ELC 2137 Lab 5: Intro to Verilog

Trevor Jackson, Carlos Hernandez, and Makenna Meyers

February 17, 2020

Summary

This lab was an introduction to coding with Verilog. We learned to code the half adder, full adder, and adder/subtractor circuits that we had assembled with wires in previous labs. The codes for the adders and their test benches can be found below in Codes 1, 2, 3, 4, 5, and 6. Expected Results Tables (ERTs) and simulation waveforms for each adder can be found in Figures 1, 2, and 3. In addition to writing and testing the code for the adders, block diagrams were drawn to illustrate the inputs, outputs, wires, and gates involved with each circuit. Those can be found in Figures 4, 5, and 6.

Q&A

1. What is one thing you still don't understand about Verilog?

   What is the most efficient way to implement the logic gates?

Results

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

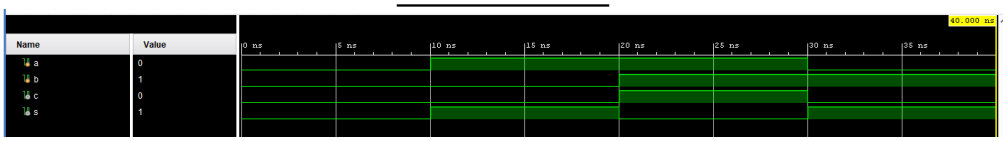

Figure 1: Half Adder ERT and Simulation Waveform

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

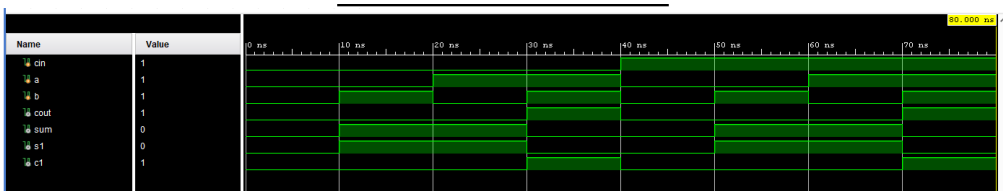

Figure 2: Full Adder ERT and Simulation Waveform

A	B	Cout	S2	S1
00	01	1	1	1
00	10	1	1	0
00	11	1	0	1
01	01	0	0	0
10	01	0	0	1
10	00	0	1	0

