LAB # 04

OBJECTIVE: Develop, implement and simulate a 4-bit ripple carry adder using 4 full adders.

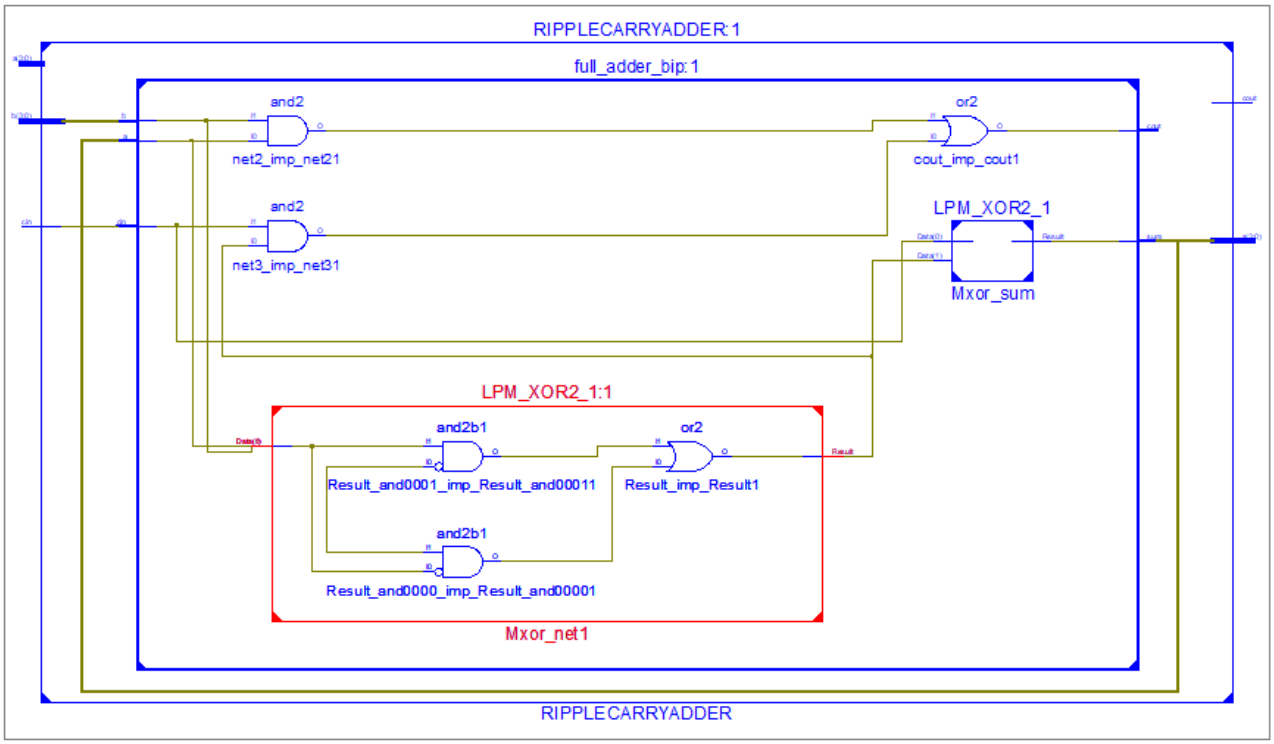
VERILOG MODULE:

```
module RIPPLECARRYADDER(
   input [3:0] a,
   input [3:0] b,
   input cin,
   output [3:0] s,
   output cout
   );
wire [3:0] c ;
assign cout = c[3];
full_adder_bip inst0(s[0], b[0], cin, s[0],c[0]);
full_adder_bip inst1(s[1], b[1], c[0], s[1],c[1]);
full_adder_bip inst2(s[2], b[2], c[1], s[2],c[2]);
full_adder_bip inst3(s[3], b[3], c[2], s[3],c[3]);
endmodule
```

full_adder_bip VERILOG:

```
module full_adder_bip(a,b,cin,sum,cout);
   input a,b,cin;
   output sum, cout;
         xor inst1 (net1,a,b);
         and inst2 (net2,a,b);
         xor inst3 (sum,net1,cin);
         and inst4 (net3,net1,cin);
         or inst5 (cout,net3,net2);
endmodule
```

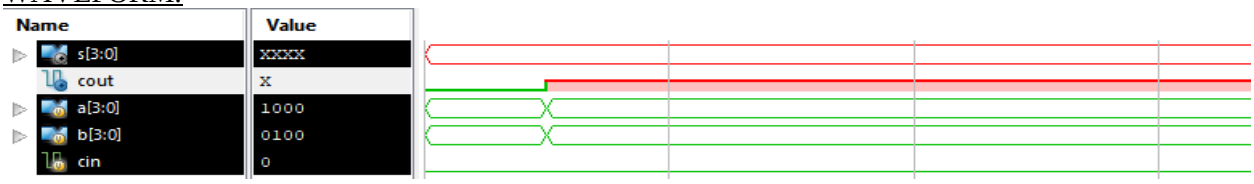
Scemantic Diagram:



RIPPLE4BITADDERTF Text Fixture Module

```
module RIPPLE4BITADDERTF;
	// Inputs
	reg [3:0] a;
	reg [3:0] b;
	reg cin;
	// Outputs
	wire [3:0] s;
	wire cout;
	// Instantiate the Unit Under Test (UUT)
	RIPPLECARRYADDER uut (
		.a(a),
		.b(b),
		.cin(cin),
		.s(s),
		.cout(cout)
	);
	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;
		cin = 0;
		// Wait 100 ns for global reset to finish
		#100;
		a=4'b1000;
		b=4'b0100;
		cin = 0;
		// Add stimulus here
	end
endmodule
```

WAVEFORM:



SUBTRACTOR:

Verilog Module:

```
module Ripplecarry(
    input [3:0] a,
    input [3:0] b,
    input cin,
    output [3:0] s,
    output cout
    );
    wire c1,c2,c3;
    Fulladder1 FA0(
    .a(a[0]),
    .b(b[0]^cin),
    .cin(cin),
    .s(s[0]),
    .cout(c1)
    );
    Fulladder1 FA1(
     .a(a[1]),
    .b(b[1]^cin),
    .cin(c1),
    .s(s[1]),
    .cout(c2)
    );
    Fulladder1 FA2(
     .a(a[2]),
    .b(b[2]^cin),
    .cin(c2),
    .s(s[2]),
    .cout(c3)
    );
    Fulladder1 FA3(
     .a(a[3]),
    .b(b[3]^cin),
    .cin(c3),
    .s(s[3]),
    .cout(cout)
    );
endmodule
```

Verilog Test Fixture Code:

```
25  module Ripplecarry_tv;
26
27     // Inputs
28     reg [3:0] a;
29     reg [3:0] b;
30     reg cin;
31
32     // Outputs
33     wire [3:0] s;
34     wire cout;
35
36     // Instantiate the Unit Under Test (UUT)
37     Ripplecarry uut (
38        .a(a),
39        .b(b),
40        .cin(cin),
41        .s(s),
42        .cout(cout)
43     );
44
45     initial begin
46        // Initialize Inputs
47        a = 4'b1000;
48        b = 4'b0100;
49        cin = 1;
50
```

Waveform: