Finite State Machines (III)

- Problem statement
- Circuit Synthesis
- Examples of Mealy and Moore models

Problem Statement

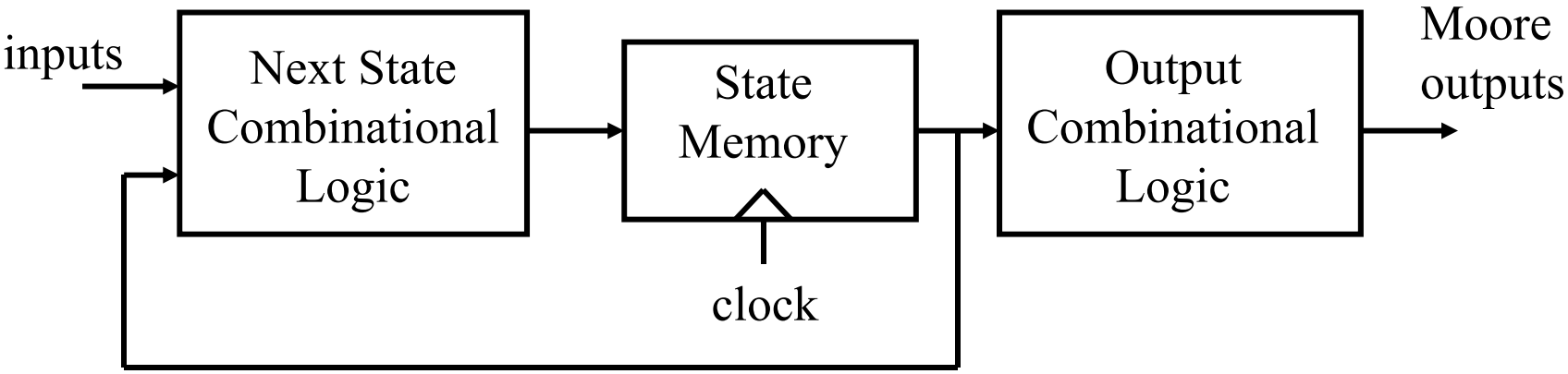
The synthesis of finite state machines usually begins with a natural language description of the problem.

- "Design a circuit to detect the sequence ……."
- "Design a counter which counts up/down …."
- "Design the controller for the cash mechanism of a vending machine ……"
- "Design a traffic light controller ….."

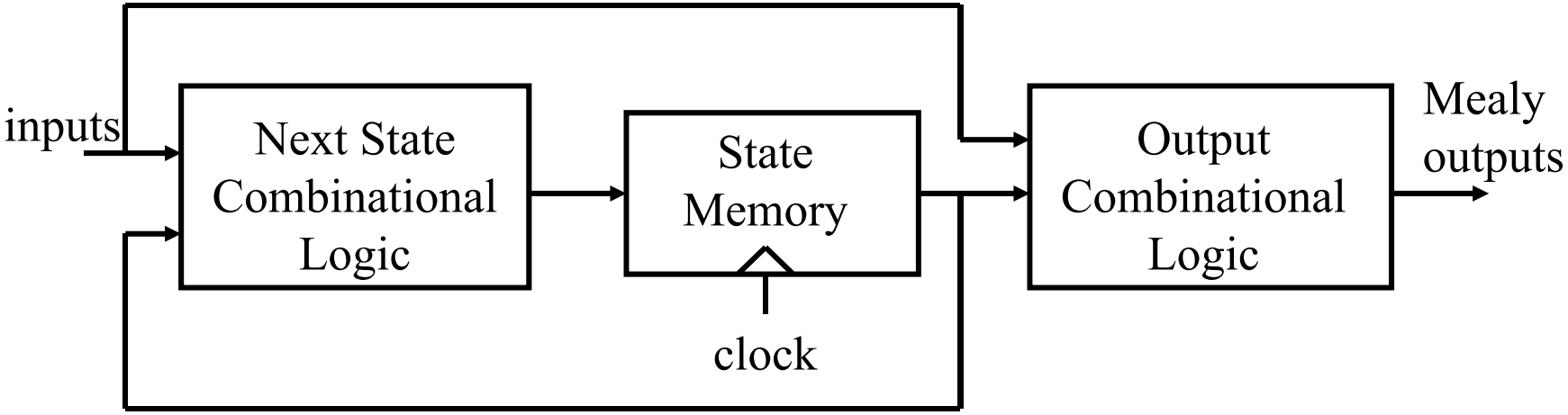
Synthesis Steps

- Obtain an abstract representation of the FSM
- State minimisation
- State Encoding
- State Tables
- Excitation equations
- Logic schematic

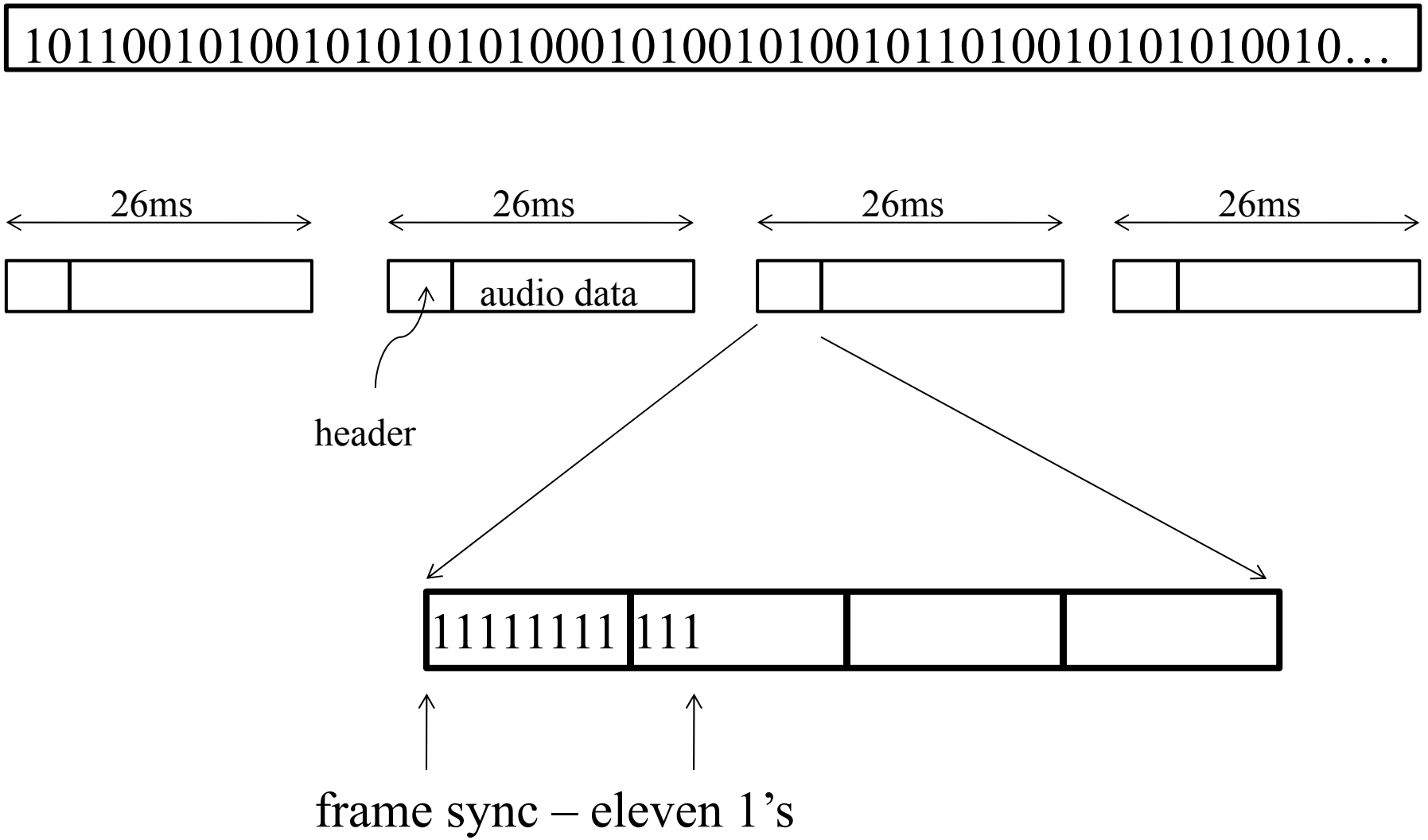
Moore Type FSM



Mealy Type FSM



MP3 Frame Sync



Sequence Detector

"Design a circuit to detect the sequence 1001 on its input line I"

If the circuit outputs a '1' when the sequence is detected, when 1001 has been received, should the circuit start looking for a new sequence from the next I input, or should it use previous values ?

