EC-204

<LAB - 5>

NITK SURATHKAL

INBASEKARAN.P

201EC226

Prof: Sumam S

1) Design a circuit that will perform addition of two single digit BCD numbers and display the output on seven segment displays. Use the Adder available in Arithmetic library in Logisim.

Solution:

Design and explanation

To implement BCD adder, we require:

• 4-bit binary adder for initial addition

• Logic circuit to detect sum greater than 9

• One more 4-bit adder to add $0110_2 01102$ in the sum if sum is greater than 9 or carry is 1

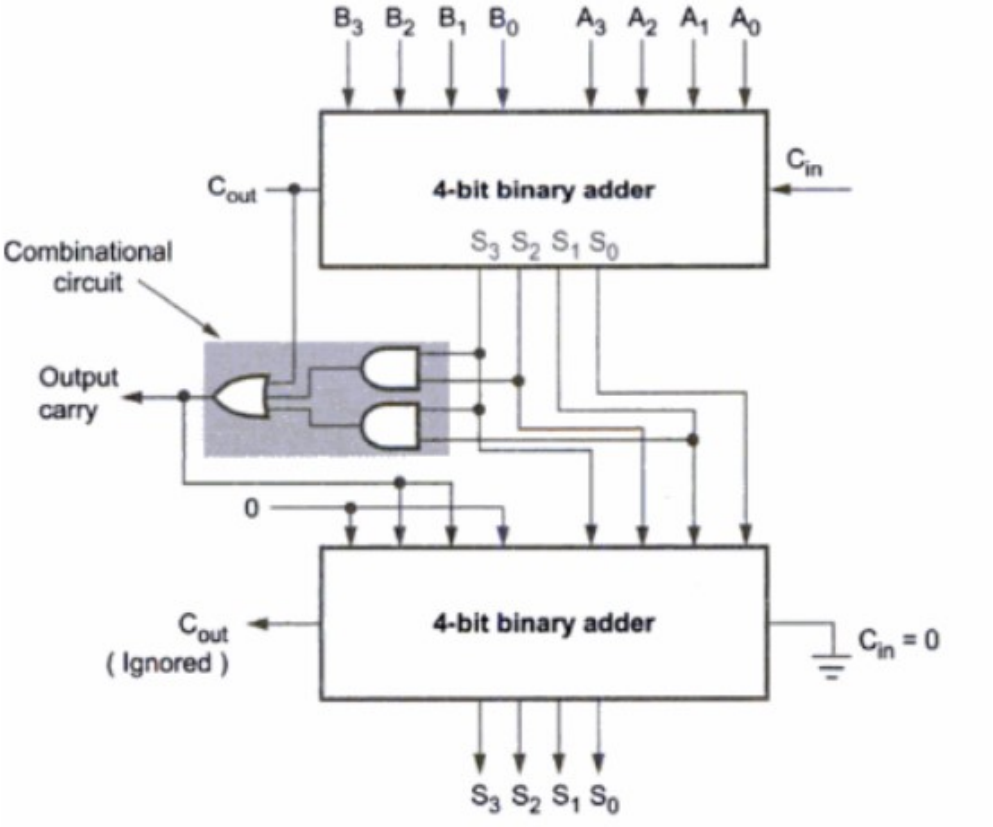
The logic circuit to detect sum greater than 9 can be determined by simplifying the Boolean expression of given truth Table.

Inputs				Output
S_3	S_2	S_1	S_0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
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	1

