classmate
Date
Page 1.1

01 Basic Concept & Finite Automata

Automata And Grammer

Automata.

An automation is an abstract model of digital computer.
An automation has a mechanism to read input from input tape.
Automata are basically language acceptors or language recognizers.
The theory of automata starts with finite automata. Basically there are two types of language : Regular language
Non Regular language.

"Finite Automata is a mathematical model which always accept regular languages."

Grammer

Grammer is nothing but the set of rules used to defined the language
these rules can be written with the help of terminal and nonterminal
Eg.
G = S → aS | 1S 0,1 terminal
S → 0 | 1 S is Non terminal.
L = { 0, 1, 11, 00, 01, 10..... }

Alphabets

An alphabet is a finite collection of symbols.

Eg $S = \{a, b, c \ldots z\}$

$W = \{0, 1\}$

String

string is finite collection of symbols from alphabet.

Eg. $\Sigma = \{a, b\}$ then string are only combination of a, b. like $\{a, b, aa, bb, ab, ba \ldots\}$

Language

The language is a collection of appropriate string

The language is defined using an input set. The input set is typically denoted by letter $\Sigma = \{0, 1\}$ i= $\{\epsilon, 0, 00, 000 \ldots\}$

Here the language L is defined as any number of of zero's

classmate
Date
Page 13

Application of Automata Theory

1. Automata theory is the base for the formal language and these formal language are useful of the programming language.

2. Automata theory plays an important role in compiler design

3. To prove the correctness of the program automata theory is used.

4. In switching theory and design and analysis of digital circuits automata theory in applied.

5. Automata theory deals with the design finite state machine.

Finite Automata (FA)

A finite Automata is a collection of 5-tuples $(Q, \Sigma, \delta, q_0, F)$ where

Q - is a finite set of state. which is Non-empty

Σ - input alphabet / input set.

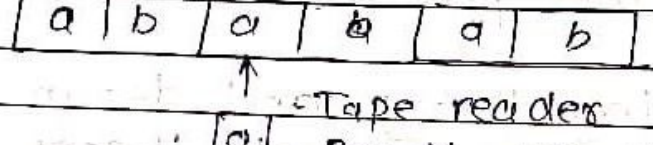
q_0 - initial state. $q_0 \in Q$.

F - is a final state.

δ - is a transition function.

Finite state Machine.

The finite automata can be represented as:



Def FSM = $(Q, \Sigma, \delta, q_0, F)$

Q = Set of state

Σ = input alphabet

δ = transition function $(Q \times \Sigma)$

q_0 = initial state.

F = Final state.

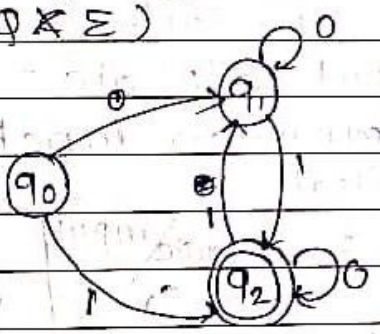
Q.1 Design FSM which accepts odd number of 1's and any number of 0's

Def FSM = $(Q, \Sigma, \delta, q_0, F)$

Q is set of state = $\{q_0, q_1, q_2\}$
Σ input alphabet = $\Sigma = \{0, 1\}$
δ = transition fun $(Q \times \Sigma)$

Q \ Σ	0	1
→ q_0	q_1	q_2
Even no of 1's q_1	q_1	q_2
odd no of 1's q_2*	q_2	q_1

Transition table



Transition diagram

q_0 = initial state = q_0
F final state = q_2

Validation.

$\vdash q_0(01110)$
$\vdash q_1(1110)$
$\vdash q_2(110)$
$\vdash q_2(10)$
$\vdash q_2(0)$
$\vdash q_2$ Accepted.

$\vdash q_0(1010)$
$\vdash q_2(010)$
$\vdash q_2(10)$
$\vdash q_1(0)$
$\vdash q_1$ Reject

Dec-19 10M.

Design FSM for language which accept $L(R) = \{ w \mid w$ string starts with 0's and followed by odd length and start with 1's followed by even length.

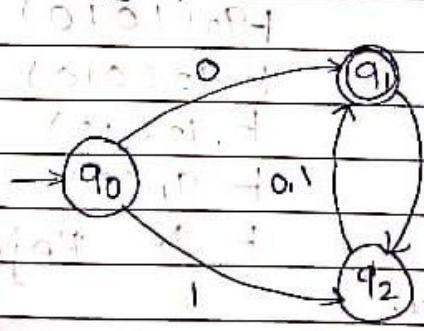
Def $FSMM = (Q, \Sigma, \delta, q_0, F)$

Q = set of state = $\{q_0, q_1, q_2\}$
Σ = input alphate = $\{0, 1\}$
δ = Transition function $Q \times \Sigma$
q_0 - initial

$Q \times \Sigma$ =

state / input	0	1
→ q_0	q_1	q_2
Even start 0's q_1*	q_2	q_2
odd start 1's q_2	q_1	q_2

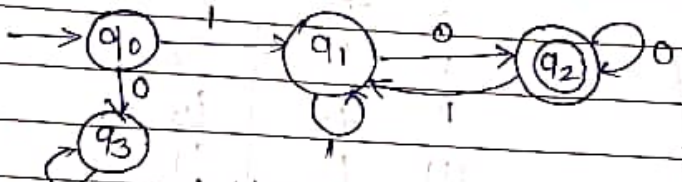
Transition diagram



$\vdash q_0(010)$
$\vdash q_1(10)$
$\vdash q_2(0)$
q_1 Accept

$\vdash q_0(1010)$
$\vdash q_2(010)$
$\vdash q_1(10)$
$\vdash q_2(0)$
$\vdash q_1$ Accept

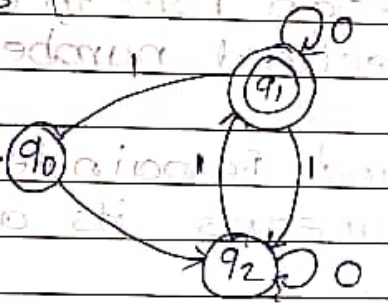
Design FSM which start with 1 and end with 0



	Sta \ I/P	0	1
	→ q_0	q_1	q_3
1	q_1	q_2	q_1
10	q_2	q_2	q_1
	q_3	q_3	q_3
	Deadsta		

Design FSM which accepts odd number of 1's and any number of 0's

	state \ I/P	0	1
	→ q_0	q_2	q_1
Even	is q_1*	q_1	q_2
odd	q_2	q_2	q_1



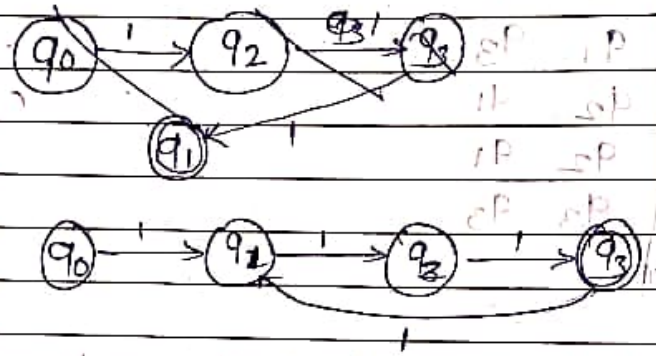
Design FSM which checks whether a unary number is divisible by '3'

The unary number made up of on's
Eg 5 is → 11111 , 3 is → 111
when we divisibity checking possible outcome is 0, 1, 2 for 3 0 means completely divisible

St \ IP	0	1

	0	1
→ q_0	q_2	q_1
q_1	q_2	q_3
q_2	q_3	q_1
* q_3	q_1	

→ q_0	q_1
1 q_1	q_2
11 q_2	q_3
111 q_3	q_1



Q. Design FSM to check whether a given decimal number is divisible by 3'

Find Remainders of 0, 1, 2

0 means its a divisible.

Def FSM $M = (Q, \Sigma, \delta, q_0, F)$

Q = set of state. = $\{q_0, q_1, q_2, q_3\}$

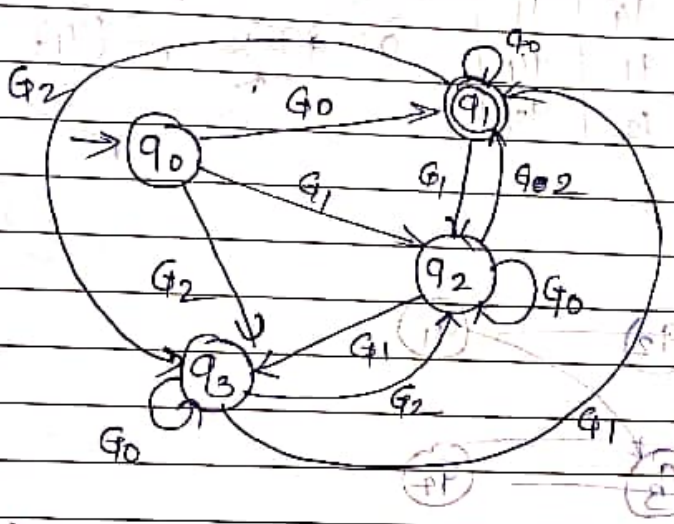
Σ = input alaphabets = $\{0,1,2,3,4,5,6,7,8,9\}$

δ = Transition function $Q \times \Sigma$

q_0 = initial state = q_0

F = final state = q_1

	G_0 0,3,6,9	G_1 1,4,7	G_2 2,5,8
$R \to q_0$	q_1	q_2	q_3
0 q_1*	q_1	q_2	q_3
1 q_2	q_2	q_3	q_1
2 q_3	q_3	q_1	q_2



$q_0(121)$
$q_2(21)$
$q_1(1)$
$q_2(1)$ which is state remainder 1

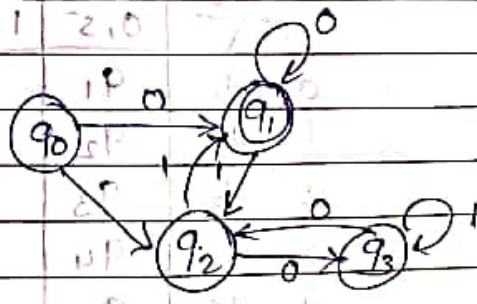
Q. Design FSM which checks whether a given binary number is divisible by three.

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

$Q =$

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

	Σ	0	1
	q_0	q_0	q_1
→	q_1*	q_2	q_3
00	q_2	q_1	q_2
01	q_3	q_3	q_1
10		q_2	q_3

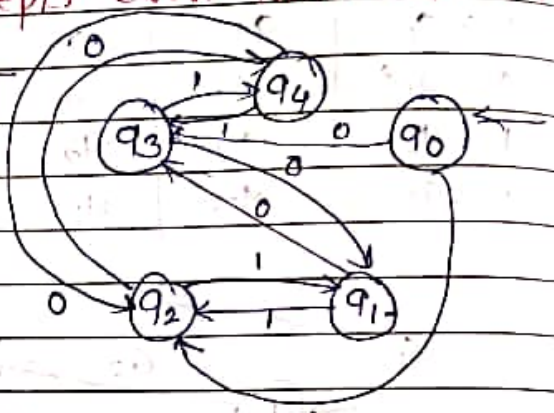


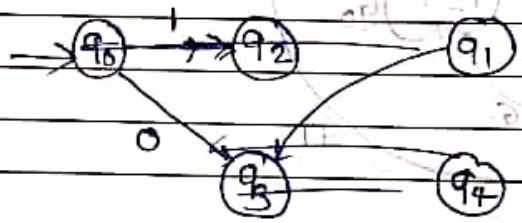
$q_0(1001)$ $q_0(0101)$
$q_2(001)$ $q_0(111)$
$q_3(01)$ $q_2(11) \Rightarrow q_1(1) = q_2$ Reject
$q_2(1) \Rightarrow q_1$ Accept Remainder 1

Date
Page 10

Design FSM which accepts even number of 0's even number of 1's

		Σ	0	1
		q_0	q_3	q_2
0's	1's			
E	E	q_1*	q_3	q_2
E	O	q_2	q_4	q_1
O	E	q_3	q_1	q_4
O	O	q_4	q_2	q_3





Design FSM decimal Number divisible by '5'

Sam

R	Σ	0,5	1,6	2,7	3,8	4,9
0	q_1	q_1	q_2	q_3	q_4	q_5
1	q_2	q_2	q_3	q_4	q_5	q_1
2	q_3	q_3	q_4	q_5	q_1	q_2
3	q_4	q_4	q_5	q_1	q_2	q_3
4	q_5	q_5	q_1	q_2	q_3	q_4

10
01

R + Group R = No 5 = state.

Binary number divisible by 5

Q \ Σ	0	1
0	q_0	q_0	q_1
1	q_1	q_2	q_3
2	q_2	q_4	q_0
3	q_3	q_1	q_2
4	q_4	q_3	q_4

100 101 1000
110 111 1001

FSM

finite Automata (without o/p)

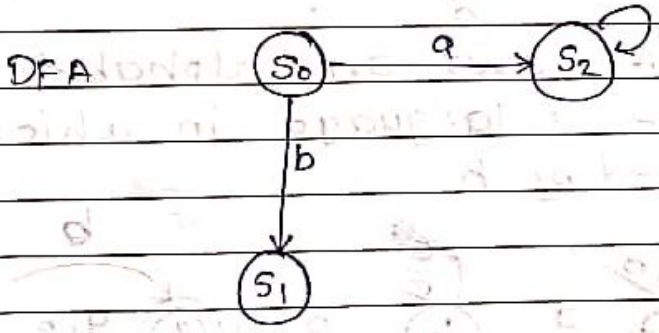
- DFA
- NFA

finite Automat (with output)

- Moore m/c
- Mealy m/c

DFA - Deterministics finite Automata.

The finite Automata is called as Deterministic finite Automata if there is only one path for specific input from current state to next state



Def DFA

$$A = (Q, \Sigma, \delta, q_0, F)$$

Q - finite set of states

Σ - finite set of input symbol

δ → Transition function

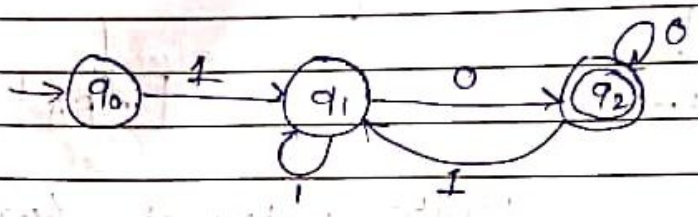
or $q_1 = \delta(S_0, b)$ or $S_2 = \delta(S_0, a)$

q_0 → initial state

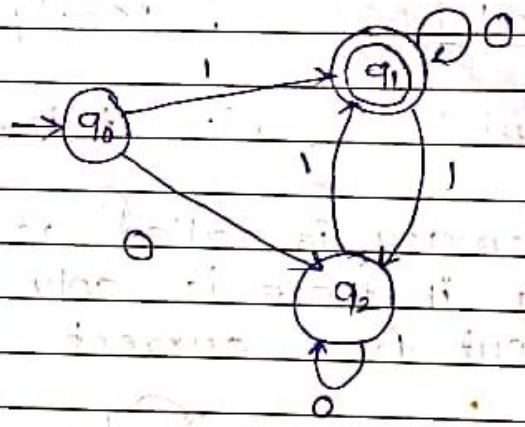
F → final state.

