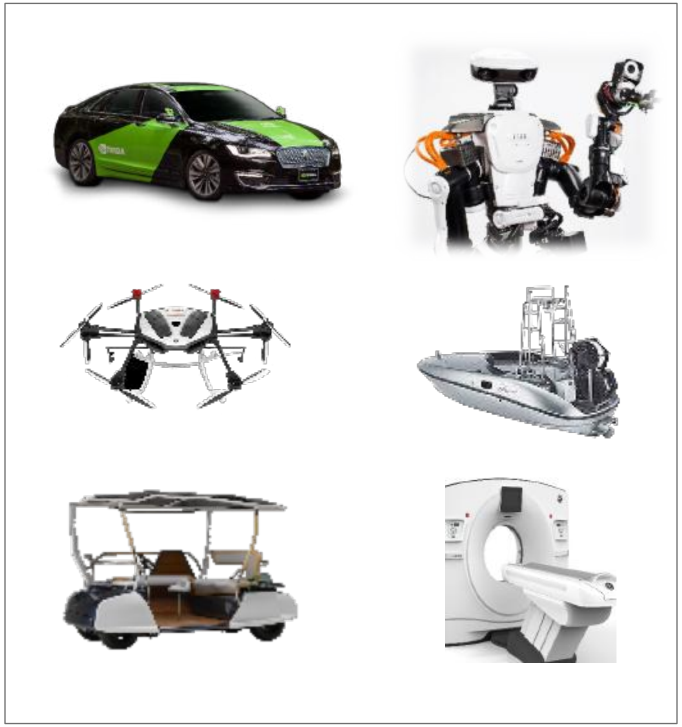
# SIMBA: SCALING DEEP-LEARNING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

Yakun Sophia Shao, Jason Clemons, Rangharajan Venkatesan, Brian Zimmer, Matthew Fojtik, Nan Jiang, Ben Keller, Alicia Klinefelter, Nathaniel Pinckney, Priyanka Raina, Stephen G. Tell, Yanqing Zhang, William J. Dally, Joel S. Emer, C. Thomas Gray, Brucek Khailany & Stephen W. Keckler

# VAST WORLD OF AI INFERENCE

## Creating A Massive Market Opportunity

MOBILE DEVICES



EMBEDDED COMPUTERS

DATACENTER COMPUTERS

# SCALABLE INFERENCE ACCELERATORS

## Challenges and Opportunities

### Motivation

- Need for fast and efficient inference accelerators from mobile to datacenter.

### Challenge

- High design cost of building unique hardware for each design target.

### Opportunities

- Deep learning inference is intrinsically scalable with abundant parallelism.
- Recent advances in package-level integration for multi-chip-module-based designs.

# THE MULTI-CHIP-MODULE APPROACH

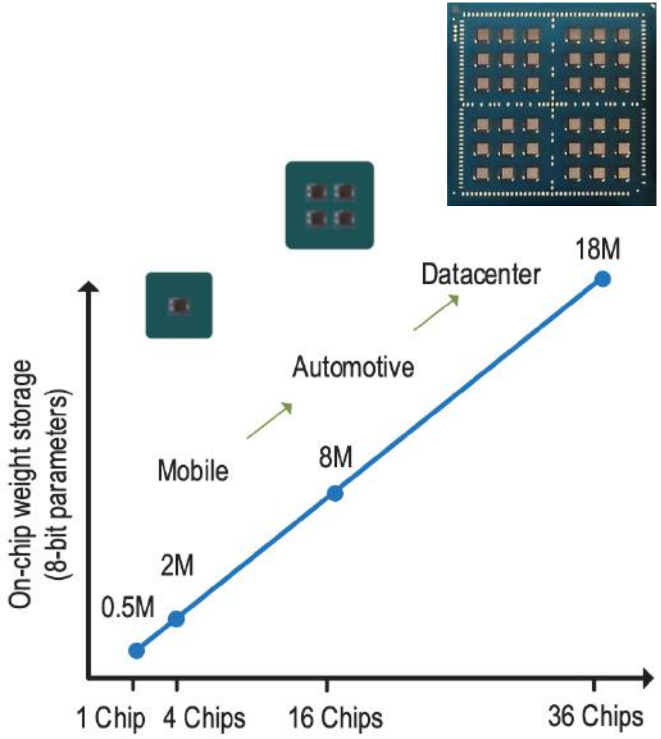
## Challenges and Opportunities

### Advantages:

Build systems larger than reticle limit
Smaller chips are cheaper to design
Smaller chips have higher yield
Faster time-to-market

### Challenges:

Area, energy, and latency for chip-to-chip communication

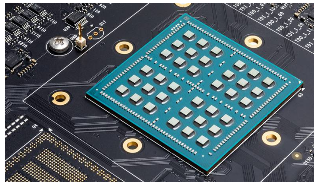


*Ref: Zimmer et al., VLSI 2019*

# SIMBA: SCALING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

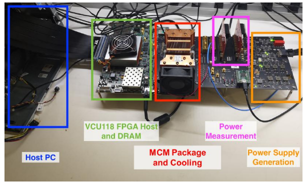
## Simba Testchip:

- Package and chiplet architecture
- Processing element design
- Baseline uniform tiling across chiplets and PEs



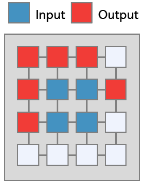
## Simba Characterization:

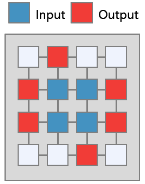
- Comparison with GPUs
- NoP bandwidth sensitivity
- NoP latency sensitivity



