Lab Assignment – 2

Mumuksh Jain – 22110160

Nishant Kumar – 22110170

1. Implement a 1-bit full adder design using structural code. Write a testbench and show the simulation results for all cases.

Code:

```
`timescale 1ns / 1ps

module adder1bit(a,b,cin,sum,carry);
input a,b,cin;
output carry,sum;

xor(f1,a,b);
xor(sum,f1,cin);

and(f2,a,b);
and(f3,f1,cin);

or(carry,f3,f2);
endmodule
```

Testbench:

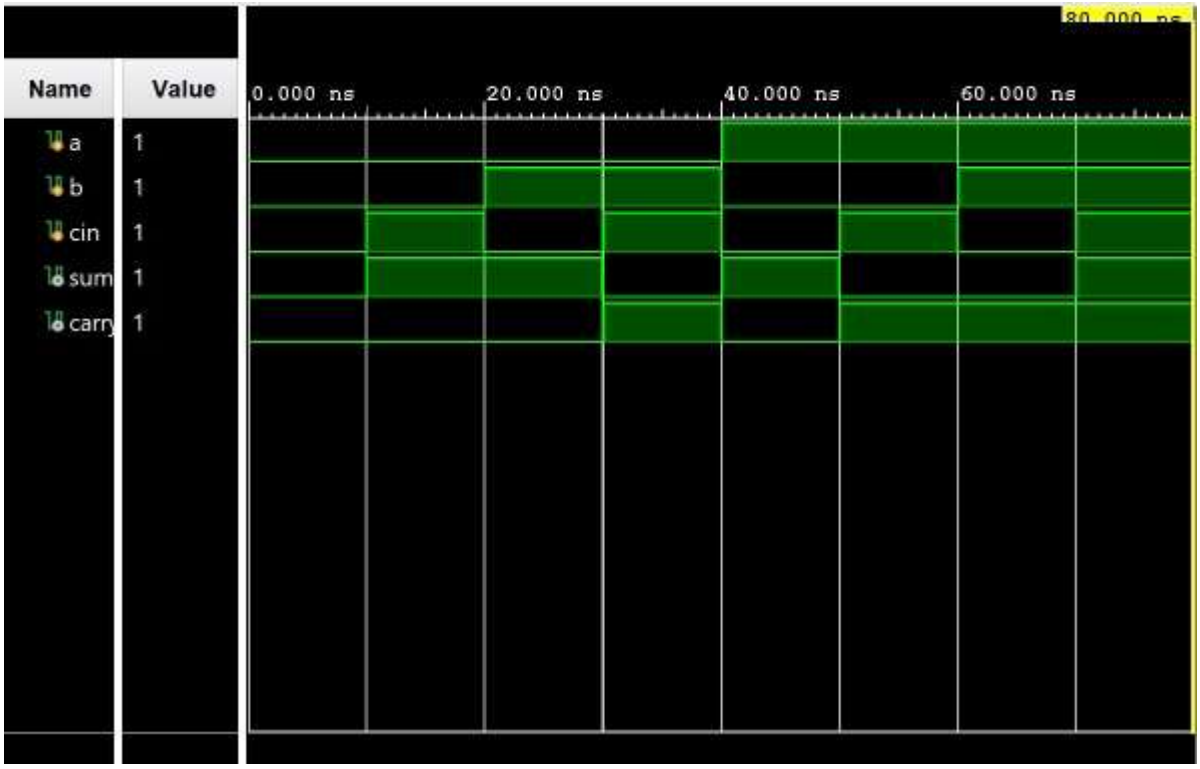
```
`timescale 1ns / 1ps

module adder1bit_tb();
reg a,b,cin;
wire sum,carry;
```

```
adder1bit uut(a,b,cin,sum,carry);

initial
begin
a=1'b0; b=1'b0; cin=1'b0;
#10;
a=1'b0; b=1'b0; cin=1'b1;
#10;
a=1'b0; b=1'b1; cin=1'b0;
#10;
a=1'b0; b=1'b1; cin=1'b1;
#10;
a=1'b1; b=1'b0; cin=1'b0;
#10;
a=1'b1; b=1'b0; cin=1'b1;
#10;
a=1'b1; b=1'b1; cin=1'b0;
#10;
a=1'b1; b=1'b1; cin=1'b1;
#10;
$finish();
end
endmodule
```

Simulation:



2. Use this 1-bit structural code to design 4-bit parallel adder. Write a testbench for random inputs and check for 4 different cases.

Code:

```
`timescale 1ns / 1ps

module paralleladder(input [3:0]a, input [3:0]b,input cin, output carry, output  [3:0]sum);

wire c1,c2,c3;

adder1bit a1(.a(a[0]),.b(b[0]),.cin(cin),.sum(sum[0]),.carry(c1));

adder1bit a2(.a(a[1]),.b(b[1]),.cin(c1) ,.sum(sum[1]),.carry(c2));

adder1bit a3(.a(a[2]),.b(b[2]),.cin(c2) ,.sum(sum[2]),.carry(c3));

adder1bit a4(.a(a[3]),.b(b[3]),.cin(c3) ,.sum(sum[3]),.carry(carry));


endmodule
```

Testbench:

```
`timescale 1ns / 1ps


module paralleladder_tb();
reg [3:0]a,b;
reg cin;
wire [3:0]sum;
wire carry;


paralleladder uut(a, b, cin, carry, sum);


initial
begin
a=4'b0000; b=4'b0000; cin=0;
#10;
a=4'b1001; b=4'b0101; cin=0;
#10;
a=4'b0011; b=4'b0110; cin=0;
#10;
a=4'b0100; b=4'b0010; cin=0;
#10;
$finish();
end
endmodule
```

Simulation:


Name
Value
a[3:0]
b[3:0]
cin
sum[3:0]
carry
0
0
0
0
0
8.923 ns
0.000 ns
20.000 ns
0
9
3
4
0
5
6
2
0
e
9
6