Optimizing for PIM Architectures

Siddharth Sahay, Angela Li

March 2020

1 Group Information

- Siddharth Sahay – ssahay2@andrew.cmu.edu
- Angela Li – qingyanl@andrew.cmu.edu

2 URL

`https://sidsahay.github.io/optimizing-for-pim/`

3 Project Description

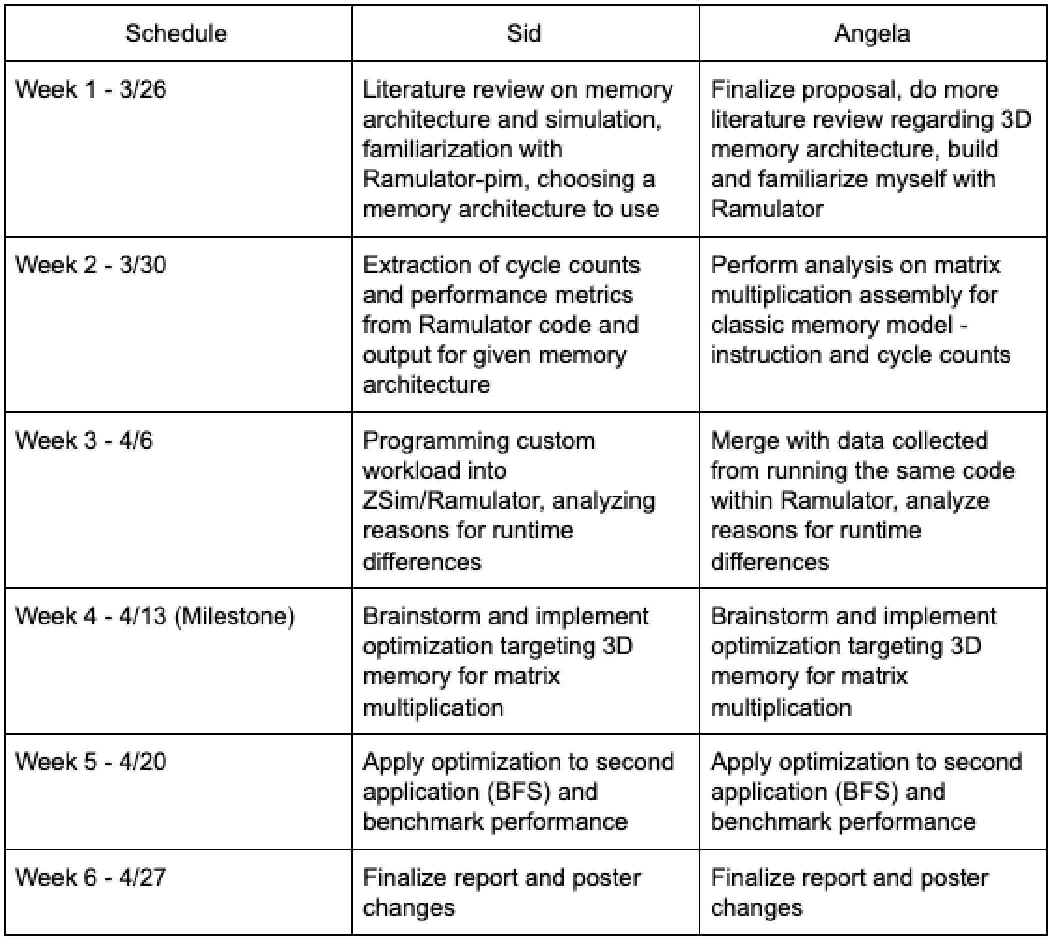
There is a tight coupling between the compiler and the underlying microarchitecture to produce code that executes efficiently. While current compilers, like GCC and LLVM, are targeted towards general purpose CPUs and GPUs, modern computing trends like large scale machine learning rely heavily on the memory system and could benefit greatly from architectures that minimize the von Neumann bottleneck. These PIM (processing-in-memory) architectures, like stacked 3D memory with inbuilt compute, may require special optimizations within the compiler to fully take advantage of the underlying hardware.

Project Goals: We have selected matrix multiplication as our base application, however, this may be subject to change if we find a better representative benchmark.

- 75% goal: Perform detailed comparison between executing on bare-metal vs. the 3D stacked memory emulator, i.e. answering why the runtimes are different and how the memory model influences the execution of each instruction, for our base application.
- 100% goal: Perform the analysis above and integrate one optimization pass that might improve the runtime with the 3D stacked memory, for our base application.
- 125% goal: Apply the optimization to a second application with different memory access patterns compared to the base, like BFS, to see whether the optimization generalizes.

4 Logistics

4.1 Plan and Schedule



Schedule	Sid	Angela
Week 1 - 3/26	Literature review on memory architecture and simulation, familiarization with Ramulator-pim, choosing a memory architecture to use	Finalize proposal, do more literature review regarding 3D memory architecture, build and familiarize myself with Ramulator
Week 2 - 3/30	Extraction of cycle counts and performance metrics from Ramulator code and output for given memory architecture	Perform analysis on matrix multiplication assembly for classic memory model - instruction and cycle counts
Week 3 - 4/6	Programming custom workload into ZSim/Ramulator, analyzing reasons for runtime differences	Merge with data collected from running the same code within Ramulator, analyze reasons for runtime differences
Week 4 - 4/13 (Milestone)	Brainstorm and implement optimization targeting 3D memory for matrix multiplication	Brainstorm and implement optimization targeting 3D memory for matrix multiplication
Week 5 - 4/20	Apply optimization to second application (BFS) and benchmark performance	Apply optimization to second application (BFS) and benchmark performance
Week 6 - 4/27	Finalize report and poster changes	Finalize report and poster changes

4.2 Milestone

By the milestone, we feel like we should be able to reach our 75% goal of explaining the difference between runtimes between the classic RAM model and the 3D stacked memory model.

There is no step between this stage and having an evaluation for an optimization pass, which is why this is our 75% goal.

4.3 Literature Review

Current papers we have looked into, by topic:

- PIM architectures:
  - Processing-in-memory: A workload-driven perspective
    `http://users.ece.cmu.edu/~saugatag/papers/19ibmjrd_pim.pdf`
  - A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing
    `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.4771&rep=rep1&type=pdf`
  - Ramulator: A Fast and Extensible DRAM Simulator
    `users.ece.cmu.edu/~omutlu/pub/ramulator_dram_simulator-ieee-cal15.pdf` item A 3D-Stacked Logic-in-Memory Accelerator for Application-Specific Data Intensive Computing
    `https://users.ece.cmu.edu/~bakin/pub/3dic13.pdf`

4.4 Resources Needed

We will be using Ramulator, an open-source hardware emulator, for finding the cycle counts for individual instructions in a 3D memory-stacked architecture. As of now, we do not foresee the need for any additional hardware.

4.5 Getting Started

Currently, we mostly have been reading up on literature in the areas of PIM and 3D memory. We also have Ramulator running locally and have been familiarizing ourselves with that.