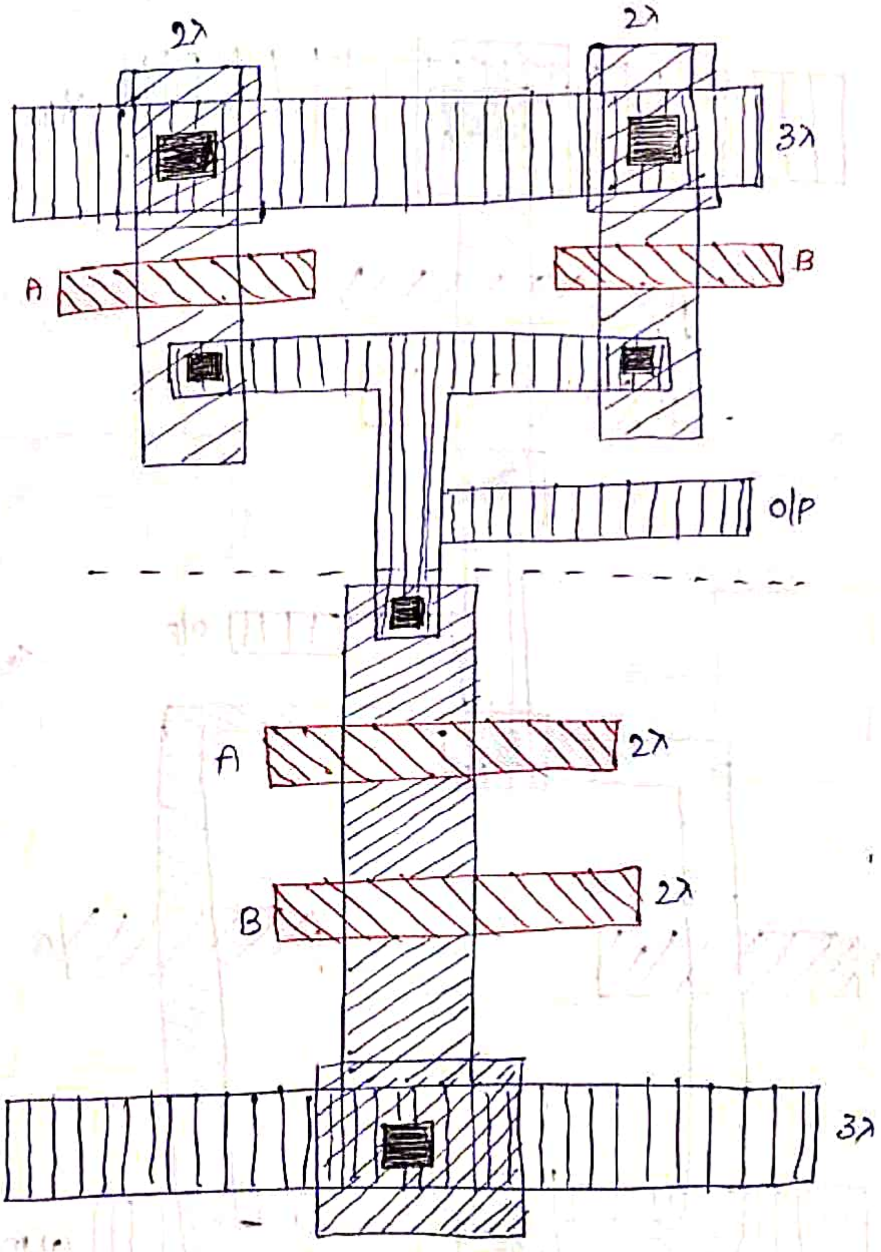
6. Draw the Layout diagram of 2-I/P CMOS NAND Gate



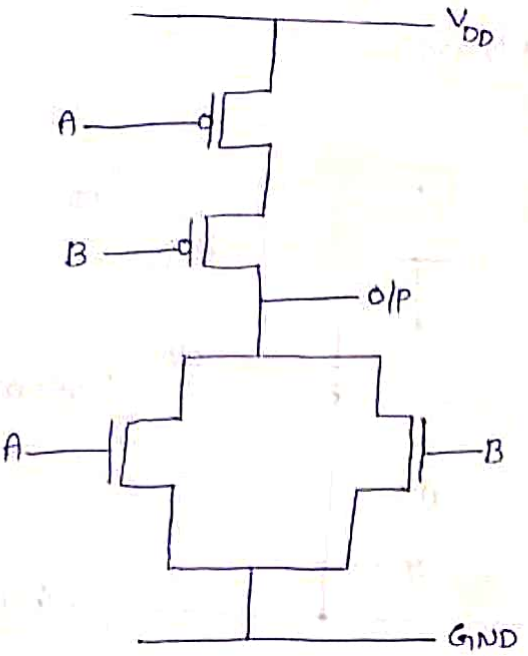
Stick diagram



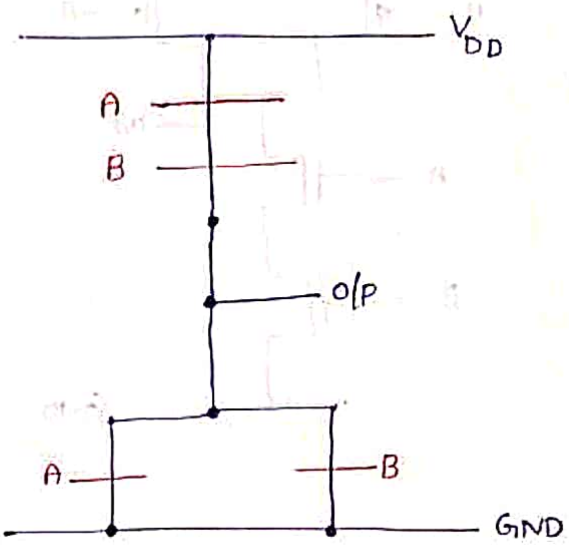
Layout :-



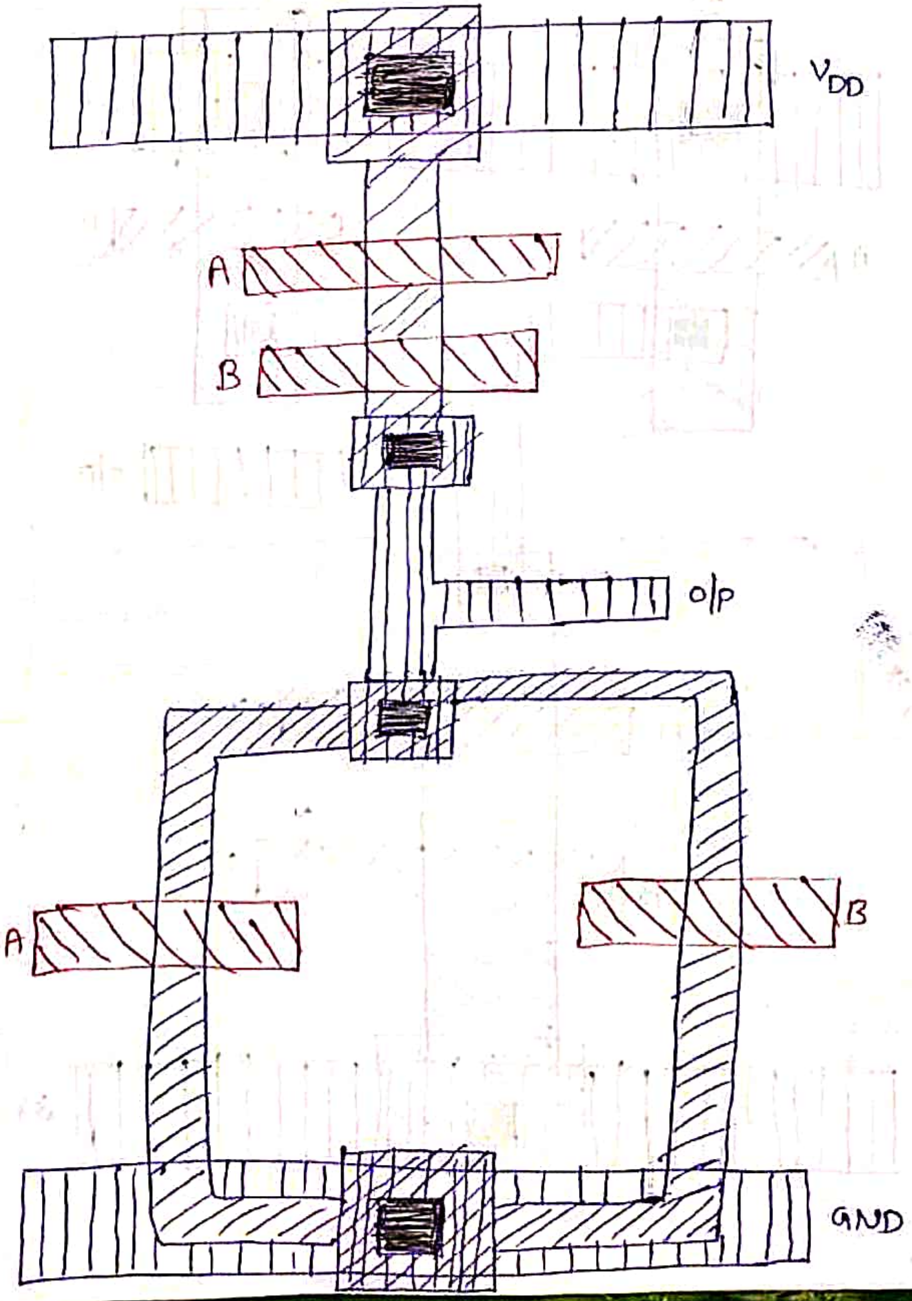
7. Draw the Layout diagram of 2-I/P CMOS NOR Gate.



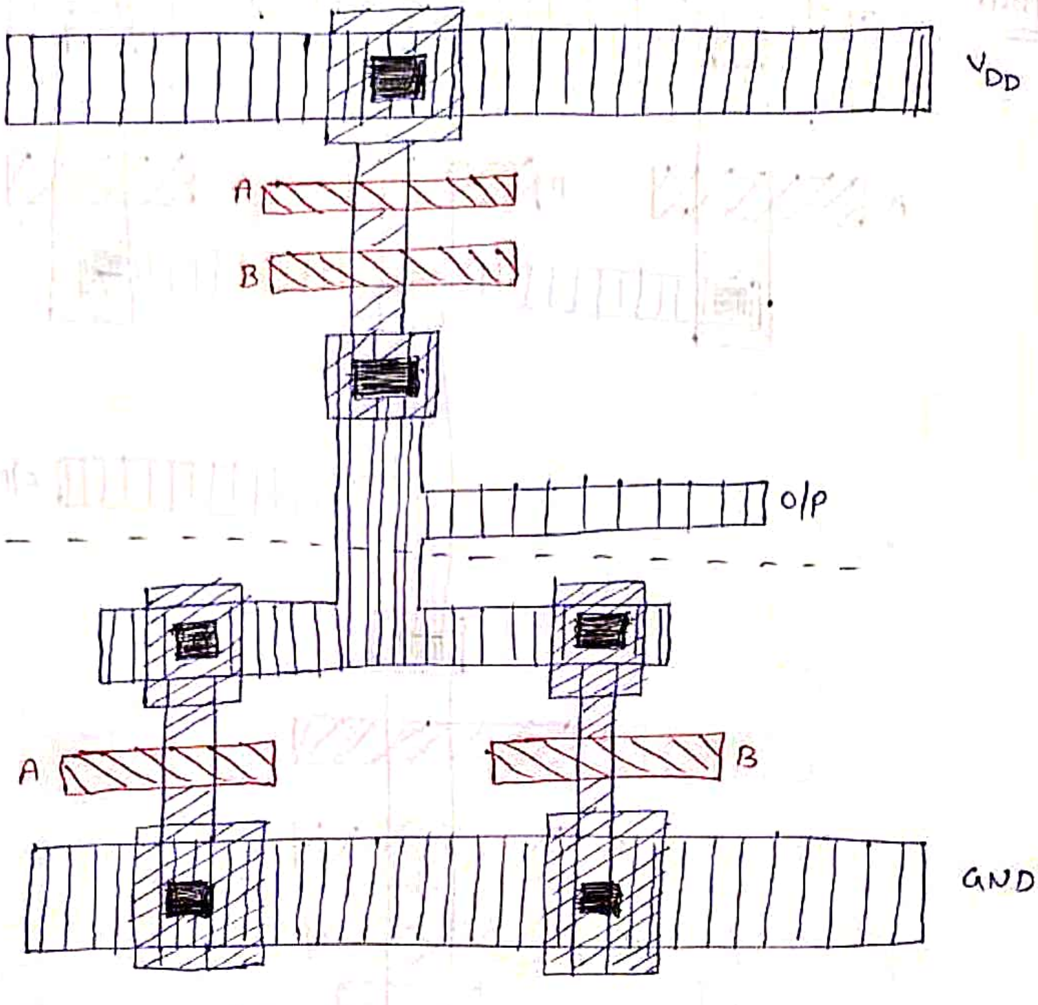
stick diagram



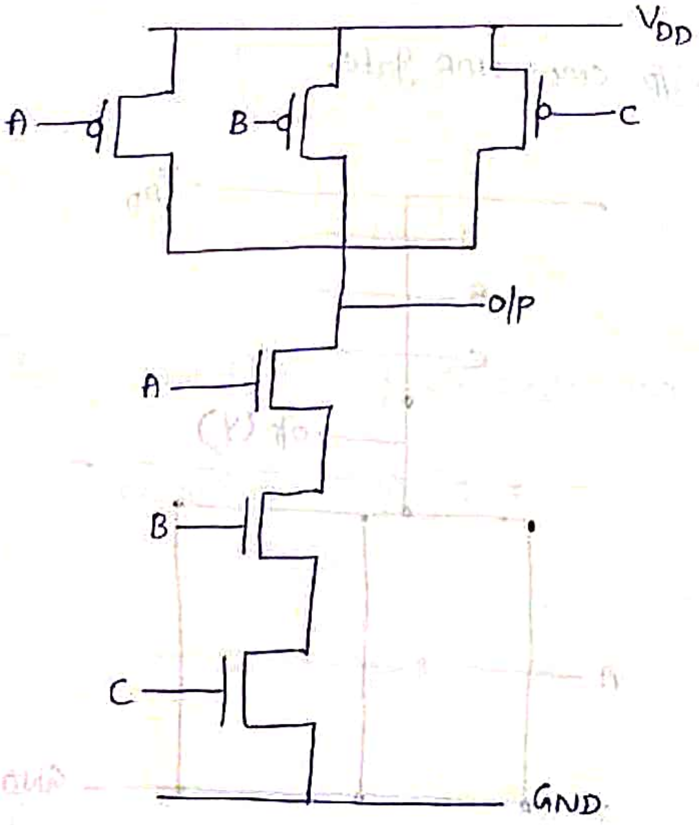
Layout



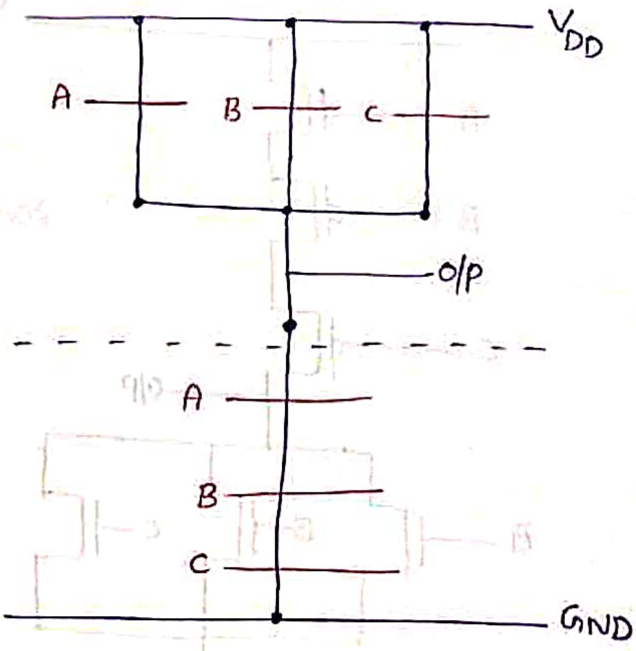
(or) Layout



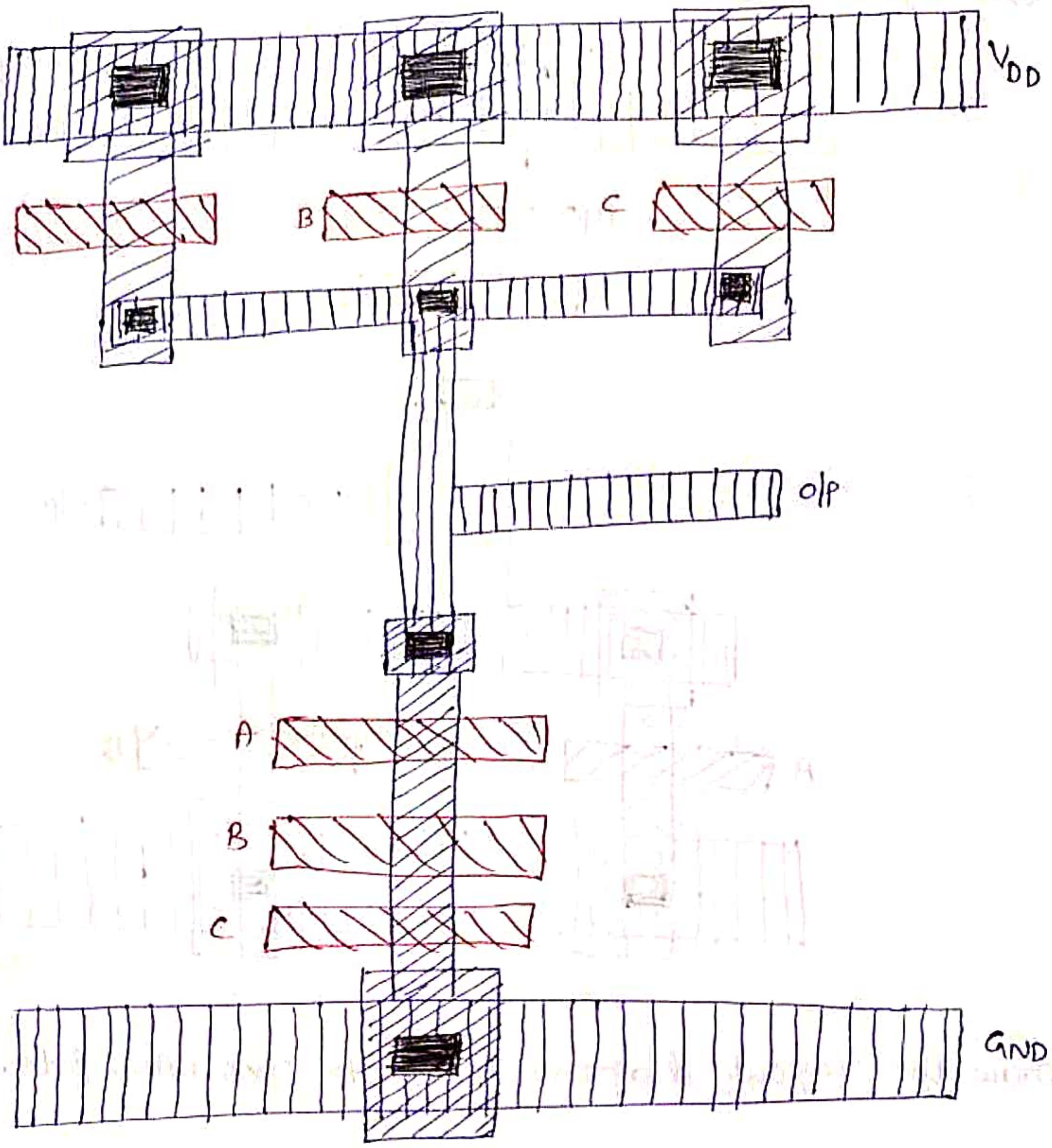
8. Draw the Layout diagram of 3-I/P CMOS NAND gate.



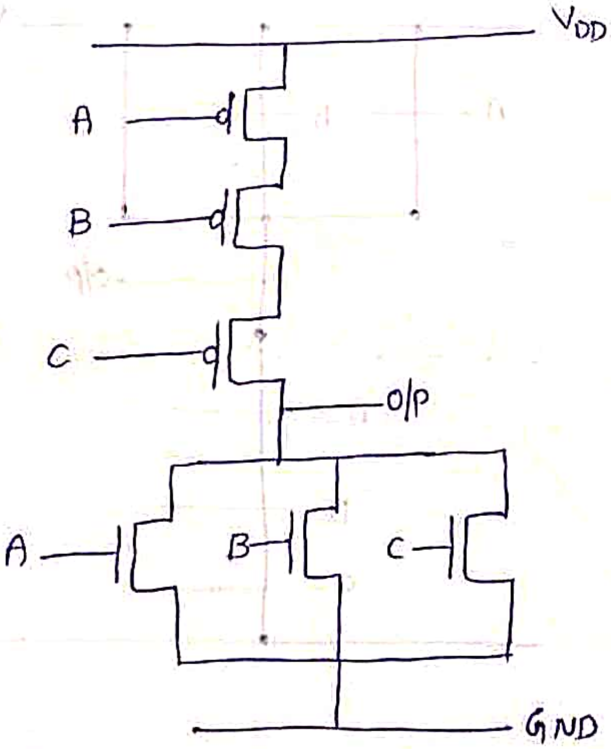
stick diagram

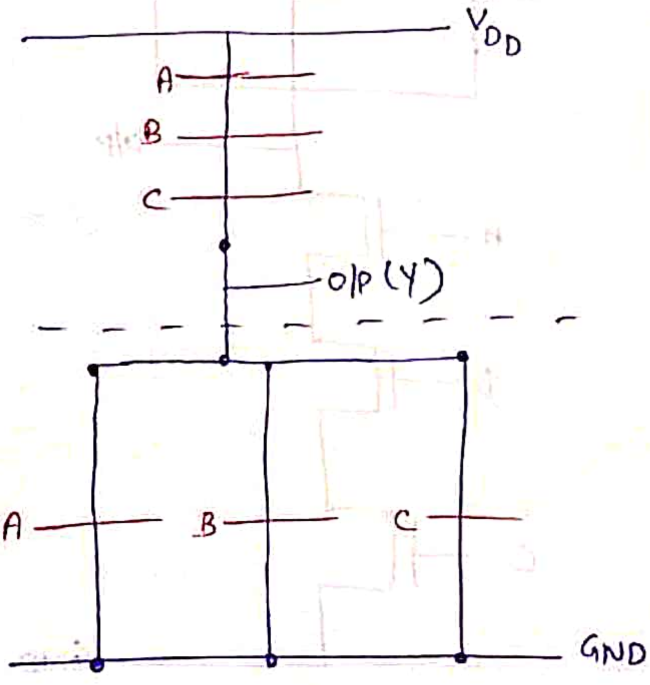


Layout

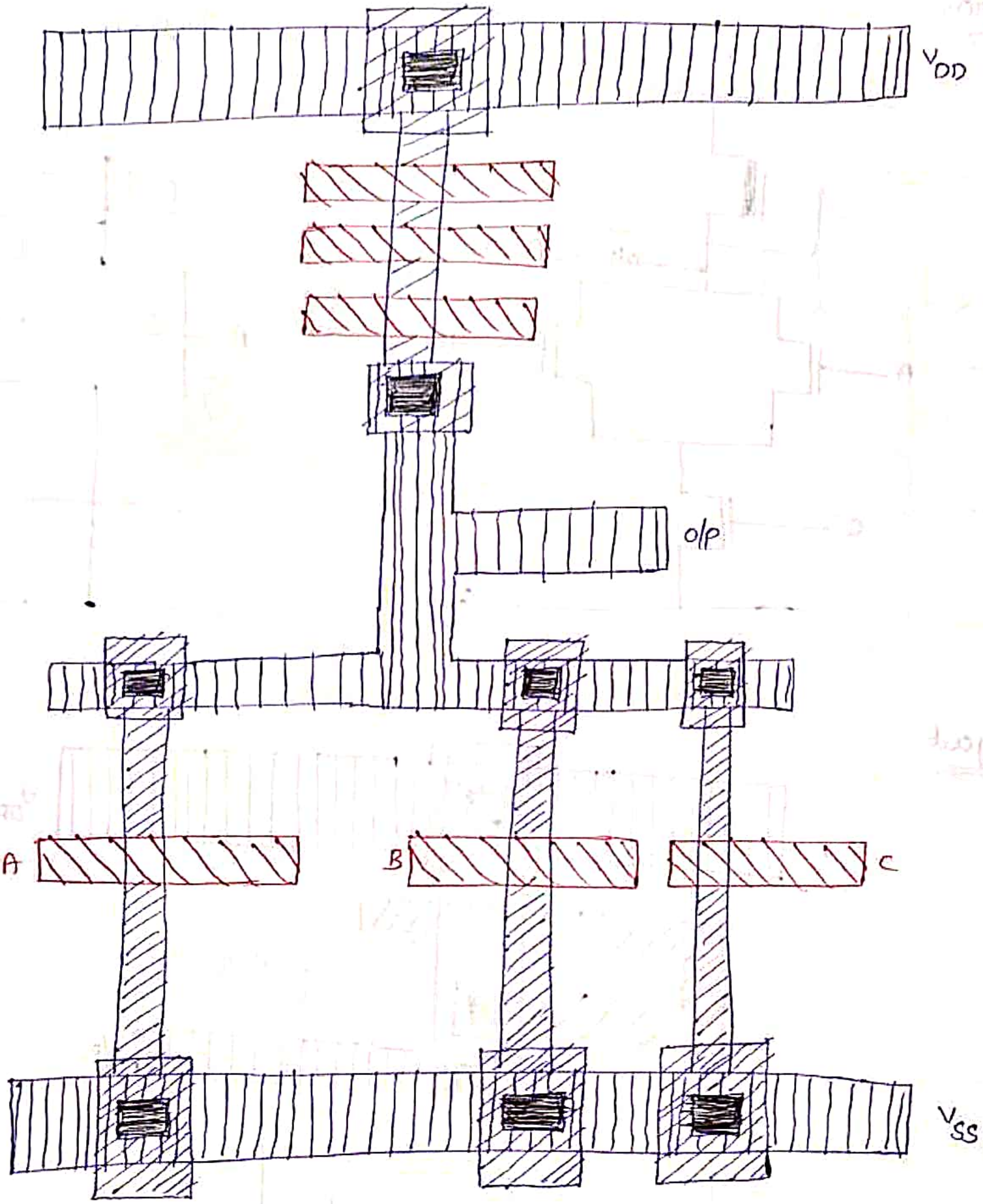


9. Draw the Layout diagram of 3-i/p CMOS NOR gate.



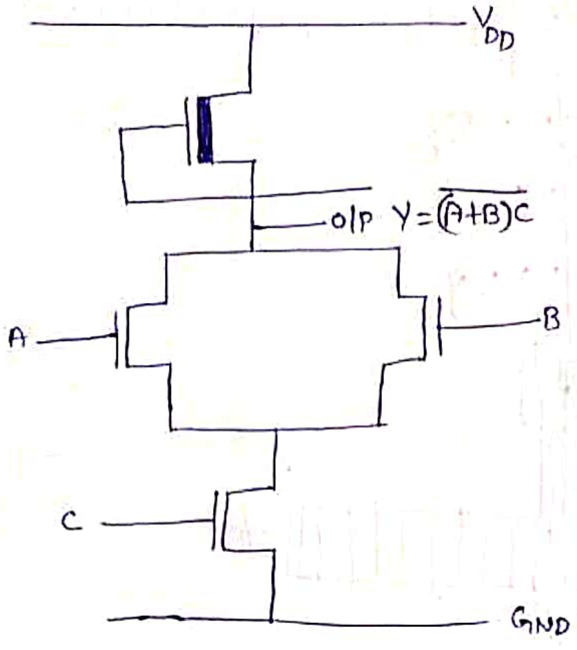


Layout

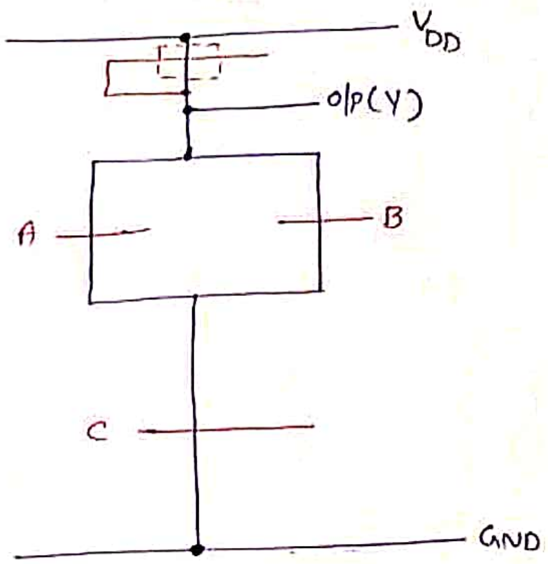


10. Draw the Layout diagrams of Nmos, Pmos, Cmos using Expression $Y=\overline{(A+B)C}$

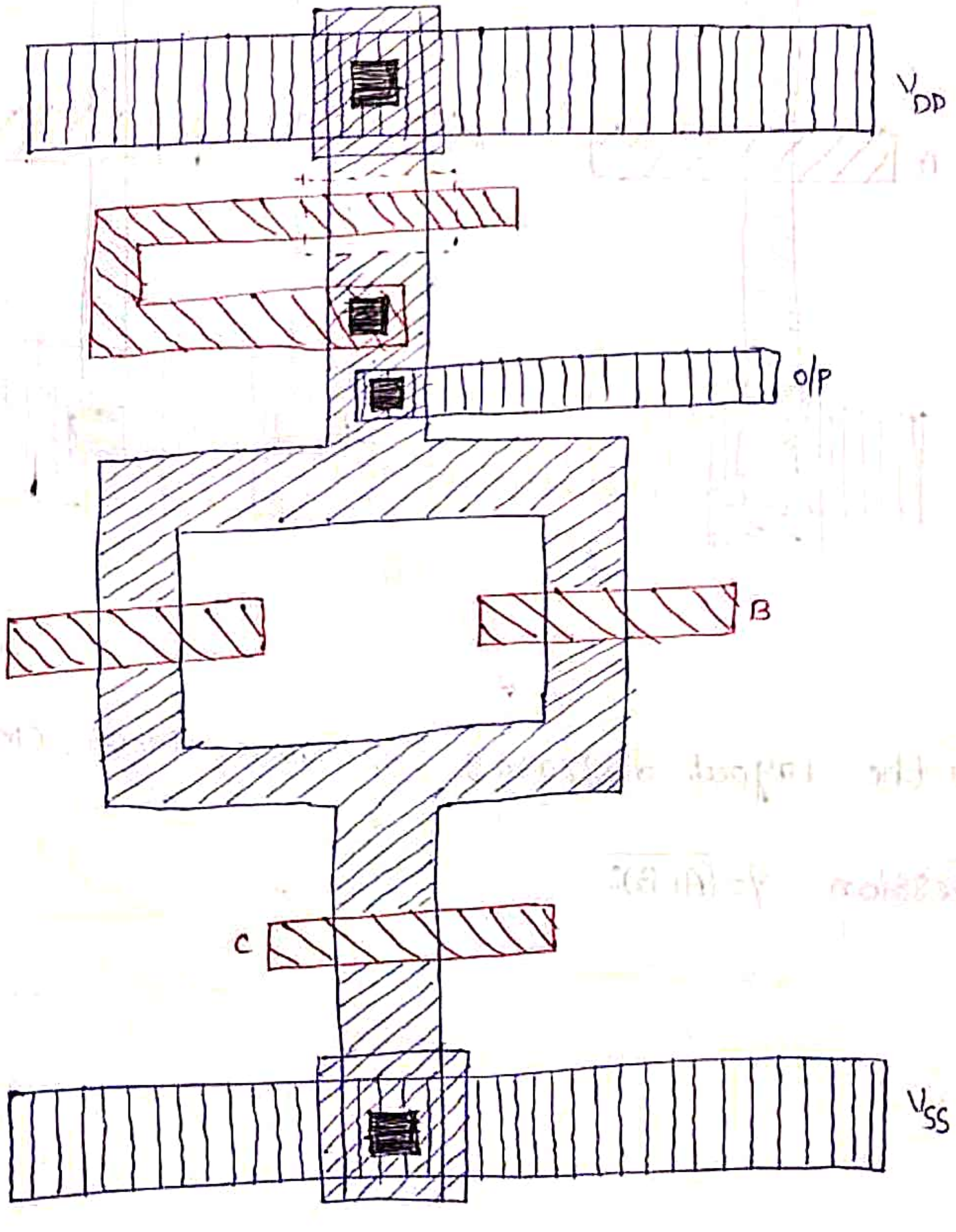
i) NMOS



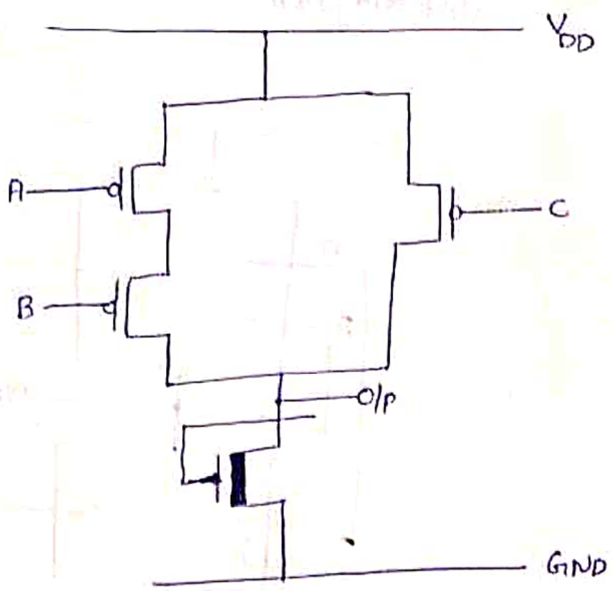
stick diagram



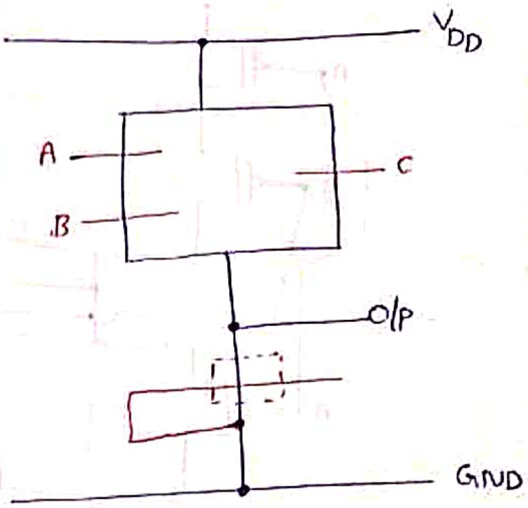
Layout



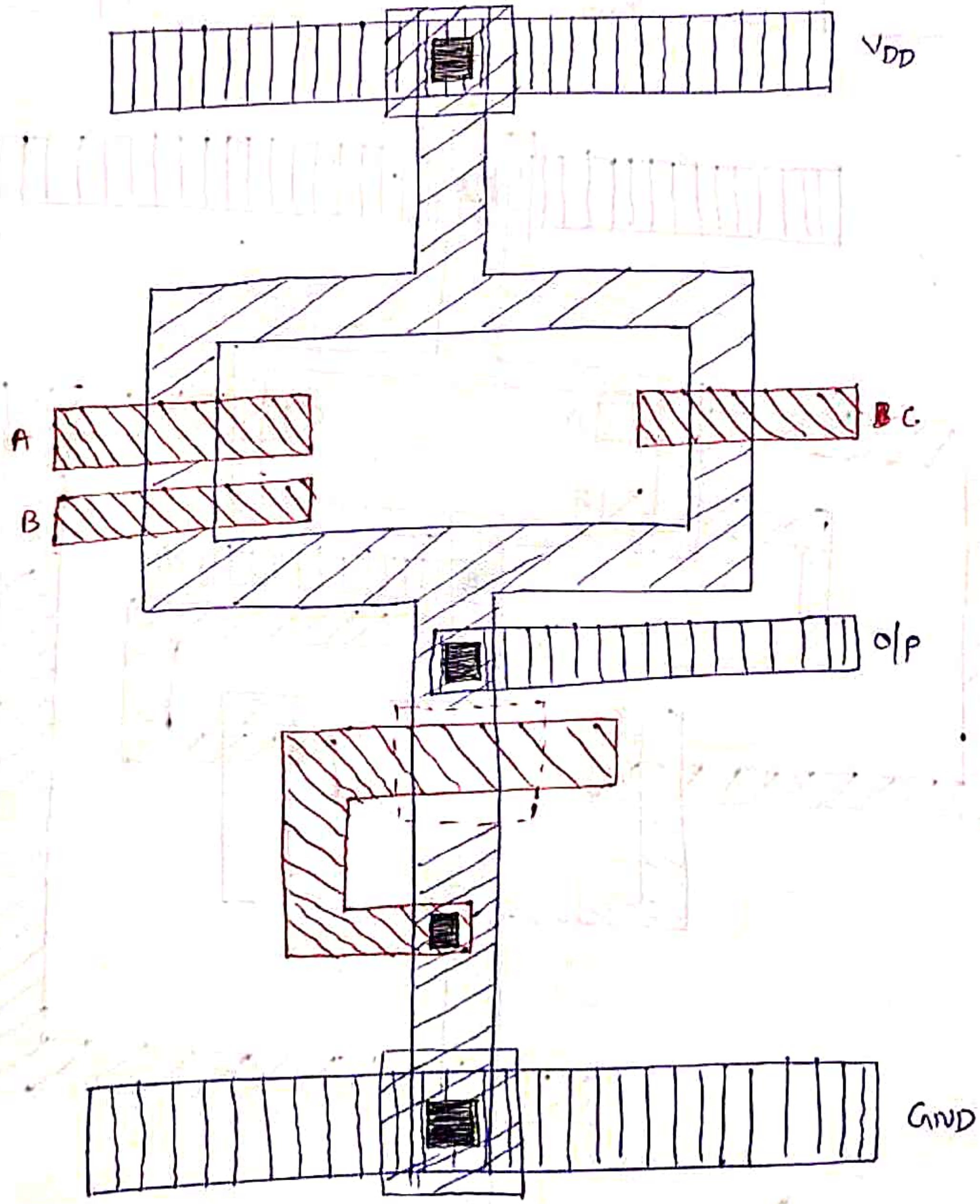
ii) PMOS



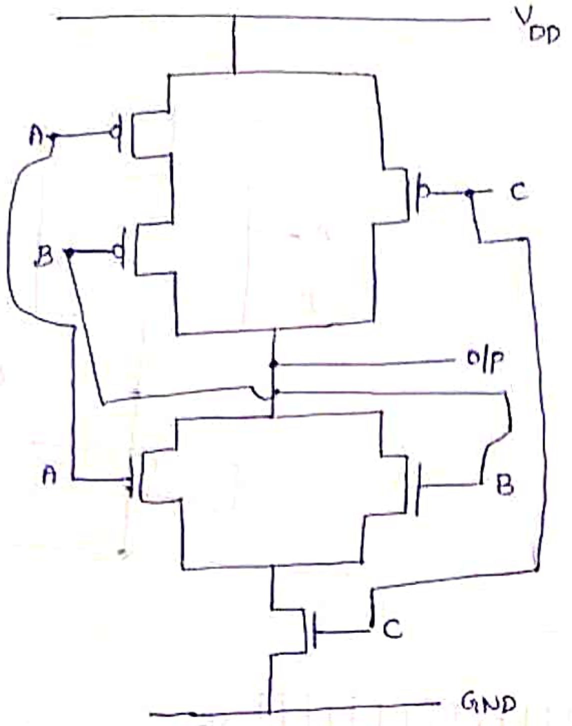
Stick diagram



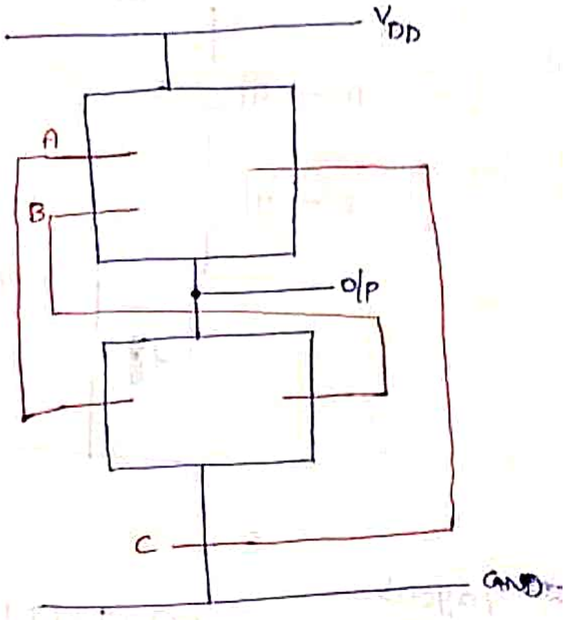
Layout



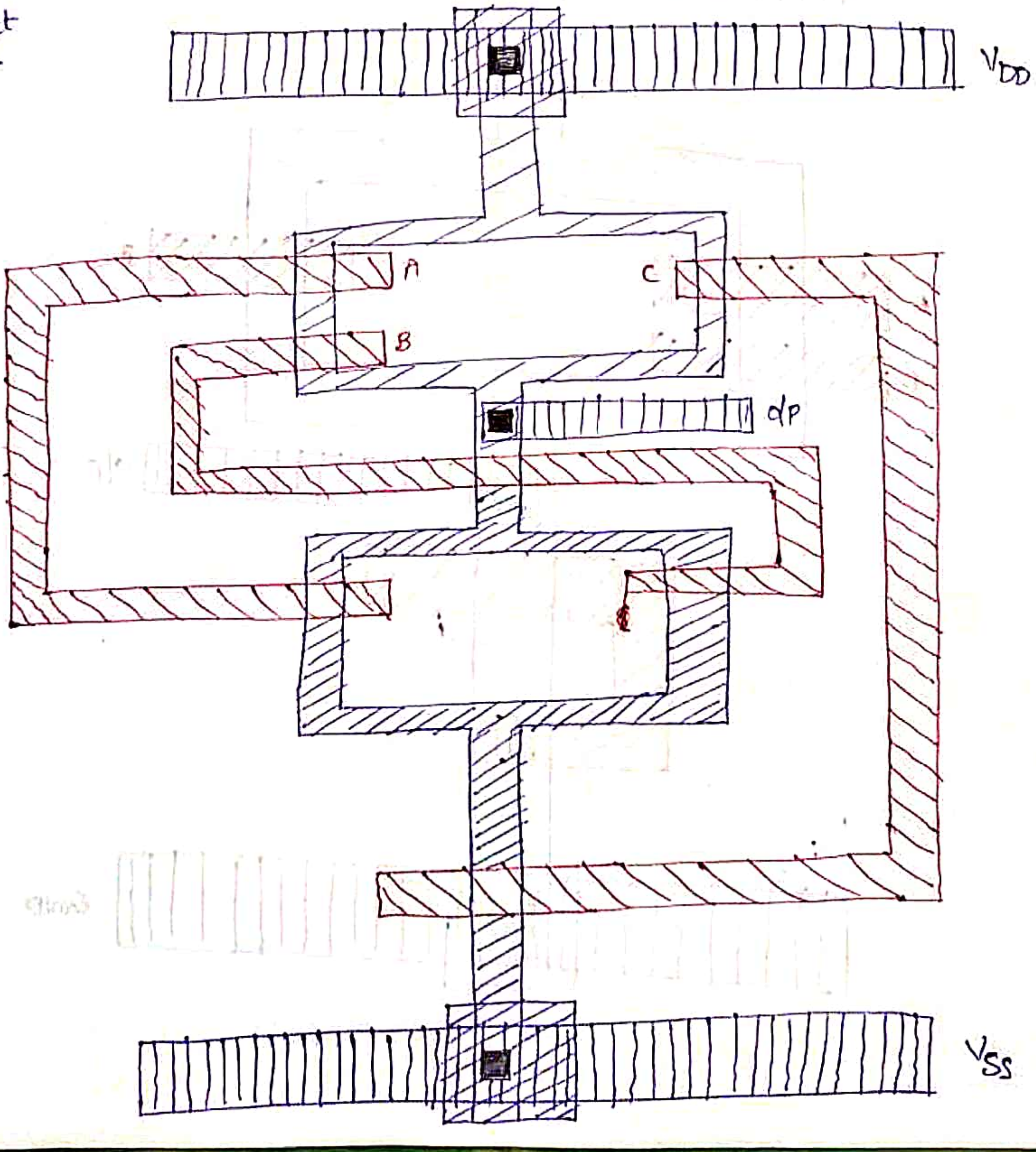
iii) CMOS



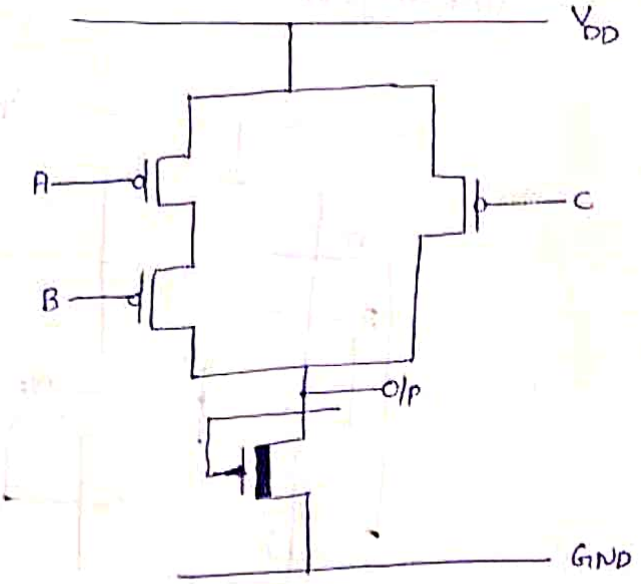
stick diagram



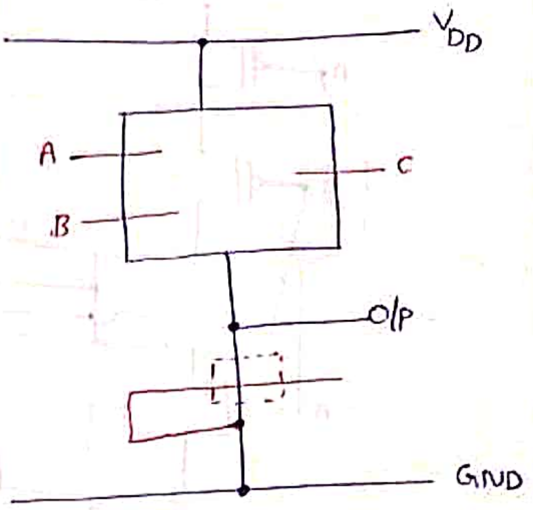
Layout



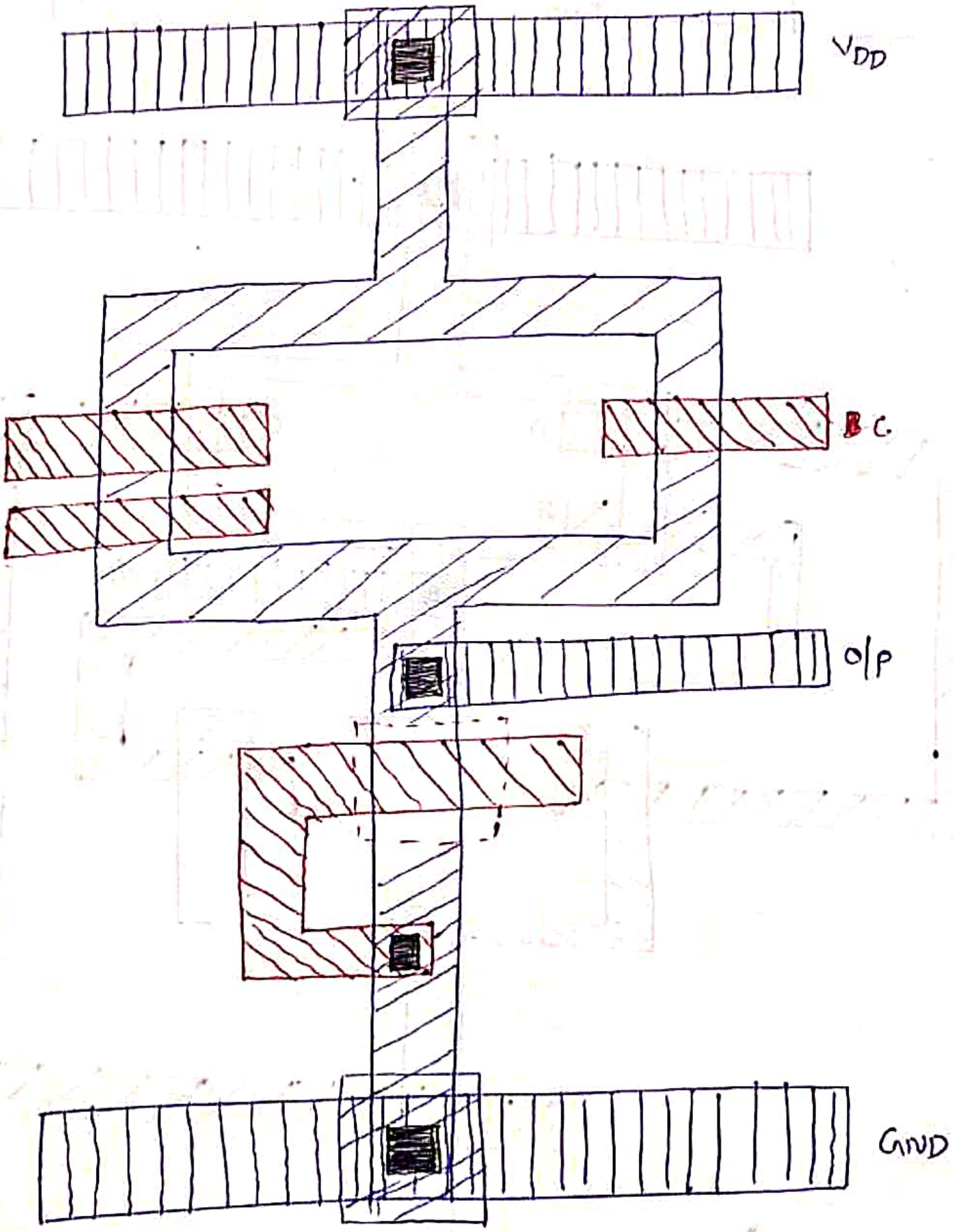
ii) PMOS



stick diagram

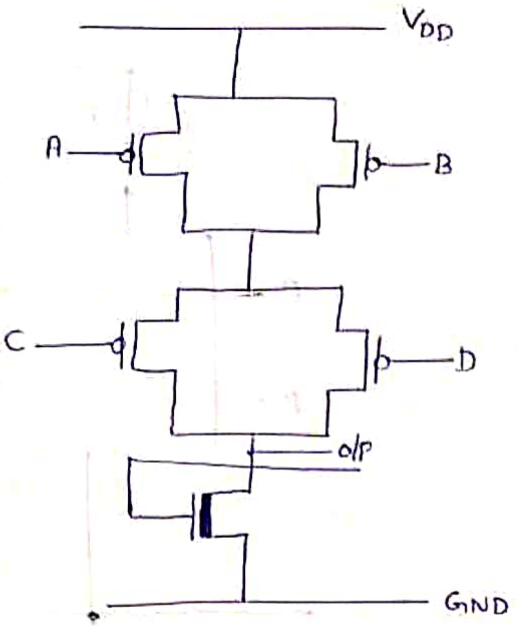


Layout

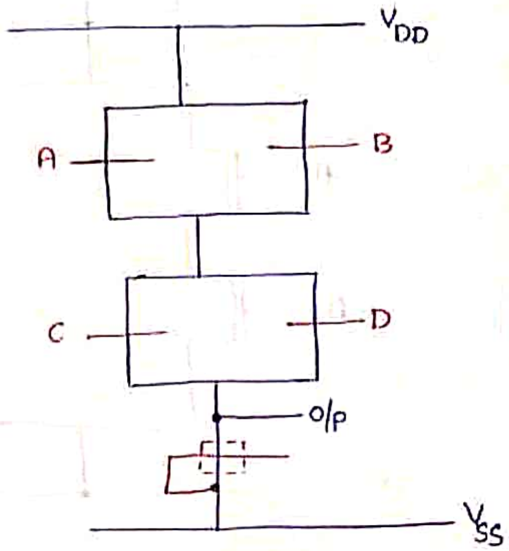


10. Draw the layout diagram of $Y=\overline{AB+CD}$ use pmos, Nmos, cmos Logic

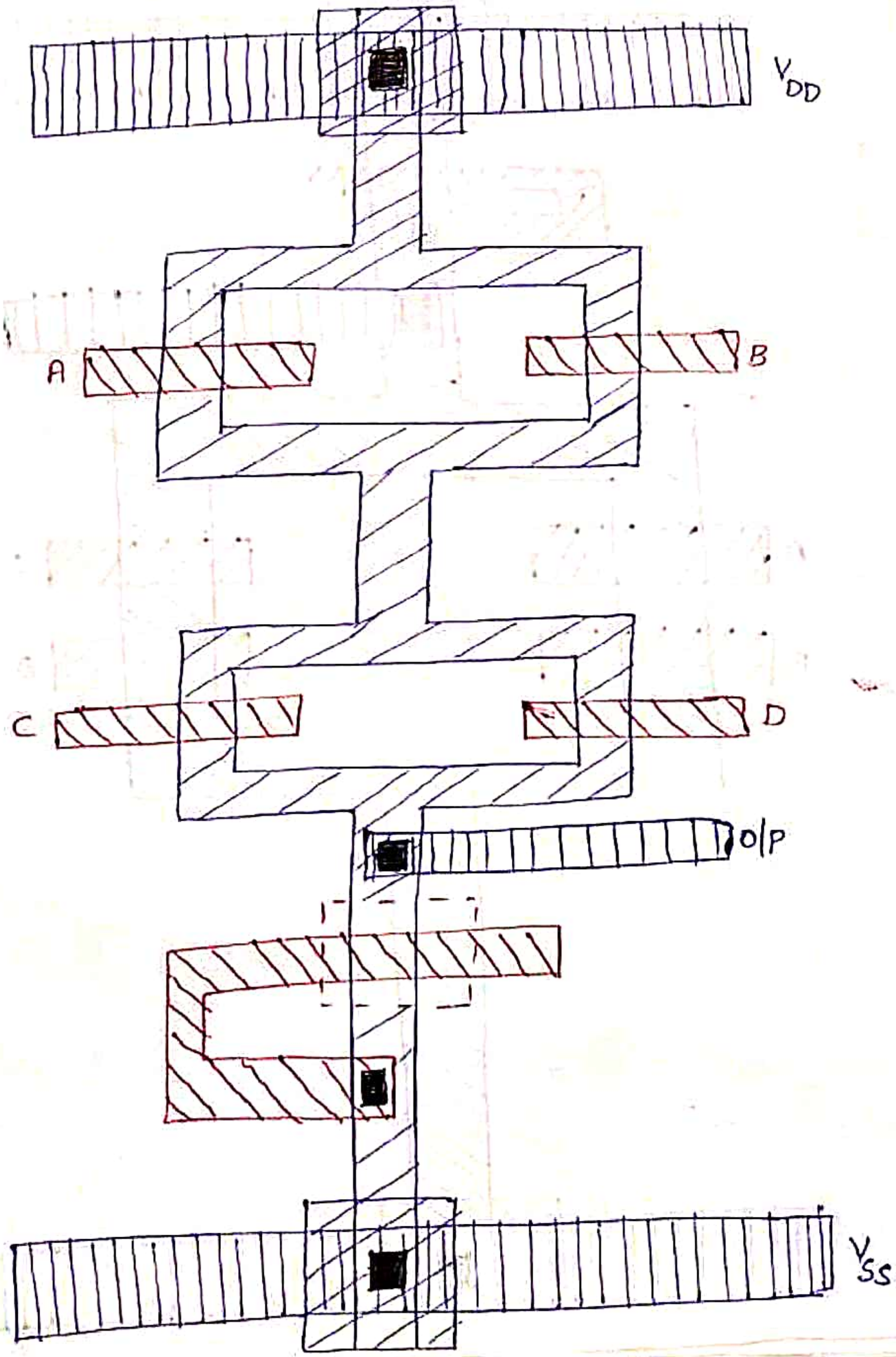
pmos



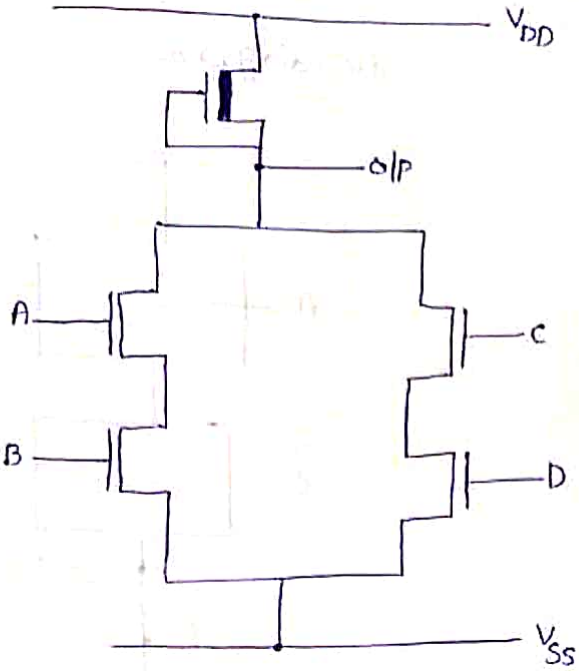
stick diagram



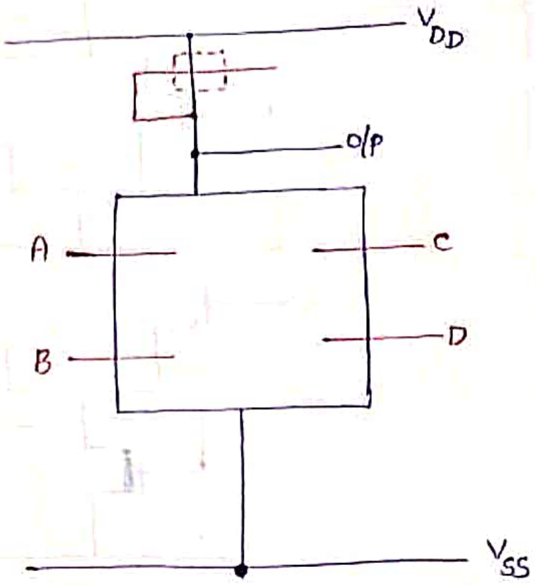
Layout



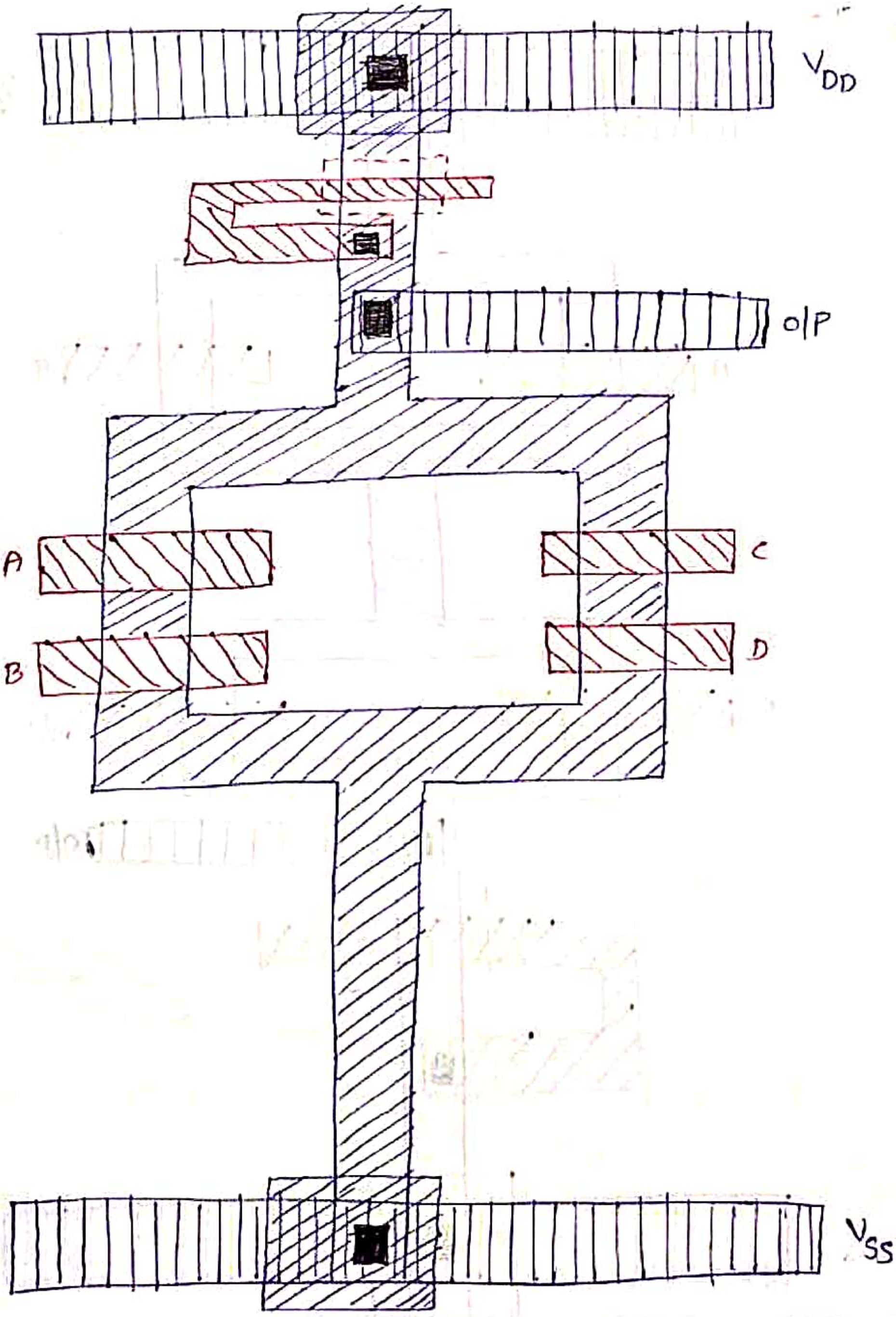
Nmos



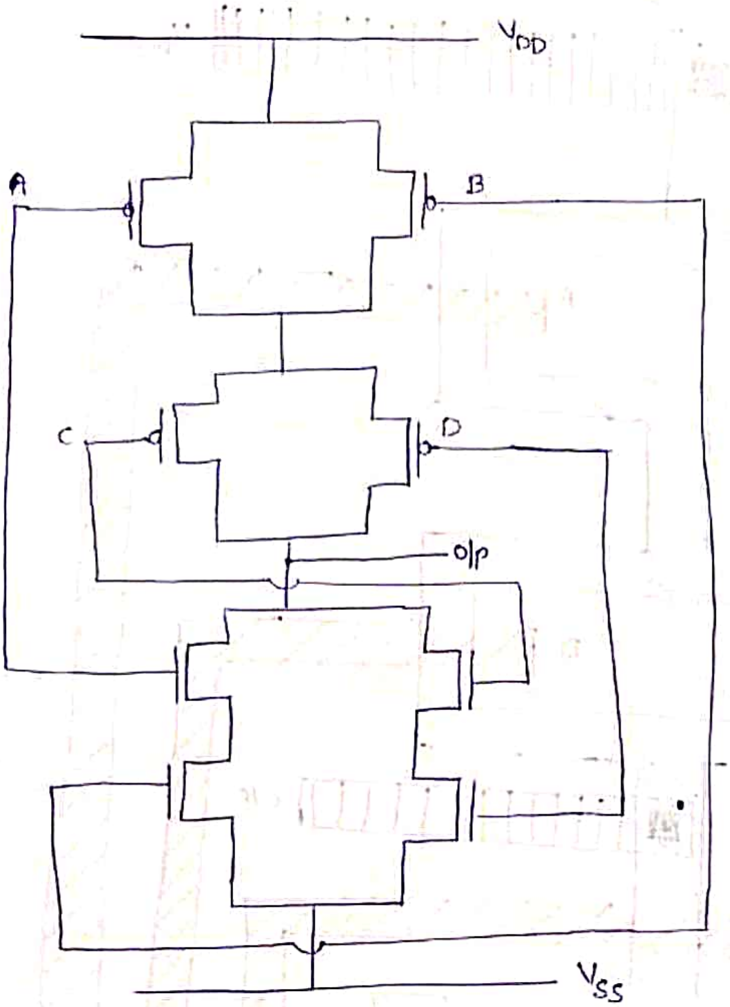
stick diagram



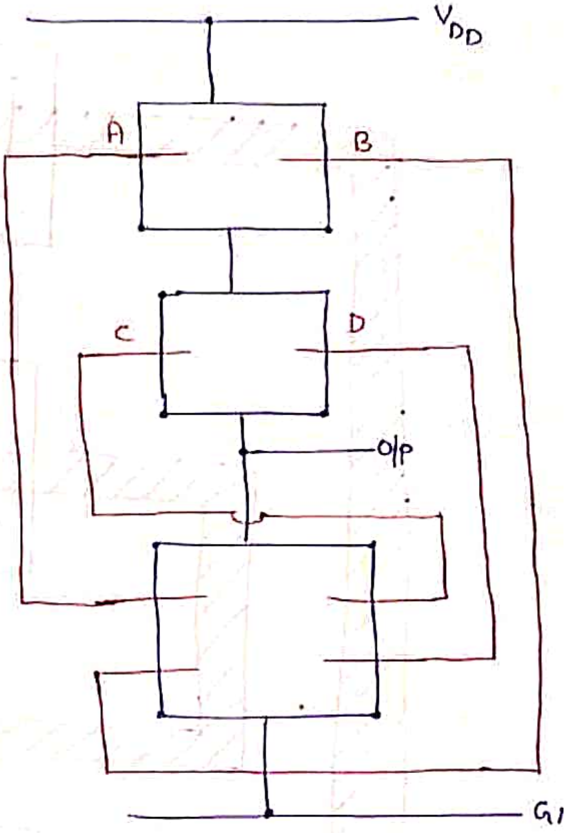
Layout



CMOS

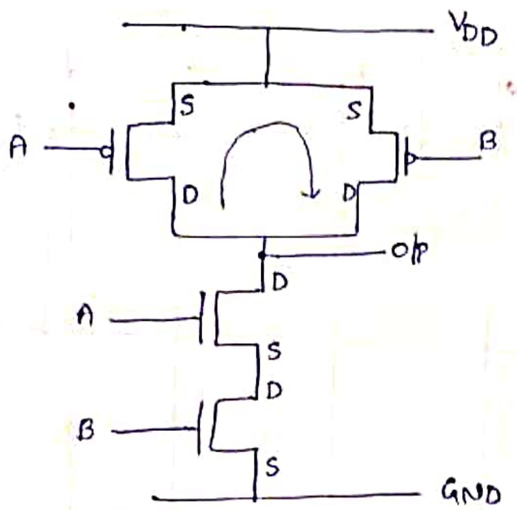


Stick diagram

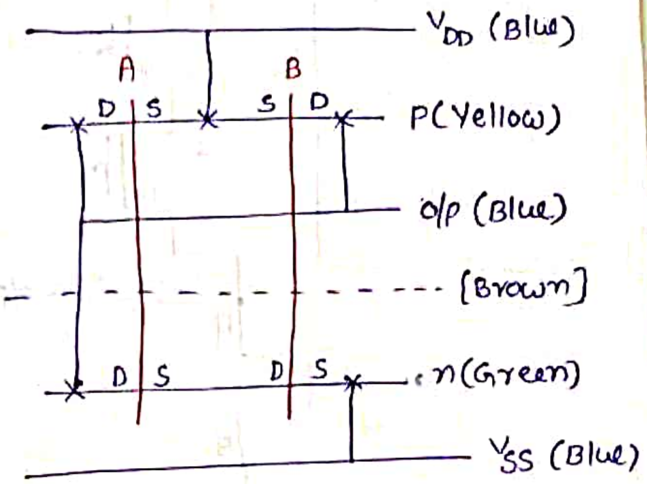


Layout

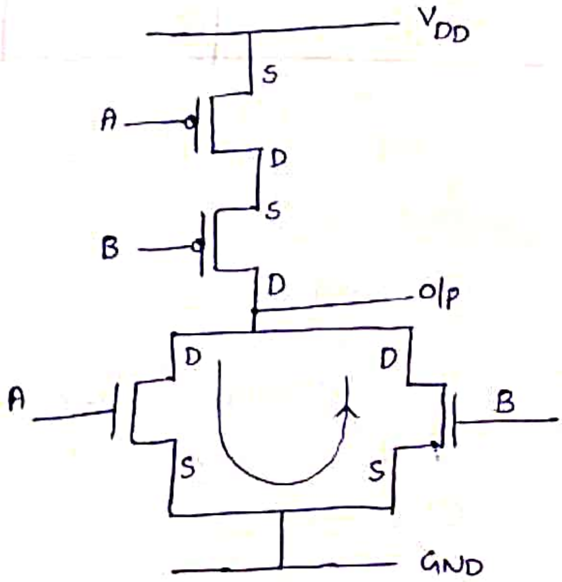
. Draw the stick diagram of 2-Input CMOS NAND gate.

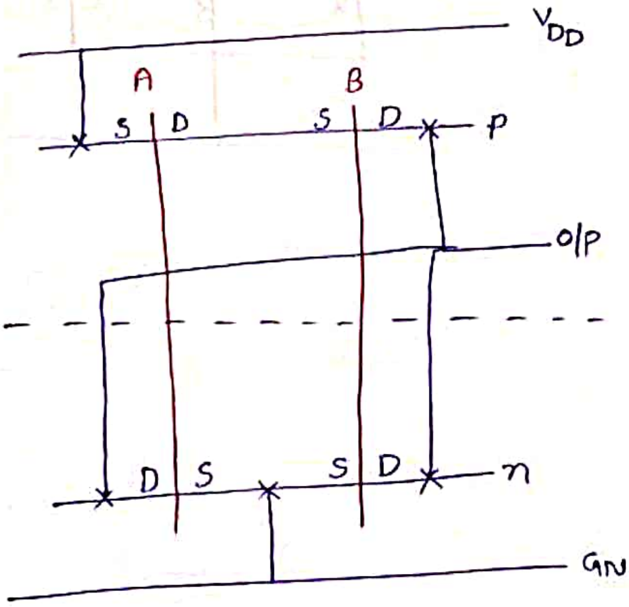


stick diagram [Euler's method]

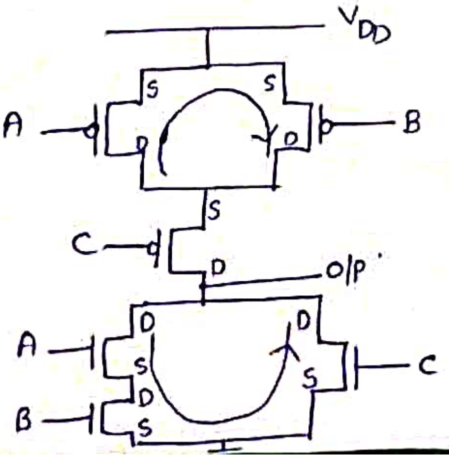


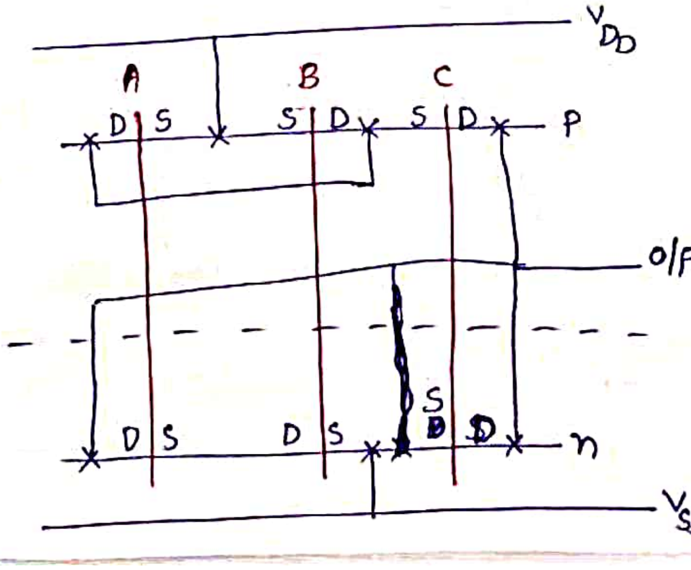
Draw the stick diagram of 2-I/p CMOS NOR gate.



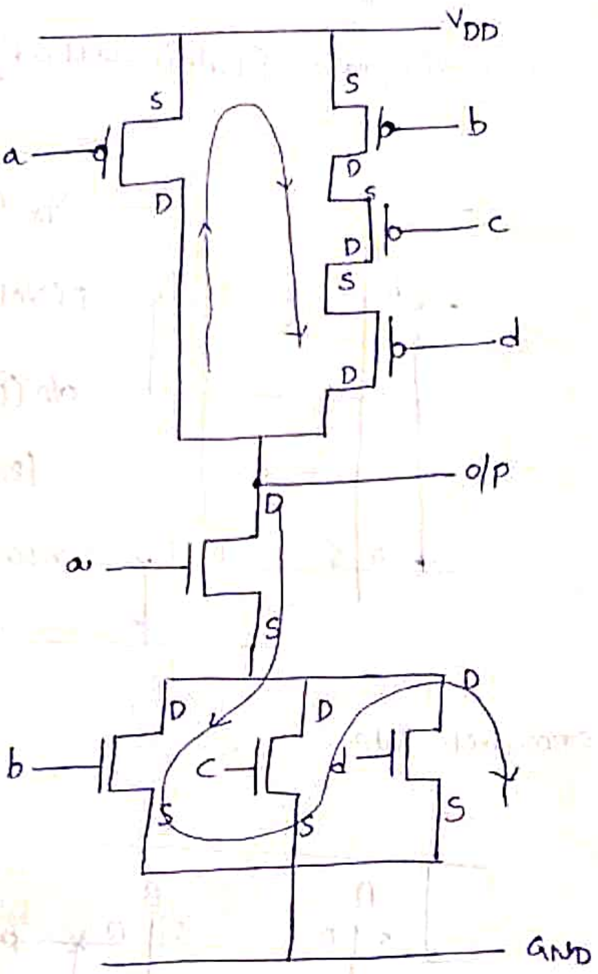


* $Y=\overline{a\cdot b+c}$ use CMOS.

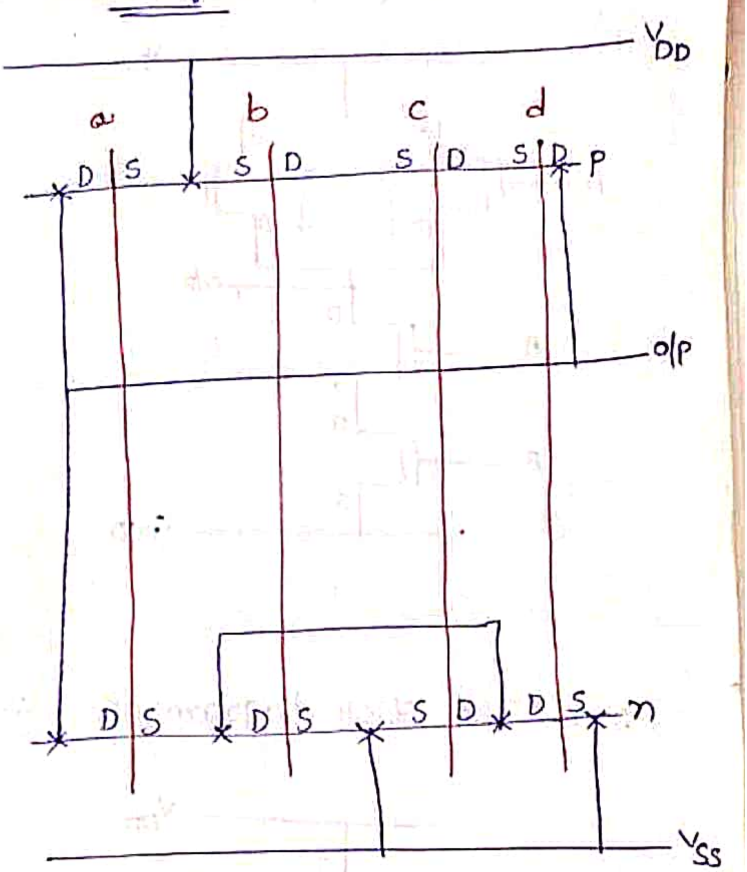




→ Draw the stick diagram of $Y = \overline{a(b+c+d)}$



Stick diagram



G.Venkata Hanuman
Asst. Prof

Basic circuit concepts

Sheet resistance (R_S):-

Sheet resistance is a measure of resistance of a thin films that have a uniform thickness.

Consider a uniform slab of conducting material of resisitivity 'ρ', width 'w', thickness 't', length 'L'.

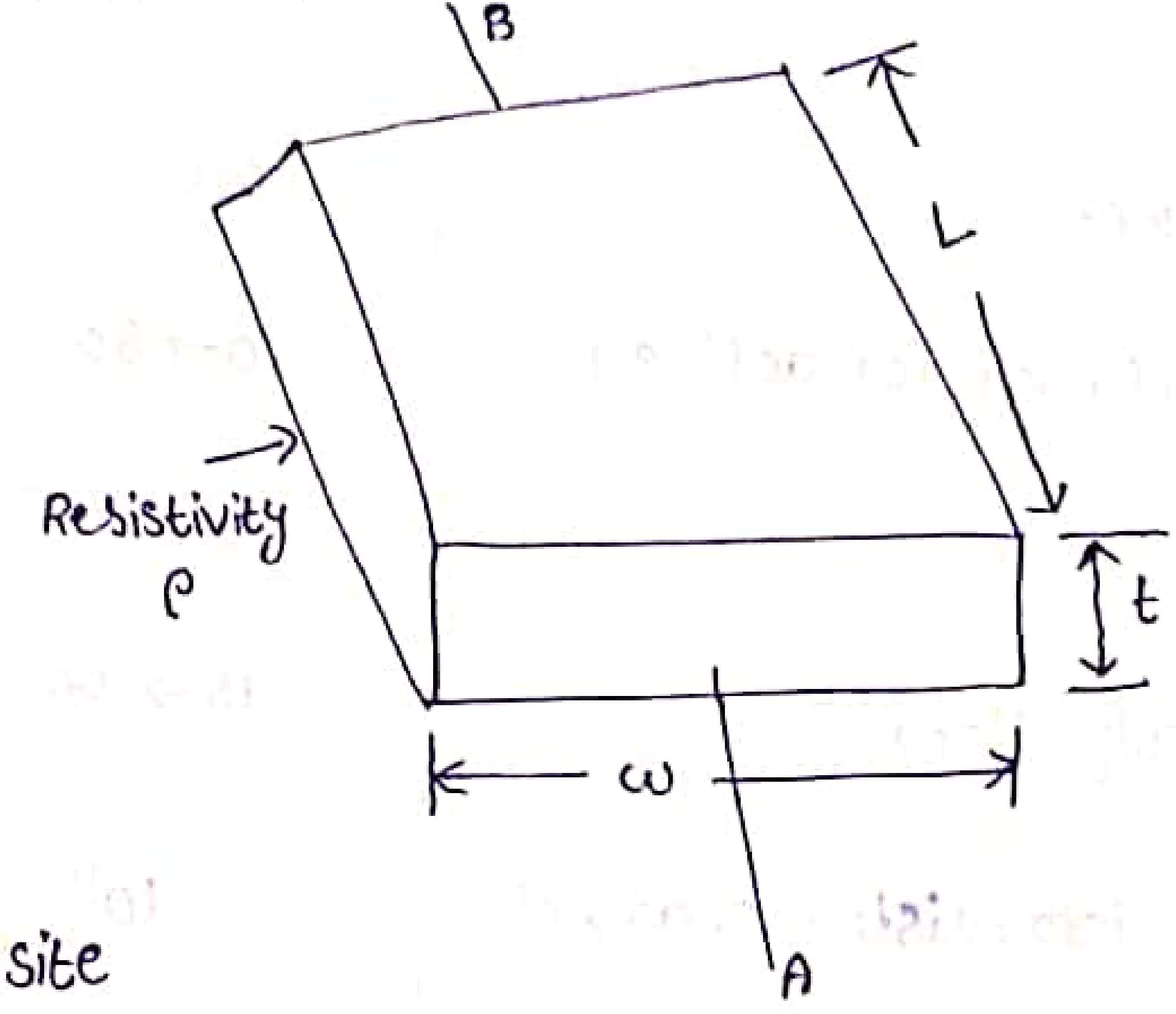


Fig :- Sheet resistance model

Resistance R_{AB} between two opposite faces.

$$R_{AB} = \frac{\rho L}{A} \text{ ohm}$$

where A → Area of cross section

$$R_{AB} = \frac{\rho L}{tw} \text{ ohm}$$

consider $L=w$, since it is a uniform slab,

$$R_{AB} = \frac{\rho L}{tL} = \frac{\rho}{t} = R_S$$

The sheet resistance $R_S = \frac{\rho}{t}$ ohm/square

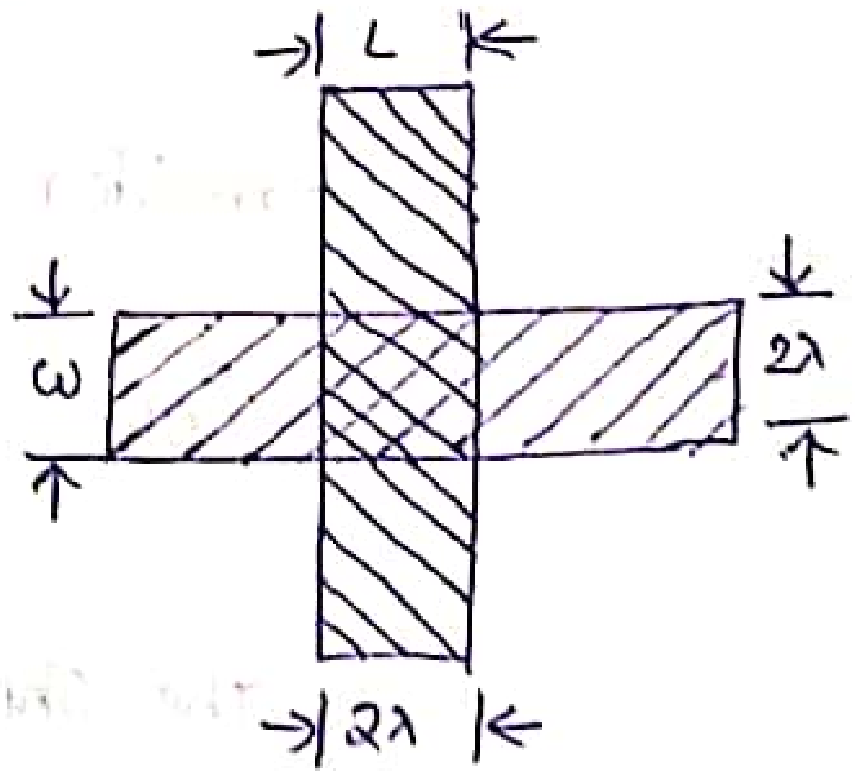
For example, 1μm per side square slab of the material has exactly same resistance as a 1cm per side square slab of the same material if the thickness is same.

Typical sheet resistance R_S of mos Layers for 5μm, orbit 2μm, orbit 1.2μm is given in table.

Layers	R_S ohm per square		
	5 μm	orbit 2μm	orbit 1.2μm
metal	0.03	0.04	0.04
Diffusion (or active)	10→50	20→45	20→45
Slicide	2→4	-	-
Polysilicon	15→100	15→30	15→30
n-transistor channel	10^4	2×10^4	2×10^4
P-transistor channel	2.5×10^4	4.5×10^4	4.5×10^4

Sheet resistance concept applied to mos transistors and Inverters :-

consider n-type transistor has a Length $L=2\lambda$ and width $\omega=2\lambda$.



