North South University

Department of Electrical & Computer Engineering

Lab Report

Experiment No:	**03**
Experiment Title:	**Design of a 4-bit Binary Up-Down Synchronous Counter**
Course Code:	**CSE332L**
Course Name:	**Computer Organization & Architecture Lab**
Name & ID:	**Ishrat Jahan,1921909042**
Date of Experiment:	**11/11/2021**
Date of Submission:	**16/11/2021**

Objective:

1. To Construct a 4bit Binary Up-Down Synchronous Counter.

2. Understanding how an Up-Counter and Down-Counter works.

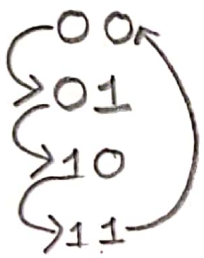
3. Learning to create T-Flip Flop from D-Flip-Flop.

4. Adjusting the mode of the Counter to make it work.

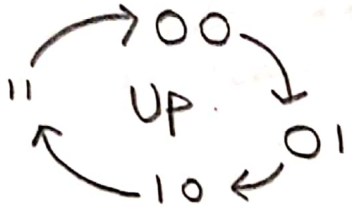
Theory

4-bit Binary Up-Down Synchronous Counter: It is a 4-bit counter that counts the binary number sequence, both up and down and we can give limitation to the count using this counter. Synchronous means all the flip-flops used in the circuit are clocked simultaneously so that the output change is related to each other when the mode is changed.

Up-Counter: It counts binary number sequence in increasing order. For a 2-bit up-counter:—

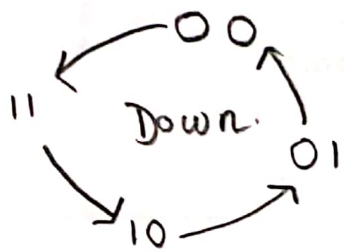


∴ After 00 it count 01, then 10, then 11 and again it's back to 00.



Down-Counter: It counts binary number sequence in decreasing order. For a 2-bit down-counter:

∴ From 11 it counts to 10, then 01, then 00 and then back to 11.



Up-down Counter: Combination of a up-counter and a down counter.

Mode: It is the selection option for controlling when the counter will behave as an up-counter or a down-counter.
We usually use: 0 → Up Counter
1 → Down Counter.

Flip-Flop: We need flip-flops for counter design because we need to know the initial state and the state we want the counter to go.
So, these requires memorization of initial state.
The number of flip-flop we use depends on the bits of the counter we are making.
For a 2-bit counter we need 2 Flip-flop, likewise for a 4-bit counter we need 4 flip-flops.

T-flip-flop: It is the toggle flip flop. It toggle's the input we provide.

"Characteristic Table"

T	$Q(t+1)$
0	$Q(t)$ [No change]
1	$Q'(t)$ [Invert]

$\therefore Q(t+1) = Q_{next}$
$\therefore Q(t) =$ Initial state

According to the characteristic table of T-Slip Slop when the input in it is 1, it toggles the value of Q(t). and when it is 0, the next state is same as present state.

Q(t) [Present State]	Q(t+1) [Next state]	T
0	0	0
0	1	1
1	0	1
1	1	0

"Excitation Table"

D-Slip-Slop: Data Slip-Slop. In this Slip-Slop. the present state and the next state is same.

D	Q_{next}
0	0
1	1

'Characteristic Table'

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

"Excitation Table"

Creating T-FlipFlop using D-flip flop

For doing this we need to figure out what input of D-flip flop will make it behave as T-flip-flop.

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

Q	T	D
0	0	0
0	1	1
1	1	0
1	0	1

$Q \oplus T = D$.
$Q \oplus D = T$

If we combing Q with T, the results of D is similar to an XOR table. Therefore, if we add an XOR gate to a T-flip-flop and then input the inputs through it, it will act as a D-flip flop.

A	B	
0	0	0
0	1	1
1	0	1
1	1	0

∴ XOR TRUTH TABLE.

