ELC 2137 Lab 09: ALU with Input Register

Maddie Vorhies

October 22, 2020

Summary

In this lab, the goal was to create an Arithmetic Logic Unit (ALU) that is capable of executing a few operations similar to a calculator. The operations that this ALU will be able to calculate are addition, subtraction, AND gates, OR gates, and XOR gates. To begin this lab I created a module known as register. The register stores a number so it can be used to execute the given operation. Then I created a test bench to make sure that my register outputs the correct number. Then I created the ALU. The ALU is a combinational circuit, which means it's constantly performing the specified calculation and producing an output. The ALU allows me to perform addition, subtraction, AND gates, OR gates, and XOR gates. Again, I created another test bench to make sure the ALU was working properly. To put these two modules together, I created a toplab module. I then programmed my board and tested out the given operations in the lab. All of my results were successful.

Results

Expected results tables

Table 1: *register* expected results table

Time (ns):	0-5	5-10	10-15	15-20	20-25	25-30	30-35	35-40	40-45	45-50	50-55
D (hex)	0	0	A	A	3	3	0	0	0→6	6	6
clk	0	1	0	1	0	1	0	1	0	1	0
en	0	0	1	1	1→0	0→1	1→0	0	0→1	1	1
rst	0	0→1	0	0	0	0	0	0	0	0	0
Q (hex)	X	X→0	0	A	A	A	A	A	A	6	6

Table 2: *alu* expected results table skeleton

Time (ns):	0-10	10-20	20-30	30-40	40-50	50-60
in0	0110	0110	0110	0110	0110	0110
in1	0010	0010	0010	0010	0010	0010
op	0000	0001	0010	0011	0100	0101
out	1000	0100	0010	0110	0100	0110

Figure 1: Simulation Waveform for Register

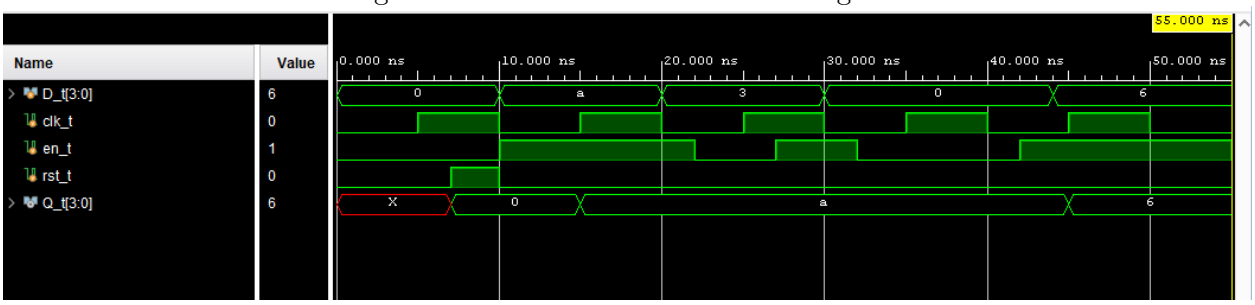


Figure 2: Simulation Waveform for ALU

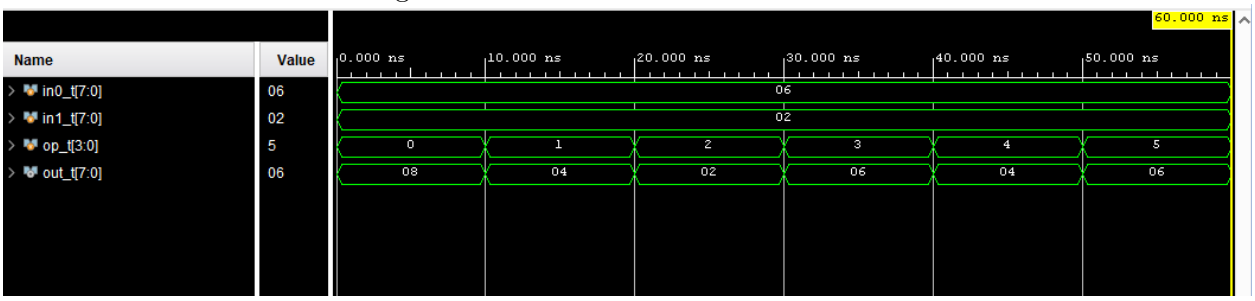
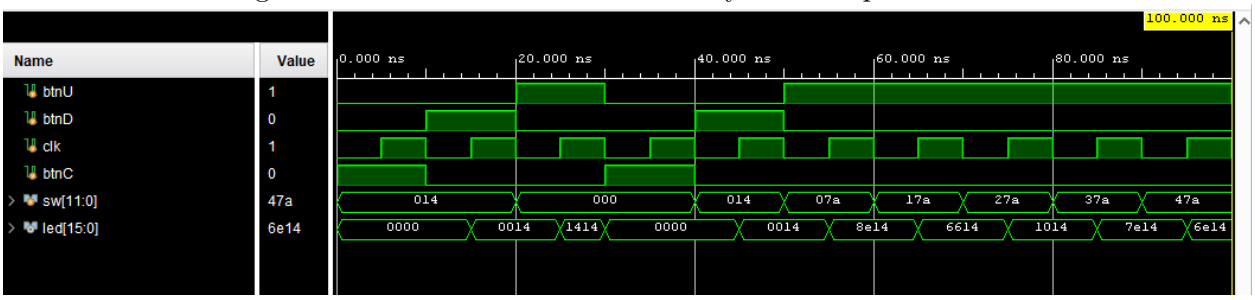


