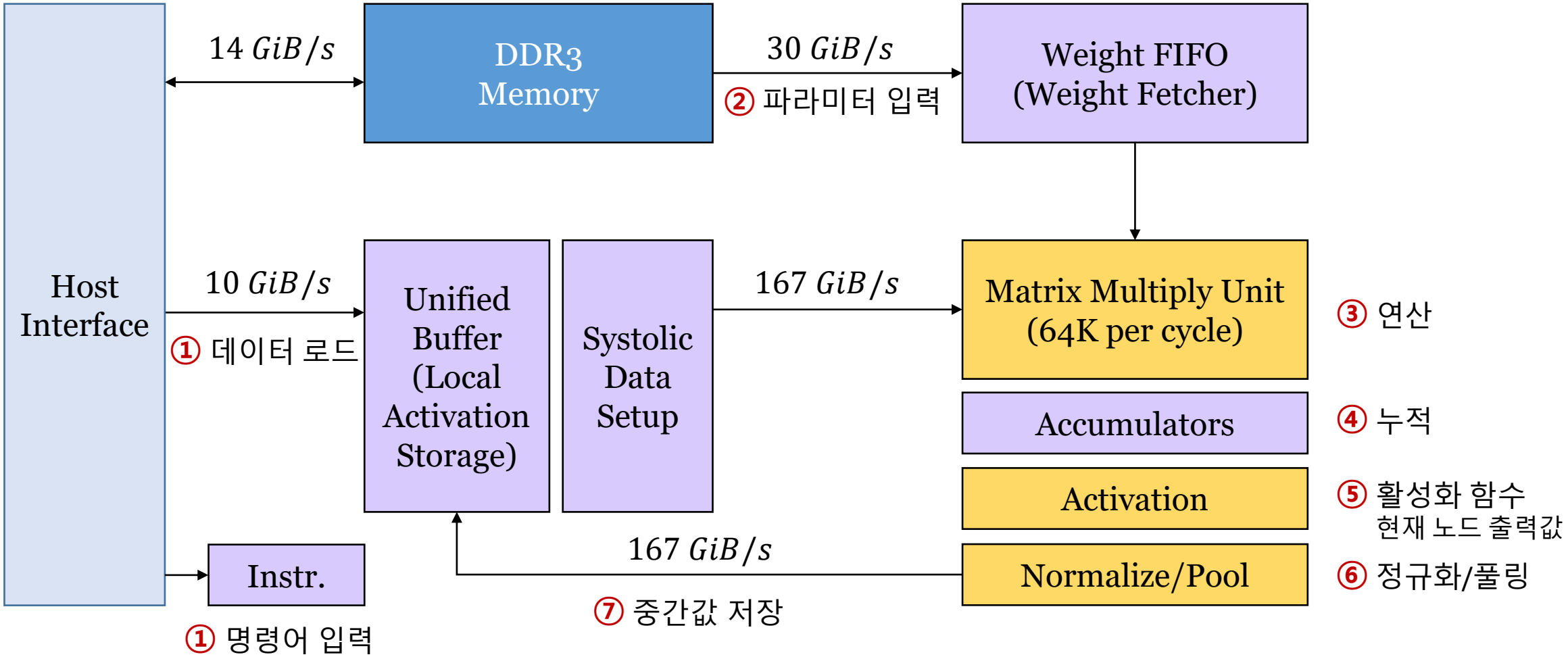
딥러닝 하드웨어 가속기

https://youtu.be/qxCcruOkvMQ

Contents

CryptoCraft LAB

Tensor Processing Unit (TPU)



