Task:

Design a 4-bit Ripple Carry Adder on FPGA.

Verilog Code:

```
// this verilog code implements an n-bit ripple carry adder using behavioral design
module ripple4_beh(X, Y, Cin, S, Cout);
        input [n-1:0] X, Y;
        input Cin;
        parameter n = 4;
        output reg [n-1:0] S;
        output reg Cout;

        // intermediate ripple carry
        reg [n:0] C;
        // index for the 'for' loop
        integer k;

        always@(X, Y, Cin)
        begin

                C[0] = Cin;

                for(k = 0; k <= n-1; k = k + 1)
                begin

                        S[k] = X[k] ^ Y[k] ^ C[k];
                        C[k+1] = (X[k] & Y[k]) | (C[k] & X[k]) | (C[k] & Y[k]);

                end
                Cout = C[n];
        end
endmodule
```

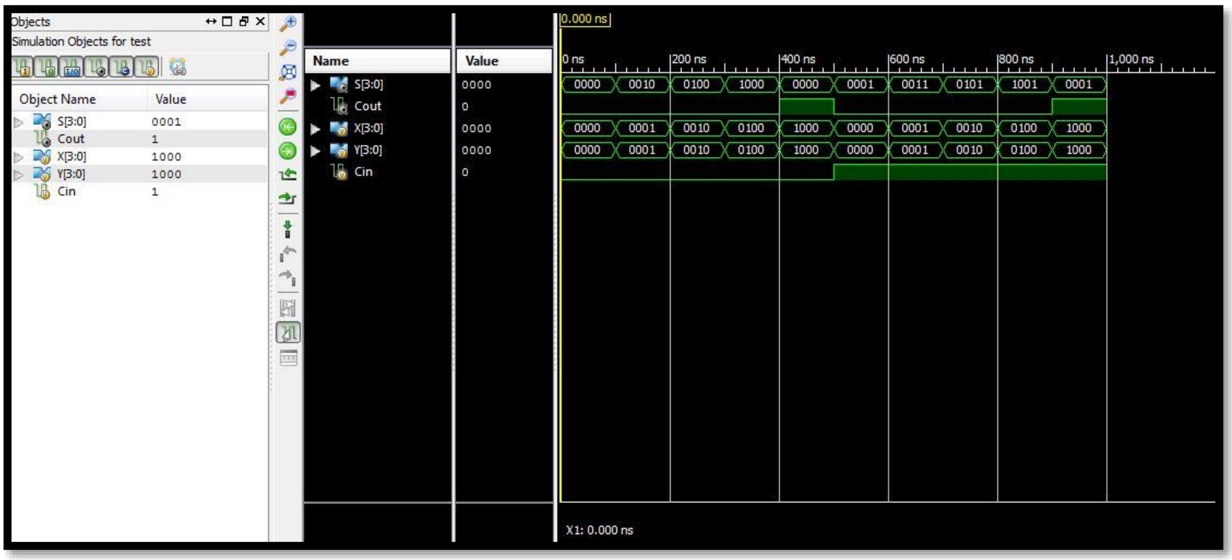
Test Bench:

```
// test bench for the behavioral model of 4-bit ripple carry adder
module tb ();
// inputs are reg type
reg Cin, X[3:0], Y[3:0];

// outputs are wire type
wire Cout, S[3:0];


// instantiation of the module
ripple_adder_behavioral dut(.X(X), .Y(Y), .Cin(Cin), .S(S), .Cout(Cout));

initial begin
// initialization of the reg variables
Cin = 0; X = 4'b0000; Y = 4'b0000; #10

Cin = 0; X = 4'b0001; Y = 4'b0001; #10
Cin = 0; X = 4'b0010; Y = 4'b0010; #10
Cin = 0; X = 4'b0100; Y = 4'b0100; #10
Cin = 0; X = 4'b1000; Y = 4'b1000; #10
Cin = 1; X = 4'b0001; Y = 4'b0001; #10
Cin = 1; X = 4'b0010; Y = 4'b0010; #10
Cin = 1; X = 4'b0100; Y = 4'b0100; #10
Cin = 1; X = 4'b1000; Y = 4'b1000; #10


end
endmodule
```

UCF:

```
# PlanAhead Generated physical constraints

NET "Cin" LOC = V8;
NET "Cout" LOC = F6;
NET "S[0]" LOC = AE24;
NET "S[1]" LOC = AD24;
NET "S[2]" LOC = AD25;
NET "S[3]" LOC = AD26;
NET "X[0]" LOC = AC24;
NET "X[1]" LOC = AC25;
NET "X[2]" LOC = AE26;
NET "X[3]" LOC = AE27;
NET "Y[0]" LOC = AF26;
NET "Y[1]" LOC = AF25;
NET "Y[2]" LOC = AG27;
NET "Y[3]" LOC = U25;
```

