

# gem5 tutorial

## References:

- <https://github.com/gem5bootcamp>



# Environment

The steps outlined in this tutorial have been tested on the computers located in **Delta Building, Room 219**.

These computers are equipped with:

- CPU: 13th Gen Intel(R) Core(TM) i5-13500
- RAM: 16 GB
- SSD: 954 GB

**Docker Desktop, WSL, Hyper-V, KVM** are pre-installed.

# Installation on Your Own Computer

You may also install Docker on your own computer. Please ensure you have at least **10 GB** of disk space available.

This instructions for **macOS** is provided.



# Install Docker Desktop on macOS

For macOS users:

1. Download [the installer](#).
2. Double-click `Docker.dmg` to open the installer, then drag the Docker icon to the **Applications** folder.
3. Double-click `Docker.app` in the **Applications** folder to start Docker.
4. The Docker menu will display the Docker Subscription Service Agreement. Select **Accept** to continue.
5. In the installation window, select **Use recommended settings (Requires password)**.
6. Select **Finish**.

## Back up files from `$HOME\workspaces\` to a flash drive

Delta 219 computers are configured to automatically revert to a previously saved system state (a restore point) without prior notice.

- Your files may be permanently deleted without warning at any time (unannounced data loss).
- You cannot predict when these reverts will occur, nor can you prevent them from happening (limited user control).

### Temporary data storage:

Your files are temporarily stored in the `$HOME\workspaces\` directory. These files are deleted during the automatic system revert process.

### Protecting your data:

To avoid data loss, regularly back up your data using external USB drives or cloud storage services.

### Reason for automatic system reverts:

This feature is designed to maintain a consistent simulation environment for all users.



# Open a terminal emulator on macOS

For macOS users:

1. Click the **Launchpad** icon in the Dock.
2. Type  in the search field.
3. Click **Terminal**.

Expected outcome: A terminal emulator will open.

# Run a container on macOS

For macOS users, run this command:

```
docker container run -it \  
--name ee6455-gem5 \  
--volume ~/workspaces/:/workspaces/ \  
--workdir /workspaces/2025/ \  
--hostname EE6455-gem5 \  
ghcr.io/gem5/devcontainer:v25-0
```

Expected outcome: An interactive TTY will indicate its readiness to accept commands.

```
root@EE6455-gem5:/workspaces/2025#
```

Alternative outcome: A newer image will be downloaded before the interactive TTY displays the username, hostname, and working directory.

```
docker container run --interactive --tty \  
--volume ~/workspaces/:/workspaces/ \  
--workdir /workspaces/2025/ \  
--hostname EE6455-gem5 \  
ghcr.io/gem5/devcontainer:v25-0  
Unable to find image 'ghcr.io/gem5/devcontainer:v25-0' locally  
v25-0: Pulling from gem5/devcontainer  
00d679a470c4: Pull complete  
c782a11a41b6: Pull complete  
4f6b9996da3d: Pull complete  
bb60cbcef558: Pull complete  
92c0abbbf0ee: Pull complete  
865d1839a002: Pull complete  
2ceb23f2c7bb: Pull complete  
31825ae2d134: Pull complete  
Digest: sha256:dc299b8bf11b324cbd89aab82bdbe31bf9ce71a33386f2a2153a590a803d2c71  
Status: Downloaded newer image for ghcr.io/gem5/devcontainer:v25-0  
root@EE6455-gem5:/workspaces/2025#
```

## Note on KVM and Full-System Simulation:

In several chapters of the gem5 bootcamp, **KVM** acceleration is used to speed up simulation.

- **Full-System (02-Using-gem5/07-full-system)** and **CHI Protocol (03-Developing-gem5-models/07-chi-protocol)** – these chapters use **KVM acceleration** to speed up full-system simulations (e.g., booting Ubuntu or running workloads with advanced systems).

Workload Example (07-full-system):

- `x86-ubuntu-24.04-boot-with-systemd`
  - With KVM: ~1–2 minutes
  - Without KVM: 3+ hours

(continued on next page)



## Supported Host OS:

- **Windows Pro / Enterprise with Hyper-V enabled**

If **nested virtualization** is enabled and you run the container with `--device /dev/kvm`, then KVM acceleration will work.

See Microsoft's official docs:

- [Overview of Nested Virtualization](#)
- [Enable Nested Virtualization](#)

- **Linux Host OS**

If your CPU/BIOS supports virtualization (Intel VT-x or AMD-V) and `/dev/kvm` is available, you can pass it into the container with `--device /dev/kvm`.

## Not Supported Host OS:

- **Windows Home Edition**

Hyper-V and nested virtualization are not supported. In this case, KVM cannot be used and simulations will run **without hardware acceleration**, which will be significantly slower.



- **macOS (Intel & Apple Silicon)**

macOS does not expose `/dev/kvm` to Docker. Even on Apple Silicon (M1/M2/M3/M4), KVM cannot be used inside containers.

Simulations must therefore run **without hardware acceleration**, which will be much slower.

**Important:** KVM requires the host ISA to match the guest ISA:

- x86 host → can accelerate x86 guest (e.g., Ubuntu x86 FS workloads)
- ARM host → can accelerate ARM guest
- Cross-ISA acceleration (e.g., x86 host running RISC-V guest) is **not supported**.

## Manage an Existing Docker Container

- Start an existing container:

```
docker start ee6455-gem5
```

- Attach a shell inside a running container:

```
docker exec -it ee6455-gem5 bash
```

- Opens a new Bash shell inside the container.
- You can have multiple terminals connected at the same time.

- Stop a running container:

```
docker stop ee6455-gem5
```



- Remove a container

```
docker rm ee6455-gem5
```

- Works only if the container is stopped.
- **Warning:** Removing a container will delete all data created inside it (unless you used volumes or bind mounts to persist the data).

# Get the source code of gem5

Clone the Bootcamp Repository (inside container):

```
time git clone --recurse-submodules \
https://nas.larc-nthu.net:8443/ee6455_2025/public-gem5bootcamp-2025 /workspaces/2025/
```

Expected outcome: A repository will be cloned, including its submodules.

```
root@EE6455-gem5:/workspaces/2025# time git clone --recurse-submodules https://nas.larc-nthu.net:8443/ee6455_2025/public-gem5bootcamp-2025 /workspaces/2025/
Cloning into '/workspaces/2025'...
warning: redirecting to https://nas.larc-nthu.net:8443/ee6455_2025/public-gem5bootcamp-2025.git/
remote: Enumerating objects: 11106, done.
remote: Counting objects: 100% (130/130), done.
remote: Compressing objects: 100% (130/130), done.
remote: Total 11106 (delta 78), reused 0 (delta 0), pack-reused 10976 (from 1)
Receiving objects: 100% (11106/11106), 385.46 MiB | 11.07 MiB/s, done.
Resolving deltas: 100% (7510/7510), done.
Updating files: 100% (966/966), done.
Submodule 'gem5' (https://github.com/gem5/gem5) registered for path 'gem5'
Submodule 'gem5-resources' (https://github.com/gem5/gem5-resources) registered for path 'gem5-resources'
Cloning into '/workspaces/2025/gem5'...
remote: Enumerating objects: 295271, done.
remote: Counting objects: 100% (604/604), done.
remote: Compressing objects: 100% (329/329), done.
remote: Total 295271 (delta 460), reused 275 (delta 275), pack-reused 294667 (from 3)
Receiving objects: 100% (295271/295271), 171.33 MiB | 5.20 MiB/s, done.
Resolving deltas: 100% (225064/225064), done.
```

# Build and Setup gem5 with MESI and CHI

1. Copy the provided build configuration files:

```
cp /workspaces/2025/gem5_build_opts/* /workspaces/2025/gem5/build_opts
```

2. Build gem5 with MESI and CHI protocols (this will take a while):

```
cd /workspaces/2025/gem5  
scons build/MESI/gem5.opt -j6  
scons build/CHI/gem5.opt -j6
```

**Why `-j6` instead of `-j$(nproc)`?**

On macOS, using all available cores (`-j$(nproc)`) may cause build errors due to memory limits, such as:

```
{standard input}: Error: open CFI at the end of file; missing .cfi_endproc directive  
g++: fatal error: Killed signal terminated program cc1plus
```

Limiting jobs to `-j6` helps avoid this issue while still keeping parallel compilation.

3. Create symbolic links so you can run them with simple commands:

```
sudo ln -sf /workspaces/2025/gem5/build/MESI/gem5.opt /usr/local/bin/gem5-mesi  
sudo ln -sf /workspaces/2025/gem5/build/CHI/gem5.opt /usr/local/bin/gem5
```

**Note:** This will overwrite any existing `/usr/local/bin/gem5` or `/usr/local/bin/gem5-mesi`.

4. Verify the builds:

Check the symbolic links:

```
ls -l $(which gem5)  
ls -l $(which gem5-mesi)
```



Check the build-time configuration:

```
gem5 -c "import m5; print(vars(m5.defines))"  
gem5-mesi -c "import m5; print(vars(m5.defines))"
```

These commands will print out the build-time configuration so you can confirm whether **Ruby**, **protocol**, and **ISA** settings are correct.





# Preserve downloaded workloads across container recreations

By default, gem5 downloads resources (kernels, disk images, benchmarks) into `/root/.cache/gem5` inside the container.

If the container is deleted, these downloads will also be lost.

To avoid re-downloading large workloads every time, you can set an environment variable to redirect the resource cache to a persistent directory:

```
export GEM5_RESOURCE_DIR=/workspaces/resources
```

This way, when the container is removed and recreated, previously downloaded workloads will still be available.



# Run a simulation using gem5

Run this command inside the container:

```
time gem5-mesi --outdir=/workspaces/m5out/ \
/workspaces/2025/materials/01-Introduction/02-getting-started/completed/basic.py
```

Expected outcome:

- The `X86DemoBoard` will be simulated using `x86-ubuntu-24.04-img` as the workload.
- On first run, gem5 will automatically download required resources:
  - `x86-linux-kernel-5.4.0-105-generic`
  - `x86-ubuntu-24.04-img` (~4.9 GB , takes some time to download and decompress)

Note: Depending on your network speed, downloading and unpacking `x86-ubuntu-24.04-img` may take several minutes.



```

root@EE6455-gem5:/workspaces/2025# time gem5-mesi --outdir=/workspaces/m5out/ \
/workspaces/2025/materials/01-Introduction/02-getting-started/completed/basic.py
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.0
gem5 compiled Jul 25 2024 18:47:27
gem5 started Aug 21 2025 09:19:18
gem5 executing on EE6455-gem5, pid 14
command line: gem5-mesi --outdir=/workspaces/m5out/ /workspaces/2025/materials/01-Introduction/02-getting-started/completed/basic.py

warn: The X86DemoBoard is solely for demonstration purposes. This board is not known to be representative of any real-world system. Use with caution.
info: Using default config
Resource 'x86-linux-kernel-5.4.0-105-generic' was not found locally. Downloading to '/root/.cache/gem5/x86-linux-kernel-5.4.0-105-generic'...
Finished downloading resource 'x86-linux-kernel-5.4.0-105-generic'.
Resource 'x86-ubuntu-24.04-img' was not found locally. Downloading to '/root/.cache/gem5/x86-ubuntu-24.04-img.gz'...
Finished downloading resource 'x86-ubuntu-24.04-img'.
Decompressing resource 'x86-ubuntu-24.04-img' ('/root/.cache/gem5/x86-ubuntu-24.04-img.gz')...
Finished decompressing resource 'x86-ubuntu-24.04-img'.
warn: Max ticks has already been set prior to setting it through the run call. In these cases the max ticks set through the `run` function is used
Global frequency set at 1000000000000 ticks per second
src/mem/dram_interface.cc:690: warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (2048 Mbytes)
src/sim/kernel_workload.cc:46: info: kernel located at: /root/.cache/gem5/x86-linux-kernel-5.4.0-105-generic
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.
    0: board.pc.south_bridge.cmos rtc: Real-time clock set to Sun Jan  1 00:00:00 2012
board.pc.com_1.device: Listening for connections on port 3456
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.
src/dev/intel_8254_timer.cc:128: warn: Reading current count from inactive timer.
board.remote_gdb: Listening for connections on port 7000
src/sim/simulate.cc:199: info: Entering event queue @ 0. Starting simulation...
src/mem/ruby/system/Sequencer.cc:680: warn: Replacement policy updates recently became the responsibility of SLICC state machines. Make sure to setMRU() near callbacks in .sm files!
build/ALL/arch/x86/generated/exec-ns.cc.inc:27: warn: instruction 'fninit' unimplemented

real    42m27.091s
user    0m38.201s
sys     0m16.827s
root@EE6455-gem5:/workspaces/2025#

```

# Retrieve the simulation statistics

- On macOS

After the simulation finishes, open `~\workspaces\m5out\stats.txt` using a text editor. This file contains the detailed statistics generated by gem5.

- Inside the container

You can also view the file directly inside the running container, for example by scrolling through the whole file:

```
root@EE6455-gem5:/workspaces/2025# less /workspaces/m5out/stats.txt
```

Expected outcome: The first few lines will resemble this:

```
----- Begin Simulation Statistics -----
simSeconds          0.020000          # Number of seconds simulated (Second)
simTicks            20000000000        # Number of ticks simulated (Tick)
finalTick           20000000000        # 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         18.88              # Real time elapsed on the host (Second)
hostTickRate        1059100397         # The number of ticks simulated per host second (ticks/s) ((Tick/Second))
hostMemory          2770924            # Number of bytes of host memory used (Byte)
simInsts            7479814           # Number of instructions simulated (Count)
simOps              34912342           # Number of ops (including micro ops) simulated (Count)
hostInstRate        396059            # Simulator instruction rate (inst/s) ((Count/Second))
hostOpRate          1848597           # Simulator op (including micro ops) rate (op/s) ((Count/Second))
board.cache_hierarchy.ruby_system.delayHistogram::bucket_size 2          # delay histogram for all message (Unspecified)
board.cache_hierarchy.ruby_system.delayHistogram::max_bucket 19          # delay histogram for all message (Unspecified)
board.cache_hierarchy.ruby_system.delayHistogram::samples      735551      # delay histogram for all message (Unspecified)
board.cache_hierarchy.ruby_system.delayHistogram::mean         1.036855      # delay histogram for all message (Unspecified)
board.cache_hierarchy.ruby_system.delayHistogram::stdev        2.687016      # delay histogram for all message (Unspecified)
```

# Retrieve the simulation configuration

- On macOS: open `~\workspaces\m5out\config.ini` using a text editor.
- Inside the container:

```
less /workspaces/m5out/config.ini
```

Expected outcome: The first few lines will resemble this:

```
[board]
type=System
children=cache_hierarchy clk_domain dvfs_handler iobus memory pc processor workload
auto_unlink_shared_backstore=false
cache_line_size=64
eventq_index=0
exit_on_work_items=true
init_param=0
m5ops_base=4294901760
```

```
mem_mode=timing
mem_ranges=0:2147483648 3221225472:3222274048
memories=board.memory.mem_ctrl.dram
mmap_using_noreserve=false
multi_thread=false
num_work_ids=16
readfile=
redirect_paths=
shadow_rom_ranges=
shared_backstore=
symbolfile=
thermal_components=
thermal_model=None
work_begin_ckpt_count=0
work_begin_cpu_id_exit=-1
work_begin_exit_count=0
work_cpus_ckpt_count=0
work_end_ckpt_count=0
work_end_exit_count=0
```