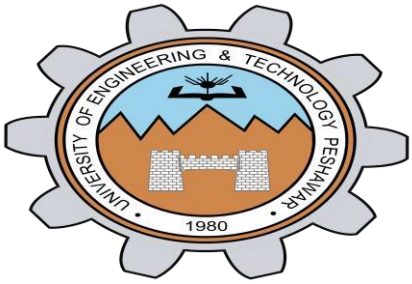
Lab report no 2



Name: Muhammad Ali

Reg no: 19pwcse1801

Section: A

Submitted to: Eng Mam Madiha Sher

Objectives:

This lab will enable students to:

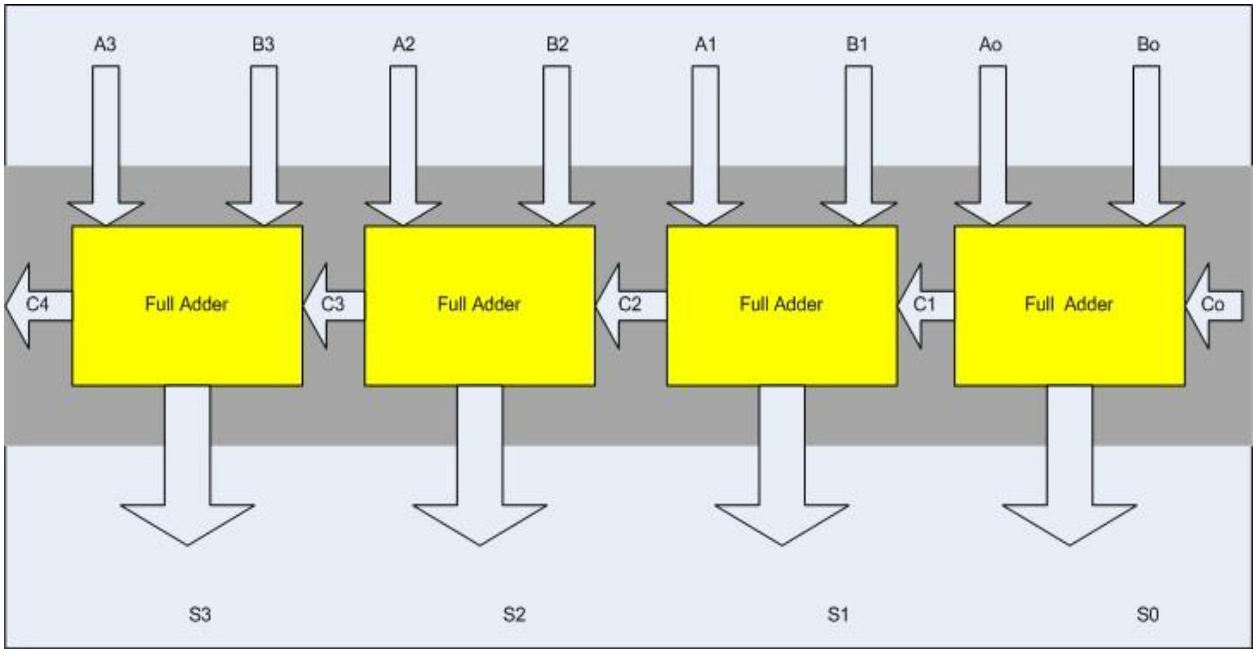
- Learn top down and bottom up design methodologies
- Data flow level modeling

TASK 1:

4 bit Ripple Carry Adder

Block Diagram:

Following is the block diagram of a 4 bit Ripple Carry Adder.



I/O Connection:

Ground “Co” Permanently and connect S0-S3 with four LEDs also connect C4 to another LED.

Steps for task 1a:

The following steps should be performed while designing a 4 bit RCA adder

ModelSim:

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.

