The George Washington University
School of Engineering & Applied Science
Electrical & Computer Engineering Department

Instructor: Prof. Louri
Semester: Fall 2022
Course: Computer Architecture & Design ECE 6005 / ECE 4535

Lab Assignment 3

Due Date: November 16
Instructions: Submit **one report** (preferably pdf format) and **separate codes in plain text files**, please do not compress them to a zip file. In the report, include source code with comments to explain the key steps, figures of your analysis. Make sure your source codes in plain text files are executable. Your codes will be downloaded and executed on simulator for functional testing.

The **purpose** of this assignment is to give you experience compiling and running the Gem5 CPU simulator.

Step 1: Write an interesting application

Write a program that implements Sieve of Eratosthenes and outputs one single integer at the end: the number of prime numbers <= 100,000,000. Compile your program as a static binary. The output should be: 5761455.

Compile it without any optimization:

'g++ -o program program.cc' **OR** 'g++ -O0 -o program program.cc'

As we discussed in the class, there are four different types of CPUs. The table below shows a brief list of them:

CPU	
AtomicSimpleCPU	The default one. Memory accesses happen instantaneously. The fastest simulation, but not realistic at all.
TimingSimpleCPU	Non-Memory instructions=1 cycle. Memory instructions depend on the memory model.
MinorCPU	Generic In-order Multi-issue core. Fixed 4-stage pipeline.
DerivO3CPU	Most detailed OOO core in gem5.

Variate the CPU clock from 1 GHz to 4 GHz (in steps of 500 MHz) with three CPU models (AtomicSimpleCPU, TimingSimpleCPU, and MinorCPU). Report (plot) the simulation times and clock numbers of the execution program for 21 (3*7) configurations.

Gem5 has following memory models (see src/mem/DRAMCtrl.py):

1) DDR3_1600_8x8 : (1.6 x 8 x 8⁄8 = 12.8 GBps)
2) HMC_2500_1x32 : (2.5 x 1 x 32⁄8 = 10.0 GBps)
3) DDR3_2133_8x8 : (2.133 x 8 x 8⁄8 = 17.0 GBps)
4) DDR4_2400_16x4 : (2.4 x 16 x 4⁄8 = 19.2 GBps)
5) DDR4_2400_8x8 : (2.4 x 8 x 8⁄8 = 19.2 GBps)
6) DDR4_2400_4x16 : (2.4 x 4 x 16⁄8 = 19.2 GBps)
7) LPDDR2_S4_1066_1x32 : (1.066 x 1 x 32⁄8 = 4.3 GBps)
8) LPDDR3_1600_1x32 : (1.6 x 1 x 32⁄8 = 6.4 GBps)
9) GDDR5_4000_2x32 : (4.0 x 2 x 32⁄8 = 32.0 GBps)
10) HBM_1000_4H_1x128 : (1.0 x 1 x 128⁄8 = 16.0 GBps)
11) HBM_1000_4H_1x64 : (1.0 x 1 x 64⁄8 = 8.0 GBps)

Variate the main memory type between DDR3_1600_x64, DDR3_2133_x64 (DDR3 with a faster clock), LPDDR2_S4_1066_x32 (low-power DRAM often found in mobile devices), and HBM_1000_4H_1x64 for two types of CPU (AtomicSimpleCPU, TimingSimpleCPU). The clock frequency is fixed at 2 GHz. Report (plot) the simulation times and clock numbers of the execution program for 8 (4*2) configurations.

Which CPU model is more sensitive to changing the CPU frequency? Why do you think this is?
Which CPU model is more sensitive to the memory technology? Why?
Is the sieve application more sensitive to the CPU frequency or the memory technology? Why?
If you were to use a different application, do you think your conclusions would change? Why?

Step 2:

Now recompile it with optimization enabled:

```
'g++ -O3 -o program2 program.cc '
```

Do previous assignments again and report differences, and what has caused these differences?