DEPARTMENT OFELECTRONICSAND COMMUNICATION ENGINEERING

(ACADEMIC YEAR: 2019-2020)

EC8095-VLSI DESIGN

(Regulation 2017)

Semester-VI

NAME-

REG NO-

MOHAMED SATHAK
A J COLLEGE OF ENGINEERING
SIRUSERI IT PARK, OMR, CHENNAI - 603103

UNIT I
INTRODUCTION TO MOS TRANSISTOR

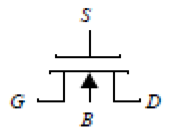
<u>MOS Transistor</u>

The metal-oxide-semiconductor field-effect transistor (MOSFET or MOS, for short) is the basic building block of contemporary digital design. Its major asset from a digital perspective is that the device performs very well as a switch, and introduces little parasitic effects. Other important advantages are its integration density combined with a relatively "simple" manufacturing process, which make it possible to produce large and complex circuits in an economical way.

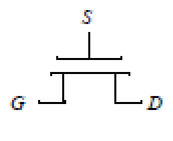
The MOSFET is a four terminal device. The voltage applied to the *gate* terminal determines if and how much current flows between the *source* and the *drain* ports. The *body* represents the fourth terminal of the transistor. Its function is secondary as it only serves to modulate the device characteristics and parameters.

At the most superficial level, the transistor can be considered to be a switch. When a voltage is applied to the gate that is larger than a given value called the *threshold voltage VT*, a conducting channel is formed between drain and source. In the presence of a voltage difference between the latter two, current flows between them. The conductivity of the channel is modulated by the gate voltage— the larger the voltage difference between gate and source, the smaller the resistance of the conducting channel and the larger the current.

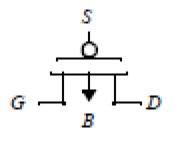
When the gate voltage is lower than the threshold, no such channel exists, and the switch is considered open. Two types of MOSFET devices can be identified. The NMOS transistor consists of *n*+ drain and source regions, embedded in a*p*-type substrate. The current is carried by electrons moving through an *n*-type channel between source and drain. This is in contrast with the *pn*-junction diode, where current is carried by both holes and electrons. MOS devices can also be made by using an *n*-type substrate and *p*+ drain and source regions. In such a transistor, current is carried by holes moving through a *p*-type channel. The device is called a *p*-channel MOS, or PMOS transistor. In a complementary MOS technology (CMOS), both devices are present. The Circuit symbols for the various MOS transistors are shown in Figure. As mentioned earlier, the transistor is a four-port device with gate, source, drain, and body terminals (Figures a and c). Since the body is generally connected to a dc supply that is identical for all devices of the same type (GND for NMOS, *Vdd* for PMOS), it is most often not shown on the schematics (Figures b and d).**If the fourth terminal is not shown, it is assumed that the body is connected to the appropriate supply.**



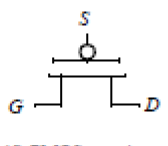
(a) NMOS transistor as 4-terminal device



(b) NMOS transistor as 3-terminal device



(a) PMOS transistor as 4-terminal device



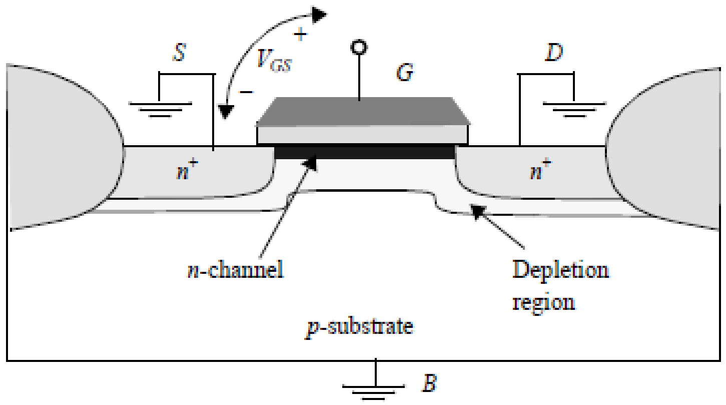
(d) PMOS transistor as 3-terminal device

i. **<u>The MOS Transistor under Static Conditions</u>**

The Threshold Voltage

Consider first the case where *VGS* = 0 and drain, source, and bulk are connected to ground. The drain and source are connected by back-to-back *pn*-junctions (substrate-source and substrate-drain). Under the mentioned conditions, both junctions have a 0 V bias and can be considered off, which results in an extremely high resistance between drain and source.

Assume now that a positive voltage is applied to the gate (with respect to the source), as shown in Figure. The gate and substrate form the plates of a capacitor with the gate oxide as the dielectric. The positive gate voltage causes positive charge to accumulate on the gate electrode and negative charge on the substrate side. The latter manifests itself initially by repelling mobile holes. Hence, a depletion region is formed below the gate. This depletion region is similar to the one occurring in a *pn*-junction diode.



As the gate voltage increases, the potential at the silicon surface at some point reaches a critical value, where the semiconductor surface inverts to *n*-type material. This point marks the onset of a phenomenon known as *strong inversion* and occurs at a voltage equal to twice the *Fermi Potential*

$$\phi_F = -\phi_T \ln\left(\frac{N_A}{n_i}\right)$$

Further increases in the gate voltage produce no further changes in the depletion layer width, but result in additional electrons in the thin inversion layer directly under the oxide. These are drawn into the inversion layer from the heavily doped *n* + source region.
Hence, a continuous *n*-type channel is formed between the source and drain regions, the conductivity of which is modulated by the gate-source voltage. In the presence of an inversion layer, the charge stored in the depletion region is fixed and equals

$$Q_{B0} = \sqrt{2qN_A\varepsilon_{si}|-2\phi_F|}$$

This picture changes somewhat in case a substrate bias voltage *VSB* is applied (*VSB* is normally positive for *n*-channel devices). This causes the surface potential required for strong inversion to increase and to become |*–2fF* + *VSB*|. The charge stored in the depletion region now is expressed by Eq.

$$Q_B = \sqrt{2qN_A\varepsilon_{si}(|-2\phi_F + V_{SB}|)}$$

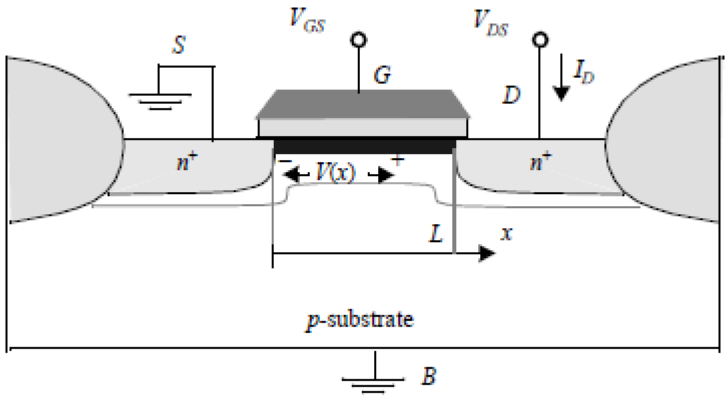
The value of *VGS* where strong inversion occurs is called the *threshold voltage VT.VT* is a function of several components, most of which are material constants such as the difference in work-function between gate and substrate material, the oxide thickness, the Fermi voltage, the charge of impurities trapped at the surface between channel and gate oxide, and the dosage of ions implanted for threshold adjustment. From the above arguments, it has become clear that the source-bulk voltage *VSB* has an impact on the threshold.
as well. Rather than relying on a complex (and hardly accurate) analytical expression for the threshold, we rely on an empirical parameter called *VT*0, which is the threshold voltage for *VSB* = 0, and is mostly a function of the manufacturing process. The threshold voltage under different body-biasing conditions can then be determined in the following manner,

$$V_T = V_{T0} + \gamma\left(\sqrt{|-2\phi_F + V_{SB}|} - \sqrt{|-2\phi_F|}\right)$$

The parameter g (gamma) is called the *body-effect coefficient*, and expresses the impact of changes in *VSB*. Observe that the threshold voltage has a **positive** value for a typical **NMOS** device, while it is **negative** for a normal **PMOS** transistor.

Resistive Operation

Assume now that *VGS* > *VT* and that a small voltage,*VDS*, is applied between drain and source. The voltage difference causes a current *ID* to flow from drain to source Figure. Using a simple analysis, a first-order expression of the current as a function of *V GS* and *VDS* can be obtained.



At a point *x* along the channel, the voltage is *V*(*x*), and the gate-to-channel voltage at that point equals *VGS* – *V*(*x*). Under the assumption that this voltage exceeds the threshold voltage all along the channel, the induced channel charge per unit area at point *x* can be computed.

$$Q_i(x) = -C_{ox}[V_{GS} - V(x) - V_T]$$

Cox stands for the capacitance per unit area presented by the gate oxide, and equals

$$C_{ox} = \frac{\varepsilon_{ox}}{t_{ox}}$$

with e*ox* = 3.97 ´ e*o* = 3.5 ´ 10-11 F/m the oxide permittivity, and *tox* is the thickness of the oxide. The latter which is 10 nm (= 100 Å) or smaller for contemporary processes. For an oxide thickness of 5 nm, this translates into an oxide capacitance of 7 fFm/m2.

The current is given as the product of the drift velocity of the carriersu*n* and the available charge. Due to charge conservation, it is a constant over the length of the channel. *W* is the width of the channel in a direction perpendicular to the current flow.

$$I_D = -\upsilon_n(x)Q_i(x)W$$

The electron velocity is related to the electric field through a parameter called the*m obility* m*n* (expressed in m2/V×s). The mobility is a complex function of crystal structure, and local electrical field. In general, an empirical value is used.

$$\upsilon_n = -\mu_n\xi(x) = \mu_n\frac{dV}{dx}$$

Combining the equations we have

$$I_D dx = \mu_n C_{ox} W(V_{GS} - V - V_T)dV$$

Integrating the equation over the length of the channel*L* yields the voltage-current relation of the transistor.

$$I_D = k'_n\frac{W}{L}\left[(V_{GS} - V_T)V_{DS} - \frac{V_{DS}^2}{2}\right] = k_n\left[(V_{GS} - V_T)V_{DS} - \frac{V_{DS}^2}{2}\right]$$

Kn’ is called the *process transconductance parameter* and equals

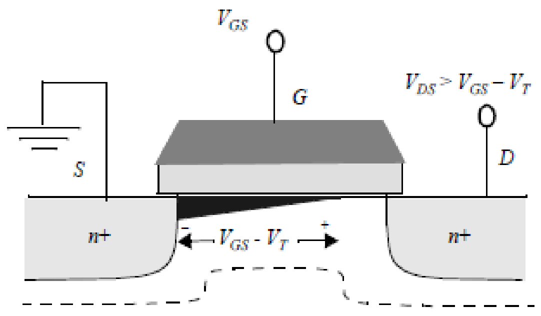
$$k'_n = \mu_n C_{ox} = \frac{\mu_n \varepsilon_{ox}}{t_{ox}}$$

The product of the process transconductance and the (*W/L*) ratio of an (NMOS) transistor is called the *gain factor kn* of the device. For smaller values of *VDS*, the quadratic factor in Eq. for I_D can be ignored, and we observe a linear dependence between *VDS* and *ID*. The operation region where Eq. for I_D holds is hence called the *resistive* or *linear* region. One of its main properties is that it displays a continuous conductive channel between source and drain regions.

The Saturation Region

As the value of the drain-source voltage is further increased, the assumption that the channel voltage is larger than the threshold all along the channel ceases to hold. This happens when *VGS* - V(*x*) < *VT*. At that point, the induced charge is zero, and the conducting channel disappears or is *pinched off*.

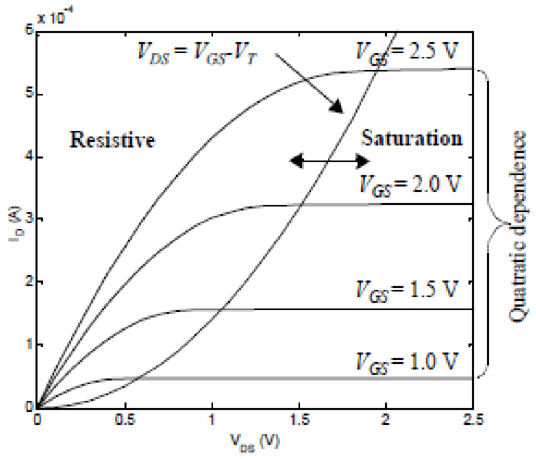
This is illustrated in Figure, which shows (in an exaggerated fashion) how the channel thickness gradually is reduced from source to drain until pinch-off occurs. No channel exists in the vicinity of the drain region. Obviously, for this phenomenon to occur, it is essential that the pinch-off condition be met at the drain region, or



$$V_{GS} - V_{DS} \leq V_T.$$

Under those circumstances, the transistor is in the *saturation* region, and Eq. for I_D no longer holds. The voltage difference over the induced channel (from the pinch-off point to the source) remains fixed at V_{GS} - V_T, and consequently, the current remains constant (or saturates). Replacing V_{DS} by V_{GS} - V_T in Eq. for I_D yields the drain current for the saturation mode. It is worth observing that, to a first agree, the current is no longer a function of V_{DS}. Notice also the *squared dependency* of the drain current with respect to the control voltage V_{GS}.

$$I_D = \frac{k'_n}{2}\frac{W}{L}(V_{GS} - V_T)^2$$



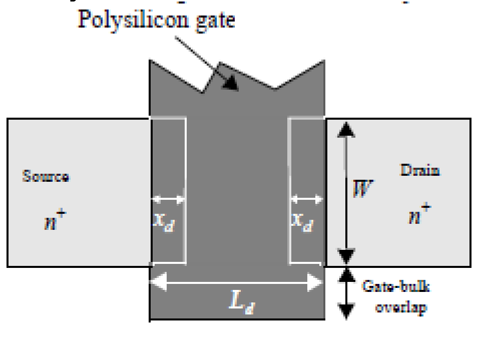
ii. Dynamic Behavior

The dynamic response of a MOSFET transistor is a sole function of the time it takes to (dis)charge the parasitic capacitances that are intrinsic to the device, and the extra capacitance introduced by the interconnecting lines . A profound understanding of the nature and the behavior of these intrinsic capacitances is essential for the designer of high-quality digital integrated circuits. They originate from three sources: the basic MOS structure, the channel charge, and the depletion regions of the reverse-biased *pn*-junctions of drain and source. Aside from the MOS structure capacitances, all capacitors are nonlinear and vary with the applied voltage, which makes their analysis hard

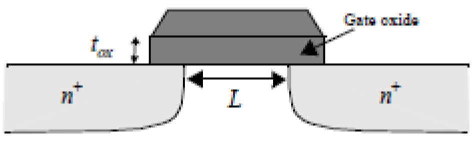
MOS Structure Capacitances

The gate of the MOS transistor is isolated from the conducting channel by the gate oxide that has a capacitance per unit area equal to *Cox* = e*ox* / *tox*. We learned earlier that from a *IV* perspective it is useful to have *Cox* as large as possible, or to keep the oxide thickness very thin. The total value of this capacitance is called the *gate capacitance Cg* and can be decomposed into two elements, each with a different behaviour. Obviously, one part of *C g* contributes to the channel charge, and is discussed in a subsequent section. Another part is solely due to the topological structure of the transistor. This component is the subject of the remainder of this section.

Consider the transistor structure of Figure. Ideally, the source and drain diffusion should end right at the edge of the gate oxide. In reality, both source and drain tend to extend somewhat below the oxide by an amount *xd*, called the *lateral diffusion*. Hence, the effective channel of the transistor *L* becomes shorter than the drawn length *Ld* (or the length the transistor was originally designed for) by a factor of D*L* = *2xd*. It also gives rise to a parasitic capacitance between gate and source (drain) that is called the *overlap capacitance.* This capacitance is strictly linear and has a fixed value



(a) Top view



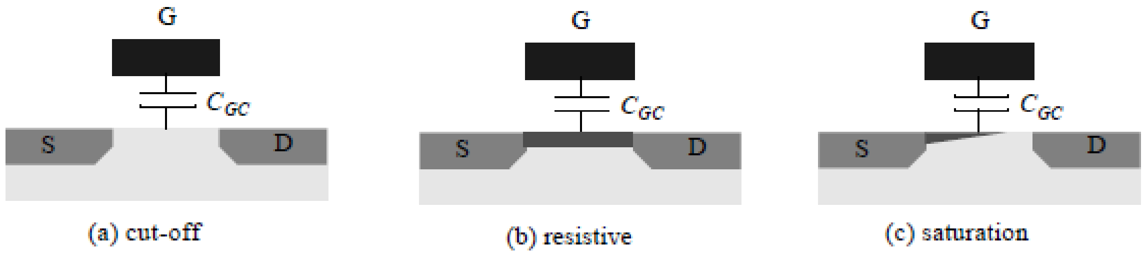
(b) Cross section

$$C_{GSO} = C_{GDO} = C_{ox} x_d W = C_o W$$

Since *xd* is a technology-determined parameter, it is customary to combine it with the oxide capacitance to yield the overlap capacitance per unit transistor width *C o* (more specifically, *Cgso* and *Cgdo*).

Channel Capacitance

Perhaps the most significant MOS parasitic circuit element, the gate-to-channel capacitance *CGC* varies in both magnitude and in its division into three components *C GCS*, *CGCD*, and *CGCB* (being the gate-to-source, gate-to-drain, and gate-to-body capacitances, respectively), depending upon the operation region and terminal voltages. This varying distribution is best explained with the simple diagrams of Figure. When the transistor is in cut-off (a), no channel exists, and the total capacitance *CGC* appears between gate and body. In the resistive region (b), an inversion layer is formed, which acts as a conductor between source and drain. Consequently, *CGCB = 0* as the body electrode is shielded from the gate by the channel. Symmetry dictates that the capacitance distributes evenly between source and drain. Finally, in the saturation mode (c), the channel is pinched off. The capacitance between gate and drain is approximately zero, and so is the gate-body capacitance. All the capacitance hence is between gate and source.



The gate-capacitance components are nonlinear and varying with the operating voltages. To make a first-order analysis possible, we will use a simplified model with a constant capacitance value in each region of operation
in the remainder of the text. The assumed values are summarized in Table.

Operation Region	C_{GCB}	C_{GCS}	C_{GCD}	C_{GC}	C_G
Cutoff	$C_{ox}WL$	0	0	$C_{ox}WL$	$C_{ox}WL+2C_oW$
Resistive	0	$C_{ox}WL / 2$	$C_{ox}WL / 2$	$C_{ox}WL$	$C_{ox}WL+2C_oW$
Saturation	0	$(2/3)C_{ox}WL$	0	$(2/3)C_{ox}WL$	$(2/3)C_{ox}WL+2C_oW$

Junction Capacitances

A final capacitive component is contributed by the reverse-biased source-body and drain body *pn*-junctions. The depletion-region capacitance is nonlinear and decreases when the reverse bias is raised as discussed earlier. To understand the components of the junction capacitance (often called the *diffusion capacitance*), we must look at the source (drain) region and its surroundings. The detailed picture, shown in Figure, shows that the junction consists of two components:

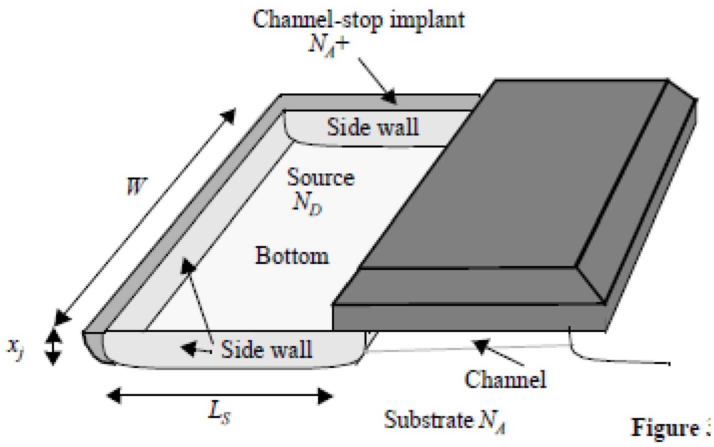


Figure :

The *bottom-plate* junction, which is formed by the source region (with doping *ND*) and the substrate with doping *NA*. The total depletion region capacitance for this component equals *Cbottom* = *CjWLS*, with *Cj* the junction capacitance per unit area as given by Eq. . As the bottom-plate junction is typically of the abrupt type, the grading coefficient *m* approaches 0.5.

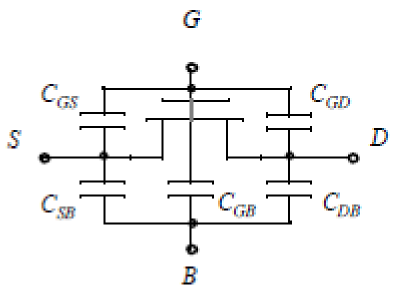
• The *side-wall* junction, formed by the source region with doping*N D* and the *p*+ channel-stop implant with doping level *NA* +. The doping level of the stopper is usually larger than that of the substrate, resulting in a larger capacitance per unit area. The side-wall junction is typically graded, and its grading coefficient varies from 0.33 to 0.5. Its capacitance value equals *Csw* = *C'jswxj* (*W* + 2 ´ *Ls*). Notice that no side-wall capacitance is counted for the fourth side of the source region, as this represents the conductive channel.

Since *xj*, the junction depth, is a technology parameter, it is normally combined with *C'jsw* into a capacitance per unit perimeter *Cjsw* = *C'jswxj*. An expression for the total junction capacitance can then be derived,

$$C_{diff} = C_{bottom} + C_{sw} = C_j \times AREA + C_{jsw} \times PERIMETER$$
$$= C_j L_S W + C_{jsw}(2L_S + W)$$

Capacitive Device Model

All the above contributions can be combined in a single capacitive model for the MOS transistor, which is shown Figure. Its components are readily identified on the basisof the preceding discussions.



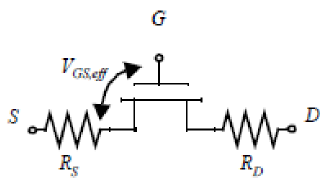
$$C_{GS} = C_{GCS} + C_{GSO};\ C_{GD} = C_{GCD} + C_{GDO};\ C_{GB} = C_{GCB}$$
$$C_{SB} = C_{Sdiff};\ C_{DB} = C_{Ddiff}$$

Source-Drain Resistance

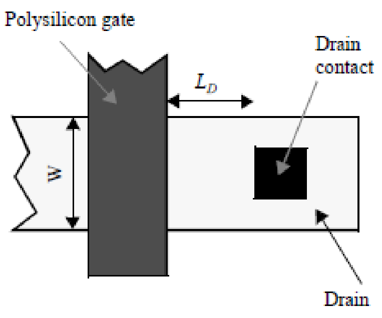
The performance of a CMOS circuit may further be affected by another set of parasitic elements, being the resistances in series with the drain and source regions, as shown in Figure . This effect become more pronounced when transistors are scaled down, as this leads to shallower junctions and smaller contact openings become smaller. The resistance of the drain (source) region can be expressed as

$$R_{S,D} = \frac{L_{S,D}}{W} R_{\square} + R_C$$

with *RC* the contact resistance, *W* the width of the transistor, and *LS,D* the length of the source or drain region . *R*o is the *sheet resistance* per square of the drain source diffusion, and ranges from 20 to 100 W/o. Observe that the resistance of a square of material is constant, independent of its size.



(a) Modeling the series resistance



(b) Parameters of the series resistance

Non ideal I-V Effects

Channel-Length Modulation

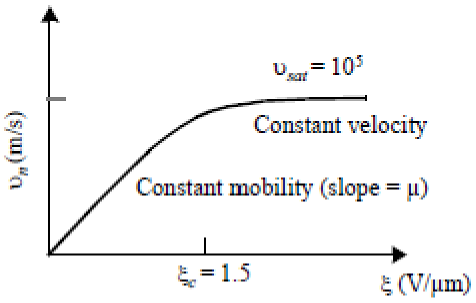
The equation seems to suggest that the transistor in the saturation mode acts as a perfect current source — or that the current between drain and source terminal is a constant, independent of the applied voltage over the terminals. This not entirely correct. The effective length of the conductive channel is actually modulated by the applied V_{DS}: increasing V_{DS} causes the depletion region at the drain junction to grow, reducing the length of the effective channel. As can be observed from Eq. For I_D in the saturation region, the current increases when the length factor L is decreased. A more accurate description of the current of the MOS transistor is therefore given in Eq.

$$I_D = I_D'(1 + \lambda V_{DS})$$

with I_D' the current expressions derived earlier, and l an empirical parameter, called the *channel-length modulation*. Analytical expressions for l have proven to be complex and inaccurate. l varies roughly with the inverse of the channel length. In shorter transistors, the drain-junction depletion region presents a larger fraction of the channel, and the channel-modulation effect is more pronounced. It is therefore advisable to resort to long-channel transistors if a high-impedance current source is needed.

Velocity Saturation

The behaviour of transistors with very short channel lengths (called *short-channel devices*) deviates considerably from the resistive and saturated models, presented in the previous paragraphs. The main culprit for this deficiency is the *velocity saturation* effect. Eq. For velocity states that the velocity of the carriers is proportional to the electrical field, independent of the value of that field. In other words, the carrier mobility is a constant. However, at high field strengths, the carriers fail to follow this linear model. In fact, when the electrical field along the channel reaches a critical value x*c*, the velocity of the carriers tends to saturate due to scattering effects (collisions suffered by the carriers). This is illustrated in Figure



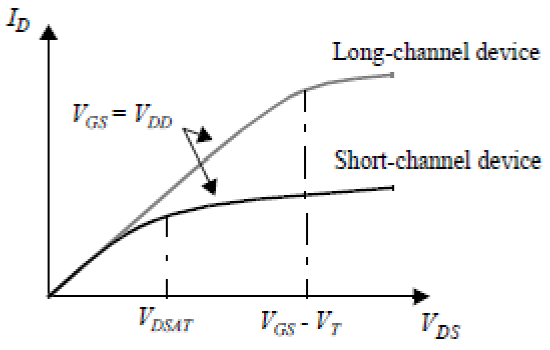
For *p*-type silicon, the critical field at which electron saturation occurs is around 1.5 □□106 V/m (or 1.5 V/□m), and the saturation velocity □*sat* approximately equals 105 m/s. The velocity as a function of the electrical field, plotted in Figure , can be roughly approximated by the following expression:

$$\upsilon = \frac{\mu_n \xi}{1 + \xi/\xi_c} \quad \text{for} \quad \xi \le \xi_c$$

$$= \upsilon_{sat} \quad \text{for} \quad \xi \ge \xi_c$$

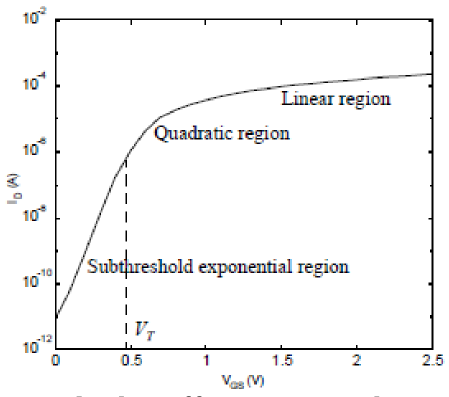
- For a short-channel device and for large enough values of *VGT*, □(*VGT*) is substantially smaller than 1, hence *VDSAT* < *VGT*. The device enters saturation before V_{DS} reaches V_{GS} - V_T. Short-channel devices therefore experience an extended saturation region, and tend to operate more often in saturation conditions than their long-channel counterparts, as is illustrated in Figure.

- The saturation current *IDSAT* displays a *linear dependence* with respect to the gate source voltage *VGS*, which is in contrast with the squared dependence in the long channel device. This reduces the amount of current a transistor can deliver for a given control voltage. On the other hand, reducing the operating voltage does not have such a significant effect in submicron devices as it would have in a long-channel transistor.



Subthreshold Conduction

A closer inspection of the *ID-VGS* curves of Figure 3.20 reveals that the current does not drop abruptly to 0 at *VGS* = *VT*. It becomes apparent that the MOS transistor is already partially conducting for voltages below the threshold voltage. This effect is called *su bthreshold* or *weak-inversion* conduction. The onset of strong inversion means that ample carriers are available for conduction, but by no means implies that no current at all can flow for gate-source voltages below *VT*, although the current levels are small under those conditions. The transition from the on- to the off-condition is thus not abrupt, but gradual.



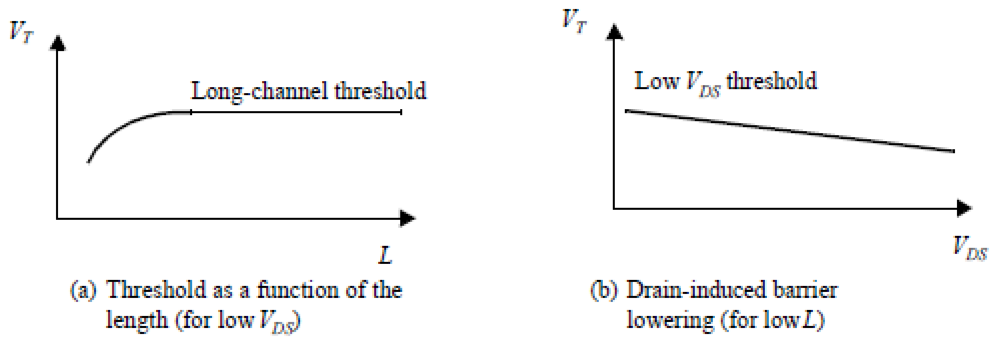
To study this effect in somewhat more detail, we redraw the*I D* versus *VGS* curve of Figure on a logarithmic scale as shown in Figure. This confirms that the current does not drop to zero immediately for*VGS* < *VT*, but actually decays in an exponential fashion, similar to the operation of a bipolar transistor. 2 In the absence of a conducting channel, the *n*+ (source) - *p* (bulk) - *n*+ (drain) terminals actually form a parasitic bipolar transistor. The current in this region can be approximated by the expression.

$$I_D = I_S e^{\frac{V_{GS}}{nkT/q}}\left(1 - e^{-\frac{V_{DS}}{kT/q}}\right)$$

Threshold Variations

the threshold voltage is only a function of the manufacturing technology and the applied body bias *VSB*. The threshold can therefore be considered as a constant over all NMOS (PMOS) transistors in a design. As the device dimensions are reduced, this model becomes inaccurate,

and the threshold potential becomes a function of*L*, *W*, and *VDS*. Two-dimensional second-order effects that were ignorable for long-channel devices suddenly become significant.
In the traditional derivation of the *VTO*, for instance, it is assumed that the channel depletion region is solely due to the applied gate voltage and that all depletion charge beneath the gate originates from the MOS field effects. This ignores the depletion regions of the source and reverse-biased drain junction, which become relatively more important with shrinking channel lengths. Since a part of the region below the gate is already depleted (by the source and drain fields), a smaller threshold voltage suffices to cause strong inversion. In other words, *VT0* decreases with *L* for short-channel devices (Figure a . A similar effect can be obtained by raising the drain-source (bulk) voltage, as this increases the width of the drain-junction depletion region. Consequently, the threshold decreases with increasing *VDS*. This effect, called the *drain-induced barrier lowering*, or *DIBL*, causes the threshold potential to be a function of the operating voltages (Figure b). For high enough values of the drain voltage, the source and drain regions can even be shorted together, and normal transistor operation ceases to exist. The sharp increase in current that results from this effect, which is called *punch-through*, may cause permanent damage to the device and should be avoided. Punch-through hence sets an upper bound on the drain-source voltage of the transistor.

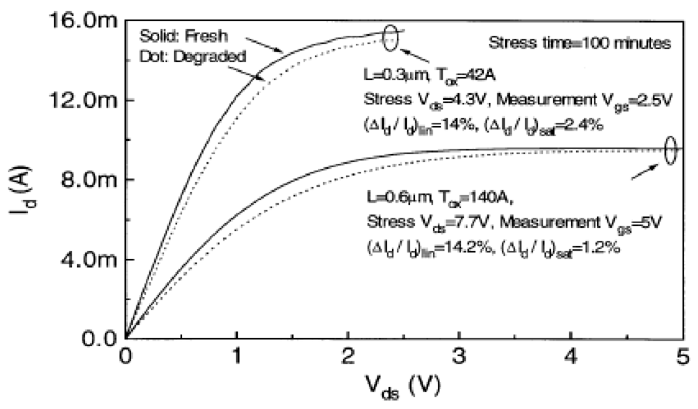


(a) Threshold as a function of the length (for low V_{DS})
(b) Drain-induced barrier lowering (for low L)

Hot-Carrier Effects

- Besides varying over a design, threshold voltages in short-channel devices also have the tendency to *drift over time*. This is the result of the *hot-carrier* effect [Hu92].
- Over the last decades, device dimensions have been scaled down continuously, while the power supply and the operating voltages were kept constant.
- The resulting increase in the electrical field strength causes an increasing velocity of the electrons, which can leave the silicon and tunnel into the gate oxide upon reaching a high-enough energy level. Electrons trapped in the oxide change the threshold voltage, typically increasing the thresholds of NMOS devices, while decreasing the *VT* of PMOS transistors. For an electron to become hot, an electrical field of at least 104 V/cm is necessary. This condition is easily met in devices with channel lengths around or below 1 mm.
- The hot-electron phenomenon can lead to a long-term reliability problem, where a circuit might degrade or fail after being in use for a while. This is illustrated in Figure , which shows the degradation in the *I -V* characteristics of an NMOS transistor after it has been subjected to extensive operation. State-of the-art MOSFET technologies therefore use specially-engineered drain and source regions to ensure that the peaks in the electrical fields are bounded, hence preventing carriers to

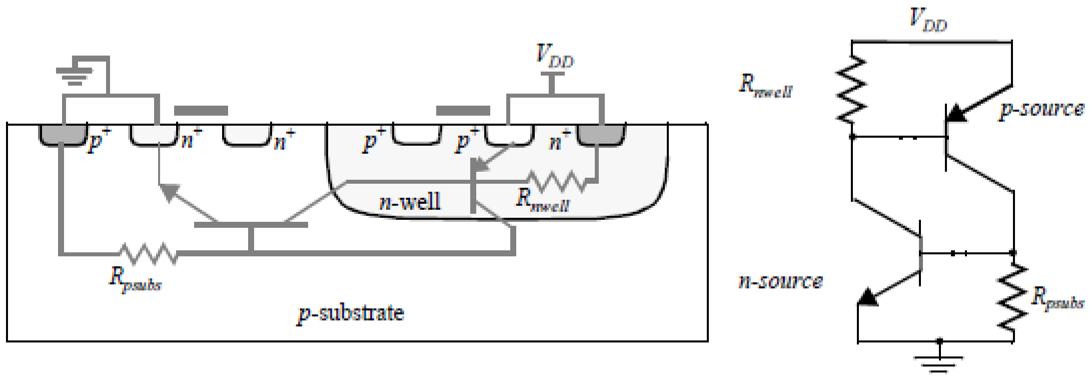
reach the critical values necessary to become hot. The reduced supply voltage that is typical for deep sub-micron technologies can in part be attributed to the necessity to keep hotcarrier effects under control.



CMOS Latchup

The MOS technology contains a number of intrinsic bipolar transistors. These are especially troublesome in CMOS processes, where the combination of wells and substrates results in the formation of parasitic *n-p-n-p* structures. Triggering these thyristor-like devices leads to a shorting of the *VDD* and *VSS* lines, usually resulting in a destruction of the chip, or at best a system failure that can only be resolved by power-down.

Consider the *n*-well structure of Figure a. The *n-p-n-p* structure is formed by the source of the NMOS, the *p*-substrate, the *n*-well and the source of the PMOS. A circuit equivalent is shown in Figure b. When one of the two bipolar transistors gets forward biased (e.g., due to current flowing through the well, or substrate), it feeds the base of the other transistor. This positive feedback increases the current until the circuit fails or burns out.



(a) Origin of latchup (b) Equivalent circuit

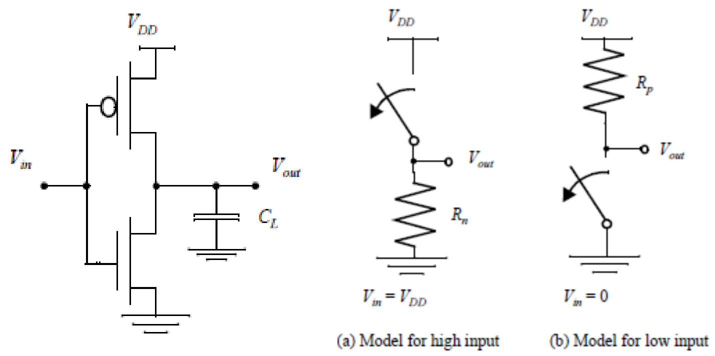
To avoid latchup, the resistances *Rnwell* and *Rpsubs* should be minimized. This can be achieved by providing numerous well and substrate contacts, placed close to the source connections of the NMOS/PMOS devices. Devices carrying a lot of current (such as transistors in the I/O drivers) should be surrounded by *guard rings*. These circular well/substrate contacts, positioned around the transistor, reduce the resistance even further and reduce the gain of the parasitic bipolars.

CMOS logic

The Static CMOS Inverter

Figure 5.1 shows the circuit diagram of a static CMOS inverter. the transistor is nothing more than a switch with an infinite off resistance (for |*VGS*| < |*VT*|), and a finite on-resistance (for |*VGS*| > |*VT*|). This leads to the following interpretation of the inverter. When *Vin* is high and equal to *VDD*, the NMOS transistor is on, while the PMOS is off. This yields the equivalent circuit of Figure a. A direct path exists between *Vout* and the ground node, resulting in a steady-state value of 0 V. On the other hand, when the input voltage is low (0 V), NMOS and

PMOS transistors are off and on, respectively. The equivalent circuit of Figure b shows that a path exists between *VDD* and *Vout*, yielding a high output voltage. The gate clearly functions as an inverter.



(a) Model for high input (b) Model for low input

A number of other important properties of static CMOS can be derived from this switch level view:

• The high and low output levels equal *VDD* and *GND*, respectively; in other words, the voltage swing is equal to the supply voltage. This results in high noise margins.

• The logic levels are not dependent upon the relative device sizes, so that the transistors can be minimum size. Gates with this property are called *ratioless*. This is in contrast with *ratioed logic*, where logic levels are determined by the relative dimensions of the composing transistors.

• In steady state, there always exists a path with finite resistance between the output and either *VDD* or *GND*. A well-designed CMOS inverter, therefore, has low *output impedance*, which makes it less sensitive to noise and disturbances. Typical values of the output resistance are in k□□range.

• The *input resistance* of the CMOS inverter is extremely high, as the gate of an MOS transistor is a virtually perfect insulator and draws no dc input current. Since the input node of the inverter only connects to transistor gates, the steady-state input current is nearly zero. A single inverter can theoretically drive an infinite number of
gates (or have an infinite fan-out) and still be functionally operational; however, increasing the fan-out also increases the propagation delay, as will become clear below. So, although fan-out does not have any effect on the steady-state behavior, it degrades the transient response.

No direct path exists between the supply and ground rails under steady-state operating conditions (this is, when the input and outputs remain constant). The absence of current flow (ignoring leakage currents) means that the gate does not consume any static power.

DC Transfer characteristics

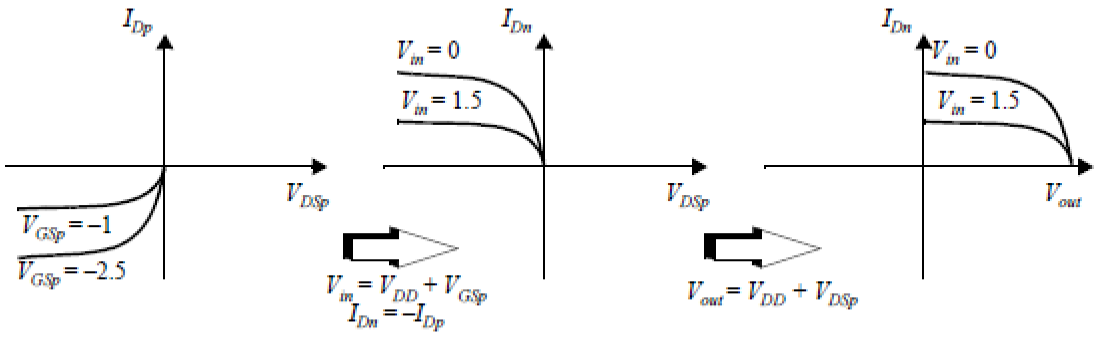
The nature and the form of the voltage-transfer characteristic (VTC) can be graphically deduced by superimposing the current characteristics of the NMOS and the PMOS devices. Such a graphical construction is traditionally called *a load-line plot*. It requires that the *I-V* curves of the NMOS and PMOS devices are transformed onto a common coordinate set. We have selected the input voltage *Vin*, the output voltage *Vout* and the NMOS drain current *IDN* as the variables of choice. The PMOS *I-V* relations can be translated into this variable space by the following relations (the subscripts *n* and *p* denote the NMOS and PMOS devices, respectively):

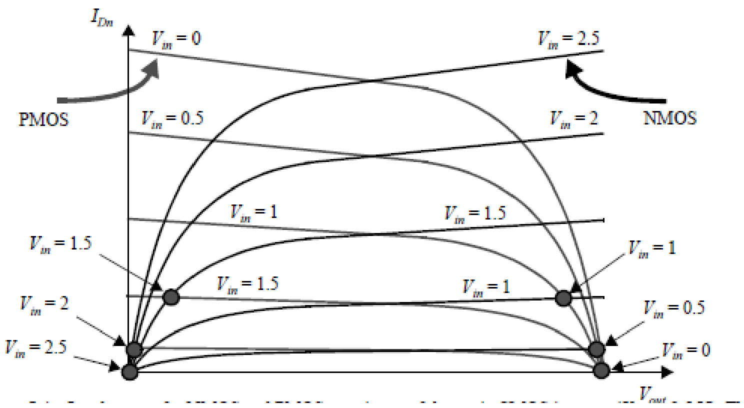
$$I_{DSp} = -I_{DSn}$$

$$V_{GSn} = V_{in} \; ; \; V_{GSp} = V_{in} - V_{DD}$$

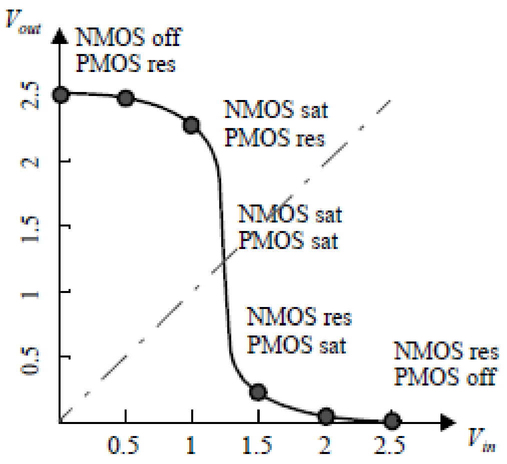
$$V_{DSn} = V_{out} \; ; \; V_{DSp} = V_{out} - V_{DD}$$

The load-line curves of the PMOS device are obtained by a mirroring around the *x*-axis and a horizontal shift over *VDD*. This procedure is outlined in Figure , where the subsequent steps to adjust the original PMOS *I-V* curves to the common coordinate set *Vin*, *Vout* and *IDn* are illustrated.



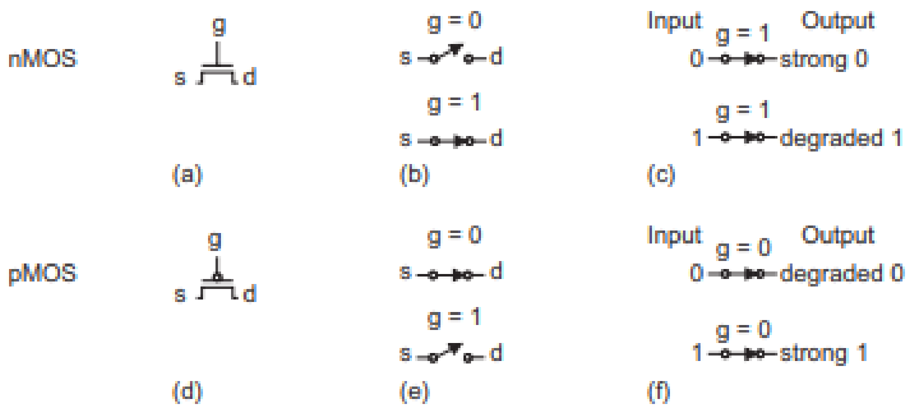


The resulting load lines are plotted in above Figure. For a dc operating points to be valid, the currents through the NMOS and PMOS devices must be equal. Graphically, this means that the dc points must be located at the intersection of corresponding load lines. A number of those points (for *Vin* = 0, 0.5, 1, 1.5, 2, and 2.5 V) are marked on the graph. As can be observed, all operating points are located either at the high or low output levels.
The VTC of the inverter hence exhibits a very narrow transition zone. This results from the high gain during the switching transient, when both NMOS and PMOS are simultaneously on, and in saturation. In that operation region, a small change in the input voltage results in a large output variation. All these observations translate into the VTC of below Figure.



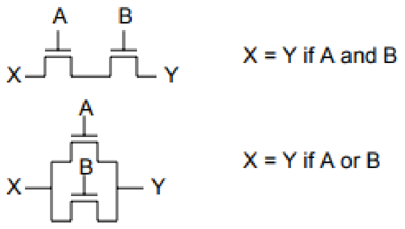
Pass Transistor Logic

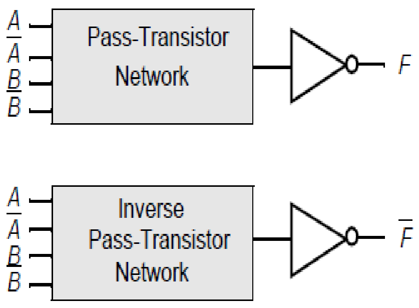
- nMOS pass strong 0 – But degraded or weak 1 ;• pMOS pass strong 1 – But degraded or weak 0 . Thus nMOS are best for pull-down network



-

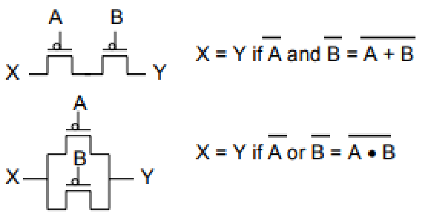
- Primary inputs drive both gate and source/drain terminals
- NMOS switch closes when the gate input is high

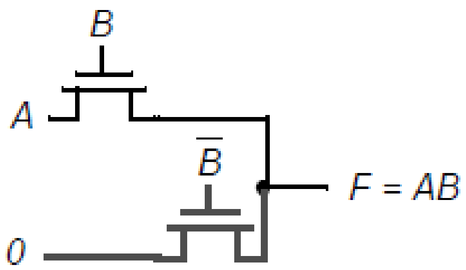




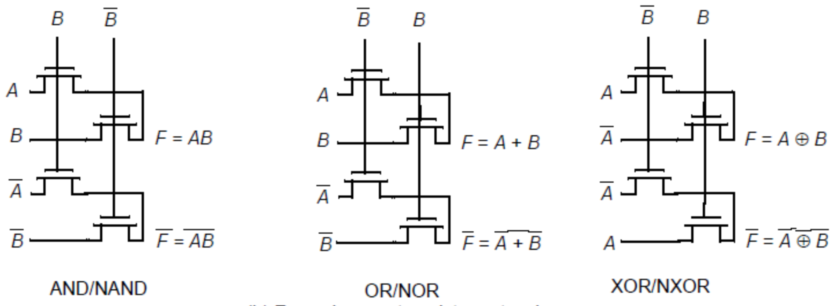
(a) Basic concept

- Remember - NMOS transistors pass a strong 0 but a weak 1
- Primary inputs drive both gate and source/drain terminals
- PMOS switch closes when the gate input is low


A
B
X
Y
$X = Y$ if $\overline{A}$ and $\overline{B} = \overline{A + B}$
A
B
X
Y
$X = Y$ if $\overline{A}$ or $\overline{B} = \overline{A \bullet B}$



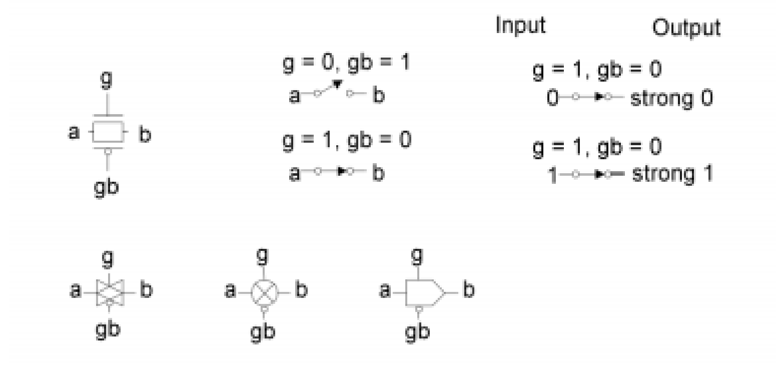
- Remember - PMOS transistors pass a strong 1 but a weak 0

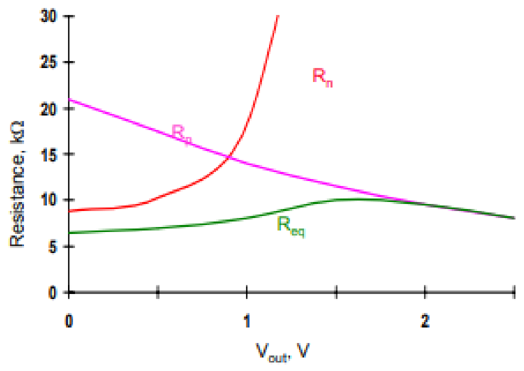


(b) Example pass-transistor networks

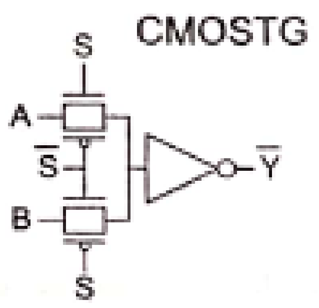
Transmission Gates

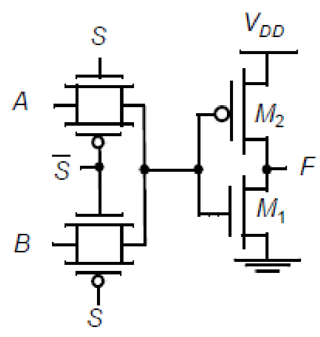
- Pass transistors produce degraded outputs
- *Transmission gates* pass both 0 and 1 well



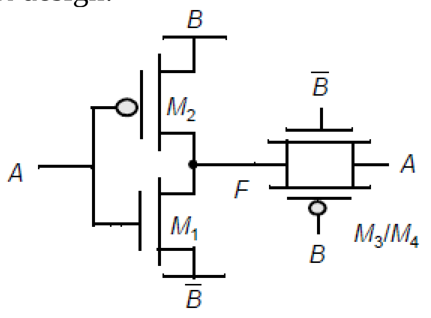


Multiplexer design:



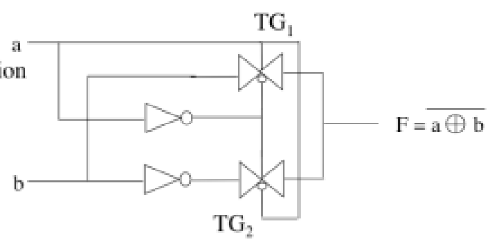


XOR design:



XNOR design:

- 8-transistor implementation



a	b	TG1	TG2	F
0	0	nonconducting	conducting	$\overline{B}$ (1)
0	1	nonconducting	conducting	B (0)
1	0	conducting	nonconducting	B (0)
1	1	conducting	nonconducting	B (1)

RC DELAY MODEL

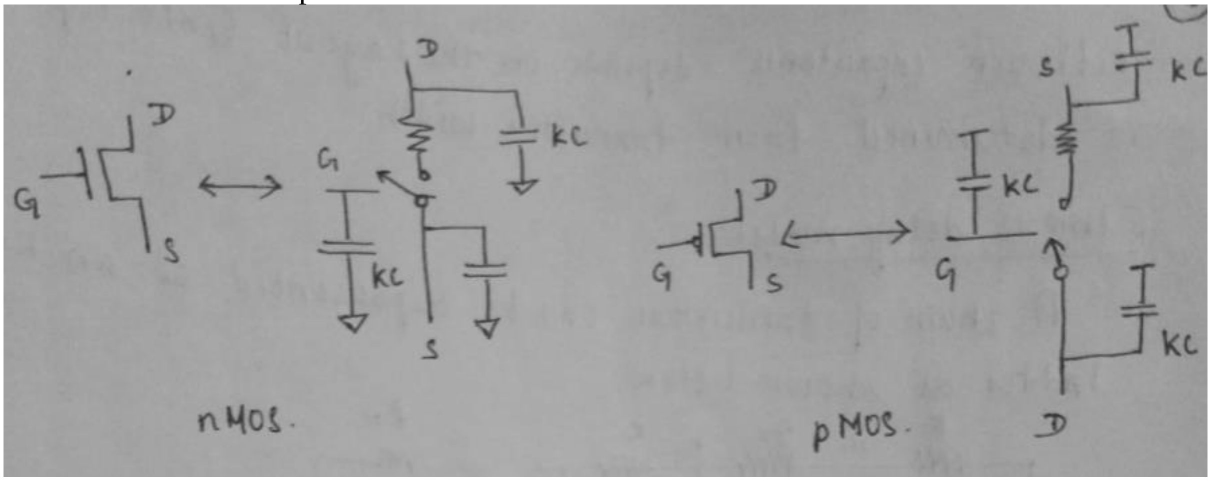
It treats transistors as switches in series with resistors.

A unit nMOS transistors has an effective resistance of R and pMOS transistor has an effective resistance 2R.

The size of the unit transistor refers to a transistor with maximum length and minimum contacted diffusion width.

AnMOS transistor of K time unit width has resistance R/K. Similarly a pMOS transistor K times unit width has resistance 2R/K[because of its lower mobility i.e is mobility µ increase and resistance R decreases].

Each transistor has a gate and diffusion capacitance cgs and Cdb and Csb. So a transistor of K times unit width has capacitance C.

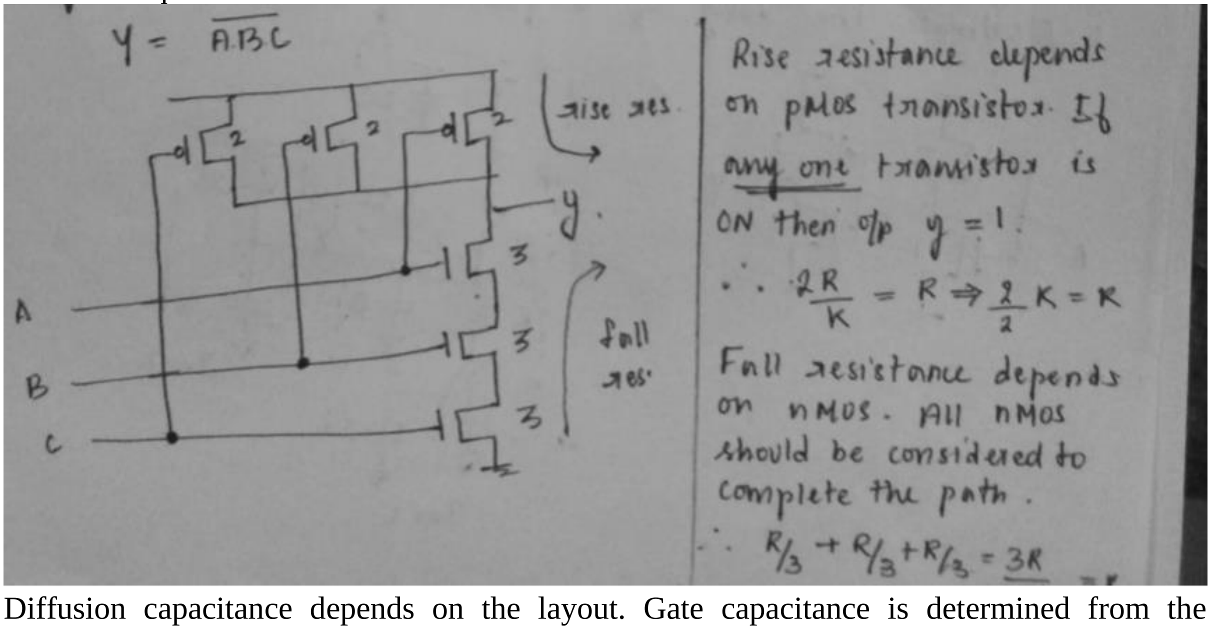


Effective resistance and capacitance

Parallel and series transistor combine like conventional resistors.

When multiple transistors are in series then the equivalent resistance is the sum of each individual resistance.When multiple transistors are in parallel the the equivalent re;sistance is lower if they are all ON.

Eg: Sketch 3input NAND gate with transistors width chosen to achieve effective rise and fall resistance equal to that of a unit inverter.



Rise resistance depends on pMos transistor. If any one transistor is ON then o/p $y = 1$.

$\therefore \frac{2R}{K} = R \Rightarrow \frac{2}{2}K = K$

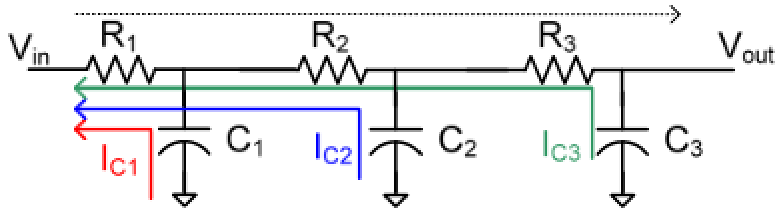
Fall resistance depends on nMOS. All nMos should be considered to complete the path.

$\therefore R/3 + R/3 + R/3 = 3R$

Diffusion capacitance depends on the layout. Gate capacitance is determined from the transistor width.

ELMORE DELAY MODEL

A chain of transistors can be represented as an RC ladder as shown below

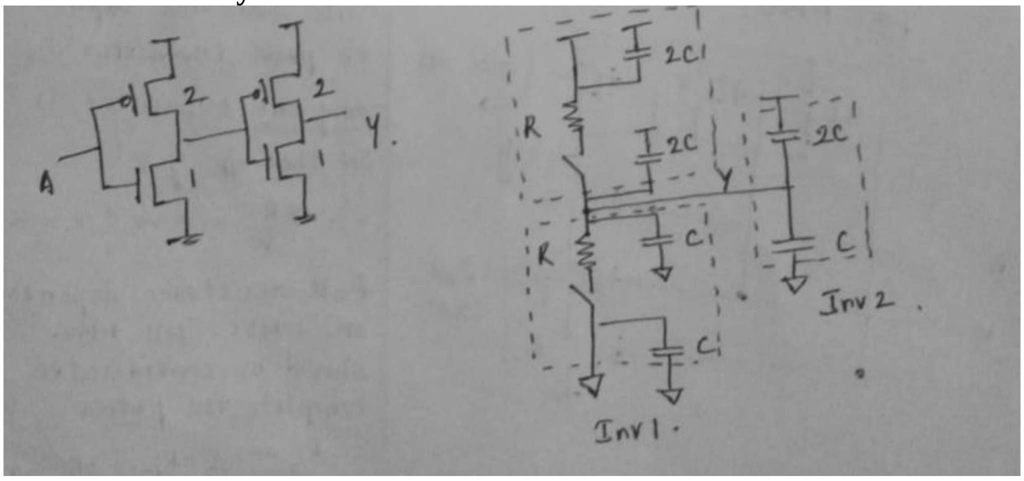


$$\boxed{\tau_{Elmore} = R_1C_1 + (R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3}$$

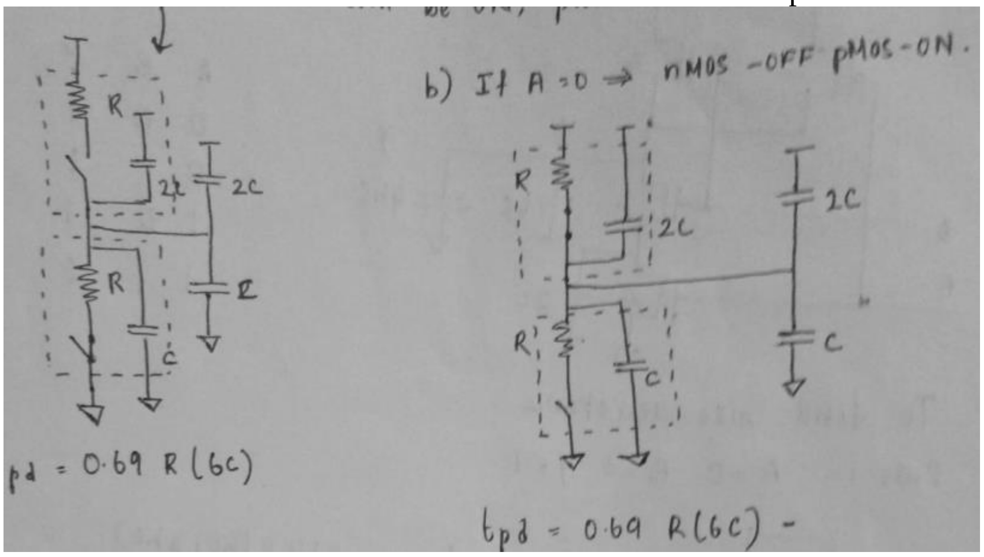
The Elmore delay model estimates the delay of an RC ladder. It is the sum of the resistance Rn-I (between that node and a supply) multiplied by the capacitance on the node.

Problem

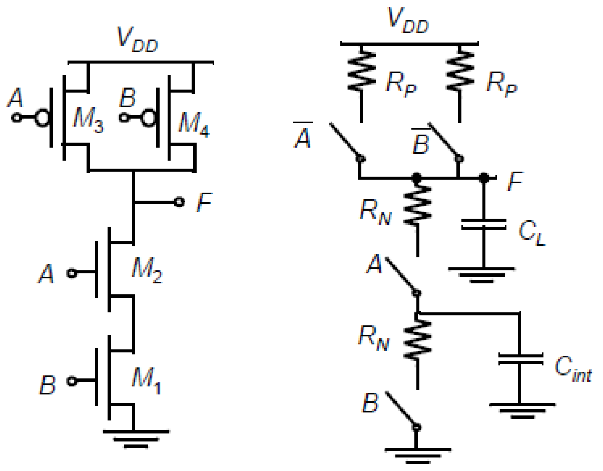
1. **Estimate the delay of a fanout of 1 inverter**



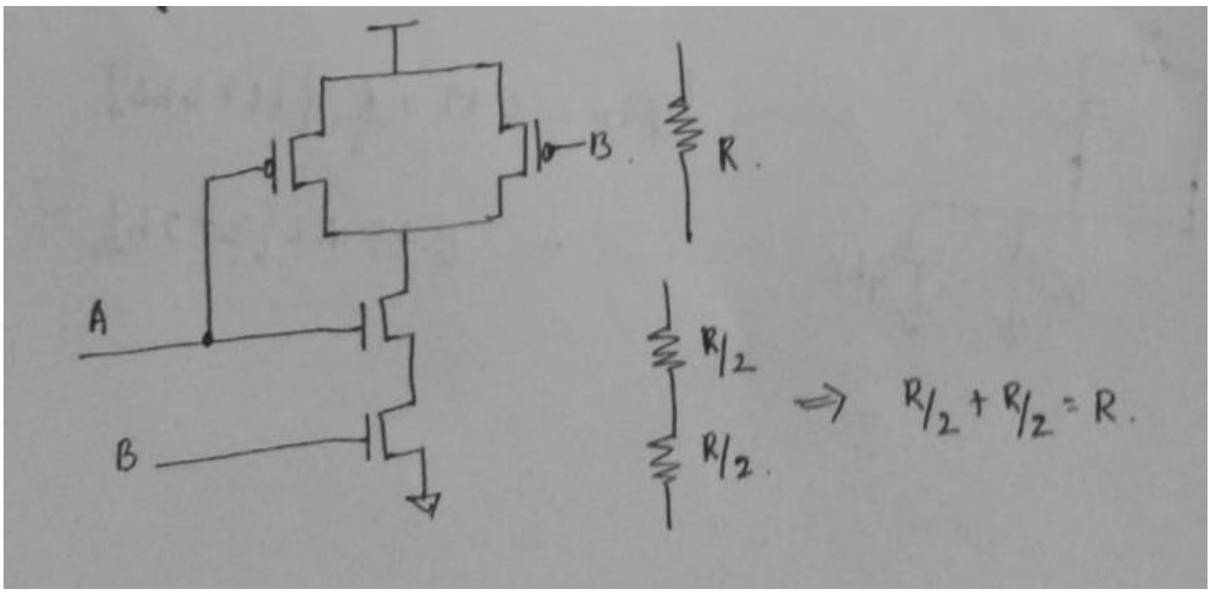
If A=1 then nMOS will be ON pMOS will be OFF

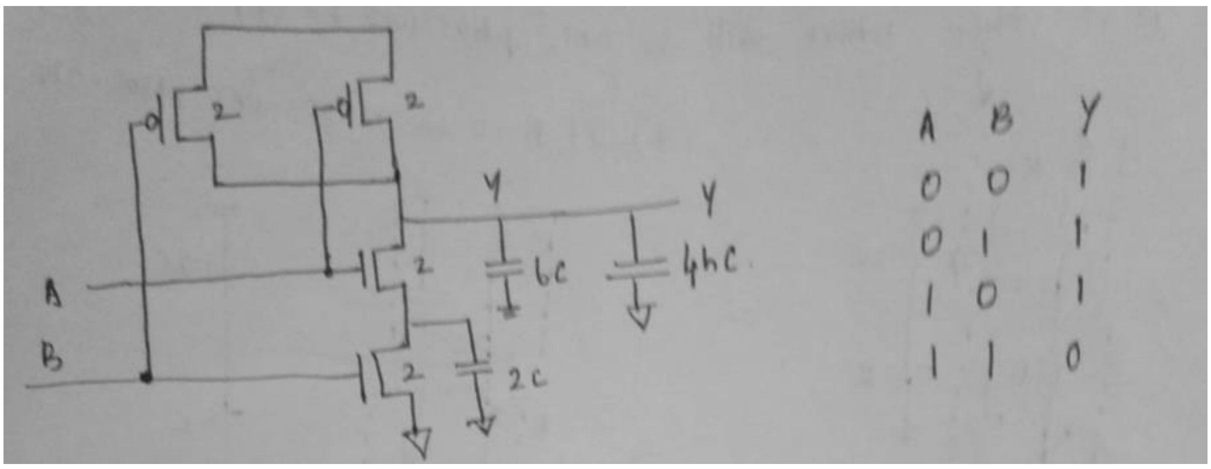


2.Sketch a 2 input NAND gate with transistor width chosen to achieve effective rise and fall resistance equal to the unit inverter. Compute the rising and falling propagation delay (in terms of R and C) of the NAND gate driving h identical NAND gates using the Elmore delay model. If C=2fF/μm and R=2.5 kΩ/μm in a 180nm process what is the delay of a fanout of 3 NAND gate.

