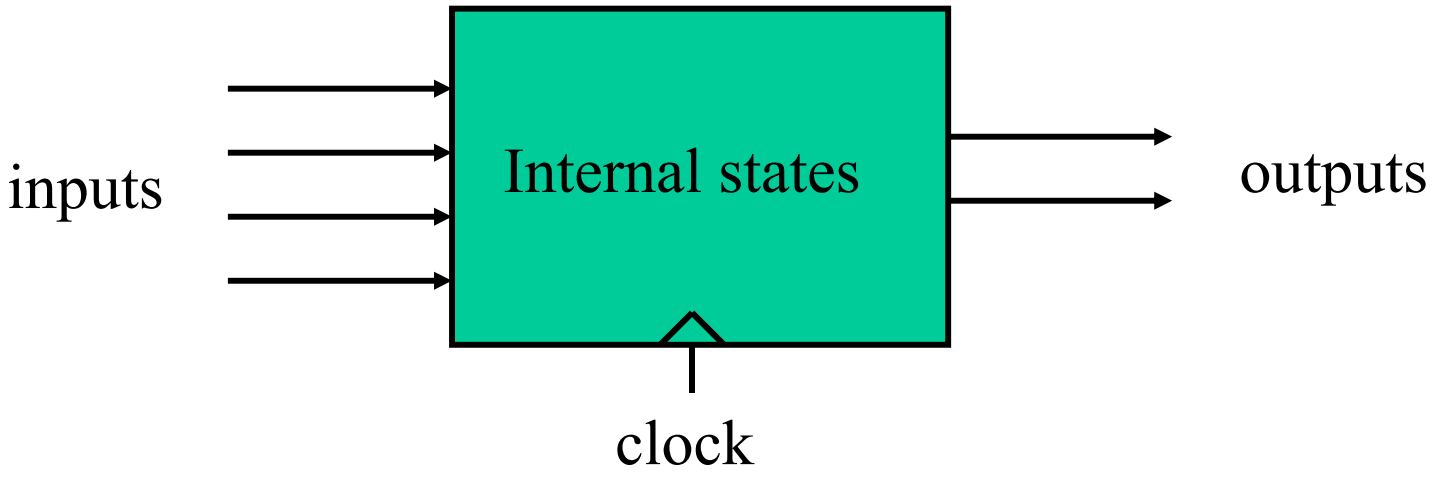
Finite State Machines (I)

- System States
- Moore and Mealy Models
- State Diagrams
- Analysis and Design of FSMs

Introduction

A Finite State Machine (FSM) is a sequential logic circuit that can exist in a finite number of states. In its most general form, it consists of inputs, outputs and internal states.



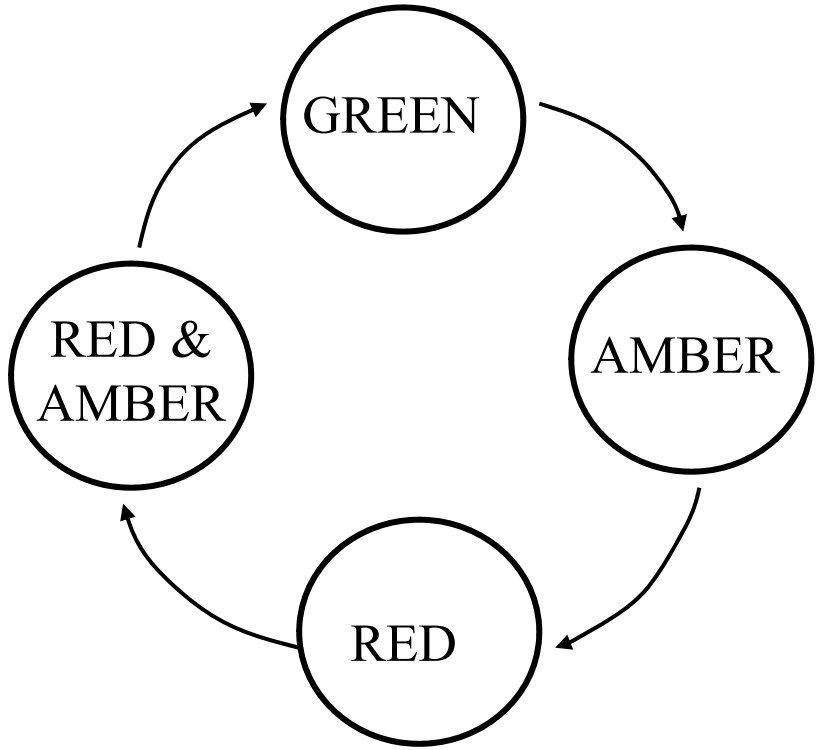
Finite state machines are useful for designs such as sequencers (traffic lights), controllers (vending machine) and sequence detectors (mp3 frame sync).