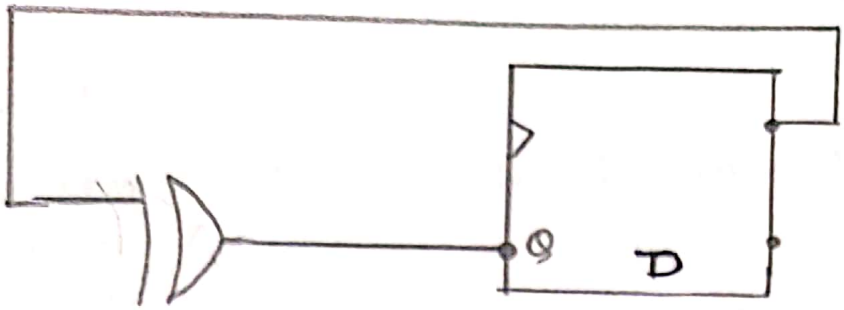


Fig: T flip-flop using D flip-flop

IC:

7404: NOT GATE
7408: AND GATE
7432: OR GATE
7486: XOR GATE
2 x 7474: Dual D flip-flop.

K-Map:- 4-variable K-Map.

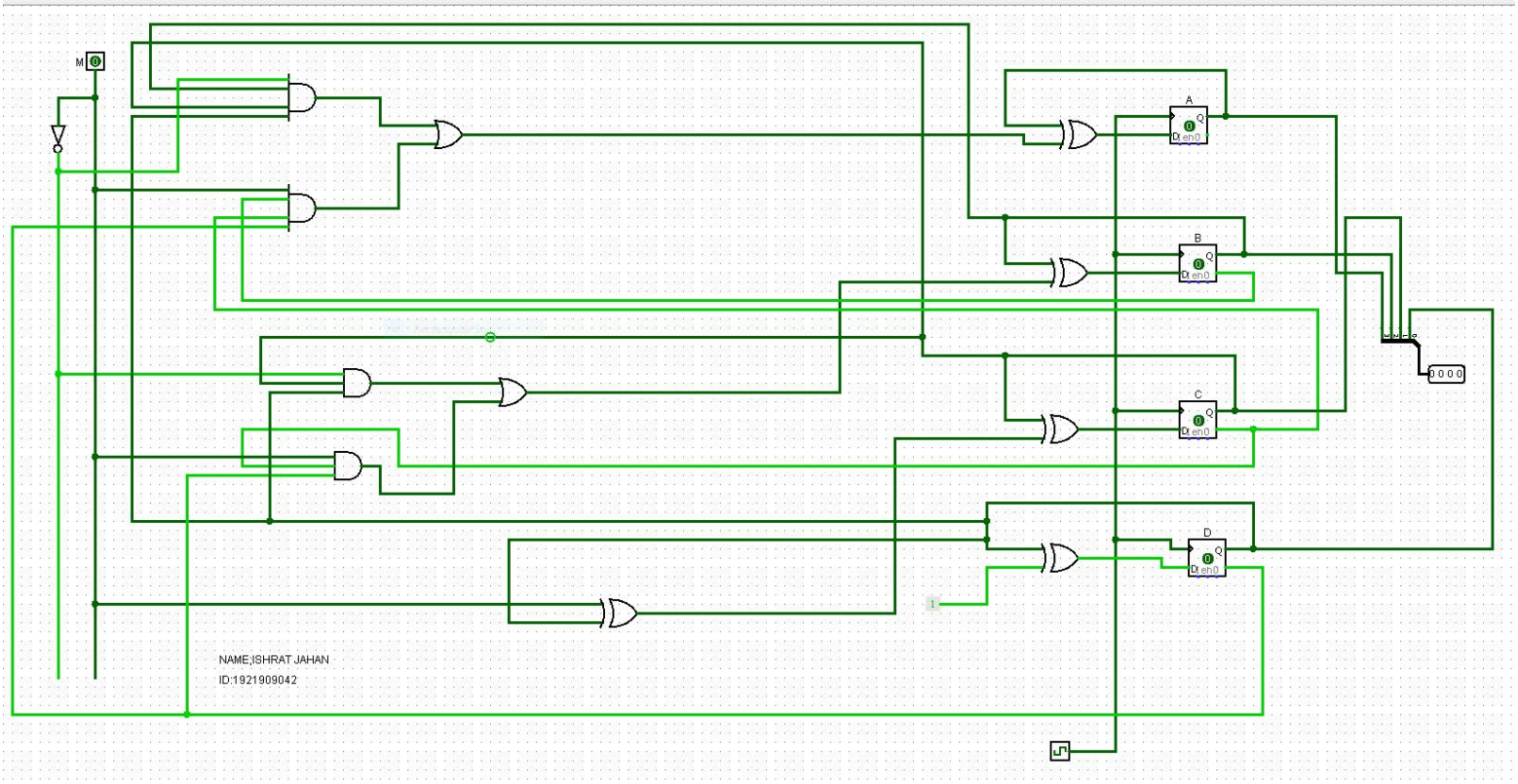
Variables: M, A, B, C, D.

