THE AMERICAN UNIVERSITY IN CAIRO

Computer Architecture Second Project

Tomasulo Algorithm Simulator

Fall 2019

Submitted By

Mohamed Shehawy

Ismail Shaheen

Samah Hussien

Amr Adel

Submitted To

Dr. Cherif Salama

Submission files: . (https://github.com/Mahfouz-z/Tomasulo_Simulator)

Abstract

This Project is a simulator aims at assessing the performance of a simple superscalar 16-bit RISC-V processor with hardware speculation and multi issue. The processor uses a simplified ISA that is similar to that of the Ridiculously simple computer (RISC-16). The simulator uses python to implement classes that simulate each partition of the hardware. These classes are instantiated in a main and ordered so that each class that simulate a hardware is updated every cycle. The main classes were for the ROB, the instruction queue, the reservation stations, and the memory. These classes were used to build the main that included issuing, executing, and writing.

Disclaimer: We didn't provide journals with the submission. Instead we provide here a link for the github repo that we used during the project that indicates the inputs of each team member. (https://github.com/Mahfouz-z/Tomasulo_Simulator)

Assumptions

1- The assembly code is provided in a text file

2- A configure text file is used to set the hardware size if a change is needed

3- No virtual memory

4- No floating-point instructions, registers, or functional units

5- No I/O instructions are supported

6- No interrupts or exceptions to be handled
7- For each program being executed it's assumed that its data is loaded to the memory.
8- Each Reservation station has functional unit dedicated to it
9- Assuming branch target is correct when needed

Steps

1- Decided to implement the project in python
2- Designed a class to perform the function of each hardware part
3- We implemented the issuing stage in the main
4- Implemented immediate sign based prediction
5- We implemented the Execution stage in the main
6- We implemented the Write stage in the main
7- We implemented the commit stage in the main
8- Handling branch misprediction

Implementation

Tomasulo Algorithm was implemented based on four stages: Issuing, Executing, Writing, Committing

For the Issuing stage, a class was implemented that takes the path of the assembly file as input to it. Inside the class the file is parsed and the instructions are pushed into a list where they could be issued using a getter function that takes

the pc as input to it. During a cycle our issue algorithm can do multi issue based on the number provided in the config file. The class includes all the functions needed by the main to make its checks for issuing.

For the Execution stage, a class for the reservation stations was implemented that had the functions needed to simulate the execution of instructions. The class had subclasses of functional units that produces the results after a certain number of cycles according to what was provided in the project prompt.

For the write stage, it was simply handled by feeding the values produced from the functional units classes to the reservation stations that needed these results. Also, the ROB class was fed with the data which needs to be written

For the commit stage,

Bonus Features Implemented

1- We input the code as assembly file with label branches
2- Support variable hardware organization

Simulation Results

For the first testcase (no branches, tested instructions):

```
In [587]: runfile('C:/Users/Dell/Documents/GitHub/
Tomasulo_Simulator/main.py', wdir='C:/Users/Dell/Documents/GitHub/
Tomasulo_Simulator')
Reloaded modules: DataMemClass, robClass, instructionUnit, RegFile,
RS_Class, Multipliers, NAND, Adders, BEQ, JMP, LW, SW
The time of excution is:21cycles
The IPC is:0.42857142857142855
```

For the second testcase, with branches:

```
In [589]: runfile('C:/Users/Dell/Documents/GitHub/
Tomasulo_Simulator/main.py', wdir='C:/Users/Dell/Documents/GitHub/
Tomasulo_Simulator')
Reloaded modules: DataMemClass, robClass, instructionUnit, RegFile,
RS_Class, Multipliers, NAND, Adders, BEQ, JMP, LW, SW
The time of excution is:14cycles
The IPC is:0.2857142857142857
the ratio of branch mispredictions is:0.5
```