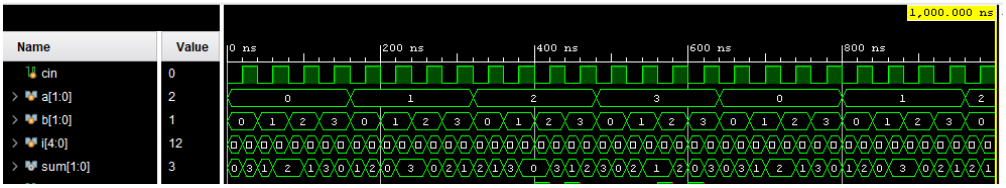

Figure 3: Adder/Subtractor ERT and Simulation Waveform

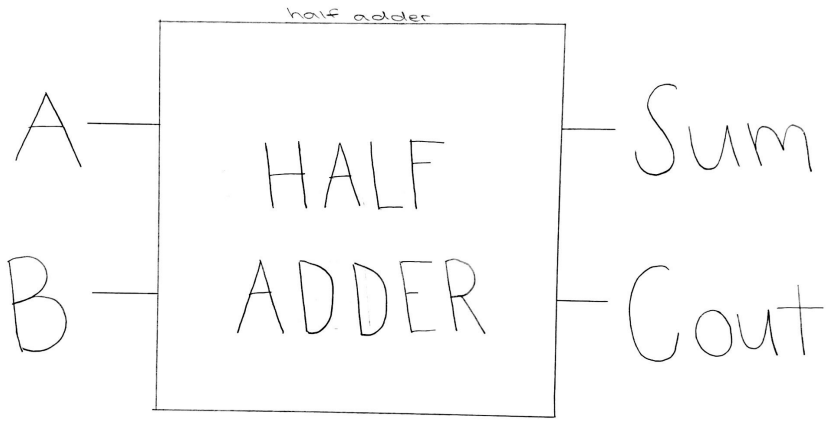

Figure 4: Half Adder Block Diagram

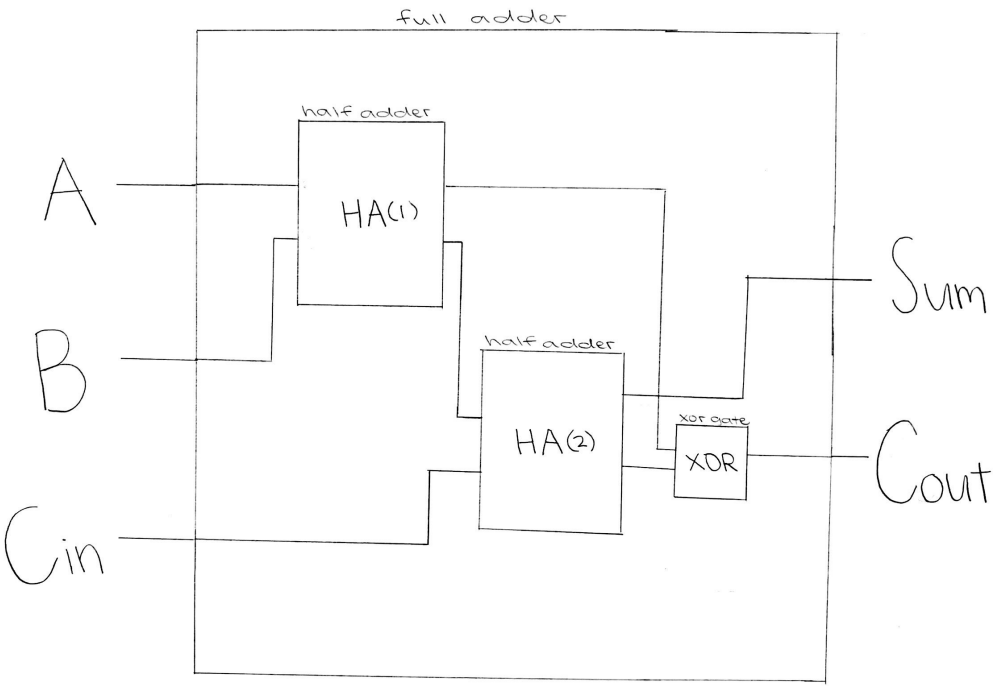

Figure 5: Full Adder Block Diagram

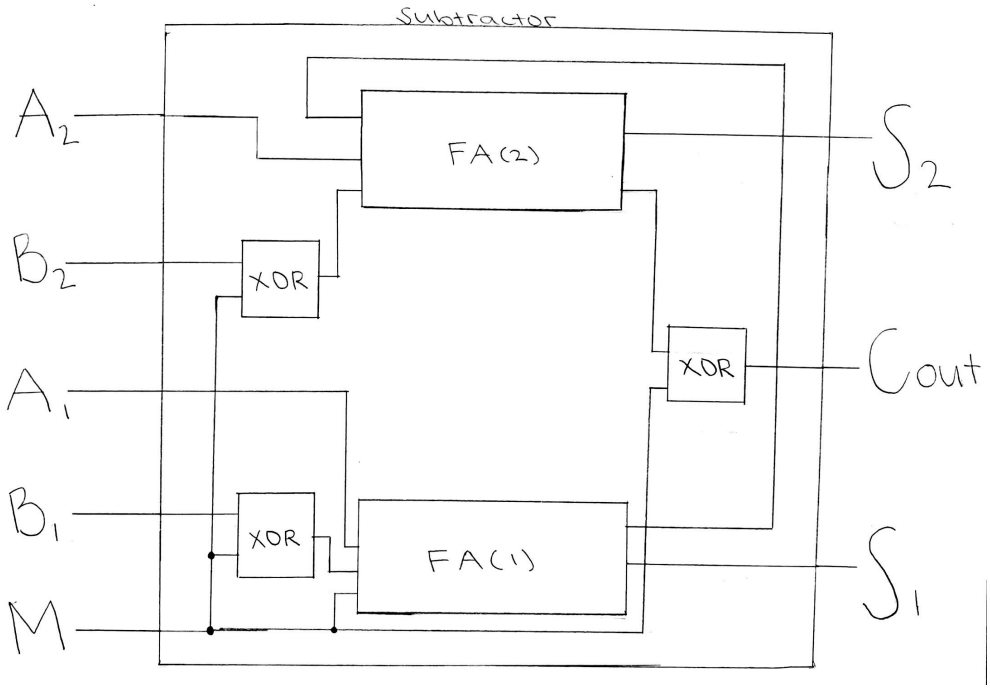

Figure 6: Adder/Subtractor Block Diagram

Code

Listing 1: Half Adder Code

```
‘timescale 1ns / 1ps
//
   //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 02/11/2020 02:33:52 PM
// Design Name:
// Module Name: halfadder
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module  halfadder (input a, b,output s, c);
assign s = a ^ b;
assign c = a & b;

endmodule
```

