DRAM simulation using DRAMsim3

DRAMsim3 is a tool to simulate the behavior of DRAMs with different configurations for any given Read/Write trace. It is widely used to estimate the time and energy spent on the data accesses through DRAMs. The tool mimics the memory controller in a host to schedule the read-write commands along with the periodic refreshes. You can read more the simulator here.

In this assignment, you will use DRAMsim3 to generate three example traces and simulate them for DDR4, GDDR6, LPDDR4, and HBM2 configurations. Finally, you will report your observations and answer the questions.

This assignment is worth 10 points and the submission is due on 11:59 pm ET on Feb 24, 2025.

Your tasks

Cloning the simulator, building, an example run

1) Please clone DRAMsim3 using the command below
    a. git clone https://github.com/umd-memsys/DRAMsim3.git
    b. cd DRAMsim3
2) Build as described in the README using -
    a. mkdir build
    b. cd build
    c. cmake ..
    d. make -j (builds dramsim3 library and executable)
3) Run DRAMsim3
    a. ./build/dramsim3main configs/DDR4_8Gb_x8_3200.ini -c 100000 -t tests/example.trace
        DRAM configuration: Details about DRAM configuration (number of ranks, banks, rows, columns, etc.) and DRAM timing parameters (tRC, tCCD, etc.). Table 15.2 in the “Memory Systems Cache DRAM Disk” textbook has a quick summary of DRAM timing parameters (in case you would like to refer).
        DRAM trace: Read-Write commands (with addresses and cycle #)
    b. This will create dramsim3.txt, dramsim3.json files as output. You can go through the output files to get familiar with the parameters that are modeled by DRAMsim3. *You don’t need to submit these.*

Trace generation and simulation

4) Copy assignment_configs/* (provided in tar file) to DRAMsim3/assignment_configs/
   a. There are many configurations in the DRAMsim3/configs/ directory. However, you will use the following configurations for this assignment:
      1. DDR4_8Gb_x8_3200.ini
      2. GDDR6_8Gb_x16.ini
      3. HBM2_8Gb_x128.ini
      4. LPDDR4_8Gb_x16_2400.ini
5) **[1 point]** Generate traces in DRAMsim3/assignment_traces/
   a. cd DRAMsim3/
   b. module load miniforge (to use python on CS portal servers)
   c. **Random Access pattern**: python scripts/trace_gen.py -s **r** -f dramsim3 -o assignment_traces -n 10000 -g 8
   d. **Streaming Access pattern**: python scripts/trace_gen.py -s **s** -f dramsim3 -o assignment_traces -n 10000 -g 8
   e. **Mix**: python scripts/trace_gen.py -s **m** -f dramsim3 -o assignment_traces -n 10000 -g 8
6) **[1 point]** Simulate for all the configs in DRAMsim3/assignment_configs for all the traces in DRAMsim3/assignment_traces:
   a. Sample command: ./build/dramsim3main -c 100000 -o sim/random -t assignment_traces/dramsim3_random_i10_n1000_rw2.trace assignment_configs/LPDDR4_8Gb_x16_2400.ini
      This will simulate a random-access trace on LPDDR4 for 100000 cycles, the output directory is sim/random/.
   b. Create separate directories for random, stream, and mix trace (as the outputs will be overwritten for a DRAM configuration)
   c. You can also use simulate.sh (provided in tar file).

Output Analysis

7) The simulations will create json and txt output files for each case. You can use json2csv.py (provided in tar file) to extract parameters required for this assignment into a csv file.
   a. Command: python json2csv.py -i sim/mix sim/random sim/stream -o sim.csv
8) **[4 points]** Compare the following for all the DRAM configurations in 4) a. for *random* access pattern
   a. average_bandwidth (GBps)
   b. average_power (mW)
   c. total_energy (pJ)
   d. average_read_latency (cycles)

   Please list your observations and insights.

9) **[2 points]** Compare the difference in *num_act_cmds* and *num_pre_cmds* for random, mix, stream accesses for all configs. Why do you think there is a huge difference in the number of activate and precharge commands for different access patterns?

10) **[1 point]** What kind of DRAM would you use for the following cases and why?
    a. a high throughput system
    b. a power-constraint system

11) **[1 point]** Observe the read and write latency histograms after running the following:
    a. Command: python scripts/plot_stats.py sim/mix/DDR4_8Gb_x8_3200.json -d sim/mix -o DDR4
    b. This will create histograms (in pdf files).
    c. Why do you think the read and write latencies are variable for different requests and not constant?

Contents of the tar file:

1) You are provided with 3_dramsim.tar.gz which contains:
    a. assignment_configs/*.ini (four DRAM configuration files)
    b. simulate.sh
    c. json2csv.py