I = 0 0 1 0 0 1 0 0 1 1 0 1 1 ……..
Z = 0 0 0 0 0 1 0 0 0 0 0 0 0 ……..

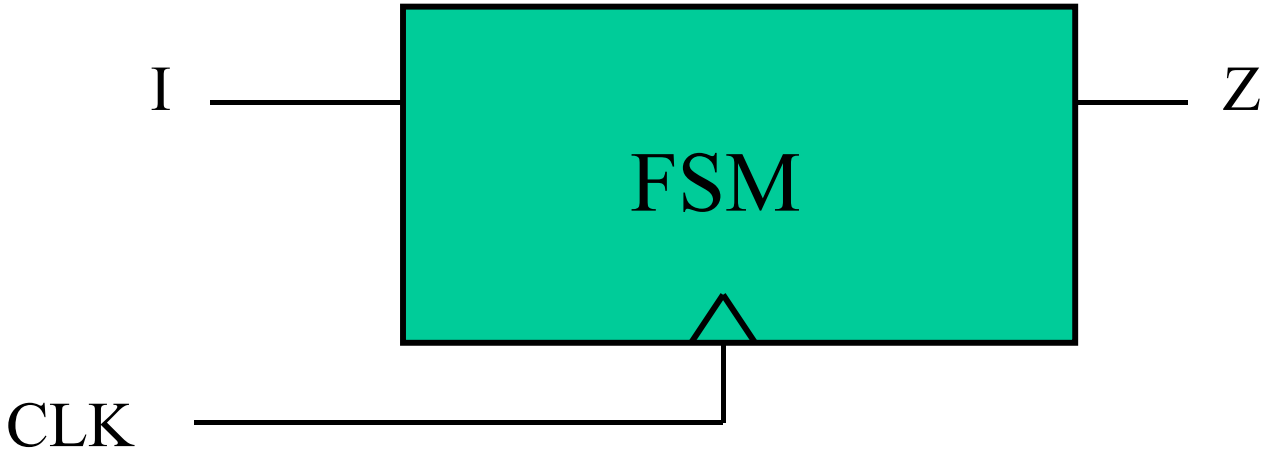
time ⟶

I = 0 0 1 0 0 1 0 0 1 1 0 1 1 ……..
Z = 0 0 0 0 0 1 0 0 1 0 0 0 0 ……..

The first case shows resetting sequential behaviour, and the second case shows non-resetting sequential behaviour.

Sequence Detector Design

The solution may use a Moore or a Mealy machine. There may be advantages of one over the other. Whichever model is chosen it must be carefully synchronised with the rest of its system.



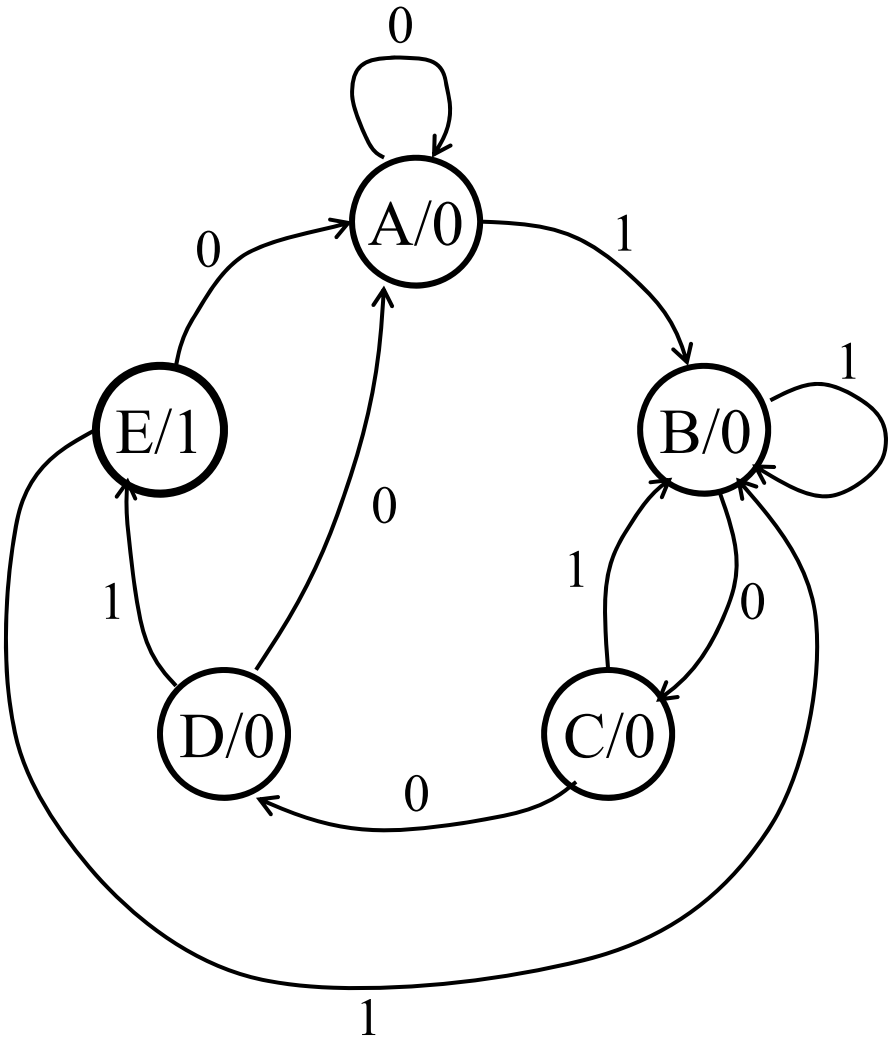
When the machine has been designed it must be simulated to ensure the correct operation.

Test sequence : 00001111010001001001100101010111

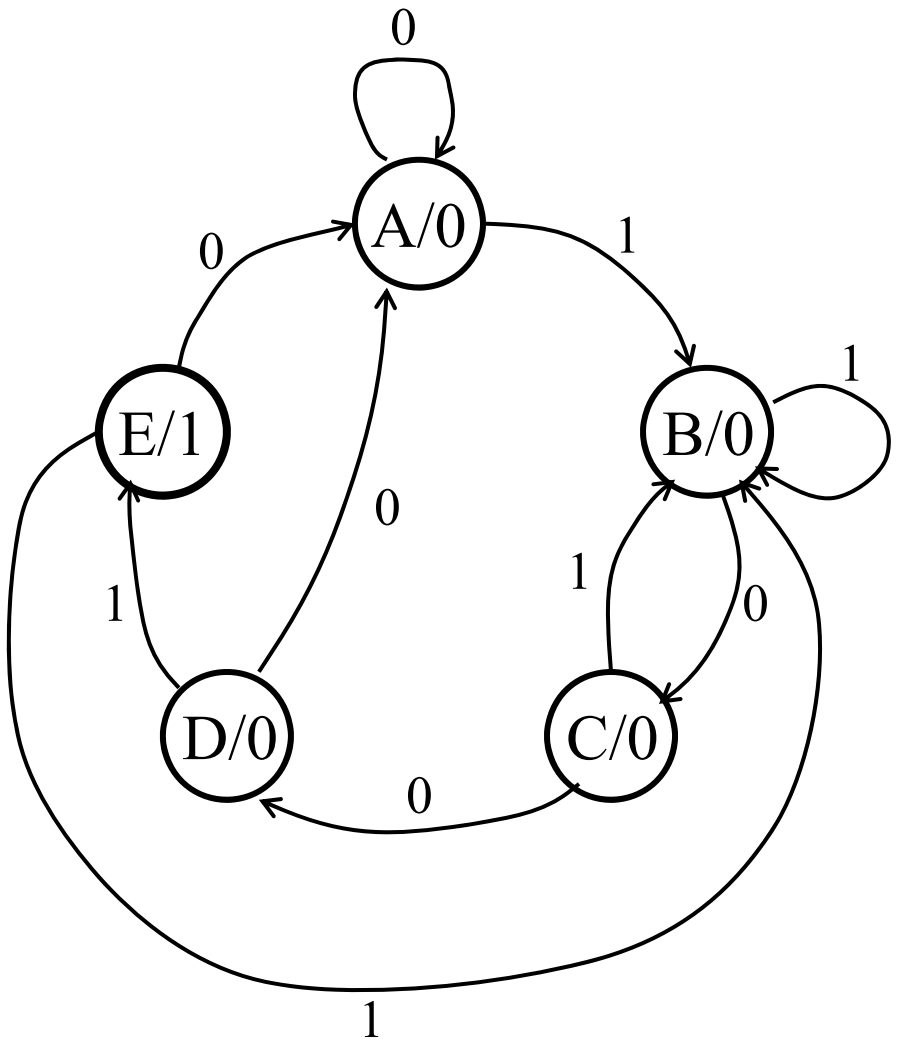
Resetting Moore Model - requires five states to detect 1001

