CS2610: Computer Organization and Architecture Lab
Assignment #6

Due Date: 24th Mar, 2022

Objective: To understand the internals of DRAM row-buffer management.

Questions:

1. Implement a Hybrid-Row-Buffer Management Policy, known as Adaptive-Page Management Policy. The details of policy are explained below:
    a. Use a 4-bit saturation counter, and choose two threshold values, namely, High-Threshold and Low-Threshold, with a difference between them not more than 6 and not less than 4.
    b. Initially, set the counter to somewhere in the middle of the range between high and low thresholds and start with Open-Page policy.
    c. switch from Open-Page to Close-Page policy when the count is greater than High-Threshold and increasing, and from Close-Page to Open-Page policy when the count is less than Low-Threshold and decreasing.
    d. If currently in Open-Page policy, and a page-hit is observed, no action to be taken.
    e. If currently in Open-Page policy, and a page-miss occurs, increment the counter.
    f. If currently in Close-Page policy, and if a page-hit with the last closed page occurs, then decrement the counter.
    g. If currently in Close-Page policy, and a page-miss occurs, no action to be taken.
2. Compare the number of cycles (output) reported by USIMM for the all the traces run with Open-Page, Close-Page, Adaptive-Page management policies. Report the best policy.
3. Compare the DRAM performance (in cycles) for the two address mapping schemes, provided in USIMM, for the best row buffer management policy determined in previous part of the question.

Make a table containing performance reported in number of DRAM cycles (as reported by simulator) for the above questions. Give a brief summary of the implementation. Provide appropriate justifications (necessary).

Instructions:

- We use the USIMM simulator for this assignment.
- You can download the simulator (version 1.3) from here: https://www.cs.utah.edu/~rajeev/jwac12/
- Instructions on how to get started are clearly mentioned in README file.
- To model a quad-core system and execute a benchmark in rate-mode on that system, specify the same benchmark four times on the command line.
  Ex: to execute 'libq' in rate mode on a quad-core system run the following command:
  ```
  bin/usimm input/4channel.cfg input/libq input/libq input/libq input/libq
  ```
- Use 4 channel configuration and 4Gb x8 devices for all the simulations (devices can be specified in main.c).