S_3S_2 \ S_1S_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

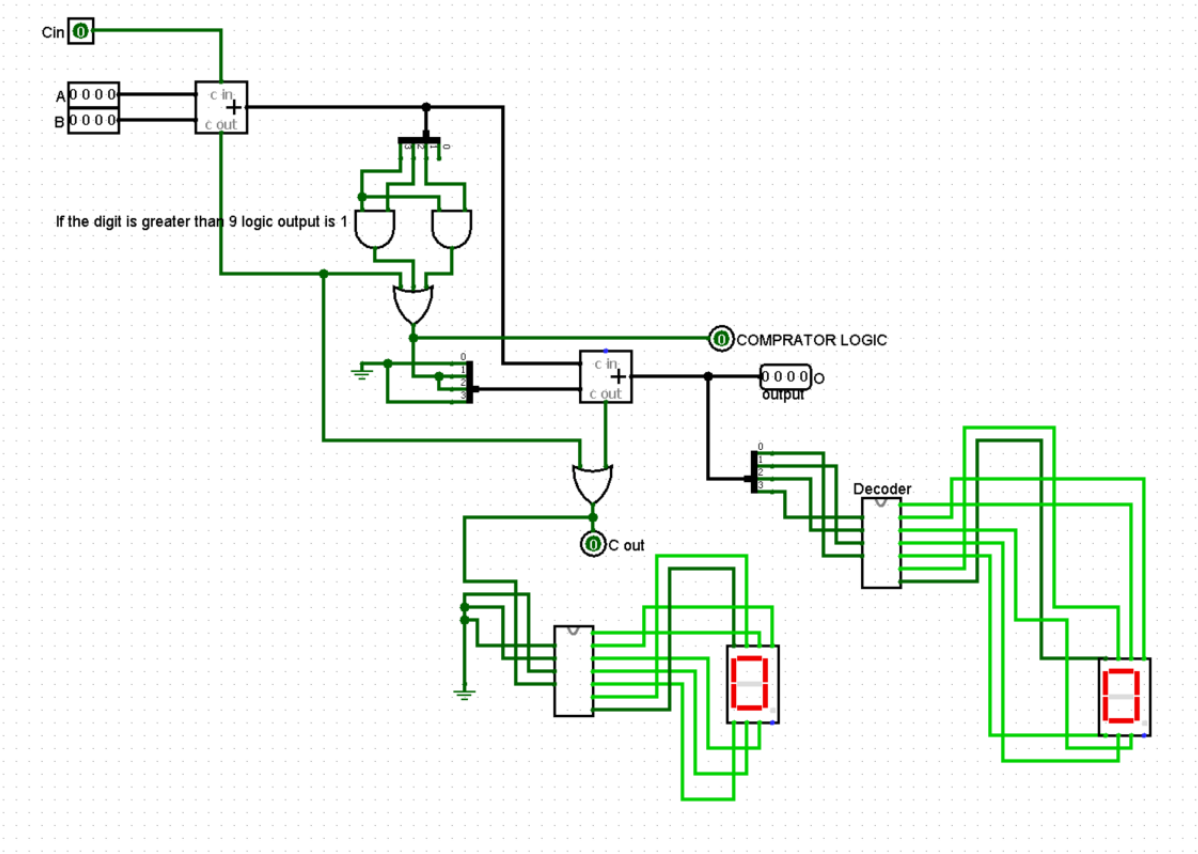
$Y=S_3S_2+S_3S_1$

Y=1 indicates sum is greater than 9. We can put one more term, Cout in the above expression to check whether carry is one. If any one condition is satisfied, we add 6(0110) in the sum.

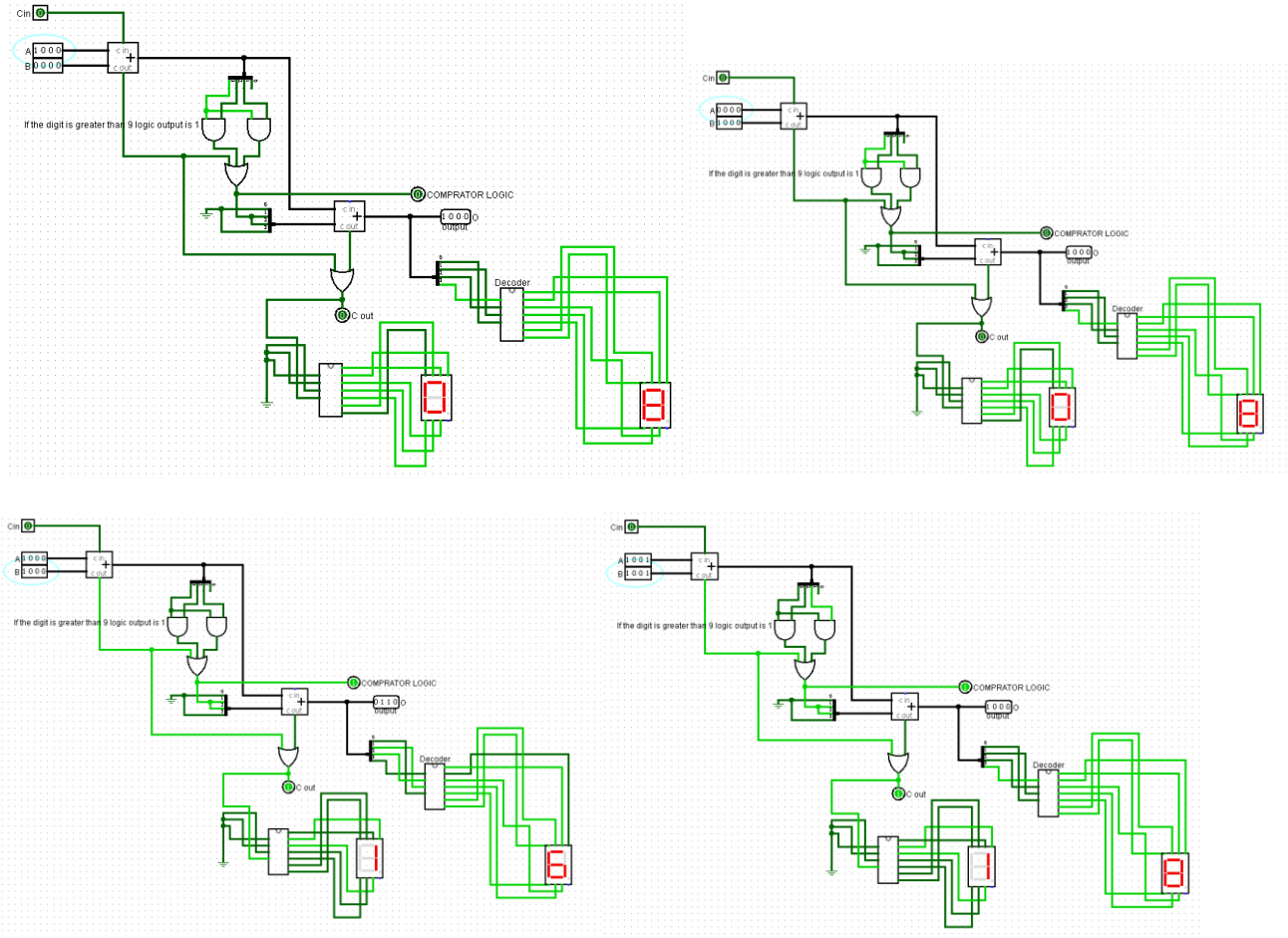


Now we just display the output in the 7 segment LED using the decoder which we implemented in Lab 1.

