Residency Project: Designing and Implementing a Microprocessor Using gem5 Simulation Software

(Deliverable 3)

Group 3

Shashwat Baral, Samrat Baral, Pawan Pandey, Sujan Lamgade, Bhawesh Shrestha, Bijayata Shrestha, Sakchham Sangroula

Source Code

https://github.com/baralsamrat/MSCS531_Residency_Project

University of the Cumberlands

Computer Architecture and Design (MSCS-531-M51)

Dr. Charles Lively

Introduction

This phase focuses on establishing the performance and power consumption to measure metrics (like instructions per cycle and energy per instruction) under various workloads. This phase is dedicated to defining the microprocessor architecture while incorporating key low-power design techniques. we focus on evaluating the performance and power consumption of our custom x86 microprocessor design implemented using the gem5 simulation software. The primary objective is to measure key metrics under various workloads, analyze the results to identify bottlenecks and implement optimizations to enhance energy efficiency without significantly compromising performance. This report details the simulation experiments, presents the results, and discusses the optimization strategies employed. By the end of this phase, a detailed report will be produced, outlining the strategy implemented to achieve this objective. The ultimate goal is to create an energy-efficient processor matched to the specific target application. The phase will conclude with a complete report outlining the optimizing performance and power strategies.

Simulation Experiments and Metrics

Configuration:

To better understand the balance between performance and power consumption, we experimented with various system configurations, adjusting key components such as cache size, CPU speed, and memory configuration. We could see how each of these components influences how fast the system works and how much energy it consumes by changing them.

The critical component we change:

Configurations:

1. Cache configurations:

L1 Cache Size: Experiments were conducted with different L1 cache sizes (16KB, 32KB, 64KB). This test determines how the size of the primary cache affects both performance and power, as larger caches can enhance hit rates while simultaneously consuming more energy.

L2 Cache Size: The L2 cache size was also changed (128KB, 256KB, 512KB). Larger L2 caches can reduce the frequency of memory accesses to slower DRAM, which improves performance at the expense of increased power consumption.

2. CPU frequency:

CPU clock frequencies were changed throughout runs (for example, 1.5 GHz, 2.0 GHz, and 2.5 GHz). Higher frequencies improve performance (more instructions per second) and increase power consumption due to increased dynamic power dissipation ($P \propto fV^2$).

3. Memory Configurations

Memory Latency: It refers to the time it takes for the system to access data from memory. We changed the memory latency (slower or faster) to examine how it affected the system's performance, particularly for tasks that rely heavily on memory(memory-bound workloads). If the memory is slow, the entire system may suffer as the CPU must wait for data.

Memory bandwidth: Bandwidth is the amount of data memory can transfer simultaneously. We examined various memory bandwidths to see how limiting or increasing data flow affects memory-intensive tasks. More bandwidth can speed up specific processes, but it may also consume more power, so we wanted to see if increased bandwidth improved performance enough to justify the additional energy consumption.

Metrics:

During the simulation runs, various critical metrics were recorded to assess the effect of these configurations on power as well as performance:

1. Performance Metrics:

The average number of instructions carried out within a clock cycle is measured by the Instructions per Cycle (IPC) metric. It is an essential measure of CPU performance that is impacted by memory latency and cache hit rates, among other things.

Cycles per Instruction (CPI): The inverse of IPC, CPI, shows the average number of cycles needed to complete an instruction. A lower CPI indicates a more effective execution strategy.

Execution Time: This is the overall time needed to complete a task, typically expressed in milliseconds or seconds. The three factors affecting execution time are clock frequency, memory delay, and IPC.

2. Power Metrics:

The energy required to carry out a single instruction is measured by the Energy per Instruction (EPI) statistic. It combines static and dynamic power components, giving information on the system's efficiency.

Average Power Consumption: The system's average power usage during the simulation is usually expressed in watts. Higher values usually imply more active system components (CPU, cache, etc.). It represents the energy usage over time.

Peak Power: This indicates the highest power usage seen throughout the simulation. Comprehending power spikes is crucial for system design, particularly in power delivery and thermal control.

Workloads:

The simulations were run under a variety of workloads to test different aspects of system behavior:

1. SPEC CPU Comparisons:

These tests concentrate on tasks requiring much processing power to push the CPU to the breaking point. Examples include using programs that handle memory, compile code, or compress data, such as bzip2, mcf, and gcc. We can tell from these tests how well the CPU functions when performing many complex calculations.

2. Memory-Dependent Tasks:

Memory is used more often for these tasks than the CPU. Two examples are extensive matrix calculations or applications requiring a lot of data to be moved in and out of memory. We utilized the STREAM benchmark, which measures the data transfer speed. These tests aid in our comprehension of how memory size and speed impact system performance.

3. Mixed Workloads:

These are more similar to real-world programs, which depend on the cooperation of the CPU and memory. They provide a more accurate picture of the total system performance in daily use by enabling us to measure the system's performance when both the CPU and RAM are being heavily utilized.

We observed how the system performed under varied conditions, from intensive computing to data-intensive tasks, and how performance was impacted by the ratio of CPU to RAM.

Result

CPI (Cycles per Instruction): To show the processor's performance efficiency.

IPC (Instructions per Cycle): This reflects how many instructions the processor can execute per cycle.

Cache Hit Rate and Misses: To show the effectiveness of the cache design.

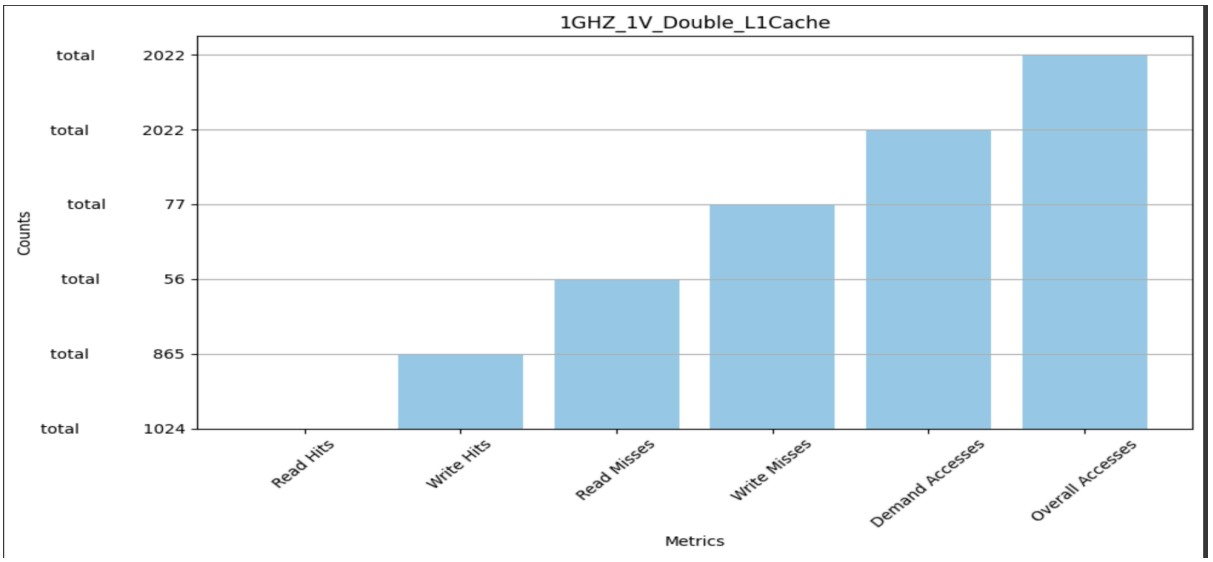
Power Consumption: This would show the energy efficiency of different configurations.