Verification of Circuit on ModelSim:



Objects

Simulation Objects for test

Object Name	Value
S[3:0]	0001
Cout	1
X[3:0]	1000
Y[3:0]	1000
Cin	1

Name	Value
S[3:0]	0000
Cout	0
X[3:0]	0000
Y[3:0]	0000
Cin	0

```
0 ns  |200 ns |400 ns |600 ns |800 ns |1,000 ns
S[3:0]: 0000 0010 0100 1000 0000 0001 0011 0101 1001 0001
X[3:0]: 0000 0001 0010 0100 1000 0000 0001 0010 0100 1000
Y[3:0]: 0000 0001 0010 0100 1000 0000 0001 0010 0100 1000
```

Figure no. 1 Verification of Circuit on ModelSim

RTL Circuit diagram:

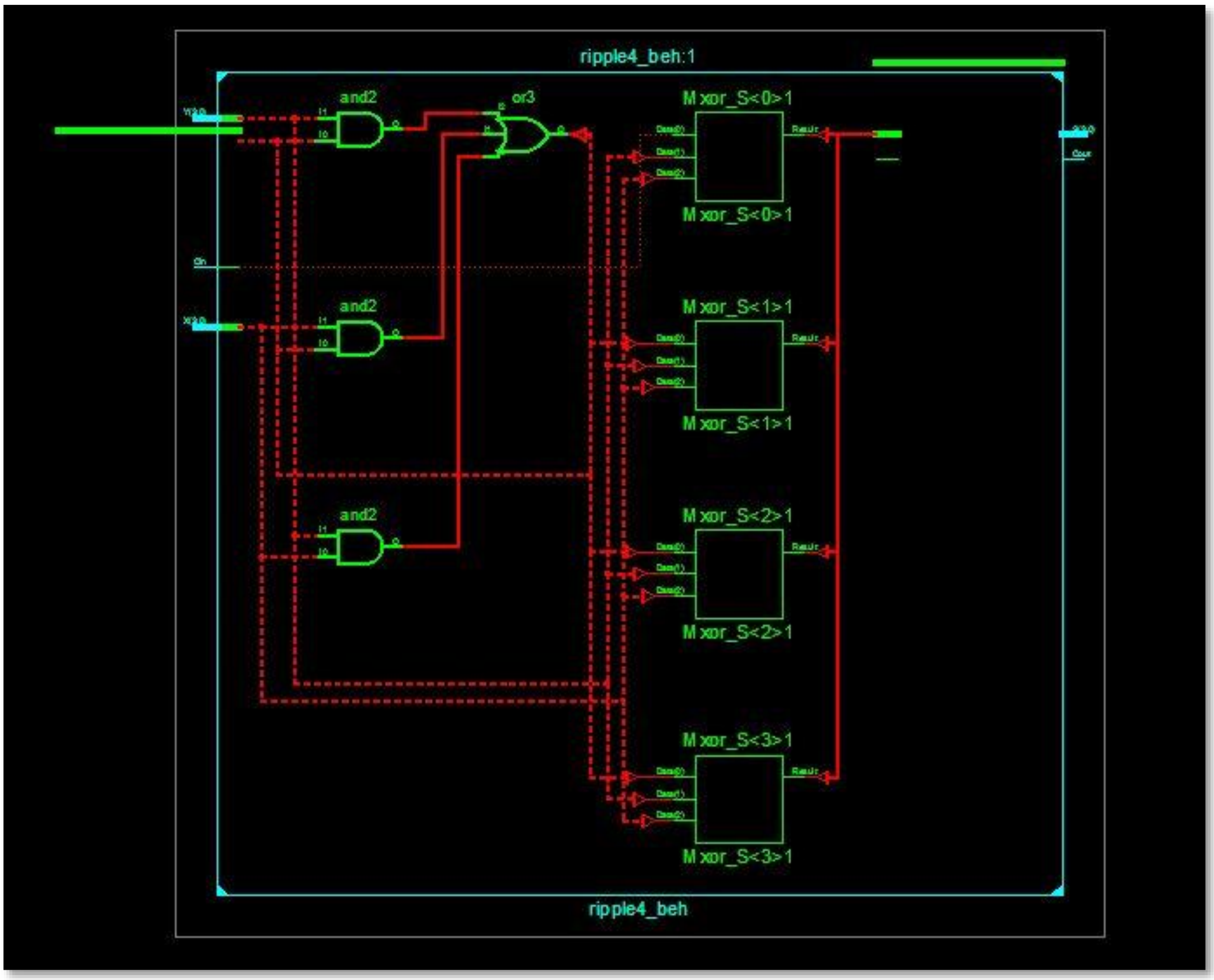


Figure no. 2 RTL Circuit Diagram