CS & IT ENGINEERING

THEORY OF COMPUTATION

REGULAR EXPRESSION

Lecture No.- 01

By- Venkat sir

Recap of Previous Lecture

Topic ?????

DFA Construction

minimization of DFA

Complement of DFA

DFA ⇒ Language

NFA ⇒ DFA

ϵ-NFA ⇒ NFA ⇒ DFA

Topics to be Covered

Topic ✓ Conversion from ∈-NFA to NFA → DFA

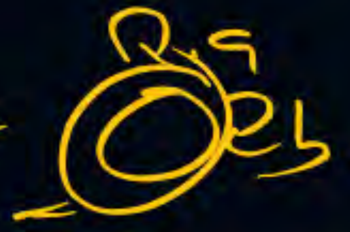
Topic ?? Mealy m/c and moore m/c

Topic ?? Regular Expression

Topic ??

Topic : Regular Expression

$L = \{\epsilon, a, b ---\} = \underline{(a+b)^*} =$



- The simplest way of representing a regular language is known as Regular expression.
- For every regular language regular expression can be constructed.

$L = \{a^n b^n \mid n \geq 0\}$ ✗

- To construct regular expression following 3 operators are used.

1 + is known as union operator

$r_1 + r_2 = \{r_1, r_2\}$

OR

2 . is known as concatenation operator

$r_1 \cdot r_2$ r_1 and r_2

3 * is known as Kleene closure operator

r_1^* :) Repeat = $\{r_1, r_1^2, r_1^3 ---\}$

Topic : NOTE

- For one regular language many number of regular expressions can be possible.

- One regular expression can generate only one regular language.

Regular Language → Regular Expression

① $L = \{\ \}$ → ϕ

② $L = \{\epsilon\}$ → ϵ

③ $L = \{a\}$ → a

④ $L = \{a, b\}$ → $\underline{\underline{a}} + \underline{\underline{b}}$

⑤ $L = \{aa, ab, ba\}$ → $aa + ab + ba$

Reg Lang $\longrightarrow$ Regular Expression

⑥ $L = \{\epsilon, a, a^2, a^3 ---\} \longrightarrow a^*$ (Kleene closure)

⑦ $L = \{a, a^2, a^3, a^4 ---\} \longrightarrow a^+$ (positive closure)

⑧ $L = \{\epsilon, a, b, aa, ab, ba, bb ---\} \longrightarrow (a+b)^*$

⑨ $L = \{a, b, aa, ab, ba, bb ---\} \longrightarrow (a+b)^+$

Reg Expr

⑩ $L = \{\underline{a}^n \underline{b}^m \mid n, m \geq 1\} \longrightarrow a^+ b^+$

⑪ $L = \{a^n b^m \mid \begin{matrix} n \geq 0 \\ m \geq 1 \end{matrix}\} \longrightarrow a^* b^+$

⑫ $L = \{a^n b^m \mid \begin{matrix} n \geq 1 \\ m \geq 2 \end{matrix}\} \longrightarrow a^+ \underline{b} \cdot \underline{b}^+$

⑬ $L = \{a^n b^m \mid \begin{matrix} n \geq 2 \\ m \geq 3 \end{matrix}\} \longrightarrow \underline{a\, a^+} b \cdot \underline{b\, b^+}$

Reg Expr

$n, m \geq 0$

(14) $L = \{a^n b^m \mid n > m\}$ → not possible

(15) $L = \{a^n b^m \mid (n > m) \text{ (and) } (n < m)\} = \{\ \} = \phi$

(16) $L = \{a^n b^m \mid \underline{n > m} \text{ (or) } n < m\} = \{a^n b^m \mid n \neq m\}$

not possible

#Q. Construct regular expression that generates set of all strings of a's and b's where 4th input symbol is a from left side.

$$\underline{(a+b)}\ \underline{(a+b)}\ \underline{(a+b)}\ \underline{a}\ \underline{\underline{(a+b)^*}}$$

#Q. Construct regular expression that generates set of all strings of a's and b's where 4th input symbol is b from end. (R.H.S)

$$\left\{ \underline{(a+b)^* \underline{b} \; \underline{(a+b)}\,\underline{(a+b)}\,\underline{(a+b)}} \right\}$$

#Q. Construct regular expression that generates set of all odd length palindrome strings over {a}.