Xilinx:

1. Make new project in Xilinx and add the files that you simulated in ModelSim.
2. Add User Constraint File inputs should be locked with the switches, C0 should be permanently "0" while S0-S4 and C4 with LEDS

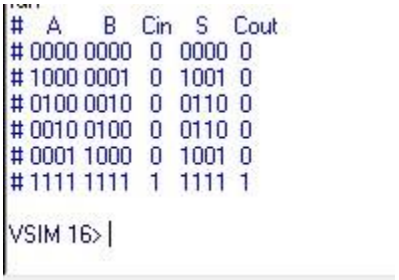
Code:

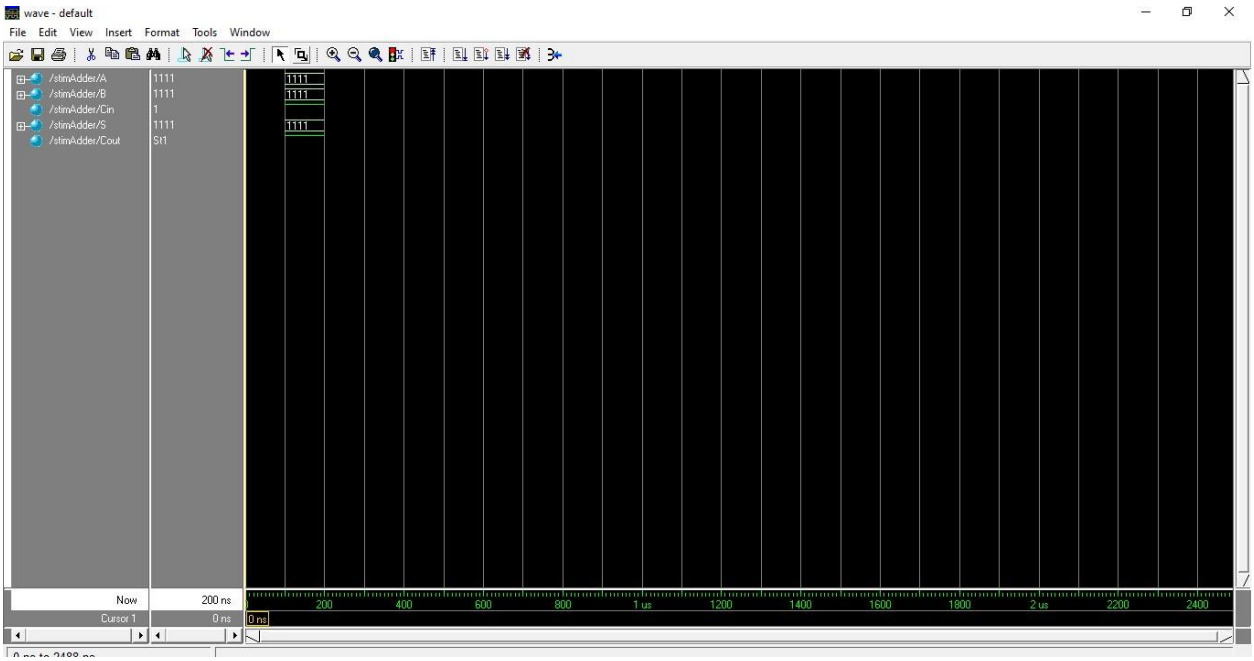
s

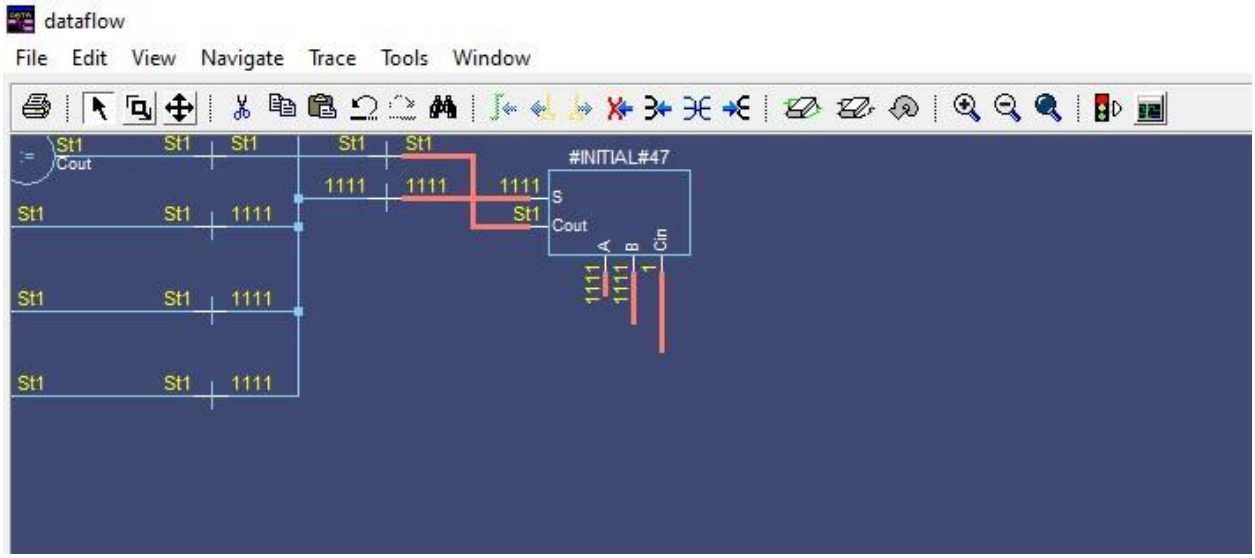
```
module sum(S, A, B);
        output S;
input A, B;     xor
x1(S, A, B);
endmodule

module carry(C, A, B);
        output C;
input A, B;     and
a1(C, A, B);
endmodule

module halfadder(S, C, A, B);
output S, C;    input A, B;
sum s1(S, A, B);        carry
c1(C, A, B);
endmodule

module fulladder(S, Cout, A, B, Cin);
output S, Cout;         input A, B,
Cin;    wire c1, c2, s1;
halfadder HA1(s1, c1, A, B);
halfadder HA2(S, c2, s1, Cin);
or o1(Cout, c1, c2);
endmodule
```

```
module fourbitAdder(S, Cout, A, B, Cin);
output [3:0] S;          output Cout;
input Cin;       input [3:0] A, B;
        wire C1, C2, C3;
        fulladder FA0(S[0], C1, A[0], B[0], Cin);
fulladder FA1(S[1], C2, A[1], B[1], C1);      fulladder
FA2(S[2], C3, A[2], B[2], C2);         fulladder FA3(S[3],
Cout, A[3], B[3], C3); endmodule

module stimAdder();
reg [0:3] A, B;
reg Cin;         wire
[0:3] S;
        wire Cout;
        fourbitAdder f1(S, Cout, A, B, Cin);
        initial begin
                $display("  A      B    Cin    S    Cout");
                A = 0; B = 0; Cin = 0;
                $monitor("%b %b   %b   %b  %b", A, B, Cin, S, Cout);
                #10 A = 8; B = 1; Cin = 0;
                #10 A = 4; B = 2; Cin = 0;
                #10 A = 2; B = 4; Cin = 0;
