By induction, it is proved that $\delta'(q'_0, x) = [P_1, P_2, \ldots\ldots P_k]$ if and only if

$$\delta(q_0, x) = \{P_1, P_2, \ldots\ldots P_k\}$$

So, $\delta'(\delta'(q'_0, x)a) = \delta'([P_1, P_2, \ldots\ldots P_k], a) = [r_1, r_2, \ldots\ldots r_k]$ if and only if

$$\delta(\{P_1, P_2, \ldots\ldots P_n\}, a) = \delta(P_1, a) \cup \delta(P_2, a) \cup \ldots\ldots \cup \delta(P_n, a)$$
$$= r_1 \cup r_2 \cup \ldots\ldots \cup r_k$$

Thus $\delta'(q'_0, xa) = \delta'(\delta'(q'_0, x)a) = [r_1, r_2, \ldots\ldots r_k]$

It is proved that the result is true for a string of length n + 1, if it is true for a string of length n. Now $\delta'(q'_0, x) \in F'$ when $\delta(q_0, x)$ to a state of Q in F.

Therefore, it is proved that L(M) = L(M′).

3.11 Dead State

The name dead state was mentioned in Example 3.6 of converting an NFA to an equivalent DFA. Dead state is a state where the control can enter and be confined till the input ended, but there is no way to come out from that state. (The string is dead or finished on entering the state.)

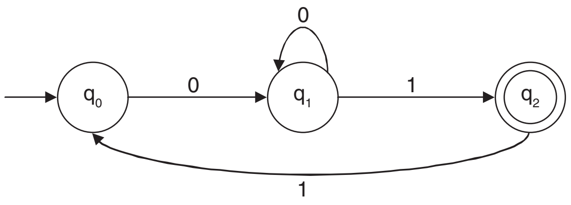

Fig. 3.22

An example for the previous finite automata is shown in Fig. 3.22; from q_0 for input 0, there is a path to go to q_1. But there is no path mentioned from q_0 for input 1. The same thing happens also for q_2 for input 0. Now, let a string 101 be given as input to the FA to test the acceptability of the string by the finite automata.

According to the condition of the acceptability, the first condition, i.e., the string will be totally traversed, is not fulfilled. Obviously, the second condition is also not fulfilled.

We have to decide that the string is not accepted by the finite automata without traversing the string totally!

For making the string totally traversed, we have to include an extra state, say q_f, where the control will go from the states for which there is only one path for one input. In the state q_f, for the inputs, there will be a self-loop. Here, q_f is the dead state.

By including the dead state, the finite automata becomes as shown in Fig. 3.23.

Dead state makes the string totally traversed.

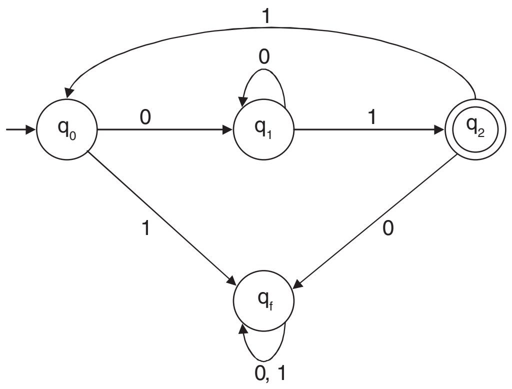

Fig. 3.23

3.12 Finite Automata with Output

Till now, we were discussing machines with no output. But two things, output and output relations, are omitted. This was mentioned in the characteristics of the automaton. As FA is a machine, it must produce output. In this section, we shall discuss finite automata with output. Finite automata with output can be divided into two types:

1. The Mealy machine
2. The Moore machine

3.12.1 The Mealy Machine

The Mealy machine was proposed by *George H. Mealy* at the Bell Labs in 1960.

The Mealy machine is one type of finite automata with output, where the output depends on the present state and the present input.

The Mealy machine consists of six touples

$$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0),$$

where

Q : finite non-empty set of states
Σ : set of input alphabets
Δ : set of output alphabets
δ : transitional function mapping $Q \times \Sigma \to Q$
λ : output function mapping $Q \times \Sigma \to \Delta$
q_0 : beginning state

3.12.2 The Moore Machine

The Moore machine was proposed by *Edward F. Moore* in IBM around 1960.

The Moore machine is one type of finite automata where output depends on the present state only, but the output is independent of the present input.

The Moore machine consists of six touples

$$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0),$$

where

Q : finite non-empty set of states
Σ : set of input alphabets
Δ : set of output alphabets
δ : transitional function mapping $Q \times \Sigma \to Q$
λ : output function mapping $Q \to \Delta$
q_0 : beginning state

3.12.3 Tabular and Transitional Representation of the Mealy and Moore Machines

3.12.3.1 Tabular

The output of the Mealy machine depends on both the present state and the present input. So, for the mealy machine, there will be n number of output columns if there are n number of input.

The tabular format of the Mealy machine is as follows.

	I/P = 0		I/P = 1	
Present State	Next State	O/P	Next State	O/P
A	D	0	B	0
B	A	1	D	0
C	B	1	A	1
D	D	1	C	0

The output of the Moore machine depends on the present state only. As in a machine, there is only one present state column, so there is only one output column.

The tabular format of the Moore machine is given in the following table.

	Next State		
Present State	I/P = 0	I/P = 1	O/P
A	A	B	0
B	A	C	0
C	A	C	1

3.12.3.2 Transitional

The Mealy machine is one type of finite automata with output. The transitional diagram is like finite automata, but the output is mentioned. As we know, the output of a Mealy machine depends on the present state and the present input. So, for a Mealy machine, the transitional arc will be labelled with both input and output like the example in Fig. 3.24.

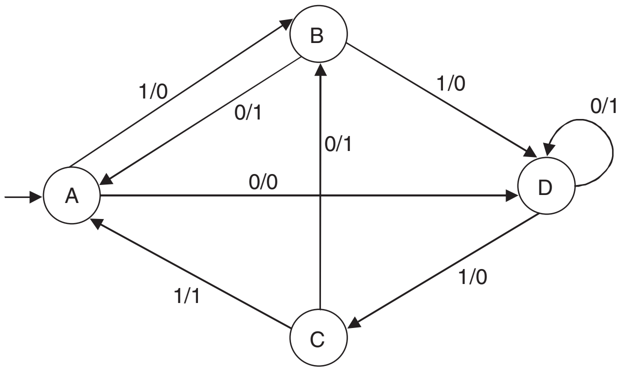

Fig. 3.24

The output of the Moore machine depends only on the present state. For a Moore machine, the states are labelled with the state name and the output like the example shown in Fig. 3.25.

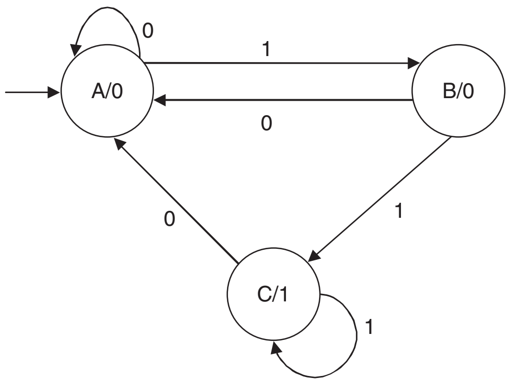

Fig. 3.25

The following are some examples of finite automata with output.

Example 3.10 Design a Mealy machine to get complement of binary number.

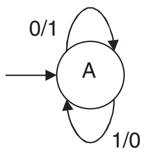

Fig. 3.26

Solution: For input 0, the machine gives the output 1, and for input 1 the machine will give the output 0. The Mealy machine is constructed as shown in Fig. 3.26.

Example 3.11 Design a Moore machine which determines the residue mod-3 for each binary string treated as binary integer.

Solution: Binary string consists of two types of characters 0 and 1. Binary integer means the decimal equivalent of a binary string as integer number. Residue mod-3 means the reminder received when the decimal equivalent of the binary number is divided by 3.

If a decimal number is divided by 3, only three types of reminders 0 or 1 or 2 are generated. The output of the Moore machine depends only on the present state. So, for three types of reminders (output), three states are needed.

Let us assume that state q_0 is producing output 0, state q_1 is producing output 1, and state q_2 is producing output 2. Now, it is needed to construct the transitional arcs. Before construction, take a table containing binary equivalent of decimal numbers from 0 to 9 and the reminders for each of them, when divided by 3.

Decimal	Binary	Reminders
0	0	0
1	1	1
2	10	2
3	11	0
4	100	1
5	101	2
6	110	0
7	111	1
8	1000	2
9	1001	0