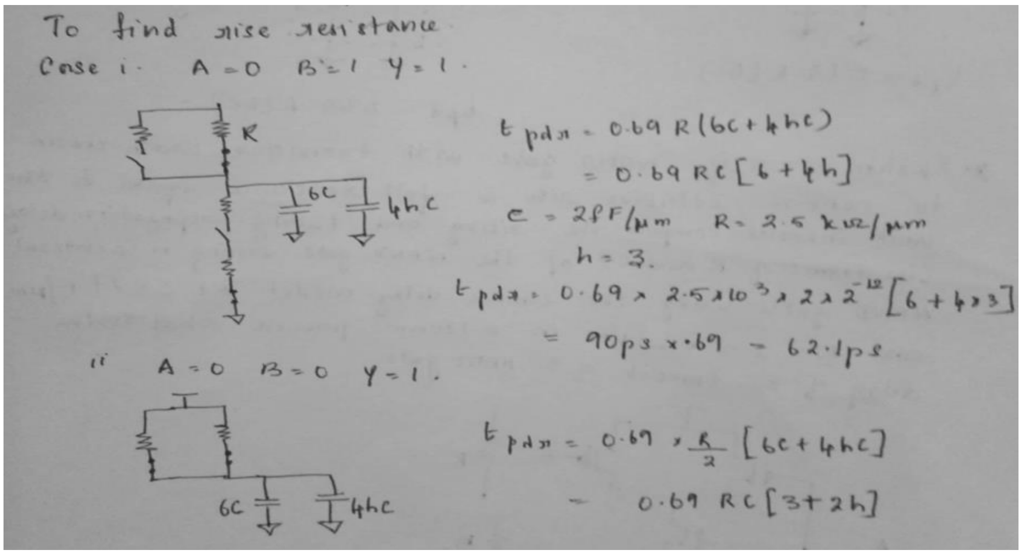


(a) Two-input NAND (b) RC equivalent model





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


K
6C
4hC
6C
4hC

To find rise resistance

Case i. A = 0 B = 1 Y = 1.

$t_{pdr} = 0.69\,R(6C + 4hC)$

$= 0.69\,RC[6 + 4h]$

$C = 2fF/\mu m$ $R = 2.5\,k\Omega/\mu m$

$h = 3.$

$t_{pdr} = 0.69 \times 2.5 \times 10^{3} \times 2 \times 2^{-12}[6 + 4 \times 3]$

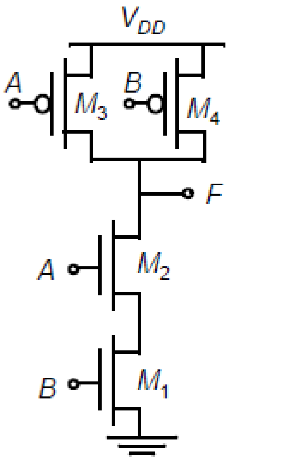
$= 90ps \times .69 = 62.1ps$

ii A = 0 B = 0 Y = 1.

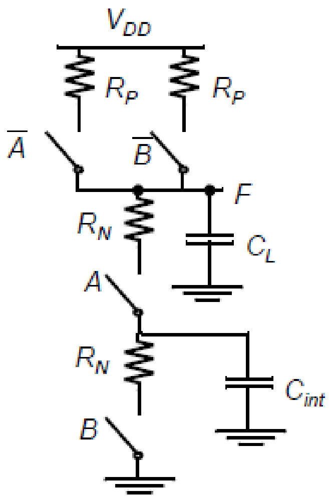
$t_{pdr} = 0.69 \times \frac{R}{2}[6C + 4hC]$

$= 0.69\,RC[3 + 2h]$

3.Consider the NAND gate of Figure . Assume NMOS and PMOS devices of 0.5mm/0.25mm and 0.75mm/0.25mm, respectively. This sizing should result in approximately equal worst-case rise and fall times (since the effective resistance of the pull-down is designed to be equal to the pull-up resistance).



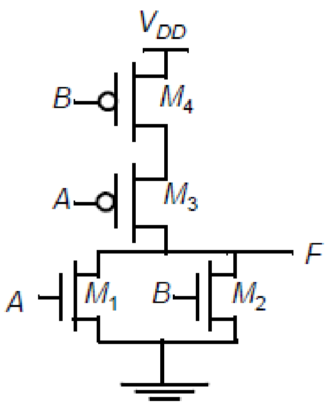
(a) Two-input NAND

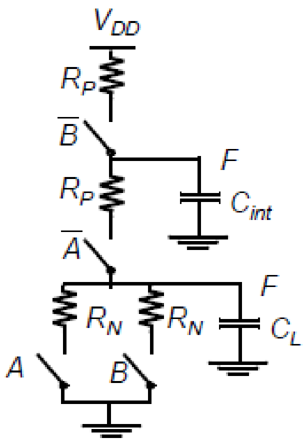


(b) RC equivalent model

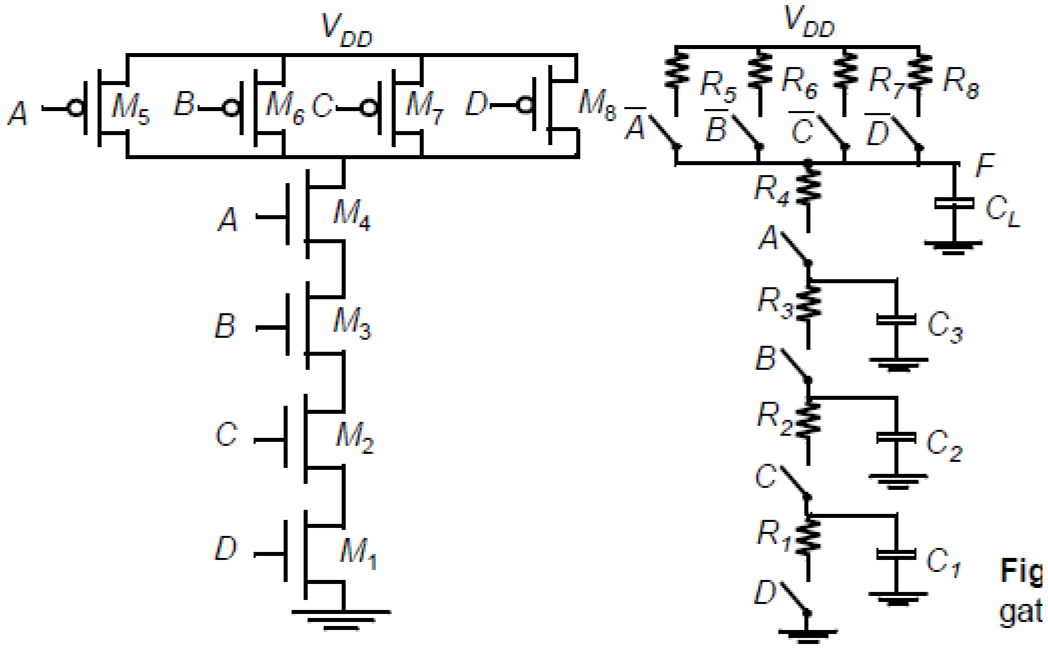
Input Data Pattern	Delay (psec)
A = B= 0→1	69
A = 1, B= 0→1	62
A= 0→1, B = 1	50
A=B=1→0	35
A=1, B = 1→0	76
A= 1→0, B = 1	57

Two Input NOR





Four input NAND gate and its RC model.



LINEAR DELAY MODEL

In general the propagation delay of a gate can be written as

d= Parasitic delay P, effort delay f.

Parasitic delay P = Inherent delay to the gate when no load is attached.

Effort delay or a stage effort depends on the complexity and fanout of the gate.

f= Logical effort g=fanout

f=g.h and d= p+;

i. Logical effort g

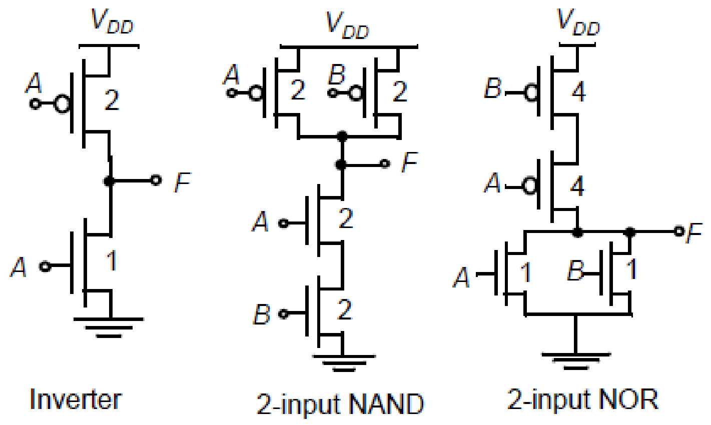
It is defined as the ratio between the input capacitances of the gate and the input capacitance of the inverter that can deliver the same amount.

g= Cin of gate/ Cin of inverter

Both delivering same current.

It represents the complexity of the gate.

Eg: Determine the logical effort of inverter NAND and NOR gate.



For inverter

Cin=3C

g=3C/3C =1

For 2 input NAND
Cin=4C
g=4C/3C =4/3

For 2 input NOR
Cin=5C
g=5C/3C =5/3

ii. Estimates of intrinsic delay factors

Gate type	p
Inverter	1
n-input NAND	n
n-input NOR	n
n-way multiplexer	$2n$
XOR, NXOR	$n2^{n-1}$

iii. Logical effort of common gate

Gate Type	Number of Inputs			
	1	2	3	n
Inverter	1			
NAND		4/3	5/3	(n+2)/3
NOR		5/3	7/3	(2n+1)/3
Multiplexer		2	2	2
XOR		4	12	

iv. Electrical effort or fanout h

It is defined as the ratio between the output capacitance to the input capacitance.
h= Cout/Cin where Cout is the capacitance of external load and Cin is the input capacitance of the gate.
D= f+p
=gh+p
G= Cout/Cin +p
Let Cin/g= drive= 1/R, then d=Cout/drive +p

v. Parasitic delay

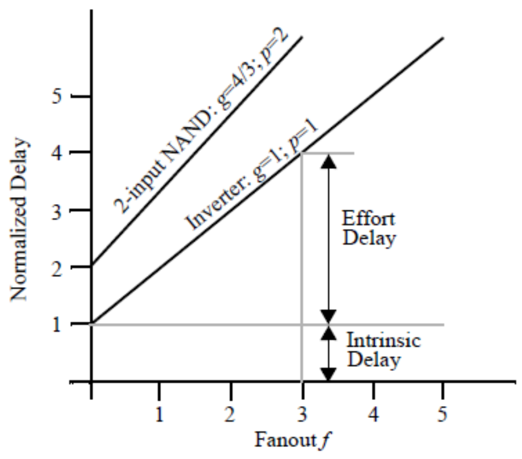
It is defined as the delay of the gate when it drives zero load. It can be estimated with RC delay models.
Mainly depends on diffusion capacitances.
The inverter has 3 units of diffusion capacitance on the outpu. So the parasitic delay is 3RC.

The normalised parasitic delay Pinv=1. It is defined as the ratio of diffusion capacitance to the gate capacitance.
The normalised parasitic delay of NAND and NOR gate is 6RC/3RC= 2 Pinv or simple 2.
Tpd=[$n^2/2+5/2n$]RC. It is the Elmore delay model.
The two main problems of complementary CMOS gates are,

1. Complexity of the gate (i.e fan in) increases and the overall capacitance also incrases which will affect the speed of the gate.
2. The number of transistors required to implement "N" inputs is 2N. This increases the implementation area.
3. The following techniques are used to reduce the dealay of larger fan in are
   Transistor sizing.
   Input reoedering.
   Progressive tr sizing.
   Logic reconstructing.

Term	Stage expression	Path expression
Logical effort	g (seeTable 1)	$G = \prod g_i$
Electrical effort	$h = \frac{C_{out}}{C_{in}}$	$H = \frac{C_{out\ (path)}}{C_{in\ (path)}}$
Branching effort	n/a	$B = \prod b_i$
Effort	$f = gh$	$F = GBH$
Effort delay	f	$D_F = \sum f_i$
Number of stages	1	N
Parasitic delay	p (seeTable 2)	$P = \sum p_i$
Delay	$d = f + p$	$D = D_F + P$

vi. Delay as a function of fanout for an inverter and a 2-input NAND.



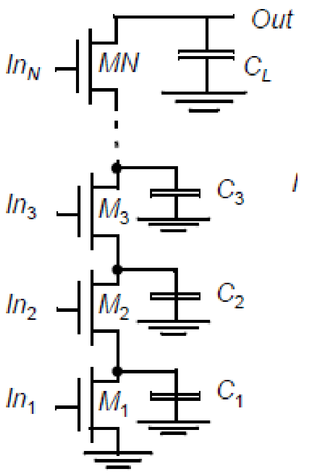
TRANSISTOR SIZING

The delay of larger fan in can be reduced by increaring the transistor sizes. This will lower the resistance of thr device which is in series and finally reduces delay.

However increasing the transistor sizes leads to larger parasitic capacitance which is not only affects the delay of the gate also increasing larger load to the next stage gate.

- **Progressive transistor sizing**

An alternate approach to uniform sizing is the progressive transistor sizing. Consider the following example in which the transistor are connected in series.



$$M_1 > M_2 > M_3 > M_N$$

The delay is given by= 0.69[R1C1+(R1+R2)C2+(R1+R2+R3)C3+(R1+R2+R3+R4)C4]

This equation indicates that the resistance R1 appears N times and R2 appears (N-1) times etc. So we have to made the transistor M size larger (morder to reduce R1) than M2 transistors M1>M2>M3>M4 is the progress sizing.

- **Input reordering**

The gate works fast when inner input signals arrives than the outer input signal.

The input which is closer to the power supply rail (Vdd or Gnd) is called outer input. The input which is closer to the output is called inner input.

In1= outer input and In3= Inner input.

Consider that In1 is the critical signal. Critical signal is the signal that arrives last of all inputs.

Consider Case(1): In3=1; In2=1

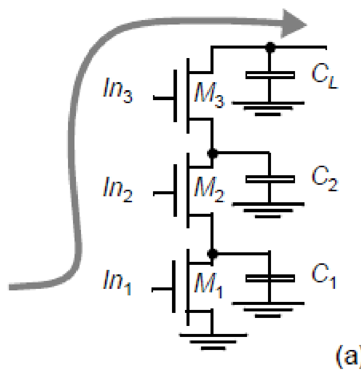
M1=OFF and M2=ON. So capacitance C1and C2 cannot discharge.

Therefore In1 arrives late. The delay of is

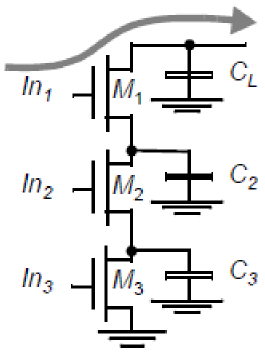
t= 0.69[R1C1+(R1+R2)C2+(R1+R2+R3)C3+(R1+R2+R3+R4)C4]

Now order the input signal In1

In this case(2):



(a)

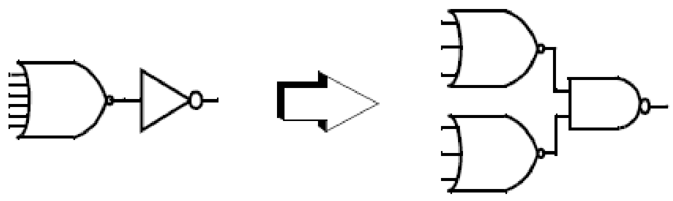


(b)

In3=1; In2=1.
Therefore M2=ON and M3=ON and M1=OFF till In1=1.
Here C2 and C1 can discharge. However Cl will discharge only after In1=1.
The delay is t=(R1+R2+R3)Cl*0.69

- **Logical reconstructing**

The delay of large number of inputs can be reduced by restricting the gate into 2/3 input gates which will speed up the gate.



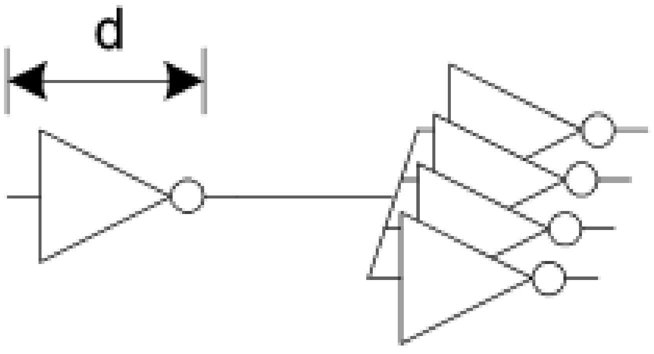
OPTIMIZING PERFORMANCE IN COMBINATIONAL NETWORKS

The method of logical effort provides a simple method to choose the fastest ciru=cuit topology and number of stages for a given function.
It allows the designer to quickly estimate the minimum possible delay for the given topology.
Logical effort is based on linear delay model. The logical effort method is used to

1. Estimate the delay of individual logic.
2. To predict the delay of multistage logic network.
3. Choose the best number of stages for a multistage network.

I. DELAY IN LOGIC GATE

The delay of a logic gate can be expressed by means of linear delay model.
Total delay d= gh+p
Propagation delay tpd= dƮ where g is the logical effort h is the electrical effort p is the parasitic delay V is the time constant for a particular process.
Eg: Estimate the delay of fanout of 4 inverter i.e inverter driving 4 identical copies. Assume the inverter is constructed in a 180nm process with Ʈ=15ps.

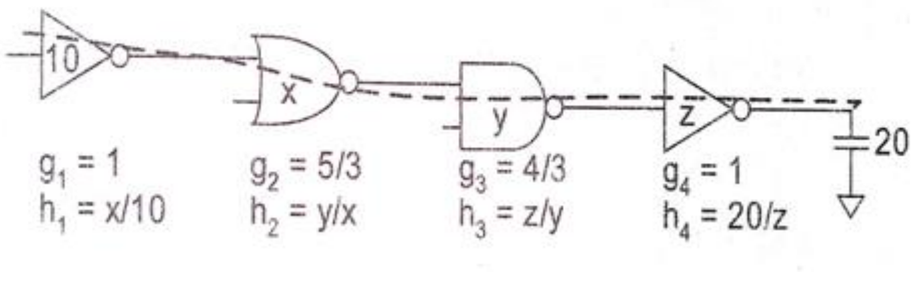


Logical effort g=1
Electrical effort h=12C/3C=4
Parasitic delay of an inverter p=1
The total delay d=gh+p
=1*4+1=5
tpd=5*15ps=75ps.

II. DELAY IN MULTISTAGE LOGIC NETWORK

Logical effort can be used to predict the delay of multistage network.

Logical effort is independent of size while electrical effort depends on size.



The path effort F is the product of the logical electrical and branching efforts of the path.

F= G.B.H;

Path logical effort G

It is defined as the product of the logical effort of each stage along the path.

G=pi gi

Path electrical effort H

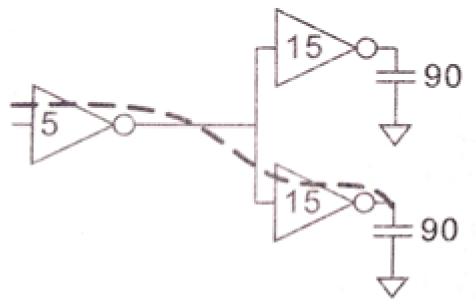
It is defined as the ratio of the output capacitance to the input capacitance. In the path electrical effort is defined as the product of stage electrical effort.

H= Cout(path)/Cin(path)

Branching effort

It is defined as the ratio of the total capacitance seen by a stage to the capacitance on the path.

b=(Con(path)+Coff(path))/Con(path)



B=15+15/15 = 2

Path branching effort

It is the product of branching effort between stages

B= pi bi

If a path has a N stages and each path has the same effort is given by

f^=gihi=F^1/N

Thus the minimum possible delay of an N stage path with the path effort and parasitic delay p is

D=NF^1/N+P

=Df+P= ∑ di

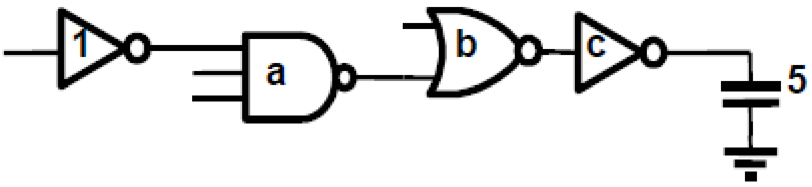
WherDf is th path b effort delay.

To achieve the least delay, the capacitance transformation formula used to find the best input capacitance of a gate for a given ouput capacitance.

Cin= (Couti.gi)/f^

Problems

Consider the logic network of Figure, which may represent the critical path of a more complex logic block. The output of the network is loaded with a capacitance which is 5 times larger than the input capacitance of the first gate, which is a minimum-sized inverter. find the path logical effort



$$G = 1 \times \frac{5}{3} \times \frac{5}{3} \times 1 = \frac{25}{9}$$

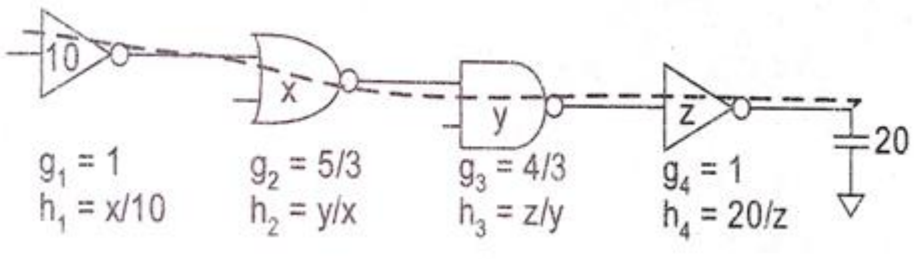
$H = FG = 125/9.$

h is $\sqrt[4]{H} = 1.93$

$f_1 = 1.93; f_2 = 1.93 \times (3/5) = 1.16; f_3 = 1.16; f_4 = 1.93.$

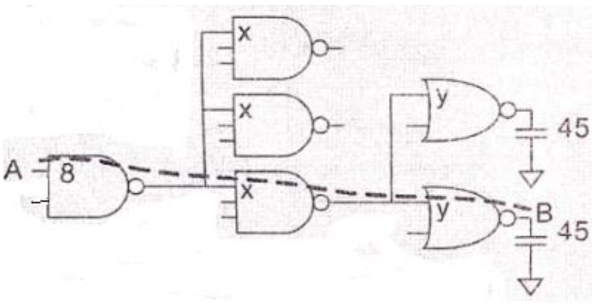
$a = f_1/g_2 = 1.16;\ b = f_1 f_2/g_3 = 1.34;\ c = f_1 f_2 f_3/g_4 = 2.60.$

Estimate the minimum delay of the path for the following diagram and choose transistor size to achieve the delay



The path logical effort G=g1g2g3g4=pi gi
=1.5/3.4/3.1=20/9
Path electrical effort H=20/10=2
Path branching effort B=1
The path effort F=GBH
=20/9*1*2=40/9
Then F^i/N= F^1/4=1.45
Path parasitic delay P=1+2+2+1=6
Minimum path delay D=NF^(1/N) +P
=4*1.451+6=11.8
tpd=D Ʈ=11.8 Ʈ
z=20*1/1.451
Cin4= Z=13.78
Cin3=y=Cin4g3/f^=12.66
Cin3=x=Cin3 g2/f^=14.54
Cin1=Cin2 g1/f^=10.02

2. **Estimate the minimum delay of the path from A to B for a given diagram and choose transistor sizes to achieve this delay. The initial NAND2 gate may present a load of 8λtransistor width on the input and output load is equivalent to 45 λ of transistor width.**



SOLUTION:

G= **πgi= g1.g2.g3= (4/3)*(**5/3)*(5/3)=100/27

B= (3x/x)*(2y/y)=6

H= Cout/Cin= 45/8

F= GBH = (100/27)*6*(45/8) = 125

F^(1/N) = f' = 125^(1/3) = 5

D= Nf' + P = (3*5)+7 =22

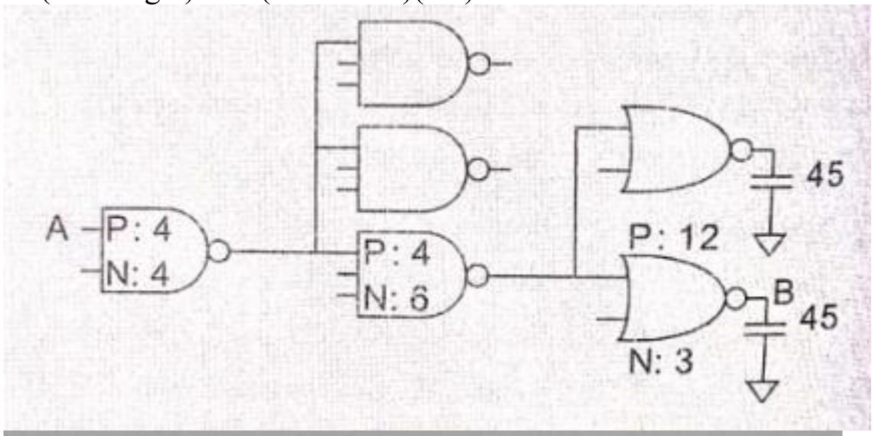
tpd= DƮ =22Ʈ

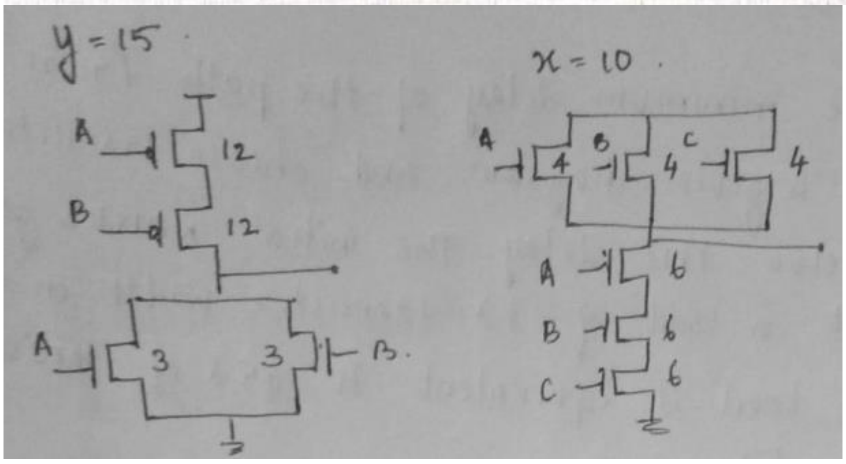
y= (Cout I * gi) / f' = Cin3 = (Cout* g3)/f' = 45(5/3)/5

y=15

x=Cin2=(15+5)(5/3)/5 = 10

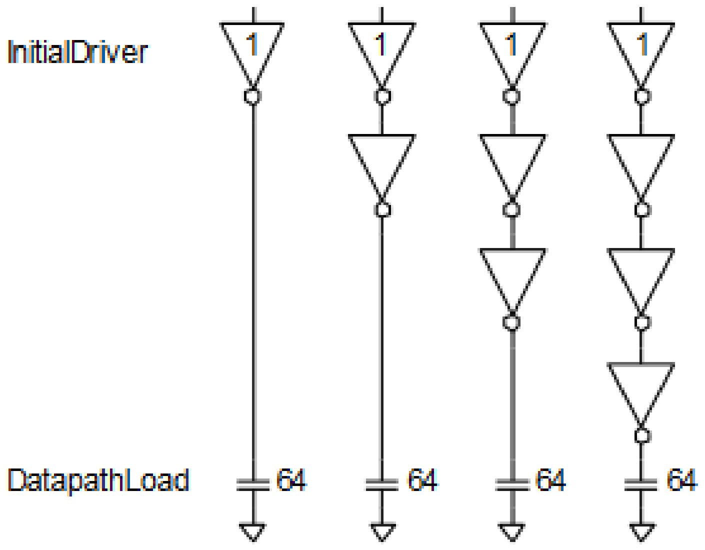
z= (Cout1*g1)/f' = (10+10+10)(4/3)/5 = 8





III. CHOOSING THE BEST NUMBER OF GATES

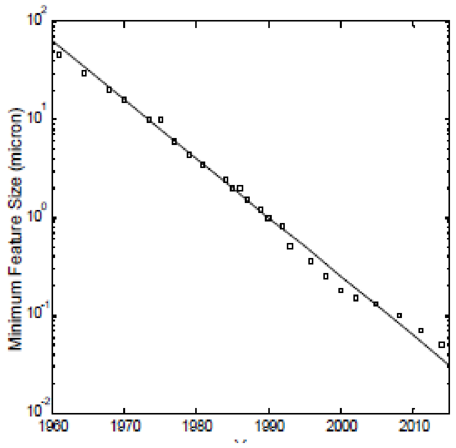
A control unit generates a signal from a unit sized invertor. The signal must drive unit sized loads in each bit slice of a 64 bit datapath. The designer can add inv to buffer the signal to drive the large load. What is the best no of inv to add and what delay can be achieved?



N=No of stages	1	2	3	4
F=gh	64	8	4	28
D=p+f	65	1+8+1+8=18	4+1+4+1+4+1=15	2.8+1+8/2.8+1+23/8+1+64/23+1=15.3
			Fastest	

Technology Scaling

Over the last decades, we have observed a spectacular increase in integration density and computational complexity of digital integrated circuits. Underlying this revolution are the advances in device manufacturing technology that allow for a steady reduction of the minimum feature size such as the minimum transistor channel length realizable on a chip. To illustrate this point, we have plotted inF igure the evolution of the (average) minimum device dimensions starting from the 1960s and projecting into the 21st century. We observe a reduction rate of approximately 13% per year, halving every 5 years. Another interesting observation is that no real sign of a slowdown is in sight, and that the breathtaking pace will continue in the foreseeable future.



This continued reduction in feature size influences the operating characteristics and properties of the MOS transistor, and indirectly the critical digital design metrics such as switching frequency and power dissipation. A first-order projection of this behaviour is called a *scaling analysis.* In addition to the minimum device dimension, we have to consider the supply voltage as a second independent variable in such a study. Different scaling scenarios result based on how these two independent variables are varied with respect to each other.
Three different models are studied in Table. To make the results tractable, it is assumed that all device dimensions scale by the same factor *S* (with *S* > 1 for a reduction in size). This includes the width and length of the transistor, the oxide thickness, and the junction depths. Similarly, we assume that all voltages, including the supply voltage and the threshold voltages, scale by a same ratio *U*. The relations governing the scaling behaviour of the dependent variables are tabulated in column. Observe that this analysis only considers short-channel devices with a linear dependence between control voltage and saturation current.

Parameter	Relation	Full Scaling	General Scaling	Fixed-Voltage Scaling
W, L, t_{ox}		$1/S$	$1/S$	$1/S$
V_{DD}, V_T		$1/S$	$1/U$	1
N_{SUB}	V/W_{depl}^2	S	S^2/U	S^2
Area/Device	WL	$1/S^2$	$1/S^2$	$1/S^2$
C_{ox}	$1/t_{ox}$	S	S	S
C_{gate}	$C_{ox}WL$	$1/S$	$1/S$	$1/S$
k_n, k_p	$C_{ox}W/L$	S	S	S
I_{sat}	$C_{ox}WV$	$1/S$	$1/U$	1
Current Density	$I_{sat}/Area$	S	S^2/U	S^2
Ron	V/I_{sat}	1	1	1
Intrinsic Delay	$R_{on}C_{gate}$	$1/S$	$1/S$	$1/S$
P	$I_{sat}V$	$1/S^2$	$1/U^2$	1
Power Density	$P/Area$	1	S^2/U^2	S^2

Full Scaling (Constant Electrical Field Scaling)
In this ideal model, voltages and dimensions are scaled by the same factor S . The goal is to keep the electrical field patterns in the scaled device identical to those in the original device. Keeping the electrical fields constant ensures the physical integrity of the device and avoids breakdown or other secondary effects. This scaling leads to greater device density (*Area*), higher performance (*Intrinsic Delay*), and reduced power consumption (*P*).
The effects of full scaling on the device and circuit parameters are summarized in the third column of Table . We use the intrinsic time constant, which is the product of the gate capacitance and the on-resistance, as a measure for the performance. The analysis shows that the on-resistance remains constant due to the simultaneous scaling of voltage swing and current level. The performance improved is solely due to the reduced capacitance. The results clearly demonstrate the beneficial effects of scaling— the speed of the circuit
increases in a linear fashion, while the power/gate scales down quadratically.

Fixed-Voltage Scaling
In reality, full scaling is not a feasible option. First of all, to keep new devices compatible with existing components, voltages cannot be scaled arbitrarily. Having to provide for multiple supply voltages adds considerably to the cost of a system. As a result, voltages have not been scaled down along with feature sizes, and designers adhere to well-defined standards

for supply voltages and signal levels. As is illustrated in Figure 3.40, 5 V was the de facto standard for all digital components up to the early 1990s, and a *fixed-voltage scaling model* was followed.

Only with the introduction of the 0.5 mm CMOS technology did new standards such as 3.3 V and 2.5 V make an inroad. Today, a closer tracking between voltage and device dimension can be observed. The reason for this change in operation model can partially be explained with the aid of the fixed-voltage scaling model, summarized in the fifth column of Table. In a velocity-saturated device, keeping the voltage constant while scaling the device dimensions does not give a performance advantage over the full-scaling model, but instead comes with a major power penalty. The gain of an increased current is simply offset by the higher voltage level, and only hurts the power dissipation. This scenario is very different from the situation that existed when transistors were operating in the long-channel mode, and the current was a quadratic function of the voltage. Keeping the voltage constant under these circumstances gives a distinct performance advantage, as it causes a net reduction in on-resistance.

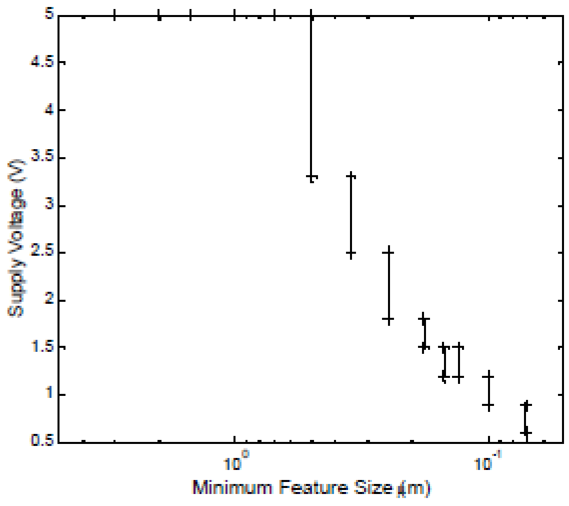
General Scaling

We observe in Figure that the supply voltages, while moving downwards, are not scaling as fast as the technology. For instance, for the technology scaling from 0.5m m to 0.1 mm, the maximum supply-voltage only reduces from 5 V to 1.5 V. The obvious question is why not to stick to the full-scaling model, when keeping the voltage higher does not yield any convincing benefits? This departure is motivated by the following argumentation:

• Some of the intrinsic device voltages such as the silicon band-gap and the built-in junction potential, are material parameters and cannot be scaled.

• The scaling potential of the transistor threshold voltage is limited. Making the threshold too low makes it difficult to turn off the device completely. This is aggravated by the large process variation of the value of the threshold, even on the same wafer.

Therefore, a more general scaling model is needed, where dimensions and voltages are scaled independently. This general scaling model is shown in the fourth column of Table. Here, device dimensions are scaled by a factor *S*, while voltages are reduced by a factor *U*. When the voltage is held constant, *U* = 1, and the scaling model reduces to the fixed-voltage model. Note that the general-scaling model offers a performance scenario identical to the full- and the fixed scaling, while its power dissipation lies between the two models (for *S* > *U* > 1).



Layout Design Rules, Gate Layouts, Stick Diagrams:

In VLSI design, as processes become more and more complex, need for the designer to understand the intricacies of the fabrication process and interpret the relations between the different photo masks is really troublesome.

Therefore, a set of layout rules, also called design rules, has been defined.

They act as an interface or communication link between the circuit designer and the process engineer during the manufacturing phase.

The objective associated with layout rules is to obtain a circuit with optimum yield (functional circuits versus non-functional circuits) in as small as area possible without compromising reliability of the circuit.

In addition, Design rules can be conservative or aggressive, depending on whether yield or performance is desired. Generally, they are a compromise between the two. Manufacturing processes have their inherent limitations in accuracy.

So the need of design rules arises due to manufacturing problems like –

• Photo resist shrinkage, tearing.

• Variations in material deposition, temperature and oxide thickness.

• Impurities.

• Variations across a wafer.

These lead to various problems like :

• **Transistor problems:**

Variations in threshold voltage: This may occur due to variations in oxide thickness, ion-implantation and poly layer. Changes in source/drain diffusion overlap. Variations in substrate.

• **Wiring problems:**

Diffusion: There is variation in doping which results in variations in resistance, capacitance.

Poly, metal: Variations in height, width resulting in variations in resistance, capacitance. Shorts and opens.

• **Oxide problems:**

Variations in height.

Lack of planarity.

• **Via problems:**

Via may not be cut all the way through.

Undersize via has too much resistance.

Via may be too large and create short.

To reduce these problems, the design rules specify to the designer certain geometric constraints on the layout artwork so that the patterns on the processed wafers will preserve the topology and geometry of the designs. This consists of minimum-width and minimum-spacing constraints and requirements between objects on the same or different layers. Apart from following a definite set of rules, design rules also come by experience.

Types of Design Rules

The design rules primary address two issues:

1. The geometrical reproduction of features that can be reproduced by the maskmaking and lithographical process, and

2. The interaction between different layers.

There are primarily two approaches in describing the design rules.

1. Linear scaling is possible only over a limited range of dimensions.

2. Scalable design rules are conservative .This results in over dimensioned and less dense design.

3. This rule is not used in real life.

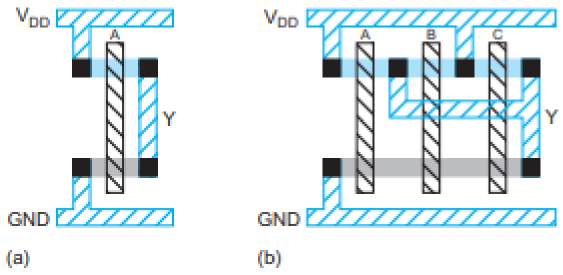
1. **Scalable Design Rules** (e.g. SCMOS, λ-based design rules): In this approach, all rules are defined in terms of a single parameter λ. The rules are so chosen that a design can be easily ported over a cross section of industrial process ,making the layout portable .Scaling can be easily done by simply changing the value of λ.
2. **Absolute Design Rules** (e.g. μ-based design rules) : In this approach, the design rules are expressed in absolute dimensions (e.g. 0.75μm) and therefore can exploit the features of a given process to a maximum degree. Here, scaling and porting is more demanding, and has to be performed either manually or using CAD tools .Also, these rules tend to be more complex especially for deep submicron. The fundamental unity in the definition of a set of design rules is the minimum line width .It stands for the minimum mask dimension that can be safely transferred to the semiconductor material .Even for the same minimum dimension, design rules tend to differ from company to company, and from process to process. Now, CAD tools allow designs to migrate between compatible processes.

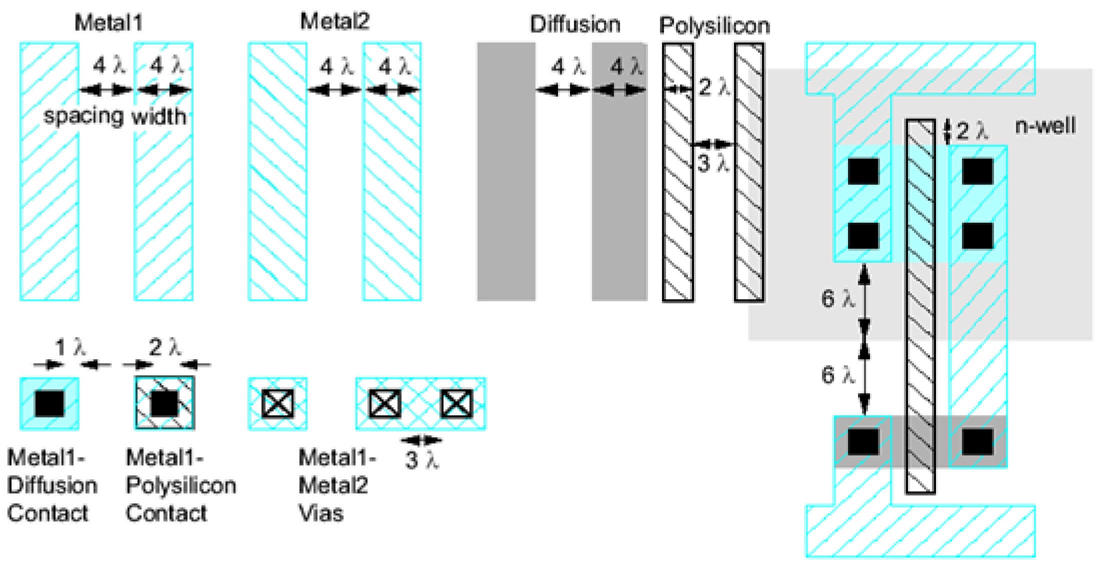
Layer Representations

With increase of complexity in the CMOS processes, the visualization of all the mask levels that are used in the actual fabrication process becomes inhibited. The layer concept translates these masks to a set of conceptual layout levels that are easier to visualize by the circuit designer. From the designer's viewpoint, all CMOS designs have the following entities:
• Two different substrates and/or wells: which are p-type for NMOS and n-type for PMOS.
• Diffusion regions (p+ and n+): which defines the area where transistors can be formed. These regions are also called active areas. Diffusion of an inverse type is needed to implement contacts to the well or to substrate. These are called select regions. • Transistor gate electrodes: Polysilicon layer
• Metal interconnect layers
• Interlayer contacts and via layers. The layers for typical CMOS processes are represented in various figures in terms of:
• A color scheme (Mead-Conway colors).
• Other color schemes designed to differentiate CMOS structures.
• Varying stipple patterns
• Varying line styles

Stick Diagrams

Another popular method of symbolic design is "Sticks" layout. In this, the designer draws a freehand sketch of a layout, using colored lines to represent the various process layers such as diffusion, metal and polysilicon .Where polysilicon crosses diffusion, transistors are created and where metal wires join diffusion or polysilicon, contacts are formed.
This notation indicates only the relative positioning of the various design components. The absolute coordinates of these elements are determined automatically by the editor using a compactor. The compactor translates the design rules into a set of constraints on the component positions, and solves a constrained optimization problem that attempts to minimize the area or cost function.
The advantage of this symbolic approach is that the designer does not have to worry about design rules, because the compactor ensures that the final layout is physically correct. The disadvantage of the symbolic approach is that the outcome of the compaction phase is often unpredictable. The resulting layout can be less dense than what is obtained with the manual approach. In addition, it does not show exact placement, transistor sizes, wire lengths, wire widths, tub boundaries.

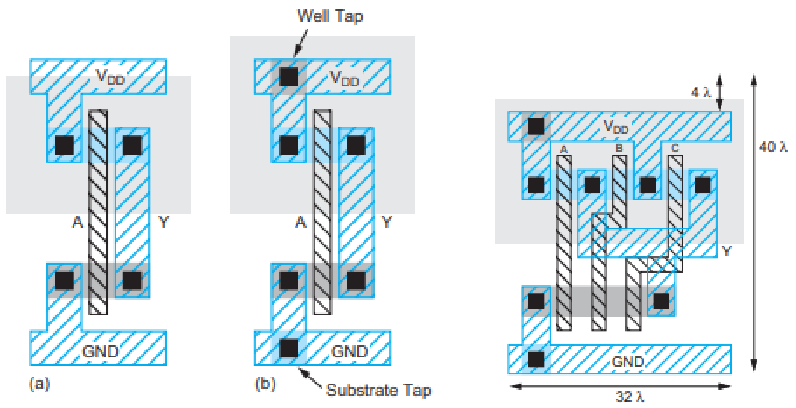




Gate Layouts

A good deal of ingenuity can be exercised and a vast amount of time wasted exploring layout topologies to minimize the size of a gate or other cell such as an adder or memory element. For many applications, a straightforward layout is good enough and can be automatically generated or rapidly built by hand. This section presents a simple layout style based on a "line of diffusion" rule that is commonly used for standard cells in automated layout systems. This style consists of four horizontal strips: metal ground at the bottom of the cell, n-diffusion, p-diffusion, and metal power at the top. The power and ground lines are often called supply rails. Polysilicon lines run vertically to form transistor gates. Metal wires within the cell connect the transistors appropriately.

Figure (a) shows such a layout for an inverter. The input A can be connected from the top, bottom, or left in polysilicon. The output Y is available at the right side of the cell in metal. Figure shows the same inverter with well and substrate taps placed under the power and ground rails, respectively



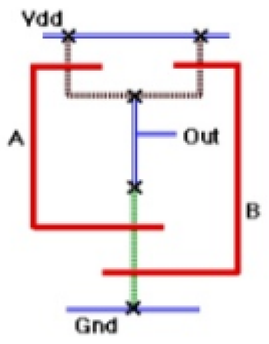
PART A

1. **Define threshold voltage of MOSFET A/M 19**
   The Threshold voltage, V for a MOS transistor can be defined as the voltage applied T between the gate and the source of the MOS transistor below which the drain to source current, I_{DS} effectively drops to zero.

2. **By what factor gate capacitance must be scaled if constant electric field scaling is employed? A/M 19**
   gate capacitance must be scaled by a factor 1/S

3. **Why NMOS device conducts strong zero and weak one? N/D 18**
   For an NMOS to pass VDD(logic 1) from input node to output node gate should be logic 1 . And the node out gets gradually charged from 0 towards VDD . When out reaches VDD-Vtn then (Vgate- Vout) = VDD-(VDD-Vtn) = Vtn which is the minimum voltage required for the NMOS to be ON state for a current to flow. So node out reaching to a potential more than VDD-Vtn turns off the NMOS. So the maximum voltage level that the output node can be charged to is VDD-Vtn.

4. **Draw the stick diagram of static CMOS 2 input NAND gate N/D 18**



5. **What is meant by velocity saturation? A/M 18**
   The saturation current increases less than quadratically with increasing Vgs .Tis is caused by two effects velocity saturation and mobility degradation.At high electric fields strengths Vds/L carrier velocity ceases to increase linearly with field strength.this is called velocity saturation and results in lower Ids ,than expected at high Vds.
6. **List the scaling properties? A/M 18**

Parameter	Relation	Full Scaling	General Scaling	Fixed-Voltage Scaling
W, L, t_{ox}		$1/S$	$1/S$	$1/S$
V_{DD}, V_T		$1/S$	$1/U$	1
N_{SUB}	V/W_{depl}^2	S	S^2/U	S^2
Area/Device	WL	$1/S^2$	$1/S^2$	$1/S^2$
C_{ox}	$1/t_{ox}$	S	S	S
C_{gate}	$C_{ox}WL$	$1/S$	$1/S$	$1/S$
k_n, k_p	$C_{ox}W/L$	S	S	S
I_{sat}	$C_{ox}WV$	$1/S$	$1/U$	1
Current Density	$I_{sat}/Area$	S	S^2/U	S^2
Ron	V/I_{sat}	1	1	1
Intrinsic Delay	$R_{on}C_{gate}$	$1/S$	$1/S$	$1/S$
P	$I_{sat}V$	$1/S^2$	$1/U^2$	1
Power Density	$P/Area$	1	S^2/U^2	S^2

7. **Why Nmos transistor is selected as pull down transistor? N/D 17**
The nmos I selected for pull down transistor because it passes strong 1's. which the used to turn off the transistor.

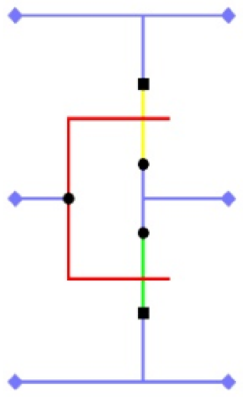
8. **What is the need of demarcation line? N/D 17**
In CMOS a demarcation line is drawn to avoid touching of p-diff with n-diff . all pmos must lie on one side of the line and all nmos will have o on other side.

9. **What is meant channel length modulation in NMOS transistors? A/M 17**
In cut off , the gate-to-source voltage is not greater than the threshold voltage, and the MOSFET is inactive .thus, channel-length modulation means that the saturation region drain current will increase slightly as the drain to source voltage is increases.

10. **Define propagation delay of a CMOS inverter? A/M 17**
Propagation delays Tplh and tphl are defined as the times required for output voltage to reach the middle between the low and high logic levels.

11. **Define body bias effect? N/D 16**
Body effect refers to the change in the threshold voltage of the device when there is a difference between substrate(body) and source voltages. Body bias is usually the lowest voltage in the chip(in case of of p-substrate).The substrate bias effect and source drain breakdown characteristics in body tied short channel silicon on insulator metal oxide semiconductor field effect transistor.

12. **Draw stick and layout diagram for cmos inverter? N/D 16**



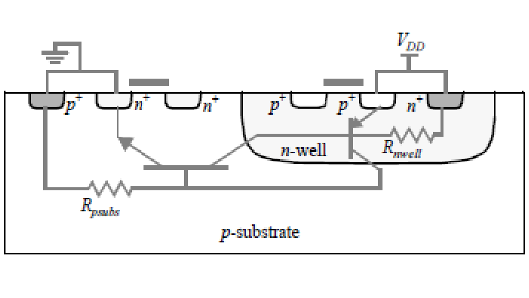
13. **State channel length modulation. Write down the equation for describing the channel length modulation effect in NMOS transistor. A/M 16**
The effective length of the conductive channel is actually modulated by the applied V_{DS}: increasing V_{DS} causes the depletion region at the drain junction to grow, reducing the length of the effective channel. As can be observed from Eq. For I_D in the saturation region, the current increases when the length factor L is decreased. A more accurate description of the current of the MOS transistor is therefore given in Eq.

$$I_D = I_D'(1 + \lambda V_{DS})$$

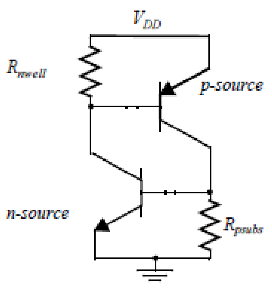
with I_D' the current expressions derived earlier, and l an empirical parameter, called the *channel-length modulation*.

14. What is latch-up? How to prevent latch-up? A/M 16

Consider the *n*-well structure of Figure a. The *n-p-n-p* structure is formed by the source of the NMOS, the *p*-substrate, the *n*-well and the source of the PMOS. A circuit equivalent is shown in Figure b. When one of the two bipolar transistors gets forward biased (e.g., due to current flowing through the well, or substrate), it feeds the base of the other transistor. This positive feedback increases the current until the circuit fails or burns out.



(a) Origin of latchup



(b) Equivalent circuit

PART B

1. Derive an expression for Ids of nMOS in linear and saturated region? **A/M 19** **(OR)** Describe the equation for source to drain current in the three region of operation of MOS transistor and draw the VI characteristics. **A/M 16**
2. Draw a CMOS inverter. Analyse the switching characteristics during the rise time when Vin changes from high to low? **A/M 19 &** derive the noise margin for CMOS inverter? **N/D 16**
3. Draw the stick diagram of CMOS inverter? **A/M 19(OR)** Write the layout design rules and draw the stick diagram and layout diagram for 4 input NAND gate and 4 input NOR. **A/M 18 (OR)** Write the layout rules and draw diagram for four input NAND and NOR gate? **N/D 17 (OR)** Draw the layout diagram for NAND and NOR gate? **A/M 17 (OR)** State the minimum width and minimum spacing lambda based design rules to draw the layout **A/M 19**
4. Explain the dynamic behaviour of MOS transistors with neat diagram **A/M 18**
5. (a)(i) Explain the electrical properties of CMOS? **N/D 17**
   (a) (ii) Discuss on scaling and its limits? **N/D 17 (OR)** Explain the need of scaling , scaling principles and fundamentals units of CMOS inverter? **A/M 17**
6. Draw and explain the dc and transfer characteristics of CMOS inverter with necessary condition for the different regions of operation? **A/M 17 (OR)** Explain the DC transfer characteristics of a CMOS inverter with necessary conditions for the different regions of operation. **A/M 16**
7. Explain the different steps involved in n-well Cmos fabrication process? **N/D 16**
8. Explain in detail about the body effect and its effect in MOS device **A/M 16**

Unit II
Combinational Logic Circuits

Introduction

In combinational circuits, the output is a function of current inputs. Eg: adder, subtractor, MUX and DEMUX. In general, the delay of a logic gate depends on its output current I, load capacitance C and output voltage ΔV. The time delay is directly proportional to C / I.

If any one of these numerator parameters is reduced or I increases then time delay t decreases.

Generally the nMOS transistor provides more current than pMOS for the same size and capacitance so nMOS network are preferred. The logical effort g is directly proportional to C/I. If g decreases speed will be increasing.

There are number of logic families to implement a given logic function. It can be choosing/ evaluation in terms of area, speed energy and power. The different logic families are

- Static CMOS design.
- Dynamic CMOS design.

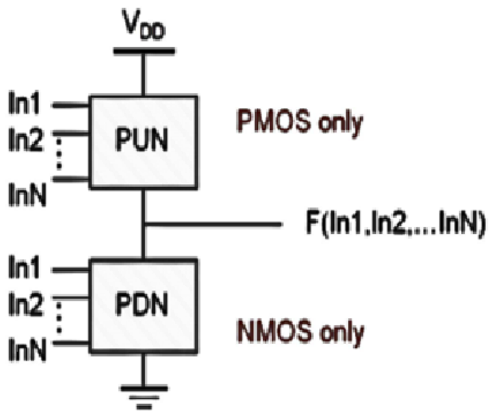
I. Static CMOS design

At each and every points in time the gate output is either connected to Vdd or Vss through a low resistance path.

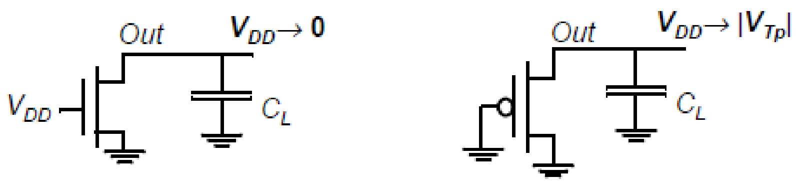
1) Complementary CMOS

It is a combination of pMOS [pull up network] and nMOS [pull down network].

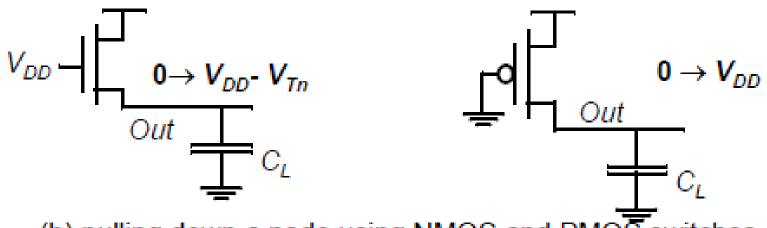
Diagram



nMOS produces “strong zero” and pMOS produces “ strong one”

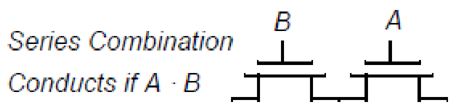


(a) pulling down a node using NMOS and PMOS switches

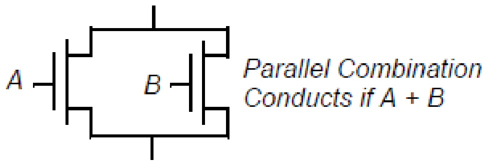


(b) pulling down a node using NMOS and PMOS switches

If nMOS devices are connected in series then the function is “AND”.If the nMOS devices are connected in parallel that represents an OR

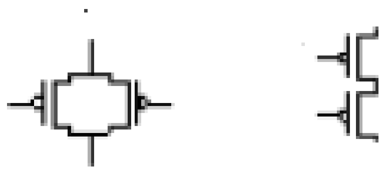


(a) series

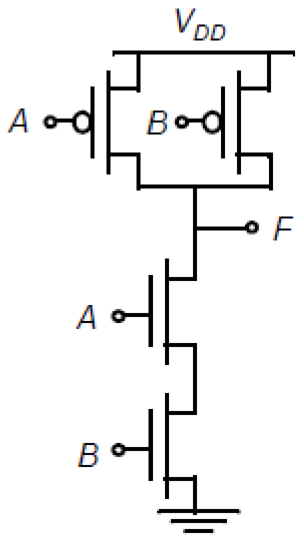


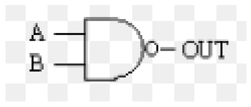
(b) parallel

For pMOS series means OR and AND means parallel. The parallel connection of transistor in Pun corresponds to a series connection in PDN. Hence they are called dual networks.



The CMOS is naturally inverting so realization of AND Or and Xor require extra inverter stage. The number of transistors required is 2N.

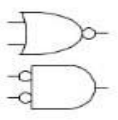


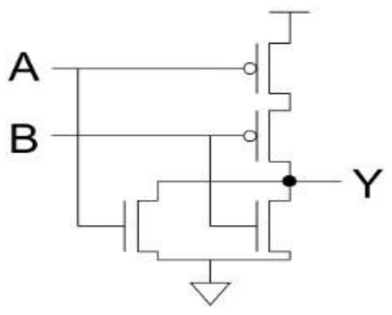


A	B	F
0	0	1
0	1	1
1	0	1
1	1	0

CMOS NOR Gate

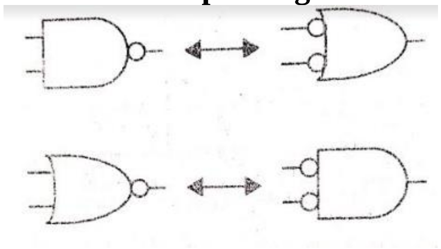
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0





Advantages
Good noise margin.
Fast.
Low power.
Rendily available in standard cell libraries.
Insensitive to device variation.
Easy to design.
Widely supported by CAD tools.
Disadvantages
For each input it requires both nMOS and pMOS.
It has relatively large logical effort i.e during falling output transition the pMOS transistors add significant capacitance. Eventhough it is not responsible for falling output transistion (1 to 0

i. **Bubble pushing:**

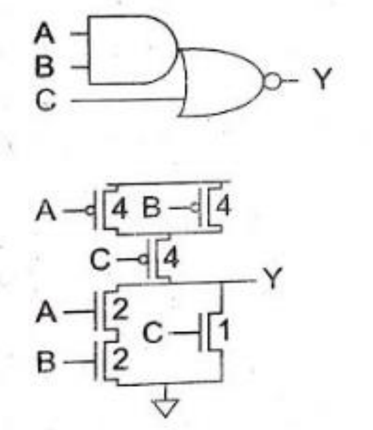


Bubble pushing is a technique to apply De Morgan's theorem directly to the logic diagram. To change the logic gate (AND to OR and OR to AND), Add **bubbles** to the inputs and outputs where there were none, and remove the original **bubbles**.

ii. **Compound Gates:**

AOI21

$Y = \overline{A \cdot B + C}$



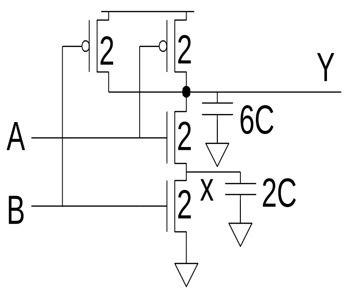
$g_A = 6/3$

$g_B = 6/3$

$g_C = 5/3$

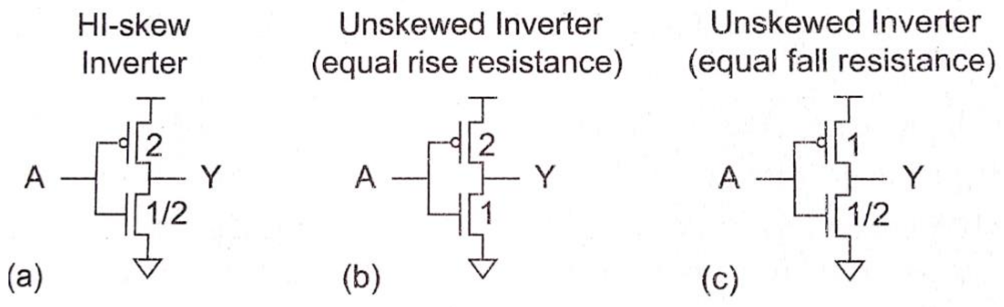
$p = 7/3$

iii. Input Ordering Delay Effect:



– If input arrival time is known,Connect latest input to inner terminal

iv. Skewed Gates:


HI-skew Inverter
Unskewed Inverter (equal rise resistance)
Unskewed Inverter (equal fall resistance)
A
2
Y
1/2
(a)
A
2
Y
1
(b)
A
1
Y
1/2
(c)

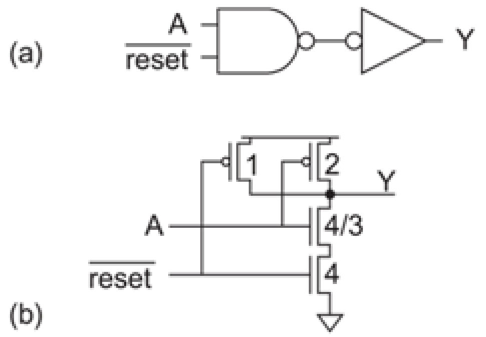
- Skewed gates favor one transition over another
- Ex: suppose rising output of inverter is most critical

– Downsize noncritical nMOS transistor

– Calculate logical effort by comparing to unskewed inverter with same effective resistance on that edge.

– g_u = 2.5 / 3 = 5/6
g_d = 2.5 / 1.5 = 5/3

v. Asymmetric Gates



Eg: Resettable Buffer optimized for data input

- Asymmetric gates favor one input over another
- Ex: suppose input A of a NAND gate is most critical

– Use smaller transistor on A (less capacitance)

– Boost size of noncritical input

– So total resistance is same

- g_A = 10/9:g_B = 2
- g_{avg} = (g_A + g_B)/2 = 14/9
- Asymmetric gate approaches g = 1 on critical input
- But total logical effort goes up

2) . Ratioed circuits:

Ratioed circuit use weak pull up device and strong pull down device.
The transfer function depends on the ratio between the strength of PUN and PDN.

Advantages:

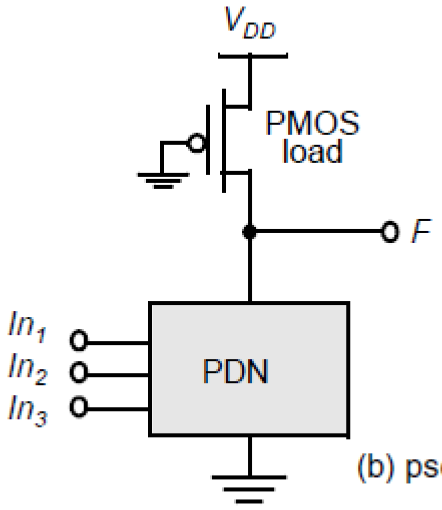
It reduces the input capacitance so that the logical effort is improved (by eliminating large pMOS transistors which load the supply).
Reduced area.

Disadvantages:

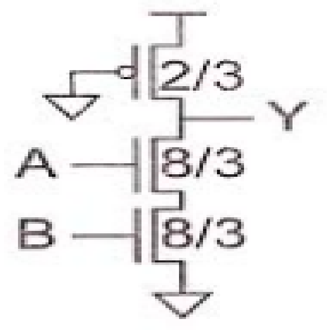
Continuously dissipates power when input=1.
poor noise margin than CMOS.

i)Pseudo nMOS:

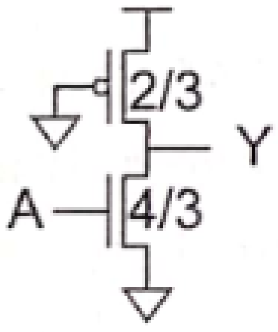
The most common form of CMOS ratioed logic.
The PDN of ratioed circuit is similar to PDN of static CMOS. But the pull up network is replaced by a single pMOS transistor where input is grounded, so is always on



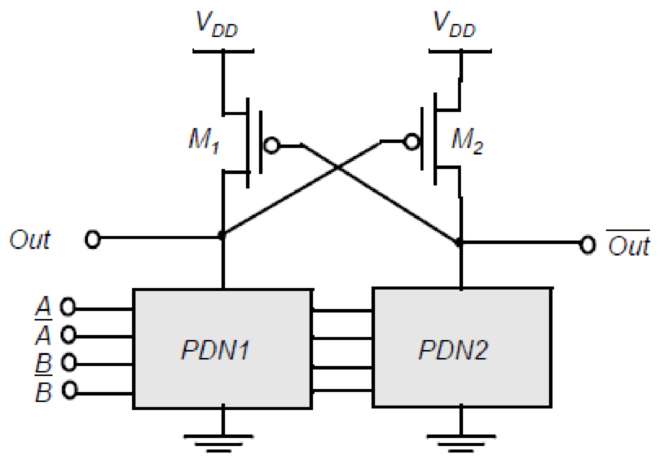
(b) pseudo-NMOS



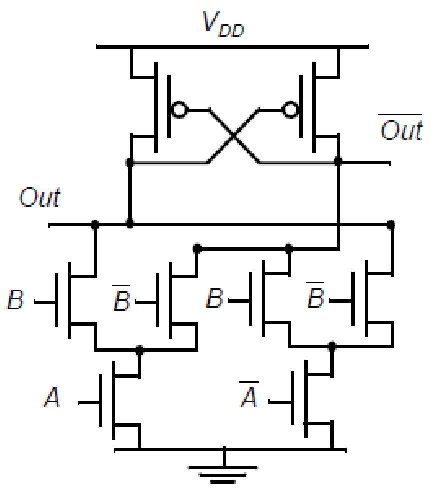
If the PUN is strong ,it produces the high V_{OL} (noise margin -good).
If the pull up network is weak it produces high rising delay(speed).
pMOS transistor width is optimised.
So the width of the transistor is selected by taking 1/4 strength(or 1/2 effective width of the nMOS transistor).
EX: Invertor using pseudo nMOS:



ii) Cascode Voltage Switch Logic(CVSL):

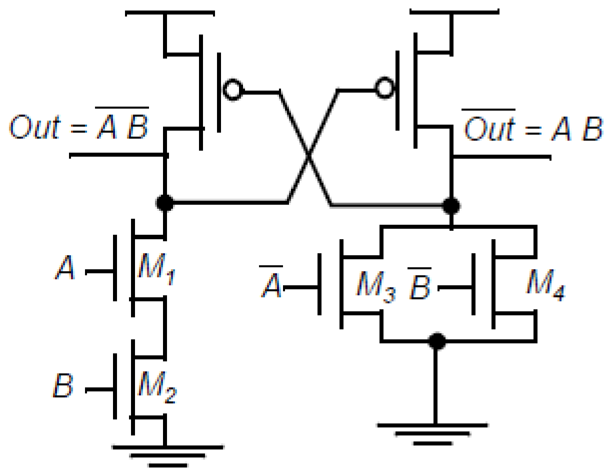


(a) Basic principle



(b) XOR-XNOR gate

- The static power dissipation problem in the ratioed circuits can be improved by using CVSL.
- It uses both true and complementary input signals and computes both true and complementary output using a pair n-MOS PDN.
- The PDN f implements the logic function similar to static CMOS gates.
- The PDN f' uses inverted inputs. The networks f and f' are complementary with parallel transistors in one network and series in another network.



-
- When A=B=1, both n-MOS are **ON** which pulls the output **y=0**. This turns on the p-MOS transistor (p1) which produces the **y'=1**.
- When A=0 B=1, n3 turns ON so it produces the output y'=0 which turns on the p2 which produces y=1.
- When A=B=0, both n3 and n4 turn ON, this will pull the output y'=0 which will turn on the p2 transistor so the y=1.
- For any given input one of the PDN will be ON and the other will be OFF, so there is no static power dissipation.
- The advantage of CVSL is speed, because all of the logic is performed with n-MOS transistor then reducing input capacitance.
- The disadvantage of CVSL is it requires both the low and high going transitions, which increases the delay. CVSL is not suited to NAND and NOR logic.

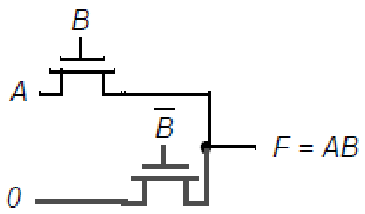
3) ***. Pass Transistor Logic***

A popular and widely-used alternative to complementary CMOS is pass-transistor logic, which attempts to reduce the number of transistors required to implement logic by allowing the primary inputs to drive gate terminals as well as source/drain terminals. This is in contrast to logic families that we have studied so far, which only allow primary inputs to drive the gate terminals of MOSFETS.

Figure shows an implementation of the AND function constructed that way, using only NMOS transistors. In this gate, if the *B* input is high, the top transistor is turned on and copies the input *A* to the output *F*. When *B* is low, the bottom pass transistor is turned on and passes a 0. The switch driven by *B* seems to be redundant at first glance. Its presence is essential to ensure that the gate is static, this is that a low-impedance path exists to the supply rails under all circumstances, or, in this particular case, when *B* is low.

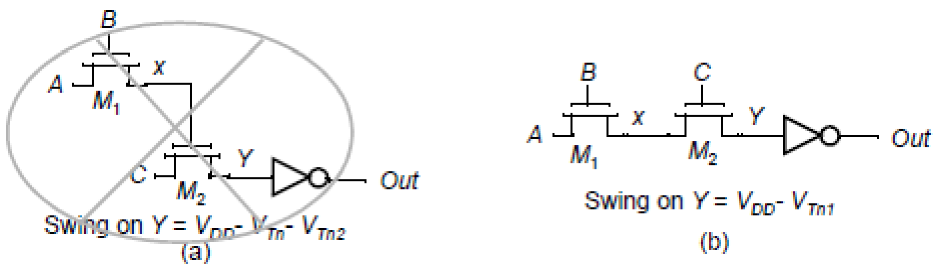
The promise of this approach is that fewer transistors are required to implement a given function. For example, the implementation of the AND gate in Figure requires 4 transistors (including the inverter required to invert *B*), while a complementary CMOS implementation would require 6 transistors. The reduced number of devices has the additional advantage of lower capacitance.

- nMOS pass strong 0 – But degraded or weak 1
- pMOS pass strong 1 – But degraded or weak 0



An NMOS device is effective at passing a 0 but is poor at pulling a node to *VDD*. When the pass transistor pulls a node high, the output only charges up to *VDD -VTn*. In fact, the situation is worsened by the fact that the devices
experience body effect, as there exists a significant source-to-body voltage when pulling high. Consider the case when the pass transistor is charging up a node with the gate and drain terminals set at *VDD*. Let the source of the NMOS pass transistor be labeled *x*. Node *x* will charge up to *VDD-VTn*(*Vx*):

$$V_x = V_{DD} - (V_{tn0} + \gamma((\sqrt{|2\phi_f| + V_x}) - \sqrt{|2\phi_f|}))$$


B
A
x
M_1
C
Y
M_2
Out
Swing on Y = V_{DD}- V_{Tn}- V_{Tn2}
(a)
B
C
A
x
Y
M_1
M_2
Out
Swing on Y = V_{DD}- V_{Tn1}
(b)

Pass-transistor gates cannot be cascaded by connecting the output of a pass gate to the gate input of another pass transistor. This is illustrated in Figure, where the output of *M*1 (node *x*) drives the gate of another MOS device. Node *x* can charge up to *VDD-VTn1*. If node *C* has a rail to rail swing, node *Y* only charges up to the voltage on node *x - VTn2*, which works out to *VDD-VTn1-VTn2*.

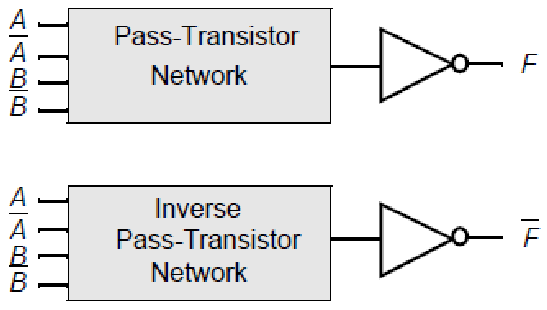
4) Differential Pass Transistor Logic/CPL

For high performance design, a differential pass-transistor logic family, called CPL or DPL, is commonly used. The basic idea (similar to DCVSL) is to accept true and complementary inputs and produce true and complementary outputs. A number of CPL gates (AND/NAND, OR/NOR, and XOR/NXOR) are shown in Figure. These gates possess a number of interesting properties:

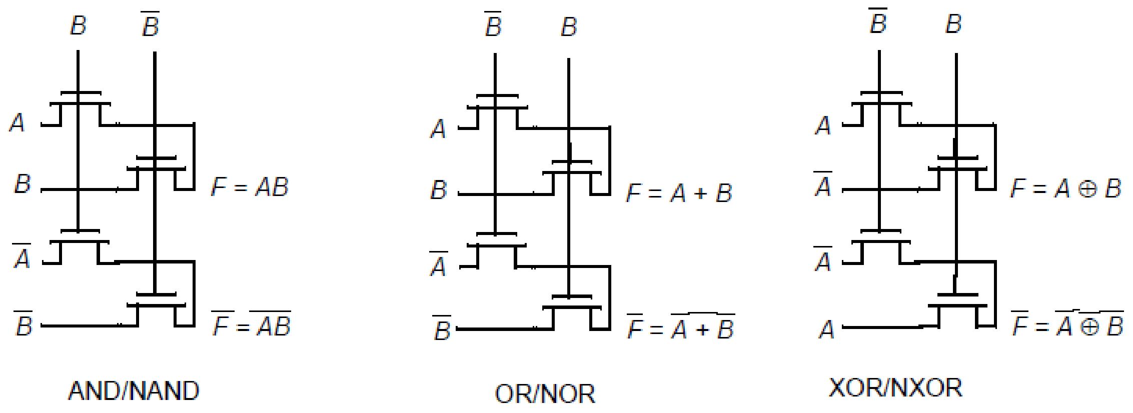
• Since the circuits are *differential*, complementary data inputs and outputs are always available. Although generating the differential signals requires extra circuitry, the differential style has the advantage that some complex gates such as XORs and adders can be realized efficiently with a small number of transistors. Furthermore, the availability of both polarities of every signal eliminates the need for extra inverters, as is often the case in static CMOS or pseudo-NMOS.

• CPL belongs to the class of *static* gates, because the output-defining nodes are always connected to either *VDD* or *GND* through a low resistance path. This is advantageous for the noise resilience.

• The design is very modular. In effect, all gates use exactly the same topology. Only the inputs are permutated. This makes the design of a library of gates very simple. More complex gates can be built by cascading the standard pass-transistor modules.



(a) Basic concept



(b) Example pass-transistor networks

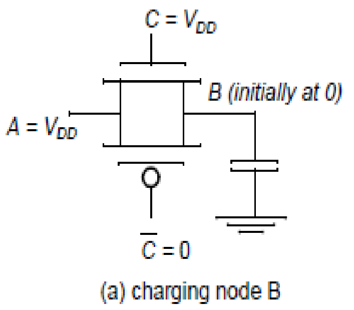
Unfortunately, differential pass-transistor logic, like single-ended pass-transistor logic, suffers from static power dissipation and reduced noise margins, since the high input to the signal-restoring inverter only charges up to *VDD-VTn*.

5) Transmission Gates

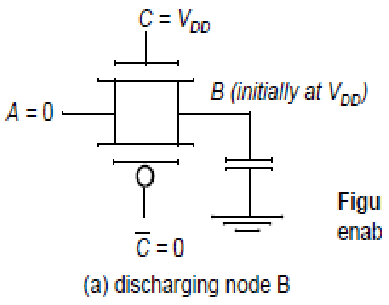
The most widely-used solution to deal with the voltage-drop problem is the use of *transmission gates*. It builds on the complementary properties of NMOS and PMOS transistors: NMOS devices pass a strong 0 but a weak 1, while PMOS transistors pass a strong 1 but a weak 0. The ideal approach is to use an NMOS to pull-down and a PMOS to pull-up. The transmission gate combines the best of both device flavors by placing a NMOS device in parallel with a PMOS device . The control signals to the transmission gate (*C* and *C*) are complementary. The transmission gate acts as a bidirectional switch controlled by the gate signal *C*. When *C* = 1, both MOSFETs are on, allowing the signal to pass through the gate. In short,

$$A = B \quad \text{if} \quad C = 1$$

On the other hand, *C* = 0 places both transistors in cutoff, creating an open circuit between nodes *A* and *B*. Figure 6.44b shows a commonly used transmission-gate symbol. Consider the case of charging node *B* to *VDD* for the transmission gate circuit in Figure . Node *A* is driven to *VDD* and transmission gate is enabled (*C* = 1 and *C*= 0). If only the NMOS pass-device were present, node *B* charges up to *VDD-VTn* at which point the NMOS device turns off. However, since the PMOS device is present and turned on (*VGSp* = *-VDD*), charging continues all the way up to *VDD*. Figure 6.45b shows the opposite case, this is discharging node *B* to 0. *B* is initially at *VDD* when node *A* is driven low. The PMOS transistor by itself can only pull down node *B* to *VTp* at which point it turns *off*. Theparallel NMOS device however stays turned on (since its *VGS* = *VDD*) and pulls node *B* all the way to *GND*. Though the transmission gate requires two transistors and more control signals, it enables rail-to-rail swing.



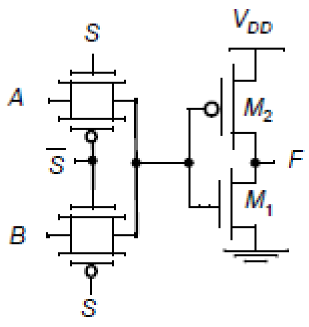
(a) charging node B



(a) discharging node B

Transmission gates can be used to build some complex gates very efficiently. Figure shows an example of a simple inverting two-input multiplexer. This gate either selects input *A* or *B* based on the value of the control signal *S*, which is equivalent to implementing the following Boolean function:

$$\overline{F} = (A \cdot S + B \cdot \overline{S})$$

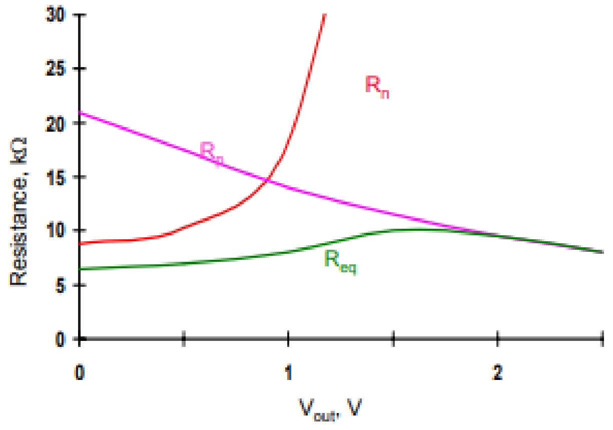


The pass-transistor and the transmission gate are, unfortunately, not ideal switches, and have a series resistance associated with it. To quantify the resistance, consider the circuit in Figure, which involves charging a node from 0 V to *VDD*. In this discussion, we use the large-signal definition of resistance, which involves dividing the voltage across the switch by the drain current. The effective resistance of the switch is modelled as a parallel connection of the resistances *Rn* and *Rp* of the NMOS and PMOS devices, defined as (*VDD* – *Vout*)/*In* and (*VDD* – *Vout*)/*Ip*, respectively. The currents through the devices are obviously dependent on the value of *Vout* and the operating mode of the transistors. During the low to- high transition, the pass-transistors traverse through a number of operation modes. For low values of *Vout*, the NMOS device is saturated and the resistance is approximated as:

$$R_n = \frac{V_{DD}-V_{out}}{I_N} = \frac{V_{DD}-V_{out}}{k'_n\left(\frac{W}{L}\right)_N\left((V_{DD}-V_{out}-V_{Tn})V_{DSAT}-\frac{V_{DSAT}^2}{2}\right)}$$
$$\approx \frac{V_{DD}-V_{out}}{k_n(V_{DD}-V_{out}-V_{Tn})V_{DSAT}}$$

The resistance goes up for increasing values of *Vout*, and approaches infinity when *Vout* reaches *VDD-VTn*, this is when the device shuts off. Similarly, we can analyze the behaviour of the PMOS transistor. When *Vout* is small, the PMOS is saturated, but it enters the linear mode of operation for *Vout* approaching *VDD*, giving the following approximated resistance:

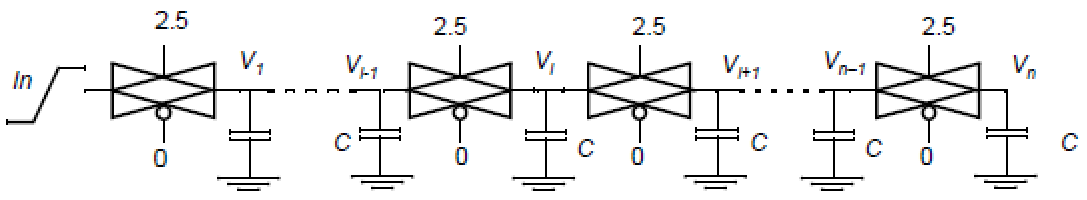
$$R_p = \frac{V_{DD}-V_{out}}{I_P} = \frac{V_{DD}-V_{out}}{k_p\cdot\left((-V_{DD}-V_{Tp})(V_{out}-V_{DD})-\frac{(V_{out}-V_{DD})^2}{2}\right)}$$
$$\approx \frac{1}{k_p(V_{DD}-|V_{Tp}|)}$$



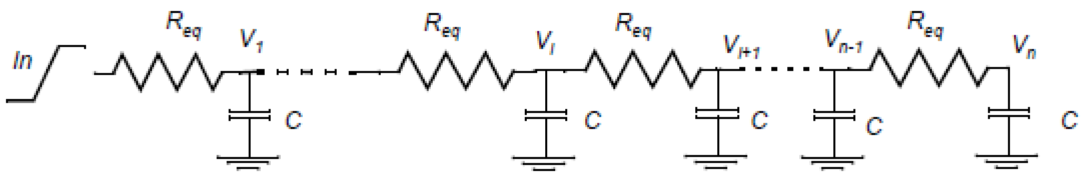
The delay of a network of *n* transmission gates in sequence can be estimated using the Elmore approximation
XOR design:

$$t_p(V_n) = 0.69\sum_{k=0}^{n} CR_{eq}k = 0.69CR_{eq}\frac{n(n+1)}{2}$$

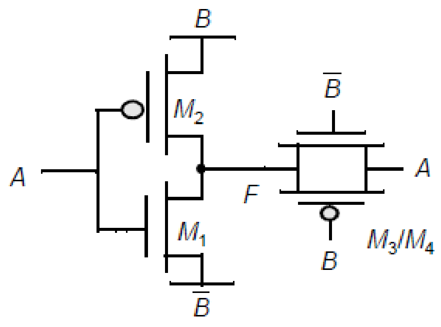
This means that the propagation delay is proportional to *n2* and increases rapidly with the number of switches in the chain.



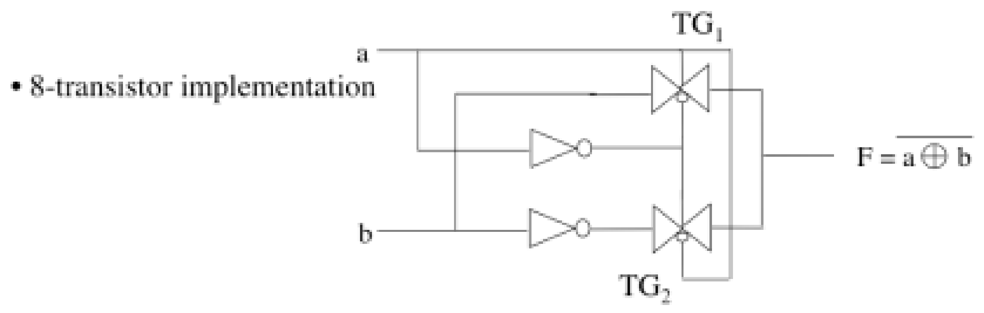
(a) A chain of transmission gates

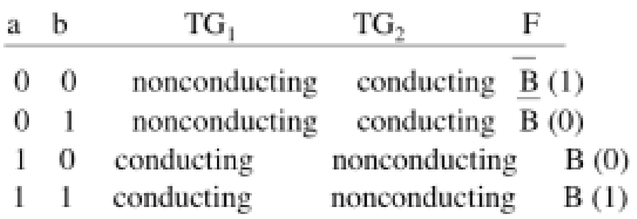


(b) Equivalent *RC* network



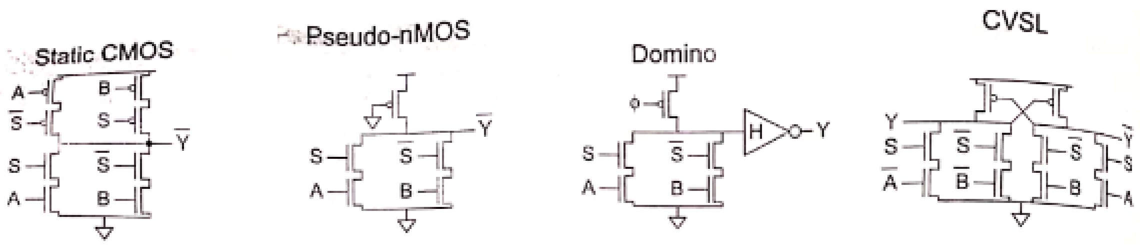
XNOR design:





a	b	TG_1	TG_2	F
0	0	nonconducting	conducting	$\overline{B}$ (1)
0	1	nonconducting	conducting	B (0)
1	0	conducting	nonconducting	B (0)
1	1	conducting	nonconducting	B (1)

Multiplexer Design using various static cmos:



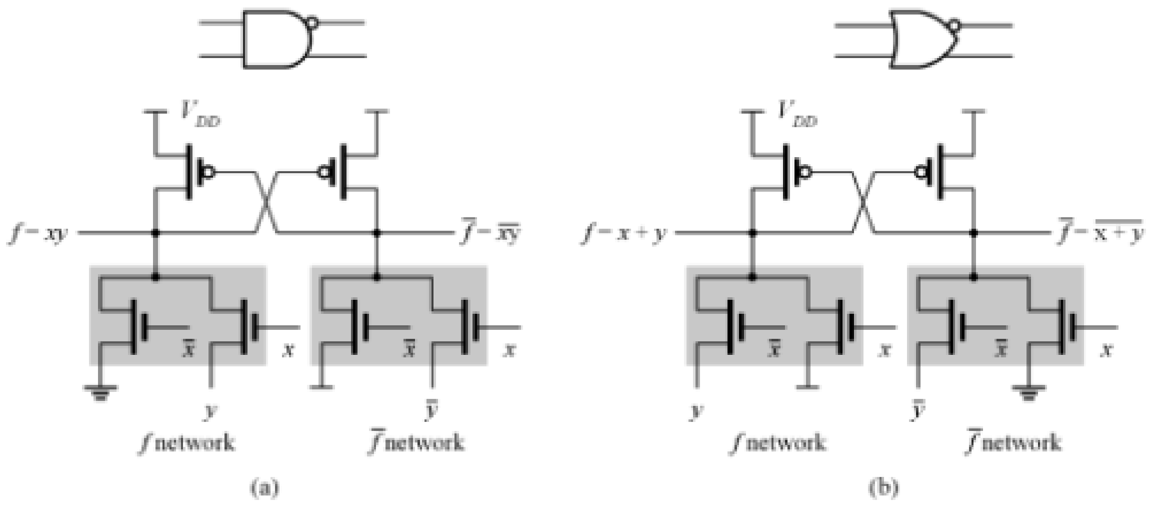
6) Differential Cascode Voltage Switch with Pass transistor

Differential Cascode Voltage Switch with Pass gate is a combination of CVSL and CPL. It uses the output stage of CVSL as its output stage and uses the logic networks of CPL to implement the desired logic function. DCVSPG can be designed using 0/1-x/x tree network paradigm.

For a NAND/AND gate

$$\bar{f} = \overline{x \cdot y} = \bar{x} + \bar{y} = \bar{x} \cdot 1 + x \cdot \bar{y}$$

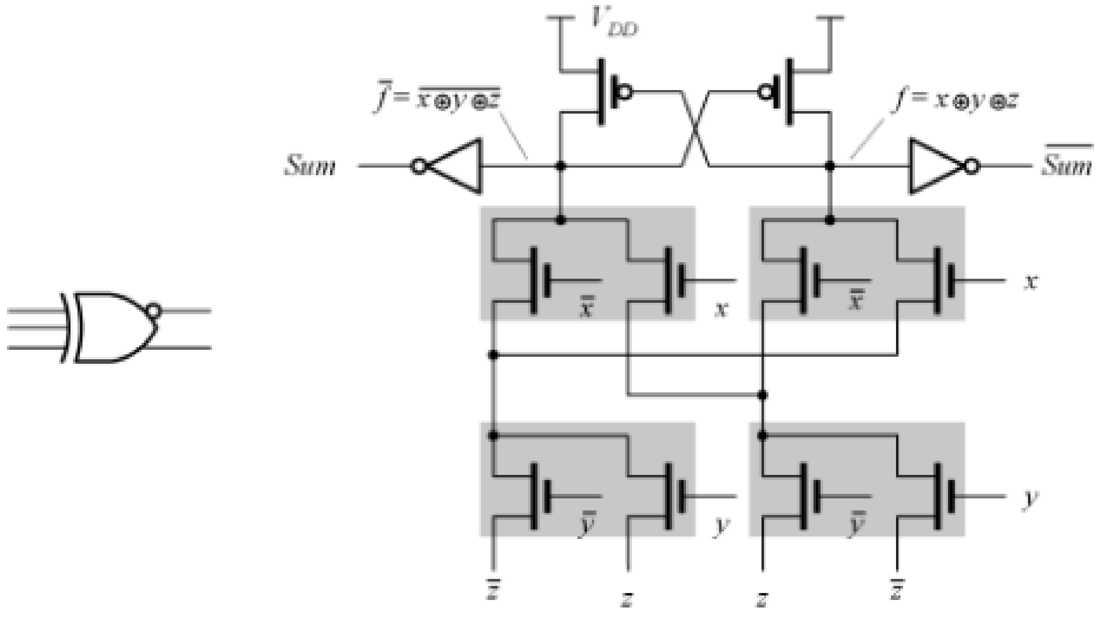
$$f = x \cdot y = \bar{x} \cdot 0 + x \cdot y$$



For XOR/XNOR

$$\begin{aligned} f &= x \oplus y \oplus z = x \cdot \bar{y} \cdot \bar{z} + \bar{x} \cdot y \cdot \bar{z} + \bar{x} \cdot \bar{y} \cdot z + x \cdot y \cdot z \\ &= \bar{x} \cdot (\bar{y} \cdot z + y \cdot \bar{z}) + x \cdot (\bar{y} \cdot \bar{z} + y \cdot z) \end{aligned}$$

$$\begin{aligned} \bar{f} &= \overline{x \oplus y \oplus z} = \bar{x} \cdot \bar{y} \cdot \bar{z} + \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot \bar{z} \\ &= \bar{x} \cdot (\bar{y} \cdot \bar{z} + y \cdot z) + x \cdot (\bar{y} \cdot z + y \cdot \bar{z}) \end{aligned}$$



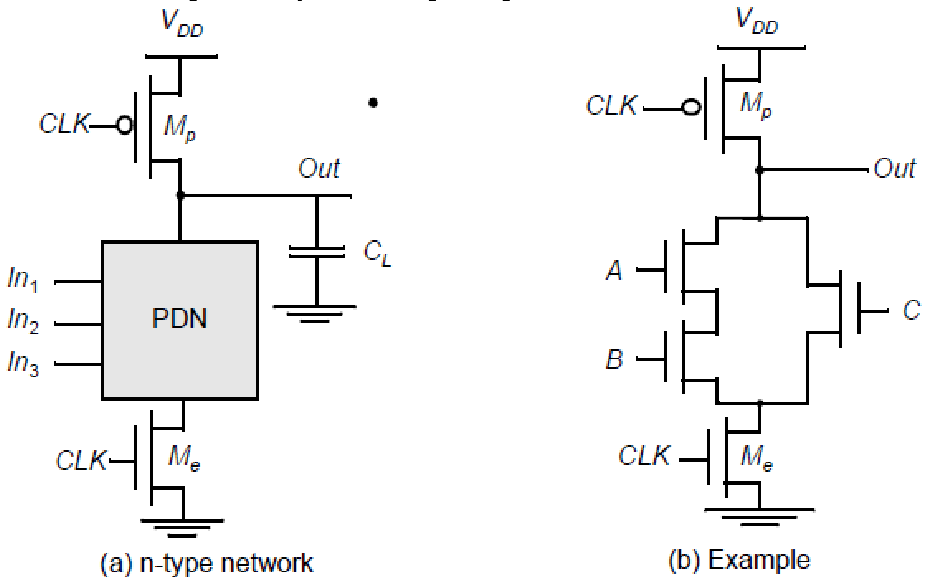
V_{DD}
$\bar{f} = \overline{x \oplus y \oplus z}$
$f = x \oplus y \oplus z$
Sum
$\overline{Sum}$
$\bar{x}$
x
$\bar{y}$
y
$\bar{z}$
z

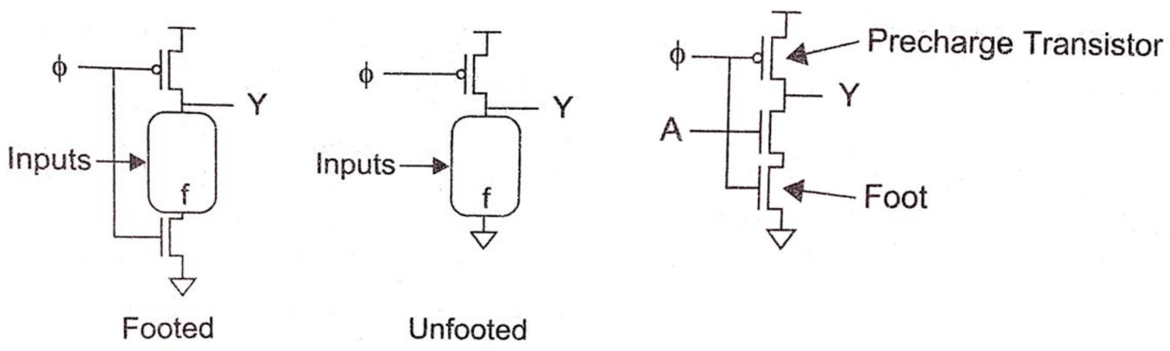
II. DYNAMIC CIRCUITS:

The drawbacks of ratioed circuits are

- Slow rising transitions
- Contention on the falling transition
- Static power dissipation
- Non-zero Vol

➢ These drawbacks can be eliminated by using dynamic circuits in which the always on p-MOS transistors are replaced by clocked pull up transistors.



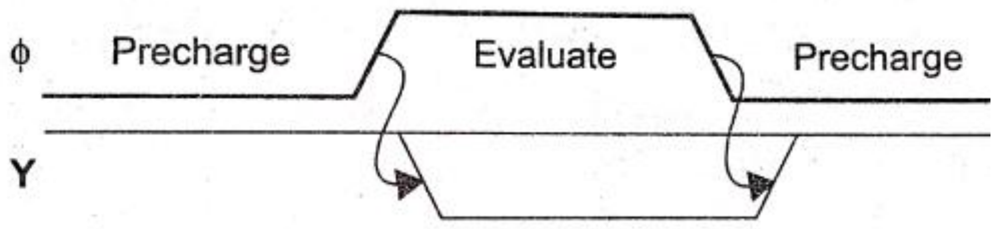


➢ The dynamic circuit operation is divided into two modes.

1. Precharge
2. Evaluation

➢ During precharge clock φ=0 so p-MOS on. Therefore y=1.

➢ During evaluation clock φ=1 so p-MOS off. Output will be 0/1 depending on the input by discharging through the PDN.

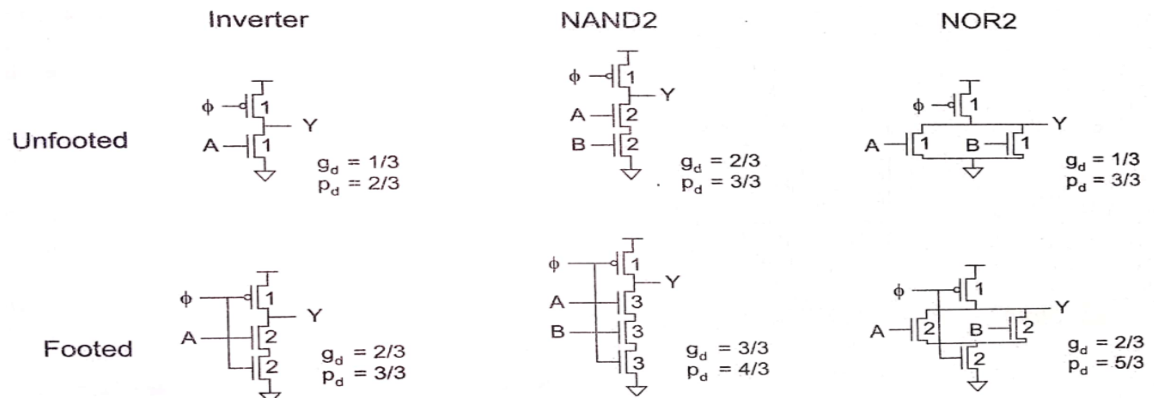


ADVANTAGES:

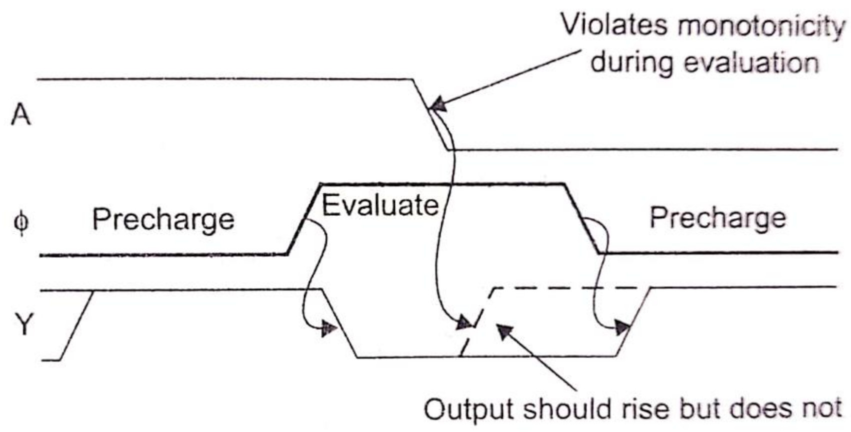
- Dynamic circuits are the fastest. Therefore Cin is less.
- It is non-ratioed.
- No contention during switching (Both p-MOS and n-MOS are not on simultaneously).
- They also have zero static power dissipation.
- Reduced implementation area.

DISADVANTAGES:

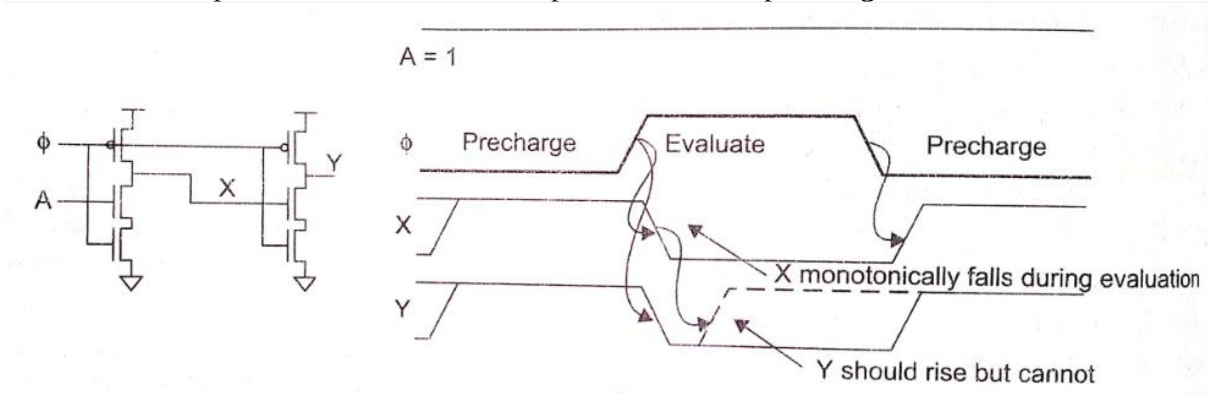
- The number of transistors (for complex gates) is substantially lower than in the static case: N + 2 versus 2N.
- It requires careful clocking.
- Power dissipation due to clock.
- Sensitive to noise during evaluation.
- In dynamic inverter, if A=1 during prechange mode, contention will take place because both the pMOS and nMOS will be ON. Because the input A cannot be generated to be during precharge.
- To avoid this contention an extra clocked evaluation transistors can be added to the bottom of the nMOS stack. The extra transistor is called as foot.
- Logical effort, parasitic delay for falling transition of footed and unfooted gates are given as follows.



- The logical effort of footed gates are larger than unfooted gates, but there is an improvement when compared to static CMOS logic.
- The precharge transistor width is chosen for twice unit resistance 2R/k=2R/1=2R, inorder to reduce the parasitic capacitance which loads the clock.
  i. MONOTONICITY:A fundamental difficulty in dynamic circuit is the monotonicity requirements. That means, during the evaluation period, the input should be monotonically rising (i.e. the input should not start high and fall low).

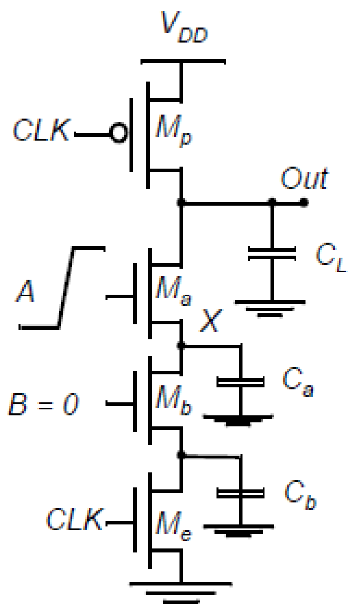

Violates monotonicity during evaluation
A
ϕ
Precharge
Evaluate
Precharge
Y
Output should rise but does not

- During precharge φ=0. So the output is pulled high.
- During evaluation phase φ=1, the input A=1 so the output is discharged low through PDN.
- After sometime , the input falls low, which will turn off the PDN . In this time , the output should be high.
  Therefore, the input is zero. Instead the output=0 until next precharge.



ii. Charge Sharing

Another important concern in dynamic logic is the impact of charge sharing. Consider the circuit of Figure. During the precharge phase, the output node is precharged to VDD. Assume that all inputs are set to 0 during precharge, and that the capacitance Ca is discharged. Assume further that input B remains at 0 during evaluation, while input A makes a 0 ® 1 transition, turning transistor Ma on. The charge stored originally on capacitor CL is redistributed over CL and Ca. This causes a drop in the output voltage, which cannot be recovered due to the dynamic nature of the circuit.



The different kinds of dynamic circuits are,

- Domino logic
- Dual rail domino logic
- Multiple output domino logic
- NP and zipper Domino

III. DOMINO LOGIC

The monotonicity problem can be overcome by placing a static CNOs inverter between the dynamic gates.

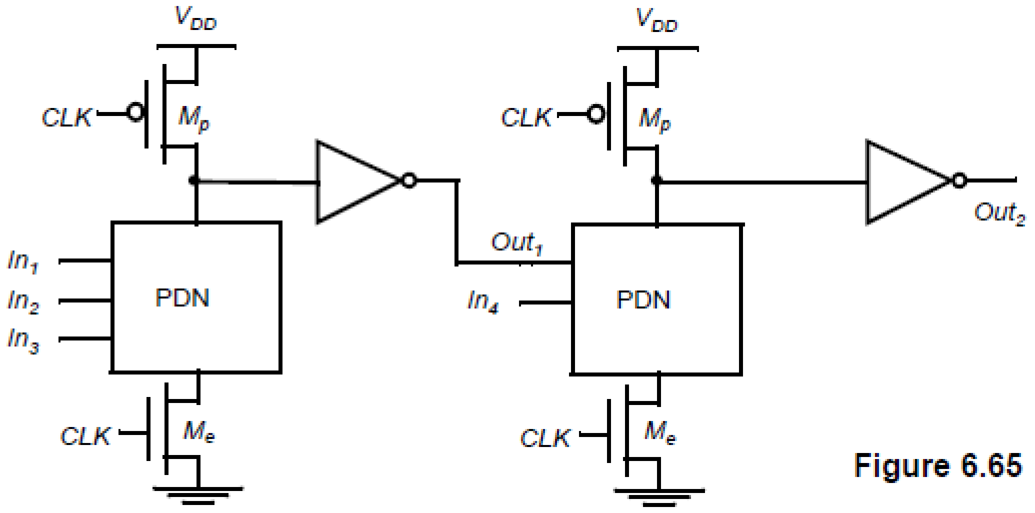
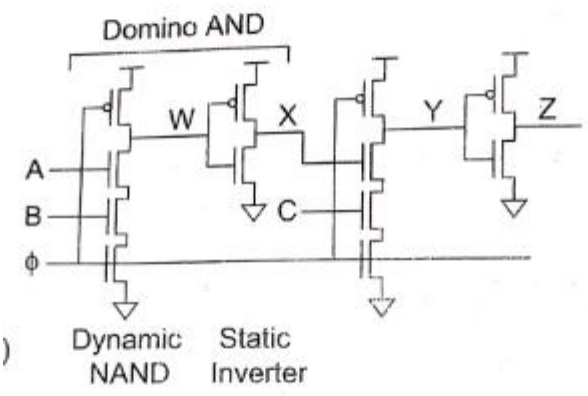
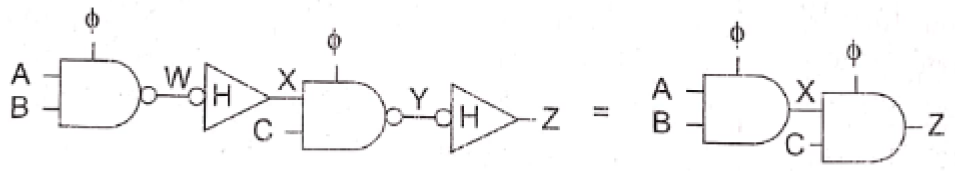
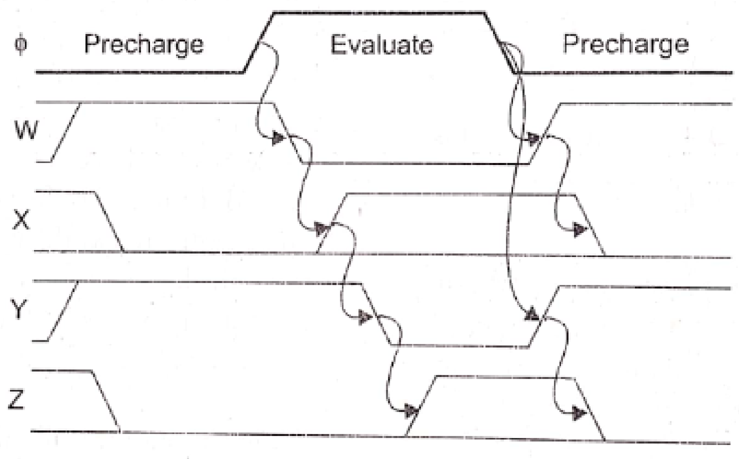


Figure 6.65 D



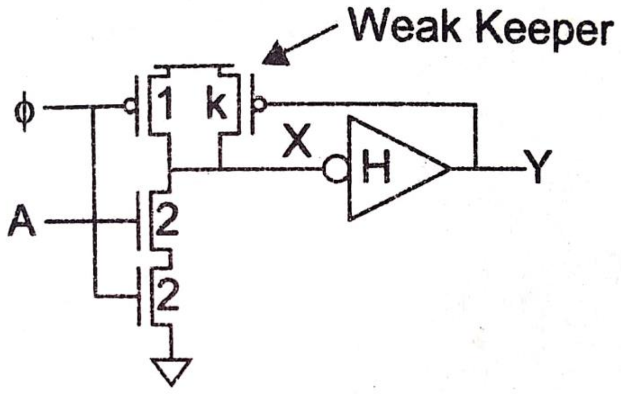


The dynamic static pair together is called a domino gate. This converts the monotonically falling signal into monotonically rising signal which is suitable for next state.
Here H-skewed static inverter is used to favor the rising output.



The precharge occurs parallel and evaluate occiurs sequentially. So precharge is usally less critical.

A second approach is to directly cascade dynamic gates without the static CMOS inverterwith delay clock to the later gates in order to ensure that the inputs are monotonic during evaluation. This is commonly usede in context addressable memories (CAM) and NOR-NOR PLA's.



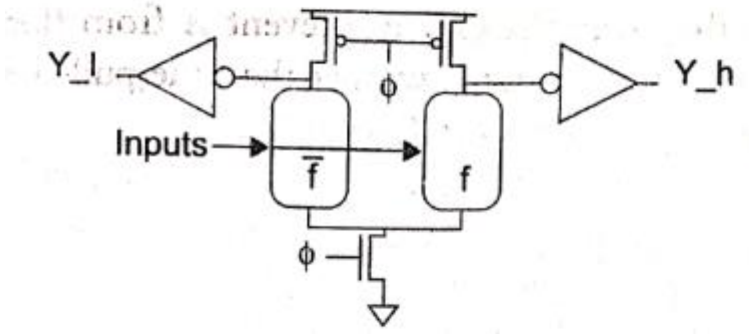
i. Dual rail Domoino logic

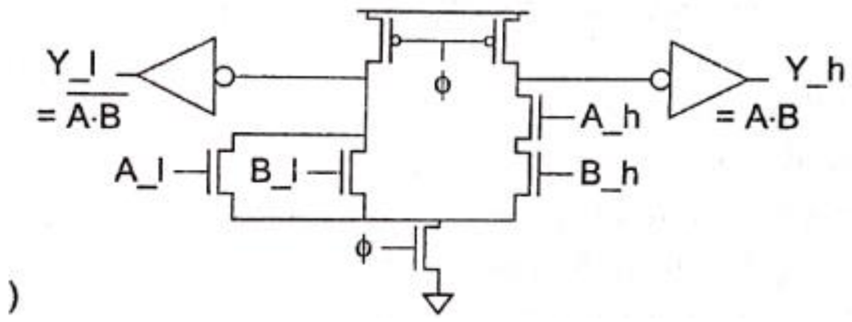
Used to implement both inverter and non-inverter functions.

DRDC accept both true and complementary input and computes both true and complementary outputs.

Similar to CVSL, pMOS are connected to precharge clock. It is also called as dynamic form of CVSL.

The disadvantage of this logic is it requires more area and more power.





i. Dual rail domino encode each signal with apair of wires.
ii. If the input wires is asserted(=1) then the output of the gate is 1 .
iii. If the complementary input is asserted high th e output is 0 .
iv. When gate is precharged, neither true or complementary input is asserted.

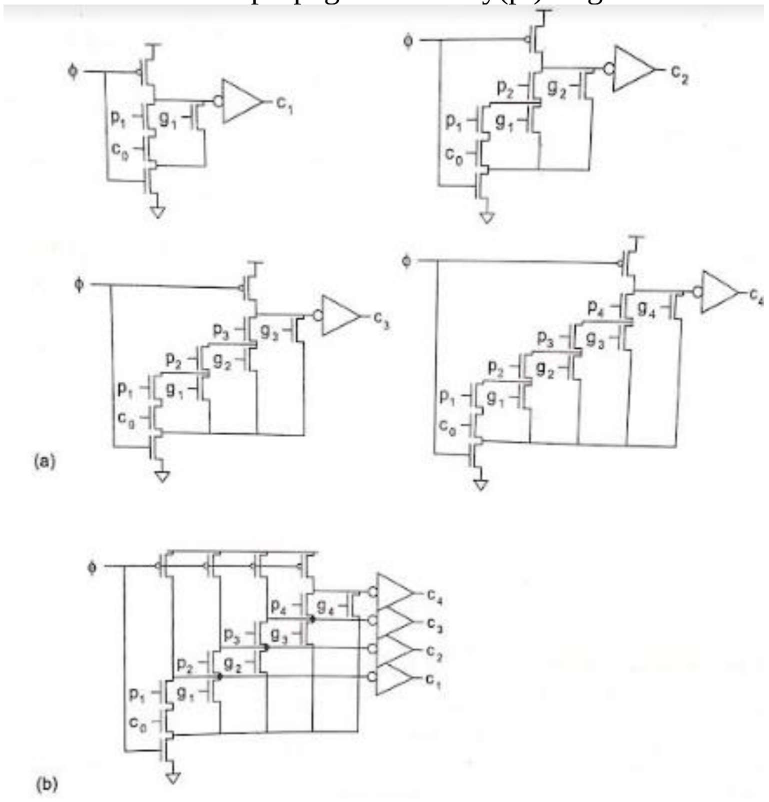
ii. **Multiple output Domino logic**

MODL is used to compute multiple function where one function is subfunction of another or shares a subfunction.

MODL saves area combining the computations into a multiple output gate.

EX: 4 bit carry propagation adder.

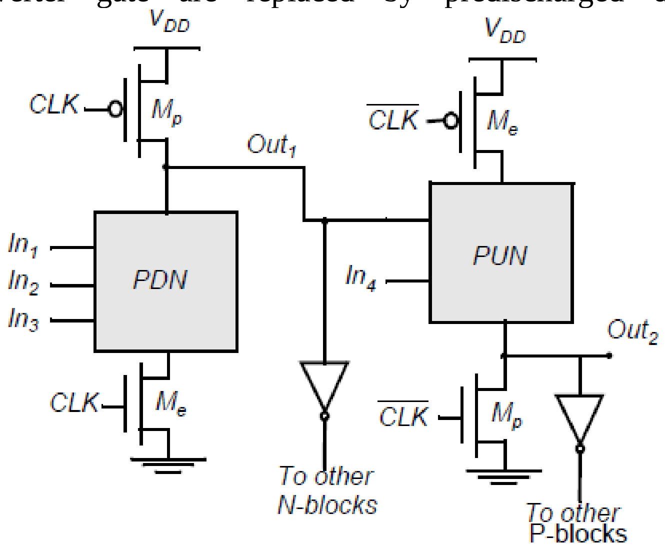
Each block can propagate the carry(pi) or generate a carry (yi). The carry out logic is



with above equations

Np and zipper domino

Another variation of domino circuit is NP domino. In this, the Hi- skewed static CMOS inverter gate are replaced by predischarged dynamic gates using pMOS logic.



When $_\varphi$=0 then 1 and 3 stage precharge high while the second stage discharges low.

When $_\varphi$=1 all the stages evaluate.

\

Power dissipation

Static CMOS gates are very power efficient because they dissipate nearly zero power while it is idle state.

INSTANTANEOUS POWER P(t): The p(t) drawn from the power supply is proportional to the supply current of iDD(t) and the supply voltage VDD.

$P(t) = i_{DD}(t)V_{DD}$

ENERGY(E): The energy consumed over sometime interval T is the integral of the instantaneous power.

$$E = \int_0^T \mathrm{iDD(t)VDDdt}$$

AVERAGE POWER (Pavg): The average power over this interval

Pavg=E/T =$\frac{1}{T}\int_0^T \mathrm{iDD(t)VDDdt}$

Power dissipation is CMOS circuits is due to

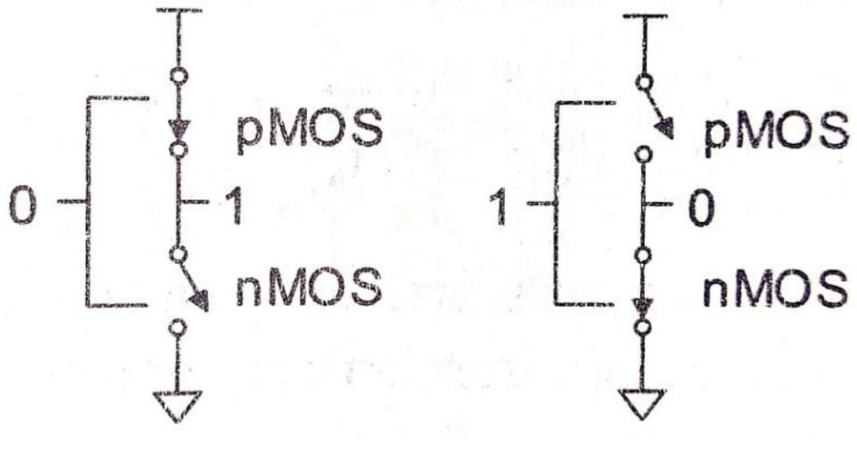
1. Static power dissipation
2. Dynamic power dissipation

a) **Static power dissipation:**

It is due to

1. Subthreshold conduction OFF transistors
2. Tunneling current through biased gates.
3. Leakage through reverse biased diodes
4. Contention current in ratioed circuits

Consider the static CMOS inverter, if the input = 0 then nMOS – OFF and pMOS – ON. So the output voltage is VDD or logic 1.



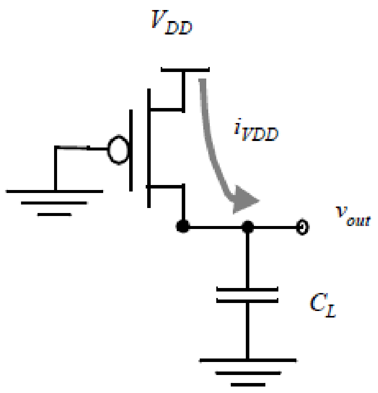
- When the input = 1 nMOS→ON and pMOS→ OFF so the output voltage is gnd or logic 0.
- i.e. One of the transistor is always OFF for any of these logic inputs. Ideally no current flows through OFF, so the power dissipation is zero. This is the advantage of CMOS over other transistor technologies.
- However the secondary effects such as subthreshold conduction tunneling and leakage leads to small amount of static current flowing through the OFF transistors. So assume these leakage current is constant.
- Therefore the instantaneous and nv power are the same. Then the static p.d. is the product of total leakage current and the supply voltage.

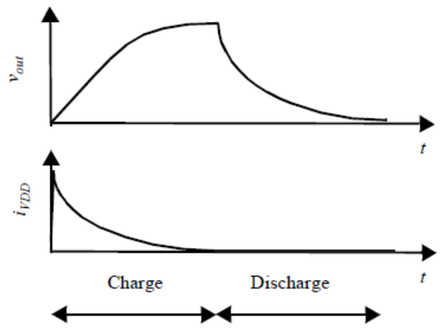
Pstatic = Istatic.VDD

b) Dynamic disspation:

It arises due to

- Charging and discharging of load capacitance
- Consider the CMOS inverter with load capacitance if Vin = 0 pMOS ON, nMOS OFF. So the current flows from VDD
- toVout to charge the load capacitance. Similarly if Vin=1 pMOS OFF nMOS ON then the current flows from load to gnd during discharging.
- So a load CL is switched between gnd and VDD at an average flow of fsw times.
- Over any interval of time 'T', the load will be charged and discharged T fsw times.
- In one complete cycle a total charge CS=CL VDDis transferred from VDD to gnd.





- The average dynamic power dissipation is

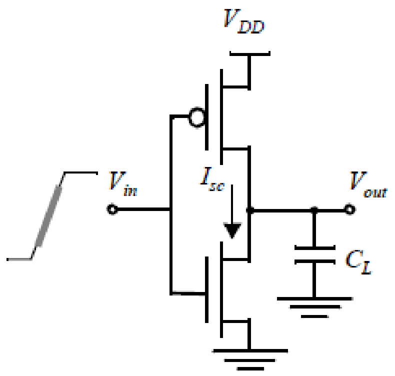
$$E_{VDD} = \int_0^\infty i_{VDD}(t)V_{DD}dt = V_{DD}\int_0^\infty C_L\frac{dv_{out}}{dt}dt = C_LV_{DD}\int_0^{V_{DD}} dv_{out} = C_LV_{DD}^2$$

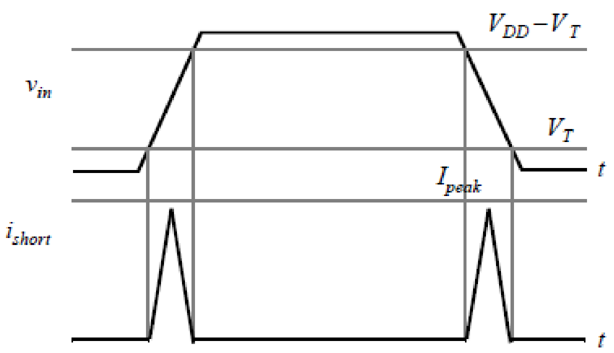
$$P_{dyn} = C_LV_{DD}^2f_{0\to 1}$$

- The fsw can be replaced by αf because most gates do not switch every clock cycle.
- Ex: α of clock is 1 because it rises and falls every cycles but most of the data has a maximum activity factor 0.5, static cMos circuit has a activity factor of 0.1

Short circuit power dissipation:

- It occurs when a direct current path between Vdd and Gnd for a short period of time during switching. During this period nMOS and pMOS transistors are conducting simultaneously.





- In general the assumption of zero rise and fall time of the input waveforms is not correct. The slope of the input signal causes direct current path from Vdd to gnd.

- The energy consumed/ switching period is given by 0→1 1→0

$$E_{dp} = V_{DD}\frac{I_{peak}t_{sc}}{2} + V_{DD}\frac{I_{peak}t_{sc}}{2} = t_{sc}V_{DD}I_{peak}$$

- Thereby the power consumption is

P = Esc /t

$$P_{dp} = t_{sc}V_{DD}I_{peak}f = C_{sc}V_{DD}^2 f$$

$$C_{sc} = t_{sc}I_{peak}/V_{DD}$$

Therefore $P_{SC} = C_{SC} V_{DD}V_{DD}\, f = C_{SC} V_{DD}^2 f$

- The short circuit dissipation is maintained by
  i. Increasing output rise/fall time larger than input rise/fall time
  ii. Reducing supply voltage.

<u>LOW POWER DESIGN:</u>

- The total power dissipation is Ptotal= Pstatic+Pdynamic.
- Power dissipation is extremely important to VLSI design because it will limit the performance of the system. So low power consumption is generally achieved by careful design .
- Power reduction technique can be classified into
  1.Dynamic power reduction
  2.Static power reduction

Dynamic power reduction:

Dynamic can be reduced by

1. Activity power reduction
2. Reducing switching capacitance
3. Reducing power supply
4. Reducing operating frequency

Reducing switching capacitance

- Device switching capacitance is reduce by choosing small transistors.so minimum sized gates can be used on non critical parts.
- Interconnect-switching capacitance can be reduced by careful floorplanning.

Reducing power supply

- Voltage is quadratically proportional to Pdynamic. So choosing the voltage supply significantly reduces power consumption.
- Voltage can be adjusted based on operating modes.
- Eg: Laptop processor may operate at high voltage and high speed when it is connected to AC adapter. But a lower voltage and speed when on battery power.

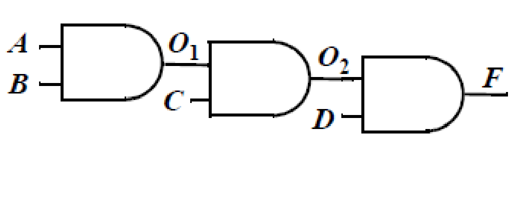
Reducing operating frequency

- In a DSP system, two multipliers running at half speed can be replaced by a single multiplier with full speed.
- In situations like this the frequency requirement is lowered.

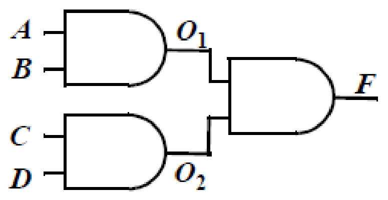
Activity factor reduction

a. Logic restructuring

The topology of a logic network may reduce its power dissipation. Consider the following function F=ABCD



Chain structure

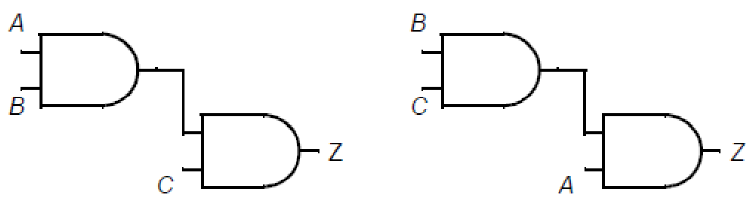


Tree structure

Out of these topologies, the chain structure as an lower switching activity then the tree structure for random input's.

Input ordering:

- The switching activity will be reduced when the input signal with a high transition rate is postponed or delayed. To explain this consider the following two static logic circuits.
- The probability that A is equal to 1 is 0.5. P(A=1) = 0.5
- Similarly P(B=1)=0.2 P(C=1)=0.1

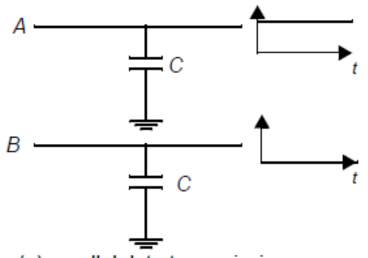


$p_{(A=1)} = 0.5$
$p_{(B=1)} = 0.2$
$p_{(C=1)} = 0.1$

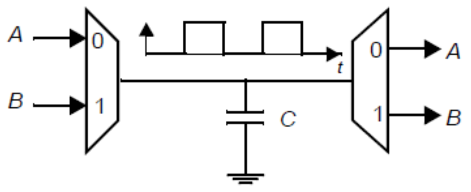
- In the first circuit; the prob that a 0 → 1 transition at the intermediate node (x) is given by = (1-PA PB)(PA PB) =(1-0.2*0.1)(0.2*0.1) = 0.0196.
- The probability that a 0 → 1 transition for second circuit is less than the first circuit. A simple reordering of the input signals are often sufficient to reduce the switching activity.

TIME MULTIPLEXING RESOURCES:

- Time multiplexing is a single hardware resource such as logic unit or a bus. It is used to minimize implementation area. But it does not reduce the switching activity always.
- For eg; consider the transmission of two input bits A and B using dedicated resources.



(a) parallel data transmission

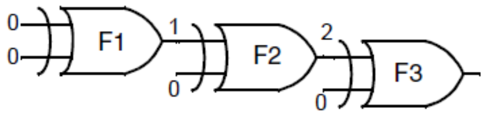


(b) serial data transmission

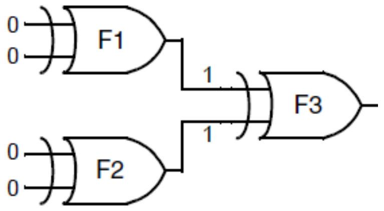
-
- Suppose A is always 1 and B is mostly 0; the switched capacitor is very low since there are very few transitions on the data bits.
- But if the data being transmitted are random; there is no difference between the parallel and serial architecture.
- In second case; the inputs A and B are transmitted by using time multiplexed approach. In this approach; the bus toggles between 0 and 1.

Glitch reduction by a balancing signal path:

- A node (or output) can exhibit multiple transitions in a single clock cycle before it is settling to the circuit logic level. These spurious transitions are known as **glitches or dynamic hazards**.
- The occurrence of glitches in a circuit is mainly due to mismatch in the path length of the network.
- It all the input signal of a gate change simultaneously no glitching occurs. But if the input signal changes at different times glitches will develop



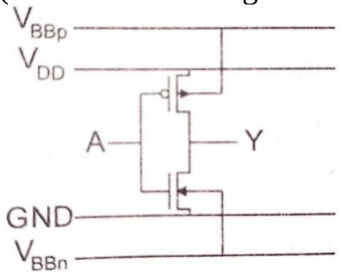
(a) Network sensitive to glitching

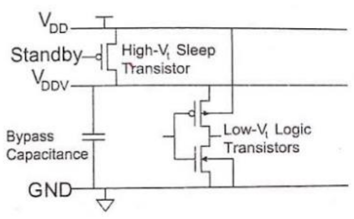


(b) Glitch-free network

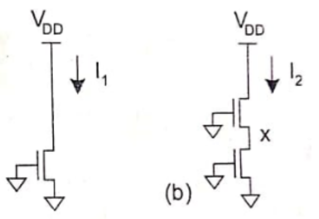
The first network suffers from glitching because of the different arrival time of the input signals for a gate.
- The second network does not affected by the glitches because all the outputs are arrived simultaneously.

STATIC POWER REDUCTION:
- Static power reduction involves minimizing Istatic
- The sub threshold current for Vgs<Vt is **EQUATION**
- So this leakage current can be reduced by
1. Lowering the temp (T decreases so Ids decreases)
2. Increasing Vsb (i.e. which will be the threshold Vge)
- Another way to control leakage is through adaptive body bias.
- Low Vt devices can be connected to RBB (reverse body bias) voltage during idle mode to limit the leakage.
- High Vt devices connected to FBB during active mode to reduce the performance.
- Applying body bias requires additional power supply rails to distribute and bias voltages.
- ((For 2nd and 3rd diagram – refer class notes))





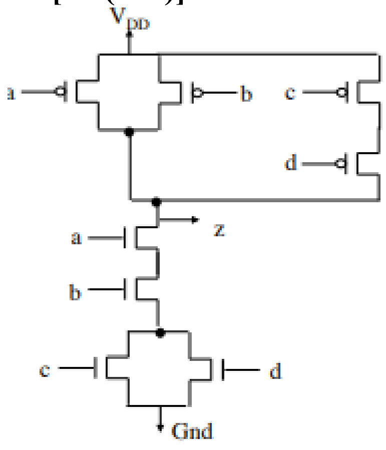
MTCMOS



Leakage stack effect

Examples of Combinational Circuits

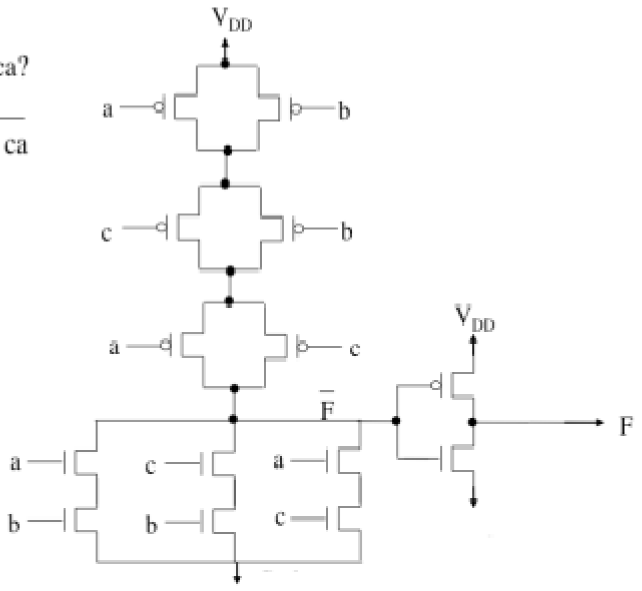
1. **Z= [ab.(c+d)]'**



2. **Example 2**

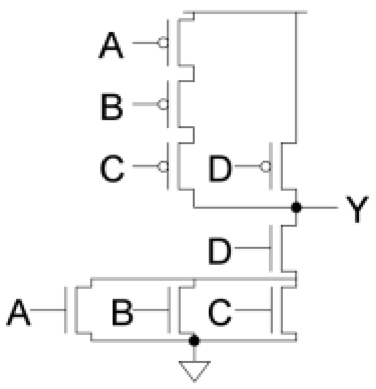
$F = ab + bc + ca?$

• $\bar{F} = \overline{ab + bc + ca}$



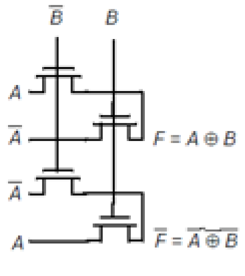
3. Example

$Y = \overline{(A + B + C) \cdot D}$



PART A

1. **Draw a 2 input XOR gate using nMOS pass transistor logic? A/M 19**



XOR/NXOR

2. **Determine the discharging time of the circuit shown in fig. When switch A is closed**

3. **Assume CL and internal capacitances C1 and C2 are charged initially. Let C1=C2=Cl=C? N/D 18**

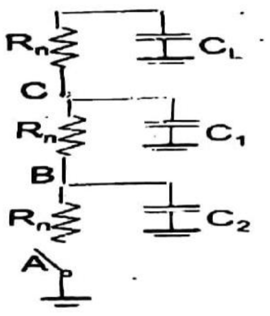
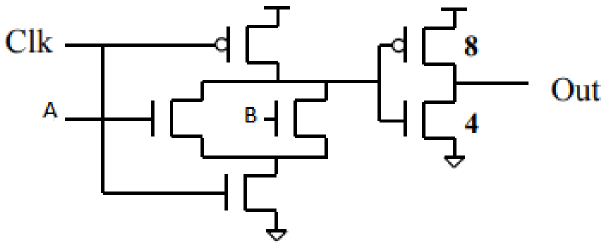
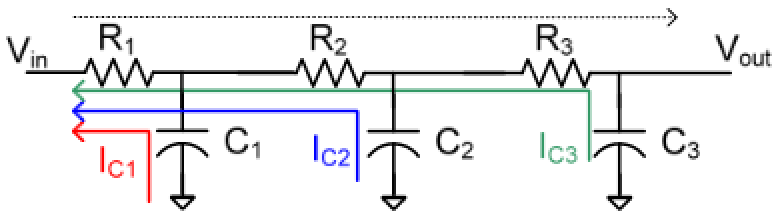


Fig 1

4. **Realise X=B+C and Y=(A.(B+C)) using multiple output domino stages?N/D 18**



5. **Define elmore's constant? A/M18, N/D 17, A/M 17**
A chain of transistors can be represented as an RC ladder as shown below



$$\boxed{\tau_{Elmore} = R_1C_1 + (R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3}$$

6. **List the types of power dissipation? A/M 19, A/M18, N/D 17**
   Static power dissipation
   Dynamic power dissipation

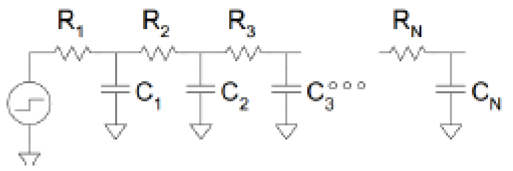
7. **State advantages of transmission gates? A/M 17**
   (I) It has double rail logic.
   (II) Transmission gate passes both 0's and 1's.

8. **List out sources of static and dynamic power consumption? N/D 16**
   The two sources of power dissipation are the dynamic power dissipation due to charging and discharging of load capacitance and the power disputation due to the sub threshold leakage.

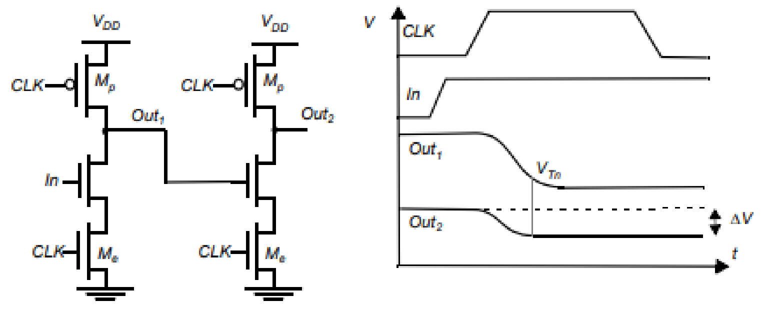
9. **What is the value of Vout in the figure shown below, Where Vth is the threshold voltage of the transistor N/D 16**
   VDD -VTn

10. **Give Elmore delay expression for propagation delay of an inverter? A/M 16**

$$t_{pd} \approx \sum_{\text{nodes } i} R_{i-to-source} C_i$$
$$= R_1C_1 + (R_1 + R_2)C_2 + \ldots + (R_1 + R_2 + \ldots + R_N)C_N$$

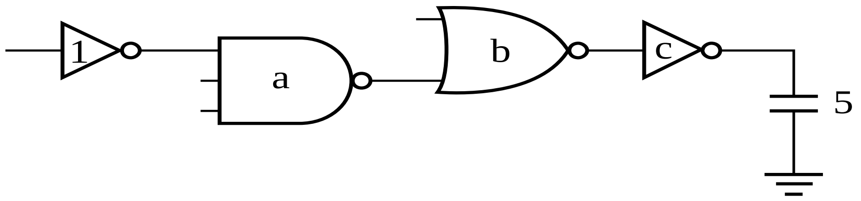


11. Why single phase dynamic logic structure cannot be cascaded? Justify? A/M 16
Dynamic gates cannot be directly cascaded. To illustrate this, consider two simple N-type dynamic inverters cascaded together, as shown in Figure. During the precharge phase (i.e., CLK =0), the output of both inverters are precharged up to VDD. Assume that the primary input In makes a 0 to 1 transition (Figure b). On the rising edge of the clock, output Out1 starts to discharge. The second output should remain in the precharged state of VDD since Out1 transitions to 0 during evaluation. However, since there is a finite propagation delay for the input to discharge Out1 to GND, the second output also starts to discharge. As long as Out1 exceeds the switching threshold of the second gate, which approximately equals V_{Tn}, a conducting path exists between Out2 and GND. Out2 therefore discharges as well, resulting in incorrect evaluation. This conducting path is only turned off when Out1 reaches V_{Tn} and shuts off the NMOS pull-down transistor. This leaves Out2 at an intermediate voltage level. The correct level will not be recovered, since dynamic gates rely on capacitive storage. The charge loss leads to reduced noise margins and eventual malfunctioning.



PART B

1. Realise a 2 input XOR using static CMOS logic, transmission gate and dynamic CMOS logic. Analyse the hardware complexity. **A/M 19-Part C**
2. Derive an expression for dynamic power dissipation. **A/M 19 (OR)** Explain static and dynamic power dissiptation in CMOS circuit with necessary diagram and expression? A/M 17 **(OR)** What are the sources of power dissipation in CMOS and discuss various design techniques to reduce power dissipation in CMOS. **A/M 16 (OR)** explain the dynamic power dissipation in CMOS circuit with diagram? **N/D 16**
3. Realise the function Y=(A+BC)D+E using static CMOS logic. **A/M 19 (OR)** Draw the CMOS logic circuit for the Boolean expression z=[A(B +C)+DE]' **A/M 18,N/D 17 (OR)** Draw the static CMOS logic circuit for the following expression **A/M 18-Part C** Y= [(A+B). (C+D)]' **(OR)**Draw the static CMOS logic circuit for the following expression **A/M 16** a) Y=(A.B.C.D) b) Y=D(A+BC)
4. Let A,B and C be the inputs of a data selector and S0 and S1 be the select lines. Realise a 4:1 data selector using i) nMOS pass transistor and ii) transmission gate approach. Compare the hardware complexity. **A/M 19**
5. (a)(ii) Explain the principle for transmission gate in CMOS design? **A/M 18, N/D 17 (OR)** Discuss in detail the characteristics of CMOS transmission gate? **A/M 16 (OR)** What is transmission gate ? explain uses of transmission gate? A/M 17
6. Design a four input NAND gate and obtain its delay during the transition from high to low? **A/M 18-Part C**
7. Obtain the logical effort and path effort of the given circuit? **A/M 18-Part C**



8. write short notes on
   i. Ratioed circuits: **N/D 16**
   ii. dynamic CMOS circuits: **N/D 16**
   iii. DCVSL logic with suitable example? A/M 17
   iv. signal integrity issues in dynamic design. **A/M 18**
   v. domino logic with diagram? **N/D 17**

DEPARTMENT OFELECTRONICSAND COMMUNICATION ENGINEERING

(ACADEMIC YEAR: 2019-2020)

EC8095-VLSI DESIGN

(Regulation 2017)

Semester-VI

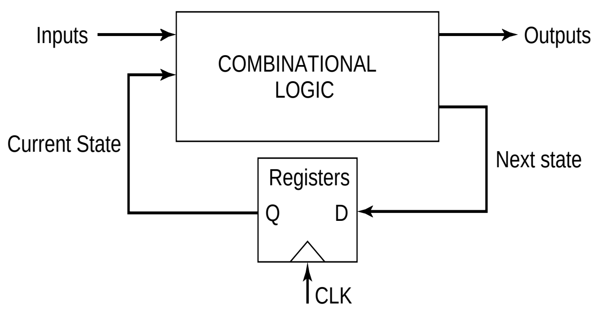
NAME-

REG NO-

MOHAMED SATHAK
A J COLLEGE OF ENGINEERING
SIRUSERI IT PARK, OMR, CHENNAI - 603103

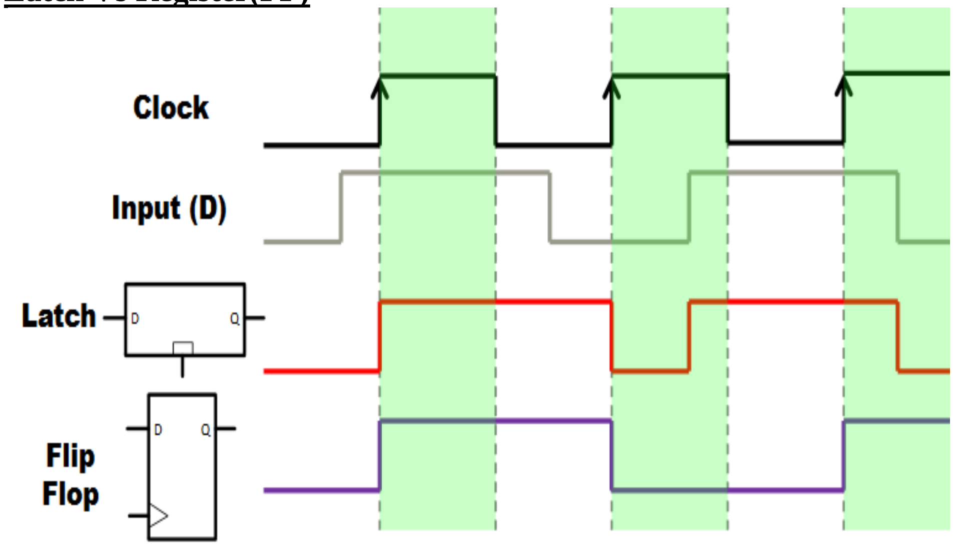
UNIT III SEQUENTIAL LOGIC CIRCUITS

Sequential Logic



The Outputs & Next State of the FSM are a function of the Current Inputs and the Current State. The next state is fed to the inputs of registers.
On the rising edge of the clock, the Next State bits are copied to the outputs of the registers (after some propagation delay),and a new cycle begins.
The register then ignores changes in the input signals until the next rising edge.

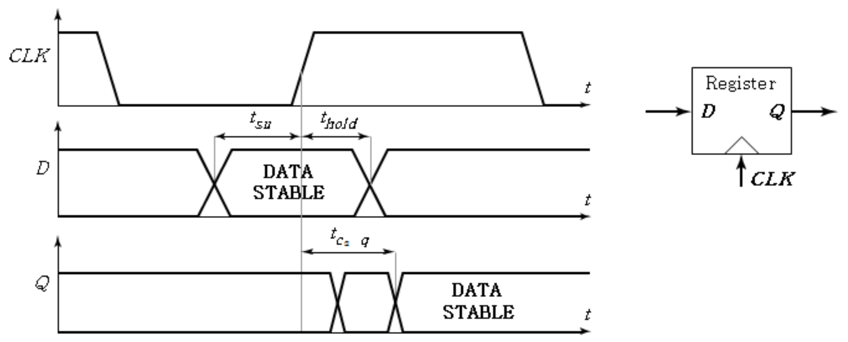
Latch Vs Register(FF)

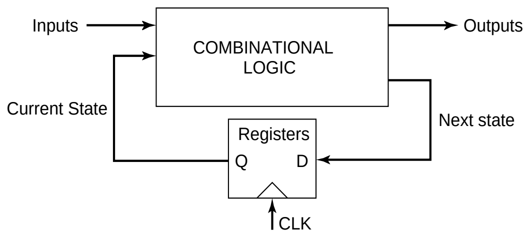


- a latch is level sensitive
- a register (or FF) is edge-triggered

Timing

The set-up time (tsu) is the time that the data inputs (D input) must be valid before the clock transition (this is, the 0 to 1 transition for a positive edge-triggered register).
The hold time (thold) is the time the data input must remain valid after the clock edge.
The data at the D input is copied to the Q output after a worst-case propagation delay(with reference to the clock edge) denoted by tc-q.





Assume that the worst-case propagation delay of the logic equals tplogic.
The minimum clock period T, required for proper operation of the sequential circuit is given by

$$T \geq t_{c-q} + t_{plogic} + t_{su}$$

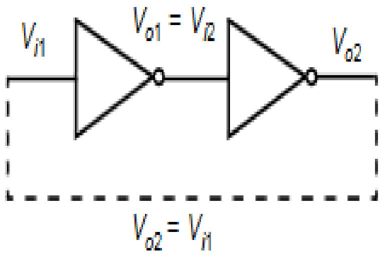
Classification of Memory Elements

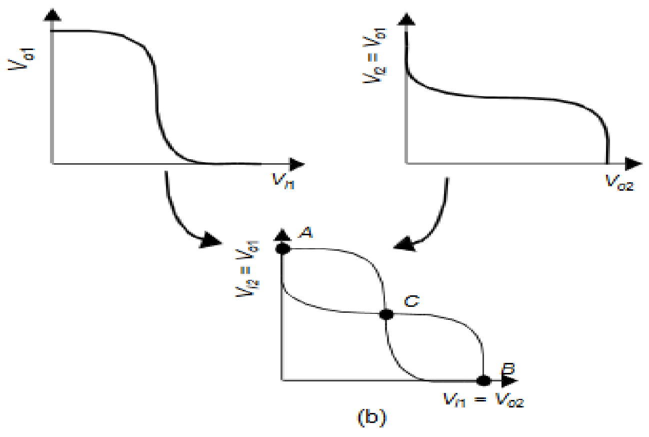
Memories can be static or dynamic.

Static Memory (or) Static Latches and Registers

- Static memories preserve the state as long as the power is turned on.
- Static memories are built using positive feedback or regeneration, where the circuit topology consists of intentional connections between the output and the input of a combinational circuit.
- Static memories are most useful when the register won't be updated for extended periods of time.
- Memory based on positive feedback fall under the class of elements called multivibrator circuits.

Bistability Principle





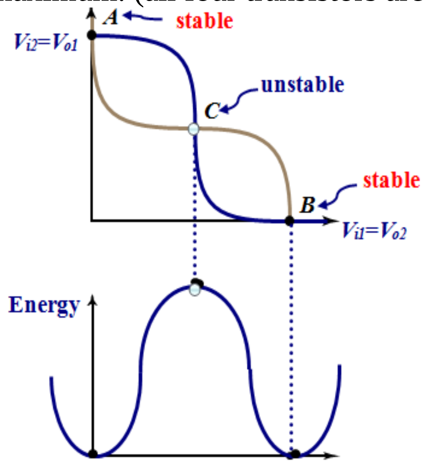
Consider two inverters connected in cascade .
The VTCs of the first inverter, that is, Vo1 versus Vi1,and the second

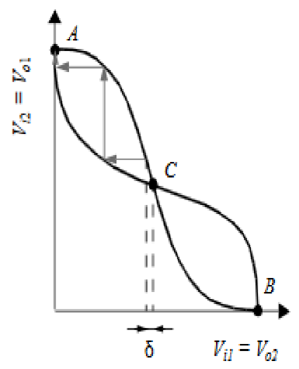
inverter (Vo2 versus Vo1).
The latter plot is rotated to accentuate that Vi2=Vo1. Assume now that the output of the second inverter Vo2 is connected to the input of the first Vi1.

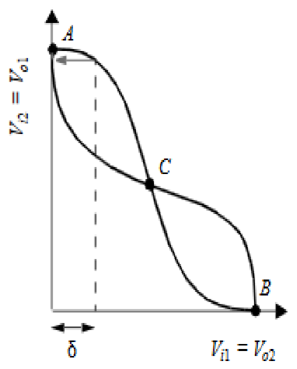
- ***A*** and ***B*** are stable points: If the circuit is initially operating at one of them, it will preserve this state. The gain is smaller than **unity**.
- ***C*** is an unstable point: The voltage gains of both inverters are larger than unity. A small voltage perturbation at this operating point will be amplified ⇒ the operating point will move to one of the stable operating points, ***A*** or ***B***.

– **Energy Levels:**

- The potential energy is at its minimum at ***A*** and ***B***, since the voltage gains of both inverters are equal to zero.
- The potential energy is at its maximum at C, since the voltage gains of both inverters are maximum. (all four transistors are in saturation modes)

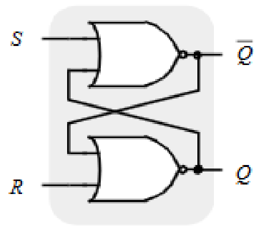


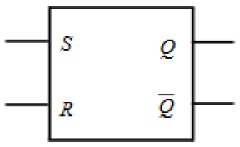


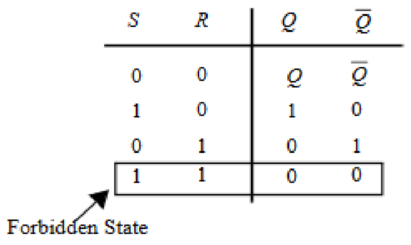


Suppose that the cross-coupled inverter pair is biased at point C.
A small deviation from this bias point, possibly caused by noise, is amplified and regenerated around the circuit loop.
A small deviation δ is applied toVi1 (biased in C). This deviation is amplified by the gain of the inverter.
The enlarged divergence is applied to the second inverter and amplified once more.
The bias point moves away from C until one of the operation points A or B is reached.
On the other hand, A and Bare stable operation points, as demonstrated in Figure. In these points, the loop gain is much smaller than unity. Even a rather large deviation from the operation point is reduced in size and disappears.

SR Flip-Flops

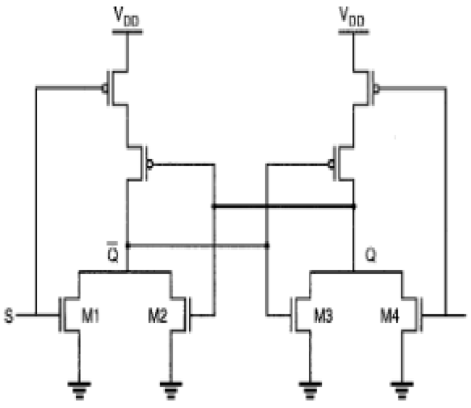


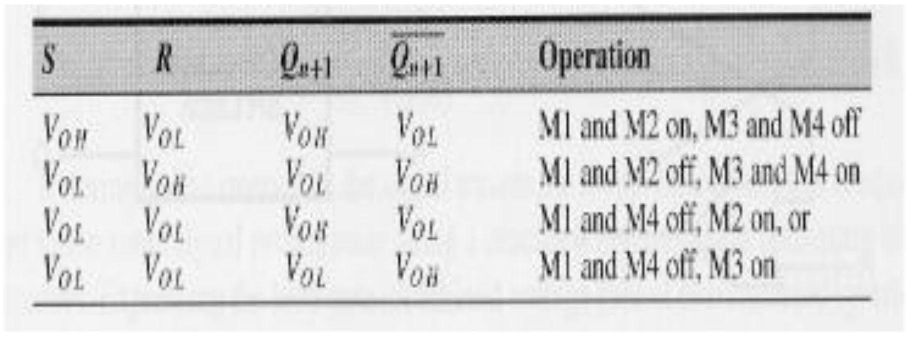




S	R	Q	$\overline{Q}$
0	0	Q	$\overline{Q}$
1	0	1	0
0	1	0	1
1	1	0	0

Forbidden State





S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
V_{OH}	V_{OL}	V_{OH}	V_{OL}	M1 and M2 on, M3 and M4 off
V_{OL}	V_{OH}	V_{OL}	V_{OH}	M1 and M2 off, M3 and M4 on
V_{OL}	V_{OL}	V_{OH}	V_{OL}	M1 and M4 off, M2 on, or
V_{OL}	V_{OL}	V_{OL}	V_{OH}	M1 and M4 off, M3 on

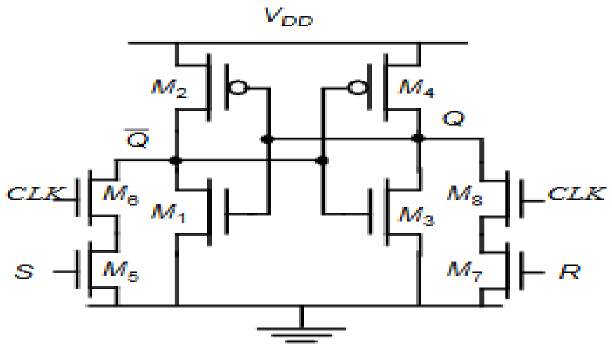
Clocked SR flip-flop

It consists of a cross-coupled inverter pair, plus 4 extra transistors to drive the flip-flop from one state to another and to provide clocked operation.

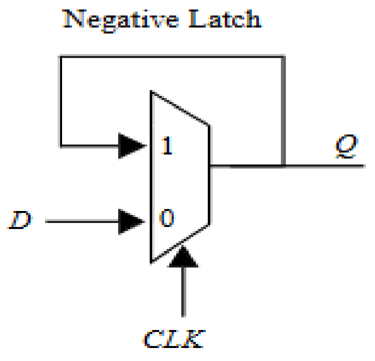
Consider the case where Q is high and an R pulse is applied. The combination of transistors M4,M7,and M8 forms a ratioed inverter.

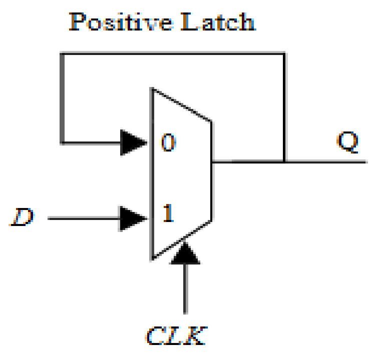
sizes of transistors : M5,M6,M7,andM8 > M1,M2,M3,andM4.

does not consume any static power.



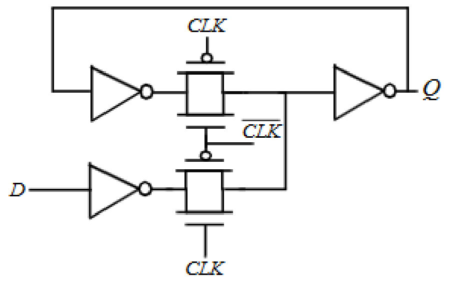
Multiplexer Based Latches



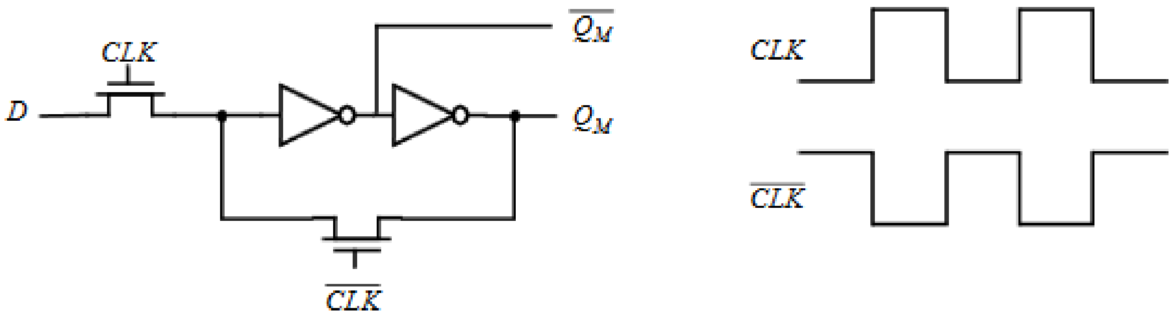


Positive latch built using transmission gates

When CLK is high, the bottom transmission gate is on and the latch is transparent-that is, the D input is copied to the Q output. During this phase, the feedback loop is open since the top transmission gate is off.

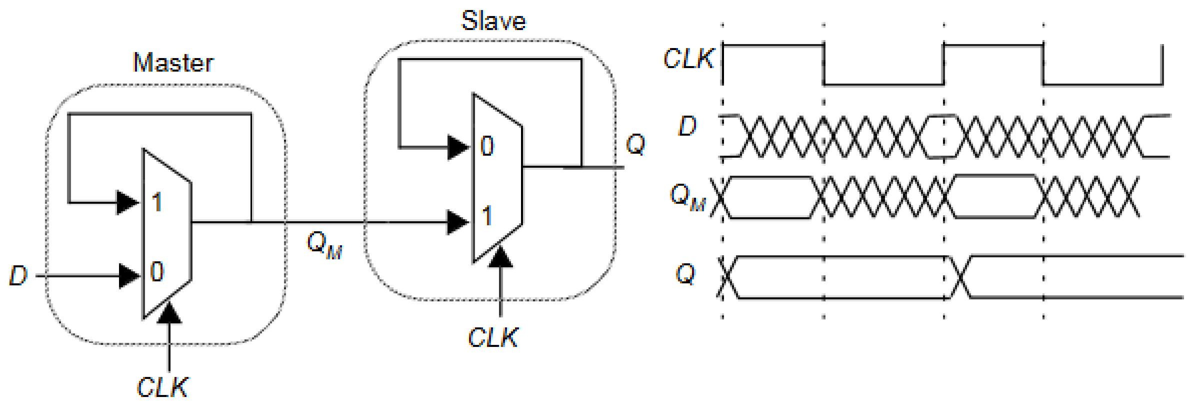


Multiplexer based NMOS latch using NMOS only pass transistors :

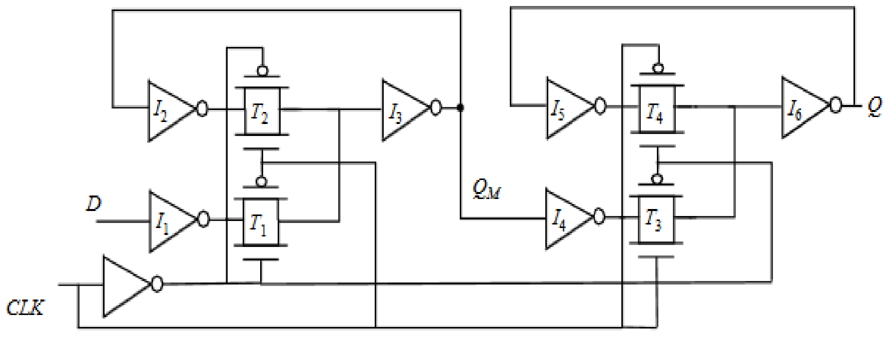


When CLK is high, the latch samples the D input, while a low clock-signal enables the feedback-loop, and puts the latch in the hold mode.
NMOS only pass transistors - pass a degraded high voltage (VDD-Vtn) to the input of the first inverter. This impacts both noise margin and the switching performance,

Master-Slave Based Edge Triggered Register



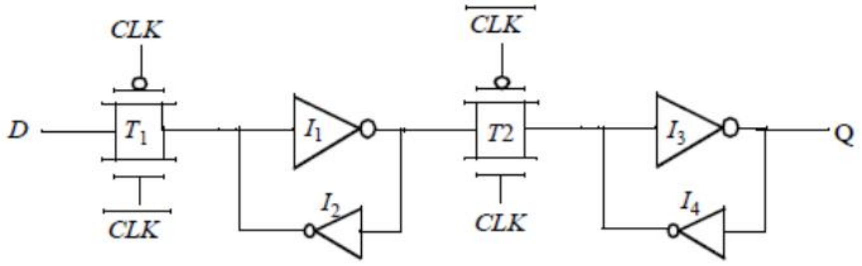
On the low phase of the clock, the master stage is transparent and the D input is passed to the master stage output, QM.
During this period, the slave stage is in the hold mode, keeping its previous value using feedback.
On the rising edge of the clock ,the master slave stops sampling the input, and the slave stage starts sampling. During the high phase of the clock, the slave stage samples the output of the master stage (QM), while the master stage remains in a hold mode



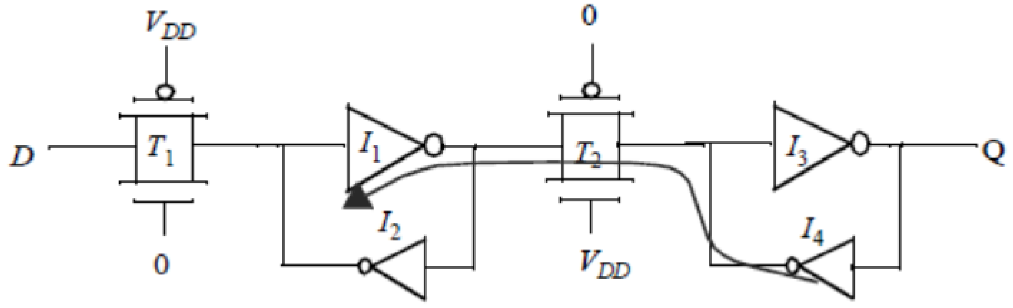
Timing Properties of the multiplexer Based Master-Slave Register.

- Assume that the propagation delay of each inverter is tpd_inv and the propagation delay of the transmission gate is tpd_tx. Also assume that the contamination delay is 0 and the inverter delay to derive CLK from CLK has a delay equal to 0.
- For the transmission gate multiplexer-based register, the input D has to propagate through I1, T1, I3 and I2 before the rising edge of the clock. This is to ensure that the node voltages on both terminals of the transmission gate T2 are at the same value. Otherwise, it is possible for the cross-coupled pair I2 and I3 to settle to an incorrect value. The set-up time is therefore equal to 3 *tpd_inv + tpd_tx .
- The propagation delay is the time for the value of QM to propagate to the output Q. Note that since we included the delay of I2 in the set-up time, the output of I4 is valid before the rising edge of clock. Therefore the delay tc-q is simply the delay through T3 and I6 (tc-q = tpd_tx + tpd_inv). • The hold time represents the time that the input must be held stable after the rising edge of the clock. In this case, the transmission gate T1 turns off when clock goes high and therefore any changes in the D-input after clock going high are not seen by the input.
- Therefore, the hold time is 0.

Reduced load clock load static master slave register.



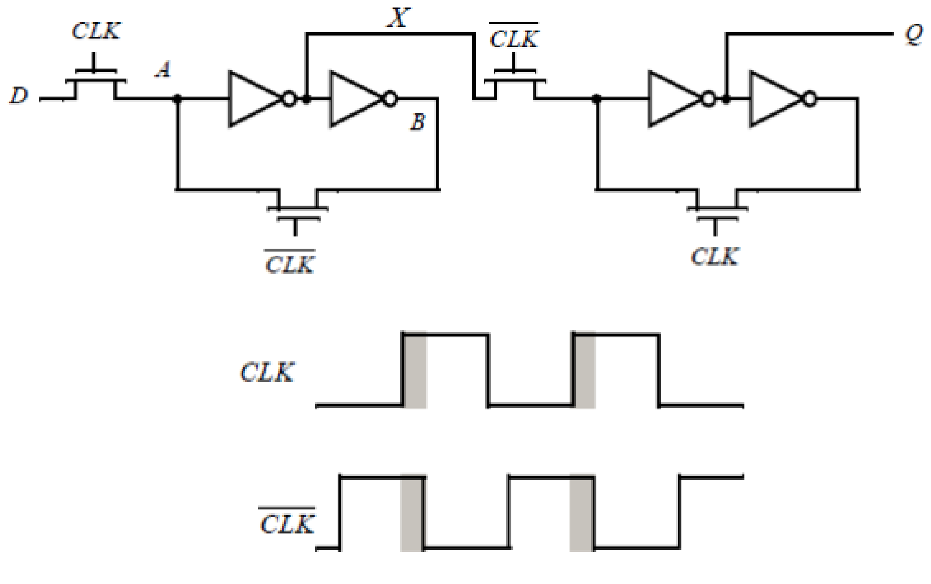
The penalty for the reduced clock load is increased design complexity. The transmission gate (T1) and its source driver must overpower the feedback inverter (I2) to switch the state of the cross-coupled inverter. The sizing requirements for the transmission gates can be derived using a similar analysis as performed for the SR flip-flop. The input to the inverter I1 must be brought below its switching threshold in order to make a transition. If minimum-sized devices are to be used in the transmission gates, it is essential that the transistors of inverter I2 should be made even weaker. This can be accomplished by making their channel-lengths larger than minimum. Using minimum or close-to-minimum size devices in the transmission gates is desirable to reduce the power dissipation in the latches and the clock distribution network.



Another problem with this scheme is the reverse conduction — this is, the second stage can affect the state of the first latch.
When the slave stage is on , it is possible for the combination of T2 and I4 to influence the data stored in I1-I2 latch. As long as I4 is a weak device, this is fortunately not a major problem.

Non-ideal clock signals

- So far, we have assumed that CLK is a perfect inversion of CLK, or in other words, that the delay of the generating inverter is zero.
- Variations can exist in the wires used to route the two clock signals, or the load capacitances can vary based on data stored in the connecting latches. This effect, known as clock skew is a major problem, and causes the two clock signals to overlap.

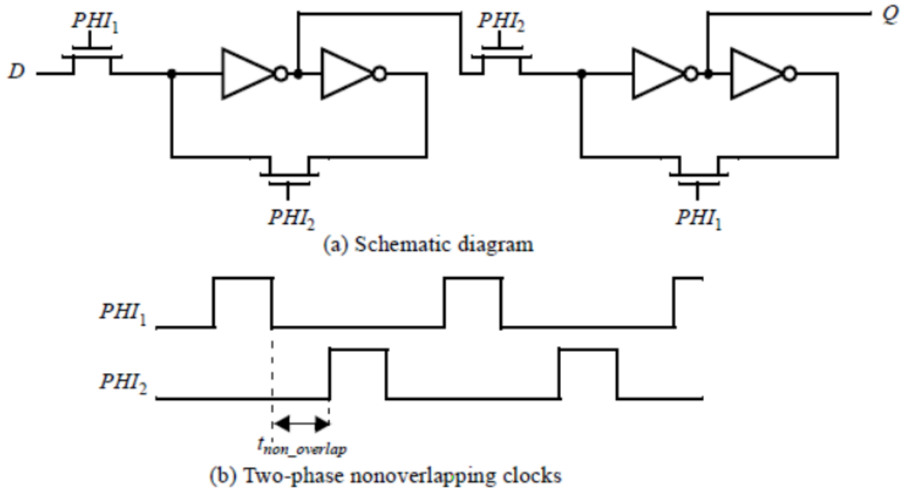


(b) Overlapping clock pairs

When the clock goes high, the slave stage should stop sampling the master stage output and go into a hold mode.
However, since CLK and CLK are both high for a short period of time (the overlap period), both sampling pass transistors conduct and there is a direct path from the D input to the Q output.
As a result, data at the output can change on the rising edge of the clock, which is undesired for a negative edge triggered register.

The is known as a **race condition** in which the value of the output Q is a function of whether the input D arrives at node X before or after the falling edge of CLK.
If node X is sampled in the metastable state, the output will switch to a value determined by noise in the system.
The primary advantage of the multiplexer-based register is that the feedback loop is open during the sampling period, and therefore sizing of devices is not critical to functionality. However, if there is clock overlap between CLK and CLK, node A can be driven by both D and B, resulting in an undefined state.

Pseudostatic two-phase D register
Those problems can be avoided by using two nonoverlapping clocks PHI1 and PHI2 instead , and by keeping the nonoverlap time tnon_overlap between the clocks large enough such that no overlap occurs even in the presence of clock-routing delays.
During the nonoverlap time, the FF is in the highimpedance state—the feedback loop is open, the loop gain is zero, and the input is disconnected. Leakage will destroy the state if this condition holds for too long a time.
Hence the name pseudostatic: the register employs a combination of static and dynamic storage approaches depending upon the state of the clock.

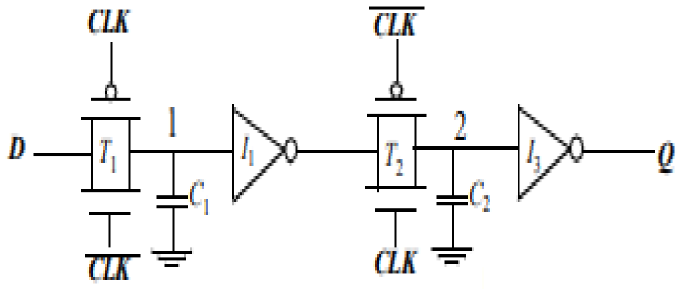


<u>Dynamic memories (or) Dynamic Latches and Registers</u>

- Dynamic memories store state for a short period of time—on the order of milliseconds.
- They are based on the principle of temporary charge storage on parasitic capacitors associated with MOS devices.
- Dynamic memories tend to simpler, resulting in significantly higher performance and lower power dissipation.
- They are most useful in datapath circuits that require high performance levels and are periodi-cally clocked.
- charge stored on a capacitor can be used to represent a logic signal.
- The absence of charge denotes a 0, while its presence stands for a stored 1.
- No capacitor is ideal, unfortunately, and some charge leakage is always present.

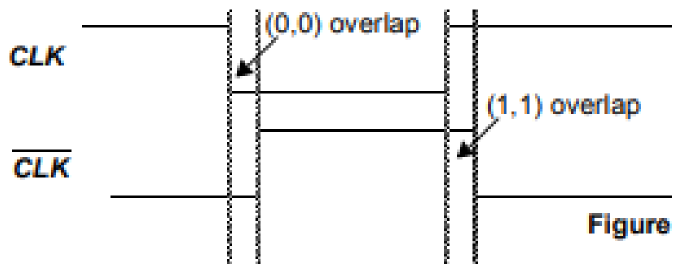
- A stored value can hence only be kept for a limited amount of time, typically in the range of milliseconds.
- If one wants to preserve signal integrity, a periodic refresh of its value is necessary

Dynamic Transmission-Gate Based Edge-triggered Registers



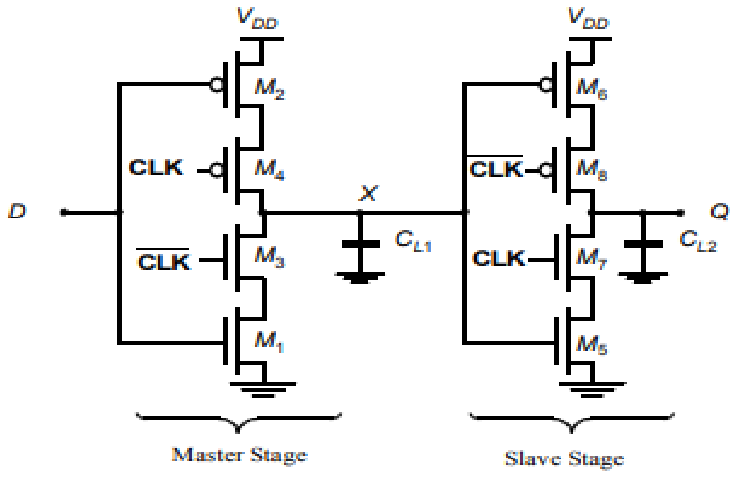
When CLK = 0, the input data is sampled on storage node 1, which has an equivalent capacitance of C1
C1= gate capacitance of I1, the junction capacitance of T1, and the overlap gate capacitance of T1.
During this period, the slave stage is in a hold mode, with node 2 in a high-impedance (floating) state.

- On the rising edge of clock, the transmission gate T2 turns on, and the value sampled on node 1 right before the rising edge propagates to the output Q (note that node 1 is stable during the high phase of the clock since the first transmission gate is turned off).
- Node 2 now stores the inverted version of node 1.
- The set-up time of this circuit is simply the delay of the transmission gate, and corresponds to the time it takes node 1 to sample the D input.
- The hold time is approximately zero, since the transmission gate is turned off on the clock edge and further inputs changes are ignored.
- The propagation delay (tc-q) is equal to two inverter delays plus the delay of the transmission gate T2.
- storage nodes (i.e., the state) has to be refreshed at periodic intervals to prevent a loss due to charge leakage, due to diode leakage as well as sub-threshold currents.
- During the 0-0 overlap period, the NMOS of T1 and the PMOS of T2 are simultaneously on, creating a direct path for data to flow from the D input of the register to the Q output.
- This is known as a race condition.



Figure

C^2 MOS Dynamic Register: A Clock Skew Insensitive Approach



- The register operates in two phases.

1. CLK =0(CLK = 1):

 The first tri-state driver is turned on, and the master stage acts as an inverter sampling the inverted version of D on the internal node X.

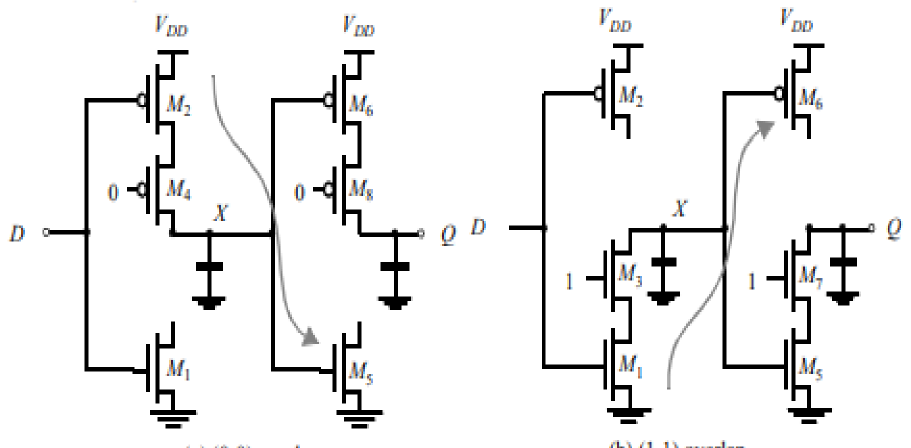
 The master stage is in the evaluation mode. Meanwhile, the slave section is in a high-impedance mode, or in a hold mode.

 Both transistors M7 and M8 are off, decoupling the output from the input. The output Q retains its previous value stored on the output capacitor CL2.

2. The roles are reversed when CLK = 1:

 The master stage section is in hold mode (M3- M4 off), while the second section evaluates (M7-M8 on).

 The value stored on CL1 propagates to the output node through the slave stage which acts as an inverter.

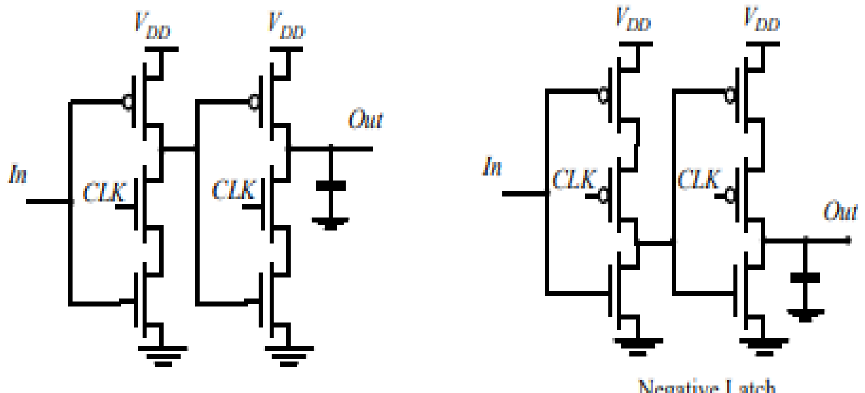


(a) (0-0) overlap (b) (1-1) overlap

- Clock overlaps activate either the pull-up or the pull-down networks of the latches, but never both of them simultaneously.
- If the rise and fall times of the clock are sufficiently slow, however, there exists a time slot where both the NMOS and PMOS transistors are conducting. This creates a path between input and output

- The circuit operates correctly as long as the clock rise time (or fall time) is smaller than approximately five times the propagation delay of the register nd output that can destroy the state of the circuit.

True Single-Phase Clocked Register (TSPCR)



Positive Latch

Negative Latch

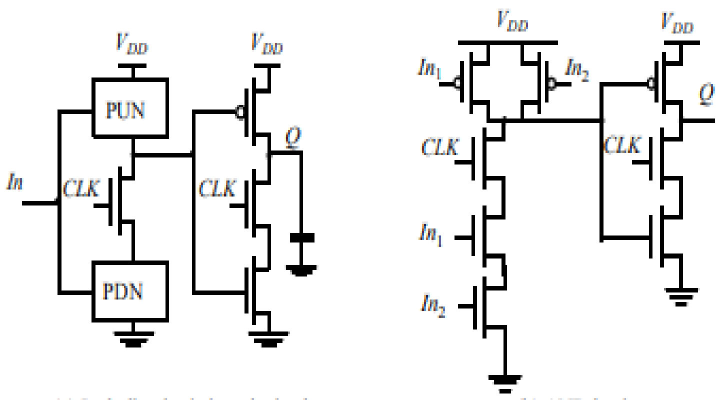
For the positive latch, when CLK is high, the latch is in the transparent mode.

- ➢ corresponds to two cascaded inverters;
- ➢ the latch is non-inverting, and propagates the input to the output.
- ➢ On the other hand, when CLK = 0, both inverters are disabled, and the latch is in hold-mode. Only the pull-up networks are still active, while the pull-down circuits are deactivated.

Positive latch operation

- When clock=1 and In=0
  M2,M3-ON; M1-OFF; X=1
  M4,M6 –ON;M5-OFF
  Therefore Out=In=0
- When clock=1 and In=1
  M1,M3-ON; M2-OFF; X=0
  M4,M5 –ON;M6-OFF
  Therefore Out=In=1

- The main advantage is the use of a single clock phase.
- The disadvantage is the slight increase in the number of transistors — 12 transistors are required.



(a) Including logic into the latch (b) AND latch

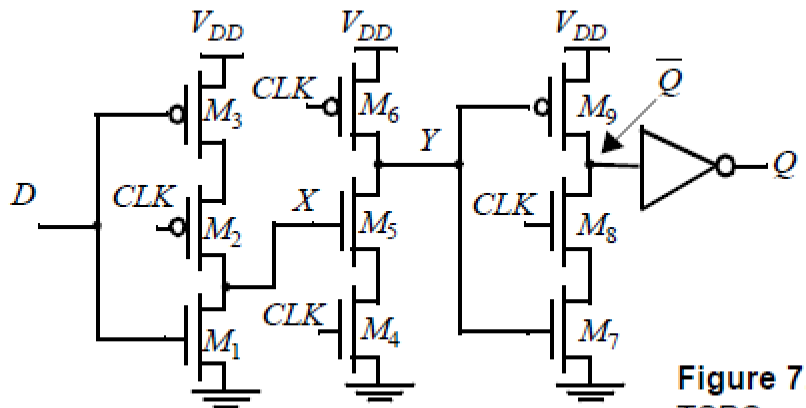
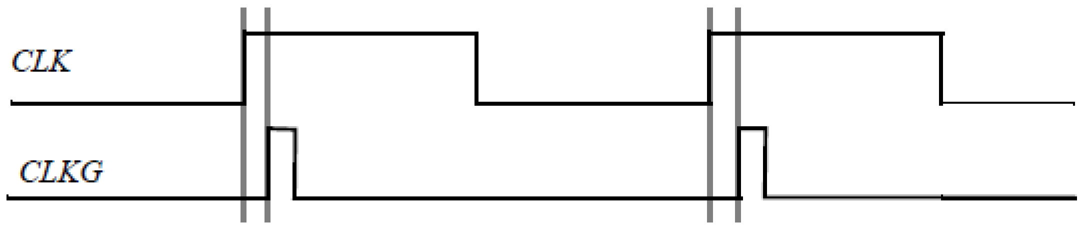


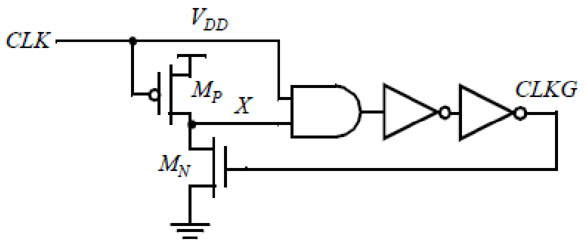
Figure 7. TSPC

- When CLK = 0, the input inverter is sampling the inverted D input on node X.
- The second (dynamic) inverter is in the precharge mode, with M6 charging up node Y to VDD.
- The third inverter is in the hold mode, since M8 and M9 are off. Therefore, during the low phase of the clock, the input to the final (static) inverter is holding its previous value and the output Q is stable.
- On the rising edge of the clock, the dynamic inverter M4-M6 evaluates. If X is high on the rising edge, node Y discharges.
- The third inverter M7-M8 is on during the high phase, and the node value on Y is passed to the output Q.
- On the positive phase of the clock, note that node X transitions to a low if the D input transitions to a high level.
- Therefore, the input must be kept stable till the value on node X before the rising edge of the clock propagates to Y. T
- his represents the hold time of the register (note that the hold time less than 1 inverter delay since it takes 1 delay for the input to affect node X).
- The propagation delay of the register is essentially three inverters since the value on node X must propagate to the output Q.
- Finally, the set-up time is the time for node X to be valid, which is one inverter delay.

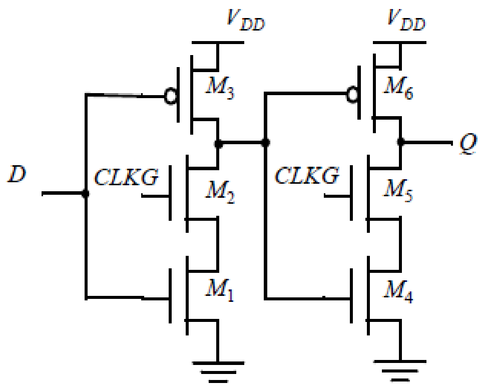
Pulse Registers

- Until now, we have used the *master-slave* configuration to create an *edge-triggered* register.
- A fundamentally different approach for constructing a register uses pulse signals. The idea is to construct a short pulse around the rising (or falling) edge of the clock.
- This pulse acts as the clock input to a latch (e.g., a TSPC flavor is shown in Figure), sampling the input only in a short window.
- Race conditions are thus avoided by keeping the opening time (i.e, the *transparent p*eriod) of the latch very short. The combination of the glitch generation circuitry and the latch results in a *positive edge-triggered* register



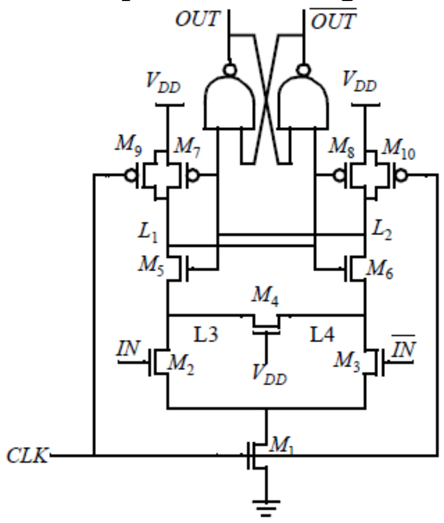


Above circuit shows an example circuit for constructing a short intentional glitch on each rising edge of the clock . When *CLK* = 0, node *X* is charged up to *VDD* (*MN* is *off* since *CLKG* is low). On the rising edge of the clock, there is a short period of time when both inputs of the AND gate are high, causing *CLKG* to go high. This in turn activates *MN*, pulling *X* and eventually *CLKG* low as shown in graph. The length of the pulse is controlled by the delay of the AND gate and the two inverters.



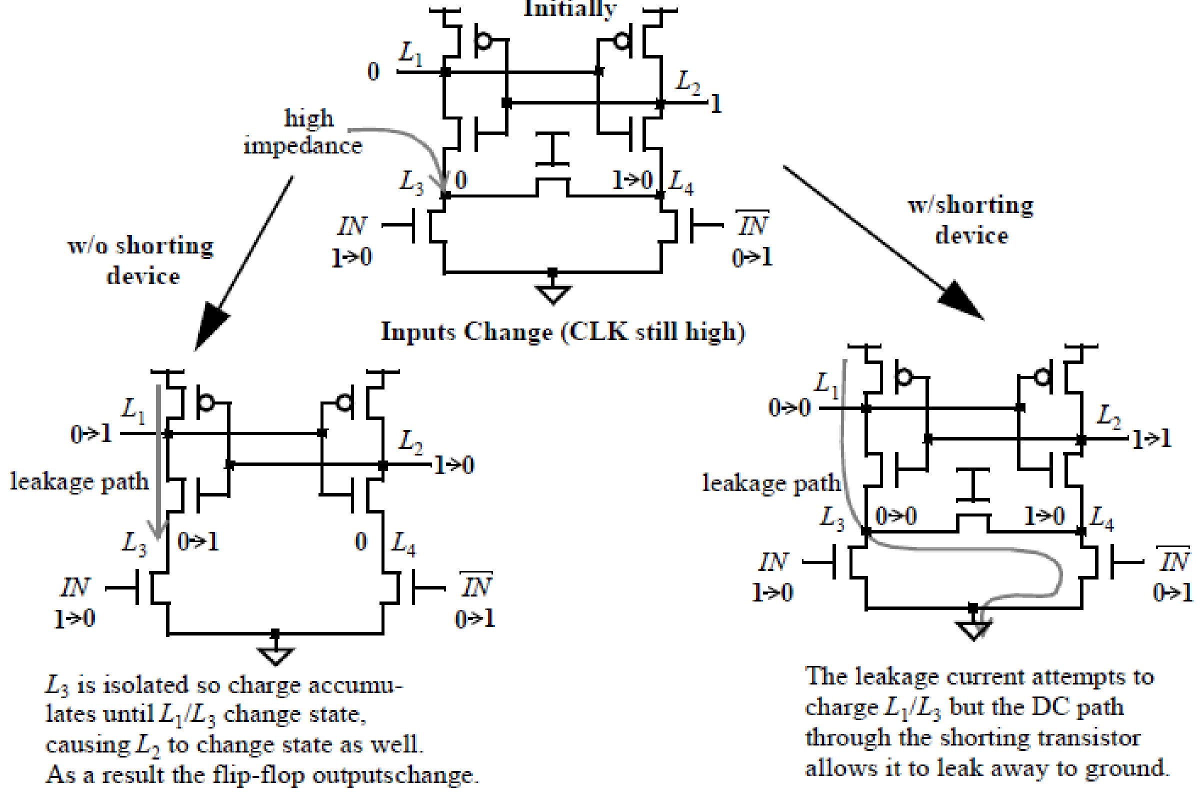
(a) register

Sense-Amplifier Based Registers



So far, we have presented two fundamental approaches towards building *edge-triggered* registers: the *master-slave* concept and the glitch technique. Figure introduces another technique that uses a *sense amplifier* structure to implement an *edge-triggered* register [Montanaro96]. *Sense amplifier* circuits accept small input signals and amplify them to generate rail-to-rail swings. As we will see,*sense amplifier* circuits are used extensively in memory cores and in low

swing bus drivers to amplify small voltage swings present in heavily loaded wires. There are many techniques to construct these amplifiers, with the use of feedback (e.g., cross-coupled inverters) being one common approach. The circuit shown in Figure uses a precharged front-end amplifier that samples the differential input signal on the rising edge of the clock signal. The outputs of front-end are fed into a NAND cross-coupled SR FF that holds the data and gurantees that the differential outputs switch only once per clock cycle. The differential inputs in this implementation don't have to have rail-to-rail swing and hence this register can be used as a receiver for a reduced swing differential bus.
The core of the front-end consists of a cross-coupled inverter (M5-M8) whose outputs (L1 and L2) are precharged using devices M9 and M10 during the low phase of the clock. As a result, PMOS transistors M7 and M8 to be turned off and the NAND FF is holding its previous state. Transistor M1 is similar to an evaluate switch in dynamic circuits and is turned off ensuring that the differential inputs don't affect the output during the low phase of the clock. On the rising edge of the clock, the evaluate transistor turn os n and the differential input pair (M2 and M3) is enabled, and the difference between the input signals is amplified on the output nodes on L1 and L2. The cross-coupled inverter pair flips to one of its the stable states based on the value of the inputs. For example, ifI N is 1, L1 is pulled to 0, and L2 remains at VDD. Due to the amplifying properties of the input stage, it is not necessary for the input to swing all the way up toVDD and enables the use of low swing signaling on the input wires.

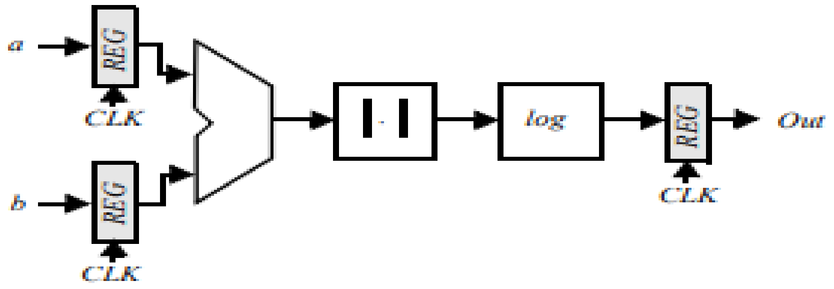


Initially
L_1
0
L_2
1
high impedance
L_3
0
1→0
L_4
IN
1→0
$\overline{IN}$
0→1
w/o shorting device
w/shorting device
Inputs Change (CLK still high)
L_1
0→1
L_2
1→0
leakage path
L_3
0→1
0
L_4
IN
1→0
$\overline{IN}$
0→1
L_3 is isolated so charge accumulates until L_1/L_3 change state, causing L_2 to change state as well. As a result the flip-flop outputschange.
L_1
0→0
L_2
1→1
leakage path
L_3
0→0
1→0
L_4
IN
1→0
$\overline{IN}$
0→1
The leakage current attempts to charge L_1/L_3 but the DC path through the shorting transistor allows it to leak away to ground.

The shorting transistor, *M*4, is used to provide a DC leakage path from either node *L*3, or *L*4, to ground. This is necessary to accommodate the case where the inputs change their value after the positive edge of *CLK* has occurred, resulting in either *L*3 or *L*4 being left in a high-impedance state with a logical low voltage level stored on the node. Without the leakage path that node would be susceptible to charging by leakage currents. The latch could then actually change state prior to the next rising edge of *CLK*

Pipelining: An approach to optimize sequential circuits

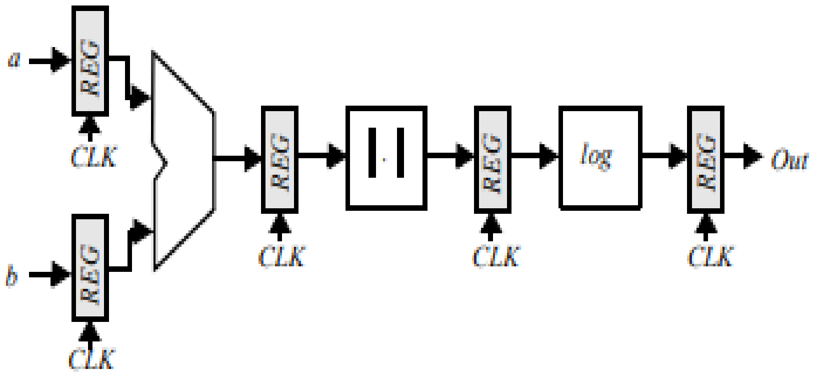
- Pipelining is a technique to improve the resource utilization, and increase the functional throughput.
- compute log(|a − b|)



- the combinatorial network, consists of the adder, absolute value, and logarithm functions.
- The minimal clock period Tmin necessary to ensure correct evaluation is given as:

$$T_{min} = t_{c\text{-}q} + t_{pd,logic} + t_{su}$$

- where tc-q and tsu are the propagation delay and the set-up time of the register, respectively.
- We assume that the registers are edge-triggered D registers.
- The term tpd,logic stands for the worst-case delay path through the combinatorial network
- Assume that each logic module has an equal propagation delay.
- We note that each logic module is then active for only 1/3 of the clock period (if the delay of the register is ignored).
- For example, the adder unit is active during the first third of the period and remains idle—this is, it does no useful computation— during the other 2/3 of the period.



