1 COMBINATIONAL MOS LOGIC CIRCUITS

1.1 INTRODUCTION

The Combinational logic circuits, or gates, perform Boolean operations on multiple input variables and determine the outputs as Boolean functions of the inputs. Logic circuits can be represented as a multiple-input, single-output system is shown in figure 1.1.

![Figure 1.1: Generic combinational logic circuit.](image)

The Combinational logic circuits are the basic building blocks of all digital systems. All input variables are represented by node voltages, referenced to the ground potential. The output node is loaded with a capacitance $C_{\text{load}}$ which represents the combined parasitic device capacitance in the circuit and the interconnect capacitance components. Static & dynamic characteristics of various combinational MOS logic circuits will be described in this chapter.

1.2 nMOS LOGIC CIRCUITS WITH A MOS LOADS

1.2.1 Two-Input NOR Gate

The circuit diagram, the logic symbol, and the corresponding truth table of the two-input depletion-load NOR gate is shown in figure 1.2.
Figure 1.2: A two-input depletion-load NOR gate, its logic symbol, and truth table.

The Boolean OR operation is performed by parallel connection of the two enhancement-type nMOS driver transistors. If the input voltage VA or VB is equal to logic-high level, the corresponding driver transistor turns on and provides a conducting path between the output node and the ground, the output voltages becomes low. When VA = VOL, VB = VOH or VA = VOH, VB = VOL, the NOR2 circuit reduces to nMOS depletion-load inverter. The output low voltage level VOL in both cases is given by

$$V_{OL} = V_{OH} - V_{T0} - \sqrt{(V_{OH} - V_{T0})^2 - \left(\frac{k_{load}}{k_{driver}}\right) \cdot |V_{T,load}(V_{OL})|^2}$$

When VA = VB = VOH, two parallel conducting paths are created between the output node and the ground, then VOL is given by

$$V_{OL} = V_{OH} - V_{T0} - \sqrt{(V_{OH} - V_{T0})^2 - \left(\frac{k_{load}}{k_{driver,A} + k_{driver,B}}\right) \cdot |V_{T,load}(V_{OL})|^2}$$

When VA = VB = VOL, both driver transistors remain cut-off, the output node voltage is pulled to a logic-high level by depletion-type nMOS load transistor.

1.2.1.1 Generalized NOR Structure with Multiple Inputs

An n-input NOR with nMOS depletion load logic and equivalent circuit are shown in figure 1.3. The combined current $I_D$ in the circuit is supplied by the driver transistors which are turned on.
The combined pull-down current is expressed as

\[
I_D = \begin{cases} 
\frac{\mu_n C_{ox}}{2} \left( \sum_{k(\text{on})} \frac{W}{L} \right) \left[ 2(V_{GS} - V_{TH})V_{out} - V_{out}^2 \right] & \text{linear} \\
\frac{\mu_n C_{ox}}{2} \left( \sum_{k(\text{on})} \frac{W}{L} \right) (V_{GS} - V_{TH})^2 & \text{saturation}
\end{cases}
\]

The multiple-input NOR gate can also be reduced to an equivalent inverter shown in figure 1.3 (b), the (W/L) ratio of the driver transistor is

\[
\left( \frac{W}{L} \right)_{\text{equivalent}} = \sum_{k(\text{on})} \left( \frac{W}{L} \right)_{k}
\]

The source terminals of all enhancement-type nMOS driver transistors are connected to ground, and the drivers do not experience any substrate-bias effect. The depletion-type nMOS load transistor is subjected to substrate-bias effect.

1.2.1.2 Transient Analysis of NOR2 Gate

The figure 1.4 shows the two-input NOR gate with all of its relevant parasitic device capacitance. The parasitic capacitances are combined into one lumped capacitance, connected between the output node and the ground.
Figure 1.4: Parasitic device capacitances in NOR2 gate and lumped equivalent load capacitance.

The combined load capacitance, $C_{load}$ is

$$C_{load} = C_{gd,A} + C_{gd,B} + C_{gd,load} + C_{db,A} + C_{db,B} + C_{sb,load} + C_{wire}$$

The load capacitance at the output node of the equivalent inverter corresponding to a NOR gate is always larger than the total lumped load capacitance of the actual inverter with same dimensions. Transient response of the NOR gate will be slower than that of the equivalent inverter.

1.2.2 Two-Input NAND Gate

The circuit diagram, the logic symbol, and the corresponding truth table of the two-input depletion-load NAND gate are shown in figure 1.5. The Boolean AND operation is performed by the series connection of the two enhancement-type nMOS driver transistors. If the input voltage $VA$ and $VB$ is equal to logic-high level, there is a conducting path between the output node and the ground, the output voltages becomes low.
The output low voltage level $V_{OL}$ in this case is given by

$$V_{OL} \approx 2 \left( V_{OH} - V_{T0} - \sqrt{(V_{OH} - V_{T0})^2 - \left( \frac{k_{load}}{k_{driver}} \right) \cdot |V_{T, load}(V_{OL})|^2} \right)$$

In all other cases either one or both of the driver transistors will be off, and the output voltage will be pulled to a logic-high level by depletion-type nMOS load transistor.

### 1.2.2.1 Generalized NAND Structure with Multiple Inputs

An $n$-input NAND with nMOS depletion load logic and equivalent inverter circuits are shown in figure 1.6.

*Figure 1.5:* A two-input depletion-load NAND gate, its logic symbol, and truth table.

*Figure 1.6:* Generalized $n$-input NAND Structure and equivalent inverter circuit.
The combined pull-down current is expressed as

\[ I_D = \frac{\mu_n C_{ox}}{2} \left( \frac{1}{\sum_{k(\text{on})} \left( \frac{W}{L} \right)_k} \right) \left\{ \frac{2(V_{in} - V_{T0})V_{out}}{(V_{in} - V_{T0})^2} \right\} \text{ linear} - \text{saturation} \]

The multiple-input NAND gate can also be reduced to an equivalent inverter shown in figure 1.6 (b), the (W/L) ratio of the driver transistor is

\[ \left( \frac{W}{L} \right)_{\text{equivalent}} = \frac{1}{\sum_{k(\text{on})} \left( \frac{W}{L} \right)_k} = \frac{1}{n} \left( \frac{W}{L} \right) \]

The series structure consisting of n driver transistors has an equivalent (W/L) ratio of (W/L)_{driver} when all inputs are logic-high. For two-input NAND gate, each driver transistor must have a (W/L) ratio twice that of equivalent inverter driver.

1.2.2.2 Transient Analysis of NAND2 Gate

The figure 1.7 shows the two-input NAND gate with all of its relevant parasitic device capacitance.

![Figure 1.7: Parasitic device capacitances in NAND2 gate.](image)
Combined load capacitance $C_{load}$, when $VB = VOH$ and $VA$ switches from $VOH$ to $VOL$ is

$$C_{load} = C_{gd,load} + C_{gd,A} + C_{db,A} + C_{sb,load} + C_{wire}$$

When $VA = VOH$ and $VB$ switches from $VOH$ to $VOL$, the $C_{load}$ is given by

$$C_{load} = C_{gd,load} + C_{gd,A} + C_{gs,A} + C_{db,A} + C_{db,B} + C_{sb,A} + C_{sb,load} + C_{wire}$$

High-to-low switching delay from signal $VB$ connected to the **bottom transistor** is larger than the high-to-low switching delay from signal $VA$ connected to the **top transistor**. Transient response of the NAND gate will be slower than that of the equivalent inverter.

### 1.3 CMOS LOGIC CIRCUITS

#### 1.3.1 CMOS Two-Input NOR Gate

The design and analysis of CMOS logic circuits are based on the principles developed for the nMOS depletion-load logic circuits. Figure 1.8 shows the circuit diagram of a two-input CMOS NOR gate.

![Figure 1.8: A CMOS NOR2 gate, and its complementary operation.](image)

**Operation:** when either one or both inputs are high, there is a conducting path between the output node and the ground created by n-net and the p-net is cut-off. If both the input voltages are low, the n-net is cut-off, then the p-net creates a conducting path between the output node and supply voltage VDD. Thus the dual the circuit structure allows that for any given input combination, the output is either to VDD or ground via a low-resistance path. The DC current path is not established between VDD and ground for any input combinations.
A CMOS NOR2 gate and its inverter equivalent circuits are shown in figure 1.9.

The Switching voltage of the CMOS inverter is given by

$$V_{th}(\text{INR}) = V_{T,n} + \frac{k_p}{k_n} \left( V_{DD} - |V_{T,p}| \right) \frac{1}{1 + \sqrt{\frac{k_p}{k_n}}}$$

If $k_n = k_p$ and $V_{Tn} = |V_{Tp}|$, the switching threshold of the inverter is $V_{DD}/2$.

The switching threshold voltage of the NOR2 gate can be obtained by using equivalent inverter approach. When both inputs are identical the parallel connected nMOS transistors can be represented by a single nMOS transistor with $2k_n$. Similarly the series-connected pMOS transistors are represented by a single pMOS transistor with $k_p/2$. Substituting these values in the switching voltage equation of inverter, the Switching voltage of the CMOS NOR2 gate is given by

$$V_{th}(\text{NOR2}) = \frac{V_{T,n} + 2k_p}{4k_n} \left( V_{DD} - |V_{T,p}| \right) \frac{1 + \frac{k_p}{k_n}}{1 + \frac{1}{2} \sqrt{\frac{k_p}{k_n}}}$$

If $k_n = k_p$ and $V_{Tn} = |V_{Tp}|$, the switching threshold of NOR2 gate is $(V_{DD} + V_{Tn})/3$.

The figure 1.10 shows the two-input NOR gate with all of its relevant parasitic device capacitance.
1.3.2 CMOS Two-Input NAND Gate

A CMOS NAND2 gate and its inverter equivalent circuits are shown in figure 1.11. The operating principle of this circuit is the exact dual of CMOS NOR2 operation explained in section 1.3.1.

CMOS inverter equivalent have nMOS pull-down device of gain factor kn/2 and a pMOS pull-up device of 2kp to achieve equivalent delay and rise/fall times. Assume both nMOS devices have the same W/L (and the same for both pMOS).
An analysis of the dc voltage transfer curve, is obtained by setting the currents in the nMOS and pMOS transistors equal, yields the switching threshold equation as given by

\[
V_{th}(\text{NAND2}) = \frac{V_{T,n} + 2 \sqrt{k_p/k_n} \left(V_{DO} - |V_{T,p}|\right)}{1 + 2 \sqrt{k_p/k_n}}
\]

**Note:**
- If \(k_n = k_p\) and \(V_{Tn} = |V_{Tp}|\), we have \(V_{th} = (2VDD - |VTp|)/3\)
- To have \(V_{th} = VDD/2\), we need \(k_n = 4 k_p\)
- A good compromise might be \(k_n = 2 k_p\)

### 1.3.3 Layout of Simple CMOS Logic Gates

Figure 1.12 shows a sample layout of CMOS 2-input NOR gate, using single-layer metal and single-layer polysilicon.

**Figure 1.12:** CMOS 2-input NOR schematic with an example layout.

Features of the layout are:
- Single vertical poly lines for each input
- Single active shapes for N and P devices, respectively
- Metal buses running horizontal

The stick diagram for the CMOS NOR2 gate is shown in the figure 1.13, which corresponds directly to the layout, but does not contain W and L information. Stick diagram is useful for planning optimum layout topology.
Figure 1.13: Stick diagram layout of CMOS NOR2 gate.

The CMOS 2-input NAND circuit and its example layout are shown in figure 1.14.

Figure 1.14: CMOS 2-input NAND schematic with its example layout.

Layout features are:
- Single poly-silicon lines (for inputs) are run vertically across both N and P active regions
- Single active shapes are used for building both nMOS devices and both pMOS devices
- Power bussing is running horizontal across top and bottom of layout
- Output wire runs horizontal for easy connection to neighboring circuit

1.4 COMPLEX LOGIC CIRCUITS

The nMOS depletion-load logic gate used to realize the Boolean function Z is shown in figure 1.15.

\[ Z = A(D + E) + BC \]
Figure 1.15: nMOS complex logic gate.

The equivalent-driver (W/L) ratio of the pull-down network is given below.

\[
\left( \frac{W}{L} \right)_{\text{equivalent}} = \left( \frac{W}{L} \right)_B + \left( \frac{W}{L} \right)_C + \left( \frac{W}{L} \right)_A + \left( \frac{W}{L} \right)_D + \left( \frac{W}{L} \right)_E
\]

The design strategy yields the following ratios for the three worst-case paths.

\[
\left( \frac{W}{L} \right)_A = \left( \frac{W}{L} \right)_D = 2 \left( \frac{W}{L} \right)_{\text{driver}}
\]

\[
\left( \frac{W}{L} \right)_A = \left( \frac{W}{L} \right)_E = 2 \left( \frac{W}{L} \right)_{\text{driver}}
\]

\[
\left( \frac{W}{L} \right)_B = \left( \frac{W}{L} \right)_C = 2 \left( \frac{W}{L} \right)_{\text{driver}}
\]

1.4.1 Complex CMOS Logic Gates

The CMOS logic gate realization of the Boolean function Z is shown in figure 1.16.

Figure 1.16: Complex CMOS logic gate.
1.4.1.2 Layout Technique using Euler Graph Method

Euler Graph Technique can be used to determine if any complex CMOS gate can be physically laid out in an optimum fashion:

- Start with either nMOS or pMOS tree (nMOS for this example) and connect lines for transistor segments, labeling devices, with vertex points as circuit nodes.
- Next place a new vertex within each confined area on the pull-down graph and connect neighboring vertices with new lines, making sure to cross each edge of the pull-down tree only once.
- The new graph represents the pull-up tree and is the dual of the pull-down tree.

The stick diagram shown in figure 1.17 (done with arbitrary gate ordering) gives a very non-optimum layout for the CMOS gate above (figure 1.16).

Figure 1.17: Stick layout of complex CMOS logic gate with arbitrary gate ordering.

By using the Euler path approach to re-order the poly-silicon lines of the previous chart, we can obtain an optimum layout. Find Euler path in both the pull-down tree graph and the pull-up tree graph with identical ordering of the inputs. Euler path: traverses each branch of the graph exactly once! By reordering the input gates as E-D-A-B-C, we can obtain an optimum layout of the given CMOS gate with single actives for both nMOS and pMOS devices (figure 1.18).

Figure 1.18: Stick layout of complex CMOS logic gate with reordering.
1.4.2 AOI and OAI Gates

1.4.2.1 AND-OR-INVERT (AOI) CMOS Gates

AOI complex CMOS gate can be used to directly implement a sum-of-products Boolean function. The pull-down N-tree can be implemented as follows:
- Product terms yield series-connected NMOS transistors
- Sums are denoted by parallel-connected legs
- The complete function must be an inverted representation

The pull-up P-tree is derived as the dual of the N-tree.

Example of CMOS AOI gate is shown in figure 1.19.

![AOI Gate and corresponding pull-down net.](image)

**Figure 1.19:** AOI Gate and corresponding pull-down net.

1.4.2.2 OR-AND-INVERT (OAI) CMOS Gates

An OAI CMOS gate is similar to the AOI gate except that it is an implementation of product-of-sums realization of a function. The N-tree is implemented as follows:
- Each product term is a set of parallel transistors for each input in the term
- All product terms (parallel groups) are put in series
- The complete function is again assumed to be an inverted representation

The P-tree can be implemented as the dual of the N-tree.

Example of CMOS OAI gate is shown in figure 1.20.

![OAI Gate and corresponding pull-down net.](image)

**Figure 1.20:** OAI Gate and corresponding pull-down net.
Note: AO and OA gates (non-inverted function representation) can be implemented directly on the P-tree if inverted inputs are available.

### 1.4.3 Pseudo-nMOS Gates

An approach to reduce the number of transistors is to use a single pMOS transistor, with its gate terminal connected to ground, as the load device. Complex gates of n-inputs can be implemented with n + 1 gates, instead 2n as in CMOS logic.

Pseudo-NMOS is a ratio circuit where dc current flows when the N pull-down tree is conducting. Must design the ratio of N devices W/L to P load device W/L so that when the N pull down leg with maximum resistance is conducting, the output is at a sufficiently low VOL.

Disadvantages are nonzero static power dissipation, the value of VOL and the noise margins are now determined by the ratio of the pMOS load trans-conductance to the pull-down or driver trans-conductance.

Pseudo-nMOS implementation of OAI gate is shown in figure 1.21.

**Problem:** Find an equivalent inverter circuit for the layout assuming the following: (W/L)p = 15 for all PMOS transistors and (W/L)n = 10 for all NMOS transistors.
1.4.4 CMOS Full-Adder Circuit

For 1-Bit Full Adder logic function sum and carry equations are given by

\[ \text{Sum} = A \ XOR B \ XOR C = ABC + AB'\!C' + A'\!BC' + A'B'C \]
\[ \text{Carry}_\text{out} = AB + AC + BC \]

Sum function can be written as \( \text{Sum} = ABC + (A + B + C) \cdot \text{Carry}_\text{out}' \).

The alternate representation of the sum function allows the 1-bit full adder to be implemented in complex CMOS with 28 transistors, is shown in figure 1.22.

**Figure 1.22**: Gate-level schematic of the one-bit full-adder circuit

Carry
\_out' internal node is used as an input to the adder complex CMOS gate.

Transistor-level schematic of the one-bit full-adder circuit using figure 1.22 is shown in figure 1.23.
1.5 CMOS TRANSMISSION GATES (PASS GATES)

CMOS TG consists of one nMOS and one pMOS transistor connected in parallel is shown in figure 1.24. CMOS TG operates as a bidirectional switch between the nodes A and B which is controlled by signal C. If C is high, both the transistors are turned on and provide a low resistance current path between the nodes A & B. If C is low, both the transistors are turned off and path between the nodes A & B will be an open circuit, called high-impedance state.

![CMOS Transmission Gate](image)

**Figure 1.24:** CMOS Transmission Gate

CMOS Transmission Gates can be used in logic design; a savings in transistors is often realized (not always).

**Operation:** Three regions of operation (charging capacitor 0 to VDD)
- Region 1  
  \[0 < V_{out} < |V_{tp}|\]  
  - N & P saturated
- Region 2  
  \[|V_{tp}| < V_{out} < V_{DD} - V_{tn}\]  
  - N saturated, P linear
- Region 3  
  \[V_{DD} - V_{tn} < V_{out} < V_{DD}\]  
  - N cut-off, P linear

The total drain current is given by \(I_D = I_{DS,n} + I_{SD,p}\)
Resistance of a CMOS transmission gate remains relatively constant (when both transistors are turned on) over the operating voltage range 0 to VDD. Use $R_{eq} \cdot C_L$ to do performance estimation where $R_{eq}$ is the average resistance of the TG and $C_L$ is the load capacitance.

Bias conditions and the operating regions of the CMOS TG, as function of the output voltage is shown in figure 1.25.

![CMOS Transmission Gate Diagram](image)

**Figure 1.25:** Bias conditions and the operating regions of the CMOS TG.

The equivalent resistance $R_{eq}$ for each transistor is

$$
R_{eq,n} = \frac{V_{DD} - V_{out}}{I_{DS,n}}
$$

$$
R_{eq,p} = \frac{V_{DD} - V_{out}}{I_{DS,p}}
$$

The total resistance of the CMOS TG will be parallel equivalent of two equivalent resistances $R_{eq}$ of nMOS and pMOS.

**Region 1:** The equivalent resistance of both devices is

$$
R_{eq,n} = \frac{2(V_{DD} - V_{out})}{k_n(V_{DD} - V_{out} - V_{T,n})^2}
$$

$$
R_{eq,p} = \frac{2(V_{DD} - V_{out})}{k_p(V_{DD} - |V_{T,p}|)^2}
$$

**Region 2:** The equivalent resistance of both devices is

$$
R_{eq,n} = \frac{2(V_{DD} - V_{out})}{k_n(V_{DD} - V_{out} - V_{T,n})^2}
$$

$$
R_{eq,p} = k_p\left[\frac{2(V_{DD} - |V_{T,p}|)(V_{DD} - V_{out}) - (V_{DD} - V_{out})^2)}{2}\right]
$$

$$
= k_p\left[2(V_{DD} - |V_{T,p}|) - (V_{DD} - V_{out})\right]
$$
Region 3: The equivalent resistance of both devices is

\[ R_{eq,\text{p}} = \frac{k_p}{2} \left( \frac{2(V_{DD} - |V_{T,\text{p}}|) - (V_{DD} - V_{out})}{2} \right) \]

Combining the equivalent resistance values found for the three operating regions, we can plot the total resistance of the CMOS TG as a function of the output voltage as shown in figure 1.26. The total equivalent resistance of the TG remains relatively constant, i.e., its value is almost independent of the output voltage. Individual equivalent resistance of both the nMOS and the pMOS transistors are strongly dependent on \( V_{out} \).

![Figure 1.26: The Equivalent resistance of CMOS TG](image)

1.5.1 CMOS TG Examples

2:1 MUX circuit implemented using CMOS TGs are as shown below.

![2:1 MUX circuit](image)

If \( S \) is high, input B is selected
If \( S \) is low, input A is selected
Six transistors CMOS TG Implementation of the XOR function is given below.

\[ F = \overline{AB} + \overline{A}B \]

XOR function with one TG, one standard inverter, and one special inverter gate powered from B to B’ (instead of Vdd and Vss) and inserted between A and the output F.

**CMOS TG Realization of a three-variable Boolean function**

If each CMOS TG circuits are realized with a full nMOS-pMOS pair, the disjoint n-well structures of the pMOS transistors and the diffusion contacts may cause a significant overall area increase. To reduce silicon area occupied by TG circuits, all pMOS transistors placed in one single n-well as shown below.
1.5.2 Complementary Pass-Transistor Logic

The complexity of CMOS pass-gate logic can be reduced by dropping the PMOS transistors and using only NMOS pass transistors (named CPL). In this case, CMOS inverters (or other means) must be used periodically to recover the full VDD level since the NMOS pass transistors will provide a VOH of \((VDD - VTn)\) in some cases. The CPL circuit requires complementary inputs and generates complementary outputs to pass on to the next CPL stage. Elimination of pMOS transistors reduces the parasitic capacitances associated with each node in the circuit. Speed of CPL is higher than full-CMOS. The threshold voltages of the nMOS transistors must be reduced to 0V through threshold-adjustment implants.

**CPL NAND2 and NOR2**

In figure 1.27 (a) is a NAND2 CPL circuit, (b) is a NOR2 CPL stage. Each circuit requires eight transistors, double that required using the conventional CMOS realizations.

![Figure 1.27: (a) CPL NAND2 gate and (b) CPL NOR2 gate](image)

**CPL XOR2 and XNOR2**

The XOR circuit shown in figure 1.28 contains a pMOS pull-up arrangement configured like a latch. If XOR is true, the upper internal node goes high to VDD – VT while the lower internal node goes low to GND, thus causing the cross-coupled PMOS load devices to latch and pull the upper internal node all the way to VDD. If XOR is false, the opposite happens. The inverters provide both true and complement outputs. Cross-coupled pMOS pull-up transistors are used to speed up the output response.

![Figure 1.27: CPL-based XOR gate.](image)
2 SEQUENTIAL MOS LOGIC CIRCUITS

2.1 INTRODUCTION

The sequential logic circuits contain one or more combinational logic blocks along with memory in a feedback loop with the logic: The next state of the machine depends on the present state and the inputs. The output depends on the present state of the machine and perhaps also on the inputs

- Mealy machine: output depends only on the state of the machine
- Moore machine: output depends on both the present state and the inputs

Classification of logic circuits based on their temporal behavior is shown in figure 2.1 below.

Sequential Circuit Types
- **Bistable circuits** have two stable operating points and will remain in either state unless perturbed to the opposite state
  - Memory cells, latches, flip-flops, and registers
- **Monostable circuits** have only one stable operating point, and even if they are temporarily perturbed to the opposite state, they will return in time to their stable operating point
- **Astable circuits** have no stable operating point and oscillate between several states
  - Ring oscillator

2.2 BEHAVIOR OF BISTABLE ELEMENTS

The basic bistable element is comprised of two identical inverters connected back-to-back, i.e. output of one to input of the other and vice-versa as shown in figure 2.2 (a). Intersecting VTC of two inverters, with the three possible operating points are shown in in figure 2.2 (b).
The memory cell (or latch) has two stable states where the dc voltage transfer curves cross at the VOH and VOL points, but also exhibits an unstable state where the VTC’s cross near their Vth switching points. In actual physical circuits the memory cell will never stay at the unstable point, since any small electrical noise in the circuit will trigger it to one side or the other.

CMOS SRAM cell shown in figure 2.3 (a) will either be in state “0” with V01 at ground and V02 at VDD or in state “1” with V01 at VDD and V02 at ground. Expected time-domain behavior of the output voltage are shown in figure 2.3 (b).

**Figure 2.2:** (a) Static behavior of two inverter basic bistable elements (b) VTC curve two inverters (c) Potential energy levels.

**Figure 2.3:** CMOS bistable element and time domain behavior of output voltages.
2.3 SR LATCH CIRCUIT

2.3.1 CMOS SR Latch: NOR Gate Version

The NOR-based SR Latch contains the basic memory cell (back-to-back inverters) built into two NOR gates to allow setting the state of the latch. The gate-level symbol and CMOS NOR-based SR latch are shown in figure 2.4.

![Gate-level symbol and CMOS NOR-based SR latch.](image)

**Figure 2.4:** gate-level symbol and CMOS NOR-based SR latch.

**Operation of NOR-based SR Latch:** If Set goes high, M1 is turned on, forcing Q’ low which, in turn, pulls Q high. If Reset goes high, M4 is turned on, Q is pulled low, and Q’ is pulled high. If both Set and Reset are low, both M1 and M4 are off, and the latch holds its existing state indefinitely. If both Set and Reset go high, both Q and Q’ are pulled low, giving an indefinite state. Therefore, R=S=1 is not allowed.

Truth table of NOR-based CMOS SR latch circuit is shown in table 2.1.

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q_{n+1}</th>
<th>Q'_{n+1}</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Q_n</td>
<td>Q'_n</td>
<td>hold</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>set</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>reset</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>not allowed</td>
</tr>
</tbody>
</table>

**Table 2.1:** Truth table of NOR-based CMOS SR latch.

The operation modes of the transistors in NOR-based CMOS SR latch circuit is shown in table 2.2.
Table 2.2: The operation modes of the transistors in NOR-based CMOS SR latch.

2.3.1.1 Transient Analysis of SR Latch

The CMOS SR NOR latch with lumped capacitances at both output nodes are shown in figure 2.5.

Figure 2.5: CMOS SR NOR latch with lumped capacitances.

The total lumped capacitance at each node is
\[
C_{\bar{Q}} = C_{\text{gb},2} + C_{\text{gb},5} + C_{\text{db},3} + C_{\text{db},4} + C_{\text{db},7} + C_{\text{db},8} + C_{\text{db},5} + C_{\text{db},6}
\]
\[
C_{\bar{Q}} = C_{\text{gb},3} + C_{\text{gb},7} + C_{\text{db},1} + C_{\text{db},2} + C_{\text{db},5} + C_{\text{db},5} + C_{\text{db},6}
\]

2.3.2: Depletion Load nMOS SR Latch: NOR Version

A depletion load version of the NOR-based SR latch is shown figure 2.6. Functionally it is the same as CMOS version. The latch is a ratio circuit. Low side conducts dc current, causing higher standby power than CMOS version
2.3.3 CMOS SR Latch: NAND Gate Version

The NAND-based SR Latch contains the basic memory cell (back-to-back inverters) built into two NAND gates to allow setting the state of the latch. The gate-level symbol and CMOS NAND-based SR latch are shown in figure 2.7.

Operation of NAND-based SR Latch: The circuit responds to active low S and R inputs: If S goes to 0 (while R = 1), Q goes high, pulling Q’ low and the latch enters Set state. If R goes to 0 (while S = 1), Q’ goes high, pulling Q low and the latch is Reset. Hold state requires both S and R to be high. S = R = 0 if not allowed, it would result in an indeterminate state.

Truth table of NAND-based CMOS SR latch circuit is shown in table 2.3.
Table 2.3: Truth table of the NAND-based CMOS SR latch.

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>$Q_{n+1}$</th>
<th>$\overline{Q}_{n+1}$</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>not allowed</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>set</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>reset</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$Q_n$</td>
<td>$\overline{Q}_n$</td>
<td>hold</td>
</tr>
</tbody>
</table>

2.3.4: Depletion Load nMOS SR Latch: NAND Version

A depletion load version of the NAND-based SR latch is shown figure 2.8. Functionally it is the same as the CMOS version.

![Figure 2.8: Depletion Load nMOS SR Latch circuit based NAND2 gate.](image)

2.4 CLOCKED LATCH AND FLIP FLOP CIRCUITS

2.4.1 Clocked SR Latch: NOR Version

The clocked NOR-based SR latch, contains the basic memory cell built into two NOR gates to allow setting the state of the latch with a clock added as shown in figure 2.9. The latch is responsive to inputs S and R only when CK is high. When CK is low, the latch retains in its current state.

![Figure 2.9: Gate-level schematic of clocked NOR-based SR latch.](image)
The timing diagram shown in figure 2.10 shows the level-sensitive nature of the clocked SR latch. Note that four times Q changes state:

- When S goes high during positive CK
- On leading CK edge after changes in S & R during CK low time
- A positive glitch in S while CK is high
- When R goes high during positive CK

![Timing Diagram](image.png)

**Figure 2.10:** Sample input and output waveform of clocked NOR-based SR latch.

CMOS AOI implementation of clocked NOR-based SR latch of figure 2.9 is shown in figure 2.11. Only 12 transistors required. When CK is low, two series legs in N tree are open and two parallel transistors in P tree are ON, thus retaining state in the memory cell. When CK is high, the circuit becomes simply a NOR-based CMOS latch which will respond to inputs S and R.

![CMOS AOI Implementation](image.png)

**Figure 2.11:** AOI-based implementation of the clocked NOR-based SR latches.
2.4.2 Clocked SR Latch: NAND Version

All NAND version of clocked SR latch with active high clock input is shown in figure 2.12. Circuit is implemented with four NAND gates, not with an AOI or OAI. 16 transistors required. The latch is responsive to S or R only if CK is high. When CK is low, the latch retains its present state.

![Gate-level schematic of clocked NAND-based SR latch with active low inputs.](image)

**Figure 2.12:** Gate-level schematic of clocked NAND-based SR latch with active low inputs.

NAND version of clocked SR latch, with active low clock input is shown in figure 2.13 (OAI structure). The changes in the input signal level will be ignored when CK = ‘1’ and inputs will influence the outputs when clock is active, i.e., CK = ‘0’.

![Gate-level schematic of clocked NAND-based SR latch with active high inputs.](image)

**Figure 2.13:** Gate-level schematic of clocked NAND-based SR latch with active high inputs.

2.4.3 Clocked CMOS JK Latch: NAND Version

The SR latch has a problem in that when both S and R are high, its state becomes indeterminate. Above problem can be overcome by adding two feedback lines from the outputs to inputs, such that all states in the truth table are allowable as shown in figure 2.14. The resulting circuits are called a JK latch.
All NAND implementation of the JK latch with active high inputs is shown in figure 2.15.

Figure 2.14: Gate-level schematic of clocked NAND-based JK latch.

Figure 2.15: All NAND implementation of clocked JK latch.

Detailed truth table of the JK latch circuit is shown in table 2.4.

<table>
<thead>
<tr>
<th>$J$</th>
<th>$K$</th>
<th>$Q_n$</th>
<th>$\overline{Q_n}$</th>
<th>$S$</th>
<th>$R$</th>
<th>$Q_{n+1}$</th>
<th>$\overline{Q_{n+1}}$</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>hold</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>reset</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>set</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>toggle</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.4: Truth table of the JK latch.
2.4.4 Clocked CMOS JK Latch: NOR Version

NOR based implementation of clocked JK latch is shown in figure 2.16. CMOS realization of this circuit is shown in figure 2.17. The AOI-based circuit structure results in a relatively low transistor count, and more compact circuit compared to all-NAND realization. If both the inputs are equal to logic “1” during active phase of the clock pulse, the output of the circuit will oscillate continuously until either the clock becomes inactive (goes to zero), or one of the input signals goes to zero.

Figure 2.16: Gate-level schematic of clocked NOR-based JK latch.

Figure 2.17: CMOS AOI realization of clocked JK latch.
Note that in order to prevent the JK Latch above from oscillating continuously during the clock active time; the clock width must be kept smaller than the switching delay time of the latch. Otherwise, several oscillations may occur before the clock goes low again. In practice this may be difficult to achieve.

### 2.4.5 Operation of the JK Latch as a Toggle Switch

If clock timing constraint described before is satisfied, the output of the JK latch will toggle only once for each clock pulse, if both inputs are equal to logic ‘1’. The circuit which is operated exclusively in this mode is called toggle switch as shown in figure 2.18.

![Figure 2.18: JK latch as a toggle switch](image)

### 2.4.6 JK Master-Slave Flip-Flop

A Flip-Flop is defined as two latches connected serially and activated with opposite phase clocks. First latch is the Master; Second latch is the Slave. Eliminates transparency, i.e. a change occurring in the primary inputs is never reflected directly to the outputs, since opposite phase clocks are used to activate the M and S latches. MS FF consisting of NAND based JK latches are shown in figure 2.19.

![Figure 2.19: MS FF consisting of NAND-based JK latches.](image)

A JK MS FF (NOR-based version) is shown in figure 2.20. The feedback paths occur from Q and Q’ slave outputs to the master inputs AOI gates. The circuit does not exhibit any tendency to oscillate when J = K = 1 no matter how long the clock period, since opposite clock phases activate the master and slave latches separately. NOR-based version uses 4 AOI CMOS gates (28 transistors). But the latches can be susceptible to “ones catching”.

2.4.6.1 JK MS Flip-Flop Problem: One’s catching

Although the JK Master-Slave Flip-Flop can be considered edge-triggered in regards to a change in $Q_s$ at the negative CLK edge, it is actually level sensitive in regards to noise on $J$ (or $K$) during the CLK high interval. Note positive glitch in $J$ which erroneously Sets the Master latch at $Q_m = 1$ during the CLK high interval and then also reflects itself in $Q_s = 1$ at the negative-going CLK edge, called “One’s Catching”. Same problem can occur with a glitch in $K$ during CLK high, causing a Reset operation. Since the master latch actually sets and latches on the noise glitch, the error is then transmitted to the slave latch during CLK’. Sample input & output waveforms of MS FF circuit are shown in figure 2.21.
2.5 CMOS D LATCH AND EDGE-TRIGGERED FLIP FLOPS

2.5.1 CMOS D-Latch Implementation

A D-latch is implemented, at the gate level, by simply utilizing a NOR-based S-R latch, connecting D to input S, and connecting D’ to input R with an inverter as shown in figure 2.22. When CK goes high, D is transmitted to output Q (and D’ to Q’). When CK goes low, the latch retains its previous state.

![Gate-level schematic and block diagram view of D-latch.](image1)

**Figure 2.22:** Gate-level schematic and block diagram view of D-latch.

The D-latch implemented with TG switches is shown in figure 2.23. The input TG is activated with CK while the latch feedback loop TG is activated with CK’. Input D is accepted when CK is high. When CK goes low, the input is open-circuited and the latch is set with the prior data D.

