Practical 16

AIM: To implement Boolean function

Theory:

F (a, b, c, d) = ∑(0,2,5,8,10,14) with a multiplexer

	D0	D1	D2	D3	D4	D5	D6	D7
A'	(0)	1	(2)	3	4	(5)	6	7
A	(8)	9	(10)	11	12	13	(14)	15
Mux i/p	1	0	1	0	0	A'	A	0

F(a, b, c, d) = π (2,6,11) with a multiplexer

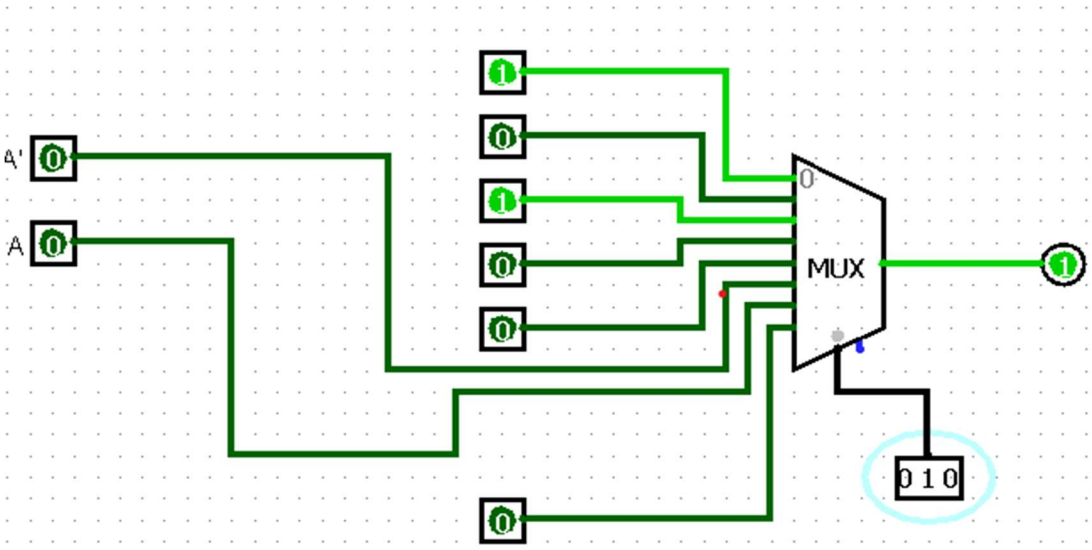
	D0	D1	D2	D3	D4	D5	D6	D7
A'	(0)	(1)	2	(3)	(4)	(5)	6	(7)
A	(8)	(9)	(10)	11	(12)	(13)	(14)	(15)
Mux i/p	1	1	A	A'	1	1	A	1

c) F (a, b, c, d) = ∑(1,3,4,11,12,13,14,15) with a multiplexer.

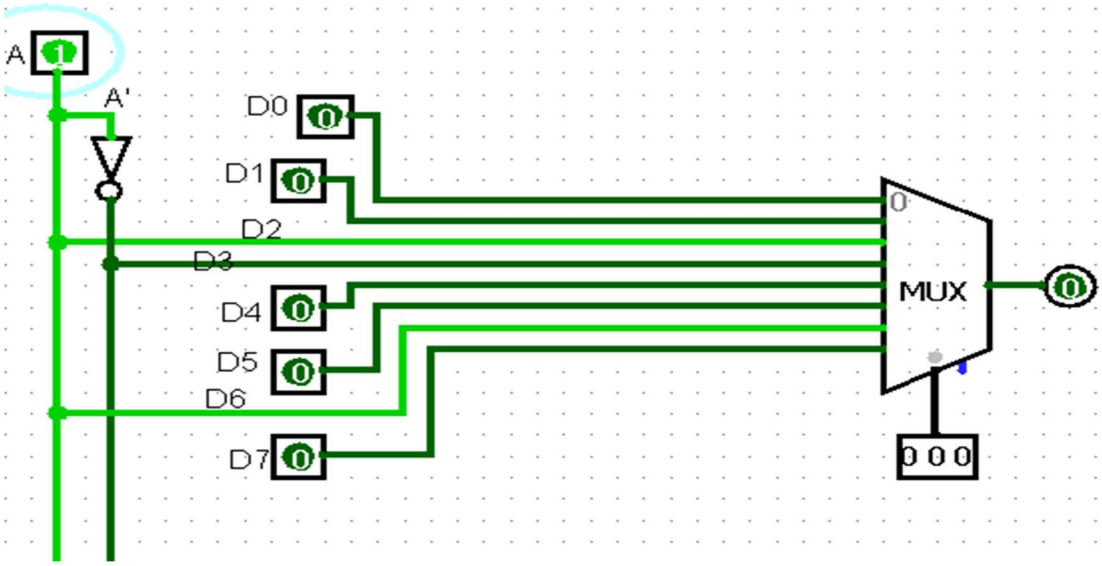
	D0	D1	D2	D3	D4	D5	D6	D7
A'	0	(1)	2	(3)	(4)	5	6	7
A	8	9	10	(11)	(12)	(13)	(14)	(15)
Mux i/p	0	A'	0	1	1	A	A	A

Implementation:-

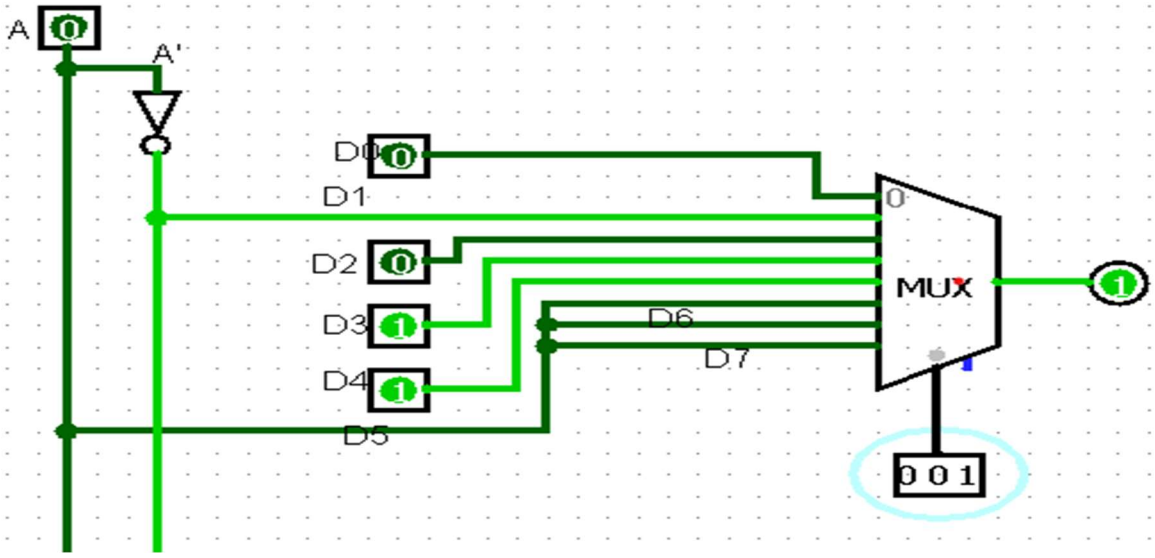
1.



