ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

MINT-FX
Microscaling Integer Accelerator Template with Flexible Nonlinear Function Execution

Lionnus Kesting, lkesting@iis.ee.ethz.ch

Arpan Suravi Prasad, prasadar@iis.ee.ethz.ch

Prof. Dr. Francesco Conti, f.conti@unibo.it

Prof. Dr. Luca Benini, lbenini@iis.ee.ethz.ch

pulp-platform.org

@pulp_platform

company/pulp-platform

youtube.com/pulp_platform

PULP Platform

Open Source Hardware, the way it should be!

ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Transformer inference at the edge



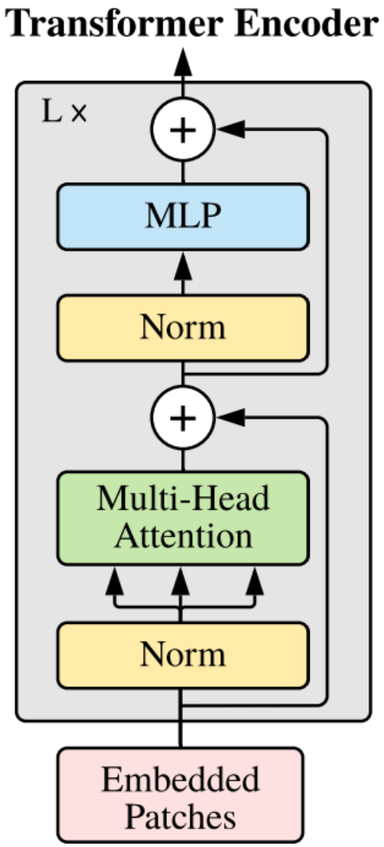


Edge devices such as **smart glasses and nano-drones** are new targets for **on-device transformer inference**

Inference is moving to the edge under **tight compute, latency, and energy budgets**

Microscaling formats, such as MXINT8, bring **near-FP32 accuracy** with **reduced area and energy implementations**





→ There is not only MXINT8 GEMM in transformers, but also **non-linearities**

Images: upper left, Ray-Ban Meta Wayfarer (meta.com); bottom left, GAP9Shield on Crazyflie 2.1, Müller et al., ETH Zürich, 2024; right, ViT encoder, Dosovitskiy et al., ICLR 2021.

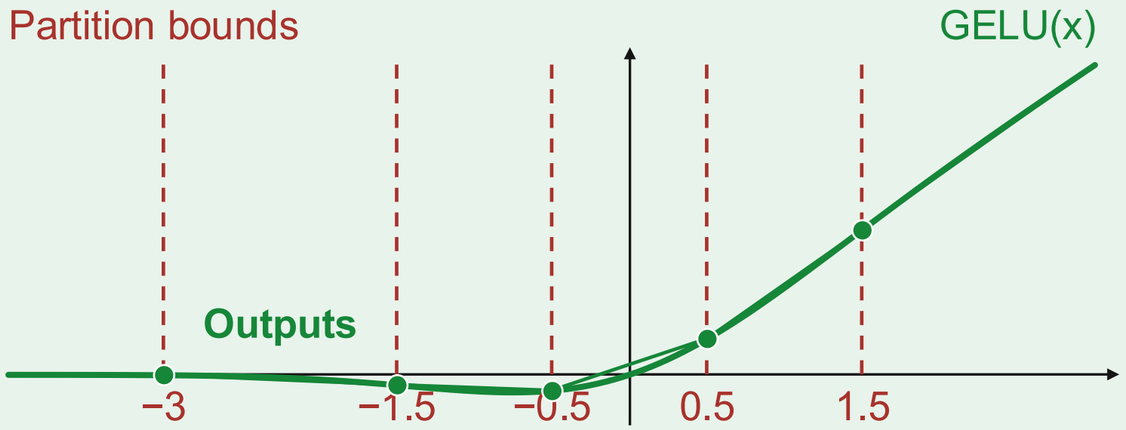
Transformers contain GEMMs and nonlinear functions

GEMM Nonlinear

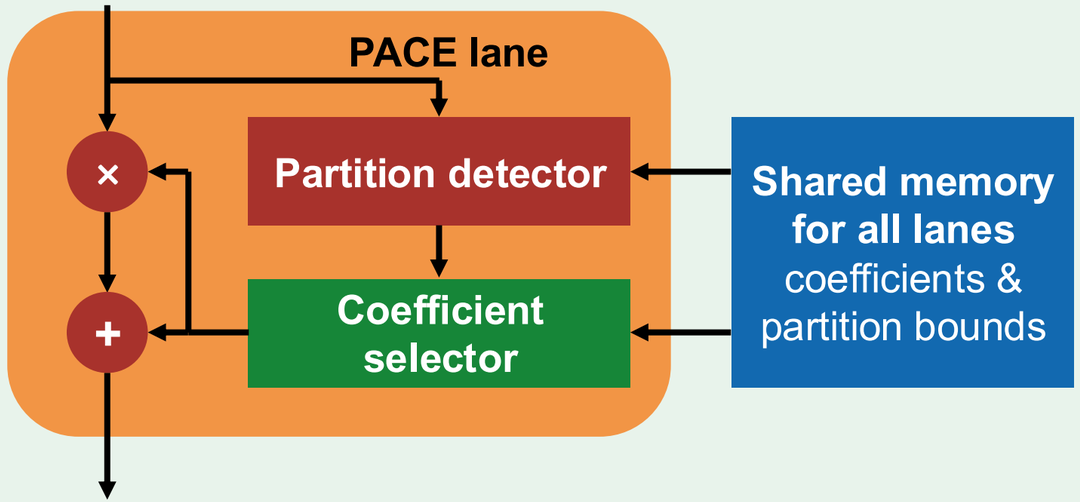


Accelerate nonlinear functions using Piecewise Polynomial Approximation (PwPA)

GELU Example



→ We use **PACE**[1] in BF16 as our PwPA unit

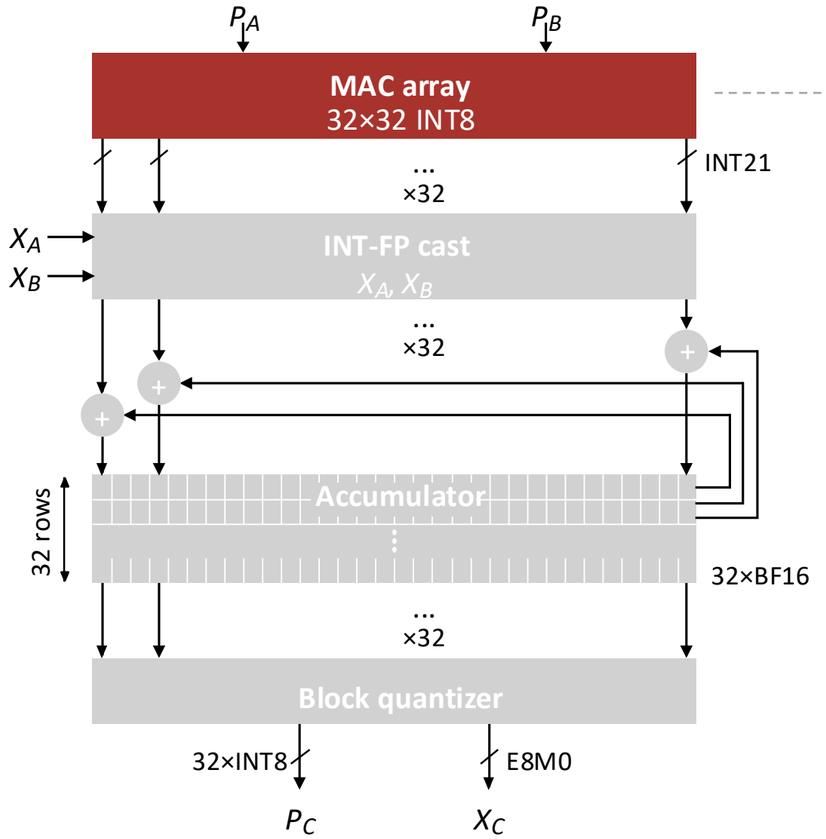


→ We propose MINT-FX: **MXINT8 accelerator template that runs both GEMM and PwPA nonlinear functions**

[1] A. S. Prasad et al., "PACE: An optimal piecewise polynomial approximation unit for flexible and efficient transformer non-linearity acceleration," in IEEE ISVLSI, 2025.

ETH zürich ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

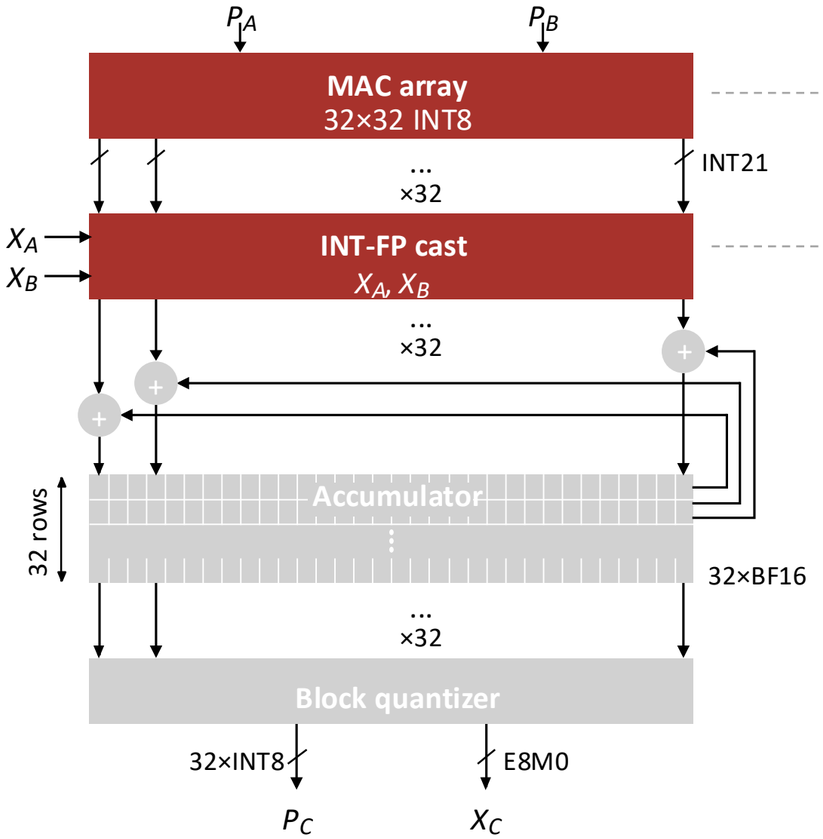
MXINT8 Accelerator Architecture



1

MAC array: 1,024 INT8 products per cycle; an integer tree reduces each 32-element MX block

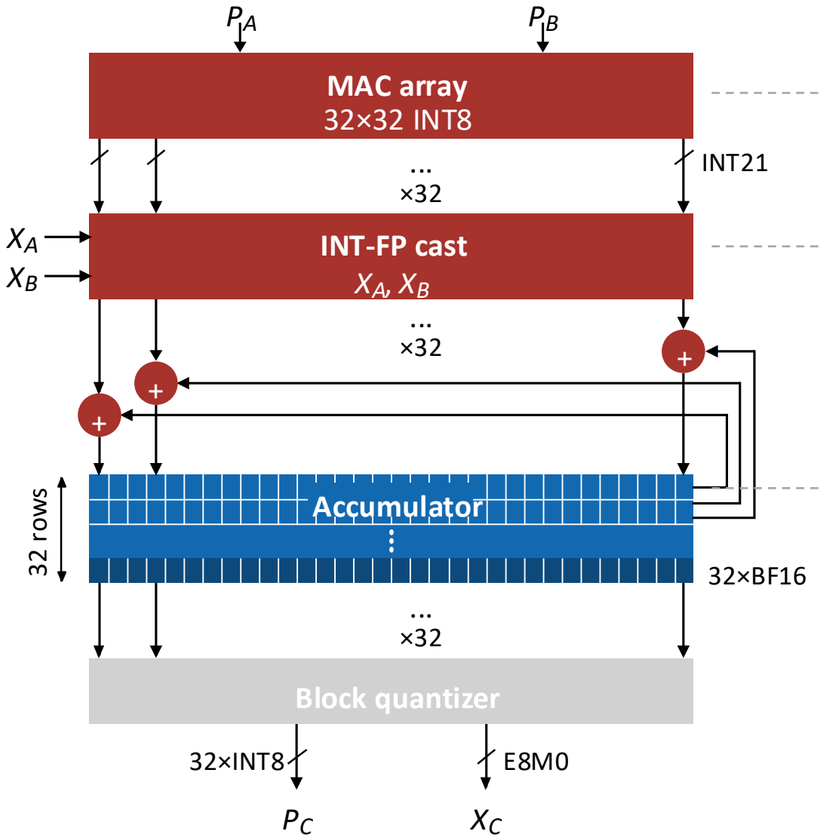
MXINT8 Accelerator Architecture



1 **MAC array:** 1,024 INT8 products per cycle; an integer tree reduces each 32-element MX block

2 **INT-to-BF16 cast:** each block sum is cast to BF16; the E8M0 scales X_A and X_B are added to the exponent

MXINT8 Accelerator Architecture

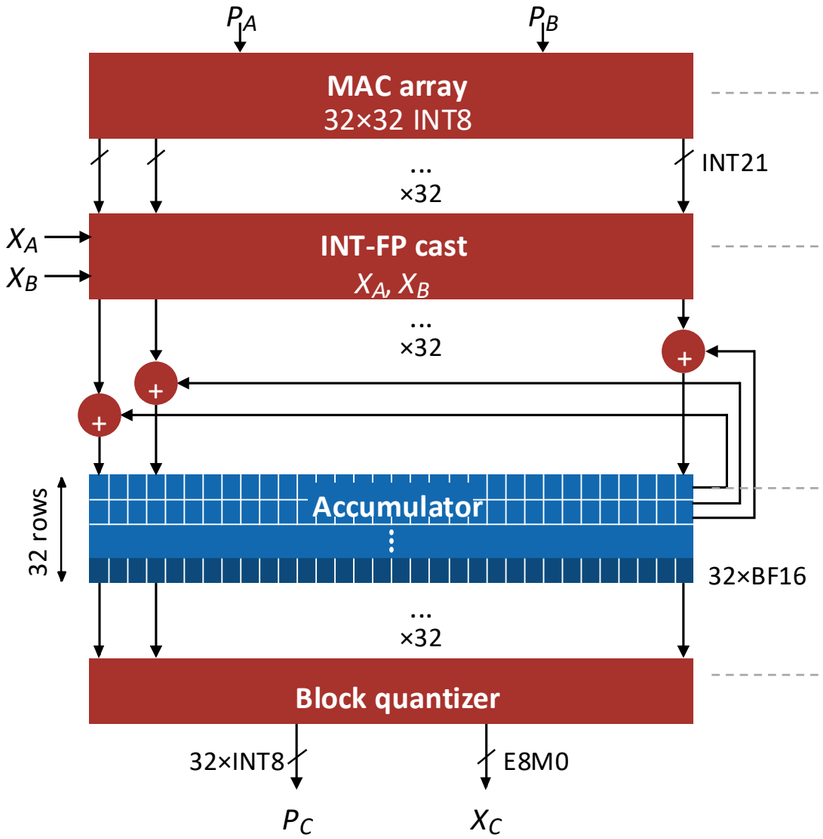


1 **MAC array:** 1,024 INT8 products per cycle; an integer tree reduces each 32-element MX block

2 **INT-to-BF16 cast:** each block sum is cast to BF16; the E8M0 scales X_A and X_B are added to the exponent

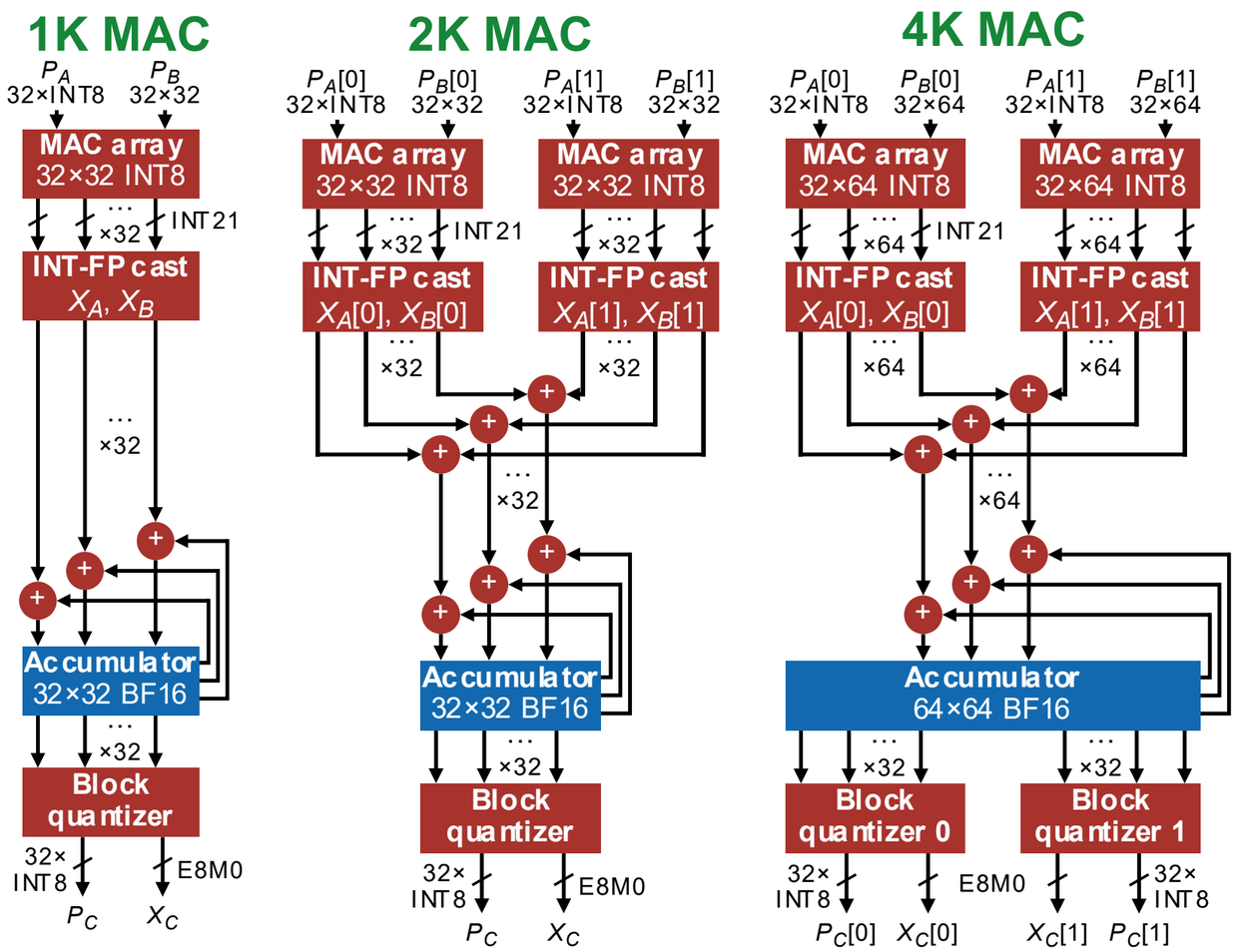
3 **BF16 accumulator:** 32×32 BF16 entries accumulate along the inner dimension and hold whole MX blocks

MXINT8 Accelerator Architecture



1. **MAC array:** 1,024 INT8 products per cycle; an integer tree reduces each 32-element MX block
2. **INT-to-BF16 cast:** each block sum is cast to BF16; the E8M0 scales X_A and X_B are added to the exponent
3. **BF16 accumulator:** 32×32 BF16 entries accumulate along the inner dimension and hold whole MX blocks
4. **Block quantizer:** after the full tile accumulation, results stream out as MXINT8 using a quantizer module

MXINT8 Accelerator Architecture



1K MAC array 32 × 32

Reduction depth 32 matches one MX block; 32 output lanes, one block quantizer

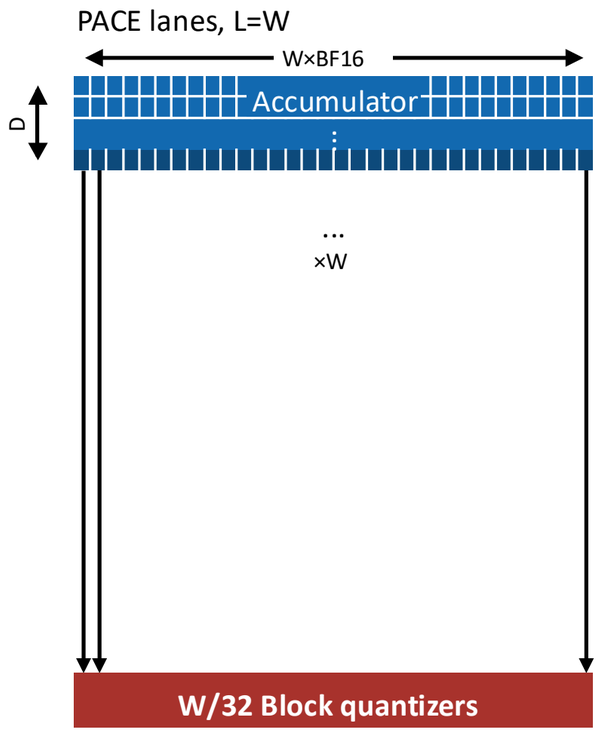
2K MAC array 64 × 32

Two integer trees, merged by one BF16 add per lane; still 32 output lanes

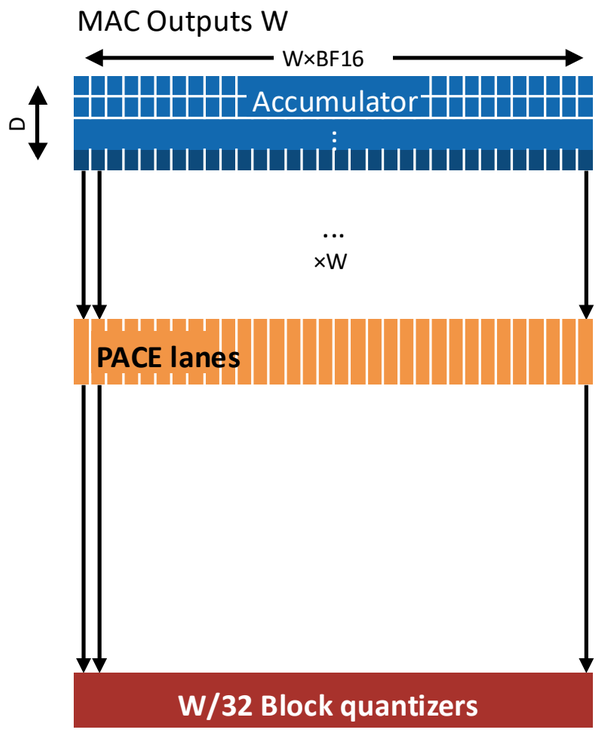
4K MAC array 64 × 64

Two trees and 64 output lanes; 64 × 64 BF16 accumulator, two block quantizers

Piecewise Polynomial Approximation (PwPA) Integration



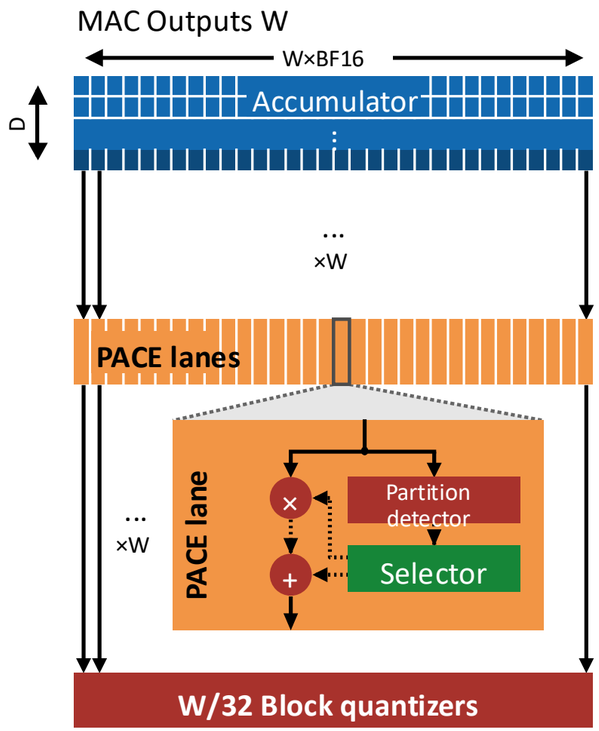
Piecewise Polynomial Approximation (PwPA) Integration



PACE integrated between accumulators and block quantizers

→ Softmax and LayerNorm re-using units dedicated dataflow control

Piecewise Polynomial Approximation (PwPA) Integration

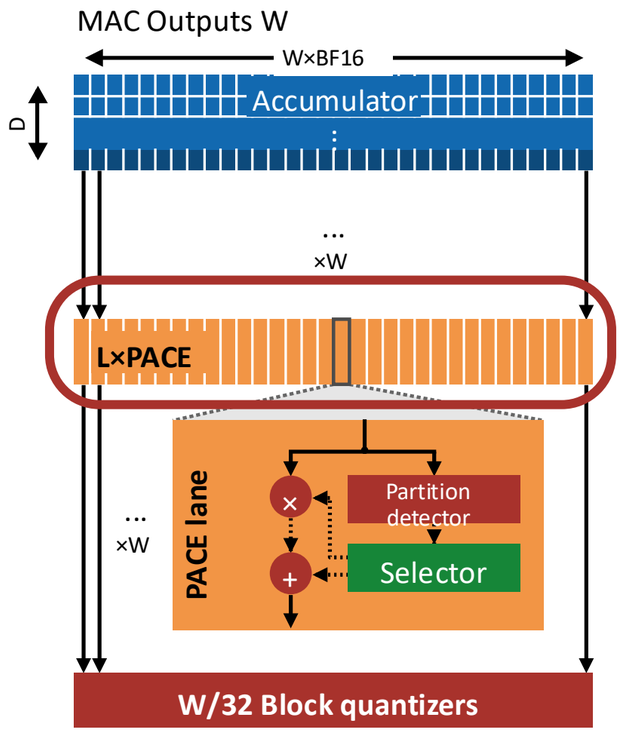


PACE integrated between accumulators and block quantizers

→ Softmax and LayerNorm re-using units dedicated dataflow control

PACE contains a BF16 FMA, partition detector and selector

Piecewise Polynomial Approximation (PwPA) Integration



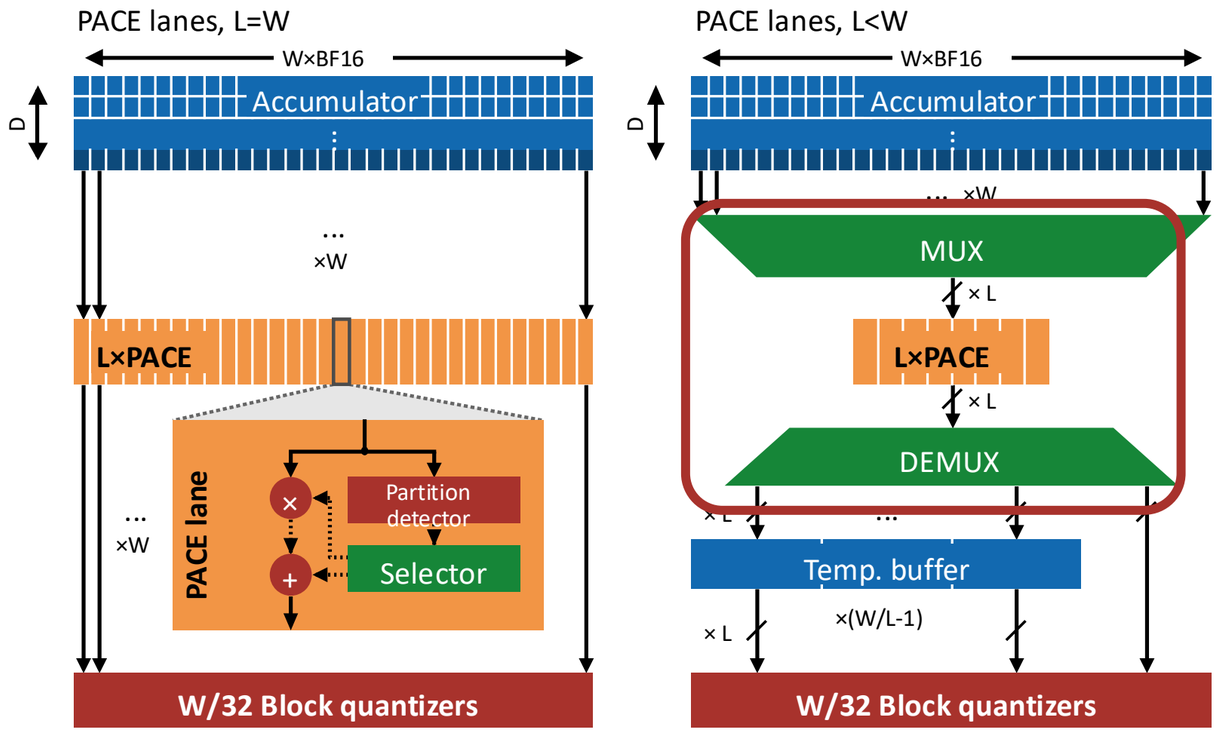
PACE integrated between accumulators and block quantizers

→ Softmax and LayerNorm re-using units dedicated dataflow control

PACE contains a BF16 FMA, partition detector and selector

Full lanes (L = W): each MAC output lane has its own PACE lane

Piecewise Polynomial Approximation (PwPA) Integration



PACE integrated between accumulators and block quantizers

→ Softmax and LayerNorm re-using units dedicated dataflow control

PACE contains a BF16 FMA, partition detector and selector

Full lanes (L = W): each MAC output lane has its own PACE lane

Fewer lanes (L < W): output lanes share PACE through a MUX and buffer: less area for more cycles

Lane ratio L/W: PACE lanes L per MAC output lanes W; full lanes means L/W = 1

→ Analyze how we can use this hardware for our target workloads

Hardware/Software Design Space Exploration

Design space exploration

MAC array	Lane ratio L/W	PwPA segments S
1K, 2K and 4K	**1, ½, ¼ and ⅛**	**4, 8, 16, 32 and 64**

Software accuracy evaluation

- PyTorch model emulating:
  - **BF16 PwPA**
  - **MXINT8**
  - **BF16 accumulation**
- 6 ViTs on ImageNet-1k, 2 language models on QNLI
- Compared with FP32 checkpoints, no fine-tuning

Hardware implementation

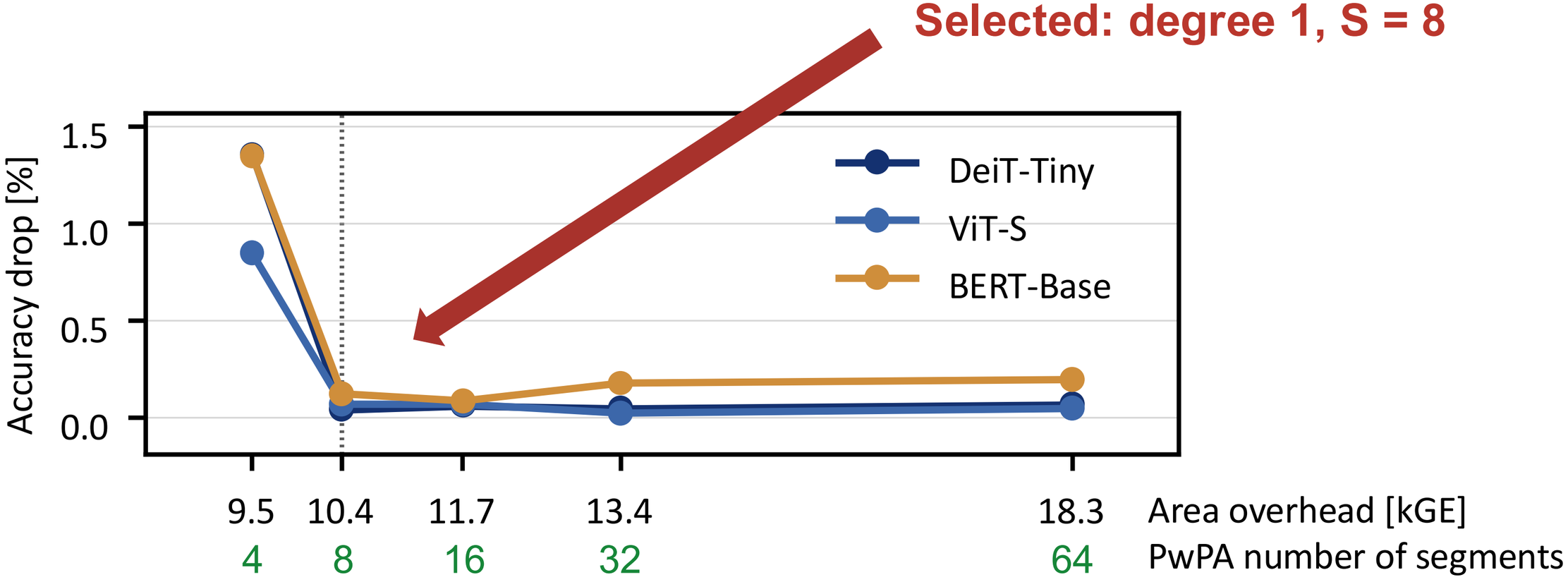
- GlobalFoundries 12 nm at 1 GHz post-layout
- RTL simulation for kernel cycles
- Post-layout power simulation for energy

Accuracy vs. PwPA segments

Area, cycles, energy

Design point selection: end-to-end latency and energy vs. area overhead, per array size

PwPA approximation accuracy



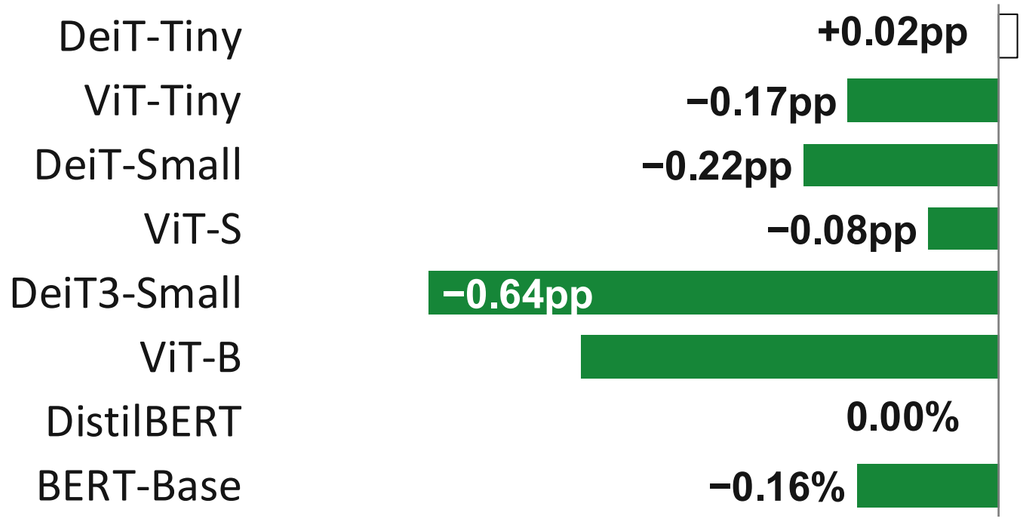
Workload accuracy analysis

1. Isolated effect of approximations

Datapath	DeiT-Tiny	Δ
FP32 checkpoint	72.19	–
MXINT8 only	72.13	−0.06
+ BF16 accumulate	72.08	−0.11
+ BF16 PwPA Nonlinearities	72.12	−0.07
Full datapath	**72.21**	**+0.02**

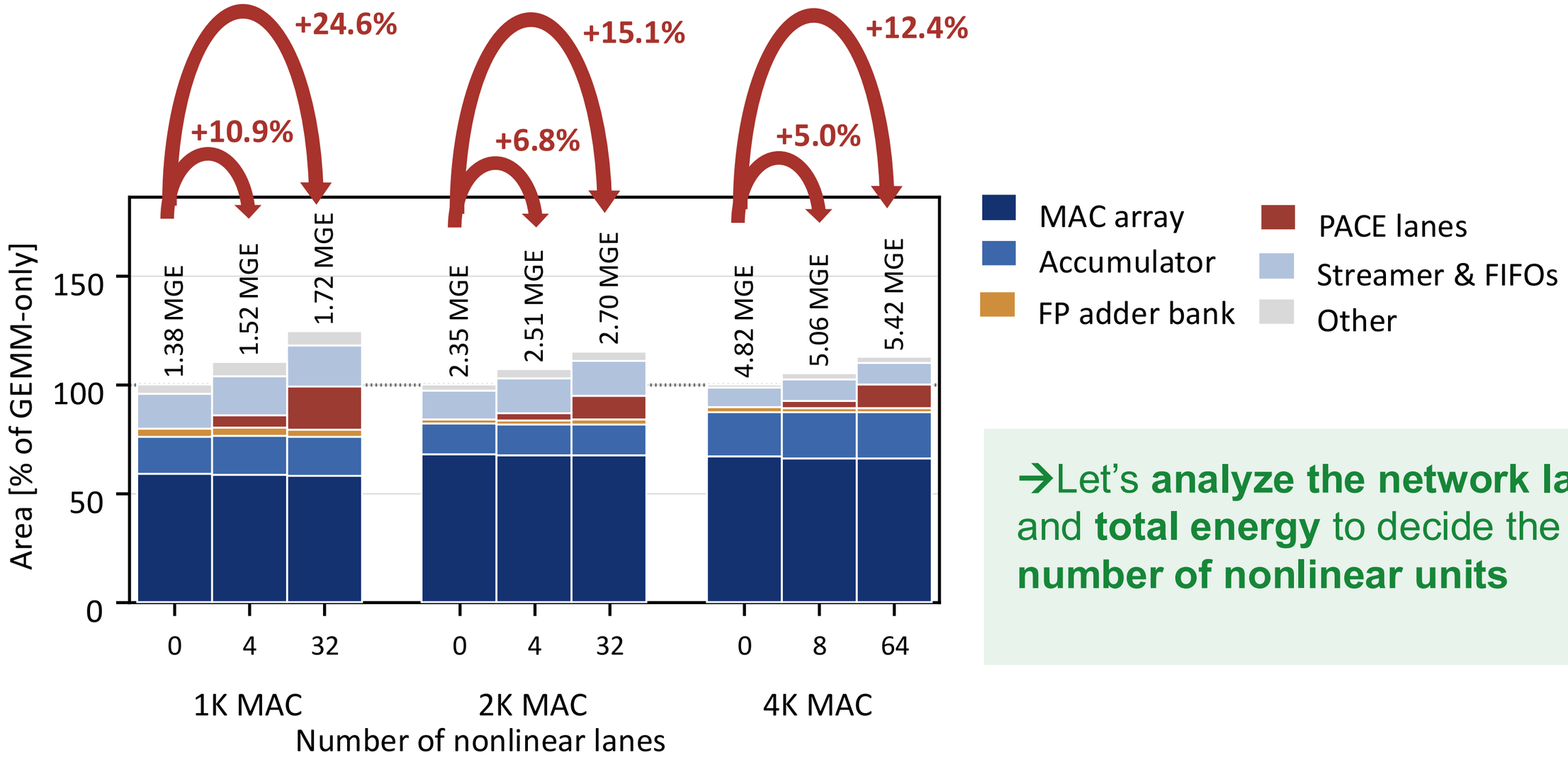
2. Accuracy across 6 ViTs & 2 language models

Accuracy change vs. FP32 checkpoint



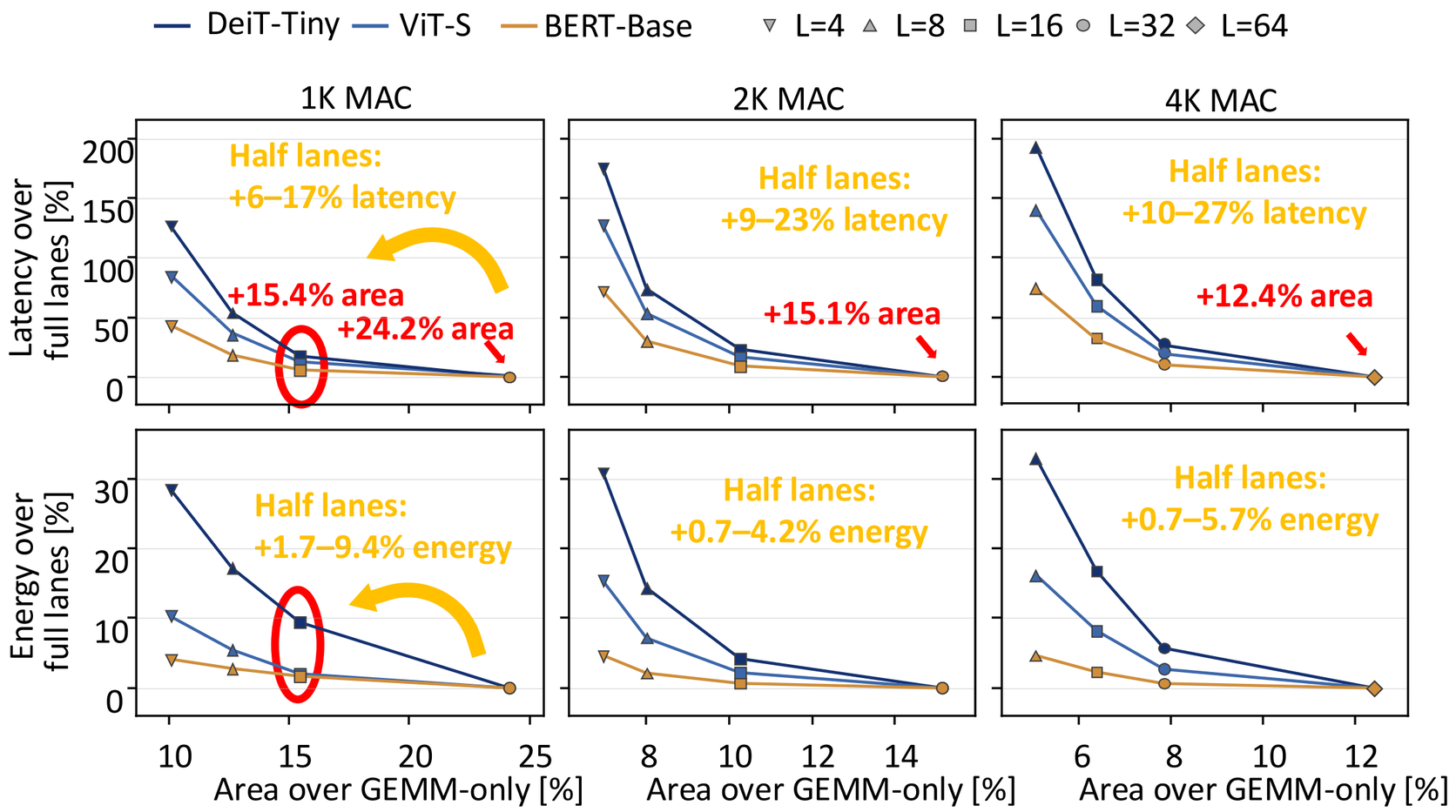
→ Every **workload stays within 0.7% of its FP32 checkpoint without fine-tuning**

Accelerator area increase with PwPA Lanes



→Let's **analyze the network latency** and **total energy** to decide the **suitable number of nonlinear units**

Selected design points → Take full PwPA lane configuration as baseline



Recommended design points

2K / 4K MAC

→ **Full lanes**

1K MAC

→ **Full lanes, or**

→ **Half lanes if tightly constrained by area**

Comparison with State-Of-The-Art

See the paper for elaboration on the comparison results.

Work	GEMM format	Nonlin. format	Tech nm	GEMM TOPS/mm²	Lane µm²	GEMM TOPS	Act. GOp/s	SM GOp/s	GEMM TOPS/W	Act. GOp/J	SM GOp/J
Cuyckens et al	MXINT8/FP8/6/4	✗	16	0.37	–	3.3	–	–	0.82	–	–
Flex-SFU	✗	INT/FP8–32	28	–	1797	–	1.2	–	–	706	–
PACE-lite	✗	FP32/16, BF16	12	–	1761	–	15.6	–	–	280	–
SoftEx	✗	BF16	12	–	2438	–	–	2.3	–	–	94
ITA	INT8	INT8	22	5.93	357	1.02	–	n.r.	16.9	–	n.r.
Prasad at al.	FP16	FP16	7	0.39	387	0.13	7.8	3.9	1.13	172	92
MINT-FX 1K, L=32	**MXINT8**	**BF16**	**12**	**9.9**	**1263**	**2.05**	**32**	**5.5**	**7.14**	**253**	**75**
MINT-FX 4K, L=64	**MXINT8**	**BF16**	**12**	**12.5**	**1132**	**8.19**	**64**	**11.5**	**7.01**	**159**	**50**

MINT-FX

One-stop accelerator supporting MXINT8 GEMM & flexible BF16 nonlinearities

WIDE CONFIGURATION RANGE

GEMM: 2.05-12.5 TOPS

Activations: 4-64 GOPS

STATE-OF-THE-ART EFFICIENCY

Area efficiency up to 12.5 TOPS/mm2

Energy efficiency up to 7.14 TOPS/W

Summary and future work

MINT-FX

Microscaling Integer Accelerator Template with Flexible Nonlinear Function Execution

Software recommendation

BF16 PwPA

Degree 1 and 8 segments

< 0.7%

accuracy loss vs. FP32

Hardware recommendation

2K / 4K MAC array
Full PwPA lanes at <16% extra area over GEMM-only

1K MAC array
Full PwPA lanes at <25% extra area over GEMM-only

1K MAC, if area is tightly constrained
Half PwPA lanes at <16% extra area, with <17% latency and <10% accelerator energy over the full lane configuration

Future work

Full deployment

Measure cluster-level energy and other bottlenecks

Other data formats

E.g. MXINT4; test accuracy, throughput and hardware cost

Separate nonlinear engine

Allows overlap with GEMM against extra data movement and area