# CS/ECE 752 (Fall'24) HW 1


- Dharaneedaran Kalyanam Sendhil ( Campus ID : 9087068475 )


Following are Screenshots while following the Step 2 of the assignment as proof of completion.

Without adding cache

```
dharaneedaran@Dharaneedaran:~/gem5$ build/X86/gem5.opt configs/tutorial/part1/simple.py
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 10 2024 08:56:21
gem5 started Sep 10 2024 22:35:36
gem5 executing on Dharaneedaran, pid 29061
command line: build/X86/gem5.opt configs/tutorial/part1/simple.py

Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
src/mem/dram_interface.cc:690: warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
src/base/statistics.hh:279: warn: One of the stats is a legacy stat. Legacy stat is a stat that does not belong to any statistics::
Group. Legacy stat is deprecated.
system.remote_gdb: Listening for connections on port 7000
Beginning simulation!
src/sim/simulate.cc:199: info: Entering event queue @ 0.  Starting simulation...
Hello world!
Exiting @ tick 454646000 because exiting with last active thread context
dharaneedaran@Dharaneedaran:~/gem5$
```

After two levels of cache

```
dharaneedaran@Dharaneedaran:~/gem5$ build/X86/gem5.opt configs/tutorial/part1/two_level.py --l2_size='1MB' --l1d_size='128kB
'
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 10 2024 08:56:21
gem5 started Sep 11 2024 22:27:27
gem5 executing on Dharaneedaran, pid 65243
command line: build/X86/gem5.opt configs/tutorial/part1/two_level.py --l2_size=1MB --l1d_size=128kB

Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
src/mem/dram_interface.cc:690: warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbyte
s)
src/base/statistics.hh:279: warn: One of the stats is a legacy stat. Legacy stat is a stat that does not belong to any stati
stics::Group. Legacy stat is deprecated.
system.remote_gdb: Listening for connections on port 7000
Beginning simulation!
src/sim/simulate.cc:199: info: Entering event queue @ 0.  Starting simulation...
Hello world!
Exiting @ tick 57562000 because exiting with last active thread context
dharaneedaran@Dharaneedaran:~/gem5$
```

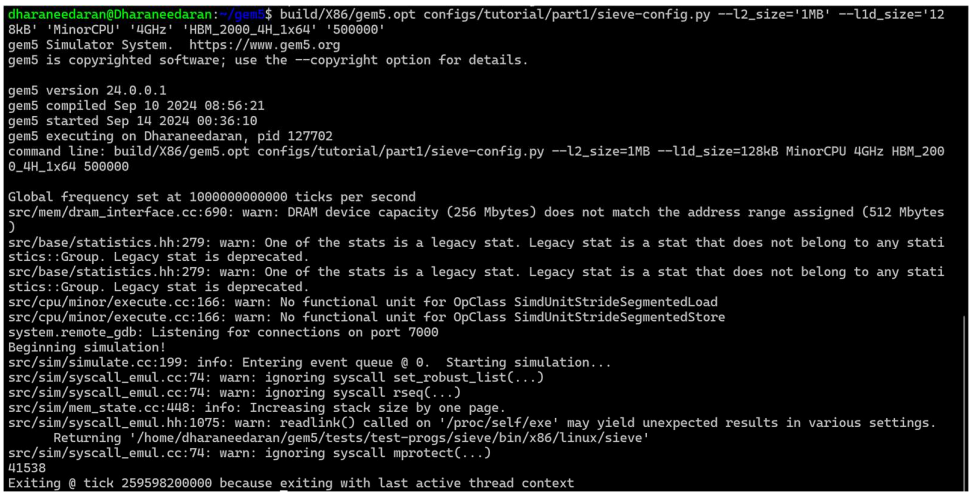
**SIMULATIONS :**

The simulations were executed sequentially using a bash script, which stored each simulation's statistics in a separate directory. The bash script is named hw1_script.sh and can be found in the zipped folder.