CIRCUIT


Cin
A 0 0 0 0
B 0 0 0 0
c in
c out
If the digit is greater than 9 logic output is 1
COMPRATOR LOGIC
0 0 0 0
output
Decoder
C out

RESULT OBTAINED



2) Design a circuit that will perform addition of two single digit BCD numbers and display the output on seven segment displays. Use the Adder available in Arithmetic library in Logisim.

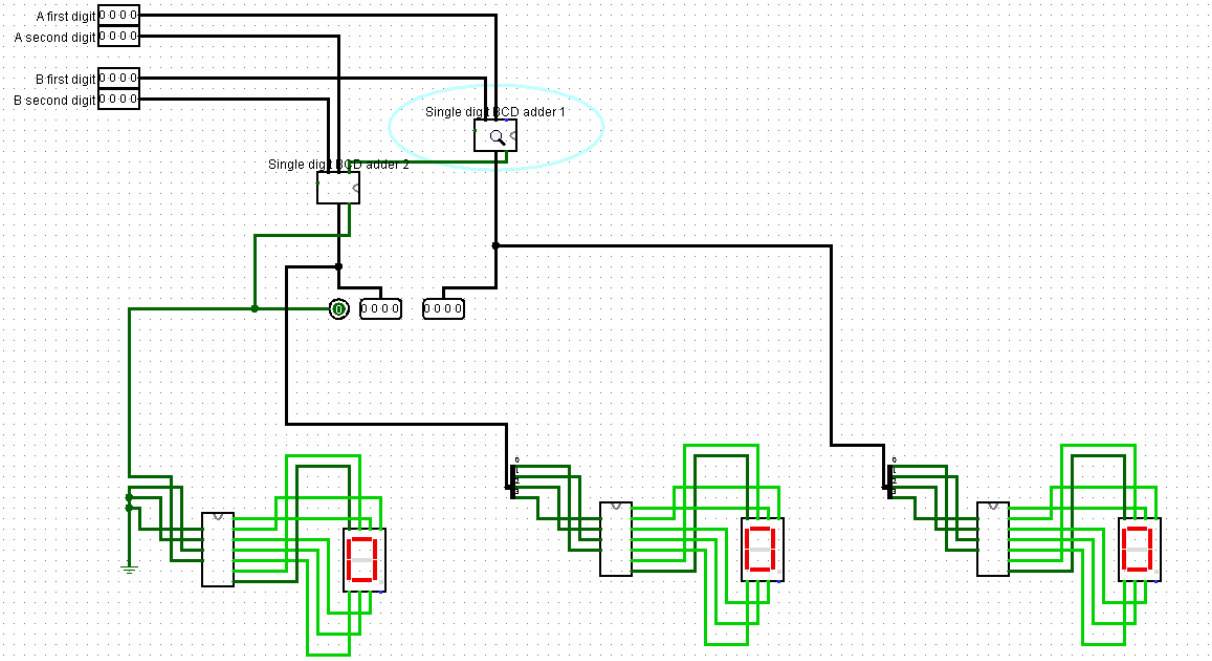
Solution:

Design and explanation

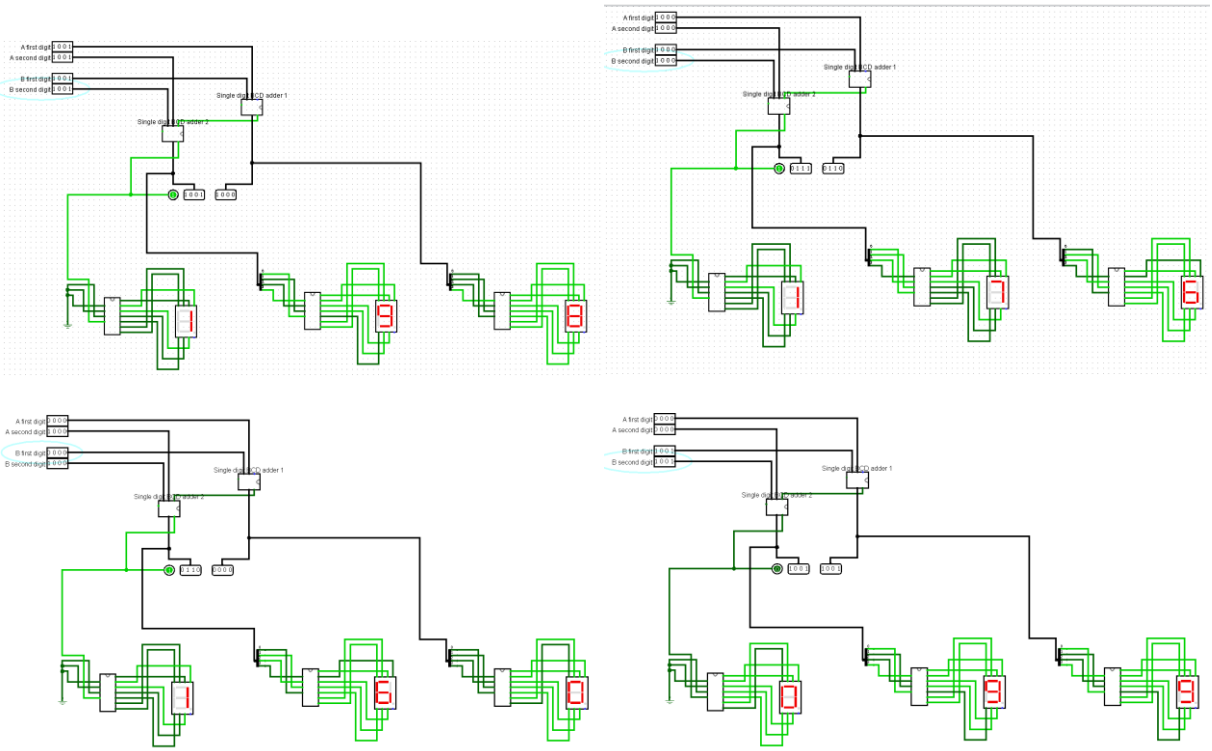
To implement BCD adder, for two digits we can use the previous single digit adder which works up to (9 + 9 = 18)

The first adder adds the first digit (BCD), in the second adder we add the second digit of the BCD number, along with the digit we also give the Cin input as the comparator logic, now it will work for two digit BCD numbers, the following output is displayed in the 7segment led.

