Computer Architecture (CSE511/ ECE511)
Assignment 2

Part-1

General Note: In assignment 1, you were able to use the available gem5 SimObjects to build your own system and play with it. In this assignment, you will get your hands dirty by diving into the gem5 coding environment and getting acquainted with its coding style. You will be able to create your own SimObjects and make it work on the gem5 platform.

The steps and deliverables for this assignment are listed below:

1. Create a SimObject called "**MathsOperations**". MathsOperations will call three events namely, **FibonacciGeneration, PrimeNumberCheck** and **GCDCalculation**. **FibonacciGeneration** will be called at tick "100". **PrimeNumberCheck** will be called at tick "1000". **GCDCalculation** will be called at tick "10000".
2. **FibonacciGeneration** will perform the generation of fibonacci sequences of a particular number. **PrimeNumberCheck** will perform a particular number if it is prime or not. **GCDCalculation** will perform GCD of two numbers. You can specify the number without taking any user input (i.e you can hardcode them within the SimObject).
3. Add DEBUG flags whose functionalities are given below. Each DEBUG flag should also have a proper description/annotation that is displayed on running the Simulation.
    a. DEBUG flag "FIBONACCISEQUENCE" will display the resultant fibonacci.
    b. DEBUG flag "FIBONACCINUMBER" will display the number used for fibonacci sequence generation.
    c. DEBUG flag "PRIMECHECK" will display the prime number or not.
    d. DEBUG flag "GCDRESULT" will display the resultant GCD operation.
    e. DEBUG flag "GCDNUMBER" will display the two numbers used in GCD operation.
4. Now that you have implemented your SimObject and added the required flags, create the configuration script to use your new SimObjects. You do not have to add a CPU or caches to the system for this assignment.

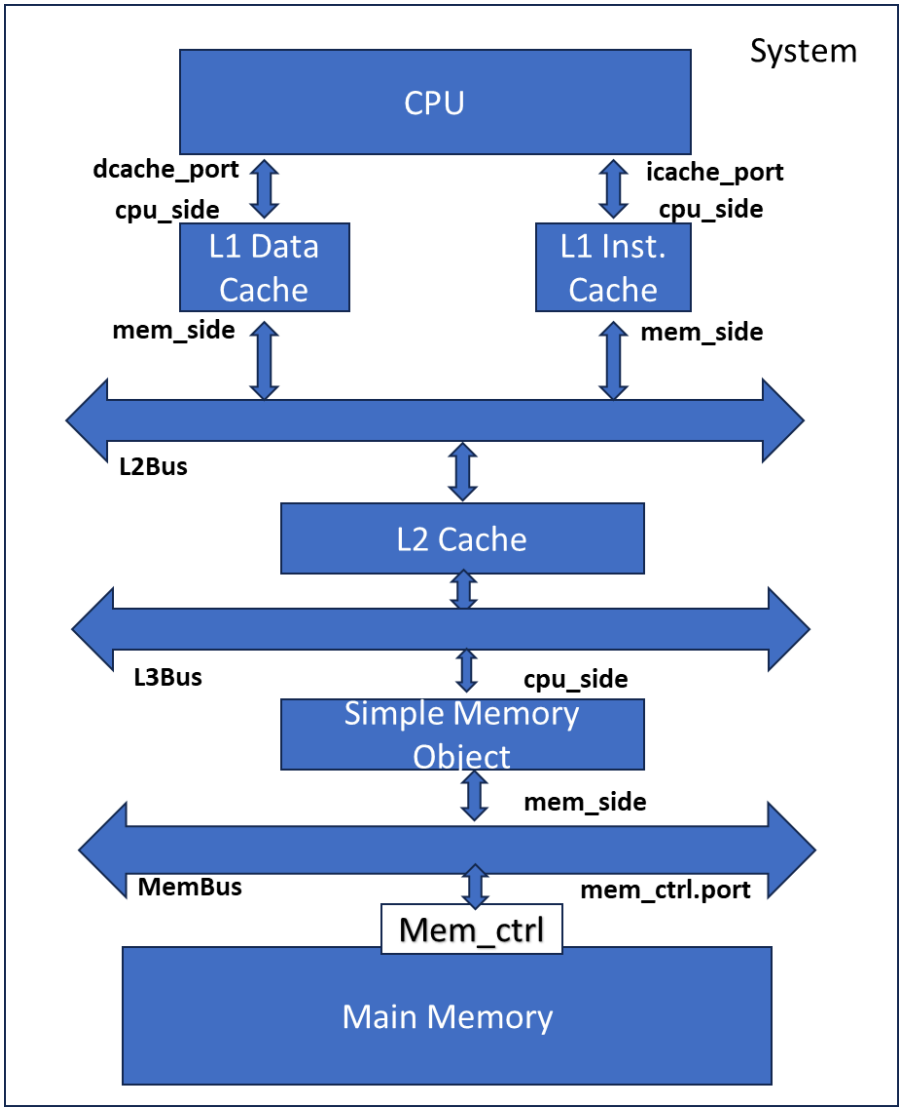
Note:

1. **You can optionally take in the cycles to execute the 3 events from the user as the bonus part instead of hardcoding them to 100, 1000 and 10000.**
2. **You can also optionally take in the elements from the user for the bonus part.**

Note that you should NOT be storing the resultant values in your code, rather you should compute them when the events are called.