Resistance defined In terms of Impedance 'z' i.e $z=\frac{L}{\omega}=\frac{2\lambda}{2\lambda}=1$

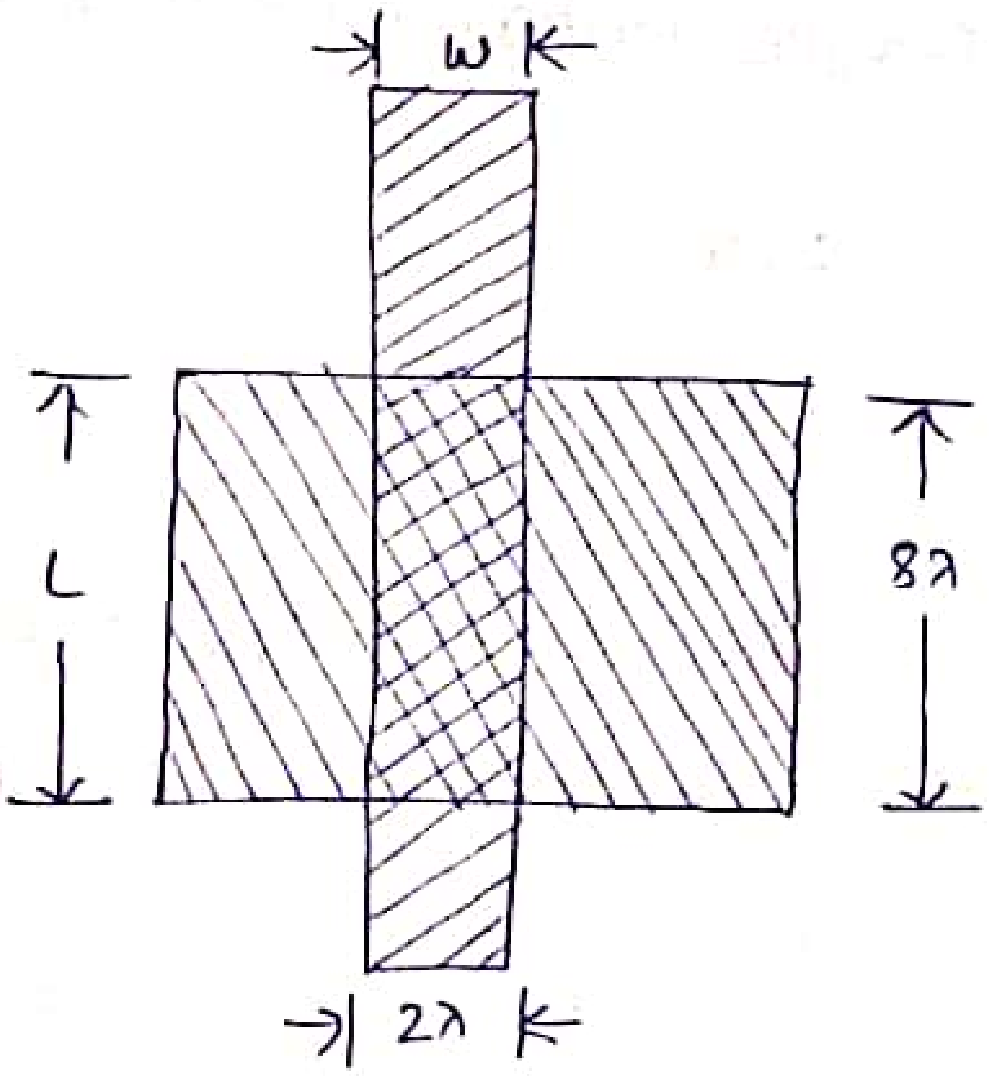
Resistance $(R)=$ 1 square × R_S $\frac{\text{ohm}}{\text{square}}$

$R=1\cdot R_S$

For 5μm technology $R_S=10^4\,\Omega$

For orbit 2μm technology $R_S=2\times10^4\,\Omega$

$Z = \frac{L}{\omega} = \frac{8\lambda}{2\lambda} = 4$



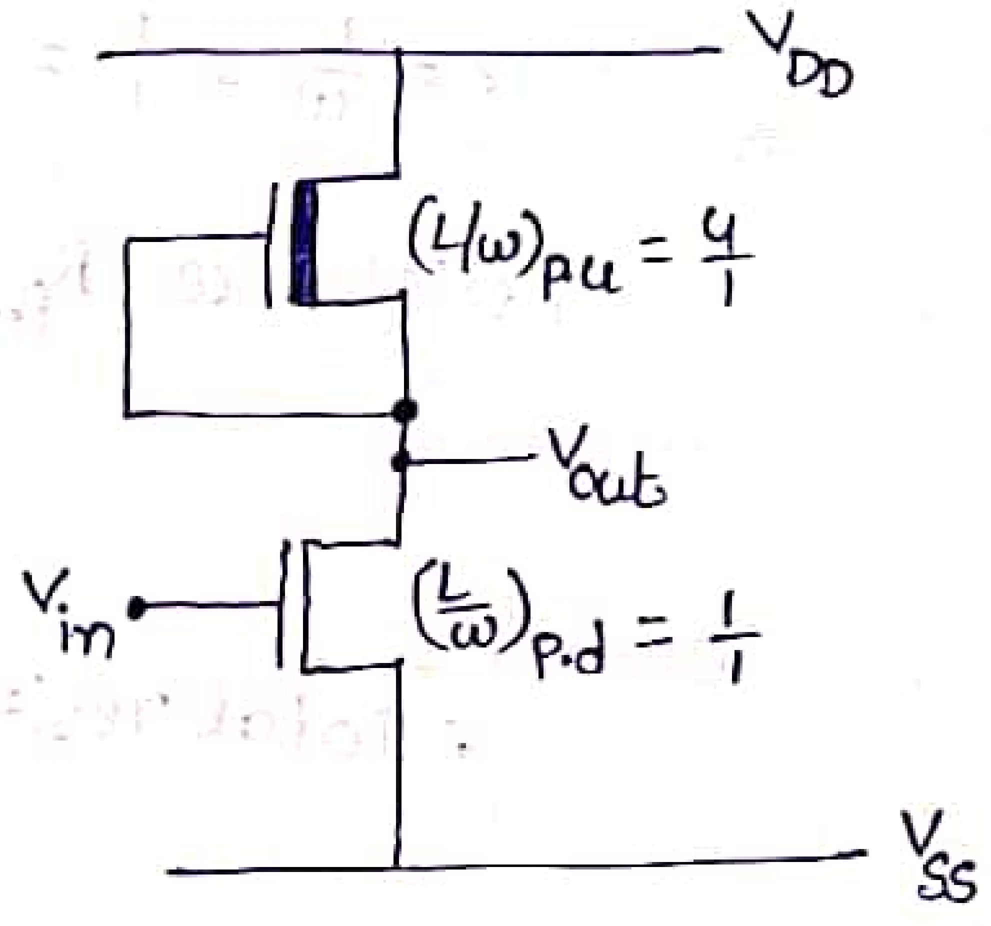
channel resistance, $R = Z \cdot R_S$

$R = 4 \times 10^4 \, \Omega$ (for 5μm technology)

$R = 8 \times 10^4 \, \Omega$ (For orbit 2μm technology)

Caluclation of Resistance of a simple Inverter :-

For depletion mode, Pull-up transistor


V_{DD}
$(L/\omega)_{pu} = \frac{4}{1}$
V_{out}
V_{in}
$(\frac{L}{\omega})_{p.d} = \frac{1}{1}$
V_{SS}

L:ω is 4:1

$Z = \frac{L}{\omega} = \frac{4}{1} = 4$

$R_{p.u} = Z \times R_S$

$= 4 \times 10^4 = 40 \, k\Omega$

Pull-down transistor, L:ω is 1:1

$Z = \frac{L}{\omega} = \frac{1}{1} = 1$

$R_{p.d} = Z \times R_S = 1 \times 10^4 = 10 \, k\Omega$

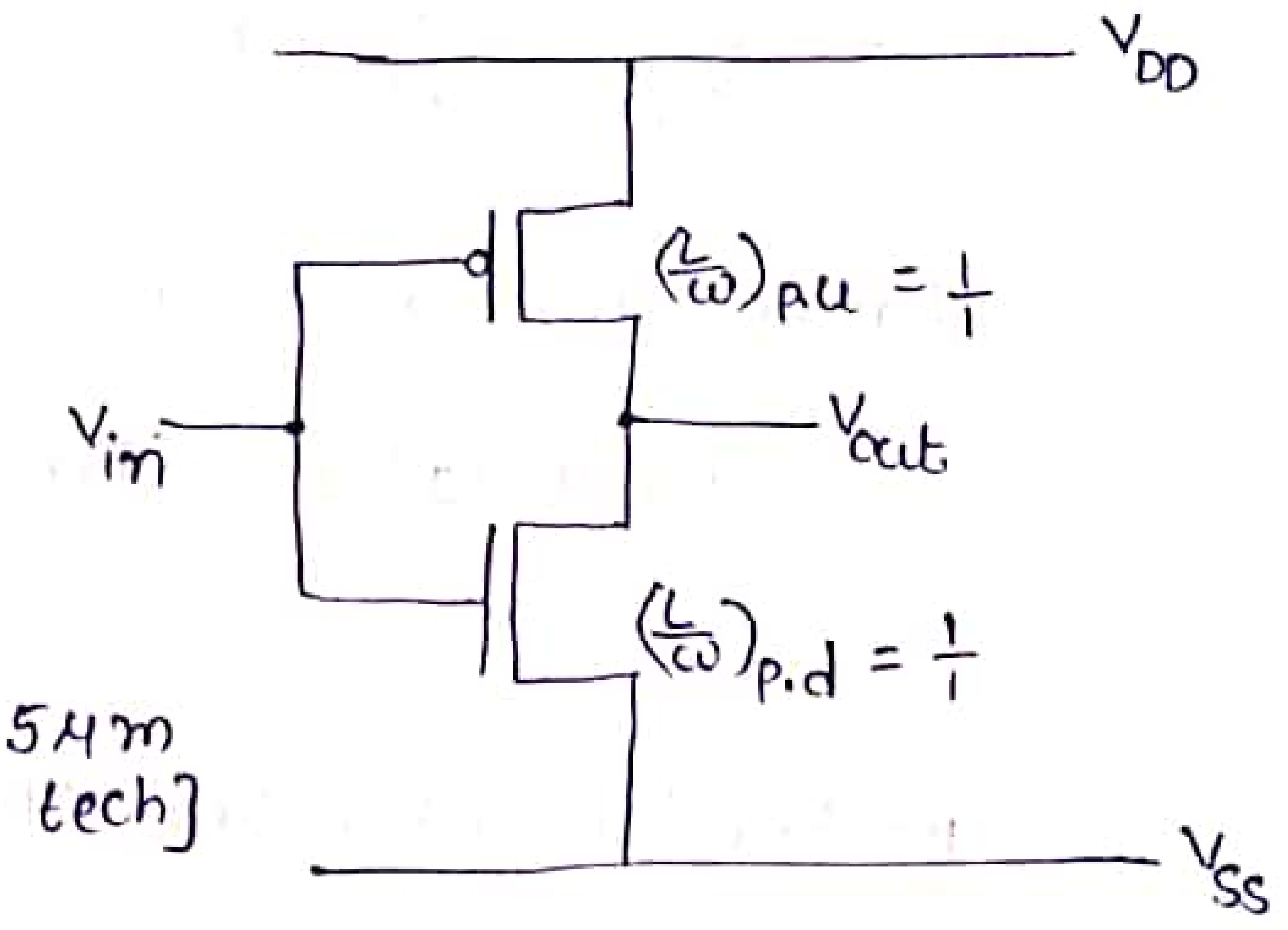
Total resistance $R = R_{p.u} + R_{p.d}$

$= 40 \, k\Omega + 10 \, k\Omega$

$\boxed{R = 50 \, k\Omega}$

consider simple cmos Inverter as shown in fig. below

P-type enhancement mode,


V_{DD}
$(\frac{L}{\omega})_{pu} = \frac{1}{1}$
V_{in}
V_{out}
$(\frac{L}{\omega})_{p.d} = \frac{1}{1}$
V_{SS}

$L:\omega = 1:1$

$z = \frac{L}{\omega} = \frac{1}{1} = 1$

Resistance $R_{p.u} = z \times R_S$

$= 1 \times 10^4 \times 2.5$ [$\because$ 5μm tech]

$= 2.5 \times 10^4\ \Omega$

n-type enhancement mode,

$L:\omega = 1:1$

$z = \frac{L}{\omega} = \frac{1}{1} = 1$

Resistance $R_{p.d} = z \times R_S$

$= 1 \times 10^4$ ($\because$ n-type 5μm technology)

$\therefore$ Total resistance $R = R_{pu} + R_{p.d}$

$= 2.5 \times 10^4 + 10^4$

$\boxed{R = 35\ k\Omega}$

Area capacitances of Layers :-

For any layer, knowing the dielectric thickness,

we can caluclate the area capacitance

$$C = \frac{\varepsilon_0 \varepsilon_{ins} A}{D}$$

where, A → Area of plates

D → thickness of SiO_2

ε_{ins} → Relative permitivity of SiO_2 = 4

ε_0 → permitivity of free space (8.85×10^{-14} F/cm)

Typical values of area capacitances as shown in fig Table.

capacitance	Value in $PF \times 10^{-4} / \mu m^2$ (Relative values in brackets)		
	$5 \mu m$	$2 \mu m$	$1.2 \mu m$
Gate to channel	4 (1.0)	8 (1.0)	16 (1.0)
Diffusion (active)	1 (0.25)	1.75 (0.22)	3.75 (0.23)
Polysilicon to substrate	0.4 (0.1)	0.6 (0.075)	0.6 (0.038)
metal 1 to substrate	0.3 (0.075)	0.33 (0.04)	0.33 (0.02)
metal 2 to substrate	0.2 (0.05)	0.17 (0.02)	0.17 (0.01)
metal 2 to metal 1	0.4 (0.1)	0.5 (0.06)	0.5 (0.03)
metal 2 to Polysilicon	0.3 (0.075)	0.3 (0.038)	0.3 (0.018)

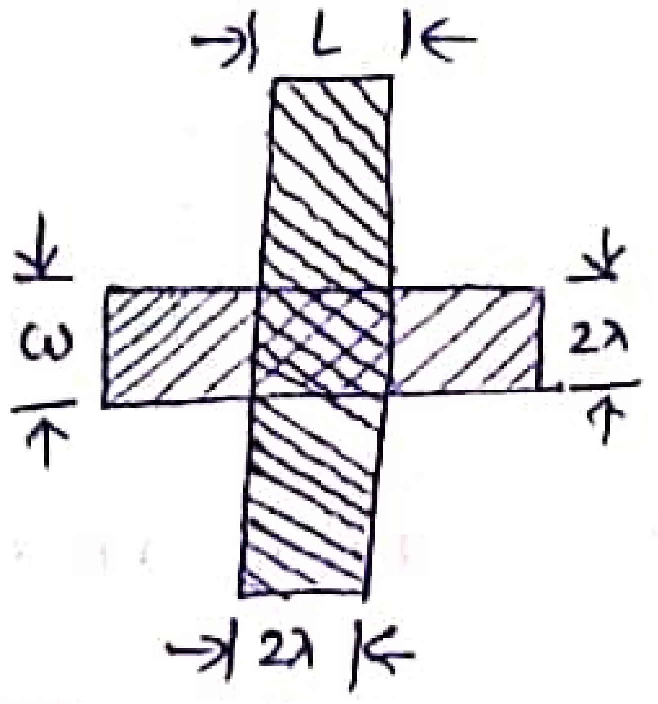
Note :- Relative value = specified value / gate to channel value for that technology

Standard unit of capacitance $\square C_g$:-

The unit is denoted by $\square C_g$.

It is defined as gate to channel capacitance of a mos transistor having $w = L =$ feature size that is standard.

For any mos process C_g is evaluated



i) For $5 \mu m$ mos circuits :-

Area / standard square $= 5 \mu m \times 5 \mu m$

$= 25 \mu m^2$

capacitance value from table $= 4\times10^{-4}\ \text{PF}/\mu m^2$ [∵ Gate to channel]

standard value $\square C_g = 25\mu m^2 \times 4\times10^{-4}\ \text{PF}/\mu m^2$

$= 0.01\ \text{PF}$

ii) For 2µm mos circuits :-

Area/standard square $= 2\mu m \times 2\mu m = 4\mu m^2$

Gate capacitance value $= 8\times10^{-4}\ \text{PF}/\mu m^2$

standard value $\square C_g = 4\mu m^2 \times 8\times10^{-4}\ \text{PF}/\mu m^2$

$= 0.032\ \text{PF}$

iii) For 1.2µm mos circuits :-

Area/standard square $= 1.2\mu m \times 1.2\mu m = 1.44\mu m^2$

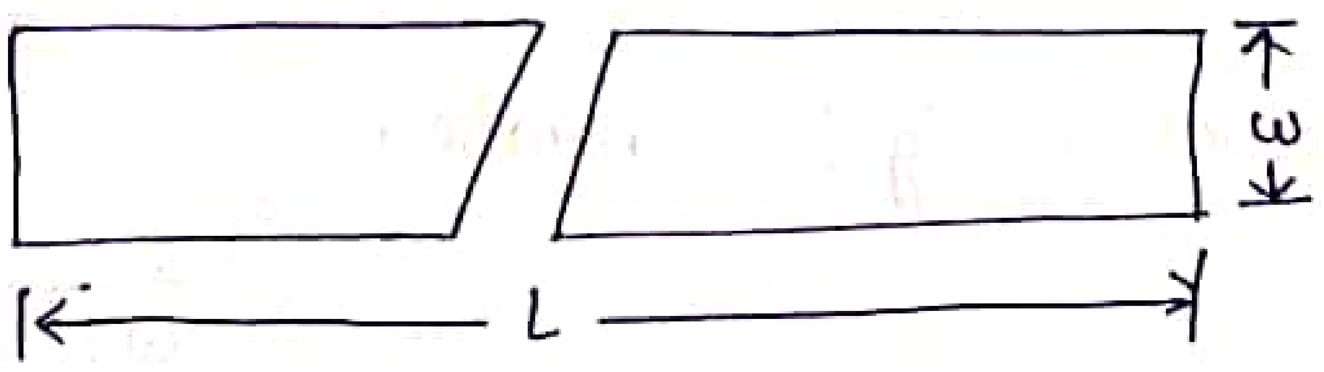
Gate capacitance value $= 16\times10^{-4}\ \text{PF}/\mu m^2$.

standard value $\square C_g = 1.44\mu m^2 \times 16\times10^{-4}\ \text{PF}/\mu m^2$

$= 0.0023\ \text{PF}$

Some area capacitance caluclations :-

consider the area of capacitance



$L = 20\lambda$
$W = 3\lambda$

Total area $= 20\lambda \times 3\lambda = 60\lambda^2$

min. channel area (feature size) $= 2\lambda \times 2\lambda = 4\lambda^2$

$$\text{Relative area} = \frac{\text{Total area}}{\text{feature size}} = \frac{60\lambda^2}{4\lambda^2} = 15$$

1. consider the area in metal 1.

capacitance to substrate = Relative area x Relative c value

$= 15 \times 0.075$

$= 1.125 \square c_g$

2. consider the same area in Poly silicon

capacitance to substrate = Relative area x Relative c value

$= 15 \times 0.1$

$= 1.5 \square c_g$

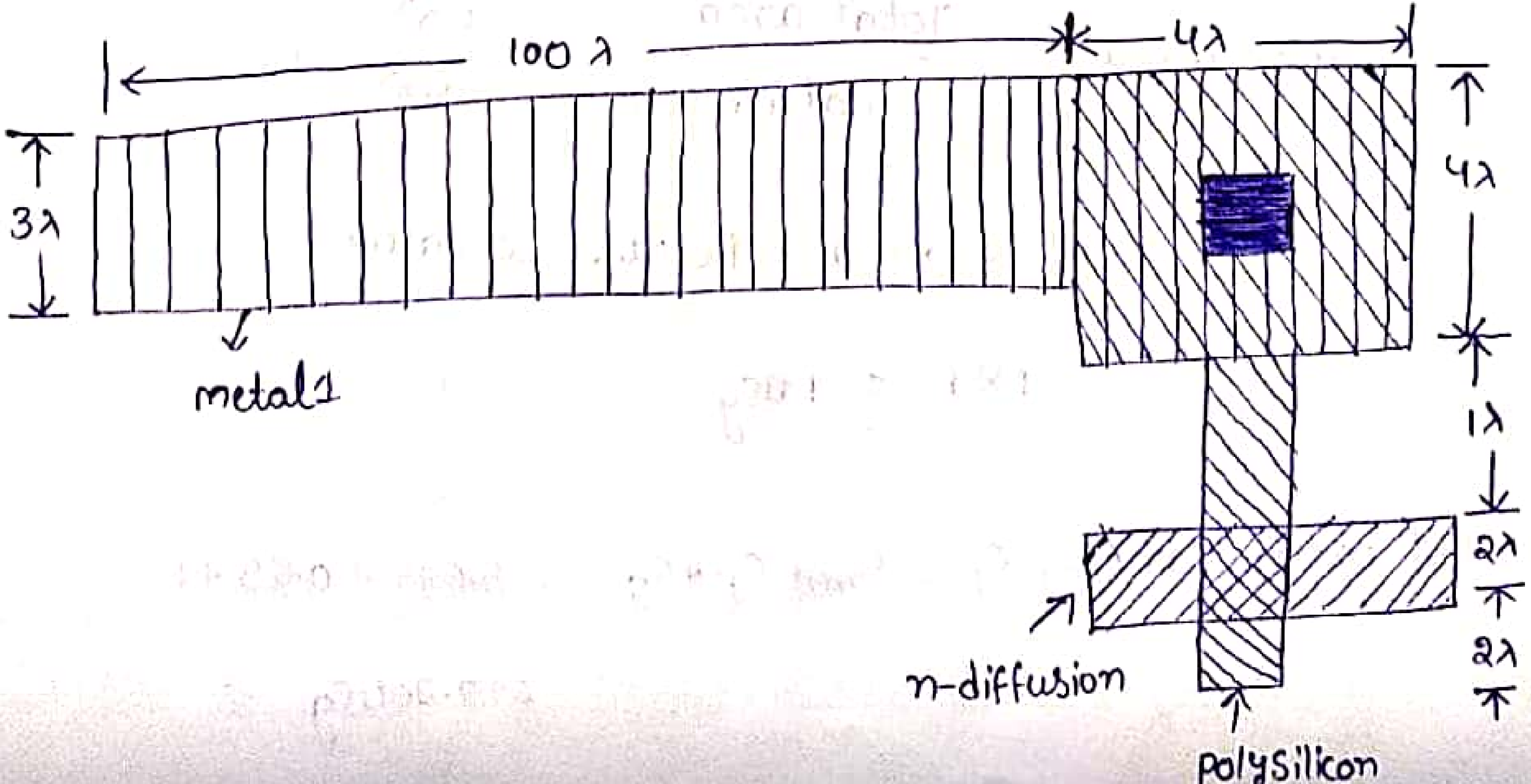
3. consider the same area in n-type diffusion

capacitance to substrate = Relative area x Relative c value

$= 15 \times 0.25$

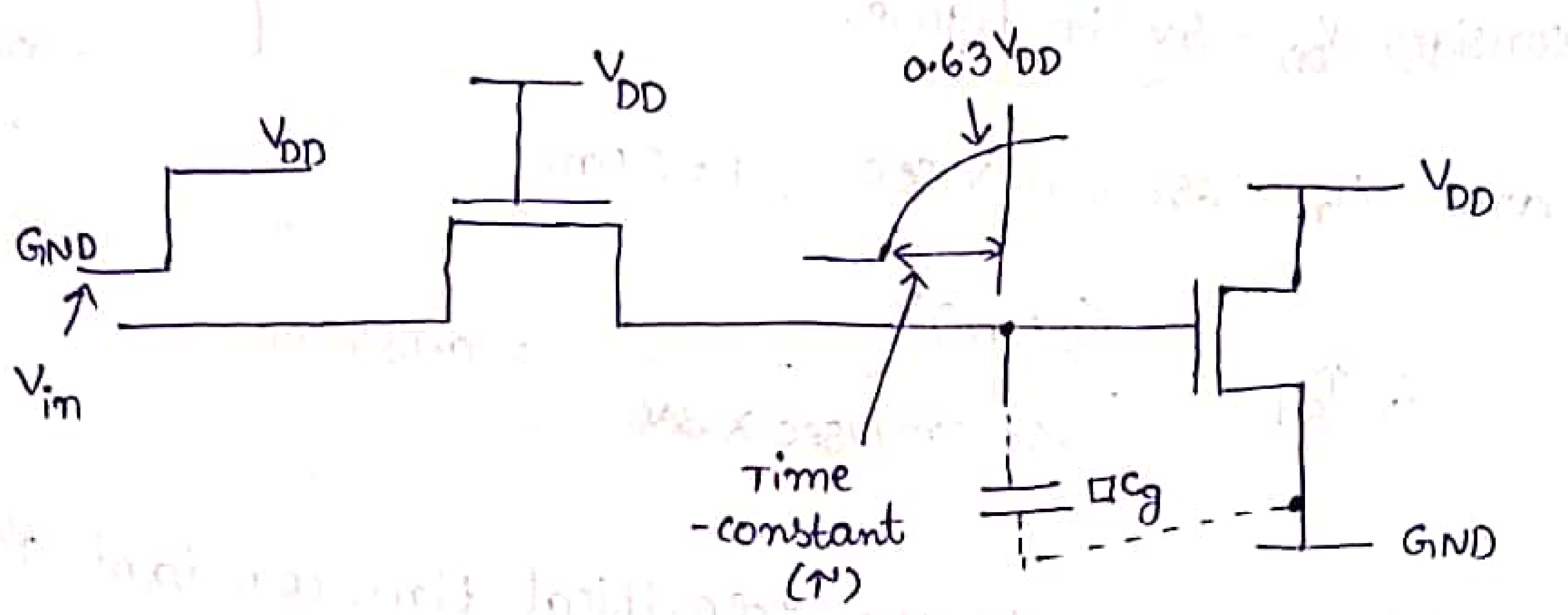
$= 3.75 \square c_g$

consider the capacitance caluclation in multi layer



Delay unit (τ) :-

we have developed the concept of sheet resistance R_S and standard gate capacitance unit $\square C_g$. If we consider the case of one standard (feature size square) gate area capacitance being charged through one feature size square of n-channel resistance (that is, through R_S for an nMOS pass transistor channel) as shown in fig.



$$\text{Time constant } (\tau) = (1R_S \text{ (n-channel)} \times 1 \square C_g) \text{ seconds}$$

5μm technology

$$\tau = 10^4 \Omega \times 0.01 \text{ PF} = 0.1 \text{ nsec}$$

orbit 2μm technology

$$\tau = 2 \times 10^4 \Omega \times 0.0032 \text{ PF} = 0.064 \text{ nsec}$$

orbit 1.2μm technology

$$\tau = 2 \times 10^4 \Omega \times 0.0023 \text{ PF} = 0.046 \text{ nsec}$$

To consider circuit wiring and parasitic capacitances must be allowed for so that the figure taken for τ is often Increased by a factor of 2 to 3 so that for $5\mu m$ circuit the worst case delay will be around $\tau = 0.2$ to 0.3 nsec. ✓

τ is not much different from
× Transit time τ_{sd}.

$$\tau_{sd} = \frac{L^2}{\mu_n V_{ds}}$$

V_{ds} varies as C_g changes from 0 volts to 63% of V_{DD} in period τ.

consider $V_{DD} = 5V$ in figure.

$\left[\because 0.63 V_{DD} = 0.63 \times 5 \cong 3V \right]$

and $\mu_n = 650\ cm^2/V\text{-}sec$, $L = 5\mu m$

$$\therefore \tau_{sd} = \frac{(5\mu m)^2}{650\ cm^2/Vsec \times 3V} = 0.13\ nsec$$

This is very close to the theoritical time constant τ caluclated above.

Thus the transit time and time constant are synchronous and can be Interchangeably used. The stray capacitances are usually allowed for doubling the theoritical values caluclated.

For $5\mu m$ mos technology, $\tau = 0.3$ nsec

For orbit $2\mu m$ mos technology, $\tau = 0.2$ nsec

For orbit $1.2\mu m$ mos technology, $\tau = 0.1$ nsec

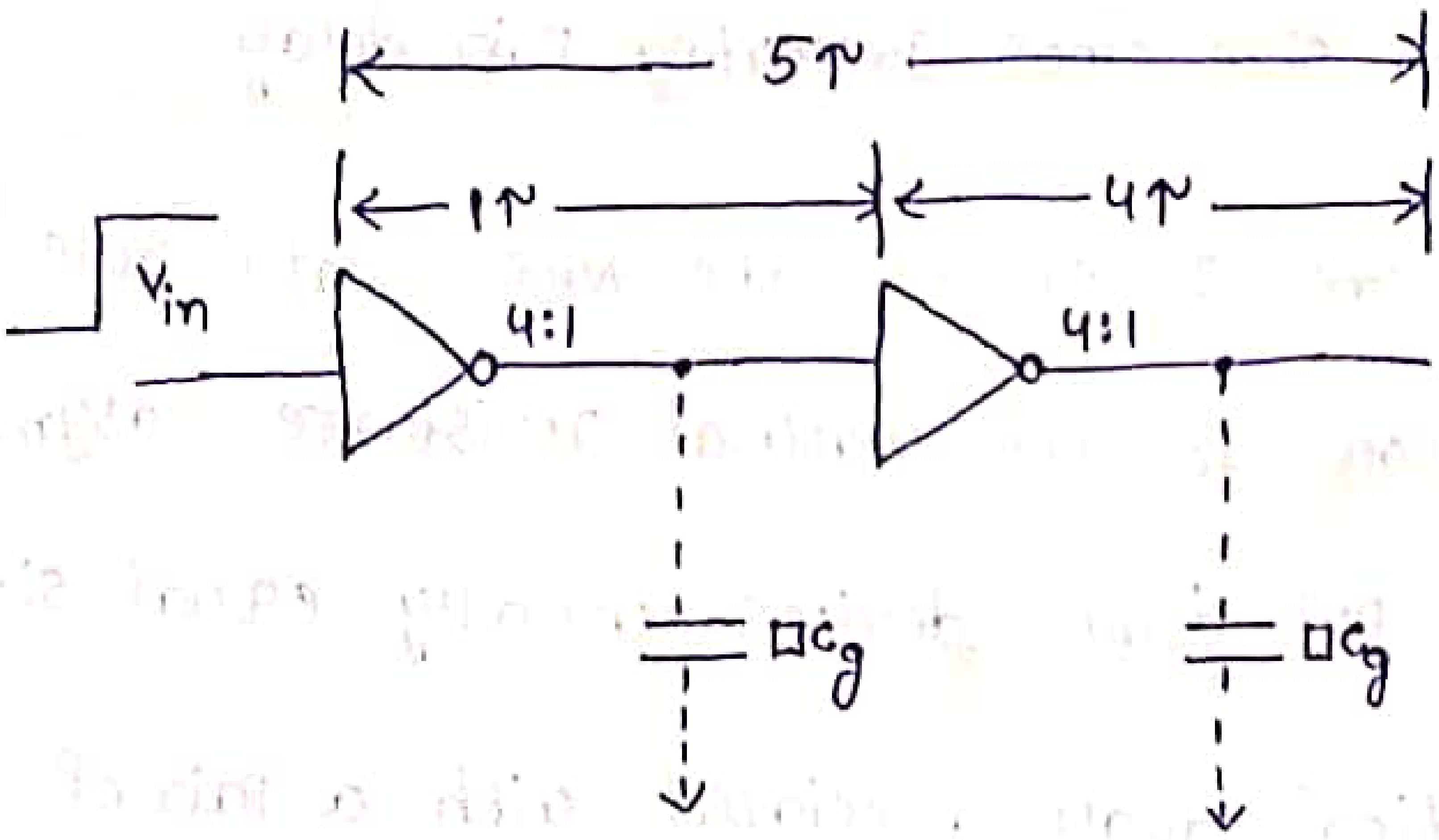
Inverter delays :-

consider the basic 4:1 ratio nmos Inverter. In order to achieve 4:1 $Z_{p.u}$ to $Z_{p.d}$ ratio, $R_{p.u}$ will be $4R_{p.d}$ and if $R_{p.d}$ is contributed by the minimum size transistor then

$$R_{p.u} = 4R_s = 4\times10^4 = 40k\Omega$$

$$R_{p.d} = 1\cdot R_s = 1\times10^4 = 10k\Omega$$

so that the delay associated with the Inverter will depend on whether it is being turned on or off.



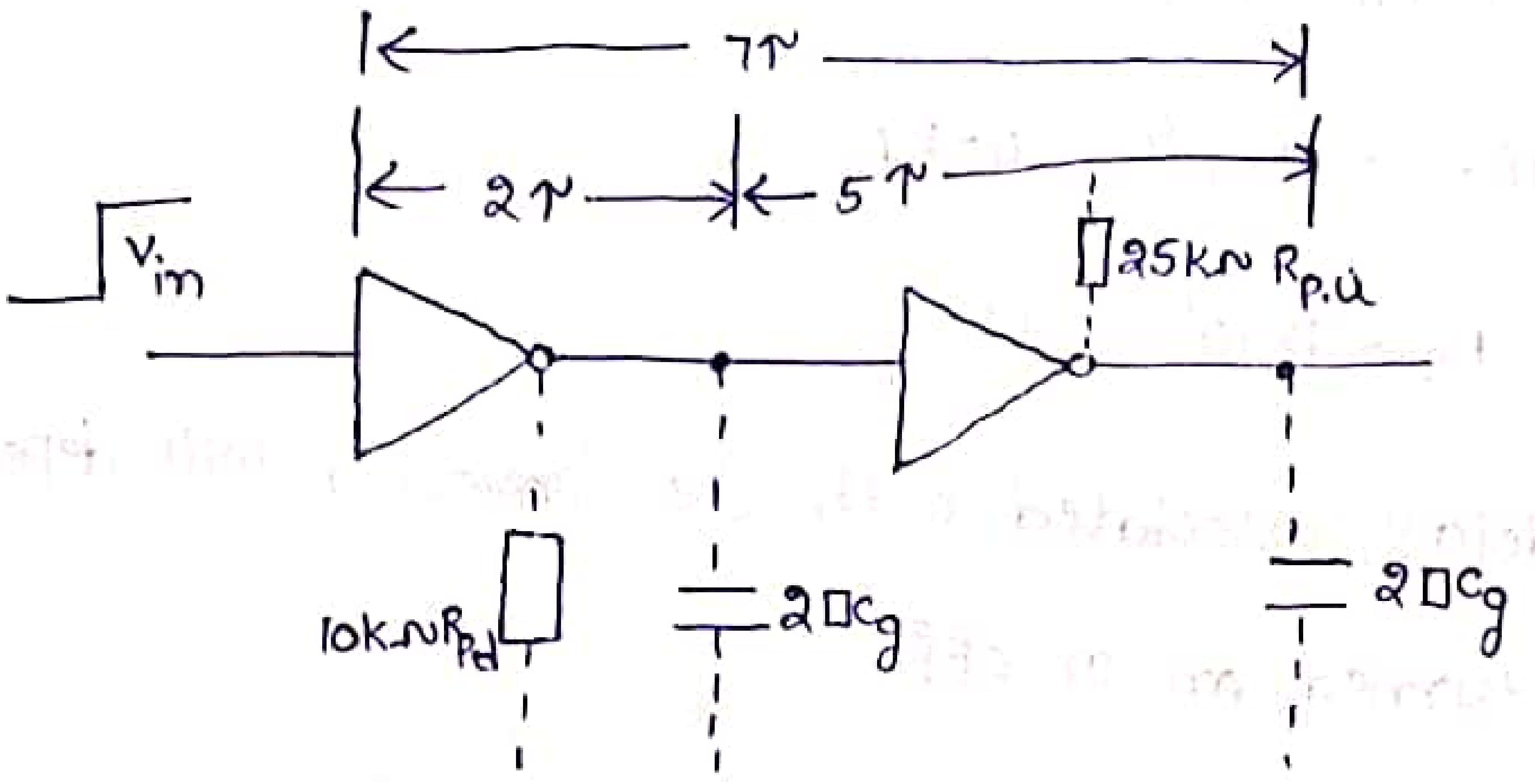
Fig①:- Nmos Inverter pair delay

consider a pair of cascaded Inverters, then the delay over the pair will be constant irrespective of the sense of the logic level transition of the Input to the first. This is shown in the fig. below and assuming $\tau = 0.3\,nsec$ and making no extra allowances for wiring capacitance, over all delay $\tau + 4\tau = 5\tau$.

Generally, the delay through a pair of similar NMOS Inverter is

$$T_d = \left[1 + \frac{Z_{p.u}}{Z_{p.d}}\right]\tau$$

$$= \left[1 + \frac{4}{1}\right]\tau = 5\tau$$



Fig②:- minimum size cmos Inverter pair delay

When considering cmos Inverters, the NMOS ratio rule no longer applies, but we must allow for the natural ~~resistance~~ asymmetry in R_s between pull-up and pull-down devices usually equal size.

fig ② shows the theoritical delay associated with a pair of minimum size Lambda based Inverters. Note that the gate capacitance $(=2\square C_g)$ is double that of the NMOS Inverter since the Input to a cmos Inverter is connected to both transistor gates. Note the allowances made for different channel resistances.

The Asymmetry of resistance values can be eliminated by Increasing the width of P-device channel by a factor 2 to 3. Note that gate Input capacitance of P-device transistor Increased by same factor.

Estimation of cmos Inverter delay :-

cmos Inverter delay estimated by splitting the output transitions into rise time τ_r and fall time τ_f corresponding to charging and discharging of the capacitive load C_L.

i) Rise time estimation :-

Assume that P-device stays in saturation for the entire charging period of the load capacitor C_L.

The current may be modeled as shown in fig. below

Saturation current for P-type transistor is given by

$$I_{dsp} = \frac{\beta_p (V_{gs} - |V_{tp}|)^2}{2} \quad —①$$

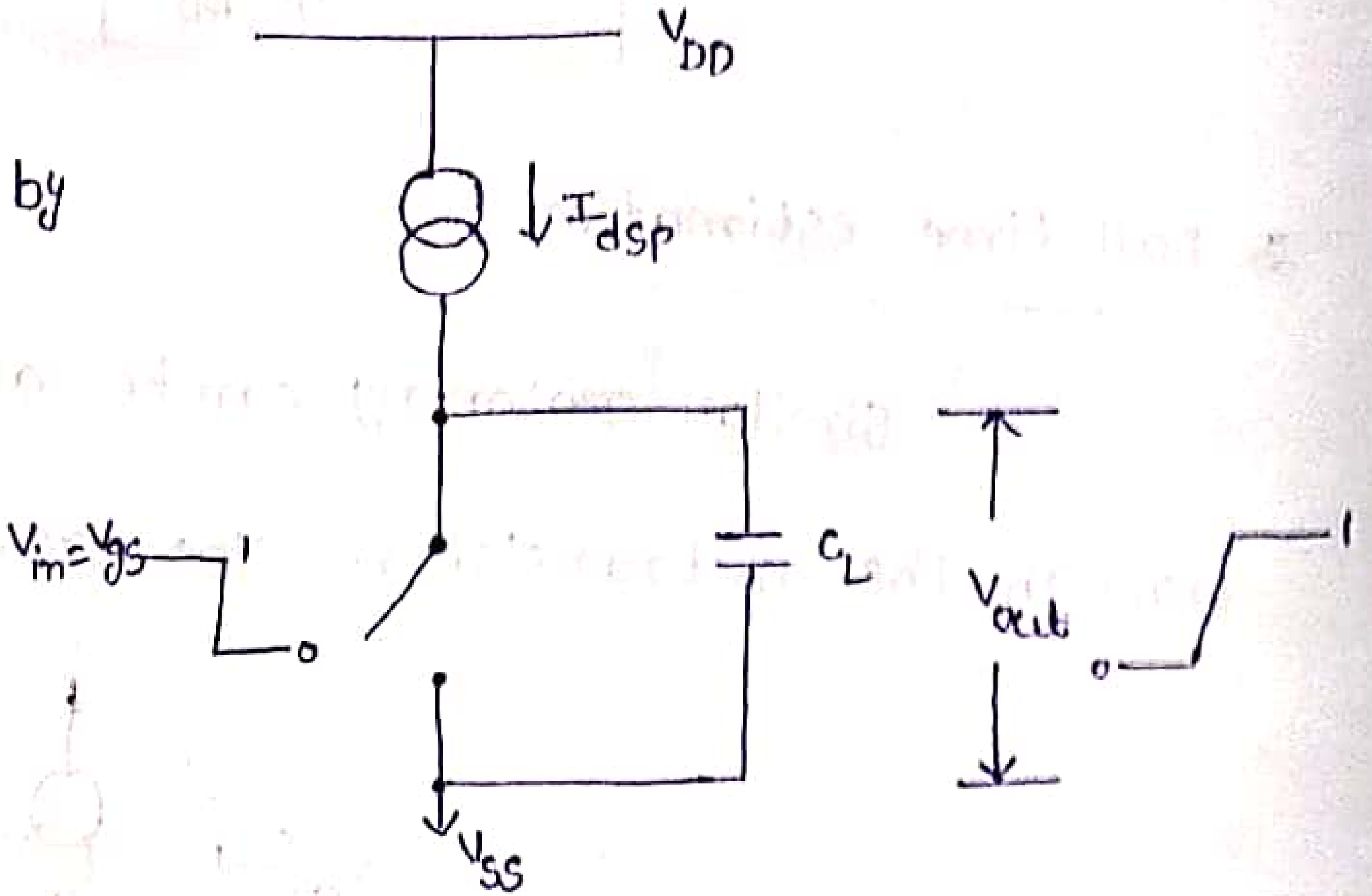


Fig :- Rise-time model

This current charges C_L and magnitude approximately constant,

$$V_{out} = \frac{I_{dsp}\, t}{C_L} \qquad \left[\because V_{out} = \frac{1}{C} \int i\, dt = \frac{1}{C_L} \int I_{dsp}\, dt = \frac{I_{dsp}}{C_L} \cdot t \right]$$

$$\Rightarrow t = \frac{V_{out}\, C_L}{I_{dsp}} \quad —②$$

$$\Rightarrow t = \frac{V_{out}\, C_L \times 2}{\beta_p (V_{gs} - |V_{tp}|)^2} \qquad [\because \text{sub. eq } ① \text{ in } ②]$$

Assume $t = \tau_r$ when $V_{out} = V_{DD}$,

$$\Rightarrow \tau_r = \frac{2 V_{DD} C_L}{\beta_p (V_{DD} - |V_{tp}|)^2}$$

consider $V_{tp} = 0.2 V_{DD}$,

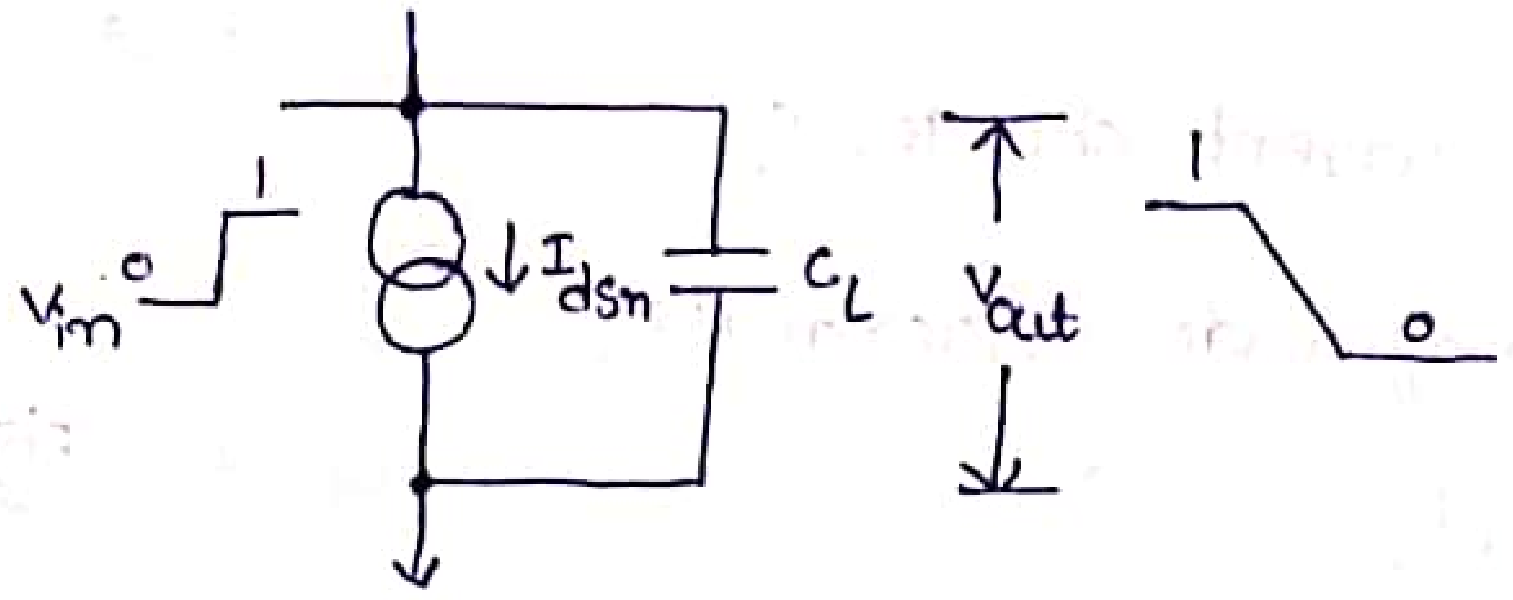
$$\tau_r = \frac{2 V_{DD} C_L}{\beta_p (V_{DD} - |0.2 V_{DD}|)^2} = \frac{2 V_{DD} C_L}{\beta_p (0.8 V_{DD})^2}$$

$$= \frac{2 C_L}{\beta_p \, 0.64 V_{DD}}$$

$$\boxed{\tau_r = \frac{3 C_L}{\beta_p V_{DD}}} \quad — (3)$$

2. Fall-time estimation :-

Similar reasoning can be applied to the discharge of C_L through the n-transistor. The circuit model as shown in fig. below



Hence, the falltime

$$\tau_f = \frac{3 C_L}{\beta_n V_{DD}} \quad — (4)$$

Fig :- Fall-time model

From eq (3), (4) we deduce that

$$\frac{\tau_r}{\tau_f} = \frac{\beta_n}{\beta_p}$$

But $\mu_n = 2.5 \mu_p$ hence $\beta_n = 2.5 \beta_p$, so that the rise time is slower by a factor 2.5 when both the n and p-devices are minimum sizes.

To achieve symmetrical operation using minimum channel length we need to keep $\omega_p = 2.5\,\omega_n$ and minimum size Lambda based geometries this would result in Inverter having Input capacitance of $1\square C_g$ (n-device) + $2.5\,\square C_g$ (p-device) $= 3.5\,\square C_g$ in total.

This is simple model is quite adequate for most Practical situations, but it should be recognized that it gives optimistic results. It provide rise time and fall time as

1. τ_r and τ_f are proportional to $1/V_{DD}$
2. τ_r and τ_f are proportional to C_L
3. $\tau_r = 2.5\,\tau_f$ for equal n-and p-transistor geometries.

Driving Large capacitive Loads :-

The problem of driving comparatively large capacitive loads arises when signals must be propagated from chip to off chip destinations.

Generally off chip capacitances,

$$C_L \geq 10^4\,\square C_g$$

The capacitances of this order must be driven through low resistances, otherwise excessively long delays will occur.

1. cascaded Inverters as drivers :-

For driving large capacitive loads, Inverters should present Low Pull-up and Pull-down resistances.

It means that mos devices must be designed with Low L:W ratios to have Low resistance values for $Z_{p.u}$ and $Z_{p.d}$. These channels must be made very wide to reduce resistance value. which consequence makes the Inverter occupy a larger area.

Another limitation of L is that it can not be reduced below the minimum feature size which makes L:W ratio large. Hence gate region area LxW becomes significantly and large capacitance is Presented at the Input. which in turn slows down the rates of change of voltage which takes place at the Input.

This situation can be Improved by using N cascaded Inverters, each one of which is larger than the preceding stage by a width factor f as shown in fig.

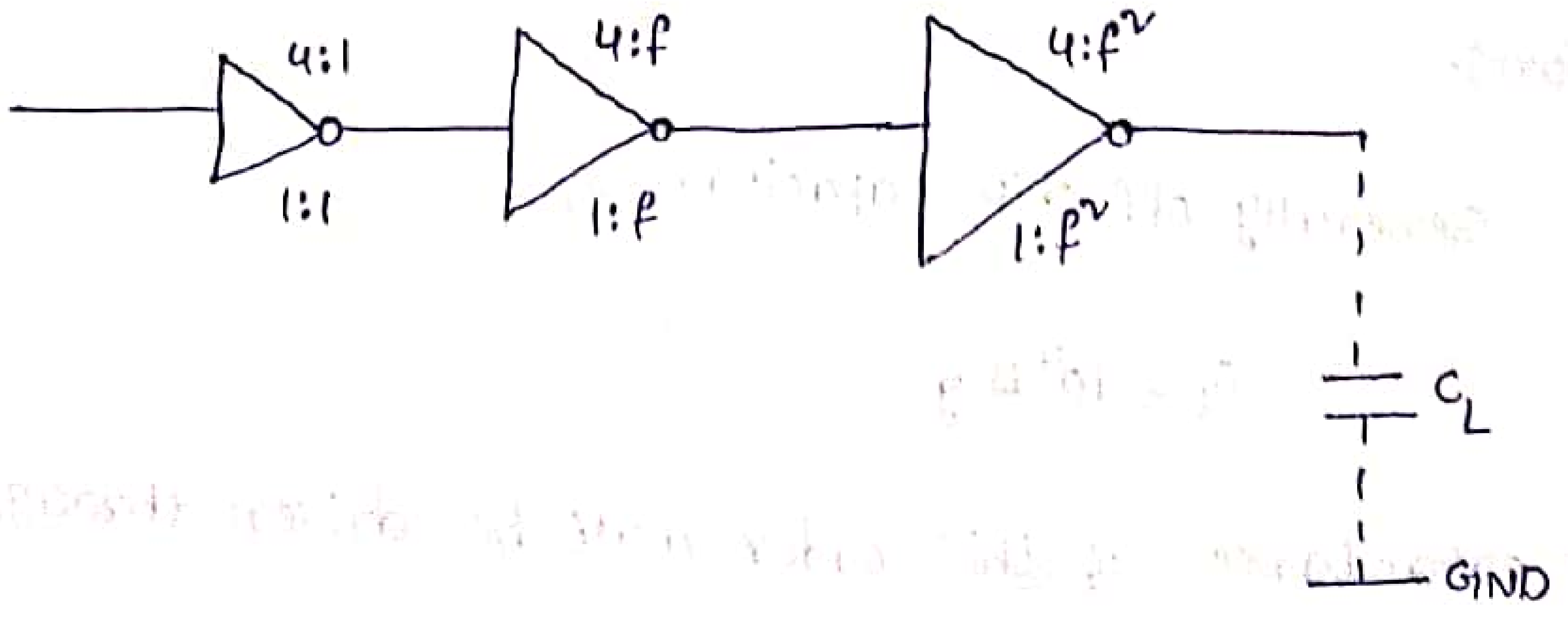


Fig :- Driving large capacitive Loads

The capacitive load presented at the Inverter Input Increases in Proportion to the Increasing width and area also Increases.

Let ΔV_{in} Indicate a logic 0 to 1 transition and ∇V_{in} Indicates Logic 1 to 0 transistion of Input voltage V_{in}.

$$\text{Delay Per stage} = f\tau \text{ for } \Delta V_{in}$$

$$= 4f\tau \text{ for } \nabla V_{in}$$

Total delay per NMOS Pair $= 5f\tau$.

similarly, delay per CMOS Pair $= 7f\tau$.

Let $y = \frac{C_L}{\square C_g} = f^N$ $\quad$ $f, N \rightarrow$ Independent Values.

To determine value of 'f' which will minimize the overall delay for given value of y.

$$\ln(y) = N \ln(f)$$

$$N = \frac{\ln(y)}{\ln(f)}$$

$N \rightarrow$ even,

$$\text{Total delay} = \frac{N}{2} 5f\tau = 2.5 N f\tau \text{ (NMOS)}$$

$$(\text{or}) = \frac{N}{2} 7f\tau = 3.5 N f\tau \text{ (CMOS)}$$

In all cases, delay $\propto Nf\tau = \frac{\ln(y)}{\ln(f)} f\tau$

Total delay is minimized if 'f' assumes the value e.

Assume $f = e$,

No.of stages $N = \ln(Y)$

For N even, $t_d = 2.5 e N \tau$ (NMOS)

$= 3.5 e N \tau$ (CMOS)

N odd, $t_d = [2.5(N-1)+1] e\tau$ (NMOS)

$= [3.5[N-1)+2] e\tau$ (CMOS)

for ΔV_{in}

and $t_d = [2.5(N-1)+4] e.\tau$ (NMOS)

$= [3.5(N-1)+5] e\tau$ (CMOS)

for ∇V_{in}

2. Super buffers:-

Theare are 2 types

1. Inverting type nmos super buffer
2. Non-Inverting type Nmos super buffer

1. Inverting type Nmos super buffer :-

Consider a positive going logic transition V_{in} at the Input, it will be seen that the Inverter formed by T_1, T_2 is turned ON and gate of T_3 is pulled down toward 'o' volts with a small delay.

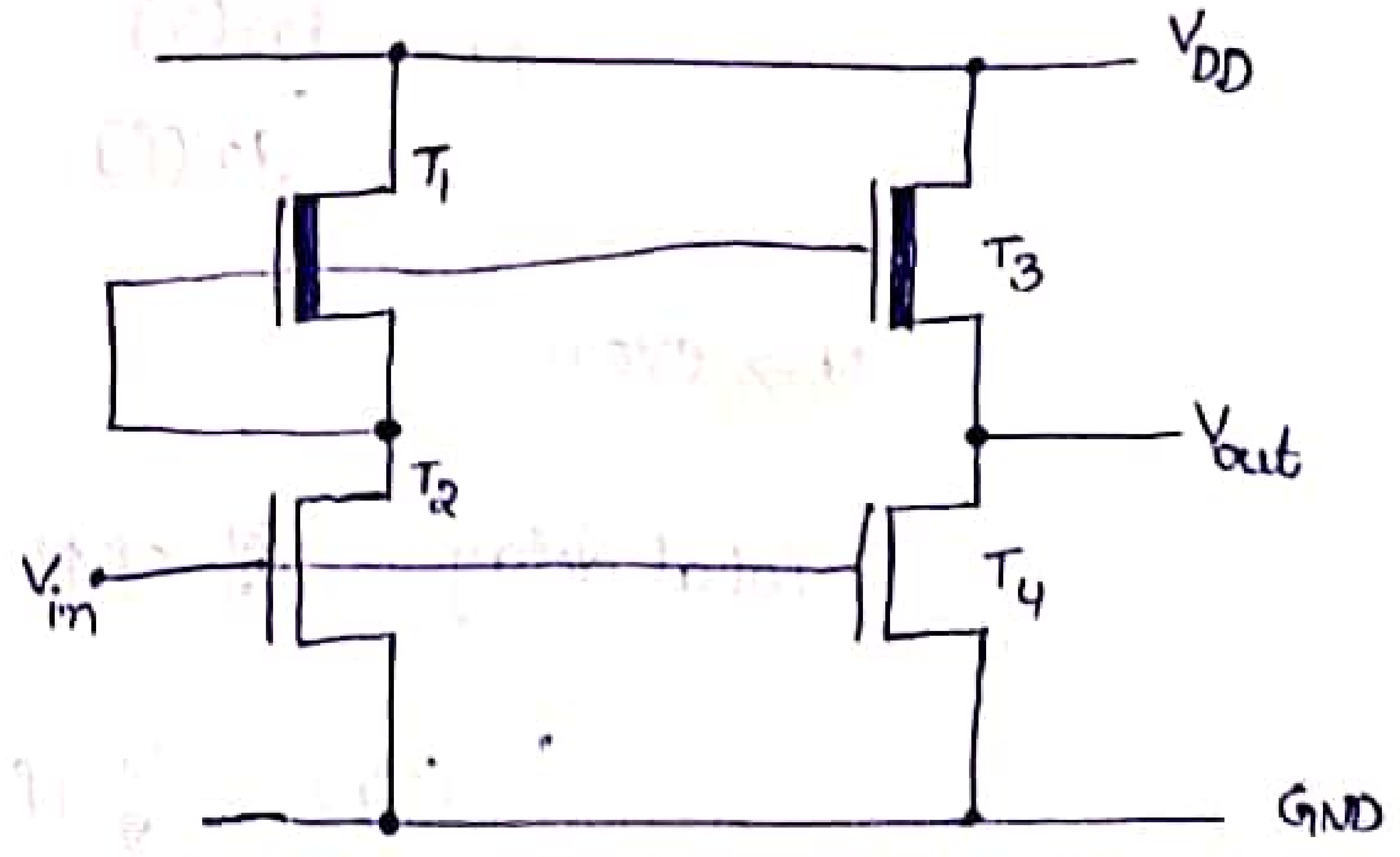


Fig:- Inverting type NMOS super buffer

Thus T_3 is cut off while T_4 is turned on and output is ~~allowed~~ pulled down Quickly.

Now consider the opposite transition : when V_{in} drops to '0' volt, then gate of T_3 is allowed to rise quickly to V_{DD}. Thus T_4 also turned off by V_{in}, T_3 is made to conduct with V_{DD} on its gate, i.e. twice the average voltage that would appear if the gate was connected to source in the conventional NMOS Inverter. Now, as $I_{ds} \propto V_{gs}$, doubling the effective V_{gs}, Increases the current and reduce the delay in charging at load capacitance of the output, so more symmetrical transitions are achieved.

I/P	O/P
0	1
1	0

2. Non-Inverting type NMOS Super buffer :-

To gain an Idea of the effectiveness of super buffers designs, we note that the structures fabricated in 5μm technology are capable of driving capacitance of 2pF with rise time of 5nsec.

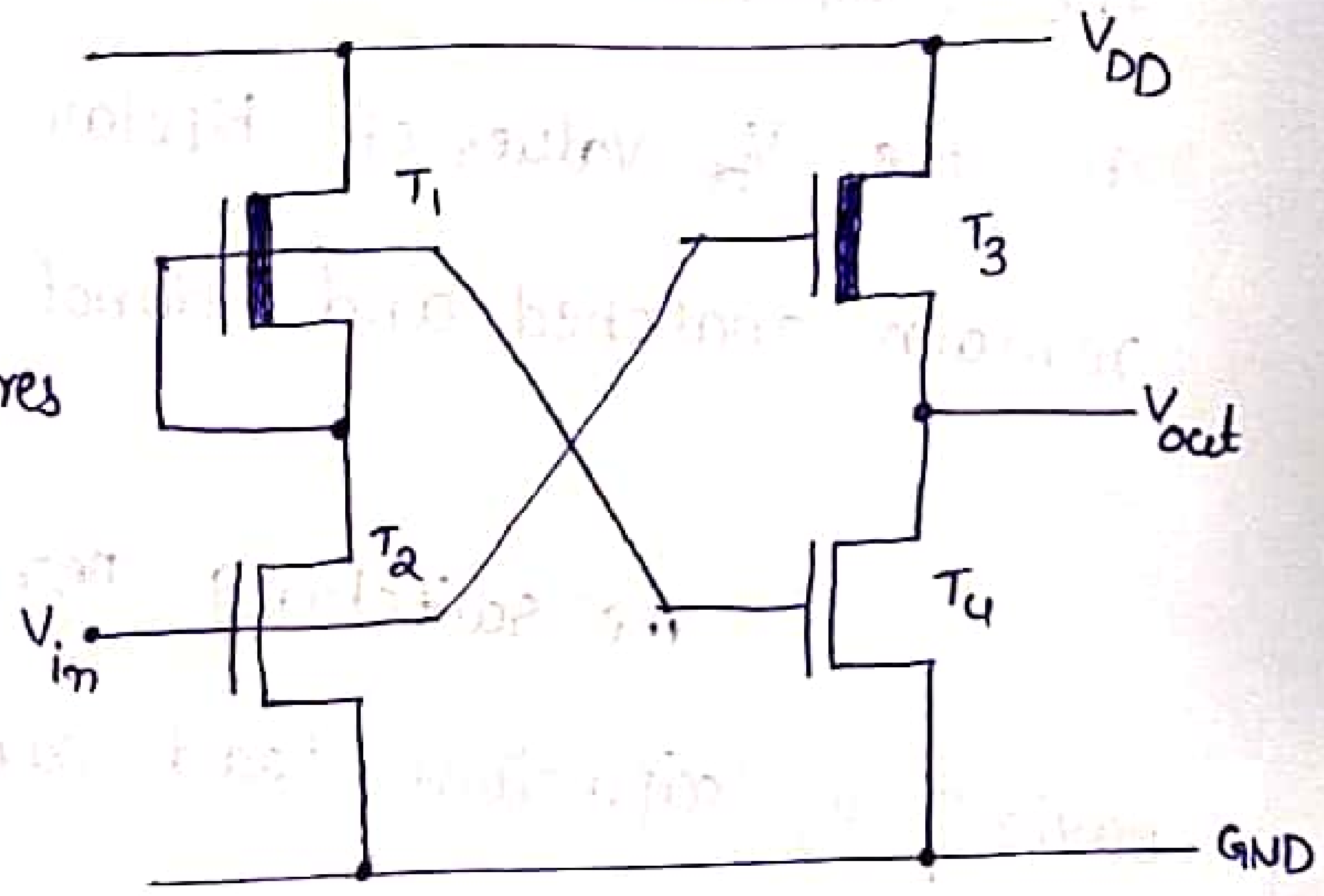


Fig:- Non-Inverting type NMOS Super buffer

I/P	O/P
1	1
0	0

3. BiCMOS drivers:-

BiCMOS technology presents the possibility of using bipolar transistor drivers as the output stage of Inverter and logic gate circuits.

→ In bipolar transistors, there is an exponential dependence of the collector current I_C on base to emitter voltage V_{BE}. Hence the bipolar transistors operated with much smaller input voltage swings than mos transistors and still switch larger currents. only a small amount of charge must be moved during switching.

→ Another consideration in bipolar devices is that of the temperature effect on input voltage V_{be}. V_{be} dependent on base width W_B, doping level N_A, electron möbility μ_n and collector-current I_c, it is linearly dependent on temperature. Now the temperature ~~depe.~~ differences across an Ic are not very high Thus the V_{be} values of bipolar devices spread over the chip remain matched and donot differ by more than few milli volts.

The switching performance of a bipolar transistor having a capacitive load can be analyzed with help of equivalent circuit as shown in figure

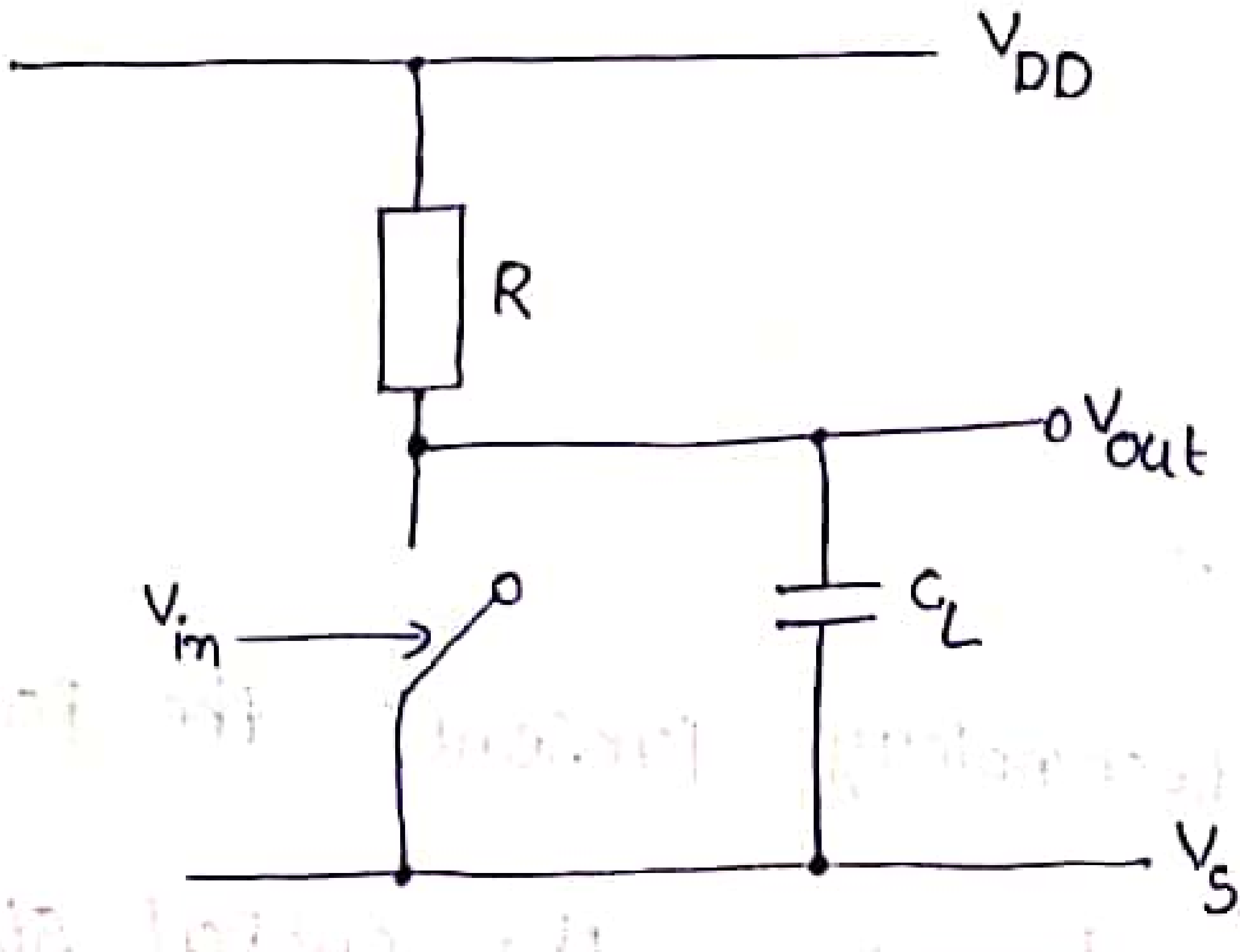


Fig:- Driving ability of bipolar transistor

The time Δt required to change the output voltage V_{out} by an amount equal to Input voltage V_{in} is

$$\Delta t = \frac{C_L}{g_m}$$

C_L → Load capacitance

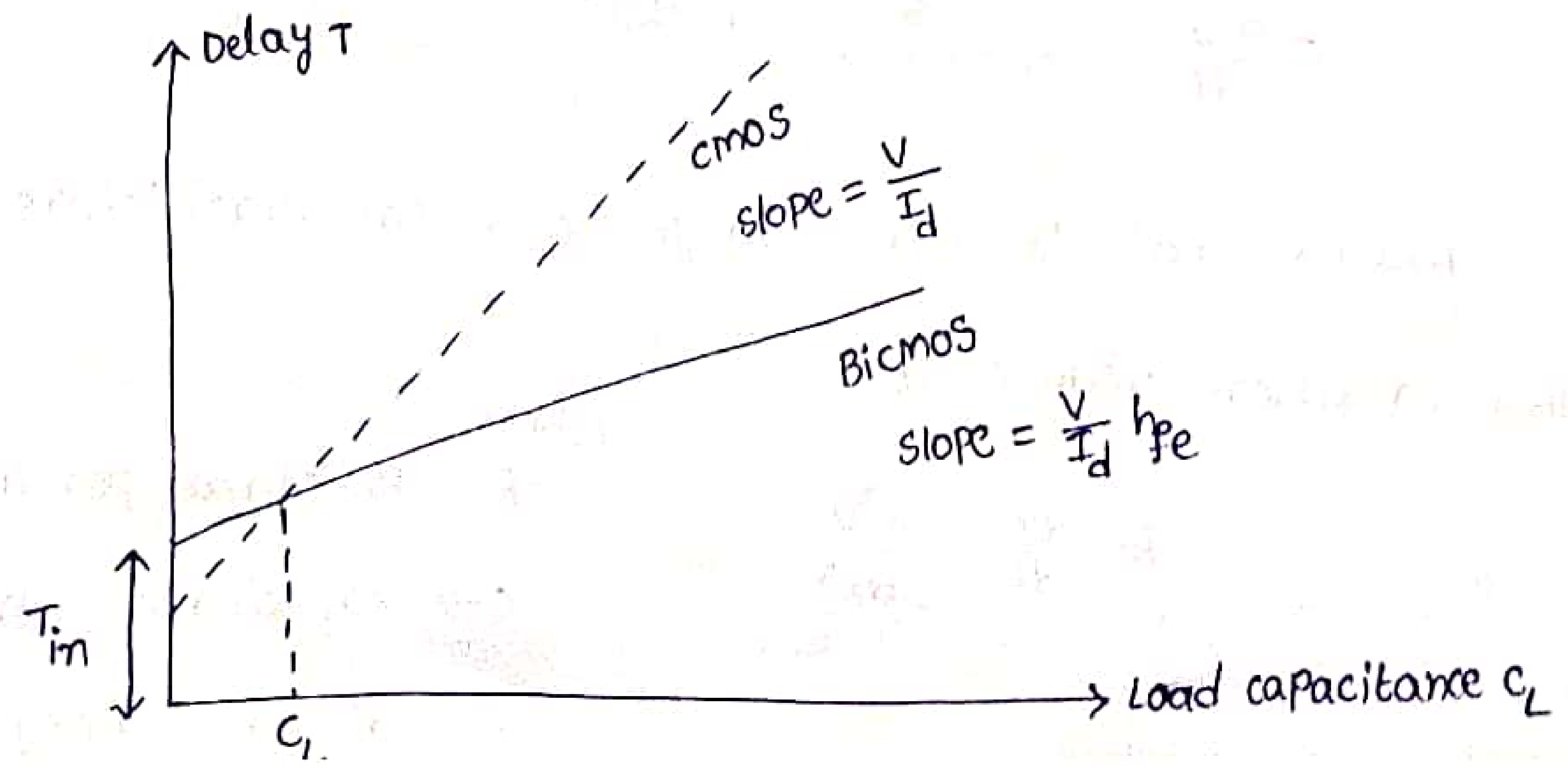
g_m → Transconductance

The value of Δt is small because transconductance (g_m) is higher.

The delay due to bipolar transistor reveals that it has 2 components T_{in} and T_L

i) T_{in} :- To charge the base emitter Junction of bipolar transistor. The time is typically 2ns for Bicmos transistor- based driver. cmos driver require 1ns for T_{in} to charge Input gate capacitance. In case of GaAS driver is almost 50-100 p.s.

ii) T_L :- Time required to charge output load capacitance C_L and equals $(V/I_d)(1/h_{fe}) C_L$. This is less for bipolar driver by a factor h_{fe} as compared to mos drivers.



Propagation delays :-

1. cascaded Pass transistors :-

The pass transistors used as parallel (or) series combination of switches in logic arrays.

consider a chain of 4 pass transistors connected in series, the gate of each transistor connected to V_{DD} (Logic1).

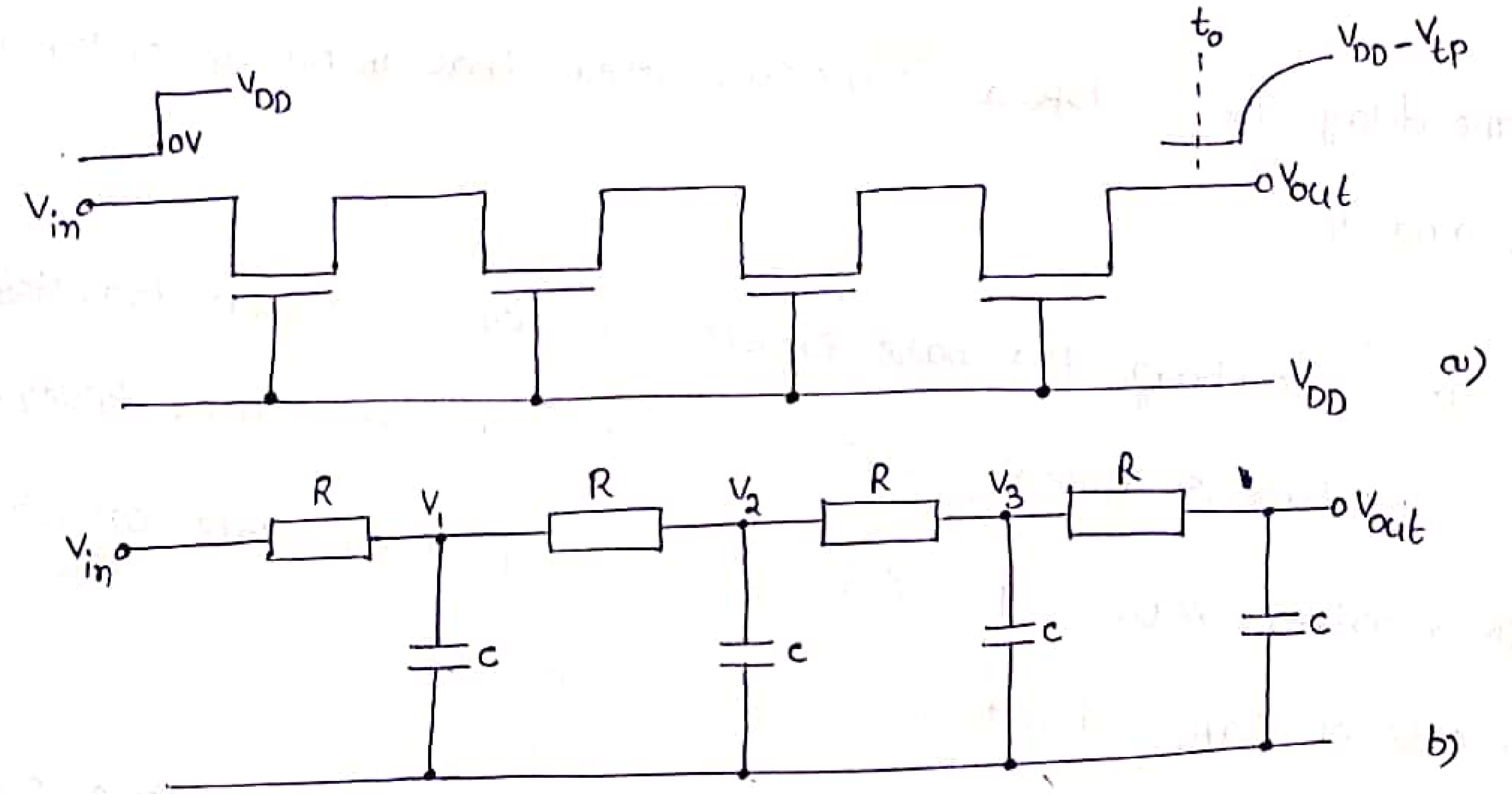


Fig :- Propagation delays in Pass transistor chain

Apply KCL at node V_2,

$$C\frac{dV_2}{dt} = I_1 - I_2 = \frac{(V_1-V_2)-(V_2-V_3)}{R}$$

Assume that there are large no.of pass transistors in series.

Then equation reduces to

$$RC\frac{dv}{dt} = \frac{d^2v}{dx^2}$$

where,

R → Resistance per unit length

C → capacitance per unit length

x → distance along the Network from Input

The propagation time τ_p for a signal to propagate a distance x is $\tau_p \propto x^2$

Let define variables r and c such that $R = rR_s$ and $c = c\Box c_g$ are lumped Network elements R and c. Then total network element is

$$R_{total} = nrR_s$$

$$C_{total} = nc\Box c_g$$

Actually 'r' is relative resistance per section in terms of R_s and c.

Total time delay τ_d for 'n' section is

$$\boxed{\tau_d = n^2 cr(\tau)}$$

If 'n' Increases, total delay Increases and in Practice no more than 4pass transistors connected in series. If the number can exceeded a buffer is Inserted between each group of 4 Pass transistors.

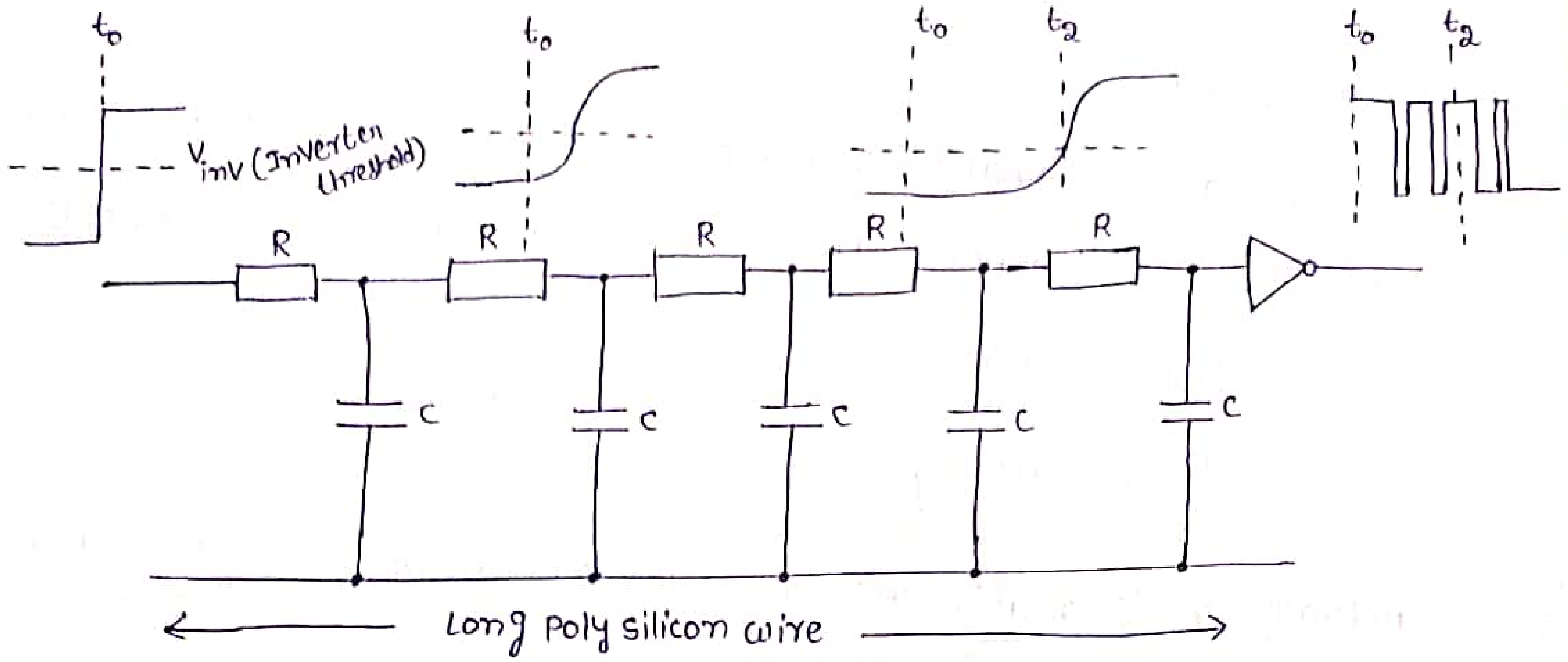
2. Design of Long Polysilicon wires :-

For long polysilicon wires, use buffers mainly

i) signal propagation is speeded up ii) Reduction in sensitivity to noise

Introduction of delays in signal propagation makes the signal more susceptible to noise as shown in fig. below

In diagram, the slow rise time of a signal at the Input of Inverter means that the Input Voltage spends a long time in the vicinity of V_{inv} so that small disturbances due to noise will switch the Inverter state between '0' and '1' as shown in output.



Hence it is necessary that long polysilicon wires by suitable buffers to avoid the effects of noise and to speed up the rise time of propagated signal edges.

wiring capacitances :-

The various sources of capacitance that contribute overall wiring capacitance are

1. Fringing fields
2. Inter layer capacitances
3. peripheral capacitances

1. Fringing fields :-

It can be major component of the overall capacitance for Interconnect wires.

For fine line metallization, the value of fringing field capacitance (C_{ff}) can be same order as the area capacitance.

Thus C_{ff} should be taken into account if accurate prediction is needed

$$C_{ff} = \varepsilon_{SiO_2}\, \varepsilon_0\, l \left[\frac{\pi}{\ln\left\{1 + \frac{2d}{t}\left(1 + \sqrt{1 + \frac{t}{d}}\right)\right\}} - \frac{t}{4d} \right]$$

where l → wire length

t → thickness of wire

d → wire to substrate separation

Total capacitance, $\boxed{C_w = C_{area} + C_{ff}}$

2. Inter layer capacitance :-

There is a chance to exists a capacitance between the layers due to parallel plate effects.

This capacitance will depends on the Layout, whether the layers cross (or) when one layer underlies another

For regular structures it is readily caluclated and contributes significantly to the accuracy of circuit modeling and delay caluclations.

3. peripheral capacitance :-

The source and drain n-diffusion regions forms Junctions with the p-substrate (or) p-well at well defined and uniform depths.

Similarly for P-diffusion regions ~~in~~ forms Junctions with the n-substrate (or) n-well at well defined and uniform depths.

For diffusion regions each diode thus formed has associated with it a peripheral capacitance of diffusion region to substrate; smaller the source (or) drain area greater the relative value of peripheral capacitance.

Total diffusion capacitance is given by

$$\boxed{C_{total} = C_{area} + C_{peri}}$$

Fan-IN and FAN-OUT :-

FAN-IN :- The no. of Inputs to a gate is called FAN-IN

FAN-OUT :- The no. of gates and length of metal tracks connected to its output

Choice of Layers :-

The following rules must be considered for the proper choice of Layers

i) Except for very small distances, Polysilicon should not be used for routing V_{DD} and V_{SS}.

ii) Long lengths should be used only after careful considerations because Polysilicon layer has relatively high value of R_S.

iii) V_{DD} and V_{SS} (GND) must be distributed only on metal layers. This is also due to R_S value

Scaling of MOS circuits

Scaling models and Scaling factors :-

Scaling means reduce the dimensions (size) of the MOS transistors.

There are 3 Scaling models

1. constant field scaling
2. constant voltage scaling
3. Lateral scaling

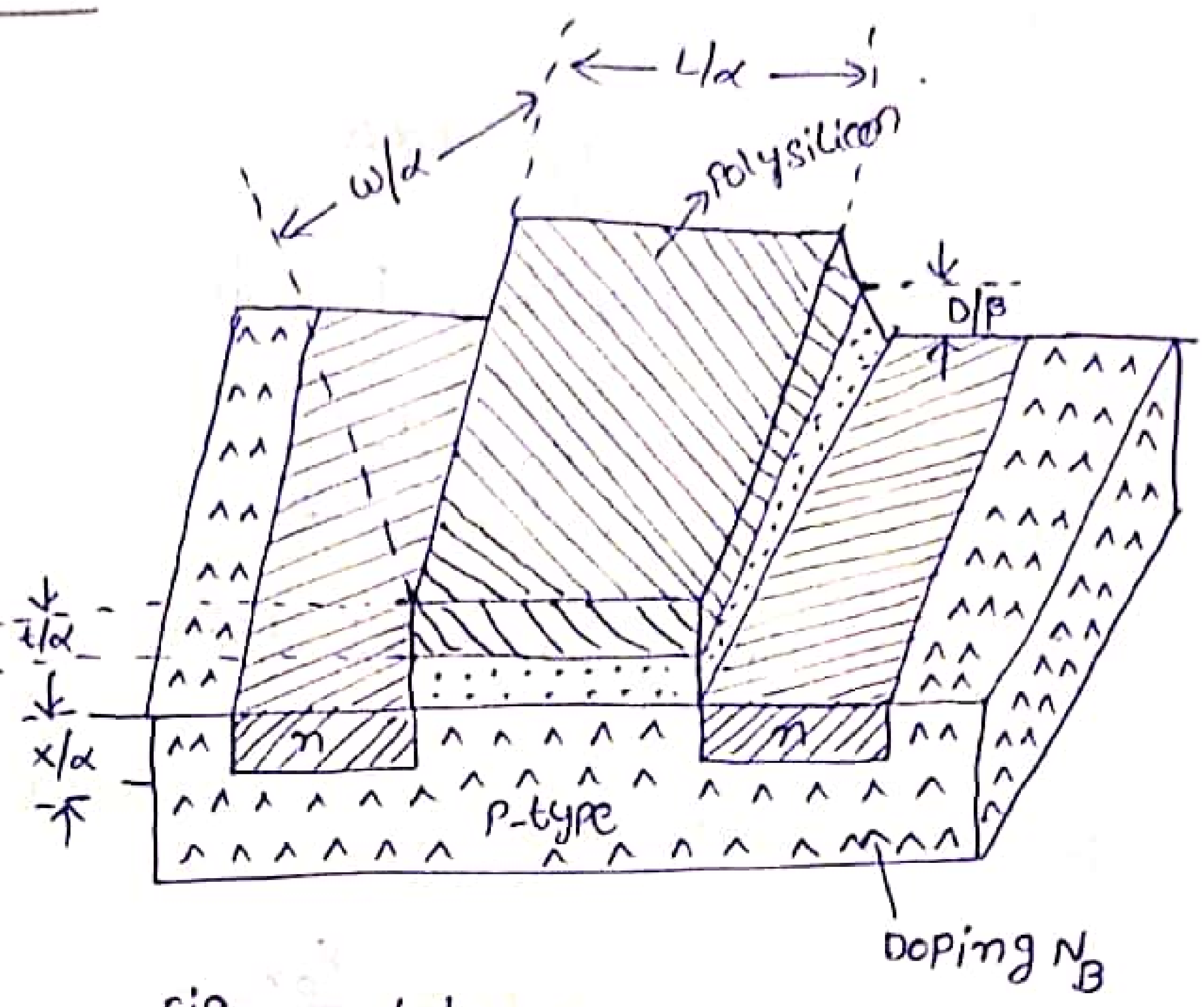


Fig:- Scaled nmos transistor

1. constant field scaling:-

All the Parameters in MOSFET are Scaled by the factor α except supply voltage V_{DD} and gate oxide thickness t_{ox}.

2. constant voltage scaling:-

The supply voltage V_{DD} and gate oxide thickness t_{ox} are scaled down by β.

3. Lateral scaling:-

This is combined model of constant field and constant voltage scaling.

Scaling factors for device parameters :-

1. Gate area (A_g) :-

W → channel width
L → channel length

$$A_g = L \times W$$

$$L \to 1/\alpha \, , \quad \beta \to 1/\alpha$$

$$\boxed{A_g = \frac{1}{\alpha^2}}$$

A_g is scaled by $1/\alpha^2$

2. Gate capacitance per unit area C_o (or) C_{ox} :-

$$C_o = \frac{\varepsilon_{ox}}{D}$$

where ε_{ox} → Permitivity of gate oxide ~~&~~

D → gate oxide thickness

$$C_o \text{ is scaled by } \frac{1}{1/\beta} = \beta$$

3. Gate capacitance (C_g) :-

$$C_g = C_o \cdot WL$$

$$C_o \to \beta \, , \quad WL \to \frac{1}{\alpha^2}$$

$$C_g = \frac{\beta}{\alpha^2}$$

C_g is scaled by $\frac{\beta}{\alpha^2}$

4. Parasitic capacitance (C_x) :-

$$C_x \propto \frac{A_x}{d}$$

$A_x \to$ Area of depletion region

d → depletion width

A_x is scaled by '$1/\alpha^2$'

d is scaled by '$1/\alpha$'

$$C_x = \frac{1}{\alpha^2} / \frac{1}{\alpha} = \frac{\cancel{\alpha}}{\alpha^{\cancel{2}}} = \frac{1}{\alpha}$$

5. carrier density in channel (Q_{on}) :-

$$Q_{on} = C_0 \cdot V_{gs}$$

Q_{on} is Average charge per unit area in the channel in ON state.

C_0 scaled by 'β'

V_{gs} scaled by $\frac{1}{\beta}$

∴ scaling factor (Q_{on}) = $\beta \cdot \frac{1}{\beta} = 1$

6. channel resistance (R_{on}):-

$$R_{on} = \frac{L}{\omega} \cdot \frac{1}{Q_{on}\mu}$$

$\mu \to$ carrier mobility in channel

∴ scaling factor (R_{on}) $= \frac{1/\alpha}{1/\alpha} \cdot \frac{1}{1}$

$= 1$

7. Gate delay (T_d) :-

$$T_d \propto C_g \cdot R_{on}$$

$$\therefore \text{scaling factor } (T_d) = \frac{\beta}{\alpha^2} \cdot 1$$

$$= \frac{\beta}{\alpha^2}$$

8. maximum operating frequency (f_0) :-

$$f_0 = \frac{W}{L} \cdot \frac{\mu C_0 V_{DD}}{C_g}$$

$$\therefore \text{scaling factor } (f_0) = \frac{1}{(\beta/\alpha^2)} = \frac{\alpha^2}{\beta}$$

9. saturation current (I_{dss}) :-

$$I_{dss} = \frac{C_0 \mu}{2} \frac{W}{L} (V_{gs} - V_t)^2$$

C_0 scaled by 'β'

V_{gs} & V_t scaled by $1/\beta$

$$\therefore \text{scaling factor } (I_{dss}) = \beta \cdot \left(\frac{1}{\beta}\right)^2 = \frac{1}{\beta}$$

10. current density (J) :-

$$J = \frac{I_{dss}}{A}$$

A → cross sectional area which is scaled by $1/\alpha^2$

$$\therefore \text{scaling factor } (J) = \frac{1/\beta}{1/\alpha^2} = \frac{\alpha^2}{\beta}$$

11: Power dissipation per gate (P_g) :-

Power dissipation per gate consists of 2types i.e static and dynamic.

static power dissipation takes place when device holds a particular state

Dynamic power dissipation takes place when device changes its state.

$$P_g\ (\text{static}) = \frac{(V_{DD})^2}{R_{on}}$$

$$P_g\ (\text{dynamic}) = E_g \cdot f_0$$

P_g (static) scaled by $\dfrac{(1/\beta^2)}{1} = \dfrac{1}{\beta^2}$ $\left[\because V_{DD} \rightarrow \frac{1}{\beta} \quad R_{on} \rightarrow 1\right]$

P_g (dynamic) scaled by $\dfrac{1}{\alpha^2\beta} \cdot \dfrac{\alpha^2}{\beta} = \dfrac{1}{\beta^2}$ $\left\{\because E_g \rightarrow 1/\alpha^2\beta \quad f_0 \rightarrow \alpha^2/\beta\right]$

$\therefore$ scaling factor (P_g) $= 1/\beta^2$

12. switching energy per gate (E_g) :-

$$E_g = (V_{DD})^2 \cdot \frac{C_g}{2}$$

$\therefore$ scaling factor for $E_g = \left(\frac{1}{\beta}\right)^2 \cdot \left(\frac{\beta}{\alpha^2}\right)$

$$= \frac{1}{\alpha^2\beta}$$

13. Power dissipation per unit area (P_a):-

$$P_a = \frac{P_g}{A_g}$$

$\therefore$ saling factor for P_a $= \dfrac{\left(\frac{1}{\beta^2}\right)}{\left(\frac{1}{\alpha^2}\right)}$

$= \dfrac{\alpha^2}{\beta^2}$

14. speed Pocven Product (P_T):-

$$P_T = P_g \cdot T_d$$

$\therefore$ scaling factors for $P_T = \frac{1}{\beta^2} \cdot \frac{\beta}{\alpha^2}$

$$P_T = \frac{1}{\alpha^2 \beta}$$

Limitations of scaling:-

i) substrate doping:-

substrate doping level has a direct relation with the built in Potential (Junction potential) V_B. consider V_B is small as compared with V_{DD}.

Scaling factors for substrate doping:-

scaling Involves reduce the channel length of mos transisto The depletion region widths must also be scaled down to prevent the source and drain depletion regions from meeting.

Depletion region width 'd' is

$$d = \sqrt{\frac{2\varepsilon_0 \varepsilon_{Si} V}{q N_B}} \quad \text{—①}$$

where

ε_{Si} → Relative Permitivity of Silicon = 12

ε_0 → Permitivity of free space = 8.85×10^{-14} F/cm

V → Effective Voltage $= V_a + V_B$

V_a → Applied voltage (maximum value = V_{DD})

V_B → Built-in potential (Junction potential)

q → charge of electron

N_B → doping level of substrate

and $V_B = \frac{KT}{q} \ln\left(\frac{N_B N_D}{n_i^2}\right)$ —②

N_D → donar concentration of source/drain

n_i → Intrinsic carrier concentration

$$d = \cancel{\sqrt{2\varepsilon_0 \varepsilon_{Si} V'}}$$

For the combined voltage and dimension scaling model applied to transistor,

$V_a = mV_B$ m → Real number,

$V = V_a + V_B$

$V = mV_B + V_B$ —③   ⇒ $V = (m+1)V_B$ —③

If the applied voltage V_a is scaled by $1/\beta$, we have

$$V_s = \frac{mV_B}{\beta} + V_B \quad \text{—④}$$

$$= \frac{mV_B + \beta V_B}{\beta} = \frac{(m+\beta)V_B}{\beta} \quad \text{—⑤}$$

From eq ③, $V_B = \frac{V}{m+1}$ — ⑥

substitute eq ⑥ in ⑤

$$V_S = \frac{(m+\beta)}{\beta} \cdot \frac{V}{m+1}$$

$$\Rightarrow \boxed{V_S = \frac{m+\beta}{\beta(m+1)}} \rightarrow \text{scaling factor}$$

V_S is the effective scaled voltage across depletion region.

N_B should scaled by $\frac{\alpha^2(\beta+m)}{m+1}$ so that scaling factor for d is $1/\alpha$.

Scaling factor for depletion width :-

from eq ①, as N_B Increased to reduce the depletion width and Increases the threshold voltage V_t.

maximum electric field Induced in the Junction is

$$E_{max} = \frac{2V}{d} \quad \text{— ⑥}$$

If N_B is Increased by a factor 'α' and if $V_a = 0$, then V_B Increased by $\ln\alpha$ and d is decreased by $\sqrt{\frac{\ln\alpha}{\alpha}}$

Therefore, the electric field 'E' across the depletion region is Increased by $\sqrt{\alpha/\ln\alpha}$ and will reach critical level E_{crit} with Increasing N_B.

The ~~maximum~~ fig (a) shows the depletion width 'd' as a function of substrate concentration N_B and supply voltage V_{DD}.

The dashed lines Indicates the max. depletion width for $E_{max} = E_{crit}$. —⑦

~~Sub. eq ⑦ in~~ From eq ①

$$d = \sqrt{\frac{2\varepsilon_0 \varepsilon_{Si}}{N_B q}(V)} \quad —⑧$$

sub. eq ⑥ in ⑧

$$d = \sqrt{\frac{2\varepsilon_0 \varepsilon_{Si}}{N_B q}\left(\frac{E_{max}\, d}{2}\right)} \quad —⑨$$

$$\left[\because E_{max} = \frac{2V}{d} \Rightarrow V = \frac{E_{max} d}{2}\right]$$

sub. eq ⑦ in ⑨

$$d = \sqrt{\frac{2\varepsilon_0 \varepsilon_{Si}}{N_B q}\left(\frac{E_{crit}\, d}{2}\right)} \quad —⑩$$

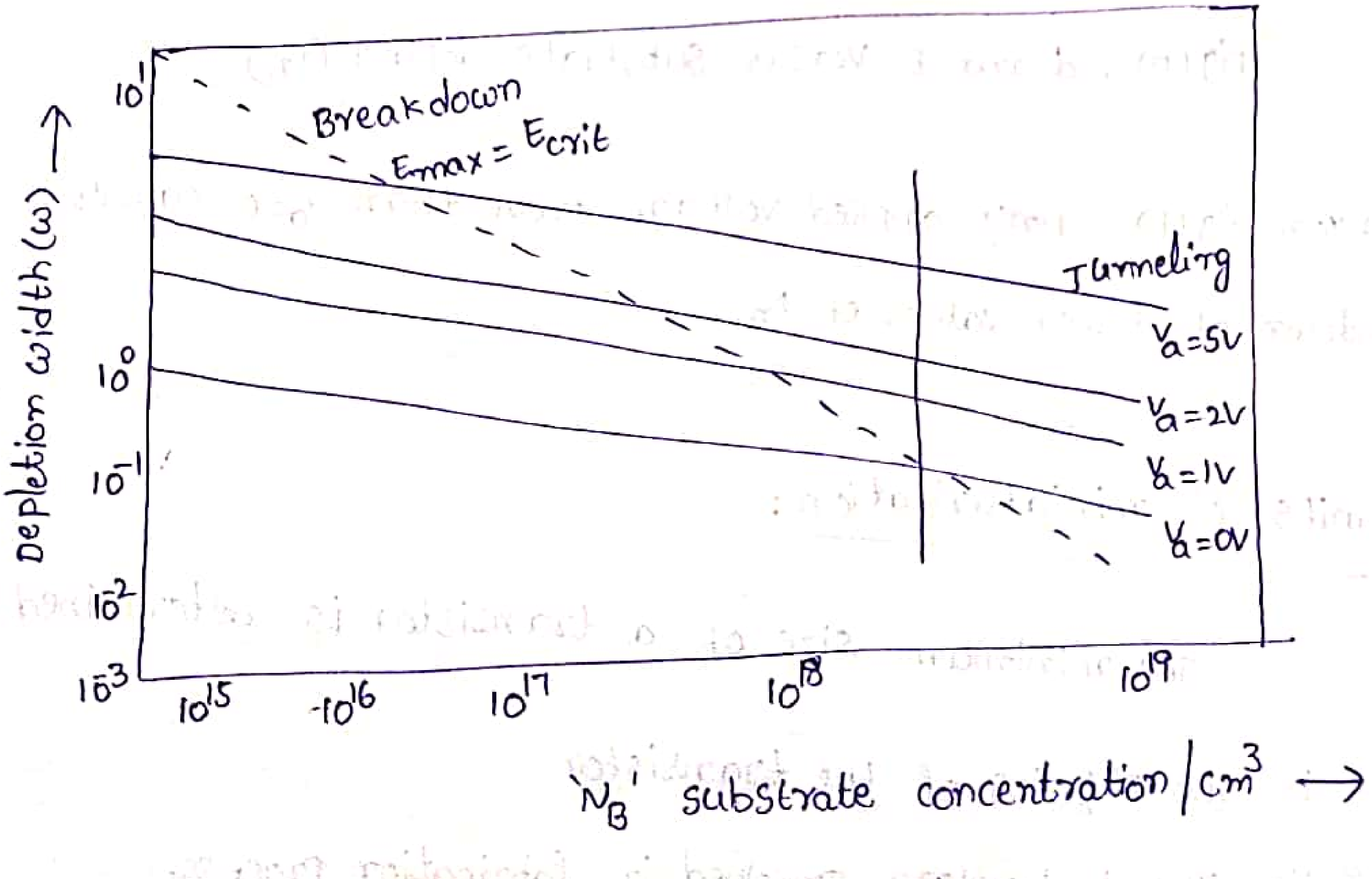


Fig (a) :- substrate concentration

In Fig (a), the region above dashed line is that where the Increased electric field 'E' will induce breakdown.

The maximum electric field E_{max} in depletion layer as a function of N_B as shown in fig (b).

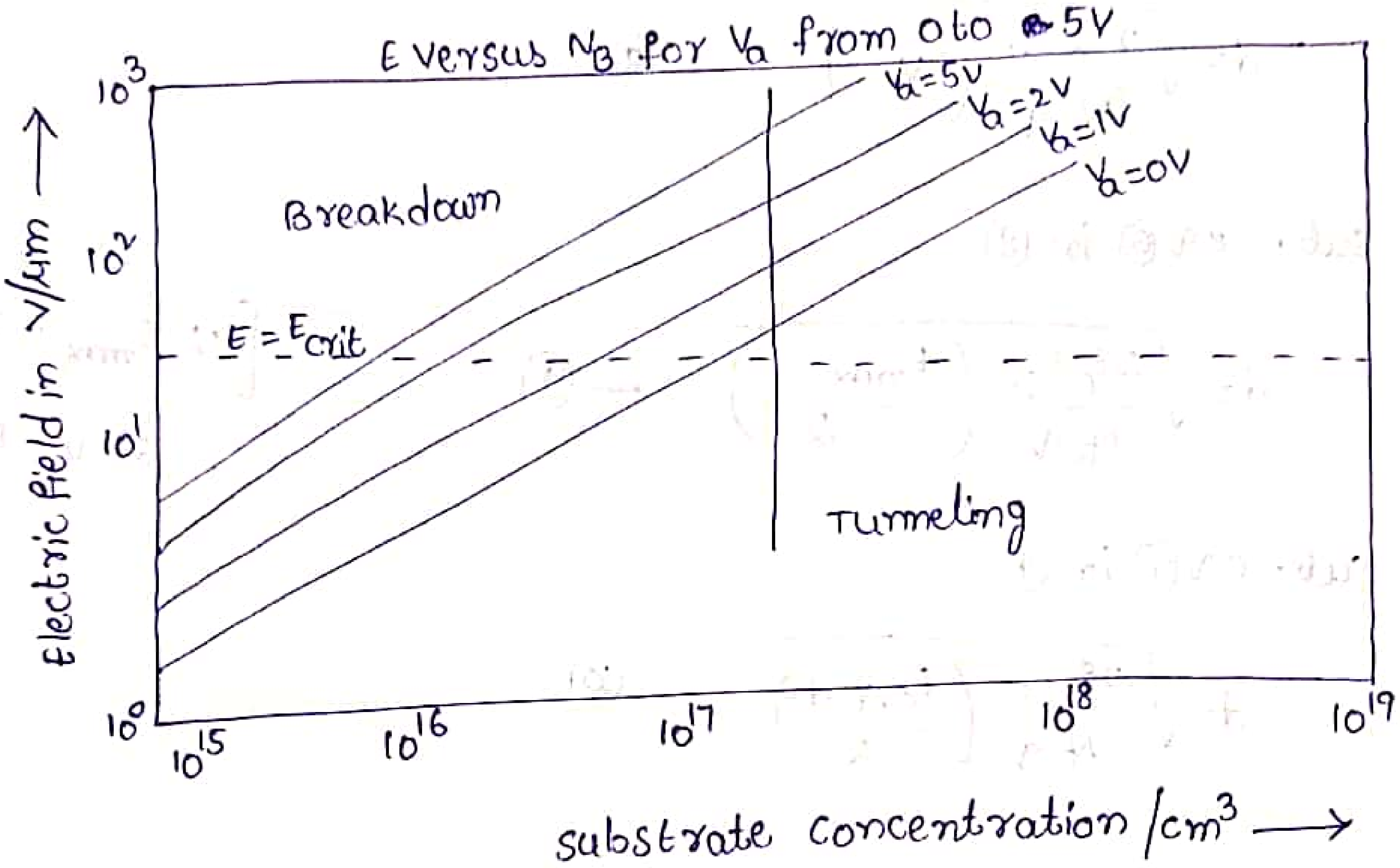


Fig (b) : d and E versus substrate doping (N_B)

From fig.(b), Any applied voltage more than $V_a=0$ causes breakdown at lower values of N_B.

ii) Limits of miniaturization :-

The minimum size of a transistor is determined by

i) The physics of the transistor

ii) The technology Involved in fabrication process.

The reduction of device geometry currently depends on alignment accuracy and resolution of Photolithographic technology the limit on feature size is at 0.3 μm which may be reduce the size using E-beam technology.

The size of transistor is defined in terms of channel length 'L'.

The time for an electron to travel from source to drain depends on channel length L.

$$V_{drift} = E\mu \quad \text{and} \quad L = 2d$$

$$\text{Transit time } \tau = \frac{L}{V_{drift}} = \frac{2d}{\mu E}$$

The maximum carrier drift velocity is approximately equal to V_{sat}.

The minimum transit time corresponds to minimum size transistor for $V_a = 0V$.

Transit time as a function of N_B and L is shown in fig 3(a),(b)

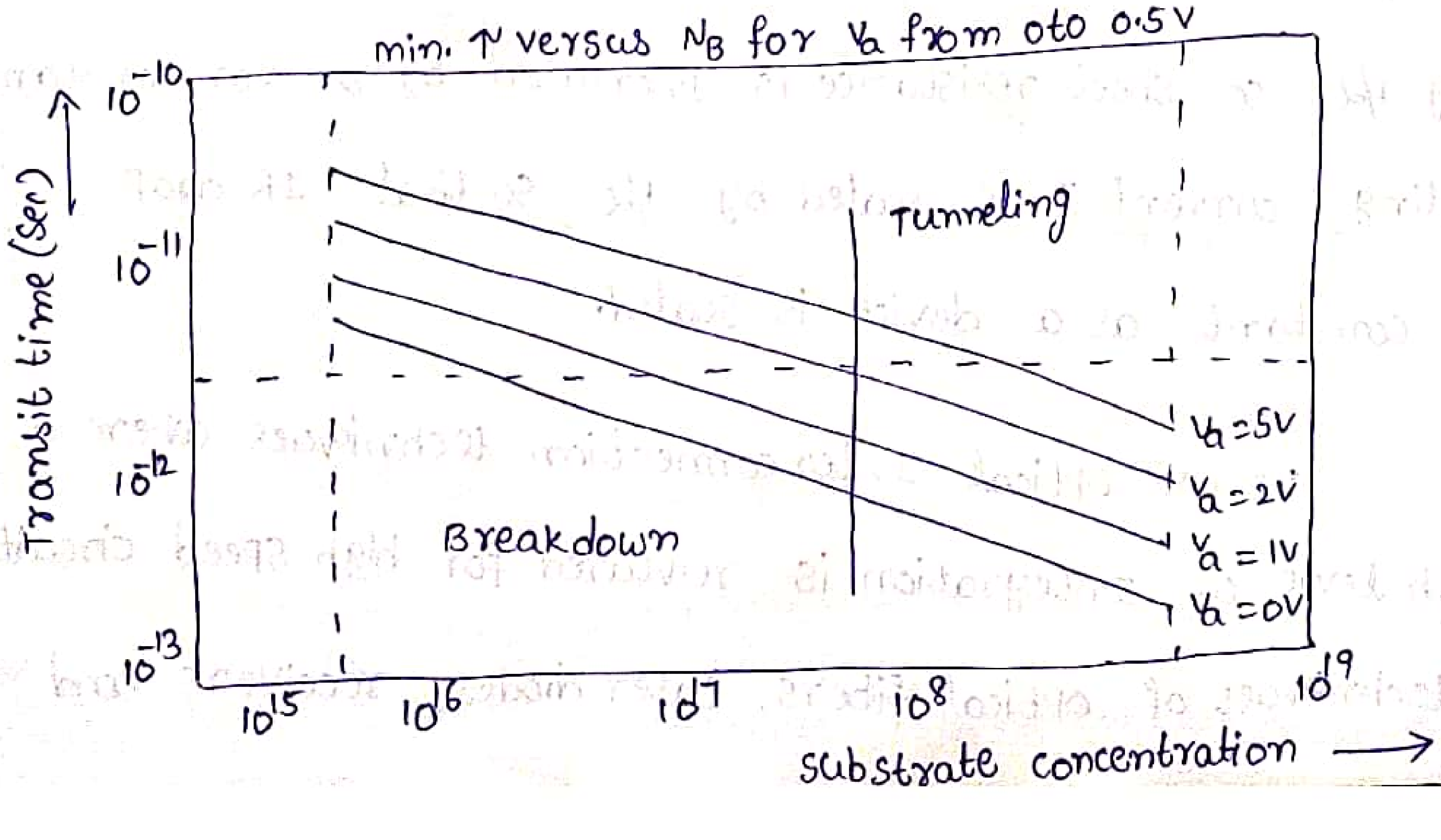


Fig: 3(a)

Fig 3(a) assumes transistor size L=2d with zero space between source and drain depletion regions.

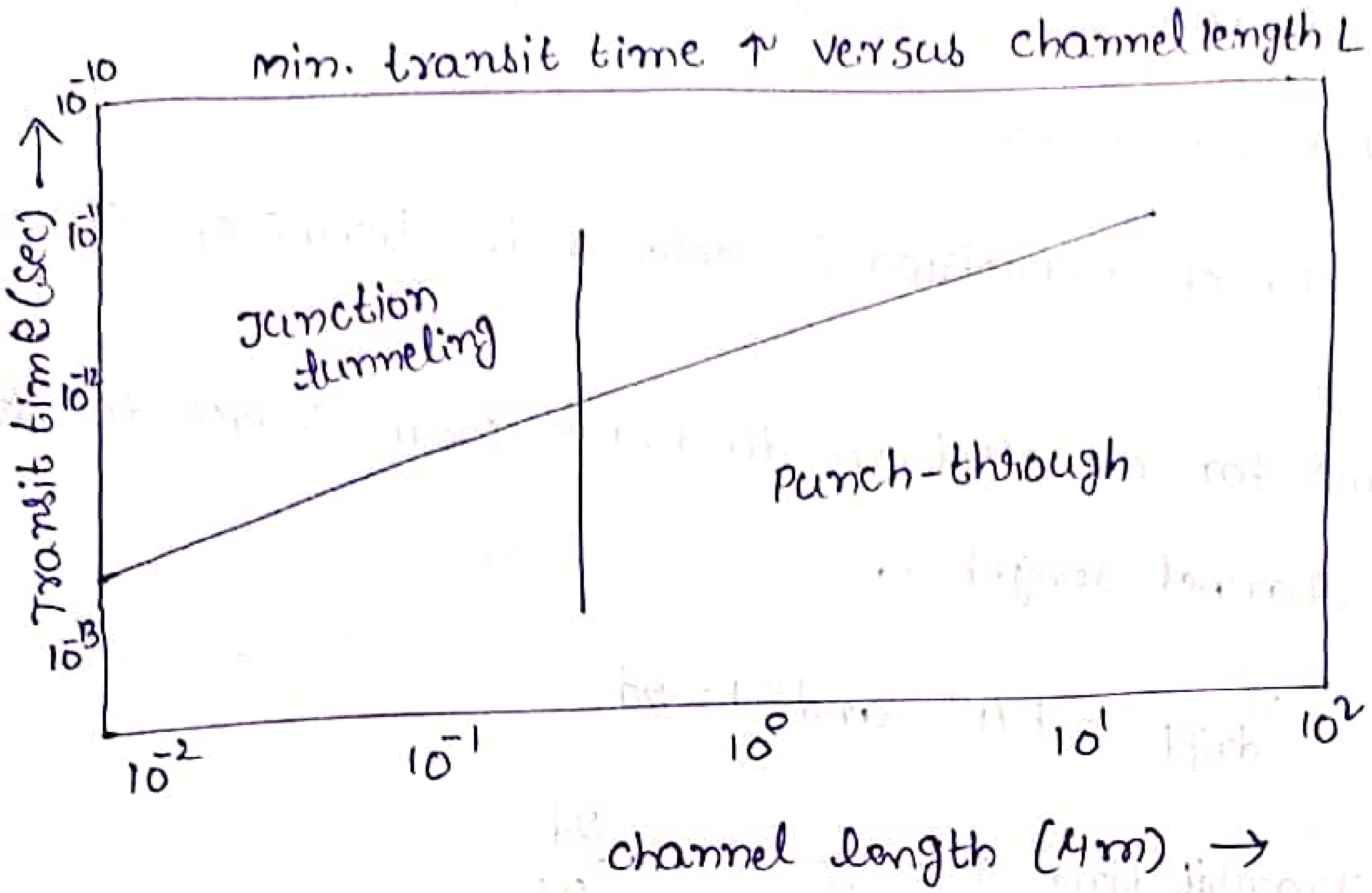


Fig 3(b) : Transit time τ versus L.

iii) Limits of Inter connect and contact resistance :-

The width, thickness and spacing of Inter connects are scaled by $1/\alpha$, cross-section areas must scaled by $1/\alpha^2$. For short distance Inter connections the conductor length is scaled by $1/\alpha$, so sheet resistance is Increased by α. For constant field scaling, current I is scaled by $1/\alpha$ so that IR drop remains constant as a device is scaled.

To use optical Inter connection techniques where a very high level of Integration is required for high speed circuits. To use techniques of optical fibers, Laser diodes, receivers and

Amplifiers must be Included in the Integration circuit.

performance will vary with matal materials used. But, rough estimations can be made for comparision with metal Interconnects.

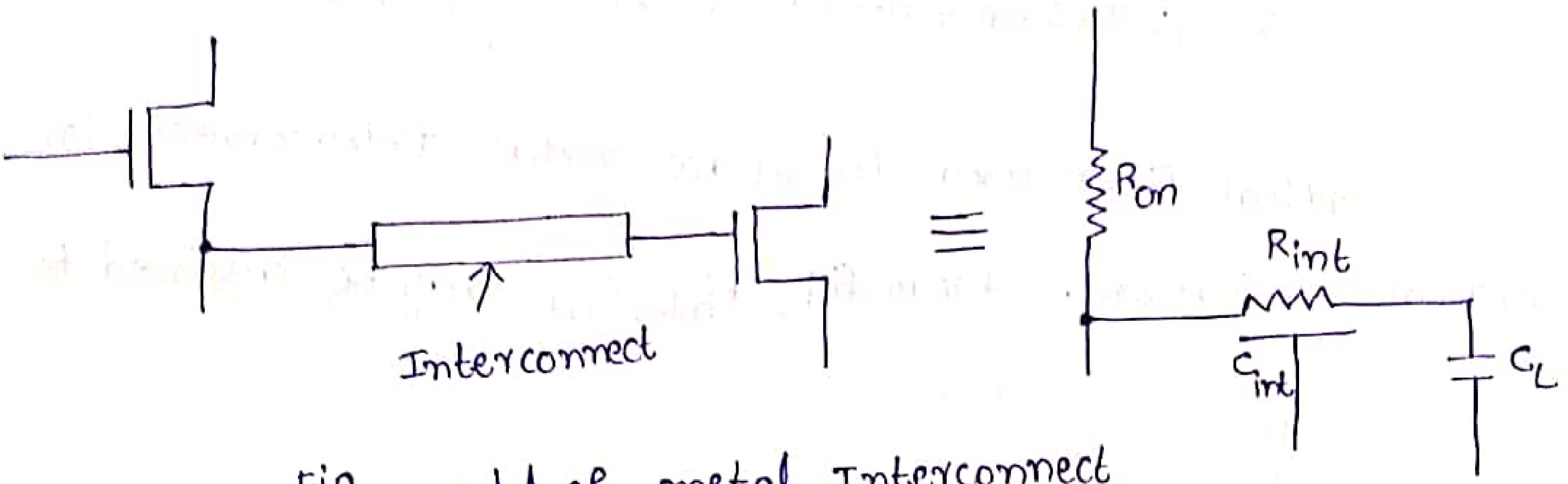


Fig:- model of metal Interconnect

The propagation delay T_p along a single aluminum Interconnect can be caluclated from

$$T_p = R_{int} C_{int} + 2.3[(R_{on} C_{int}) + (R_{on} C_L) + (R_{int} C_L)]$$

where $T_p = 2.3(R_{on} + R_{int}) C_{int}$

But $R_{int} = \frac{\rho L}{WH}$

and $C_{int} = \varepsilon_{ox} [1.15\ W/t_{ox} + 2.28 (H/t_{ox})^{0.222}] L$

where L, W and H are length, width, Height of Interconnect

$\rho \to$ Resitivity of Interconnect

$R_{on} \to$ Transistor ON resistance

$R_{int} \to$ Resistance of Interconnect

$C_{int} \to$ capacitance of Interconnect

$t_{ox} \to$ Thickness of dielectric oxide

$\varepsilon_{ox} = 3.45 \times 10^{-5}$ PF/μm → Permitivity of SiO_2

consider $\rho = 3\mu\Omega$-cm for Aluminium, $t_{ox} = 0.8\mu m$ for thick oxide

$L = 1cm$, $W = 3\mu m$, $H = 1\mu m$ then T_p is given by

$$T_p = (2.3 \times 5k\Omega + 0.1k\Omega)\, 2.5 \times 10^{-12} = 29\,nsec$$

optical fibers used to replace metal Interconnects in critical applications. From fig, R_{int}, C_{int} may be assumed to zero.

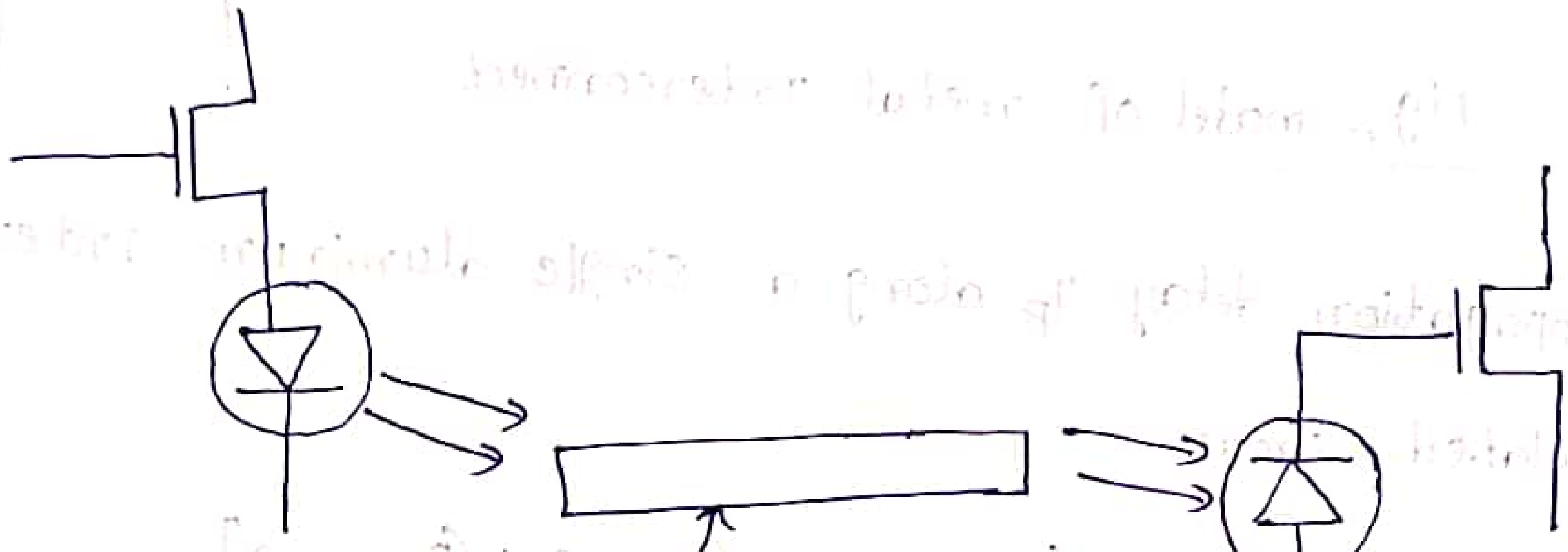


Fig:- Electro-optical Interconnection

$$T_p = 2.3\, R_{on} C_L + t_{laser} + t_{int} + t_{rec}$$

where, C_L → Input capacitance of Laser diode

t_{Laser} → Delay time through Laser diode

t_{int} → Propagation delay time along the optical fiber Interconnect

t_{rec} → Receiver delay time

$t_{int} = \frac{nL}{c}$ where L → Length of fiber

n → Refractive Index of optical fiber material

c → Speed of light

Laser diodes and receivers are high speed devices having self-delays arround 100 psec. The refractive Index is between 1.5 and 2. capacitance of discrete laser diode about 1PF.

The propagation delay using these values,

$$T_p = 2.3\times5\times10^3 \times 1\times10^{-12} + 1\times10^{-10} + \frac{2\times10^{-4}}{3\times10^8} + 1\times10^{-10} = 11.7 \text{ nsec}$$

The Propagation delay against varying length 'L' of Interconnects are shown in below fig (a) & propagation delay against width of Interconnect as shown in fig (b)

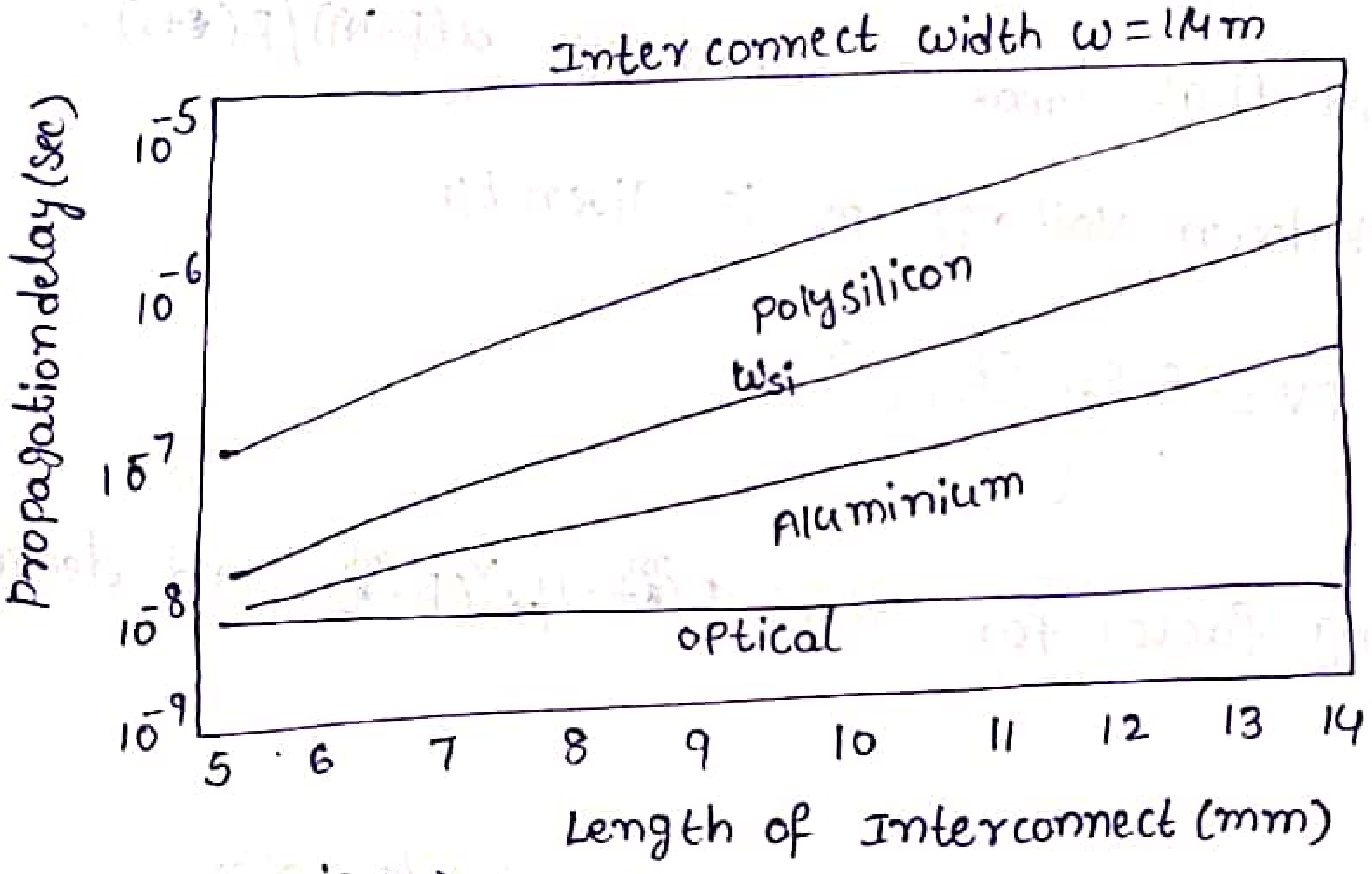


Fig (a).

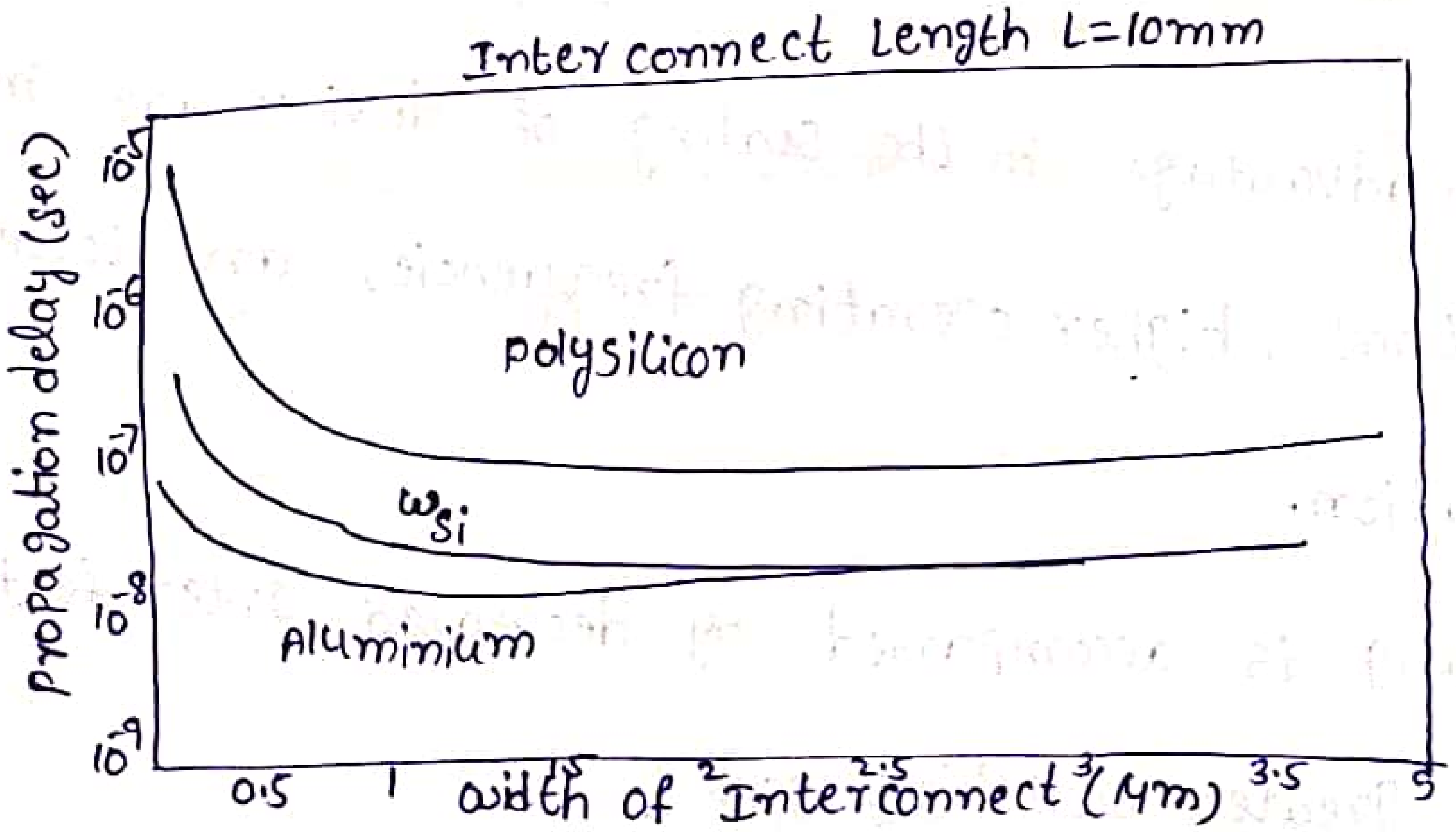


Fig (b)

Limits due to subthreshold currents :-

The subthreshold current $I_{sub} \propto e^{(V_{gs}-V_t) q/KT}$

The transistor is in OFF state, then the $V_{gs}-V_t$ is -ve and should be large as possible to minimize I_{sub}.

with the scaling voltages, $(V_{gs}-V_t)/KT$ reduces due to which subthreshold current Increases. To avoid this, both V_{gs} and V_t may be scaled along with V_{DD} by a larger factor. However, this causes electric field strength to Increase and there by lower break down voltages.

we derive that E_{max} is scaled by $\alpha(\beta+m)/\beta(m+1)$.

Junction break down voltage BV is given by

$$BV = \frac{\varepsilon_0 \varepsilon_{si} (E_{crit})^2}{2 N_B q}$$

Thus scaling factor for BV is $\beta(m+1)/\alpha^2(\beta+m)$ and decrease with scaling.

Limits on Logic levels and Supply voltage due to noise :-

main advantages in the scaling of devices are smaller gate delay time, higher operating frequencies and Lower Power dissipation.

scaling is accompanied by decreased Inter-feature spacing and greater switching speed.

The mean square current fluctuation in the channel is given by

$$i^2 = 4KTR_n \Delta f\, g_m$$

R_n → equivalent noise resistance at Input

Δf → Bandwidth

when transistor works in saturation, g_m does not have a linear relationship with gate voltage V_g.

$$g_m = V_p B$$

where $B = \frac{W\mu C_{ox}}{L}$

V_p is pinch-off voltage given by

$$V_p = V_g' - \frac{1}{2}\left(\frac{a}{C_{ox}}\right)^2 \left[\frac{(1+4V_g' C_{ox})^{1/2}}{a} - 1\right]$$

where, $V_g' = V_g - V_t + V_B$

$$a = (2\,\varepsilon_{Si}\, q N_B)^{1/2}$$

Equivalent resistance R_n is given by

$$R_n = \left(\frac{1V_g'}{2V_p'} + \frac{1}{6}\right) g_m{-1}$$

where $V_p' = V_p + V_B$

Since V_p is a monotonically decreasing function of gate oxide thickness t_{ox} and substrate doping N_B.

Thermal noise $R_n g_m$ is given by

$$R_n g_m = \frac{1}{2}\left(\frac{V_g - V_t + V_B}{V_p + V_B}\right) + \frac{1}{6}$$

Thermal noise $R_n g_m$ directly dependent on t_{ox} and N_B and some what on V_g. This is graphically shown in fig.

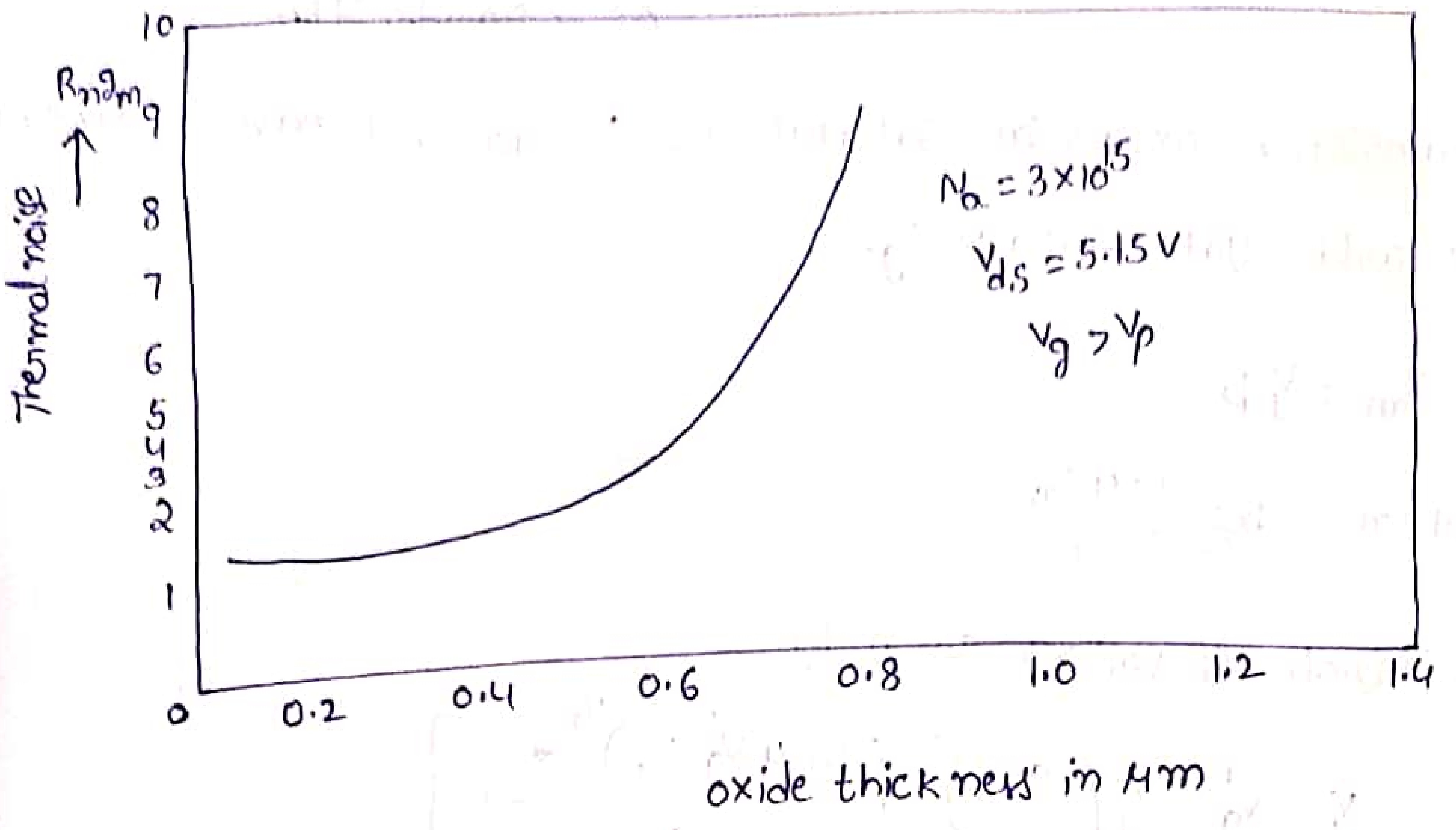


Fig:- a) Thermal noise versus oxide thickness

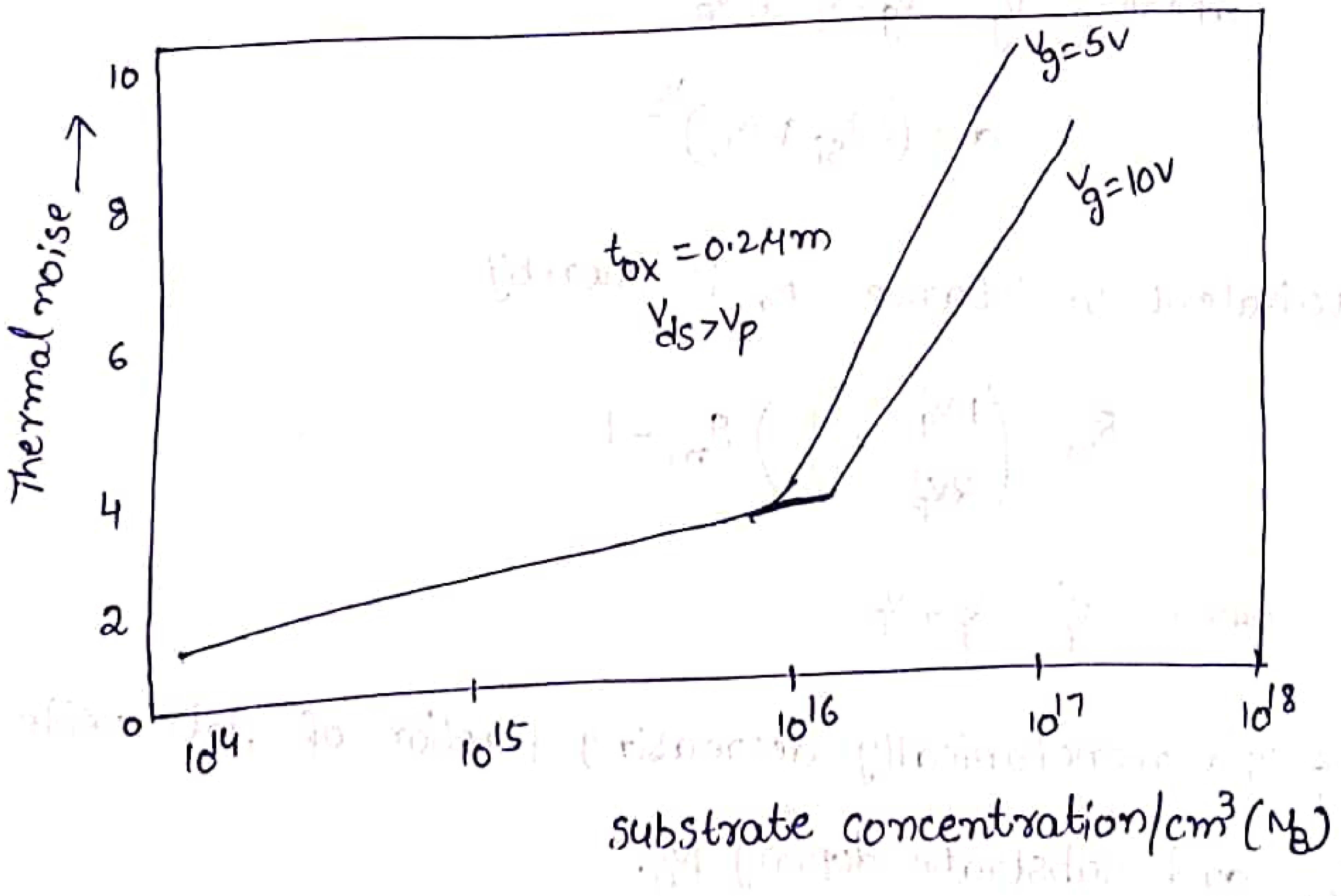


fig: b) substrate doping

In constant field scaling model, V_g is scaled by $1/\alpha$, N_B and C_{ox} are scaled by α. Hence the product of $R_n I_m$ is decreased to Increased value of C_{ox}. As a result, the ratio of Logic level to thermal noise undergoes degradation by almost same factor.

Another type of noise is flicker noise which is the result of fluctuations of carriers tapped in channel by surface states. current fluctuations Δi at the out put is given by

$$\Delta i^2 = \frac{q \mu S I V_d}{f L^2}$$

where $S = dn_t/dn$ is the surface state efficiency

$I =$ D.c drain current

$V_d =$ Applied drain voltage

'S' is a process dependent factor, the flicker noise has a scaling factor of one for constant field scaling (or) α^2/β^2 for combined scaling model.

Another noise sources considered those occuring due to mutual Inductive and mutual capacitive coupling and Lowest usual operating voltages. There can occur a cross talk between

2 parallel signal lines on a chip. The cross talk noise Increases as operating frevuency Increased and t_r, the rise time of coupled signal, is reduced.

The designer has to provide precautions against other noise sources due to external Influences, such as radio frequency signals, unterminated signal lines and lines with non-uniform Impedance characteristics, voltage drops on Power lines (or) ground connections etc..

one serious effect of scaling down is that it enhances the effect of Internally and externally generated noise which degrades both reliability and production of high density chip Layouts.

Limits due to current density :-

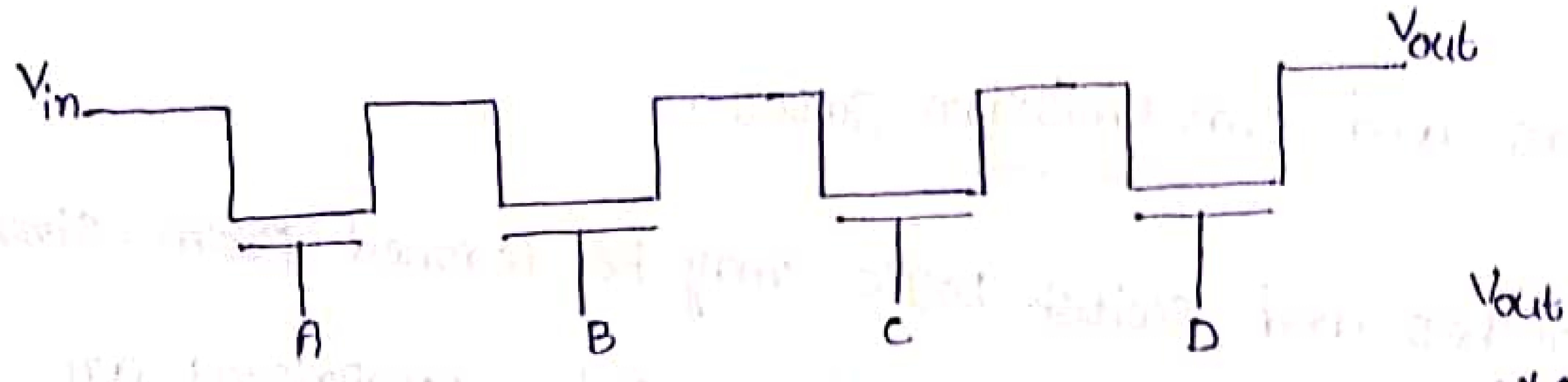
most widely used material for Interconnections in VLSI chips is high purity aluminium. Aluminium has high conductivity, scaling down of dimensions also Increases the current density in Interconnects by same factor for the constant field scaling. hence current density through Interconnects Increases.

The ~~d~~ current density of aluminium approaches 10^6 Amps/cm^2 the Inter connects are likely to be burned off owing to metal migration. current densities are set well below this limit and figures of J=1 to 2 mA/μm^2 are preferred.

Switch Logic :-

Switch Logic is based on the pass transistor (or) on transmission gates. This approach is fast for small arrays and takes no static current from the supply rails.

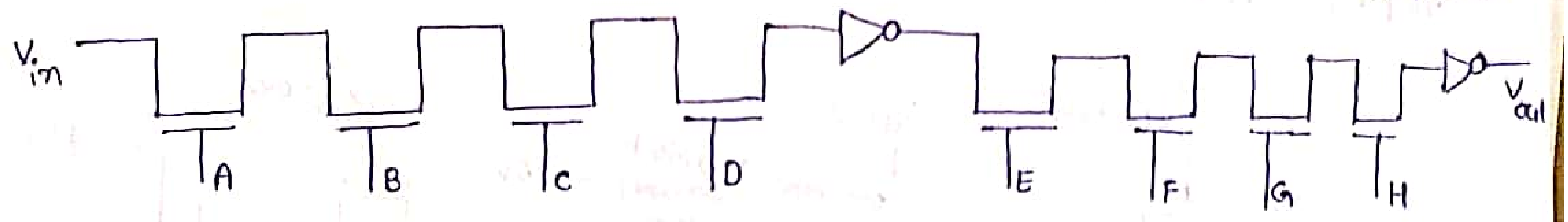
Basic And and OR connections are set out but many combinations of switches are possible.



$V_{out} = V_{in}$

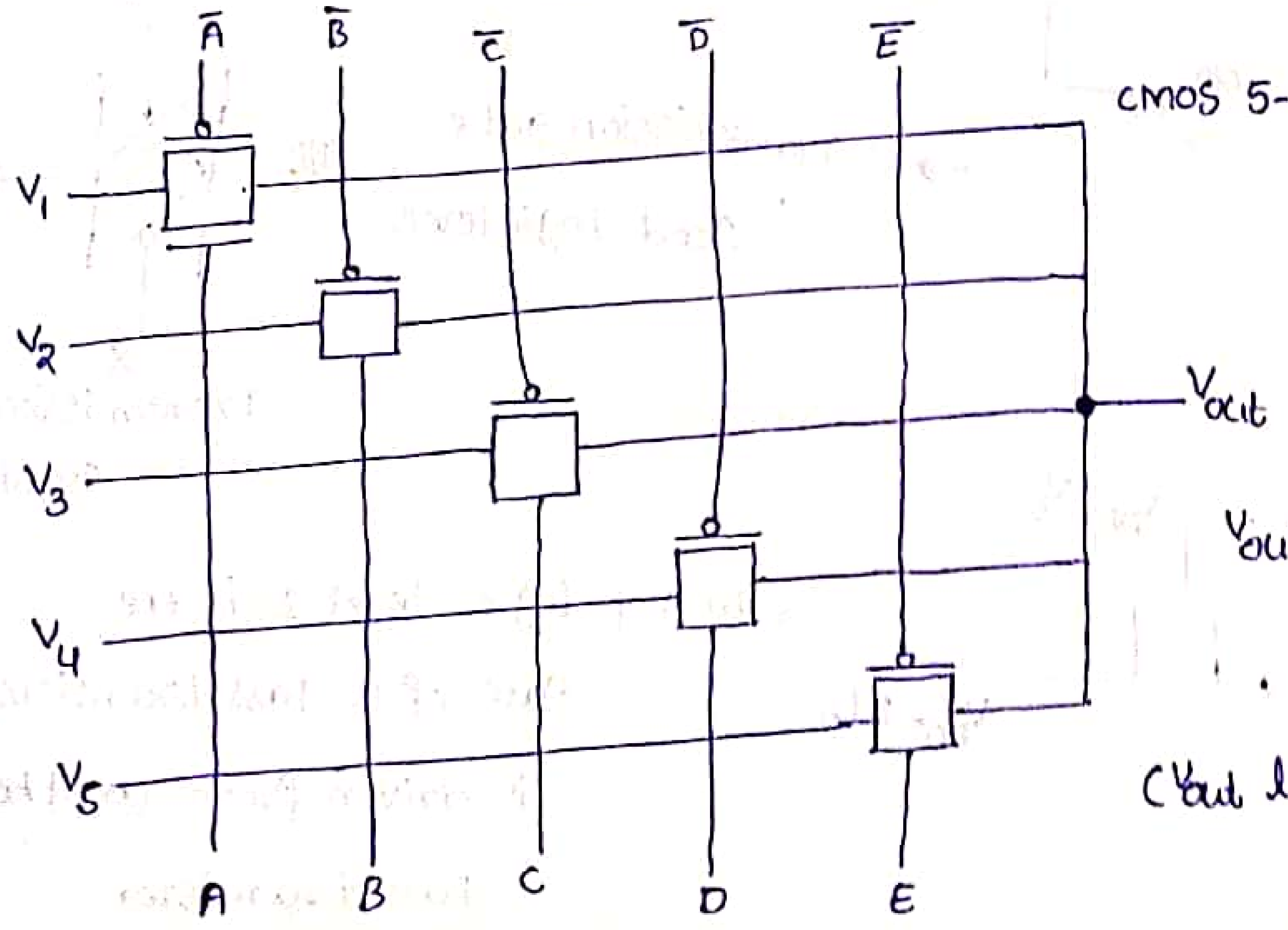
when $A \cdot B \cdot C \cdot D = 1$

(V_{out} Logic levels will be degraded by V_t effects)



$V_{out} = V_{in}$ when $A \cdot B \cdot C \cdot D \cdot E \cdot F \cdot G \cdot H = 1$

$V_{out} = ?$ when $A \cdot B \cdot C \cdot D \cdot E \cdot F \cdot G \cdot H \neq 1$

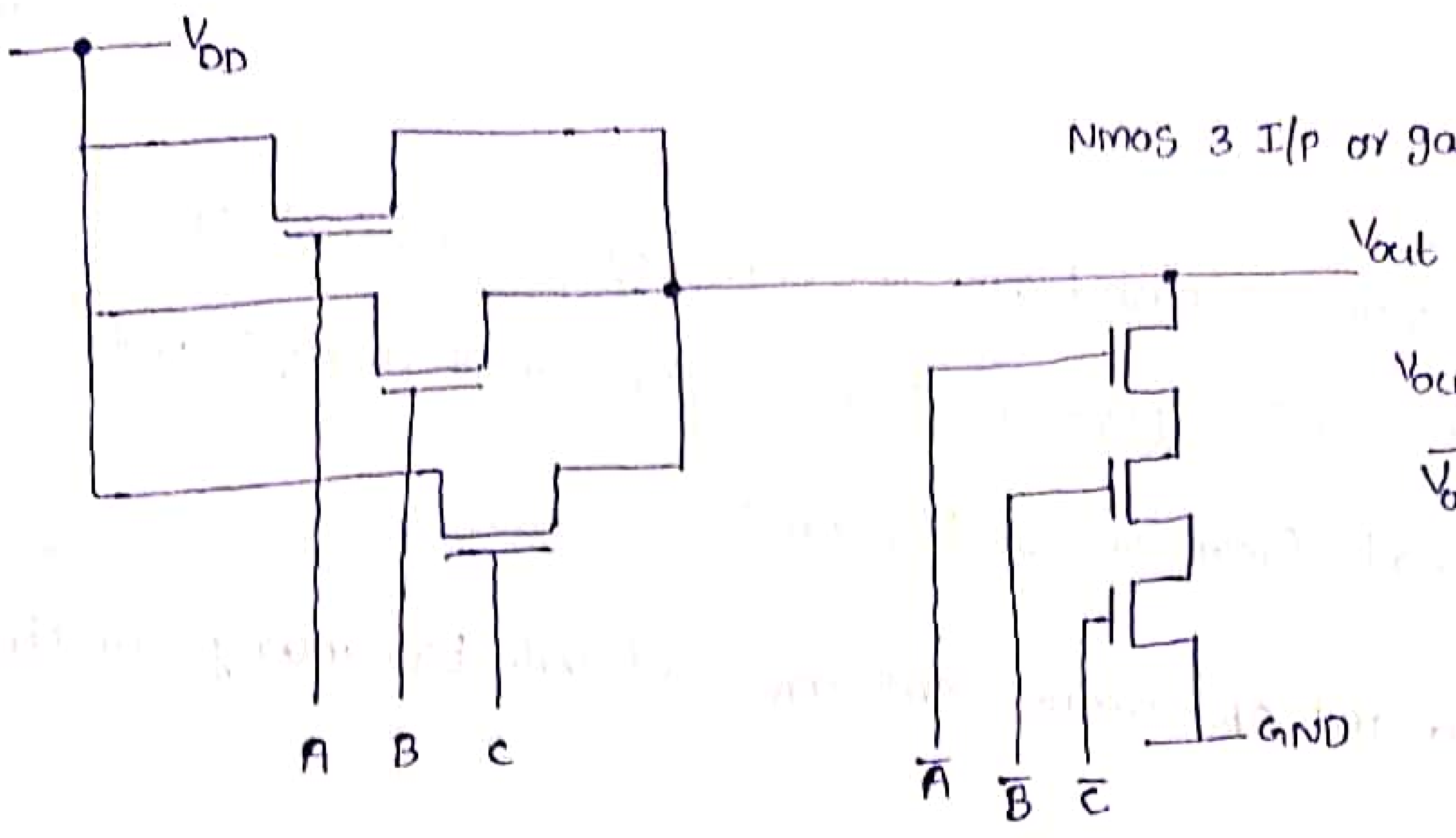


CMOS 5-way selector

$V_{out} = V_1 \cdot A + V_2 B + V_3 C + V_4 D + V_5 E$

assume A, B, C, D, E → mutually exclusive

(V_{out} logic levels will not be degraded by V_t effects)



Nmos 3 I/p or gate

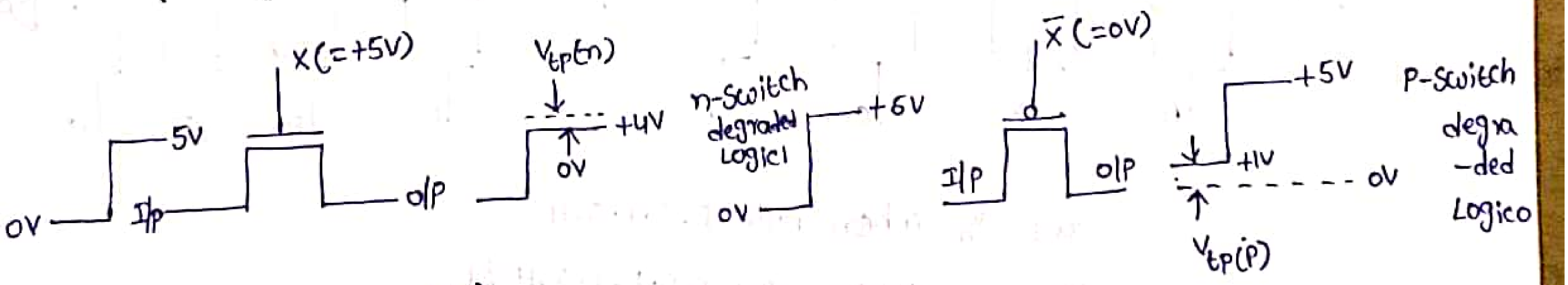
$V_{out} = A+B+C$ (degraded by V_t)

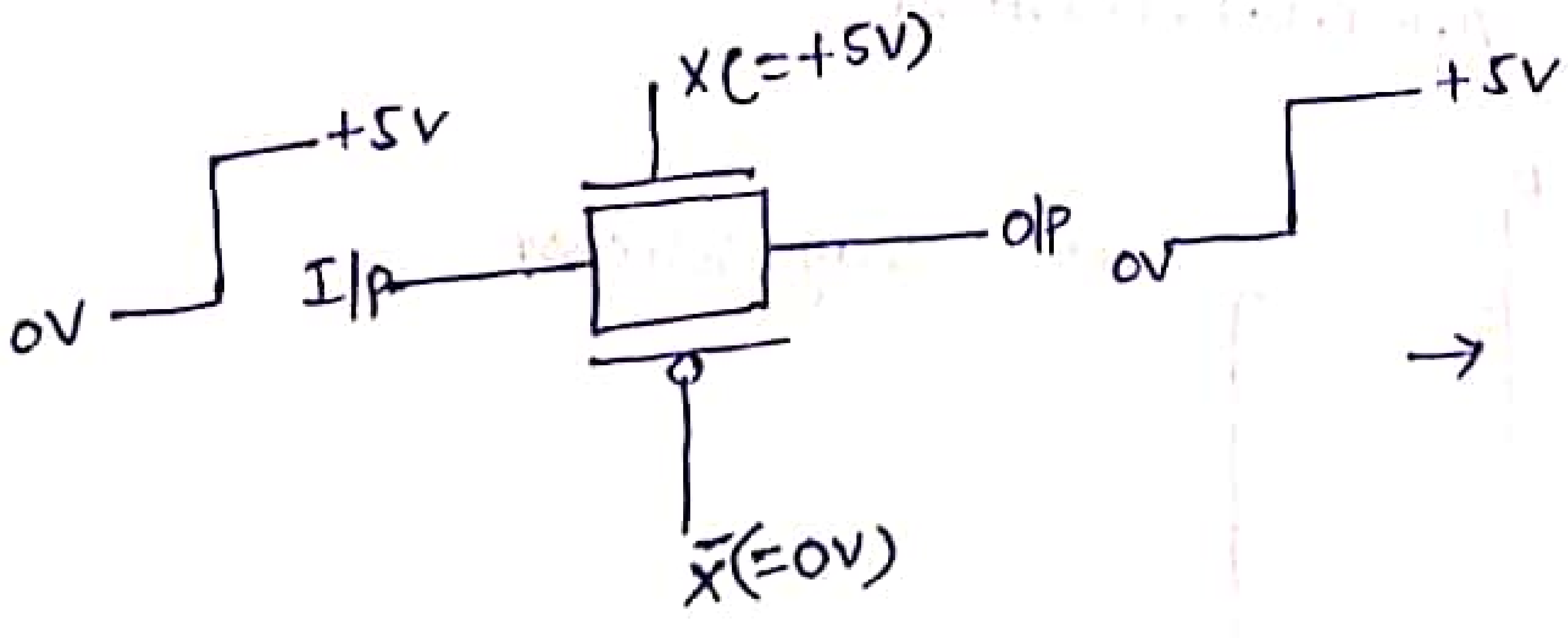
$\overline{V_{out}} = \bar{A}\cdot\bar{B}\cdot\bar{C}$

Note the arrangement to satisfy both logic '1' and logic '0' states

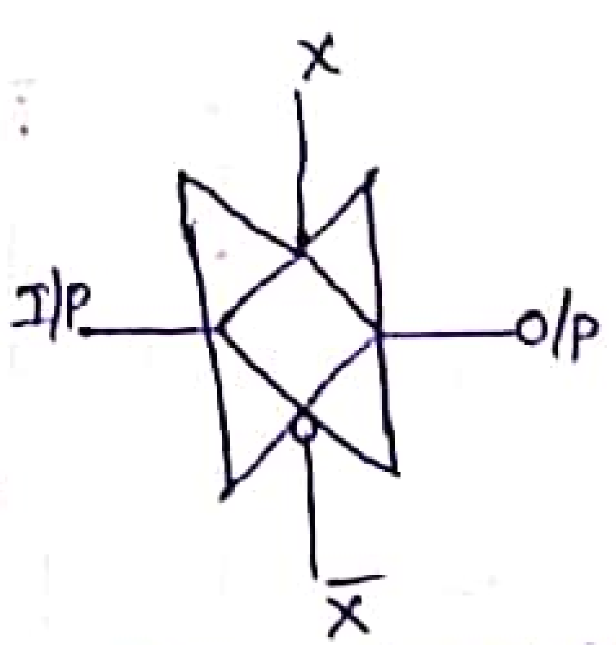
Pass transistors and Transmission gates :-

switches and switch Logic may be formed from simple n- (or) P- pass transistors (or) transmission gates. comprising an n-Pass and P-Pass transistor in parallel as shown in figure.

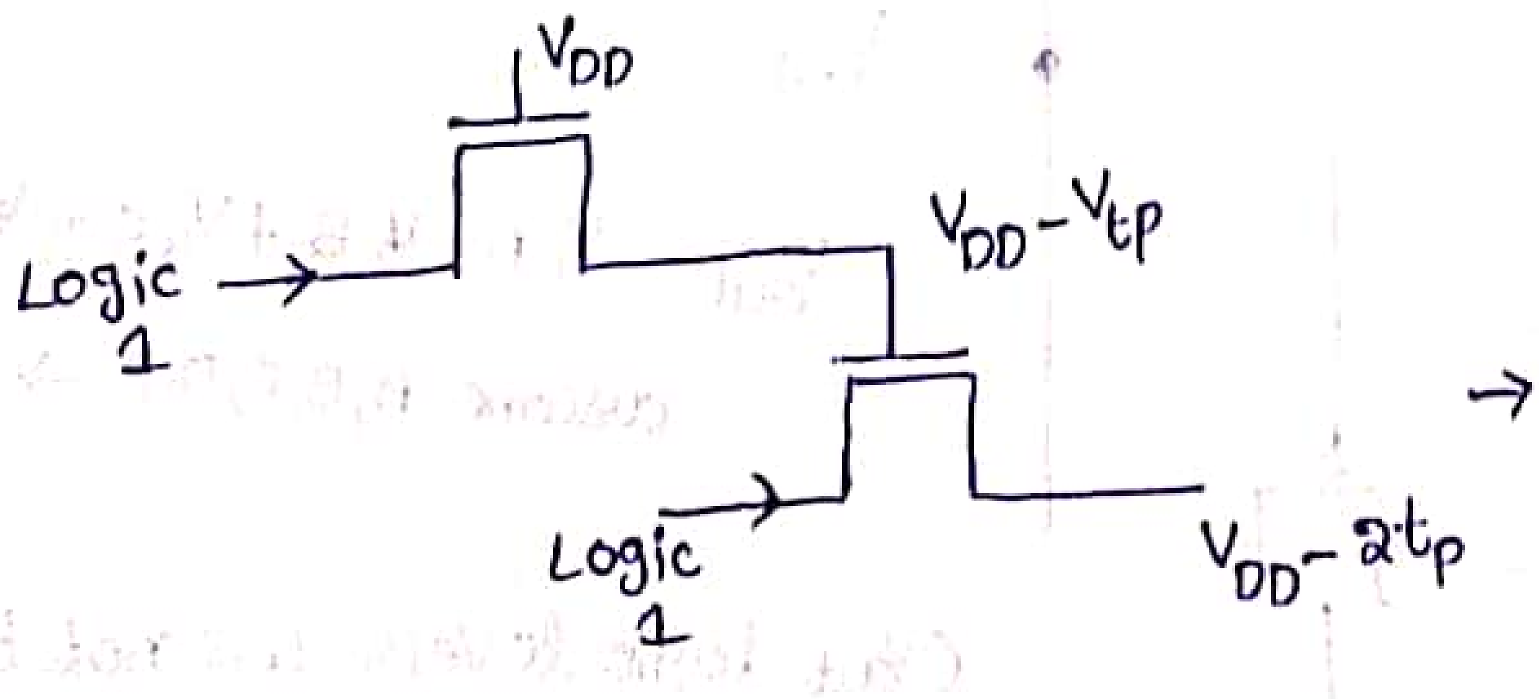




→ Transmission gate good Logic levels



Transmission gate symbol



→ Loss of logic level '1' if the gate of a pass transistor is driven from another pass transistor

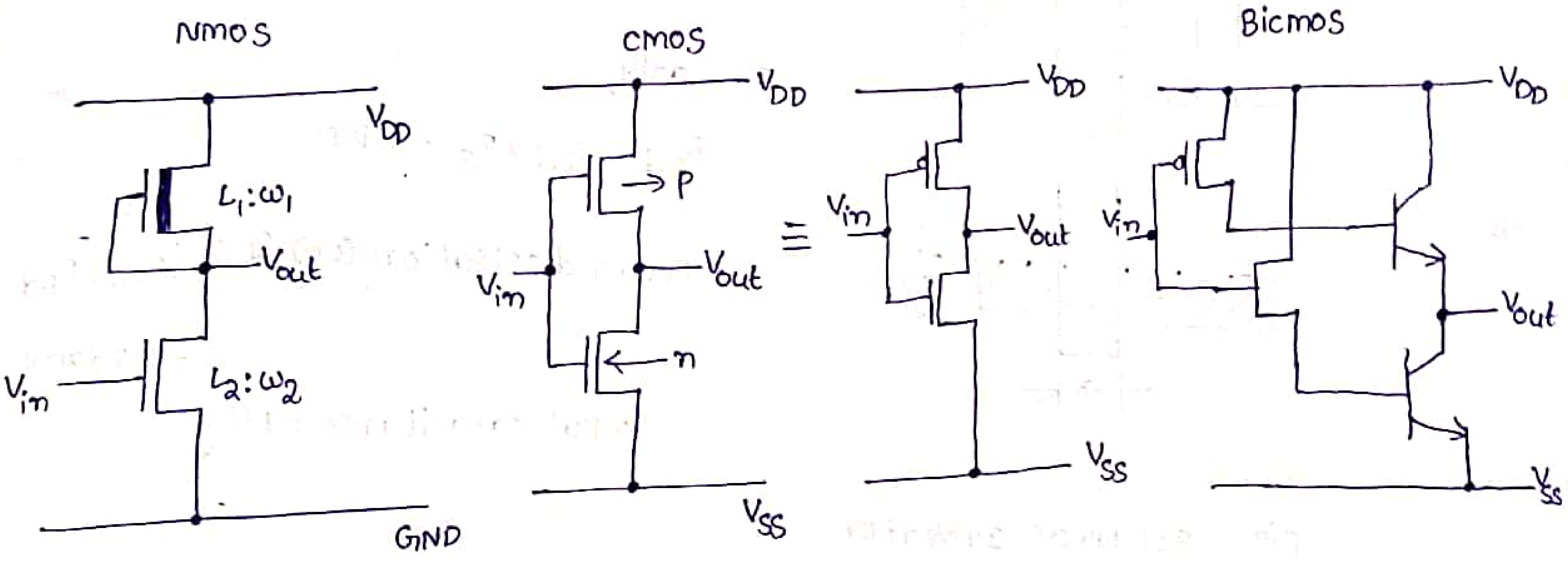
When using Nmos switch logic, there is one restriction which must always observed : No pass transistor gate Input may be driven through one (or) more pass transistors as shown in above figure.

Logic levels propagated through pass transistors are degraded by threshold voltage effects.

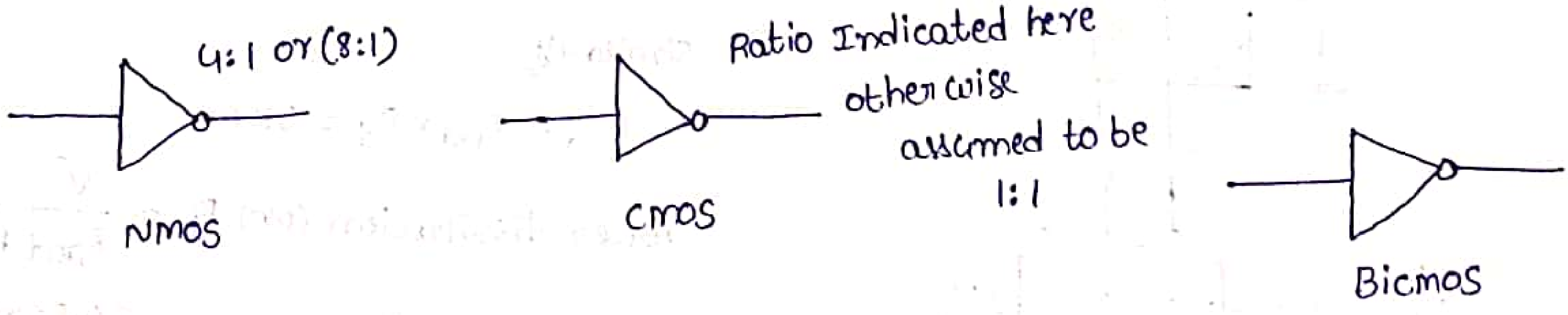
Gate Logic (Restoring Logic) :-

i) Inverter

The NMOS Inverter, having $Z_{p.u}/Z_{p.d}$ ratio and/or the channel length to width for MOS transistor as shown in figure.