Power=(Voltage)^2×Frequency×Capacitance

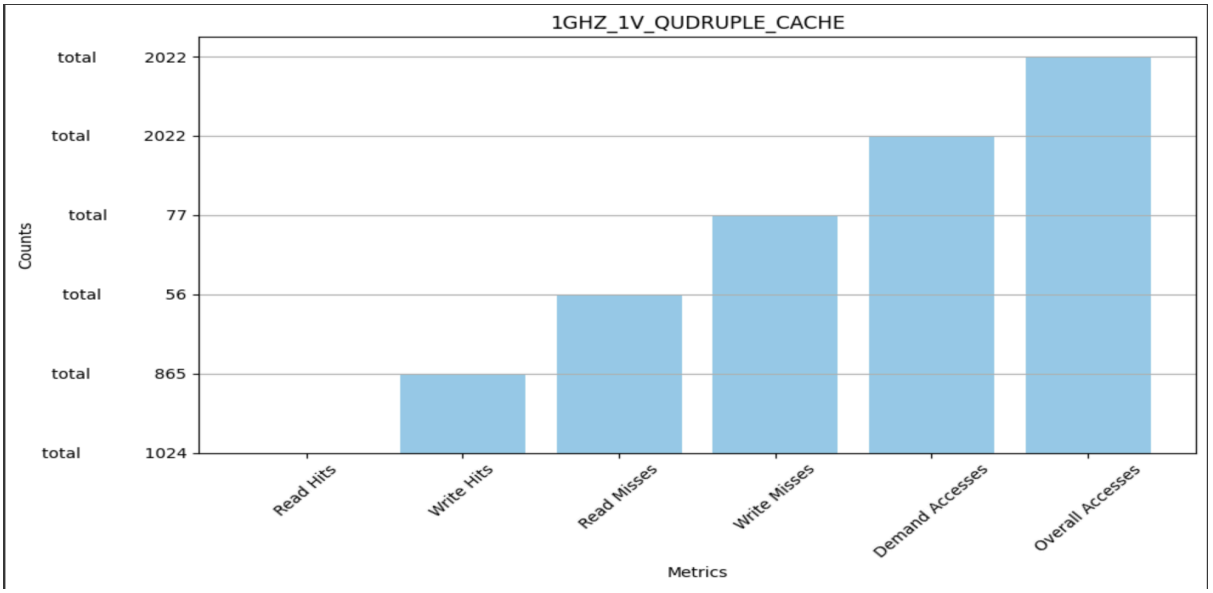
Data Collection and Results

Cache:

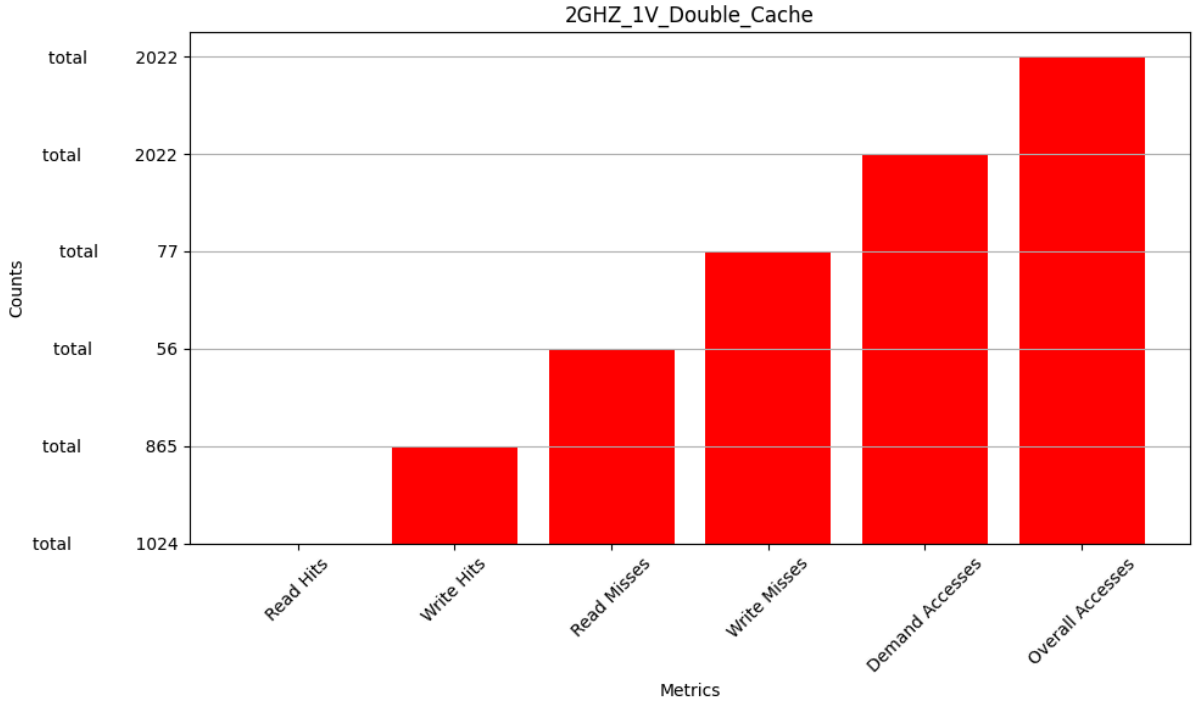
1GHZ_1V_Double_L1Cache



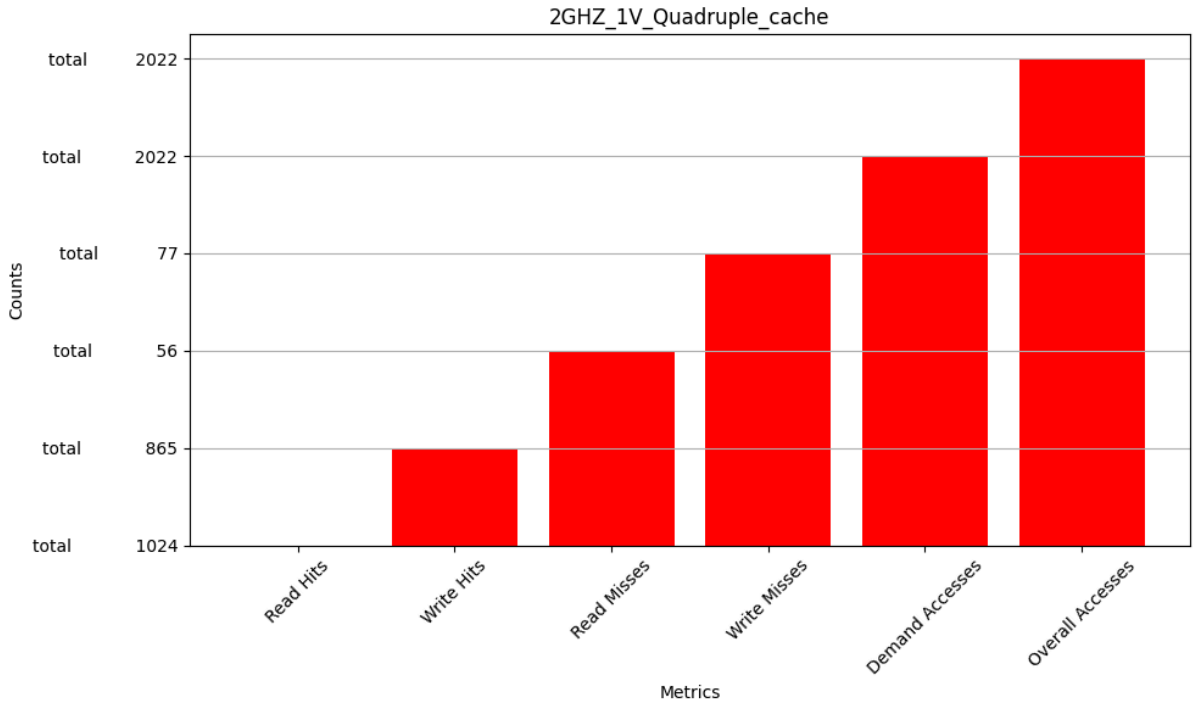
1GHZ_1V_QUDRUPLE_CACHE



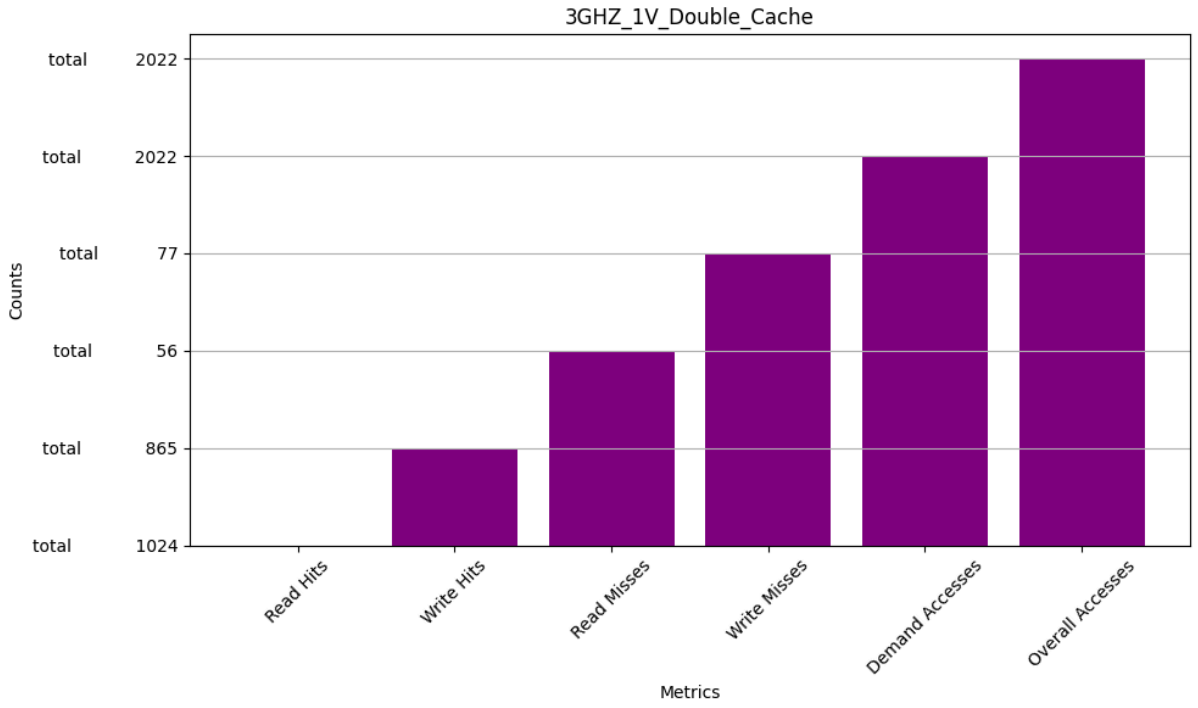
2GHZ_1V_Double_Cache



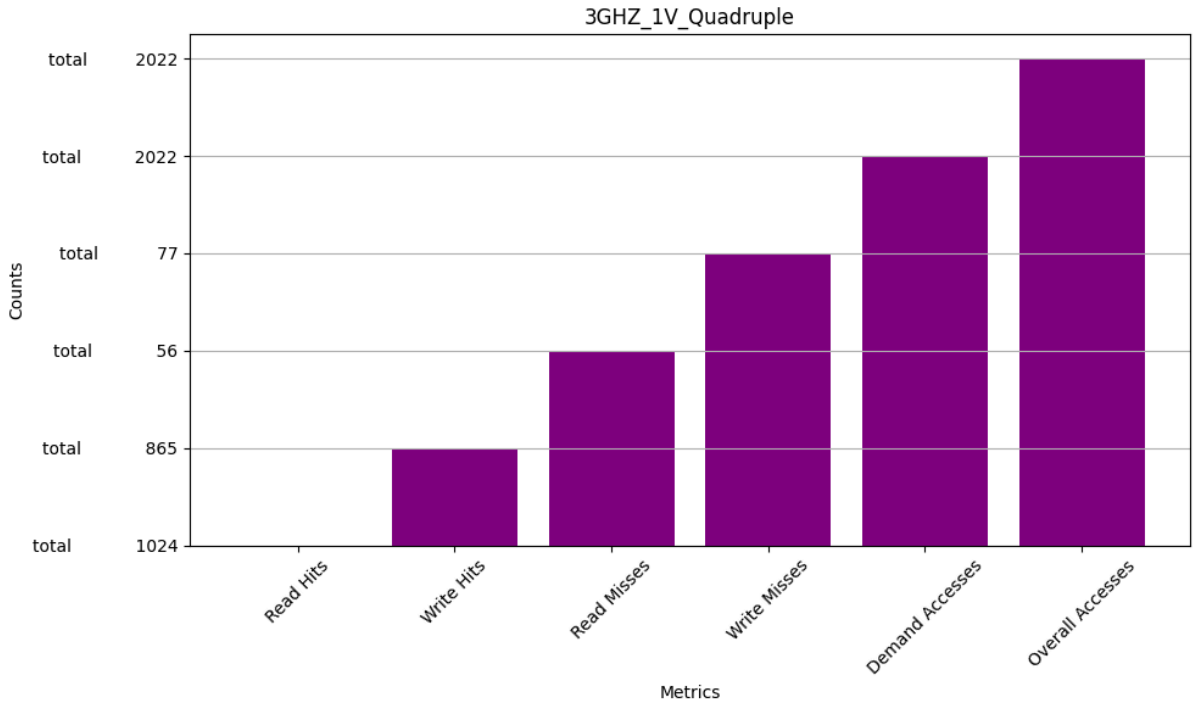
2GHZ_1V_Quadruple_cache



