CPEG324 Lab 4: Single-Cycle Calculator in VHDL

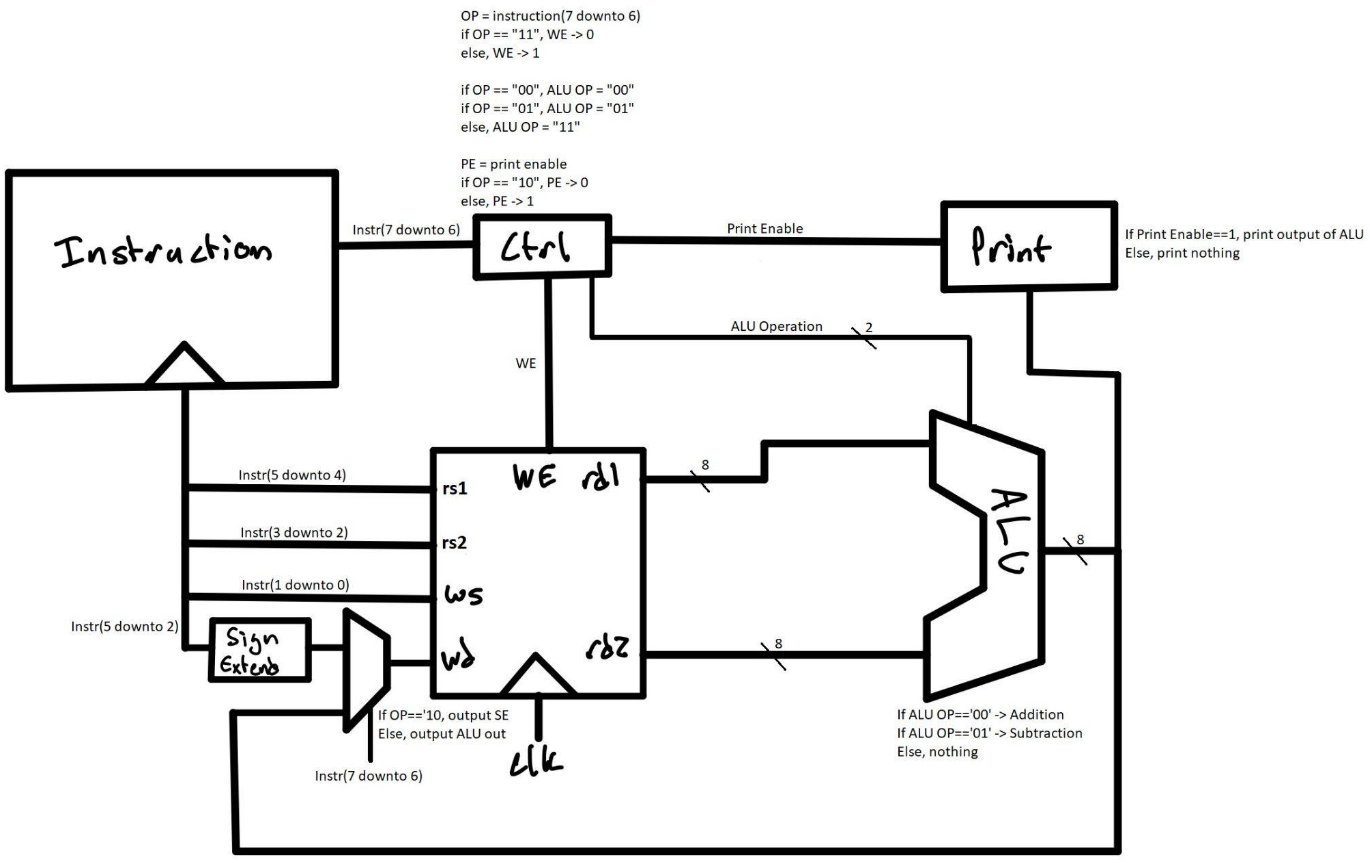
By: James David and Ethan Conway

Abstract: The objective of this lab is to implement a calculator based on our calculator ISA design. We should also develop a VHDL testbench to test the functionality and verify the correctness of our implementation with VHDL simulation.

Division of Labor: The division of labor for this project was split mainly between the circuit designing and the implementation. Ethan focused on the designing and drawing of the circuit, while James did the implementation of the circuit with some help from Ethan.

Detailed Strategy/Results:

Circuit Design:



Description:

- The main components of this datapath are the instruction fetch, the register file, the ALU, the controller, and the Print Module. This circuit does add, sub, load immediate, and print instructions. The register reads from the last 6 bits of the instruction to determine rs1, rs2, and ws. The register will write data from wd to the register selected from ws when there is a rising edge of clock and if WE is high, which is determined by the instruction used. The ALU will take in 2 8 bit inputs, and an opcode input. The ALU has 3 operations, add, sub, and nothing. The operation used is determined by the instruction. The print module will print the value from the ALU if print enable is high. The controller takes in the opcode as input, and decides WE, ALU opcode, and print enable. The mux before WD determines if wd will be reading from a sign-extended input from the instruction, or the output of the ALU.