Part-2

Similar to the configuration that you did in the assignment 1. Make a configuration script to simulate the system given in the figure below using CPU models **RiscvO3CPU()**. Run a benchmark application, " qsort_small from MiBench benchmark suite [1]" in system emulation mode. The configuration of caches is as given; L1 Data Cache (size = 16kB, Associativity = 2, latencies = 2); L1 Inst Cache (size = 16kB, Associativity = 2, latencies = 2); L2 Cache (size = 256kB, Associativity = 4, latencies =10).



Your job is to make the Simple memory object (for example, let us call this SimObject as **SimpleOBJ**). This will contain two ports. It will just simply pass the request through the CPU side(in this problem, from L2 cache misses) to the memory side (to the main memory). The cpu_side (port) will receive a request (basically a slave), and the mem_side (port) will send the request to the memory bus (basically a master). The functionality of this port is the one implemented on the gem5 website.
(https://www.gem5.org/documentation/learning_gem5/part2/memoryobject/)

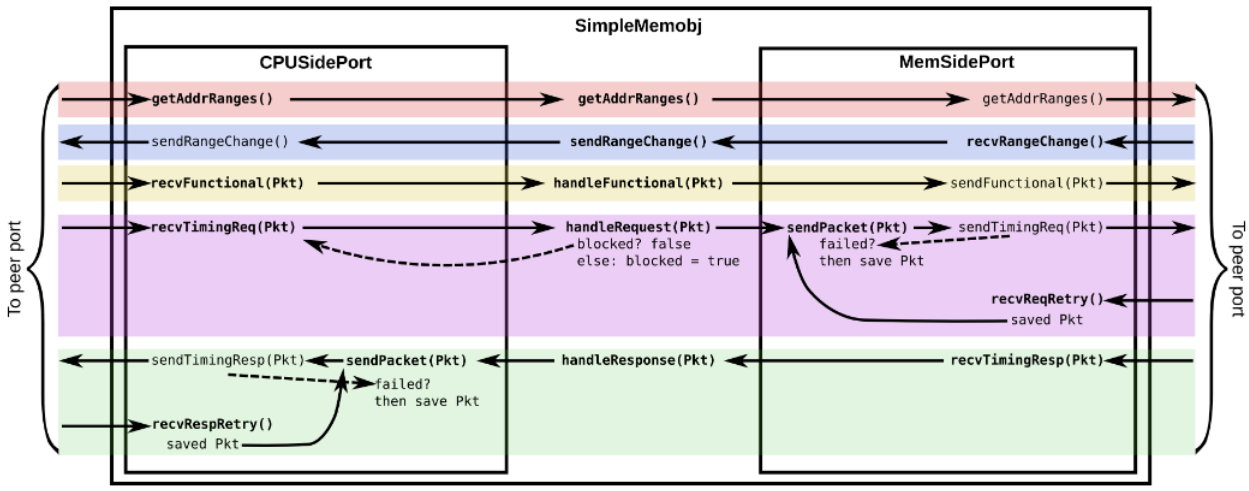


Fig. The relationships between the CPUSidePort, MemSidePort, and SimpleOBJ. As taken from the gem5 website[2].

Implement a debug flag similar to the one in the example mentioned in [2] and store that in a text file using the command given below.

```
build/RISCV/gem5.opt --debug-flags=SimpleMemobj configs/tutorial/two_level.py > file.txt
```

The file location and the Python file name may be different from the command given above so please check your file location and Python file name.

The file.txt will be similar to the figure shown below. You have to explain why sometimes you have stalling during evaluation.

```
Global frequency set at 1000000000000 ticks per second
      0: system.memobj: Sending new ranges
gem5 Simulator System.  https://www.gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 23.1.0.0
gem5 compiled Sep  4 2024 09:33:30
gem5 started Sep  4 2024 11:46:29
gem5 executing on 9dc06c1cd045, pid 14958
command line: build/RISCV/gem5.opt --debug-flags=SimpleMemobj1 configs/tutorial/two_level.py

Starting simulation
  23000: system.memobj: Got request for addr 0x100
  94000: system.memobj: Got response for addr 0x100
 144000: system.memobj: Got request for addr 0x140
 201000: system.memobj: Got response for addr 0x140
 201000: system.memobj.cache_port: Sending retry req for -1
 201000: system.memobj: Got request for addr 0x1580
 258000: system.memobj: Got response for addr 0x1580
 307000: system.memobj: Got request for addr 0x15c0
 364000: system.memobj: Got response for addr 0x15c0
 364000: system.memobj.cache_port: Sending retry req for -1
 364000: system.memobj: Got request for addr 0x1600
 421000: system.memobj: Got response for addr 0x1600
 469000: system.memobj: Got request for addr 0x1640
 526000: system.memobj: Got response for addr 0x1640
```

Deliverable for both parts:

1. You should also create a ReadMe file that explains the codes/scripts submitted.
2. Your README should also include the steps to compile and run the code.
3. Submit all the simulation script and codes in a zip file with the naming convention: <SA2_Rollnumber>.zip

Note:

1. The debug flag is taken from example [2] so you can give any name of your choice.

2. If you wish to give a different name for the SimObject please make sure that the same name is NOT present in gem5.

3. *The second part takes around 3 hours or more (depending on your system) to run so plan your time accordingly.*

Resource:https://www.gem5.org/documentation/learning_gem5/part2/environment/ and the following tutorials on the same website will give you related information.

[1]https://github.com/embecosm/mibench/tree/master/automotive/qsort

[2]https://www.gem5.org/documentation/learning_gem5/part2/memoryobject/