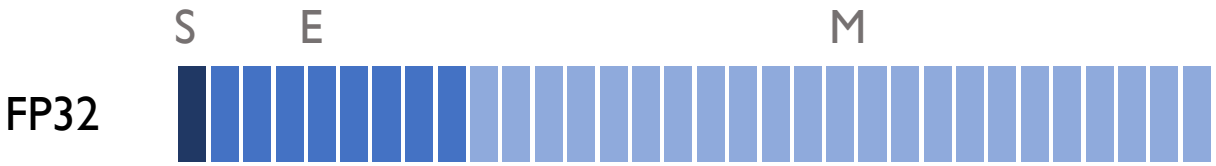
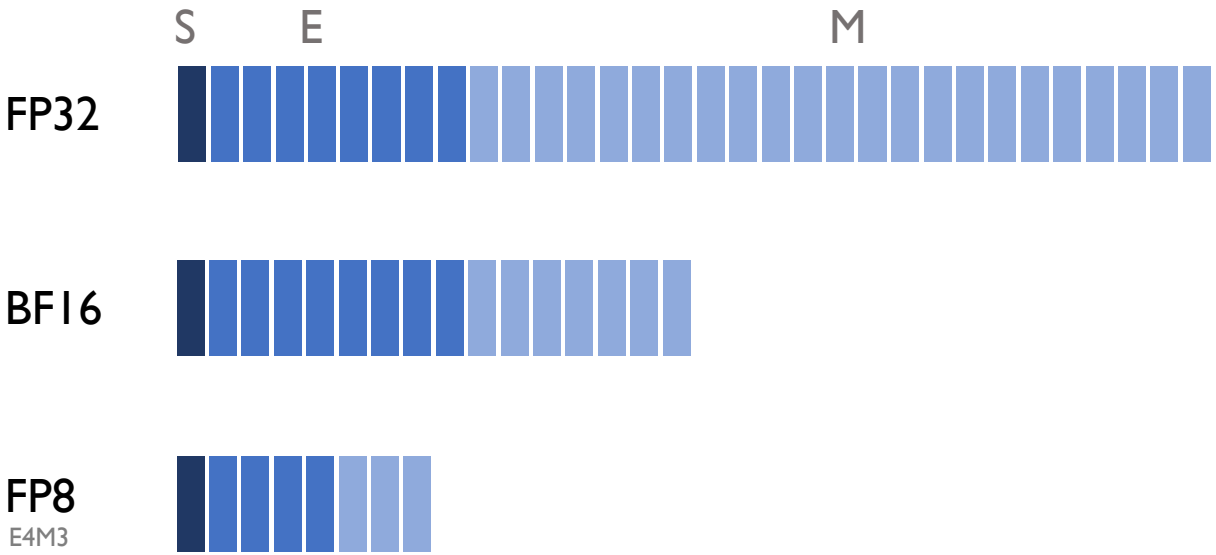
Carat: Unlocking Value-Level Parallelism for Multiplier-Free GEMMs

Zhewen Pan Joshua San Miguel Di Wu

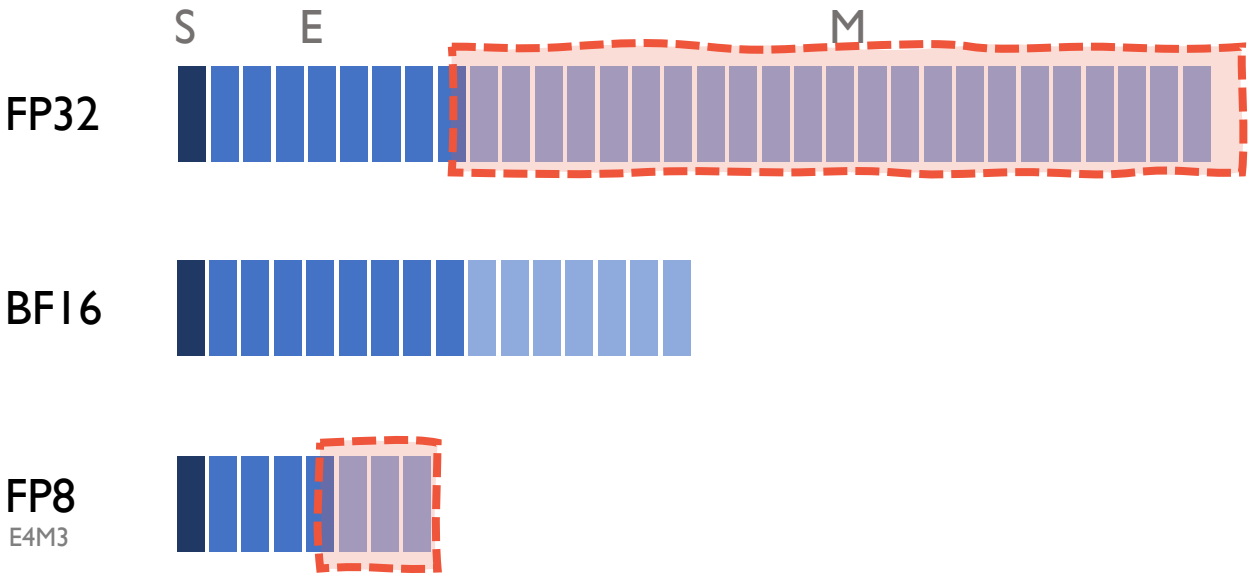
Trend towards **lower precision**



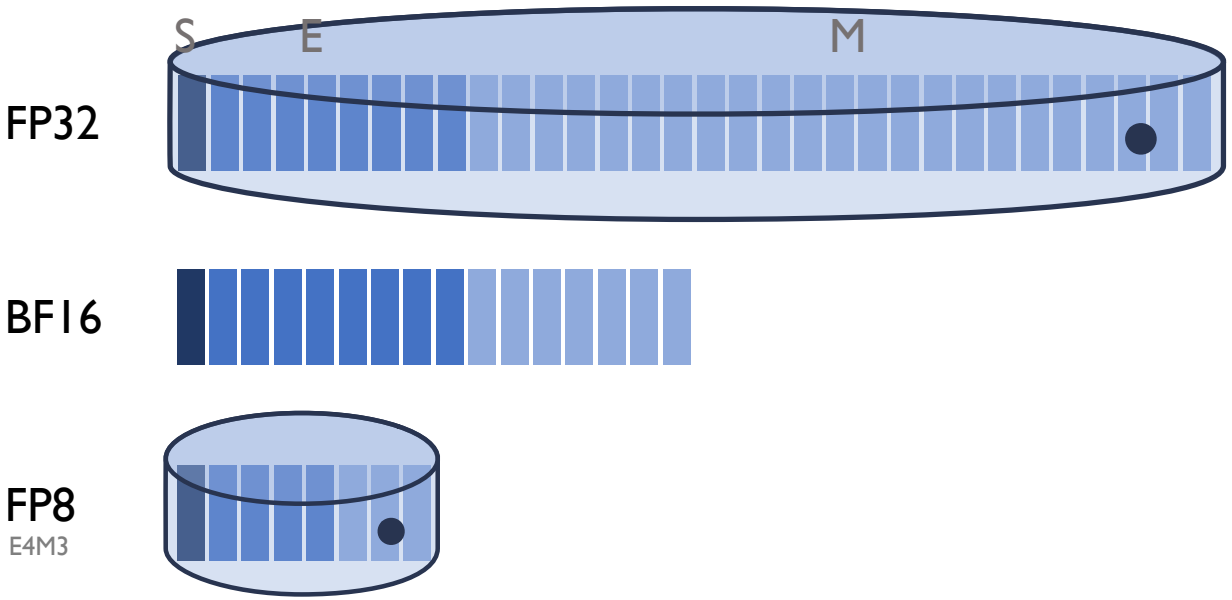
Trend towards **lower precision**



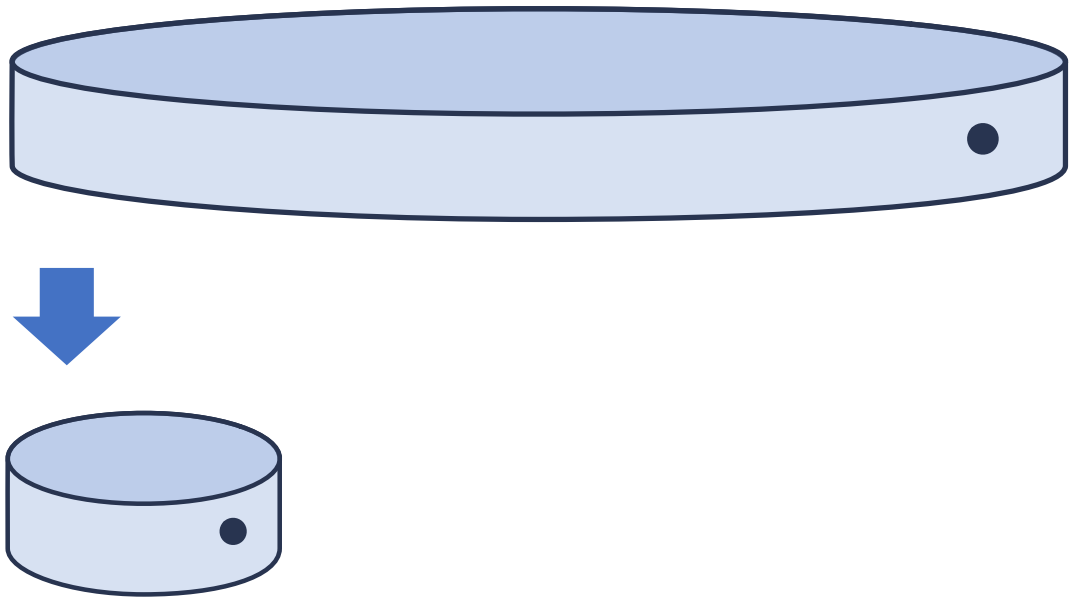
Trend towards **lower precision**



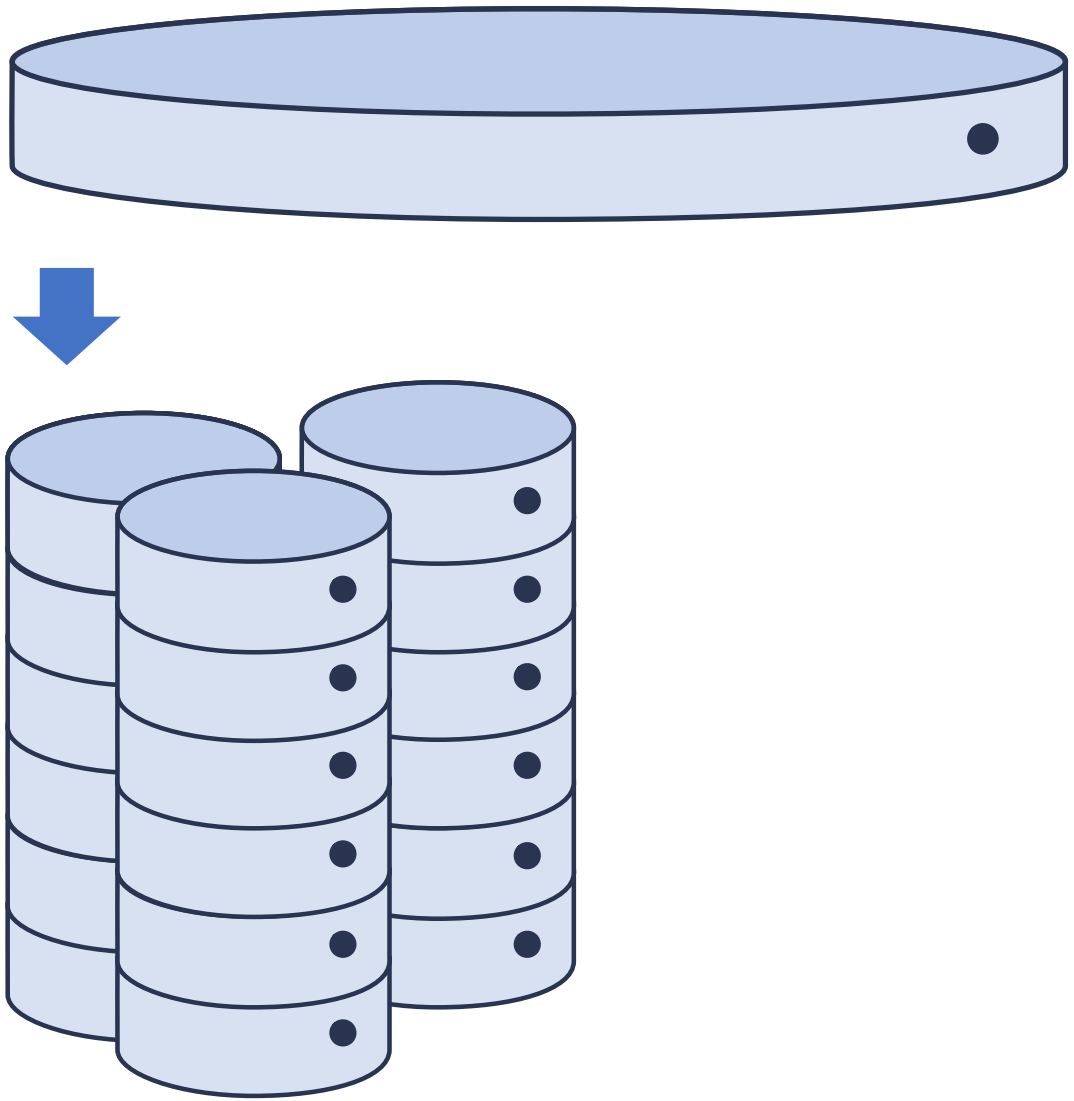
Trend towards **lower precision**



Trend towards **lower precision**



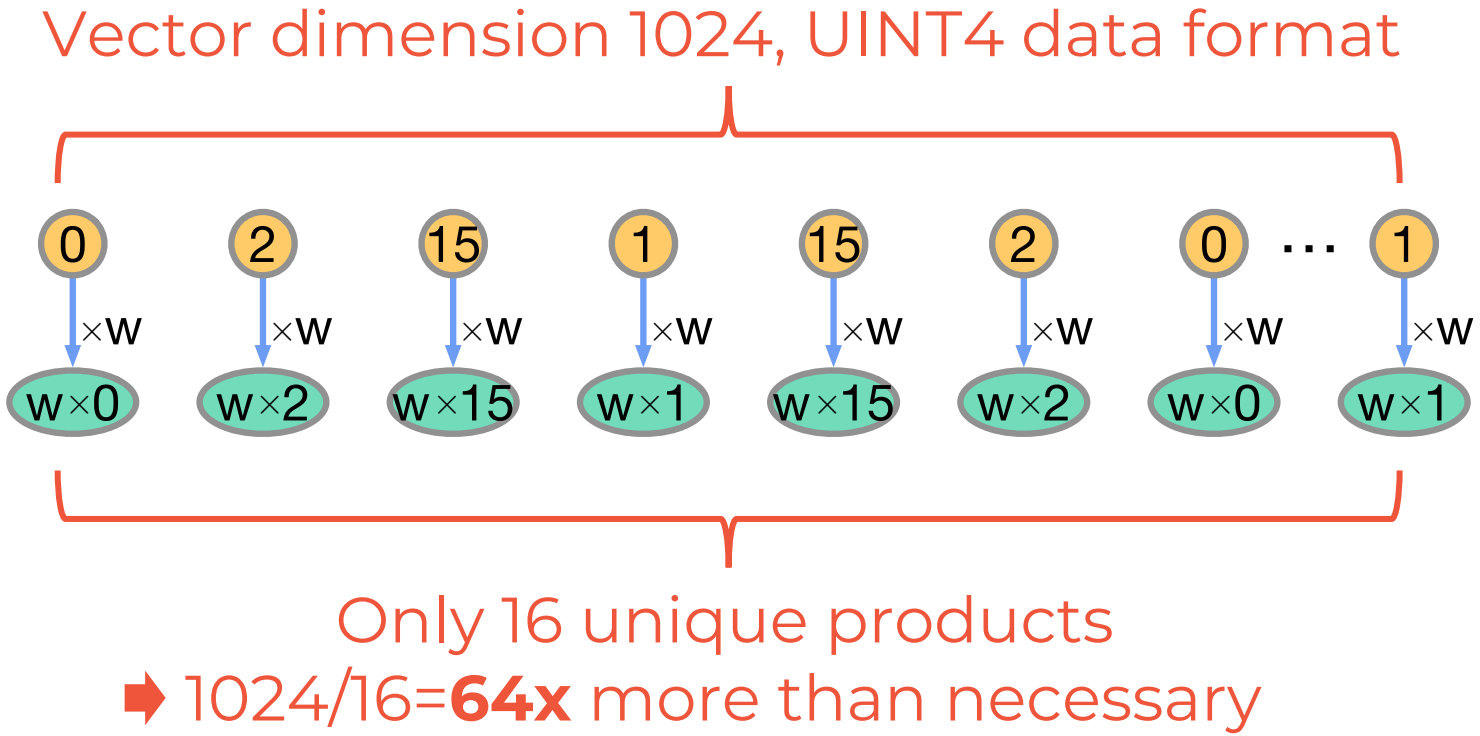
Trend towards **lower precision** and **larger batch size**.



Trend towards **lower precision** and **larger batch size**.

Not all computations are mandatory in existing GEMMs architectures.

Scalar-vector multiplication



Value-Level Parallelism (VLP)

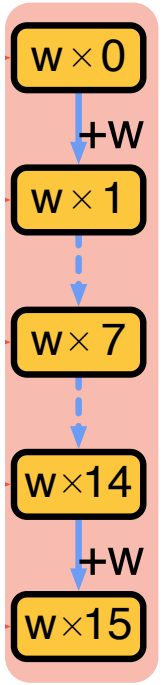
Value-Level Parallelism (VLP)

Concept

Value-Level Parallelism (VLP)

Value Reuse

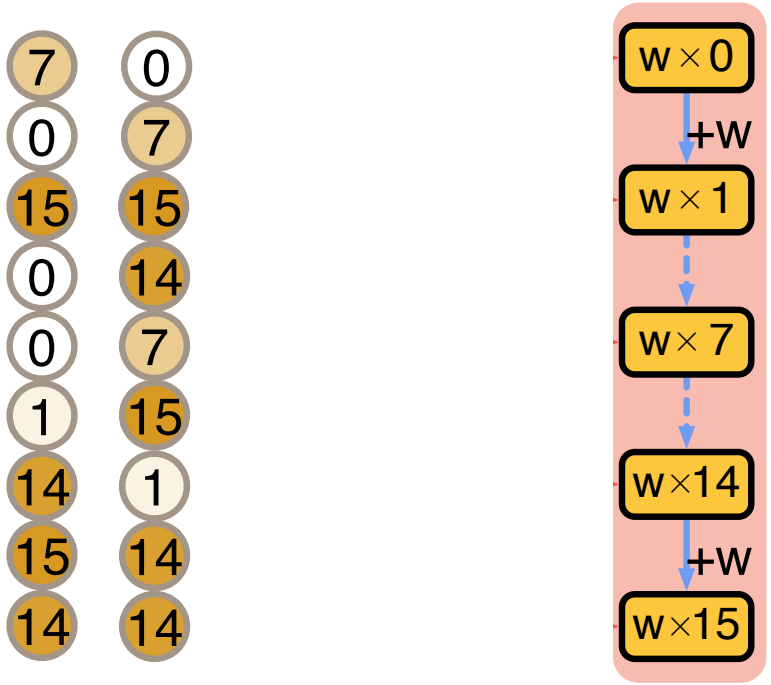
Concept



Value-Level Parallelism (VLP)

Value Reuse

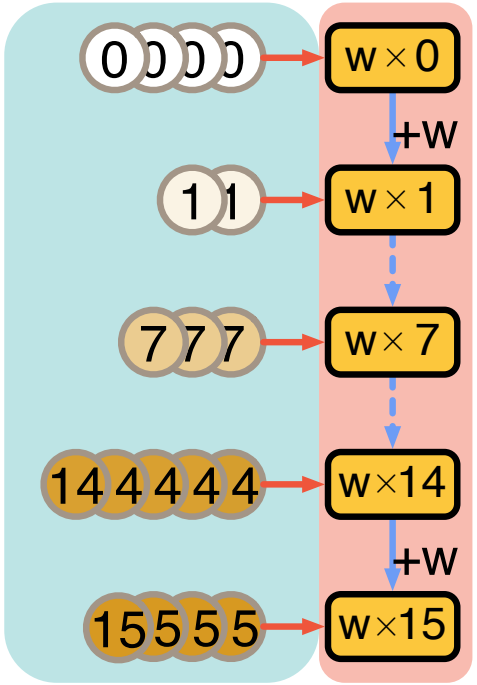
Concept



Value-Level Parallelism (VLP)

Value Reuse + **Subscription**

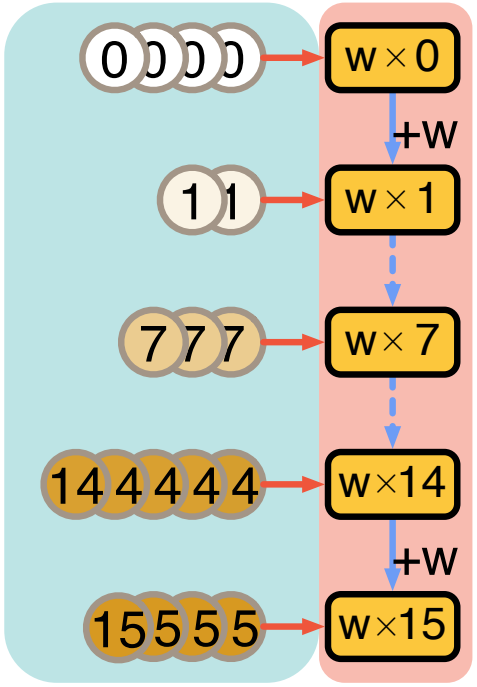
Concept



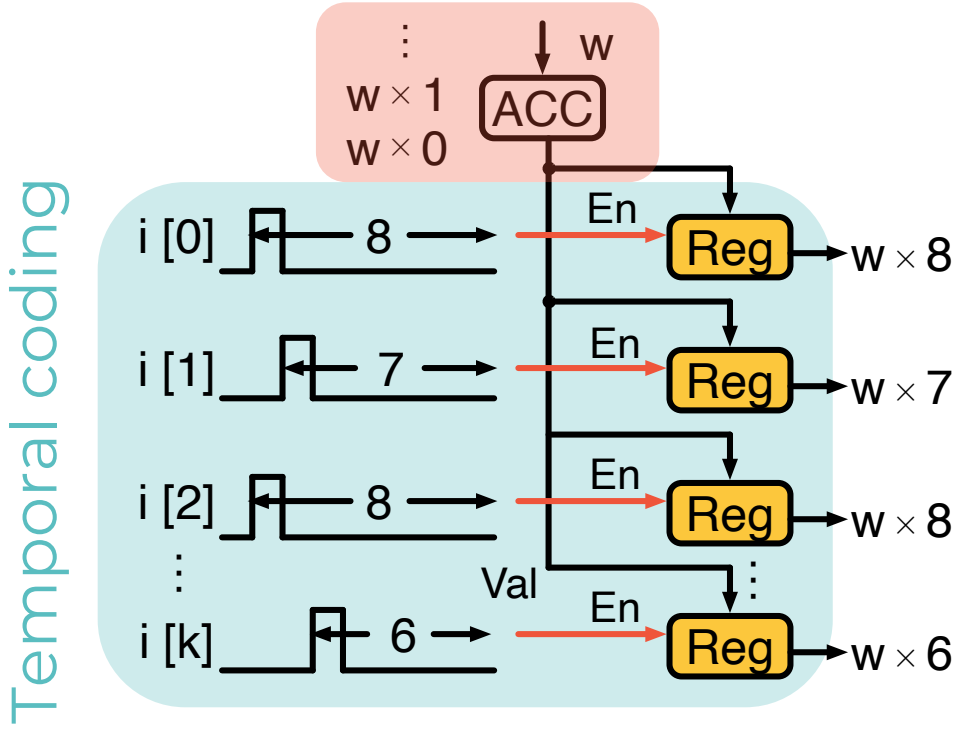
Value-Level Parallelism (VLP)

Value Reuse + **Subscription**

Concept

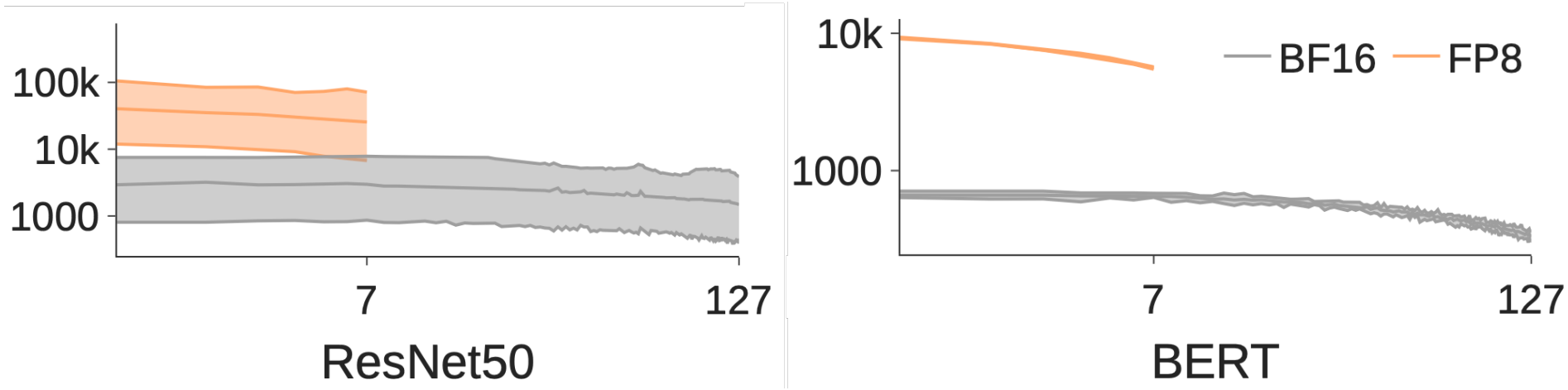


Architecture

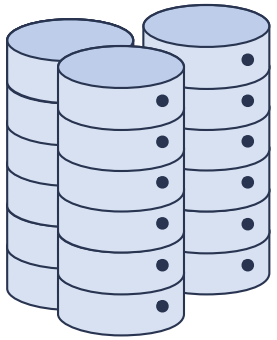


Value Reuse Opportunities

Y-axis: Number of inputs
X-axis: Unique mantissa value



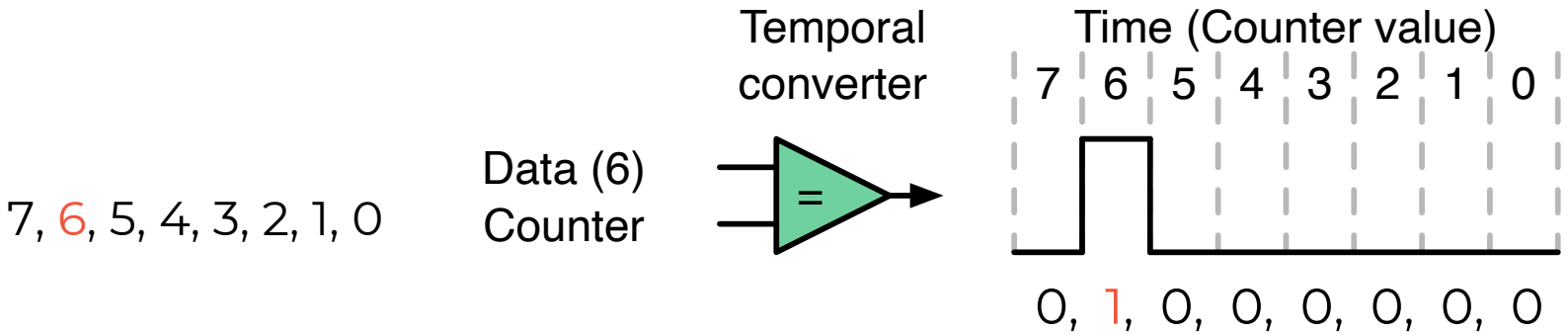
Carat



Value-Level Parallelism

Temporal Coding 101

Timing of a spike represents data value.

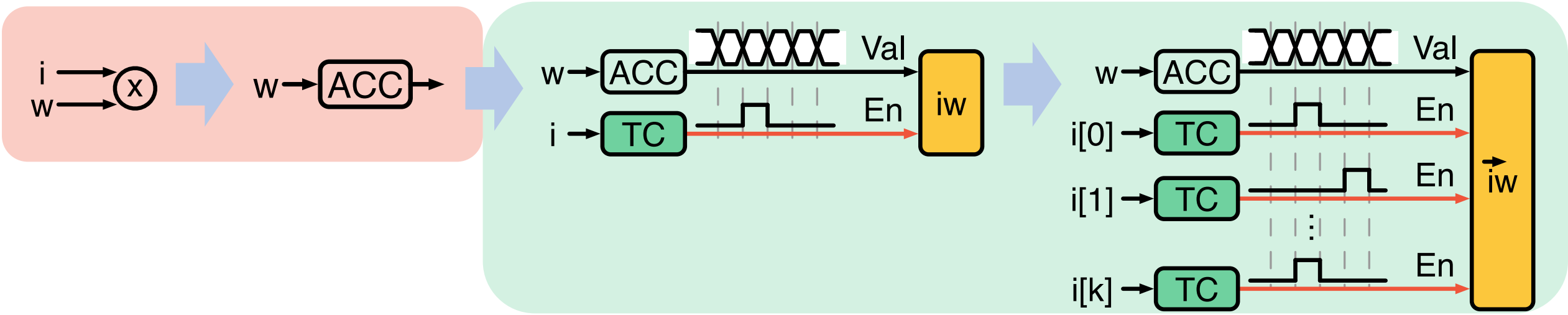


Value reuse + subscription

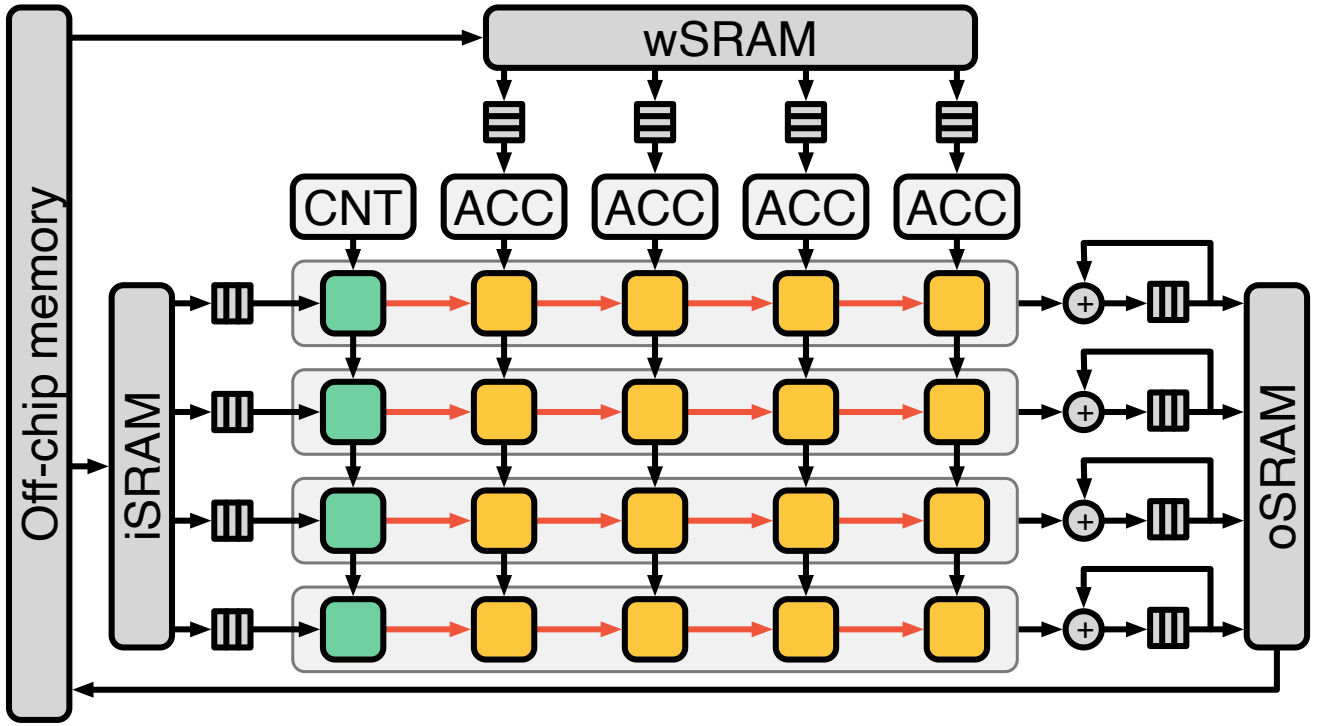
Transforms a multiplication into accumulations

Temporal input subscribes to product register

Parallel independent input subscription



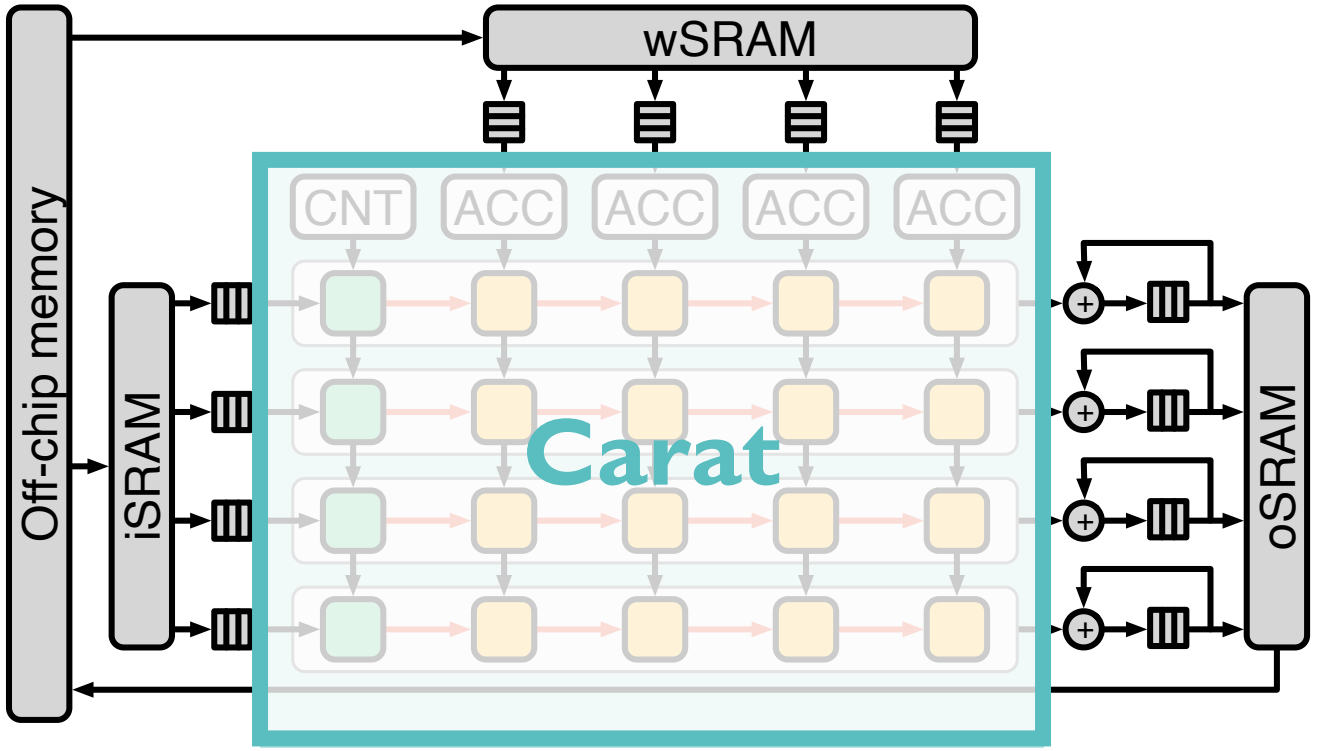
Overview



Overview

Carat Array

Computes outer product of input and weight vectors



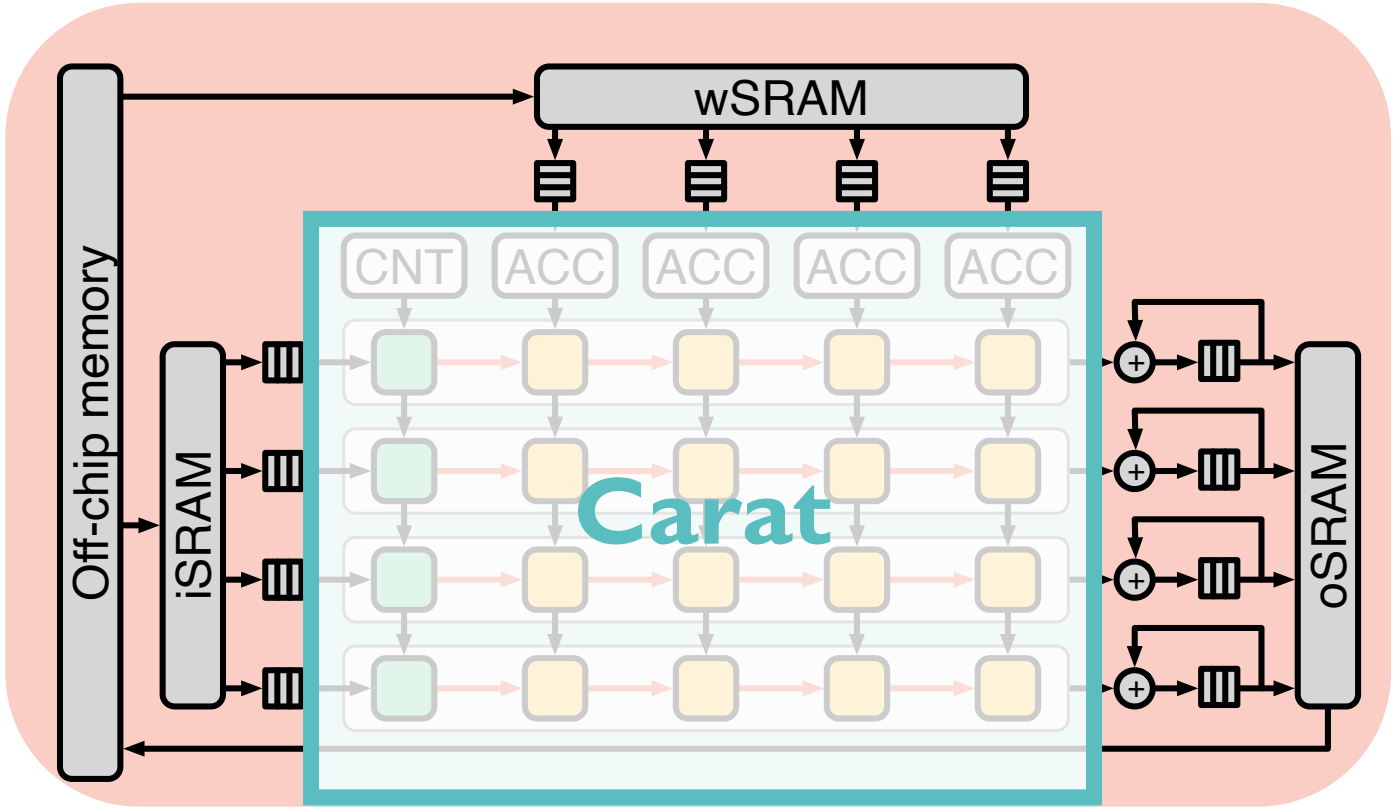
Overview

Carat Array

Computes outer product of input and weight vectors

Memory hierarchy and peripheral logic

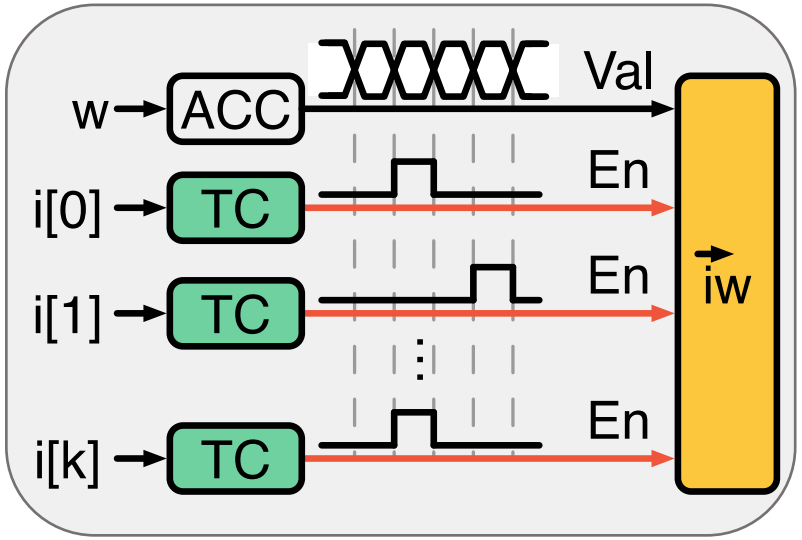
Identical to existing GEMMs accelerators



Processing Element (PE) Array

PE Column

Scalar-vector multiplication

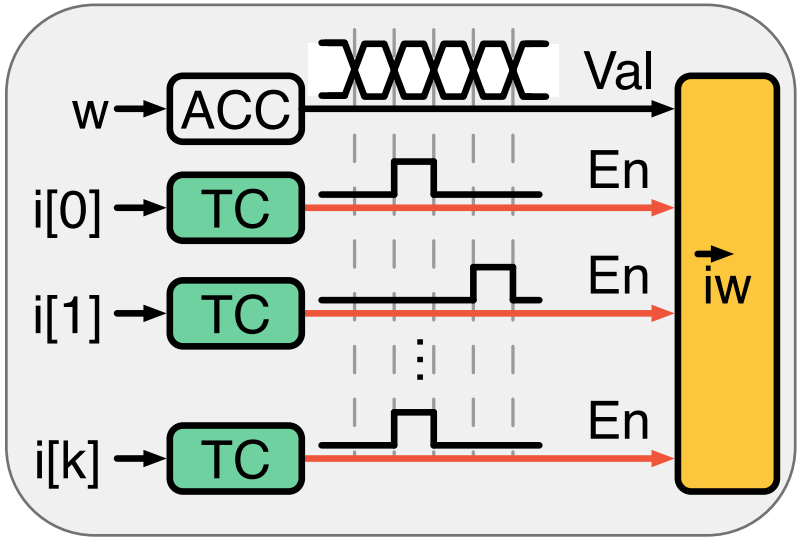


Processing Element (PE) Array

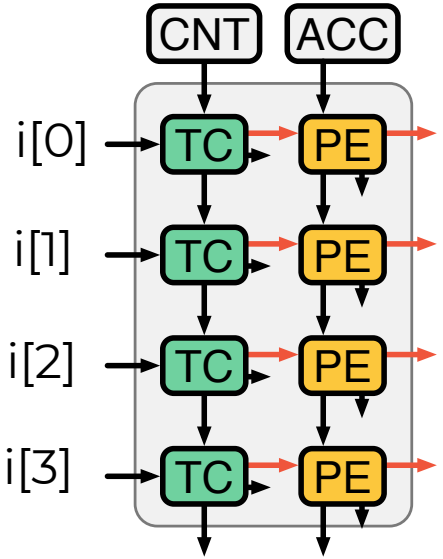
PE Column

Scalar-vector multiplication

Temporal Converter (TC)



PE

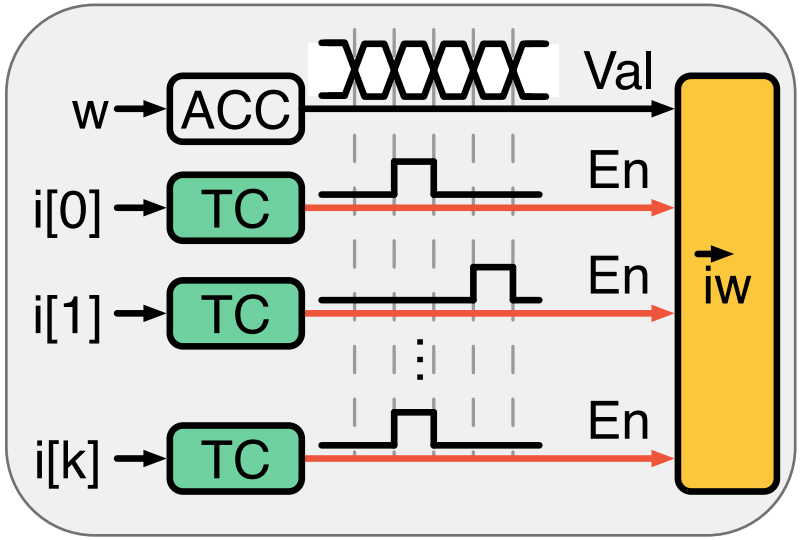


Processing Element (PE) Array

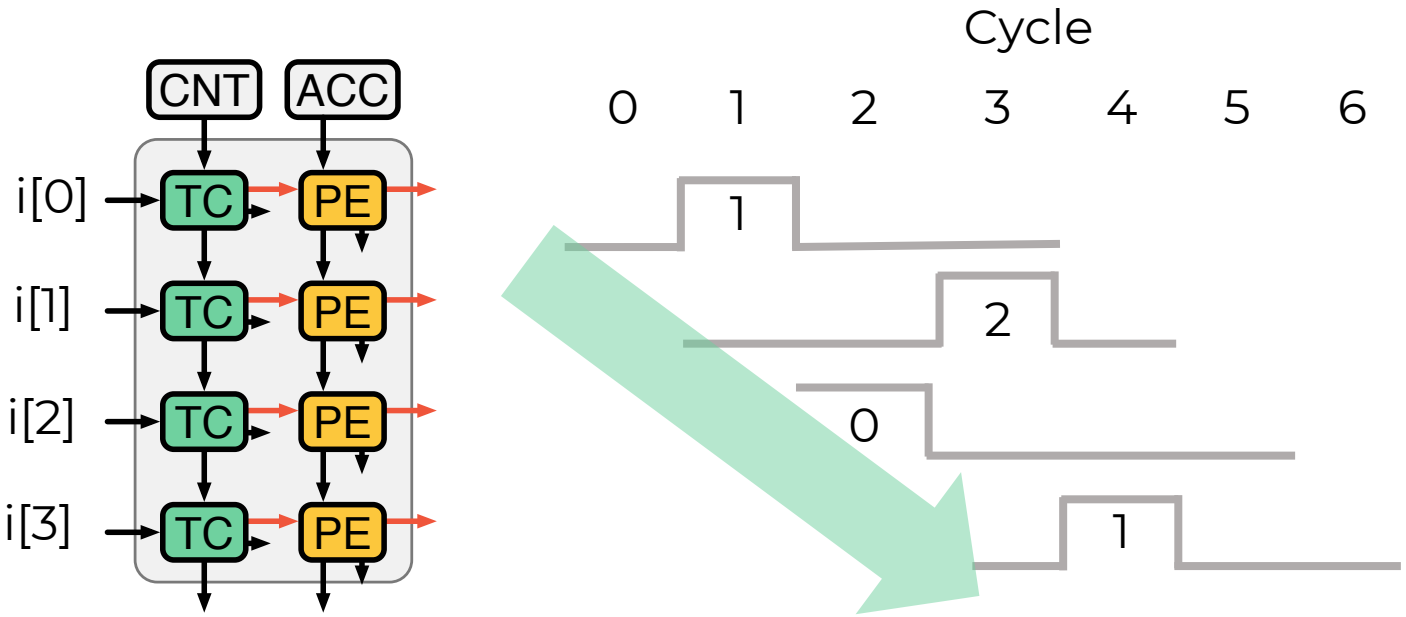
PE Column

Scalar-vector multiplication

Temporal Converter (TC)



PE



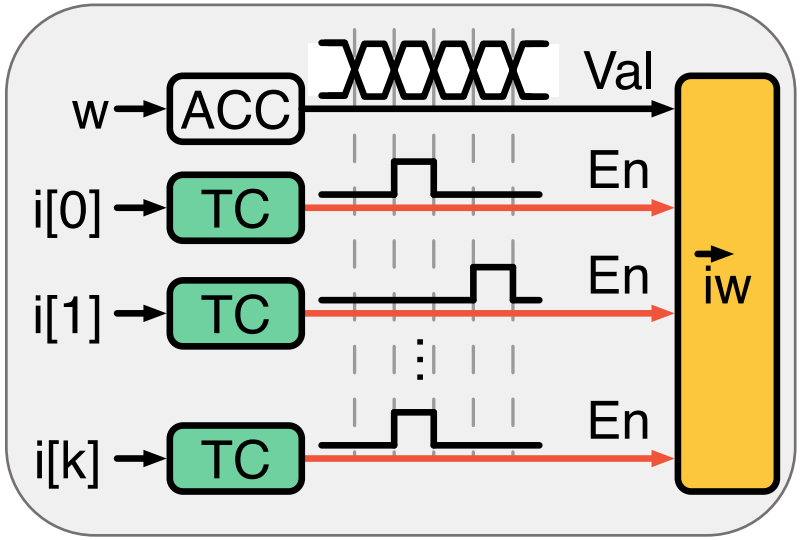
Processing Element (PE) Array

PE Column

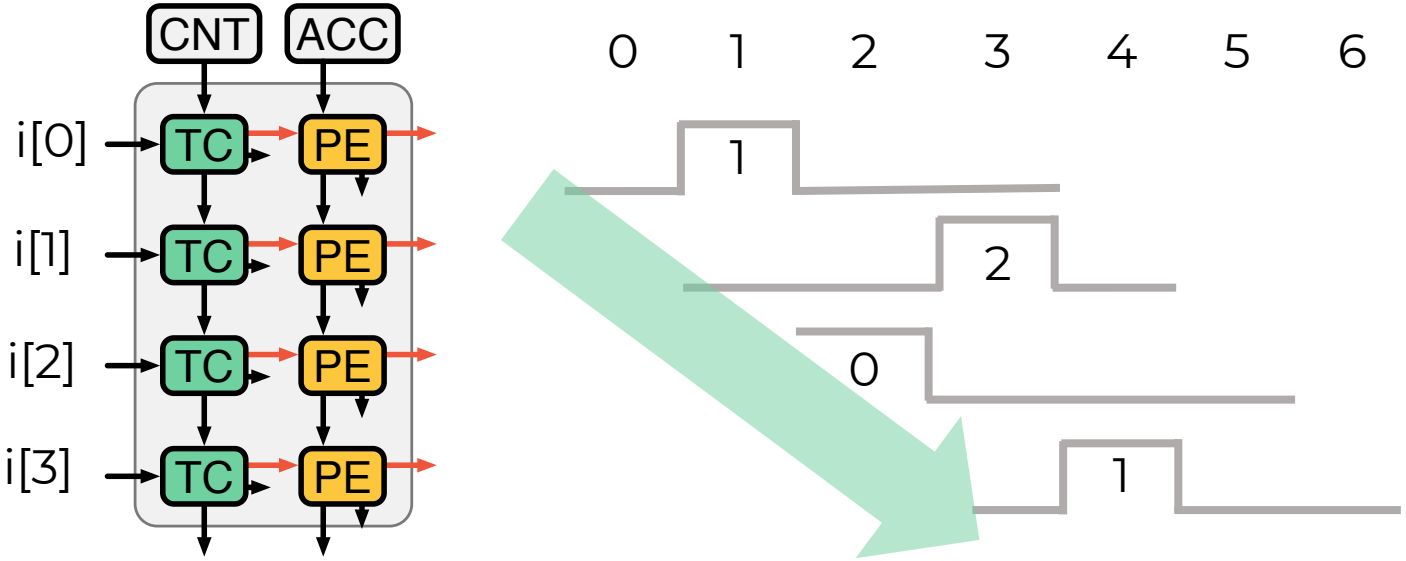
Scalar-vector multiplication

Temporal Converter (TC)

Convert **mantissa** bits to temporal coding



PE



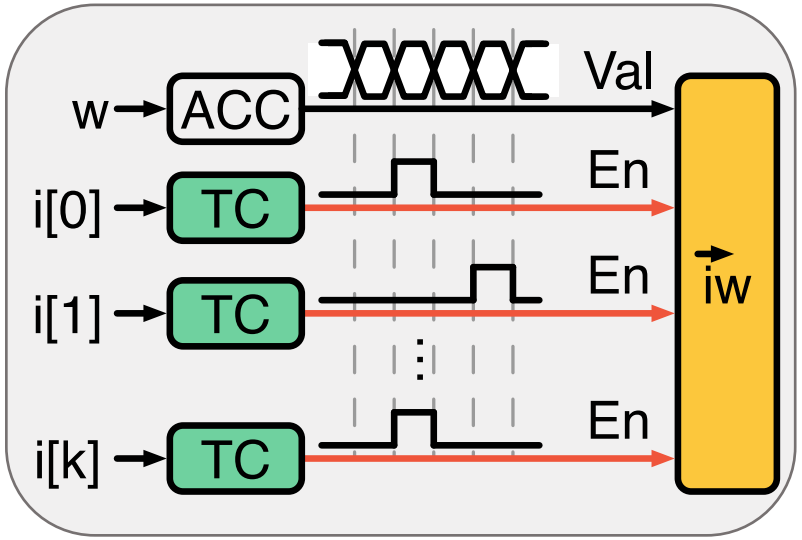
Processing Element (PE) Array

PE Column

Scalar-vector multiplication

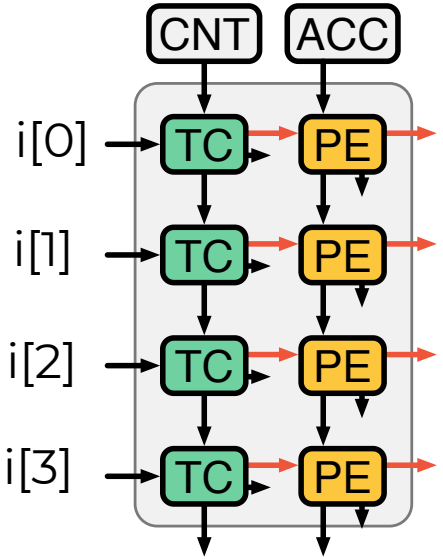
Temporal Converter (TC)

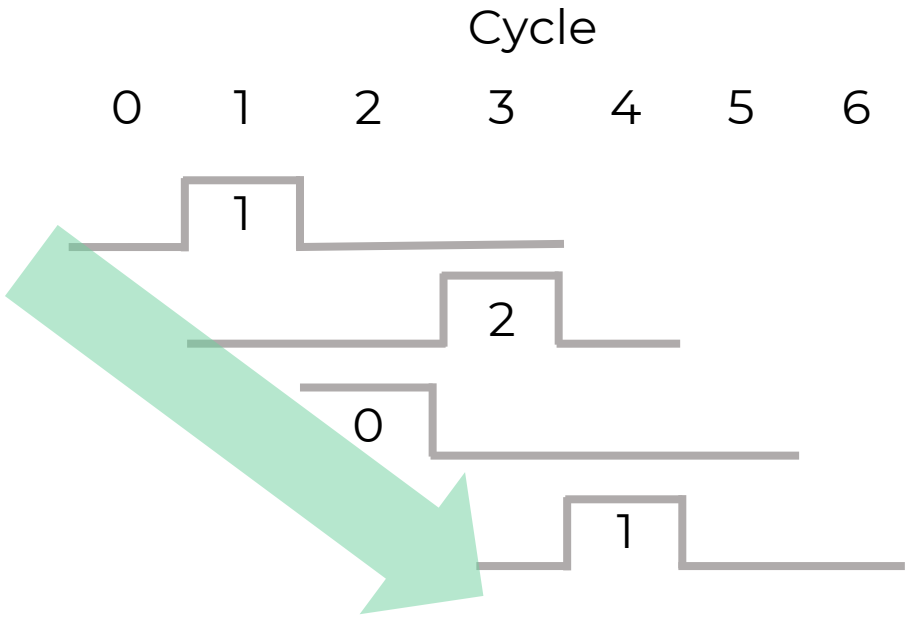
Convert **mantissa** bits to temporal coding

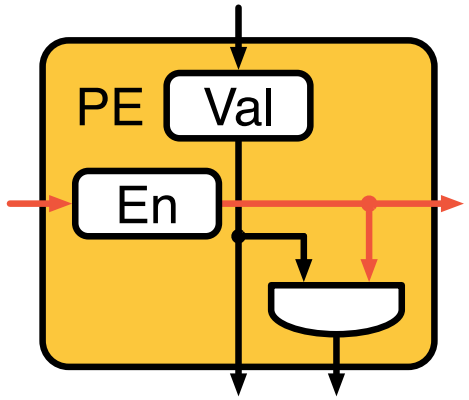


PE

Input subscription

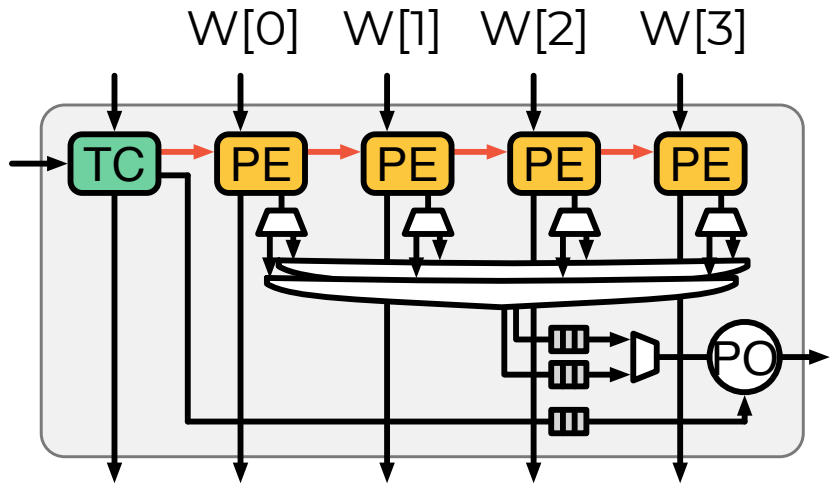






Processing Element (PE) Array

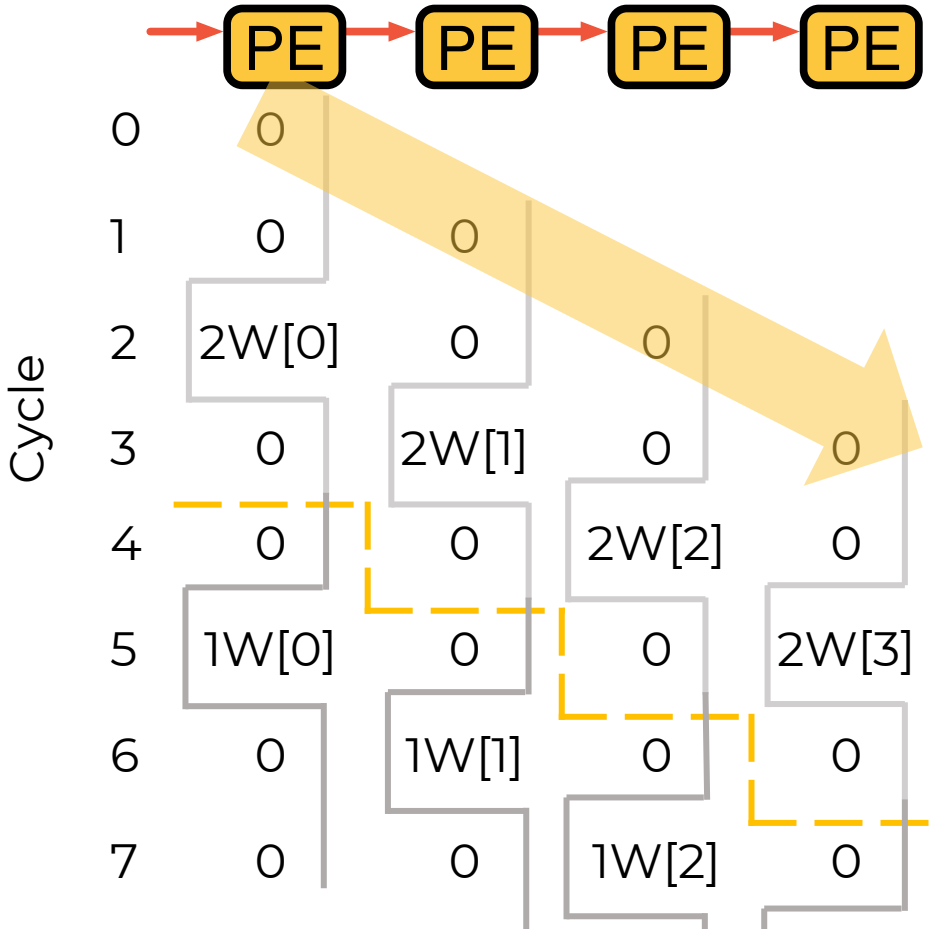
PE Row

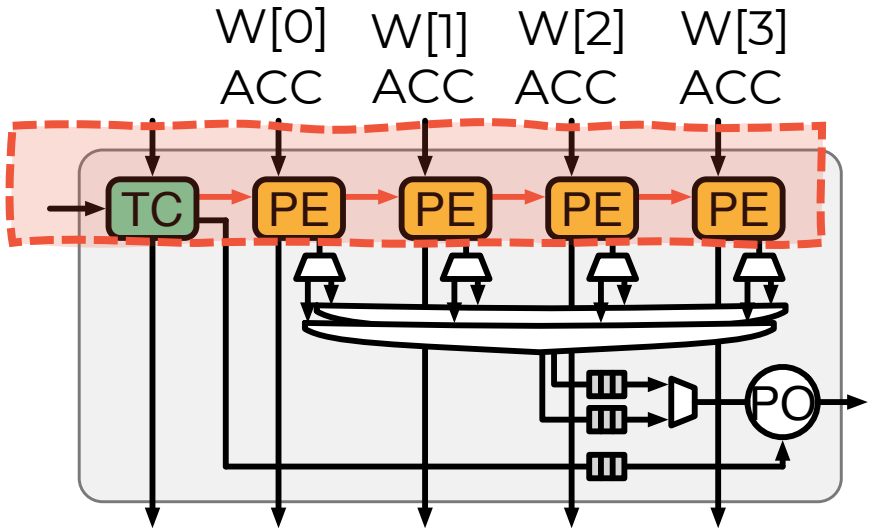


Processing Element (PE) Array

PE Row

Pipelining



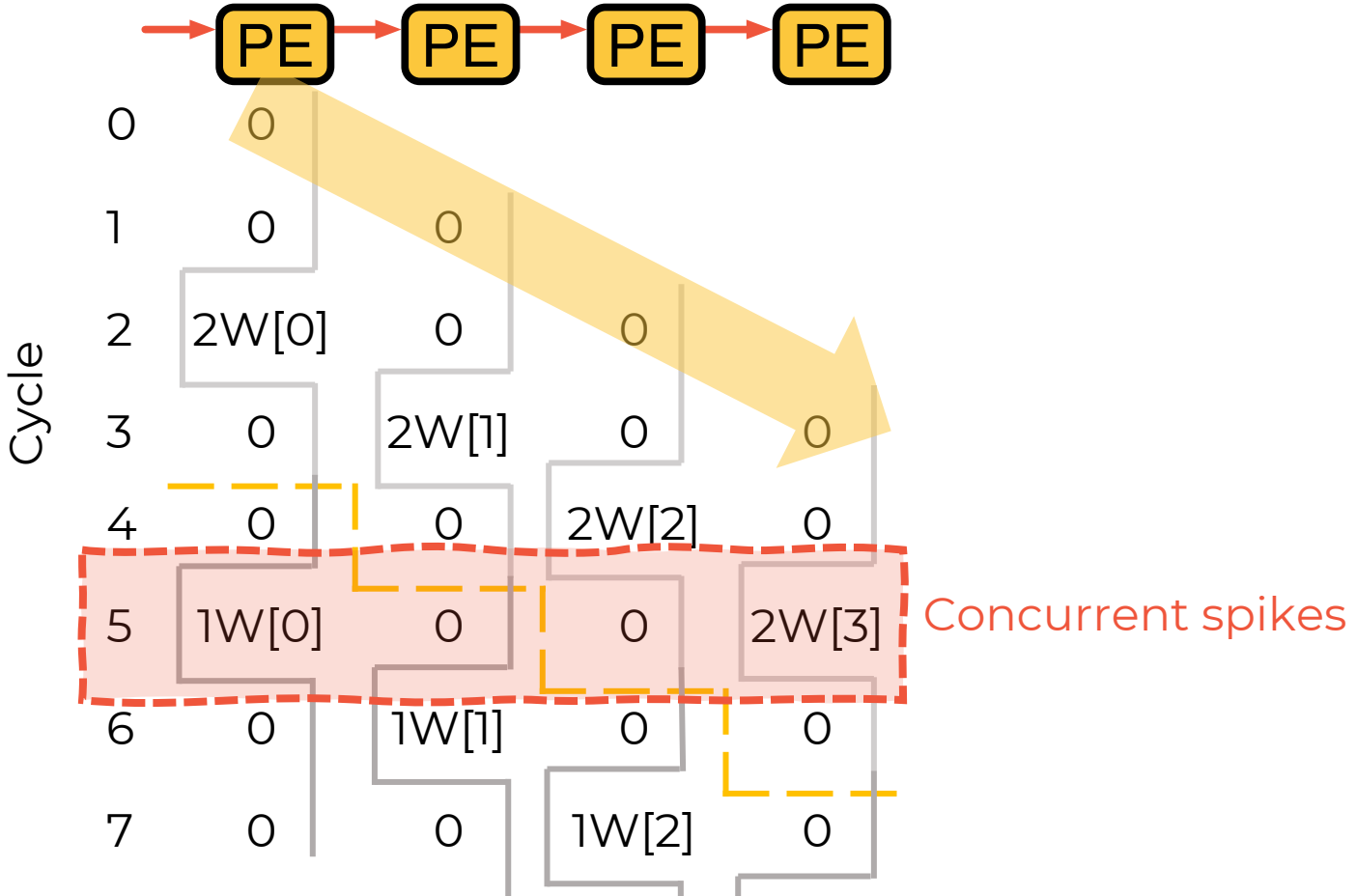


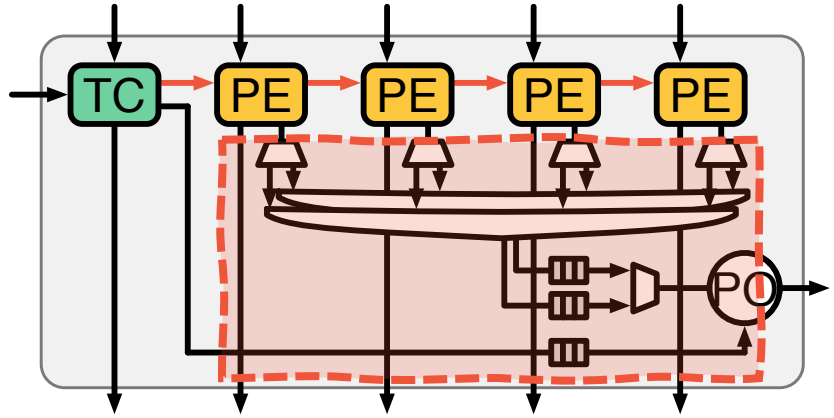
Processing Element (PE) Array

PE Row

Pipelining

Concurrent spike multiplexing





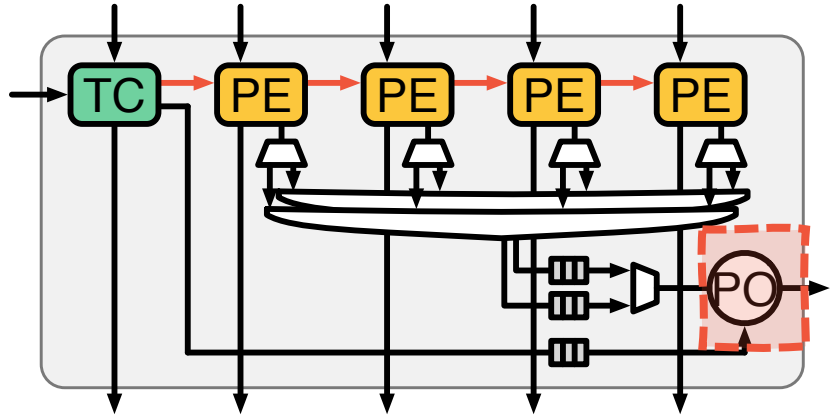
Processing Element (PE) Array

PE Row

Pipelining

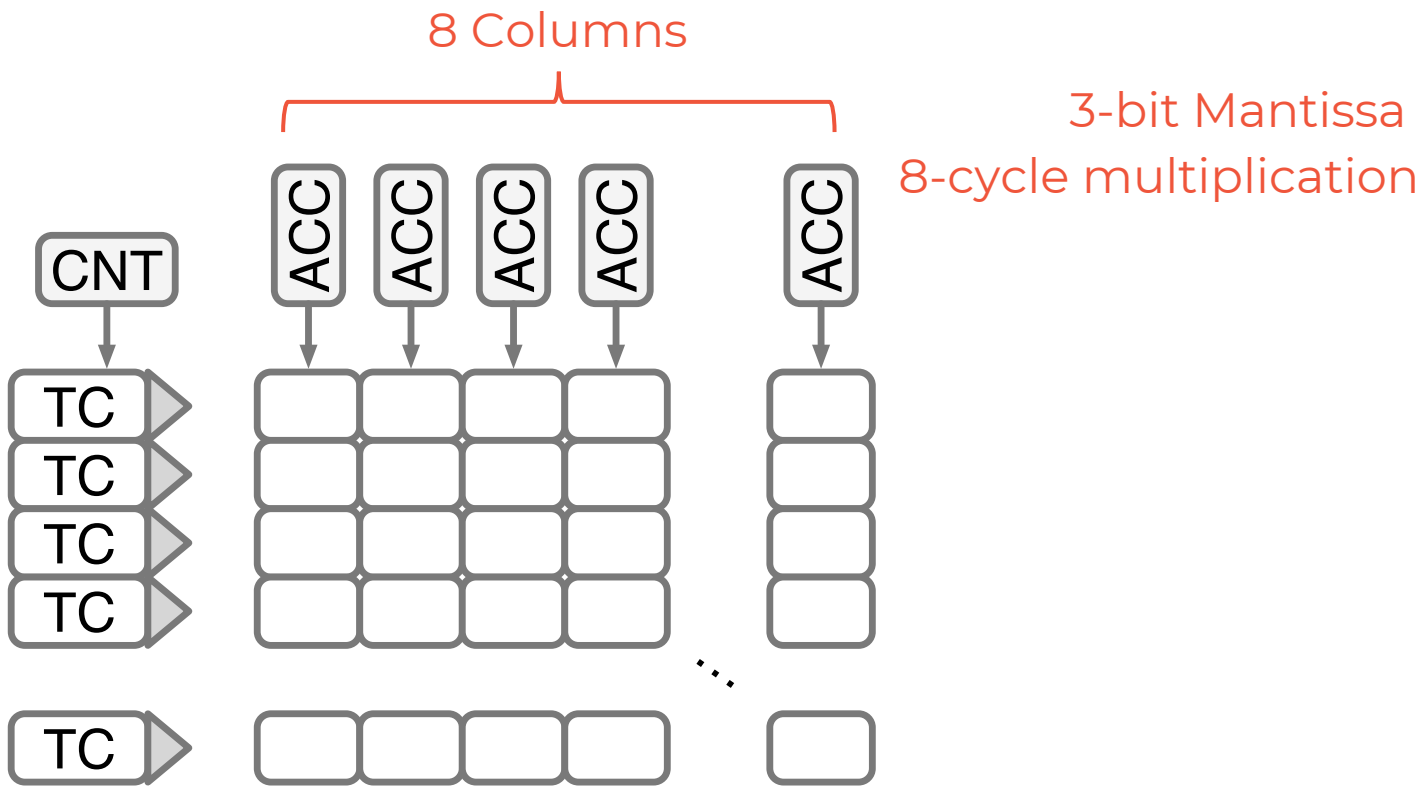
Concurrent spike multiplexing

Special value post-processing



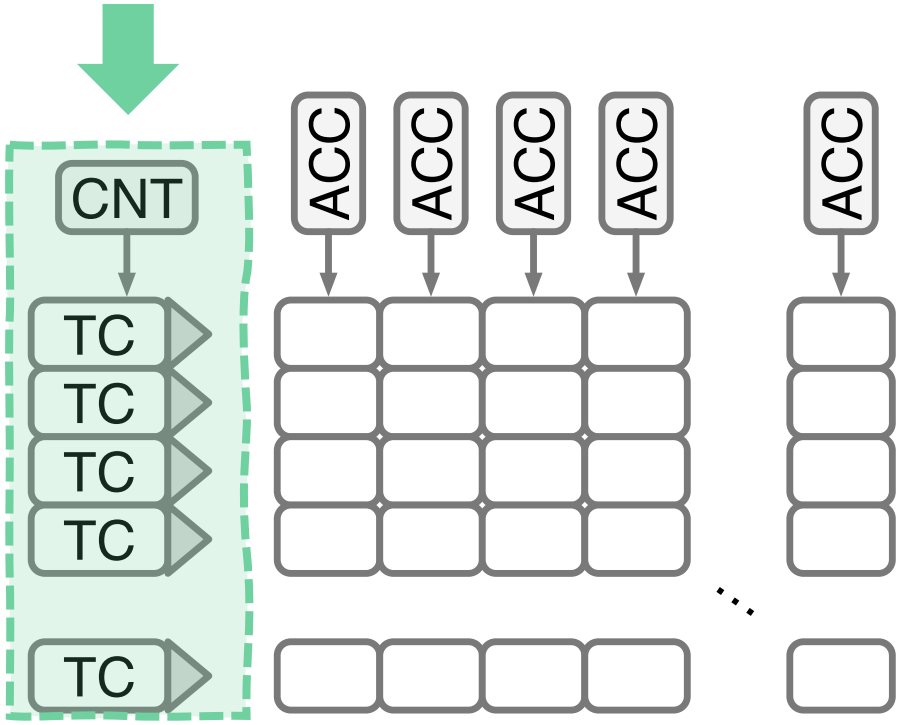
Processing Element (PE) Array

PE Array



Processing Element (PE) Array

PE Array

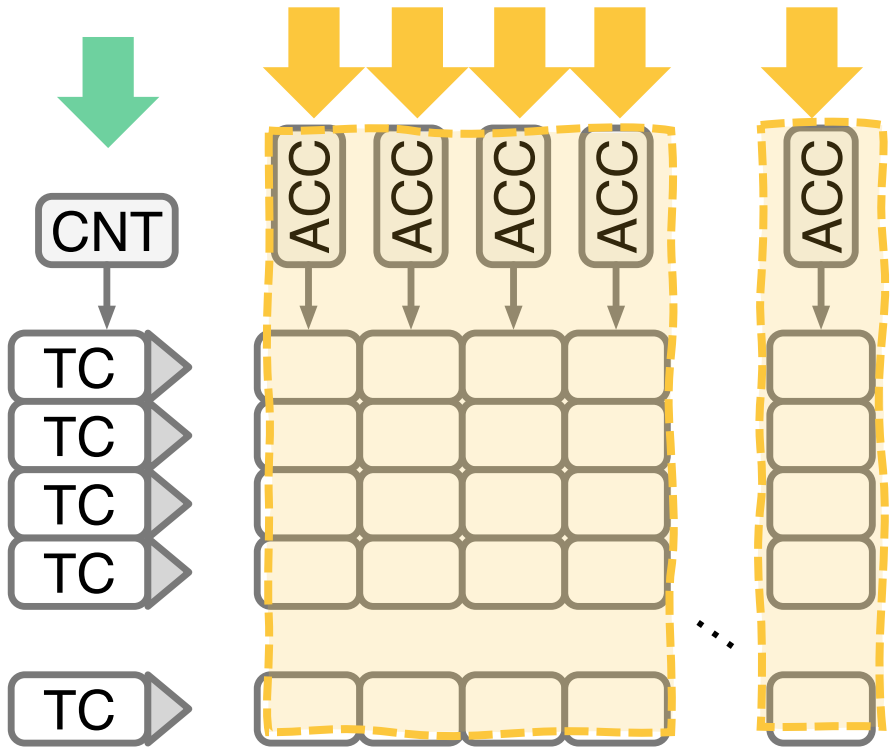


TC Column

Propagates counter value

Processing Element (PE) Array

PE Array



TC Column

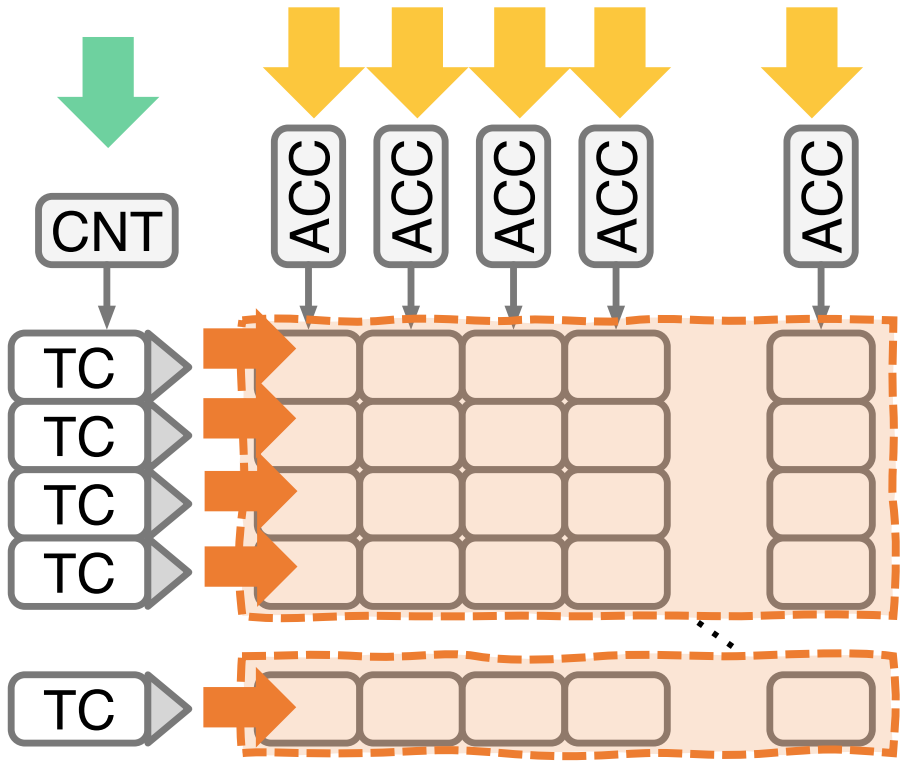
Propagates counter value

PE Column

Propagates accumulator value

Processing Element (PE) Array

PE Array



TC Column

Propagates counter value

PE Column

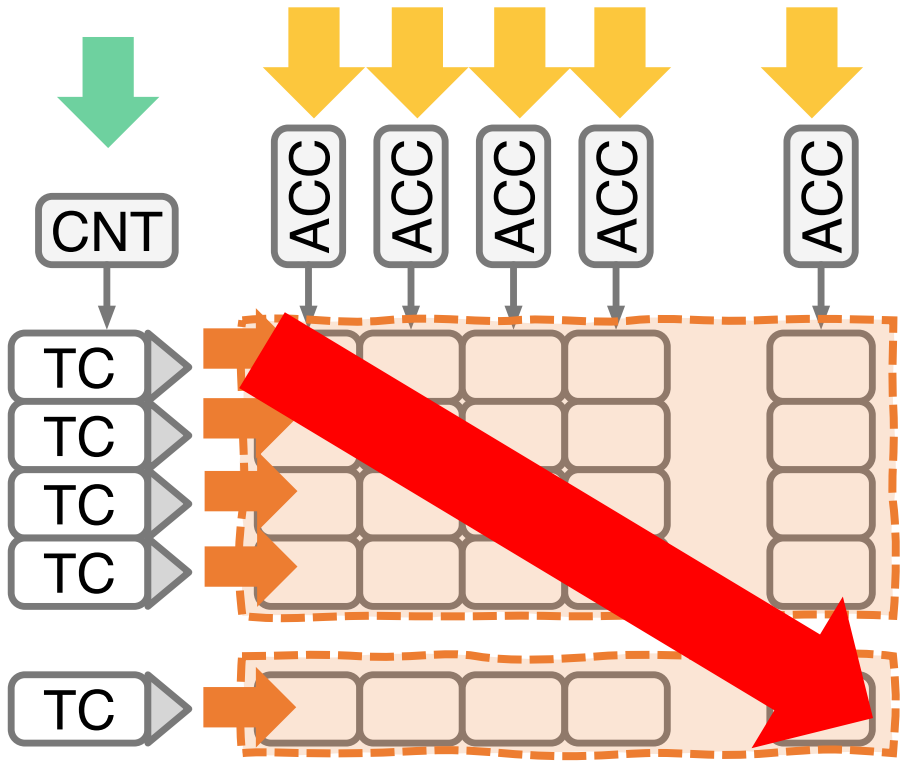
Propagates accumulator value

PE Row

Propagates temporal signal

Processing Element (PE) Array

PE Array



TC Column

Propagates counter value

PE Column

Propagates accumulator value

PE Row

Propagates temporal signal

More in the paper

optimization
- ❑ Accumulation fast-forwarding
- ❑ Counter sharing

FP handling
- ❑ Subnormal adjustment
- ❑ Product masking
- ❑ Exponent expansion

- ❑ Walkthrough timing example

Evaluation

Setup

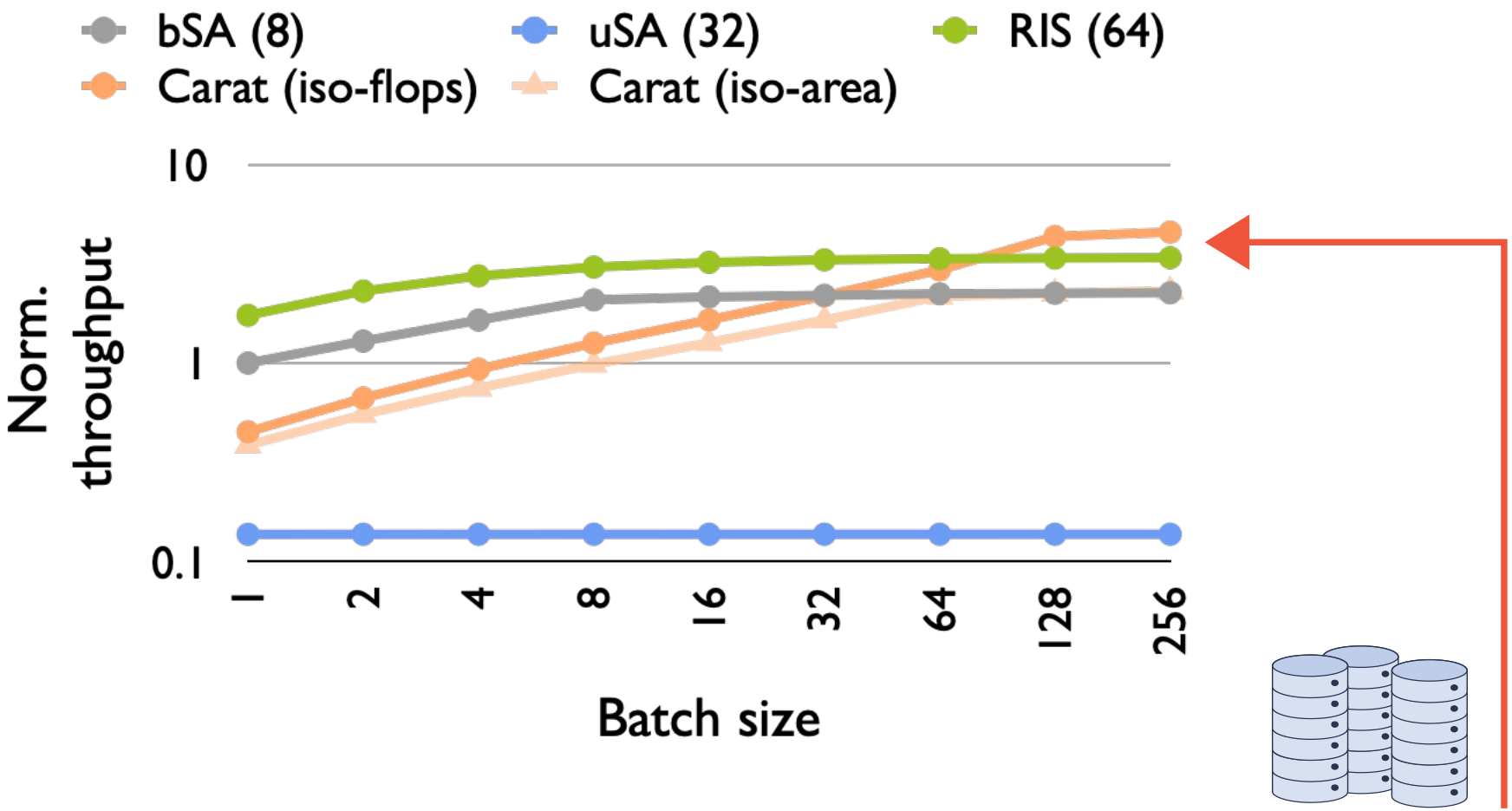
Workload

MLPerf benchmark suite

Baseline

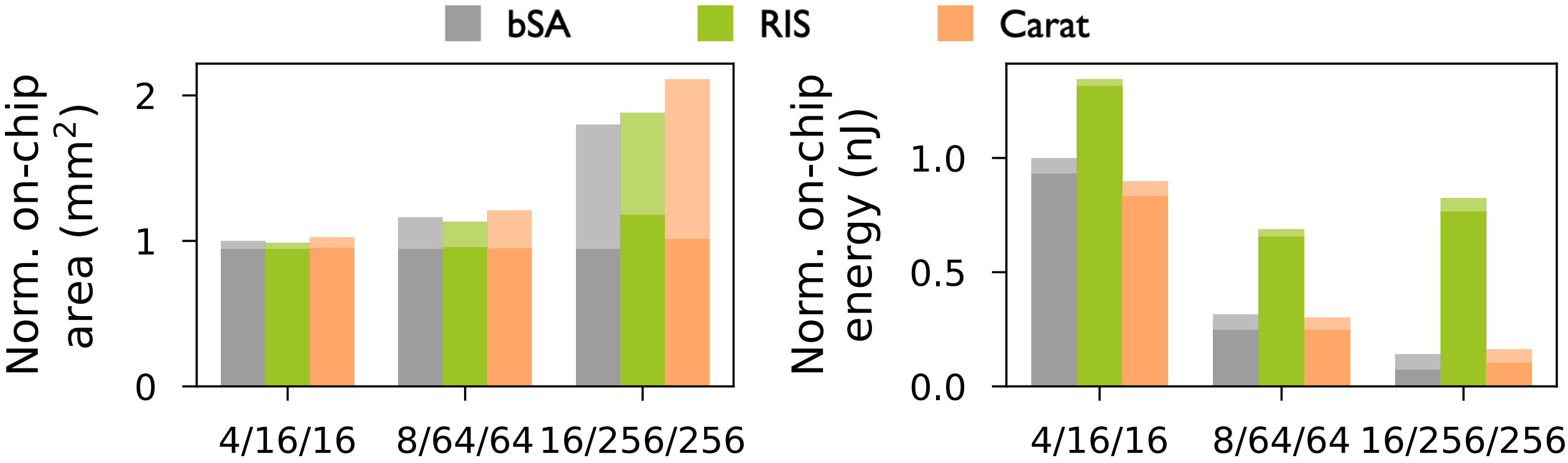
- ❑ Binary systolic array (bSA)
- ❑ Unary systolic array (uSA)
- ❑ Compute reuse-based accelerator (RIS)

Throughput vs Batch Size



Large batch size exposes richer VLP opportunities

Area, energy (iso-area)



Carat: Unlocking Value-Level Parallelism for Multiplier-Free GEMMs

Zhewen Pan Joshua San Miguel Di Wu

More in the paper

optimization

- ❑ Accumulation fast-forwarding
- ❑ Counter sharing

FP handling

- ❑ Subnormal adjustment
- ❑ Product masking
- ❑ Exponent expansion

- ❑ Walkthrough timing example

Start from 8xW

Concurrent generation

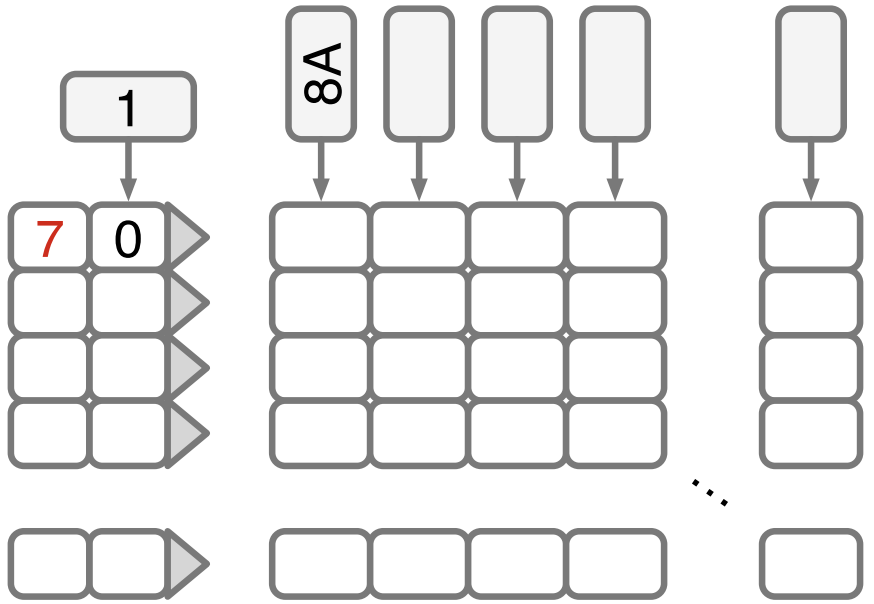
Subnormal inputs

Zero or NAN inputs

Accumulator overflow

Processing Element (PE) Organization

Cycle 0



TC Column

Propagates counter value

PE Column

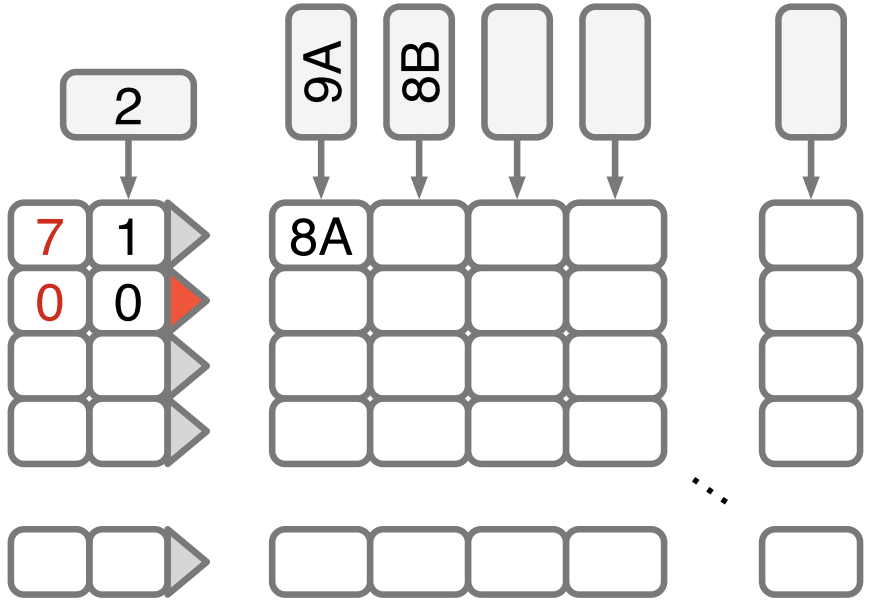
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 1



TC Column

Propagates counter value

PE Column

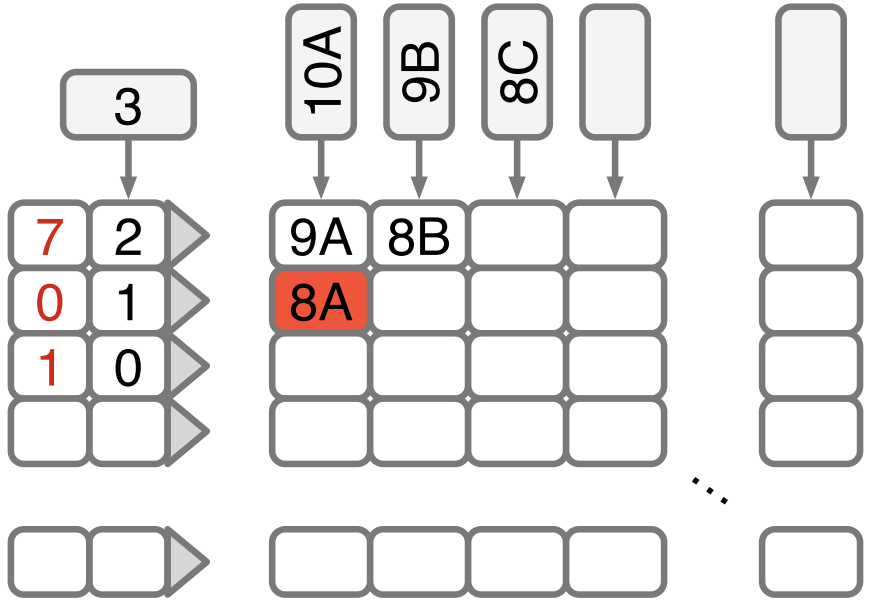
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 2



TC Column

Propagates counter value

PE Column

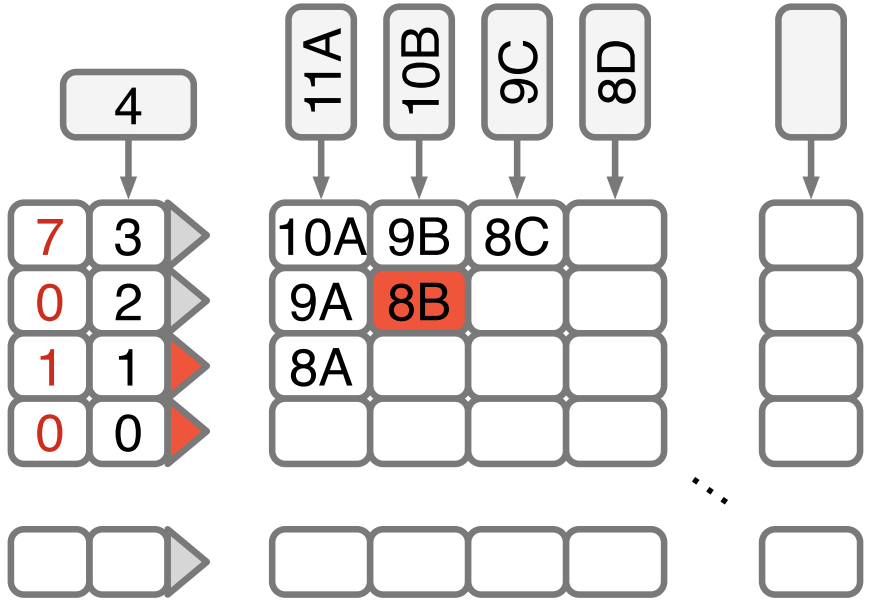
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 3



TC Column

Propagates counter value

PE Column

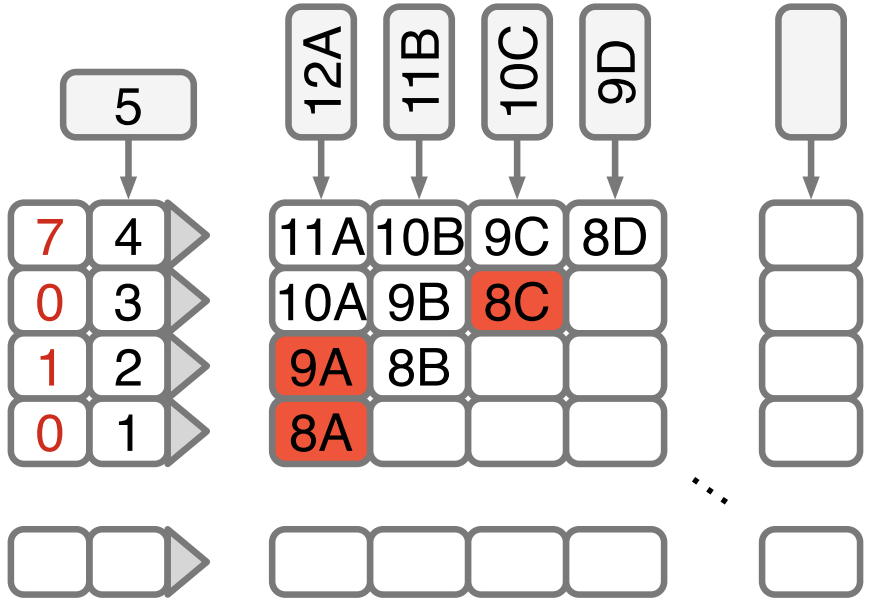
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 4



TC Column

Propagates counter value

PE Column

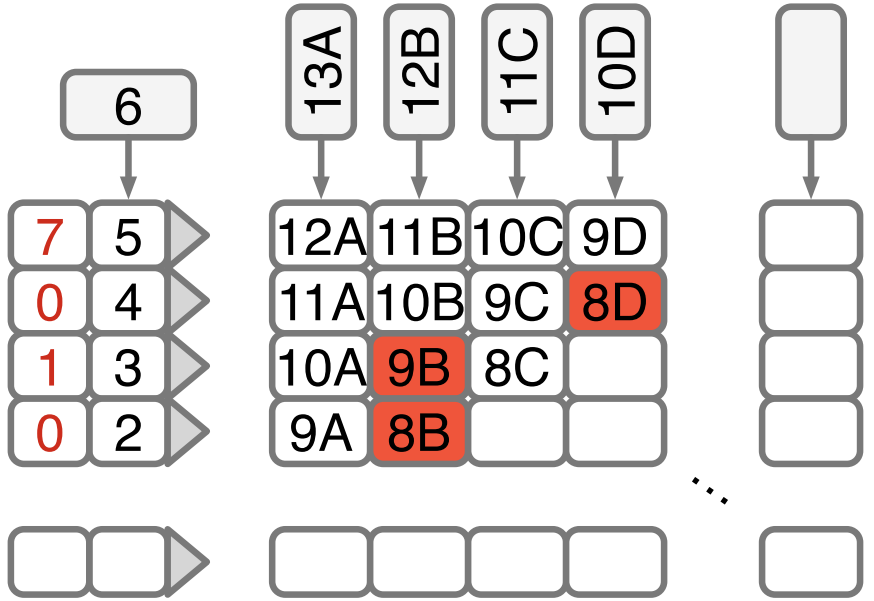
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 5



TC Column

Propagates counter value

PE Column

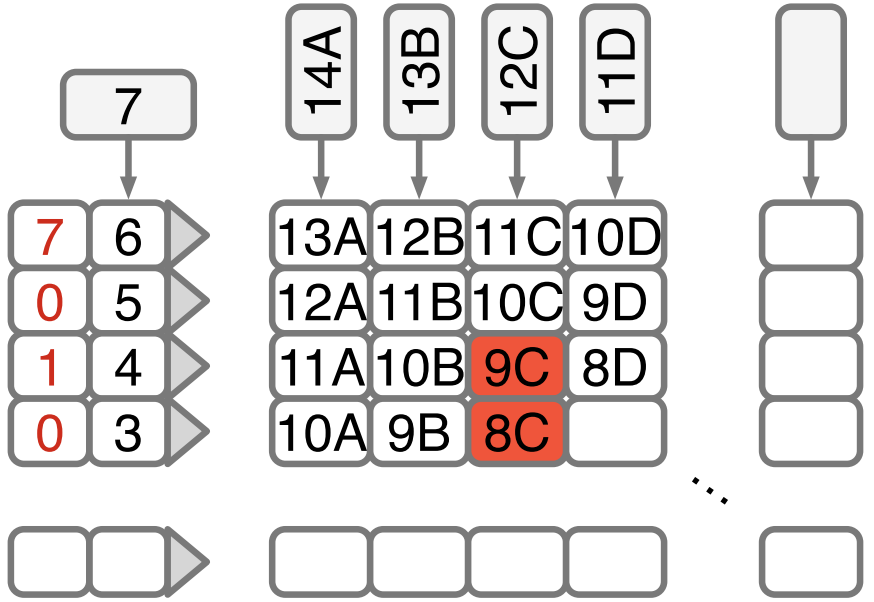
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 6



TC Column

Propagates counter value

PE Column

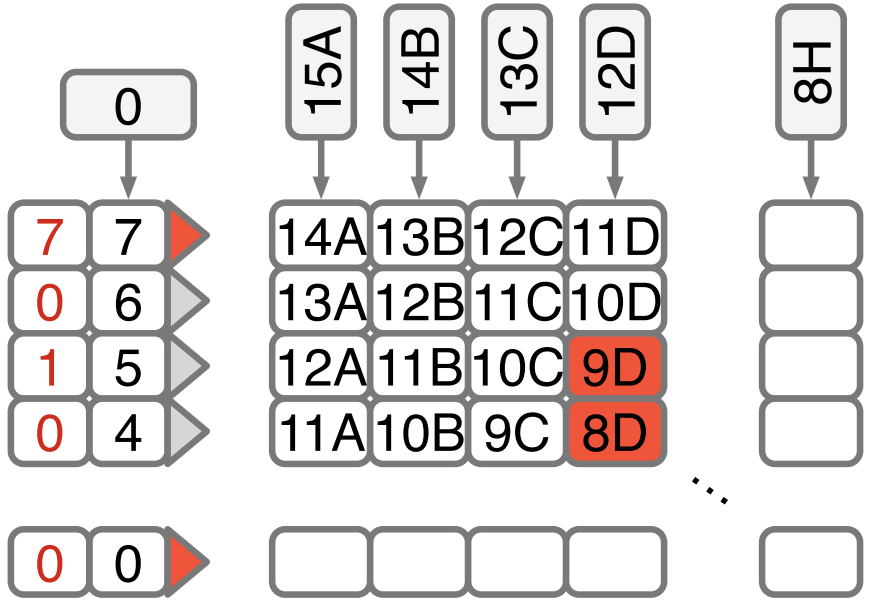
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 7



TC Column

Propagates counter value

PE Column

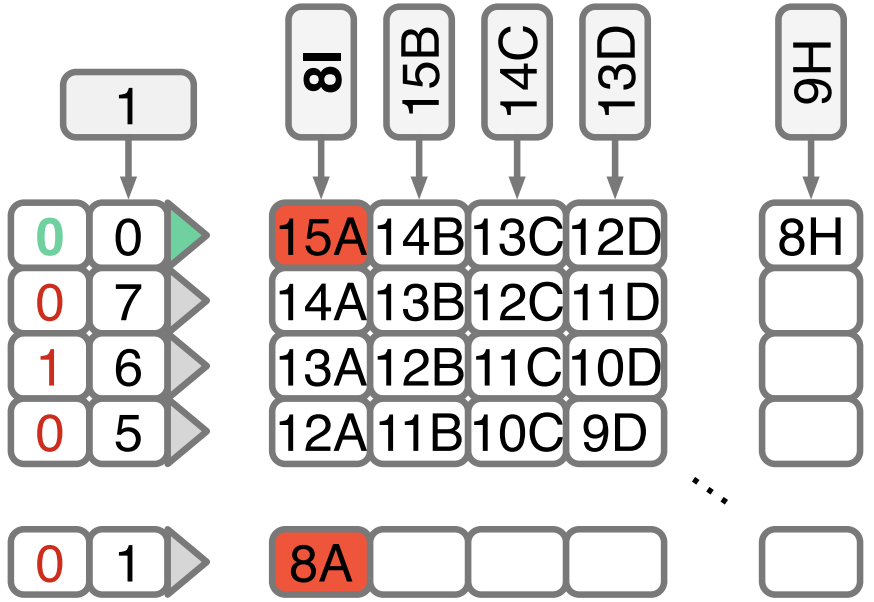
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 8



TC Column

Propagates counter value

PE Column

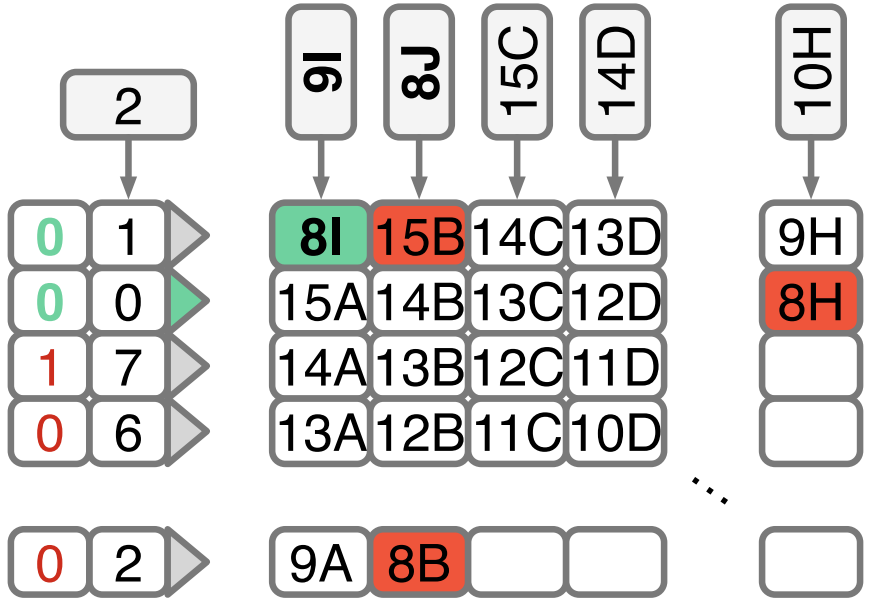
Propagates accumulator value

PE Row

Propagates temporal signal

Walkthrough Timing Example

Cycle 9



TC Column

Propagates counter value

PE Column

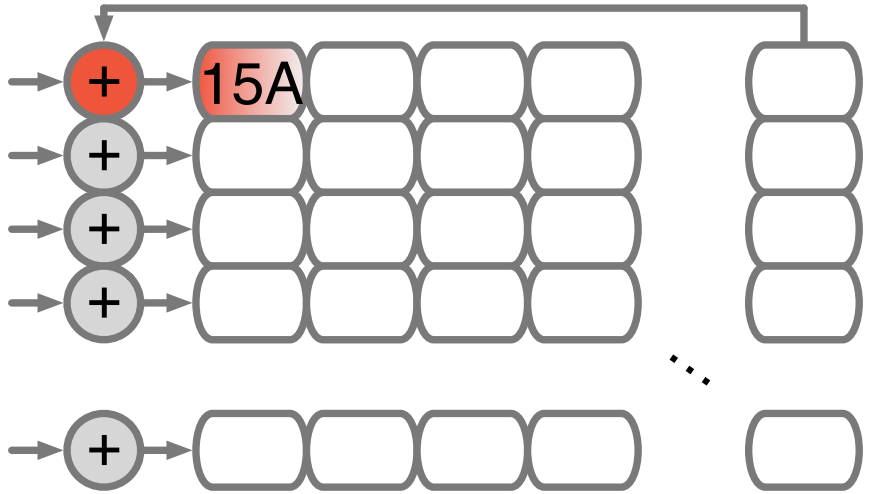
Propagates accumulator value

PE Row

Propagates temporal signal

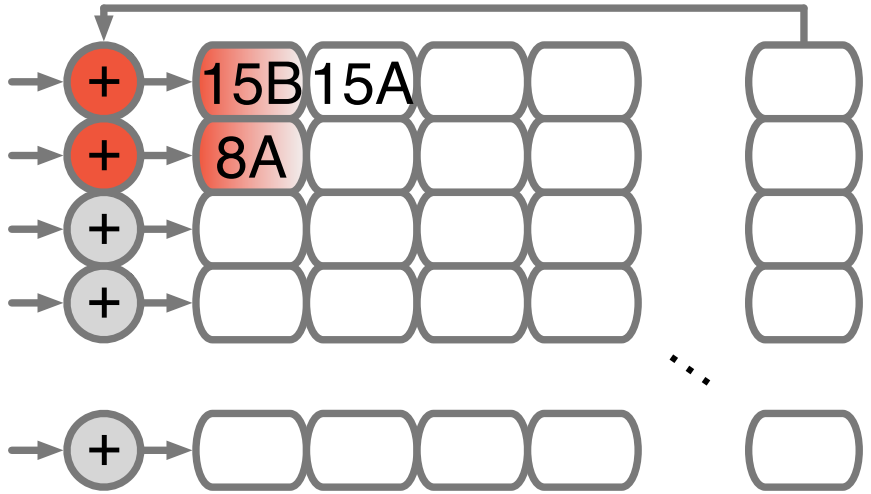
Walkthrough Timing Example

Cycle 16



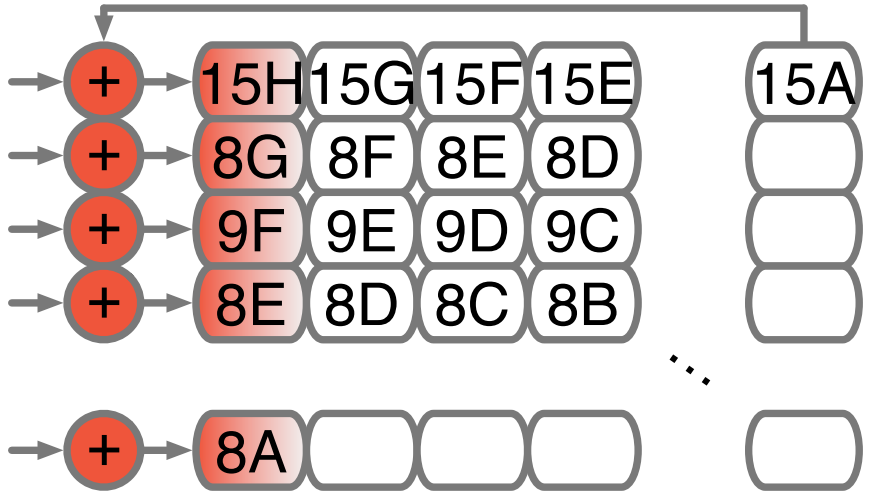
Walkthrough Timing Example

Cycle 17



Walkthrough Timing Example

Cycle 23



Evaluation Setup

Workload

MLPerf benchmark suite

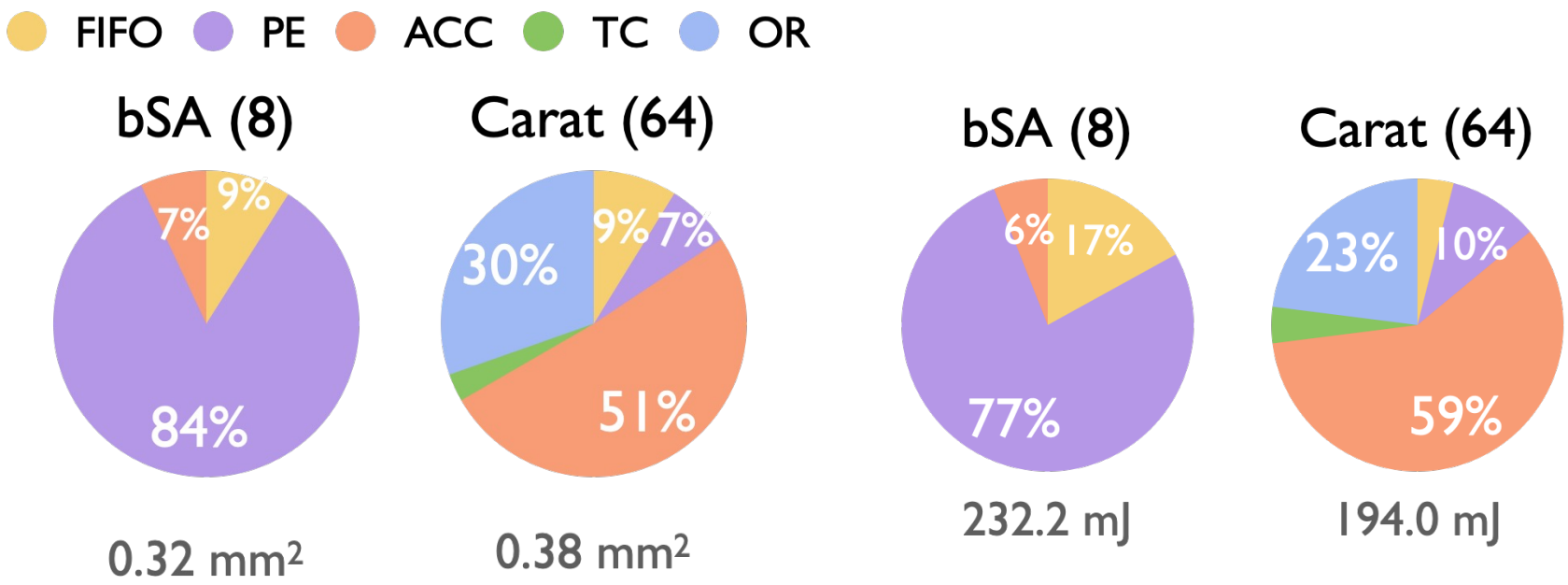
Baseline

- ❑ Binary systolic array (bSA): weight stationary
- ❑ uSystolic (uSA)
- ❑ Compute reuse-based accelerator (RIS): 50% input similarity

Hardware Configuration

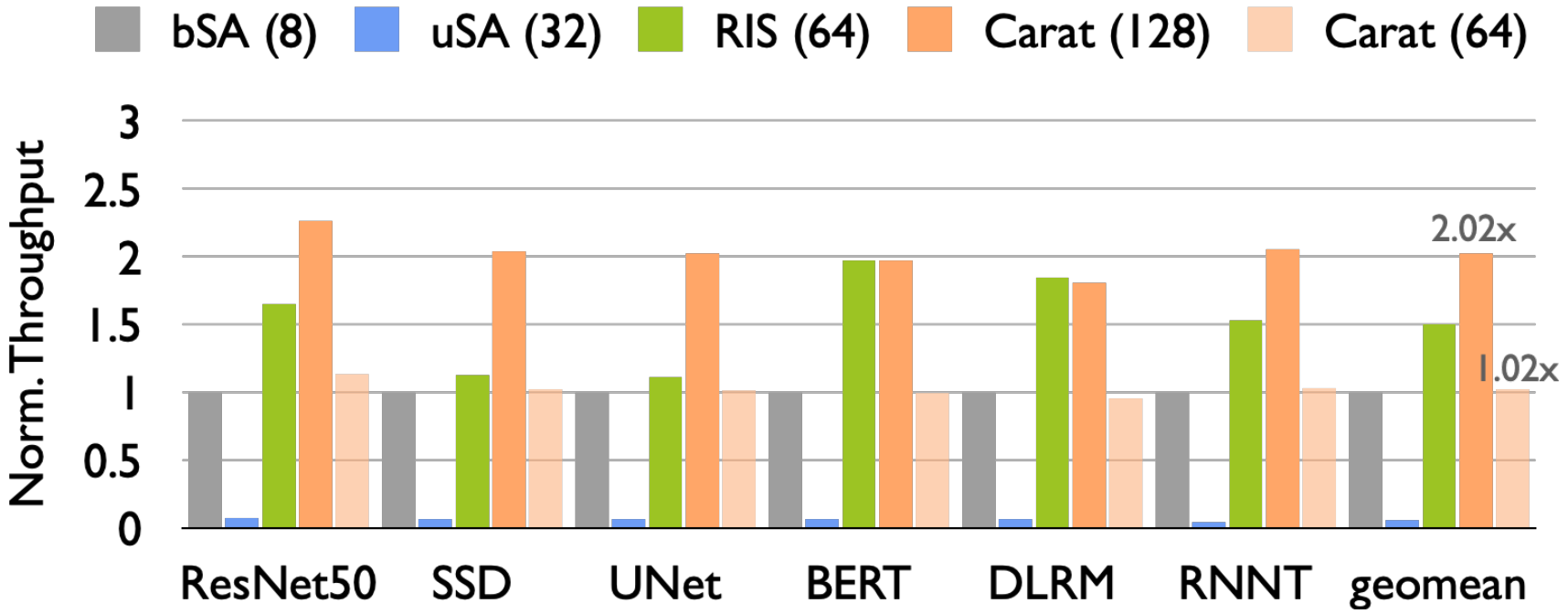
Config.	Carat	Binary systolic array (bSA)	Computation Reuse (RIS)	Unary systolic array (uSA)
i/w/o SRAM (KB)	128			N/A
Array height	32-512	4-16	1	16-64
Array width	8	Same as height	16-256	Same as height
MUL/ACC word	FP8/BF16			INT8/INT24

Different Design Space



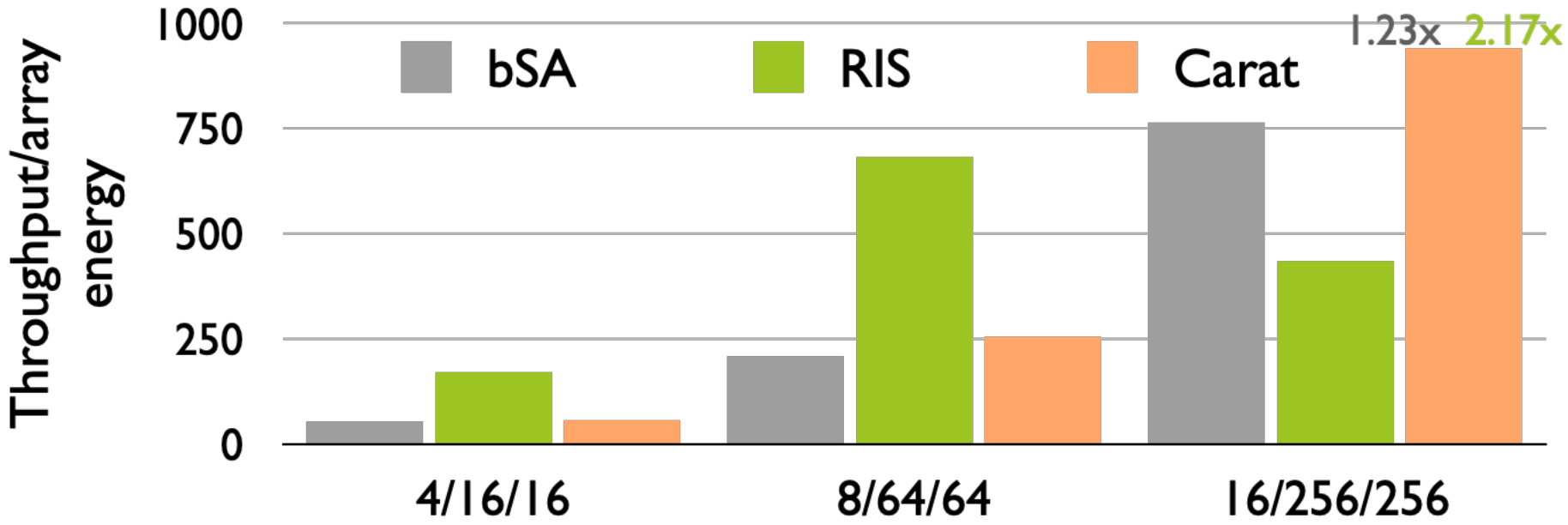
Carat PE only contributes to **7%** of array area and **10%** array energy.

