Residency Project: Designing and Implementing a Microprocessor Using gem5

Simulation Software

(Deliverable 2)

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

Phase 2: Implementation Using Gem5

This phase focuses on implementing the designed microprocessor architecture using the gem5 simulation software using M1 Macbook 13 running MacOS. The implementation involves translating the architectural specifications into gem5 configurations, integrating the defined low-power features, and validating the design through preliminary simulations.

Setup Gem5 Environment

Installation and Configuration of gem5 on macOS with M1 Processor The following steps outline the installation and configuration process.

Prerequisites ensure your development environment meets the following requirements:

1. **OS**: macOS Monterey (12.0) or later.

2. **scons:** Build system used by gem5.

3. **python@3.9:** Specific Python version compatible with gem5.

4. **swig:** Interface compiler needed for generating Python bindings.

5. **protobuf:** Protocol Buffers library for serialization.

6. **gcc:** GNU Compiler Collection for building gem5.

7. **git:** Version control system to clone the gem5 repository.

Local Machine Specs

1. **Model:** Macbook 13 Air

2. **OS:** MacOS Sonoma (UNIX\LINUX based system)

3. **Processor:** 8 Core M1 ARM Processor
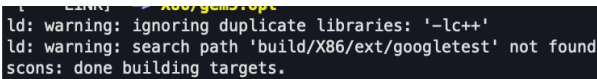
4. **Ram**: 8GB (Unifed Memory)

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

Cloning the gem5 Repository

Clone the gem5 source code from the official repository:

git clone https://github.com/gem5/gem5

Building gem5

Navigate to the gem5 directory and build the simulator for the X86 architecture:

1. **ARM to x86 Translation:** Since gem5 is being built for the X86 ISA on an ARM-based processor (M1), ensure that cross-compilation is configured correctly. gem5 supports cross-compilation, but additional configuration may be necessary.

2. **Familiarization with gem5 Components and Scripting** gem5 uses Python scripts for configuration, allowing customization of simulation parameters (Binkert et al., 2011). Familiarize yourself with:

   a. **Configuration Scripts:** Located in the configs directory, these scripts define system components like CPUs, caches, and memory.

   b. **SimObjects:** The fundamental building blocks representing hardware components.

   c. **Event-Driven Simulation:** gem5 operates on an event-driven model, where events trigger actions in the simulation timeline.

3. **Implementation** on bash

   a. cd gem5
   b. scons build/X86/gem5.opt -j 4

   ```
   [ LINK]   > X86/gem5.opt
   ld: warning: ignoring duplicate libraries: '-lc++'
   ld: warning: search path 'build/X86/ext/googletest' not found
   scons: done building targets.
   ```
   c.

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

Implement Microprocessor Design

Gem5 Configurations Architectural Specifications

Create a custom configuration script called "config.py" to define the processor based on the architectural specifications:

ISA: X86.

CPU Model: Use Intel Processor for an out-of-order CPU mode

Pipeline Depth: Configure according to the 7-stage pipeline with a delay of 1 cycle. The pipeline stages roughly correspond to the following stages in a 7-stage pipeline:

1. **Fetch (IF):** Instruction fetch stage.

2. **Decode (ID):** Instruction decode stage.

3. **Rename (REN):** Register renaming stage.

4. **Dispatch/Issue (DIS/IS):** Instructions are dispatched to the issue queue.

5. **Execute (EX):** Execution of instructions.

6. **Writeback (WB)**: Results are written back.

7. **Commit (COM):** Instructions are committed in order.

Configuring the Cache Hierarchy is as follows.

❖ **Pipeline stage width** sets fetchWidth, decodeWidth, renameWidth, issueWidth, and commitWidth parameters that define how many instructions can be processed simultaneously at each stage. Setting them to 4 means each stage can handle up to 4 instructions per cycle.

❖ **Reorder Buffer (ROB)** sets *numROBEntries* parameter which sets the size of the ROB, which is critical for out-of-order execution. A larger ROB allows more instructions to be in flight, effectively deepening the pipeline.

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

❖ **Instruction Queues** define the sizes of the instruction and load-store queues, respectively on *numIQEntries*, and *numLSQEntries* parameters.

❖ **Pipeline Stage latency** configs delays between stages (e.g., *decodeToFetchDelay*, *renameToDecodeDelay*, etc.) represent the number of cycles it takes for instructions to move from one stage to the next.

❖ **Multi-level cache hierarchy**:

➢ **L1l Cache:** 16 KB, 8-way set-associative for instruction and data caches.

➢ **L1d Cache:** 32 KB, 8-way set-associative for instruction and data caches.

➢ **L2 Cache:** 256 KB, 8-way set-associative.

Implementing Low-power Features within gem5

Dynamic Voltage and Frequency Scaling (DVFS) by defining multiple clock domains and adjusting frequencies during simulation (Yassin et al., 2021). During the simulation, you can schedule events to change the frequency based on workload conditions.

❖ **Clock Gating reduces** dynamic power consumption (Reddy et al., 2017) and adjusts the activity thresholds to simulate clock gating effectiveness.

❖ **Power Gating** in gem5 doesn't natively support power gating and export activity statistics for integration with external power modeling tools like McPAT or use gem5's power models to estimate the impact (Reddy et al., 2017).

Validation of Implementation

Executing Simulations:

File name: config.py

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

Run simulations using the custom script:

```
└ ./build/x86/gem5.opt configs/deprecated/example/Group3_M51.py -c tests/test-progs/hello/bin/x86/linux/hello
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 23 2024 16:42:01
gem5 started Oct 20 2024 11:00:42
gem5 executing on dhcp-10-37-3-22.nlc.guest.dcccd.edu, pid 76199
command line: ./build/x86/gem5.opt configs/deprecated/example/Group3_M51.py -c tests/test-progs/hello/bin/x86/linux/hello
```

Debugging and Refining the Design

Executing Simulations:

 The custom configuration script (config.py) can be found here.

Run simulations using the custom script and utilize gem5's debugging tools:

- **Debug Flags**: Activate with --debug-flags=Exec, Cache, Power**.**

- **Stats Files:** Analyze stats.txt for performance metrics like IPC (Instructions Per

    Cycle), cache hit rates, and power estimates.

Addressing Errors

```
 ~/Projects/gem5  git residency *1 ~1 +9 ?4                                      ✓  10:55:55 ⊘
└ ./build/x86/gem5.opt configs/deprecated/example/Group3_M51.py -c tests/test-progs/hello/bin/x86/linux/hello
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 23 2024 16:42:01
gem5 started Oct 20 2024 10:55:56
gem5 executing on dhcp-10-37-3-22.nlc.guest.dcccd.edu, pid 76030
command line: ./build/x86/gem5.opt configs/deprecated/example/Group3_M51.py -c tests/test-progs/hello/bin/x86/linux/hello

Global frequency set at 1000000000000 ticks per second
warn: failed to generate dot output from m5out/config.dot
src/base/statistics.hh:279: warn: One of the stats is a legacy stat. Legacy stat is a stat that does not belong to any statistics::Group.
 Legacy stat is deprecated.
fatal: system.cpu_clk_domain.clock without default or user set value
 ~/Projects/gem5  git residency *1 ~1 +9 ?4                                   1 x  10:55:56 ⊘
```

```
 ~/Projects/gem5  git residency *1 ~1 +9 ?4                              1 x  10:5556 ⊘
└ ./build/x86/gem5.opt configs/deprecated/example/Group3_M51.py -c tests/test-progs/hello/bin/x86/linux/hello
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 23 2024 16:42:01
gem5 started Oct 20 2024 10:56:44
gem5 executing on dhcp-10-37-3-22.nlc.guest.dcccd.edu, pid 76106
command line: ./build/x86/gem5.opt configs/deprecated/example/Group3_M51.py -c tests/test-progs/hello/bin/x86/linux/hello

Global frequency set at 1000000000000 ticks per second
warn: failed to generate dot output from m5out/config.dot
src/base/statistics.hh:279: warn: One of the stats is a legacy stat. Legacy stat is a stat that does not belong to any statistics::Group.
 Legacy stat is deprecated.
system.remote_gdb: Listening for connections on port 7000
src/mem/cache/base.cc:199: fatal: Cache ports on system.cpu.dcache are not connected
Memory Usage: 413189808 KBytes
 ~/Projects/gem5  git residency *1 ~1 +9 ?4                                   1 x  10:56:44 ⊘
```

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

```
system.cpu.icache.cpu_side = system.cpu.icache_port
system.cpu.dcache.cpu_side = system.cpu.dcache_port

system.cpu.icache.mem_side = system.membus.cpu_side_ports
system.cpu.dcache.mem_side = system.membus.cpu_side_ports

system.cpu.createInterruptController()
```

```
self.system.cpu_clk_domain.clock = freq
self.system.clk_domain.clock = freq
self.system.cpu_voltage_domain.voltage = voltage
self.system.cpu_clk_domain.voltage_domain.voltage = voltage
return self.system
```

Validation Procedures

Performance analysis of the gem5 implementation includes microprocessor design elements

such as cache configuration (L1ICache, L1DCache, L2Cache). These caches directly impact

Instruction Throughput and Cache Performance metrics. By setting properties such as

associativity, latency, and queue sizes, you can analyze how effectively the design achieves

the desired Instructions Per Cycle (IPC) and cache hit/miss ratios.

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

Power estimation includes dynamic voltage and frequency scaling (DVFS) through the gate_control class allows the toggling of the system's frequency and voltage. This is essential for estimating power consumption, as it simulates power adjustments under varying workload conditions. By using gem5's built-in power models or external tools like McPAT, you can analyze how changes in frequency and voltage affect overall power consumption.

Low-power feature Verification was the implementation of DVFS (toggle_gate_control function) allows verification of DVFS Effectiveness, ensuring that frequency scaling is applied and impacts power consumption. The Clock Gating feature can also be examined by observing the dynamic power changes during periods of idle activity, which can be simulated by adjusting the clock and analyzing the activity statistics provided in gem5's output (stats.txt).

Limitations

Fixed Pipeline Structure: The DerivO3CPU model has a fixed pipeline structure. Modifying the number of pipeline stages would require changes to the CPU model's source code.

Simulating Pipeline Depth: Adjusting delays and buffer sizes allows for simulating different pipeline depths without changing the underlying CPU model.

References

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

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

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