범용성과 효율성 높은 PIM 아키텍처 시뮬레이션 방법론

2024. 1. 5.

작성자

서울대학교 이헌준

.

목차

1. 연구 목표 및 계획

본 연구팀의 목표는 다양한 형태의 Process in Memory (PIM) 아키텍처를 빠르고 범용성 있게 시뮬레이션할 수 있는 CPU 기반 시뮬레이터를 설계하는 것이다. 이를 위해서 본 연구는 다양한 PIM 아키텍처를 범용성 있는 계층 구조로 추상화하고, 각 계층의 구조를 재구성할 수 있는 형태로 추상화된 시뮬레이터를 설계한다. 그리고 모든 하드웨어 모듈을 각 시뮬레이션 Cycle 마다 시뮬레이션 하는 Clock-Driven 시뮬레이션 방식이 아닌 시뮬레이션 트레이스에 맞춰서 특정 이벤트가 발생하는 경우에만 시뮬레이션을 수행하는 Event-Driven 시뮬레이션 방식을 도입하여 시뮬레이터의 성능을 비약적으로 상승시킨다.

추가로, 본 연구팀은 최근 크게 주목을 받고 있는 수직 적층 구조의 디바이스를 활용한 아날로그 PIM 아키텍처의 성능을 정확하게 모델링 하기 위해서 3D NAND Flash를 활용한 PIM 아키텍처의 성능 관련 파라미터를 예측할 수 있는 방법론을 제안한다.

2. 범용성 있는 시뮬레이터 설계 방법론

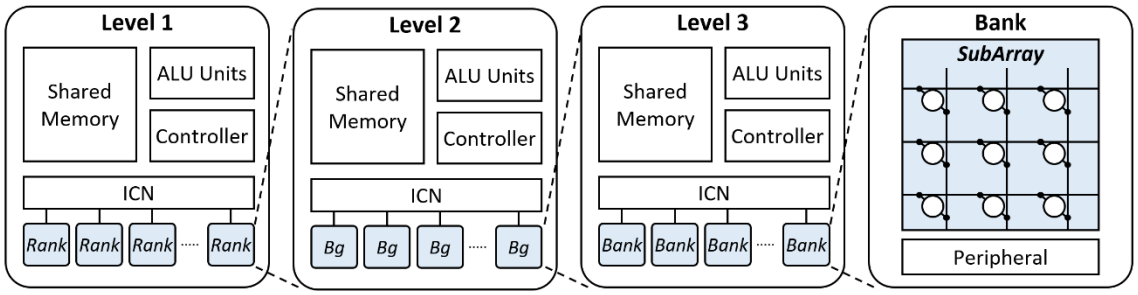

그림 1. 본 연구팀이 제안하는 범용성 있는 계층 구조 시뮬레이션의 추상적인 도식

PIM 아키텍처는 연산을 처리하는 방법론에 따라 크게 3가지 카테고리로 나뉘게 된다. 먼저 Process-Near Memory (PNM)의 경우 메모리의 최상위 계층 (e.g., DIMM)에 연산기를 배치시켜 메모리와 프로세서 사이 Bandwidth를 절약하는 역할을 한다. Digital PIM (D-PIM)의 경우 일반적으로 메모리 어레이 (Memory Bank) 옆에 프로세서를 배치시켜 여러 Bank에서 읽어오는 데이터를 병렬적으로 처리한다. 마지막으로 Analog PIM (A-PIM)의 경우 메모리 어레이 내부적으로 연산을 수행하여 데이터를 읽는 과정과 동시에 아날로그 기반 연산을 수행하여 PIM 연산의 에너지 효율성을 높이고 특정 연산의 경우 병렬성을 추가로 높인다.

따라서 본 연구팀은 기존 메모리 시뮬레이터의 계층 구조를 활용하고 [1], 각 계층 구조에 선택적으로 Shared Memory Unit, ALU Unit, Controller 등의 모듈을 생성할 수 있도록 시뮬레이터를 설계한다 (그림 1). 이와 같은 시뮬레이션 프레임워크는 유저가 시뮬레이션의

타겟이 되는 아키텍처에 맞춰서 각 메모리 계층에 알맞은 형태의 모듈을 생성하고 임의의 시뮬레이션 방식에 대한 성능을 시뮬레이션할 수 있도록 하게 한다.

예를 들어, DIMM-Level에서 Embedding Reduction과 같은 연산을 수행할 수 있도록 설계된 TensorDIMM 같은 아키텍처를 시뮬레이션하기 위해, 유저는 가장 위의 계층에 속하는 DIMM에 Shared Memory로 Register File을 생성하고, ALU Unit으로 Vector ALU Unit을 생성하여 전체적인 성능을 예측할 수 있게 된다. 반면 최근 삼성에서 개발하는 HBM-PIM을 시뮬레이션하기 위해서 시뮬레이터의 Bank 마다 Shared Memory로 Register File을 생성하고 ALU Unit으로 Vector ALU Unit을 생성하면 된다. 마지막으로 ReRAM을 기반으로 동작하는 ISAAC과 같은 Analog PIM의 경우는 여러 Bank가 합쳐진 Core와 여러 Core가 합쳐진 Tile이 독립적으로 동작할 수 있도록 Controller를 생성하고 각 Bank는 Matrix-Vector Multiply를 수행할 수 있도록 주변회로와 같이 생성된다.

3. 고성능 시뮬레이터 설계 방법론

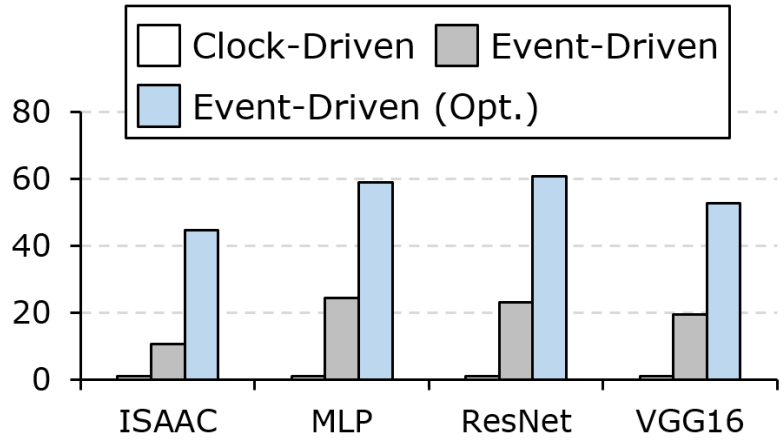

그림 2. 본 연구팀이 구현한 시뮬레이터 최적화 방법론을 통한 성능 향상

하지만, 본 연구팀이 개발한 기본 시뮬레이션 방법론은 매 시뮬레이션 Cycle 마다 전체 메모리 계층에 존재하는 모듈을 시뮬레이션 해야 되기 때문에 성능이 매우 떨어지게 된다. 기존 메모리 시뮬레이션과 다르게 PIM 아키텍처를 시뮬레이션 하기 위해서는 각 메모리 모듈 사이의 통신을 시뮬레이션 할 수 있어야 된다. 이를 지원하기 위해서 기존 PIM 시뮬레이터는 전체 모듈을 매 시뮬레이션 Cycle 마다 변화를 연산하는 Cycle-Accurate 시뮬레이션 방식을 도입한다. 하지만, 이와 같은 Cycle-Accurate 시뮬레이션 방식은 대규모 PIM 아키텍처를 시뮬레이션 하는 경우 전체 시뮬레이션 성능이 떨어지게 된다.

따라서 본 연구팀은 PIM의 각 모듈을 시뮬레이션 하는 과정에서 특정 이벤트가 발생하는 경우에만 특정 모듈에 대한 연산을 수행하는 Event-Driven 시뮬레이션 방식을 도입하고 Event-Driven 시뮬레이션을 수행하는 데이터 구조를 수정하여 고성능 PIM 시뮬레이션을

가능하게 한다. 시뮬레이션은 크게 일반 모듈을 시뮬레이션 하는 부분과, 모듈에 대한 연산을 스케쥴링 해주는 스케쥴러로 이루어져 있다. 스케쥴러는 먼저 각 시뮬레이션 Cycle 마다 시뮬레이터가 관련한 연산을 수행해야 되는 모듈을 가지고 있다. 이를 바탕으로 일반 시뮬레이션 관련 코드는 해당 모듈들에 대한 연산을 수행한다. 연산을 처리하는 과정에서 시뮬레이터는 크게 (1) 해당 모듈이 다음 연산을 처리해야 되는 시간을 계산하고 (e.g., 특정 instruction에 대한 연산이 끝나는 Cycle) (2) 해당 모듈을 처리하는 과정에서 다른 모듈에 영향을 주는 시간 (e.g., NoC를 통해서 다른 Module에 데이터가 전달되는 시간) 관련 정보를 스케쥴러에 전달한다. 이와 같은 방법론을 통해서 시뮬레이터는 전체 모듈에 대한 연산을 하지 않고 시뮬레이션 시간을 크게 단축시킨다. 또한 본 연구팀은 스케쥴러를 최적화된 데이터구조로 관리하여 전체 시뮬레이션 성능을 추가적으로 높인다.

그림 2는 본 연구팀이 개발한 시뮬레이션 방식을 통해서 얻을 수 있는 성능 향상을 보여준다. Clock-Driven은 기본 베이스라인 시뮬레이터, Event-Driven은 데이터구조 최적화 없이 동작하는 Event-Driven 시뮬레이터, Event-Drive (Opt)는 추가적으로 데이터구조 최적화까지 진행한 시뮬레이터를 나타낸다. 본 연구팀의 분석 결과 Event-Driven 시뮬레이션은 평균 19.4배의 성능 향상을 얻고 데이터구조 최적화를 통해 Event-Driven (Opt)는 2.8배 추가적인 성능 향상을 얻는다.

4. 수직 적층 구조 디바이스 성능 모델링 방법론

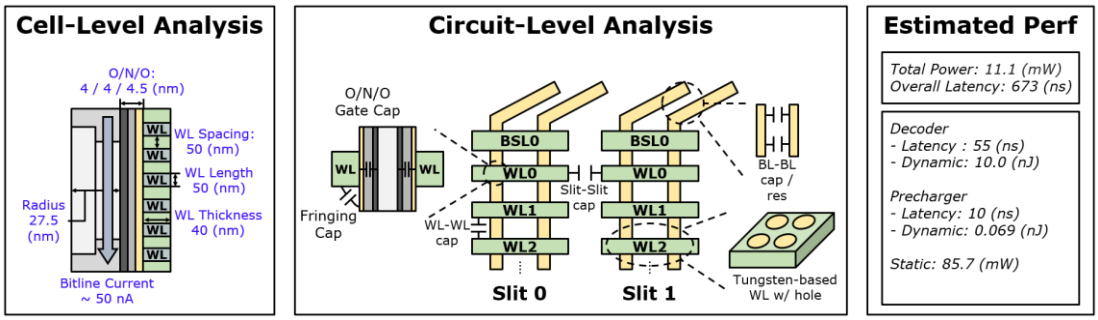

그림 3. 본 연구팀이 개발한 적층형 구조의 성능 시뮬레이션 방법론

마지막으로 본 연구팀은 3차원 구조의 3D NAND Flash를 바탕으로 아날로그 기반 PIM 연산을 수행하는 경우 발생하는 에너지와 시간을 예측하기 위한 회로 레벨 시뮬레이터를 개발한다. 개발한 회로 레벨 시뮬레이터는 앞서 개발한 시뮬레이터를 돌리는데 필요한 에너지 및 시간 관련 파라미터를 제공한다. 이와 같이 개발한 시뮬이터는 크게 2가지 구조로 나뉘어 있다. 먼저, Cell-Level 분석과 관련된 부분은 실제 3D NAND Flash의 내부 셀의 구조를 바탕으로 Parasitic Capacitance와 Resistance를 계산한다. 그리고 반면 Circuit-Level 분석은 이와 같이 계산된 Capacitance와 Resistance를 바탕으로 RC Delay와 CV^2를 바탕으로 하는 에너지 소모량을 계산한다. 이를 위해 본 연구팀은 기존 Storage관련 Circuit-

Level 시뮬레이터인 NVSim을 수정하여 사용한다.

5. 결론

최근 삼성, 하이닉스를 포함한 메모리 회사들을 기반으로 다양한 PIM 기반 아키텍처가 개발되고 있다. 하지만, 이와 같은 아키텍처는 메모리 계층에서 연산기가 배치된 위치, 연산 방법론등에 따라서 상이한 에너지 효율성과 연산 처리 속도를 가지게 된다. 따라서 다양한 구조의 PIM 아키텍처를 범용성있고 빠르게 처리할 수 있는 시뮬레이터가 필수적이다. 본 연구팀이 제안하는 계층 구조를 가지는 Event-Driven 시뮬레이션 프레임워크는 이와 같은 문제를 효과적으로 해결하여 차세대 PIM 아키텍처의 개발에 크게 기여할 것으로 기대된다.