

Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks

Yu-Hsin Chen¹, Joel Emer^{1,2}, Vivienne Sze¹

¹ MIT ² NVIDIA



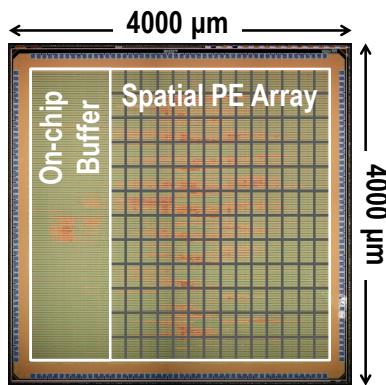
Massachusetts
Institute of
Technology



nVIDIA®

Contributions of This Work

- A novel **energy-efficient CNN dataflow** that has been verified in a fabricated chip, *Eyeriss*.
- A **taxonomy of CNN dataflows** that classifies previous work into three categories.
- A **framework** that compares the energy efficiency of different dataflows under same area and CNN setup.



Eyeriss [ISSCC, 2016]

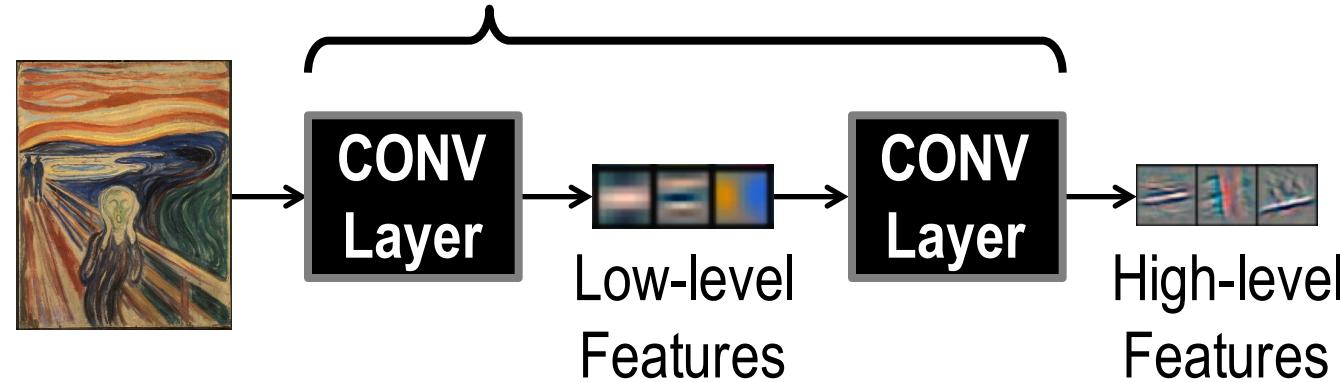
A reconfigurable CNN
processor

35 fps @ 278 mW*

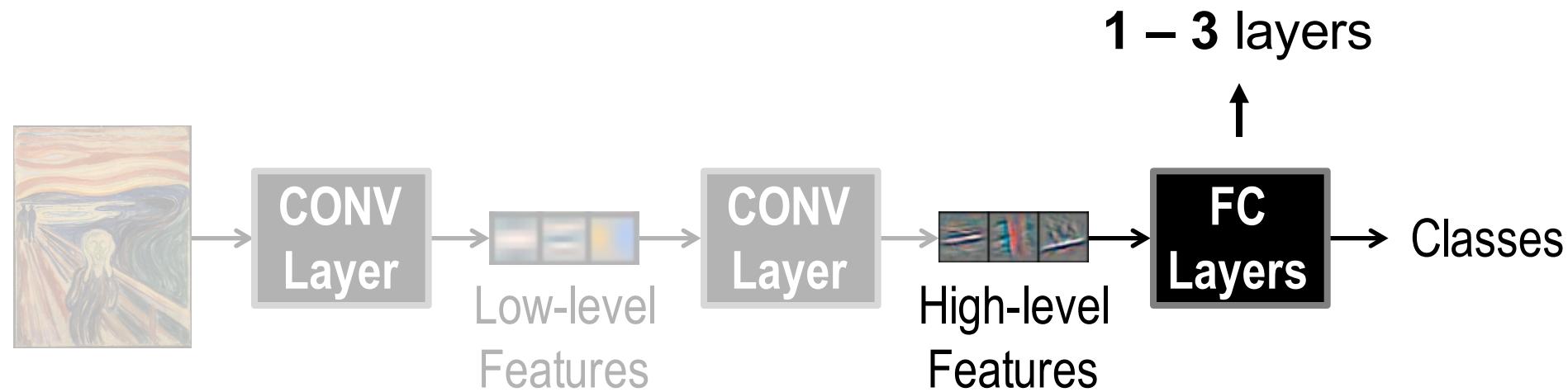
* AlexNet CONV layers

Deep Convolutional Neural Networks

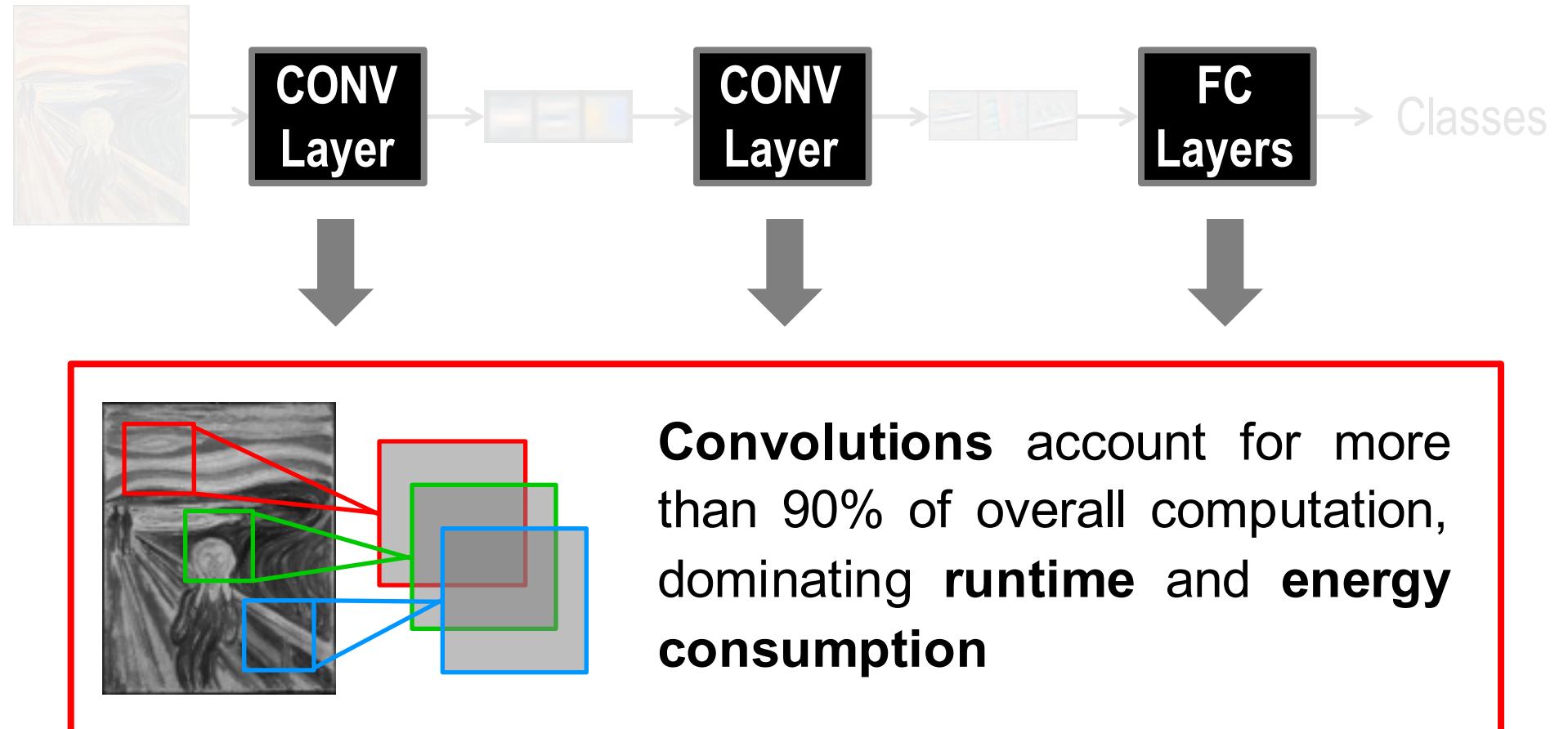
Modern *deep* CNN: up to 1000 CONV layers



