ATME College of Engineering, Mysuru
Department of Electronics and Communication Engg.

VLSI Design and Testing(BEC602)

Sequential MOS Logic Circuits

1 Introduction

In **combinational logic circuits**, the output at any instant is *exclusively determined* by the *current input values*, assuming propagation delay is negligible.

- There is **no memory** of past inputs.
- These circuits are called **non-regenerative circuits** because:
  - There is **no feedback** from the output back to the input.
  - The circuit **cannot store** or remember previous states.

In contrast, **sequential logic circuits** can remember past inputs. Here, the *output depends on*:

- The **present inputs**, and
- The **history of past inputs** (i.e., the circuit has memory).

Figure below conceptually shows a sequential circuit composed of a combinational logic block and a memory element connected in a feedback loop.

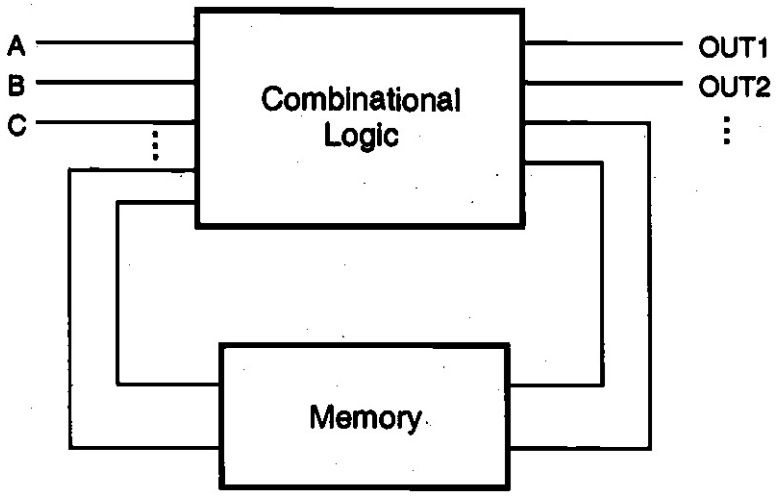

Figure 1: Sequential circuit consisting of a combinational logic block and a memory block in the feedback loop

The feedback in sequential circuits gives rise to **regenerative behavior**, allowing them to *retain state* and thus act as memory elements.

Types of Regenerative Circuits

Regenerative circuits can be classified into the following categories:

Table 1: Classification of Regenerative Circuits

Type	Stable States	Behavior Summary	Example
Bistable	Two	Can remain in either of two stable states indefinitely under appropriate conditions.	Latches, flip-flops
Monostable	One	Returns to the single stable state after a temporary state change caused by an external trigger.	One-shot pulse generator
Astable	None	Continuously oscillates between states with no stable operating point.	Ring oscillator

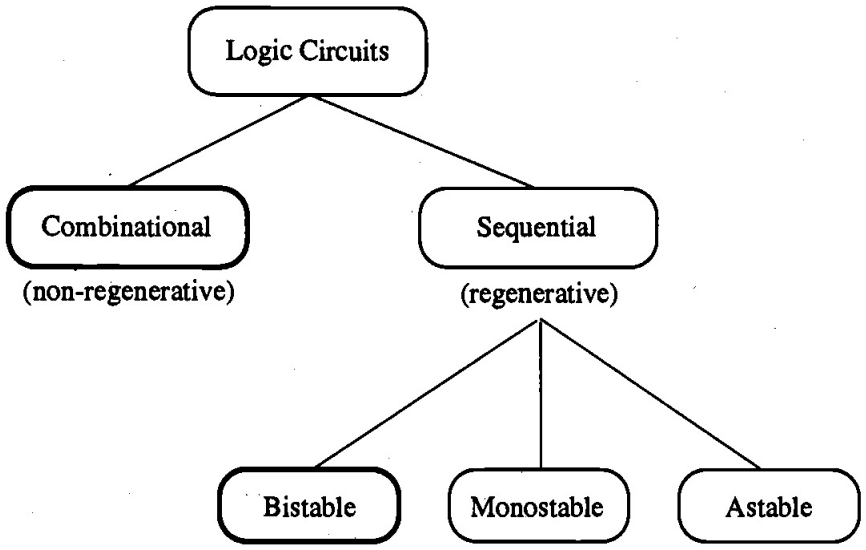

Figure 2: Classification of logic circuits based on their temporal behavior

Importance of Bistable Circuits

Among the three types, **bistable circuits** are the most widely used and significant in digital systems. They form the foundation for:

- **Latches**
- **Flip-flops**
- **Registers**
- **Memory elements**

2 Behavior of Bistable Elements

- A **bistable circuit** is a fundamental digital building block with **two stable states**.
- It is widely used in memory elements such as *latches, flip-flops, and registers.*
- The simplest bistable circuit can be constructed using **two cross-coupled inverters**.

2.1 Cross-Coupled Inverter

Two inverters are connected in a feedback loop as shown in figure below.

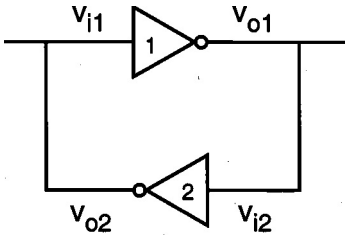

Figure 3: Cross-Coupled Inverter

In this configuration:

- Output of Inverter 1 is connected to the input of Inverter 2
- Output of Inverter 2 is connected to the input of Inverter 1

Thus, we have:

$$v_{O1} = v_{I2}, \quad v_{O2} = v_{I1}$$

2.1.1 Voltage Transfer Characteristics (VTC)

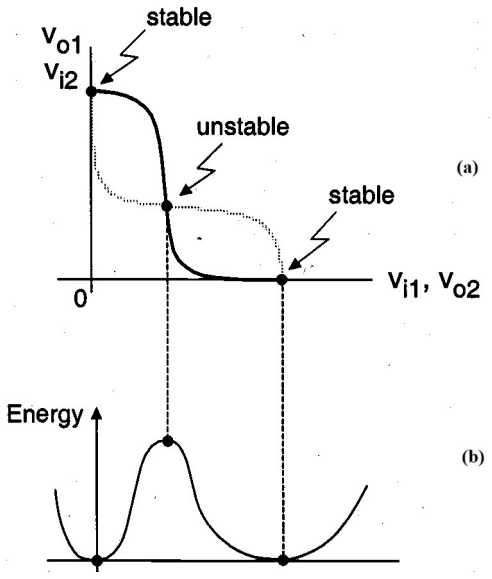

Figure 4: (a) VTC of Cross - Coupled Inverter; (b) Potential energy landscape

The voltage transfer curves (VTC) of both inverters can be plotted on the same graph as sown in above figure(a). The intersection points indicate possible operating points of the circuit.

- There are **three intersection points**
- Two of them are **stable**
- One is **unstable**

Stability analysis:

- At stable points: inverter gain $< 1 \Rightarrow$ small disturbances decay.
- At the unstable point: inverter gain $> 1 \Rightarrow$ disturbances are amplified.

Note

Consider the figure (a):

- For inverter 1, at stable points, the slope (or gain), i.e., $\Delta V_{o1}/\Delta V_{i1}$ is less than 1
- For inverter 2, at stable points, the slope (or gain), i.e., $\Delta V_{o2}/\Delta V_{i2}$ is less than 1
- For inverter 1 and 2, at unstable point, the slope(or gain), i.e., $\Delta V_{o1}/\Delta V_{i1}$ and $\Delta V_{o2}/\Delta V_{i2}$ is greater than 1

2.1.2 Energy Landscape Analogy

The bistable nature of a cross-coupled inverter circuit can be understood not only through voltage transfer characteristics but also through a qualitative analysis of the total potential energy levels at different operating points refer figure (b).

- The circuit exhibits three possible operating points:
  - Two **stable** operating points
  - One **unstable** operating point
- At the two stable points, the **total potential energy** is at a **minimum**. This occurs when the **voltage gains** of both inverters are approximately zero.
- At the intermediate operating point, the **potential energy is at a maximum**. Here, both inverters exhibit **maximum voltage gain**, making the system highly sensitive to small perturbations.
- A small disturbance at the unstable point causes the system to transition into one of the two stable states.
- This bistable energy profile explains why the cross-coupled inverter configuration is widely used in digital memory circuits such as **SR latches** and **flip-flops**, which rely on having two stable states to store binary information.

2.2 CMOS Two-Inverter Bistable Element

- Figure below shows the circuit diagram of a **CMOS bistable element** constructed using two cascaded CMOS inverters.

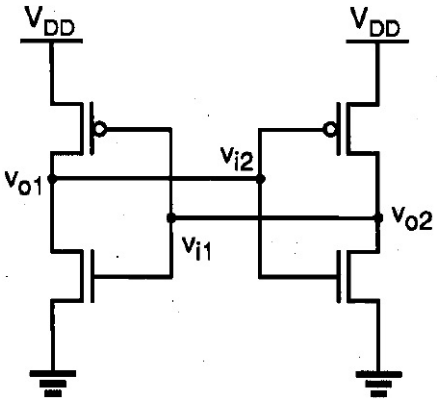

Figure 5: Circuit diagram of a CMOS bistable element

- These inverters are connected in a **positive feedback loop** — the output of the first inverter is the input to the second, and vice versa.
- This configuration creates a system with two stable states and one unstable state.

2.2.1 Unstable Operating Point

At the **unstable operating point**, both outputs are at an intermediate voltage level. At this condition:

- All four transistors (2 NMOS and 2 PMOS) are in **saturation region**.
- This results in the **maximum loop gain** of the system.
- The operating point is highly sensitive to even the smallest perturbation.

This point is referred to as a **metastable state**.

2.2.2 Behavior Under Perturbation

If the circuit is initialized exactly at the unstable point, any small **voltage perturbation** or **noise** will move the system away from it. Due to the **positive feedback**:

- The small disturbance gets amplified.
- The outputs of the inverters **diverge rapidly**.
- The system settles in one of two stable states (refer the figure shown below):
  - One output reaches **VOH** (logic high).
  - The other output reaches **VOL** (logic low).

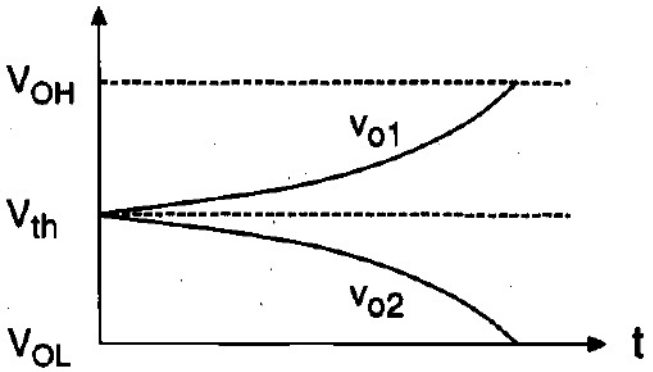

Figure 6: One possible time-domain response of the output voltages when the circuit is initially biased at its unstable operating point.

2.2.3 Polarity of Divergence

The direction in which each output voltage diverges is determined by the **initial perturbation polarity**. For example:

- If the output of inverter 1 is slightly higher than that of inverter 2, it will rise further toward **VOH**.
- Consequently, the output of inverter 2 will fall toward **VOL**.

This positive feedback ensures that the circuit acts as a **memory element**.

2.2.4 Key Takeaways

- The CMOS two-inverter bistable element is fundamental to **binary state storage**.
- The metastable point is **unstable** and cannot retain a state in the presence of noise.
- **Positive feedback** ensures rapid convergence to a stable logic level.
- This circuit is the basis for **SRAM cells, latches, and flip-flops**.

3 The SR Latch Circuit

3.1 Introduction to Bistable Elements

- A bistable element constructed using two cross-coupled inverters has two stable operating states.
- As long as a power supply is available, the circuit retains its current state, making it suitable for memory storage.
- However, the simple two-inverter setup does not allow external control to change its state.

3.2 Need for Triggering Inputs

To enable external control and allow a change of state, additional switching elements (inputs) must be added. The resulting circuit is a CMOS SR Latch (see figure below), which includes:

- S (Set) input
- R (Reset) input

This circuit is often referred to as an SR flip-flop, as it can toggle between two stable states.