A : Reset state, waiting for first 1
B : 1 received , waiting for 0
C : 10 received , waiting for 0
D : 100 received, waiting for 1
E : 1001 received, output 1

Resetting Moore Model - requires five states to detect 1001

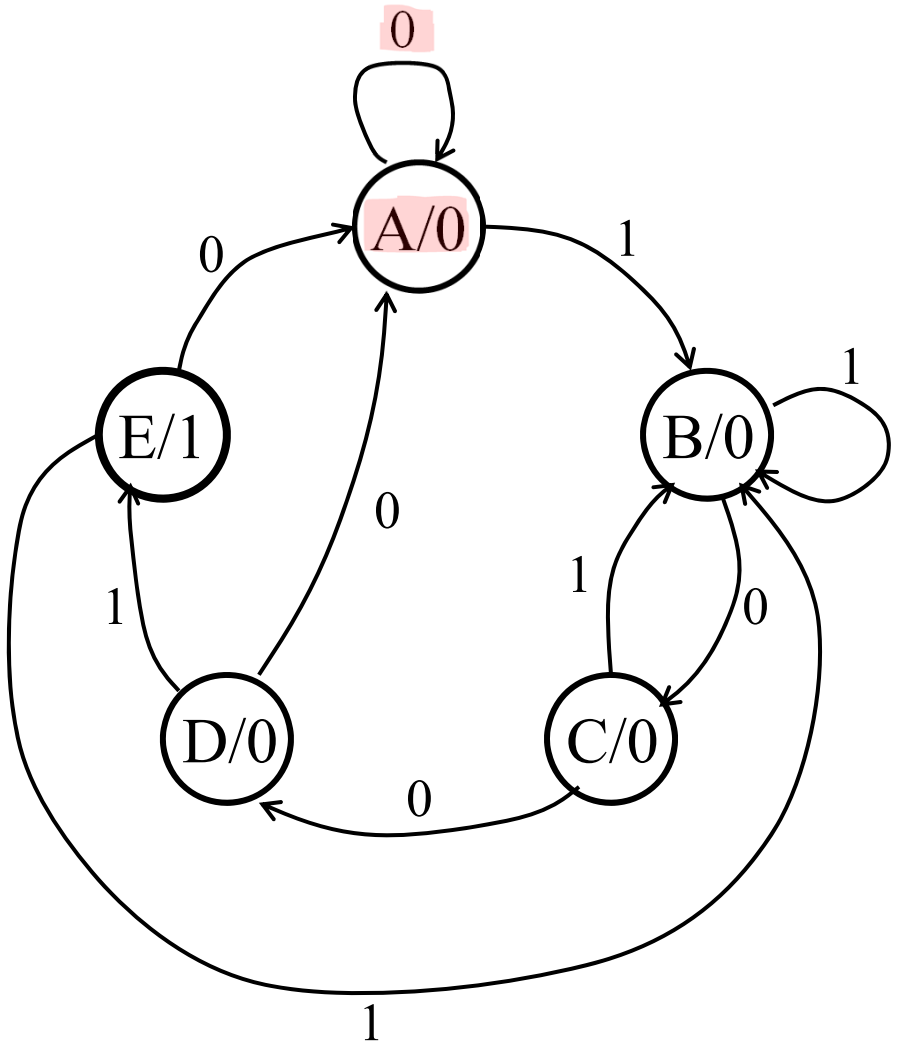


Assign a coding for each state.



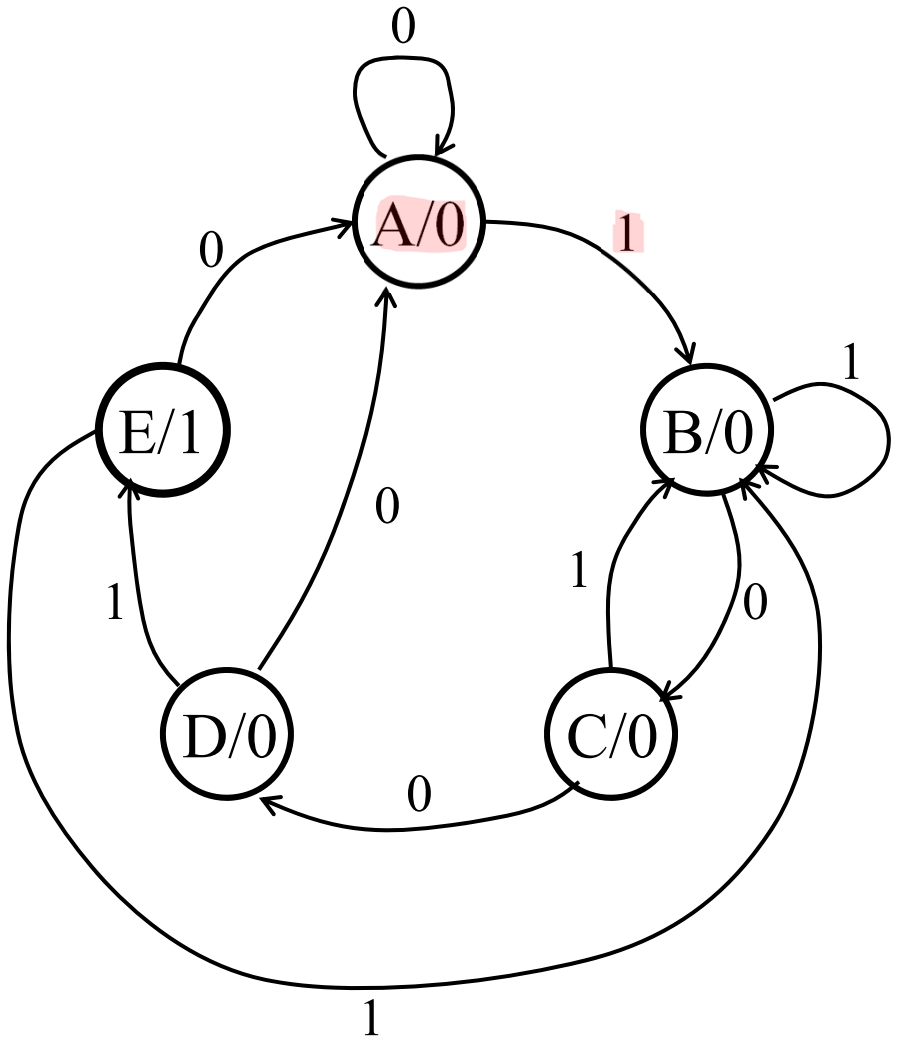
	Present State			I	Next State			
	Q_2	Q_1	Q_0		Q_2	Q_1	Q_0	Z
A	0	0	0	0				
				1				
B	0	0	1	0				
				1				
C	0	1	0	0				
				1				
D	0	1	1	0				
				1				
E	1	0	0	0				
				1				
	1	0	1					
	1	1	0		unused			
	1	1	1					

Resetting Moore Model - requires five states to detect 1001



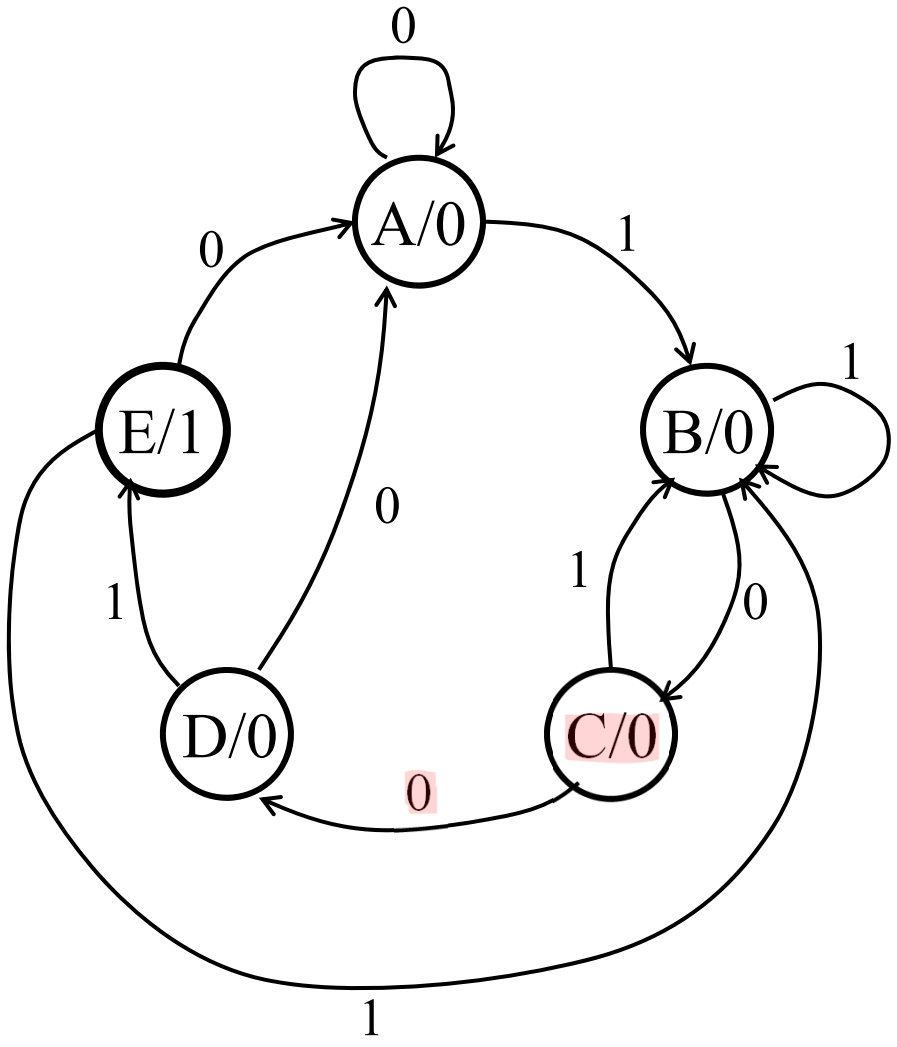
	Present State			I	Next State			
	Q2	Q1	Q0		Q2	Q1	Q0	Z
A	0	0	0	0	0	0	0	0
				1	0	0	1	0
B	0	0	1	0	0	1	0	0
				1	0	0	1	0
C	0	1	0	0	0	1	1	0
				1	0	0	1	0
D	0	1	1	0	0	0	0	0
				1	1	0	0	0
E	1	0	0	0	0	0	0	1
				1	0	0	1	1
	1	0	1					
	1	1	0		unused			
	1	1	1					

Resetting Moore Model - requires five states to detect 1001



	Present State			I	Next State			
	Q2	Q1	Q0		Q2	Q1	Q0	Z
A	0	0	0	0	0	0	0	0
				1	0	0	1	0
B	0	0	1	0	0	1	0	0
				1	0	0	1	0
C	0	1	0	0	0	1	1	0
				1	0	0	1	0
D	0	1	1	0	0	0	0	0
				1	1	0	0	0
E	1	0	0	0	0	0	0	1
				1	0	0	1	1
	1	0	1					
	1	1	0		unused			
	1	1	1					

Resetting Moore Model - requires five states to detect 1001



	Present State			I	Next State			
	Q2	Q1	Q0		Q2	Q1	Q0	Z
A	0	0	0	0	0	0	0	0
				1	0	0	1	0
B	0	0	1	0	0	1	0	0
				1	0	0	1	0
C	0	1	0	0	0	1	1	0
				1	0	0	1	0
D	0	1	1	0	0	0	0	0
				1	1	0	0	0
E	1	0	0	0	0	0	0	1
				1	0	0	1	1
	1	0	1					
	1	1	0		unused			
	1	1	1					

Resetting Moore Model - excitation equations

	Present State				Next State			
	Q_2	Q_1	Q_0	I	Q_2	Q_1	Q_0	Z
A	0	0	0	0	0	0	0	0
				1	0	0	1	0
B	0	0	1	0	0	1	0	0
				1	0	0	1	0
C	0	1	0	0	0	1	1	0
				1	0	0	1	0
D	0	1	1	0	0	0	0	0
				1	1	0	0	0
E	1	0	0	0	0	0	0	1
				1	0	0	1	1

$$Z = Q_2\overline{Q}_1\overline{Q}_0$$

$$Q_{2\,(t+1)} = \overline{Q}_2 Q_1 Q_0\, I$$

$$Q_{1\,(t+1)} = \overline{Q}_2\overline{Q}_1 Q_0\,\overline{I} + \overline{Q}_2 Q_1\overline{Q}_0\,\overline{I}$$

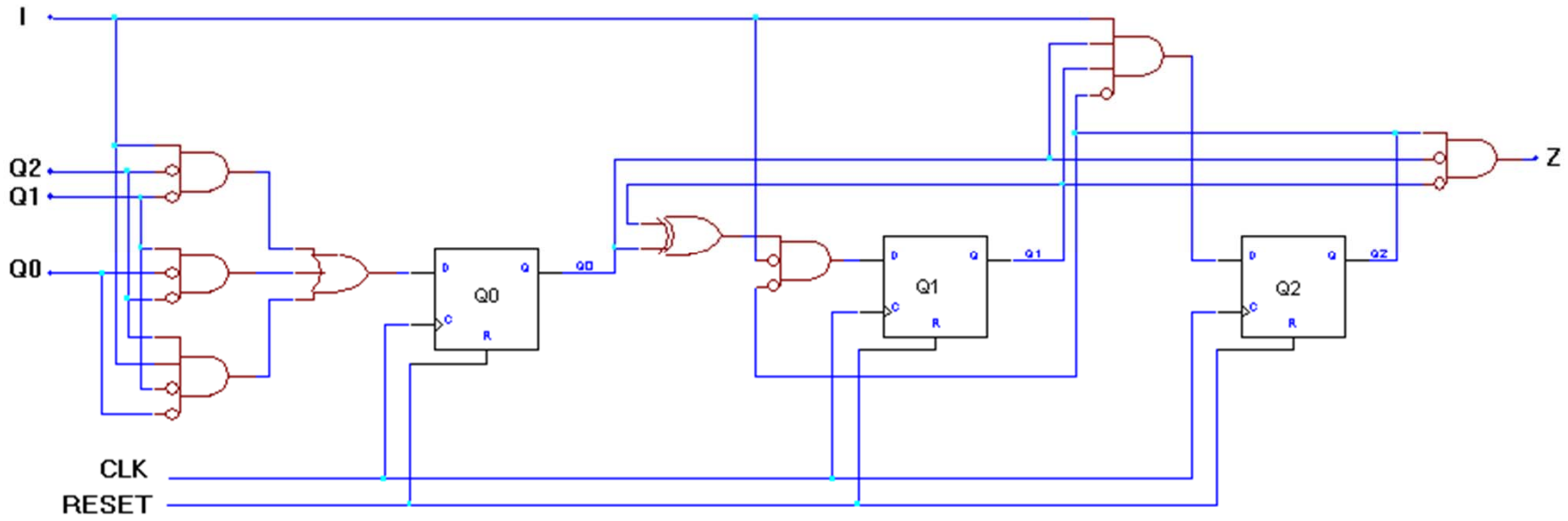
$$= \overline{Q}_2\,\overline{I}\,(\,\overline{Q}_1 Q_0 + Q_1\overline{Q}_0\,)$$

