Fall 2017 Term Project

Sabbir Ahmed

Overview

The objective of the project is to experience the design issues of advanced computer architectures through the design of a simulator for a simplified MIPS computer using high level programming languages. The specification requirements for the project are detailed in the prompt.

Simulator

The project is implemented in an object-oriented design using Python. The program initializes by parsing the input instructions and data memory files. The instructions and data memory are stored in a fixed-length array to simulate a system memory. The program then simulates the execution of the instructions to unroll any subroutines. Unrolling the instructions also resolves all conditional and unconditional branching, as well as premature halts. The unrolled instructions are then mapped to sufficient number of cycles with empty stages in a two-dimensional matrix. The program fills these uninitialized stages by iterating through the transposed matrix cycle-by-cycle as opposed to instruction-by-instruction. Once all the stages have been accounted for, the program generates the statistics required and sanitizes and formats the output before dumping to `output.txt`.

Usage

The simple helper script `make.sh` may be used to execute the simulation with the required file inputs. Running the following command will execute the simulation: `./make.sh`

The Makefile is used for testing and debugging purposes.

Requirements

- Python 2.7

Testing and Troubleshooting

Current issues with the simulator include the reliability of data forwarding and data dependencies. The pipelining of the stages of the instructions along with data hazards worked fine before the introduction of instruction and data cache misses. The pipeline currently stalls successfully for instruction and data cache misses, but provides inaccurate clock cycles for instructions such as `ADD` and `ADDI` under specific circumstances. The sample input provided resolves these issues, however, the simulator has not been tested with other instruction files.

The simulator has also been tested with the input data memory from `data.txt` being manipulated.