Exploring Memory Technology Simulators

Reza Adinepour

Computer Engineering Department, Tehran Ploytechnic

adinepour@aut.ac.ir

March 11, 2024
Memory Technologies - Spring 2024

Amirkabir University of Technology
(Tehran Polytechnic)

Agenda

Introduction

Why we should use simulators?

1. Simulators are vital for understanding computer architecture
2. Two main categories:
   1. Memory simulators
      => focus solely on memory components
   2. Full-system simulators
      => emulate all computer components
3. Efficient design relies on effective simulation tools
4. Comprehensive insights through full-system simulation
5. Maximize performance with accurate simulators

Full-system simulators

Full-system simulators emulate the entire computer system, providing a holistic view. For example, we can refer to the following simulators:

1. GEM5
2. QEMU
3. Bochs
4. SimpleScalar

It's worth noting that a notable and highly regarded emulator in this field is **GEM5**.

Memory simulators

Memory simulators focus on simulating specific memory components. Examples include:

1. **CACTI**
2. **NVSIM**
3. **DRAMSim**
4. DiskSim
5. Ramulator
6. OpenRAM
7. HSPICE

In this talk, we will review the first 3 cases and **SimpleScalar** in the category of Full-system simulators.

CACTI

”In 1993, Dr. Jupi and Dr. Wilton pioneered the first simulation, and CACTI was subsequently developed through HP company tests.”

1. Although this simulator simulates all memory levels, its main use is in the analysis of **Caches**
2. This simulator takes a set of memory parameters as input
3. It calculates various parameters such as **Access time**, **Power**, **Cycle time**, and **Area**
4. CACTI is available in two varieties: Web version and C++ Source code

Next, we will explain how to install and work with the uncompiled version of this emulator

CACTI (Cont.)

Advantages:

1. Open source
2. To be general
3. High speed
4. High flexibility in personalization

Disadvantages:

1. Approximate calculations
2. Productivity gap
3. Not real time
4. It doesn't have a strong community

CACTI (Cont.)

How install and compile CACTI?
In first we should install dependencies.

Install dependencies

```
$ sudo apt-get update
$ sudo apt-get install build-essential
```

After install dependencies we should clone repository.

Clone repository

```
$ git clone
https://github.com/HewlettPackard/cacti.git
```

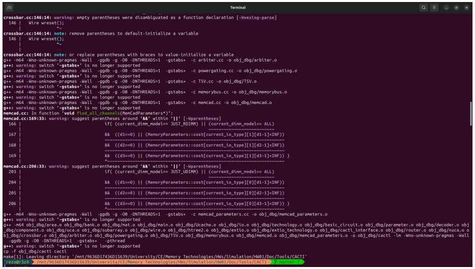
CACTI (Cont.)

Now we build CACTI:

Build

```
$ cd CACTI
$ make
```

After the build is completed, you will see an output like Figure 1

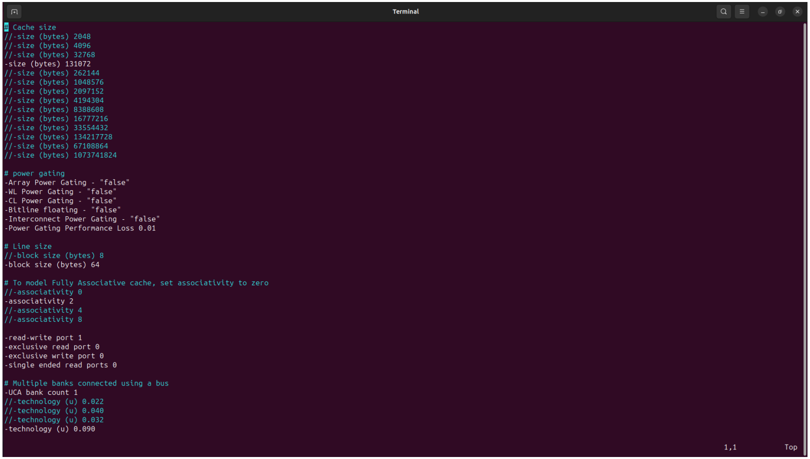


```
Terminal
crossbar.cc:146:14: warning: empty parentheses were disambiguated as a function declaration [-Wvexing-parse]
  146 |   Wire wreset();
      |              ^~
crossbar.cc:146:14: note: remove parentheses to default-initialize a variable
  146 |   Wire wreset();
      |              ^~
      |              --
crossbar.cc:146:14: note: or replace parentheses with braces to value-initialize a variable
g++ -m64 -Wno-unknown-pragmas -Wall  -ggdb -g -O0 -DNTHREADS=1  -gstabs+  -c arbiter.cc -o obj_dbg/arbiter.o
g++: warning: switch '-gstabs+' is no longer supported
g++ -m64 -Wno-unknown-pragmas -Wall  -ggdb -g -O0 -DNTHREADS=1  -gstabs+  -c powergating.cc -o obj_dbg/powergating.o
g++: warning: switch '-gstabs+' is no longer supported
g++ -m64 -Wno-unknown-pragmas -Wall  -ggdb -g -O0 -DNTHREADS=1  -gstabs+  -c TSV.cc -o obj_dbg/TSV.o
g++: warning: switch '-gstabs+' is no longer supported
g++ -m64 -Wno-unknown-pragmas -Wall  -ggdb -g -O0 -DNTHREADS=1  -gstabs+  -c memorybus.cc -o obj_dbg/memorybus.o
g++: warning: switch '-gstabs+' is no longer supported
g++ -m64 -Wno-unknown-pragmas -Wall  -ggdb -g -O0 -DNTHREADS=1  -gstabs+  -c memcad.cc -o obj_dbg/memcad.o
g++: warning: switch '-gstabs+' is no longer supported
memcad.cc: In function 'void find_all_channels(MemCadParameters*)':
memcad.cc:169:33: warning: suggest parentheses around '&&' within '||' [-Wparentheses]
  166 |                               if( (current_dimm_model== JUST_RDIMM) || (current_dimm_model== ALL)
      |                                                                        ~~~~~~~~~~~~~~~~~~~~~~~~~~
  167 |                               &&  ((d1==0) || (MemoryParameters::cost[current_io_type][1][d1-1]<INF))
      |                               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  168 |                               &&  ((d2==0) || (MemoryParameters::cost[current_io_type][1][d2-1]<INF))
      |                               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  169 |                               &&  ((d3==0) || (MemoryParameters::cost[current_io_type][1][d3-1]<INF)) )
      |                               ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
memcad.cc:206:33: warning: suggest parentheses around '&&' within '||' [-Wparentheses]
  203 |                               if( (current_dimm_model== JUST_UDIMM) || (current_dimm_model== ALL)
      |                                                                        ~~~~~~~~~~~~~~~~~~~~~~~~~~
  204 |                               &&  ((d1==0) || (MemoryParameters::cost[current_io_type][0][d1-1]<INF))
      |                               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  205 |                               &&  ((d2==0) || (MemoryParameters::cost[current_io_type][0][d2-1]<INF))
      |                               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  206 |                               &&  ((d3==0) || (MemoryParameters::cost[current_io_type][0][d3-1]<INF)) )
      |                               ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
g++ -m64 -Wno-unknown-pragmas -Wall  -ggdb -g -O0 -DNTHREADS=1  -gstabs+  -c memcad_parameters.cc -o obj_dbg/memcad_parameters.o
g++: warning: switch '-gstabs+' is no longer supported
g++ -m64 obj_dbg/area.o obj_dbg/bank.o obj_dbg/mat.o obj_dbg/main.o obj_dbg/Ucache.o obj_dbg/io.o obj_dbg/technology.o obj_dbg/basic_circuit.o obj_dbg/parameter.o obj_dbg/decoder.o obj
_dbg/component.o obj_dbg/uca.o obj_dbg/subarray.o obj_dbg/wire.o obj_dbg/htree2.o obj_dbg/extio.o obj_dbg/extio_technology.o obj_dbg/cacti_interface.o obj_dbg/router.o obj_dbg/nuca.o o
bj_dbg/crossbar.o obj_dbg/arbiter.o obj_dbg/powergating.o obj_dbg/TSV.o obj_dbg/memorybus.o obj_dbg/memcad.o obj_dbg/memcad_parameters.o -o obj_dbg/cacti -lm -Wno-unknown-pragmas -Wall
  -ggdb -g -O0 -DNTHREADS=1  -gstabs+   -pthread
g++: warning: switch '-gstabs+' is no longer supported
cp -f obj_dbg/cacti cacti
make[1]: Leaving directory '/mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/CACTI'
 reza@r3z4  /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/CACTI  master ? 
```

Figure 1: Successful build

CACTI (Cont.)

You can set cache configs in `cache.cfg` file like figure



```
# Cache size
//-size (bytes) 2048
//-size (bytes) 4096
//-size (bytes) 32768
-size (bytes) 131072
//-size (bytes) 262144
//-size (bytes) 1048576
//-size (bytes) 2097152
//-size (bytes) 4194304
//-size (bytes) 8388608
//-size (bytes) 16777216
//-size (bytes) 33554432
//-size (bytes) 134217728
//-size (bytes) 67108864
//-size (bytes) 1073741824

# power gating
-Array Power Gating - "false"
-WL Power Gating - "false"
-CL Power Gating - "false"
-Bitline floating - "false"
-Interconnect Power Gating - "false"
-Power Gating Performance Loss 0.01

# Line size
//-block size (bytes) 8
-block size (bytes) 64

# To model Fully Associative cache, set associativity to zero
//-associativity 0
-associativity 2
//-associativity 4
//-associativity 8

-read-write port 1
-exclusive read port 0
-exclusive write port 0
-single ended read ports 0

# Multiple banks connected using a bus
-UCA bank count 1
//-technology (u) 0.022
//-technology (u) 0.040
//-technology (u) 0.032
-technology (u) 0.090
                                                          1,1           Top
```

Figure 2: cache config file

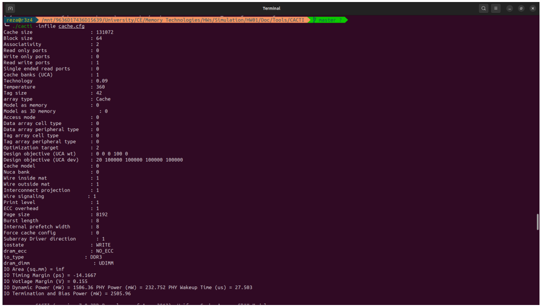
CACTI (Cont.)

Run simulation with this command:

Run

```
$ ./cacti -infile cache.cfg
```

The simulation output is as follows:



```
reza@r3z4  /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/CACTI  master ?
└ ./cacti -infile cache.cfg
Cache size                    : 131072
Block size                    : 64
Associativity                 : 2
Read only ports               : 0
Write only ports              : 0
Read write ports              : 1
Single ended read ports       : 0
Cache banks (UCA)             : 1
Technology                    : 0.09
Temperature                   : 360
Tag size                      : 42
array type                    : Cache
Model as memory               : 0
Model as 3D memory              : 0
Access mode                   : 0
Data array cell type          : 0
Data array peripheral type    : 0
Tag array cell type           : 0
Tag array peripheral type     : 0
Optimization target           : 2
Design objective (UCA wt)     : 0 0 0 100 0
Design objective (UCA dev)    : 20 100000 100000 100000 100000
Cache model                   : 0
Nuca bank                     : 0
Wire inside mat               : 1
Wire outside mat              : 1
Interconnect projection       : 1
Wire signaling               : 1
Print level                   : 1
ECC overhead                  : 1
Page size                     : 8192
Burst length                  : 8
Internal prefetch width       : 8
Force cache config            : 0
Subarray Driver direction       : 1
iostate                       : WRITE
dram_ecc                      : NO_ECC
io_type                     : DDR3
dram_dimm                       : UDIMM
IO Area (sq.mm) = inf
IO Timing Margin (ps) = -14.1667
IO Votlage Margin (V) = 0.155
IO Dynamic Power (mW) = 1506.36 PHY Power (mW) = 232.752 PHY Wakeup Time (us) = 27.503
IO Termination and Bias Power (mW) = 2505.96
```

Figure 3: Output report

NVSIM

1. NVSIM simulator is a tool for analyzing and simulating non-volatile memories
2. It is primarily used for analyzing and estimating the area, power, and energy consumed
3. Unlike CACTI simulator, NVSIM simulator supports the simulation and analysis of new emerging memories like:
    1. PCM (Phase Change Memory)
    2. STT RAM (Spin Torque Transfer RAM)
    3. ReRAM (Resistive RAM)
    4. FBDRAM (Floating Body Dynamic RAM)
    5. eDRAM
4. Developed with C++

NVSIM (Cont.)

