Analysis of Sieve of Eratosthenes using gem5 Simulator

The gem5 simulator tool is used to simulate a ARM 64 bit Architecture for measuring the changes in sim_seconds metric over the ARM-64 binary workload of Sieve of Eratosthenes algorithm with input of 1000000.

Configurations used:

CPU: SimpleTimingCPU and MINOR

cache_hierarchy:

1. No Cache
2. 2-Level Cache:

- -32 KiB direct mapped L1 instruction cache, 64 KiB direct mapped L1 data cache, and 4 MiB 8-way L2 unified cache.
- L1 cache (64K L1 i-cache, 128K L1 d-cache) using
  the same L2 cache.

Memory:

1. DDR3_1600_8x8
2. DDR3_2133_8x8
3. LPDDR3_S4_1066_1x32

Clock Cycles:

1 GHz to 3GHz (incrementing 500 MHz)

The runs where the config was without any cache, ran on the CPUs SimpleTimingCPU and MINOR with clock cycle 1GHz marked the metric simSeconds as 3.724730 and 0.818893 in seconds respectively. With cache_hierarchy as 32 KiB direct mapped L1 instruction cache, 64 KiB direct mapped L1 data cache, and 4 MiB 8-way L2 unified cache the simSeconds as 0.087649 and 0.055429. The config with caches took a lot less time to execute than the no cache model.

As the clock cycle increased from 1GHz to 3 GHz (with 500MHz increments), over the above config with caches the simSeconds metric has reduced significantly.

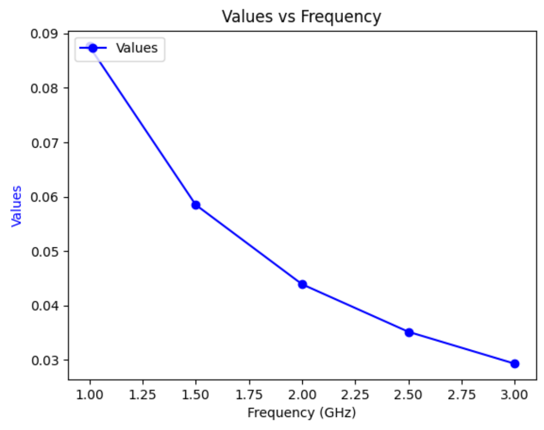


Fig 1.0, SimpleTiming CPU

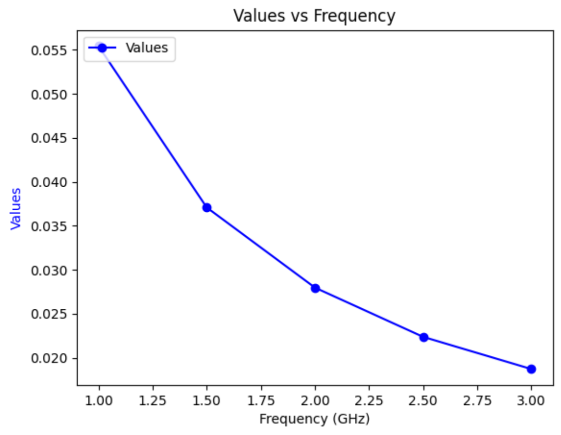


Fig 1.1, MINOR

Fig 1.0 and Fig1.1 shows a significant decrease in the metric sim_seconds with increase on clock cycle.

With caches changed:

The caches were changed to 2-way L1 cache with size 32KiB(i_cache), 64 KiB(d_cache) and 4 MiB 8-way L2 unified cache and the simulation was run on SimpleTimingCPU and MINOR varying the clock cycles as specified in the previous step. The graphs 2.0 and 2.1 shows the decrease in the simSeconds.