Date
Page 1-13

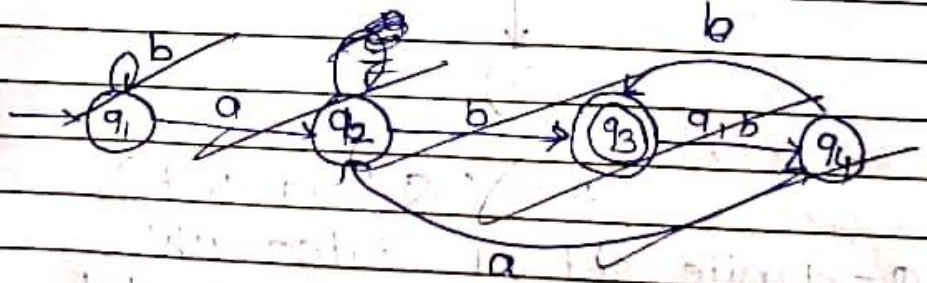
Design DFA which accepts only those strings which start with 1 ands with 0.



Design FA which accepts odd number of 1's and any number o's.

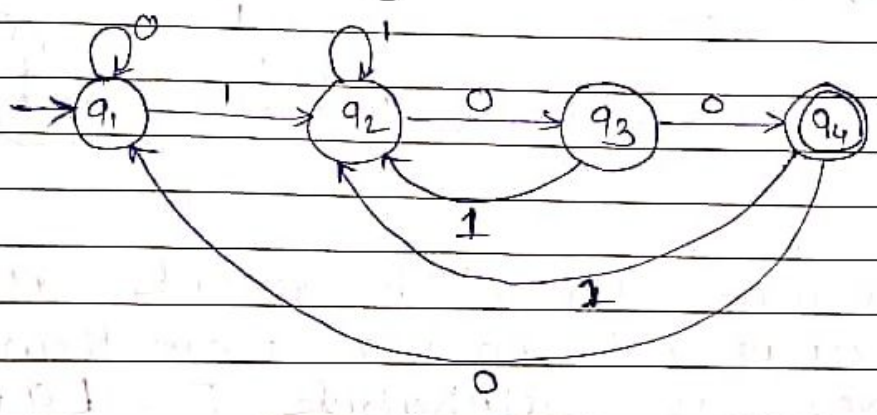


May14 10M. Design DFA over an alphabet {a,b} to recognize a language in which every a is followed by b



May 19 5M

Design a DFA accepts string of 0's and 1's ending with the string 100



May 18 10M

Construct DFA $M = \{Q, \Sigma, q_0, \delta, q_f\}$

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

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

δ = Transition function.

$q_2 = \delta(q_1, 1)$, $q_3 = \delta(q_2, 0)$, $q_4 = \delta(q_3, 0)$

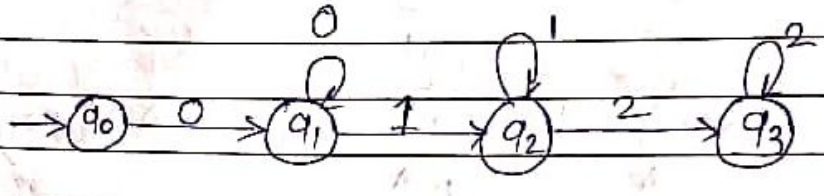
q_0 = Initial state = q_1

f = final state. = q_4

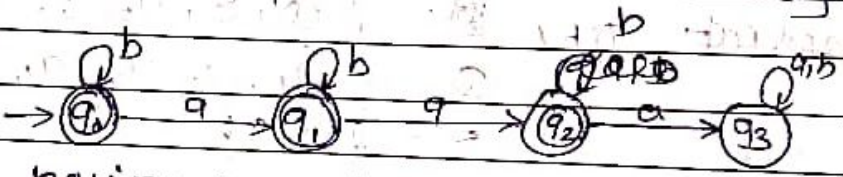
Validation

$\vdash q_1(01100)$	$\vdash q_1(11101)$	$\vdash q_1(100)$
$\vdash q_1(1100)$	$\vdash q_2(1101)$	$\vdash q_2(00)$
$\vdash q_2(100)$	$\vdash q_2(101)$	$\vdash q_3(0)$
$\vdash q_3(00)$	$\vdash q_2(01)$	$\vdash q_4$
$\vdash q_3(0)$	$\vdash q_3(1)$	Accept
$\vdash q_4$	$\vdash q_2$	
Accept	Reject	

