ECE-GY 6913, Computing System Architecture

Project (Phase I)

Komal Niraula (kn2505)

Optimizations and Features to Improve Performance

1. Parallel Execution:
    - Utilizing multi-core for parallel execution. Then, different cores can handle independent sets of instructions or threads. This will ensure that multiple streams of instructions are executed simultaneously.

2. Pipelining:
    - Implement a five-stage pipeline to process multiple instructions simultaneously at different stages (Fetch, Decode, Execute, Memory Access, Write-Back). Execution of one is overlapped by others, increasing the throughput considerably.
    - Ensure pipeline efficiency by dividing the execution flow into well-defined stages.

3. Hazard Detection and Forwarding:
    - Add hazard detection logic to identify data dependencies between instructions in the pipeline.
    - Implement forwarding (bypassing) to directly send data from one stage to another, minimizing pipeline stalls caused by dependencies.

4. Branch Prediction:
    - Integrate a branch predictor to speculatively fetch and decode instructions from the predicted branch path. This optimizes control flow in pipelined execution.

5. Optimize Data Access with Cache Memory:
    - Introduce instruction and data caches, which hold the memory locations that are most frequently used.

6. Reduce File I/O Overhead:
    - Use buffered I/O to batch read/write operations to/from files like imem.txt and dmem.txt in order to minimize the disk I/O latency.