Computer Architecture Phase 1

Demian Marín Martínez - dm6430

Friday 7 November 2025

1. Draw the schematic for a single stage processor and fill in your code in the to run the simulator. (20 points)
2. Draw the schematic for a five stage pipelined processor and fill in your code to run the simulator. The processor should be able to take care of RAW and control hazards by stalling and forwarding. (20 points)
3. Measure and report average CPI, Total execution cycles, and Instructions per cycle for both these cores by adding performance monitors to your code. (Submit code and print results to console or a file.) (5 points)
4. Compare the results from both the single stage and the five stage pipelined processor implementations and explain why one is better than the other. (5 points)
5. What optimizations or features can be added to improve performance? (Extra credit 1 point)

1.

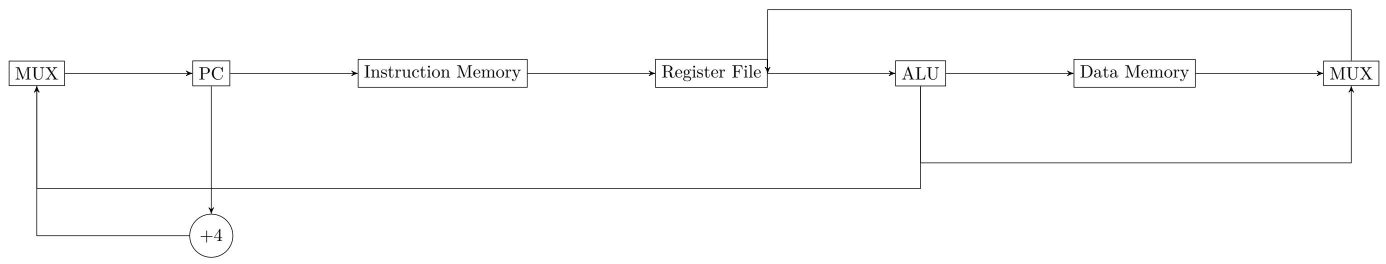

Figure 1: Single Stage Processor

2.

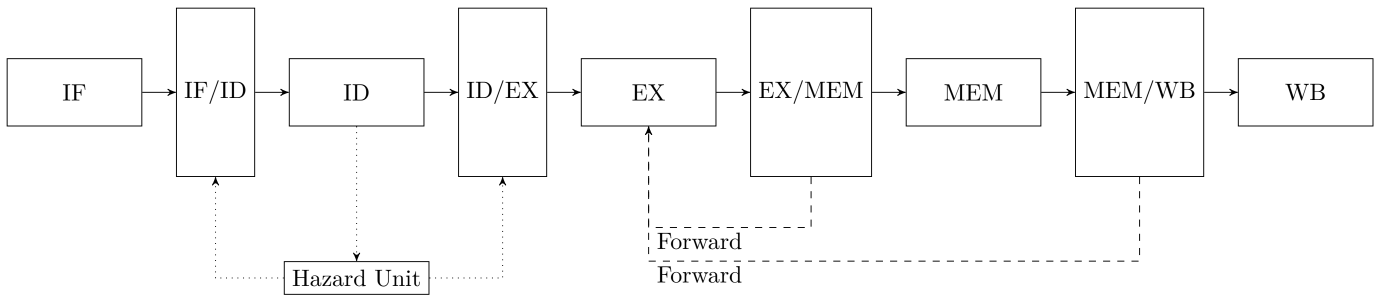

Figure 2: Five Stage Pipelined Processor

3. The performance metrics for the single-stage and five-stage processors are as follows:
   - **Single Stage Core**
     - Number of cycles taken: 36
     - Cycles per instruction (CPI): 1.0286
     - Instructions per cycle (IPC): 0.9722
   - **Five Stage Core**
     - Number of cycles taken: 9

  - Cycles per instruction (CPI): 1.5
  - Instructions per cycle (IPC): 0.6667

4. The five-stage pipelined processor is better than the single-stage processor, even though its CPI is higher. Here's why:

- **Throughput:** The five-stage processor has a much higher throughput. It can execute multiple instructions concurrently in different stages of the pipeline. This is evident from the total execution cycles: the five-stage processor took only 9 cycles to complete the program, while the single-stage processor took 36 cycles.
- **Clock Speed:** Pipelining allows for a faster clock speed. In a single-stage processor, the clock cycle is determined by the longest instruction. In a pipelined processor, the clock cycle is determined by the longest stage, which is much shorter. This means that even though an instruction takes 5 cycles to complete, a new instruction can start every cycle, leading to a significant speedup.
- **CPI vs. IPC:** While the single-stage processor has a better CPI (closer to 1), this is misleading. The CPI for the pipelined processor is higher due to stalls from hazards (data and control). However, because the clock cycle is much faster and it has higher throughput, the overall performance is better. The IPC for the single-stage processor is higher because it completes an instruction in almost every cycle, but each of those cycles is much longer.

In conclusion, the five-stage pipelined processor is significantly better due to its ability to overlap the execution of instructions, leading to a much lower total execution time. 5. Several optimizations and features can be added to improve the performance of the five-stage pipelined processor:

- **Branch Prediction:** To reduce the penalty of control hazards, a branch predictor can be implemented. Instead of stalling the pipeline every time a branch is encountered, the processor can predict the outcome of the branch and speculatively fetch and execute instructions from the predicted path. If the prediction is correct, there is no penalty. If it is incorrect, the pipeline is flushed, but this is often better than always stalling.
- **Superscalar Execution:** A superscalar processor can execute more than one instruction per clock cycle by having multiple pipelines. For example, a 2-way superscalar processor could fetch, decode, and execute two instructions at a time, potentially doubling the IPC.
- **Out-of-Order Execution:** This allows the processor to execute instructions as soon as their operands are available, rather than strictly in program order. This can hide the latency of instructions with long execution times (like loads from memory) and improve the utilization of the execution units.
- **More Advanced Forwarding:** While the current implementation has forwarding, more complex forwarding paths can be added to handle more types of data hazards without stalling.
- **Cache Hierarchy:** Adding a multi-level cache (L1, L2, L3) can significantly reduce the average memory access time. A faster cache can provide data to the processor much more quickly than main memory, reducing the number of stalls due to memory access.