The gem5 Tutorial @ISCA 2024

Plan for the day

- Intro
  - Break
- Using gem5, standard library, and gem5-resources
- **[Alen Sabu]** Elfies and gem5
  - Lunch
- Extending gem5
  - Break
- **[Matt S., Matt P., & Vishnu]** Running ML workloads and gem5's GPU model

Introduction

In this section we'll talk about gem5's history, the purpose and uses of computer architecture simulation, some nomeclature, and gem5's software architecture

Outline

- What is gem5 and a bit of history
- My perspective on architecture simulation
- gem5's (software) architecture
- Getting started using gem5

Created at Michigan by students of Steve Reinhardt, principally Nate Binkert.

"A tool for simulating systems"

Two Views of M5

1. A framework for event-driven simulation
   - Events, objects, statistics, configuration
2. A collection of predefined object models
   - CPUs, caches, busses, devices, etc.

- This tutorial focuses on #2
- You may find #1 useful even if #2 is not

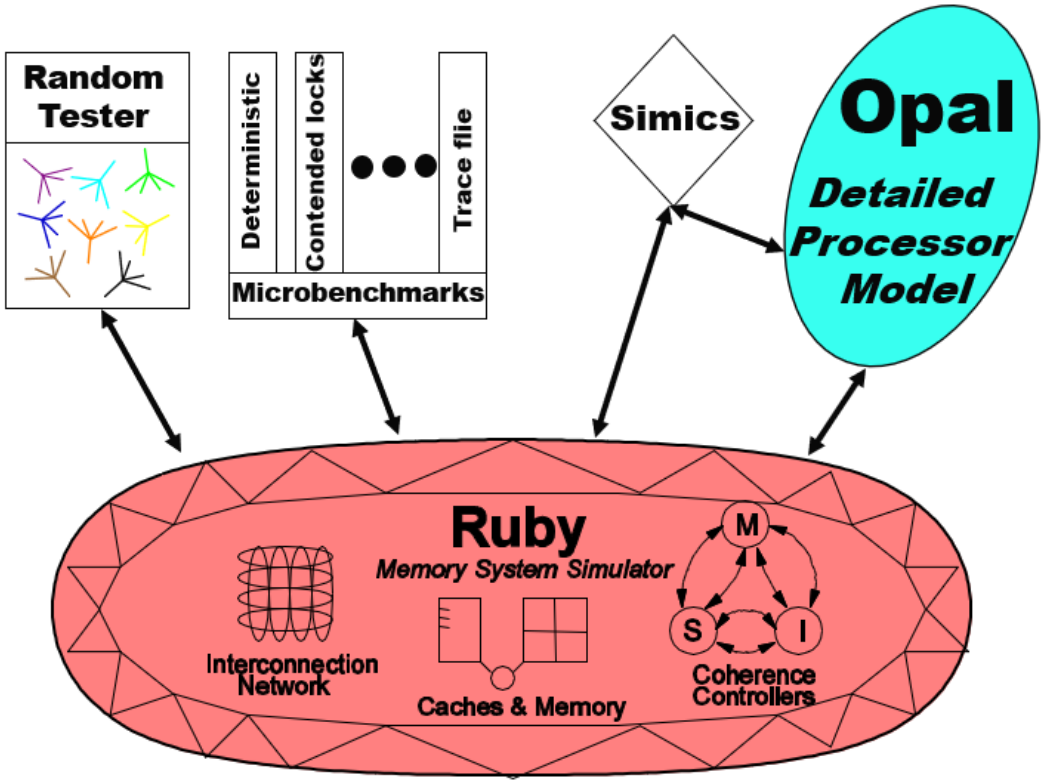
Created at Michigan by students of Steve Reinhardt, principally Nate Binkert.

"A tool for simulating systems"

Created at Wisconsin by students of Mark Hill and David Wood.

Detailed memory system

GEMS From 50,000 Feet



Created at Michigan by students of Steve Reinhardt, principally Nate Binkert.

"A tool for simulating systems"

Created at Wisconsin by students of Mark Hill and David Wood.

Detailed memory system

What is gem5?

Michigan m5 + Wisconsin GEMS = gem5

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

Lowe-Power et al. **The gem5 Simulator: Version 20.0+**. ArXiv Preprint ArXiv:2007.03152, 2021. https://doi.org/10.48550/arXiv.2007.03152

Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K. Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R. Hower, Tushar Krishna, Somayeh Sardashti, Rathijit Sen, Korey Sewell, Muhammad Shoaib, Nilay Vaish, Mark D. Hill, and David A. Wood. 2011. **The gem5 simulator**. *SIGARCH Comput. Archit. News* 39, 2 (August 2011), 1-7. DOI=http://dx.doi.org/10.1145/2024716.2024718

gem5-20+: A new era in CA simulation

gem5 Version 20.0

Abdul Mutaal Ahmad
Adrian Herrera
Adrien Pesle
Adrià Armejach
Akash Bagdia
Alec Roelke
Alexandru Dutu
Ali Jafri
Ali Saidi
Amin Farmahini
Anders Handler
Andrea Mondelli
Andrea Pellegrini
Andreas Hansson
Andreas Sandberg
Andrew Bardsley
Andrew Lukefahr
Andrew Schultz
Andriani Mappoura
Ani Udipi
Anis Peysieux
Anouk Van Laer
Arthur Perais
Ashkan Tousi
Austin Harris
Avishai Tvila
Ayaz Akram

