Exploring Memory Technology Simulators

Reza Adinepour

Computer Engineering Department, Tehran Ploytechnic

adinepour@aut.ac.ir

May 9, 2024

Memory Technologies - Spring 2024

Amirkabir University of Technology
(Tehran Polytechnic)

Agenda

DRAMSIM

1. DRAMSim use for simulate Dynamic RAMs.
   1.1 DRAM modeling it's very important because the technology is trying to provide CPU and DRAM integrated in one chip.
   1.2 This provides high density:
      1.2.1 High density
      1.2.2 Optimal performance
      1.2.3 Lower power consumption
2. DRAMSim is provide in three version:
   2.1 DRAMSim 1
   2.2 DRAMSim 2
   2.3 DRANSim 3
   In this talk, we discuss about the last version of DRAMSim
3. DRAMSim developed in C++ and write in modularly.

DRAMSIM (Cont.)

1. DRAMSim can be connected to GEM5
2. DRAMSim can simulate following protocol:
   2.1 DDR3
   2.2 DDR4
   2.3 LPDDR3
   2.4 LPDDR4
   2.5 GDDR5
   2.6 GDDR6
   2.7 HBM
   2.8 HMC
   2.9 STT-MRAM

The structure of main block of DRAMSim is shown in figure 1.

DRAMSIM (Cont.)

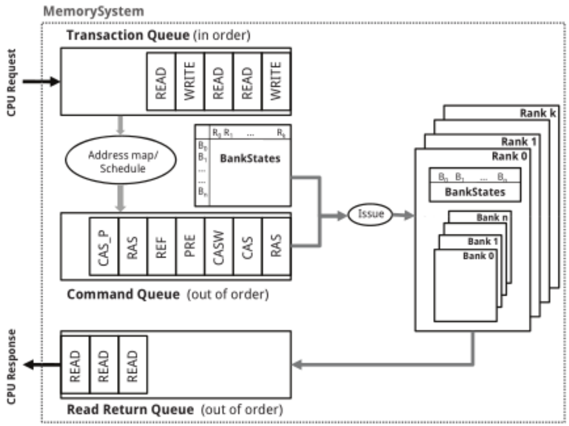


Figure: Main block of DRAMSim

DRAMSIM (Cont.)

Advantages:

1. The possibility of simulating new DRAM technologies like DDR4 and GDDR6
2. High flexibility in configuration
3. Synchronize with system simulators

Disadvantages:

1. Dependence on the model and configuration
2. Don't report power consumption and area

DRAMSIM (Cont.)

How install and build DRAMSim?
We should clone repository in first step:

Clone repository

```
$ git clone https://github.com/umd-memsys/DRAMsim3
$ DRAMsim3cd
```

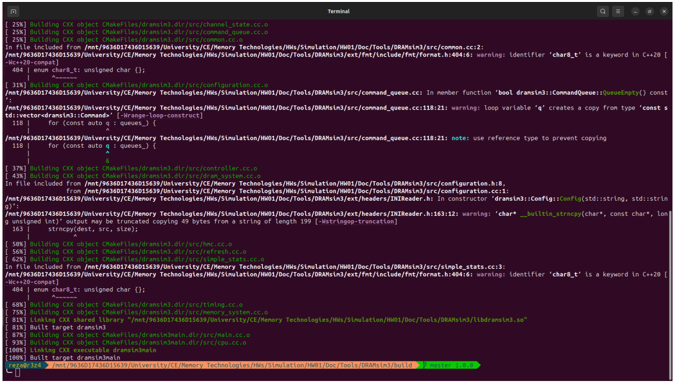
now we should build it:

Build

```
$ mkdir build
$ cd buildcd
$ cmake ..
$ make -j4
$ -DTHERMAL=1.. cmake
```

DRAMSIM (Cont.)

If the simulation builds successfully, you can see **Built target** on your terminal like figure 2



```
[ 25%] Building CXX object CMakeFiles/dramsim3.dir/src/channel_state.cc.o
[ 25%] Building CXX object CMakeFiles/dramsim3.dir/src/command_queue.cc.o
[ 25%] Building CXX object CMakeFiles/dramsim3.dir/src/common.cc.o
In file included from /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/common.cc:2:
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/ext/fmt/include/fmt/format.h:404:6: warning: identifier ‘char8_t’ is a keyword in C++20 [
-Wc++20-compat]
  404 | enum char8_t: unsigned char {};
      |      ^~~~~~~
[ 31%] Building CXX object CMakeFiles/dramsim3.dir/src/configuration.cc.o
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/command_queue.cc: In member function ‘bool dramsim3::CommandQueue::QueueEmpty() const
’:
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/command_queue.cc:118:21: warning: loop variable ‘q’ creates a copy from type ‘const s
td::vector<dramsim3::Command>’ [-Wrange-loop-construct]
  118 |     for (const auto q : queues_) {
      |                     ^
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/command_queue.cc:118:21: note: use reference type to prevent copying
  118 |     for (const auto q : queues_) {
      |                     ^
      |                     &
[ 37%] Building CXX object CMakeFiles/dramsim3.dir/src/controller.cc.o
[ 43%] Building CXX object CMakeFiles/dramsim3.dir/src/dram_system.cc.o
In file included from /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/configuration.h:8,
                 from /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/configuration.cc:1:
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/ext/headers/INIReader.h: In constructor ‘dramsim3::Config::Config(std::string, std::strin
g)’:
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/ext/headers/INIReader.h:163:12: warning: ‘char* __builtin_strncpy(char*, const char*, lon
g unsigned int)’ output may be truncated copying 49 bytes from a string of length 199 [-Wstringop-truncation]
  163 |     strncpy(dest, src, size);
      |            ^
[ 50%] Building CXX object CMakeFiles/dramsim3.dir/src/hmc.cc.o
[ 56%] Building CXX object CMakeFiles/dramsim3.dir/src/refresh.cc.o
[ 62%] Building CXX object CMakeFiles/dramsim3.dir/src/simple_stats.cc.o
In file included from /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/src/simple_stats.cc:3:
/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/ext/fmt/include/fmt/format.h:404:6: warning: identifier ‘char8_t’ is a keyword in C++20 [
-Wc++20-compat]
  404 | enum char8_t: unsigned char {};
      |      ^~~~~~~
[ 68%] Building CXX object CMakeFiles/dramsim3.dir/src/timing.cc.o
[ 75%] Building CXX object CMakeFiles/dramsim3.dir/src/memory_system.cc.o
[ 81%] Linking CXX shared library "/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/libdramsim3.so"
[ 81%] Built target dramsim3
[ 87%] Building CXX object CMakeFiles/dramsim3main.dir/src/main.cc.o
[ 93%] Building CXX object CMakeFiles/dramsim3main.dir/src/cpu.cc.o
[100%] Linking CXX executable dramsim3main
[100%] Built target dramsim3main
reza@r3z4 > /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/DRAMsim3/build > master 1.0.0 >
╰─
```

Figure: DRAMSim built target

DRAMSIM (Cont.)

How can run sample simulation?

1. in first, create a folder for save output file of simulation:

Create output directory

```
$ mkdir output
```

2. then, with this command, run simulation for `sample_trace.txt` config file:

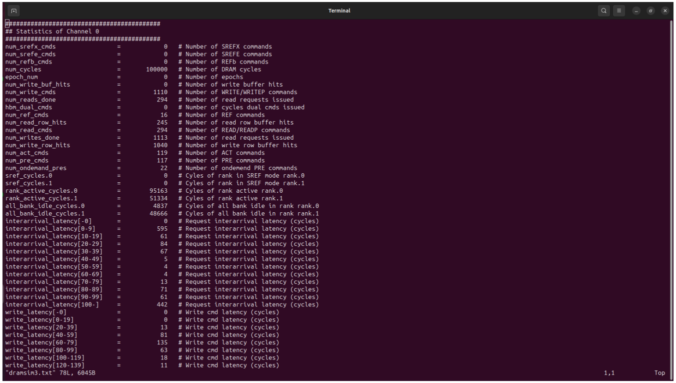
Run simulation

```
$ ./build/dramsim3main configs/DDR4_8Gb_x8_3200.ini
-c 100000 -t tests/example_trace.txt -o output/
```

every various configurations files, located in `configs/` directory. for this simulation we use `DDR4_8Gb` config file.