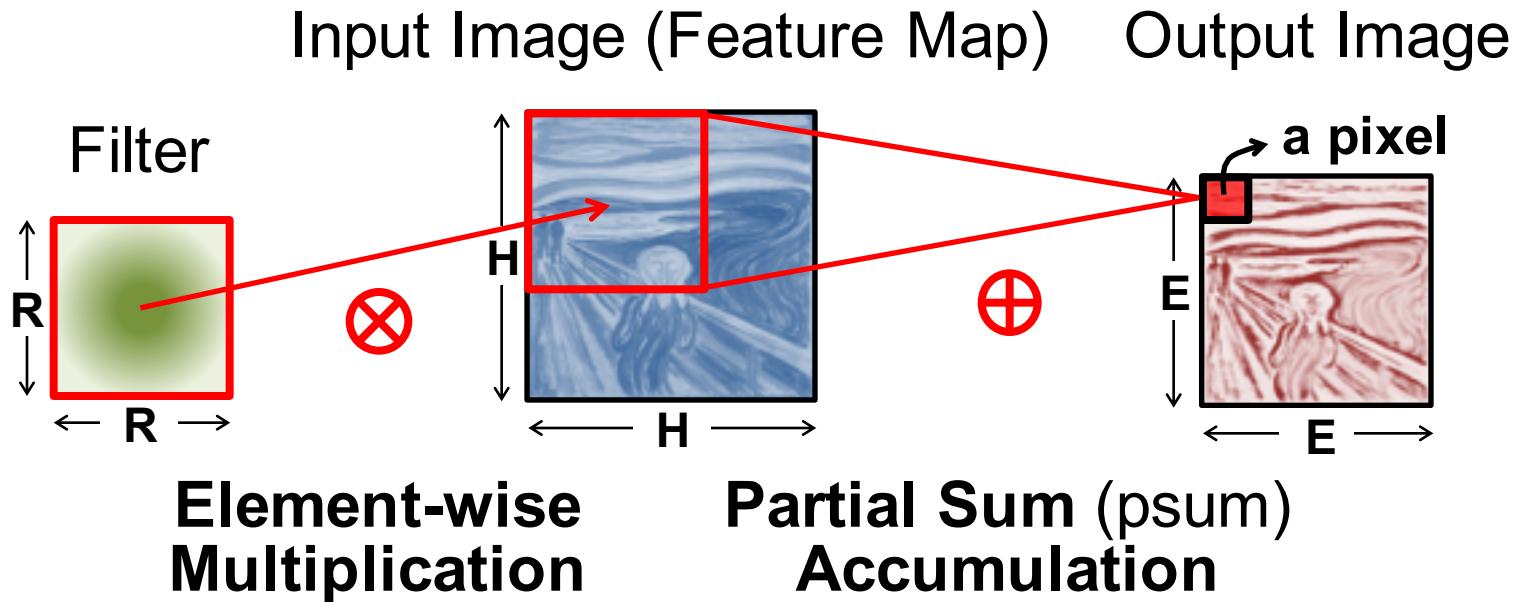
Deep Convolutional Neural Networks



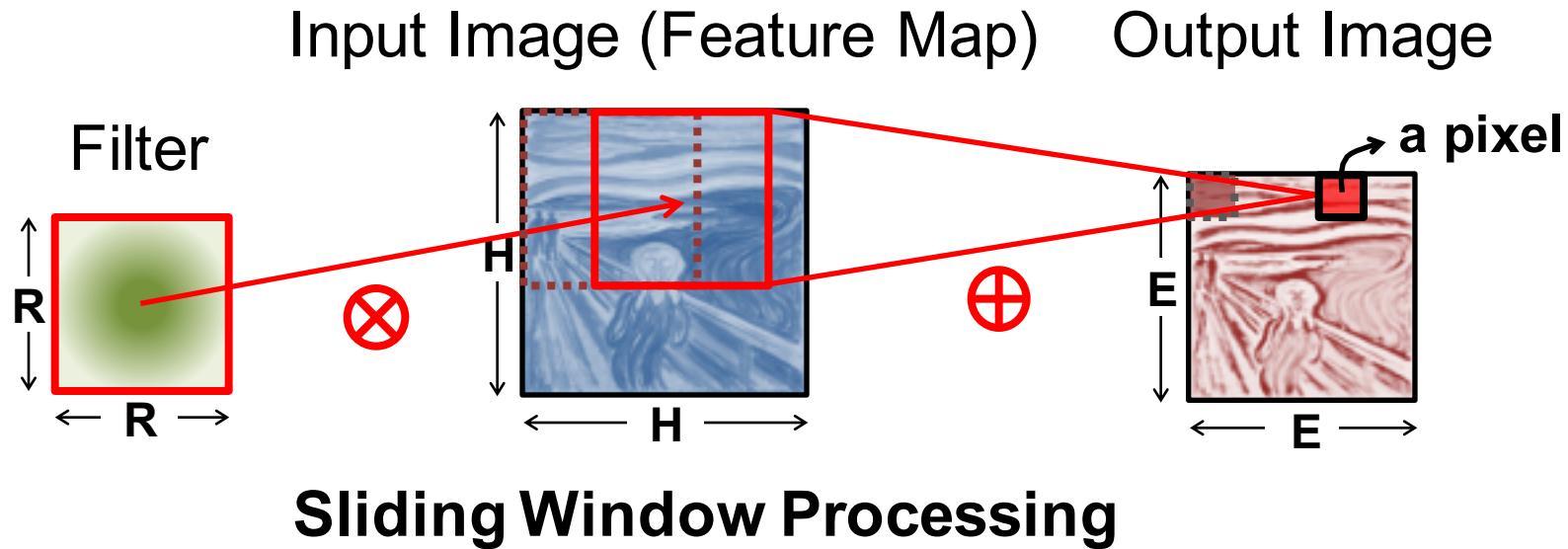
Deep Convolutional Neural Networks



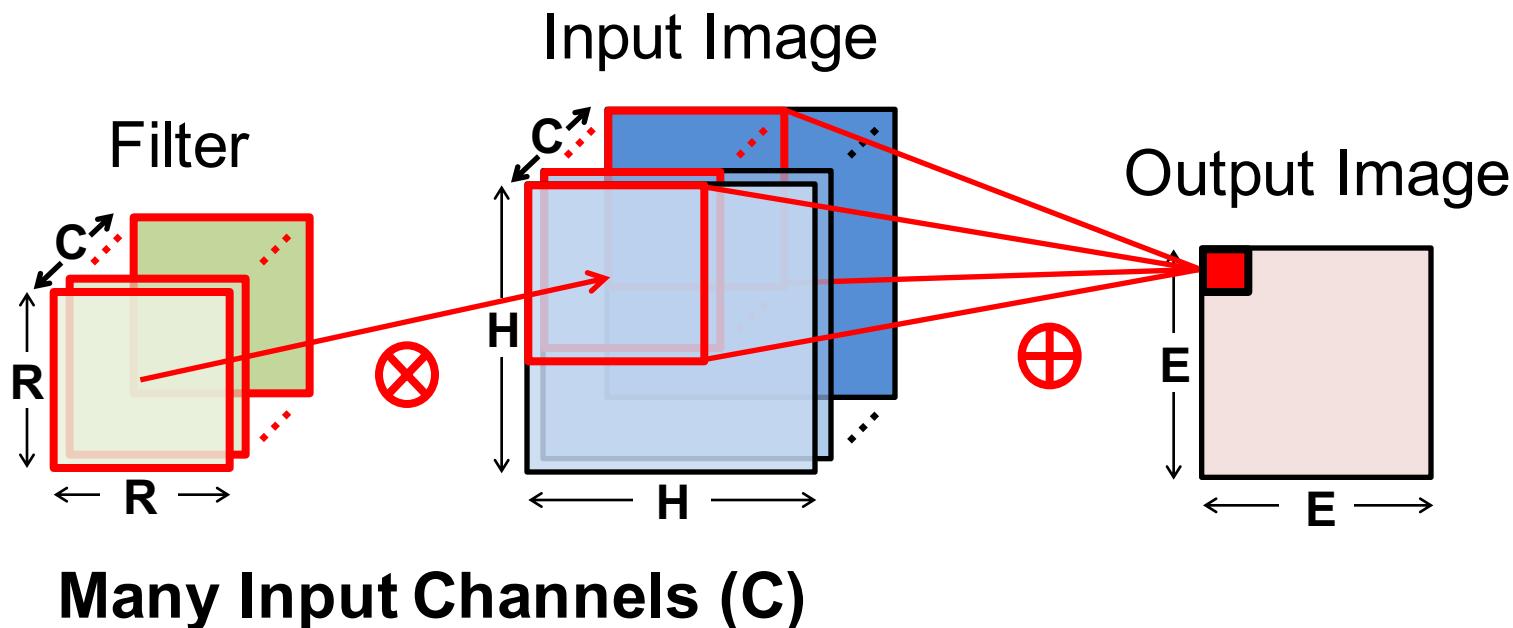
High-Dimensional CNN Convolution



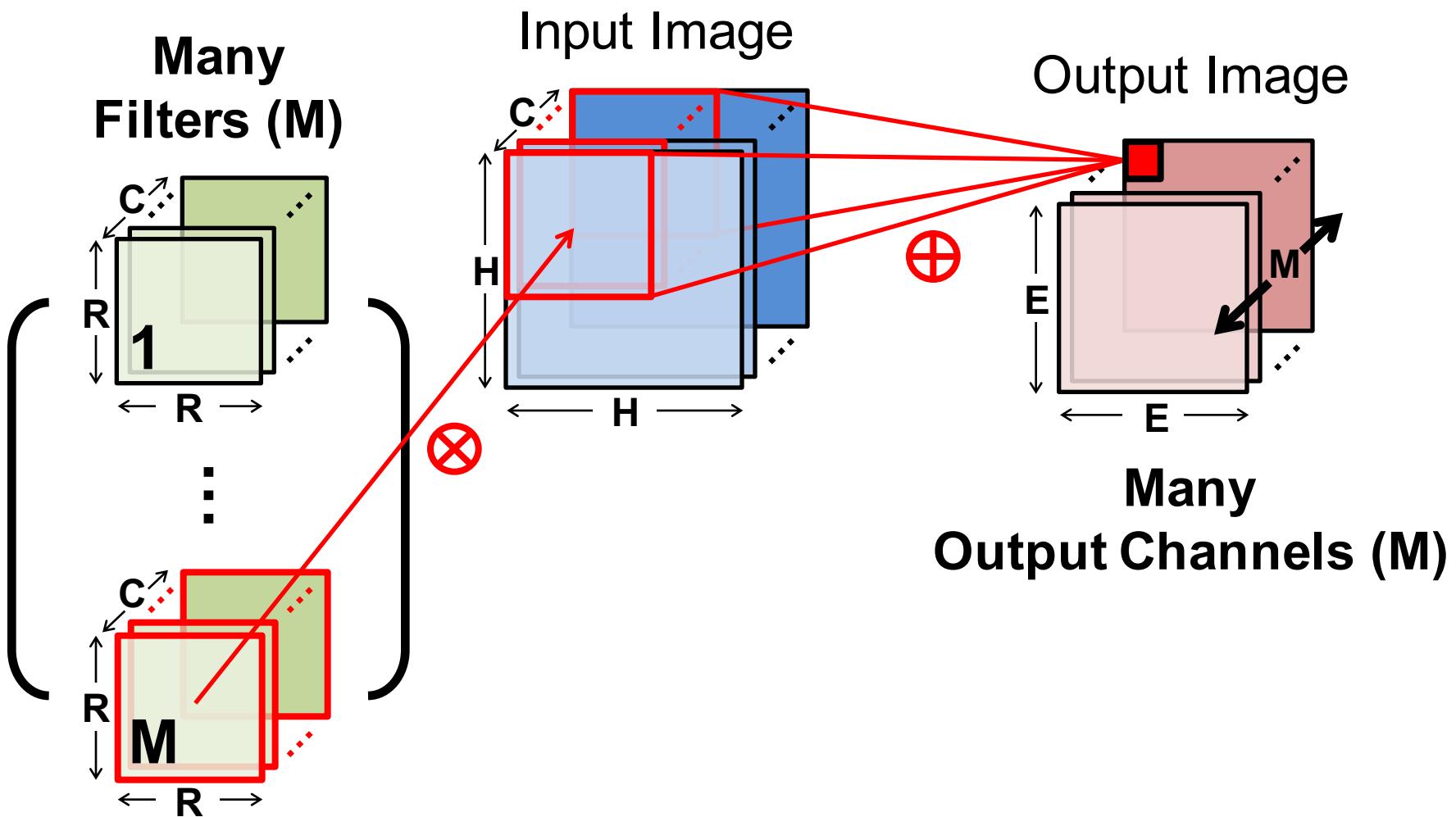
High-Dimensional CNN Convolution



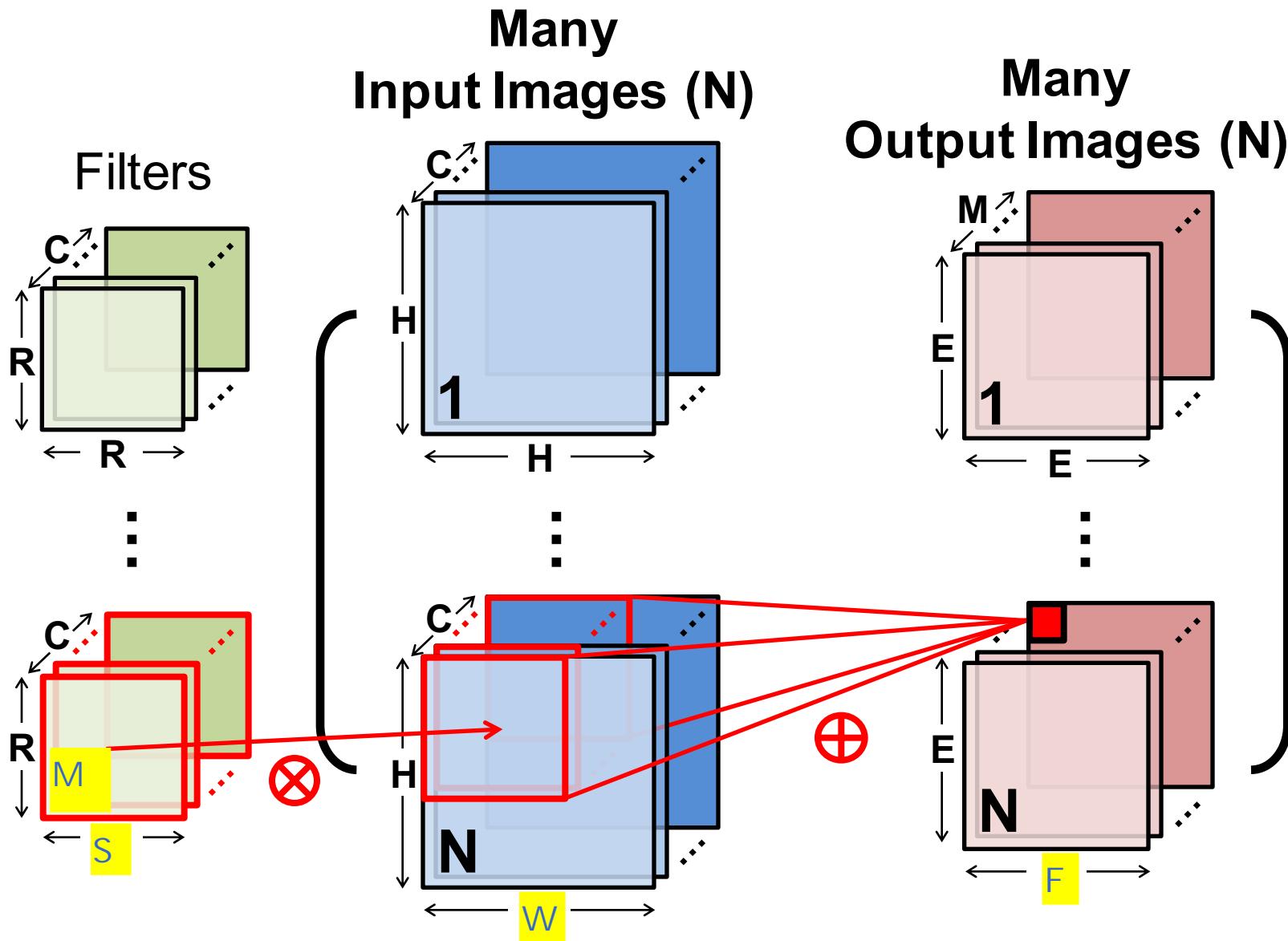
High-Dimensional CNN Convolution



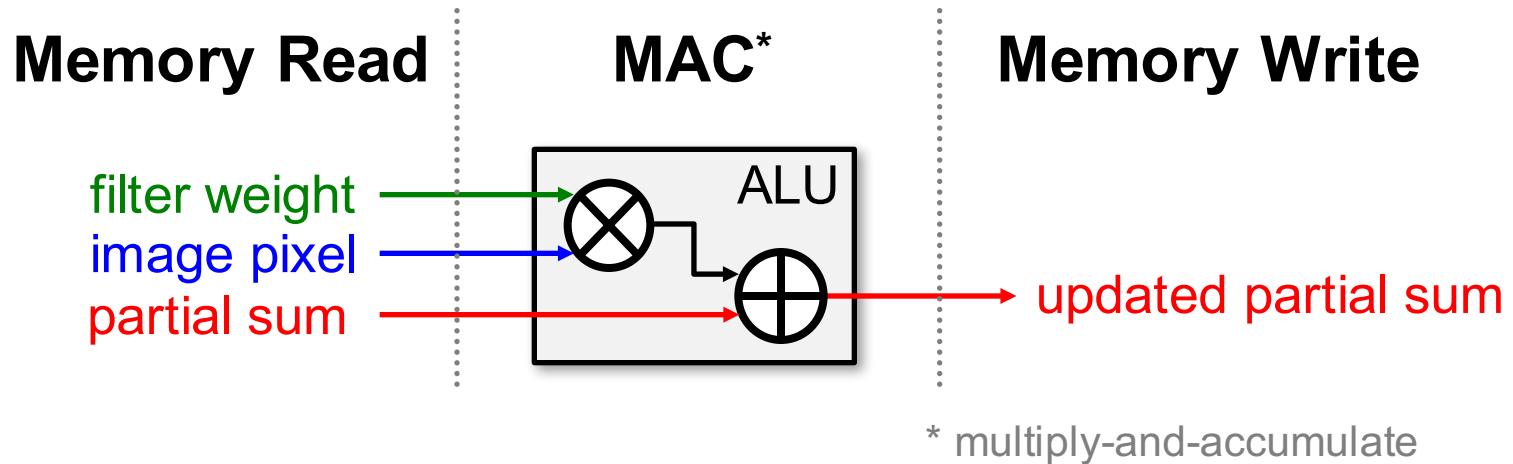
High-Dimensional CNN Convolution



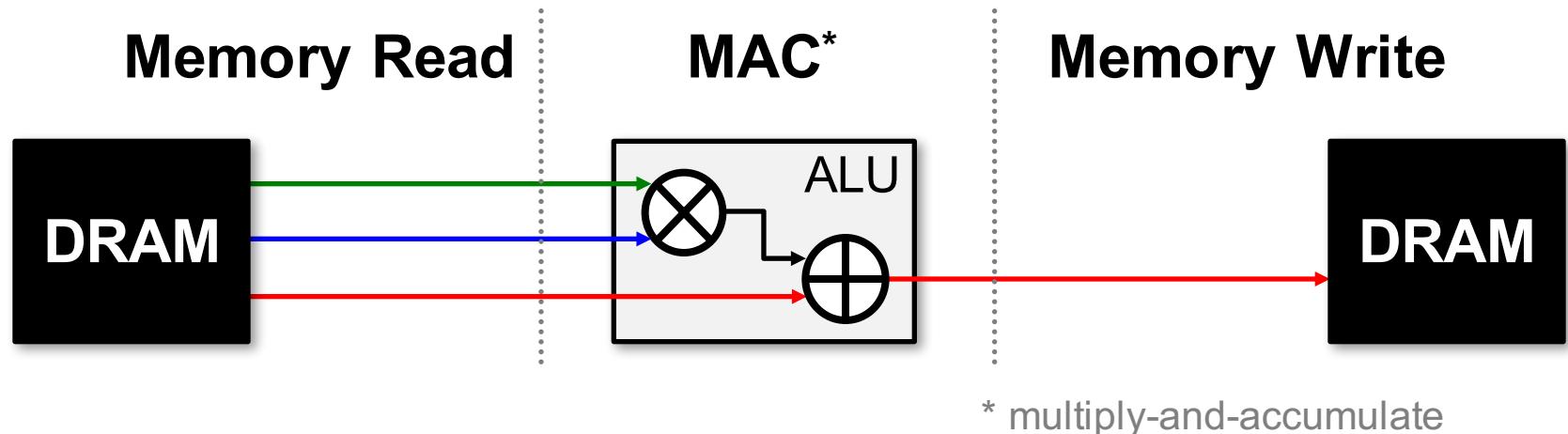
High-Dimensional CNN Convolution



Memory Access is the Bottleneck



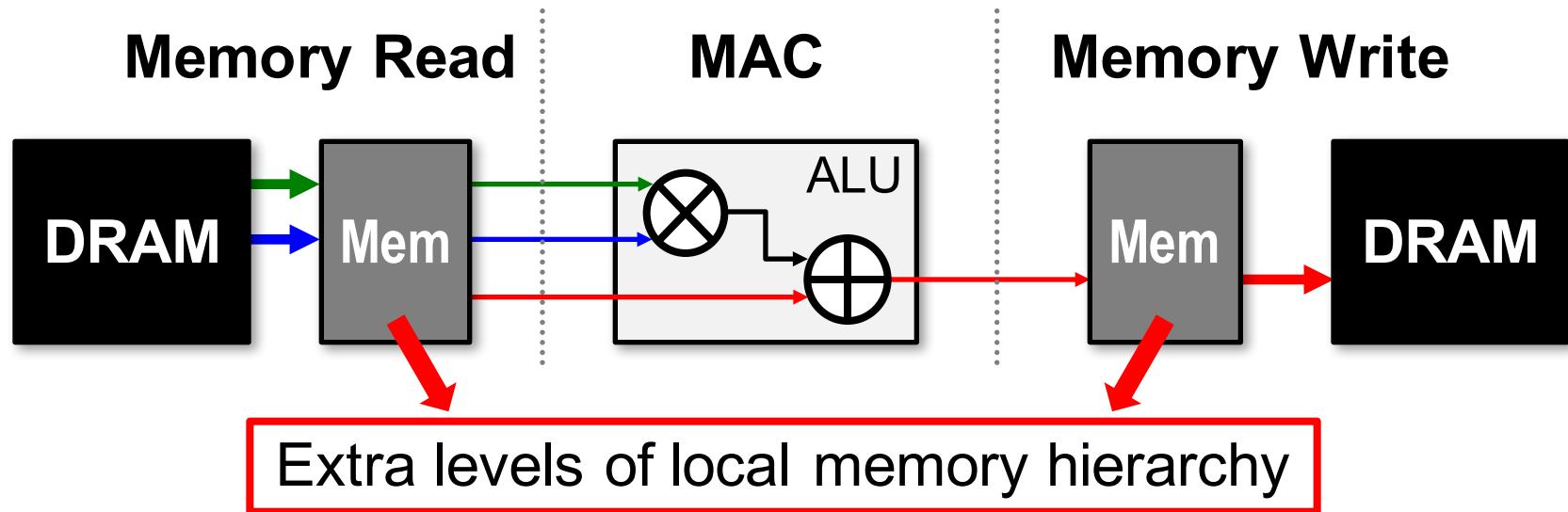
Memory Access is the Bottleneck



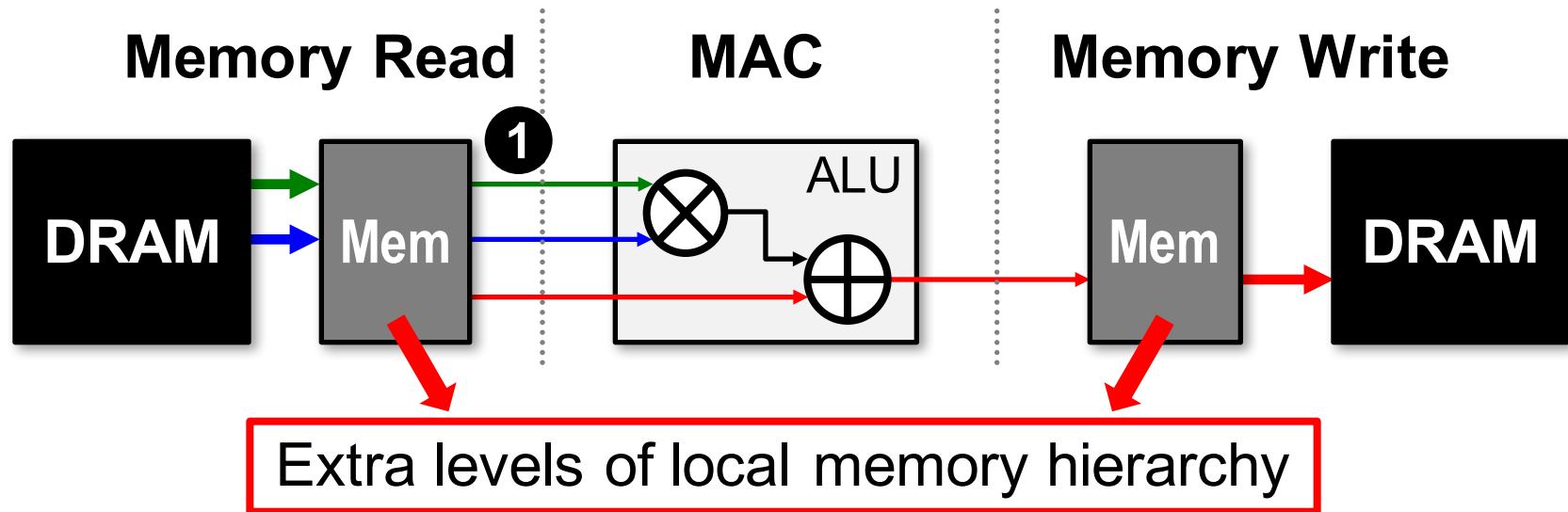
Worst Case: all memory R/W are **DRAM** accesses