Clock Period	Adder	Absolute Value	Logarithm
1	$a_1 + b_1$		
2	$a_2 + b_2$	$\|a_1 + b_1\|$	
3	$a_3 + b_3$	$\|a_2 + b_2\|$	$\log(\|a_1 + b_1\|)$
4	$a_4 + b_4$	$\|a_3 + b_3\|$	$\log(\|a_2 + b_2\|)$
5	$a_5 + b_5$	$\|a_4 + b_4\|$	$\log(\|a_3 + b_3\|)$

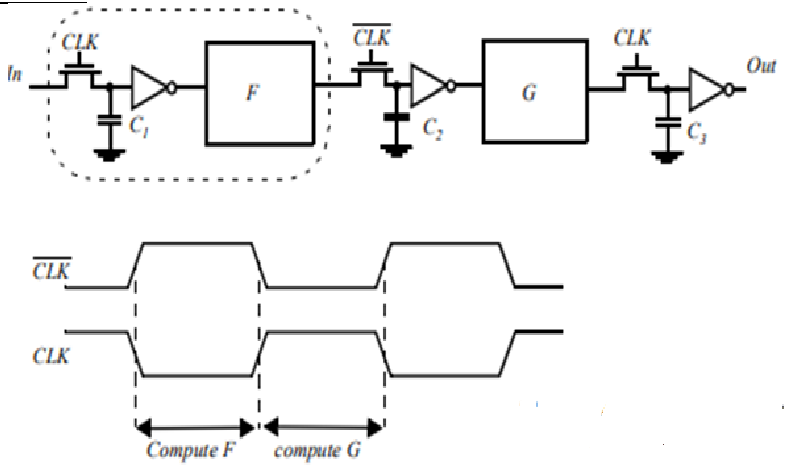
Advantage of pipelined operation

- The minimum clock period of the modified circuit.
- The combinational circuit block has been partitioned into three sections, each of which has a smaller propagation delay than the original function.
- This effectively reduces the value of the minimum allowable clock period:

$$T_{min,pipe} = t_{c-q} + \max(t_{pd,add}, t_{pd,abs}, t_{pd,log})$$

-
- Suppose that all logic blocks have approximately the same propagation delay, and that the register overhead is small with respect to the logic delays.
- The pipelined network outperforms the original circuit by a factor of three under these assumptions, or T min,pipe= Tmin/3.
- The increased performance comes at the relatively small cost of two additional registers, and an increased latency.
- Latency is defined here as the number of clock cycles it takes for the data to propagate from the input to the output.

<u>Latch -Based Pipelines</u>



Pipelined circuits can be constructed using level-sensitive latches instead of edge-triggered registers.
The pipeline system is implemented based on pass-transistor-based positive and negative latches instead of edge triggered registers.

- **The dynamic-logic rule: Inputs to a dynamic *CLKn (CLKp) block are only allowed to***
- make a single 0 → 1 (1→ 0) transition during the evaluation period (Chapter 6).

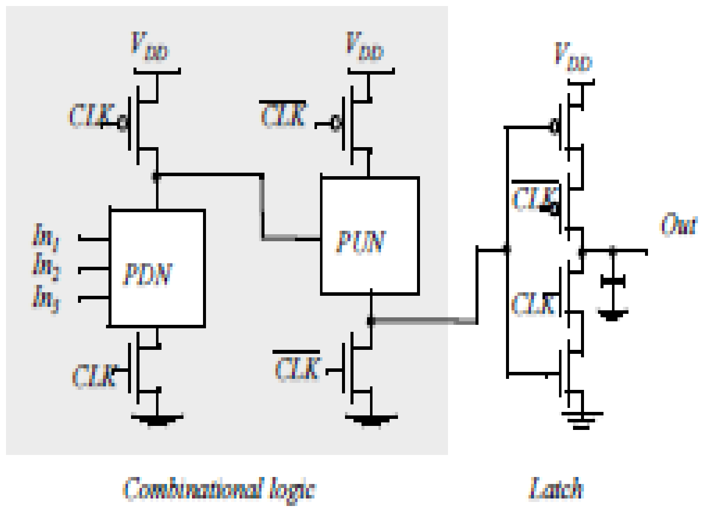
- **The C2MOS rule: In order to avoid races, the number of static inversions between** C2MOS latches should be even.
- Revised C2MOS rule:
- The number of static inversions between C2MOS latches should be even (in the absence of dynamic nodes); if dynamic nodes are present, the number of static inverters between a latch and a dynamic gate in the logic block should be even. The number of static inversions between the last dynamic gate in a logic block and the latch should be even as well.

NORA-CMOS—A Logic Style for Pipelined Structures

- Each module consists of a block of combinational logic that can be a mixture of static and dynamic logic, followed by a C2MOS latch.
- Logic and latch are clocked in such a way that both are simultaneously in either evaluation, or hold (precharge) mode.
- A block that is in evaluation during *CLK = 1 is called a CLK-module, while the inverse is called a CLKbar-module.*

	CLK block		$\overline{CLK}$ block	
	Logic	Latch	Logic	Latch
CLK = 0	Precharge	Hold	Evaluate	Evaluate
CLK = 1	Evaluate	Evaluate	Precharge	Hold

-
- A NORA data path consists of a chain of alternating *CLK and CLK-bar modules.*
- *While* one class of modules is precharging with its output latch in hold mode, preserving the previous output value, the other class is evaluating.
- Data is passed in a pipelined fashion from module to module.



(a) CLK-module

- **The dynamic-logic rule: Inputs to a dynamic *CLKn (CLKp) block are only allowed to*** make a single 0 → 1 (1→ 0) transition during the evaluation period (Chapter 6).
- **The C2MOS rule: In order to avoid races, the number of static inversions between** C2MOS latches should be even.

- Revised C2MOS rule: The number of static inversions between C2MOS latches should be even (in the absence of dynamic nodes); if dynamic nodes are present, the number of static inverters between a latch and a dynamic gate in the logic block should be even. The number of static inversions between the last dynamic gate in a logic block and the latch should be even as well.

Non-Bistable Sequential Circuits

The most important property of such a circuit is that it has two stable states, and is hence called *bistable*. The *bistable* element is not the only sequential circuit of interest. Other regenerative circuits can be catalogued as *astable* and *monostable*. The former act as oscillators and can, for instance, be used for on-chip clock generation. The latter serve as pulse generators, also called *one-shot circuits*. Another interesting regenerative circuit is the Schmitt trigger. This component has the useful property of showing hysteresis in its dc characteristic—s tis switching threshold is variable and depends upon the direction of the transition (low-to-high or high-to-low). This peculiar feature can come in handy in noisy environments.

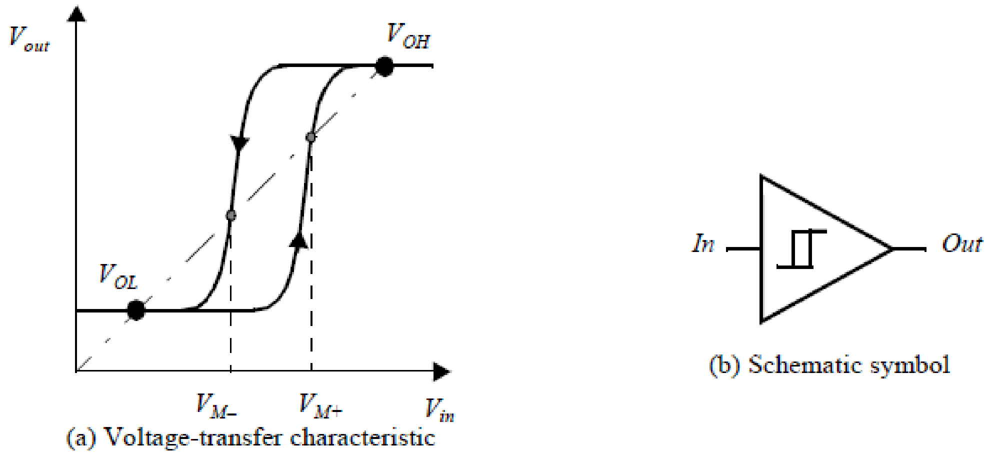
The Schmitt Trigger

Definition

A Schmitt trigger [Schmitt38] is a device with two important properties:

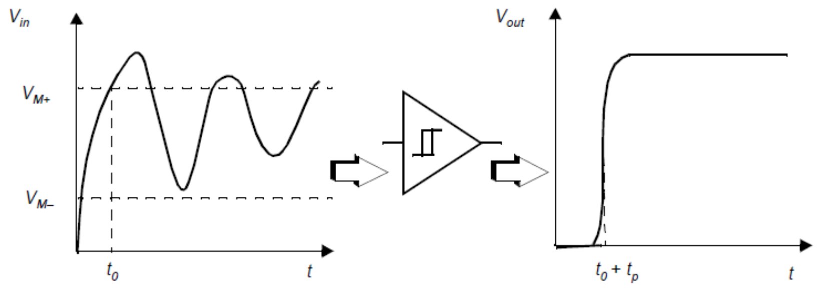
1. It responds to a slowly changing input waveform with a *fast transition time at the output*.

2. The voltage-transfer characteristic of the device displays *different switching thresholds for positive- and negative-going input signals* . This is demonstrated in Figure , where a typical voltage-transfer characteristic of the Schmitt trigger is shown (and its schematics symbol). The switching thresholds for thel ow-to-high and highto-low transitions are called *VM+* and *VM-*, respectively. The *hysteresis voltage* is defined as the difference between the two.



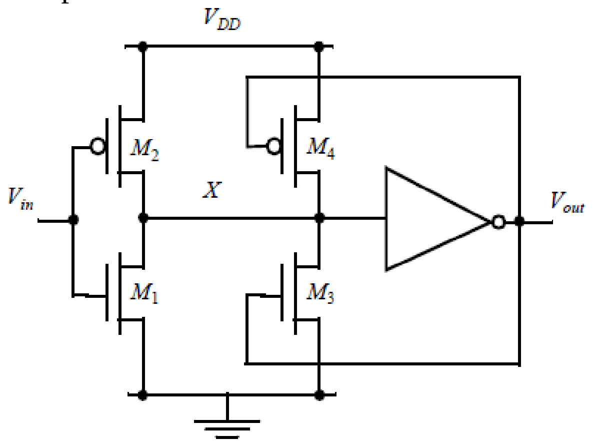
(a) Voltage-transfer characteristic (b) Schematic symbol

One of the main uses of the Schmitt trigger is to turn a noisy or slowly varying input signal into a clean digital output signal. This is illustrated in Figure . Notice how the hysteresis suppresses the ringing on the signal. At the same time, the fast low-to-high (and high-to-low) transitions of the output signal should be observed. For instance, steep signal slopes are beneficial in reducing power consumption by suppressing direct-path currents. The “secret” behind the Schmitt trigger concept is the use of positive feedback.



CMOS Implementation

One possible CMOS implementation of the Schmitt trigger is shown in Figure. The idea behind this circuit is that the switching threshold of a CMOS inverter is determined by the (*kn/kp*) ratio between the NMOS and PMOS transistors. Increasing the ratio results in a reduction of the threshold, while decreasing it results in an increase in*V M*. Adapting the ratio depending upon the direction of the transition results in a shift in the switching threshold and a hysteresis effect. This adaptation is achieved with the aid of feedback.



Suppose that *Vin* is initially equal to 0, so that *Vout* = 0 as well. The feedback loop biases the PMOS transistor *M*4 in the conductive mode while *M*3 is off. The input signal effectively connects to an inverter consisting of two PMOS transistors in parallel*M* (2 and *M*4) as a pull-up network, and a single NMOS transistor (*M*1) in the pull-down chain. This modifies the effective transistor ratio of the inverter to *kM*1/(*kM*2+*kM*4), which moves the switching threshold upwards.

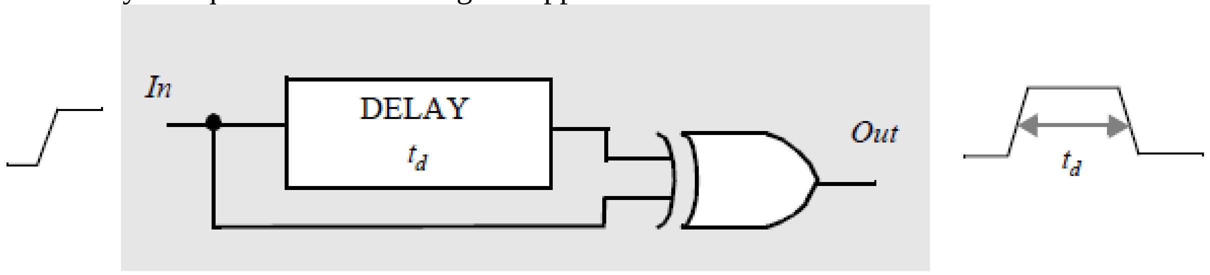
Once the inverter switches, the feedback loop turns off *M*4, and the NMOS device *M*3 is activated. This extra pull-down device speeds up the transition and produces a clean output signal with steep slopes.

A similar behavior can be observed for the high-to-low transition. In this case, the pull-down network originally consists of *M*1 and *M*3 in parallel, while the pull-up network is formed by *M*2. This reduces the value of the switching threshold to *VM*–.

Monostable Sequential Circuits

A monostable element is a circuit that generates *a pulse of a predetermined width* every time the quiescent circuit is triggered by a pulse or transition event. It is called *m onostable* because it has only one stable state (the quiescent one). A trigger event, which is either a signal transition or a pulse, causes the circuit to go temporarily into another quasi-stable state. This means that it eventually returns to its original state after a time period determined by the circuit parameters.

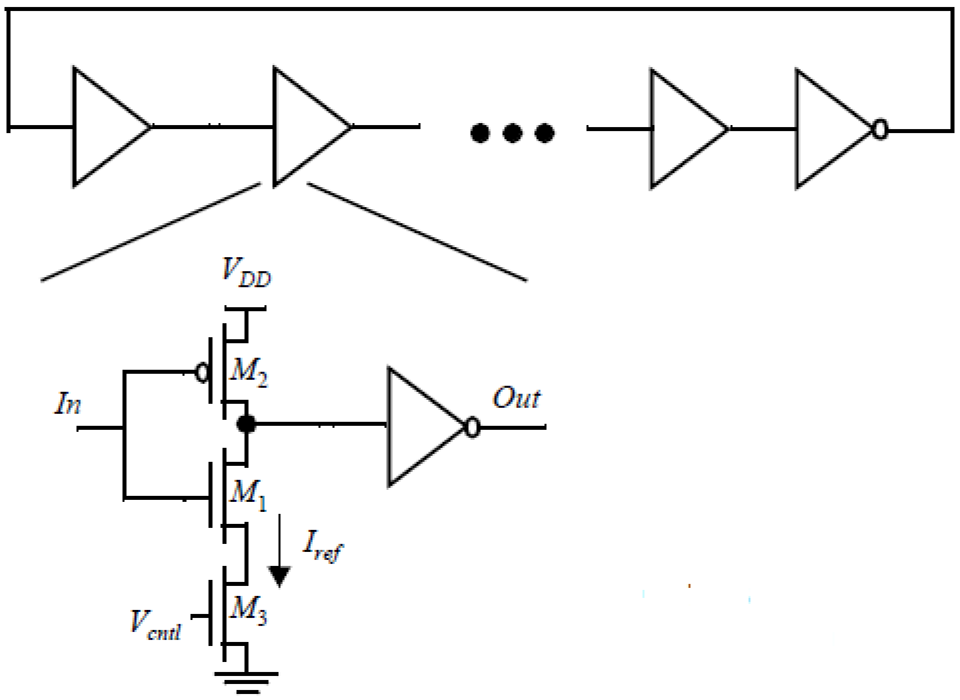
This circuit, also called a*o ne-shot*, is useful in generating pulses of a known length. This functionality is required in a wide range of applications.



We have already seen the use of a one-shot in the construction of glitch registers. Another notorious example is the *address transition detection* (ATD) circuit, used for the timing generation in static memories. This circuit detects a change in a signal, or group of signals, such as the address or data bus, and produces a pulse to initialize the subsequent circuitry. The most common approach to the implementation of one-shots is the use of a simple
delay element to control the duration of the pulse. The concept is illustrated in Figure . In the quiescent state, both inputs to the XOR are identical, and the output is low. A transition on the input causes the XOR inputs to differ temporarily and the output to go high. After a delay *td* (of the delay element), this disruption is removed, and the output goes low again. A pulse of length *td* is created. The delay circuit can be realized in many different ways, such as an *RC*-network or a chain of basic gates.

Astable Circuits

An astable circuit has no stable states. The output oscillates back and forth between two quasi-stable states with a period determined by the circuit topology and parameters (delay, power supply, etc.). One of the main applications of oscillators is the on-chip generation of clock signals. This application is discussed in detail in a later chapter (on timing).



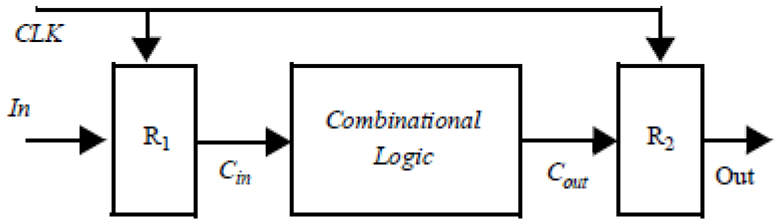
The ring oscillator is a simple, example of an astable circuit. It consists of an odd number of inverters connected in a circular chain. Due to the odd number of inversions, no stable operation point exists, and the circuit oscillates with a period equal to 2´ *tp* ´ *N*, with *N* the number of inverters in the chain and *tp* the *propagation delay* of each inverter.
The ring oscillator composed of cascaded inverters produces a waveform with a fixed oscillating frequency determined by the delay of an inverter in the CMOS process.
In many applications, it is necessary to control the frequency of the oscillator. An example of such a circuit is the *voltage-controlled oscillator (VCO)*, whose oscillation frequency is a function (typically non-linear) of a control voltage. The standard ring oscillator can be modified into a *VCO* by replacing the standard inverter with a *current-starved* inverter as shown in Figure [Jeong87]. The mechanism for controlling the delay of each inverter
is to limit the current available to discharge the load capacitance of the gate.

TIMING CLASSIFICATION OF DIGITAL SYSTEMS

In digital systems, signals can be classified depending on how they are related to a local clock [Messerschmitt90] [Dally98]. Signals that transition only at predetermined periods in time can be classified as synchronous, mesochronous, or plesiochronous with respect to a system clock. A signal that can transition at arbitrary times is considered asynchronous.

Synchronous Interconnect

A synchronous signal is one that has the exact same frequency, and a known fixed phase offset with respect to the local clock. In such a timing methodology, the signal is "synchronized" with the clock, and the data can be sampled directly without any uncertainty. In digital logic design, synchronous systems are the most straightforward type of interconnect, where the flow of data in a circuit proceeds in lockstep with the system clock as shown below.

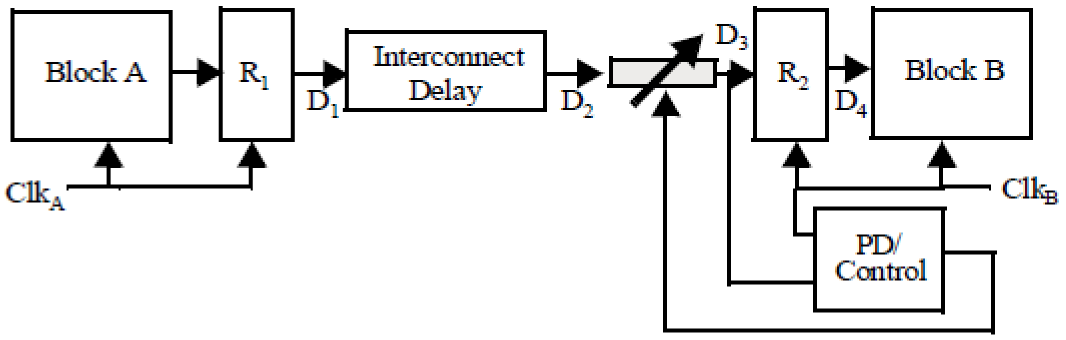


Here, the input data signal In is sampled with register R1 to give signal Cin, which is synchronous with the system clock and then passed along to the combinational logic block. After a suitable setting period, the output Cout becomes valid and can be sampled by R2 which synchronizes the output with the clock. In a sense, the "certainty period" of signal Cout, or the period where data is valid is synchronized with the system clock, which allows register R2 to sample the data with complete confidence. The length of the "uncertainty period," or the period where data is not valid, places an upper bound on how fast a synchronous interconnect system can be clocked.

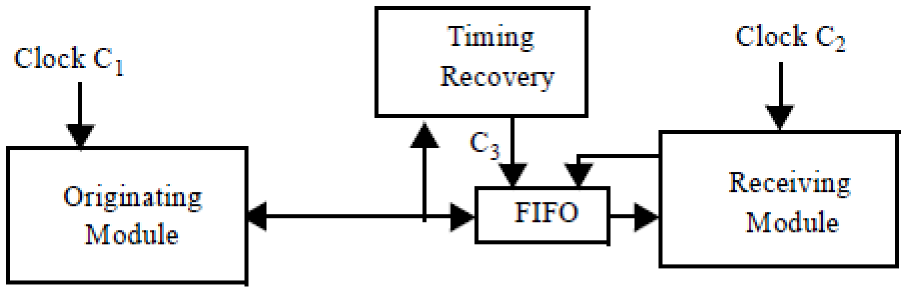
Mesochronous interconnect

A mesochronous signal is one that has the same frequency but an unknown phase offset with respect to the local clock ("meso" from Greek is middle). For example, if data is being passed between two different clock domains, then the data signal transmitted from the first module can have an unknown phase relationship to the clock of the receiving module. In such a system, it is not possible to directly sample the output at the receiving module because of the uncertainty in the phase offset. A (mesochronous) synchronizer can be used to synchronize the data signal with

the receiving clock as shown below. The synchronizer serves to adjust the phase of the received signal to ensure proper sampling.
In Figure, signal D1 is synchronous with respect to ClkA. However, D1 and D2 are mesochronous with ClkB because of the unknown phase difference between ClkA and ClkB and the unknown interconnect delay in the path between Block A and Block B. The role of the synchronizer is to adjust the variable delay line such that the data signal D3 (a delayed version of D2) is aligned properly with the system clock of block B. In this example, the variable delay element is adjusted by measuring the phase difference between the received signal and the local clock. After register R2 samples the incoming data during the certainty period, then signal D4 becomes synchronous with ClkB.



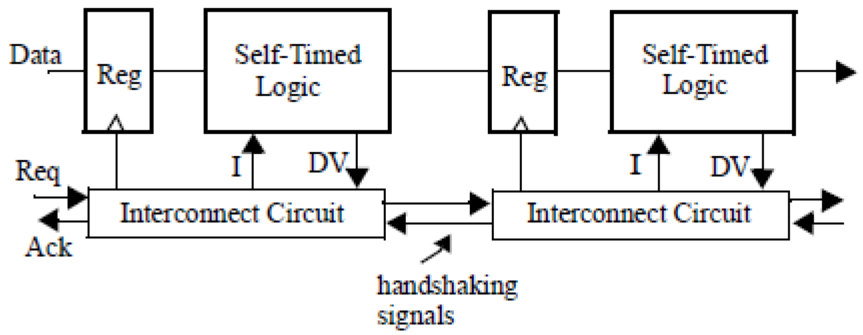
Plesiochronous Interconnect
A plesiochronous signal is one that has nominally the same, but slightly different frequency as the local clock (“plesio” from Greek is near). In effect, the phase difference drifts in time. This scenario can easily arise when two interacting modules have independent clocks generated from separate crystal oscillators. Since the transmitted signal can arrive at the receiving module at a different rate than the local clock, one needs to utilize a buffering scheme to ensure all data is received. Typically, plesiochronous interconnect only occurs in distributed systems like long distance communications, since chip or even board level circuits typically utilize a common oscillator to derive local clocks. A possible framework for plesiochronous interconnect is shown in Figure



In this digital communications framework, the originating module issues data at some unknown rate characterized by *C*1, which is plesiochronous with respect to *C*2. The timing recovery unit is responsible for deriving clock *C*3 from the data sequence, and buffering the data in a FIFO. As a result, *C*3 will be synchronous with the data at the input of the FIFO and will be mesochronous with *C*1. Since the clock frequencies from the originating and receiving modules are mismatched, data might have to be dropped if the transmit frequency is faster, and data can be duplicated if the transmit frequency is slower than the receive frequency. However, by making the FIFO large enough, and periodically resetting the system whenever an overflow condition occurs, robust communication can be achieved.

Asynchronous Interconnect

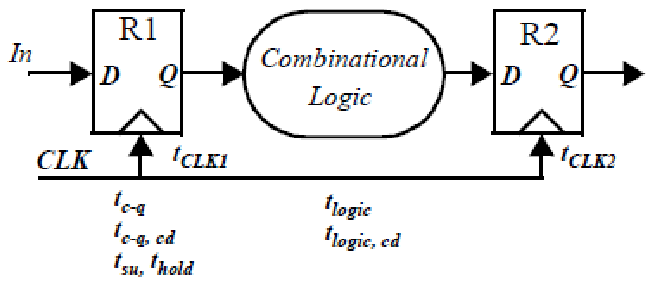
Asynchronous signals can transition at any arbitrary time, and are not slaved to any local clock. As a result, it is not straightforward to map these arbitrary transitions into a synchronized data stream. Although it is possible to synchronize asynchronous signals by detecting events and introducing latencies into a data stream synchronized to a local clock, a more natural way to handle asynchronous signals is to simply eliminate the use of local clocks and utilize a self-timed asynchronous design approach. In such an approach, communication between modules is controlled through a handshaking protocol to perform the proper ordering of commands



When a logic block completes an operation, it will generate a completion signal *DV* to indicate that output data is valid. The handshaking signals then initiate a data transfer to the next block, which latches in the new data and begins a new computation by asserting the initialization signal *I*. Asynchronous designs are advantageous because computations are performed at the native speed of the logic, where block computations occur whenever data becomes available. There is no need to manage clock *skew*, and the design methodology leads to a very modular approach where interaction between blocks simply occur through a handshaking procedure. However, these handshaking protocols result in increased complexity and overhead in communication that can reduce performance.

SYNCHRONOUS DESIGN— AN IN-DEPTH PERSPECTIVE

i. Synchronous Timing Basics



Virtually all systems designed today use a periodic *synchronization* signal or clock. The generation and distribution of a clock has a significant impact on performance and power dissipation. For a positive *edge-triggered* system, the rising edge of the clock is used to denote the beginning and completion of a clock cycle. In the ideal world, assuming the clock paths from a central distribution point to each register are perfectly balanced, the phase of the clock (i.e., the position of the clock edge relative to a reference) at various points in the system is going to be exactly equal. However, the clock is neither perfectly periodic nor perfectly simultaneous. This results in performance degradation and/or circuit malfunction. Figure shows the basic structure of a synchronous pipelined datapath. In the ideal scenario, the clock at registers 1 and 2 have the same clock period and transition at the exact same time. The following timing parameters characterize the timing of the sequential circuit.

• The contamination (minimum) delay *tc-q,cd*, and maximum propagation delay of the register *tc-q*.

• The set-up (*tsu*) and hold time (*thold*) for the registers.

• The contamination delay *tlogic,cd* and maximum delay *tlogic* of the combinational logic.

tclk1 and *tclk2*, corresponding to the position of the rising edge of the clock relative to a global reference.

Under ideal conditions (*tclk1* = *tclk2*), the worst case propagation delays determine the minimum clock period required for this sequential circuit. The period must be long enough for the data to propagate through the registers and logic and be set-up at the destination register before the next rising edge of the clock. This constraint is given by

$$T > t_{c-q} + t_{logic} + t_{su}$$

At the same time, the hold time of the destination register must be shorter than the minimum propagation delay through the logic network,

$$t_{hold} < t_{c-q,\,cd} + t_{logic,\,cd}$$

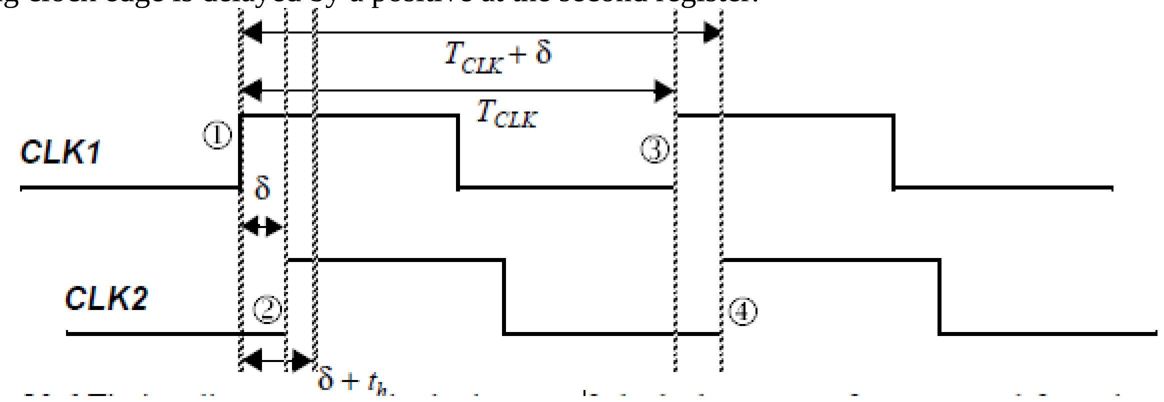
The above analysis is simplistic since the clock is never ideal. As a result of process and environmental variations, the clock signal can have *spatial* and *temporal* variations.

ii. Clock Skew

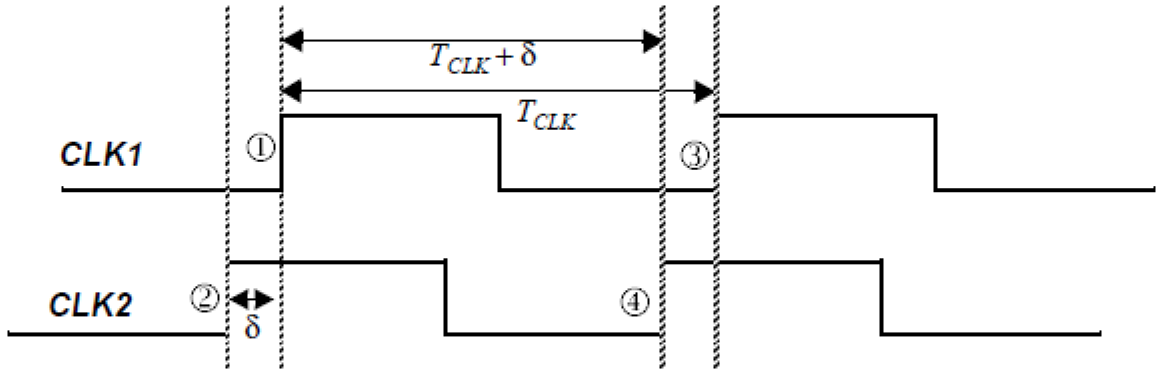
The spatial variation in arrival time of a clock transition on an integrated circuit is commonly referred to as *clock skew*. The *clock skew* between two points *i* and *j* on a IC is given by

$$\delta(i,j) = t_i - t_j,$$

where *ti* and *tj* are the position of the rising edge of the clock with respect to a reference. Consider the transfer of data between registers *R*1 and *R2* in Figure . The clock skew can be positive or negative depending upon the routing direction and position of the clock source. The timing diagram for the case with positive skew is shown in Figure. As the figure illustrates, the rising clock edge is delayed by a positive at the second register.



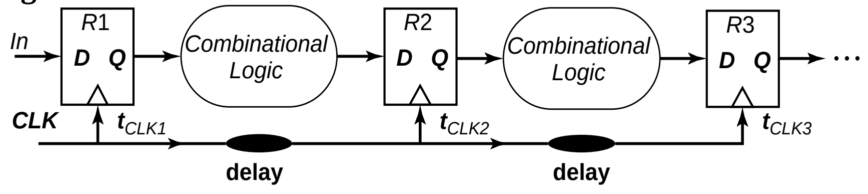
Clock skew is caused by static path-length mismatches in the clock load and by definition skew is constant from cycle to cycle. That is, if in one cycle *CLK2* lagged *CLK1* by δ, then on the next cycle it will lag it by the same amount. It is important to note that clock skew does not result in clock period variation, but rather phase shift.



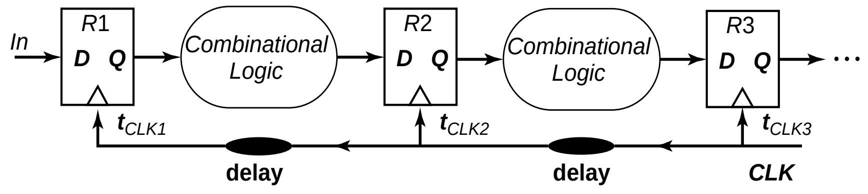
The constraint on the minimum clock period can then be derived as:

$$T + \delta \geq t_{c-q} + t_{logic} + t_{su} \quad \text{or} \quad T \geq t_{c-q} + t_{logic} + t_{su} - \delta$$

Positive and Negative Skew



(a) Positive skew



(b) Negative skew

δ > **0**—This corresponds to a clock routed in the same direction as the flow of the data through the pipeline (Figure). In this case, the skew has to be strictly controlled and satisfy

$$\delta + t_{hold} < t_{(c-q,\,cd)} + t_{(\text{logic},\,cd)}$$

or

$$\delta < t_{(c-q,\,cd)} + t_{(\text{logic},\,cd)} - t_{hold}$$

If this constraint is not met, the circuit does malfunction **independent of the clock period**. Reducing the clock frequency of an edge-triggered circuit does not help get around skew problems! On the other hand, positive skew increases the throughput of the circuit as expressed by Eq.

$$T + \delta \geq t_{c-q} + t_{logic} + t_{su} \quad \text{or} \quad T \geq t_{c-q} + t_{logic} + t_{su} - \delta$$

, because the clock period can be shortened by δ. The extent of this improvement is limited as large values of δ soon provoke violations of Eq..

• δ < **0**—When the clock is routed in the opposite direction of the data (Figure 10.8b), the skew is negative and condition

$$\delta + t_{hold} < t_{(c-q,\,cd)} + t_{(logic,\,cd)}$$

or

$$\delta < t_{(c-q,\,cd)} + t_{(logic,\,cd)} - t_{hold}$$

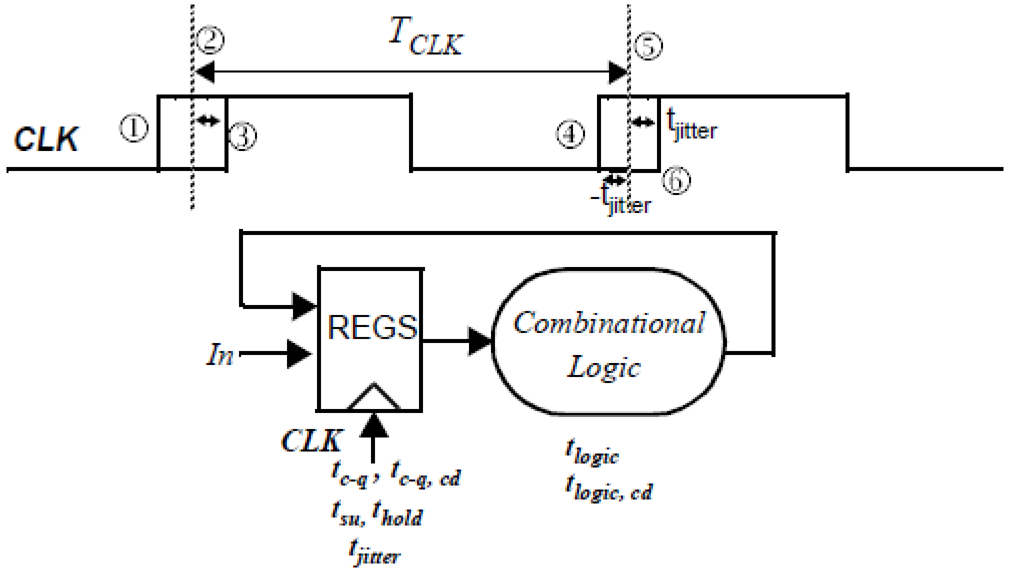
is unconditionally met. The circuit operates correctly independent of the skew. The skew reduces the time available for actual computation so that the clock period has to be increased by |δ|. In summary, routing the clock in the opposite direction of the data avoids disasters but hampers the circuit performance .

iii. Clock Jitter

Clock jitter refers to the temporal variation of the clock period at a given point — that is, the clock period can reduce or expand on a cycle-by-cycle basis. It is strictly a temporal uncertainty measure and is often specified at a given point on the chip. Jitter can be measured and cited in one of many ways. *Cycle-to-cycle jitter* refers to time varying deviation of a single clock period and for a given spatial location *i* is given as *Tjitter,i*(*n*) = *Ti, n+1* - *Ti,n*
- *TCLK*, where *Ti,n* is the clock period for period *n*, *Ti, n+1* is clock period for period *n*+1, and *TCLK* is the nominal clock period.

Jitter directly impacts the performance of a sequential system. Figure shows the nominal clock period as well as variation in period. Ideally the clock period starts at edge 2and ends at edge 5 and with a nominal clock period of *TCLK*. However, as a result of *jitter*, the worst case scenario happens when the leading edge of the current clock period is delayed (edge 3), and the leading edge of the next clock period occurs early (edge 4). As a result, the total time available to complete the operation is reduced by 2 *tjiiter* in the worst case and is given by

$$T_{CLK} - 2t_{jitter} \geq t_{c-q} + t_{logic} + t_{su} \quad \text{or} \quad T \geq t_{c-q} + t_{logic} + t_{su} + 2t_{jitter}$$



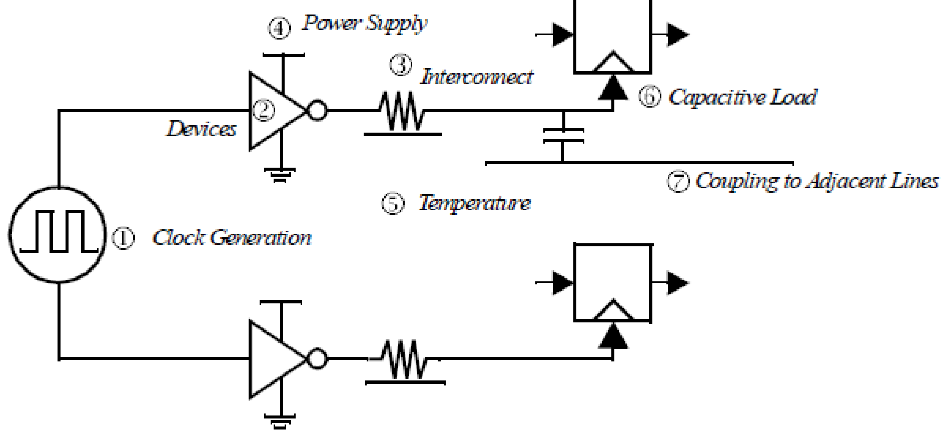
iv. Sources of Skew and Jitter

Clock-Signal Generation (1)

The generation of the clock signal itself causes *jitter.* A typical on-chip clock generator, as described at the end of this chapter, takes a low-frequency reference clock signal, and produces a high-frequency global reference for the processor. The core of such a generator is a Voltage-Controlled Oscillator (VCO). This is an analog circuit, sensitive to intrinsic device noise and power supply variations. A major problem is the coupling from the surrounding noisy digital circuitry through the substrate. This is particularly a problem in modern fabrication processes that combine a lightly-doped epitaxial layer and a heavily doped substrate (to combat latch-up). This causes substrate noise to travel over large distances on the chip. These noise source cause temporal variations of the clock signal that propagate unfiltered through the clock drivers to the flip-flops, and result in *cycle-to-cycle* clock-period variations. This *jitter* causes performance degradation.

Manufacturing Device Variations (2)

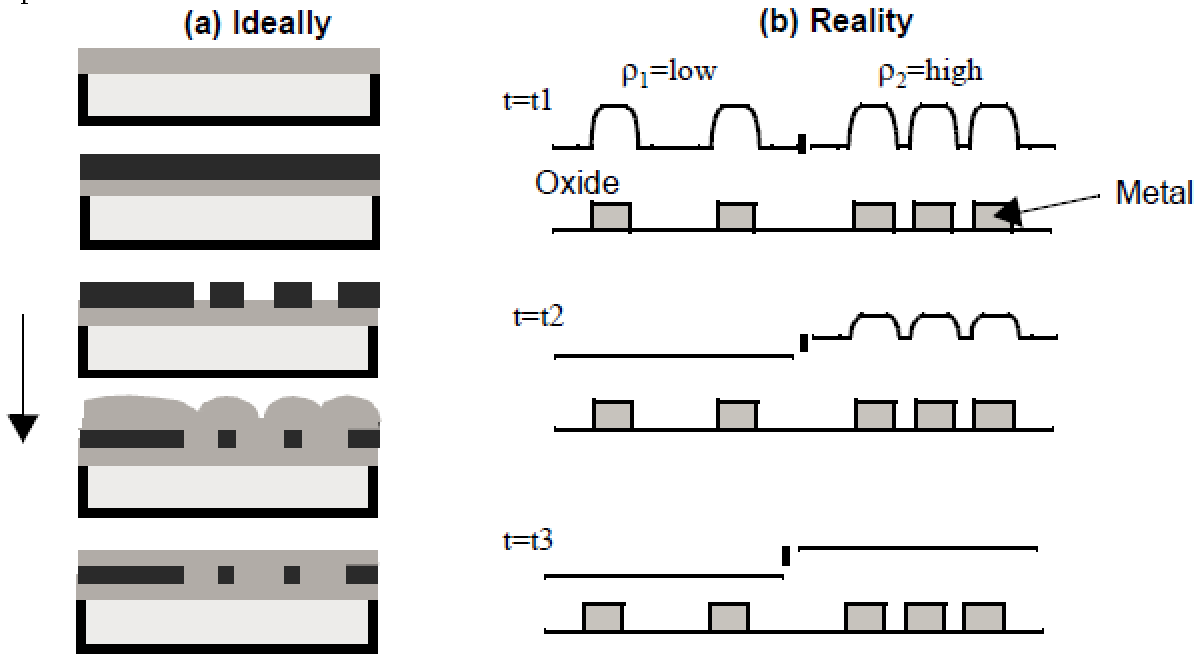
Distributed buffers are integral components of the clock distribution networks, as they are required to drive both the register loads as well as the global and local interconnects. The matching of devices in the buffers along multiple clock paths is critical to minimizing timing uncertainty. Unfortunately, as a result of process variations, devices parameters in the buffers vary along different paths, resulting in static *skew*. There are many sources of variations including oxide variations (that affects the gain and threshold), dopant variations, and lateral dimension (width and length) variations. The doping variations can affect the depth of junction and dopant profiles and cause variations in electrical parameters such as device threshold and parasitic capacitances. The orientation of polysilicon can also have a big impact on the device parameters. Keeping the orientation the same across the chip for the clock drivers is critical. Variation in the polysilicon critical dimension, is particularly important as it translates directly into MOS transistor channel length variation and resulting variations in the drive current and switching characteristics. Spatial variation usually consists of wafer level (or within-wafer) variation and die-level (or within-die) variation. At least part of the variation is systematic and can be modeled and compensated for. The random variations however, ultimately limits the matching and *skew* that can be achieved



Interconnect Variations (3)

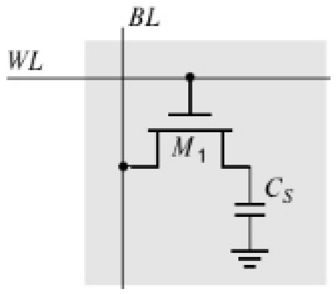
Vertical and lateral dimension variations cause the interconnect capacitance and resistance to vary across a chip. Since this variation is static, it causes *skew* between different paths. One important source of interconnect variation is the *Inter-level Dielectric (ILD)* thickness variations.

In the formation of aluminum interconnect, layers of silicon dioxide are interposed between layers of patterned metallization. The oxide layer is deposited over a layer of patterned metal features, generally resulting in some remaining step height or surface topography. *Chemical-mechanical polishing* (CMP) is used to "planarize" the surface and remove topography resulting from deposition and etch (as shown in Figure).While at the feature scale (over an individual metal line), CMP can achieve excellent planarity, there are limitations on the planarization that can be achieved over a global range. This is primarily caused due to variations in polish rate that is a function of the circuit layout density and pattern effects. Figure shows this effect where the polish rate is higher for the lower spatial density region, resulting in smaller dielectric thickness and higher capacitance. The assessment and control of variation is of critical importance in semiconductor process development and manufacturing. Significant advances have been made to develop analytical models for estimating the ILD thickness variations based on spatial density.
Since this component is often predictable from the layout, it is possible to actually correct for the systematic component at design time (e.g., by adding appropriate delays or making the density uniform by adding "dummy fills"). Figure shows the spatial pattern density and ILD thickness for a high performance microprocessor. The graphs show that there is clear correlation between the density and the thickness of the dielectric. So clock distribution networks must exploit such information to reduce clock *skew*.
Other interconnect variations include deviation in the width of the wires and line spacing. This results from photolithography and etch dependencies. At the lower levels of metallization, lithographic effects are important while at higher levels etch effects are important that depend on width and layout. The width is a critical parameter as it directly impacts the resistance of the line and the wire spacing affects the wire-to-wire capacitance, which the dominant component of capacitance.



PART A

1. **Draw a one transistor dynamic RAM cell?   A/M 19**



2. **Define clock skew?   A/M 19,A/M 18**

   Clock skew is a phenomenon in synchronous digital circuit systems in which the same **clock** signal arrives at different components at different times. The instantaneous difference between the arrival times of any two **clocks** is called their **skew**.

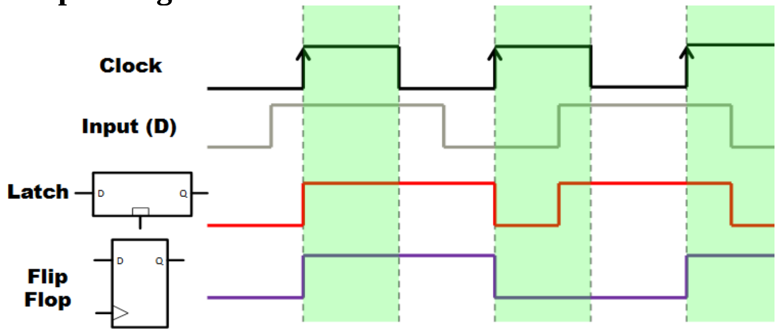
3. **List out the advantages and limitations of 3T DRAM over 1 T DRAM?N/D 18**

   For a 3T DRAM , there are no constraints on device ratios and Reads are non-destructive. On the other hand, the read-out of the 1T DRAM cell is destructive; read and refresh operations are necessary for correct operation. 1T cell requires presence of an extra capacitance that must be explicitly included in the design.

4. **List out the advantages of C2MOS logic based register over pass transistor based master slave? N/D 18**

   C2MOS logic based register is insensitive to clock overlap
   It avoids Race Conditions

5. **Compare registers and Latches?   A/M 18**



– a latch is level sensitive
– a register (or FF) is edge-triggered

6. **What is NORA CMOS?   N/D 17**

   A race free dynamic CMOS technique for pipelined logic structures .

7. **Define clock jitter?   N/D 17**

   Jitter is the timing variation of a set of signal edges from their ideal values . jitters in a clock signals are typically caused by noise or other disturbances in the system.

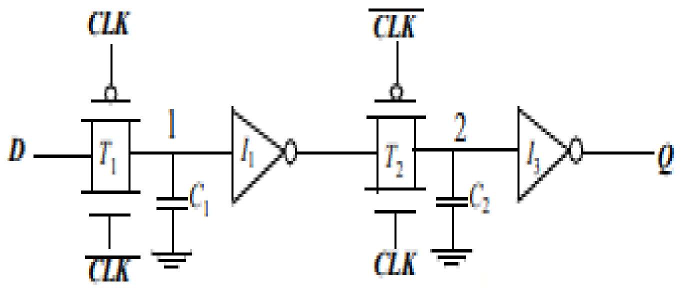
8. **What is pipelining? A/M 17 ,N/D 16**

Pipelining is an implementation technique where multiple instructions are overlapped in execution. The computer pipeline is divided into stages. Each stage completes a part of an instruction in parallel. Pipelining is a popular design technique often used to accelerate the operation of the datapaths in digital processors.

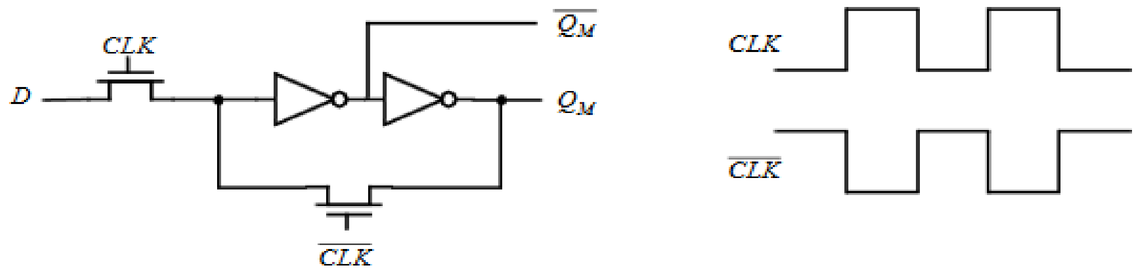
9. **Compare asynchronous design and synchronous design? A/M 17**

1 Synchronous	2 asynchronous
It is easy to design	It is difficult to design
A clocked flip flop acts as memory element.	An unclocked flip flop or time delay is used as memory elemet
They are slower	They are faster

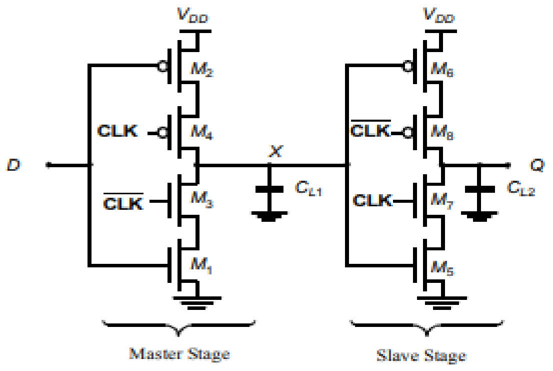
7. **Draw the schematic of Dynamic Edge Triggered Register? N/D 16**



8. **Draw the switch level schematic of multiplexer based nMOS latch using nMOS only pass transistor for multiplexers? A/M 16**



9. **What is clocked CMOS register? A/M 16**



1. CLK =0(CLK = 1):

The first tri-state driver is turned on, and the master stage acts as an inverter sampling the inverted version of D on the internal node X.

The master stage is in the evaluation mode. Meanwhile, the slave section is in a high-impedance mode, or in a hold mode.

Both transistors M7 and M8 are off, decoupling the output from the input. The output Q retains its previous value stored on the output capacitor CL2.

2. The roles are reversed when CLK = 1:

The master stage section is in hold mode (M3- M4 off), while the second section evaluates (M7-M8 on).

The value stored on CL1 propagates to the output node through the slave stage which acts as an inverter.

PART B

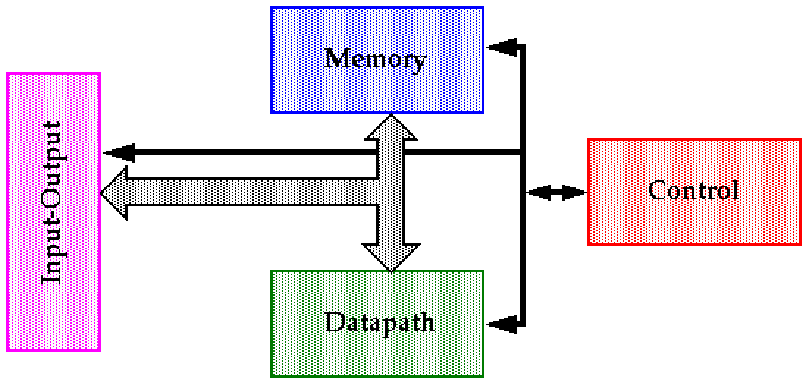
1. Design a D Latch using Transmission gate . Using which realise a two phase non overlapping master slave negative edge triggered D flip flop? **A/M 19 (OR)**
2. Explain the operation of master-slave based edge triggered register. **A/M 16 (OR)** Discuss the CMOS register concept and design master slave triggered register, explain its operation with overlapping periods? **A/M 18 (OR)** Draw and explain the operation of conventional, pulsed and resettable latches? **A/M 17**
3. Elucidate in detail low power SRAM circuit **A/M 19**
4. Explain the memory architecture and its control circuits in detail? **A/M 18**
5. Discuss about the design of sequential dynamic circuits and its pipelining concept? **N/D 17 (OR)** Discuss in detail various pipelining approaches to optimize sequential circuits. **A/M 16 (OR)** Explain the timing basics and clock distribution techniques in synchronous design in detail? **N/D 17 (OR)** Explain timing issues and pipelining? **A/M 17**
6. Explain the operation of true single phased clocked register? **A/M 17 (OR)** explains the operation of true single phased clocked register? **N/D 16**
7. Discuss on various static latches and registers? **N/D 16**
8. Explain NORA CMOS latches **N/D 16**

UNIT-4
DESIGN OF ARITHMETIC BUILDING BLOCKS AND SUBSYSTEM

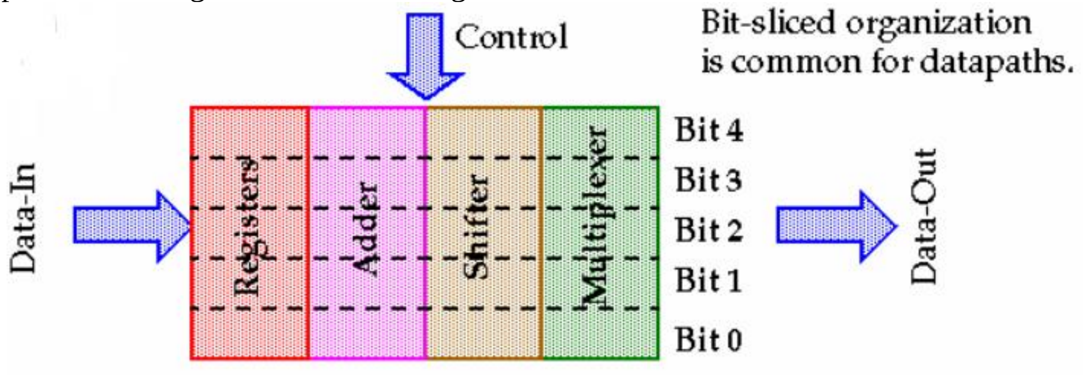
ARITHMETIC BUILDING BLOCKS:

1. DATA PATH CIRCUITS:

A digital processor consists of datapath, memory, control and I/O blocks as shown below.



- The datapath is the core of the digital processor where all the computations are performed .The other blocks are supporting units that can be used to either store the results produced by the datapath.
- A typical datapath consists of an interconnection of basic combinational functions such as arithmetic operations (addition, multiplication, shift comparison, etc) and logic operations (AND, OR, etc).
- Datapaths are arranged in a bit sliced organisation.



- Instead of operating on single bit digital signals the data in a processor are arranged in word based fashion.
- For example the microprocessor datapaths are 32 or 64 bits wide whereas the dedicated signal processing datapaths such as in DSL modems magnetic disk or CD players are 5 to 24 bits
- Since the same operation is frequently performed on each bit of the data word, the datapath of 32 bit MP consists of 32 bit slices each slice operating on a single bit, hence the term bit sliced.
- Bit slices are either identical or reassemble a similar structure for all bits.

- In general, the datapath designer can concentrate on the design of a single slice that is repeated 32 times.

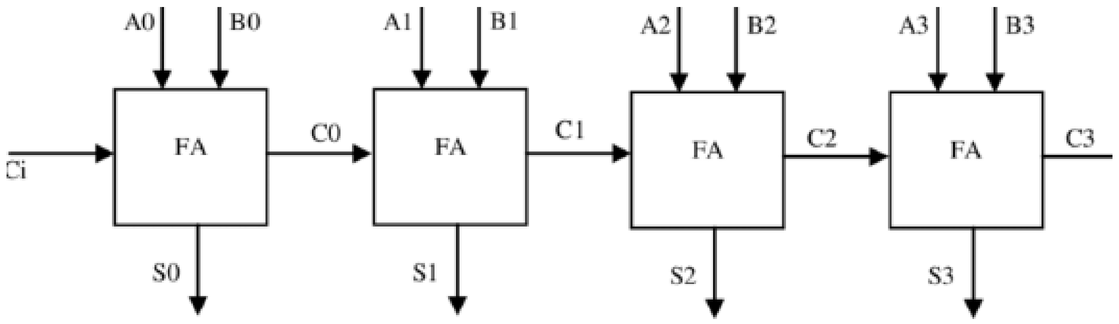
2. ADDER:

- Addition is the most commonly used arithmetic operation. Also it is a speed limiting element. Therefore careful optimization is very important. This optimization can be done in circuit or logic level.
- The following adders are optimized at circuit level.
  * Mirror adder
  * Transmission gate based adder
  * Manchester carry chain adder
  * Ripple carry adder

  The following adders are optimized in logic level
  * Carry skip/bypass
  * Carry select
  * Carry look ahead
  * Carry save

i. RIPPLE CARRY ADDER:

An N-bit adder can be constructed by N full adder (FA) in series C_0,k-1 to C_I, k for k=1 to N=1 and the first carry in the O^th block [$C_{i,}$0]is zero. This configuration is called ripple carry adder, since the carry bit ripples (or propagates) from 1st stage to the next.



- The delay through the circuit depends on the applied input signals. For some input signals there is no rippling effect, whereas in other cases, the carry has no ripple from LSB to MSB bits.
- The propagation delay of such a structure (are called the critical path) is defined as the worst case delay over all possible input patterns.
- The delay of the adder is proportional to the number of bits in the input nord N and is given by

  $t_{adder} = (N-1)\, t_{carry} + t_{sum}$

  where t_{carry} is the propagation delay od one adder from Ci to Co

  t_{sum} is the propagation delay of (sum) one adder from Ci to S.

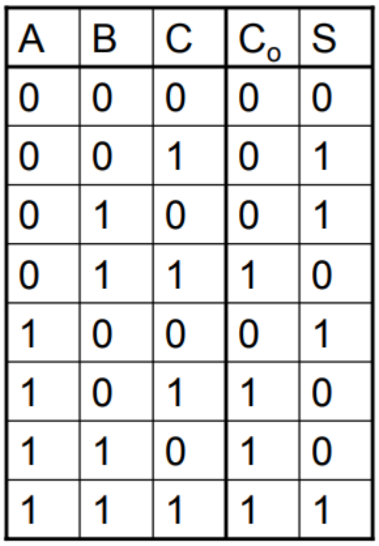
- The equation implies that:

The propagation delay of RCA is linearly proportional to N. If N increases then delay also increases which is undesirable.

- Optimisation is required in carry to reduce t $_{carry}$ .and it is multiplied by (N-1).
- Effect of t $_{carry}$ on t $_{adder}$ is more than the effect of t $_{sum}$.

Circuit Optimisation of single full adder:

In a full adder, A and B are adder inputs. Ci is the carry input, S is the sum and Co is the carry output.

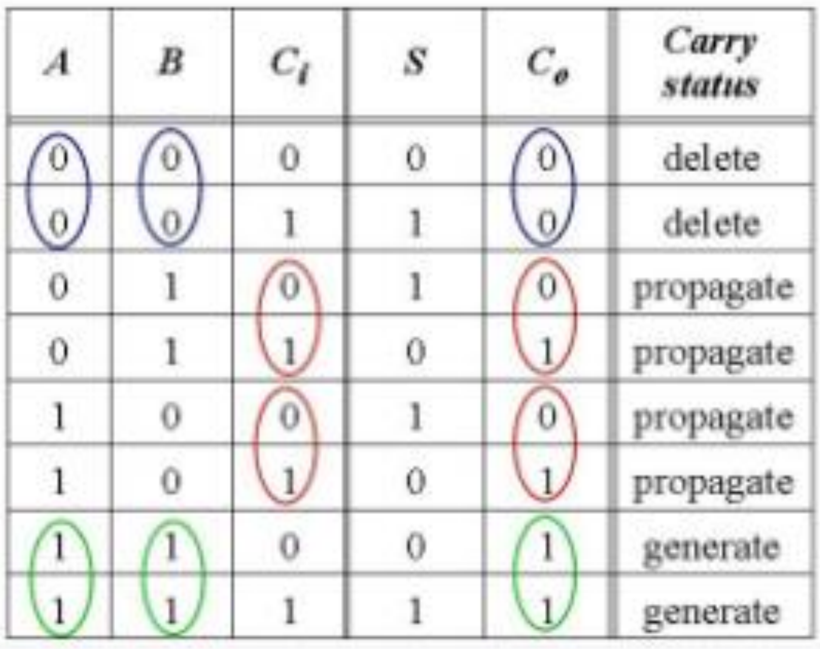


A	B	C	C_o	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

S = A’ B’ Ci + A B’ Ci’ + A’ B Ci’+ ABC

= A xor B xor C

Co = AB+ BCi+ ACi



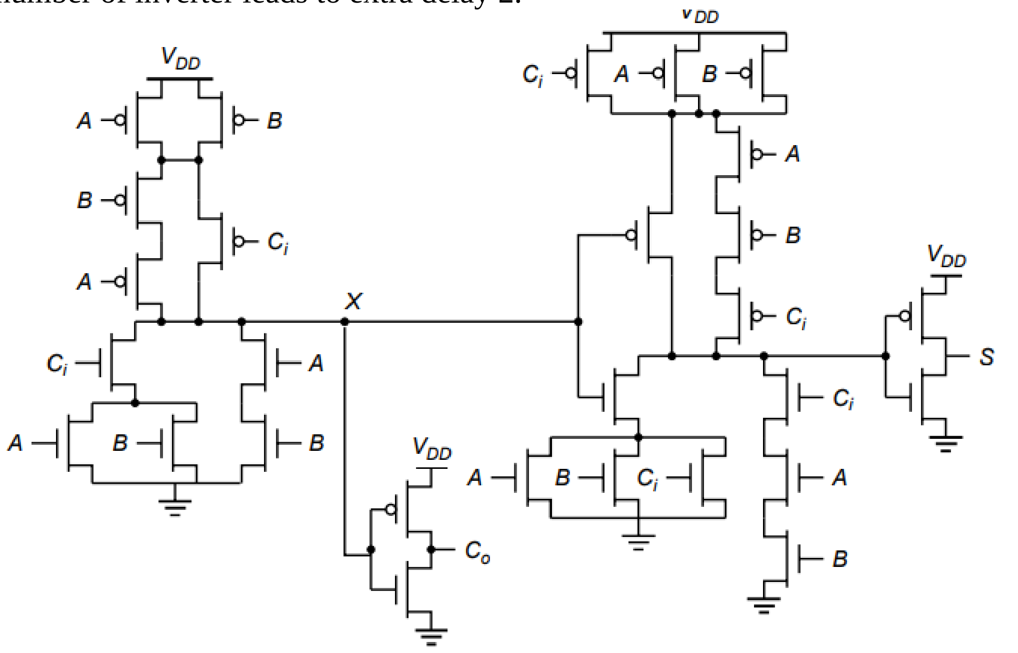
A	B	C_i	S	C_o	*Carry status*
0	0	0	0	0	delete
0	0	1	1	0	delete
0	1	0	1	0	propagate
0	1	1	0	1	propagate
1	0	0	1	0	propagate
1	0	1	0	1	propagate
1	1	0	0	1	generate
1	1	1	1	1	generate

Generate (G) = AB

Propagate (P) = $A \oplus B$

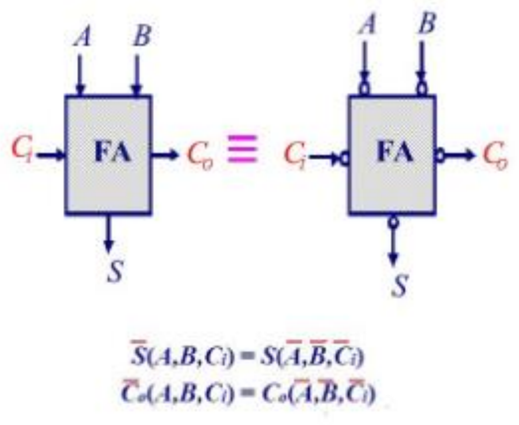
Delete = $\overline{A}\ \overline{B}$

These equations can be implemented using CMOS logic using 28 transistors. In addition to consuming a large area, this circuit has the following disadvantages.
* A large number of PMOS transistors are connected in series which leads to slow operation 1.
* Co is obtained using inverter. The N input adder the propagation of carry through N number of inverter leads to extra delay 2.

