Design of Synchronous Sequential Circuits

October 12, 2020

Introduction

- In case of a sequential circuit, the outputs depend not only on the present inputs but also on the internal state of the circuit.
- The internal state also changes with time.
- The number of states of a circuit is finite, and hence, a sequential circuit is also called a Finite State Machine (FSM).
- Most of the practical circuits are sequential in nature.
- In case of combinational circuit, truth table, Boolean expressions can be used to represent the behaviour of the circuit. Similarly, a FSM can be represented either in the form of state table or state diagram.

Examples

- A circuit to detect 3 or more consecutive 1's in a serial bit stream.
- The bits are applied serially in synchronism with clock.
- The output will become 1 when it detects 3 or more consecutive 1's in the stream.

Input	0 1 1 0 1 1 1 1 0 1 0 1 1 1
output	0 0 0 0 0 0 1 1 0 0 0 0 0 1

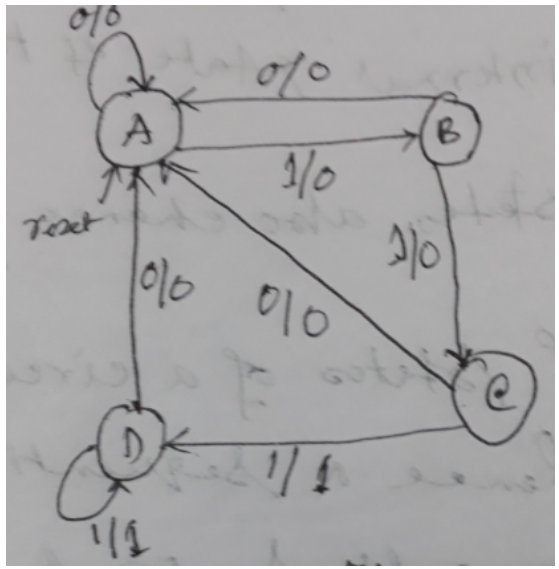

Figure: State Diagram of the proposed circuit

Examples

Table: State Table of the Proposed Circuit

	I/P = 0		I/P = 1	
PS	NS	O/P	NS	O/P
A	A	0	B	0
B	A	0	C	0
C	A	0	D	1
D	A	0	D	1

Mealy and Moore FSM

- Formal definition of FSM: A deterministic FSM can be mathematically defined as a 6 tuples: $(\Sigma, \Gamma, S, s_0, \delta, \omega)$
- $\Sigma \Rightarrow$ Set of input combinations. $\Gamma \Rightarrow$ Set of output combinations. $S \Rightarrow$ A finite set of states. $s_0 \subset S \Rightarrow$ is the initial state. $\delta \Rightarrow$ is the state transition function (it gives the next state). $\omega \Rightarrow$ is the output function.
- $\delta : S \times \Sigma$ - Present state and present inputs determine the next state (NS).
- $\omega : S \times \Sigma \rightarrow \Gamma$ For Mealy machine (output depends on PS + Inputs)
- For Moore machine: $\omega : S \rightarrow \Gamma$ (output depends on the present state).
- Pictorial Depiction.

Design of Synchronous Sequential Circuits

- From a description of the problem, obtain a state table or state diagram.
- Check whether the table contains any redundant states; if so remove them.
- Select a state assignment and determine the type of Flip-flop.
- Derive transition and output tables.
- Derive the excitation table and obtain excitation and output functions.
- Minimize the function and obtain the circuit diagram.

Construction of State diagram/Table from Specification

- A state diagram specifies the different states of a FSM, the condition under which state changes occur, and the corresponding output.
- States are denoted as circles, and labelled with unique symbols.
- Transitions are represented as directed arrows between pair of states.
- Each transition is labelled by α/β, where α denotes input combination and β denotes output combination.
- A state table is an alternative way of representing state diagram.
- For each value of present state, it specifies the next state and output for every input combination.

Example -1

Design a serial parity detector

- A continuous stream of bits is fed to a circuit in synchronism with clock. The circuit will generating a bit stream as output, where a 0 will indicate even number of 1's seen so far and a 1 will indicate odd numbers of 1's seen so far.
- The state diagram of the aforesaid problem is given below:

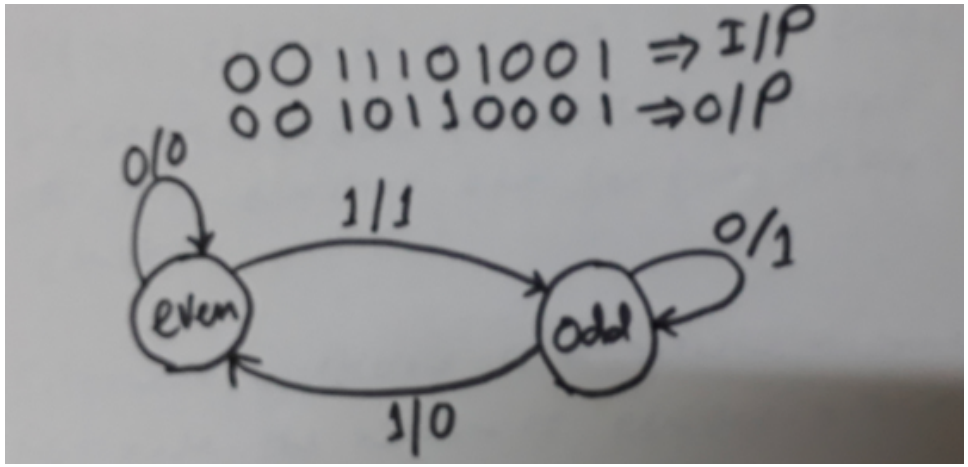

Figure: State Diagram of the proposed circuit

Table: State Table of the Proposed Circuit

	I/P = 0		I/P = 1	
PS	NS	O/P	NS	O/P
even	even	0	odd	1
odd	odd	1	even	0

Example-2

Design a sequence detector

- A circuit accepts a serial bit stream 'X' as input and produce a serial bit stream 'Z' as output.
- Whenever the bit pattern '0110' appears in the input stream, its output $Z = 1$ and at all other times, $Z = 0$.
- Overlapping occurrences of the patterns are also detected.
- This is an example of Mealy Machine.
- The state diagram of the proposed circuit is given below:

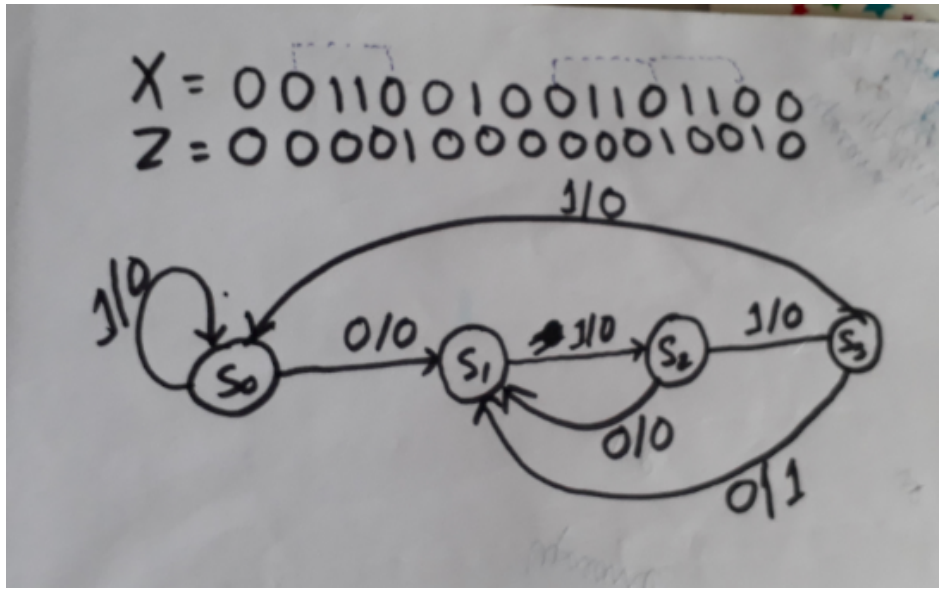

Figure: State Diagram of the proposed circuit

State Table Minimization

- Basic Concept:
  - Identify equivalent states and merge them into a single state.
  - Can lead to more compact representation.
- State Equivalence: Two states S_i and S_j are said to be equivalent if and only if for every single input sequence X, the outputs are the same and the next states are equivalent.
  $\omega(S_i, X) = \omega(S_j, X)$ and $\delta(S_i, X) \equiv \delta(S_j, X)$
  Where $\omega(S_i, X)$ is the output for the present state S_i and input X and $\delta(S_i, X)$ is the corresponding next state.

Steps for Minimization using Implication Table

1. Construct an Implication table that contains a square for every pair of states.
2. Compare each pair of rows in the state table, if the output associated with state S_i and S_j are different, place 'X' in square S_i - S_j to indicate S_i and S_j are not equivalent. If the outputs are same, place the implied pair in the square S_i - S_j. (if the next states of S_i and S_j are m and n respectively for some input X, then m - n is an implied pair.
3. Go through the table square by square if square S_i - S_j contains the implied pair m -n and square m - n contains a 'X' then, S_i and S_j are not equivalent and place 'X' in square S_i - S_j.
4. If any 'X' is added in step 3, repeat step 3 until no more 'X' can be added.
5. For each square S_i - S_j that does not contain 'X', the states S_i and S_j are equivalent.