![CMOS implementation of D-latch.](image2)

**Figure 2.23:** CMOS implementation of D-latch.

A schematic view of the D-Latch can be obtained using simple switches in place of the TG’s as shown in figure 2.24. When CK = 1, the input switch is closed allowing new input data into the latch. When CK = 0, the input switch is opened and the feedback loop switch is closed, setting the latch.

![Schematic view of D-Latch.](image3)
Figure 2.24: Schematic view of D-latch using inverters.

An alternate (preferred) version of the CMOS D-Latch is implemented with two tri-state inverters and a normal CMOS inverter is shown in figure 2.25. Functionally it is similar to previous chart D-Latch: When CLK is high, the first tri-state inverter sends the inverted input through to the second inverter, while the second tri-state is in its high Z state. Output Q is following input D. When CLK is low, the first tri-state goes into its high Z state, while the second tri-state inverter closes the feedback loop, holding the data Q and Q' in the latch.

Figure 2.25: CMOS implementation of D-latch.

2.5.1.1 CMOS D-Latch Timing

**Latch Timing:** For the system to work correctly, the set-up time, hold time, and pulse-width must be sufficient for each bistable element. Set-up time is the minimum time that the data input of a bistable element to be held stable prior to the active clock signal. Hold time is the minimum time that the data input of the bistable element must be held stable after the active clock signal disappears.
**Timing diagram:** In order to guarantee adequate time to get correct data at the first inverter input before the input switch opens, the data must be valid for a given time ($T_{\text{setup}}$) prior to the CLK going low. To guarantee adequate time to set the latch with correct data, the data must remain valid for a time ($T_{\text{hold}}$) after the CLK goes low. Violations of $T_{\text{setup}}$ and $T_{\text{hold}}$ can cause metastability problems and chaotic transient behavior as shown in figure 2.26.

![Figure 2.26: CMOS D-latch timing.](image)

**2.5.2 CMOS D Flip-Flop**

Figure 2.27 shows a D Flip-Flop, constructed by cascading two D-Latch circuits from the previous slide: Master latch is positive level sensitive (receives data when $CK = 1$). Slave latch is negative level sensitive (receives data $Q_m$ when $CK = 0$), the circuit is **negative-edge triggered**. Master latch receives input $D$ until the $CK$ falls from 1 to 0, at which point it sets that data in the master latch and sends it through to the output $Q_s$.

![Figure 2.27: CMOS negative edge-triggered MS D-FF.](image)

Another implementation of edge-triggered D-FF is shown in figure 2.28; which consists of six NAND3 gates. This D-FF is positive edge triggered as illustrated in figure 2.29. Initially all the signal values except for $S$ are 0 i.e, $(S, R, CK, D) = (1, 0, 0, 0)$ and the $Q = 0$. In the second phase, both $D$ and $R$ switch to 1 i.e, $(S, R, CK, D) = (1, 1, 0, 1)$ but no change in $Q$
occurs and the \( Q = 0 \). In the third phase, if \( CK = 1 \) i.e, \((S, R, CK, D) = (1, 1, 1, 1)\), the output of gate 2 switches to 0, which in turn sets the output of the last stage SR latch to 1. Thus the output of this D-FF switches to 1 at the positive going edge of the clock signal, \( CK \). In the ninth phase of the waveform, \( Q \) output is not affected by negative-going edge of \( CK \), or by other signal changes.

Figure 2.28: NAND3-based positive edge-triggered D-FF.

Figure 2.29: Timing diagram of positive edge-triggered D-FF.
3.1 INTRODUCTION

Dynamic logic circuits offer several advantages in realizing high-density, high performance digital system where reduction of circuit delay and silicon area is important. Operation of all dynamic logic gates depend on temporary storage of charge in parasitic node capacitances, instead of relying on steady-state circuit behavior. Dynamic logic circuits require periodic clock signals in order to control charge refreshing.

Dynamic logic techniques save area by reducing the number of transistors per gate, and save power by reducing the number of gates and the static current in structures such as flip-flops & shift registers. Dynamic CMOS circuits save chip area while enhancing speed over conventional CMOS circuits, but precautions must be taken to ensure proper operation. Use of common clock signals the system enables synchronize the operation of various circuit blocks. Capability of temporarily storing a state at a capacitive node allows implementing simple sequential circuits with memory functions.

Disadvantage of dynamic storage is the use of small-sized, leaky capacitors for storing logic values. They must be clocked at a minimum operating frequency in order to maintain their charge.

3.1.1 Dynamic D-Latch

The Dynamic D-latch circuit is shown in figure 3.1. Parasitic input capacitance Cx plays an important role in dynamic operation of circuit. Input pass transistor is driven by external clock signal. When CK = 1, MP turns on, Cx charged or discharged through MP, depending on D voltage level, Q assumes same logic level as input. When CK = 0, MP turns off, Cx is isolated from D, amount charge stored in Cx during last cycle determines output voltage level Q.

![Dynamic D-latch](image)

Figure 3.1: Dynamic D-latch

3.2 PRINCIPLES OF PASS TRANSISTOR CIRCUITS

3.2.1 Basic Principles of Pass Transistor Circuits

The fundamental building block of dynamic logic circuits, consisting of an nMOS pass transistor, MP driving the gate of another nMOS transistor is shown in figure 3.2. The pass transistor, MP is driven by periodic clock signal and acts as access switch to either to charge up or down the parasitic capacitance Cx, depending on Vin.
Two operations are possible when \( CK = 1 \). Logic ‘1’ transfer and logic ‘0’ transfer. The output of depletion-load nMOS inverter depends on voltage \( V_x \). MP provides only current path to the intermediate capacitive node (soft node) \( X \). When \( CK = 0 \), the MP ceases to conduct and charge stored in the parasitic capacitor \( C_x \) continues to determine output level of the inverter.

### 3.2.2 Logic ‘1’ Transfer

If the soft node voltage is equal to 0 initially, i.e., \( V_x(t = 0) = 0 \) V. Logic ‘1’ level is applied to the input terminal, which corresponds to \( Vin = VDD \). When \( CK \) changes from 0 to 1, MP will be in saturation. The equivalent circuit for logic ‘1’ transfer is shown in figure 3.3.

![Figure 3.3: Equivalent circuit for logic ‘1’ transfer](image)

The pass transistor MP operating in the saturation region starts to charge up the capacitor \( C_x \), since, \( I = C \frac{dV}{dt} \). Thus,

\[
C_x \frac{dV_x}{dt} = \frac{k_n}{2} (V_{DD} - V_x - V_{T,n})^2
\]

\[
\int_0^t dt = \frac{2C_x}{k_n} \int_0^{V_x} \frac{dV_x}{(V_{DD} - V_x - V_{T,n})^2}
\]

\[
= \frac{2C_x}{k_n} \left( \frac{1}{(V_{DD} - V_x - V_{T,n})} \right) V_x
\]

\[
t = \frac{2C_x}{k_n} \left[ \left( \frac{1}{V_{DD} - V_x - V_{T,n}} \right) - \left( \frac{1}{V_{DD} - V_{T,n}} \right) \right]
\]

Above equation can be solved for \( V_x(t) \), as follows

\[
V_x(t) = (V_{DD} - V_{T,n}) \frac{\left( \frac{k_n(V_{DD} - V_{T,n})}{2C_x} \right) t}{1 + \left( \frac{k_n(V_{DD} - V_{T,n})}{2C_x} \right) t}
\]
Variation of node voltage $V_x$ w.r.t. last equation is plotted as a function of time is shown in figure 3.4.