2.



3.



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

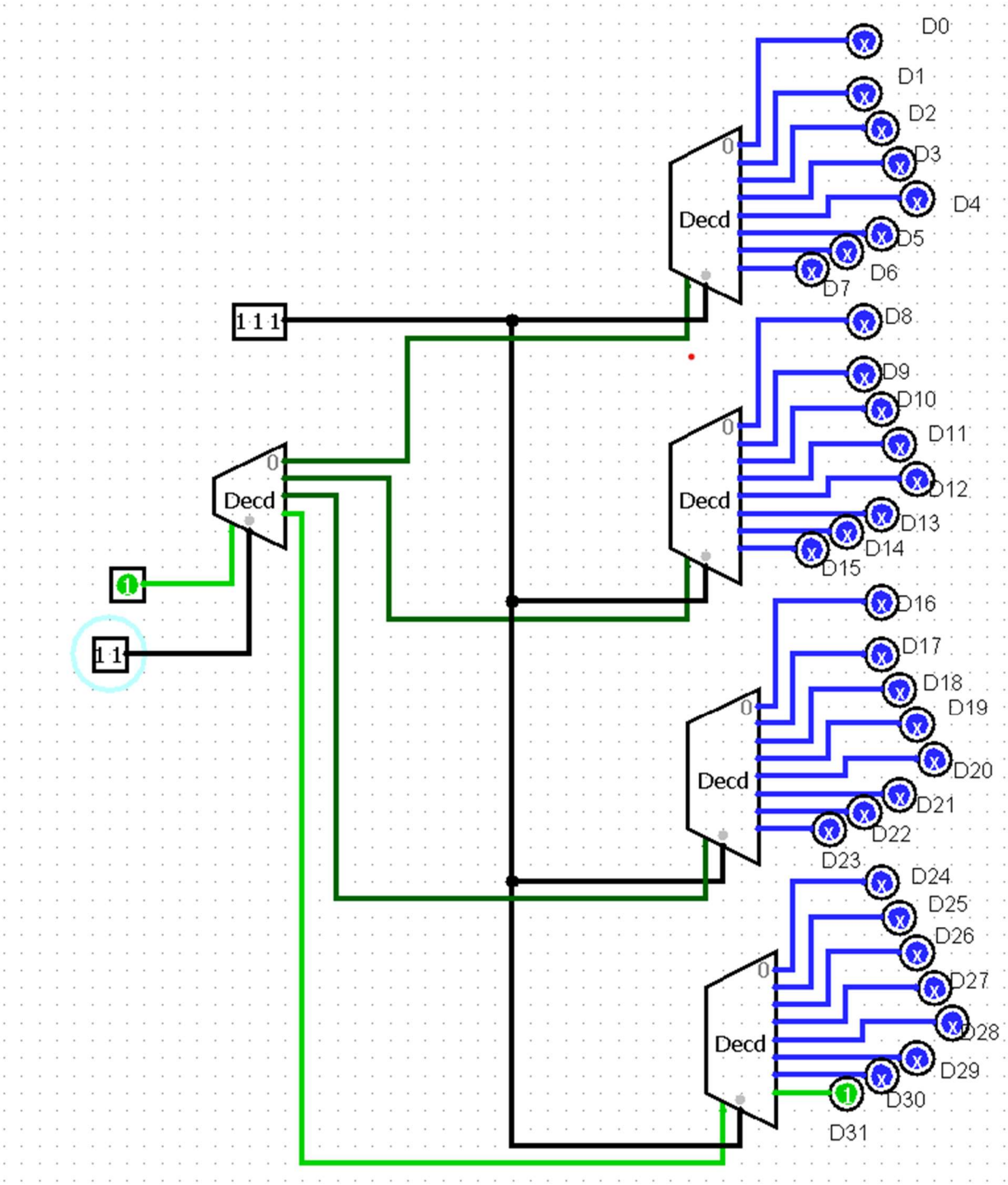
Practical 17

AIM: Design 5 to 32 line decoder using basic decoders

Theory:

A4	A3	A2	A1	A0	Output (Y0-Y31)
0	0	0	0	0	Y0
0	0	0	0	1	Y1
0	0	0	1	0	Y2
0	0	0	1	1	Y3
0	0	1	0	0	Y4
0	0	1	0	1	Y5
0	0	1	1	0	Y6
0	0	1	1	1	Y7
0	1	0	0	0	Y8
0	1	0	0	1	Y9

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 18

AIM: Design parallel adder circuit which can decrement given input value

Theory:

Parallel Adder Circuit with Decrement Capability

To design a parallel adder circuit that can decrement a given input value, we can use a combination of logic gates and a parallel adder. Here's a possible implementation:

Circuit Components:

4-bit parallel adder (e.g., 7483 or 74LS283)

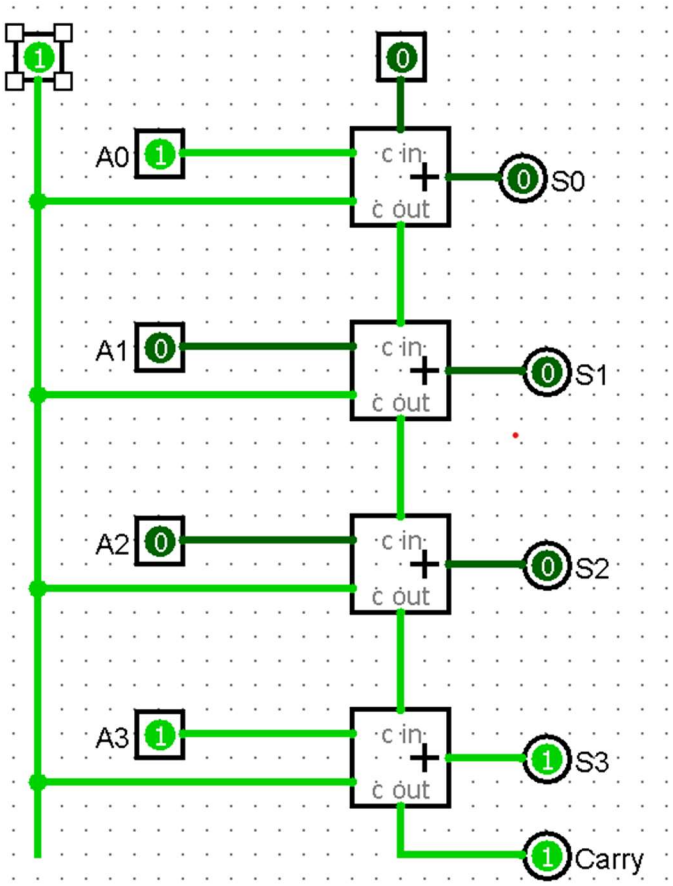
2's complement generator (e.g., XOR gates and inverters)

