



Arab American university

Majer :computer science

Course: Computer Architecture

Supervisor: Dr . Adnan Abu Asba

Students:

Sajed Kittanh

Mustafa Jarrar

Jameel Hamarshe

**Topic: Gem5 Study on heterogenous
performance**

1. Introduction and Project Objective

This report details a comparative performance study of two distinct CPU architectures—the Superscalar Out-of-Order (O3CPU) and the In-Order (X86 Minor CPU)—using the gem5 simulation framework. The primary objective was to quantitatively analyse the impact of pipeline complexity and execution paradigm on critical performance metrics under a consistent workload.

1.1 CPU Models and Metrics :

The evaluation focused on three fundamental performance factors:

- Total Execution Time**
- Total CPU Cycles**
- Instructions Per Cycle (IPC).**

2. Simulation Methodology

The gem5 simulator provided a controlled virtual environment, ensuring that core system parameters, such as cache hierarchy and size, remained constant across both simulation runs. The single variable modified was the CPU core type, guaranteeing a fair and isolated comparison of architectural efficiency. The workload, measured by the Total Committed Instructions, remained virtually identical (5,701 vs. 5,714 instructions), validating the comparison base.

3. Results and Performance Analysis

The simulation results conclusively demonstrated the significant performance advantages conferred by the Superscalar Out-of-Order architecture.

3.1. Quantification of Performance Gains

The O3CPU achieved substantial improvements across all time-based and efficiency :

3.2. Efficiency and Parallelism

The most significant finding is the 75% increase in IPC achieved by the O3CPU (0.14 vs. 0.08). IPC is the direct measure of how many instructions the core completes per clock cycle, quantifying the O3CPU's effective use of instruction-level parallelism (ILP). This efficiency gain is the root cause of the 43.1% reduction in total clock cycles required for the execution.

3.3. Cost of Pipeline Limitation (Idle Cycles)

The analysis of idle time highlights the inherent limitation of the In-Order architecture:

- The X86 Minor CPU accumulated 55,070 Total Idle Cycles. This vast number of stalled cycles confirms the poor tolerance of In-Order pipelines to data dependencies and memory latency. When an instruction stalls (e.g., waiting for data from memory), the entire pipeline stalls behind it.
- The O3CPU, by contrast, effectively minimizes these stalls. Its ability to reorder and execute subsequent, independent instructions around a stalled instruction drastically reduces the total effective wait time, directly translating into the observed cycle savings.

3.4. O3 Microarchitectural Utilization

The O3CPU's internal metrics confirm its robust design and utilization:

- Max Issue Width (8): The core is physically configured to dispatch up to 8 instructions per cycle, a key superscalar feature.
- Renaming Operation Count (17,534): The successful execution of over 17,000 Register Renaming operations is the technical proof of its Out-of-Order capability. This mechanism eliminates *false dependencies*, freeing the core to maximize ILP.
- Mean Issue Rate (0.76 Insts/Cycle): The high average issue rate confirms that the core was actively utilizing its wide pipeline resources throughout the simulation.

3.5. Speculation Cost

The O3CPU's aggressive speculative execution was highly efficient:

- The ratio of Instructions Issued (17,261) to Instructions Committed (5,701) shows a significant volume of speculative work being performed.
- Critically, only 33 instructions were squashed (discarded) due to mispredicted branches. This low squash count indicates a highly accurate branch predictor, ensuring that the performance gains from speculation far outweigh the minimal overhead cost.

4. Conclusion

- The simulation results provide empirical evidence affirming the superior performance of the Superscalar Out-of-Order (O3) architecture.
- The O3CPU achieved better results across all metrics—program speed, computational efficiency, and resource utilization—primarily by

leveraging instruction-level parallelism, register renaming, and efficient speculative execution.

- **This study successfully validates the design philosophy of modern high-performance processors and confirms that architectural complexity, specifically the implementation of OOO execution, is crucial for achieving high performance in contemporary computing environments. The gem5 simulator was an invaluable tool in providing this deep, quantitative insight into CPU behaviour.**