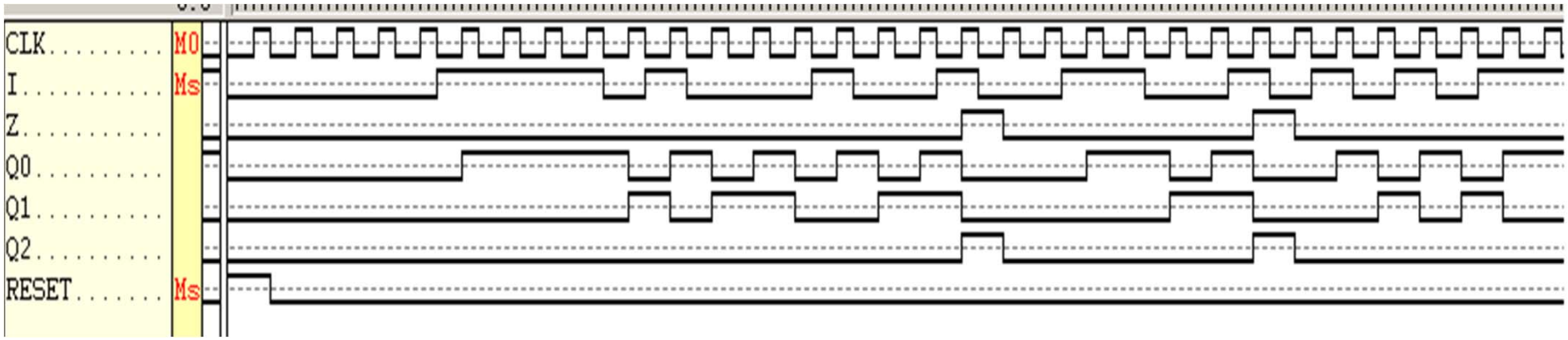
$$= \overline{Q}_2\,\overline{I}\,(\,Q_0 \oplus Q_1\,)$$

$$Q_{0\,(t+1)} = \overline{Q}_2\overline{Q}_1\overline{Q}_0\, I + \overline{Q}_2\overline{Q}_1 Q_0\, I + \overline{Q}_2 Q_1\overline{Q}_0 + Q_2\overline{Q}_1\overline{Q}_0\, I$$

$$= \overline{Q}_2\overline{Q}_1\, I + \overline{Q}_2 Q_1\overline{Q}_0 + Q_2\overline{Q}_1\overline{Q}_0\, I$$

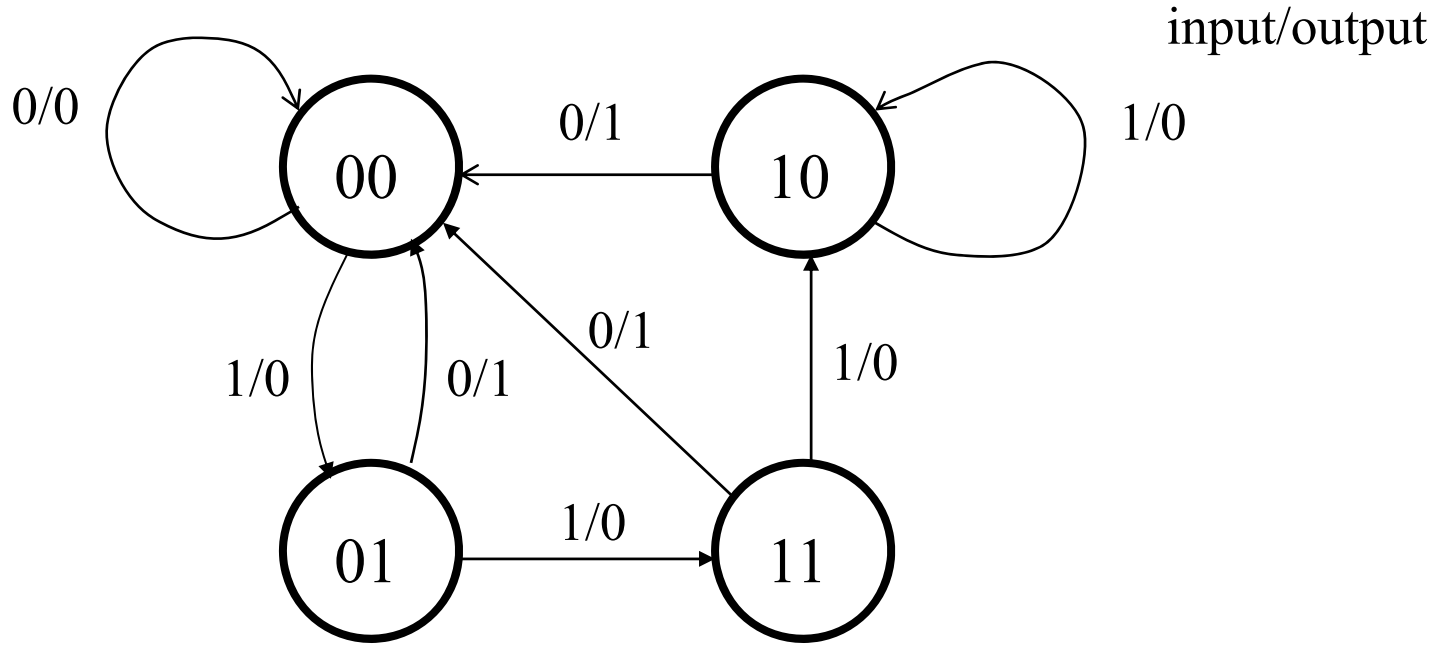
Resetting Moore Model - circuit and simulation



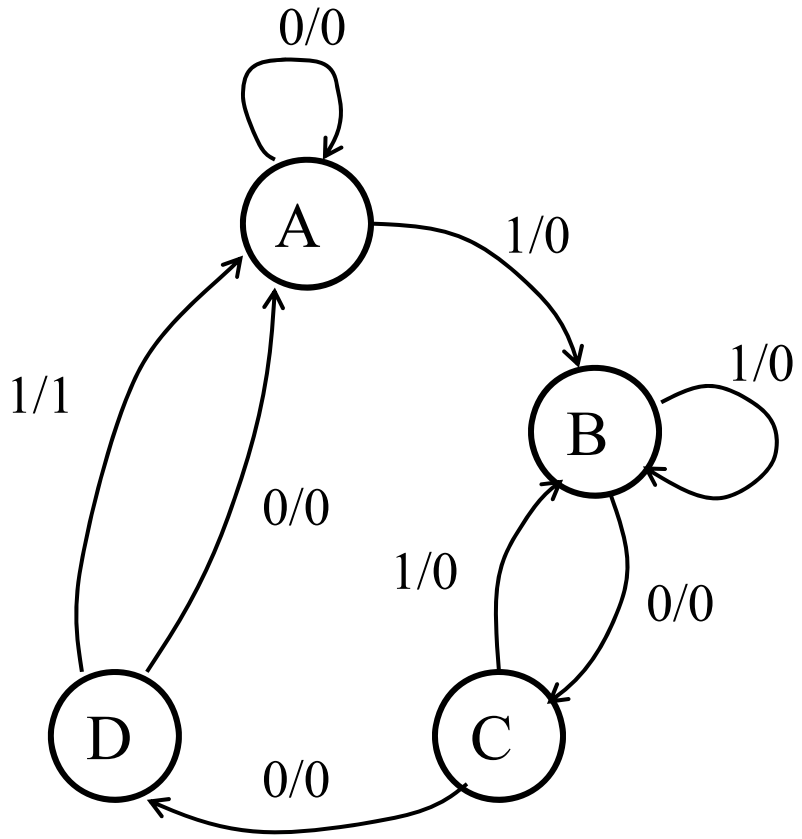


Mealy model

The state diagram for a Mealy machine shows both the input and output values separated by a slash along the directed lines between states.



Resetting Mealy Model - requires four states to detect 1001



	Present State		I	Next State		
	Q_1	Q_0		Q_1	Q_0	z
A	0	0	0	0	0	0
			1	0	1	0
B	0	1	0	1	0	0
			1	0	1	0
C	1	0	0	1	1	0
			1	0	1	0
D	1	1	0	0	0	0
			1	0	0	1

Resetting Mealy Model - excitation equations

	Present State			Next State		
	Q_1	Q_0	I	Q_1	Q_0	z
A	0	0	0	0	0	0
			1	0	1	0
B	0	1	0	1	0	0
			1	0	1	0
C	1	0	0	1	1	0
			1	0	1	0
D	1	1	0	0	0	0
			1	0	0	1

$$Z = Q_1 Q_0 I$$

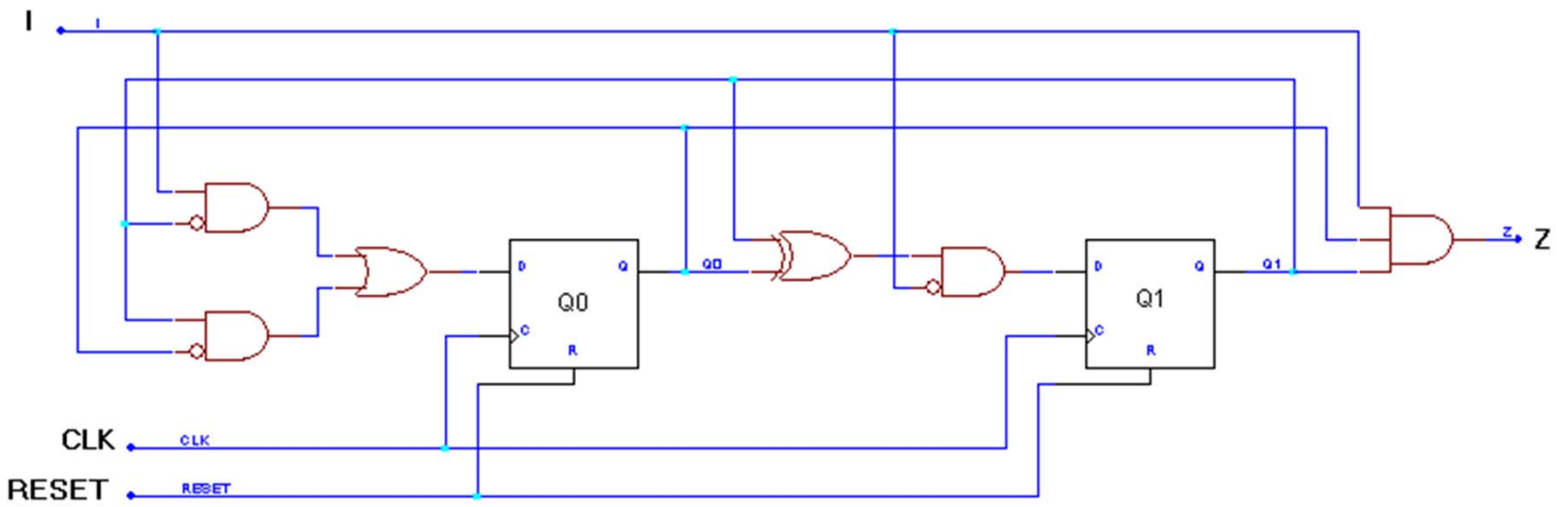
$$Q_{1\,(t+1)} = \overline{Q}_1 Q_0 \overline{I} + Q_1 \overline{Q}_0 \overline{I}$$

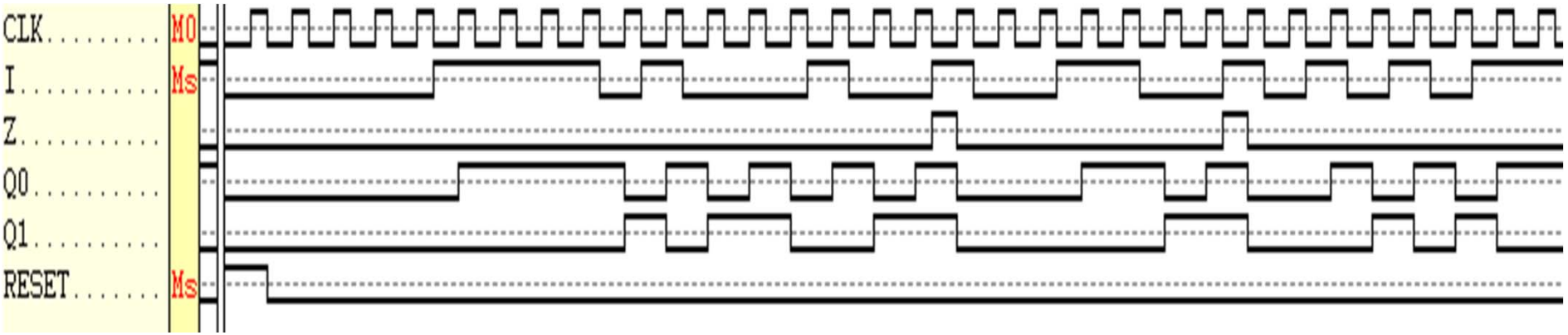
$$= \overline{I} (Q_1 \oplus Q_0)$$

$$Q_{0\,(t+1)} = \overline{Q}_1 \overline{Q}_0 I + \overline{Q}_1 Q_0 I + Q_1 \overline{Q}_0$$

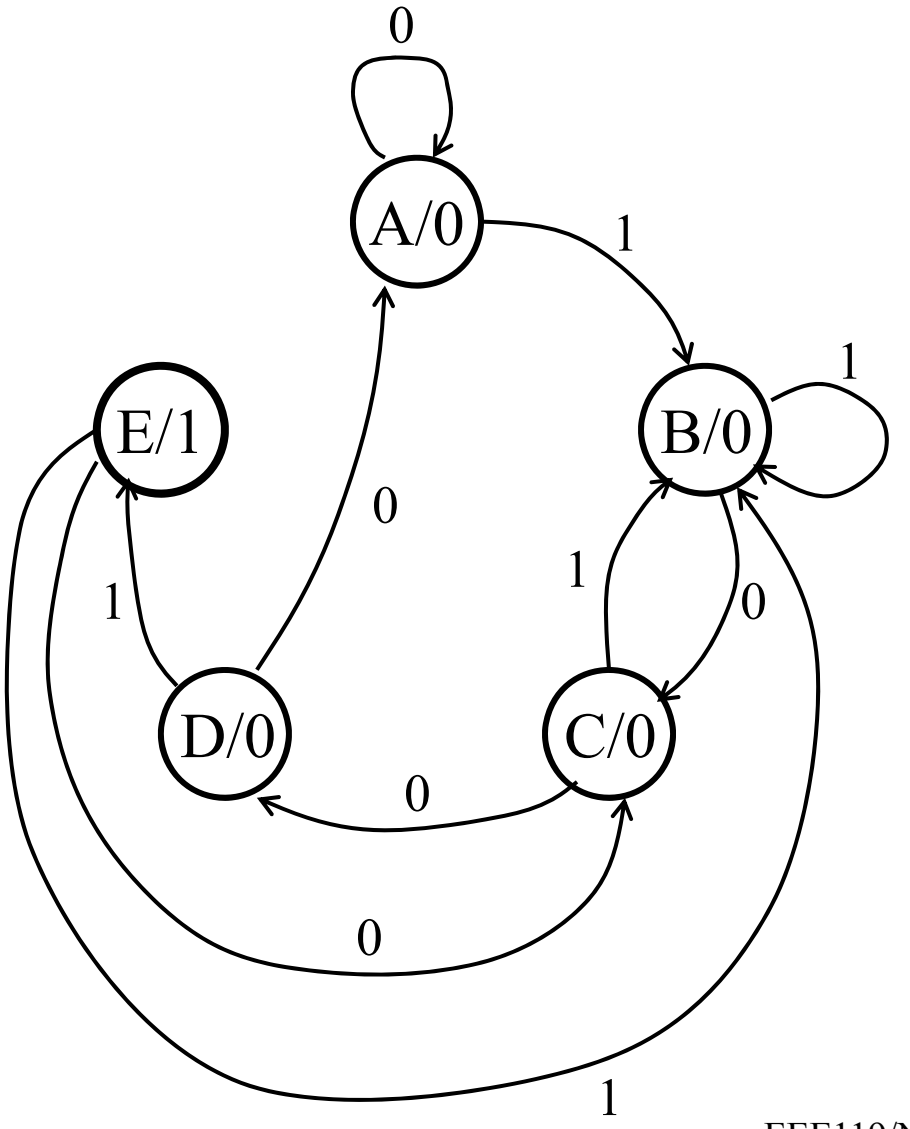
$$= \overline{Q}_1 I + Q_1 \overline{Q}_0$$

Resetting Mealy Model - circuit and simulation





Non-resetting Moore Model - requires five states to detect 1001



	Present State			I	Next State			
	Q_2	Q_1	Q_0		Q_2	Q_1	Q_0	Z
A	0	0	0	0	0	0	0	0
				1	0	0	1	0
B	0	0	1	0	0	1	0	0
				1	0	0	1	0
C	0	1	0	0	0	1	1	0
				1	0	0	1	0
D	0	1	1	0	0	0	0	0
				1	1	0	0	0
E	1	0	0	0	0	1	0	1
				1	0	0	1	1
	1	0	1					
	1	1	0		unused			
	1	1	1					

Non-resetting Moore Model - excitation equations

	Present State				Next State			
	Q_2	Q_1	Q_0	I	Q_2	Q_1	Q_0	Z
A	0	0	0	0	0	0	0	0
				1	0	0	1	0
B	0	0	1	0	0	1	0	0
				1	0	0	1	0
C	0	1	0	0	0	1	1	0
				1	0	0	1	0
D	0	1	1	0	0	0	0	0
				1	1	0	0	0
E	1	0	0	0	0	1	0	1
				1	0	0	1	1

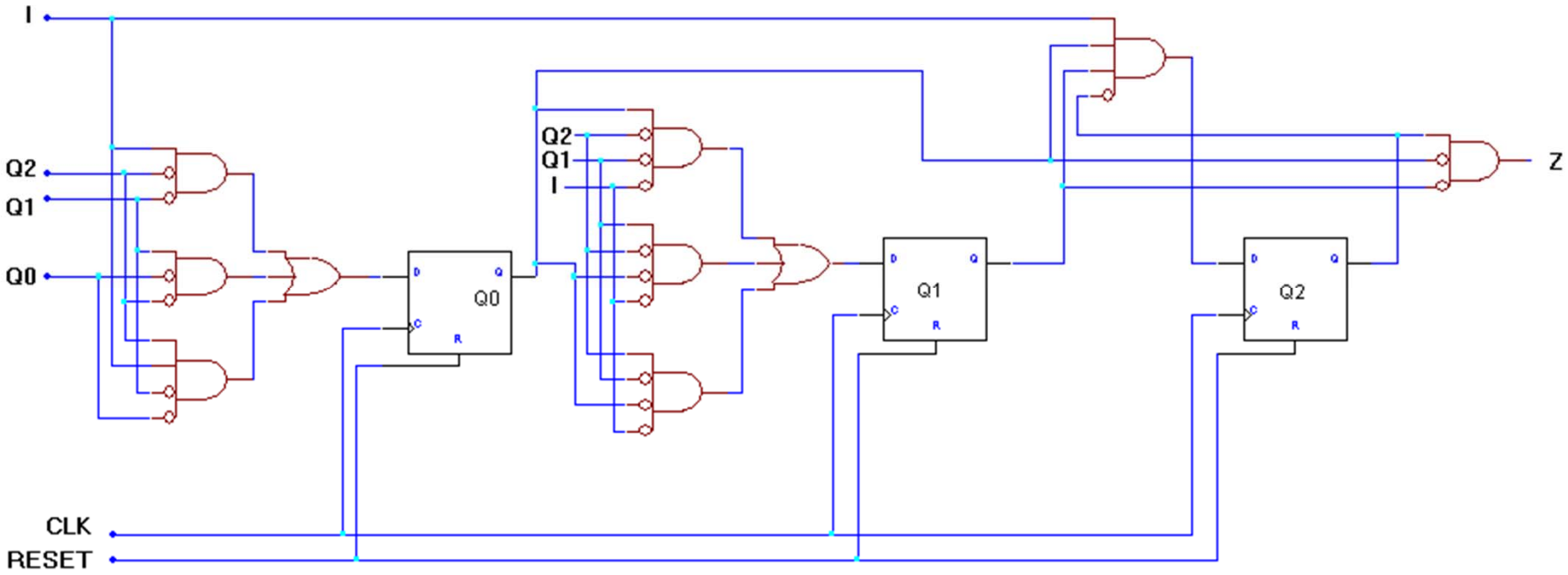
$$Z = Q_2\overline{Q}_1\overline{Q}_0$$

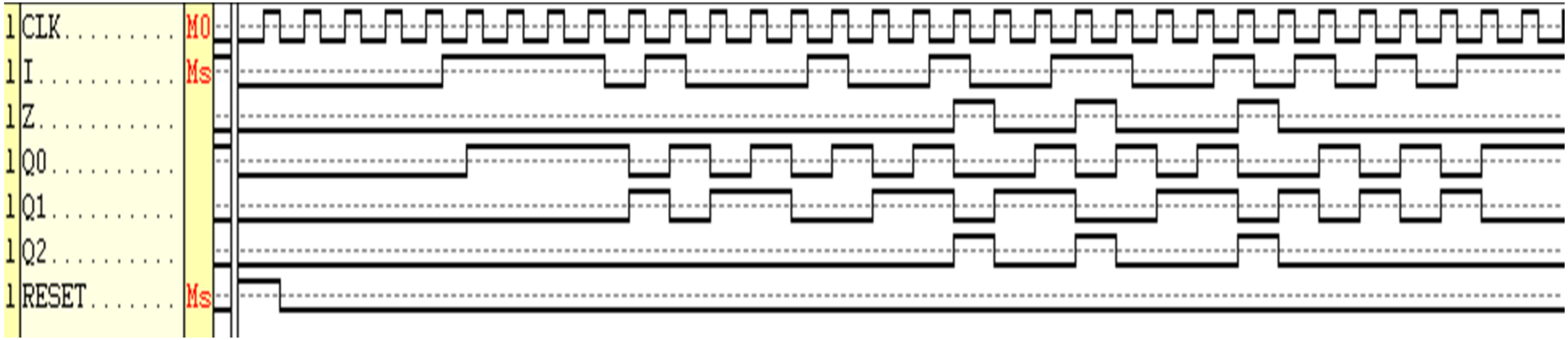
$$Q_{2\,(t+1)} = \overline{Q}_2 Q_1 Q_0\, I$$

$$Q_{1\,(t+1)} = \overline{Q}_2\overline{Q}_1 Q_0\,\overline{I} + \overline{Q}_2 Q_1\overline{Q}_0\,\overline{I} + Q_2\overline{Q}_1\overline{Q}_0\,\overline{I}$$

$$Q_{0\,(t+1)} = \overline{Q}_2\overline{Q}_1\, I + \overline{Q}_2 Q_1\overline{Q}_0 + Q_2\overline{Q}_1\overline{Q}_0\, I$$

