INDIAN INSTITUTE OF TECHNOLOGY JODHPUR

Lab2 – SystemC RISC-V
EEL7210 – Hardware Software Co-Design

Submitted by Rohit Mathur (M21AIE249)

Maximum Points: 10 Deadline: Feb 5, 2023

The deliverables for this assignment are at the following github repo:

https://github.com/m21aie249/eel7210-lab2

Problem Statement:
You are provided the code for Systemc-based RISCV processor at here:

https://github.com/binodkumar23/IITJ_IDDEDA/tree/main/SystemC_RISCV
The documentation for the code is available at here:
https://github.com/binodkumar23/IITJ_IDDEDA/blob/main/SystemC_RISCV/Design_Project_Report.pdf

Tasks:

a) **Simulate the processor design and understand the design code along with testbench.**

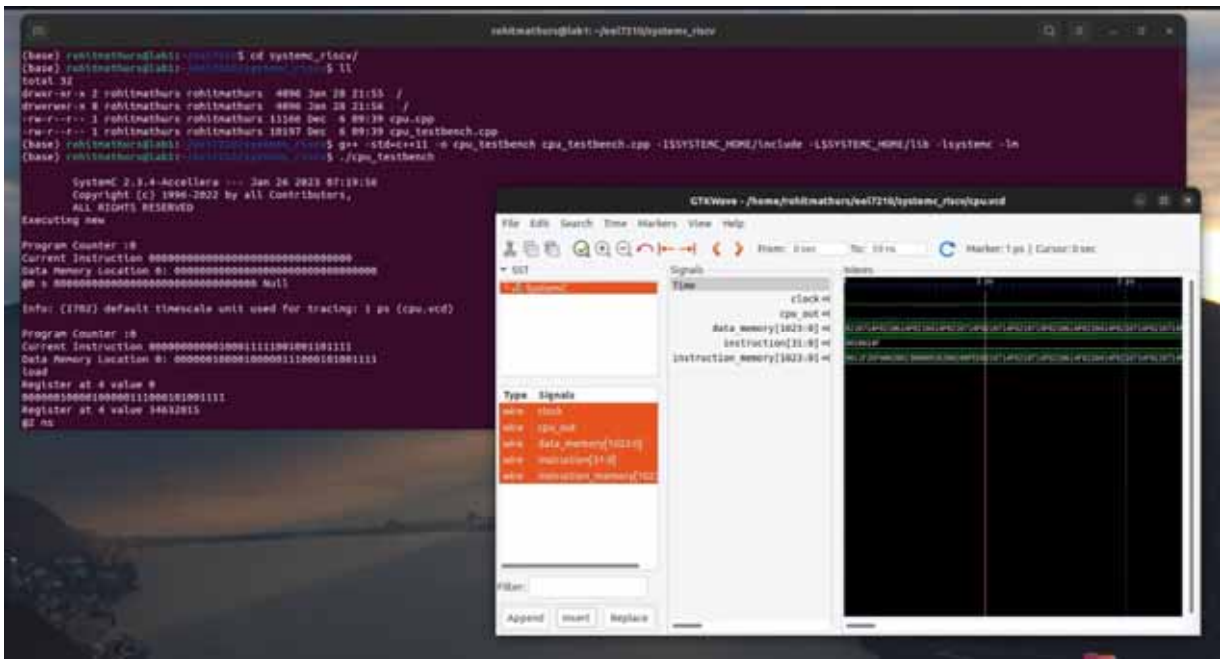
The code at the given github repo was forked over, unnecessary files removed and then compiled and simulated.

Commands used:

g++ -std=c++11 -o cpu_testbench cpu_testbench.cpp -I$SYSTEMC_HOME/include -L$SYSTEMC_HOME/lib -lsystemc -lm

./cpu_testbench

Screenshot:



b) Extend the processor design for at least 5 new instructions. Simulate these new instructions and explain this in your report.

Step 1: The original code had a few discrepancies in the opcodes and func3 and func7 assignments. All the codes were corrected[1], and simulation was redone to ensure nothing broke

Step 2: Handlers for the following instructions were added to the code:

1. sll: Shift Left Logical (R Type)
2. srl: Shift Right Logical (R Type)
3. lb: Load Byte (I Type)
4. blt: Branch < (B Type)
5. bge Branch >= (B Type)

