

Architecture Design for Highly Flexible and Energy-Efficient Deep Neural Network Accelerators

Yu-Hsin Chen

May 2, 2018

Thesis Advisors: **Prof. Vivienne Sze, Prof. Joel Emer**

Thesis Committee: **Prof. Daniel Sanchez**

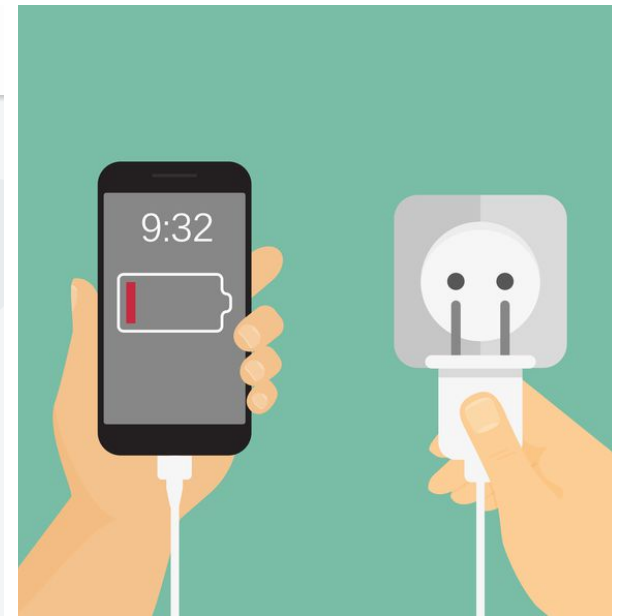
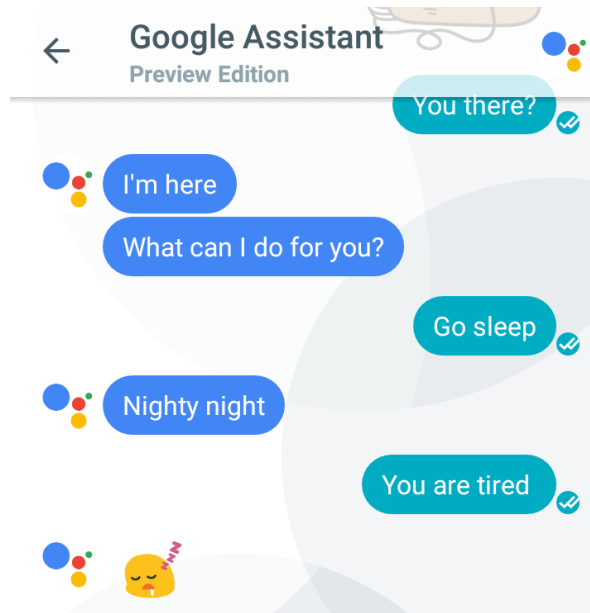
Deep Neural Networks are Everywhere



[image sources: Amazon, nature, sciencenews.org]

New Challenges for Hardware Systems

- **Performance:** High Throughput / Low Latency
- **High Energy Efficiency**
- **High Flexibility**



[images sources: Uber, livinglifetechway.com, lifewire.com]

Focus of Thesis

Eyeriss

Architecture for DNN acceleration optimized for *performance, energy efficiency* and *flexibility*

Focus of Thesis

Eyeriss

Architecture for DNN acceleration optimized for *performance*, *energy efficiency* and *flexibility*

Eyeriss v1

Data Reuse Optimization

Targeting **large DNNs** with hundreds of filters and channels

Focus of Thesis

Eyeriss

Architecture for DNN acceleration optimized for *performance*, *energy efficiency* and *flexibility*

Eyeriss v1

Data Reuse Optimization

Targeting **large DNNs** with hundreds of filters and channels

Eyeriss v2

Processing Element Utilization

Targeting **both large and compact DNNs** with highly varying size and computation

7 Key Contributions of Thesis

- **DNN Accelerator Architecture Analysis Methodologies**
 - Dataflow Taxonomy
 - Energy Estimation Framework
 - Performance Modeling Framework
 - Analogy to General-Purpose Architecture
- **Eyeriss v1 Architecture**
 - Energy-Efficient Dataflow
 - Flexible Mapping Strategy for High Utilization
 - Energy-Efficient and Flexible NoC
 - Sparsity: Reduce DRAM Accesses with Feature Map Compression
 - Sparsity: Reduce Energy Consumption with Zero-Data Gating
 - Prototype Chip Development
 - System Demonstration on Prototype Chip
- **Eyeriss v2 Architecture**
 - Flexible Dataflow
 - Flexible Hierarchical NoC
 - Sparsity: Feature Map and Weight Compression
 - Sparsity: Operation Skipping for Higher Performance

What We Will Cover Today

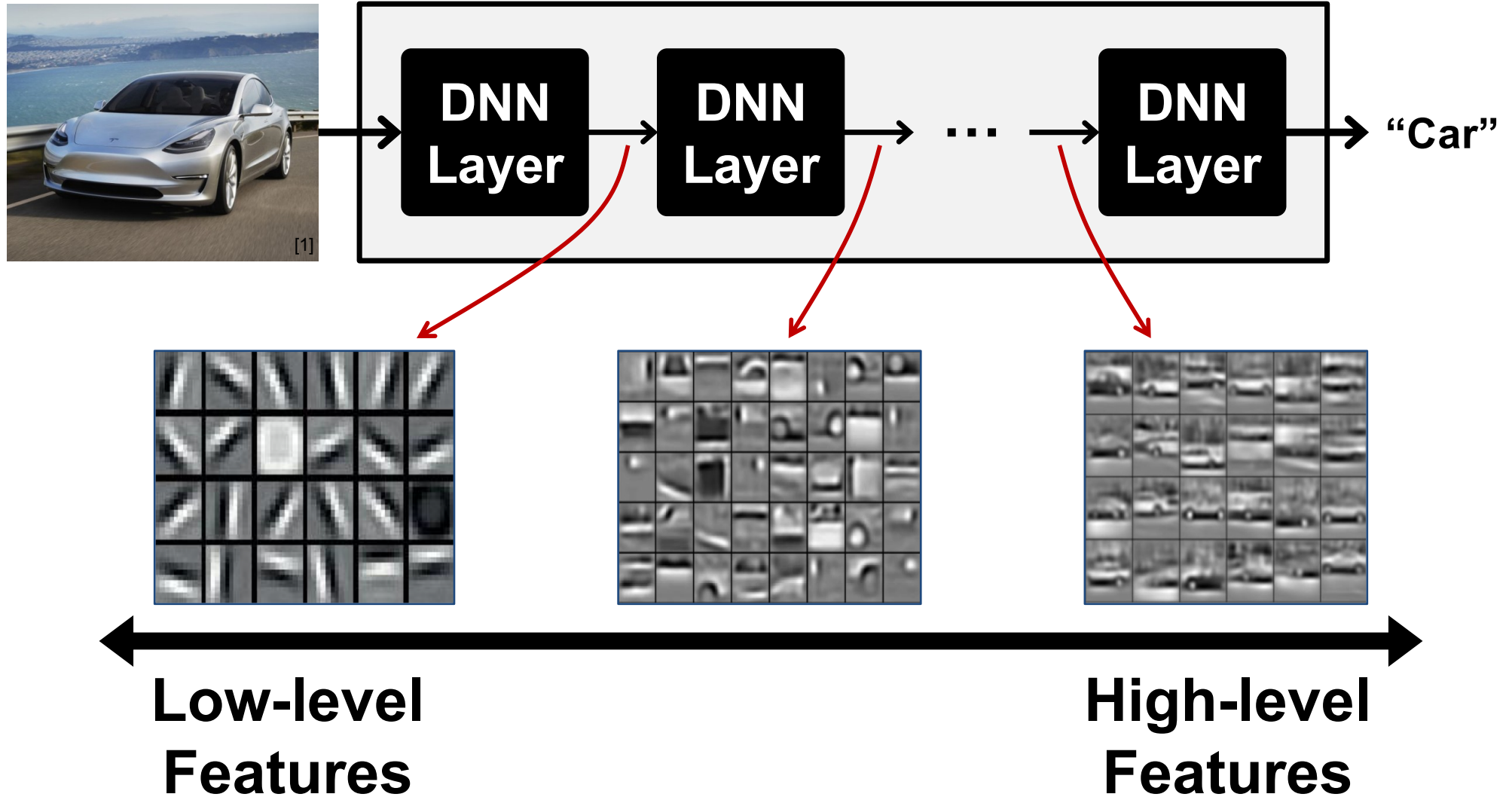
- **DNN Accelerator Architecture Analysis Methodologies**
 - Dataflow Taxonomy
 - Energy Estimation Framework
 - Performance Modeling Framework
 - Analogy to General-Purpose Architecture
- **Eyeriss v1 Architecture**
 - Energy-Efficient Dataflow
 - Flexible Mapping Strategy for High Utilization
 - Energy-Efficient and Flexible NoC
 - Sparsity: Reduce DRAM Accesses with Feature Map Compression
 - Sparsity: Reduce Energy Consumption with Zero-Data Gating
 - Prototype Chip Development
 - System Demonstration on Prototype Chip
- **Eyeriss v2 Architecture**
 - Flexible Dataflow
 - Flexible Hierarchical NoC
 - Sparsity: Feature Map and Weight Compression
 - Sparsity: Operation Skipping for Higher Performance

Summary of PhD Publications

- **Eyeriss v1**
 - *ISCA* 2016 (**most downloaded from ISCA 2016**)
 - *Micro Top-Picks* 2017 (**top-12 comp. arch. papers of 2017**)
 - *ISSCC* 2016
 - *JSSC* 2017 (**most downloaded JSSC paper in 2017**)
- **Eyeriss v2**
 - *SysML* 2018
 - *Micro* 2018 (in submission)
- **Survey and Tutorial**
 - *Proceedings of IEEE* 2017 (**cover article**)
 - *ISCAS* 2017
 - *CICC* 2017 (**best invited paper**)

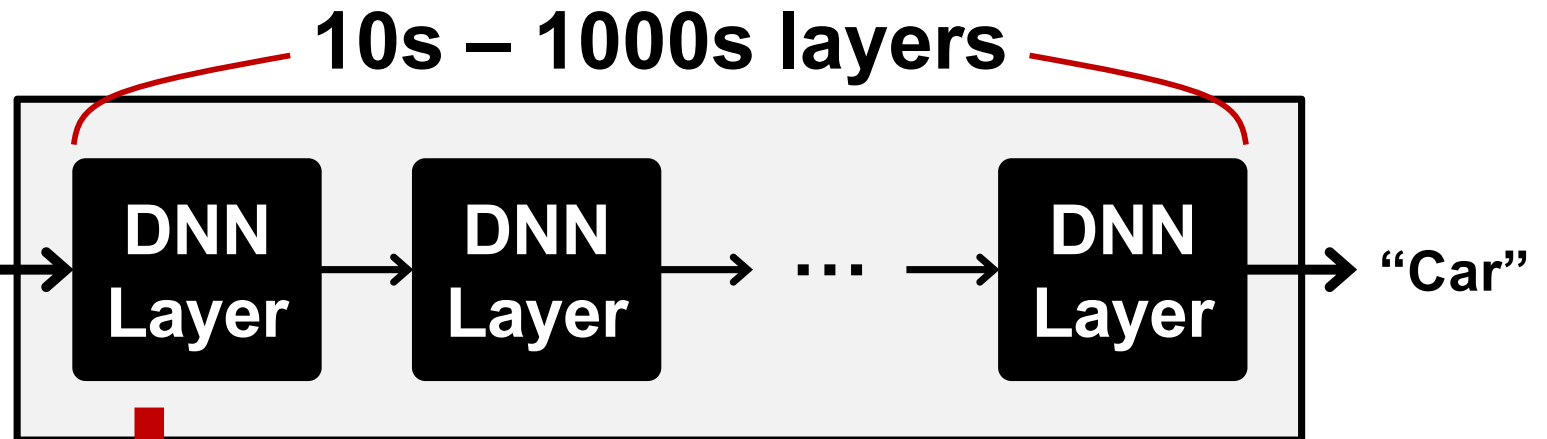
Primer on Deep Neural Networks

Deep Neural Network (DNN)

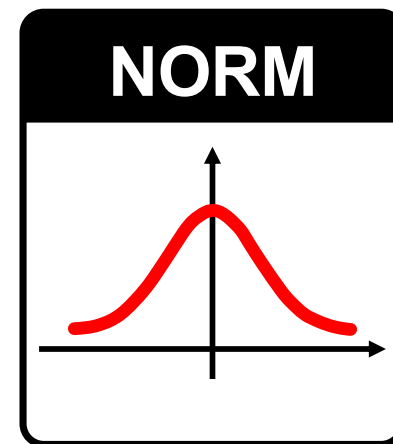
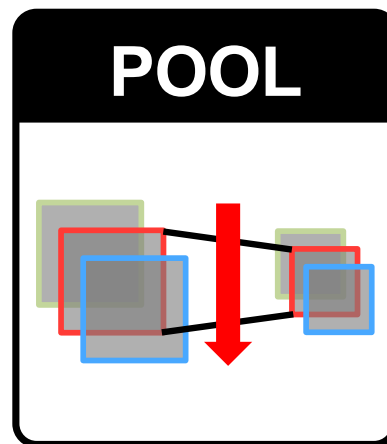
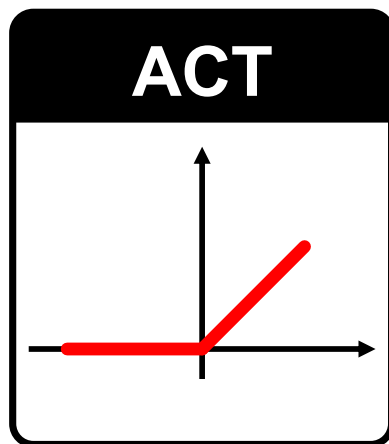
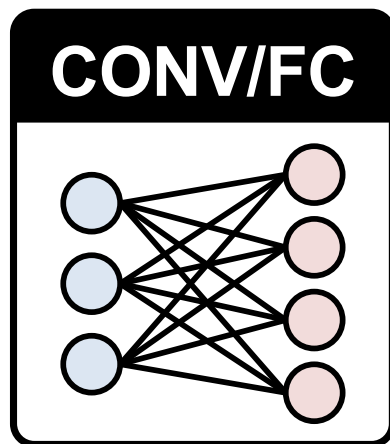


[1] Tesla

11 Primer on Deep Neural Networks



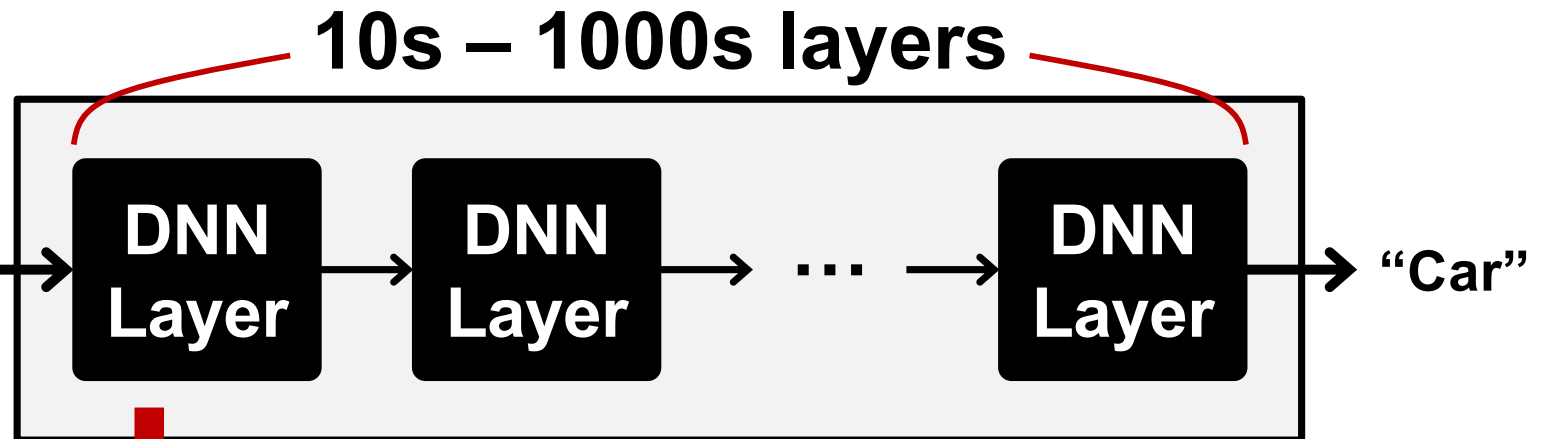
many layer types to choose from



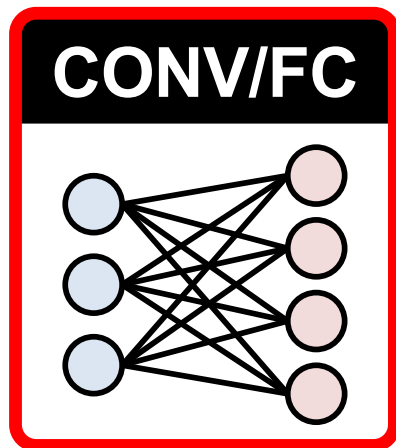
...

[1] Tesla

12 Primer on Deep Neural Networks



many layer types to choose from



CONV and FC layers account for 90% to 99% of the computation

ACT

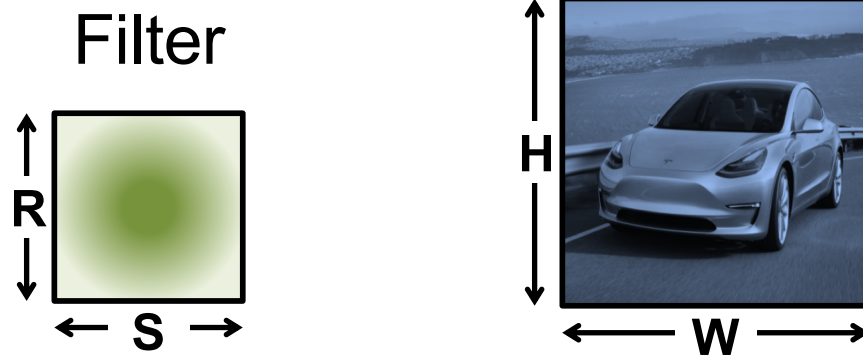
POOL

NORM

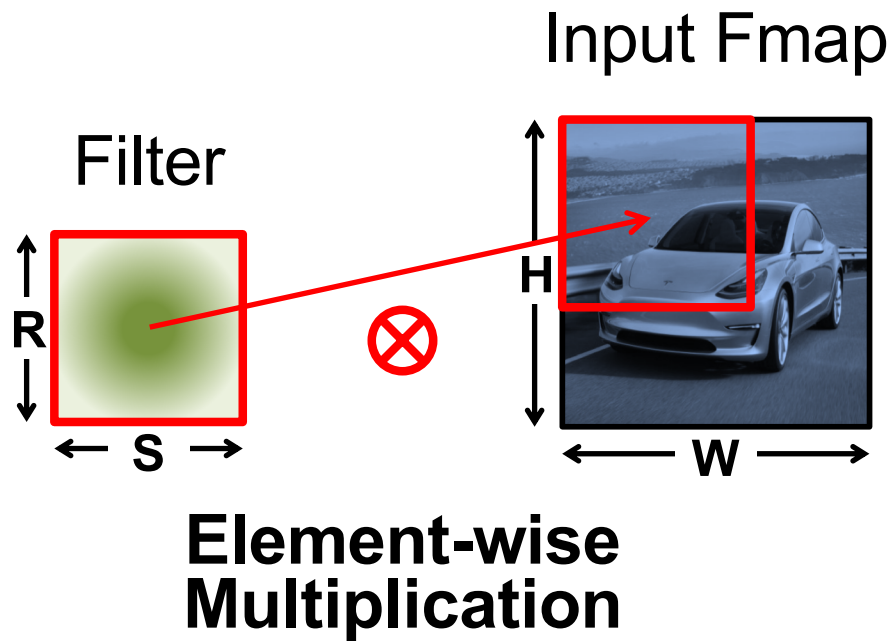
[1] Tesla

High-Dimensional Convolution (CONV/FC)

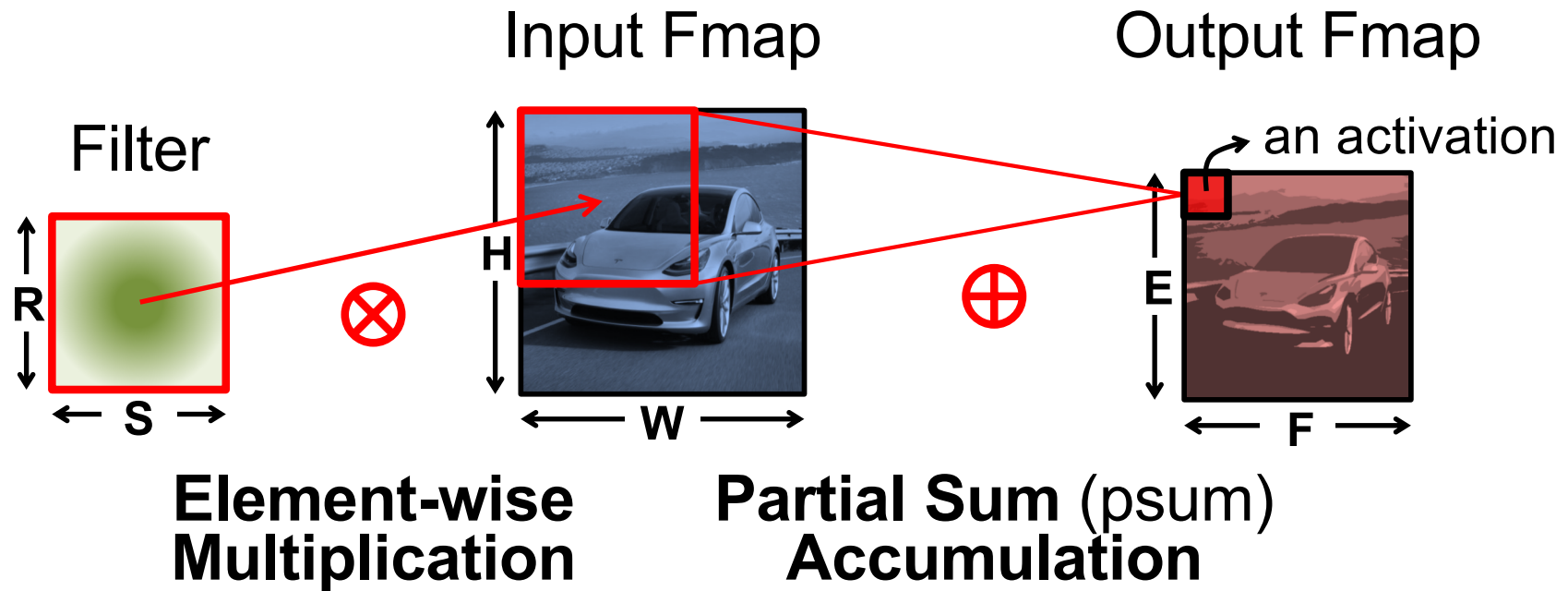
Input Feature Map (Fmap)



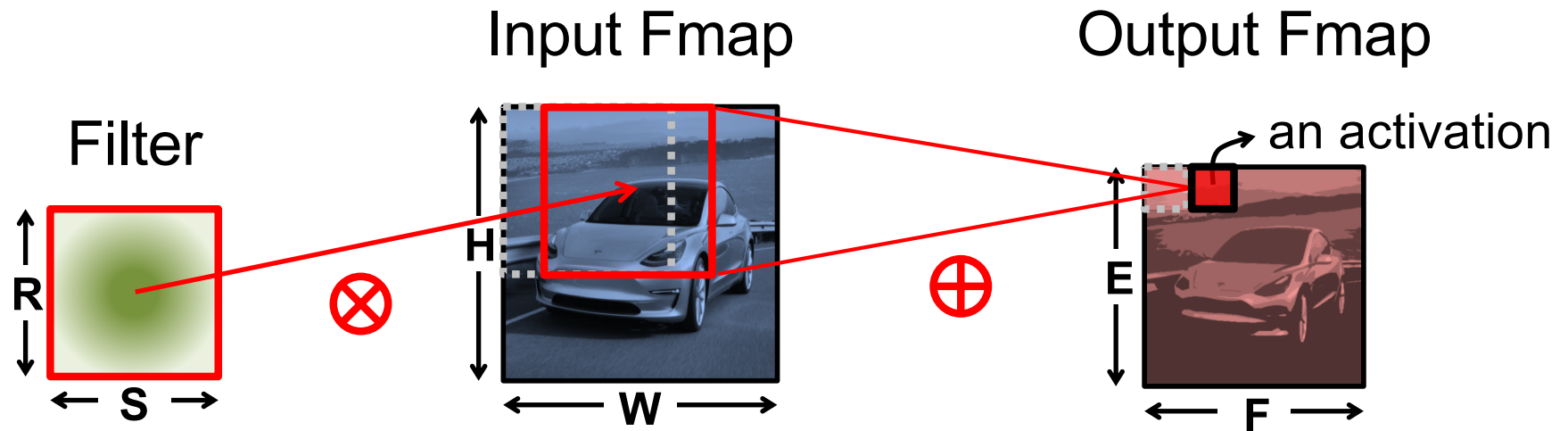
High-Dimensional Convolution (CONV/FC)



High-Dimensional Convolution (CONV/FC)

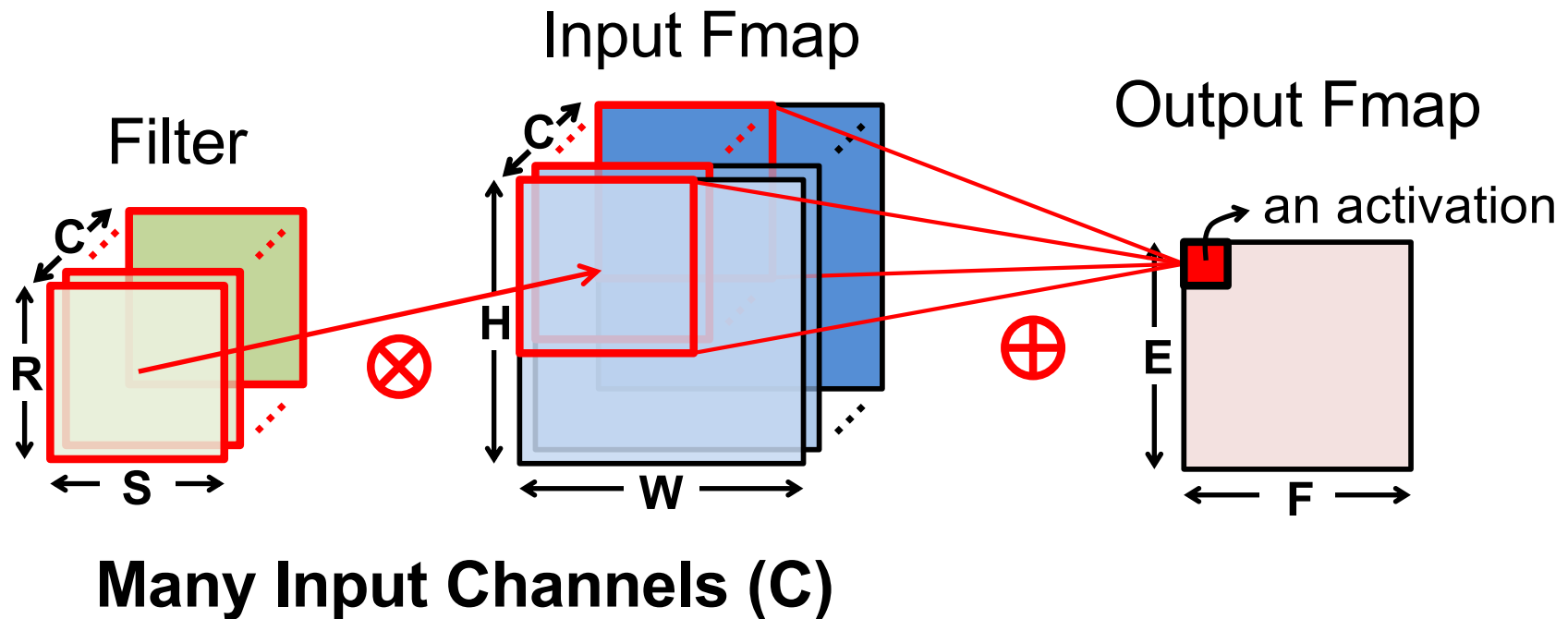


High-Dimensional Convolution (CONV/FC)

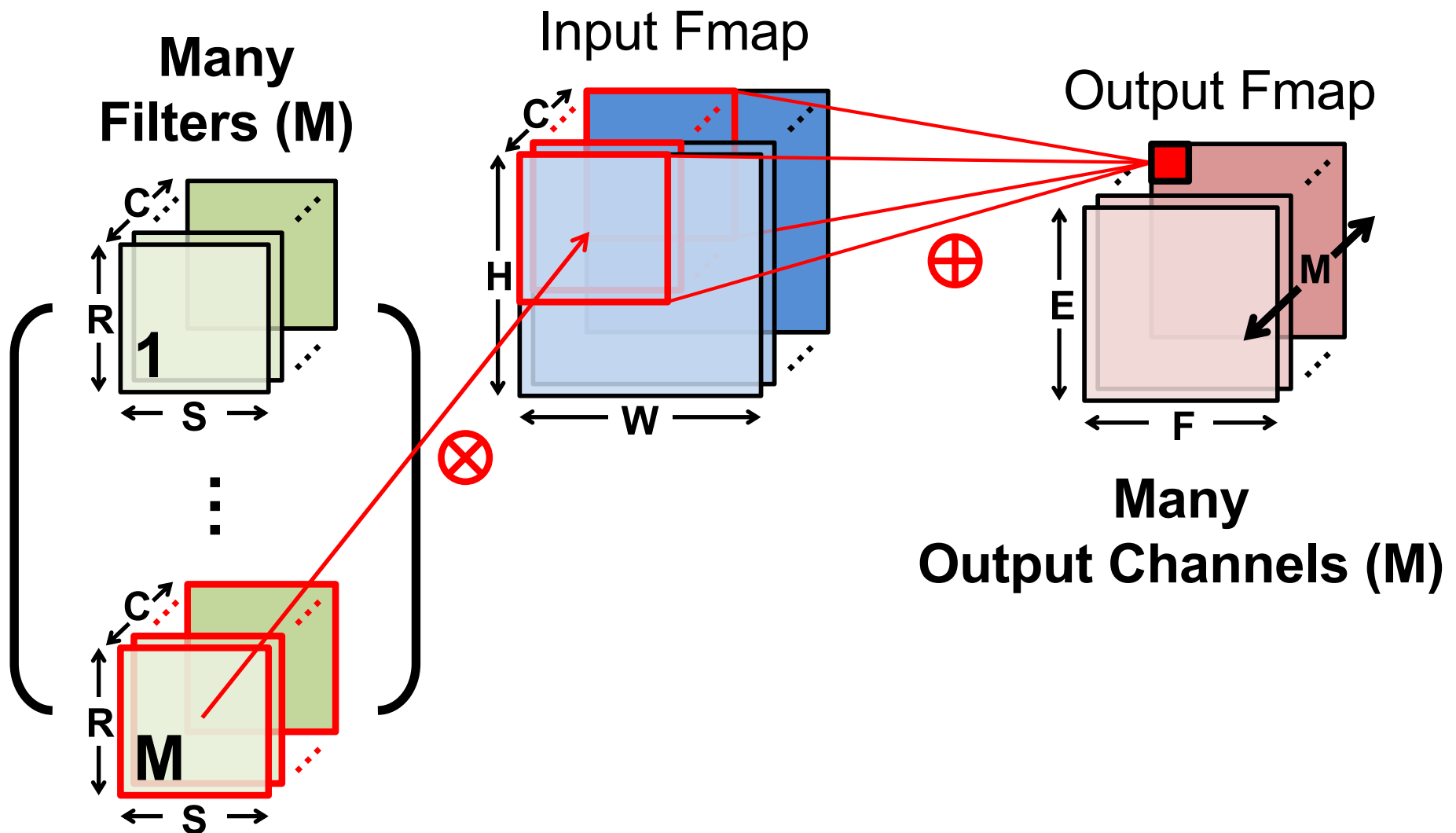


Sliding Window Processing
(for CONV layers only)

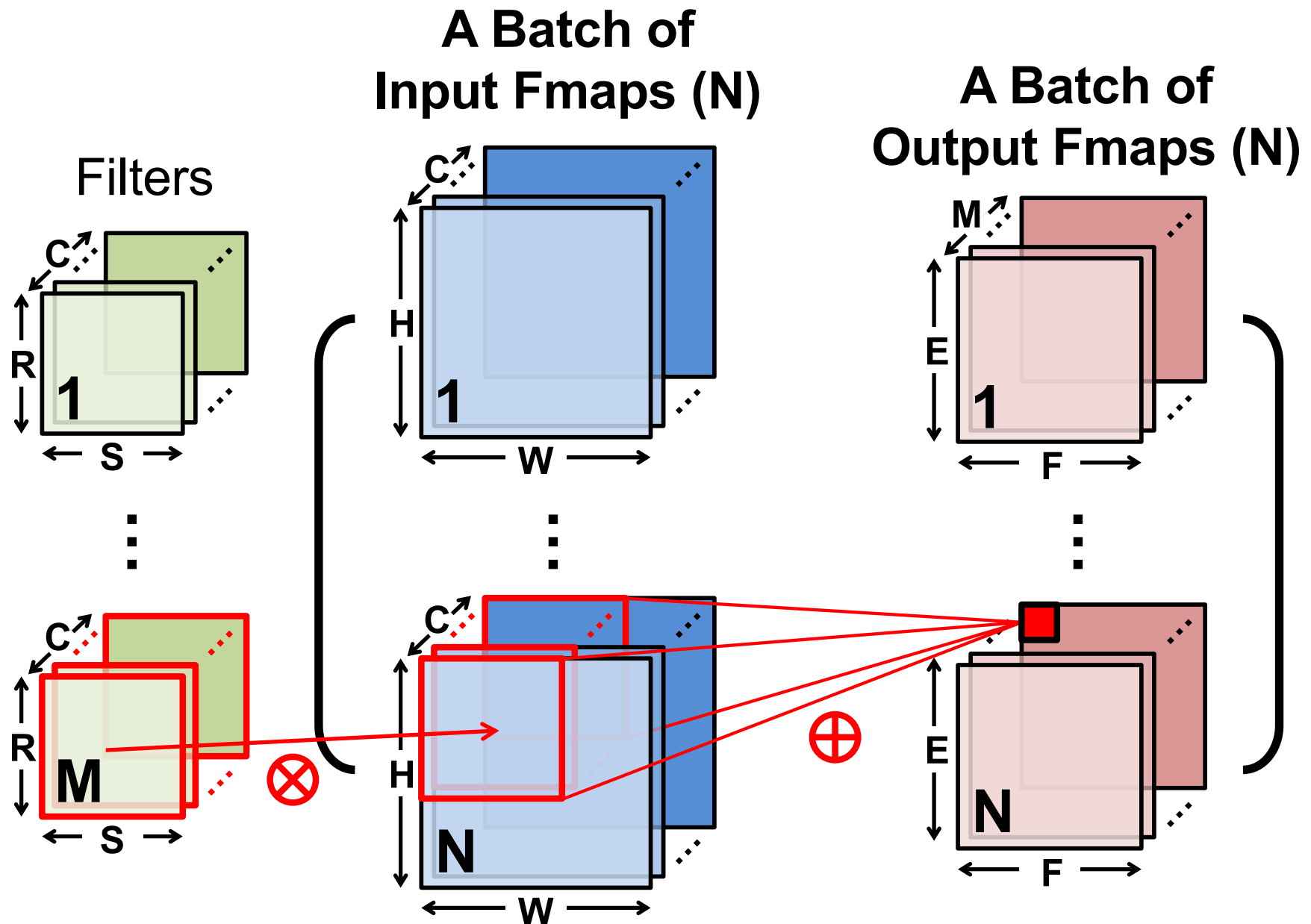
High-Dimensional Convolution (CONV/FC)



High-Dimensional Convolution (CONV/FC)



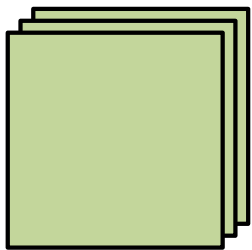
High-Dimensional Convolution (CONV/FC)



AlexNet^[1] Layer Shape Configurations

Layer	Filter Size (R)	# Filters (M)	# Channels (C)	Stride
1	11×11	96	3	4
2	5×5	256	48	1
3	3×3	384	256	1
8	1×1	1000	4096	1

Layer 1



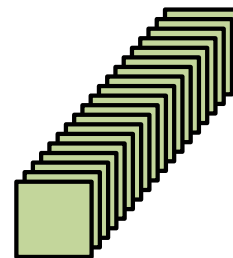
34k weights
105M MACs*

Layer 2



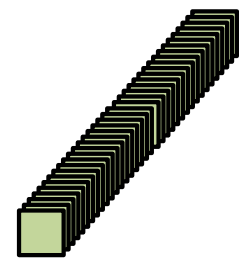
307k weights
224M MACs

Layer 3



885k weights
150M MACs

Layer 8

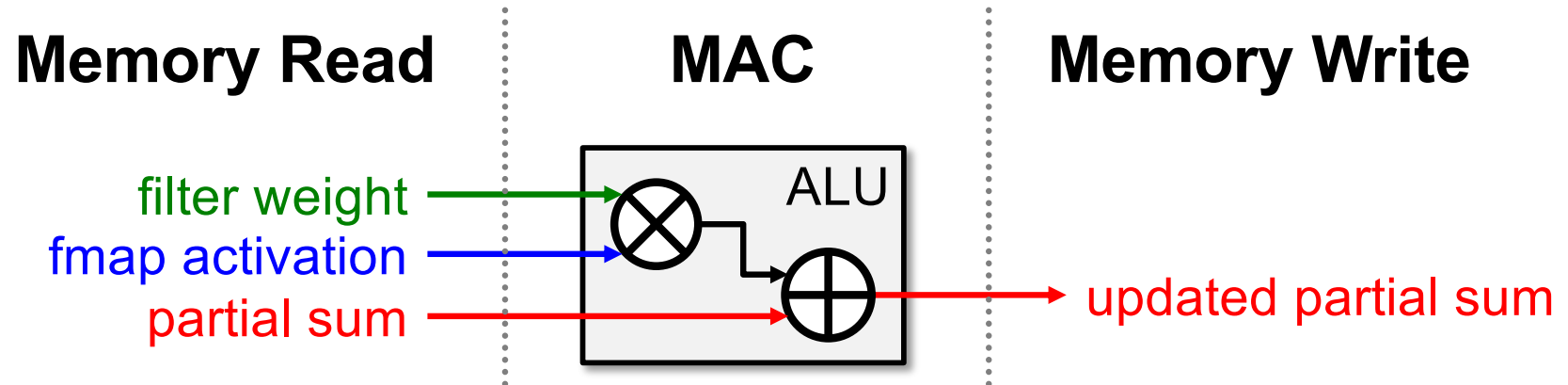


4096k weights
4M MACs

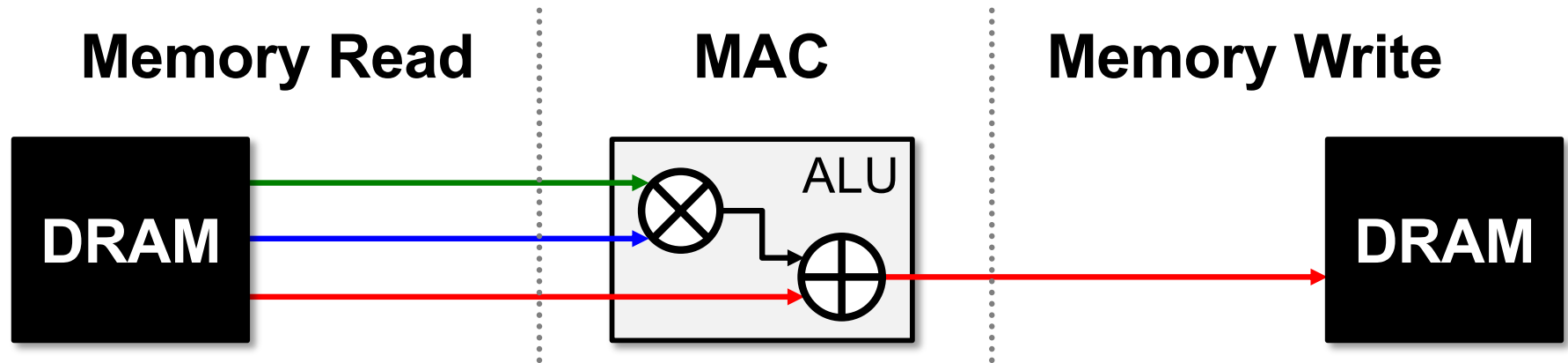
* Multiply-and-Accumulate

[1] Krizhevsky, *NIPS* 2012

Memory Access is the Bottleneck



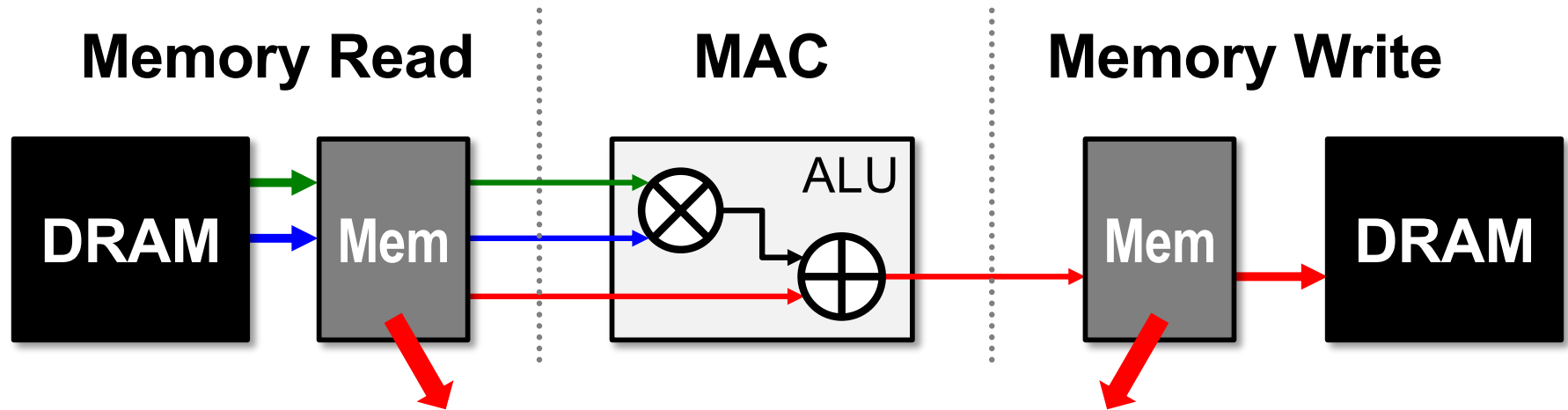
Memory Access is the Bottleneck



Worst Case: all memory R/W are **DRAM** accesses

- Example: **AlexNet** has **724M** MACs
→ **2896M** DRAM accesses required

Leverage Local Memory for Data Reuse



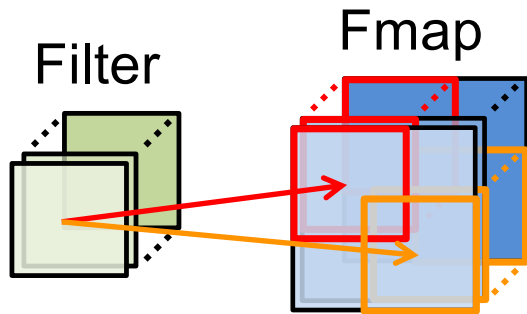
Extra levels of local memory hierarchy

Smaller, but Faster and more Energy-Efficient

Types of Data Reuse in a DNN

Convolutional Reuse

CONV layers only
(sliding window)



Reuse:

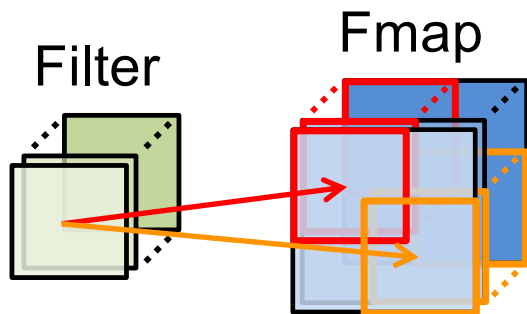
Fmap Activations

Filter Weights

Types of Data Reuse in a DNN

Convolutional Reuse

CONV layers only
(sliding window)



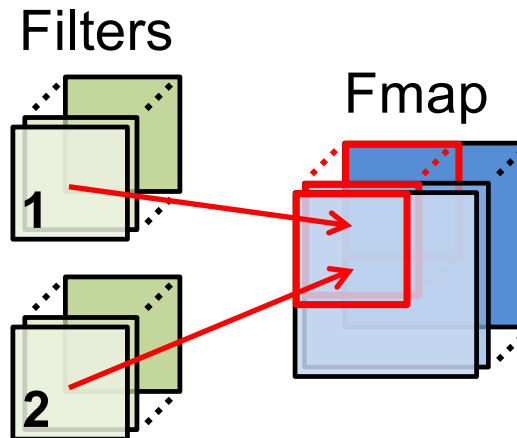
Reuse:

Fmap Activations

Filter Weights

Fmap Reuse

CONV and FC layers

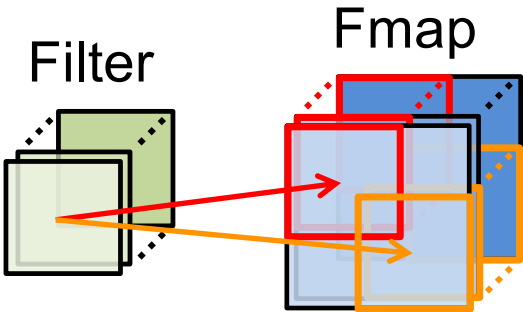


Reuse:

Fmap Activations

Convolutional Reuse

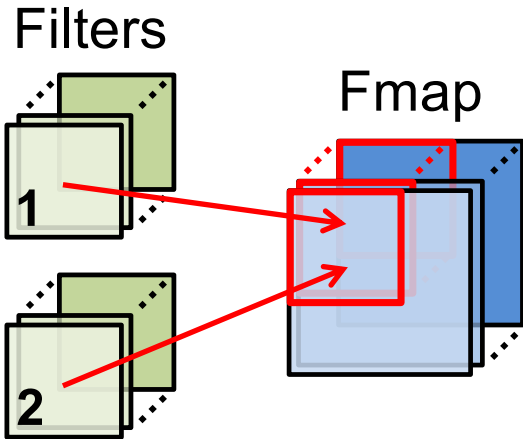
CONV layers only
(sliding window)



Reuse:
 Fmap Activations
 Filter Weights

Fmap Reuse

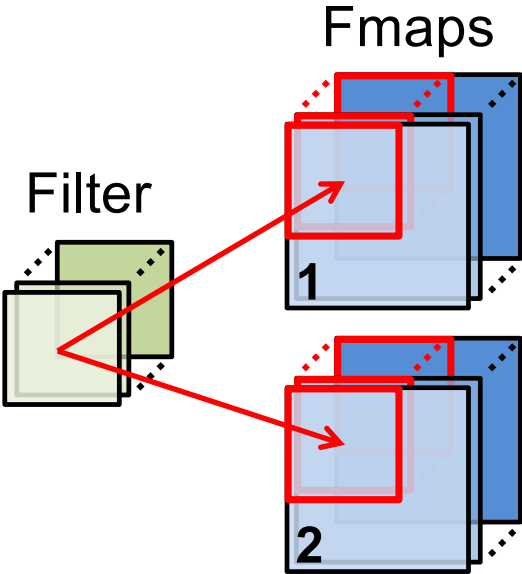
CONV and FC layers



Reuse:
 Fmap Activations

Filter Reuse

CONV and FC layers
(batch size > 1)



Reuse:
 Filter Weights

Types of Data Reuse in a DNN

Convolutional Reuse

CONV layers only
(sliding window)

Fmap Reuse

CONV and FC layers

Filter Reuse

CONV and FC layers
(batch size > 1)

If all data reuse is exploited, DRAM accesses in AlexNet can be reduced from **2896M** to **61M** (best case)

Reuse:

Fmap Activations

Filter Weights

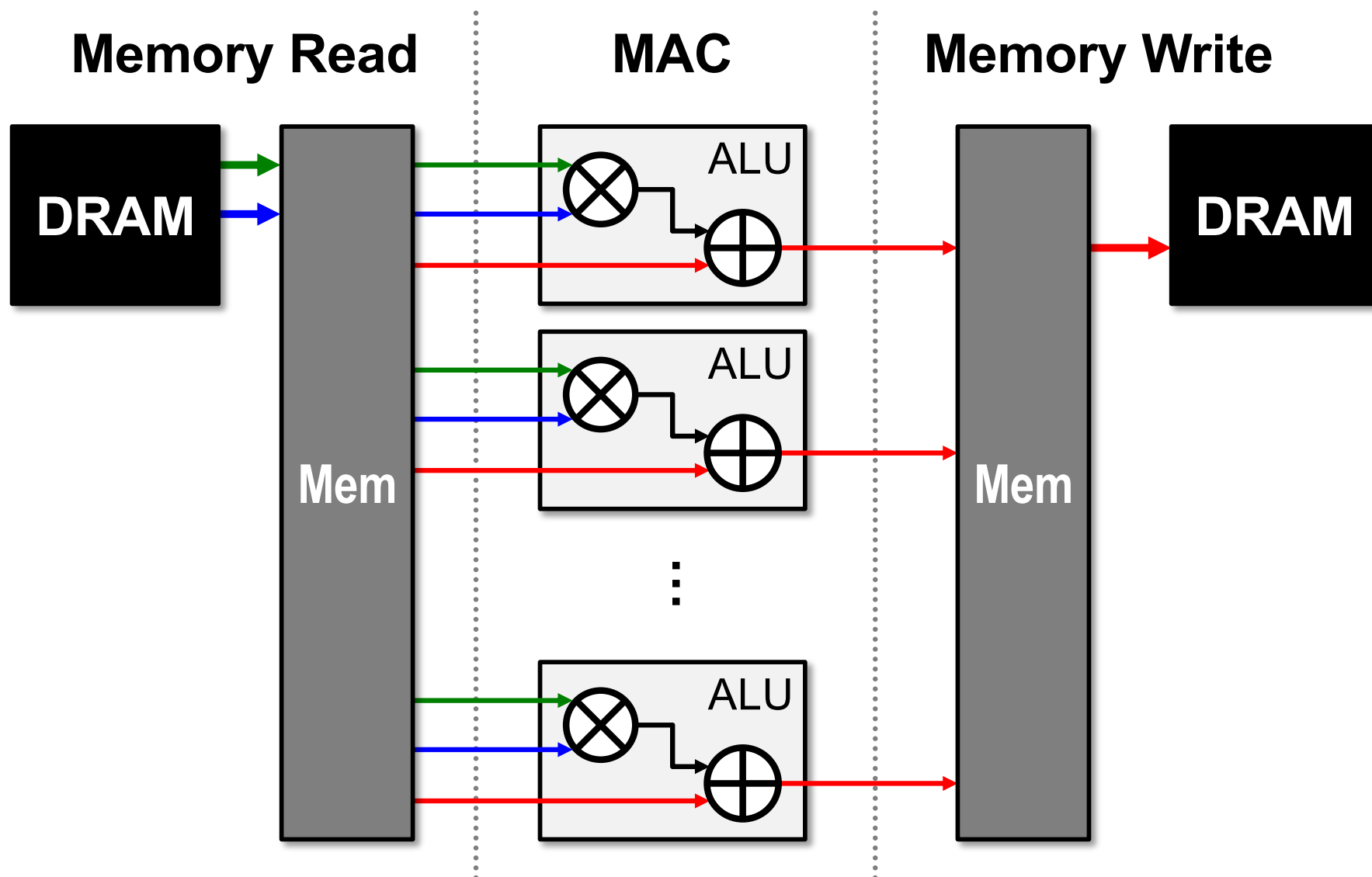
Reuse:

Fmap Activations

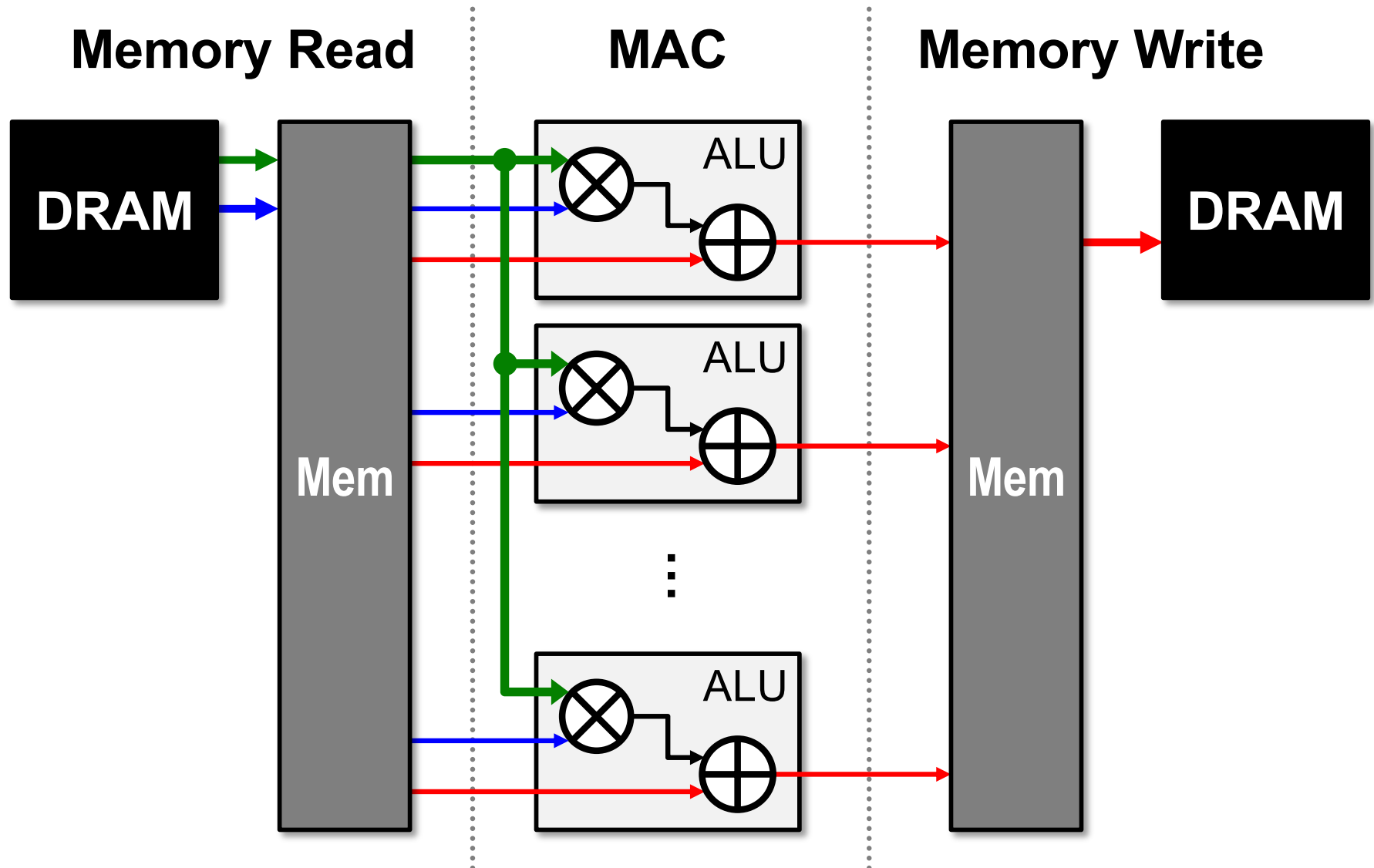
Reuse:

Filter Weights

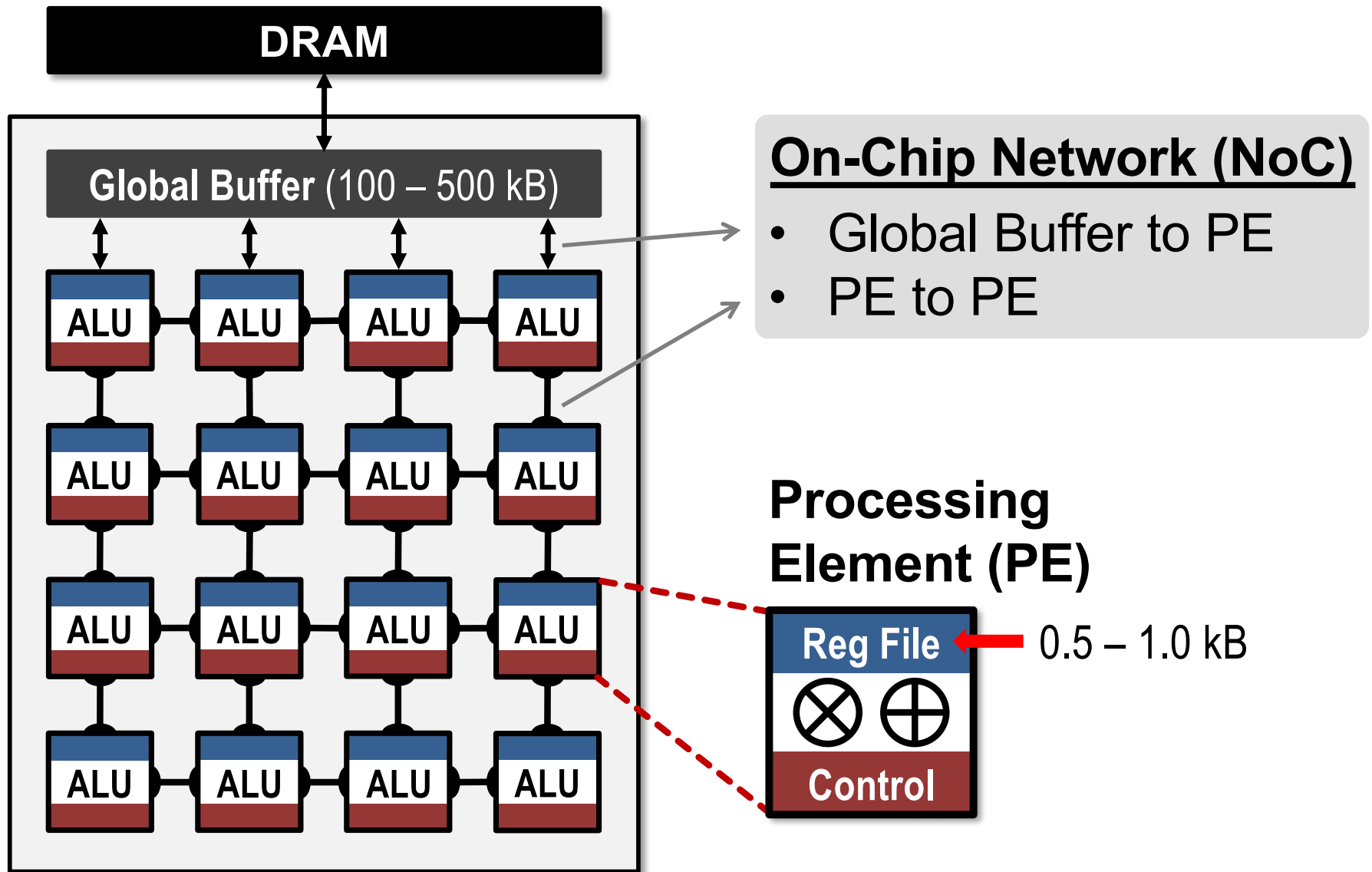
Leverage Parallelism for Higher Performance



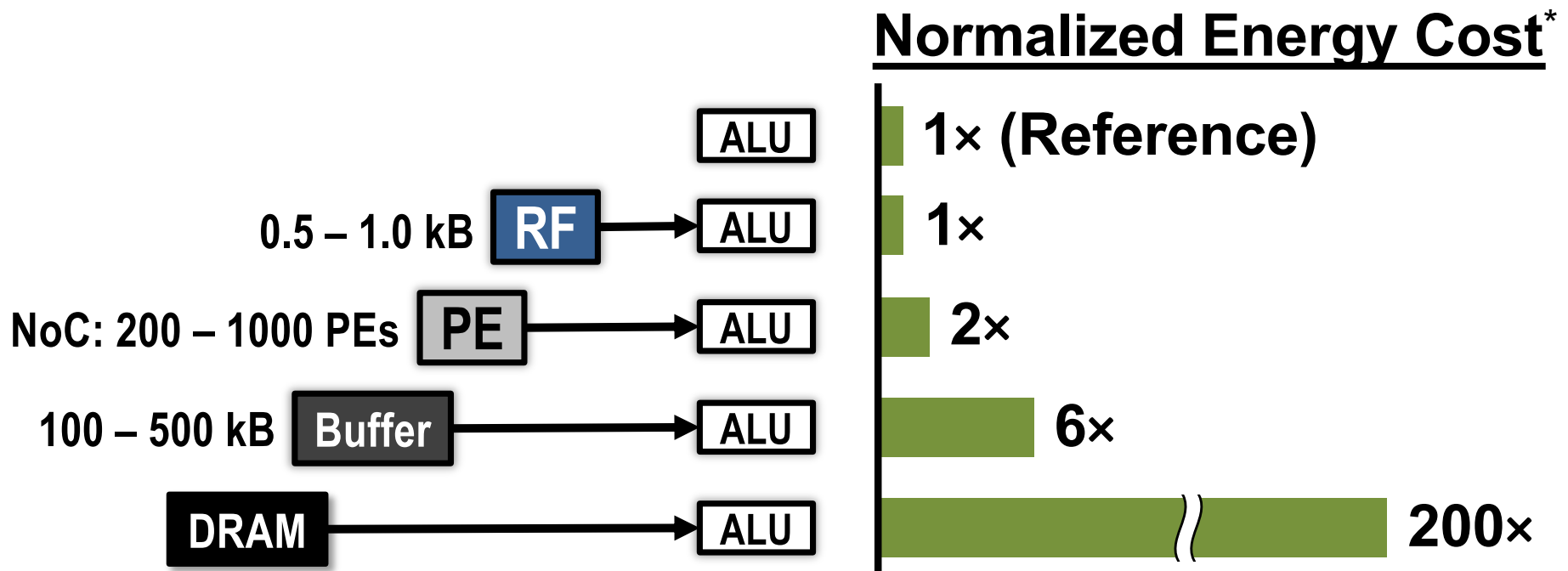
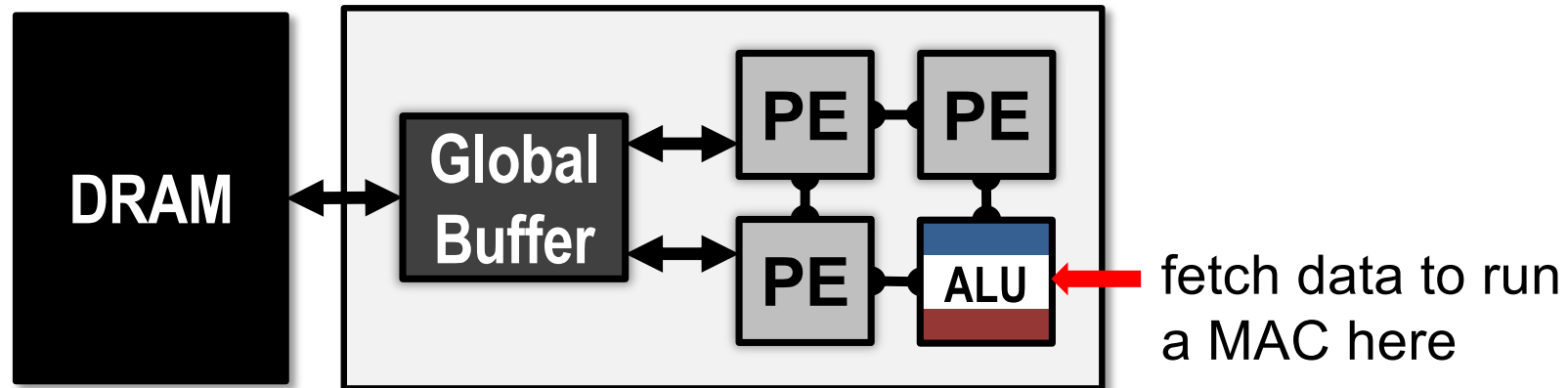
Leverage Parallelism for *Spatial* Data Reuse



Spatial Architecture



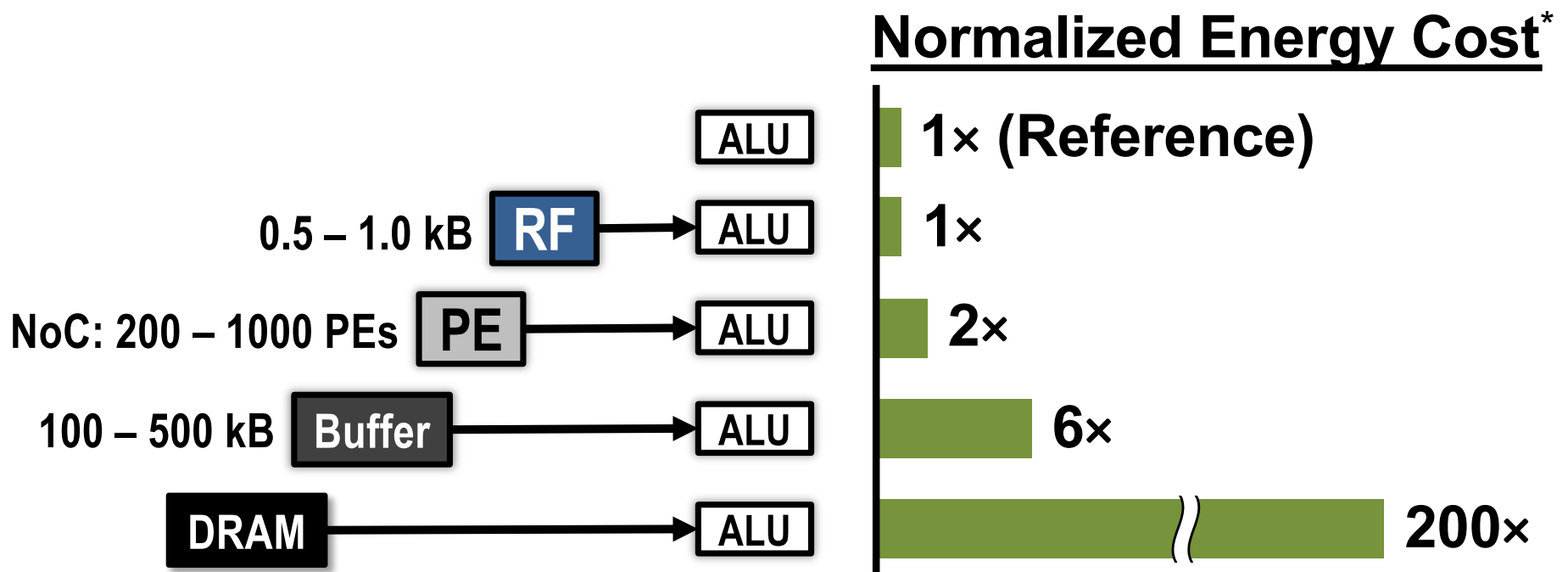
Multi-Level Low-Cost Data Access



* Measured from a commercial 65nm process

Multi-Level Low-Cost Data Access

A **Dataflow** is required to maximally exploit **data reuse** with the **low-cost memory hierarchy** and **parallelism**



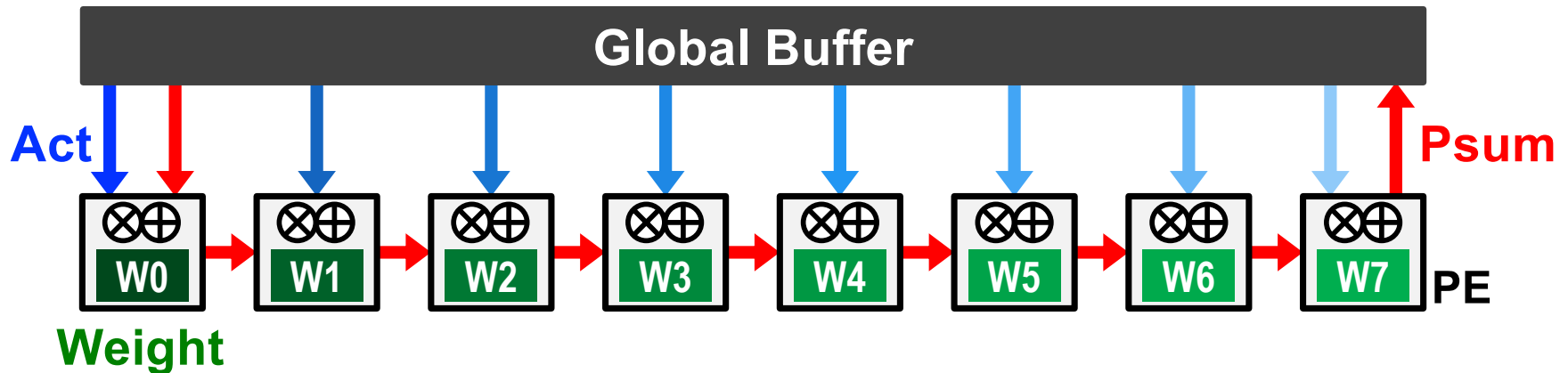
* Measured from a commercial 65nm process

Taxonomy of Existing Dataflows

- Weight Stationary (WS)
- Output Stationary (OS)
- No Local Reuse (NLR)

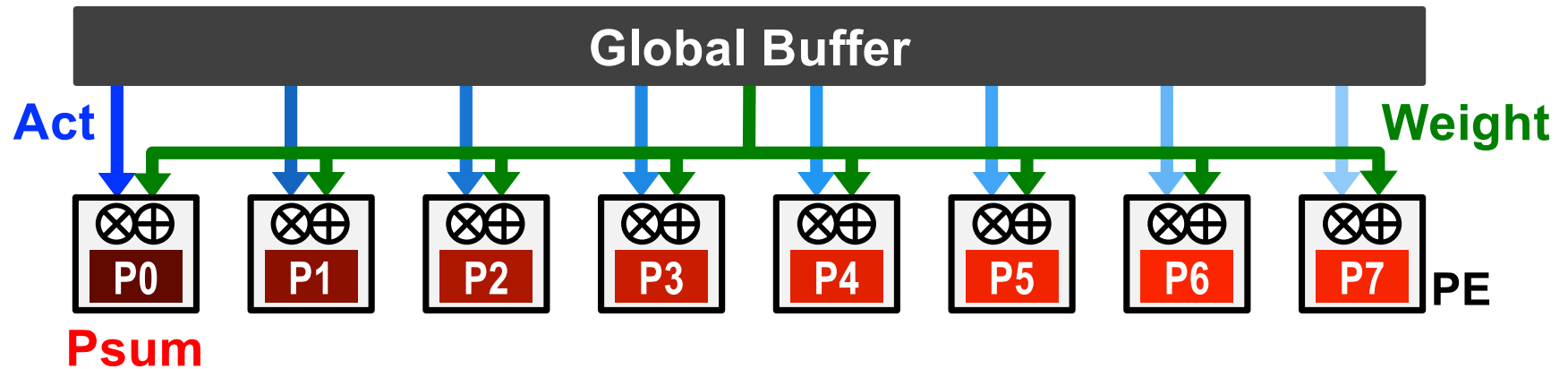
[Chen, ISCA 2016]

Weight Stationary (WS)



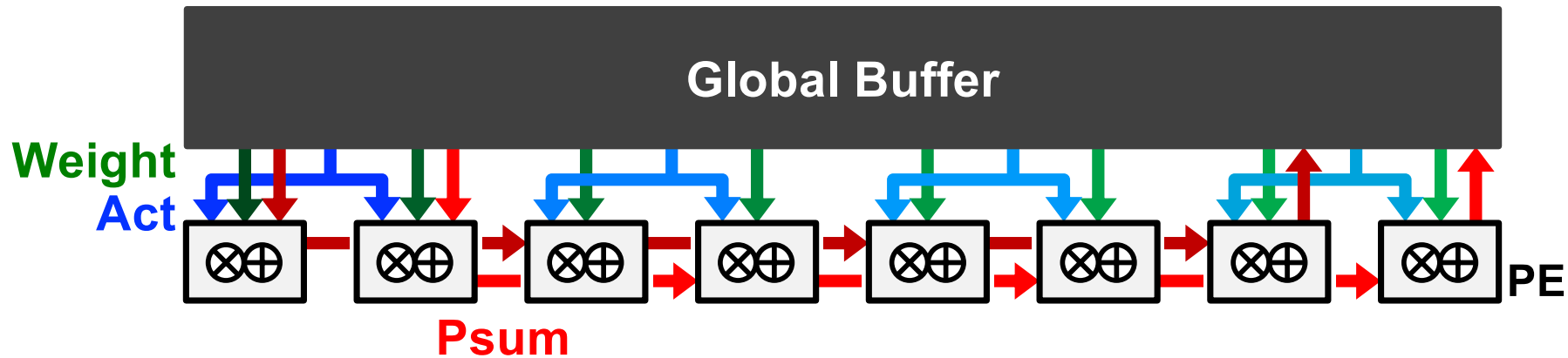
- Minimize **weight** read energy consumption
 - maximize convolutional and filter reuse of weights
- Examples:
 - [nn-X (NeuFlow), CVPRW 2014] [Park, ISSCC 2015]
 - [Origami, GLSVLSI 2015] [Google TPU, ISCA 2017]

Output Stationary (OS)



- Minimize **partial sum** R/W energy consumption
 - maximize local accumulation
- Examples:
 - [Gupta, *ICML* 2015] [ShiDianNao, *ISCA* 2015]
 - [ENVISION, *ISSCC* 2017] [Thinker, *JSSC* 2017]

No Local Reuse (NLR)



- Use a **large global buffer** as shared storage
 - Reduce **DRAM** access energy consumption
- **Examples:**

[DianNao, *ASPLOS* 2014] [DaDianNao, *MICRO* 2014]

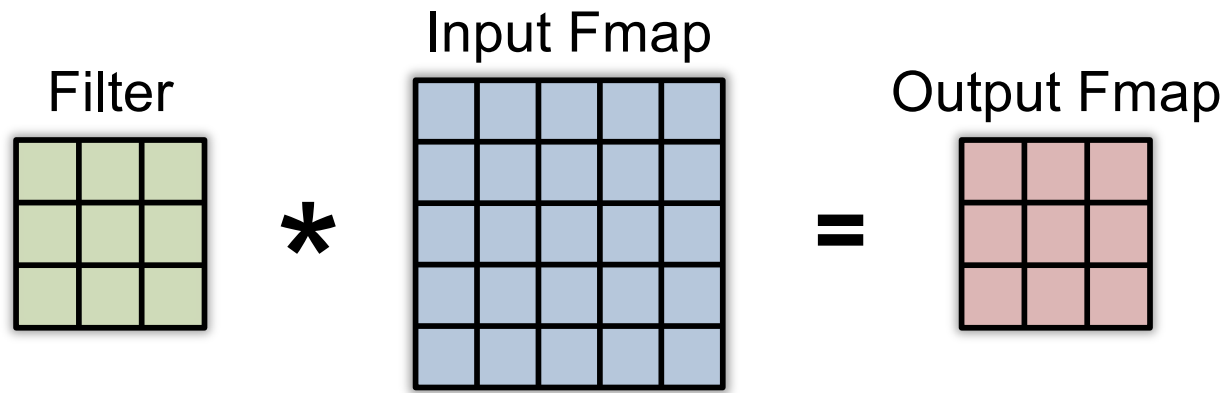
[Zhang, *FPGA* 2015]

Energy-Efficient Dataflow: Row Stationary (RS)

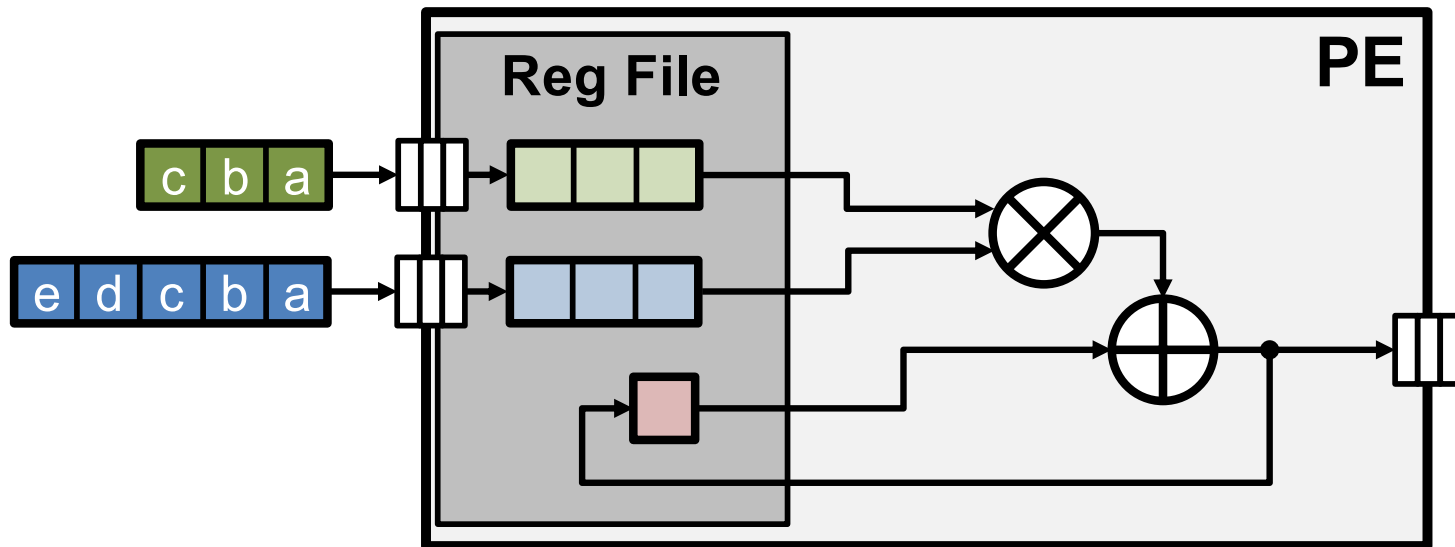
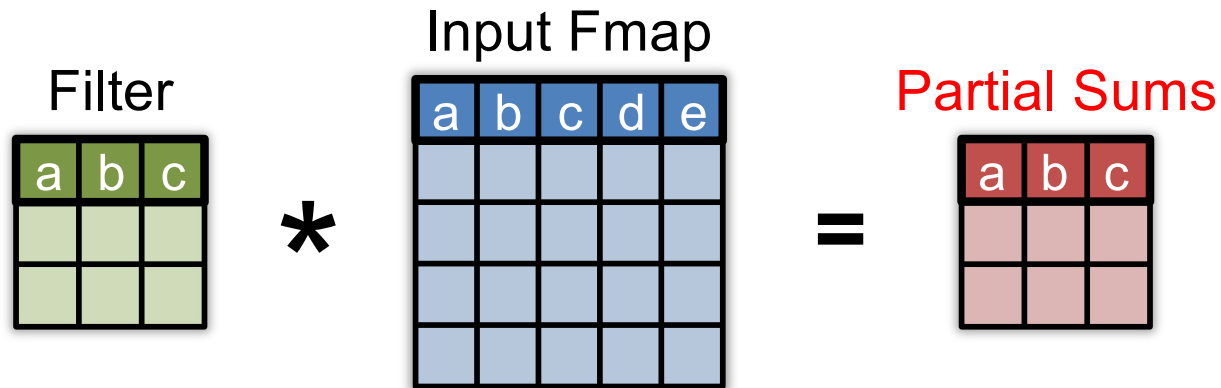
- **Maximize** reuse at RF
- Optimize for **overall** energy efficiency instead for *only* a certain data type

[Chen, ISCA 2016]

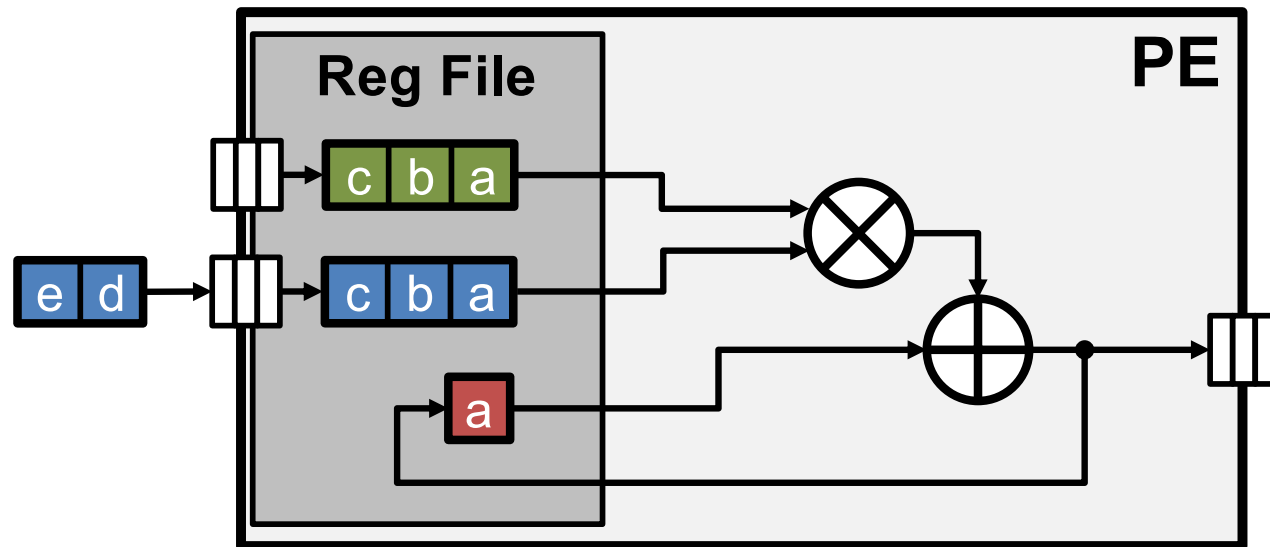
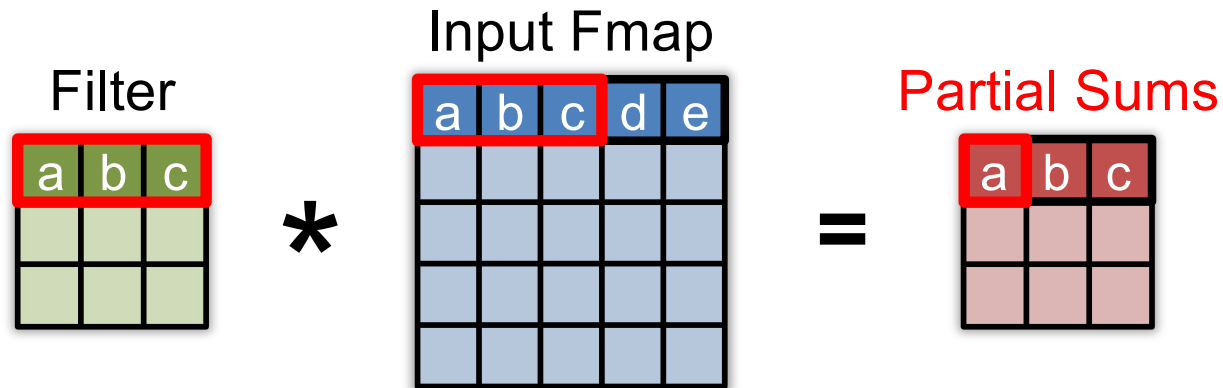
1D Row Convolution in PE



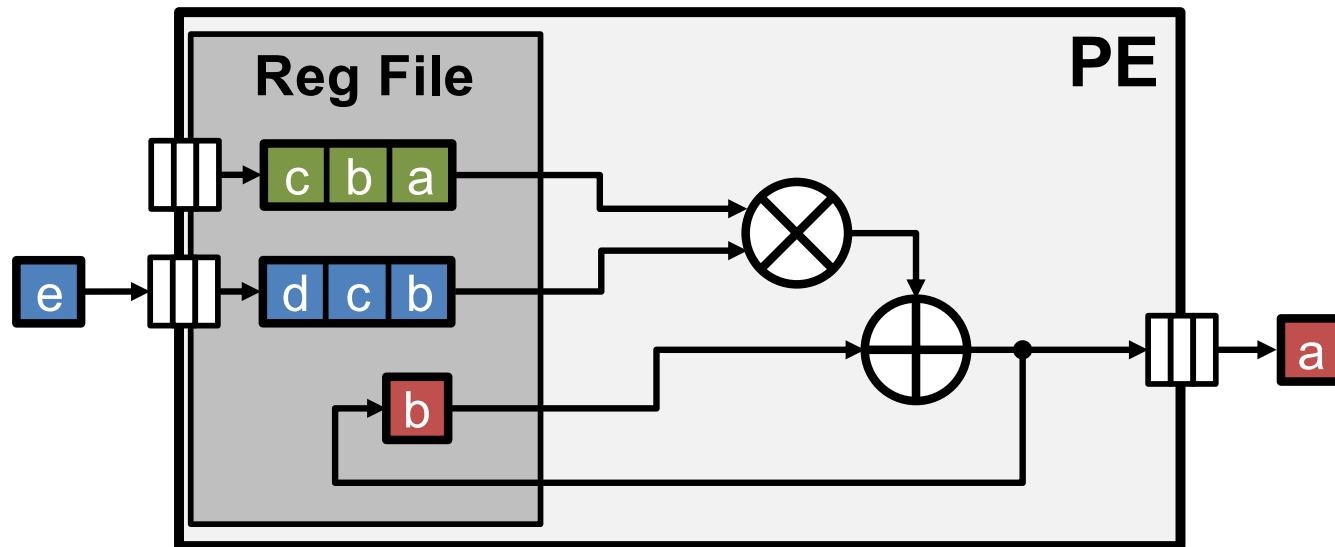
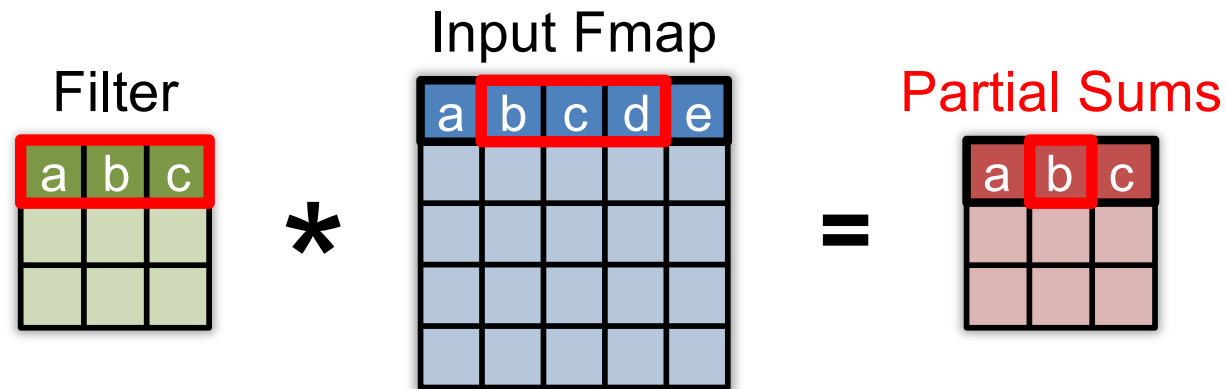
1D Row Convolution in PE



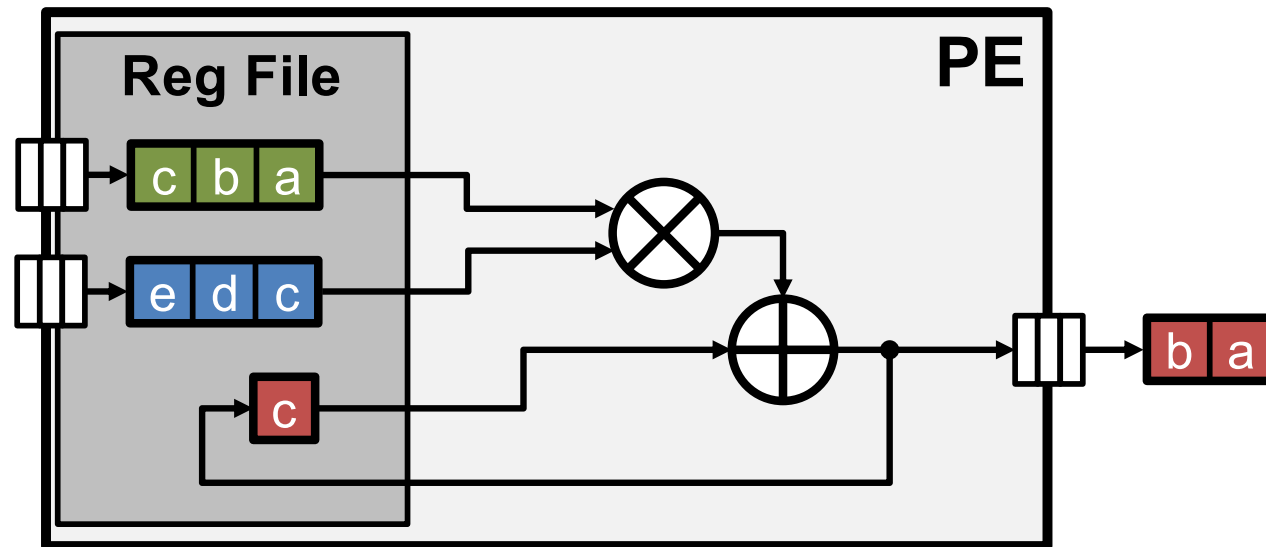
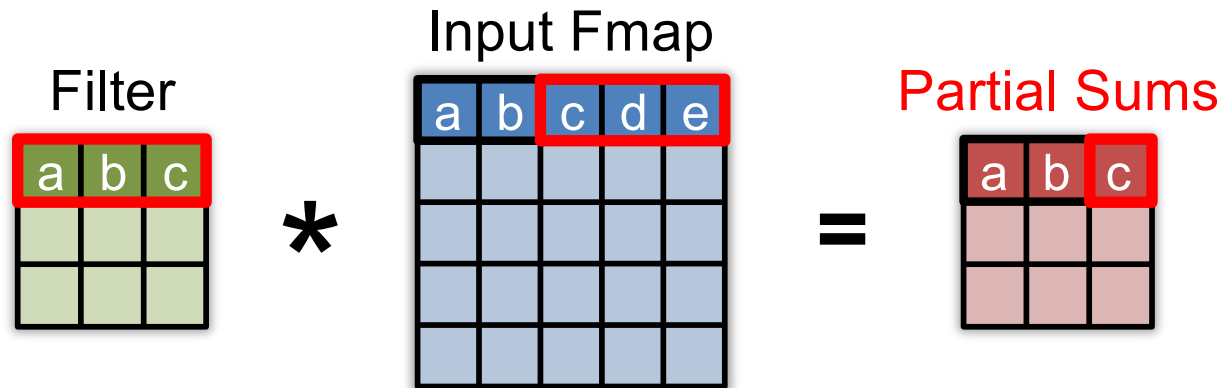
1D Row Convolution in PE



1D Row Convolution in PE

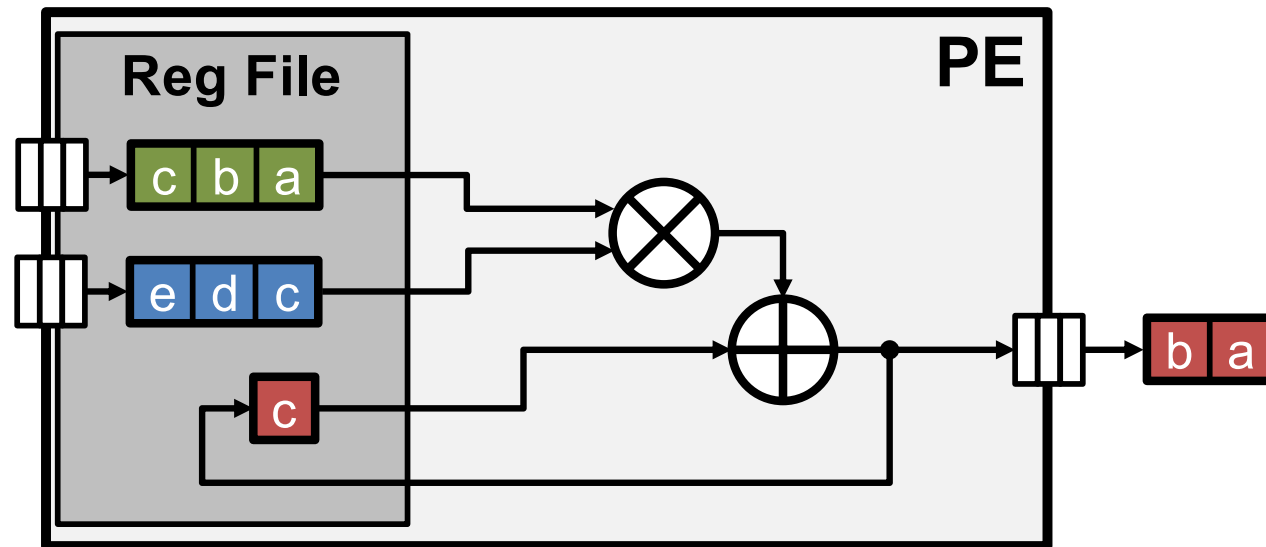


1D Row Convolution in PE

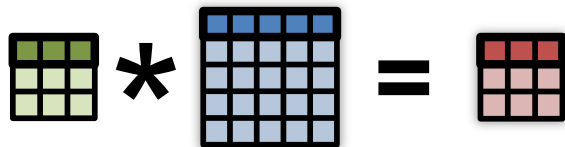


1D Row Convolution in PE

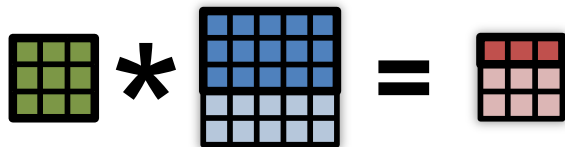
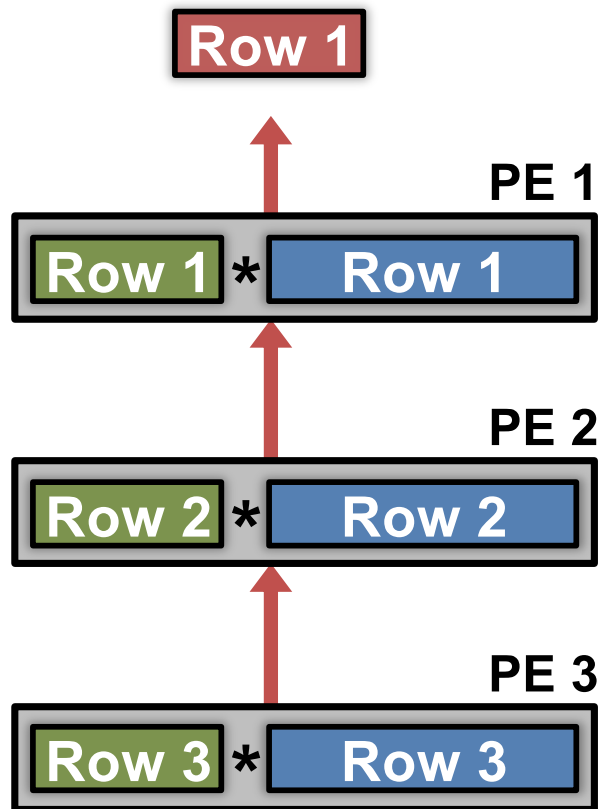
- Maximize row **convolutional reuse** in RF
 - Keep a **filter** row and **fmap** sliding window in RF
- Maximize row **psum accumulation** in RF