                #10 A = 1; B = 8; Cin = 0;
#10 A = 15; B = 15; Cin = 1;
        end
endmodule
```

Output:







Task 1b:

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

Code:

```
module halfadder(S, C, A, B);
        output S, C;    input
A, B;   assign {C, S} = A +
B; endmodule
```

```
module fulladder(S, Cout, A, B, Cin);
        output S, Cout;         input
A, B, Cin;      wire c1, c2, s1;
halfadder HA1(s1, c1, A, B);
halfadder HA2(S, c2, s1, Cin);
assign Cout = c1 | c2; endmodule

module fourbitAdder_DF(S, Cout, A, B, Cin);
        output [3:0] S;
output Cout;    input
Cin;    input [3:0] A, B;
        wire C1, C2, C3;
        fulladder FA0(S[0], C1, A[0], B[0], Cin);
fulladder FA1(S[1], C2, A[1], B[1], C1);      fulladder
FA2(S[2], C3, A[2], B[2], C2);        fulladder FA3(S[3],
Cout, A[3], B[3], C3); endmodule

module stim_DF_RCA();
reg [0:3] A, B;
reg Cin;        wire [0:3]
S;
        wire Cout;
        fourbitAdder_DF d1(S, Cout, A, B, Cin);
initial begin
                $display("  A     B   Cin   S   Cout");
                A = 0; B = 0; Cin = 0;
                $monitor("%b %b   %b   %b  %b", A, B, Cin, S, Cout);
                #5 A = 8; B = 1; Cin = 0;
                #5 A = 4; B = 2; Cin = 0;
                #5 A = 2; B = 4; Cin = 0;
                #5 A = 1; B = 8; Cin = 0;
        #5 A = 15; B = 15; Cin = 1;   end
endmodule
```

Output:

```
run
#   A     B   Cin   S    Cout
# 0000 0000  0  0000  0
# 1000 0001  0  1001  0
# 0100 0010  0  0110  0
# 0010 0100  0  0110  0
# 0001 1000  0  1001  0
# 1111 1111  1  1111  1

VSIM 21>
```