# ECE 201 Spring 2022 - Project 4

Assigned by Prof. Wujie Wen
**Due May. 6th, 2022 23:59pm**

## Introduction

In this project, you will be working on the Champsim simulator. You will need to understand and modify (part of) the simulator to evaluate how different cache parameters impact performance. You will also need to analyze the simulation results and explain why certain scenarios happen.

- Please finish one by one in sequence.
- Keep all your source codes in the workstation, and include your running directory in the submitted report (e.g. **/proj/ece201-spring2022/XXX/Proj4)**. TA and instructors will go to your running directory and conduct simulations for grading purposes.
- Submit a project report (pdf format, one file including all parts) through coursesite.

## Instructions before you start

This programming assignment will use the Champsim simulator. Like project 1 and 2, you should use ECE teaching servers to finish project4, please conduct the following steps sequentially:

1. Create a working folder for proj4 and download Champsim following below commands:

   cd /proj/ece201-spring2022/xxx/ (xxx is your user id)

   mkdir proj4

   cd proj4

   ***git clone https://github.com/ChampSim/ChampSim.git*** (This command clones the ChampSim repo, you can check more details/instructions about ChampSim on https://github.com/ChampSim/ChampSim)

2. Run the following commands and copy config.sh from ClassShare folder. Note this step is essential as it provides the correct setting for python version (python3.6). The original downloaded config.sh does not work in our environment due to lower python version 3.

   ```
   cd ChampSim/
   cp /proj/ece201-spring2022/ClassShare/config.sh ./
   ```

3. Run the following commands to link different traces or benchmarks stored in ClassShare folder to your local running folder. 4 traces are selected for your cache performance evaluation in this project.

```
ln -s /proj/ece201-spring2022/ClassShare/600.perlbench_s-210B.champsimtrace.xz trace1.xz
ln -s /proj/ece201-spring2022/ClassShare/657.xz_s-56B.champsimtrace.xz trace2.xz
ln -s /proj/ece201-spring2022/ClassShare/649.fotonik3d_s-1B.champsimtrace.xz trace3.xz
ln -s /proj/ece201-spring2022/ClassShare/429.mcf-22B.champsimtrace.xz trace4.xz
```

4. For this programming assignment, the baseline configuration is a single-core processor with bimodal branch predictor, no L1I/L1D/L2C/LLC data prefetchers, and LRU replacement policy for the last-level cache. Build the baseline setting and make file by running the following commands:

***./config.sh champsim_config.json***

***scl enable devtoolset-7 bash***

***make***

After typing the above three commands, there will be a file named ***champsim*** located in ***/bin*** folder, now the bin/champsim becomes executable based on your configuration.

5. Run simulations and collect simulation results (e.g. cache miss number) for different benchmarks by using the following command as a reference:

   **bin/champsim --warmup_instructions 10000000 --simulation_instructions 200000000 trace1.xz > result-trace1.txt**

It means you run the simulation with settings:

warm up **10 million** instructions

simulate **200 million** instructions

on the trace ***trace1.xz (or* 600.perlbench_s-210B.champsimtrace.xz trace1.xz)*.**

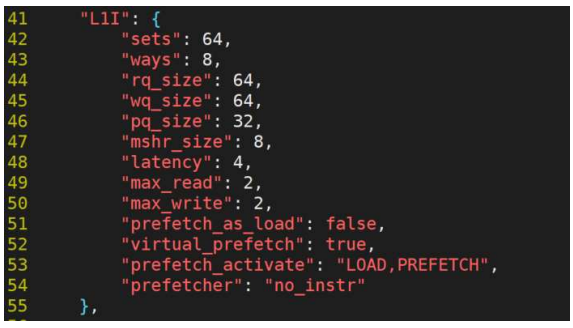
*It might take a while (e.g. 10-30min) to finish the simulation and get the result.*

You can collect the results related to L1 instruction cache (L1I), L1 data cache (L2D), L2 cache (L2C), Last level cache (LLC) etc. from the file **result-trace1.txt.** Note all these results are dedicated to trace 1 based on the given configuration file champsim_config.json. Different traces and configurations will lead to different results.

6. For different question requirements about the parameter settings. You can change the corresponding parameters in the ***champsim_config.json*** file (see snapshot below), for example:

   L1I cache set number -> “sets”, L1I cache associativity number->“ways”
   L1I cache speed -> “latency”
   L1D cache set number -> “sets”, L1D cache associativity number->“ways”
   L1D cache speed -> “latency”

Note after you change the configuration in **champsim_config.json** file, you should **rebuild** the ***/bin/champsim*** **by repeating step 4** before you run the step 5 for simulation/results collection.



```
41     "L1I": {
42         "sets": 64,
43         "ways": 8,
44         "rq_size": 64,
45         "wq_size": 64,
46         "pq_size": 32,
47         "mshr_size": 8,
48         "latency": 4,
49         "max_read": 2,
50         "max_write": 2,
51         "prefetch_as_load": false,
52         "virtual_prefetch": true,
53         "prefetch_activate": "LOAD,PREFETCH",
54         "prefetcher": "no_instr"
55     },
```

```
57     "L1D": {
58         "sets": 64,
59         "ways": 12,
60         "rq_size": 64,
61         "wq_size": 64,
62         "pq_size": 8,
63         "mshr_size": 16,
64         "latency": 5,
65         "max_read": 2,
66         "max_write": 2,
67         "prefetch_as_load": false,
68         "virtual_prefetch": false,
69         "prefetch_activate": "LOAD,PREFETCH",
70         "prefetcher": "no"
71     },
```

## Grading Criteria

Programming assignments should be done **individually**. Your submission will be graded based on the following criteria:

1. Data presentation. Are simulation results clearly presented?

2. Performance metrics selection. Are the selected performance metrics and baseline appropriate?

3. Result analysis. Describe your observations, analyze the results, and discuss whether and why the results make sense.

Important Tips: Start as early as possible, it will take a while for you to conduct simulations of the 4 benchmarks for each problem.

## Problem #1: Cache Miss Rate (28 points)

1. Using the baseline setting, collect and compare the cache miss rates of the L1-data, L1-instruction, L2, and last-level cache, for the 4 given traces (12 points).

2. Qualitatively, does it make sense to split L2 cache into L2-D and L2-I caches? Why or why not? (16 points)

## Problem #2: L1 Data Cache of 4 traces by varying the configuration (36 points)

1. What is the performance impact of a slower L1 data cache? e.g. double the latency in ***champsim_config.json*** *and check/compare the results* (12 points).

2. What is the performance impact of a smaller L1 data cache? e.g. decrease cache size by half (associativity) in ***champsim_config.json*** *and check/compare the results*. (12 points)

3. How important is associativity of L1 data cache? Conventional wisdom is that associativity is only important when the cache is small, is this true based on your results? (12 points)

## Problem #3: L1 Instruction Cache of 4 traces by varying the configuration (36 points)

1. What is the performance impact of a slower L1 instruction cache? e.g. double the latency in ***champsim_config.json*** *and check/compare the results.* (12 points)

2. What is the performance impact of a smaller L1 instruction cache? decrease cache size by half (associativity) in ***champsim_config.json*** *and check/compare the results.* (12 points)

3.How important is the associativity of L1 instruction cache? (12 points)