Mealy to Moore:

$$M_0 = (Q \times \Delta, \Sigma, \Delta, \delta', \lambda', (q_0, \epsilon))$$

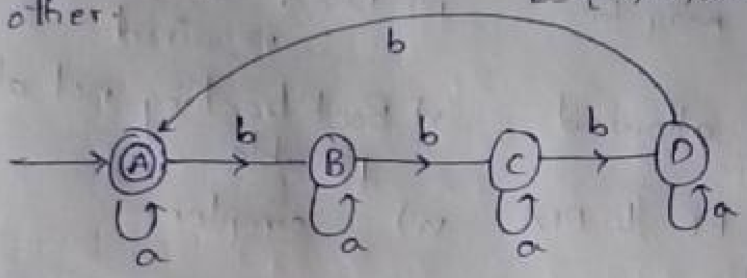
$$\delta'((q,b),a) = (\delta(q,a), \lambda(q,a))$$

$$\lambda'(q,a) = \lambda(\delta(q,a))$$

IMP Que's

1a) DFA: An FA is called DFA if it has ~~b~~ only one path for specific input value from one state to other.

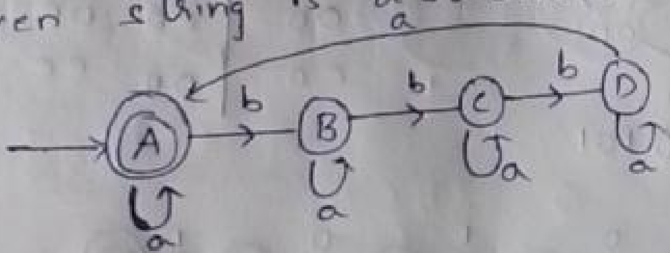
L = {a, aa, aaa, aaaa, a babbab, ...}



Transition Table

δ	a	b
→(A)	A	B
B	B	C
C	C	D
D	D	A

Given string is "ababbab"



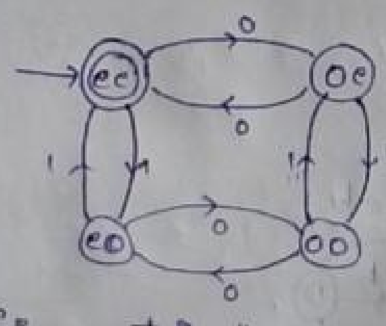
Moves are

$$A \xrightarrow{a} A \xrightarrow{b} B \xrightarrow{a} B \xrightarrow{b} C \xrightarrow{b} D \xrightarrow{a} D \xrightarrow{b} A$$

b)

DFA	NFA
i) Deterministic FA	i) Non-Deterministic FA
ii) Only one transition for a i/p symbol	ii) Any no of transitions
iii) Cannot use ϵ-transition	iii) Can use ϵ-transition i.e. Empty string transition
iv) DFA can be understood as one machine	iv) NFA can be understood as multiple little machines computing at same time.
v) Next possible state is set	v) ~~Next~~ Each pair of states & i/p symbols have many possible next states.
vi) difficult to construct	vi) easier to construct
vii) All DFA are NFA	vii) Not all NFA are DFA.
viii) Requires more space	viii) Requires less space than DFA.
ix) Dead state may be required.	ix) Dead state may not be required.
x) Back tracking allowed	x) Back tracking not always possible
xi) Conversion of RE to DFA is difficult	xi) simpler.
xii) $\delta : Q \times \Sigma \rightarrow Q$	xii) $Q \times \Sigma \rightarrow 2^Q$

2) a) $L = \{00, 0011, 0000, 1111, \ldots\}$



Transition Table

δ	0	1
→ (ee)	oe	eo
oe	ee	oo
oo	eo	oe
eo	oo	ee

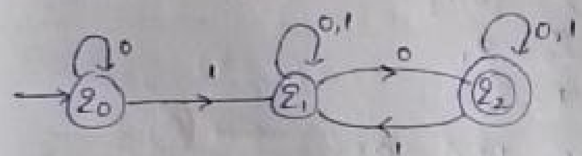
Given string '110101'

$\rightarrow (ee) \xrightarrow{1} (eo) \xrightarrow{1} (ee) \xrightarrow{0} (oe) \xrightarrow{1} (oo) \xrightarrow{0} (eo) \xrightarrow{1} (ee)$

b)

Moore	Mealy
i) o/p depends only on present state	i) o/p depends on both present state & present i/p.
ii) If i/p changes, o/p does not change	ii) If i/p changes, o/p also changes.
iii) More states required	iii) less states required.
iv) React slower to i/p's	iv) React faster to i/p's.
v) o/p is placed on states	v) o/p is placed on transitions.
vi) Easy to design	vi) Difficult to design
vii) Synchronous o/p & state generation	vii) Asynchronous o/p generation.
viii) less H/w requirement for circuit implementation	viii) More ..

3) b)



Given NFA $M = (Q, \Sigma, \delta, q_0, F)$

where $Q = (q_0, q_1, q_2)$

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

$\delta = Q \times \Sigma \rightarrow 2^Q$

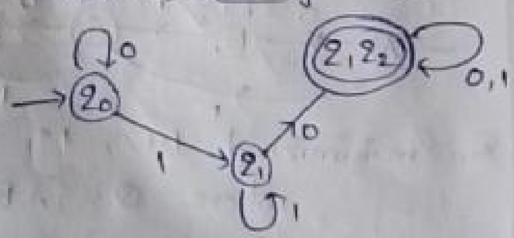
$q_0 = q_0$

$F = q_2$

Now Converting NFA to DFA. The transition Table is

δ	0	1
$\rightarrow q_0$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1, q_2]$	$[q_1]$
$*[q_1 q_2]$	$[q_1 q_2]$	$[q_1 q_2]$

Transition Diagram is



a) Minimal DFA:

DFA which is equivalent to original DFA but with reduced no of states.

Given, TT is

δ	0	1
$\rightarrow q_1$	q_2	q_3
q_2	q_4	q_5
q_3	q_6	q_7
q_4	q_4	q_5
q_5	q_6	q_7
$* q_6$	q_4	q_5
q_7	q_6	q_7.

Given DFA, $M = (Q, \delta, \Sigma, q_0, F)$

where $Q = \{q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$

$\delta = Q \times \Sigma \rightarrow Q$

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

$q_0 = q_1$

$F = q_6$

Minimize the given DFA as

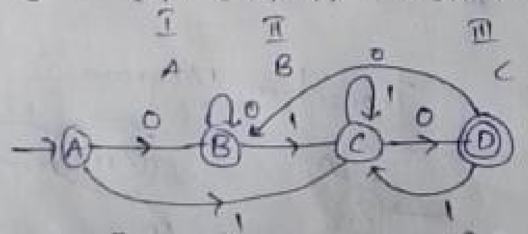
$\pi_0 = (q_1, q_2, q_3, q_4, q_5, q_6, q_7)$

$\pi_1 = \underset{\text{I}}{(q_1, q_2, q_3, q_4, q_5, q_7)}\ \underset{\text{II}}{(q_6)}$

$\pi_2 = \underset{\text{I}}{(q_1, q_2, q_4)}\ \underset{\text{II}}{(q_3, q_5, q_7)}\ \underset{\text{III}}{(q_6)}$

$\pi_3 = \underset{\text{I}}{(q_1)}\ \underset{\text{II}}{(q_2, q_4)}\ \underset{\text{III}}{(q_3, q_5, q_7)}\ \underset{\text{IV}}{(q_6)}$

A B C D



$\therefore$ The minimized DFA, $M = \{Q, \Sigma, \delta, q_0, F\}$

where $Q = \{A, B, C, D\}$

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

$q_0 = A$

$F = D$

$\delta = Q \times \Sigma \rightarrow Q$.

a) NFA-ϵ:-

Without reading any input symbol we can move on jump from one state to another by using NFA-ϵ Transition.

Given

NFA-ϵ $M = \{Q, \Sigma, \delta, q_0, F\}$

where $Q = \{0, 1, 2, 3\}$

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

$q_0 = 0$

$F = 1$

$\delta = Q \times \Sigma \rightarrow 2^Q$

Now, NFA-ϵ transition to NFA without ϵ transition. First we need to findout ϵ-closure for each & every state and find out new transition function for each & every state by using given i/p alphabets.

ϵ-closure $(0) = \{0, 1\}$

ϵ-closure $(1) = \{1\}$

ϵ-closure $(2) = \{2\}$

ϵ-closure $(3) = \{3, 1\}$

W.K.T, $\hat{\delta}(q_0, \epsilon) = \epsilon\text{-closure}(q_0)$

$\hat{\delta}(q_0, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), a))$

$\delta(0, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(0, \epsilon), a))$

$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(0), a))$

$= \epsilon\text{-closure}(\delta(\{0, 1\}, a))$

$= \epsilon\text{-closure}(\delta(0, a) \cup \delta(1, a))$

$= \epsilon\text{-closure}(1, 2)$

$= \{1, 2\}$

$\delta(0, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(0, \epsilon), b))$

$= \epsilon\text{-closure}(\delta(0, b) \cup \delta(1, b))$

$= \emptyset$

$\delta(1,a) = \epsilon\text{-closure}(\delta(\hat{\delta}(1,\epsilon),a))$

$= \epsilon\text{-closure}(\delta(1,a))$

$= \epsilon\text{-closure}(\{1,2\})$

$= \{1,2\}$

$\delta(1,b) = \epsilon\text{-closure}(\delta(\hat{\delta}(1,\epsilon),b))$

$= \epsilon\text{-closure}(\delta(1,b))$

$= \epsilon\text{-closure}(\phi)$

$= \phi$

$\delta(2,a) = \epsilon\text{-closure}(\delta(\hat{\delta}(2,\epsilon),a))$

$= \epsilon\text{-closure}(\delta(2,a))$

$= \phi$

$\delta(2,b) = \epsilon\text{-closure}(\delta(\hat{\delta}(2,\epsilon),b))$

$= \epsilon\text{-closure}(\delta(2,b))$

$= \epsilon\text{-closure}(3)$

$= \{1,3\}$

$\delta(3,a) = \epsilon\text{-closure}(\delta(\hat{\delta}(3,\epsilon),a))$

$= \epsilon\text{-closure}(\delta(3,a) \cup \delta(1,a))$

$= \epsilon\text{-closure}(\{1,2\})$

$= \epsilon\text{-closure}(1) \cup \epsilon\text{-closure}\{2\}$

$= \{1,2\}$

$\delta(3,b) = \epsilon\text{-closure}(\delta(\hat{\delta}(3,\epsilon),b))$

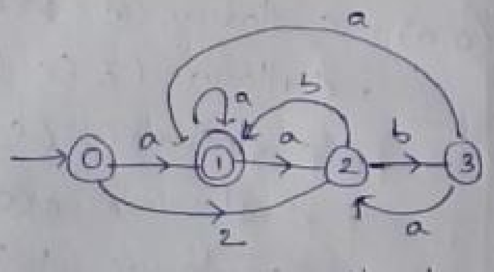
$= \epsilon\text{-closure}(\delta(3,b) \cup \delta(1,b))$