CIRCUIT



RESULT OBTAINED



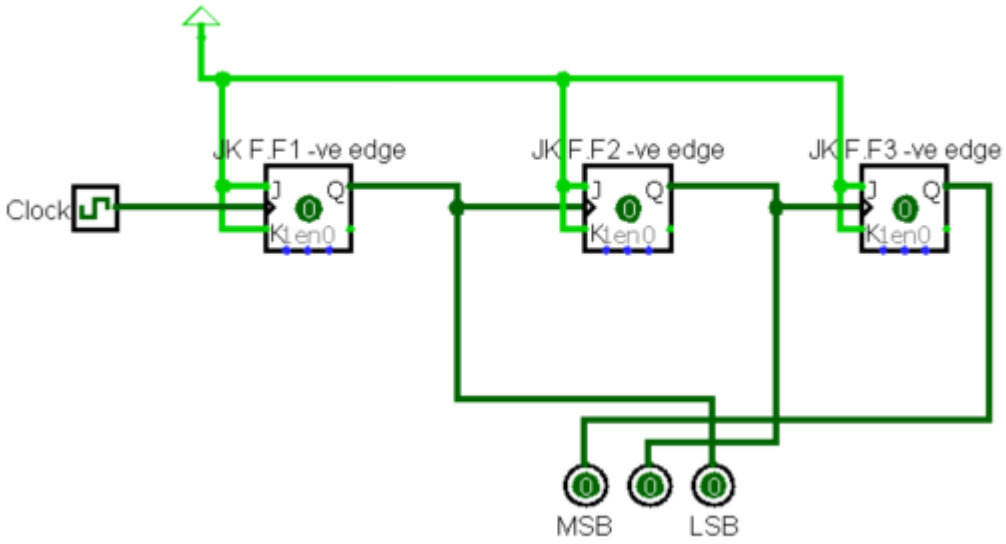
3) (a) Design a modulo 8 ripple up counter using JKFF (b) modify it to a modulo-8 ripple up-down counter

Solution 8 ripple up:

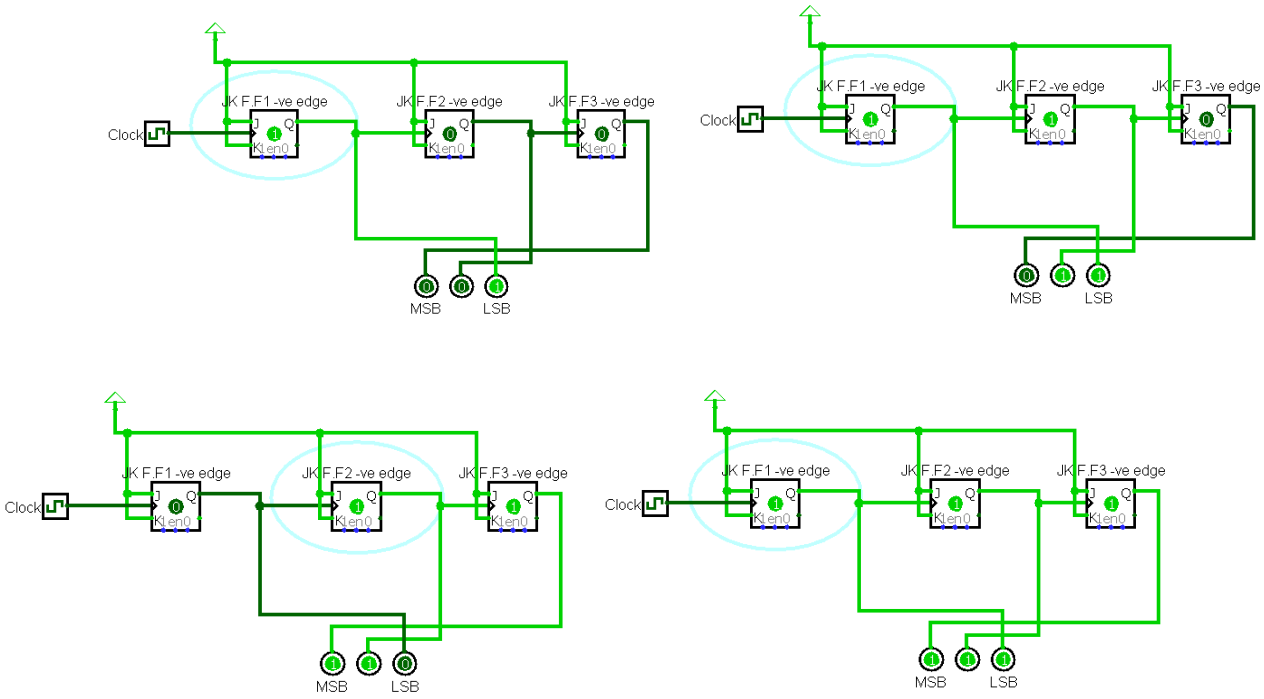
Design and explanation

We design the mod 8 counter using 3 JK FF in a cascade manner, the values of J and K are always one and the clocks are different for each flipflop, we give the output of the previous flipflop as the clock input to the next flipflop which halves the frequency, by doing this we can count up to 7. The value stored in FF1 is the LSB, and FF3 is the MSB

