CS425: Computer Systems Architecture

Programming Assignment 2
Assignment Date: 08/12/2023
Due Date: 22/12/2023 - 23:59

Instructions: Put your answers in a .pdf file and send them together with your other deliverables via email to HY425 course e-mail (hy425@csd.uoc.gr). Set the email subject: HY425 – Programming Assignment 2.

The Limits of ILP

The purpose of this assignment is to introduce you to architectural exploration using simulation and help you familiarise with some of the details of Out-of-Order CPU by exploring some of its design points and potential performance improvements. You have to run simulations with different parameters (the actual list of parameters can be found below) and see the impact of alternative design choices in terms of Instructions Per Cycle (IPC). Note at this point, that you are free to refer to other metrics, as well.

For the simulations you will use the gem5 simulator (www.gem5.org). Although the assignment does not require prior experience with gem5, it is recommended that you take some time to follow the gem5 tutorial:

www.gem5.org/documentation/learning_gem5/introduction/

Simulator

In this assignment you will compile gem5 to target the x86 ISA. The build requirements can be found here: www.gem5.org/documentation/learning_gem5/part1/building/
You are able to work on this assignment solely in your own environment..
You can get a snapshot of the simulator by typing the following git command:

```
git clone -b hy425_pa2 https://github.com/sotot0/gem5.git
```

This command will automatically clone from the main repository and checkout to hy425_pa2 branch of it. You will work only on this branch.
The repository contains:
1. the gem5 directory
2. and the benchmarks directory (hy425)

After you have installed the indicated dependencies, compile the simulator by typing:

```
python3 `which scons` build/X86/gem5.opt -j <Avail. Cores>
```

inside the gem5 directory. The compilation step takes time so complete it as soon as possible!

To run a batch of five (5) benchmarks with the simulator you have to use the following command inside the hy425 directory:

```
./runall.sh [List of arguments]
```

This command will run all the benchmarks while simulating a setup with an Out-Of-Order CPU (O3CPU), a good performing branch predictor (TournamentBP) and reasonable settings for caches and other components. You are encouraged to examine thoroughly the runall.sh and README files to figure out the different arguments that you will be allowed to change. The arguments that are related to the branch predictor (TournamentBP) are not allowed to change. Also, arguments that are related to cache and CPU domains are not allowed to change during the assignment. Note that the -I *[Instruction Count]* argument means that for each benchmark, gem5 simulates *[Instruction Count]* instructions. You can change this parameter during early testing but for generating final numbers you should have at-least 50 million instructions. If you set a very large number of instructions the simulation time will increase dramatically. (Even with the parameter set to 50 million instructions you may experience long simulation times especially when running benchmark1 and benchmark4). Ensure that your comparisons are based on the same instruction count between different benchmarks and/or settings to extract uniform and fair results. Also, ensure that this parameter is large enough to extract meaningful results. An estimated number would be between 50-80 million instructions. In any case, this is another reason to begin the assignment as soon as possible!

Explore OoO CPU Design Points

Your main task is to explore the given design space of an OoO CPU by simulating benchmarks and observe mainly the corresponding IPC values. Study the slides presented in the class and the paper from David W. Wall: "Limits of Instruction-Level Parallelism" - you can find it on the website of the course.

The list of the specified design parameters is the following:

- LSQ_ENTRIES: Number of LoadQueue/StoreQueue entries (same value for both)
  - Range: 16, 32, 64, 128
  - Default: 16
  - Today: up-to 128

- ROB_ENTRIES: Number of Reorder Buffer entries
  - Range: 64, 128, 256, 512
  - Default: 64
  - Today: up-to 384

- FUINT_NUM: Number of Simple Int Functional Units (ADD etc.)
  - Range: 2, 4, 8, 16
  - Default: 2
  - Today: up-to 5

- FUCOM_NUM: Number of Complex Int Functional Units (DIV, MULT etc.)
  - Range: 2, 4, 8
  - Default: 2
  - Today: up-to 2

- FUFP_NUM: Number of Floating Point Function Units (FADD etc.)
  - Range: 2, 4, 8
  - Default: 2
  - Today: up-to 3

- FULS_NUM: Number of Load/Store Functional Units (same value for both)
  - Range: 1, 2, 4
  - Default: 1
  - Today: up-to 3

Provide your answers in a PDF report and use graphs and/or tables for the following:

1. Explore IPC improvements using the ranges of these parameters compared to the default values. One suggested plan is to tweak each parameter while having the others set to default. Once you pick the highest performing value for each parameter, you can use this value to continue the exploration. However, you are free to suggest your own exploration plan!

2. Find the combination of parameter values with the minimum product MiP (i.e. MiP = LSQ_ENTRIES * ROB_ENTRIES * FUINT_NUM * FUCOM_NUM * FUFP_NUM * FULS_NUM) that maximises IPC using only realistic ranges. However, you are encouraged to report any performance gain with non-realistic parameters (if any).

3. Is there a saturation point of each benchmark's IPC? Explain your answers.

It is important to get familiar with the generated m5out directories under /benchmarkN directories after each run. These directories include files showing the simulated configuration and their statistical information.

Bonus Question

Use the generated statistics (e.g. instruction mix) to infer information and/or other characteristics about the behaviour of each benchmark. Does your answer to question 3 also help to infer such characteristics? Explain your approach and observations.

You have to deliver one zip/tar.gz with the following files:
- Your report in PDF format
- Any custom scripts that you used
- Any other files that you find useful
- m5out directories that contain useful statistics