Synchronous Sequential Circuits

Sequential circuits

The information stored in the memory elements at any given time defines the present state of sequential circuits.

The present state & the external inputs determine the outputs & next state of sequential circuit.

Sequential ckt's can be specified by a time sequence of external inputs, internal states & outputs.

Eg: Counters, Registers

Memory element used in sequential ckt's is a Flip Flop which is capable of storing 2-bit binary information.

![Block diagram of sequential circuit]

Outputs not only dependent on the present input conditions but also on past history.

If past history is provided by feedback from
Combinational circuits

1) The O/P variables are at all times dependent on the combination of I/P variables.

2) Memory unit is not required in the combinational circuits.

3) Faster in speed than the combinational circuits.

4) Easy to design.

5) E.g.: Parallel Adder

Sequential circuits

The O/P variables depend not only on the present I/P variable, but they also depend upon past history of these I/P variables.

Memory unit is required to store the past history of I/P variables in sequential ckt.

Faster in speed than the combinational circuits.

Comparatively hard to design.

E.g.: Serial Adder

Clock

→ Clock signal is a particular type of signal that oscillates between a high & a low state and is utilized to co-ordinates actions of circuit.

→ It is produced by clock generator.

→ The time required to complete one cycle is called 'clock period' (or) 'clock cycle'.

www.studentsfocus.com
Types of Triggering:
1) Level Triggering
   * Positive level triggering
   * Negative level triggering
2) Edge Triggering
   * Positive edge triggering
   * Negative edge triggering

SR Latch

- 2-bit memory cell
- Two inverters 3,4 are connected to enter the digital information R [reset]
- $S$ set & $R$ for gate $A$ is Rs - Latch
- 'R' Reset. :: This is called as RS Latch.

Cases:

Case 1: $S = R = 0$
- Here $S = R = 1$
  1) If $A = 1$, $B$ & $R$ 11Ps are NAND gate 2 are both 1 & hence ORP $B = 0$, Since $B = 0$
  2) $S = 1$, ORP NAND gate 1 is 1 (i.e. $A = 1$

Case 2: $S = 1; R = 0$
- Since $S = 0$, the ORP of NAND gate 2, $B = 1$.

www.studentsfocus.com
The inputs \( S=1, R=0 \), makes \( Q=1 \).

i.e. Set state

Case 3:

\[ S=0 \quad R=1 \]

In this case, \( S=1 \) & \( R=0 \). Since

\[ R=0 \], the OLP of NAND gate 2, \( \overline{Q} = 1 \).

For NAND gate 1, both OLP \( \overline{Q} \) & \( \overline{S} \) are 1, thus OLP \( Q=0 \).

The \( S=0, R=1 \) makes \( Q=0 \), i.e. Reset State

Case 4:

\[ S=1 \quad R=1 \]

When \( S=R=1 \), both OLP \( \overline{Q} \) & \( \overline{S} \) try to become 1 which is not allowed hence this OLP condition is prohibited.

Gated SR latch

The circuit behaves as SR latch when \( EN=1 \), and retains its previous state when \( EN=0 \).

Logic symbol of Gated SR latch.
Latches & Flip-Flops

→ Both are basic building blocks of most sequential circuits.

→ The main difference between latches and flip-flops is the method used for changing their state.

Latch

- Controlled by an enable signal, they are level triggered, either true or false.

Flip-Flops

- FIF's are pulse on clock edge triggered.

SR Flip-Flop

Positive Edge triggered

The circuit is similar to SR latch except enable signal is replaced by the clock pulse (CP) followed by the true edge detector circuit. The edge detector circuit is a differentiator.

![SR Flip-Flop using NAND gate](attachment:SR_Flip_Flop_NAND.png)
Case 2: S = 0, R = 1 & clock pulse is applied

\[ Q_{n+1} = 0 \]  This is indicated in 2nd row of Truth Table.

Case 3: S = 1, R = 0 & clk pulse is applied,

\[ Q_{n+1} = 1 \]  This is indicated in the 3rd row of Truth Table.

Case 4: S = 1, R = 1 & clk pulse is applied,

The state of the FLF is undefined and therefore is indicated as indeterminate in 4th row of the Truth Table.

<table>
<thead>
<tr>
<th>CP</th>
<th>S</th>
<th>R</th>
<th>( Q_n )</th>
<th>( Q_{n+1} )</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No change</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>No change</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Reset</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Reset</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Set</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>Indeterminate</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>X</td>
<td>Indeterminate</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>No change</td>
</tr>
</tbody>
</table>

Logic Symbol:

![Logic Symbol Diagram]

www.studentsfocus.com
Negative Edge Triggered SR Flip Flop

Negative edge detector circuit is used & circuit will change response at the input of clock pulse.

Truth Table

<table>
<thead>
<tr>
<th>CP</th>
<th>S</th>
<th>R</th>
<th>Qn</th>
<th>Qn+1</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No Change</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>No Change (NC)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Reset</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Indeterminate</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Set</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Indeterminate</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>No Change</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>(NC)</td>
</tr>
</tbody>
</table>

CP & DIP waveforms for negative edge triggered CK: SR FF.

D-Flip Flop

Basic building block of D-FF is SR FF.

SR FF has 2 data input: S & R. The S input is made high to store 1 in FF & R input is made high to restore 0 in FF.

From Truth Table, when both inputs are same the output either does not change or it is invalid. [if D->00, NC & D->11, invalid]

In many practical applications, these input conditions are avoided.
These ilp conditions can be avoided by making them complement of each other. This modified SR flip-flop is known as D-flip-flop.

![SR Flip-flop diagram]

is D flip-flop using NAND gates.

The D ilp goes directly to 'S' ilp, & its complement is applied to R ilp.

Due to these connections, only 2 ilp conditions exist, either $S=0 \times R=1$ (on)

$S=1$ & $R=0$.

![D flip-flop truth table]

Truth Table of D Flip-flop

From truth table, we can realize that $Q_{n+1}$ function follows D ilp at time going edges of clock pulses.

Hence the characteristic equation as $Q_{n+1} = D$.

However, the $Q$ is this flip-flop.
According to the truth table of the D-Flip-Flop, when J = K = 0, the D-Flip-Flop does not change.

- J = K = 0: No change.
\( J = 1 \) \( K = 0 \) \( Q = 1 \) i.e. Set State

Case 3: \( J = 0 \), \( K = 1 \)
\( Q = 0 \), \( \bar{Q} = 1 \), when \( J = 0 \), \( K = 1 \) & \( Q = 0 \), \( S = 0 \), \( R = 0 \)
Since \( S = 0 \), there is no change in \( Q \)
\( Q = 0 \), \( \bar{Q} = 1 \)
\( R = 1 \), \( \bar{R} = 0 \) when \( J = 0 \), \( K = 1 \) & \( Q = 1 \), \( S = 0 \), \( R = 0 \)
From Truth Table it is Reset state &
\( Q \) will be '0'.

\( J = 1 \), \( K = 0 \) makes \( Q = 0 \) i.e. Reset state

Case 4: \( J = K = 1 \)
\( Q = 0 \), \( \bar{Q} = 1 \), when \( J = K = 1 \) & \( Q = 0 \), \( S = 1 \) \& \( R = 0 \)
From Truth Table it is Set state & \( Q \) in 1
\( Q = 1 \), \( \bar{Q} = 0 \) when \( J = K = 1 \) \& \( Q = 1 \), \( S = 0 \) \& \( R = 0 \)
From Truth Table it is Reset state & \( Q \) is '0'.