CIRCUIT



RESULT OBTAINED



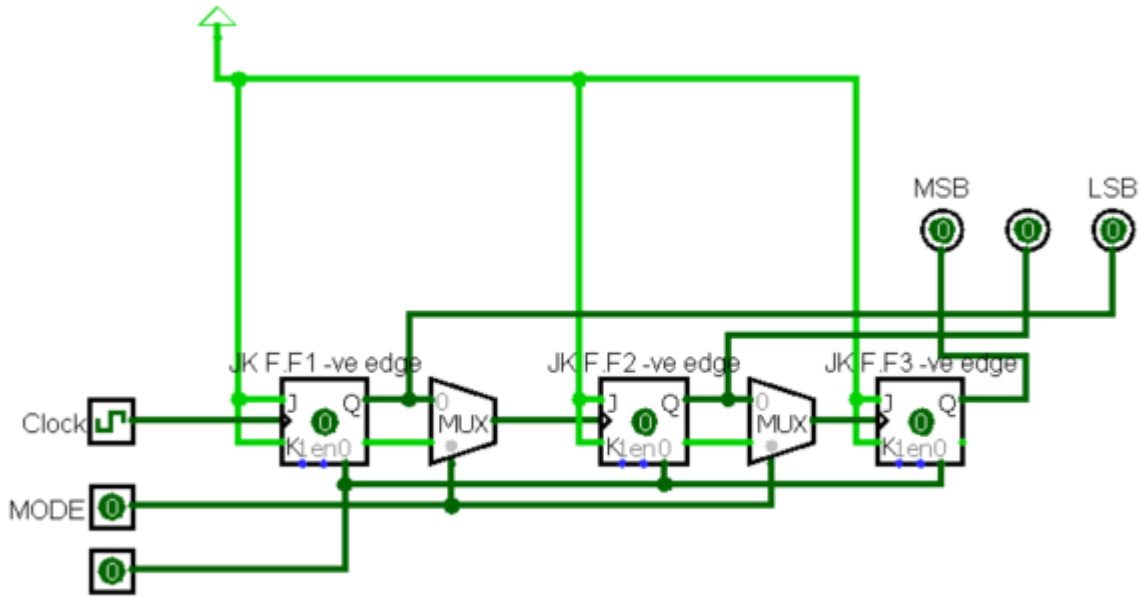
Counts upto 7 and repeats from 0 after that.

Solution modulo-8 ripple up-down counter:

Design and explanation

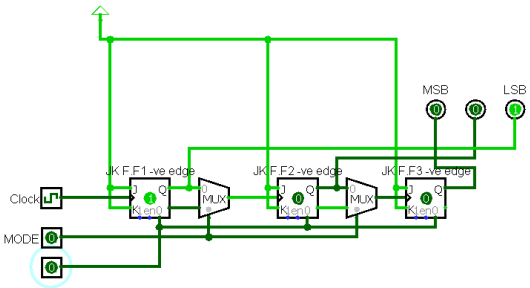
To design an up down counter we can use a Mode variable which when 0 does up counting and 1 does down counting. To perform down counting we just need to give the compliment of the clock to the next flip flop, now the counter counts from backwards when Mode is equal to 1.

