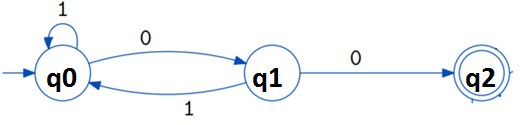
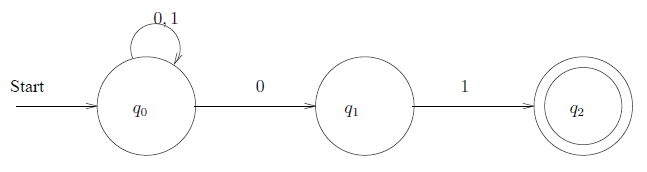
**Finite state machine (deterministic and non-deterministic)**

* **Deterministic (DFA)** :- On applying an input there is one and only one state to which the automaton can transition from its current state



* The language accepted by a DFA **M** is denoted **L(M)**
* It is the set of all strings such that when M starts in its initial state, it ends up in an accepting state.
* If a language is accepted by a DFA, it is said to be a **regular language**.
* Like for above DFA M, regular language will be (1\* (01)\*)\*00.
* **Nondeterministic (NDFA / NFA)** :- On applying an input there may be **more than one state**, to which the automaton can transition from its current state. Means an automaton can be in several states at once (even on same input).

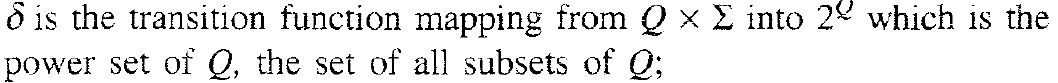
**Example**:-Draw a nondeterministic automaton that accepts all type of strings of 0s and 1s that end in 01.



* The difference between the DFA and the NFA is the type of transition function δ.
* In DFA transition function δ is defined as:-

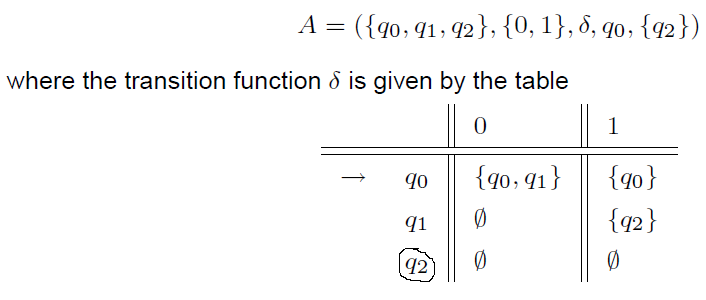
Q X Σ into Q, means on applying an input Ʃ with state Q, it will transit to only one state Q.

* But for a NFA δ is a function



* It takes a state and input symbol, but returns a set of zero or more states (rather than returning exactly one state, as in the DFA)

**Example 1 :- An NDFA accepting strings that end with 01.**



**Draw a transition system for example 1.**

**Example 2 :-Construct a DFA M1 equivalent to**

**NDFA M = ( { q0, q1}, { 0, 1}, δ, q0, { q0 } ),**

**where δ is defined by its transition state table as**

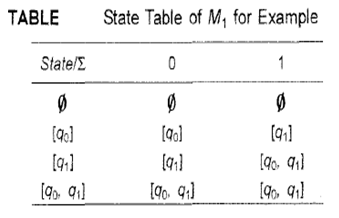
|  |  |  |
| --- | --- | --- |
| State | Input  0 | Input  1 |
| q0 | q0 | q1 |
| q1 | q1 | q0, q1 |

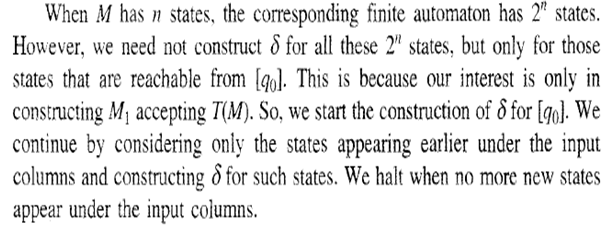
**Steps for the deterministic automaton M1 (2Q, { 0, 1 }, δ, [q0], F):-**

* **Make all subsets of states {q0, q1} as follows**

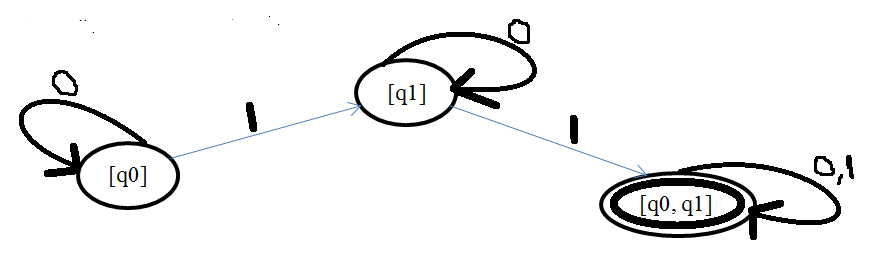
**ϕ , [q0], [q1], [q0, q1]**

* **Identify initial state. Here initial state is [q0].**
* **Final states are the subsets which contain final state as one of the member. Here [q0], [q0, q1] are final states.**
* **Define δ transition function by using table.**

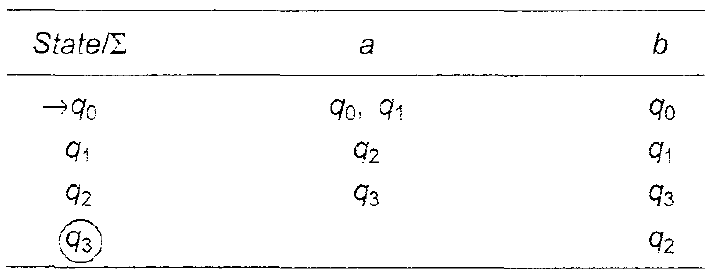




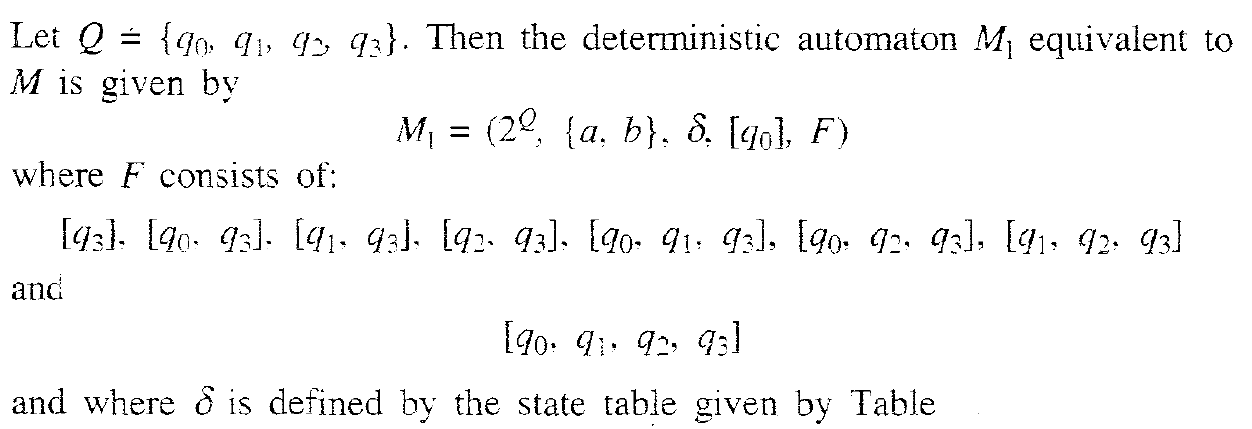
**Transition diagram for DFA M1**



**Q1. Construct a DFA equivalent to M = ({ q0, q1, q2, q3 }, { 0, 1 }, δ, q0, {q3})**

****

**Solution:-**

****

|  |  |  |
| --- | --- | --- |
| **States / Ʃ** | **Input 0** | **Input 1** |
| **[q0]** |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Construction of a Transition System M accepting L(G) for a Regular Grammar G**

Let G = ( {A0,A1,….An}, Ʃ, P, A0 ).We construct a transition system M whose (i) states correspond to variables. (ii) initial state corresponds to A0, and (iii) transitions in M correspond to productions in P. As the last production applied in any derivation is of the form Ai 🡪 a, the corresponding transition terminates at a new state, and this is the unique final state.

We define M = ({q0, q1,…..qf}, Ʃ, δ, q0, {qf}), where δ is defined as follows:

(i) Each production Ai 🡪 aAj induces a transition from qi to qj with label a.

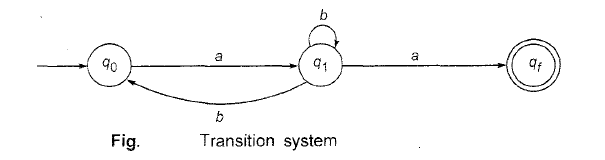
(ii) Each production Ak🡪 a induces a transition from qk to qf with label a.

**Example 1:-**

**Let G= ({** A0 , A1 **},**{a, b}, P,A0**), where P consist of** A0 🡪 aA1, A1 🡪 bA1, A1 🡪 a, A1 🡪 bA0. Construct a transition system M accepting L(G).

**Sol:-**

* Let M = ({q0, q1, qf}, {a, b}, δ, q0, {qf}), where q0 and q1 correspond to A0 and A1, respectively and qf is the new (final) state introduced.
* A0 🡪aA1 induces a transition from q0 to q1 with label a.
* Similarly, A1🡪bA1 and A1 🡪bA0 induce transitions from q1 to q1 with label b and from q1 to q0 with label b, respectively.
* A1 🡪 a induces a transition from q1 and qf with label a.
* Finally corresponding transition system is drawn as follows.

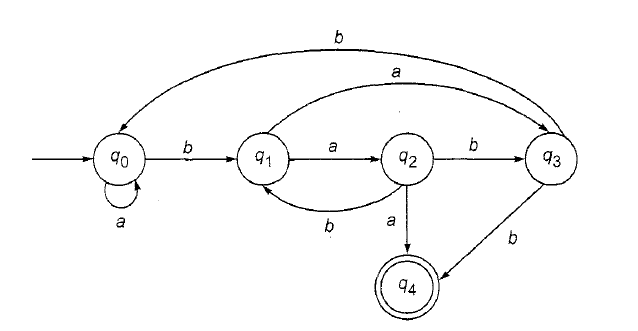
****

**Example 2:- Construct Finite automaton for given grammar G.**

****

**Sol:-**

* Let M = ({ q0, q1, q2, q3, qf }, {a, b}, δ, q0, { qf }), where q0, q1, q2, q3 correspond to A0 , A1, A2 , A3 respectively and qf is the new (final) state introduced.
* Now by looking productions P, draw transition system.

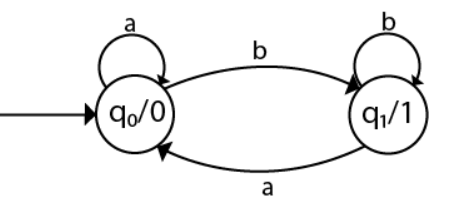


**Q1:- Construct a finite automaton recognizing L(G), where G is the grammar**

**S🡪 aS | bA | b and A🡪aA | bS | a.**

* **Mealy and Moore model (Finite Automata with Outputs)**
* **Finite State Machine (FSM) can have only one state at a time, means at a time one state, applying one input and hence move to another state.**
* **Assume FSM with certain *OUTPUT*, then how it looks like !!!!!**
* **So FSM with input along with output (I/O) are called *Moore machine / Mealy machine*.**

**Example 1:- Moore Machine**

****

**The transition table of given Moore machine is as follows:**

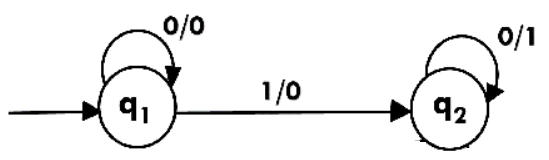
|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Input a** | **Input b** | **Output (λ)** |
| **q0** | **q0** | **q1** | **0** |
| **q1** | **q0** | **q1** | **1** |

**Assume input string is abab, Then**

**δ(q0,abab) :- q0----a-----q0----b-----q1----a-----q0----b----q1**

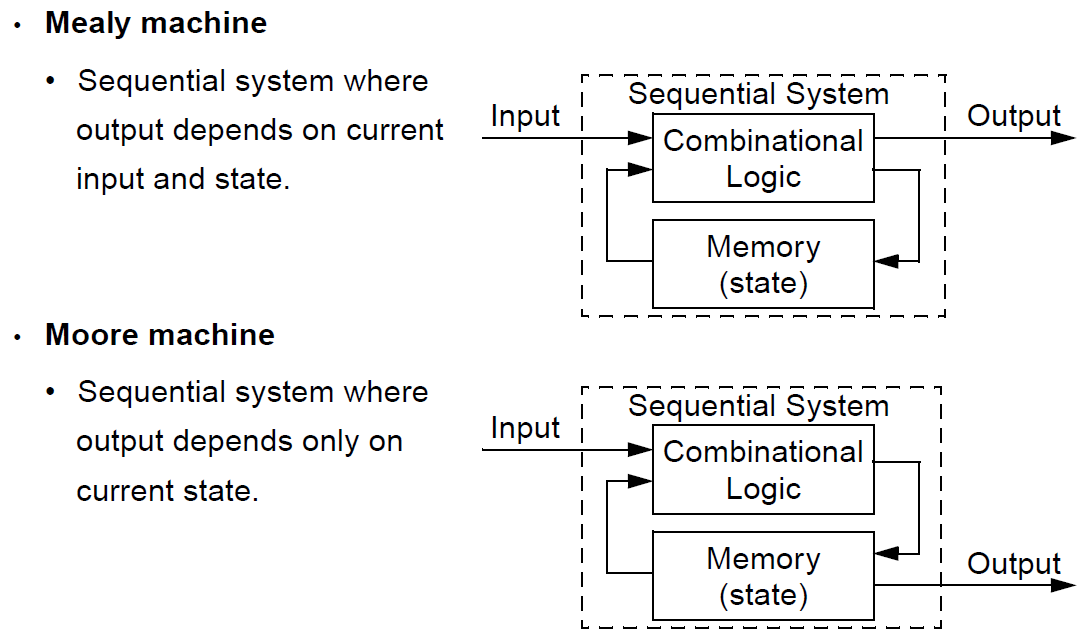
**So output string will be 00101.**

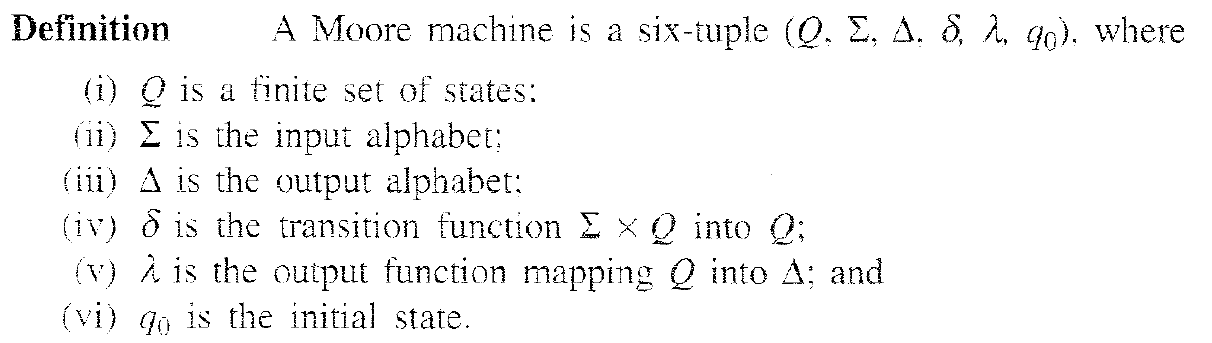
**Example 2:- Mealy Machine**

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Input 0** | **Output** | **Input 1** | **Output** |
| **q1** | **q1** | **0** | **q2** | **0** |
| **q2** | **q2** | **1** | **-** | **-** |

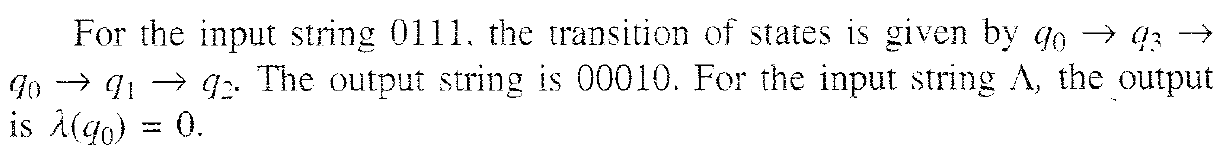
**δ(q1, 0100):- q1-----0-----q1-----1------q2----0-------q2-----0------q2 O/p (0011)**

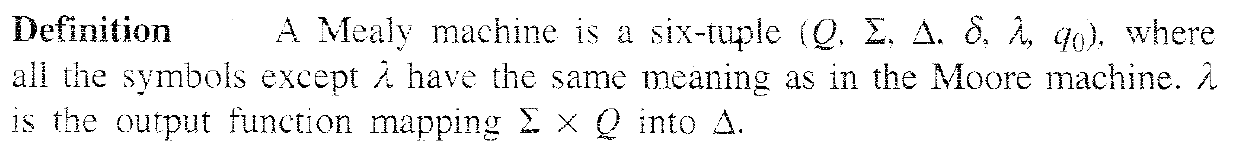
****

****

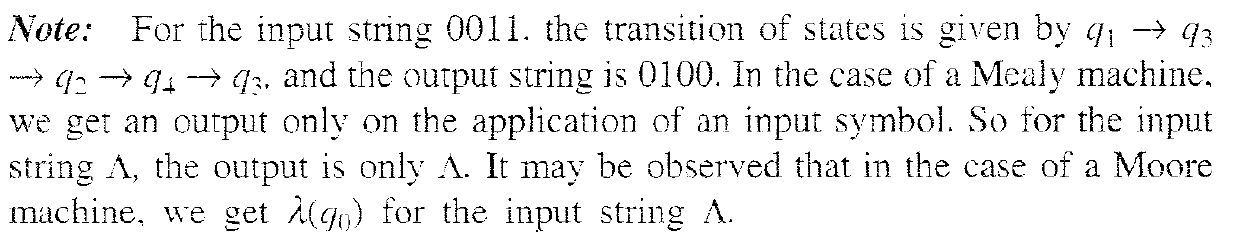
**Moore Machine Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Next State** | | **Output (λ)** |
| **Input a=0** | **Input a=1** |
| **q0** | **q3** | **q1** | **0** |
| **q1** | **q1** | **q2** | **1** |
| **q2** | **q2** | **q3** | **0** |
| **q3** | **q3** | **q0** | **0** |

****

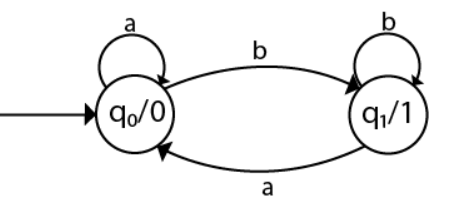
****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Present State** | **Next State** | | | |
| **Input 0** | **Output (λ)** | **Input 1** | **Output (λ)** |
| **q1** | **q3** | **0** | **q2** | **0** |
| **q2** | **q1** | **1** | **q4** | **0** |
| **q3** | **q2** | **1** | **q1** | **1** |
| **q4** | **q4** | **1** | **q3** | **0** |

****

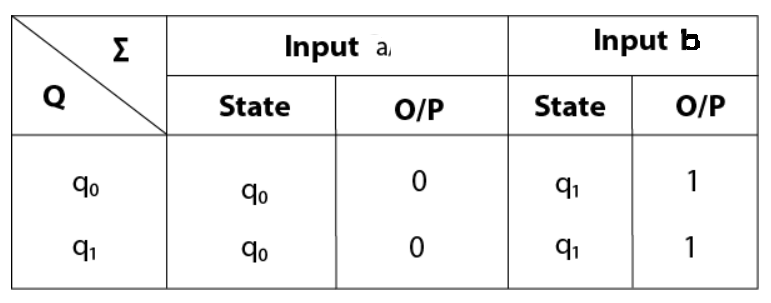
* **In the case of a Moore machine, if the input string is of length n, the output string is of length n + 1.**
* **In the case of a Mealy machine if the input string is of length n. the output string is also of the same length n.**
* **Procedure for Transforming a Moore machine into a Mealy machine.**

**Example 1 :- Convert the following Moore machine into its equivalent Mealy machine**.

****

**The transition table of given Moore machine is as follows:**

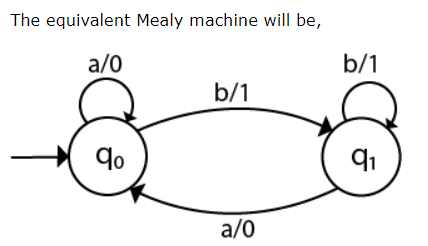
|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Input a** | **Input b** | **Output (λ)** |
| **q0** | **q0** | **q1** | **0** |
| **q1** | **q0** | **q1** | **1** |

****

See the Moore machine table, for every input symbol,

* Form the **pair** consisting of the **next state and the corresponding output**.
* For example, the **states q0 and q1** in the next state column of state **q0**, should be associated with **outputs 0 and 1**, respectively.
* We follow this step for **each row**.

**Hence the transition table for the Mealy machine can be drawn as follows:**

****

**Example 2 :- Consider the Moore machine described by the transition table as follows. We will see steps, how equivalent Mealy machine is constructed.**

**Moore Machine Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Next State** | | **Output (λ)** |
| **Input a=0** | **Input a=1** |
| **q0** | **q3** | **q1** | **0** |
| **q1** | **q1** | **q2** | **1** |
| **q2** | **q2** | **q3** | **0** |
| **q3** | **q3** | **q0** | **0** |

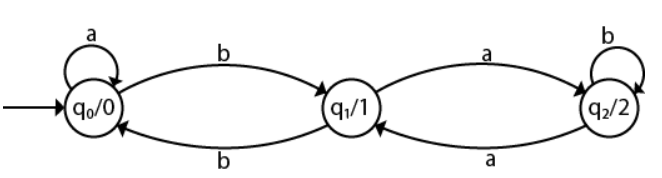
See the Moore machine table, for every input symbol,

* Form the **pair** consisting of the **next state and the corresponding output**.
* For example, the **states q3 and q1** in the next state column of state **q0**, should be associated with **outputs 0 and 1**, respectively.
* We follow this step for **each row**.
* Finally, the **transition table for the Mealy machine** is constructed as follows.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Present State** | **Next State** | | | |
| **Input a=0** | **Output (λ)** | **Input a=1** | **Output (λ)** |
| **q0** | **q3** | **0** | **q1** | **1** |
| **q1** | **q1** | **1** | **q2** | **0** |
| **q2** | **q2** | **0** | **q3** | **0** |
| **q3** | **q3** | **0** | **q0** | **0** |

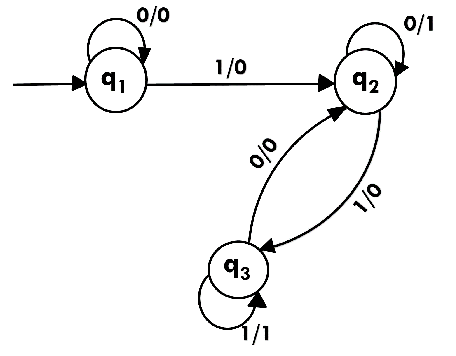
**Q. Convert the given Moore machine into its equivalent Mealy machine.**

**The transaction diagram for the given problem can be drawn as:**

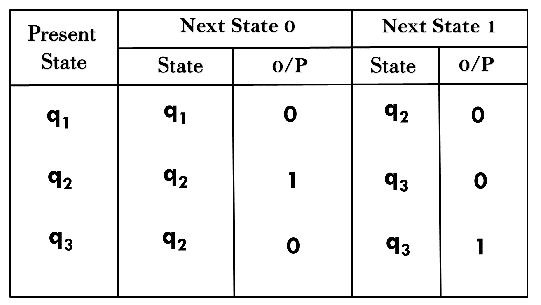
****

* **Procedure for Transforming a Mealy machine into a Moore machine**

**Example 1:- Convert the following Mealy machine into equivalent Moore machine**.



**Transition table for above Mealy machine is as follows:**

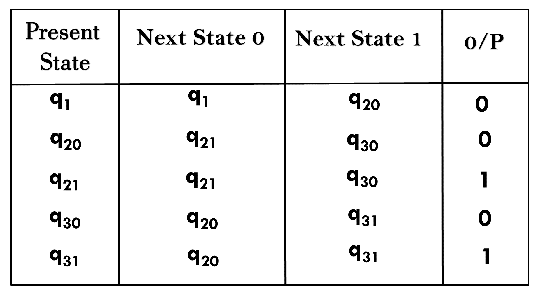


* To start, look into the **next state column** for **any state**, say **qi**.
* We split **qi** into **several different states** (based on the **number of different outputs associated** with **qi**.
* Like here next state **q1** has only **one output 0**.
* Next state **q2 and q3** both have **output 0 and 1**. So we will create **two new states each for q2 and q3.**
* **For q2**, two states will be **q20 (with output 0)** and **q21 (with output 1)**.
* Similarly, for **q3** two states will be **q30 (with output 0)** and **q31 (with output 1)**.

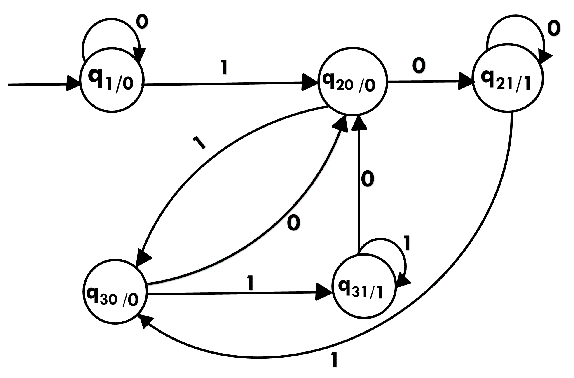
**Table showing the splitting of the states based on their respective output(s) under input column(s).**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Present State** | **Next State** | | | |
| **Input 0** | **Output (λ)** | **Input 1** | **Output (λ)** |
| **q1** | **q1** | **0** | **q20** | **0** |
| **q20** | **q21** | **1** | **q30** | **0** |
| **q21** | **q21** | **1** | **q30** | **0** |
| **q30** | **q20** | **0** | **q31** | **1** |
| **q31** | **q20** | **0** | **q31** | **1** |

* After splitting the states which has different output associated it, now remove the individual output column(s) from the table.
* Like here as next transition state **q2 and q3** have both **output 0 and 1** respectively. So we created **two states each for these states as q20 (with output 0), q21 (with output 1) and q30 (with output 0), q31 (with output 1)**.
* **Finally Transition table for Moore machine will be as follows:**



**Transition diagram for Moore machine will be:**



**Example 2 :- Consider the Mealy machine described by the transition table as follows. We will see steps, how equivalent Moore machine is constructed.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Present State** | **Next State** | | | |
| **Input 0** | **Output (λ)** | **Input 1** | **Output (λ)** |
| **q1** | **q3** | **0** | **q2** | **0** |
| **q2** | **q1** | **1** | **q4** | **0** |
| **q3** | **q2** | **1** | **q1** | **1** |
| **q4** | **q4** | **1** | **q3** | **0** |

