Lab D - Report

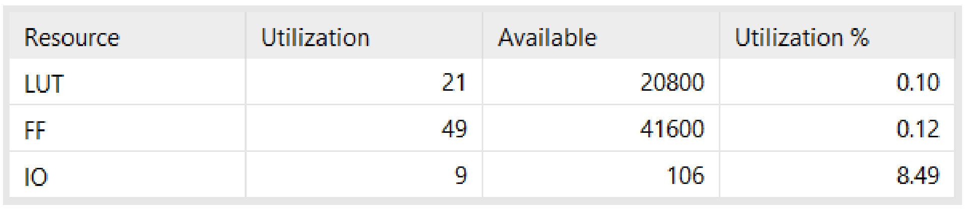
Stephen Komolafe (21336975)

A new project was set up, adding the bargaphtest.v, bargraph.v and clock.v files as source files and bargraphtest.xdc as a constraint file. The Basys3 board was required to be selected when setting up the project to ensure compatibility with the physical board once connected to the pc. However, due to using the latest version of Vivado (2023.2), the Basys3 board was not present in the board selection menu. To resolve this the xc7a35tcpg236-1 was selected from the parts menu instead as the demonstrators had mentioned that it functions the same as choosing the Basys3 board.

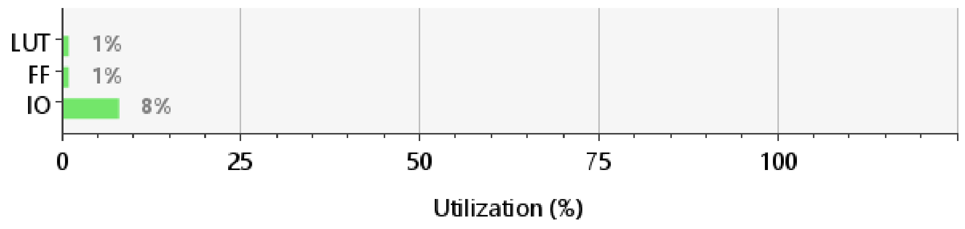
Once the project had been set up the physical board was connected to the pc using a USB cable. In order to be able to program the board a bitstream had to be generated via clicking “Generate Bitstream” under “Program and Debug” in the Flow Navigator section of the Vivado GUI. From there the selected design (bargaphtest.v) was programmed onto the board.

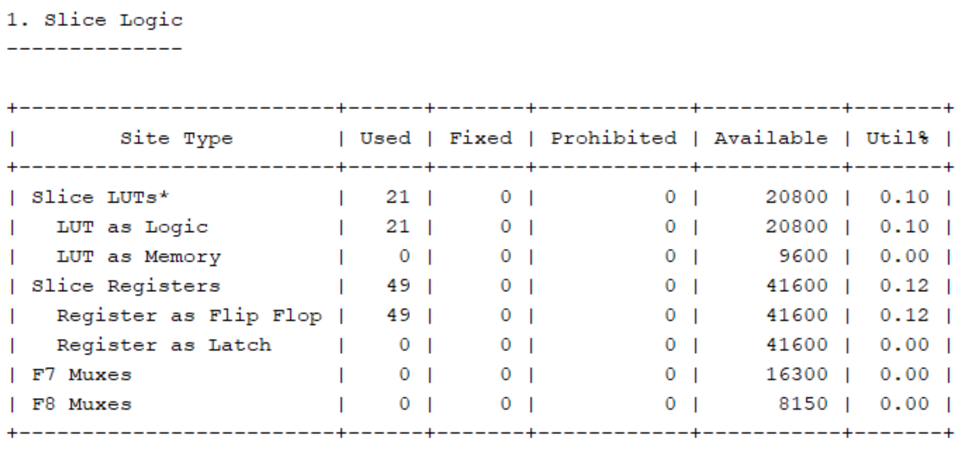
It was evident that the board was programmed successfully as 8 of the 16 LEDs on the board (LD0 - LD7) would turn on one after the other with what appears to be a delay of time T that would increment by some amount after each subsequent LED. To best describe it, the first few LEDs in the 8 LED sequence would turn on near instantaneously whereas with the last few LEDs it was more apparent that there was a delay.

The resource utilization data from the synthesis and implementation report was observed and recorded as seen below.



Resource	Utilization	Available	Utilization %
LUT	21	20800	0.10
FF	49	41600	0.12
IO	9	106	8.49





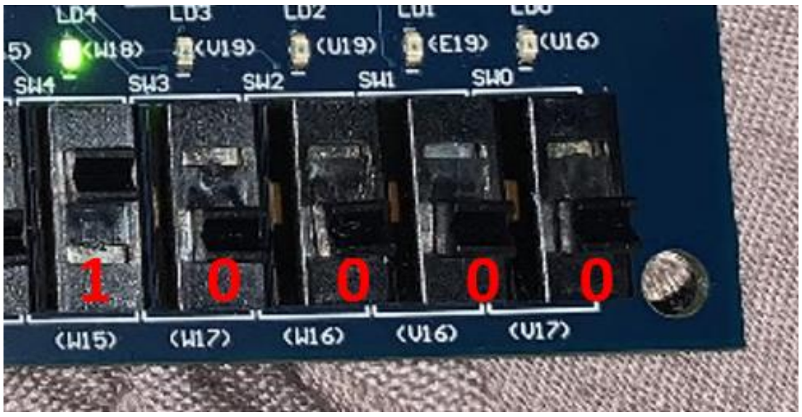
```
1. Slice Logic
--------------

+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |   21 |     0 |          0 |     20800 |  0.10 |
|   LUT as Logic          |   21 |     0 |          0 |     20800 |  0.10 |
|   LUT as Memory         |    0 |     0 |          0 |      9600 |  0.00 |
| Slice Registers         |   49 |     0 |          0 |     41600 |  0.12 |
|   Register as Flip Flop |   49 |     0 |          0 |     41600 |  0.12 |
|   Register as Latch     |    0 |     0 |          0 |     41600 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     16300 |  0.00 |
| F8 Muxes                |    0 |     0 |          0 |      8150 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
1.1 Summary of Registers by Type
--------------------------------

+-------+--------------+-------------+--------------+
| Total | Clock Enable | Synchronous | Asynchronous |
+-------+--------------+-------------+--------------+
| 0     |            _ |           - |            - |
| 0     |            _ |           - |          Set |
| 0     |            _ |           - |        Reset |
| 0     |            _ |         Set |            - |
| 0     |            _ |       Reset |            - |
| 0     |          Yes |           - |            - |
| 0     |          Yes |           - |          Set |
| 0     |          Yes |           - |        Reset |
| 0     |          Yes |         Set |            - |
| 49    |          Yes |       Reset |            - |
+-------+--------------+-------------+--------------+


2. Memory
---------

+----------------+------+-------+------------+-----------+-------+
|    Site Type   | Used | Fixed | Prohibited | Available | Util% |
+----------------+------+-------+------------+-----------+-------+
| Block RAM Tile |    0 |     0 |          0 |        50 |  0.00 |
|   RAMB36/FIFO* |    0 |     0 |          0 |        50 |  0.00 |
|   RAMB18       |    0 |     0 |          0 |       100 |  0.00 |
+----------------+------+-------+------------+-----------+-------+

3. DSP
------

+-----------+------+-------+------------+-----------+-------+
| Site Type | Used | Fixed | Prohibited | Available | Util% |
+-----------+------+-------+------------+-----------+-------+
| DSPs      |    0 |     0 |          0 |        90 |  0.00 |
+-----------+------+-------+------------+-----------+-------+


4. IO and GT Specific
---------------------

+-----------------------------+------+-------+------------+-----------+-------+
|          Site Type          | Used | Fixed | Prohibited | Available | Util% |
+-----------------------------+------+-------+------------+-----------+-------+
| Bonded IOB                  |    9 |     0 |          0 |       106 |  8.49 |
| Bonded IPADs                |    0 |     0 |          0 |        10 |  0.00 |
| Bonded OPADs                |    0 |     0 |          0 |         4 |  0.00 |
| PHY_CONTROL                 |    0 |     0 |          0 |         5 |  0.00 |
| PHASER_REF                  |    0 |     0 |          0 |         5 |  0.00 |
| OUT_FIFO                    |    0 |     0 |          0 |        20 |  0.00 |
| IN_FIFO                     |    0 |     0 |          0 |        20 |  0.00 |
| IDELAYCTRL                  |    0 |     0 |          0 |         5 |  0.00 |
| IBUFDS                      |    0 |     0 |          0 |       104 |  0.00 |
| GTPE2_CHANNEL               |    0 |     0 |          0 |         2 |  0.00 |
| PHASER_OUT/PHASER_OUT_PHY   |    0 |     0 |          0 |        20 |  0.00 |
| PHASER_IN/PHASER_IN_PHY     |    0 |     0 |          0 |        20 |  0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY |    0 |     0 |          0 |       250 |  0.00 |
| IBUFDS_GTE2                 |    0 |     0 |          0 |         2 |  0.00 |
| ILOGIC                      |    0 |     0 |          0 |       106 |  0.00 |
| OLOGIC                      |    0 |     0 |          0 |       106 |  0.00 |
+-----------------------------+------+-------+------------+-----------+-------+
5. Clocking
-----------

+------------+------+-------+------------+-----------+-------+
|  Site Type | Used | Fixed | Prohibited | Available | Util% |
+------------+------+-------+------------+-----------+-------+
| BUFGCTRL   |    1 |     0 |          0 |        32 |  3.13 |
| BUFIO      |    0 |     0 |          0 |        20 |  0.00 |
| MMCME2_ADV |    0 |     0 |          0 |         5 |  0.00 |
| PLLE2_ADV  |    0 |     0 |          0 |         5 |  0.00 |
| BUFMRCE    |    0 |     0 |          0 |        10 |  0.00 |
| BUFHCE     |    0 |     0 |          0 |        72 |  0.00 |
| BUFR       |    0 |     0 |          0 |        20 |  0.00 |
+------------+------+-------+------------+-----------+-------+
```

```
6. Specific Feature
-------------------

+-------------+------+-------+------------+-----------+-------+
|  Site Type  | Used | Fixed | Prohibited | Available | Util% |
+-------------+------+-------+------------+-----------+-------+
| BSCANE2     |    0 |     0 |          0 |         4 |  0.00 |
| CAPTUREE2   |    0 |     0 |          0 |         1 |  0.00 |
| DNA_PORT    |    0 |     0 |          0 |         1 |  0.00 |
| EFUSE_USR   |    0 |     0 |          0 |         1 |  0.00 |
| FRAME_ECCE2 |    0 |     0 |          0 |         1 |  0.00 |
| ICAPE2      |    0 |     0 |          0 |         2 |  0.00 |
| PCIE_2_1    |    0 |     0 |          0 |         1 |  0.00 |
| STARTUPE2   |    0 |     0 |          0 |         1 |  0.00 |
| XADC        |    0 |     0 |          0 |         1 |  0.00 |
+-------------+------+-------+------------+-----------+-------+
```

```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   49 |        Flop & Latch |
| CARRY4   |   16 |          CarryLogic |
| LUT6     |    9 |                 LUT |
| OBUF     |    8 |                  IO |
| LUT3     |    5 |                 LUT |
| LUT2     |    5 |                 LUT |
| LUT4     |    4 |                 LUT |
| LUT5     |    3 |                 LUT |
| LUT1     |    2 |                 LUT |
| IBUF     |    1 |                  IO |
| BUFG     |    1 |               Clock |
+----------+------+---------------------+
```