May 18 10M. Construct the DFA that accepts the language represented by $0^*1^*2^*$

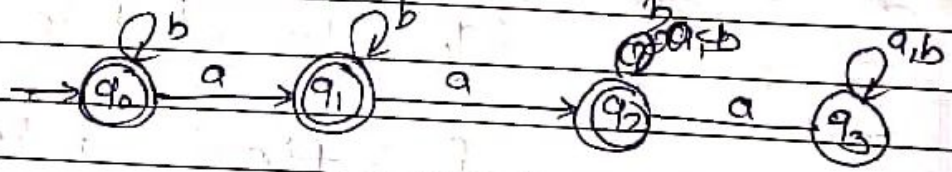


Dec 18 10M. Design DFA which accepts all the strings not having more than 2'a over the alphabets $\Sigma = \{a, b\}$

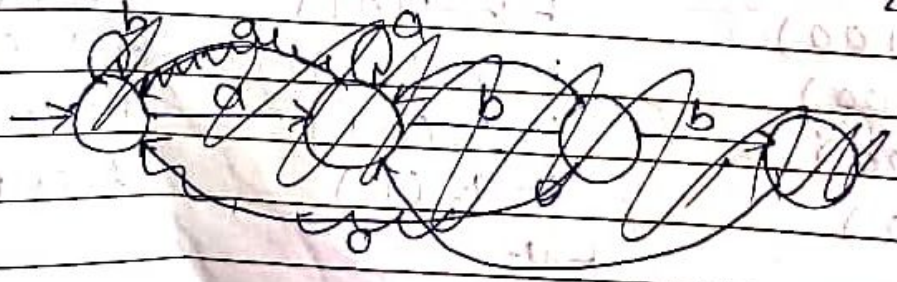


Alternate Sol^n: 1. find having more than 2'a
2. find complement of the mlc

final → Non final
Nonfinal → final

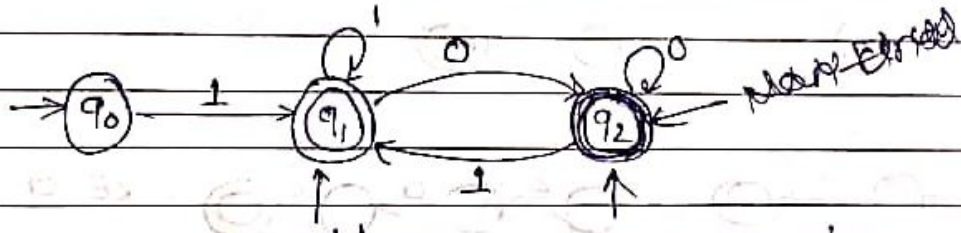


Q. Design DFA to accept string of a's & b's ending with 'abb' over the $\Sigma = \{a, b\}$



Design DFA to accept odd and even numbers represented using binary notation.

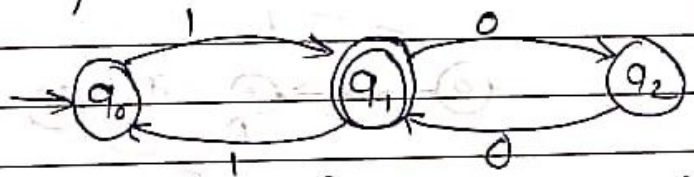
The binary number that ends with 0 is even number. and
The binary number that ends with 1 is odd number.



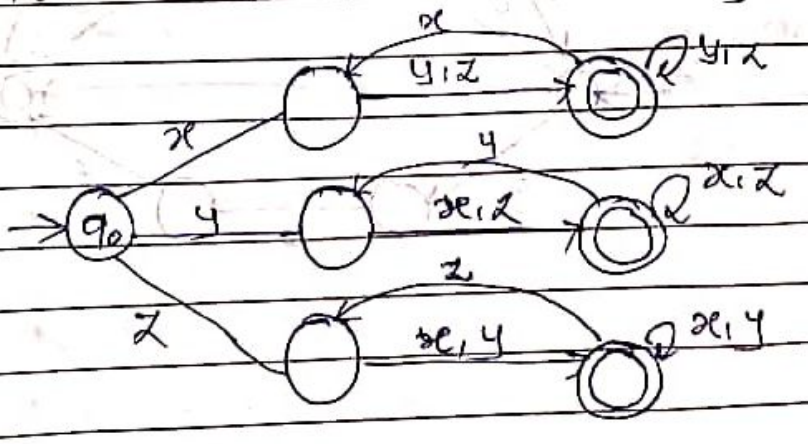
Dec 10 10M

Design DFA to accept

i) a set of all strings with odd number of 1's followed by even number of zeros



ii) a set of all strings with begin and end with different letters $\Sigma = \{x, y, z\}$



