Computer Architecture
Introduction to gem5

Spring 2024

Professor Asadi
Mohsen Ghasemi & Khosro Moeini

Topics

This Session:

- Introduction to gem5
- Introduction to CPU Model
- Introduction Memory System
- “Hello World!” in gem5

Next Session:

- Creating your own configuration script
- In-depth look at gem5 CPU Model
- In-depth look at gem5 Memory System

Introduction to gem5

What is gem5?

The gem5 simulator is a modular platform for computer-system architecture research, encompassing system-level architecture as well as processor microarchitecture.

gem5 was originally conceived for computer architecture research in academia, but it has grown to be used in computer system design by academia, industry for research, and in teaching.

What is gem5?

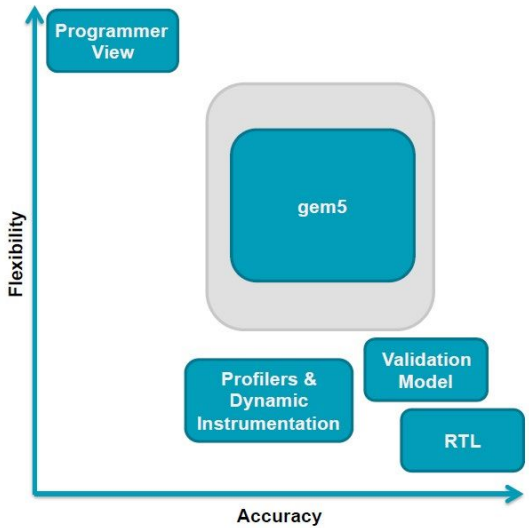
- University of Michigan m5 project
- University of Wisconsin–Madison GEMS project
- Merge of m5 and GEMS in 2011
- gem5 has been cited by over 2900 publications
- Used by many industrial research labs including:
  - AMD Research, Google, HP, Samsung

Why gem5?

- Accuracy and flexibility
- ISA support
- Patches & Add-ons
- Simulation modes

Accuracy & Flexibility

gem5 provides good accuracy as well as flexibility.



ISA Support

gem5 supports Many ISAs:

- ARM
- x86
- MIPS
- RISC-V
- And more…

Patches & Add-ons

Different patches and add-ons over the years. Add-ons like:

- GPU
  - GPGPU-Sim
- Main Memory
  - DRAMSim2
  - Ramulator
  - ThyNVM
  - NVMain
- SSD
  - MQSim

Simulation modes

gem5 support different simulation modes:

- Full System (FS) Mode
  - Simulating an entire system
  - Booting operating system
  - Requires OS image
  - More detailed
- System Call Emulation (SE) Mode
  - Simpler
  - Faster

We are going to use SE mode in this presentation.

gem5 CPU Model

CPU Models

gem5 provides four CPU models:

- Simple one-CPI CPU
- Detailed model of an in-order CPU
- Detailed model of an out-of-order CPU (OoO, O3)
- KVM-based CPU (Virtualization)

SimpleCPU

The SimpleCPU is a purely functional, in-order model that is suited for cases where a detailed model is not necessary. Recently broken into three classes:

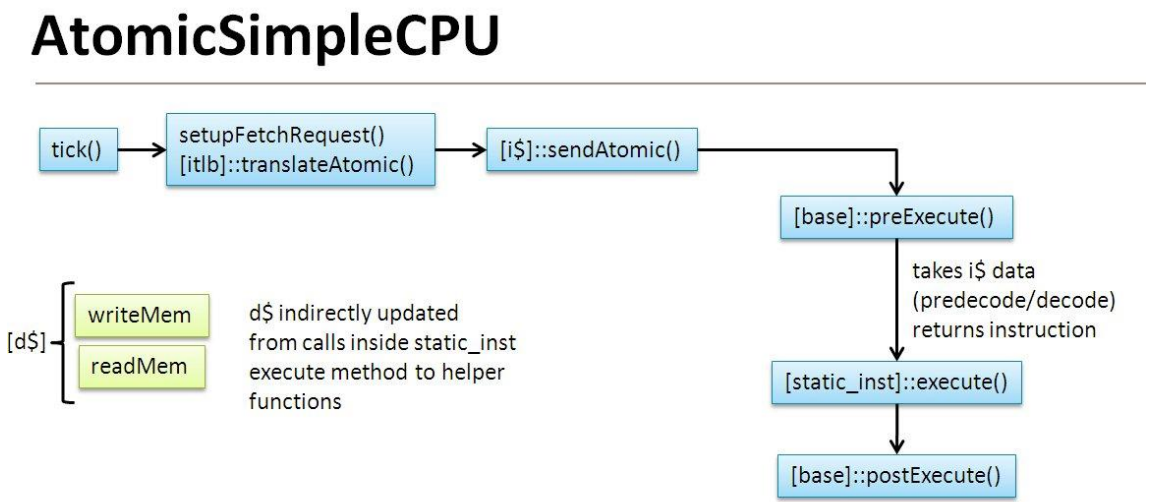
- BaseSimpleCPU
- AtomicSimpleCPU
- TimingSimpleCPU

BaseSimpleCPU

- Handles:
  - Fetch request
  - Pre-execute setup
  - Post-execute actions
  - Advancing the PC to the next instruction
- Can not run on its own:
  - You must use one of the classes that inherits from it:
    - AtomicSimpleCPU
    - TimingSimpleCPU

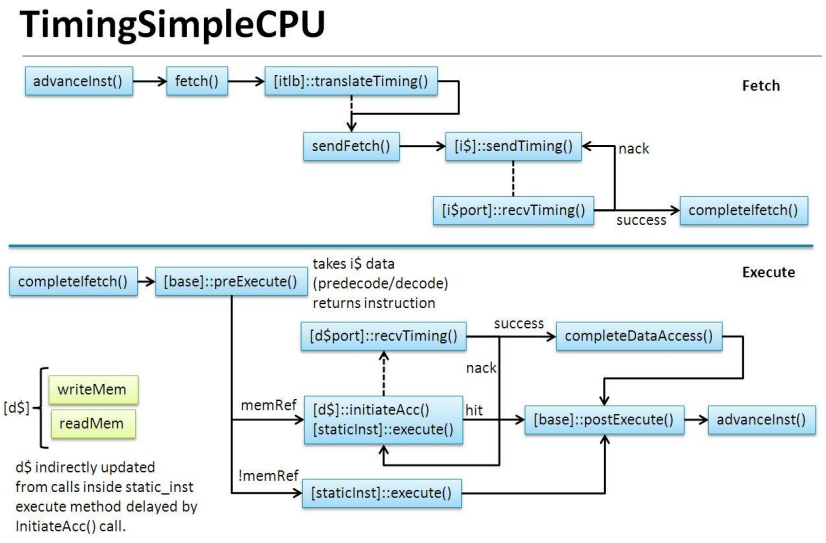
AtomicSimpleCPU

Uses atomic memory accesses (more details next session).



TimingSimpleCPU

Uses timing memory accesses (more details next session).



gem5 Memory System

Memory System

- MemObjects
- Ports
- Connections
- Request
- Packet
- Access Types

MemObjects

All objects that connect to the memory system inherit from MemObject

- This class adds the following virtual functions:
  - `getMasterPort(const std::string &name, PortID idx)`
  - `getSlavePort(const std::string &name, PortID idx)`
- This interface is used to connect MemObjects together

Ports

Used to interface memory objects together:

- MasterPort sends requests and receives responses
- SlavePort receives requests and sends responses
- There are two groups of functions in the port object:
  - `send*`
  - `recv*`
- Example:
  - Master: `myPort->sendTimingReq(pkt)`
  - Slave: `peer->recvTimingReq(pkt)`

Ports

What types of ports a CPU has?

Only MasterPort. Called a master module.

What types of ports a memory controller has?

Only SlavePort. Called a slave module.

What types of ports a cache has?

Both types. Called an interconnect component.

Connections

Two objects can specify that their ports should be connected using the assignment operator.

`A.port1 = B.port2` has the same meaning as `B.port2 = A.port1`.

Request

Request: Used to encapsulates the original request issued by a CPU or I/O device. Request object fields include:

- Virtual address. May be invalid if the request was issued on a physical address.
- Physical address
- Data size
- Time the request was created
- The ID of the CPU/thread that caused this request. May be invalid if the request was not issued by a CPU (e.g., a device access or a cache writeback).
- The PC that caused this request. May be invalid if the request was not issued by a CPU.

Packet

Packet: Used to encapsulate a transfer between two objects in the memory system. A packet contains the following:

- The address
- Data Size
- A pointer to the data being manipulated
- Status
- A pointer to the request
- And more…

Access Types

There are three types of accesses supported by the ports.

1. Timing: The most detailed access. Best effort for realistic timing. Include the modeling of queuing delay and resource contention.
2. Atomic: When a atomic access is sent the response is provided when the function returns. Atomic and timing accesses can not coexist in the memory system.
3. Functional: Like atomic accesses functional accesses happen instantaneously. Can coexist in the memory system with atomic or timing accesses.

“Hello World!” in gem5

Building gem5

You don’t need to. We will give you a VM.

If you really want to build it yourself, follow these steps. Linux is highly recommended.

“Hello World!” in gem5

gem5 executable @ ../build/X86/gem5.opt

Config file @ configs/learning_gem5/part1/simple.py

To run hello world:

./build/X86/gem5.opt configs/learning_gem5/part1/simple.py

```
Usage
=====
  gem5.opt [gem5 options] script.py [script options]

gem5 is copyrighted software; use the --copyright option for details.

Options
=======
--help, -h              show this help message and exit
--build-info, -B        Show build information
--copyright, -C         Show full copyright information
--readme, -R            Show the readme
--outdir=DIR, -d DIR    Set the output directory to DIR [Default: m5out]
--redirect-stdout, -r   Redirect stdout (& stderr, without -e) to file
--redirect-stderr, -e   Redirect stderr to file
--silent-redirect       Suppress printing a message when redirecting stdout or
                        stderr
--stdout-file=FILE      Filename for -r redirection [Default: simout.txt]
--stderr-file=FILE      Filename for -e redirection [Default: simerr.txt]
--listener-mode={on,off,auto}
                        Port (e.g., gdb) listener mode (auto: Enable if
                        running interactively) [Default: auto]
--allow-remote-connections
                        Port listeners will accept connections from anywhere
                        (0.0.0.0). Default is only localhost.
--interactive, -i       Invoke the interactive interpreter after running the
                        script
--pdb                   Invoke the python debugger before running the script
--path=PATH[:PATH], -p PATH[:PATH]
                        Prepend PATH to the system path when invoking the
                        script
--quiet, -q             Reduce verbosity
--verbose, -v           Increase verbosity
-m mod                  run library module as a script (terminates option
                        list)
-c cmd                  program passed in as string (terminates option list)
-P                      Don't prepend the script directory to the system path.
                        Mimics Python 3's `-P` option.
-s                      IGNORED, only for compatibility with python. don'tadd
                        user site directory to sys.path; also PYTHONNOUSERSITE

Statistics Options
------------------
--stats-file=FILE       Sets the output file for statistics [Default:
                        stats.txt]
--stats-help            Display documentation for available stat visitors

Configuration Options
---------------------
```

“Hello World!” Statistics

Statistics @ m5out/stats.txt

You can change it using --out-dir=<your_out_dir> and --stats-file=<your_stat_file> arguments!

```
---------- Begin Simulation Statistics ----------
simSeconds                                   0.000006                       # Number of seconds simulated (Second)
simTicks                                      5943000                       # Number of ticks simulated (Tick)
finalTick                                     5943000                       # Number of ticks from beginning of simulation (restored from checkpoints and never reset) (Tick)
simFreq                                  1000000000000                       # The number of ticks per simulated second ((Tick/Second))
hostSeconds                                      0.02                       # Real time elapsed on the host (Second)
hostTickRate                                322954027                       # The number of ticks simulated per host second (ticks/s) ((Tick/Second))
hostMemory                                     650136                       # Number of bytes of host memory used (Byte)
simInsts                                        5701                       # Number of instructions simulated (Count)
simOps                                         10302                       # Number of ops (including micro ops) simulated (Count)
hostInstRate                                  302681                       # Simulator instruction rate (inst/s) ((Count/Second))
hostOpRate                                    546786                       # Simulator op (including micro ops) rate (op/s) ((Count/Second))
system.clk_domain.clock                         1000                       # Clock period in ticks (Tick)
system.cpu.numCycles                           11887                       # Number of cpu cycles simulated (Cycle)
system.cpu.cpi                              2.080329                       # CPI: cycles per instruction (core level) ((Cycle/Count))
system.cpu.ipc                              0.480693                       # IPC: instructions per cycle (core level) ((Count/Cycle))
system.cpu.numWorkItemsStarted                     0                       # Number of work items this cpu started (Count)
system.cpu.numWorkItemsCompleted                   0                       # Number of work items this cpu completed (Count)
system.cpu.commitStats0.numInsts                5714                       # Number of instructions committed (thread level) (Count)
system.cpu.commitStats0.numOps                 10315                       # Number of ops (including micro ops) committed (thread level) (Count)
system.cpu.commitStats0.numInstsNotNOP             0                       # Number of instructions committed excluding NOPs or prefetches (Count)
system.cpu.commitStats0.numOpsNotNOP               0                       # Number of Ops (including micro ops) Simulated (Count)
system.cpu.commitStats0.cpi                 2.080329                       # CPI: cycles per instruction (thread level) ((Cycle/Count))
system.cpu.commitStats0.ipc                 0.480693                       # IPC: instructions per cycle (thread level) ((Count/Cycle))
```

Next Session Topics

- Creating your own configuration script
- In-depth look at gem5 CPU Model
- In-depth look at gem5 Memory System

Don’t miss it!

Thank you!