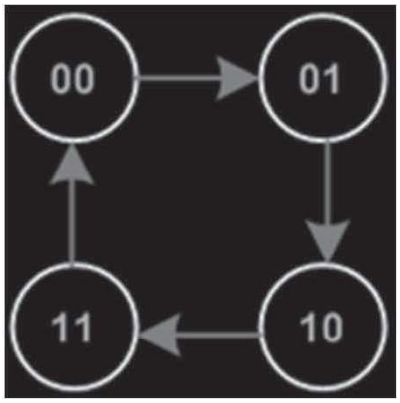
DESIGNING SEQUENTIAL LOGIC CIRCUITS

Steps in designing sequential logic circuits

1) Draw state diagram.
2) Draw the state table (excitation table) for each output.
3) Draw the K-map for each output.
4) Draw the circuit.

Step 1 State Diagram

- Example 1 : state diagram of the 2-bit binary counter. Note: the diagram has no input



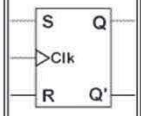
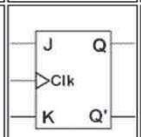
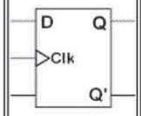
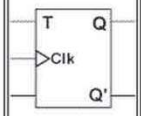
- A state diagram that is made from circles (the states) and arrows (going to the next state) and describes visually the operation of our circuit. It builds up the relationship between various states and later we will shows how inputs affect the states.

Step 2 State Table

- The state table is the same as the excitation table of a flip-flop
- This table gives the inputs required to produce the specific outputs. (refer the excitation table)
- Example: State table based on T Flip Flop

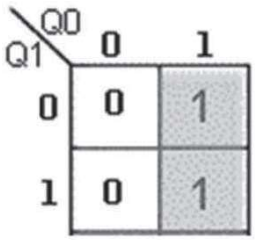
Previous		Next		FF inputs	
Q1	Q0	Q1+	Q0+	T1	T0
0	0	0	1	0	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	0	0	1	1

Summary of the Types of Flip-flop Behaviour

FLIP-FLOP NAME	FLIP-FLOP SYMBOL	CHARACTERISTIC TABLE	CHARACTERISTIC EQUATION	EXCITATION TABLE
SR		S R $Q_{(next)}$ 0 0 Q 0 1 0 1 0 1 1 1 ?	$Q_{(next)} = S + R'Q$ $SR = 0$	Q $Q_{(next)}$ S R 0 0 0 X 0 1 1 0 1 0 0 1 1 1 X 0
JK		J K $Q_{(next)}$ 0 0 Q 0 1 0 1 0 1 1 1 Q'	$Q_{(next)} = JQ' + K'Q$	Q $Q_{(next)}$ J K 0 0 0 X 0 1 1 X 1 0 X 1 1 1 X 0
D		D $Q_{(next)}$ 0 0 1 1	$Q_{(next)} = D$	Q $Q_{(next)}$ D 0 0 0 0 1 1 1 0 0 1 1 1
T		T $Q_{(next)}$ 0 Q 1 Q'	$Q_{(next)} = TQ' + T'Q$	Q $Q_{(next)}$ T 0 0 0 0 1 1 1 0 1 1 1 0

Step 3 K-mapping

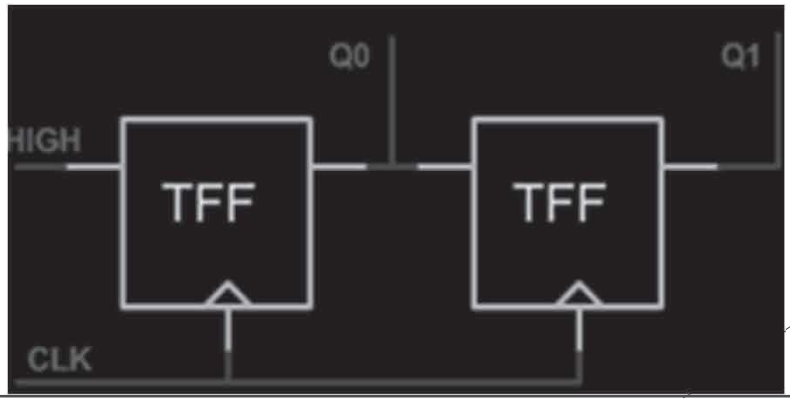
- we can draw a K-map for the inputs i.e. T1 and T0 in the above table. From the table we deduct that we don't need to draw K-map for T0, as it is high for all the state combinations. But for T1 we need to draw the K-map as shown below, using SOP



