

DIGITAL SYSTEM DESIGN LAB

LAB #02



Spring 2021

CSE308L DSD LAB

Submitted by: **Shah Raza**

Registration No. : **18PWCSE1658**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: _____

Submitted to:

Engr. Madiha Sher

Wednesday, April 28th, 2021

Department of Computer Systems Engineering
University of Engineering and Technology, Peshawar

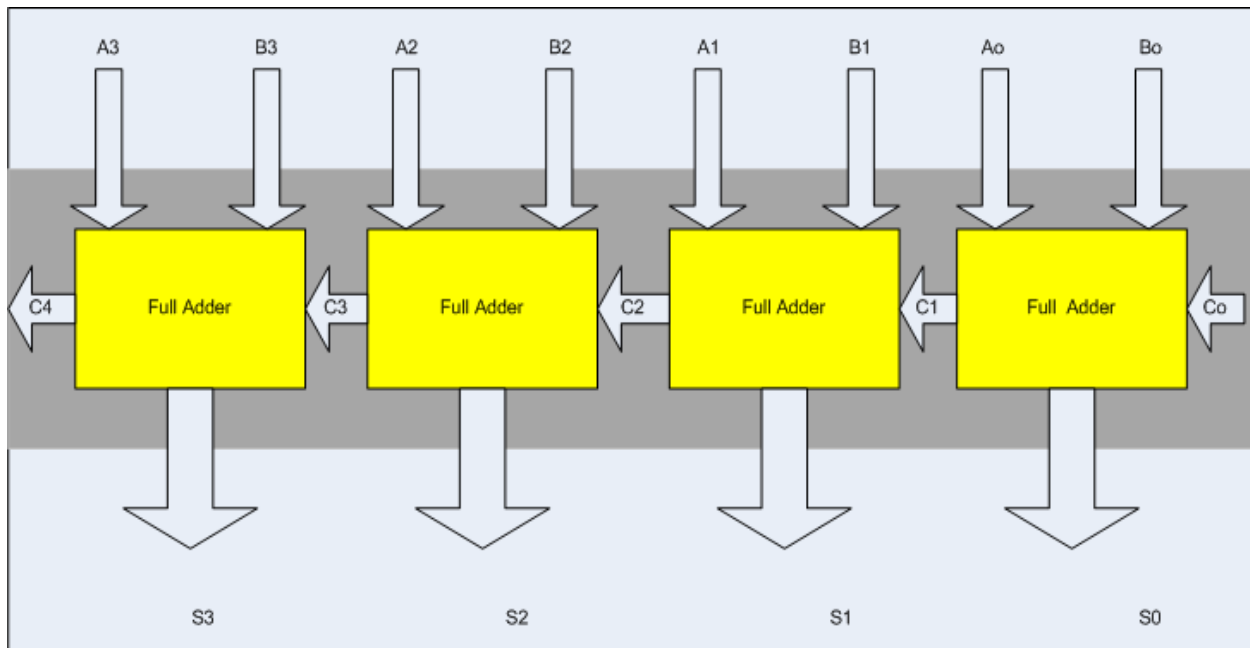
Objectives:

This lab will enable students to:

- Learn top down and bottom up design methodologies
- Data flow level modeling

Task # 01:

1. First implement a Full adder using data gate level modeling.
2. Simulate the Full adder with a test bench.
3. Instantiate the Full adder four times and connect the circuit as shown.
4. Now again write a test bench and simulate the 4 bit RCA.



Problem Analysis:

Truth Table:

Full Adder:

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

Simplified Expressions:

$S = A \oplus B \oplus C$

$C_{out} = AB + BC + AC$

Code:

Half Adder:

```
module HalfAdder(Sum,Cout,A,B);
    input A,B;
    output Sum,Cout;

    xor (Sum,A,B);
    and (Cout,A,B);
endmodule
```

Test Bench for Half Adder:

```
module testHalfAdder;
    reg A,B;
    wire Sum,Cout;

    HalfAdder ha(Sum,Cout,A,B);

    initial begin
        $display("A B S C");
        $monitor("%b %b %b %b",A,B,Sum,Cout);
        A = 0; B=0;
        #5
        A = 0; B=1;
        #5
        A = 1; B=0;
        #5
        A = 1; B=1;
    end
endmodule
```

Full Adder:

```
module FullAdder(Sum,Cout,A,B,Cin);
    input A,B,Cin;
    output Sum,Cout;
    wire S,C,C1;

    HalfAdder HA1(S,C,A,B);
    HalfAdder HA2(Sum,C1,S,Cin);
    or (Cout,C,C1);
endmodule
```

