STATE TABLE

A **state table** is a tabular representation of a **sequential circuit's behavior**, detailing the **present state**, the **inputs**, the resulting **next state**, and the **outputs**, essentially converting a state diagram into a precise list for implementation, showing how a circuit moves from one state to another based on clock pulses and inputs, using 0s and 1s for on/off states.

Key Components of a State Table:

- **Present State (PS):** The current condition of the flip-flops (memory elements).
- **Input(s) (I):** External signals that affect the circuit's transition.
- **Next State (NS):** The state the flip-flops will transition to after the next clock pulse.
- **Output(s) (O):** The value(s) the circuit produces during the present state.

1. **Represents Memory:** It describes a system that "remembers" past events, unlike combinational circuits.
2. **Tabular Form:** It organizes the pictorial state diagram into rows and columns, making it systematic.
3. **Defines Logic:** It specifies the exact requirements for the next-state logic and output logic, crucial for designing the actual hardware (flip-flops and gates).

Example Structure (for a simple circuit):

Present State (PS)	Input (I)	Next State (NS)	Output (O)
0	0	0	0
0	1	1	0
1	0	0	1
1	1	1	1

Purpose:

- **Design Specification:** Acts as a blueprint for building sequential circuits like counters and registers.
- **State Reduction:** Helps identify and eliminate redundant states to simplify the circuit, reducing cost and complexity.

State Diagram

A state diagram (or state machine diagram) is a crucial visual tool showing a sequential circuit's behavior: states (circles) represent memory conditions, and directed arrows (edges) show transitions between states triggered by inputs, detailing the next state and outputs, making it ideal for designing counters, control units, and protocol systems.