T1= Q0

Step 4 Circuit

- There is nothing special in drawing the circuit, it is the same as any circuit drawing from K-map output. Below is the circuit of 2-bit up counter using the T flip-flop.



- Based on the previous discussion, counter circuits like this can easily be implemented using T flip flop or JK flip flop as we rely on the toggle operation for frequency division.
- What if a D flip flop is used?

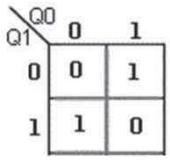
Step 2 State Table

- The state table is the same as the excitation table of a flip-flop
- This table gives the inputs required to produce the specific outputs. (refer the excitation table)
- Example: State table based on D Flip Flop

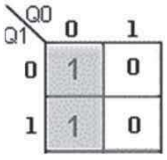
Previous		Next		FF inputs	
Q1	Q0	Q1+	Q0+	D1	D0
0	0	0	1	0	1
0	1	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0

Step 3 K-mapping

- we can draw a K-map for the inputs

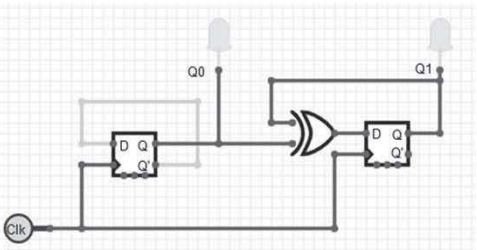


D1= Q0 ⊕ Q1



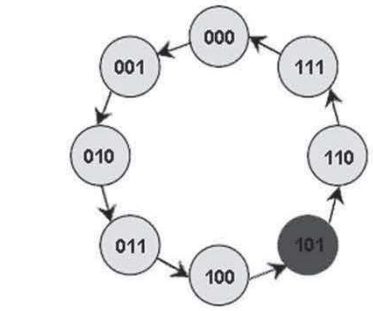
D0 = Q0'

Step 4 Circuit



Example 2 Using J-K flip-flops

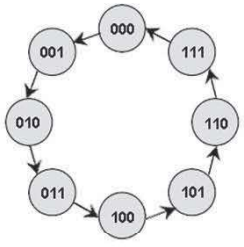
- A counter is first described by a state diagram, which is shows the sequence of states through which the counter advances when it is clocked



5

Deriving the State table from the state diagram

Present State Q2 Q1 Q0	Next State Q2 Q1 Q0
0 0 0	0 0 1
0 0 1	0 1 0
0 1 0	0 1 1
0 1 1	1 0 0
1 0 0	1 0 1
1 0 1	1 1 0
1 1 0	1 1 1
1 1 1	0 0 0

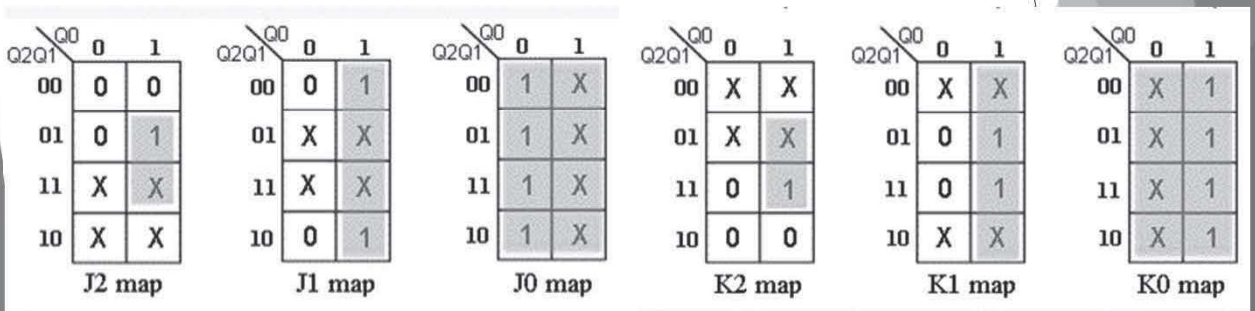


Deriving the Excitation table from the state table (using J-K Flip flops)

Output State Transitions		Flip-flop inputs		
Present State Q2 Q1 Q0	Next State Q2 Q1 Q0	J2 K2	J1 K1	J0 K0
0 0 0	0 0 1	0 X	0 X	1 X
0 0 1	0 1 0	0 X	1 X	X 1
0 1 0	0 1 1	0 X	X 0	1 X
0 1 1	1 0 0	1 X	X 1	X 1
1 0 0	1 0 1	X 0	0 X	1 X
1 0 1	1 1 0	X 0	1 X	X 1
1 1 0	1 1 1	X 0	X 0	1 X
1 1 1	0 0 0	X 1	X 1	X 1

Refer to the excitation table for the values of J and K

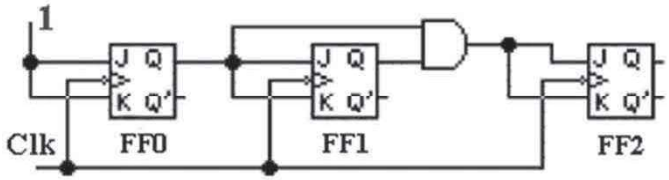
- transfer the JK states of the flip-flop inputs from the excitation table to Karnaugh maps to derive a simplified Boolean expression for each flip-flop input.



Creating the circuit

- The 1s in the Karnaugh maps with "don't cares" and the following expressions for the J and K inputs of each flip-flop are obtained:

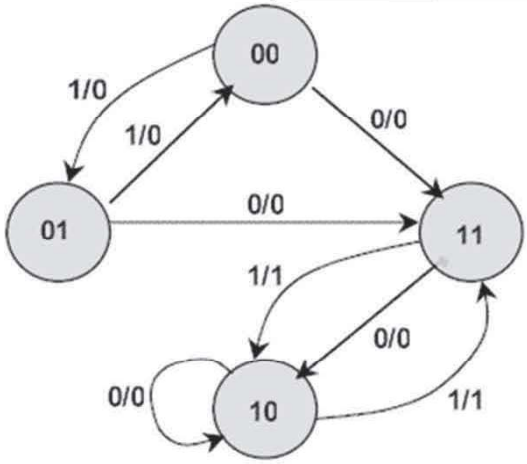
  J0 = K0 = 1
  J1 = K1 = Q0
  J2 = K2 = Q1*Q0



Note: recall the synchronous counter design, creating base k counters and gray code counters is not a problem anymore

A state diagram with 1 input and 1 output(other than the state)

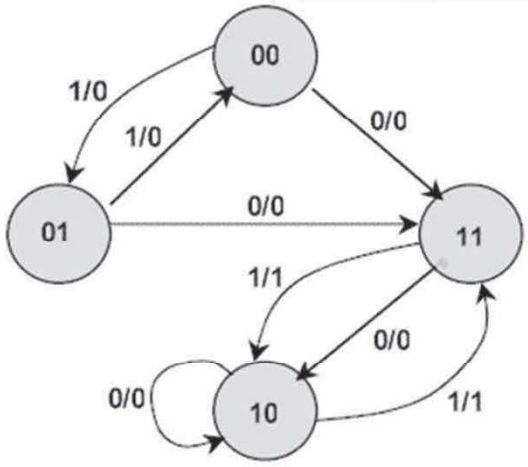
- The binary number inside each circle identifies the state the circle represents.
- The directed lines are labeled with two binary numbers separated by a slash (/). The input value that causes the state transition is labeled first. The number after the slash symbol / gives the value of the output.



- For example, the directed line from state 00 to 01 is labelled 1/0, meaning that, if the sequential circuit is in a present state and the input is 1, then the next state is 01 and the output is 0. If it is in a present state 00 and the input is 0, it will remain in that state.
- A directed line connecting a circle with itself indicates that no change of state occurs.
- The state diagram provides exactly the same information as the state table and is obtained directly from the state table.

Example 3: A state diagram with 1 input and 1 output

Input	Present state		Next State		Output
X	A	B	A	B	Z
0	0	0	1	1	0
0	0	1	1	1	0
0	1	0	1	0	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	1	0	1



Input	Present state		Next State		Output		
X	A	B	A	B	Z	DA	DB
0	0	0	1	1	0	1	1
0	0	1	1	1	0	1	1
0	1	0	1	0	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	1
1	0	1	0	0	0	0	0
1	1	0	1	1	1	1	1
1	1	1	1	0	1	1	0

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

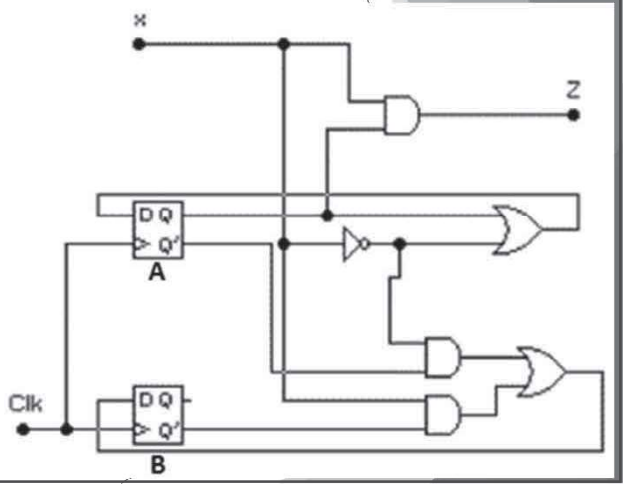
Z = XA

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

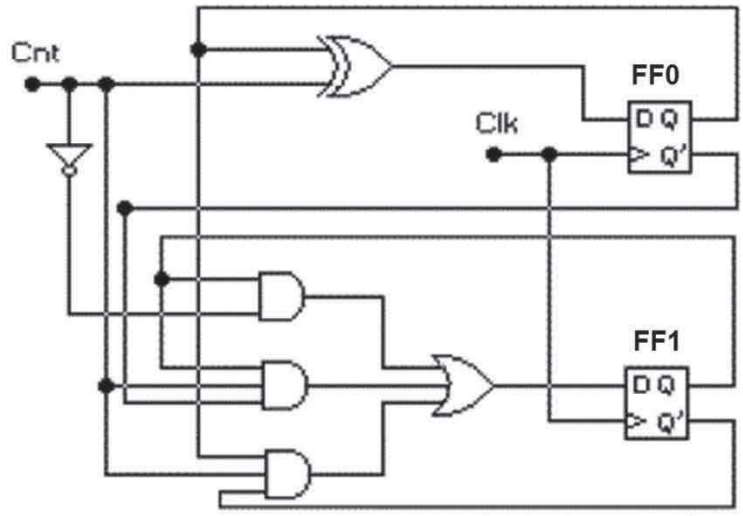
DA = X' + A

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

DA = X'A' + XB'



Example 4 Derive the state table and state diagram for the sequential circuit



Step 1

First we derive the Boolean expressions for the inputs of each flip-flops in the schematic, in terms of external input Cnt and the flip-flop outputs Q1 and Q0. Since there are two D flip-flops in this example, we derive two expressions for D1 and D0:

- **D0 = Cnt⊕Q0 = Cnt'*Q0 + Cnt*Q0'**
- **D1 = Cnt'*Q1 + Cnt*Q1'*Q0 + Cnt*Q1*Q0'**

These Boolean expressions are called **excitation equations** since they represent the inputs to the flip-flops of the sequential circuit in the next clock cycle.

Step 2

Derive the next-state equations by converting these excitation equations into flip-flop characteristic equations. In the case of D flip-flops, Q(next) = D. Therefore the next state equal the excitation equations.

- **Q0(next) = D0 = Cnt'*Q0 + Cnt*Q0'**
- **Q1(next) = D1 = Cnt'*Q1 + Cnt*Q1'*Q0 + Cnt*Q1*Q0'**

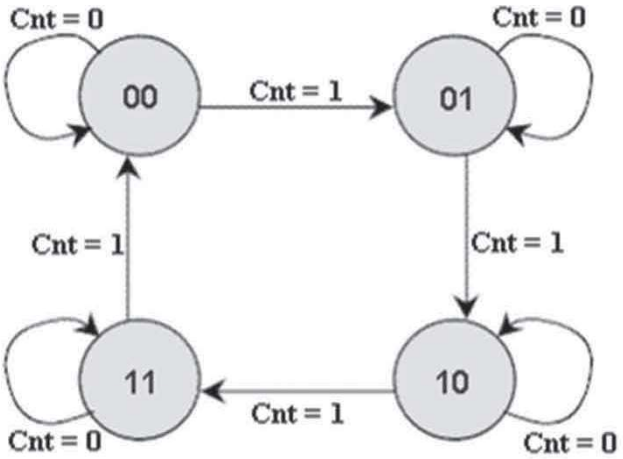
Step 3

- Convert these next-state equations into tabular form called the next-state table.

Input	Previous state		Next state	
Cnt	Q1	Q0	Q1+	Q0+
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

Step 4

- The state diagram is generated directly from the next-state table



Moore and Mealy Machines

A synchronous sequential circuit is also called as **Finite State Machine** (FSM), if it has finite number of states.
There are two types of FSMs.

- Mealy Machine
  - A Mealy Machine is an FSM whose output depends on the present state as well as the present input
- Moore Machine
  - Moore machine is an FSM whose outputs depend on only the present state.

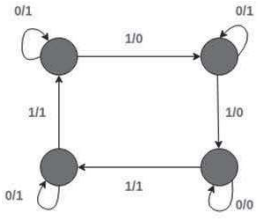


Figure - Mealy machine

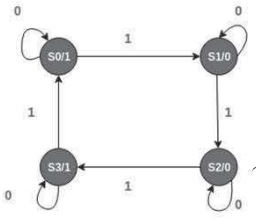
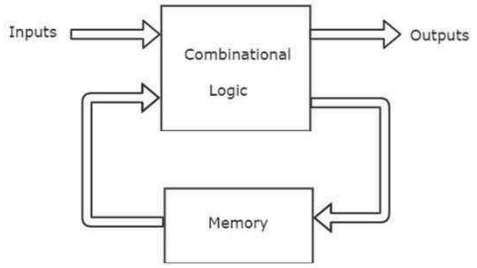


Figure - Moore machine

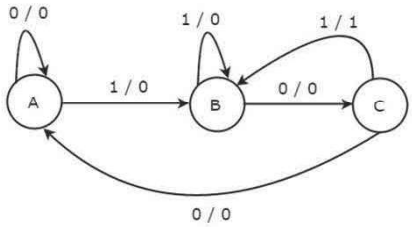
Mealy State Machine

- A Finite State Machine is said to be Mealy state machine, if outputs depend on both present inputs & present states.
- The **block diagram** of Mealy state machine:



There are two parts present in Mealy state machine. Those are combinational logic and memory. Memory is useful to provide some or part of previous outputs ***present states*** as inputs of combinational logic.
So, based on the present inputs and present states, the Mealy state machine produces outputs. Therefore, the outputs will be valid only at positive ***ornegative*** transition of the clock signal.

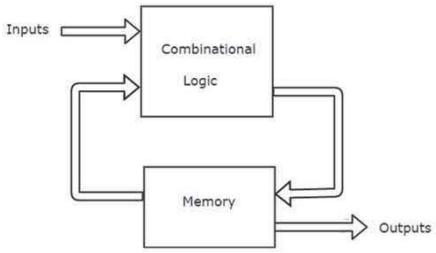
- The **state diagram** of Mealy state machine is shown in the following figure.



- There are three states, namely A, B & C. These states are labelled inside the circles & each circle corresponds to one state. Transitions between these states are represented with directed lines. Here, 0 / 0, 1 / 0 & 1 / 1 denotes **input / output**. In the above figure, there are two transitions from each state based on the value of input, x.
- In general, the number of states required in Mealy state machine is less than or equal to the number of states required in Moore state machine. There is an equivalent Moore state machine for each Mealy state machine.

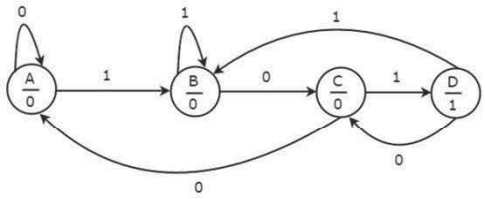
Moore State Machine

- A Finite State Machine is said to be Moore state machine, if outputs depend only on present states.
- The **block diagram** of Moore state machine



- there are two parts present in Moore state machine. Those are combinational logic and memory. In this case, the present inputs and present states determine the next states. So, based on next states, Moore state machine produces the outputs. Therefore, the outputs will be valid only after transition of the state.

- The **state diagram** of Moore state machine is shown in the following figure.



- there are four states, namely A, B, C & D. These states and the respective outputs are labelled inside the circles. Here, only the input value is labeled on each transition. In the above figure, there are two transitions from each state based on the value of input, x.
- In general, the number of states required in Moore state machine is more than or equal to the number of states required in Mealy state machine. There is an equivalent Mealy state machine for each Moore state machine. So, based on the requirement we can use one of them.

Algorithmic State Machines

Every **digital system** can be partitioned into two parts. Those are data path ***digital*** circuits and control circuits. Data path circuits perform the functions such as storing of binary information ***data*** and transfer of data from one system to the other system. Whereas, control circuits determine the flow of operations of digital circuits.

It is difficult to describe the behavior of large state machines using state diagrams. To overcome this difficulty, Algorithmic State Machine ASM charts can be used. ***ASM* charts** are similar to flow charts. They are used to represent the flow of tasks to be performed by data path circuits and control circuits.

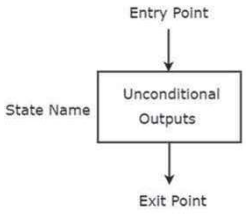
Basic Components of ASM charts

Following are the three basic components of ASM charts.

- State box
- Decision box
- Conditional output box

State box

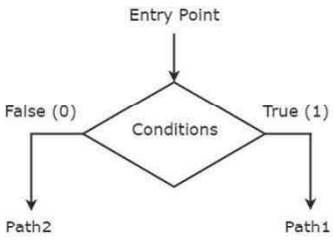
- State box is represented in rectangular shape. Each state box represents one state of the sequential circuit. The **symbol** of state box is shown in the following figure.



- It is having one entry point and one exit point. Name of the state is placed to the left of state box. The unconditional outputs corresponding to that state can be placed inside state box. **Moore** state machine outputs can also be placed inside state box.a

Decision box

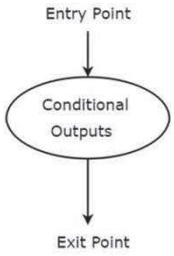
- Decision box is represented in diamond shape. The **symbol** of decision box is shown in the following figure.



- It is having one entry point and two exit paths. The inputs or Boolean expressions can be placed inside the decision box, which are to be checked whether they are true or false. If the condition is true, then it will prefer path1. Otherwise, it will prefer path2.

Conditional output box

- Conditional output box is represented in oval shape. The **symbol** of conditional output box is shown in the following figure.



- It is also having one entry point and one exit point similar to state box. The conditional outputs can be placed inside state box. In general, **Mealy** state machine outputs are represented inside conditional output box. So, based on the requirement, we can use the above components properly for drawing ASM charts.