Below are the different simulation cases along with their corresponding stats.txt file names. All stats.txt files are located in the folder named stats_folder.

| CPU Model | CPU Frequency (GHz) | Memory | stats.txt filename |
|---|---|---|---|
| TimingSimpleCPU | 1 | DDR3_1600_8x8 | stats_01.txt |
| TimingSimpleCPU | 2 | DDR3_1600_8x8 | stats_02.txt |
| TimingSimpleCPU | 4 | DDR3_1600_8x8 | stats_03.txt |
| MinorCPU | 1 | DDR3_1600_8x8 | stats_04.txt |
| MinorCPU | 2 | DDR3_1600_8x8 | stats_05.txt |
| MinorCPU | 4 | DDR3_1600_8x8 | stats_06.txt |
| TimingSimpleCPU | 4 | DDR3_2133_8x8 | stats_07.txt |
| TimingSimpleCPU | 4 | LPDDR2_S4_1066_1x32 | stats_08.txt |
| TimingSimpleCPU | 4 | HBM_1000_4H_1x64 | stats_09.txt |
| TimingSimpleCPU | 4 | HBM_2000_4H_1x64 | stats_10.txt |
| MinorCPU | 4 | DDR3_2133_8x8 | stats_11.txt |
| MinorCPU | 4 | LPDDR2_S4_1066_1x32 | stats_12.txt |
| MinorCPU | 4 | HBM_1000_4H_1x64 | stats_13.txt |
| MinorCPU | 4 | HBM_2000_4H_1x64 | stats_14.txt |

**The following is an example screenshot for a simulation**




**NOTE:**

For n = 1000000, my simulations were taking me approximately 20 minutes for TimingSimpleCPU and around 60-70 minutes for Minor CPU. Hence, I reduced the value of n to 500000 (half a million). The simulations were all run on my local PC, and hence could have been the reason.

**OBSERVATIONS**

| CPU Model | CPU Frequency (GHz) | Memory | CPI | simTicks |
|---|---|---|---|---|
| TimingSimpleCPU | 1 | DDR3_1600_8x8 | 3.778371 | 1545140637000 |
| TimingSimpleCPU | 2 | DDR3_1600_8x8 | 3.778643 | 772625958500 |
| TimingSimpleCPU | 4 | DDR3_1600_8x8 | 3.779185 | 386368459750 |
| MinorCPU | 1 | DDR3_1600_8x8 | 2.53795 | 1037878376000 |
| MinorCPU | 2 | DDR3_1600_8x8 | 2.538231 | 518996741000 |
| MinorCPU | 4 | DDR3_1600_8x8 | 2.538799 | 259556407500 |
| TimingSimpleCPU | 4 | DDR3_2133_8x8 | 3.779566 | 386362596250 |
| TimingSimpleCPU | 4 | LPDDR2_S4_1066_1 x32 | 3.779566 | 386407384250 |
| TimingSimpleCPU | 4 | HBM_1000_4H_1x64 | 3.779423 | 386392731250 |
| TimingSimpleCPU | 4 | HBM_2000_4H_1x64 | 3.77956 | 386406703750 |
| MinorCPU | 4 | DDR3_2133_8x8 | 2.538744 | 259550759750 |
| MinorCPU | 4 | LPDDR2_S4_1066_1 x32 | 2.539186 | 259595985500 |
| MinorCPU | 4 | HBM_1000_4H_1x64 | 2.539041 | 259581127000 |
| MinorCPU | 4 | HBM_2000_4H_1x64 | 2.539208 | 259598200000 |

1. What metric should you use to compare the performance between different system configurations? Why is this the appropriate metric?

   Ans. The best metric to use for evaluating the performance of various system configurations is execution time (or its inverse, throughput), as it captures the exact amount of time needed to finish a given operation and makes comparisons between settings easy. In gem5, the execution time is the metric named as “simTicks” or “simSeconds” (where, simSeconds = simTicks/simFreq). It would seem better to take simTicks as the metric since more precise.

