CHAPTER VIII

FINITE STATE MACHINES (FSM)

STATE MACHINES

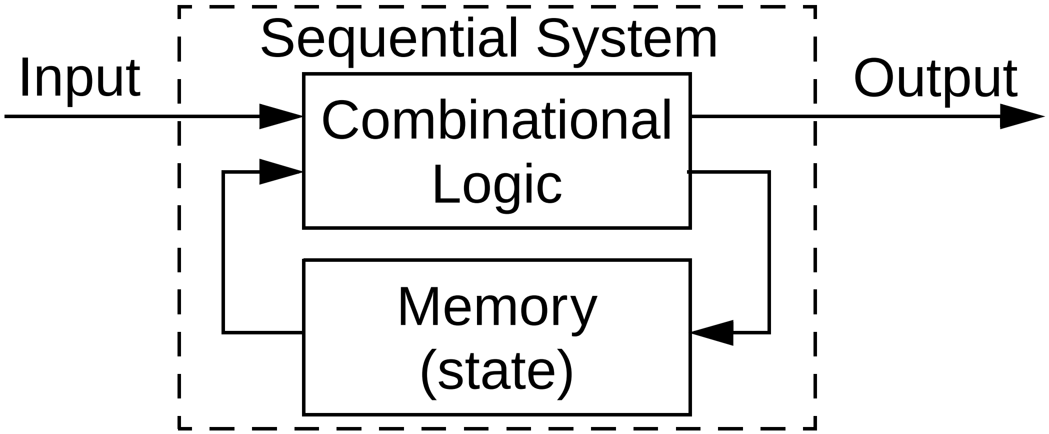
INTRODUCTION

- From the previous chapter we can make simple memory elements.
  - Latches as well as latches with control signals
  - Flip-flops
  - Registers
- The goal now is to use the memory elements to hold the running state of the machine.
  - The state of the machine can be used to perform sequential operations.
  - This chapter will discuss how to represent the state of the machine for design and communication purposes.

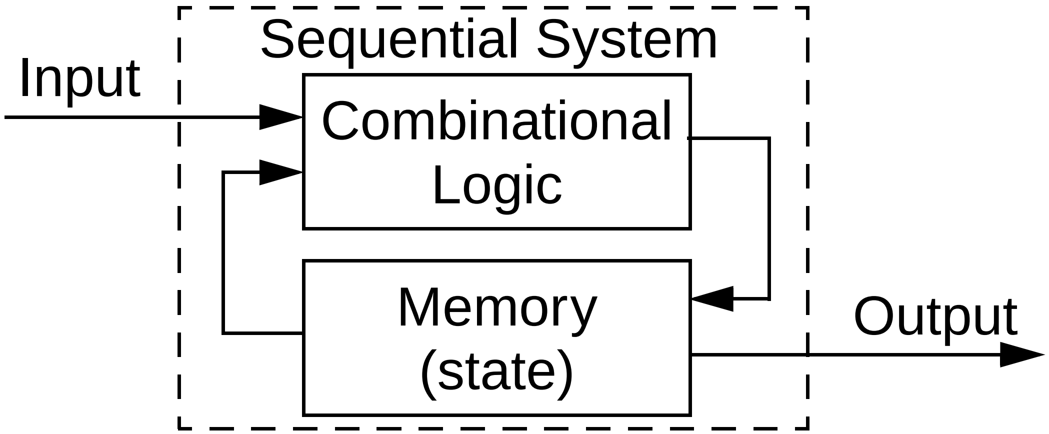
STATE MACHINES

MEALY & MOORE MACHINES

- **Mealy machine**
  - Sequential system where output depends on current input and state.



- **Moore machine**
  - Sequential system where output depends only on current state.



STATE MACHINES

SYNC. & ASYNC. SYSTEMS

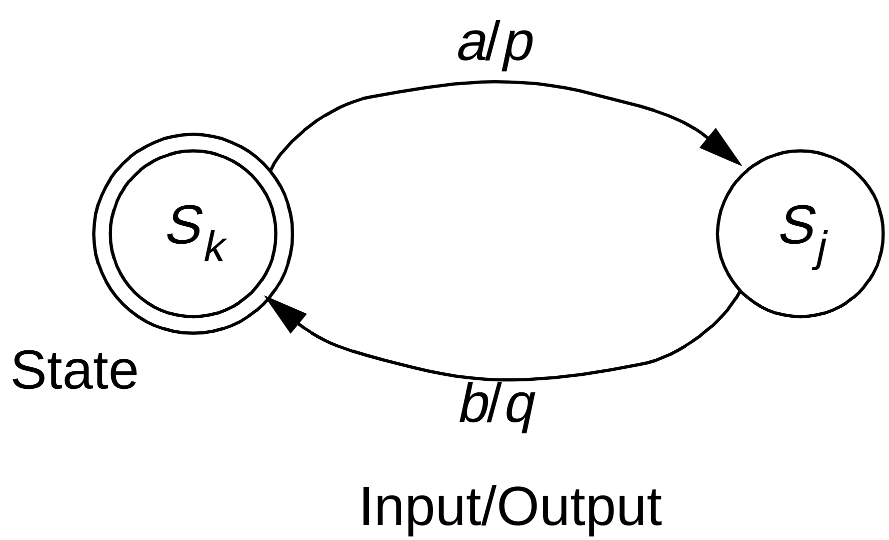
- **Synchronous sequential system**
  - Behaviour depends on the inputs and outputs at discrete instants of time.
  - Flip-flops, registers, and latches that are enabled/controlled with a signal derived from clock form a synchronous sequential system.
- **Asynchronous sequential system**
  - Behaviour depends on inputs at any instant of time.
  - Latches without control signals behave in an asynchronous manner.

- The state machines discussed in this chapter will be synchronous sequential systems (i.e. controlled by a clock)
  - This allows us to form timed Boolean functions such as
    - $\mathbf{N}(t) = \mathbf{D_A}(t + 1)$ where $\mathbf{N}$ is the next state of a D flip-flop $\mathbf{D_A}$.

STATE DIAGRAMS

ELEMENTS OF DIAGRAMS

- A state diagram represents a finite state machine (FSM) and contains
  - **Circles**: represent the machine states
    - Labelled with a binary encoded number or S_k reflecting state.
  - **Directed arcs**: represent the transitions between states
    - Labelled with input/output for that state transition.



Input: $x(t) \in \{a, b\}$

Output: $z(t) \in \{p, q\}$

State: $s(t) \in \{S_k, S_j\}$

Initial state: $s(0) = S_k$

STATE DIAGRAMS

PROPERTIES

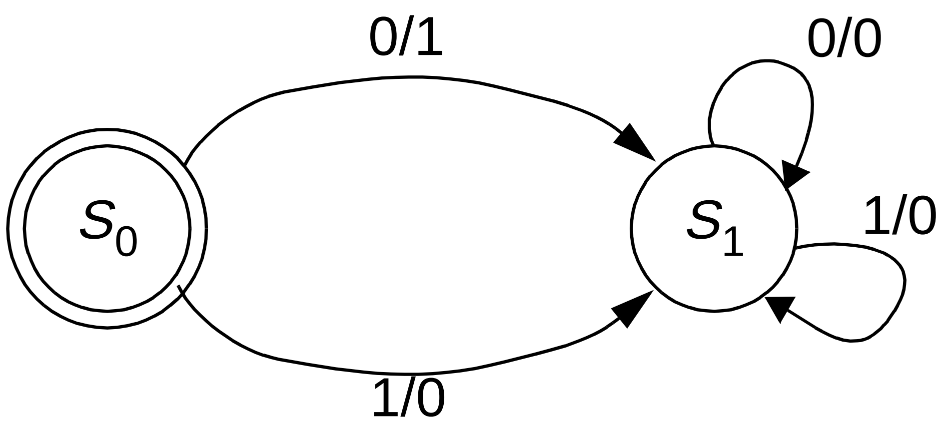
- Some restrictions that are placed on the state diagrams:
  - FSM can only be in **one state at a time**!
    - Therefore, only in one state, or one circle, at a time.
  - State transitions are followed only **on clock cycles**. (synchronous!)

- Mealy machines and Moore machines can be labelled differently.
  - **Mealy machine**: Since output depends on state and inputs:
    - Label directed arcs with **input/output** for that state transition.
  - **Moore machine**: Since output depends only on state:
    - Label directed arcs with **input** for that state transistion.
    - Label state circles with S_k/**output**.

STATE DIAGRAMS

STATE DIAGRAM EXAMPLES

- The following is a simple example. What does this state machine do?



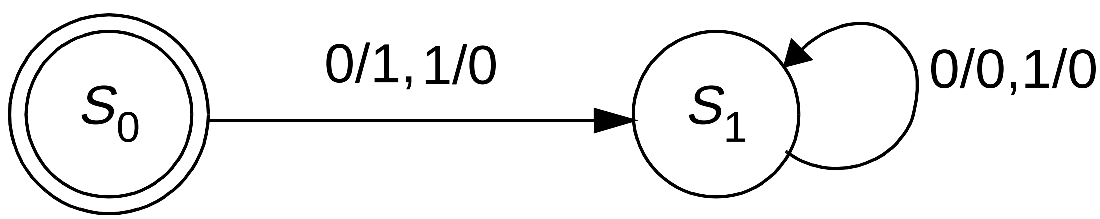
Input: $x(t) \in \{0, 1\}$

Output: $z(t) \in \{0, 1\}$

State: $s(t) \in \{S_0, S_1\}$

Initial state: $s(0) = S_0$

- Here is a simplified way of forming the above state machine.

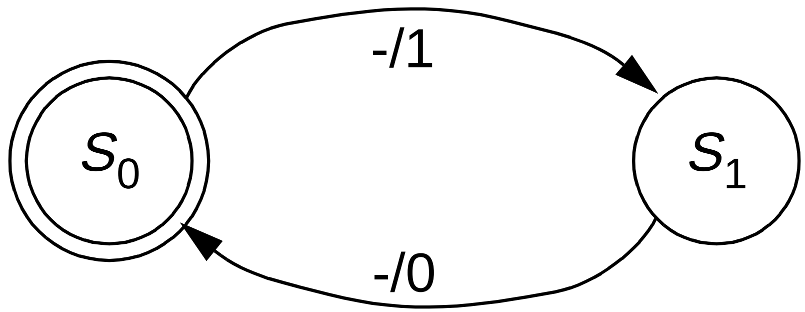


- An input of 0 or 1 causes the transition with output 1 and 0, respectively.

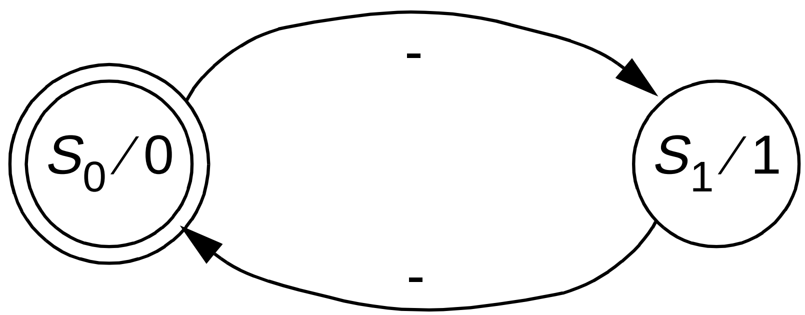
STATE DIAGRAMS

BIT FLIPPER EXAMPLE

- Consider the simple bit flipper looked at the in previous chapter. How would a state diagram be formed?
  - Below is one possible way of drawing the state diagram for the bit flipper.



