Shortcut Mining: Exploiting Cross-layer Shortcut Reuse in DCNN Accelerators

Arash Azizi and Lizhong Chen

STAR Lab*

Oregon State University

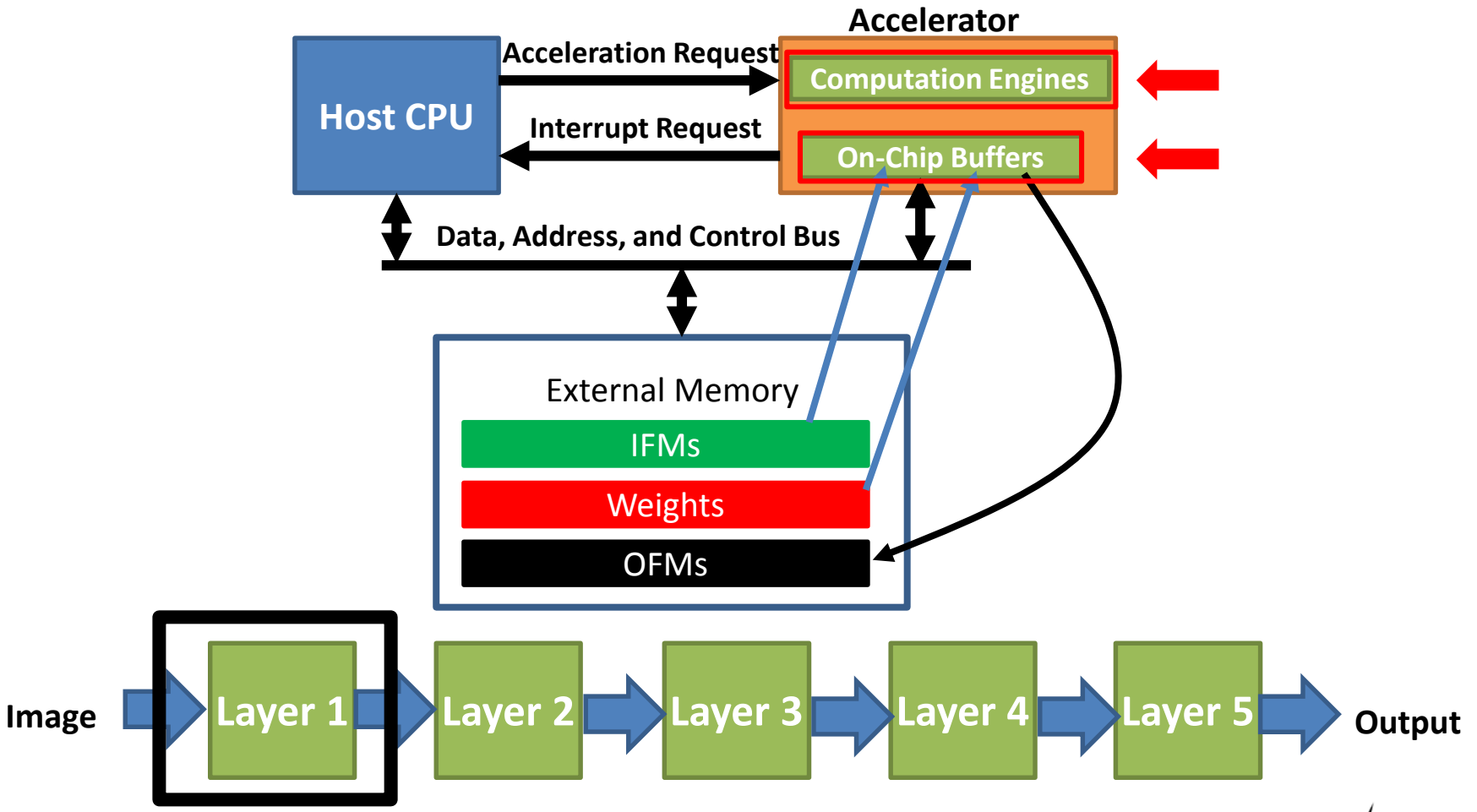
February 18, 2019

*System Technology and Architecture Research (STAR) Lab

Introduction:
DNN accelerators architecture and how they work

Introduction and Motivation

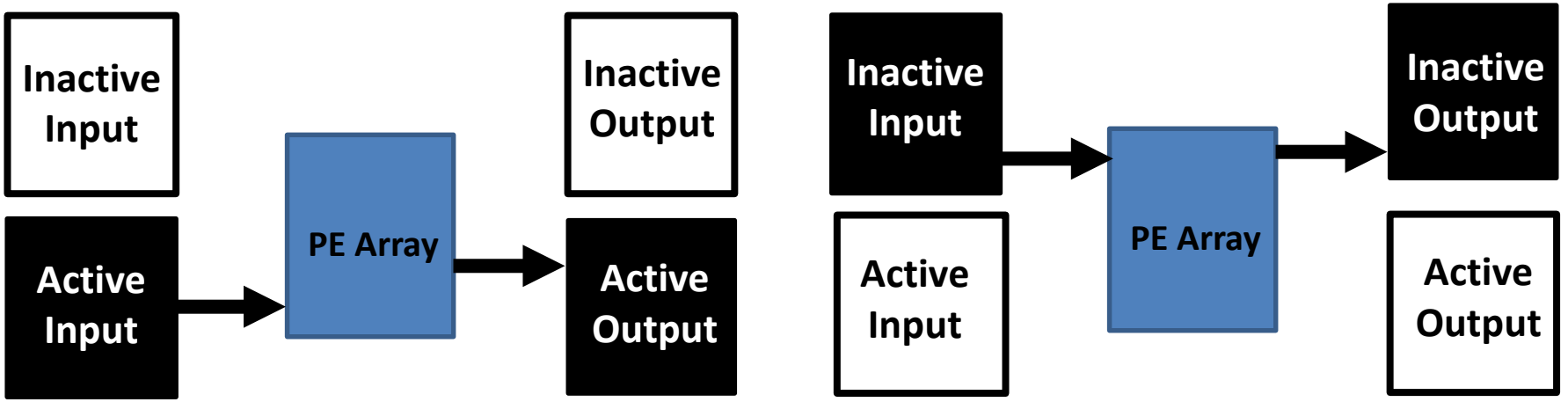
- Layer-wise DNN Accelerators



S.T.A.R Oregon State

Ping-Pong Buffers and Banked Memory

- Decoupled access/execute architecture (**Ping-Pong Buffers**) is used to overlap communication latency with computation time

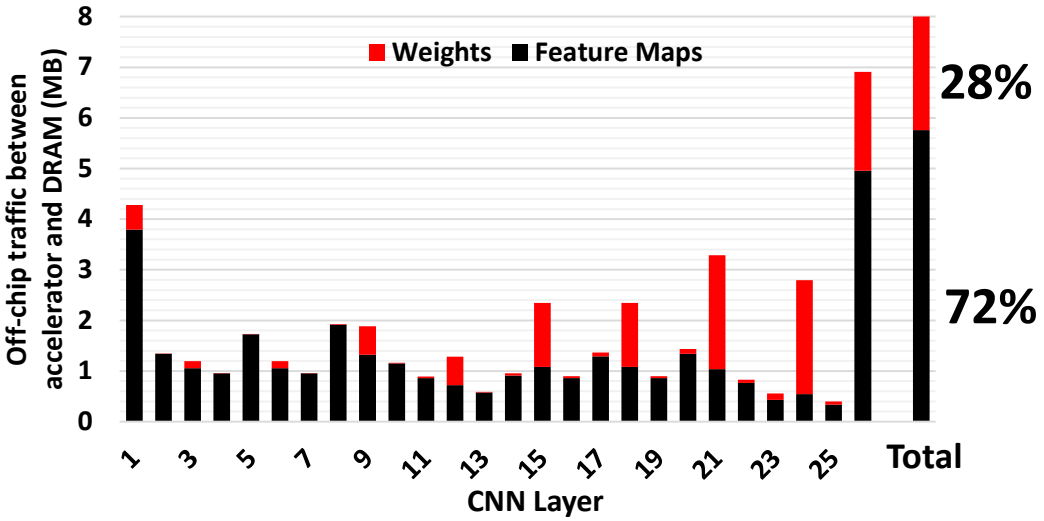


- On-Chip buffers are organized as banked memory
  - To avoid long word-lines and bit-lines that have high capacitance
  - Banked buffers provide more read and write ports, allowing multiple simultaneous accesses to feature maps and weights

Motivations for this work

Motivations

- Off-Chip accesses are very expensive in terms of energy consumption.
- The percentage of data from feature maps increases rapidly.



Off-chip Traffic of a SqueezeNet with 26 layers

- Existing techniques in reducing data from weights and from feature maps are not equally effective.
  - Effective techniques such as pruning
  - Most of existing works don't explore cross-layer data reuse among all the layers

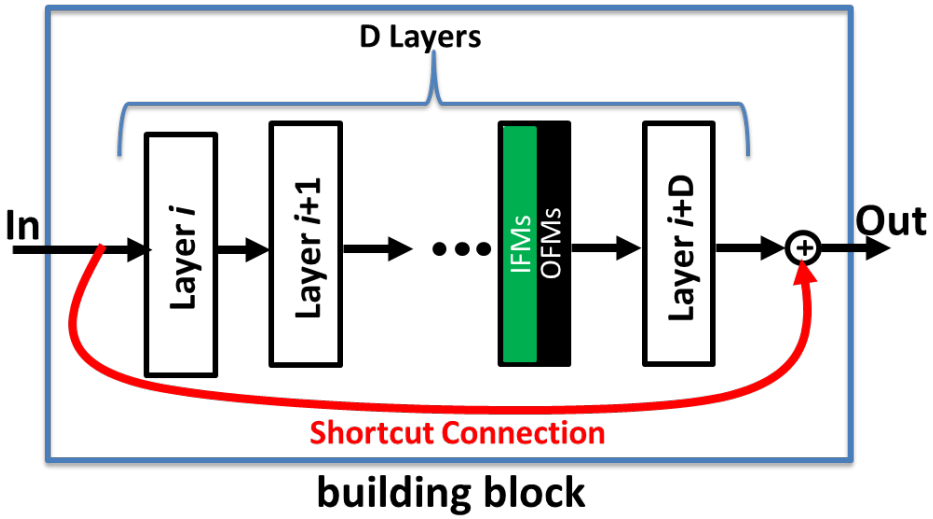
Motivations

- In layer-wise model, feature maps of a layer which remains on-chip can be reused for the next layer processing.
  - With larger on-chip buffers significant feature maps remains on-chip

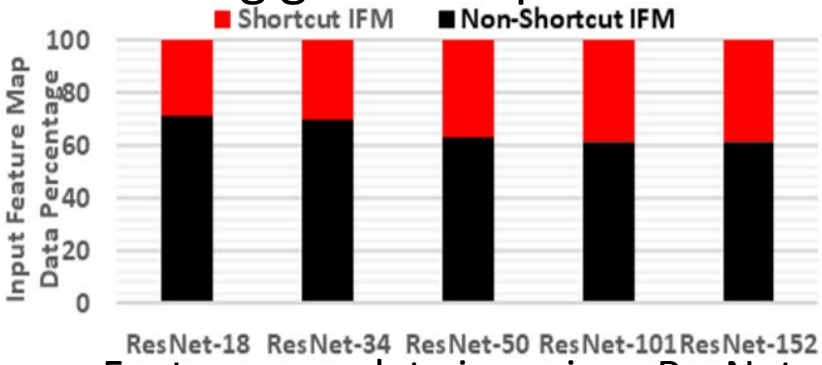
On-chip buffers in several modern platforms

Chips	On-Chip Memory	Integration Technology Node
Xilinx Virtex UltraScale+ VU37P	47.2 MB	16nm
Google's TPU	28MB	28nm
Intel Stratix 10	30.5MB	14nm

- Shortcut connections are critical in developing highly accurate deep network.
  - Address the gradient "wash out" or vanishing gradients problem



building block

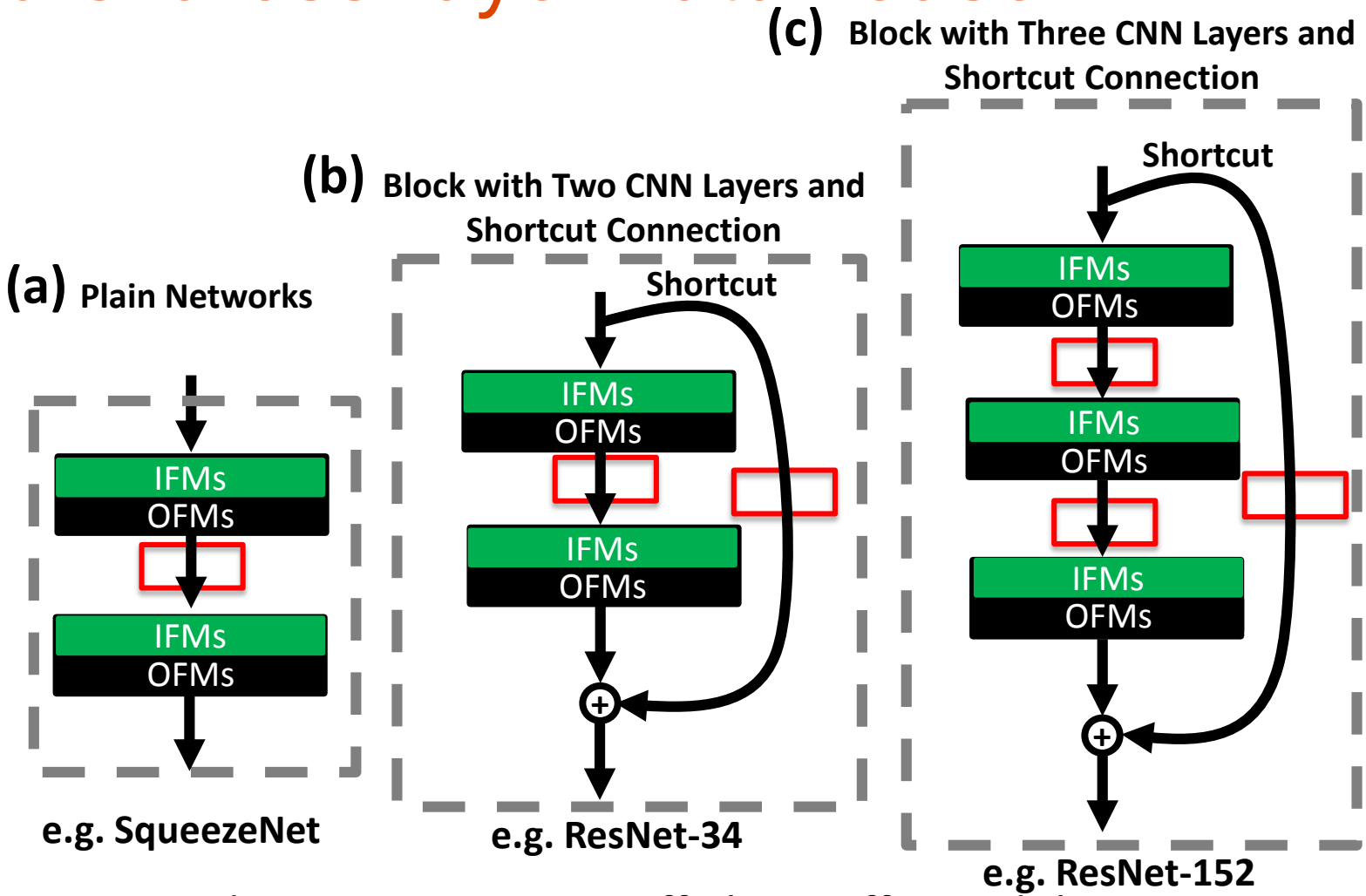


Feature map data in various ResNets

S.T.A.R Oregon State

Goals, Challenges, Approach, and Outcomes

Goals: Cross-layer Data Reuse



- Cross-Layer data reuse can save off-chip traffic, and there are two types of cross-layer data reuse:
  - Feature Map Reuse
  - Shortcut Reuse

Proposed Approach (Shortcut Mining)

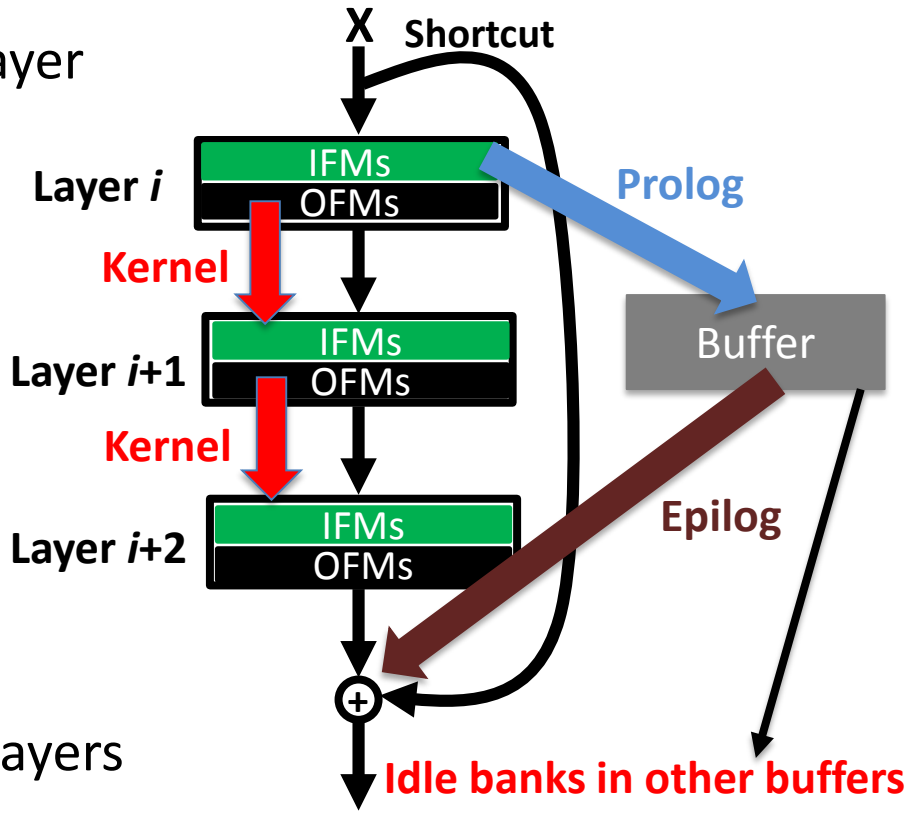
- Shortcut Mining: A sequence of procedures that allow shortcut and feature maps data to be reused across any number of layers in the building

- **Kernel** reuses OFMs of the previous layer as the IFMs of next layer

- **Prolog** saves the remaining IFMs marks it as reusable shortcut

- **Epilog** restores the saved shortcut data

- **Challenges:**



- ❑ **Finding Buffer Space for Shortcuts:**
  - Shortcut should be kept multiple layers
  - Need an extra buffer, negative impact on the other buffers
- ❑ **Devising Unified Solution for Various Networks** with any number of layers

S.T.A.R Oregon State

Research outcomes

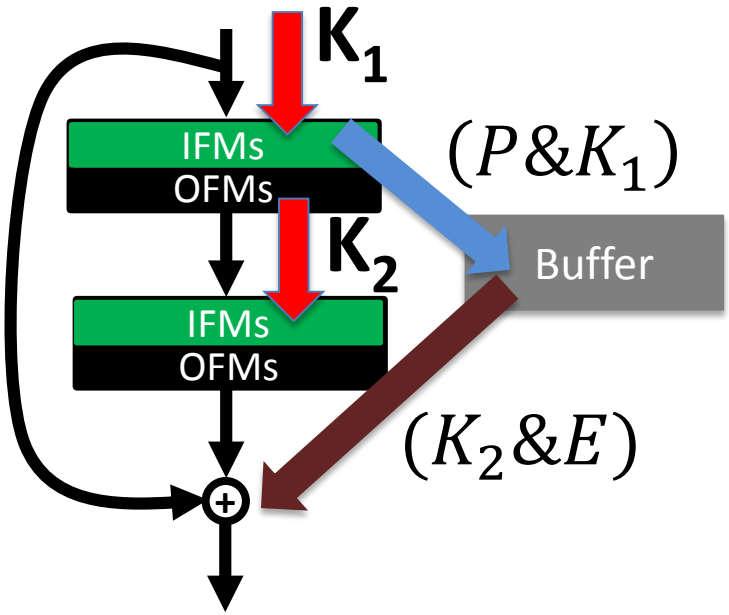
- Outcomes
  - Extensive data reuse among layers
  - Applicable to different building blocks with any number of layers
  - Introducing the abstraction of logical buffers for a new flexible on-chip memory architecture
  - High utilization of on-chip memory

- Results
  - 43% to 72% off-chip traffic reduction
  - Near 25% lower on-chip power for residual networks
  - 1.93X throughput improvement over state-of-the-art design

How procedures reuse data across any number of layer?

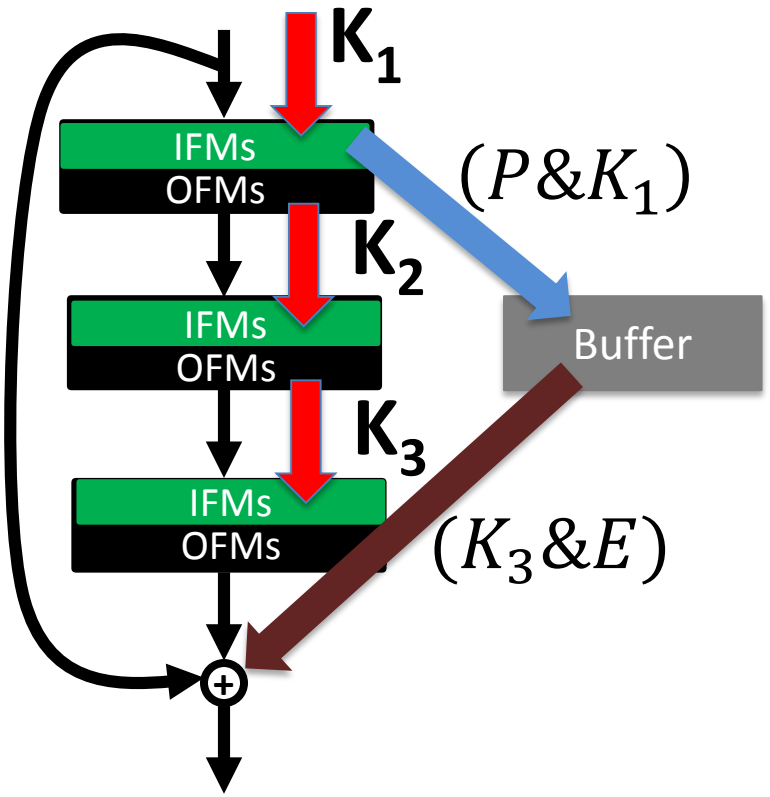
Cross-layer Data Reuse Examples

Residual Building Block with 2 layers



$(P\&K_1) + (K_2\&E)$

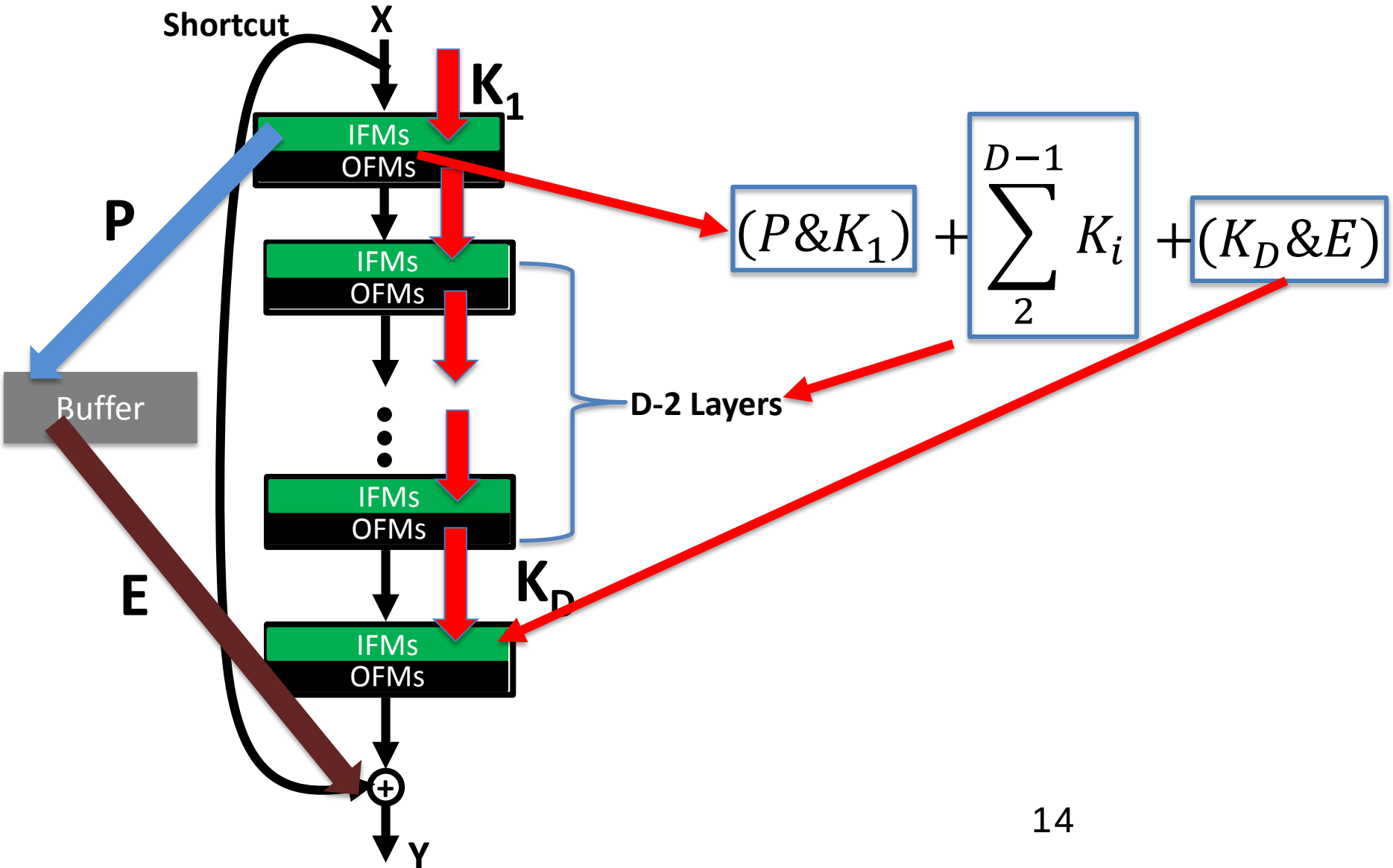
Residual Building Block with 3 layers



$(P\&K_1) + K_2 + (K_3\&E)$

Shortcut Mining

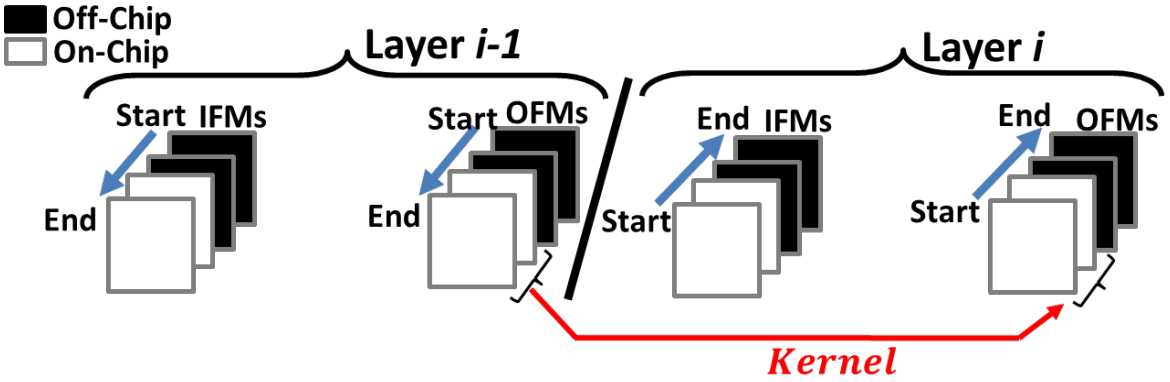
- A sequence of operations that allow shortcut data to be reused across any number of layers in the building block



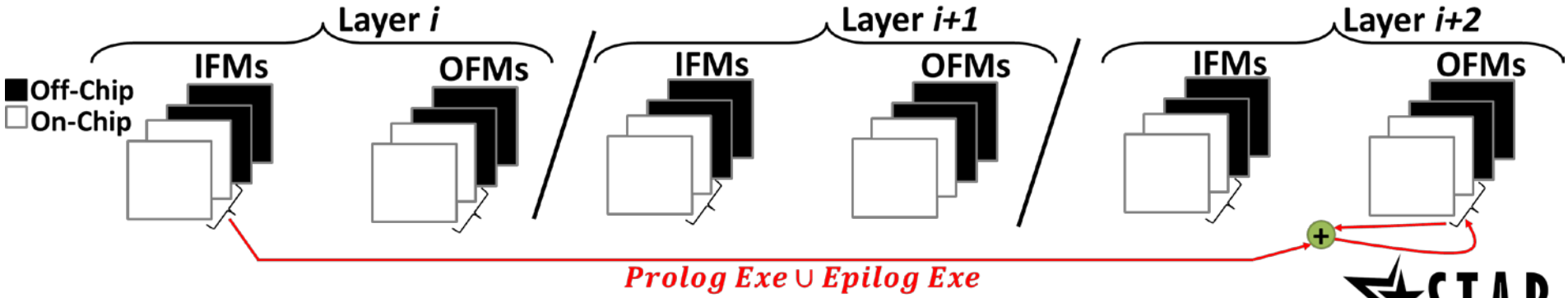
How each procedure works?

Procedures Overview

- Kernel procedure:
  - Reusing the available on-chip OFMs
  - Keeping a certain region of the buffer space untouched to keep shortcut for multiple layers

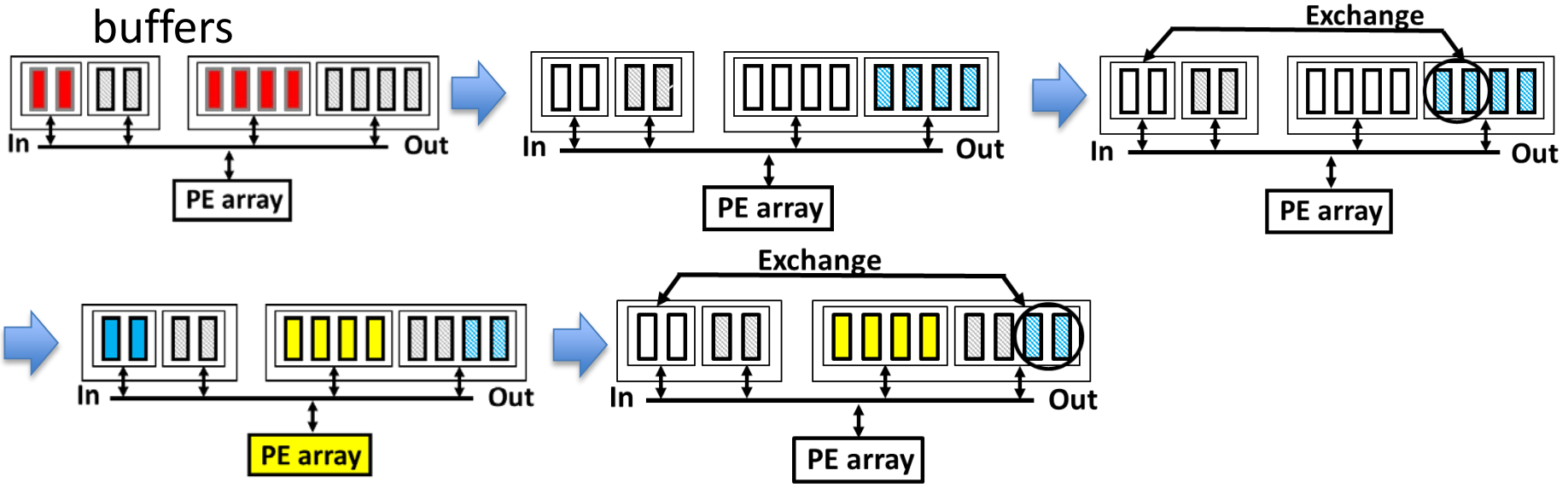


- Prolog procedure: Storing the on-chip IFMs as shortcuts for future reuse
- Epliog procedure : Restoring the shortcuts and using them in computation

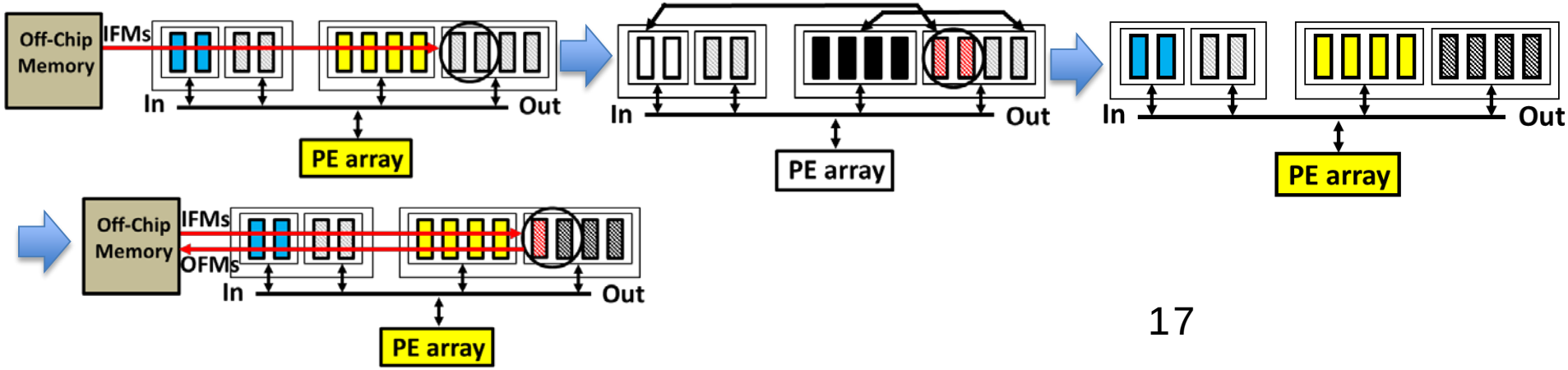


Kernel Procedure

- Kernel procedure consists of two phases
  - Reuse Phase: reusing OFMs as IFMs through back exchanging between buffers

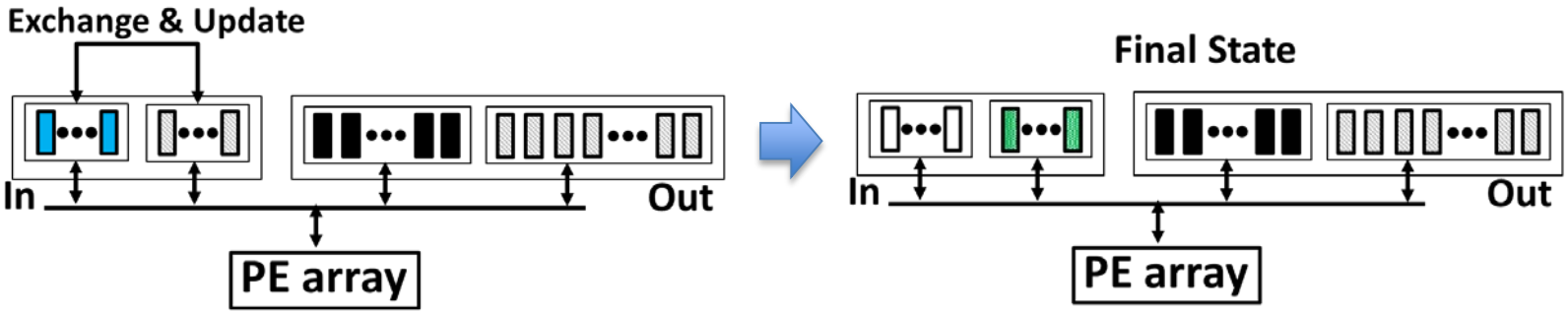


  - Preload and Write Back Phase: writing back the computed OFMs and loading IFMs if it is needed

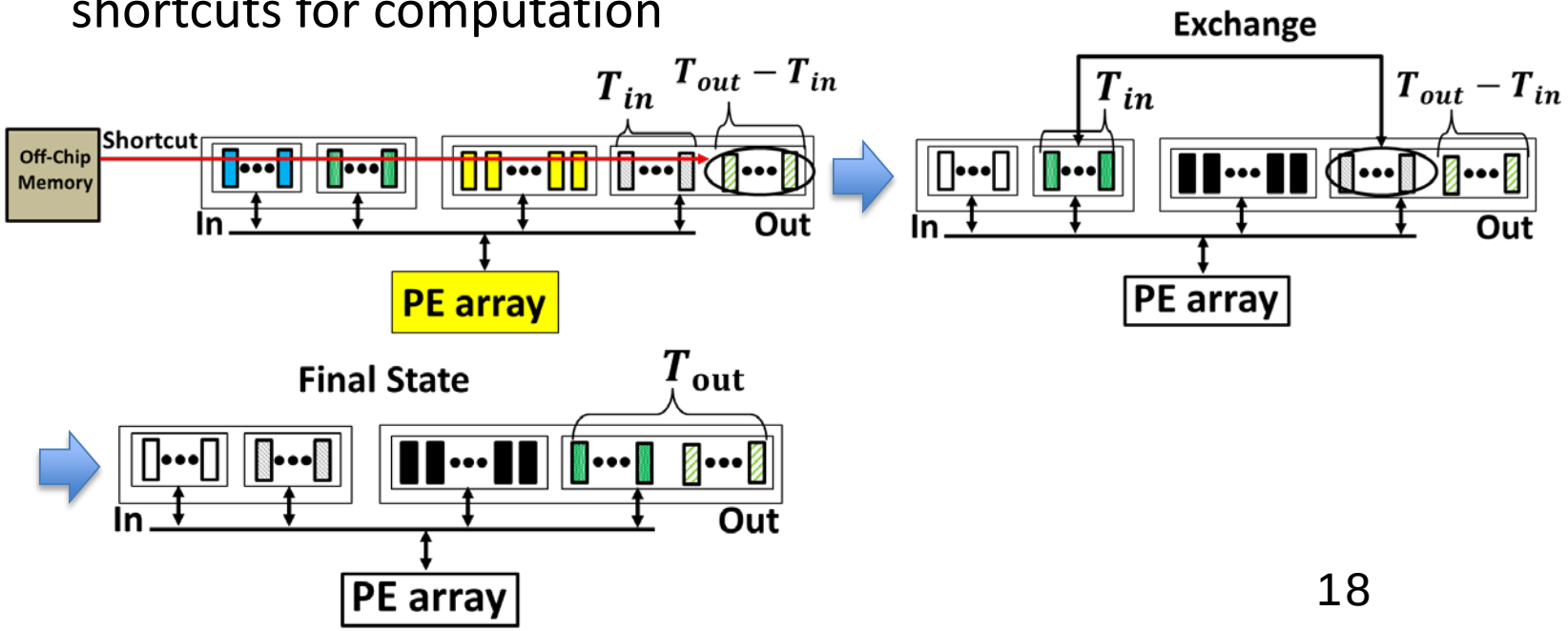


Prolog and Epilog Procedures

- Prolog procedure: Storing the IFMs as shortcuts

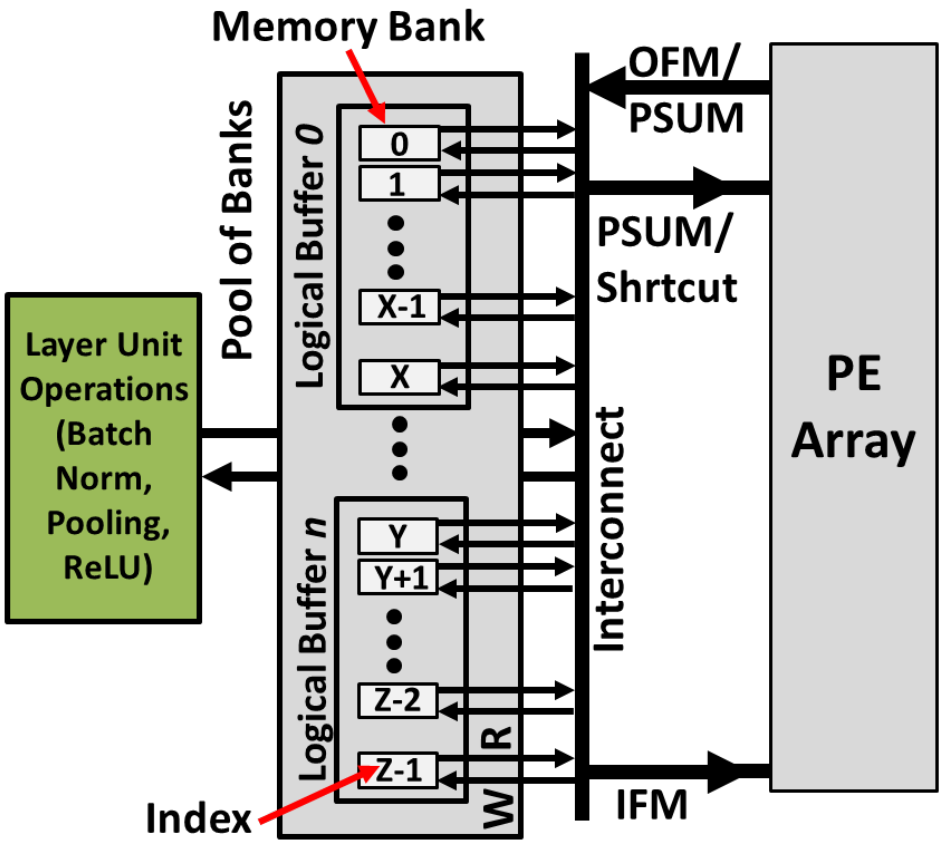


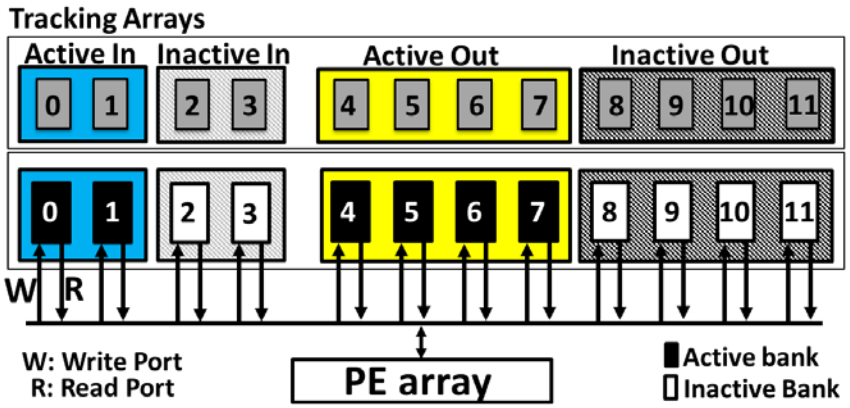
- Epilog procedure: 1-Preloading shortcuts if it is needed 2-Restoring the shortcuts for computation



Decoupled Physical-Logical Buffers

- During procedures running, each bank may appear in different buffers
- Logical buffers should be created and constructed dynamically during run time
- With static buffer architecture, procedures cannot be run due to static bank assignment

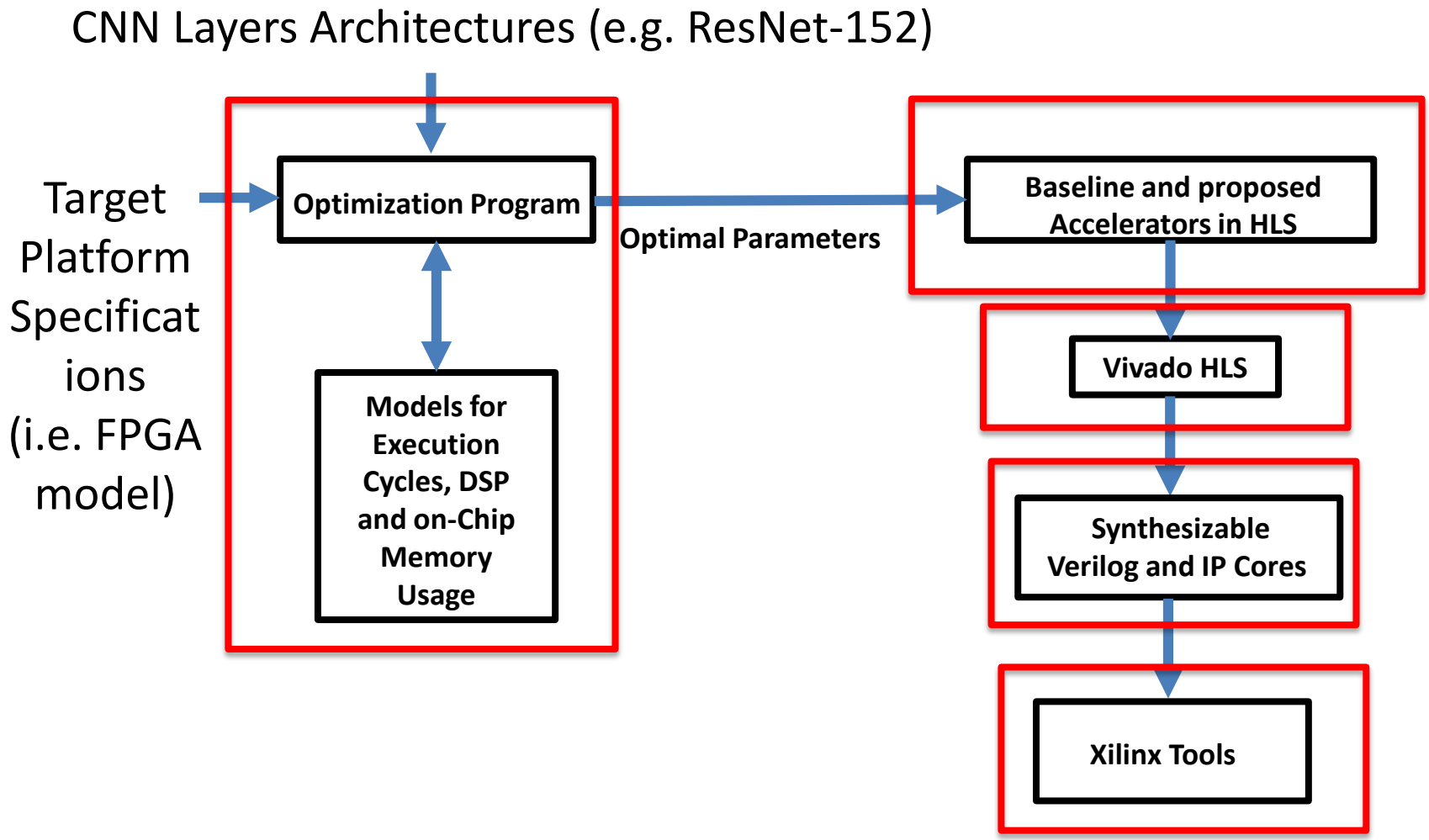




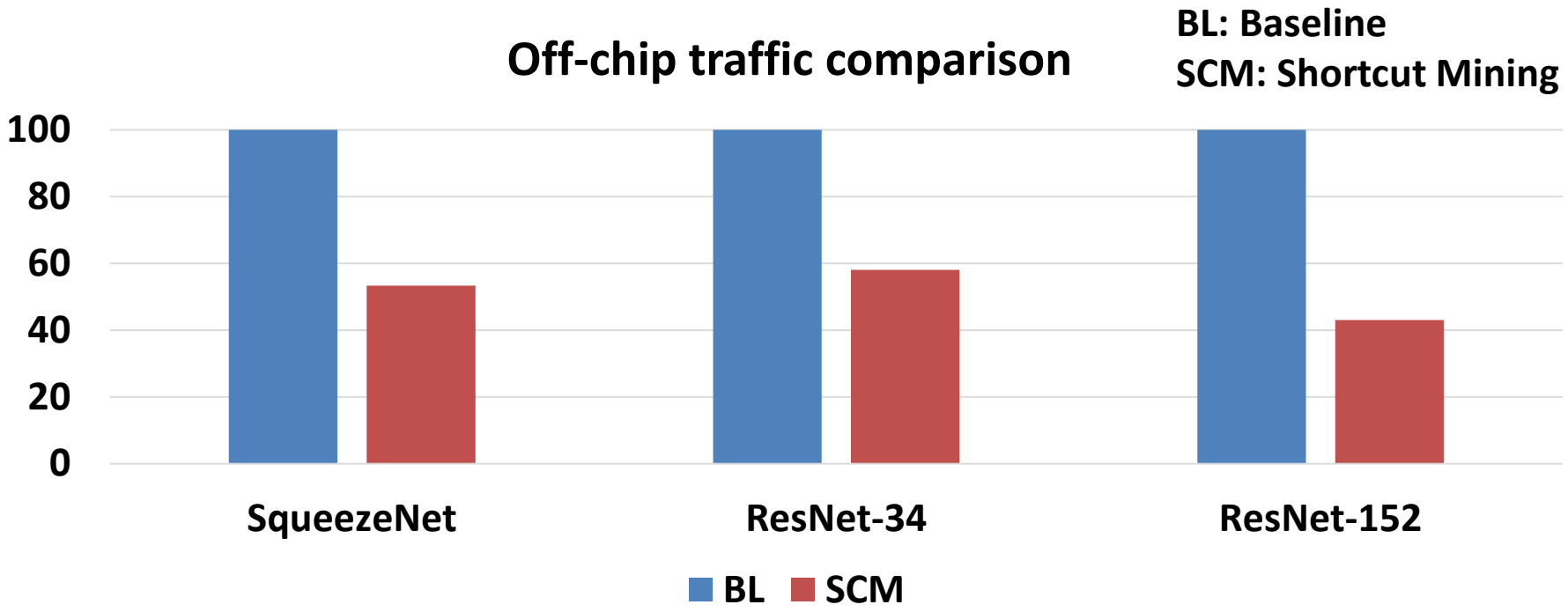
S.T.A.R Oregon State

Evaluation and Results

Methodology



Off-Chip Traffic

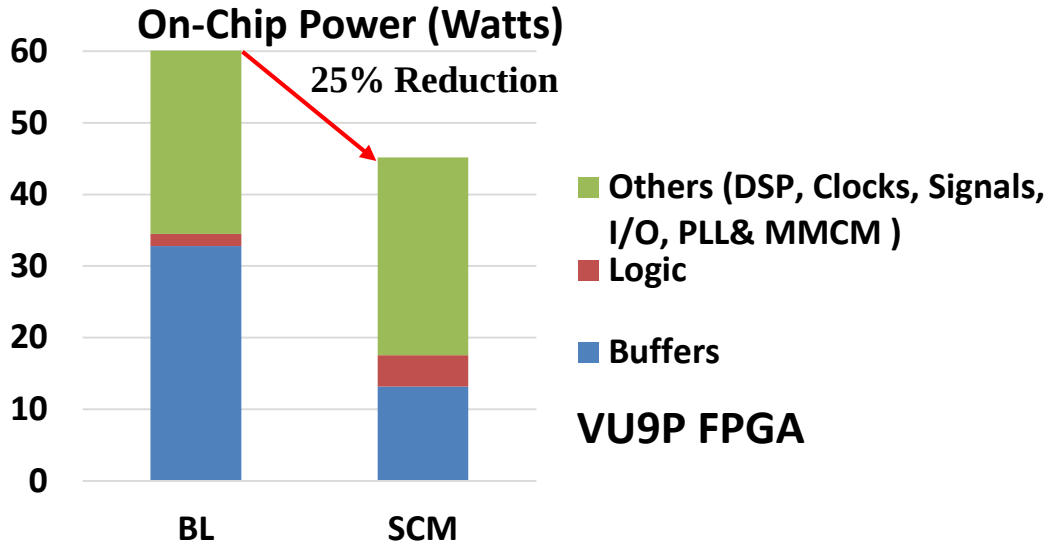


- SqueezeNet, ResNet-34, ResNet-152
- VU9P: 53.3%, 58%, and 43%

- While the proposed approach reduce the off-chip memory traffic, it shows better memory utilization
  - E.g. 14% smaller on-chip memory for SqueezeNet

On-Chip Power and Performance

- On-chip power



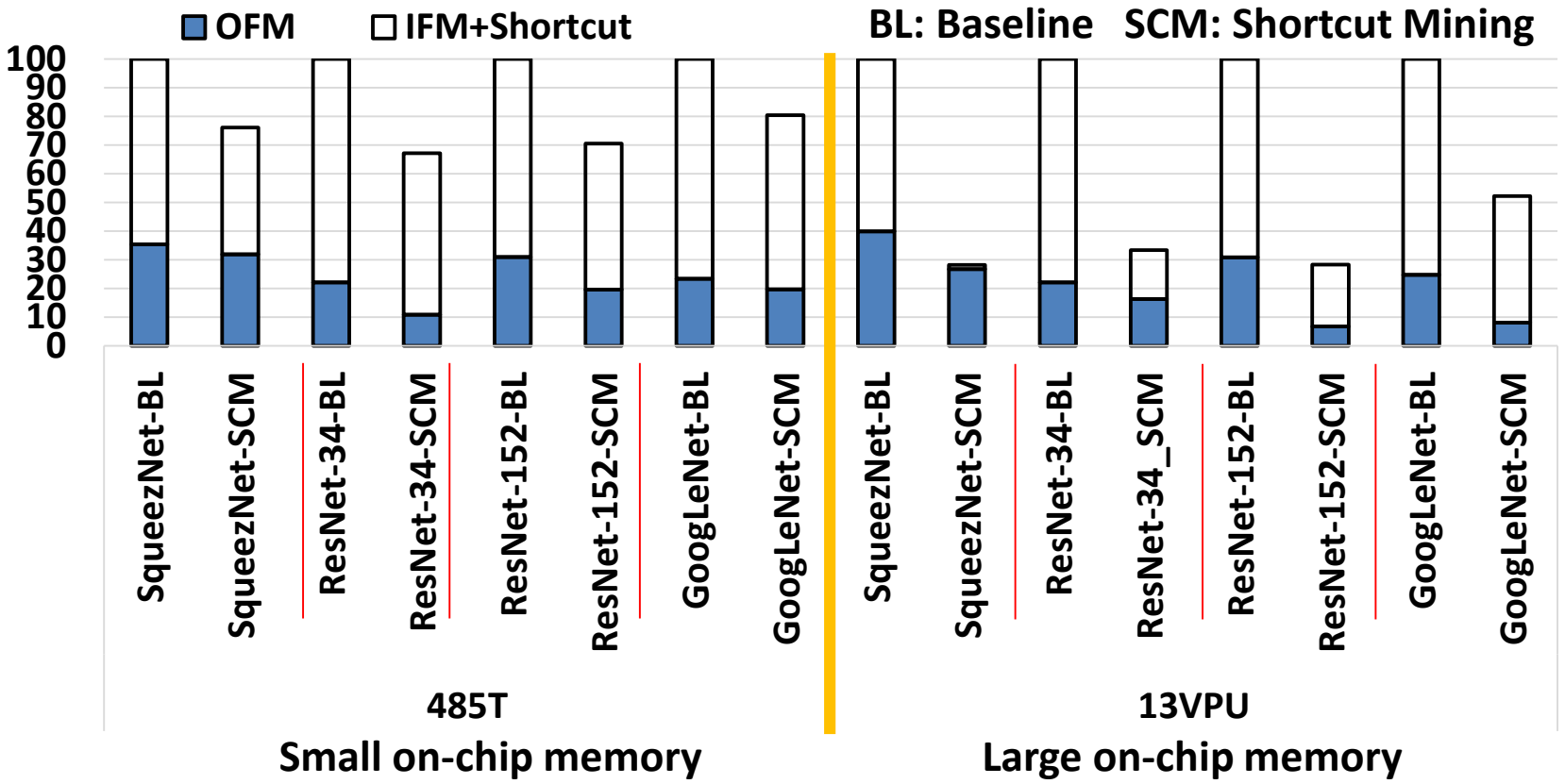
Power breakdown of ResNet-152.

- Performance

	State-of-the-art	SCM
FPGA	Arria-10 GX 1150	Virtex-7 485T
Frequency (MHz)	150	150
Network	ResNet-152	ResNet-152
Data Format	16-bit	16-bit
Latency (ms)	71.71	35.24
Throughput (GOPS)	315.48	608.28

1.9X improvement

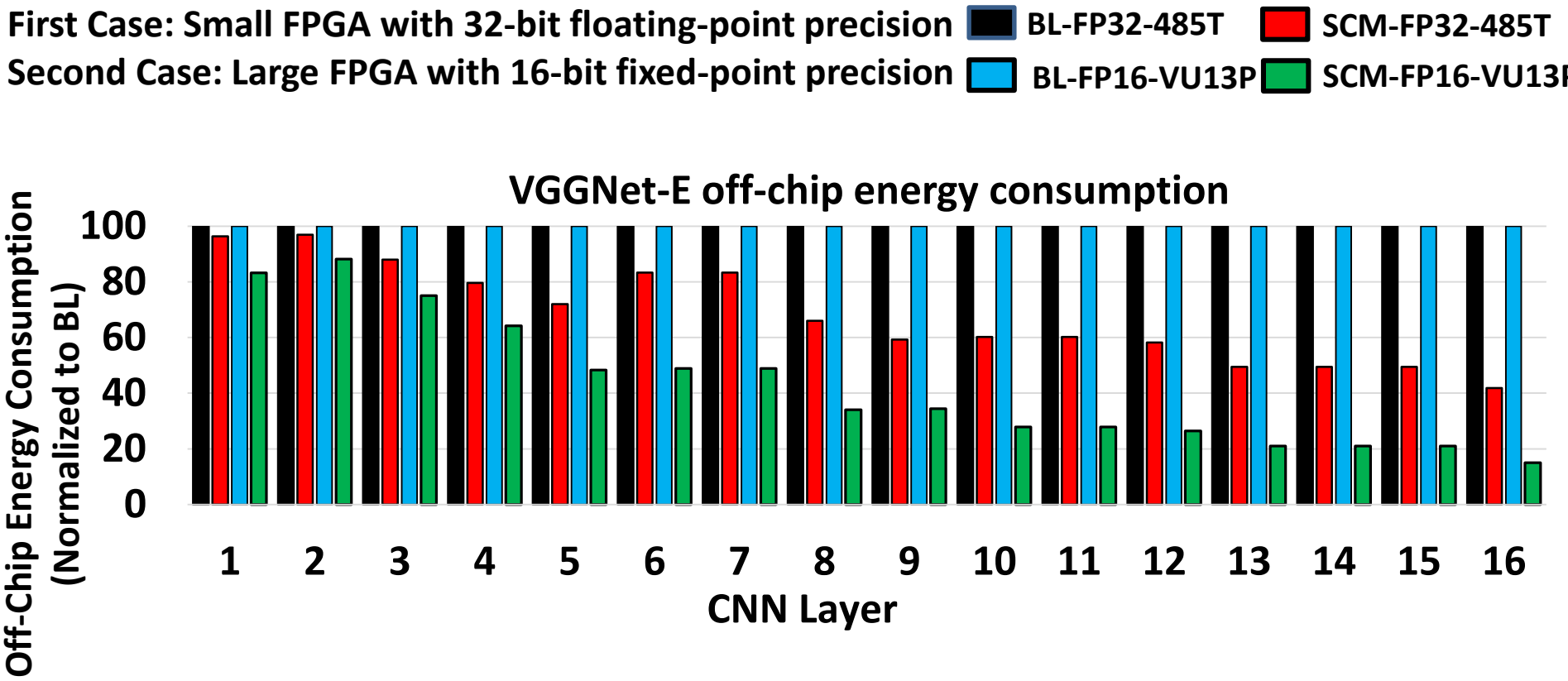
Scalability of Off-Chip Traffic Reduction



- SqueezeNet, ResNet-34, ResNet-152 and GoogLeNet
- 485T: 24%, 33%, 30% and 20%
- 13VPU: 72%, 67%, 72%, and 48%

S.T.A.R Oregon State

Layer-by-layer Analysis



- On average, the off-chip energy consumption is reduced by 25.2% and 50% for the first and second case, respectively.

Conclusion

- For fast and power efficient acceleration of deep networks, off-chip traffic should be reduced

- We propose Shortcut Mining design
  - Avoid re-fetching data for feature maps and shortcuts
  - Applicable to building blocks with any number of layers
  - Achieve scalable off-chip traffic reduction
  - High utilization of on-chip memory

- Significant improvements for off-chip traffic, power and throughput

Thank you!