$\longrightarrow$ malayalam $\longleftarrow$

$$\{a^1, a^3, a^5, a^7 \ldots\} = \underline{a(aa)^*}$$

$\{$nitin, liril$\}$

#Q. Construct regular expression that generates set of all even length palindrome strings over {a, b}.

not possible

$\{a \cdots z\}$

#Q. Construct regular expression that generates set of all odd length palindrome strings of English language.

not possible

Topic : NOTE

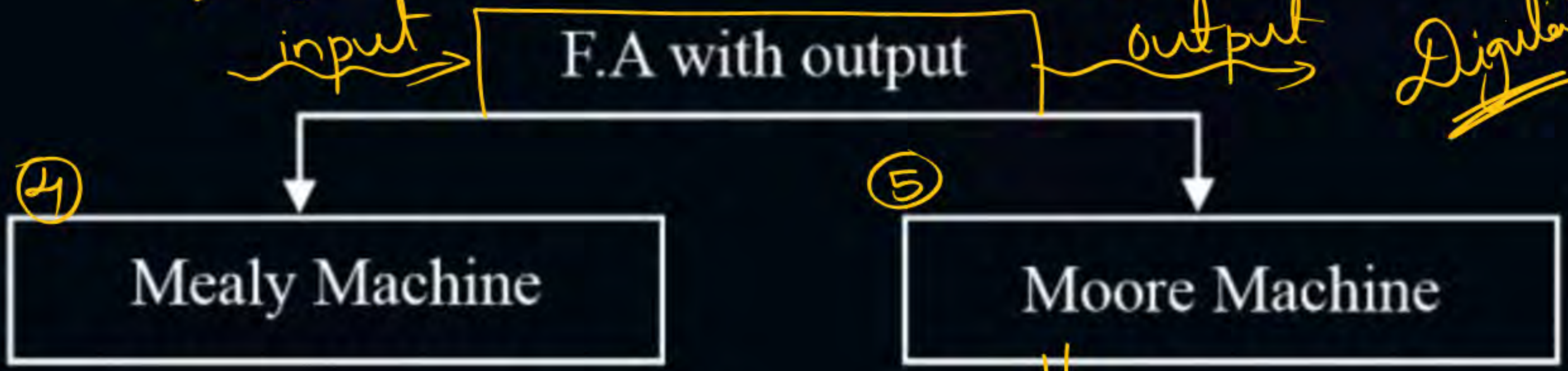
- Palindrome languages over more than one symbol are not regular .Hence regular expression not possible.

- ✓ Palindrome languages over one symbol are regular.