  - Since the bit flipper is a Moore machine, the state diagram can also be



STATE DIAGRAMS

PATTERN DETECT EXAMPLE

- Suppose we want a sequential system that has the following behaviour

Input: $x(t) \in \{0, 1\}$

Output: $z(t) \in \{0, 1\}$

Function: $$z(t) = \begin{cases} \mathbf{1} & \text{if } x(t-3, t) = \mathbf{1101} \\ \mathbf{0} & \text{otherwise} \end{cases}$$

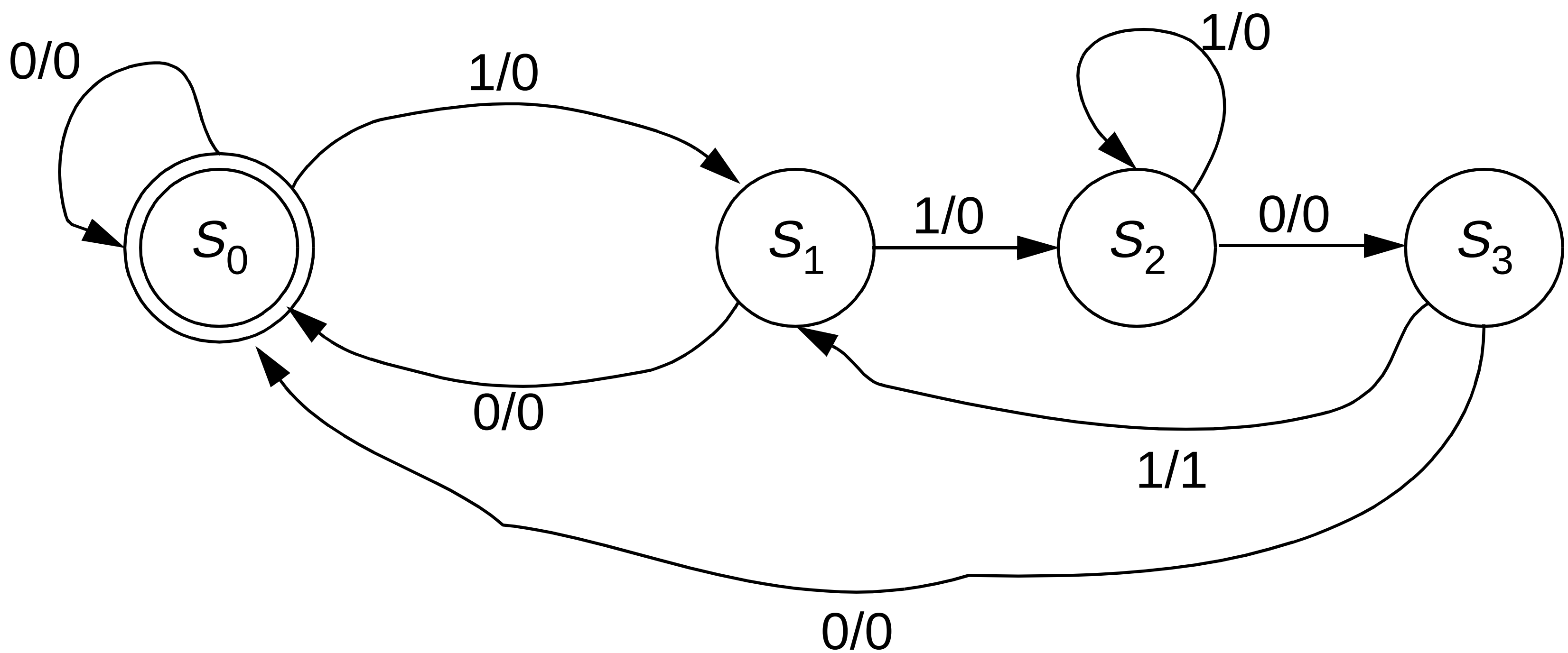
- Effectively, the system should output a **1** when the last set of four inputs have been **1101**.
- For instance, the following output $z(t)$ is obtained for the input $x(t)$

t	0123456789...
$x(t)$	100100100100**1101**0**1101101**00**1101**001
$z(t)$	???0000000000000**1**0000**1**00**1**00000**1**000

STATE DIAGRAMS

PATTERN DETECT EXAMPLE

- The following state diagram gives the behaviour of the desired 1101 pattern detector.
  - Consider S_0 to be the initial state, S_1 when first symbol detected (**1**), S_2 when subpattern **11** detected, and S_3 when subpattern **110** detected.



STATE TABLES

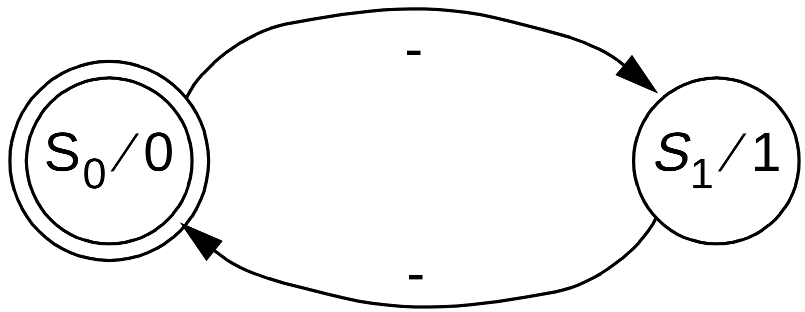
INTRODUCTION

- State tables also express a systems behaviour and consists of
  - Present state
    - The present state of the system, typically given in binary encoded form or with S_k. So, a state of S_5 in our state diagram with 10 states would be represented as 0101 since we require 4 bits.
  - Inputs
    - Whatever external inputs used to cause the state transitions.
  - Next state
    - The next state, generally in binary encoded form.
  - Outputs
    - Whatever outputs, other then the state, for the system. Note that there would be no outputs in a Moore machine.