AB \ CD	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

M=0

AB \ CD	00	01	11	10
00	16	20	28	24
01	17	21	29	25
11	19	23	31	27
10	18	22	30	26

M=1


M
A
B
C
D
0 0 0 0
NAME;ISHRAT JAHAN
ID:1921909042

TRUTH TABLE

M	A	B	C	D
0	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
0	1	0	1	1
0	1	1	0	0
0	1	1	0	1
0	1	1	1	0
0	1	1	1	1
1	0	0	0	0
1	0	0	0	1
1	0	0	1	0
1	0	0	1	1
1	0	1	0	0
1	0	1	0	1
1	0	1	1	0
1	0	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

M: Mode

Present State: A B C D

A+	B+	C+	D+
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1
0	0	0	0
1	1	1	1
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0

A+,B+,C+,D+ =Next State

Ta	Tb	Tc	Td
0	0	0	1
0	0	1	1
0	0	0	1
0	1	1	1
0	0	0	1
0	0	1	1
0	0	0	1
1	1	1	1
0	0	0	1
0	0	1	1
0	0	0	1
0	1	1	1
0	0	0	1
0	0	1	1
0	0	0	1
1	1	1	1
1	1	1	1
0	0	0	1
0	0	1	1
0	0	0	1
0	1	1	1
0	0	0	1
0	0	1	1
0	0	0	1
1	1	1	1
0	0	0	1
0	0	1	1
0	0	0	1
0	1	1	1
0	0	0	1
0	0	1	1

Flip-flop Outputs

K-MAP for Ta

AB/CD	00	01	11	10
00				
01				
11		1	1	
10				

M=0

AB/CD	00	01	11	10
00	1			1
01				
11				
10				

M=1

Ta=M'BCD+MB'C'D'

K-MAP for Tb

AB/CD	00	01	11	10
00				
01				
11	1	1	1	1
10				

M=0

AB/CD	00	01	11	10
00	1	1	1	1
01				
11				
10				

M=1

Tb=M'CD+MC'D'

K-MAP for Tc

AB/CD	00	01	11	10
00				
01	1	1	1	1
11	1	1	1	1
10				

M=0

AB/CD	00	01	11	10
00	1	1	1	1
01				
11				
10	1	1	1	1

M=1

Tc=M'D+MD'

K-MAP for Td

AB/CD	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

M=0

AB/CD	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

M=1

Td=1

Discussion:

The experiment of Lab08 was based on "Design of a 4 bit-Binary Up-Down Synchronous Counter." In this lab we learned how a up and down counter works simultaneously and how we can construct a synchronous up-down counter of different bits.

At first, we studied the concept of a counter. We learned how a counter counts binary numbers of different bits. We learnt up-down counter means it counts the binary sequence both in up direction and downward direction. To clear our understanding, we were shown the up-counting and Down counting for 2 bits. For a two bit number, we get 4 combinations: 00, 01, 10, 11. For up-count, if it starts from 00, it goes to 01, then to 10, then to 11 and back to 00 again. For down count, it counts in decreasing order. If it starts from 00, it will go to 11, then to 10, then to 01 and back to 00 again. So, the sequence that counts binary number is the Up-Counter and the one that counts in decreasing order is the Down-Counter. Secondly, we learned how we can change the mode for making the controller behave as an Up-Counter and Down-Counter as per our wants. When the mode=0, it behaves as an Up-Counter and while it is 1, it behaves as an Down-Counter.

After that we learnt how we can design a counter and what are needed in Logisim to design it. As the counter changes state; It jumps from one binary number combination to another according to the type of counter selected, it requires

something that will remember the initial state it was in. If it doesn't remember the initial state, it wouldn't know where to go next. The memorization of initial states or any states can only be done by Flip-flops. So we learned, why we need to use Flip-flops for counter design. Moreover, if the counter is 2 bit we use 2 Flip-flop and if it is 4 bit, we will use 4-Flip-flops. As per our lab manual, we were expected to make T-flip-flop from D-flip-flop and use it. This was because, T-flip-flops are expensive and rarely available. So we convert D-flip-flops to T-flip-flops.