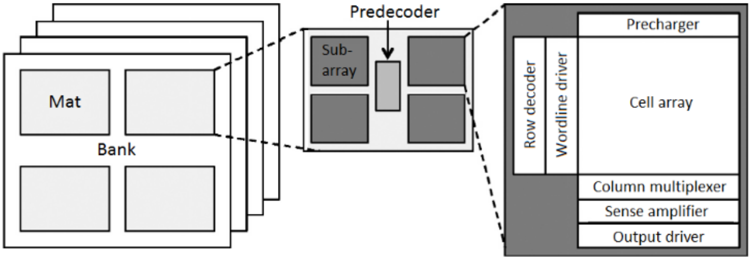


Figure 4: Memory hierarchy in NVSIM

NVSIM (Cont.)

Advantages:

1. Open source
2. Support for the simulation of emerging memories
3. low level Changeability and personalization

Disadvantages:

1. Not real time
2. There is no official version (In this talk i use modified version of simulator)

NVSIM (Cont.)

How install and compile CACTI?

In first clone repository:

clone repository

```
$ git clone
https://github.com/lpentecost/nvsim-merged
```

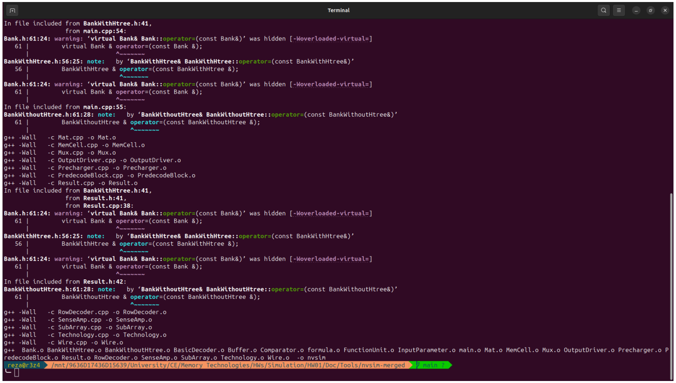
go to repository directory and make it:

build

```
$ cd nvsim-merged
$ make
```

NVSIM (Cont.)

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

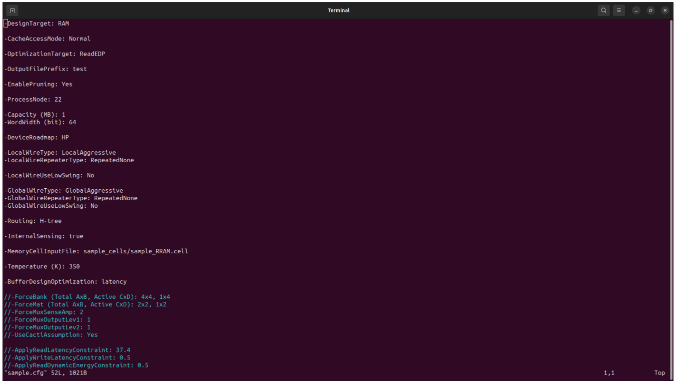


```
In file included from BankWithHtree.h:41,
                 from main.cpp:54:
Bank.h:61:24: warning: ‘virtual Bank& Bank::operator=(const Bank&)’ was hidden [-Woverloaded-virtual=]
   61 |         virtual Bank & operator=(const Bank &);
      |                        ^~~~~~~~
BankWithHtree.h:56:25: note:   by ‘BankWithHtree& BankWithHtree::operator=(const BankWithHtree&)’
   56 |         BankWithHtree & operator=(const BankWithHtree &);
      |                         ^~~~~~~~
Bank.h:61:24: warning: ‘virtual Bank& Bank::operator=(const Bank&)’ was hidden [-Woverloaded-virtual=]
   61 |         virtual Bank & operator=(const Bank &);
      |                        ^~~~~~~~
In file included from main.cpp:55:
BankWithoutHtree.h:61:28: note:   by ‘BankWithoutHtree& BankWithoutHtree::operator=(const BankWithoutHtree&)’
   61 |         BankWithoutHtree & operator=(const BankWithoutHtree &);
      |                            ^~~~~~~~
g++ -Wall   -c Mat.cpp -o Mat.o
g++ -Wall   -c MemCell.cpp -o MemCell.o
g++ -Wall   -c Mux.cpp -o Mux.o
g++ -Wall   -c OutputDriver.cpp -o OutputDriver.o
g++ -Wall   -c Precharger.cpp -o Precharger.o
g++ -Wall   -c PredecodeBlock.cpp -o PredecodeBlock.o
g++ -Wall   -c Result.cpp -o Result.o
In file included from BankWithHtree.h:41,
                 from Result.h:41,
                 from Result.cpp:38:
Bank.h:61:24: warning: ‘virtual Bank& Bank::operator=(const Bank&)’ was hidden [-Woverloaded-virtual=]
   61 |         virtual Bank & operator=(const Bank &);
      |                        ^~~~~~~~
BankWithHtree.h:56:25: note:   by ‘BankWithHtree& BankWithHtree::operator=(const BankWithHtree&)’
   56 |         BankWithHtree & operator=(const BankWithHtree &);
      |                         ^~~~~~~~
Bank.h:61:24: warning: ‘virtual Bank& Bank::operator=(const Bank&)’ was hidden [-Woverloaded-virtual=]
   61 |         virtual Bank & operator=(const Bank &);
      |                        ^~~~~~~~
In file included from Result.h:42:
BankWithoutHtree.h:61:28: note:   by ‘BankWithoutHtree& BankWithoutHtree::operator=(const BankWithoutHtree&)’
   61 |         BankWithoutHtree & operator=(const BankWithoutHtree &);
      |                            ^~~~~~~~
g++ -Wall   -c RowDecoder.cpp -o RowDecoder.o
g++ -Wall   -c SenseAmp.cpp -o SenseAmp.o
g++ -Wall   -c SubArray.cpp -o SubArray.o
g++ -Wall   -c Technology.cpp -o Technology.o
g++ -Wall   -c Wire.cpp -o Wire.o
g++  Bank.o BankWithHtree.o BankWithoutHtree.o BasicDecoder.o Buffer.o Comparator.o formula.o FunctionUnit.o InputParameter.o main.o Mat.o MemCell.o Mux.o OutputDriver.o Precharger.o P
redecodeBlock.o Result.o RowDecoder.o SenseAmp.o SubArray.o Technology.o Wire.o  -o nvsim
 reza@r3z4  /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/nvsim-merged  main ? 
╰─
```

Figure 5: NVSIM successful build

NVSIM (Cont.)

Now we should set the config file like CACTI in `.cfg` file. for simulate simple design we use `samole.cfg` which the config of a 64 bit memristor.



```
-DesignTarget: RAM

-CacheAccessMode: Normal

-OptimizationTarget: ReadEDP

-OutputFilePrefix: test

-EnablePruning: Yes

-ProcessNode: 22

-Capacity (MB): 1
-WordWidth (bit): 64

-DeviceRoadmap: HP

-LocalWireType: LocalAggressive
-LocalWireRepeaterType: RepeatedNone

-LocalWireUseLowSwing: No

-GlobalWireType: GlobalAggressive
-GlobalWireRepeaterType: RepeatedNone
-GlobalWireUseLowSwing: No

-Routing: H-tree

-InternalSensing: true

-MemoryCellInputFile: sample_cells/sample_RRAM.cell

-Temperature (K): 350

-BufferDesignOptimization: latency

//-ForceBank (Total AxB, Active CxD): 4x4, 1x4
//-ForceMat (Total AxB, Active CxD): 2x2, 1x2
//-ForceMuxSenseAmp: 2
//-ForceMuxOutputLev1: 1
//-ForceMuxOutputLev2: 1
//-UseCactiAssumption: Yes

//-ApplyReadLatencyConstraint: 37.4
//-ApplyWriteLatencyConstraint: 0.5
//-ApplyReadDynamicEnergyConstraint: 0.5
"sample.cfg" 52L, 1021B                                                                        1,1          Top
```

Figure 6: `sample.cfg` config file

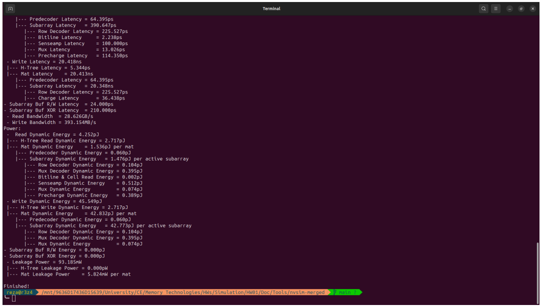
NVSIM (Cont.)

Run simulation:

Run

```
$ ./nvsim sample.cfg
```

output as follow:



Terminal

```
    |--- Predecoder Latency = 64.395ps
    |--- Subarray Latency   = 390.647ps
       |--- Row Decoder Latency = 225.527ps
       |--- Bitline Latency     = 2.238ps
       |--- Senseamp Latency    = 100.000ps
       |--- Mux Latency         = 13.026ps
       |--- Precharge Latency   = 114.350ps
 - Write Latency = 20.418ns
 |--- H-Tree Latency = 5.344ps
 |--- Mat Latency    = 20.413ns
    |--- Predecoder Latency = 64.395ps
    |--- Subarray Latency   = 20.348ns
       |--- Row Decoder Latency = 225.527ps
       |--- Charge Latency      = 36.438ps
- Subarray Buf R/W Latency  = 24.000ps
- Subarray Buf XOR Latency  = 210.000ps
 - Read Bandwidth  = 28.626GB/s
 - Write Bandwidth = 393.154MB/s
Power:
 -  Read Dynamic Energy = 4.252pJ
 |--- H-Tree Read Dynamic Energy = 2.717pJ
 |--- Mat Dynamic Energy    = 1.536pJ per mat
    |--- Predecoder Dynamic Energy = 0.060pJ
    |--- Subarray Dynamic Energy   = 1.476pJ per active subarray
       |--- Row Decoder Dynamic Energy = 0.104pJ
       |--- Mux Decoder Dynamic Energy = 0.395pJ
       |--- Bitline & Cell Read Energy = 0.002pJ
       |--- Senseamp Dynamic Energy    = 0.512pJ
       |--- Mux Dynamic Energy         = 0.074pJ
       |--- Precharge Dynamic Energy   = 0.389pJ
 - Write Dynamic Energy = 45.549pJ
 |--- H-Tree Write Dynamic Energy = 2.717pJ
 |--- Mat Dynamic Energy    = 42.832pJ per mat
    |--- Predecoder Dynamic Energy = 0.060pJ
    |--- Subarray Dynamic Energy   = 42.773pJ per active subarray
       |--- Row Decoder Dynamic Energy = 0.104pJ
       |--- Mux Decoder Dynamic Energy = 0.395pJ
       |--- Mux Dynamic Energy         = 0.074pJ
- Subarray Buf R/W Energy = 0.000pJ
- Subarray Buf XOR Energy = 0.000pJ
 - Leakage Power = 93.185mW
 |--- H-Tree Leakage Power = 0.000pW
 |--- Mat Leakage Power    = 5.824mW per mat

Finished!
reza@r3z4 /mnt/9636D17436D15639/University/CE/Memory Technologies/HWs/Simulation/HW01/Doc/Tools/nvsim-merged  main ?
╰
```

Figure 7: Output of simulation

DRAMSIM

1. DRAMSim use for simulate Dynamic RAMs.
    1. DRAM modeling it's very important because the technology is trying to provide CPU and DRAM integrated in one chip.
    2. This provides high density:
        1. High density
        2. Optimal performance
        3. Lower power consumption
2. DRAMSim is provide in three version:
    1. DRAMSim 1
    2. DRAMSim 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:
    1. DDR3
    2. DDR4
    3. LPDDR3
    4. LPDDR4
    5. GDDR5
    6. GDDR6
    7. HBM
    8. HMC
    9. STT-MRAM

The structure of main block of DRAMSim is shown in figure 8.

DRAMSIM (Cont.)

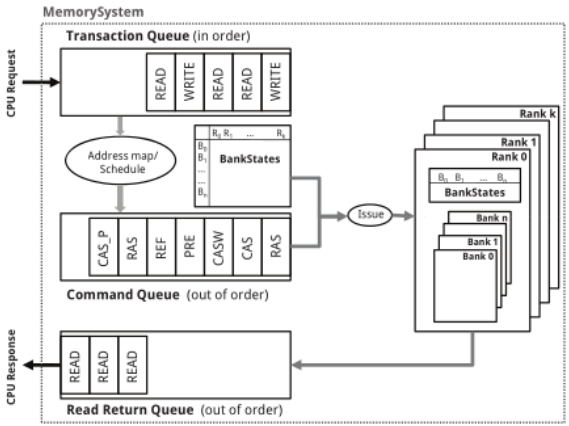


Figure 8: Main block of DRAMSim

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
```

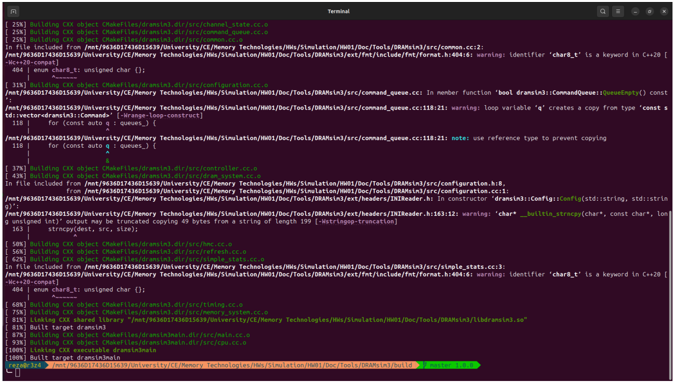
now we should build it:

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 9



```
[ 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 9: 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:

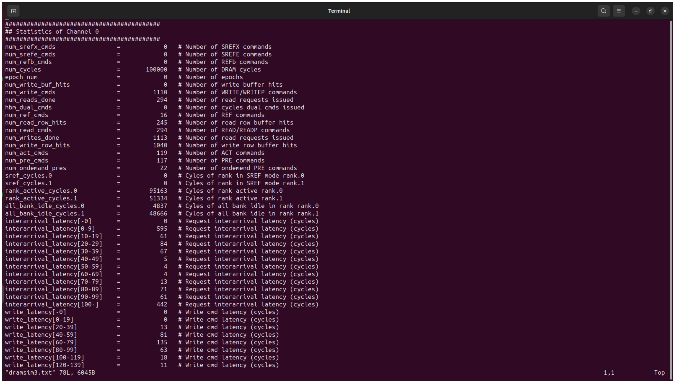
Run simulation

```
$ ./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:



```
###########################################
## 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 10: 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.)

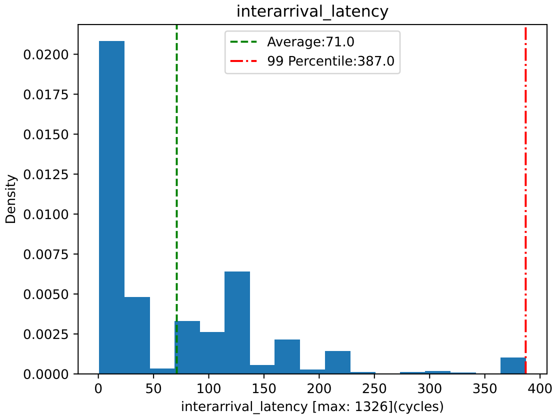


Figure 11: Interarrival latancy

DRAMSIM (Cont.)

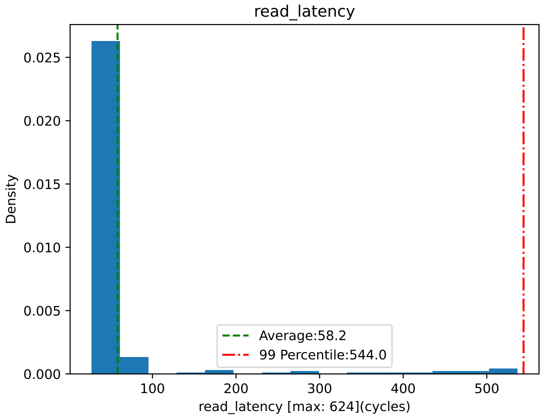


Figure 12: Read latancy

DRAMSIM (Cont.)

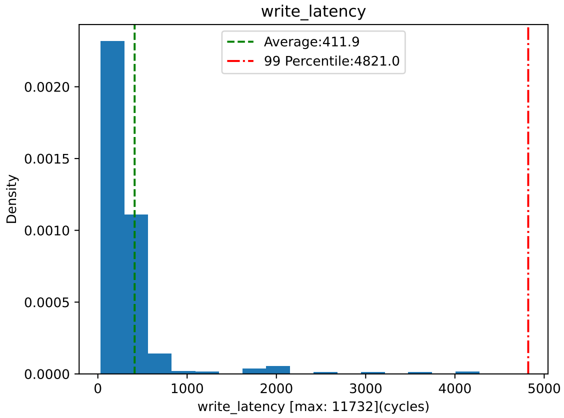


Figure 13: Write latancy

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:
    1. **Sim-fast:** simulate without considering cache, pipeline and any type of micro architecture
    2. **Sim-safe:** simulate with considering access to memories
    3. **Sim-profile:** report number of simulations and dynamic instructions
    4. **Sim-cache:** simulate a system with access to cache
    5. **Sim-bpred:** report total branch prediction of program
    6. **Sim-outorder:** All the previous features are collected in this

SimpleScaler (Cont.)

The structure of SimpleScaler is shown in bellow:

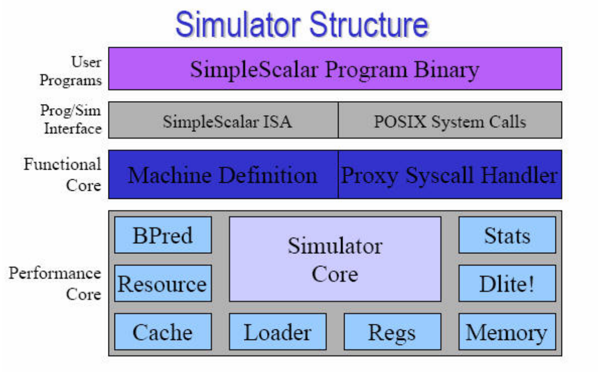


Figure 14: Structure of SimpleScaler

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
```

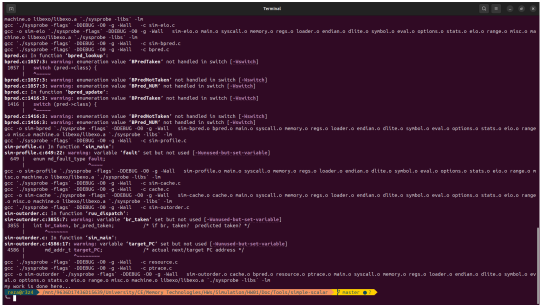
SimpleScaler (Cont.)

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 15: Build successful

SimpleScaler (Cont.)

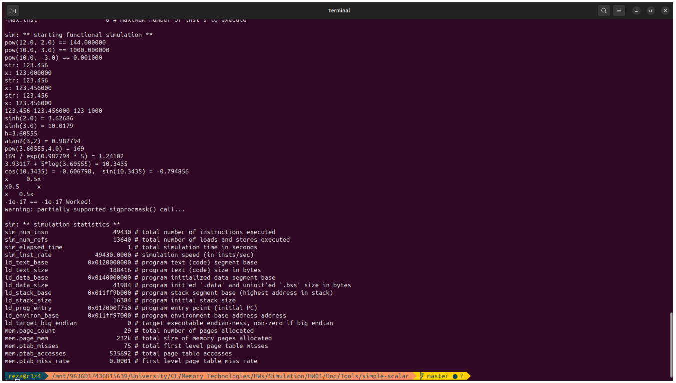
Run simulation:
The default program's `.exe` file is located in the `tests/bin/` path. also the source code of program located in `tests/src/` directory. in this simulation we use `test-math` program. this program calculates sine, tangent and several other mathematical operations for various inputs.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

References

Back to start

[1] S. Senni, *Exploration of non-volatile magnetic memory for processor architecture*, 2015.

[2] 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.

[3] 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.

[4] 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.

The End

Questions? Comments?

You can find this slides here:

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