Assignment 1: gem5 Tutorial

Arpit Prasad
COL718 - Architecture for High Performance Computers

August 16, 2025

1 Overview

In this assignment we perform sweep analysis of systems by varying the type of CPU, CPU frequency and Memory Configuration. For this assignment we have used the following type for each of the mentioned variables:

1. CPU Types:
   (a) TIMING: In Order Execution
   (b) O3: Out of Order Exection
2. CPU Frequency: A sweep from 0.6GHz to 3.3GHz in steps of 0.2GHz
3. Memory Configuations:
   (a) Single Channel DDR3 1600MHz
   (b) Single Channel DDR4 2400MHz
   (c) High Bandwidth Memory

Following are the plots and variation of performances obtained in the sweep.

2 Matrix Multiplication

Reference for matrix Multiplication was taken from the following https://www.akkadia.org/drepper/cpumem

Here we perform block wise multiplication. This allows an entire block to be loaded onto the cache at once, and hence lesser miss rate.

The binary produced from this code was done with following command

```
g++ -O3 mm.c -o mm
```

Listing 1: Shell Script used to obtain binary

for agressive optimisations.

The size of the matrices was assumed to be 100x100

3 Performance Metrics

The following is the overview of the performance metrics that are used to analyse the performance of a code with respect to a Model

Note, the performance metrics are evaluated on SE mode (System Exection Mode)

1. IPC: Instructions Per Cycle, indicating the throughput of the system
2. Simulation Seconds: Amount of time in seconds to simulate a binary on the simulated device

4 Plots

4.1 IPC vs Frequency

First we keep Memory Configuation fixed and vary CPU Types and then CPU Type fixed and vary Memory Configuation

4.1.1 CPU Variation

From Fig. 1 the IPC remains nearly constant for both the types of CPU. This is because IPC is the measure of instructions per cycle. Therefore, as frequency increases the number of instruction per second increases and and the same time number of cycles per second. Therefore their division remains constant.
We can also see O3 being out of order has higher IPC than TIMING which has in order exection

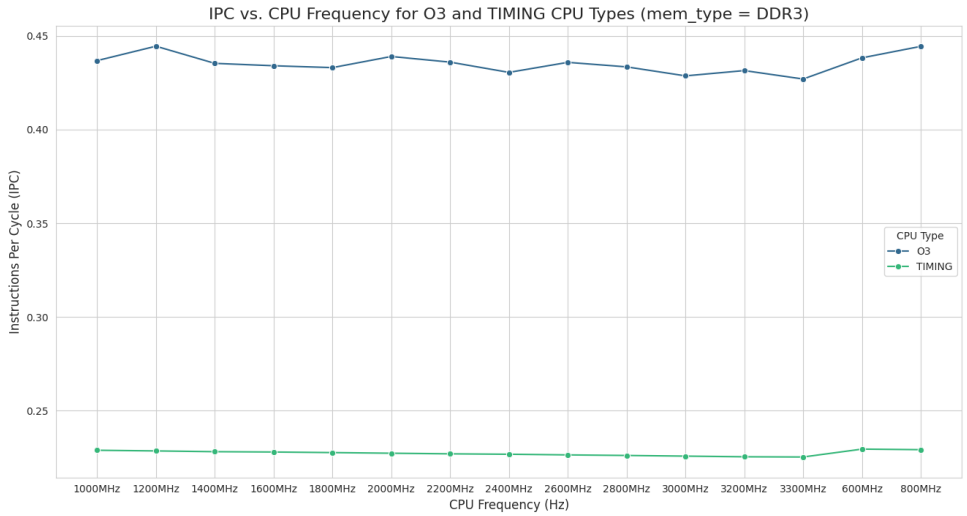

Figure 1: Variation of IPC when changing the frequency of the chipset, given a particular Memory Configuation

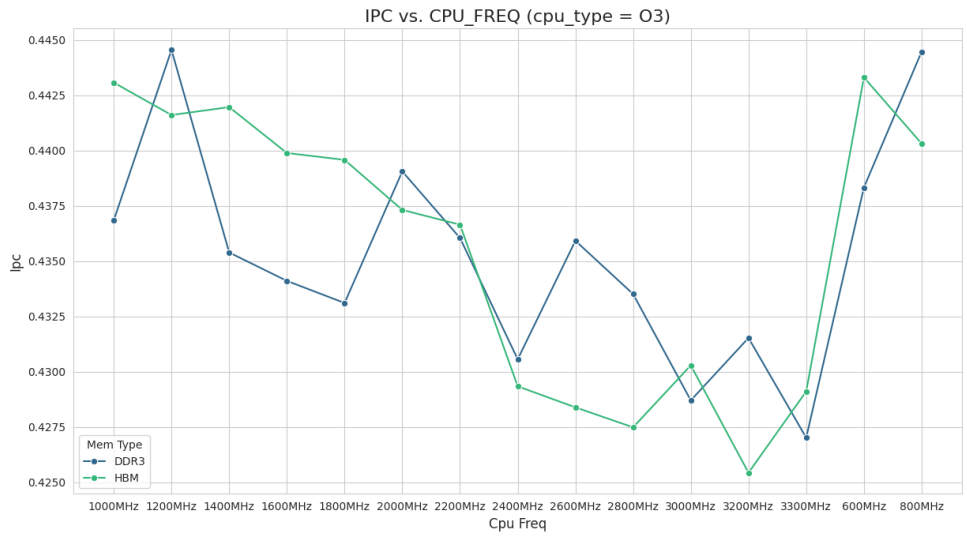

Figure 2: Variation of IPC when changing frequency of the chipset, given a pparticular CPU Type

4.1.2 Memory Configuation Variation

From Fig. 2 we again notice the value of IPC almost remains constant. (Note the scale of the graph makes it appear visually to vary highly.) Similar arguments as mentioned above apply here. HBM should technically be faster than DDR3, however this cannot be reflected from the IPC vs Frequency graph due to seconds cancelling on the division of Instructions/sec and Cycles/sec

4.2 Simulation Seconds vs Frequency

Again, we follow the same pattern, i.e., first we keep Memory Configuation fixed and vary CPU Types and then CPU Type fixed and vary Memory Configuation.

4.2.1 CPU Variation

From Fig. 3, the variation we observe a downward trend in the graph. Since as the frequency increases the amount of time taken for simulation decreases, since the number of instructions exected per second increases

Again CPU Type O3 has smaller simulation time for a given frequency since it operates on out of order execution. Whereas, TIMING operates in order execution

4.2.2 Memory Configuration Variation

From Fig. 4 we again observe a downward sloping curve. However, here we observe the exact similar values of the different types of memory configurations. For different types of memory units we obsever similar values of simulation times given a frequency because as sweep through the frequency, either the max frequency of transfer of data from memory unit

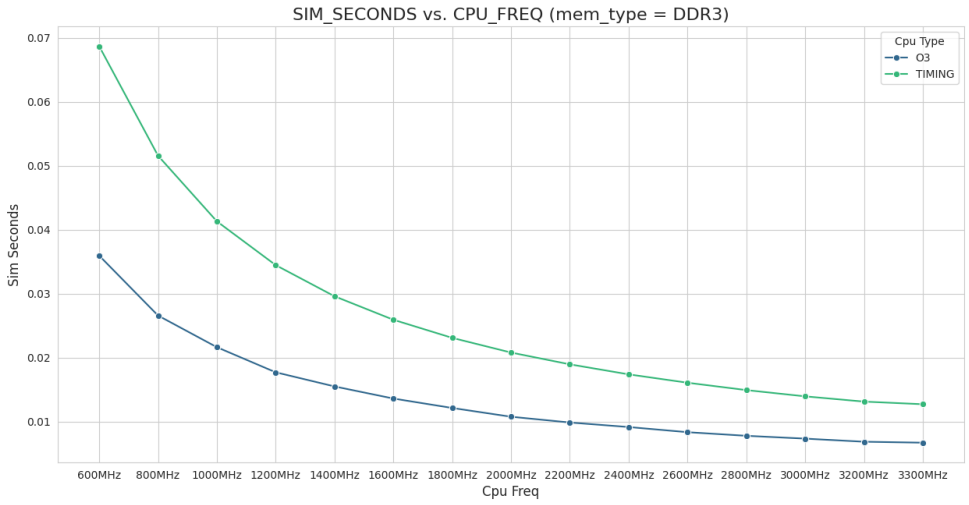

Figure 3: Simulation Seconds vs Frequency, given a Memory Configuation

limits the cpu frequency when it is greater than memory's frequency or vice versa. Therfore at frequency lower than memory frequency, data transfers occur at cpu frequency. At higher cpu frequency, the data transfer is limited by the max frequency of memory

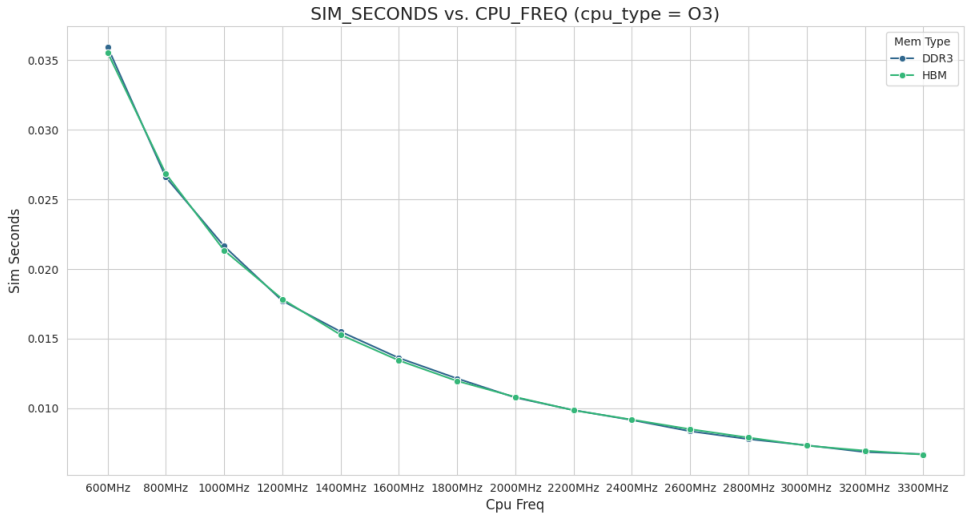

Figure 4: Simulation Seconds vs Frequency, given a CPU Type

4.3 Latency

We consider the average read and write Latencies of the systems, using a memory of Single Channel DDR3

From Table 1, clearly both the read and the write latencies are better for O3, which is executes instructions out of order.

Table 1: Variation of Latencies with CPU Types

CPU Type	Order of Exection	Frequency	Read Latency (ticks / count)	Write Latency (ticks / count)
O3	Out of Order	600MHz	27663.28	1020144433.51
TIMING	In Order	600MHz	29226.18	2080891024.17
O3	Out of Order	800MHz	28242.87	741798455.68
TIMING	In Order	800MHz	28712.47	1489806432.79

Latencies do not show variation with increase in frequency since they are measured in ticks per count. As frequency increases the number of ticks per second and count per second both increase and hence their division remains nearly constant.