3.3 Circuit Structure

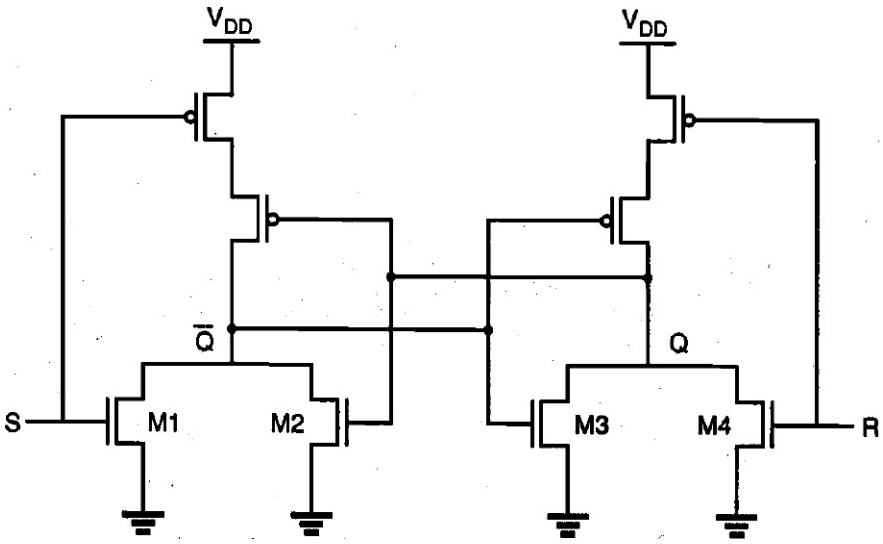

Figure 7: CMOS SR latch circuit based on NOR2 gates

The SR latch is implemented using two CMOS NOR gates:

- Each NOR gate's output is fed into one input of the other NOR gate (cross-coupling).
- The second input of each NOR gate is connected to the external control input (S or R).

The gate-level schematic and its corresponding block diagram representation are illustrated in figure below.

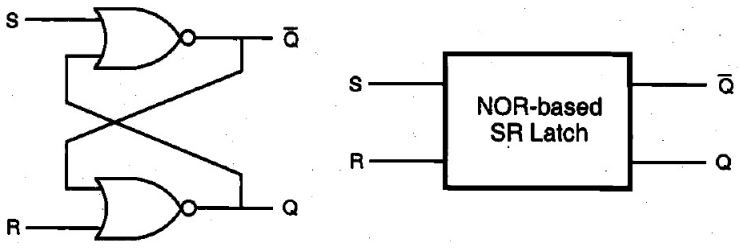

Figure 8: Gate-level schematic and block diagram of the NOR-based SR latch.

These figures clearly show how the SR latch structure supports both internal feedback and external control via S and R inputs.

The circuit provides two outputs:

- Q: Primary output
- $\overline{Q}$: Complement of Q

By definition:

- Set state: $Q = 1$, $\overline{Q} = 0$
- Reset state: $Q = 0$, $\overline{Q} = 1$

3.4 Operating Conditions

- If $S = 0$ and $R = 0$: The latch holds its previous state (memory function).
- If $S = 1$ and $R = 0$: The latch is set; $Q = 1$, $\overline{Q} = 0$.
- If $S = 0$ and $R = 1$: The latch is reset; $Q = 0$, $\overline{Q} = 1$.
- If $S = 1$ and $R = 1$: Both outputs go to 0, which violates the complementarity condition. This is a forbidden or invalid state.
- The truth table of the NOR-based SR latch is summarized in the following:

Table 2: Truth Table of NOR-Based SR Latch

S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
0	0	Q_n	$\overline{Q}_n$	Hold (No Change)
1	0	1	0	Set
0	1	0	1	Reset
1	1	0	0	Not Allowed (Invalid)

3.5 Transistor-Level Operation of CMOS SR Latch

The operation of the CMOS SR latch circuit can be examined more closely by analyzing the behavior of its four nMOS transistors: M1, M2, M3, and M4.

Case 1: $S = V_{OH}$**,** $R = V_{OL}$

- Transistors M1 and M2 (connected in parallel) are turned **ON**.
- This pulls node $\overline{Q}$ to logic-low level, i.e., V_{OL}.
- Transistors M3 and M4 are **OFF**, so node Q rises to logic-high level V_{OH}.
- The latch is in the **Set** state.

Case 2: $S = V_{OL}$, $R = V_{OH}$

- Transistors M1 and M2 are **OFF**.
- Transistors M3 and M4 are turned **ON**.
- Node Q is pulled to V_{OL} and node $\overline{Q}$ rises to V_{OH}.
- The latch is in the **Reset** state.

Case 3: $S = V_{OL}$, $R = V_{OL}$

- Both M1 and M4 (trigger transistors) are **OFF**.
- Depending on the previous state:
  - Either M2 is **ON** $\rightarrow \overline{Q} = V_{OL}$, $Q = V_{OH}$
  - Or M3 is **ON** $\rightarrow Q = V_{OL}$, $\overline{Q} = V_{OH}$
- The latch **holds its previous state**.

How Transistors Turn ON or OFF in CMOS SR Latch

Case 1: $S = V_{OH}$, $R = V_{OL}$

- M1 gate gets $V_{OH} \Rightarrow$ M1 ON $\Rightarrow \overline{Q} = V_{OL} \Rightarrow$ M3 gate gets $V_{OL} \Rightarrow$ M3 OFF, M4 gate gets $V_{OL} \Rightarrow$ M4 OFF $\Rightarrow$ Q=$V_{OH} \Rightarrow$ The latch is in the **Set** state.

Case 2: $S = V_{OL}$, $R = V_{OH}$

- M4 gate gets $V_{OH} \Rightarrow$ M4 ON $\Rightarrow Q = V_{OL} \Rightarrow$ M2 gate gets $V_{OL} \Rightarrow$ M2 OFF, M1 gate gets $V_{OL} \Rightarrow$ M1 OFF $\Rightarrow \overline{Q} = V_{OH} \Rightarrow$ The latch is in the **Reset** state.