![Graph showing variation of node voltage $V_x$](image)

**Figure 3.4:** Variation of node voltage as a function of time during logic ‘1’ transfer.

The fact that the node voltage $V_x$ has an upper limit of $V_{\text{max}} = (V_{DD} - V_{T,n})$ has significant implication for circuit design.

**Pass transistors in series:**
The node voltages in the pass transistor chain during logic ‘1’ transfer are as shown in the figure 3.5.

![Diagram of pass transistors in series](image)

**Figure 3.5:** Node voltages in a pass transistors chain during logic ‘1’ transfer.

With threshold voltage of all transistors are same, the node voltage at the end of the pass transistor chain will become one threshold voltage lower than $V_{DD}$, regardless of number of pass transistors in chain.

**Pass transistors driving gate of another Pass transistors:**
Node voltages during the logic ‘1’ transfer, when each pass transistor is driving another pass transistor are as shown in figure 3.6. In designing nMOS pass transistors logic, one must never drive a pass transistor with the output of another pass transistor.

![Diagram of pass transistors driving gate of another pass transistor](image)

**Figure 3.6:** Node voltages during logic ‘1’ transfer, when each pass transistors is driving another pass transistor.
3.2.3 Logic ‘0’ Transfer

If the that soft node voltage is equal to 1 initially, i.e., $V_x(t = 0) = V_{\text{max}} = (V_{\text{DD}} - V_{T,n})$. Logic ‘0’ level is applied to the input terminal, which corresponds to $V_{\text{in}} = 0$ V. When $CK$ changes from 0 to 1, $MP$ will be in linear region. The equivalent circuit for logic ‘0’ transfer is shown in figure 3.7.

![Figure 3.7: Equivalent circuit for logic ‘0’ transfer](image)

The direction of current flow through will be opposite to that during charge-up event. The pass transistor $MP$ operating in the linear region discharges the parasitic capacitor $C_x$ as follows:

$$-C_x \frac{dV_x}{dt} = \frac{k_n}{2} (2(V_{DD} - V_{T,n})V_x - V_x^2)$$

$$dt = -\frac{2C_x}{k_n} \cdot \frac{dV_x}{2(V_{DD} - V_{T,n})V_x - V_x^2}$$

Integrating above equation w.r.t. $t$ we get,

$$t = \frac{C_x}{k_n(V_{DD} - V_{T,n})} \ln \left( \frac{2(V_{DD} - V_{T,n}) - V_x}{V_x} \right)$$

Variation of node voltage $V_x$ w.r.t. last equation is plotted as function of time is shown in figure 3.8.

![Figure 3.8: Variation of node voltage as a function of time during logic ‘0’ transfer](image)

Fall time for the soft node voltage $V_x$ can be calculated from previous equation.

$$\tau_{fall} = 2.74A \cdot \frac{C_x}{k_n(V_{DD} - V_{T,n})}$$

Advantages of pass transistors are:

- They are not ratioed devices and can be minimum geometry
- They do not have a path from plus supply to ground, do not dissipate standby power
- They are used as function block
3.2.4 Charge Storage & Charge Leakage

During inactive clock phase (when Ck = 0), the charge leakage from the soft node is shown in figure 3.8.

![Figure 3.8: Charge leakage from the soft node.](image)

Assume that logic high voltage has transferred to soft node during CK = 1 and now both Vin and CK = 0. The charge stored in Cx will gradually leak away, primarily due to the leakage currents associated with the pass transistor. Gate current of the inverter driver transistor is negligible. Figure 3.9 shows the simplified cross-section of the nMOS pass transistor, showing the leakage components responsible for draining the soft capacitance Cx.

![Figure 3.9: Cross-section of the nMOS pass transistor.](image)

The leakage current is given by

\[ I_{\text{leakage}} = I_{\text{subthreshold}}(M_P) + I_{\text{reverse}}(M_P) \]

The certain portion of total soft-node capacitance Cx is due to reverse biased drain-substrate junction, which is also a function of soft-node voltage Vx. The equivalent circuit used for analyzing the charge leakage process is shown in figure 3.10.

![Figure 3.10: Analyzing the charge leakage process.](image)
3.3 VOLTAGE BOOTSTRAPPING

Voltage bootstrapping is a technique used for overcoming threshold voltage drops in digital circuits. Consider the circuit shown in figure 3.11, where voltage $V_x \leq V_{DD}$, M2 will operate in saturation. When $V_{in} = 0$,

$$V_{out(max)} = V_x - VT_2(V_{out})$$

![Figure 3.11: Enhancement-type circuit in which output node is weakly driven.](image)

To overcome threshold voltage drop and to obtain $V_{DD}$ at output node, $V_x$ must be increased. In figure 3.12 third transistor M3 is added.

![Figure 3.12: During bootstrapping arrangement to boost $V_x$, during switching.](image)

The two capacitor $C_s$ and $C_{boot}$ represent capacitances which dynamically couple the voltage $V_x$ to the ground and to the output. The circuit will produce a high $V_x$ during switching, so threshold voltage can be overcome at the output node.

$$V_x \geq V_{DD} + VT_2(V_{out})$$

Initially $V_{in} = 1$, M1 is in linear region & M2 in saturation and output voltage is low. Since $I_{D3} = 0$, $V_x = V_{DD} - VT_3(V_x)$

If input switches from 1 to 0 at $t = 0$, M1 turns off and $V_{out}$ will start to rise. The change in output voltage level will now be coupled to $V_x$ through bootstrap capacitor $C_{boot}$. The transient current flowing through $C_s$ and $C_{boot}$ is

$$i_{C_s} \approx i_{C_{boot}} \Rightarrow C_s \frac{dV_x}{dt} \approx C_{boot} \frac{d(V_{out} - V_x)}{dt}$$
Cs is the sum of parasitic source-to-substrate capacitance of M3 and gate-to-substrate capacitance of M2. To obtain large Cboot in comparison to Cs, an extra dummy transistor is added to the circuit as shown in figure 3.13. The dummy transistors drain and source terminals are connected together, it acts as an MOS capacitor between Vx and Vout.

![Image](image1.png)

**Figure 3.13:** Realization of the bootstrapping capacitor with a dummy MOS device.

### 3.4 DYNAMIC PASS TRANSISTOR CIRCUITS

The multi-stage pass transistor logic driven by two non-overlapping clocks are shown in figure 3.14. Circuit consists of cascaded combinational logic gates, which are interconnected through nMOS pass transistors. All inputs are driven by a single clock signal.

![Image](image2.png)

**Figure 3.14:** Multistage pass transistor logic driven by two non-overlapping clocks.

To drive the pass transistors, two non-overlapping clock signals Φ1 and Φ2 are used, as shown in figure 3.15. When clock Φ1 is active, inputs are applied to stage 1 (and also for 3) through pass transistors, while input capacitance of stage 2 retains their previously set logic levels. Next phase, when clock Φ2 is active, inputs are applied to stage 2 through pass transistors, while input capacitance of stage 1 & 3 retains their previously set logic levels. This signal timing scheme is called two-phase clocking.
Figure 3.15: Non-overlapping clock signals used for two-phase synchronous operation.

3.4.1 Depletion-Load Dynamic SR Circuit

Figure shows the three stage of a depletion-load nMOS dynamic shift register circuit driven with two-phase clocking.

Figure 3.16: Three stages of a depletion-load nMOS dynamic SR circuit driven with two-phase clocking.

A figure 3.17 shows a two-stage circuit example implemented using depletion-load nMOS complex logic gates.

Figure 3.17: A two-stage synchronous complex logic circuit example.
Depletion-load nMOS implementation of synchronous complex logic of figure 3.17 is as shown in the figure 3.18.

![Figure 3.18](image)

**Figure 3.18:** Depletion-load nMOS implementation of synchronous complex logic.

**Note:** Half-period length of the clock signal must be longer than the largest single-stage signal propagation delay in the circuit.

Enhancement-load dynamic shift register is as shown in figure 3.19. Dynamic shift register implemented with a technique named “**ratioed dynamic logic**”. The $\phi_1$ and $\phi_2$ are non-overlapping clocks. When $\phi_1$ is high, $\text{Cin}_1$ charges to $V_{dd} - V_t$ if $\text{Vin}$ is high or to GND if $\text{Vin}$ is low. When $\phi_1$ drop and $\phi_2$ comes up, the input data is trapped on $\text{Cin}_1$ and yields a logic output on $\text{Cout}_1$ which is transferred to $\text{Cin}_2$. When $\phi_2$ drops and $\phi_1$ comes up again, the logic output on $\text{Cout}_1$ is trapped on $\text{Cin}_2$, which yields a logic output on $\text{Cout}_2$, which is transferred to $\text{Cin}_3$, etc.

To avoid losing too much voltage on the logic high level, $\text{Cout}_n \gg \text{Cin}_{n+1}$ is desired. Each inverter must be ratioed to achieve a desired $\text{VOL}$ (e.g. when $\phi_2$ are high on 1st inverter).

![Figure 3.19](image)

**Figure 3.19:** Ratioed dynamic logic.

General circuit structure of ratioed synchronous dynamic logic is shown in figure 3.20. The power supply current flows only when the load devices are activated by the clock signal, the overall power consumption of dynamic enhancement-load logic is lower than depletion-load logic.
Dynamic shift register implemented with a technique named “ratioless dynamic logic” is shown in figure 3.21. $\phi_1$ and $\phi_2$ are non-overlapping clocks. When $\phi_2$ is high transferring data to stage 2, $\phi_1$ has already turned off the stage 1 load transistor, allowing a $V_{OL} = 0$ to be obtained without a ratio condition between load and driver transistors. When clock $\phi_1$ becomes active again, the valid output level across Cout2 is determined and transferred into Cin3, also new input is accepted into Cin1 during this phase. Since valid $V_{OL} = 0V$ can be achieved regardless of the driver-to-load ratio, this arrangement is called ratioless dynamic logic.

General circuit structure of ratioless synchronous dynamic logic is shown in figure 3.22.
3.5 DYNAMIC CMOS CIRCUIT TECHNIQUES

3.5.1 CMOS Transmission Gate Logic

Dynamic CMOS TG logic is shown in figure 3.23. The two-phase clocking in CMOS TG logic requires a total of four clock signals.

![Example of Dynamic CMOS TG logic.](image)

**Figure 3.23:** Example of Dynamic CMOS TG logic.

Basic building block of CMOS TG dynamic shift register is shown in figure 3.24. It consists of CMOS inverter, driven by CMOS TG. When CK = 1, TG turns on, Vin is transferred onto parasitic capacitance Cx via TG. When CK = 0, TG turns off, voltage level across Cx can be preserved until next cycle.

![Basic building block of CMOS TG dynamic shift register.](image)

**Figure 3.24:** Basic building block of CMOS TG dynamic shift register.

Low on-resistance of CMOS TG, results in smaller delay compared to nMOS switches. No threshold voltage (Vt) drop across the CMOS TG.

3.5.1.1 Single-Phase CMOS TG Dynamic Shift Register

TG of the odd-numbered stages would conduct during CK =1, TG of the even-numbered stages is off as shown in figure 3.25. The inputs are permitted in alternating half cycles. Clock signal and its complement do not contribute a truly non-overlapping signal pair. Clock skew between CK & CK’ may unavoidable because one of the signals is generated by inverting the others. Hence true two-phase clocking preferred over single-phase clocking in dynamic CMOS TG.
3.5.2 Dynamic CMOS Logic (Precharge-Evaluate Logic)

In Dynamic CMOS logic, IDD Path is turned off when clock-disabled and/or the output is evaluated when clock enabled. Circuit operation is based on first pre-charging the output node capacitance and evaluating the output level according to the applied inputs as shown in figure 3.26. Both operations are scheduled by a single clock signal which drives one nMOS and one pMOS transistor in each dynamic stage.

A dynamic logic gate example is shown in figure 3.27. When clk is low (pre-charge phase), Mp turns on & Me turns off, the output voltage is charge to VDD through pMOS transistor. Input voltages applied in this phase have no influence on output level.

When clk is high (evaluate phase), Mp turns off & Me turns on, the output node voltage may now remain in logic high level or drop to a low level, depending on input voltage levels. If input signals create a conducting path between the output node and the ground, the output capacitance will discharge toward VOL = 0V. Final discharged output level depends on time span of evaluation phase.
Figure 3.27: Dynamic CMOS logic gate implementing a complex Boolean function.

General structures of precharge-high, evaluate-low (a) and precharge-low, evaluate-high (b) are shown in figure 3.28. A clocked, precharged-high, nMOS NOR2 gate and clocked, precharged-high, nMOS NAND2 gate are shown in figure 3.29.

Figure 3.28: General structure of precharge-high, evaluate-low (a) and precharge-low, evaluate-high (b).

Figure 3.29: A clocked, precharged-high, nMOS NOR2 gate and clocked, precharged-high, nMOS NAND2 gate.
A clocked, precharged-low, nMOS NOR2 gate and clocked, precharged-low, nMOS NAND2 gate are shown in figure 3.30.

![Figure 3.30: A clocked, precharged-low, nMOS NOR2 gate and clocked, precharged-low, nMOS NAND2 gate.](image)

Precharge-low circuits are slower to pull up than the equivalent precharge-high circuits are to pull-down unless the channel widths of the pMOS FETS are increased. Precharge logic gives a very fast circuit, but circuit designers must beware of noise problems associated with circuits.

### 3.5.2.1 Cascaded Dynamic CMOS Logic Gates: Evaluate Problem

Consider the two-stage cascaded structure shown in figure 3.31. During pre-charge phase, both output voltages Vout1 & Vout2 are pulled up, external inputs are also applied. Assume Vout1 is logic 0 during evaluation phase; input for second stage is logic 1. When evaluation phase begins, both Vout1 & Vout2 are logic-high. Vout1 drops to its correct logic level after a certain delay. Evaluation of second stage is done concurrently, with high value of Vout1, the Vout2 at the end of evaluation phase will be erroneously low. Precharged nMOS gates cannot drive similar gates due to the fact that their outputs are precharged high, and become inputs to the driven gates, causing them to evaluate during precharge cycle. Alternative clocking schemes and structures must be developed to overcome this problem. One typical example is Domino logic which will be discussed in Unit 5.

![Figure 3.31: Cascading problem in Dynamic CMOS logic.](image)