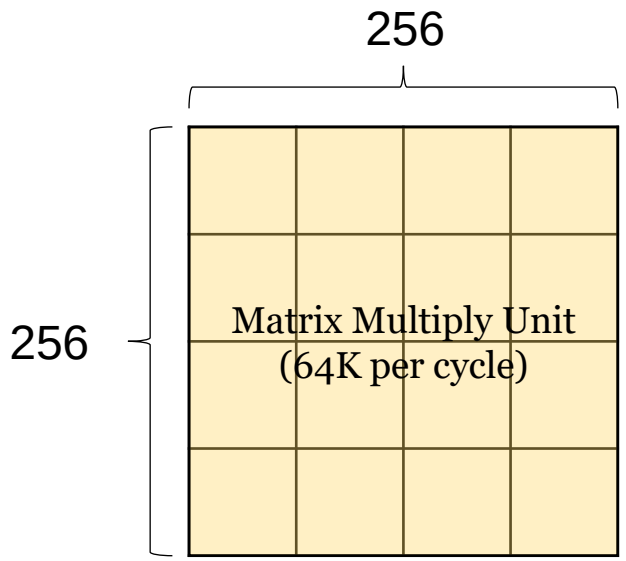
*weight : 추론 시, 고정된 상수이므로 미리 DDR3 메모리에 로드

Matrix Multiply Unit

- 256x256 개의 곱셈기로 구성
- 전체 곱셈기에 operand를 모두 공급하는 것은 대역폭에 부담

  → systolic array
- 추론의 경우 8-bits 부동소수점 연산으로 충분
  - 8-bits입력 2개 / 16-bits 출력 1개

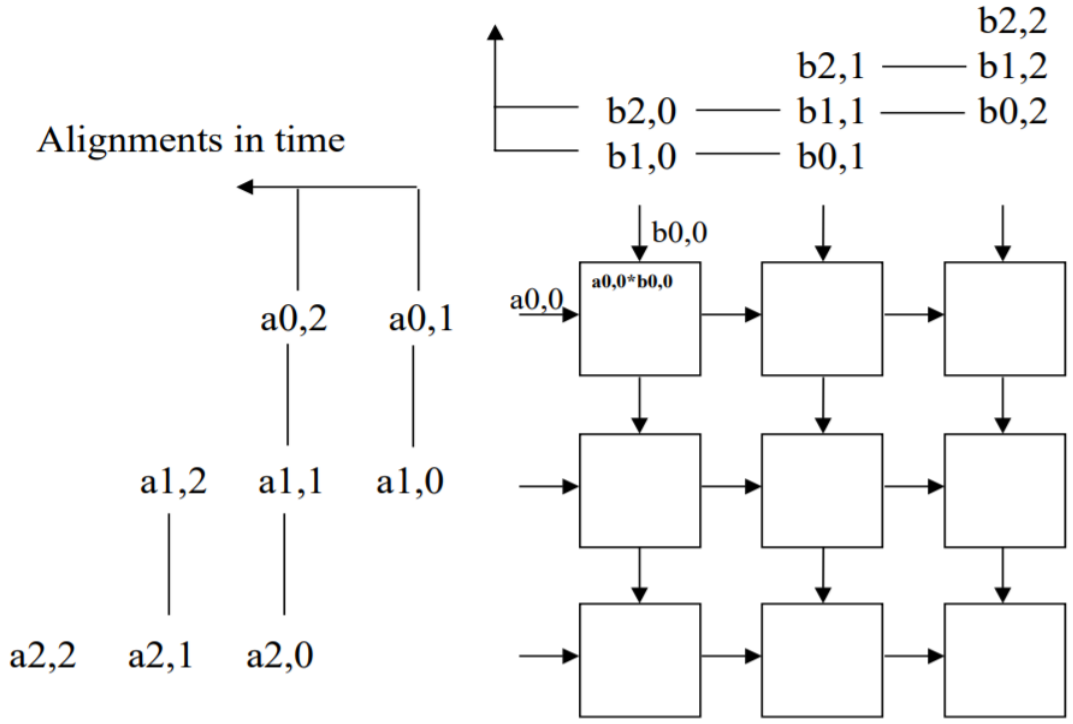
    → 추론에 최적화



Systolic Data Setup

- 연산기의 출력 값이 인접한 다음 줄의 연산기의 입력으로 전달
- 메모리 액세스 빈도를 낮춰 전력 소모 감소
- inference 고속 수행

  511 cycle 동안 최대 65,536번의 곱셈 및 덧셈 가능



3x3 필터

Unified Buffer

- 24MB On-Chip Memory

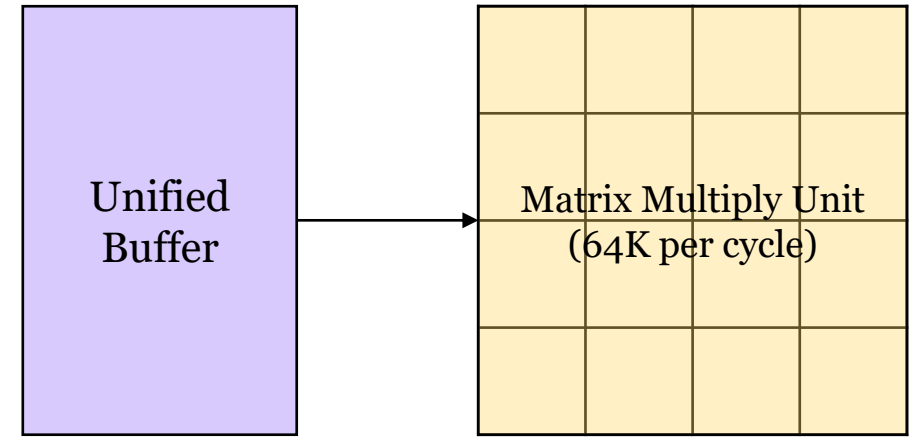
  DDR3 메모리까지 접근할 필요 없이 큰 연산 가능

- 256개의 8-bits 저장 공간이 하나의 단위

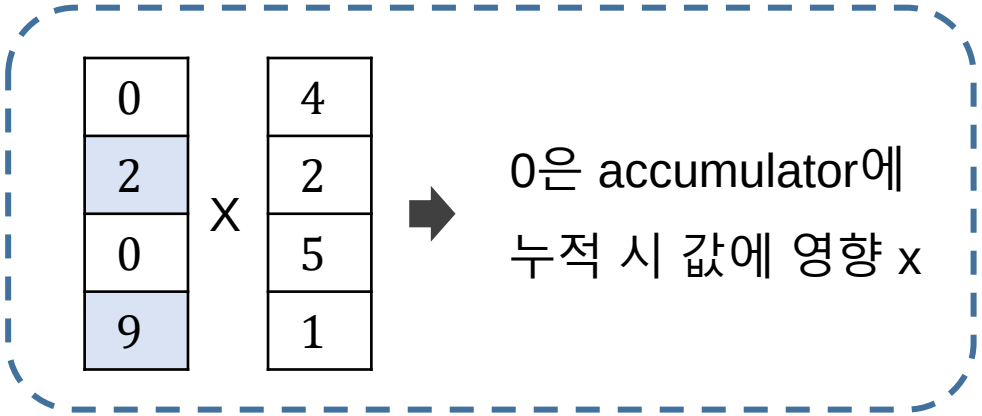
  Matrix Multiply Unit에 데이터 제공 용이

- data가 256x256보다 큰 경우

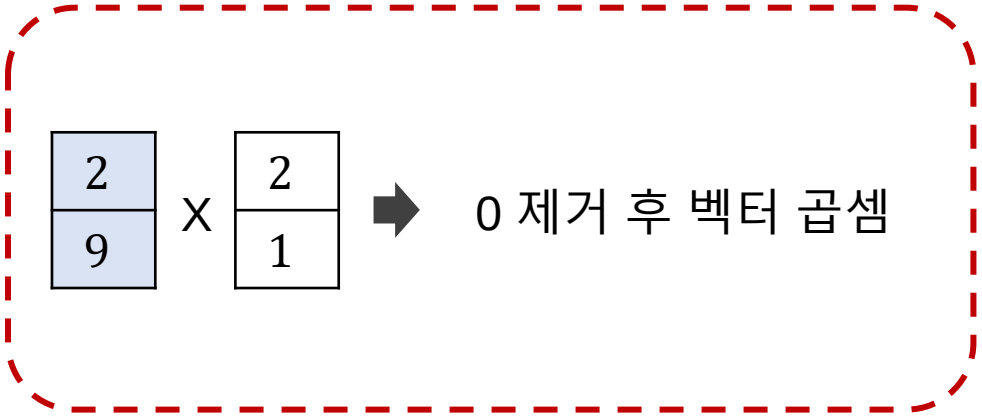
  → 버퍼 저장 후 다시 입력



Zero Value Skipping



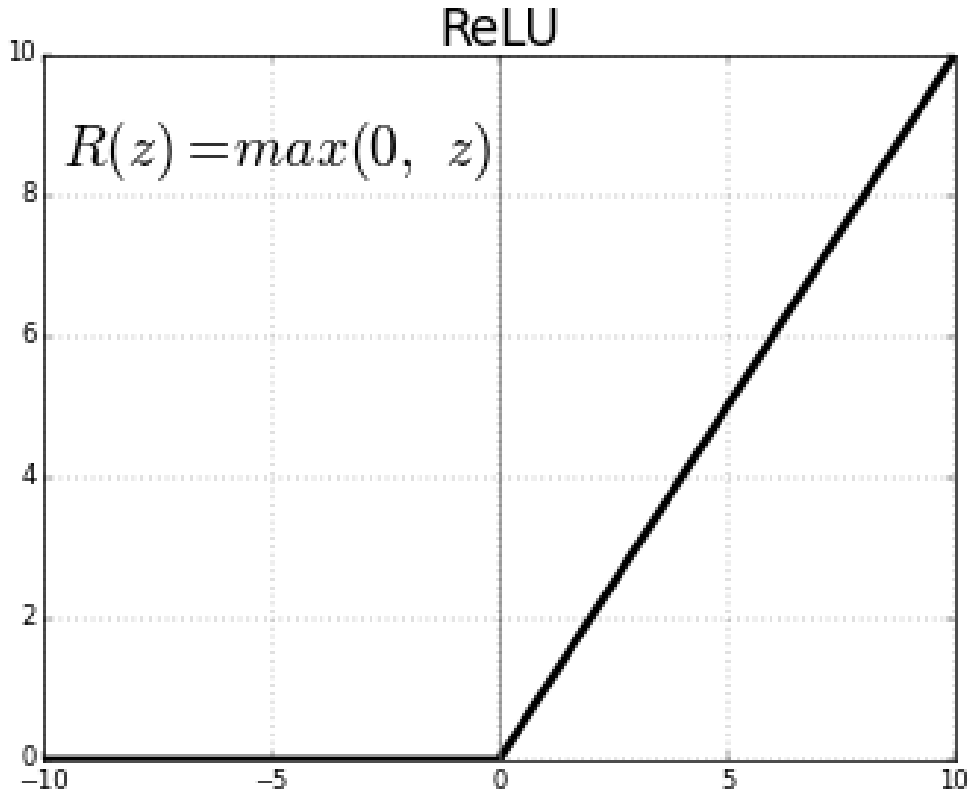
4 cycle



2 cycle

연산 및 전력 소모 감소

Zero Value Skipping

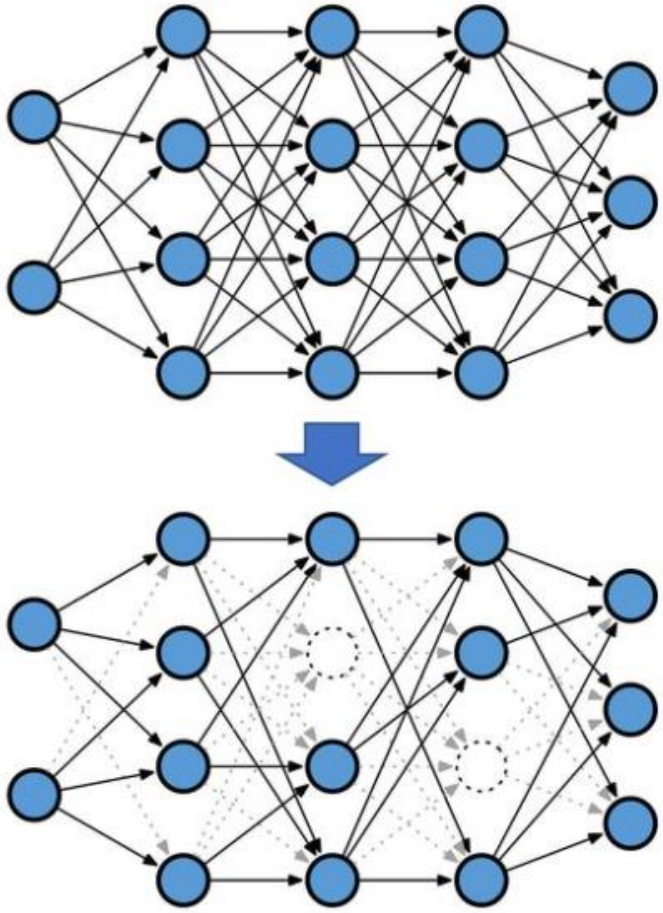


- 입력이 음수일 경우 출력 0
- 데이터에도 0값이 많음
- convolution layer의 연산 중

  37~50%는 0 값 곱셈 연산

Zero Value Skipping

- pruning



Weight Matrix

		6	4		
			3	5	7
				8	
8		4	3		
5	9				

|Weights| > Threshold

↓ Computation and storage

임계치 이하의 weights 를 배제하여

matrix 내부 값들이 대부분 0

↓

데이터 값들이 0으로 변환되어

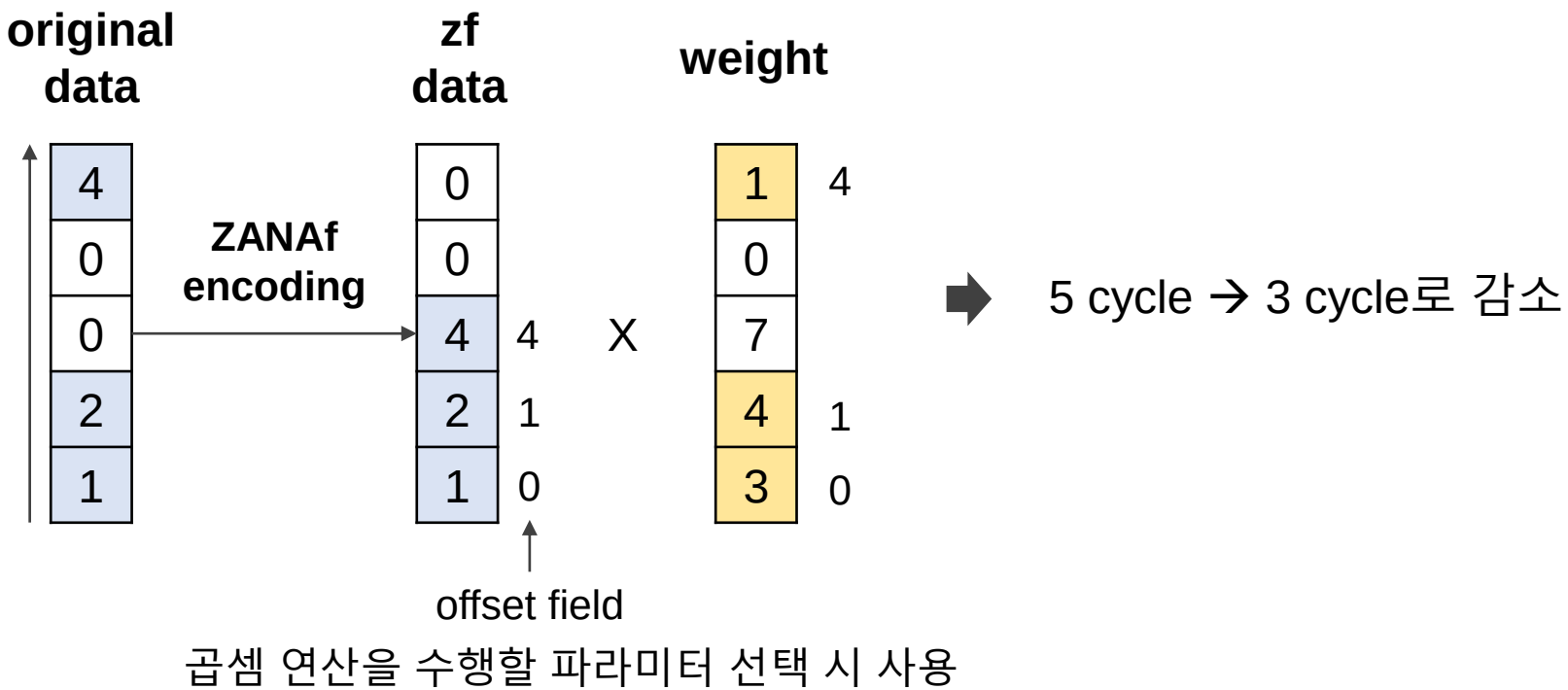
불필요한 연산이 대다수

↓

Cnvlutin 가속기 제안

Cnvlutin

- zero value skipping 통해 convolution layer의 연산을 가속
- zero-Free Neuron Array format (ZFNAf)으로 인코딩
- 인접한 데이터 16개 단위로 수행 (PE별로 할당)