STATE TABLES

BIT FLIPPER EXAMPLE

- Consider again the bit flipper example with state diagram



- The state table for this state diagram would be

Present State	Input	Next State	Output
S_0 or **0**	**-**	**1**	**-**
S_1 or **1**	**-**	**0**	**-**

STATE TABLES

TRANSLATE FROM DIAGRAM

- From a state diagram, a state table is fairly easy to obtain.
  - Determine the number of states in the state diagram.
  - If there are m states and n 1-bit inputs, then there will be $m2^n$ rows in the state table.
    - Example: If there are 3 states and 2 1-bit inputs, each state will have $2^2 = 4$ possible inputs, for a total of 3*4=12 rows.
  - Write out for each state, the 2^n possible input rows.
  - For each state/input pair, follow the directed arc in the state diagram to determine the next state and the output.

STATE TABLES

PATTERN DETECT EXAMPLE

- If we consider the pattern detection example previously discussed, the following would be the state table.

Present State	P_1	P_0	Input X	Next State	N_1	N_0	Output Z
S_0 or	**0**	**0**	**0**	S_0 or	**0**	**0**	**0**
S_0 or	**0**	**0**	**1**	S_1 or	**0**	**1**	**0**
S_1 or	**0**	**1**	**0**	S_0 or	**0**	**0**	**0**
S_1 or	**0**	**1**	**1**	S_2 or	**1**	**0**	**0**
S_2 or	**1**	**0**	**0**	S_3 or	**1**	**1**	**0**
S_2 or	**1**	**0**	**1**	S_2 or	**1**	**0**	**0**
S_3 or	**1**	**1**	**0**	S_0 or	**0**	**0**	**0**
S_3 or	**1**	**1**	**1**	S_1 or	**0**	**1**	**1**

STATE TABLES

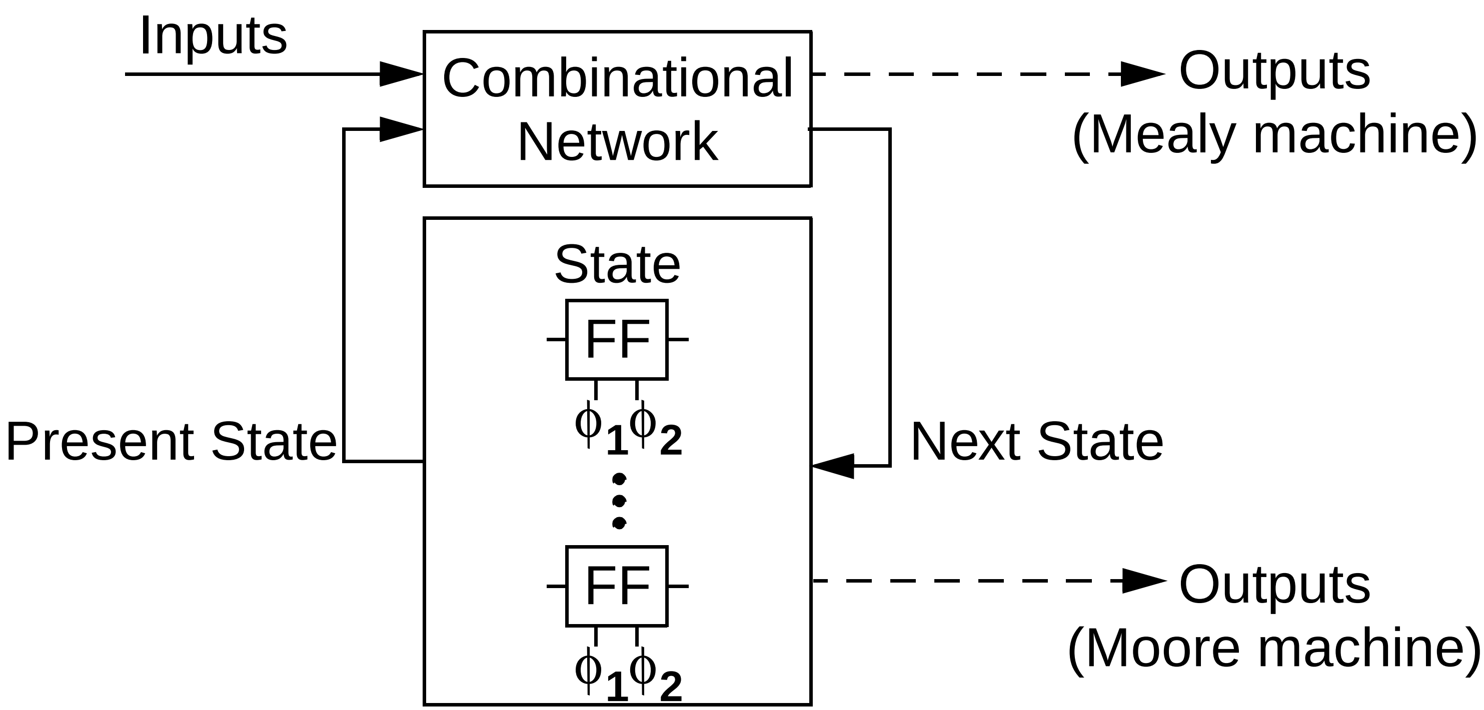
TRANSLATE TO DIAGRAM

- If given a state table, the state diagram can be developed as follows.
  - Determine the number of states in the table and draw a state circle corresponding to each one.
    - Label the circle with the state name for a Mealy machine.
    - Label the circle with the state name/output for a Moore machine.
  - For each row in the table, identify the present state circle and draw a directed arc to the next state circle.
    - Label the arc with the input/output pair for a Mealy machine.
    - Label the arc with the input for a Moore machine.

SEQ. CIRCUITS

INTRODUCTION

- With the descriptions of a FSM as a state diagram and a state table, the next question is how to develop a sequential circuit, or logic diagram from the FSM.
- Effectively, we wish to form a circuit as follows.



SEQ. CIRCUITS

FROM STATE TABLE

- The procedure for developing a logic circuit from a state table is the same as with a regular truth table.
  - Generate Boolean functions for
    - each external outputs using external inputs and present state bits
    - each next state bit using external inputs and present state bits
  - Use Boolean algebra, Karnaugh maps, etc. as normal to simplify.
  - Draw a register for each state bit.
  - Draw logic diagram components connecting external outputs to external inputs and outputs of state bit registers (which have the present state).
  - Draw logic diagram components connecting inputs of state bits (for next state) to the external inputs and outputs of state bit registers (which have the present state).

