Project 3
Design of A SPARCv8-Compatible Processor

Description

- In this project, you will design and implement a SPARCv8-compatible processor using your preferred HDL language and CAD tools.
- The minimum requirements is to design a processor with a 5-stage pipeline, multiple functional units with varying latencies, and direct mapped instruction and data caches
- For more points, you can add other features (see below) to your processor

Grading

- 5-Stage pipeline + direct-mapped caches (30 pts)
- 5-Stage pipeline + set-associative caches (35 pts)
- Above + super-scalar pipeline (40 pts)
- All of the above + out-of-order execution (45 pts)
- Multi-cycle divider and pipelined multiplier on top of any of the above (50 pts).
    - A pipelined multiplier accepts a new multiplication operation every cycle but generates the result after multiple cycles (just like a normal pipeline)
    - A multi-cycle divider accepts a new operation when the "start" signal is given and takes "n" cycles to finish it. After that, it can accept another operation.
- Branch prediction and speculative execution on top of any of the above (55 pts)
- SMT on top of any of the above (60 pts)

Project Report

In addition to submitting your code, you should also submit a short report with a high-level overview of your processor pipeline, the implemented features and details of each pipeline stage—in particular,

- General flavor of your processor: scalar or super-scalar, out-of-order or in-order, etc.
- Instruction and data cache details
- How your caches connect to the main memory (e.g., how you arbitrate between them)
- Functional units implementations

Project Report (2)

- Memory unit implementation
- How you handle data and control flow dependencies
- How you implement register window overflow and underflow situations
- A well-organized and comprehensive report can substantially improve your project grade. It can help me and the TA identify important aspects of your design that we might overlook otherwise
- You should **email your reports to me and the TA** by the project deadline

Target Instruction Set

Your processor should implement the user-mode (non-privileged) subset of SPARCv8 instructions. You can ignore the ISA subset related to the "Floating Point" instructions, "Alternate Address Spaces", "Ancillary State Registers" and the "Co-Processor" as well as any instruction that is marked as "privileged" in the SPARCv8 manual. Specifically, your processor needs to implement all the instructions (and the requisite architectural state) described in the following sections of the manual:

- Load/store instructions: B.1, B.4, B.7, B.8
- Arithmetic/logical/shift instructions: B.11, B.12, B.13, B.14, B.15, B.16, B.17, B.18, B.19
- Control transfer instructions: B.21, B.24, B.25, B.27
- Misc. instructions: B.9, B10, B20, B28 (only RDY), B29 (only WRY), B.30 (treat as NOP), B.31, B.32 (treat as NOP unless you have caches)

Target Instruction Set (2)

- Traps and Interrupts: Your processor need not deal with external interrupts. It should treat all "exceptions" (i.e., traps caused by instructions) as precise—no deferred interrupts. Your implementation should correctly check for and generate all the exceptions described in the semantics of your implemented instructions.
- Virtual Memory and MMU: Your processor need not include any virtual memory support. Hence, you don't need to implement an MMU and can ignore the issues related to the Address Space Identifiers (ASI).