Listing 2: Half Adder Test Bench Code

```
‘timescale 1ns / 1ps

module  halfadder_test ();
    reg a, b;
    wire c, s;

    halfadder  ha0(
    .a(a), .b(b),
    .c(c), .s(s)
    );

    initial  begin
        a=0; b=0;  #10;
        a=1; b=0;  #10;
        a=1; b=1;  #10;
        a=0; b=1;  #10;
```

```
    $finish;
   end
endmodule
```

Listing 3: Full Adder Code

```
`timescale 1ns / 1ps
//
   //////////////////////////////////////////////////////////////////////////////////

// Company:
// Engineer:
//
// Create Date: 02/11/2020 03:26:50 PM
// Design Name:
// Module Name: fulladder
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module  fulladder (input a, b, cin ,output sum , cout);

wire s1 , c1, c2;
halfadder  ha0(.a(a), .b(b), .s(s1), .c(c1));
halfadder  ha1(.a(a), .b(b), .s(sum), .c(cout));

or o1(cout,c1,c2);
/*assign  c1 = a&b;
assign  s1 = a^b;
assign  c2 = cin & s1;

assign  sum = cin & s1;
assign  cout = c1 | c2; */
endmodule
```

Listing 4: Full Adder Test Bench Code

```
`timescale 1ns / 1ps

module  fulladder_test ();
reg cin;
reg [3:0] a , b;
```

```
wire  cout , sum;

halfadder  ha0(.a(a), .b(b), .s(s1), .c(c1));
initial begin
    cin=0; a=0; b=0;  #10;
    cin=0; a=0; b=1;  #10;
    cin=0; a=1; b=0;  #10;
    cin=0; a=1; b=1;  #10;
    cin=1; a=0; b=0;  #10;
    cin=1; a=0; b=1;  #10;
    cin=1; a=1; b=0;  #10;
    cin=1; a=1; b=1;  #10;

    $finish;
end

halfadder  ha1(.a(a), .b(b), .s(sum), .c(cout));
initial begin
    cin=0; a=0; b=0;  #10;
    cin=0; a=0; b=1;  #10;
    cin=0; a=1; b=0;  #10;
    cin=0; a=1; b=1;  #10;
    cin=1; a=0; b=0;  #10;
    cin=1; a=0; b=1;  #10;
    cin=1; a=1; b=0;  #10;
    cin=1; a=1; b=1;  #10;

    $finish;
end
endmodule
```

Listing 5: Adder/Subtractor Code

```
`timescale 1ns / 1ps

module addersub(input cin,
                input [1:0] a, b,
                output cout, V,
                output [1:0] sum);

        wire x0,x1;
        wire c0;

        xor (x0, b[0], cin);
        xor (x1, b[1], cin);
        xor (cout, c1, cin);
        xor (V, c1, c0);

        fulladder fa0(.a(a[0]), .b(x0), .cin(cin), .sum(sum[0]), .cout(c0)
           );
        fulladder fa1(.a(a[1]), .b(x1), .cin(c0), .sum(sum[1]), .cout(c1))
           ;
```

```
        endmodule
```

Listing 6: Adder/Subtractor Test Bench Code

```
 `timescale 1ns / 1ps

module addersub_test();
    reg cin;
    reg [1:0] a, b;
    reg [4:0] i;
    wire [1:0] sum;
    wire cout;

    addersub AS(.a(a), .b(b), .cin(cin), .sum(sum), .cout(cout));


    initial
        begin
            for(i=0; i <= 8'hff; i = i+1)
                begin
                    a[1:0] = i [4:3];
                    b[1:0] = i [2:1];
                    cin = i[0];
                    #20;
                end
         end
    endmodule
```