Control logic (e.g., AND gates and OR gates)

Circuit Operation:

1. The 4-bit input value is applied to the 2's complement generator.
2. The 2's complement generator produces the 2's complement of the input value, which is essentially the negative of the input value.
3. The 2's complement value is applied to the parallel adder, along with a constant value of 1.
4. The parallel adder performs the addition of the 2's complement value and the constant value, producing the decremented value.
5. The control logic selects either the original input value or the decremented value, based on a control signal (e.g., a decrement enable signal).
6. The selected value is applied to the 4-bit output.

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 19

AIM: Design the circuit diagram which can explain the cascading concept in De-multiplexer

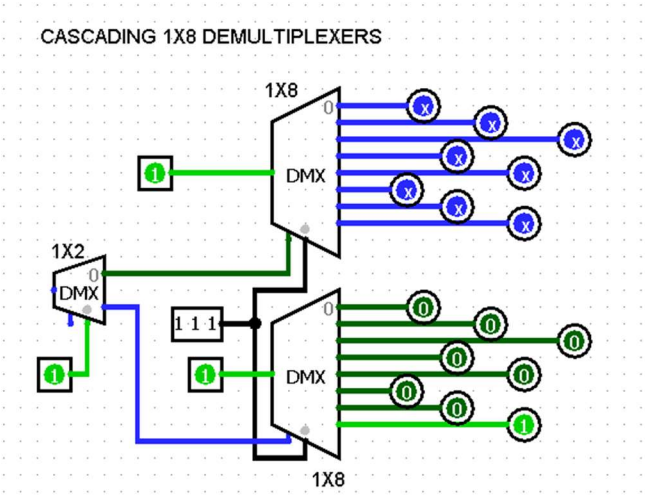
Theory:

Cascading Concept in De-Multiplexers:

Cascading is a technique used to increase the number of output lines in a De-Multiplexer (DEMUX) by connecting multiple smaller DEMUXes together. This allows us to create larger DEMUXes with more output lines, while still using smaller, more manageable DEMUX circuits.

S1	S0	Input (X)	Output 0	Output 1	Output 2	Output 3
0	0	X	Y	0	0	0
0	1	X	0	Y	0	0
1	0	X	0	0	Y	0
1	1	X	0	0	0	Y

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 20

AIM: Design the circuit diagram which can explain the cascading concept in Multiplexer

Theory:

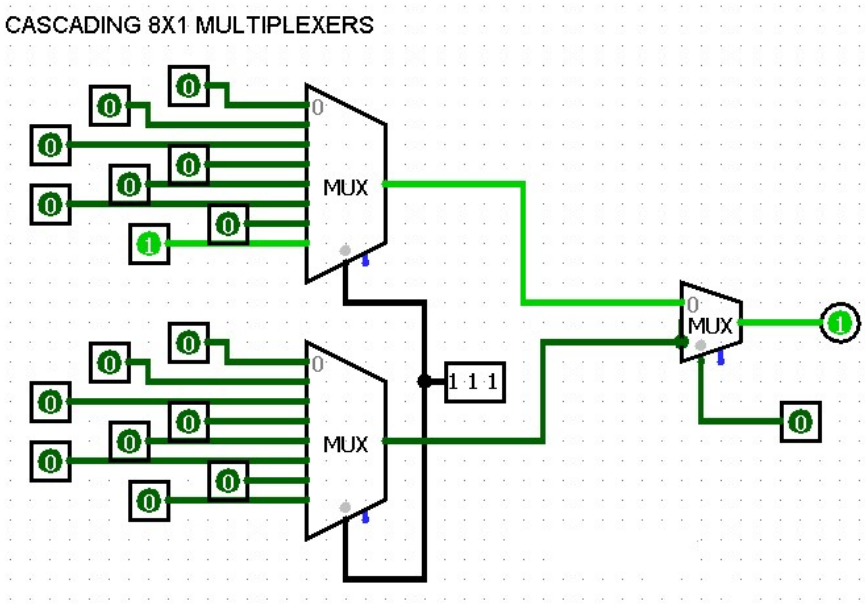
Cascading Concept in Multiplexers:

Cascading is a technique used to increase the number of input lines in a Multiplexer (MUX) by connecting multiple smaller MUXes together. This allows us to create larger MUXes with more input lines, while still using smaller, more manageable MUX circuits

MUX 2:

S1	S0	Data In 4	Data In 5	Data In 6	Data In 7	Output (Y2)
0	0	Y	0	0	0	Y
0	1	0	Y	0	0	Y
1	0	0	0	Y	0	Y
1	1	0	0	0	Y	Y

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 21

AIM: Design the circuit diagram which can explain the utility of MUX-DEMUX

Theory:

S1	S0	Data In 0	Data In 1	Data In 2	Data In 3	Output (Y)	Data Out 0	Data Out 1	Data Out 2	Data Out 3
0	0	Y	0	0	0	Y	Y	0	0	0
0	1	0	Y	0	0	Y	0	Y	0	0
1	0	0	0	Y	0	Y	0	0	Y	0
1	1	0	0	0	Y	Y	0	0	0	Y

Explanation:

The MUX (Multiplexer) takes four input data lines (Data In 0 to Data In 3) and selects one of them to send to the output (Y) based on the select inputs S1 and S0.

The DEMUX (Demultiplexer) takes the output (Y) from the MUX and sends it to one of the four output data lines (Data Out 0 to Data Out 3) based on the select inputs S1 and S0.

The select inputs S1 and S0 are used to control the MUX and DEMUX. The truth table shows the output (Y) and the corresponding data out lines for each combination of S1 and S0.

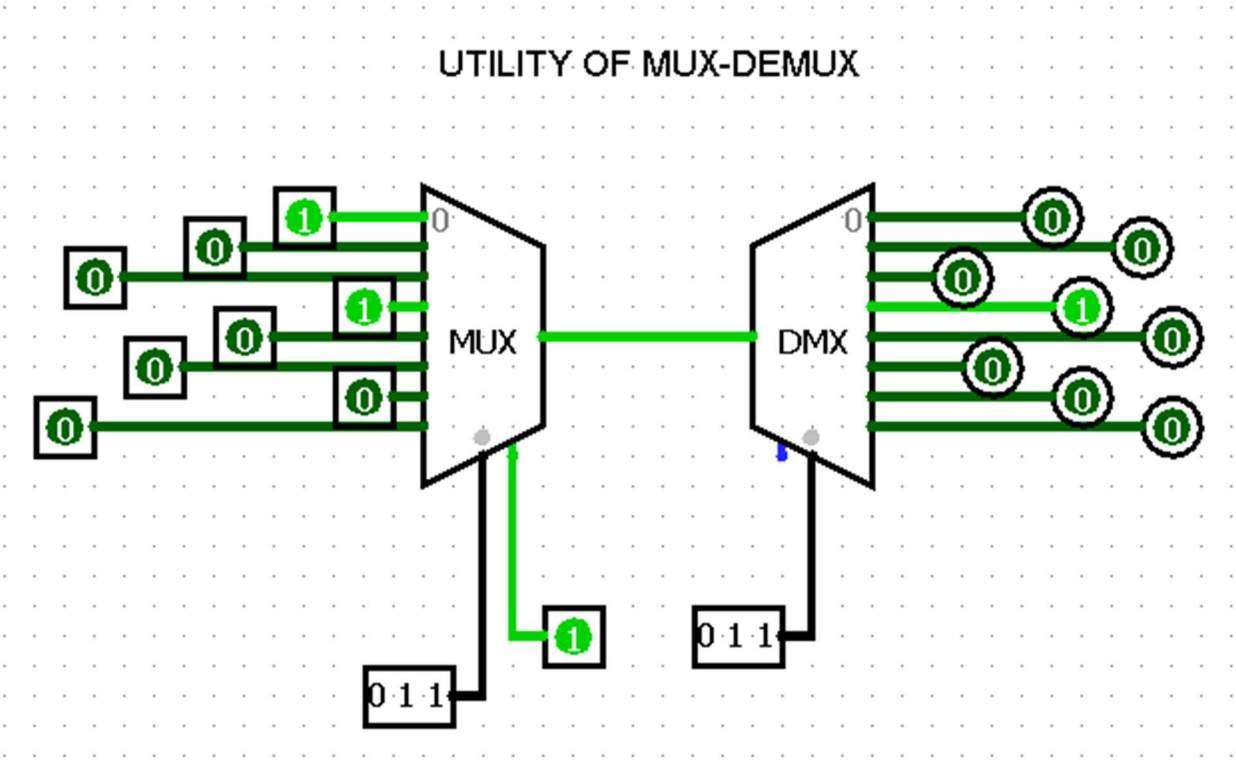
Utility of MUX-DEMUX:

The MUX-DEMUX system allows multiple data lines to share a single transmission line, increasing the efficiency of data transmission.

It enables the selection of one of multiple data sources to be sent to a single output, or the distribution of a single input to multiple outputs.

MUX-DEMUX systems are widely used in digital communication systems, computer networks, and other applications where data needs to be multiplexed or demultiplexed.

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 22

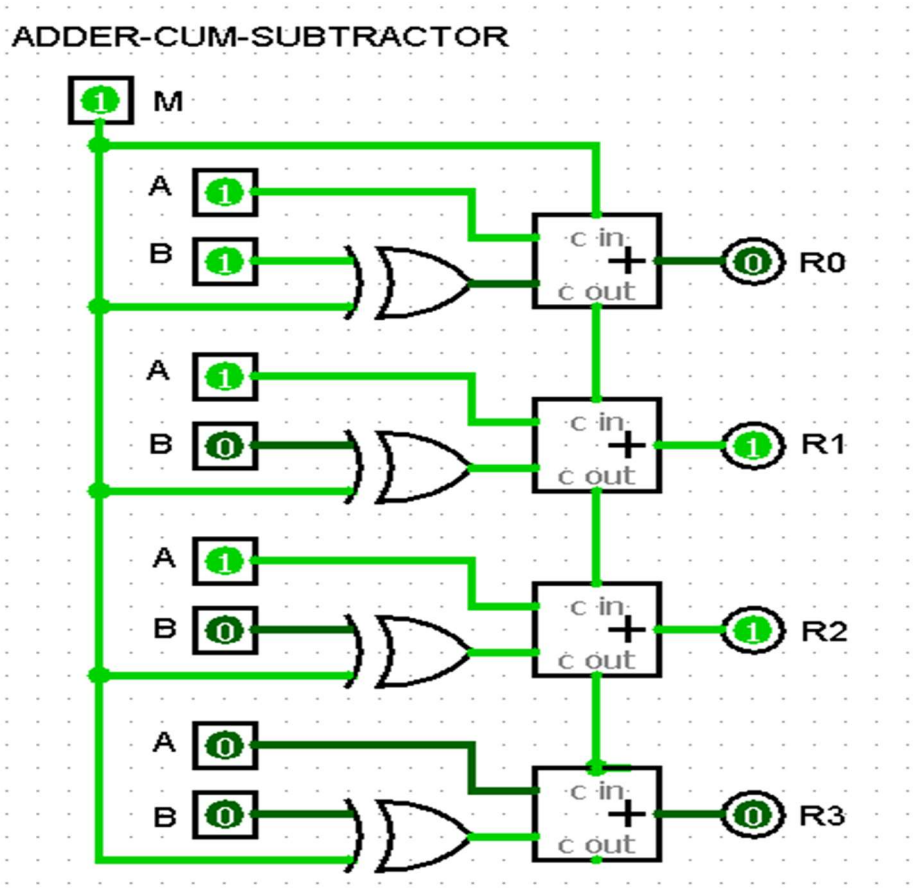
AIM: Design the circuit diagram to show a common adder-cum-subtraction

Theory:

Truth Table:

A	B	Cin	M	S	Cout
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	1	0
0	0	1	1	1	0
0	1	0	0	1	0
0	1	0	1	1	1
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	0	1
1	0	1	1	0	1
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	1	1
1	1	1	1	1	1

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 23

AIM: Explain the working of all the following flip flop in Logisim simulator a) S-R flip flop b) D flip flop c) J-K flip flop d) T flip flop

- **S-R FLIP FLOP**

State	S	R	Q	Q'	Description
Set	1	0	0	1	Set Q'>>1
	1	1	0	1	No change
Reset	0	1	1	0	Reset Q'>>0
	1	1	1	0	No change
Invalid	0	0	1	1	Invalid Condition

- D FLIP FLOP

Clk	D	Q	Q'	State
0	0	Q	Q'	No change in state
1	0	0	1	Resets Q to 0
1	1	1	1	Sets Q to 1