System States

The state diagram shows a traffic light sequence with four states.

The states are represented by circles and the transitions between the states by directed arrows.

For a synchronous FSM, movement between states will happen when an active edge from the free running clock is applied.



It is possible to have both synchronous and asynchronous sequential systems. For synchronous systems, the design process is simpler and more reliable.

State Encoding

The state is defined by the output of flip-flops which form the memory part of the system. It can be represented in two ways.

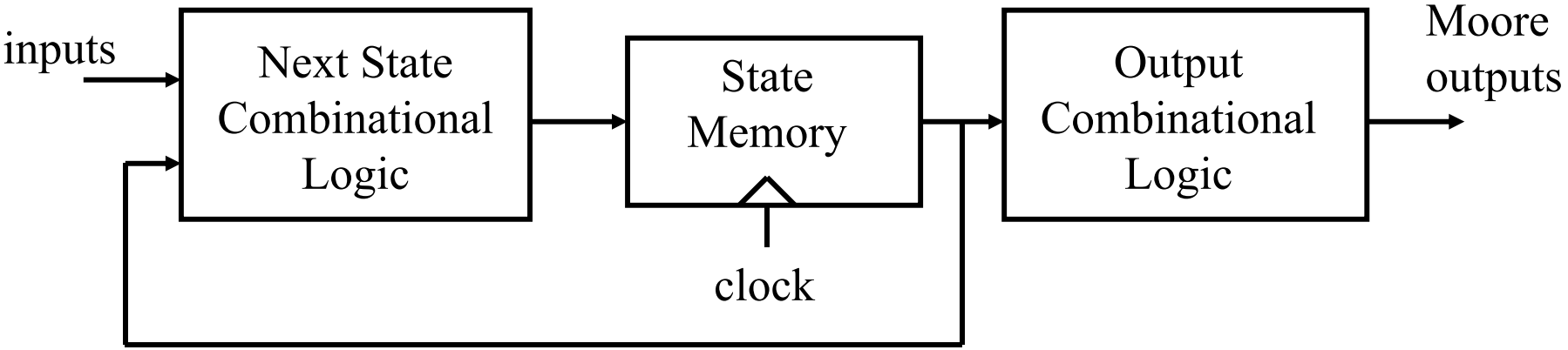
Binary Encoding: As each flip-flop can be in one of two possible states (0,1), then a circuit with n flip-flops has 2^n possible states.

One-Hot Encoding: One flip-flop per state. The active (hot) state is represented by a 1. All other state bits are 0.

	Binary Encoding	Gray Encoding	One Hot Encoding
GREEN	0 0	0 0	0 0 0 1
AMBER	0 1	0 1	0 0 1 0
RED	1 0	1 1	0 1 0 0
RED/AMBER	1 1	1 0	1 0 0 0

Moore Type FSM

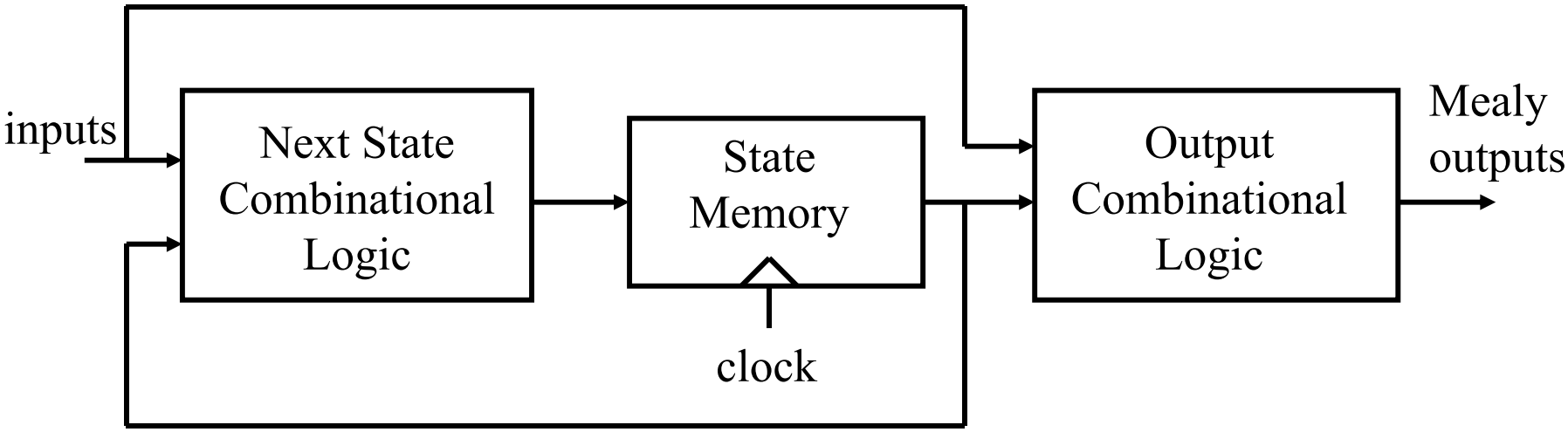
State machines can be divided into two types depending upon how the output is formed. The Moore model describes a general synchronous circuit where the external outputs are only a function of the circuit's present state.



The Moore machine output may be the state value itself, or may be decoded from the state value by the output combinational logic. Because of this, the outputs are synchronised with the clock.