After these, our next topic was Flip-flops and how we will convert it. For that, we started with T-flip flop. As it's name suggests, "Toggle Flip-Flop", it toggles the initial state. As per the characteristic table of T-flip-flop, when $T=0$, $Q_{next} = Q(t)$ where $Q(t)$ is the initial state and when $T=1$, $Q_{next} = Q'(t)$. Then we drew the excitation table of T-flip-flop for 2 bits. Then we moved on to D-flip-flop. According to the characteristic table of D-flipflop, when $D=0$, $Q_{next}=0$, and when $D=1$, $Q_{next}=1$. Therefore it's known as data flip-flop as it just forward the data. After that, we just combined the T and D values, with all four combinations accordingly and examined them. We drew a table comprising, Q, T, and D. As per the table we noticed, if we XOR, $D \oplus Q$ the D-flip-flop behaves as T-flip flop. So, we just needed to add an XOR gate to D-flip and XOR the input ~~with~~ to the gate with Q from the Flip-flop.

Lastly, we drew our truth table for the 2-bit counter. As it's an Up-Down counter, there were 8 combination in total. 4 were for M=0, and four for M=1. We completed the whole table according to our knowledge of how Up and down counter works. Then we compared the inputs of the present state and next state and filled out T_A and T_B. After that we used our knowledge of K-Maps to find the equations of T_A and T_B. As they were 5 variable K-Map we used two four variable K-Map, one for M=0 and the other for M=1 to equate the equation. We solved the K-Maps and were all set. All these were done by us simultaneously following our lab instructor. So we were cleared of everything we need to make Up-Down Counter, and were instructed to draw the circuit of 4-bit Binary Up-Down Counter. for our lab Performance

Firstly, I proceeded to draw the truth table. As it was 4-bit counter, it will have 16 combination for Up-counter and 16 combinations for down counter. In total there were 32 rows in the truth table. The initial states were A, B, C, D and M was the mode. The next state were A+, B+, C+, D+. The Flip-flop results were T_A, T_B, T_C, T_D. I carefully solved the truth table, first for the Up-Counter and then for the down Counter. Then I compared the values of each input in present state to it's next state input and filled the columns T_A, T_B, T_C, T_D.

After that, I drew the K-Maps and worked out the equations for T_A, T_B, T_C, T_D. After completion of this, I proceeded to draw the circuit in Logisim.

In Logisim, firstly I opted to make the mode button by which I will control the counter. I took a input pin, labelled it 'M', changed its facing and drew a vertical line from the input pin. This line is "M". Then I took a not gate and placed it beside 'M' line and joined it's input pin to the M line. This creates "M'". Then I proceeded to create the T-flip-flop. As it's a 4-bit counter, I took 4-D-flip-flop. The D- portion of each flip-flop was connected with an XOR gate from the gates section. The XOR gates had two inputs. One of them is the equations of $T_A/T_B/T_C/T_D$ and the other is supposed to be Q. So, I connected Q ~~portion~~, by drawing a line from each flip-flop to the XOR gate of each of them. Then from the Wiring section I took a clock, and joined the same clock with all of the four-flip-flops. They have the same clock because the counter is supposed to be synchronous. All the flip-flops were labelled as A, B, C, D. Then I proceeded to draw T_A, T_B, T_C, T_D as per the equations I got. I drew the circuit for all of them, varying the number of inputs as required and selecting gates from the Gates section as need. I connected all by final circuit of T_A, T_B, T_C, to the other input of the XOR gate. As the $T_D = 1$, so I took a constant from ~~wi~~ wiring section and

connected it with the XOR gate of the last flip-flop.

For displaying the output, which was 4 bit, I took an output pin, changed it's data bits and connected a splitter with it. Then I took the output's from each flip-flop and connected it to the splitter. The A flip-flop being the MSB was connected to the splitter's B part and D being the LSB was connected with 0. Finally after completion, I checked the outputs by varying M and they matched. The circuit worked correctly.

The problem that I faced during this experiment was that, the transition from one state to another state of binary number was too fast for me to catch due to weak eyesight. So, I had to draw the circuit thrice as everytime I checked values it didn't match. But everytime the circuit was correct and it was my eyesight. And I ended up submitting the work really late. Other than that, it takes time, as there were 32 combinations in the truth table to solve. and then draw K-Maps. Other than that, the circuit was easy to built.