Figure 3: Simulation Waveform for basys3lab9 Top Simulation



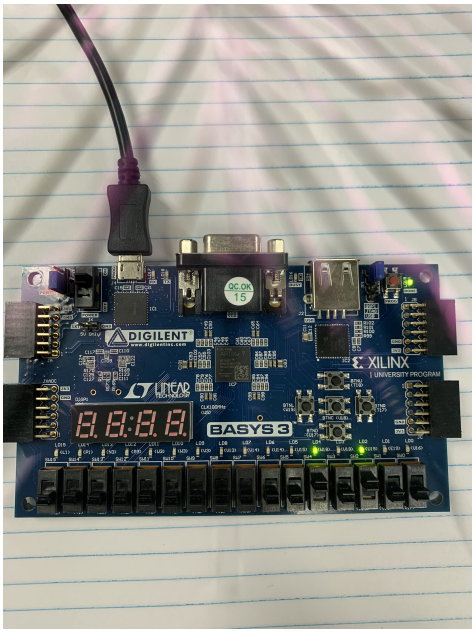
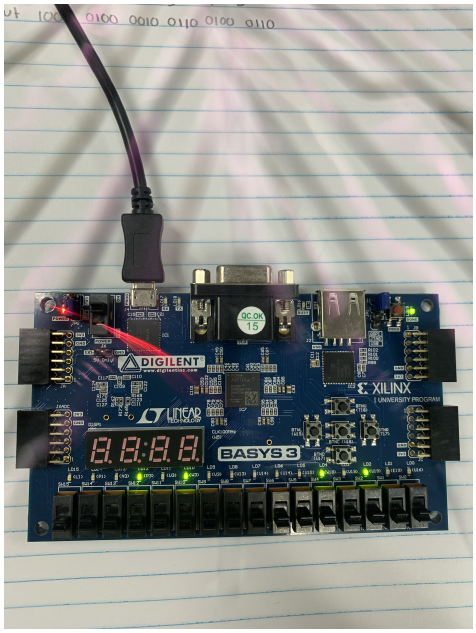
Display 14 on switches 7-0	Display 14 on switches 15-8
	

Table 3: Board Pictures

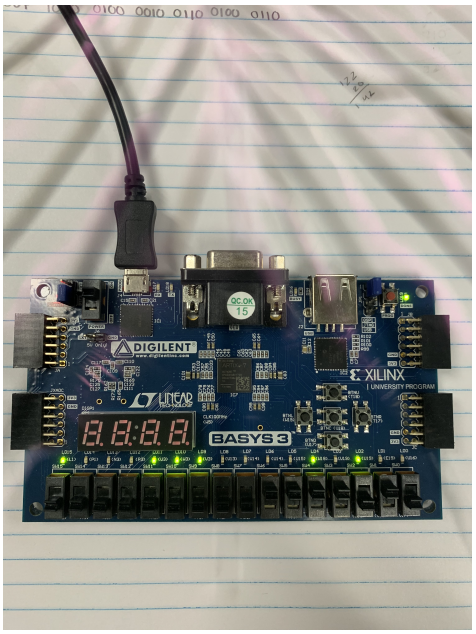
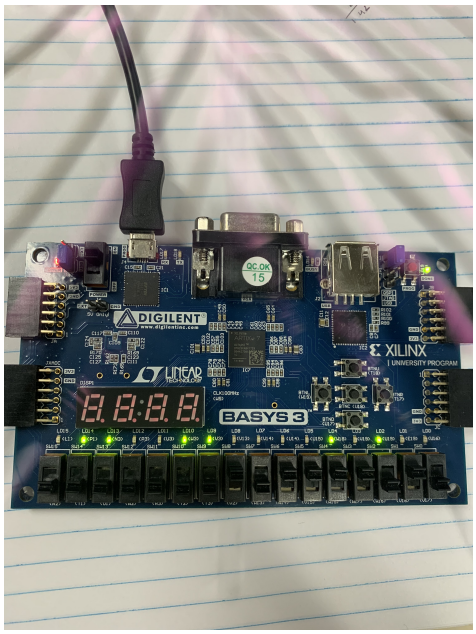
Add 14 adn 7A	Subtract 14 and 7A
	

