AOC Proposal
Systolic Tensor Array

Group｜把家齊高中還給家齊

Member

電機所114	張育誠	N26121892
工科系113	劉子齊	E94096209
電機系113	洪翊碩	E24091027
電機系113	陳喬雅	E14093140
電機系113	鄭喆嚴	E14096724

TABLE OF CONTENTS

01 Model, Quantization and Pruning

02 PE Architecture and Dataflow

03 Our Accelerator Architecture and SPEC

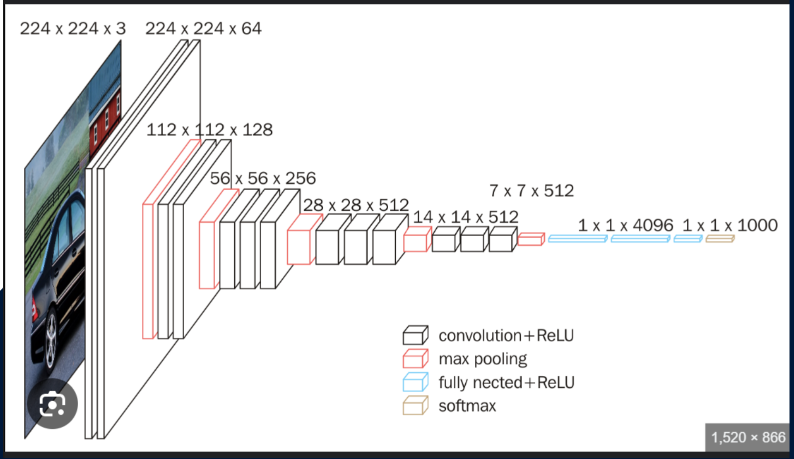
04 Analysis and Question

01

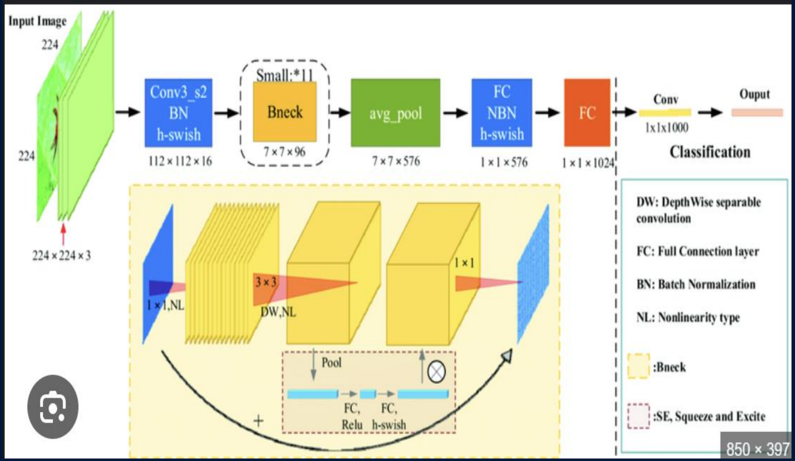
Model, Quantization and Pruning

Model

VGG16:

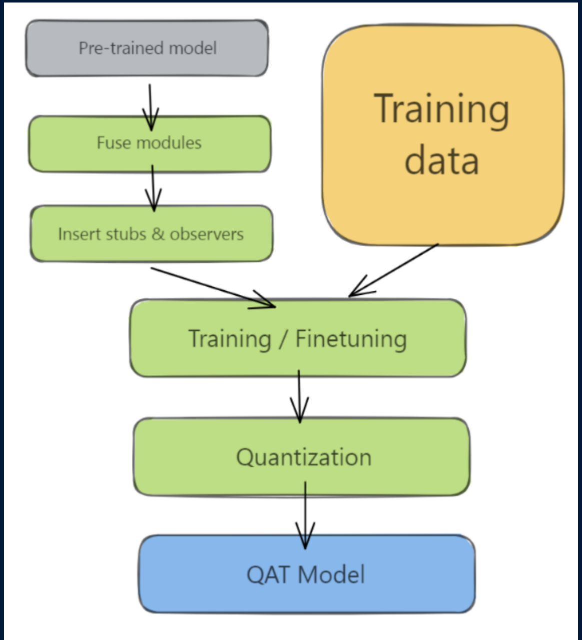


Mobile NetV3:

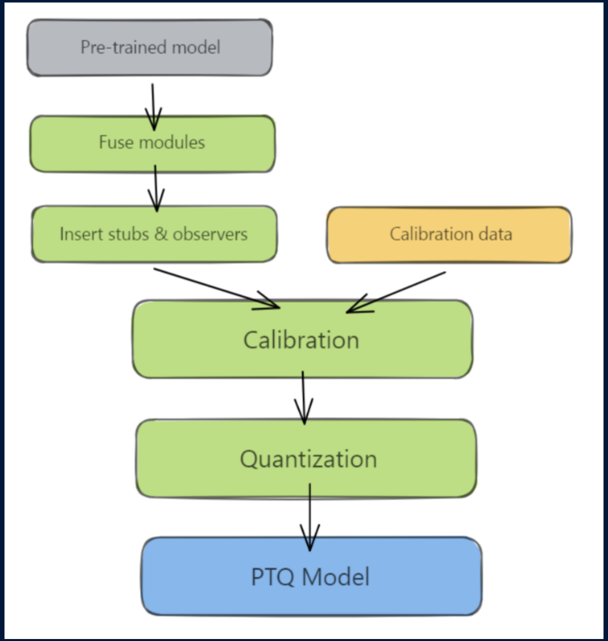


Quantization

QAT:

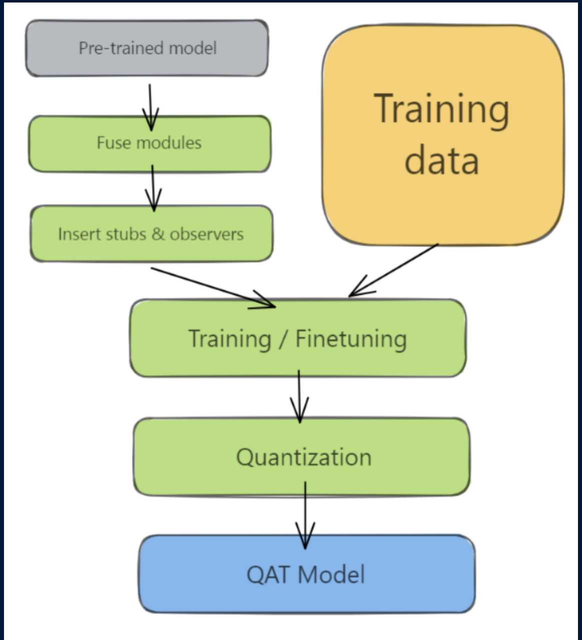


PTQ:



Quantization

QAT:



Pre-trained model

Fuse modules

Insert stubs & observers

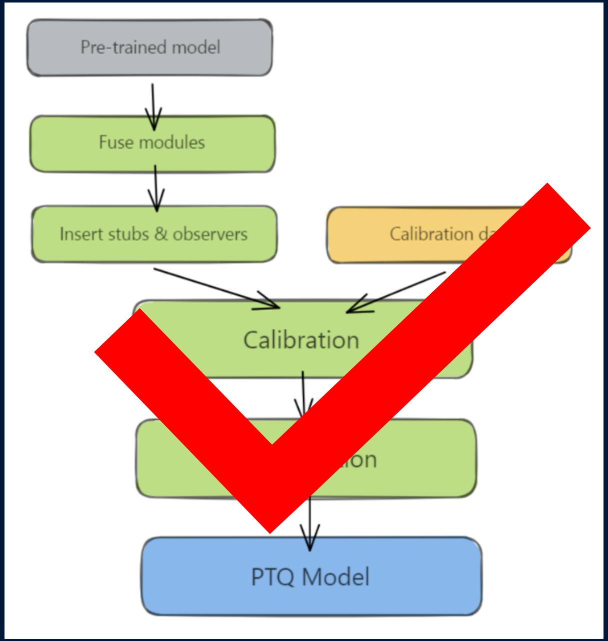
Training data

Training / Finetuning

Quantization

QAT Model

PTQ:



Pre-trained model

Fuse modules

Insert stubs & observers

Calibration d

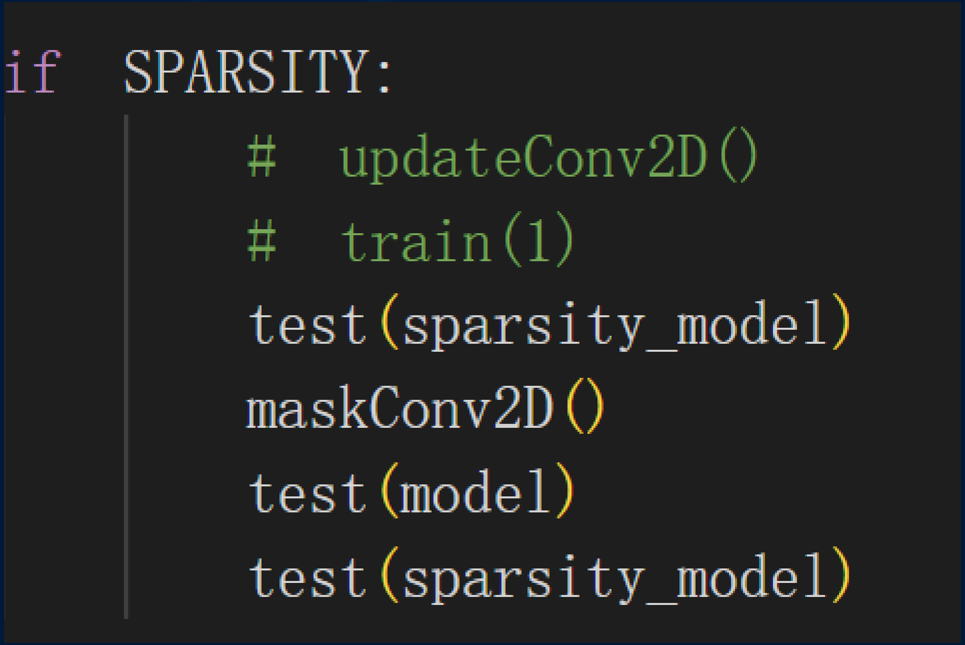
Calibration

ion

PTQ Model

Pruning

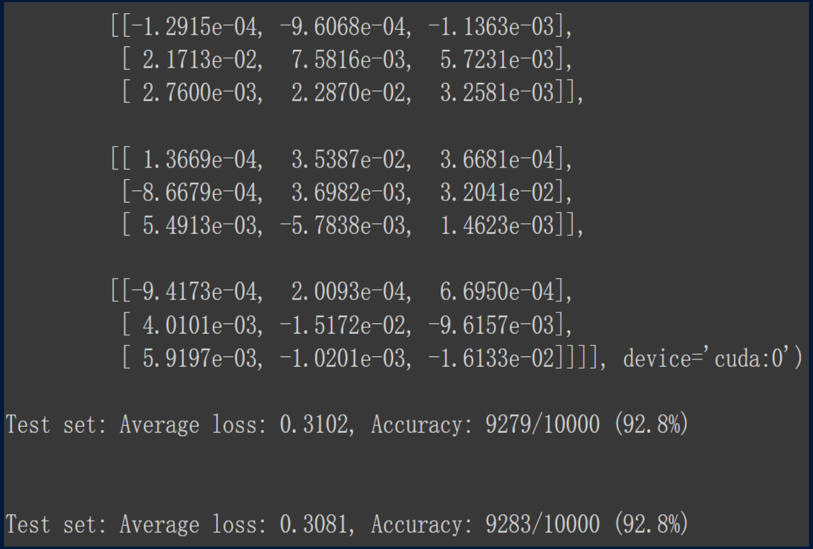
Fix position pruning:



```
if SPARSITY:
    # updateConv2D()
    # train(1)
    test(sparsity_model)
    maskConv2D()
    test(model)
    test(sparsity_model)
```

Pruning

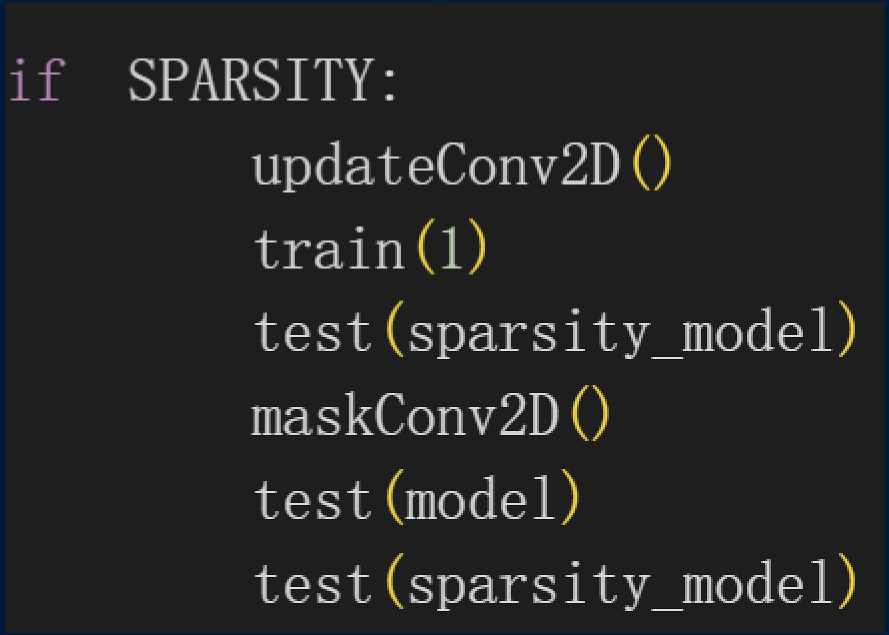
Original



```
         [[-1.2915e-04, -9.6068e-04, -1.1363e-03],
          [ 2.1713e-02,  7.5816e-03,  5.7231e-03],
          [ 2.7600e-03,  2.2870e-02,  3.2581e-03]],

         [[ 1.3669e-04,  3.5387e-02,  3.6681e-04],
          [-8.6679e-04,  3.6982e-03,  3.2041e-02],
          [ 5.4913e-03, -5.7838e-03,  1.4623e-03]],

         [[-9.4173e-04,  2.0093e-04,  6.6950e-04],
          [ 4.0101e-03, -1.5172e-02, -9.6157e-03],
          [ 5.9197e-03, -1.0201e-03, -1.6133e-02]]]], device='cuda:0')

Test set: Average loss: 0.3102, Accuracy: 9279/10000 (92.8%)


Test set: Average loss: 0.3081, Accuracy: 9283/10000 (92.8%)
```

Pruning

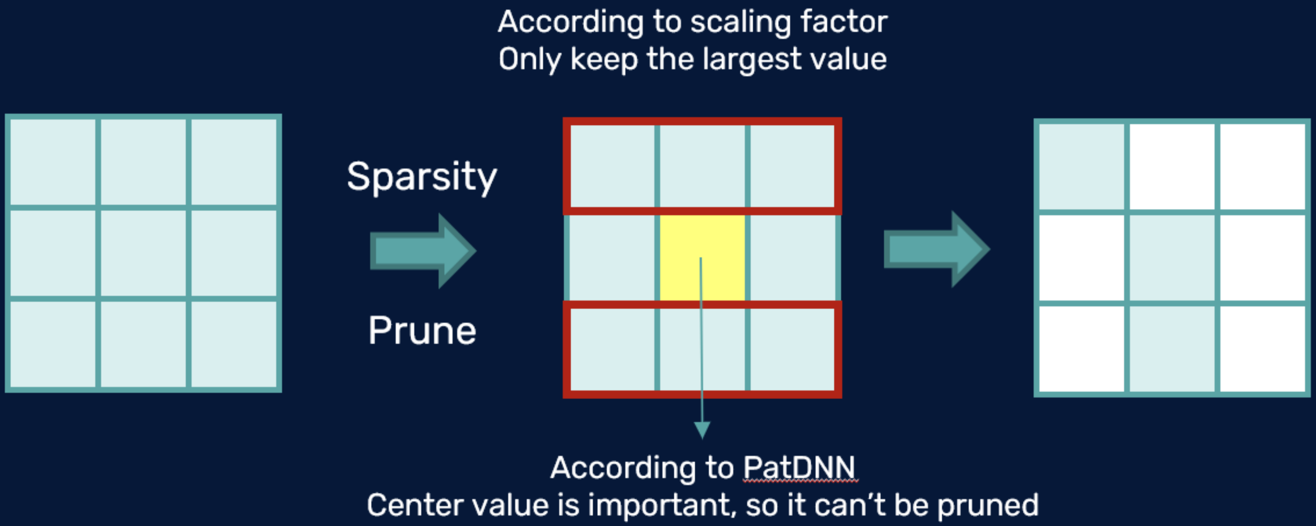
Saliency-based pruning:



```
if  SPARSITY:
        updateConv2D()
        train(1)
        test(sparsity_model)
        maskConv2D()
        test(model)
        test(sparsity_model)
```

Pruning

Saliency-based pruning:

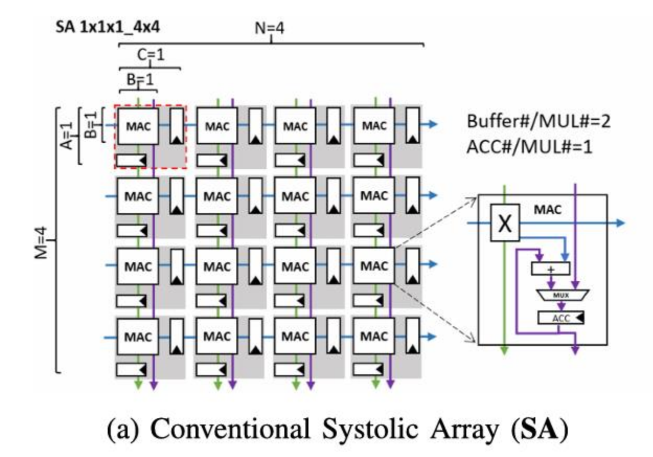

According to scaling factor
Only keep the largest value
Sparsity
Prune
According to PatDNN
Center value is important, so it can't be pruned

02

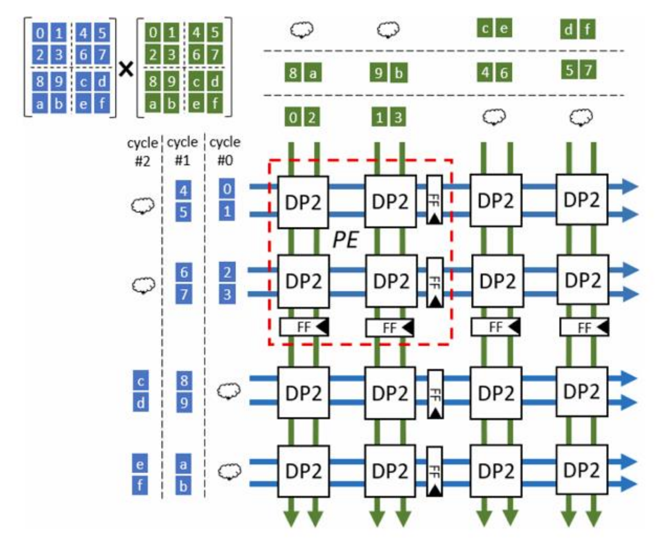
PE Architecture and Dataflow

GEMM Accelerator

For Dense Matrix



(a) Conventional Systolic Array (SA)



Systolic Tensor Array

GEMM Accelerator

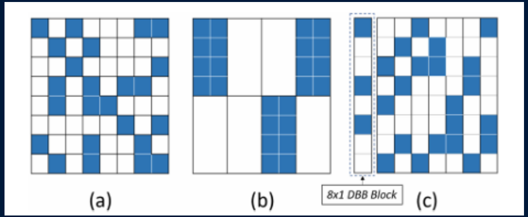
For Sparse Matrix

DBB | Density-Bound Block

Constraint mechanism for

the number of non-zero values

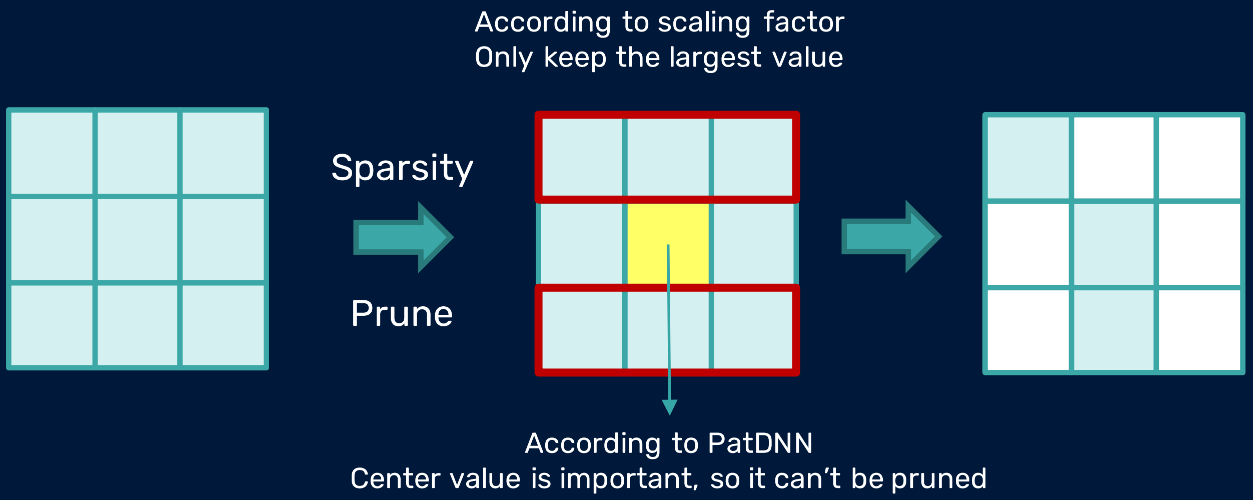
permissible in each block of a matrix.



GEMM Accelerator

For Sparse Matrix

Our DBB Method

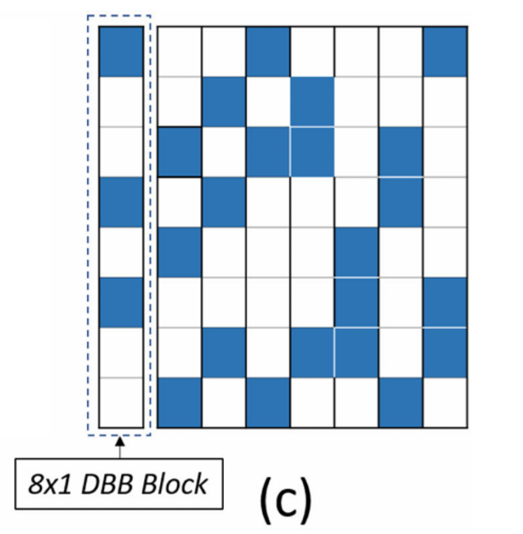


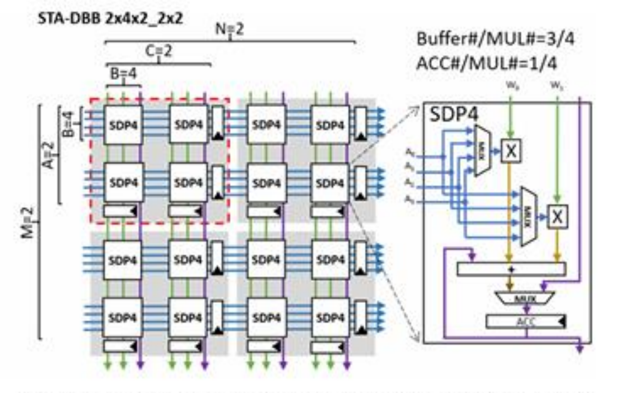
According to scaling factor
Only keep the largest value

According to PatDNN
Center value is important, so it can't be pruned

GEMM Accelerator

For Sparse Matrix



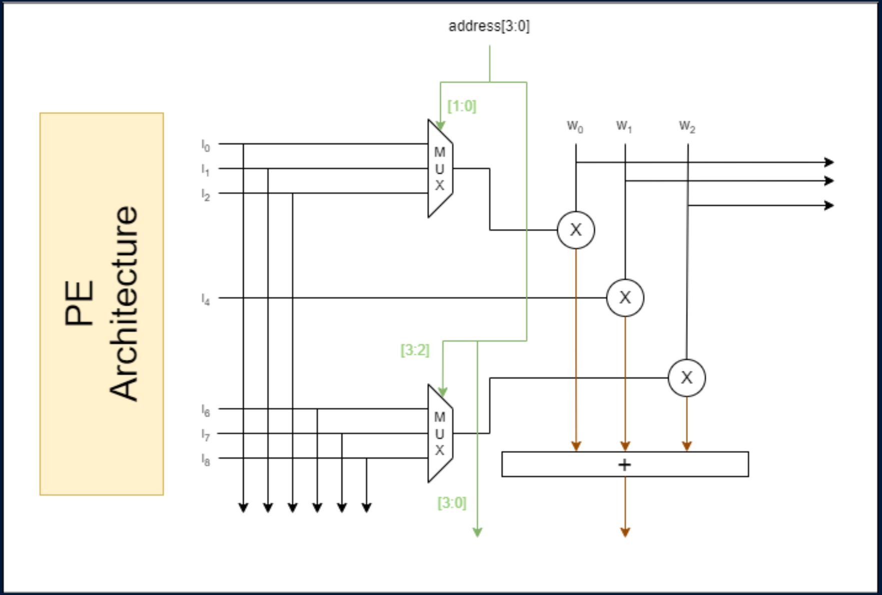


(c) Systolic Tensor Array for DBB (**STA-DBB**)

PE Architecture

I_0	I_1	I_2
I_3	I_4	I_5
I_6	I_7	I_8

W_0	W_0	W_0
	W_1	
W_2	W_2	W_2

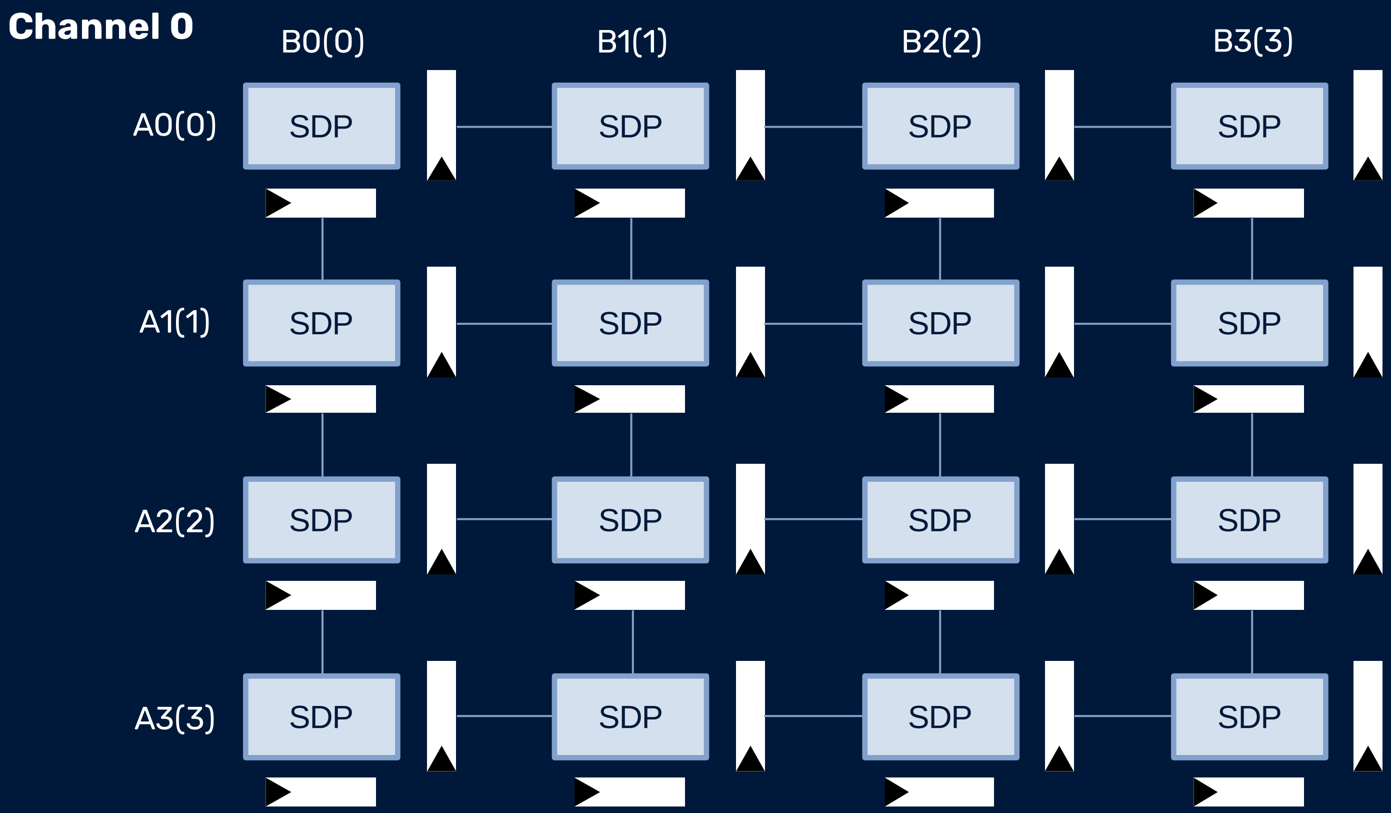


PE I/O Ports

Signal	I/O	Bits	Function
ifmap_i	Input	56 bits	I_0 -> [7:0] $I_0, I_1, I_2, I_4, I_6, I_7, I_8$
weight_i	Input	24 bits	W_0 -> [7:0] W_0, W_1, W_2
address_i	Input	4 bits	[1:0] for upper weight [3:2] for lower weight
ifmap_o	Output	56 bits	
weight_o	Output	24 bits	
address_o	Output	4 bits	
result	Output	18 bits	Product and Add

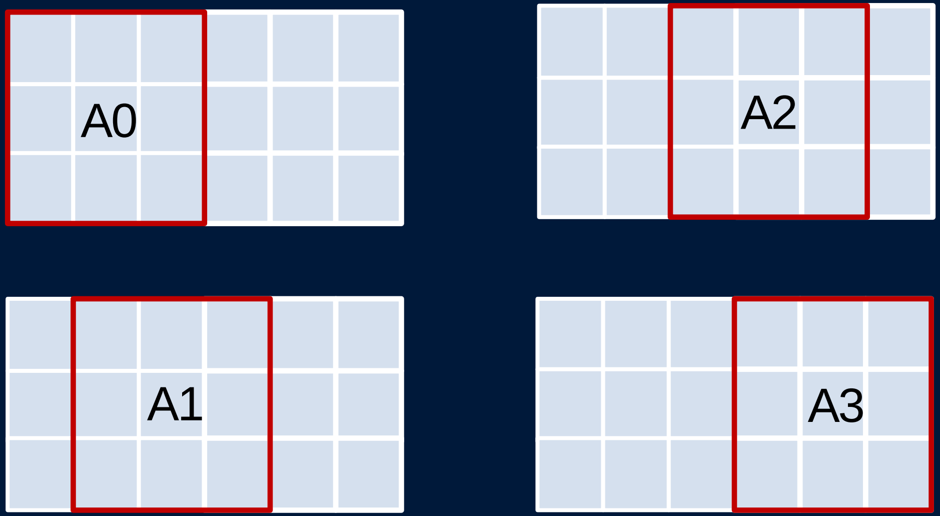
PE Array Architecture

A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send



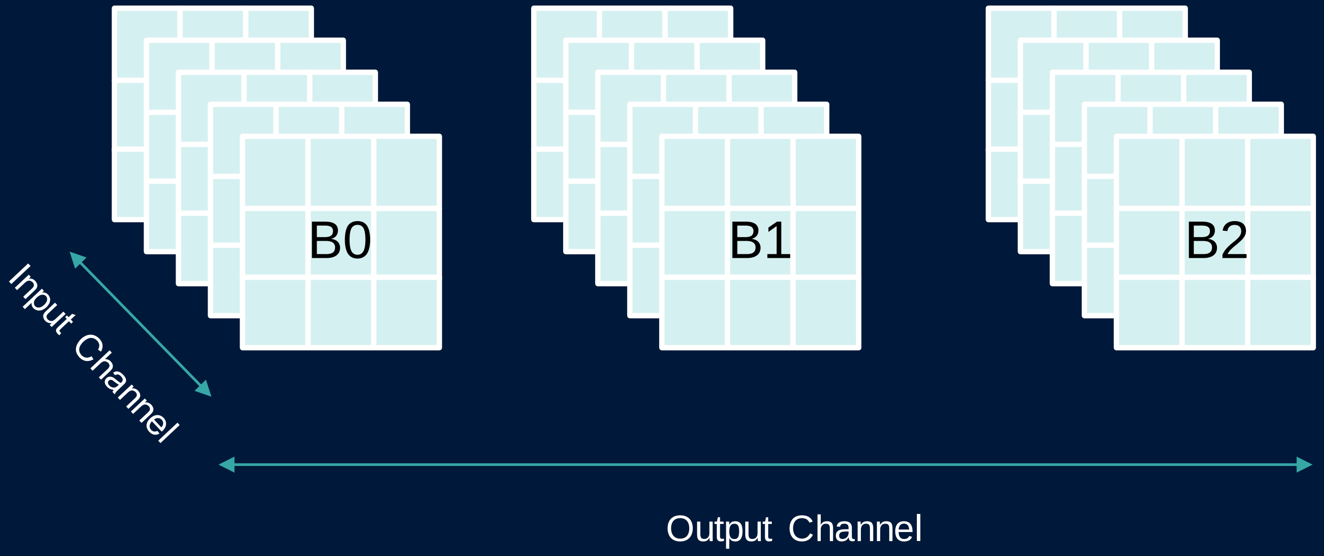
PE Array Architecture

A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send



PE Array Architecture

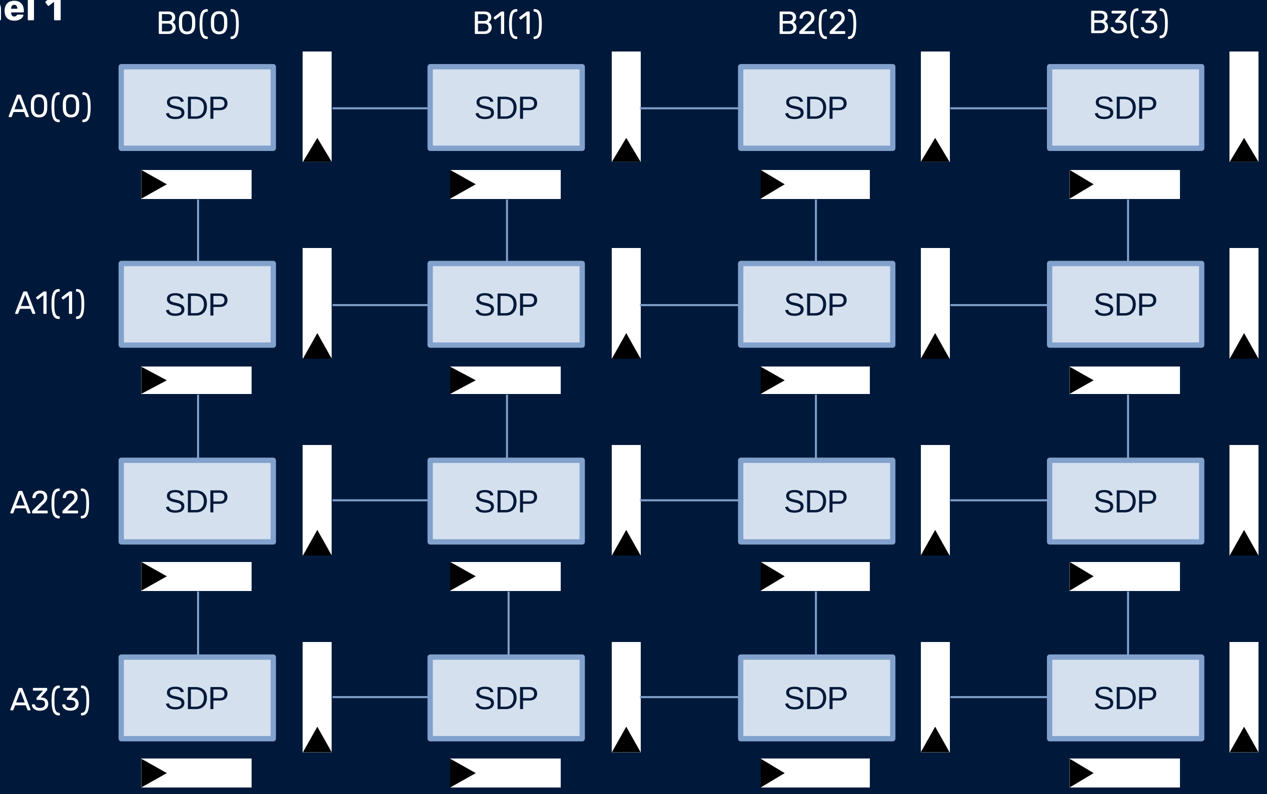
A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send



PE Array Architecture

A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send

Channel 1

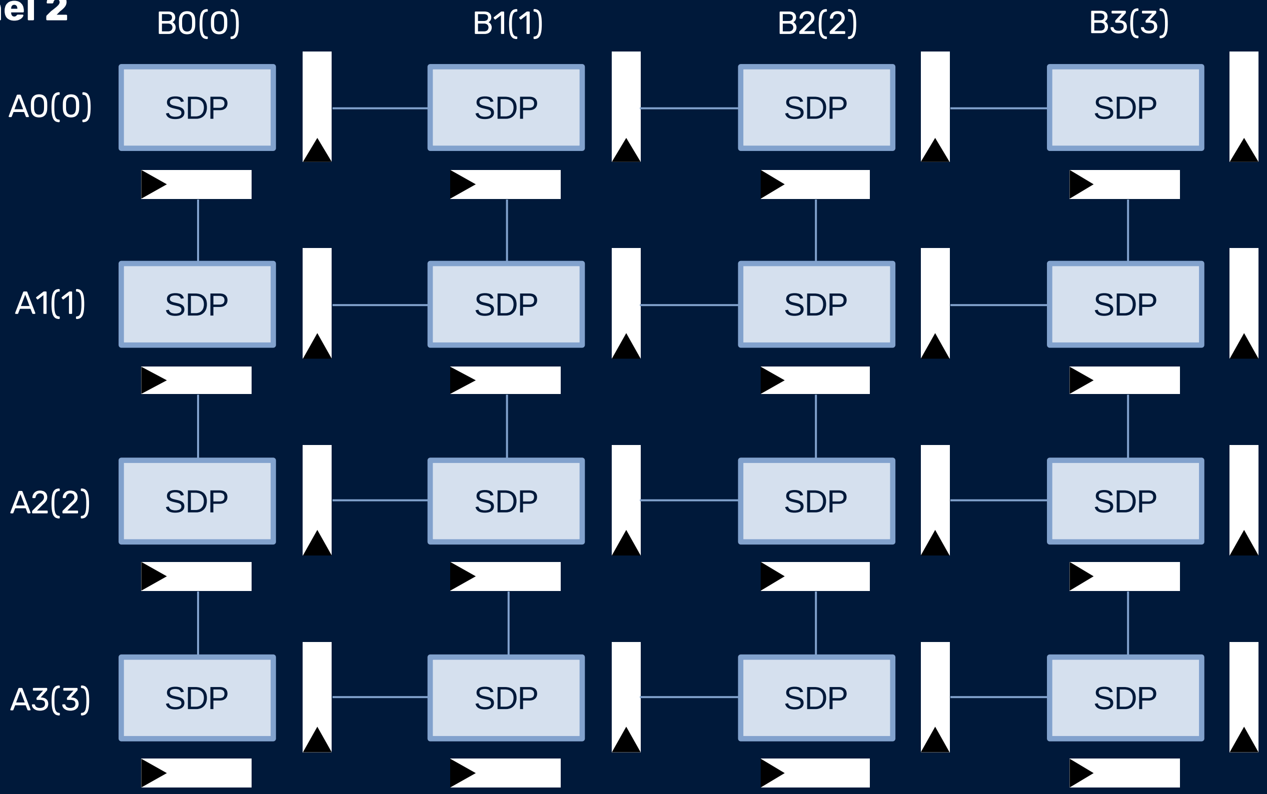


	B0(0)	B1(1)	B2(2)	B3(3)
A0(0)	SDP	SDP	SDP	SDP
A1(1)	SDP	SDP	SDP	SDP
A2(2)	SDP	SDP	SDP	SDP
A3(3)	SDP	SDP	SDP	SDP

PE Array Architecture

A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send

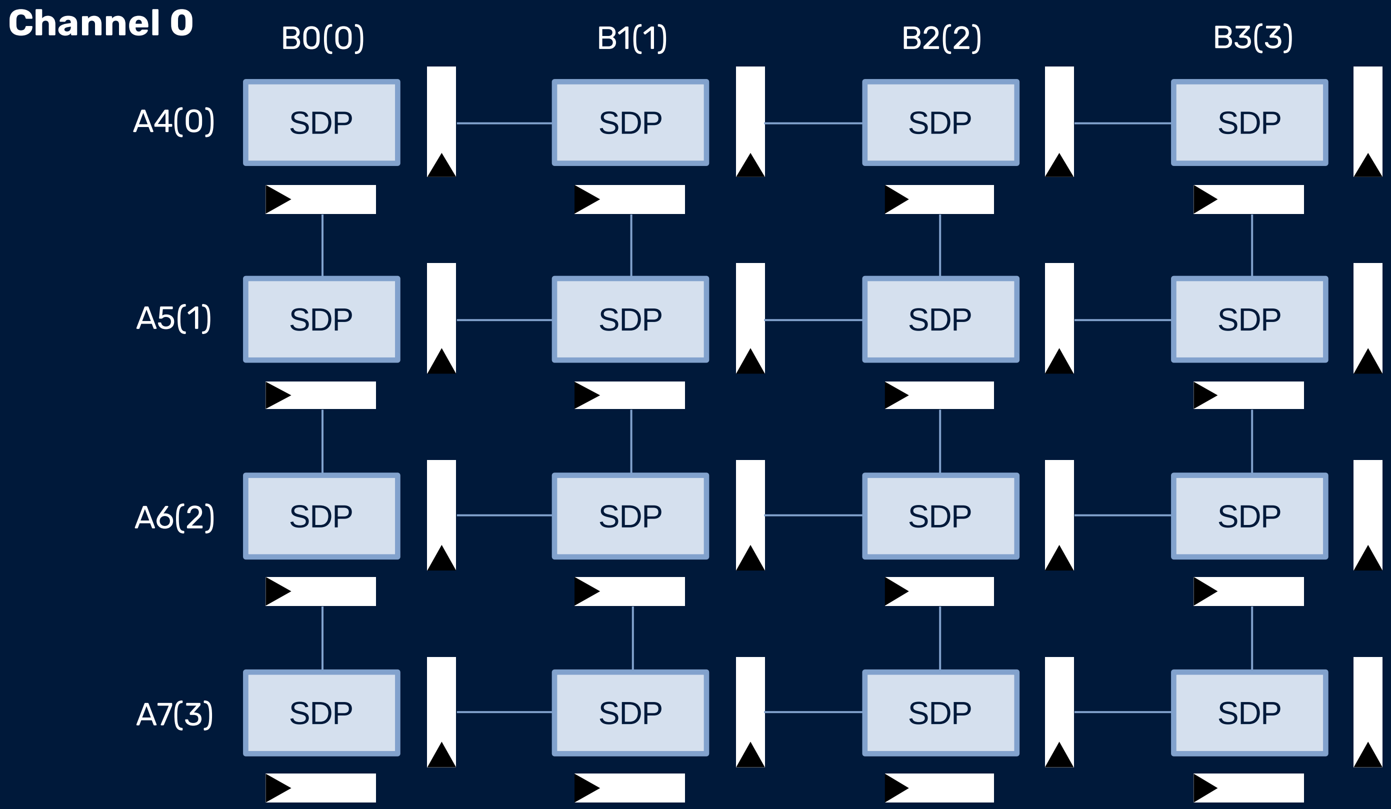
Channel 2



	B0(0)	B1(1)	B2(2)	B3(3)
A0(0)	SDP	SDP	SDP	SDP
A1(1)	SDP	SDP	SDP	SDP
A2(2)	SDP	SDP	SDP	SDP
A3(3)	SDP	SDP	SDP	SDP

PE Array Architecture

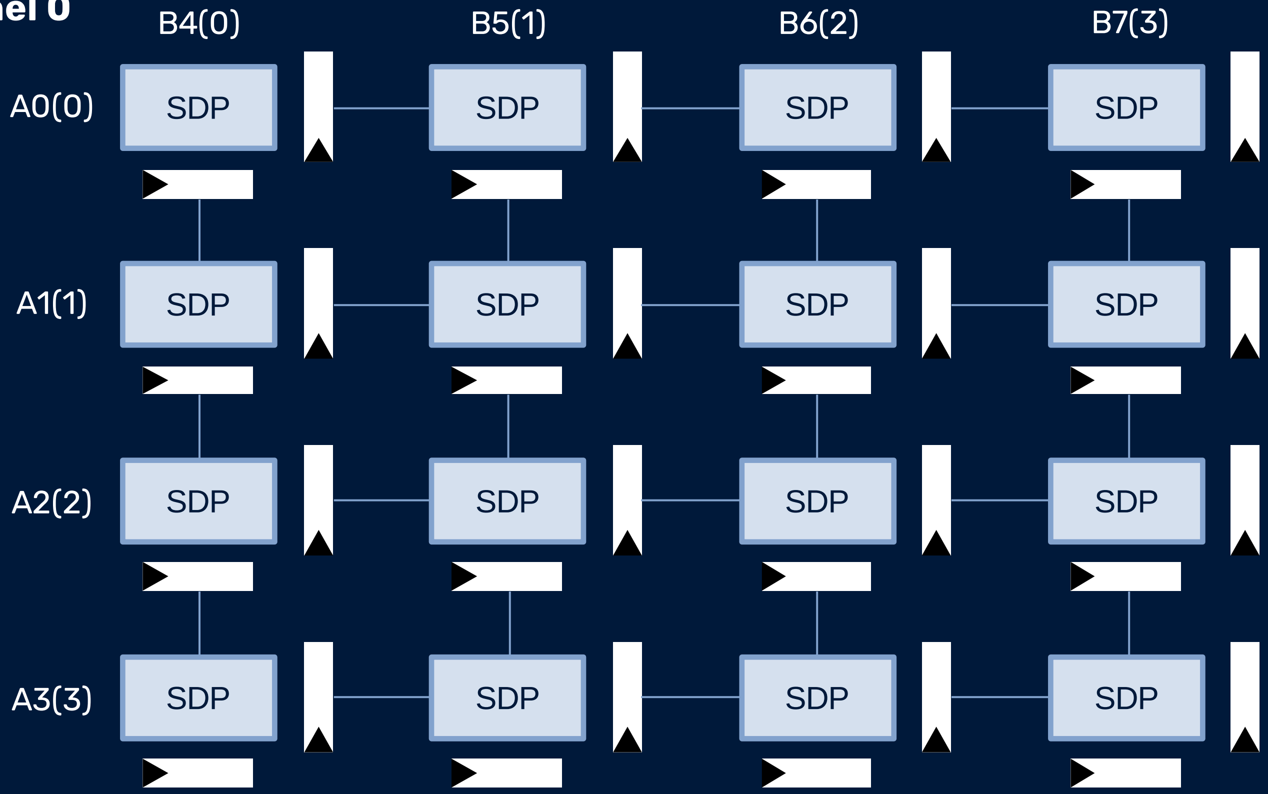
A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send



PE Array Architecture

A for 3*3 Ifmap Sliding Window
B for 3*3 Weight
(num) num :which clock to send

Channel 0



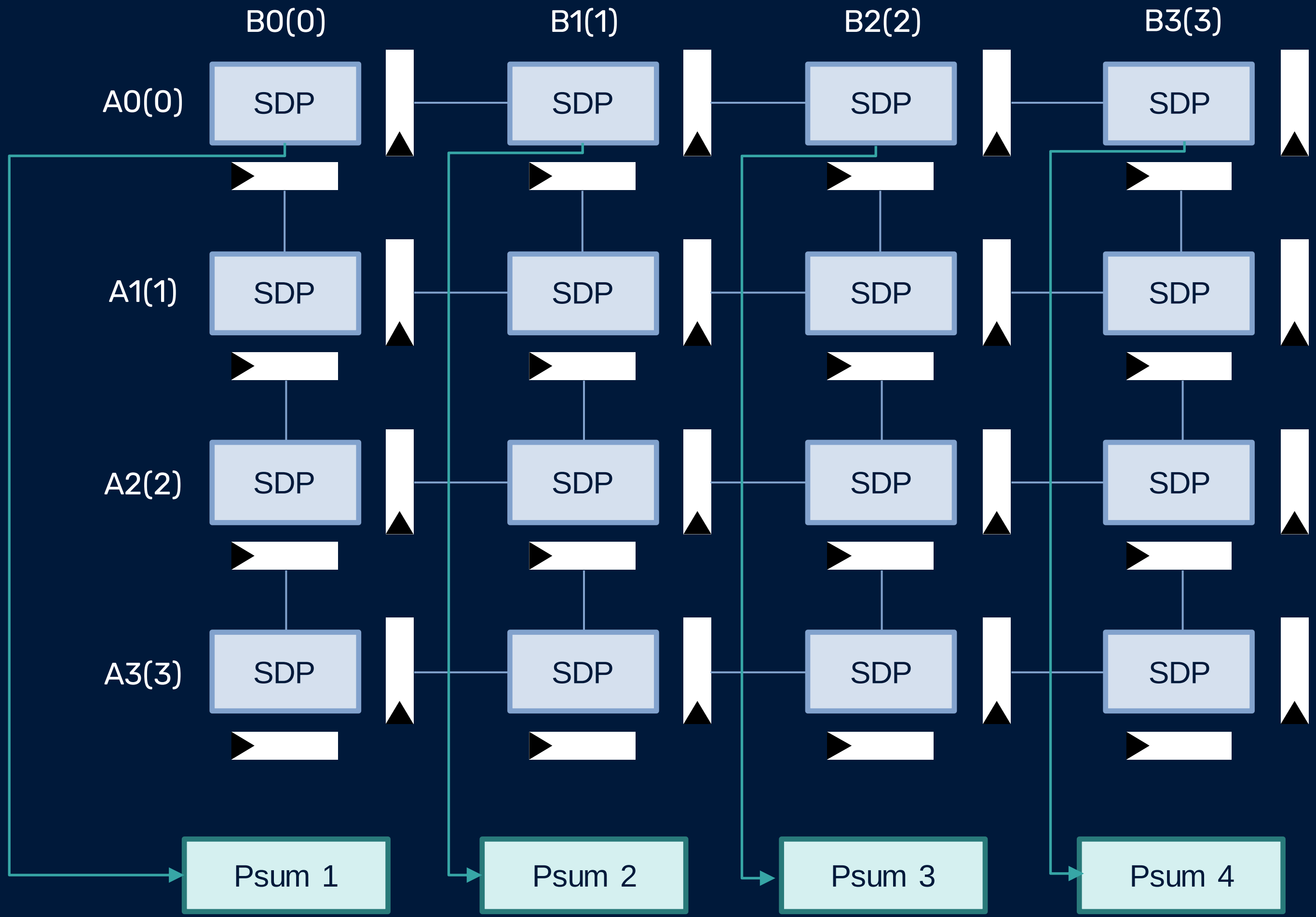
B4(0) B5(1) B6(2) B7(3)

A0(0) SDP SDP SDP SDP

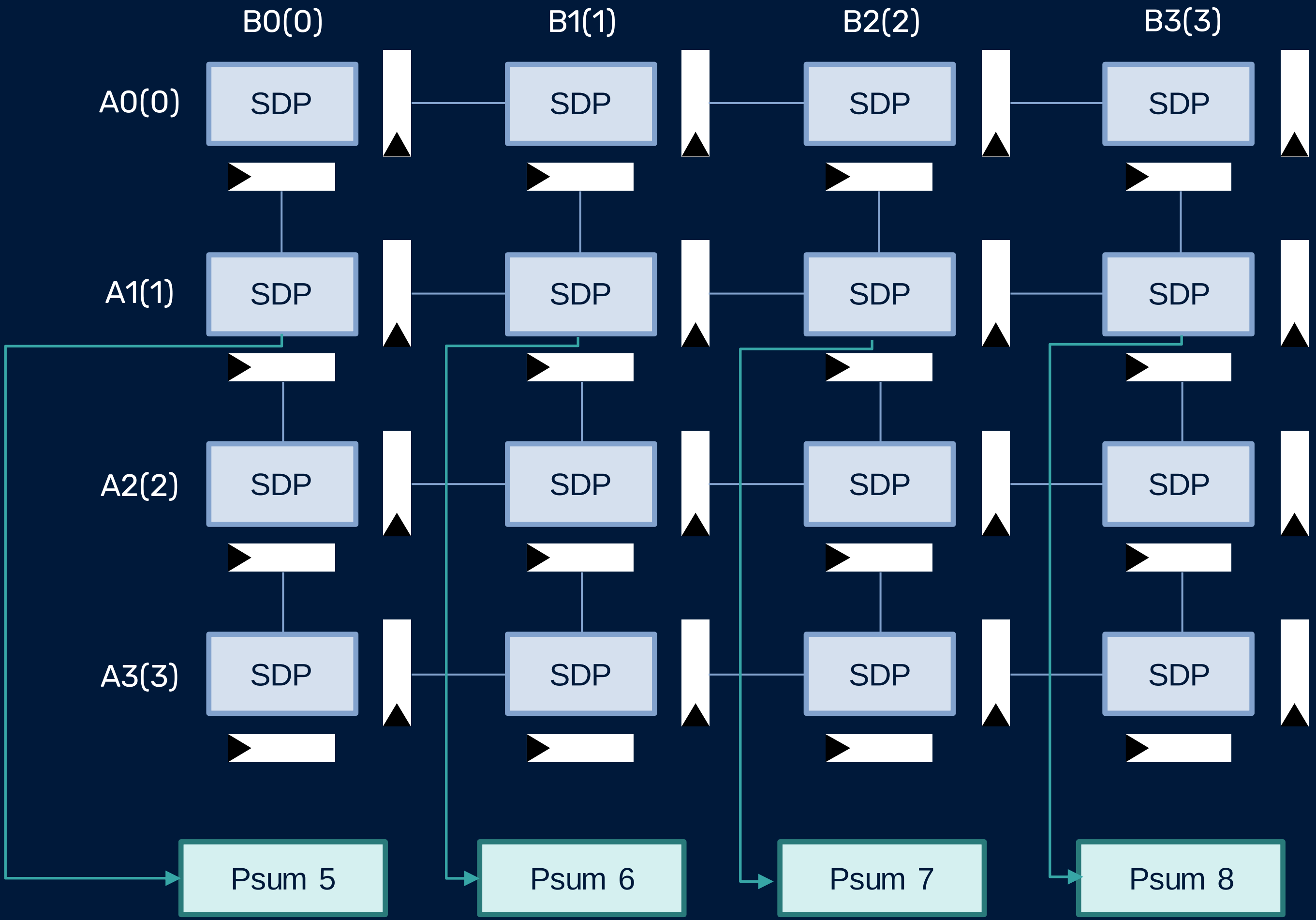
A1(1) SDP SDP SDP SDP

A2(2) SDP SDP SDP SDP

A3(3) SDP SDP SDP SDP



	B0(0)	B1(1)	B2(2)	B3(3)
A0(0)	SDP	SDP	SDP	SDP
A1(1)	SDP	SDP	SDP	SDP
A2(2)	SDP	SDP	SDP	SDP
A3(3)	SDP	SDP	SDP	SDP
	Psum 1	Psum 2	Psum 3	Psum 4

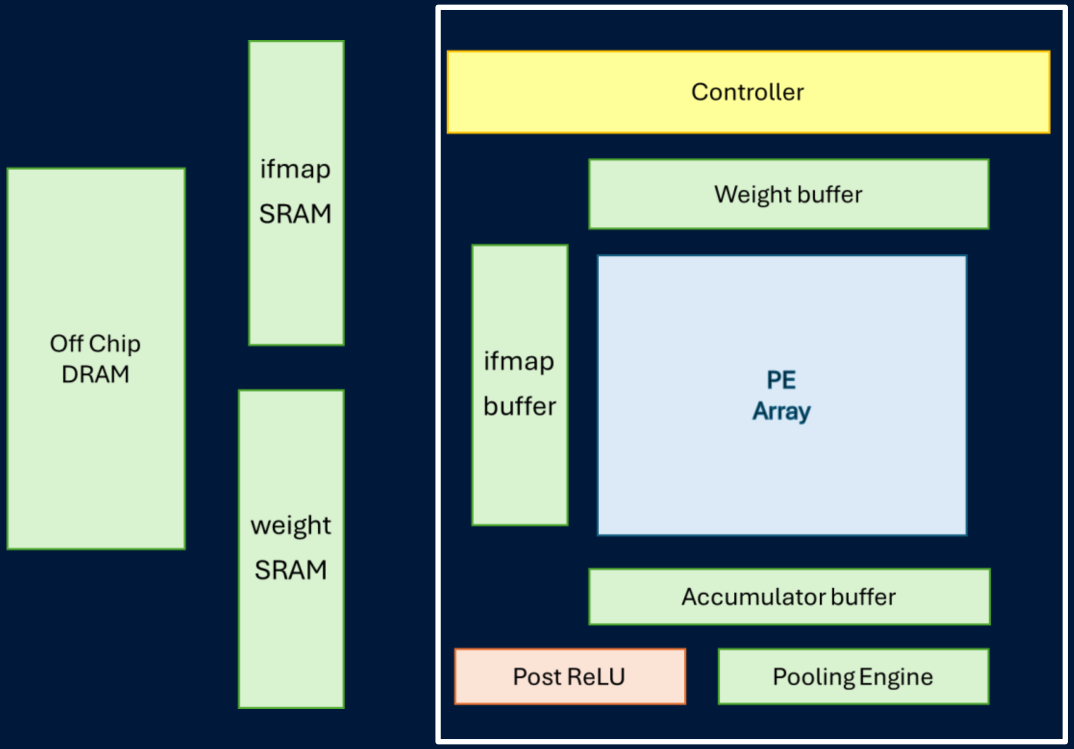


	B0(0)	B1(1)	B2(2)	B3(3)
A0(0)	SDP	SDP	SDP	SDP
A1(1)	SDP	SDP	SDP	SDP
A2(2)	SDP	SDP	SDP	SDP
A3(3)	SDP	SDP	SDP	SDP
	Psum 5	Psum 6	Psum 7	Psum 8

03

Our Accelerator Architecture and SPEC

Architecture



SPEC

SRAM

Type	Size	Why we use this size?
Ifmap SRAM	16KB	512(input channel) * 3(filter row) • 10(6+4) * 1Byte = 15KB
Weight SRAM	16KB	512(input channel) * 8(output channel) * 3(NNZ) * 10(int8+2 bit address) = 15KB

SPEC

On-Chip Buffer

Type	Size	Why we use this size?
Ifmap Buffer	576B	32(input channel) * 3(filter row) * 6(ifmap column) * 1Byte = 576B
Weight Buffer	480B	32(input channel) * 3(filter NNZ) * 4(PE a row) * 10bit (int8 + 2 bit address) = 480B
Accumulator Buffer	64B	32 Bits * 16 (PE) = 64B

SPEC

Bandwidth

Type	Size	Why we use this size?
PE to Ifmap Buffer	28B	7(element)*4(PE a row)*1Byte = 28B
PE to Weight Buffer	15B	3(NNZ in a filter) * 4(PE a col) * 10 bits = 15B
PE to Accumulator Buffer	64B	4Byte * 16 = 64B

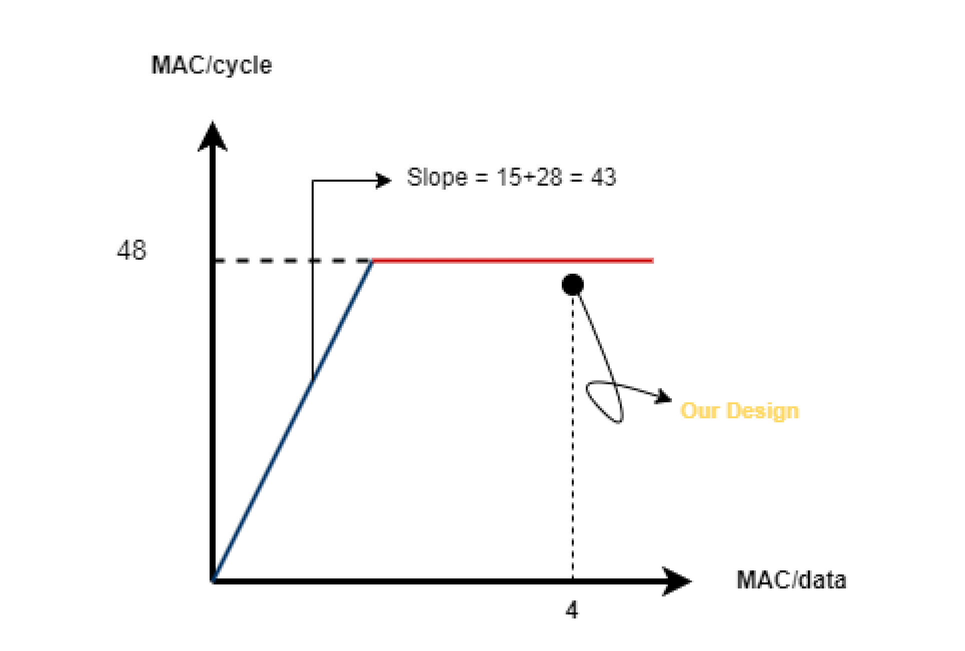
04

Analysis and Question

Advantages

- Increased Efficiency
- Enhanced Area and Power Efficiency
- Efficient Data Flow

Roofline Model



Problem Disscusion

Q1: Is the Accumulator on-chip bandwidth (64B) too large?

Q2: Is the number of times data is fetched from DRAM too many?

(Up to 64 times per layer)

RESOURCES

Course

- AI on Chip NCKU by Professor Chia-Chi Tsai

Paper Reference

- **Liu, Z. G., Whatmough, P. N., & Mattina, M. (2020). Systolic tensor array: An efficient structured-sparse GEMM accelerator for mobile CNN inference. IEEE Computer Architecture Letters, 19(1), 34-37.**
- **Niu, W., Ma, X., Lin, S., Wang, S., Qian, X., Lin, X., ... & Ren, B. (2020, March). Patdnn: Achieving real-time dnn execution on mobile devices with pattern-based weight pruning. In Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (pp. 907-922).**

THANKS!