SEQ. CIRCUITS

PATTERN DETECT EXAMPLE

- Following the procedure outlined, Boolean functions for the pattern detector state table can be formed using Karnaugh maps as follows.

N_1:

X \ P_1P_0	00	01	11	10
0	0	0	1	0
1	1	1	0	0

N_0:

X \ P_1P_0	00	01	11	10
0	0	1	0	0
1	1	0	1	0

Z:

X \ P_1P_0	00	01	11	10
0	0	0	0	0
1	0	0	1	0

$$N_1 = X\overline{P_1} + \overline{X}P_1P_0$$

$$N_0 = \overline{X}\,\overline{P_1}P_0 + XP_1P_0 + X\overline{P_1}\,\overline{P_0} = \overline{X}\,\overline{P_1}P_0 + X(\overline{P_1 \oplus P_0})$$

$$Z = XP_1P_0$$

SEQ. CIRCUITS

PATTERN DETECT EXAMPLE

- Notice that the previous Boolean functions can also be expressed with time as follows.

$$\mathbf{N_1}(t) = \mathbf{P_1}(t+\mathbf{1}) = \mathbf{X}(t)\cdot\overline{\mathbf{P_1}(t)} + \overline{\mathbf{X}(t)}\cdot\mathbf{P_1}(t)\cdot\mathbf{P_0}(t)$$

$$\begin{aligned}\mathbf{N_0}(t) = \mathbf{P_0}(t+\mathbf{1}) &= \overline{\mathbf{X}(t)}\cdot\overline{\mathbf{P_1}(t)}\cdot\mathbf{P_0}(t) + \mathbf{X}(t)\cdot\mathbf{P_1}(t)\cdot\mathbf{P_0}(t) \\ &\quad + \mathbf{X}(t)\cdot\overline{\mathbf{P_1}(t)}\cdot\overline{\mathbf{P_0}(t)} \\ &= \overline{\mathbf{X}(t)}\cdot\overline{\mathbf{P_1}(t)}\cdot\mathbf{P_0}(t) + \mathbf{X}(t)\cdot\overline{\mathbf{P_1}(t)\oplus\mathbf{P_0}(t)}\end{aligned}$$

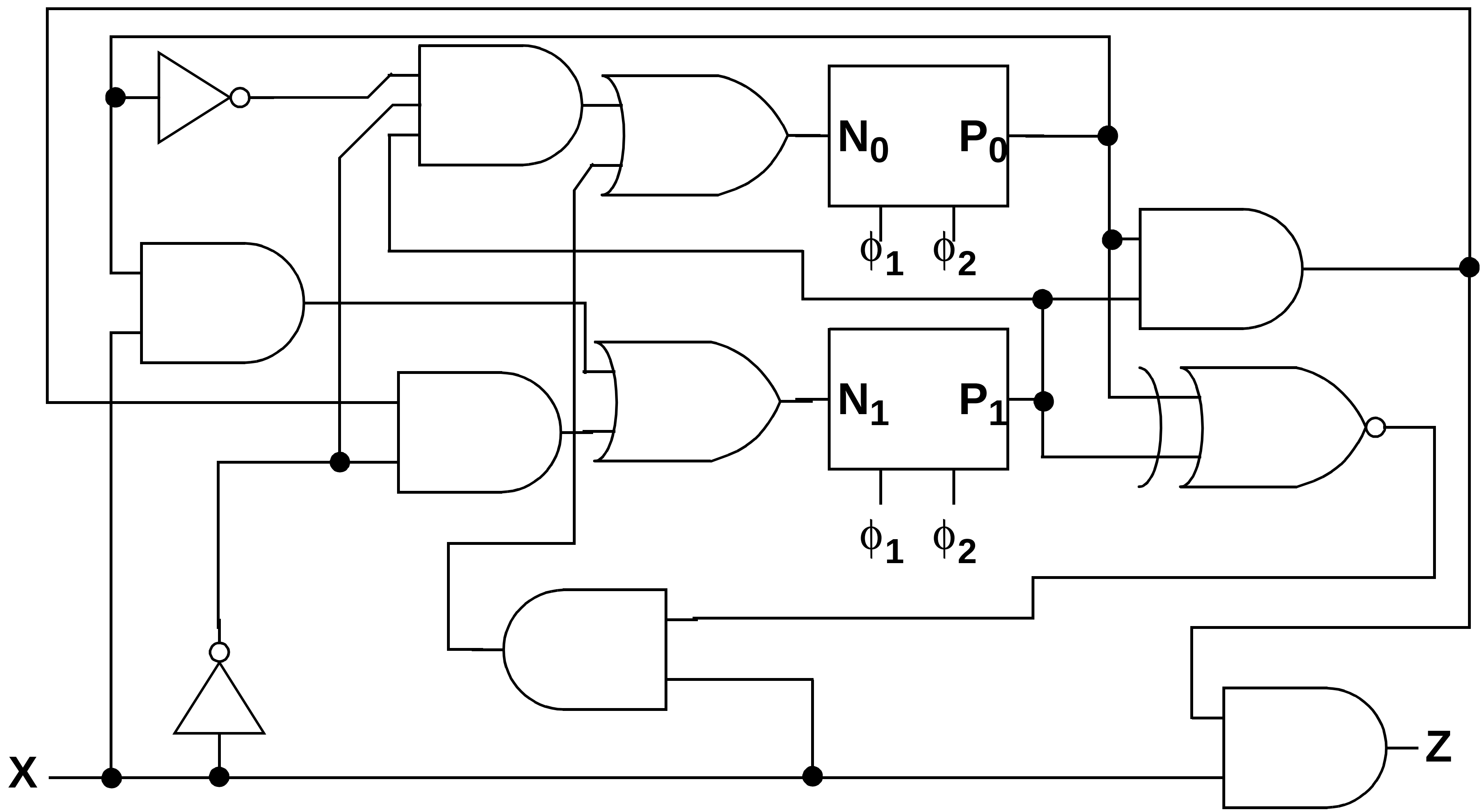
$$\mathbf{Z}(t) = \mathbf{X}\cdot\mathbf{P_1}(t)\cdot\mathbf{P_0}(t)$$

- An important thing to note in these equations is the relation between the present states P and the next states N.

