COL216 Assignment 7 Report

Navneel Singhal, Akash S

March 2020

Implementation details

We developed a simple MIPS processor simulator in C++ simulating a subset of the MIPS instruction set (add, sub, sll, srl, sw, lw, beq, bne, blez, bgtz, j, jr, jal, addi (implemented for loading the data)).

Specifications of MIPS instructions to be simulated

1. The instructions may have commas or may have spaces.
2. The offset has to be in base 10. The instructions should be in small case.
3. The registers may be represented as $t0 or t0 or even as their corresponding numerical indices.
4. The end of instructions has to be specified by END_INSTRUCTIONS, and the end of execution (equivalent to exiting the program) has to be specified by END.

Features of our simulator

Our simulator has the following features:

1. We take an input file name as a command line argument which has the number of clock cycles each instruction takes, and those values are used in finding the statistics in the end of execution.
2. After every step, we print what the instruction has done and also the state of the register file.
3. At the end of execution we also print the state of the register file and the memory, the average IPC and the total number of clock cycles it took us.
4. We exit the program if a wrong instruction has been encountered.

Testing

Most part of this code was already made and tested (using qtspim) during assignment 3, 4, 5, 6 for interpreting the MIPS code for checking the correctness of the FPGA MIPS processor. We have submitted few testcases along with the source file. These include the following.

1. A simple test case which uses all the jump and branch instructions at least once.
2. An implementation of a recursive procedure that finds the sum of numbers from 1 to 5.
3. An implementation of a small test case.
4. Test cases from previous assignments.