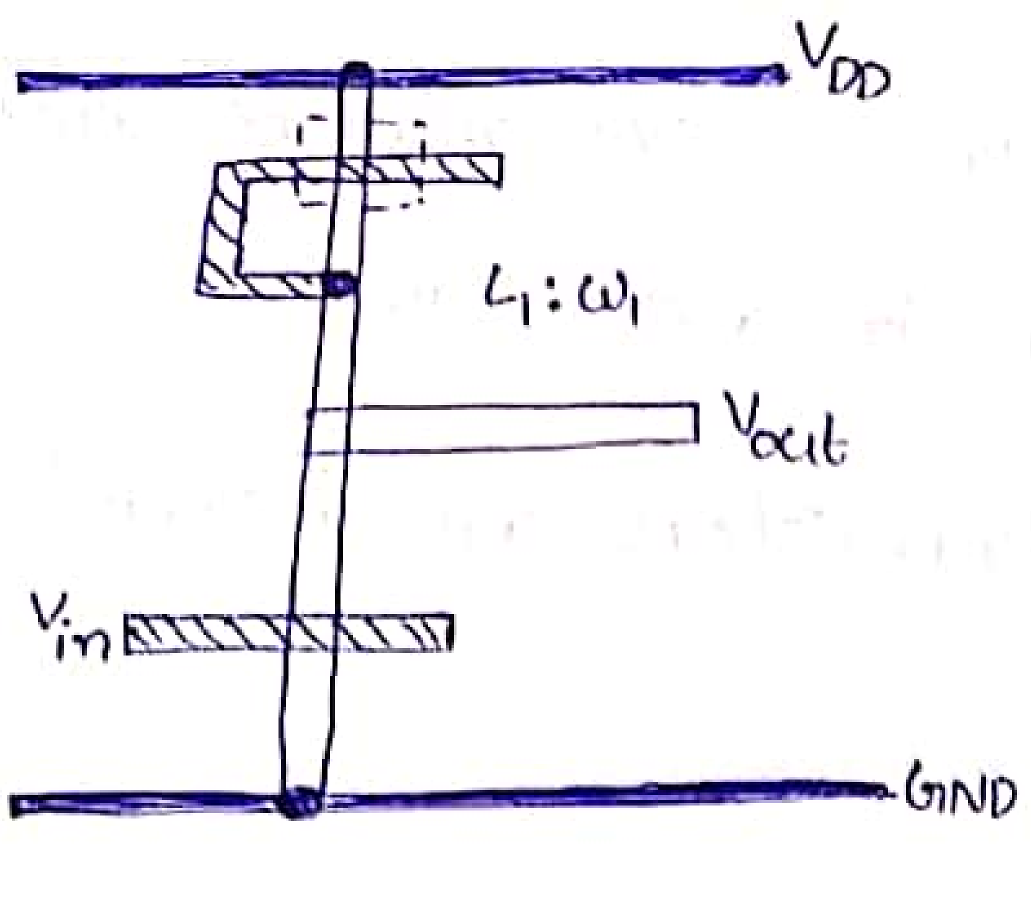


a) circuit symbols (Note:- n-and P-transistors assumed to be min. size unless stated otherwise)

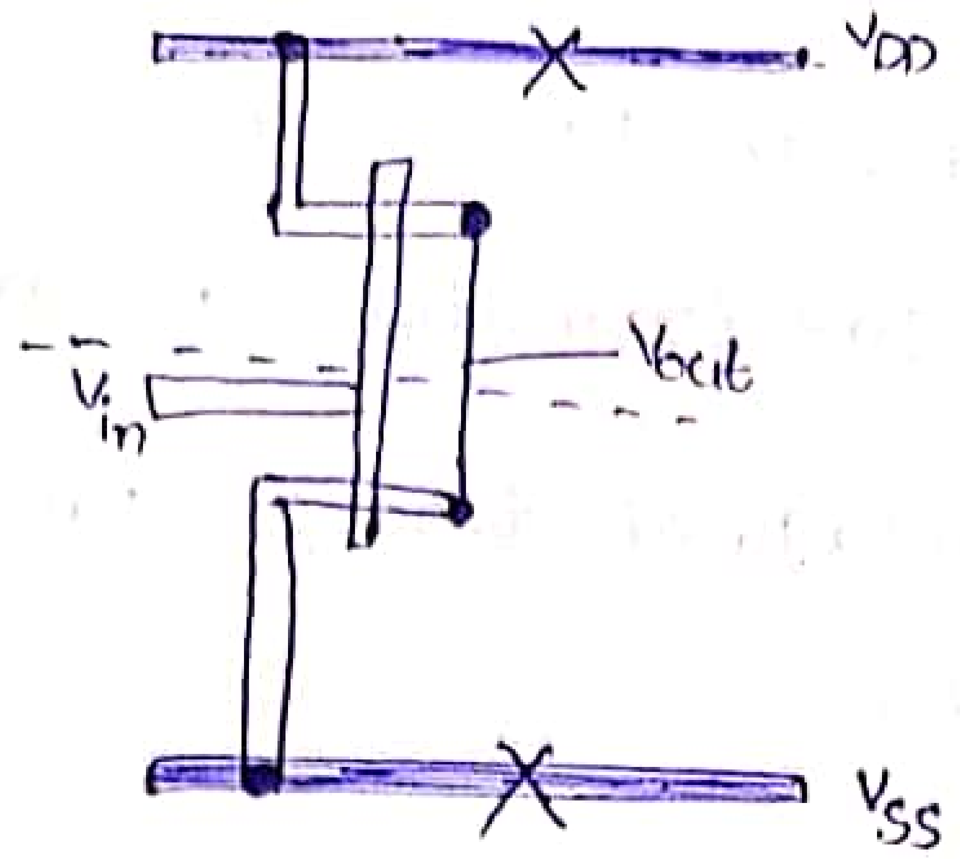


Ratio Indicated here otherwise assumed to be 1:1

b) Logic symbols



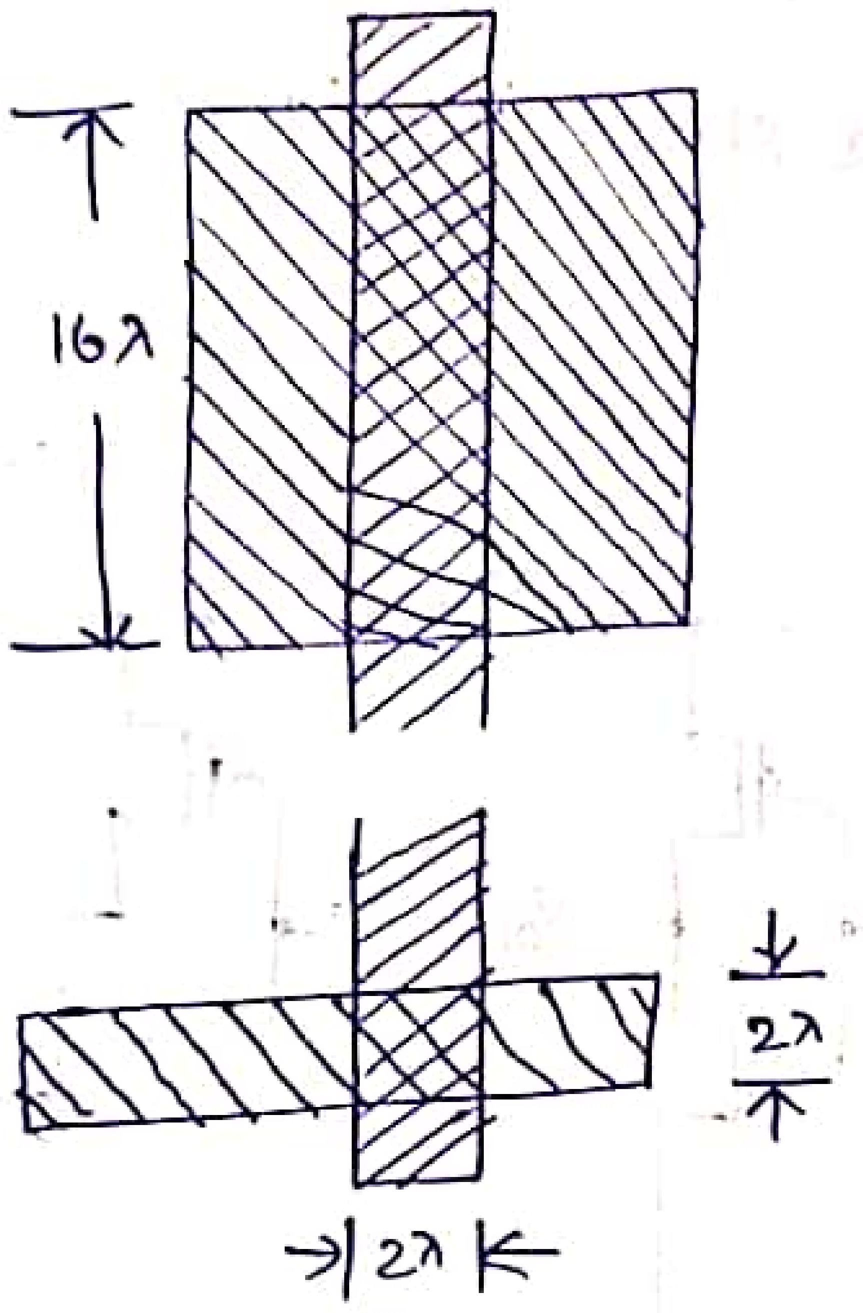
NMOS



CMOS

overall ratio $= \frac{L_1/W_1}{L_2/W_2}$

Two - Input NMOS, CMOS and BiCMOS NAND gates



$Z_{p.u} = L_{p.u}/W_{p.u} = 8$

$R_{p.u} = Z_{p.u} \times R_S = 80k\Omega$ (NMOS)

Similarly,

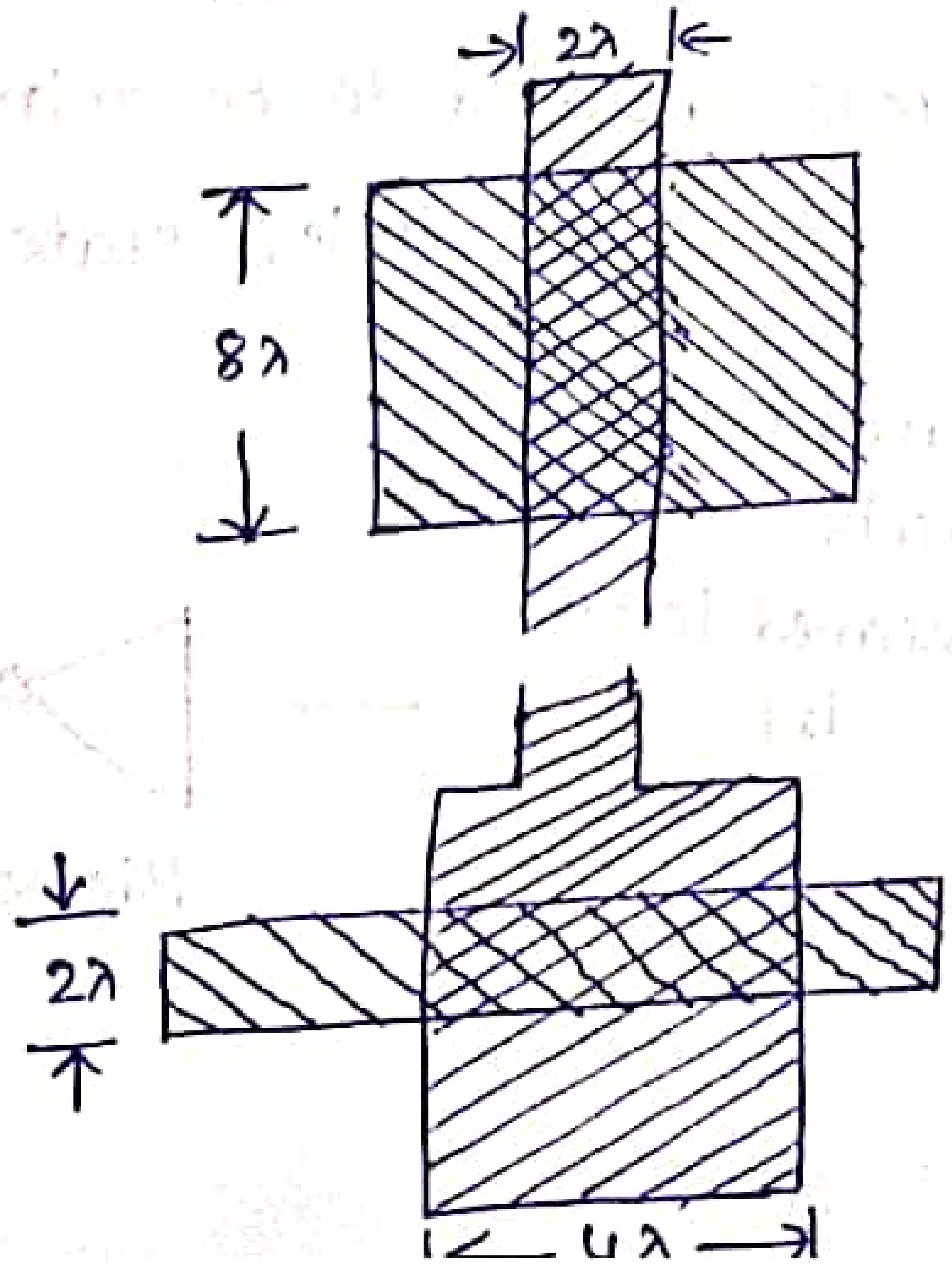
$R_{p.d} = Z_{pd} \times R_S = 10k\Omega$

Power dissipation (or) $P_d = \frac{V^2}{R_{pu}+R_{pd}}$

$= 0.28mW$

Input capacitance $= 1\square C_g$

Fig :- 8:1 NMOS Inverter



$Z_{p.u} = L_{p.u}/W_{p.u} = 4$

$R_{p.u} = Z_{pu} \times R_S = 40k\Omega$ (NMOS)

Similarly,

$R_{pd} = Z_{pd} \times R_S = 5k\Omega$

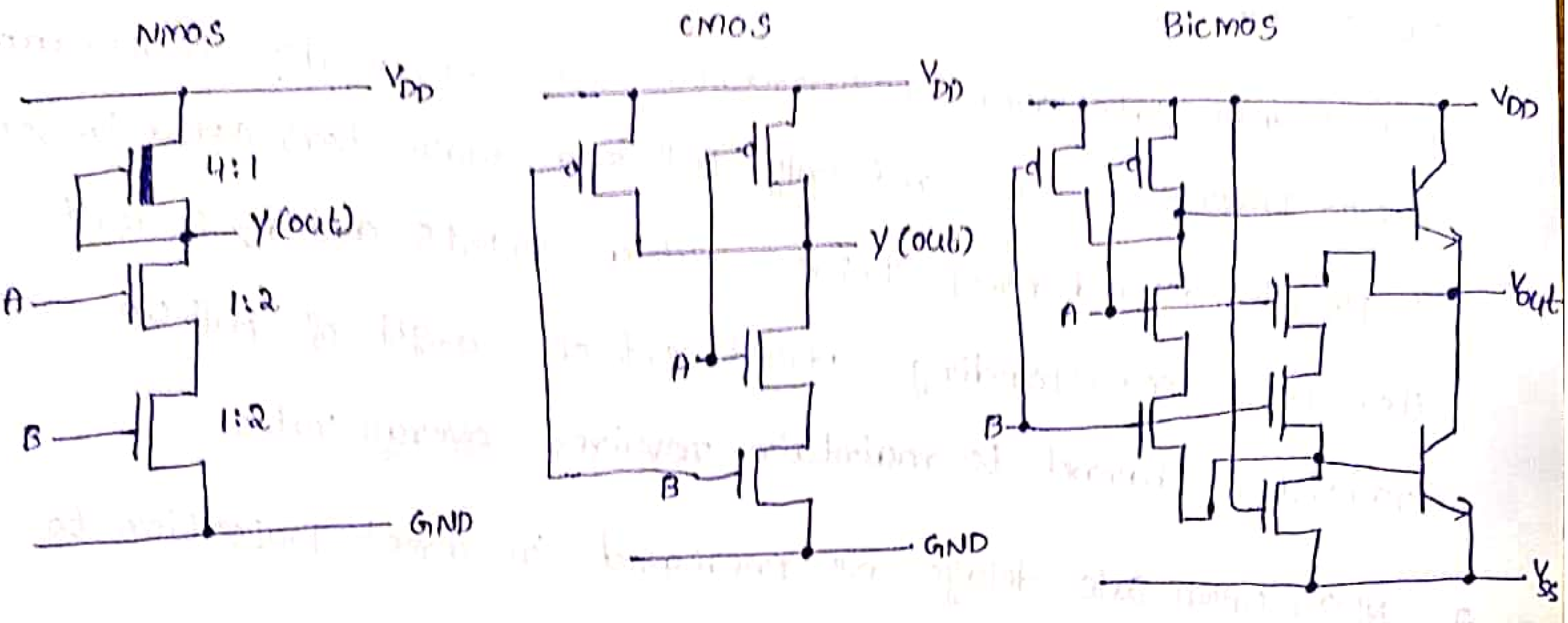
Power dissipation (or) $P_d = \frac{V^2}{R_{pu}+R_{pd}}$

$= 0.56mW$

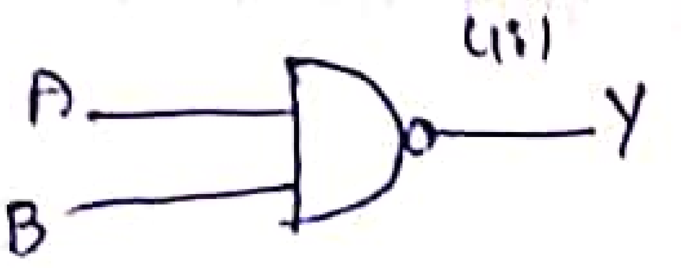
Input capacitance $= 2\square C_g$

Fig :- 8:1 NMOS Inverter (Alternative)

Two Input NMOS, CMOS, Bicmos NAND gates as shown in fig. below.



a) circuit diagrams



b) Logic symbol

consider the simple circuit model of the gate in the condition when all n-pull down transistors are conducting as shown in fig. below

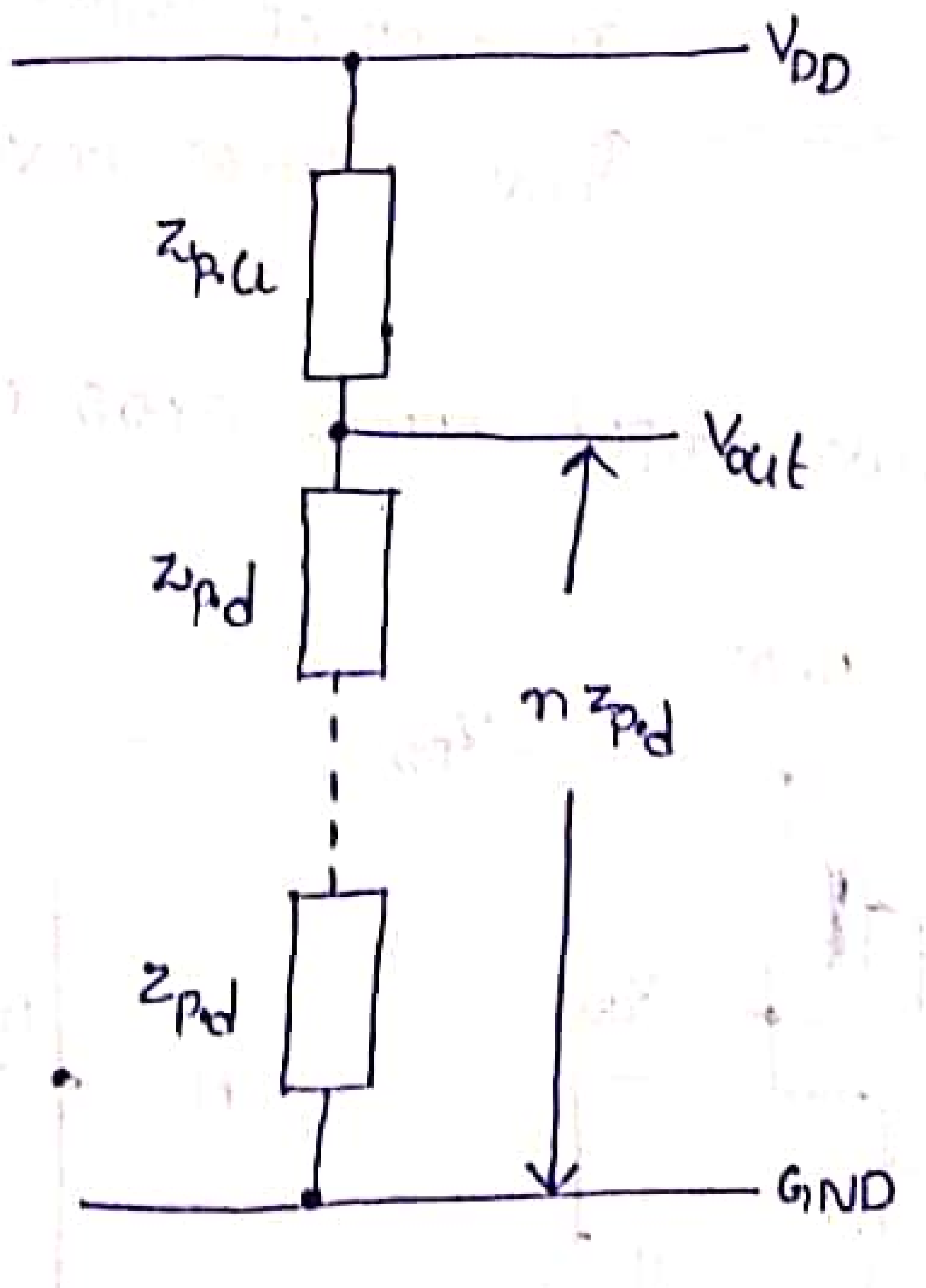


Fig:- NMOS NAND ratio determination

$V_{out} \le V_t = 0.2\,V_{DD}$

Thus $\dfrac{V_{DD} \times n \cdot Z_{p.d}}{nZ_{p.d} + Z_{p.u}} \le 0.2\,V_{DD}$

Z_{pd} applies for any one pull-down transistor.

The boundary condition is,

$$\frac{nZ_{p.d}}{nZ_{p.d} + Z_{p.u}} = 0.2$$

NMOS NAND ratio $= \dfrac{nZ_{p.d}}{nZ_{p.d}} = \dfrac{4}{1}$

The Pull-up to Pull-down ratio must be 4:1

It contains 2 significant factors

1. NMOS NAND gate area requirements are considerably greater than NMOS Inverter, since not only Pull-down transistors added in series to provide desired no.of Inputs, but as Inputs added, so must there be a corresponding adjustment of length of Pull-up transistor, channel to maintain required overall ratio.

2. NMOS NAND gate delays are Increased in direct Proportion to the No.of Inputs added.

If each Pull-down transistor kept to minimum size ($2\lambda \times 2\lambda$) then each will present $1\square C_g$ at its Input.

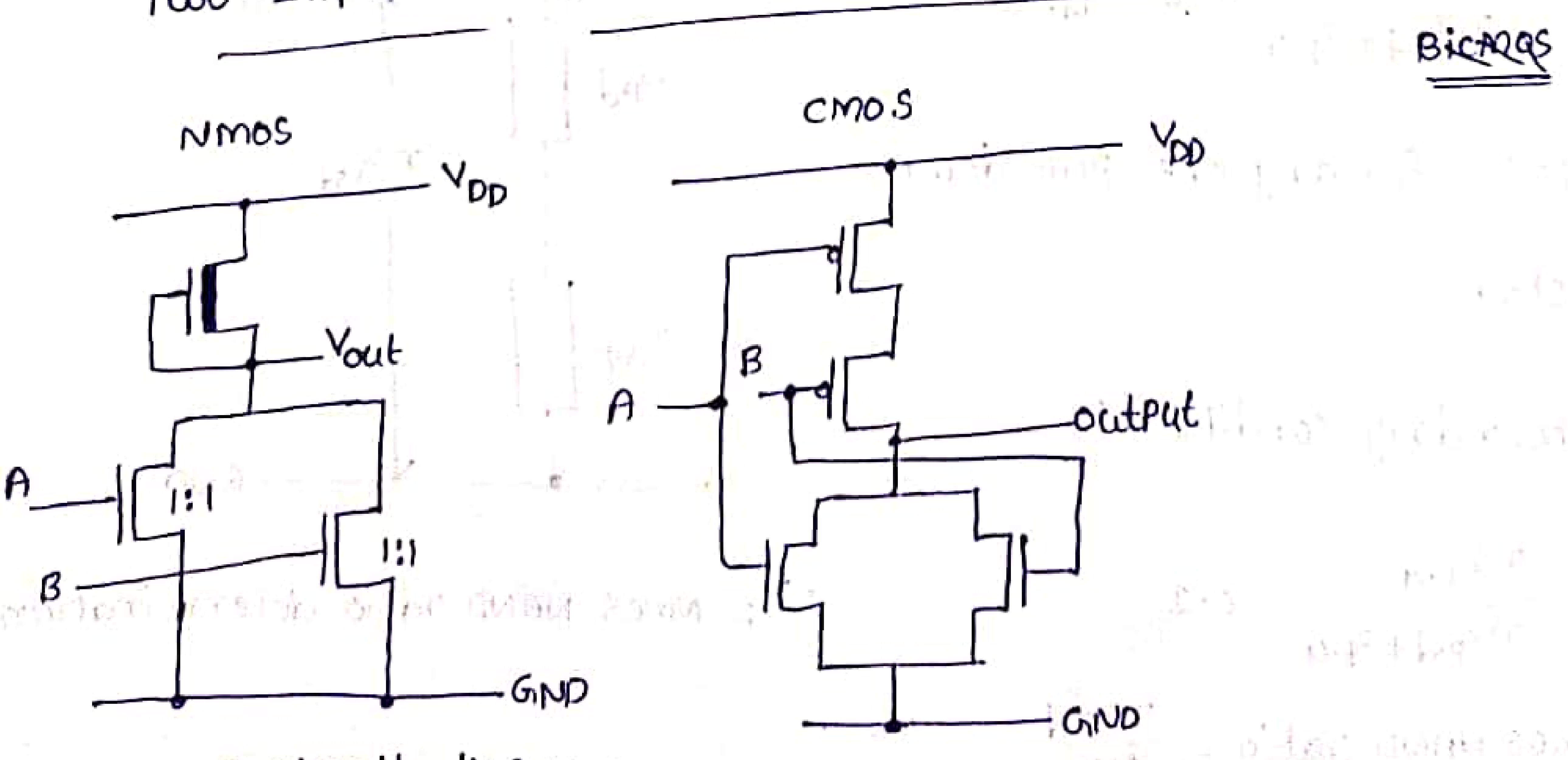
The delay associated with NMOS NAND are

$$\boxed{\tau_{NAND} = n\tau_{Inv}}$$

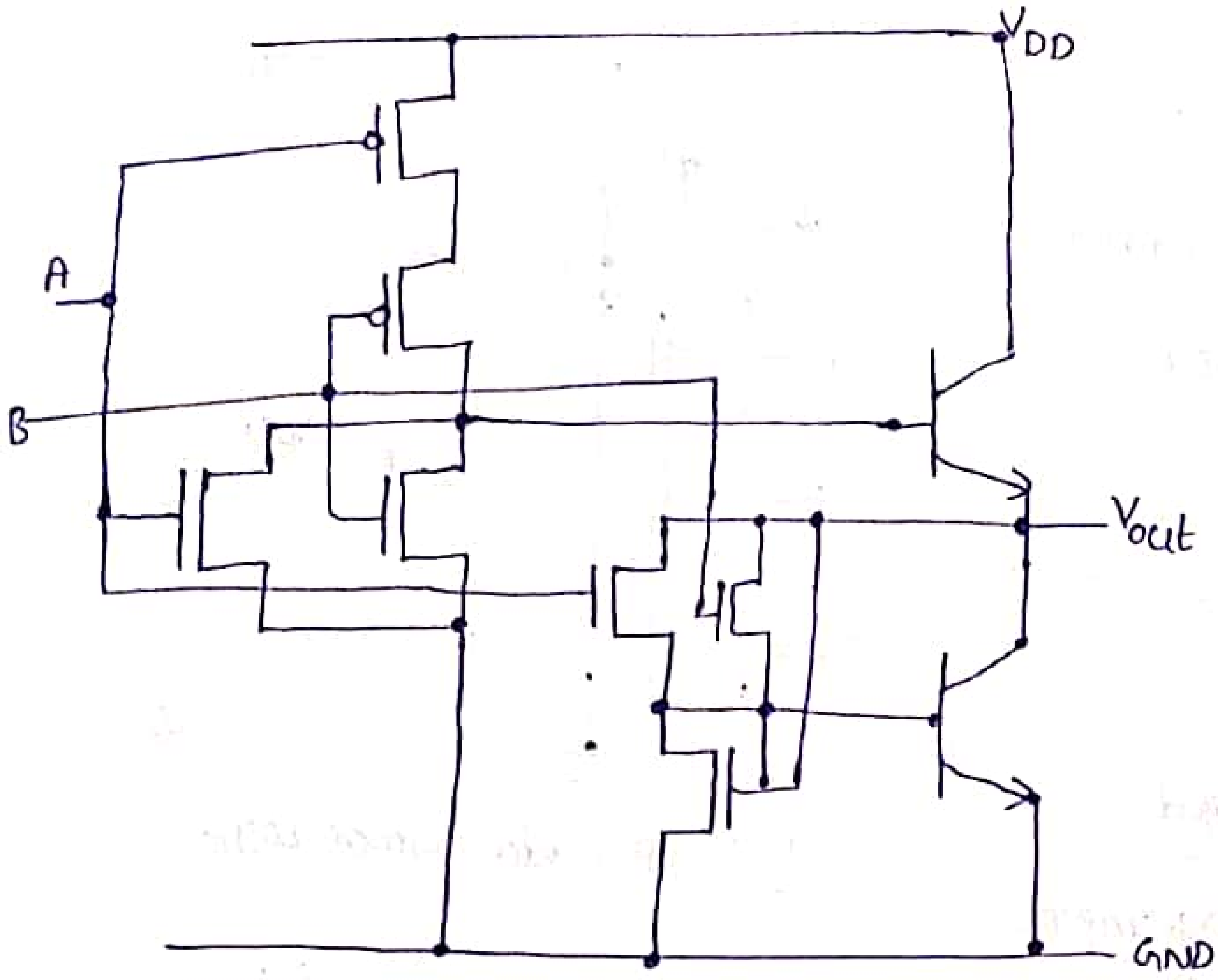
$n \rightarrow$ No.of Inputs

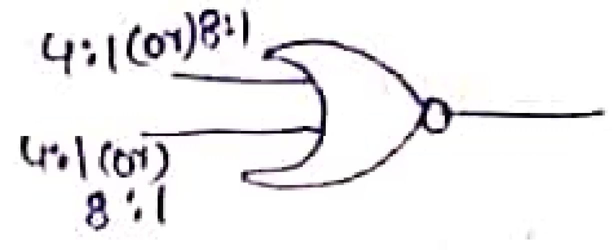
$\tau_{Inv} \rightarrow$ NMOS Inverter delay

Two-Input NMOS, CMOS and BiCMOS NOR gates :-



a) circuit diagrams





b) Logic symbol

The area occupied by the NMOS NOR gate is reasonable since the pull-up transistor dimensions are unaffected by the no. of Inputs.

In CMOS NOR gate, consists of a pull-up p transistor based structure, which Implements the logic 1 conditions, and a n-transistor Implements the logic 0 conditions at the output.

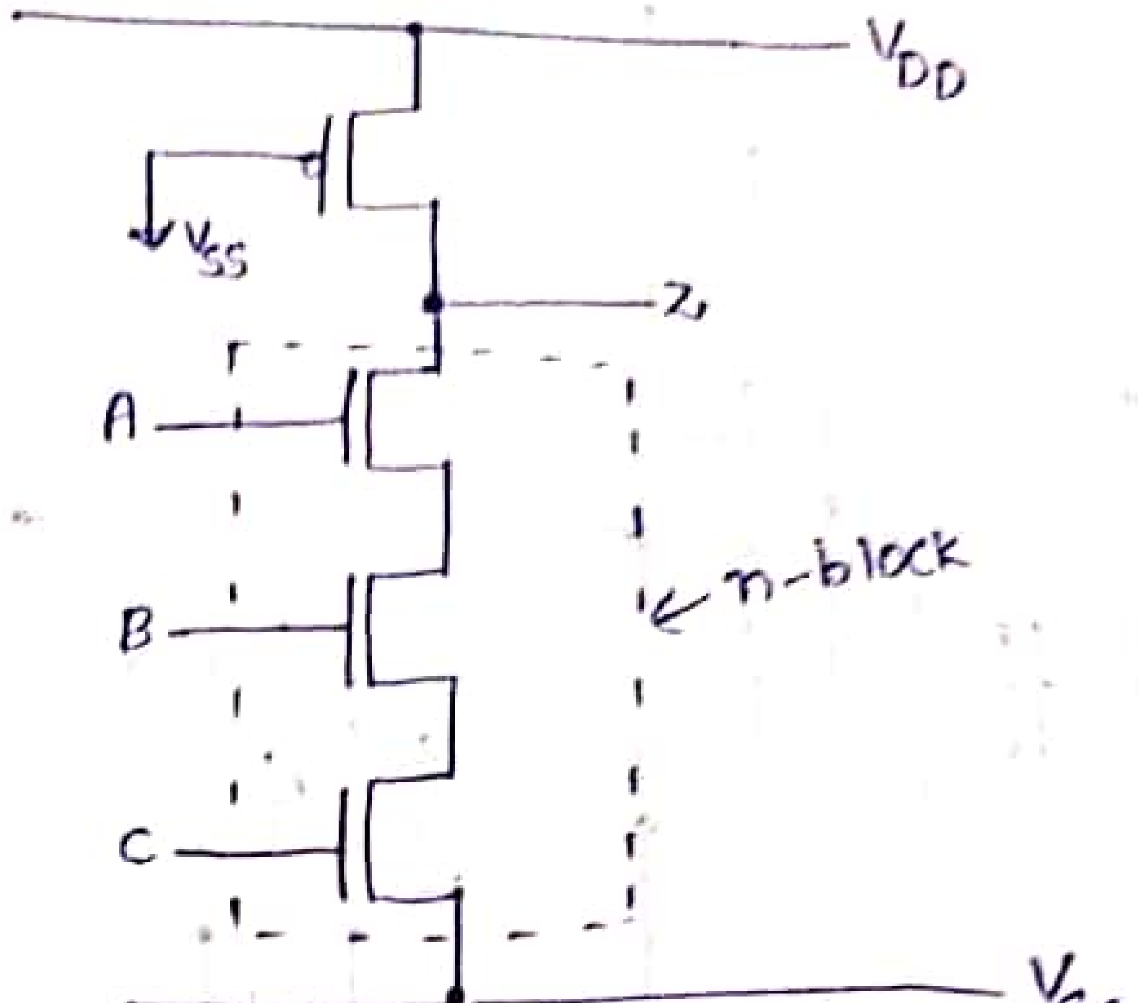
For BiCMOS NOR gate most useful where a large fan-out is required.

other forms of CMOS Logic :- mainly consists of

1. pseudo- NMOS Logic
2. Dynamic CMOS Logic
3. clocked CMOS (C^2MOS) Logic
4. CMOS domino Logic
5. n-p CMOS Logic

1. Pseudo- Nmos Logic :.

If we replace the depletion mode pull-up transistor of standard Nmos circuits with a p-transistor gate connected to V_{SS}.



Fig①.- Pseudo - Nmos Logic

To determine the required ratio, we consider the arrangement as shown in fig ② is which pseudo-Nmos Inverter is being driven by another similar Inverter.

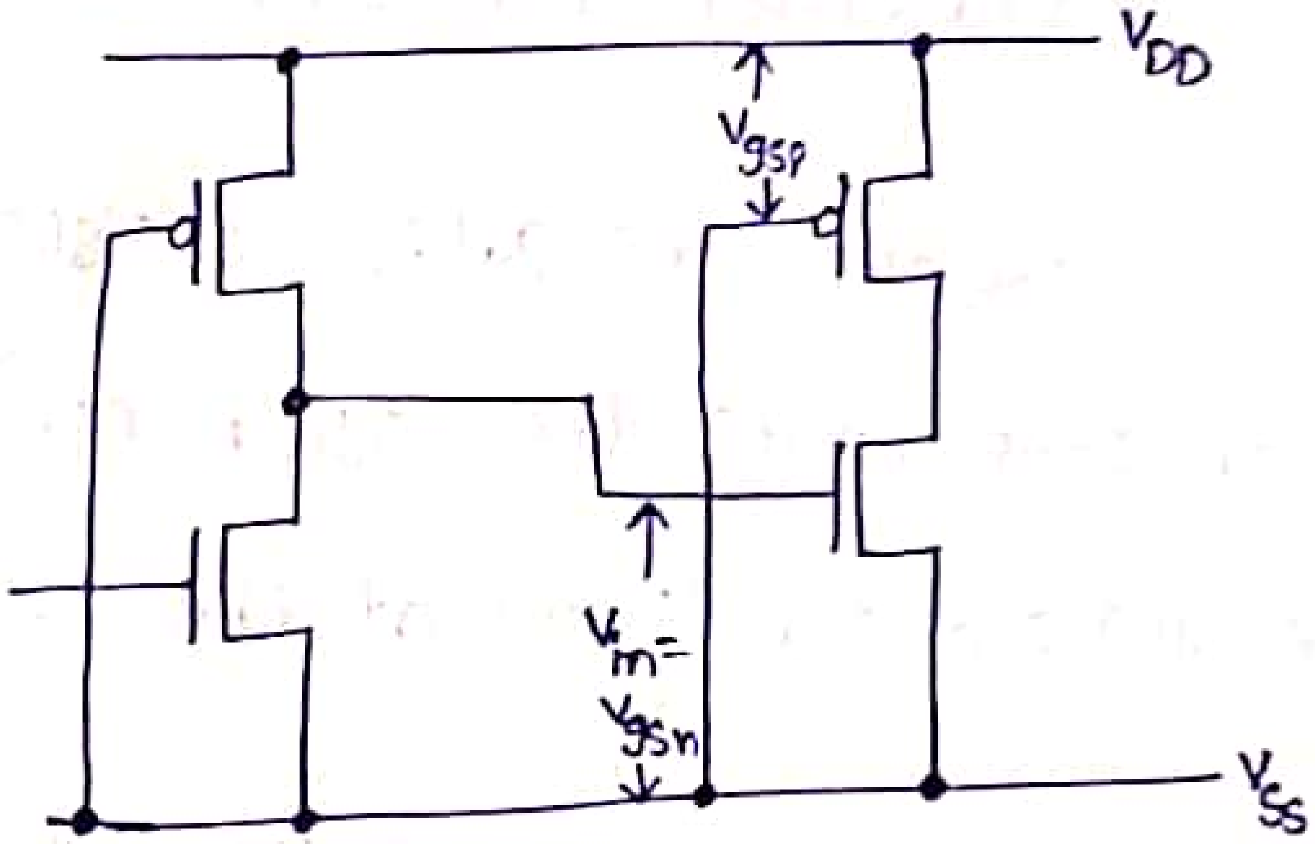


Fig :- pseudo- Nmos Inverter when driven from a Similar Inverter

For Nmos analysis, we consider

$V_{inv} = \frac{V_{DD}}{2}$

At this point n-device is in saturation $(0 < V_{gsn} - V_{tn} < V_{dsn})$ and p-device is operating in resistive region $(0 < V_{dsp} < V_{gsp} - V_{tp})$

Equating currents of the n-transistor & p-transistor,

$$V_{inv} = V_{tn} + \frac{(2\mu_p/\mu_n)^{1/2} \left[(-V_{DD} - V_{tn}) V_{dsp} - V_{dsp}^2\right]^{1/2}}{(Z_{p.u}/Z_{p.d})^{1/2}}$$

where $Z_{p.u} = L_p/W_p$

$Z_{p.d} = L_n/W_n$

$V_{inv} = 0.5\ V_{DD}$

$V_{tn} = |V_{tp}| = 0.2\ V_{DD}$

$V_{DD} = 5V$

$\mu_n = 2.5\ \mu_p$

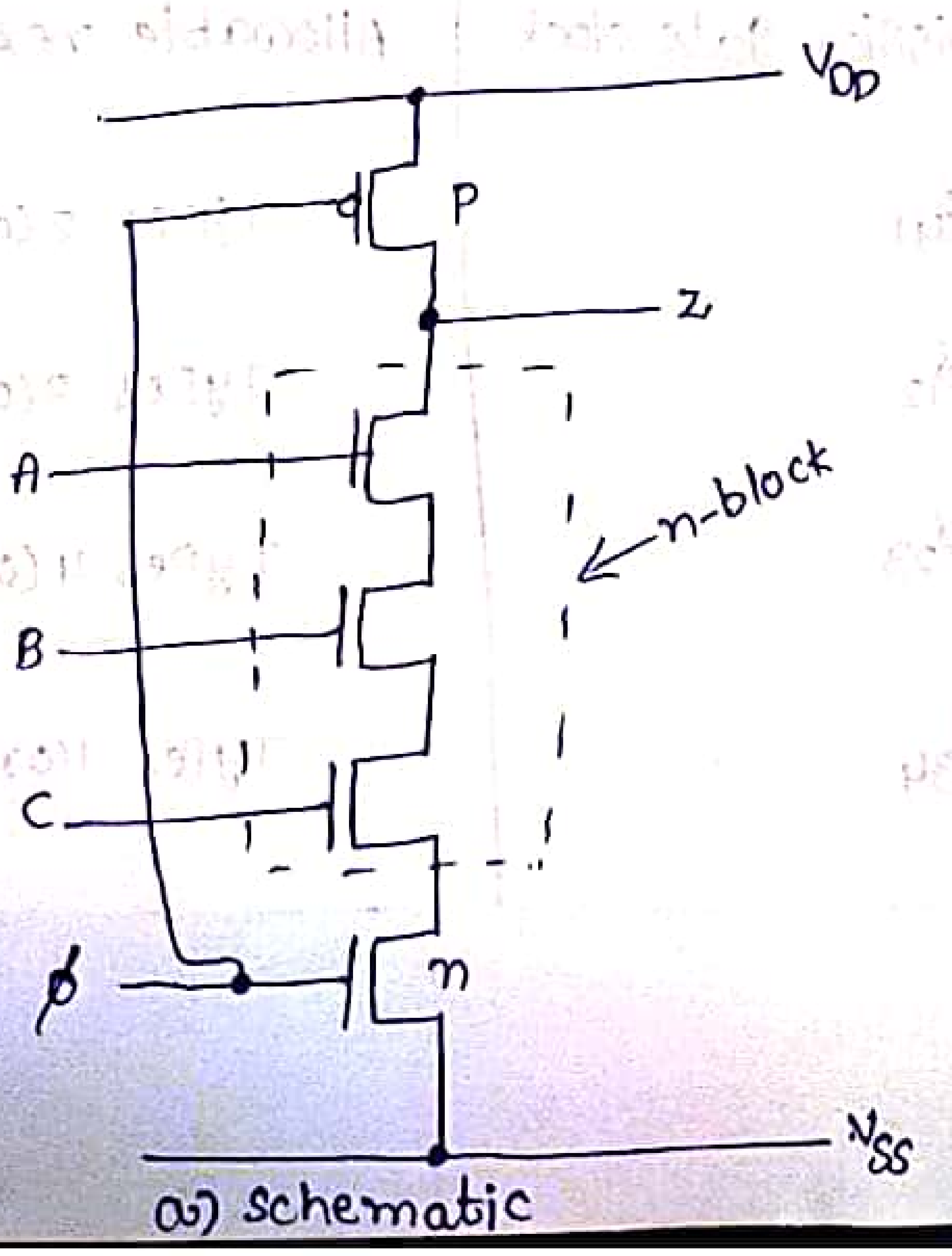
we obtain $\frac{Z_{p.u}}{Z_{p.d}} = \frac{3}{1}$

The channel sheet resistance of the P-pullup is about 2.5 times that of the n-pull down, and allowing for the ratio 3:1,

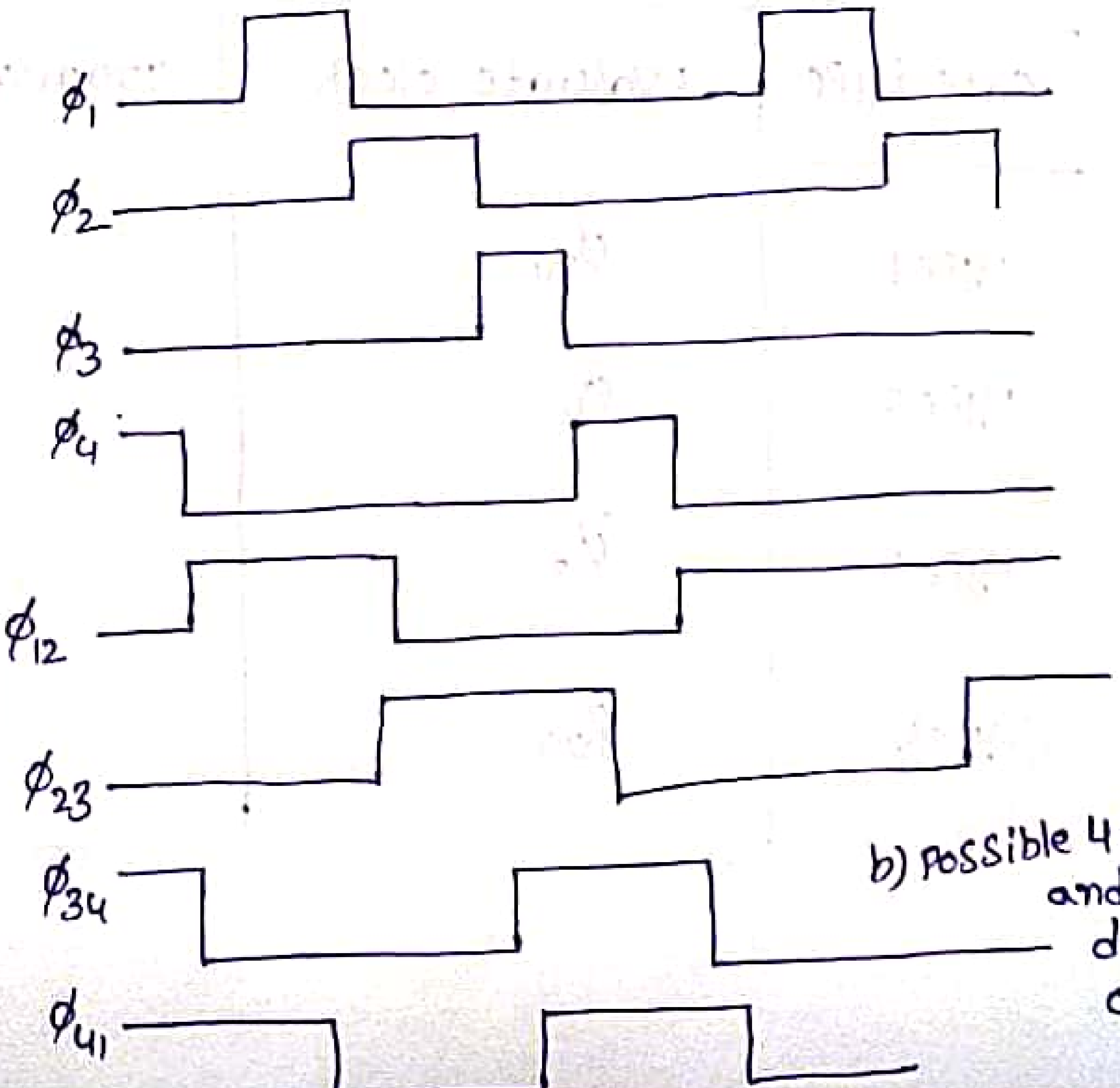
2) Dynamic cmos Logic :-

1. charge sharing may be problem unless the Inputs are constrained not to change during the on period of the clock.

2. single phase dynamic Logic structures can not be cascaded since, owing to circuit delays, an Incorrect Input to next stage may be present when evaluation begins, so output is discharged and wrong output results.



a) schematic



b) possible 4φ and derived clocks

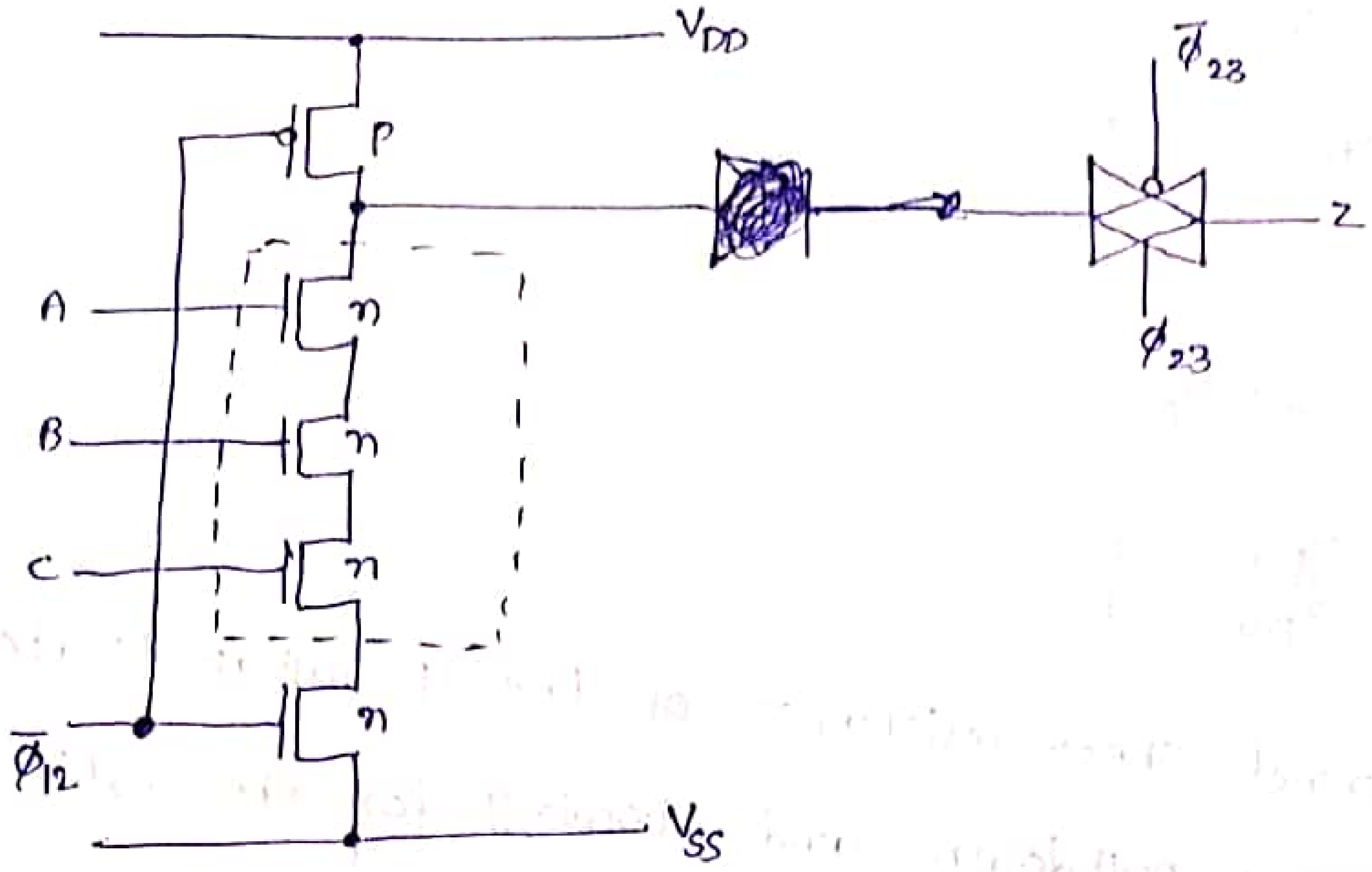


Fig c) :- Type 3 arrangement

Fig :- Dynamic cmos Logic three-Input NAND gate

P-transistor is used for the non-time-critical pre charging of the outline 'z' so that output capacitance is charged to V_{DD} during the off period of the clock signal ϕ.

To avoid erroneous evaluations, the gates must be connected in allowable sequences as shown in table.

Gate type	Evaluate clock	Transmission gate clock	Allowable next types
Type 1	$\overline{\phi}_{34}$	ϕ_{41}	Types 2 (or) 3
Type 2	$\overline{\phi}_{41}$	ϕ_{12}	Types 3 (or) 4
Type 3	$\overline{\phi}_{12}$	ϕ_{23}	Types 4 (or) 1
Type 4	$\overline{\phi}_{23}$	ϕ_{34}	Types 1 (or) 2

G. Venkata Hanuman
Asst. Prof

4. Chip Input and output circuits

Introduction :-

→ The I/o circuits, clock generation, Distribution circuits are essential to VLSI chip design.

→ The design Quality of these circuits is a critical factor that determines Reliability, Integrity and Interchip communication speed of chip.

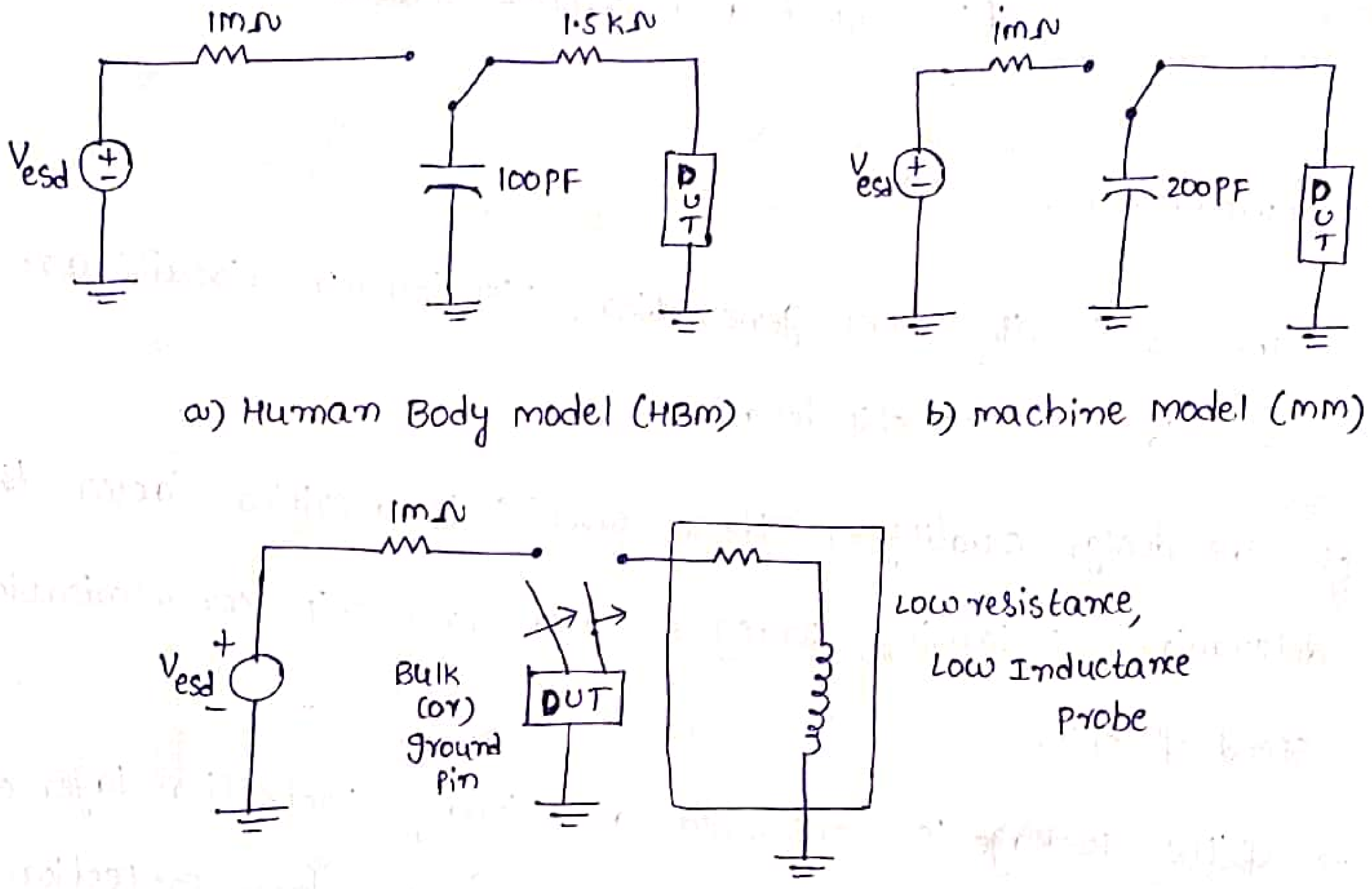
→ If the Package is considered a primary Protection layer of silicon chip. I/o frame, clock circuits are secondary protection layers.

→ External hazards such as ESD (Electro static discharges) Noise should be filtered out before propagating to Internal circuits for their protection.

ESD Protection :-

Electro Static discharge is one of the most Prevelant cause for chip failures in both chips manufacturing and field openings.

ESD occurs when the charge stored in machines (or) humans discharged to chip on contact (or) By static Induction.



a) Human Body model (HBm)

b) machine model (mm)

c) charged- Device model (CDm)

1. HBm :-

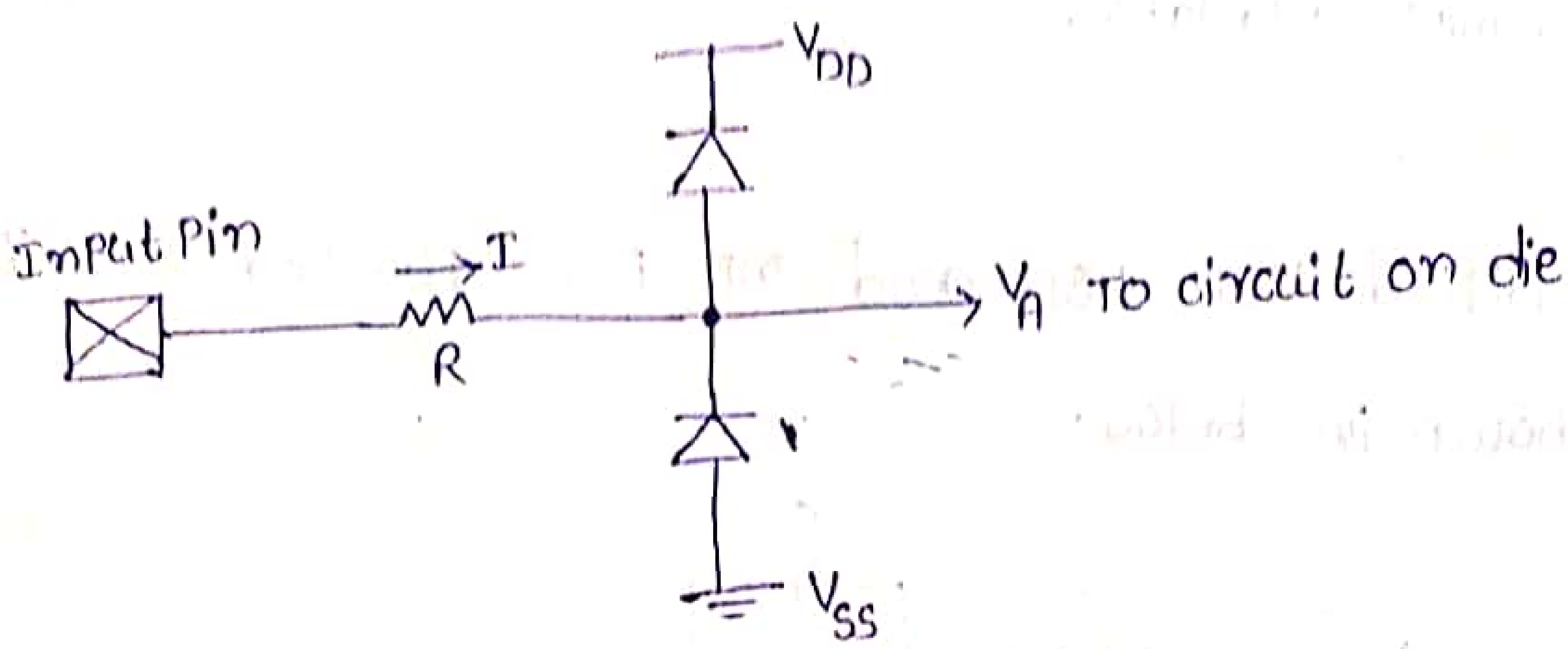
→ A human walking across synthetic carpet in 80% of relative humidity can Potentially Induce 1.5KV of static Voltage stress.

→ In HBM mode, touch from a charged person's finger is simulated by discharging 100PF capacitor through a 1.5 KΩ resistor.

→ A protection Network must be Inserted into I/o circuits of chip so that the ESD effect can be filtered out before it is propagated to Internal logic circuit.

Protection Network

The Protection Network consists of a diffused resistor diode structure with equivalent circuit is



The Input resistance is normally 200Ω to 3KΩ

The diodes clamp the signal level with in a certain Voltage rang in order to minimize ESD.

$$-0.7V < V_A \leq (0.7V + V_{DD})$$

To protect the diode structure, the current through the diode is limited to less than several tens of milli amperes. For this purpose, poly silicon series resistors are used but they are damaged due to dielectric break down at high electric fields.

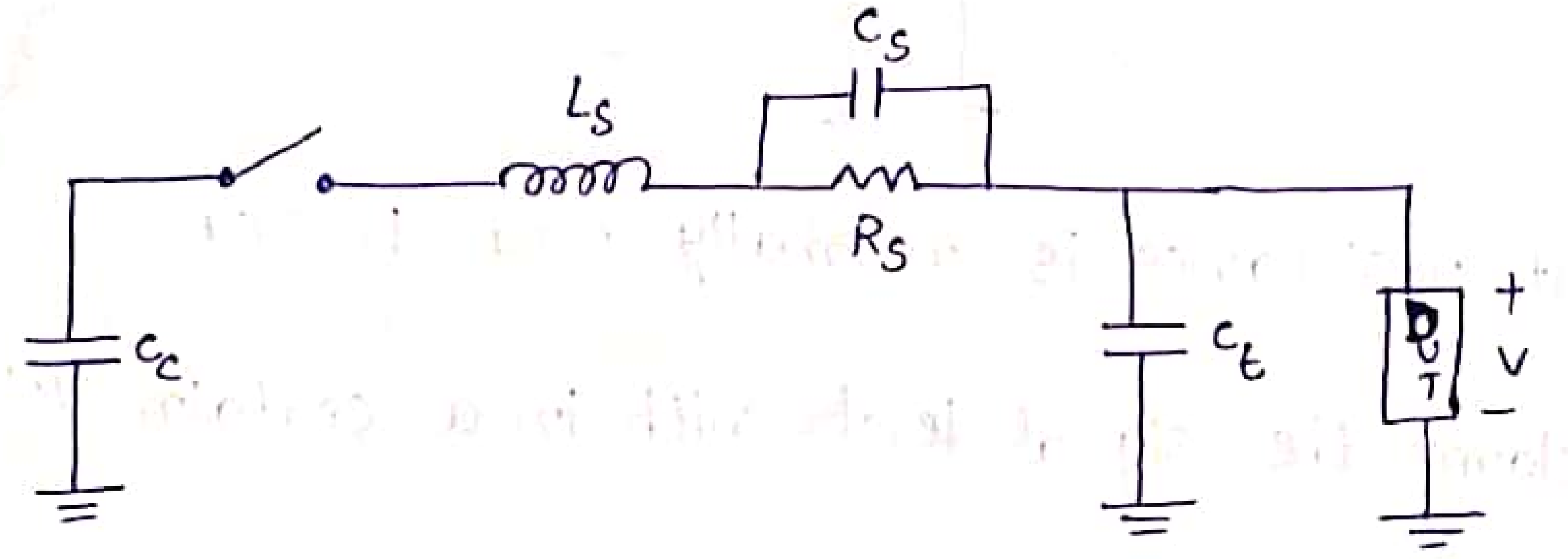
2. mm (machine model) :-

In this model, the body resistance is absent, contact with machines can be higher stress.

3. charged device model (CDM) :-

Electro static charge builds up on a chip due to Improper grounding and then discharges when a Low-resistance path becomes available.

→ simplified HBM ESD and mm ESD Lumped circuit models are shown in below.



component	HBM	mm
C_c (PF)	100	200
R_S (Ω)	1500	25
L_S (μH)	5	2.5
C_S (PF)	1	0
C_t (PF)	10	10

This is overcome by the use of additional thick oxide transistions as shown in figure.

In this circuit,

M_1 → Thick oxide punch through device

M_2 → Thick oxide Nmos transistor

M_3 → Thick oxide N-mos transistor in saturation mode

For positive Inputs M_1, M_2 have threshold values of 20 to 30V

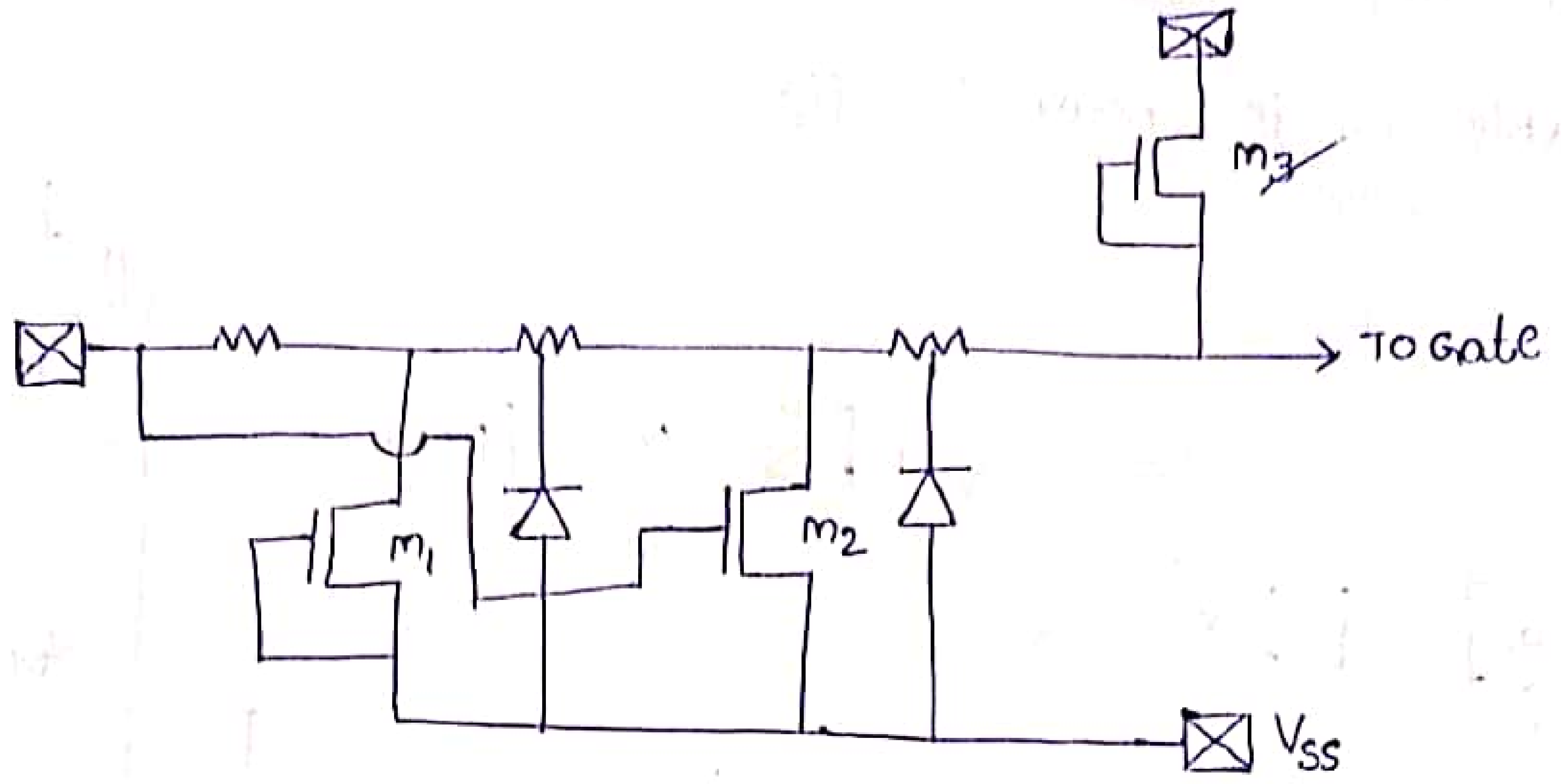
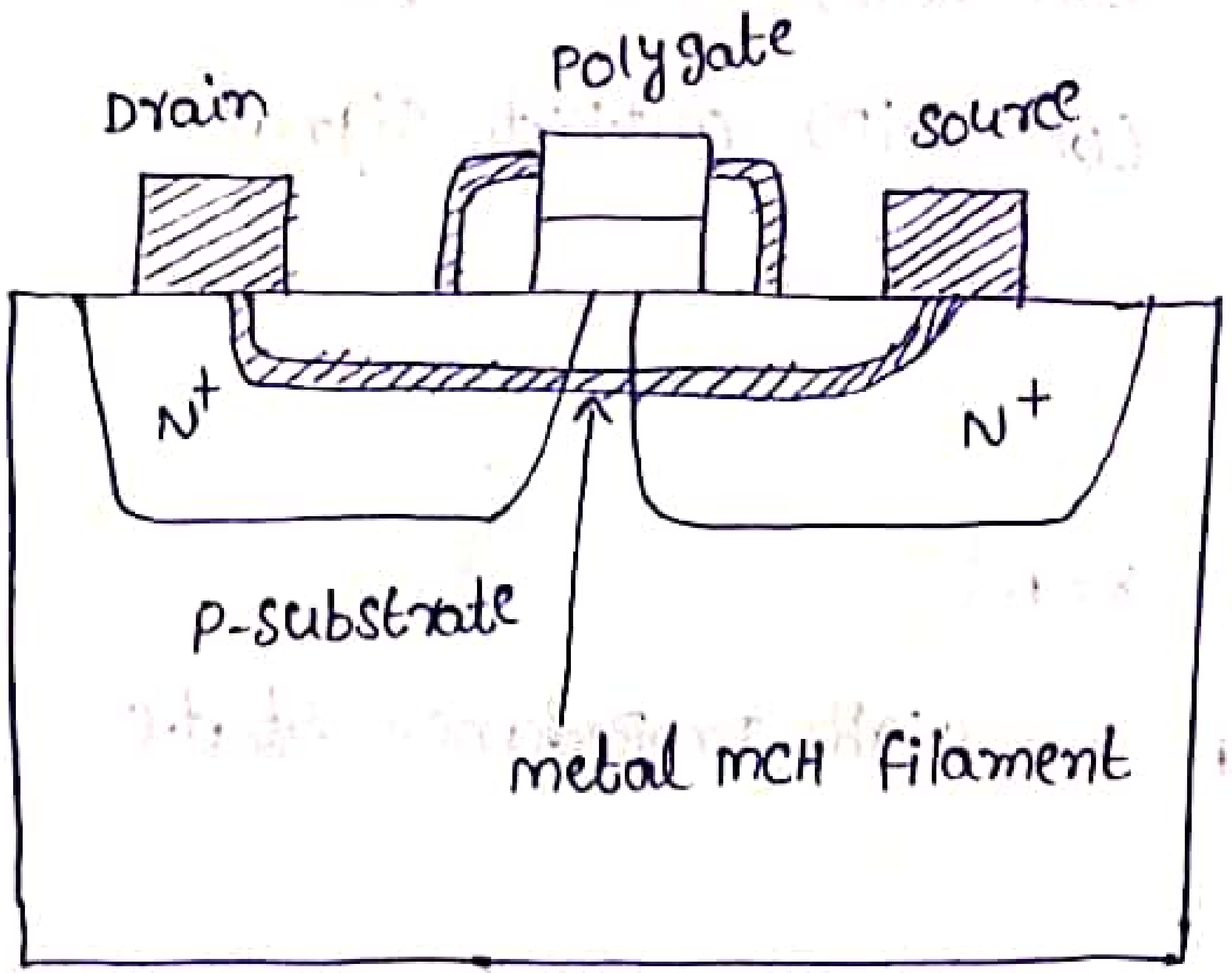


Fig :- Protection Network with thick oxide transistor.

ESD failure modes :-

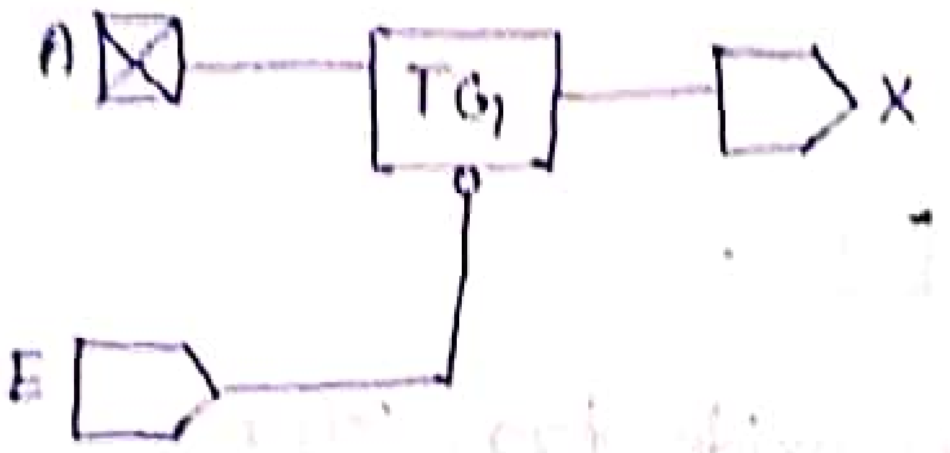
The below figure shows ESD failure modes caused by ESD Induced heat dissipation is an Nmos transistor along with a scanning Electron microscopy (SEM).



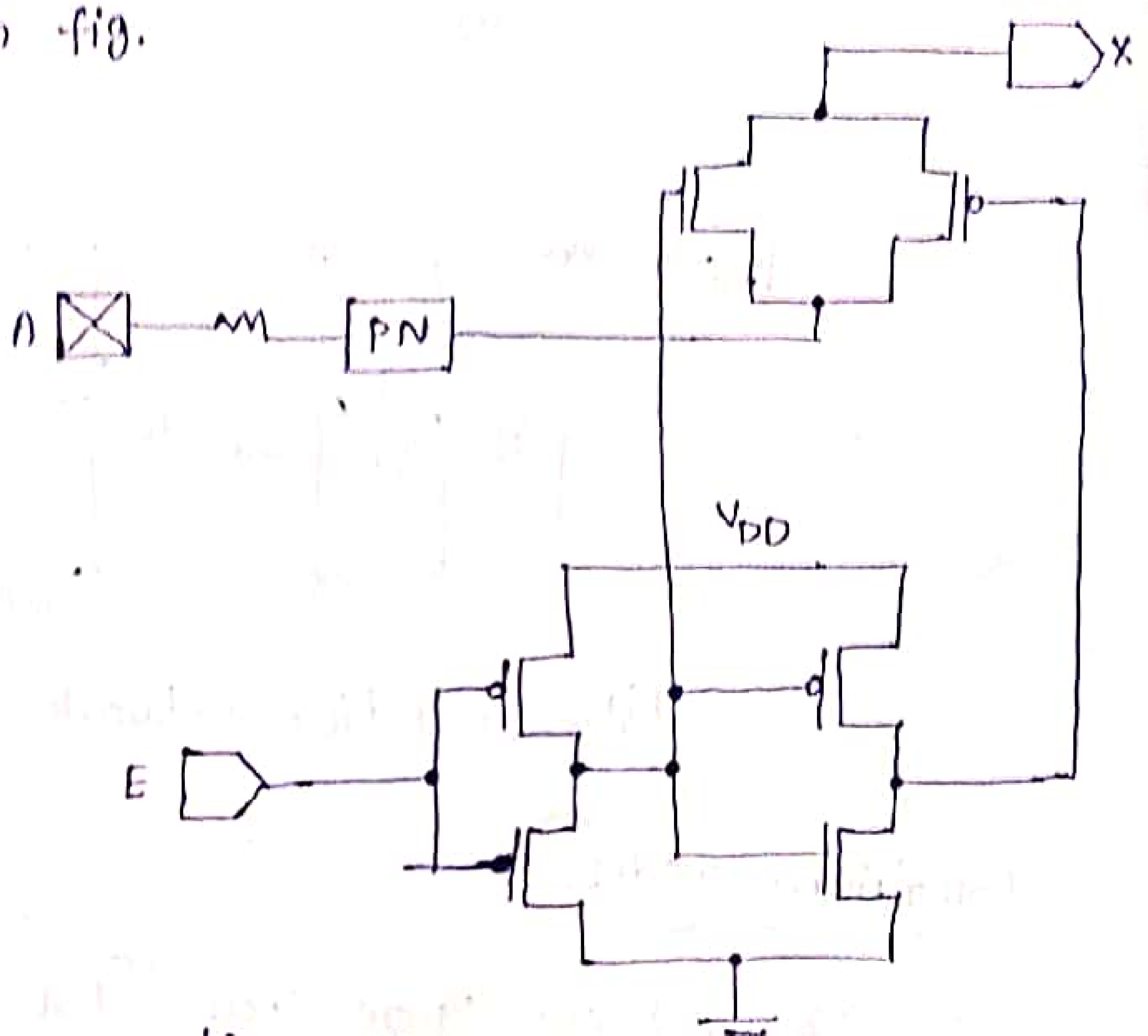
Typical ESD failure modes.

Input circuits :-

A simple Input circuit consists of transmission gate and Enable (E) is shown in fig.



a) Symbol



TG → Transmission gate

PN → protection Network

b) Input series transmission gate circui

A → External (OFF-chip) Input signal

E → Internal (ON-chip) Enable signal

X → Internal (ON-chip) output signal

conditions

1. when E=0, X=A
2. when E=1, X = High-Impedance state

The Incoming signal A is fed to transmission gate through protection Network. The Input Pad circuit modules have a built in Internal pull up (or) pull down resistor with a resistance of 200kΩ to 1MΩ.

For Example,

Inverting Input circuit with TTL-to- cmos level shift as shown in fig.

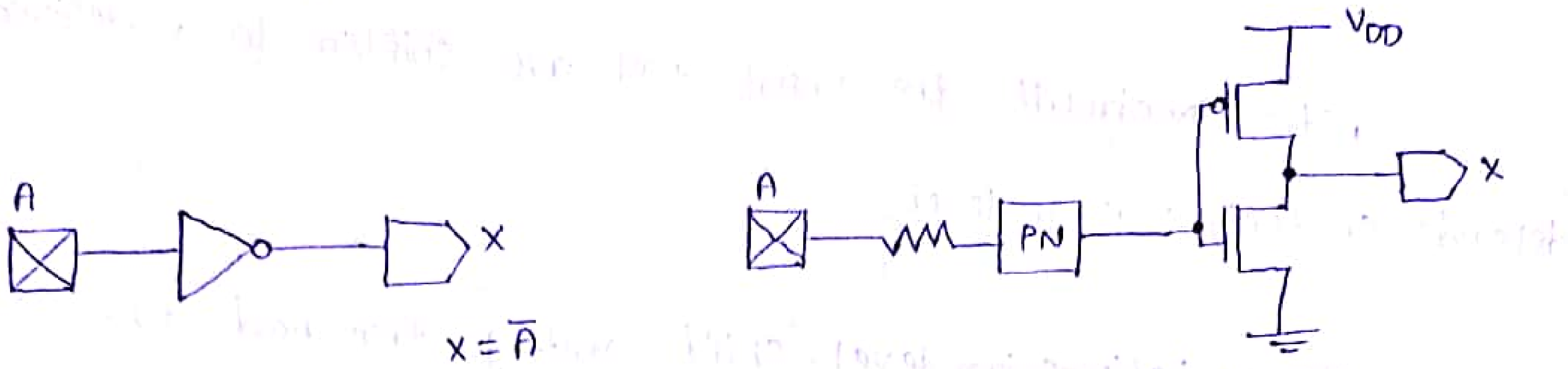


Fig:- Inverting Input circuit with Protection N/w

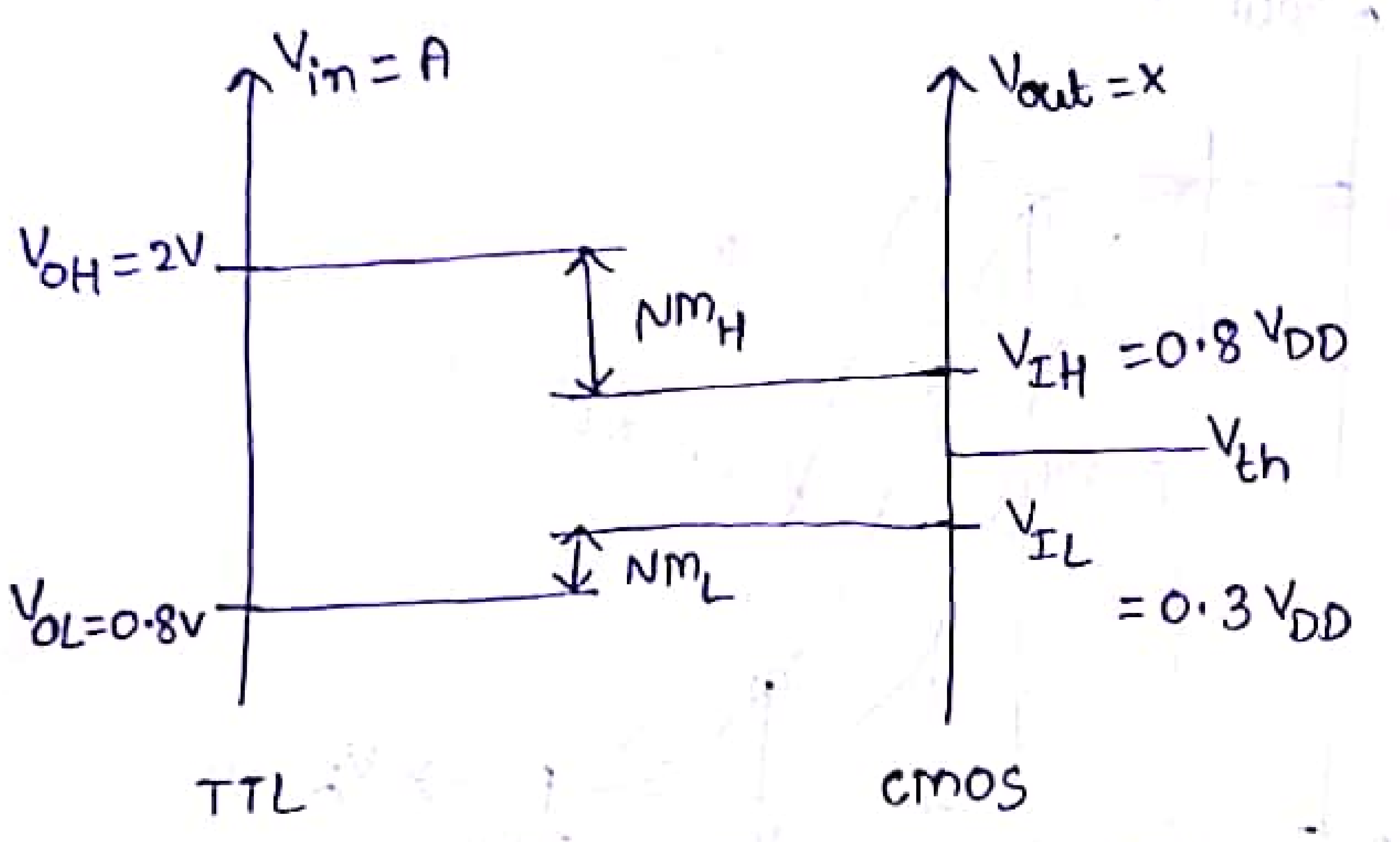


Fig:- Noise margin levels

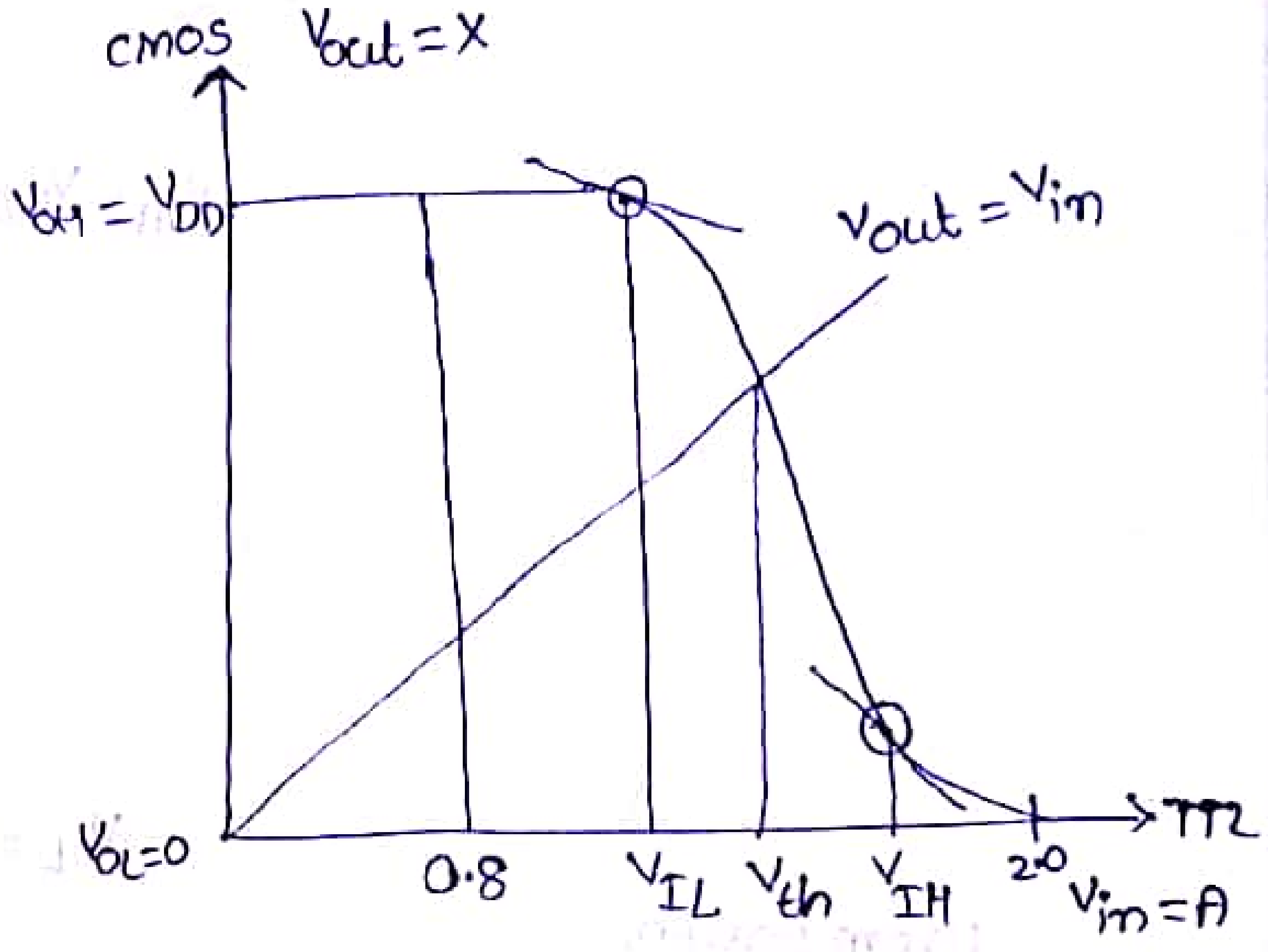


Fig:- voltage transfer characteristics

For TTL, The Noise margin levels are $V_{OL} = 0.8V$, $V_{OH} = 2V$

For cmos, the noise margin levels are $V_{IL} = 0.3 V_{DD}$, $V_{IH} = 0.8 V_{DD}$

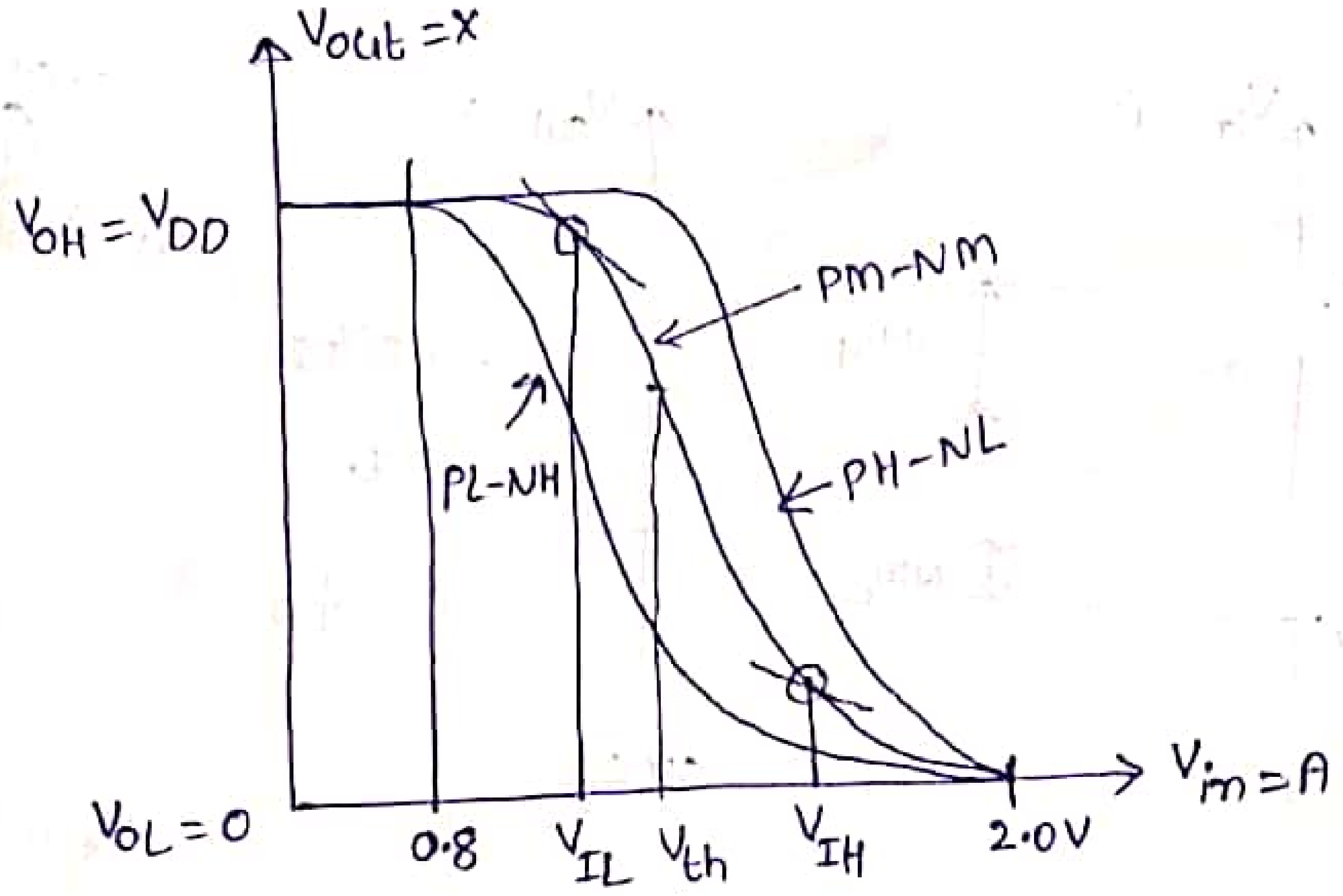
$$NM_L = V_{IL} - V_{OL}$$

$$NM_H = V_{OH} - V_{IH}$$

From Voltage transfer characteristics curve, the voltages below 0.8V should be Interpreted as Low and voltages above 2V Interpreted as High.

After PN circuit, the Input levels are shifted to a desired level, depends on their voltage levels.

The variations in level-shift voltage threshold transfer characteristics dueto process variations as shown in fig.



From Graph,

PM-NM → Nominal processing

PH-NL → strong PMOS, weak NMOS Process corner

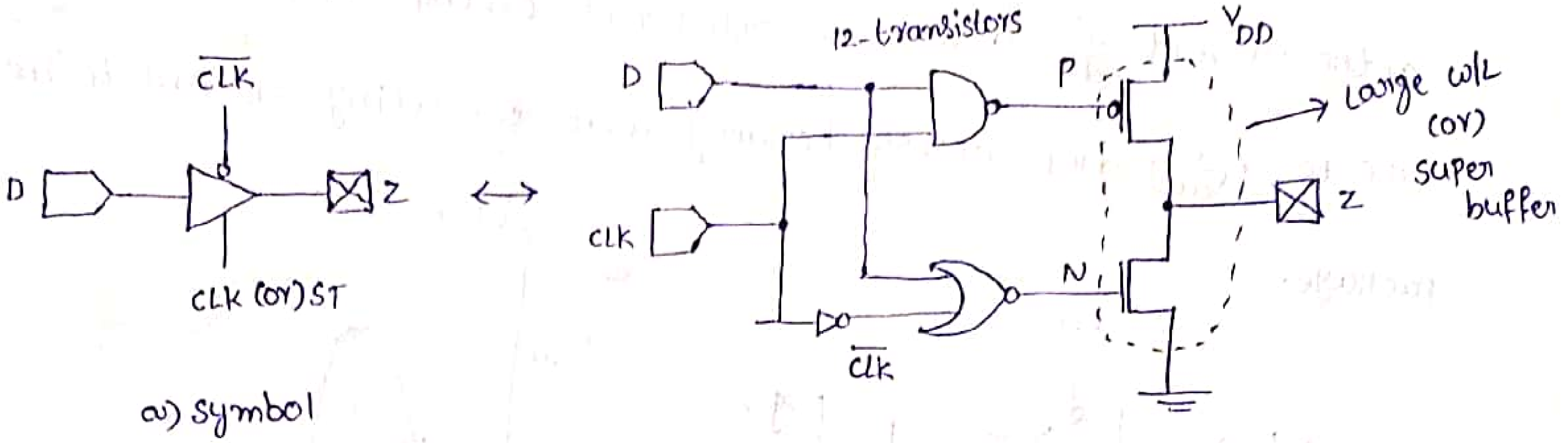
PL-NH → weak PMOS, Strong NMOS Process corner

* strong NMOS, PMOS → Low V_{tno}, Low $|V_{top}|$; high k_n, high k_p

* weak NMOS, PMOS → High V_{tno}, High $|V_{top}|$; Low k_n, Low k_p

Output circuits and $L\left(\frac{di}{dt}\right)$ Noise :-

Tristable output circuit as shown in fig. below



a) symbol

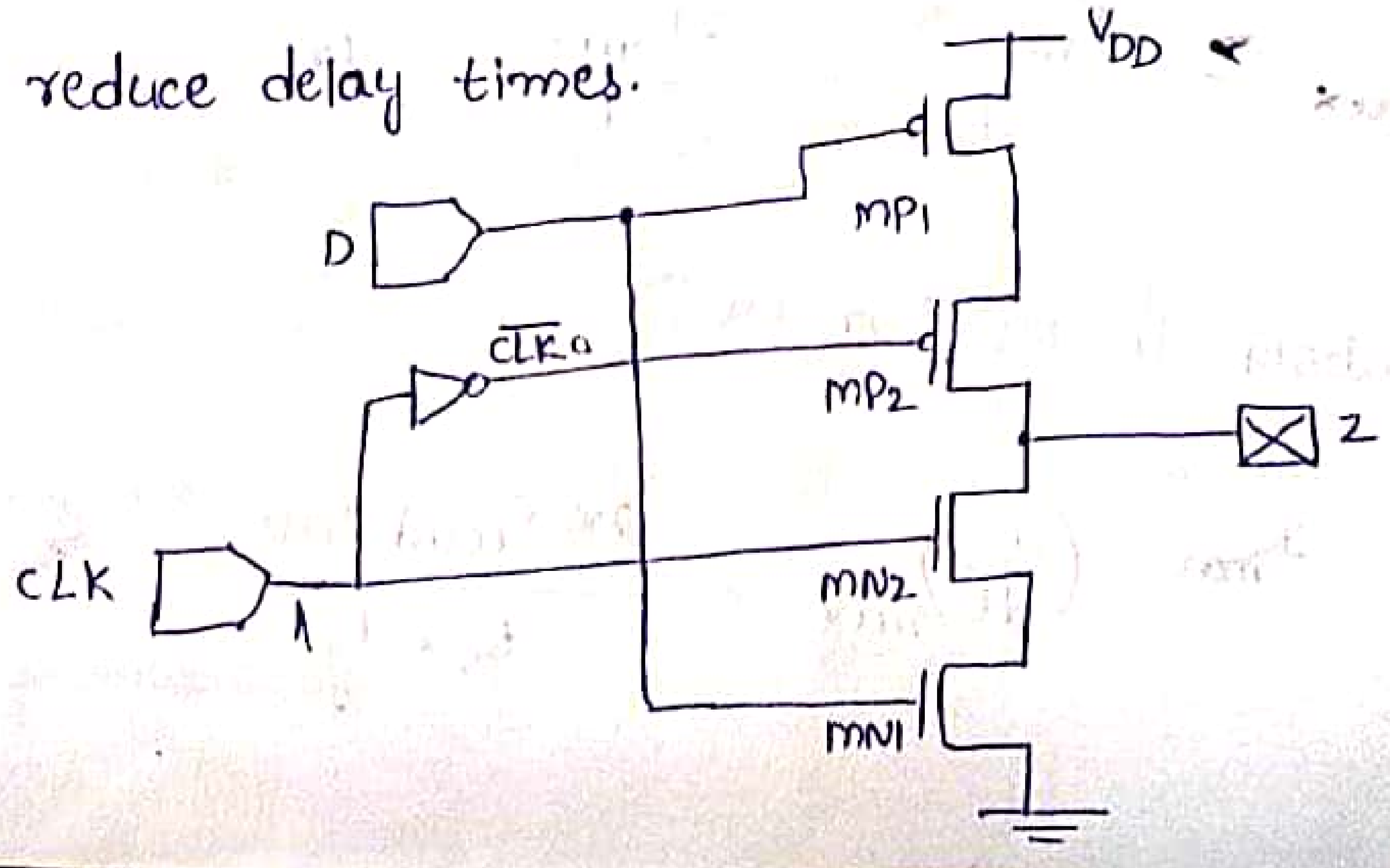
b) circuit

If CLK = 1 , Z = D

If CLK = 0 , Z = High Impedance

CLK	D	P	N	Z
1	1	0	0	1=D
1	0	1	1	0=D
0	X	1	0	High z

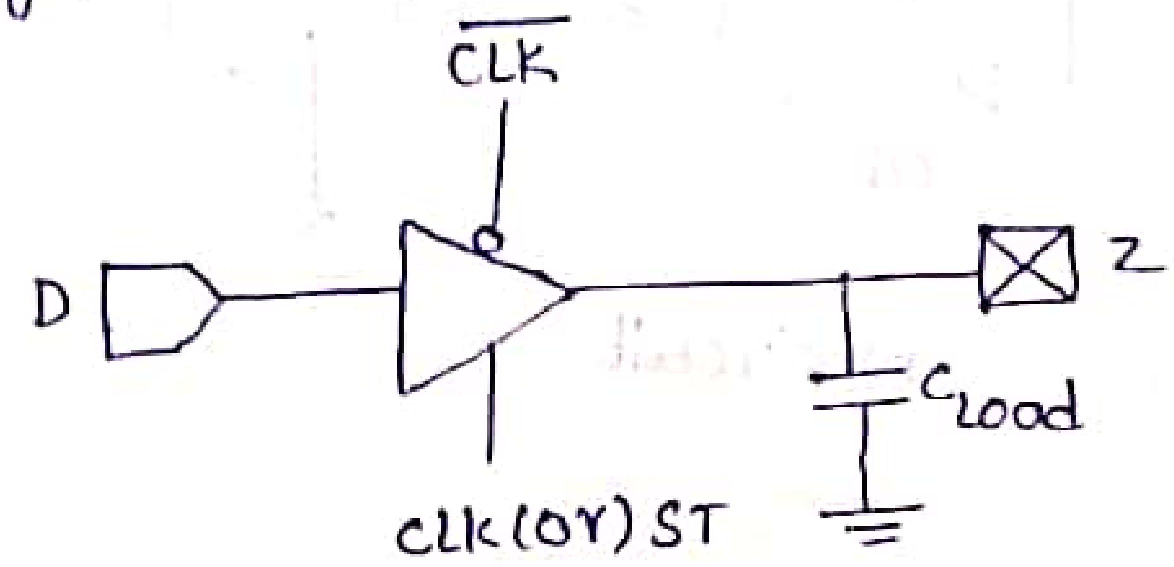
The Large w/L requires sufficient current sink (or) source and also reduce delay times.



If CLK = 0 → MN2, MP2 OFF ⇒ Z = High z

If CLK = 1 → MN2, MP2 ON ⇒ Z = D

unfortumatelly, such a requirement demands a high rate of change in the current $\frac{di}{dt}$ and cause significant on-chip Noise problems due to $L\left(\frac{di}{dt}\right)$ drop across bonding wire connecting o/p pad to the package.



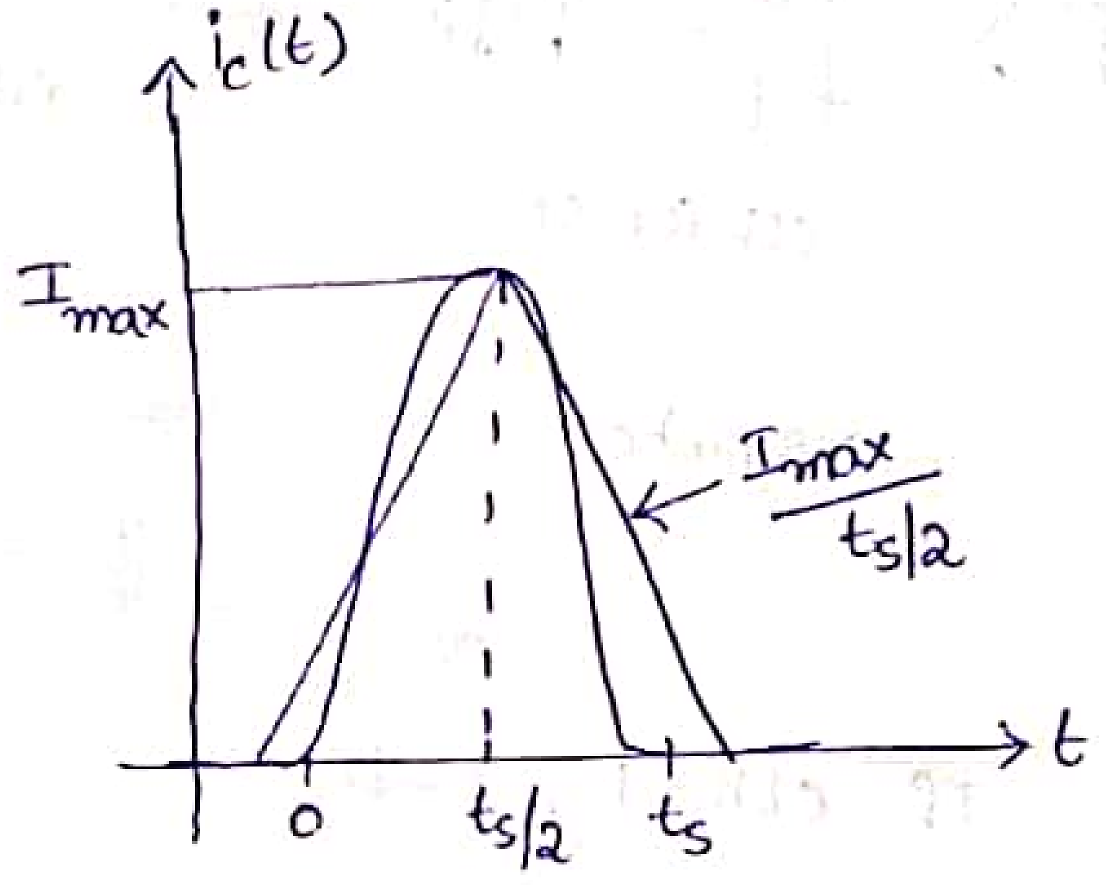


Fig:- o/p circuit current waveform during switching

During switching period,

$$I_{max} = C_{load}\left(\frac{dV_{out}}{dt}\right)_{max}$$

$$= \frac{C_{load}\, V_{DD}}{t_s/2} \quad -①$$

$$\left[\because V_o = \frac{1}{C}\int i(t)\,dt \qquad \frac{dV_o}{dt} = \frac{1}{C}\, i(t) \qquad \Rightarrow i(t) = C\frac{dV_o}{dt}\right]$$

$$\left(\frac{di}{dt}\right)_{max} \geq \frac{I_{max}}{t_s/2} = \frac{2I_{max}}{t_s} \quad -②$$

substitute eq ① in eq ②

$$\cancel{I_{max}}\left(\frac{di}{dt}\right)_{max} \geq \frac{2\times C_{load}\, V_{DD}}{t_s \times \frac{t_s}{2}} \geq \frac{4C_{load}\, V_{DD}}{(t_s)^2}$$

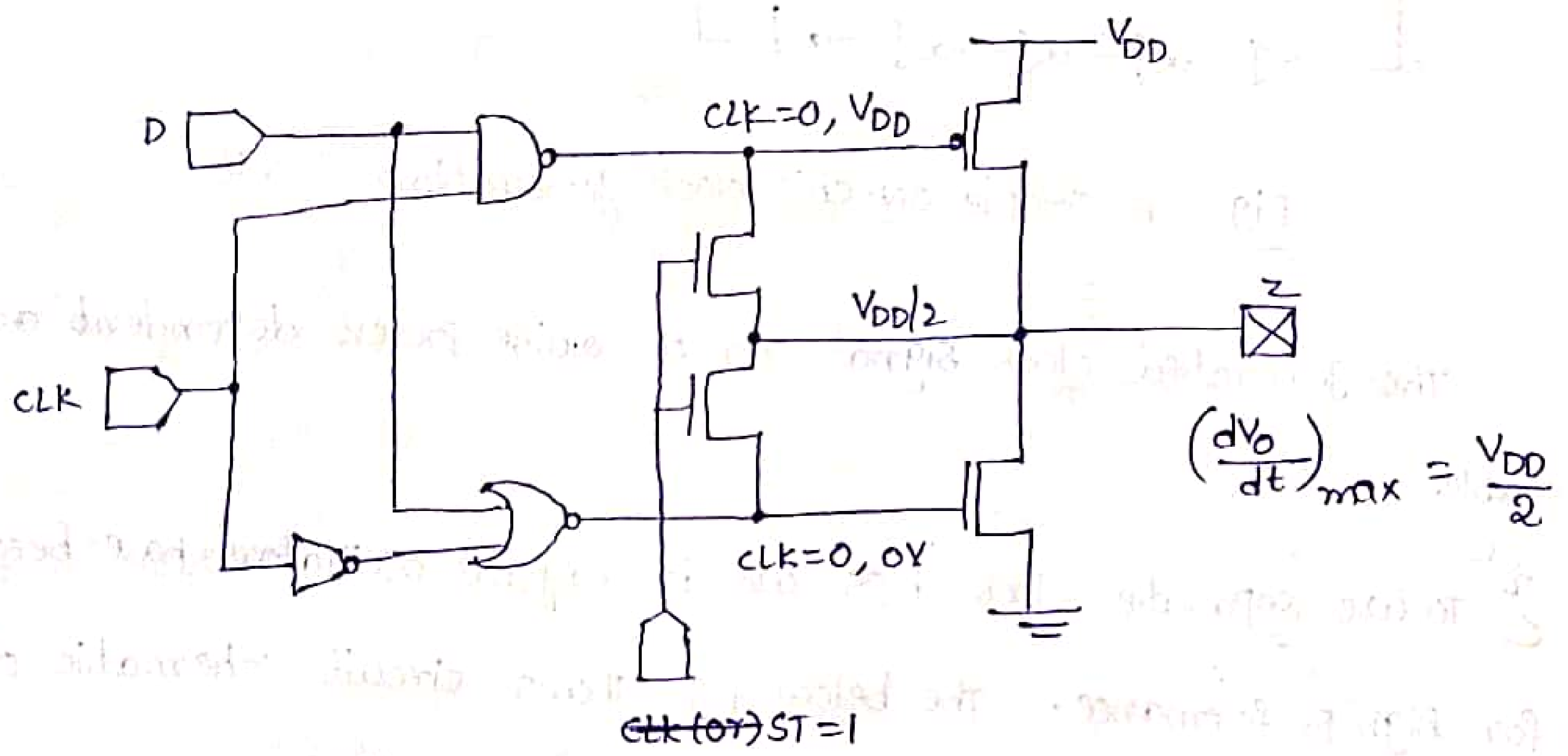
For Example,

$C_{Load} = 1PF$, $L = 1nH$, $V_{DD} = 3.5V$, $t_s = 1ns$

$$\therefore \left(\frac{di}{dt}\right)_{max} \geq \frac{4\times(1\times10^{-12})\times 3.5}{(1\times10^{-9})^2} = \frac{14mA}{ns}$$ ~~≥14mV~~

$$L\left[\frac{di}{dt}\right]_{max} \geq 14mV$$

The circuit for output reduces $L\frac{di}{dt}$ noise as shown in fig.



$$\left(\frac{dV_o}{dt}\right)_{max} = \frac{V_{DD}}{2}$$

when ST=1, CLK=0 → Inverter o/p z to $\frac{V_{DD}}{2}$

onchip clock generation and distribution :-

Clock signals are the heart beats of digital systems. Hence, the stability of clock signals is highly Important. Ideally, clock signals should have minimum rise time and falltimes specified by duty cycles and zero skew.

In reality clock signals have non zero skews and rise time & fall times; duty cycles can also vary.

A simple technique for on-chip generation of a primary clock signal would be use a ring oscillator as shown in fig.

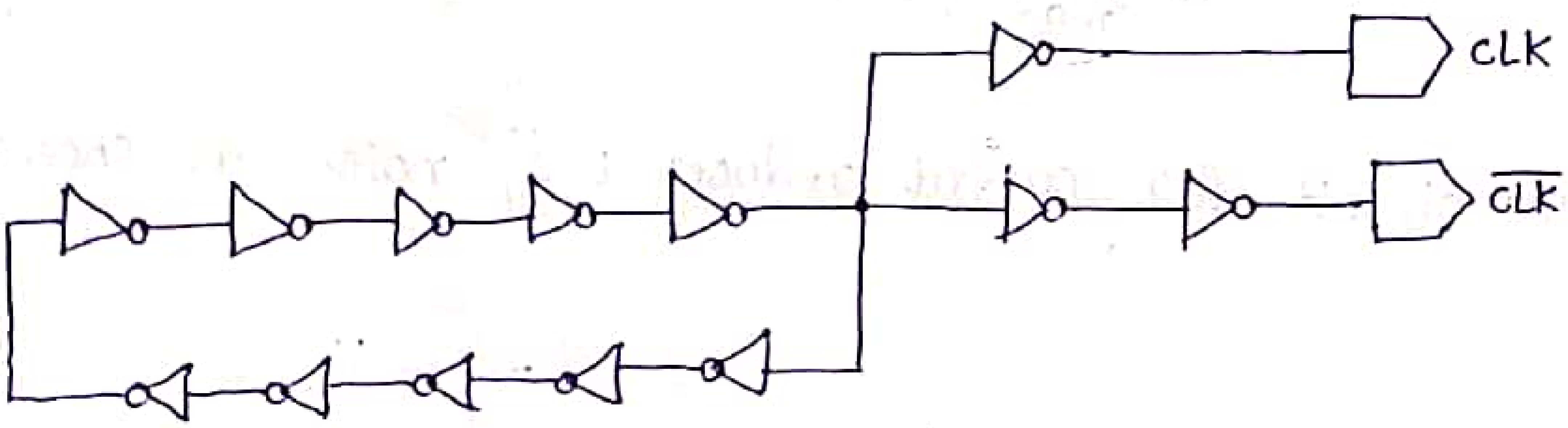


Fig:- A simple ON-chip clock generation circuit

The generated clock signal can be auite process dependent and unstable.

To use separate clock chips use in crystal oscillators have been used for high performance. The below fig. shows circuit schematic of Pierce crystal oscillator with good frequency stability.

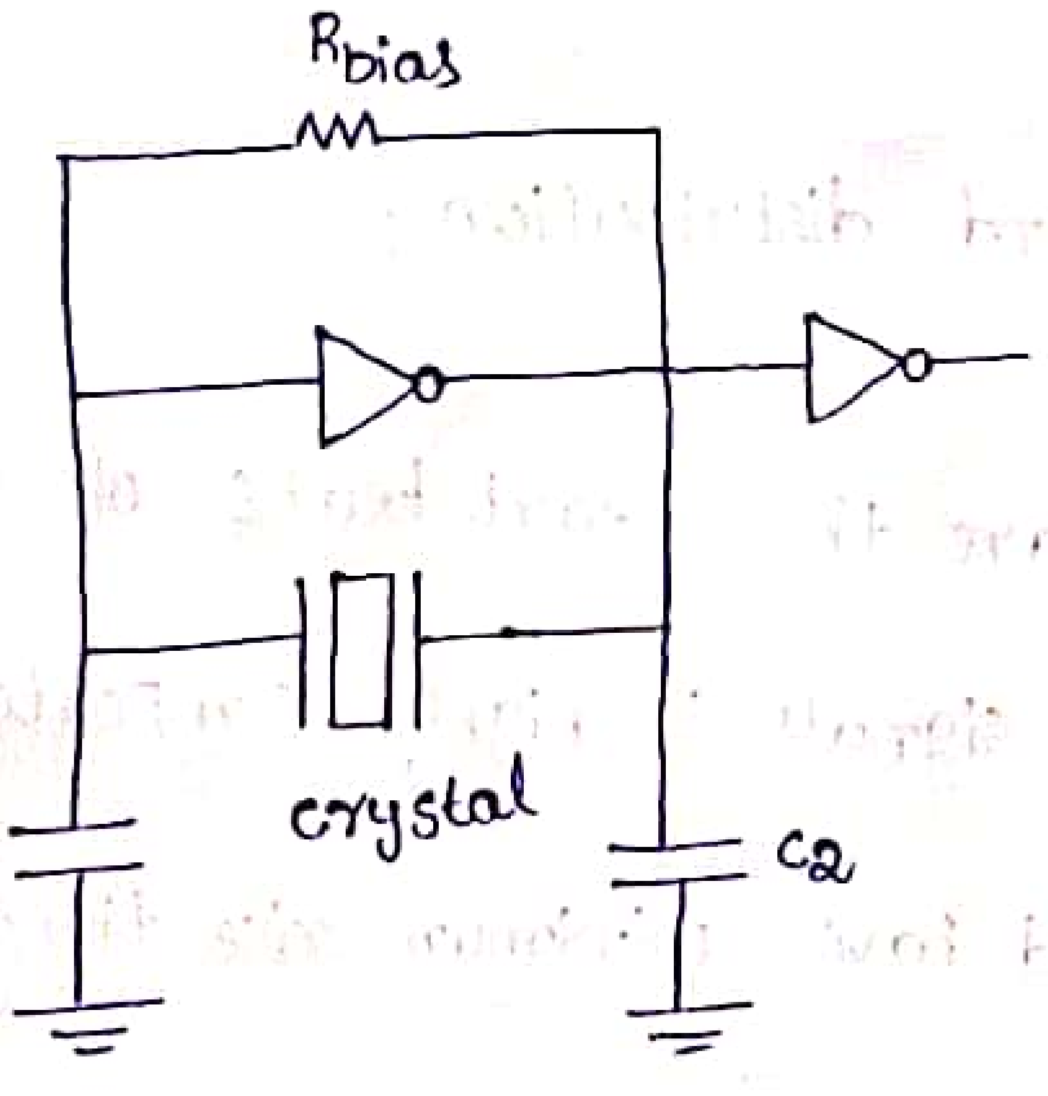
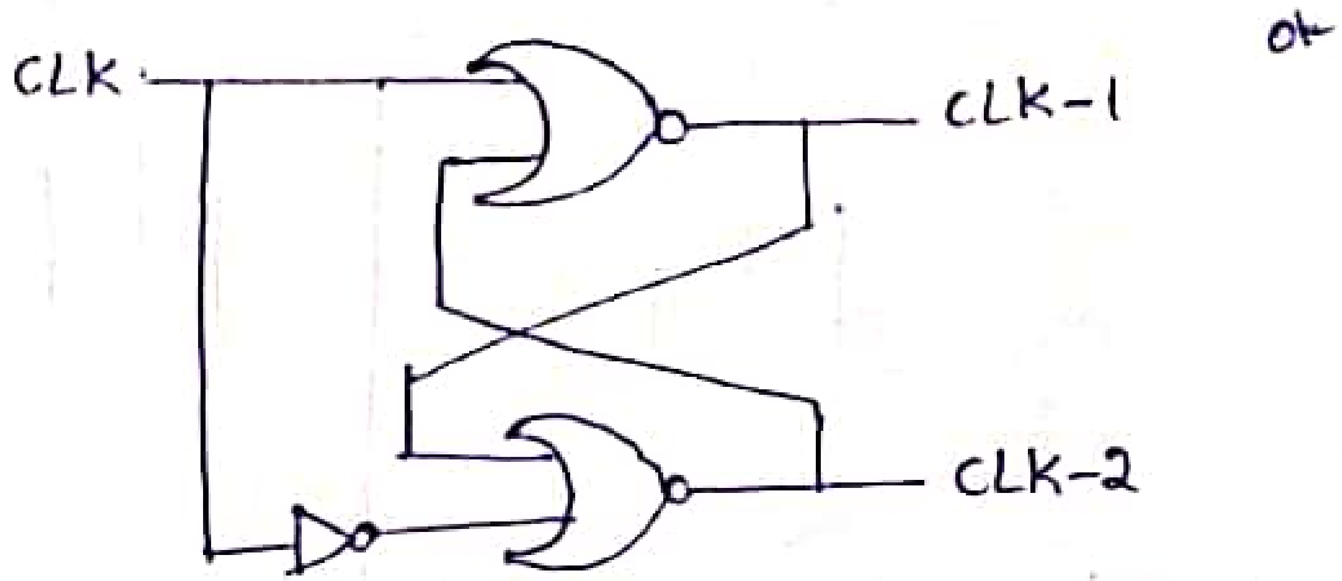


Fig:- Pierce crystal oscillator circuit

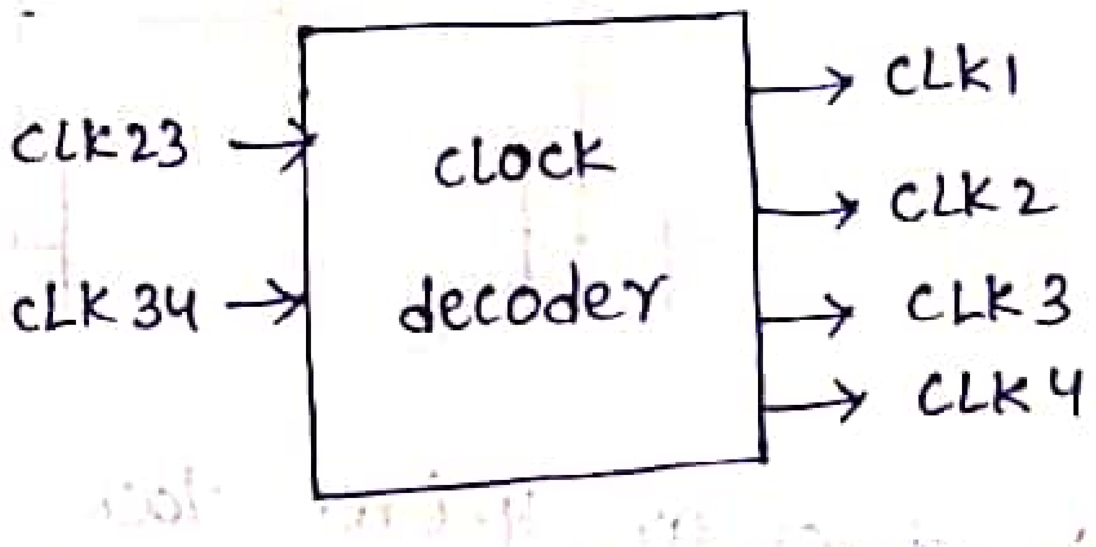
From above fig, crystal can be represented as series RLC circuit i.e higher series resistance, Lower oscillation frequency.

The Inverter across the crystal Provides necessary voltage differential and External Inverter Provides Amplification to drive clock Loads.

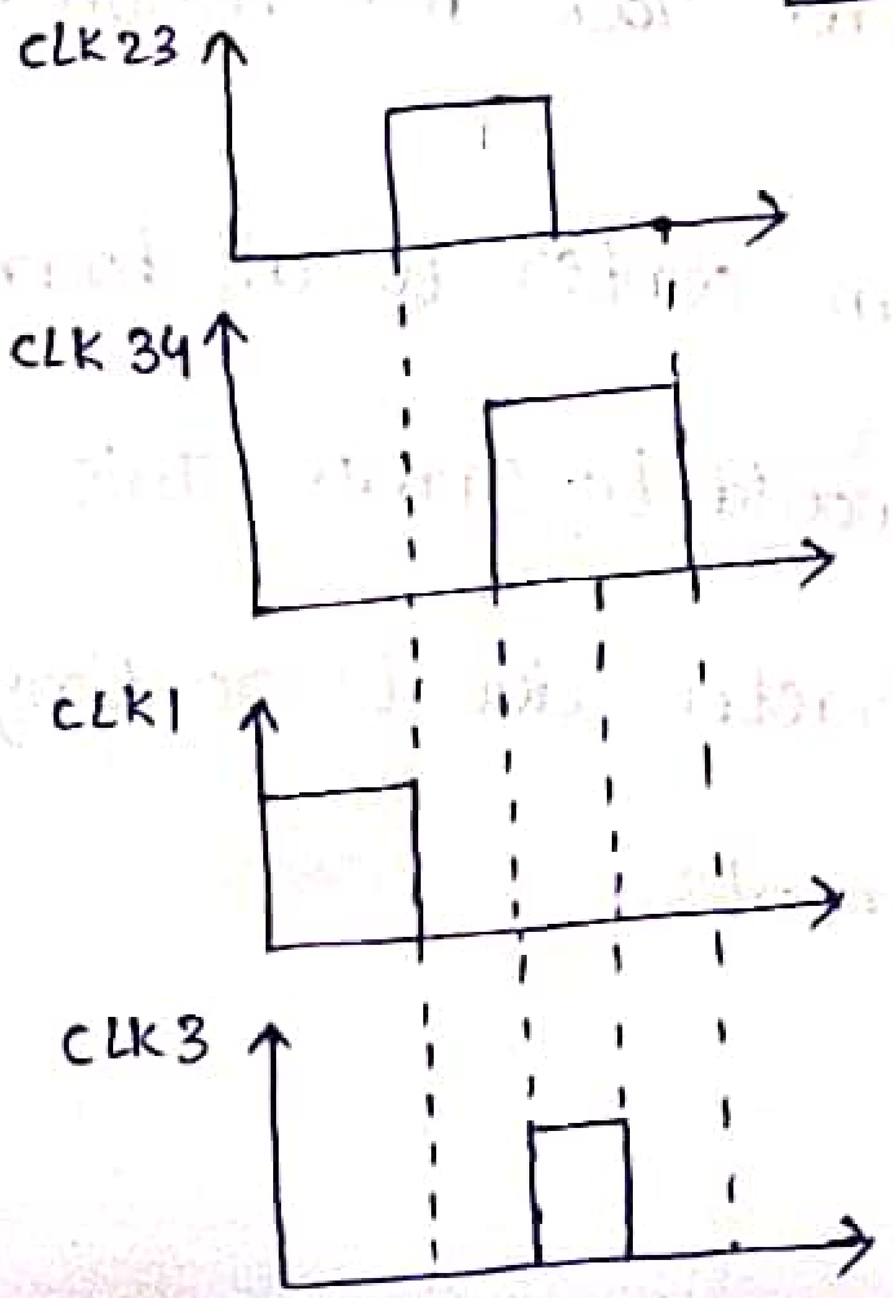
A simple circuit that generates CLK1, CLK2 from original clock signals as shown in fig.

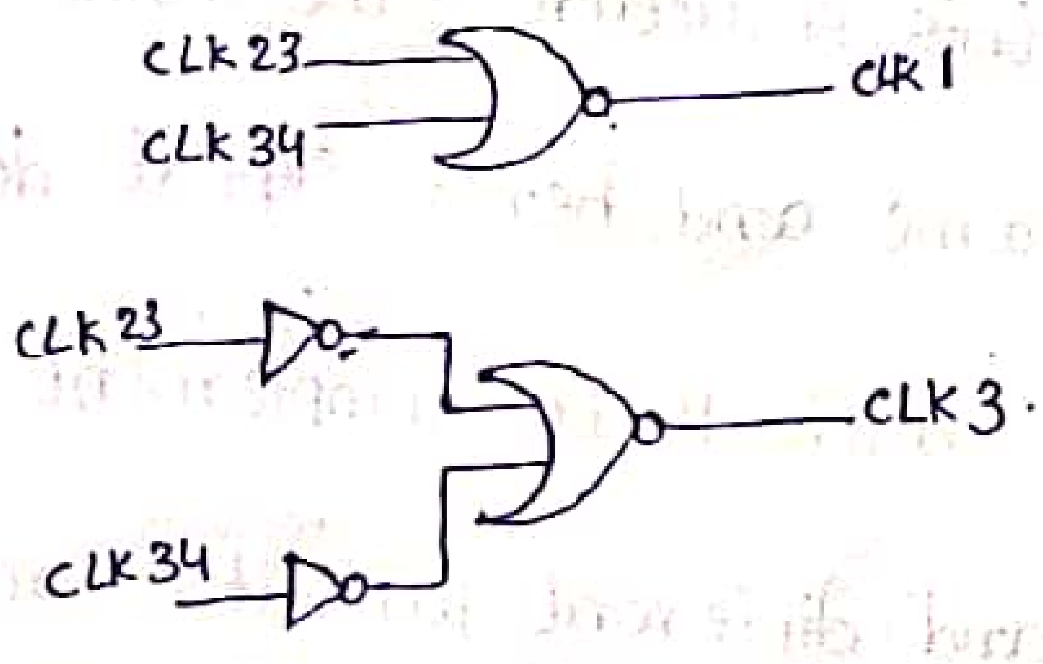


The clock decoder circuit that takes in the Primary clock signals and generates 4 phase signals as shown in: fig. below



(a): symbolic representation





b) waveforms and gate level Implementation

Since clock signals are required almost uniformly over the chip area, it is desirable that all clock signals are distributed with a uniform delay.

An ideal distribution Network would be H-tree structure as shown in fig. below.

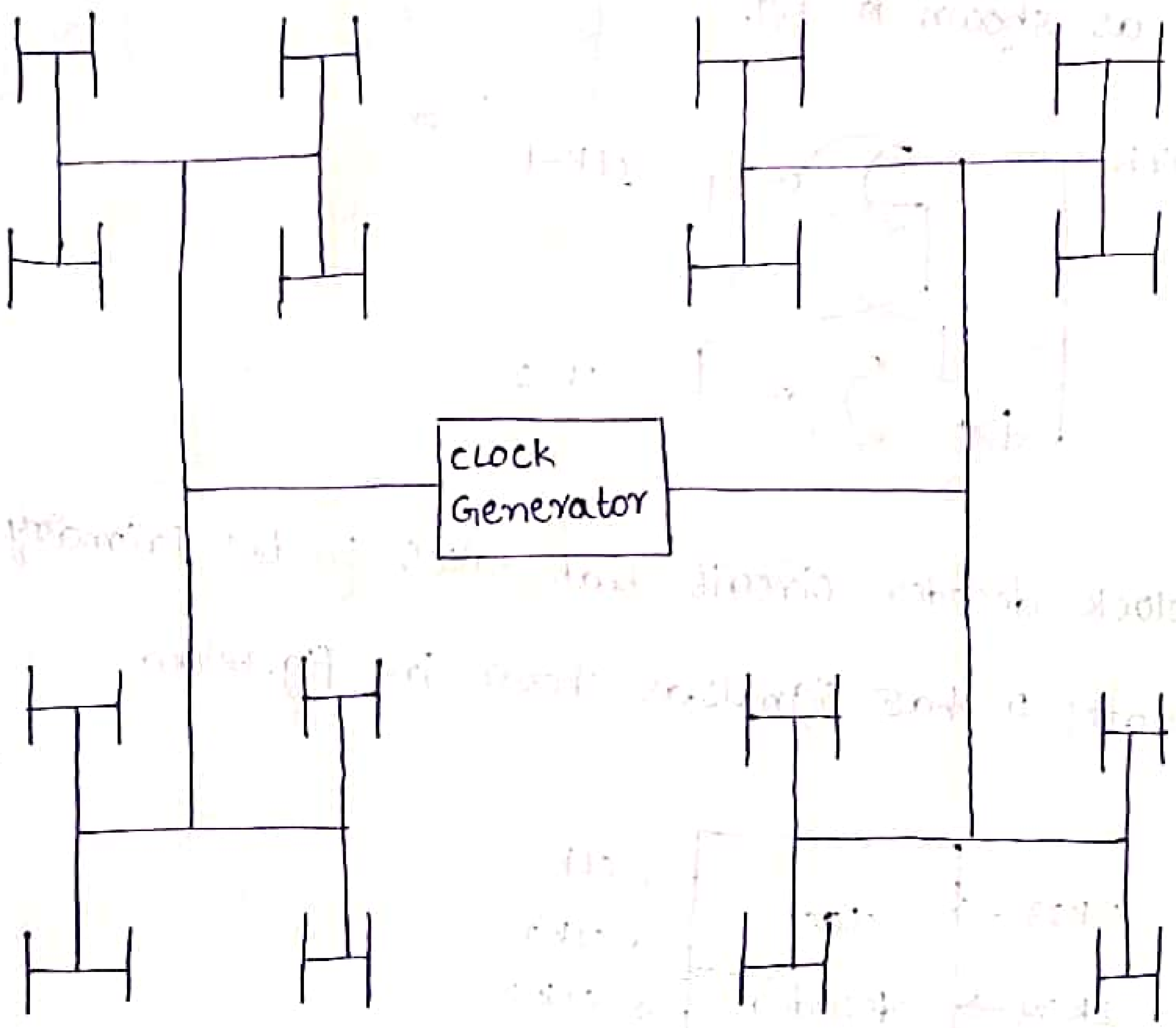


Fig:- General Layout of an H-tree clock distribution Network

From this structure, the distances from center to all branch points are same and hence signal delays would be same. This structure is difficult to Implement in practice due to routing constraints and different fanouts requirements.

Regardless of the exact geometry of clock distribution Network, the clock signals must be buffered in multiple stages as shown in fig. to handle high fanout loads.

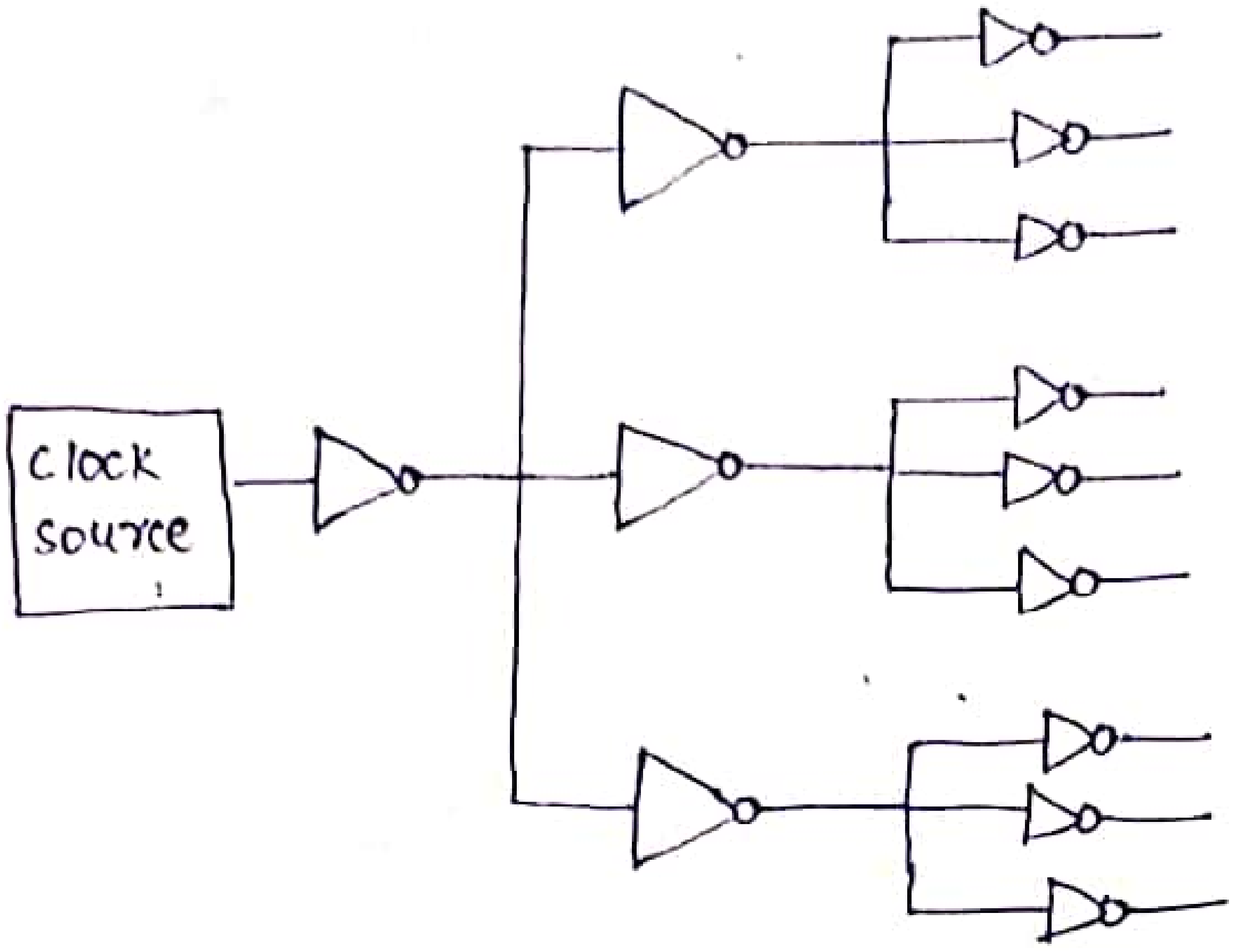


Fig :- Three-level buffered clock distribution Network

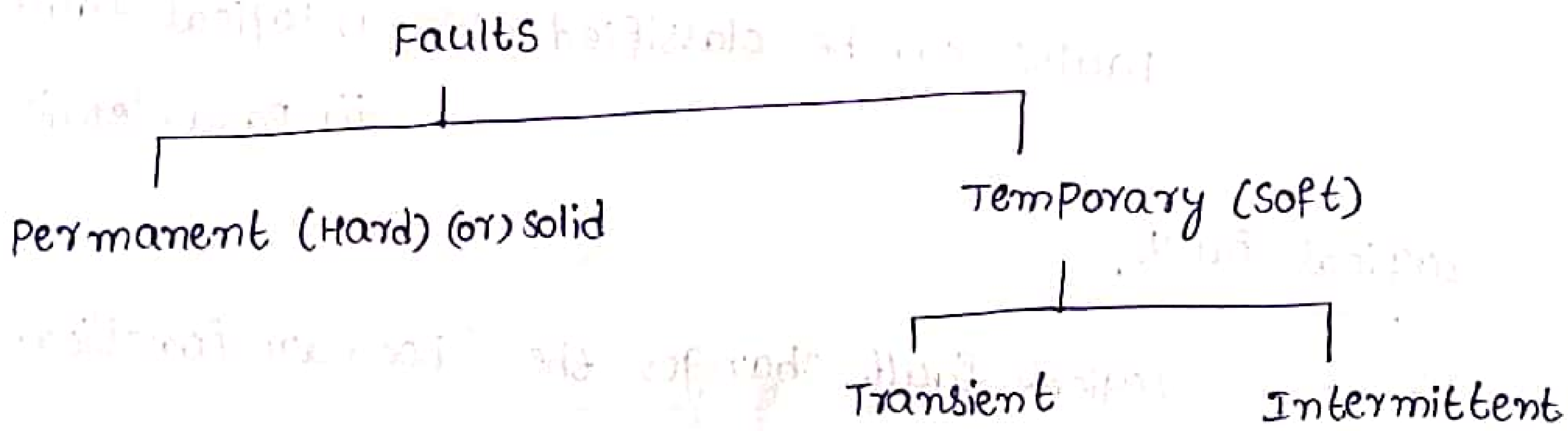
Design for testability

Fault :- It is defined as the representation of a defect reflecting a physical condition that causes a circuit to fail to perform in a required manner.

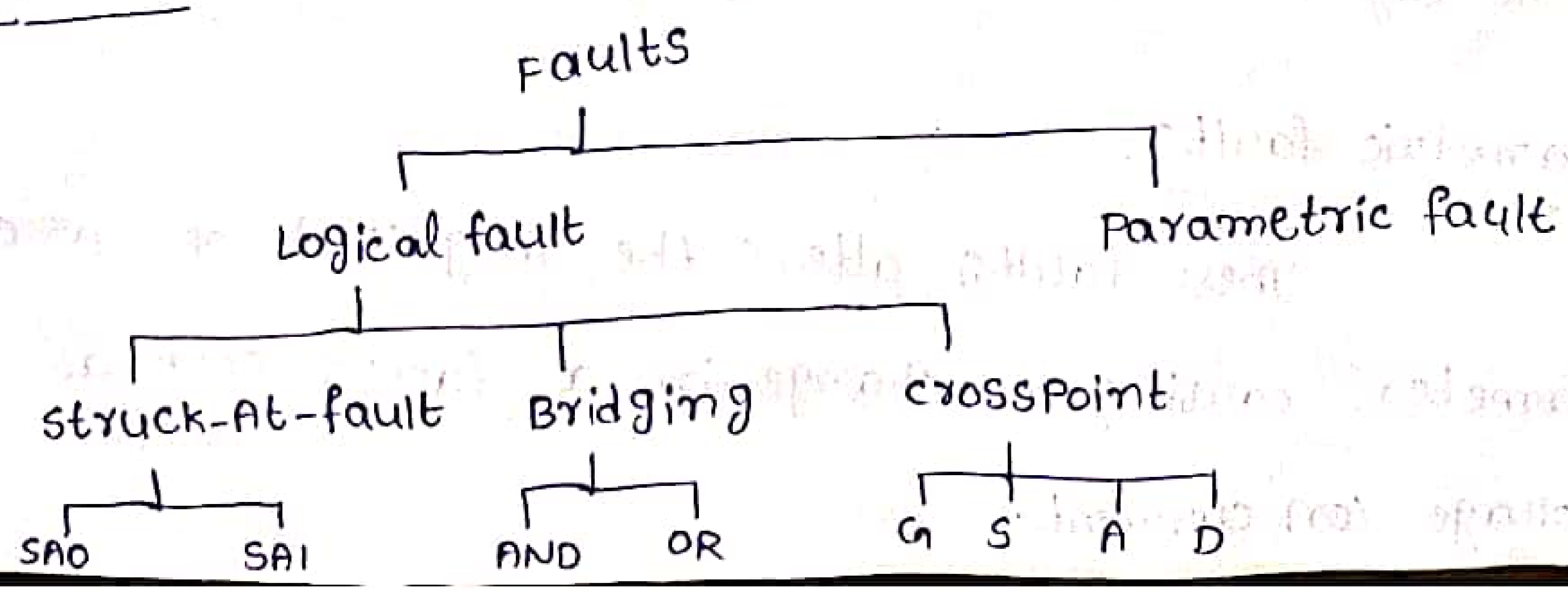
Failure :- A failure is a deviation in the performance of a circuit (or) system from its specified behaviour and represents irreversible state of component such that it must be repaired in order for it to provide its Intended design function.

Fault types and models :-

A Fault is a physical defect of one (or) more components (or) connections of the circuits.



Based on effect



Permanent faults :- These are caused by breaking (or) wearing out of a component.

Such faults are always present and donot appear, disappear, (or) change their nature during operation.

Temporary faults :- These faults occur during certain Interval of time. These faults can be either transient (or) Intermittent.

i) Transient fault :- These faults caused by some externally Induced signal perturbation, such as power supply fluctuations.

ii) Intermittent fault :- These faults occurs when a component is in the process of developing a permanent fault.

Depending on effect

Faults can be classified as
i) Logical fault
ii) Parametric fault

Logical fault :-

Logical fault changes the Boolean function realized by the digital circuits.

Parametric faults :-

These faults alters the magnitude of a circuit parameter, causing a change in a factor such as circuit speed, voltage (or) current.

Logical faults :- It contains 3types

i) stuck-At faults

ii) Bridging faults

iii) Log cross point faults

i) stuck-at-fault :- These faults occured If a signal line appears to have its value fixed at either a Logical 0 (or) Logical 1, irrespective of Input signals applied to the circuit.

when the signal line is always at Logical '1' the fault is known as stuck-at-one (SA1) and the signal line is at Logical 0 the fault is known as stuck-at-zero (SA0)

ii) Bridging faults :-

These faults occured if two signals lines are shorted together. The faults may be either AND-~~or~~ (or) an OR-type of Bridging fault.

iii) cross Point faults :-

These faults occur in Programmable logic arrays due to extra (or) missing devices (such as diodes (or) transistor) are called cross Point faults.

Based on effect, These are classified into Growth (G) faults, Shrinkage (S) faults, Appearance (A) faults, Disappearance (D) faults.

→ If a circuit has only one fault at any given time, is called. "single fault". If there are 2 (or) more faults in the circuit, the circuit is said to have "multiple faults".

Fault models :-

A model for how fault occur and their Impact on circuits is called fault model.

There are 2 fault models : i) stuck-at faults
ii) stuck-open (or) stuck-short faults

i) stuck-at faults :-

stuck-at zero (SA0), stuck-at one (SA1)

→ stuck-at fault model corresponds to real faults but it does not represent all possible faults.

→ The fault models can be deduced by using basic circuits such as AND, OR, Inverter and tri-state buffer.

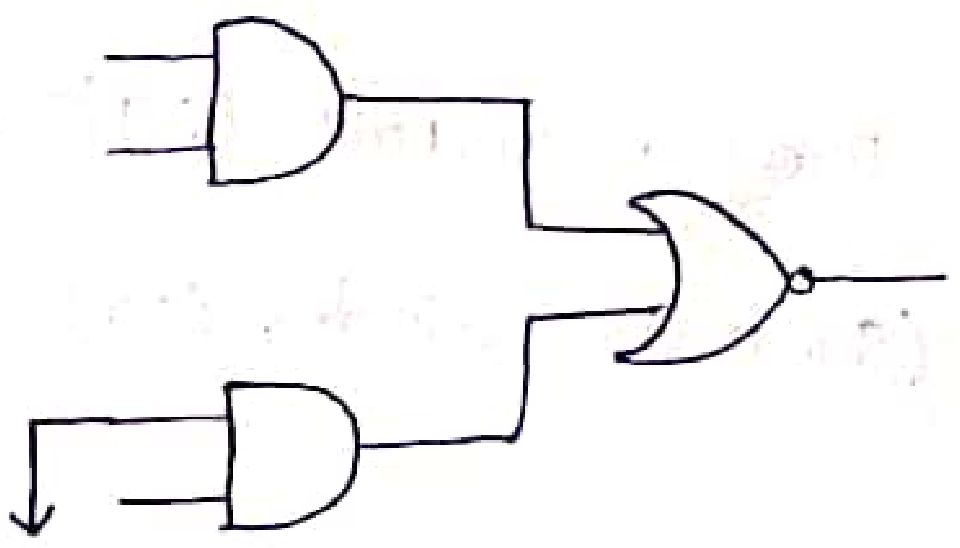


Fig :- Stuck-At-0 (SA0)

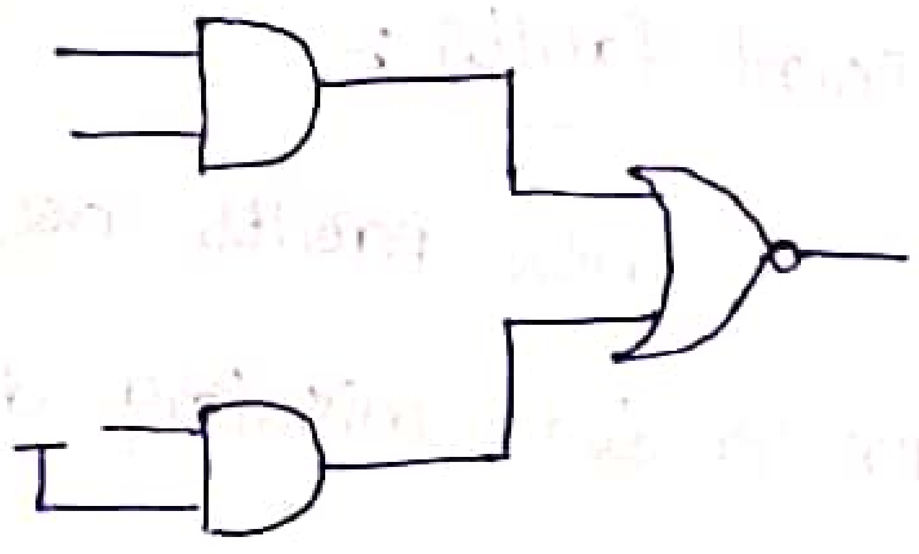


Fig :- Stuck-At-1 (SA1)

→ stuck-at- faults mostly occur due to shorting of gate oxide (NMOS gate to GND (or) PMOS gate to V_{DD}) (or) metal-to-metal shorts.

ii) stuck-open and stuck-short faults:-

These faults are usually referred as "transistor faults". physical faults which occurs at manufacturing level are called as defects.

The electrical (or) logic-level faults by physical defects are referred as defect oriented faults such as open links, Improper semiconductor doping, Bridging faults etc...

Ex:- consider CMOS NOR Logic

$Q_1, Q_2 \rightarrow$ PMOS transistors

$Q_3, Q_4 \rightarrow$ NMOS transistors

when I/p A & B are '0',

Q_1, Q_2 are shorted and Q_3, Q_4 are open.

Therefore, when A=B=0, o/p 'c' is connected to V_{DD}.

III^ly A=B=1, o/p 'c' is connected to ground i.e '0'.

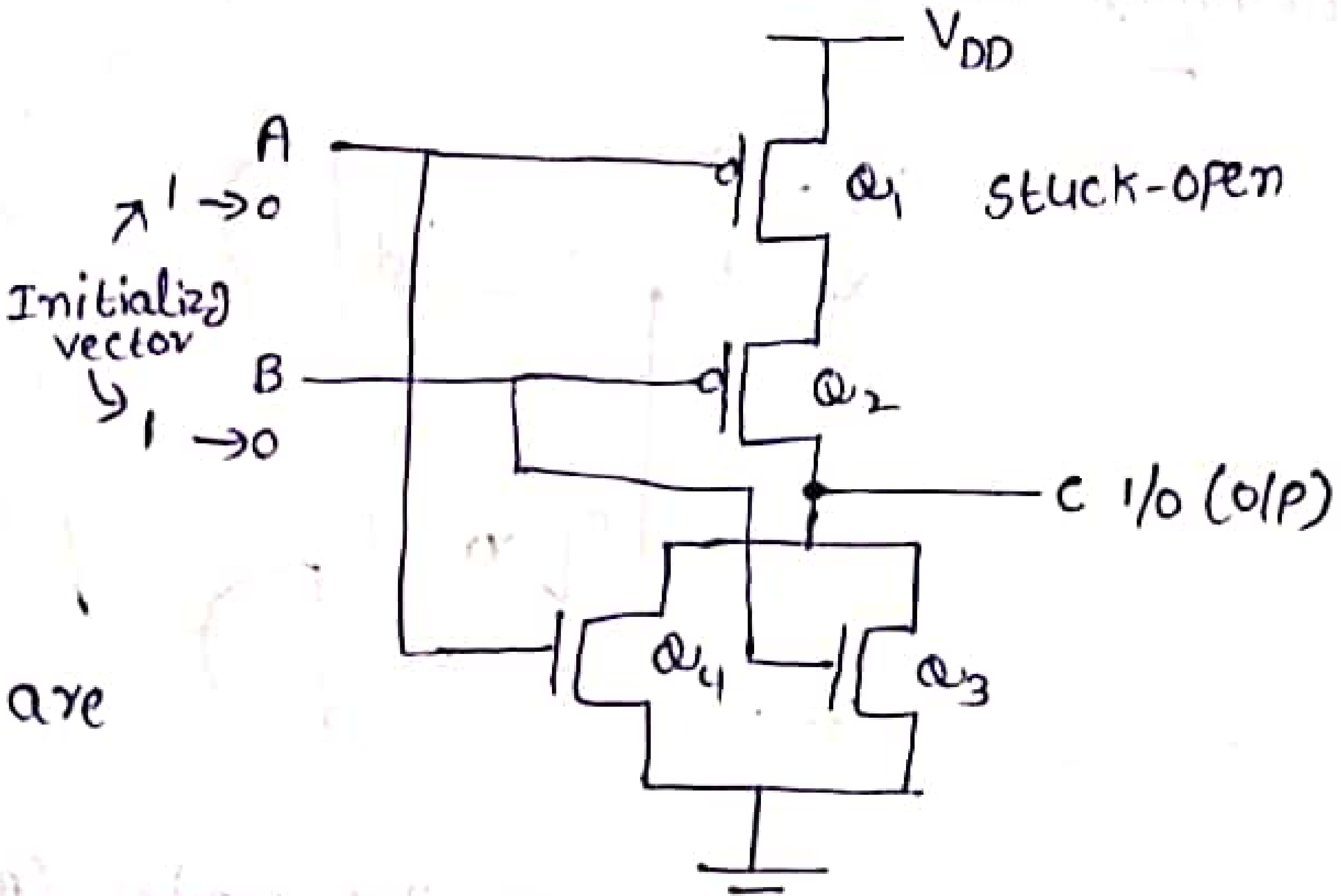


Fig:- CMOS NOR logic

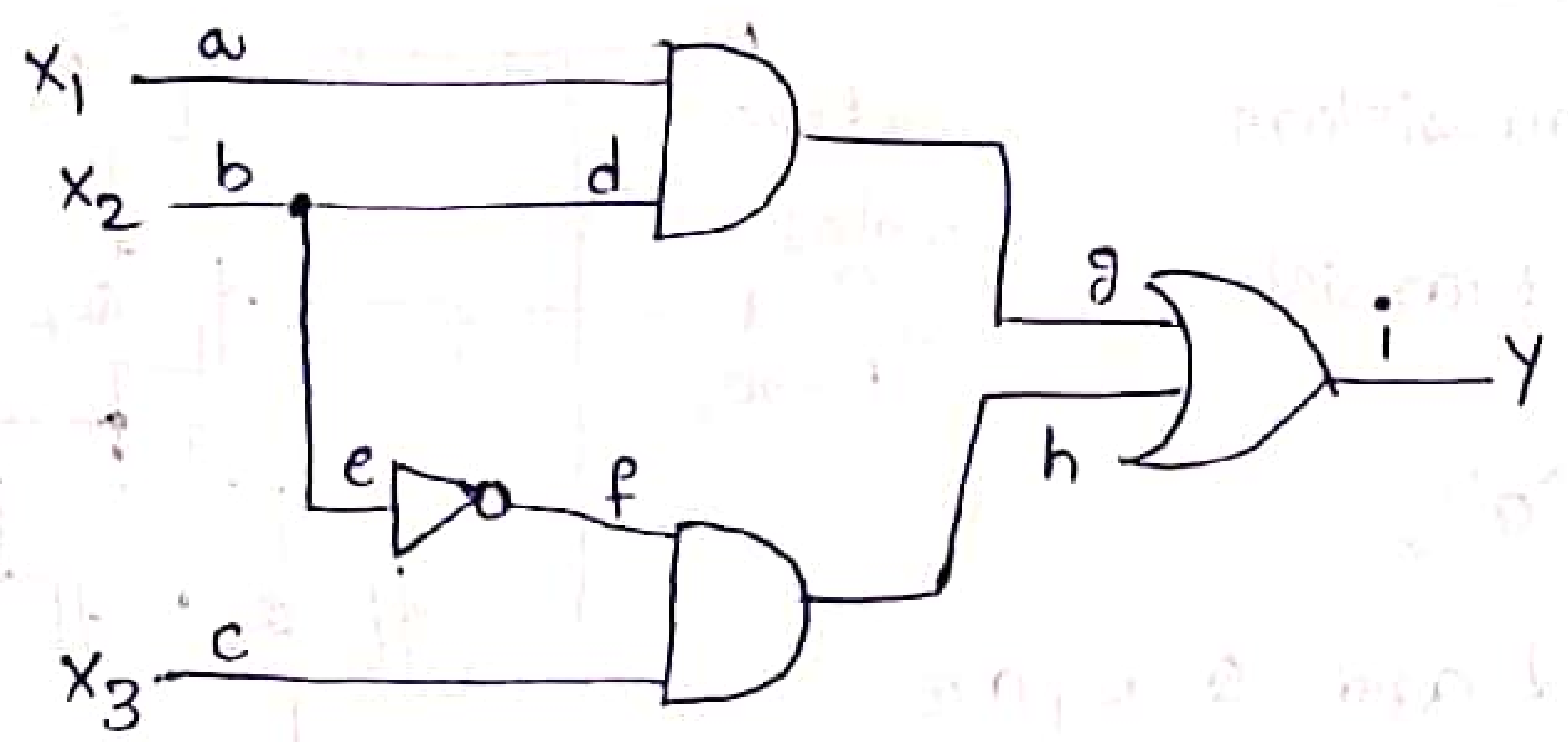
→ suppose fault Q_1 stuck-open. If A=B=0 then Q_1, Q_2 are shorted in fault free circuit but Q_2 is shorted in faulty Q_3, Q_4 are open. hence o/p 'c' is '1' in good circuit but is floating (Neither V_{DD} nor ground) in faulty circuit.

→ The good and faulty states of o/p 'c' are denoted by z & $\frac{1}{z}$.

→ The output node c has a parasitic capacitance. for detecting a fault, it should be ensured that the value of z is 0. It can be done by preceding A=B=0 as Initializing vector to A=1, B=0. This sets o/p node c to '0' in faulty circuit by discharging node capacitance to ground potential.

→ To complete the test another I/p from 10 to 00 is applied. It produces an o/p 0→1 in good circuit & 0→0 in faulty circuit.

Example of stuck at fault :-



It consists of 9 possible nodes (a,b,c,d,e,f,g,h,i).

Any of these nodes can be shorted to power supply (stuck at 1) (or) shorted to ground (stuck at 0). hence 9 nodes contribute 18 potential fault sites.

The truth table for fault free circuit (Y) and faulty circuits as given by

	$x_1 x_2 x_3$	000	001	010	011	100	101	110	111
Fault free o/p	Y	0	1	0	0	0	1	1	1
FAULTY OUTPUTS	a SA 0	0	1	0	0	0	1	0	0
	a SA1	0	1	1	1	0	1	1	1
	bSA0	0	1	0	1	0	1	0	1
	bSA1	0	0	0	0	1	1	1	1
	CSA0	0	0	0	0	0	0	1	1
	CSA1	1	1	0	0	1	1	1	1
	dSA0	0	1	0	0	0	1	0	0
	dSA1	0	1	0	0	1	1	1	1
	eSA0	0	1	0	1	0	1	1	1
	eSA1	0	0	0	0	0	0	1	1
	fSA0	0	0	0	0	0	0	1	1
	fSA1	0	1	0	1	0	1	1	1
	gSA0	0	1	0	0	0	1	0	0
	gSA1	1	1	1	1	1	1	1	1
	hSA0	0	0	0	0	0	0	1	1
	hSA1	1	1	1	1	1	1	1	1
	iSA0	0	0	0	0	0	0	0	0
	iSA1	1	1	1	1	1	1	1	1

Fault detection:-

The task of determining whether a fault is Present (or) not is called "fault detection" and the task of Isolating the fault is called "fault location".

Fault diagnosis :- The combined task of fault detection and fault location is referred as fault diagnosis.

Test vector (TV) :- The Input combination which in the presence of a fault Produces an output different from fault free output is known as Test vector.

Test set :- The set of test vectors used for testing the circuit is called test set.

Fault Simulation :- It is the Process of verifying test set.

Fault coverage :- The test set refers to the % of faults that can be detected by the test set.

From truth table,

consider node is stuck at 0, i.e connect to ground.

For Input values 110 and node a is stuck at zero produces output 'o' but the fault free circuit as o/p '1'. hence we observe that. fault is present it should be discarded;

The line dSA1, eSA0, fSA1 all other faults detect with 2 (or) more test vectors. Test vectors 011, 100 must be

Included in any set of test vectors that will obtain 100%. fault coverage.

These 2 test vectors detect total of 10 faults, remaining 8 faults can be detected with test vectors 001 & 110 ; therefore, this set of 4 test vectors obtain 100% single stuck-at fault coverage for this circuit.

controllability and observability (Design strategies for testing) :-

controllability :- It is defined as the capability of a node being driven to 1 (or) 0 through a circuit Inputs. If this node can be driven faithfully to 1 and 0, it is regarded as controllable.

→ controllability is Important while accessing the degree of difficulty of testing a Particular signal in a circuit.

→ A node with little controllability may take hundreds of cycles to get it to right state.

observability :- It is defined as the capability of logic state of this node being observed at the circuit's outputs.

→ observability is a degree to which it can be observed that node at output operates correctly.

→ Higher observability Indicates less no. of cycles required to measure output node value.

→ Poor observability Indicates sequential circuits with long feedback loops.

Design for testability :- (DFT)

Testability is based on 2 concepts i.e controllability and observability.

Some nodes in sequential circuits can be very difficult to control and observe. for example, most significant bit of an n-bit counter can only obsened after 2^{n-1} clock cycles.

It contains 3types

1. Ad-hoc testing
2. scan based (or) Level-sensitive scan design (LSSD)
3. Built-in self-test (BIST)

1. AD-hoc DFT techniques :-

Ad hoc testing combines a collection of tricks and techniques that can be used to Increase the observability and controllability of a design and that are generally applied in an application-dependent fashion.

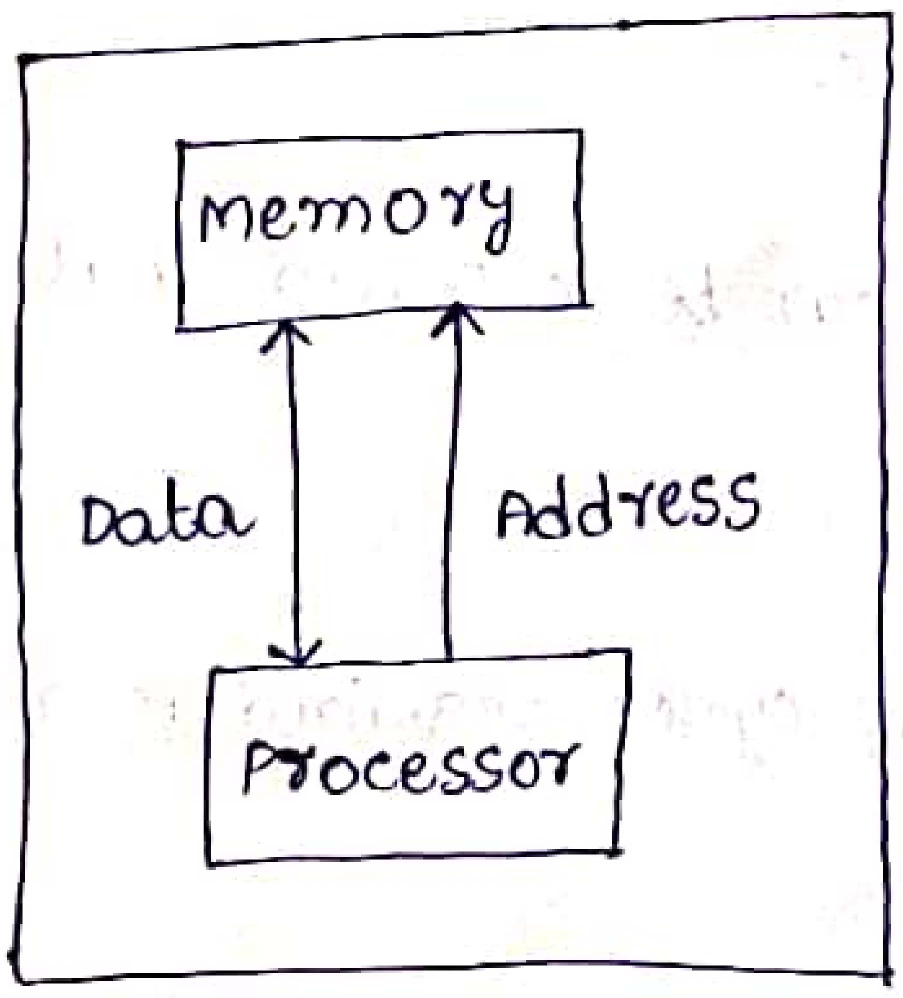


Fig (a) : Design with low testability

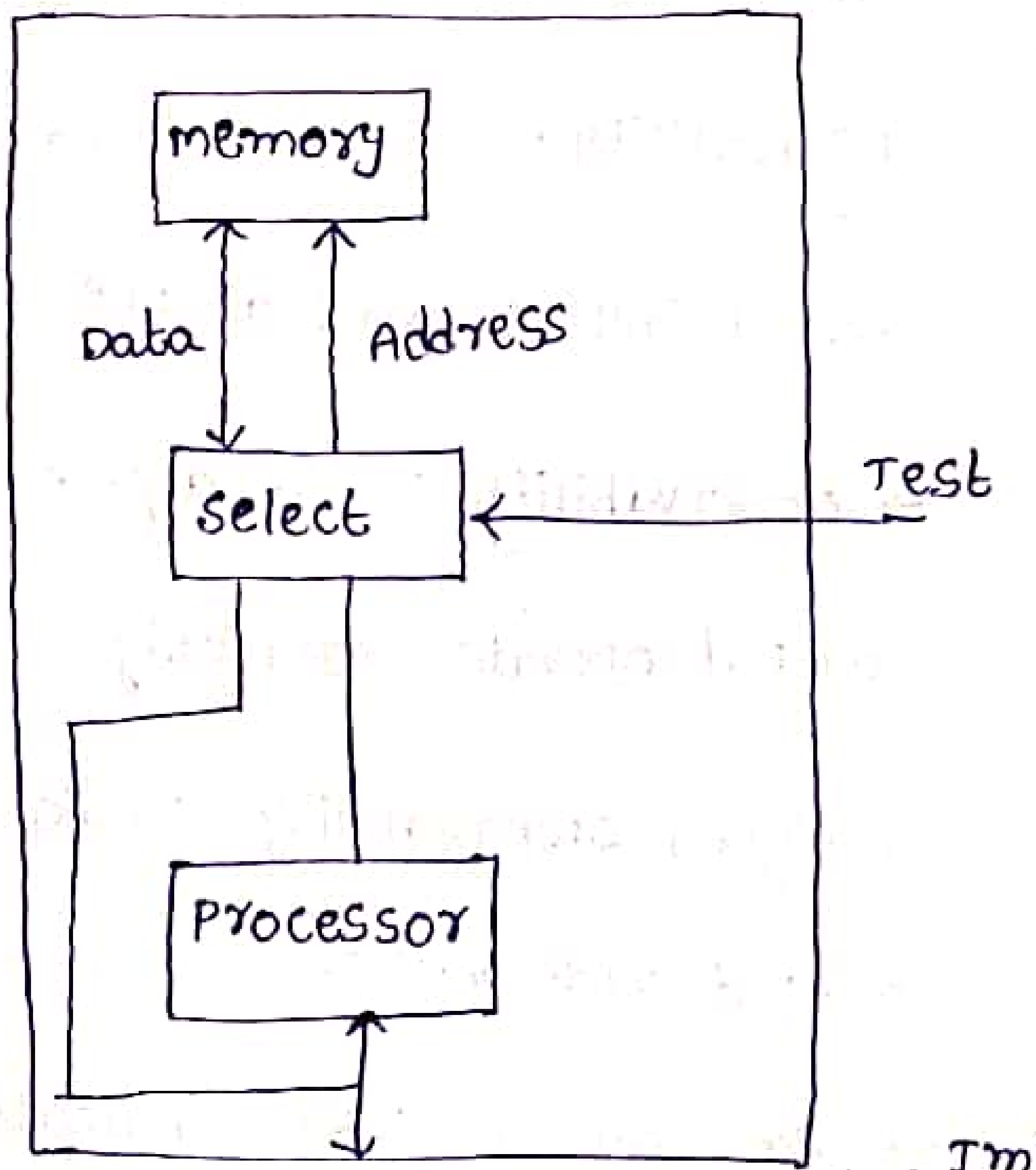


Fig (b) : Adding a multiplexer (selector) Improve testabi

From fig (a), represents simple processor with its data memory. under normal configuration, the memory is only accessible through the processor.

writing and reading a data value into and out of a single memory position requires a no. of clock cycles.

The controllability and observability of the memory can be Improved by adding multiplexers on the data and address buses as shown in fig (b).

During normal mode, these selectors direct the memory ports to the processor. During test mode, the data and address ports are connected directly to I/o pins, and testing memory can proceed more efficiently.

To reduce the extra test pins, one can multiplex test signals and functional signals on same pads. For example, the I/o bus in fig (b) serves as a data bus during normal operation and provides and collects the test patterns during testing.

An extensive collection of. ad-hoc testing approaches has been devised. Examples include the partioning of large state machines, addition of extra test points, provision of reset states, and Introduction of test buses.

most of these techniques depends on Application and architecture.

2. Boundary scan testing :-

Boundary scan as defined by the IEEE std.-1149.1 standard, is an Integrated method for testing Interconnects on Printed circuit boards (PCBs) that are Implemented at Integrated circuit (IC) level.

Architecture:-

The boundary-scan test architecture provides a test Interconnects between Integrated circuits on a board with out using physical test Probes.

cells on device Primary Inputs are referred as "Input cells", cells on primary outputs are referred as "output cells". The Input and output is relative to the Internal Logic of the device.

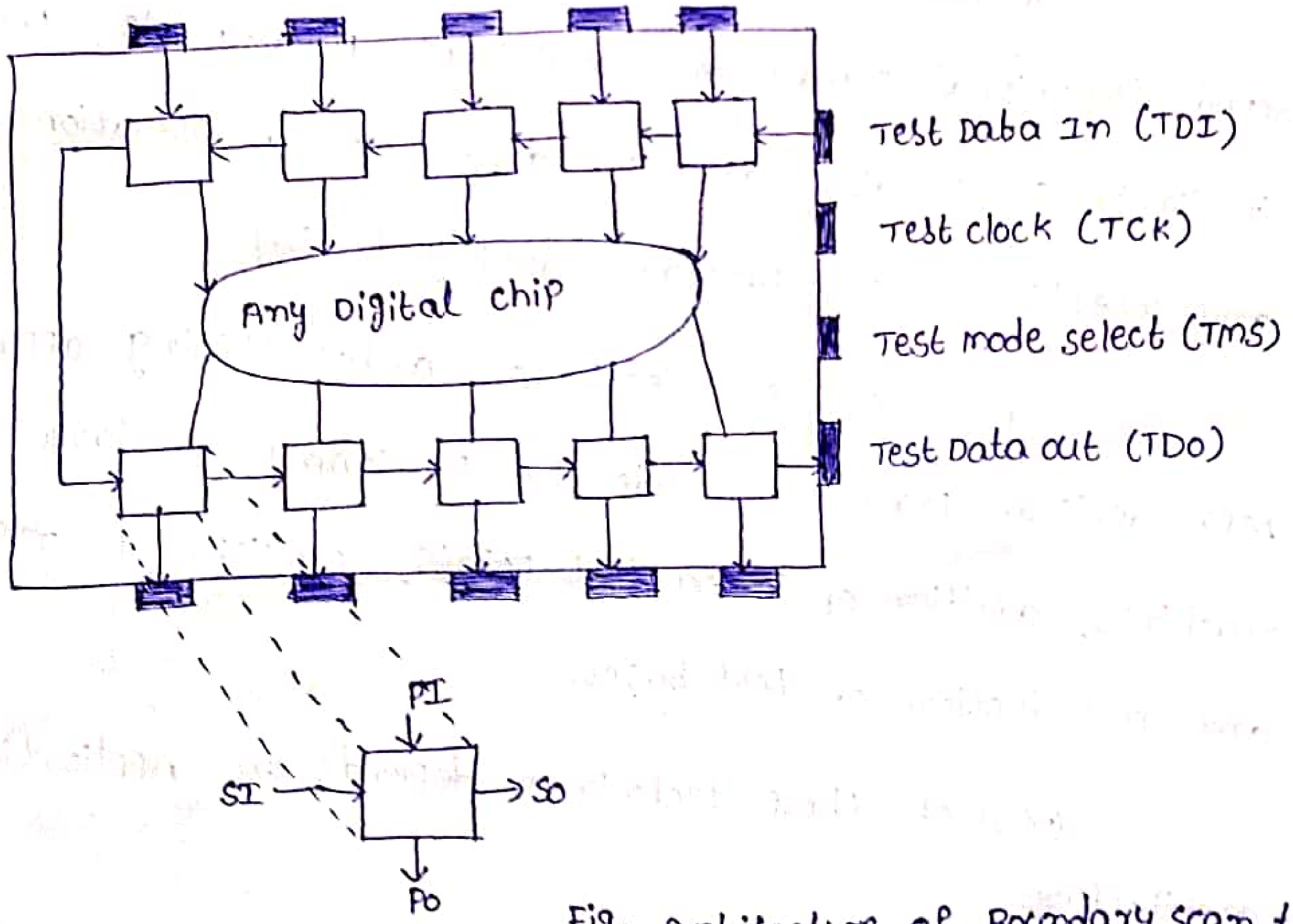


Fig:- Architecture of Boundary scan test

The collection of boundary-scan cells is configured into a Parallel-in, Parallel-out shift register.

A Parallel load operation called a capture operation causes the signal values on device Input Pins to be loaded into Input cells and signal values passing from Internal Logic to device output Pins to be loaded into output cells.

A Parallel unload operation called an update operation causes signal values already present in the output scan cells to be Passed out through the device output Pins.

Signal values already present in Input scan cells will be Passed into the Internal Logic. data can also be shifted around the shift register from Test data in (TDI) and terminating at device output Pin is called Test data OUT (TDO).

The test clock, TCK, is fed in via yet another dedicated device Input Pin and various modes of operation are controlled by a dedicated Test mode select (TMS) serial control signal.

The boundary scan elements contribute nothing the functionality of the Internal Logic. The boundary scan Path is Independent of the function of the device. The value of scan Path is at board level as shown in fig. below.

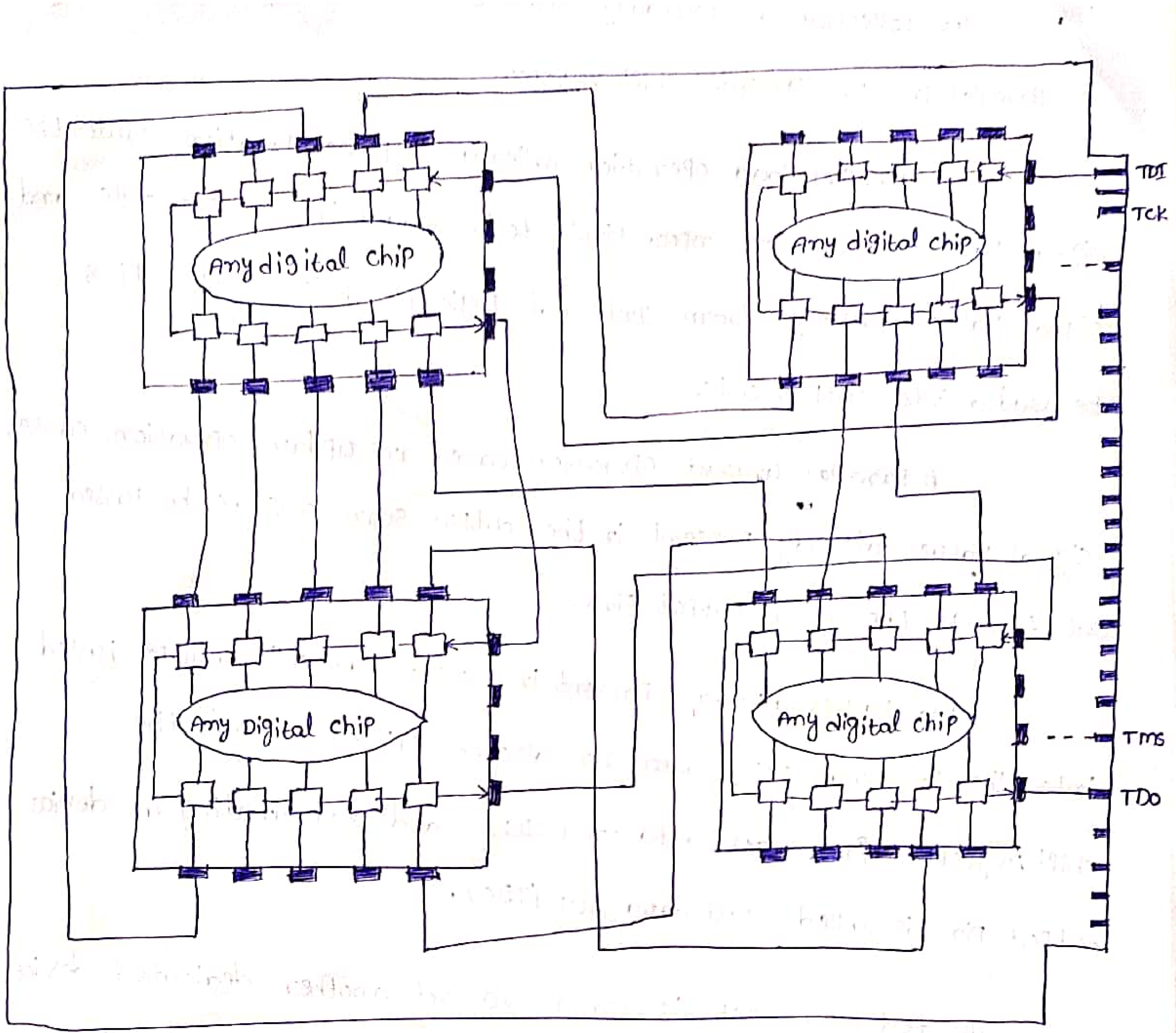


Fig:- Boundary scan test (Board level)

From above figure, 4 Boundary scan devices. There is an edge- ~~contr~~ connector Input called TDI connected to the TDI of 1st device. TDO from the first device connected to TDI of 2nd sdevice, and so on, creating a global scan path terminating at edge connector output called TDO. TCK is connected in Parallel to each device. TCK Input. TMS is connected in Parallel to each device TMS Input.

- captured data is serially shifted out and externally compared to expected results. Forced test data is serially shifted into Boundary scan cells. All of this is controlled from a serial data Path called "scan Path" (or) "scan chain".

Each scan has 4 modes of operation :

1. Normal mode : Data_In is Passed straight though to Data_out
2. update mode : content of the update hold cell is Passed through Data_out.
3. capture mode : Data_In signal is routed to the Input capture scan cell and the value is captured by next clock DR. clock DR is derivative of TCK
4. Shift mode : scan_out of one capture scan cell is Passed to scan_In of the next capture scan cell via a hard-wired Path.

Advantages of Boundary scan :-

1. Shorter test time
2. High test coverage
3. Increased Diagnostic capability
4. Lower captial equipment cost

3. Built-In Self test (BIST) :-

BIST is the technique of designing additional hardware and software features into Integrated circuits to allow them to perform self-testing i.e. testing of their own operation using own circuits, there by reducing dependence on an external automated test equipment (ATE).

BIST is a design-for-testability (DFT) technique, because it makes the electrical testing of a chip eaiser, faster, more efficient and less costly. This concept is applicable to any kind of circuit.

The general format of a Buit-In self test as shown in fig.

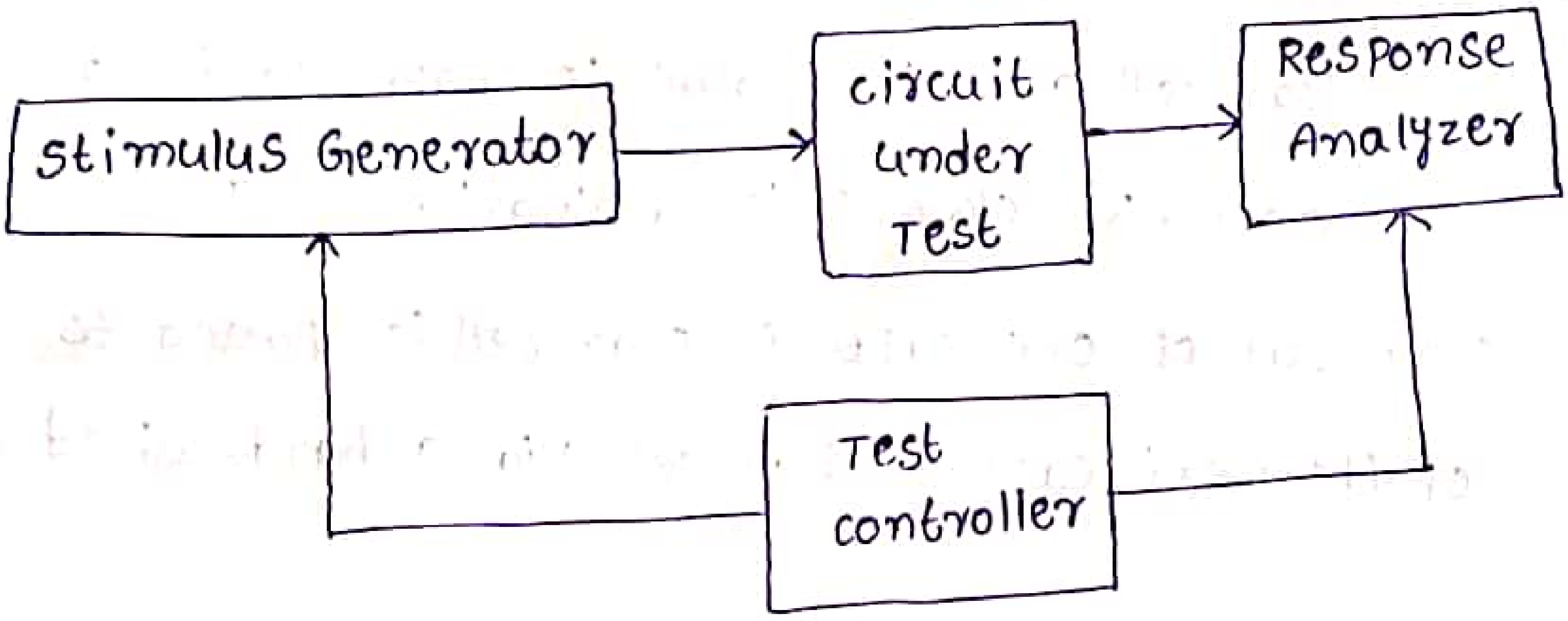


Fig :- General format of a BIST

supplying Test patterns to the device under test and means of comparing the device's response to a known correct sequence.

Stimulus Generator :- It is also called Test Pattern generator.

There are many ways for generating stimuli. most widely used are

1. Exhaustive approach
2. Random approach

1. Exhaustive approach:-

In exhaustive approach the test length is 2^n, where 'n' is no. of Inputs. In this method all detectable faults will be detected.

Ex:- N-bit counter

2. Random approach:-

In this method, random testing that Implies the application of a randomly choosen subset of 2^n possible Input patterns This subset should be selected so that a reasonable fault coverage is obtained.

Ex:- Pseudo random pattern generator is the Linear feed back shift register (or LFSR).

Response Analyzer:-

The response analyzer compares the generated response with the expected response stored in an on-chip memory, but this approach represents too much area over head to be practical. A cheaper technique is to compress the responses before comparing them.

Storing the compressed response of the correct circuit requires only a minimum amount of memory. The compressed output is called the signature of the circuit, and overall approach is dubbed signature analysis.

An example of signature analyzer that compresses a single bit stream is shown in fig.

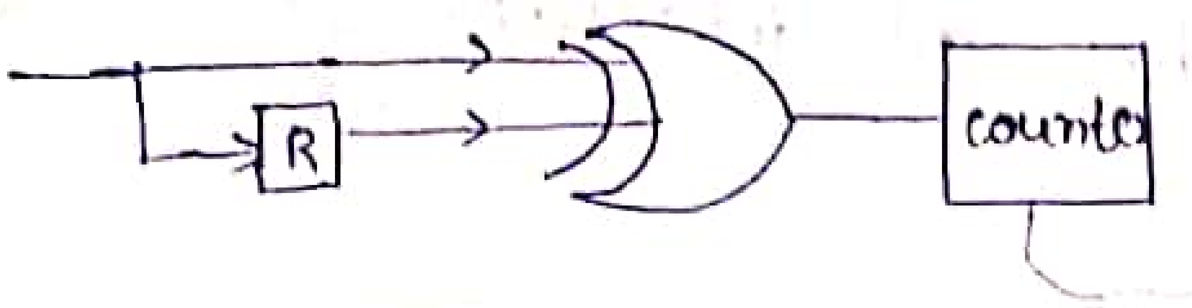


Fig :- single bit stream signature analysis

This circuits simply counts the number of 0→1 and 1→0 transitions in the Input stream. This compression does not guarantee that the received sequence is the correct one.

Another technique, represents a modification of the Linear feed back shift register and has the advantage that the same hardware can be used for both pattern generation and signature analysis.

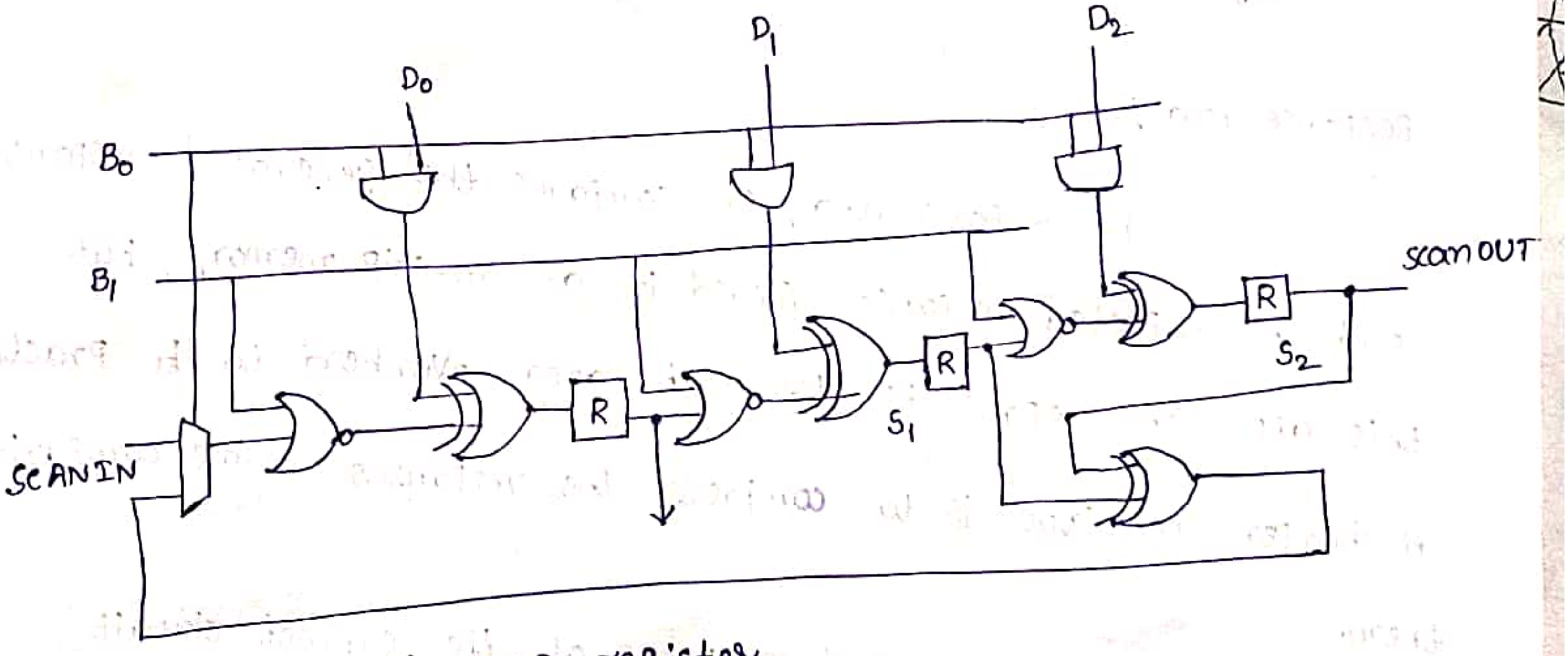


Fig :- BIBO register

Each Incoming data word is XOR'd with the contents of Linear feed back shift register (LFSR). At the end of test sequence, LFSR contains the signature, of data sequence.

which can be compared with signature of correct circuit.

This register not only used for signature analyzer but also used as a scan register, depend on the values of control signals B0, B1.

This test approach, which combines all the different techniques, is called Built-In logic block observation (BILBO).

B_0	B_1	operation mode
0	0	Normal
0	1	Scan
1	0	Pattern generation (or) signature analysis
1	1	Reset

∴ The circuit can be used as register and scan register depends on the values of control signals B_0 and B_1.

Advantages of BIST :-

1. Lower cost of test
2. Better fault coverage
3. Shorter test times if the BIST can be designed to test more structures in Parallel.
4. Easier customer support
5. capability to Perform tests outside the production.
6. Allow the consumers them selves to test the chips Prior to mounting.

Dis advantages :-

1. Additional silicon area and fab processing requirements for the BIST circuits.

2. Reduced Access times

3. Additional Pin requirements to connect out side world

4. Possible issues with the correctness of BIST results

Issues need to consider when Implementing BIST :-

1. Faults to be covered by BIST and how these will be tested for

2. How much chip area will be occupied by BIST circuits

3. External supply and excitation requirements of BIST

4. Test time and effectiveness of the BIST.

5. Flexibility and changeability of BIST

6. How much the BIST will Impact the production electrical test processes that are already in place.

Chapter 2
FPGA Architectures: An Overview

Field Programmable Gate Arrays (FPGAs) were first introduced almost two and a half decades ago. Since then they have seen a rapid growth and have become a popular implementation media for digital circuits. The advancement in process technology has greatly enhanced the logic capacity of FPGAs and has in turn made them a viable implementation alternative for larger and complex designs. Further, programmable nature of their logic and routing resources has a dramatic effect on the quality of final device's area, speed, and power consumption.

This chapter covers different aspects related to FPGAs. First of all an overview of the basic FPGA architecture is presented. An FPGA comprises of an array of programmable logic blocks that are connected to each other through programmable interconnect network. Programmability in FPGAs is achieved through an underlying programming technology. This chapter first briefly discusses different programming technologies. Details of basic FPGA logic blocks and different routing architectures are then described. After that, an overview of the different steps involved in FPGA design flow is given. Design flow of FPGA starts with the hardware description of the circuit which is later synthesized, technology mapped and packed using different tools. After that, the circuit is placed and routed on the architecture to complete the design flow.

The programmable logic and routing interconnect of FPGAs makes them flexible and general purpose but at the same time it makes them larger, slower and more power consuming than standard cell ASICs. However, the advancement in process technology has enabled and necessitated a number of developments in the basic FPGA architecture. These developments are aimed at further improvement in the overall efficiency of FPGAs so that the gap between FPGAs and ASICs might be reduced. These developments and some future trends are presented in the last section of this chapter.

2.1 Introduction to FPGAs

Field programmable Gate Arrays (FPGAs) are pre-fabricated silicon devices that can be electrically programmed in the field to become almost any kind of digital circuit or system. For low to medium volume productions, FPGAs provide cheaper solution and faster time to market as compared to Application Specific Integrated Circuits (ASIC) which normally require a lot of resources in terms of time and money to obtain first device. FPGAs on the other hand take less than a minute to configure and they cost anywhere around a few hundred dollars to a few thousand dollars. Also for varying requirements, a portion of FPGA can be partially reconfigured while the rest of an FPGA is still running. Any future updates in the final product can be easily upgraded by simply downloading a new application bitstream. However, the main advantage of FPGAs i.e. flexibility is also the major cause of its draw back. Flexible nature of FPGAs makes them significantly larger, slower, and more power consuming than their ASIC counterparts. These disadvantages arise largely because of the programmable routing interconnect of FPGAs which comprises of almost 90% of total area of FPGAs. But despite these disadvantages, FPGAs present a compelling alternative for digital system implementation due to their less time to market and low volume cost.

Normally FPGAs comprise of:

- Programmable logic blocks which implement logic functions.
- Programmable routing that connects these logic functions.
- I/O blocks that are connected to logic blocks through routing interconnect and that make off-chip connections.

A generalized example of an FPGA is shown in Fig. 2.1 where configurable logic blocks (CLBs) are arranged in a two dimensional grid and are interconnected by programmable routing resources. I/O blocks are arranged at the periphery of the grid and they are also connected to the programmable routing interconnect. The "programmable/reconfigurable" term in FPGAs indicates their ability to implement a new function on the chip after its fabrication is complete. The reconfigurability/programmability of an FPGA is based on an underlying programming technology, which can cause a change in behavior of a pre-fabricated chip after its fabrication.

2.2 Programming Technologies

There are a number of programming technologies that have been used for reconfigurable architectures. Each of these technologies have different characteristics which in turn have significant effect on the programmable architecture. Some of the well known technologies include static memory [122], flash [54], and anti-fuse [61].

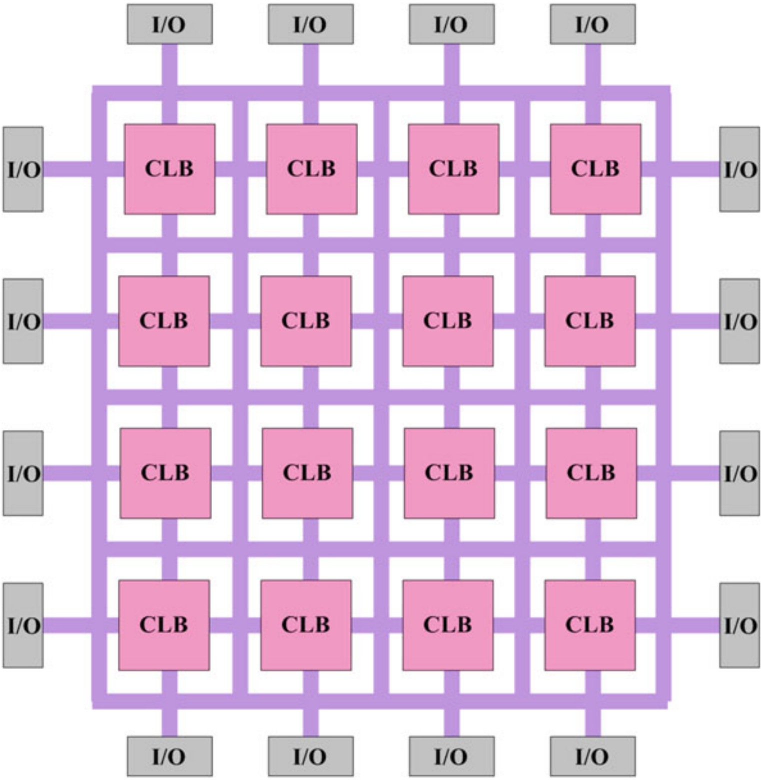


Fig. 2.1 Overview of FPGA architecture [22]

2.2.1 SRAM-Based Programming Technology

Static memory cells are the basic cells used for SRAM-based FPGAs. Most commercial vendors [76, 126] use static memory (SRAM) based programming technology in their devices. These devices use static memory cells which are divided throughout the FPGA to provide configurability. An example of such memory cell is shown in Fig. 2.2. In an SRAM-based FPGA, SRAM cells are mainly used for following purposes:

1. To program the routing interconnect of FPGAs which are generally steered by small multiplexors.
2. To program Configurable Logic Blocks (CLBs) that are used to implement logic functions.

SRAM-based programming technology has become the dominant approach for FPGAs because of its re-programmability and the use of standard CMOS process technology and therefore leading to increased integration, higher speed and lower

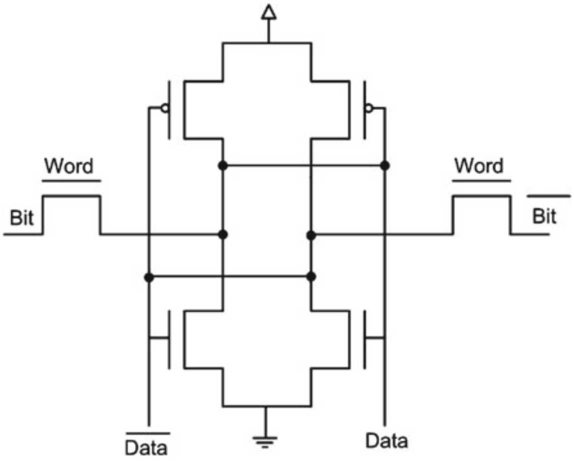


Fig. 2.2 Static memory cell

dynamic power consumption of new process with smaller geometry. There are however a number of drawbacks associated with SRAM-based programming technology. For example an SRAM cell requires 6 transistors which makes the use of this technology costly in terms of area compared to other programming technologies. Further SRAM cells are volatile in nature and external devices are required to permanently store the configuration data. These external devices add to the cost and area overhead of SRAM-based FPGAs.

2.2.2 Flash Programming Technology

One alternative to the SRAM-based programming technology is the use of flash or EEPROM based programming technology. Flash-based programming technology offers several advantages. For example, this programming technology is nonvolatile in nature. Flash-based programming technology is also more area efficient than SRAM-based programming technology. Flash-based programming technology has its own disadvantages also. Unlike SRAM-based programming technology, flash-based devices can not be reconfigured/reprogrammed an infinite number of times. Also, flash-based technology uses non-standard CMOS process.

2.2.3 Anti-fuse Programming Technology

An alternative to SRAM and flash-based technologies is anti-fuse programming technology. The primary advantage of anti-fuse programming technology is its low area. Also this technology has lower on resistance and parasitic capacitance than other two programming technologies. Further, this technology is non-volatile in nature. There are however significant disadvantages associated with this programming technology. For example, this technology does not make use of standard CMOS process. Also, anti-fuse programming technology based devices can not be reprogrammed.

In this section, an overview of three commonly used programming technologies is given where all of them have their advantages and disadvantages. Ideally, one would like to have a programming technology which is reprogrammable, non-volatile, and that uses a standard CMOS process. Apparently, none of the above presented technologies satisfy these conditions. However, SRAM-based programming technology is the most widely used programming technology. The main reason is its use of standard CMOS process and for this very reason, it is expected that this technology will continue to dominate the other two programming technologies.

2.3 Configurable Logic Block

A configurable logic block (CLB) is a basic component of an FPGA that provides the basic logic and storage functionality for a target application design. In order to provide the basic logic and storage capability, the basic component can be either a transistor or an entire processor. However, these are the two extremes where at one end the basic component is very fine-grained (in case of transistors) and requires large amount of programmable interconnect which eventually results in an FPGA that suffers from area-inefficiency, low performance and high power consumption. On the other end (in case of processor), the basic logic block is very coarse-grained and can not be used to implement small functions as it will lead to wastage of resources. In between these two extremes, there exists a spectrum of basic logic blocks. Some of them include logic blocks that are made of NAND gates [101], an interconnection of multiplexors [44], lookup table (LUT) [121] and PAL style wide input gates [124]. Commercial vendors like Xilinx and Altera use LUT-based CLBs to provide basic logic and storage functionality. LUT-based CLBs provide a good trade-off between too fine-grained and too coarse-grained logic blocks. A CLB can comprise of a single basic logic element (BLE), or a cluster of locally interconnected BLEs (as shown in Fig. 2.4). A simple BLE consists of a LUT, and a Flip-Flop. A LUT with k inputs (LUT-k) contains 2^k configuration bits and it can implement any k-input boolean function. Figure 2.3 shows a simple BLE comprising of a 4 input LUT (LUT-4) and a D-type Flip-Flop. The LUT-4 uses 16 SRAM bits to implement any 4 inputs boolean function. The output of LUT-4 is connected to an optional Flip-Flop. A multiplexor selects the BLE output to be either the output of a Flip-Flop or the LUT-4.

A CLB can contain a cluster of BLEs connected through a local routing network. Figure 2.4 shows a cluster of 4 BLEs; each BLE contains a LUT-4 and a Flip-Flop. The BLE output is accessible to other BLEs of the same cluster through a local routing network. The number of output pins of a cluster are equal to the total number of BLEs in a cluster (with each BLE having a single output). However, the number of input pins of a cluster can be less than or equal to the sum of input pins required

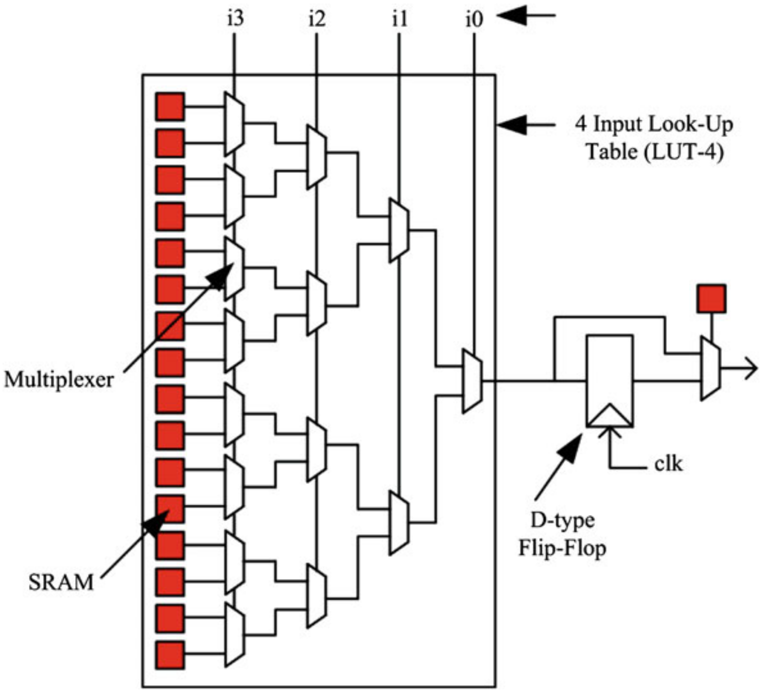


Fig. 2.3 Basic logic element (BLE) [22]

by all the BLEs in the cluster. Modern FPGAs contain typically 4 to 10 BLEs in a single cluster. Although here we have discussed only basic logic blocks, many modern FPGAs contain a heterogeneous mixture of blocks, some of which can only be used for specific purposes. Theses specific purpose blocks, also referred here as hard blocks, include memory, multipliers, adders and DSP blocks etc. Hard blocks are very efficient at implementing specific functions as they are designed optimally to perform these functions, yet they end up wasting huge amount of logic and routing resources if unused. A detailed discussion on the use of heterogeneous mixture of blocks for implementing digital circuits is presented in Chap. 4 where both advantages and disadvantages of heterogeneous FPGA architectures and a remedy to counter the resource loss problem are discussed in detail.

2.4 FPGA Routing Architectures

As discussed earlier, in an FPGA, the computing functionality is provided by its programmable logic blocks and these blocks connect to each other through programmable routing network. This programmable routing network provides routing

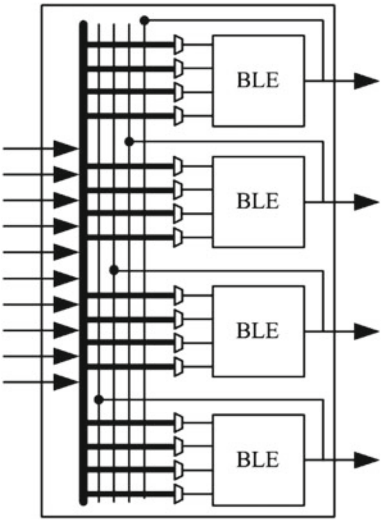


Fig. 2.4 A configurable logic block (CLB) having four BLEs [22]

connections among logic blocks and I/O blocks to implement any user-defined circuit. The routing interconnect of an FPGA consists of wires and programmable switches that form the required connection. These programmable switches are configured using the programmable technology.

Since FPGA architectures claim to be potential candidate for the implementation of any digital circuit, their routing interconnect must be very flexible so that they can accommodate a wide variety of circuits with widely varying routing demands. Although the routing requirements vary from circuit to circuit, certain common characteristics of these circuits can be used to optimally design the routing interconnect of FPGA architecture. For example most of the designs exhibit locality, hence requiring abundant short wires. But at the same time there are some distant connections, which leads to the need for sparse long wires. So, care needs to be taken into account while designing routing interconnect for FPGA architectures where we have to address both flexibility and efficiency. The arrangement of routing resources, relative to the arrangement of logic blocks of the architecture, plays a very important role in the overall efficiency of the architecture. This arrangement is termed here as global routing architecture whereas the microscopic details regarding the switching topology of different switch blocks is termed as detailed routing architecture. On the basis of the global arrangement of routing resources of the architecture, FPGA architectures can be categorized as either hierarchical [4] or island-style [22]. In this section, we present a detailed overview of both routing architectures.

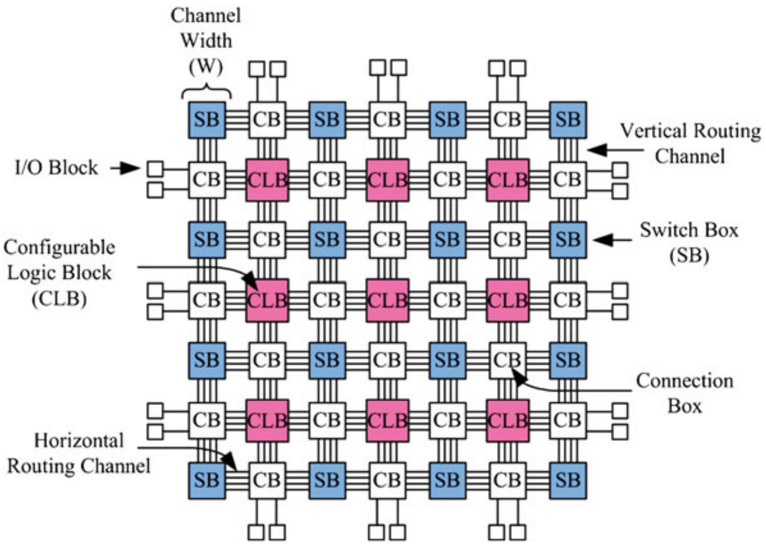


Fig. 2.5 Overview of mesh-based FPGA architecture [22]

2.4.1 Island-Style Routing Architecture

Figure 2.5 shows a traditional island-style FPGA architecture (also termed as mesh-based FPGA architecture). This is the most commonly used architecture among academic and commercial FPGAs. It is called island-style architecture because in this architecture configurable logic blocks look like islands in a sea of routing interconnect. In this architecture, configurable logic blocks (CLBs) are arranged on a 2D grid and are interconnected by a programmable routing network. The Input/Output (I/O) blocks on the periphery of FPGA chip are also connected to the programmable routing network. The routing network comprises of pre-fabricated wiring segments and programmable switches that are organized in horizontal and vertical routing channels.

The routing network of an FPGA occupies 80–90% of total area, whereas the logic area occupies only 10–20% area [22]. The flexibility of an FPGA is mainly dependent on its programmable routing network. A mesh-based FPGA routing network consists of horizontal and vertical routing tracks which are interconnected through switch boxes (SB). Logic blocks are connected to the routing network through connection boxes (CB). The flexibility of a connection box (Fc) is the number of routing tracks of adjacent channel which are connected to the pin of a block. The connectivity of input pins of logic blocks with the adjacent routing channel is called as Fc(in); the connectivity of output pins of the logic blocks with the adjacent routing channel is called as Fc(out). An Fc(in) equal to 1.0 means that all the tracks of adjacent routing channel are connected to the input pin of the logic block. The flexibility of switch box (Fs) is the total number of tracks with which every track entering in the switch

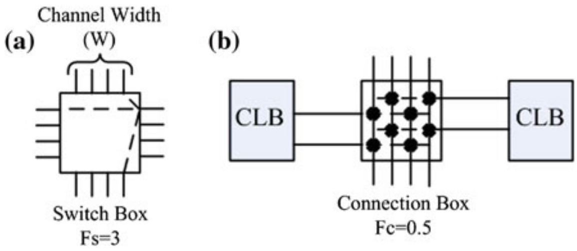


Fig. 2.6 Example of switch and connection box

box connects to. The number of tracks in routing channel is called the channel width of the architecture. Same channel width is used for all horizontal and vertical routing channels of the architecture. An example explaining the switch box, connection box flexibilities, and routing channel width is shown in Fig. 2.6. In this figure switch box has Fs = 3 as each track incident on it is connected to 3 tracks of adjacent routing channels. Similarly, connection box has Fc(in) = 0.5 as each input of the logic block is connected to 50% of the tracks of adjacent routing channel.

The routing tracks connected through a switch box can be bidirectional or unidirectional (also called as directional) tracks. Figure 2.7 shows a bidirectional and a unidirectional switch box having Fs equal to 3. The input tracks (or wires) in both these switch boxes connect to 3 other tracks of the same switch box. The only limitation of unidirectional switch box is that their routing channel width must be in multiples of 2.

Generally, the output pins of a block can connect to any routing track through pass transistors. Each pass transistor forms a tristate output that can be independently turned on or off. However, single-driver wiring technique can also be used to connect output pins of a block to the adjacent routing tracks. For single-driver wiring, tristate elements cannot be used; the output of block needs to be connected to the neighboring routing network through multiplexors in the switch box. Modern commercial FPGA architectures have moved towards using single-driver, directional routing tracks. Authors in [51] show that if single-driver directional wiring is used instead of bidirectional wiring, 25% improvement in area, 9% in delay and 32% in area-delay can be achieved. All these advantages are achieved without making any major changes in the FPGA CAD flow.

In mesh-based FPGAs, multi-length wires are created to reduce delay. Figure 2.8 shows an example of different length wires. Longer wire segments span multiple blocks and require fewer switches, thereby reducing routing area and delay. However, they also decrease routing flexibility, which reduces the probability to route a hardware circuit successfully. Modern commercial FPGAs commonly use a combination of long and short wires to balance flexibility, area and delay of the routing network.

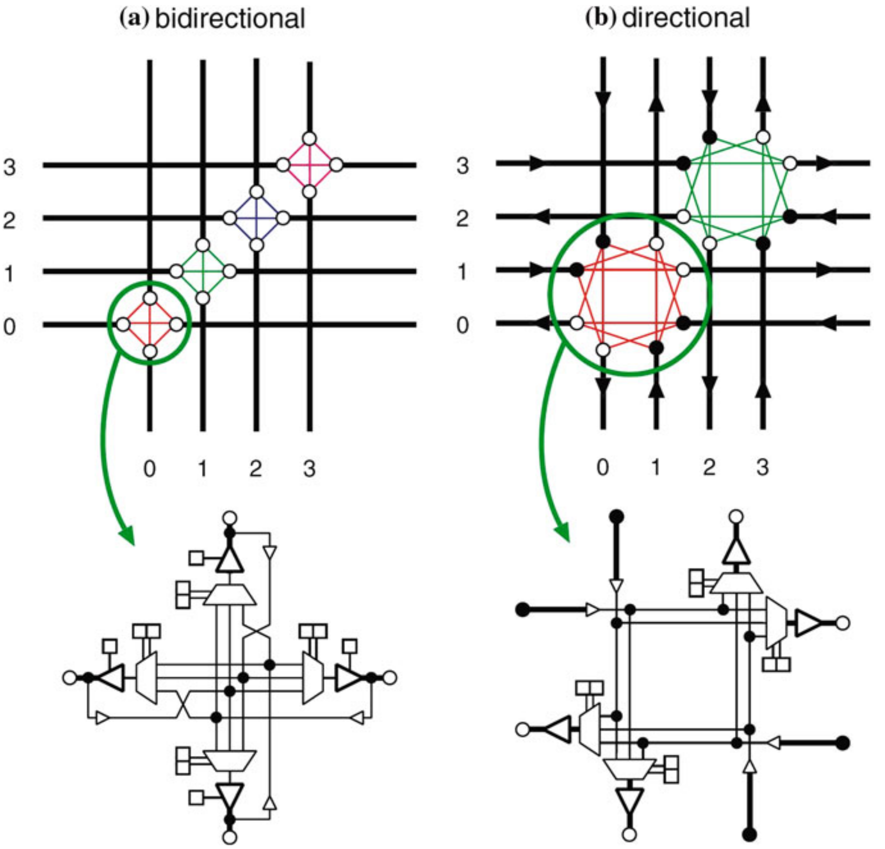


Fig. 2.7 Switch block, length 1 wires [51]

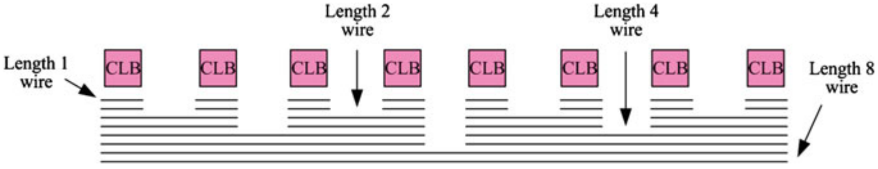


Fig. 2.8 Channel segment distribution

2.4.1.1 Altera's Stratix II Architecture

Until now, we have presented a general overview about island-style routing architecture. Now we present a commercial example of this kind of architectures. Altera's Stratix II [106] architecture is an industrial example of an island-style FPGA (Fig. 2.9). The logic structure consists of LABs (Logic Array Blocks), memory blocks, and digital signal processing (DSP) blocks. LABs are used to

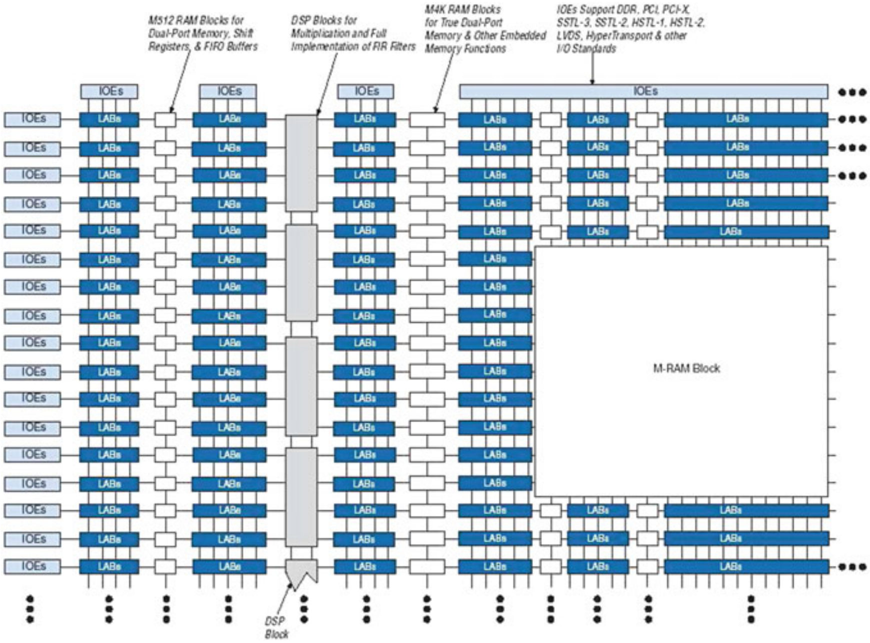


Fig. 2.9 Altera's stratix-II block diagram

implement general-purpose logic, and are symmetrically distributed in rows and columns throughout the device fabric. The DSP blocks are custom designed to implement full-precision multipliers of different granularities, and are grouped into columns. Input- and output-only elements (IOEs) represent the external interface of the device. IOEs are located along the periphery of the device.

Each Stratix II LAB consists of eight Adaptive Logic Modules (ALMs). An ALM consists of 2 adaptive LUTs (ALUTs) with eight inputs altogether. Construction of an ALM allows implementation of 2 separate 4-input Boolean functions. Further, an ALM can also be used to implement any six-input Boolean function, and some seven-input functions. In addition to lookup tables, an ALM provides 2 programmable registers, 2 dedicated full-adders, a carry chain, and a register-chain. Full-adders and carry chain can be used to implement arithmetic operations, and the register-chain is used to build shift registers. Outputs of an ALM drive all types of interconnect provided by the Stratix II device. Figure 2.10 illustrates a LAB interconnect interface.

Interconnections between LABs, RAM blocks, DSP blocks and the IOEs are established using the Multi-track interconnect structure. This interconnect structure consists of wire segments of different lengths and speeds. The interconnect wire-segments span fixed distances, and run in the horizontal (row interconnects) and vertical (column interconnects) directions. The row interconnects (Fig. 2.11) can be used to route signals between LABs, DSP blocks, and memory blocks in the same row. Row interconnect resources are of the following types:

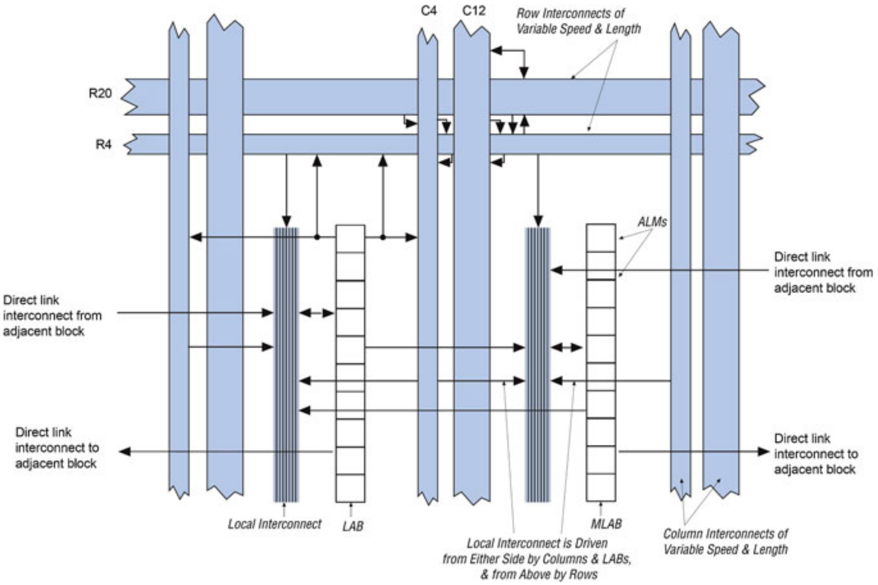

C4
C12
Row Interconnects of Variable Speed & Length
R20
R4
ALMs
Direct link interconnect from adjacent block
Direct link interconnect from adjacent block
Direct link interconnect to adjacent block
Direct link interconnect to adjacent block
Local Interconnect
LAB
MLAB
Local Interconnect is Driven from Either Side by Columns & LABs, & from Above by Rows
Column Interconnects of Variable Speed & Length

Fig. 2.10 Stratix-II logic array block (LAB) structure

- Direct connections between LABs and adjacent blocks.
- R4 resources that span 4 blocks to the left or right.
- R24 resources that provide high-speed access across 24 columns.

Each LAB owns its set of R4 interconnects. A LAB has approximately equal numbers of driven-left and driven-right R4 interconnects. An R4 interconnect that is driven to the left can be driven by either the primary LAB (Fig. 2.11) or the adjacent LAB to the left.

Similarly, a driven-right R4 interconnect may be driven by the primary LAB or the LAB immediately to its right. Multiple R4 resources can be connected to each other to establish longer connections within the same row. R4 interconnects can also drive C4 and C16 column interconnects, and R24 high speed row resources.

Column interconnect structure is similar to row interconnect structure. Column interconnects include:

- Carry chain interconnects within a LAB, and from LAB to LAB in the same column.
- Register chain interconnects.
- C4 resources that span 4 blocks in the up and down directions.
- C16 resources for high-speed vertical routing across 16 rows.

Carry chain and register chain interconnects are separated from local interconnect (Fig. 2.10) in a LAB. Each LAB has its own set of driven-up and driven-down C4 interconnects. C4 interconnects can also be driven by the LABs that are immediately

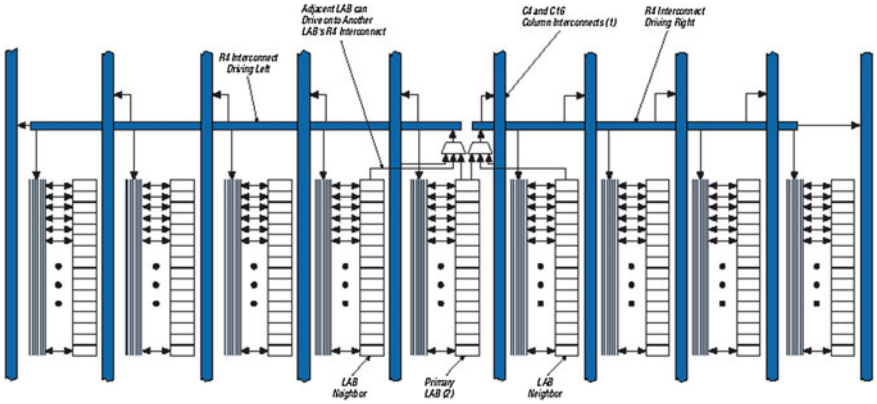


Fig. 2.11 R4 interconnect connections

adjacent to the primary LAB. Multiple C4 resources can be connected to each other to form longer connections within a column, and C4 interconnects can also drive row interconnects to establish column-to-column interconnections. C16 interconnects are high-speed vertical resources that span 16 LABs. A C16 interconnect can drive row and column interconnects at every fourth LAB. A LAB local interconnect structure cannot be directly driven by a C16 interconnect; only C4 and R4 interconnects can drive a LAB local interconnect structure. Figure 2.12 shows the C4 interconnect structure in the Stratix II device.

2.4.2 Hierarchical Routing Architecture

Most logic designs exhibit locality of connections; hence implying a hierarchy in placement and routing of connections between different logic blocks. Hierarchical routing architectures exploit this locality by dividing FPGA logic blocks into separate groups/clusters. These clusters are recursively connected to form a hierarchical structure. In a hierarchical architecture (also termed as tree-based architecture), connections between logic blocks within same cluster are made by wire segments at the lowest level of hierarchy. However, the connection between blocks residing in different groups require the traversal of one or more levels of hierarchy. In a hierarchical architecture, the signal bandwidth varies as we move away from the bottom level and generally it is widest at the top level of hierarchy. The hierarchical routing architecture has been used in a number of commercial FPGA families including Altera Flex10K [10], Apex [15] and ApexII [16] architectures. We assume that Multilevel hierarchical interconnect regroups architectures with more than 2 levels of hierarchy and Tree-based ones.

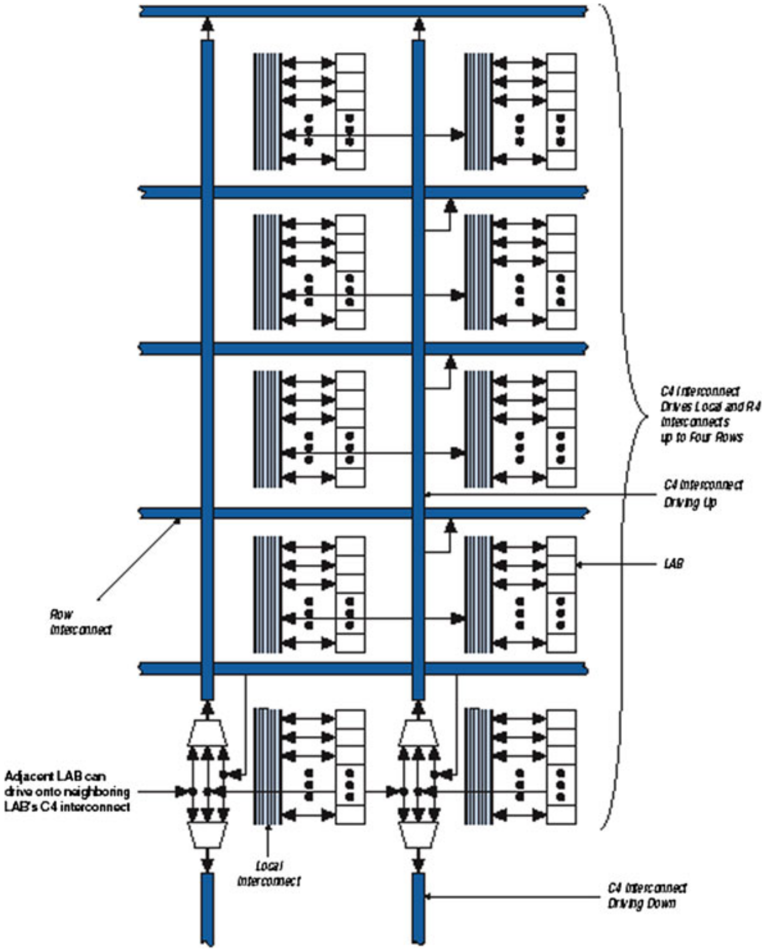

C4 Interconnect Drives Local and R4 Interconnects up to Four Rows
C4 Interconnect Driving Up
LAB
Row Interconnect
Adjacent LAB can drive onto neighboring LAB's C4 interconnect
Local Interconnect
C4 Interconnect Driving Down

Fig. 2.12 C4 interconnect connections

2.4.2.1 HFPGA: Hierarchical FPGA

In the hierarchical FPGA called HFPGA, LBs are grouped into clusters. Clusters are then grouped recursively together (see Fig. 2.13). The clustered VPR mesh architecture [22] has a Hierarchical topology with only two levels. Here we consider multilevel hierarchical architectures with more than 2 levels. In [1] and [129] various hierarchical structures were discussed. The HFPGA routability depends on switch boxes topologies. HFPGAs comprising fully populated switch boxes ensure 100% routability but are very penalizing in terms of area. In [129] authors explored the HFPGA architecture, investigating how the switch pattern can be partly depopulated while maintaining a good routability.

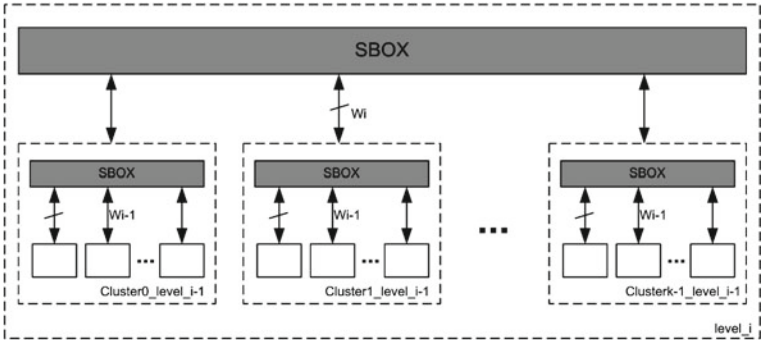


Fig. 2.13 Hierarchical FPGA topology

2.4.2.2 HSRA: Hierarchical Synchronous Reconfigurable Array

An example of an academic hierarchical routing architecture is shown in Fig. 2.14. It has a strictly hierarchical, tree-based interconnect structure. In this architecture, the only wire segments that directly connect to the logic units are located at the leaves of the interconnect tree. All other wire segments are decoupled from the logic structure. A logic block of this architecture consists of a pair of 2-input Look Up Table (2-LUT) and a D-type Flip Flop (D-FF). The input-pin connectivity is based on a choose-k strategy [4], and the output pins are fully connected. The richness of this interconnect structure is defined by its base channel width c and interconnect growth rate p. The base channel width c is defined as the number of tracks at the leaves of the interconnect Tree (in Fig. 2.14, $c = 3$). Growth rate p is defined as the rate at which the interconnect bandwidth grows towards the upper levels. The interconnect growth rate can be realized either using non-compressing or compressing switch blocks. The details regarding these switch blocks is as follows:

- Non-compressing (2:1) switch blocks—The number of tracks at the upper level are equal to the sum of the number of tracks of the children at lower level. For example, in Fig. 2.14, non-compressing switch blocks are used between levels 1, 2 and levels 3, 4.
- Compressing (1:1) switch blocks—The number of tracks at the upper level are equal to the number of tracks of either child at the lower level. For example, in Fig. 2.14, compressing switch blocks are used between levels 2 and 3.

A repeating combination of non-compressing and compressing switch blocks can be used to realize any value of p less than one. For example, a repeating pattern of (2:1, 1:1) switch blocks realizes $p = 0.5$, while the pattern (2:1, 2:1, 1:1) realizes $p = 0.67$. An architecture that has only 2:1 switch blocks provides a growth rate of $p = 1$.

Another hierarchical routing architecture is presented in [132] where the global routing architecture (i.e. the position of routing resources relative to logic resources

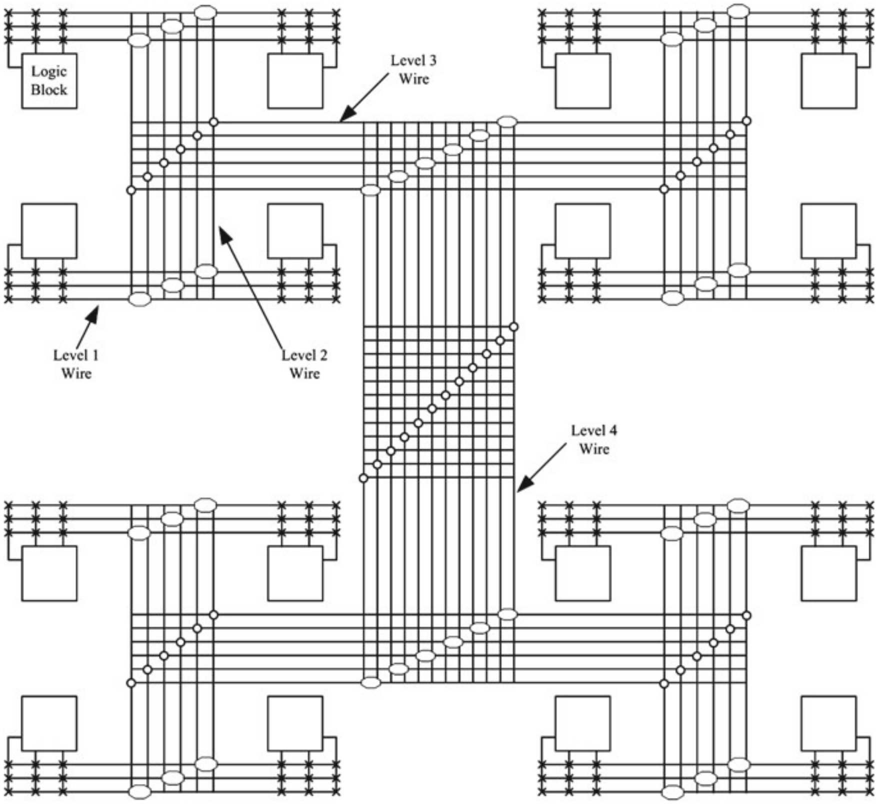


Fig. 2.14 Example of hierarchical routing architecture [4]

of the architecture) remains the same as in [4]. However, there are several key differences at the level of detailed routing architecture (i.e. the way the routing resources are connected to each other, flexibility of switch blocks etc.) that separate the two architectures. For example the architecture shown in Fig. 2.14 has one bidirectional interconnect that uses bidirectional switches and it supports only arity-2 (i.e. each cluster can contain only two sub-clusters). On contrary, the architecture presented in [132] supports two separate unidirectional interconnect networks: one is downward interconnect whereas other is upward interconnect network. Further this architecture is more flexible as it can support logic blocks with different sizes and also the clusters/groups of the routing architecture can have different arity sizes. Further details of this architecture, from now on alternatively termed as tree-based architecture, are presented in next chapter.

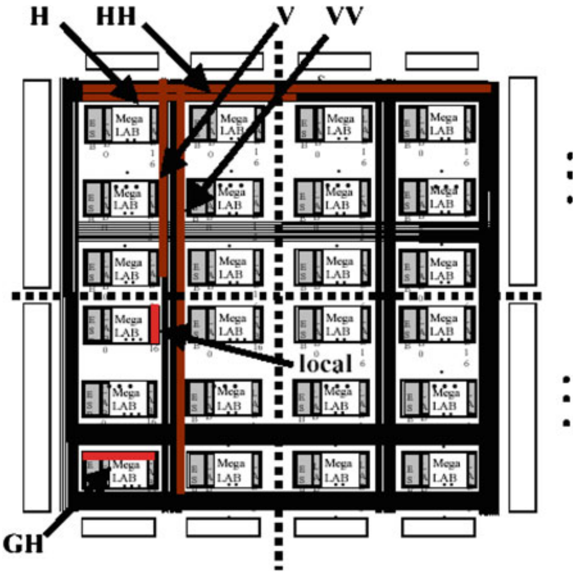


Fig. 2.15 The APEX programmable logic Devices [87]

2.4.2.3 APEX: Altera

APEX architecture is a commercial product from Altera Corporation which includes 3 levels of interconnect hierarchy. Figure 2.15 shows a diagram of the APEX 20K400 programmable logic device. The basic logic-element (LE) is a 4-input LUT and DFF pair. Groups of 10 LEs are grouped into a logic-array-block or LAB. Interconnect within a LAB is complete, meaning that a connection from the output of any LE to the input of another LE in its LAB always exists, and any signal entering the input region can reach every LE.

Groups of 16 LABs form a MegaLab. Interconnect within a MegaLab requires an LE to drive a GH (MegaLab global H) line, a horizontal line, which switches into the input region of any other LAB in the same MegaLab. Adjacent LABs have the ability to interleave their input regions, so an LE in LAB_i can usually drive LAB_{i+1} without using a GH line. A 20K400 MegaLab contains 279 GH lines.

The top-level architecture is a 4 by 26 array of MegaLabs. Communication between MegaLabs is accomplished by global H (horizontal) and V (vertical) wires, that switch at their intersection points. The H and V lines are segmented by a bidirectional segmentation buffer at the horizontal and vertical centers of the chip. In Fig. 2.15, We denote the use of a single (half-chip) line as H or V and a double or full-chip line through the segmentation buffer as HH or VV. The 20K400 contains 100 H lines per MegaLab row, and 80 V lines per LAB-column.

In this section, so far we have given an overview of the two routing architectures that are commonly employed in FPGAs. Both architectures have their positive and negative points. For example, hierarchical routing architectures exploit the

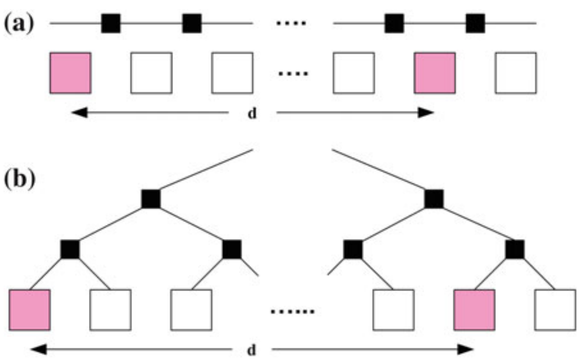


Fig. 2.16 **a** Number of series switches in a mesh structure **b** Number of series switches in a tree structure

locality exhibited by the most of the designs and in turn offer smaller delays and more predictable routing compared to island-style architectures. The speed of a net is determined by the number of routing switches it has to pass and the length of wires. In a mesh-based architecture, the number of segments increase linearly with manhattan distance *d* between the logic blocks to be connected. However, for tree-based architecture the distance *d* between the blocks to be connected increases in a logarithmic manner [82]. This fact is illustrated in Fig. 2.16. On the other hand, scalability is an issue in hierarchical routing architectures and there might be some design mapping issues. But in the case of mesh-based architecture, there are no such issues as it offers a tile-based layout where a tile once formed can be replicated horizontally and vertically to make as large architecture as we wish.

2.5 Software Flow

FPGA architectures have been intensely investigated over the past two decades. A major aspect of FPGA architecture research is the development of Computer Aided Design (CAD) tools for mapping applications to FPGAs. It is well established that the quality of an FPGA-based implementation is largely determined by the effectiveness of accompanying suite of CAD tools. Benefits of an otherwise well designed, feature rich FPGA architecture might be impaired if the CAD tools cannot take advantage of the features that the FPGA provides. Thus, CAD algorithm research is essential to the necessary architectural advancement to narrow the performance gaps between FPGAs and other computational devices like ASICs.

The software flow (CAD flow) takes an application design description in a Hardware Description Language (HDL) and converts it to a stream of bits that is eventually programmed on the FPGA. The process of converting a circuit description into a format that can be loaded into an FPGA can be roughly divided into five distinct steps, namely: synthesis, technology mapping, mapping, placement and routing. The final output of FPGA CAD tools is a bitstream that configures the state of the memory

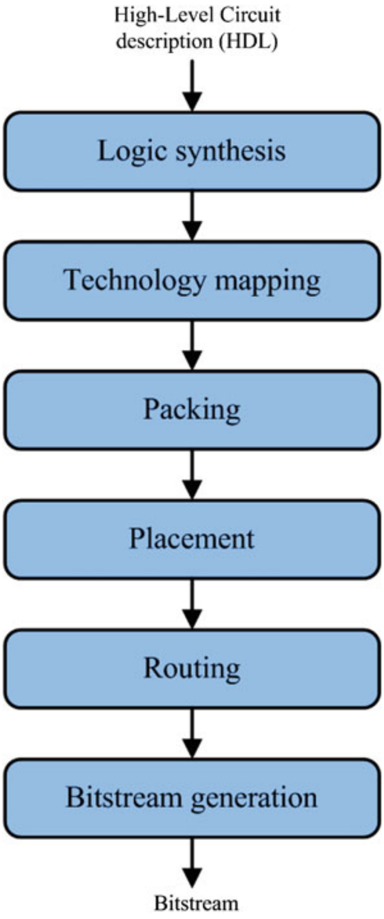


Fig. 2.17 FPGA software flow

bits in an FPGA. The state of these bits determines the logical function that the FPGA implements. Figure 2.17 shows a generalized software flow for programming an application circuit on an FPGA architecture. A description of various modules of software flow is given in the following part of this section. The details of these modules are generally indifferent to the kind of routing architecture being used and they are applicable to both architectures described earlier unless otherwise specified.

2.5.1 Logic Synthesis

The flow of FPGA starts with the logic synthesis of the netlist being mapped on it. Logic synthesis [26, 27] transforms an HDL description (VHDL or Verilog) into a set of boolean gates and Flip-Flops. The synthesis tools transform the

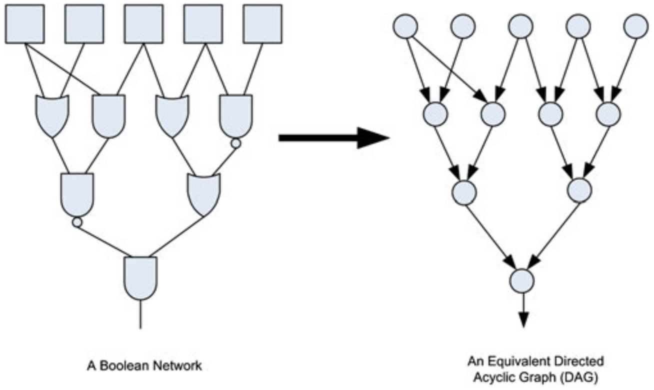


Fig. 2.18 Directed acyclic graph representation of a circuit

register-transfer-level (RTL) description of a design into a hierarchical boolean network. Various technology-independent techniques are applied to optimize the boolean network. The typical cost function of technology-independent optimizations is the total literal count of the factored representation of the logic function. The literal count correlates very well with the circuit area. Further details of logic synthesis are beyond the scope of this book.

2.5.2 Technology Mapping

The output from synthesis tools is a circuit description of Boolean logic gates, flip-flops and wiring connections between these elements. The circuit can also be represented by a Directed Acyclic Graph (DAG). Each node in the graph represents a gate, flip-flop, primary input or primary output. Each edge in the graph represents a connection between two circuit elements. Figure 2.18 shows an example of a DAG representation of a circuit. Given a library of cells, the technology mapping problem can be expressed as finding a network of cells that implements the Boolean network. In the FPGA technology mapping problem, the library of cells is composed of k-input LUTs and flip-flops. Therefore, FPGA technology mapping involves transforming the Boolean network into k-bounded cells. Each cell can then be implemented as an independent k-LUT. Figure 2.19 shows an example of transforming a Boolean network into k-bounded cells. Technology mapping algorithms can optimize a design for a set of objectives including depth, area or power. The FlowMap algorithm [64] is the most widely used academic tool for FPGA technology mapping. FlowMap is a breakthrough in FPGA technology mapping because it is able to find a depth-optimal solution in polynomial time. FlowMap guarantees depth optimality at the expense of logic duplication. Since the introduction of FlowMap, numerous technology mappers have been designed that optimize for area and run-time while still maintaining

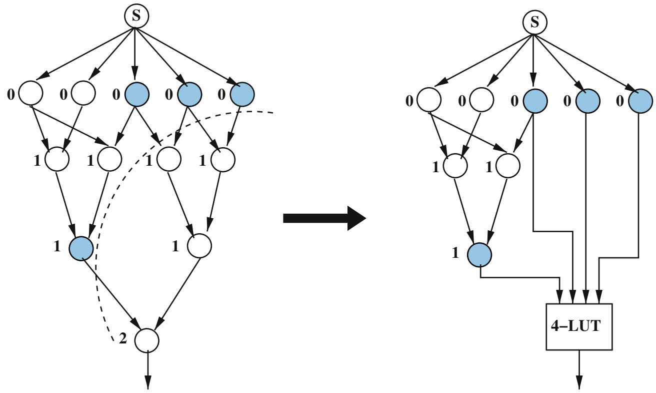


Fig. 2.19 Example of technology mapping

the depth-optimality of the circuit [65–67]. The result of the technology mapping step generates a network of k-bounded LUTs and flip-flops.

2.5.3 Clustering/Packing

The logic elements in a Mesh-based FPGA are typically arranged in two levels of hierarchy. The first level consists of logic blocks (LBs) which are k-input LUT and flip-flop pairs. The second level hierarchy groups *k* LBs together to form logic blocks clusters. The clustering phase of the FPGA CAD flow is the process of forming groups of *k* LBs. These clusters can then be mapped directly to a logic element on an FPGA. Figure 2.20 shows an example of the clustering process.

Clustering algorithms can be broadly categorized into three general approaches, namely top-down [39, 78], depth-optimal [84, 100] and bottom-up [14, 17, 43]. Top-down approaches partition the LBs into clusters by successively subdividing the network or by iteratively moving LBs between parts. Depth-optimal solutions attempt to minimize delay at the expense of logic duplication. Bottom-up approaches are generally preferred for FPGA CAD tools due to their fast run times and reasonable timing delays. They only consider local connectivity information and can easily satisfy clusters pin constraints. Top-down approaches offer the best solutions; however, their computational complexity can be prohibitive.

2.5.3.1 Bottom-up Approaches

Bottom-up approaches build clusters sequentially one at a time. The process starts by choosing an LB which acts as a cluster seed. LBs are then greedily selected and added to the cluster, applying various attraction functions. The VPack [14] attraction

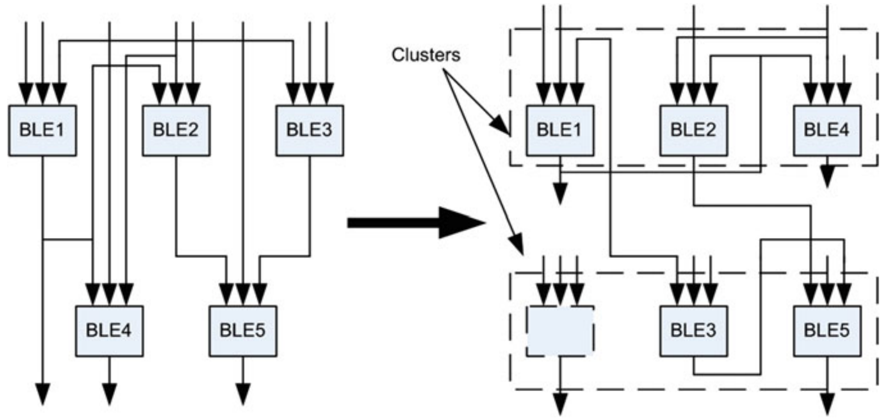


Fig. 2.20 Example of packing

function is based on the number of shared nets between a candidate LB and the LBs that are already in the cluster. For each cluster, the attraction function is used to select a seed LB from the set of all LBs that have not already been packed. After packing a seed LB into the new cluster, a second attraction function selects new LBs to pack into the cluster. LBs are packed into the cluster until the cluster reaches full capacity or all cluster inputs have been used. If all cluster inputs become occupied before this cluster reaches full capacity, a hill-climbing technique is applied, searching for LBs that do not increase the number of inputs used by the cluster. The VPack pseudo-code is outlined in algorithm 2.1.

T-VPack [22] is a timing-driven version of VPack which gives added weight to grouping LBs on the critical path together. The algorithm is identical to VPack, however, the attraction functions which select the LBs to be packed into the clusters are different. The VPack seed function chooses LBs with the most used inputs, whereas the T-VPack seed function chooses LBs that are on the most critical path. VPack's second attraction function chooses LBs with the largest number of connections with the LBs already packed into the cluster. T-VPack's second attraction function has two components for a LB B being considered for cluster C:

$$Attraction(B, C) = \alpha.Crit(B) + (1 - \alpha)\frac{\mid Nets(B) \cap Nets(C) \mid}{G} \tag{2.1}$$

where $Crit(B)$ is a measure of how close LB B is to being on the critical path, $Nets(B)$ is the set of nets connected to LB B, $Nets(C)$ is the set of nets connected to the LBs already selected for cluster C, α is a user-defined constant which determines the relative importance of the attraction components, and G is a normalizing factor. The first component of T-VPack's second attraction function chooses critical-path LBs, and the second chooses LBs that share many connections with the LBs already packed into the cluster. By initializing and then packing clusters with

```
UnclusteredLBs = PatternMatchToLBs(LUTs,Registers);
LogicClusters = NULL;
while UnclusteredLBs != NULL do
    C = GetLBwithMostUsedInputs(UnclusteredLBs);
    while $| C | < k$ do
        /*cluster is not full*/
        BestLB = MaxAttractionLegalLB(C,UnclusteredLBs);
        if BestLB == NULL then
            /*No LB can be added to this cluster*/
            break;
        endif
        $UnclusteredLBs = UnclusteredLB - BestLB$;
        $C = C \cup BestLB$;
    endw
    if $| C | < k$ then
        /*Cluster is not full - try hill climbing*/
        while $| C | < k$ do
            BestLB = MinClusterInputIncreaseLB(C,UnclusteredLBs);
            $C = C \cup BestLB$;
            $UnclusteredLBs = UnclusteredLB - BestLB$;
        endw
        if ClusterIsIllegal(C) then
            RestoreToLastLegalState(C,UnclusteredLBs);
        endif
    endif
    $LogicClusters = LogicClusters \cup C$;
endw
```

Algorithm 2.1 Pseudo-code of the VPack Algorithm [22]

critical-path LBs, the algorithm is able to absorb long sequences of critical-path LBs into clusters. This minimizes circuit delay since the local interconnect within the cluster is significantly faster than the global interconnect of the FPGA. RPack [43] improves routability of a circuit by introducing a new set of routability metrics. RPack significantly reduced the channel widths required by circuits compared to VPack. T-RPack [43] is a timing driven version of RPack which is similar to T-VPack by giving added weight to grouping LBs on the critical path. iRAC [17] improves the routability of circuits even further by using an attraction function that attempts to encapsulate as many low fanout nets as possible within a cluster. If a net can be completely encapsulated within a cluster, there is no need to route that net in the external routing network. By encapsulating as many nets as possible within clusters, routability is improved because there are less external nets to route in total.

2.5.3.2 Top-down Approaches

The K-way partitioning problem seeks to minimize a given cost function of such an assignment. A standard cost function is net cut, which is the number of hyperedges that span more than one partition, or more generally, the sum of weights of such hyperedges. Constraints are typically imposed on the solution, and make the problem difficult. For example some vertices can be fixed in their parts or the total vertex weight in each part must be limited (balance constraint and FPGA clusters size). With balance constraints, the problem of partitioning optimally a hypergraph is known to be NP-hard [85]. However, since partitioning is critical in several practical applications, heuristic algorithms were developed with near-linear runtime. Such move-based heuristics for k-way hypergraph partitioning appear in [24, 34, 110].

Fiduccia-Mattheyses Algorithm

The Fiduccia-Mattheyses (FM) heuristics [34] work by prioritizing moves by gain. A move changes to which partition a particular vertex belongs, and the gain is the corresponding change of the cost function. After each vertex is moved, gains for connected modules are updated.

```
partitioning = initial_solution;
while solution quality improves do
    Initialize gain_container from partitioning;
    solution_cost = partitioning.get_cost();
    while not all vertices locked do
        move = choose_move();
        solution_cost += gain_container.get_gain(move);
        gain_container.lock_vertex(move.vertex());
        gain_update(move);
        partitioning.apply(move);
    endw
    roll back partitioning to best seen solution;
    gain_container.unlock_all();
endw
```

Algorithm 2.2 Pseudo-code for FM Heuristic [38]

The Fiduccia-Mattheyses (FM) heuristic for partitioning hypergraphs is an iterative improvement algorithm. FM starts with a possibly random solution and changes the solution by a sequence of moves which are organized as passes. At the beginning of a pass, all vertices are free to move (unlocked), and each possible move is labeled with the immediate change to the cost it would cause; this is called the gain of the move (positive gains reduce solution cost, while negative gains increase it). Iteratively, a move with highest gain is selected and executed, and the moving vertex is locked, i.e., is not allowed to move again during that pass. Since moving a vertex can change gains of adjacent vertices, after a move is executed all affected gains are updated. Selection and execution of a best-gain move, followed by gain update, are repeated until every vertex is locked. Then, the best solution seen during the pass is adopted as the starting solution of the next pass. The algorithm terminates when a

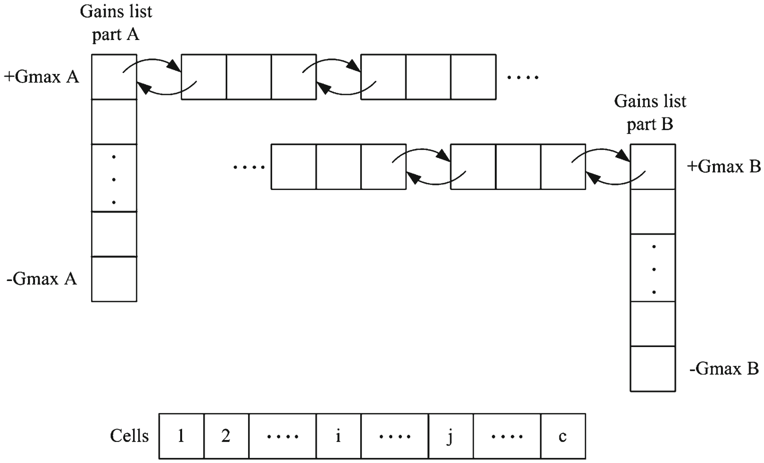


Fig. 2.21 The gain bucket structure as illustrated in [34]

pass fails to improve solution quality. Pseudo-code for the FM heuristic is given in algorithm 2.2.

The FM algorithm has 3 main components (1) computation of initial gain values at the beginning of a pass; (2) the retrieval of the best-gain (feasible) move; and (3) the update of all affected gain values after a move is made. One contribution of Fiduccia and Mattheyses lies in observing that circuit hypergraphs are sparse, and any move's gain is bounded between plus and minus the maximal vertex degree G_{max} in the hypergraph (times the maximal hyperedge weight, if weights are used). This allows prioritizing moves by their gains. All affected gains can be updated in amortized-constant time, giving overall linear complexity per pass [34]. All moves with the same gain are stored in a linked list representing a "gain bucket". Figure. 2.21 presents the gain bucket list structure. It is important to note that some gains G may be negative, and as such, FM performs hill-climbing and is not strictly greedy.

Multilevel Partitioning

The multilevel hypergraph partitioning framework was successfully verified by [31, 48, 49] and leads to the best known partitioning results ever since. The main advantage of multilevel partitioning over flat partitioners is its ability to search the solution space more effectively by spending comparatively more effort on smaller coarsened hypergraphs. Good coarsening algorithms allow for high correlation between good partitioning for coarsened hypergraphs and good partitioning for the initial hypergraph. Therefore, a thorough search at the top of the multilevel hierarchy is worthwhile because it is relatively inexpensive when compared to flat partitioning of the original hypergraph, but can still preserve most of the possible improvement.

The result is an algorithmic framework with both improved runtime and solution quality over a completely flat approach. Pseudo-code for an implementation of the multilevel partitioning framework is given in algorithm 2.3.

```
level = 0;
hierarchy[level] = hypergraph;
min_vertices = 200;
while hierarchy[level].vertex_count() > min_vertices do
    next_level = cluster(hierarchy[level]);
    level = level + 1;
    hierarchy[level] = next_level;
endw
partitioning[level] = a random initial solution for top-level hypergraph;
FM(hierarchy[level], partitioning[level]);
while level>0 do
    level = level - 1;
    partitioning[level] = project(partitioning[level+1], hierarchy[level]);
    FM(hierarchy[level], partitioning[level]);
endw
```

Algorithm 2.3 Pseudo-code for the Multilevel Partitioning Algorithm [38]

As illustrated in Fig. 2.22, multilevel partitioning consists of 3 main components: clustering, top-level partitioning and refinement or "uncoarsening". During clustering, hypergraph vertices are combined into clusters based on connectivity, leading to a smaller, clustered hypergraph. This step is repeated until obtaining only several hundred clusters and a hierarchy of clustered hypergraphs. We describe this hierarchy, as shown in Fig. 2.22, with the smaller hypergraphs being "higher" and the larger hypergraphs being "lower". The smallest (top-level) hypergraph is partitioned with a very fast initial solution generator and improved iteratively, for example, using the FM algorithm. The resulting partitioning is then interpreted as a solution for the next hypergraph in the hierarchy. During the refinement stage, solutions are projected from one level to the next and improved iteratively. Additionally, the hMETIS partitioning program [49] introduced several new heuristics that are incorporated into their multilevel partitioning implementation and are reportedly performance critical.

2.5.4 Placement

Placement algorithms determine which logic block within an FPGA should implement the corresponding logic block (instance) required by the circuit. The optimization goals consist in placing connected logic blocks close together to minimize the required wiring (wire length-driven placement), and sometimes to place blocks to balance the wiring density across the FPGA (routability-driven placement) or to maximize circuit speed (timing-driven placement). The 3 major classes of

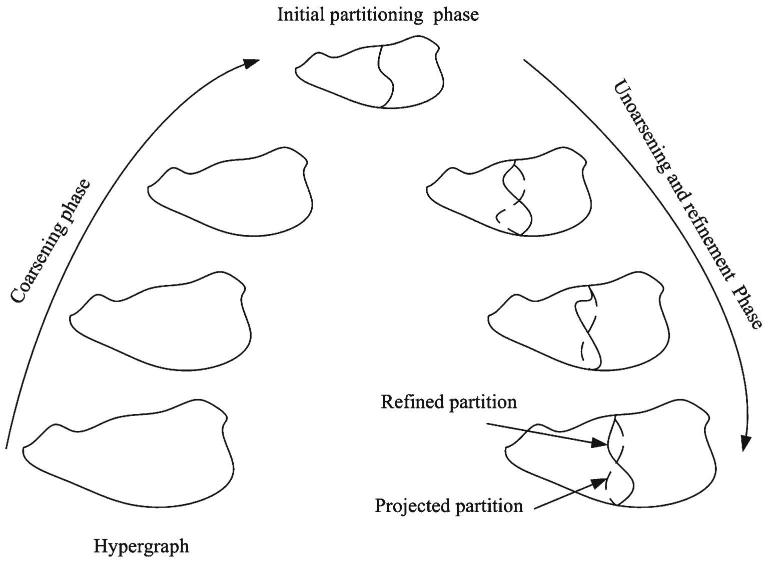


Fig. 2.22 Multilevel hypergraph bisection

placers in use today are min-cut (Partitioning-based) [6, 40], analytic [32, 53] which are often followed by local iterative improvement, and simulated annealing based placers [37, 105]. To investigate architectures fairly we must make sure that our CAD tools are attempting to use every FPGA's feature. This means that the optimization approach and goals of the placer may change from architecture to architecture. Partitioning and simulated annealing approaches are the most common and used in FPGA CAD tools. Thus we focus on both techniques in the sequel.

2.5.4.1 Simulated Annealing Based Approach

Simulated annealing mimics the annealing process used to cool gradually molten metal to produce high-quality metal objects [105]. Pseudo-code for a generic simulated annealing-based placer is shown in algorithm 2.4. A cost function is used to evaluate the quality of a given placement of logic blocks. For example, a common cost function in wirelength-driven placement is the sum over all nets of the half perimeter of their bounding boxes. An initial placement is created by assigning logic blocks randomly to the available locations in the FPGA. A large number of moves, or local improvements are then made to gradually improve the placement. A logic block is selected at random, and a new location for it is also selected randomly. The change in cost function that results from moving the selected logic block to the proposed new location is computed. If the cost decreases, the move is always accepted and the block is moved. If the cost increases, there is still a chance to accept the move, even though it makes the placement worse. This probability of acceptance is

```
S = RandomPlacement();
T = InitialTemperature();
$R_{limit} = InitialR_{limit}$;
while ExitCriterion() == false do
    while InnerLoopCriterion() == false do
        $S_{new} = GenerateViaMove(S, R_{limit})$;
        $\Delta C = Cost(S_{new}) - Cost(S)$;
        r = random(0,1);
        if $r < e^{-\frac{\Delta C}{T}}$ then
            $S = S_{new}$;
        endif
    endw
    T = UpdateTemp();
    $R_{limit} = UpdateR_{limit}()$;
endw
```

Algorithm 2.4 Generic Simulated Annealing-based Placer [22]

given by $e^{-\frac{\Delta C}{T}}$, where ΔC is the change in cost function, and T is a parameter called temperature that controls probability of accepting moves that worsen the placement. Initially, T is high enough so almost all moves are accepted; it is gradually decreased as the placement improves, in such a way that eventually the probability of accepting a worsening move is very low. This ability to accept hill-climbing moves that make a placement worse allows simulated annealing to escape local minima of the cost function.

The R_{limit} parameter in algorithm 2.4 controls how close are together blocks must be to be considered for swapping. Initially, R_{limit} is fairly large, and swaps of blocks far apart on a chip are more likely. Throughout the annealing process, R_{limit} is adjusted to try to keep the fraction of accepted moves at any temperature close to 0.44. If the fraction of moves accepted, α, is less than 0.44, R_{limit} is reduced, while if α is greater than 0.44, R_{limit} is increased.

In [22], the objective cost function is a function of the total wirelength of the current placement. The wirelength is an estimate of the routing resources needed to completely route all nets in the netlist. Reductions in wirelength mean fewer routing wires and switches are required to route nets. This point is important because routing resources in an FPGA are limited. Fewer routing wires and switches typically are also translated into reductions of the delay incurred in routing nets between logic blocks. The total wirelength of a placement is estimated using a semi-perimeter metric, and is given by Eq. 2.2. N is the total number of nets in the netlist, $bbx(i)$ is the horizontal span of net i, $bby(i)$ is its vertical span, and $q(i)$ is a correction factor. Figure 2.23 illustrates the calculation of the horizontal and vertical spans of a hypothetical net that has 6 terminals.

$$WireCost = \sum_{i=1}^{N} q(i) \times (bb_x(i) + bb_y(i)) \tag{2.2}$$

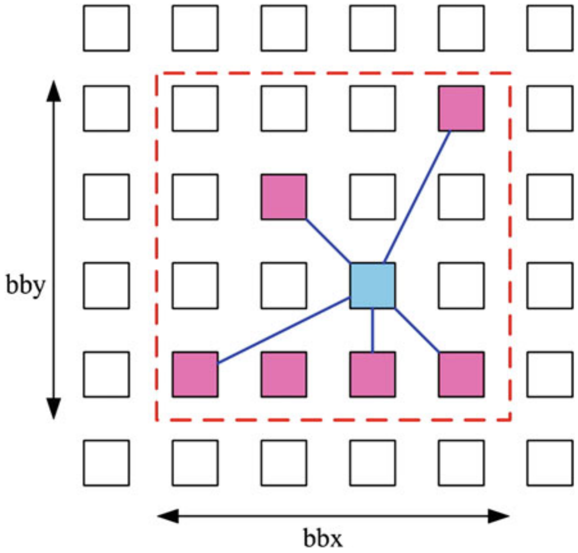


Fig. 2.23 Bounding box of a hypothetical 6-terminal net [22]

The temperature decrease rate, the exit criterion for terminating the anneal, the number of moves attempted at each temperature (InnerLoopCriterion), and the method by which potential moves are generated are defined by the annealing schedule. An efficient annealing schedule is crucial to obtain good results in a reasonable amount of CPU time. Many proposed annealing schedules are "fixed" schedules with no ability to adapt to different problems. Such schedules can work well within the narrow application range for which they are developed, but their lack of adaptability means they are not very general. In [86] authors propose an "adaptive" annealing schedule based on statistics computed during the anneal itself. Adaptive schedules are widely used to solve large scale optimization problems with many variables.

2.5.4.2 Partitioning Based Approach

Partitioning-based placement methods, are based on graph partitioning algorithms such as the Fiduccia-Mattheyses (FM) algorithm [34], and Kernighan Lin (KL) algorithm [6]. Partitioning-based placement are suitable to Tree-based FPGA architectures. The partitioner is applied recursively to each hierarchical level to distribute netlist cells between clusters. The aim is to reduce external communications and to collect highly connected cells into the same cluster.

The partitioning-based placement is also used in the case of Mesh-based FPGA. The device is divided into two parts, and a circuit partitioning algorithm is applied to determine the adequate part where a given logic block must be placed to minimize the number of cuts in the nets that connect the blocks between partitions, while leaving highly-connected blocks in one partition.

A divide-and-conquer strategy is used in these heuristics. By partitioning the problem into sub-parts, a drastic reduction in search space can be achieved. On the whole, these algorithms perform in the top-down manner, placing blocks in the general regions which they should belong to. In the Mesh FPGA case, partitioning-based placement algorithms are good from a "global" perspective, but they do not actually attempt to minimize wirelength. Therefore, the solutions obtained are sub-optimal in terms of wirelength. However, these classes of algorithms run very fast. They are normally used in conjunction with other search techniques for further quality improvement. Some algorithms [130] and [95] combine multi-level clustering and hierarchical simulated annealing to obtain ultra-fast placement with good quality. In the following chapters, the partitioning-based placement approach will be used only for Tree-based FPGA architectures.

2.5.5 Routing

The FPGA routing problem consists in assigning nets to routing resources such that no routing resource is shared by more than one net. $Pathfinder$ [80] is the current, state-of-the-art FPGA routing algorithm. $Pathfinder$ operates on a directed graph abstraction $G(V, E)$ of the routing resources in an FPGA. The set of vertices V in the graph represents the IO terminals of logic blocks and the routing wires in the interconnect structure. An edge between two vertices represents a potential connection between them. Figure 2.24 presents a part of a routing graph in a Mesh-based interconnect.

Given this graph abstraction, the routing problem for a given net is to find a directed tree embedded in G that connects the source terminal of the net to each of its sink terminals. Since the number of routing resources in an FPGA is limited, the goal of finding unique, non-intersecting trees for all the nets in a netlist is a difficult problem.

$Pathfinder$ uses an iterative, negotiation-based approach to successfully route all the nets in a netlist. During the first routing iteration, nets are freely routed without paying attention to resource sharing. Individual nets are routed using $Dijkstra$'s shortest path algorithm [111]. At the end of the first iteration, resources may be congested because multiple nets have used them. During subsequent iterations, the cost of using a resource is increased, based on the number of nets that share the resource, and the history of congestion on that resource. Thus, nets are made to negotiate for routing resources. If a resource is highly congested, nets which can use lower congestion alternatives are forced to do so. On the other hand, if the alternatives are more congested than the resource, then a net may still use that resource.

The cost of using a routing resource n during a routing iteration is given by Eq. 2.3.

$$c_n = (b_n + h_n) \times p_n \tag{2.3}$$

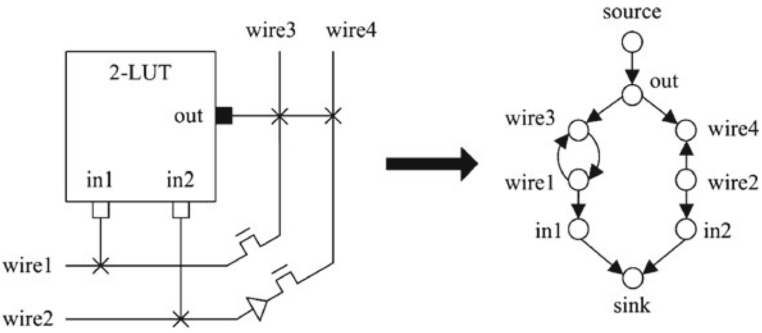


Fig. 2.24 Modeling FPGA architecture as a directed graph [22]

b_n is the base cost of using the resource n, h_n is related to the history of congestion during previous iterations, and p_n is proportional to the number of nets sharing the resource in the current iteration. The p_n term represents the cost of using a shared resource n, and the h_n term represents the cost of using a resource that has been shared during earlier routing iterations. The latter term is based on the intuition that a historically congested node should appear expensive, even if it is slightly shared currently. Cost functions and routing schedule were described in details in [22]. The Pseudo-code of the $Pathfinder$ routing algorithm is presented in algorithm 2.5.

```
Let: $RT_i$ be the set of nodes in the current routing of net i
while shared resources exist do
    /*Illegal routing*/
    foreach net, i do
        rip-up routing tree $RT_i$;
        $RT(i) = s_i$;
        foreach sink $t_{ij}$ do
            Initialize priority queue PQ to $RT_i$ at cost 0;
            while sink $t_{ij}$ not found do
                Remove lowest cost node m from PQ;
                foreach fanout node n of node m do
                    Add n to PQ at $PathCost(n) = c_n + PathCost(m)$;
                endfch
            endw
            foreach node n in path $t_{ij}$ to $s_i$ do
                /*backtrace*/
                Update $c_n$;
                Add n to $RT_i$;
            endfch
        endfch
    endfch
    update $h_n$ for all n;
endw
```

Algorithm 2.5 Pseudo-code of the $Pathfinder$ Routing Algorithm [80]

An important measure of routing quality produced by an FPGA routing algorithm is the critical path delay. The critical path delay of a routed netlist is the maximum delay of any combinational path in the netlist. The maximum frequency at which a netlist can be clocked has an inverse relationship with critical path delay. Thus, larger critical path delays slow down the operation of netlist. Delay information is incorporated into $Pathfinder$ by redefining the cost of using a resource n (Eq. 2.4).

$$c_n = A_{ij} \times d_n + (1 - A_{ij}) \times (b_n + h_n) \times p_n \tag{2.4}$$

The c_n term is from Eq. 2.3, d_n is the delay incurred in using the resource, and A_{ij} is the criticality given by Eq. 2.5.

$$A_{ij} = \frac{D_{ij}}{D_{max}} \tag{2.5}$$

D_{ij} is the maximum delay of any combinational path going through the source and sink terminals of the net being routed, and D_{max} is the critical path delay of the netlist. Equation 2.4 is formulated as a sum of two cost terms. The first term in the equation represents the delay cost of using resource n, while the second term represents the congestion cost. When a net is routed, the value of A_{ij} determines whether the delay or the congestion cost of a resource dominates. If a net is near critical (i.e. its A_{ij} is close to 1), then congestion is largely ignored and the cost of using a resource is primarily determined by the delay term. If the criticality of a net is low, the congestion term in Eq. 2.4 dominates, and the route found for the net avoids congestion while potentially incurring delay.

$Pathfinder$ has proved to be one of the most powerful FPGA routing algorithms to date. The negotiation-based framework that trades off delay for congestion is an extremely effective technique for routing signals on FPGAs. More importantly, $Pathfinder$ is a truly architecture-adaptive routing algorithm. The algorithm operates on a directed graph abstraction of an FPGA's routing structure, and can thus be used to route netlists on any FPGA that can be represented as a directed routing graph.

2.5.6 Timing Analysis

Timing analysis [99] is used for two basic purposes:

- To determine the speed of circuits which have been completely placed and routed,
- To estimate the slack [68] of each source-sink connection during routing (placement and other parts of the CAD flow) in order to decide which connections must be made via fast paths to avoid slowing down the circuit.

First the circuit under consideration is presented as a directed graph. Nodes in the graph represent input and output pins of circuit elements such as LUTs, registers, and I/O pads. Connections between these nodes are modeled with edges in the graph. Edges are added between the inputs of combinational logic Blocks (LUTs) and their outputs. These edges are annotated with a delay corresponding to the physical delay between the nodes. Register input pins are not joined to register output pins. To determine the delay of the circuit, a breadth first traversal is performed on the graph starting at sources (input pads, and register outputs). Then the arrival time, $T_{arrival}$, at all nodes in the circuit is computed with the following equation:

$$T_{arrival}(i) = \max_{j \in fanin(i)}\{T_{arrival}(j) + delay(j, i)\}$$

where node i is the node currently being computed, and $delay(j, i)$ is the delay value of the edge joining node j to node i. The delay of the circuit is then the maximum arrival time, D_{max}, of all nodes in the circuit.
To guide a placement or routing algorithm, it is useful to know how much delay may be added to a connection before the path that the connection is on becomes critical. The amount of delay that may be added to a connection before it becomes critical is called the slack of that connection. To compute the slack of a connection, one must compute the required arrival time, $T_{required}$, at every node in the circuit. We first set the $T_{required}$ at all sinks (output pads and register inputs) to be D_{max}. Required arrival time is then propagated backwards starting from the sinks with the following equation:

$$T_{required}(i) = \min_{j \in fanout(i)}\{T_{required}(j) - delay(j, i)\}$$

Finally, the slack of a connection (i, j) driving node, j, is defined as:

$$Slack(i, j) = T_{required}(j) - T_{arrival}(i) - delay(i, j)$$

2.5.7 Bitstream Generation

Once a netlist is placed and routed on an FPGA, bitstream information is generated for the netlist. This bitstream is programmed on the FPGA using a bitstream loader. The bitstream of a netlist contains information as to which SRAM bit of an FPGA be programmed to 0 or to 1. The bitstream generator reads the technology mapping, packing and placement information to program the SRAM bits of Look-Up Tables. The routing information of a netlist is used to correctly program the SRAM bits of connection boxes and switch boxes.

2.6 Research Trends in Reconfigurable Architectures

Until now in this chapter a detailed overview of logic architecture, routing architecture and software flow of FPGAs is presented. In this section, we highlight some of the disadvantages associated with FPGAs and further we describe some of the trends that are currently being followed to remedy these disadvantages. FPGA-based products are basically very effective for low to medium volume production as they are easy to program and debug, and have less NRE cost and faster time-to-market. All these major advantages of an FPGA come through their reconfigurability which makes them general purpose and field programmable. But, the very same reconfigurability is the major cause of its disadvantages; thus making it larger, slower and more power consuming than ASICs.

However, the continued scaling of CMOS and increased integration has resulted in a number of alternative architectures for FPGAs. These architectures are mainly aimed to improve area, performance and power consumption of FPGA architectures. Some of these propositions are discussed in this section.

2.6.1 Heterogeneous FPGA Architectures

Use of hard-blocks in FPGAs improves their logic density. Hard-Blocks, in FPGAs increase their density, performance and power consumption. There can be different types of hard-blocks like multipliers, adders, memories, floating point units and DSP blocks etc. In this regard, [19] have incorporated embedded floating-point units in FPGAs, [30] have developed virtual embedded block methodology to model arbitrary embedded blocks on existing commercial FPGAs. Here some of the academic and commercial architectures are presented that make use of hard-blocks to improve overall efficiency of FPGAs.

2.6.1.1 Versatile Packing, Placement and Routing VPR

Versatile Packing, Placement and Routing for FPGAs (commonly known as VPR) [14, 22, 120] is the most widely used academic mesh-based FPGA exploration environment. It allows to explore mesh-based FPGA architectures by employing an empirical approach. Benchmark circuits are mapped, placed and routed on a desired FPGA architecture. Later, area and delay of FPGAs are measured to decide best architectural parameters. Different CAD tools in VPR are highly optimized to ensure high quality results.

Earlier version of VPR supported only homogeneous achitectures [120]. However, the latest version of VPR known as VPR 5.0 [81] supports hard-blocks (such as multiplier and memory blocks) and single-driver routing wires. Hard-blocks are restricted to be in one grid width column, and that column can be composed of only similar type of blocks. The height of a hard-block is quantized and it must be an integral multiple of grid units. In case a block height is indivisible with the height of FPGA, some grid locations are left empty. Figure 2.25 illustrates a heterogeneous FPGA with 8 different kinds of blocks.

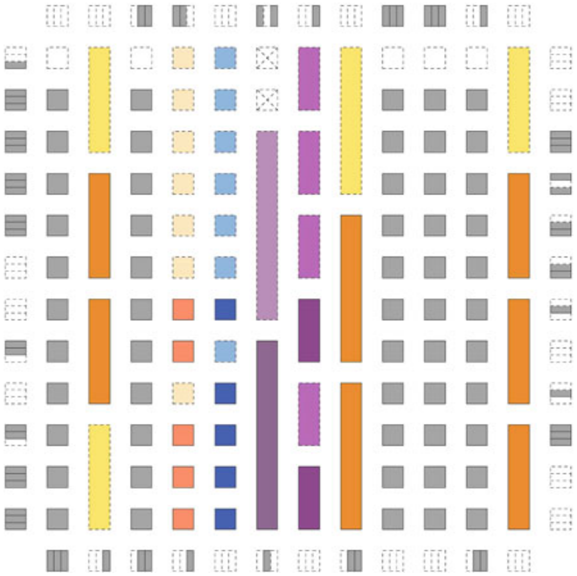


Fig. 2.25 A heterogeneous FPGA in VPR 5.0 [81]

2.6.1.2 Madeo, a Framework for Exploring Reconfigurable Architectures

Madeo [73] is another academic design suite for the exploration of reconfigurable architectures. It includes a modeling environment that supports multi-grained, heterogeneous architectures with irregular topologies. Madeo framework initially allows to model an FPGA architecture. The architecture characteristics are represented as a common abstract model. Once the architecture is defined, the CAD tools of Madeo are used to map a target netlist on the architecture. Madeo uses same placement and routing algorithms as used by VPR [120]. Along with placement and routing algorithms, it also embeds a bitstream generator, a netlist simulator, and a physical layout generator in its design suite. Madeo supports architectural prospection and very fast FPGA prototyping. Several FPGAs, including some commercial architectures (such as Xilinx Virtex family) and prospective ones (such as STMicro LPPGA) have been modeled using Madeo. The physical layout is produced as VHDL description.

2.6.1.3 Altera Architecture

Altera's Stratix IV [107] is an example of a commercial architecture that uses a heterogeneous mixture of blocks. Figure 2.26 shows the global architectural layout of Stratix IV. The logic structure of Stratix IV consists of LABs (Logic Array Blocks), memory blocks and digital signal processing (DSP) blocks. LABS are distributed symmetrically in rows and columns and are used to implement general purpose logic. The DSP blocks are used to implement full-precision multipliers of different

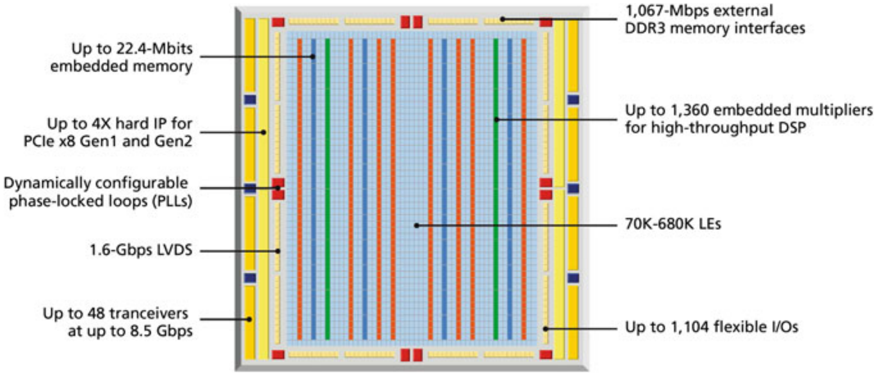

Up to 22.4-Mbits embedded memory
Up to 4X hard IP for PCIe x8 Gen1 and Gen2
Dynamically configurable phase-locked loops (PLLs)
1.6-Gbps LVDS
Up to 48 tranceivers at up to 8.5 Gbps
1,067-Mbps external DDR3 memory interfaces
Up to 1,360 embedded multipliers for high-throughput DSP
70K-680K LEs
Up to 1,104 flexible I/Os

Fig. 2.26 Stratix IV architectural elements

granularities. The memory blocks and DSP blocks are placed in columns at equal distance with one another. Input and Output (I/Os) are located at the periphery of architecture.

Logic array blocks (LABs) and adaptive logic modules (ALMs) provide the basic logic capacity for Stratix IV device. They can be used to configure logic functions, arithmetic functions, and register functions. Each LAB consists of ten ALMs, carry chains, arithmetic chains, LAB control signals, local interconnect, and register chain connection lines. The local interconnect connects the ALMs that are inside same LAB. The direct link allows a LAB to drive into the local interconnect of its left or right neighboring LAB. The register chain connects the output of ALM register to the adjacent ALM register in the LAB. A memory LAB (MLAB) is a derivative of LAB which can be either used just like a simple LAB, or as a static random access memory (SRAM). Each ALM in an MLAB can be configured as a 64 × 1, or 32 × 2 blocks, resulting in a configuration of 64 × 10 or 32 × 20 simple dual-port SRAM block. MLAB and LAB blocks always coexist as pairs in Stratix IV families.

The DSP blocks in Stratix IV are optimized for signal processing applications such as Finite Impulse Response (FIR), Infinite Impulse Response (IIR), Fast Fourier Transform functions (FFT) and encoders etc. Stratix IV device has two to seven columns of DSP blocks that can implement different operations like multiplication, multiply-add, multiply-accumulate (MAC) and dynamic arithmetic or logical shift functions. The DSP block supports different multiplication operations such as 9 × 9, 12 × 12, 18 × 18 and 36 × 36 multiplication operations. The Stratix IV devices contain three different sizes of embedded SRAMs. The memory sizes include 640-bit memory logic array blocks (MLABs), 9-Kbit M9K blocks, and 144-Kbit M144K blocks. The MLABs have been optimized to implement filter delay lines, small FIFO buffers, and shift registers. M9K blocks can be used for general purpose memory applications, and M144K are generally meant to store code for a processor, packet buffering or video frame buffering.

2.6.2 FPGAs to Structured Architectures

The ease of designing and prototyping with FPGAs can be exploited to quickly design a hardware application on an FPGA. Later, improvements in area, speed, power and volume production can be achieved by migrating the application design from FPGA to other technologies such as Structured-ASICs. In this regard, Altera provides a facility to migrate its Stratix IV based application design to HardCopy IV [56]. Altera gives provision to migrate FPGA-based applications to Structured-ASIC. Their Structured-ASIC is called as HardCopy [56]. The main theme is to design, test and even initially ship a design using an FPGA. Later, the application circuit that is mapped on the FPGA can be seamlessly migrated to HardCopy for high volume production. Their latest HardCopy-IV devices offer pin-to-pin compatibility with the Stratix IV prototype, making them exact replacements for the FPGAs. Thus, the same system board and softwares developed for prototyping and field trials can be retained, enabling the lowest risk and fastest time-to-market for high-volume production. Moreover, when an application circuit is migrated from Stratix IV FPGA prototype to Hardcopy-VI, the core logic performance doubles and power consumption reduces by half.

The basic logic unit of HardCopy is termed as HCell. It is similar to Stratix IV logic cell (LAB) in the sense that the fabric consists of a regular pattern which is formed by tiling one or more basic cells in a two dimensional array. However, the difference is that HCell has no configuration memory. Different HCell candidates can be used, ranging from fine-grained NAND gates to multiplexors and coarse-grained LUTs. An array of such HCells, and a general purpose routing network which interconnects them is laid down on the lower layers of the chip. Specific layers are then reserved to form via connections or metal lines which are used to customize the generic array into specific functionality. Figure 2.27 illustrates the correspondence between an FPGA and a compatible structured ASIC. There is a one to one layout-level correspondence between MRAMs, phase-lock loops (PLLs), embedded memories, transceivers, and I/O blocks. The soft-logic DSP multipliers and logic cell fabric of the FPGA are re-synthesized to structured ASIC fabric. However, they remain functionally and electrically equivalent in FPGAs and HardCopy ASICs.

Apart from Altera, there are several other companies that provide a solution similar to that of Altera. For example, the eASIC Nextreme [41] uses an FPGA-like design flow to map an application design on SRAM programmable LUTs, which are later interconnected through mask programming of few upper routing layers. Tierlogic [113] is a recently launched FPGA vendor that offers 3D SRAM-based TierFPGA devices for prototyping and early production. The same design solution can be frozen to a TierASIC device with one low-NRE custom mask for error-free transition to an ASIC implementation. The SRAM layer is placed on an upper 3D layer of TierFPGA. Once the TierFPGA design is frozen, the bitstream information is used to create a single custom mask metal layer that will replace the SRAM programming layer.

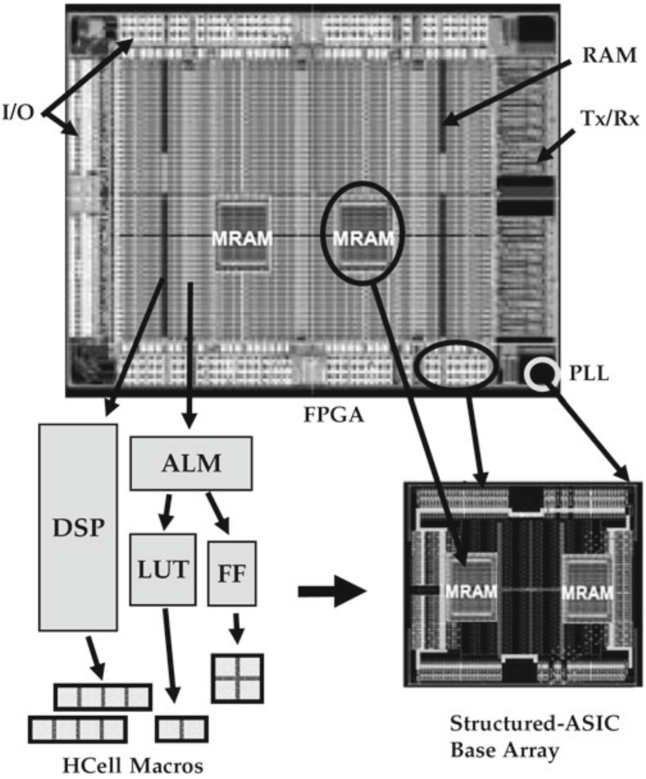


Fig. 2.27 FPGA/Structured-ASIC (HardCopy) Correspondence [59]

2.6.3 Configurable ASIC Cores

Configurable ASIC Core (cASIC) [35] is another example of reconfigurable devices that can implement a limited set of circuits which operate at mutually exclusive times. cASICs are intended as accelerator in domain-specific systems-on-a-chip, and are not designed to replace the entire ASIC-only chip. The host would execute software code, whereas compute-intensive sections can be executed on one or more cASICs. So, to execute the compute intensive sections, cASICs implement only data-path circuits and thus supports full-word blocks only (such as 16-bit wide multipliers, adders, RAMS, etc). Since the application domain of cASICs is more specific, they are significantly smaller than FPGAs. As hardware resources are shared between different netlists, cASICs are even smaller than the sum of the standard-cell based ASIC areas of individual circuits.

2.6.4 Processors Inside FPGAs

Considerable amount of FPGA area can be reduced by incorporating a microprocessor in an FPGA. A microprocessor can execute any less compute intensive task, whereas compute-intensive tasks can be executed on an FPGA. Similarly, a microprocessor based application can have huge speed-up gains if an FPGA is attached with it. An FPGA attached with a microprocessor can execute any compute intensive functionality as a customized hardware instruction. These advantages have compelled commercial FPGA vendors to provide microprocessor in their FPGAs so that complete system can be programmed on a single chip. Few vendors have integrated fixed hard processor on their FPGA (like AVR Processor integrated in Atmel FPSLIC [18] or PowerPC processors embedded in Xilinx Virtex-4 [126]). Others provide soft processor cores which are highly optimized to be mapped on the programmable resources of FPGA. Altera's Nios [90] and Xilinx's Microblaze [88] are soft processor meant for FPGA designs which allow custom hardware instructions. [96] have shown that considerable area gains can be achieved if these soft processors for FPGAs are optimized for particular applications. They have shown that unused instructions in a soft processor can be removed and different architectural tradeoffs can be selected to achieve on average 25% area gain for soft processors required for specific applications. Reconfigurable units can also be attached with microprocessors to achieve execution time speedup in software programs. [28, 70, 104] have incorporated a reconfigurable unit with microprocessors to achieve execution-time speedup.

2.6.5 Application Specific FPGAs

The type of logic blocks and the routing network in an FPGA can be optimized to gain area and performance advantages for a given application domain (controlpath-oriented applications, datapath-oriented applications, etc). These types of FPGAs may include different variety of desired hard-blocks, appropriate amount of flexibility required for the given application domain or bus-based interconnects rather than bit-based interconnects. Authors in [83] have presented a reconfigurable arithmetic array for multimedia applications which they call as CHESS. The principal goal of CHESS was to increase arithmetic computational density, to enhance the flexibility, and to increase the bandwidth and capacity of internal memories significantly beyond the capabilities of existing commercial FPGAs. These goals were achieved by proposing an array of ALUs with embedded RAMs where each ALU is 4-bit wide and supports 16 instructions. Similarly, authors in [42] present a coarse-grained, field programmable architecture for constructing deep computational pipelines. This architecture can efficiently implement applications related to media, signal processing, scientific computing and communications. Further, authors in [128] have used bus-based routing and logic blocks to improve density of FPGAs for datapath circuits. This is a partial multi-bit FPGA architecture that is designed to exploit the regularity that most of the datapath circuits exhibit.

2.6.6 Time-Multiplexed FPGAs

Time-multiplexed FPGAs increase the capacity of FPGAs by executing different portions of a circuit in a time-multiplexed mode [89, 114]. An application design is divided into different sub-circuits, and each sub-circuit runs as an individual context of FPGA. The state information of each sub-circuit is saved in context registers before a new context runs on FPGA. Authors in [114] have proposed a time-multiplexed FPGA architecture where a large circuit is divided into sub-circuits and each sub-circuit is sequentially executed on a time-multiplexed FPGA. Such an FPGA stores a set of configuration bits for all contexts. A context is shifted simply by using the SRAM bits dedicated to a particular context. The combinatorial and sequential outputs of a sub-circuit that are required by other sub-circuits are saved in context registers which can be easily accessed by sub-circuits at different times.

Time-Multiplexed FPGAs increase their capacity by actually adding more SRAM bits rather than more CLBs. These FPGAs increase the logic capacity by dynamically reusing the hardware. The configuration bits of only the currently executing context are active, the configuration bits for the remaining supported contexts are inactive. Intermediate results are saved and then shared with the contexts still to be run. Each context takes a micro-cycle time to execute one context. The sum of the micro-cycles of all the contexts makes one user-cycle. The entire time-multiplexed FPGA or its smaller portion can be configured to (i) execute a single design, where each context runs a sub-design, (ii) execute multiple designs in time-multiplexed modes, or (iii) execute statically only one single design. Tabula [109] is a recently launched FPGA vendor that provides time-multiplexed FPGAs. It dynamically reconfigures logic, memory, and interconnect at multi-GHz rates with a Spacetime compiler.

2.6.7 Asynchronous FPGA Architecture

Another alternative approach that has been proposed to improve the overall performance of FPGA architecture is the use of asynchronous design elements. Conventionally, digital circuits are designed for synchronous operation and in turn FPGA architectures have focused primarily on implementing synchronous circuits. Asynchronous designs are proposed to improve the energy efficiency of asynchronous FPGAs since asynchronous designs offer potentially lower energy as energy is consumed only when necessary. Also the asynchronous architectures can simplify the design process as complex clock distribution networks become unnecessary.

The first asynchronous FPGA was developed by [57]. It consisted the modified version of previously developed synchronous FPGA architecture. Its logic block was similar to the conventional logic block with added features of fast feedback and a latch that could be used to initialize an asynchronous circuit. Another asynchronous architecture was proposed in [112]. This architecture is designed specifically for dataflow applications. Its logic block is similar to that of synchronous architecture, along with it consists of units such as split unit which enables conditional forwarding of data and a merge unit that allows for conditional selection of data from different sources. An alternative to fully asynchronous design is a globally asynchronous, locally synchronous approach (GALS). This approach is used by [69] where authors have introduced a level of hierarchy into the FPGA architecture. Standard hard or soft synchronous logic blocks are grouped together to form large synchronous blocks and communication between these blocks is done asynchronously. More recently, authors in [131] have applied the GALS approach on Network on Chip architectures to improve the performance, energy consumption and the yield of future architectures in a synergistic manner.

It is clear that, despite each architecture offering its own benefits, a number of architectural questions remain unresolved for asynchronous FPGAs. Many architectures rely on logic blocks similar to those used for synchronous designs [57, 69] and, therefore, the same architectural issues such as LUT size, cluster size, and routing topology must be investigated. In addition to those questions, asynchronous FPGAs also add the challenge of determining the appropriate synchronization methodology.

2.7 Summary and Conclusion

In this chapter initially a brief introduction of traditional logic and routing architectures of FPGAs is presented. Later, different steps involved in the FPGA design flow are detailed. Finally various approaches that have been employed to reduce few disadvantages of FPGAs and ASICs, with or without compromising their major benefits are described. Figure 2.28 presents a rough comparison of different solutions used to reduce the drawbacks of FPGAs and ASICs. The remaining chapters of this book will focus on the exploration of tree-based FPGA architectures using hard-blocks, tree-based application specific Inflexible FPGAs (ASIF), and their automatic layout generation methods.

This book presents new environment for the exploration of tree-based heterogeneous FPGAs. This environment is used to explore different architecture techniques for tree-based heterogeneous FPGA architecture. This book also presents an optimized environment for mesh-based heterogeneous FPGA. Further, the environments of two architectures are evaluated through the experimental results that are obtained by mapping a number of heterogeneous benchmarks on the two architectures.

Altera [11] has proposed a new idea to prototype, test, and even ship initial few designs on an FPGA, later the FPGA based design can be migrated to Structured-ASIC (known as HardCopy). However, migration of an FPGA-based product to Structured-ASIC supports only a single application design. An ASIF retains this

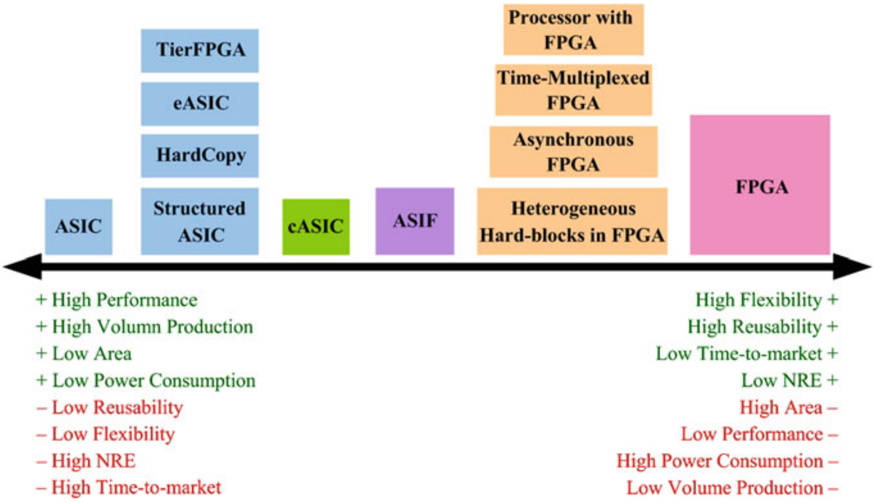


Fig. 2.28 Comparison of different solutions used to reduce ASIC and FPGA drawbacks

property, and can be a possible future extension for the migration of FPGA-based applications to Structured-ASIC. Thus when an FPGA-based product is in the final phase of its development cycle, and if the set of circuits to be mapped on the FPGA are known, the FPGA can be reduced to an ASIF for the given set of application designs. This book presents a new tree-based ASIF and a detailed comparison of tree-based ASIF is performed with mesh-based ASIF. This book also presents automatic layout generation techniques for domain-specific FPGA and ASIF architectures.

Springer

http://www.springer.com/978-1-4614-3593-8

Tree-based Heterogeneous FPGA Architectures
Application Specific Exploration and Optimization
Farooq, U.; Marrakchi, Z.; Mehrez, H.
2012, XVI, 188 p., Hardcover
ISBN: 978-1-4614-3593-8

Altera FLEX 8000 Block Diagram

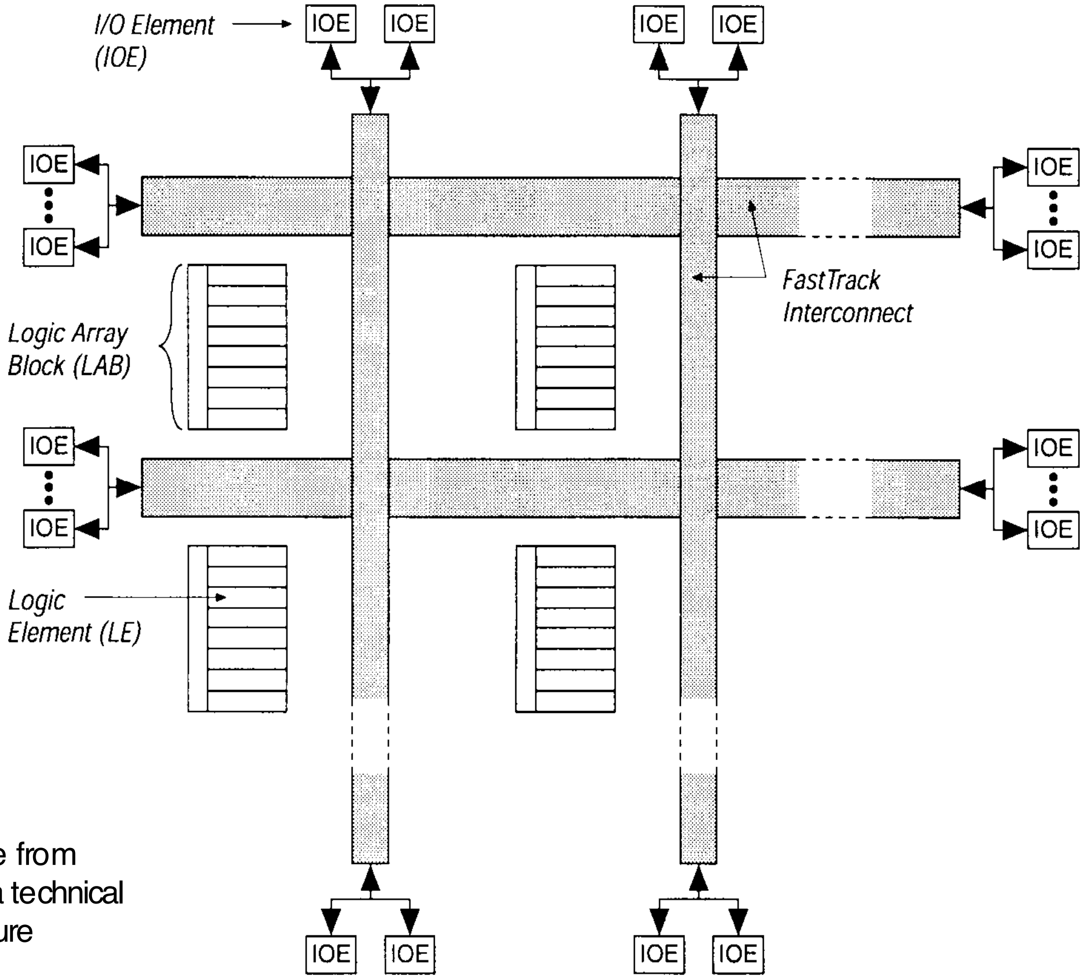


Figure from Altera technical literature

- FLEX 8000 chip contains 26–162 LABs
  - Each LAB contains 8 Logic Elements (LEs), so a chip contains 208–1296 LEs, totaling 2,500–16,000 usable gates
  - LABs arranged in rows and columns, connected by FastTrack Interconnect, with I/O elements (IOEs) at the edges

Altera FLEX 8000 Logic Array Block

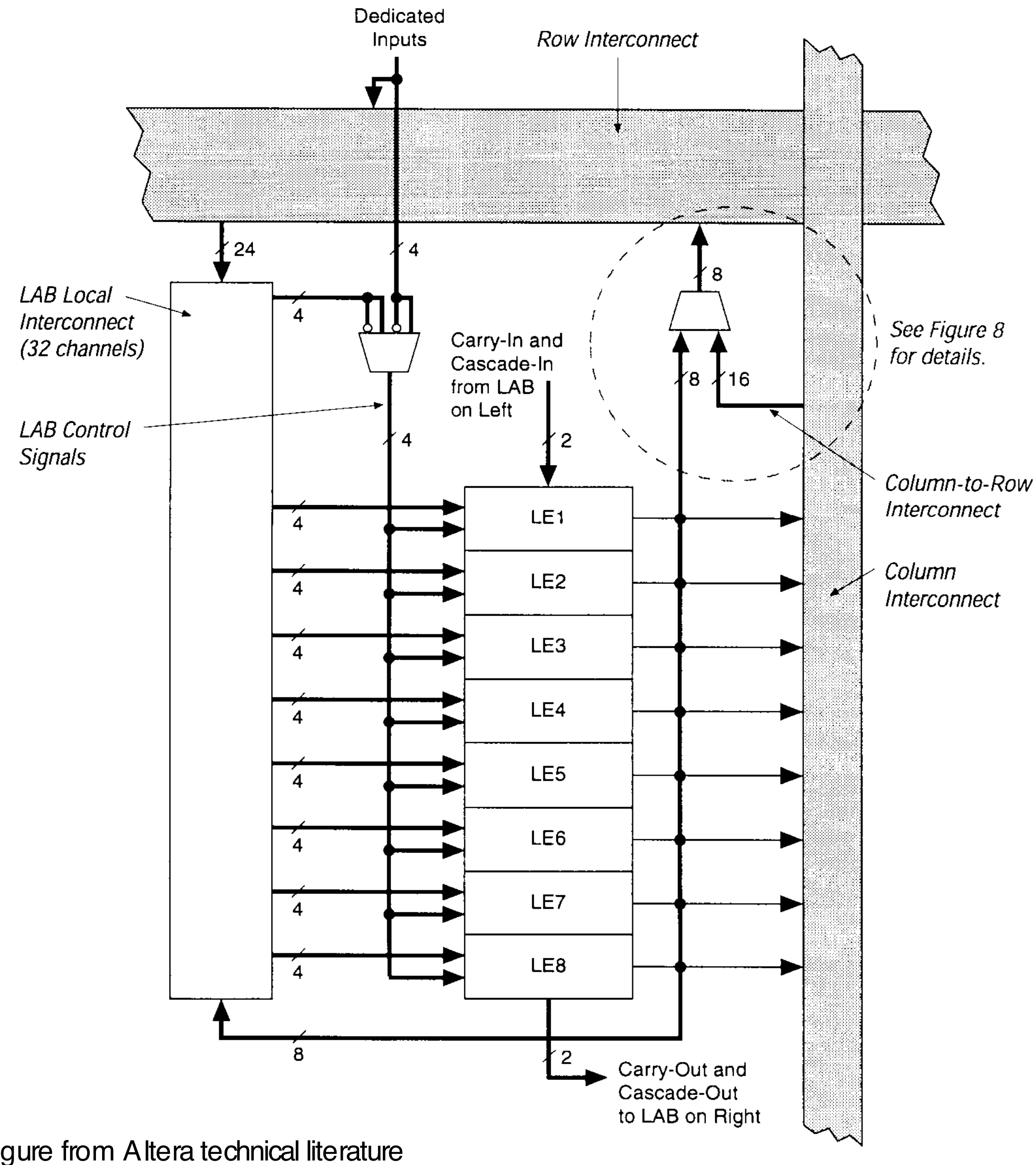

Dedicated Inputs
Row Interconnect
24
4
LAB Local Interconnect (32 channels)
4
Carry-In and Cascade-In from LAB on Left
8
See Figure 8 for details.
8
16
LAB Control Signals
4
2
Column-to-Row Interconnect
LE1
LE2
LE3
LE4
LE5
LE6
LE7
LE8
Column Interconnect
4
4
4
4
4
4
4
4
8
2
Carry-Out and Cascade-Out to LAB on Right

Figure from Altera technical literature

- LAB = 8 LEs, plus local interconnect, control signals, carry & cascade chains

Altera FLEX 8000 Logic Element

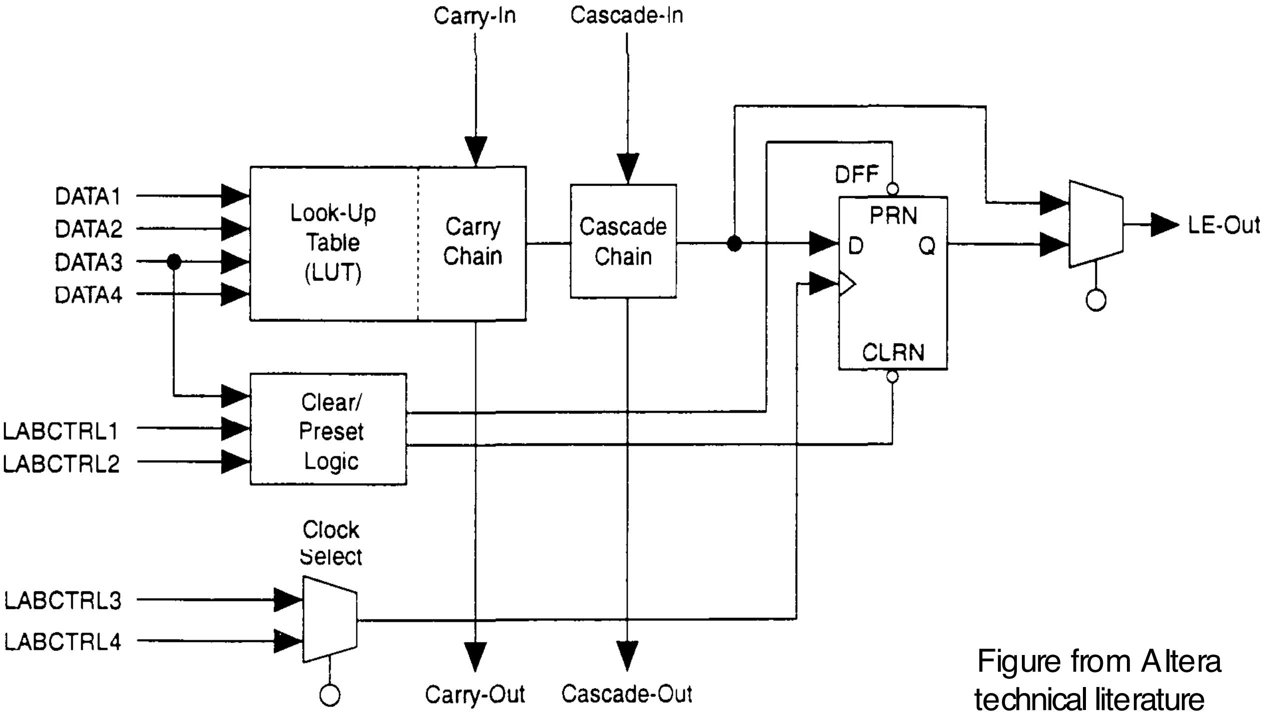


Figure from Altera technical literature

- Each Logic Element (LE) contains:
  - 4-input Look-Up Table (LUT)
    - Can produce any function of 4 variables
  - Programmable flip-flop
    - Can configure as D, T, JR, SR, or bypass
    - Has clock, clear, and preset signals that can come from dedicated inputs, I/O pins, or other LEs
  - Carry chain & cascade chain

Altera FLEX 8000 Carry Chain (Example: n-bit adder)

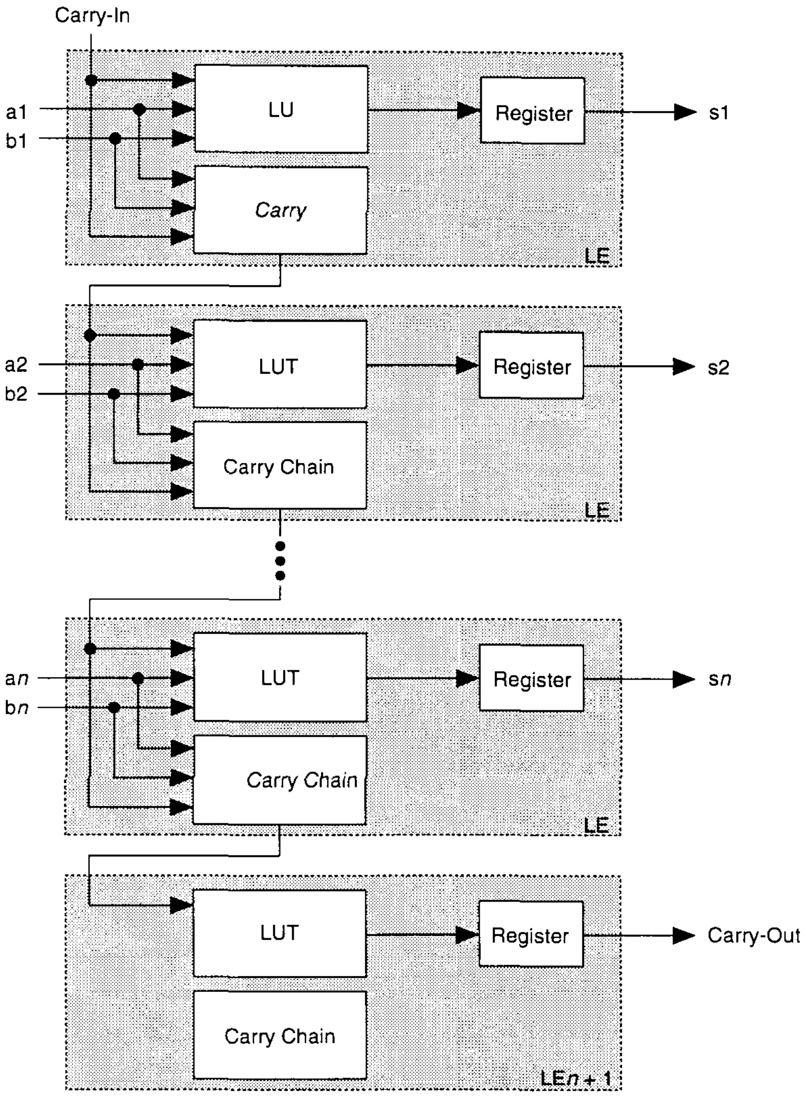


Figure from Altera technical literature

- *Carry chain* provides very fast (< 1ns) carry-forward between LEs
  - Feeds both LUT and next part of chain
  - Good for high-speed adders & counters

Altera FLEX 8000 Cascade Chain

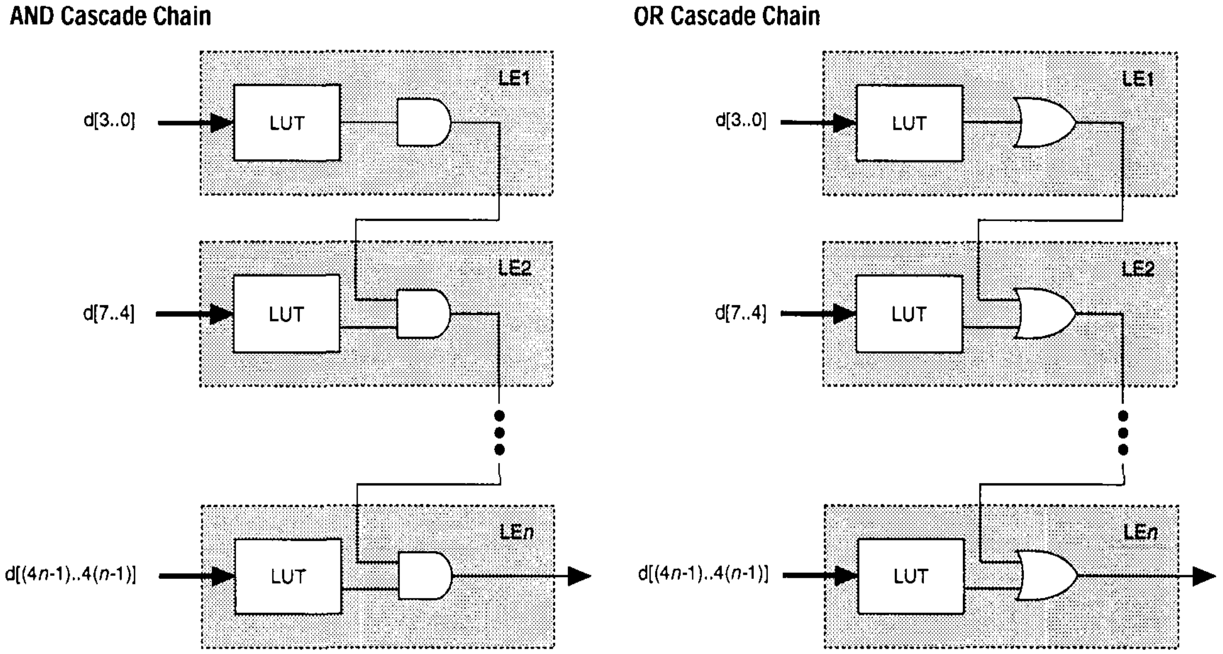
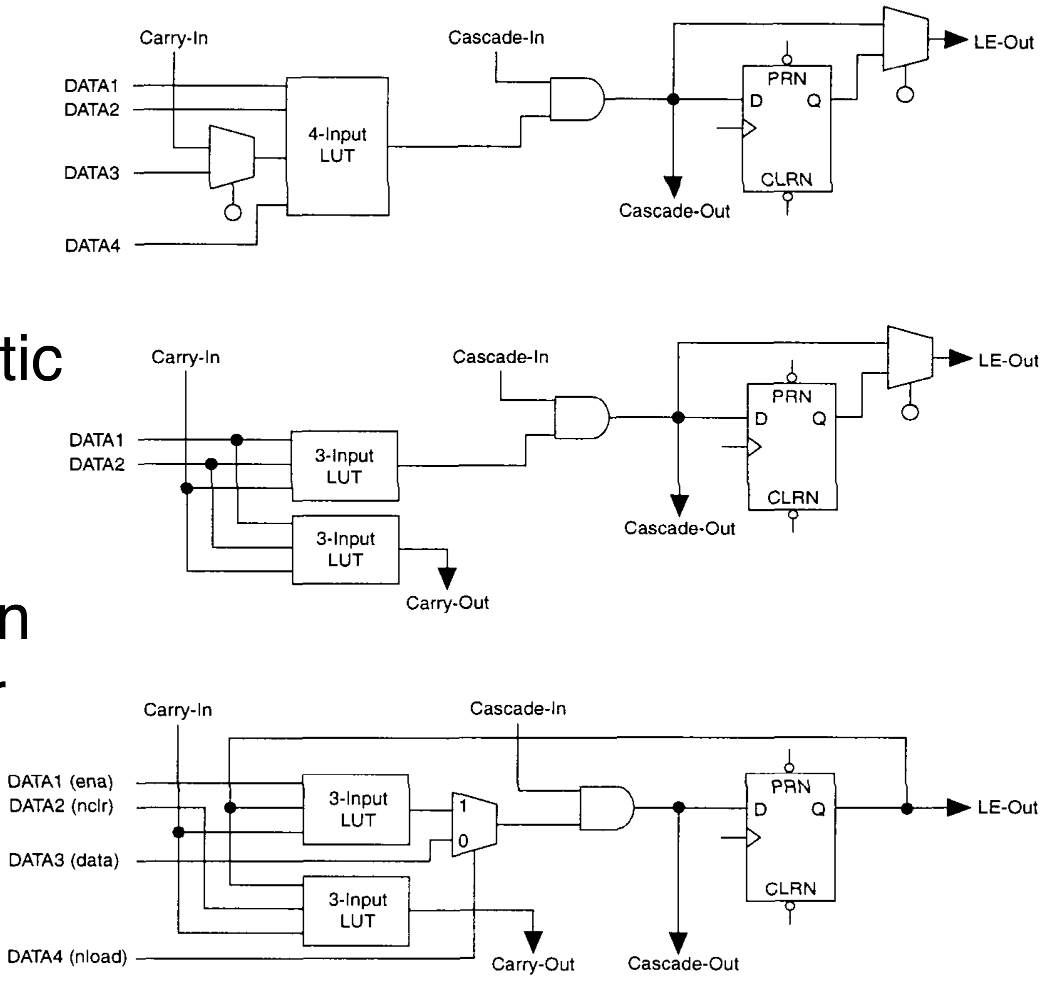


Figure from Altera technical literature

- *Cascade chain* provides wide fan-in
  - Adjacent LE's LUTs can compute parts of the function in parallel; cascade chain then serially connects intermediate values
  - Can use either a logical AND or a logical OR (using DeMorgan's theorem) to connect outputs of adjacent LEs
  - Each additional LE provides 4 more inputs to the width of the function

Altera FLEX 8000 LE Operating Modes



Normal

Arithmetic

Up/down Counter

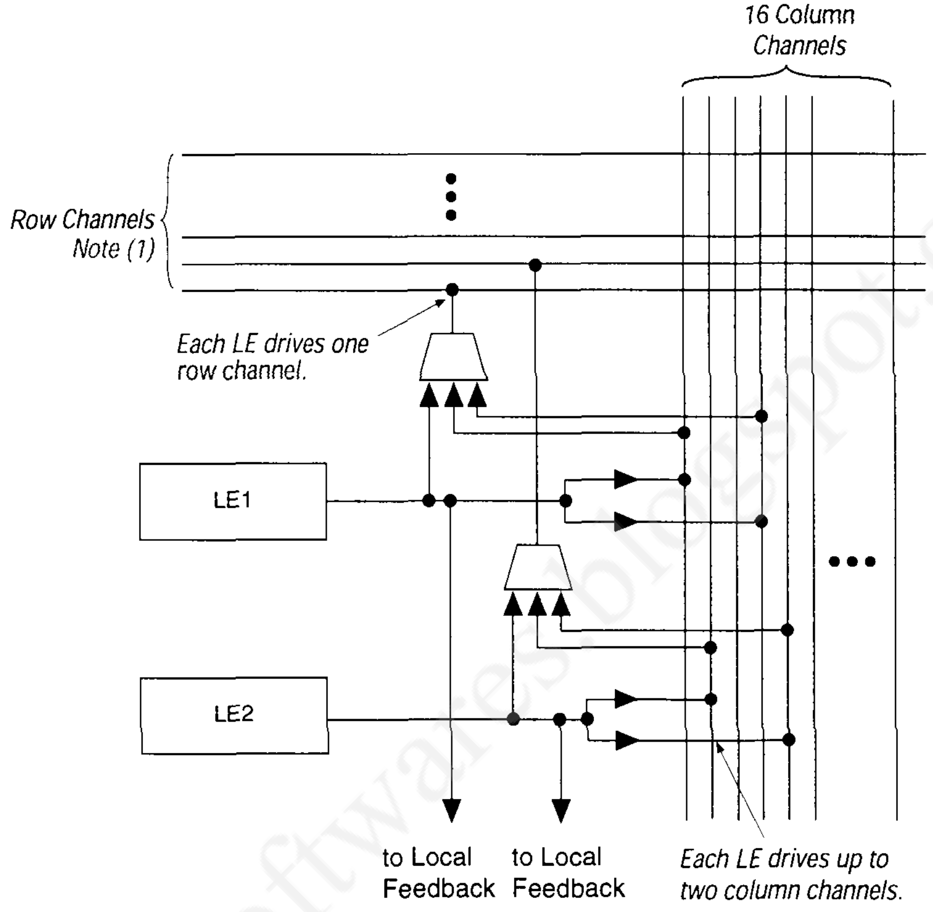
Figure from Altera technical literature

- Each mode uses LE resources differently
  - 7 out of 10 inputs (4 data from LAB local interconnect, feedback from register, and carry-in & cascade-in) go to specific destinations to implement the function
  - Remaining 3 provide clock, clear, and preset for register

Altera FLEX 8000 Operating Modes (cont.)

- Normal mode
  - Used for general logic applications, and wide decoding functions that can benefit from the cascade chain
- Arithmetic mode
  - Provides two 3-input LUTs to implement adders, accumulators, and comparators
    - One LUT provides a 3-bit function
    - Other LUT generates a carry bit
- Up/down counter mode
  - Provides counter enable, synchronous up / down control, and data loading options
  - Uses two 3-input LUTs
    - One LUT generates counter data
    - Other LUT generates fast carry bit
    - Use 2-to-1 multiplexer for synchronous data loading, clear and preset for asynchronous data loading

Altera FLEX 8000 FastTrack Interconnect



Note:
(1) See Table 4 for the number of row channels.

Figure from Altera technical literature

- Device-wide rows and columns
  - Each LE in LAB drives 2 column (total 16) channels, which connects… that column
  - Each LE in LAB drives 1 row channel, which connects to other LABs in that row
    - 3-to-1 muxs connect either LE outputs or column channels to row channels

Altera FLEX 8000 I/O Elements

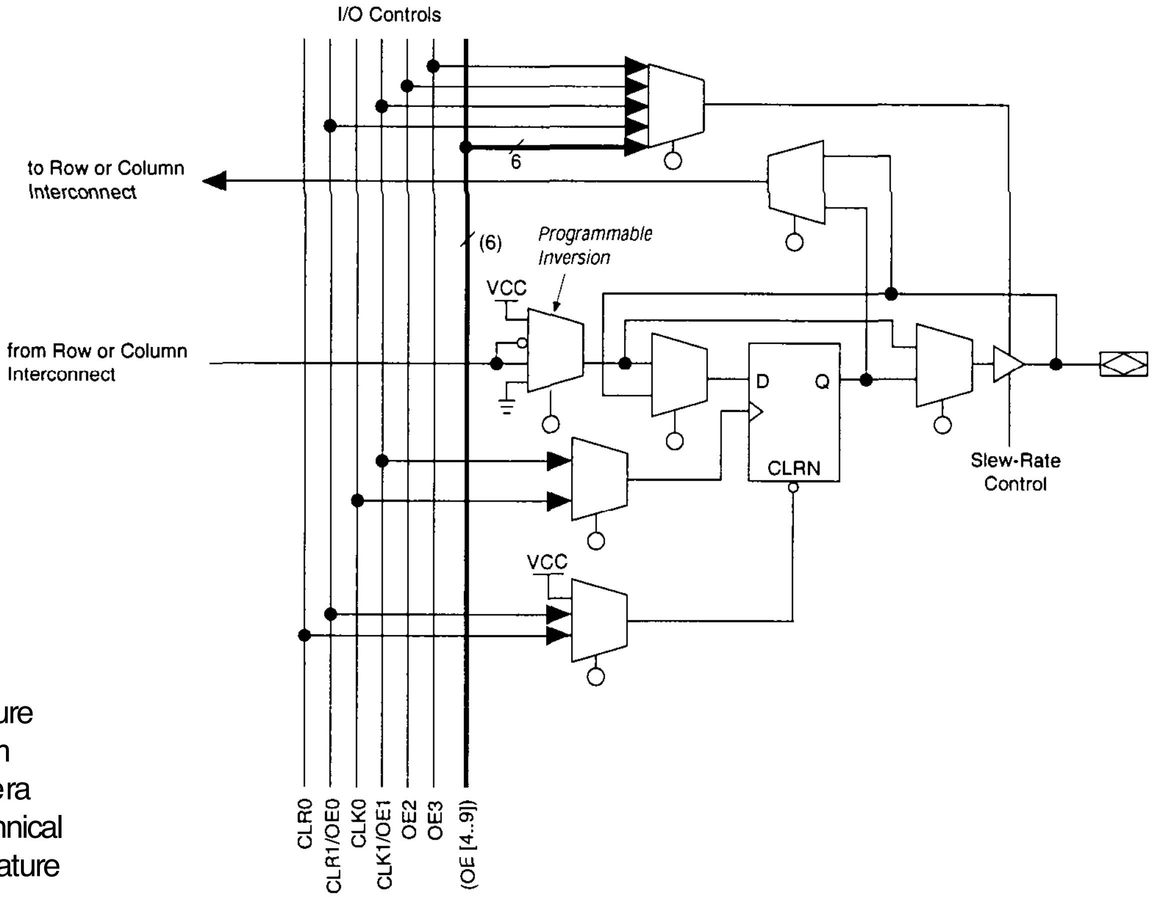


Figure from Altera technical literature

- Eight I/O Elements (IOEs) are at the end of each row and column
  - Some restrictions on how many row / column channels each IOE connects to
  - Contains a register that can be used for either input or output
    - Associated I/O pins can be used as either input, output, or bidirectional pins

Altera FLEX 8000 Configuration

- Loading the FLEX 8000's SRAM with programming information is called *configuration*, and takes about 100ms
  - After configuration, the device initializes itself (resets its registers, enables its I/O pins, and begins normal operation)
  - Configuration & initialization = command mode, normal operation = user mode
- Six configuration schemes are available:
  - Active serial — FLEX gives configuration EPROM clock signals (not addresses), keeps getting new values in sequence
  - Active parallel up, active parallel down — FLEX 8000 gives configuration EPROM sequence of addresses to read data from
  - Passive parallel synchronous, passive parallel asynchronous, passive serial — passively receives data from some host

Altera FLEX 8000 Block Diagram (Review)

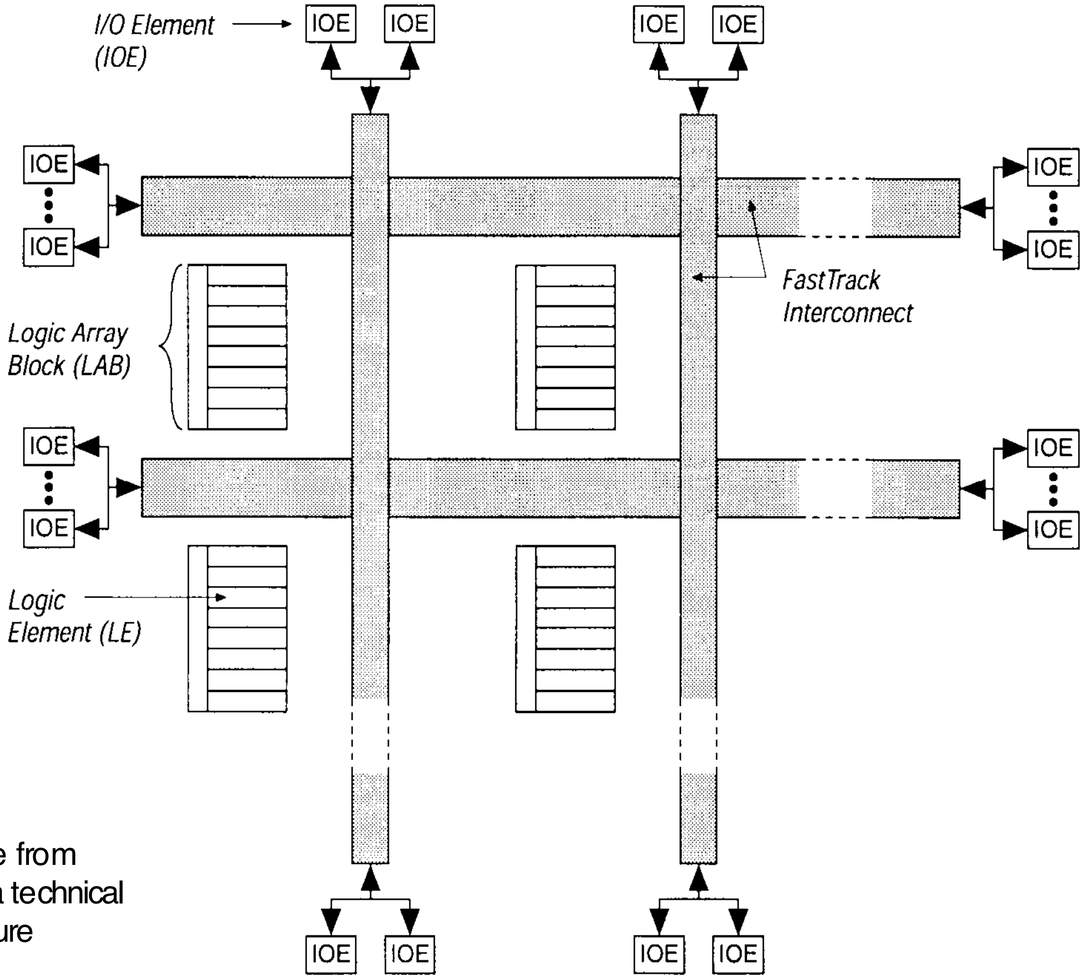


Figure from Altera technical literature

- FLEX 8000 chip contains 26–162 LABs
  - Each LAB contains 8 Logic Elements (LEs), so a chip contains 208–1296 LEs, totaling 2,500–16,000 usable gates
  - LABs arranged in rows and columns, connected by FastTrack Interconnect, with I/O elements (IOEs) at the edges

Altera FLEX 10K Block Diagram

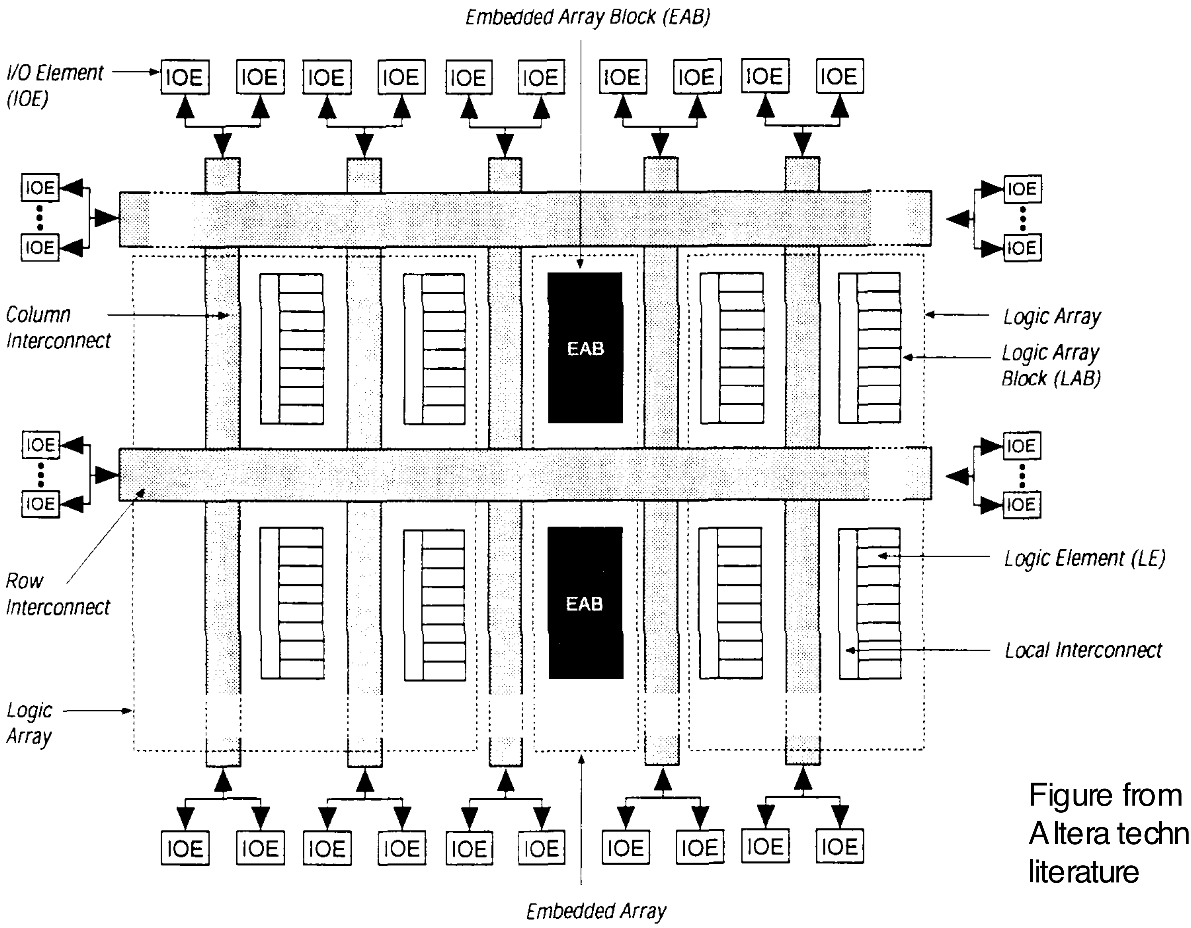


Figure from Altera technical literature

- FLEX 10K chip contains 72–1520 LABs
  - Each LAB contains 8 Logic Elements (LEs), so a chip contains 576–12,160 LEs, totaling 10,000–250,000 usable gates
- Each chip also contains 3–20 Embedded Array Blocks (EABs), which can provide 6,164–40,960 bits of RAM

Altera FLEX 10K
Embedded Array Blocks (EABs)

- Each chip contains 3–20 EABs, each of which can be used to implement either logic or memory

- When used to implement logic, an EAB can provide 100 to 600 gate equivalents (in contrast, a LAB provides 96 g.e.'s)
  - Provides a very large LUT
    - Very fast — faster than general logic, since it's only a single level of logic
    - Delay is predictable — each RAM block is not scattered throughout the chip as in some FPGAs
  - Can be used to create complex logic functions such as multipliers (e.g., a 4x4 multiplier with 8 inputs and 8 outputs), microcontrollers, large state machines, and DSPs
  - Each EAB can be used independently, or combined to implement larger functions

Altera FLEX 10K
Embedded Array Blocks (cont.)

- Using EABs to implement memory, a chip can have 6K–40K bits of RAM
  - Each EAB provides 2,048 bits of RAM, plus input and output registers
  - Can be used to implement synchronous RAM, ROM, dual-port RAM, or FIFO
  - Each EAB can be configured in the following sizes:
    - 256x8, 512x4, 1024x2, or 2048x1
  - To get larger blocks, combine multiple EABs:
    - Example: combine two 256x8 RAM blocks to form a 256x16 RAM block
    - Example: combine two 512x4 RAM blocks to form a 512x8 RAM block
    - Can even combine all EABs on the chip into one big RAM block
    - Can combine so as to form blocks up to 2048 words without impacting timing

Altera FLEX 10K
Embedded Array Blocks (cont.)

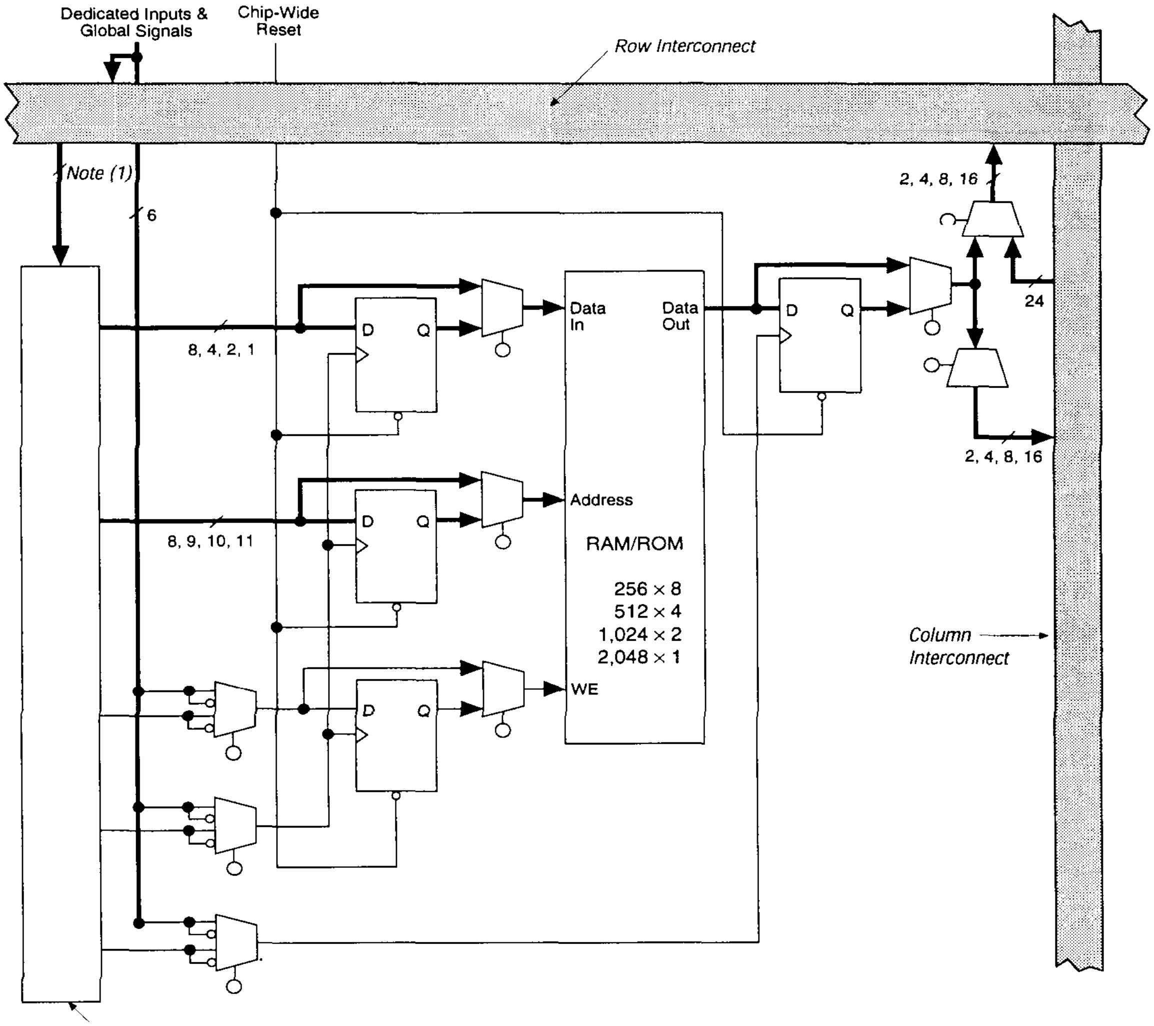


Figure from Altera technical literature

- EAB gets input from a row channel, and can output to up to 2 row channels and 2 column channels

- Input and output buffers are available

Altera APEX 20K Overview

- APEX 20K chip contains:
  - 256–3,456 LABs, each of which contains 10 Logic Elements (LEs), so a chip contains 2,560–51,840 Les, 162,000–2,391,552 usable gates
  - 16–216 Embedded System Blocks (EABs), each of which can provide 32,768–442,368 bits of memory
    - Can implement CAM, RAM, dual-port RAM, ROM, and FIFO
- Organization:
  - MultiCore architecture, combining LUT, product-terms, & memory in one structure
    - Designed for "system on a chip"
  - MegaLAB structures,each of which contains 16 LABs, one ESB, and a MegaLAB interconnect (for routing within the MegaLAB)
    - ESB provides product terms ***or*** memory

APEX LABs and Interconnect

- Logic Array Block (LAB)
  - 10 LEs
  - Interleaved local interconnect (each LE connects to 2 local interconnect, each local interconnect connects to 10 LEs)
    - Each LE can connect to 29 other Les through local interconnect
- Logic Element (LE)
  - 4-input LUT, carry chain, cascade chain, same as FLEX devices
  - Synchronous and asynchronous load and clear logic
- Interconnect
  - MegaLAB interconnect between 16 LABs, etc. inside each MegaLAB
  - FastTrack row and column interconnect between MegaLABs

APEX Embedded System Blocks (ESBs)

- Each ESB can act as a macrocell and provide product terms
  - Each ESB gets 32 inputs from local interconnect, from adjacent LAB or MegaLAB interconnect
  - In this mode, each ESB contains 16 macrocells, and each macrocell contains 2 product terms and a programmable register (parallel expanders also provided)
- Each ESB can also act as a memory block (dual-port RAM, ROM, FIFO, or CAM memory) configured in various sizes
  - Inputs from adjacent local interconnect, which can be driven from MegaLAB or FastTrack interconnect
  - Outputs to MegaLAB and FastTrack, some outputs to local interconnect

G.Venkata Hanuman
Asst. Prof

6. Low Power cmos Logic circuits

Power consumption :-

The Average Power consumption in cmos digital circuits can be expressed as

i) Dynamic (or) switching Power consumption

ii) short circuit Power consumption

iii) Leakage Power consumption

i) switching Power dissipation :-

when the output node voltage of a cmos Logic gate makes a logic transition. In digital cmos circuits, switching power is dissipated when energy is drawn from the power supply to charge up the output node capacitance.

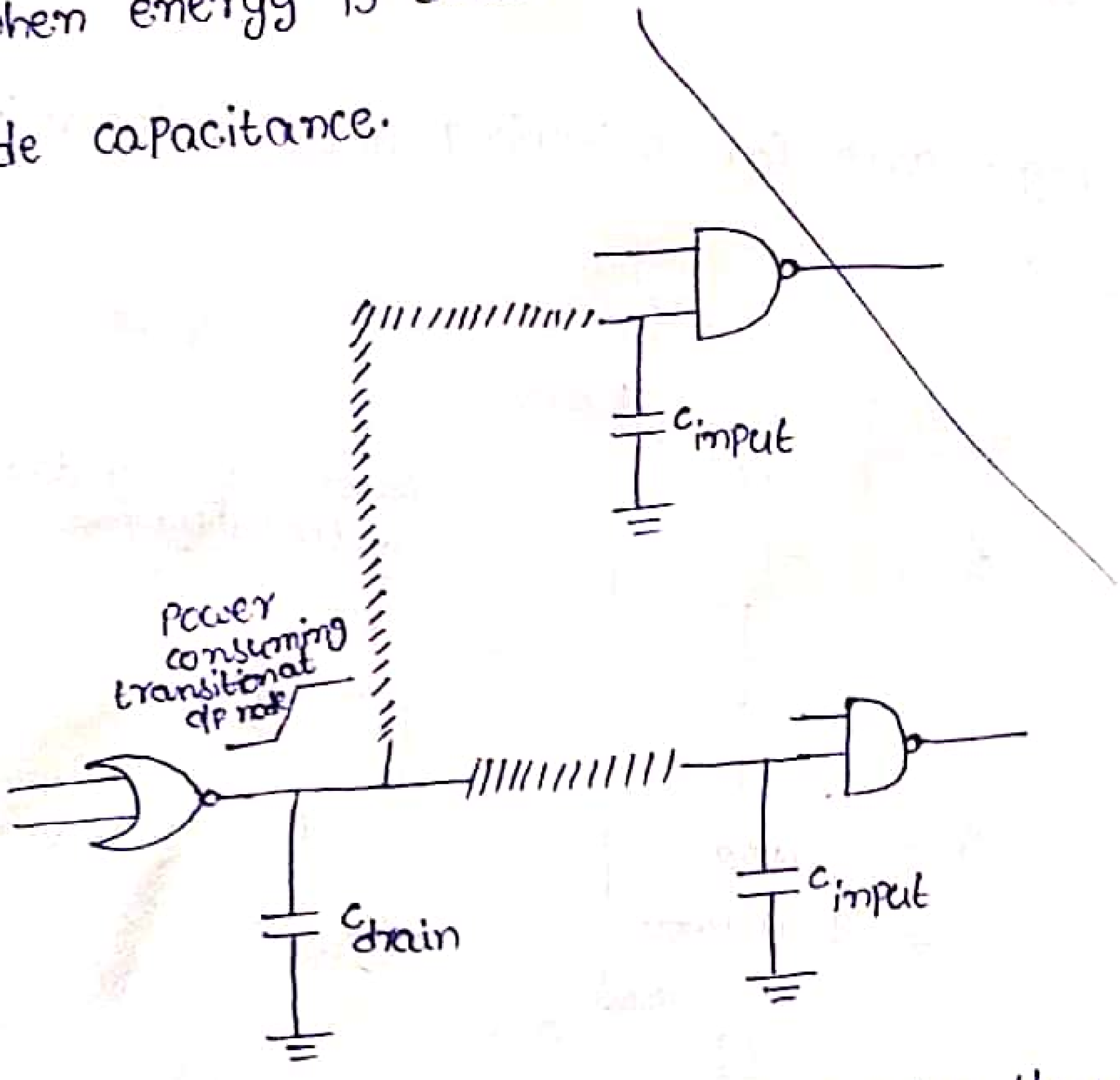


Fig:- NOR gate driving two NAND gates through Interconnection lines

During this charge up phase, output node voltage typically makes a full transition from 0 to V_{DD}.

During discharging (charge down phase), output voltage drops from V_{DD} to 0.

one half of the energy is drawn from the power supply is dissipated as heat in conducting pmos transitors. No energy is drawn from the power supply during charge-down phase.

From the figure, two Input NOR gate drives 2 NAND gates, through Interconnection lines. The total capacitive load at the output of NOR gate consists of

i) The output node capacitance of gate itself

ii) The total Interconnect capacitance

iii) Input capacitances of driven gates.

The cmos Logic gate for switching power caluclation as shown in fig.