DRAMSIM (Cont.)

after simulation is finished, you can see output in `output/` directory in `dramsim3.txt` file like bellow:


Terminal

```
###########################################
## Statistics of Channel 0
###########################################
num_srefx_cmds              =          0   # Number of SREFX commands
num_srefe_cmds              =          0   # Number of SREFE commands
num_refb_cmds               =          0   # Number of REFb commands
num_cycles                  =     100000   # Number of DRAM cycles
epoch_num                   =          0   # Number of epochs
num_write_buf_hits          =          0   # Number of write buffer hits
num_write_cmds              =       1110   # Number of WRITE/WRITEP commands
num_reads_done              =        294   # Number of read requests issued
hbm_dual_cmds               =          0   # Number of cycles dual cmds issued
num_ref_cmds                =         16   # Number of REF commands
num_read_row_hits           =        245   # Number of read row buffer hits
num_read_cmds               =        294   # Number of READ/READP commands
num_writes_done             =       1113   # Number of read requests issued
num_write_row_hits          =       1040   # Number of write row buffer hits
num_act_cmds                =        119   # Number of ACT commands
num_pre_cmds                =        117   # Number of PRE commands
num_ondemand_pres           =         22   # Number of ondemend PRE commands
sref_cycles.0               =          0   # Cyles of rank in SREF mode rank.0
sref_cycles.1               =          0   # Cyles of rank in SREF mode rank.1
rank_active_cycles.0        =      95163   # Cyles of rank active rank.0
rank_active_cycles.1        =      51334   # Cyles of rank active rank.1
all_bank_idle_cycles.0      =       4837   # Cyles of all bank idle in rank rank.0
all_bank_idle_cycles.1      =      48666   # Cyles of all bank idle in rank rank.1
interarrival_latency[-0]    =          0   # Request interarrival latency (cycles)
interarrival_latency[0-9]   =        595   # Request interarrival latency (cycles)
interarrival_latency[10-19] =         61   # Request interarrival latency (cycles)
interarrival_latency[20-29] =         84   # Request interarrival latency (cycles)
interarrival_latency[30-39] =         67   # Request interarrival latency (cycles)
interarrival_latency[40-49] =          5   # Request interarrival latency (cycles)
interarrival_latency[50-59] =          4   # Request interarrival latency (cycles)
interarrival_latency[60-69] =          4   # Request interarrival latency (cycles)
interarrival_latency[70-79] =         13   # Request interarrival latency (cycles)
interarrival_latency[80-89] =         71   # Request interarrival latency (cycles)
interarrival_latency[90-99] =         61   # Request interarrival latency (cycles)
interarrival_latency[100-]  =        442   # Request interarrival latency (cycles)
write_latency[-0]           =          0   # Write cmd latency (cycles)
write_latency[0-19]         =          0   # Write cmd latency (cycles)
write_latency[20-39]        =         13   # Write cmd latency (cycles)
write_latency[40-59]        =         81   # Write cmd latency (cycles)
write_latency[60-79]        =        135   # Write cmd latency (cycles)
write_latency[80-99]        =         63   # Write cmd latency (cycles)
write_latency[100-119]      =         18   # Write cmd latency (cycles)
write_latency[120-139]      =         11   # Write cmd latency (cycles)
"dramsim3.txt" 78L, 6045B                                          1,1        Top
```

Figure: Output report

DRAMSIM (Cont.)

we can plot read latancy, interarrival latancy, write latancy and ... with some python scripts located in `script/` directory.

plot

```
$ python3 scripts/plot_stats.py output/dramsim3.json
```

the output of simulation:

DRAMSIM (Cont.)

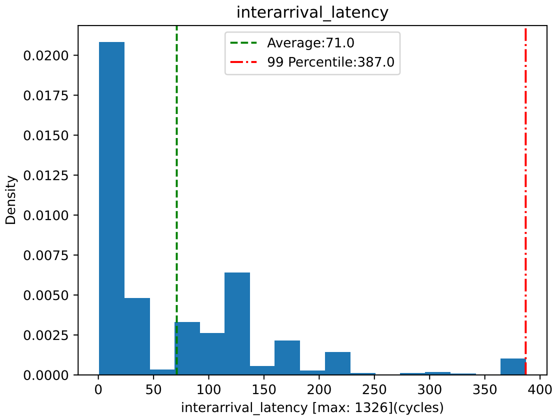


Figure: Interarrival latancy

DRAMSIM (Cont.)