- J-K flip flop

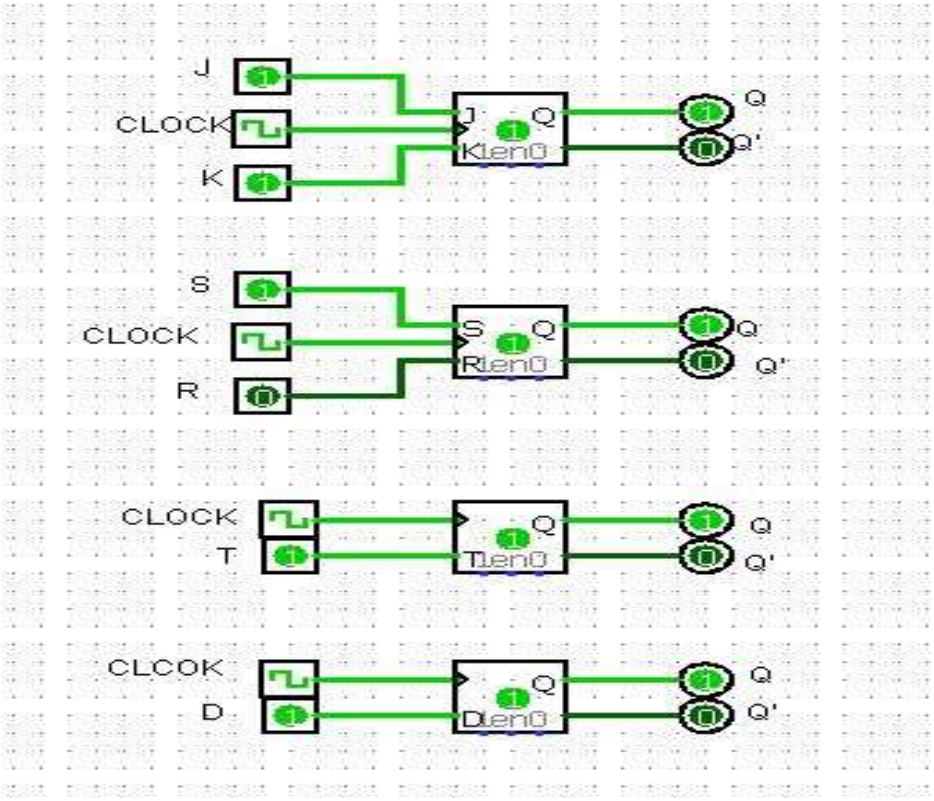
Clock	J	K	Q_{n+1}	State
0	x	x	Q_n	
1	0	0	Q_n	Hold
1	0	1	0	Reset
1	1	1	1	Set
1	1	1	$\overline{Q}_n$	Toggle

- T flip flop

TRUTH TABLE

CLK	T	Q n	Q n+1
↓	0	0	0
↓	0	1	1
↓	1	0	1
↓	1	1	0

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 24

AIM: Design master slave flip flop (using every flip flops)

SR-MASTER SLAVE

S	R	Q(t+1)	Qn(t+1)	Meaning
0	0	Q	Qn	Hold
0	1	0	1	Reset
1	0	1	0	Set
1	1	?	?	Undefined

JK-MASTER SLAVE

J	K	Q Before CK Pulse	Q After CK Pulse	Comments
0	0	0	0	No Change
0	0	1	1	
1	0	0	1	$Q = J$
1	0	1	1	
0	1	0	0	$\overline{Q} = K$
0	1	1	0	
1	1	0	1	Outputs Toggle
1	1	1	0	

D-MASTER SLAVE

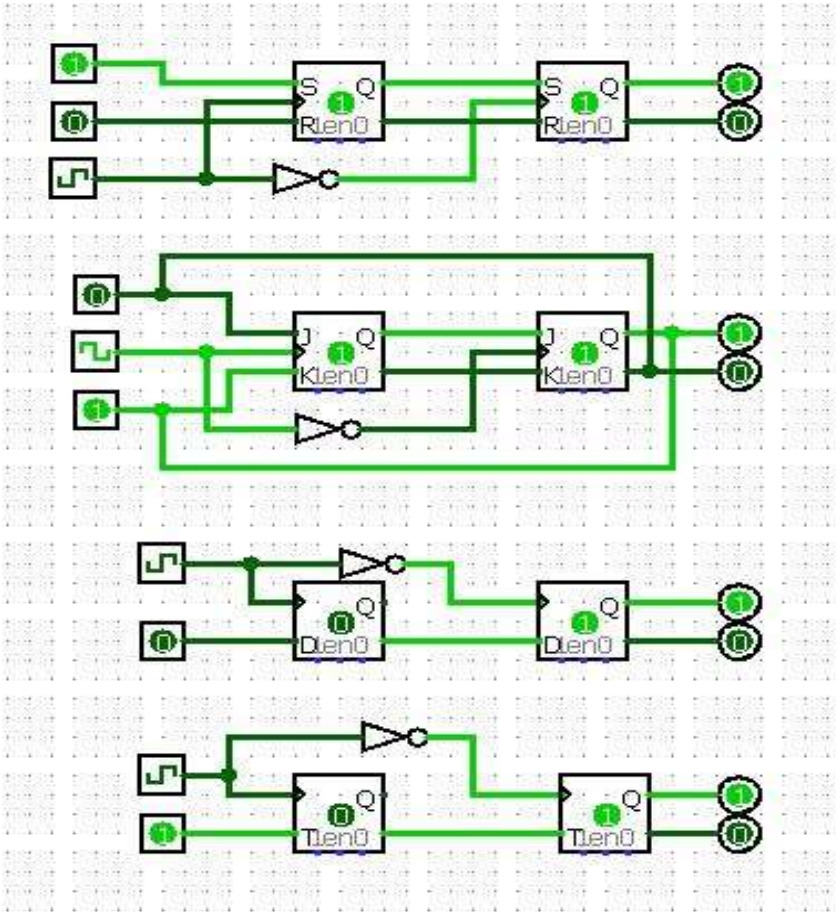
Clk	D	Q	Q'	State
0	0	Q	Q'	No change in state
1	0	0	1	Resets Q to 0
1	1	1	1	Sets Q to 1

T-MASTER SLAVE

TRUTH TABLE

CLK	T	Q n	Q n+1
↓	0	0	0
↓	0	1	1
↓	1	0	1
↓	1	1	0

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 25

AIM: Prepare J-K, D and T flip flop using S-R Flip flop

Theory: -

JK using SR

J	K	Q(t)	Q(t+1)
0	0	Q(t)	Q(t)
0	1	0	0
1	0	1	1
1	1	Q(t)'	Q(t)'

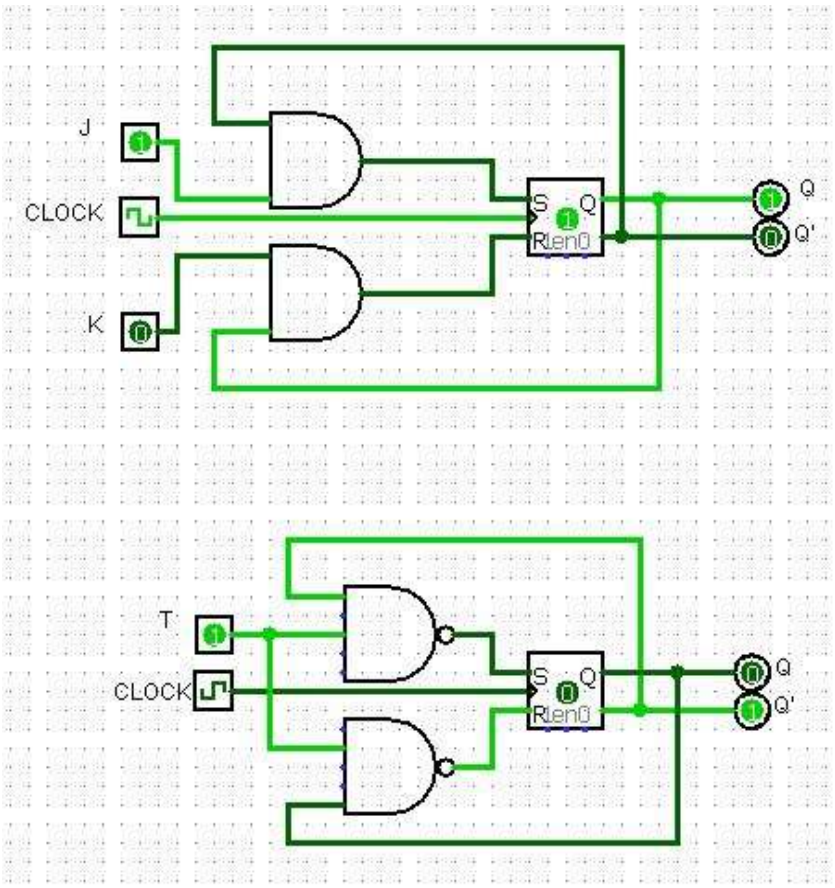
D using SR

D	Q(t)	Q(t+1)
0	Q(t)	0
1	Q(t)	1

T using SR

T	Q(t)	Q(t+1)
0	Q(t)	Q(t)
1	Q(t)	Q(t)'

Implementation:-


J
CLOCK
K
S
Q
R
Q
Q'
T
CLOCK
S
Q
R
Q
Q'
D
CLOCK
S
Q
R
Q
Q'

Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 26

AIM: Prepare S-R, D and T flip flop using J-K Flip flop

Theory: -

SR using JK

S	R	Q(t)	Q(t+1)
0	0	Q(t)	Q(t)
0	1	0	0
1	0	1	1
1	1	undefined	undefined

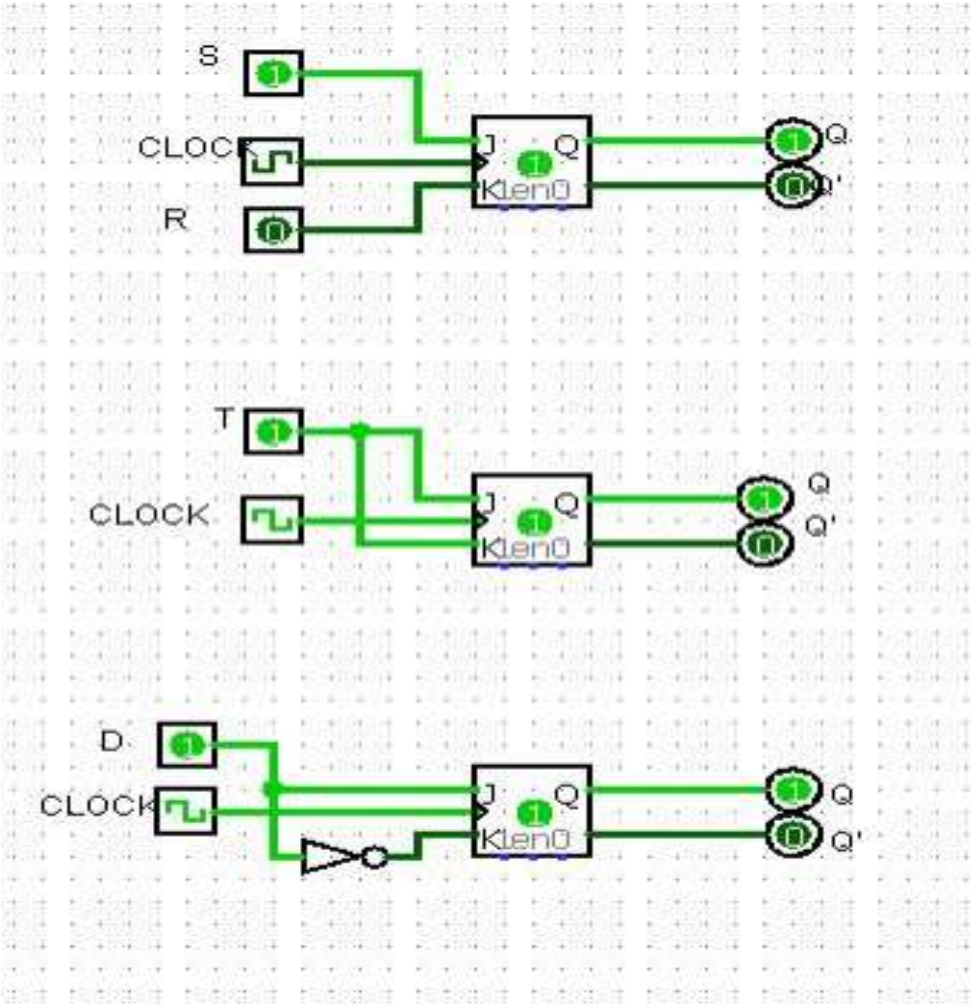
D using JK

D	Q(t)	Q(t+1)
0	Q(t)	0
1	Q(t)	1

T using JK

T	Q(t)	Q(t+1)
0	Q(t)	Q(t)
1	Q(t)	Q(t)'

