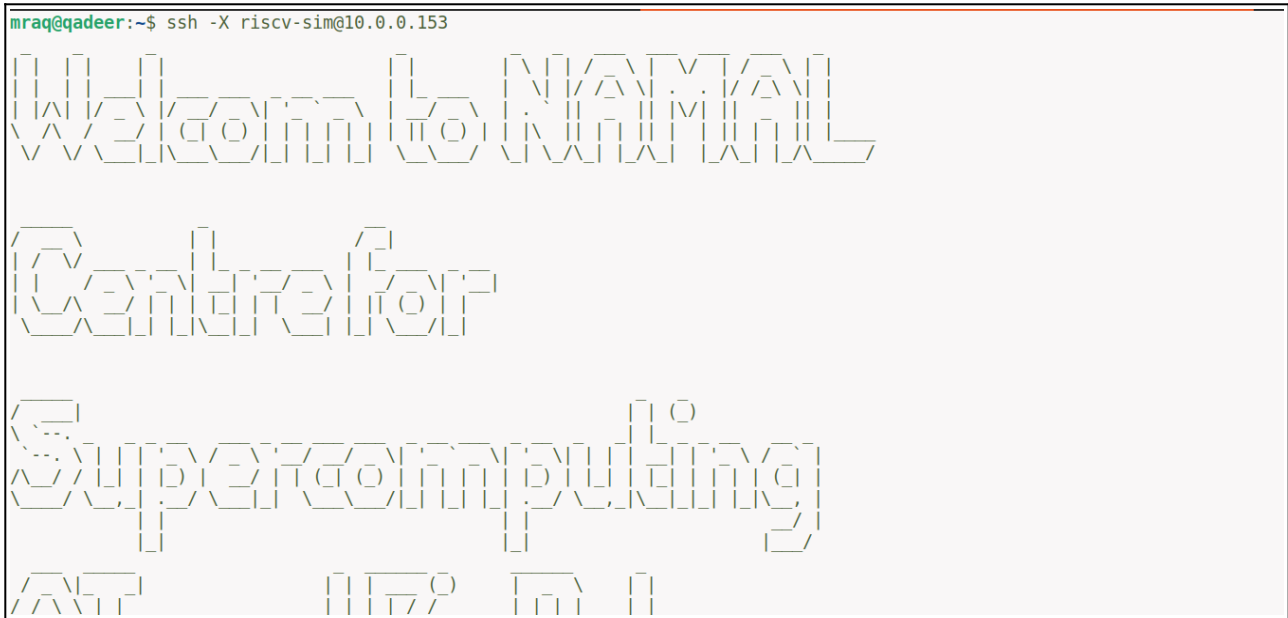


# Hands On Session \_ Spike Simulator

- **Step 1 : First of all how to access HPC cluster ?**

**Open terminal and run this command:**

`ssh -X riscv-sim@10.0.0.153`



**Figure 1: Interface of HPC Cluster**

- **Step 2: Run these commands in the terminal:**  
`cd demo_simulators/`
- **Step 3: create directory of your name:**  
`mkdir qadeer`
- **Step 4: Write C code of hello world**  
`nano helloworld.c`
- **Step 5: Create riscv elf format binary using gcc**  
`riscv64-unknown-elf-gcc -o hello helloworld.c`

Summery of the commands above is below:

```
riscv-sim@ucerd-hpc:~$ cd demo_simulators/
riscv-sim@ucerd-hpc:~/demo_simulators$ mkdir qadeer
riscv-sim@ucerd-hpc:~/demo_simulators$ cd qadeer
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ nano helloworld.c
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ riscv64-unknown-elf-gc
riscv64-unknown-elf-gcc      riscv64-unknown-elf-gcc-nm      riscv64-unknown-elf-gcov-dump
riscv64-unknown-elf-gcc-13.2.0  riscv64-unknown-elf-gcc-ranlib  riscv64-unknown-elf-gcov-tool
riscv64-unknown-elf-gcc-ar      riscv64-unknown-elf-gcov
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ riscv64-unknown-elf-gcc -o hello helloworld.c
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$
```

Figure 2: Summary of commands

- **Step 6 : How to run it on spike**

*spike pk ./hello*

- **Step 7: Debugging using spike**

*spike -d pk ./hello*

- **How to debug the code ?**

**Run above command and continue pressing enter button the terminal will show like this:**

```
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ spike -d pk ./hello
(spike)
core  0: 0x00000000000001000 (0x00000297) auipc    t0, 0x0
(spike)
core  0: 0x00000000000001004 (0x02028593) addi     a1, t0, 32
(spike)
core  0: 0x00000000000001008 (0xf1402573) csrr     a0, mhartid
(spike)
core  0: 0x0000000000000100c (0x0182b283) ld       t0, 24(t0)
(spike)
core  0: 0x00000000000001010 (0x00028067) jr       t0
(spike)
core  0: >>>> MEM_START
core  0: 0x00000000080000000 (0x1f80006f) j        pc + 0x1f8
(spike)
core  0: >>>> do_reset
core  0: 0x000000000800001f8 (0x00000093) li      ra, 0
..
```

Figure 3 : Summary of commands

- **Step 8: Run this following command below**

reg 0 // All registers in that core

- **You can check the value of any register after number of clock cycle**

```
(spike) reg 0
zero: 0x0000000000000000 ra: 0x0000000000000000 sp: 0x0000000000000000 gp: 0x0000000000000000
tp: 0x0000000000000000 t0: 0x0000000080000000 t1: 0x0000000000000000 t2: 0x0000000000000000
s0: 0x0000000000000000 s1: 0x0000000000000000 a0: 0x0000000000000000 a1: 0x0000000000001020
a2: 0x0000000000000000 a3: 0x0000000000000000 a4: 0x0000000000000000 a5: 0x0000000000000000
a6: 0x0000000000000000 a7: 0x0000000000000000 s2: 0x0000000000000000 s3: 0x0000000000000000
s4: 0x0000000000000000 s5: 0x0000000000000000 s6: 0x0000000000000000 s7: 0x0000000000000000
s8: 0x0000000000000000 s9: 0x0000000000000000 s10: 0x0000000000000000 s11: 0x0000000000000000
t3: 0x0000000000000000 t4: 0x0000000000000000 t5: 0x0000000000000000 t6: 0x0000000000000000
(spike) reg 0 sp
0x0000000000000000
(spike) □
```

Figure 4 : Check reg value

- **Step 9 : Cache simulation**

*spike --ic=4:8:16 pk ./hello //instruction cache simulation*

*spike --dc=4:8:16 pk ./hello //data cache simulation*

*spike -l2=64:4:64 pk ./hello //l2cache simulation*

*some commands output is shown below further you can run by your own*

```
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ spike --ic=4:8:16 pk ./hello
bbl loader
Hello World
I$ Bytes Read:          921062
I$ Bytes Written:       0
I$ Read Accesses:      328857
I$ Write Accesses:      0
I$ Read Misses:        9076
I$ Write Misses:        0
I$ Writebacks:         0
I$ Miss Rate:          2.760%
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ spike pk ./hello
bbl loader
Hello World
riscv-sim@ucerd-hpc:~/demo_simulators/qadeer$ □
```

Figure 5 : Spike Simulation

## **Tasks**

**Task 1:** Write a c code of swapping variables and debug it on spike. Find the value of reg s2 after 10 clock cycles.

### **Task 2:**

Perform the cache simulation on data cache and find the hit & miss rate of the data cache.

The data cache specification are given below:

- Sets: 16 bytes
- Ways: 4 bytes
- Block size: 2 bytes