Cnvlutin

- offset field (4-bits/data) 저장할 공간 필요

  → 메모리 크기 증가

- cycle 감소로 인한 절전효과

  but 다수의 벡터를 동시 연산할 경우 성능 향상 제한

Cambricon

- 인공 신경망을 위한 Instruction Set Architecture (ISA)
- 신경망 기술을 제공하는데에 과도한 하드웨어 리소스 사용

  → instruction set 수준의 유연성 확보 필요
- 스칼라, 벡터, 매트릭스 연산에 각각 특화된 명령어
- 각 모델에 대한 명령어가 아니라 범용적으로 사용 가능

Instructino Type	Examples
Control	Jump, conditional branch
Data Transfer	Load, store, move
Computation	Multiply, dot product, random vector generator
Logical	compare

Cambricon

vector, matrix 단위 연산

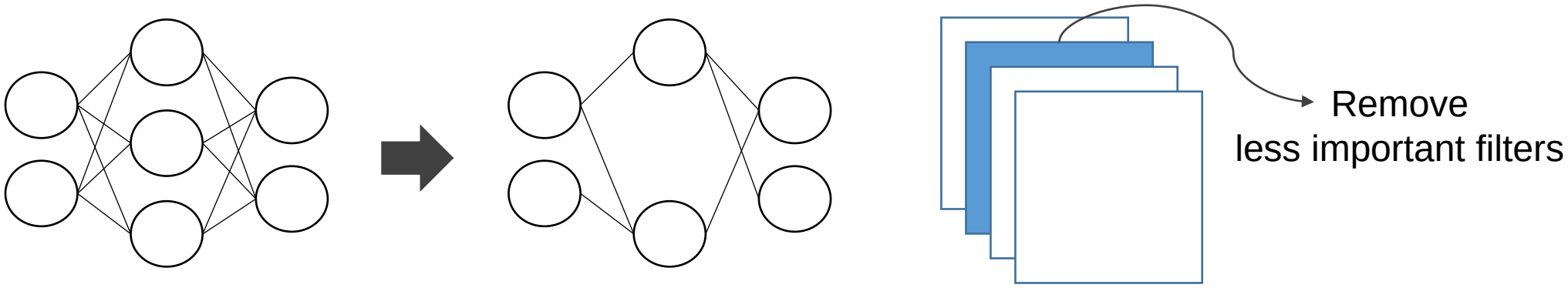
On-chip scratchpad memory 사용

복잡하고 긴 명령어 (CISC)
→ 간단하고 짧은 명령어 (RISC)

복잡성 감소 및 에너지 효율성

Deep Learning Compression

- pruning, quantization, Huffman coding
- $weights < |threshold|$ 인 $weights$ 제거 후 재 훈련
- memory-efficient, energy-efficient and computation efficiency



sparse한 모델으로 만들어서 연산량 감소

EIE (Efficient Inference Engine)