SEQ. CIRCUITS

PATTERN DETECT EXAMPLE

- The following logic circuit implements the pattern detect example.



FSM EXAMPLES

EXAMPLE #1

- Consider the following system description.
  - A sequential system has
    - One input = {**a**, **b**, **c**}
    - One output = {**p**, **q**}
  - Output is
    - **q** when input sequence has even # of **a**'s and odd # of **b**'s
    - **p** otherwise

FSM EXAMPLES

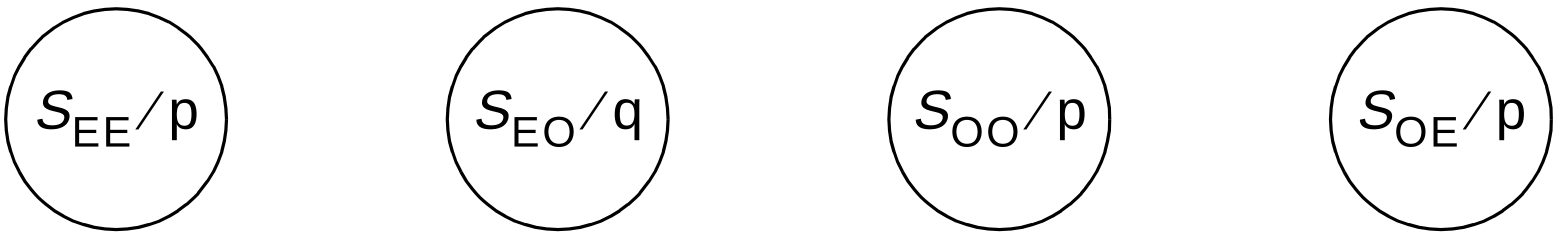
EXAMPLE #1

- We can begin forming a state machine for the system description by reviewing the possible states. In addition, assign each state a state name.
  - S_{EE}: even # of **a**'s and even # of **b**'s / output is **p**
  - S_{EO}: even # of **a**'s and odd # of **b**'s / output is **q**
  - S_{OO}: odd # of **a**'s and odd # of **b**'s / output is **p**
  - S_{OE}: odd # of **a**'s and even # of **b**'s / output is **p**
- Note that this machine can be a Moore machine. So, we can associate the output with each state.

FSM EXAMPLES

EXAMPLE #1

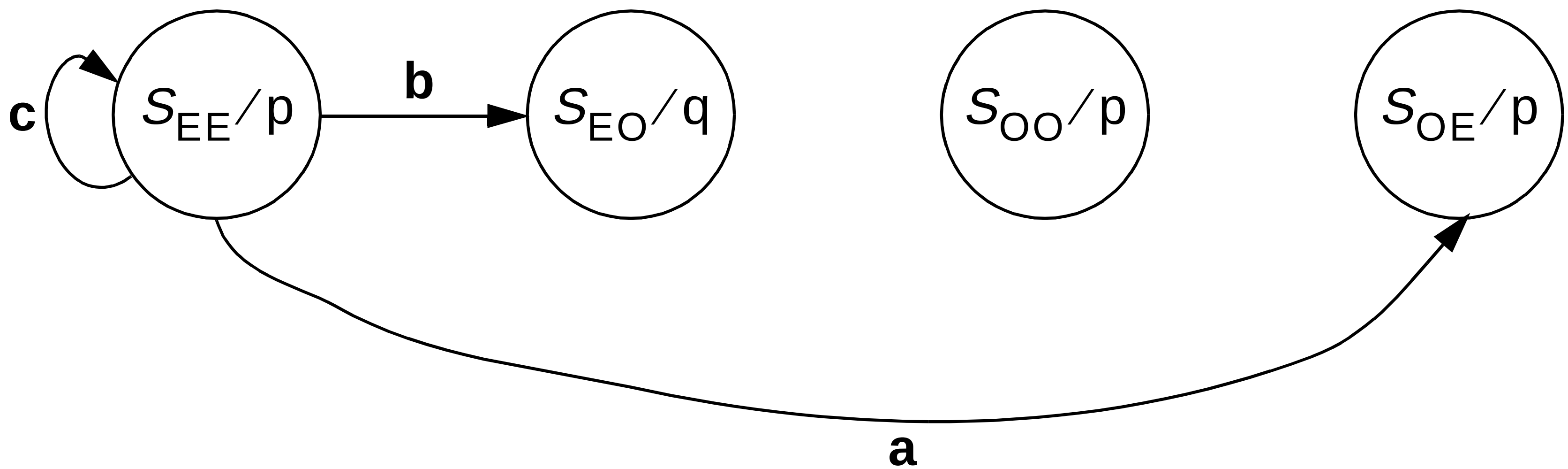
- Now draw a circle with each state.



FSM EXAMPLES

EXAMPLE #1

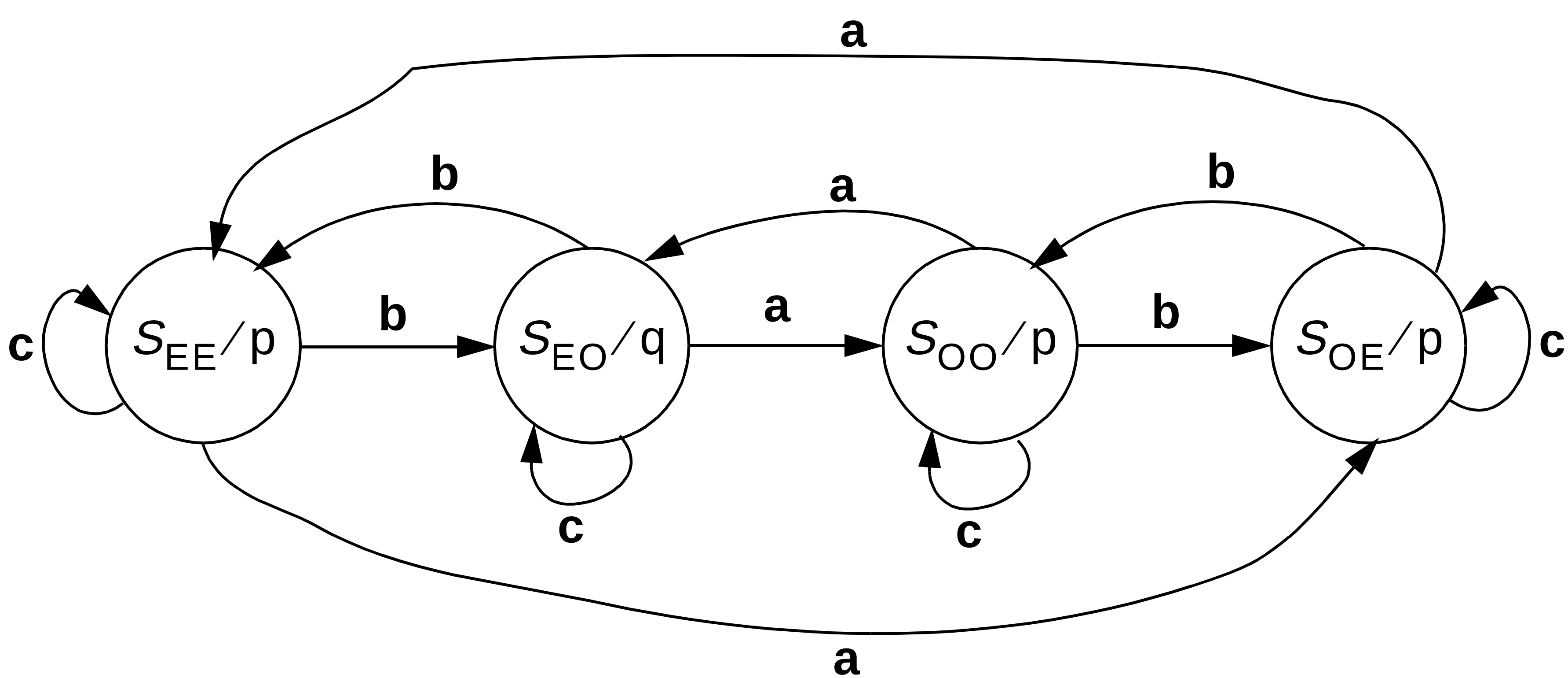
- Finally, for each state, consider the effect for each possible input.
  - For instance, starting with state S_{EE}, the next state for the three input **a**, **b**, and **c** are determined as follows.



FSM EXAMPLES

EXAMPLE #1

- Finishing the state diagram, the following is obtained.



FSM EXAMPLES

EXAMPLE #1

- A state table can also be formed for this state diagram as follows.
  - First, assign a binary number to each state
    - $S_{EE} = \mathbf{00}$, $S_{EO} = \mathbf{01}$, $S_{OO} = \mathbf{10}$, $S_{OE} = \mathbf{11}$
  - Assign a binary number to each input
    - **a** = 00, **b** = 01, **c** = 10
  - Assign a binary number to each output
    - **p** = 0, **q** = 1
  - Then for each state, find the next state for each input. In this case there are three possible input values, so, three possible state transitions from each state.
- The state table on the following slide shows the results for this example.

FSM EXAMPLES

EXAMPLE #1

Present State P_1 P_0	Input X	Next State N_1 N_0	Output Z
S_{EE} = **0** **0**	**a** = **00**	S_{OE} = **1** **1**	**p** = **0**
S_{EE} = **0** **0**	**b** = **01**	S_{EO} = **0** **1**	**p** = **0**
S_{EE} = **0** **0**	**c** = **10**	S_{EE} = **0** **0**	**p** = **0**
S_{EO} = **0** **1**	**a** = **00**	S_{OO} = **1** **0**	**q** = **1**
S_{EO} = **0** **1**	**b** = **01**	S_{EE} = **0** **0**	**q** = **1**
S_{EO} = **0** **1**	**c** = **10**	S_{EO} = **0** **1**	**q** = **1**
S_{OO} = **1** **0**	**a** = **00**	S_{EO} = **0** **1**	**p** = **0**
S_{OO} = **1** **0**	**b** = **01**	S_{OE} = **1** **1**	**p** = **0**
S_{OO} = **1** **0**	**c** = **10**	S_{OO} = **1** **0**	**p** = **0**
S_{OE} = **1** **1**	**a** = **00**	S_{EE} = **0** **0**	**p** = **0**
S_{OE} = **1** **1**	**b** = **01**	S_{OO} = **1** **0**	**p** = **0**
S_{OE} = **1** **1**	**c** = **10**	S_{OE} = **1** **1**	**p** = **0**

FSM EXAMPLES

EXAMPLE #1

- The Boolean function for the output can be determined from a Karnaugh map as follows.
  - Note that an input of **11** is not possible since we only have three inputs that we have assigned to **00**, **01**, and **10**. We can therefore use don't cares for this possible input.

X_1X_0 \ P_1P_0	00	01	11	10
00	0	1	0	0
01	0	1	0	0
11	X	X	X	X
10	0	1	0	0

$$Z = \overline{P_1}P_0$$

FSM EXAMPLES

EXAMPLE #1

- The Boolean function for the next state bit can also be determined from Karnaugh maps as follows.

X_1X_0 \ P_1P_0	00	01	11	10
00	1	1	0	0
01	0	0	1	1
11	X	X	X	X
10	0	0	1	1

$$N_1 = \overline{P_1 \oplus X_1 \oplus X_0}$$

X_1X_0 \ P_1P_0	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	X	X	X	X
10	0	1	1	0

$$N_0 = P_0X_1 + \overline{P_0}\,\overline{X_1} = \overline{P_0 \oplus X_1}$$

FSM EXAMPLES

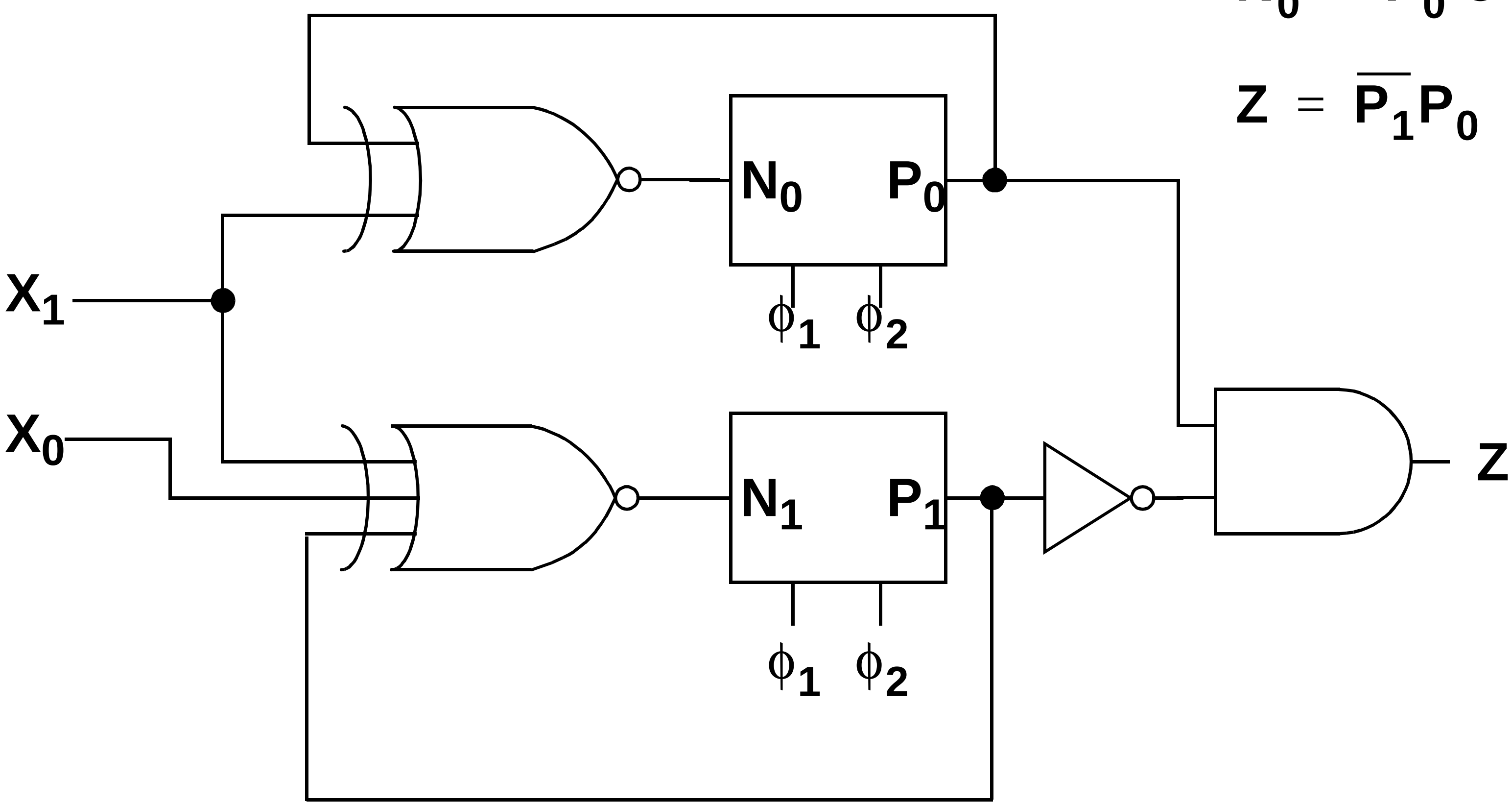
EXAMPLE #1

- The following logic circuit can be made with these Boolean functions.

$$N_1 = \overline{P_1 \oplus X_1 \oplus X_0}$$

$$N_0 = \overline{P_0 \oplus X_1}$$

$$Z = \overline{P_1} P_0$$



FSM EXAMPLES

EXAMPLE #2

- A sequential circuit is defined by the following Boolean functions with input $\mathbf{X}$, present states $\mathbf{P_0}$, $\mathbf{P_1}$, and $\mathbf{P_2}$, and next states $\mathbf{N_0}$, $\mathbf{N_1}$, and $\mathbf{N_2}$.
  - $\mathbf{N_2} = \mathbf{X}(\mathbf{P_1} \oplus \mathbf{P_0}) + \overline{\mathbf{X}}\overline{(\mathbf{P_1} \oplus \mathbf{P_0})}$
  - $\mathbf{N_1} = \mathbf{P_2}$
  - $\mathbf{N_0} = \mathbf{P_1}$
  - $\mathbf{Z} = \mathbf{X}\mathbf{P_1}\mathbf{P_2}$
- Derive the state table.
- Derive the state diagram.

FSM EXAMPLES

EXAMPLE #2

- The state table is formed as follows.

Present State P_2	P_1	P_0	Input X	Next State N_2	N_1	N_0	Output Z
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	1	0	0	0
0	1	0	0	0	0	1	0
0	1	0	1	1	0	1	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	1	0	0
1	0	0	1	0	1	0	0
1	0	1	0	0	1	0	0
1	0	1	1	1	1	0	0
1	1	0	0	0	1	1	0
1	1	0	1	1	1	1	1
1	1	1	0	1	1	1	0
1	1	1	1	0	1	1	1

FSM EXAMPLES

EXAMPLE #2

- The state diagram can be drawn as follows.