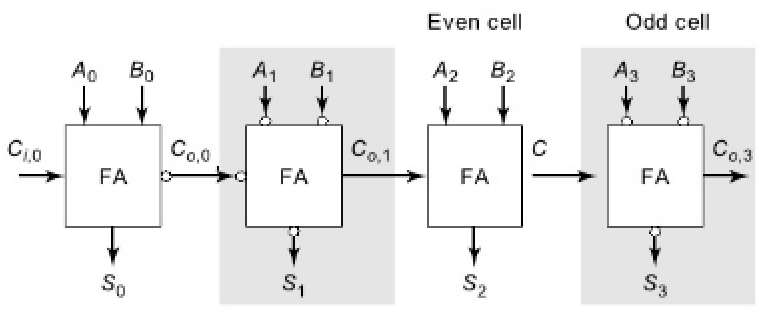


28 Transistors

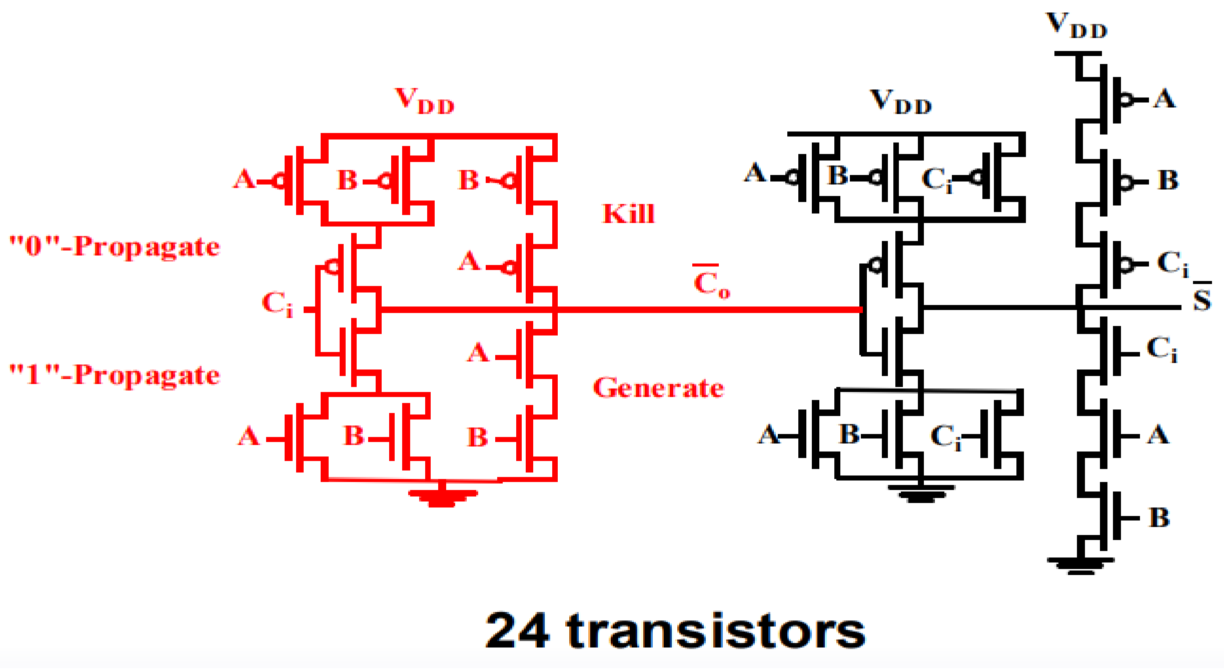
- Problem1 can be reduced by using mirror adder. This adder based on propagate, delete/kill, generate model, It is given by
  - Propagate P = A xor B or A+B then Co = Ci
  - Delete (kill) D = A’ B’ then Co = 0
  - Generate G = AB then Co = 1
- This circuit has completely symmetrical PMOS and NMOS chain
- Maximum of 2 transistors are connected in series (in carry path)
- The 2nd problem can be eliminated by using inverting property. i.e.
  “Inverting all inputs of a full adder results in inverted value at all outputs”
- This can be expressed as


A
B
C_i
FA
C_o
≡
S
$\overline{S}(A,B,C_i) = S(\overline{A},\overline{B},\overline{C_i})$
$\overline{C_o}(A,B,C_i) = C_o(\overline{A},\overline{B},\overline{C_i})$

The RCA can be constructed by using normal cell FA and inverted cell FA to eliminate inverter stage.



- The drawback of the above block is
  *No regular data path
  *Sum generation and inputs A', B_1', A_2' and B_2' requires one extra logic stage but it is not an issue, because it appears only once in circuit.



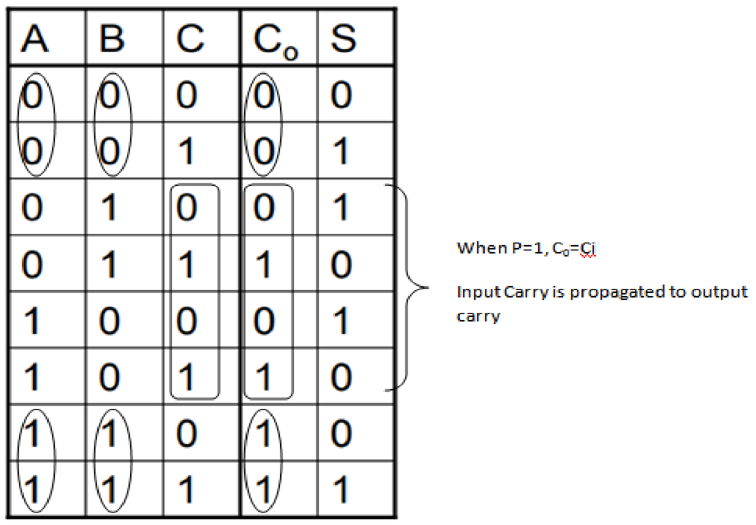
ii. TRANSMISSION GATE BASED ADDER:

- A full adder can be designed using a MUX and XOR' s. The MUX and XOR design is impractical in complementary CMOS logic.
- If the MUX and XOR are implemented byv using TG. It is useful. This is also based on propagate generate model.
- **Carry generation:**

  When P = A xor B => Co = Ci

  When P = 0 C_0=A=B

  C_o = PCi+P' A



A	B	C	C_o	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

When P=1, C_0=Ci

Input Carry is propagated to output carry

Sum generation

When P=A xor B = 1

S=Ci'

When P=0

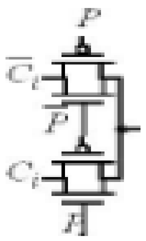
S=Ci

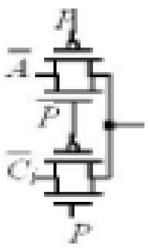
S=PCi' + P'Ci

The equation of S and C can be used to implement transmission gate based adder.

A	B	C	C_o	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

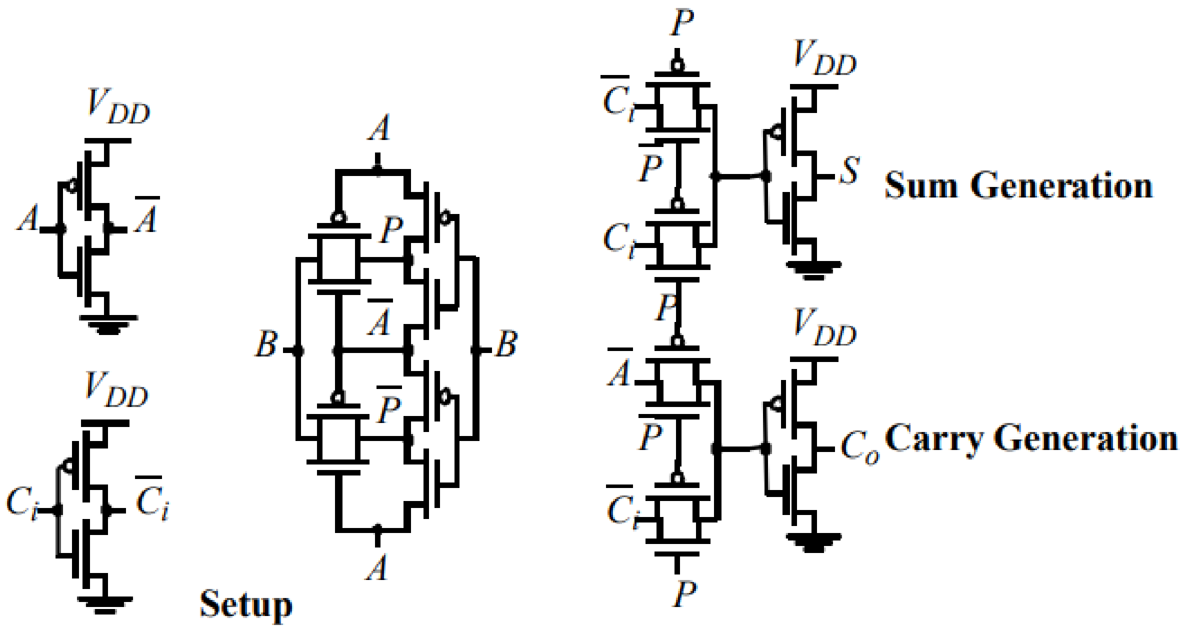
Sum generation & Carry generation:





For sum,
When P=1, P’=0 and
T1 is on.
S=Ci’
When P=0, P’=1 and
T2 is on
S=Ci

For Carry,
When P=1, P’=0 and
T1 is on
C0=Ci
When P=0, P’=1 and
T2 is on
Co=A
This design has equal delays for S and Co.



The setup signals required for the design are
Case (i)
A=0,B=1 then
T1 is ON and P=B=1
So, N1 and N2 are ON P'=A=0

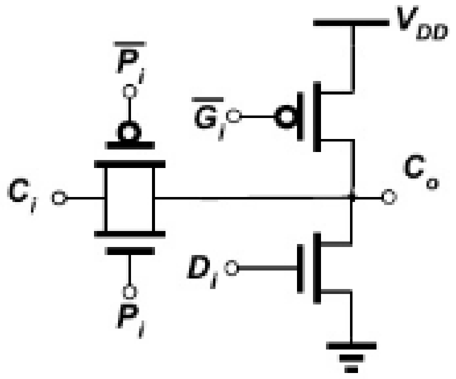
Case (ii)
A=1, B=0 then T2 is ON P'=B=0
At the same time P1 and P2 both are ON since B=0
P=A=1
The design requires totally 20 transistors (8 – S &C0 generation, 12-setup signal generation)

iii. MANCHESTER CARRY CHAIN ADDER:

- A Manchester carry chain adder uses cascade of pass transistors to implement the carry chain.
- This adder is also based on propagate/generate /delete model.

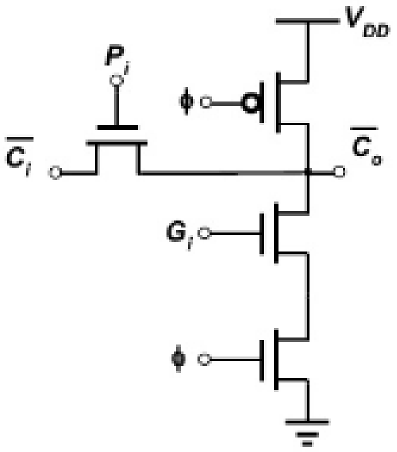
Generate (G) = AB	Then $C_0 = 1$
Propagate (P) = $A \oplus B$	$C_0 = C_i$
Delete = $\overline{A}\,\overline{B}$	$C_0 = 0$

- The Manchester carry gate using static logic is



When P=1;P'=0 then TG =ON so C_0=Ci
When G=1;G'=0 P1=ON
C0=VDD=1
When D=1;N1=ON therefore C_0=0

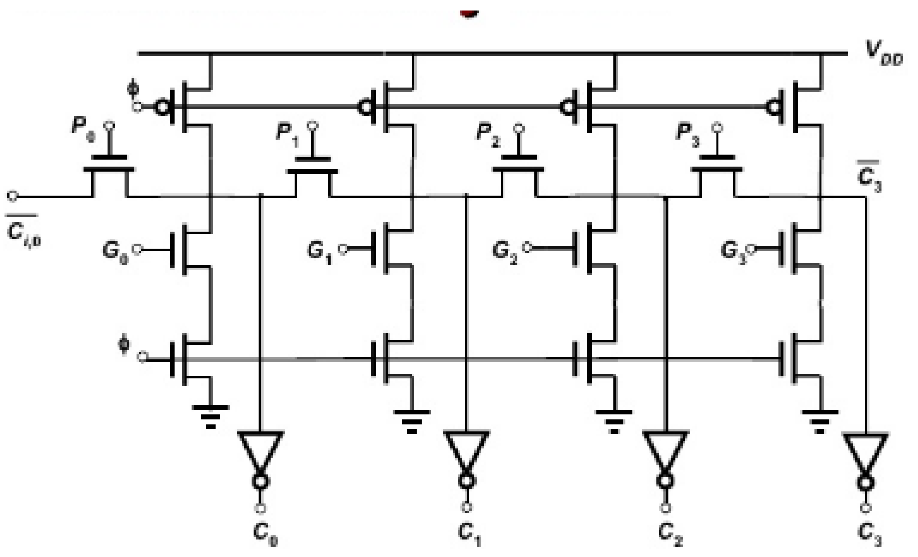
- The manachester carry gate using dynamic logic is



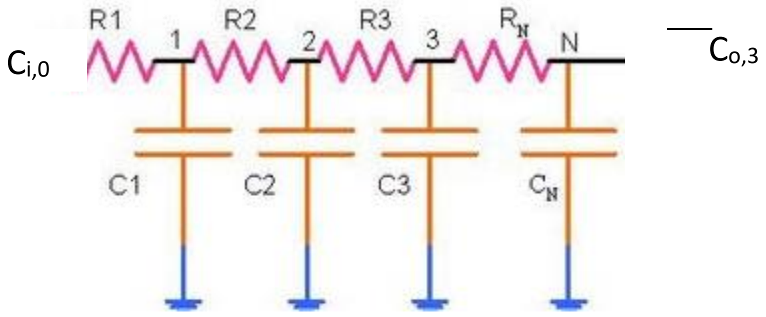
In dynamic logic there are 2 modes of operation

- Φ=0 (Pre-charge mode) eliminates the need for delete /kill signal because Carry out bar = 1 and carry out =0
- In evaluation mode if G=1 then C_0 = 1, if P=1 then C_0=C_i

- A 4 bit MCA designed using dynamic logic is given by



➢ The series of PN in the carry chain path can be modelled using RC delay model



➢ The prop delay is

$$\tau_{Di} = \sum_{k=1}^{N} C_k R_{ik}$$

= 0.69[R1Ci+(R1C1)C2+(R1+R2+R3)C3+(R1+R2+R3+R4)C4]
= 0.69[RC+2RC+3RC+4RC] if R1=R2=R3=R4;C1=C2=C3=C4
= 0.69[10RC]
For N input
tp=0.69N(N+1)RC/2

iv. <u>CARRY-LOOKAHEAD LOOKAHEAD ADDERS</u>

We know that

$$C_i = G_i + P_i C_{i-1} = \overline{K_i + P_i \overline{C}_{i-1}}$$
$$\overline{C}_i = K_i + P_i \overline{C}_{i-1} = \overline{G_i + P_i C_{i-1}}$$

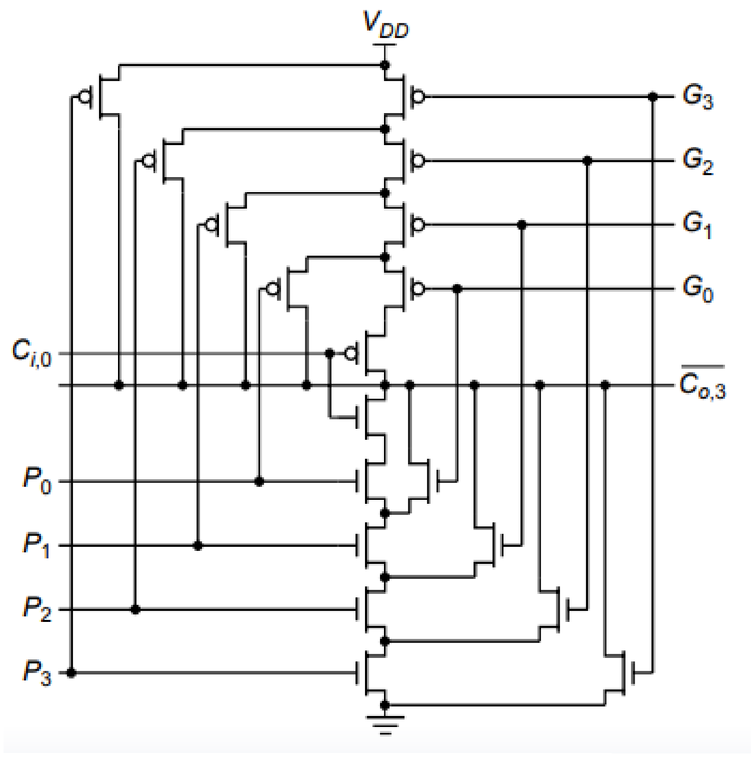
$$G_i = A_i B_i$$
$$P_i = A_i + B_i$$
$$K_i = \overline{A_i + B_i} = \overline{A_i}\,\overline{B_i}$$

Therefore for a 4 bit adder

$$C_1 = G_1 + P_1 C_0$$
$$C_2 = G_2 + P_2 (G_1 + P_1 C_0)$$
$$C_3 = G_3 + P_3 (G_2 + P_2 (G_1 + P_1 C_0))$$
$$C_4 = G_4 + P_4 (G_3 + P_3 (G_2 + P_2 (G_1 + P_1 C_0)))$$

Expanding

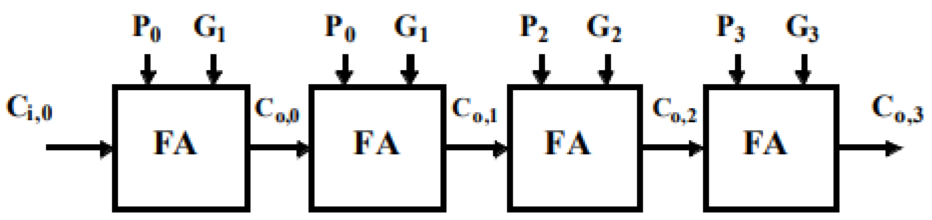
$$C_1 = G_1 + P_1 C_0$$
$$C_2 = G_2 + P_2 G_1 + P_2 P_1 C_0$$
$$C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_0$$
$$C_4 = G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 + P_4 P_3 P_2 P_1 C_0$$



$$C_{o,k} = G_k + P_k(G_{k-1} + P_{k-1}C_{o,k-2})$$
$$C_{o,k} = G_k + P_k(G_{k-1} + P_{k-1}(\ldots + P_1(G_0 + P_0 C_{i,0})))$$

v. THE CARRY BYPASS ADDER:

Consider the 4 bit adder block shown below



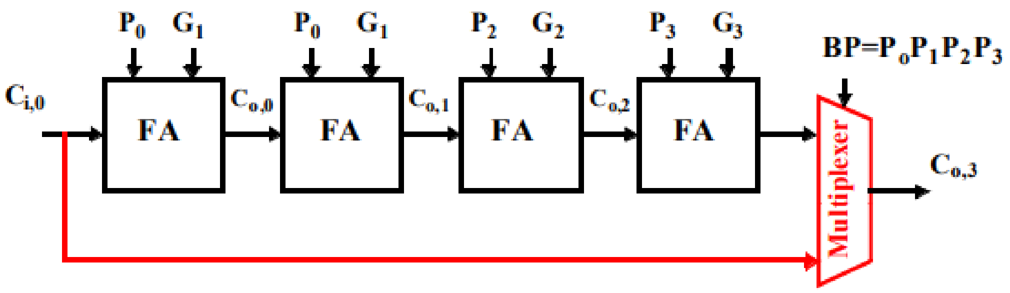
➢ Suppose the values of Ak and Bk (k=0,12,3) are such that all are prop signals Pk are high.An incoming carry Ci,0=1 propogates under those condition through the complete adder chain and causes an outgoing carry C0,3=1

If(P0P1P2P3=1) then C0,3=Ci,0.

Else either DELETE or GENERATE occurred .

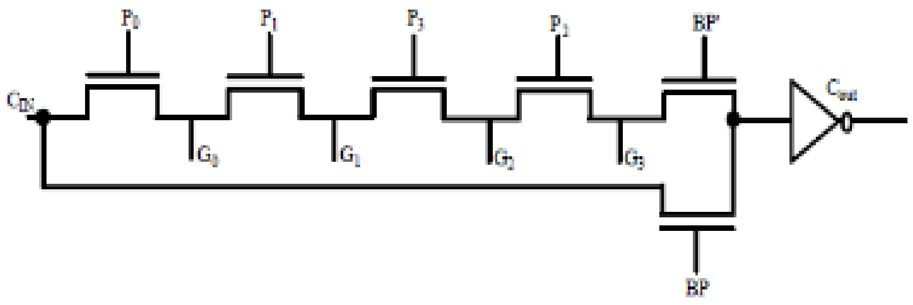
➢ When BP=p0,p1,p2,p3=1, the incoming carry is forwarded to the next block through the bypass transistor M6

If BP=0, normal carry propagation.



Idea: If (P0 and P1 and P2 and P3 = 1)
then $C_{o3} = C_0$, else “kill” or “generate”.

➢ The Manchester carry chain implementation of bypass adder is

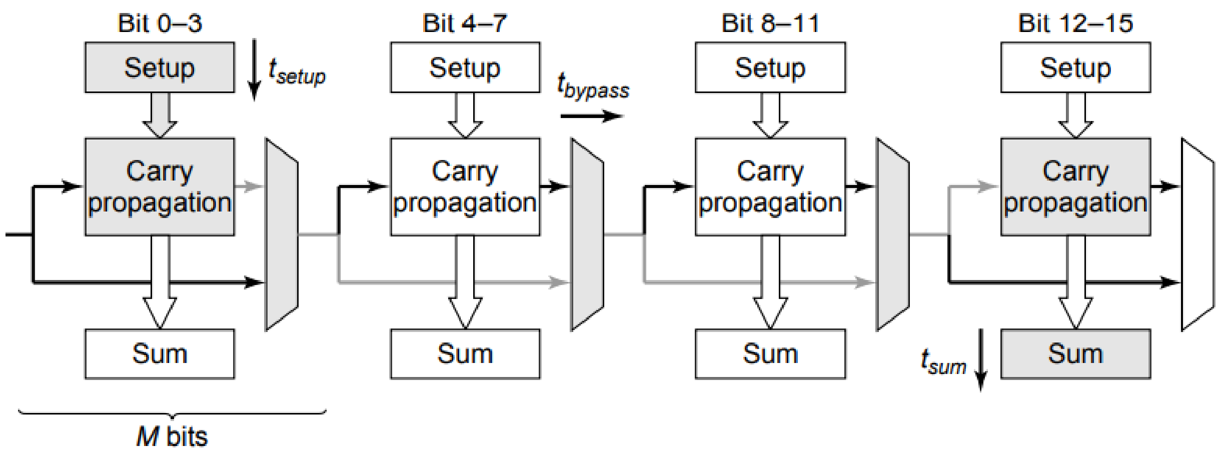


When BP=P0P1P2P3=1, then Ci,0 is connected to C0’3 through N6 and inverter.

➢ A 16 bits carry save adder can be designed by dividing 16 by equal length bypass stage. Each stage
Contains M bits (i.e. M =no of adder)

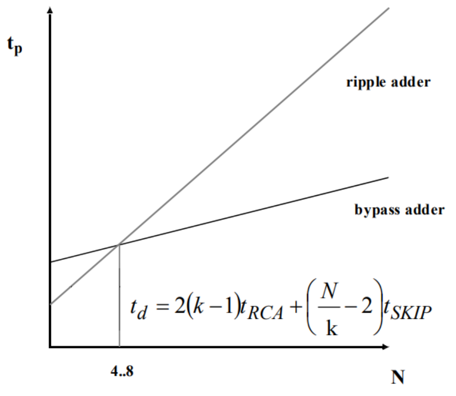
No of stages = N/M=16/4=4

Therefore, M=4



$$t_{adder} = t_{setup} + Mt_{carry} + (N/M-1)t_{bypass} + (M-1)t_{carry} + t_{sum}$$

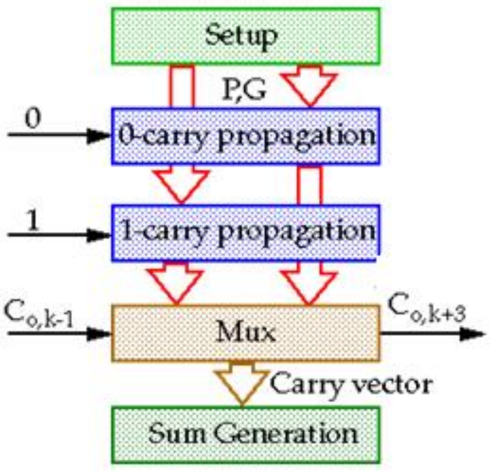
- In this, setup and sum of all stages are produced simultaneously. But the carry have to propagate through all the stages (which is called worst stage).so the propagation delay increases.
- BEST case is the carry directly connected to the output without propagating through all the stages. This occurs if all the propagate signals P0 to P15 are one.
- For example, the input carry propagates through first stage and bypass/skip [(N/M)-2] stages (2&3)
  And finally reaches at the output by propogation delay is
  $tp=t_{setup} + M\ t_{carry} + [(N/M)-2]t_{bypass} + M\ t_{carry} + t_{sum}$
  where
  t_{setup} →time required to generate P and G signals.
  T_{carry} →time required to propagate carry from input to output by one FA
  T_{bypass}→time required to bypass/skip the stages.
  T_{sum} →time required to produce sum output
  M →no of FA in each stage.
- The disadvantage of carry bypass adder is tehe delay due to bypass stage which consists of mux.eventhough the delay increases gradually than the RCA



For less than 8 bit ripple carry adder is good,but larger input bypass adder is best.

vi. CARRY SELECT ADDER:

- In this, two structure can be used to produce carry output with assumption of both possible to values of carry input simultaneously in advance. Out of these two carry outputs, the correct result is selected with mux based on incoming carry from previous stage this can be shown in below diagram.
- This block of adder adding the bits from k to k+3
- Instead of waiting on the arrival of carry C0,k-1 from previous block, this block generates output with the assumptions of 0 and 1.after arriving C0,k-1 the mux selects any one carry signal.



There are two types of CSA

1. LINEAR CARRY SELECT ADDER:

- In this CSA, all the stages having equal number of adders. Total no of adders are divided equally (N/M)to get the no. of stages. For example,16 bit LCSA is given below.
- In this 16 bit LCSA all the stages are simultaneously generating setup signal by using input bits.
- Then all the 0-carry and 1-carry of all the blocks simultaneously generating carry output with assumption of 0 and 1 respectively.
- After that, first mux select the carry from both 0- carry and 1-carry base on the carry from previous block. Then second mux starts the select carry from second stage . 0-carryand 1-carry based on the carry from 1 st stage.
- Then the worst case delay of 16 bit LCSA is

$$t_p = t_{setup} + Mt_{carry} + (N/M)\, t_{mux} + t_{sum}.$$

Where

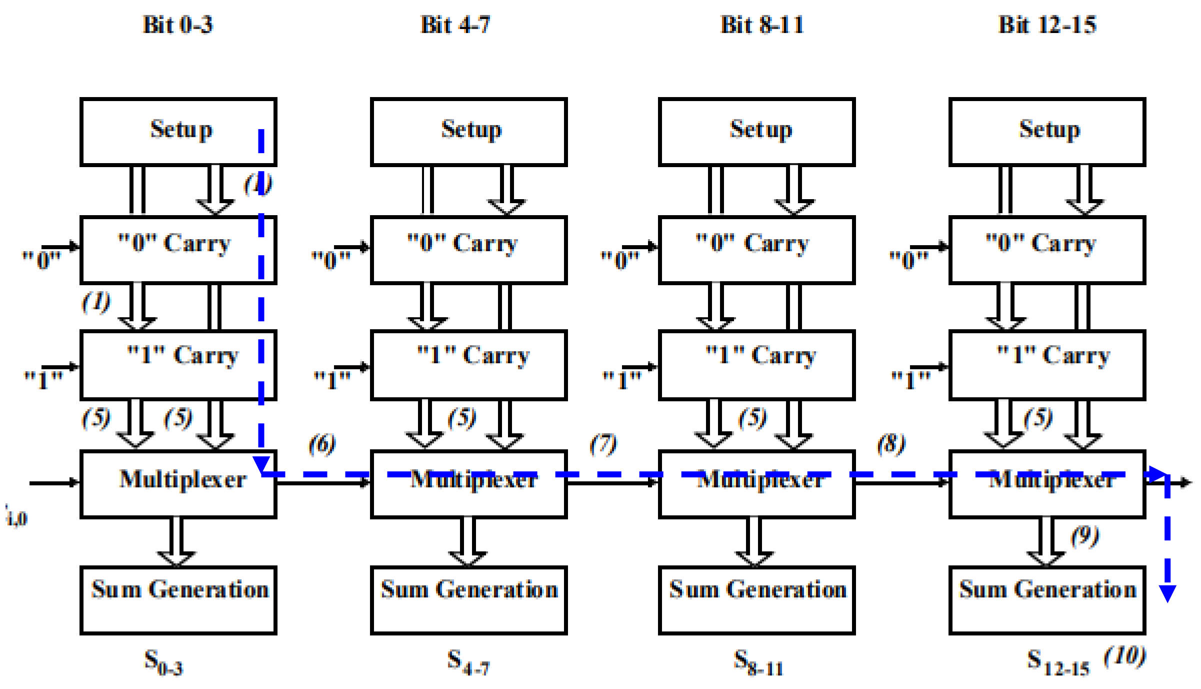
T_{setup}→tine required to generate the setup signals G and P.

t_{carry}→time required to prop from Ci,0 to C0,0through FA.

(N/M)→no of stages

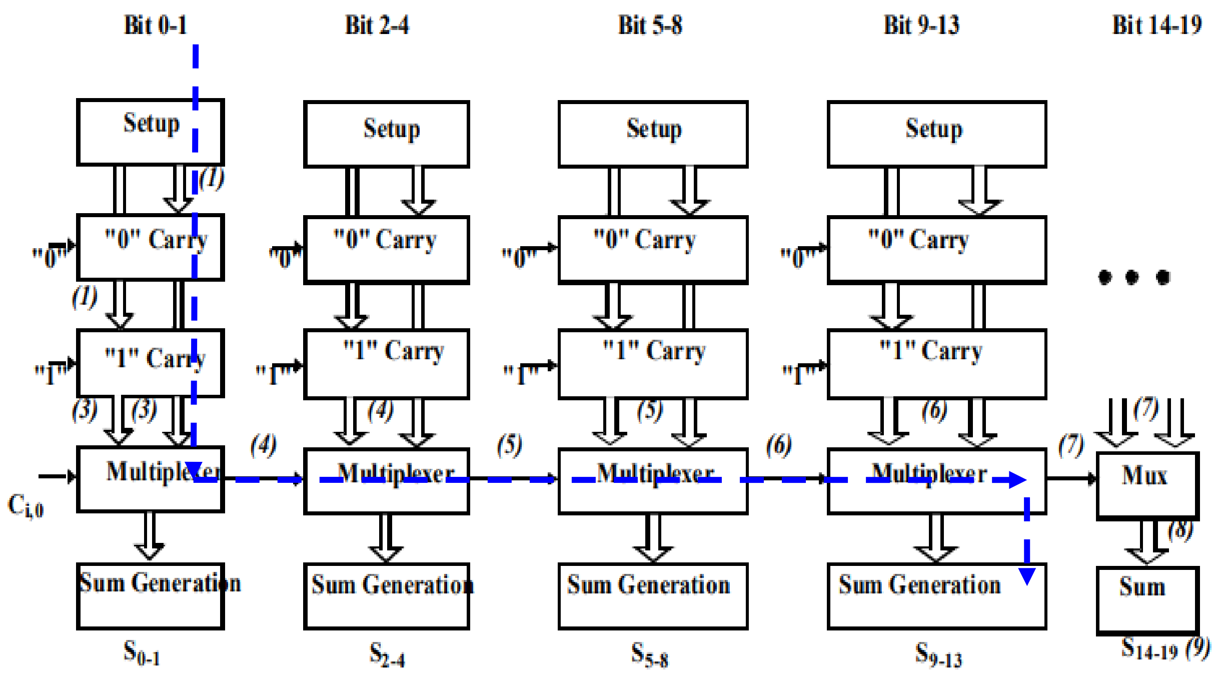
T_{mux}→time required for the carry to prop through mux

T_{mux}→time required to generate sum output.



$$t_{add} = t_{setup} + \left(\frac{N}{M}\right) t_{carry} + M t_{mux} + t_{sum}$$

2. SQUARE ROOT CARRY SELECT ADDER:



$$t_{add} = t_{setup} + P \cdot t_{carry} + (\sqrt{2N}) t_{mux} + t_{sum}$$

- To improve the performance of LCSA, the no of bits in each stage can be made to increasing progressively.For eg,the 1 stage can add 2 bits,2 stage can add 3 bits and 3 stage 4 and so on. This topology is faster than the LCSA,even though an axtra stage is needed.
- For 20 bit adder , the tp is same for both linear and square root CSA.
- Assume that an N bit adder contains “P” no of stages and first stage adds ‘M’no of bits.
- An Additional bit is added to each subsequent stage.

Therefore,P=0 toP-1.

Total no of address=N=M+(M+1)+(M+2).......+[M+(P-1)]

N=MP+(1+2+3+........P-1)

=MP+P(P-1)/2=MP+(P^2/2)-(P/2)

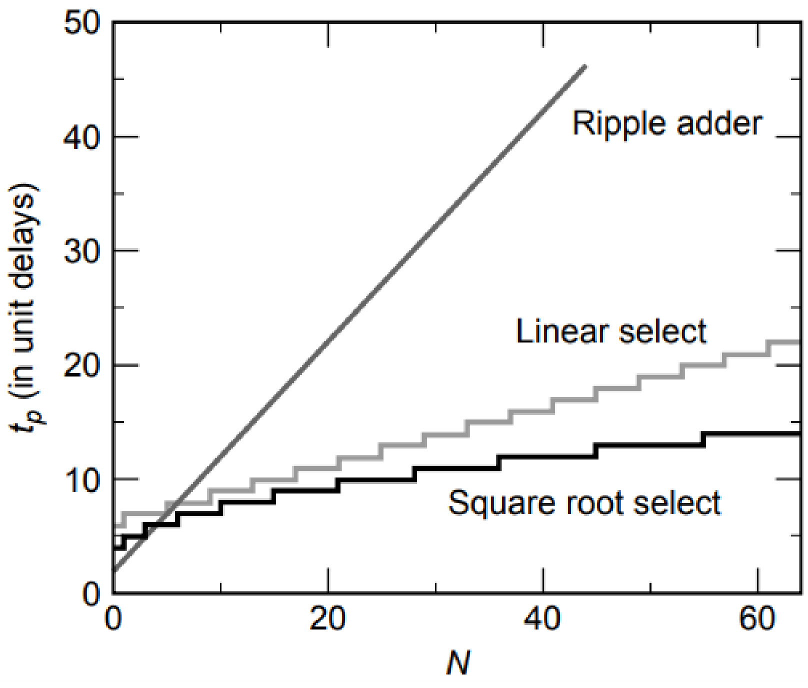
=(p^2/2)+p(M-(1/2))

N=P^2/2 if p^2/2 >>P()M-(1/2)

P^2=2N.

P=2^(1/2)

Comparison of adders



3. MULTIPLIERS:-

- Multipliers are comple adder arrays and are expensive and slow operation. It is used in digital signal processor and microprocessor.
- Consider two unsigned binary number X and Y that are M and N bits wide respectively.

$$X = \sum_{n=0}^{N-1} x_n 2^n , \qquad Y = \sum_{n=0}^{N-1} y_n 2^n$$

$$Z = XY = \left(\sum_{n=0}^{N-1} x_n 2^n\right)\left(\sum_{m=0}^{N-1} x_m 2^m\right)$$

$$= \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} x_n y_m 2^{n+m} = \sum_{n=0}^{2N-1} z_n 2^n$$

-
- A simplest way to perform multiplication is to use a single 2 ^{i}/p adder and the multiplication takes place in M cycles using N bit adder if the i/p's are M and N bits wide.

Multiplication has following 3 functions

1. Partial product generation, Partial product accumulation, Finally addition

				1	0	1	0	1	0	Multiplicand
x						1	0	1	1	Multiplier
				1	0	1	0	1	0	Partial products
			1	0	1	0	1	0		
		0	0	0	0	0	0			
+	1	0	1	0	1	0				
	1	1	1	0	0	1	1	1	0	Result

Multiplicand					y_3	y_2	y_1	y_0	$\times$
Multiplier					x_3	x_2	x_1	x_0	$=$
Partial Products					x_0y_3	x_0y_2	x_0y_1	x_0y_0	
				x_1y_3	x_1y_2	x_1y_1	x_1y_0		
			x_2y_3	x_2y_2	x_2y_1	x_2y_0			
		x_3y_3	x_3y_2	x_3y_1	x_3y_0				
Result	z_7	z_6	z_5	z_4	z_3	z_2	z_1	z_0	

1. Partial product generation

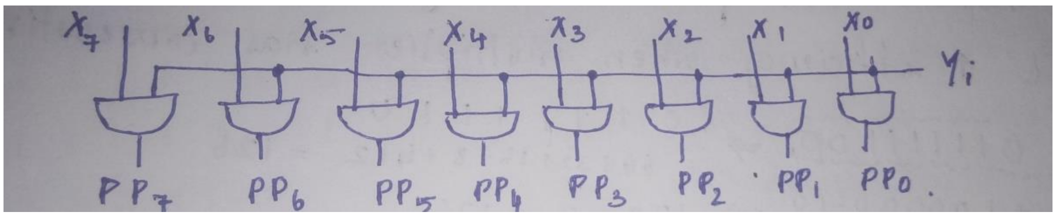
- ❖ Each partial product is generated by multiplying the multiplicand x with a multiplier y which is an AND operation. The usual algorithm for multiplying signed and unsigned integers are
- ❖ Example 1: 13*-6 (00001101 *11111010)

```
                       0  0  0  0  1  1  0  1
                  X    1  1  1  1  1  0  1  0
-------------------------------------------------
                     | 0  0  0  0  0  0  0  0
                  0  | 0  0  0  1  1  0  1
               0  0  | 0  0  0  0  0  0
            0  0  0  | 0  1  1  0  1
         0  0  0  0  | 1  1  0  1
      0  0  0  0  1  | 1  0  1
   0  0  0  0  1  1  | 0  1
0  0  0  0  1  1  0  | 1
-------------------------------------------------
0  0  0  1  0  0  0  | 1  0  1  1  0  0  1  0
-------------------------------------------------
\_______ _______/      \__________ __________/
        V                         V
Discard                          -78
```

Example 2:42*11

```
         1  0  1  0  1  0
            x  1  0  1  1
--------------------------
         1  0  1  0  1  0
      1  0  1  0  1  0
   0  0  0  0  0  0
1  0  1  0  1  0
--------------------------
1  1  1  0  0  1  1  1  0
--------------------------
\__________ ___________/
           V
                  462
```

- ❖ To generate the partial products, the following circuits can be used.
- ❖



- A technique that works equally well for both negative and positive numbers are called the Booth Algorithm. In Booth algorithm every two overlapping bits are recoded into single bit and one bit Append/pad at LSB as '0'

Multiplex0 Bit i Bit i-1	Recoded bits
00	0 x Multiplicand
01	+1 x Multiplicand
10	-1 x Multiplicand
11	0 x Multiplicand

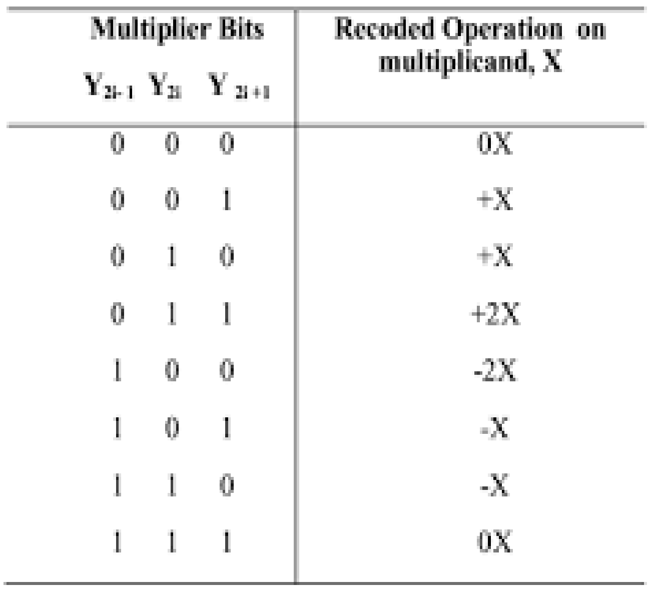
- Example: 13*-6 = 11010
  11--- (0*Multiplicand)
  10--- (-1*Multiplicand)
  01--- (1*Multiplicand)
  10--- (-1*Multiplicand)
  00 --- (0*Multiplicand)

- Example:

$$
\begin{array}{cccc|ccccccc}
 & & & & & & 0 & 1 & 1 & 0 & 1 \\
 & & & & & X & 0 & -1 & +1 & -1 & 0 \\
\hline
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & \\
 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & & \\
 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & & & \\
 & 0 & 0 & 0 & 0 & 0 & 0 & & & & \\
\hline
\multicolumn{4}{c}{\underbrace{1\quad 1\quad 1\quad 1}_{\text{Sign bits}}} & \multicolumn{7}{c}{\underbrace{0\quad 1\quad 1\quad 0\quad 0\quad 1\quad 0}_{-78}}
\end{array}
$$

- The booth algorithm has 2 attractive features.
1) It handles both positive and negative multipliers uniformly
2) It increases efficiency when multiplier has consecutive **1**'s.
   Ex: 011111100 = [+100000-10] skipping operation because of 0's
- The speed is dependent on the data. On an average the speed of doing multiplication with booth algorithm is same as with normal algorithm. Modified booth algorithm can be used to reduce the partial products.
- In modified booth algorithm, append/pad one sign bit in MSB if the no of multipliers bit is odd number.

TABLE I. RADIX - 4 BOOTH RECODING



Multiplier Bits Y_{2i-1}	Y_{2i}	Y_{2i+1}	Recoded Operation on multiplicand, X
0	0	0	0X
0	0	1	+X
0	1	0	+X
0	1	1	+2X
1	0	0	-2X
1	0	1	-X
1	1	0	-X
1	1	1	0X

Example : 13*-6 = 01101*11010
Since the no of bits is odd, append one bit in LSB and MSB
Therefore multiplier = 1110100
The overlapping bits are
111 --- (0* Multiplicand)
101 --- (-1 * Multiplicand)
100 --- (-2* Multiplicand)
Therefore multiplier is 01101
Multiplicand is 0 -1 -2

						0	1	1	0	1
					X	0	-1	-2		
	1	1	1	1	1	0	0	1	1	0
	1	1	1	1	0	0	1	1	*	*
	0	0	0	0	0	0	*	*		
1	1	1	1	0	1	1	0	0	1	0

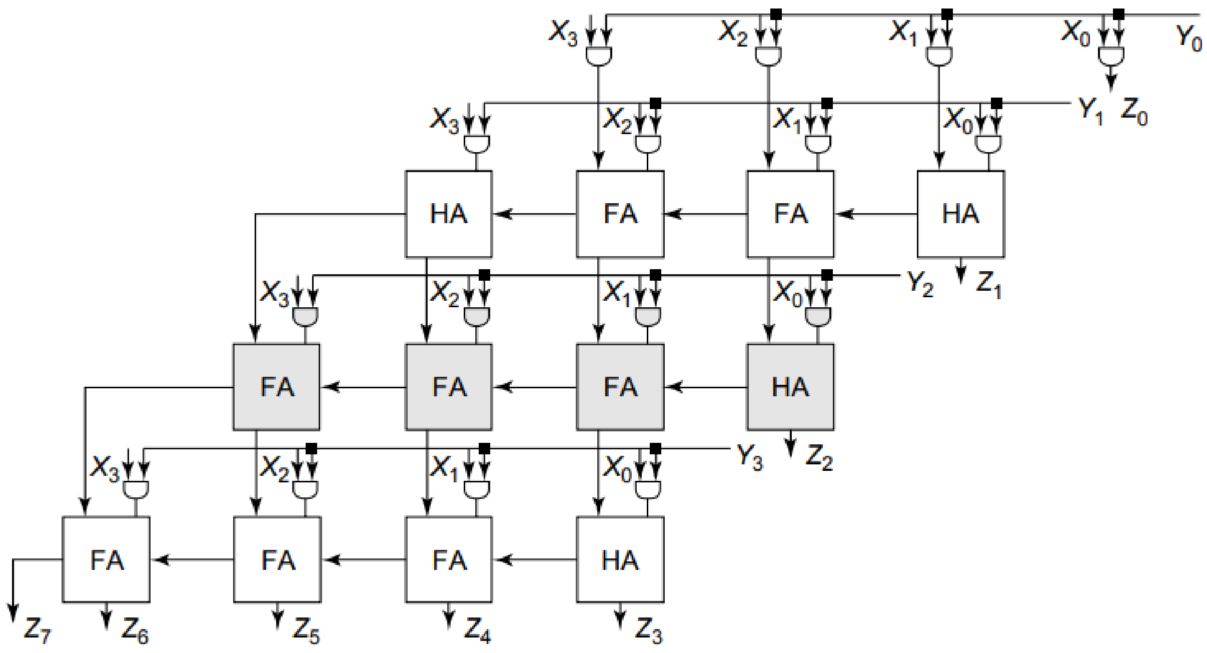
Sign bits -78.

- ❖ Only 3 partial products so required operation. **1**'s reduced from 5 to 3. There by the performance will be increased.

2. Partial Product Accumulation
After generating the partial products, they must be added in accumulator.

I. Array Multiplier

A straight forward method to accumulate the partial products by using no of adders is called array multiplier.

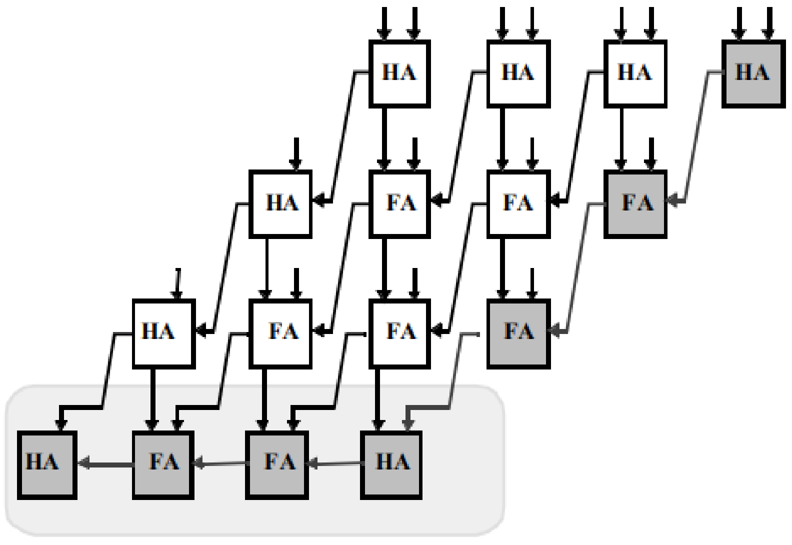


❖ In this all the critical paths are having same length, the performance is poor thereby another multiplier which user CSA yields marginal benefit is called carry save multiplier.

II. Carry save multiplier

❖ In this multipliers, the multiplication result does not change when the carry bits are passed diagonally downwards instead of moving right.

❖ The speed of the carry save multiplier is more than the array multiplier but it req's an extra adder called vector merging adder to generate the final result .



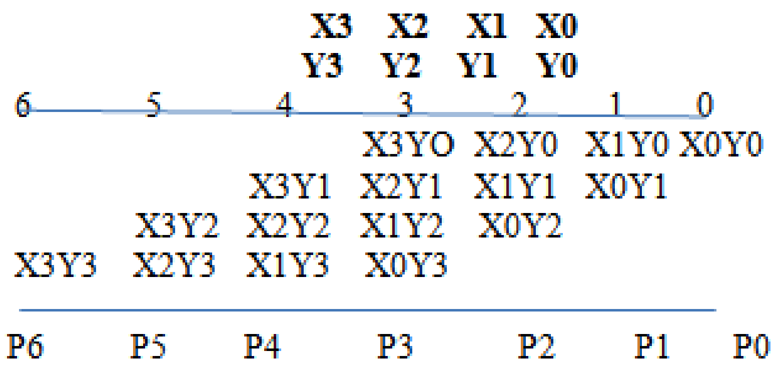
Vector Merging Adder

❖ The final stage carry and sum's are merged using either RCA or fast carry propagate adder called carry look ahead adder. The advantage of this method is critical path is shorter.

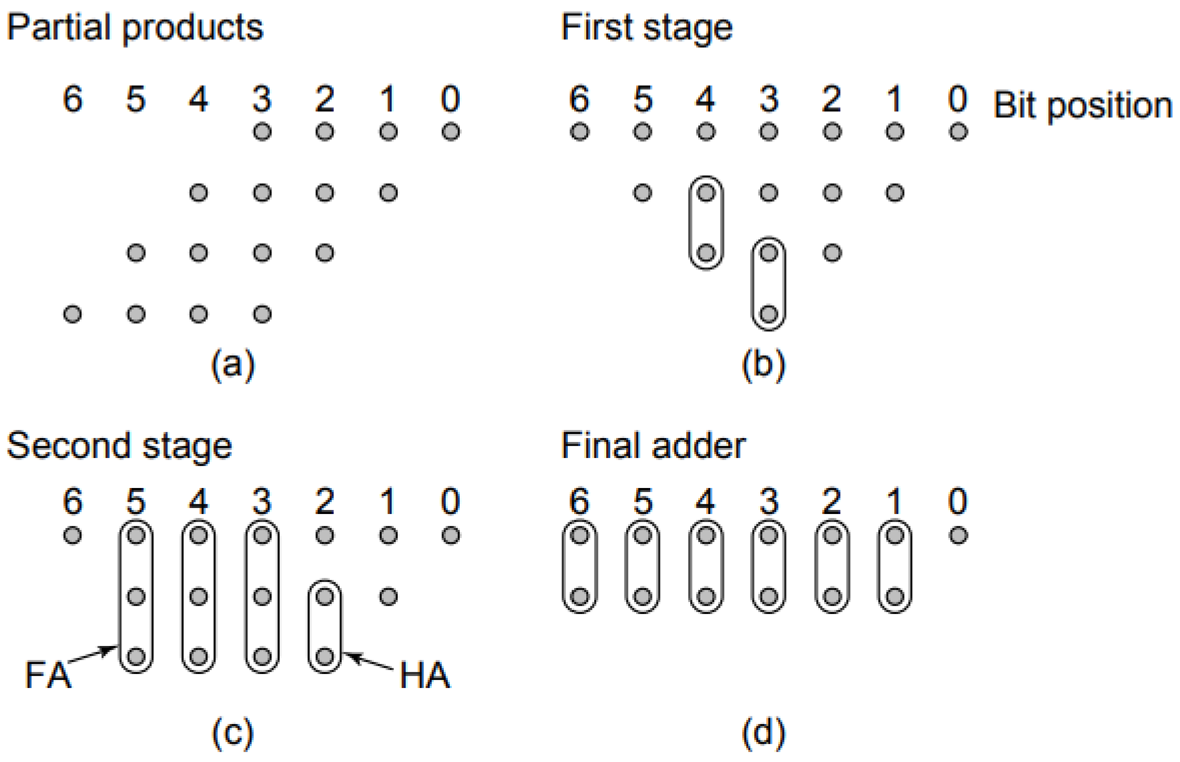
III. Wallace Tree Multiplier:

In this multiplier the partial sum adders can be rearranged in a tree like structure. This reduces the length of the critical path and no of adders required for multiplication.

Similarly for 4x4 multipliers



			X3	X2	X1	X0
			Y3	Y2	Y1	Y0
6	5	4	3	2	1	0
			X3YO	X2Y0	X1Y0	X0Y0
		X3Y1	X2Y1	X1Y1	X0Y1	
	X3Y2	X2Y2	X1Y2	X0Y2		
X3Y3	X2Y3	X1Y3	X0Y3			
P6	P5	P4	P3	P2	P1	P0



This can be arranged as a tree like structure

FIRST STAGE

```
6         5      4        3      2        1      0
X3Y3   X3Y2   X3Y1   X3Y0   X2Y0   X1Y0  X0Y0
          X2Y3  X2Y2   X2Y1   X1Y1  X0Y1
                   X1Y3   X1Y2  XOY2
C1,S1-HA<-            C0   X0Y3 ---> HA-S0,C0
```

SECOND STAGE

```
6         5            4          3        2         1      0
X3Y3   X3Y2        X3Y1    X3Y0    X2Y0   X1Y0  X0Y0
          X2Y3     X2Y2    X2Y1    X1Y1   X0Y1
           C4        S1      S0     X0Y2
                     C3      C2           --->
```

FINAL STAGE

```
 6       5     4       3     2          1        0
X3Y3   S5   S4    S3   X2Y0   X1Y0   X0Y0
 C5    C4   C3    C2   S2     X0Y1
```

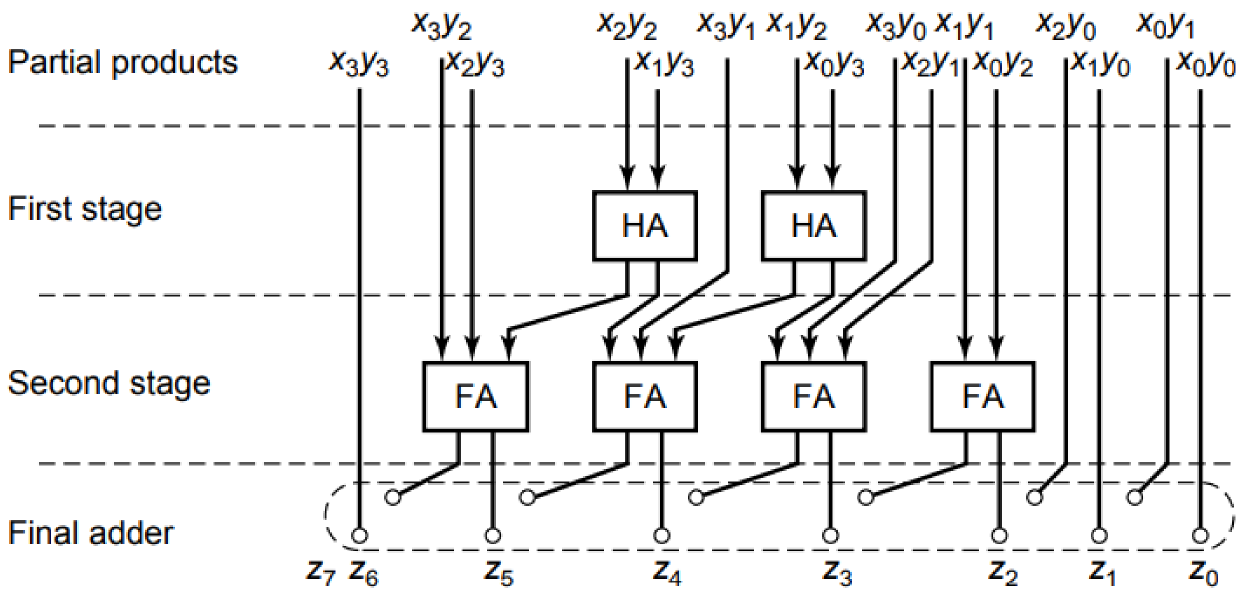
The W T M req's 3HA's and 3FA's except final stage . compared with the 6 FA's and 6 HA's in the CSM. Therefore there is substantial reduction in hardware and delay.

FINAL ADDITION

- ❖ The final step for completing the multiplication is to combine the result in the FA .
- ❖ The choice of adder in final stage depends on the structure of the structure of the accumulation array. CLA is preferable if all the input bits to adder arrive at the same time output is preferable.

Disadvantage

The structure of the multiplier is very irregular



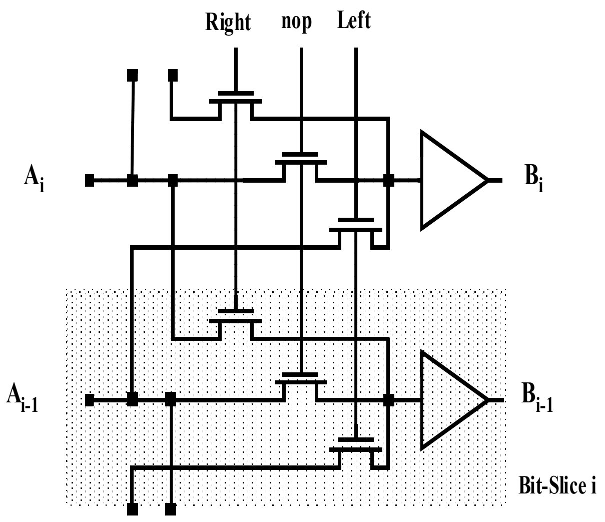
4. SHIFTERS

Types

Logical Shift: – Shifts number left or right and fills with 0's
Arithmetic Shift: – Shifts number left or right.
Rotate: – Shifts number left or right and fills with lost bits

1. Binary shifter

A_i	A_{i-1}	rgt	nop	left	B_i	B_{i-1}
A_1	A_0	0	1	0	A_1	A_0
A_1	A_0	1	0	0	0	A_1
A_1	A_0	0	0	1	A_0	0



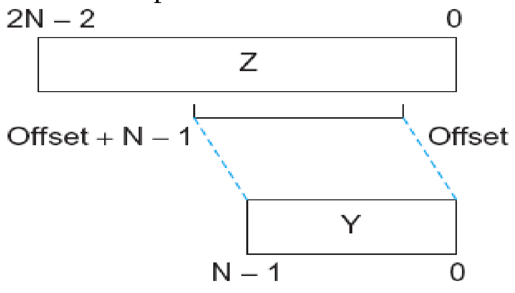
...

Disadv

Too slow for large shift values

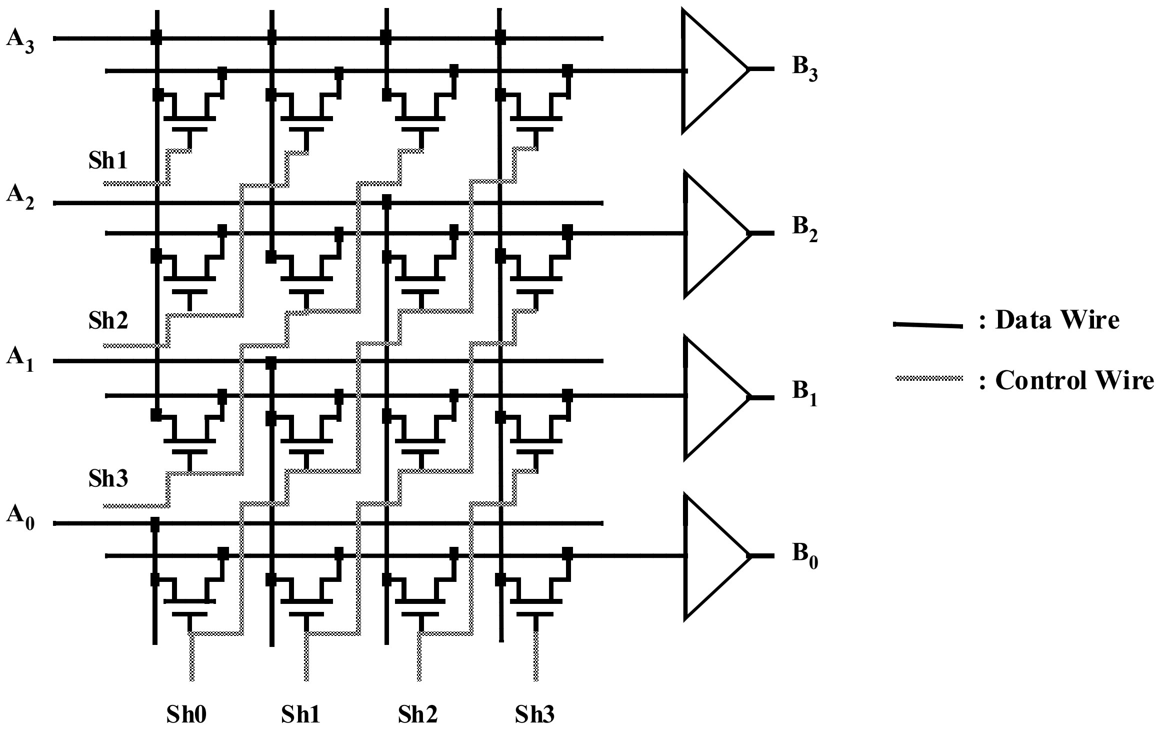
2. Funnel shifter

- ❑ A funnel shifter can do all six types of shifts
- ❑ Selects N-bit field Y from 2N–1-bit input
  - – Shift by k bits (0 ≤ k < N)
  - – Logically involves N N:1 multiplexers



_

3. BARREL SHIFTER



- Number of rows = data word length
- Control wire routed diagonally
- Signal goes through only one transmission gate (theoretically delay is constant for shift value and shifter size)
- Reality – delay depends on shift widths due to parasitic capacitance
- Layout and area dominated by wiring and not active elements
- Need decoder to interpret shift data to route signal to appropriate wire
- The barrel shifter consists of an array of transistors In this, the no of rowa equal to the word length of the data and the no of columns equal to the maximum shift width.
- But in this case, both are set equal to four .The control wires are routed diagonally through the array. The operation is given below.
- A major advantages of this shifter is that the signal has to pass through only one point. So the delay is constant and is independent of the shift value or shifter sixe.

Disadvantages

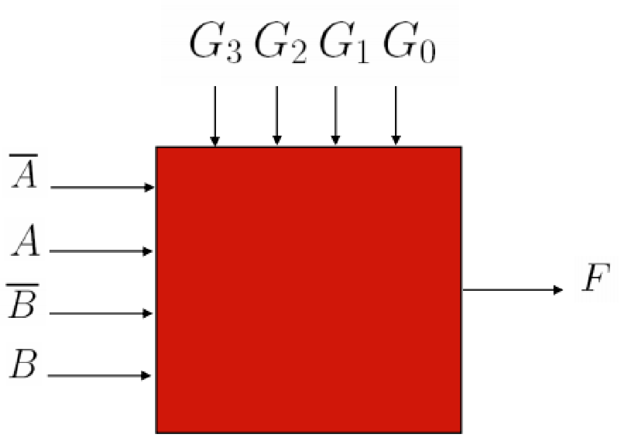
It requires 4 control signals. This can be decrease by using encoder and decoder.

4. ALU:

- ALU computes a variety of logical and arithmetic functions based on opcode.
- May offer complete set of functions of two variables or a subset.
- ALU built around adder, since carry chain determines delay.
- ALU should be able to perform functions:
  – logical and function
  – logical or function
  – arithmetic add function

– arithmetic subtract function
– arithmetic slt (set-less-then) function
– logical nor function

- • ALU control lines define a function to be performed on A and B.