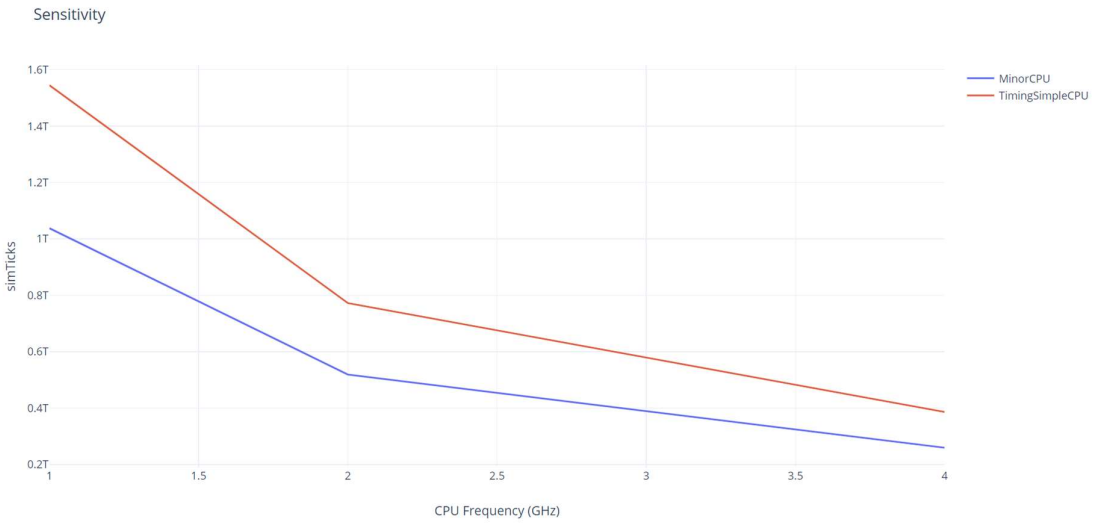
   Although the Iron Law is used a measurement of CPU performance, considering the CPI (Which varies with different implementations of the same ISA) and clock frequency, it does not consider the change of memory technology and size. The hit rates of memory and its technology could play into the performance of the system configuration.

2. Which CPU model is more sensitive to changing the CPU frequency? Why?

   Ans. The basic formula for sensitivity=$\partial x/\partial f(x)$, in which a function f(x) representing the outcome and x is the factor of interest. This can also be interpreted as the slope

that is obtained by plotting x vs f(x). Here f(x) is the CPU frequency and x is the CPU is execution time.

Plotting a graph using the first 6 test cases :




We can see that although both follow almost similar slope, the TimingSimpleCPU can be said to be slighty more sensitive. Although that is nearly negligible.

Since this performance comparison is between two different CPUs, with the same memory technology, we can compare it using,

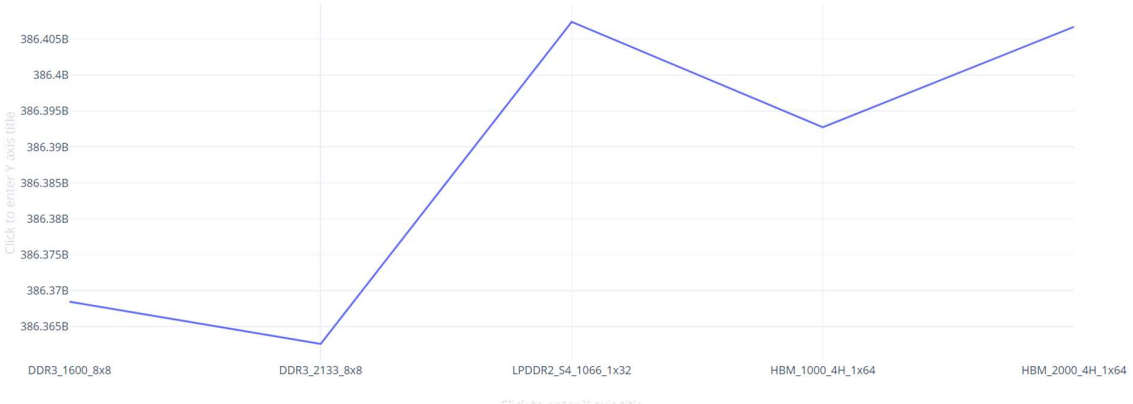
$$\text{Time for program} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}$$

Since the intstructions simulated per program is very close (since the ISA that both the CPUs are based on and the compiler we are running it on are the same) and the same for a specific CPU type across clock frequency, CPI is the same for all frequencies of that given a CPU model, the only difference in time for program differs in time/cycle = 1/clk frequency. Hence, the sensitivity for both the CPU models to clock frequency is exactly the same.

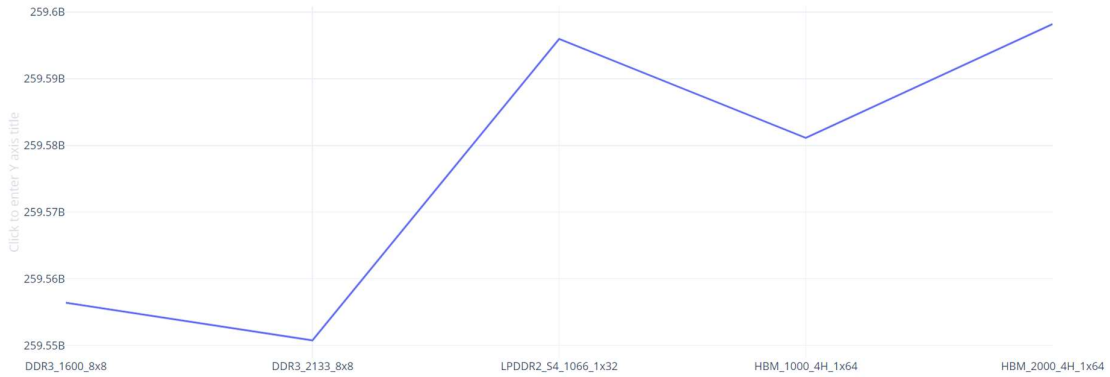
3. Which CPU model is more sensitive to changing the memory technology? Why?

   Ans. Plotting the change in execution time of both the CPUs with varying memory, we get:

   1. TimingSimpleCPU




2. MinorCPU




The graph, although quite closely related, we can deduct which CPU is more affected in its performance by checking for the percentage change in difference in execution time when we shift from one memory to another. Listed Below:

| | TimingSimpleCPU | MinorCPU |
|---|---|---|
| DDR3_1600_8x8 - DDR3_2133_8x8 | -1.51759E-05 | -2.17592E-05 |
| DDR3_2133_8x8 - LPDDR2_S4_1066_1x32 | 0.000115922 | 0.000174246 |
| LPDDR2_S4_1066_1x32 - HBM_1000_4H_1x64 | -3.79211E-05 | -5.7237E-05 |
| HBM_1000_4H_1x64 - HBM_2000_4H_1x64 | 3.61614E-05 | 6.57713E-05 |
| Greatest % Difference | 0.000115922 | 0.000182778 |

As we can see, the Minor CPU seems to have more fluctuations in its execution time due to changes in memory technology. Hence, we could say MinorCPU is more sensitive.

4. Is your sieve application more sensitive to the CPU model, the memory technology, or CPU frequency? Why?

Ans. Comparing the execution time of all the test cases, it can be seen by comparison that although the sieve application is sensitive to CPU model, the memory technology, and CPU frequency, CPU frequency has the most effect on its execution time.

We can deduce in both TimingSimpleCPU and MinorCPU case that, speed of execution of the application is directly proportional to the clock speed (ie, time/cycle) and hence has the highest effect. Although the execution on MinorCPU seems to be faster, the increase or step up from TimingSimpleCPU is not as significant as doubling the clock frequency is. The memory technology change seems to be giving only a minute change in execution time.

5. If you were to use a different application, do you think your conclusions would change? Why?

Ans. No, the answers should not deviate at all for cases 1-3 and for 4, the reason it would change is only if the function is so memory intensive that the memory becomes a large bottleneck, and the efficiency of the memory outweighs the processors speed.