$\{WW^R\}$ $W \in (a+b)^*$

$\Sigma = \{a, b\}$

input



F.A with output

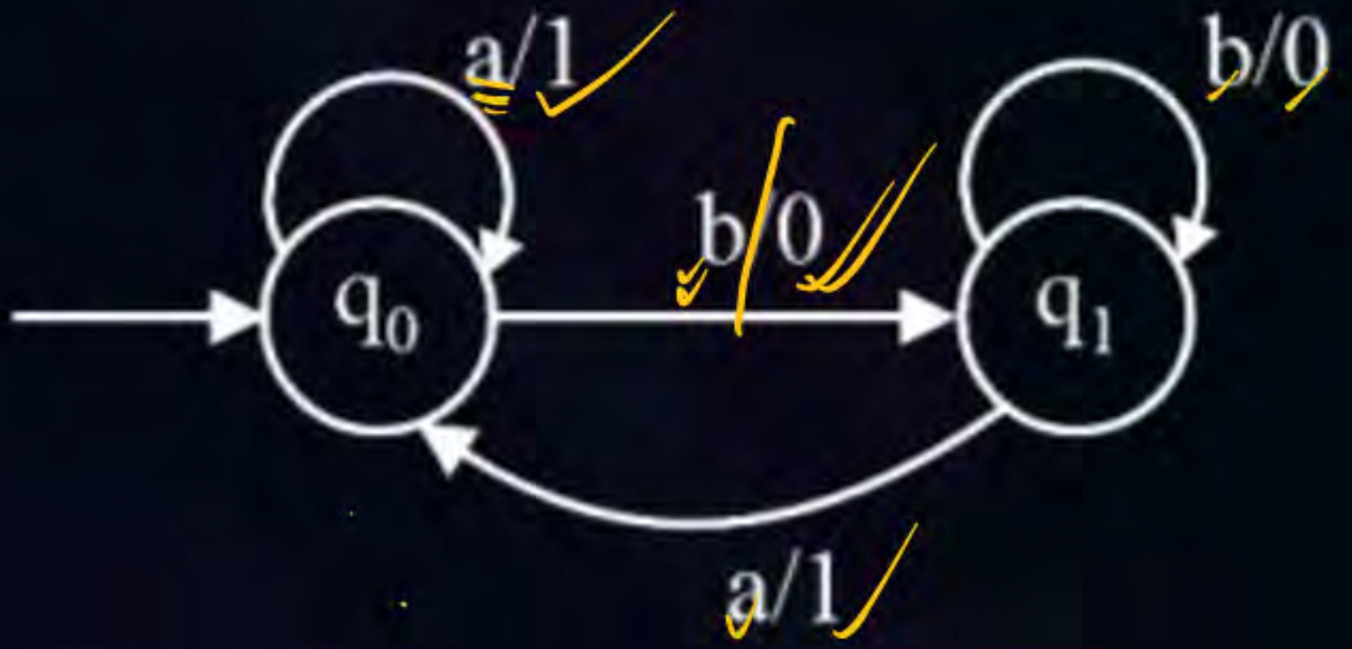
(4) Mealy Machine

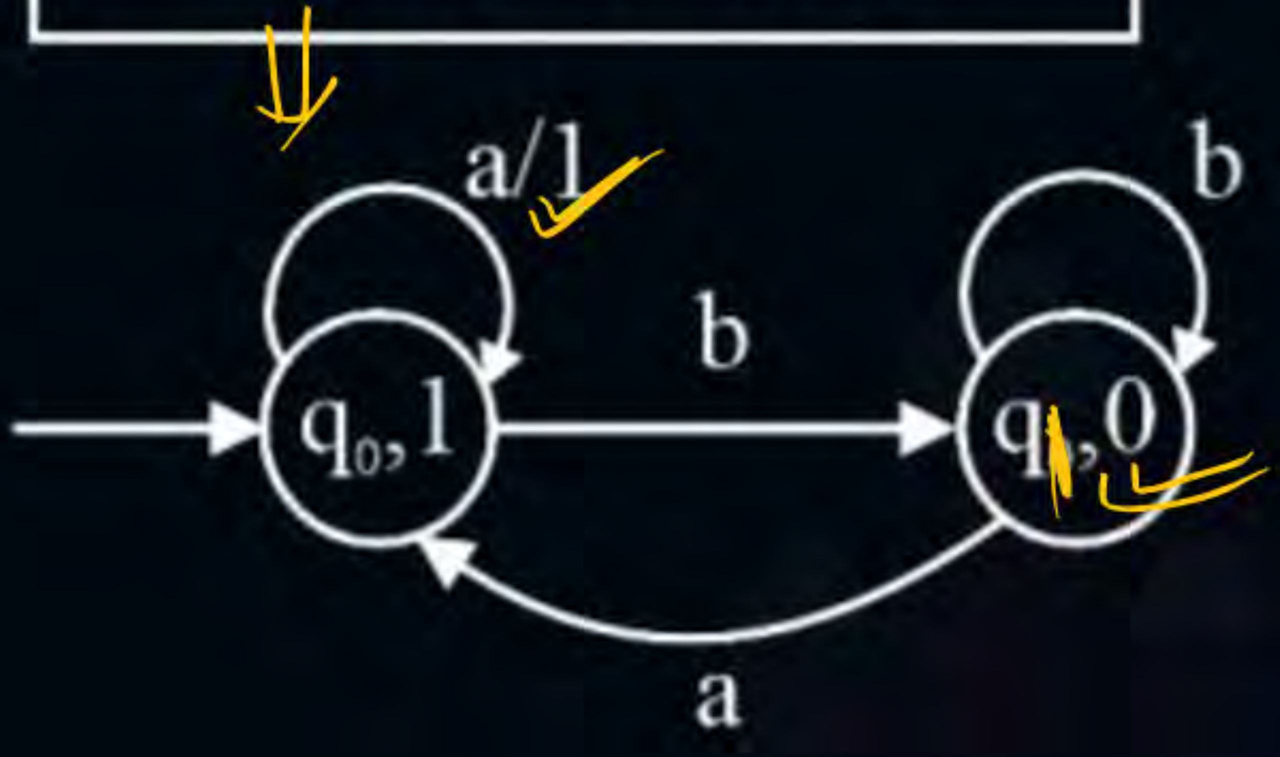
(5) Moore Machine

output

$\Delta = \{0, 1\}$

Digital Circuit Design





① DFA

② NFA

③ ε-NFA

④ Mealy m/c

⑤ Moore m/c

- Mealy Machine:
- It is a mathematical model in which output is associated
- with transition.

- Moore Machine:
- It is a mathematical model in which output is associated With state
- with state.

formal Definition : $\boxed{(Q, \Sigma, q_0, \Delta, \delta, \lambda)}$

✓ Q : finite no.of states

Σ : input alphabet

✓ q_0 : initial state

Δ : output alphabet

✓ δ : transition function : $\boxed{Q \times \Sigma \rightarrow Q}$

✓ λ : output function

Mealy m/c $\lambda : Q \times \Sigma \rightarrow \Delta$

Moore m/c $\lambda : Q \rightarrow \Delta$

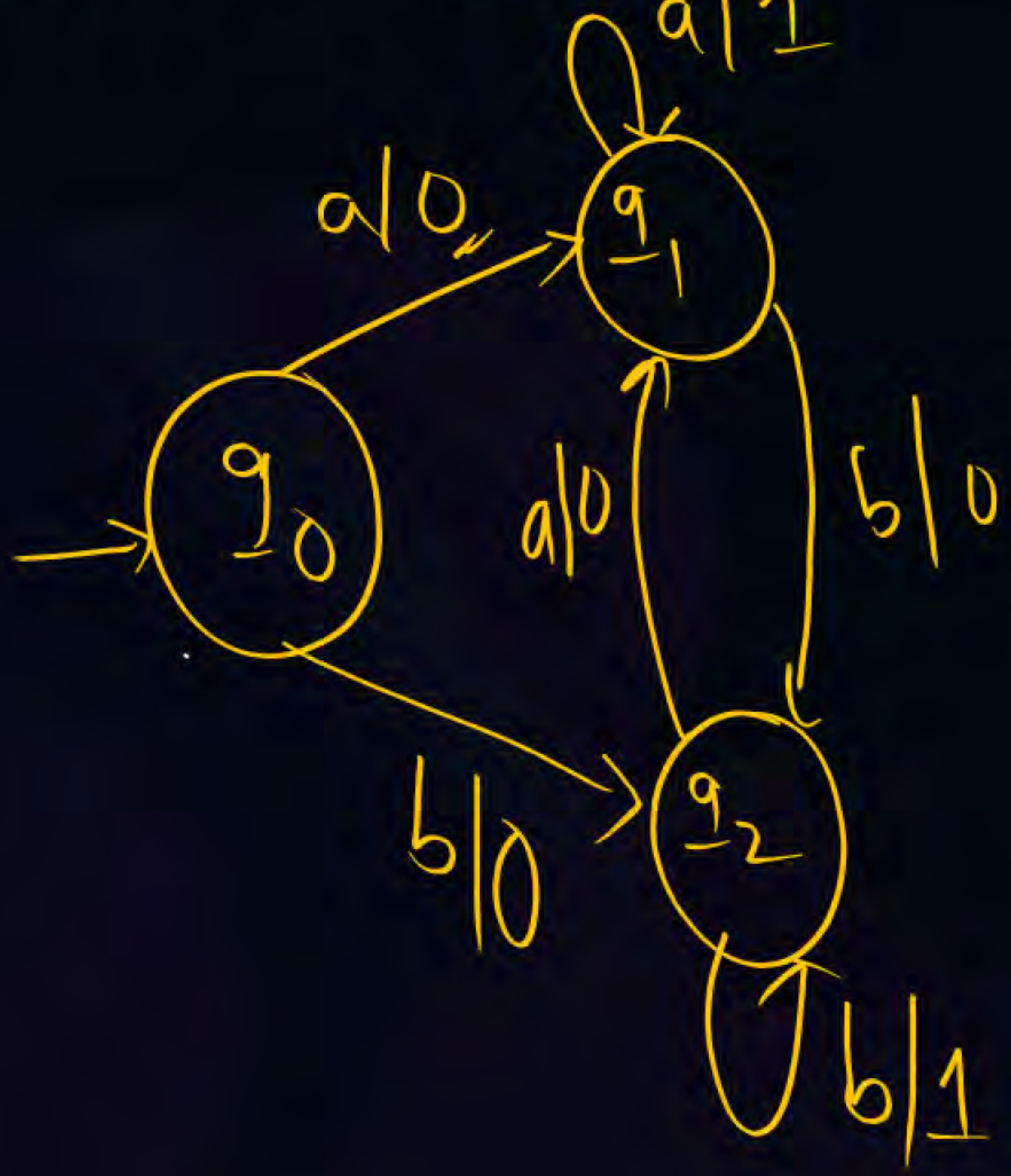
NOTE

{ no final state present in Mealy and Moore m/c }

[NAT]

aaaaabb bbbbaa

#Q. Construct mealy machine that takes all strings of a's and b's as input and produces 1 as output if last two symbols in the input are same otherwise produces 0 as output.



$\Sigma = \{a, b\}$

a's b's → $\boxed{\text{Mealy m/c}}$ → $\Delta = \{0, 1\}$

$\left\{\begin{matrix} aa \\ bb \end{matrix}\right\} \longrightarrow 1$

$\left\{\begin{matrix} a \\ b \\ ba \end{matrix}\right\} \longrightarrow 0$

[NAT]

10000

#Q. Construct mealy machine that takes all strings of 0's and 1's as input and produces A as output if input ending with 10 or produces B as output if input ending with 11 otherwise produces output C.