Throughput @ batch size 256

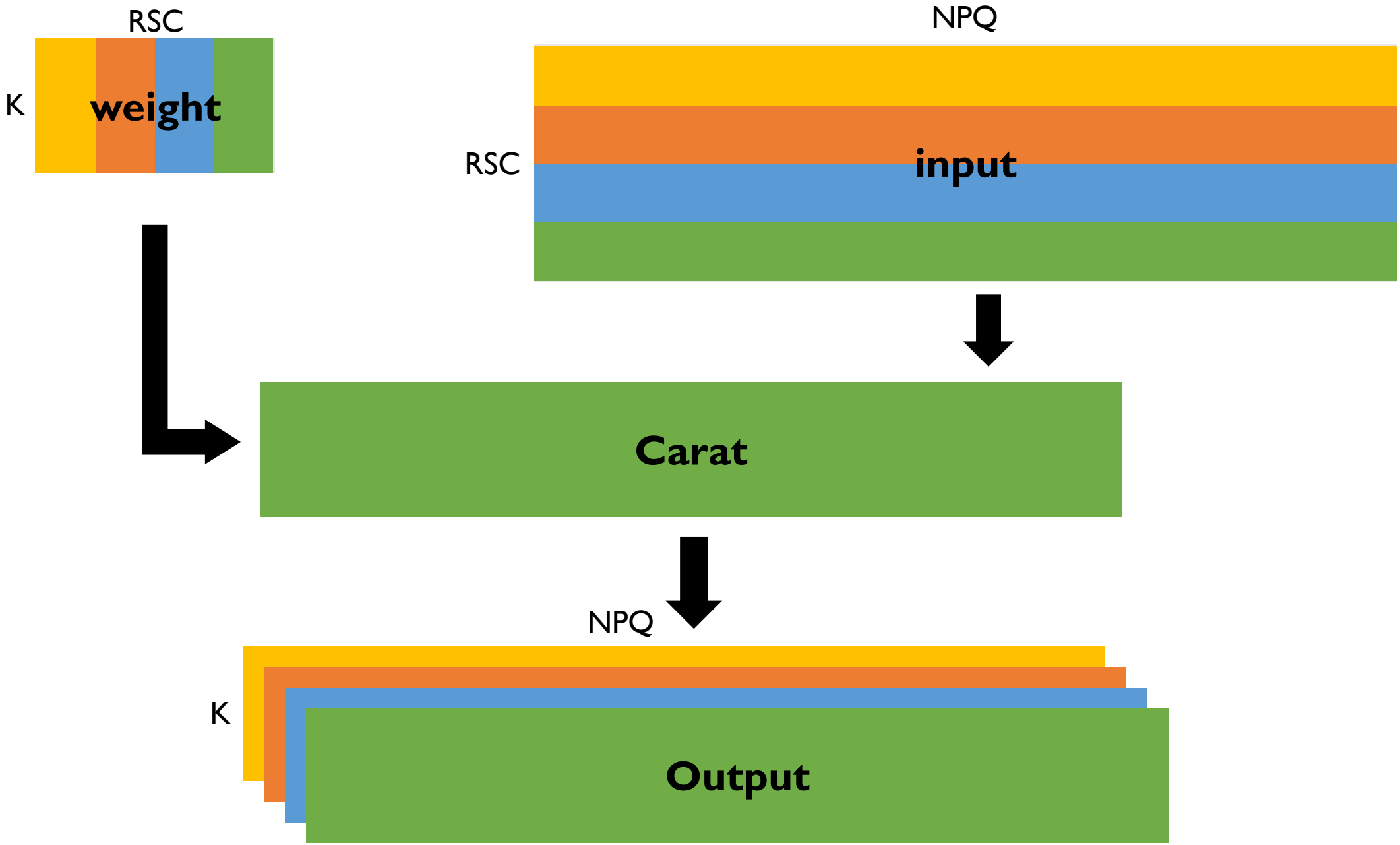


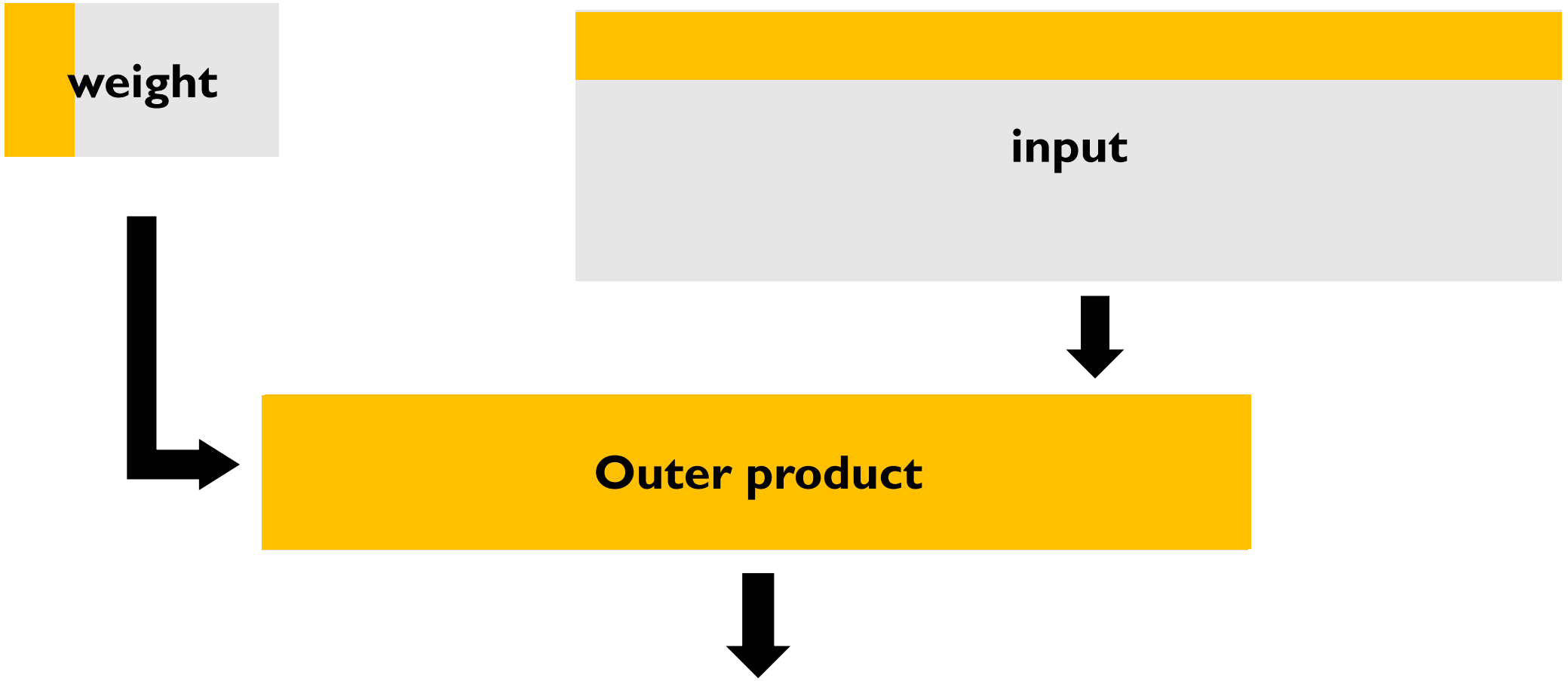
Throughput of **iso-area** Carat is comparable (1.02x) to that of bSA;
Throughput of **iso-flops** Carat is 2.02x better than that of bSA.

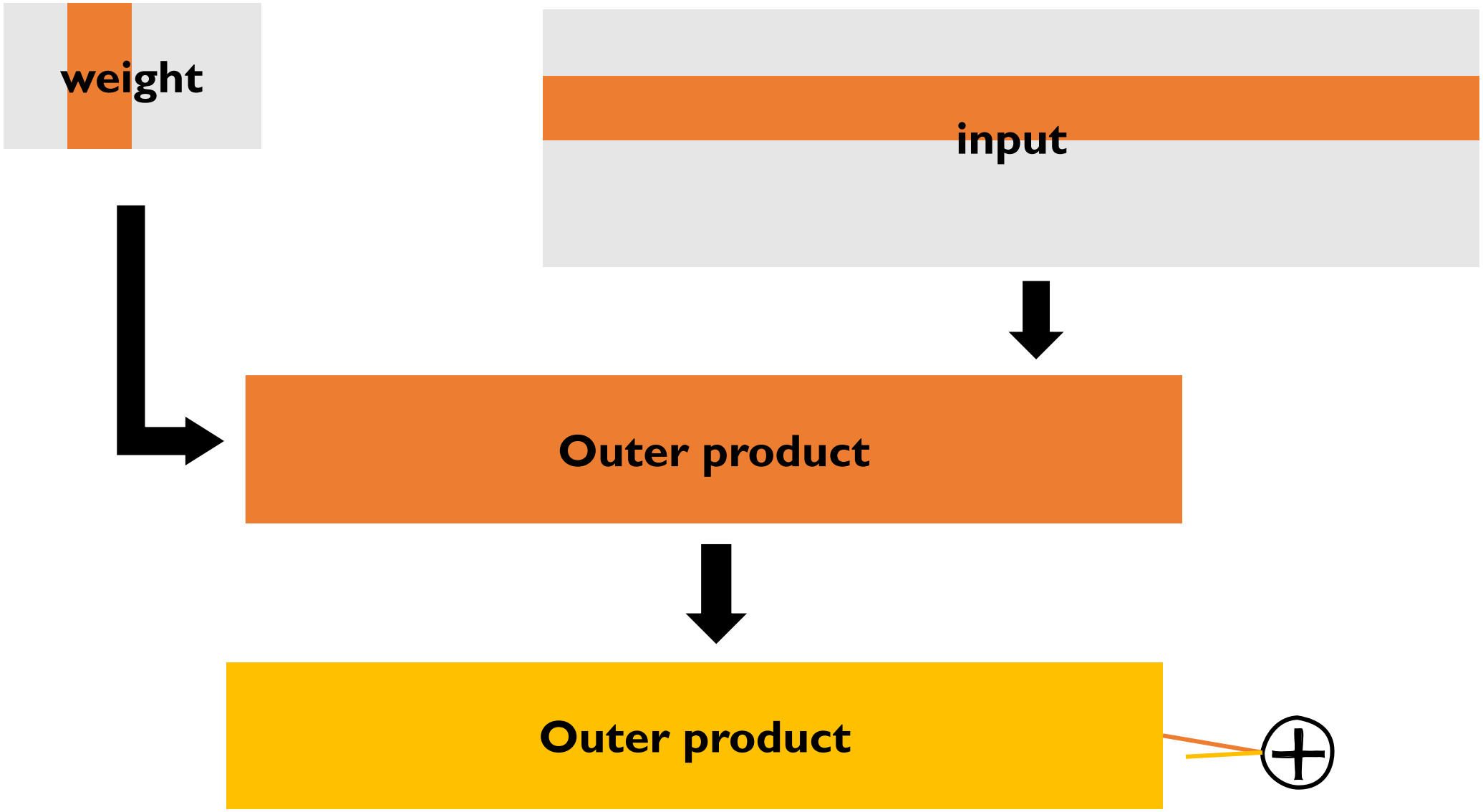
Energy Efficiency

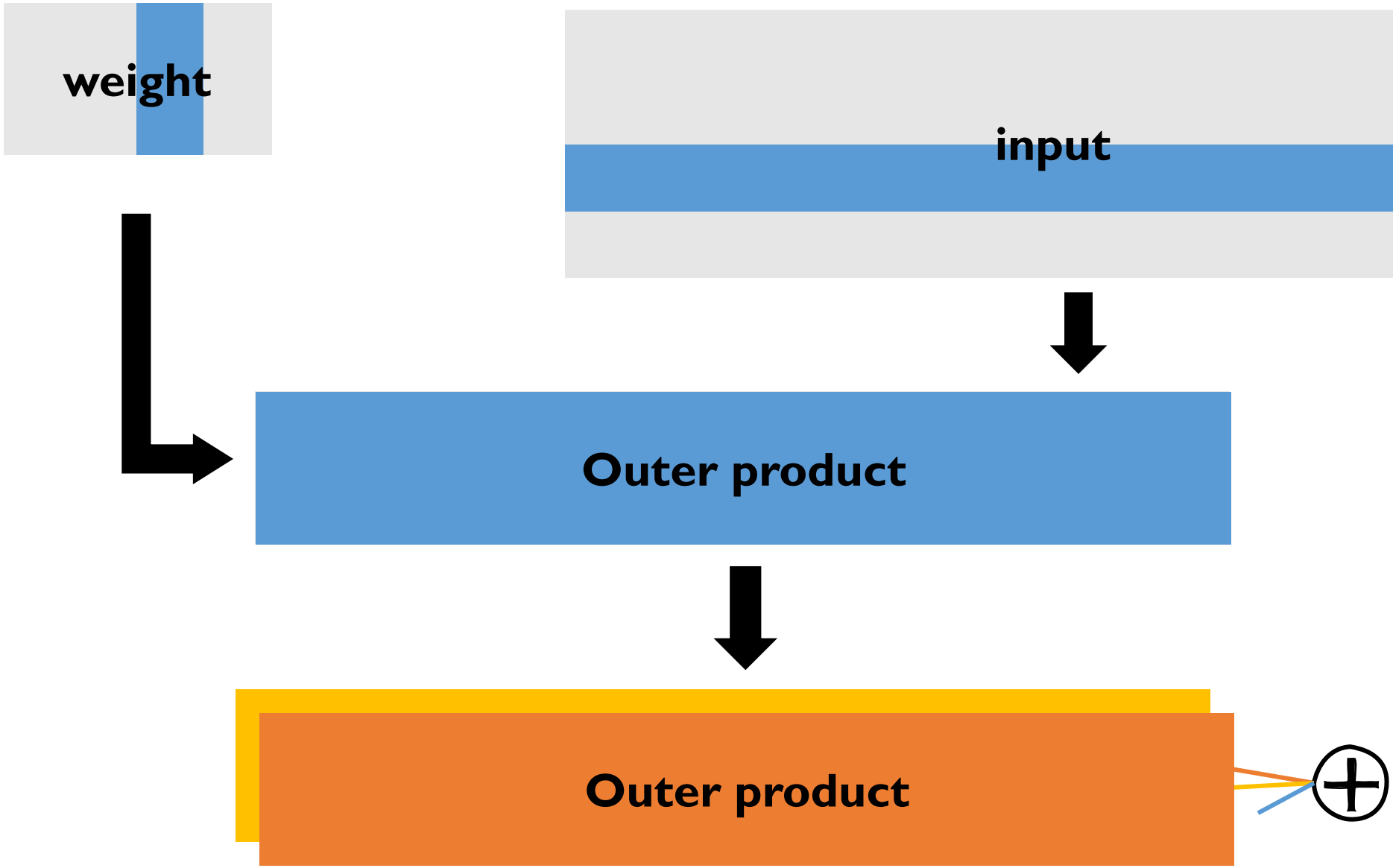


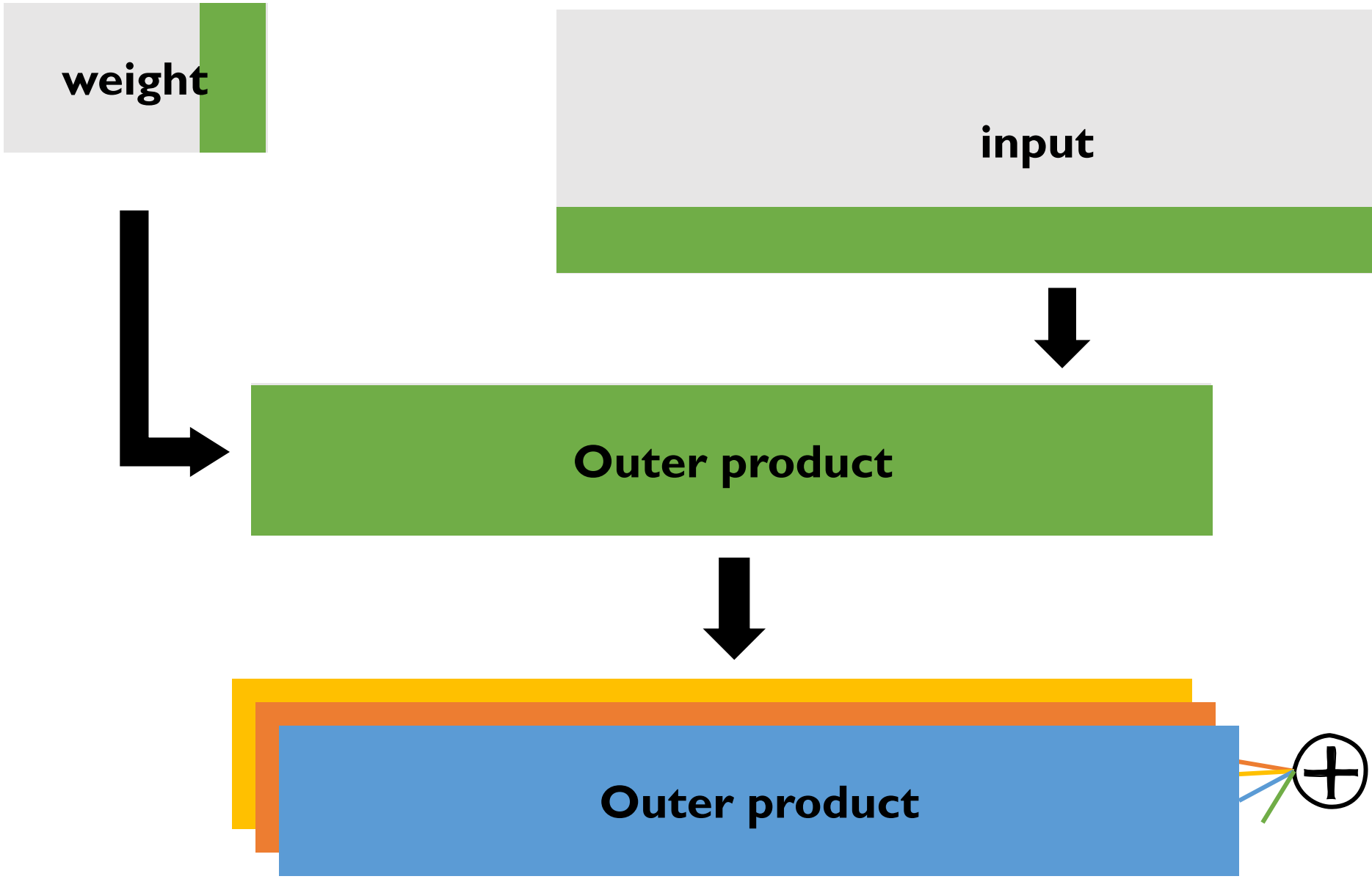
Carat achieves **1.23x** and **2.17x** array energy efficiency than bSA and RIS.


RSC
K
weight
NPQ
RSC
input
Carat
NPQ
K
Output


weight
input
Outer product


weight
input
Outer product
Outer product


weight
input
Outer product
Outer product


weight
input
Outer product
Outer product