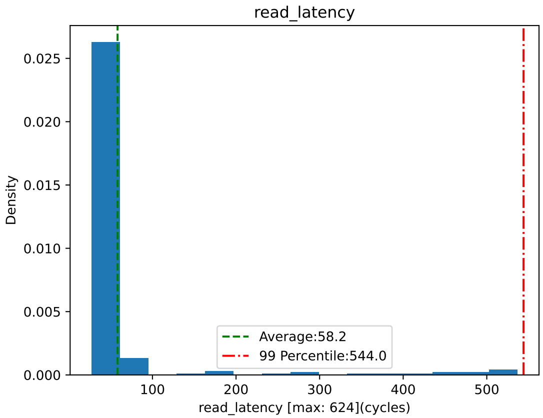


Figure: Read latancy

DRAMSIM (Cont.)

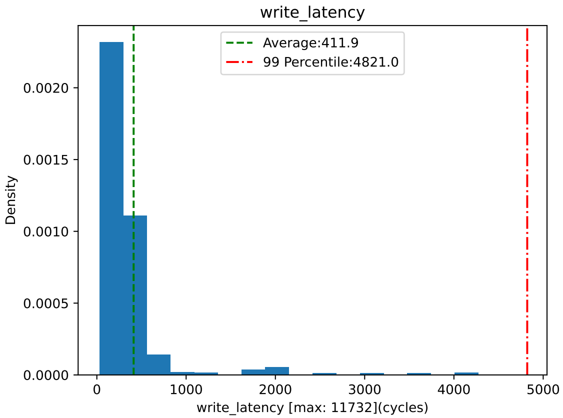


Figure: Write latancy

SimpleScaler

1. This simulator was the doctoral thesis of Mr. Austin Todd from University of Wisconsin, which was written in C language
2. This simulator is not just for memories. like Gem5, it is a system simulator.
3. By default, this simulator is capable of simulating Alpha and PISA ISA. but other ISAs can also be added to it.
4. With SimpleScaler we can simulate this Micro Architecture:
    - 4.1 **Sim-fast:** simulate without considering cache, pipeline and any type of micro architecture
    - 4.2 **Sim-safe:** simulate with considering access to memories
    - 4.3 **Sim-profile:** report number of simulations and dynamic instructions
    - 4.4 **Sim-cache:** simulate a system with access to cache
    - 4.5 **Sim-bpred:** report total branch prediction of program
    - 4.6 **Sim-outorder:** All the previous features are collected in this

SimpleScaler (Cont.)

The structure of SimpleScaler is shown in bellow:

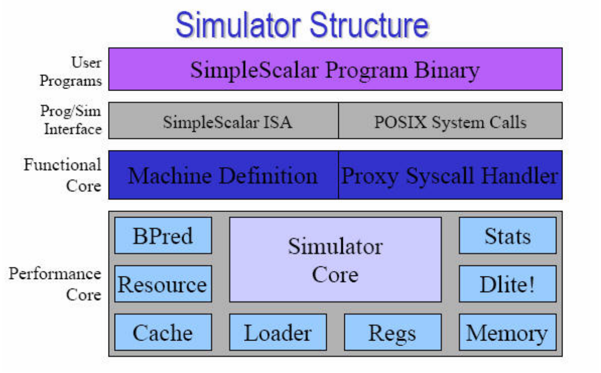


Figure: Structure of SimpleScaler

SimpleScaler (Cont.)

Advantages:

1. Open source
2. System level computer with more detail
3. Support for different architectures

Disadvantages:

1. No direct access to memory
2. Not support a new memory technologies
3. Don't report analysis with detail like `stats` file in GEM5

SimpleScaler (Cont.)

How install and build SimpleScaler?
We should clone repository in first step:

Clone repository

```
$ git clone
https://github.com/stevekuznetsov/simple-scalar.git
$ simple-scalar
```

before build, we need install dependencies:

Install dependencies

```
$ sudo apt-get update
$ sudo apt-get update install build-essential
$ sudo apt-get update install flex bison
$ sudo apt-get update install libx11-dev
```

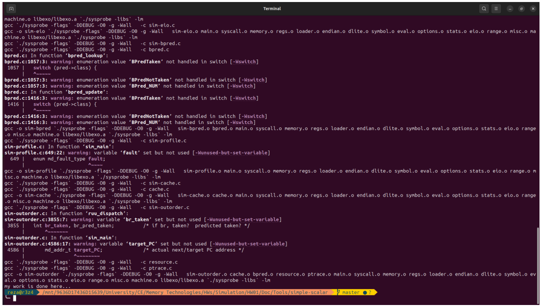
SimpleScaler (Cont.)

