gem5 - an overview

Syam Sankar
Ph.D Scholar (Prime Minister's Research Fellow)
MARS Lab, Dept. of CSE, IIT Guwahati

Architectural Simulators

Why researchers need simulators?

- To prototype and evaluate ideas

- To collaborate with people
  - in industry and academia

- Which simulator to use?
  - code quality
  - modularity
  - flexibility

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.”

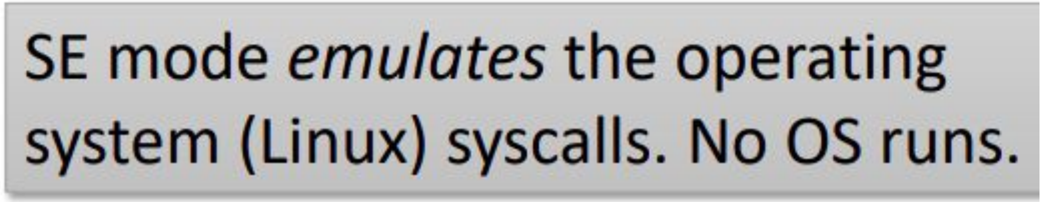
gem5 Simulator

- Flexible
  - diverse set of CPU models
  - system execution models
  - memory system models

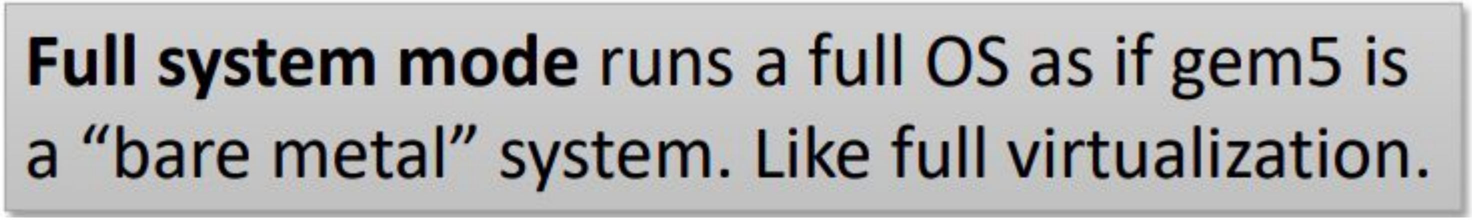
- Modular
  - object oriented design
  - everything represented as objects

    (CPU, caches,buses,devices etc)
- Event driven
  - objects schedules their own events

- **gem5** is written primarily in **C++** and python.
- It can simulate a complete system with devices and an operating system in **full system mode (FS mode)**, or user space only programs where system services are provided directly by the simulator in **syscall emulation mode (SE mode)**

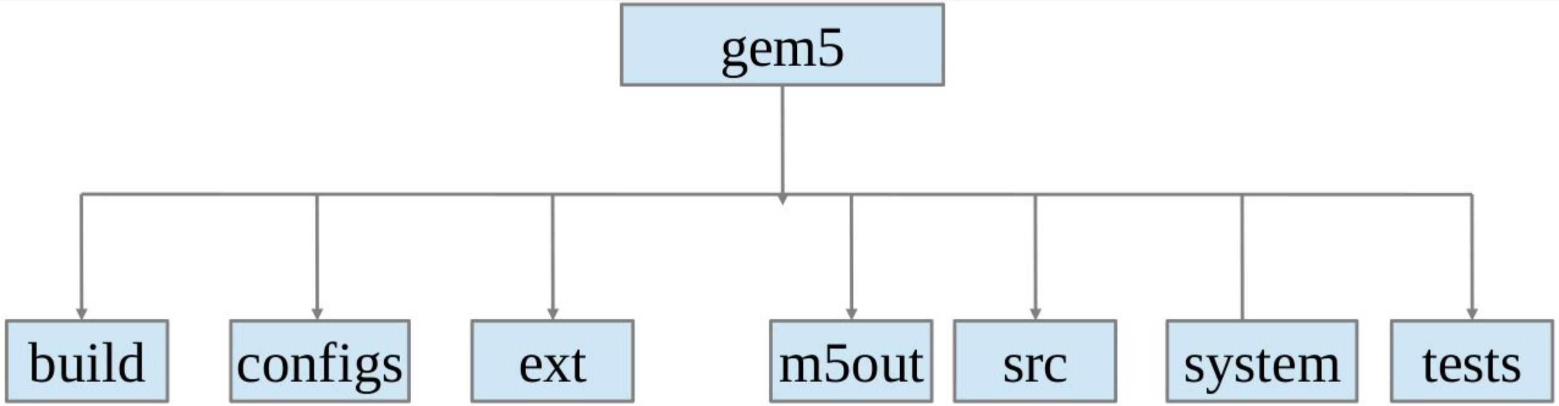


SE mode *emulates* the operating system (Linux) syscalls. No OS runs.



Full system mode runs a full OS as if gem5 is a "bare metal" system. Like full virtualization.

gem5 Design



build - ISA which is being created

configs - simulation configuration scripts (in python)

ext - external tools gem5 supports

src - gem5 source code

system - bootloaders for use in simulated systems

tests - files used for testing (examples)

m5out– simulation statistics

gem5 ISAs

src/arch/
- alpha
- arm
- hsail
- mips
- power
- riscv
- sparc
- x86

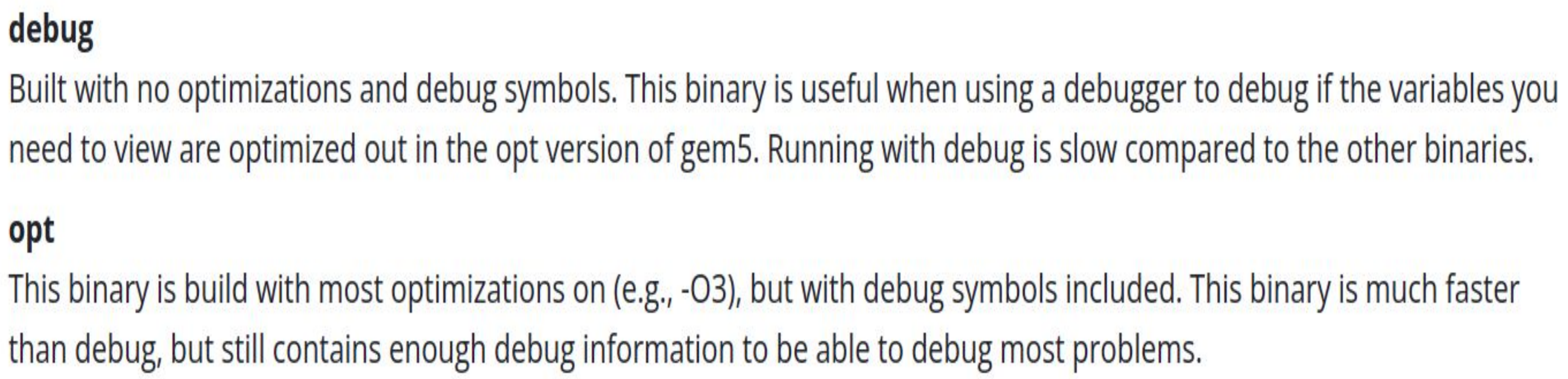
Not all equally well supported. **ARM, X86** most used/tested.

Each directory contains devices, ISA-specific objects, system interface, **ISA definition**

gem5 binary types

The SCons scripts in gem5 currently have 5 different binaries you can build for gem5:

debug, opt, fast, prof, and perf



debug

Built with no optimizations and debug symbols. This binary is useful when using a debugger to debug if the variables you need to view are optimized out in the opt version of gem5. Running with debug is slow compared to the other binaries.

opt

This binary is build with most optimizations on (e.g., -O3), but with debug symbols included. This binary is much faster than debug, but still contains enough debug information to be able to debug most problems.

CPU models

- AtomicSimple
- TimingSimple
- InOrder
- O3

Detailed CPU Models

- **In-Order**
- Detailed in-order CPU
- Default 5-stage pipeline
  - F, D, E, M, W
- Configured to model
  - different pipeline stages
  - Issue width

- **Out-of-Order**
- Detailed out-of-order CPU
- Simulate super scalar architectures
- Default 7-stage pipeline
  - Fetch, Decode, Rename, Issue, Execute, Writeback and Commit

Memory models

- Classic
- Ruby

Memory Models in gem5

Classic Memory Model

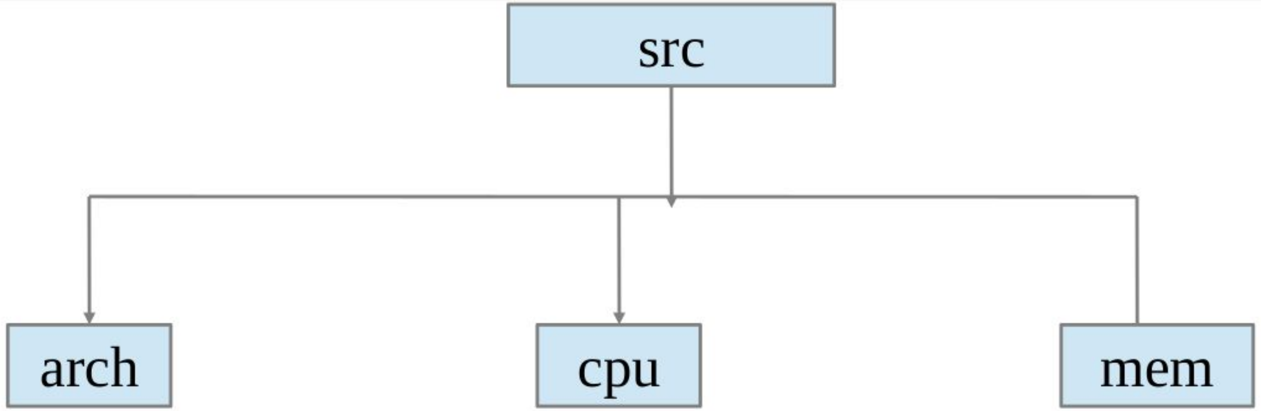
- inherited from the M5 simulator
- fast, flexible and easily configurable memory system.

Memory Models in gem5

Ruby Memory Model

- inherited from the GEMS simulator
- Can simulate variety of memory systems

gem5 Design



arch : ISA supported by gem5

alpha, x86, riscv, sparc, power, mips etc.

cpu : cpu models supported by gem5

minor, o3, simple

mem : memory models

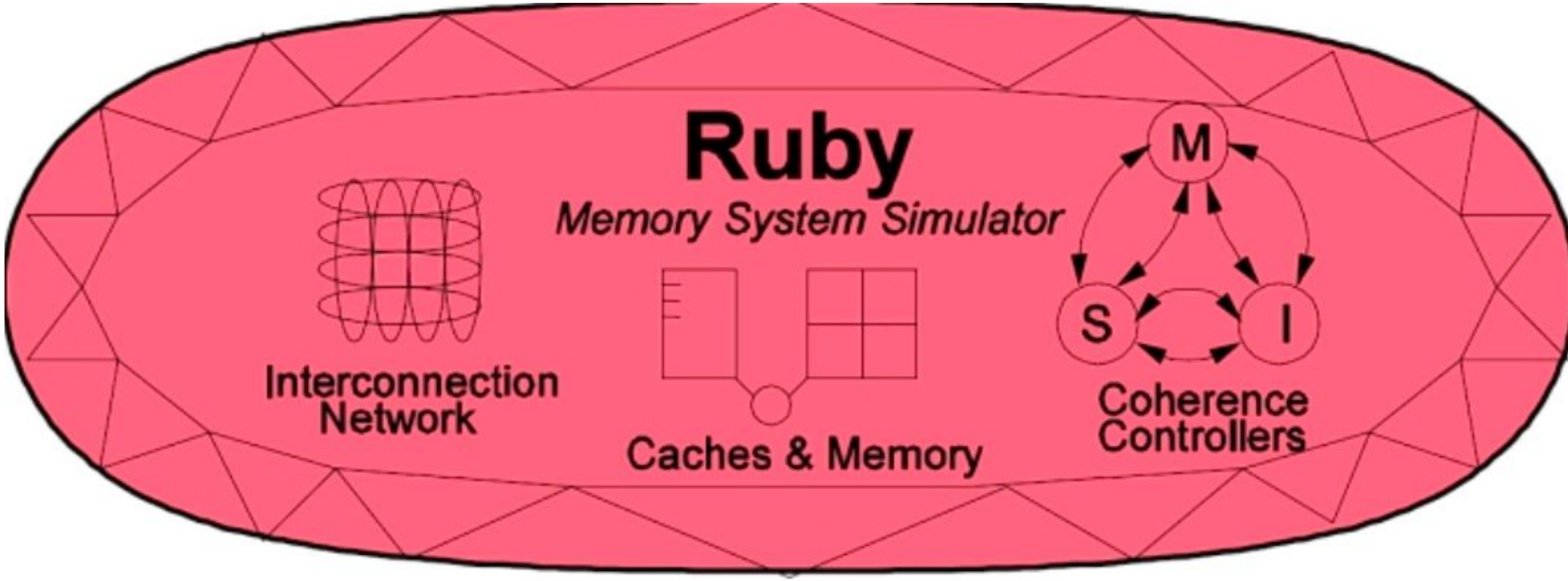
Cache

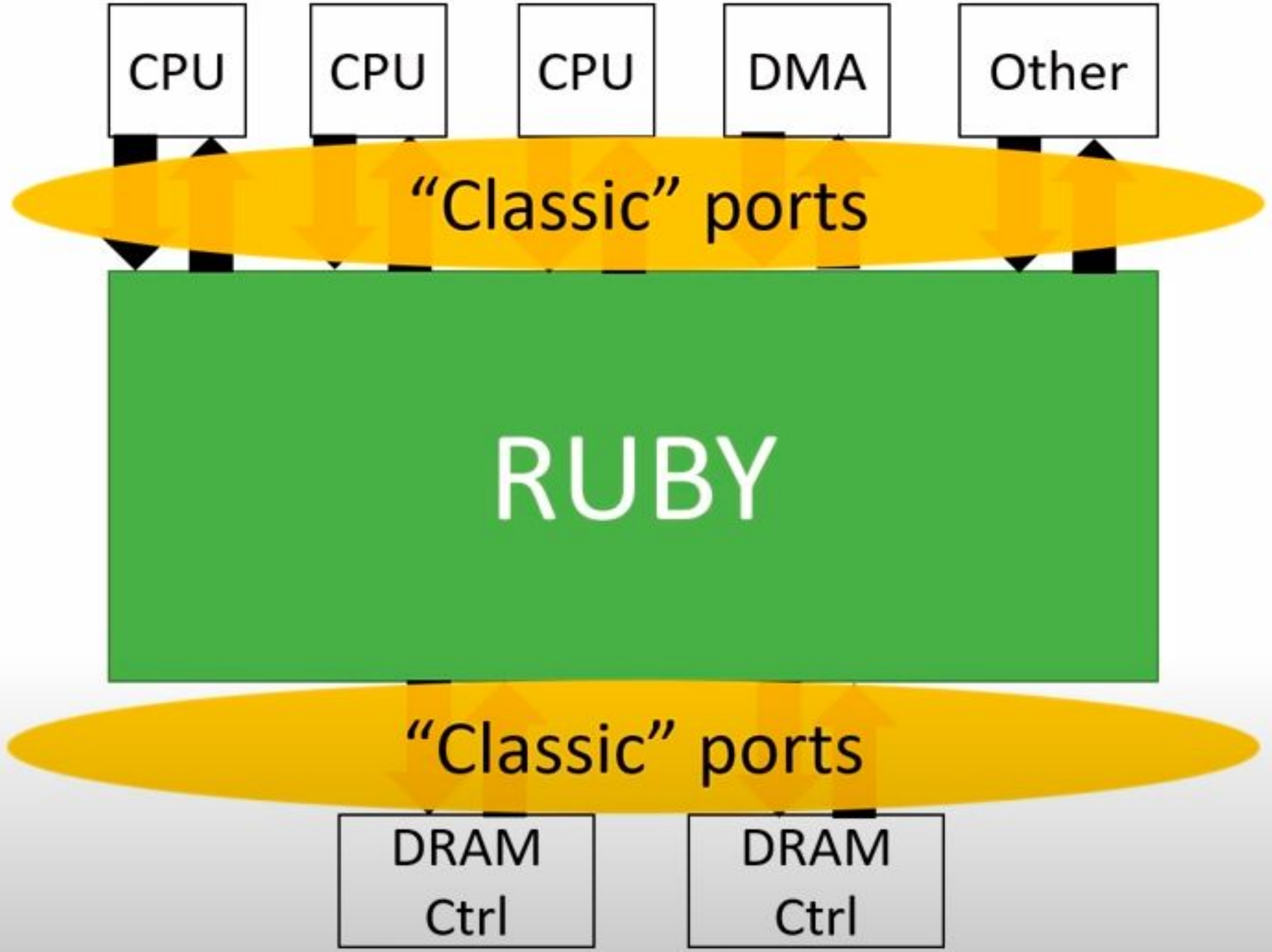
Ruby - garnet

Ruby

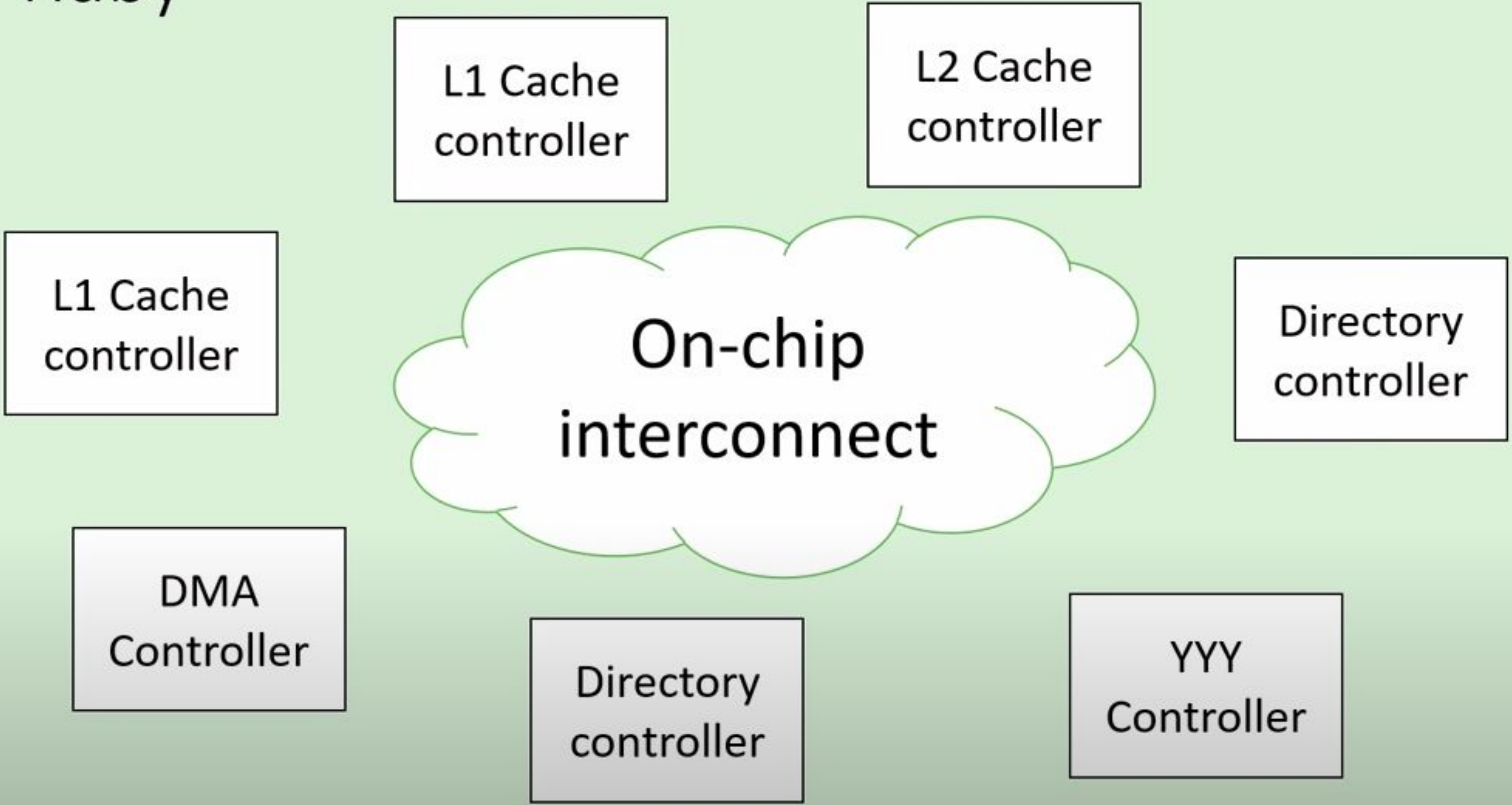
- Ruby implements a detailed simulation model for the memory subsystem.

- It models inclusive/exclusive cache hierarchies with various replacement policies, coherence protocol implementations, interconnection networks, DMA and memory controllers, various *sequencers* that initiate memory requests and handle responses.


Ruby
Memory System Simulator
Interconnection Network
Caches & Memory
M
S
I
Coherence Controllers


CPU
CPU
CPU
DMA
Other
“Classic” ports
RUBY
“Classic” ports
DRAM Ctrl
DRAM Ctrl

Ruby

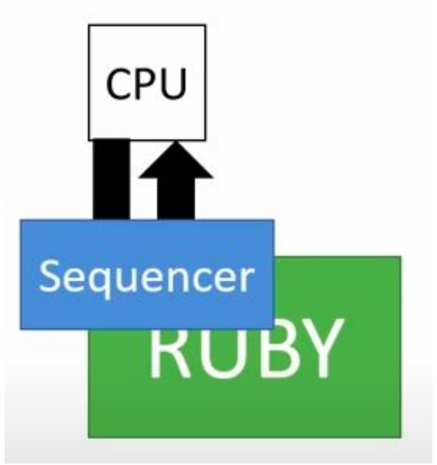

L1 Cache controller
L2 Cache controller
L1 Cache controller
On-chip interconnect
Directory controller
DMA Controller
Directory controller
YYY Controller

Ruby- Sequencer

Two names, same thing: RubyPort and Sequencer

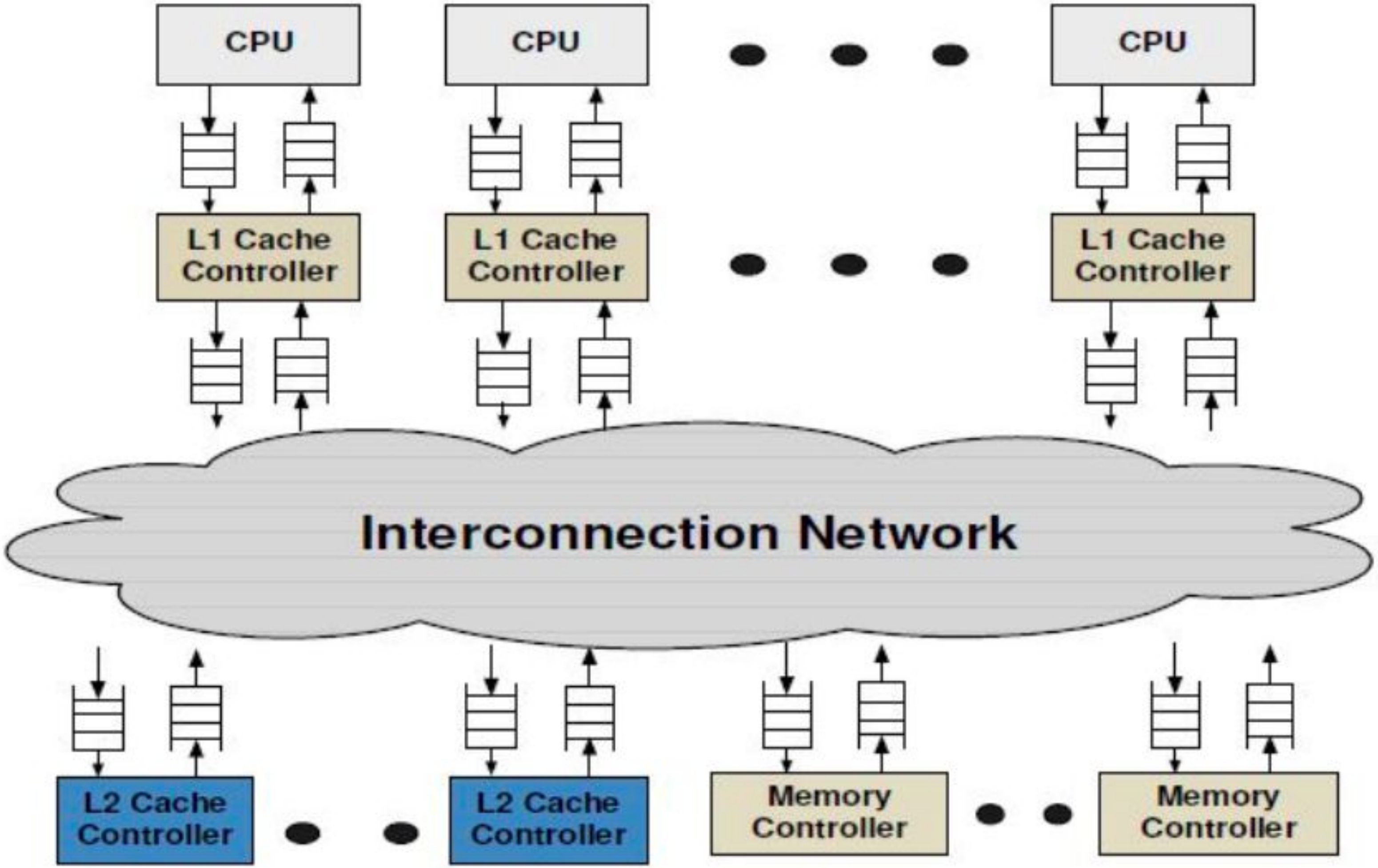
Sequencer is a MemObject (classic ports)

Converts gem5 packets to RubyRequests



Interconnection network

- Simple
- **Garnet**


CPU
CPU
CPU
L1 Cache Controller
L1 Cache Controller
L1 Cache Controller
Interconnection Network
L2 Cache Controller
L2 Cache Controller
Memory Controller
Memory Controller

Building gem5

[only refer this]

1. https://www.gem5.org/documentation/learning_gem5/part1/building/

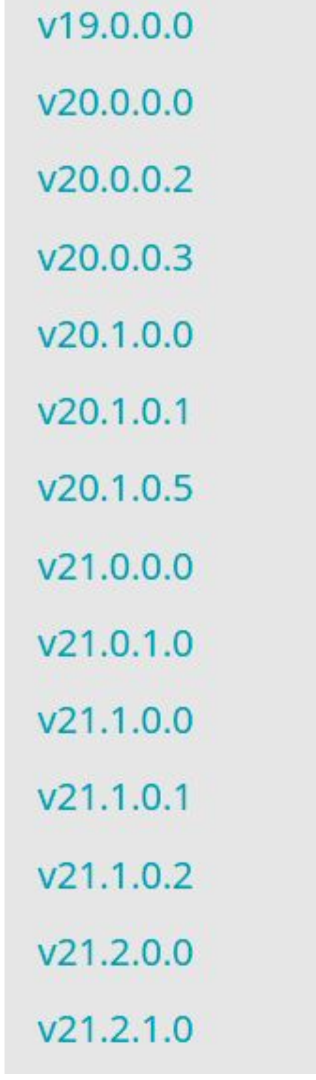
2. https://www.gem5.org/documentation/general_docs/building#dependencies

Building gem5

- gem5 has been designed with a Linux environment in mind.

  Ubuntu 18.04 and Ubuntu 20.04

- As of gem5 21.0, they support building and running gem5 with **Python 3.6+** only.
- gem5 20.0 was last version of gem5 to provide support for Python 2.

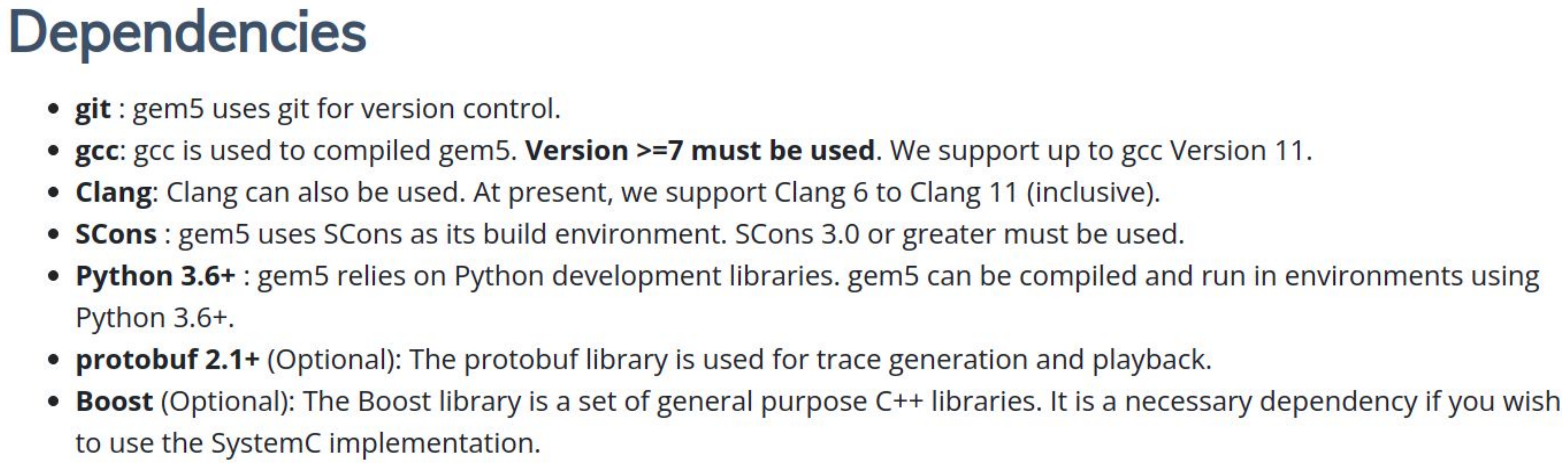


v19.0.0.0
v20.0.0.0
v20.0.0.2
v20.0.0.3
v20.1.0.0
v20.1.0.1
v20.1.0.5
v21.0.0.0
v21.0.1.0
v21.1.0.0
v21.1.0.1
v21.1.0.2
v21.2.0.0
v21.2.1.0

- To build gem5, we will use SCons [software construction tool]
- gem5's build system is based on SCons, an open source build system implemented in Python.
- SCons uses the SConstruct file (`gem5/SConstruct`) to set up a number of variables and then uses the SConscript file in every subdirectory to find and compile all of the gem5 source.

- SCons automatically creates a `gem5/build` directory when first executed.
- In this directory you'll find the files generated by SCons, the compiler, etc. There will be a separate directory for each set of options (ISA and cache coherence protocol) that you use to compile gem5.

Building gem5



Dependencies

- **git** : gem5 uses git for version control.
- **gcc**: gcc is used to compiled gem5. **Version >=7 must be used**. We support up to gcc Version 11.
- **Clang**: Clang can also be used. At present, we support Clang 6 to Clang 11 (inclusive).
- **SCons** : gem5 uses SCons as its build environment. SCons 3.0 or greater must be used.
- **Python 3.6+** : gem5 relies on Python development libraries. gem5 can be compiled and run in environments using Python 3.6+.
- **protobuf 2.1+** (Optional): The protobuf library is used for trace generation and playback.
- **Boost** (Optional): The Boost library is a set of general purpose C++ libraries. It is a necessary dependency if you wish to use the SystemC implementation.

Building gem5: Getting the code

```
git clone https://gem5.googlesource.com/public/gem5
```

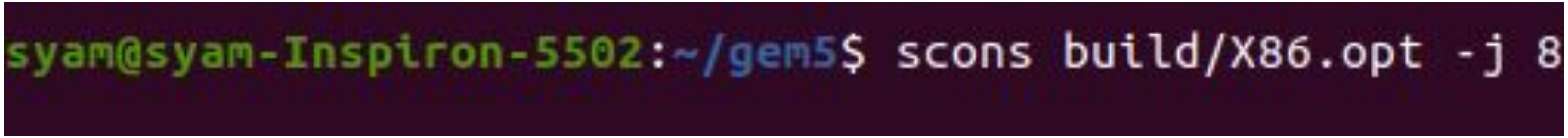
Within the root of the gem5 directory, gem5 can be built with SCons using:

```
scons build/{ISA}/gem5.{variant} -j {cpus}
```

where `{ISA}` is the target (guest) Instruction Set Architecture, and `{variant}` specifies the compilation settings. For most intents and purposes `opt` is a good target for compilation. The `-j` flag is optional and allows for parallelization of compilation with `{cpus}` specifying the number of threads. A single-threaded compilation from scratch can take up to 2 hours on some systems. We therefore strongly advise allocating more threads if possible.

For example, to build gem5 on 8 threads with `opt` and targeting x86:

```
scons build/X86/gem5.opt -j 8
```



```
syam@syam-Inspiron-5502:~/gem5$ scons build/X86.opt -j 8
```

Python in a non-default location

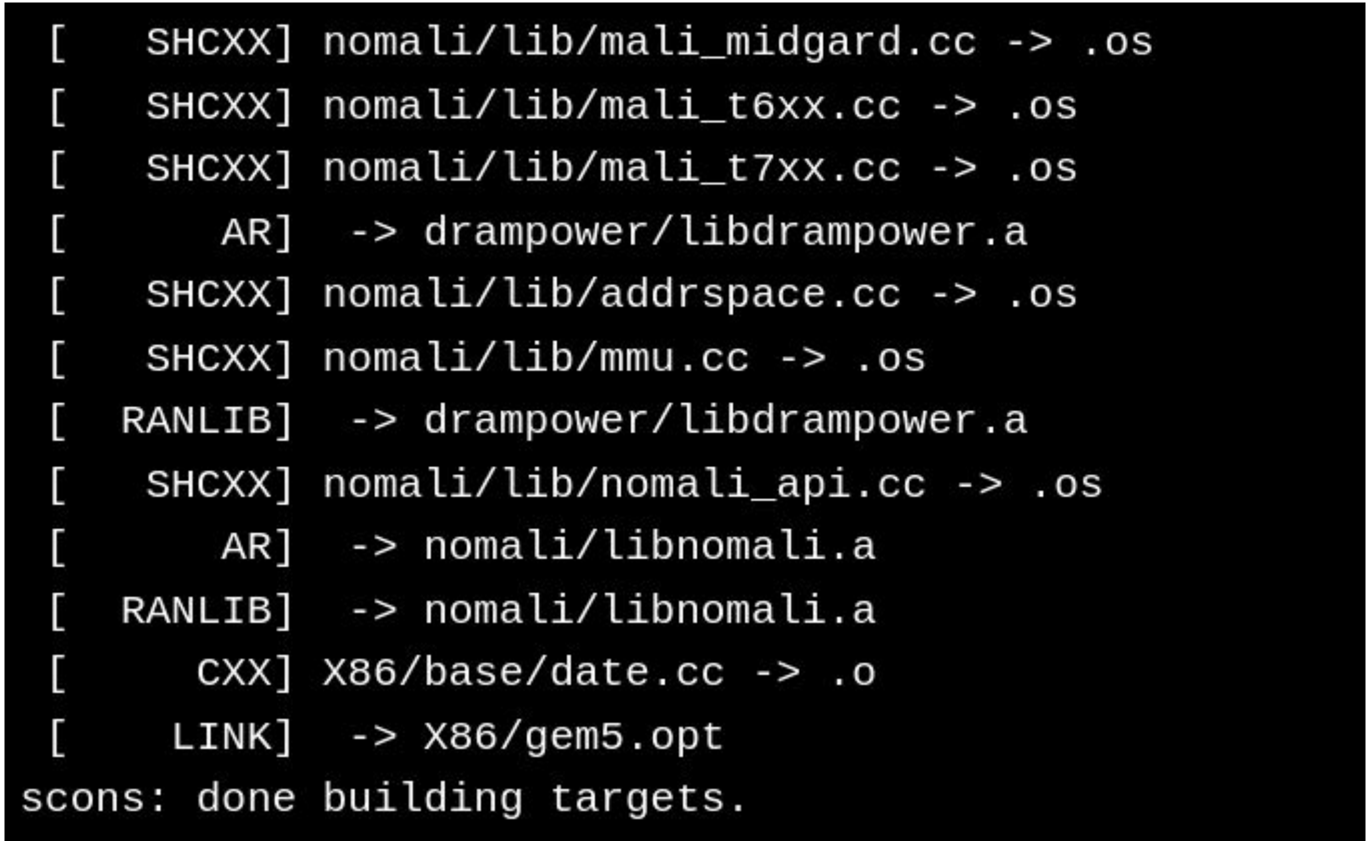
- If you use a non-default version of Python, (e.g., version 3.8 when 2.7 is your default), there may be problems when using SCons to build gem5.

- To fix this, you can force SCons to use your environment's Python version by running `` python3 `which scons` build/X86/gem5.opt `` instead of `scons build/X86/gem5.opt`.

```
python3 `which scons` build/X86/gem5.opt -j 8
```

Building target ISA successfully !

Output:



```
 [    SHCXX] nomali/lib/mali_midgard.cc -> .os
 [    SHCXX] nomali/lib/mali_t6xx.cc -> .os
 [    SHCXX] nomali/lib/mali_t7xx.cc -> .os
 [       AR]  -> drampower/libdrampower.a
 [    SHCXX] nomali/lib/addrspace.cc -> .os
 [    SHCXX] nomali/lib/mmu.cc -> .os
 [   RANLIB]  -> drampower/libdrampower.a
 [    SHCXX] nomali/lib/nomali_api.cc -> .os
 [       AR]  -> nomali/libnomali.a
 [   RANLIB]  -> nomali/libnomali.a
 [      CXX] X86/base/date.cc -> .o
 [     LINK]  -> X86/gem5.opt
scons: done building targets.
```

Your feelings now!

Running gem5

- The gem5 binary takes, as a parameter, a python script which sets up and executes the simulation.

- In this script, you create a system to simulate, create all of the components of the system, and specify all of the parameters for the system components. Then, from the script, you can begin the simulation.

- This script is completely user-defined. You can choose to use any valid Python code in the configuration scripts.

- There are a number of example configuration scripts that ship with gem5 in `configs/learning_gem5` or `configs/examples`.

Let’s start a simple script (for ex: say simple.py - python script defines the system to model) that can run gem5

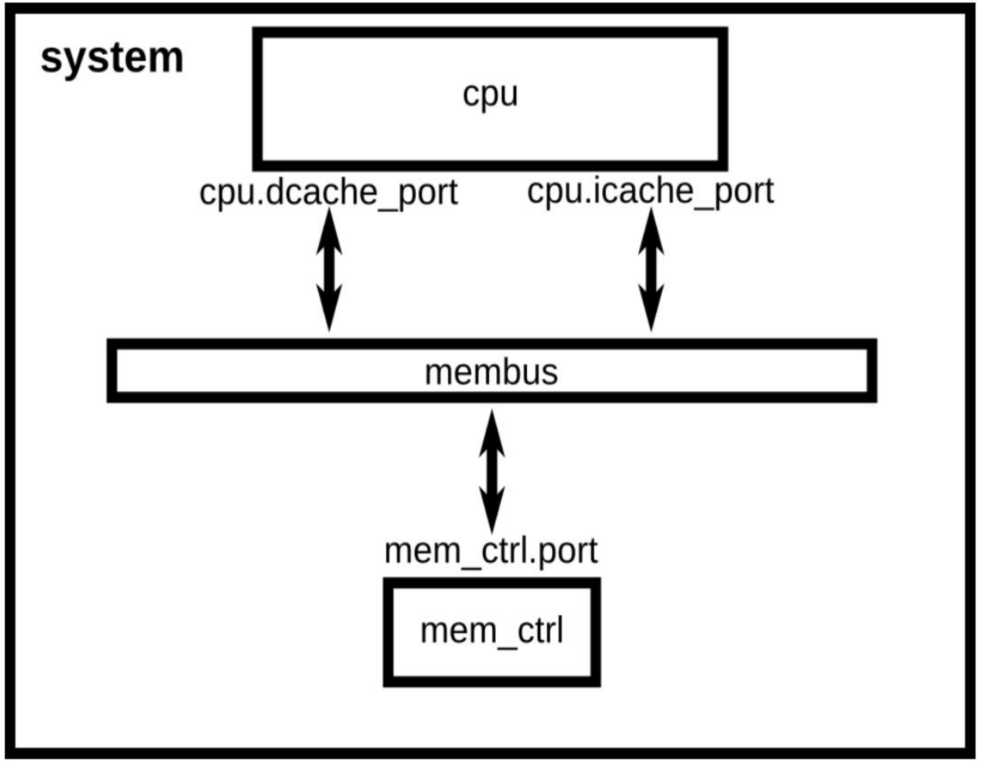
Consider a simple system

that we want to simulate:-

Single CPU connected to a memory bus

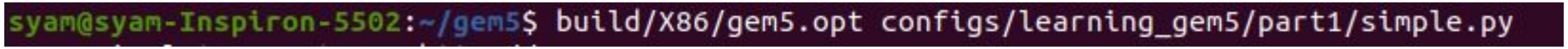
Ref:

https://www.youtube.com/watch?v=fD3hhNnfL6k

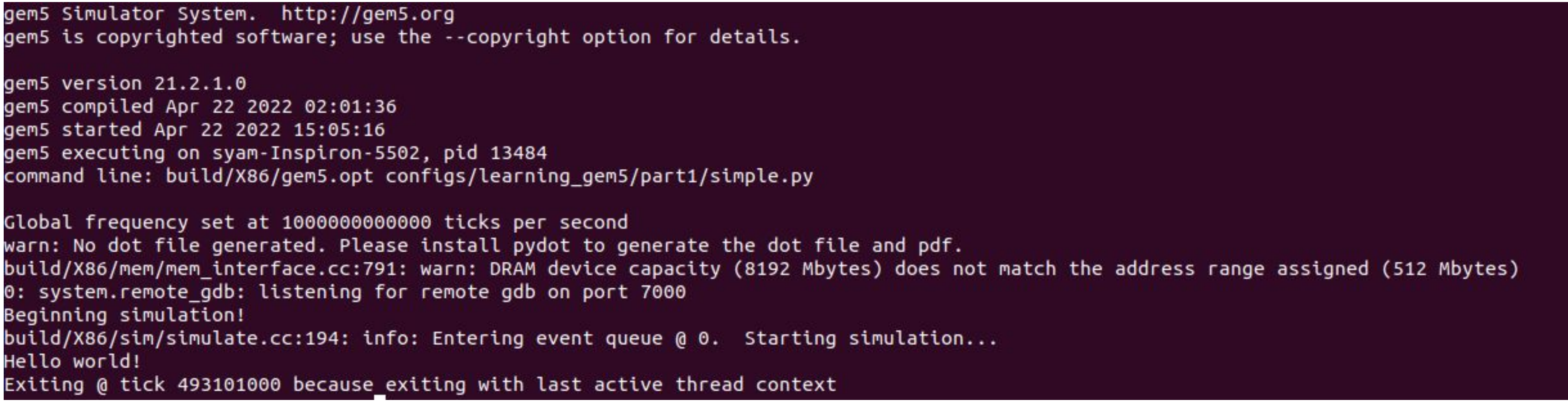


Run

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



```
syam@syam-Inspiron-5502:~/gem5$ build/X86/gem5.opt configs/learning_gem5/part1/simple.py
```

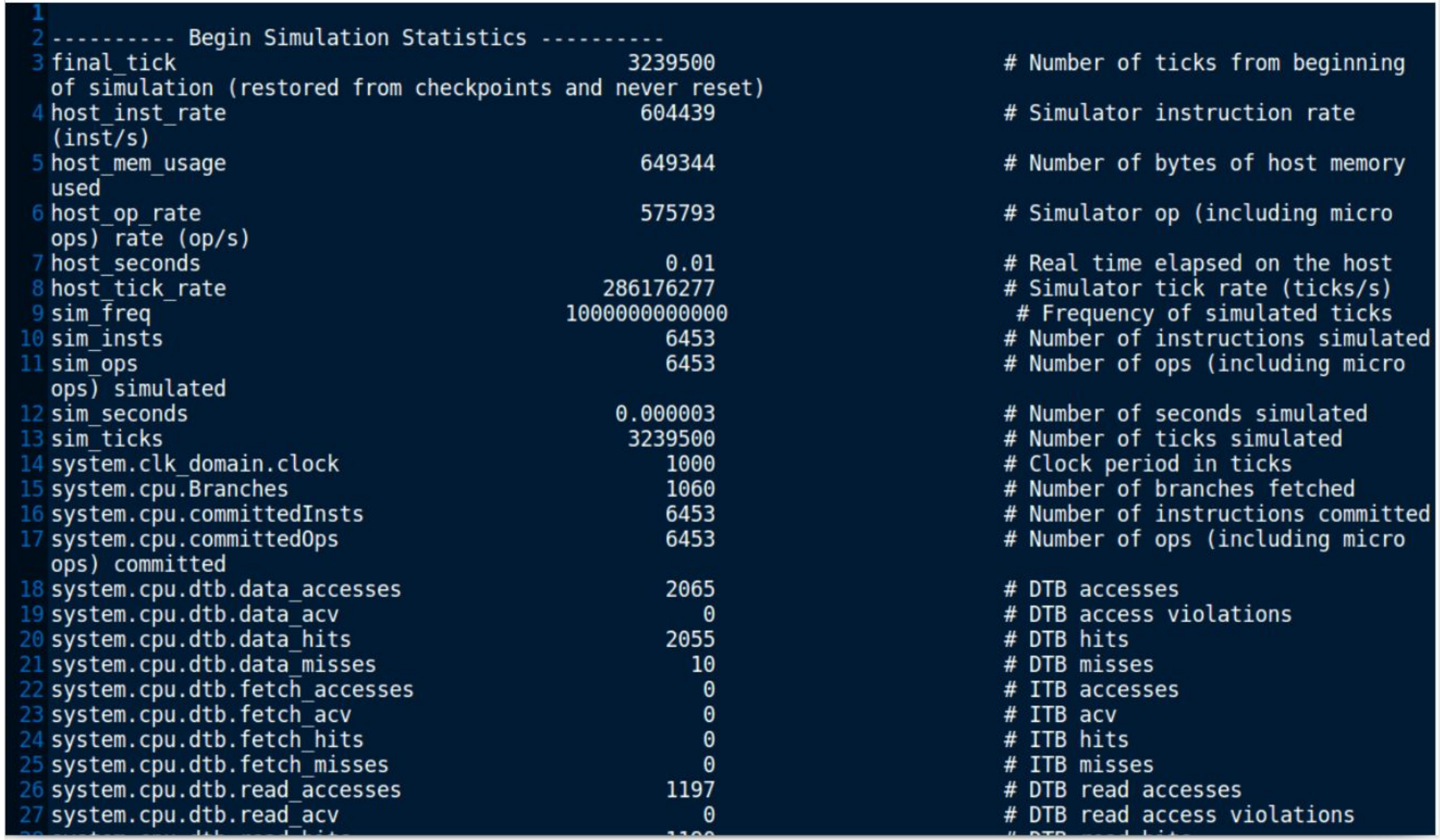


```
gem5 Simulator System.  http://gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 21.2.1.0
gem5 compiled Apr 22 2022 02:01:36
gem5 started Apr 22 2022 15:05:16
gem5 executing on syam-Inspiron-5502, pid 13484
command line: build/X86/gem5.opt configs/learning_gem5/part1/simple.py

Global frequency set at 1000000000000 ticks per second
warn: No dot file generated. Please install pydot to generate the dot file and pdf.
build/X86/mem/mem_interface.cc:791: warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
0: system.remote_gdb: listening for remote gdb on port 7000
Beginning simulation!
build/X86/sim/simulate.cc:194: info: Entering event queue @ 0.  Starting simulation...
Hello world!
Exiting @ tick 493101000 because exiting with last active thread context
```

Statistics (gem5/m5out)



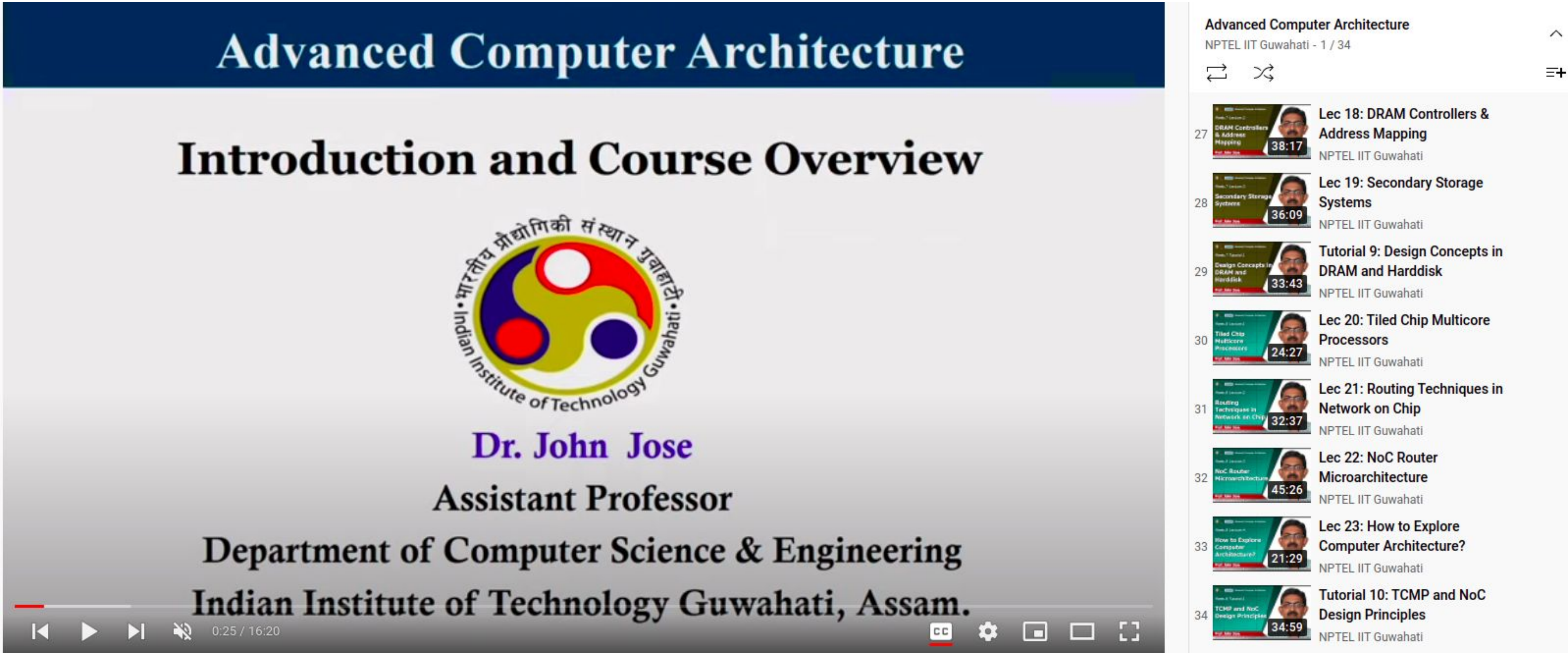
```

---------- Begin Simulation Statistics ----------
final_tick                                  3239500                       # Number of ticks from beginning
of simulation (restored from checkpoints and never reset)
host_inst_rate                               604439                       # Simulator instruction rate
(inst/s)
host_mem_usage                               649344                       # Number of bytes of host memory
used
host_op_rate                                 575793                       # Simulator op (including micro
ops) rate (op/s)
host_seconds                                   0.01                       # Real time elapsed on the host
host_tick_rate                            286176277                       # Simulator tick rate (ticks/s)
sim_freq                               1000000000000                       # Frequency of simulated ticks
sim_insts                                      6453                       # Number of instructions simulated
sim_ops                                        6453                       # Number of ops (including micro
ops) simulated
sim_seconds                                0.000003                       # Number of seconds simulated
sim_ticks                                   3239500                       # Number of ticks simulated
system.clk_domain.clock                        1000                       # Clock period in ticks
system.cpu.Branches                            1060                       # Number of branches fetched
system.cpu.committedInsts                      6453                       # Number of instructions committed
system.cpu.committedOps                        6453                       # Number of ops (including micro
ops) committed
system.cpu.dtb.data_accesses                   2065                       # DTB accesses
system.cpu.dtb.data_acv                           0                       # DTB access violations
system.cpu.dtb.data_hits                       2055                       # DTB hits
system.cpu.dtb.data_misses                       10                       # DTB misses
system.cpu.dtb.fetch_accesses                     0                       # ITB accesses
system.cpu.dtb.fetch_acv                          0                       # ITB acv
system.cpu.dtb.fetch_hits                         0                       # ITB hits
system.cpu.dtb.fetch_misses                       0                       # ITB misses
system.cpu.dtb.read_accesses                   1197                       # DTB read accesses
system.cpu.dtb.read_acv                           0                       # DTB read access violations
```

Gem5 Learning- NPTEL

1. Gem5 -Overview - https://youtu.be/9_RBQXtExrw
2. gem5 Simulator - Cache Optimisation- https://youtu.be/-riAWSuNwvU
3. gem5 Simulator - NoC Optimisation - https://youtu.be/tdFFsnuJWn0

Network on Chip - NPTEL [Lect: 20-23]



Advanced Computer Architecture

Introduction and Course Overview

Indian Institute of Technology Guwahati

Dr. John Jose

Assistant Professor

Department of Computer Science & Engineering

Indian Institute of Technology Guwahati, Assam.

0:25 / 16:20

Advanced Computer Architecture

NPTEL IIT Guwahati - 1 / 34

27 Lec 18: DRAM Controllers & Address Mapping 38:17 NPTEL IIT Guwahati

28 Lec 19: Secondary Storage Systems 36:09 NPTEL IIT Guwahati

29 Tutorial 9: Design Concepts in DRAM and Harddisk 33:43 NPTEL IIT Guwahati

30 Lec 20: Tiled Chip Multicore Processors 24:27 NPTEL IIT Guwahati

31 Lec 21: Routing Techniques in Network on Chip 32:37 NPTEL IIT Guwahati

32 Lec 22: NoC Router Microarchitecture 45:26 NPTEL IIT Guwahati

33 Lec 23: How to Explore Computer Architecture? 21:29 NPTEL IIT Guwahati

34 Tutorial 10: TCMP and NoC Design Principles 34:59 NPTEL IIT Guwahati

Ref:

https://www.gem5.org/documentation/

Happy Learning…!

Thank you