Step 3: The testbench code has the instruction memory, the simulated program code was updated and the binary sequences [2] are as follows:

add x5, x0, x1 00000000000100000000001010110011
sub x5, x0, x1 01000000000100000000001010110011
and x5, x0, x1 00000000000100000111001010110011
or x5, x0, x1 00000000000100000110001010110011

xor x5, x0, x1 00000000000100000100001010110011
add x5, x0, x1 00000000000100000000001010110011
sub x5, x0, x1 01000000000100000000001010110011
and x5, x0, x1 00000000000100000111001010110011
sll x5, x0, x5 00000000101000000010010101100 11
srl x5, x0, x5 00000000101000001010010101100 11
beq x0, x2, 0 00000000001000000000000001100011
bne x0, x1, 0 00000000000100000001000001100011
bge x0, x2, 0 00000000001000000101000001100011
lw x3, 0 00000000000000000010000110000011
lb x3, 0 00000000000000000000000110000011
blt x2 x0 0 00000000000000001010000000110011

This sequence was repeated twice.
Registers were initially programmed with these values:

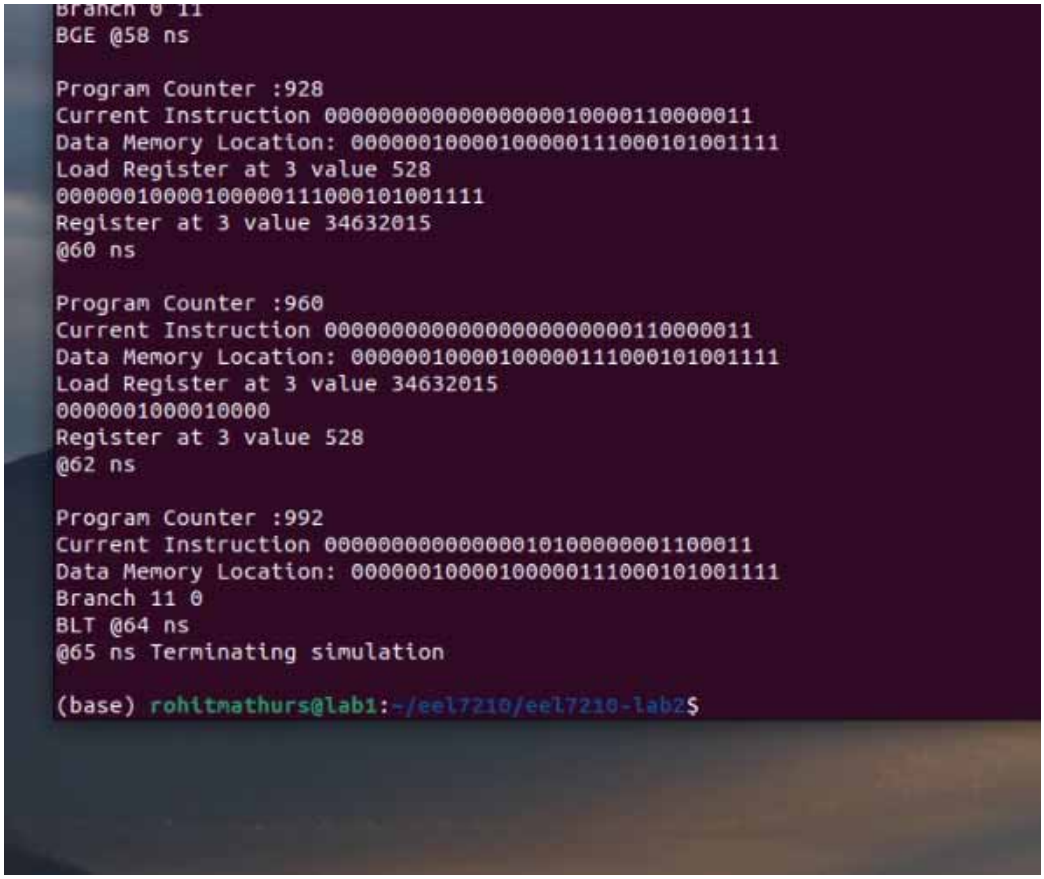
```
register_memory[0] = 11;
register_memory[1] = 11;
register_memory[2] = 0;
register_memory[3] = 3;
register_memory[5] = 1;
```

Step 4: The code was simulated, and output was checked. The simulation terminated successfully with no errors

Commands used:

g++ -std=c++11 -o cpu_testbench cpu_testbench.cpp -I$SYSTEMC_HOME/include -L$SYSTEMC_HOME/lib -lsystemc -lm

./cpu_testbench



Other Information:

1. The VCD dump does not display the "instruction" sequence correctly. The code does not handle it, although there is a provision for it
2. Some signals are unused, such as cpu_out, their purpose is not clear
3. The code is not equipped to provide meaningful messages for errors and corner cases. Did not add them for the scope of this lab exercise.

References:

[1] RISC-V Instructions: https://github.com/jameslzhu/riscv-card/blob/master/riscv-card.pdf
[2] RISC-V Instruction Encoder/Interpreter, LupLab @ University of California, Davis: https://luplab.gitlab.io/rvcodecjs/