$= \epsilon\text{-closure}(\phi)$

$= \phi$

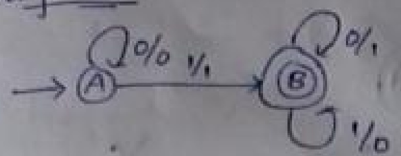
Transition Table is

δ	a	b
0	1, 2	ϕ
1	1, 2	ϕ
2	ϕ	1, 3
3	1, 2	ϕ

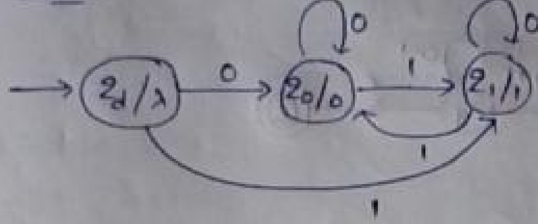


NFA without ϵ

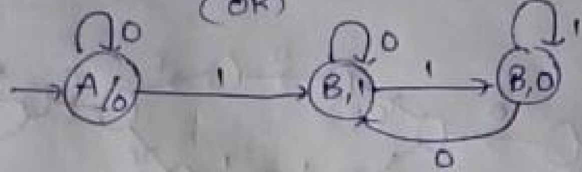
b) Mealy Machine :



Moore machine



(OR)



5) a) Regular Expression:

The language accepted by finite automata can be easily described by simple expression. called Regular expressions.

→ A RE can also be described as a sequence of pattern that defines a string.

i) String over $\Sigma = \{a, b\}$ ending with ab

$$RE = \Sigma^* ab$$
$$= (a+b)^* ab$$

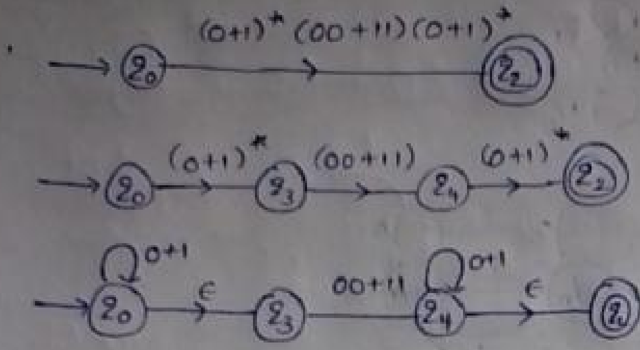
ii) string over $\Sigma = \{0,1\}$ containing 10

$$RE = \Sigma^* 10 \Sigma^*$$
$$= (0+1)^* 10 (0+1)^*$$

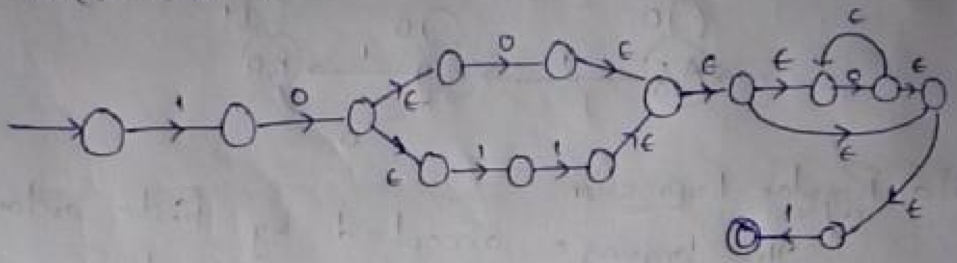
iii) string over $\Sigma = \{0,1\}$ that contain 1 in 3rd position from right end

$$RE = \Sigma^* 1 (0+1)(0+1)$$
$$= (0+1)^* 1 (0+1)(0+1)$$

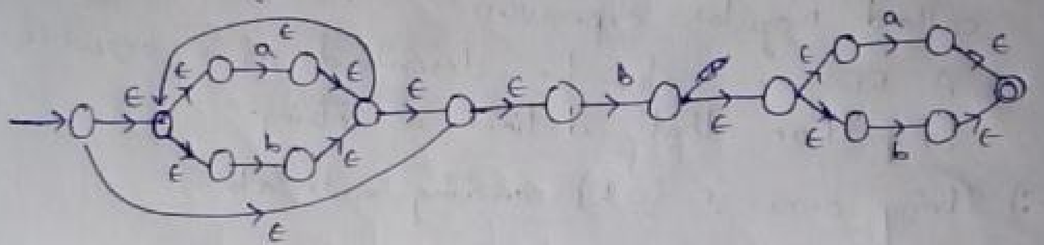
b) $(0+1)^*(00+11)(0+1)^*$



7)a) i) $10(0+11)0^*1$



ii) $(a+b)^* b(a+b)$



b) Pumping Lemma:-

Pumping lema is used to prove that set of languages (or) certainsets are not regular.

Alg.

~~ste~~ for any regular language L, there exists an integer n, such that for all $x \in L$ with $|x| \geq n$, there exists $u,v,w \in \Sigma^*$ such that $x=uvw$ and

1) $|uv| \leq n$
2) $|v| \geq 1$
3) for all $i \geq 0$, $uv^iw \in L$.

step-1:- Assume that (L) is a R.L , Let n is the no of states in L

2) choose a string w , such that $|W| \geq n$ by using PL to write $W = xyz$

i) $|Y| > 0$

ii) $|XY| \leq n$

3) find a suitable integer (i) such that xy^iz does not belongs to L . Hence the above language is not regular.

5) a) i) $\Sigma = \{a, b, c\}$

$$RE = \Sigma^* ab \Sigma^* = (a+b+c)^* ab(a+b+c)^*$$

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

$$RE = (01)^+ + (10)^+ + 0(10)^* + 1(01)^*$$

iii)

b) Identity Rules :

Let P, Q and R be the RE then the identity rules are

i) $\epsilon R = R\epsilon = R$

ii) $\epsilon^* = \epsilon$ ϵ is null string

iii) $(\phi)^* = \epsilon$ ϕ is empty string

iv) $\phi R = R\phi = \phi$

v) $\phi + R = R$

vi) $R + R = R$

vii) $RR^* = R^*R = R^+$

viii) $(R^*)^* = R^*$

ix) $\epsilon + RR^* = R^*$

x) $(P+Q)R = PR + QR$

xi) $(P+Q)^* = (P^*Q^*)^* = (P^* + Q^*)^*$

xii) $R^*(\epsilon + R) = (\epsilon + R)R^* = R^*$

xiii) $(R + \epsilon)^* = R^*$

xiv) $\epsilon + R^* = R^*$

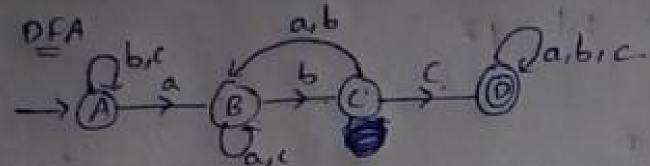
xv) $(PQ)^*P = P(QP)^*$

xvi) $R^*R + R = R^*R$

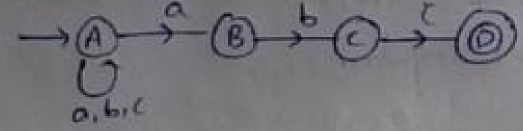
9) a) i) NFA

abbabc

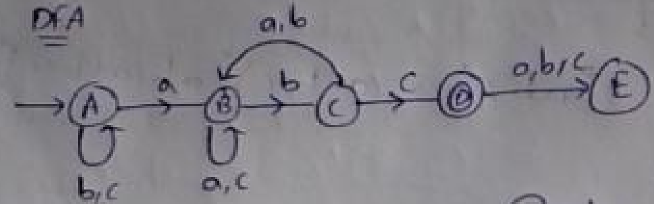
DFA



ii) NFA

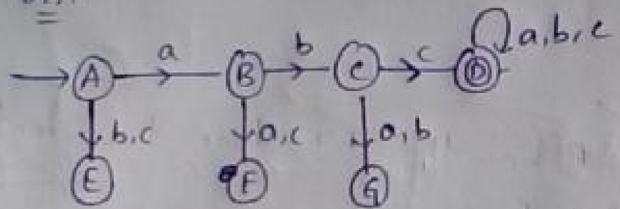


DFA

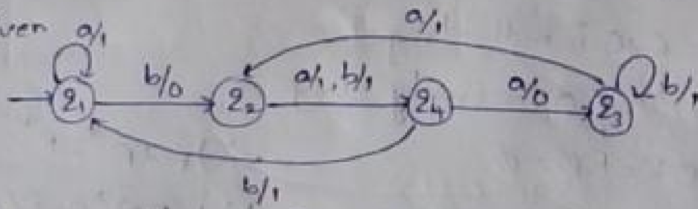


iii) NFA:

DFA



b) Given



The Mealy machine is given as

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

where $Q = \{q_1, q_2, q_3, q_4\}$