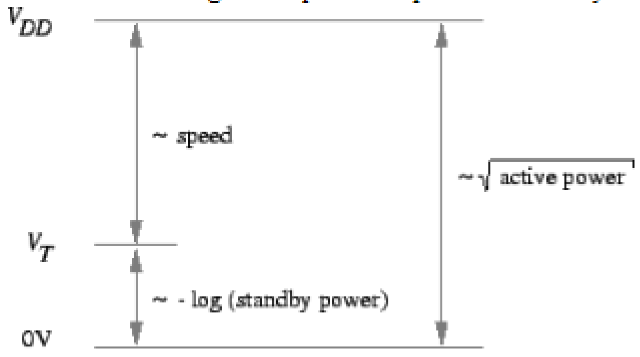
$$F(A,B) = G_0 \cdot \overline{A}\,\overline{B} + G_1 \cdot \overline{A}\,B + G_2 \cdot A\,\overline{B} + G_3 \cdot A\,B$$

A	B	F
0	0	G_0
0	1	G_1
1	0	G_2
1	1	G_3

G_3	G_2	G_1	G_0	$F(A,B)$	G_3	G_2	G_1	G_0	$F(A,B)$
0	0	0	0	0	1	0	0	0	$A \cdot B = \overline{\overline{A} + \overline{B}}$
0	0	0	1	$\overline{A} \cdot \overline{B} = \overline{A + B}$	1	0	0	1	$A \cdot B + \overline{A} \cdot \overline{B}$
0	0	1	0	$\overline{A} \cdot B = \overline{A + \overline{B}}$	1	0	1	0	B
0	0	1	1	$\overline{A}$	1	0	1	1	$\overline{A \cdot \overline{B}} = \overline{A} + B$
0	1	0	0	$A \cdot \overline{B} = \overline{\overline{A} + B}$	1	1	0	0	A
0	1	0	1	$\overline{B}$	1	1	0	1	$\overline{\overline{A} \cdot B} = A + \overline{B}$
0	1	1	0	$A \cdot \overline{B} + \overline{A} \cdot B$	1	1	1	0	$\overline{\overline{A} \cdot \overline{B}} = A + B$
0	1	1	1	$\overline{A \cdot B} = \overline{A} + \overline{B}$	1	1	1	1	1

POWER AND SPEED TRADEOFFS

Figure 2.1: Qualitative relations between supply and threshold voltage and speed and power efficiency

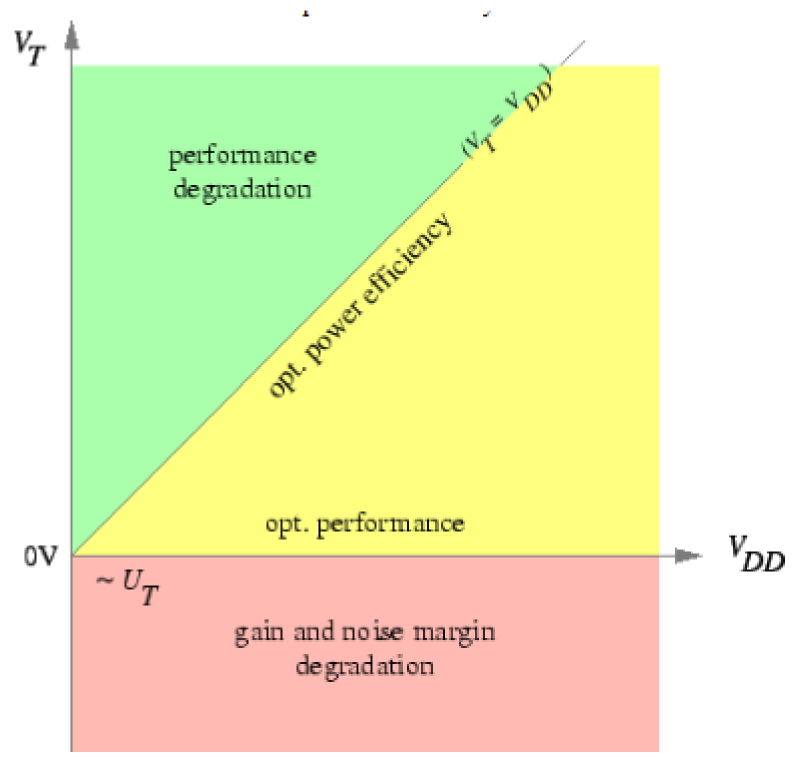


V_T	power eff.	speed	noise margins	
$> V_{DD}$	good	bad	good	
$\approx V_{DD}$	good	poor	good	(mod. inversion determines speed)
$0 < V_T < V_{DD}$	fair	fair	fair	
≈ 0	poor	good	poor	(mod. inversion determines power eff.)
<0	bad	good	bad	

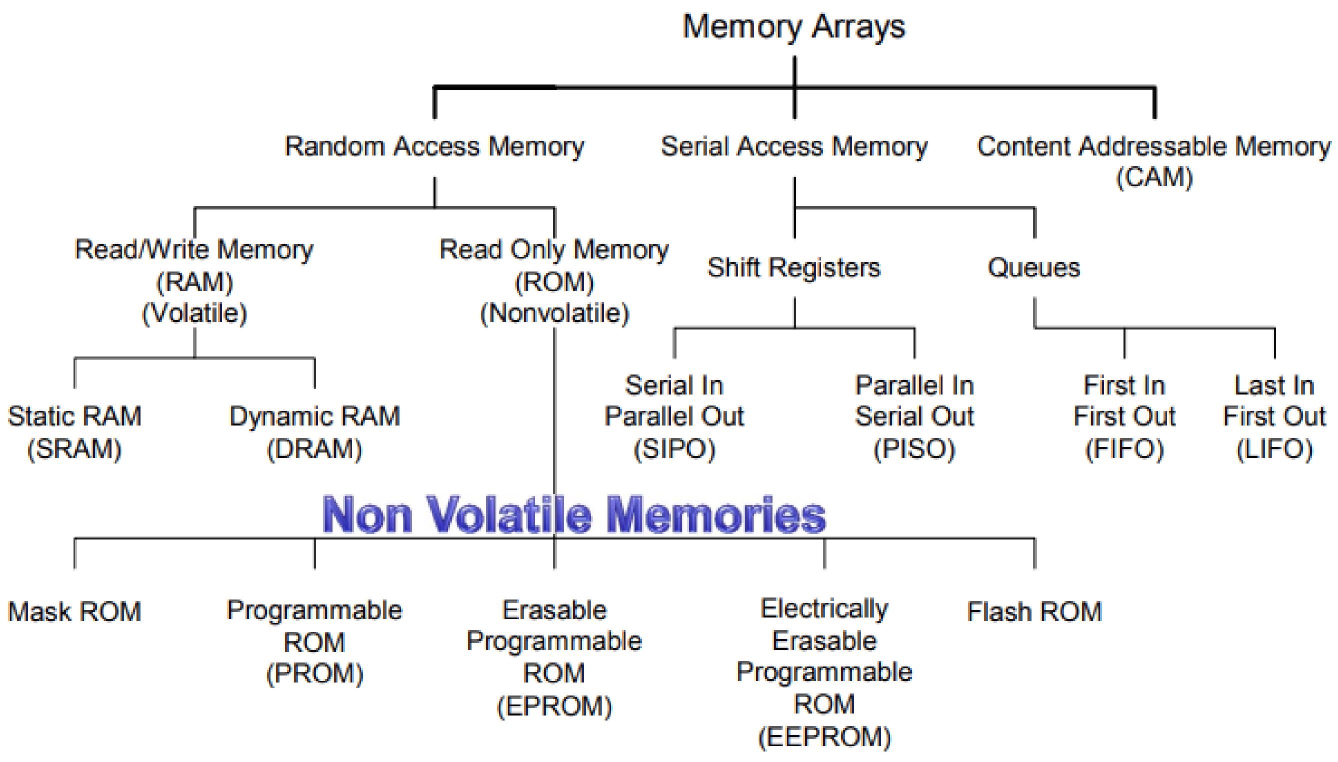
Most notably, the loci of optimum power efficiency (at acceptable speed) and for optimum speed are in the vicinity of $V_T = V_{DD}$ and $V_T = 0$.

The condition $V_{GS} \approx V_T$, which marks the moderate-inversion region of a MOSFET is critical for the speed in the first case and critical for the power consumption and noise immunity in the second case.

Case study: Impact of supply and threshold voltage on speed and power efficiency: the area around $V_{DD} = 1.5\text{V}$ marks the conventional strategy, at $V_{DD} = 0.3\text{V}$ the optimum energy efficiency is reached, and area around $V_{DD} = 0.5\text{V}$ indicates good energy efficiency and speed.

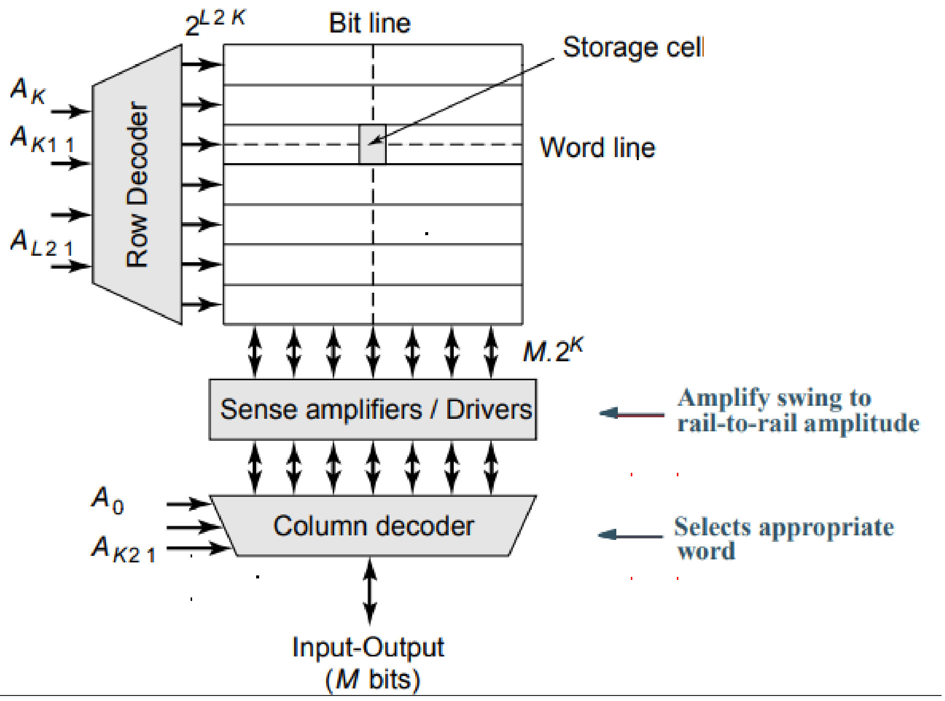


MEMORY AND ARRAY STRUCTURES:


Memory Arrays
Random Access Memory
Serial Access Memory
Content Addressable Memory (CAM)
Read/Write Memory (RAM) (Volatile)
Read Only Memory (ROM) (Nonvolatile)
Shift Registers
Queues
Static RAM (SRAM)
Dynamic RAM (DRAM)
Serial In Parallel Out (SIPO)
Parallel In Serial Out (PISO)
First In First Out (FIFO)
Last In First Out (LIFO)
Non Volatile Memories
Mask ROM
Programmable ROM (PROM)
Erasable Programmable ROM (EPROM)
Electrically Erasable Programmable ROM (EEPROM)
Flash ROM

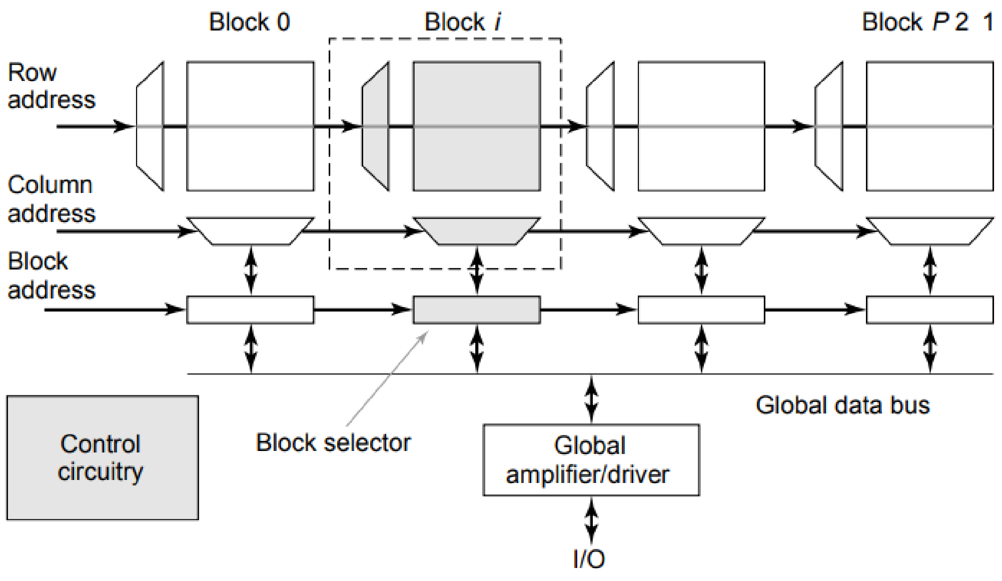
<u>MEMORY ARCHITECTURES AND BUILDING BLOCKS</u>:

Array-Structured Memory Architecture Structured Memory Architecture

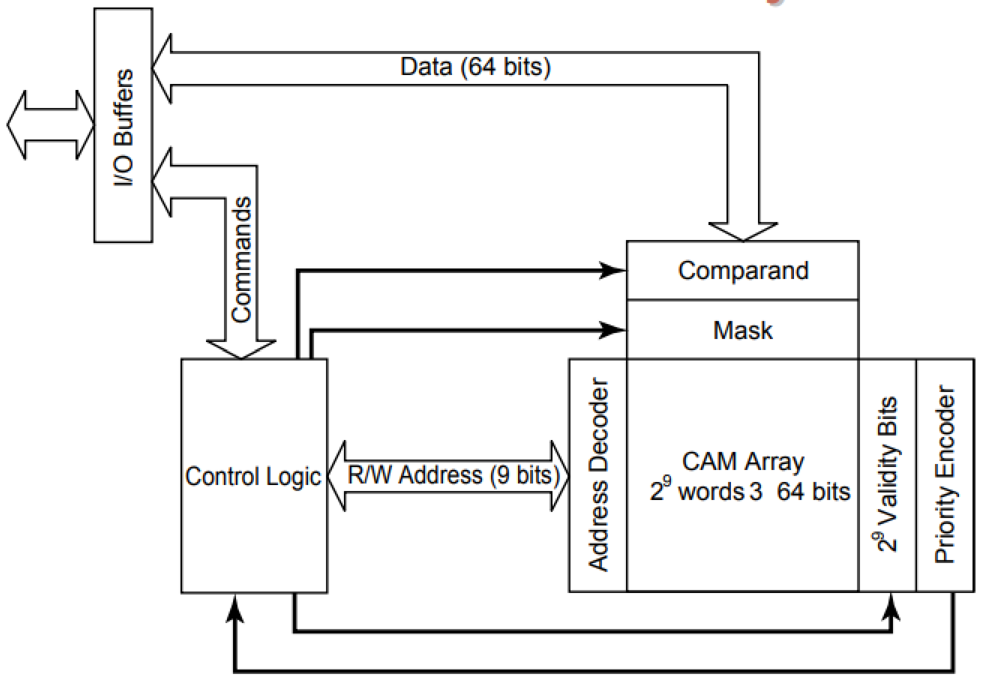


- Address of n bits, split into two parts
  - "Row" (n-k bits)
  - "Column" (k bits)
- n-k row address bitsθ used to decode 1 of 2n-k rows
- All cells on selectedθ row sensed simultaneously
- Array reads outθ (2k)*(2m) bits – called an open row
- C Column addressθ bits select one of 2k words from open row
-

Hierarchical Memory Architecture



Contents Contents-Addressable Memory



<u>MEMORY CORE :</u>

I. Random access memory

RAMs are of two types, static and dynamic. Circuits similar to basic D flip-flop are used to construct static RAMs (SRAMs) internally. A typical SRAM cell consists of six transistors which are connected in such a way as to form a regenerative feedback. In contrast to DRAM, the information stored is stable and does not require clocking or refresh cycles to sustain it. Compared to DRAMs, SRAMs are much faster having typical access times in the order of a few nanoseconds. Hence SRAMs are used as level 2 cache memory. Dynamic RAMs do not use flip-flops, but instead are an array of cells, each containing a transistor and a tiny capacitor. '0's and '1's can be stored by charging or discharging the capacitors. The electric charge tends to leak out and hence each bit in a DRAM must be refreshed every few milliseconds to prevent loss of data. This requires external logic to take care of refreshing which makes interfacing of DRAMs more complex than SRAMs. This disadvantage is compensated by their larger capacities. A high packing density is achieved since DRAMs require only one transistor and one capacitor per bit. This makes them ideal to build main memories. But DRAMs are slower having delays in the order tens of nanoseconds. Thus the combination of static RAM cache and a dynamic RAM main memory attempts to combine the good properties of each.

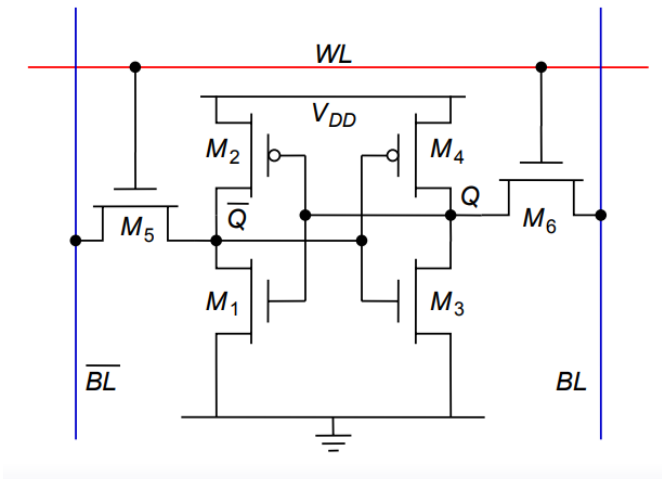
Types

1. SRAM
2. DRAM

Features:

- Volatile: loses its data when the power is turned off •
- Read and written quickly
- SRAM uses cross-coupled inverters

1.SRAM



To read SRAM

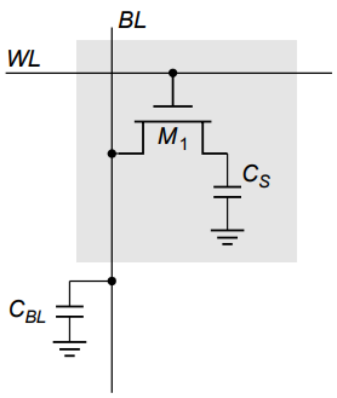
- ➢ Precharge both bitlines high
- ➢ Then turn on wordline
- ➢ One of the two bitlines will be pulled down by the cellθ
- ➢ Ex: A = 0, A_b = 1θ – bit discharges, bit_b stays high

To write into SRAM

- ➢ Drive one bitline high, the other low
- ➢ Then turn on wordline
- ➢ Bitlines overpower cell with new value
- ➢ Ex: A = 0, A_b = 1, bit = 1, bit_b = 0θ – Force A_b low, then A rises high

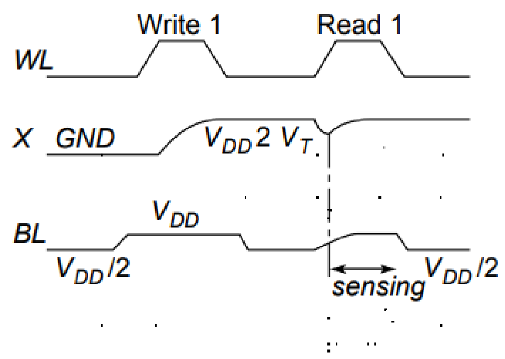
2.DRAM

- ➢ Data bits stored on a capacitor
- ➢ Called dynamic because the value needs to be refreshed (rewritten) periodically and after being read
- ➢ 1T DRAM requires a sense amplifier for each bit line, due to charge redistribution read-out. ‰
- ➢ DRAM memory cells are single ended in contrast to SRAM cells.
- ➢ The read-out of the 1T DRAM cell is destructive; read and refresh operations are necessary for correct operation.

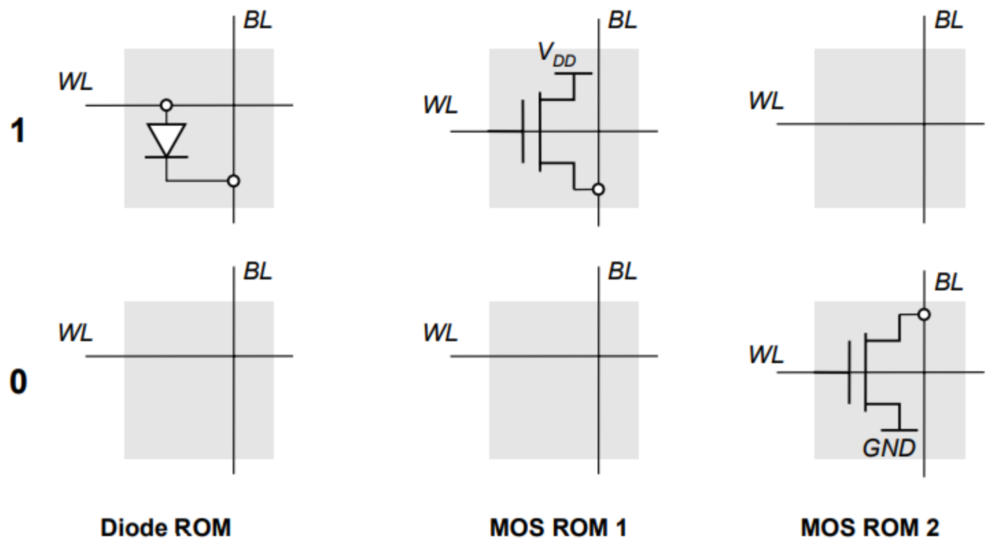


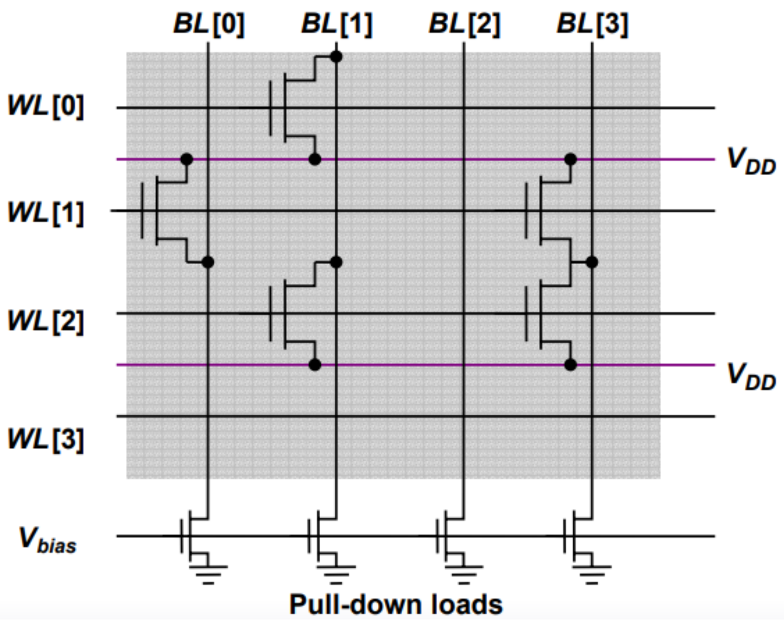
Write: CS is charged or discharged by asserting WL and BL.

Read: Charge redistribution takes places between bit line and storage capacitance



II. Read Only Memory





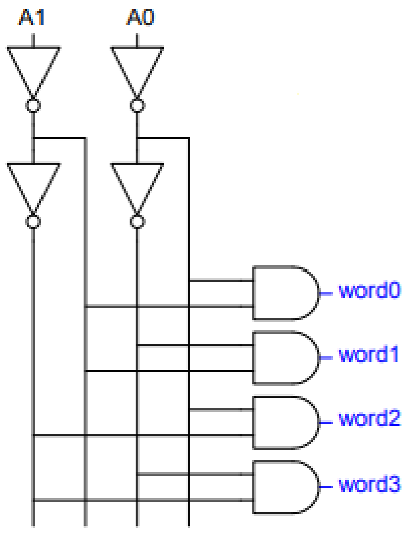
Example: 0100,1001,0101,0000

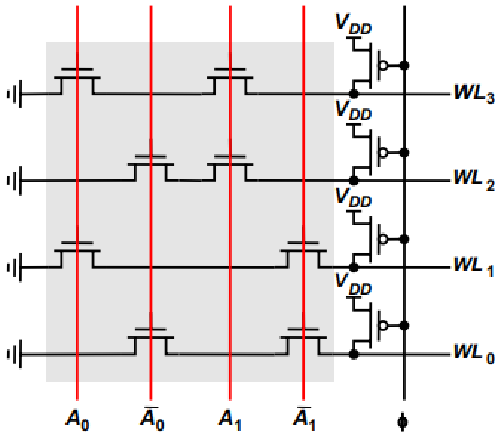
MEMORY PERIPHERAL CIRCUITS:

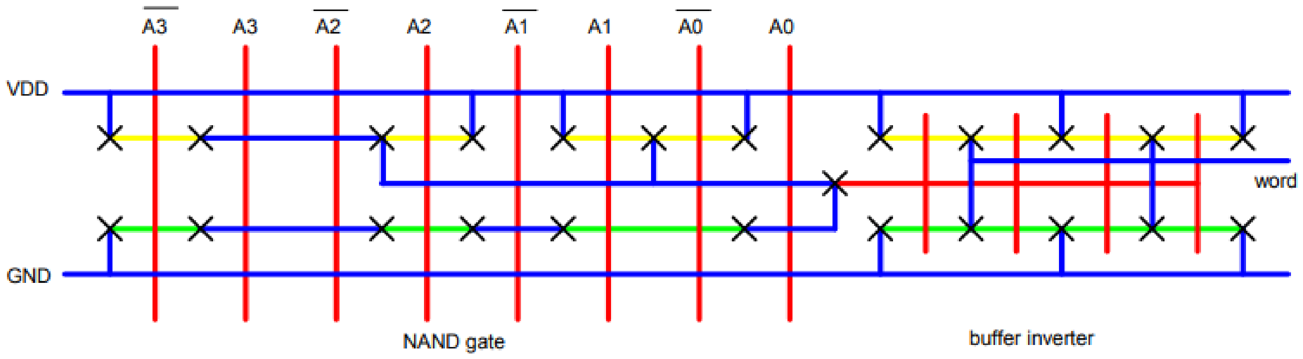
1. Decoders
2. Sense Amplifiers
3. Input/Output Buffers
4. Control / Timing Circuitry

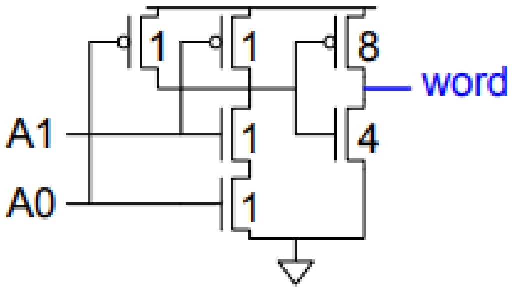
1. Decoders

n:2n decoder consists of 2n n-input AND gates
– One needed for each row of memory
– Build AND from NAND or NOR gates









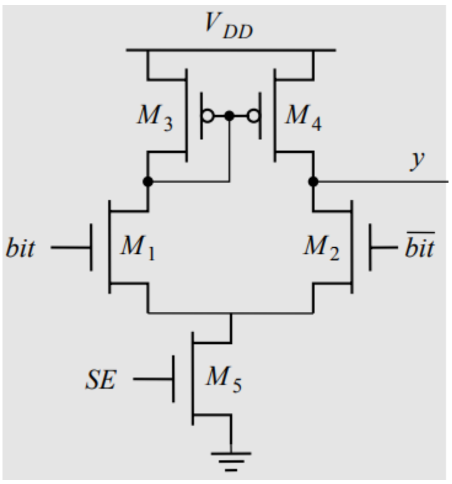
2. Sense Amplifiers

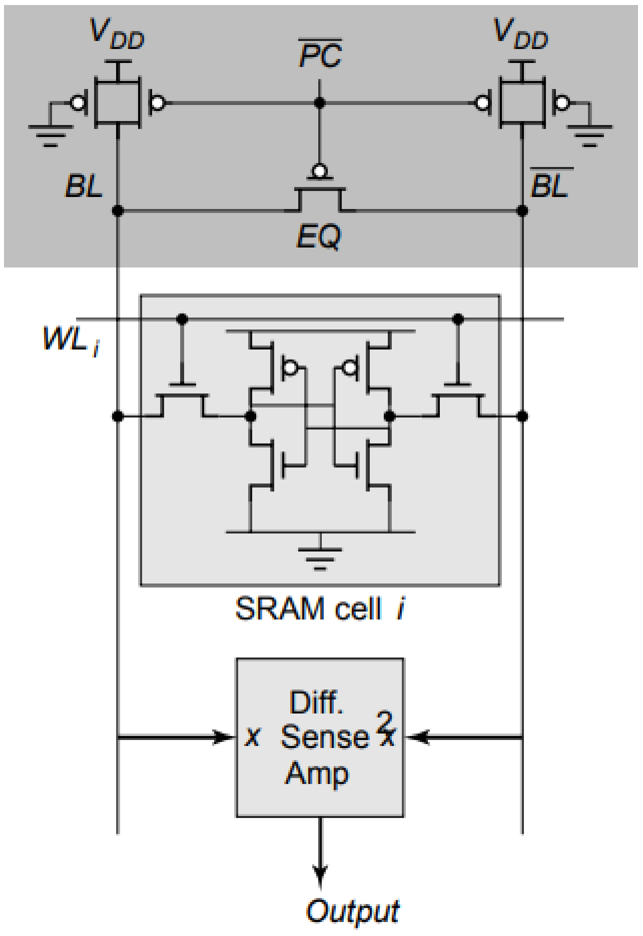
- Bitlines have many cells attached

$$t_{pd} \propto (C/I)\,\Delta V$$

  - Even with shared diffusion contacts, 64C of diffusion capacitance (big C)
  - Discharged slowly through small transistors (small I)
- Sense amplifiers are triggered on small voltage swing V)Δ(reduce

Differential Sense Amplifier





(a) SRAM sensing scheme

PART A

1. **Define kill, generate and propagate terms in Carry look ahead adder? A/M 19**

$$G_i = A_i\,B_i$$

$$P_i = A_i + B_i$$

$$K_i = \overline{A_i + B_i} = \overline{A}_i\,\overline{B}_i$$

2. **State radix 2 booth encoding table. A/M 19**

Multiplex0 Bit i Bit i-1	Recoded bits
00	0 x Multiplicand
01	+1 x Multiplicand
10	-1 x Multiplicand
11	0 x Multiplicand

3. **The circuit in fig 2 shows a carry propagation path in an adder circuit? Let A,B,C are the inputs to adder circuit and φ is the clock signal. Write the logic expressions for the signals X,Y to generate carry? N/D 18**

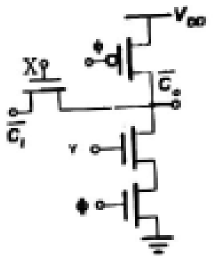


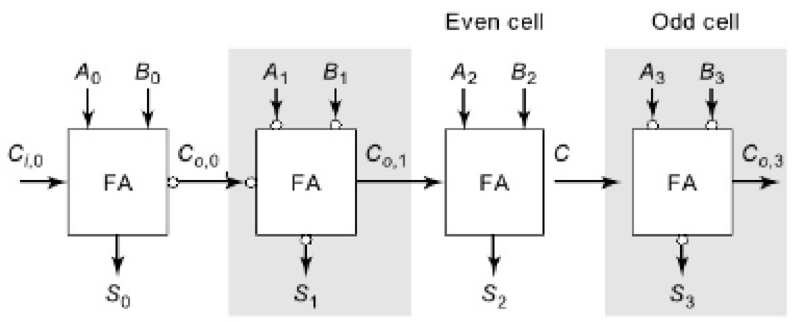
Fig. 2

Generate (G) = AB

Propagate (P) = A ⊕ B

Delete = $\overline{A}\,\overline{B}$

4. **Draw a 4 bit ripple carry and find its critical path delay? N/D 18**



$t_{adder} = (N-1)\,t_{carry} + t_{sum}$

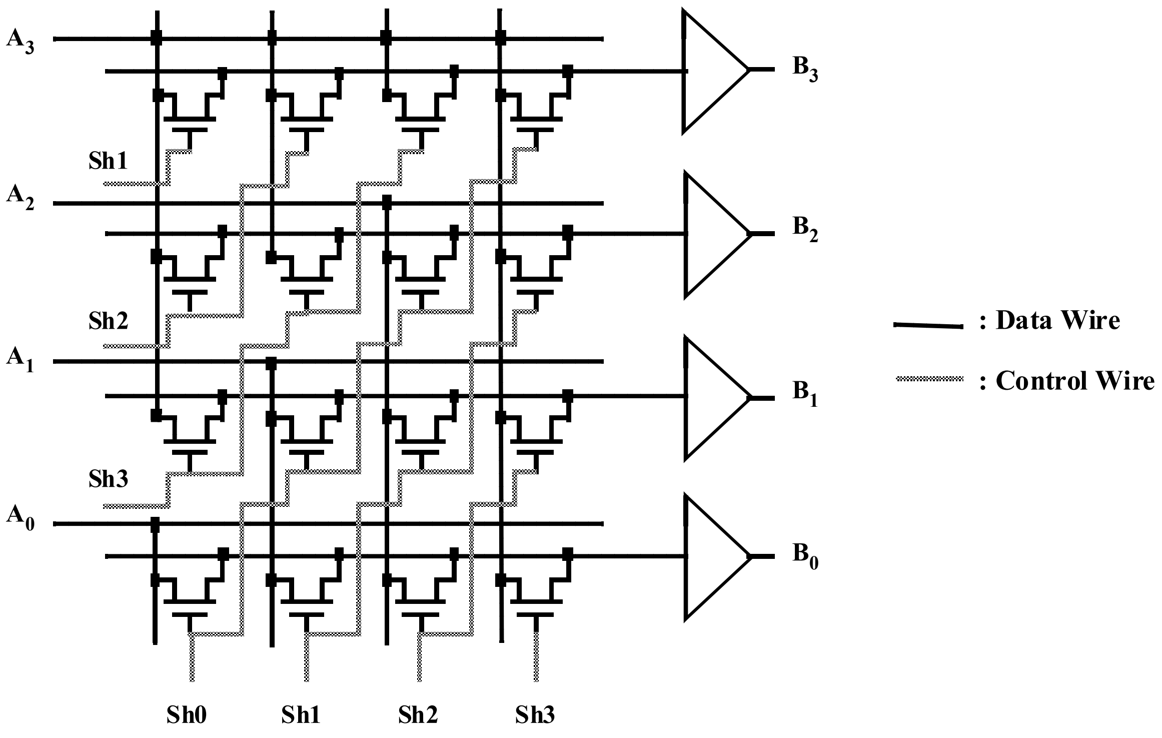
5. **Write full adder output in terms of propagate and generate? A/M 18**

   Generate (G) = AB

   Propagate (P) = $A \oplus B$

   Delete = $\overline{A}\,\overline{B}$

6. **Draw the structure of 4x4 barrel shifter? A/M 18**



7. **How to design a high speed adder?** **N/D 17**
8. **What is latency? N/D 17**
   In general clock latency is defined as the amount of time taken by the clock signal in travelling from its source to the sinks.

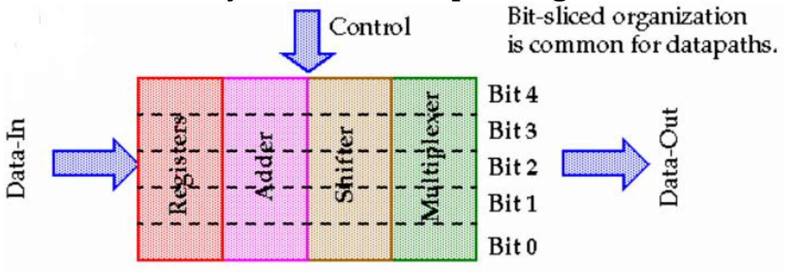
9. **List out components of datapath?** **A/M 17**
   Processor
   Alu
   Registers
   Internal buses

10. **Give the application of high speed adders?** **A/M 17**
    (I)carry bypass adder
    (II) linear carry select adder
    (III) square root carry select adder
    (Iv) carrylookahead adder

11. **Why is barrel shifter very useful in designing of arithmetic circuit? N/D 16**
    A barrel shifter is a digital circuit that can shift a data word by a specified number of bits without the use of any sequential logic only one pure combination logic . a barrel shifter is often used to shift and rotate n bits in modern microprocessors.

12. **Write principle of any one fast adders? N/D 16**
    The carry lookahead adder improves speed by reducing the amount of time required to complete a task.

13. **What is meant by bit-sliced data path organization? A/M 16**



14. **Determine propagation delay of n-bit carry select adder? A/M 16**
    $t_p = t_{setup} + Mt_{carry} + (N/M)\ t_{mux} + t_{sum}$.

PART B

1. Apply radix 2 booth encoding to realze a 4 bit signed multiplier for (-10)*(-11) **A/M 19-Part C**
2. Derive the necessary expressions for a 4 bit carry look ahead adder and realise carry out expressions using dynamic CMOS logic. **A/M 19 (OR)** Explain the concept of carry look ahead adder with neat diagram? **A/M 18**
3. Draw the structure of ripple carry adder and explain its operation. How the drawback in ripple carry adder is overcome by carry look ahead adder and discuss? **N/D 17 (OR)** Explain the operation of a basic 4 bit adder; describe the different approaches of improving speed of an adder? **N/D 16**
4. Design a 16 bit carry bypass and carry select adder and discuss their features. **A/M 16**
5. Design a 4bit unsigned array multiplier and analyse its hardware complexity **A/M 19 (OR)** Design a 4X4 array multiplier and write down the equation for delay. **A/M 16**
6. Design a clock distribution network based on H tree model for 16 nodes **A/M 18-Part C**
7. Design a multiplier for 5 bit by 3 bit. Explain the operation and summarise the number of adders. Discuss it over Wallace multiplier? **A/M 18, N/D 17**
8. Explain booth multiplier with a suitable example? A/M 17 **(OR)** Explain the operation of booth multiplication with suitable example? Justify how booths algorithm speeds up the multiplication? **N/D 16**
9. Discuss about area and speed trade-off? A/M 17

DEPARTMENT OFELECTRONICSAND COMMUNICATION ENGINEERING

(ACADEMIC YEAR: 2019-2020)

EC8095-VLSI DESIGN

(Regulation 2017)

Semester-VI

NAME-

REG NO-

MOHAMED SATHAK
A J COLLEGE OF ENGINEERING
SIRUSERI IT PARK, OMR, CHENNAI - 603103

UNIT V - IMPLEMENTATION STRATEGIES AND TESTING

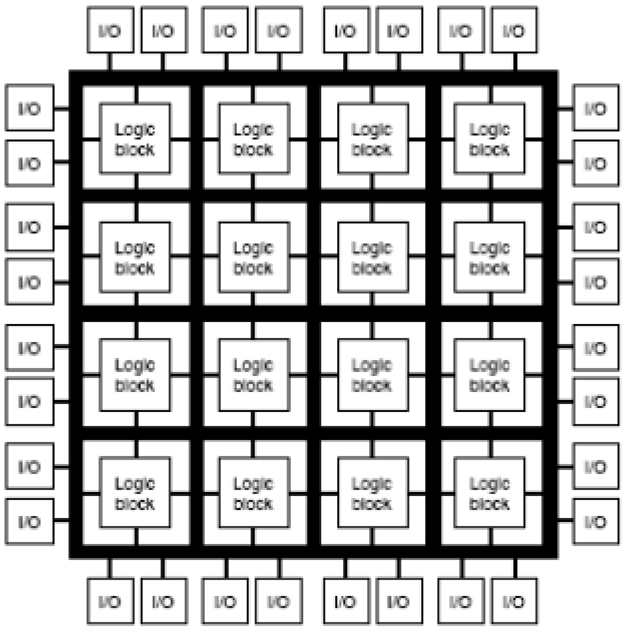
Field Programmable Gate Arrays

- A Field-Programmable Gate Array (FPGA) is an integrated circuit that can be configured by the user to emulate any digital circuit
- An FPGA can be seen as an array of Configurable Logic Blocks (CLBs) connected through programmable interconnect (Switch Boxes)
- Dominant digital design implementation
- Ability to re-configure FPGA to implement any digital logic function
  - Partial re-configuration allows a portion of the FPGA to be continuously running while another portion is being re-configured
- FPGAs also contain analog circuitry features including a programmable slew rate and drive strength, differential comparators on I/O designed to be connected to differential signaling channels.
- Mixed-signal FPGAs contains ADCs and DACs with analog signal conditional blocks allowing them to operate as a system-on-chip (SoC)

XILINX FPGA :

Consists of

- Programmable Logic blocks -to implement combinational and sequential logic
- Programmable Interconnect- wires to connect inputs and outputs to logic blocks
- Programmable I/O blocks-special logic blocks at periphery of device for external connections



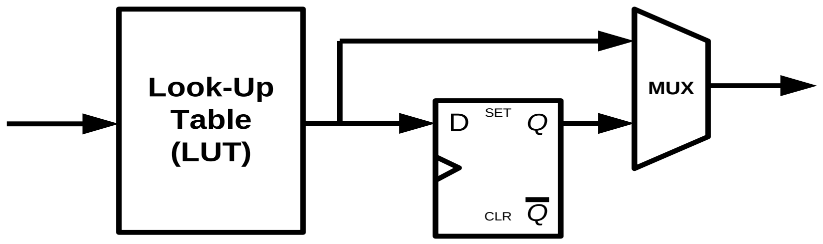
Configurable Logic Blocks (CLBs)

Two 4-input function generators (F and G) offer unrestricted versatility. Most combinatorial logic functions need four or fewer inputs. However, a third function generator (H) is provided. The H function generator has three inputs. Either zero, one, or both of these inputs can be the outputs of F and G; the other input(s) are from outside the CLB. The CLB can, therefore, implement certain functions of up to nine variables, like parity check or expandable-identity comparison of two sets of four inputs.

Function Generators: Four independent inputs are provided to each of two func-tion generators (F1 - F4 and G1 - G4). These function generators, with outputs labeled F' and G', are each capable of implementing any arbitrarily defined Boolean function of four inputs. The function generators are implemented as memory look-up tables. The propagation delay is therefore independent of the function implemented. A third function generator, labeled H', can implement any Boolean function of its three inputs. Two of these inputs can optionally be the F' and G' functional generator outputs. Alternatively, one or both of these inputs can come from outside the CLB (H2, H0). The third input must come from outside the block (H1). Signals from the function generators can exit the CLB on two outputs. F' or H' can be connected to the X output. G' or H' can be connected to the Y output. A CLB can be used to implement any of the following functions:

- any function of up to four variables, plus any second function of up to four unrelated variables, plus any third function of up to three unrelated variables
- any single function of five variables
- any function of four variables together with some functions of six variables
- some functions of up to nine variables

Implementing wide functions in a single block reduces both the number of blocks required and the delay in the signal path, achieving both increased capacity and speed. The versatility of the CLB function generators significantly improves system speed. In addition, the design-software tools can deal with each function generator independently. This flexibility improves cell usage.



Flip-Flops: The CLB can pass the combinatorial output(s) to the interconnect network, but can also store the combinatorial results or other incoming data in one or two flip-flops, and connect their outputs to the interconnect network as well. The two edge-triggered D-type flip-flops have common clock (K) and clock enable (EC) inputs. Either or both clock inputs can also be permanently enabled.

Latches: (XC4000EX only) The CLB storage elements can also be configured as latches. The two latches have common clock (K) and clock enable (EC) inputs.

Clock Input: Each flip-flop can be triggered on either the rising or falling clock edge. The clock pin is shared by both storage elements. However, the clock is individually invertible for each storage element. Any inverter placed on the clock input is automatically absorbed into the CLB.

Clock Enable: The clock enable signal (EC) is active High. The EC pin is shared by both storage elements. If left unconnected for either, the clock enable for that storage element defaults to the active state. EC is not invertible within the CLB.

Set/Reset: An asynchronous storage element input (SR) can be con-figured as either set or reset. This configuration option determines the state in which each flip-flop becomes operational after configuration. It also determines the effect of a Global Set/Reset pulse during normal operation, and the effect of a pulse on the SR pin of the CLB. All three set/ reset functions for any single flip-flop are controlled by the same configuration data bit.

Global Set/Reset: A separate Global Set/Reset line (not shown in Figure 7) sets or clears each storage element during power-up, reconfiguration, or when a dedicated Reset net is driven active. This global net (GSR) does not compete with other routing resources; it uses a dedicated distribution network.

Data Inputs and Outputs: The source of a storage element data input is programmable. It is driven by any of the functions F', G', and H', or by the Direct_In (DIN) block input. The flip-flops or latches drive the XQ and YQ CLB outputs. Two fast feed-through paths are available, as shown in Figure 1.7. A two-to-one multiplexer on each of the XQ and YQ outputs selects between a storage element output and any of the control inputs. This bypass is sometimes used by the automated router to repower internal signals.
Control Signals: Multiplexers in the CLB map the four control inputs (C1 - C4 in Figure 1.7) into the four internal control signals (H1, DIN/H2, SR/H0, and EC). Any of these inputs can drive any of the four internal control signals. When the logic function is enabled, the four inputs are:

- EC - Enable Clock
- SR/H0 - Asynchronous Set/Reset or H function generator Input 0
- DIN/H2 - Direct In or H function generator Input 2
- H1 - H function generator Input 1

When the memory function is enabled, the four inputs are:

- EC - Enable Clock
- WE - Write Enable
- D0 - Data Input to F and/or G function generator
- D1 - Data input to G function generator (16x1 and 16x2 modes) or 5th Address bit (32x1 mode)

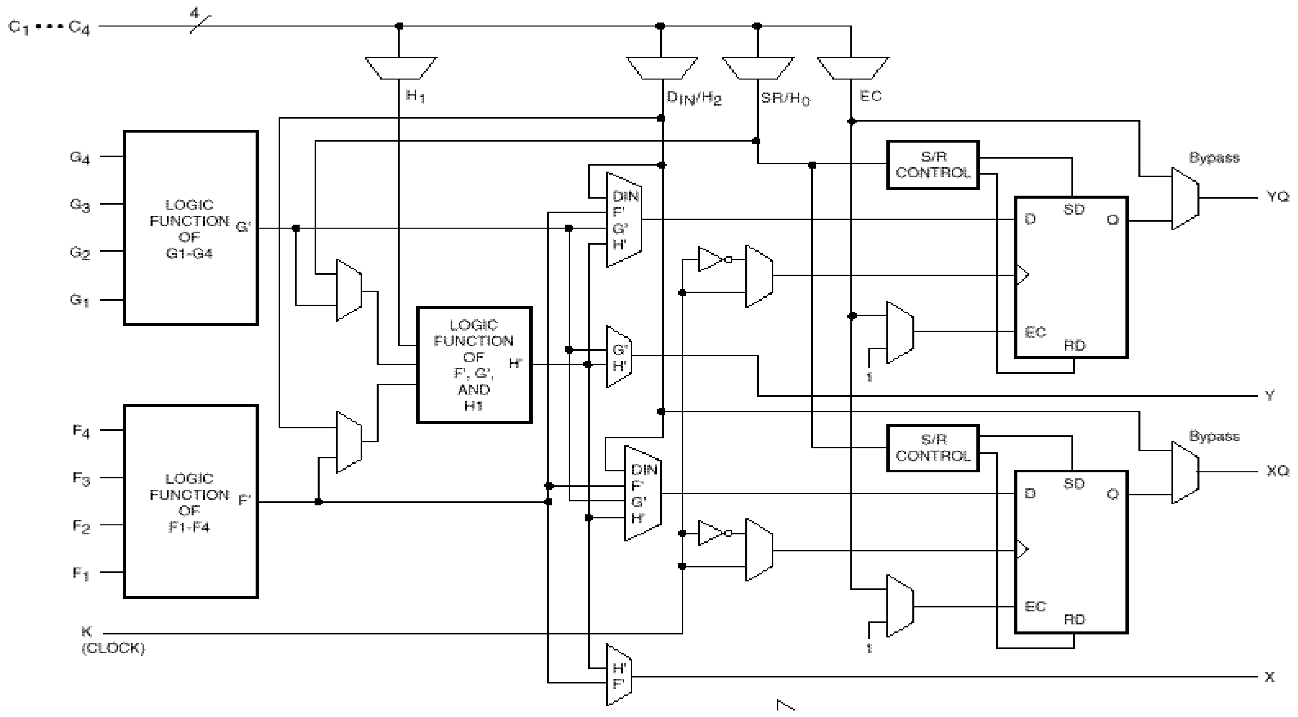


Fig: Xilinx XC4000 CLB

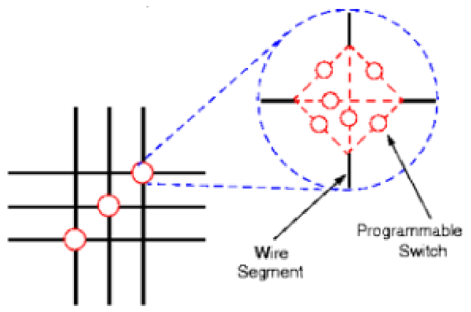
Programmable Interconnects

- All Programmable_Interconnects_ share the common property: Configurable in one of the two positions – 'ON' or 'OFF'

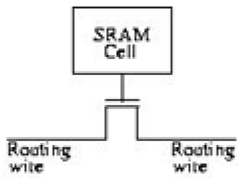
➢ Can be classified into three categories:
  - SRAM based
  - Fuse based

   - EPROM/EEPROM/Flash based

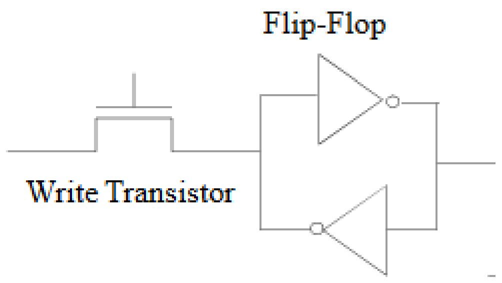
- Desired properties:
   - Minimum area consumption
   - Low on resistance; High off resistance
   - Low parasitic capacitance to the attached wire
   - Reliability in volume production



SRAM Based



- Cross coupled inverters (Flip-Flop) is used to store the switch status (4 Transistors)
- Write Transistor to write Configuration status
- Total: 6 Transistors
- Volatile
- Needs an external storage
- Needs a power-on configuration mechanism
- In-circuit re-programmable
- Lesser configuration time
- Occupies relatively larger area



Flash Transistor/EPROM/EEPROM Based

• MOS transistor with a floating gate
• Conducts when not programmed off
• Can be electrically programmed ‘off’ or ‘on’

EPROM Programming Technology:

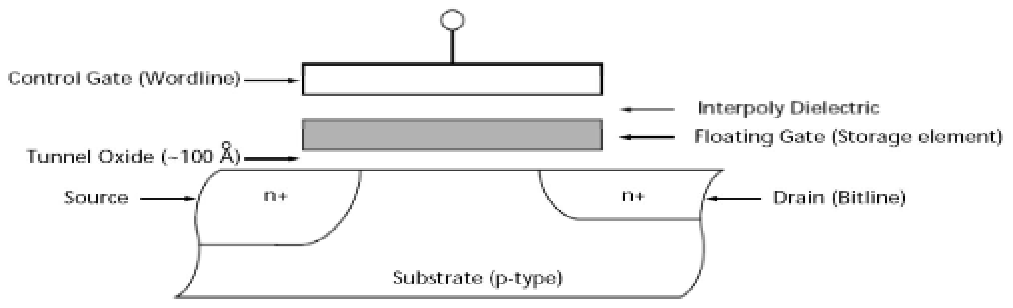
- Two gates: Floating and Select
- Normal mode:
  - No charge on floating gate
  - Transistor behaves as normal n-channel transistor
- Floating gate charged by applying high voltage
  - Threshold of transistor (as seen by gate) increases
  - Transistor turned off permanently

- Re-programmable by exposing to UV radiation
- No external storage mechanism
- Re-programmable (Not all!)
- Not in-system re-programmable
- Re-programming is a time consuming task

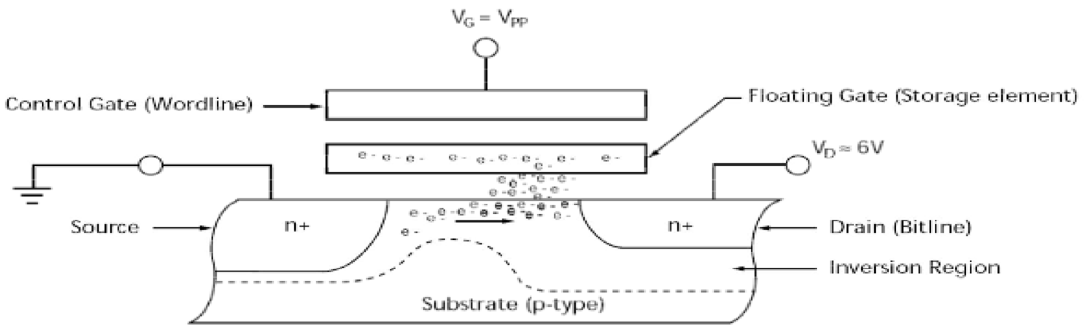
EEPROM Programming Technology:

- Two gates: Floating and Select
- Functionally equivalent to EPROM; Construction and structure differ
- Electrically Erasable: Re-programmable by applying high voltage
  - (No UV radiation expose!)
- Re-programmable; In general, in-system re-programmable
- Re-programming consumes lesser time compared to EPROM technology
- Multiple voltage sources may be required
- Area occupied is twice that of EPROM!

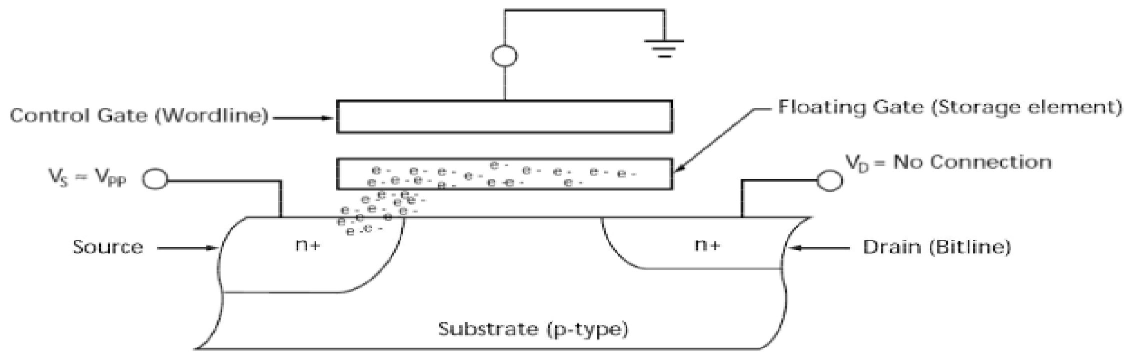
Structure



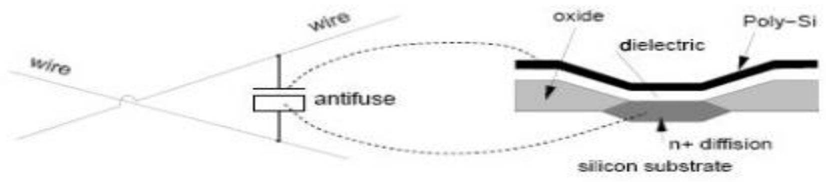
While programmed for OFF



To reprogramme:



Antifuse Based

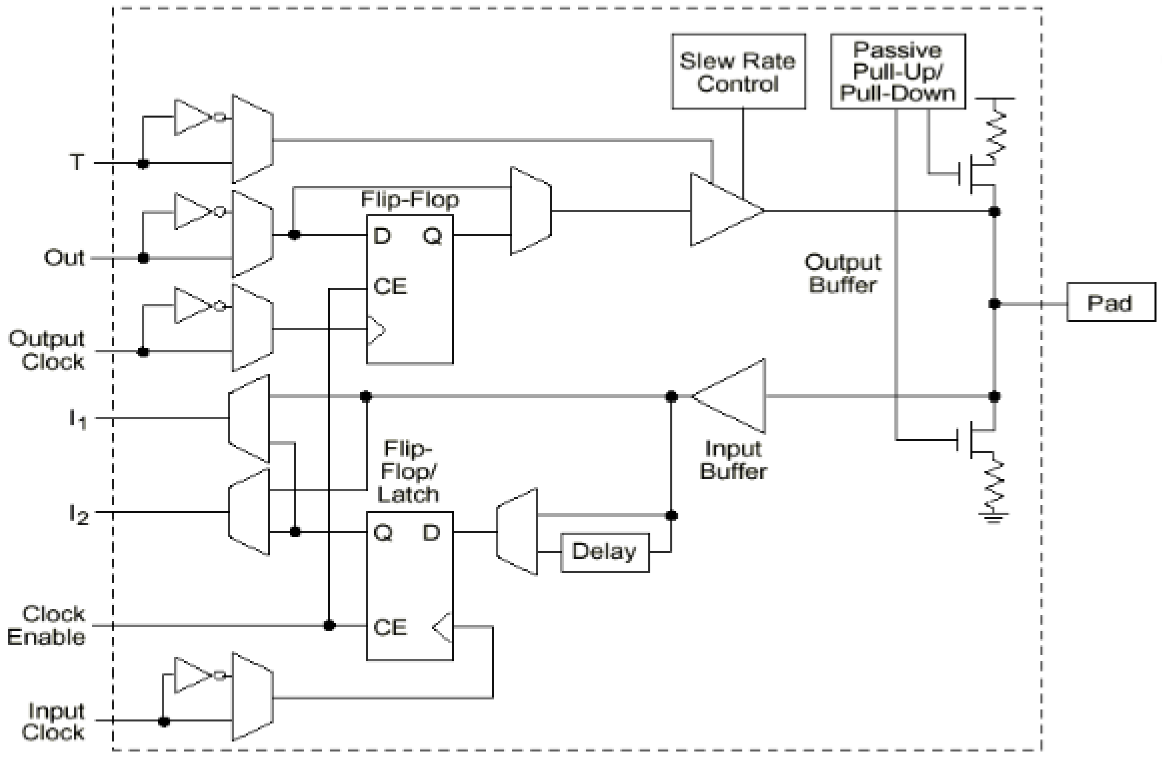


- all anti-fuse programming elements
  - Uses materials which normally resides in high impedance state
  - But can be fused irreversibly into low impedance state by applying high voltage
- Very low ON Resistance (Faster implementation of circuits)
- Limited size of anti-fuse elements; Interconnects occupy relatively lesser area
  - Offset : Larger transistors needed for programming
- One Time Programmable
  - Cannot be re-programmed
    - (Design changes are not possible)
  - Retain configuration after power off

Name	Volatile	Re-programm-able	Delay	Area
Flash	No	In-circuit	Large	Medium
SRAM	Yes	In-circuit	Large	Large
Anti-fuse	No	No	Small	Small

Programmable IOB

User-configurable input/output blocks (IOBs) provide the interface between external package pins and the internal logic. Each IOB controls one package pin and can be con-figured for input, output, or bidirectional signals. Figure shows a simplified block diagram of the XC4000E IOB.



Input Signals: Two paths, labeled I1 and I2 in Figure, bring input signals into the array. Inputs also connect to an input register that can be programmed as either an edge-triggered flip-flop or a level-sensitive latch.
Variations with inverted clocks are available, and some combinations of latches and flip-flops can be implemented in a single IOB.
The inputs can be globally configured for either TTL (1.2V, default) or CMOS thresholds, using an option in the Make-Bits program. There is a slight hysteresis of about 300mV.
The output levels are also configurable; the two global adjustments of input threshold and output level are independent. Inputs of the low-voltage devices must be configured as CMOS at all times.
Registered Inputs: The I1 and I2 signals that exit the block can each carry either the direct or registered input signal. The input and output storage elements in each IOB have a common clock enable input, which, through configuration, can be activated individually for the input or output flip-flop, or both. This clock enable operates exactly like the EC pin on the XC4000-Series CLB. It cannot be inverted within the IOB.
Optional Delay: The data input to the register can optionally be delayed by several nanoseconds.
Output Signals: Output signals can be optionally inverted within the IOB, and can pass directly to the pad or be stored in an edge-triggered flip-flop.
An active-High 3-state signal can be used to place the out-put buffer in a high-impedance state, implementing 3-state outputs or bidirectional I/O. Under configuration control, the output (OUT) and output 3-state (T) signals can be inverted. The polarity of these signals is independently configured for each IOB.

FPGA ROUTING ARCHITECTURE

All internal connections are composed of metal segments with programmable switching points and switching matrices to implement the desired routing. A structured, hierarchical matrix of routing resources is provided to achieve efficient automated routing.
There are several types of interconnect:

- CLB routing is associated with each row and column of the CLB array.
- IOB routing forms a ring (called a VersaRing) around the outside of the CLB array. It connects the I/O with the internal logic blocks.
- Global routing consists of dedicated networks primarily designed to distribute clocks throughout the device with minimum delay and skew. Global routing can also be used for other high-fanout signals.

Five interconnect types are distinguished by the relative length of their segments: single-length lines, double-length lines, quad and octal lines (XC4000EX only), and longlines.

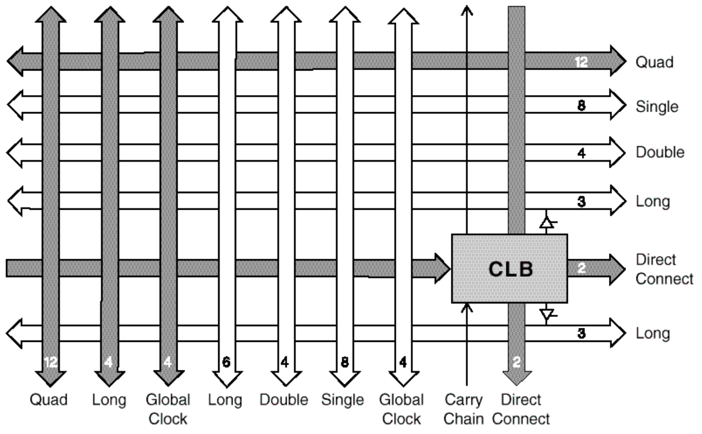
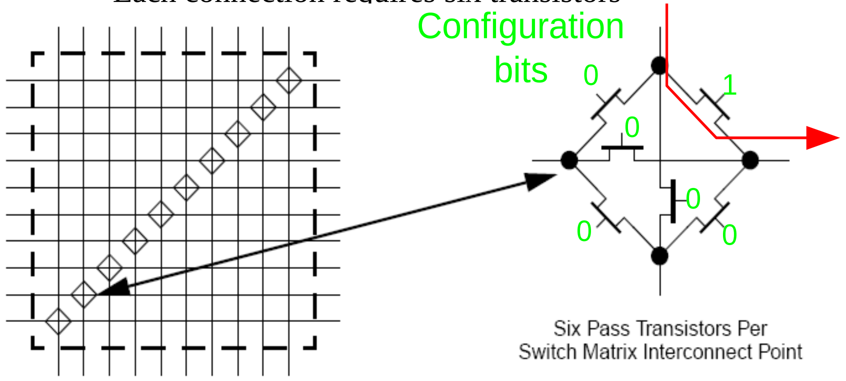
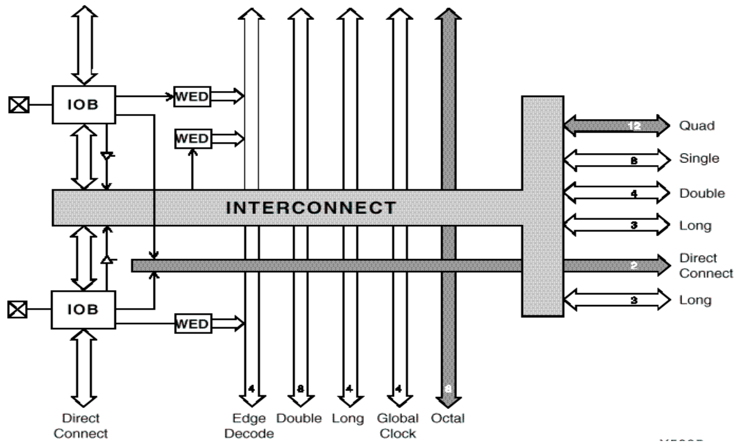


Fig:CLB Routing Connections

- Wire segments
  - Single length lines
  - Spans single CLB
  - Connects adjacent CLBs
  - Used to connect signals that do not have critical timing requirements
- Double length lines
  - Spans two CLBs
  - Uses half as much switch as a single length connection
  - Long lines
  - Low skew; Used for signals such as clock
  - Relatively rare resource
- Switch Matrix
  - Every line is connected to lines on the other three direction
  - Each connection requires six transistors

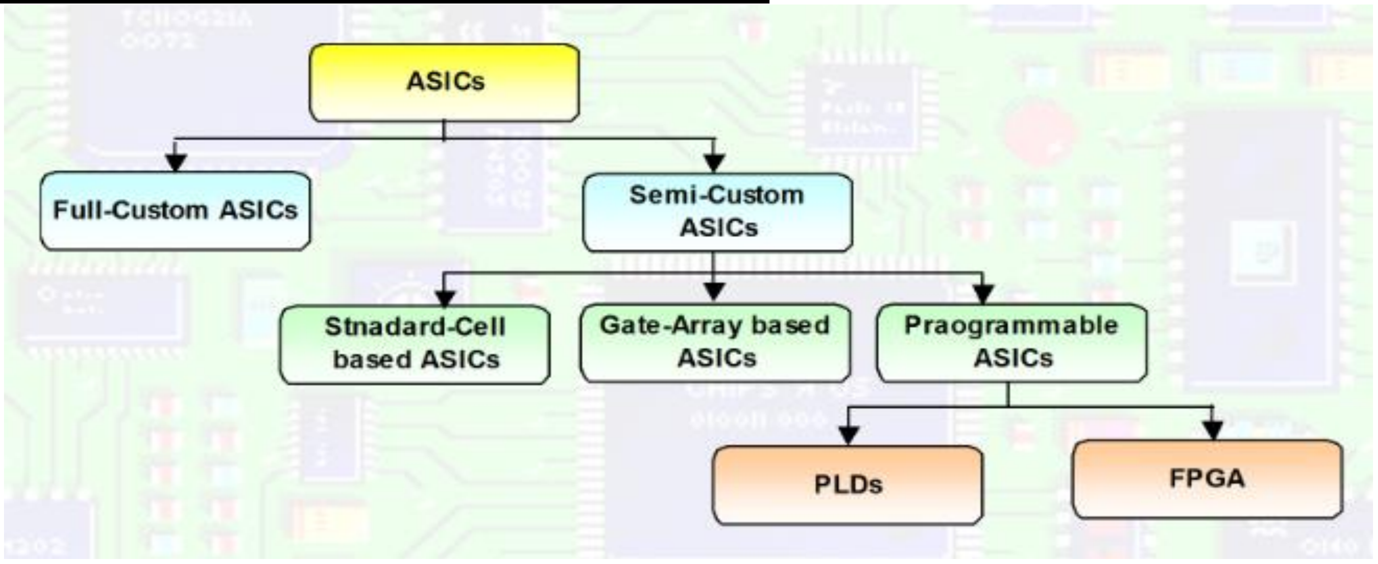


XC4000-Series devices have additional routing around the IOB ring. This routing is called a VersaRing. The VersaRing facilitates pin-swapping and redesign without affecting board layout. Included are eight double-length lines spanning two CLBs (four IOBs), and four longlines. Global lines and Wide Edge Decoder lines are provided.
XC4000EX devices also include eight octal lines. A high-level diagram of the VersaRing is shown in Figure below. The shaded arrows represent routing present only in XC4000EX devices.