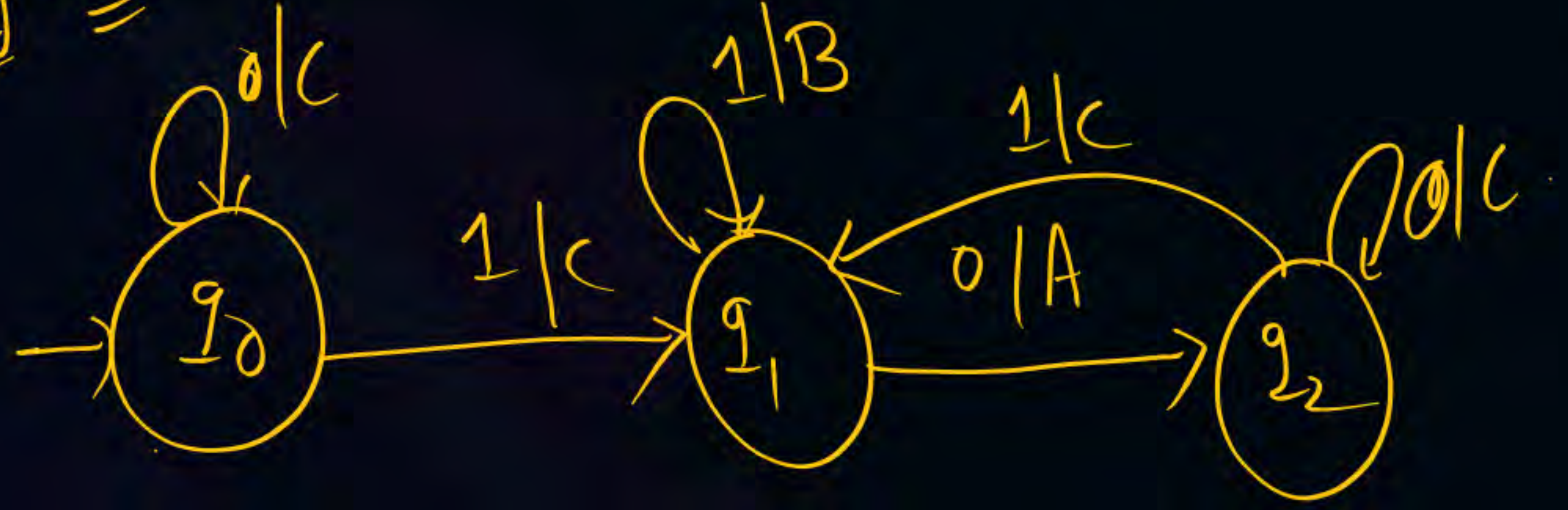
0000

1010

$\Sigma = \{0, 1\}$

$\Delta = \{A, B, C\}$

Mealy m/c

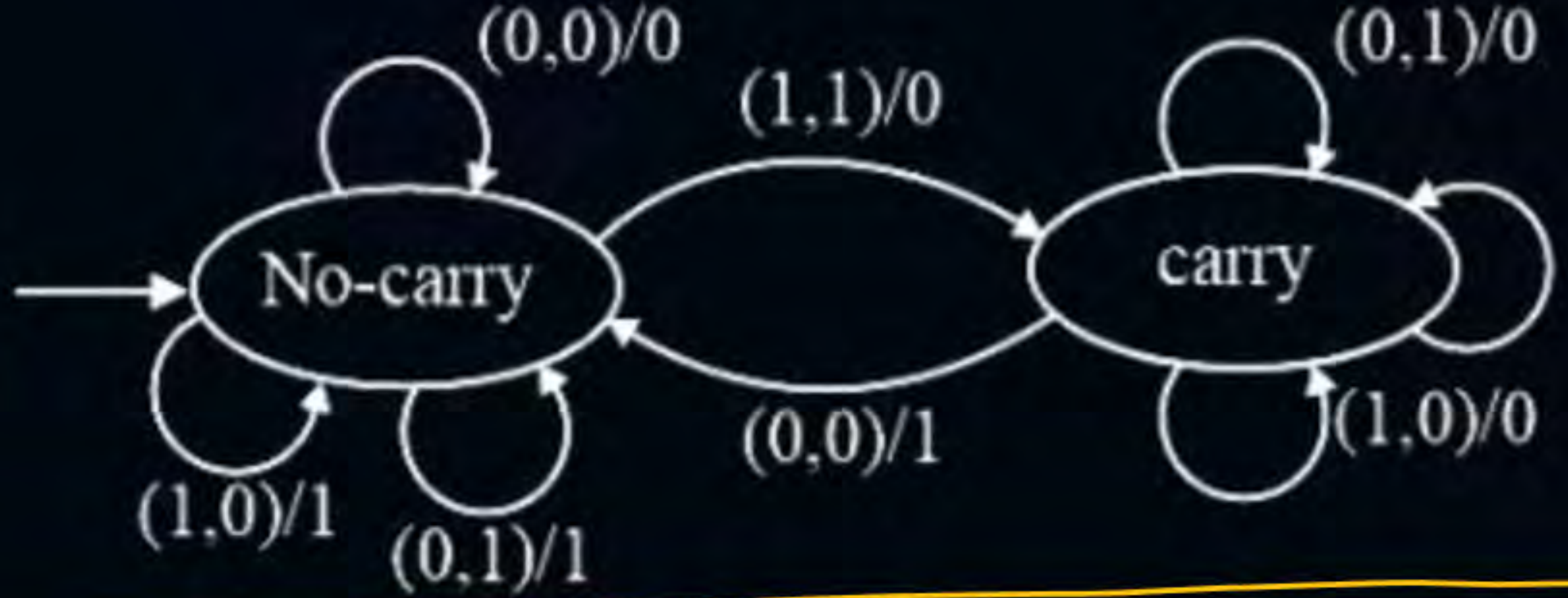