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 27

AIM: Prepare J-K, S-R and T flip flop using D Flip flop

Theory: -

JK using D

J	K	Q(t)	Q(t+1)
0	0	Q(t)	Q(t)
0	1	0	0
1	0	1	1
1	1	Q(t)'	Q(t)'

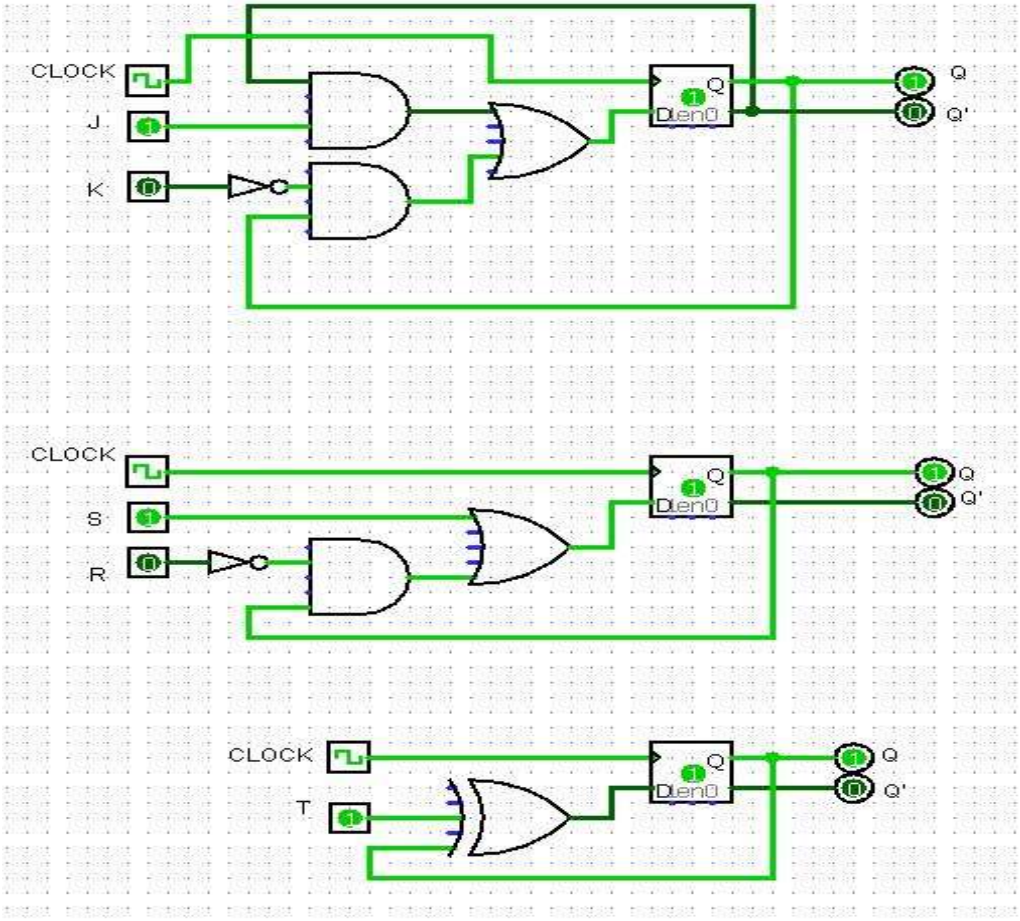
SR using D

S	R	Q(t)	Q(t+1)
0	0	Q(t)	Q(t)
0	1	0	0
1	0	1	1
1	1	undefined	undefined

T using D

T	Q(t)	Q(t+1)
0	Q(t)	Q(t)
1	Q(t)	Q(t)'

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 28

AIM; Prepare J-K, D and S-R flip flop using T Flip flop

Theory: -

JK using T

J	K	Q(t)	Q(t+1)
0	0	Q(t)	Q(t)
0	1	0	0
1	0	1	1
1	1	Q(t)'	Q(t)'

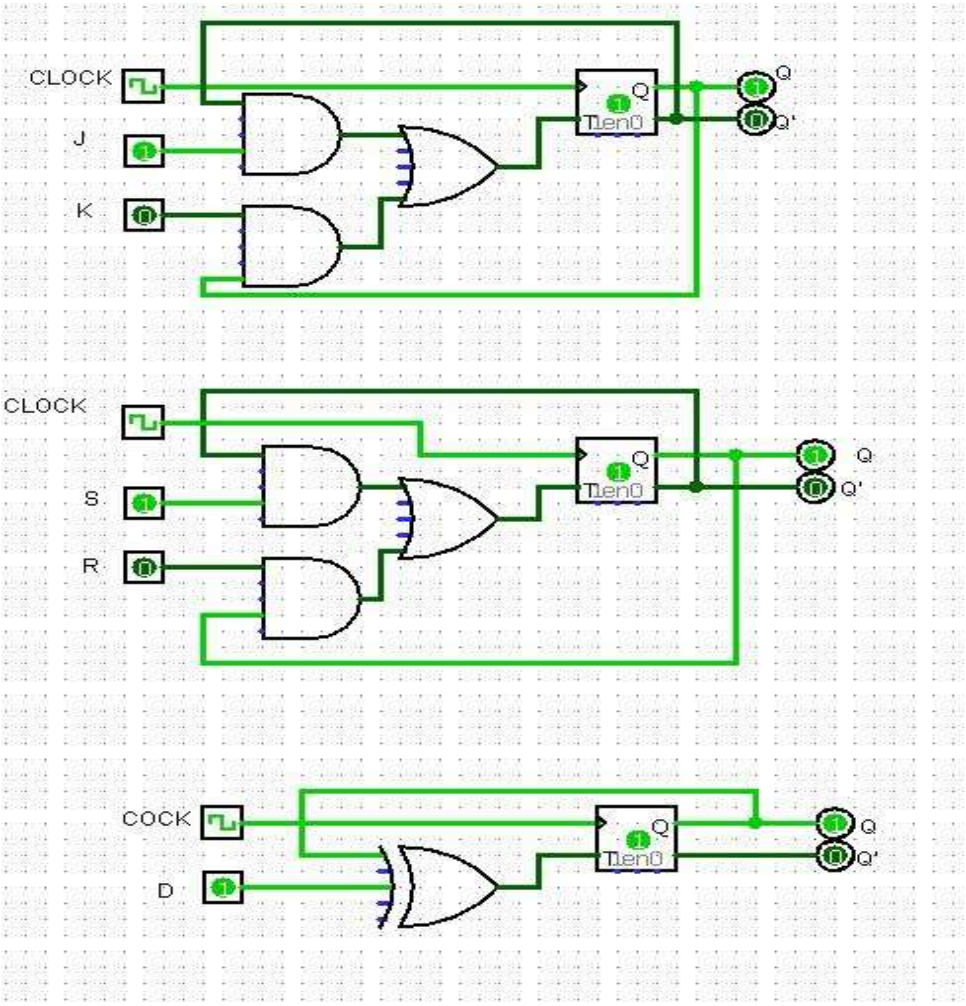
D using T

D	Q(t)	Q(t+1)
0	Q(t)	0
1	Q(t)	1

SR using T

S	R	Q(t)	Q(t+1)
0	0	Q(t)	Q(t)
0	1	0	0
1	0	1	1
1	1	undefined	undefined

Implementation:-



Rubric wise marks obtained:

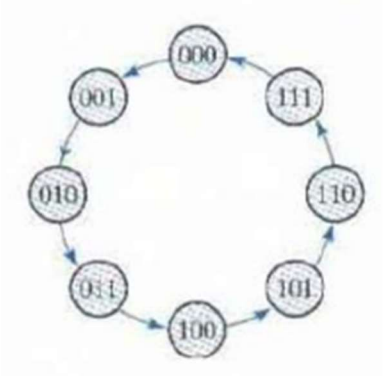
Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

Practical 29

AIM: Prepare a sequential circuit diagram to explore the functionality of given state diagram

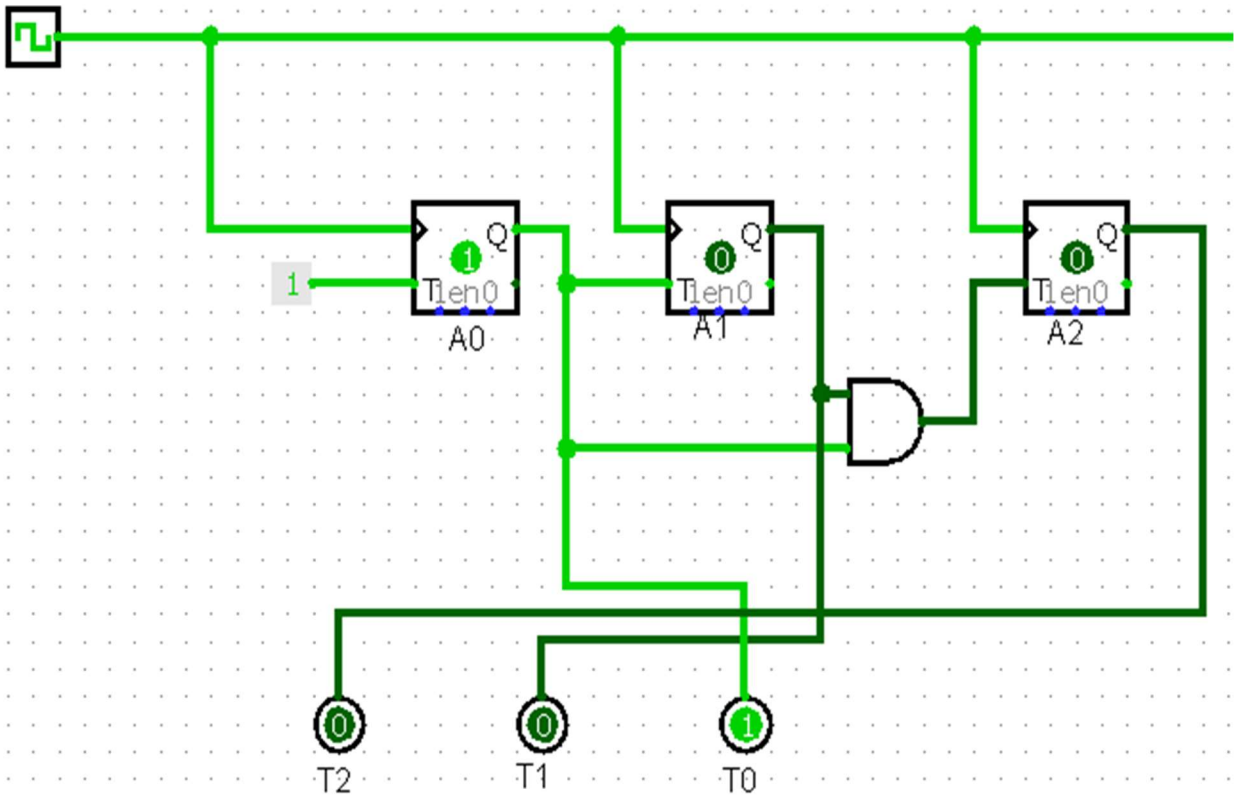
Theory: -



State Table for Three-Bit Counter

Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	1	1
1	1	1	0	0	0	1	1	1

Implementation:-



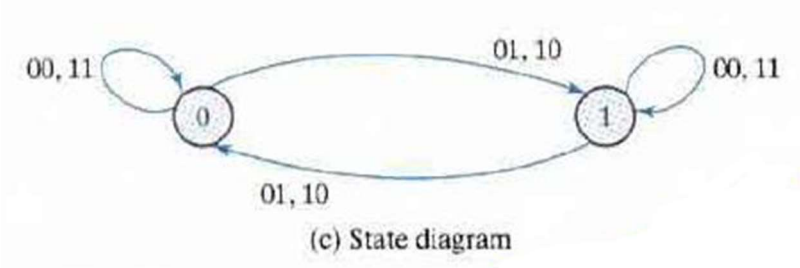
Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature

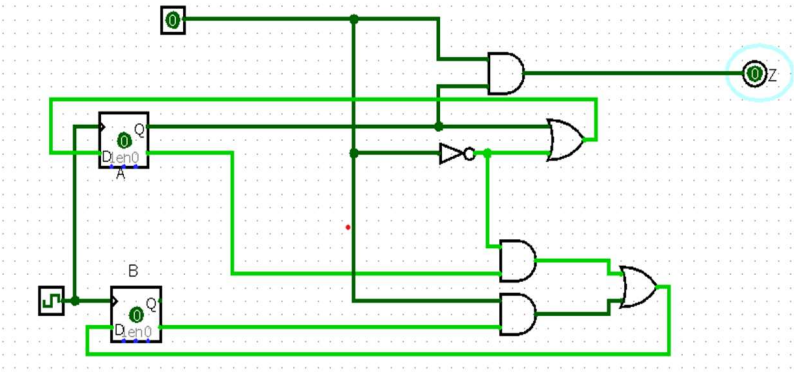
Practical 30

AIM: Prepare a sequential circuit diagram using D flip flop of the given state diagram



(c) State diagram

Implementation:-



Rubric wise marks obtained:

Rubrics	Regularity		Problem Understanding & Implementation of Solution in Simulator		Testing of the Solution		Documentation		Mock Viva Test		Total out of 10
	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	Good (2)	Avg. (1)	

Faculty Signature