Project 2. 5-Stage Pipelined Processor

Due date: 5/14 23:59

E-mail: sungho517@kaist.ac.kr

1. Overview

In the Project 2, you will design a 5-stage pipelined processor, which is compatible with Cortex-M0. ***You only need to implement the functions included in the Project 1.*** The instruction set simulator you made in the Project 1 will be useful when you debug the processor.

2. Attached Files

(1) Processor
   A. CortexM0.v – 5-stage pipelined processor **(You will modify and submit)**
   B. MemModel.v – Memory functional model for main memory
   C. RegisterFileModel.v –Registerfile functional model in the processor
   D. tb.v – Testbench file
   E. tb.f – File list for your test
   F. test.hex – Simple program that initializes registers and operates '5+7'
   G. test.dis – Corresponding disassembly
   H. ALU_TEST.v, ALU_TEST.f, ALU_TEST_inst.hex – Sample testcase

(2) Instructions for Verilog Simulation
   A. Contains instructions for Verilog simulation using 'Icarus Verilog' software in Window environment.
   B. You can use 'Icarus Verilog' if you do not have Verilog compile environment.
   C. Otherwise, you can use your compiler if you have one.

   cf) Note that if you use 'Icarus Verilog', **the file path should not contain Korean.**

3. Requirements

(1) Put all the relevant codes in 'CortexM0.v'
   A. Do not make other source files.

(2) Only the annotated part of 'your code here*' is allowed to be modified in 'CortexM0.v'
   A. Do not change in/out ports of CortexM0.v.
   B. Use given 'RegisterFileModel.v' for registerfile, and 'MemModel.v' for main memory.

(3) Do not include the Verilog System Tasks and Functions such as '$display' in 'CortexM0.v'

(4) Report file, {Student_ID}_{Name}_report.pdf, which explains your hardware including block diagram of its data-path, your test results, and reasons why you make such test program (*10 pages or less allowed)

4. Testcase

A sample testcase, ALU_TEST, is provided. It is designed to run a program involving several operations and will be included in the grading process.

Note that there are additional testcases besides ALU_TEST. Thus, passing ALU_TEST alone does not guarantee full credit if other testcases are not passed.

The following demonstrates how to execute ALU_TEST using 'Icarus Verilog' and presents the corresponding results.

```
C:₩Processor>iverilog -c ALU_TEST.f -o ALU_TEST

C:₩Processor>vvp ALU_TEST
WARNING: MemModel.v:34: $readmemh(test.hex): Not enough words in the file for the requested range [0:4095].
VCD info: dumpfile myfile.dmp opened for output.
WARNING: ALU_test.v:100: $readmemh(ALU_test_data.hex): Too many words in the file for the requested range [0:145].
ADD_REG_test passed
ADD_IMM_test passed
SUB_REG_test passed
SUB_IMM_test passed
MUL_test passed
LSL_IMM_test passed
LSL_REG_test passed
LSR_REG_test passed
LSR_IMM_test passed
AND_test passed
XOR_test passed
OR_test passed
NOT_test passed
NOT_test passed

C:₩Processor>
```

5. Submission

1. **Due date: 5/14 23:59**
2. Submit following 2 files on the KLMS :
   'CortexM0.v', '{Student_ID}_{Name}_report.pdf'
3. Assessment
   A. Correctness of operation – Several test programs will be executed and the results will be checked
   B. Quality of the source code including annotations
   C. Quality of the report
   D. **NOTE**: If you submit past the due date, your grade will be deducted
   E. **NOTE:** If you do not satisfy the requirements, your grade will be deducted
   F. **NOTE**: If you copy other's work, you will not receive any credit