Exposing Processing-in-Memory Capabilities to the Operating System

- Advisor: Lenny Yu Liang

Background.

Processing-in-Memory (PIM) is a promising approach for reducing the growing cost of data movement in data-intensive workloads, including AI and large language model inference. However, different PIM architectures provide different computation and memory capabilities, such as supported operations, memory-access restrictions, execution granularity, and data-placement constraints. These properties can directly affect operating-system decisions, yet current operating systems have no common way to discover and reason about them. Project Goal. This project investigates which PIM hardware capabilities need to be exposed to the operating system and how they can be represented through a lightweight interface. The goal is not to design a new PIM architecture, but to identify a minimal set of OS-visible PIM capabilities that can support hardware-aware operating-system decisions.

Tasks.

- Study a small set of representative PIM architectures and their software/hardware interfaces.
- Identify PIM properties that directly affect OS decisions, such as supported operations, memory accessibility, execution granularity, and data-placement constraints.
- Classify these properties into a minimal set of OS-visible PIM capabilities.
- Design a lightweight interface for exposing these capabilities to the operating system.
- Implement a prototype using Linux/QEMU or an existing PIM simulation platform.
- Demonstrate the interface with one representative OS use case.

Requirements.

Good programming skills in C/C++ and basic knowledge of operating systems and computer architecture. Familiarity with Linux is expected. Prior experience with Linux kernel development, QEMU, Ramulator, memory systems, or PIM is helpful but not required. What You Will Learn. You will gain hands-on experience with Linux, memory systems, PIM architectures, hardware/software interfaces, and system prototyping.