10 → A

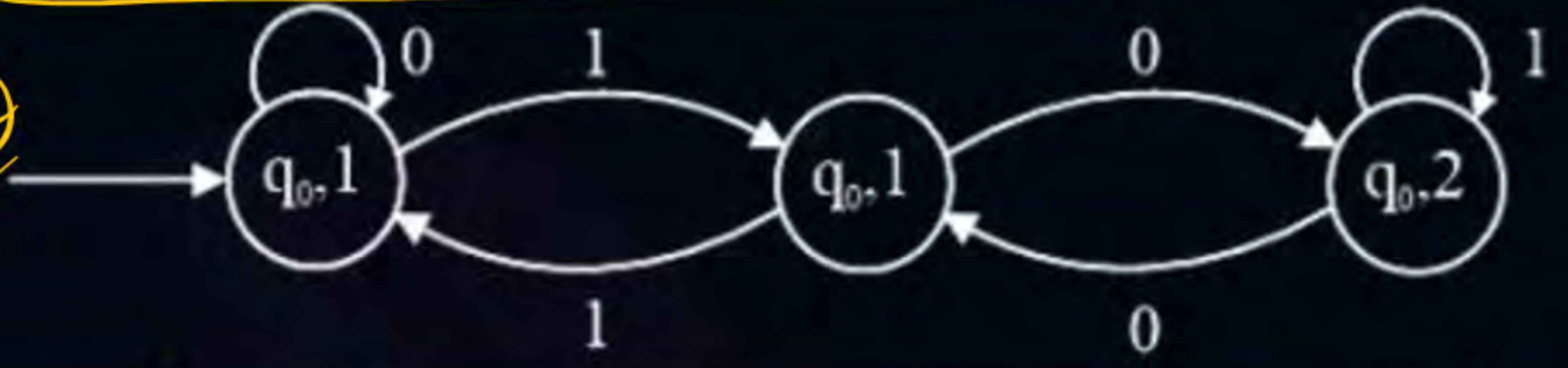
11 → B

0, 1, 01 → C

①

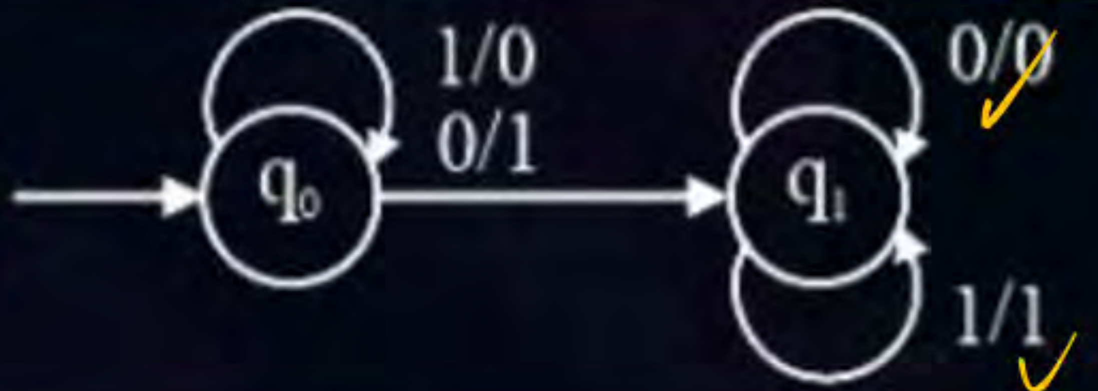


②



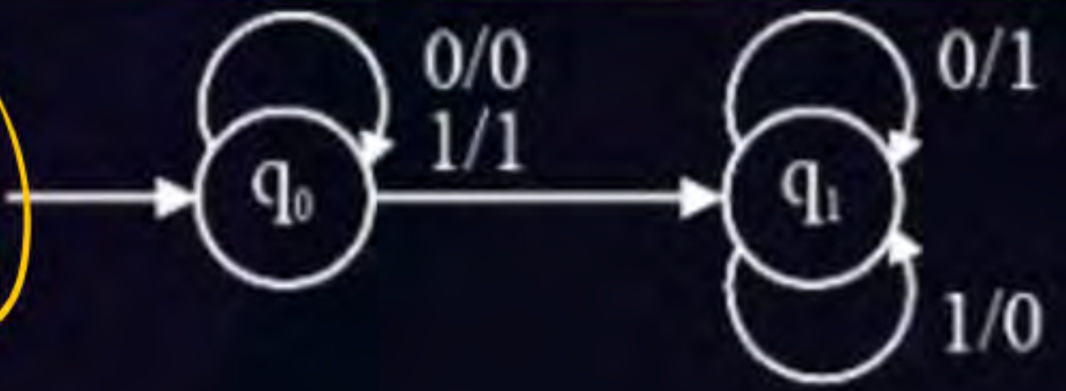
Moore

③



Mealy one

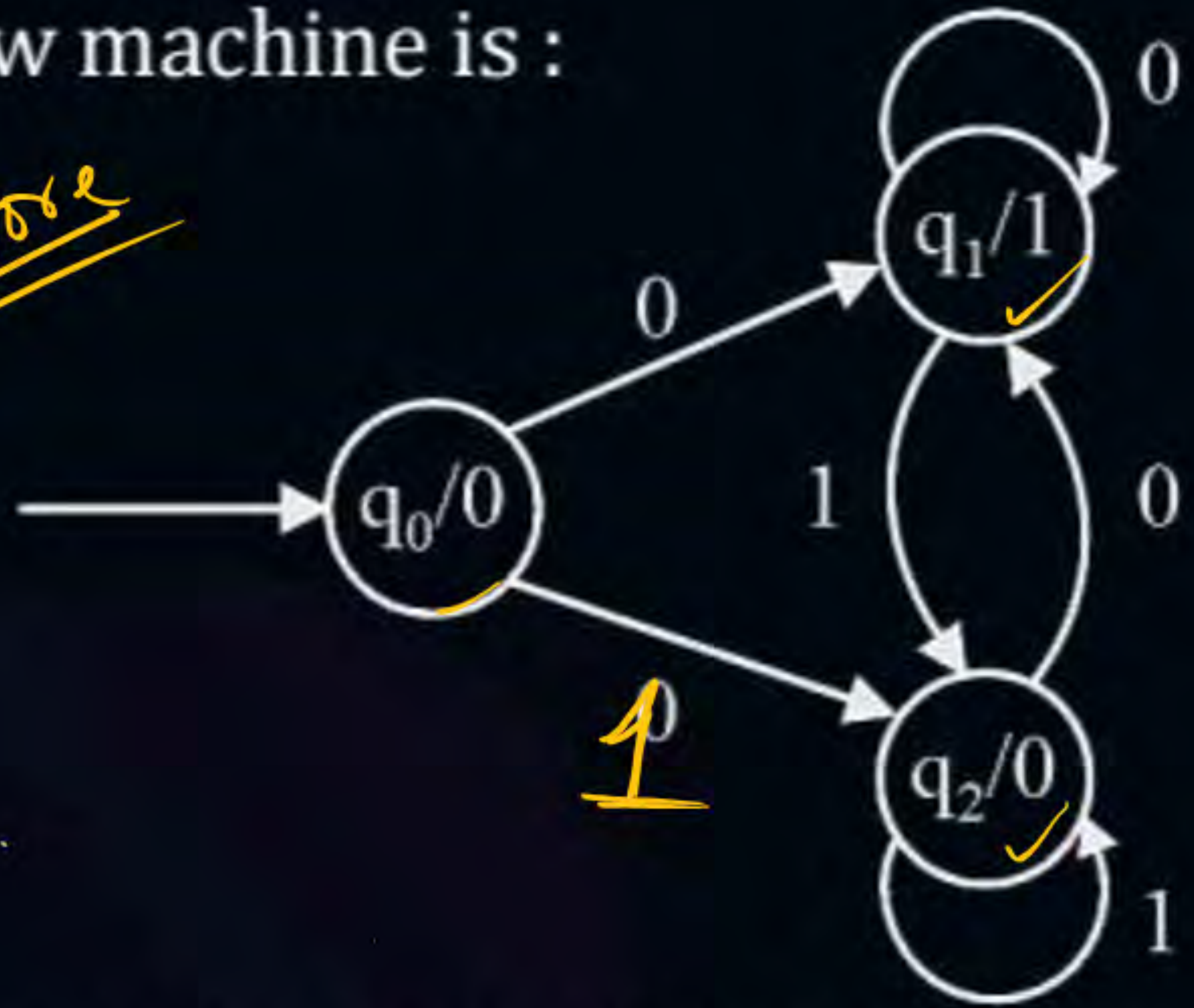
④



[MCQ]