Instruction formats for ISA:

Add: Opcode 00

OP | rs1 | rs2 | ws

Add the contents of register selected by rs1 & rs2, and store the 8 bit signed result in register selected by ws

Sub: Opcode 01

OP | rs1 | rs2 | ws

Subtract the contents of register selected by rs1 & rs2, and store the 8 bit signed result in register selected by ws

Load Word: Opcode 10

OP | int | ws

Load immediate signed int value to register selected by ws. Int will be sign extended from 4 bits to 8 bits

Print: Opcode 11

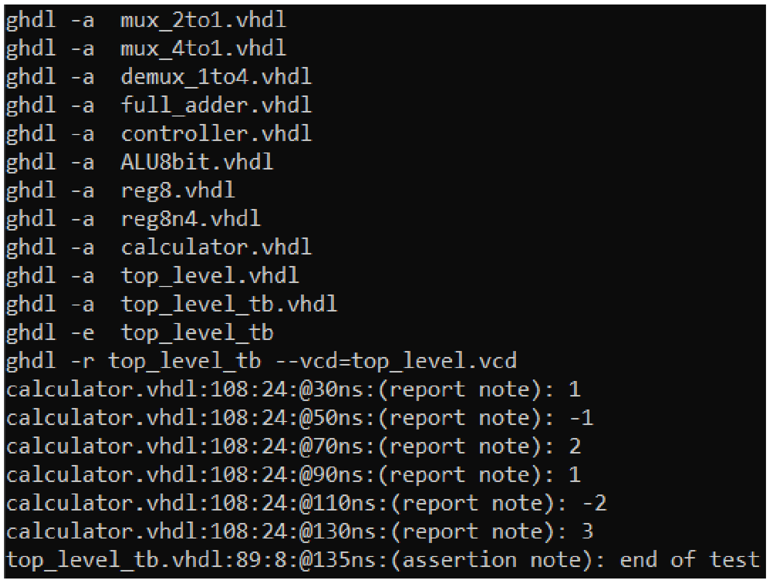
OP | rs1 | 0000

Print contents of register selected by rs1. Value is reported as a signed integer if simulated, and outputs 8 bit result to LEDs if on the FPGA board.

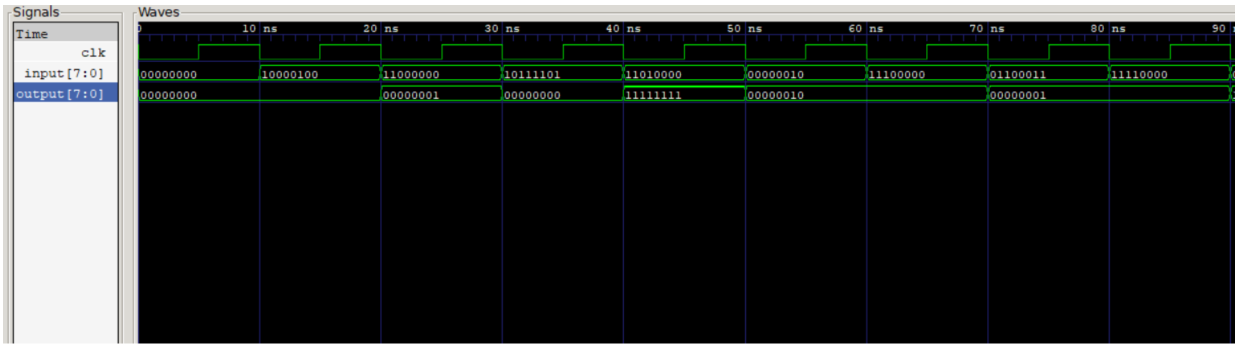
ISA Testbench:

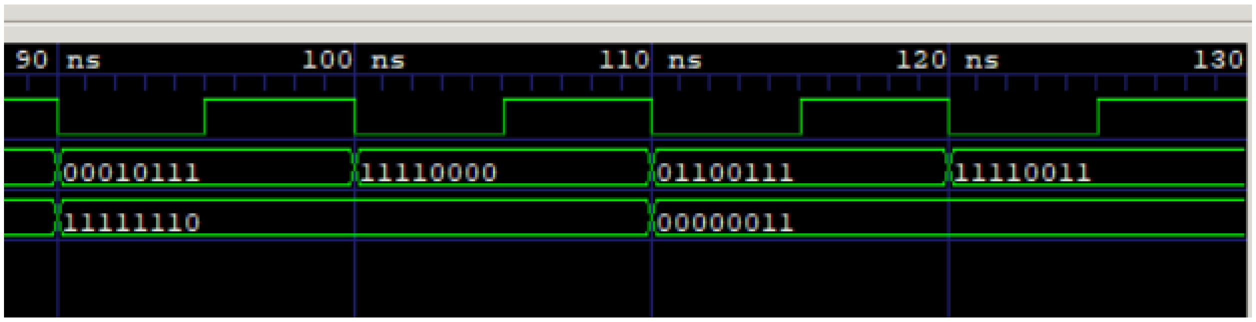
```
00000000 #add rs1 rs1 rs1

10000100 #lw 0001 rs1

11000000 #pr rs1, output is 1

10111101 #lw 1111 rs2

11010000 #pr rs2, output is -1

00000010 #add rs1 rs1 rs3

11100000 #pr rs3, output is 2

01100011 #sub rs3 rs1 rs4

11110000 #pr rs4, output is 1

00010111 #add rs2 rs2 rs4

11110000 #pr rs4, output is -2

01100111 #sub rs3 rs2 rs4

11110000 #pr rs4, output is 3
```

Testbench Output and Waveform:



```
ghdl -a  mux_2to1.vhdl
ghdl -a  mux_4to1.vhdl
ghdl -a  demux_1to4.vhdl
ghdl -a  full_adder.vhdl
ghdl -a  controller.vhdl
ghdl -a  ALU8bit.vhdl
ghdl -a  reg8.vhdl
ghdl -a  reg8n4.vhdl
ghdl -a  calculator.vhdl
ghdl -a  top_level.vhdl
ghdl -a  top_level_tb.vhdl
ghdl -e  top_level_tb
ghdl -r top_level_tb --vcd=top_level.vcd
calculator.vhdl:108:24:@30ns:(report note): 1
calculator.vhdl:108:24:@50ns:(report note): -1
calculator.vhdl:108:24:@70ns:(report note): 2
calculator.vhdl:108:24:@90ns:(report note): 1
calculator.vhdl:108:24:@110ns:(report note): -2
calculator.vhdl:108:24:@130ns:(report note): 3
top_level_tb.vhdl:89:8:@135ns:(assertion note): end of test
```





Optional Task:

- We chose to complete Task 2 of this lab, so we put the calculator circuit on a FPGA board. For the FPGA board, we chose to use the Zybo Z7-10 board. The main board only has 4 switches and 4 main leds, so we have 2 PMODs, one for adding more switches, and one for adding more LEDs. The switch PMOD adds 4 more switches, and the LED

PMOD adds 8 more LEDs. The circuit logic stayed mostly the same from simulation to FPGA board, but we had to add more. We have the input to the circuit be decided by 8 total switches, and 1 button (BTN0). The switches represent the 8 bits of the instruction, and the button represents clock. The output of the circuit is 8 bits, which is displayed on the 8 LEDs added by the PMOD. LED0 on the board will light up when BTN0 is pressed to represent clock. We mapped pins for each input and output in the constraints file for the Zybo Z7-10, and run the code. The circuit runs perfectly, but we needed to add button debouncing for the clock button. All code and constraints used are in the zip file included below, along with the bitstream created by Vivado. Picture of board and layout of Switches is found in Appendix III

Conclusion: Overall, I believe that we completed this project to the best of our ability. I believe that we successfully implemented a calculator based on our calculator ISA design. I also believe that we successfully developed a VHDL testbench to test the functionality of our calculator, and successfully verified the correctness of our implementation. Given more time, I believe that we could have implemented a branch/jump instruction into our implementation. Although, I think we did a good job of implementing the required part of the calculator onto our FPGA board. The area of the project that gave us the most difficulty was the implementation of the calculator onto the FPGA board. It was tough trying to relearn the software that was used to program the FPGA board, but I believe that we did a good job on the implementation in the end.

Appendix I: Notebooks

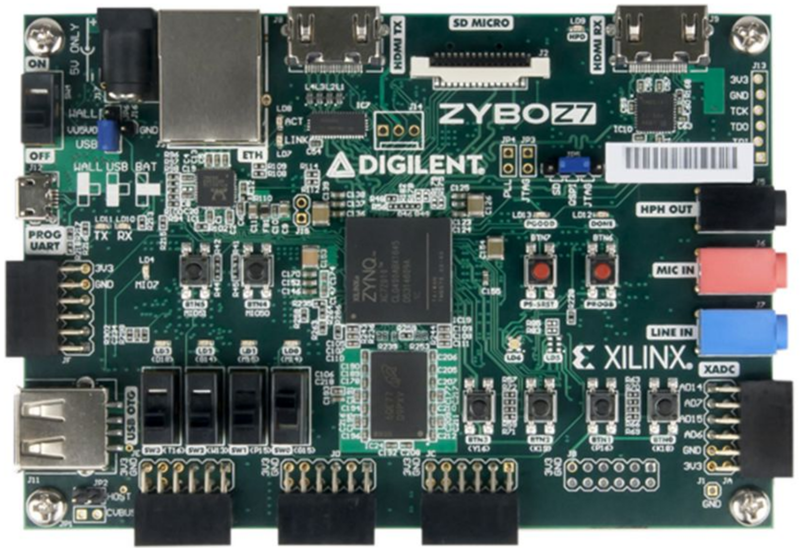
- James David: 12 Total Hours Spent
- Ethan Conway: 11 Total Hours Spent