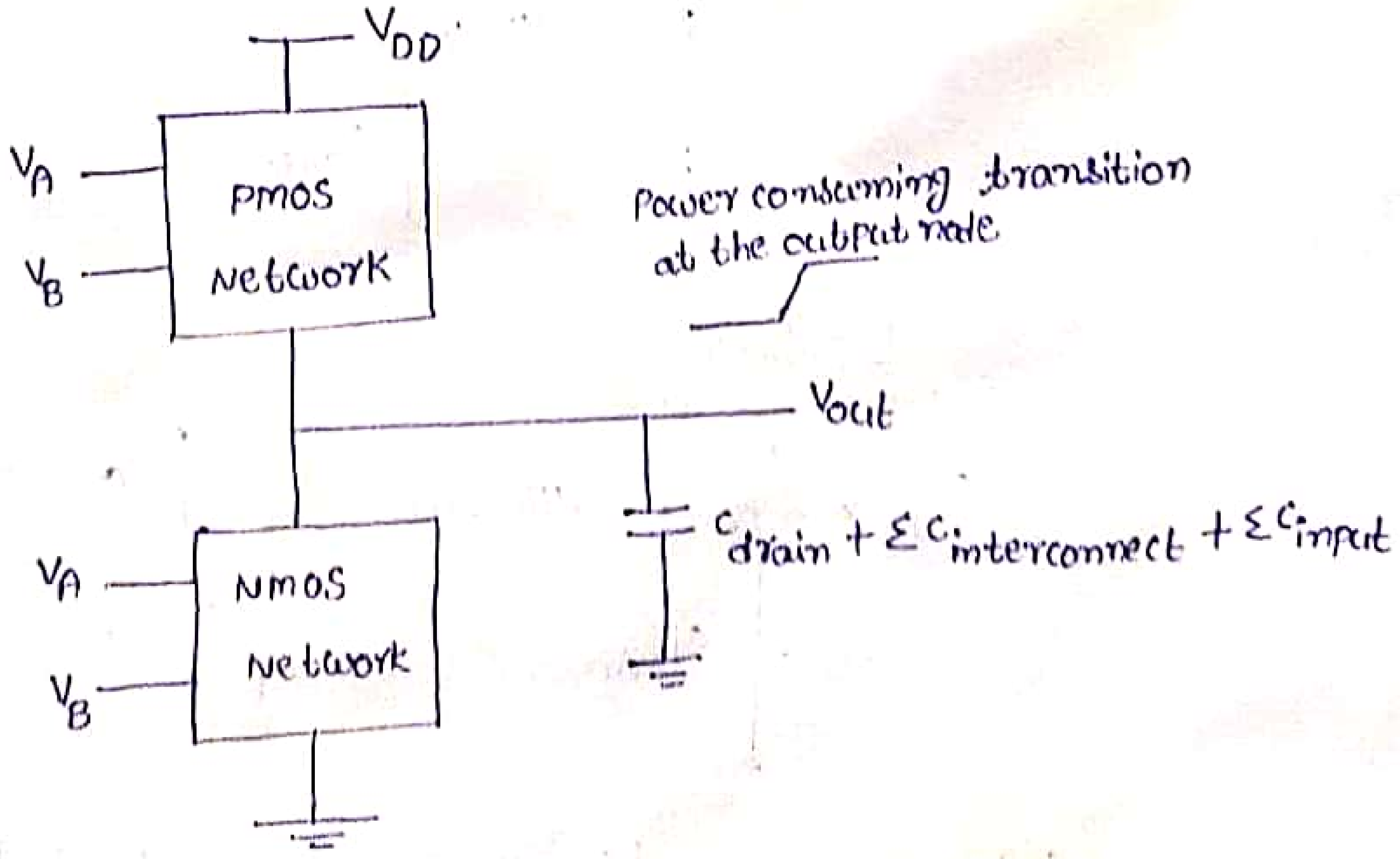

V_{DD}
V_A
V_B
PMOS Network
Power consuming transition at the output node
V_{out}
$C_{drain} + \Sigma C_{interconnect} + \Sigma C_{input}$
V_A
V_B
NMOS Network

Fig:- cmos Logic gate for switching power caluclation

The energy required to charge up the output node to V_{DD} and charge down the total output load capacitance to ground level.

The Average dynamic Power consumption in cmos Logic gates

$$P_{avg} = \frac{1}{T} C_{Load} V_{DD}^2$$

(or)

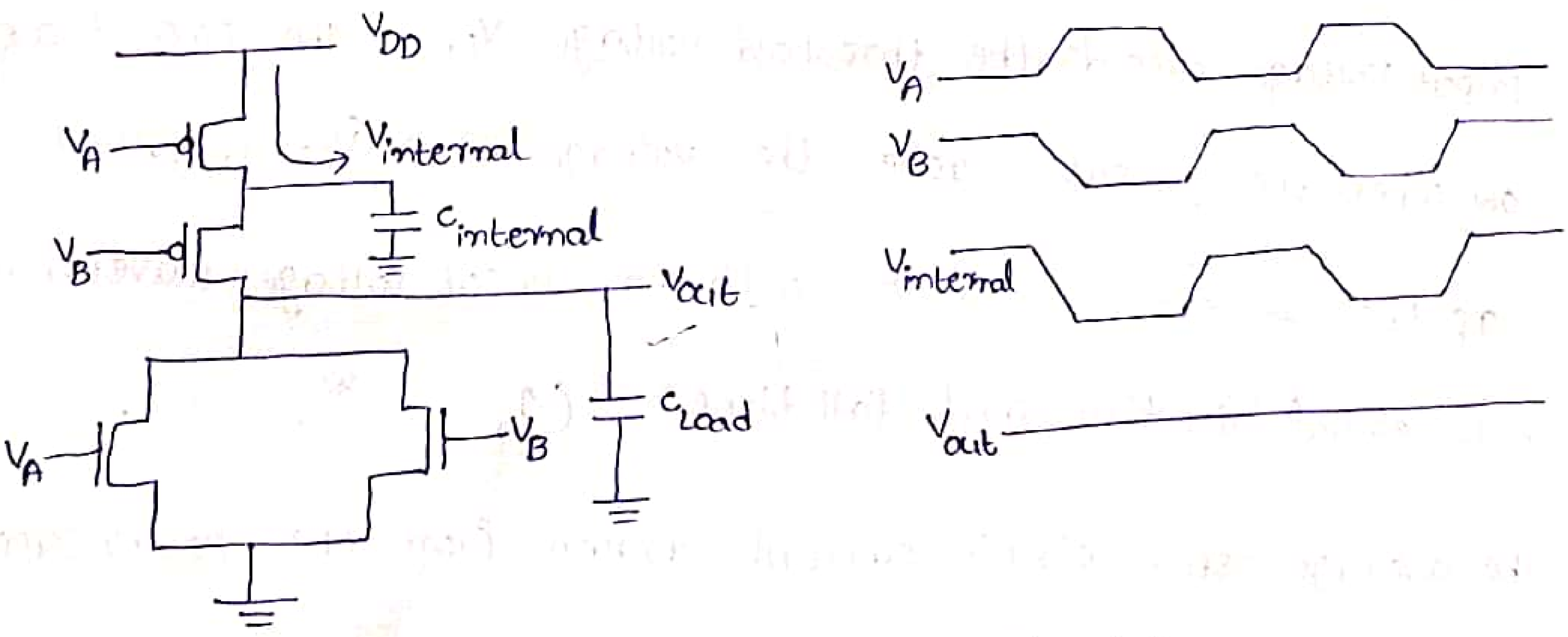
$$P_{avg} = C_{Load} V_{DD}^2 f_{CLK}$$

To Introduce α_T (node transition factor), which is the effective no. of Power consuming voltage transitions experienced per clock cycle.

Then, average switching power consumption becomes

$$P_{avg} = \alpha_T C_{Load} V_{DD}^2 f_{CLK}$$

consider 2-Input cmos NOR gate results in dynamic Power dissipation even if the output node voltage remains unchanged.



The Dynamic Power consumption can be reduced by

i) Reduction of Power Supply voltage V_{DD}

ii) Reduction of Switching Probability

iii) Reduction of Load capacitance

ii) Short circuit Power dissipation:-

The circuit current component which passes through both Nmos and Pmos devices during switching does not contribute to the charging of the capacitances in the circuit hence it is called "short circuit current component".

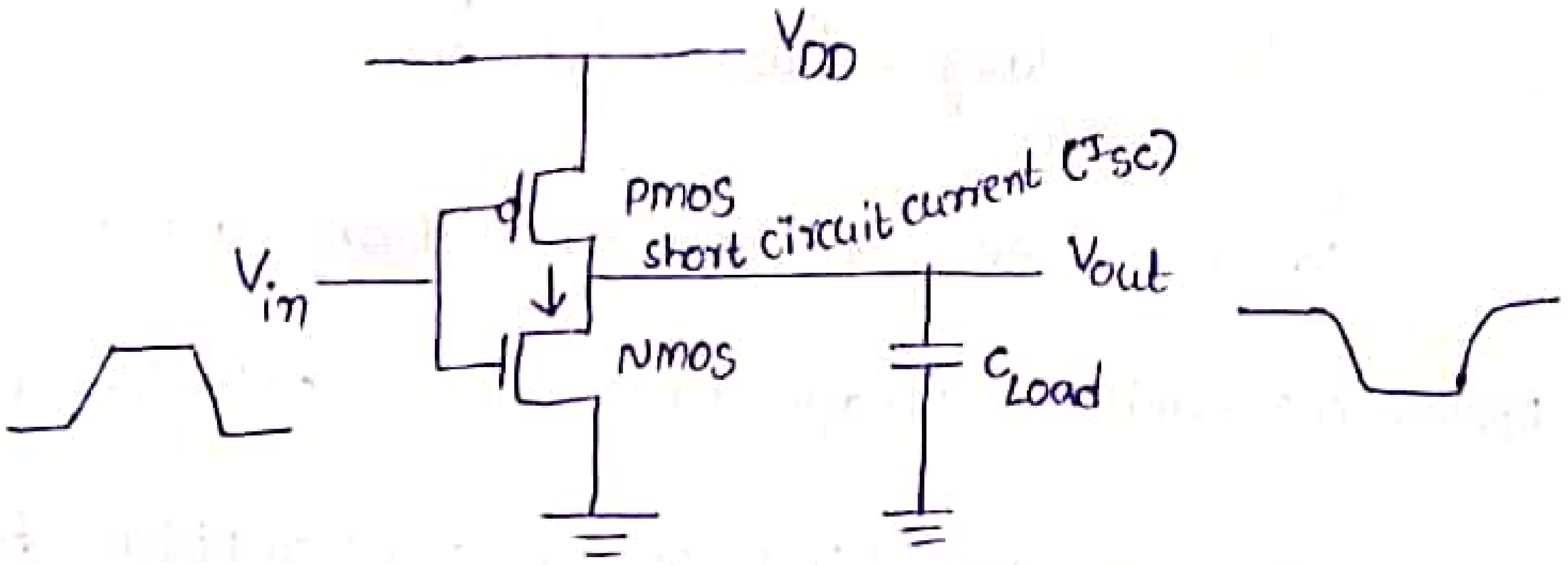


Fig:- short circuit current during switching

consider a symmetric cmos Inverter, with $K_n = K_p = K$ and $V_{Tn} = |V_{TP}| = V_T$ and with a very small capacitive Load.

The Nmos transistor in the circuit starts conducting when the Input Voltage exceeds the threshold voltage V_{Tn}. The Pmos transistor ON when the Input reaches the voltage level ($V_{DD} - V_{TP}$).

If the Inverter is driven with an Input voltage waveform with equal rise time and fall times ($\tau_{rise} = \tau_{fall} = \tau$).

The average short circuit current drawn from the power supply is

$$I_{avg}\,(\text{short-circuit}) = \frac{1}{12}\,\frac{K\tau f_{CLK}}{V_{DD}}\,(V_{DD} - 2V_T)^3$$

The short circuit power dissipation becomes

$$P_{avg}\,(\text{short-circuit}) = \frac{1}{12} K\tau f_{CLK} (V_{DD} - 2V_T)^3$$

Note that short circuit power dissipation is linearly Proportional to Input signal rise & fall times and also transconductance of the transistors.

The Input - output Voltage waveforms as shown in fig.

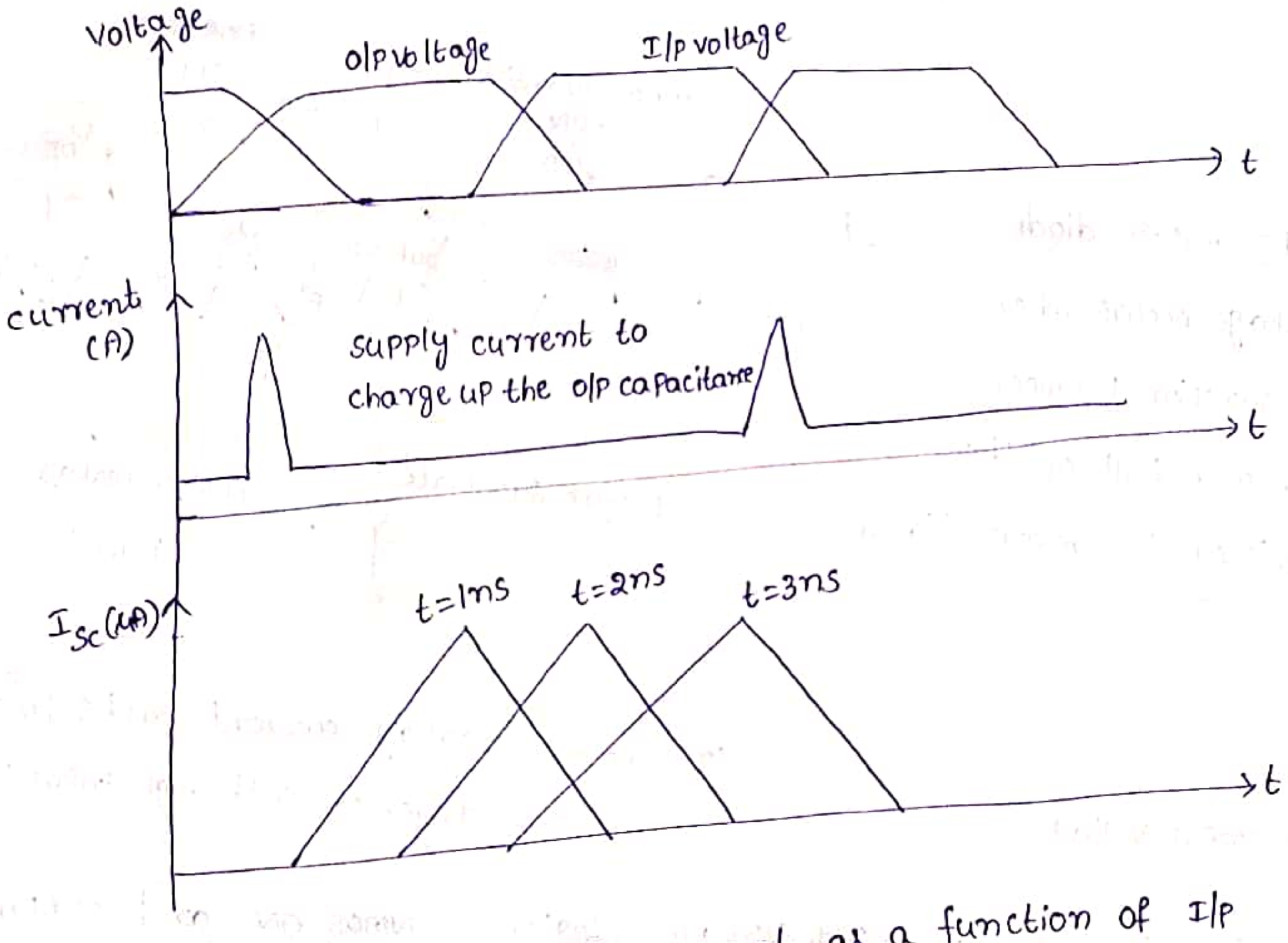


Fig:- short circuit current as a function of I/p rise time / fall time

iii) Leakage Power dissipation:-

Nmos, Pmos transistors used in cmos Logic gate have Non zero reverse Leakage and subthreshold currents.

In cmos VLSI chip containing very large no. of transistors, these currents contribute overall power dissipation.

The magnitude of leakage currents determined by

i) Reverse Leakage current

ii) Sub threshold Leakage current

i) Reverse Leakage current in a cmos Inverter :-

The reverse diode leakage occurs when p-n Junction between drain and bulk of the transistor is reverse biased.

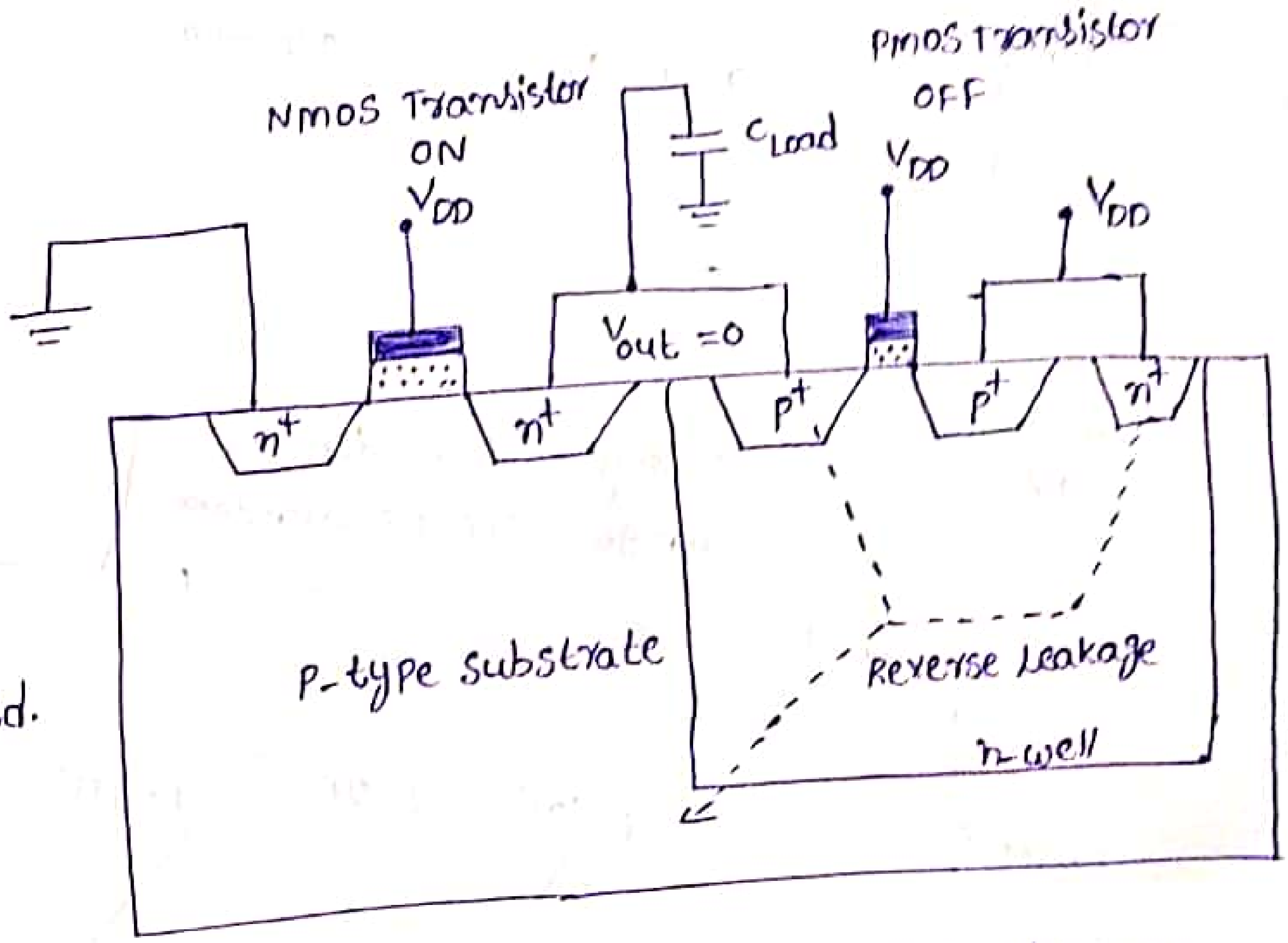


Fig :- Reverse Leakage current Paths in a cmos Inverter with high Input Voltage

High Input Voltage

By applying high input voltage (Logic1), Nmos ON and output node voltage discharged to zero. Pmos OFF, The reverse potential difference of V_{DD} between drain and N-well causing a diode leakage current through drain Junction. N-well region of pmos transistor also reverse biased with V_{DD} w.r.to p-substrate.

Low Input voltage

PMOS ON, The output node voltage charged to V_{DD}.

The reverse Potential difference between Nmos drain region and p-type substrate causes reverse leakage current which is drawn from power supply.

The reverse leakage current of p-n Junction is

$$I_{reverse} = A J_S \left(e^{\frac{q V_{bias}}{KT}} - 1 \right)$$

where,

V_{bias} → Reverse bias voltage

J_S → Reverse saturation current density

A → Area

ii) subthreshold Leakage current :-

which is due to carrier diffusion between source and drain regions of the transistor in weak Inversion.

The behaviour of mos transistor in subthreshold operating region is similar to Bipolar device.

If there is no switching activity in the circuit, subthreshold leakage current can occur.

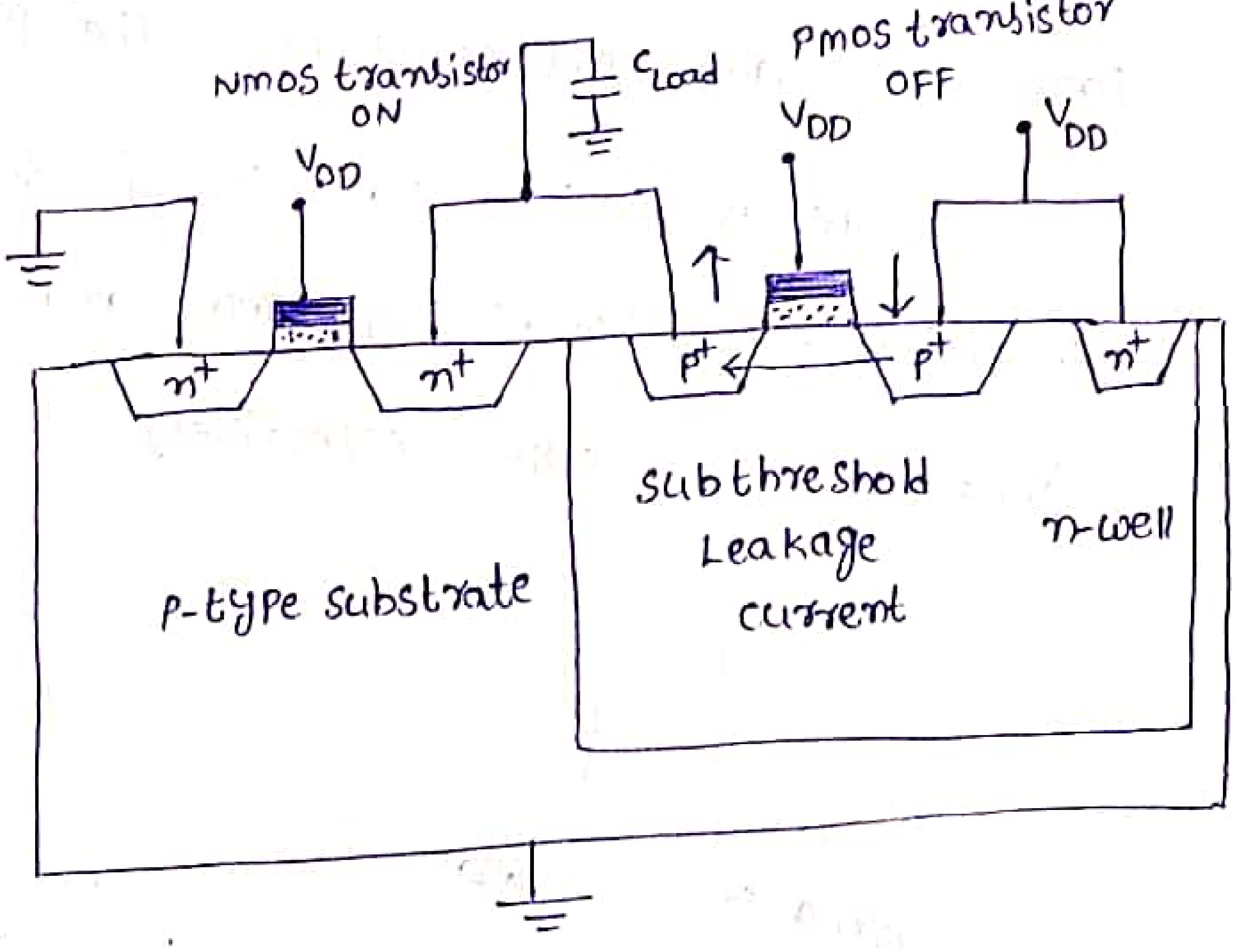


Fig:- subthreshold Leakage current Path in a cmos Inverter with high Input voltage

$$I_{D(\text{subthreshold})} = \frac{q D_n W x_c n_0}{L_B} e^{\frac{q\phi_r}{KT}} e^{q/KT\,(A V_{GS} + B V_{DS})}$$

The total power dissipation in cmos digital circuits is

$$P_{total} = \alpha_T C_{load} V_{DD}^2 f_{CLK} + V_{DD}(I_{short\text{-}circuit} + I_{Leakage} + I_{static})$$

Low Power design through voltage scaling :-

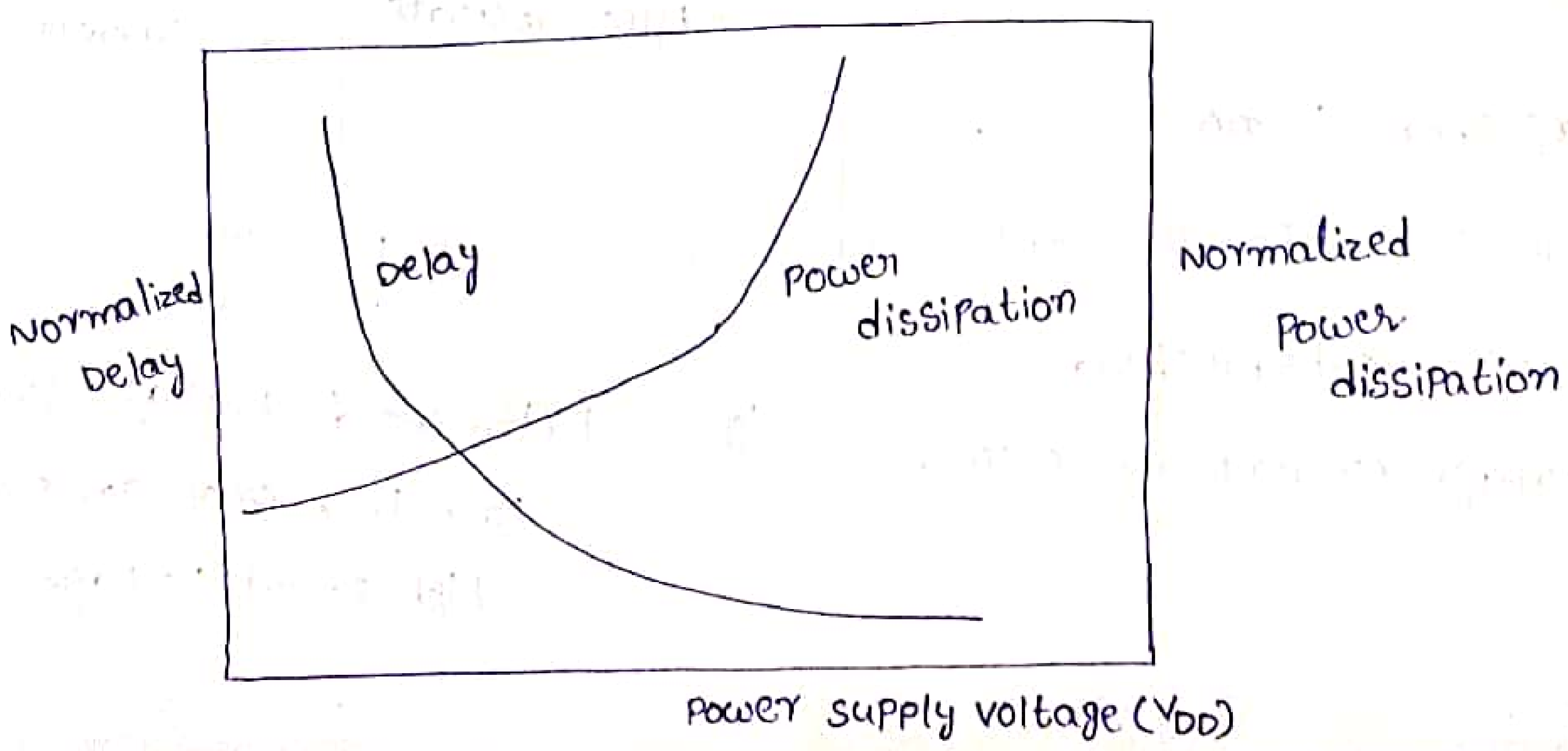
The average Power $P_{avg} = C_{load} V_{DD}^2 f_{CLK}$

The Average switching Power dissipation is Proportional to the square of the Power supply voltage, hence reduction of V_{DD} will reduce power dissipation.

If the power supply voltage is scaled down while all Parameters are kept constant, the Propagation delay time Increase.

The normalized variation of delay as a function of V_{DD}.

where threshold voltages of Nmos and pmos transistor are $V_{Tn} = 0.8V$, $V_{TP} = -0.8V$ respectively.



For Example, reducing the threshold voltage from 0.8V to 0.2V can Improve the delay at $V_{DD}=2V$ by a factor 2.

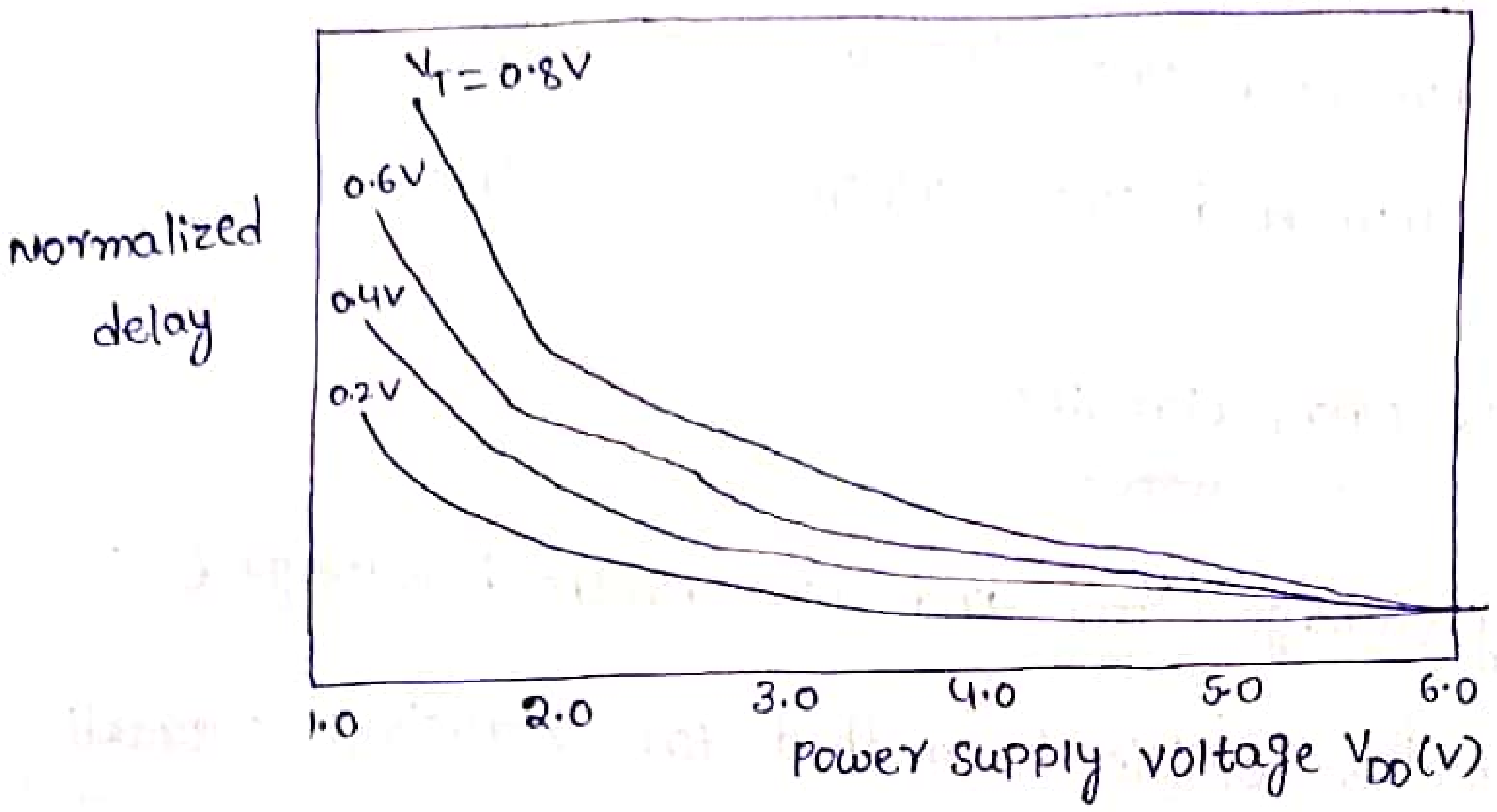


Fig:- Variation of Normalized Propagation delay of CMOS Inverter, as a function of V_{DD} and Threshold voltage (V_T)

The Influence of threshold voltage reduction upon propagation delay is especially pronounced at low power supply voltages for $V_{DD}<2V$. It should be noted, however, that using low V_T transistors raises significant concerns about noise margins and subthreshold conduction. Smaller threshold voltages lead to smaller noise margins for the CMOS Logic gates.

The Threshold Voltages smaller than 0.2V, Leakage Power dissipation due to subthreshold conduction may become a very significant component of the overall power consumption.

Two circuit design techniques used to overcome the leakage and high power dissipation associated with Low - V_T circuits The techniques are

i) Variable - Threshold cmos (VT cmos) circuits

ii) multiple - Threshold cmos (MTcmos) circuits

i) Variable - Threshold cmos circuits

Low supply voltage (V_{DD}) and Low Threshold voltage (V_T) in cmos logic circuits is an efficient method for reducing overall power dissipation & maintaining high speed performance.

In cmos Logic circuits, the substrate terminals of all nmos transistors are connected to ground while substrate termina of all pmos transistors are connected to V_{DD}.

In VTcmos circuit technique, transistors are designed with low threshold voltage, and substrate bias of Nmos, pmos transistors are generated by a variable ~~threshold~~ substrate bias control circuit as shown in fig.

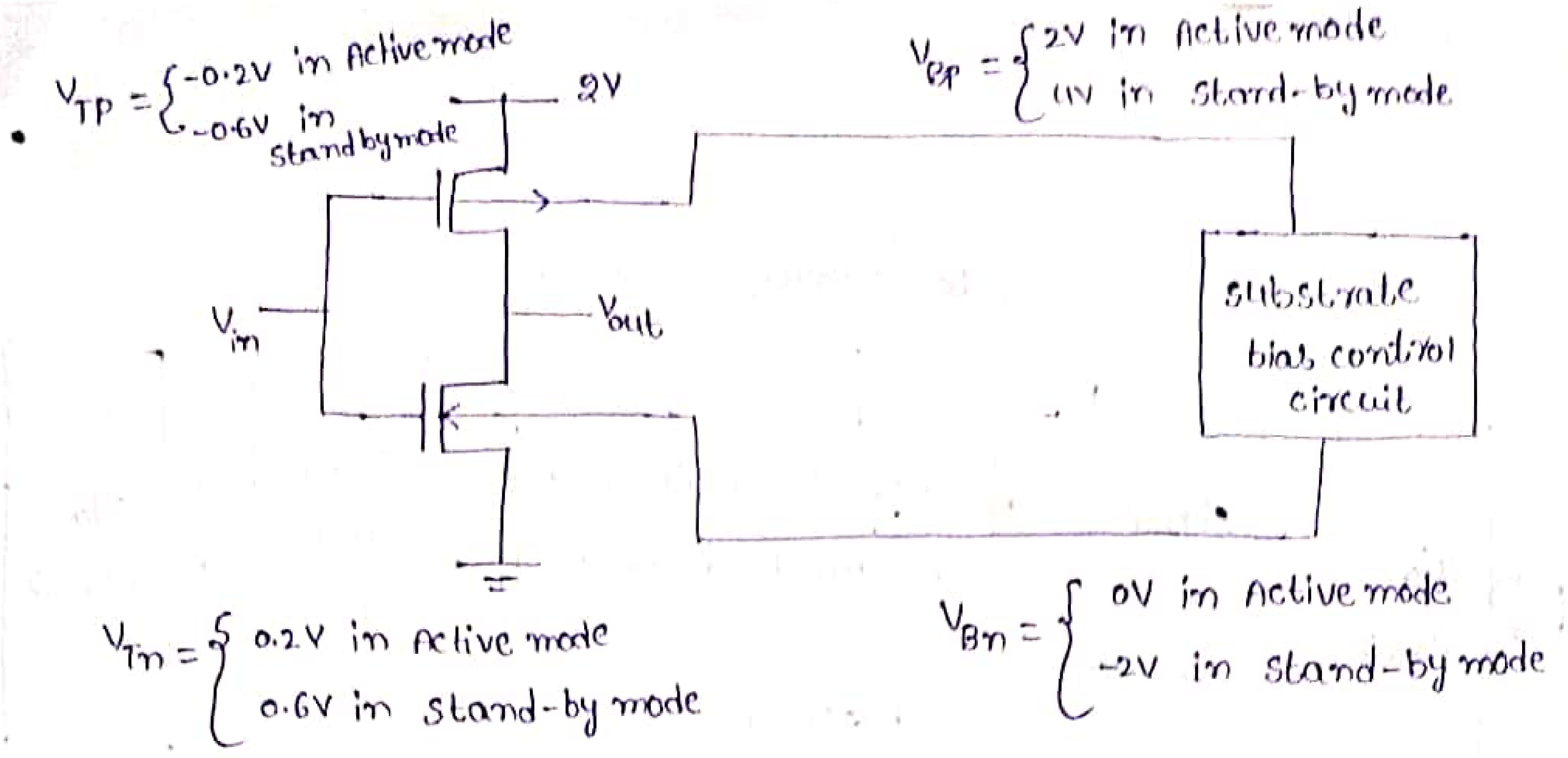

$V_{TP} = \begin{cases} -0.2V \text{ in Active mode} \\ -0.6V \text{ in Stand by mode} \end{cases}$
2V
$V_{Bp} = \begin{cases} 2V \text{ in Active mode} \\ 4V \text{ in Stand-by mode} \end{cases}$
V_{in}
V_{out}
substrate bias control circuit
$V_{Tn} = \begin{cases} 0.2V \text{ in Active mode} \\ 0.6V \text{ in stand-by mode} \end{cases}$
$V_{Bn} = \begin{cases} 0V \text{ in Active mode} \\ -2V \text{ in stand-by mode} \end{cases}$

Fig :- Variable Threshold (VTcmos) cmos Inverter circuit

In Active mode, substrate bias voltage for nmos transistor $V_{Bn} = 0$ and P-mos transistor $V_{BP} = V_{DD}$. Thus Inverter transistors do not experience any body effect (back gate-bias effect).

In stand-by mode, substrate bias control circuit generates a lower substrate bias voltage for nmos transistor and higher substrate bias voltage for pmos transistor. The magnitudes of the threshold voltages of V_{Tn} and V_{TP} both Increase in stand-by mode. due to back-gate bias effect.

The VTcmos circuit can also used to control the threshold voltages of transistors in order to reduce leakage currents.

The block diagram of Low-Power chip with Low Internal supply voltage V_{DDL} and threshold voltage control as shown in fig. below

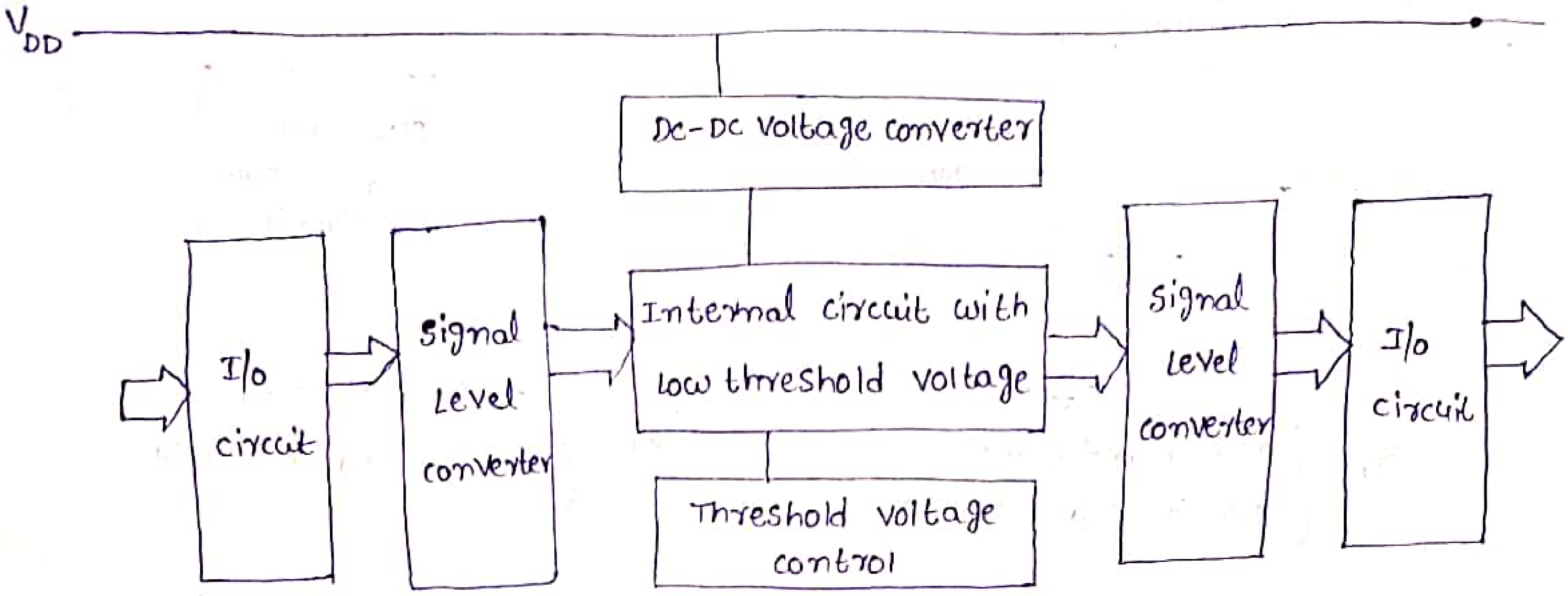


Fig:- Block diagram of Low-Power chip

- I/o circuit of chip usually operate with a higher external supply voltage, in order to Increase noise margins to enable communication with peripheral devices.

- An on chip DC-DC voltage converter generates Low Internal supply voltage V_{DDL}.

- Two signal level converters are used to reduce voltage swing of the the Incoming Input signals and Increase voltage swing of the outgoing output signals.

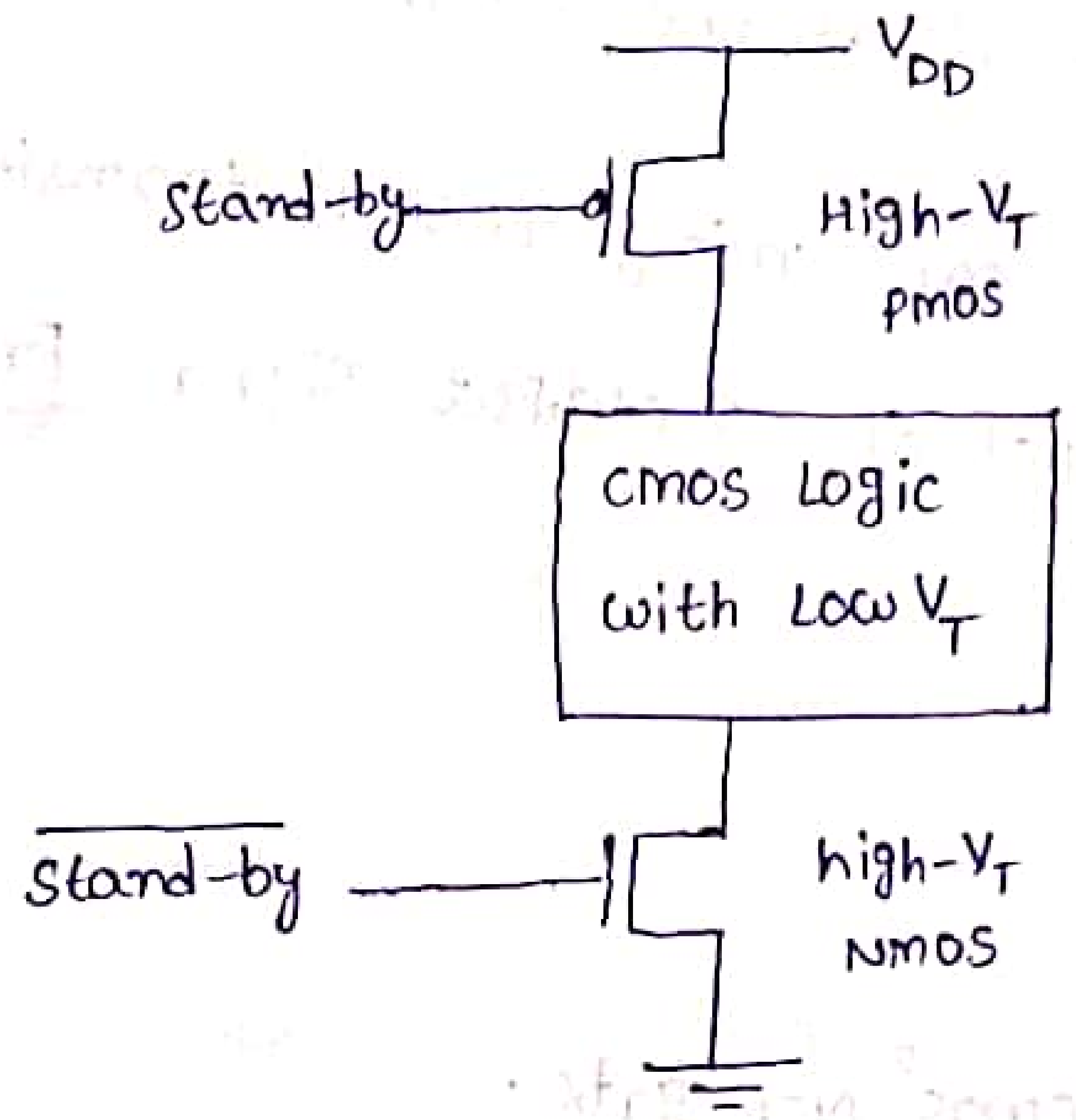
- The Internal Low-voltage circuitry can be designed using VTCmos techniques, where the threshold voltage control unit adjust the substrate bias in order to suppress leakage currents.

ii) Multiple-Threshold CMOS (MT CMOS) circuits :-

In this technique, stand-by mode is based on 2 different types of transistors (NMOS, PMOS) with 2 different threshold voltages:

i) Low-V_T transistors used to design Logic gates where switching speed is essential.

ii) High-V_T transistors used to prevent Leakage dissipation.



Prevents subthreshold Leakage in stand-by mode

High-speed operation with Low Per Power consumption

Prevents subthreshold Leakage in stand-by mode

Fig:- Structure of MTCMOS Logic gate

→ In Active mode, high-V_T transistors are turned on and the Logic gates consists of Low-V_T transistors operate with Low switching power dissipation and small propagation delay.

→ In stand-by mode, high-V_T transistors are turned off and conduction paths for any subthreshold Leakage current that may originate from Internal Low-V_T circuitry are effectively cutoff.

Estimation and optimization of switching activity :-

The node transition factor α_T, which is the effective no. of Power consuming voltage transitions experienced by output capacitance per clock cycle. This parameter also called "switching Activity factor".

consider 2 signal Probabilities P_0, P_1.

P_0 → Probability of having Logic '0' at the output

P_1 → Probability of having Logic '1' at the output.

The Probability that power-consuming (0 to 1) transition occurs at the output node is product of 2 output signal Probabilities

$$P_{(0\to1)} = P_0 \cdot P_1$$

For Example,

consider a static cmos 2-I/P NOR gate.

A	B	$Z(\overline{A+B})$
0	0	1
0	1	0
1	0	0
1	1	0

The Probability that output getting '1' → $\frac{1}{4}$.

Probability that output getting '0' → $\frac{3}{4}$

The probability that Power-consuming transition at output node is

$$P_{0\to1} = P_0 \cdot P_1$$
$$= \frac{3}{4} \times \frac{1}{4} = \frac{3}{16}$$

The state transition diagram as

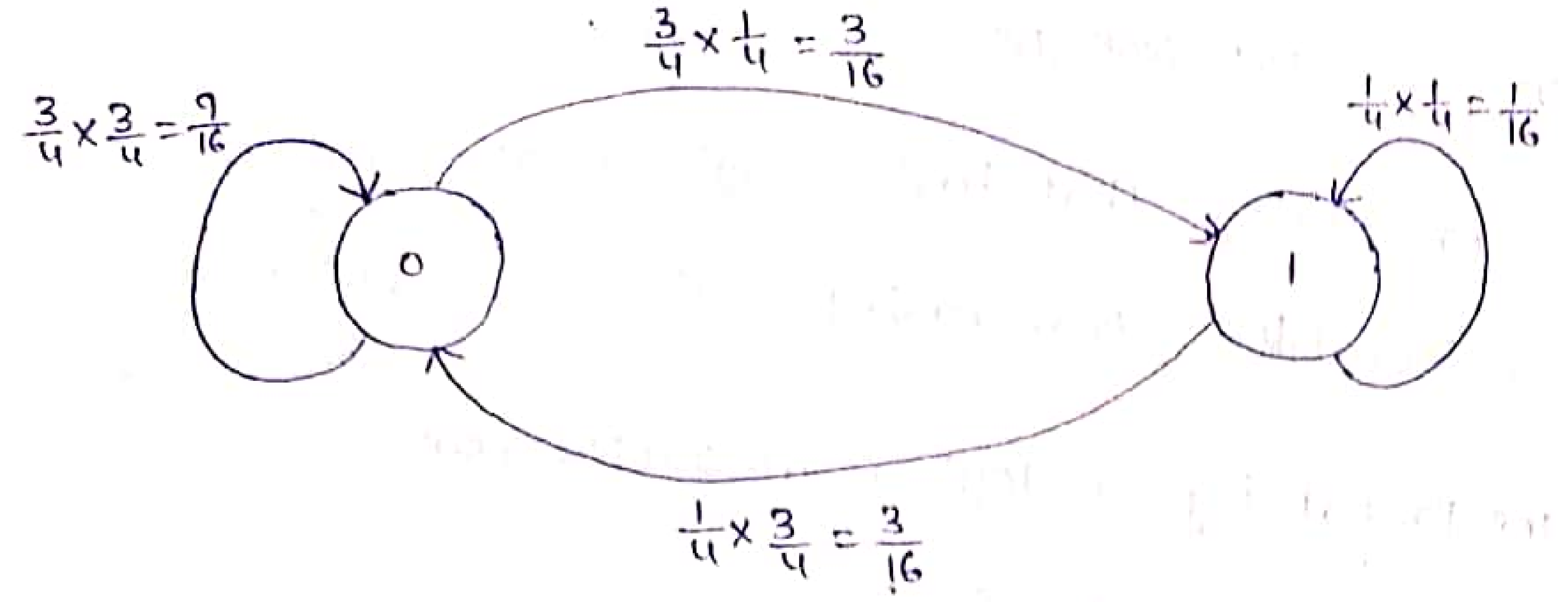

$\frac{3}{4}\times\frac{1}{4}=\frac{3}{16}$
$\frac{3}{4}\times\frac{3}{4}=\frac{9}{16}$
$\frac{1}{4}\times\frac{1}{4}=\frac{1}{16}$
0
1
$\frac{1}{4}\times\frac{3}{4}=\frac{3}{16}$

Fig:- state transition diagram for 2-I/P NOR gate

Generally, cmos Logic gate with n- I/P variables, contains 2^n output combinations.

$n_0 \rightarrow$ Probability of output will be '0'

$n_1 \rightarrow$ Probability of output will be '1'

Then $P_0 = \frac{n_0}{2^n}$, $P_1 = \frac{n_1}{2^n}$

The output node voltage from '0' to '1' is

$$P_{0\to1} = P_0 . P_1 = \frac{n_0}{2^n} \cdot \frac{n_1}{2^n}$$

$$= \left(\frac{n_0}{2^n}\right)\cdot\left(\frac{2^n - n_0}{2^n}\right)$$

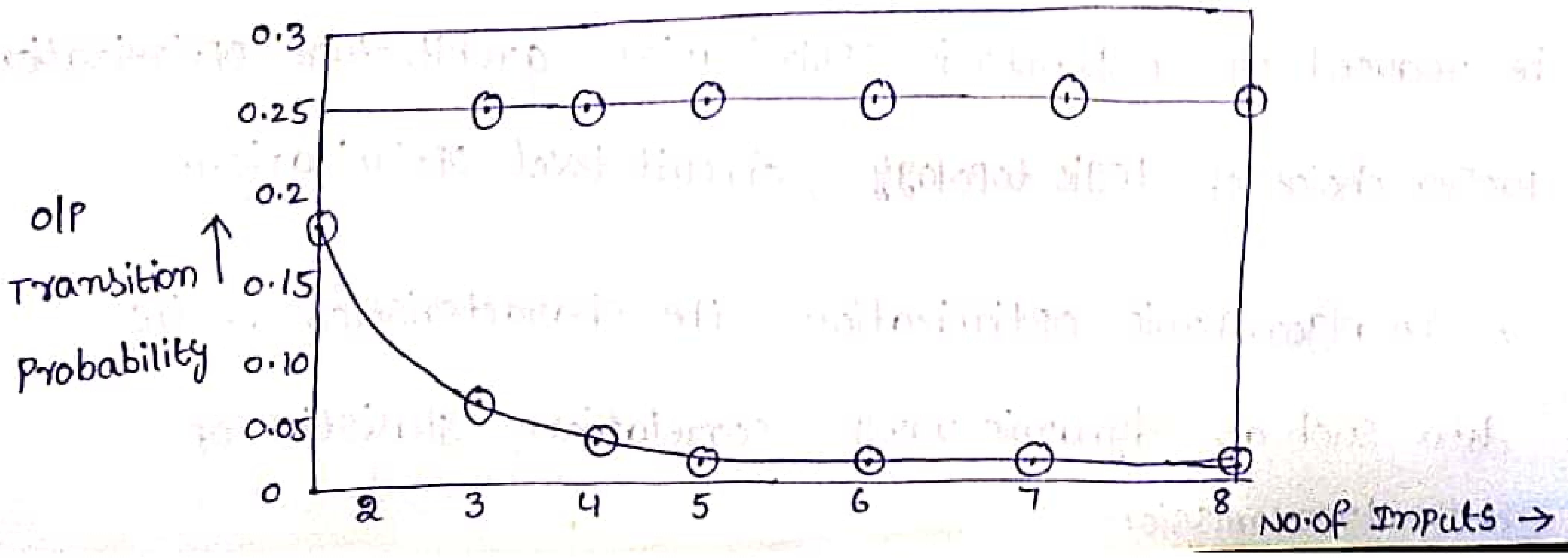
$\left[\because P_0 + P_1 = 1\right.$

$\frac{n_0}{2^n} + \frac{n_1}{2^n} = 1$

$n_0 + n_1 = 2^n$

$\left. n_1 = 2^n - n_0\right]$

The output transition probabilities of different logic gates as a function of No. of Inputs as shown in fig.



consider, 2-Input NOR gate,

P_{1A} → Probability that having logic '1' at Input 'A'

P_{1B} → Probability that having Logic '1' at Input 'B'

The probability of logic '1' at output node is

$$P_1 = (1-P_{1A})\cdot(1-P_{1B})$$

Power consuming output transition is

$$P_{0\to1} = P_0 \cdot P_1 = (1-P_1)P_1$$

$$= (1-P_1)P_1$$

$$= (1-(1-P_{1A})(1-P_{1B}))((1-P_{1A})(1-P_{1B}))$$

similarly, 2-Input NAND gate,

Probability of Logic '0' at output is $P_0 = P_A P_B$

Probability of Logic '1' at output is $P_1 = 1-P_0$

$$= 1-(P_A P_B)$$

The power consuming output transition is

$$P_{0\to1} = P_0 \cdot P_1$$

$$= P_A P_B (1-P_A P_B)$$

Reduction of switching activity :-

Switching Activity in cmos digital Integrated circuits can be reduced by Algorithmic optimization, architecture optimization, by proper choice of Logic topology, circuit level optimization.

→ In Algorithmic optimization, the characteristics of the data such as dynamic range, correlation, statistics of data transmission.

i) To use Gray codes Instead of binary codes because to reduce no. of transitions.

ii) To use Sign-magnitude representation Instead of 2's complement representation.

→ Architecture-level measure to reduce switching activity is based on delay balancing and reduction of glitches.

Glitch reduction:-

In multilevel Logic circuits, the Propagation delay from one Logic block to next can cause signal transitions (or) Glitches as a result of critical races (or) dynamic hazards.

Generally, all Input signals of a gate change Simultaneously no glitching occurs. Glitch (or) dynamic hazard occur if Input signals change at different times.

In some cases, the signal glitches are only Partial i.e node Voltage does not make full transition between ground and V_{DD} levels.

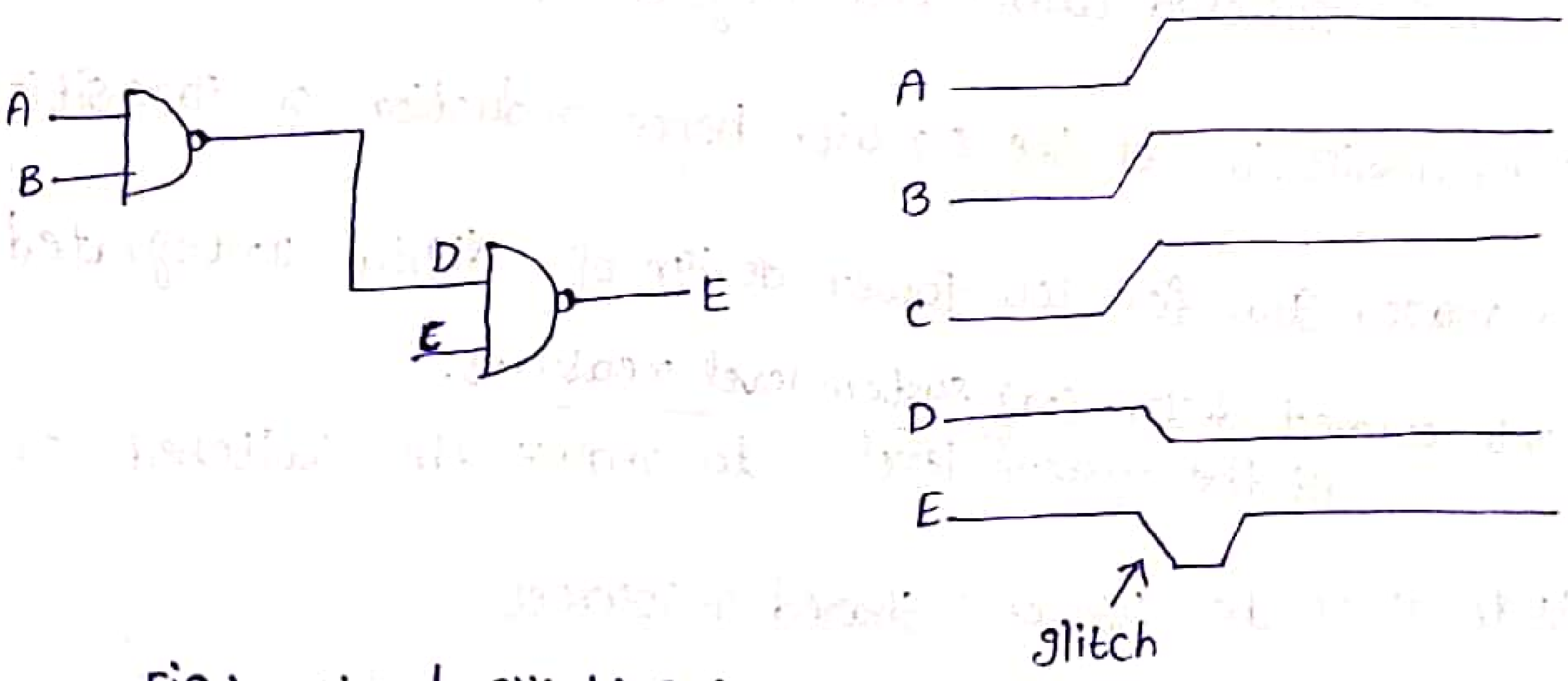
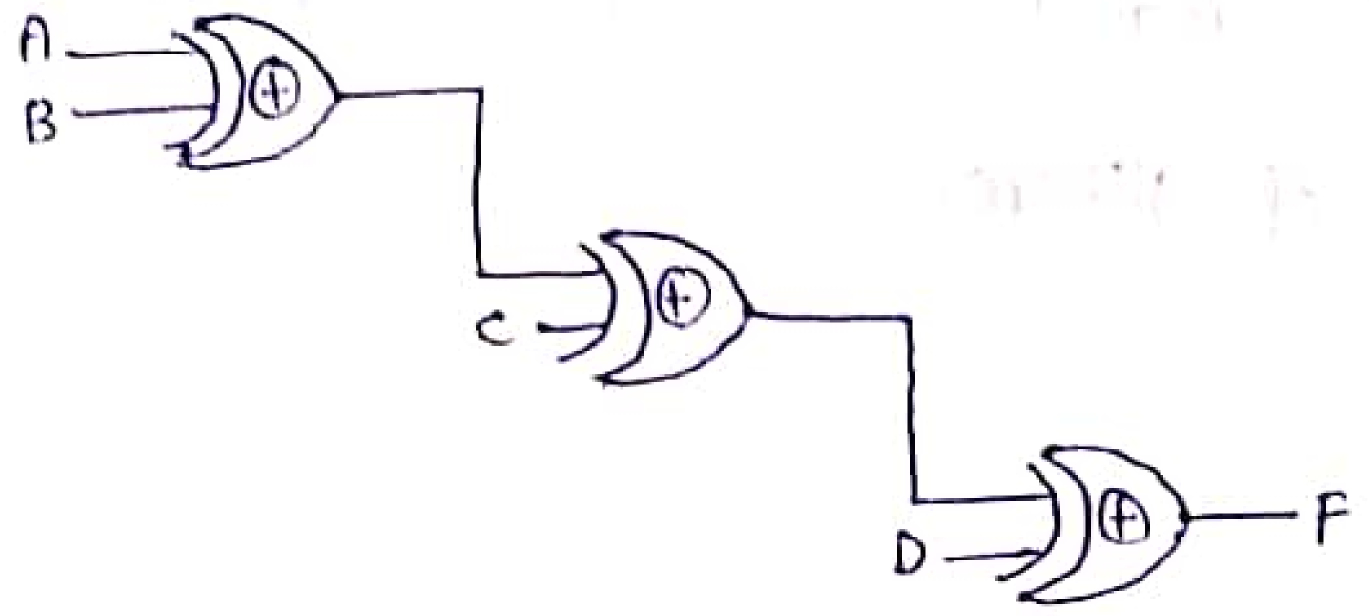


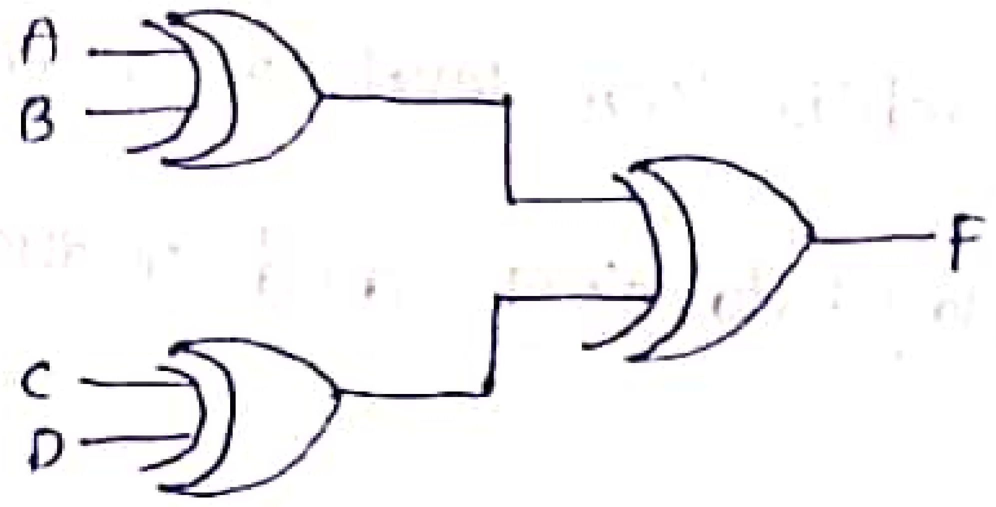
Fig:- Signal glitching in multi-level static cmos circuits

Glitches occur Primarily due to mismatch (or) Imbalance in the Path lengths in Logic Networks.

consider 4-I/p XOR gate structure, All XOR gates have same delay and 4-I/p signals arrive at same time.



a)



b)

From fig (a), will suffer from glitching due to Input arrival times are different.

From fig (b), All Inputs arrive at same time so there is no glitching occurs.

from fig (b), results in smaller Propagation delay.

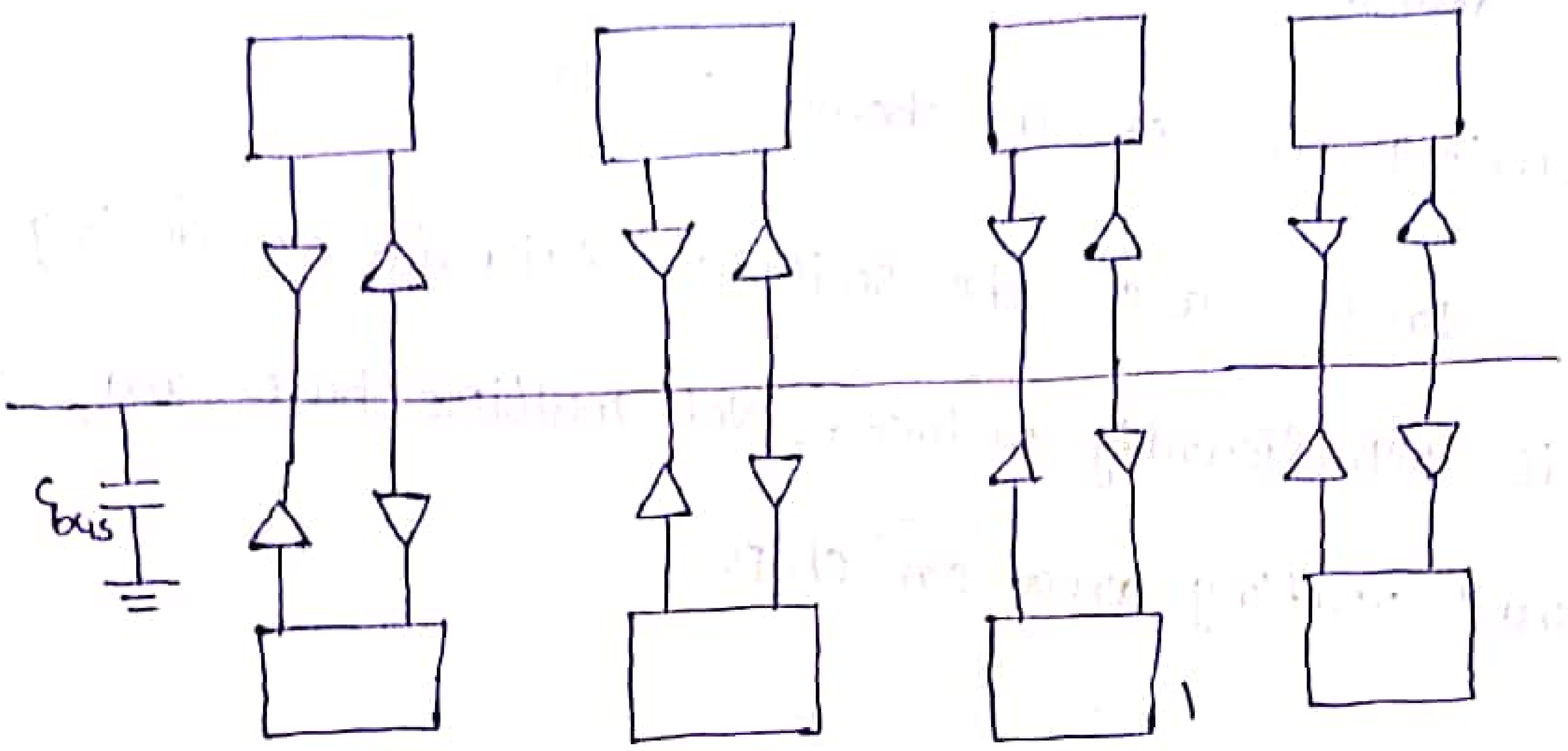
Reduction of Switched capacitance :-

Switched capacitance plays a significant role in dynamic Power dissipation of the circuit. hence reduction of Parasitic capacitance is a major goal for Low-Power design of digital Integrated circuits.

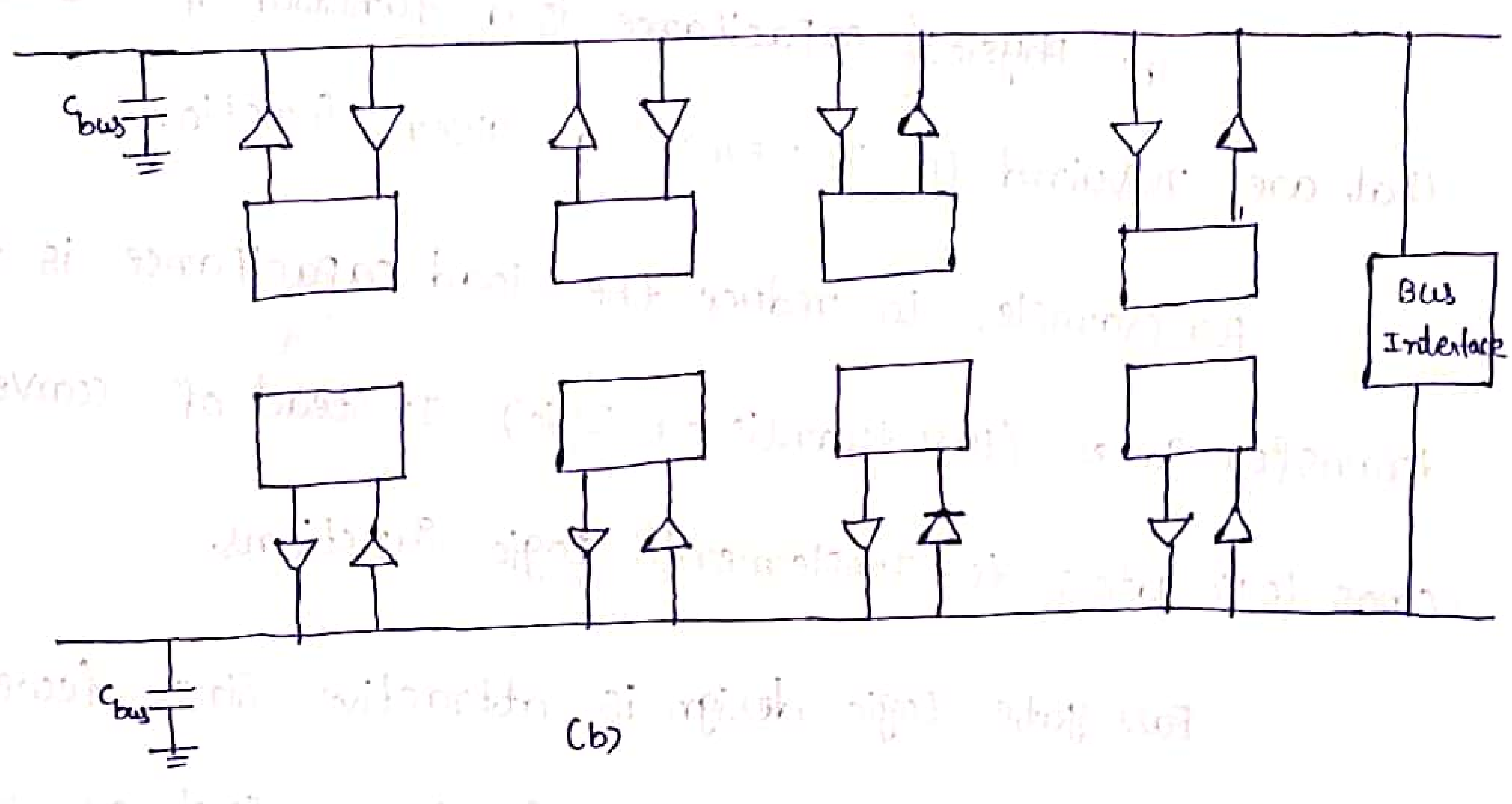
Interconnect design (or) system level measures:-

At the system level, to reduce the switched capacitance is to limit the use of shared resources.

A simple example is the use of global bus structure for data transmission between large no. of operational modules.



(a)



(b)

From fig (a), this structure results in a large bus capacitance due to. i) The large no. of drivers and receivers sharing the same transmission medium.

ii) The parasitic capacitance of the long bus line.

Obviously, driving the large bus capacitance will require a significant amount of Power consumption during each bus access. Alternatively, the global bus structure can be partioned into no. of smaller dedicated buses to handle data transmission between neighbouring modules as shown in fig (b).

In this case, the switched capacitance during each bus access is significantly reduced, yet multiple buses may Increase the overall routing area on chip.

circuit-level measures :-

The Physical capacitance is a function of no. of transistors that are required to Implement a given function.

For example, to reduce the load capacitance is to use transfer gates (Pass transistor logic) Instead of conventional cmos Logic gates to Implement logic functions.

Pass gate Logic design is attractive Since fewer transistors are required certain functions such as XOR, XNOR. In arithmetic operations where binary adders and multipliers are used, Pass transistor logic offers significant advantages.

clock design:-

To reduce the switching activity in cmos Logic circuits is the use of gated clock signals.

The block diagram of an N-bit comparator circuit which is designed using the gated clock technique.

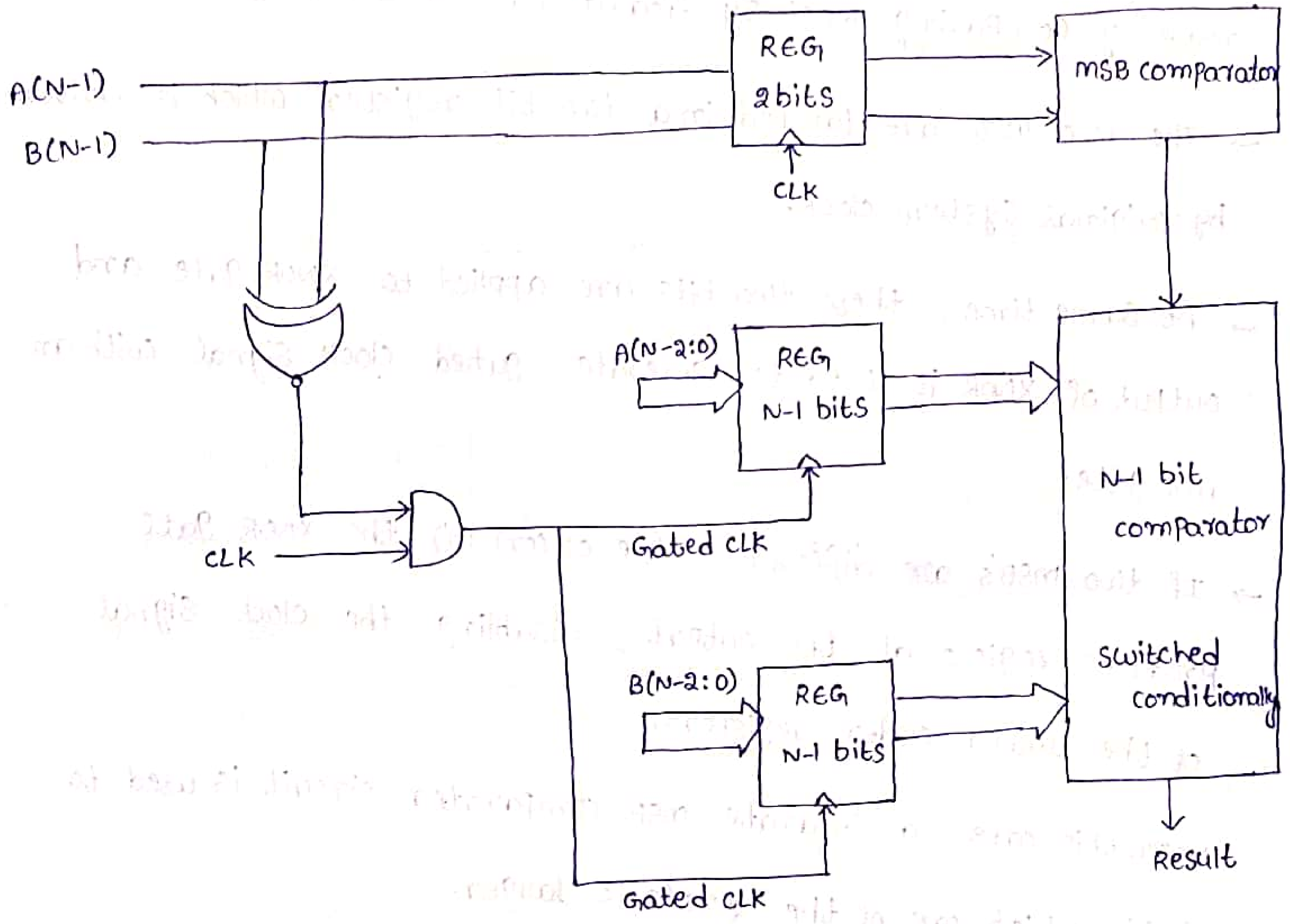


Fig:- Block diagram of an N-bit no. comparator with gated clock

→ The circuit compares the magnitudes of the two unsigned N-bit binary numbers (A and B) and produces an output to Indicate which one is larger.

→ All Input bits are first Latched into 2 ~~bi~~ N-bit registers, and applied to comparator circuit.

→ In this case, two N-bit registers arrays dissipate Power in eve clock cycle.

→ Yes, if the most significant bits, A(N-1) and B(N-1) of two binary numbers are different from each other, then decision can be made by comparing most significant bits (MSBs) only.

→ The two MSBs are latched in a two-bit register which is driven by original system clock.

→ At same time, these two bits are applied to XNOR gate and output of XNOR is used to generate gated clock signal with an AND gate.

→ If two MSB's are different (i.e 01 (or) 10) the XNOR gate produces Logic 0 at the output, disabling the clock signal of the lower-order registers.

→ In this case, a separate MSB comparator circuit is used to decide which one of the 2 no's is larger.

→ If 2 MSBs are Identical (i.e 00 (or) 11), the gated clock signal is applied to lower-order registers and decision is made by (N-1) bit comparator circuit.

→ The amount of power dissipated in lower-order registers and (N-1) bit comparator circuit can be quite significant, if the bit-length (N) is large.

→ Assume that, the Incoming binary no's are randomly distributed, we can see that the gated clock strategy effectively reduces over all switching Power dissipation of S/m approximately 50% since a large portion of the s/m is disabled for half of all Input combinations.

Power Grid :-

The Power distribution system design is an area of Increasing semiconductor Industry. According to data in more than 50% of tape outs using 0.13 micron technology would fail, if the Power distributed system were not validated.

Lower operating voltages, Increased device Integration density and leakage currents, higher operating frequencies and use of low power design techniques; they all tend to stress the Power grid as technology evolves.

mainly there are 4 major Problems that affect Power-distribution system are

1. Voltage drop
2. Ground bounce
3. L di/dt Noise
4. Electro migration

Voltage drop also called IR drop, is the voltage reduction that occurs on power supply networks.

The IR drop can be static (or) dynamic and results from the existence of non- ideal elements : The resistance with in the power and ground supply wiring and capacitance between them.

static voltage drop considers only Average currents, dynamic voltage drop considers current waveforms with in clock cycles.

Similar effects may be found in ground wiring, usually referred as ground bounce.

Both effects contribute lower operating voltages, which Increase overall time response of a device.

The $L\frac{di}{dt}$ Noise is caused by current spikes on wires that will Induce voltage changes on these wires and their neighboring wires, due to Inductance coupling.

The power grid verification is usually accomplished by simulation. The disadvantage of simulation is that stimuli must be generated very carefully such that the relevant scenarios are accounted.

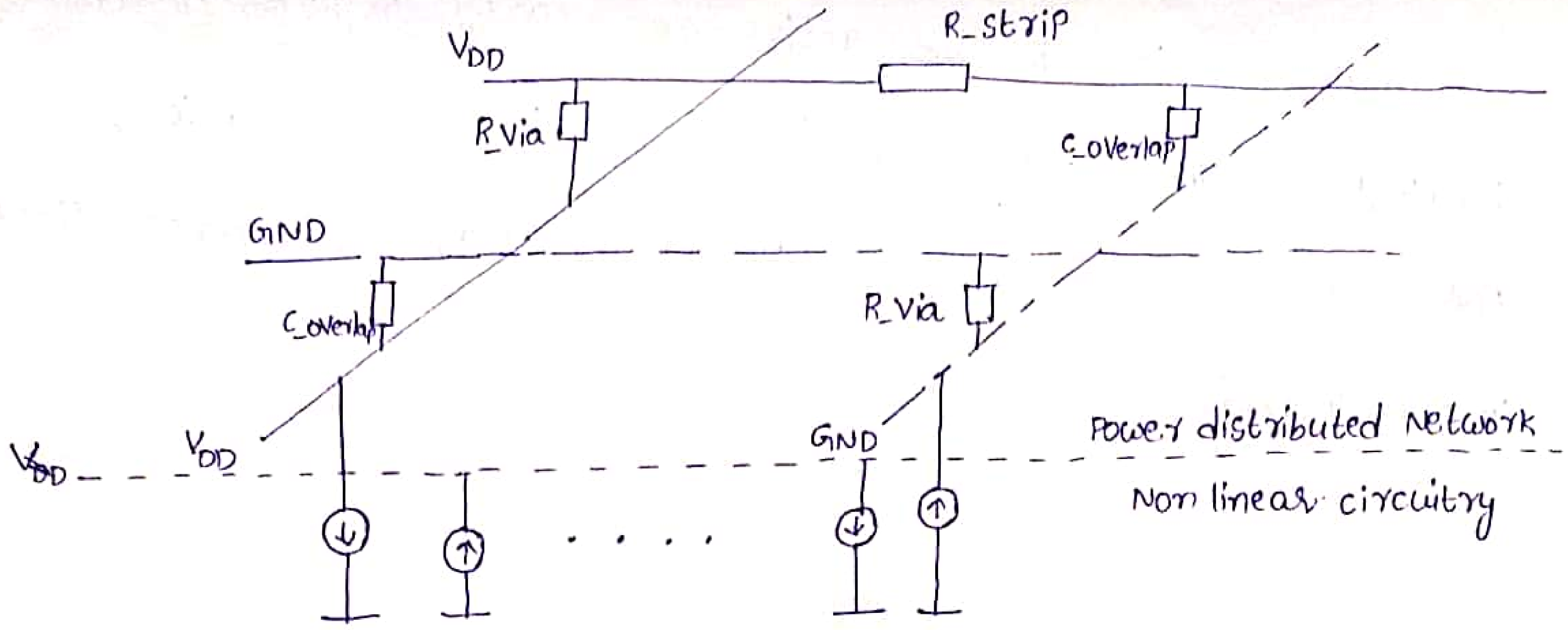


Fig :- Simplified Power grid model

simulating the power grid with all the devices might be Impossible for VLSI circuits, as it consume too many resources. Furthermore, simulating for all possible device settings is also Impossible, as it would take too long.

The Size of current designs, it is also Impossible to assume that designer Intervention will be Sufficient to generate appropriate sets of Stimuli for grid verification.

Power grid Simulation that considers all the circuit devices as simultaneously active is clearly unnecessary. Voltage drop and ground bounce may occur if there is a significant no. of devices becoming active in a short period of time and drawing current from close regions of the power grid.

we propose a technique to determine, with in a time frame, how many devices become active on nearby regions of the power grid. The higher no. of devices be in this situation, the greater the possibility of voltage drop/ground bounce effects in power grid.