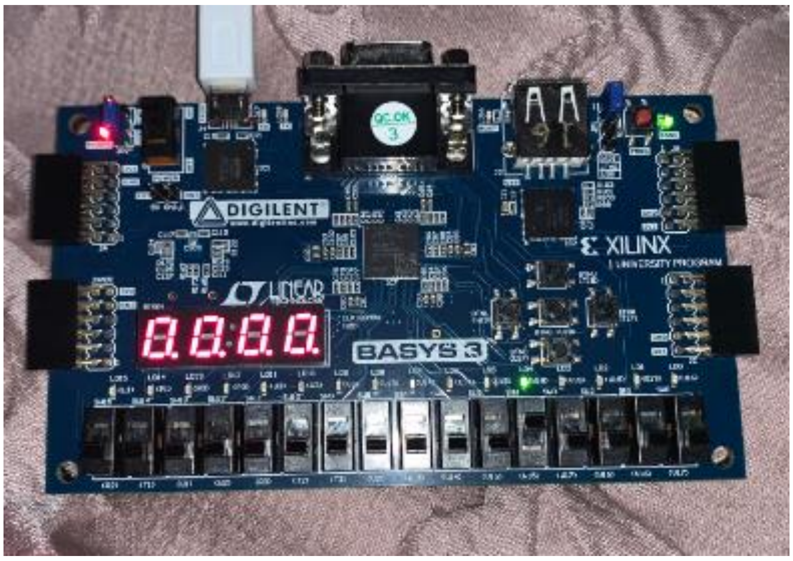
Being tasked to portray the binary equivalent of the board number using the on-board input switches, the Verilog code of the source and constraints file had to be altered. New constraints were implemented in the bargraphtest.xdc file to declare the pin (V17, V16, W16, etc.), port (sw0-sw7) and 3-volt power source (IOSTANDARD LVCMOS33) of each switch. The logic code of bargraph.v was altered so that if any specific switch were to be flipped it's corresponding LED would toggle on. Once programmed onto the board a photo of the working design was captured.

Board number(decimal): 16 Board number(binary): 1 0 0 0 0



Appendix

Full Board Photo:



Code(**Altered/Added parts**):

-<u>**bargraphtest.xdc**</u>

```
set_property PACKAGE_PIN W5 [get_ports CCLK]

set_property PACKAGE_PIN U16 [get_ports LD0]

set_property PACKAGE_PIN E19 [get_ports LD1]

set_property PACKAGE_PIN U19 [get_ports LD2]

set_property PACKAGE_PIN U15 [get_ports LD5]

set_property PACKAGE_PIN V19 [get_ports LD3]

set_property PACKAGE_PIN W18 [get_ports LD4]

set_property PACKAGE_PIN U14 [get_ports LD6]

set_property PACKAGE_PIN V14 [get_ports LD7]

set_property IOSTANDARD LVCMOS33 [get_ports LD1]

set_property IOSTANDARD LVCMOS33 [get_ports LD7]

set_property IOSTANDARD LVCMOS33 [get_ports LD2]

set_property IOSTANDARD LVCMOS33 [get_ports LD3]

set_property IOSTANDARD LVCMOS33 [get_ports LD4]

set_property IOSTANDARD LVCMOS33 [get_ports LD5]

set_property IOSTANDARD LVCMOS33 [get_ports LD6]

set_property IOSTANDARD LVCMOS33 [get_ports LD0]

set_property IOSTANDARD LVCMOS33 [get_ports CCLK]
set_property PACKAGE_PIN V17 [get_ports sw0]

set_property IOSTANDARD LVCMOS33 [get_ports sw0]

set_property PACKAGE_PIN V16 [get_ports sw1]

set_property IOSTANDARD LVCMOS33 [get_ports sw1]

set_property PACKAGE_PIN W16 [get_ports sw2]

set_property IOSTANDARD LVCMOS33 [get_ports sw2]

set_property PACKAGE_PIN W17 [get_ports sw3]

set_property IOSTANDARD LVCMOS33 [get_ports sw3]

set_property PACKAGE_PIN W15 [get_ports sw4]

set_property IOSTANDARD LVCMOS33 [get_ports sw4]

set_property PACKAGE_PIN V15 [get_ports sw5]

set_property IOSTANDARD LVCMOS33 [get_ports sw5]

set_property PACKAGE_PIN W14 [get_ports sw6]

set_property IOSTANDARD LVCMOS33 [get_ports sw6]

set_property PACKAGE_PIN W13 [get_ports sw7]

set_property IOSTANDARD LVCMOS33 [get_ports sw7]
```

-bargraphtest.v

```
// Basys Board and Spartan-3E Starter Board

// LED Bar Graph Test bargraphtest.v

// c 2008 Embedded Design using Programmable Gate Arrays  Dennis Silage

module bargraphtest(input CCLK, sw0, sw1, sw2, sw3, sw4, sw5, sw6, sw7, output LD7, LD6, LD5, LD4, LD3,
LD2, LD1, LD0);

        wire [7:0] data;

        clock M0 (CCLK, 625000, clock);

        bargraph M1 (clock, sw0, sw1, sw2, sw3, sw4, sw5, sw6, sw7, LD7, LD6, LD5, LD4, LD3, LD2, LD1, LD0);

        gendata M2 (clock, data);

endmodule

// generate bar graph test data

module gendata(input clock, output reg [7:0] gdata);

        always@(negedge clock)

                gdata=gdata+1;

endmodule
```