* To start, look into the **next state column** for **any state**, say **qi**,
* We split **qi** into **several different states** (based on the **number of different outputs associated** with **qi**.
* For example, here **q1** is associated with **one output 1** and **q2** is associated with **two different outputs 0 and 1.**
* Similarly, **q3** is associated with **one output 0 and q4** is associated with the **outputs 0 & 1.**
* So, we **split q2 into q20 and q21**. Similarly, **q4 is split into q40 and q41**.
* Now State Table can be reconstructed for the new states as follows.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Present State** | **Next State** | | | |
| **Input 0** | **Output (λ)** | **Input 1** | **Output (λ)** |
| **q1** | **q3** | **0** | **q20** | **0** |
| **q20** | **q1** | **1** | **q40** | **0** |
| **q21** | **q1** | **1** | **q40** | **0** |
| **q3** | **q21** | **1** | **q1** | **1** |
| **q40** | **q41** | **1** | **q3** | **0** |
| **q41** | **q41** | **1** | **q3** | **0** |

**The pair of states and outputs in the next state column can be rearranged as given by next Table. This Table shows the Moore machine.**

**Revised State Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Present State** | **Next State** | | **Output (λ)** |
| **Input 0** | **Input 1** |
| **q1** | **q3** | **q20** | **1** |
| **q20** | **q1** | **q40** | **0** |
| **q21** | **q1** | **q40** | **1** |
| **q3** | **q21** | **q1** | **0** |
| **q40** | **q41** | **q3** | **0** |
| **q41** | **q41** | **q3** | **1** |

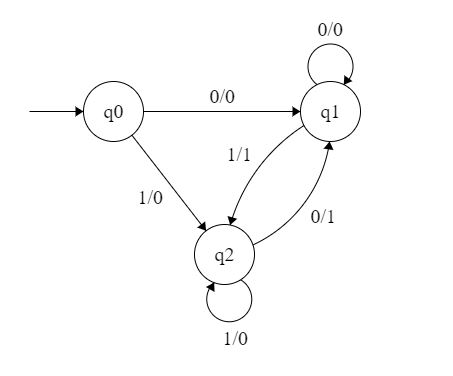
* Here we observe that the **initial state q1 is associated with output 1**, means with **input Ʌ (null)** also, we get an **output of 1.**
* Means this Moore machine **accepts a zero length sequence (null sequence)** which is not accepted by the Mealy machine.
* To overcome this situation, either we must **neglect the response of a Moore machine to input Ʌ (null)**,
* Solution:- **Add a new starting state q0**, whose **state transitions are identical with those of q1 but whose output is 0. (like here yellow highlighted row starting with q0 is added as the first row, just to show the output as ‘0’.**
* So final Table for Moore Machine is as follows.

|  |  |  |  |
| --- | --- | --- | --- |
| **Present State** | **Next State** | | **Output (λ)** |
| **Input 0** | **Input 1** |
| **q0** | **q3** | **q20** | **0** |
| **q1** | **q3** | **q20** | **1** |
| **q20** | **q1** | **q40** | **0** |
| **q21** | **q1** | **q40** | **1** |
| **q3** | **q21** | **q1** | **0** |
| **q40** | **q41** | **q3** | **0** |
| **q41** | **q41** | **q3** | **1** |

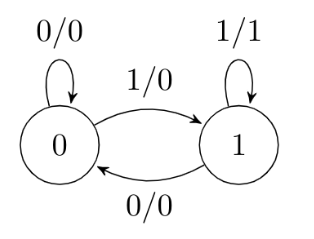
**Q1. Construct a Mealy machine which can output EVEN, ODD according as the total number of l' s encountered is even or odd. The input symbols are 0 and 1.**

**Sol:- ??**

**Q2. Convert the following Mealy machine into equivalent Moore machine**.



**Q3.** **Design Mealy machine for a string of {0,1}\*, in which  it produces another string by replacing the first 1 in any subsequence of consecutive 1’s by a 0 .**

****