Now we can build simulator:

Build

```
$ make config-alpha
```

If the build is successful, your terminal output will look like this:



```
machine.o libexo/libexo.a `./sysprobe -libs` -lm
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c sim-eio.c
gcc -o sim-eio `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-eio.o main.o syscall.o memory.o regs.o loader.o endian.o dlite.o symbol.o eval.o options.o stats.o eio.o range.o misc.o ma
chine.o libexo/libexo.a `./sysprobe -libs` -lm
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c sim-bpred.c
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c bpred.c
bpred.c: In function ‘bpred_lookup’:
bpred.c:1057:3: warning: enumeration value ‘BPredTaken’ not handled in switch [-Wswitch]
 1057 |   switch (pred->class) {
      |   ^~~~~~
bpred.c:1057:3: warning: enumeration value ‘BPredNotTaken’ not handled in switch [-Wswitch]
bpred.c:1057:3: warning: enumeration value ‘BPred_NUM’ not handled in switch [-Wswitch]
bpred.c: In function ‘bpred_update’:
bpred.c:1416:3: warning: enumeration value ‘BPredTaken’ not handled in switch [-Wswitch]
 1416 |   switch (pred->class) {
      |   ^~~~~~
bpred.c:1416:3: warning: enumeration value ‘BPredNotTaken’ not handled in switch [-Wswitch]
bpred.c:1416:3: warning: enumeration value ‘BPred_NUM’ not handled in switch [-Wswitch]
gcc -o sim-bpred `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-bpred.o bpred.o main.o syscall.o memory.o regs.o loader.o endian.o dlite.o symbol.o eval.o options.o stats.o eio.o range
.o misc.o machine.o libexo/libexo.a `./sysprobe -libs` -lm
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c sim-profile.c
sim-profile.c: In function ‘sim_main’:
sim-profile.c:649:22: warning: variable ‘fault’ set but not used [-Wunused-but-set-variable]
  649 |   enum md_fault_type fault;
      |                      ^~~~~
gcc -o sim-profile `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-profile.o main.o syscall.o memory.o regs.o loader.o endian.o dlite.o symbol.o eval.o options.o stats.o eio.o range.o m
isc.o machine.o libexo/libexo.a `./sysprobe -libs` -lm
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c sim-cache.c
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c cache.c
gcc -o sim-cache `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-cache.o cache.o main.o syscall.o memory.o regs.o loader.o endian.o dlite.o symbol.o eval.o options.o stats.o eio.o range
.o misc.o machine.o libexo/libexo.a `./sysprobe -libs` -lm
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c sim-outorder.c
sim-outorder.c: In function ‘ruu_dispatch’:
sim-outorder.c:3855:7: warning: variable ‘br_taken’ set but not used [-Wunused-but-set-variable]
 3855 |   int br_taken, br_pred_taken;          /* if br, taken?  predicted taken? */
      |       ^~~~~~~~
sim-outorder.c: In function ‘sim_main’:
sim-outorder.c:4586:17: warning: variable ‘target_PC’ set but not used [-Wunused-but-set-variable]
 4586 |       md_addr_t target_PC;              /* actual next/target PC address */
      |                 ^~~~~~~~~
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c resource.c
gcc `./sysprobe -flags` -DDEBUG -O0 -g -Wall   -c ptrace.c
gcc -o sim-outorder `./sysprobe -flags` -DDEBUG -O0 -g -Wall   sim-outorder.o cache.o bpred.o resource.o ptrace.o main.o syscall.o memory.o regs.o loader.o endian.o dlite.o symbol.o ev
al.o options.o stats.o eio.o range.o misc.o machine.o libexo/libexo.a `./sysprobe -libs` -lm
my work is done here...
 reza@r3z4  /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/simple-scalar  master ● ?
╰─
```

Figure: Build successful

SimpleScaler (Cont.)