## Simba NoP-Aware Tiling:

- Non-uniform work partitioning
- Communication-aware data placement
- Cross-layer pipelining

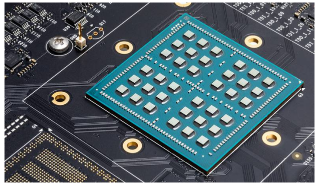




# SIMBA: SCALING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

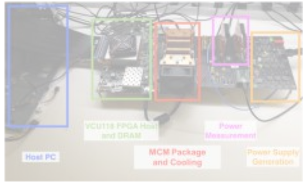
## Simba Testchip:

- Package and chiplet architecture
- Processing element design
- Baseline uniform tiling across chiplets and PEs



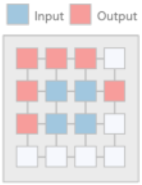
## Simba Characterization:

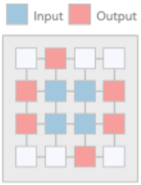
- Comparison with GPUs
- NoP bandwidth sensitivity
- NoP latency sensitivity



## Simba NoP-Aware Tiling:

- Non-uniform work partitioning
- Communication-aware data placement
- Cross-layer pipelining





# SIMBA: SCALABLE MCM-BASED ARCHITECTURE

## Simba Package and Chiplet

### Package and chiplet spec

6mm^2 chiplet in TSMC 16nm

36 chiplets/package

### Chip-to-chip interconnect

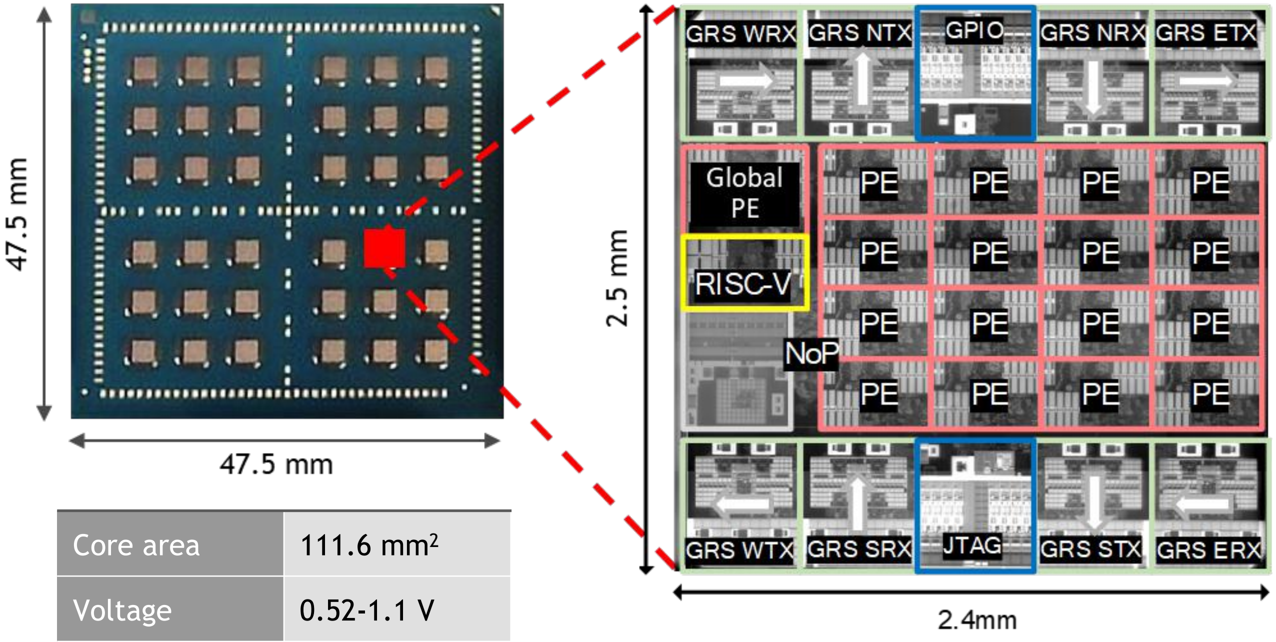
Ground-Referenced Signaling

### Efficient compute tiles

128 TOPS

0.11 pJ/Op

8-bit integer datapath

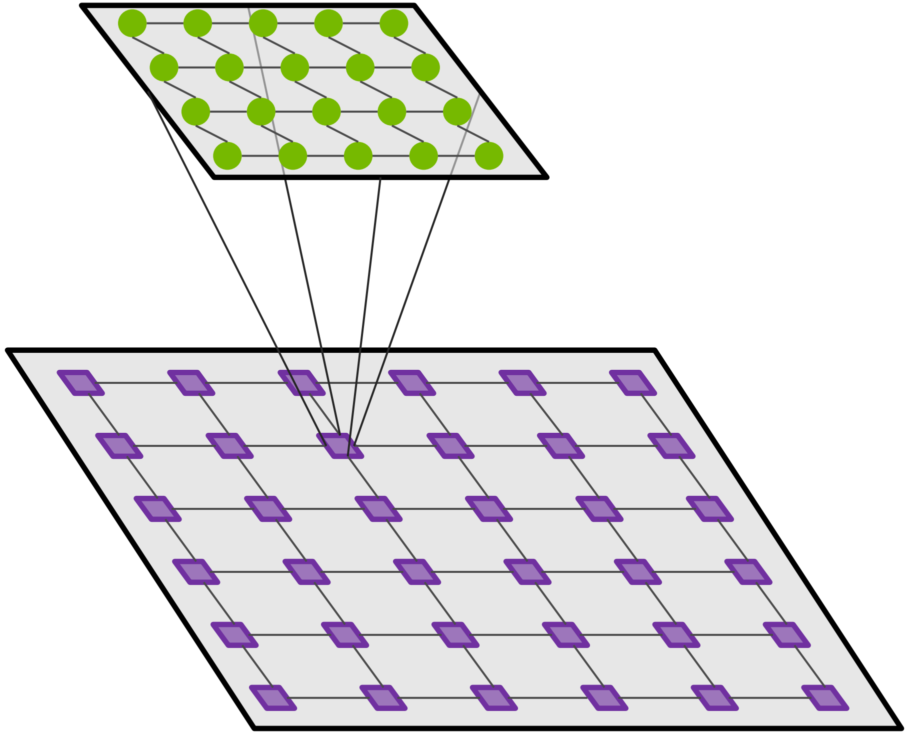


| | |
|---|---|
| Core area | 111.6 mm$^2$ |
| Voltage | 0.52-1.1 V |
| Frequency | 0.48-1.8 GHz |
| SRAM | 624KB/chip<br>23MB/package |

*Ref: Zimmer et al., VLSI 2019*

# SIMBA: NON-UNIFORM COMMUNICATION

## Network-on-Chip (NoC) and Network-on-Package (NoP)



### NETWORK-ON-CHIP (NoC)

4x5 mesh topology connects 16 PEs, one Global PE, and one RISC-V.

Cut-through routing with multicast support.

10ns per hop, 68GB/s/PE

### NETWORK-ON-PACKAGE (NoP)

6x6 mesh topology connects 36 chiplets in package.

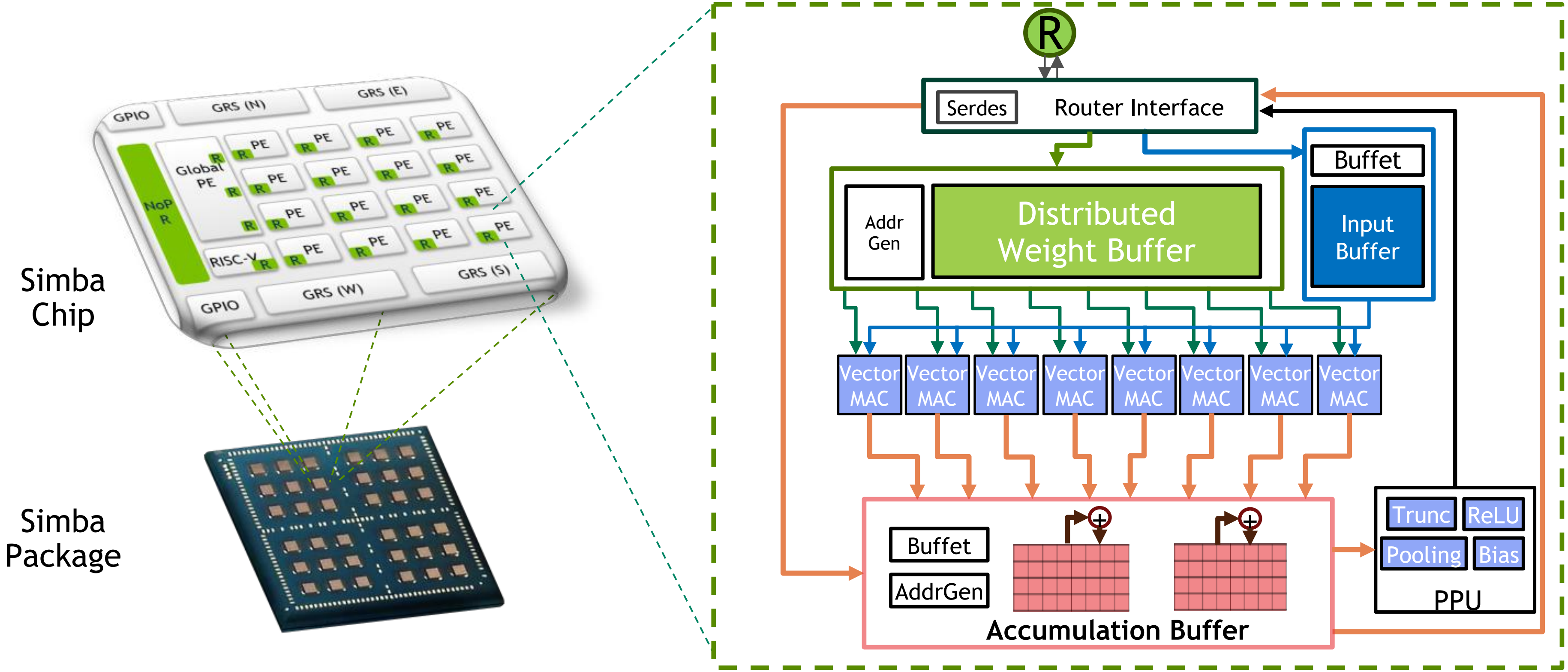
A NoP router per chiplet.

Configurable routing to avoid bad links/chiplets.

20ns per hop, 100GB/s/chiplet

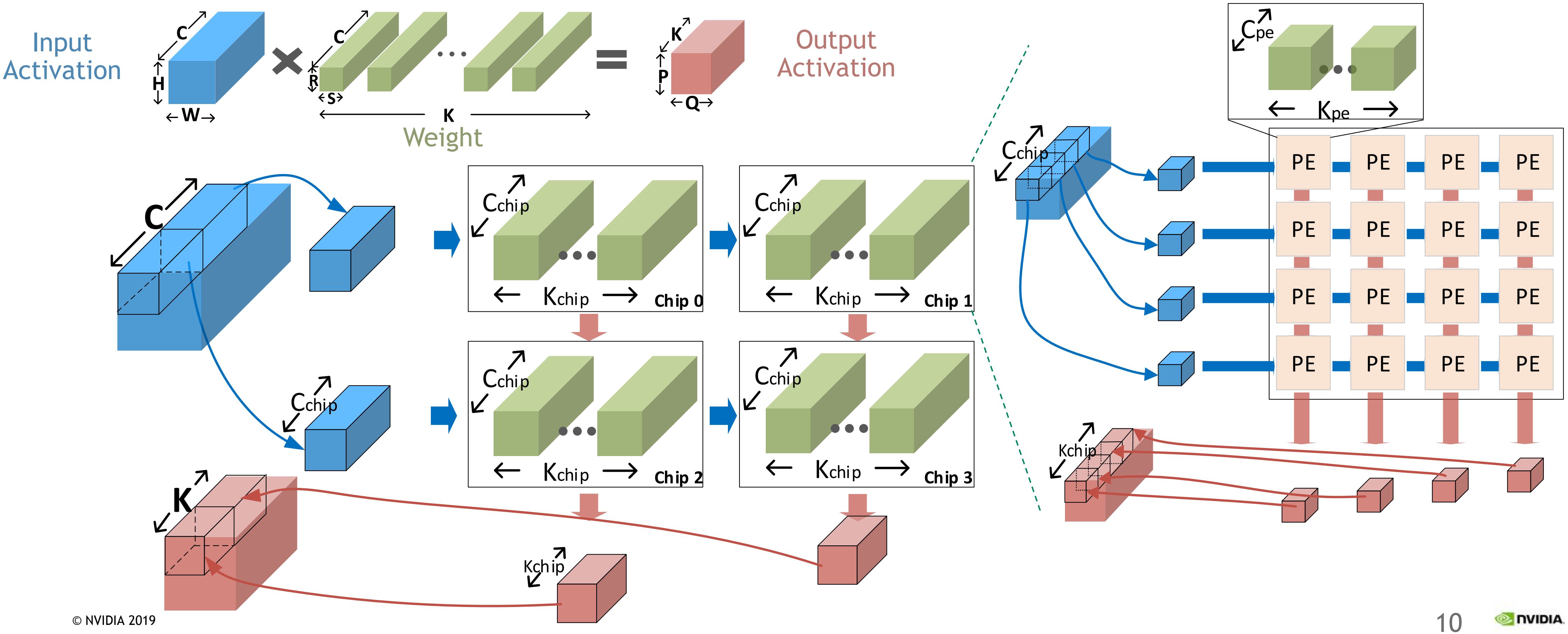
# SIMBA: SCALABLE MCM-BASED ARCHITECTURE

## Spatial Architecture with Distributed Memory



# BASELINE: UNIFORM TILING ACROSS NOC/NOP

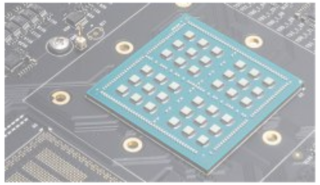
## Exploiting Model and Data Parallelism



# SIMBA: SCALING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

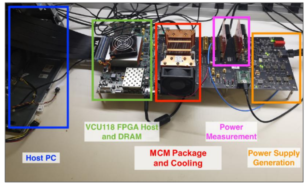
## Simba Testchip:

- Package and chiplet architecture
- Processing element design
- Baseline uniform tiling across chiplets and PEs



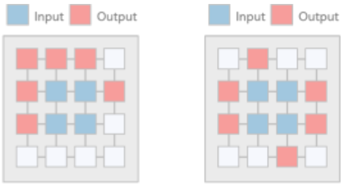
## Simba Characterization:

- Comparison with GPUs
- NoP bandwidth sensitivity
- NoP latency sensitivity



## Simba NoP-Aware Tiling:

- Non-uniform work partitioning
- Communication-aware data placement
- Cross-layer pipelining



# SIMBA MEASUREMENT SETUP

Measurements begin after weights and activations are loaded from FPGA DRAM

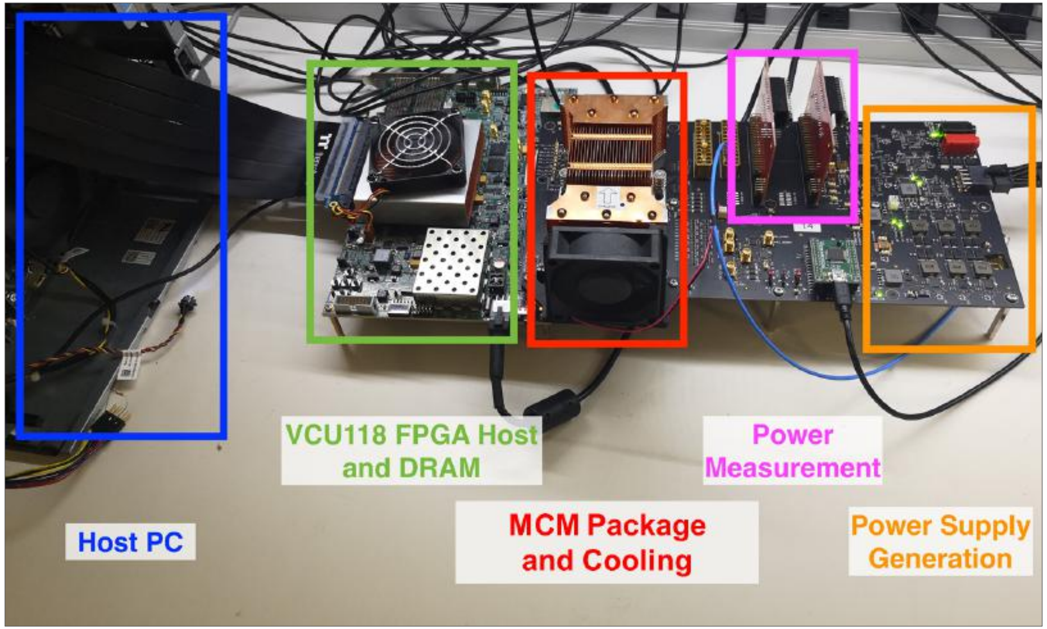
- Weights are loaded to PE memory
- Activations are loaded to Global PE

Operating points

- Voltage: 0.72 V
- Frequency: 1.03 GHz
- GRS Bandwidth: 11 Gbps/pin

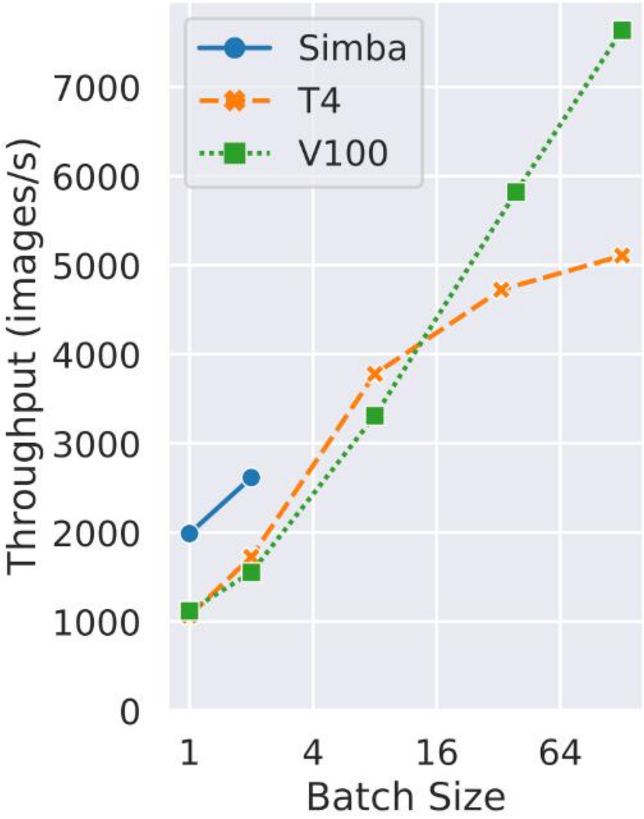
Networks

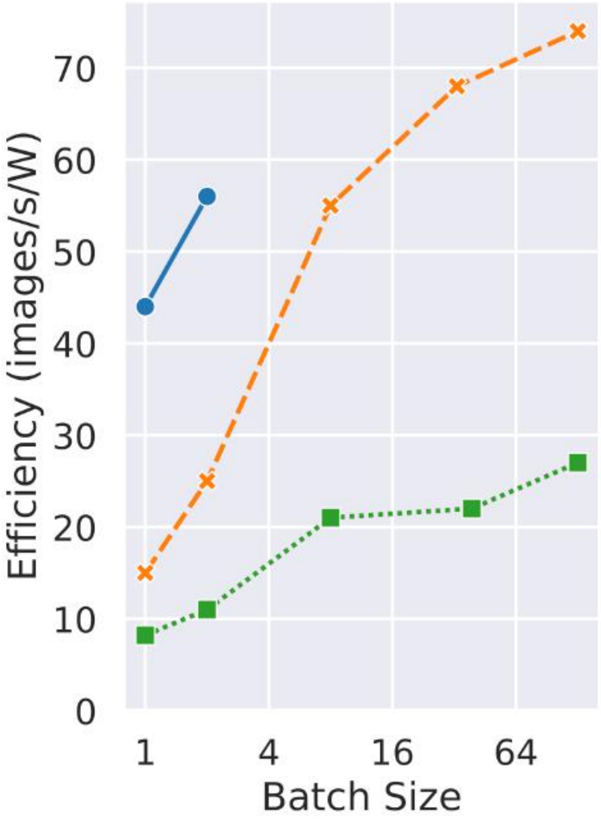
- DriveNet, AlexNet, ResNet-50



# SIMBA CHARACTERIZATION

## Comparison with GPUs running ResNet-50

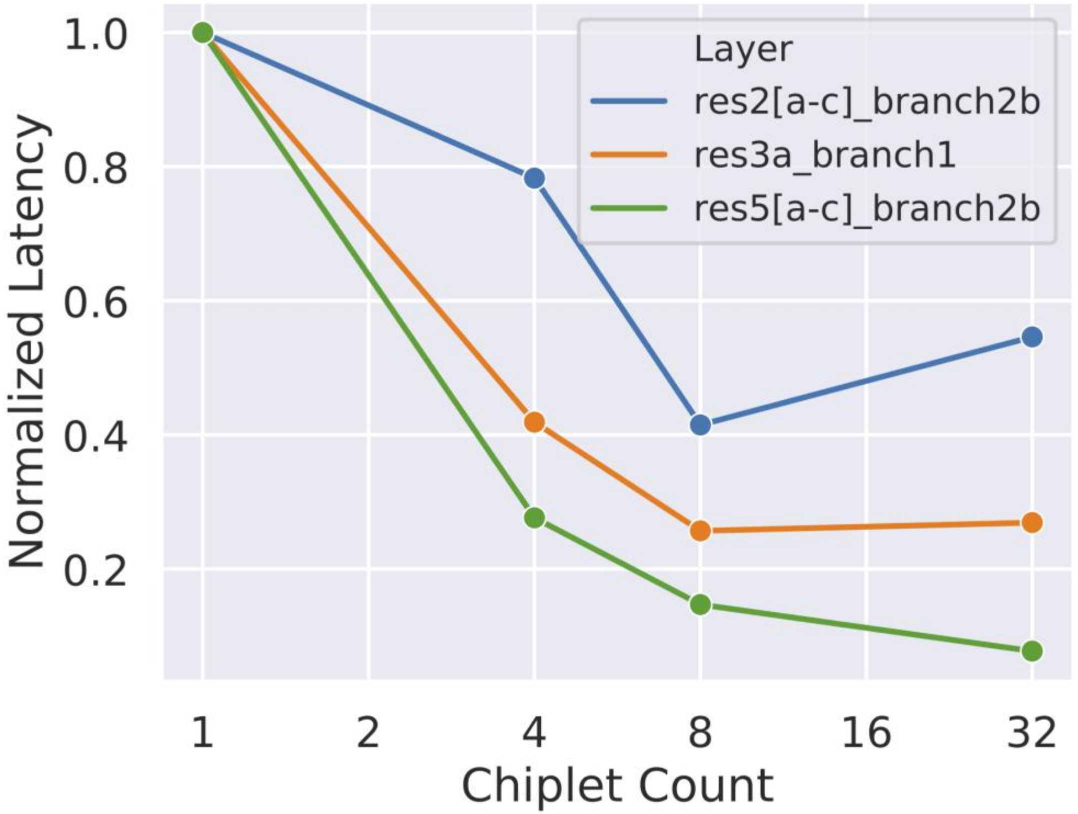




# SIMBA CHARACTERIZATION

## Layer Sensitivity

- Running three ResNet-50 layers across different number of chiplets.
- Increasing the number of active chiplets does not always translate to performance gains.
- The cost of communication hinders the ability to exploit parallelism.

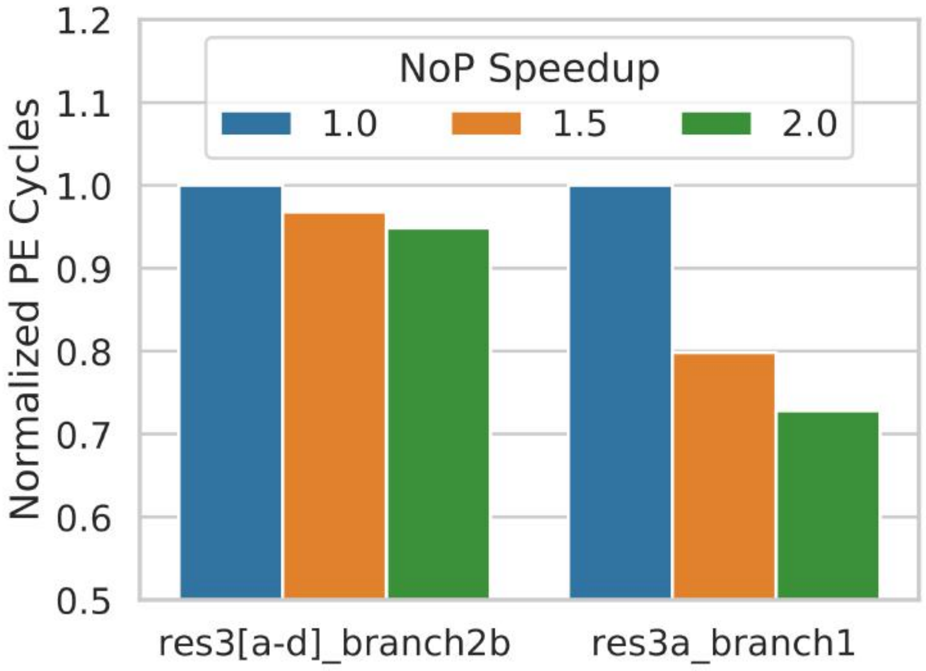


# SIMBA CHARACTERIZATION

## NoP Bandwidth Sensitivity

- Running two ResNet-50 layers across 32 chiplets with different NoP bandwidths by adjusting the ratio between NoP and PE frequencies.

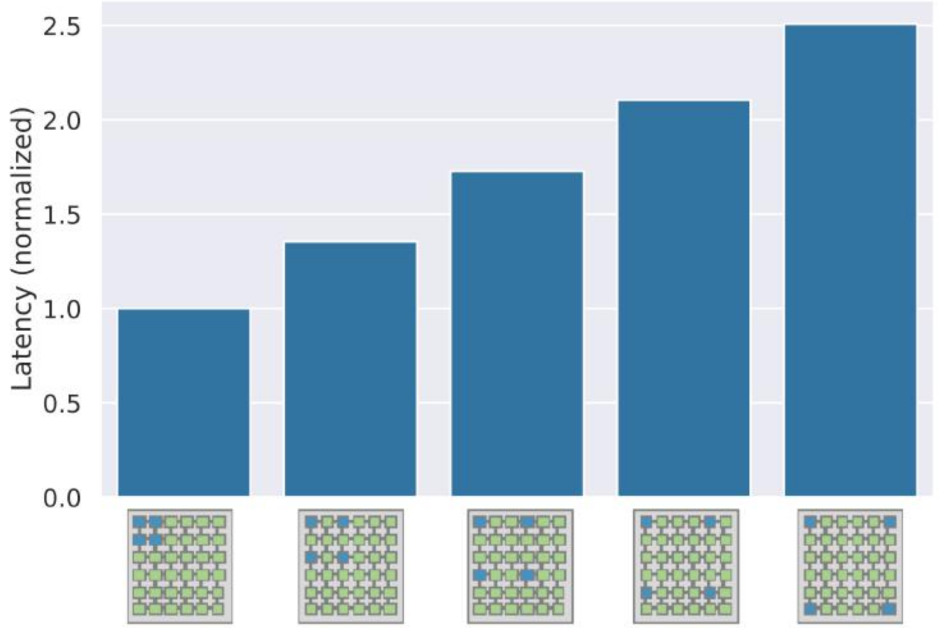
- End-to-end performance is sensitive to NoP bandwidth, especially for applications with a significant amount of communication.



# SIMBA CHARACTERIZATION

## NoP Latency Sensitivity

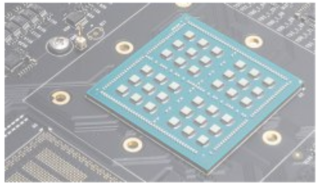
- Mapping res4a_branch1 to four chiplets with different data placements (shown in the X-axis)
- Inter-chiplet traffic type:
  - Input activation multicast
  - Output activation writeback
- Communication latency plays a key role in achieving good performance/efficiency in a large-scale system.



# SIMBA: SCALING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

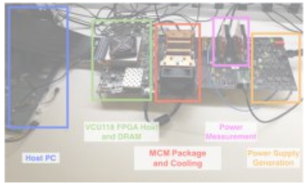
## Simba Testchip:

- Package and chiplet architecture
- Processing element design
- Baseline uniform tiling across chiplets and PEs



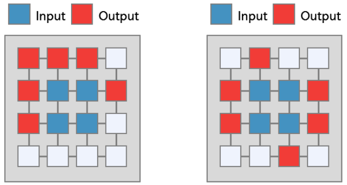
## Simba Characterization:

- Comparison with GPUs
- NoP bandwidth sensitivity
- NoP latency sensitivity



## Simba NoP-Aware Tiling:

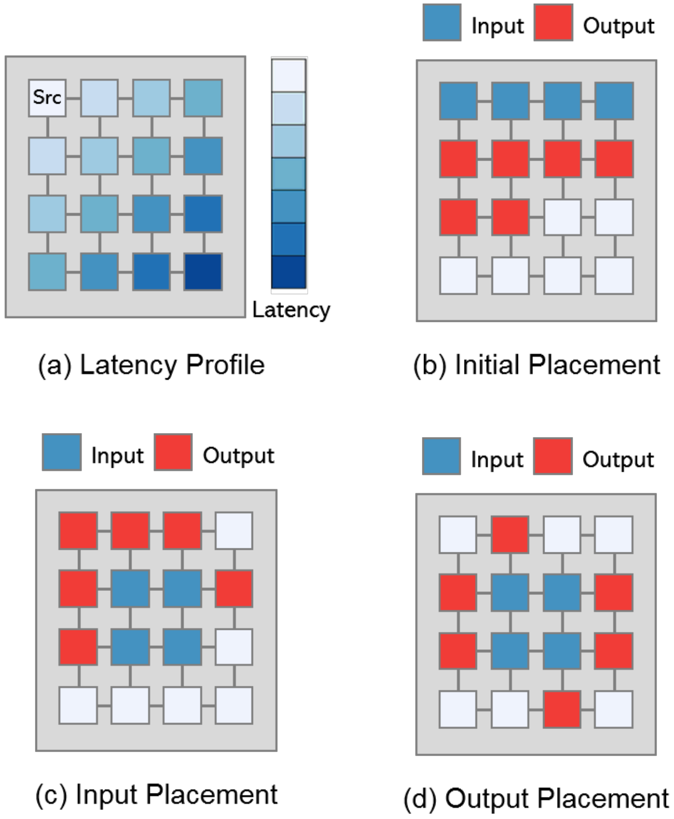
- Non-uniform work partitioning
- Communication-aware data placement
- Cross-layer pipelining



# SIMBA NOP-AWARE TILING

## Communication-Aware Data Placement

- Communication latency is highly sensitive to the physical location of data, which is explicitly managed by programmers.
- We use an iterative algorithm to place the input and output activations to minimize the number of NoP hops.
- We keep the same tiling in the updated placements to reduce the number of variables.



(a) Latency Profile

(b) Initial Placement

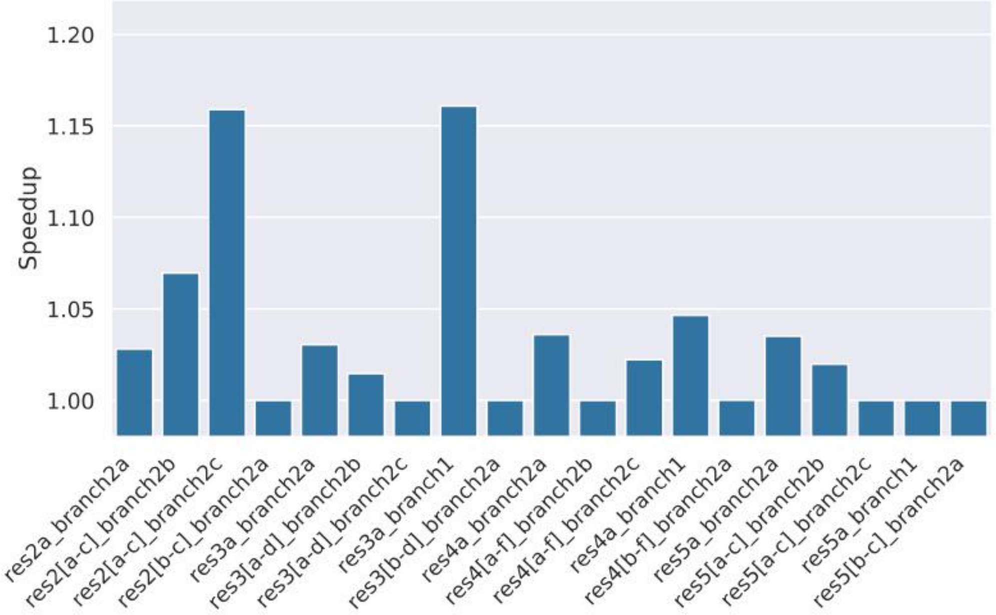
(c) Input Placement

(d) Output Placement

# SIMBA NOP-AWARE TILING

## Communication-Aware Data Placement

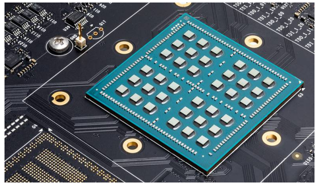
- Communication latency is highly sensitive to the physical location of data, which is explicitly managed by programmers.
- We use an iterative algorithm to place the input and output activations to minimize the number of NoP hops.
- We keep the same tiling in the updated placements to reduce the number of variables.



# SIMBA: SCALING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

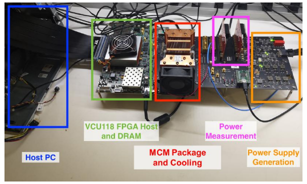
## Simba Testchip:

- Package and chiplet architecture
- Processing element design
- Baseline uniform tiling across chiplets and PEs



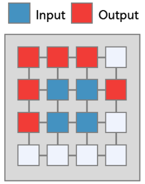
## Simba Characterization:

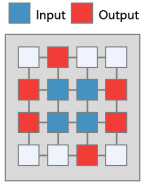
- Comparison with GPUs
- NoP bandwidth sensitivity
- NoP latency sensitivity

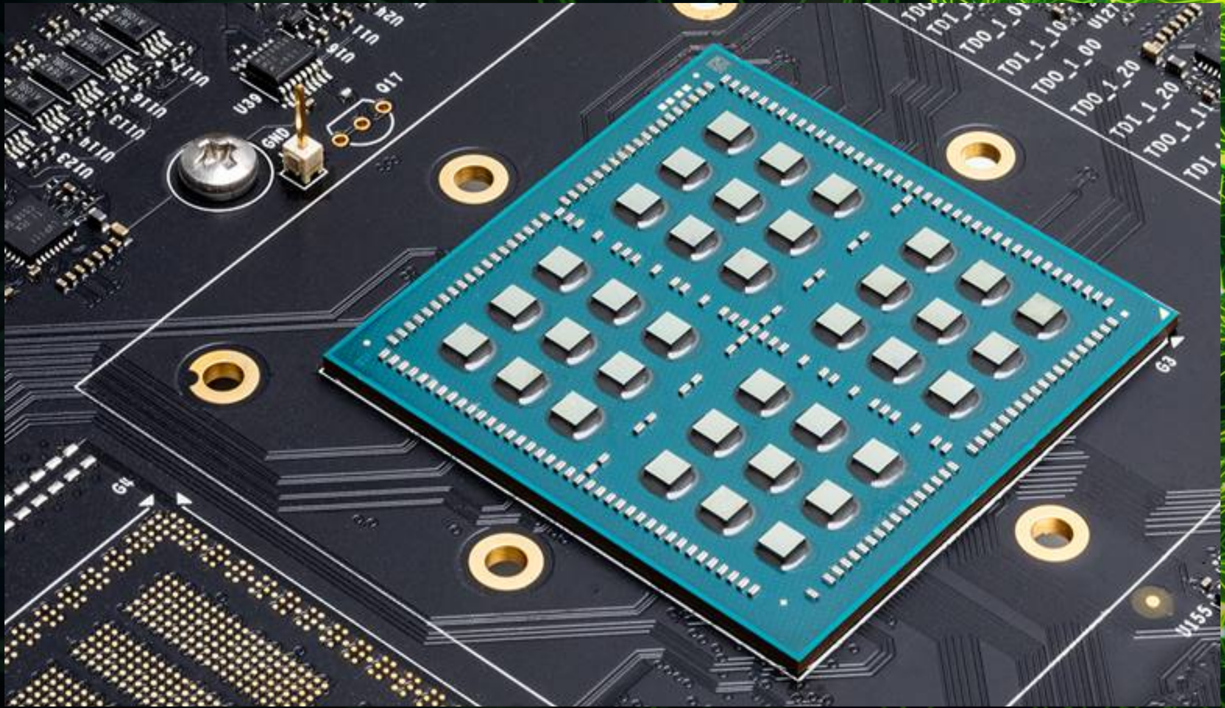


## Simba NoP-Aware Tiling:

- Non-uniform work partitioning
- Communication-aware data placement
- Cross-layer pipelining







# SIMBA: SCALING DEEP-LEARNING INFERENCE WITH MULTI-CHIP-MODULE-BASED ARCHITECTURE

NVIDIA®