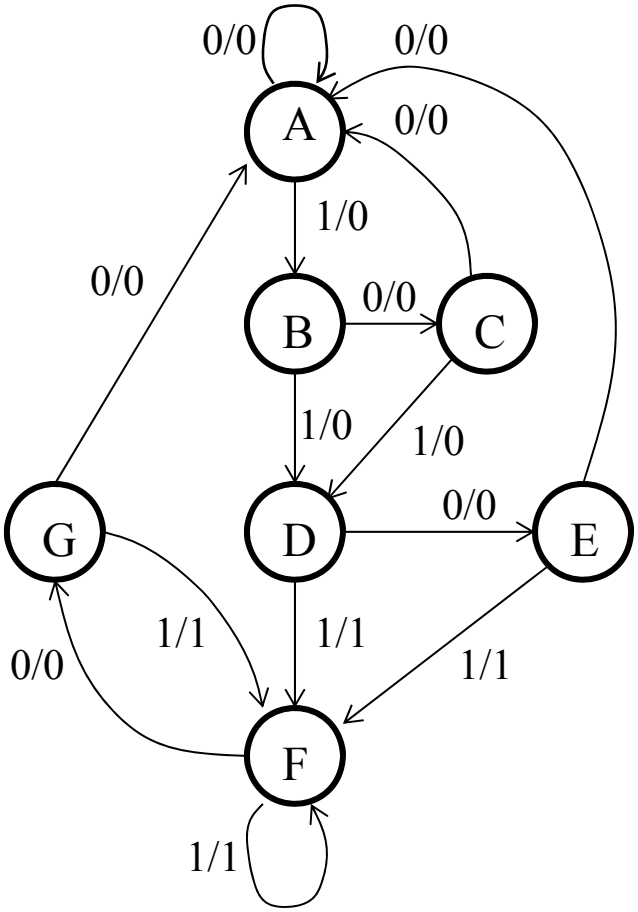
Non-resetting Moore Model - circuit and simulation





State Reduction

It may be possible to reduce the number of states by merging equivalent states. The state diagram below gives the state table shown.



Present State	in I	Next State	out Z
A	0	A	0
	1	B	0
B	0	C	0
	1	D	0
C	0	A	0
	1	D	0
D	0	E	0
	1	F	1
E	0	A	0
	1	F	1
F	0	G	0
	1	F	1
G	0	A	0
	1	F	1

State Reduction

Present State	in I	Next State	out Z
A	0	A	0
	1	B	0
B	0	C	0
	1	D	0
C	0	A	0
	1	D	0
D	0	E	0
	1	F	1
E	0	A	0
	1	F	1
F	0	G	0
	1	F	1
G	0	A	0
	1	F	1

Equivalent states

For two states to be equivalent, each input must send the circuit to the same next state and give the same output. One of them can then be removed.

In the example, states G and E satisfy this rule. Therefore they are equivalent and one of these states can be removed.

In the table, state G has been removed. Where state G occurs in the next state column, it is replaced by state E.

The table now shows that states D and F are equivalent. State F can be removed and replaced by state D.

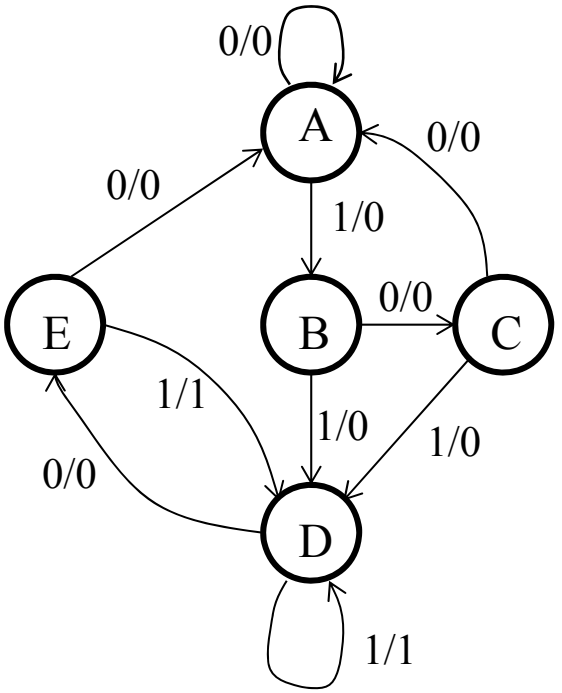
Present State	in I	Next State	out Z
A	0	A	0
	1	B	0
B	0	C	0
	1	D	0
C	0	A	0
	1	D	0
D	0	E	0
	1	F	1
E	0	A	0
	1	F	1
F	0	E	0
	1	F	1

Equivalent states

Present State	in I	Next State	out Z
A	0	A	0
	1	B	0
B	0	C	0
	1	D	0
C	0	A	0
	1	D	0
D	0	E	0
	1	F	1
E	0	A	0
	1	F	1
F	0	E	0
	1	F	1

This results in the reduced state table and state diagram shown.

Present State	in I	Next State	out Z
A	0	A	0
	1	B	0
B	0	C	0
	1	D	0
C	0	A	0
	1	D	0
D	0	E	0
	1	D	1
E	0	A	0
	1	D	1

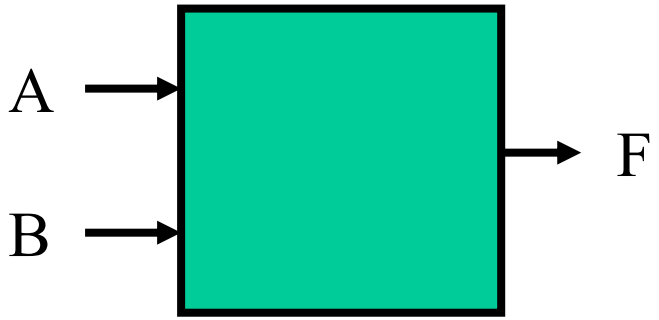


This may (or may not) result in a simpler logic circuit.

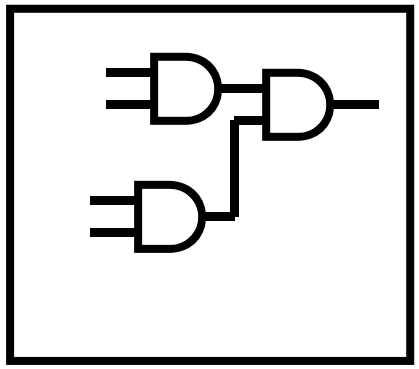
Verilog Module

The basic building block in Verilog is called a module. It is declared with the keyword **module** and always ends with the keyword **endmodule**. A digital system can be described in Verilog by a set of these modules.

Port declarations detail the interface of a module to other modules or the outside world.

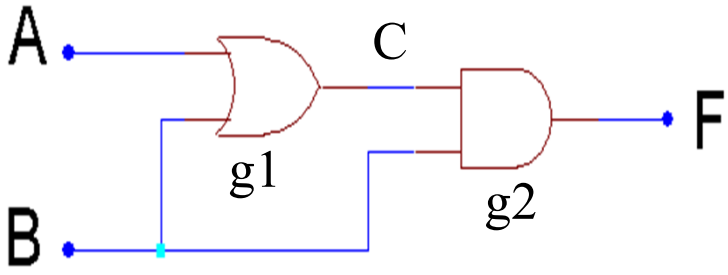


The module body describes the function of the model and the relationship between the ports.



Structural Description

The structure of a circuit can be described in Verilog using predefined gate primitives. These include **not**, **and**, **nand**, **or**, **nor**, **xor**, **xnor**.



Text after // is interpreted as a comment. Keywords are in bold.

```
module example1 (A, B, F);        // module name and port list
input A, B;                       // declare input ports
output F;                         // declare output ports
wire C;                           // declare internal connection
or g1(C, A, B);                   // or gate with optional name g1
and g2(F, C, B);                  // and gate with optional name g2
endmodule                         // gate outputs come first in list
```

Summary

The design sequence for a finite state machine has been detailed.

The solution can be a Moore model or Mealy model.

The FSM must be carefully synchronised to the rest of its system.