Test Bench for Full Adder:

```
module testFullAdder;
    reg A,B,Cin;
    wire Sum,Cout;

    FullAdder fa(Sum,Cout,A,B,Cin);

    initial begin
        $display("A B Cin S Cout");
        $monitor("%b %b %b %b %b",A,B,Cin,Sum,Cout);
        A=0;B=0;Cin=0;
        #5
        A=0;B=1;Cin=0;
        #5
        A=1;B=0;Cin=0;
        #5
        A=1;B=1;Cin=0;
        #5
        A=0;B=0;Cin=1;
        #5
        A=0;B=1;Cin=1;
        #5
        A=1;B=0;Cin=1;
        #5
        A=1;B=1;Cin=1;
    end
endmodule
```

4-bit Ripple Carry Adder:

```
module fourBitAdder(Sum,Cout,A,B);
    input [3:0] A,B;
    wire [2:0] Cin;
    output [3:0] Sum;
    output Cout;

    FullAdder FA1(Sum[0],Cin[0],A[0],B[0],1'b0);
    FullAdder FA2(Sum[1],Cin[1],A[1],B[1],Cin[0]);
    FullAdder FA3(Sum[2],Cin[2],A[2],B[2],Cin[1]);
    FullAdder FA4(Sum[3],Cout,A[3],B[3],Cin[2]);
endmodule
```

Test Bench for 4-bit Ripple Carry Adder:

```
module test4bitAdder;
    reg [3:0] A,B;
    wire [3:0] Sum;
    wire Cout;

    fourBitAdder A4(Sum,Cout,A,B);

    initial begin
        $display("A B S Cout");
        $monitor("%d %d %d %d",A,B,Sum,Cout);
        A= 4'b0000;B=4'b0000;
        #5
        A= 4'b0000;B=4'b0001;
        #5
        A= 4'b0000;B=4'b0010;
        #5
        A= 4'b0000;B=4'b0011;
        #5
        A= 4'b0010;B=4'b0011;
        #5
        A= 4'b0011;B=4'b0011;
        #5
        A= 4'b0100;B=4'b0010;
        #5
        A= 4'b1100;B=4'b0001;
        #5
        A= 4'b1011;B=4'b1011;
        #5
        A= 4'b1110;B=4'b1011;
        #5
        A= 4'b1111;B=4'b1011;
        #5
        A= 4'b1111;B=4'b1110;
        #5
        A= 4'b1111;B=4'b1111;
    end
endmodule
```

Output:

```

# A B S Cout
# 0 0 0 0
# 0 1 1 0
# 0 2 2 0
# 0 3 3 0
# 2 3 5 0
# 3 3 6 0
# 4 2 6 0
# 12 1 13 0
# 11 11 6 1
# 14 11 9 1
# 15 11 10 1
# 15 14 13 1
# 15 15 14 1

```

Waveform:

At 25ns:

/test4bitAdder/A	0011	0000					0010	0011	0100	1100	1011	1110	1111		
/test4bitAdder/B	0011	0000	0001	0010	0011				0010	0001	1011			1110	1111
/test4bitAdder/Sum	0110	0000	0001	0010	0011	0101	0110	0110	1101	0110	1001	1010	1101	1110	
/test4bitAdder/Cout	0														

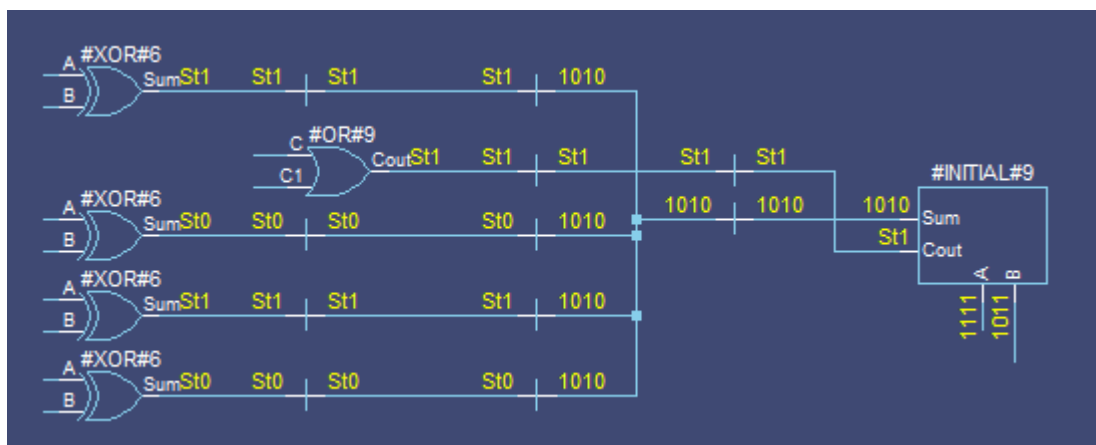
At 35ns:

/test4bitAdder/A	1100	0000					0010	0011	0100	1100	1011	1110	1111		
/test4bitAdder/B	0001	0000	0001	0010	0011				0010	0001	1011			1110	1111
/test4bitAdder/Sum	1101	0000	0001	0010	0011	0101	0110	0110	1101	0110	1001	1010	1101	1110	
/test4bitAdder/Cout	0														

At 50ns:

/test4bitAdder/A	1111	0000					0010	0011	0100	1100	1011	1110	1111		
/test4bitAdder/B	1011	0000	0001	0010	0011				0010	0001	1011			1110	1111
/test4bitAdder/Sum	1010	0000	0001	0010	0011	0101	0110	0110	1101	0110	1001	1010	1101	1110	
/test4bitAdder/Cout	1														

Dataflow:



Task # 02:

Design the 4 bit full adder using data flow level modeling.