-bargraph.v

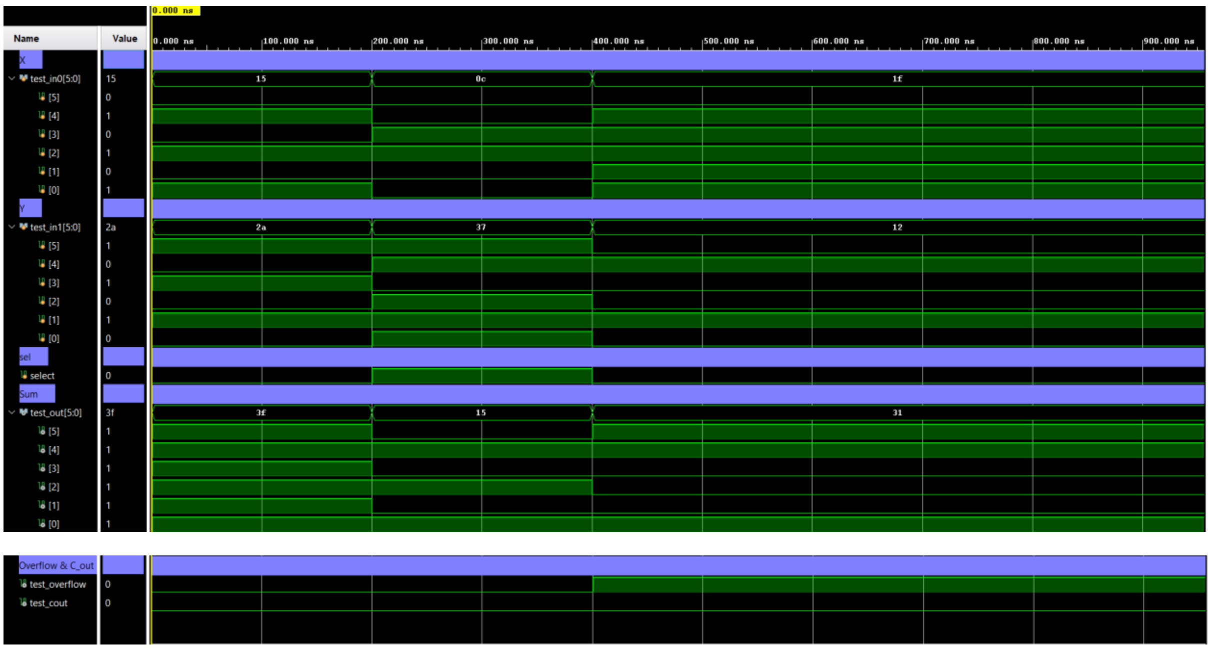
```
// Basys Board and Spartan-3E Starter Board
// LED Bar Graph bargraph.v
// c 2007 Embedded System Design in Verilog
//        using Programmable Gate Arrays  Dennis Silage
module bargraph(input clock, sw0, sw1, sw2, sw3, sw4, sw5, sw6, sw7, output LD7, LD6, LD5, LD4, LD3, LD2, LD1, LD0);
        reg [7:0] leddata;              // LED data
assign LD7=leddata[7];
assign LD6=leddata[6];
assign LD5=leddata[5];
assign LD4=leddata[4];
assign LD3=leddata[3];
assign LD2=leddata[2];
assign LD1=leddata[1];
assign LD0=leddata[0];
always@(posedge clock)
         begin
                leddata=8'b00000000;
                if (sw0==1)
                        leddata[0]=8'b00000001;
                if (sw1==1)
                        leddata[1]=8'b00000011;
                if (sw2==1)
                        leddata[2]=8'b00000111;
                if (sw3==1)
                        leddata[3]=8'b00001111;
                if (sw4==1)
                        leddata[4]=8'b00011111;
                if (sw5==1)
                        leddata[5]=8'b00111111;
                if (sw6==1)
                        leddata[6]=8'b01111111;
                if (sw7==1)
                        leddata[7]=8'b11111111;
        end
endmodule
```

Lab C write up:

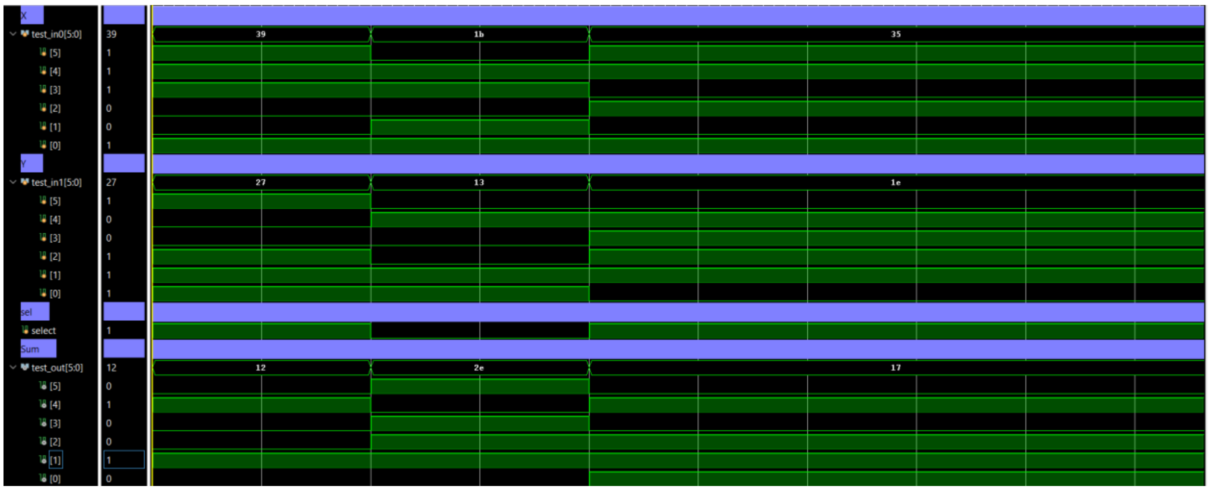
Test ID	x	y	sel	Cout (exp)	Overflow (exp)	Sum (exp)	Cout (obs)	Overflow (obs)	Sum (obs)	Pass/Fail
1	6'b010101	6'b101010	0	0	0	6'b111111	0	0	6'b111111	Pass
2	6'b001100	6'b110111	1	0	0	6'b010101	0	0	6'b010101	Pass
3	-6'b100001	6'b010010	0	0	1	6'b110001	0	1	6'b110001	Pass
4	6'b111001	6'b100111	1	1	0	6'b010010	1	0	6'b010010	Pass
5	6'b011011	-6'b101101	0	0	1	6’b101110	0	1	6’b101110	Pass
6	6'b110101	6’b011110	1	1	1	6’b010111	1	1	6’b010111	Pass
7	6’b111111	6’b111111	0	1	0	6’b111110	1	0	6’b111110	Pass
8	6’b000000	6’b000000	1	1	0	6’b000000	1	0	6’b000000	Pass
9	-6’b011010	-6’b101011	0	1	0	6’b111011	1	0	6’b111011	Pass

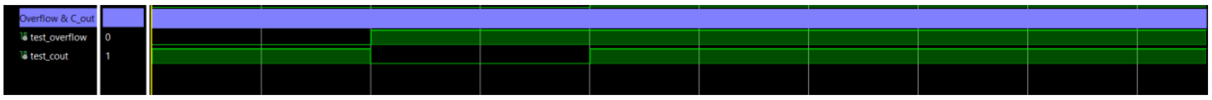
The circuit has passed all the tests as the observed output, overflow and sum values to what was expected. If I were to conduct this experiment again, I would increase the amount of test vectors with select as 1 in order to further verify the accuracy of the circuit when preforming 2’s complement subtraction.

Testbench Waveform[1-3]

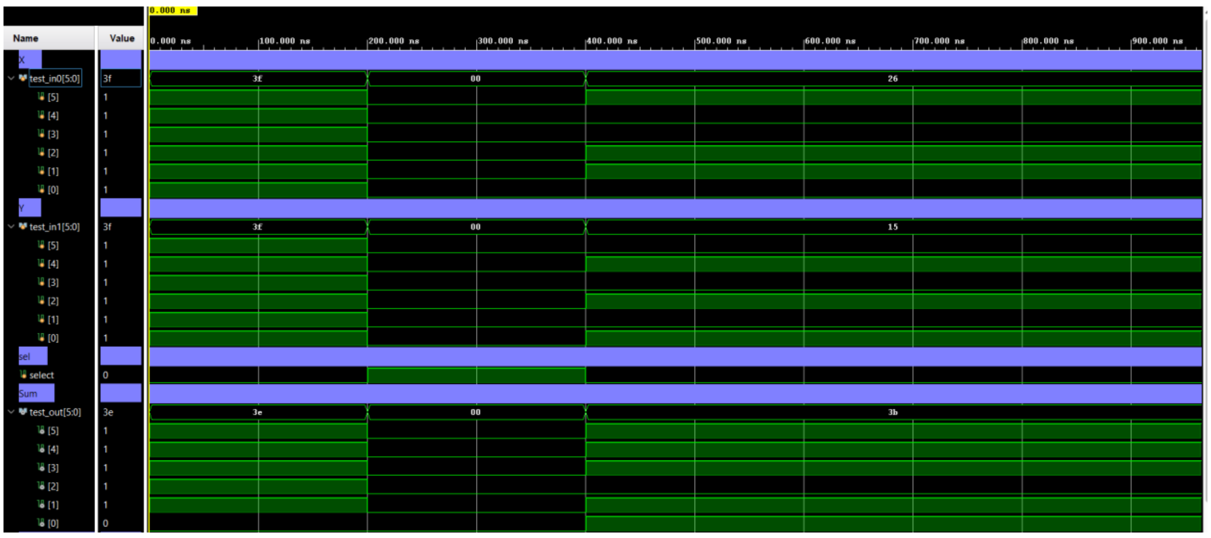


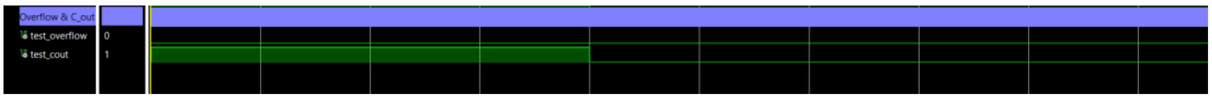
Testbench Waveform[3-6]



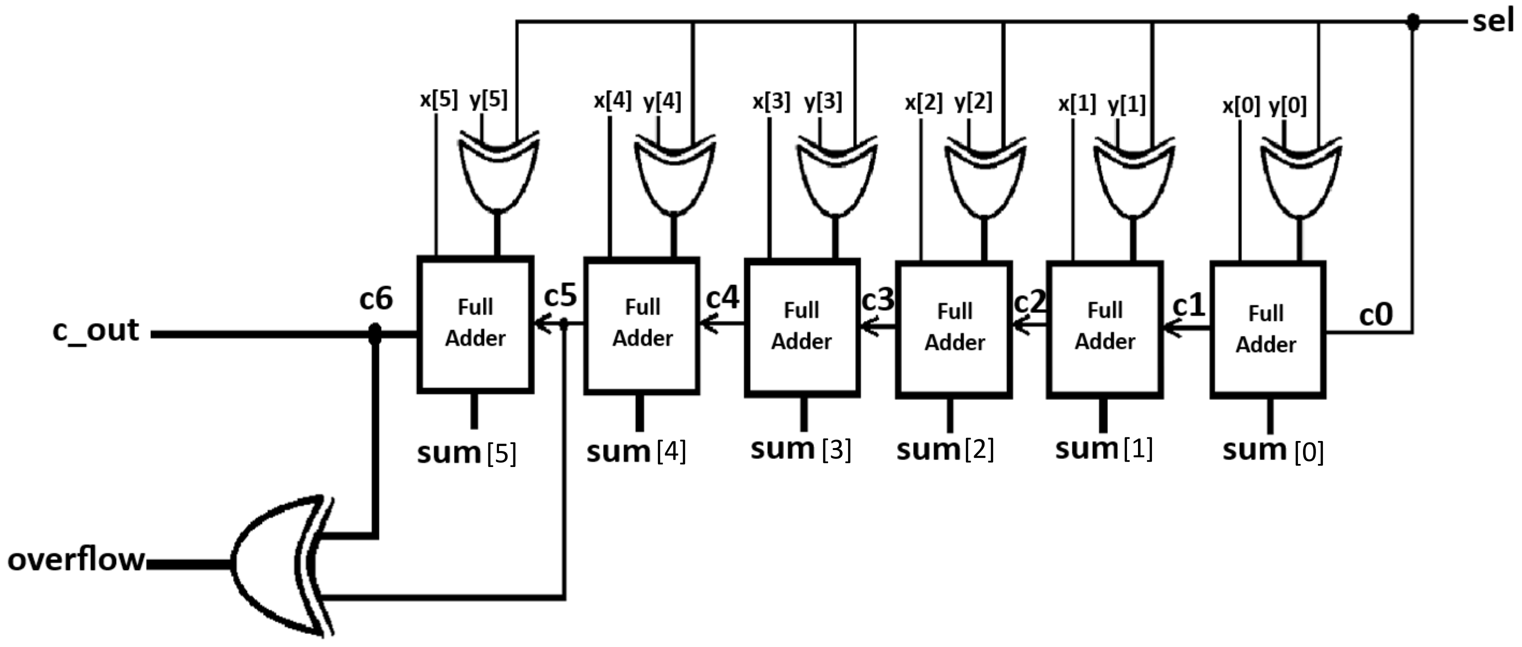


Testbench Waveform[6-9]





Block Diagram



-full_adder.v (from Blackboard)

```
module FullAdder(a, b, cin, s, cout);

 // 3C7 LabD 2010

 // a and b are the bits to add

 // cin is carry in

 input wire a, b, cin;


 // s is the sum of a and b. cout is any carry out bit

 // wires since just using assign here

 output wire s, cout;


 // logic for sum and carry

 assign s = cin ^ a ^ b;

 assign cout = (b & cin) | (a & cin) | (a & b);


endmodule
```

-6bit_ripple_adder.v

```
module sixbit_ripple_adder(

   input wire [5:0] x, y,              // 6-bit two's complement numbers to add

   input sel,                       // sel -> [0 = add] [1 = sub]
```

```
  output wire overflow,               // Output flagging overflow in the sum output
         [5:0] sum,              // 2s complement sum of x and y
          c_out,                 // MSB Carry out from the sum
      wire c1, c2, c3, c4, c5, c6    // carry output for each full adder
  );
  assign overflow = c6 ^ c5 ;         //overflow = c6 xor c5
  assign c0 = sel;                   //initialise 1st output as the current value of sel
  FullAdder adder_1(.a(x[0]),.b(y[0] ^ sel), .cin(sel), .s(sum[0]), .cout(c1));   // 1st bit
  FullAdder adder_2(.a(x[1]),.b(y[1] ^ sel), .cin(c1), .s(sum[1]), .cout(c2));    // 2nd bit
  FullAdder adder_3(.a(x[2]),.b(y[2] ^ sel), .cin(c2), .s(sum[2]), .cout(c3));    // 3rd bit
  FullAdder adder_4(.a(x[3]),.b(y[3] ^ sel), .cin(c3), .s(sum[3]), .cout(c4));    // 4th bit
  FullAdder adder_5(.a(x[4]),.b(y[4] ^ sel), .cin(c4), .s(sum[4]), .cout(c5));    // 5th bit
  FullAdder adder_6(.a(x[5]),.b(y[5] ^ sel), .cin(c5), .s(sum[5]), .cout(c6));    // 6th bit
  assign c_out = c6;                 // declare final carry output as output of ripplr adder
endmodule
```

-sbra_testbench.v (LabB - eq2_tb.v used as reference)

```
// Listing 1.7
// The `timescale directive specifies that
// the simulation time unit is 1 ns  and
// the simulation timestep is 10 ps
`timescale 1 ns/10 ps


module sbra_testbench;
  // signal declaration
  reg  [5:0] test_in0, test_in1;
  reg select;
  wire [5:0] test_out;
  wire test_overflow;
  wire test_cout;
  // instantiate the circuit under test
  sixbit_ripple_adder uut (.x(test_in0),.y(test_in1),.sum(test_out),.sel(select),.overflow(test_overflow),.c_out(test_cout));


  //  test vector generator
  initial
  begin
```

```
//test vector 1

test_in0 = 6'b010101;

test_in1 = 6'b101010;

select = 0;

#200;

// test vector 2

test_in0 = 6'b001100;

test_in1 = 6'b110111;

select = 1;

#200;

// test vector 3

test_in0 = -6'b100001;

test_in1 = 6'b010010;

select = 0;

#200;

//test vector 4

test_in0 = 6'b111001;

test_in1 = 6'b100111;

select = 1;

#200;

// test vector 5

test_in0 = 6'b011011;

test_in1 = -6'b101101;

select = 0;

#200;

// test vector 6

test_in0 = 6'b110101;

test_in1 = 6'b011110;

select = 1;

#200;

// test vector 7

test_in0 = 6'b111111;

test_in1 = 6'b111111;

select = 0;

#200;
```

```
    // test vector 8
    test_in0 = 6'b000000;
    test_in1 = 6'b000000;
    select = 1;
    #200;
    // test vector 9
    test_in0 = -6'b011010;
    test_in1 = -6'b101011;
    select = 0;
    #200;
    // stop simulation
    //$stop;
  end
endmodule
```