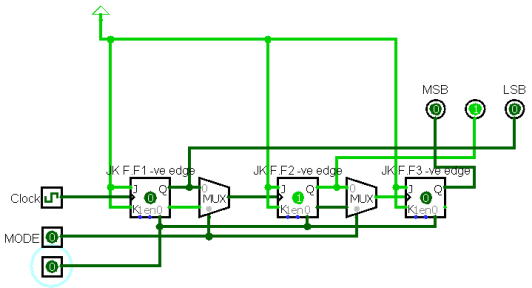
CIRCUIT

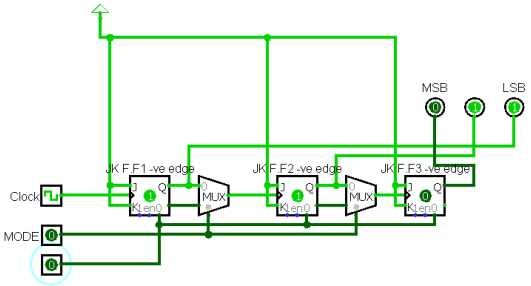


RESULT OBTAINED

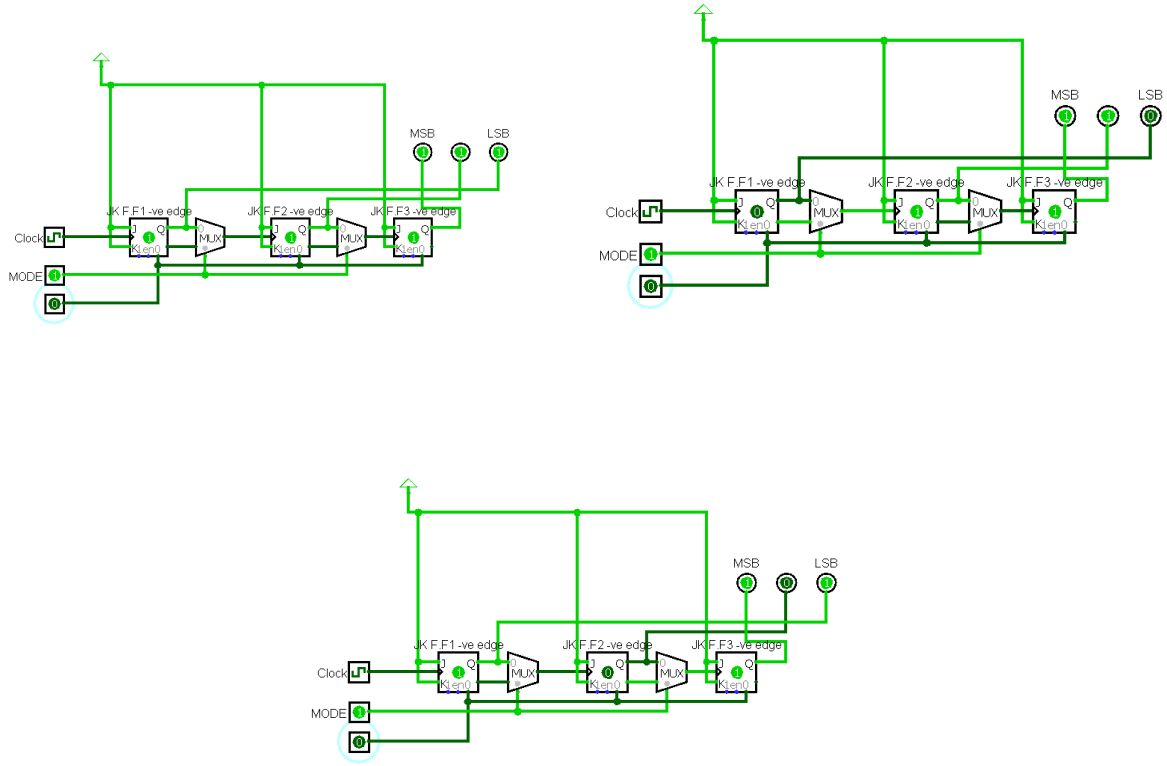
Up count







Down count



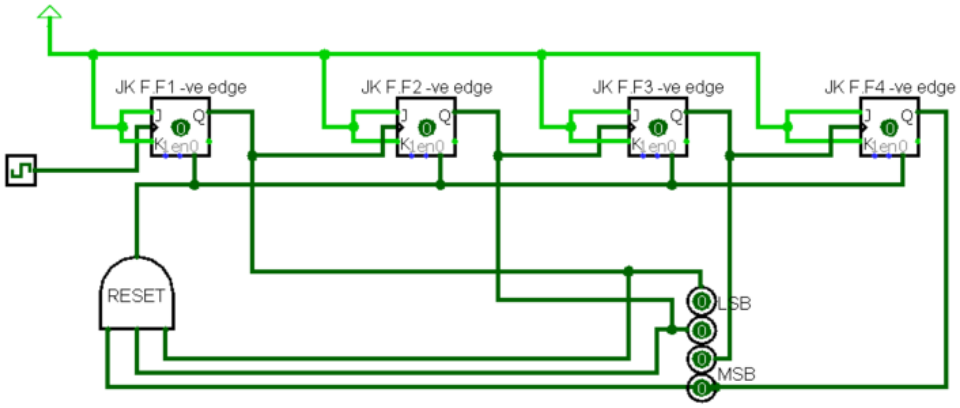
4) Design a modulo-11 ripple counter using JKFF

Solution:

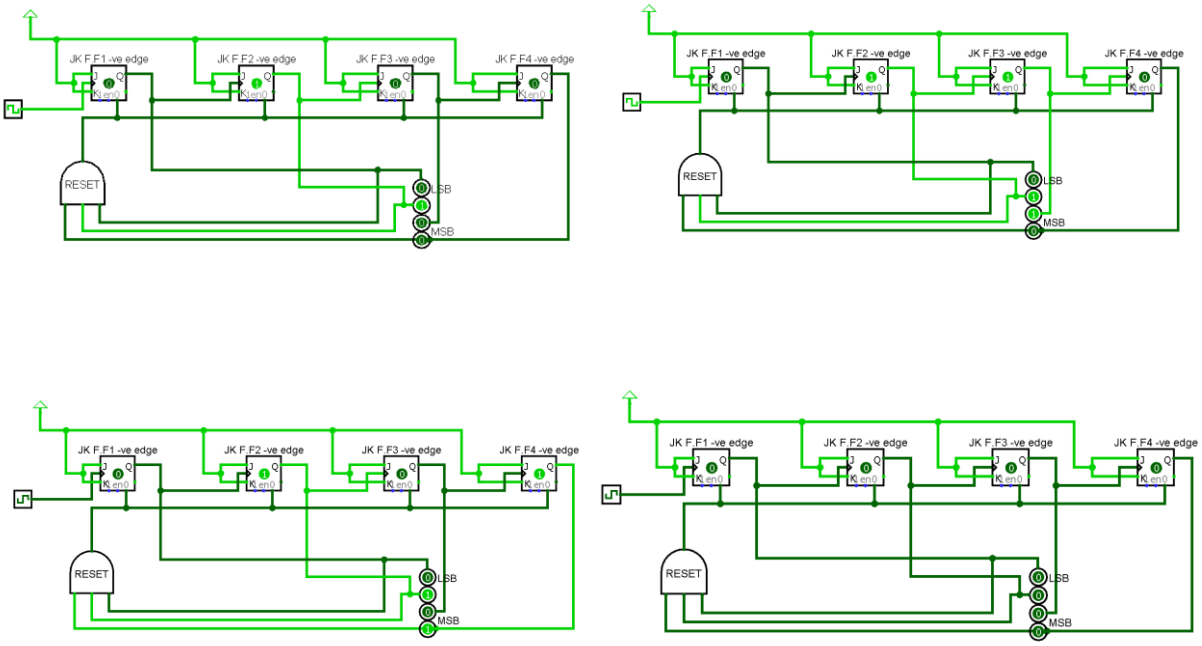
Design and explanation

As seen above we can make a 4-bit counter using 4 FF which count up to 15 and we can use asynchronous reset which will set the current state of the FF to 0. When the current state is (1011) that is 11 we need to reset the FF by using the hard reset.

CIRCUIT



RESULT OBTAINED



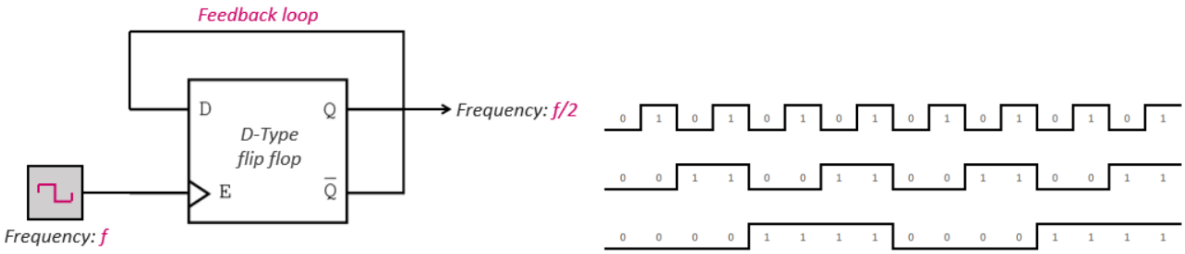
As we can see the counter is reset after 10 when the current state is 11 now we perform the asynchronous reset and start counting form 0 again.

5) Design a modulo-6 ripple counter using DFF and display the output using 7 segment LED

Solution:

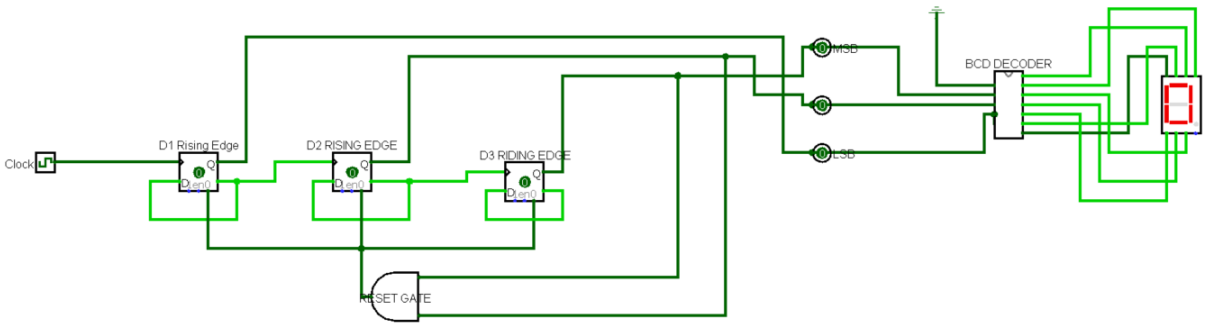
Design and explanation

We can implement the counter using D FF by creating a feedback loop (connecting the output Q to the Data pin, D) and applying a regular clock signal to the Enabler pin (E), the resulting signal (pin Q) has the frequency of the input signal divided by two.

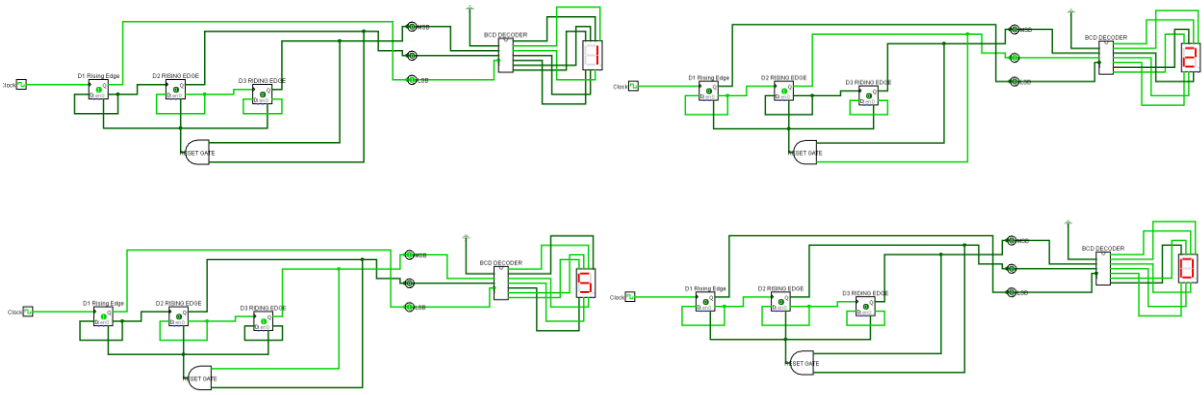


Now we can implement a counter which counts up to 15 using 3 FF and we can use the asynchronous reset at 110 (6) and start counting form zero. After that we just display the output in the 7 segment LED using the decoder which we implemented in Lab 1.