Table 4: Board Pictures

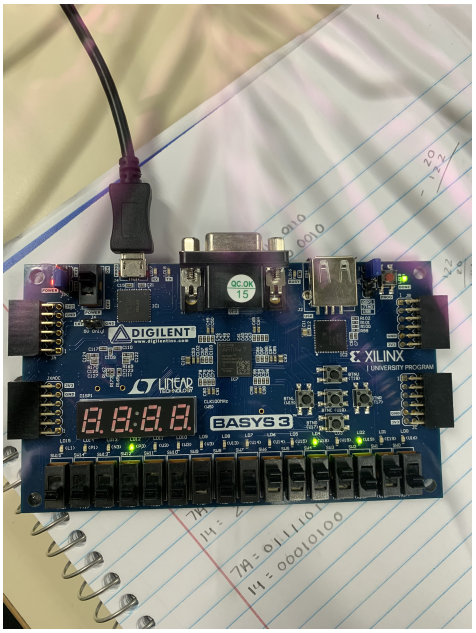
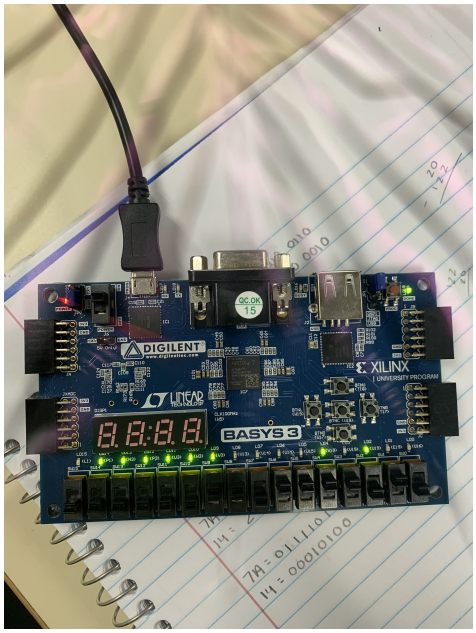
AND 14 and 7A	OR 14 and 7A
	

Table 5: Board Pictures

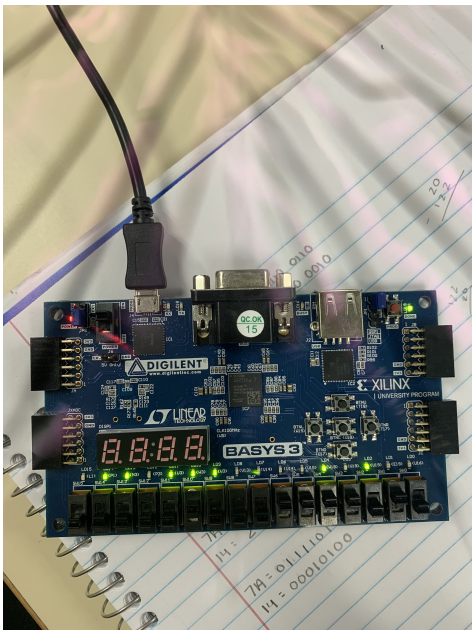
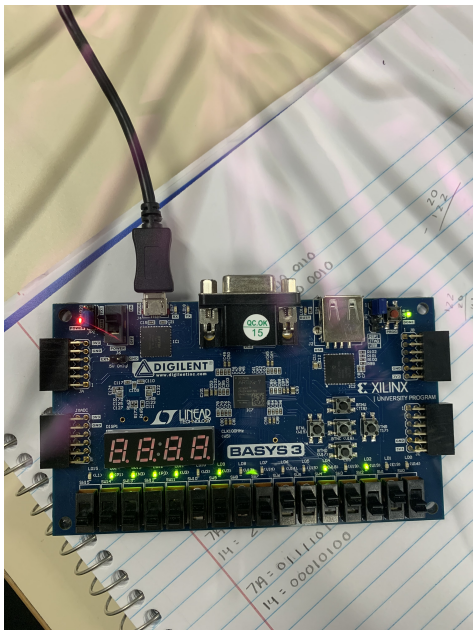
XOR 14 and 7A	default for 14 and 7A
	