Mealy Type FSM

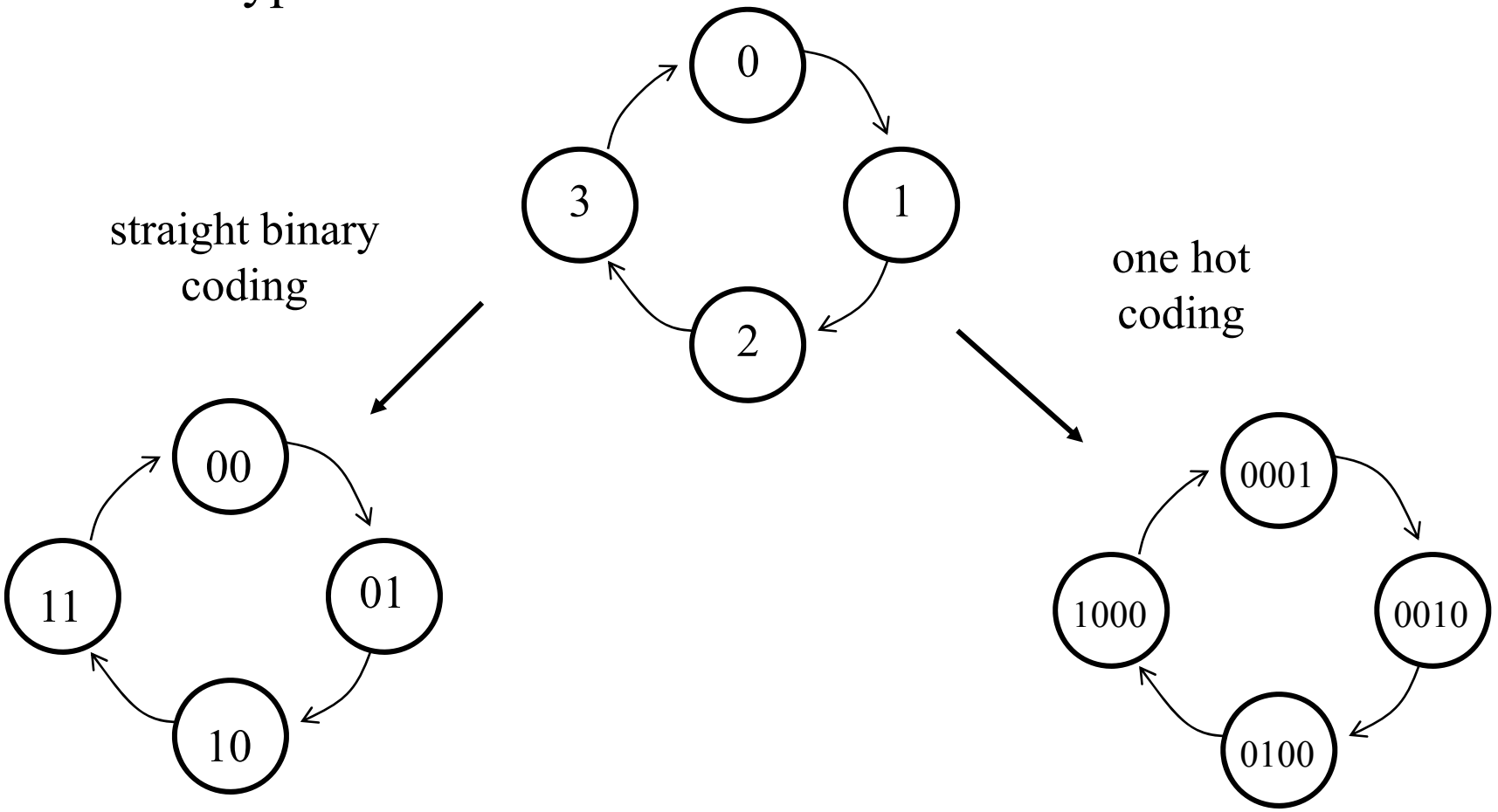
In the Mealy model the external outputs are a function of the present state and also the input.



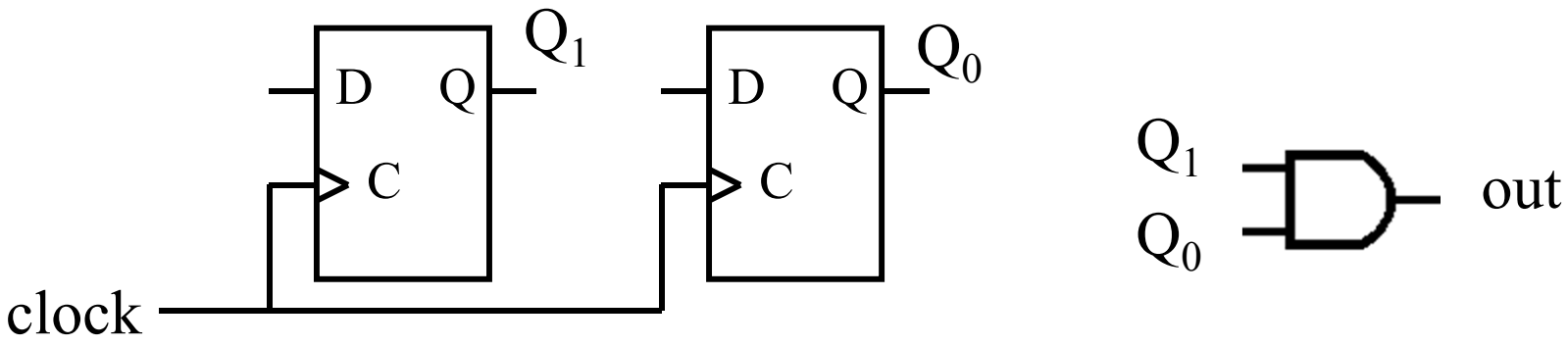
The outputs can change if the inputs change during the clock cycle and may give false values at the output. Thus, input changes must be carefully synchronised. This can be achieved by changing the inputs at the in-active clock edge and sampling the output just before the next active clock edge.

Autonomous Circuits

Autonomous circuits have no external inputs apart from the clock line. They move through a set cycle of states as the circuit is clocked. These are Moore type machines.



For the up-counter shown previously, with straight binary encoding, two flip-flops are required to store the state. In general, the states do not have to follow a binary sequence and the output does not have to be the output of the flip-flops. The output could be decoded from the flip-flops.

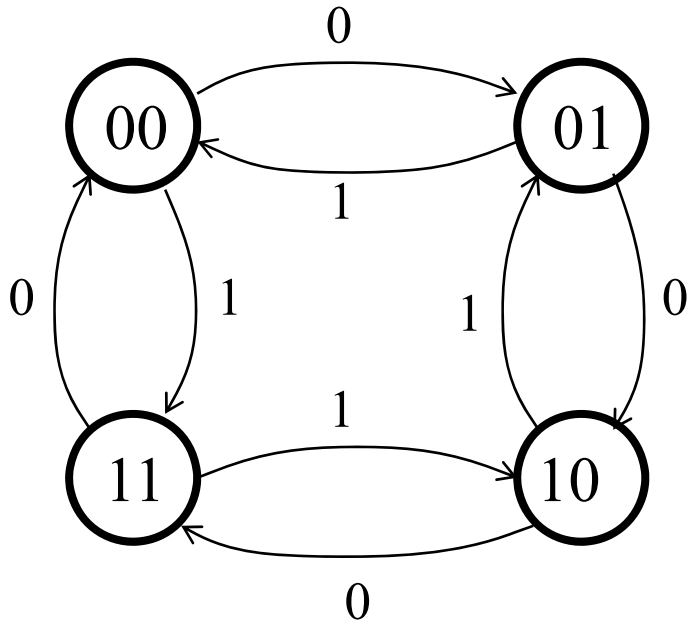


Consider the requirement for a circuit that gives an output pulse every four clock cycles. This could be achieved by AND'ing the state outputs. This will be decoded to give an output pulse for the duration of state 3.

