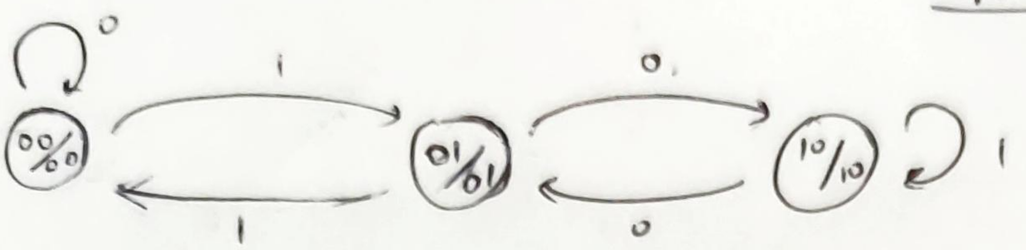
GROUP 20

Q1)



← FSM

State Table

Present States (P_0)	Present States (P_1)	Input (I)	Next State (N_0)	Next State (N_1)	Output (y_0)	Output (y_1)
0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	0	1
1	0	0	0	1	1	0
1	0	1	1	0	1	0
1	1	0	x	x	x	x
1	1	1	x	x	x	x

The state encoding denotes the remainder calculated after reading the last bit, starting from the MSB.

The Last output is the Remainder.

$$N_0 = (P_0)(I) + (P_1)(I)'$$

$$N_1 = (P_0)(I)' + (P_0)'(P_1)'(I)$$

$$y_0 = P_0$$

$$y_1 = P_1$$

QUESTION 2

We design a moore machine for this question. Two states, one for each floor are selected. As it can be seen that the outputs, i.e., the state of the lights, depends on the floor only. The input can also be encoded as a binary, up and down (1 and 0). The transitions are shown in the fsm below.

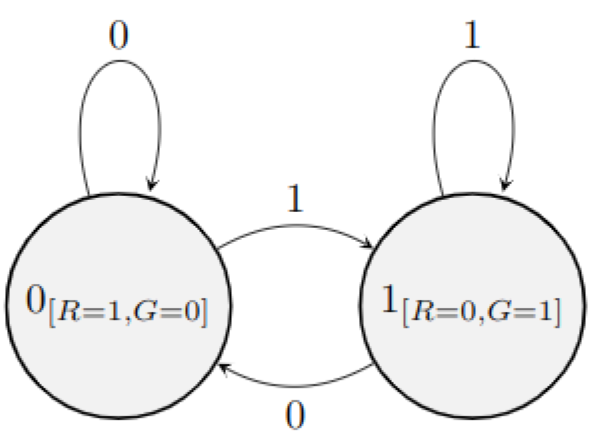
Encodings

STATE	ENCODING
Ground	0
First	1

INPUT	ENCODING
Down	0
Up	1

OUTPUT	ENCODING
Light off	0
Light on	1

FSM



Elevator FSM

Pressing up on the ground floor leads to a transition to the first floor. Pressing down on the first floor leads to a transition to the ground floor. Pressing down on the ground floor and up on the first floor has no effect and the lift stays on the same floor. Red light is on and green light is off when the elevator is on the ground floor. Red light is off and green light is on when the elevator is on the first floor.

NEXT STATE AND OUTPUT TABLE

PS	INPUT (I)	NS	OUTPUT	
			R	G
0	0	0	1	0
0	1	1	1	0
1	0	0	0	1
1	1	1	0	1

NEXT STATE AND OUTPUT FUNCTIONS

It can be observed from the table that

1. NS = I
2. R = not PS
3. G = PS

QUESTION 3

We design a moore machine for this question. 6 states, 1 initial and 5 states for determining what length of the string entered so far is correct are maintained. The output (unlock) depends only on the state and will be 0 for all states except for the state where the length of the matched string becomes 5 where it will be 1. The input can be one of 0,1 or reset. The 6 states can be encoded using 3 bits and the 3 inputs using 2 bits. The output will be binary, 0 or 1.

Encodings

STATE (MATCHED STRING)	LENGTH OF MATCHED STRING	ENCODING
Empty string (Initial)	0	000
“0”	1	001
“01”	2	010
“010”	3	011
“0101”	4	100
“01011”	5	101

INPUT	ENCODING
0	00
1	01
reset	10

OUTPUT	ENCODING
Locked	0
Unlocked	1

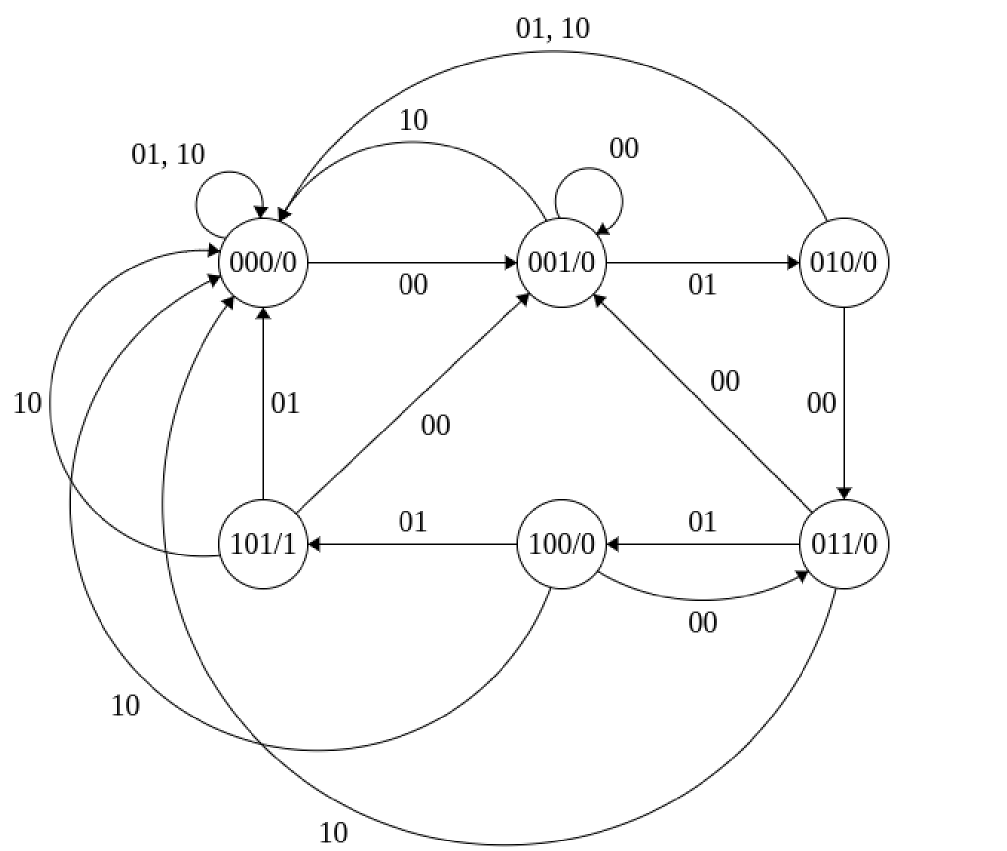
NEXT STATE AND OUTPUT TABLE

PS			INPUT		NS			OUTPUT
p0	p1	p2	i0	i1	n0	n1	n2	
0	0	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0
0	0	0	1	0	0	0	0	0
0	0	0	1	1	-	-	-	0
0	0	1	0	0	0	0	1	0
0	0	1	0	1	0	1	0	0
0	0	1	1	0	0	0	0	0
0	0	1	1	1	-	-	-	0
0	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	0
0	1	0	1	1	-	-	-	0
0	1	1	0	0	0	0	1	0
0	1	1	0	1	1	0	0	0
0	1	1	1	0	0	0	0	0
0	1	1	1	1	-	-	-	0
1	0	0	0	0	0	1	1	0
1	0	0	0	1	1	0	1	0
1	0	0	1	0	0	0	0	0
1	0	0	1	1	-	-	-	0
1	0	1	0	0	0	0	1	1
1	0	1	0	1	0	0	0	1
1	0	1	1	0	0	0	0	1

1	0	1	1	1	-	-	-	1
1	1	0	0	0	-	-	-	-
1	1	0	0	1	-	-	-	-
1	1	0	1	0	-	-	-	-
1	1	0	1	1	-	-	-	-
1	1	1	0	0	-	-	-	-
1	1	1	0	1	-	-	-	-
1	1	1	1	0	-	-	-	-
1	1	1	1	1	-	-	-	-

NEXT STATE AND OUTPUT FUNCTIONS

$$n0 = (i1 \cdot p1 \cdot p2) + (i1 \cdot p0 \cdot \overline{p2})$$
$$n1 = (\overline{i0} \cdot \overline{i1} \cdot p1 \cdot \overline{p2}) + (\overline{i0} \cdot \overline{i1} \cdot p0 \cdot \overline{p2}) + (i1 \cdot \overline{p0} \cdot \overline{p1} \cdot p2)$$
$$n2 = (\overline{i0} \cdot \overline{i1}) + (\overline{i0} \cdot p0 \cdot \overline{p2})$$
$$output = (p0 \cdot p2)$$



n0		p0,p1,p2							
		000	001	011	010	110	111	101	100
i0,i1	00	0	0	0	0	-	-	0	0
	01	0	0	1	0	-	-	0	1
	11	-	-	-	-	-	-	-	-
	10	0	0	0	0	-	-	0	0

n1		p0,p1,p2							
		000	001	011	010	110	111	101	100
i0,i1	00	0	0	0	1	-	-	0	1
	01	0	1	0	0	-	-	0	0
	11	-	-	-	-	-	-	-	-
	10	0	0	0	0	-	-	0	0

n2		p0,p1,p2							
		000	001	011	010	110	111	101	100
i0,i1	00	1	1	1	1	-	-	1	1
	01	0	0	0	0	-	-	0	1
	11	-	-	-	-	-	-	-	-
	10	0	0	0	0	-	-	0	0

output		p0,p1,p2							
		000	001	011	010	110	111	101	100
i0,i1	00	0	0	0	0	-	-	1	0
	01	0	0	0	0	-	-	1	0
	11	0	0	0	0	-	-	1	0
	10	0	0	0	0	-	-	1	0

4.) ⇒ Input format = b0 b1 b2

b0 = To denote if 5 ruppes coin is inserted
(1 if yes else 0)

b1 = To denote if 10 rupees coin is inserted
(1 if yes else 0)

b2 = To denote if 20 rupees coin is inserted
(1 if yes else 0)

⇒ Output format = b0 b1 b2

b0 = To denote if con ~~be~~ to be dispensed
(1 if yes else 0)

b1 = To denote if 5 rupees coin is to be dispensed
(1 if yes else 0)

b2 = To denote if 10 rupees coin is to be dispensed
(1 if yes else 0).

⇒ Price of each ~~coin~~ con = Rs.30.

Current State	Input = 001		Input = 010		Input = 100	
	NS	Output	NS	Output	NS	Output
0	20	000	10	000	5	000
5	25	000	15	000	10	000
10	0	100	20	000	25	000
15	0	110	25	000	20	000
20	0	101	0	100	25	000
25	0	111	0	110	0	100

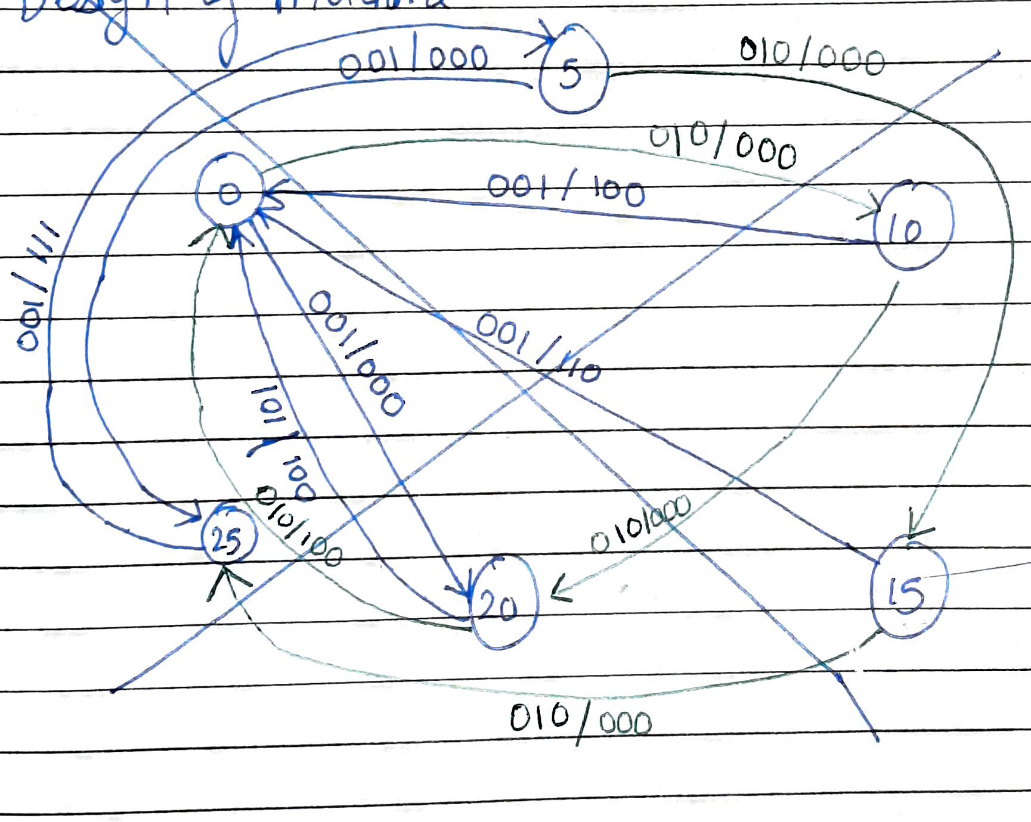
Description of machine:-

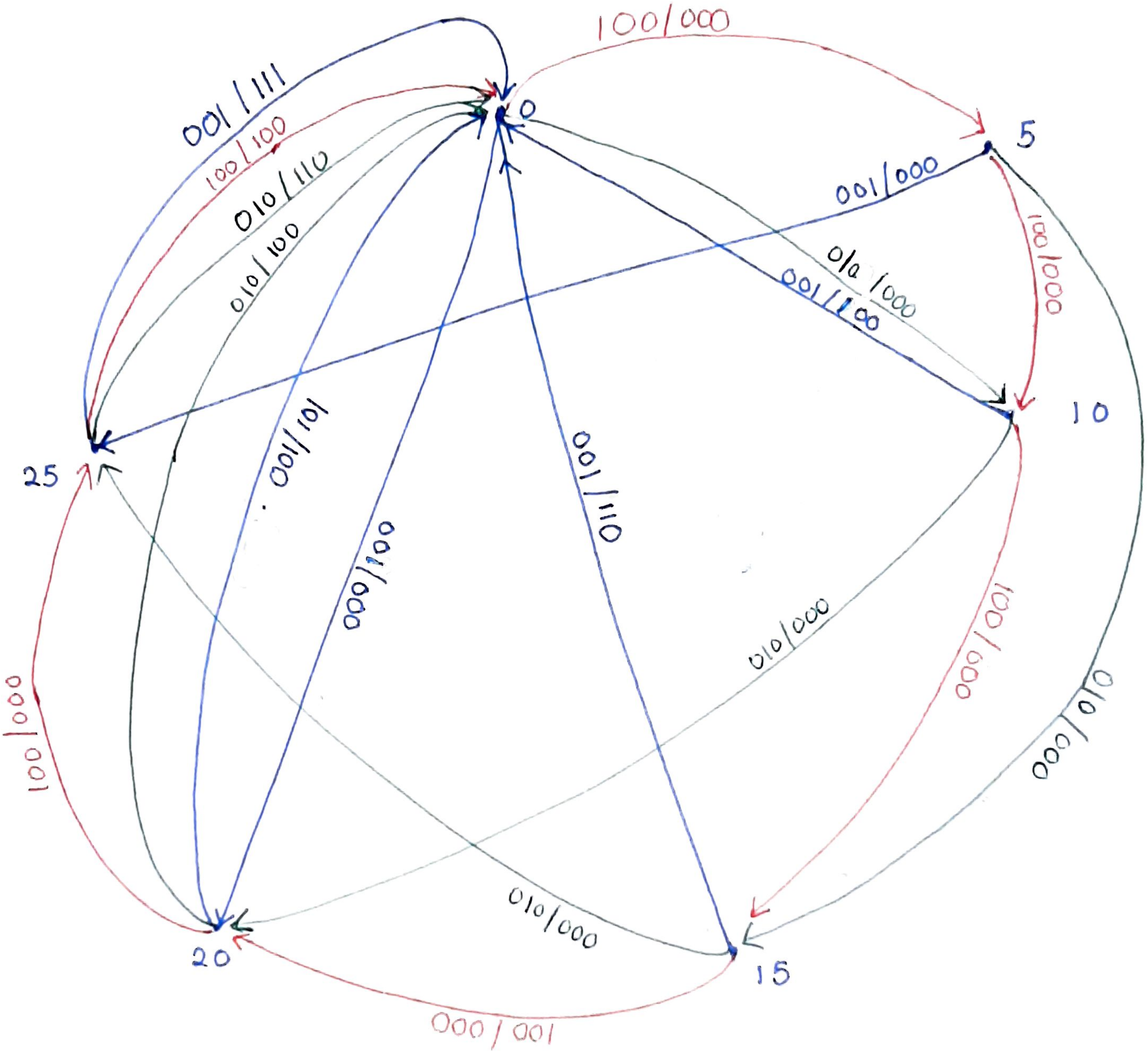
⇒ Machine has six states which signify the money present inside machine during the process.

⇒ Since, any amount ≥30, will lead of dispensing of coin and corresponding change. So, states range from 0 to 25 only with state being multiples of 5.

⇒ There are 3 possible inputs i.e. 100, 010, 101

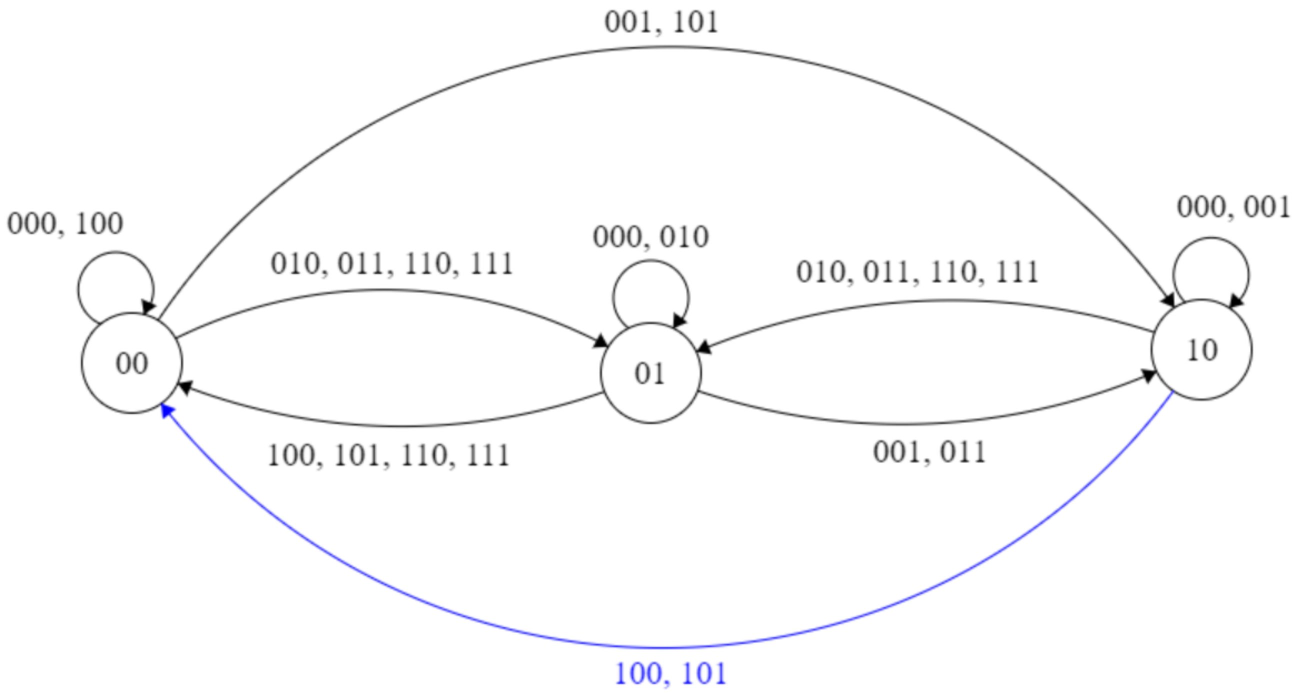
Design of machine:-




0
5
10
15
20
25
100/000
001/000
100/000
0/0/000
001/100
000/001
000/010
010/000
001/110
000/100
001/101
010/100
010/110
100/100
001/111
100/000
000/001
010/000

Q5)

FSM



In the State Encoding,
00 represents the Ground Floor
01 represents the First Floor
10 represents the Second Floor

For Ground Floor (00), the output is (R=1, G=0, B=0)
For First Floor (01), the output is (R=0, G=1, B=0)
For Second Floor (10), the output is (R=0, G=0, B=1)

State Transition Table

Present States		**Input**			**Next State**		**Output**		
P_0	P_1	F	S	T	N_0	N_1	R	G	B
0	0	0	0	0	0	0	1	0	0
0	0	0	0	1	1	0	1	0	0
0	0	0	1	0	0	1	1	0	0
0	0	0	1	1	0	1	1	0	0
0	0	1	0	0	0	0	1	0	0
0	0	1	0	1	1	0	1	0	0

0	0	1	1	0	0	1	1	0	0
0	0	1	1	1	0	1	1	0	0
0	1	0	0	0	0	1	0	1	0
0	1	0	0	1	1	0	0	1	0
0	1	0	1	0	0	1	0	1	0
0	1	0	1	1	1	0	0	1	0
0	1	1	0	0	0	0	0	1	0
0	1	1	0	1	0	0	0	1	0
0	1	1	1	0	0	0	0	1	0
0	1	1	1	1	0	0	0	1	0
1	0	0	0	0	1	0	0	0	1
1	0	0	0	1	1	0	0	0	1
1	0	0	1	0	0	1	0	0	1
1	0	0	1	1	0	1	0	0	1
1	0	1	0	0	0	0	0	0	1
1	0	1	0	1	0	0	0	0	1
1	0	1	1	0	0	1	0	0	1
1	0	1	1	1	0	1	0	0	1
1	1	0	0	0	x	x	x	x	x
1	1	0	0	1	x	x	x	x	x
1	1	0	1	0	x	x	x	x	x
1	1	0	1	1	x	x	x	x	x
1	1	1	0	0	x	x	x	x	x
1	1	1	0	1	x	x	x	x	x
1	1	1	1	0	x	x	x	x	x
1	1	1	1	1	x	x	x	x	x

$N_0 = (P_0)' . (P_1)' . S' . T + (P_1) . F' . T + (P_0) . F' . S'$

N0	F,S,T							
(P0),(P1)	000	001	011	010	110	111	101	100
00	0	1	0	0	0	0	1	0
01	0	1	1	0	0	0	0	0
11	-	-	-	-	-	-	-	-
10	1	1	0	0	0	0	0	0

$N_1 = (P_1)' . S + (P_1) . F' . T'$

N1	F,S,T							
(P0),(P1)	000	001	011	010	110	111	101	100
00	0	0	1	1	1	1	0	0
01	1	0	0	1	0	0	0	0
11	-	-	-	-	-	-	-	-
10	0	0	1	1	1	1	0	0

$R = (P_0)' . (P_1)'$

R		P1	
		0	1
P0	0	1	0
	1	0	-

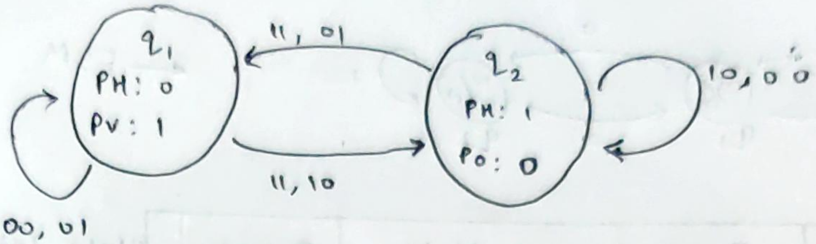
G = P1

G		P1	
		0	1
P0	0	0	1
	1	0	-

B = P0

B		P1	
		0	1
P0	0	0	0
	1	1	-

Q6)



State Encoding : $q_1 \rightarrow 01$

$q_2 \rightarrow 10$

The state ~~error~~ encoding denotes the pass horizontal traffic and pass vertical traffic condition on the current state.

Present States (P_0)	(P_1)	Input (H)	(V)	Next State (N_0)	(N_1)	Output (PH)	(PV)
0	0	0	0	x	x	x	x
0	0	0	1	x	x	x	x
0	0	1	0	x	x	x	x
0	0	1	1	x	x	x	x
0	1	0	0	0	1	0	1
0	1	0	1	0	1	0	1
0	1	1	0	1	0	0	1
0	1	1	1	1	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	1	1	0
1	0	1	0	1	0	1	0
1	0	1	1	0	1	1	0
1	1	0	0	x	x	x	x
1	1	0	1	x	x	x	x
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	x

$N_0 = (P_0)'(H) + (P_1)'(V)'$

$N_1 = (P_0)'(H)' + (P_1)'(V)$ (N_1 can also be implemented as N_0')

$PH = P_0$

$PV = P_1$

Q7)

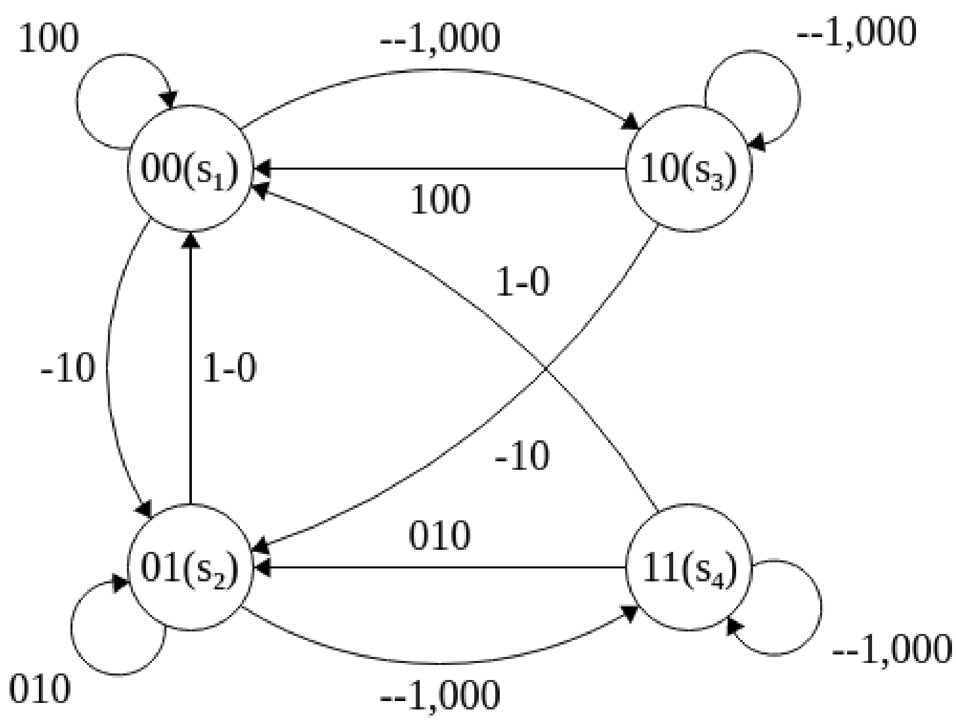
Assumption

The output ST determines input T, for example if the timeout duration is 5 seconds then this is a possible stream

Time (s)	0	1	2	3	4	5	6	7
T	0	0	1	1	1	1	1	0
ST	0	1	0	0	0	0	0	0

Setting ST = 1 at t=1 means that T is 1 from t=2 to t=6

FSM Details



Start state = s3 -> we are assuming we start at PH=1, PV=0, ST=0

State encoding

State name	encoding
s1	00
s2	01
s3	10
s4	11

State Table

Present State	Input			Next State	Output		
	H	V	T		PH	PV	ST
00	-	-	1	10	1	0	0
	0	0	0	10	1	0	0
	-	1	0	01	0	1	1
	1	0	0	00	1	0	1
01	-	-	1	11	0	1	0
	0	0	0	11	0	1	0
	1	-	0	00	1	0	1
	0	1	0	01	0	1	1
10	-	-	1	10	1	0	0
	0	0	0	10	1	0	0
	1	0	0	00	1	0	1
	-	1	0	01	0	1	1
11	-	-	1	11	0	1	0
	0	0	0	11	0	1	0
	1	-	0	00	1	0	1
	0	1	0	01	0	1	1

NEXT STATE -> OUTPUT MAPPING

NEXT STATE	OUTPUT		
	PH	PV	ST
00	1	0	1
01	0	1	1
10	1	0	0
11	0	1	0

DFF Excitation Table

Present State		Input			Next State		Output		
A	B	H	V	T	D_A	D_B	PH	PV	ST
0	0	0	0	1	1	0	1	0	0
0	0	0	1	1	1	0	1	0	0
0	0	1	1	1	1	0	1	0	0
0	0	1	0	1	1	0	1	0	0
0	0	0	0	0	1	0	1	0	0
0	0	0	1	0	0	1	0	1	1
0	0	1	1	0	0	1	0	1	1
0	0	1	0	0	0	0	1	0	1
0	1	0	0	1	1	1	0	1	0
0	1	0	1	1	1	1	0	1	0
0	1	1	0	1	1	1	0	1	0
0	1	1	1	1	1	1	0	1	0
0	1	0	0	0	1	1	0	1	0
0	1	1	0	0	0	0	1	0	1
0	1	1	1	0	0	0	1	0	1
0	1	0	1	0	0	1	0	1	1

1	0	0	0	1	1	0	1	0	0
1	0	0	1	1	1	0	1	0	0
1	0	1	0	1	1	0	1	0	0
1	0	1	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1	0	0
1	0	1	0	0	0	0	1	0	1
1	0	0	1	0	0	1	0	1	1
1	0	1	1	0	0	1	0	1	1
1	1	1	0	1	1	1	0	1	0
1	1	1	1	1	1	1	0	1	0
1	1	0	0	1	1	1	0	1	0
1	1	0	1	1	1	1	0	1	0
1	1	0	0	0	1	1	0	1	0
1	1	1	0	0	0	0	1	0	1
1	1	1	1	0	0	0	1	0	1
1	1	0	1	0	0	1	0	1	1

Reasoning for transitions

Description	What the FSM does
Timeout is activated after a pass signal is given.	Whenever the output changes from PH=0, PV=1 to PH=1,PV=0 or vice versa, ST is always set to 1
If there's no traffic after a timeout, then timeout is not active.	000 input loops on s3 and s4 states which don't activate the timer
The pass signal doesn't change when timeout is active.	Input --1 never changes the output PH and PV values
If only horizontal or vertical traffic is	Any change in signal leads to outputs with

present when timeout is inactive, that should be passed and timeout activated.	ST=1
If there's no traffic, the current pass signal is to be maintained.	001 and 000 dont change output PH and PV
If both traffic are present when timeout is inactive, the one that was not favoured last time should be favoured this time and timeout should be activated -- favouring is applicable only when both are present together when there timeout is inactive.	Input 110 leads to 1. PH=1 PV=0 to go to PH=0 PV=1 and ST=1 2. PH=0 PV=1 to go to PH=1 PV=0 and ST=1

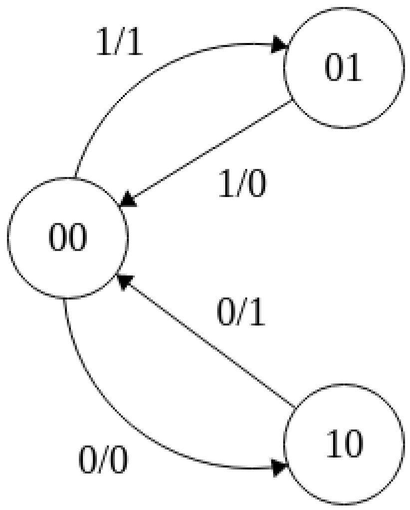
K-Map for D_A

Da	H,V,T 000	001	011	010	110	111	101	100
A,B 00	1	1	1	0	0	1	1	0
01	1	1	1	0	0	1	1	0
11	1	1	1	0	0	1	1	0
10	1	1	1	0	0	1	1	0

K-Map for D_B

Db	H,V,T 000	001	011	010	110	111	101	100
A,B 00	0	0	0	1	1	0	0	0
01	1	1	1	1	0	1	1	0
11	1	1	1	1	0	1	1	0
10	0	0	0	1	1	0	0	0

Q8)
The FSM is as follows:

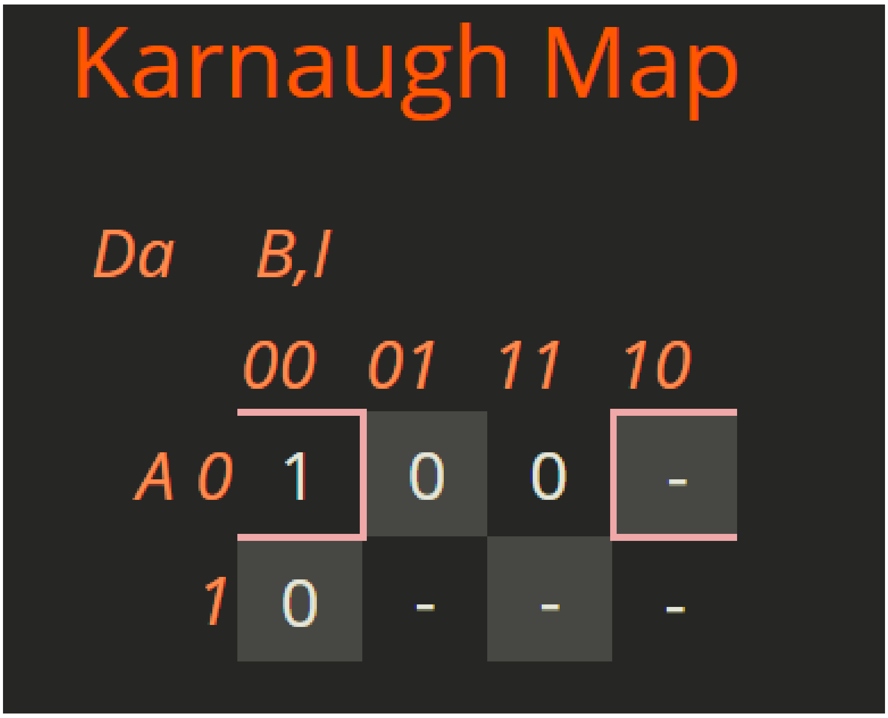


The state 00 is the idle state, 01 is used for showing 1-0 transition when data bit 1 is received and 10 is used for showing 0-1 transition when data bit 0 is received.

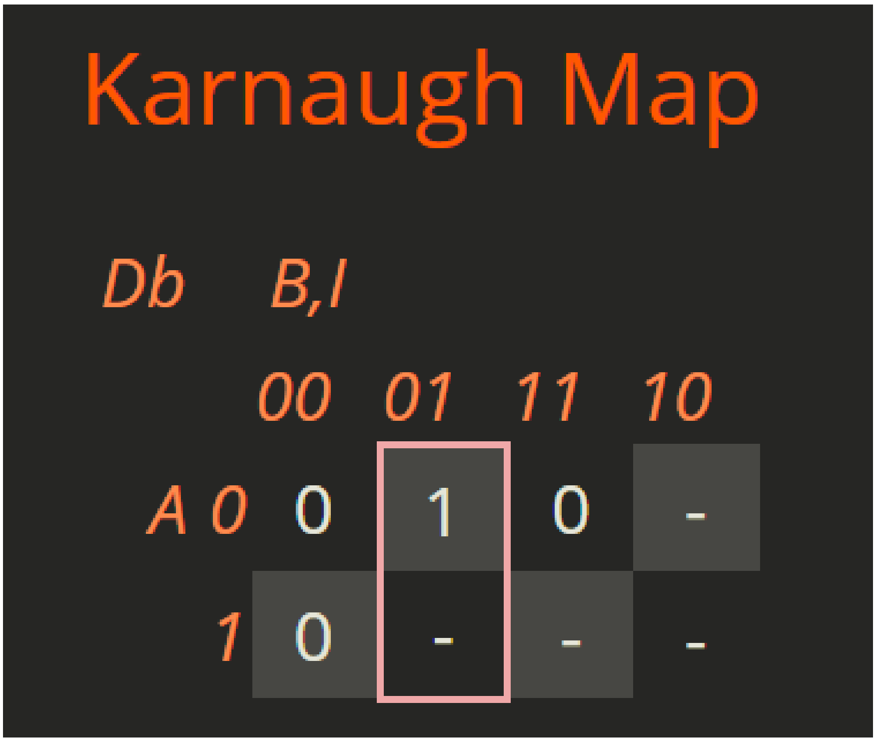
The state table is as follows:

Current State		Input	Next State		Output
A	B	I	D_A	D_B	Y
0	0	0	1	0	0
0	0	1	0	1	1
0	1	0	x	x	x
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	x	x	x
1	1	0	x	x	x
1	1	1	x	x	x

1. $D_A = A' . I'$



2. $D_B = B' . I$



3. $Y = (A + B' . I) . V$