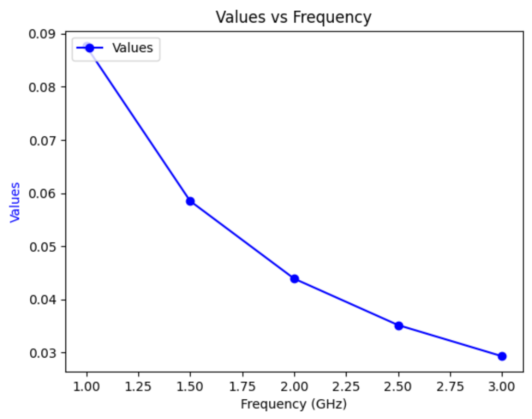


Fig 2.0 SimpleTimingCPU

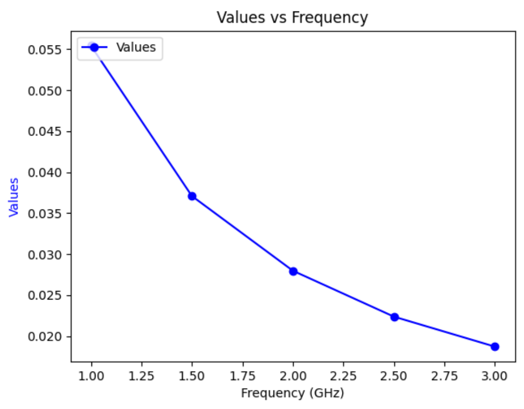
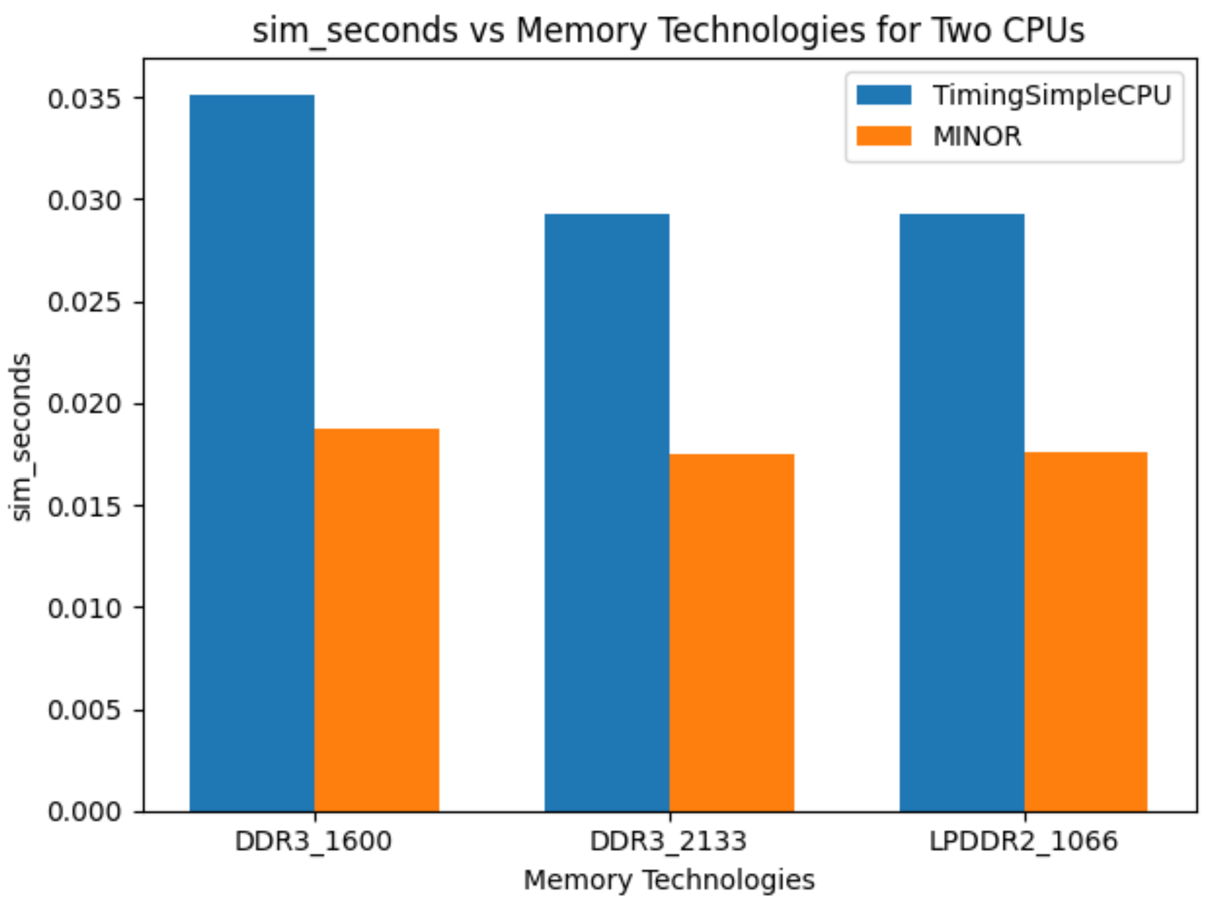


Fig 2.1 MINOR

The remaining runs with the comparison as above were of same nature.

The minor CPU model exhibited the highest percentage of change with respect to change in the frequency of the CPU.

The memory technologies implemented are the DDR#_1600_8x8, DDR#_2133_8x8 and LPDDR2_1066_1x32, we take all the CPUs operating at highest frequency



and with 2Level caches, this acts as a baseline for evaluation, and we will cycle through the memory technologies.

From the above graph, we can conclude that TimingSimpleCPU has the highest change with memory technologies.

The workload sieve has more significant changes in the sim_seconds metric when there is a change in the CPU frequency. I believe this is because, there is more executions of instruction on higher frequencies of the processor.

The effect of caches on performance.

The caches configuaration performance is dependent on the workload and it's implementation. For example the sieve file for the project implements dynamic memory allocation, in this case:

The smaller direct map L1 cache behaves perform worse than the large direct-mapped L1 cache, as it wouldn't be able to hold the allocated memory, a larger cache generally reduces the misses, and ultimately enhance the execution time of the program. A set associative mapped cache will reduce the misses from a direct mapped cache.Since the direct mapped L1 cache is direct mapped and the smaller L1 cache is set associative there wouldn't be any significant difference in sim_seconds, since for larger cache, the data misses would be lower because of it's size and for the smaller cache the cache misses would be lower because of it's associativity.

If we were to use a different application the conclusions would change, the cache hierarchy would act differently based on the design of the code implementation. There could be multi threading that can be incorporated in many languages, the optimization techniques over a given solution would change drastically based on the problem statement.