Case 3: $S = V_{OL}$, $R = V_{OL}$ (Latch holds previous state)

Case 3a: Previous $Q = V_{OH}$	**Case 3b: Previous** $Q = V_{OL}$
M1 gate = $V_{OL} \Rightarrow$ OFF	M1 gate = $V_{OL} \Rightarrow$ OFF
M2 gate = V_{OH} (from Q) $\Rightarrow$ ON $\Rightarrow \overline{Q} = V_{OL}$	M2 gate = V_{OL} (from Q) $\Rightarrow$ OFF $\Rightarrow \overline{Q} = V_{OH}$
M3 gate = V_{OL} (from $\overline{Q}$) $\Rightarrow$ OFF	M3 gate = V_{OH} (from $\overline{Q}$) $\Rightarrow$ ON
M4 gate = $V_{OL} \Rightarrow$ OFF $\Rightarrow Q = V_{OH} \Rightarrow$ **previous state**	M4 gate = $V_{OL} \Rightarrow$ OFF $\Rightarrow Q = V_{OL} \Rightarrow$ **previous state**

- The above analysis focuses only on the nMOS transistor network.
- For simplicity, the operating states of the complementary pMOS transistors are not explicitly listed.
- The static operation modes and voltage levels of the NOR-based CMOS SR latch circuit are summarized in the following table:

Table 3: Operation modes of the transistors in the NOR-based CMOS SR latch circuit

S	R	Q_{n+1}	$\overline{Q}_{n+1}$	nMOS Transistor Operation
V_{OH}	V_{OL}	V_{OH}	V_{OL}	M1, M2 ON; M3, M4 OFF
V_{OL}	V_{OH}	V_{OL}	V_{OH}	M1, M2 OFF; M3, M4 ON
V_{OL}	V_{OL}	V_{OH}	V_{OL}	M1, M4 OFF; M2 ON
V_{OL}	V_{OL}	V_{OL}	V_{OH}	M1, M4 OFF; M3 ON

3.6 Transient Analysis of CMOS SR Latch

To study dynamic behavior, we consider a switching event where the latch changes state. For instance:

- Reset $\rightarrow$ Set: Applying $S = 1$, $R = 0$
- Set $\rightarrow$ Reset: Applying $S = 0$, $R = 1$

In both cases, the outputs undergo opposite transitions:

- One output rises from logic 0 to logic 1
- The other falls from logic 1 to logic 0

3.6.1 Coupled Behavior and Simplification

Ideally, simultaneous switching requires solving two coupled differential equations, which is complex. Instead, we simplify by assuming:

> *The rise and fall occur sequentially, not simultaneously.*

This approximation results in a slight overestimation of switching time but provides useful first-order analysis.

3.6.2 Capacitance at Output Nodes

The total parasitic capacitance at nodes Q and $\overline{Q}$ can be approximated as:

$$C_Q \approx C_{gb2} + C_{gb5} + C_{db3} + C_{db4} + C_{db7} + C_{sb7} + C_{db8}$$
$$C_{\overline{Q}} \approx C_{gb3} + C_{gb7} + C_{db1} + C_{db2} + C_{db5} + C_{sb5} + C_{db6}$$

The circuit diagram of the SR latch is shown in figure below together with the lumped load capacitances at the nodes Q and $\overline{Q}$

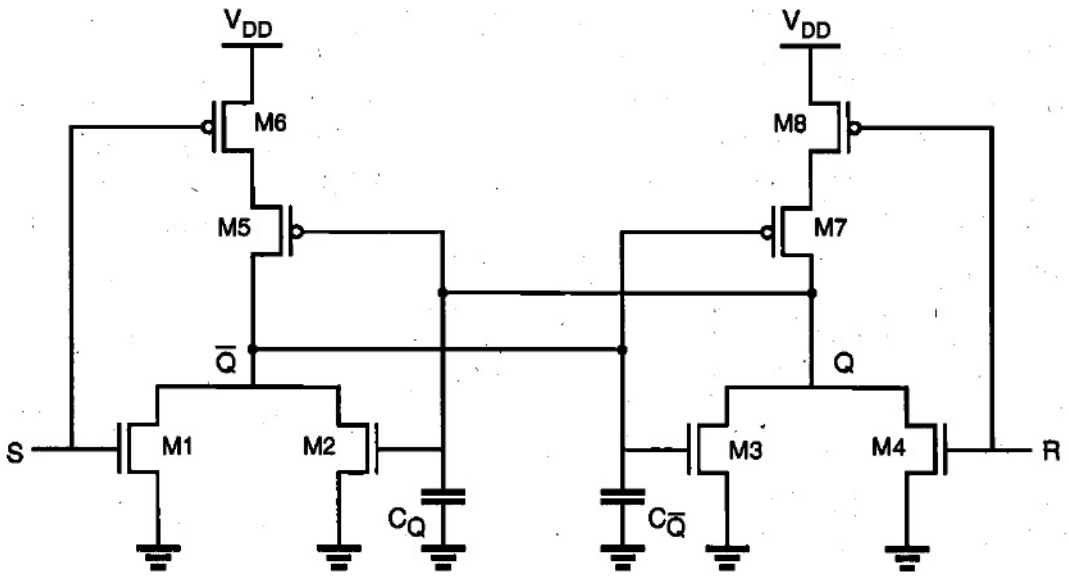

Figure 9: CMOS SR latch showing the lumped load capacitances at both output nodes

Understanding Capacitance at Output Nodes Q and $\overline{Q}$

Goal: Estimate the total parasitic capacitance at output nodes Q and $\overline{Q}$ due to MOSFET intrinsic capacitances.

Source of Capacitances:

- C_{gb}: Gate-to-bulk capacitance — if the node **drives the gate** of a transistor.
- C_{db}: Drain-to-bulk capacitance — if the node is connected to the **drain** of a transistor.
- C_{sb}: Source-to-bulk capacitance — if the node is connected to the **source** of a transistor, especially in stacked configurations.
- In stacked transistor configurations, both source and drain may contribute to output node capacitance, because intermediate nodes can swing with output signals (In our case: M5 & M7).

