University of Pittsburgh

CHARM: Composing Heterogeneous AcceleRators for Matrix Multiply on Versal ACAP Architecture

Peipei Zhou's Group

https://peipeizhou-eecs.github.io/

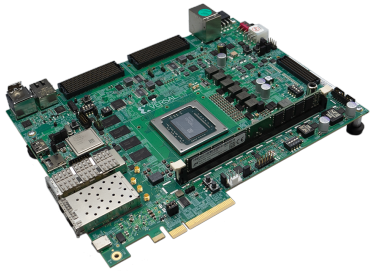
Customized Architecture and Programming
Heterogeneous Computing with FPGA, GPU, ASIC, NPU

University of Pittsburgh

2023/04 @ 2023 CMU Crossroads
Peipei Zhou, Assistant Professor

- Experience:
  - 2012-2019: CS Ph.D. @ UCLA with Prof. Jason Cong
  - 2019-2021: Staff Software Engr. @ Enflame, AI ASIC Startup
  - 2021/09 – now: Assistant Professor @ Pitt-ECE
- Research:
  - **A**rchitecture: FPGA, AI ASIC, GPU, etc.
  - **A**bstraction: Compiler Support
  - **A**pplication: Artificial Intelligence, Genome Pipeline
- Teaching:
  - Reconfigurable Computing in Deep Learning, 2022 Spring
  - Computer Organization and Architecture, 2022 Fall
  - Embedded System, Reconfigurable Computing in Deep Learning, 2023 Spring

Peipei Zhou's Group
https://peipeizhou-eecs.github.io/



Xilinx ACAP: Heterogeneous SoC

Research Highlights

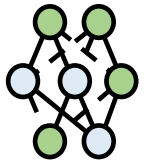
https://peipeizhou-eecs.github.io/

	Chip	Node	Data Center
Application	ICCAD'16: Caffeine, Towards Uniformed Representation and Acceleration for Deep Convolutional Neural Networks **[Best Paper]**	ISPASS'18: Doppio: I/O-Aware Performance Analysis, Modeling and Optimization for In-Memory Computing Framework(🔥 **Best Paper Nominee**)	IMSB'15:CS-BWAMEM: A fast and scalable read aligner at the cloud scale for whole genome sequencing
Abstraction	FCCM'16: Energy Efficiency of Full Pipelining DAC'16: Bandwidth Optimization Through On-Chip Memory Restructuring for HLS	FCCM'18: ST-Accel: A High-Level Programming Platform for Streaming Applications on FPGA	FPGA'21: MOCHA: Multinode Cost Optimization in Heterogeneous Clouds with Accelerators
Architecture	FCCM'14: A Fully Pipelined and Dynamically Composable Architecture of CGRA. FPGA'16: Accelerator-Rich Architecture FCCM'18: Latte: Locality Aware Transformation for High-Level Synthesis	ICCAD'18: SODA: Stencil with Optimized Dataflow Architecture (🔥 **Best Paper Nominee**). FCCM'20: Algorithm-Hardware Co-design for BQSR Acceleration in Genome Analysis ToolKit	TECS'21: Algorithm-hardware Co-design of Attention Mechanism on FPGA Devices TODAES'21: EF-Train: Enable Efficient On-device CNN Training on FPGA Through Data Reshaping for Online Adaptation or Personalization.

Research Highlights

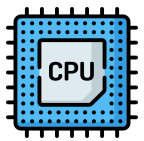
https://peipeizhou-eecs.github.io/

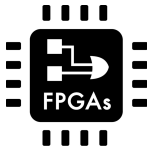
Application

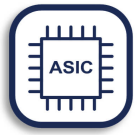



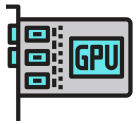
Abstraction

Architecture









Chip

FPGA23: CHARM: Composing Heterogeneous AcceleRators for Matrix Multiply on Versal ACAP Architecture, Proceedings of the 2023 ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '23), February 12–14, 2023, Monterey, CA, USA. ACM, New York, NY, USA, 12 pages. (🔥 **New Paper**). https://doi.org/10.1145/3543622.3573210.

DAC23: High Performance, Low Power Matrix Multiply Design on ACAP: from Architecture, Design Challenges and DSE Perspectives To appear in Proceedings of the 60th ACM/IEEE Design Automation Conference, San Francisco, California, USA, (DAC '23), July 9–13, 2023, San Francisco, CA, USA. (🔥 **New Paper**)

Node

ISPASS'18: Doppio: I/O-Aware Performance Analysis, Modeling and Optimization for In-Memory Computing Framework(🔥 **Best Paper Nominee**)

FCCM'18: ST-Accel: A High-Level Programming Platform for Streaming Applications on FPGA

ICCAD'18: SODA: Stencil with Optimized Dataflow Architecture (🔥 **Best Paper Nominee**).

FCCM'20: Algorithm-Hardware Co-design for BQSR Acceleration in Genome Analysis ToolKit

Data Center

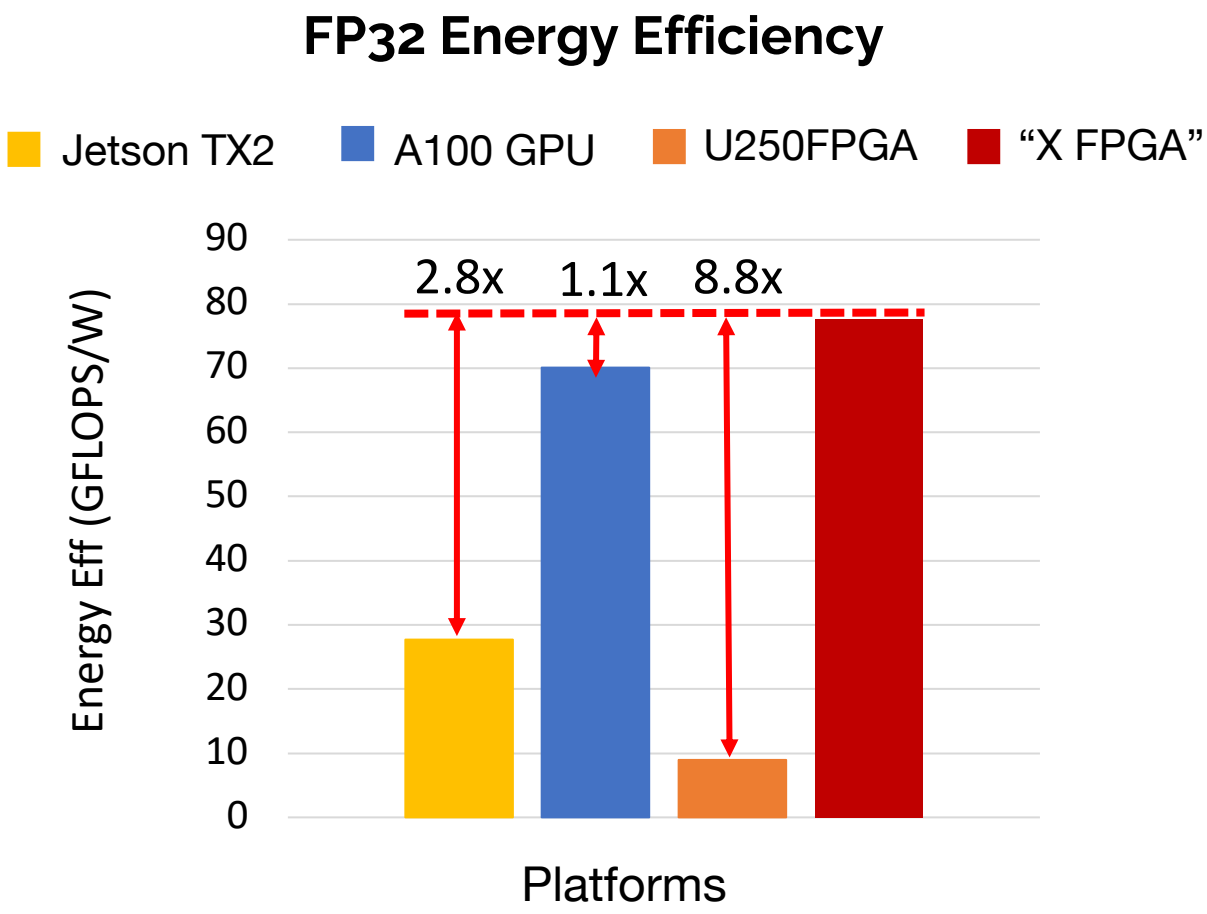
IMSB'15:CS-BWAMEM: A fast and scalable read aligner at the cloud scale for whole genome sequencing

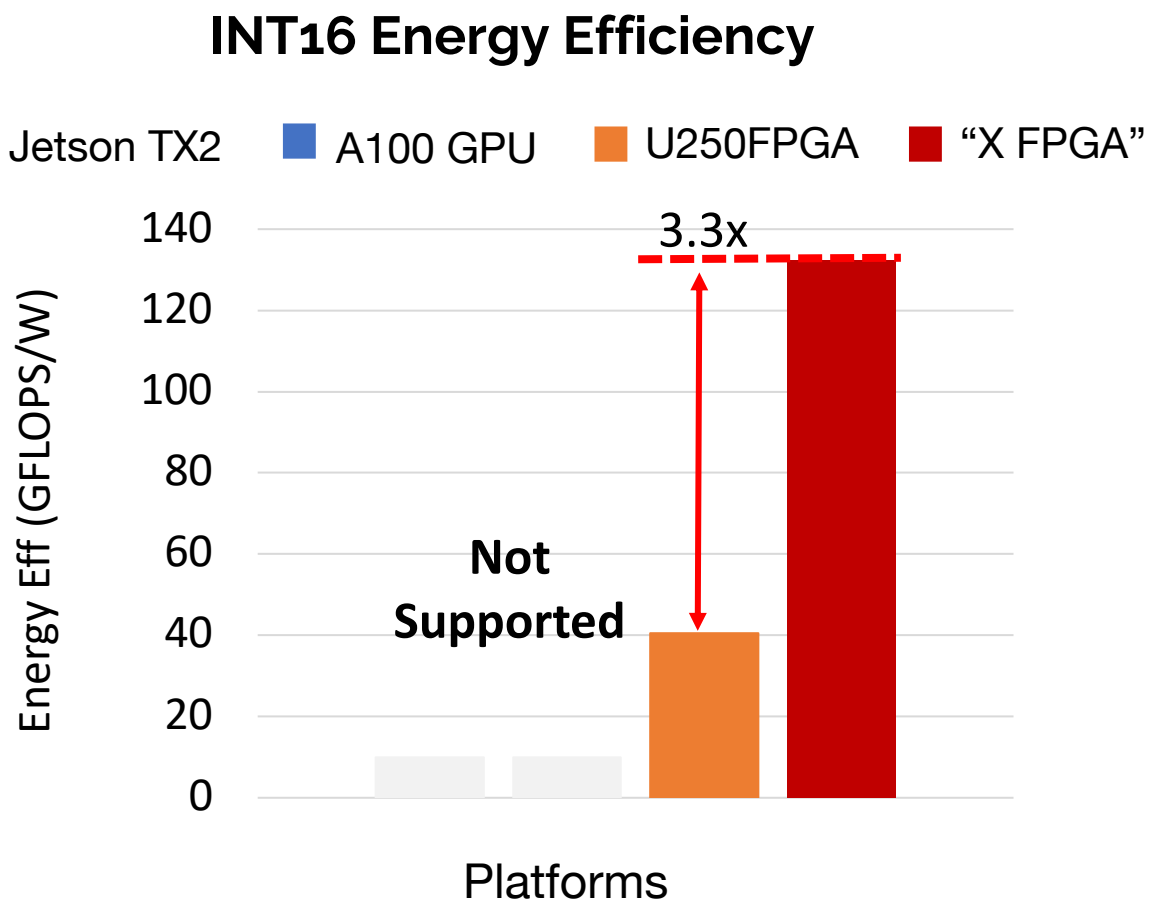
FPGA'21: MOCHA: Multinode Cost Optimization in Heterogeneous Clouds with Accelerators

TECS'21: Algorithm-hardware Co-design of Attention Mechanism on FPGA Devices
TODAES'21: EF-Train: Enable Efficient On-device CNN Training on FPGA Through Data Reshaping for Online Adaptation or Personalization.

“FPGA is Dead” vs “Long Live the FPGA”

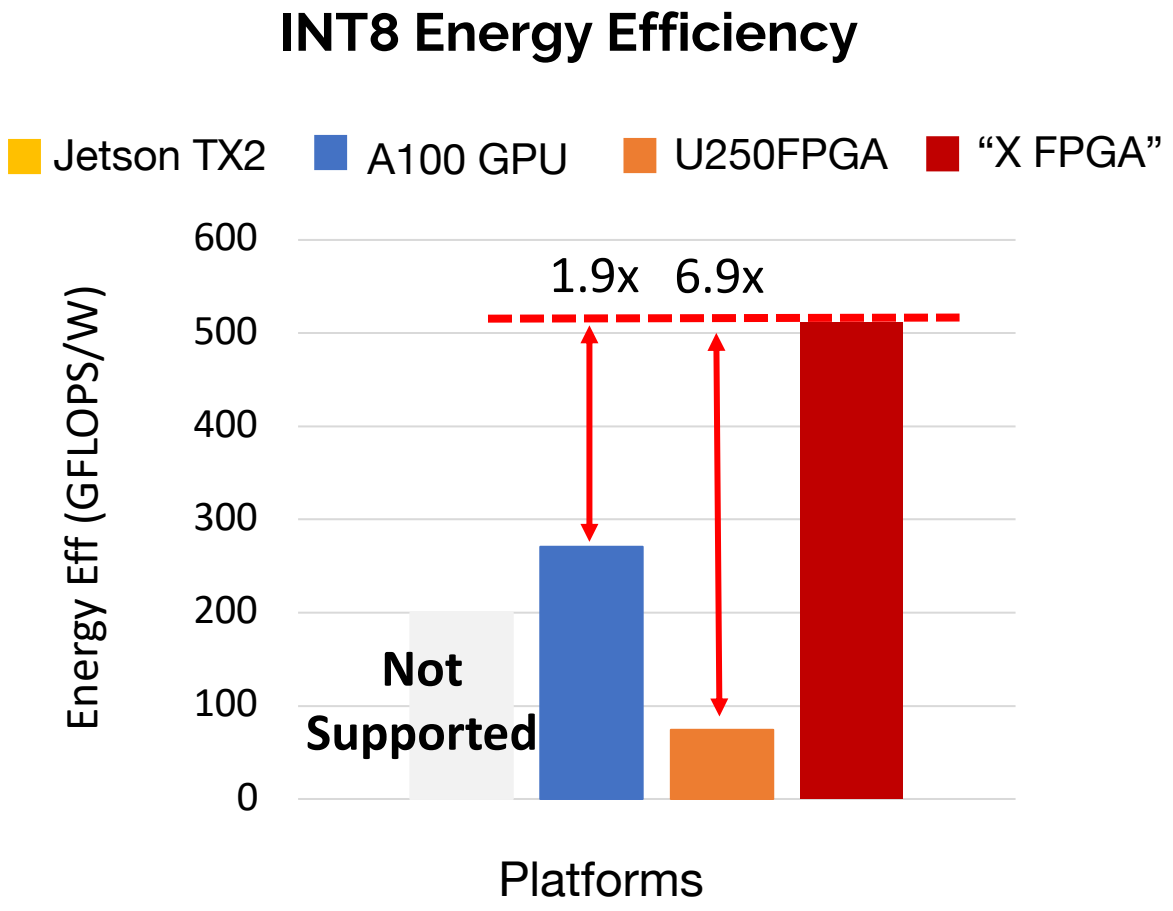
- **Single GEMM Energy Efficiency Comparisons**



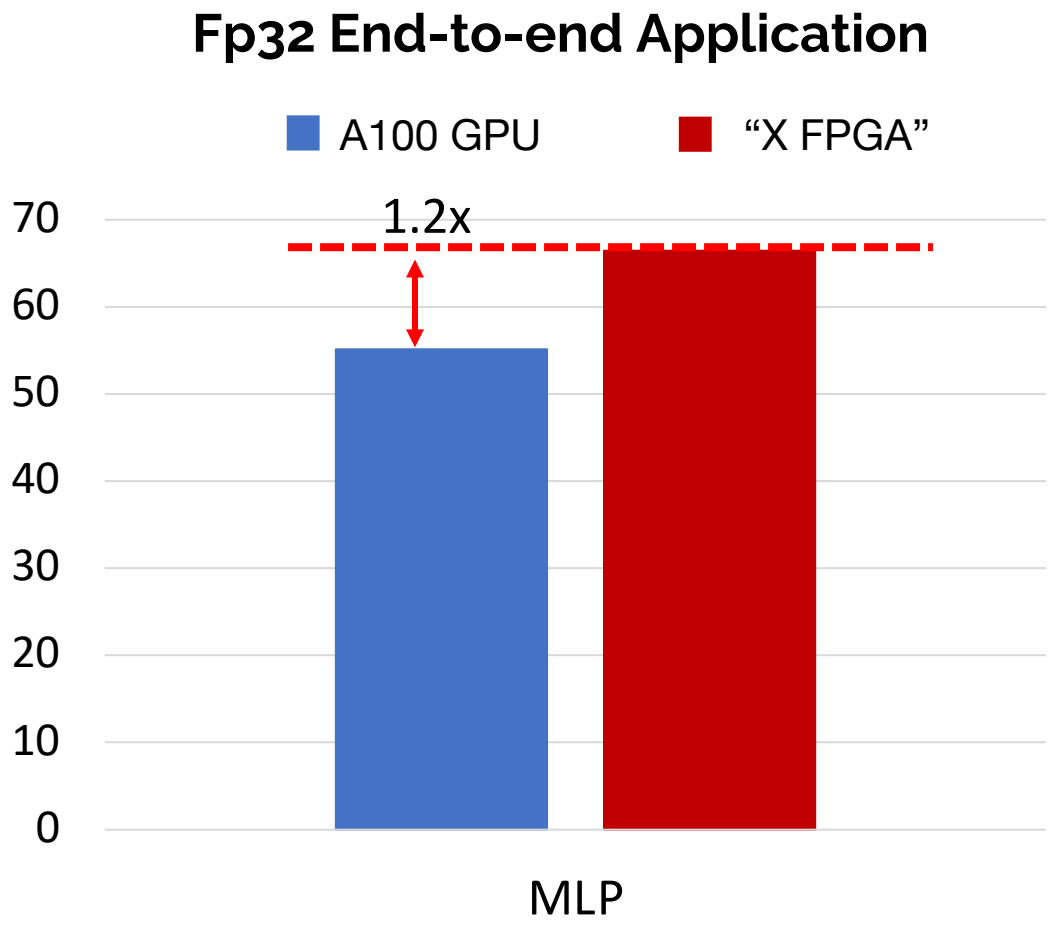


“FPGA is Dead” vs “Long Live the FPGA”

- **Single GEMM Energy Efficiency Comparisons**

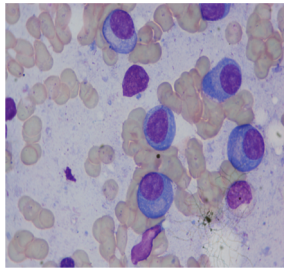


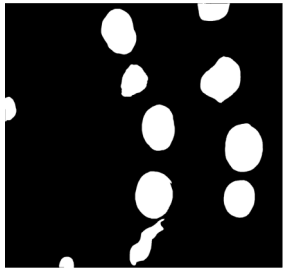
- **End-to-end Application**



“X FPGA”: AMD/Xilinx Versal ACAP

Health: Medical Image Segmentation





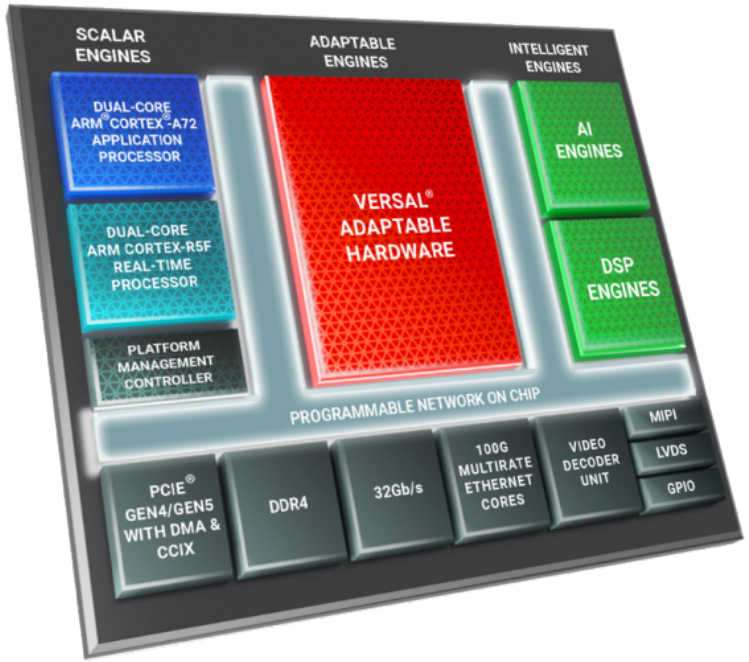
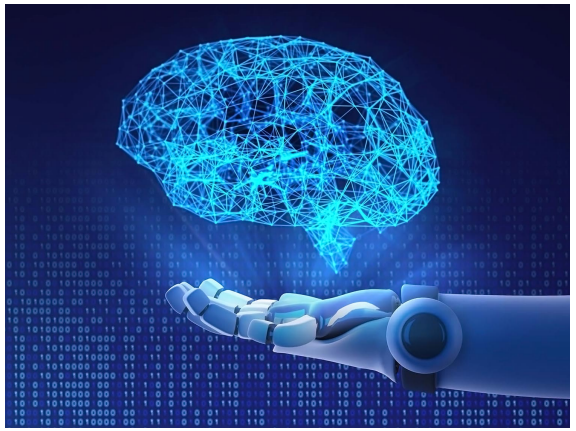
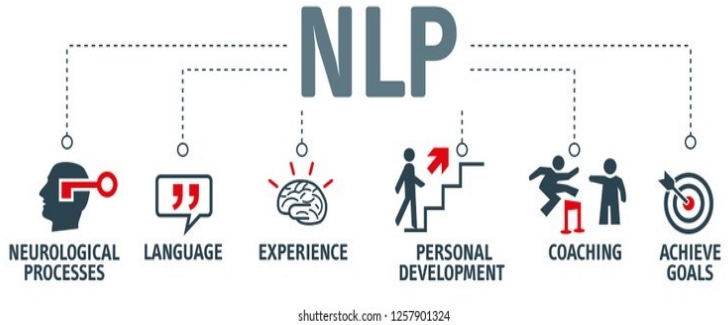


Image Recognition

Recommendation Systems

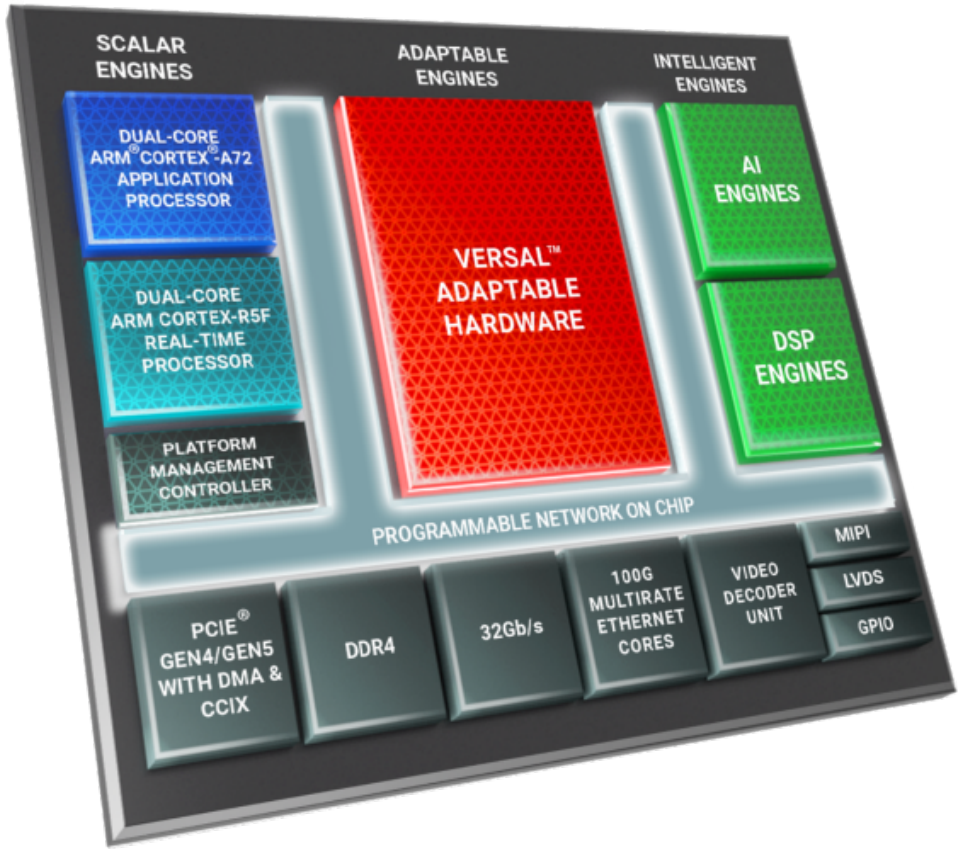


Natural Language processing



Background Story

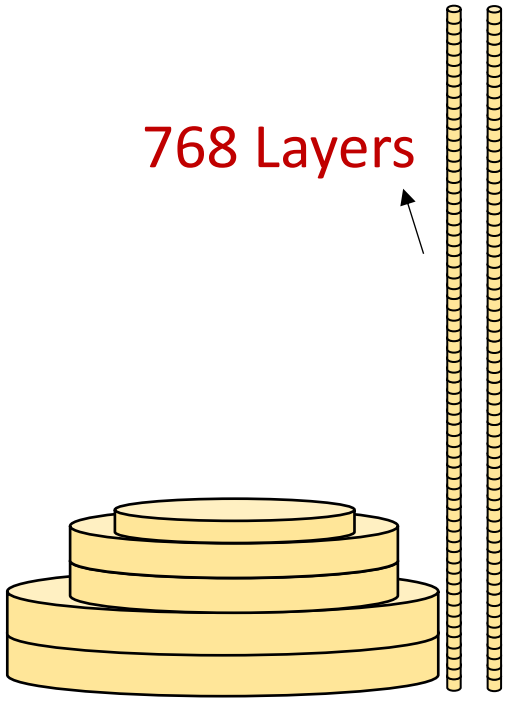
Versal ACAP Overview



After CHARM: 1609 GFLOPS

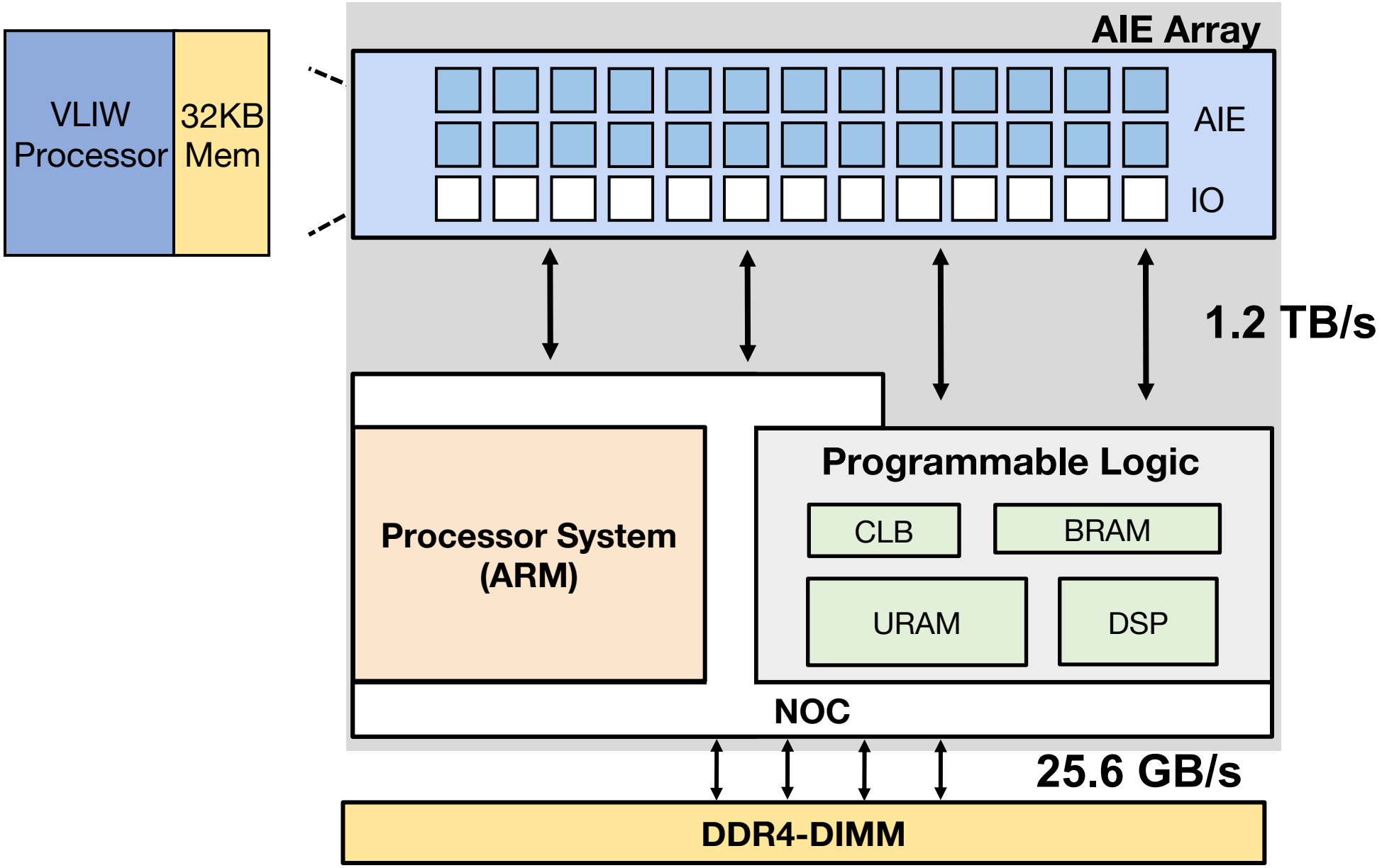
32.5x

Before CHARM: 49.5 GFLOPS



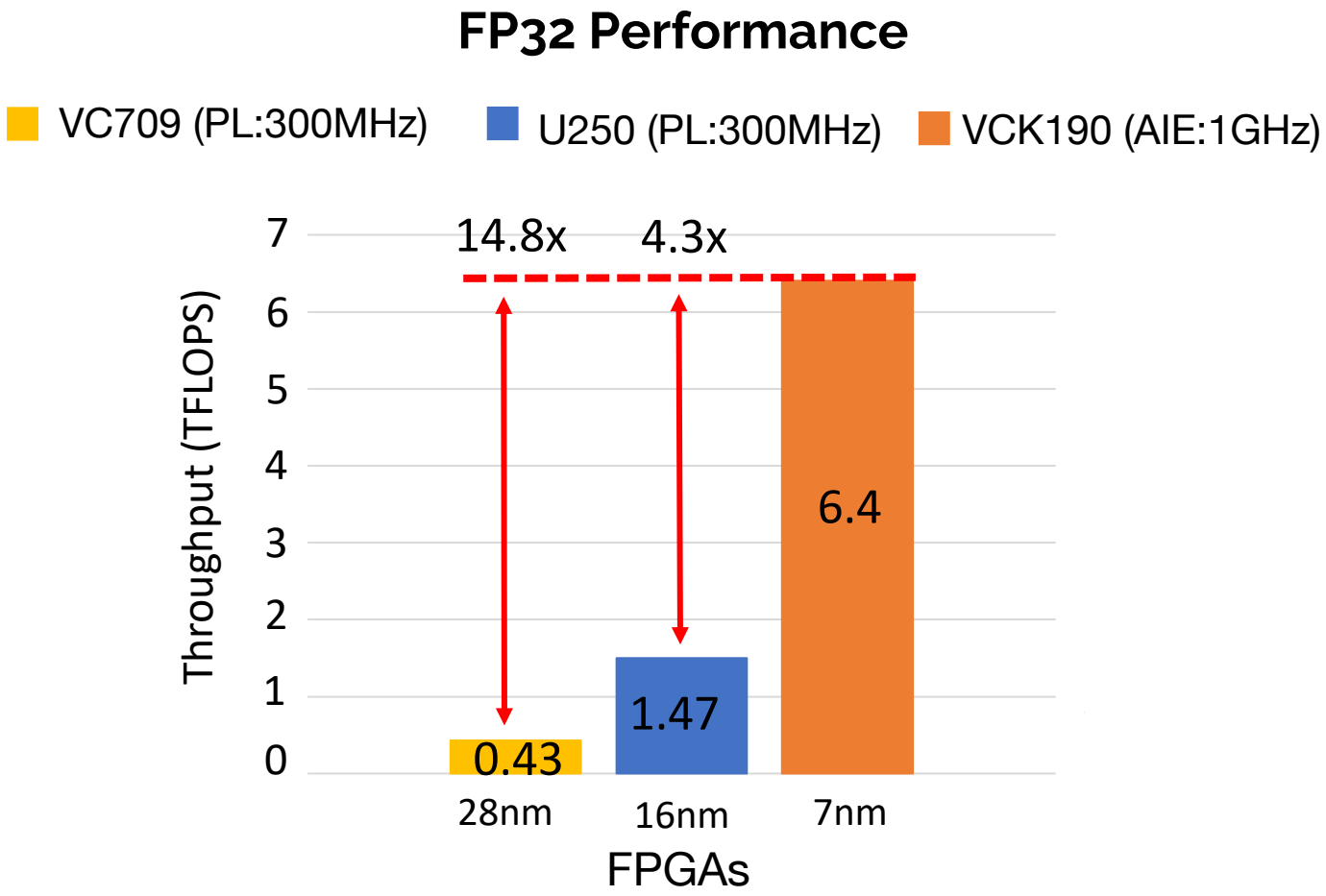
ViT

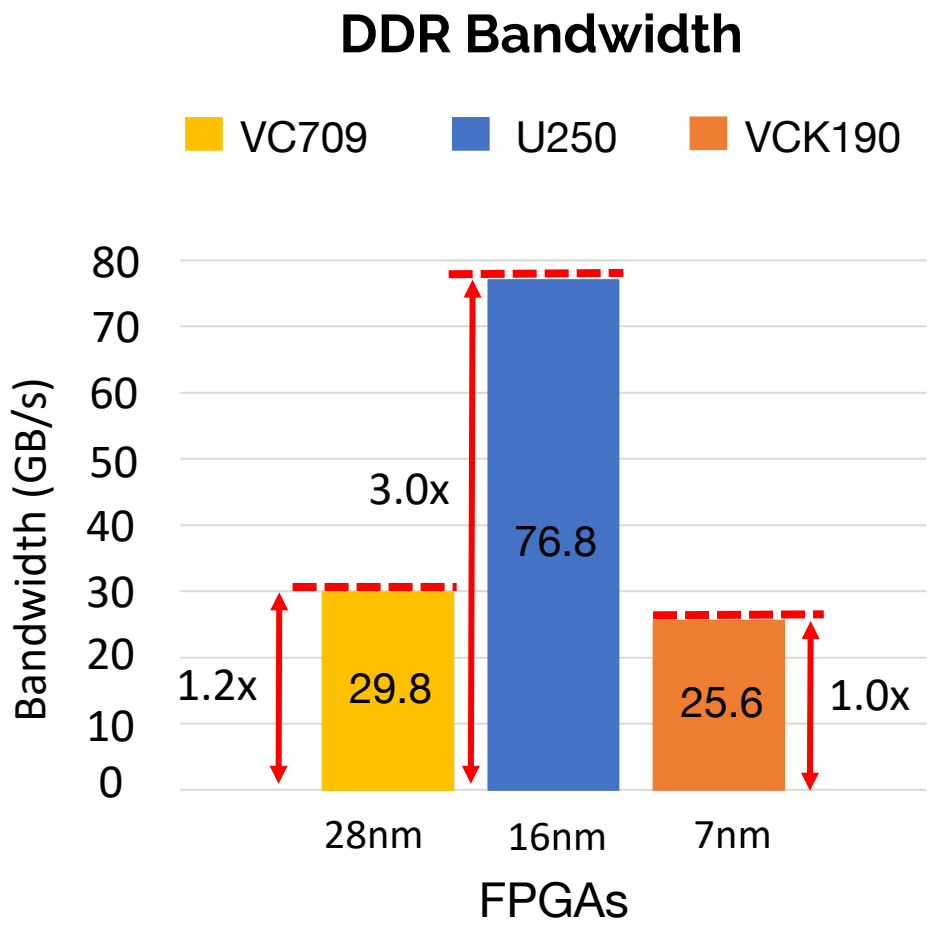
Versal ACAP Architecture



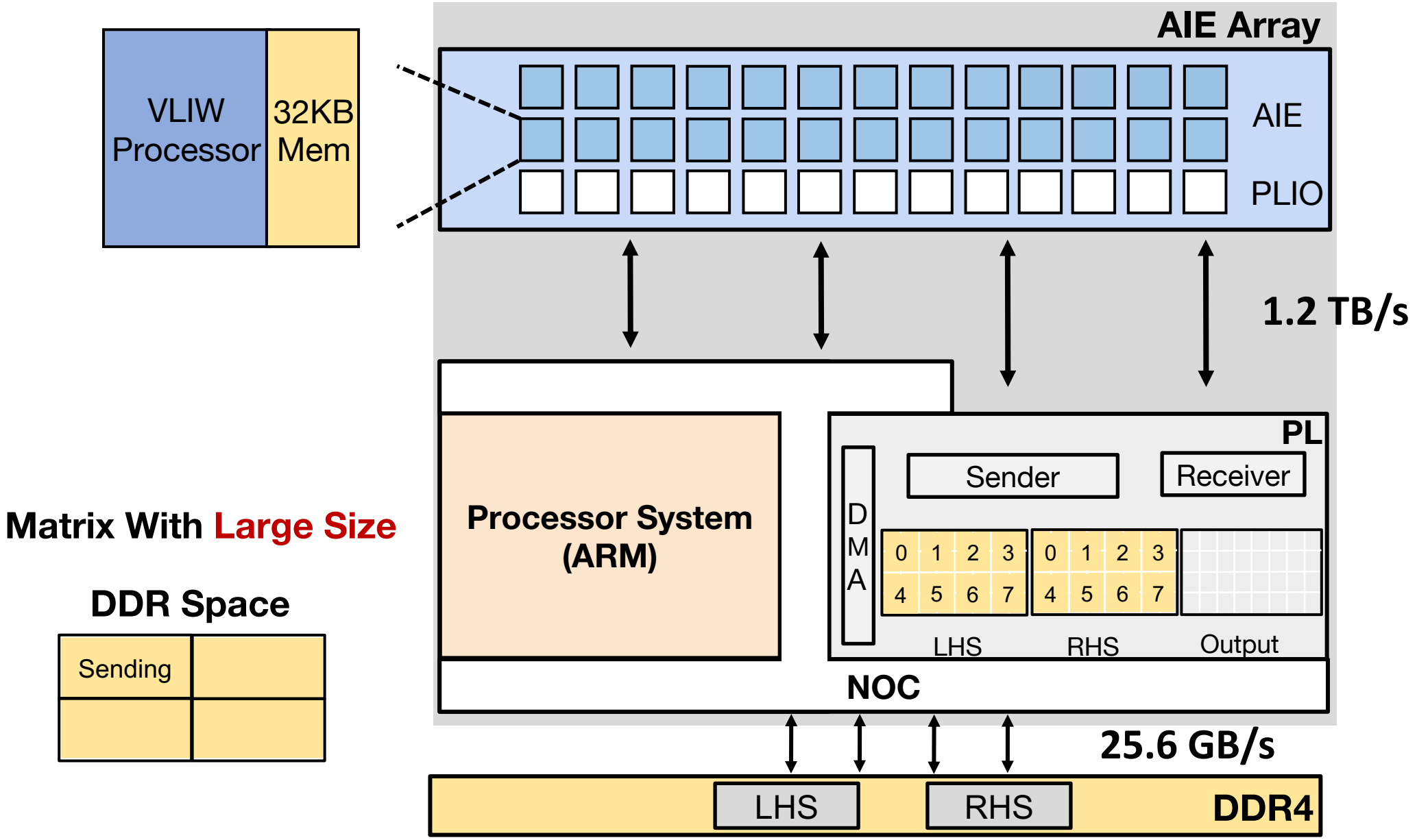
Challenge 1

- The computation scaling vs. off-chip communication scaling

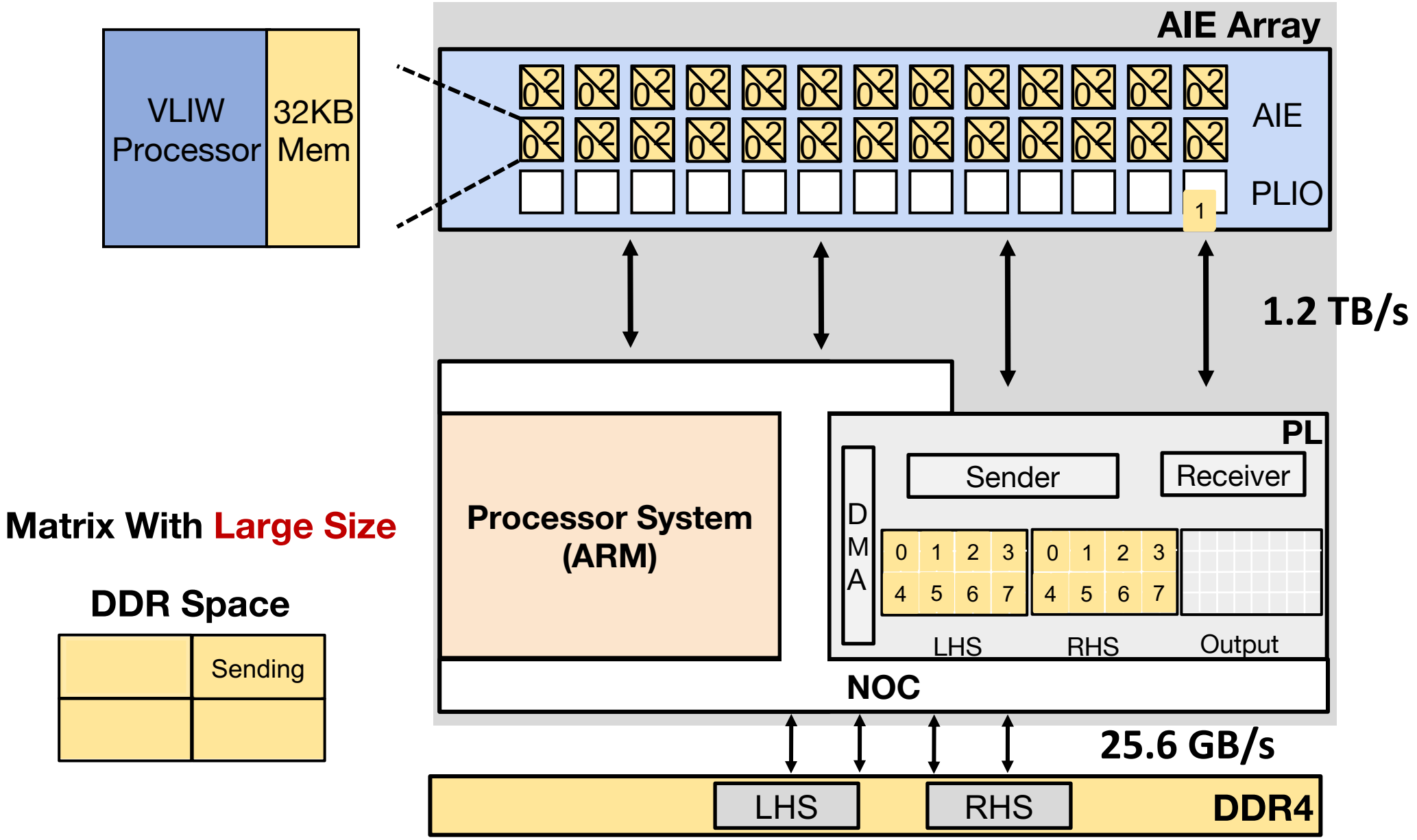




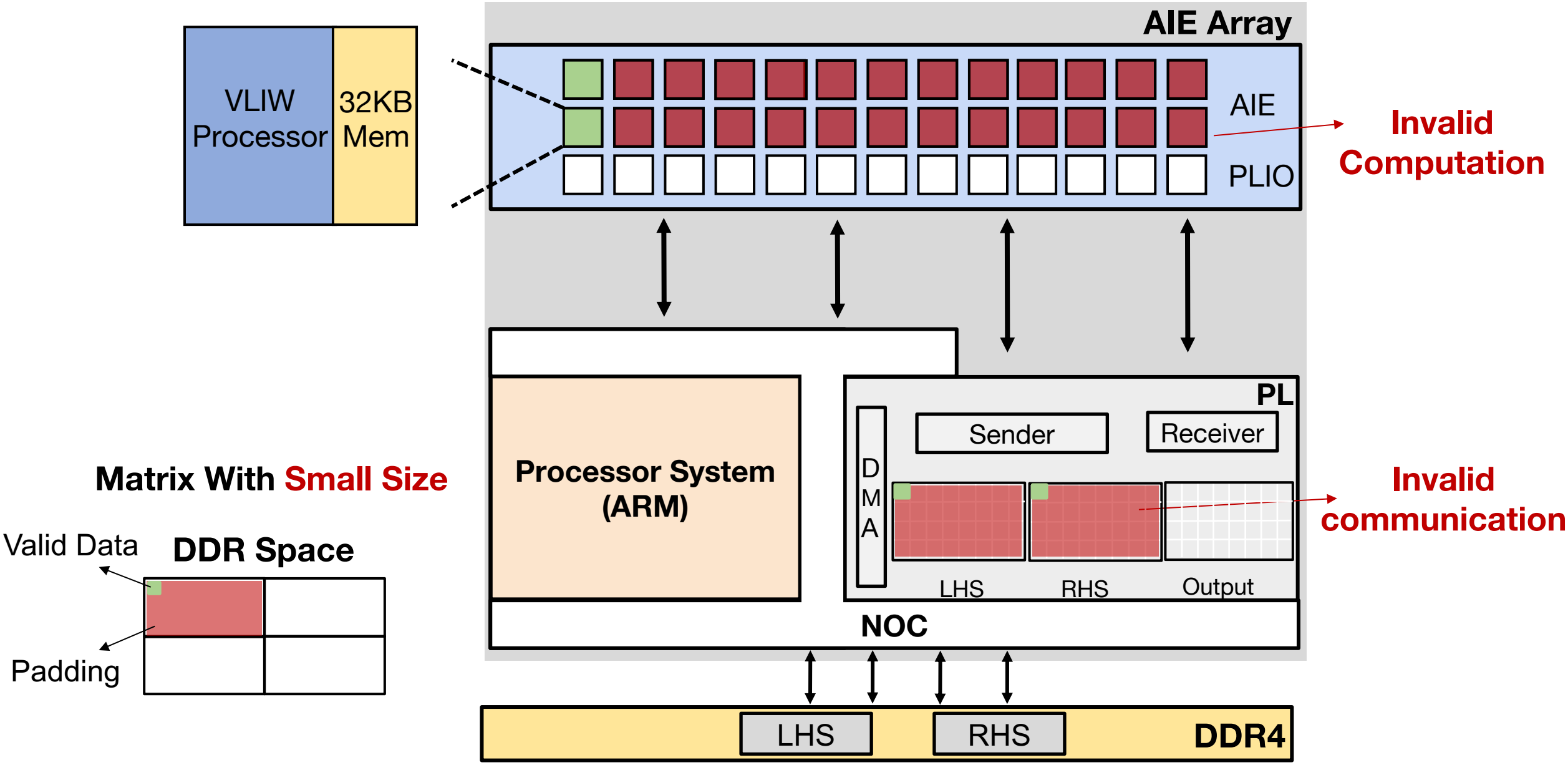
Versal Programming Model



Versal Programming Model



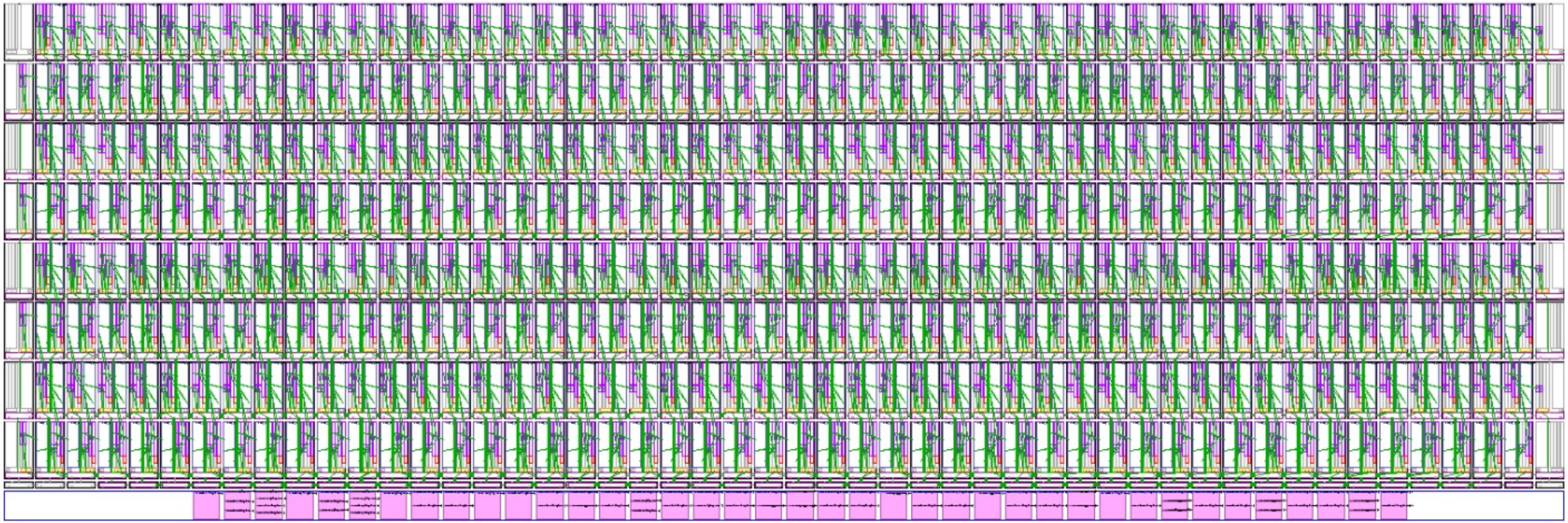
Versal Programming Model



Challenge 2

- **Huge computation resources in one monolithic accelerator** vs. **MM layers of small sizes**

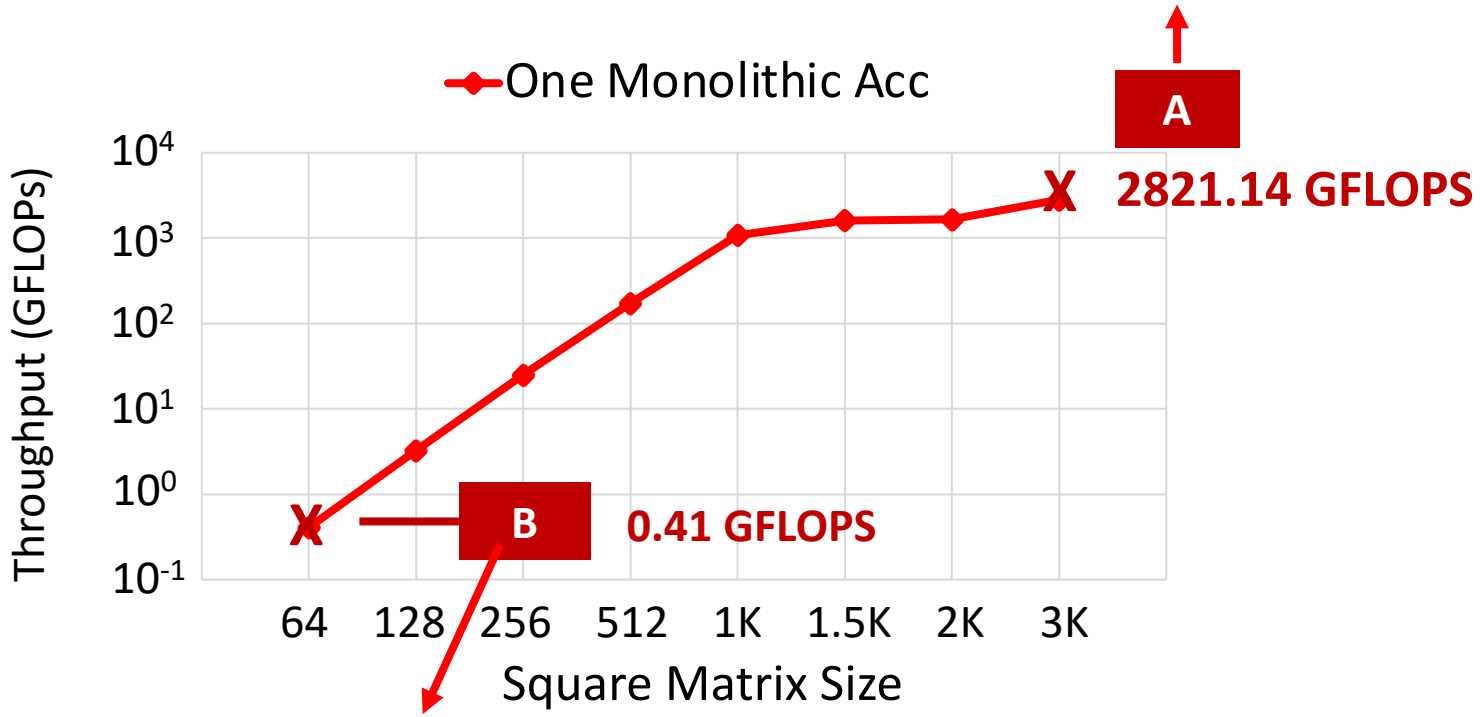
  Matrix Multiply on 384 AIEs



Challenge 2

- **Huge computation resources in one monolithic accelerator** vs. **MM layers of small sizes**

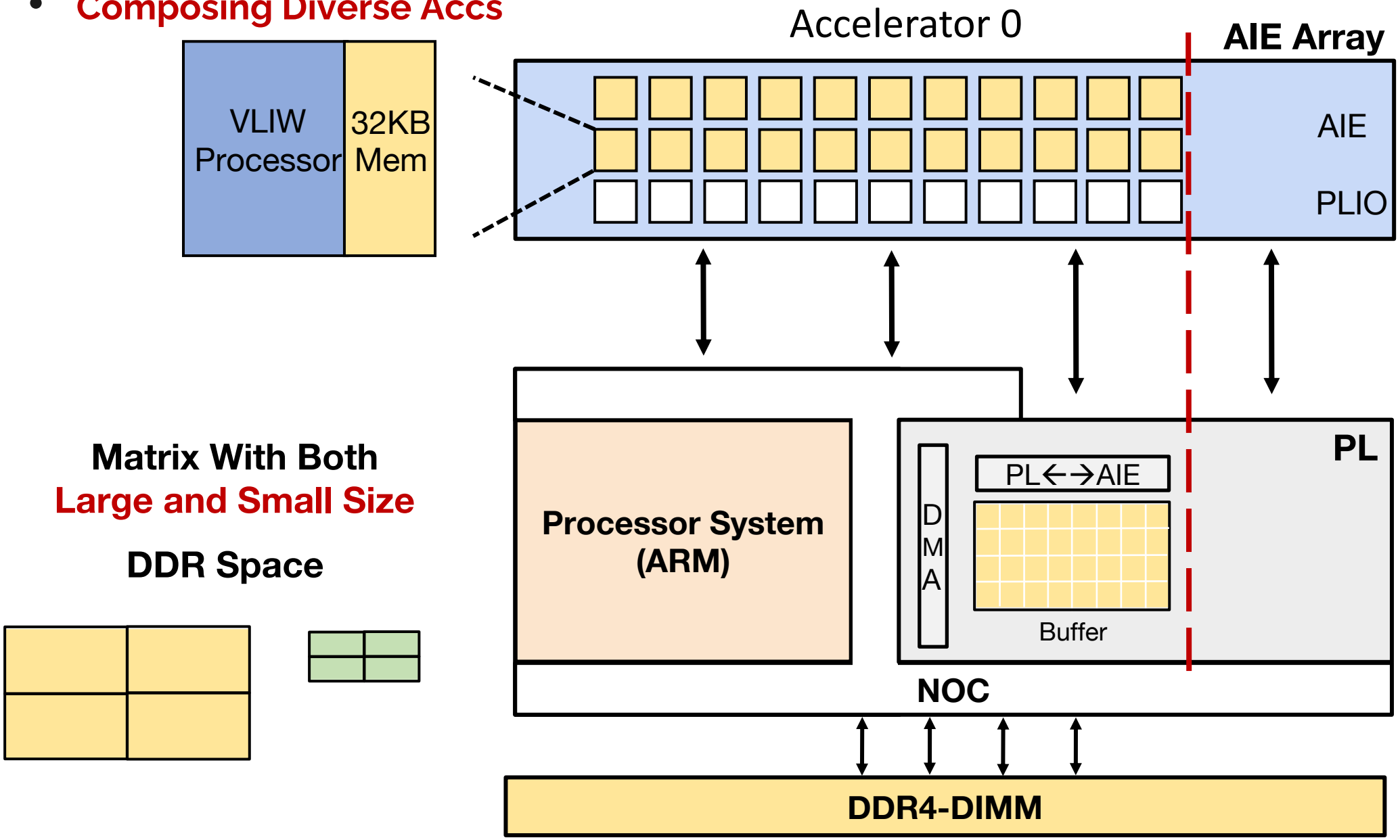
Explore data reuse by allocating large on-chip buffers



Shape mismatch, waste on both computation and communication

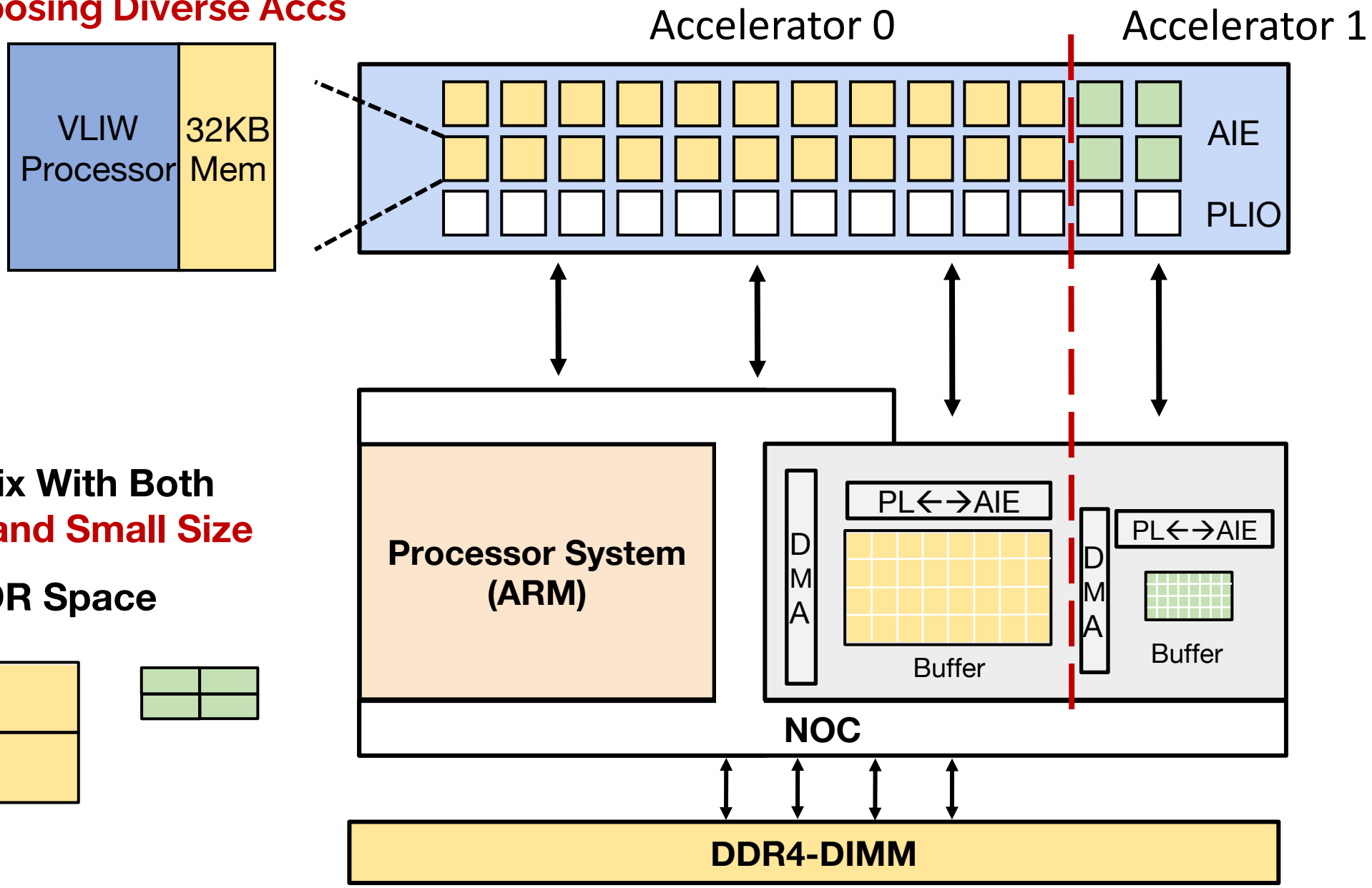
Motivation Example

- **Composing Diverse Accs**



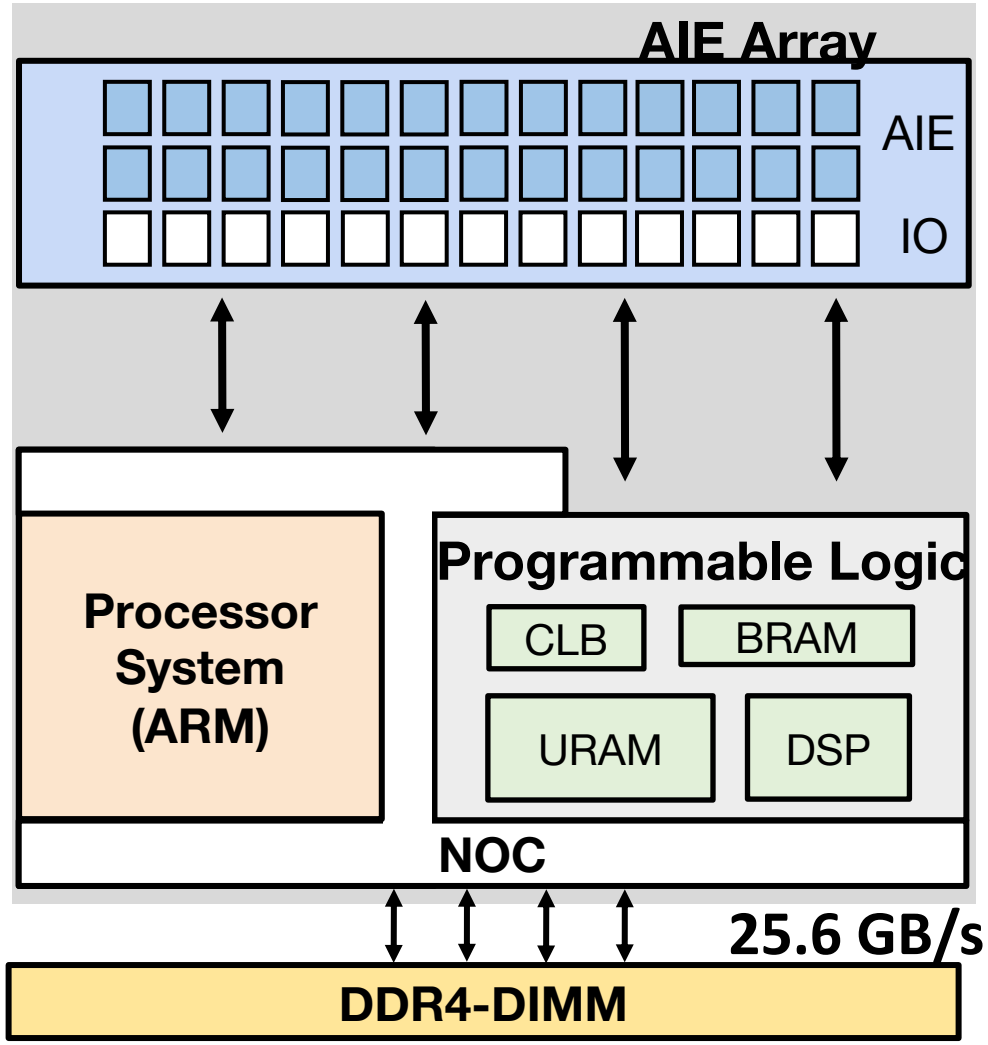
Motivation Example

- **Composing Diverse Accs**



Challenge 3

High level complexity



→ **Huge Programming Efforts**

AIE C/C++	HLS C/C++	Host C/C++
900 LOC	1000 LOC	300 LOC

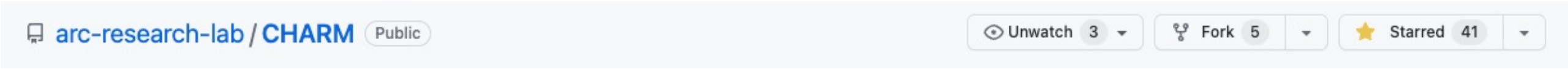
“CHARM: Composing Heterogeneous Accelerators for Matrix Multiply on Versal ACAP Architecture”

Challenges

- Heterogeneity makes it non-trivial to make a system design with good performance on Versal
- Off-chip bandwidth doesn’t scaling as fast as computation
- High parallelism makes one-size-fit-for-all solution inefficient

Solutions

- White-box open-sourced Framework
  https://github.com/arc-research-lab/CHARM
- Hugely explore the on-chip reuse
- Two-step Accelerator Composing Algorithm

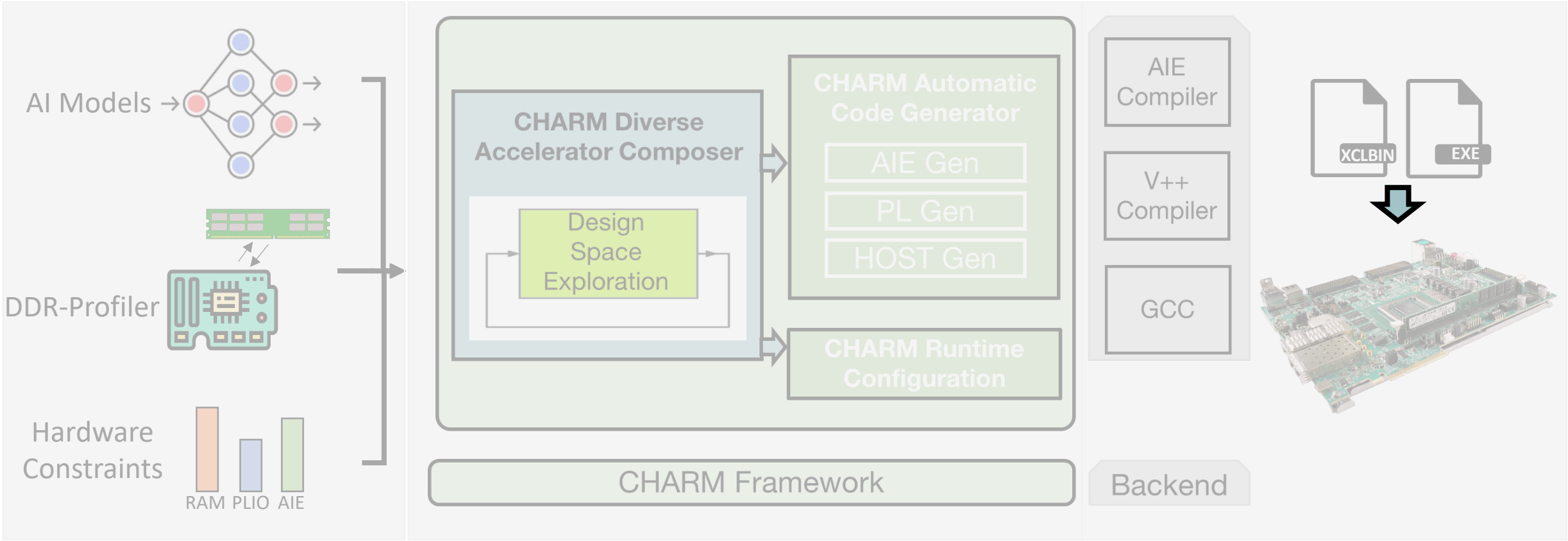


CHARM Compilation Flow

CHARM Input and Output (IOP)

- **Input**
  1) MM-based AI Model ;
  2) Profiling of Off-chip Bandwidth ;
  3) Hardware resource constrains ;

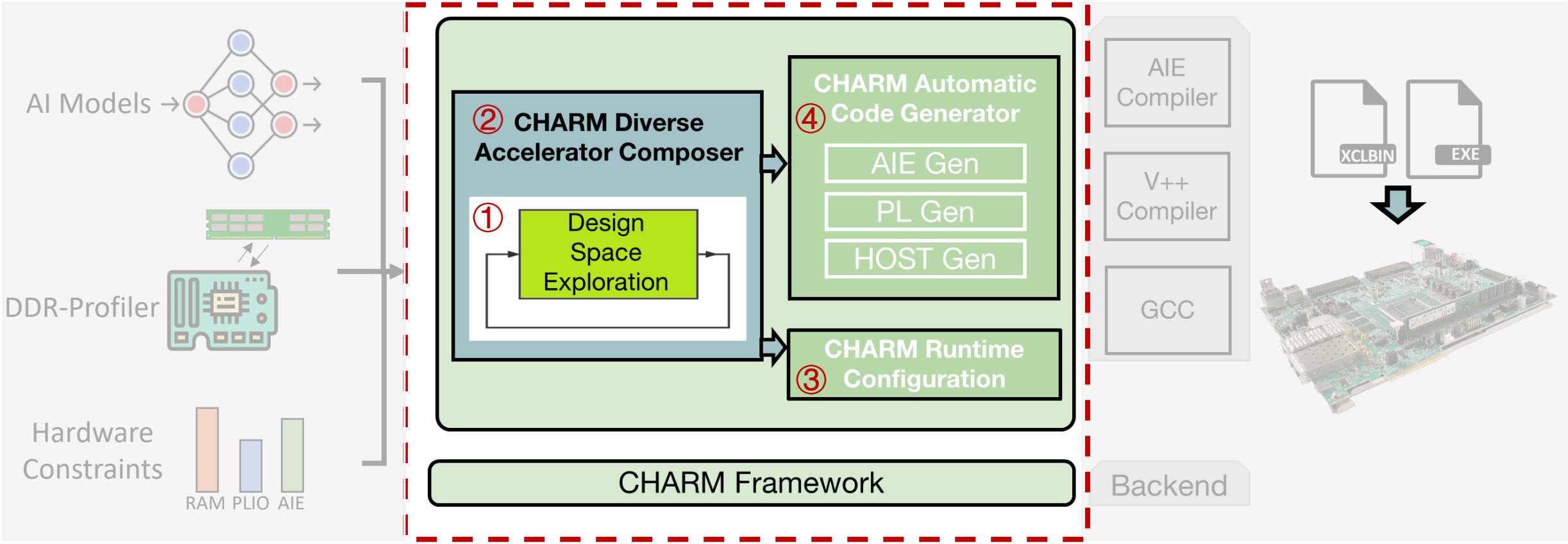
- **Output**
  1) Bitstream Running on the AIE and PL (AIE/V++ Compiler)
  2) Executable Running on ARM CPU (GCC Cross Compilation)



CHARM Framework Overview

CHARM Components

① **Single Accelerator Design Space Exploration**
② **Diverse Accelerator Composer**
③ **Runtime Configuration**
④ **Automatic Code Generator**



CHARM Framework Overview

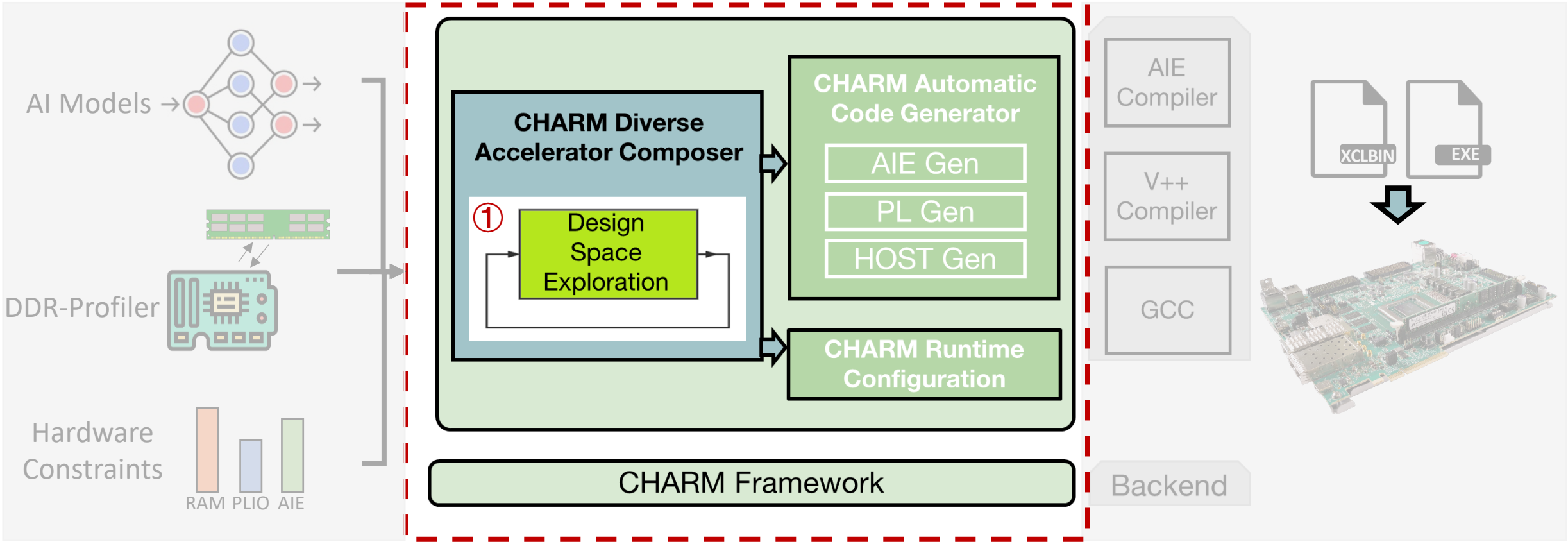
CHARM Components

① Single Accelerator Design Space Exploration

1) Fully-pipelined AIE Processor Design

2) Hugely IO Reused AIE Array Design

3) On-chip Buffer Reused PL Design



CHARM Framework Components

① Single Accelerator Design Space Exploration

1) Fully-pipelined AIE Processor Design

```
#define TI 32
#define TK 32
#define TJ 32
```

3 Lines of Code

```
void mm_kernel(
    input_window_float * restrict L,// LHS
    input_window_float * restrict R, // RHS
    output_window_float * restrict O ) { // Output
    preload(L,R);
    for(int m.3 = 0; m.3 < TI/PI; m.3++)
    chess_prepare_for_pipelining
    chess_loop_range(TI/PI, ) {
        for(int n.3 = 0; n.3 < TJ/PJ; n.3++)
        chess_prepare_for_pipelining
        chess_loop_range(TJ/PJ, ) {
            v8float acc0 = null_v8float();
            v8float acc1 = null_v8float();
            for(int k.3 = 0; k.3 < TK/PK - 1; k.3++)
            chess_prepare_for_pipelining
            chess_loop_range(TK/PK - 1, ) {
                [acc0; acc1] = MatMul_without_store(
                    L(m.3, k.3), R(k.3, j.3), [acc0; acc1]);
            }
            MatMul_with_store(L(m.3, TK/PK - 1), R(TK/PK - 1,
                              j.3), [acc0; acc1], O(i, j));
        }
    }
}
```

>100 Lines of Code

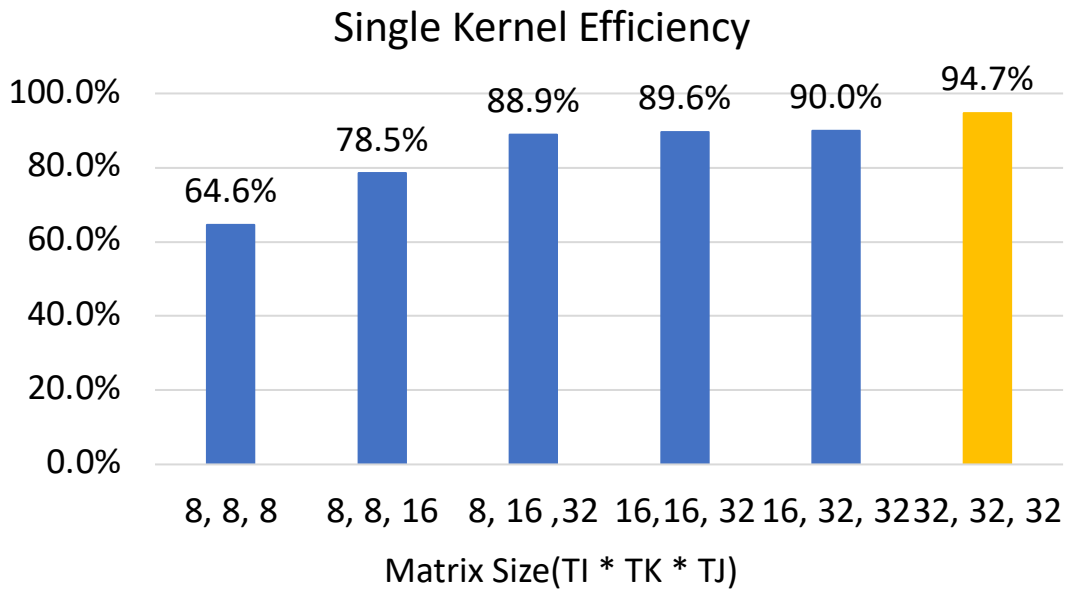


Table 2: Single AIE MM comparison under fp32 data type.

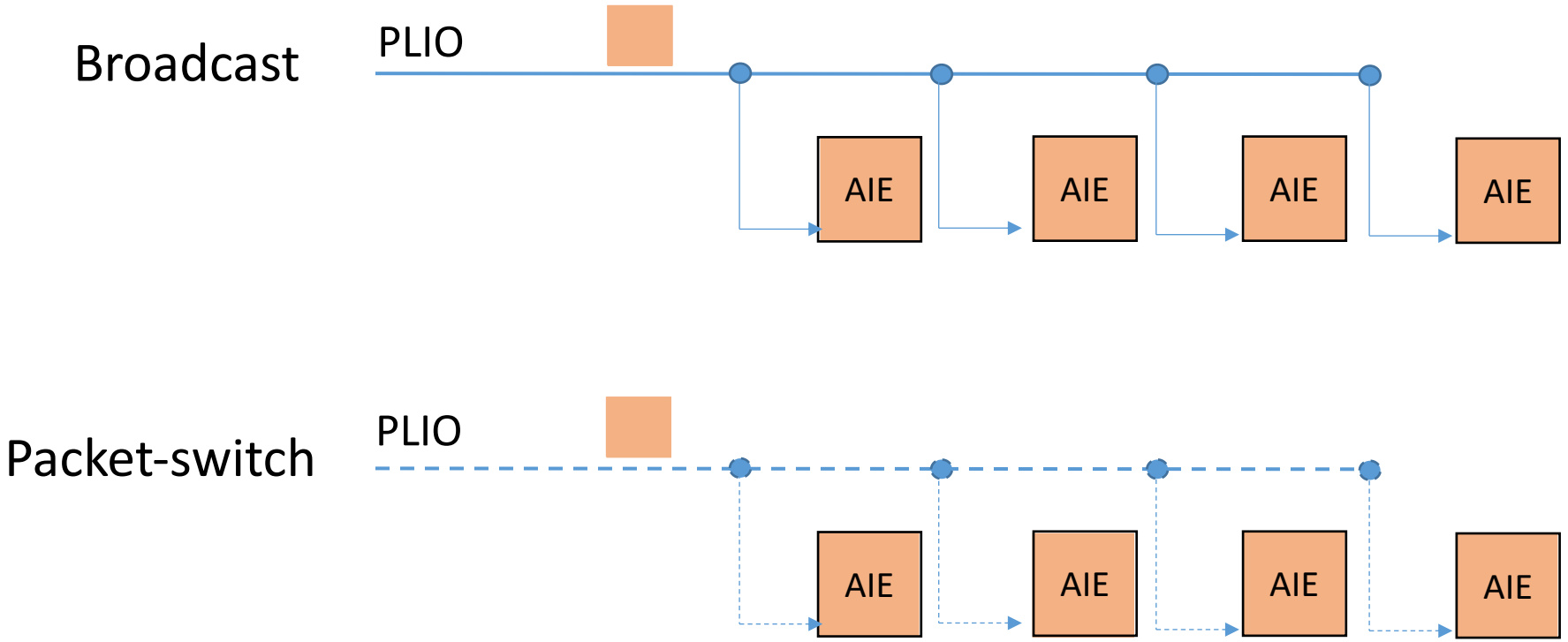
Size: M x K x N	H-GCN[48] MACs/Cyc	H-GCN[48] Eff	CHARM (this work) MACs/Cyc	CHARM (this work) Eff	CHARM (this work) Eff gain
16 × 16 × 16	2.34	29.30%	6.18	77.22%	***2.64x***
32 × 32 × 32	3.64	45.50%	7.57	94.70%	***2.08x***
64 × 64 × 8	3.64	45.50%	7.54	94.29%	***2.07x***

[48] Zhang, Chengming, et al. "H-GCN: A graph convolutional network accelerator on Versal ACAP architecture." FPL'22.

CHARM Framework Components

① Single Accelerator Design Space Exploration

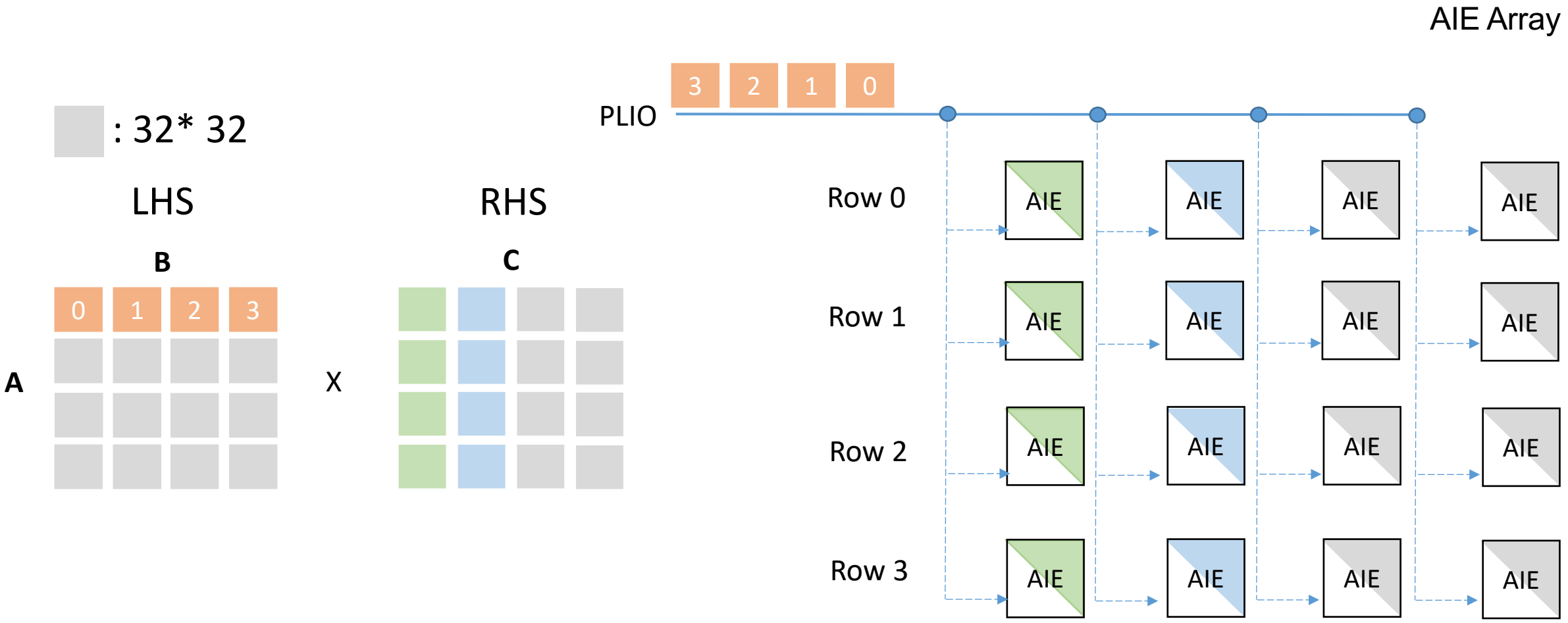
2) Hugely IO Reused AIE Array Design : Broadcast-Packet switch Connection



CHARM Framework Components

① Single Accelerator Design Space Exploration

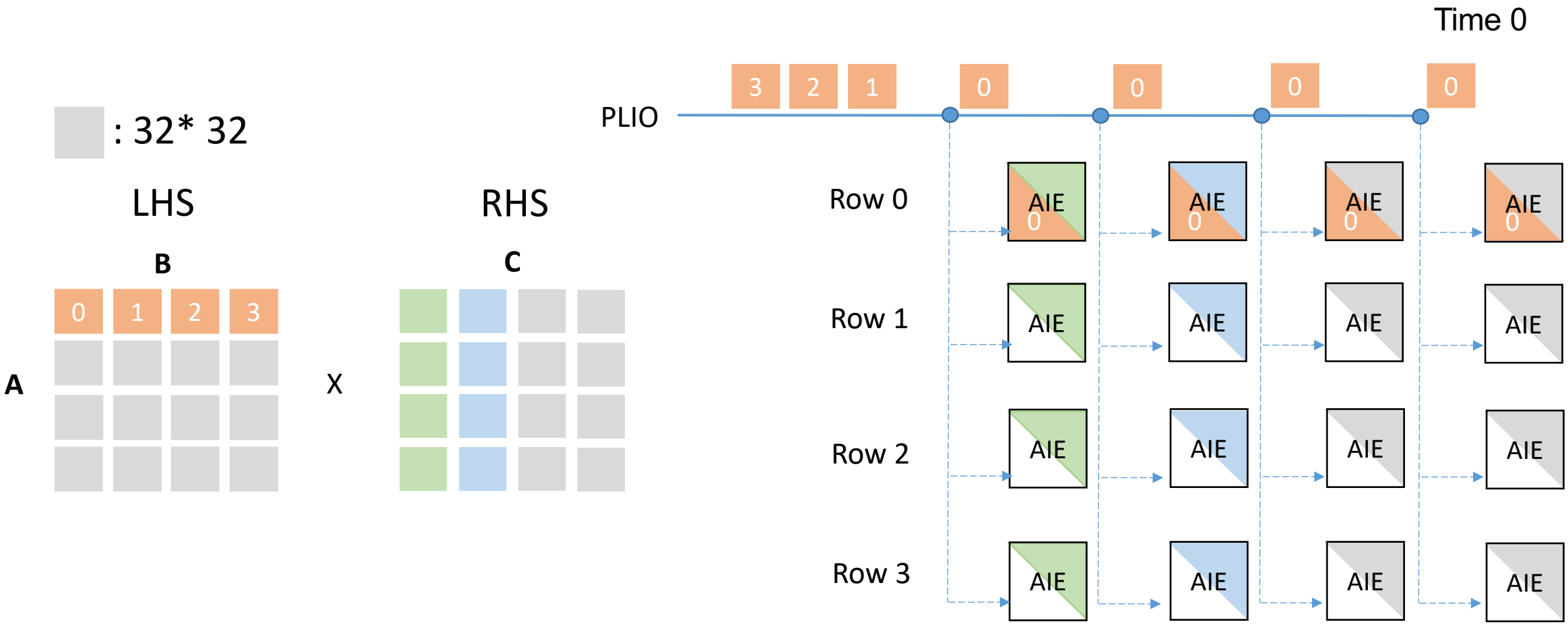
2) Hugely IO Reused AIE Array Design : Broadcast-Packet switch Connection



CHARM Framework Components

① Single Accelerator Design Space Exploration

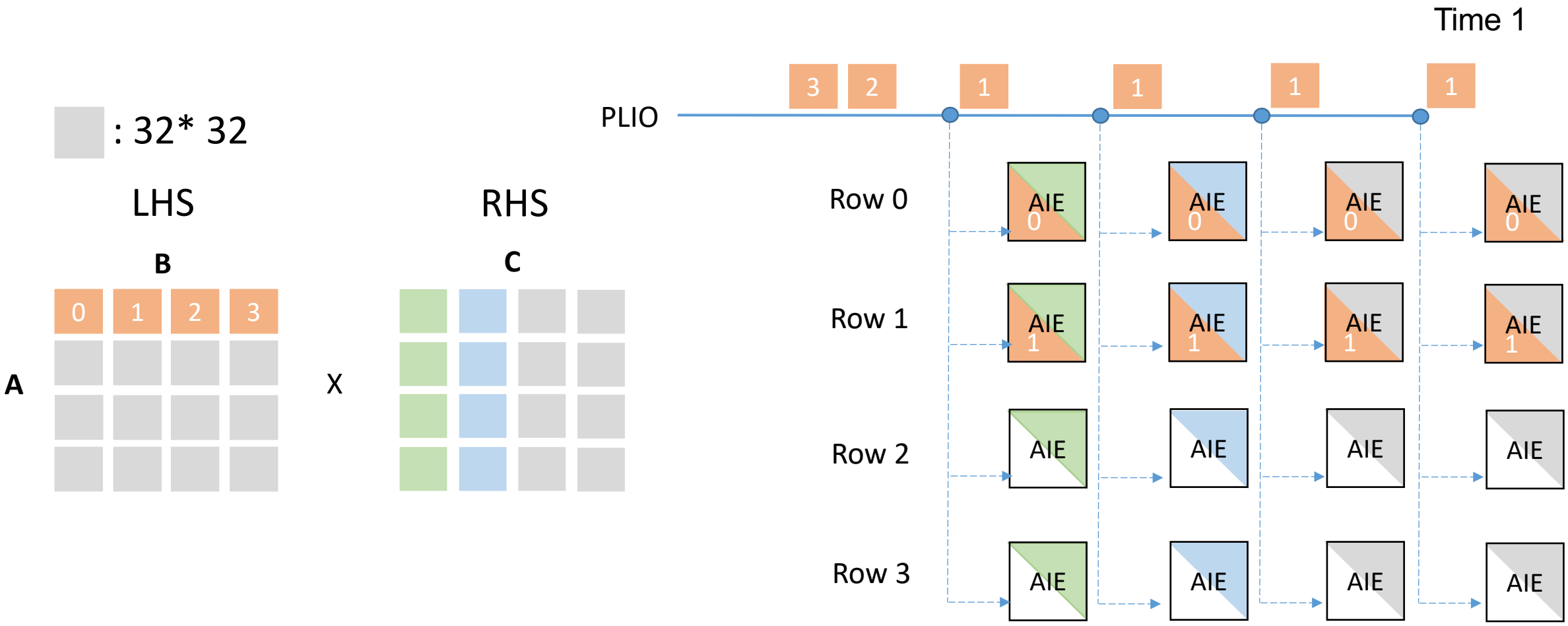
2) Hugely IO Reused AIE Array Design : Broadcast-Packet switch Connection



CHARM Framework Components

① Single Accelerator Design Space Exploration

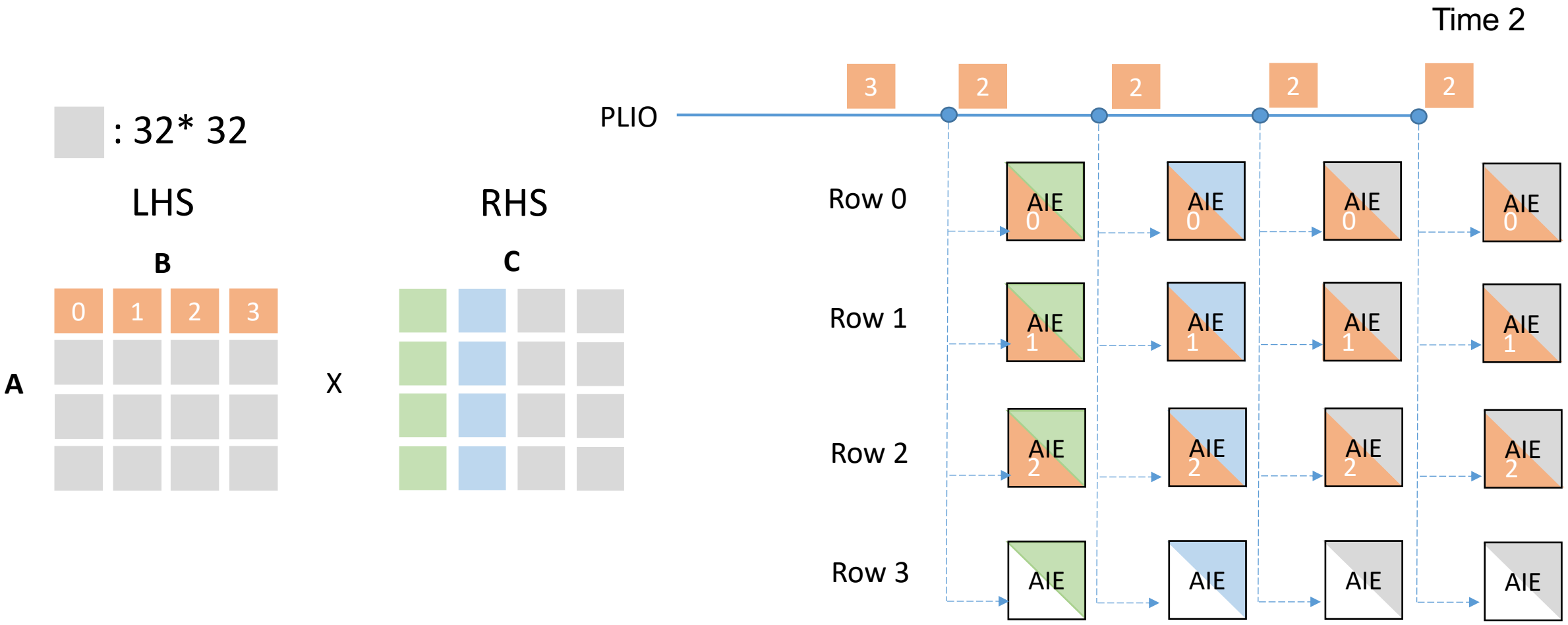
2) Hugely IO Reused AIE Array Design : Broadcast-Packet switch Connection



CHARM Framework Components

① Single Accelerator Design Space Exploration

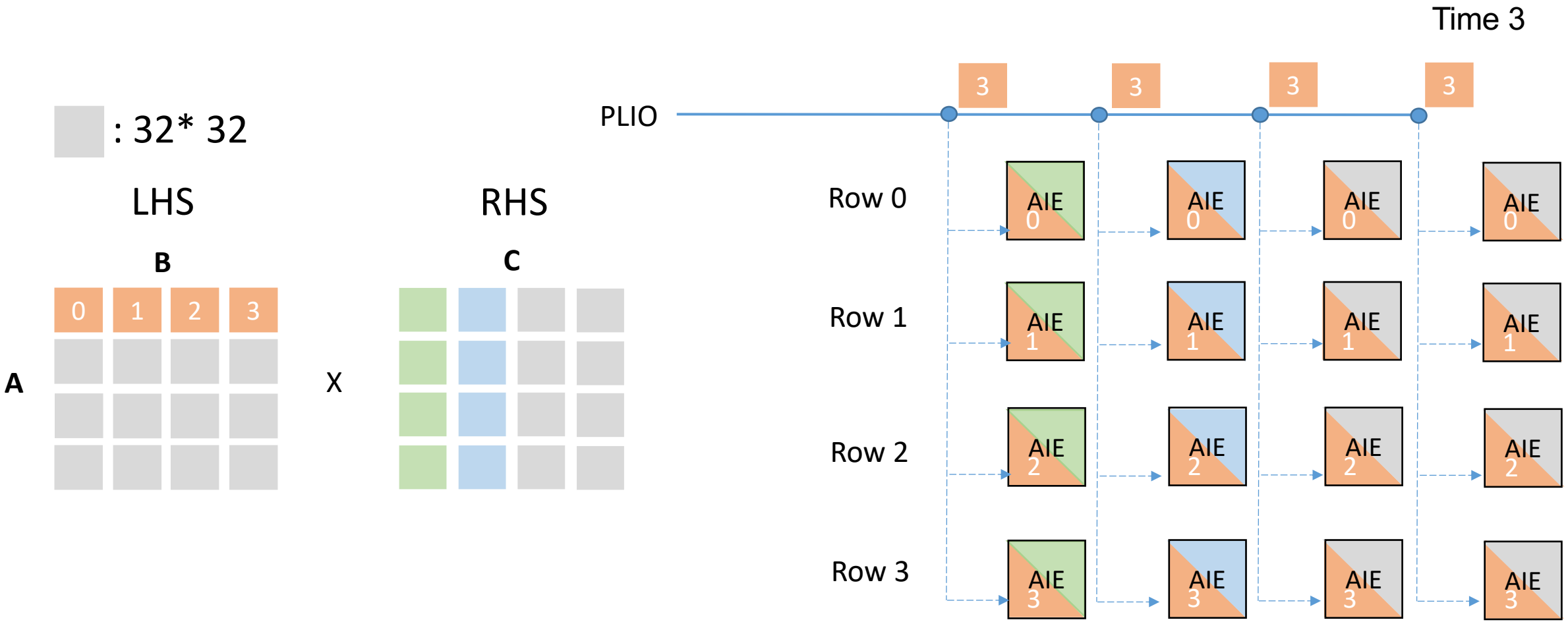
2) Hugely IO Reused AIE Array Design : Broadcast-Packet switch Connection



CHARM Framework Components

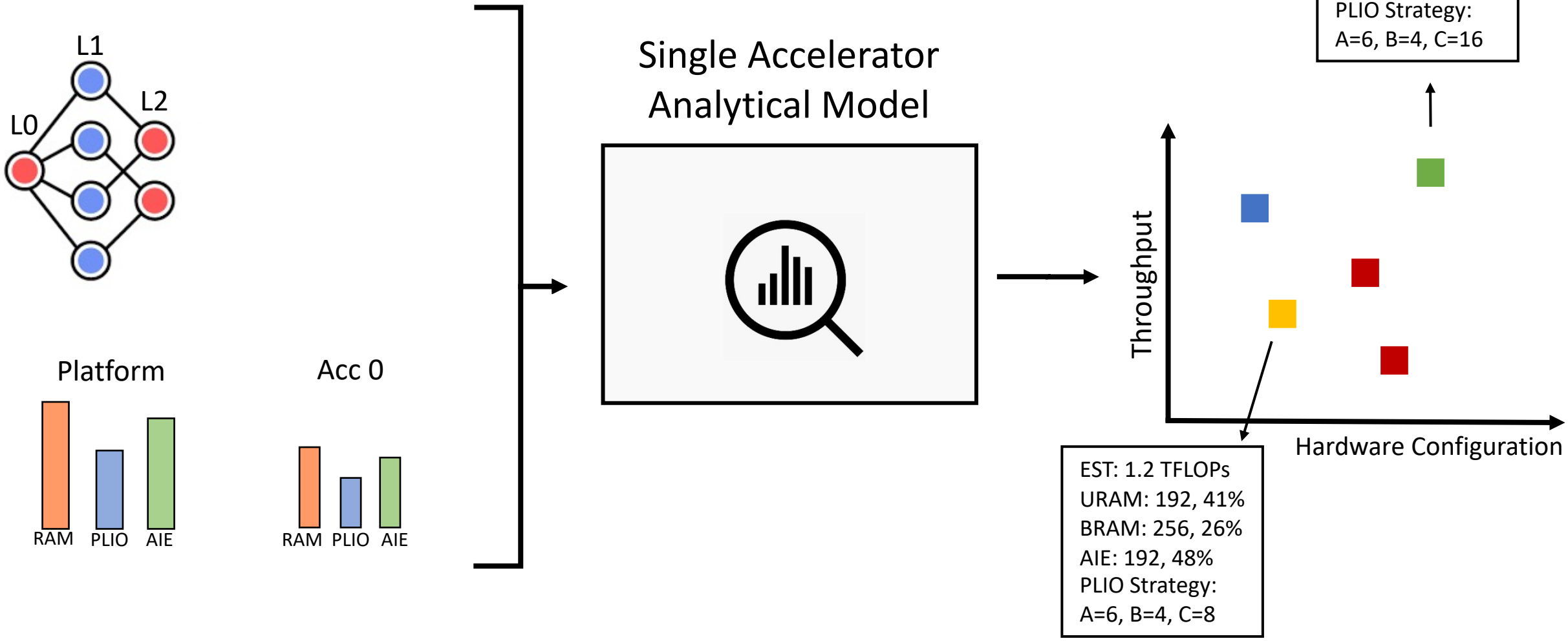
① Single Accelerator Design Space Exploration

2) Hugely IO Reused AIE Array Design : Broadcast-Packet switch Connection



CHARM Framework Components

① Single Accelerator Design Space Exploration



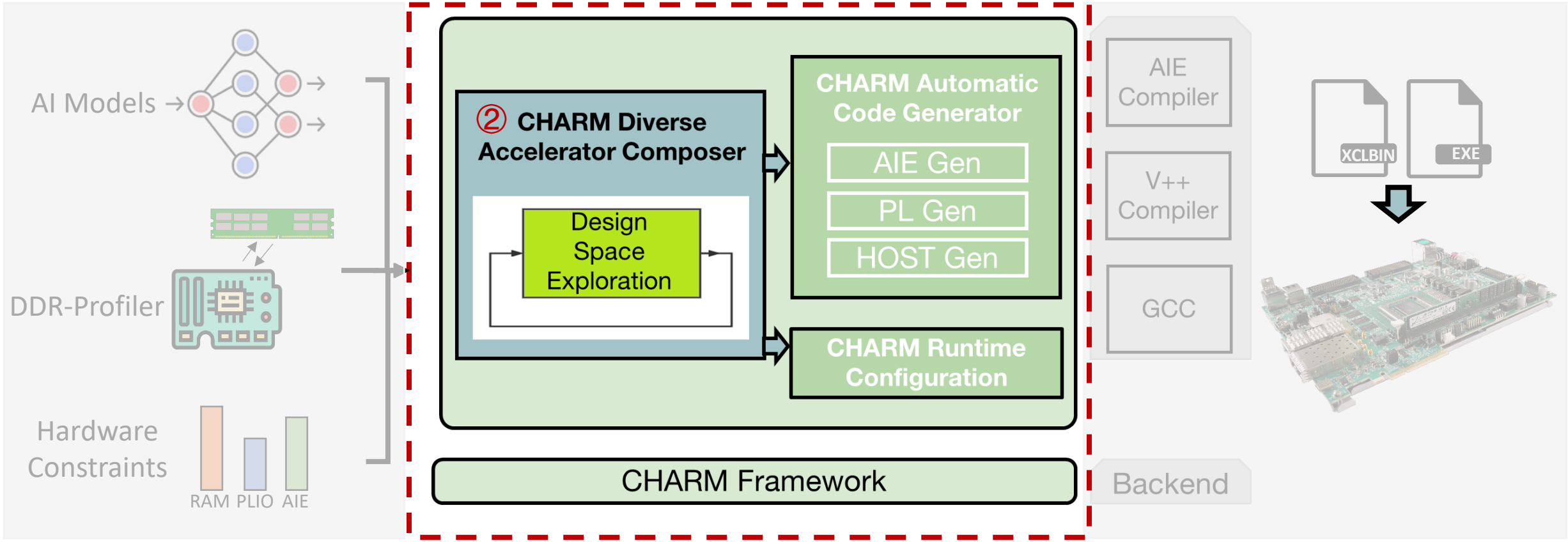
CHARM Framework Overview

CHARM Components

② **Diverse Accelerator Composer**

1) **Workload Assignment**

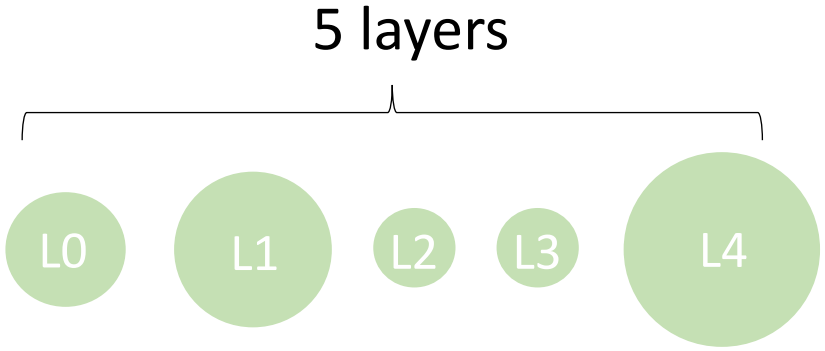
2) **Hardware Resource Partitioning**

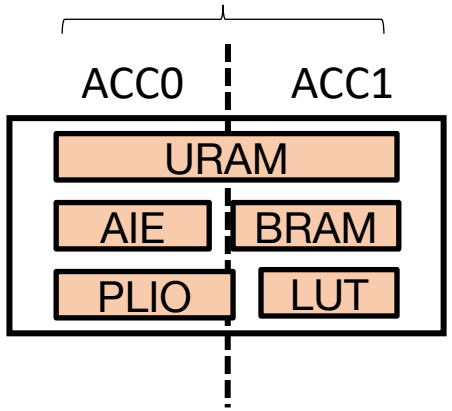


CHARM: Diverse Accelerator Composing

- **Two-step Search Algorithm**

How to assign **M layers** in one AI Model to **N accelerators**?





1st Step: Workload Assignment

2nd Step: Hardware Resource Partitioning

CHARM: Diverse Accelerator Composing

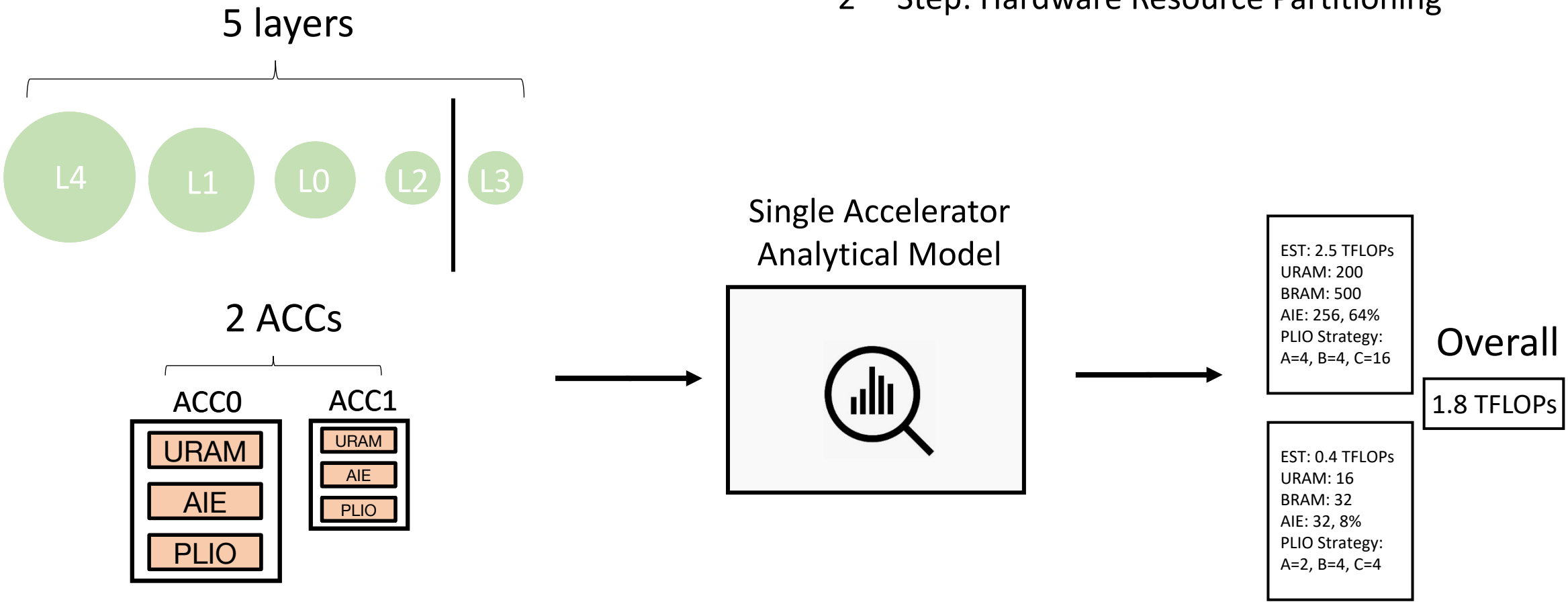
- **Two-step Search Algorithm**

How to assign **M layers** in one AI Model to **N accelerators**?

① ②

1st Step: Workload Assignment

2nd Step: Hardware Resource Partitioning



CHARM: Diverse Accelerator Composing

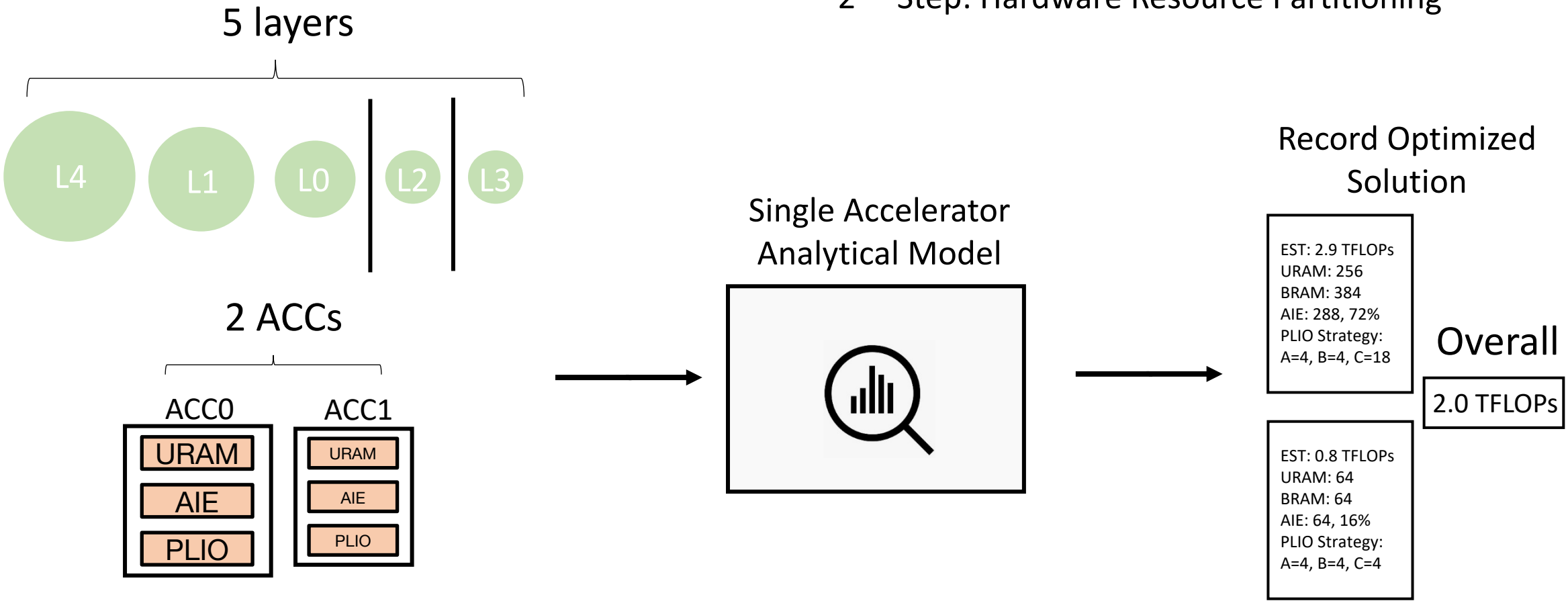
- **Two-step Search Algorithm**

How to assign **M layers** in one AI Model to **N accelerators**?

① ②

1st Step: Workload Assignment

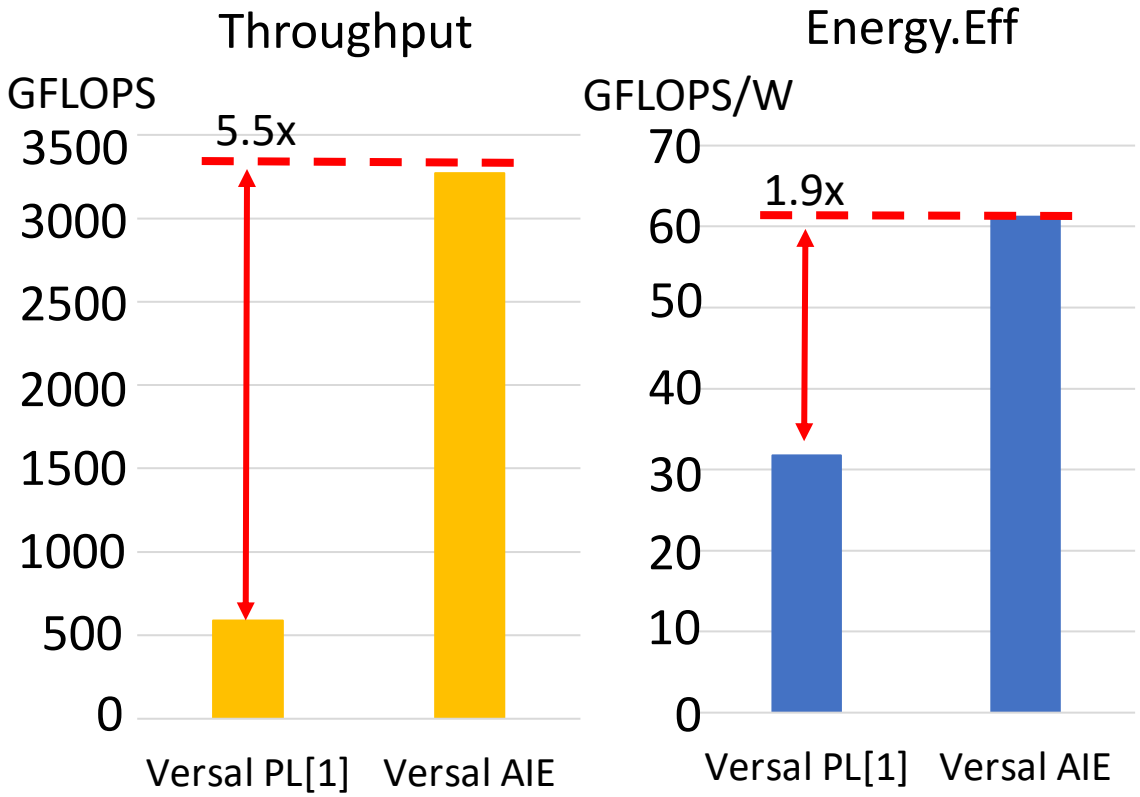
2nd Step: Hardware Resource Partitioning



Experiment Results

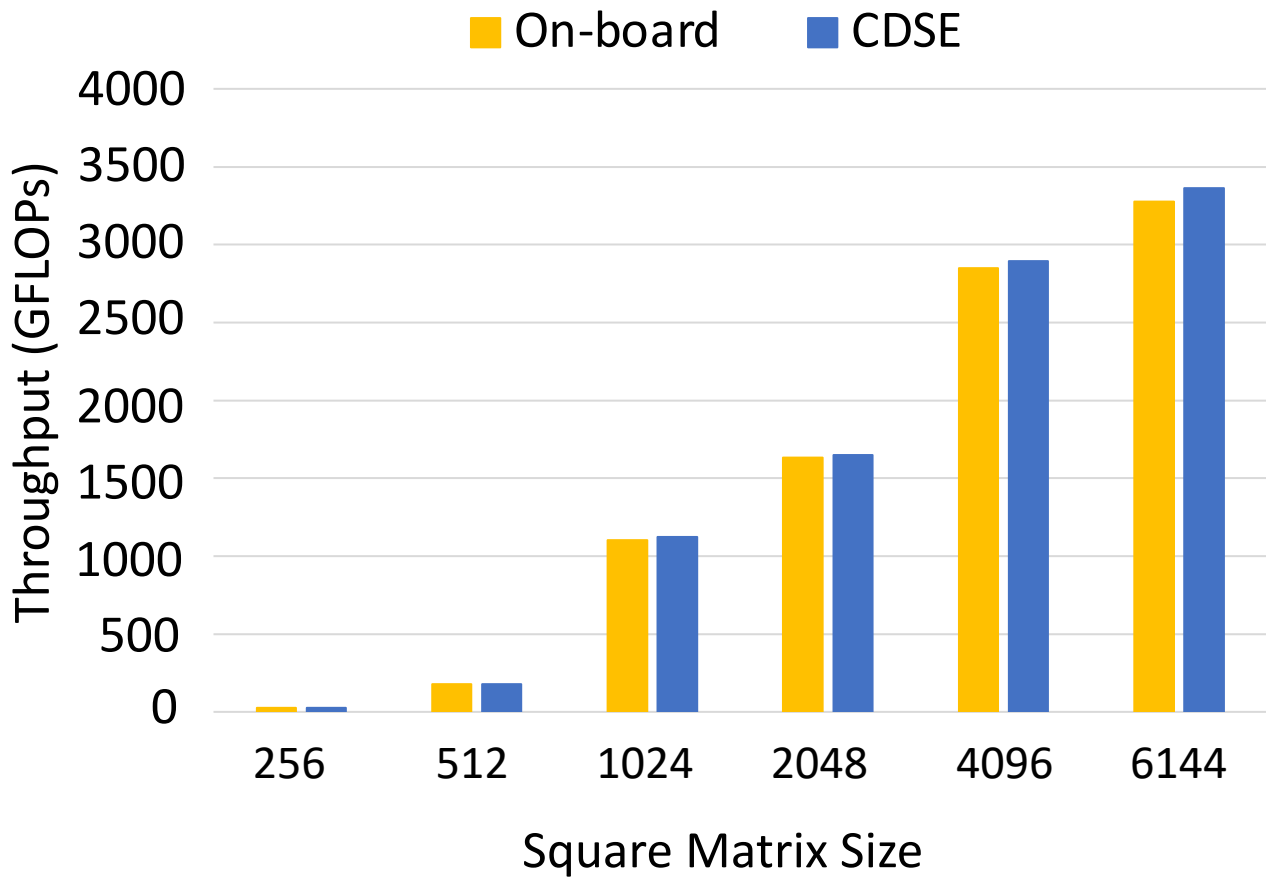
- **Single Accelerator Design**

5.54x, 1.93x gain on Throughput and Energy.Eff



AutoSA[1]: Wang, Jie, Licheng Guo, and Jason Cong. "AutoSA: A polyhedral compiler for high-performance systolic arrays on FPGA." *The 2021 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. 2021.

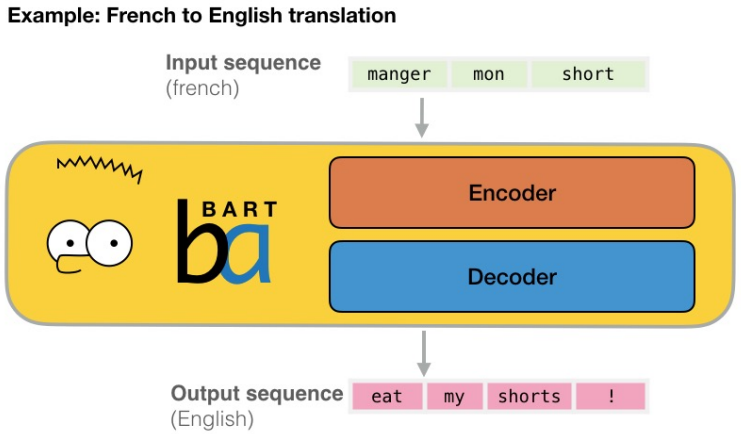
DSE within 5% error rate



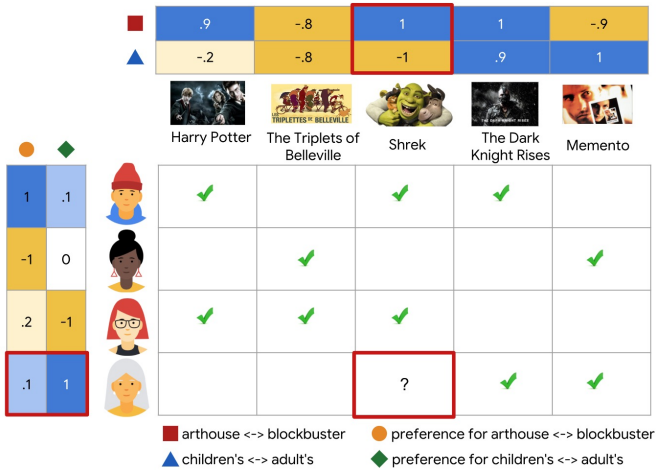
Experiment Results

- Applications

Bert-Large
(Natural Language Processing)

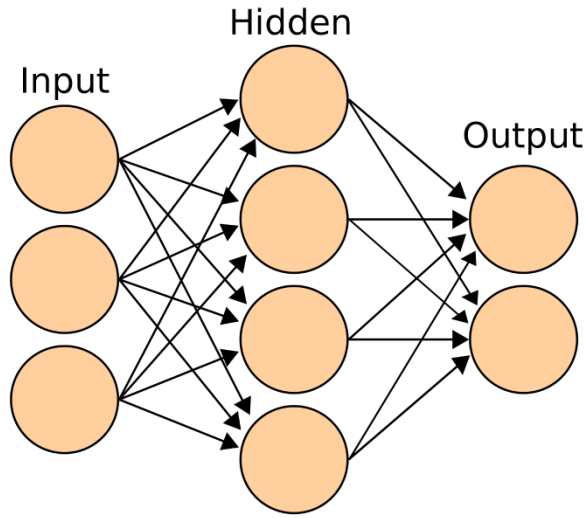


Neural Collaborative Filtering
(Recommendation)



Vision Transformer (Classification)

Multilayer Perceptrons
(Regression, Classification)



Experiment Results

- Applications Characteristics

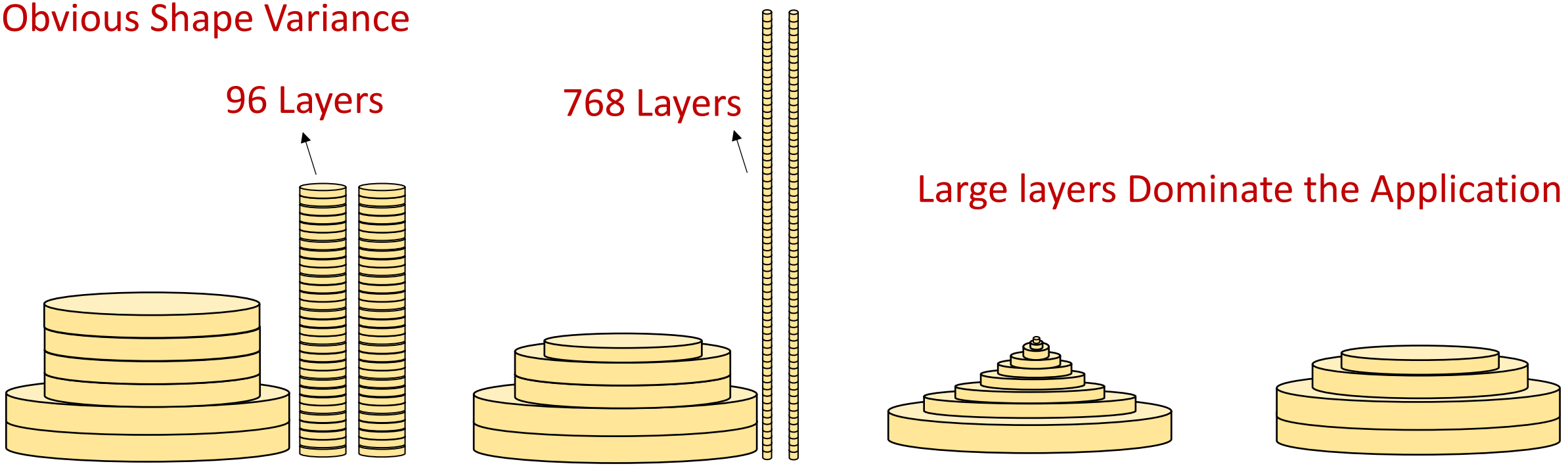
Even Larger Shape Variance

Obvious Shape Variance

96 Layers

768 Layers

Large layers Dominate the Application



BERT

ViT

NCF

MLP

Experiment Results

- **Composing Diverse Accelerators**

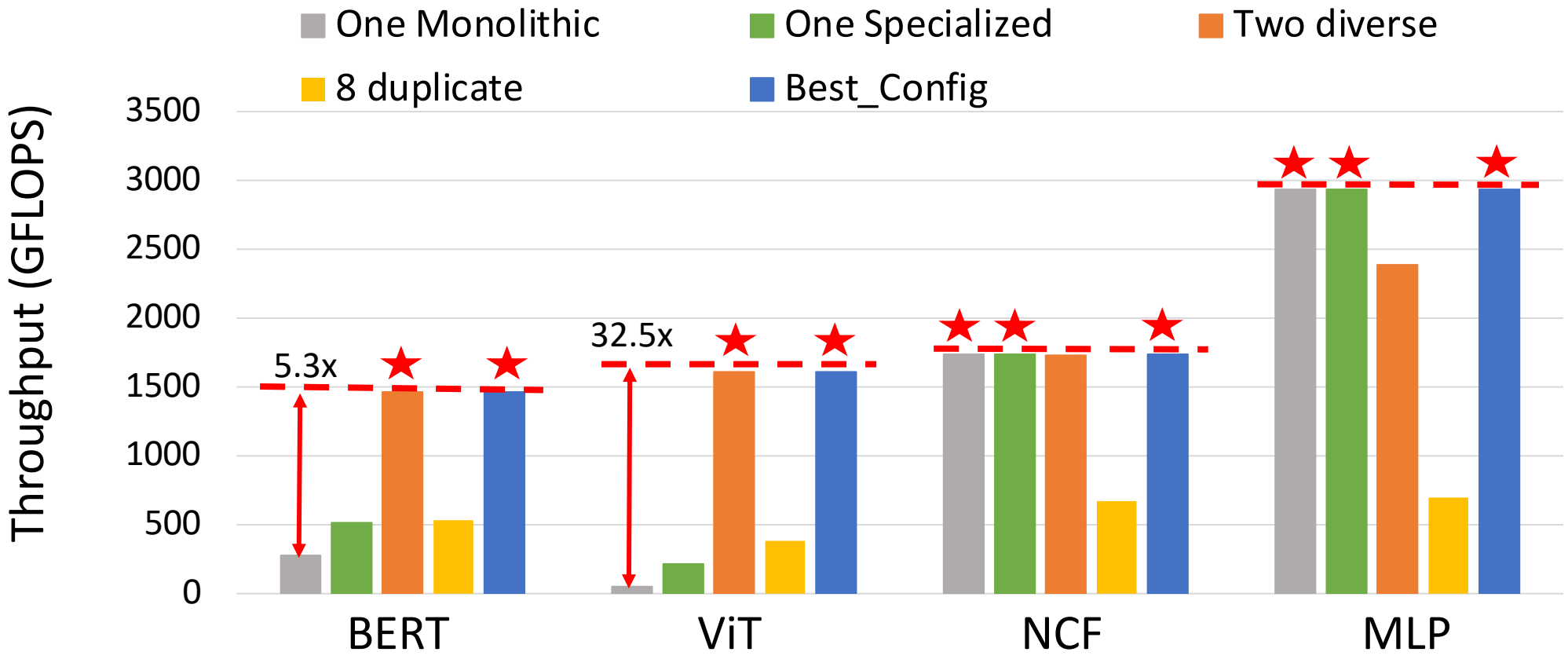
Throughput Comparison Among Different CHARM Strategies

BERT: 1.46 TFLOPS, 5.3x gain over one monolithic design

ViT: 1.61 TFLOPS, 32.5x gain over one monolithic design

NCF: 1.74 TFLOPS for one accelerator design

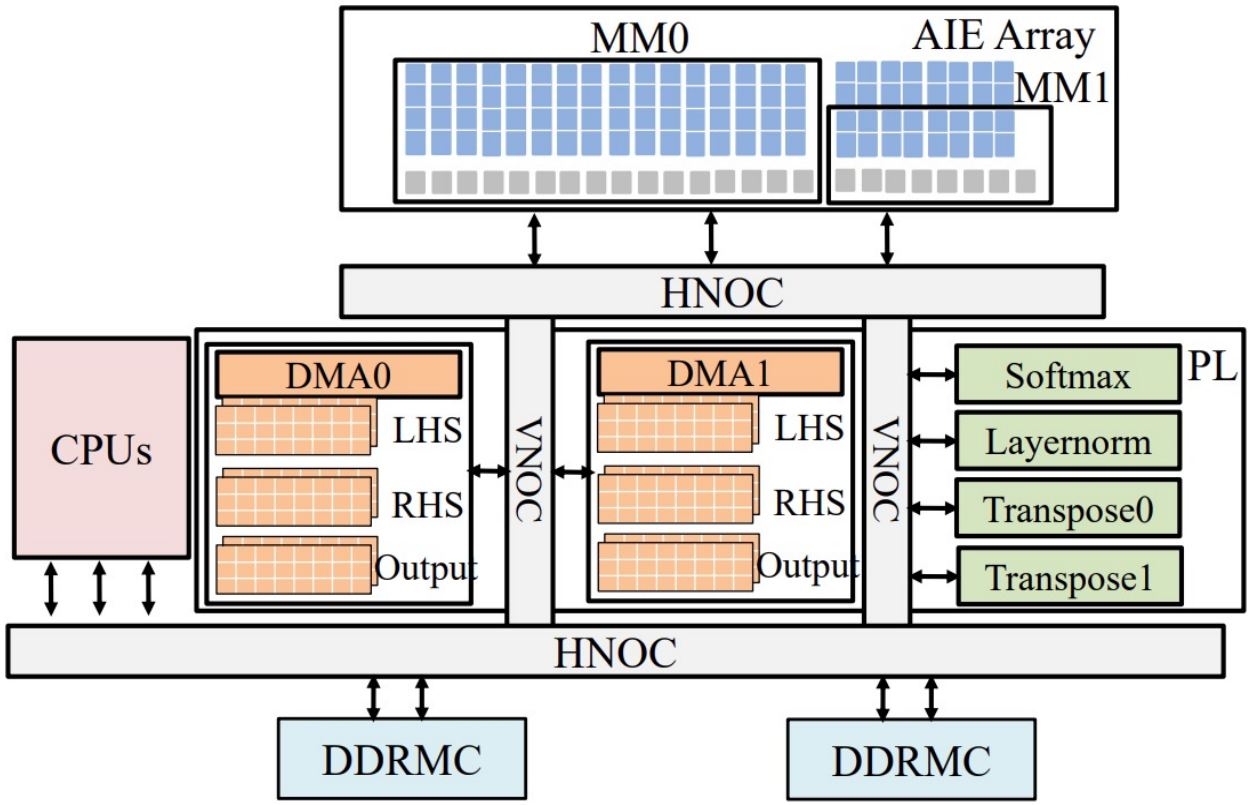
MLP: 2.94 TFLOPS for one accelerator design

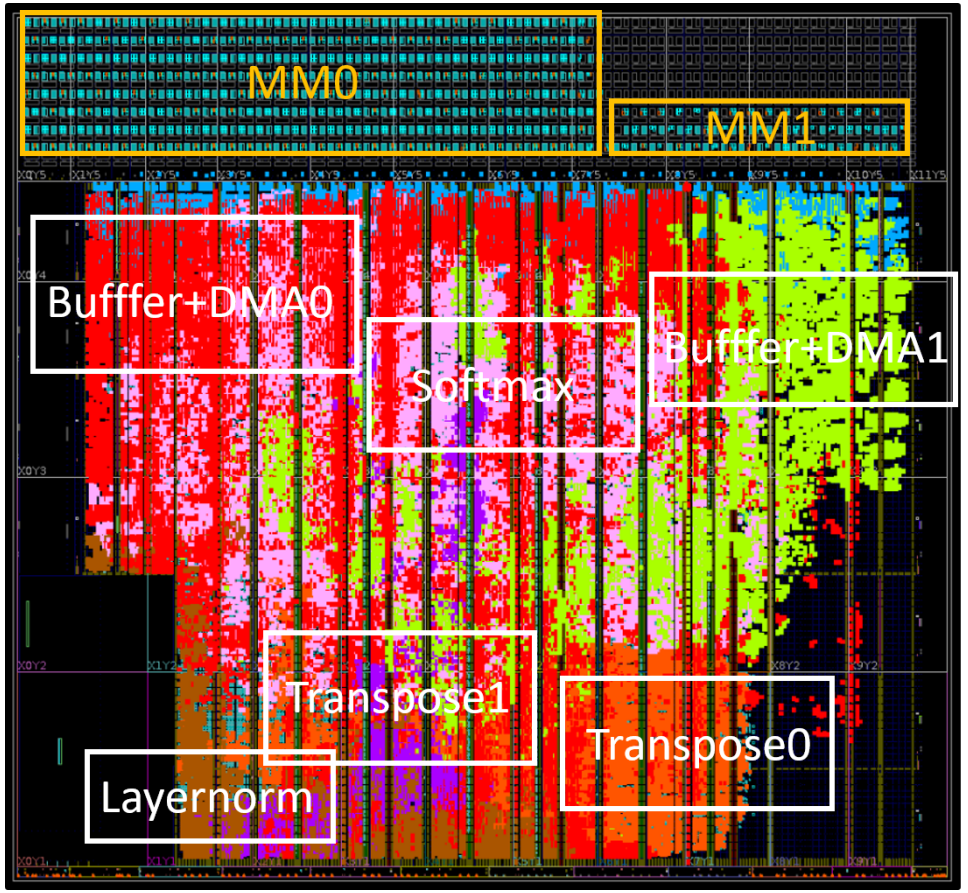


Experiment Results

- **Implementation Layout**

Two Diverse Accelerators Design of BERT





Thank You & Questions

1. Jinming Zhuang, Jason Lau, Hanchen Ye, Zhuoping Yang, Yubo Du, Jack Lo, Kristof Denolf, Stephen Neuendorffer, Alex K. Jones, Jingtong Hu, Deming Chen, Jason Cong, **Peipei Zhou** (2023). CHARM: Composing Heterogeneous AcceleRators for Matrix Multiply on Versal ACAP Architecture (🔥📣New Paper & Project🔥📣!). Proceedings of the 2023 ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '23), February 12–14, 2023, Monterey, CA, USA. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3543622.3573210.

2. Jinming Zhuang, Zhuoping Yang, **Peipei Zhou** (2023). High Performance, Low Power Matrix Multiply Design on ACAP: from Architecture, Design Challenges and DSE Perspectives (🔥📣New Paper & Project🔥📣!). To appear in Proceedings of the 60th ACM/IEEE Design Automation Conference, San Francisco, California, USA, (DAC '23), July 9–13, 2023, San Francisco, CA, USA. Full Paper Accepted (acceptance ratio is 23 percent).

https://peipeizhou-eecs.github.io/

https://github.com/arc-research-lab/CHARM

https://dl.acm.org/doi/10.1145/3543622.3573210



Backup Slides: Experiment Results

- **Search Time Comparison for BERT**

  Two step search algorithm: 170s VS exhaustive search: 33mins