Example

Consider the following state table:

Table: An example state table

	I/P = 0		I/P = 1	
PS	NS	O/P	NS	O/P
a	a	0	b	0
b	c	0	d	0
c	a	0	d	0
d	e	0	f	1
e	a	0	f	1
f	g	0	f	1
g	a	0	f	1

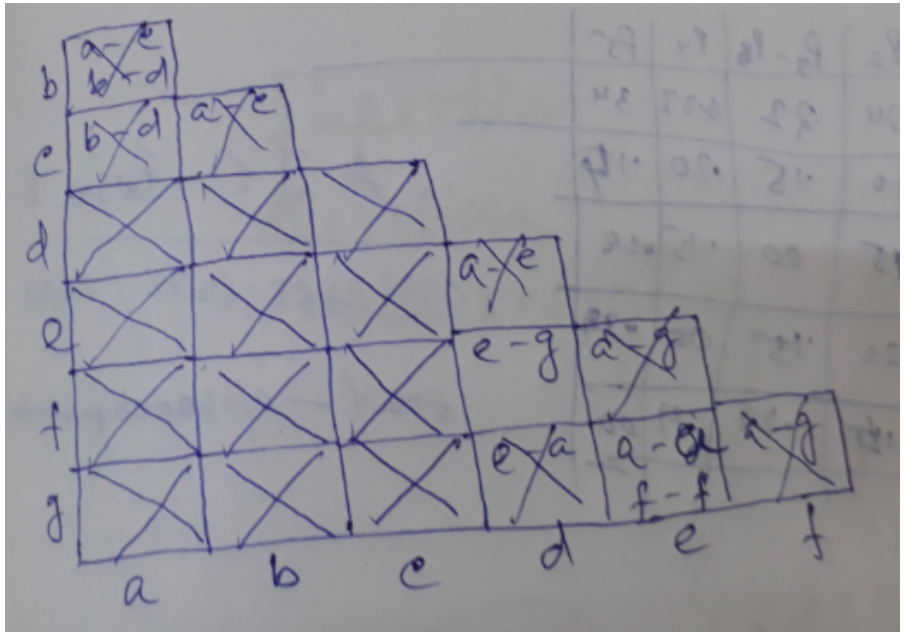

Figure: Implication table for the above mention state table

Serial Adder Design

- A serial adder has two external inputs and one external outputs.
- The state table and state diagram are shown below:

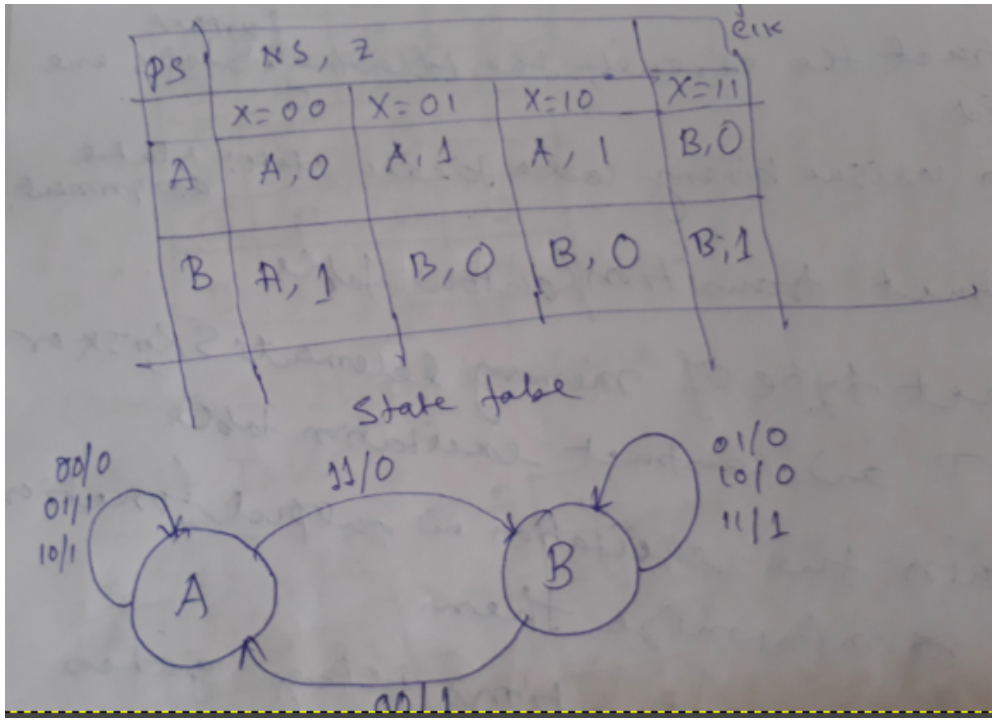

Figure: State table and state diagram of serial adder.