I/O Routing

APPLICATION SPECIFIC INTEGRATED CIRCUITS



Classification of ASIC:
An ASIC is classified into
1. Full customASIC
2. Semi custom ASIC

FULL Custom ASIC

Full custom includes all possible logic cells and mask layers that are customized These are very expensive to manufacture and design - Example is microprocessor.
In full custom ASIC an engineer design some or all logic cells circuits, or layout specifically for one ASIC

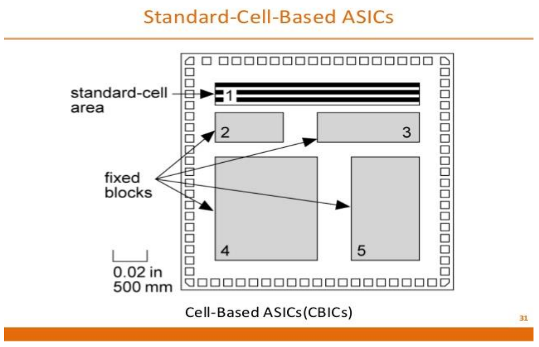
Advantages:
Offer highest performance
lowest cost (smallest die size)
Disadvantages:
Increased design time
Increased Complexity
Higher design cost
Higher risk.
Some Examples:
Microporcessor,
High-Voltage Automobile Control Chips
Analog - Digital Communication Chips
Sensors and Actuator

Semi Custom ASIC

In semicustom ASIC all the logic cells are predesigned and some of the mask layers are customized The types of semicustom ASIC are
1.Standard cell based ASIC
2. Gate array basedASiC

Standard cell basedASIC:

A cell based ASIC or cell based IC(CBIC) uses predesigned logic cells like AND gates, OR gates, multiplexers, Flip flops. The predefined logic cells are known as standard cells The standard cell areas are called flexible blocks The flexible blocks used in combination with larger predesigned cells, like micro controllers and micro processors, these are called mega cells. Wiring cells in Standard Cell based ASICs are
Feedthrough cell
•Piece of metal that is used to pass a signal through a cell or to a space in a cell waiting to be used as a feedthrough
Spacer cells
•The width of each row of standard cells is adjusted so that they may be aligned using spacer cells
Row end cells
•The power buses, or rails, are then connected to additional vertical power rails using row - end cells at the aligned ends of each standard - cell block
Power cells
• If the rows of standard cells are long, then vertical power rails can also be
run in metal 2 through the cell rows using special power cells that just connect to VDD and GND
•Usually the designer manually controls the number and width of the vertical
power rails connected to the standard - cell blocks during physical design



.
fig.cell based ASIC img

Advantages:
Less cost
Less time
Reduced Risk
Transistor operates at maximum speed.
Disadvantages
Expense of designing standard cell library is high
Time needed to fabricate all layers for each new design is high

Gate array based ASIC
Gate array (GA) based ASIC has predefined transistors on the silicon wafer. The predefined pattern of transistors on a gate array is the base array. The base array is made up of a smallest element called primitive cell To distinguish this type of gate array from other types of gate array this is often called MASKED GATEARRAY (MGA) MACROS the logic cells in a gate array library are called macro The types of MGA or gate array based ASIC are
1. Channeled gate array
2. Channel less gate array
3. Structured gate array

Channeled Gate Array :
A Channeled gate array has space between the rows of transistor for wiring.
Features
1. Only the interconnect is customized
2. Interconnect uses predefined spaces between rows of base cells
3. Manufacturing lead time is between two days and two weeks
Adv:
Specific space for interconnection
Disadv:
compared to CBIC space is not adjustable

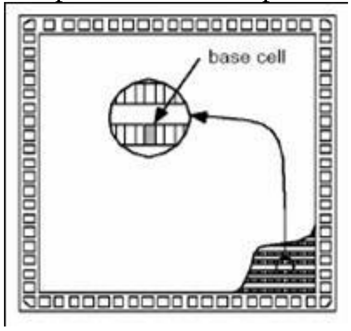


Fig: Channeled Gate Array

Channel less Gate Array :
It is also known as channel free gate array The routing on a channel less gate array uses rows of unused transistors
Features:
1. Top few mask layers are customized interconnect.
2. Manufacturing lead time is between two days and two weeks
Adv :
•Logic density is higher for channelless gate array
•Contact layers are customized

Disadv:
•No specific area for routing
•Rows of transistors used for routing are not used for other purpose

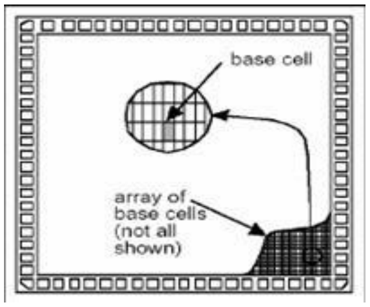
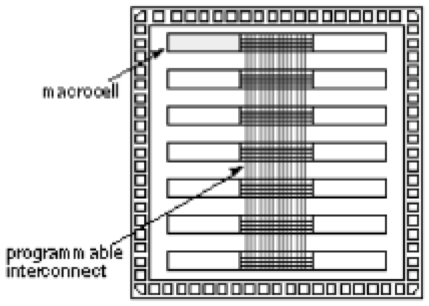


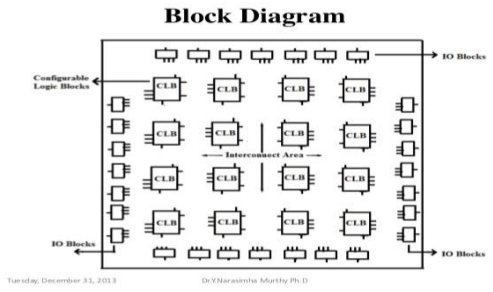
Fig: Channel less Gate Array

Structured Gate Array:
It can be either channeled or channel less, but it includes custom block.
It is also known as master slice or master image.
This embedded area either contains a different base cell that is more suitable for building memory cells. Features:
1. only the interconnect is customized
2. Custom blocks can be embedded
3. Manufacturing lead time is between two days and two weeks
Advantages
1. Improved area efficiency
2. Increased performance
3. Lower cost
4. Faster turnaround
Disadvantages
Embedded function is fixed

Programmable ASIC
In which all the all the logic cells are predesigned and none of the mask layers are customized
The two types are I. Programmable logic device 2. Field programmable gate array
Programmable logic device: (PLD)
Programmable logic devices are standard IC and available in standard configuration.PLD may be configured or programmed
Features
1. No customized mask layers or logic cells
2. Fast design turn around
3. Single large block of programmable interconnect
4. Matrix of large macro cells



Field programmable gate array: (FPGA)
Complex PLD's are called FPGA.
FPGA are growing rapidly and replace TTL in microelectronic system
Characteristics:
1. No mask layers are customized.
2. Programming basic logic cells and interconnects.
3. Core with regular array of programmable basic logic cells that implement combinational and sequential logic.
4. Matrix of programmable interconnect surrounds the basic logic cells.
5. Programmable l/O cells surround the core
6. Design turnaround is few hours.


Block Diagram
IO Blocks
Configurable Logic Blocks
CLB
Interconnect Area
IO Blocks
IO Blocks
Tuesday, December 31, 2013
Dr.Y.Narasimha Murthy Ph.D

DESIGN FOR TESTABILITY

Introduction

The embedded system is an information processing system that consists of hardware and software components. Nowadays, the number of embedded computing systems in areas such as telecommunications, automotive electronics, office automation, and military applications are steadily growing. This market expansion arises from greater memory densities as well as improvements in embeddable processor cores, intellectual-property modules, and sensing technologies. At the same time, these improvements have increased the amount of software needed to manage the hardware components, leading to a higher level of system complexity. Designers can no longer develop high-performance systems from scratch but must use sophisticated system modeling tools. The increased complexity of embedded systems and the reduced access to internal nodes has made it not only more difficult to diagnose and locate faulty components, but also the functions of embedded components may be difficult to measure. Creating testable designs is key to developing complex hardware and/or software systems that function reliably throughout their operational life. Testability can be defined with respect to a fault. A fault is testable if there exists a well-specified procedure (e.g., test pattern generation, evaluation, and application) to expose it, and the procedure is implementable with a reasonable cost using current technologies. Testability of the fault therefore represents the inverse of the cost in detecting the fault. A circuit is testable with respect to a fault set when each and every fault in this set is testable.

Design-for-testability techniques improve the controllability and observability of internal nodes, so that embedded functions can be tested. Two basic properties determine the testability of a node:

1) controllability, which is a measure of the difficulty of setting internal circuit nodes to 0 or 1 by assigning values to primary inputs (PIs), and

2) observability, which is a measure of the difficulty of propagating a node's value to a primary output (PO).

A node is said to be testable if it is easily controlled and observed. For sequential circuits, some have added predictability, which represents the ability to obtain known output values in response to given input stimuli. The factors affecting predictability include initializability, races, hazards, oscillations, etc.

DFT techniques include analog test busses and scan methods. Testability can also be improved with BIST circuitry, where signal generators and analysis circuitry are implemented on chip. Without testability, design flaws may escape detection until a product is in the hands of users; equally, operational failures may prove difficult to detect and diagnose.

Increased embedded system complexity makes thorough assessment of system integrity by testing external black-box behavior almost impossible. System complexity also complicates test equipment and procedures.

Design for testability should increase a system's testability, resulting in improved quality while reducing time to market and test costs. Designers must ensure a testable, finished design regardless of implementation decisions. Supporting hardware-software codesign' requires "cotesting" techniques, which draw hardware and software test techniques together into a cohesive whole.

Design for Testability Techniques

Design for testability (DFT) refers to those design techniques that make the task of subsequent testing easier. There is definitely no single methodology that solves all embedded system-testing problems. There also is no single DFT technique, which is effective for all kinds of circuits. DFT techniques can largely be divided into two categories, i.e., ad hoc techniques and structured (systematic) techniques. DFT methods for digital circuits:

- Ad-hoc methods
- Structured methods:
    - Scan
    - Partial Scan
    - Built-in self-test
    - Boundary scan

<u>Ad-hoc DFT methods</u>

Good design practices learnt through experience are used as guidelines for ad-hoc DFT. Some important guidelines are given below.

Things to be followed

- Large circuits should be partitioned into smaller sub-circuits to reduce test costs. One of the most important steps in designing a testable chip is to first partition the chip in an appropriate way such that for each functional module there is an effective (DFT) technique to test it. Partitioning must be done at every level of the design process, from architecture to circuit, whether testing is considered or not. Partitioning can be functional (according to functional module boundaries) or physical (based on circuit topology). Partitioning can be done by using multiplexers and/or scan chains.
- Test access points must be inserted to enhance controllability & observability of the circuit. Test points include control points (CPs) and observation points (OPs). The CPs

are active test points, while the OPs are passive ones. There are also test points, which are both CPs and OPs. Before exercising test through test points that are not PIs and POs, one should investigate into additional requirements on the test points raised by the use of test equipments.

- Circuits (flip-flops) must be easily initializable to enhance predictability. A power-on reset mechanism controllable from primary inputs is the most effective and widely used approach.
- Test control must be provided for difficult-to-control signals.
- Automatic Test Equipment (ATE) requirements such as pin limitation, tri-stating, timing resolution, speed, memory depth, driving capability, analog/mixed-signal support, internal/boundary scan support, etc., should be considered during the design process to avoid delay of the project and unnecessary investment on the equipments.
- Internal oscillators, PLLs and clocks should be disabled during test. To guarantee tester synchronization, internal oscillator and clock generator circuitry should be isolated during the test of the functional circuitry. The internal oscillators and clocks should also be tested separately.
- Analog and digital circuits should be kept physically separate. Analog circuit testing is very much different from digital circuit testing. Testing for analog circuits refers to real measurement, since analog signals are continuous (as opposed to discrete or logic signals in digital circuits). They require different test equipments and different test methodologies. Therefore they should be tested separately.

Things to be avoided

- Asynchronous(unclocked) logic feedback in the circuit must be avoided. A feedback in the combinational logic can give rise to oscillation for certain inputs. Since no clocking is employed, timing is continuous instead of discrete, which makes tester synchronization virtually impossible, and therefore only functional test by application board can be used.
- Monostables and self-resetting logic should be avoided. A monostable (one-shot) multivibrator produces a pulse of constant duration in response to the rising or falling transition of the trigger input. Its pulse duration is usually controlled externally by a resistor and a capacitor (with current technology, they also can be integrated on chip). One-shots are used mainly for 1) pulse shaping, 2) switch-on delays, 3) switch-off delays, 4) signal delays. Since it is not controlled by clocks, synchronization and precise duration control are very difficult, which in turn reduces testability by ATE. Counters and dividers are better candidates for delay control.
- Redundant gates must be avoided.
- High fanin/fanout combinations must be avoided as large fan-in makes the inputs of the gate difficult to observe and makes the gate output difficult to control.
- Gated clocks should be avoided. These degrade the controllability of circuit nodes.

The above guidelines are from experienced practitioners. These are not complete or universal. In fact, there are drawbacks for these methods:

i. There is a lack of experts and tools.
ii. Test generation is often manual
iii. This method cannot guarantee for high fault coverage.
iv. It may increase design iterations.
v. This is not suitable for large circuits

Scan Design Approaches for DFT

Objectives of Scan Design

1. Scan design is implemented to provide controllability and observability of internal state variables for testing a circuit.
2. It is also effective for circuit partitioning.
3. A scan design with full controllability and observability turns the sequential test problem into a combinational one.

Scan Design Requirements

1. Circuit is designed using pre-specified design rules.
2. Test structure (hardware) is added to the verified design.
    i. One (or more) test control (TC) pin at the primary input is required.
    ii. Flip-flops are replaced by scan flip-flops (SFF) and are connected so that they behave as a shift register in the test mode. The output of one SFF is connected to the input of next SFF. The input of the first flip-flop in the chain is directly connected to an input pin (denoted as SCANIn), and the output of the last flipflop is directly connected to an output pin (denoted as SCANOUT). In this way, all the flip-flops can be loaded with a known value, and their value can be easily accessed by shifting out the chain. Figure shows a typical circuit after the scan insertion operation.
    iii. Input/output of each scan shift register must be available on PI/PO.
3. Combinational ATPG is used to obtain tests for all testable faults in the combinational logic.
4. Shift register tests are applied and ATPG tests are converted into scan sequences for use in manufacturing test.

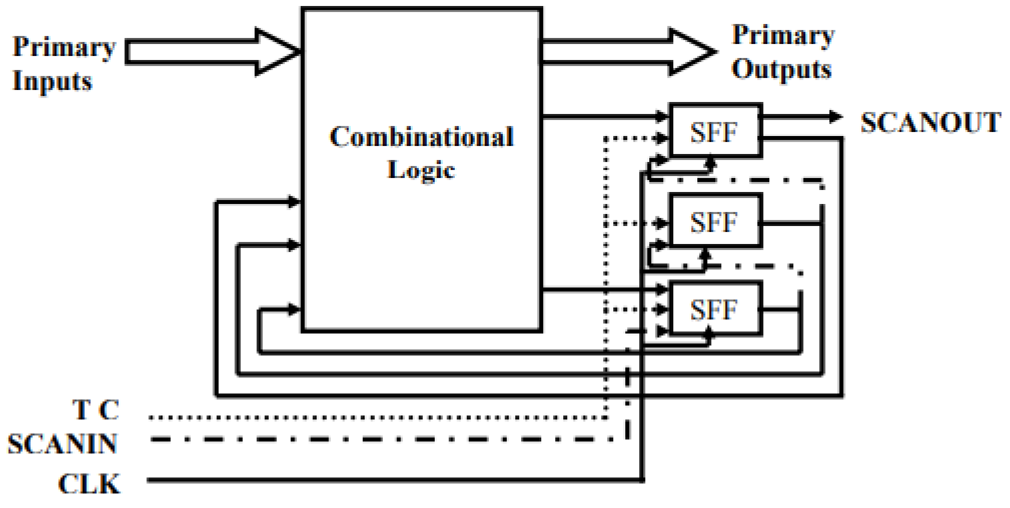


Fig shows a scan structure connected to design. The scan flip-flips (FFs) must be interconnected in a particular way. This approach effectively turns the sequential testing problem into a combinational one and can be fully tested by compact ATPG patterns. Unfortunately, there are two types of overheads associated with this technique that the designers care about very much. These are the hardware overhead (including three extra pins, multiplexers for all FFs, and extra routing area) and performance overhead (including multiplexer delay and FF delay due to extra load).

Scan Design Rules

- ➢ Only clocked D-type master-slave flip-flops for all state variables should be used.
- ➢ At least one PI pin must be available for test. It is better if more pins are available.
- ➢ All clock inputs to flip-flops must be controlled from primary inputs (PIs). There will be no gated clock. This is necessary for FFs to function as a scan register.
- ➢ Clocks must not feed data inputs of flip-flops. A violation of this can lead to a race condition in the normal mode.

Scan Variations

There have been many variations of scan as listed below, few of these are discussed here.

i. MUXed Scan
ii. Scan path
iii. Scan-Hold Flip-Flop
iv. Serial scan
v. Level-Sensitive Scan Design (LSSD)
vi. Scan set
vii. Random access scan

MUX Scan

It was invented at Stanford in 1973 by M. Williams & Angell. ƒ In this approach a MUX is inserted in front of each FF to be placed in the scan chain.

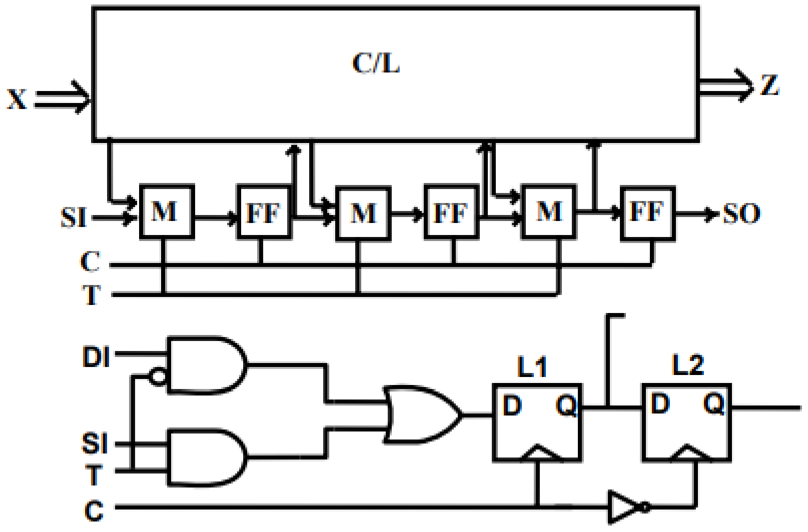


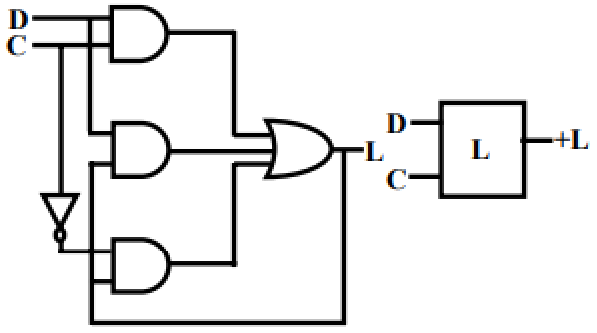
Fig. shows that when the test mode pin T=0, the circuit is in normal operation mode and when T=1, it is in test mode (or shift-register mode). ƒ The scan flip-flips (FFs) must be interconnected in a particular way. This approach effectively turns the sequential testing problem into a combinational one and can be fully tested by compact ATPG patterns. ƒ There are two types of overheads associated with this method. The hardware overhead due to three extra pins, multiplexers for all FFs, and extra routing area. The performance overhead includes multiplexer delay and FF delay due to extra load

Level-Sensitive Scan Design (LSSD)

This approach was introduced by Eichelberger and T. Williams in 1977 and 1978.

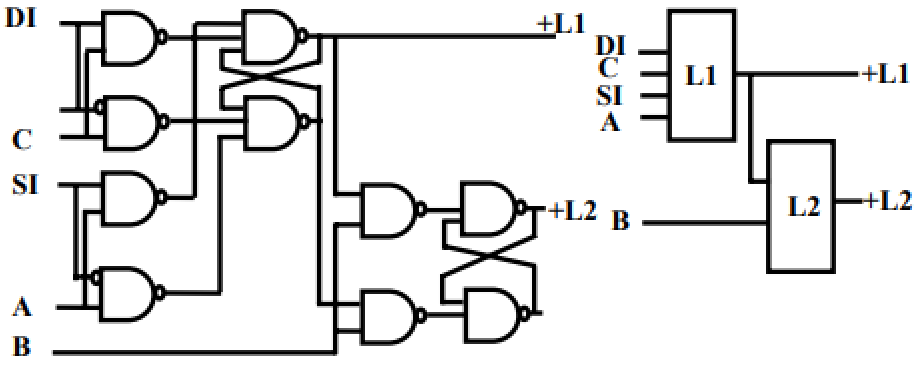
It is a latch-based design used at IBM.

It guarantees race-free and hazard-free system operation as well as testing. It is insensitive to component timing variations such as rise time, fall time, and delay. It is faster and has a lower hardware complexity than SR modification.
It uses two latches (one for normal operation and one for scan) and three clocks. Furthermore, to enjoy the luxury of race-free and hazard-free system operation and test, the designer has to follow a set of complicated design rules.
A logic circuit is level sensitive (LS) iff the steady state response to any allowed input change is independent of the delays within the circuit. Also, the response is independent of the order in which the inputs change.



C D	+L
0 0	L
0 1	L
1 0	0
1 1	1

Fig. 39.4 A polarity-hold latch



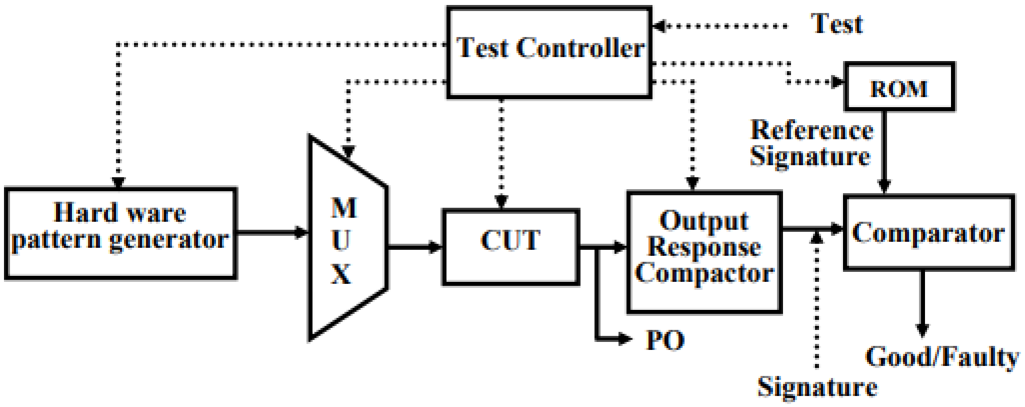
LSSD requires that the circuit be LS, so we need LS memory elements as defined above. Figure shows an LS polarity-hold latch. The correct change of the latch output (L) is not dependent on the rise/fall time of C, but only on C being `1' for a period of time greater than or equal to data propagation and stabilization time. Figure 39.5 shows the polarity-hold shift-register latch (SRL) used in LSSD as the scan cell. The scan cell is controlled in the following way:
• Normal mode: A=B=0, C=0 → 1.
• SR (test) mode: C=0, AB=10→ 01 to shift SI through L1 and L2.
Advantages of LSSD
1. Correct operation independent of AC characteristics is guaranteed. 2. FSM is reduced to combinational logic as far as testing is concerned. 3. Hazards and races are eliminated, which simplifies test generation and fault simulation.
Drawbacks of LSSD
1. Complex design rules are imposed on designers. There is no freedom to vary from the overall schemes. It increases the design complexity and hardware costs (4-20% more hardware and 4 extra pins).
2. Asynchronous designs are not allowed in this approach.

3. Sequential routing of latches can introduce irregular structures.
4. Faults changing combinational function to sequential one may cause trouble, e.g., bridging and CMOS stuck-open faults.
5. Test application becomes a slow process, and normal-speed testing of the entire test sequence is impossible.
6. It is not good for memory intensive designs.

BUILT-IN-SELF-TEST (BIST)

Introduction

BIST is a design-for-testability technique that places the testing functions physically with the circuit under test (CUT), as illustrated in Figure. The basic BIST architecture requires the addition of three hardware blocks to a digital circuit: a test pattern generator, a response analyzer, and a test controller. The test pattern generator generates the test patterns for the CUT. Examples of pattern generators are a ROM with stored patterns, a counter, and a linear feedback shift register (LFSR). A typical response analyzer is a comparator with stored responses or an LFSR used as a signature analyzer. It compacts and analyzes the test responses to determine correctness of the CUT. A test control block is necessary to activate the test and analyze the responses. However, in general, several test-related functions can be executed through a test controller circuit.



As shown in Figure, the wires from primary inputs (PIs) to MUX and wires from circuit output to primary outputs (POs) cannot be tested by BIST. In normal operation, the CUT receives its inputs from other modules and performs the function for which it was designed. During test mode, a test pattern generator circuit applies a sequence of test patterns to the CUT, and the test responses are evaluated by a output response compactor. In the most common type of BIST, test responses are compacted in output response compactor to form (fault) signatures. The response signatures are compared with reference golden signatures generated or stored onchip, and the error signal indicates whether chip is good or faulty.

Four primary parameters must be considered in developing a BIST methodology for embedded systems; these correspond with the design parameters for on-line testing techniques

- Fault coverage: This is the fraction of faults of interest that can be exposed by the test patterns produced by pattern generator and detected by output response monitor. In presence of input bit stream errors there is a chance that the computed signature matches the golden signature, and the circuit is reported as fault free. This undesirable property is called masking or aliasing.
- Test set size: This is the number of test patterns produced by the test generator, and is closely linked to fault coverage: generally, large test sets imply high fault coverage. *f*

- Hardware overhead: The extra hardware required for BIST is considered to be overhead. In most embedded systems, high hardware overhead is not acceptable.
- Performance overhead: This refers to the impact of BIST hardware on normal circuit performance such as its worst-case (critical) path delays. Overhead of this type is sometimes more important than hardware overhead.

Issues for BIST

i. Area Overhead: Additional active area due to test controller, pattern generator, response evaluator and testing of BIST hardware.
ii. Pin Overhead: At least 1 additional pin is needed to activate BIST operation. Input MUX adds extra pin overheads.
iii. Performance overhead: Extra path delays are added due to BIST.
iv. Yield loss increases due to increased chip area.
v. Design effort and time increases due to design BIST.
vi. The BIST hardware complexity increases when the BIST hardware is made testable.

Benefits of BIST

i. It reduces testing and maintenance cost, as it requires simpler and less expensive ATE.
ii. BIST significantly reduces cost of automatic test pattern generation (ATPG).
iii. It reduces storage and maintenance of test patterns.
iv. It can test many units in parallel.
v. It takes shorter test application times.
vi. It can test at functional system speed.

BIST can be used for non-concurrent, on-line testing of the logic and memory parts of a system. It can readily be configured for event-triggered testing, in which case, the BIST control can be tied to the system reset so that testing occurs during system start-up or shut down.

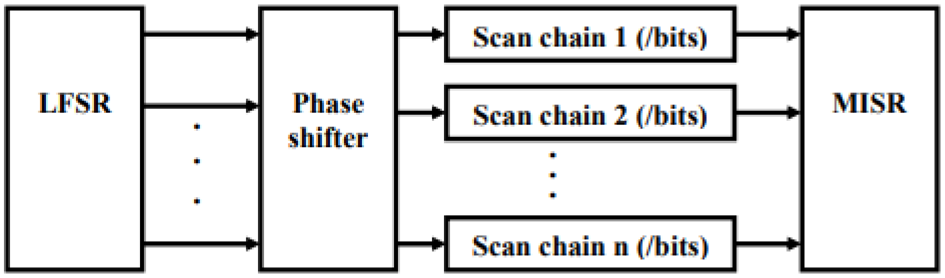
BIST can also be designed for periodic testing with low fault latency. This requires incorporating a testing process into the CUT that guarantees the detection of all target faults within a fixed time.

On-line BIST is usually implemented with the twin goals of complete fault coverage and low fault latency. Hence, the test generation (TG) and response monitor (RM) are generally designed to guarantee coverage of specific fault models, minimum hardware overhead, and reasonable set size. These goals are met by different techniques in different parts of the system.

TG and RM are often implemented by simple, counter-like circuits, especially linear-feedback shift registers (LFSRs). The LFSR is simply a shift register formed from standard flip-flops, with the outputs of selected flip-flops being fed back (modulo-2) to the shift register’s inputs. When used as a TG, an LFSR is set to cycle rapidly through a large number of its states. These states, whose choice and order depend on the design parameters of the LFSR, define the test patterns. In this mode of operation, an LFSR is seen as a source of (pseudo) random tests that are, in principle, applicable to any fault and circuit types.

An LFSR can also serve as an RM by counting (in a special sense) the responses produced by the tests. An LFSR RM’s final contents after applying a sequence of test responses forms a fault signature, which can be compared to a known or generated good signature, to see if a fault is present. Ensuring that the fault coverage is sufficiently high and the number of tests is sufficiently low are the main problems with random BIST methods. Two general approaches have been proposed to preserve the cost advantages of LFSRs while making the generated test sequence much shorter. Test points can be inserted in the CUT to improve controllability and observability; however, they can also result in performance loss. Alternatively, some determinism can be introduced into the generated test sequence, for example, by inserting specific “seed” tests that are known to detect hard faults.

A typical BIST architecture using LFSR is shown in Figure. Since the output patterns of the LFSR are time-shifted and repeated, they become correlated; this reduces the effectiveness of the fault detection. Therefore a phase shifter (a network of XOR gates) is often used to decorrelate the output patterns of the LFSR. The response of the CUT is usually compacted by a multiple input shift register (MISR) to a small signature, which is compared with a known fault free signature to determine whether the CUT is faulty.



BIST Test Pattern Generation Techniques

- Stored patterns

An automatic test pattern generation (ATPG) and fault simulation technique is used to generate the test patterns. A good test pattern set is stored in a ROM on the chip. When BIST is activated, test patterns are applied to the CUT and the responses are compared with the corresponding stored patterns. Although stored-pattern BIST can provide excellent fault coverage, it has limited applicability due to its high area overhead.

- Exhaustive patterns

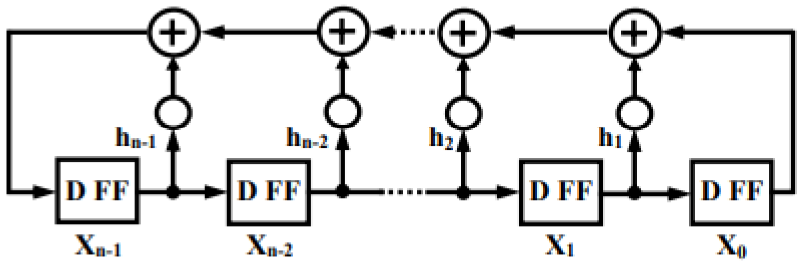
Exhaustive pattern BIST eliminates the test generation process and has very high fault coverage. To test an n-input block of combinational logic, it applies all possible 2n -input patterns to the block. Even with high clock speeds, the time required to apply the patterns may make exhaustive pattern BIST impractical for a circuit with n>20.

- Pseudo-exhaustive patterns

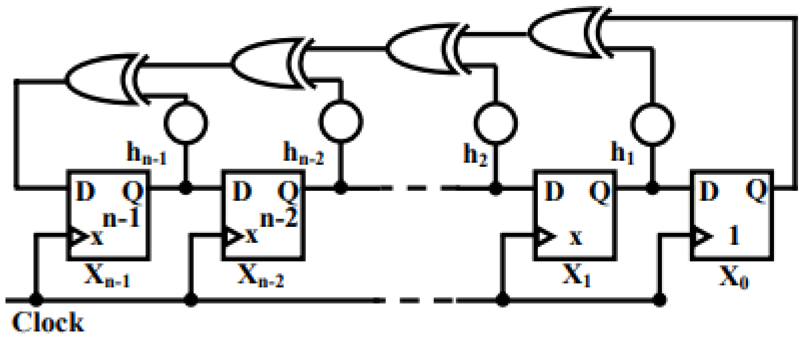
In pseudo-exhaustive pattern generation, the circuit is partitioned into several smaller subcircuits based on the output cones of influence, possibly overlapping blocks with fewer than n inputs. Then all possible test patterns are exhaustively applied to each sub-circuit. The main goal of pseudo-exhaustive test is to obtain the same fault coverage as the exhaustive testing and, at the same time, minimize the testing time. Since close to 100% fault coverage is guaranteed, there is no need for fault simulation for exhaustive testing and pseudo-exhaustive testing. However, such a method requires extra design effort to partition the circuits into pseudo-exhaustive testable sub-circuits. Moreover, the delivery of test patterns and test responses is also a major consideration. The added hardware may also increase the overhead and decrease the performance.

Pseudo-Random Pattern Generation

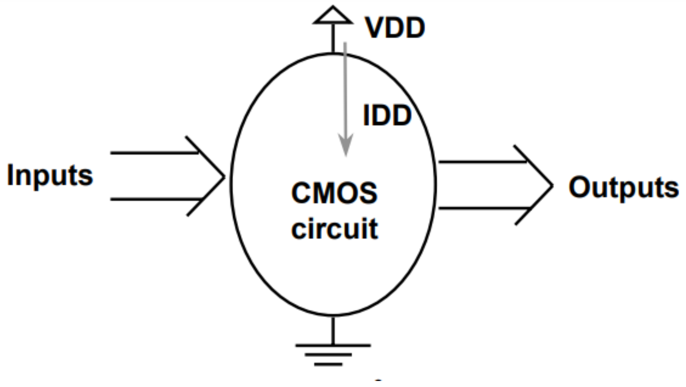
A string of 0's and 1's is called a pseudo-random binary sequence when the bits appear to be random in the local sense, but they are in someway repeatable. The linear feedback shift register (LFSR) pattern generator is most commonly used for pseudo-random pattern generation. In general, this requires more patterns than deterministic ATPG, but less than the exhaustive test. In contrast with other methods, pseudo-random pattern BIST may require a long test time and necessitate evaluation of fault coverage by fault simulation. This pattern type, however, has the potential for lower hardware and performance overheads and less design effort than the preceding methods. In pseudorandom test patterns, each bit has an approximately equal probability of being a 0 or a 1. The number of patterns applied is typically of the order of 103 to 107 and is related to the circuit's testability and the fault coverage required. Linear feedback shift register reseeding is an example of a BIST technique that is based on controlling the LFSR state. LFSR reseeding may be static, that is LFSR stops generating patterns while loading seeds, or dynamic, that is, test generation and seed loading can proceed simultaneously. The length of the seed can be either equal to the size of the LFSR (full reseeding) or less than the LFSR (partial reseeding). In a dynamic reseeding technique that allows partial reseeding is proposed to encode test vectors. A set of linear equations is solved to obtain the seeds, and test vectors are ordered to facilitate the solution of this set of linear equations.



Above Figure shows a standard, external exclusive-OR linear feedback shift register. There are n flip-flops (Xn-1,……X0) and this is called n-stage LFSR. It can be a near-exhaustive test pattern generator as it cycles through 2n -1 states excluding all 0 states. This is known as a maximal length LFSR. Below Figure shows the implementation of a n-stage LFSR with actual digital circuit.



IDDQ TESTING
IDD --- Current flow through VDD
Q --- Quiescent state
 IDDQ Testing --- Detecting faults by monitoring IDDQ



Normal IDDQ: ~10-9Amp. Abnormal IDDQ: >10-5Amp.

Advantages of IDDQ Testing

i. Fault effect is easy to detect
ii. Many realistic faults are detectable
iii. ATPG is relatively simple
iv. Test length is shorter
v. Built-in current sensing is possible

Faults That Can Be Detected

• Bridge & stuck-on faults
• Break faults
— Line break
— Gate break
— Drain break
— Source break
• Transistor stuck-open faults
• Other faults

Circuit Constraints

To ensure IDDQ detectability, two conditions must be satisfied:
1. Normal IDDQ must be small
2. Faults must result in large IDDQ

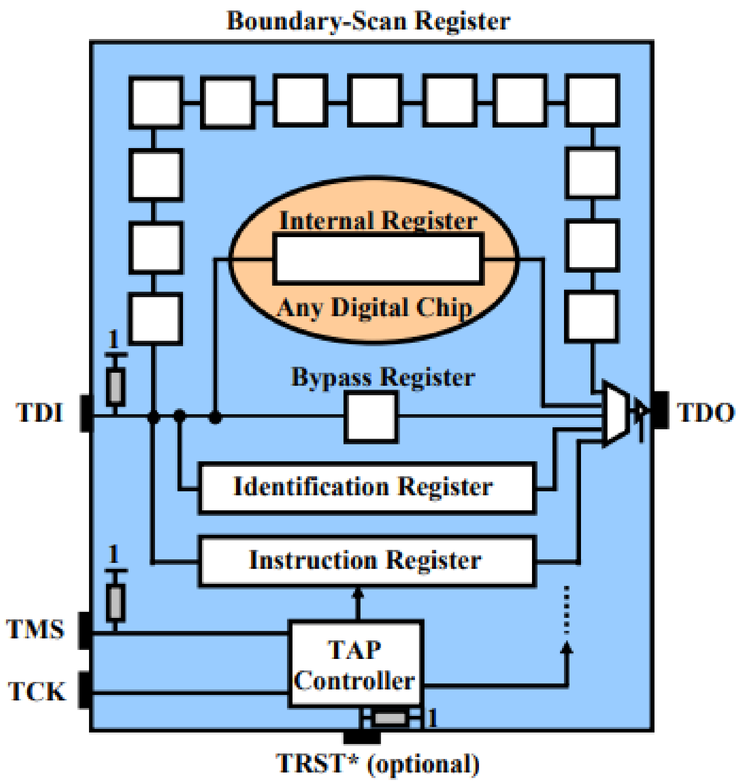
BOUNDARY SCAN

Boundary Scan is a family of test methodologies aiming at resolving many test problems: from chip level to system level, from logic cores to interconnects between cores, and from digital circuits to analog or mixed-mode circuits. It is now widely accepted in industry and has been considered as an industry standard in most large IC system designs. Boundary-scan, as defined by the IEEE Std. 1149.1 standard [1-3], is an integrated method for testing interconnects on printed circuit board that is implemented at the IC level.

Boundary Scan Architecture

The boundary-scan test architecture provides a means to test interconnects between integrated circuits on a board without using physical test probes. It adds a boundary-scan cell that includes a multiplexer and latches, to each pin on the device. Figure [1] illustrates the main elements of a universal boundary-scan device. The Figure shows the following elements:
• Test Access Port (TAP) with a set of four dedicated test pins: Test Data In (TDI), Test Mode Select (TMS), Test Clock (TCK), Test Data Out (TDO) and one optional test pin Test Reset (TRST*).
• A boundary-scan cell on each device primary input and primary output pin, connected internally to form a serial boundary-scan register (Boundary Scan).
• A TAP controller with inputs TCK, TMS, and TRST*.
• An n-bit (n >= 2) instruction register holding the current instruction.
• A 1-bit Bypass register (Bypass).
• An optional 32-bit Identification register capable of being loaded with a permanent device identification code.
The test access ports (TAP), which define the bus protocol of boundary scan, are the additional I/O pins needed for each chip employing Std.1149.1a. The TAP controller is a 16-state final state machine that controls each step of the operations of boundary scan. Each instruction to be carried out by the boundary scan architecture is stored in the Instruction Register. The various control signals associated with the instruction are then provided by a decoder. Several Test Data Registers are used to stored test data or some system related information such as the chip ID, company name, etc.

1149.1 Chip Architecture



Bus Protocol

The Test Access Ports (TAPs) are genral purpose ports and provide access to the test function of the IC between the application circuit and the chip's I/O pads. It includes four mandatory pins TCK, TDI, TDO and TMS and one optional pin TRST* as described below. All TAP inputs and outputs shall be dedicated connections to the component (i.e., the pins used shall not be used for any other purpose).

• Test Clock Input (TCK): a clock independent of the system clock for the chip so that test operations can be synchronized between the various parts of a chip. It also synchronizes the operations between the various chips on a printed circuit board. As a convention, the test instructions and data are loaded from system input pins on the rising edge of TCK and driven through system output pins on its falling edge. TCK is pulsed by the equipment controlling the test and not by the tested device. It can be pulsed at any frequency (up to a maximum of some MHz). It can be even pulsed at varying rates.

• Test Data Input (TDI): an input line to allow the test instruction and test data to be loaded into the instruction register and the various test data registers, respectively.

• Test Data Output (TDO): an output line used to serially output the data from the JTAG registers to the equipment controlling the test.

• Test Mode Selector (TMS): the test control input to the TAP controller. It controls the transitions of the test interface state machine. The test operations are controlled by the sequence of 1s and 0s applied to this input. Usually this is the most important input that has to be controlled by external testers or the on-board test controller. Test Reset Input (TRST*): The optional TRST* pin is used to initialize the TAP controller, that is, if the TRST* pin is used, then the TAP controller can be asynchronously reset to a TestLogic-Reset state when a 0 is applied at TRST*. This pin can also be used to reset the circuit under test, however it is not recommended for this application.

Boundary Scan Cell

The IEEE Std. 1149.1a specifies the design of four test data registers as shown in Figure 2. Two mandatory test data registers, the bypass and the boundary-scan resisters, must be included in any boundary scan architecture. The boundary scan register, though may be a little confusing by its name, refers to the collection of the boundary scan cells. The other registers, such as the device identification register and the design-specific test data registers, can be added optionally.

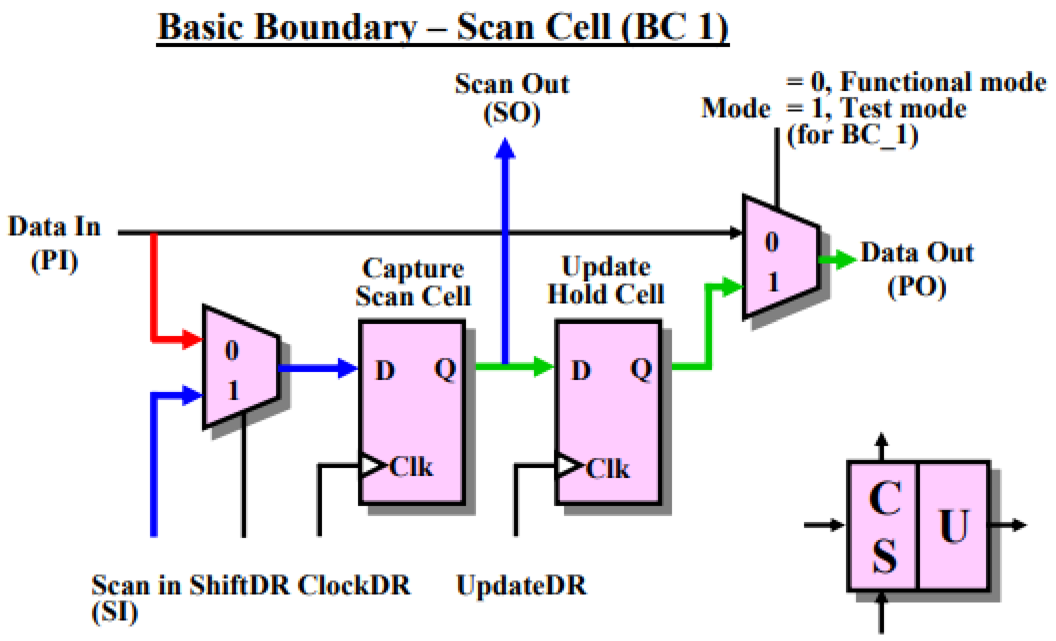


Figure 3 shows a basic universal boundary-scan cell, known as a BC_1. The cell has four modes of operation: normal, update, capture, and serial shift. The memory elements are two D type flip-flops with front-end and back-end multiplexing of data. It is important to note that the circuit shown in Figure 3 is only an example of how the requirement defined in the Standard could be realized. The IEEE 1149.1 Standard does not mandate the design of the circuit, only its functional specification. The four modes of operation are as follows:

1) During normal mode also called serial mode, Data_In is passed straight through to Data_Out.

2) During update mode, the content of the Update Hold cell is passed through to Data_Out. Signal values already present in the output scan cells to be passed out through the device output pins. Signal values already present in the input scan cells will be passed into the internal logic.

3) During capture mode, the Data_In signal is routed to the input Capture Scan cell and the value is captured by the next ClockDR. ClockDR is a derivative of TCK. Signal values on device input pins to be loaded into input cells, and signal values passing from the internal logic to device output pins to be loaded into output cells

4) During shift mode, the Scan_Out of one Capture Scan cell is passed to the Scan_In of the next Capture Scan cell via a hard-wired path. The Test ClocK, TCK, is fed in via yet another dedicated device input pin and the various modes of operation are controlled by a dedicated Test Mode Select (TMS) serial control signal. Note that both capture and shift operations do not interfere with the normal passing of data from the parallel-in terminal to the parallel-out terminal. This allows on the fly capture of operational values and the shifting out of these values for inspection without interference. This application of the boundary-scan register has tremendous potential for real-time monitoring of the operational status of a system — a sort of electronic camera taking snapshots — and is one reason why TCK is kept separate from any system clocks.

TAP Controller

The operation of the test interface is controlled by the Test Access Port (TAP) controller. This is a 16-state finite state-machine whose state transitions are controller by the TMS signal; the state transition diagram is shown in Figure 41.7. The TAP controller can change state only at the rising edge of TCK and the next state is determined by the logic level of TMS. In other words, the state transition in Figure follows the edge with label 1 when the TMS line is set to 1, otherwise the edge with label 0 is followed. The output signals of the TAP controller corresponding to a subset of the labels associated with the various states. As shown in Figure , the TAP consists of four mandatory terminals plus one optional terminal.

The main functions of the TAP controller are:

• To reset the boundary scan architecture,
• To select the output of instruction or test data to shift out to TDO,
• To provide control signals to load instructions into Instruction Register,
• To provide signals to shift test data from TDI and test response to TDO, and
• To provide signals to perform test functions such as capture and application of test data.

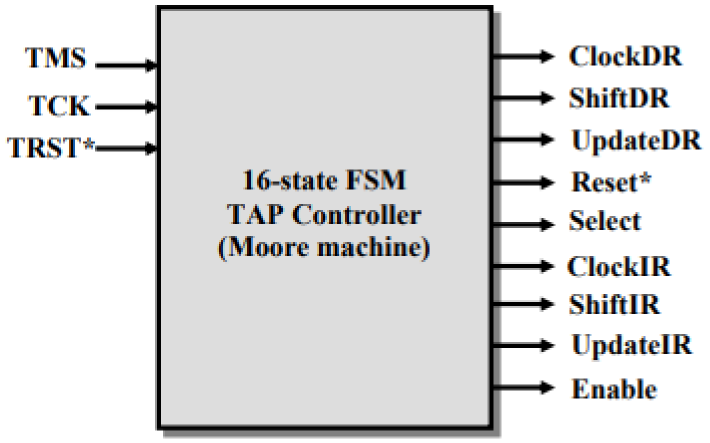
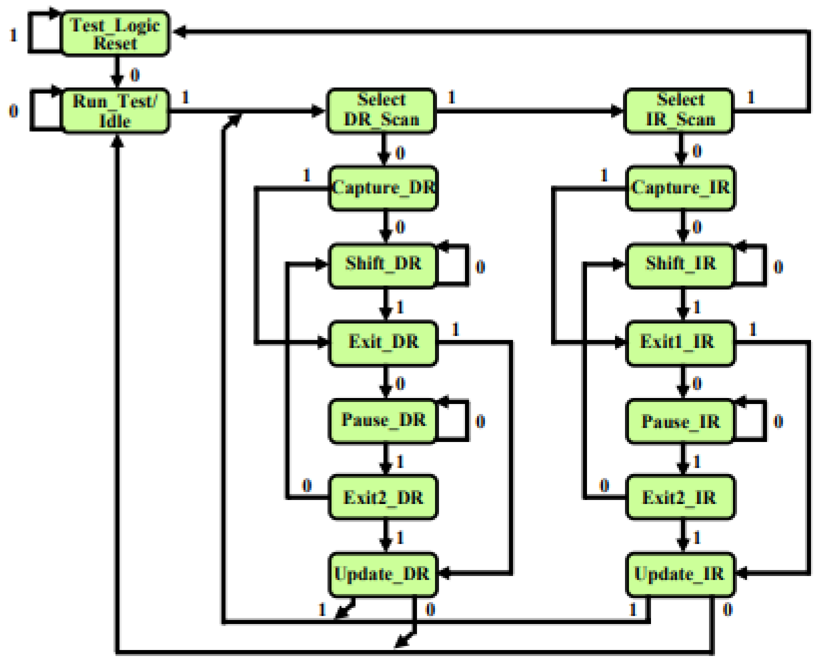


Figure shows a top-level view of TAP Controller. TMS and TCK (and the optional TRST*) go to a 16-state finite-state machine controller, which produces the various control signals. These signals include dedicated signals to the Instruction register (ClockIR, ShiftIR, UpdateIR) and generic signals to all data registers (ClockDR, ShiftDR, UpdateDR). The data register that actually responds is the one enabled by the conditional control signals generated at the parallel outputs of the Instruction register, according to the particular instruction. The other signals, Reset, Select and Enable are distributed as follows:

• Reset is distributed to the Instruction register and to the target Data Register
• Select is distributed to the output multiplexer
• Enable is distributed to the output driver amplifier It must be noted that the Standard uses the term Data Register to mean any target register except the Instruction register.

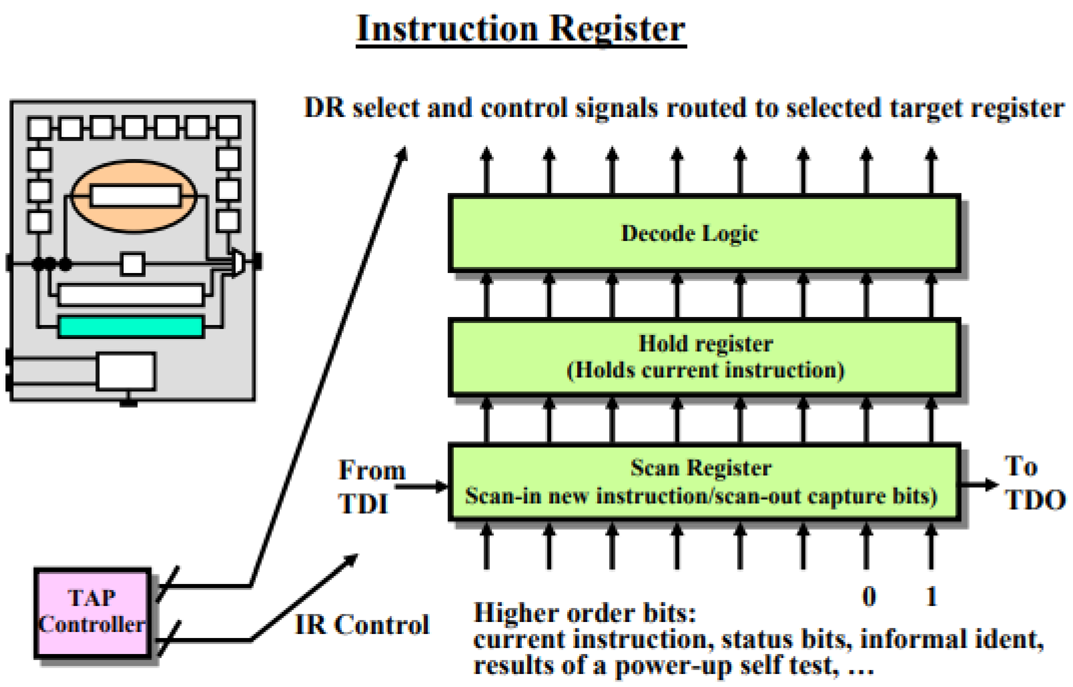
Below figure shows the 16-state state table for the TAP controller. The value on the state transition arcs is the value of TMS. A state transition occurs on the positive edge of TCK and the controller output values change on the negative edge of TCK. The 16 states can be divided into three parts. The first part contains the reset and idle states, the second and third parts control the operations of the data and instruction registers, respectively. Since the only difference between the second and the third parts are on the registers they deal with, in the following only the states in the first and second parts are described. Similar description on the second part can be applied to the third part.



1. **Test-Logic-Reset**: In this state, the boundary scan circuitry is disabled and the system is in its normal function. Whenever a Reset* signal is applied to the BS circuit, it also goes back to this state. One should also notice that whatever state the TAP controller is at, it will goes back to this state if 5 consecutive 1's are applied through TMS to the TAP controller.
2. **Run-Test/Idle**: This is a state at which the boundary scan circuitry is waiting for some test operations such as BIST operations to complete. One typical example is that if a BIST operation requires 216 cycles to complete, then after setting up the initial condition for the BIST operation, the TAP controller will go back to this state and wait for 216 cycles before it starts to shift out the test results.
3. **Select-DR-Scan**: This is a temporary state to allow the test data sequence for the selected test-data register to be initiated. Capture_DR Shift_DR Exit_DR Pause_DR Exit2_DR Update_DR 1 1 1 1 1 1 1 0 0 0 0 0 0 0 Select IR_Scan Capture_IR Shift_IR Exit1_IR Pause_IR Exit2_IR Update_IR 1 1 1 1 1 1 0 0 0 0 0 0 0 Fig. 41.7 State transition diagram of TAP controller
4. **Capture-DR**: In this state, data can be loaded in parallel to the data registers selected by the current instruction.
5**. Shift-DR**: In this state, test data are scanned in series through the data registers selected by the current instruction. The TAP controller may stay at this state as long as TMS=0. For each clock cycle, one data bit is shifted into (out of) the selected data register through TDI (TDO).
6. **Exit-DR**: All parallel-loaded (from the Capture-DR state) or shifted (from the Shift-DR state) data are held in the selected data register in this state.
7. **Pause-DR**: The BS pauses its function here to wait for some external operations. For example, when a long test data is to be loaded to the chip(s) under test, the external tester may need to reload the data from time to time. The Pause-DR is a state that allows the boundary scan architecture to wait for more data to shift in.
8. **Exit2-DR**: This state represents the end of the Pause-DR operation, allows the TAP controller to go back to ShiftDR state for more data to shift in.
9. **Update-DR**: The test data stored in the first stage of boundary scan cells is loaded to the second stage in this state.

Instruction Register

As shown in Figure, an Instruction register has a shift scan section that can be connected between TDI and TDO, and a hold section that holds the current instruction. There may be some decoding logic beyond the hold section depending on the width of the register and the number of different instructions. The control signals to the Instruction register originate from the TAP controller and either cause a shift-in/shift-out through the Instruction register shift section, or cause the contents of the shift section to be passed across to the hold section (parallel Update operation). It is also possible to load (Capture) internal hard-wired values into the shift section of the Instruction register. The Instruction register must be at least two-bits long to allow coding of the four mandatory instructions — Extest, Bypass, Sample, Preload — but the maximum length of the Instruction register is not defined. In capture mode, the two least significant bits must capture a 01 pattern. (Note: by convention, the least-significant bit of any register connected between the device TDI and TDO pins, is always the bit closest to TDO.) The values captured into higher-order bits of the Instruction register are not defined in the Standard. One possible use of these higher-order bits is to capture an informal identification code if the optional 32-bit Identification register is not implemented. In practice, the only mandated bits for the Instruction register capture is the 01 pattern in the two least-significant bits.


Instruction Register
DR select and control signals routed to selected target register
Decode Logic
Hold register
(Holds current instruction)
Scan Register
Scan-in new instruction/scan-out capture bits)
From TDI
To TDO
TAP Controller
IR Control
0
1
Higher order bits:
current instruction, status bits, informal ident,
results of a power-up self test, …

Design for manufacturability

Circuits can be optimized for manufacturability to increase their yield. This can be done in a number of different ways.

Physical

At the physical level (i.e., mask level), the yield and hence manufacturability can be improved by reducing the effect of process defects. The design rules for particular processes will frequently have guidelines for improving yield. The following list is representative:

Increase the spacing between wires where possible—this reduces the chance of a defect causing a short circuit.

Increase the overlap of layers around contacts and vias—this reduces the chance that a misalignment will cause an aberration in the contact structure. Increase the number of vias at wire intersections beyond one if possible—this reduces the chance of a defect causing an open circuit. Increasingly, design tools are dealing with these kinds of optimizations automatically.

Redundancy

Redundant structures can be used to compensate for defective components on a chip. For example, memory arrays are commonly built with extra rows. During manufacturing test, if one of the words is found to be defective, the memory can be reconfigured to access the spare row instead. Laser-cut wires or electrically programmable fuses can be used for configuration. Similarly, if the memory has many banks and one or more are found to be defective, they can be disabled, possibly even under software control.

Power

Elevated power can cause failure due to excess current in wires, which in turn can cause metal migration failures. In addition, high-power devices raise the die temperature, degrading device performance and, over time, causing device parameter shifts. The method of dealing with this component of manufacturability is to minimize power through design techniques. In addition, a suitable package and heat sink should be chosen to remove excess heat.

Process Spread

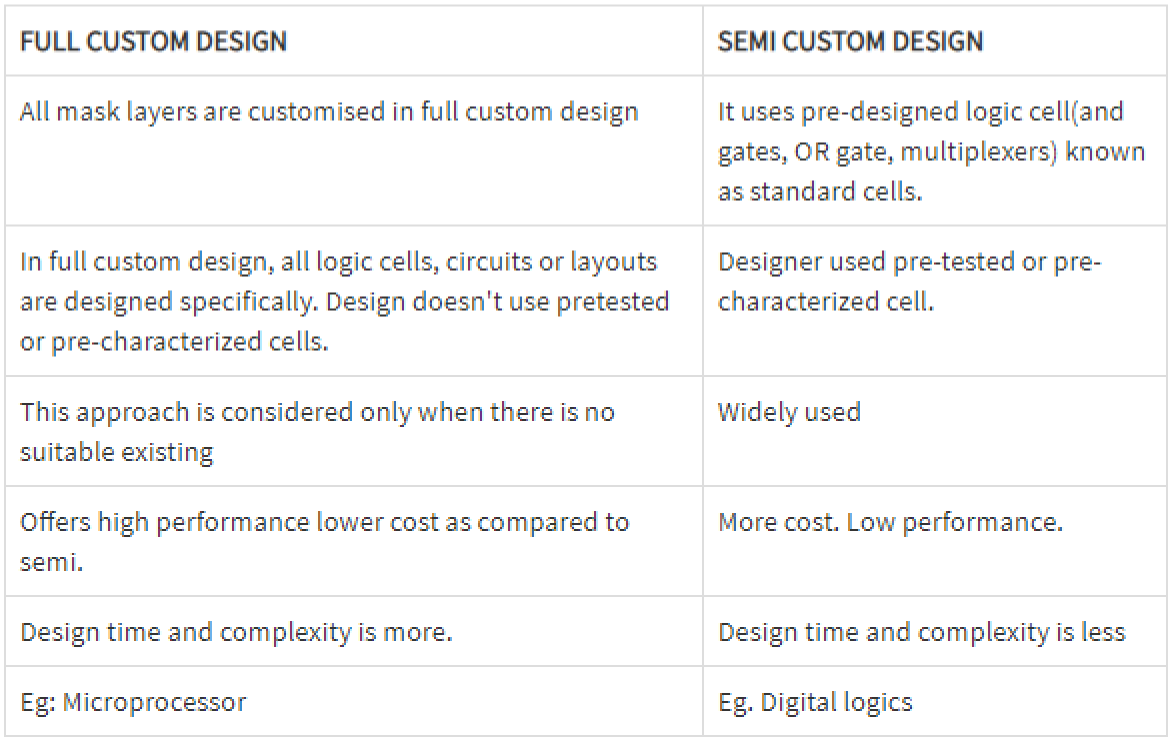
We have seen that process simulations can be carried out at different process corners. Monte Carlo analysis can provide better modeling for process spread and can help with centering a design within the process variations.

Yield Analysis

When a chip has poor yield or will be manufactured in high volume, dice that fail manufacturing test can be taken to a laboratory for yield analysis to locate the root cause of the failure. If particular structures are determined to have caused many of the failures, the layout of the structures can be redesigned. For example, during volume production ramp-up for the Pentium microprocessor, the silicide over long thin polysilicon lines was found to crack and raise the wire resistance [Needham98]. This in turn led to slower-than-expected operation for the cracked chips. The layout was modified to widen polysilicon wires or strap them with metal wherever possible, boosting the yield at higher frequencies.

PART A

1. **Differentiate full custom and semi custom design? A/M 19 ,N/D 18**

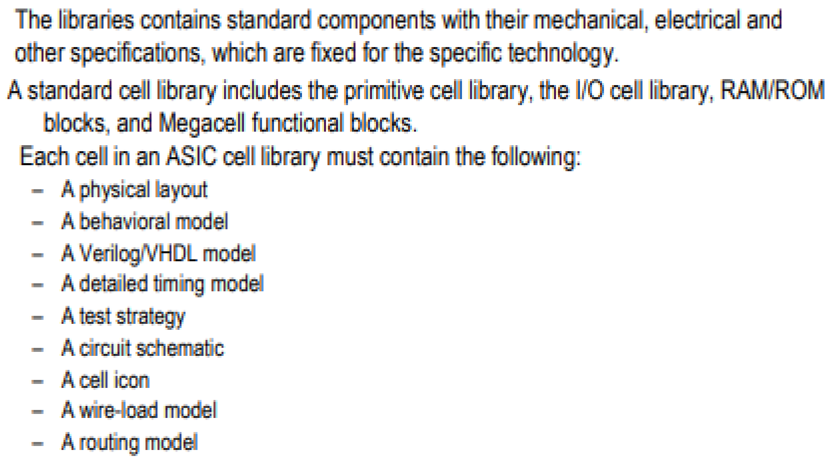


FULL CUSTOM DESIGN	SEMI CUSTOM DESIGN
All mask layers are customised in full custom design	It uses pre-designed logic cell(and gates, OR gate, multiplexers) known as standard cells.
In full custom design, all logic cells, circuits or layouts are designed specifically. Design doesn't use pretested or pre-characterized cells.	Designer used pre-tested or pre-characterized cell.
This approach is considered only when there is no suitable existing	Widely used
Offers high performance lower cost as compared to semi.	More cost. Low performance.
Design time and complexity is more.	Design time and complexity is less
Eg: Microprocessor	Eg. Digital logics

2. **State 3 important blocks in FPGA architecture? A/M 19**
FPGA architecture consists of three types of modules. They are I/O **blocks** or Pads, Switch Matrix/ Interconnection Wires and Configurable logic **blocks** (CLB)

3. **Compare between Xilinx CLB interconnect and Altera LAB interconnect? N/D 18**

4. **What is the role of cell libraries in ASIC design? A/M 18**



The libraries contains standard components with their mechanical, electrical and other specifications, which are fixed for the specific technology.

A standard cell library includes the primitive cell library, the I/O cell library, RAM/ROM blocks, and Megacell functional blocks.

Each cell in an ASIC cell library must contain the following:

- A physical layout
- A behavioral model
- A Verilog/VHDL model
- A detailed timing model
- A test strategy
- A circuit schematic
- A cell icon
- A wire-load model
- A routing model

5. **What are the two types of routing? A/M 18**
global routing
detailed routing.
Global routing first partitions the routing region into tiles and decides tile-to-tile paths for all nets while attempting to optimize some given objective function (e.g., total wirelength

and circuit timing). Then, guided by the paths obtained in global routing, detailed routing assigns actual tracks and vias for nets

6. **What is ULSI? N/D 17**
   Ultra large scale integration is the process of integrating or embedding millions of transistor on a single ship.

7. **Write the various types of routing procedures? N/D 17**
   Act 1
   Act 2
   Act3

8. **What is CBIC?** **A/M 17**
   Cell based IC uses the predesigned logic cells like AND gates,OR gates, multiplexers and flip flop.

9. **Name the elements in configuration logic blocks?** **A/M 17**
   Input ouput
   Enable clock
   Direct reset
   Clock

10. **What is the standard cell based ASIC design? N/D 16**
    Cell based IC uses the predesigned logic cells like AND gates,OR gates, multiplexers and flip flop.

11. **What is antifuse?state the merits and demerits? N/D 16**
    Antifuse in integrated circuit. antifuses are widely used in permanently program integrated circuits.

12. **What are feed-through cells? State their uses? A/M 16**
    Feed through cells needed for vertical routing for routing using the same metal layer(s) as within cells.

13. **State the features of full custom design? A/M 16**
    A Full custom ASIC is one which includes some (possibly all) logic cells that are customized and all mask layers that are customized.
    The manufacturing lead time (the time required just to make an IC not including design time) is typically eight weeks for a full-custom IC.

PART B

1. Explain the various types of ASIC with neat diagram? **A/M 18, N/D 17**
2. Draw and explain the building block of FPGA architecture? **A/M 19, A/M 18, N/D 17 ,A/M 17**
3. Describe FPGA interconnect routing resources with neat diagrams. **A/M 19**
4. (a) Explain different types of ASCII with neat diagram? A/M 17
5. (a) Discuss the different types of programming technology used in FPGA design? **N/D 16**
6. (b) Briefly explain about the semi custom ASIC WITH ITS claasification? **N/D 16**

7. (a) With neat sketch explain the CLB,IOB and programmable interconnects of an FPGA device **A/M 16**
8. (b) Write brief notes on **A/M 16**
   i) Full custom ASIC
   ii) Semi custom ASIC