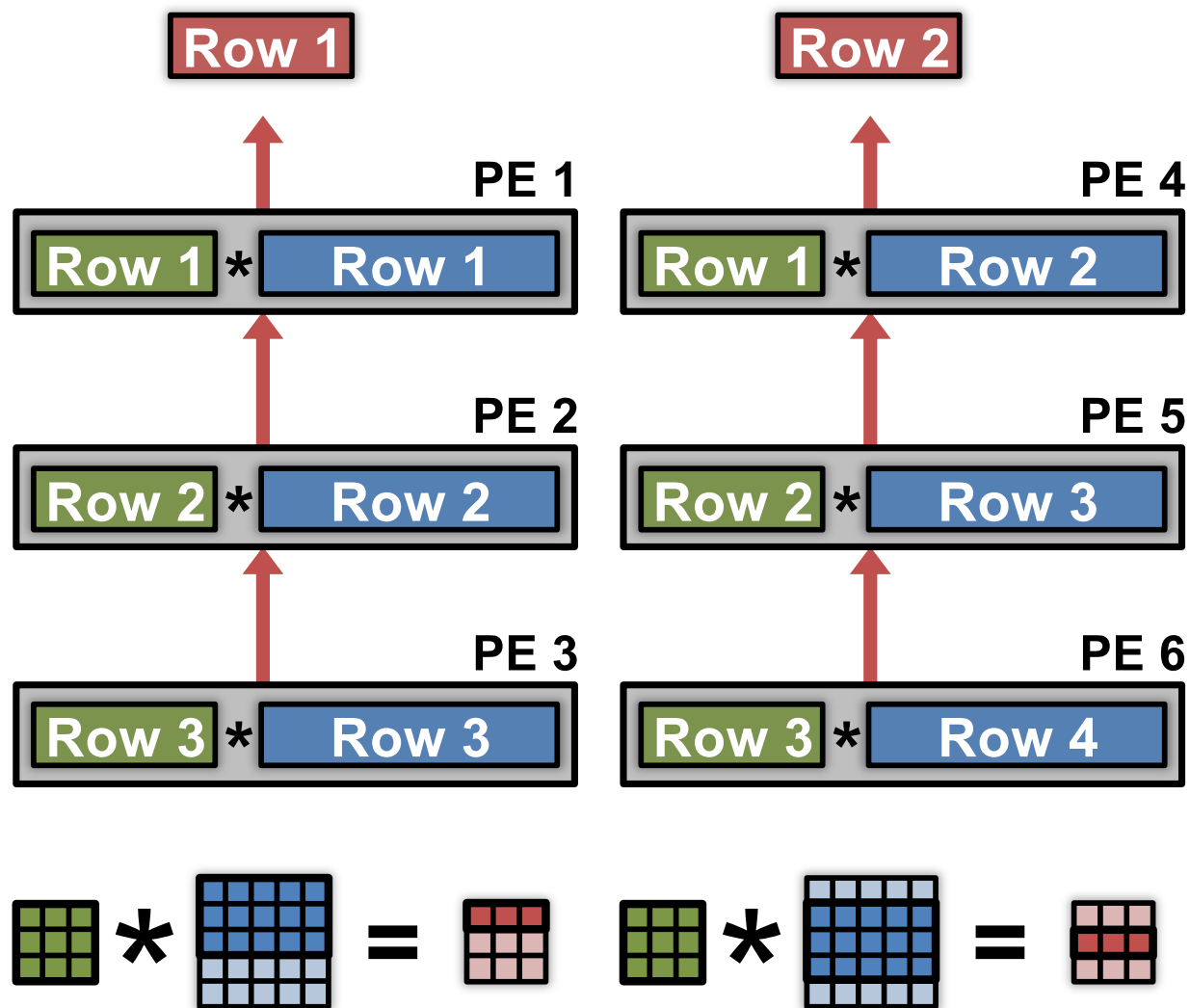
2D Convolution in PE Array



45 2D Convolution in PE Array

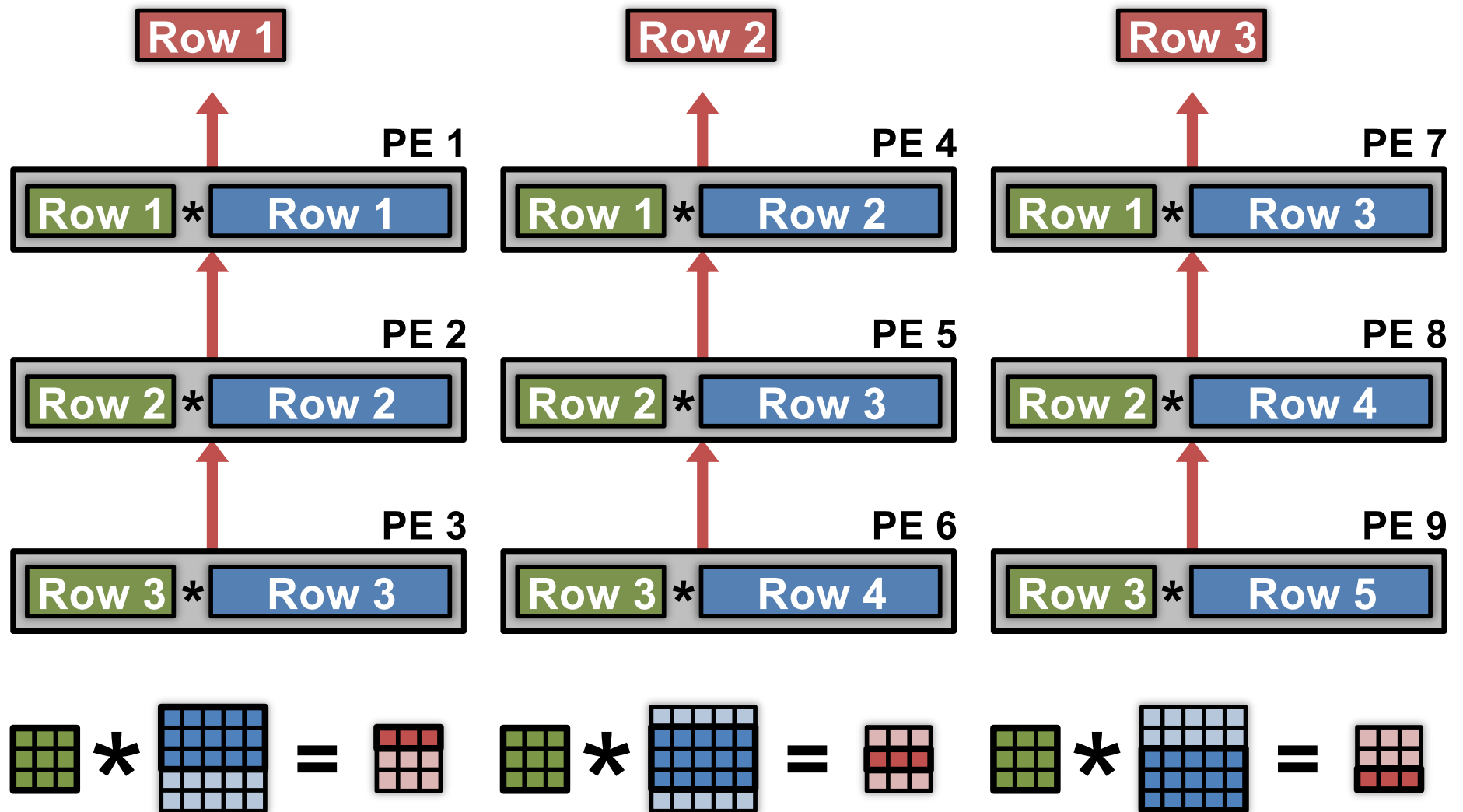


2D Convolution in PE Array

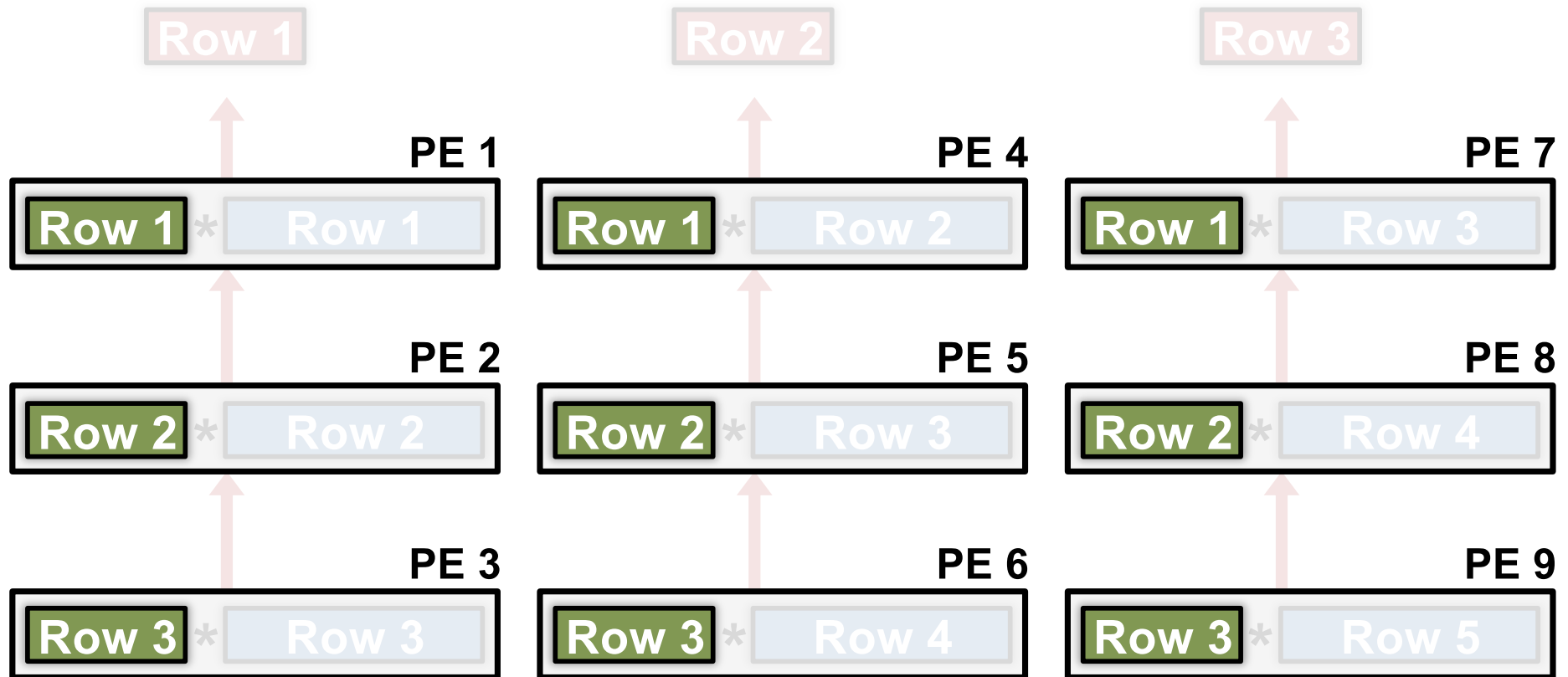


47

2D Convolution in PE Array

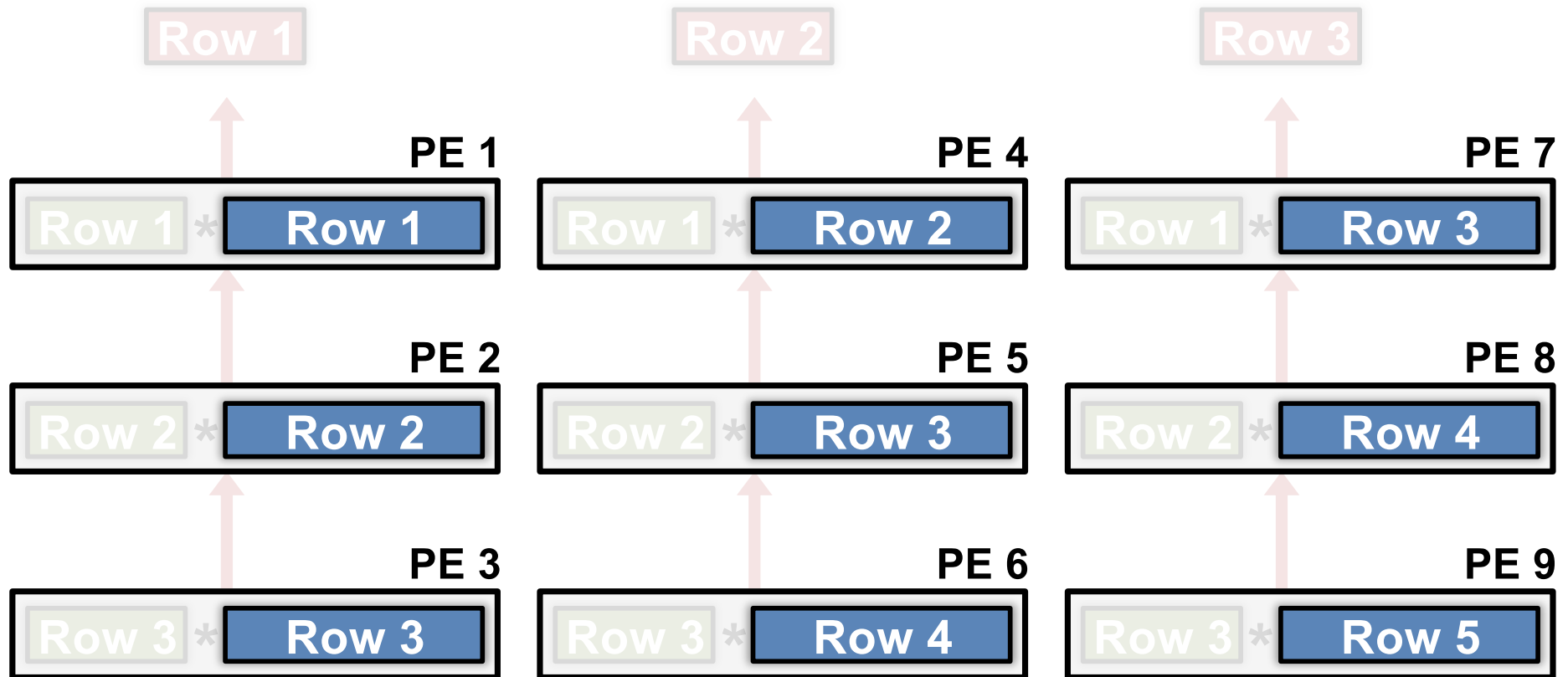


Convolutional Reuse Maximized



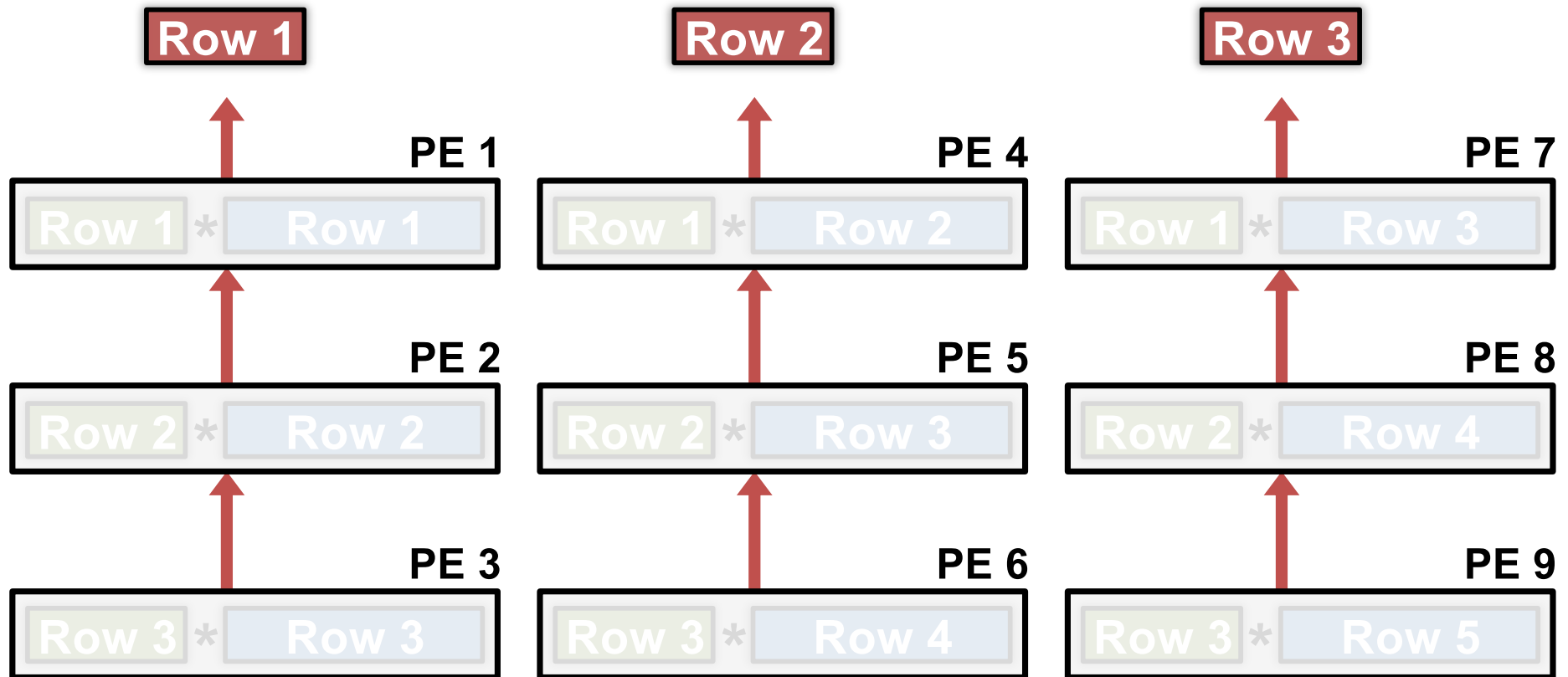
Filter rows are reused across PEs horizontally

Convolutional Reuse Maximized



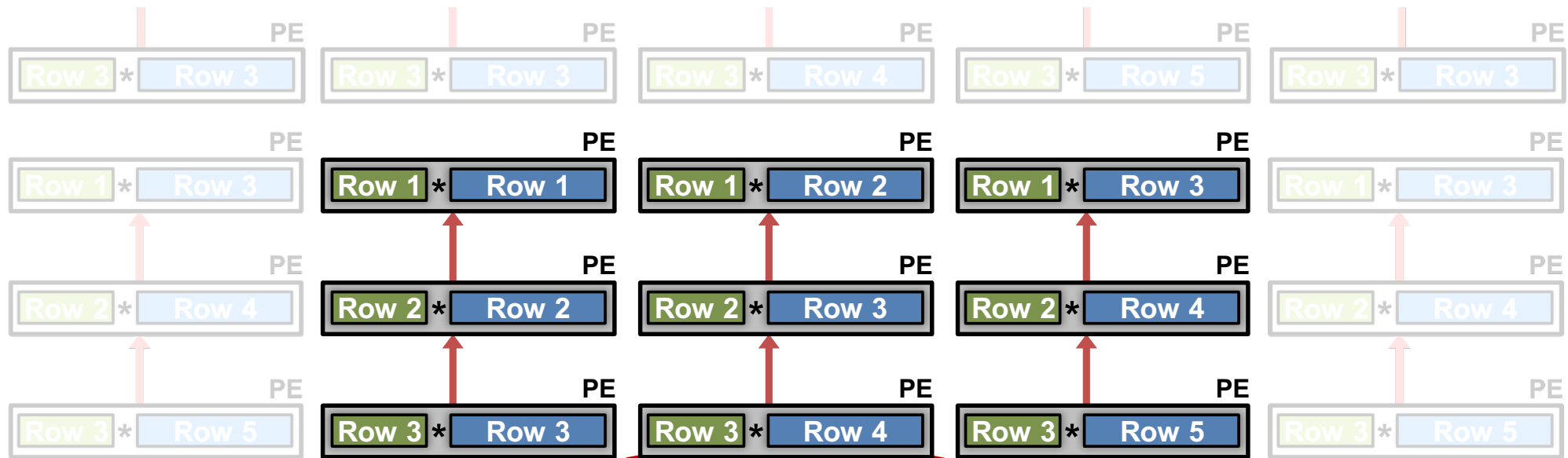
Fmap rows are reused across PEs diagonally

Maximize 2D Accumulation in PE Array



Partial sums accumulate across PEs **vertically**

Flexibility to Map Multiple Dimensions



Multiple **fmaps**:



Multiple **filters**:

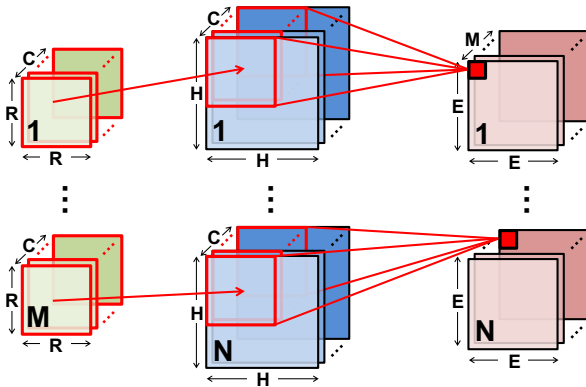


Multiple **channels**:

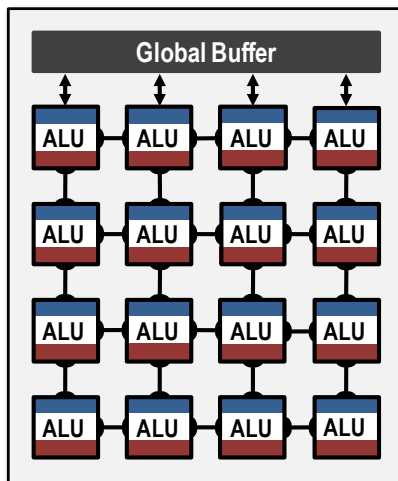


Finding the Optimal Mapping

DNN Configurations

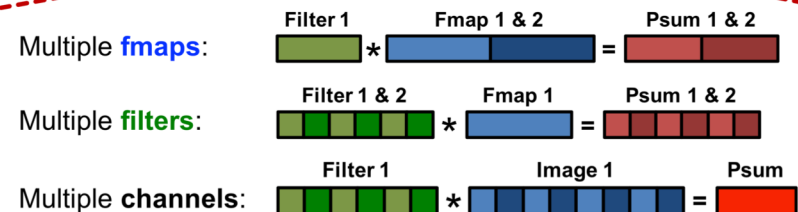
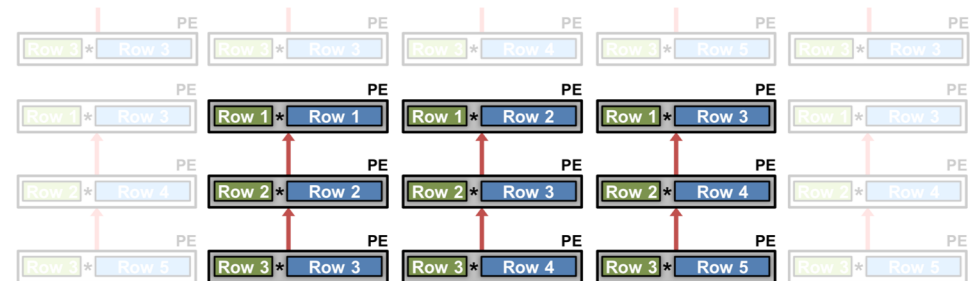


Hardware Resources



Optimization
Compiler

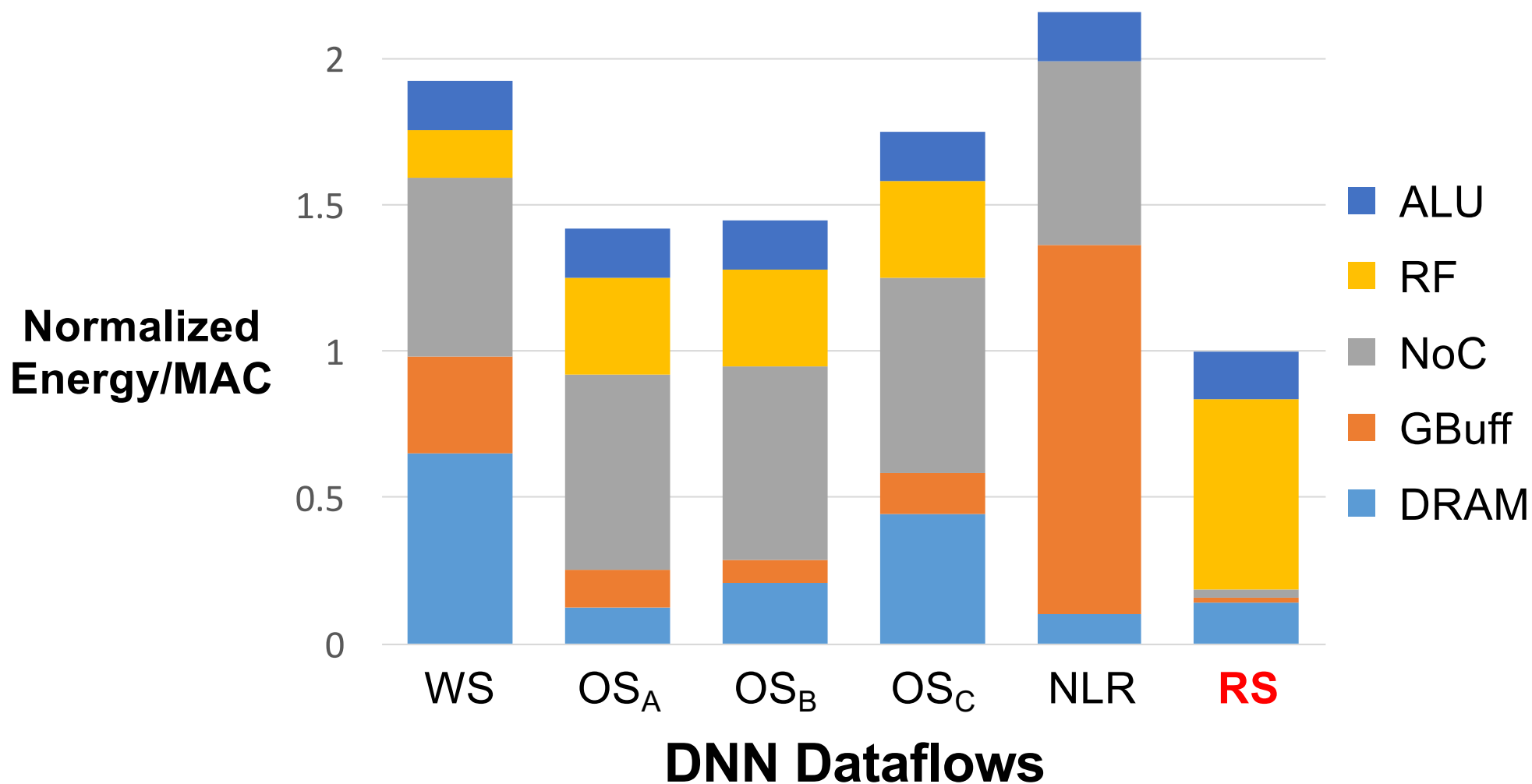
Row Stationary Mapping



Dataflow Simulation Results

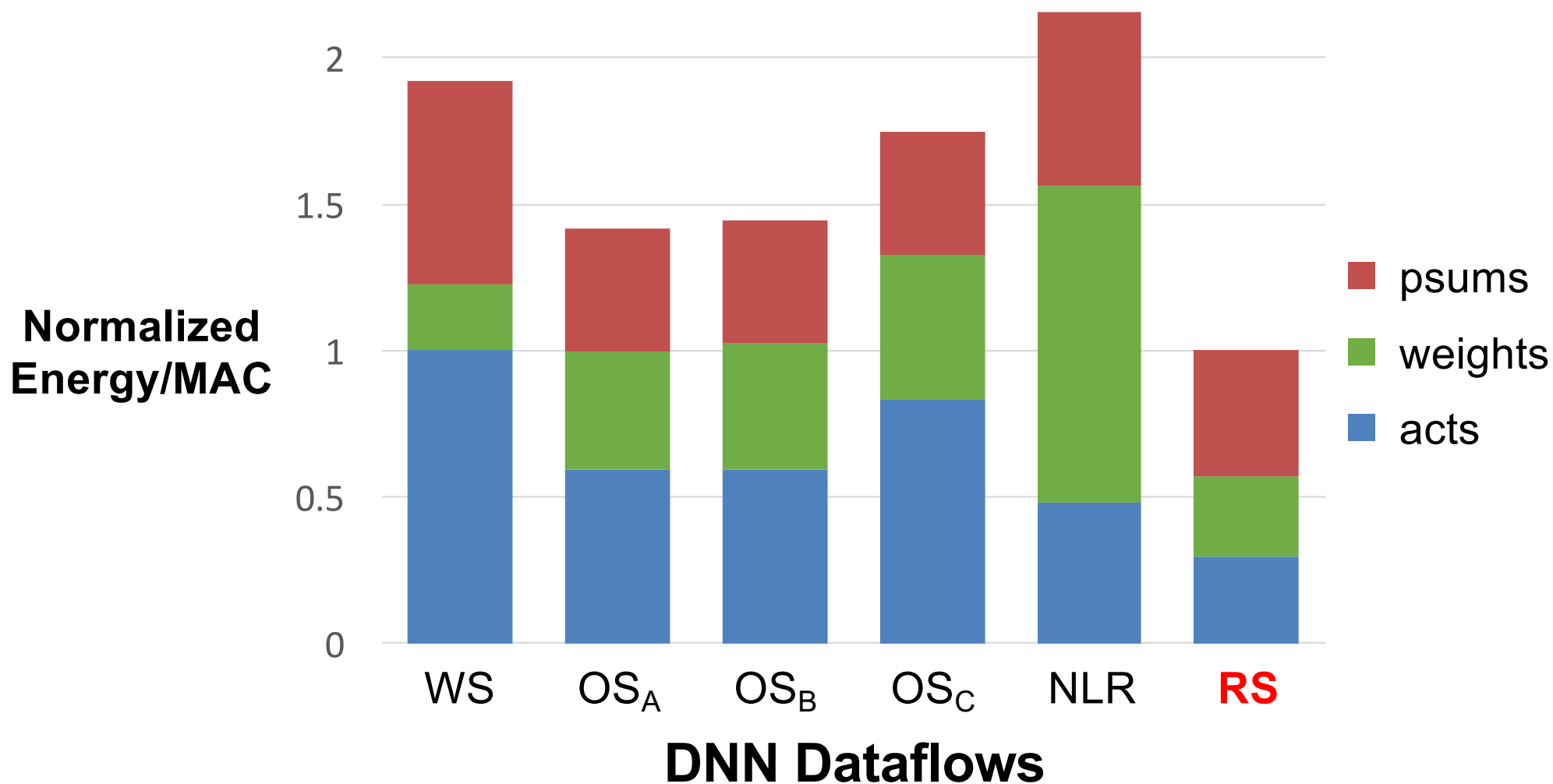
- Same total hardware area
- 256 PEs
- 16b AlexNet Configuration
- Batch size = 16

Dataflow Comparison: CONV Layers



RS uses **1.4× – 2.5× lower** energy than other dataflows

Dataflow Comparison: CONV Layers



RS optimizes for the best **overall** energy efficiency

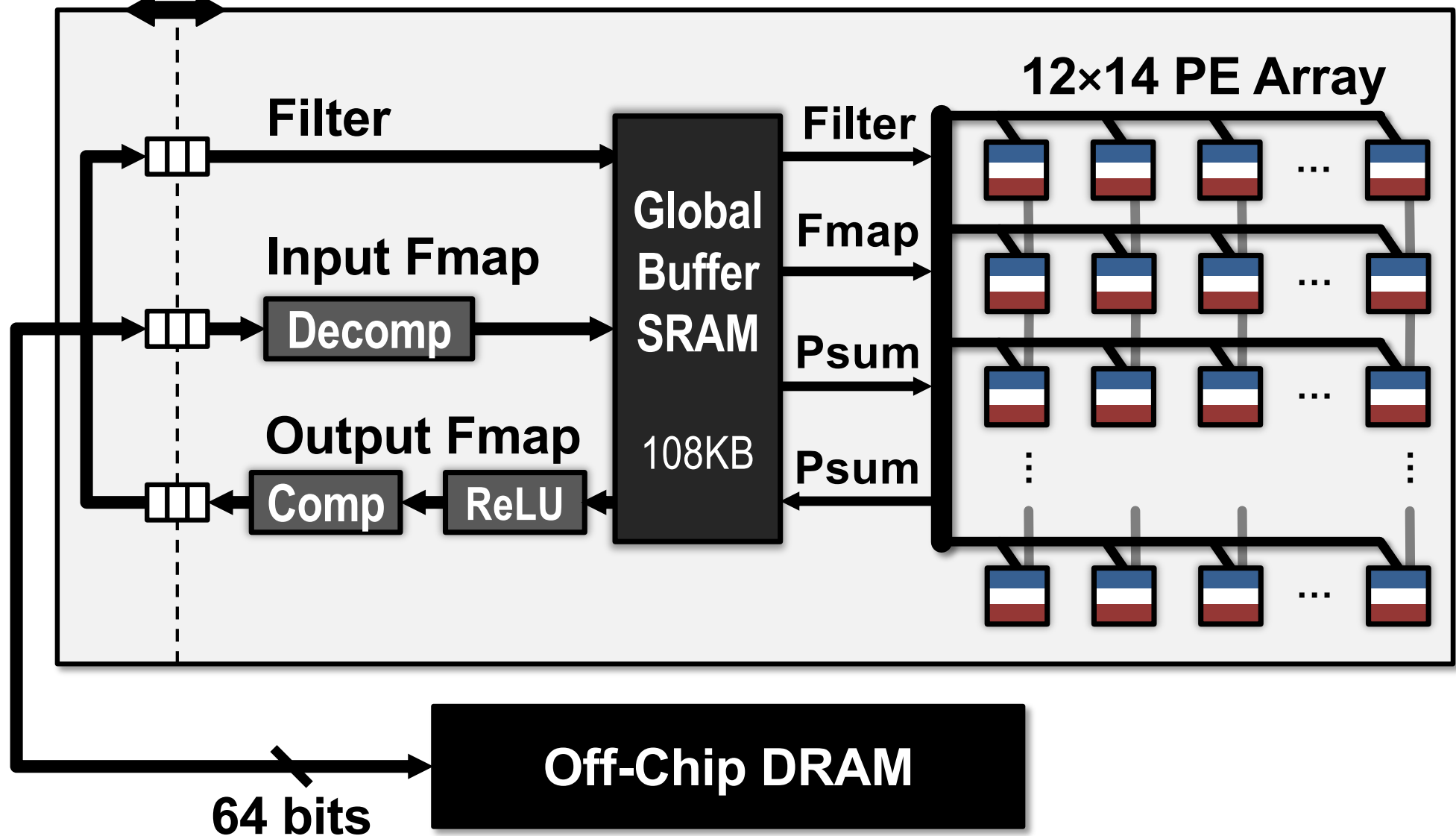
Eyeriss v1 Architecture

- **Optimizes Data Reuse** with RS dataflow
- **Improves PE Utilization** with flexible mappings
- **Reduces Processing Power & DRAM Traffic** with data sparsity

[Chen, *ISSCC* 2016, *JSSC* 2017]

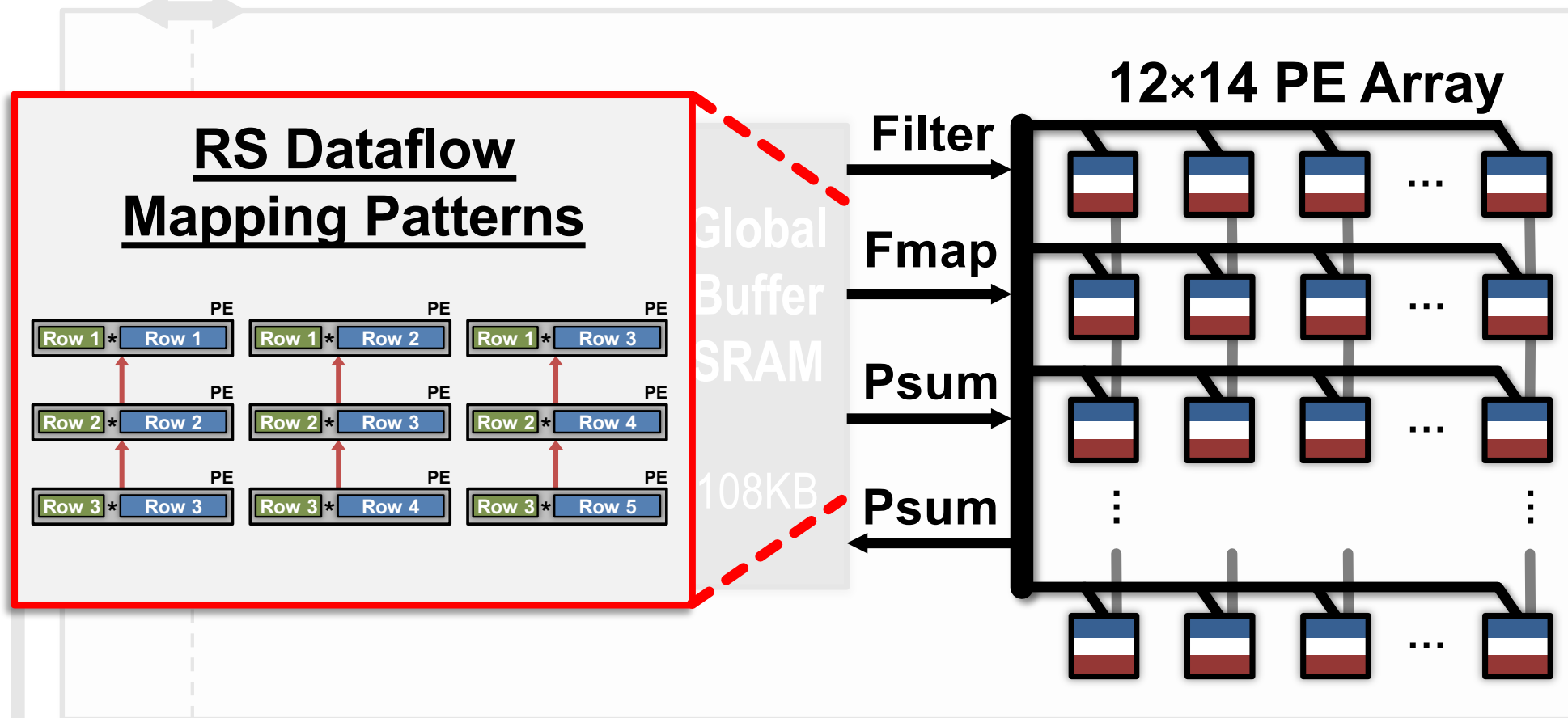
Eyeriss v1 Architecture for RS Dataflow

Link Clock | Core Clock



Flexibility Required for Mapping

Link Clock | Core Clock

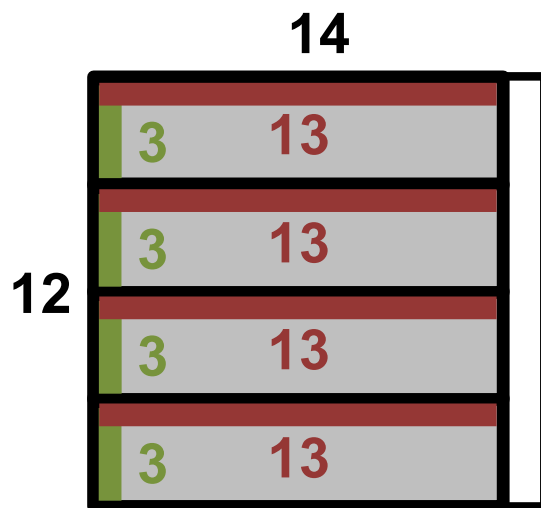
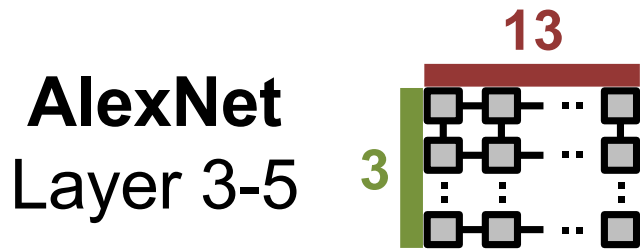


How to map **different shapes** with a **fixed-size PE array**?

64 bits

Flexible Mapping Strategy

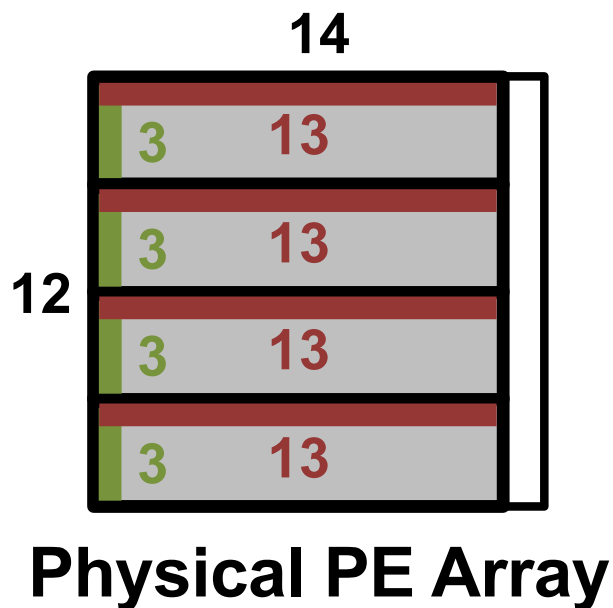
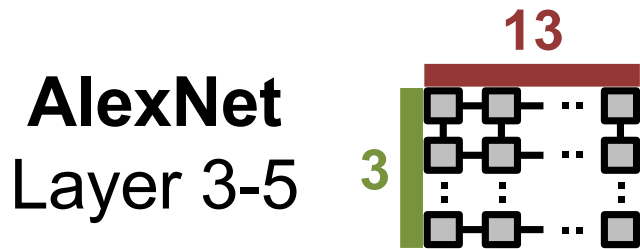
Replication



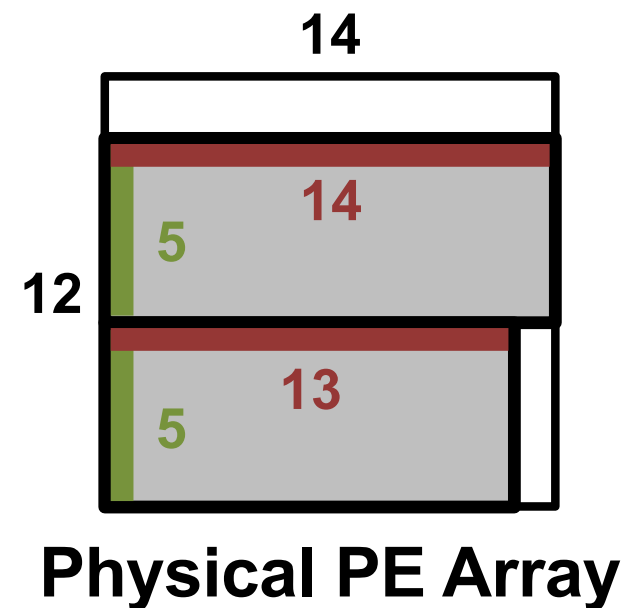
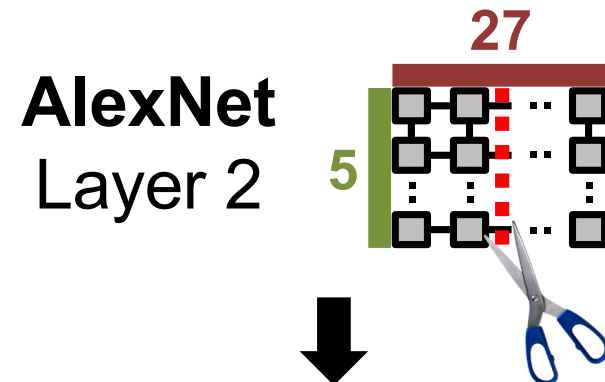
Physical PE Array

Flexible Mapping Strategy

Replication

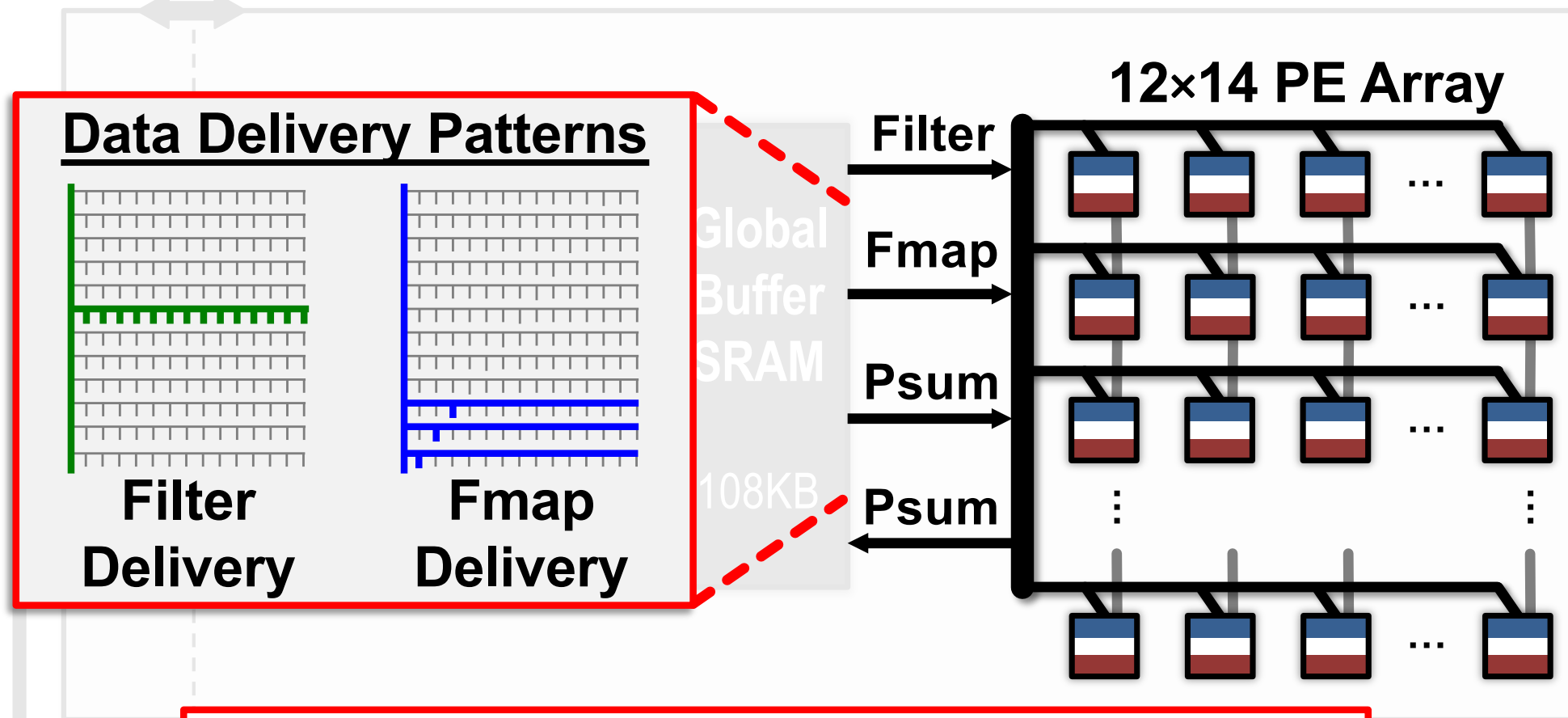


Folding



Multicast Network for Data Delivery

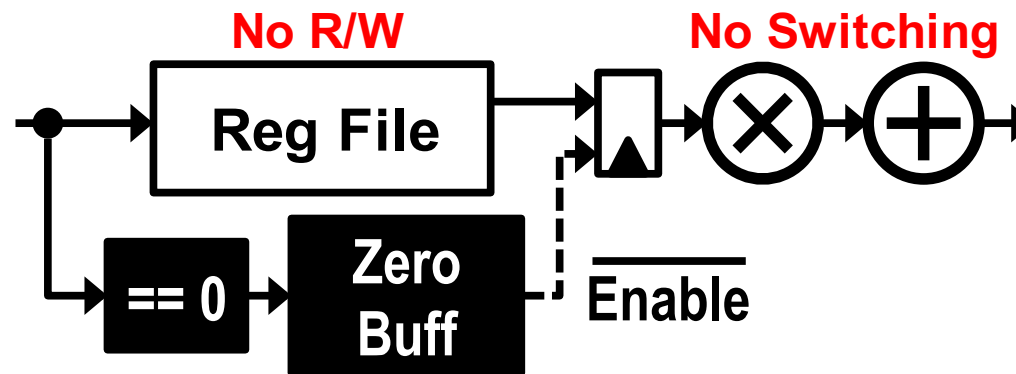
Link Clock | Core Clock



Supports any delivery pattern & saves over 80% energy compared to broadcast

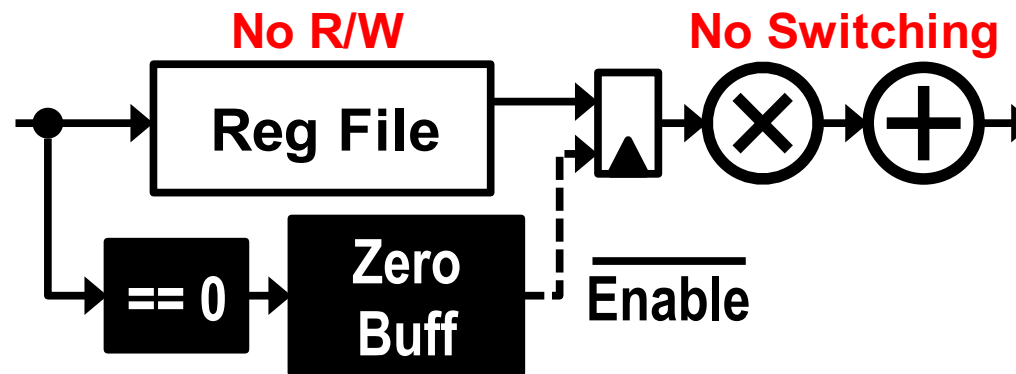
Exploit Data Sparsity

- Save **45%** PE power with **Zero-Gating Logic**

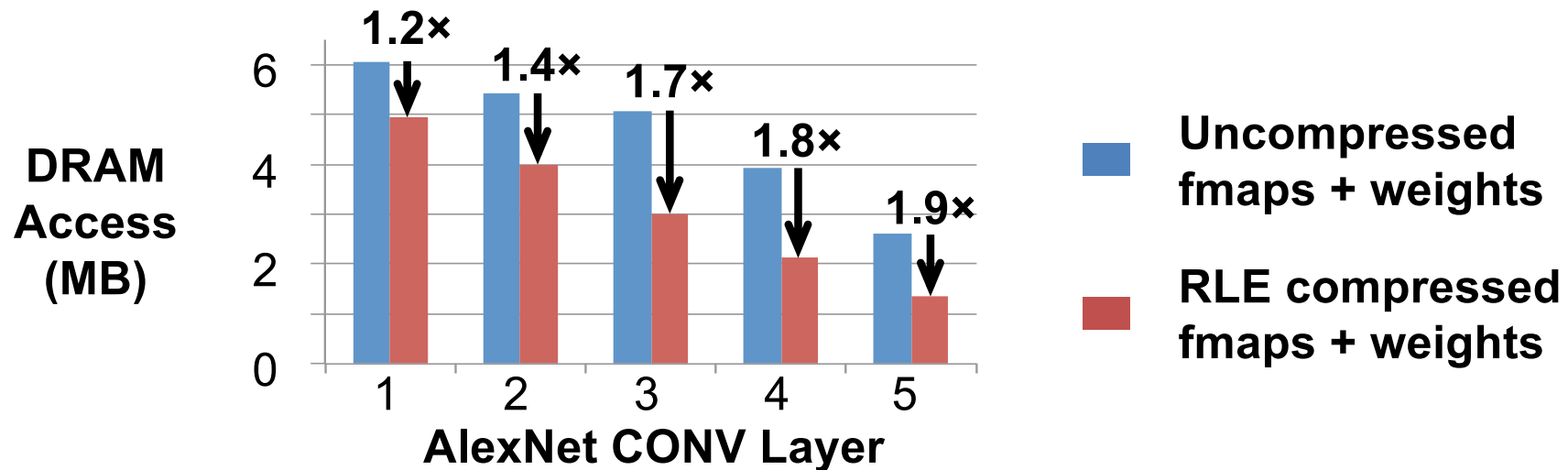


Exploit Data Sparsity

- Save **45%** PE power with **Zero-Gating Logic**

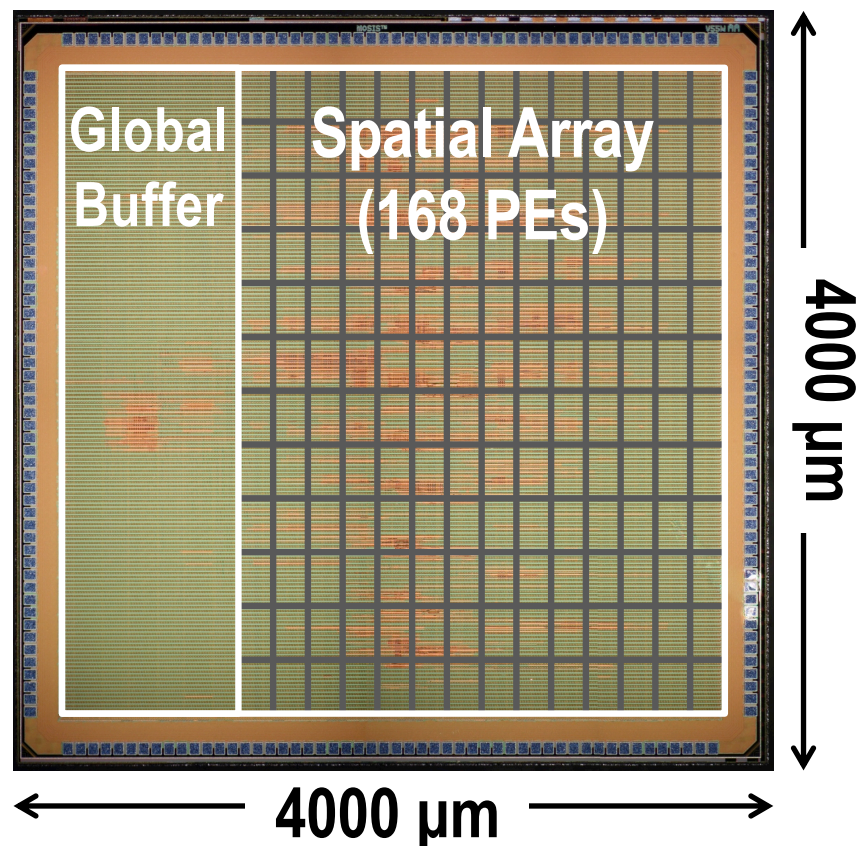


- Reduce off-chip DRAM traffic with **compression**

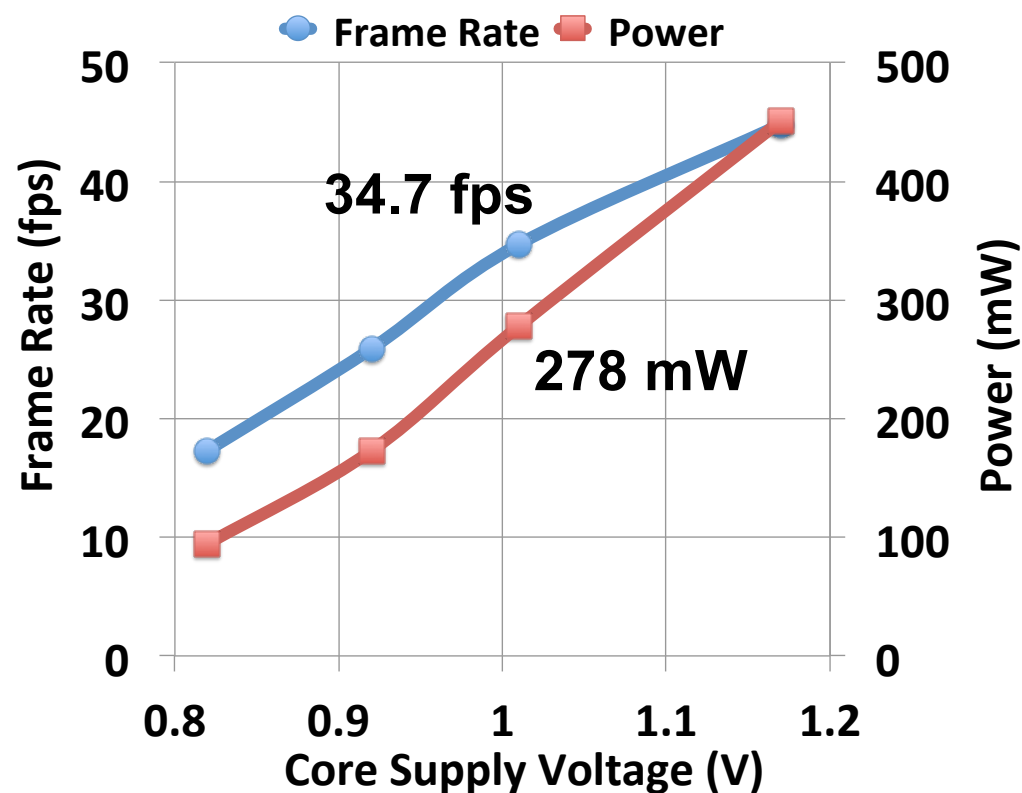


Eyeriss v1 Chip Measurement Results

Fabricated in a 65nm process



AlexNet CONV Layers



To execute 2.66 GMACs (**16GB** inputs and **5.4GB** outputs),
only accesses **208.5MB** GBuff and **15.4MB** DRAM

65

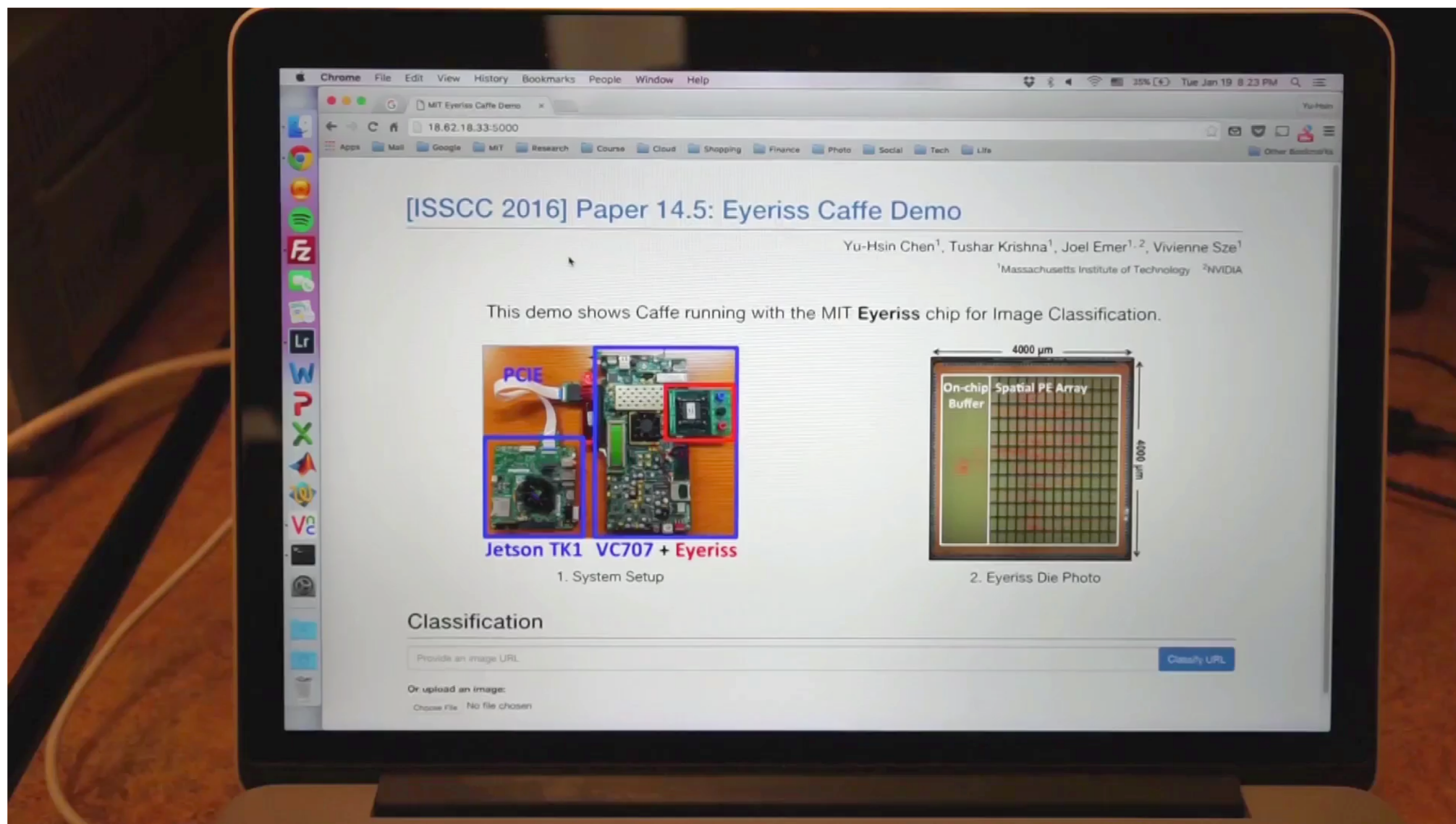
Comparison to a Mobile GPU

	<i>Eyeriss v1</i>	NVIDIA TK1 (Jetson Kit)
Technology	65nm	28nm
Clock Rate	200MHz	852MHz
# Multipliers	168	192
On-Chip Storage	Buffer: 108KB Spad: 75.3KB	Shared Mem: 64KB Reg File: 256KB
Word Bit-Width	16b Fixed	32b Float
Throughput¹	34.7 fps	68 fps
Measured Power	278 mW	Idle/Active ² : 3.7W/10.2W
Energy Efficiency	124.8 image/J	10.5 image/J
DRAM Bandwidth	127 MB/s	1120 MB/s ³

1. AlexNet CONV Layers Only
2. Board Power
3. Modeled from [Tan, SC 2011]

10× higher energy efficiency

Demo of Image Classification on Eyeriss



Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories
of DNN dataflow

Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories
of DNN dataflow

Row-Stationary (RS) Dataflow

1.4 – 2.5× more energy efficient
than existing DNN dataflows

Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories
of DNN dataflow

Row-Stationary (RS) Dataflow

1.4 – 2.5× more energy efficient
than existing DNN dataflows

Eyeriss v1 Architecture

Flexibility

Performance

Energy Efficiency

- Optimizes data reuse with RS dataflow (**F** **E**)

Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories of DNN dataflow

Row-Stationary (RS) Dataflow

1.4 – 2.5× more energy efficient than existing DNN dataflows

Eyeriss v1 Architecture

Flexibility

Performance

Energy Efficiency

- Optimizes data reuse with RS dataflow (**F** **E**)
- Saves 45% PE power & up to 1.9× DRAM traffic with sparsity (**E**)

Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories of DNN dataflow

Row-Stationary (RS) Dataflow

1.4 – 2.5× more energy efficient than existing DNN dataflows

Eyeriss v1 Architecture

Flexibility

Performance

Energy Efficiency

- Optimizes data reuse with RS dataflow (F E)
- Saves 45% PE power & up to 1.9× DRAM traffic with sparsity (E)
- Improves PE utilization with flexible mapping strategy (F P)

Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories of DNN dataflow

Row-Stationary (RS) Dataflow

1.4 – 2.5× more energy efficient than existing DNN dataflows

Eyeriss v1 Architecture

Flexibility

Performance

Energy Efficiency

- Optimizes data reuse with RS dataflow (F E)
- Saves 45% PE power & up to 1.9× DRAM traffic with sparsity (E)
- Improves PE utilization with flexible mapping strategy (F P)
- Supports any delivery patterns with multicast NoC while saving over 80% energy compared to broadcast (F E)

Eyeriss v1: Summary of Contributions

Dataflow Taxonomy

3 major categories of DNN dataflow

Row-Stationary (RS) Dataflow

1.4 – 2.5× more energy efficient than existing DNN dataflows

Eyeriss v1 Architecture

Flexibility

Performance

Energy Efficiency

- **10× more energy-efficient** than a mobile GPU!
- Demonstrated an **image classification system** with the Eyeriss v1 chip

Survey on Efficient Processing of DNNs

Hardware Architectures for Deep Neural Networks

ISCA Tutorial

June 24, 2017

Website: <http://eyeriss.mit.edu/tutorial.html>



Massachusetts
Institute of
Technology



NVIDIA

<http://eyeriss.mit.edu/tutorial.html>

December 2017 | Volume 105 | Number 12

Proceedings OF THE IEEE

Efficient Processing of Deep Neural Networks: A Tutorial and Survey

System Scaling With Nanostructured Power and RF Components

Nonorthogonal Multiple Access for 5G and Beyond

Point of View: Beyond Smart Grid—A Cyber-Physical-Social System in Energy Future

Scanning Our Past: Materials Science, Instrument Knowledge, and the Power Source Renaissance

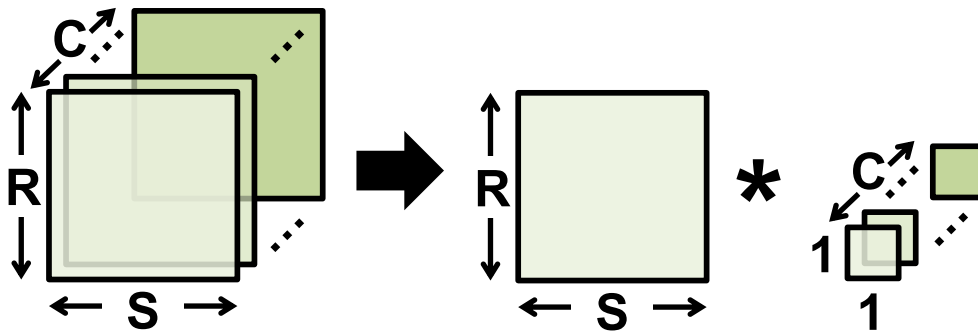


V. Sze, Y.-H. Chen,
T.-J. Yang, J. Emer,
*“Efficient Processing of
Deep Neural Networks:
A Tutorial and Survey,”
Proceedings of the IEEE,
Dec. 2017*

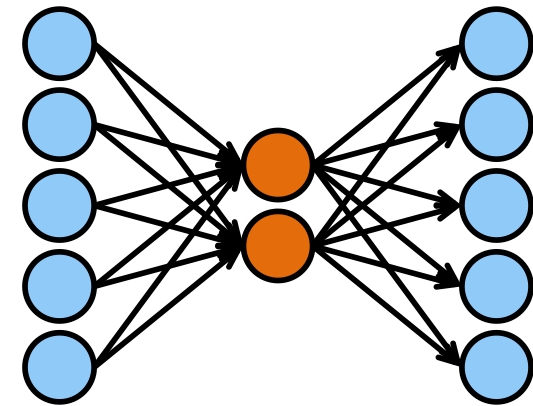
We identified various **limitations** to existing approaches

DNNs are Becoming More Compact!

Filter Decomposition



Bottleneck Layer



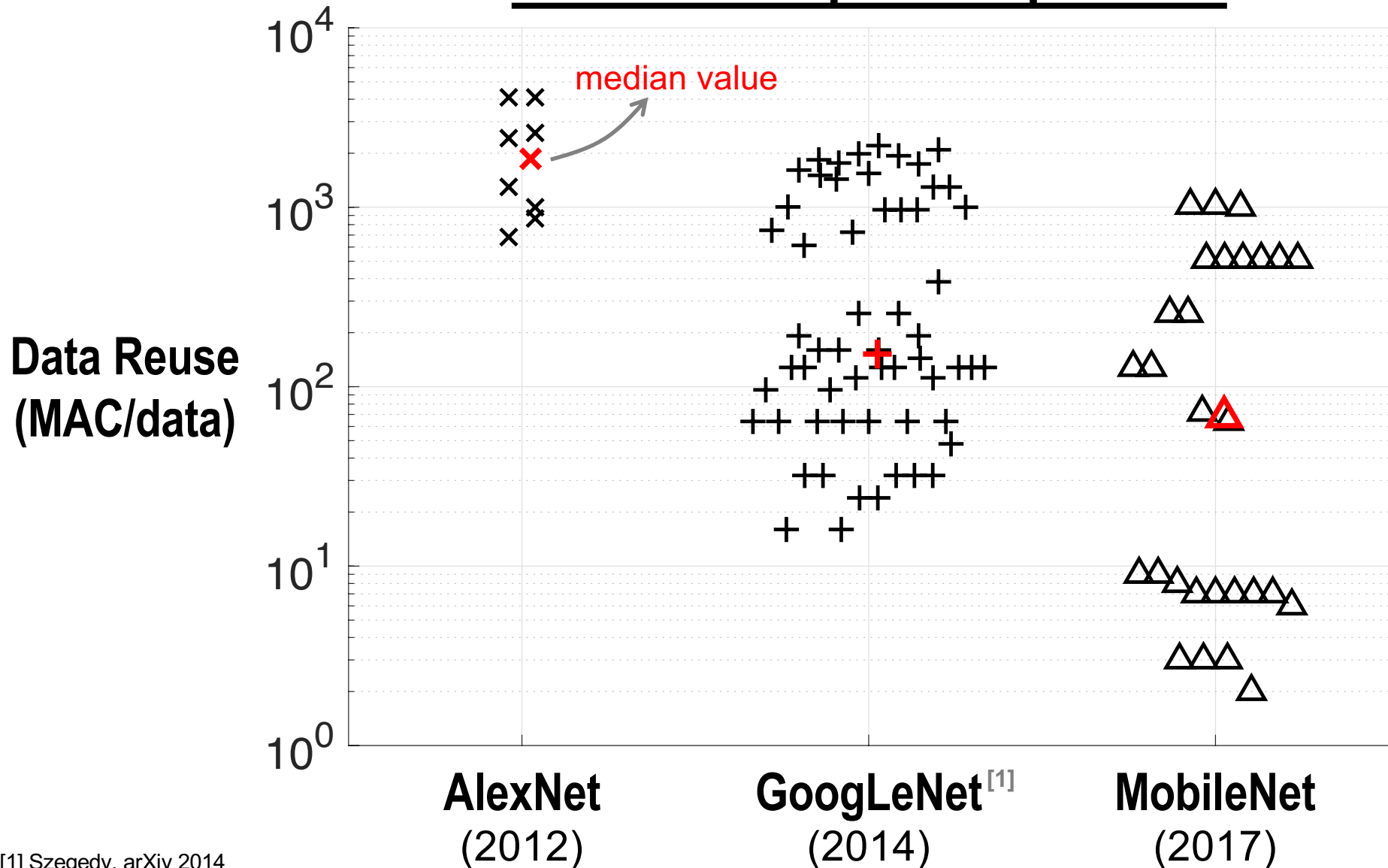
	Year	Accuracy*	# Layers	# Weights	# MACs
AlexNet	2012	80.4%	8	61M	724M
MobileNet ^[1]	2017	89.5%	28	4M	569M

* ImageNet Classification Top-5

[1] Howard, *arXiv* 2017

Data Reuse Going Against Our Favor

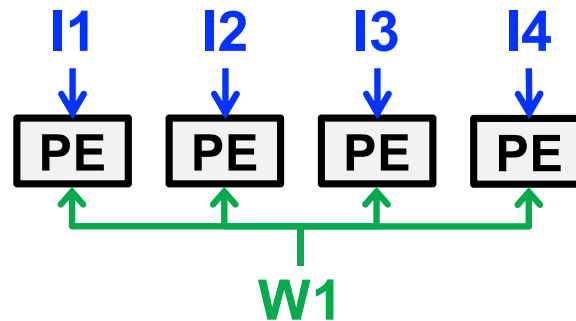
Amount of Input Fmap Reuse



[1] Szegedy, arXiv 2014

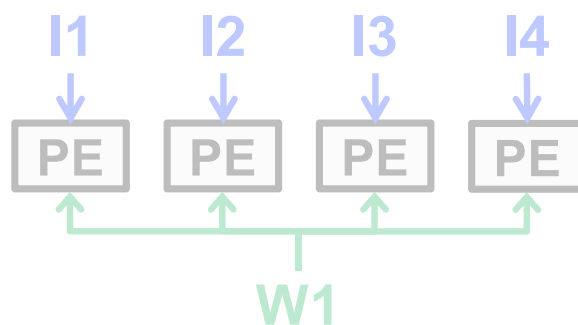
How Does Reuse Affect Performance?

Example: reuse the same **Weight** with different **Inputs**



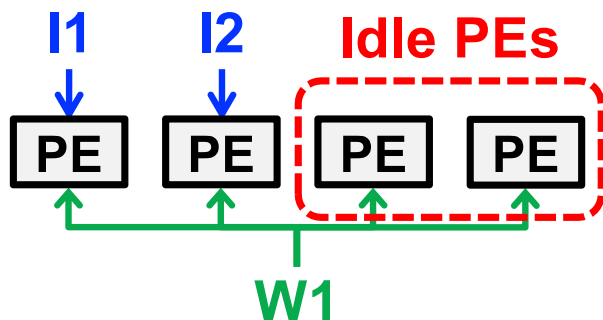
How Does Reuse Affect Performance?

Example: reuse the same **Weight** with different **Inputs**



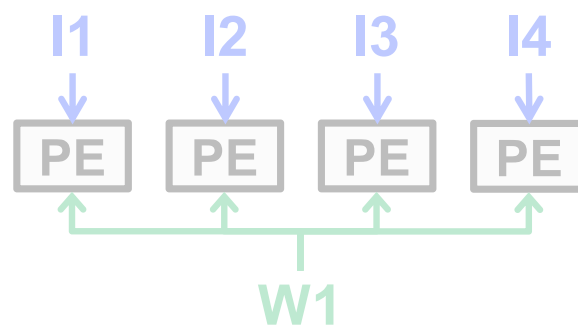
If weight reuse is low, **performance will go down!**

Case 1: Dataflow not flexible enough



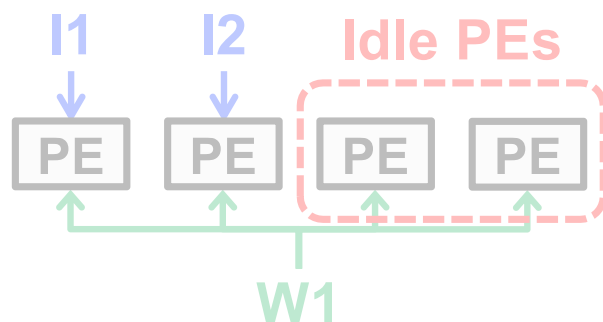
How Does Reuse Affect Performance?

Example: reuse the same **Weight** with different **Inputs**

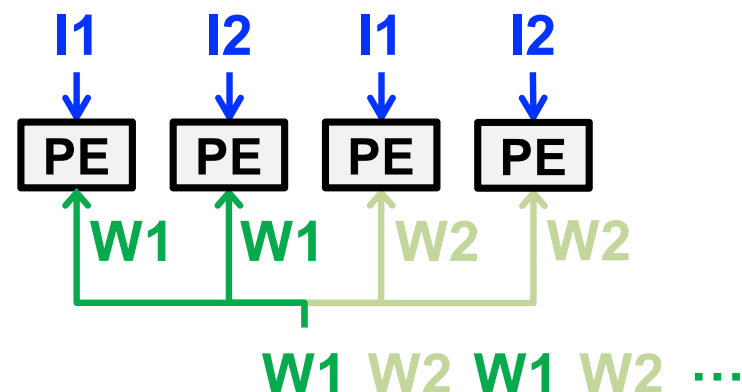


If weight reuse is low, **performance will go down!**

Case 1: Dataflow not flexible enough

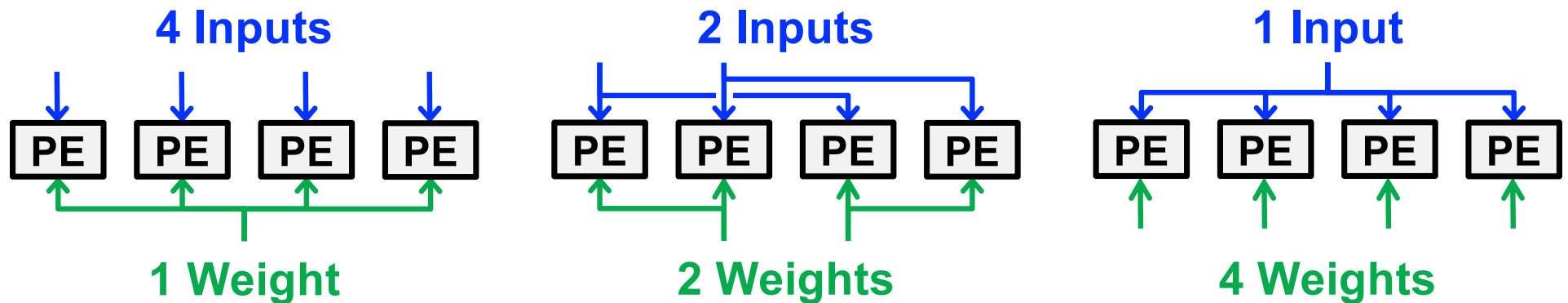


Case 2: Insufficient NoC bandwidth



A More Flexible Mapping Strategy

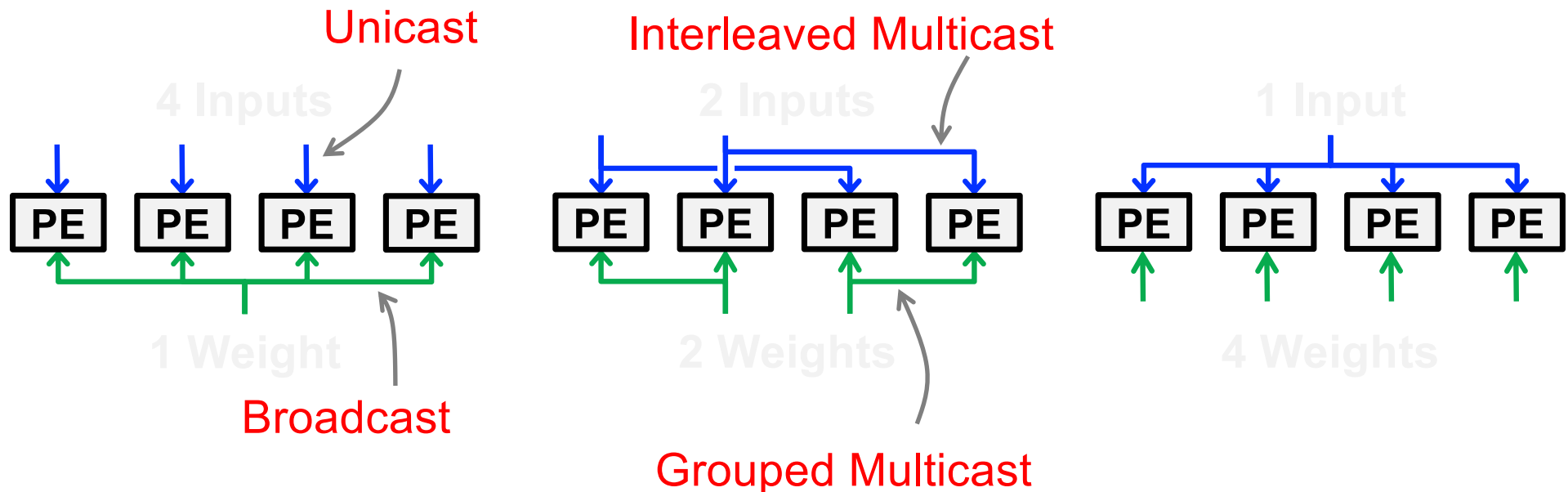
- **Dataflow:** always keep as many PEs busy as possible
- **NoC:** adapt to the bandwidth and reuse requirements



A More Flexible Mapping Strategy

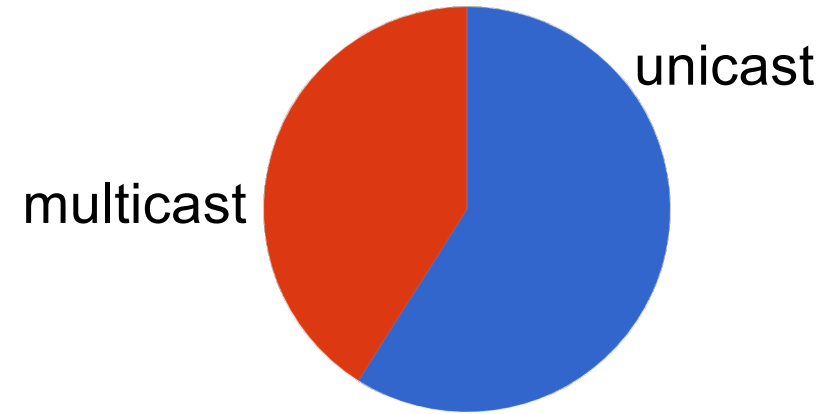
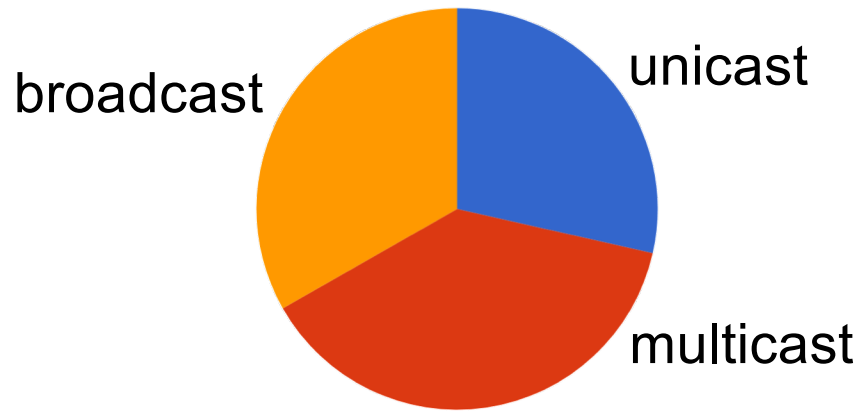
- **Dataflow**: always keep as many PEs busy as possible
- **NoC**: adapt to the bandwidth and reuse requirements

4 Data Delivery Patterns



Delivery of Input Fmaps (RS)

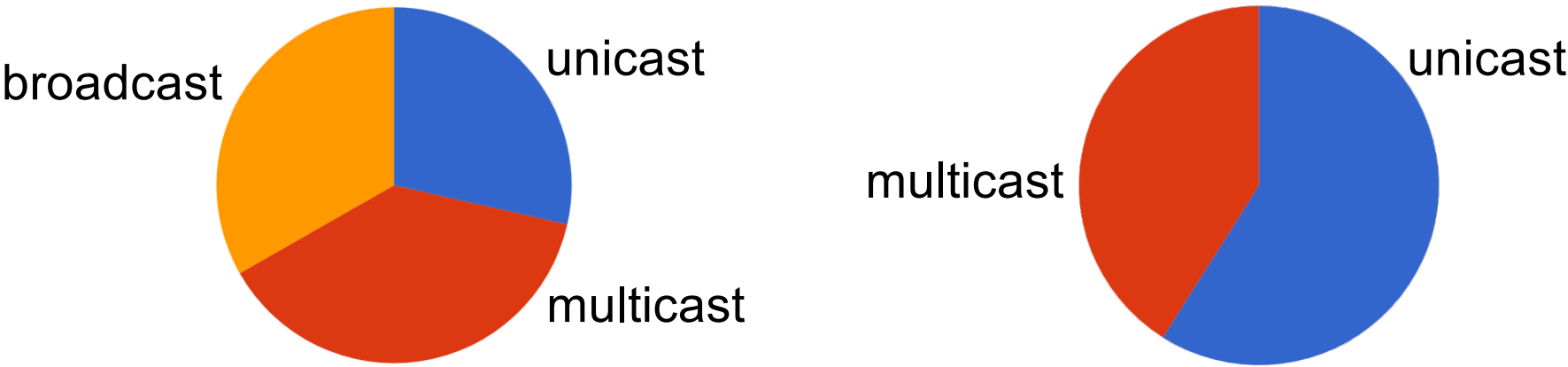
Horizontal — **AlexNet** — Vertical



Horizontal

AlexNet

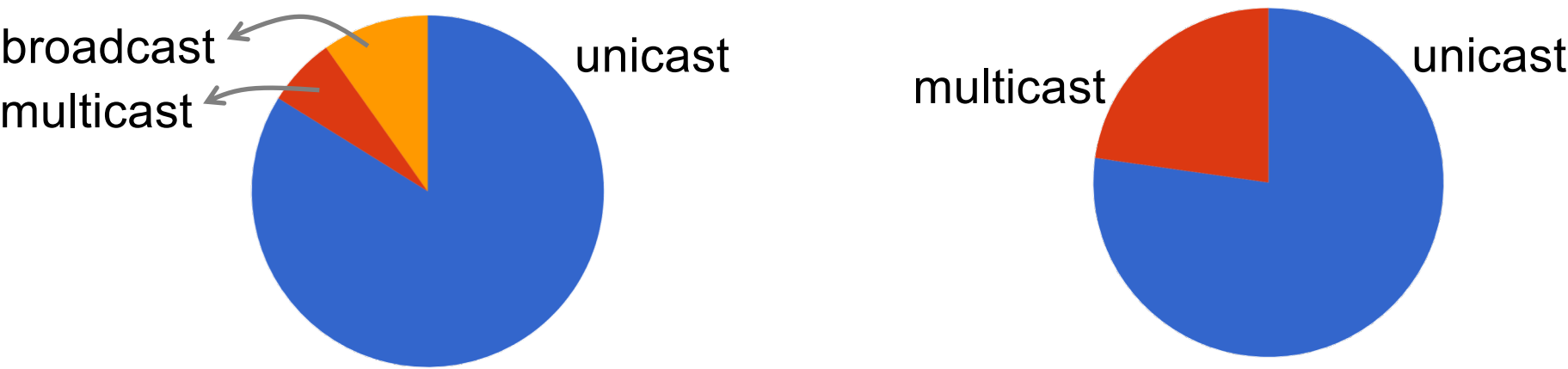
Vertical



Horizontal

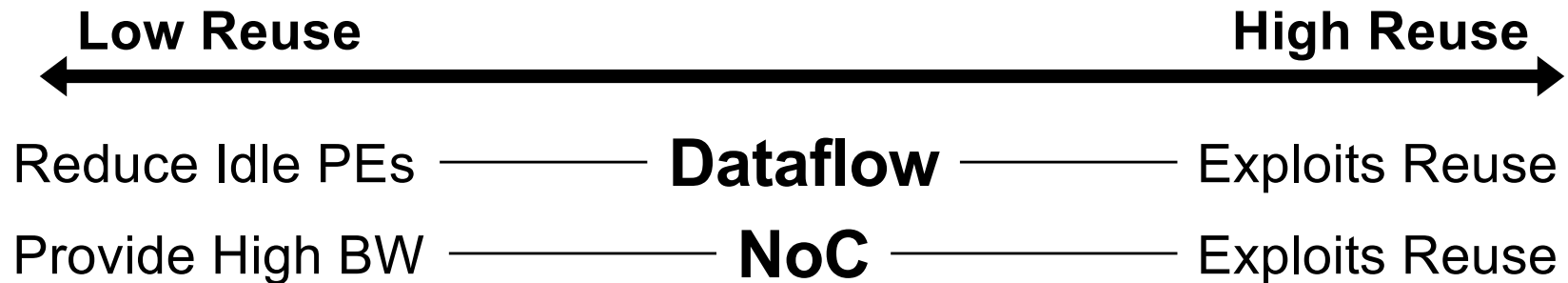
MobileNet

Vertical



Eyeriss v2: Design Goals

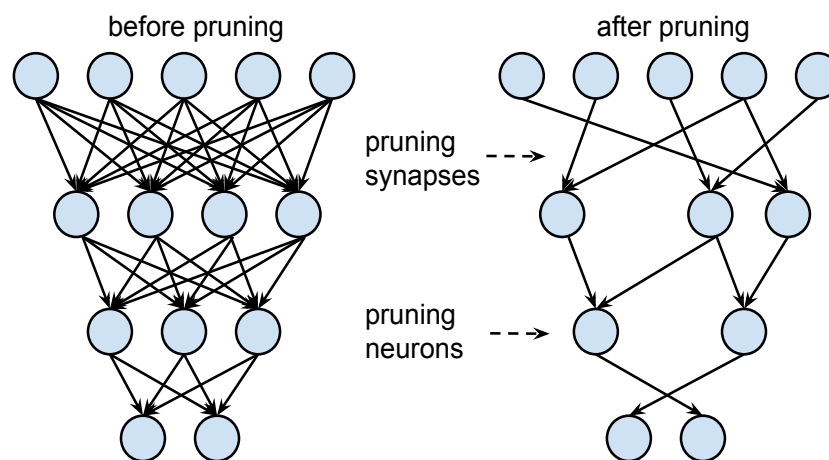
- Supports a wide range of data reuse



[Chen, submitted to *Micro* 2018]

DNNs are Becoming More Compact!