State Assignment and Transition/output Table

- There are two states and so one bit is sufficient. Suppose A = 0 and B = 1. Note that this not unique.
- Transition/output Table is shown below:

Table: Transition/output Table of a Serial Adder.

	NS				Output (Z)			
PS	00	01	10	11	00	01	10	11
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

- Selection of memory element: Suppose we use D flip-flop

Excitation and Output Table

- The excitation/output table is show below:

Table: Transition/output Table of a Serial Adder.

PS(y)	X_1	X_2	NS	D	Z
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	1	1

- The expression for $Z = X_1 \oplus X_2 \oplus y$ and $D = X_1X_2 + X_1y + X_2y$

Example 2: Design of a Sequence Detector

- A circuit accepts a serial bit stream 'X' as input and produce a serial bit stream 'Z' as output.
- Whenever the bit pattern '0110' appears in the input stream, its output $Z = 1$ and at all other times, $Z = 0$.
- Overlapping occurrences of the patterns are also detected.
- This is an example of Mealy Machine.
- The state diagram of the proposed circuit is given below:

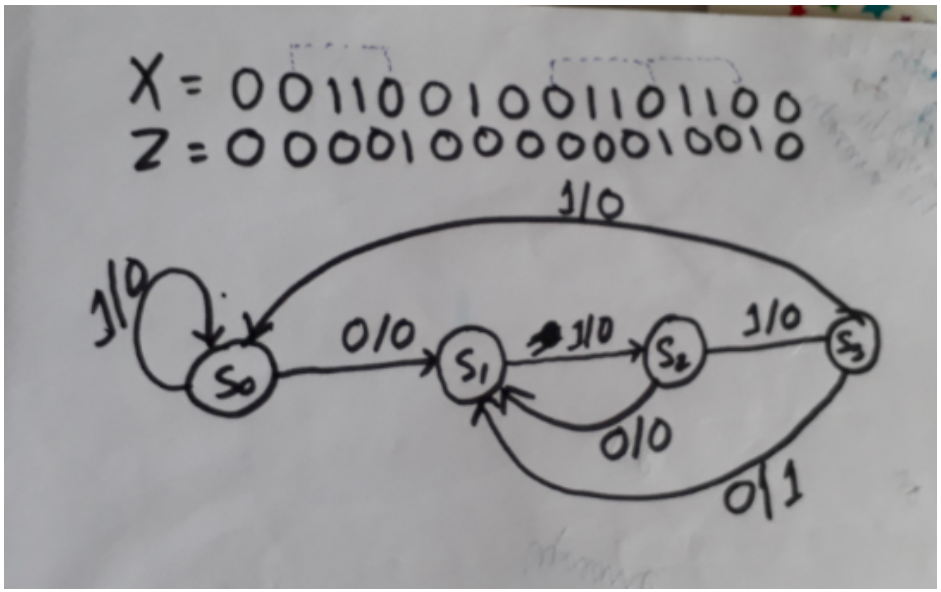

Figure: State Diagram of the proposed circuit

Example 2: Design of a Sequence Detector

- The state table of the sequence detector is as follows:

PS	NS, Z	
	$X = 0$	$X = 1$
S_0	S_1, 0	S_0, 0
S_1	S_1, 0	S_2, 0
S_2	S_1, 0	S_3, 0
S_3	S_1, 1	S_0, 0

- The states are assigned as $S_0 = 00$, $S_1 = 01$, $S_2 = 10$, and $S_3 = 11$. The transition/output table is shown below:

PS	NS		Z	
	$X = 0$	$X = 1$	$X = 0$	$X = 1$
00	01	00	0	0
01	01	10	0	0
10	01	11	0	0
11	01	00	1	0

Example 2: Design of a Sequence Detector

- Suppose we use T flip-flop as memory element, then the excitation table is as follows:

X	y_1	y_2	Y_1	Y_2	T_1	T_2	Z
0	0	0	0	1	0	1	0
0	0	1	0	1	0	0	0
0	1	0	0	1	1	1	0
0	1	1	0	1	1	0	1
1	0	0	0	0	0	0	0
1	0	1	1	0	1	1	0
1	1	0	1	1	0	1	0
1	1	1	0	0	1	1	0

The expressions for T_1, T_2 and Z are as follows:
$T_1 = \overline{X}y_1 + Xy_2$, $T_2 = \overline{X}\overline{y_2} + Xy_2 + y_1y_2$ and $Z = \overline{X}y_1y_2$