State Diagrams

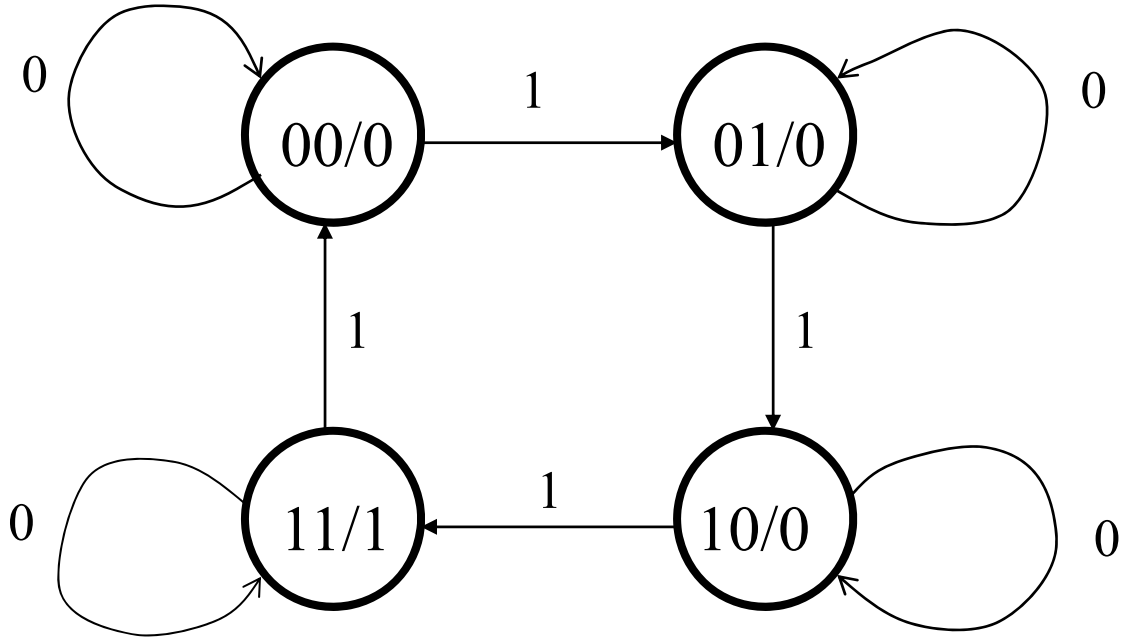
Moore model

Because the outputs are only a function of the present state, the state diagram for a Moore machine only needs to show the circuit inputs along the directed lines between states.

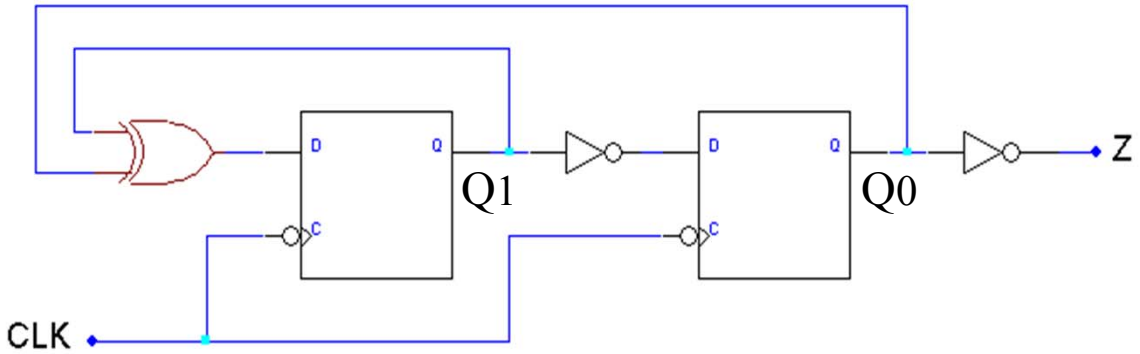


The circuit shown has a single input (I). Multiple inputs could be shown e.g. inputs XY would show 00,01,10,11 on the directed lines.

In the case of a counter, the output is usually taken straight from the flip-flops and the output is completely defined by the state. If the output value is some function of the present state, it is shown inside the circle alongside the present state.



FSM Analysis Example



Next state equations:

$$Q_{1(t+1)} = Q_1 \text{ xor } Q_0$$

$$Q_{0(t+1)} = \overline{Q}_1$$

Output: $Z = \overline{Q}_0$

$Q_{1(t+1)} = Q_1 \text{ xor } Q_0$ Output: $Z = \overline{Q_0}$

$Q_{0(t+1)} = \overline{Q_1}$

Two flip-flops gives four possible states (A,B,C,D).

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

$Q_{1(t+1)} = Q_1 \text{ xor } Q_0$ Output: $Z = \overline{Q_0}$

$Q_{0(t+1)} = \overline{Q_1}$

Two flip-flops gives four possible states (A,B,C,D).

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

$Q_{1(t+1)} = Q_1 \text{ xor } Q_0$ Output: $Z = \overline{Q_0}$

$Q_{0(t+1)} = \overline{Q_1}$

Two flip-flops gives four possible states (A,B,C,D).

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

$Q_{1(t+1)} = Q_1 \text{ xor } Q_0$ Output: $Z = \overline{Q_0}$

$Q_{0(t+1)} = \overline{Q_1}$

Two flip-flops gives four possible states (A,B,C,D).

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

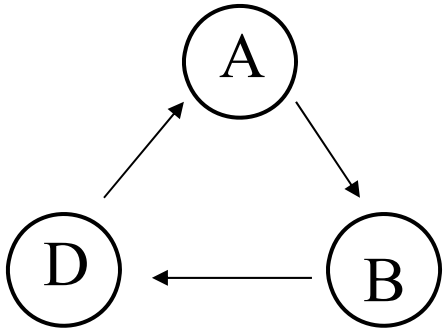
$Q_{1(t+1)} = Q_1 \text{ xor } Q_0$ Output: $Z = \overline{Q_0}$

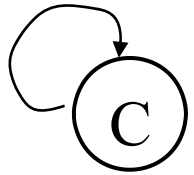
$Q_{0(t+1)} = \overline{Q_1}$

Two flip-flops gives four possible states (A,B,C,D).

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

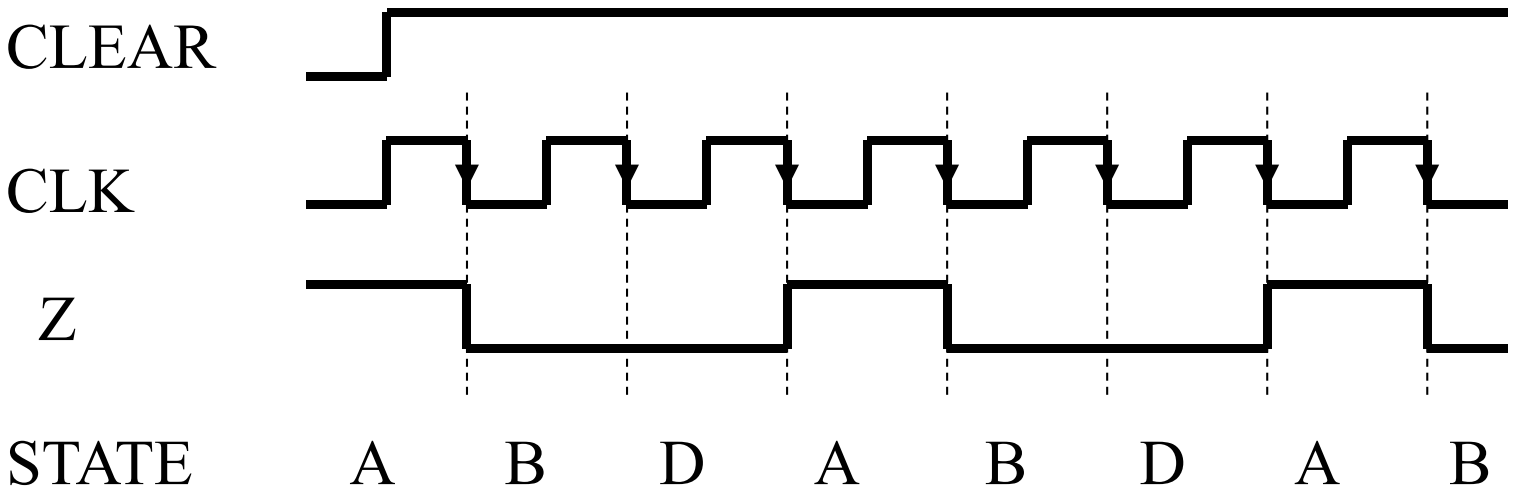
	Present State			Next State		
State	Q_1	Q_0	Z	Q_1	Q_0	State
A	0	0	1	0	1	B
B	0	1	0	1	1	D
C	1	0	1	1	0	C
D	1	1	0	0	0	A





Timing Diagram:

Consider the system to have an active low clear input, again omitted for clarity, that will put it into an initial state of 00 (state A).