Code:

Full Adder:

```
module FA(Sum,Cout,A,B,Cin);
    input A,B,Cin;
    output Sum,Cout;

    assign {Cout,Sum} = A+B+Cin;
endmodule
```

Test Bench for Full Adder:

```
module testFullAdder;
    reg A,B,Cin;
    wire Sum,Cout;

    FullAdder fa(Sum,Cout,A,B,Cin);

    initial begin
        $display("A B Cin S Cout");
        $monitor("%b %b %b %b %b",A,B,Cin,Sum,Cout);
        A=0;B=0;Cin=0;
        #5
        A=0;B=1;Cin=0;
        #5
        A=1;B=0;Cin=0;
        #5
        A=1;B=1;Cin=0;
        #5
        A=0;B=0;Cin=1;
        #5
        A=0;B=1;Cin=1;
        #5
        A=1;B=0;Cin=1;
        #5
        A=1;B=1;Cin=1;
    end
endmodule
```

4-bit RCA:

```
module FBA(Sum,Cout,A,B);
    input [3:0] A,B;
    output [3:0] Sum;
    output Cout;
    wire [2:0] Cin;

    FA f1(Sum[0],Cin[0],A[0],B[0],1'b0);
    FA f2(Sum[1],Cin[1],A[1],B[1],Cin[0]);
    FA f3(Sum[2],Cin[2],A[2],B[2],Cin[1]);
    FA f4(Sum[3],Cout,A[3],B[3],Cin[2]);
endmodule
```

Test Bench for 4-bit RCA:

```
module testFBA;
    reg [3:0] A,B;
    wire [3:0] Sum;
    wire Cout;

    FBA A4(Sum,Cout,A,B);





    initial begin
        $display("A B S Cout");
        $monitor("%d %d %d %d",A,B,Sum,Cout);
        A= 4'b0000;B=4'b0000;
        #5
        A= 4'b0000;B=4'b0001;
        #5
        A= 4'b0000;B=4'b0010;
        #5
        A= 4'b0000;B=4'b0011;
        #5
        A= 4'b0010;B=4'b0011;
        #5
        A= 4'b0011;B=4'b0011;
        #5
        A= 4'b0100;B=4'b0010;
        #5
        A= 4'b1100;B=4'b0001;
        #5
        A= 4'b1011;B=4'b1011;
        #5
        A= 4'b1110;B=4'b1011;
        #5
        A= 4'b1111;B=4'b1011;
        #5
        A= 4'b1111;B=4'b1110;
        #5
        A= 4'b1111;B=4'b1111;
    end
endmodule
```


Output:





```
# A B S Cout
# 0 0 0 0
# 0 1 1 0
# 0 2 2 0
# 0 3 3 0
# 2 3 5 0
# 3 3 6 0
# 4 2 6 0
# 12 11 3 0
# 11 11 6 1
# 14 11 9 1
# 15 11 10 1
# 15 14 13 1
# 15 15 14 1
```

Waveform:





At 25ns:

 /test4bitAdder/A	0011	0000				0010	0011	0100	1100	1011	1110	1111		
 /test4bitAdder/B	0011	0000	0001	0010	0011			0010	0001	1011			1110	1111
 /test4bitAdder/Sum	0110	0000	0001	0010	0011	0101	0110	0110	1101	0110	1001	1010	1101	1110
 /test4bitAdder/Cout	0													

At 35ns:

 /test4bitAdder/A	1100	0000				0010	0011	0100	1100	1011	1110	1111		
 /test4bitAdder/B	0001	0000	0001	0010	0011			0010	0001	1011			1110	1111
 /test4bitAdder/Sum	1101	0000	0001	0010	0011	0101	0110	0110	1101	0110	1001	1010	1101	1110
 /test4bitAdder/Cout	0													

At 50ns:

 /test4bitAdder/A	1111	0000				0010	0011	0100	1100	1011	1110	1111		
 /test4bitAdder/B	1011	0000	0001	0010	0011			0010	0001	1011			1110	1111
 /test4bitAdder/Sum	1010	0000	0001	0010	0011	0101	0110	0110	1101	0110	1001	1010	1101	1110
 /test4bitAdder/Cout	1													

Dataflow:

