Lab 6 report

Conner Yin, Yen-Jung(Tim) Lu

1. Include: Introduction, procedure, results, conclusions and references.

Introduction:

In this lab, students would learn how to design and test a 4-bit arithmetic logic unit(ALU) with adder/subtractor functions. Students will use these functions to create a mini-calculator. MUX hierarchy will also be utilized in the design of the arithmetic logic unit.

Procedure:

First of all, we need to compile and simulate the verilog code of *myALU4,* the simulator output will be Y[0], and the inputs are A[0],B[0], P[2], and P[1]. After we are done with the simulation, we connect the inputs and outputs for the ALU on the schematic. As the schematic is created and compiled, we would download the circuit to the FPGA and start to do the testing to check the arithmetic operation by using the test plan. We added a 4-bit wide 2:1 MUX and a *8dff* component to the schematic after the testing. Later on, we rearranged the inputs and outputs of the 8dff component and the ALU. Afterward, we implemented another testing to check on the functionality of the mini-calculator.

Results:

We successfully implemented the functions of the mini-calculator with the circuit. After the second modification with the schematic, we make sure that the user is able to save the result of the calculation in the memory and show it on the seven segment display.

Conclusions:

In this lab, we learned how to implement the basic calculations on the circuit by designing the schematics and verilog codes in Quartus. Moreover, we also learned the method to arrange the MUX hierarchy to set the seven segment display to show the saved value of the calculation.

References:

Appendix A, Table of 4-bit Unsigned Numbers and 4-bit Two's Complement Signed Numbers

2. Include schematic, Verilog code, test plan, and results for each operation.

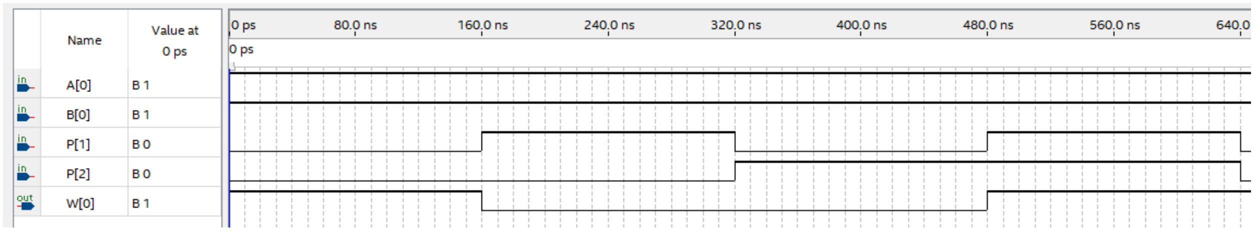


Fig 1. Simulation results for myALU4 inner-working test (part 1)

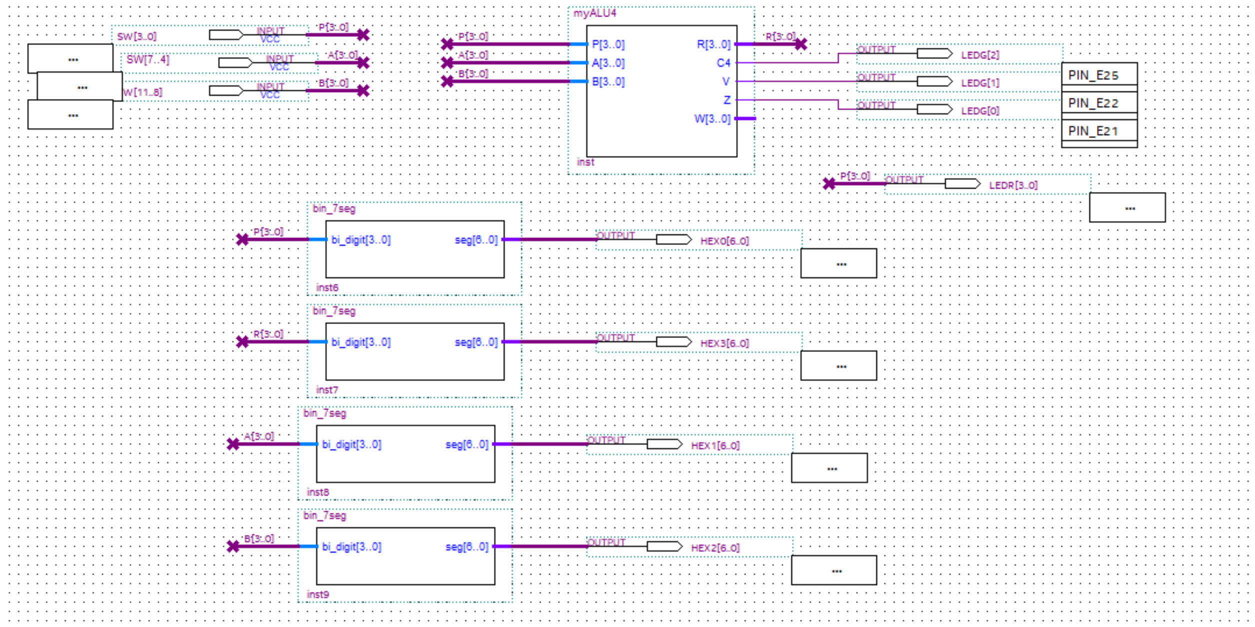
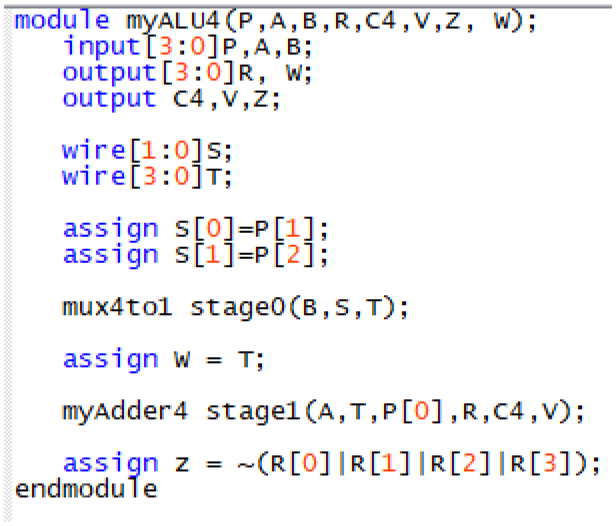
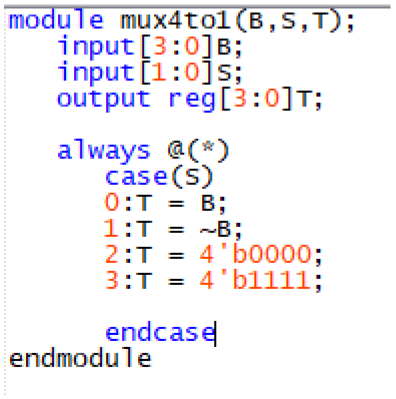


Fig 2. Schematic for small myALU4 (part 1)



```
module myALU4(P,A,B,R,C4,V,Z, W);
    input[3:0]P,A,B;
    output[3:0]R, W;
    output C4,V,Z;

    wire[1:0]S;
    wire[3:0]T;

    assign S[0]=P[1];
    assign S[1]=P[2];

    mux4to1 stage0(B,S,T);

    assign W = T;

    myAdder4 stage1(A,T,P[0],R,C4,V);

    assign Z = ~(R[0]|R[1]|R[2]|R[3]);
endmodule
```

Fig 3. Verilog code for myALU4



```
module mux4to1(B,S,T);
   input[3:0]B;
   input[1:0]S;
   output reg[3:0]T;

   always @(*)
      case(S)
      0:T = B;
      1:T = ~B;
      2:T = 4'b0000;
      3:T = 4'b1111;

      endcase
endmodule
```

Fig 4. Verilog code for mux4to1

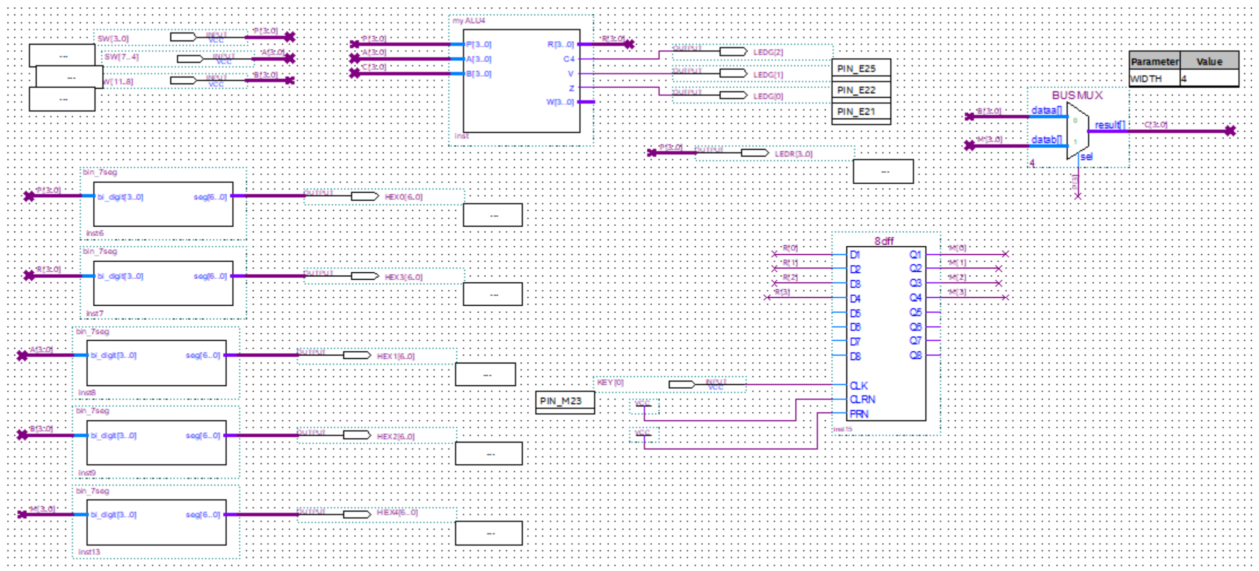


Fig 5. Schematic for myALU4 with the memory save implementation (part 2)

3. Describe how negative results appear in the seven-segment display.

Negative numbers appear as the hex value of F - the absolute value of the number + 1 (e.g. -3 would appear as F-3+1 = D). Another way to calculate it is to invert the value and add 1 (2's complement form)

4. List three pairs of A, B inputs that will cause the Z output to be 1 when the operation is P = 0 0 0 0 and C0 is 0.

P = 0000 is the add function, and Z will output 1 only if all sum values are 0. Therefore the sum must either be 0, or all the numbers must be 0 with carry-over. Therefore, some possible solutions are (0,0), (D, E), (F, F)

5. List three pairs of A, B inputs that will cause the Z output to be 1 when the operation is P = 0 0 1 1 and C0 is 0.

P = 0011 is a subtract function, and Z will output 1 only if all sum values are 0. Values of sum will only be equal if A = B. Therefore, some possible solutions are (1,1), (3,3), (8, 8)

6. If the B input switches are zero, describe a sequence of operations using memory that you could use to get a minicalculator output of –A.

- Set P to 0100 (transfer the value of A)
- Set the memory to the value of the sum (memory copies A)
- Set A to 0
- Set P to 0011 (mimics the subtraction function)

7. How would you connect two 4-bit ALUs to make an 8-bit ALU?

The 4 LSBs and c0 will be inputted into one instance of a 4-bit ALU. The 4 MSBs will be inputted into the second instance of the 4-bit ALU, and the c4 output from the initial instance of the 4-bit ALU will be inputted as the c0 for the second instance.