\( J = K = 1 \), toggles the \( Q \) output.

\[ T \quad \text{Flip Flop} \]

\[ \rightarrow \text{Toggle Flip Flop} \]

\[ \rightarrow \text{Is the modification of J K Flip} \]

\[ \rightarrow \text{Can be obtained from J K flip by connecting both flip flops, J K together.} \]

\[ \rightarrow \text{When } T = 0, J \& K \text{ together hence there is no change in } Q \]

\[ \rightarrow \text{When } T = 1, J, K = 1 \text{ & hence } Q \text{ toggles.} \]
characteristic Equations of Flip Flop & a Mark

<table>
<thead>
<tr>
<th>Flip Flop</th>
<th>Characteristic Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR</td>
<td>( Q_{n+1} = S + \overline{R} Q_n )</td>
</tr>
<tr>
<td>D</td>
<td>( Q_{n+1} = D )</td>
</tr>
<tr>
<td>JK</td>
<td>( Q_{n+1} = \overline{J} Q_n + K Q_n )</td>
</tr>
<tr>
<td>T</td>
<td>( Q_{n+1} = \overline{T} Q_n + T Q_n )</td>
</tr>
</tbody>
</table>

Clocked Sequential Circuits:
- Used as memory elements, which change their individual states in synchronism with periodic clock signal.
- The change in status of FF & change in status of entire circuit occurs at transition of clock signal.

Present state: The status of all state variables at some time \( t \), before next clock edge, represents condition called present state.

Next state: The status of state variable at some time, \( t + 1 \), represents a condition called next state.

Moore Model:
- Depends only on the present state of flip-flops.
- \([P] \) is used to determine the \([P] \) of the FFs. It is not used to determine the output.
11) Example of Moore Model (JK FF & AND gate)

The OIP \( Y \) is given as:

\[ Y = E_A \cdot E_B \]  \[ \text{[in this case]} \]

The above fig. JK FF & AND gate with I/P \( X \) & OIP \( Y \) is used.

11) General Representation of Moore Model

- Here the OIP varies in synchronism with clock I/P.

Mealy Model
- The OIP of sequential circuit depends on the present state of FF's and the I/P's.
12. Example of Mealy Model

- Change in the IIP within the clock pulse can not affect the state of FF, but they can affect the OLP of ckt.
- Due to this, if IIP variations are not synchronised with ckt, the derived OLP will also not be synchronised with clock, we get false OLP.

<table>
<thead>
<tr>
<th>Moore Model</th>
<th>Mealy Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Its OLP is a fn. of present state only.</td>
<td>Its OLP is a fn. of present state as well as past state.</td>
</tr>
<tr>
<td>2) IIP changes does not affect the OLP.</td>
<td>IIP changes may affect the OLP of ckt.</td>
</tr>
<tr>
<td>3) Requires more no. of states for implementing same fn.</td>
<td>It requires less no. of states for implementing same fn.</td>
</tr>
</tbody>
</table>
Representation of Sequential Circuits

State Diagram

→ Pictorial rep. of behaviour of sequential cir.

→ State is represented by a circle.
→ Transition between states is indicated by directed lines connecting the circles.

→ If a directed line connects a circle with itself, indicates that next state is same as present state.
→ The directed lines are labelled with the binary number inside each circle identifies the state represented by the circle.

→ The directed lines are labelled with two binary numbers separated by the symbol '/' . The '1'P value that causes the state transition is labelled 1st & '0'P value during the present state is labelled after '/' .
For convenience, the state diagram is converted to a state table. (On State synthesis Table)

### Mealy

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X = 0</td>
<td>X = 1</td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td>AB</td>
<td>Y</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>c</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>c</td>
<td>d</td>
<td>c</td>
</tr>
<tr>
<td>d</td>
<td>b</td>
<td>d</td>
</tr>
</tbody>
</table>

### Moore

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X = 0</td>
<td>X = 1</td>
<td>Y</td>
</tr>
<tr>
<td>AB</td>
<td>AB</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>c</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>c</td>
<td>d</td>
<td>c</td>
</tr>
</tbody>
</table>

### Transition Table

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X = 0</td>
<td>X = 1</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>AB</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Analysis of Clocked Sequential Circuits:

1) Determine the F/F, I/P eqns., & the O/P eqns. from the sequential ckt.
2) Derive the transition eqns.
3) Plot the next step map for each F/F.
4) Plot the transition table.
5) Draw the state table.

Design of Clock Sequential Circuits:

1) Obtain the state table from the ckt. information such as state diagram, timing-diagram, or other pertinent information.
2) The number of states may be reduced by state reduction technique. If the sequential ckt. can be categorized by I/P-O/P relationships independent of the number of states.
3) Assign binary values to each state in the state table.
4) Determine the no. of F/F needed to design assign a letter symbol to each.
5) Choose the type of F/F to be used.
6) From the state table, derive the circuit excitation & O/P tables.
7) Using K-map (or) any other simplification method, derive the excitation tables.
8. Draw the logic diagram.

Stat Reduc.

1) Determine the state table for the given state diagram.

2) Find the equivalent states.

\[ \text{Eq} \quad \text{Minimize the state Table shown given below.} \]

\[
\begin{array}{|c|c|}
\hline
\text{Present State} & \text{Next State, } x(01P) \\
\hline
A & B, 0 \\
B & B, 0 \\
C & B, 0 \\
D & E, 1 \\
E & B, 0 \\
\hline
\end{array}
\]

Slove-

1) Find the equivalent states:

\[
\begin{array}{|c|c|c|}
\hline
\text{Present State} & \text{Next State, } x(01P) & \text{Equivalent States} \\
\hline
A & B, 0 & C, 0 \\
B & B, 0 & D, 0 \\
C & B, 0 & C, 0 \\
D & E, 1 & D, 0 \\
E & B, 0 & D, 0 \\
\hline
\end{array}
\]

\[
\text{The states } A, C \text{ and states } B \& E \text{ generate the same next state.}
\]

\[
\text{Hence, states } A, B, C \text{ are equivalent.}
\]
Replace redundant states with equivalent states.

**Minimized State Table**

| Present State | Next State, \( Z(0|P) \) |
|---------------|--------------------------|
| \( x = 0 \)   | \( B, 0 \)               |
| \( x = 1 \)   | \( A, D \)               |
| \( A \)       | \( B, 0 \)               |
| \( B \)       | \( D, 0 \)               |
| \( D \)       | \( B, 1 \)               |
| \( D \)       | \( A, 0 \)               |

**Counters:**

- Positive edge triggered \( n \)-bit counter
- Negative edge triggered \( n \)-bit counter

Counter is a register capable of counting the no. of clock pulses arriving at its \( CLK \) input.

![Diagram of counters](image)

\( a) \) Positive edge triggered \( n \)-bit counter
\( b) \) Negative edge triggered \( n \)-bit counter

External clock is applied to the \( CLK \) input.
Counter can be five edge triggered on five edge triggered.

The maximum count that the binary counter can count is 2^n - 1.

After reaching the maximum count, the counter resets to 0 on arrival of the next clk pulse and it starts counting again.

Synchronous Counters

1. FFs are connected so that the Q/P of 1st FF drives the clock for the next FF.

2. All FFs are not clocked simultaneously.

3. Logic circuit is very simple even for more number of states.

4. Low speed as clk is propagated through no. of FFs before it reaches last FF.

A synchronous counter

A synchronous counter

A synchronous counter

Design involves complex logic ckt.

As no. of stages?

An all clk to simultaneous

given to all FFs.

Hence there are

High speed counters

are preferred when given design.
Modulus Counter:

- Total no. of counts = \(2^n\) 
- Stable states a counter can indicate is called Modulus.
- Modulus of a 4 stage counter would be 16, since it is capable of indicating 0000 to 1111.

- The term "modulo" is used to describe the count capability of counters.

\[ \text{Eq:} \mod 6 \text{ counter goes through states 0 to 5} \]
\[ \mod 4 \text{ counter goes through states 0 to 3}. \]