- Example: AlexNet [NIPS 2012] has **724M** MACs
→ **2896M** DRAM accesses required

Memory Access is the Bottleneck



Memory Access is the Bottleneck

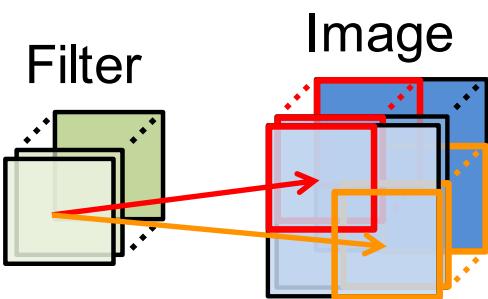


Opportunities: **① data reuse**

Types of Data Reuse in CNN

Convolutional Reuse

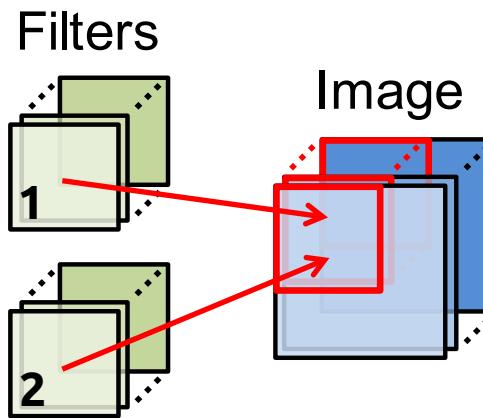
CONV layers only
(sliding window)



Reuse:
Image pixels
Filter weights

Image Reuse

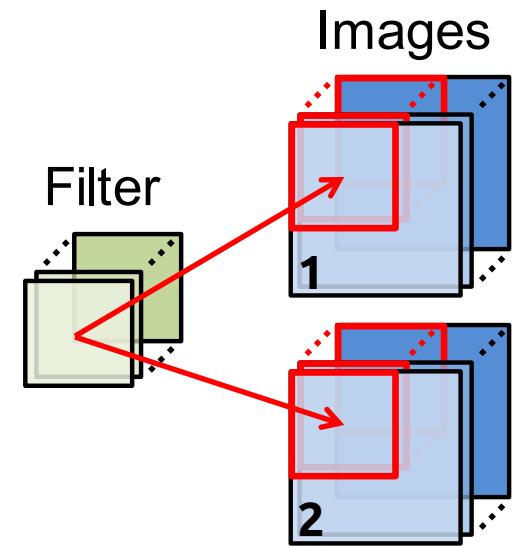
CONV and FC layers



Reuse: Image pixels

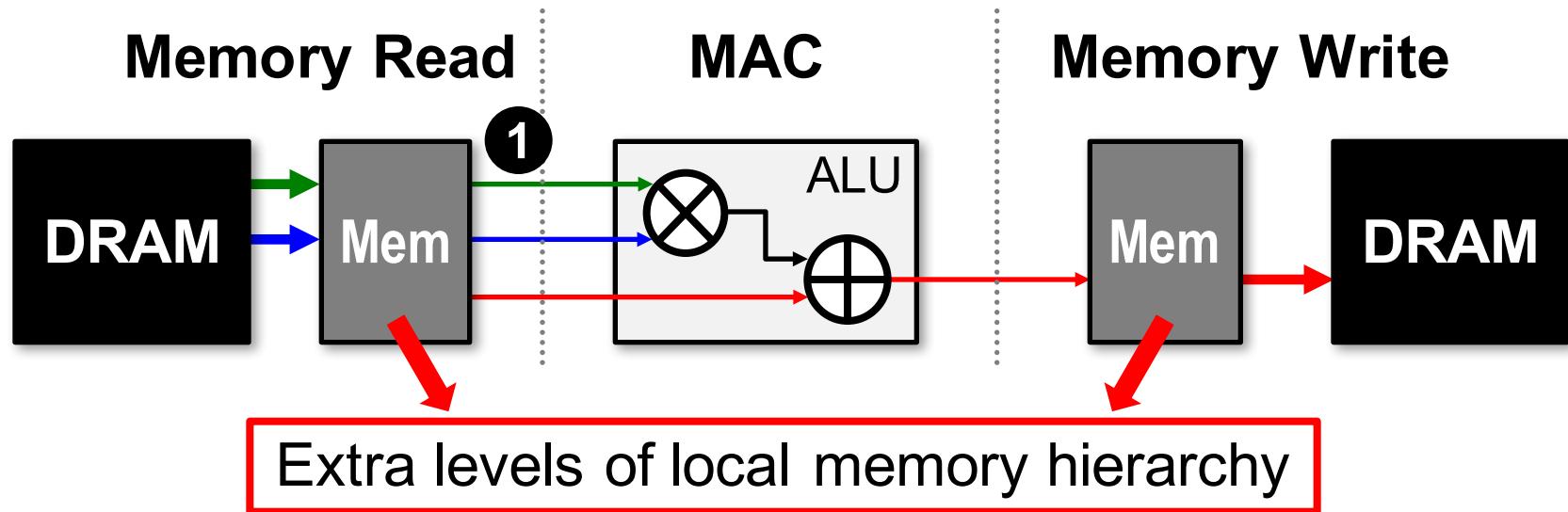
Filter Reuse

CONV and FC layers
(batch size > 1)



Reuse: Filter weights

Memory Access is the Bottleneck

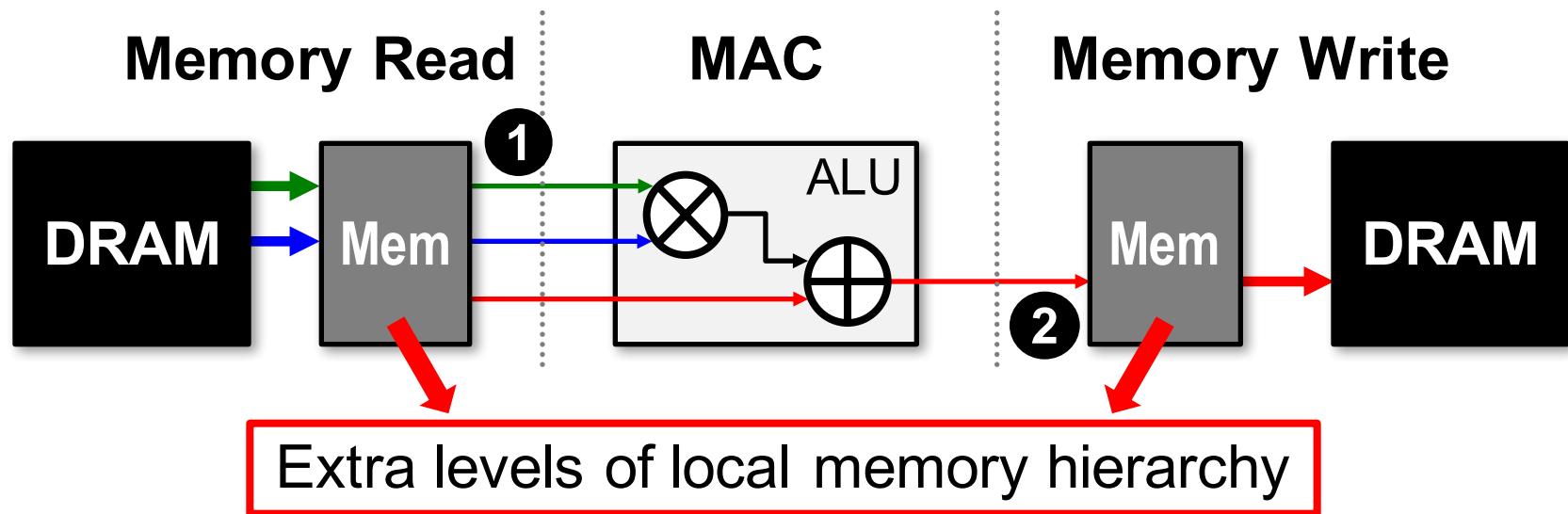


Opportunities: **① data reuse**

- ①** Can reduce DRAM reads of **filter/image** by up to **500x****

** AlexNet CONV layers

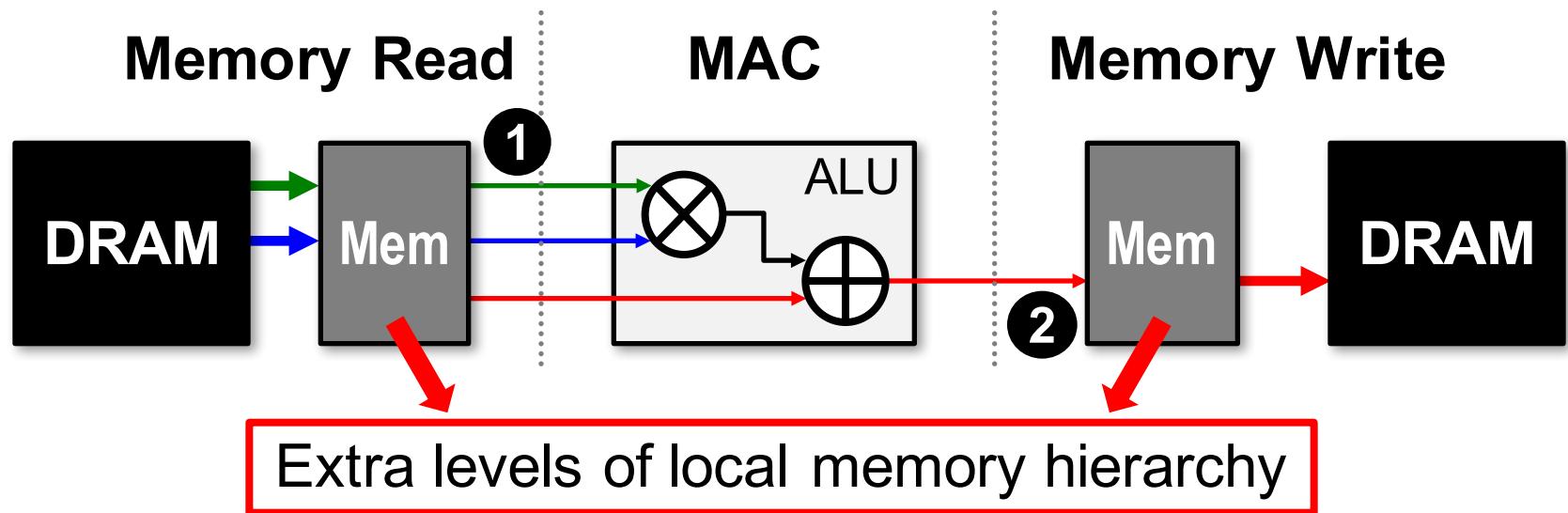
Memory Access is the Bottleneck



Opportunities: **① data reuse ② local accumulation**

- 1** Can reduce DRAM reads of **filter/image** by up to **500×**
- 2** **Partial sum** accumulation does **NOT** have to access DRAM

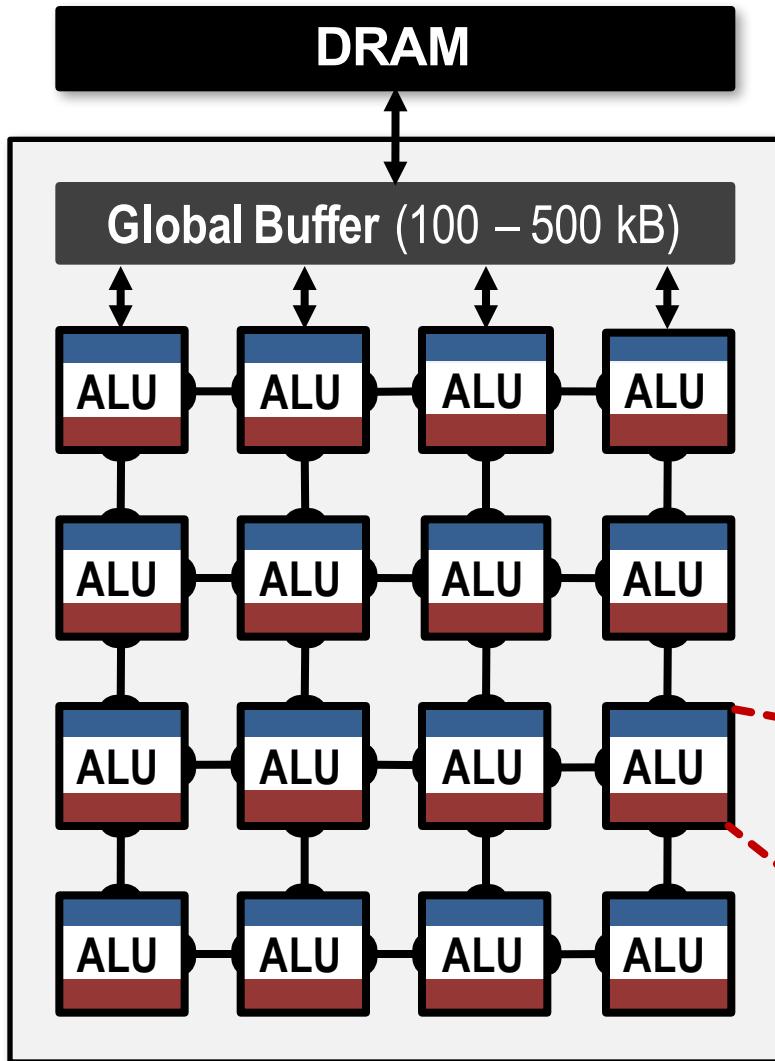
Memory Access is the Bottleneck



Opportunities: **① data reuse ② local accumulation**

- ①** Can reduce DRAM reads of **filter/image** by up to **500×**
- ②** **Partial sum** accumulation does **NOT** have to access DRAM
 - Example: DRAM access in AlexNet can be reduced from **2896M** to **61M** (best case)

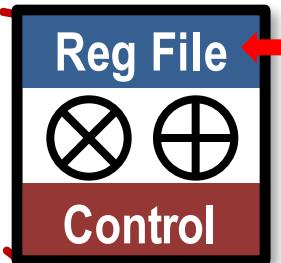
Spatial Architecture for CNN



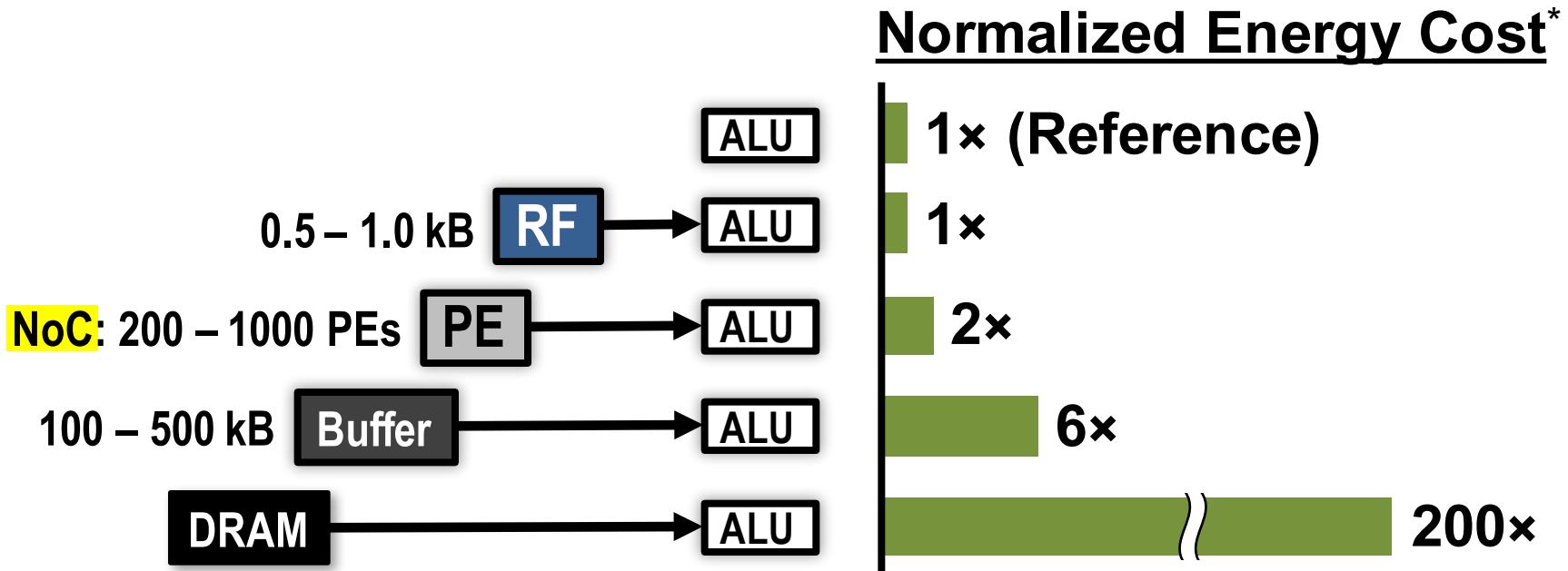
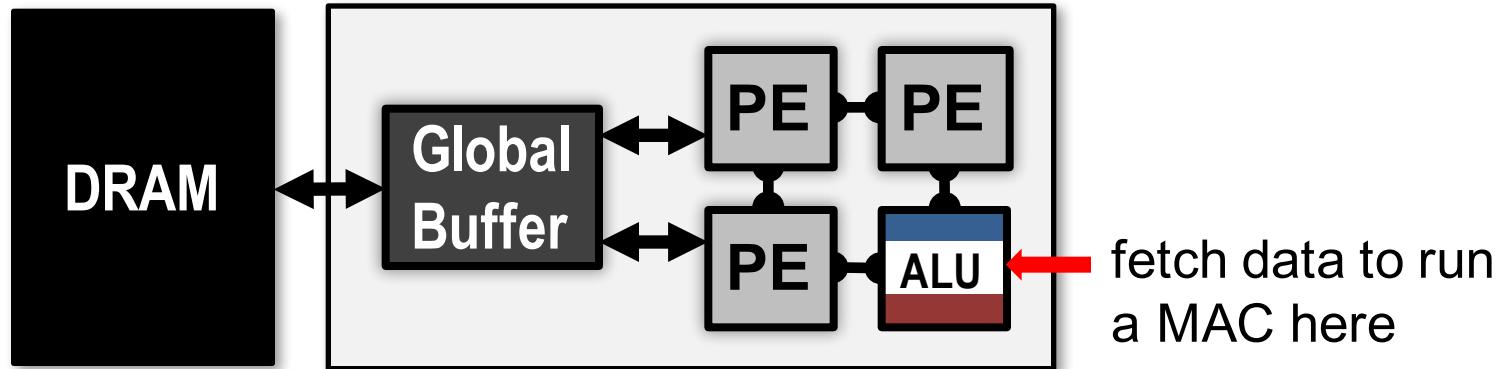
Local Memory Hierarchy

