What is gem5 and where do I get it?

Andreas Sandberg & Nikos Nikoleris

ARM Research

Why gem5?

Runs real workloads

- Runs complex workloads like Android & ChromeOS

System-level insights

- Device interactions (storage, NICs, …)
- OS interactions like PA fragmentation

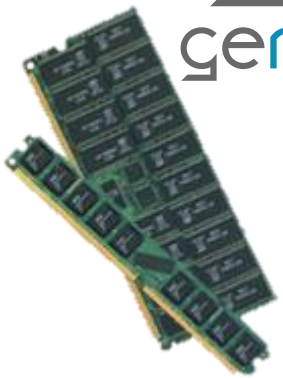
Can be wired to custom models

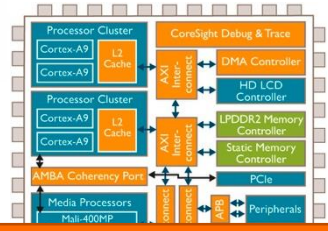
- *Add detail where it matters, when it matters!*

Rapid *early* prototyping

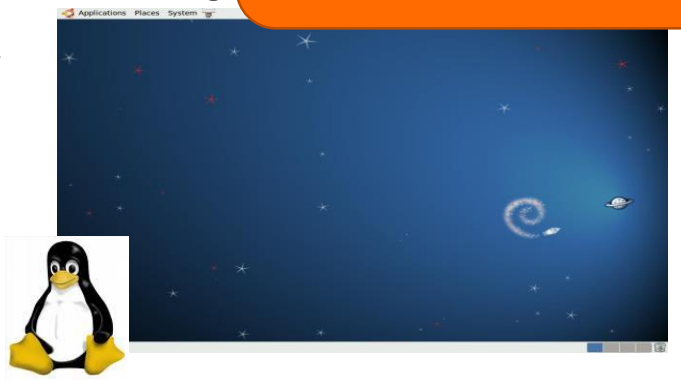
- Parameterized models enable rapid design space exploration

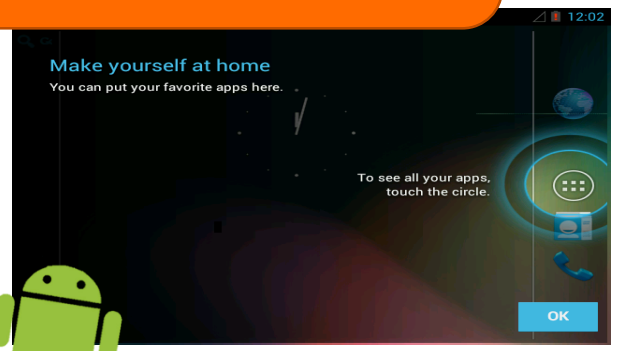
Large user base in industry & academia



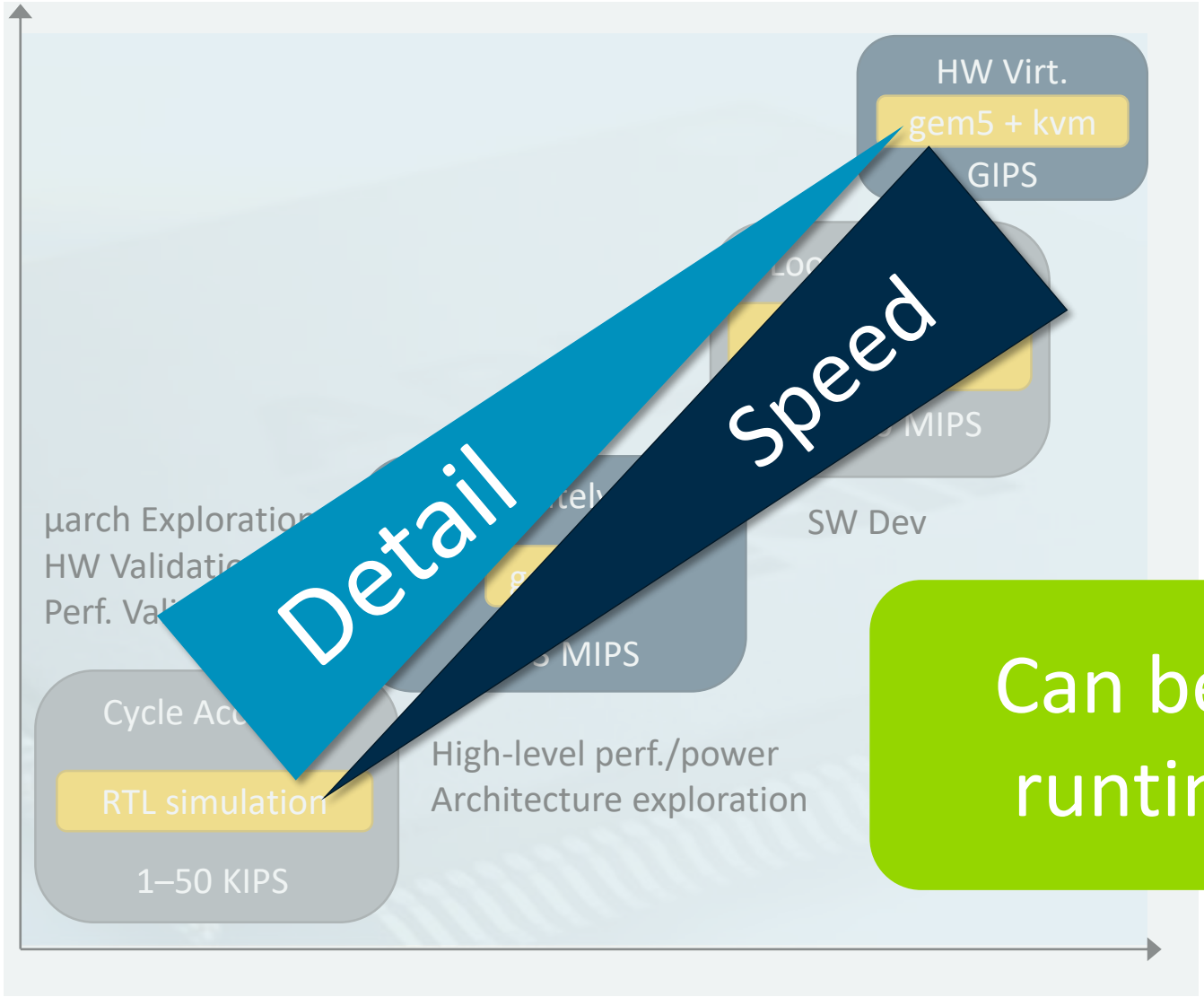


But **not** a microarchitectural model out of the box!





Configurable level of detail



Can be changed at runtime in gem5!

arm

When not to use gem5

Performance validation

- gem5 is not a cycle-accurate microarchitecture model!
- This typically requires more accurate models such as RTL simulation.

Core microarchitecture exploration*

- gem5's core models were not designed to replace more accurate microarchitectural models.

To validate functional correctness

- New (e.g., Armv8.0+) or optional instructions are sometimes not implemented.
- gem5 is not as rigorously tested as commercial products.

How to get involved

Tutorials: http://gem5.org/Tutorials

- Jason's Learning gem5 tutorials: http://learning.gem5.org/
- The ASPLOS 2017 slides provide an up-to-date general overview

Mailinglists: http://gem5.org/Mailing_Lists

- gem5-dev: Development discussions
- gem5-users: Using gem5 and running experiments

Contribute some code: https://gem5-review.googlesource.com/ (see CONTRIBUTING.md)

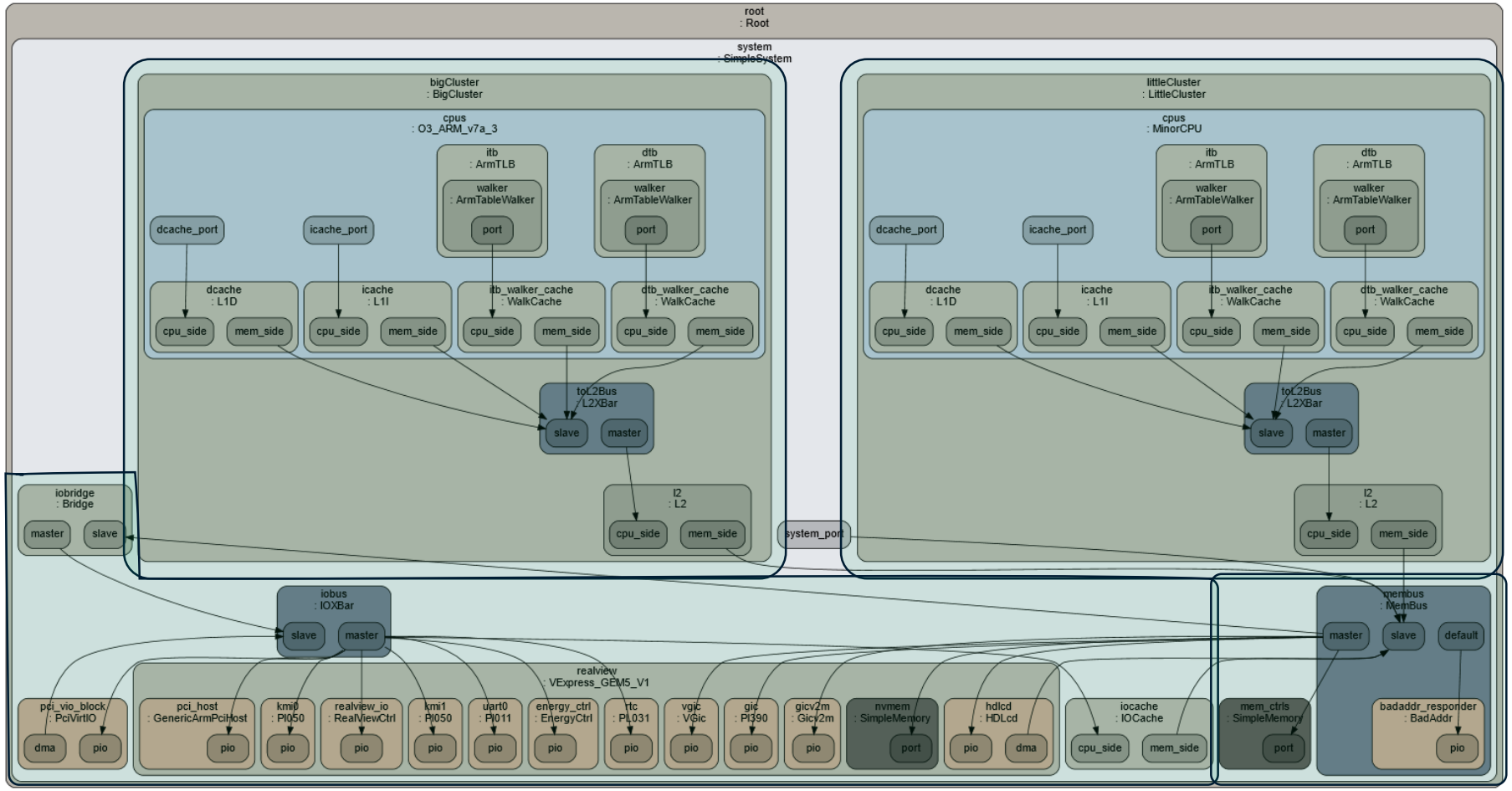
gem5 is a community effort!

arm

A brief technical overview

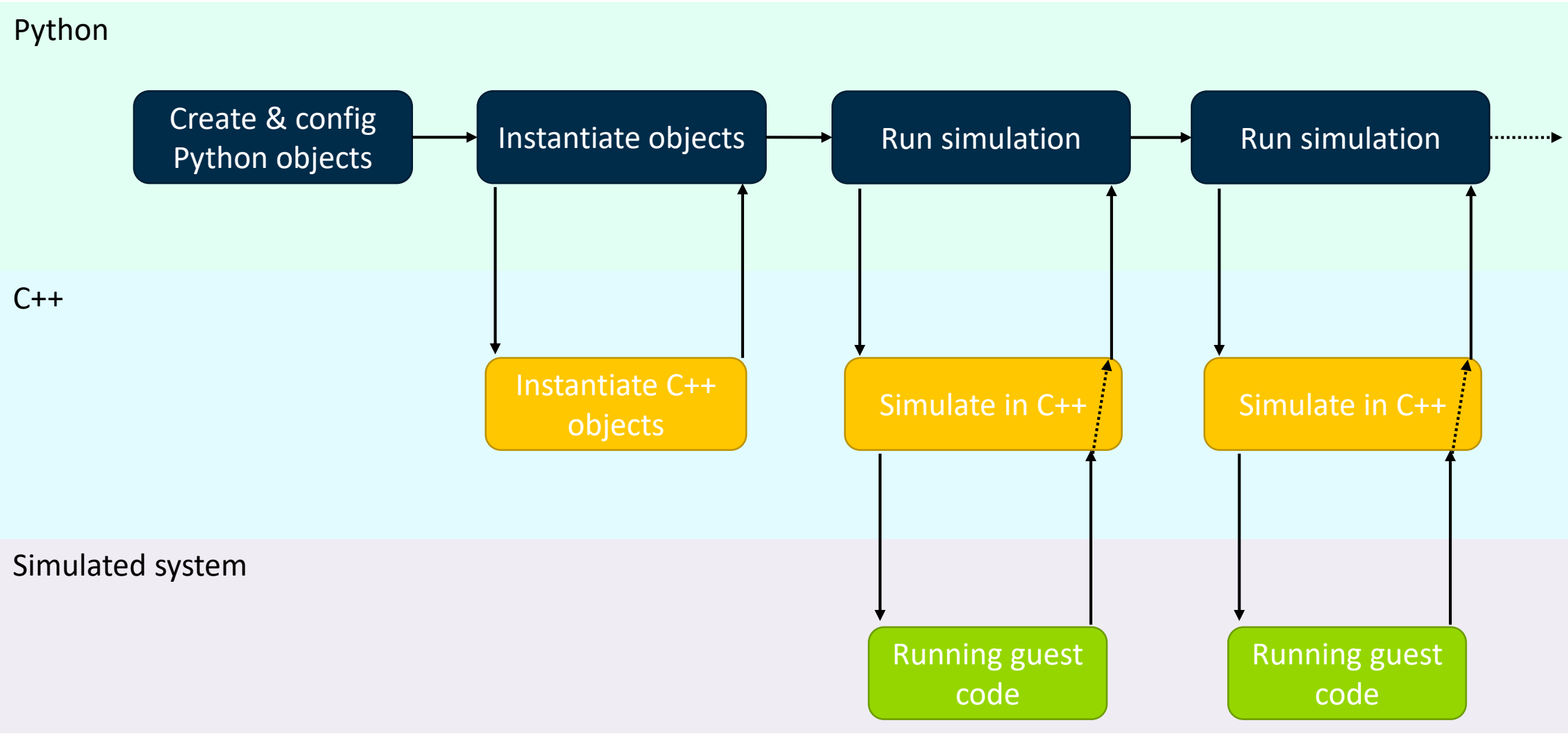
arm

Example System



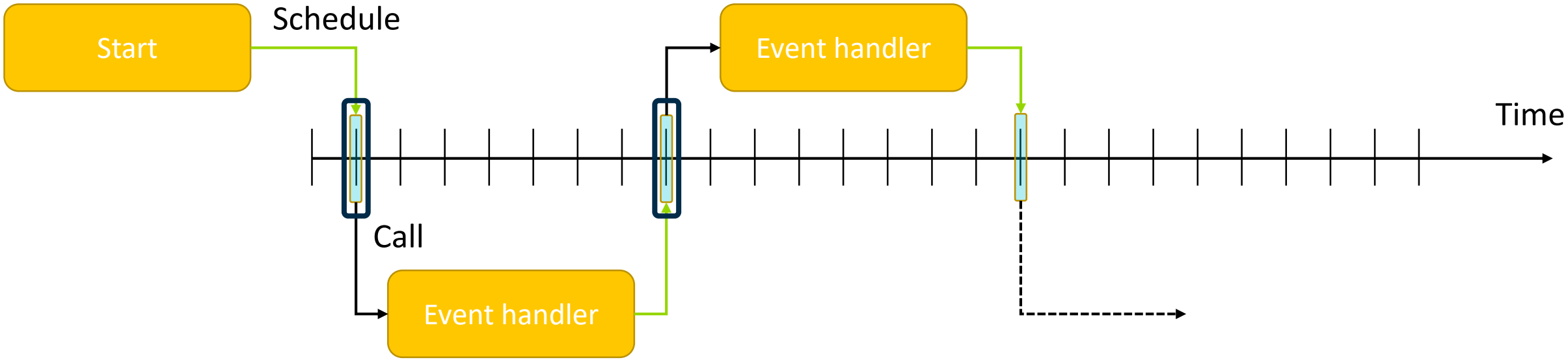
arm

Configuring and running gem5



arm

Simulating Time



Discrete: Handles time in discrete steps (ticks)

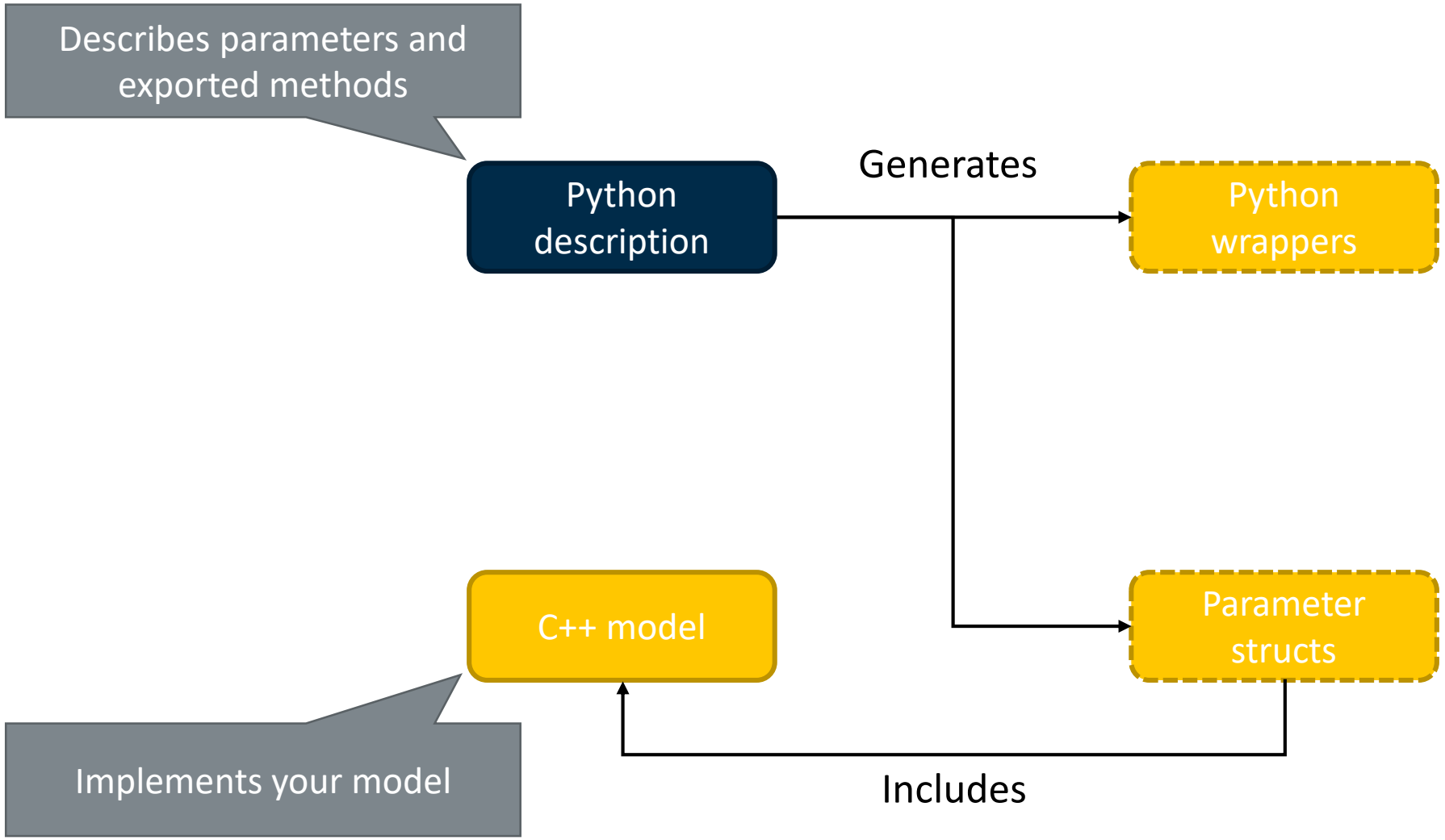
- Usually 1THz in gem5

Simulator skips to the next event on the timeline

- More efficient than traditional clocked simulators

arm

How are models implemented



arm

Examples

Configuration & running:

- Syscall emulation: `configs/learning_gem5/part1`
- Full-system: `configs/example/arm/{fs_bigLittle.py, devices.py}`

Simple memory-mapped IO devices: `IsaFake`

- See: `src/dev/isa_fake.{cc,hh}` and `src/dev/Device.py`
- Simple PCI devices with interrupts: `PciVirtIO`
- See: `src/dev/virtio/pci.{cc,hh}` and `src/dev/VirtIO.py`

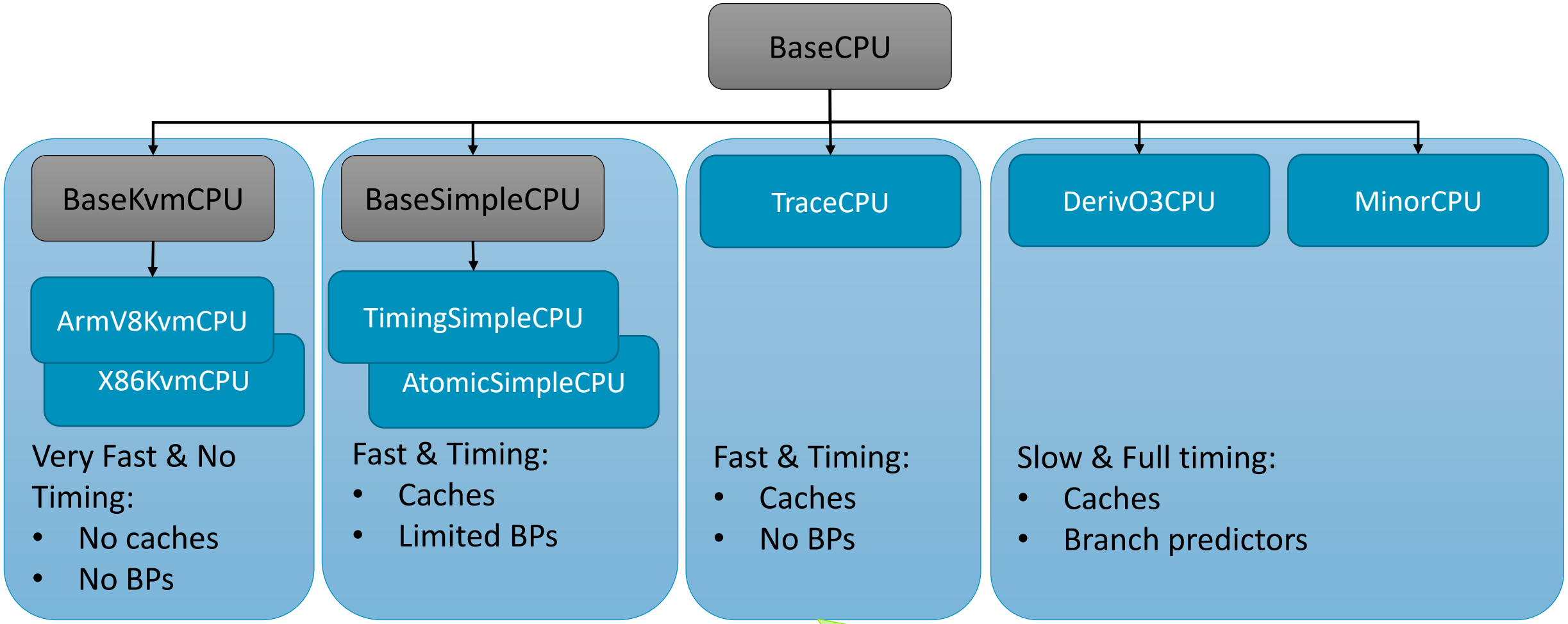
More complex PCI device with DMA: `CopyEngine`

- See: `src/dev/pci/copy_engine.{cc,hh}` and `src/dev/pci/CopyEngine.py`

CPU models

arm

CPU models overview



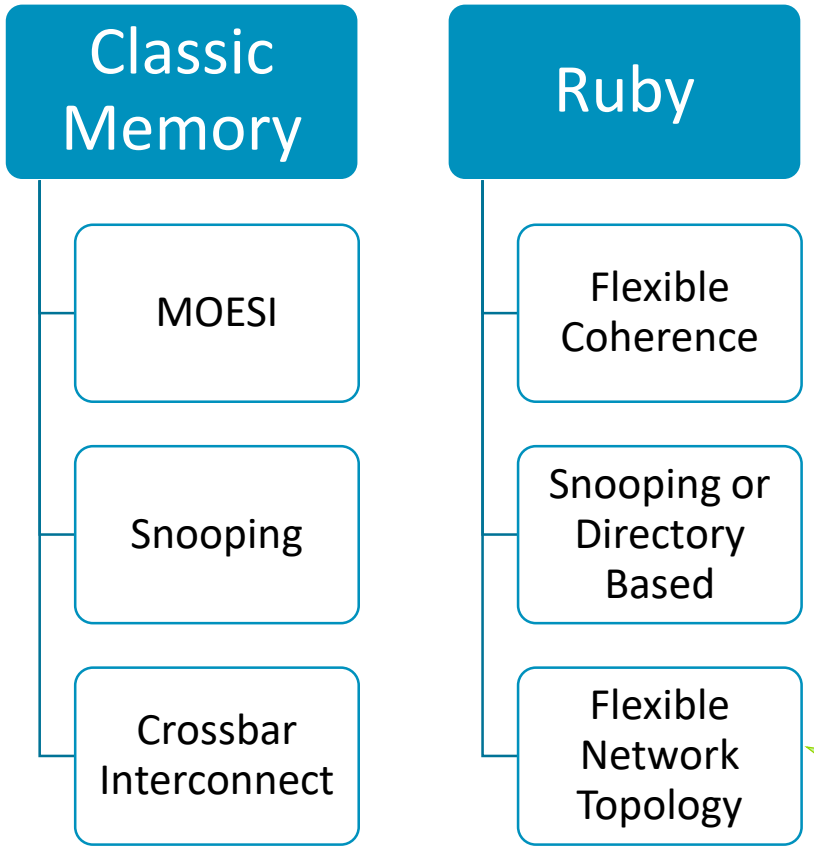
14:15 - Trace-driven simulation of multithreaded applications in gem5
14:45 - Generating Synthetic Traffic for Heterogeneous Architectures

arm

Memory systems

arm

On-chip memory system

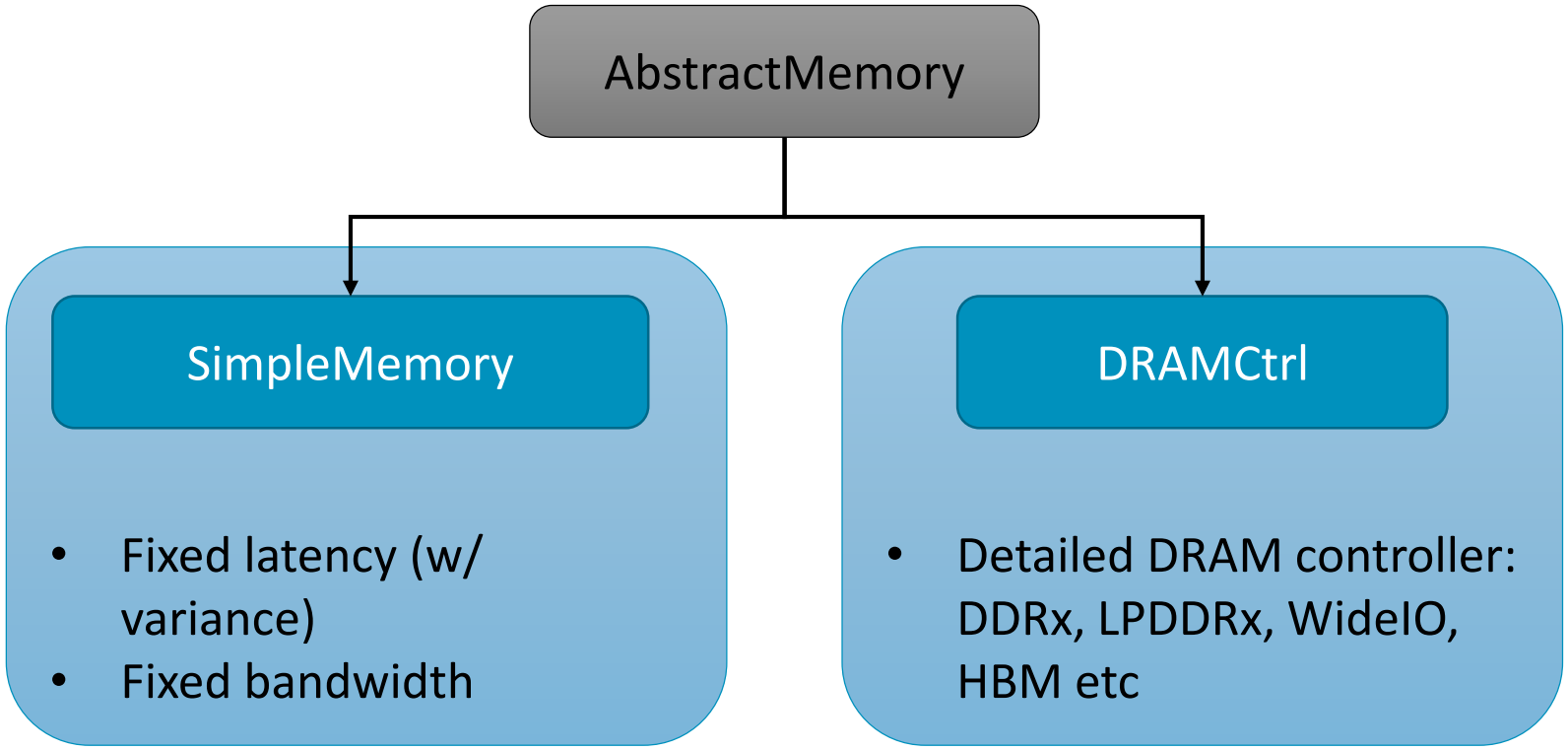


09:45 - Learning gem5: Modeling Cache Coherence with gem5

11:15 - A Detailed On-Chip Network Model inside a Full-System Simulator

arm

Off-chip memory system



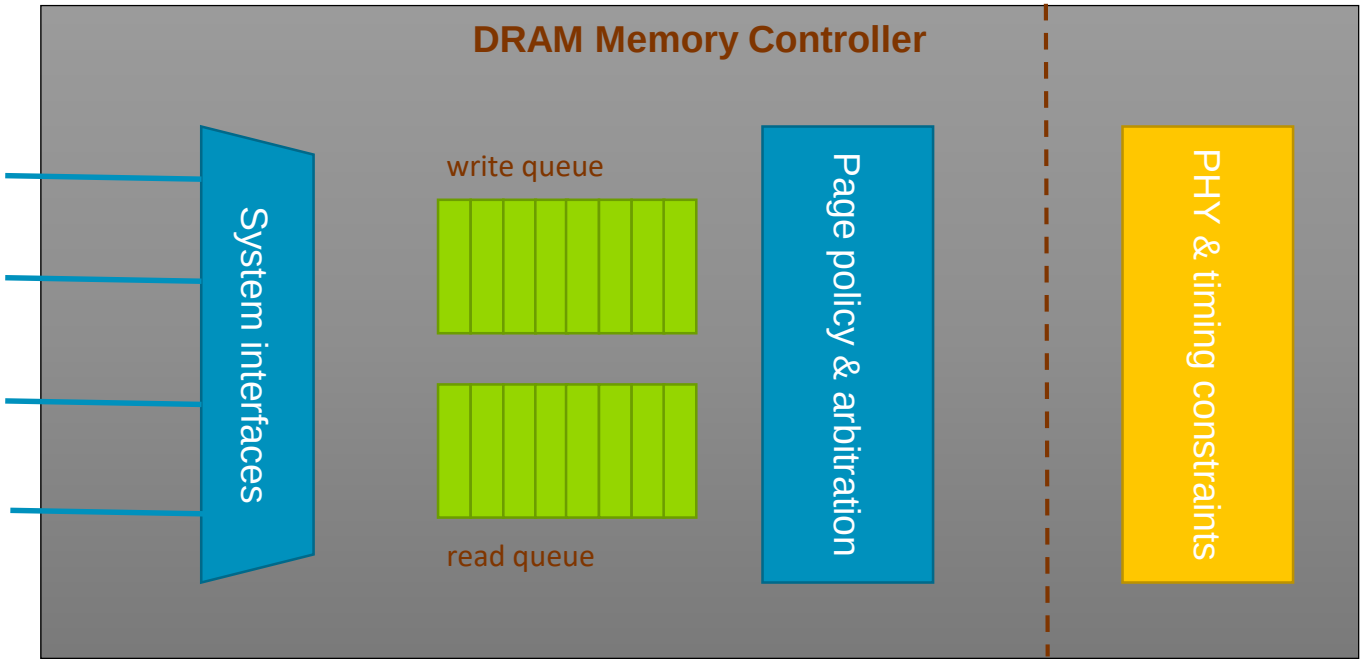
arm

Top-down DRAM controller model

Don’t model the actual DRAM, only the timing constraints

- DDR3/4, LPDDR2/3/4, WIO1/2, GDDR5, HBM, HMC, even PCM
- See *src/mem/DRAMCtrl.py* and *src/mem/dram_ctrl.{hh, cc}*

11:45 - Integrating and quantifying the impact of low power modes in the DRAM controller in gem5



Device width
Burst length
#ranks, #banks
Page size

tRCD
tCL
tRP
tRAS
tBURST
tRFC & tRFEI
tWTR
tRRD
tFAW/tTAW
…

Hansson et al, *Simulating DRAM controllers for future system architecture exploration,* ISPASS’14

arm

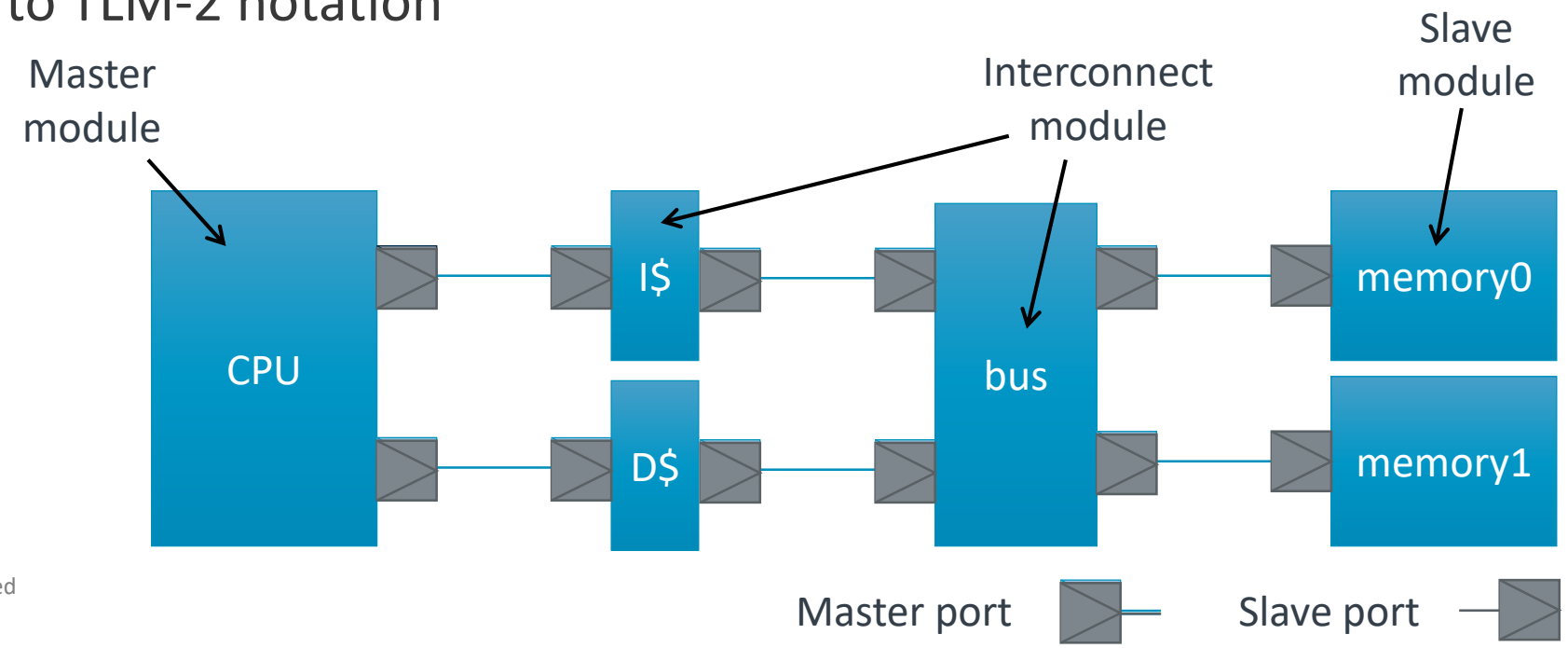
Ports: Connecting memory objects

MemObjects are connected through master and slave ports

A master module has at least one master port, a slave module at least one slave port, and an interconnect module at least one of each