Appendix II: VHDL Files

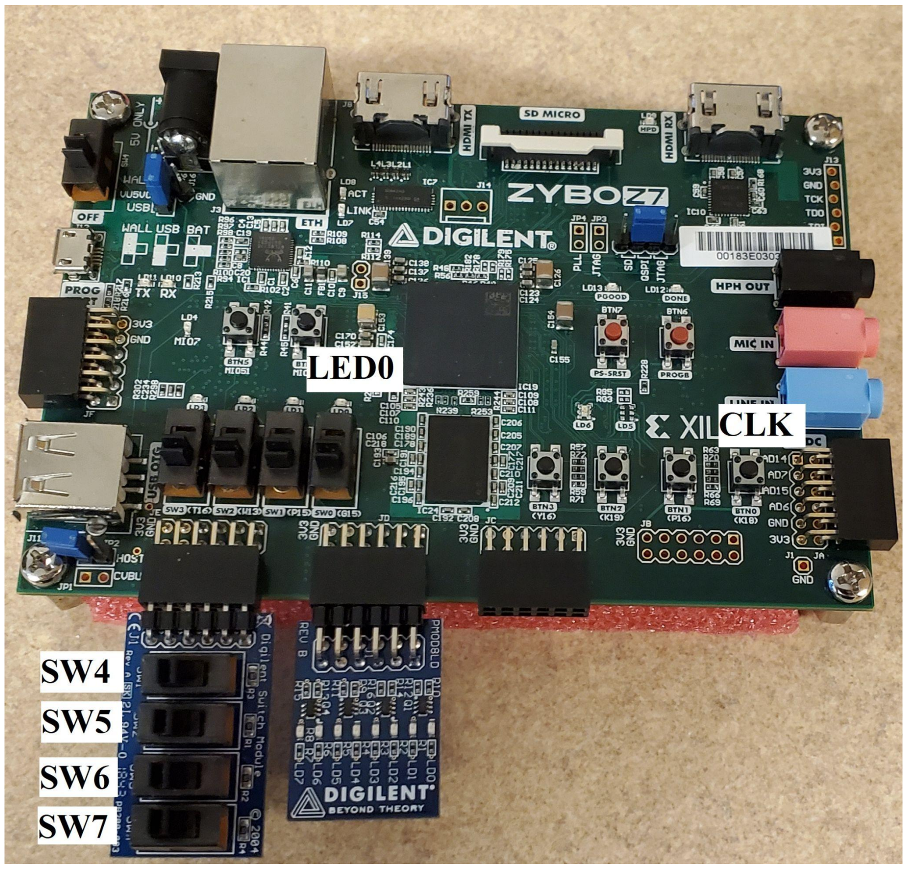
- Waveform
- Simulation files
- FPGA implementation files

Appendix III: Zybo Z7-10 Board and PMODs

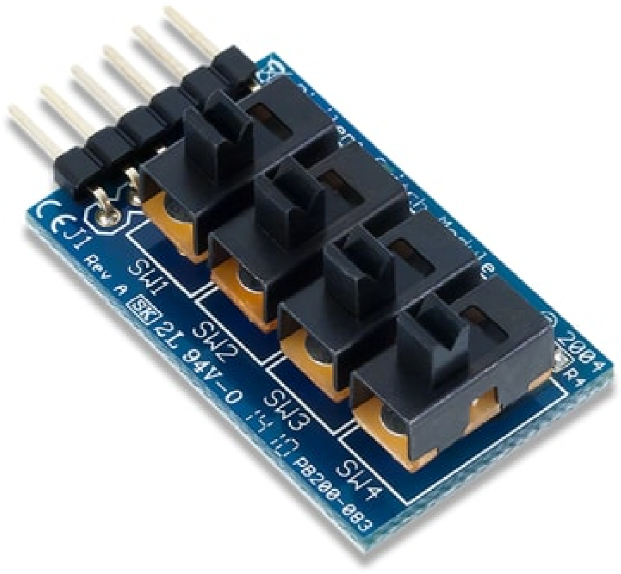
- Main FPGA Board: Zybo Z7-10



- Main board with PMOD additions and Labels:



- 4 Switch PMOD (plugged into Port JE)



- 8 LED PMOD used: