ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

PACE-lite: Compact and Efficient Piecewise Polynomial Approximation for Transformer Nonlinearity Acceleration

Arpan Suravi Prasad*, Gamze İslamoğlu*, Luca Bertaccini*, Davide Rossi*, Francesco Conti*, Luca Benini*‡

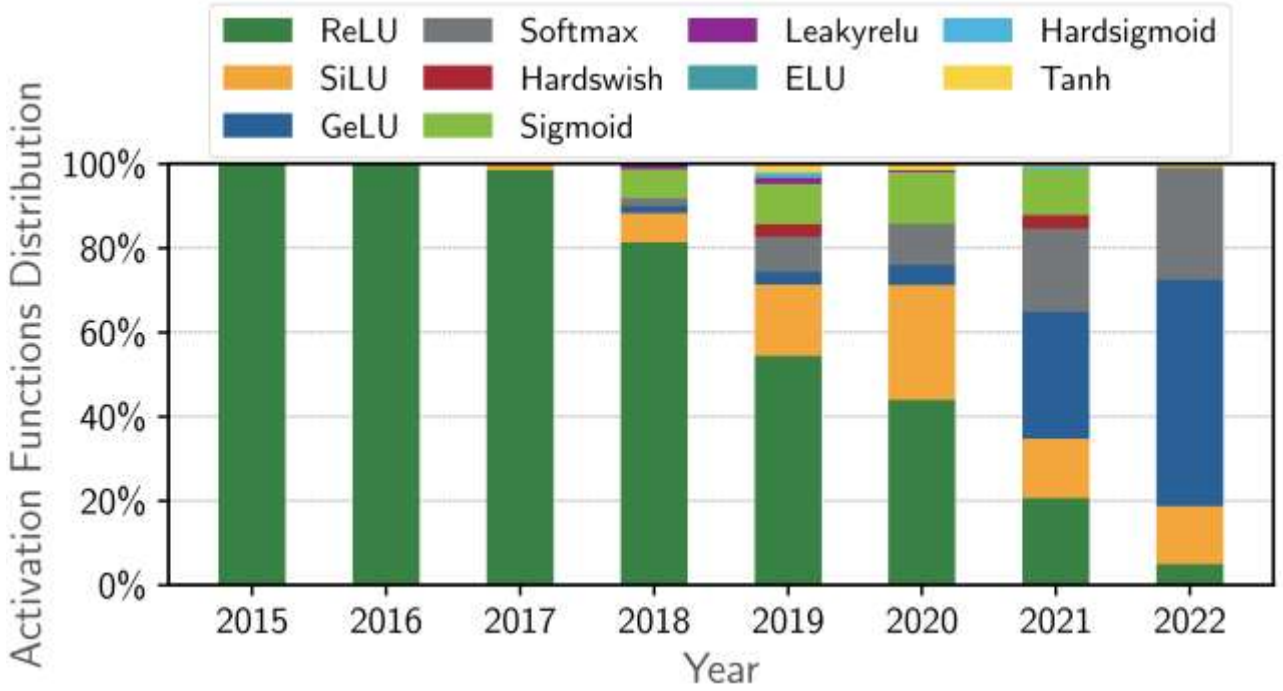
**Integrated Systems Laboratory, ETH Zurich;*
‡*DEI, University of Bologna*

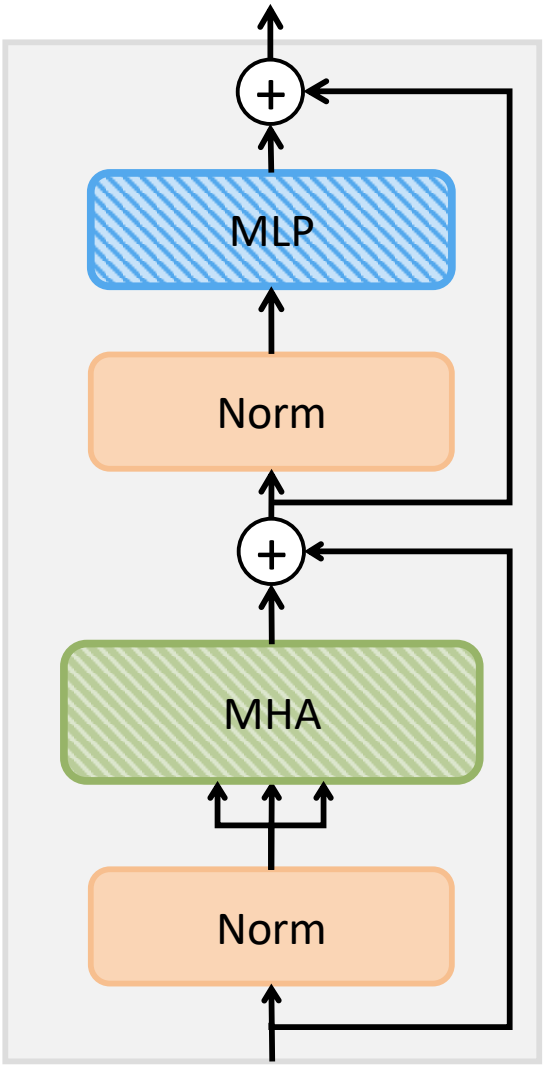
PULP Platform
Open Source Hardware, the way it should be!

pulp-platform.org
@pulp_platform
company/pulp-platform
youtube.com/pulp_platform

DNNs rely on complex nonlinear functions

- Linear operators dominate → aggressive acceleration
- Complex and diverse nonlinearities → need ***speed and flexibility***



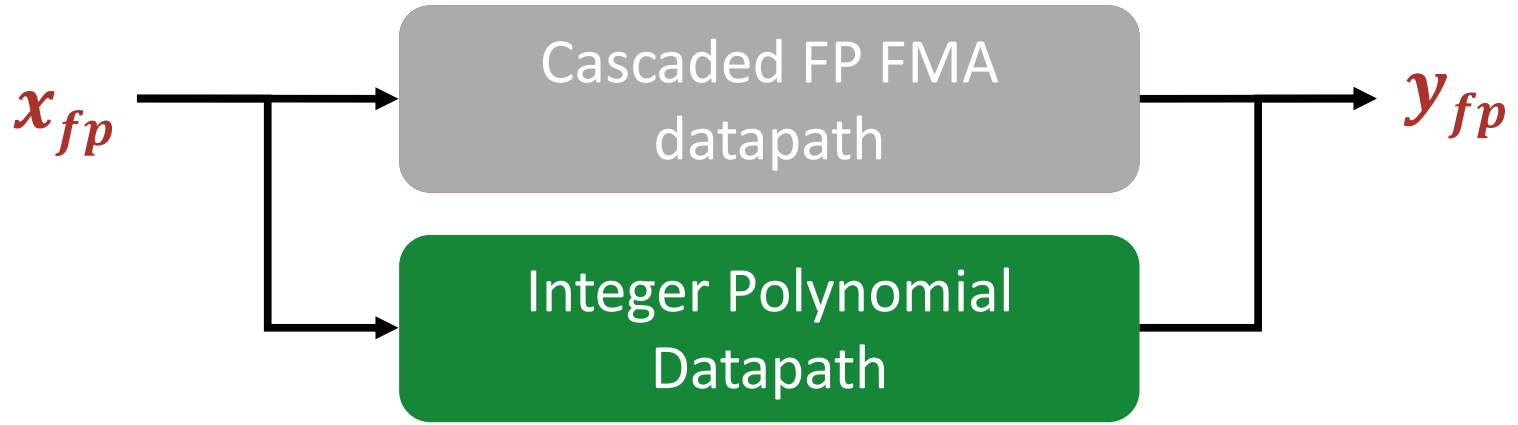


Andri et al. "Flex-SFU: Activation Function Acceleration with Non-Uniform Piecewise Approximation." *IEEE TCAD 2025*.

Contributions: Our Approach to Non-linearities

- PACE-lite – Compact **P**iecewise Polynomial **A**pproximation **C**ompute **E**ngine
- Integer Piecewise Polynomial Approximation (iPwPA)
  - Evaluated on various SoA DNNs (< 1% accuracy drop)
  - Drop-in approximation: no re-training/fine-tuning
  - Handles widely diverse nonlinearities
- Area-efficient parametric datapath
  - Support multi-precision FP inputs (FP32/FP16/BFP16) in runtime
  - Up to 81% area savings compared to pure FP32-based FMA datapath
- System–level evaluation
  - Integrated as a coprocessor to an octa-core RISC-V compute cluster
  - Low area overhead: 5.9% of the cluster
  - 44.1× throughput and 16.7× energy efficiency
- 3.1× Energy-efficient compared to SoA non-linear approximation hardware

iPwPA: Integer datapath for polynomial evaluation



convert input, output and coefficients to **integer**

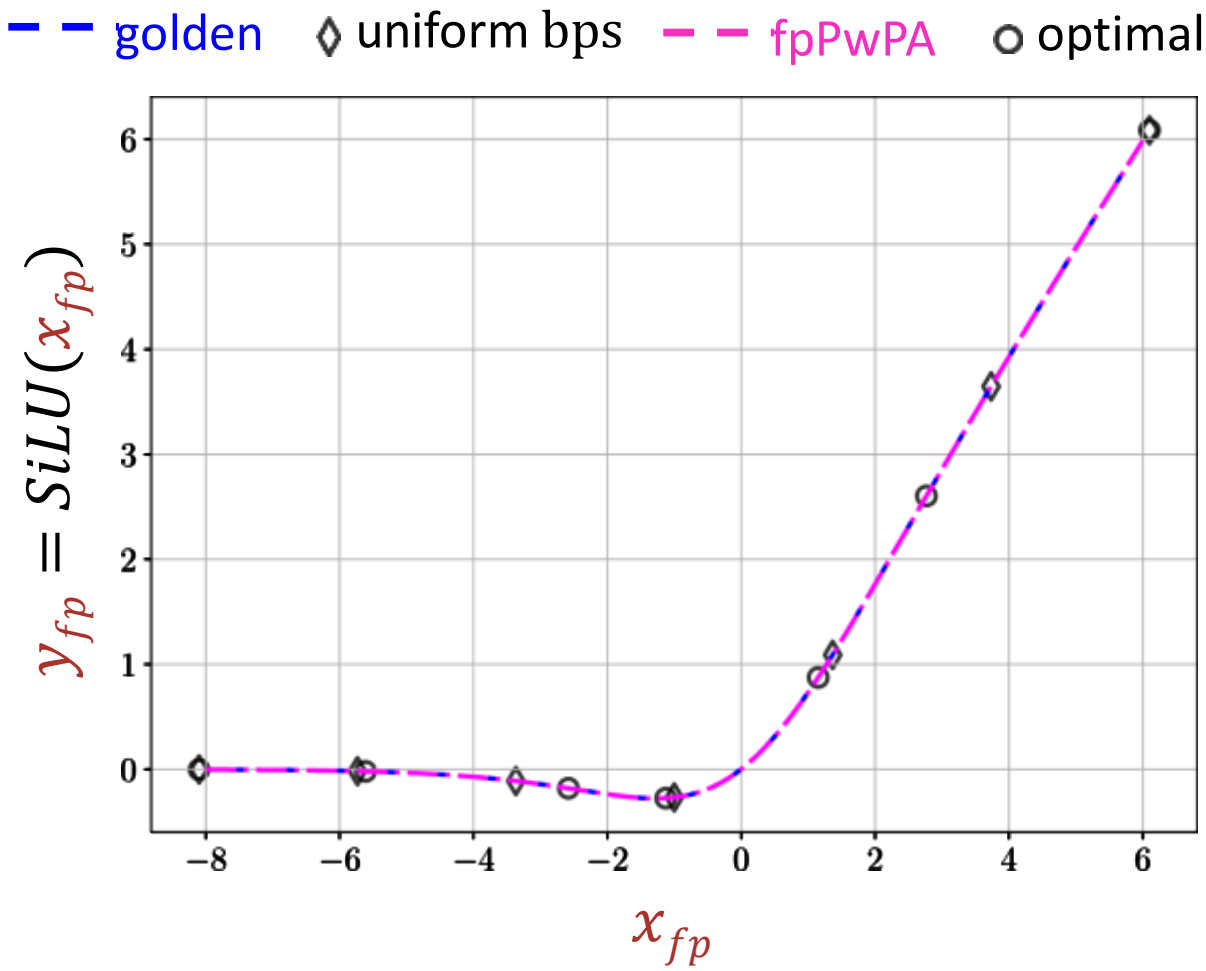
$$x_{int} = int(x_{fp} * 2^{scalex})$$
$$y_{int} = int(y_{fp} * 2^{scaley})$$

$$y_{int} = anx_{int}^{n} + a_{n\_1}x_{int}^{n-1} + .. + a_0$$
$$a_{nq} = int(a_n)$$

For a lower approximation error on integerizing coefficients $y_{int} \gg x_{int}$

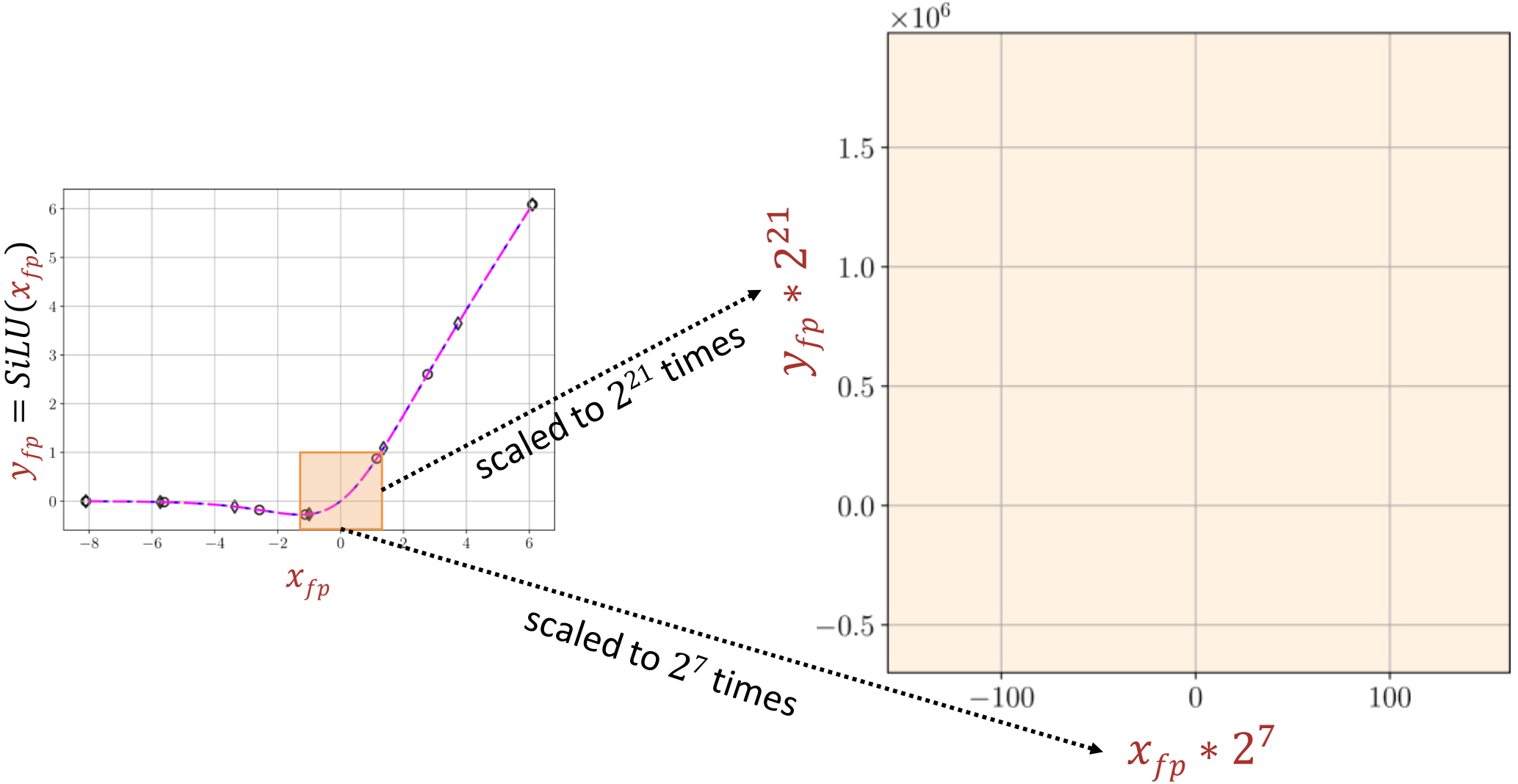
x_{fp} → scale up → x_{int} → Polynomial evaluation with integer coeffs a_{nq} → y_{int} → scale down → y_{fp}

FpPwPA: Piecewise Polynomial Approximation

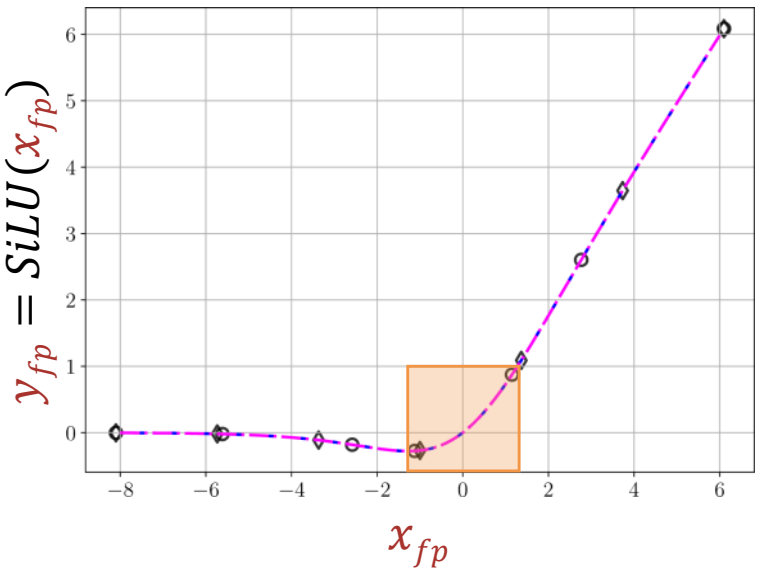


- Golden SiLU function
- Uniform bp initialization
- Optimal bp
  - Chebyshev fit & gradient descent

iPwPA: Single partition polynomial fit

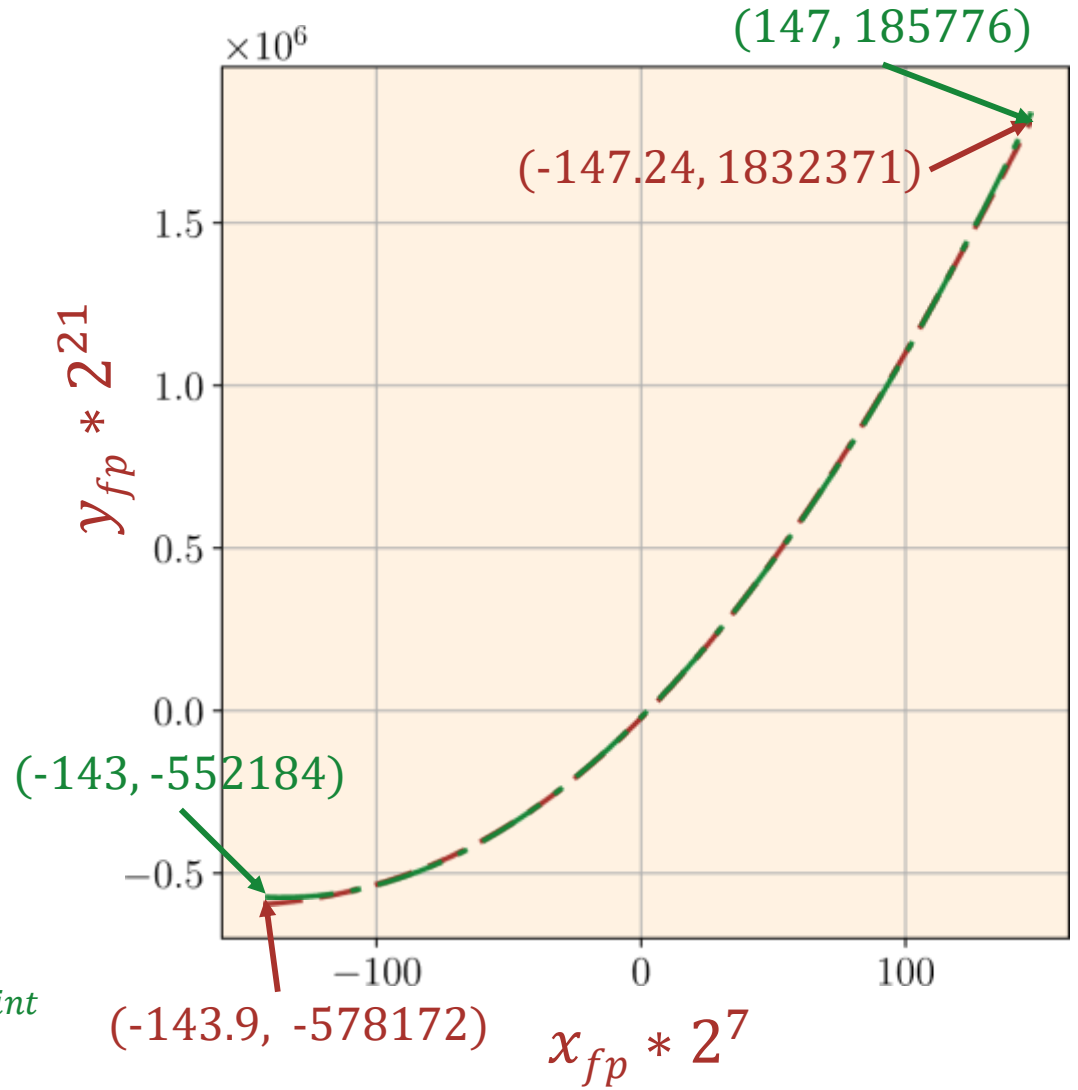


iPwPA: Single partition polynomial fit

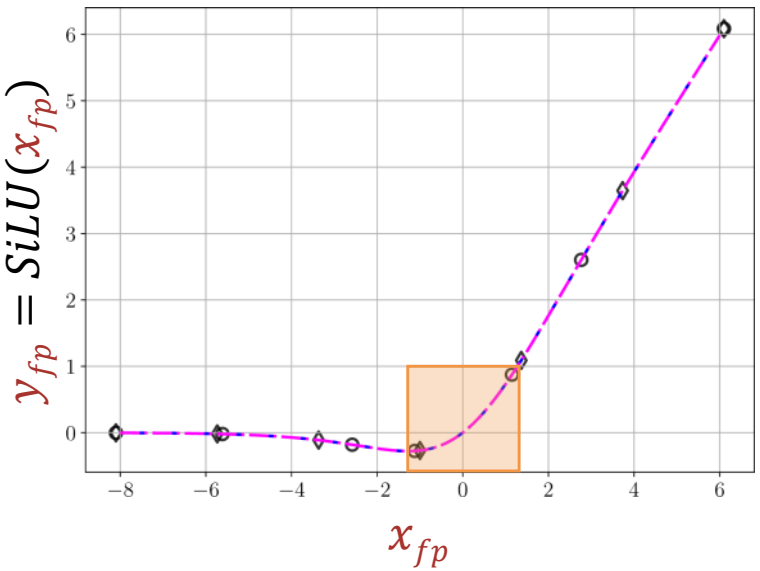


polynomial fit with $y_{fp} * 2^{21}$ and $x_{fp} * 2^{7}$

integer polynomial evaluation with x_{int} and y_{int}

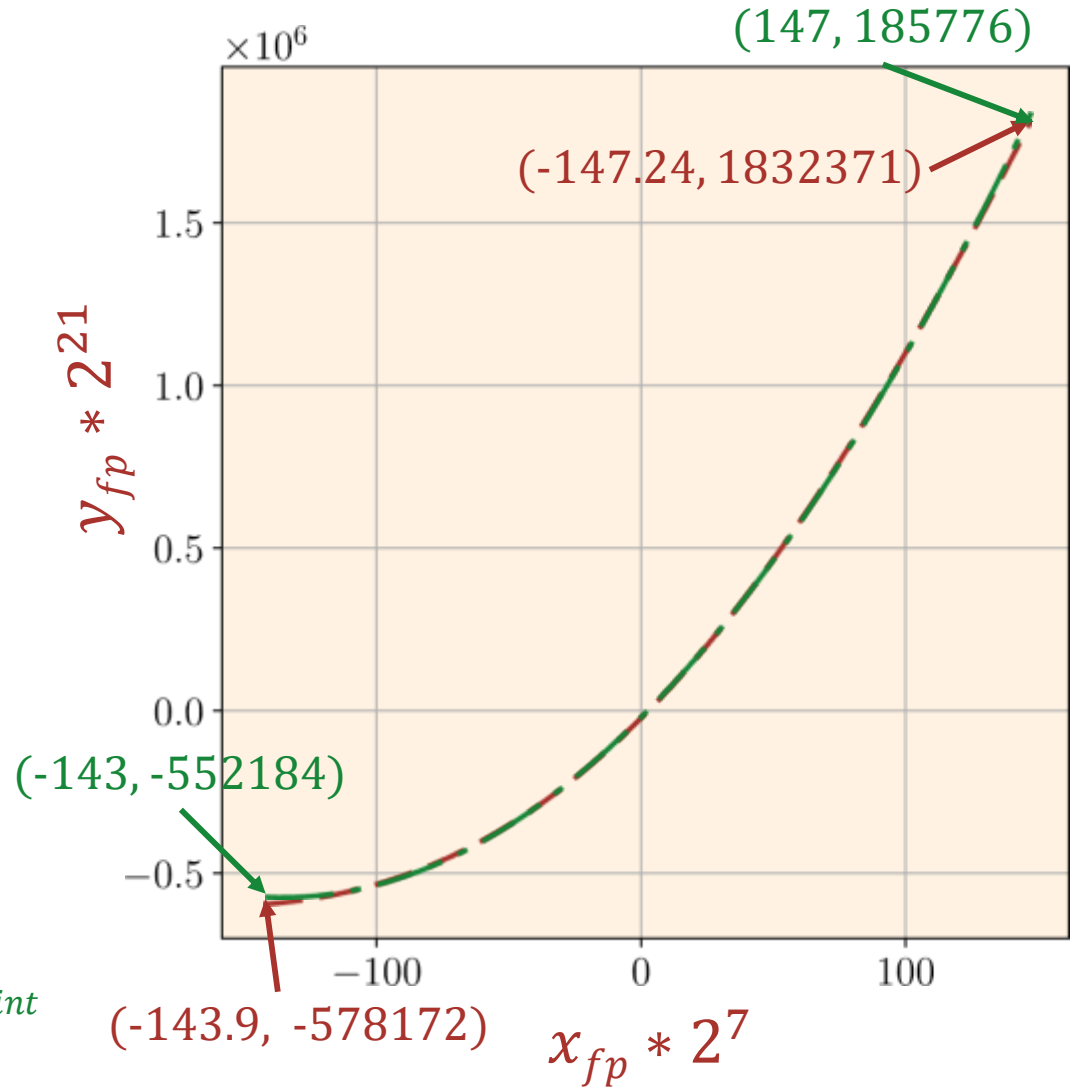


iPwPA: Single partition polynomial fit

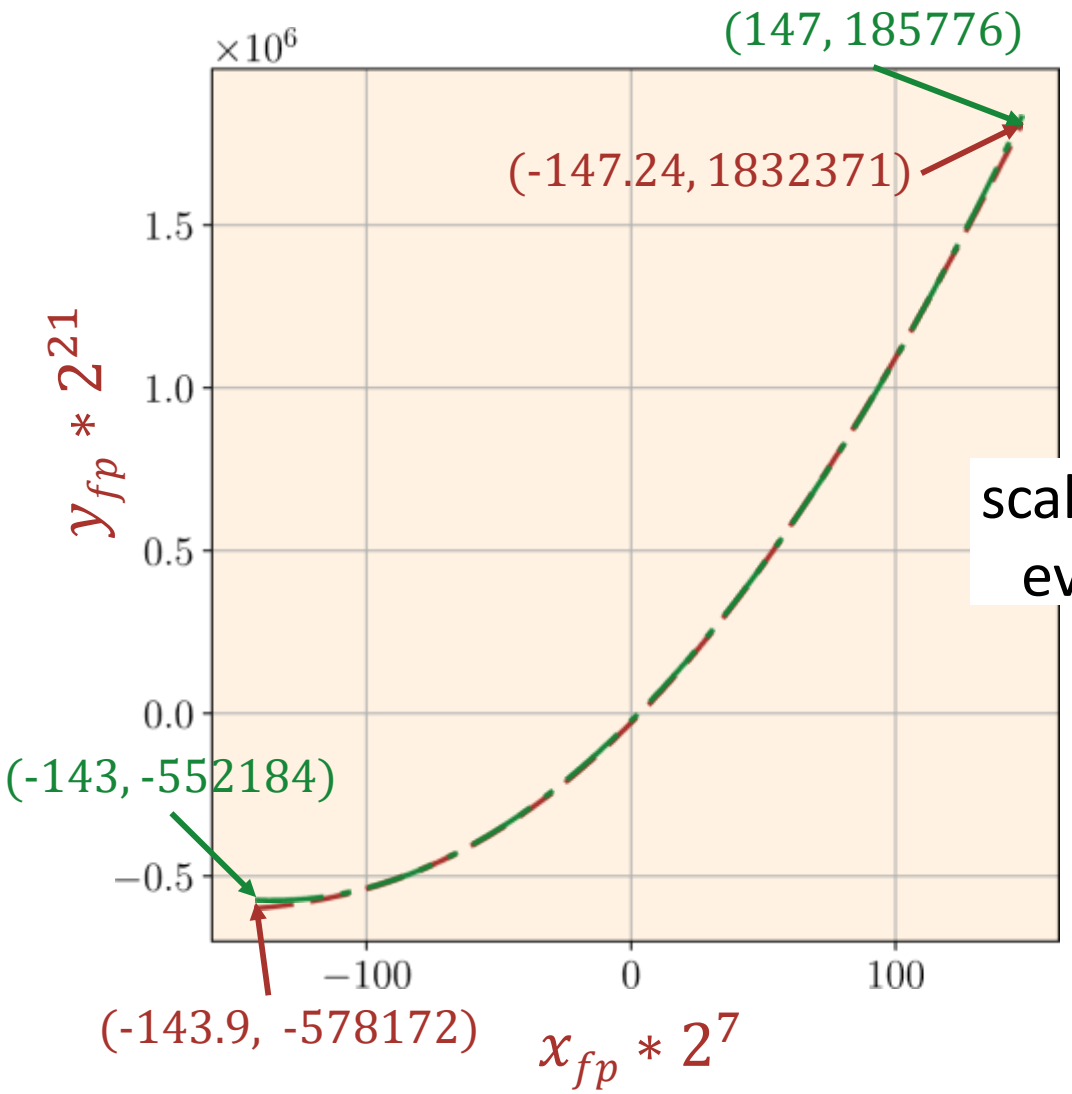


polynomial fit with $y_{fp} * 2^{21}$ and $x_{fp} * 2^{7}$

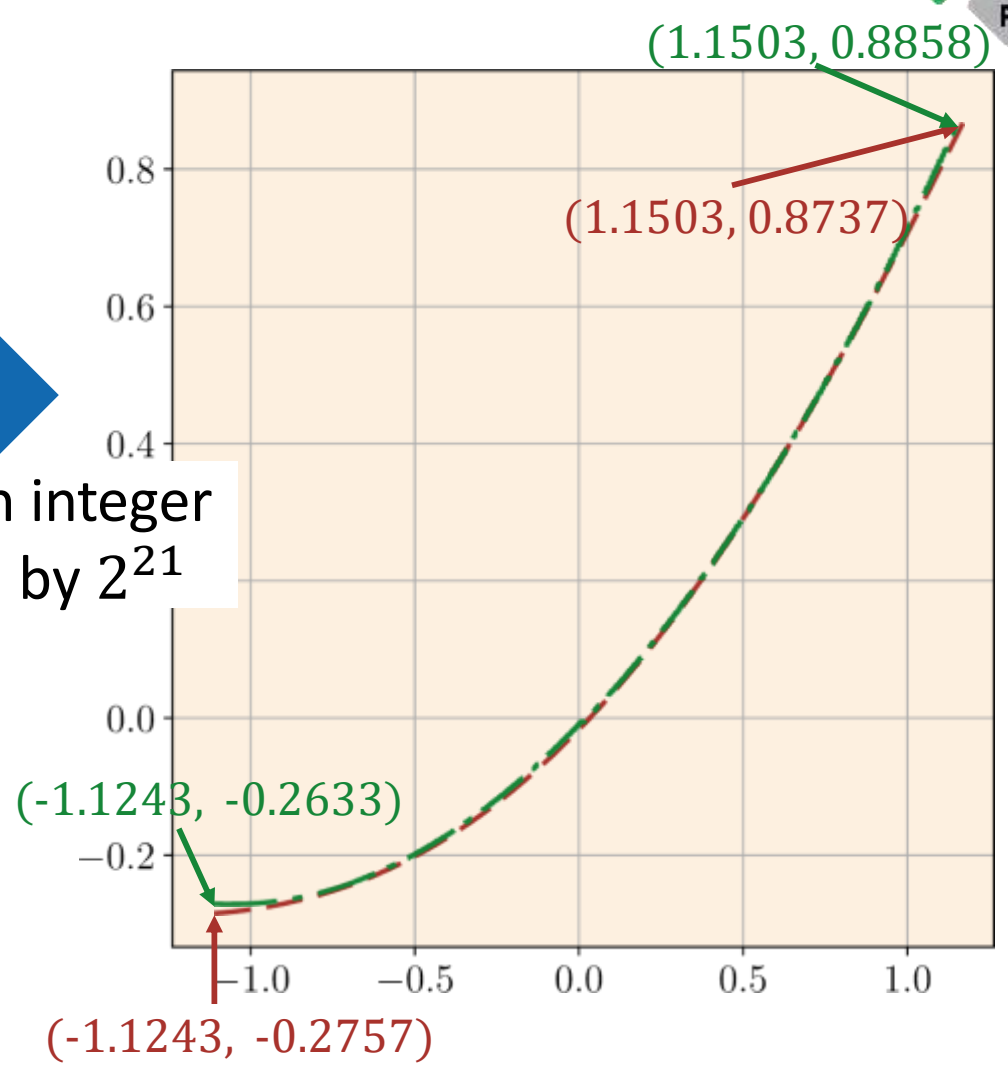
integer polynomial evaluation with x_{int} and y_{int}



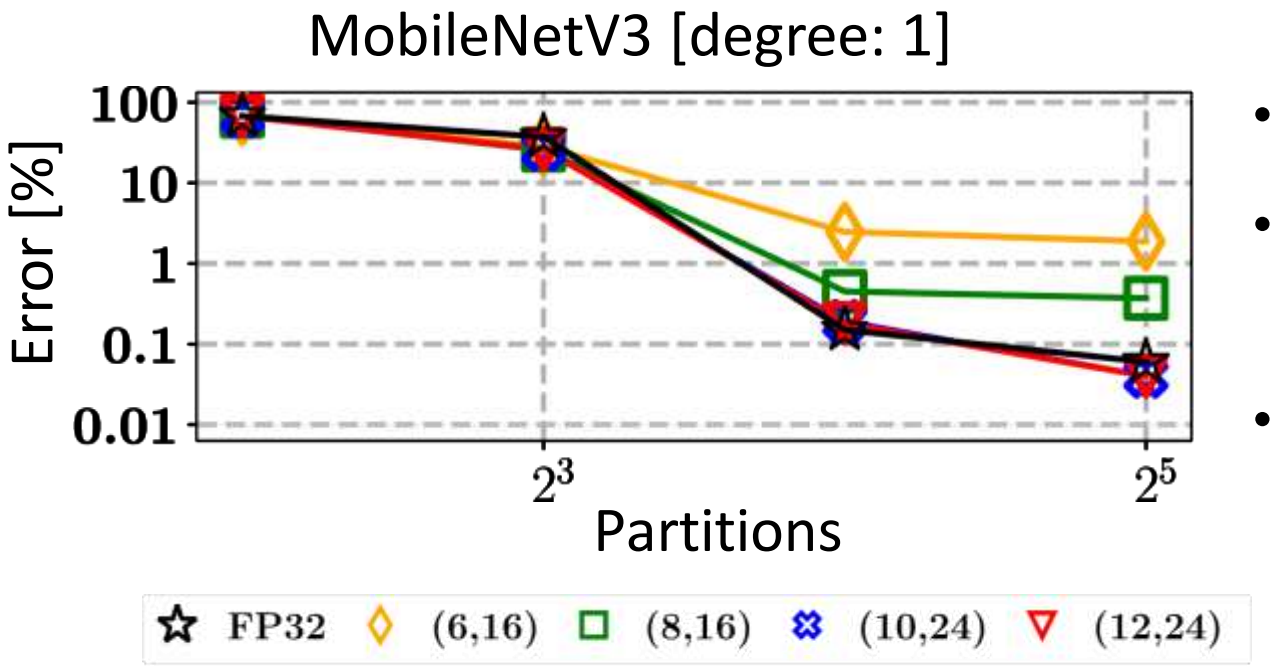
iPwPA: Single partition polynomial fit



scaled down integer evaluation by 2^{21}

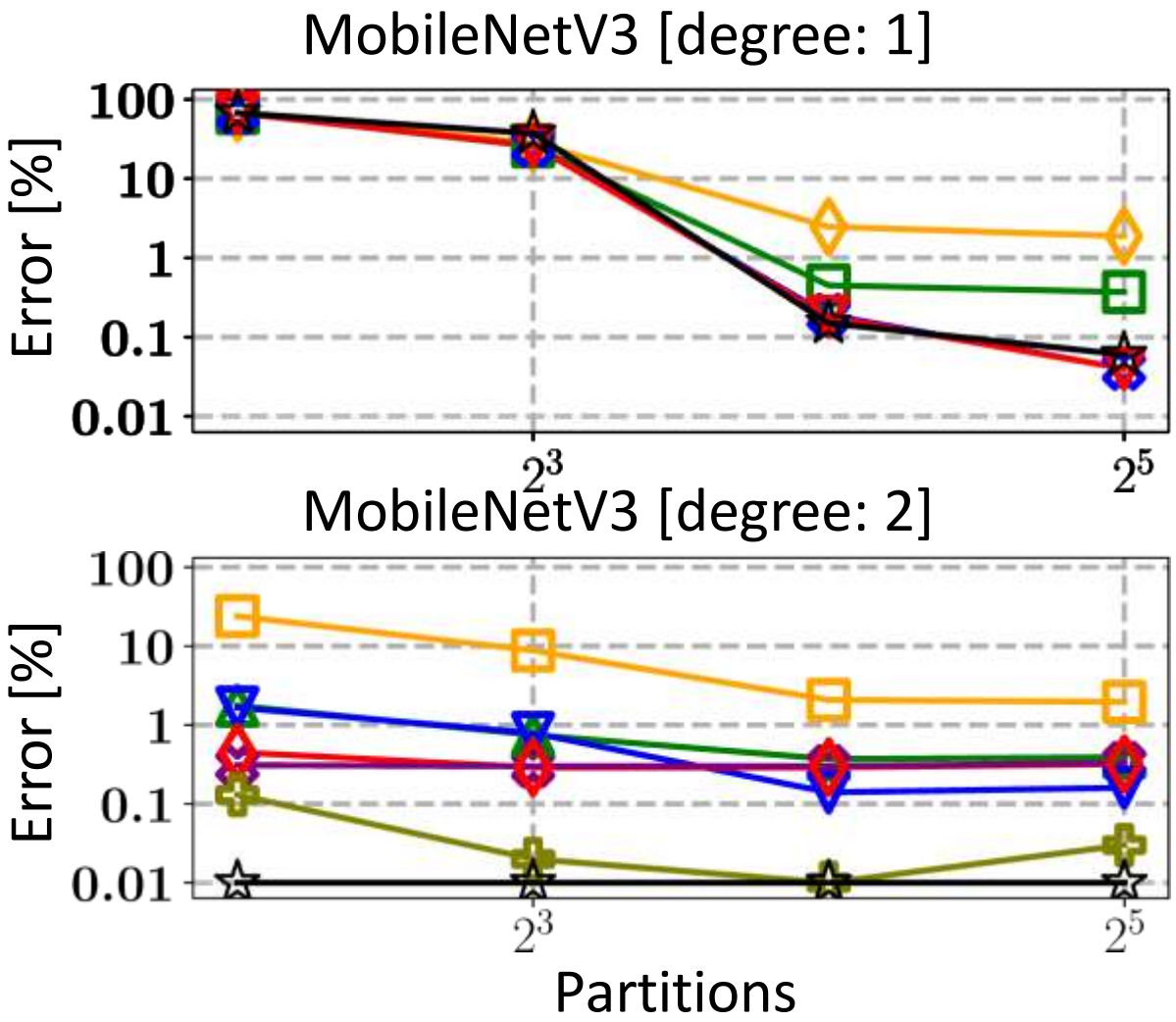


Results: Approximation evaluation on SoA DNN



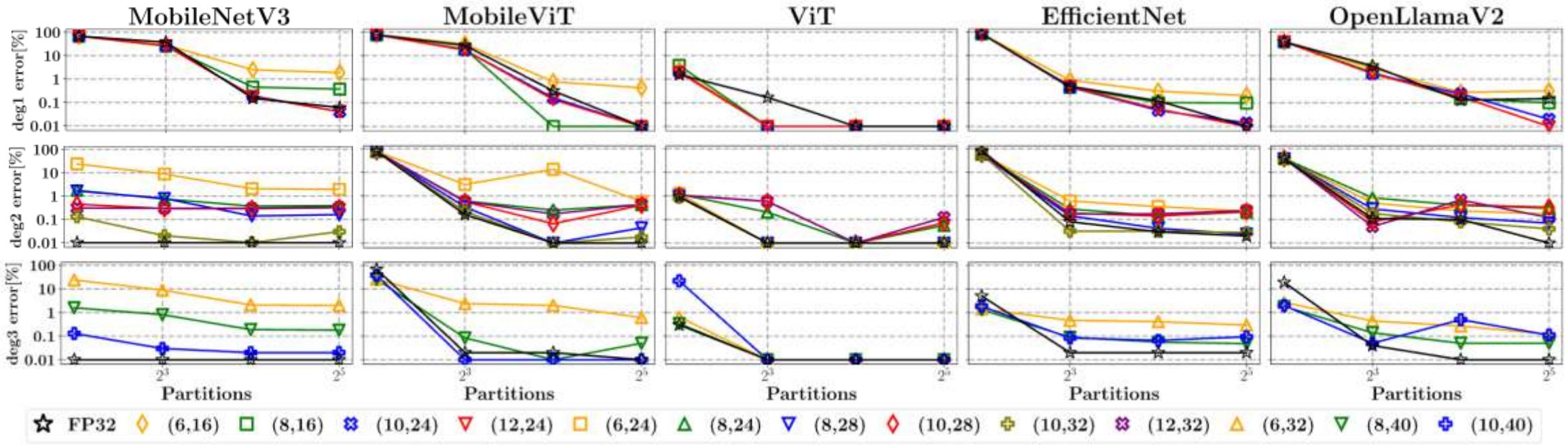
- Classification on ImageNet dataset
- FpPwPA as baseline implementation
  - Less than 1% for Partitions >= 16
- iPwPA $(B_x, B_y) = (6,16)$ > 1% error
  - B_x : Maximum bitwidth of x_{int}
  - B_y : Maximum bitwidth of y_{int}
- $(10,24)$ & $(12,24)$ achieve near FP32

Results: Approximation evaluation on SoA DNN



- Error↓ with degree ↑
- Except $(6,24)$ all have errors < 1%

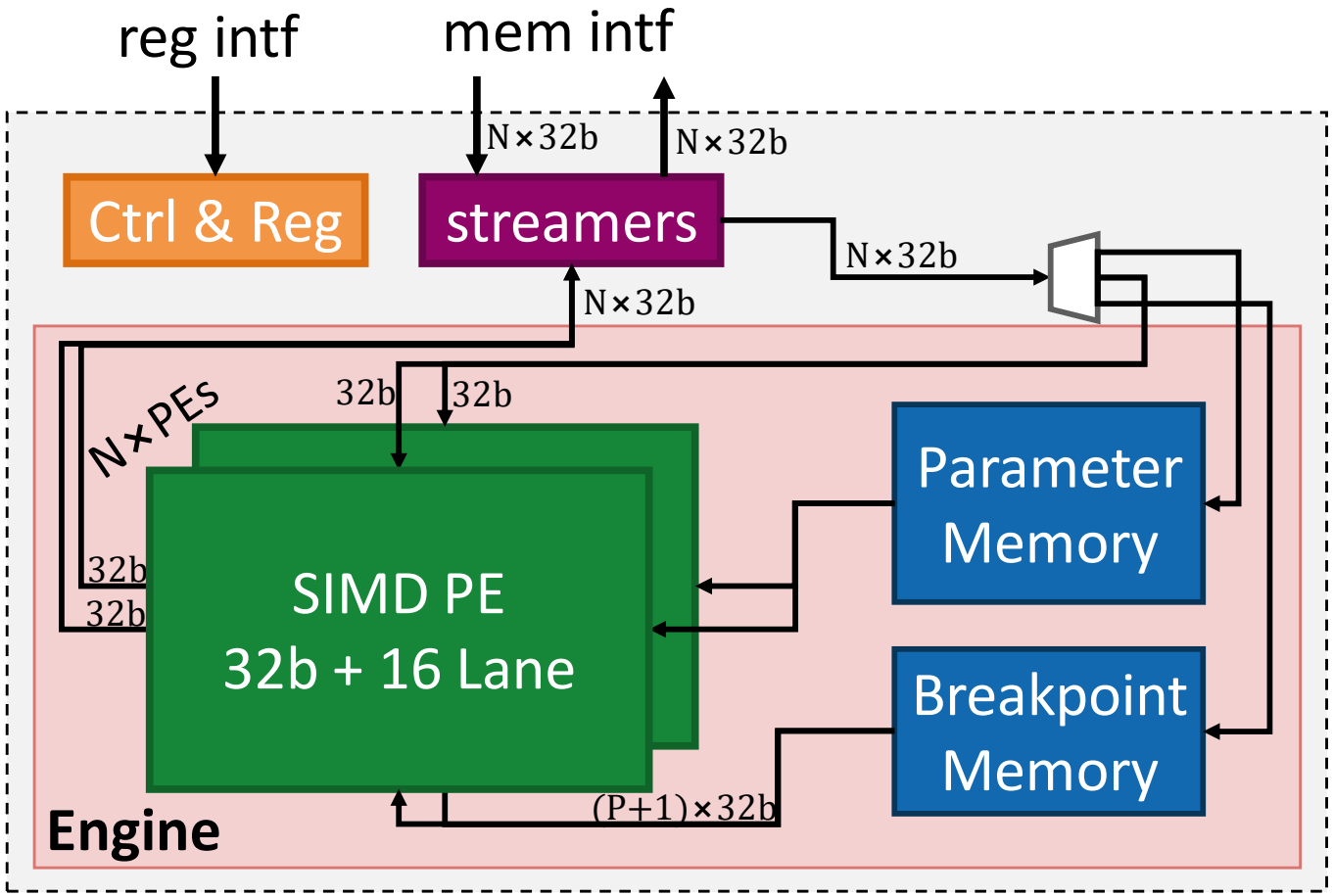
Results: iPwPA across different workload



- Model type CNNs (MobileNetV3, EfficientNetV2), Transformers (MobileViT, ViT), LLM (OpenLlamaV2)
- Nonlinearities: HardSwish, SiLU, GeLU, Exponent, Sigmoid
- On GPU RTX 3080, image models on ImageNet classification and LLM on LMEval harness, on tests PIQA, OpenBookQA, Winogrande, ARC Easy, and ARC Challenge
- All workloads reach < 1% error with iPwPA, near FP32 accuracy

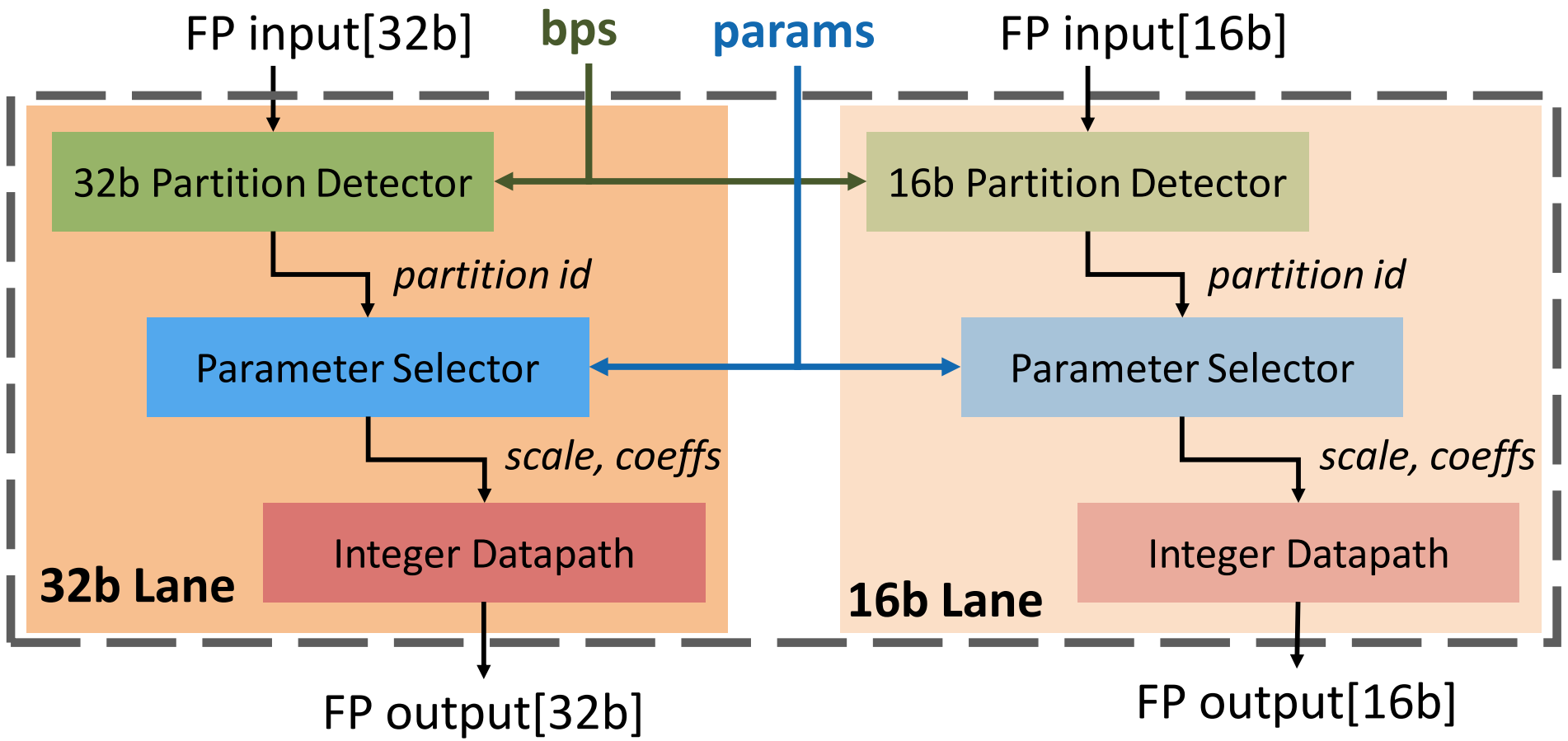
Architecture: standalone co-processor

- Parametric N×PEs in Engine
  - 16b lane: FP16/BFP16
  - 32b lane: FP32/FP16/BFP16
- Breakpoint memory – bps
- Parameter memory – scales, coeffs
- Parameters and breakpoints are broadcasted to all PEs
- Memory access through streamers
  - Initialize parameter & breakpoint memory
  - read input data & write output
- configuration through reg intf
  - Pointers of input, output, parameter etc.
  - Operand type, workload length

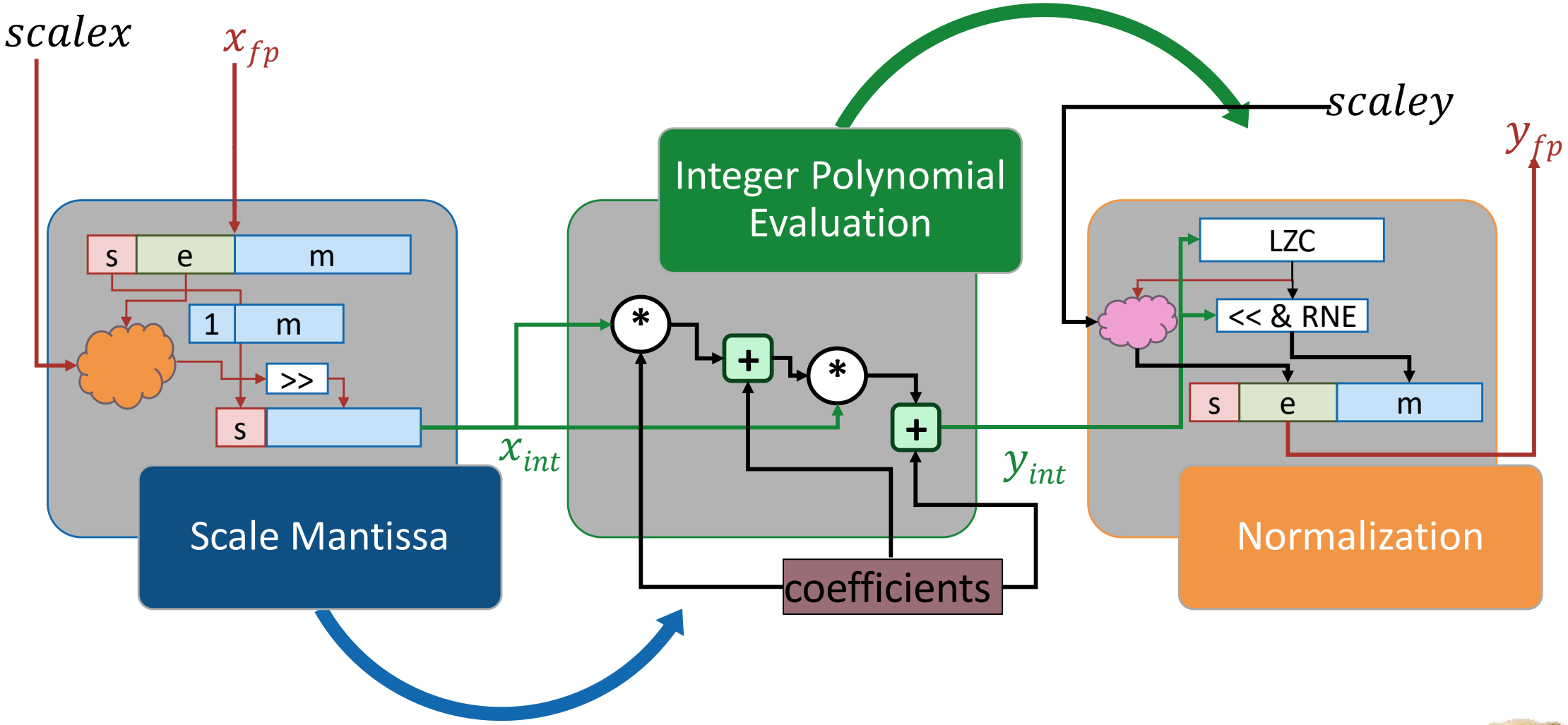


Architecture: PE structure

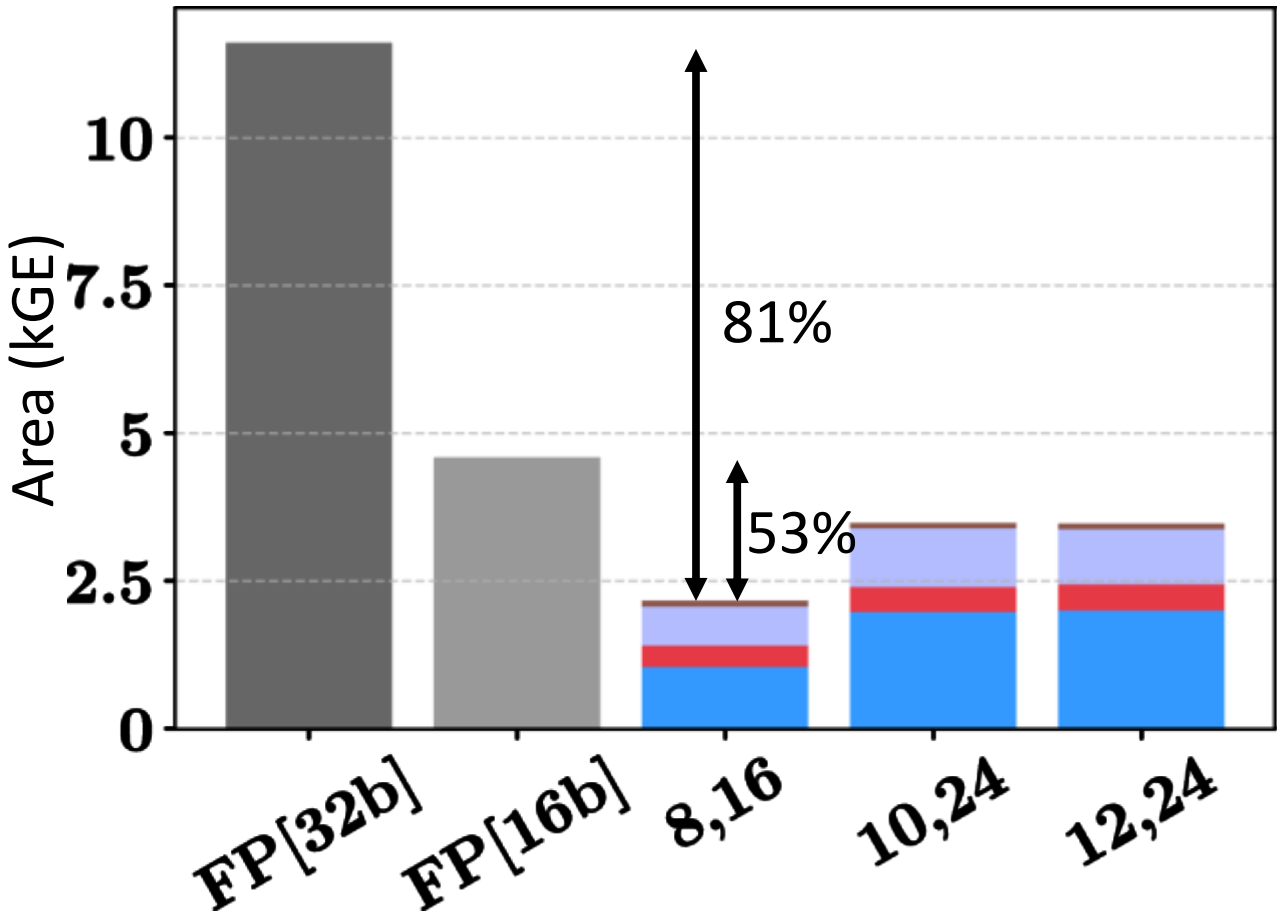
- 32b lane performs 1 × FP32/FP16/BFP16 and 16b performs 1 × FP16/BFP16



Architecture: Integer datapath(degree = 2)

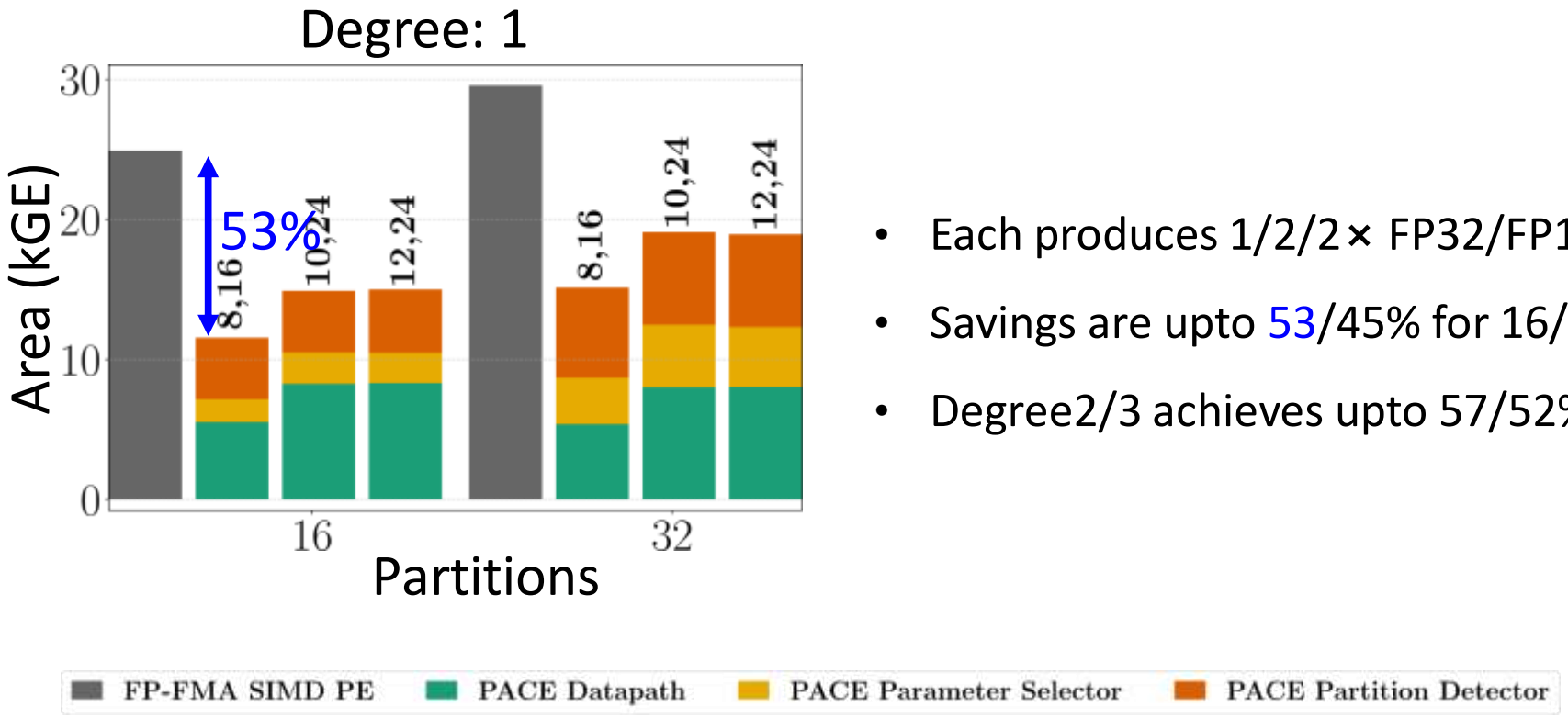


Results: Datapath area comparison (B_x, By)



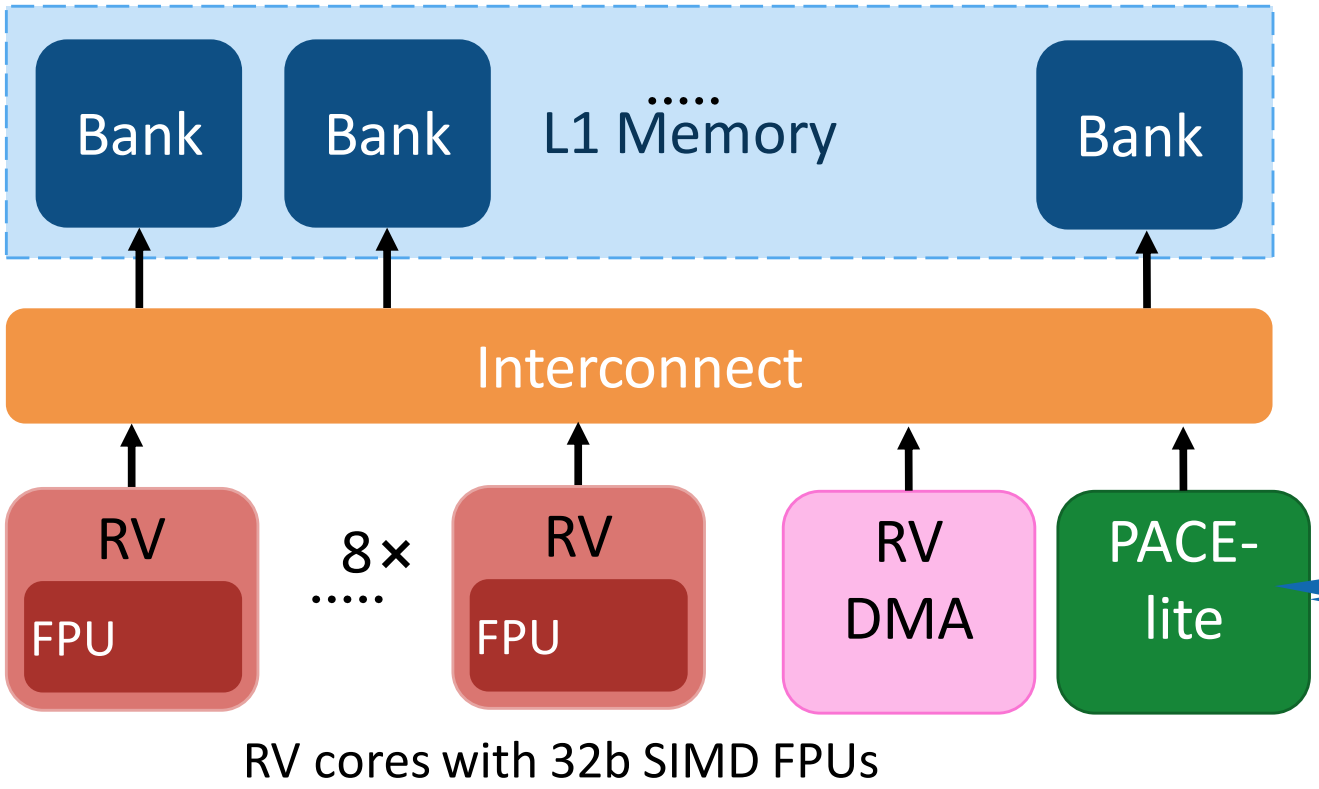
- Synthesis area GF12, 1GHz, degree–1
- Selected configurations < 1% error
- Datapath produces 1 PolyEval/cycle
- FMA based 32b/16b cascaded datapath
- Multiplier, Scale Mantissa, Normalization
- Area savings are
  - Upto 81% compared to FP[32b]
  - Upto 53% compared to FP[16b]
- Savings are
  - Upto 55/80% for FP[32b]/FP[16b] for degee2 and 32/70% for degree3

Results: PE area comparison



- Each produces 1/2/2× FP32/FP16/BFP16 PolyEval/cycle
- Savings are upto 53/45% for 16/32 partitions
- Degree2/3 achieves upto 57/52% savings

Results: System Level Evaluation



5.9% area of PACE-lite enhanced cluster

44.1× higher throughput & 16.7× energy efficiency compared to floating point PwPA SW emulation with FPUs

8× PEs <2, 16, 10, 32>
<degree, parts, Bx, By>

PACE-Lite standalone peak throughput: 7.9/15.6 GPolyEval/s for FP32/16
PACE-Lite standalone peak energy efficiency: 124/200/231 GPolyEval/s/W for FP32/16/BFP16

Comparative Analysis: Standalone PACE-lite PE

Work	Diverse	Precision	Tech [nm]	Frequency [GHz]	Area [µm²]	Throughput [GPolyEval/s]	Efficiency [GPolyEval/s/W]
[1]Diaz-Conti et. al	✓	Fixed	NA	NA	NA	NA	NA
[2]Reggiani et. al	✓	I/FP32/16/8	28	0.6	9791	0.6/1.2/2.4	214
This Work	✓	FP32/16	12	1	2884	1/2/2	234/307/322

- [2] doesn't include the power or area of the MAC engine in its calculation
- Considering only the Partition detector and Parameter Selector, PACE-lite offers
  - 1.6× more throughput
  - 6.5× energy efficiency

Comparative Analysis: Standalone PACE-lite accelerator

Work	Diverse	Precision	Tech [nm]	Frequency [GHz]	Area [µm^2]	Throughput [GPolyEval/s]	Efficiency [GPolyEval/s/W]
[3]Belano et. al	✗	BFP16	12	1	39000	4.5	84.4
[4]Islamoglu et. al	✗	INT8	22	0.5	5709	NA	NA
[5]Zhu et.al	✗	NA	28	2.78	10081	22.24	NA
This Work	✓	FP32/FP16/ BFP16	12	0.95	10538	7.9/15.6/ 15.6	124/200/ 231

- Only PACE-lite supports diverse non-linearity
- [3] implements softmax and uses exponent with RISC-V cores to perform SiLU and GeLU
- PACE-lite reaches the SoA in diverse non-linear computation
  - 3.5× higher throughput compared to [3]
  - 3.1× higher energy efficiency compared to [3]

Conclusion: Summarising PACE-lite

PACE-lite is a **compact**, efficient accelerator to execute **diverse** nonlinearity

Integer approximation achieves **less than 1%** error at various bitwidth configurations

On the datapath level, **81%** less area compared to the FP32-based FMA

At the system level, **44×** boost in **throughput** and **16×** boost in **efficiency** with 5.9% area

3.1× Energy efficient compared to SoA

SoA References

1. González-Díaz_Conti, Griselda, et al. "Hardware-based activation function-core for neural network implementations." *Electronics* 11.1 (2021): 14.
2. Reggiani, Enrico, Renzo Andri, and Lukas Cavigelli. "Flex-sfu: Accelerating dnn activation functions by non-uniform piecewise approximation." *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2023.
3. Belano, Andrea, et al. "A Flexible Template for Edge Generative AI with High-Accuracy Accelerated Softmax & GELU." *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* (2025).
4. Islamoglu, Gamze, et al. "Ita: An energy-efficient attention and softmax accelerator for quantized transformers." *2023 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2023.
5. Zhu, Danyang, et al. "Efficient precision-adjustable architecture for softmax function in deep learning." *IEEE Transactions on Circuits and Systems II: Express Briefs* 67.12 (2020): 3382-3386.