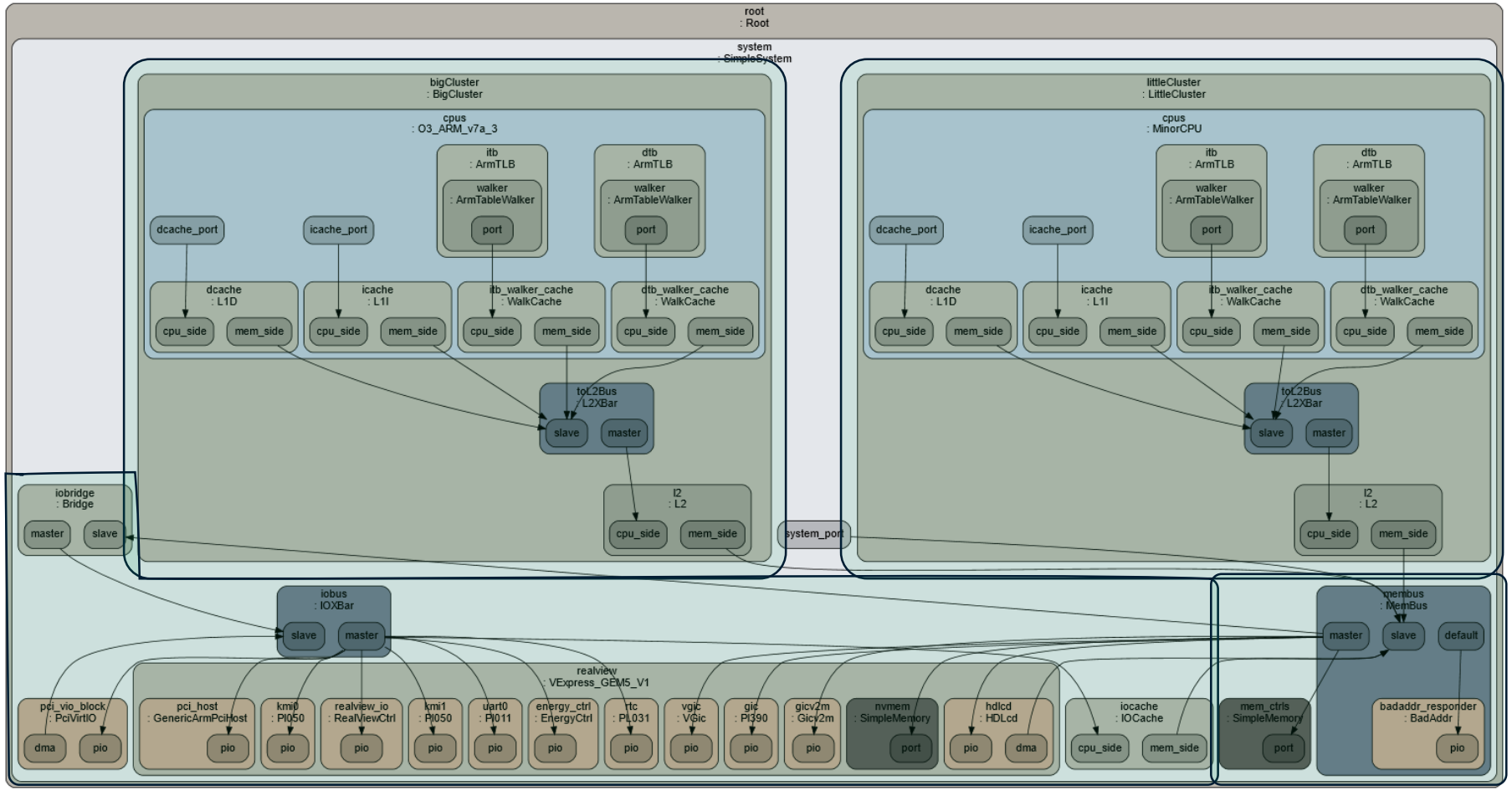
- A master port always connects to a slave port
- Similar to TLM-2 notation

15:15 - System Simulation with gem5, SystemC and other Tools



arm

System Overview



arm

Thank You!
Danke!
Merci!
谢谢!
ありがとう!
Gracias!
Kiitos!
감사합니다
धन्यवाद

Workshop schedule

09.30	Interacting with gem5 using workload-automation & devlib
09.45	ARM Research Starter Kit: System Modeling using gem5
10.00	Break
10.15	Debugging a target-agnostic JIT compiler with gem5
10.30	Learning gem5: Modeling Cache Coherence with gem5
11.00	**Break**
11.15	A Detailed On-Chip Network Model inside a Full-System Simulator
11.45	Integrating and quantifying the impact of low power modes in the DRAM controller in gem5
12.00	Break
12.15	CPU power estimation using PMCs and its application in gem5
12.45	gem5: empowering the masses

13.00	**Lunch**
14.15	Trace-driven simulation of multithreaded applications in gem5
14.45	Generating Synthetic Traffic for Heterogeneous Architectures
15.00	Break
15.15	System Simulation with gem5, SystemC and other Tools
15.30	COSSIM: An Integrated Solution to Address the Simulator Gap for Parallel Heterogeneous Systems
15.45	Simulation of Complex Systems Incorporating Hardware Accelerators
16.15	**Break**
16.30	**Introduction to ARM Research**
18.20	**Poster session**
20.00	**Dinner**

arm