Table 6: Board Pictures

Code

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

// Company:
// Engineer:
//
// Create Date: 10/22/2020 11:03:46 AM
// Design Name:
// Module Name: register
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module register #(parameter N = 8)
    (
    input clk, rst, en,
    input [N-1:0] D,
    output reg [N-1:0] Q
    );

    always @(posedge clk, posedge rst)
    begin
       if (rst==1)
          Q <= 0;
       else if (en==1)
          Q <= D;
    end

endmodule
```

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

// Company:
// Engineer:
//
// Create Date: 10/22/2020 11:07:10 AM
// Design Name:
// Module Name: register_test
```

```
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module register_test();

   reg [3:0] D_t;
   reg clk_t, en_t, rst_t;
   wire [3:0] Q_t;

   register #(.N(4)) dut(.D(D_t), .clk(clk_t), .en(en_t), .rst(rst_t), .Q(
      Q_t));

   always begin
      clk_t = ~clk_t; #5;
   end

   initial begin
      clk_t = 0; en_t = 0; rst_t = 0; D_t = 4'h0; #7;
      rst_t = 1; #3;
      D_t = 4'hA; en_t = 1; rst_t = 0; #10;
      D_t = 4'h3;    #2;
      en_t = 0;      #5;
      en_t = 1;      #3;
      D_t = 4'h0;    #2;
      en_t = 0;      #10;
      en_t = 1;      #2;
      D_t = 4'h6;    #11;
      $finish;
   end

endmodule
```

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

// Company:
// Engineer:
//
// Create Date: 10/22/2020 11:24:25 AM
// Design Name:
```

```
// Module Name: alu
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module alu #(parameter N = 8)
    (
    input [N-1:0] in0,
    input [N-1:0] in1,
    input [3:0] op,
    output reg [N-1:0] out
    );

    parameter ADD = 0;
    parameter SUB = 1;
    parameter AND = 2;
    parameter OR = 3;
    parameter XOR = 4;

    always @*
    begin
       case(op)
          ADD: out = in0 + in1;
          SUB: out = in0 - in1;
          AND: out = in0 & in1;
          OR: out = in0 | in1;
          XOR: out = in0 ^ in1;
          default: out = in0;
       endcase
    end

endmodule
```

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

// Company:
// Engineer:
//
// Create Date: 10/22/2020 11:30:52 AM
// Design Name:
```

```
// Module Name: alu_test
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module alu_test();

    reg [7:0] in0_t;
    reg [7:0] in1_t;
    reg [3:0] op_t;
    wire [7:0] out_t;

    alu dut (
       .out(out_t),
       .in0(in0_t),
       .in1(in1_t),
       .op(op_t)
    );

    initial begin

       in0_t = 4'h6; in1_t = 4'h2; op_t = 4'h0; #10;
       in0_t = 4'h6; in1_t = 4'h2; op_t = 4'h1; #10;
       in0_t = 4'h6; in1_t = 4'h2; op_t = 4'h2; #10;
       in0_t = 4'h6; in1_t = 4'h2; op_t = 4'h3; #10;
       in0_t = 4'h6; in1_t = 4'h2; op_t = 4'h4; #10;
       in0_t = 4'h6; in1_t = 4'h2; op_t = 4'h5; #10;
       $finish;

    end

endmodule
```

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

// Company:
// Engineer:
//
// Create Date: 10/22/2020 12:23:55 PM
// Design Name:
```

```
// Module Name: top_lab9
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
   //////////////////////////////////////////////////////////////////////////////////



module top_lab9(
    input btnU,
    input btnD,
    input [11:0] sw,
    input clk,
    input btnC,
    output [15:0] led
    );

    wire [7:0] reg0_out;
    wire [7:0] alu_out;
    wire [8:0] reg1_out;

    register Reg0 (
       .D(sw[7:0]),
       .en(btnD),
       .clk(clk),
       .rst(btnC),
       .Q(reg0_out[7:0])
    );

    alu ALU(
       .in1(reg0_out[7:0]),
       .in0(sw[7:0]),
       .op(sw[11:8]),
       .out(alu_out[7:0])
    );

    register Reg1 (
       .D(alu_out[7:0]),
       .en(btnU),
       .clk(clk),
       .rst(btnC),
       .Q(reg1_out[7:0])
    );

    assign led[7:0] = reg0_out[7:0];
```

```
    assign led[15:8] = reg1_out[7:0];


endmodule
```