1. Capacitance at Q:

$$C_Q \approx C_{gb2} + C_{gb5} + C_{db3} + C_{db4} + C_{db7} + C_{sb7} + C_{db8}$$

- C_{gb2}, C_{gb5}: Gates of M2, M5 are driven by Q
- $C_{db3}, C_{db4}, C_{db7}$: Drains of M3, M4, M7 are connected to Q
- C_{sb7}: Source of stacked M7 is dynamically connected to Q
- C_{db8}: Drain M8 is dynamically connected to Q through M7

2. Capacitance at $\overline{Q}$:

$$C_{\overline{Q}} \approx C_{gb3} + C_{gb7} + C_{db1} + C_{db2} + C_{db5} + C_{sb5} + C_{db6}$$

- C_{gb3}, C_{gb7}: Gates of M3, M7 are driven by $\overline{Q}$
- $C_{db1}, C_{db2}, C_{db5}$: Drains of M1, M2, M5 are connected to $\overline{Q}$
- C_{sb5}: Source of stacked M5 is dynamically connected to $\overline{Q}$
- C_{db6}: Drain M6 is dynamically connected to $\overline{Q}$ through M5

3.6.3 Rise Time Estimation

Assuming a transition from reset to set:

$$\tau_{\text{rise},Q(SR-latch)} = \tau_{\text{rise, Q (NOR2)}} + \tau_{\text{fall, }\overline{Q}\text{ (NOR2)}}$$

Assumptions:

- M_1 turns ON $\Rightarrow$ $\overline{Q}$ falls
- M_3 turns OFF $\Rightarrow$ Q rises
- M_2 and M_4 are OFF (although M_2 might turn ON during transition, helping $\overline{Q}$ fall faster)

This method avoids simultaneous differential equations and gives a simplified delay estimate.

3.7 Depletion-Load nMOS NOR-Based SR Latch

The NOR-based SR latch can also be implemented using two cross-coupled **depletion-load nMOS NOR2 gates**, as shown in figure below.

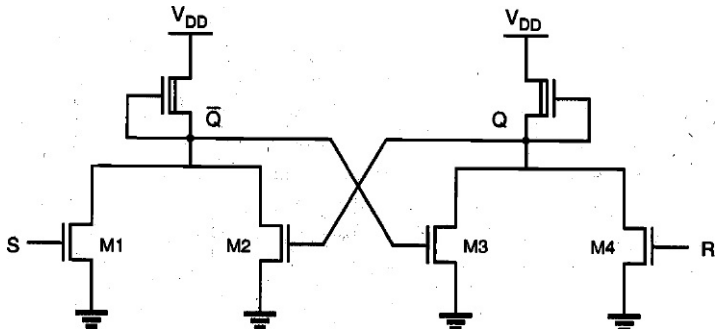

Figure 10: Depletion-load nMOS SR latch circuit based on NOR2 gates

3.7.1 Depletion-Load nMOS vs. CMOS SR Latch

Logic Functionality:

- The logical behavior of the depletion-load nMOS NOR-based SR latch is identical to that of the CMOS-based implementation.

Implementation Comparison:

- **CMOS Implementation**:
  - Offers better power efficiency.
  - No static power dissipation while holding a state.
  - Provides full output voltage swing between 0 and V_{DD}.
- **nMOS Depletion-Load Implementation**:
  - Logic works the same, but suffers from static power dissipation.
  - Output levels may not reach full rail voltages.

3.8 NAND-Based SR Latch

3.8.1 Circuit Structure

The CMOS NAND - based SR latch is shown in figure below.

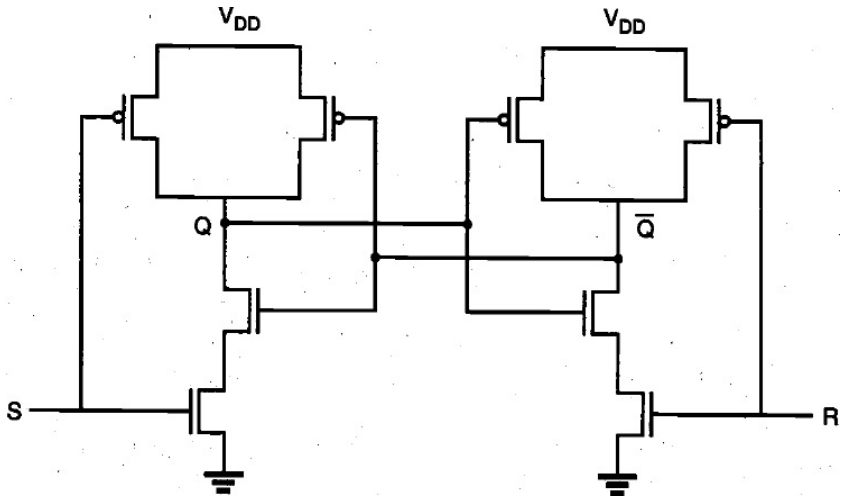

Figure 11: CMOS SR latch circuit based on NAND2 gates

- It uses two NAND2 gates with cross-coupled feedback
- Inputs: S, R (active-low)
- Outputs: Q, $\overline{Q}$

3.8.2 NAND SR Latch Behavior

1. **Set State:**
   - $S = 0$, $R = 1$
   - $\Rightarrow Q = 1$, $\overline{Q} = 0$
2. **Reset State:**

- $S = 1$, $R = 0$
- $\Rightarrow Q = 0$, $\overline{Q} = 1$

3. **Hold State:**
   - $S = 1$, $R = 1$
   - $\Rightarrow$ Previous state is held
4. **Invalid State:**
   - $S = 0$, $R = 0$
   - $\Rightarrow Q = \overline{Q} = 1$ (invalid)

The truth table is shown below:

Table 4: Truth table of NAND - based SR latch

S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
1	1	Q_n	$\overline{Q}_n$	Hold (no change)
0	1	1	0	Set
1	0	0	1	Reset
0	0	1	1	Not Allowed

The gate-level schematic and the corresponding block diagram representation of the NAND-based SR latch circuit are shown in figure below:

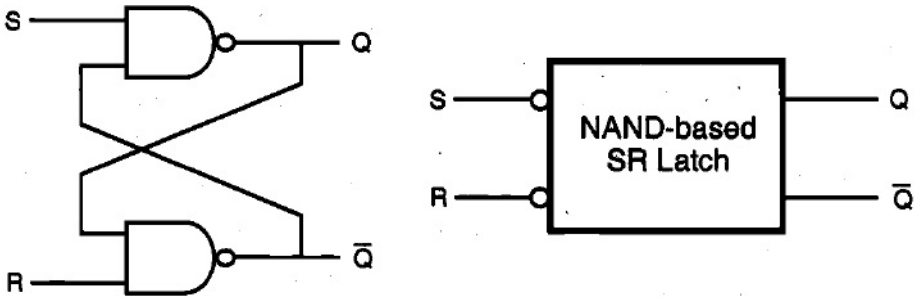

Figure 12: Gate-level schematic and block diagram of the NAND-based SR latch

3.8.3 Comparison with NOR SR Latch

- NAND latch is **active-low triggered**
- NOR latch is **active-high triggered**
- CMOS NAND latch dissipates negligible static power
- NAND latch with depletion-load NMOS gates is possible, but CMOS offers better noise margins and full output swing

The NAND-based SR latch can also be implemented by using two cross-coupled depletion-load NAND2 gates, as shown in figure below:

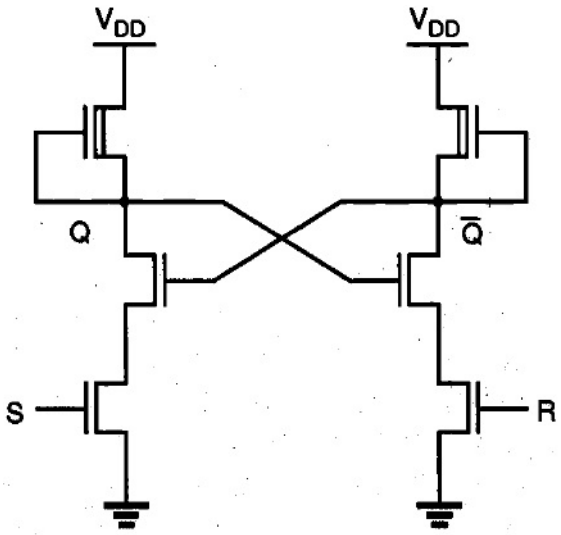

Figure 13: Depletion-load nMOS NAND-based SR latch circuit

4 Clocked Latch and Flip-Flop Circuits

4.1 Clocked SR Latch

4.1.1 Introduction

- In the previous section, we examined the behavior of the basic SR latch, which is an asynchronous sequential circuit.
- These circuits respond to changes in their inputs at a time determined by internal circuit delays.
- However, many digital systems require synchronous operation, where outputs change only in response to a clock signal.

4.1.2 Need for Clocked Latch

- To ensure that the SR latch responds to inputs in a synchronized fashion, we introduce a gating clock signal (CK).
- This clock restricts changes in the output to specific periods (when the clock is active), enabling predictable and stable circuit behavior.
- The clock (CK) is typically a periodic square waveform.
- It is applied simultaneously to all clocked logic gates in the system.
- Inputs S and R affect the latch only during the active level of the clock.

4.1.3 Clocked NOR-Based SR Latch

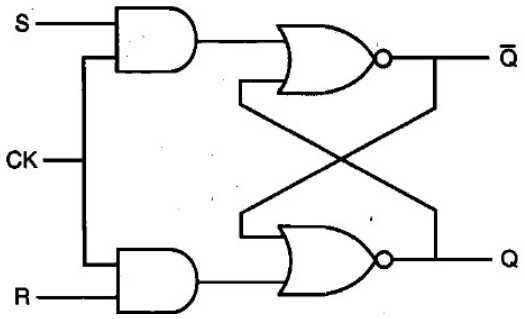

Figure 14: Gate-level schematic of the clocked NOR-based SR latch

- **If CK = 0:**
  The AND gates output 0 regardless of S and R, so the SR latch maintains its current state.
- **If CK = 1:**
  The values of S and R propagate to the NOR-based SR latch, potentially altering its state.
- **If $S = R = 1$ during CK = 1:**
  Both outputs of the latch go to 0 momentarily. When CK returns to 0, the state becomes indeterminate, depending on circuit delay mismatches.
- To illustrate the operation of the clocked SR latch, a sample sequence of CK, S, and R waveforms, and the corresponding output waveform Q are shown in figure below:

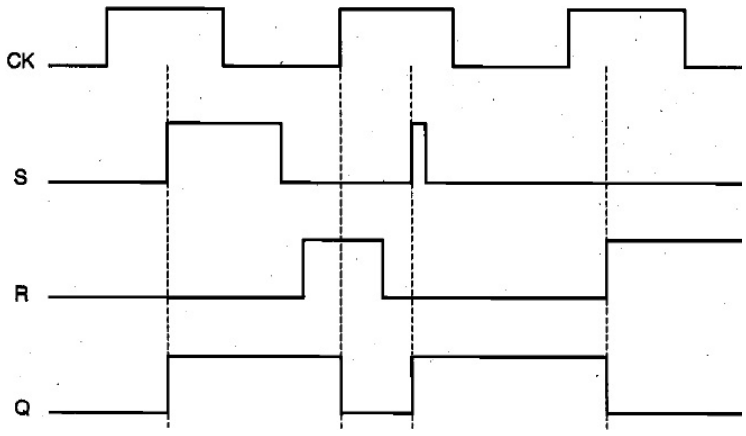

Figure 15: Sample input and output waveforms illustrating the operation of the clocked NOR - based SR latch circuit.

- The circuit is level-sensitive during CK = 1.
- Even narrow spikes or glitches in S or R during CK = 1 can trigger a change in state if the pulse width exceeds loop delay.

4.1.4 CMOS Implementation Using AOI Gates

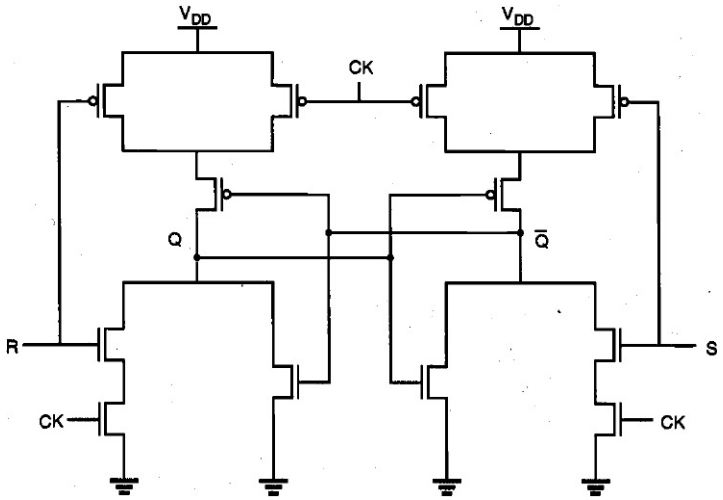

Figure 16: AOI-based implementation of the clocked NOR-based SR latch circuit

- Uses AND-OR-Invert (AOI) gates for compact design.
- Reduces the total transistor count compared to using two AND2 and two NOR2 gates.
- Offers better noise margins and lower static power dissipation.

4.1.5 Clocked NAND-Based SR Latch (Active-Low Inputs)

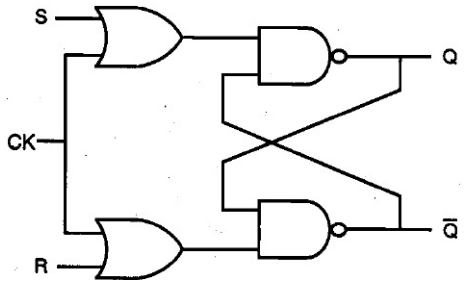

Figure 17: Gate-level schematic of the clocked NAND-based SR latch circuit, with active low inputs

- Inputs S, R, and CK are all active low.
- If CK = 1: Inputs have no effect on the output (latch holds state).
- If CK = 0: Inputs S and R determine the next state:
  - $S = 0$, $R = 1 \Rightarrow Q = 1$ (Set)
  - $S = 1$, $R = 0 \Rightarrow Q = 0$ (Reset)
- Implemented using an OAI (OR-AND-Invert) structure.

4.1.6 Clocked NAND-Based SR Latch (Active-High Inputs)

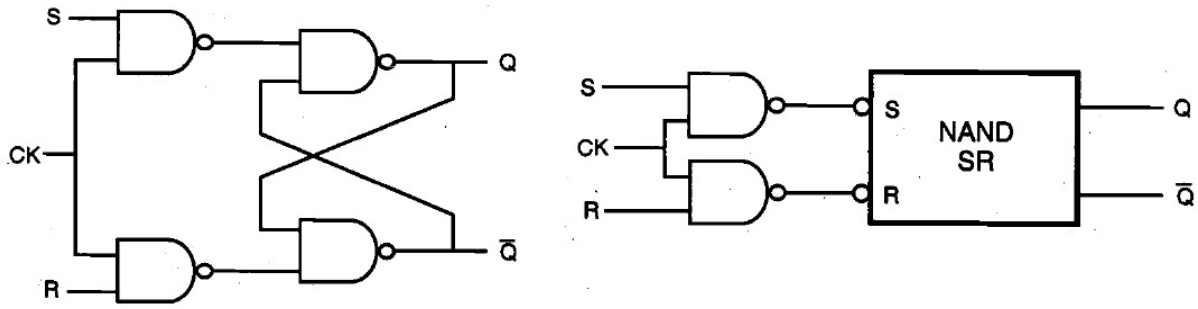

Figure 18: Gate-level schematic of the clocked NAND-based SR latch circuit, with active high inputs, and Partial block diagram representation of the same circuit.

- Inputs S, R, and CK are all active high.
- CK $= 1$, $S = 1$, $R = 0 \Rightarrow$ Set
- CK $= 1$, $S = 0$, $R = 1 \Rightarrow$ Reset
- CK $= 0 \Rightarrow$ Latch holds current state.
- Higher transistor count compared to the active-low version.

4.2 Clocked JK Latch

4.2.1 Introduction

- All simple and clocked SR latch circuits discussed earlier suffer from a common drawback: they have a **not-allowed input combination**.
- When both inputs (S and R) are activated simultaneously, the circuit enters an **indeterminate state**.
- This issue can be resolved by adding **feedback paths from the outputs to the inputs**, resulting in a new type of latch called the **JK Latch**, as shown in figure below.

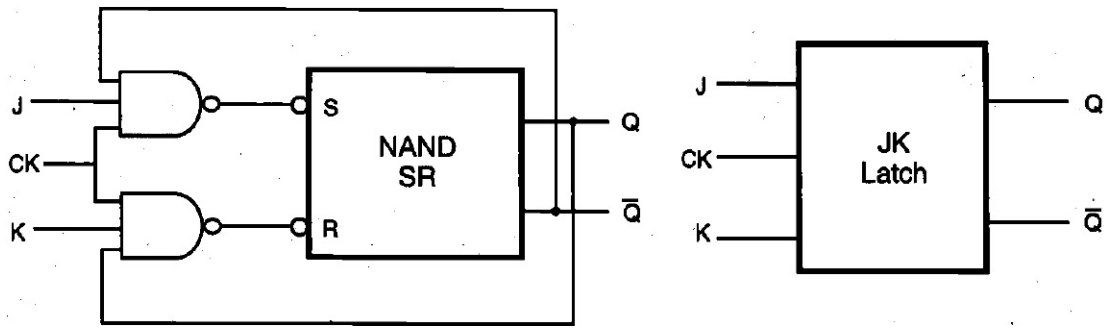

Figure 19: Gate-level schematic of the clocked NAND-based JK latch circuit

4.2.2 JK Latch: Structure and Working Principle

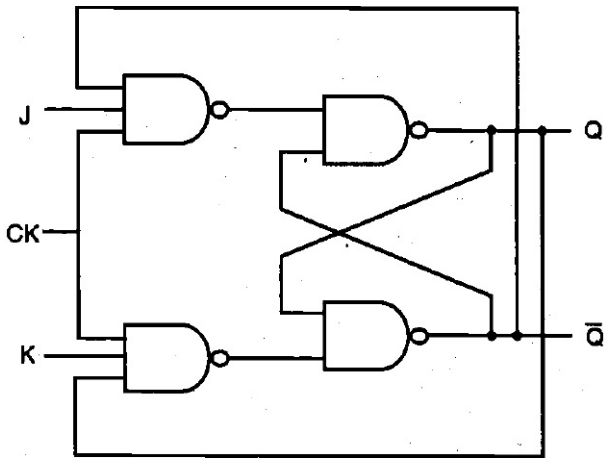

Figure 20: All-NAND implementation of the clocked JK latch circuit

- The **JK latch** enhances the basic SR latch by eliminating the forbidden input condition.
- It consists of an **all-NAND gate structure** with active-high inputs.
- The **J** and **K** inputs correspond to the **Set** and **Reset** inputs of the SR latch, respectively.
- The latch responds to inputs only when the **clock (CK)** is active (`CK = 1`).
- It preserves its state when the clock is **inactive** (`CK = 0`).

4.2.3 Input Conditions and Output Response

- The input conditions and output responses can be summarized in following truth table:

Table 5: Truth table of JK latch

J	**K**	$\mathbf{Q_{n+1}}$	**Operation**
0	0	Q_n	Hold (no change)
1	0	1	Set
0	1	0	Reset
1	1	$\overline{Q_n}$	Toggle

4.2.4 Detailed Truth Table of JK Latch

- The above conditions are detailed in following table:

Table 6: Detailed Truth Table of JK Latch

J	K	Q_n	$\overline{Q}_n$	S	R	Q_{n+1}	Operation
0	0	0	1	1	1	0	Hold
		1	0	1	1	1	
0	1	0	1	1	1	0	Reset
		1	0	1	0	0	
1	0	0	1	0	1	1	Set
		1	0	1	1	1	
1	1	0	1	0	1	1	Toggle
		1	0	1	0	0	

4.2.5 Toggle Mode and Clock Timing Constraint

- While the JK latch resolves the undefined state issue of the SR latch, it introduces a **new potential problem**:
  - If `J = K = 1` during an active clock pulse (`CK = 1`), the **output toggles continuously**.
  - To avoid this, the **clock pulse width** must be **less than the input-to-output propagation delay**.
  - This ensures that only **one toggle** occurs per clock pulse.
- If this timing constraint is met, the JK latch toggles its state only **once per clock cycle** when `J = K = 1`.
- This behavior is illustrated in figure below.

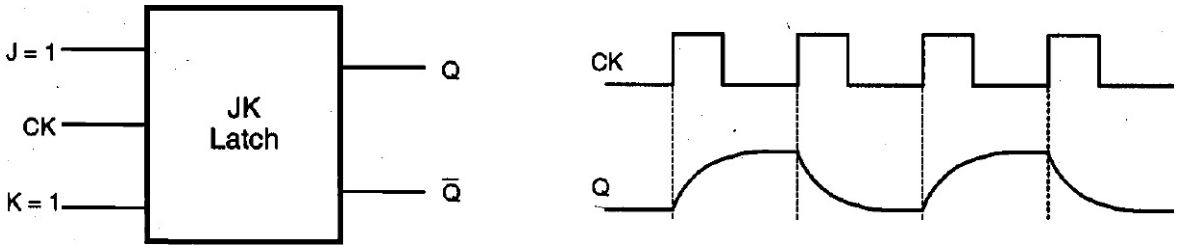

Figure 21: Operation of the JK latch as a toggle switch

- When the JK latch operates exclusively with `J = K = 1`, it functions as a **Toggle Switch**.
- Each active clock pulse causes the output to **change state** (i.e., from 0 to 1 or from 1 to 0).

4.2.6 NOR-based JK Latch Implementation

- Figure below shows an alternative implementation of the clocked JK latch using **NOR gates**.

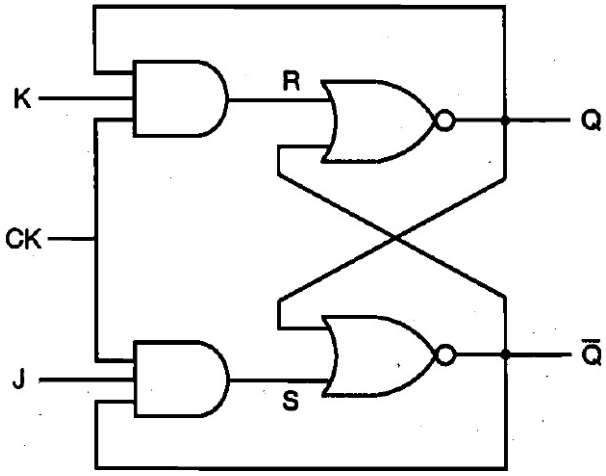

Figure 22: Gate-level schematic of the clocked NOR-based JK latch circuit

- This version, as opposed to the all-NAND realization, presents the following advantages:
  - The circuit employs a **CMOS realization** with an **AOI (AND-OR-Invert)** structure.
  - The AOI-based design results in a **lower transistor count**.
  - Consequently, the circuit becomes **more compact and efficient**.
- The AOI realization of JK latch is shown in figure below:

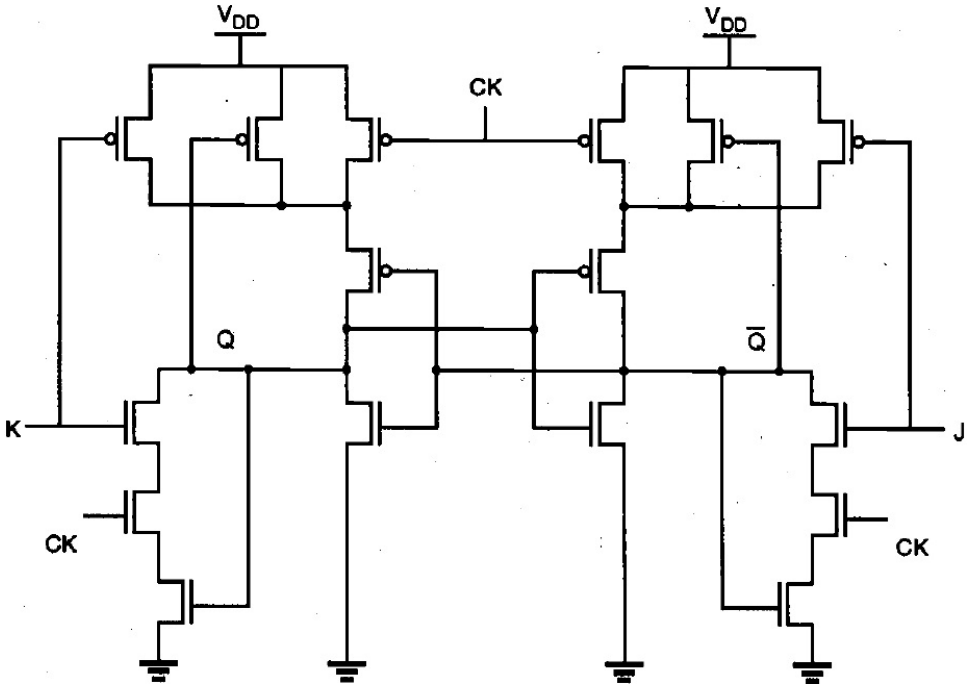

Figure 23: CMOS AOI realization of the JK latch

- This implementation is especially favorable in VLSI designs where area and power efficiency are critical.