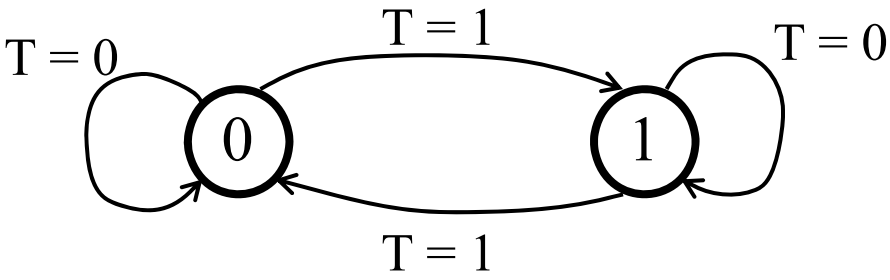


The circuit gives an output pulse every third clock cycle. State C is an unused state and if the circuit entered it, it would get stuck there.

Synchronous Sequential Design

Design a synchronous sequential circuit with an output **F** that has two stable states '0' and '1', and a single input **T**. When **T** is '0' the output remains unchanged. When **T** is 1, the output toggles. Use a D-Type flip-flop and any other logic gates required.

1. Produce a state diagram



2. Produce a state transition table

F	T	$F_{(next)}$
0	0	0
0	1	1
1	0	1
1	1	0

3. Derive excitation equations

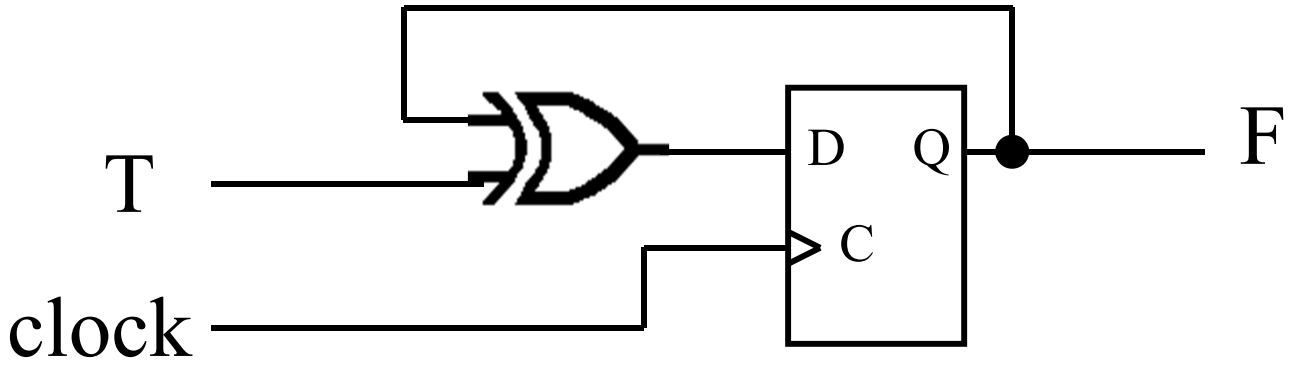
$$F_{(next)} = \overline{F}.T + F.\overline{T} = F \oplus T$$

4. Produce logic diagram

For a D-Type flip-flop, $Q_{next} = D$

The excitation equation gives the combinational logic that must be placed at the D input to obtain the required next state.

$$F_{(next)} = F \oplus T$$



This circuit would require a flip-flop with preset or clear to initialise it to a known state. In general, preset and clear signals are omitted for clarity as they are not part of the sequential design process.

Summary

- A sequential system can be considered to exist in one of a finite number of states.
- The system moves between the states on the application of active clock edges.
- A Moore machine has outputs formed from the state.
- A Mealy machine has outputs formed from the state and the current inputs.