Bagus Hanindhito
Benjamin Nash
Bertrand Marquis
Binh Pham
Bjoern A. Zeeb
Blake Hechtman
Bobby R. Bruce
Brandon Potter
Brian Grayson
Cagdas Dirik
Chander Sudanthi
Chen Zou
Chris Adeniyi-Jones
Chris Emmons Evans
Christian Menard
Christoph Pfister
Christopher Torng
Chuan Zhu
Chun-Chen Hsu
Ciro Santilli
Clint Smullen

Curtis Dunham
Dam Sunwoo
Dan Gibson
Daniel Carvalho
Daniel Johnson
Daniel Sanchez
David Guillen-Fandos
David Hashe
David Oehmke
Djordje Kovacevic
Dongxue Zhang
Doğukan Korkmaztürk
Dylan Johnson
Earl Ou
Edmund Grimley Evans
Emilio Castillo
Erfan Azarkhish
Eric Van Hensbergen
Erik Hallnor
Erik Tomusk
Faissal Sleiman
Fernando Endo

Gabe Black
Gabe Loh
Gabor Dozsa
Gedare Bloom
Gene WU
Gene Wu
Geoffrey Blake
Georg Kotheimer
Giacomo
Gabrielli
Hamid Reza Khaleghzadeh
Hanhwi Jang
Hoa Nguyen
Hongil Yoon
Hsuan Hsu
Hussein Elnawawy
Ian Jiang
IanJiangICT
Ilias Vougioukas
Isaac Richter
Isaac Sánchez Barrera
Ivan Pizarro
Jack Whitham
Jairo Balart

Jakub Jermar
James Clarkson
Jan-Peter Larsson
Jason Lowe-Power
Javier Bueno Hedo
Javier Cano-Cano
Javier Setoain
Jieming Yin
Jing Qu
Jiuyue Ma
Joe Gross
Joel Hestness
John Alsop
John Kalamatianos
Jordi Vaquero
Jose Marinho
Jui-min Lee
Kanishk Sugand
Karthik Sangaiah
Ke Meng
Kevin Brodsky
Kevin Lim
Khalique

Koan-Sin Tan
Korey Sewell
Krishnendra Nathella
Lena Olson
Lisa Hsu
Lluc Alvarez
Lluís Vilanova
Mahyar Samani
Malek Musleh
Marco Balboni
Marco Elver
Marjan Fariborz
Matt DeVuyst
Matt Evans
Matt Horsnell
Matt Poremba
Matt Sinclair
Matteo Andreozzi
Matteo M. Fusi
Matthew Poremba
Matthias Hille
Matthias Jung
Maurice Becker
Maxime

Martinasso Maximilian Stein Maximilien Breughe Michael Adler Michael LeBeane Michael Levenhagen Michiel Van Tol Miguel Serrano Mingyuan Mitch Hayenga Mohammad Alian Monir Mozumder Moyang Wang Mrinmoy Ghosh Nathan Binkert Nathanael Premillieu Nayan Deshmukh Neha Agarwal Nicholas Lindsay Nicolas Derumigny

Nicolas Zea
Nikos Nikoleris
Nils Asmussen
Nuwan Jayasena
Ola Jeppsson
Omar Naji
Pablo Prieto
Palle Lyckegaard
Pau Cabre
Paul Rosenfeld
Polina Dudnik
Polydoros Petrakis
Pouya Fotouhi
Prakash Ramrakhyani
Pritha Ghoshal
Radhika Jagtap
Rahul Thakur
Reiley Jeapaul
Rekai Gonzalez-Alberquilla
Rene de Jong
Ricardo Alves
Richard D. Strong
Richard Strong
Rico Amslinger

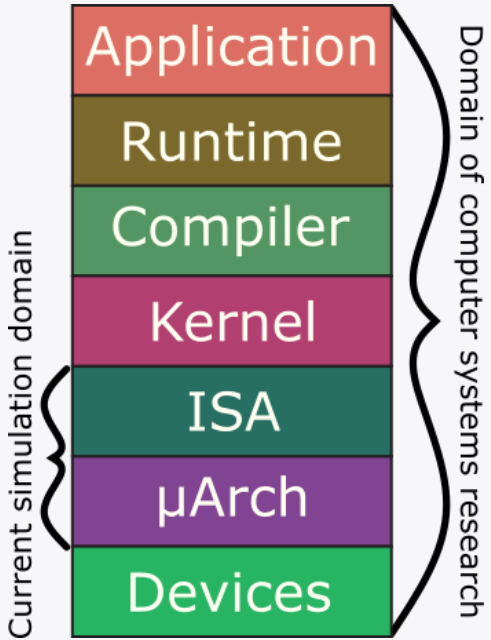
Riken Gohil
Rizwana Begum
Robert Kovacsics
Robert Scheffel
Rohit Kurup
Ron Dreslinski
Ruben Ayrapetyan
Rune Holm
Sandipan Das
Santi Galan
Sascha Bischoff
Sean McGoogan
Sean Wilson
Sergei Trofimov
Severin Wischmann
Shawn Rosti
Sherif Elhabbal
Siddhesh Poyarekar
Somayeh Sardashti
Sooraj Puthoor
Sophiane Senni
Soumyaroop Roy

Srikant Bharadwaj
Stan Czerniawski
Stanislaw Czerniawski
Stephan Diestelhorst
Stephen Hines
Steve Raasch
Swapnil Haria
Taeho Kgil
Tao Zhang
Thomas Grass
Tiago Mück
Tim Harris
Timothy Hayes
Timothy M. Jones
Tom Jablin
Tommaso Marinelli
Tony Gutierrez
Trivikram Reddy
Tuan Ta
Tushar Krishna
Umesh Bhaskar

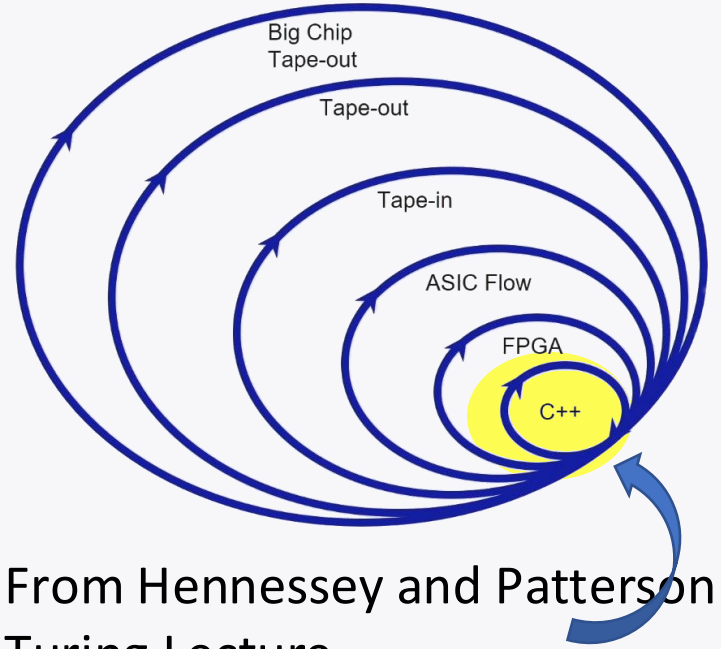
Uri Wiener
Victor Garcia
Vilas Sridharan
Vince Weaver
Vincentius Robby
Wade Walker
Weiping Liao
Wendy Elsasser
William Wang
Willy Wolff
Wang
seanzw
Éder F. Zulian

Your name here!

gem5's goals



Agile Hardware Dev. Methodology



From Hennessey and Patterson Turing Lecture

gem5

gem5's goals

- Anyone (including non-architect) can download and use gem5

- Used for cross-stack research:

  Change kernel, change runtime, change hardware, all in concert

  Run full ML stacks, full AR/VR stacks… other emerging apps

  We're close… just a lot of rough edges! You can help!

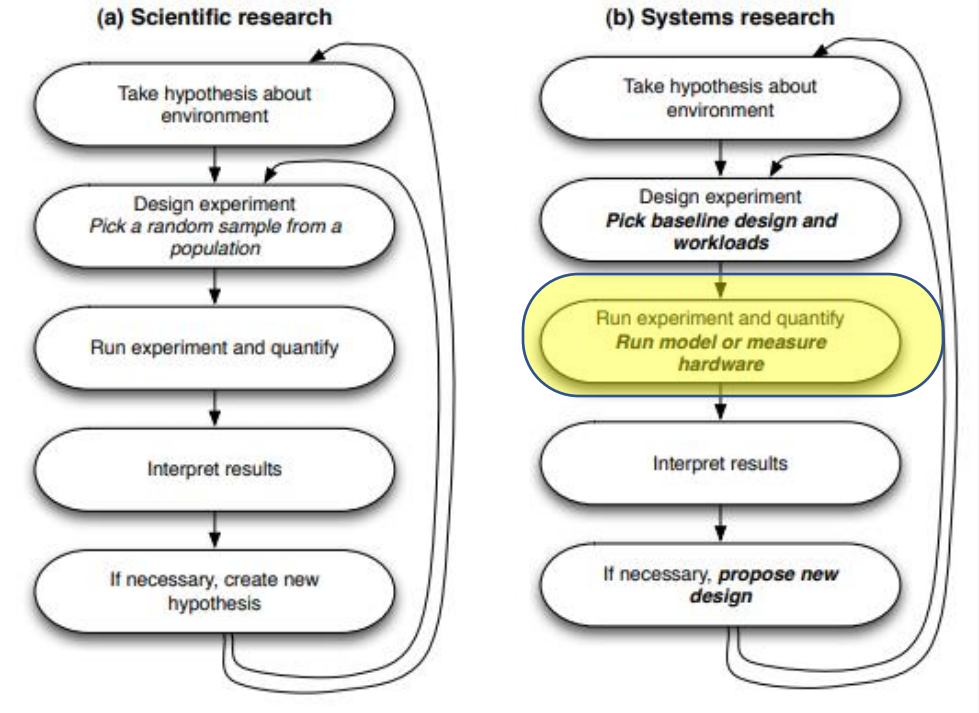
The gem5 community

- 100s of contributors & 1000s(?) of users
- Aim to meet the needs of
  - Academic research (most of you all!)
  - Industry research and development
  - Classroom use
- Code of conduct (see repo)

- **I want to see the community grow!**

My views on simulation

Computer systems research/engineering



From Computer Architecture Performance Evaluation Methods by Lieven Eeckhout

Computer architecture simulation!

gem5

Why simulation

- Need a tool to evaluate systems that don't exist (yet)
  - Performance, power, energy, etc.
- Very costly to actually make the hardware
- Computer systems are complex with many interdependent parts
  - Not easy to be accurate without the full system
- Simulation can be parameterized
  - Design-space exploration
  - Sensitivity analysis

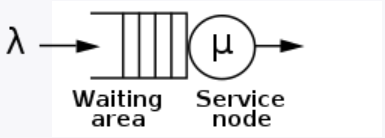
Alternatives to cycle-level simulation

Analytic models

Amdahl's Law:

$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

Queueing models:



gem5

Kinds of simulation

- Functional simulation
- Instrumentation-based
- Trace-based
- Execution-driven
- Full system

Kinds of simulation

- Functional simulation
  - Executes programs correctly. Usually no timing information
  - Used to validate correctness of compilers, etc.
  - RISC-V Spike, QEMU, gem5 “atomic” mode
- Instrumentation
  - Often binary translation. Runs on actual hardware with callbacks
  - Like trace-based. Not flexible to new ISA. Some things opaque
  - PIN, NVBit

Kinds of simulation

- Trace-based simulation
  - Generate addresses/events and re-execute
  - Can be fast (no need to do functional simulation). Reuse traces
  - If execution depends on timing, this will not work!
  - “Specialized” simulators for single aspect (e.g., just cache hit/miss)
- Execution-driven
  - Functional and timing simulation is combined
  - gem5 and many others
  - gem5 is “execute in execute” or “timing directed”

Full system simulation

- Components modeled with enough fidelity to run mostly unmodified apps
- Often “Bare metal” simulation
- All of the program is functionally emulated by the simulator
- Often means running the OS in the simulator, not faking it

- “Full system” simulators are often combine functional and execution-based

Nomenclature

Host: the actual hardware you're using

Running things directly on the hardware:

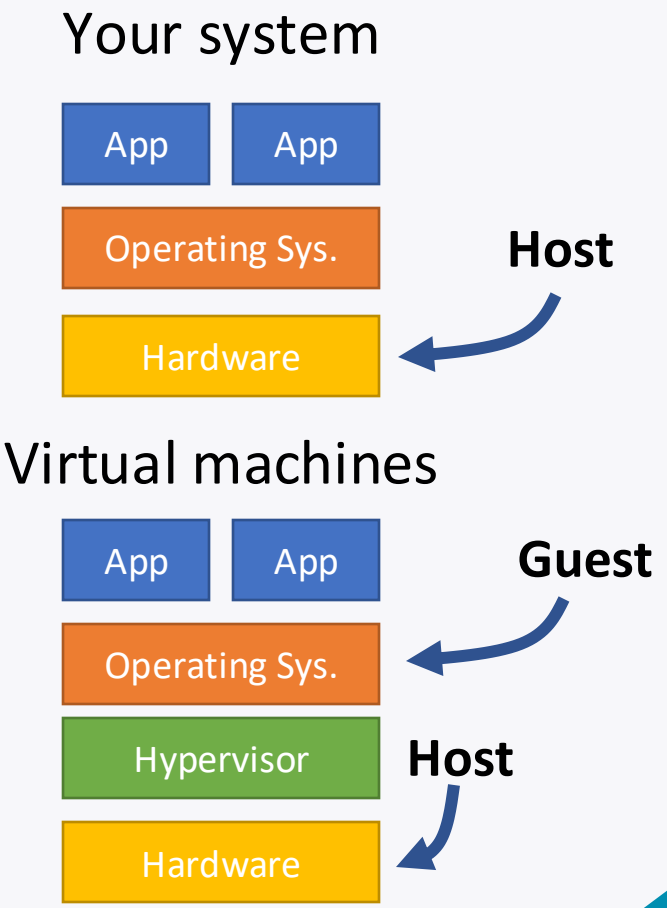
Native execution

Guest: Code running on top of "fake" hardware

OS in virtual machine is guest OS

Running "on top of" hypervisor

Hypervisor is emulating hardware



gem5

Nomenclature

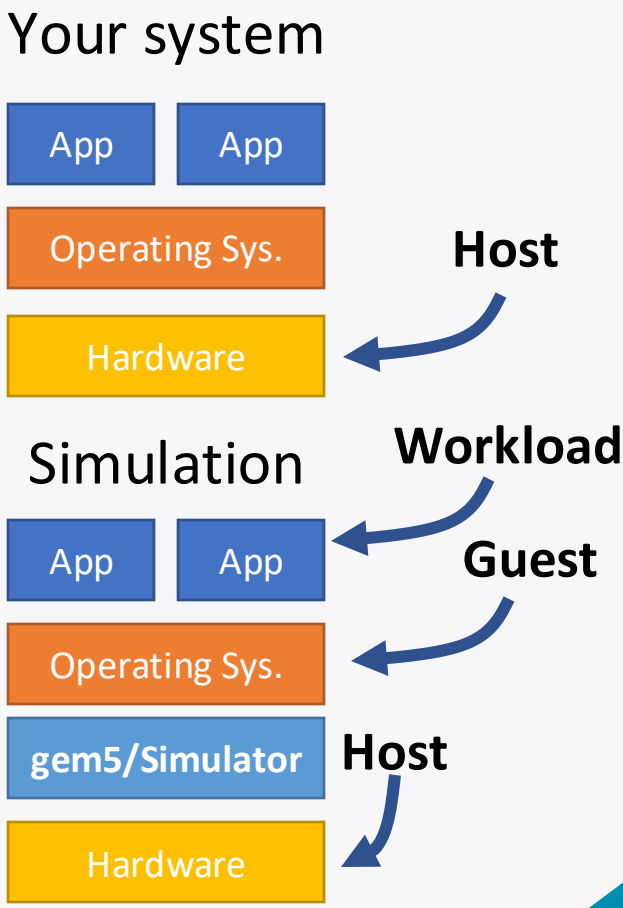
Host: the actual hardware you're using

Simulator: Runs on the host
Exposes hardware to the guest

Guest: Code running on *simulated* hardware
OS running on gem5 is guest OS
gem5 is simulating hardware

Simulator's code: Runs natively
executes/emulates the guest code

Guest's code: (or benchmark, workload, etc.)
Runs on gem5, not on the host.



gem5

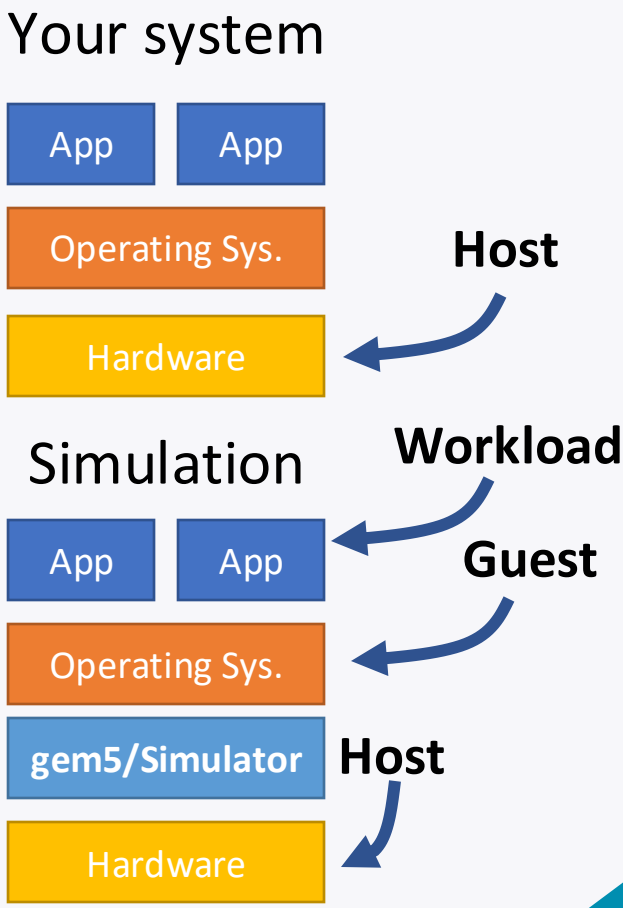
Nomenclature

Host: the actual hardware you're using

Simulator: Runs on the host
Exposes hardware to the guest

Simulator's performance:
Time to run the simulation on host
Wallclock time as you perceive it

Simulate*d* performance:
Time predicted by the simulator
Time for guest code to run on simulator



gem5

Tradeoffs

Development time: time to make the simulator/models
Evaluation time: wallclock time to run the simulator
Accuracy: How close is the simulator to *real* hardware
Coverage: How broadly can the simulator be used?

	Development time	Evaluation time	Accuracy	Coverage
functional simulation	excellent	good	poor	poor
instrumentation	excellent	very good	poor	poor
specialized cache and predictor simulation	good	good	good	limited
full trace-driven simulation	poor	poor	very good	excellent
full execution-driven simulation	very poor	very poor	excellent	excellent

https://www.morganclaypool.com/doi/abs/10.2200/S00273ED1V01Y201006CAC010

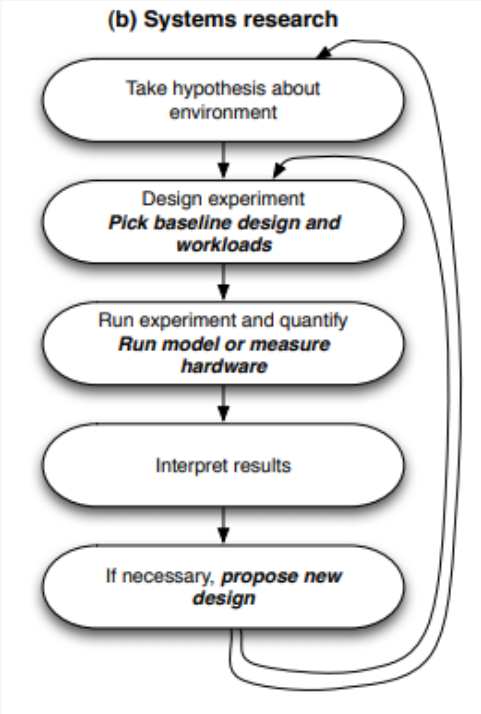
gem5

What “level” should we simulate?

- Ask yourself: What fidelity is required for this question?
  - Example: New register file design
- Often, the answer is a mix.

- gem5 is well suited for this mix
  - Models with different fidelity
  - Drop-in replacements for each other

- “Cycle level” vs “cycle accurate”



RTL simulation

- RTL: Register transfer level/logic
  - The “model” is the hardware design
  - You specify every wire and every register
  - Close to the actual ASIC
- This is “cycle accurate” as it should be the same in the model and in an ASIC
- Very high fidelity, but at the cost of configurability
  - Need the entire design
  - More difficult to combine functional and timing

Cycle-level simulation

- Models the system cycle-by-cycle
- Often “event-driven” (we’ll see this soon)
- Can be quite accurate
  - Not the exact same cycle-by-cycle as the ASIC, but similar timing
- Easily parameterizeable
  - No need for a full hardware design
- Faster than cycle-accurate
  - Can “cheat” and functionally emulate some things

gem5's architecture

C++

Models → gem5 has 100s of models

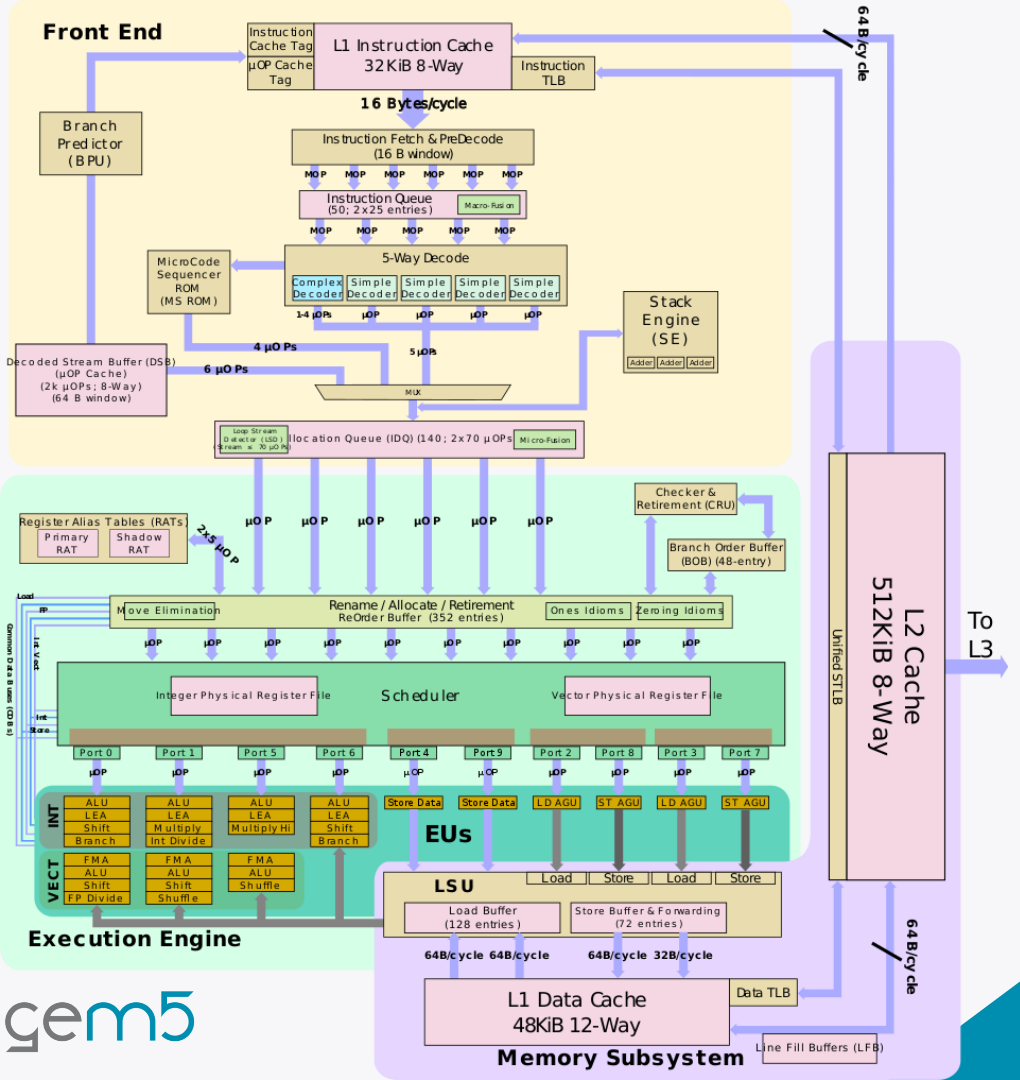
- Cache
  - parameters: size
- Core
  - parameters
    - width
    - isq
    - stages
- DRAM
  - parameters
    - tRCD
    - tRAS
    - tCAS
    - ⋮

gem5 architecture: SimObject

- **Model**
  C++ code in src/
- **Parameters**
  Python code in src/
  In SimObject declaration file
- **Instance or configuration**
  A particular choice for the parameters
  In standard library, your extensions, or python runscript

Model vs parameters

- Generic model and timing in C++
- Expose parameters to Python
- Set parameters and connections in Python



https://en.wikichip.org/wiki/intel/microarchitectures/sunny_cove

gem5

Some nomenclature

You can ***extend*** a model, to model new things
You would want to *inherit* from the object in C++

```
class O3CPU : public BaseCPU
{
```

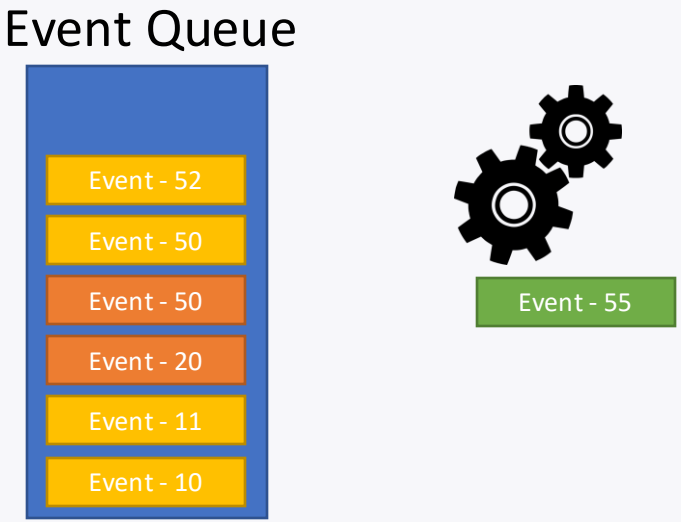
You can ***specialize*** a model with specific parameters
You would want to *inherit* from the object in python

```
class i7CPU(O3CPU):
   issue_width = 10
```

gem5

gem5 architecture: Simulating

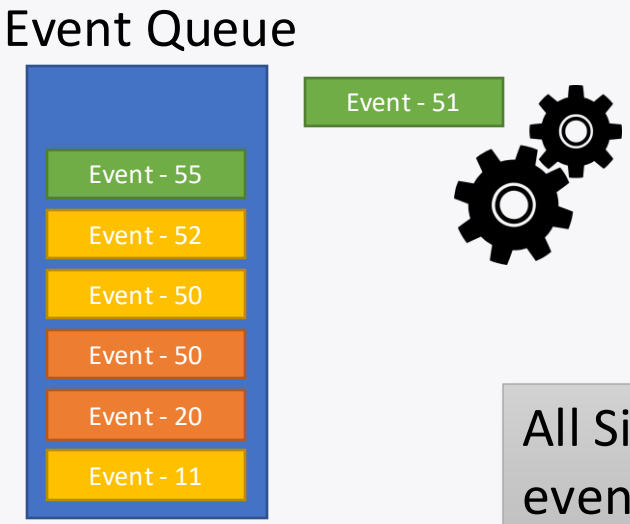
gem5 is a **discrete event simulator**



1) Event at head dequeued
2) Event executed
3) More events queued

gem5 architecture: Simulating

gem5 is a **discrete event simulator**

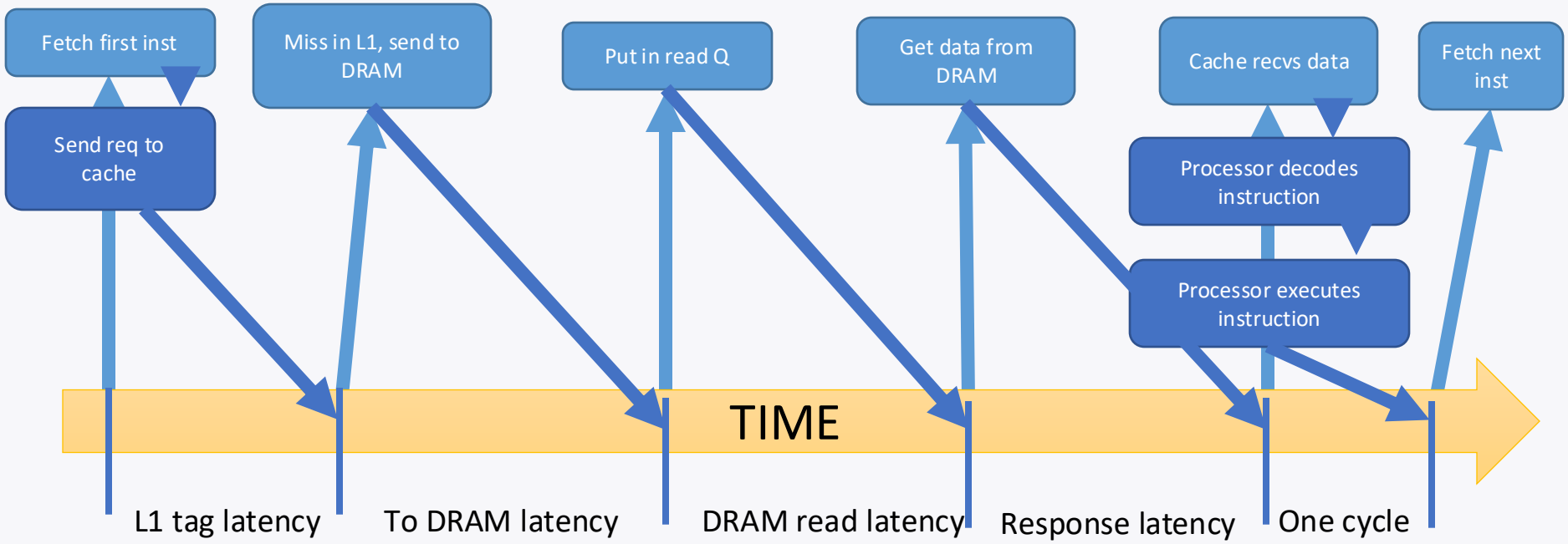


1) Event at head dequeued
2) Event executed
3) More events queued

All SimObjects can enqueue events to the event queue

gem5

Discrete event simulation example



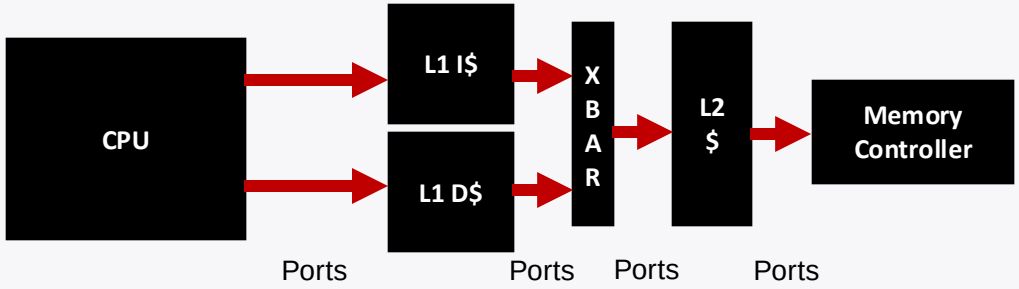
gem5

Discrete event simulation

- "Time" needs a unit
  In gem5, we use a unit called "Tick"
- Need to convert a simulation "tick" to user-understandable time
  E.g., seconds
- This is the global simulation tick rate
  Usually this is 1 ps per tick or 10^{12} ticks per second

gem5's Main abstractions

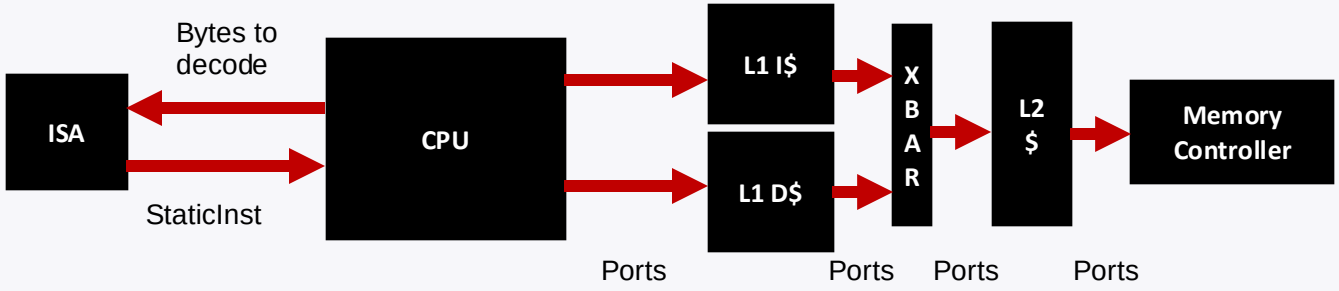
- **ISA vs CPU model** & **Memory request** *abstractions*
- **Ports** allow you to send requests and receive responses
  - **Ports** are unidirectional (two types, request/response)
  - Anything* with a *Request* port can be connected to any *Response* port



gem5

gem5's Main abstractions

- **ISA vs CPU model** & **Memory requests** *abstractions*
- **CPU model** *abstracted* from ISA
  - **Any*** CPU model can be used with any* ISA
  - **ISA** provides *StaticInst* with *execute(), initiateAccess(), etc*.
  - If you implement the interface, you can add new CPU model or new ISA



gem5