NFA - Non Deterministic Automata

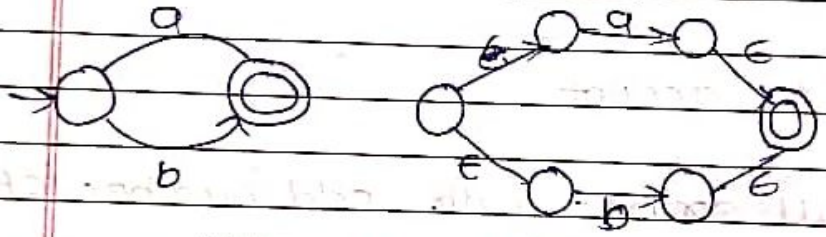
$M = (Q, \Sigma, \delta, q_0, F)$

standard NFA

DFA NFA RE. Language

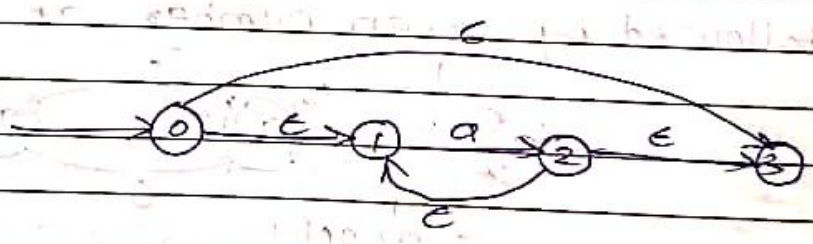
RE = ∈ {∈}

RE = a {a}

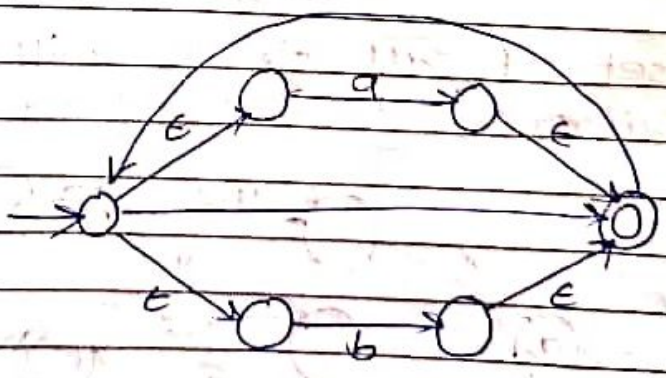


RE = a+b {a, b}

a^* =

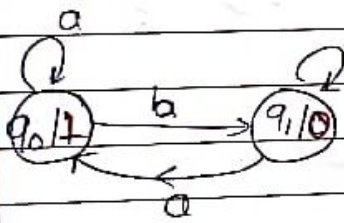


$(a+b)^*$ =



FA with o/p

moore m/c	$M = (Q, \Sigma, \delta, q_0, \Delta, \lambda)$	mealy m/c
o/p is n+1 Bit	Q = finite set of state Σ = input alphabet δ = Transition function $Q \times \Sigma \to Q$ q_0 = initial state Δ = o/p alphabet λ = o/p function.	o/p is n bit
No final state		No final state.

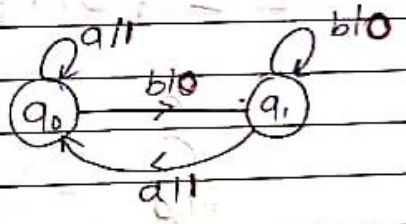


$\lambda : Q \to \Delta$

for every state o/p is associated.

means
$q_0 \to 1$
$q_1 \to 0$

Validation o/p
$\vdash q_0(ab)$ 1
$\vdash q_0(b)$ 1
$\vdash q_1$ 0
So o/p → 110



$\lambda : Q \times \Sigma \to \Delta$

op is associated. for a state for a given input there is some o/p

means
$(q_0, a) \to 1$
$(q_0, b) \to 0$
$(q_1, b) \to 0$
$(q_1, a) \to 1$

	o/p
$\vdash q_0(a,b)$	-
$\vdash q_0(b)$	1
$\vdash q_1$	0

So o/p is n bit

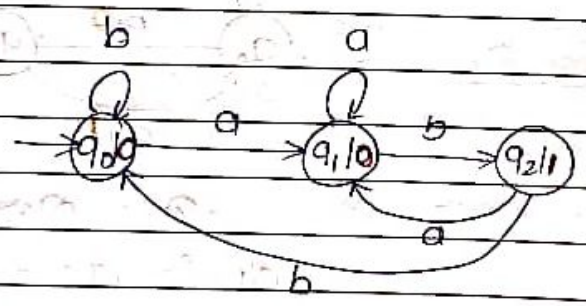
Design Moore machine ~~example~~ for counting the occurrence of substring 'ab'

construct a moore machine that takes set of all strings over the Σ $\{a, b\}$ as i/p and prints '1' as o/p for every occurance of 'ab' as a substring.

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

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

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



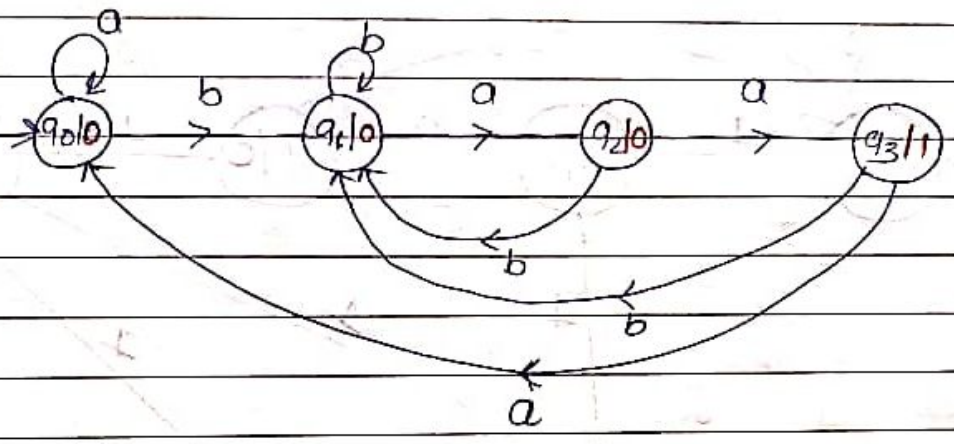
Validation

$\vdash q_0(ab)$	0	$\vdash q_0(abab)$	0
$\vdash q_1(b)$	0	$\vdash q_1(bab)$	0
$\vdash q_1$	1	$\vdash q_2(ab)$	1
		$\vdash q_1(b)$	0
so o/p 001		$\vdash q_2$ -	1
1 no of ab		so o/p 00101	
		2 no of ab	

construct a moore mlc that takes set of all string over {a,b} and counts no of occurrences of substring 'baa'.

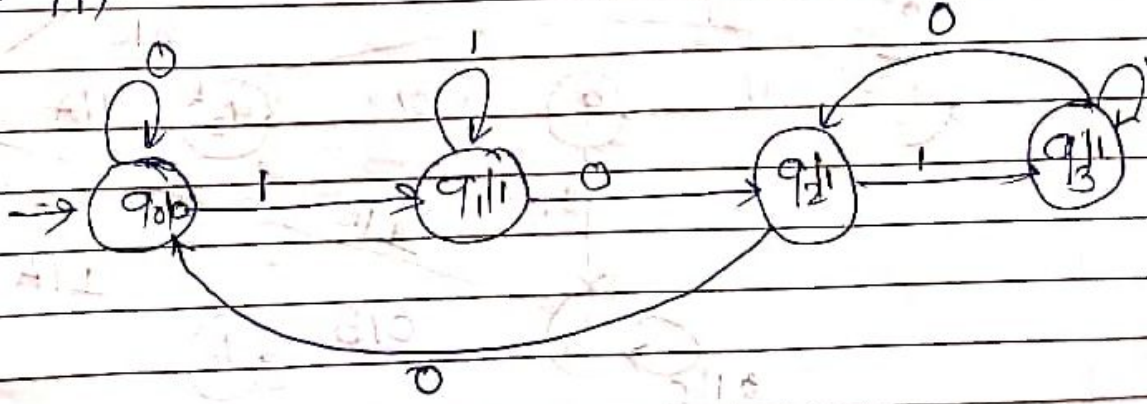
$\Sigma = \{a, b\}$

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



May-19 10M Design moore mlc for following if inputs ends in '101' then output should be A, if inputs ends '110' output should be B. otherwise output should be C and convert into mealy machine.

DEC-19 10M construct moore and mealy machine to convert each occurrence of 101 by 111

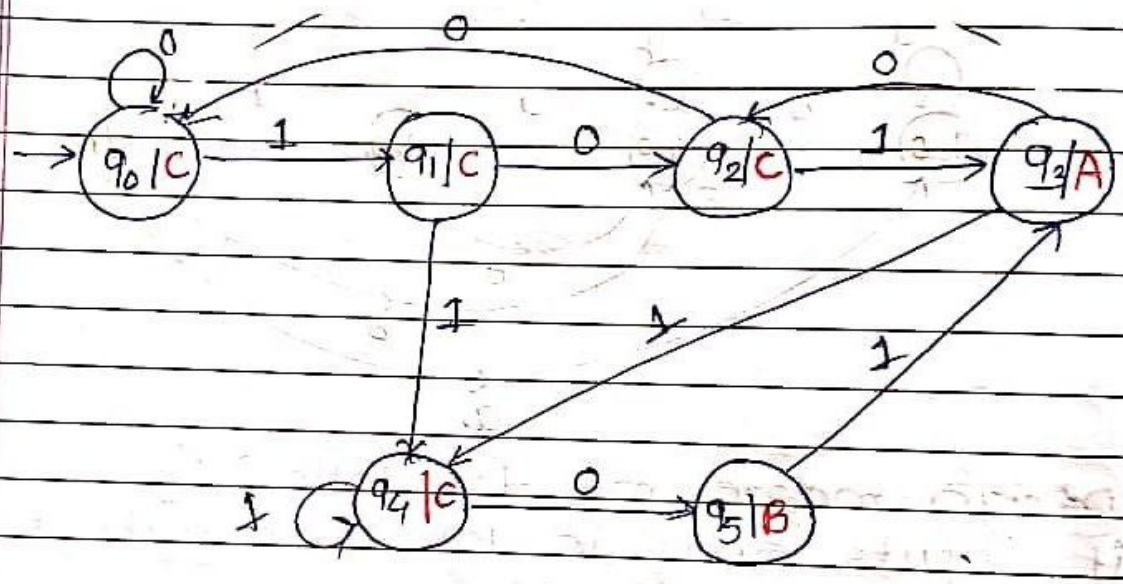


Moore Machine

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

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

$M = \{Q, \Sigma, \delta, q_0, \Delta, T\}$

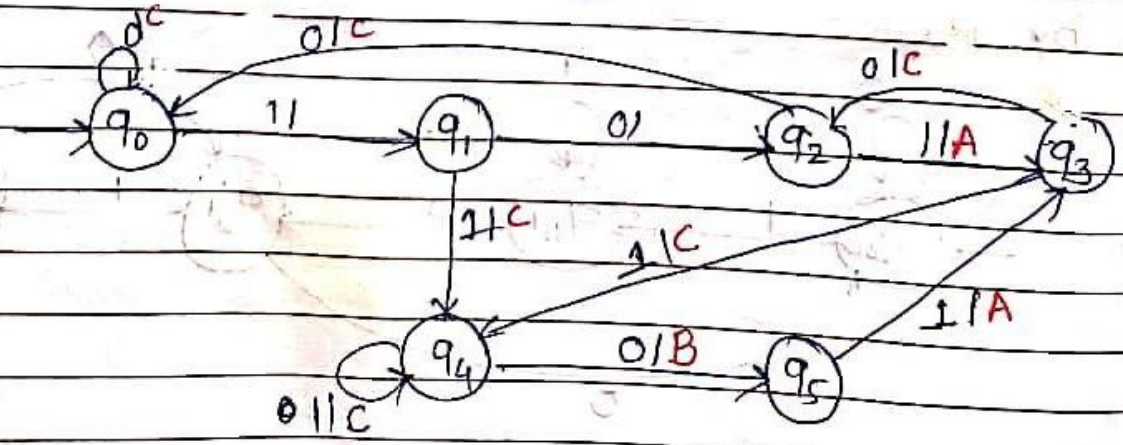


Moore m/c → mealy m/c

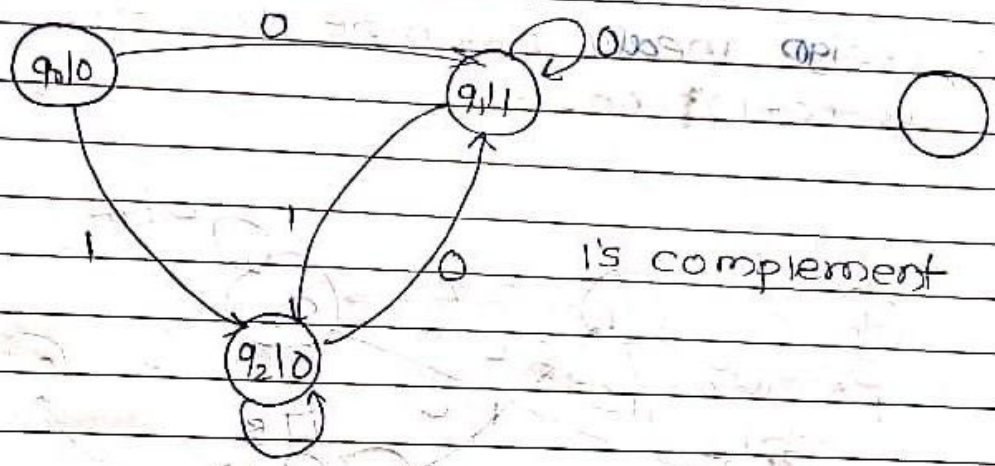
↑ o/p is associated on state

↑ o/p is associated on transition

Give the output to incoming transition



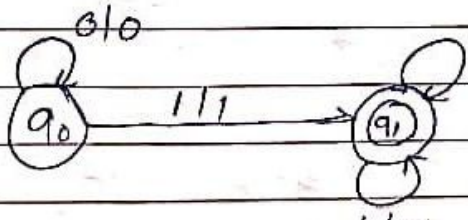
Q. convert moore and mealy machine to find out 2's complement of a binary
May-19
10M



2's complement

1's calculate 1's complement binary no 1 to 0 and 0 to 1 then add 1 to it

1. start from right to left
2. Ignore all 0's
3. when 1 comes ignore it and then take 1's complement of every digit



101111	1000
010000 ← 1's	0111
+ 1	+ 1
010001	1000

NOV 18 construct moore & mealy mealy machine to convert each occurence of 100 by 101

may 18 Design mealy machine for the language as $(0+1)^*(00+11)$