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

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

$\delta = Q \times \Sigma \to Q$

$q_0 = q_1$

$\lambda = Q \to \Delta$

Now Converting Mealy machine to moore

$M_0 = (Q \times \Delta, \Sigma, \Delta, \delta', \lambda'(q_1, 1))$

$Q' = Q \times \Delta = (q_1, q_2, q_3, q_4) \times (0, 1)$

$Q' = \{(q_1,0)(q_1,1)(q_2,0)(q_2,1)(q_3,0)(q_3,1)(q_4,0)(q_4,1)\}$

W.K.T $\delta'((q_0, b), a) = (\delta(q_0, a), \lambda(q_0, a))$

$\cancel{\delta'((q_1,a),0) = (\delta(q_1,0), \lambda(q_1,a))}$

$=$

$\delta'((q_1,0),a) = (\delta(q_1,a), \lambda(q_1,a))$

$= (q_1, 1)$

$\delta'((q_1,0),b) = (\delta(q_1,b), \lambda(q_1,b))$

$= (q_2, 0)$

$\delta'((q_1,1),a) = \delta(q_1,a), \lambda(q_1,a)$

$= (q_1, 1)$

$\delta'((q_1,1),b) = (\delta(q_1,b), \lambda(q_1,b))$

$= (q_2, 0)$

$\delta'((q_2,0),a) = (\delta(q_2,a), \lambda(q_2,a))$

$= (q_4, 1)$

$\delta'((q_2,0),b) = (\delta(q_2,b), \lambda(q_2,b))$

$= (q_4, 1)$

$\delta'((q_2,1),a) = (\delta(q_2,a), \lambda(q_2,a))$

$= (q_4, 1)$

$\delta'((q_2,1),b) = (\delta(q_2,a), \lambda(q_2,a))$

$= (q_4, 1)$

$\delta'((q_3,0),a) = (\delta(q_3,a), \lambda(q_3,a))$

$= (q_2, 1)$

$\delta'((q_3,0),b) = (\delta(q_3,b), \lambda(q_3,b))$

$= (q_3, 1)$

$$\delta'((q_3,1),a) = (\delta(q_3,a), \lambda(q_3,a))$$
$$= (q_2,1)$$
$$\delta'((q_3,1),b) = (\delta(q_3,b), \lambda(q_3,b))$$
$$= (q_3,1)$$
$$\delta'((q_4,0),a) = (\delta(q_4,a), \lambda(q_4,a))$$
$$= (q_3,0)$$
$$\delta'((q_4,0),b) = (\delta(q_4,b), \lambda(q_4,b))$$
$$= (q_1,1)$$
$$\delta'((q_4,1),a) = (\delta(q_4,a), \lambda(q_4,a))$$
$$= (q_3,0)$$
$$\delta'((q_4,1),b) = (\delta(q_4,b), \lambda(q_4,b))$$
$$= (q_1,1)$$

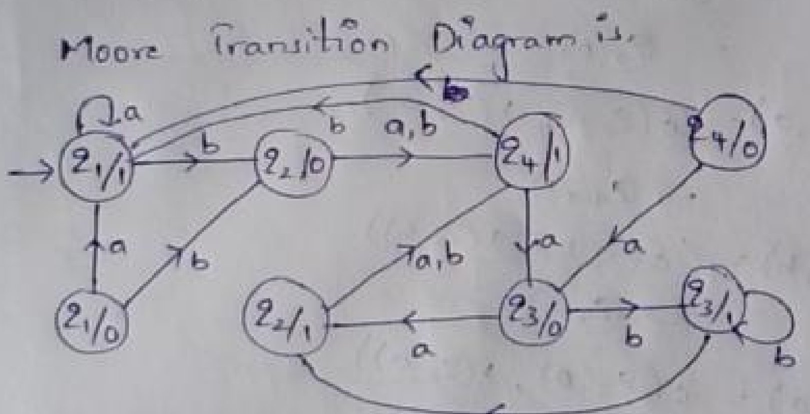
$\lambda'(q_1,0)=0$ $\quad\lambda'(q_3,0)=0$

$\lambda'(q_1,1)=1$ $\quad\lambda'(q_3,1)=1$

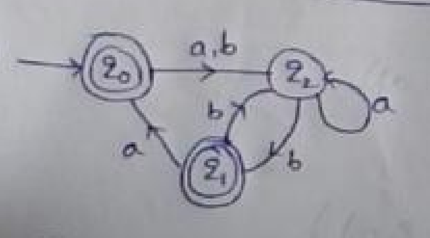
$\lambda'(q_2,0)=0$ $\quad\lambda'(q_4,0)=0$

$\lambda'(q_2,1)=1$ $\quad\lambda'(q_4,1)=1$

Moore Transition Diagram is.



6) a)



$q_0 = \epsilon + q_1 a$ — (1)

$q_1 = q_2 b$ — (2)

$q_2 = q_1 b + q_2 a + q_0 a + q_0 b$ — (3)

From (3);

$q_2 = q_2 bb + q_2 a + q_0(a+b)$

$q_2 = q_2(bb+a) + q_0(a+b)$

$q_2 = q_0(a+b)(bb+a)^*$ — (4)

substitute q_2 in ②

$$q_1 = q_0 (a+b)(bb+a)^* b \quad ⑤$$

substitute ⑤ in ①

$$q_0 = \epsilon + q_0 (a+b)(bb+a)^* ba$$

$$q_0 = q_0 (a+b)(bb+a)^* ba + \epsilon$$

$$\boxed{q_0 = ((a+b)(bb+a)^* ba)^*}$$

substitute q_0 in ⑤

$$\boxed{q_1 = ((a+b)(bb+a)^* ba)^* (a+b)(bb+a)^* b}$$

$$\boxed{q_2 = ((a+b)(bb+a)^* ba)^* (a+b)(bb+a)^*}$$

b) Closure Properties of RL:-

closure properties on regular languages are defined as certain operations on regular language which are guaranteed to produce R.L.

→ Consider L and M are R.L

i) ~~If~~ ~~&~~ The union $L \cup M$ is also regular.

ii) The intersection $L \cap M$ is also regular

iii) Their concatenation LM is also regular

iv) Its Kleen closure L^* will also be regular.

v) If $L(G)$ is a regular language, its complement $L'(G)$ will also be regular. Complement of a language can be found by subtracting strings which are in $L(G)$ from all possible strings.

vi) +ve closure ~~of~~ L^+ will also be regular.

11) b)

DPDA	NPDA
i) less powerful than NPDA	i) More powerful than DPDA
ii) possible to convert every DPDA to a corresponding NPDA	ii) not possible to convert every NPDA into DPDA
iii) Language accepted by DPDA is subset of language accepted by NPDA	iii) not
iv) Language accepted by DPDA is called DCFL	iv) NCFL
v) for every i/p with the current state, there is only one move	v) " we can have multiple moves.