Network Pruning



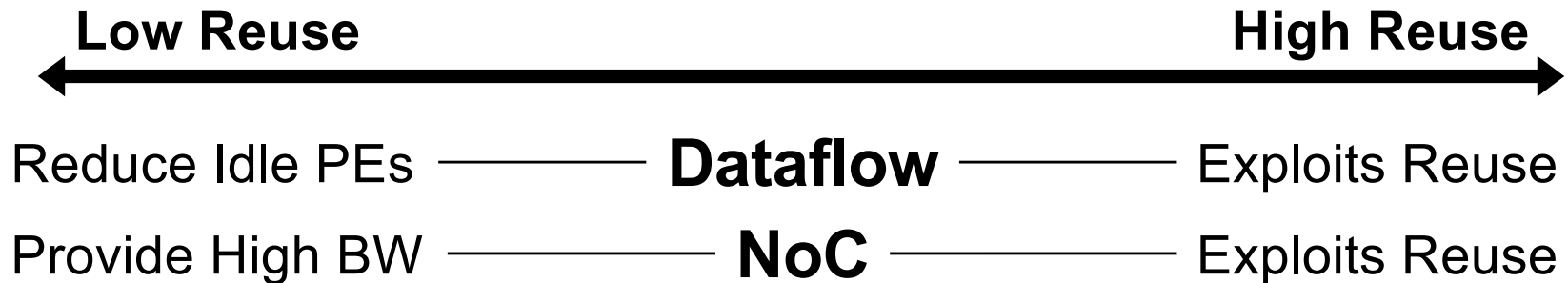
	Year	Accuracy*	# Layers	# Weights	# MACs
AlexNet	2012	80.4%	8	61M	724M
AlexNet^[1] (pruned)	2017	79.6%	8	5.7M (non-zero)	58M (non-zero)

* ImageNet Classification Top-5

[1] Yang, CVPR 2017

Eyeriss v2: Design Goals

- Supports a wide range of data reuse

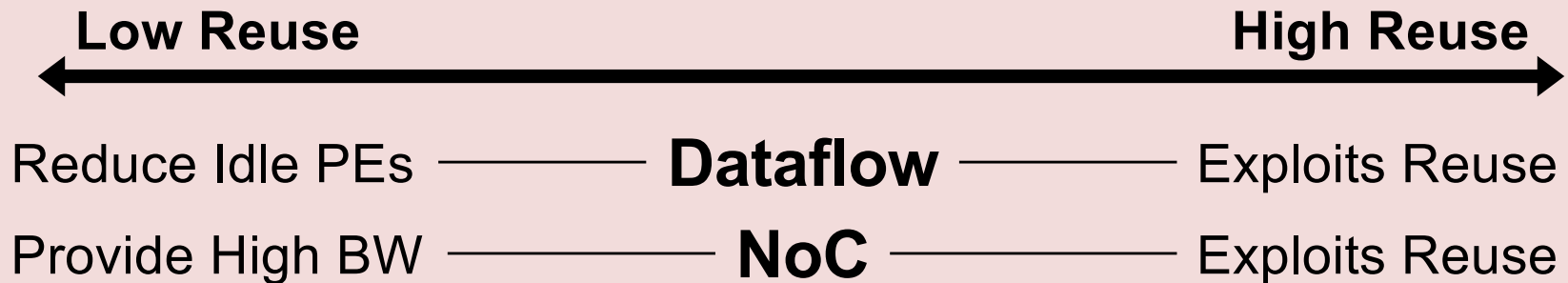


- Supports a wide range of data sparsity
 - exploits sparsity in **both feature maps and weights** to achieve higher throughput and energy efficiency

[Chen, submitted to *Micro* 2018]

Eyeriss v2: Design Goals

- Supports a wide range of data reuse



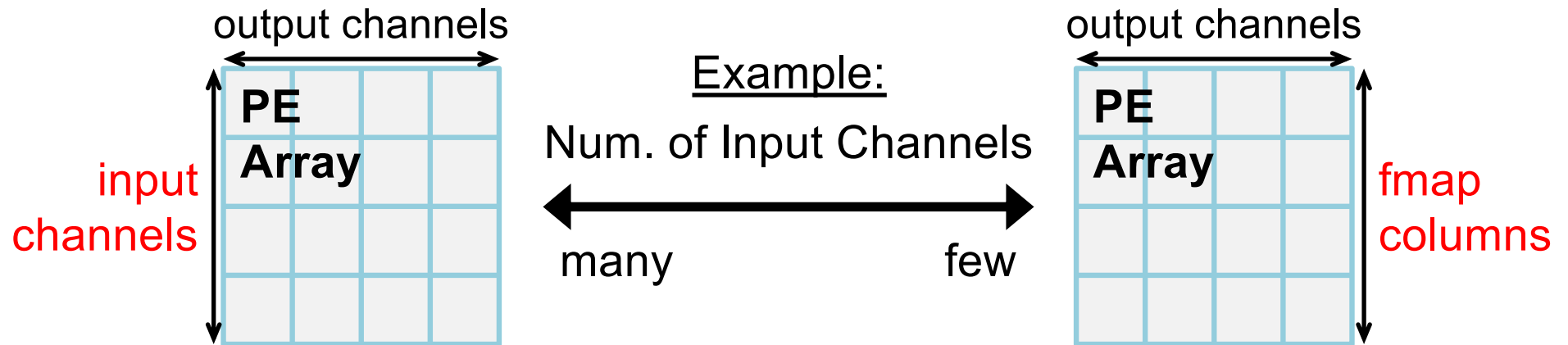
Today's Focus

- Supports a wide range of data sparsity
 - exploits sparsity in both feature maps and weights to achieve higher throughput and energy efficiency

[Chen, submitted to *Micro* 2018]

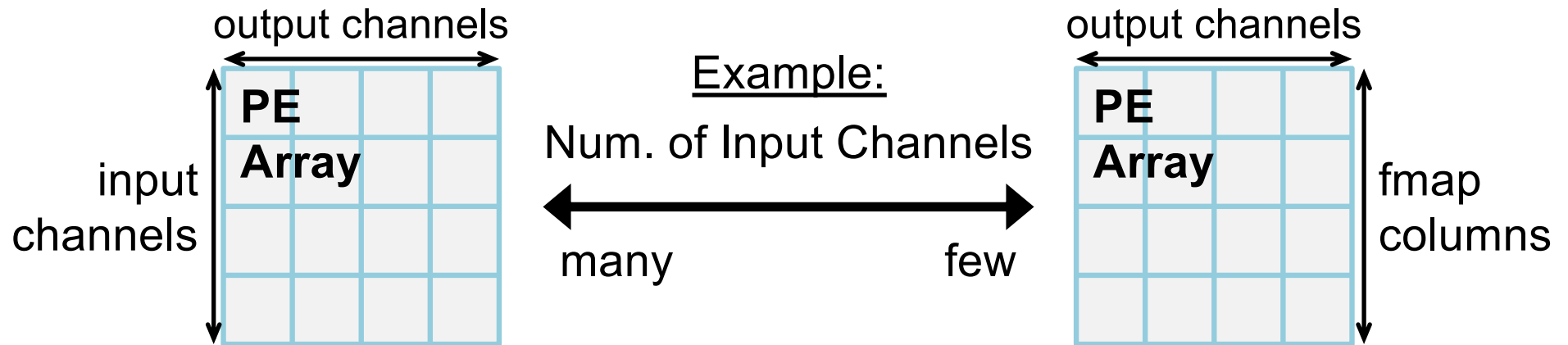
Row-Stationary Plus (RS+) Dataflow

- Any data dimension can be mapped spatially

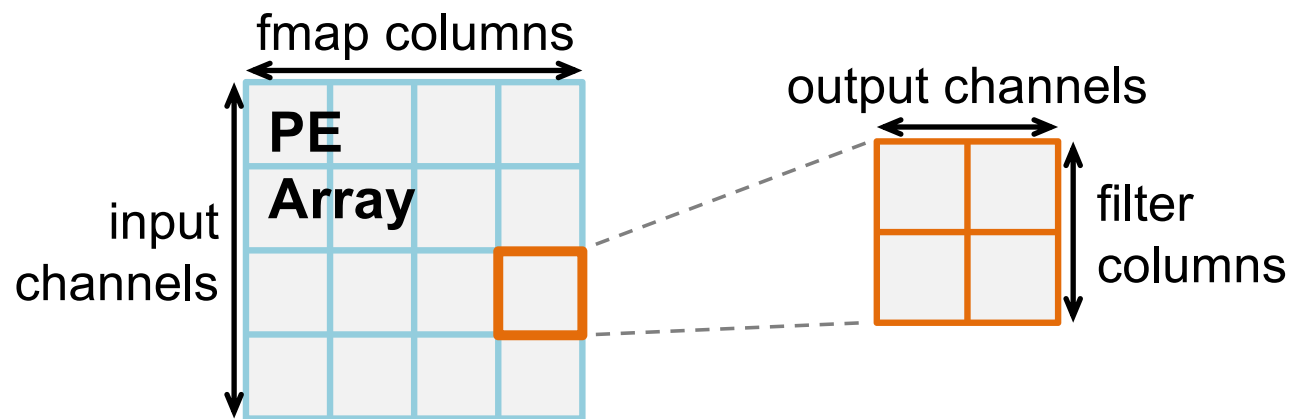


Row-Stationary Plus (RS+) Dataflow

- Any data dimension can be mapped spatially



- Mult. data dimensions on the same PE array dimension



Row-Stationary Plus (RS+) Dataflow

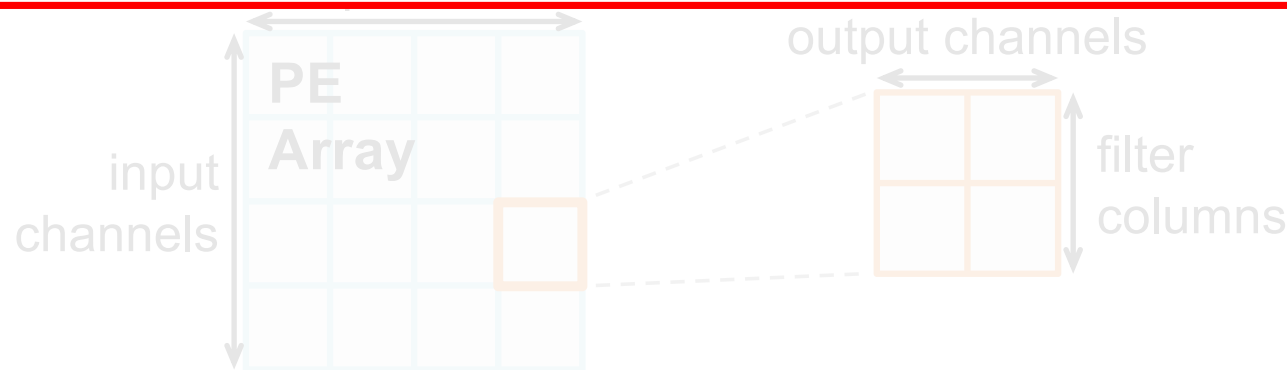
- Any data dimension can be mapped spatially



Example:

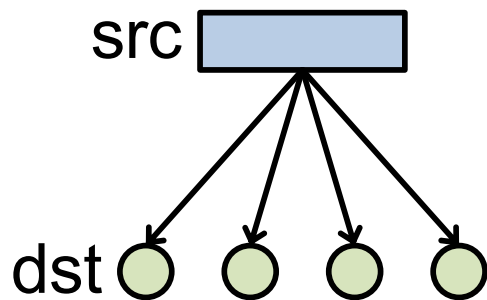


- Minimize the number of idle PEs** with very flexible spatial mappings
- RS+ is a superset of RS → **same or better performance**



On-Chip Network (NoC) is the Bottleneck

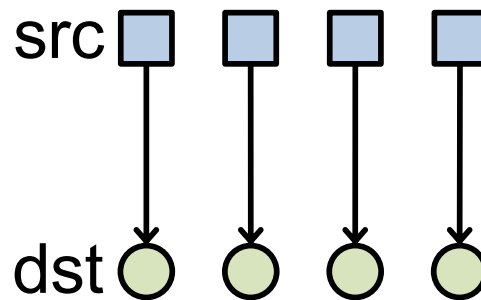
Broadcast Network (Eyeriss v1 NoC)



High Reuse

Low Bandwidth

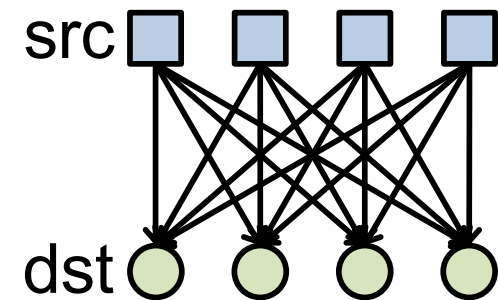
Unicast Networks



Low Reuse

High Bandwidth

All-to-All Networks

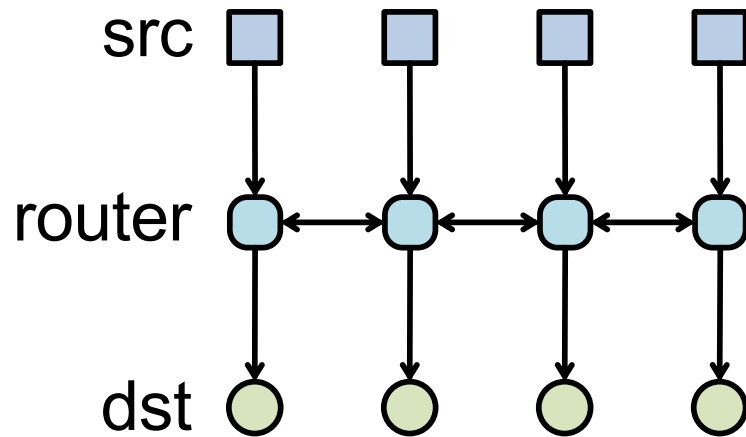


High Reuse

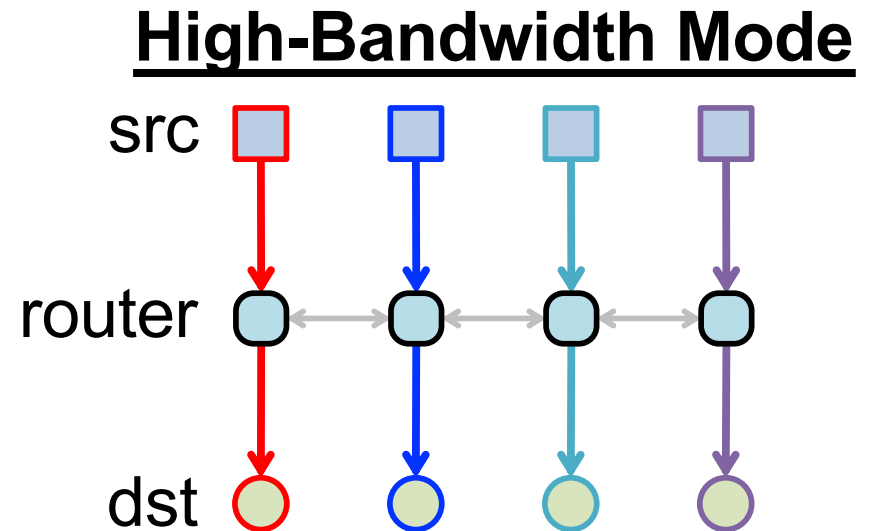
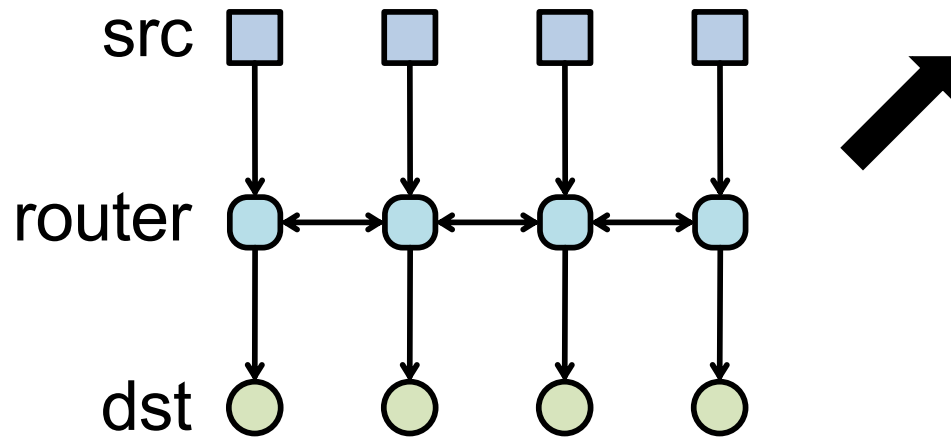
High Bandwidth

Hard to Scale

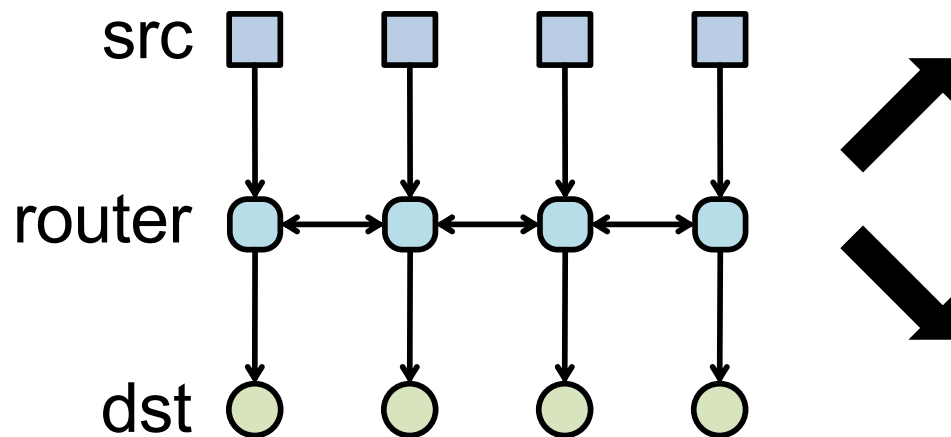
Mesh Network – Best of Both Worlds



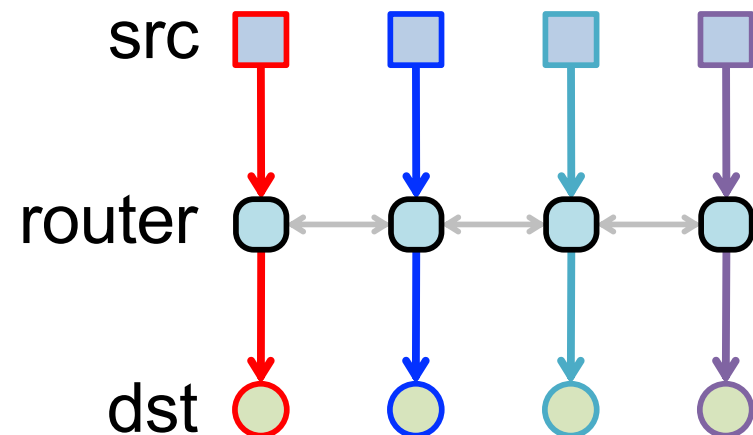
Mesh Network – Best of Both Worlds



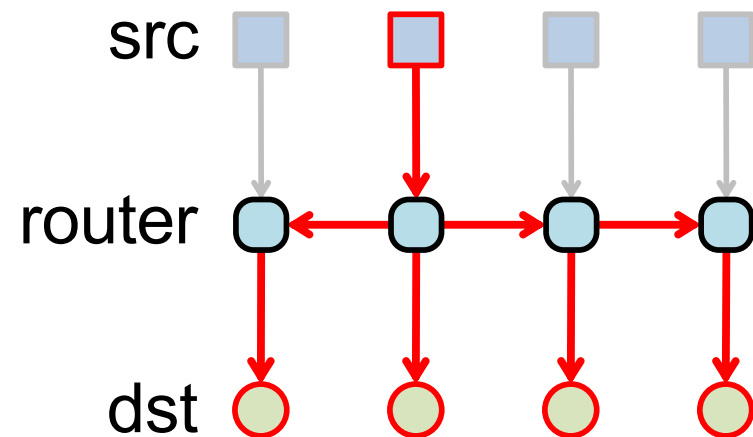
Mesh Network – Best of Both Worlds



High-Bandwidth Mode

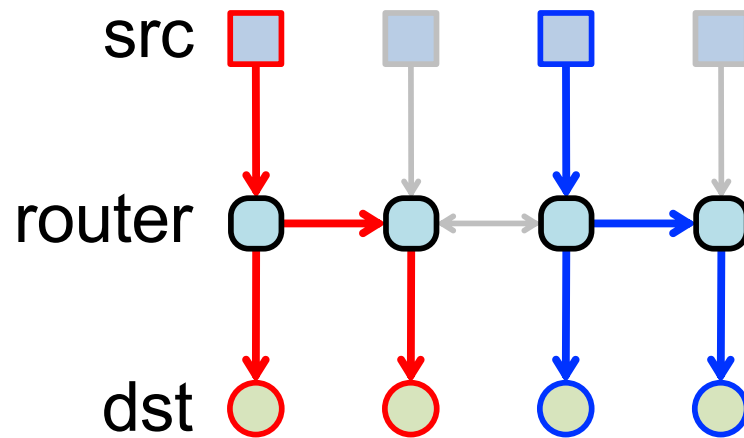


High-Reuse Mode



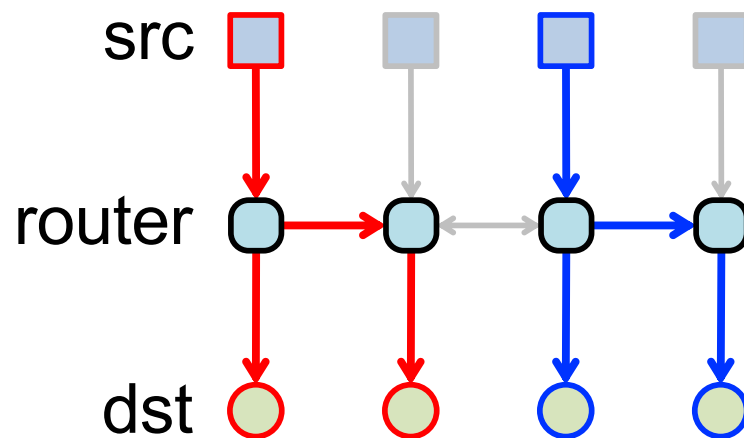
Mesh Network – More Complicated Cases

Grouped-Multicast Mode

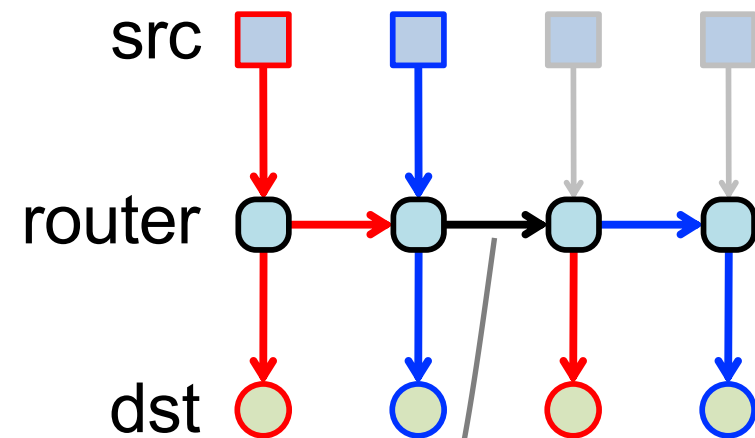


Mesh Network – More Complicated Cases

Grouped-Multicast Mode



Interleaved-Multicast Mode

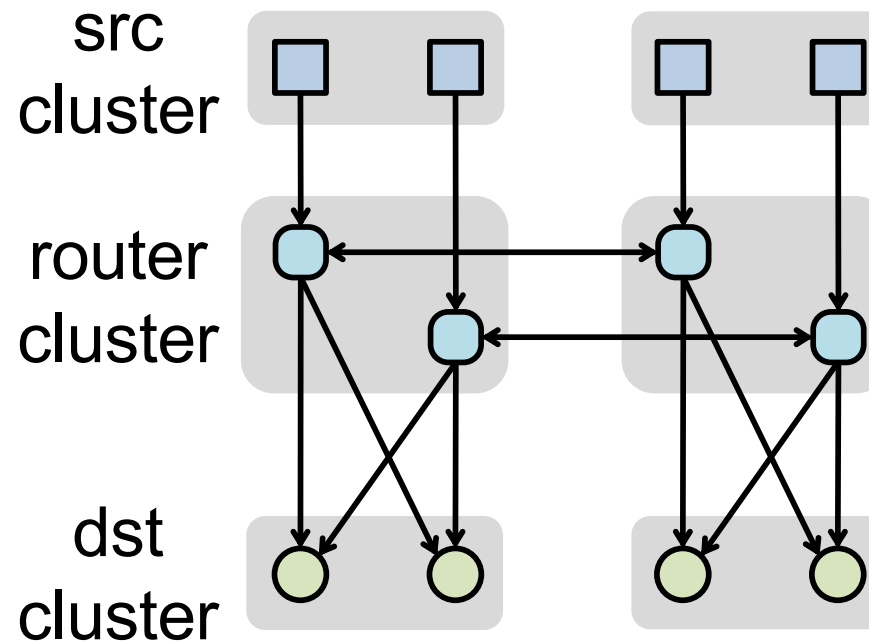


Bandwidth-limited route
(flow control required)

Hierarchical Mesh Network

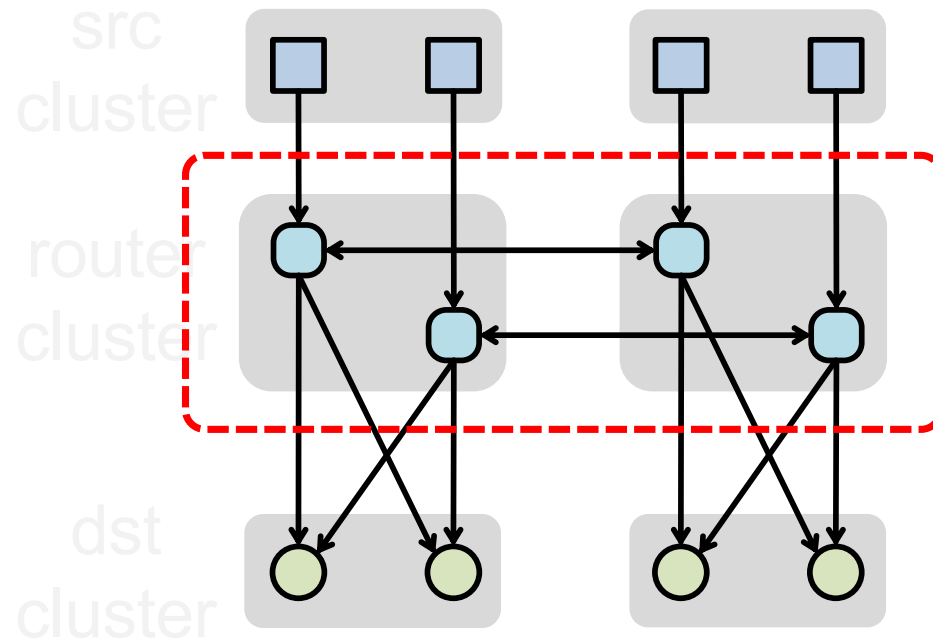
- Flexible to support patterns ranging from high reuse to high bandwidth scenarios
- Can be easily scaled at a low cost

Hierarchical Mesh Network

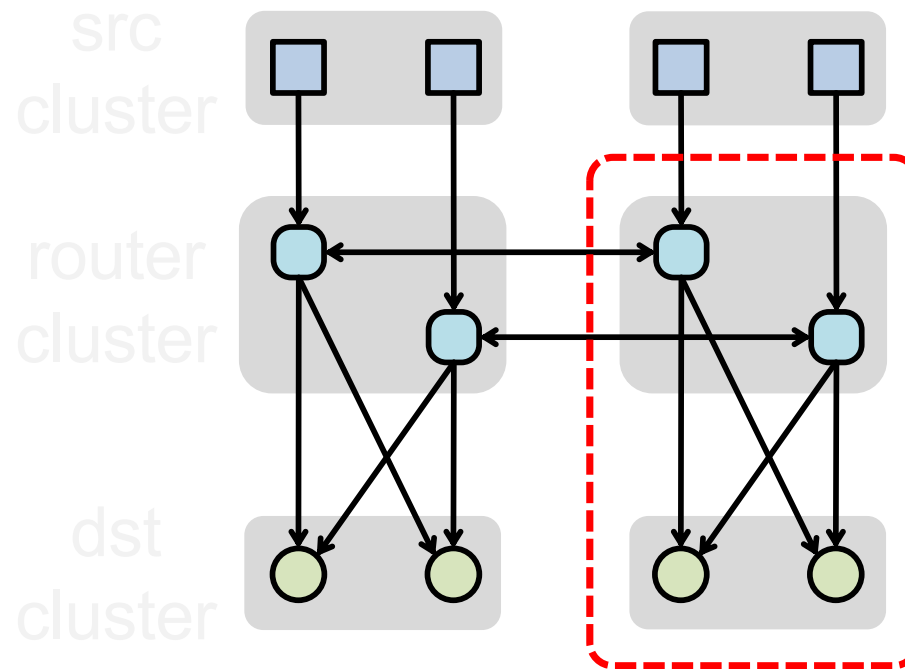


Hierarchical Mesh Network

Mesh Network for inter-cluster connections



Hierarchical Mesh Network

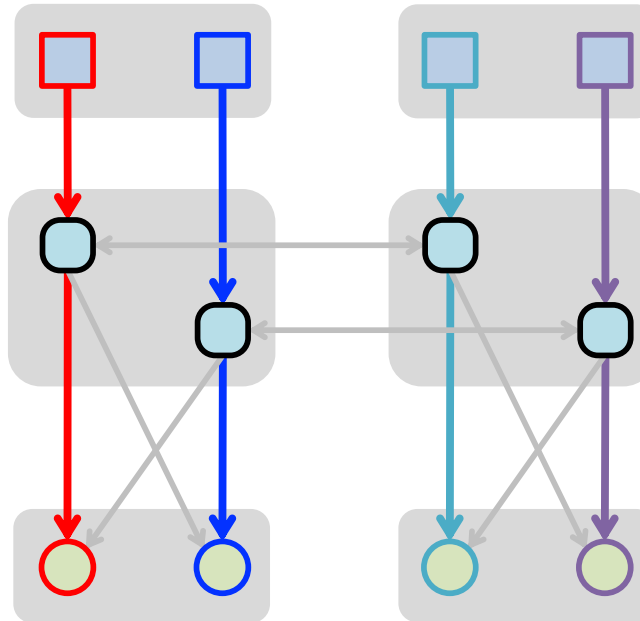


All-to-All Network for **intra-cluster** connections

Complexity is contained within a cluster

Hierarchical Mesh Network

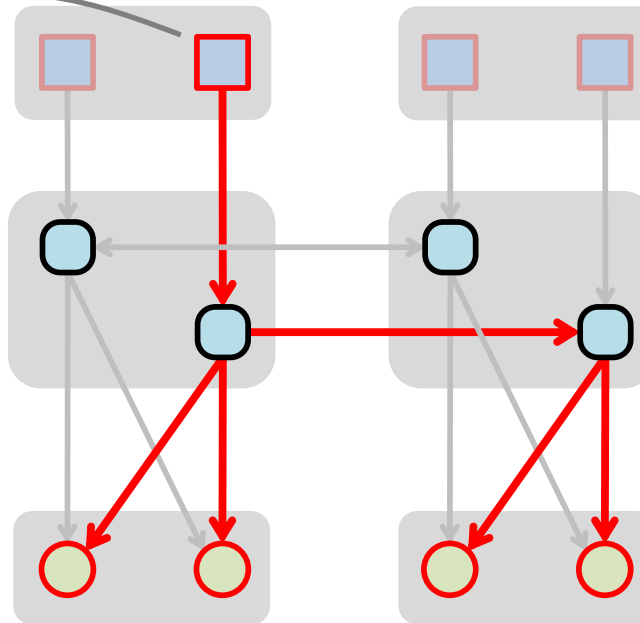
High-Bandwidth Mode



Hierarchical Mesh Network

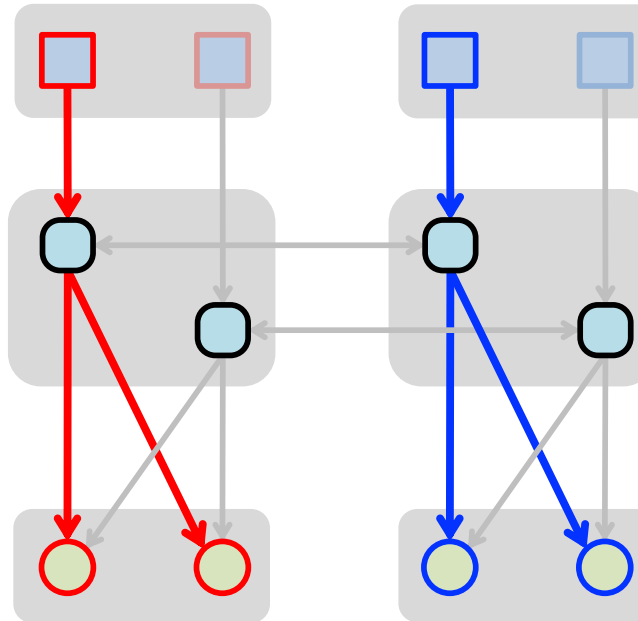
High-Reuse Mode

from any one src



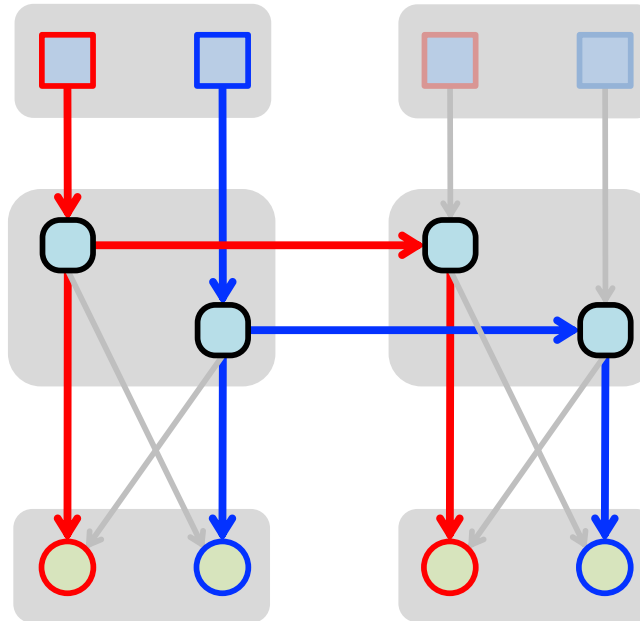
Hierarchical Mesh Network

Grouped-Multicast Mode



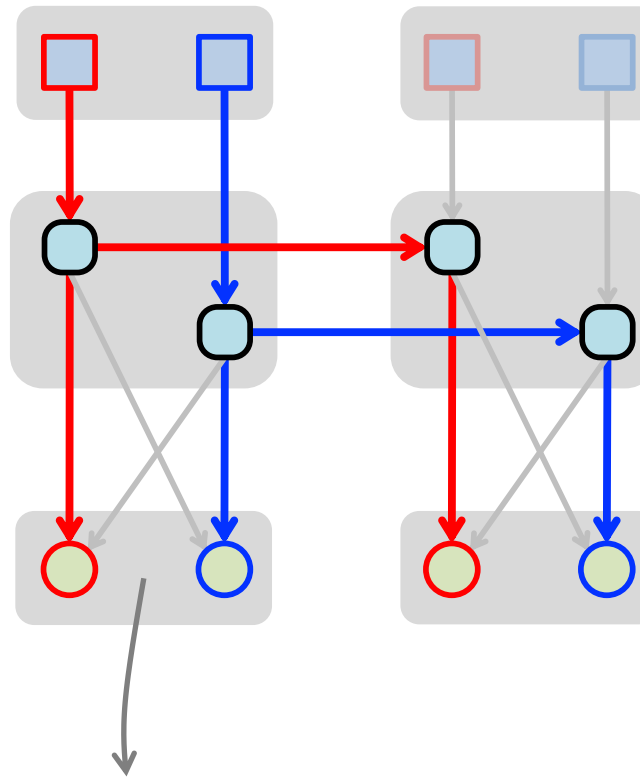
Hierarchical Mesh Network

Interleaved-Multicast Mode



Hierarchical Mesh Network

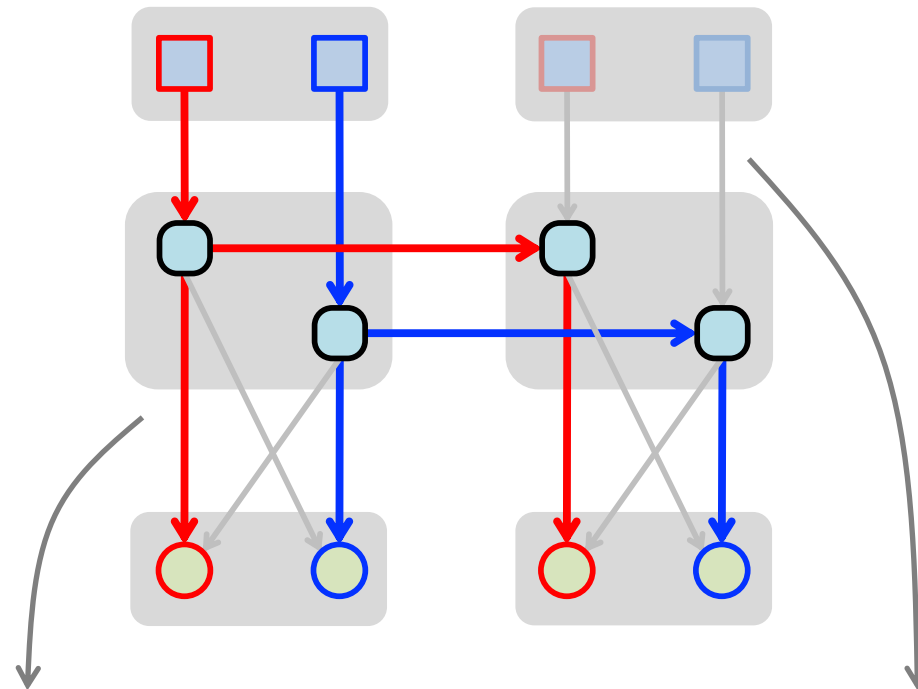
Interleaved-Multicast Mode



Can interleave more by scaling up the cluster size

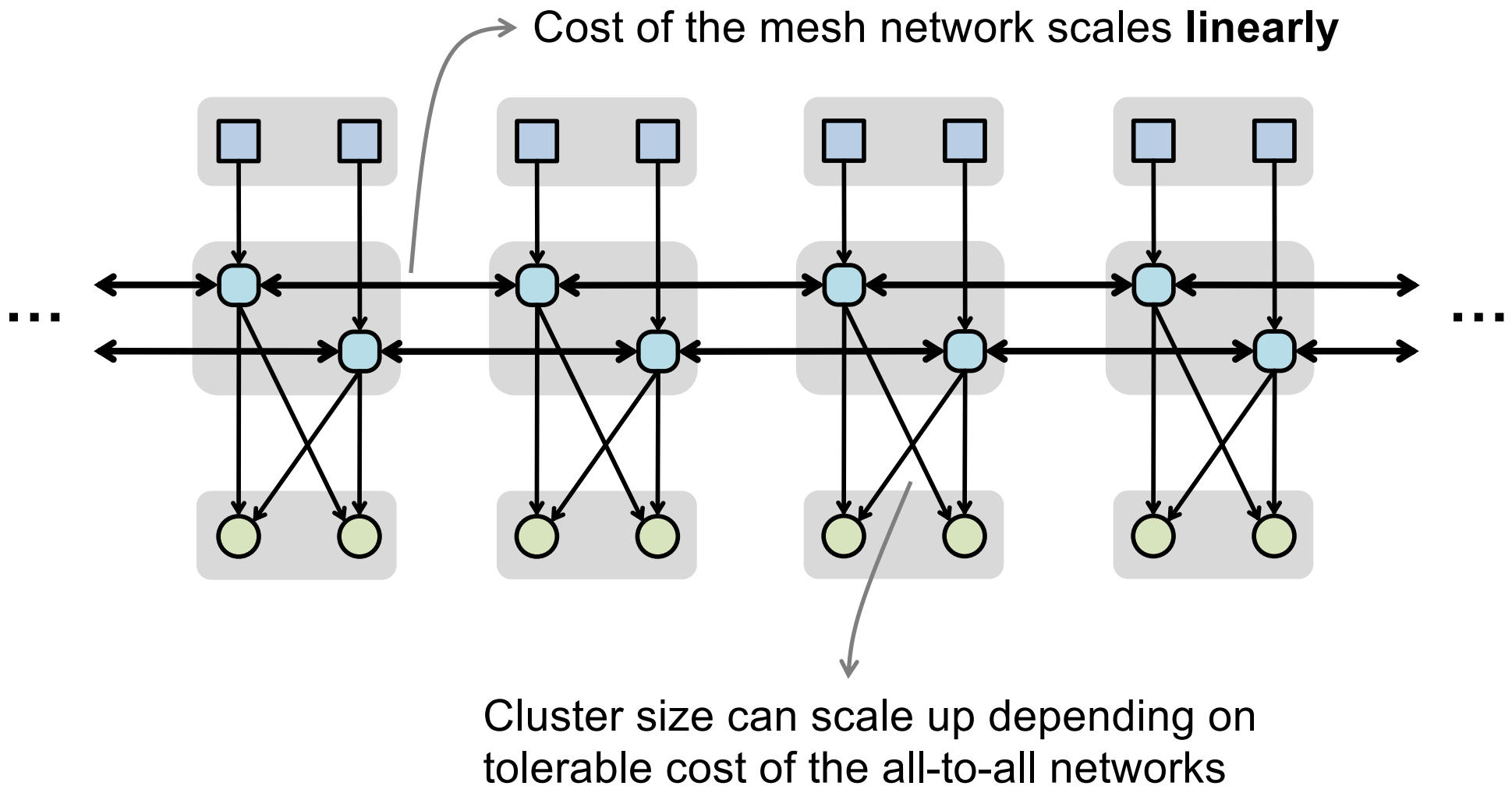
Hierarchical Mesh Network

Interleaved-Multicast Mode

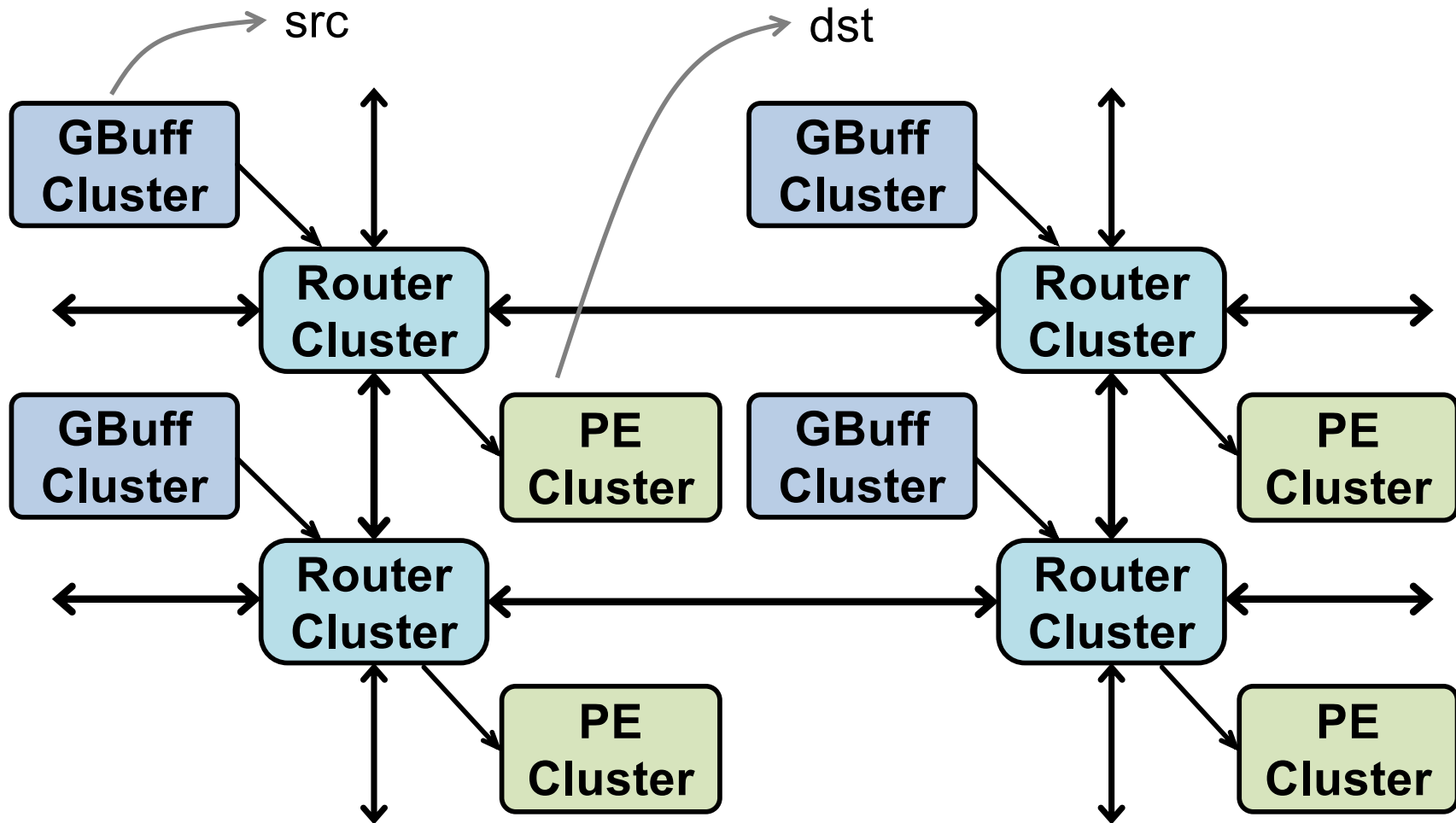


All routes are determined **at configuration time**
→ **Routers** are **circuit-switched** (only **MUXes**)

Scaling the Hierarchical Mesh Network

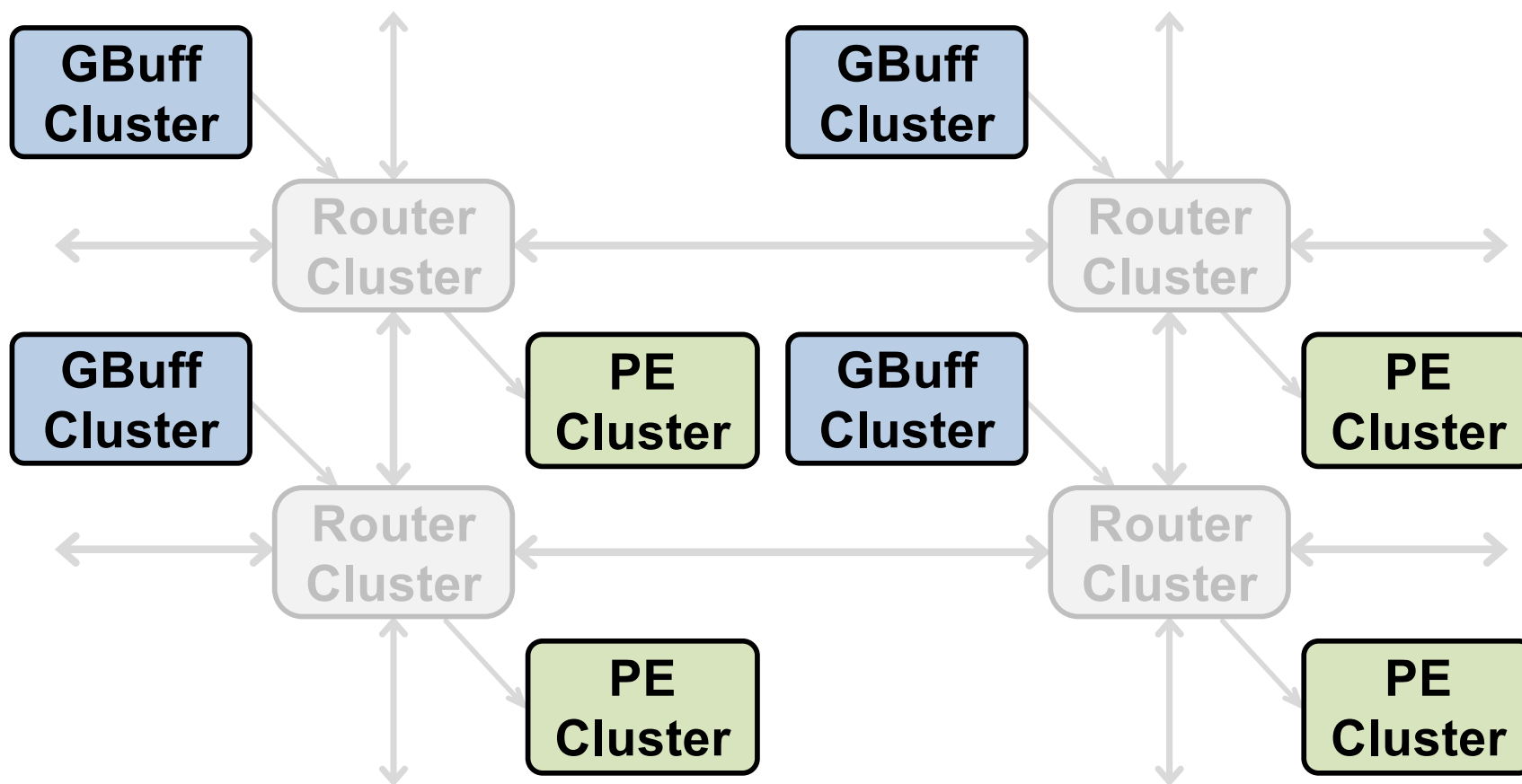


Eyeriss v2 Architecture



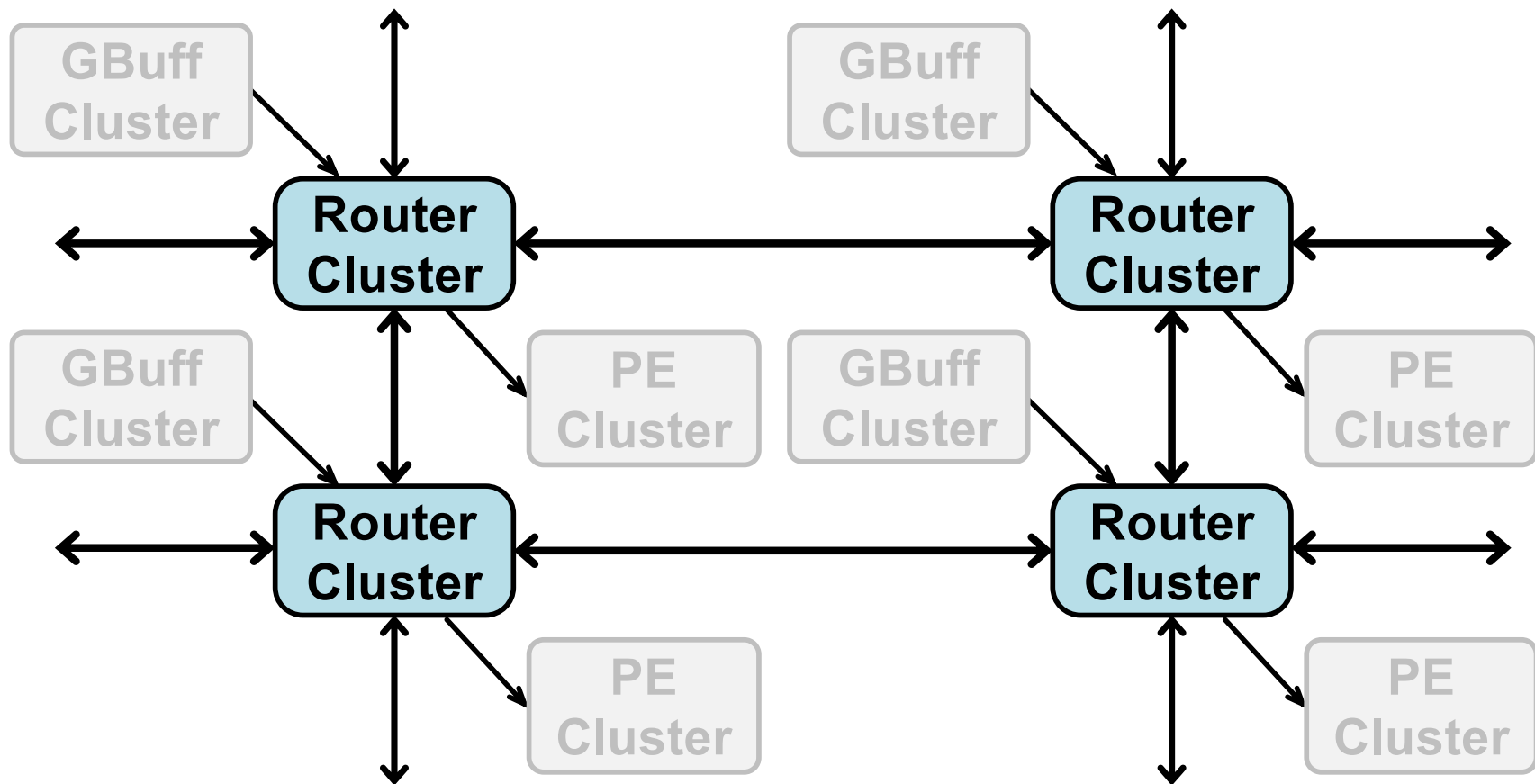
Eyeriss v2 Architecture

11.25 kB



4x4 PE Array

Eyeriss v2 Architecture

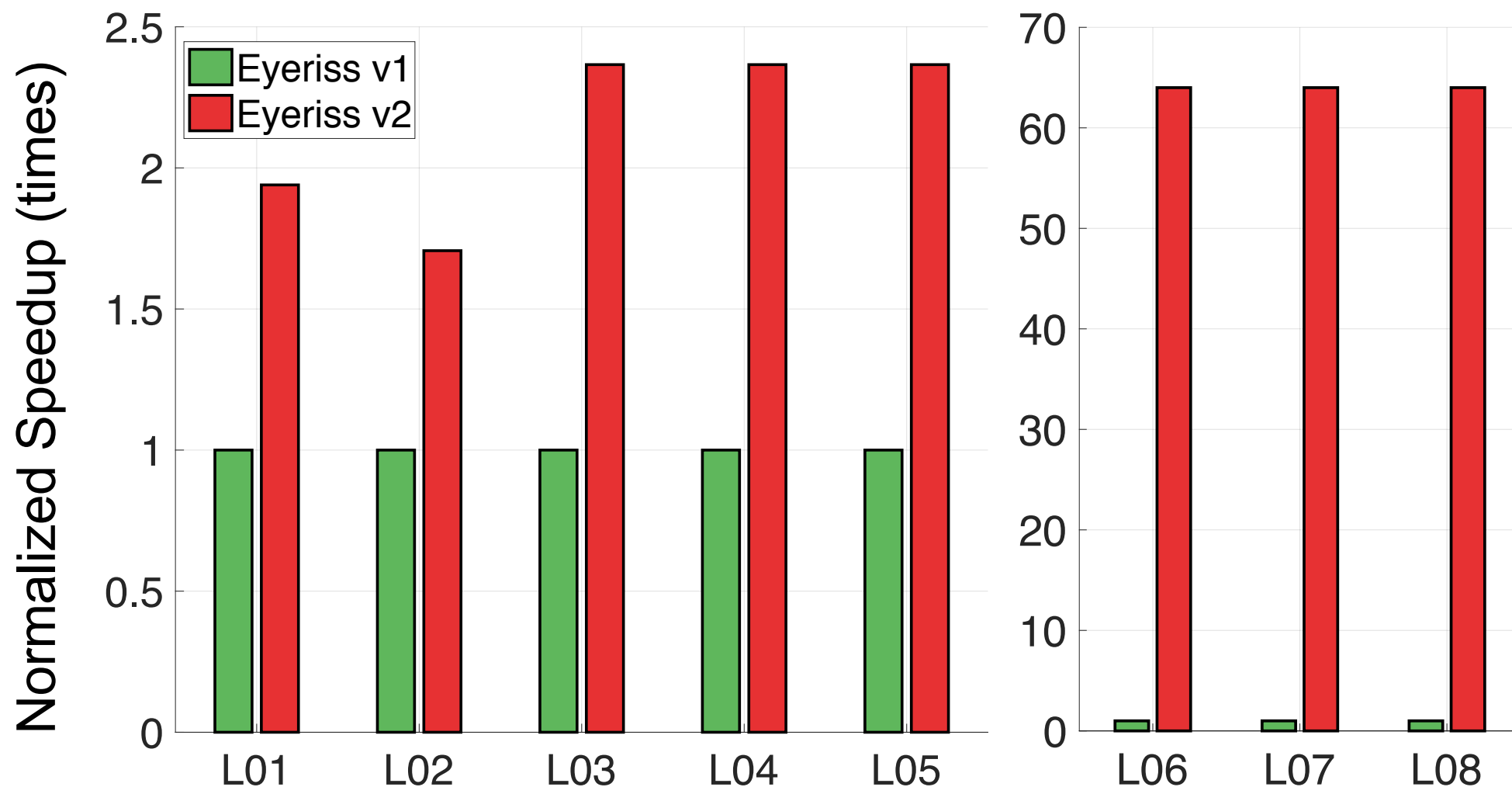


The NoC consumes around **3% – 5% of total Energy**

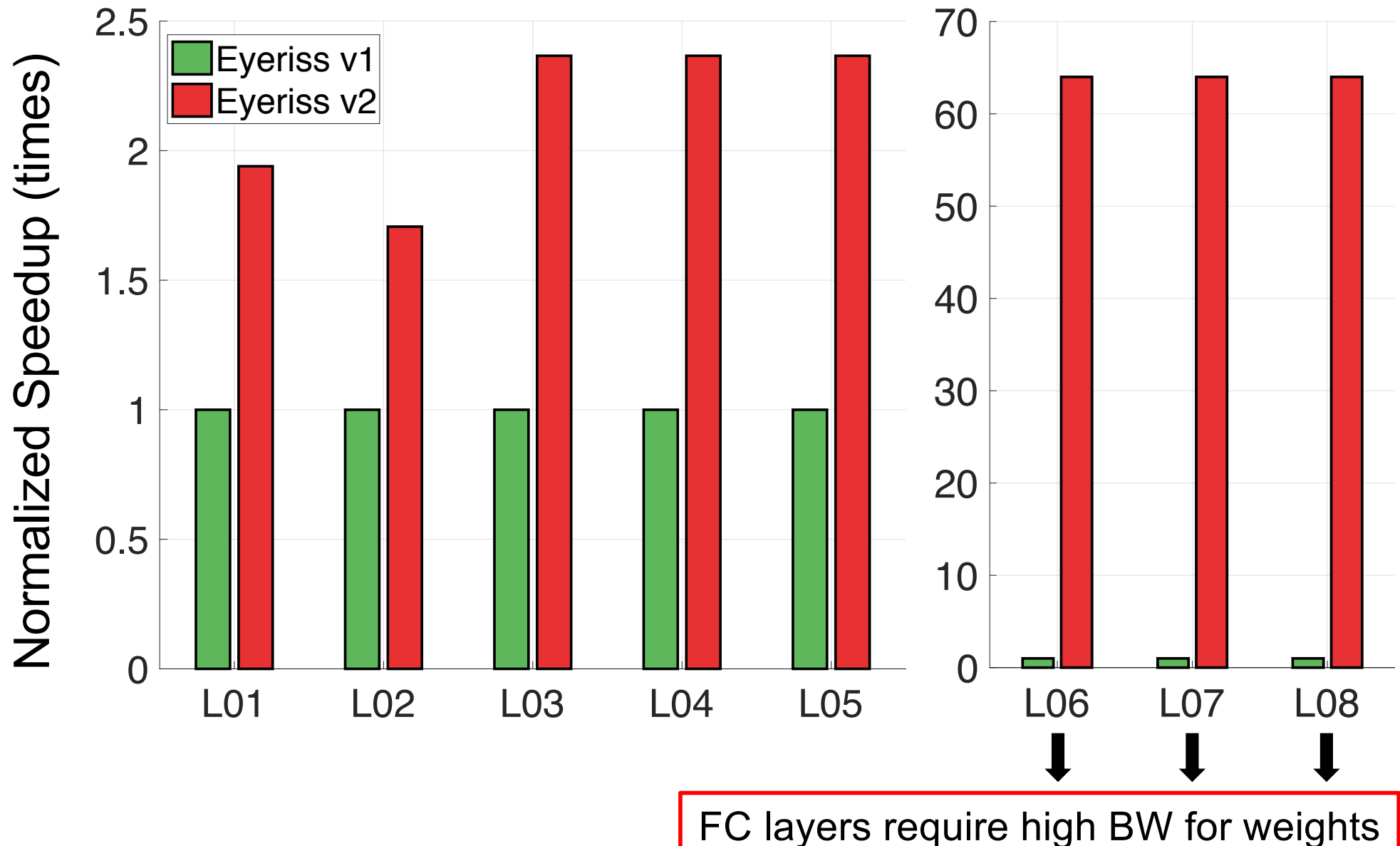
Simulation Results

- Throughput Comparison: **Eyeriss v1 vs. v2**
- Both have **256 PEs**
 - Eyeriss v1: 16×16 PE Array
 - Eyeriss v2: 16 Clusters, each with 16 PEs
- **Batch Size = 1**

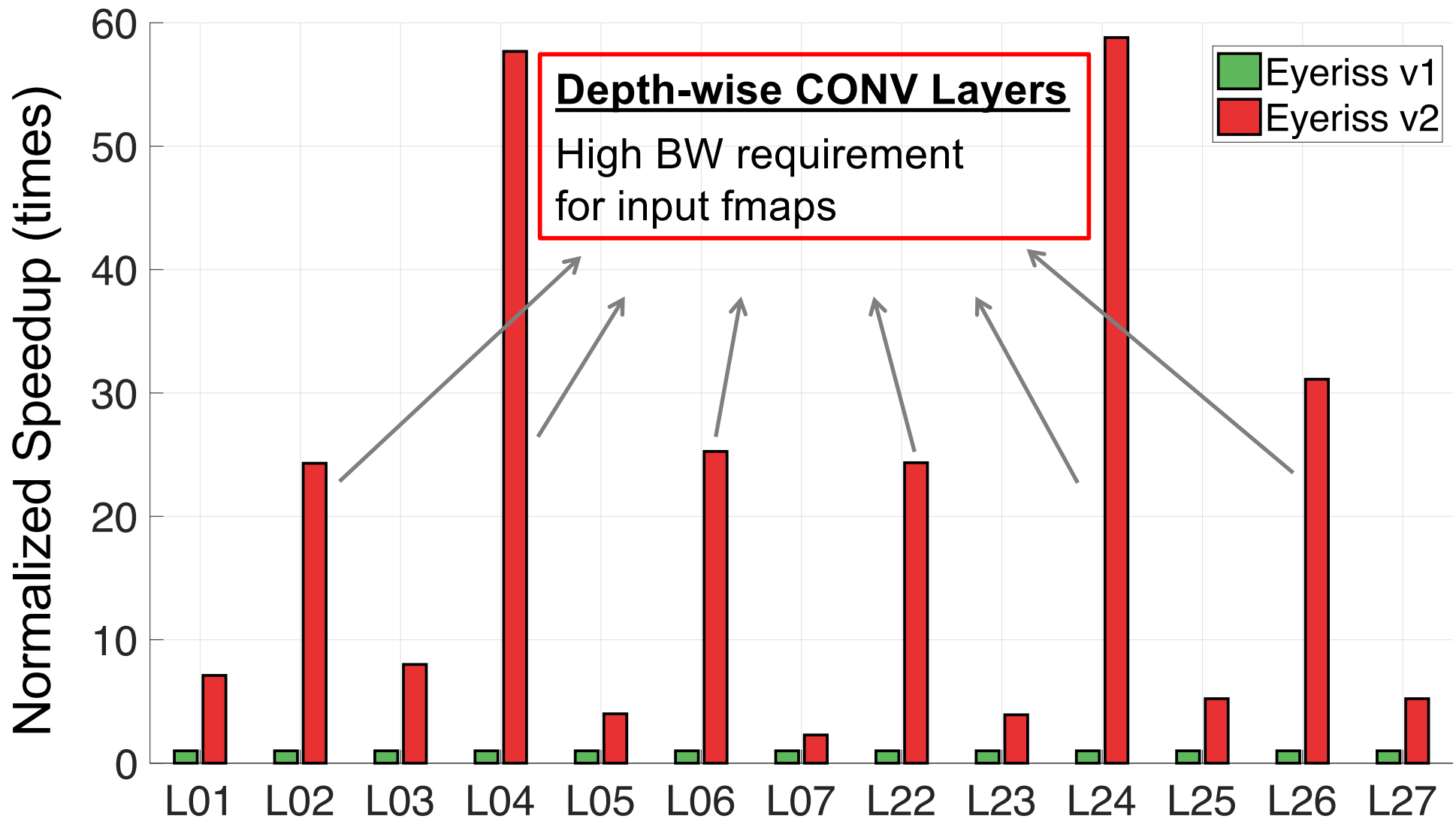
Throughput Comparison: AlexNet



Throughput Comparison: AlexNet



Throughput Comparison: MobileNet



Throughput Comparison: Summary

Throughput Speedup of Eyeriss v2 over Eyeriss v1

	256 PEs
AlexNet	17.9×
GoogLeNet	10.4×
MobileNet	15.7×
Overall	13.3×

Throughput Comparison: Summary

Throughput Speedup of Eyeriss v2 over Eyeriss v1

	256 PEs	1024 PEs	16384 PEs
AlexNet	17.9×	71.5×	1086.7×
GoogLeNet	10.4×	37.8×	448.8×
MobileNet	15.7×	57.9×	873.0×
Overall	13.3×	50.3×	693.3×

Eyeriss v2 has over **10×** throughput speedup than Eyeriss v1

Eyeriss v2: Summary of Contributions

Flexibility**Performance****Energy Efficiency**

Supports a Wide Range of Data Reuse

- **Minimizes number of idle PEs** with the RS+ dataflow (**F** **P**)
- Provides sufficient **bandwidth** and exploits **data reuse** with the hierarchical mesh NoC (**F** **E** **P**)

Eyeriss v2: Summary of Contributions

Flexibility

Performance

Energy Efficiency

Supports a Wide Range of Data Reuse

- Minimizes number of idle PEs with the RS+ dataflow (F P)
- **Over 10× Speedup in Throughput than Eyeriss v1**

Eyeriss v2: Summary of Contributions

Flexibility**Performance****Energy Efficiency**

Supports a Wide Range of Data Reuse

- Minimizes number of idle PEs with the RS+ dataflow (F P)
- **Over 10× Speedup in Throughput than Eyeriss v1**

Supports a Wide Range of Data Sparsity

- Processes **both feature maps and weights** in compressed format to improve throughput and energy efficiency (P E)
- Processes in raw format when sparsity is low (F)

Eyeriss v2: Summary of Contributions

Flexibility**Performance****Energy Efficiency**

Supports a Wide Range of Data Reuse

- Minimizes number of idle PEs with the RS+ dataflow (F P)
- **Over 10× Speedup in Throughput than Eyeriss v1**

Supports a Wide Range of Data Sparsity

- Processes both feature maps and weights in compressed format to improve energy efficiency (F P)
- **An additional 3.0× Speedup in Throughput**
- Processes in raw format when sparsity is low (F)

Conclusion

- **Data reuse** is the key to achieving **high energy efficiency**.
- High PE utilization with **adaptive on-chip networks** is the key to achieving **high performance**.
- **Flexible hardware** is required to deal with the **diverse set of DNNs**, which is constantly evolving.
- Co-design of **dataflow** and **hardware** is critical for the optimization of **performance**, **energy efficiency** and **flexibility** for DNN accelerators.
- Achieving this balance will open up more opportunities for AI to be applied in real-world applications.