CIRCUIT

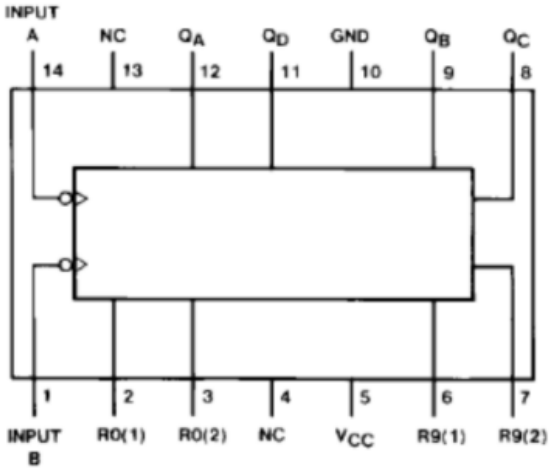


RESULT OBTAINED



6) Draw the circuit diagram to implement a counter that can count from 00-99 using decade counters 7490.

Solution:



Reset Inputs				Output			
R0(1)	R0(2)	R9(1)	R9(2)	Q_D	Q_C	Q_B	Q_A
H	H	L	X	L	L	L	L
H	H	X	L	L	L	L	L
X	X	H	H	H	L	L	H
X	L	X	L	COUNT			
L	X	L	X	COUNT			
L	X	X	L	COUNT			
X	L	L	X	COUNT			

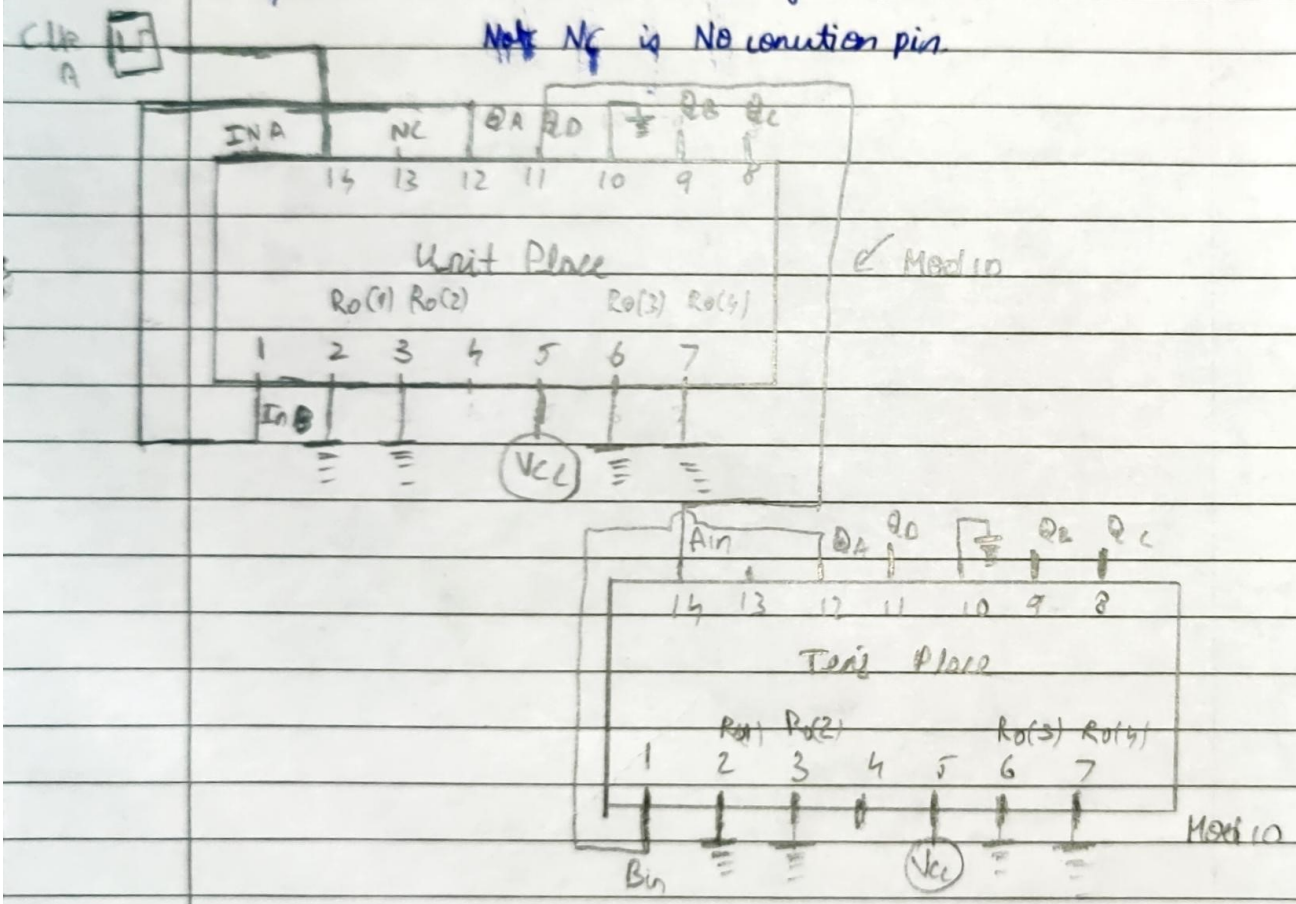
74 90 is ~~a decade counter~~ has two counters %2 and %5 counters.

Q_A → is the output of the 2% counter

Q_B Q_C Q_D is the output of 5% counter
(LSB) (MSB)

When we cascade mod m and mod n counters, the result is a mod m.n counter, using the above logic we can implement %100 counter as follows.

Note: NC is NO connection pin.



Tens place	Unit Place
Q_D Q_C Q_B Q_A	Q_D Q_C Q_B Q_A
MSB → LSB	MSB → LSB

⇒ We cascade 2 mod 10 counters and get a mod 100 counter.