PIM 기반 가속기 아키텍처를 활용한 AI 연산의 정확성 및 신뢰성 개선

*허예리, 정의영
연세대학교 전기전자공학과
e-mail: hyl5543@yonsei.ac.kr, eychung@yonsei.ac.kr

Improving accuracy and reliability of AI calculations using PIM-based accelerator architecture

*Yeri Heo, Eui-Young Chung
Yonsei University Department of Electrical and Electronic Engineering

Abstract

This paper explores the implementation and enhancement of Processing-In-Memory (PIM) architectures, specifically focusing on the Analog PIM for applications requiring high accuracy, such as automotive driver-assistance systems (ADAS). The necessity for precise and real-time data processing within vehicle Advanced Processors (AP) highlights the critical role of PIM technologies, which significantly improve data processing speeds through simultaneous activation of multiple rows for memory-intensive operations like Multiply-Accumulate (MAC) and General Matrix to Vector Multiplication (GEMV). However, this leads to increased susceptibility to errors due to the continuous memory access and thermal effects on transistor switching.

To address these challenges, this study introduces a robust Error Correction Code (ECC) algorithm tailored for Analog PIM systems to enhance accuracy and reliability in AI computations. The effectiveness of the ECC algorithm is evaluated through a System-C based simulator that models the physical parameters of actual circuits, providing insights into hardware configurations and operational efficiency.

The implementation of ECC has shown to reduce the accuracy degradation from 24.37% to 2.5% under defect conditions at the 10,000th iteration, signifying a 21.8% improvement in system reliability. Moreover, this research extends to examine the impact of varying ECC application intervals on the accuracy, underscoring the balance between system reliability and performance. The findings suggest that ECC parameters must be carefully considered alongside hardware specifications such as area, energy consumption, and processing time to optimize PIM deployment in real-world applications.

I. 서론

일반적인 컴퓨팅 시스템에서 AI 응용 프로그램의 경우 근소한 accuracy의 차이는 치명적이지 않을 수 있지만, 자동 운전 보조 시스템(ADAS) 기술에서는 주변 환경을 정확하게 인식하고 짧은 순간에 적절한 반응을 요구한다. 이러한 요구 사항은 ADAS 기술에서 AI 연산의 높은 accuracy를 필수적으로 만들며, 이는 차량 내부에서 실시간으로 데이터를 처리하여 운전 지원 시스템이나 자율 주행 기능을 지원하는 차량용 AP에서 매우 중요한 요소이다. 차량용 AP는 일반적으로 edge-device로 분류되며, 이러한 장치에서는 빠르고 효율적인 데이터 처리를

위해 처리 속도의 개선과 에너지 효율의 극대화가 필요하기 때문에 PIM 사용이 유용하다.

PIM 아키텍처는 Analog PIM과 Digital PIM으로 나뉘며, 특히 Analog PIM은 모든 셀을 사용하여 weight를 저장하고 한 번에 다수의 Row를 활성화하여 MAC, GEMV 연산을 수행함으로써 데이터 처리 속도를 크게 향상시킬 수 있다. 그러나 이 과정에서 발생하는 연속적인 메모리 접근은 트랜지스터의 반복적인 스위칭으로 인한 전자기장 및 열 발생으로 인접한 셀에 영향을 주게 되고, 이는 저장된 weight의 정보가 변질될 가능성을 높인다.

Digital PIM은 메모리에서 데이터를 읽은 후 오류 정정 코드(ECC)를 거쳐 Processing 로직에 사용되지만, Analog PIM은 같은 메모리 어레이에서 MAC, GEMV 명령이 실행되며 저장된 데이터로 직접 연산이 진행되기 때문에 Digital PIM과는 다른 접근 방식이 필요하다.
본 연구는 Analog PIM 구조의 AI 가속기를 모델링하고 이에 적합한 ECC 보안 알고리즘을 개발하여 AI 연산의 accuracy를 높이기 위해 시도된다. 시뮬레이터는 System-C로 구성되어 실제 회로의 물리적인 파라미터를 입력 받아 해당 하드웨어와 유사한 결과를 제공할 수 있다.

II. 본론

1. Processing in Memory

Processing In Memory(PIM)는 데이터를 프로세서로 이동시키는 대신 메모리 내에서 직접 데이터를 처리하는 기술을 말한다. 이 접근 방식은 데이터 전송에 필요한 시간과 에너지를 절약함으로써 전체 시스템의 성능과 에너지 효율을 향상시키는 데 기여한다. 특히, Analog PIM은 이러한 컴퓨팅 인 메모리 아키텍처의 한 형태로, 고속 연산과 에너지 효율성에서 큰 이점을 가지고 있다.

Analog PIM은 아날로그 신호를 활용하여 메모리 셀 내에서 직접 연산을 수행한다. 이 구조는 전통적인 디지털 방식의 메모리와 달리, 메모리 셀 자체가 연산 기능을 갖추고 있어 처리 과정에서 데이터 이동이 필요하지 않다. 다음은 Analog PIM의 주요 동작 원리이다.

Weight 저장: Analog PIM 구조에서는 각 메모리 셀이 신경망의 가중치(weight)를 저장한다. 이 가중치는 아날로그 형태의 전하(Charge)로서 메모리 셀에 저장되기도 하며, 정수 비트로 소수점을 표현해 가중치를 저장하기도 한다.
Row 활성화: 데이터 처리를 위해 특정 row를 활성화한다. 활성화된 row의 셀들은 입력 데이터와 결합되어 연산에 참여하게 된다.
병렬 처리: Analog PIM은 여러 row를 동시에 활성화시켜 병렬적으로 데이터를 처리할 수 있다. 이를 통해 MAC 연산과 같은 복잡한 행렬 연산을 빠르게 수행할 수 있다.
MAC 및 GEMV 연산: 입력 데이터와 가중치가 저장된 셀이 활성화되면, 해당 셀들은 곱셈과 누적 연산(MAC)을 수행한다. 또한, 일반 행렬과 벡터 곱셈(GEMV) 연산도 이와 유사한 방식으로 처리된다.

Analog PIM은 이러한 고속의 데이터 처리 능력 덕분에 차량 내 ADAS와 같이 실시간 데이터 처리가 필수적인 응용 분야에서 특히 유용하게 사용될 수 있다.

2. System Architecture

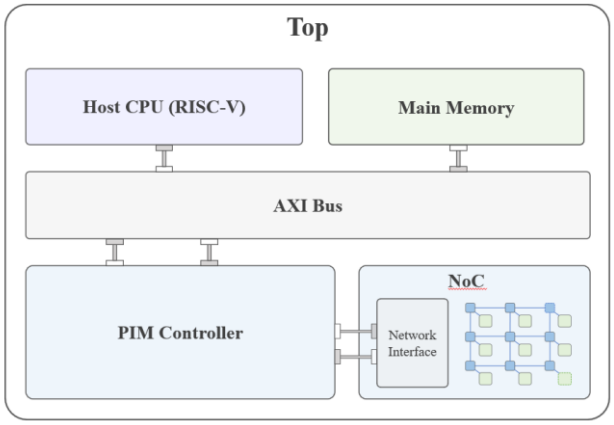

그림 1. Top Architecture

모델링한 PIM 기반의 System Architecture이다.

Host CPU: open source인 RISC-V를 사용한다. RISC-V는 모듈성이 뛰어나기 때문에 사용자가 필요에 따라 ISA를 확장할 수 있어 시스템의 유연성을 높일 수 있다.
Main Memory: Synopsys의 Platform Architect에서 제공하는 라이브러리인 DDR IP를 사용한다.
PIM Controller: RISC-V에서 발생하는 신경망 연산과 관련된 명령을 수신하여 NoC에 전달하는

컨트롤러 역할을 수행한다. 또한 Direct Memory Access(DMA) 기능을 통해 메인 메모리에 접근하여 각 연산에 필요한 데이터를 읽어와 NoC에 전달한다. 이 Controller는 RISC-V의 명령 정보를 효율적으로 처리하여, Data flow를 최적화한다.

NoC: PIM 컨트롤러에서 전달받은 정보를 기반으로 라우터를 통해 PIM core에 명령을 전달하고 연산을 수행한다. NoC는 PIM core 사이의 모든 데이터 통신을 관리하며, 고속 데이터 전송과 낮은 지연시간을 보장하여 전체적인 시스템 성능을 극대화하는 역할을 한다.

RISC-V와 Main Memory, PIM Controller는 AXI4 Bus로 연결된다.

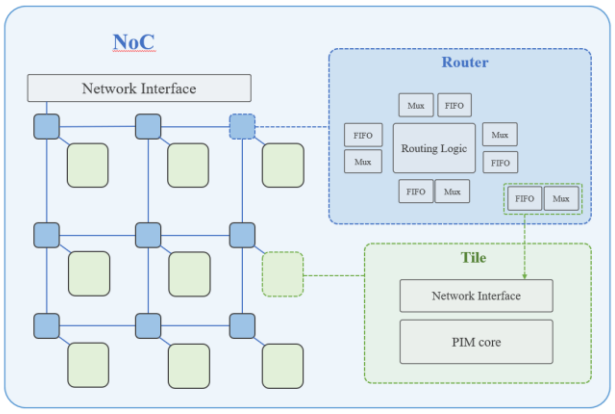

그림 2. NoC Architecture

그림 2는 NoC의 전체 아키텍처로, 기본적인 2D Mesh 토폴로지로 구성된다.

Network Interface(NoC): PIM 컨트롤러에서 전달받은 정보를 패킷 형태로 변환한다. 패킷은 Op 코드, 소스 정보, 목적지 정보, 데이터 길이, PIM 주소, 라우터 방향 등의 정보로 구성된다.
Router: 생성된 패킷 정보를 기반으로 패킷을 라우팅한다.

Tile: 라우터에서 전달받은 패킷에서 Op 코드와 데이터를 전달받아 PIM core에서 연산을 수행한다. 이 과정에서 Analog PIM의 메모리 셀에 대한 ECC가 진행된다.

Ⅲ. 구현

3.1 ECC Design

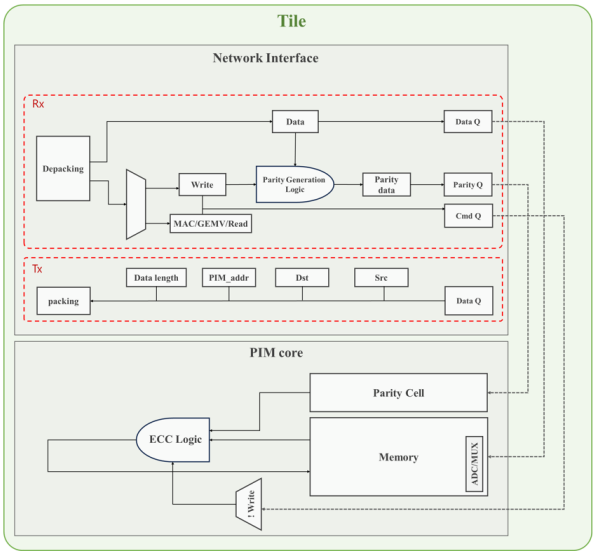

그림 3. PIM Tile 내부의 Parity & ECC Logic

Network Interface (Tile): 라우터에서 전달받은 패킷을 분해하여 Op 코드와 데이터를 PIM 코어로 전달한다. Op 코드가 Write일 경우 신경망 모델의 가중치를 저장한다는 의미이다. 이때 가중치의 원본 데이터에 대한 패리티 비트를 생성하여 PIM 코어 내의 패리티 셀에 저장한다. 또한, PIM 코어에서 연산된 결과 데이터와 부가 정보를 패킹 하여 라우터로 전달한다.

PIM core: 전달받은 Op 코드에 따라 Write, Read, MAC, GEMV 연산을 수행한다. 가중치의 원본 데이터에 대한 패리티 비트는 패리티 셀에 저장하고, 가중치와 입력 데이터는 메모리 셀에 저장한다. Op 코드가 Write일 경우 가중치를 저장하는 과정이므로 ECC 로직은 활성화되지 않는다. 그러나 MAC이나 GEMV 연산의 경우 가중치를 사용하기 때문에 메모리에 저장된 가중치에 대한 ECC를 수행한다.

3.2 ECC Algorithm

Algorithm 1 parity generation algorithm

```
1: function parity_gen(data, index)
2:     Initialize n = DATA_SIZE(Byte) * 8
3:         Initialize p_bits = parity_bit num
4:             Loop i from 0 to p_bits - 1
5:                 Calculate parity_position = 2^i
6:                 Initialize count = 0
7:                     Loop j from 1 to n
8:                     If j & parity_position is true, increment count based on data bit at position
9:                     End Loop
10:                If count is odd, update parity or actual_parity based on ECC status
11:            End Loop
12: end function
```

그림 4. Parity Generation Algorithm

그림 4는 원본 데이터에서 패리티 비트를 생성하는 알고리즘을 보여준다. System-C로 구현되어 있으며, 로직은 함수 단위로 구성된다.
원본 데이터를 입력 받아 각 데이터 위치에 대한 패리티 비트를 생성한다.

Algorithm 2 ECC algorithm

```
1: function ECC_func(data, parity)
2:     Initialize n, total_segments, bytes_per_segment
3:         Seed random number generator
4:         If total_index meets condition, induce errors at random positions
5:             Loop segment from 0 to total_segments - 1
6:                 Reset actual_parity and set ECC mode
7:                 Call parity_gen for current data segment
8:                 Calculate parity difference
9:                 If there is a parity error, correct the error at calculated position
10:            End Loop
11:        Increment total_index
12: end function
```

그림 5. ECC Algorithm

메모리 셀에 대한 Read, MAC, GEMV 명령이 PIM 코어로 전달되면 해당 함수가 호출되어 일정 크기의 데이터와 패리티 셀에 저장된 패리티 비트를 입력 받는다. 현재 저장된 가중치로 새로운 패리티 비트를 생성하고, 이를 기존 패리티 비트와 비교하여 오류가 발생한 위치를 찾아 수정한다.

3.3 Evaluation

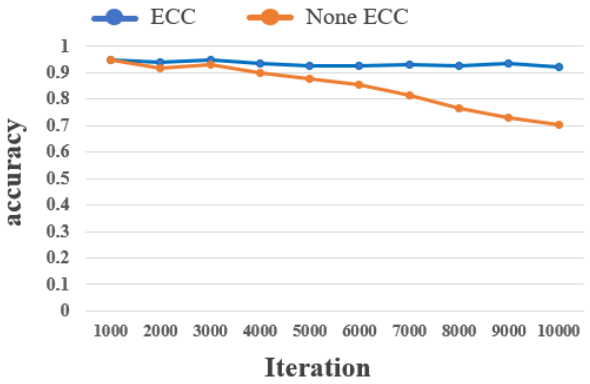

그림 6. Normalized Accuracy Based on Iteration

그림 6은 MLP 연산의 반복(Iteration)에 따른 정규화 된 정확도를 보여준다. 반복 횟수의 증가는 저장된 가중치의 재사용 빈도를 높이고, 결과적으로 PIM 메모리 셀의 접근 횟수도 증가시킨다. 이 과정에서 발생하는 메모리 셀의 결함을 모델링한 결과, ECC를 적용하지 않은 경우 10,000번째 반복에서 초기 정확도 0.948이 0.7043으로 약 24.37% 감소했다.

반면, ECC를 적용한 경우 동일한 조건에서 정확도는 0.923으로 감소하여 약 2.5%의 감소에 불과했다. 이는 ECC 알고리즘을 적용함으로써 PIM 기반 AI 연산의 정확도가 21.8% 향상된 효과를 얻을 수 있음을 시사한다.

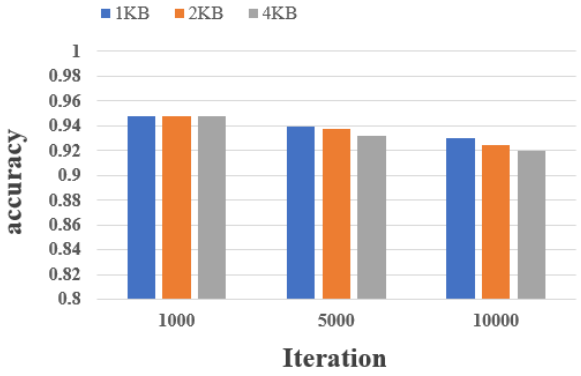

그림 7. Normalized accuracy based on ECC data range and Iteration

그림 7에서는 전체 PIM 메모리 셀의 수를 78,400개로 모델링하여 ECC 적용 간격에 따른 accuracy 변화를 조사하였다. 분석 결과, 10,000번째 iteration에서 1KB 간격으로 ECC를 적용한 경우 accuracy는 0.93으로 약 1.8% 감소하였다. 또한, 2KB 간격에서는 0.925로 2.3%의 감소를, 4KB 간격에서는 0.92로 2.8%의 감소를 보였다. 이는 ECC 데이터 간격이 AI 연산의 accuracy에 미치는 영향을 명확히 보여준다.

Ⅳ. 결론 및 향후 연구 방향

본 논문에서는 ADAS 기술과 같이 높은 accuracy를 요구하는 AI 응용 분야에서 Analog PIM의 신뢰성을 향상하는 방법론을 제시하였다. Analog PIM 구조를 활용함으로써 데이터 처리 속도는 현저히 향상되었으나, 연속적인 메모리 접근이 유발하는 에러의 가능성 또한 증가하는 문제에 직면하였다. 이를 해결하기 위해, 본 연구는 효율적인 ECC 보안 알고리즘을 개발하여 Analog PIM 시스템의 에러 최소화 및 데이터 무결성 보장 방안을 마련하였다.

System-C를 활용한 시뮬레이션을 통해 실제 하드웨어의 물리적 특성을 모델링하고 다양한 공정 정보, 셀 어레이, 메모리 스펙 정보를 입력 변수로 사용하여 하드웨어 구성에 따른 성능을 평가하였다. 실험 결과, ECC 알고리즘 적용 후 시스템의 정확성이 약 20% 향상되었음을 확인하였다. 이는 Analog

PIM이 실시간 고 정밀 연산을 요구하는 분야에서의 활용 가능성을 크게 높이며, 해당 기술의 상용화 가능성을 제시한다. 또한 ECC 적용 간격이 시스템의 신뢰성 및 성능에 미치는 영향을 실험적으로 확인하였다. 이러한 연구 결과는 ECC에 관한 여러 파라미터가 실제 하드웨어의 면적, 에너지 소비, 처리 시간과 함께 고려되어야 함을 시사한다.

Ⅴ. 사사

이 논문은 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2022-0-00050, 데이터 플로우 구조 기반 PIM의 실행 및 프로그래밍 모델 개발)

본 연구는 산업통상자원부(1415181094)와 KSRC 지원 사업인 미래반도체소자 원천기술개발사업(20019456)의 연구결과로 수행되었음

참고문헌

[1] Boraten, Travis, and Avinash Karanth Kodi. "Packet security with path sensitization for NoCs." 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2016. Intel Corporations, Intel StrongArm SA-1110 Microprocessor Develper's Manual, June 2000.

[2] Mestiri, Hassen, Yahia Salah, and Achref Addali Baroudi. "A secure network interface for on-chip systems." 2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA). IEEE, 2020.

[3] Cılasun, Hüsrev, et al. "Error Detection and Correction for Processing in Memory (PiM)." arXiv preprint arXiv:2207.13261 (2022).

[4] Zubair, Kazi Abu, et al. "FAT-PIM: Low-Cost Error Detection for Processing-In-Memory." arXiv preprint arXiv:2207.12231 (2022).

[5] Lv, Minjie, et al. "Efficient repair analysis algorithm exploration for memory with redundancy and in-memory ECC." IEEE Transactions on Computers 70.5 (2020): 775-788.