- DNN 가속기의 전력 소비 요인

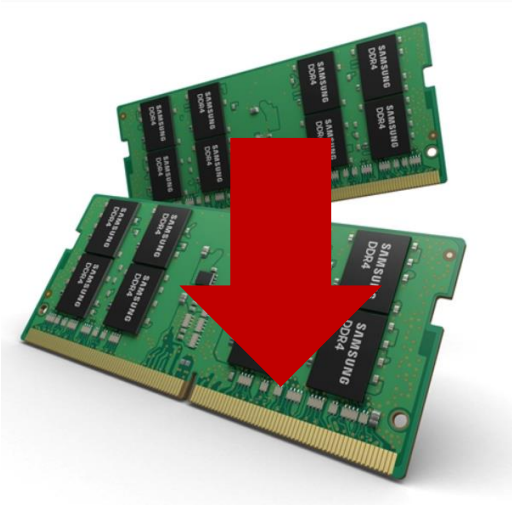
  부동 소수점 곱셈 연산 & 외부 메모리 접근

  DRAM 접근이 곱셈 연산에 비해 약 173배의 전력 소모

- Idea

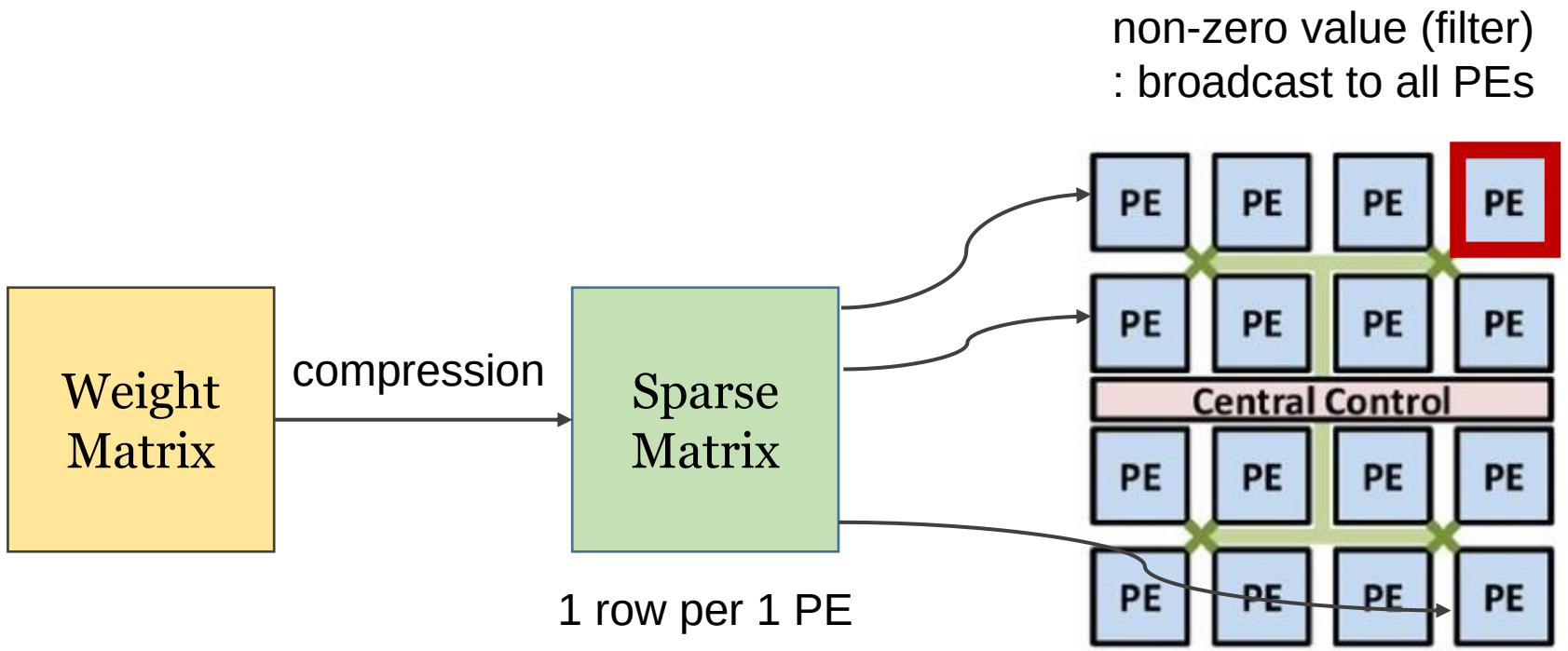
  DRAM 접근 줄여야함

  Deep compression → sparse weights → 모바일 장치에서 사용 가능



EIE (Efficient Inference Engine)

- 압축된 DNN의 추론에 특화된 구조
- 다수의 PEs, 분산된 SRAM (162KB per PE)



1. SRAM에서
   non-zero weights 로드
2. Arithmetic units
   : multiply and accumulate
3. Save to SRAM

Q & A