#Q. The below machine is :

Moore



0 1
1 1
1 0

A A Mealy machine to find 2's complement of a number

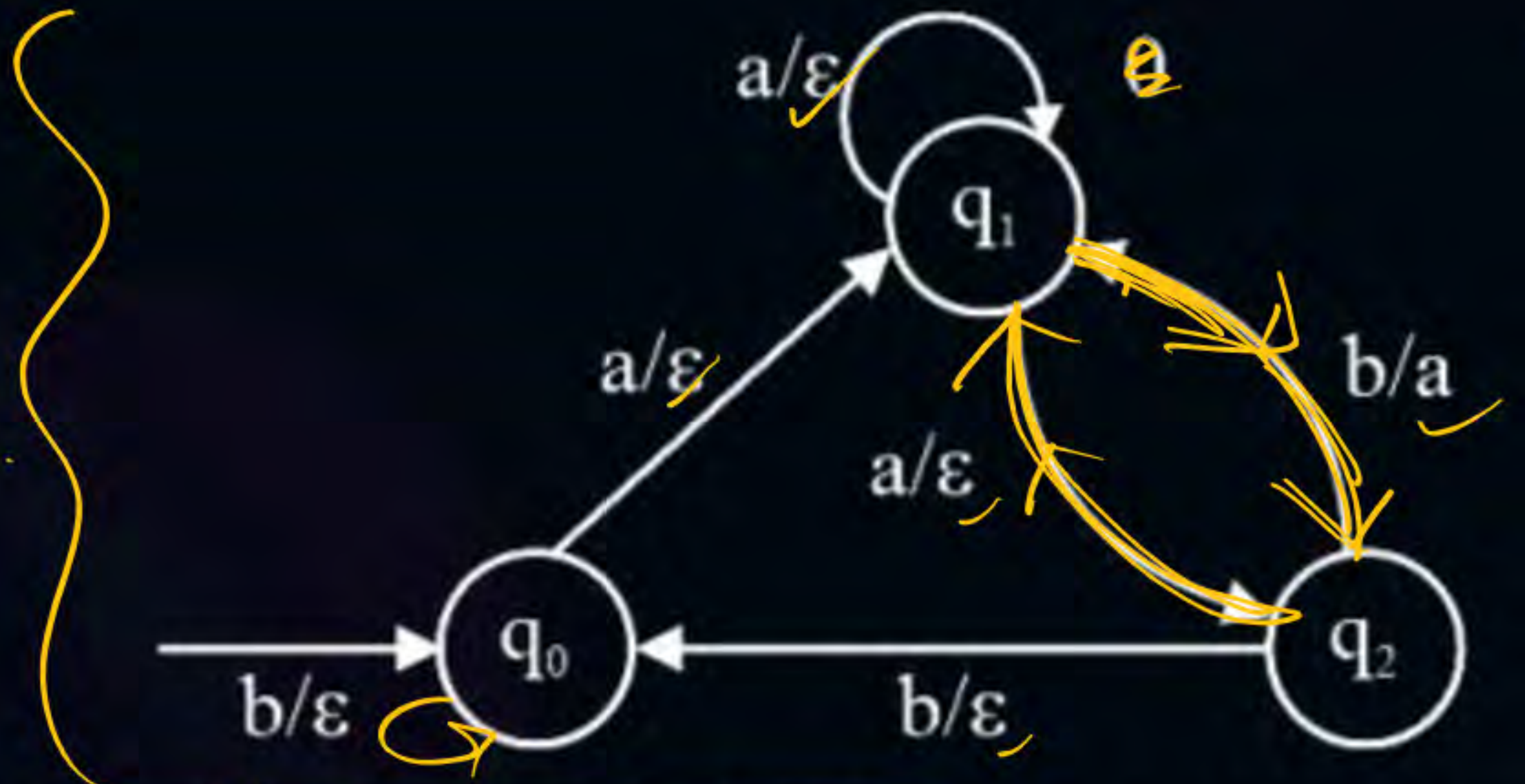
B A Moore machine to find 2's complement of a number

C A Mealy machine to find 1's complement of a number

[✓ D] A Moore machine to find 1's complement of a number

[MCQ]

#Q. Consider the following finite state transducer where the label on an edge x/t denotes if the input is x, follow the arrow and emit t



Mealy m/c

For the input, aabbbaaabbbabaabb the output is :

a a a a

[✓ A aaaa]

B aaaaaaaaa

C ab ab ab ab

D abbbabbbbababb

THANK - YOU