Indian Institute of Technology
Hyderabad

Department of Electrical Engineering

Experiment 9: Sequence Detector

(Moore Machine Implementation using Flip-Flops and Logic Gates)

Submitted By:
Krishna Hanumanth Patil
Roll No: EE24BTECH11036
Deepak Kumar Ahirwar
Roll No: EE24BTECH11014

Submitted To:
Prof. Gajendranath Chaudhury
Department of Electrical Engineering

Course Code: EE1501
Course Name: Electric Circuits Laboratory

April 21, 2025

Contents

1 What is a Finite State Machine

A Finite State Machine (FSM) is a mathematical model of computation used to represent the behavior of sequential systems. It operates in one of a finite number of states at any given time and transitions between states in response to external inputs. The machine's next state and output are determined by its current state and input, governed by well-defined transition and output functions.

The FSM is characterized by:

- **External Inputs:** Signals received from outside the system, used to determine transitions.
- **Externally Visible Outputs:** Signals generated based on the internal state and/or inputs.
- **Internal State:** Stored information that represents the current status of the system, typically maintained using flip-flops.

The FSM transitions between a finite number of internal states in response to clocked inputs, producing outputs according to its defined behavior.

2 Finite State Machines: Moore and Mealy Models

The finite state machine (FSM) can be classified into two main types based on how the outputs are generated: the **Moore Model** and the **Mealy Model**. These two models define the relationship between the current state, inputs, and outputs in different ways, affecting how the system's behavior is observed.

In an FSM, the behavior is typically classified as either **Moore** or **Mealy**, depending on how outputs are derived.

2.1 Moore Model

In the **Moore Model**, the **outputs** depend solely on the **current state** of the machine. The output remains constant for each state and only changes when the system transitions to a new state.

2.1.1 Key Features of the Moore Model:

- Outputs depend only on the current state.
- The output changes only when the system enters a new state.
- Simpler to design because the output is directly linked to the state, and no input is involved in determining the output.

Mealy Model

In the **Mealy Model**, the **outputs** depend on both the **current state** and the **inputs**. This allows the output to change immediately when the input changes, even without a state transition.

Key Features of the Mealy Model:

- Outputs depend on both the current state and the inputs.
- The output can change immediately with the input, without needing a state transition.
- More compact and efficient since outputs can change based on inputs without transitioning to a new state.

For this experiment, we are using the **MOORE MODEL** to detect a sequence, namely **11011** ; in fact we will try to detect it overlappingly and repeatedly .

3 Hardware Requirements for Moore Model Sequence Detector

- 3 J-K flip flops (2 IC 74LS76N) (T flip flops would be even better)
- 10 and gates or 4 3-input and gate ic (IC 74LS11N)
- 6 or gates or 3 2-input or gate ic (IC 74LS32N)
- one nand gate (IC 74HC00N)
- an led or any thing that can be used to show that the sequence is detected
- Jumper wires
- Breadboards

4 State Diagram

Below shown diagram represents the required FSM that as to be implemented for our experiment .

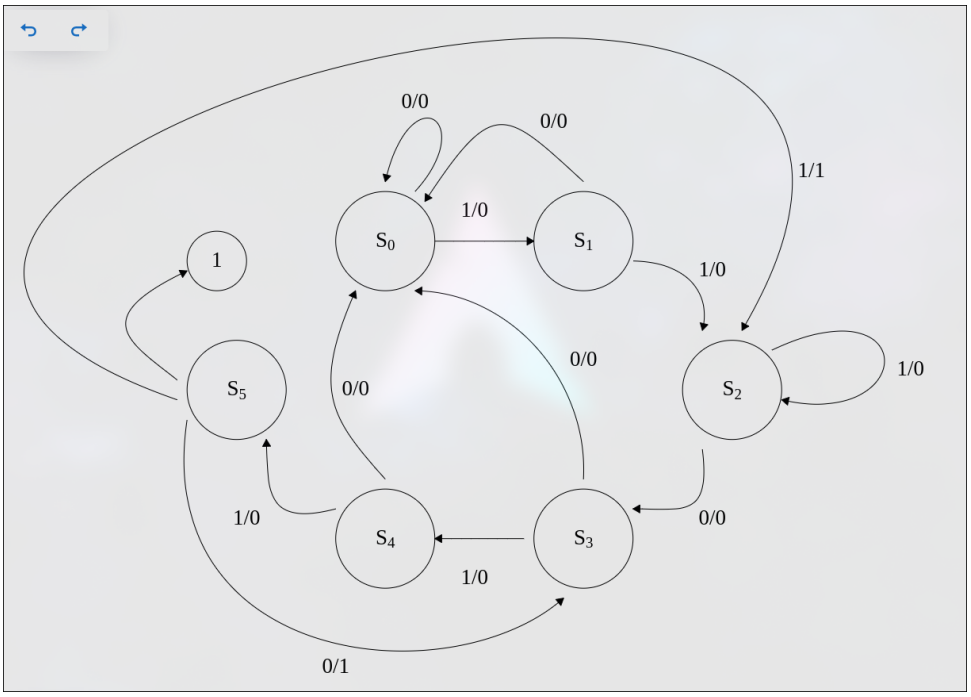

Figure 1: State Diagram

As shown in the diagram, the sequence detector can be represented as a set of different states, we use it to get a state transition table which is discussed in the next section .

5 State Transition Table

Table 1: State Transition Table with Q(t) and Q(t+1)
Group Labels

Q(t)				Q(t) T Flip-Flop Inputs			Q(t+1) State			Y
Q_2	Q_1	Q_0	X	T_2	T_1	T_0	Q_2	Q_1	Q_0	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0	0	0	0
0	0	1	1	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0	1	1	0
0	1	0	1	0	0	0	0	1	0	0
0	1	1	0	0	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0
1	0	0	0	1	0	0	0	0	0	0

Q(t)				Q(t) T Flip-Flop Inputs			Q(t+1) State			Y
Q_2	Q_1	Q_0	X	T_2	T_1	T_0	Q_2	Q_1	Q_0	
1	0	0	1	0	0	1	1	0	1	0
1	0	1	0	1	1	0	0	1	1	1
1	0	1	1	1	1	1	0	1	0	1
1	1	0	0	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X

This is the state transition table , using this we determine the SOP form of the boolean functions for T_0, T_1, T_2, y so that the state machine be implemented using T flip flops and obviously some logic gates.This can be easily done using K-maps.

6 Determining the boolean functions

Let's, start with the K-map for Q_0,

$Q_2Q_1 \backslash Q_0x$	00	01	11	10
00	0	1	1	1
01	1	0	1	1
11	X	X	X	X
10	0	1	1	0

Now , after simplifying, we get

$$\boxed{T_0 = \overline{Q_2}Q_0 + Q_1\overline{x} + \overline{Q_1}x}$$

Next, for Q_1,

$Q_2Q_1 \backslash Q_0x$	00	01	11	10
00	0	0	1	0
01	0	0	1	1
11	X	X	X	X
10	0	0	1	1

Now, after simplifying, we get,

$$\boxed{T_1 = Q_2Q_0 + Q_1Q_0 + Q_0x}$$

Next for Q_2,

$Q_2Q_1 \backslash Q_0x$	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	X	X	X	X
10	1	0	1	1

Now, after simplifying, we get,

$$\boxed{T_2 = Q_2x + Q_2Q_0 + Q_1Q_0x}$$

Finally, for y,

$Q_2Q_1 \backslash Q_0x$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	0	0	1	1

So,

$$\boxed{y = Q_2Q_0}$$