Instructor: **Dr. Shayan (Sean) Taheri**

Setting Up Gem5

```
echo "export GEM5_HOME=$PWD" >> ~/.bashrc

source ~/.bashrc
```

Building X86 Architecture (Compilation)

Format: `scons <build dir>/<configuration>/<target>`

In the root directory, please enter: `scons build/X86/gem5.opt -j4`

Configuration: ALPHA, ARM, MIPS, POWER, SPARC, or X86.

Typical Value for Target: gem5.debug, gem5.opt, gem5.fast, gem5.prof, or gem5.perf.

-jN: Turning on multi-thread compilation. Parameter **N** specifies the number of threads.

Notice: the output of compilation process is an executable binary file.

Gem5 Modes

1) Full System (FS): For booting operating system.

2) Syscall Emulation (SE): For running individual applications.

Notice: SE cannot guarantee to support the simulation of all applications.

Running Gem5 Simulation

Format: `<gem5 binary> [gem5 binary's options] <simulation script (python)> [script's options]`

Setting Up Environment for SPEC Benchmarks

1) Enter the following command only one time:

```
echo export PERL5LIB=/opt/software/architecture/util:\$PERL5LIB >> ~/.bashrc

source ~/.bashrc
```

2) For each directory under simulation, enter the following command:

`/opt/software/architecture/util/setenvall.pl`

SimPoints (Simulation Points) of SPEC and PARSEC Benchmarks

They are in the following directory:

SPEC: `/opt/software/architecture/spec2k6_simpoint`

PARSEC: `/opt/software/architecture/PARSEC2.1.ckpts/v5`

Notice: The options for using checkpoints are according to the following:

1) `--checkpoint-dir=` : It needs the directory of the checkpoint.

2) `--checkpoint-restore=` : It needs the instruction number of the checkpoint to be restored.

3) `--simpoint`: It specifies the usage of "SimPoint" instruction number.

Out-Of-Order (O3) Processor

The command of simulation uses "Atomic CPU" by default, which is an extremely fast but less accurate CPU model.

For turning on the O3 CPU model, provide the following flags for the python script:

```
--caches --l2cache --cpu-type=detailed
```

Running SPEC Benchmarks (Example)

```
$GEM5_HOME/build/X86/gem5.opt --stats-file=tonto.o3stats.txt
$GEM5_HOME/configs/cmp.py -b tonto
--checkpoint-dir=/opt/software/architecture/spec2k6_simpoint/465.tonto
/ -r 426005016 -S --caches --l2cache --cpu-type=detailed
--maxinsts=10000000
```

Notice: The configuration of a simulation can be found in "config.ini" and "config.json" files.

Processor Parameters:

Some of the processor parameters can be modified in "cmp.py" file.

Architectural Modifications

After the desired modifications have been done on the processor (in Source Files), the gem5 needs to be compiled again.

Debug/Verify Architectural Modifications

The applied modifications on the architecture of the processor can be verified by providing **`--debug-flags=IQ`** option to the gem5.opt executable binary file.

Applying Patches

Several patches are required to be used for correct simulation of gem5 with ruby memory system using ALPHA processor or etc.

To apply patches, enter: `/opt/software/architecture/util/patches/apply.sh`

Enabling Ruby Memory System

```
scons PROTOCOL=MOESI_CMP_directory RUBY=True build/ALPHA/gem5.opt
```

Full System Simulation

Please follow the below steps:

1) Enter the following command only once:

```
echo "export M5_PATH=$PWD" >> ~/.bashrc

source ~/.bashrc
```

2) Acquire system images and kernel.

```
ln -s /opt/software/architecture/fs-alpha/binaries

ln -s /opt/software/architecture/fs-alpha/disks
```

PARSEC Benchmarks

They consist of several parallel programs that can be used for comprehensive evaluation of multi-core processors. These benchmarks are built into FS system images.

Running PARSEC Benchmarks (Example)

```
./build/ALPHA/gem5.opt ./configs/example/ruby_fs.py -n 8
--restore-with-cpu=detailed --cpu-type=detailed --mem-size=1024MB
--checkpoint-dir=/opt/software/architecture/PARSEC2.1.ckpts/v5/blacksc
holes -r 1
```