

Assignment 2: Using GEM5 with a "Hello World" Program for the x86 ISA

**Milan Bista**

University of Cumberlands

MSCS-531-M50: Computer Architecture and Design

Instructors: Machica Mcclain / Charles Lively

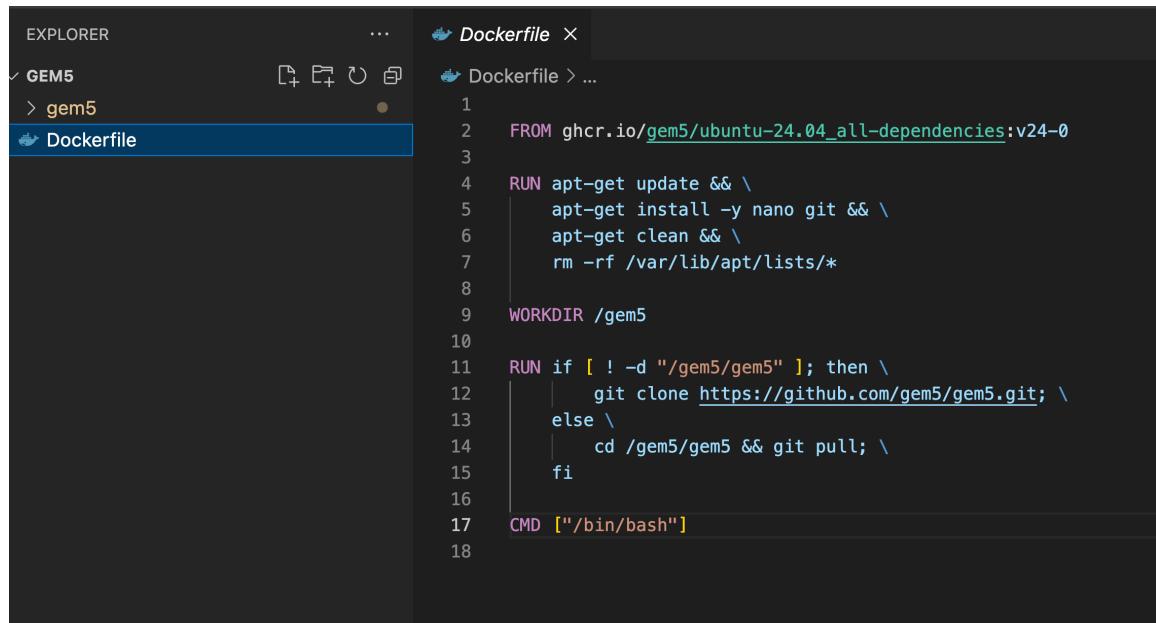
## 1. Environment Setup:

The first step is to install required dependencies and clone the gem5 from github

- *sudo apt-get update*
- *sudo apt-get install python3 scons gcc g++*

I am using **Mac M1** which is a **64-bit ARM** processor so, I have created a Docker file and ran docker image for the gem5 simulator and **mounted the volume** to my local folder

- docker build -t gem5-custom-image .
- docker run --name gem5-container --platform linux/arm64 -u \$UID:\$GID --volume "/Users/milanbista/Documents/Cumberlands/ComputerArchitecture&Organization/gem5:/gem5" --rm -it gem5-custom-image



A screenshot of the Visual Studio Code interface. On the left, the Explorer sidebar shows a project named 'GEM5' with a folder named 'gem5'. In the center, there is a code editor window titled 'Dockerfile'. The code in the editor is as follows:

```
FROM ghcr.io/gem5/ubuntu-24.04_all-dependencies:v24-0
RUN apt-get update && \
    apt-get install -y nano git && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*
WORKDIR /gem5
RUN if [ ! -d "/gem5/gem5" ]; then \
    git clone https://github.com/gem5/gem5.git; \
else \
    cd /gem5/gem5 && git pull; \
fi
CMD ["/bin/bash"]
```

```
(base) milanbista@Milans-MacBook-Pro gem5 % docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
698d09e06cd8 gem5-custom-image "/bin/bash" 3 days ago Up 3 days gem5-container
(base) milanbista@Milans-MacBook-Pro gem5 %
```

## 2. Clone gem5 Repository:

I have cloned the gem5 repository from following git repository URL. Since I am using a Docker file I have written a custom run command in above dockerfile to clone the gem5 repository whenever an image is created.

- git clone <https://github.com/gem5/gem5>
- cd /gem5

```
I have no name!@698d09e06cd8:/gem5$ ls
Dockerfile  gem5
I have no name!@698d09e06cd8:/gem5$
```

## 3. Building gem5 for x86:

I ran the following command for the gem5 simulator build for the x86 architecture. This process took about 25 minutes on my system.

- scons build/X86/gem5.opt -j4

The build file can be found in the volume as  
Cd build/x86/

```
I have no name!@698d09e06cd8:/gem5$ ls
Dockerfile  gem5
I have no name!@698d09e06cd8:/gem5$ cd gem5
I have no name!@698d09e06cd8:/gem5$ ls
CODE-OF-CONDUCT.md  KCONFIG.md      README.md    TESTING.md   build_tools  include      milan-test      requirements.txt  system      tests
CONTRIBUTING.md    LICENSE        RELEASE-NOTES.md build        configs     m5out       optional-requirements.txt  site_scons  test2.py  util
COPYING            MAINTAINERS.yaml  build_opts  ext         milan-local-resources  pyproject.toml  src        test3.py
I have no name!@698d09e06cd8:/gem5$ cd build
I have no name!@698d09e06cd8:/gem5/gem5$ ls
x86
I have no name!@698d09e06cd8:/gem5/gem5$ build x86
I have no name!@698d09e06cd8:/gem5/gem5/build/x86$ ls
arch  base  config  cpu  debug  dev  enums  ext  gem5.build  gem5.opt  gem5py  gem5py_m5  kern  learning_gem5  mem  params  proto  python  sim  sst  systemc
I have no name!@698d09e06cd8:/gem5/gem5/build/x86$
```

## 4. Writing the "Hello World" Program:

I wrote the program hello.c in milan-test folder

```
gem5$ docker run --name gem5-container --platform linux/arm64 -u 501:20 --volume ~/Documents/Cumberlands/ComputerArchitecture&Organization/gem5:/gem5 --rm -it gem5...  
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ls  
resources sample2.py simple.py test1.py  
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ nano hello.c  
Unable to create directory //local/share/nano/: No such file or directory  
It is required for saving/loading search history or cursor positions.  
  
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ cat hello.c  
#include <stdio.h>  
  
int main(){  
    printf("Hello, World!\n");  
    return 0;  
}  
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$
```

## 5. Compiling the Program:

I compiled the file hello.c using gcc compiler with below command line script and successfully created a binary

- *gcc hello.c -o hello*

```
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ls
resources sample2.py simple.py test1.py
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ nano hello.c
Unable to create directory //local/share/nano/: No such file or directory
It is required for saving/loading search history or cursor positions.

I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ cat hello.c
#include <stdio.h>

int main(){
    printf("Hello, World!\n");
    return 0;
}

I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ gcc hello.c -o hello
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ./hello
Hello, World!
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ls
hello hello.c resources sample2.py simple.py test1.py
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$
```

### Troubleshoot:

I encountered a problem when compiling hello.c in my VS-code which is mounted to a volume. When compiling the code in local machine, it creates a MacOS ARM 64 binary executable which

is incompatible to run in x86 64 bit processor. The solution is to compile the within the container itself which will create ELF(Executable and Linkable Format) which is for x86 processor

The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows a tree view of files and folders, including GEM5, milan-test, resources, hello, hello.c, sample2.py, simple.py, and test1.py.
- TERMINAL**: Displays the command-line output of the terminal session.
- Code Editor**: Shows the content of `hello.c`:

```
1 #include <stdio.h>
2
3 int main(){
4     printf("Hello, World!\n");
5     return 0;
6 }
7
```

- Terminal Output**: Shows the following commands and their results:

  - (base) milanbista@Milans-MacBook-Pro milan-test % gcc hello.c -o hello
  - (base) milanbista@Milans-MacBook-Pro milan-test % file hello
  - hello: Mach-O 64-bit executable arm64
  - (base) milanbista@Milans-MacBook-Pro milan-test %

- Bottom Terminal**: Shows a second terminal window with the following command and output:

```
gem5 -- docker run --name gem5-container --platform linux/arm64 -u 501:20 --volume ~/Documents/Cumberlands/ComputerArchitecture&Organization/gem5:/gem5/gem5 --rm -it gem5...  
I have no name@698d09e06cd8:/gem5/gem5/milan-test$ gcc hello.c -o hello  
I have no name@698d09e06cd8:/gem5/gem5/milan-test$ file hello  
hello: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=5c613775d2a4a3eb772fb7c4eaaf368c86a13  
d64, for GNU/Linux 3.2.0, not stripped  
I have no name@698d09e06cd8:/gem5/gem5/milan-test$ ./hello  
Hello, World!  
I have no name@698d09e06cd8:/gem5/gem5/milan-test$
```

## 6. Running the Program in gem5:

## ***Setup a Simple Simulation Script:***

I wrote a python script run\_hello.py to run the simulation. The file contains a simple 1 core CPU with L1 Cache connected to Membus

```
[I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ cat run_hello.py
import m5
from m5.objects import *
import sys

#Setting L1 Cache System
class L1Cache(Cache):
    assoc = 2
    tag_latency = 2
    data_latency = 2
    response_latency = 2
    mshrs = 4
    tgts_per_mshr = 20

class L1ICache(L1Cache):
    size = '16kB'

class L1DCache(L1Cache):
    size = '64kB'

#Create the system for our architecture simulation
system = System()
system.clk_domain = SrcClockDomain()
system.clk_domain.clock = '1GHz'
system.clk_domain.voltage_domain = VoltageDomain()

#Simple CPU and Memory Config
system.mem_mode = 'timing'
system.mem_ranges = [AddrRange('512MB')]
system.cpu = TimingSimpleCPU()
system.membus = SystemXBar()

#Memory Control
system.mem_ctrl = MemCtrl()
system.mem_ctrl.dram = DDR3_1600_8x8()
system.mem_ctrl.dram.range = system.mem_ranges[0]
system.mem_ctrl.port = system.membus.mem_side_ports

#Creating a Simple L1 Cache System
system.cpu.icache = L1ICache() # L1 Instruction Cache
system.cpu.dcache = L1DCache() # L1 Data Cache

#Connecting CPU and Cache with iCache and dCache ports
system.cpu.icache.cpu_side = system.cpu.icache_port
system.cpu.dcache.cpu_side = system.cpu.dcache_port

#Connecting L1 Cache to membus
system.cpu.icache.mem_side = system.membus.slave
system.cpu.dcache.mem_side = system.membus.slave

#This specific interrupts is required for X86 processor
system.cpu.createInterruptController()
system.cpu.interrupts[0].pio = system.membus.mem_side_ports
system.cpu.interrupts[0].int_requestor = system.membus.cpu_side_ports
system.cpu.interrupts[0].int_responder = system.membus.mem_side_ports
system.system_port = system.membus.slave

#Taking input from command line for binary file
binary = sys.argv[sys.argv.index('-c') + 1]

# Create a process for the provided binary
# And create a workload
process = Process()
process.cmd = [binary]
system.cpu.workload = process
system.cpu.createThreads()
system.workload = SEWorkload.init_compatible(binary)

#Simulation Configuration
root = Root(full_system=False, system=system)
m5.instantiate()
print("Beginning of the Simulation!!!!")
exit_event = m5.simulate()
print('Exiting @ tick {} because {}'.format(m5.curTick(), exit_event.getCause()))
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$
```

### *Run gem5 Simulation:*

I ran the simulation using following command where I am using the gem5 x86 simulation build to test run\_hello.py system passing hello as a binary file

*./build/X86/gem5.opt run\_hello.py -c hello*

```
● ● ● █ gem5 — docker run --name gem5-container --platform linux/arm64 -u 501:20 --volume ~/Documents/Cumberlands/ComputerArchitecture&Organization/g
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ./build/x86/gem5.opt run_hello.py -c hello
gem5 Simulator System. https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 6 2024 17:16:33
gem5 started Sep 22 2024 17:08:28
gem5 executing on 698d09e06cd8, pid 659
command line: ./build/x86/gem5.opt run_hello.py -c hello

warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
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 (512 Mbytes)
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.
system.remote_gdb: Listening for connections on port 7000
Running in the simulation!!!
```

## Troubleshooting:

When I ran the command for the first time there were a lot of error, particularly deprecated code where membus use `cpu_side_ports` instead of master slave nomenclature. And, I had some issues with properly connecting the correct ports from CPU to L1Cache to membus.

```
gem5@698d09e6cd8:/gem5/gem5/milan-test$ ./build/x86/gem5.opt run_hello.py -c hello
gem5 Simulator System. https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

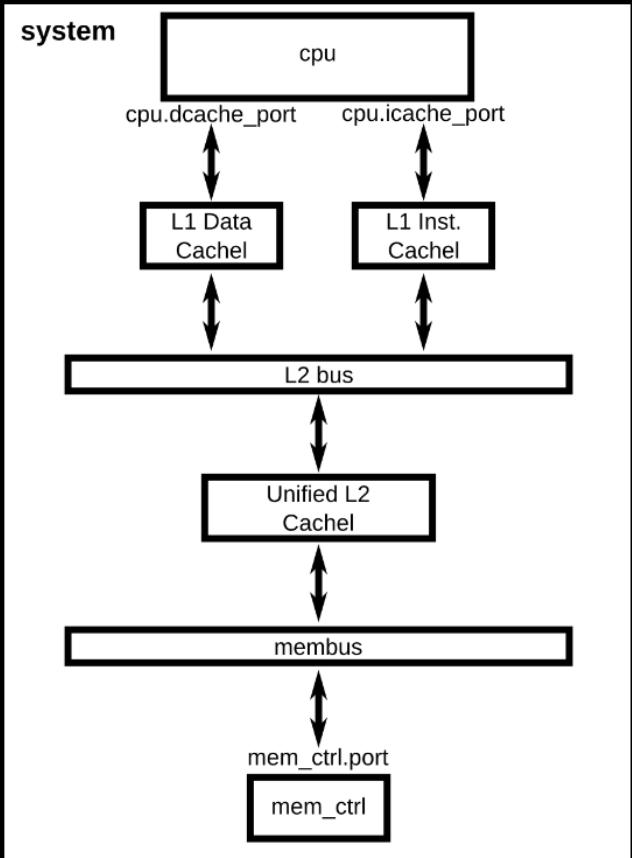
gem5 version 24.0.0.1
gem5 compiled Sep  6 2024 17:16:33
gem5 started Sep 22 2024 17:08:28
gem5 executing on 698d09e6cd8, pid 659
command line: ./build/x86/gem5.opt run_hello.py -c hello

warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
Global frequency set at 100000000000 ticks per second
src/mem/dram_interface.cc:690: warn: DRAM device capacity (8192 Mbytes) does not match the address range assigned (512 Mbytes)
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.
system.remote_gdb: Listening for connections on port 7000
Renaming of the Simulation!!!!
```

So, I referred to gem5 official documentation to learn about the architecture and was able to correctly configure the system which helped me run the simulation correctly.

 **Adding cache to the configuration script**

Using the [previous configuration script as a starting point](#), this chapter will walk through a more complex configuration. We will add a cache hierarchy to the system as shown in the figure below. Additionally, this chapter will cover understanding the gem5 statistics output and adding command line parameters to your scripts.



```
graph TD; subgraph system [system]; direction TB; cpu[cpu] -- "cpu.dcache_port" --> l1dcache[L1 Data Cachel]; cpu -- "cpu.icache_port" --> l1icache[L1 Inst. Cachel]; l1dcache <--> l1icache; l1dcache --> l2bus[L2 bus]; l1icache --> l2bus; l2bus --> ul2cachel[Unified L2 Cachel]; ul2cachel --> membus[membus]; membus --> memctrl[mem_ctrl]; end;
```

The corrected run\_hello.py script and the output given below, which can also be found in my github account at <https://github.com/mbista25742/MSCS-531-M50-Assignment2>

```
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ cat run_hello.py
import m5
from m5.objects import *
import sys

#Setting L1 Cache System
class L1Cache(Cache):
    assoc = 2
    tag_latency = 2
    data_latency = 2
    response_latency = 2
    mshrs = 4
    tgts_per_mshr = 20

class L1ICache(L1Cache):
    size = '16kB'

class L1DCache(L1Cache):
    size = '64kB'

#Create the system for our architecture simulation
system = System()
system.clk_domain = SrcClockDomain()
system.clk_domain.clock = '1GHz'
system.clk_domain.voltage_domain = VoltageDomain()

#Simple CPU and Memory Config
system.mem_mode = 'timing'
system.mem_ranges = [AddrRange('8192MB')]
system.cpu = TimingSimpleCPU()
system.membus = SystemXBar()

#Memory Control
system.mem_ctrl = MemCtrl()
system.mem_ctrl.dram = DDR3_1600_8x8()
system.mem_ctrl.dram.range = system.mem_ranges[0]
system.mem_ctrl.port = system.membus.mem_side_ports

#Creating a Simple L1 Cache System
system.cpu.icache = L1ICache() # L1 Instruction Cache
system.cpu.dcache = L1DCache() # L1 Data Cache

#Connecting CPU and Cache with iCache and dCache ports
system.cpu.icache.cpu_side = system.cpu.icache_port
system.cpu.dcache.cpu_side = system.cpu.dcache_port

#Connecting L1 Cache to membus
system.cpu.icache.mem_side = system.membus.cpu_side_ports
system.cpu.dcache.mem_side = system.membus.cpu_side_ports

#This specific interrupts is required for X86 processor
system.cpu.createInterruptController()
system.cpu.interrupts[0].pio = system.membus.mem_side_ports
system.cpu.interrupts[0].int_requestor = system.membus.cpu_side_ports
system.cpu.interrupts[0].int_responder = system.membus.mem_side_ports
system.system_port = system.membus.cpu_side_ports

#Taking input from command line for binary file
binary = sys.argv[sys.argv.index('-c') + 1]

# Create a process for the provided binary
# And create a workload
process = Process()
process.cmd = [binary]
system.cpu.workload = process
system.cpu.createThreads()
system.workload = SWorkload.init_compatible(binary)

#Simulation Configuration
root = Root(full_system=False, system=system)
m5.instantiate()
print("Beginning of the Simulation!!!!")
exit_event = m5.simulate()
print('Exiting @ tick {} because {}'.format(m5.curTick(), exit_event.getCause()))
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$
```

And the output is:

---

```

I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ./build/x86/gem5.opt run_hello.py -c hello
gem5 Simulator System. https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 24.0.0.1
gem5 compiled Sep 6 2024 17:16:33
gem5 started Sep 22 2024 17:20:03
gem5 executing on 698d09e06cd8, pid 672
command line: ./build/x86/gem5.opt run_hello.py -c hello

Global frequency set at 10000000000 ticks per second
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
system.remote_gdb: Listening for connections on port 7000
Beginning of the Simulation!!!!
src/sim/mem_state.cc:199: info: Entering event queue @ 0. Starting simulation...
src/sim/mem_state.cc:448: info: Increasing stack size by one page.
src/sim/syscall_emul.cc:74: warn: ignoring syscall set_robust_list(...)
src/sim/syscall_emul.cc:74: warn: ignoring syscall rseq(...)
src/sim/syscall_emul.cc:74: warn: ignoring syscall mprotect(...)
src/sim/syscall_emul.cc:74: warn: ignoring syscall mprotect(...)
src/sim/syscall_emul.cc:74: warn: ignoring syscall mprotect(...)
Hello, World! from C program
Exiting @ tick 492059000 because exiting with last active thread context
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ls
__pycache__ a.out caches.py hello hello.c m5out resources run_hello.py sample2.py simple.py test1.py test2.py test3.py test4.py
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ cd m5out
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ ls
citations.bib config.board.cache_hierarchy.ruby_system.dot config.board.cache_hierarchy.ruby_system.pdf config.board.cache_hierarchy.r
I have no name!@698d09e06cd8:/gem5/gem5/milan-test$ cat stats.txt

----- Begin Simulation Statistics -----
simSeconds 0.000492 # Number of seconds simulated (Second)
simTicks 492059000 # Number of ticks simulated (Tick)
finalTick 492059000 # Number of ticks from beginning of simulation (restored from checkpoint)
simFreq 10000000000000 # The number of ticks per simulated second ((Tick/Second))
hostSeconds 0.21 # Real time elapsed on the host (Second)
hostTickRate 2291993889 # The number of ticks simulated per host second (ticks/s) ((Tick/Second))
hostMemory 8869144 # Number of bytes of host memory used (Byte)
simInsts 97587 # Number of instructions simulated (Count)
simOps 187979 # Number of ops (including micro ops) simulated (Count)
hostInstRate 451614 # Simulator instruction rate (inst/s) ((Count/Second))
hostOpRate 869681 # Simulator op (including micro ops) rate (op/s) ((Count/Second))
system.clk_domain.clock 1000 # Clock period in ticks (Tick)
system.clk_domain.voltage_domain.voltage 1 # Voltage in Volts (Volt)
system.cpu.numCycles 492059 # Number of cpu cycles simulated (Cycle)
system.cpu.cpi 5.039420 # CPI: cycles per instruction (core level) ((Cycle/Count))
system.cpu.ipc 0.198436 # 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 97642 # Number of instructions committed (thread level) (Count)
system.cpu.commitStats0.numOps 188073 # 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 5.039420 # CPI: cycles per instruction (thread level) ((Cycle/Count))
system.cpu.commitStats0.ipc 0.198436 # IPC: instructions per cycle (thread level) ((Count/Cycle))
system.cpu.commitStats0.numMemRefs 0 # Number of memory references committed (Count)
system.cpu.commitStats0.numFpInsts 6152 # Number of float instructions (Count)
system.cpu.commitStats0.numIntInsts 182418 # Number of integer instructions (Count)
system.cpu.commitStats0.numLoadInsts 24804 # Number of load instructions (Count)
system.cpu.commitStats0.numStoreInsts 12789 # Number of store instructions (Count)
system.cpu.commitStats0.numVecInsts 0 # Number of vector instructions (Count)
system.cpu.commitStats0.committedInstType::No_OpClass 1291 0.6% 0.69% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::IntAlu 143393 76.24% 76.93% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::IntMult 685 0.36% 0.36% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::IntDiv 1407 0.75% 0.75% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatAdd 390 0.21% 0.21% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatCmp 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatCvt 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatMult 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatMultAcc 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatDiv 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatMisc 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::FloatSqrt 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::SimdAdd 392 0.21% 0.21% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::SimdAddAcc 0 0.00% 0.00% # Class of committed instruction. (Count)
system.cpu.commitStats0.committedInstType::SimdAlu 863 0.46% 0.46% # Class of committed instruction. (Count)

```

Overall, this assignment was fun and provided me with valuable insights into gem5 simulation. It deepened my understanding of how CPUs and caches are configured, enhancing my knowledge in this area significantly.