OoO & Speculation Project

To Be Done in Groups of 2

Lab Overview

The objective of this lab is to explore the interaction between out-of-order (OoO) execution, branch and memory speculation, and a two-level cache hierarchy using the gem5 simulator. The study focuses on quantifying the effect of these microarchitectural features on IPC, AMAT, cache behavior, and effective stall cycles.

Simulation Mode: SE Mode

- **Syscall Emulation (SE) Mode** is to be used:
  - Provides a controlled environment without OS interference.
  - Ideal for small, reproducible workloads.
  - Supports accurate measurement of OoO, speculation, and cache interactions.
- The gem5 configuration should include:
  - DerivO3CPU (OoO CPU model)
  - Private L1 instruction and data caches
  - Shared L2 cache
  - Configurable branch predictor
  - Optional memory-dependence predictor (speculative loads)

Microbenchmarks

Three categories of microbenchmarks are recommended:

1. **Compute-bound Kernel**: ALU-intensive loops to stress OoO execution.
2. **Pointer-chasing Kernel**: Linked-list traversal to test memory dependence speculation.
3. **Streaming Kernel**: Sequential array accesses to test cache prefetching and L1/L2 behavior.

You may implement these in C/C++ and cross-compile for the ISA supported by gem5.

Experimental Parameters

For each microbenchmark, students should perform simulations varying:

- OoO CPU parameters: ROB size, issue/commit width
- L1/L2 cache parameters: size, associativity, replacement policy
- Branch speculation: enabled/disabled, predictor type
- Memory-dependence speculation: enabled/disabled
- Optional prefetcher: on/off

Metrics to Collect

Students should collect and analyze the following metrics from gem5:

- **Performance:** IPC (Instructions per Cycle)
- **Cache Statistics:** L1D, L1I, and L2 hit rates; MPKI
- **Memory Latency:** AMAT at each cache level
- **Speculation Statistics:** Branch misprediction rate, memory-dependence speculation success/failure, pipeline flushes
- **Effective Stall Cycles:** Cycles lost waiting on memory after overlapping with OoO instructions
- **Energy Proxies:** Weighted access counts (L1, L2, DRAM)

Expected Workflow

1. Compile or cross-compile microbenchmarks for the chosen ISA.
2. Configure gem5 in SE mode with desired CPU and cache parameters.
3. Run baseline simulations (OoO CPU, no memory-dependence speculation).
4. Implement memory-dependence predictor (optional for bonus).
5. Sweep design parameters as listed above.
6. Parse and analyze gem5 stats to produce tables, plots, and comparative metrics.

Submission Guidelines

- A zipped Project folder containg
  - **Design Document (2–3 pages)**: Describe CPU and cache configurations, and expected behavior.
  - **gem5 Config Scripts**: Include all scripts used to launch experiments.
  - **Source Code**: Microbenchmarks and optional memory-dependence predictor modifications.
  - **Raw Stats**: gem5 output for all runs.
  - **Plots and Tables**: IPC, AMAT, hit rates, speculation success/failure, and effective stall cycles.
  - **Analysis Report (3–5 pages)**: Trends, insights, and final design recommendations.
  - **Optional for *Ego* Points**: Fully implemented memory-dependence predictor with recovery and demonstration of performance impact.
  - Clearly Identify who has done what part of the assignment