- Global Buffer
- Direct inter-PE network
- PE-local memory (RF)

Processing
Element (PE)



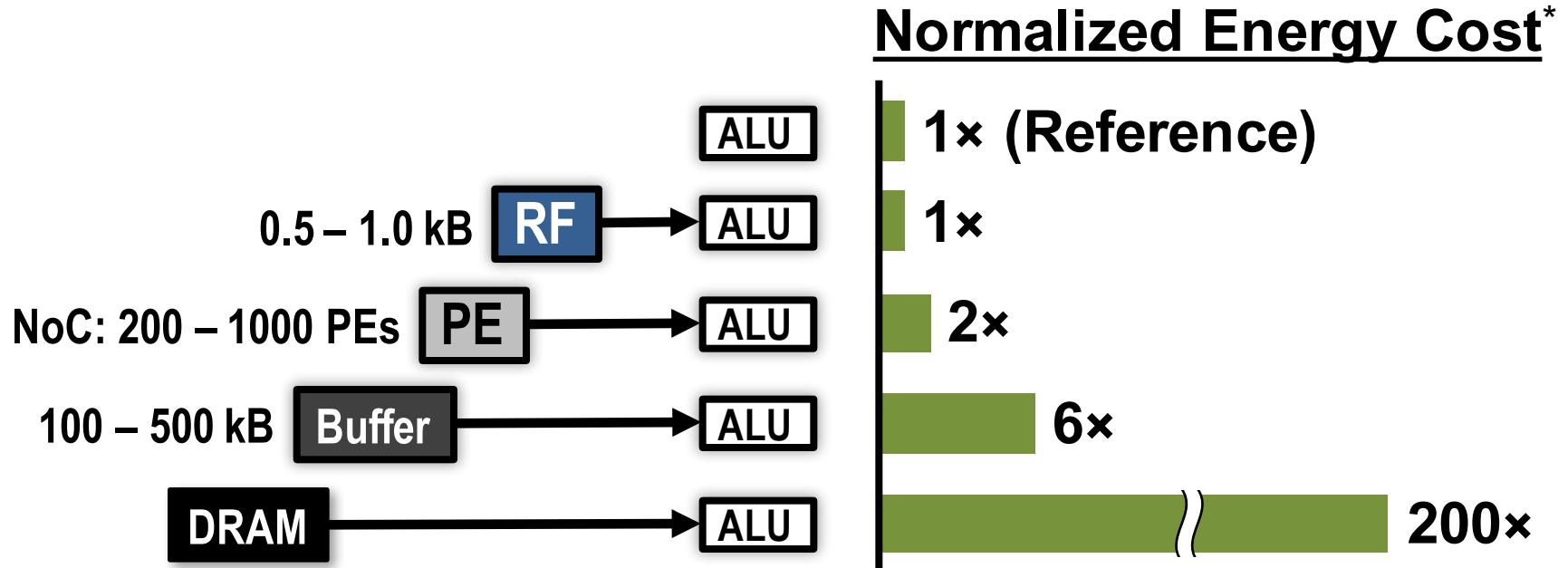
Low-Cost Local Data Access



Low-Cost Local Data Access

How to exploit ① data reuse and ② local accumulation
with *limited* low-cost local storage?

specialized processing dataflow required!

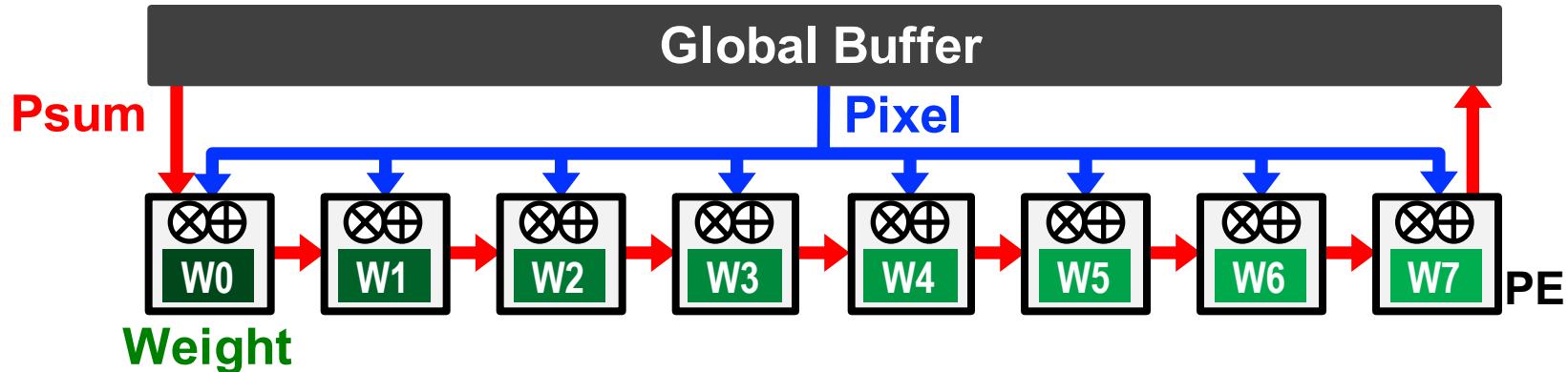


* measured from a commercial 65nm process

Taxonomy of Existing Dataflows

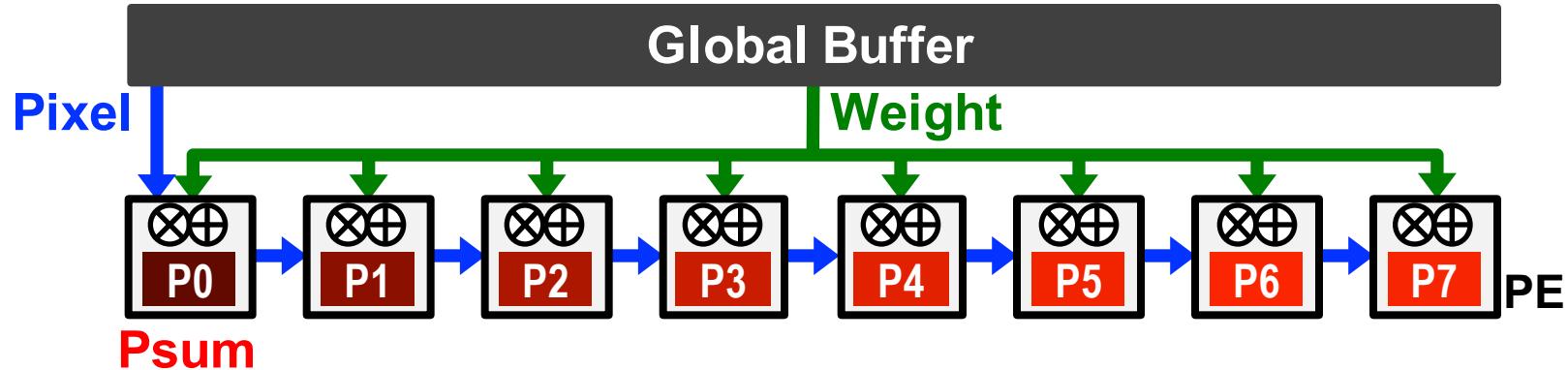
- Weight Stationary (WS)
- Output Stationary (OS)
- No Local Reuse (NLR)

Weight Stationary (WS)



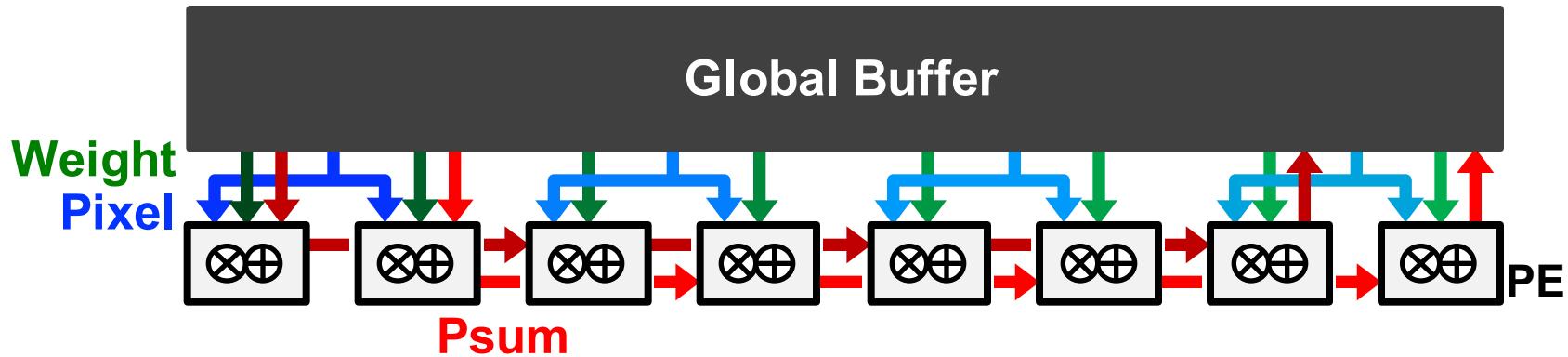
- **Minimize weight** read energy consumption
 - maximize convolutional and filter reuse of weights
- **Examples:**
 - [Chakradhar, /SCA 2010] [nn-X (NeuFlow), CVPRW 2014]
 - [Park, /SSCC 2015] [Origami, GLSVLSI 2015]

Output Stationary (OS)



- **Minimize partial sum** R/W energy consumption
 - maximize local accumulation
- **Examples:**
 - [Gupta, ICML 2015]
 - [ShiDianNao, ISCA 2015]
 - [Peemen, ICCD 2013]

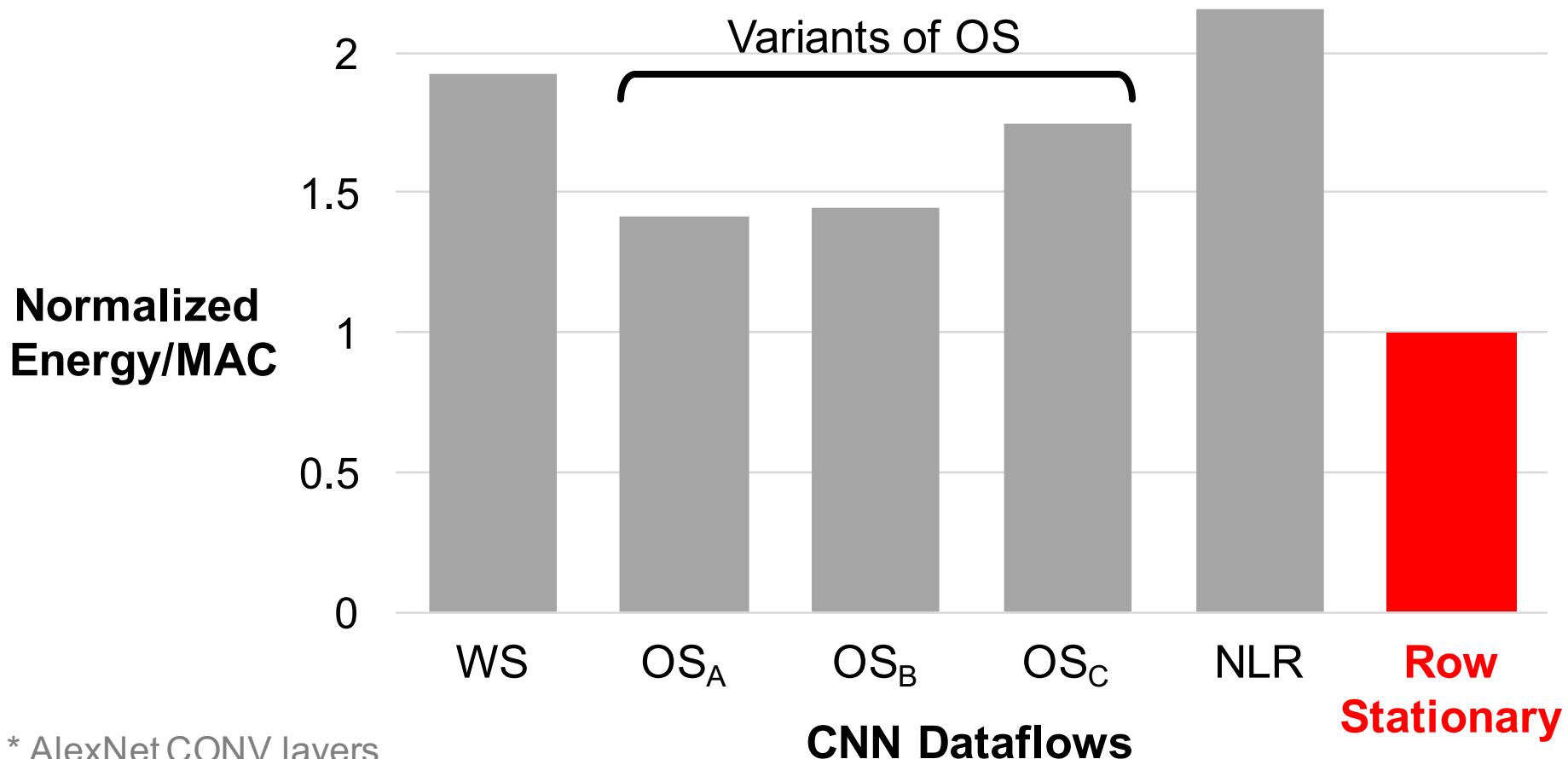
No Local Reuse (NLR)



- Use a **large global buffer** as shared storage
 - Reduce **DRAM** access energy consumption
- **Examples:**
 - [DianNao, ASPLOS 2014]
 - [DaDianNao, MICRO 2014]
 - [Zhang, FPGA 2015]

Energy Efficiency Comparison

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

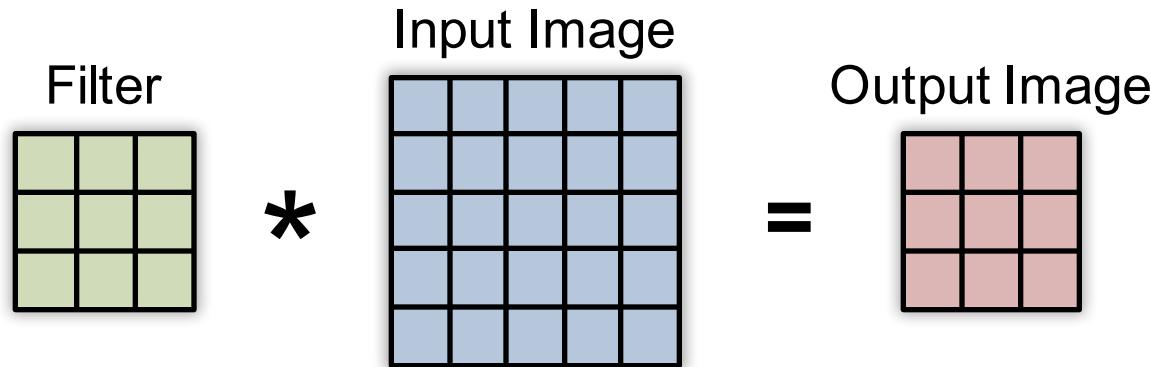


* AlexNet CONV layers

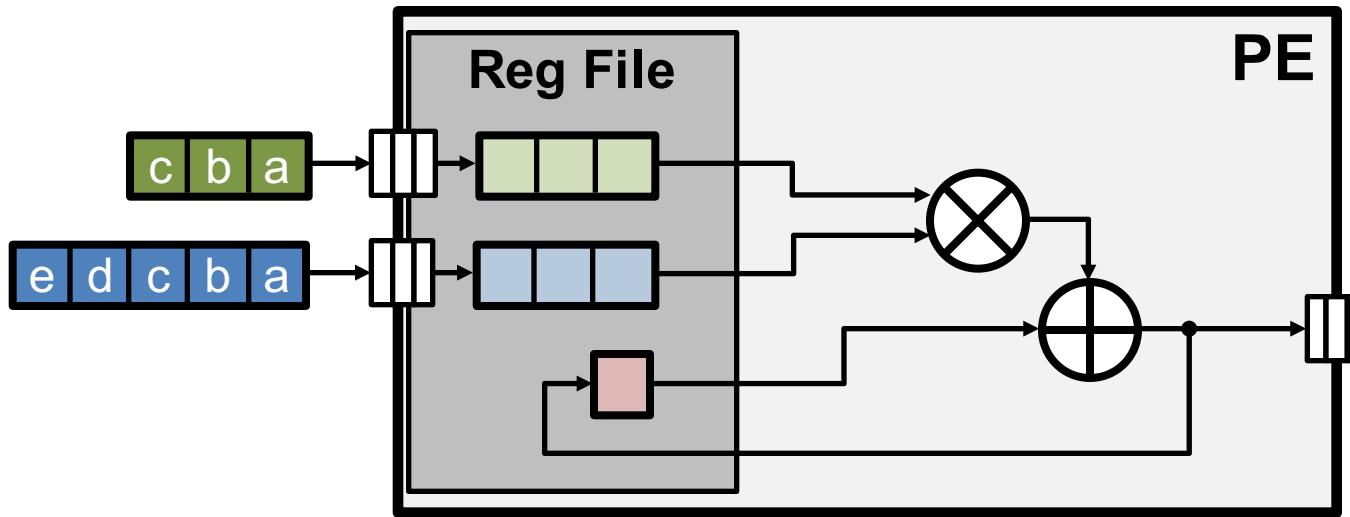
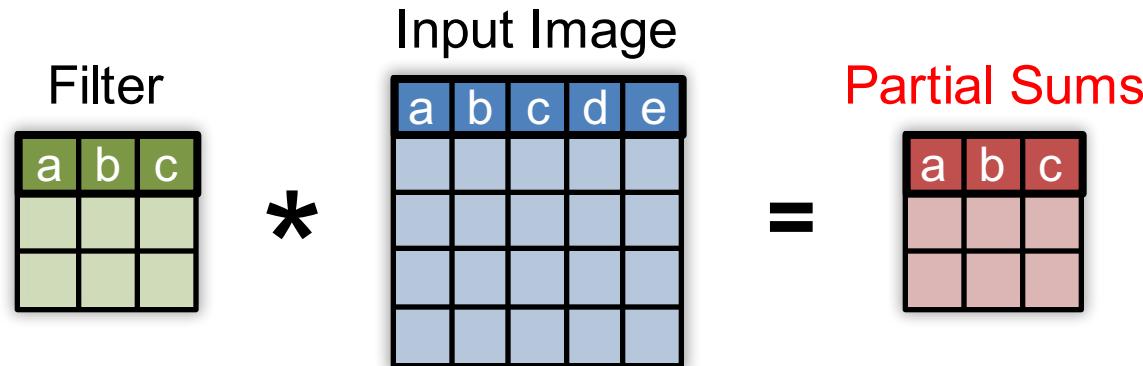
Energy-Efficient Dataflow: Row Stationary (RS)

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

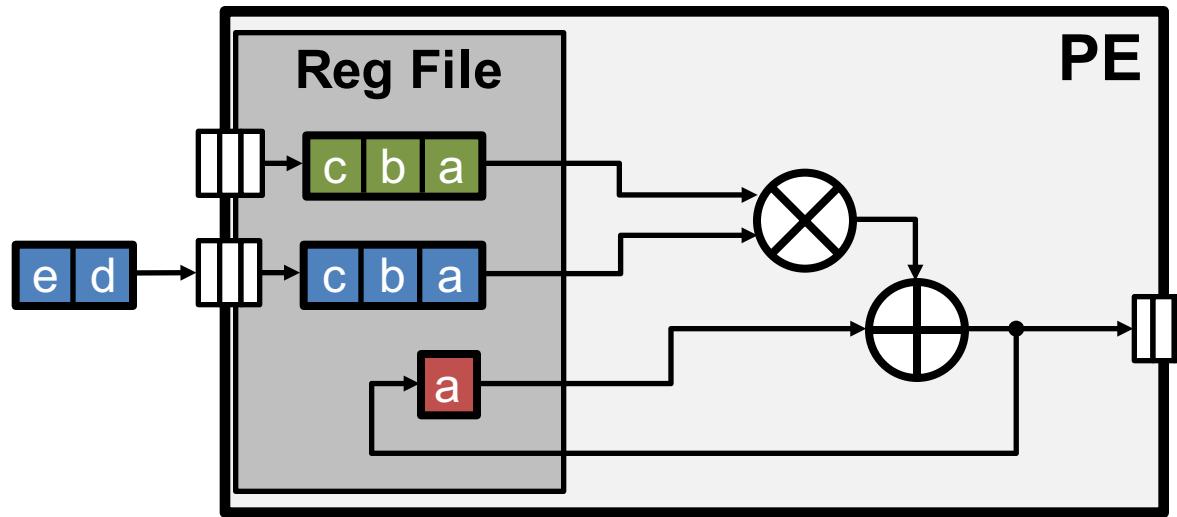
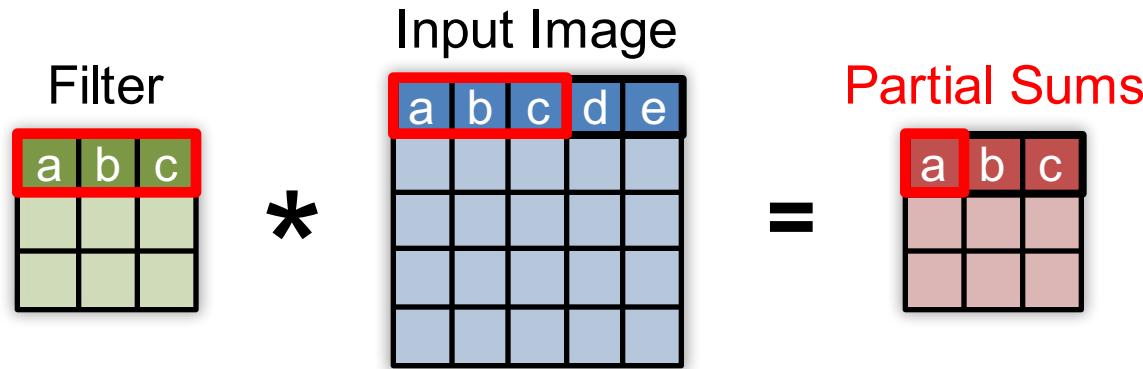
1D Row Convolution in PE



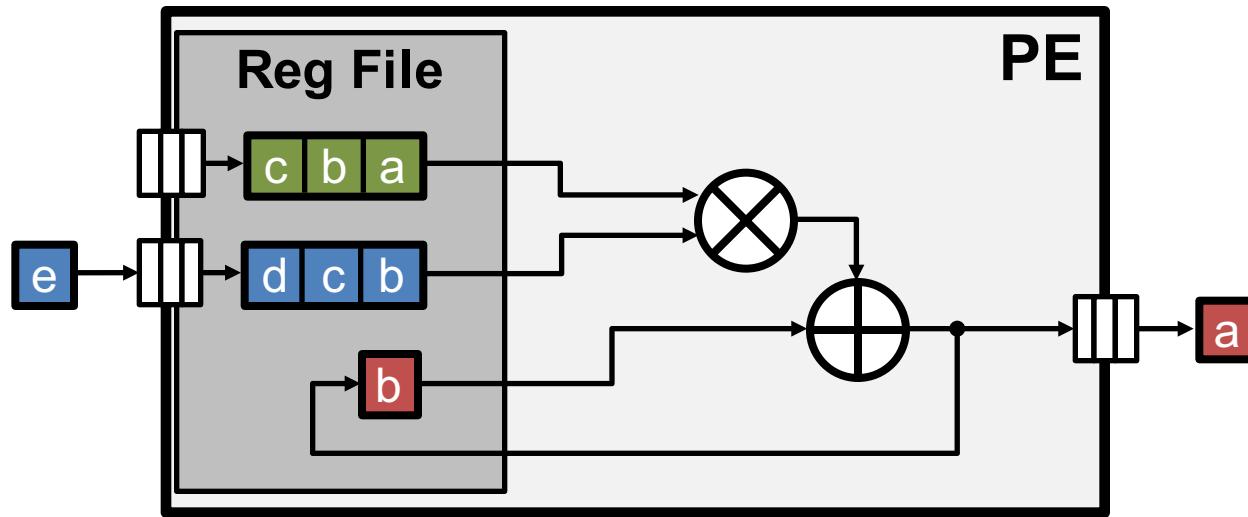
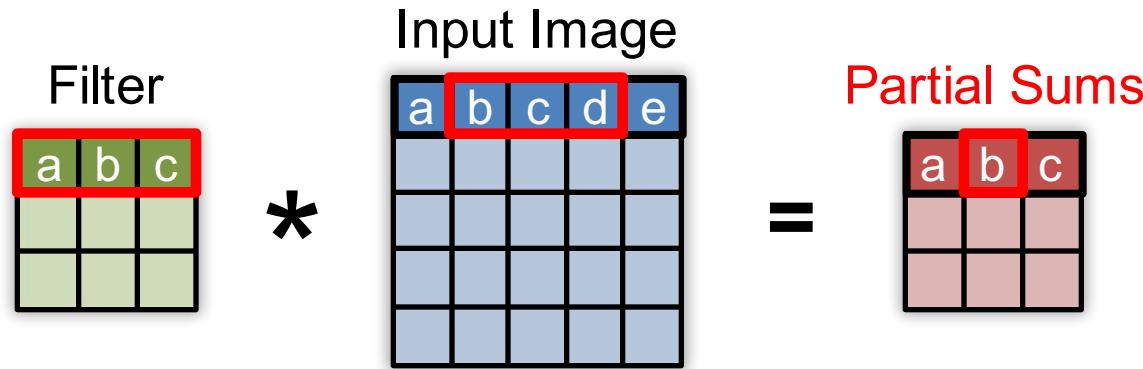
1D Row Convolution in PE



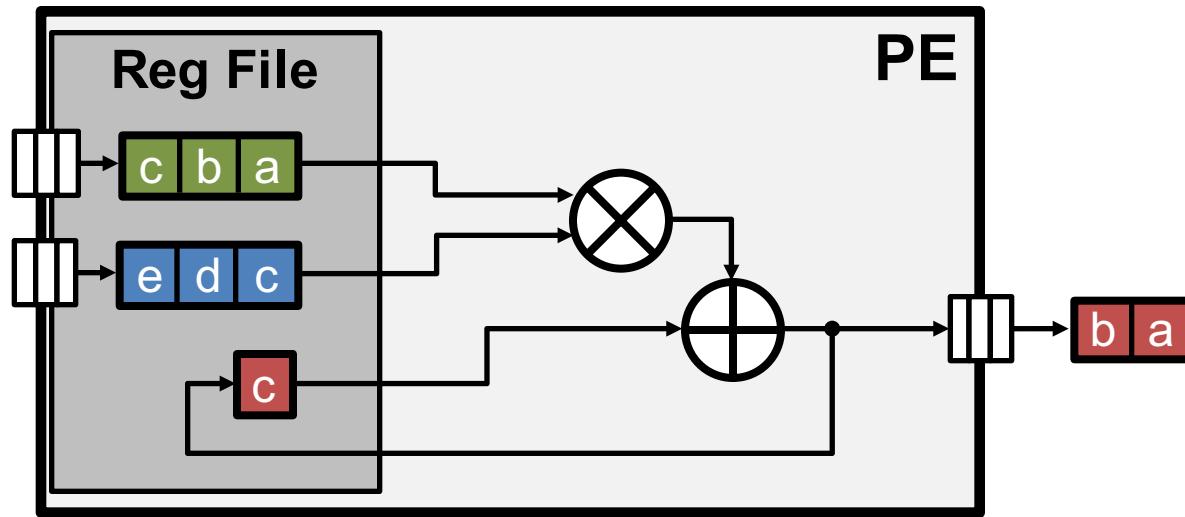
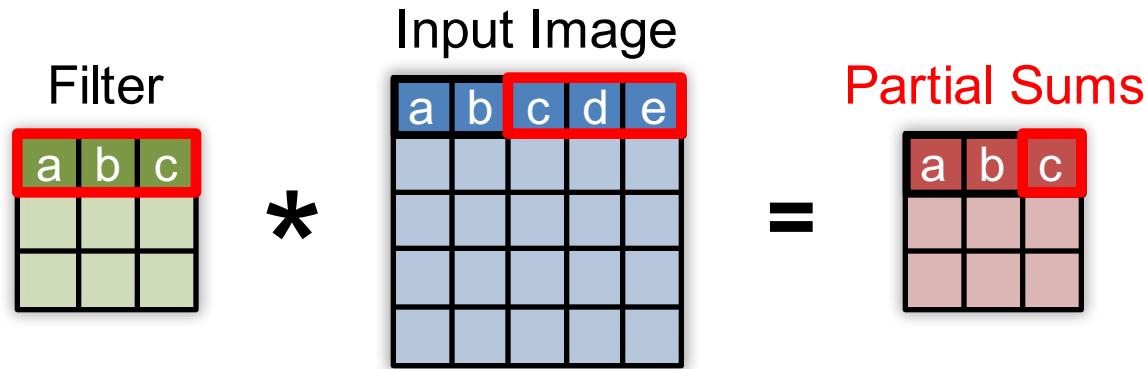
1D Row Convolution in PE



1D Row Convolution in PE

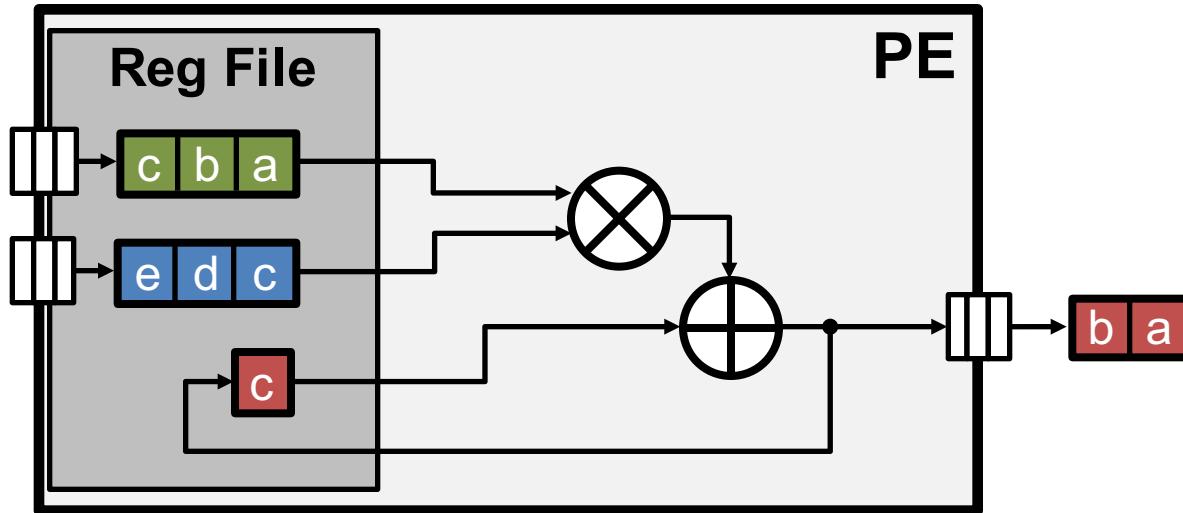


1D Row Convolution in PE



1D Row Convolution in PE

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

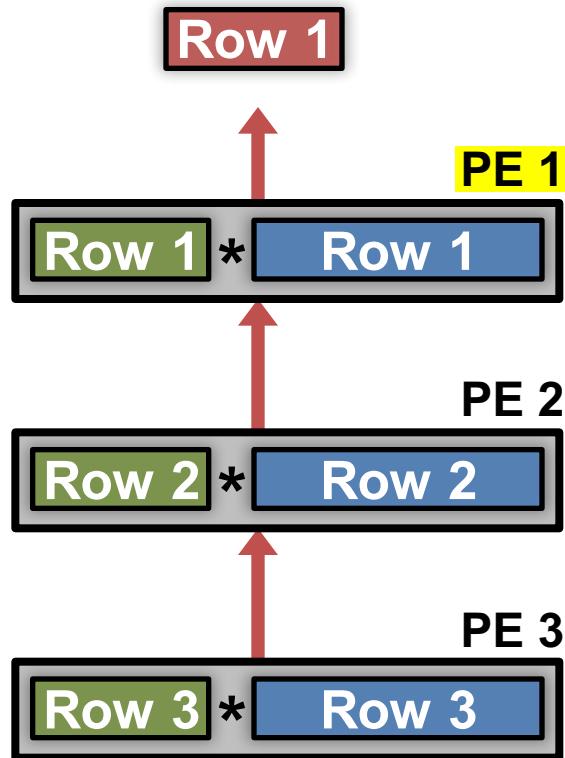


2D Convolution in PE Array



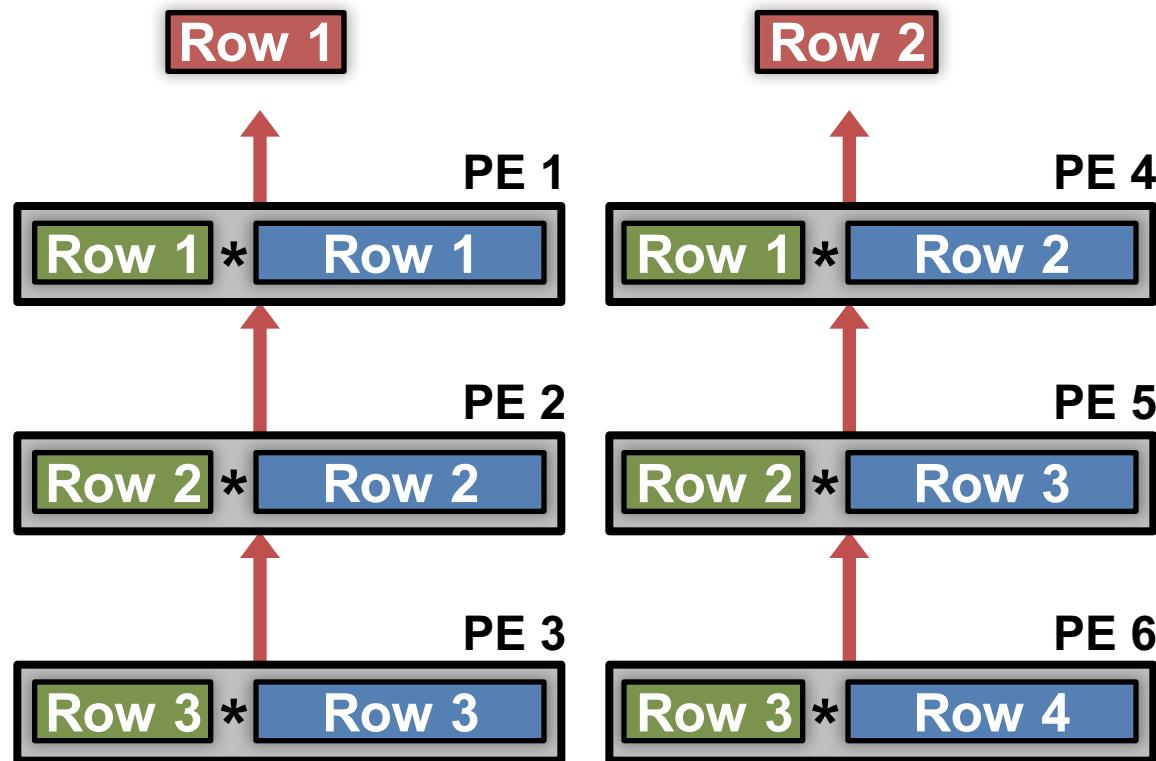
$$\begin{array}{c} \text{Input} \\ \begin{matrix} \text{Green} & \text{Yellow} \\ \text{Yellow} & \text{Green} \end{matrix} \end{array} * \begin{array}{c} \text{Filter} \\ \begin{matrix} \text{Blue} & \text{Red} & \text{Green} \\ \text{Red} & \text{Blue} & \text{Green} \\ \text{Green} & \text{Green} & \text{Blue} \end{matrix} \end{array} = \begin{array}{c} \text{Output} \\ \begin{matrix} \text{Red} & \text{Red} \\ \text{Red} & \text{Red} \end{matrix} \end{array}$$

2D Convolution in PE Array



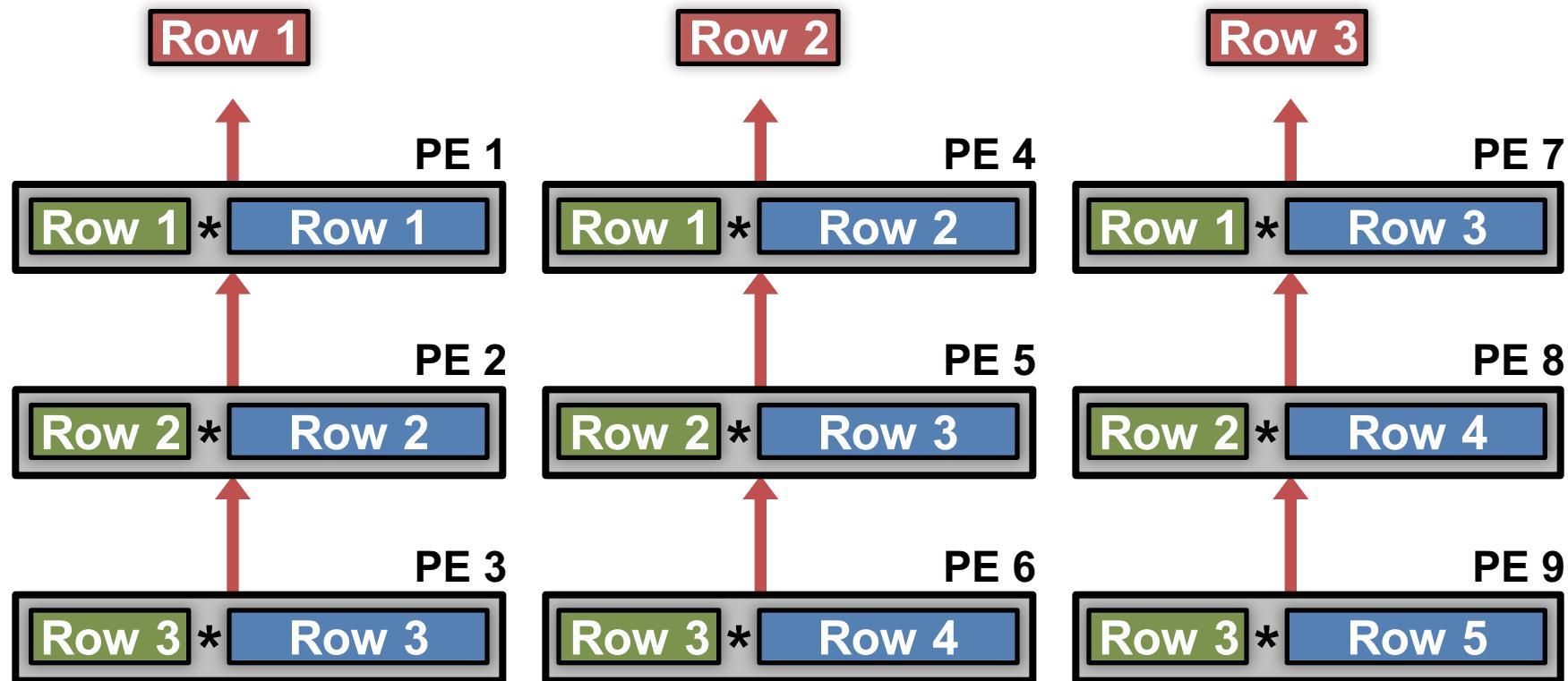
The diagram illustrates the result of a matrix multiplication. On the left, there is a green 3x3 matrix icon followed by a black asterisk symbol (*). To its right is a blue 3x3 matrix icon. An equals sign (=) follows these symbols. To the right of the equals sign is a red 3x3 matrix icon, representing the resulting product of the two matrices.

2D Convolution in PE Array



$$\begin{array}{c} \text{grid icon} \\ \times \end{array} \quad \begin{array}{c} \text{grid icon} \\ = \end{array} \quad \begin{array}{c} \text{grid icon} \end{array}$$
$$\begin{array}{c} \text{grid icon} \\ \times \end{array} \quad \begin{array}{c} \text{grid icon} \\ = \end{array} \quad \begin{array}{c} \text{grid icon} \end{array}$$

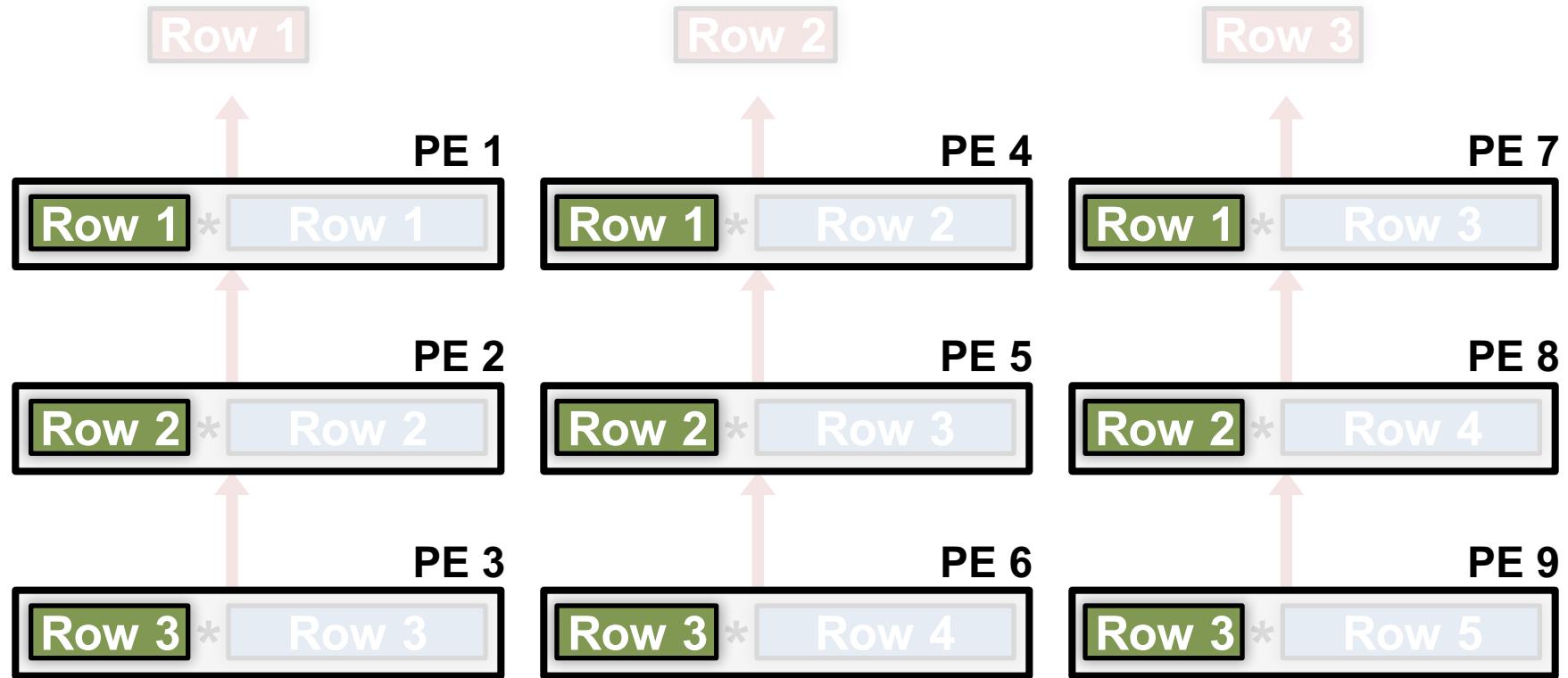
2D Convolution in PE Array



Below the main diagram, three examples show the element-wise multiplication ($*$) of two 3x3 grids:

- $\begin{bmatrix} \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \end{bmatrix} * \begin{bmatrix} \text{blue} & \text{blue} & \text{blue} \\ \text{blue} & \text{blue} & \text{blue} \\ \text{blue} & \text{blue} & \text{blue} \end{bmatrix} = \begin{bmatrix} \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \end{bmatrix}$
- $\begin{bmatrix} \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \end{bmatrix} * \begin{bmatrix} \text{blue} & \text{blue} & \text{blue} \\ \text{blue} & \text{blue} & \text{blue} \\ \text{blue} & \text{blue} & \text{blue} \end{bmatrix} = \begin{bmatrix} \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \end{bmatrix}$
- $\begin{bmatrix} \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \end{bmatrix} * \begin{bmatrix} \text{blue} & \text{blue} & \text{blue} \\ \text{blue} & \text{blue} & \text{blue} \\ \text{blue} & \text{blue} & \text{blue} \end{bmatrix} = \begin{bmatrix} \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \\ \text{green} & \text{green} & \text{green} \end{bmatrix}$

Convolutional Reuse Maximized



Filter rows are reused across PEs **horizontally**

Convolutional Reuse Maximized

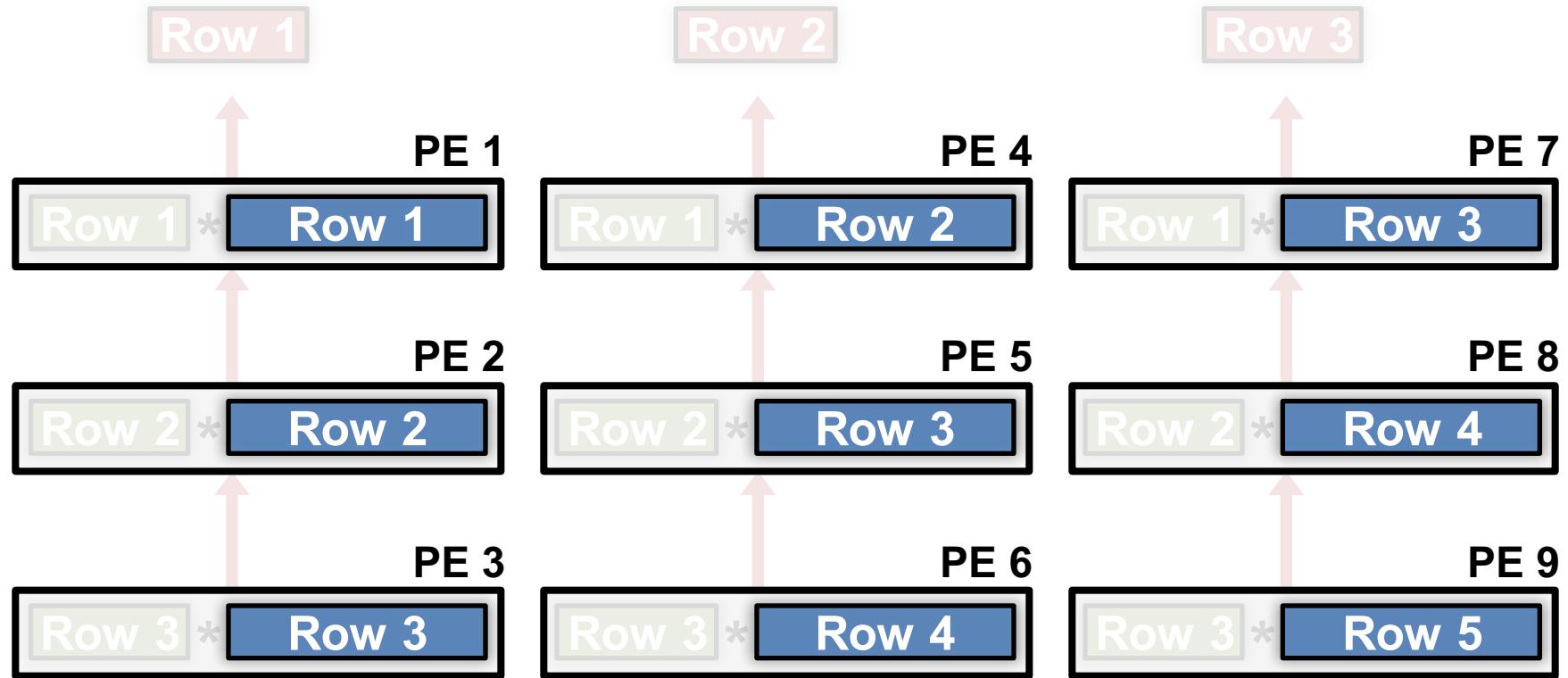
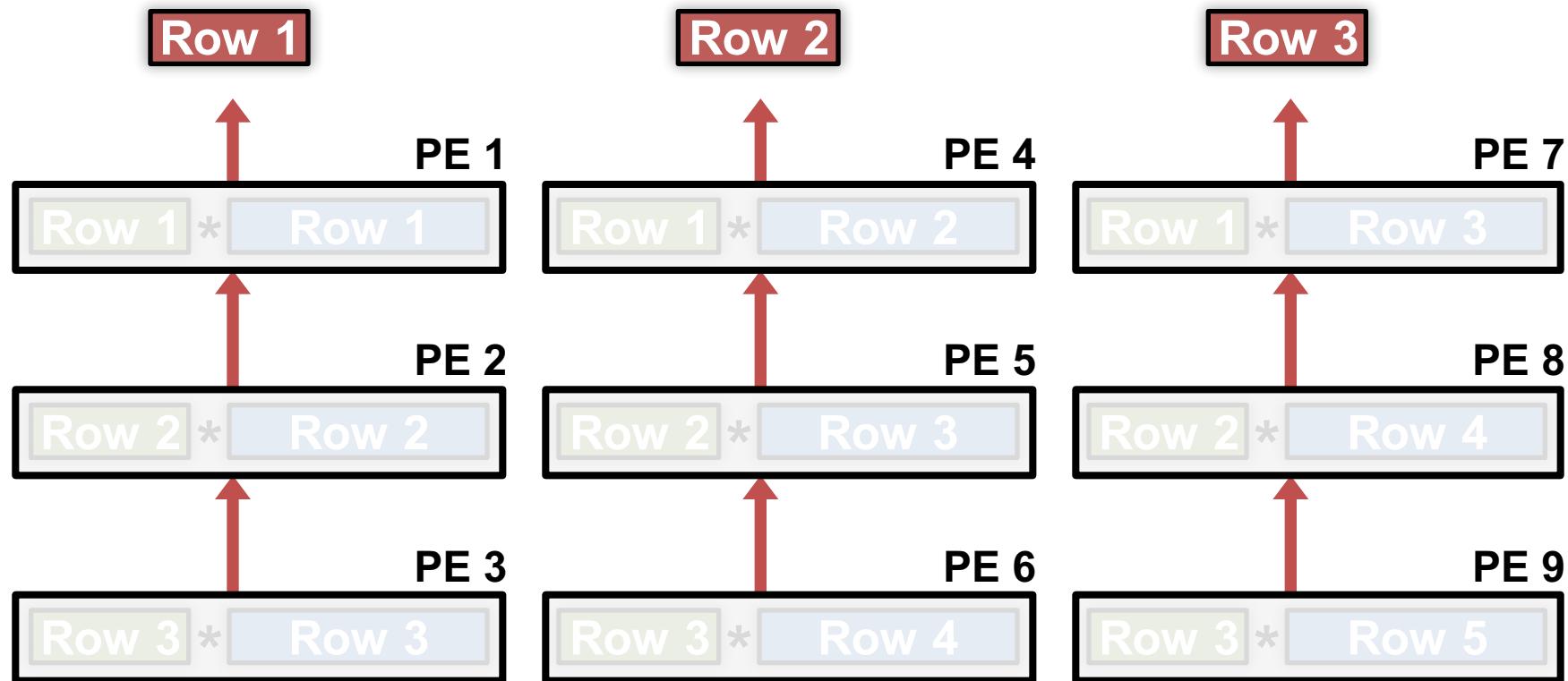


Image rows are reused across PEs **diagonally**

Maximize 2D Accumulation in PE Array



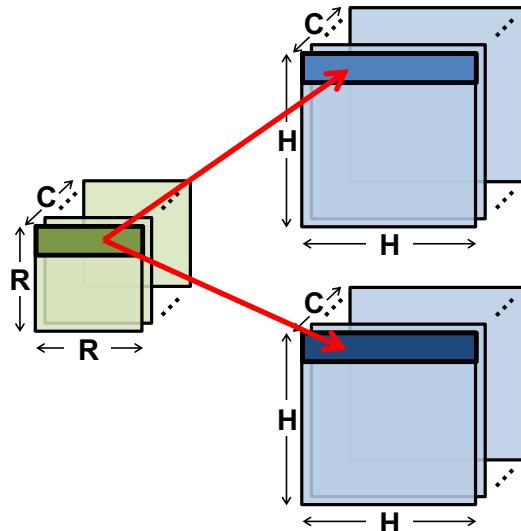
Partial sums accumulate across PEs **vertically**

Dimensions Beyond 2D Convolution

- 1 Multiple Images
- 2 Multiple Filters
- 3 Multiple Channels

Filter Reuse in PE

1 Multiple Images



2 Multiple Filters

3 Multiple Channels

Channel 1

Filter 1

Row 1

Image 1

Row 1

Psum 1

Row 1

Channel 1

Filter 1

Row 1

Image 2

Row 1

Psum 2

Row 1

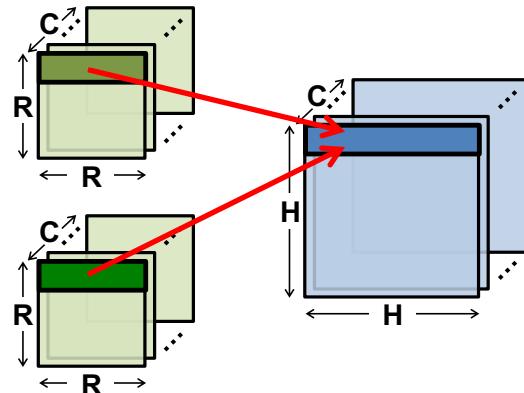
share the same filter row

Processing in PE: **concatenate image rows**

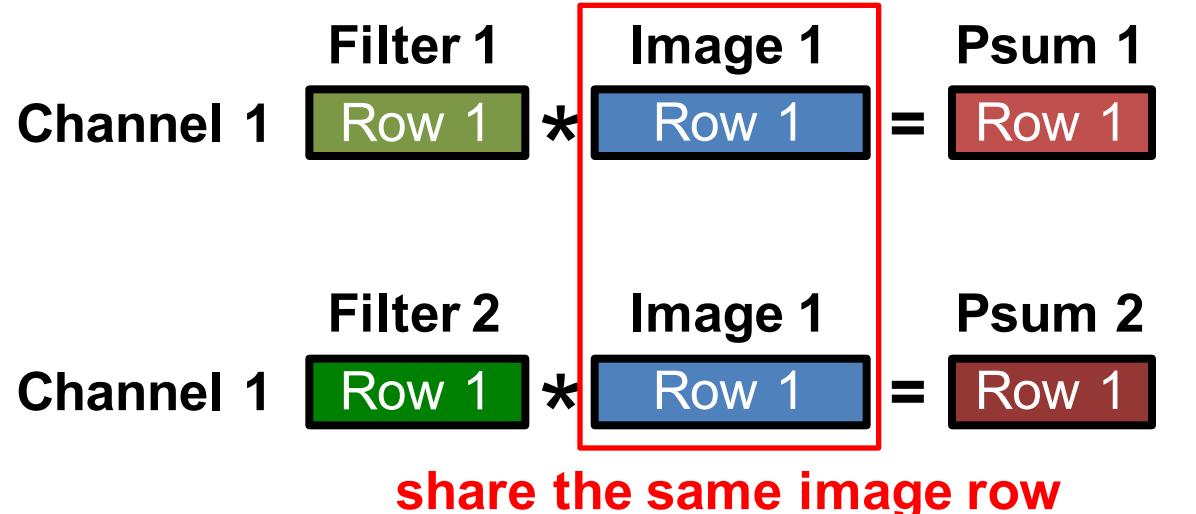
$$\begin{array}{c} \text{Filter 1} & \text{Image 1 \& 2} & \text{Psum 1 \& 2} \\ \text{Channel 1} & \text{Row 1} * \text{Row 1} \quad \text{Row 1} & = \text{Row 1} \quad \text{Row 1} \end{array}$$

Image Reuse in PE

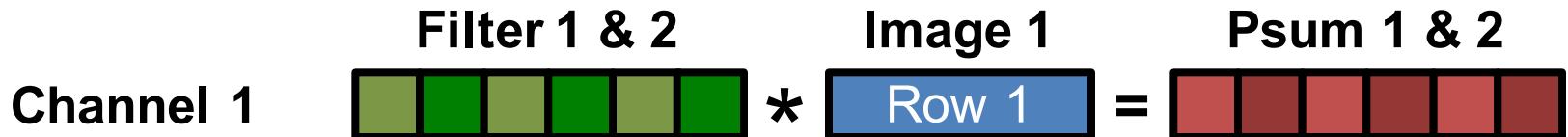
1 Multiple Images



2 Multiple Filters



Processing in PE: interleave filter rows

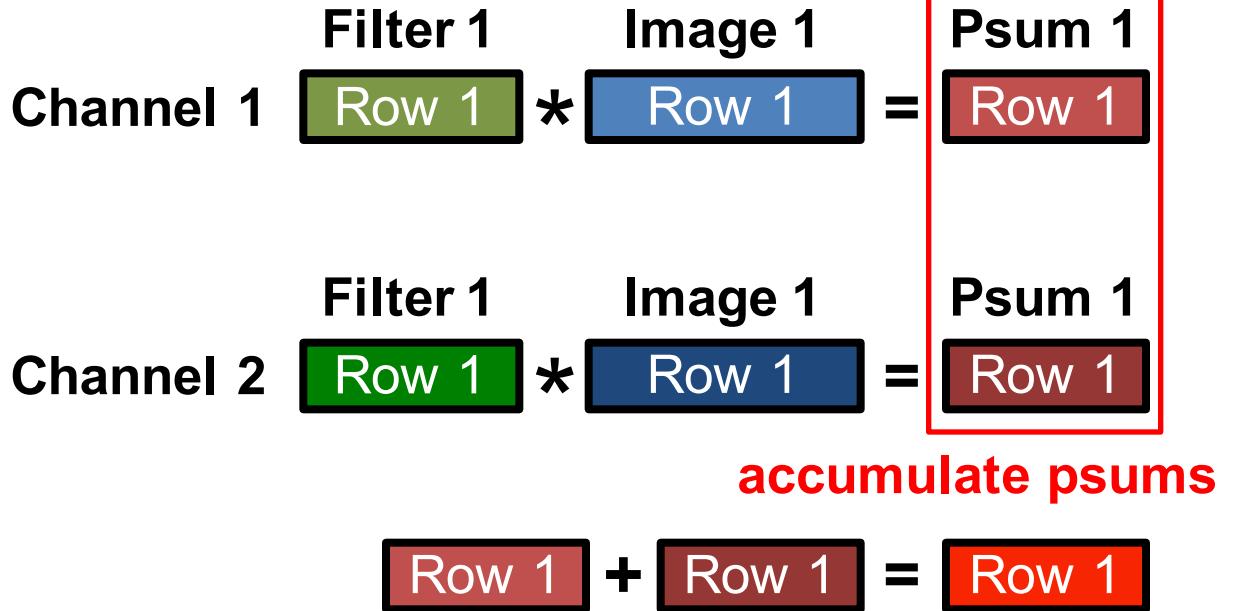
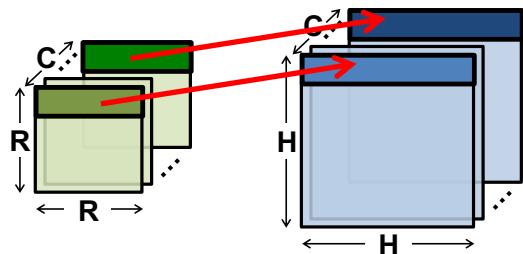


Channel Accumulation in PE

1 Multiple Images

2 Multiple Filters

3 Multiple Channels

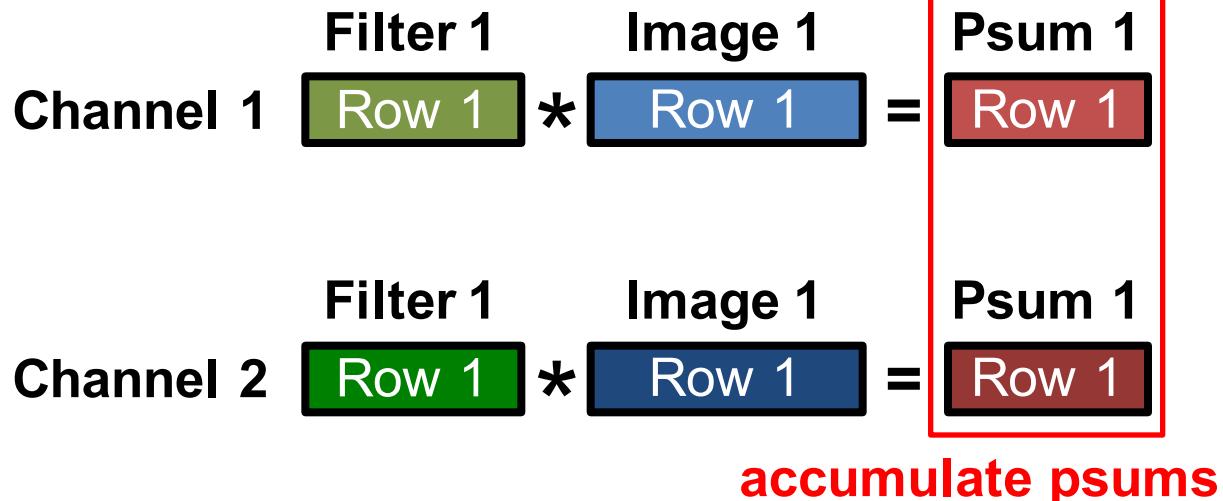
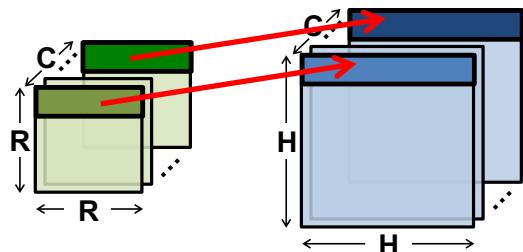


Channel Accumulation in PE

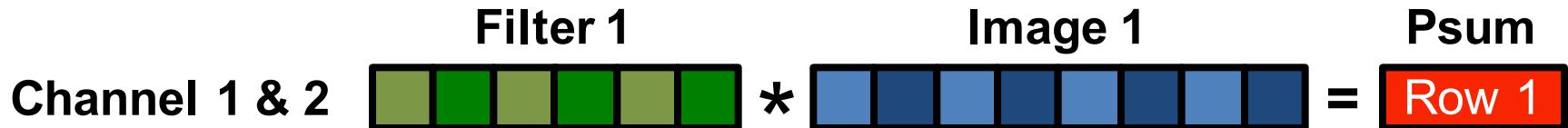
1 Multiple Images

2 Multiple Filters

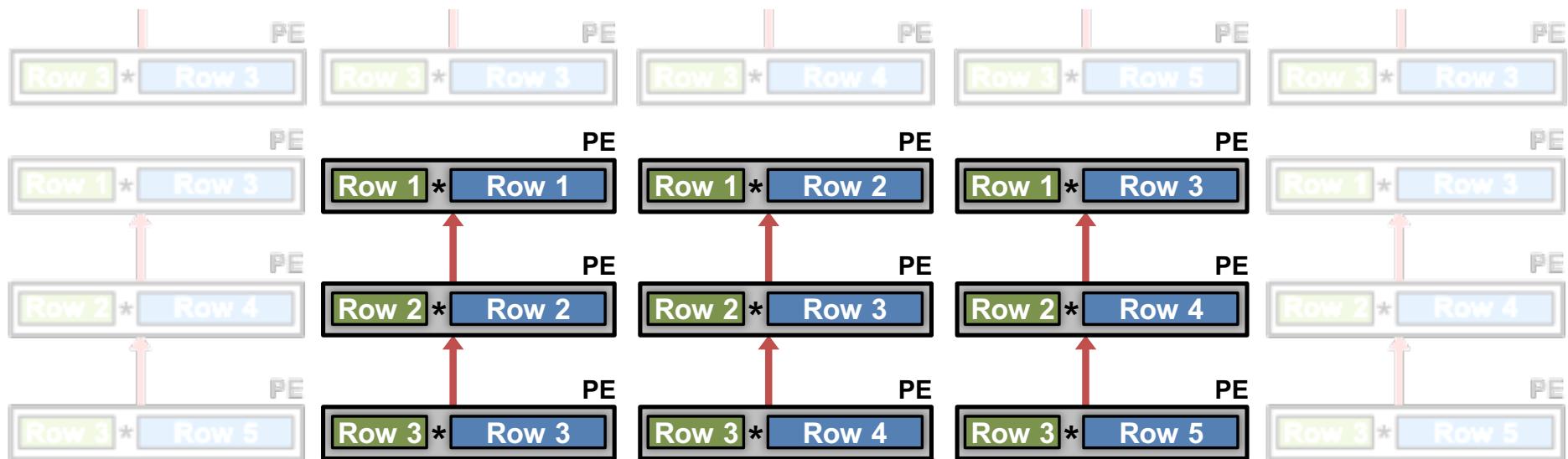
3 Multiple Channels



Processing in PE: interleave channels



CNN Convolution – The Full Picture



Multiple **images**:

$$\text{Filter 1} * \text{Image 1 & 2} = \text{Psum 1 & 2}$$

Multiple **filters**:

$$\text{Filter 1 & 2} * \text{Image 1} = \text{Psum 1 & 2}$$

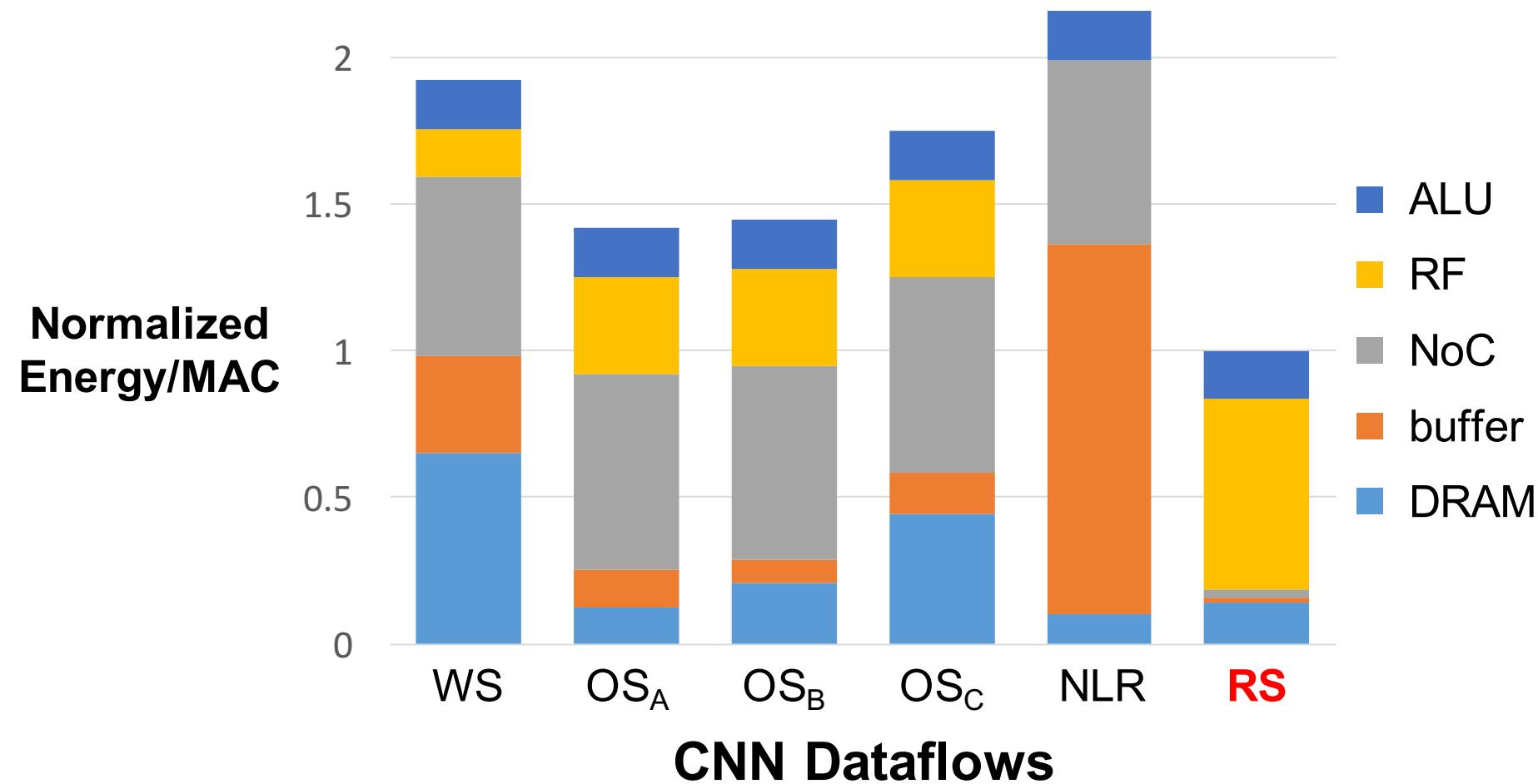
Multiple **channels**:

$$\text{Filter 1} * \text{Image 1} = \text{Psum}$$

Simulation Results

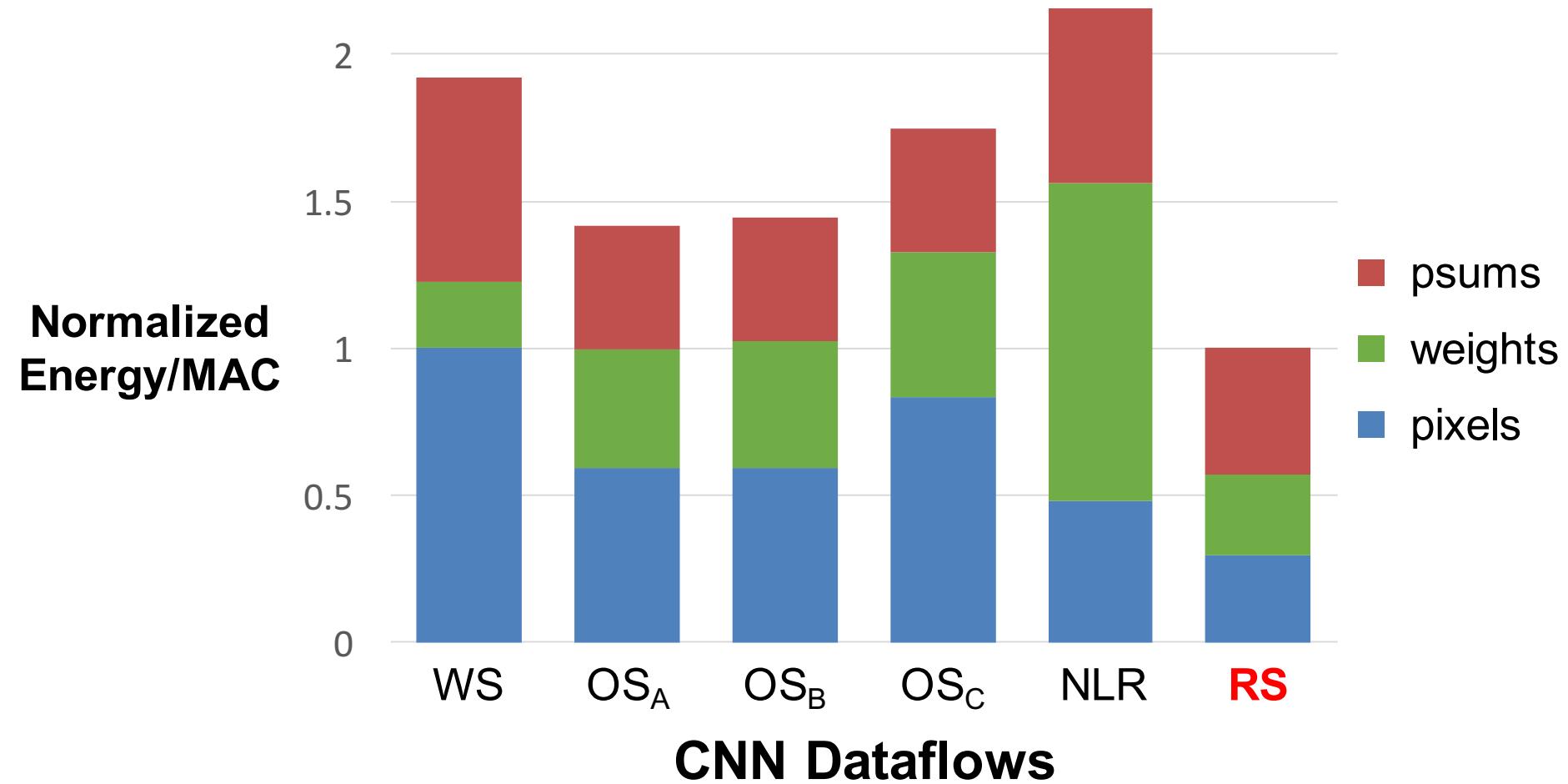
- Same total hardware area
- 256 PEs
- AlexNet Configuration
- Batch size = 16

Dataflow Comparison: CONV Layers



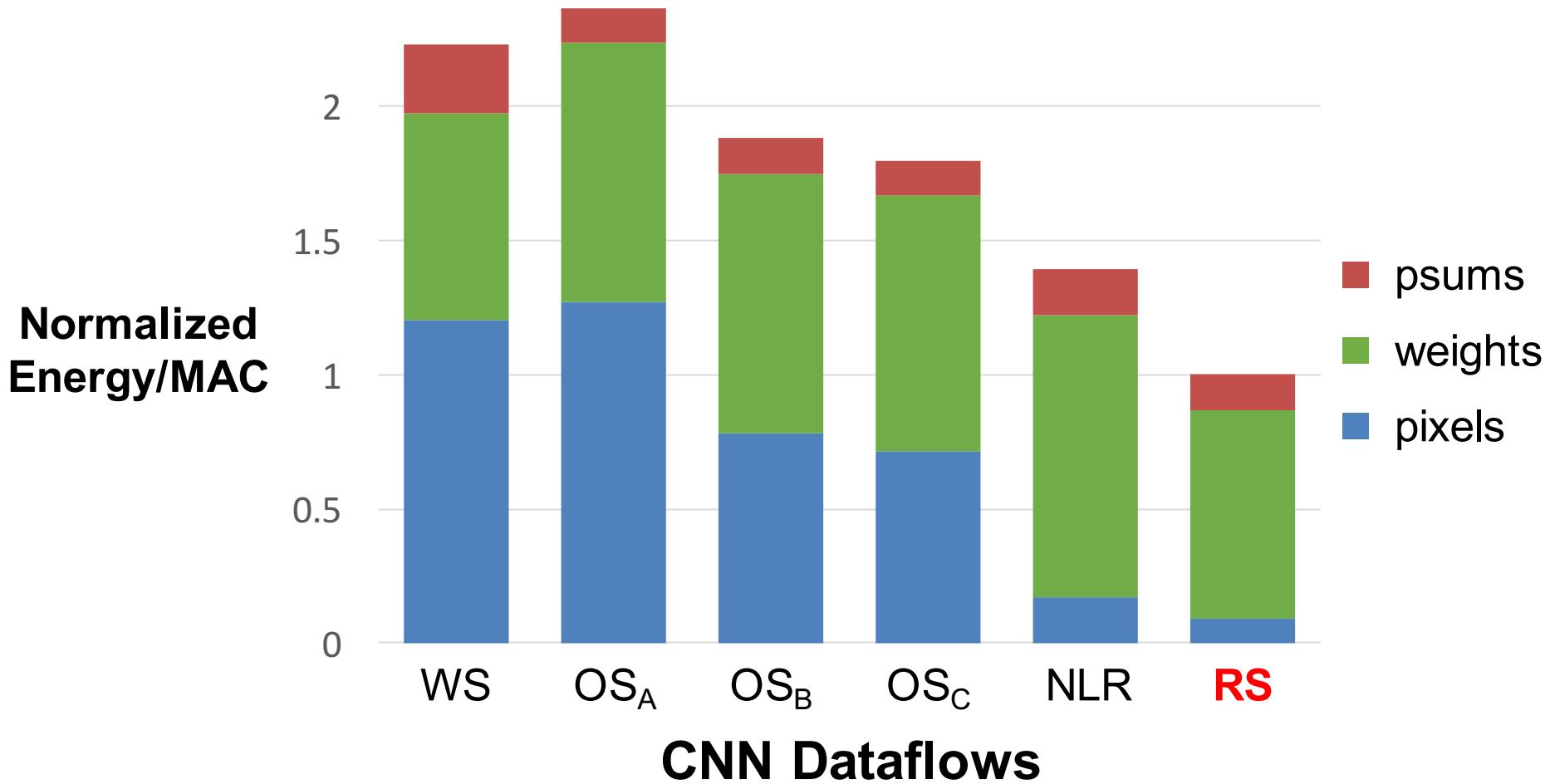
RS uses 1.4x – 2.5x lower energy than other dataflows

Dataflow Comparison: CONV Layers



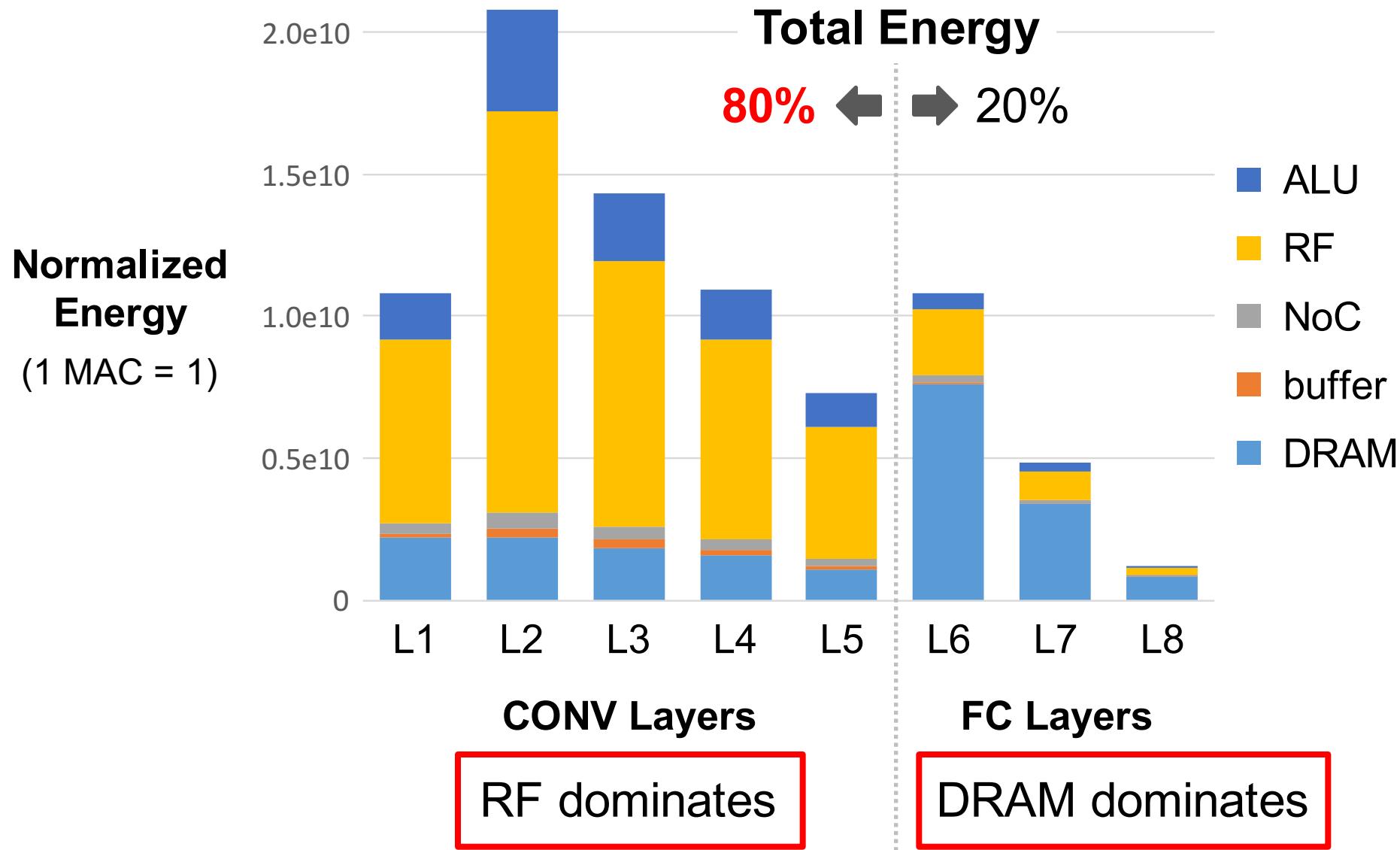
RS optimizes for the best **overall** energy efficiency

Dataflow Comparison: FC Layers



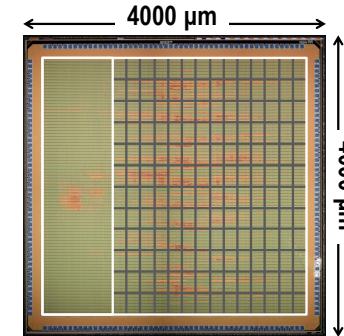
RS uses at least 1.3× lower energy than other dataflows

Row Stationary: Layer Breakdown



Summary

- We propose a **Row Stationary** (RS) dataflow to exploit the **low-cost local memories** in a spatial architecture.
- RS optimizes for best **overall** energy efficiency while existing CNN dataflows only focus on certain data types.
- RS has higher energy efficiency than existing dataflows
 - **1.4x – 2.5x** higher in **CONV** layers
 - at least **1.3x** higher in **FC** layers. (batch size ≥ 16)
- We have verified RS in a fabricated CNN processor chip, **Eyeriss**



Thank You

Learn more about **Eyeriss** at
<http://eyeriss.mit.edu>