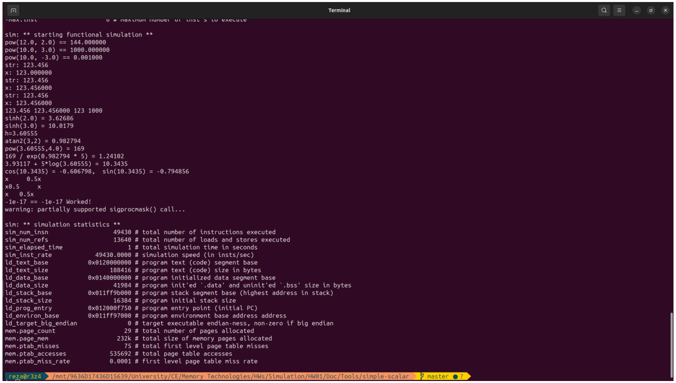
Run simulation:
The default program's .exe file is located in the tests/bin/ path. also the source code of program located in tests/src/ directory. in this simulation we use test-math program. this program calculates sine, tangent and several other mathematical operations for various inputs.
Run simulation with this command:

Build

```
$ ./sim-safe tests/bin/test-math
```

SimpleScaler (Cont.)

The output report of simulation as bellow:



```
-max:inst                    0 # maximum number of inst's to execute

sim: ** starting functional simulation **
pow(12.0, 2.0) == 144.000000
pow(10.0, 3.0) == 1000.000000
pow(10.0, -3.0) == 0.001000
str: 123.456
x: 123.000000
str: 123.456
x: 123.456000
str: 123.456
x: 123.456000
123.456 123.456000 123 1000
sinh(2.0) = 3.62686
sinh(3.0) = 10.0179
h=3.60555
atan2(3,2) = 0.982794
pow(3.60555,4.0) = 169
169 / exp(0.982794 * 5) = 1.24102
3.93117 + 5*log(3.60555) = 10.3435
cos(10.3435) = -0.606798,  sin(10.3435) = -0.794856
x      0.5x
x0.5     x
x   0.5x
-1e-17 == -1e-17 Worked!
warning: partially supported sigprocmask() call...

sim: ** simulation statistics **
sim_num_insn                  49430 # total number of instructions executed
sim_num_refs                  13640 # total number of loads and stores executed
sim_elapsed_time                  1 # total simulation time in seconds
sim_inst_rate            49430.0000 # simulation speed (in insts/sec)
ld_text_base           0x0120000000 # program text (code) segment base
ld_text_size                 188416 # program text (code) size in bytes
ld_data_base           0x0140000000 # program initialized data segment base
ld_data_size                  41984 # program init'ed `.data' and uninit'ed `.bss' size in bytes
ld_stack_base          0x011ff9b000 # program stack segment base (highest address in stack)
ld_stack_size                 16384 # program initial stack size
ld_prog_entry          0x012000f750 # program entry point (initial PC)
ld_environ_base        0x011ff97000 # program environment base address address
ld_target_big_endian              0 # target executable endian-ness, non-zero if big endian
mem.page_count                   29 # total number of pages allocated
mem.page_mem                   232k # total size of memory pages allocated
mem.ptab_misses                  75 # total first level page table misses
mem.ptab_accesses            535692 # total page table accesses
mem.ptab_miss_rate           0.0001 # first level page table miss rate

 reza@r3z4  /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/simple-scalar  master ● ?
```

Figure: Report of `test-math` program

References

Back to start

- S. Senni, *Exploration of non-volatile magnetic memory for processor architecture*, 2015.
- N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "Cacti 6.0: A tool to understand large caches," *University of Utah and Hewlett Packard Laboratories, Tech. Rep*, vol. 147, 2009.
- P. Rosenfeld, E. Cooper-Balis, and B. Jacob, "Dramsim2: A cycle accurate memory system simulator," *IEEE computer architecture letters*, vol. 10, no. 1, pp. 16–19, 2011.
- S. Li, Z. Yang, D. Reddy, A. Srivastava, and B. Jacob, "Dramsim3: A cycle-accurate, thermal-capable dram simulator," *IEEE Computer Architecture Letters*, vol. 19, no. 2, pp. 106–109, 2020.
- D. Wang, B. Ganesh, N. Tuaycharoen, K. Baynes, A. Jaleel,

The End

Questions? Comments?

You can find this slides here:

github.com/M-Sc-AUT/M.Sc-Computer-Architecture/Memory Technologies