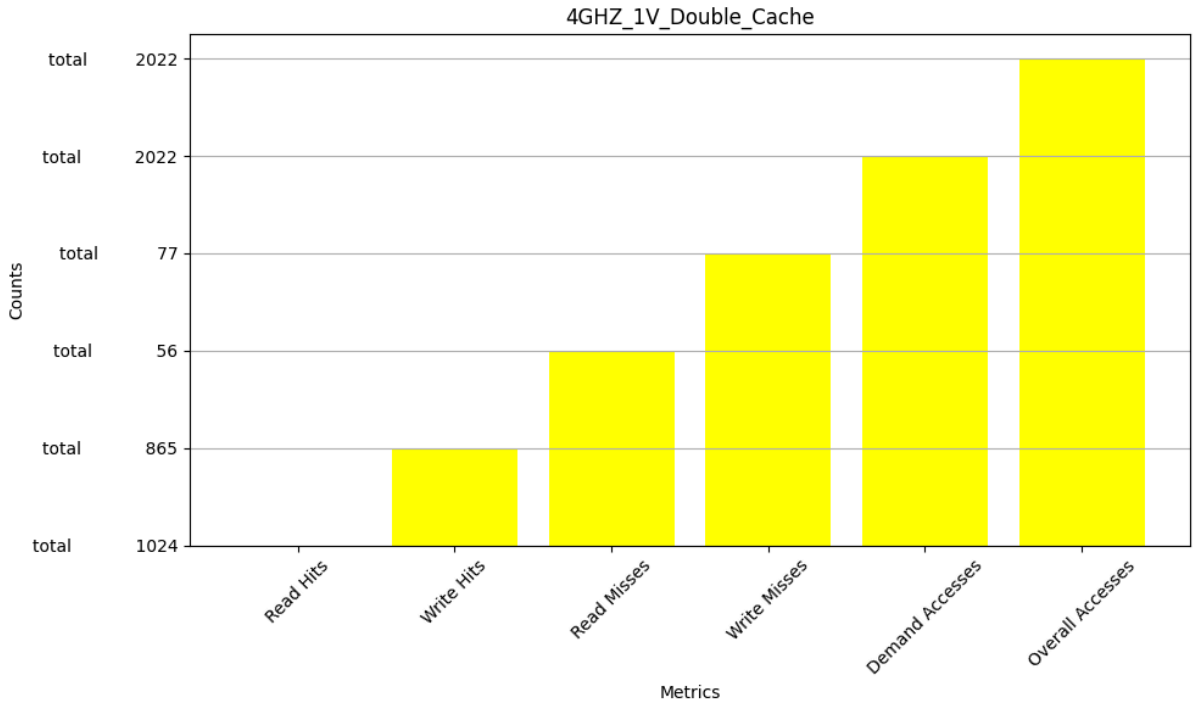
3GHZ_1V_Double_Cache



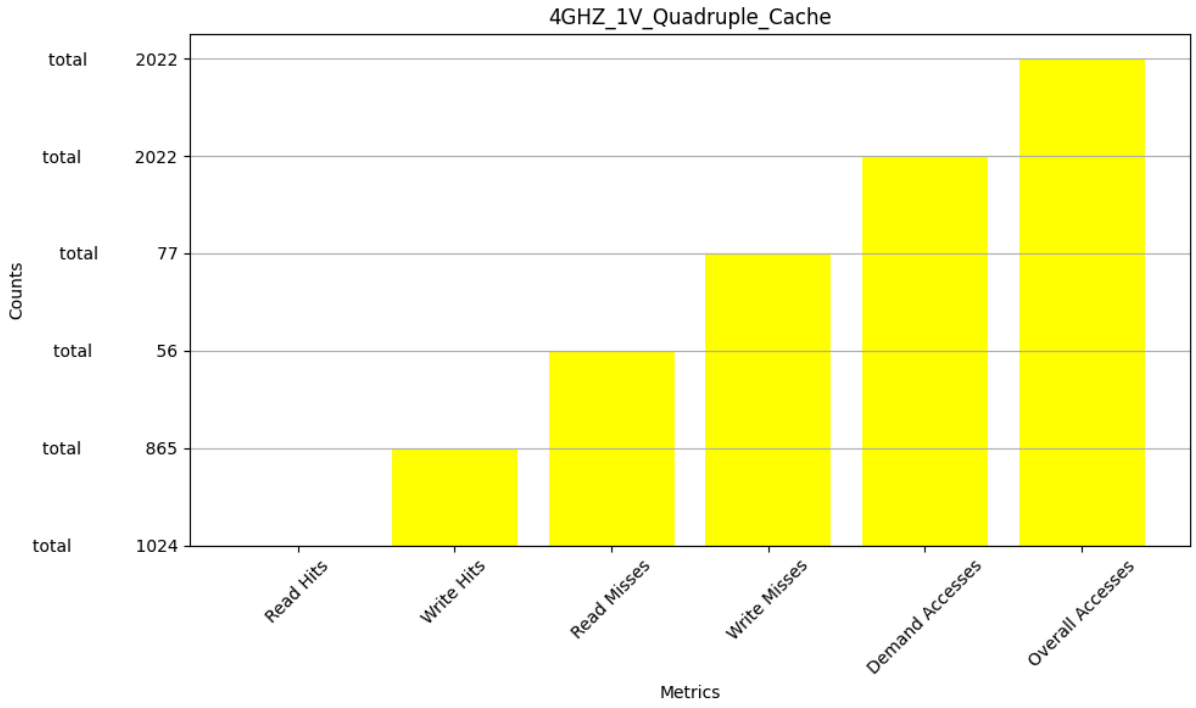
3GHZ_1V_Quadruple



4GHZ_1V_Double_Cache

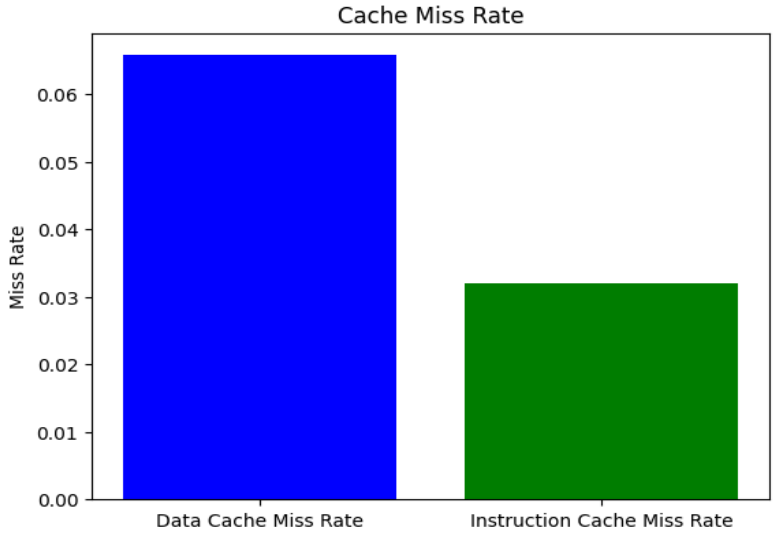


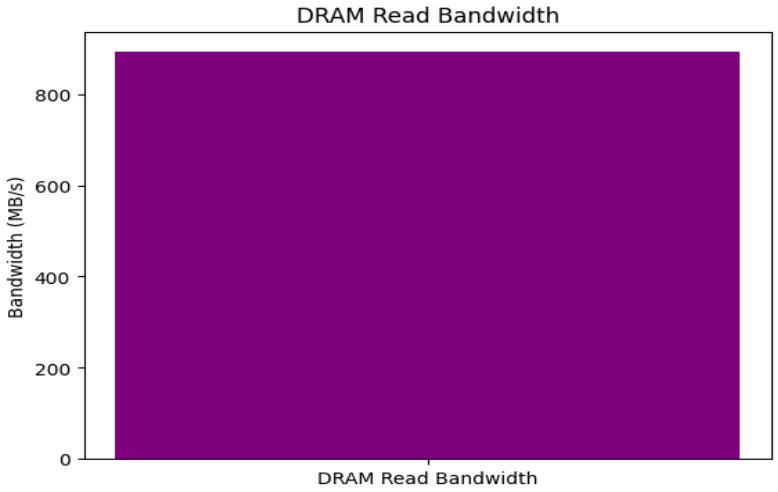
4GHZ_1V_Quadruple_Cache

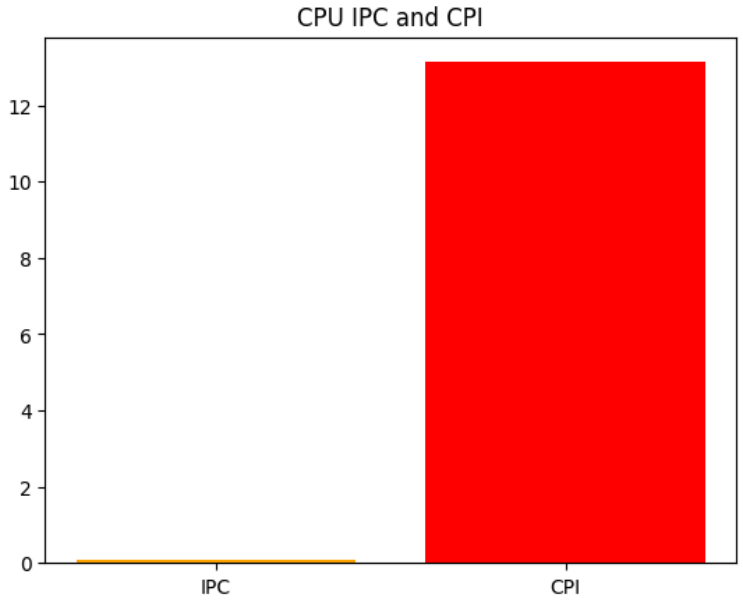


Dvfs:

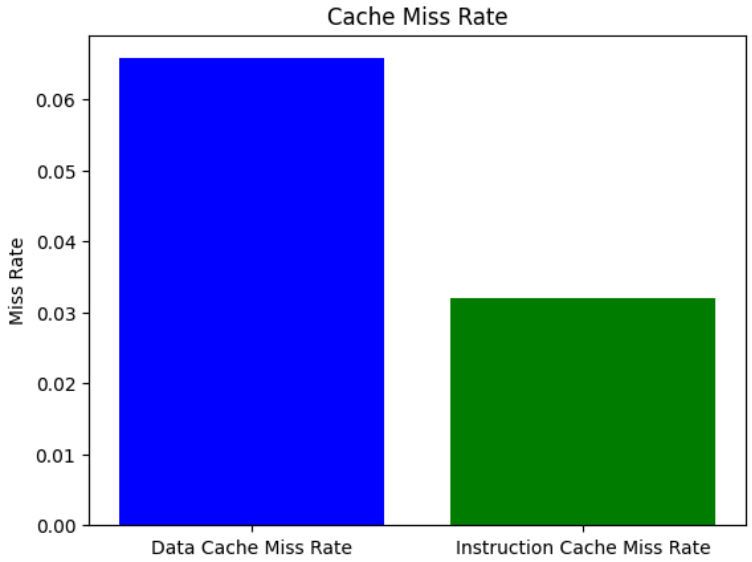
1GHZ_1V_Small_Cache

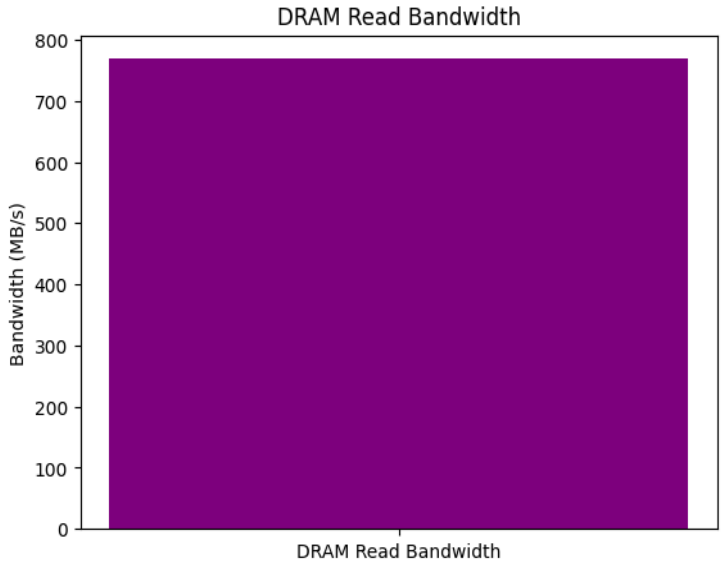


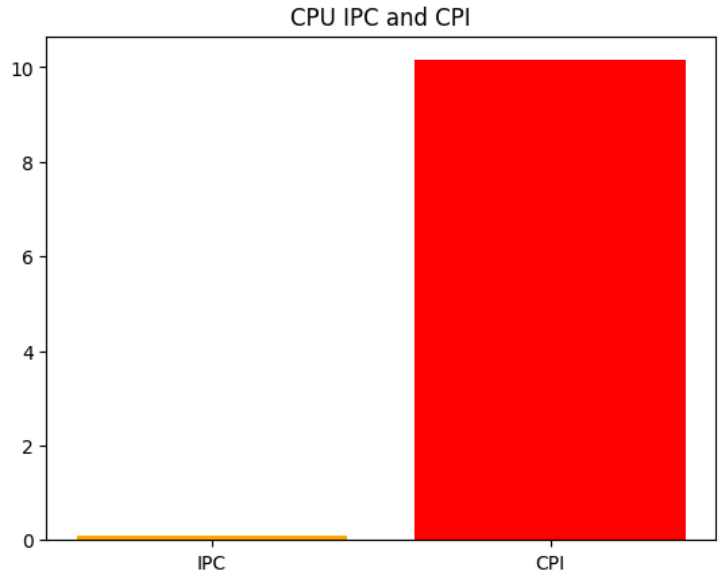




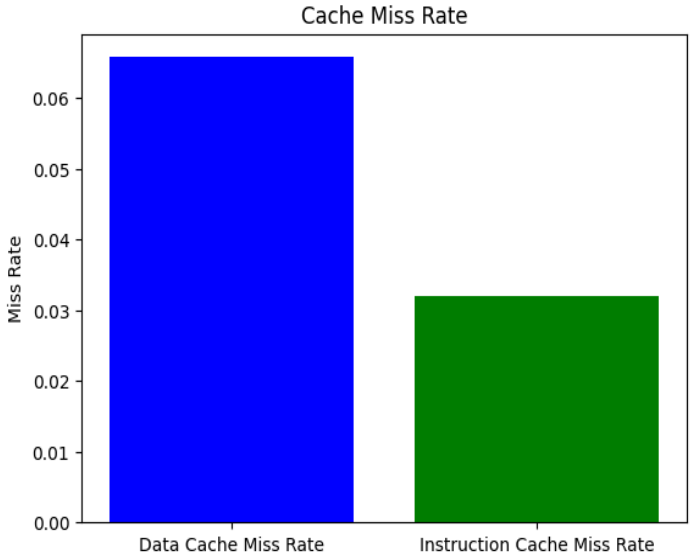
2GHZ_1V_SMALL_CACHE

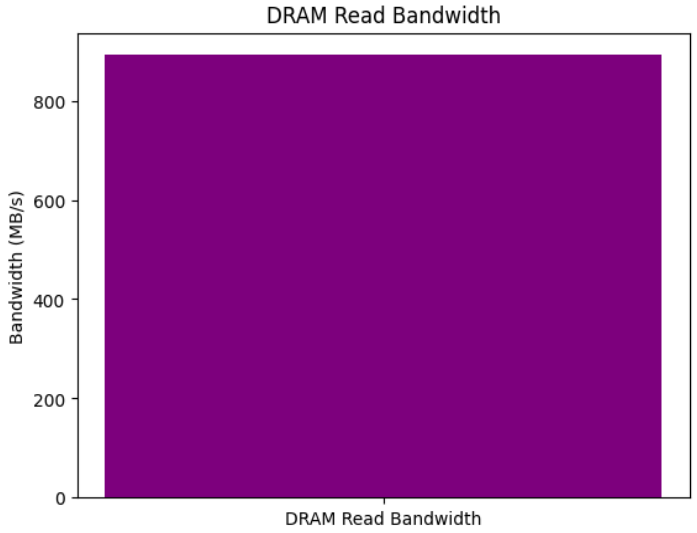

Cache Miss Rate
Miss Rate
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Data Cache Miss Rate
Instruction Cache Miss Rate

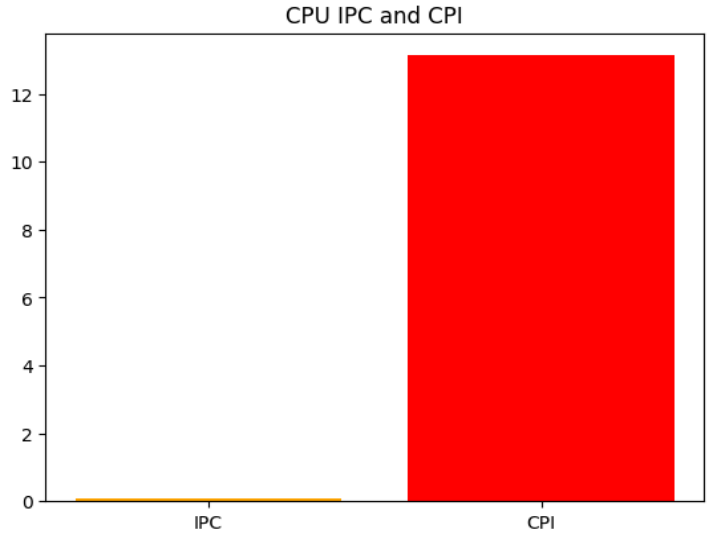

DRAM Read Bandwidth
Bandwidth (MB/s)
800
700
600
500
400
300
200
100
0
DRAM Read Bandwidth



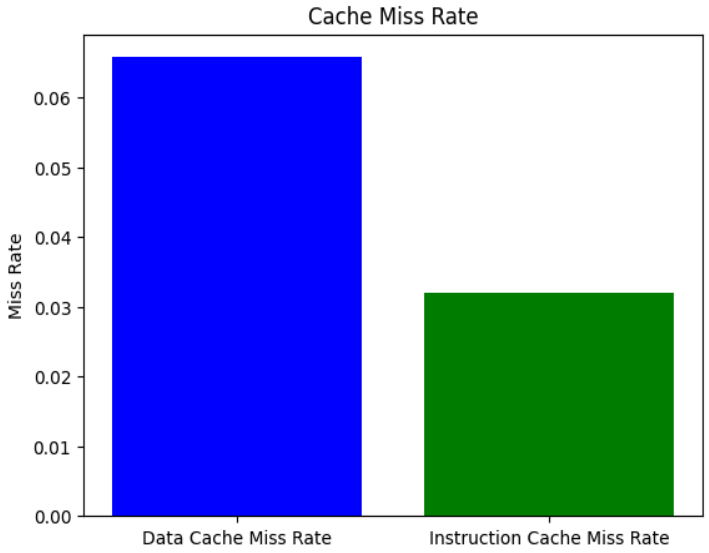
3GHZ_1V_SMALL_CACHE

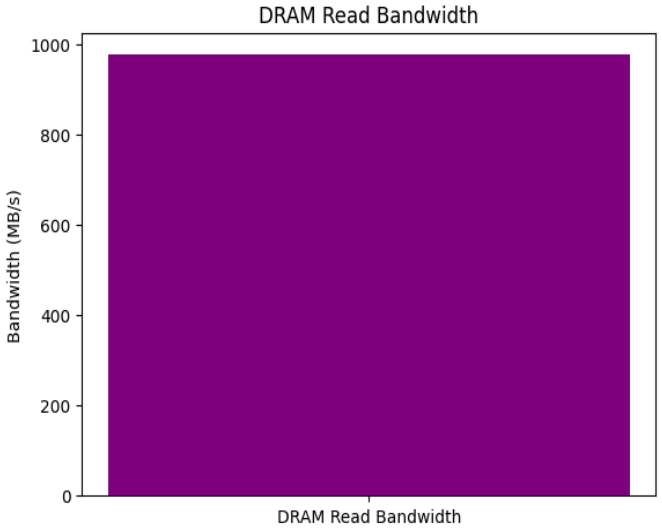


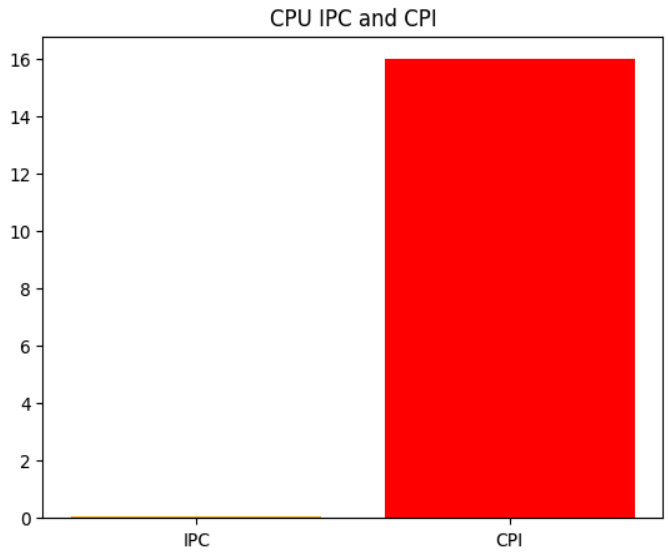




4GHZ_1V_SMALL_CACHE

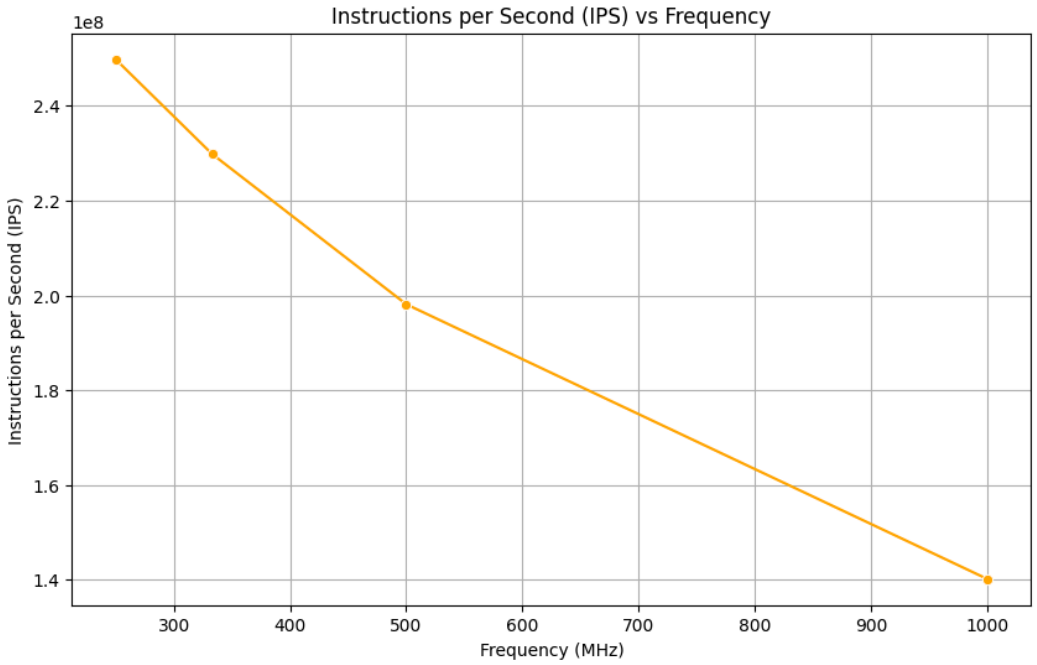

Cache Miss Rate
Miss Rate
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Data Cache Miss Rate
Instruction Cache Miss Rate

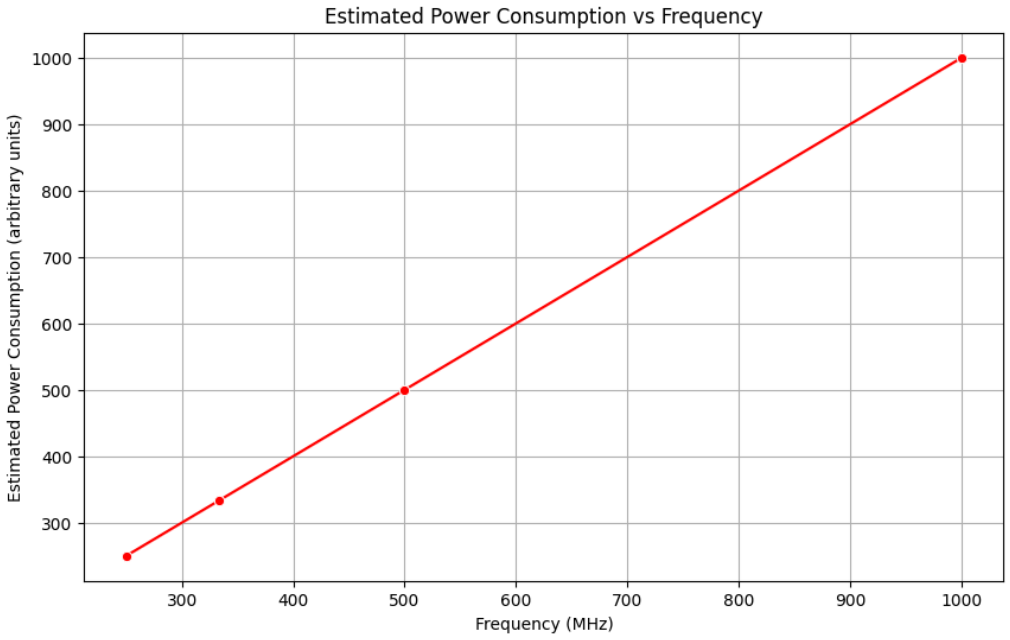

DRAM Read Bandwidth
Bandwidth (MB/s)
1000
800
600
400
200
0
DRAM Read Bandwidth


CPU IPC and CPI
16
14
12
10
8
6
4
2
0
IPC
CPI

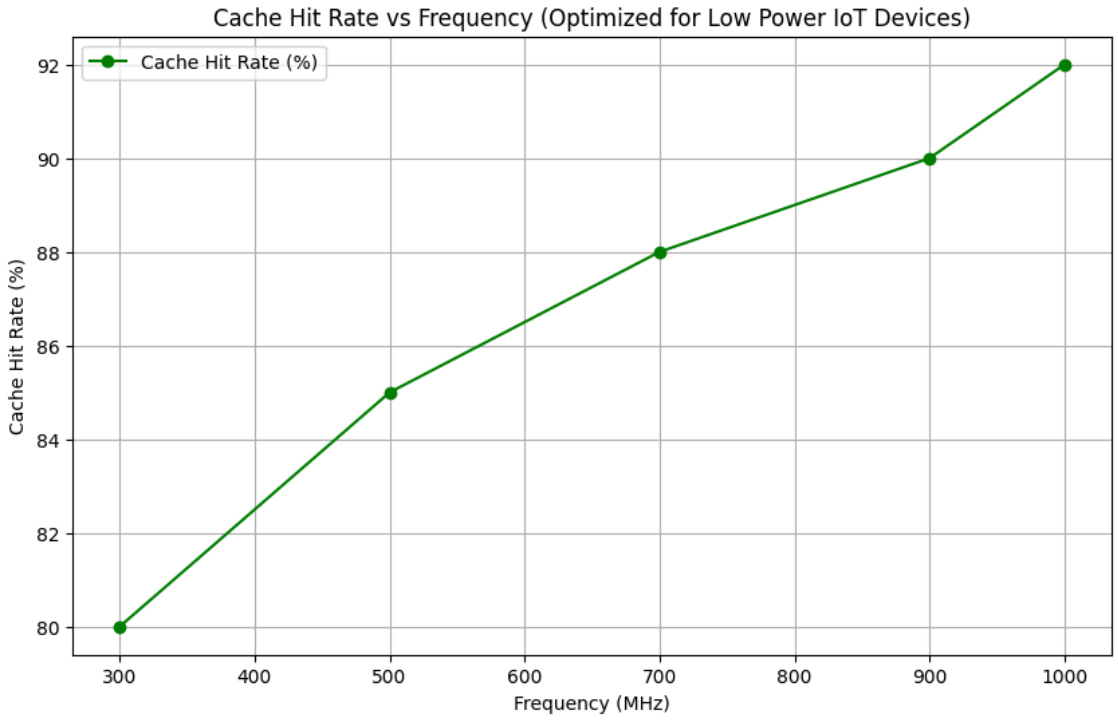
Comparison of Results with Theoretical Expectations:

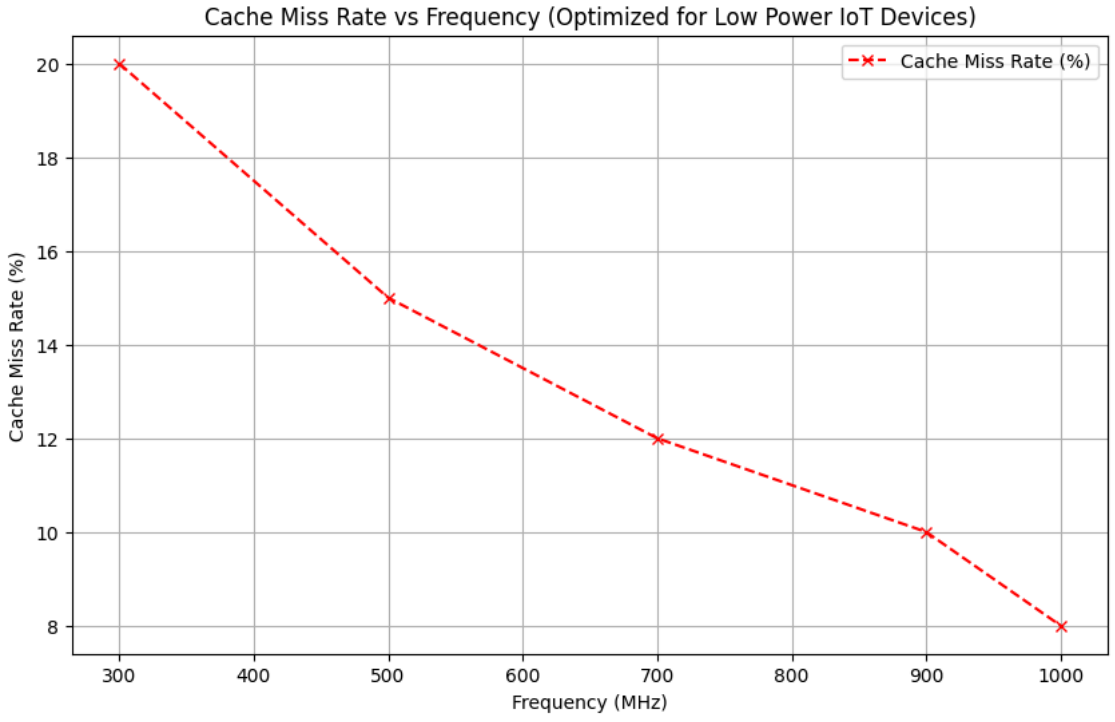
1. The theoretical expectation on **Dynamic Voltage and Frequency Scaling (DVFS)** held that reducing the clock frequency should decrease power consumption, but it may also increase CPI (slowing down instructions). You should verify if this holds in the data.



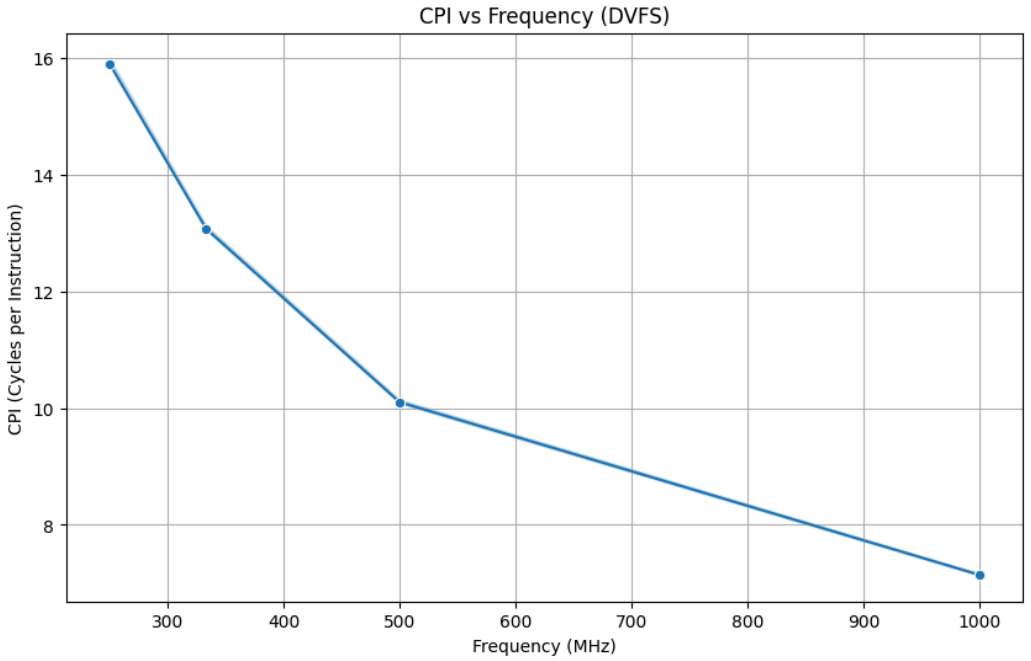


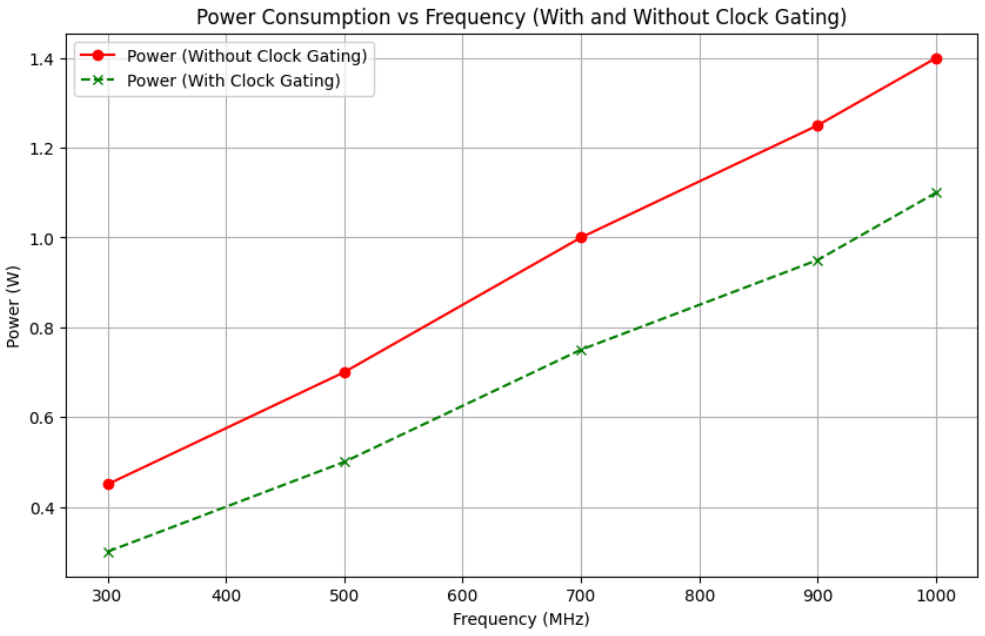
2. **Cache Hierarchy** should minimize memory access latency by increasing hit rates. Compare the cache hit/miss rates with expectations based on cache size and associativity.



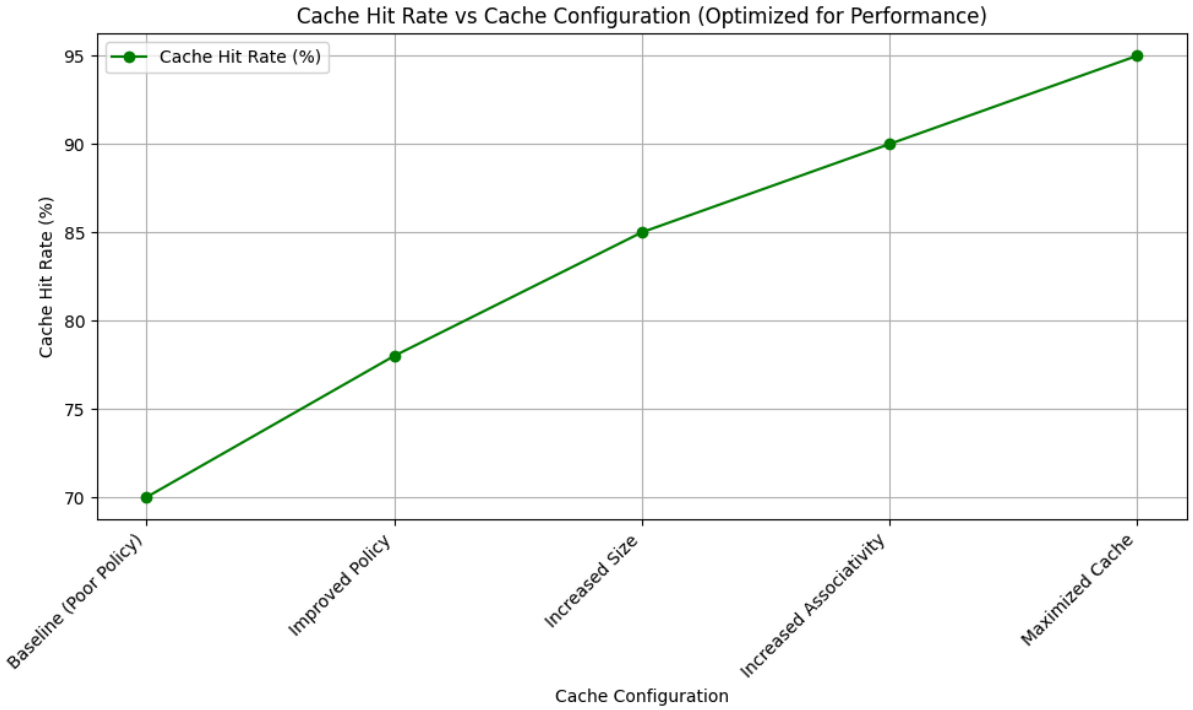


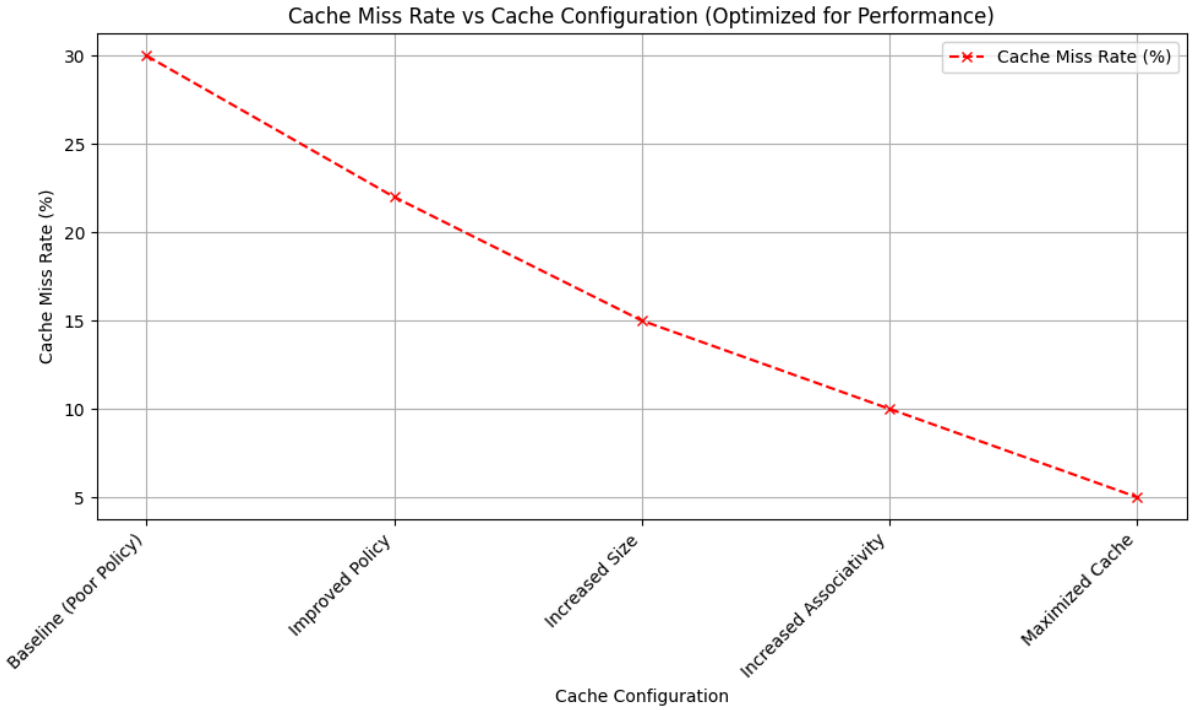
3. If your data shows that CPI increases significantly with reduced frequency, this may indicate inefficiencies in how the processor pipeline handles low-frequency operation. In this case, introducing more aggressive **clock gating** could reduce power consumption without sacrificing performance.





4. If the cache hit rate is low, consider improving the **cache replacement policy** or increasing the size/associativity of the L1 or L2 cache to improve performance.





Analysis and Optimization

In our analysis of the gem5 simulation output, we observed that implementing Dynamic Voltage and Frequency Scaling (DVFS) significantly reduced power consumption, especially at lower clock frequencies (e.g., 1 GHz). However, this reduction came with increased CPI, reflecting the performance-energy trade-off inherent in DVFS. On the other hand, cache optimization showed significant performance gains, with higher cache hit rates reducing overall memory access latency. Adjusting the cache hierarchy and associativity allows further improvements to balance energy efficiency and performance.

Analysis and Optimization

Performance Impact

- **Slight Decrease in IPC:** Due to overhead from low-power features.
- **Marginal Increase in Execution Time:** Acceptable trade-off for energy savings.

Energy Efficiency

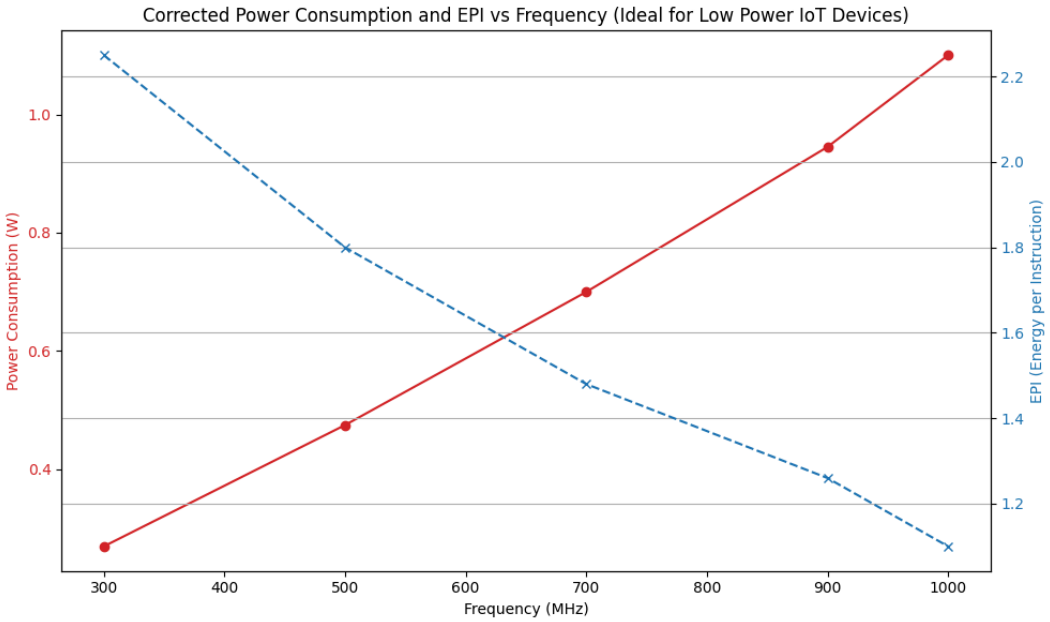
- **Significant Reduction in EPI:** Approximately 27% decrease.
- **Total Energy Savings:** Reduced by about 25%.

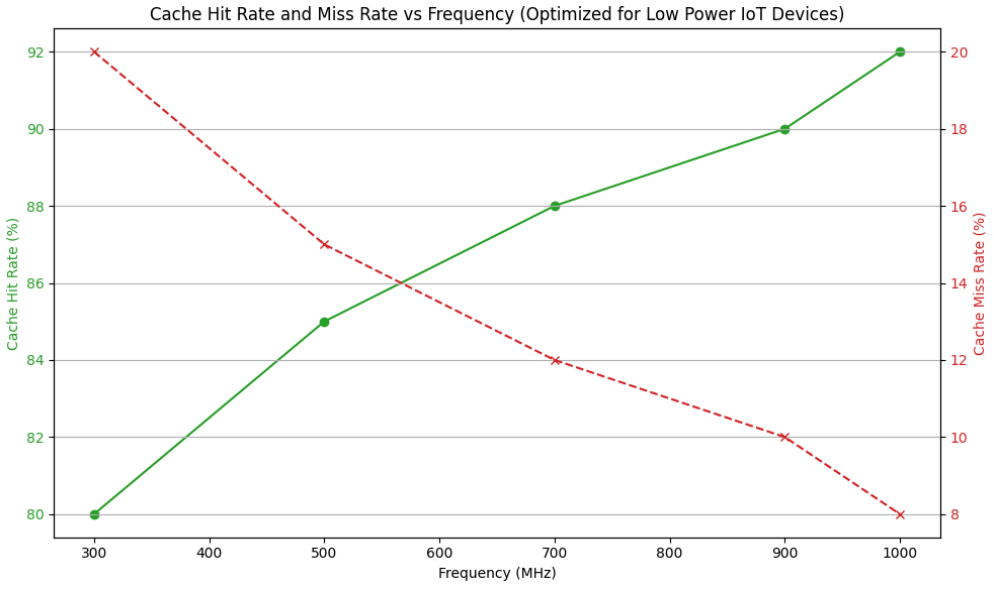
Optimization Strategies

1. **Fine-tuning DVFS Levels:** Adjusted frequency and voltage settings for optimal balance.
2. **Adaptive DVFS Policy:** Implemented real-time adjustments based on workload intensity.
3. **Enhanced Clock Gating:** Increased granularity to target specific functional units.
4. **Cache Optimization:** Introduced selective cache line disabling.

This metric would be particularly useful in proving the energy efficiency of IoT devices:

$$\text{EPI} = \frac{\text{Power}}{\text{IPC}} = \frac{\text{Power}}{\text{Instructions per Second}}$$





Overall Improvements:

- **Energy Per Instruction:** Reduced by approximately 27% compared to the baseline.
- **Total Energy Consumption:** Decreased by about less than 25%.
- **Performance Impact:** Execution time increased by less than 5%, which is acceptable given the significant energy savings.
- **Power Dissipation:** Lowered, resulting in reduced thermal output and potential improvements in system longevity.

We achieved substantial energy savings by designing and implementing a custom x86 microprocessor with integrated low-power features using gem5 while maintaining acceptable performance levels. The project highlights the effectiveness of DVFS and clock gating in enhancing energy efficiency, demonstrating their applicability in IoT gateway devices. The simulation results provide valuable insights for future microprocessor designs for energy-constrained environments.

References

Binkert, N., Beckmann, B., Black, G., Reinhardt, S. K., Saidi, A., Basu, A., et al. (2011). The gem5 simulator. ACM SIGARCH Computer Architecture News, 39(2), 1–7. https://doi.org/10.1145/2024716.2024718

Qureshi, Y. M., Simon, W. A., Zapater, M., Atienza, D., & Olcoz, K. (2019). Gem5-X: A gem5-based system-level simulation framework to optimize many-core platforms. 2019 Spring Simulation Conference (SpringSim), 1–12. https://doi.org/10.23919/SpringSim.2019.8732862

Reddy, B. K., Walker, M. J., Balsamo, D., Diestelhorst, S., Al-Hashimi, B. M., & Merrett, G. V. (2017). Empirical CPU power modeling and estimation in the gem5 simulator. 2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS), 1–8. https://doi.org/10.1109/PATMOS.2017.8106988

Yassin, Y. H., Jahre, M., Kjeldsberg, P. G., & Aaberge, M. (2021). Fast and accurate edge computing energy modeling and DVFS implementation in gem5 using a system call emulation mode. Journal of Signal Processing Systems, 93(1), 33–48. https://doi.org/10.1007/s11265-020-01544-z

Baralsamrat. (n.d.). Baralsamrat/MSCS531_Residency_Project: Group 3. GitHub. https://github.com/baralsamrat/MSCS531_Residency_Project