2023 Conference on Next Generation Arithmetic (CoNGA)

PLAUs: Posit Logarithmic Approximate Units to implement low-cost operations with real numbers

Raul Murillo, David Mallasén,
Alberto A. Del Barrio and Guillermo Botella

Complutense University of Madrid, Spain

UNIVERSIDAD COMPLUTENSE MADRID
ArTeCS
Fundación BBVA
Comunidad de Madrid
UNIÓN EUROPEA
Fondos Estructurales
Invertimos en su futuro
UNIÓN EUROPEA
Fondo Social Europeo
El Fondo Social Europeo invierte en tu futuro

Outline

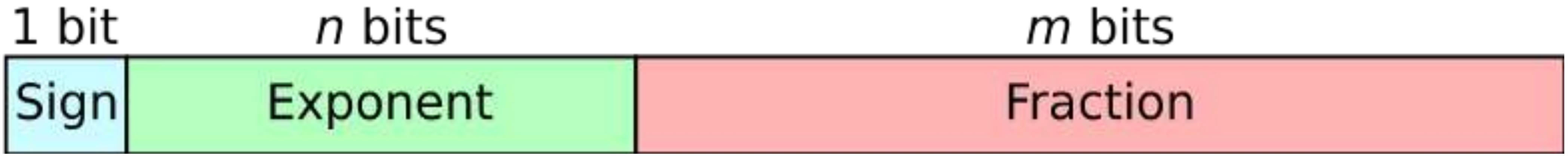
- **Background**
- Posit Logarithmic Approximate Units (PLAUs)
- Implementation analysis
- Applications
- Conclusions

Background

Scientific applications compute real numbers ($\log_2(x)$, ⅓)

How to represent real numbers in computers?

- Floating-point (IEEE 754™)
  - Used in most modern computers

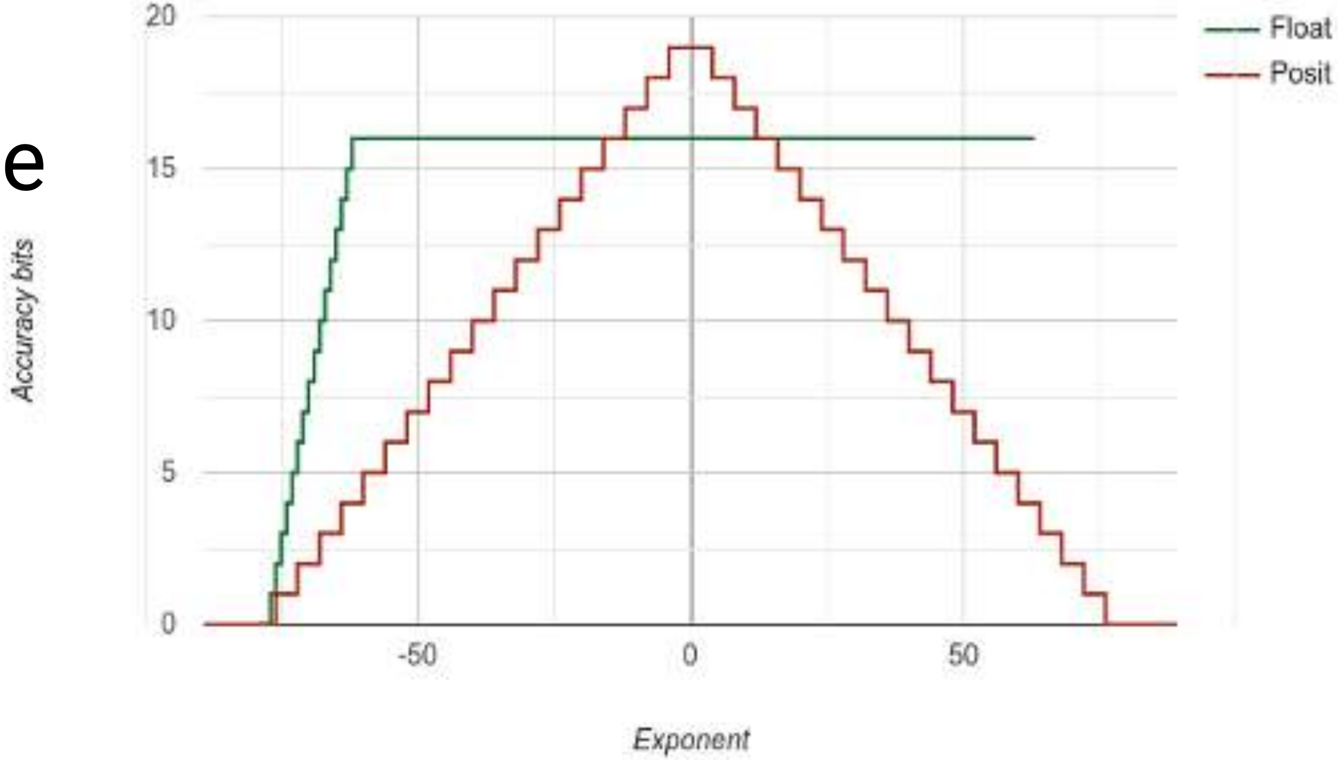


- Emerging alternatives: bfloat16, TensorFloat, posits…

Posit™ Arithmetic

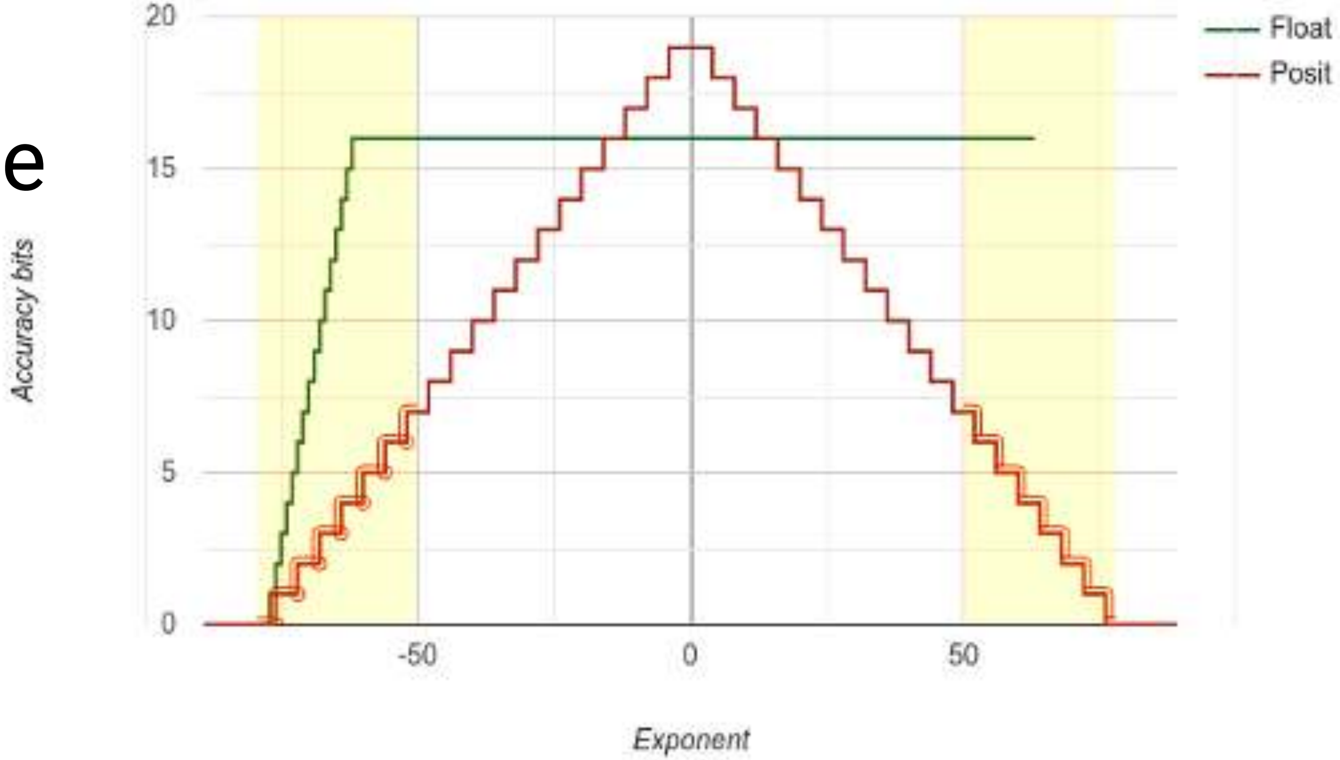
1 bit	k+1 bits	2 bits	m bits
Sign	Regime	Exponent	Fraction

- New variable-length field: *Regime*
- Trade-off accuracy – dynamic range
- Only two special cases Zero and ±∞ (NaR)
- Single rounding mode
- Easy comparison



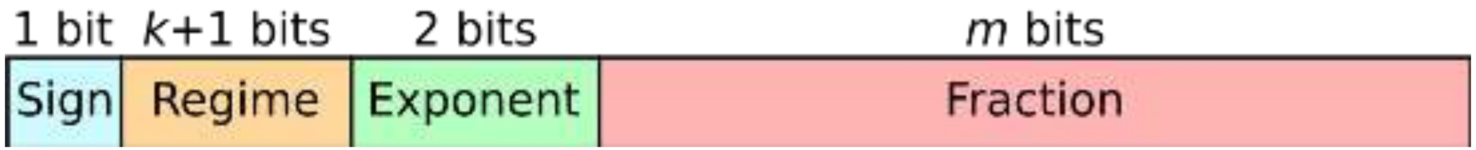
Posit™ Arithmetic

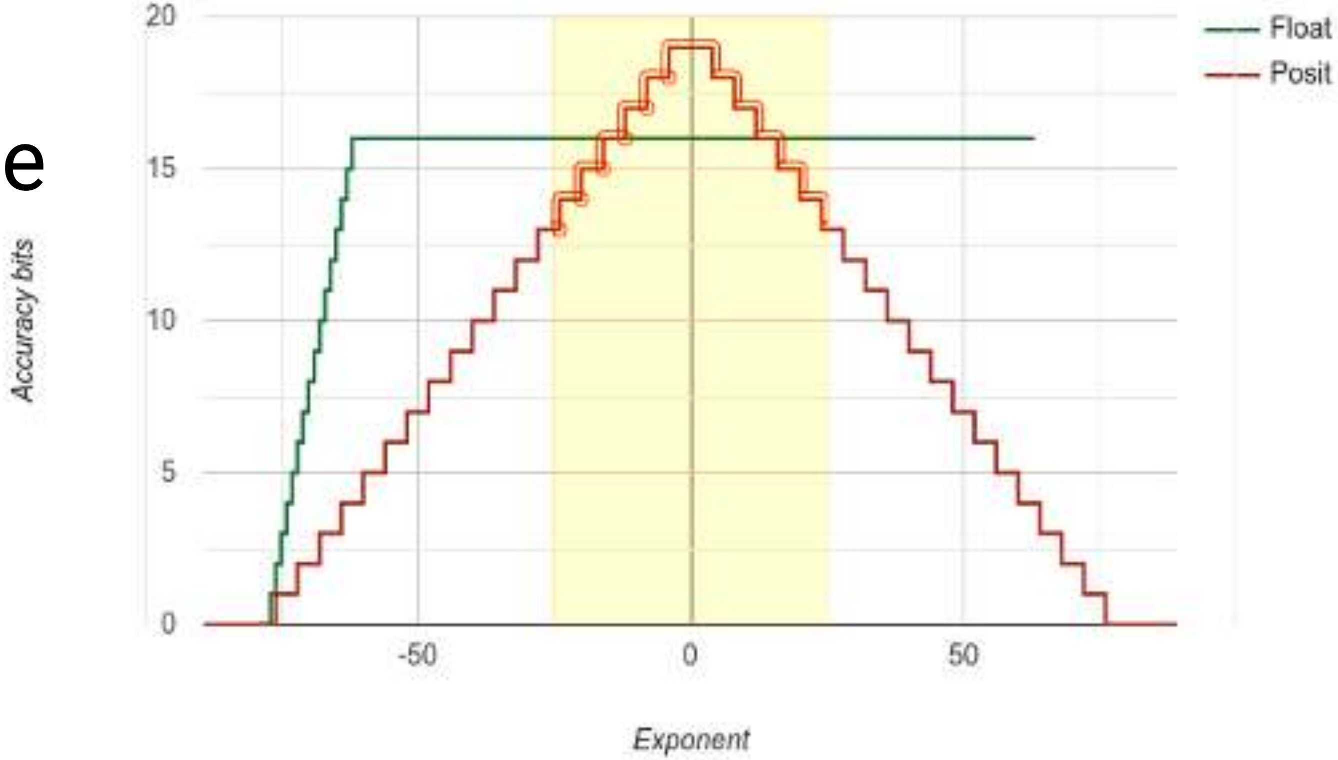
- New variable-length field: *Regime*
- Trade-off accuracy – dynamic range
- Only two special cases Zero and ±∞ (NaR)
- Single rounding mode
- Easy comparison



Posit™ Arithmetic

- New variable-length field: *Regime*
- Trade-off accuracy – dynamic range
- Only two special cases Zero and ±∞ (NaR)
- Single rounding mode
- Easy comparison



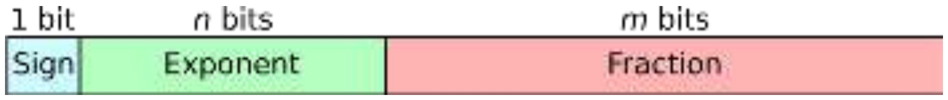


Drawbacks?

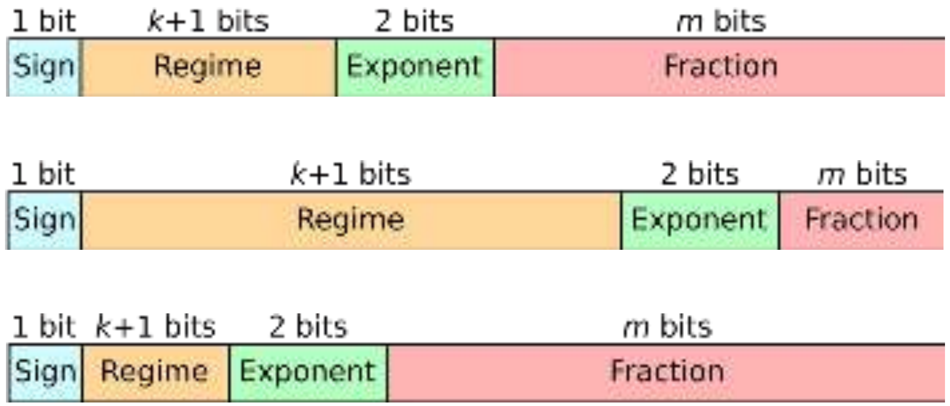
- Not implemented in computers
- High Area/Latency overhead

High implementation cost

Float



Posit



?

?

?

Outline

- Background
- **Posit Logarithmic Approximate Units (PLAUs)**
- Implementation analysis
- Applications
- Conclusions

Posit Logarithmic Approximate Units (PLAUs)

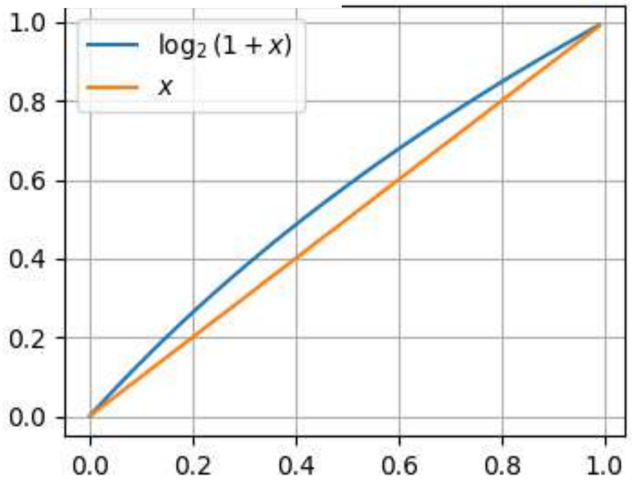
- Use logarithm properties and approximation

$$\log_b(xy) = \log_b(x) + \log_b(y)$$

$$\log_b\left(\frac{x}{y}\right) = \log_b(x) - \log_b(y)$$

$$\log_b(\sqrt[y]{x}) = \frac{\log_b(x)}{y}$$

$\log(1+x) \approx x$, for small x



[1] John N. Mitchell, "Computer multiplication and division using binary logarithms." *IRE Transactions on Electronic Computers* 4 (1962): 512-517.
[2] Raul Murillo, et al. "PLAM: A posit logarithm-approximate multiplier." *IEEE Transactions on Emerging Topics in Computing* 10.4 (2021): 2079-2085.

Posit Logarithmic Approximation

- Standard Posit value:

$$X = (-1)^s \times 2^{4r} \times 2^e \times (1+f)$$

$$f \in [0, 1)$$

- Logarithmic approximation:

$$\begin{aligned} \log_2 X &= 4 \times r + e + \log_2(1+f) \\ &\approx 4 \times r + e + f \end{aligned}$$

PLAUs – Multiplication

$k = 4\,reg + exp$

$$P_{exact} = (-1)^{s_A + s_B} \times 2^{k_A + k_B} \times (1 + f_A) \times (1 + f_B)$$

$$\log_2 X = \ \approx 4 \times r + e + f$$

$$\log_b(xy) = \log_b(x) + \log_b(y)$$

$$P_{approx} = \begin{cases} (-1)^{s_A + s_B} \times 2^{k_A + k_B} \times (1 + f_A + f_B) & \text{if } f_A + f_B < 1, \\ (-1)^{s_A + s_B} \times 2^{k_A + k_B + 1} \times (f_A + f_B) & \text{if } f_A + f_B \geq 1. \end{cases}$$

PLAUs – Division

$$k = 4\,reg + exp$$

$$Q_{exact} = (-1)^{s_A - s_B} \times 2^{k_A - k_B} \times (1 + f_A)/(1 + f_B)$$

$$\boxed{\log_2 X = \approx 4 \times r + e + f}$$

$$\boxed{\log_b\left(\frac{x}{y}\right) = \log_b(x) - \log_b(y)}$$

$$Q_{approx} = \begin{cases} (-1)^{s_A - s_B} \times 2^{k_A - k_B} \times (1 + f_A - f_B) & \text{if } f_A - f_B \geq 0, \\ (-1)^{s_A - s_B} \times 2^{k_A - k_B - 1} \times (2 + f_A - f_B) & \text{if } f_A - f_B < 0. \end{cases}$$

PLAUs – Square Root

$$k = 4\,reg + exp$$

$$R_{exact} = \sqrt{2^k \times (1+f)} = 2^{k/2} \times \sqrt{1+f}$$

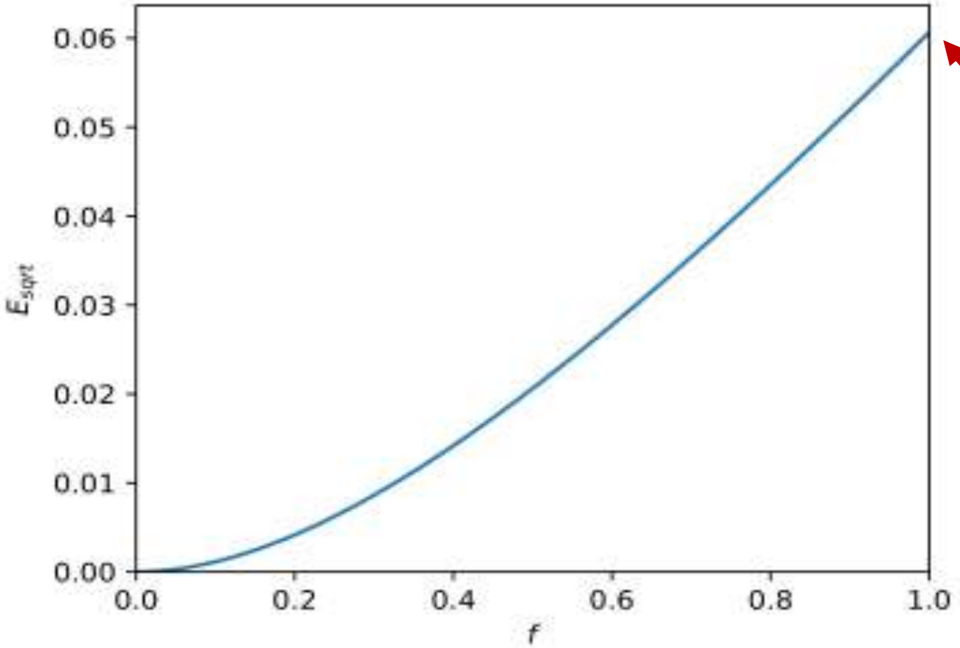
$$\log_2 X = \approx 4 \times r + e + f$$

$$\log_b(\sqrt[y]{x}) = \frac{\log_b(x)}{y}$$

$$R_{approx} = 2^{k/2} \times (1 + f/2)$$

Error for PLAUs

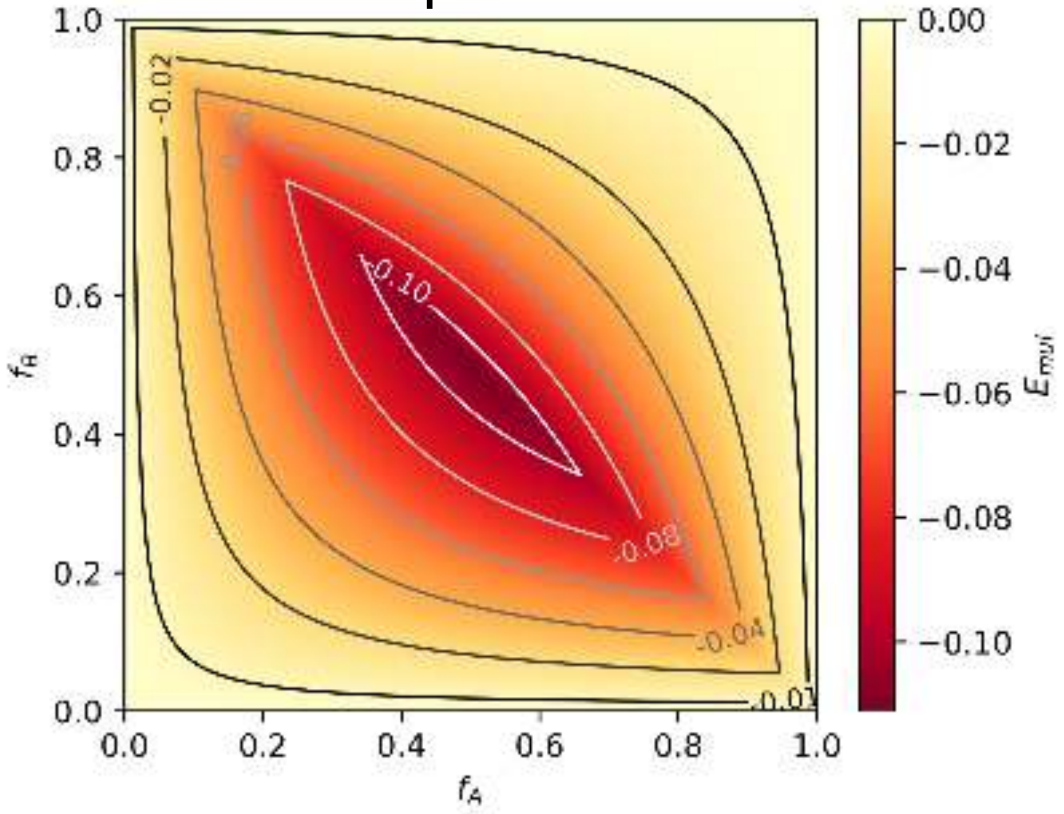
$$\left.\begin{aligned} R_{exact} &= 2^{k/2} \times \sqrt{1+f} \\ R_{approx} &= 2^{k/2} \times (1+f/2) \end{aligned}\right\} \Rightarrow \quad E_{sqrt} = \frac{1+f/2}{\sqrt{1+f}} - 1$$



Max is $3/\sqrt{8} - 1$, or 6.06%

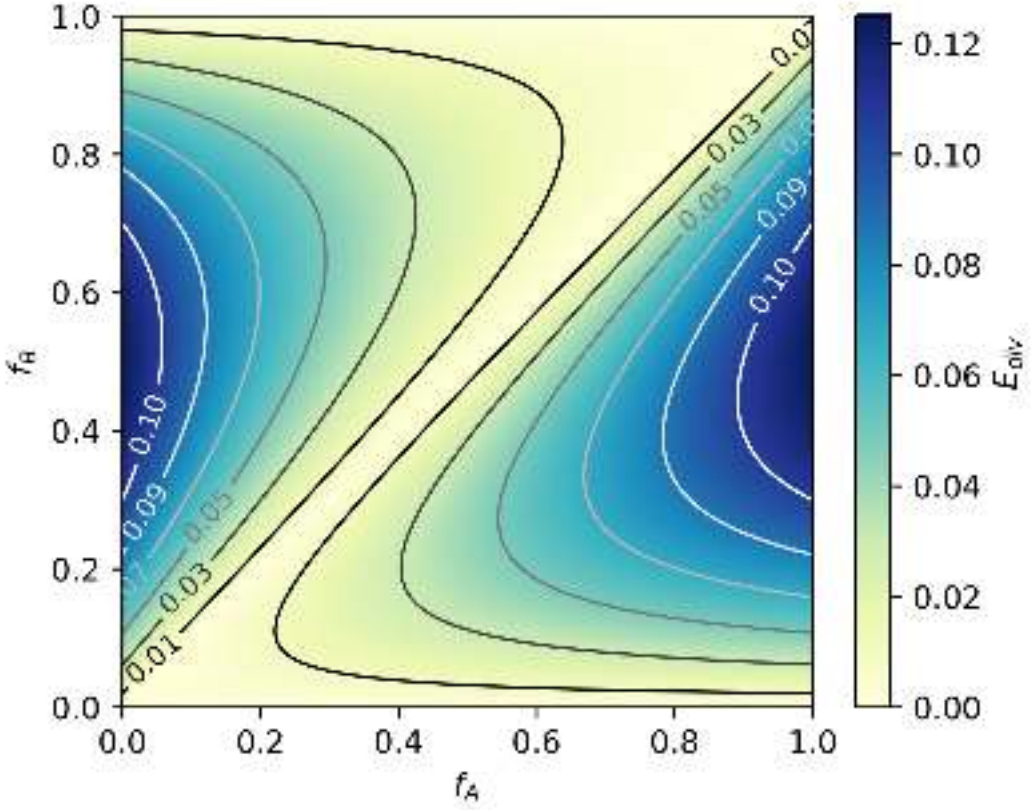
Error for PLAUs (II)

Multiplication



Relative error: -11.11% to 0 %

Division



Relative error: 0% to 12.5%

Outline

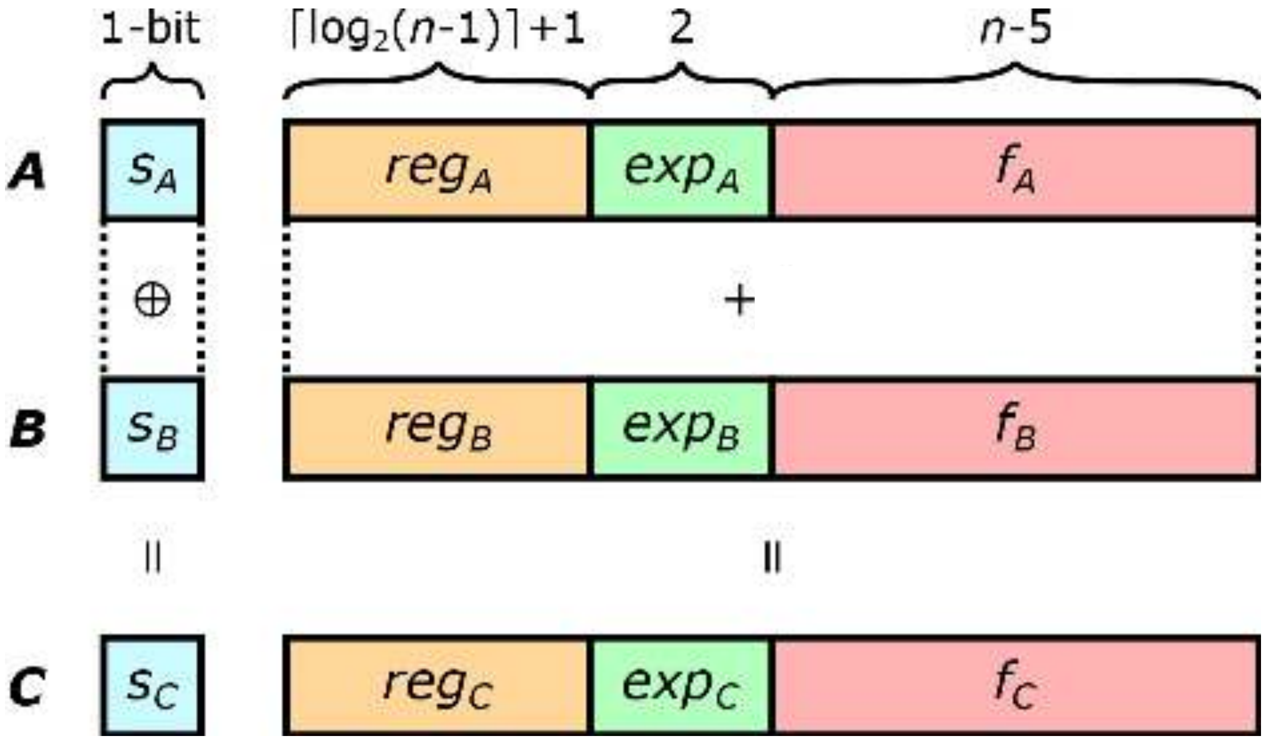
- Background
- Posit Logarithmic Approximate Units (PLAUs)
- **Implementation analysis**
- Applications
- Conclusions

Efficient HW Implementation for PLAUs

$$P_{approx} = (-1)^{s_A + s_B} \times 2^{k_A + k_B} \times (1 + f_A + f_B) \quad \text{if } f_A + f_B < 1$$

- for division!

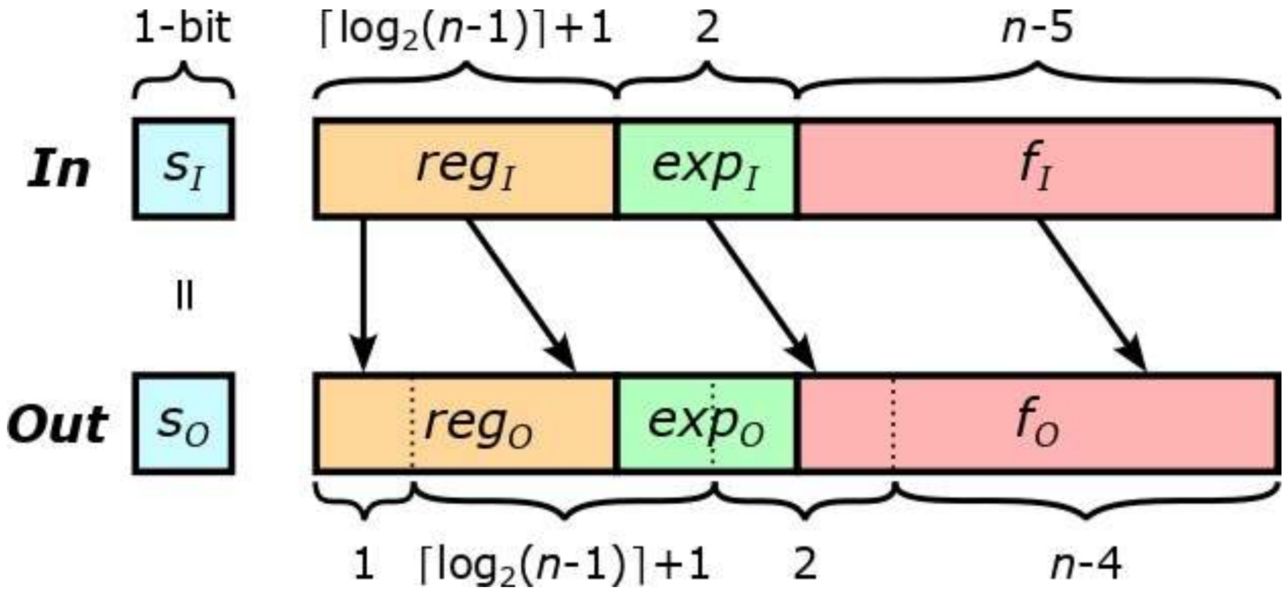
- Just single add and *xor*
- No need to handle separate cases



Efficient HW Implementation for PLAUs (II)

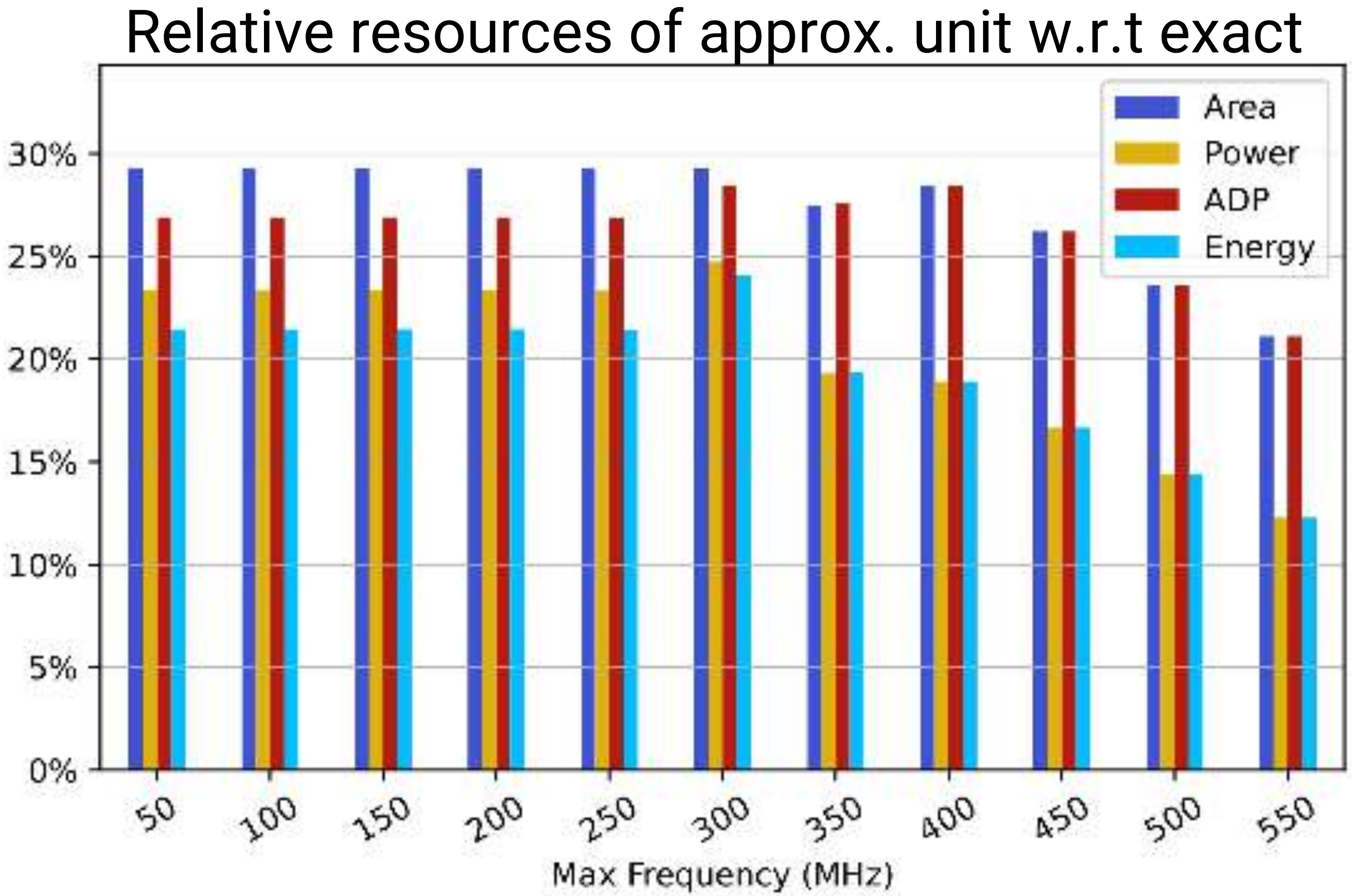
$$R_{approx} = 2^{k/2} \times (1 + f/2)$$

- Right shift
- Need to keep MSB
- Keep LSB for rounding



ASIC Synthesis – Multiplier

Synopsys DC
45-nm TSMC



- +70% area savings
- 75% – 80% less energy
- Up to 1150MHz

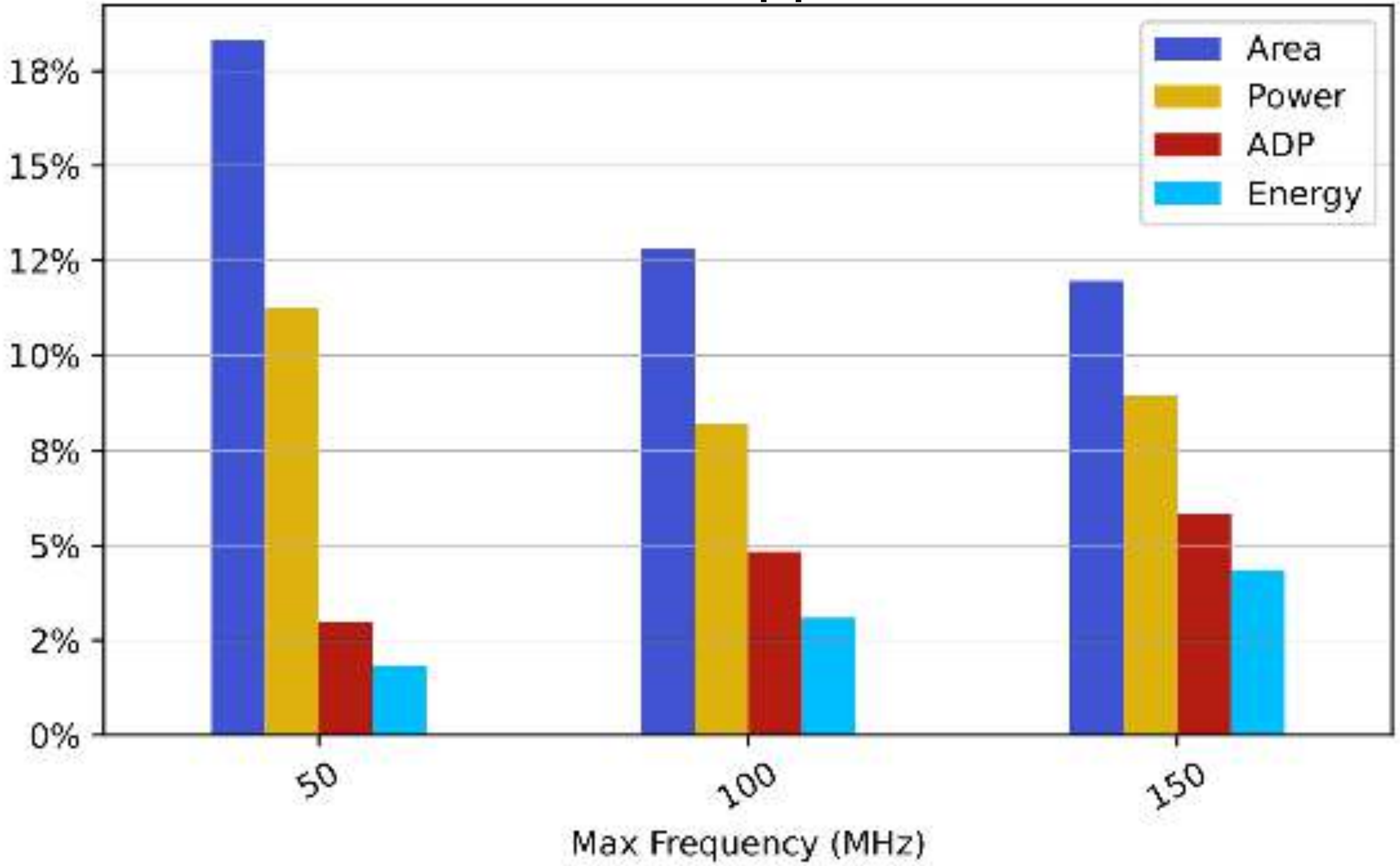
Exact Multiplier from:

[3] Raul Murillo, et al. "Comparing different decodings for posit arithmetic." *CoNGA 2022.*

ASIC Synthesis – Divider

Synopsys DC
45-nm TSMC

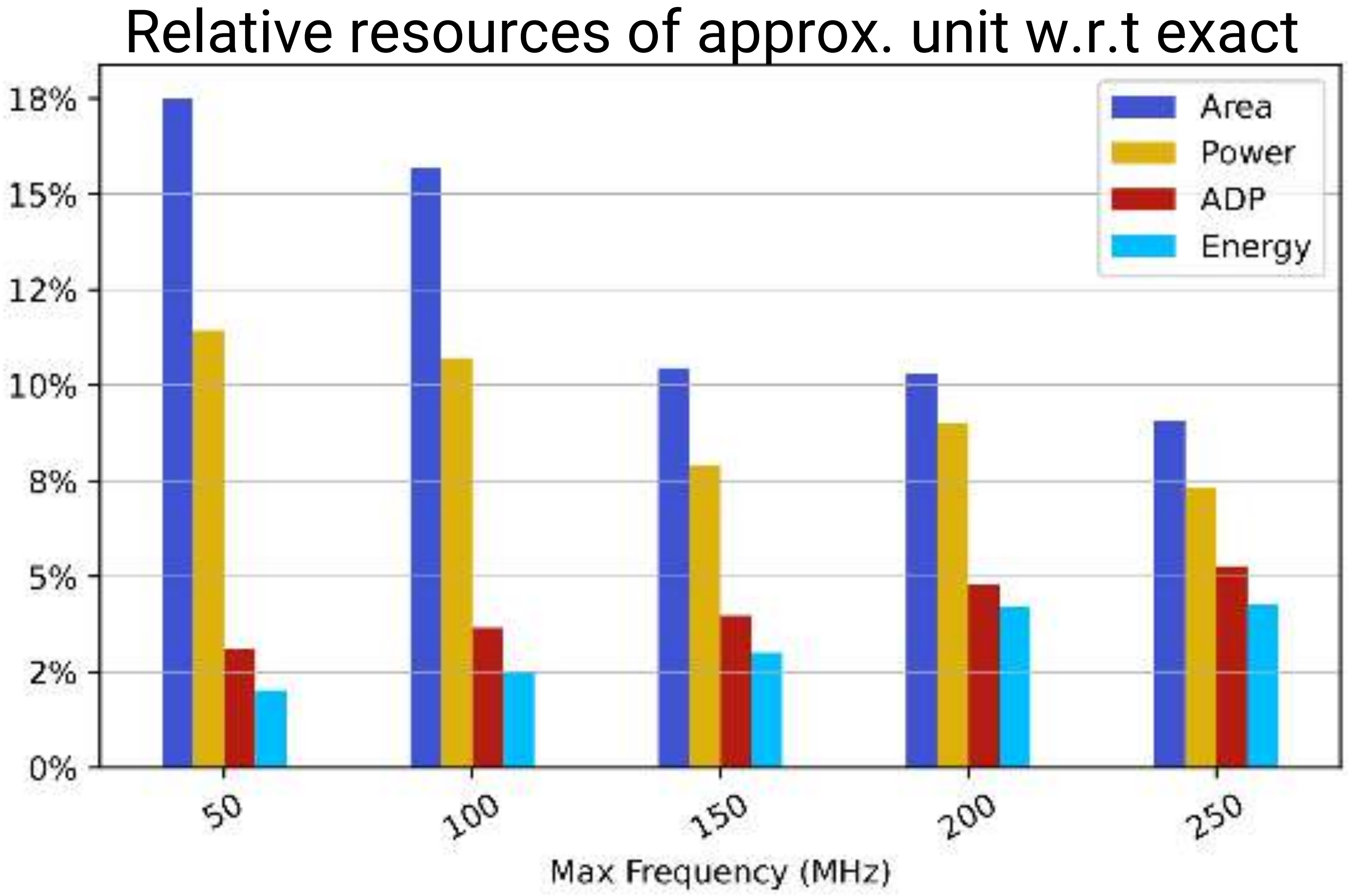
Relative resources of approx. unit w.r.t exact



- 81% – 89% area savings
- 89% – 91% power savings
- 99% – 96% less energy
- Up to 1150MHz

ASIC Synthesis – Square Root

Synopsys DC
45-nm TSMC



- Area reduction by 82% – 91%
- ADP and energy savings of ~98% – 94%
- Up to 1150MHz

Outline

- Background
- Posit Logarithmic Approximate Units (PLAUs)
- Implementation analysis
- **Applications**
- Conclusions

Applications – Computer Vision

- Blur filter (div)

Exact

Approximate

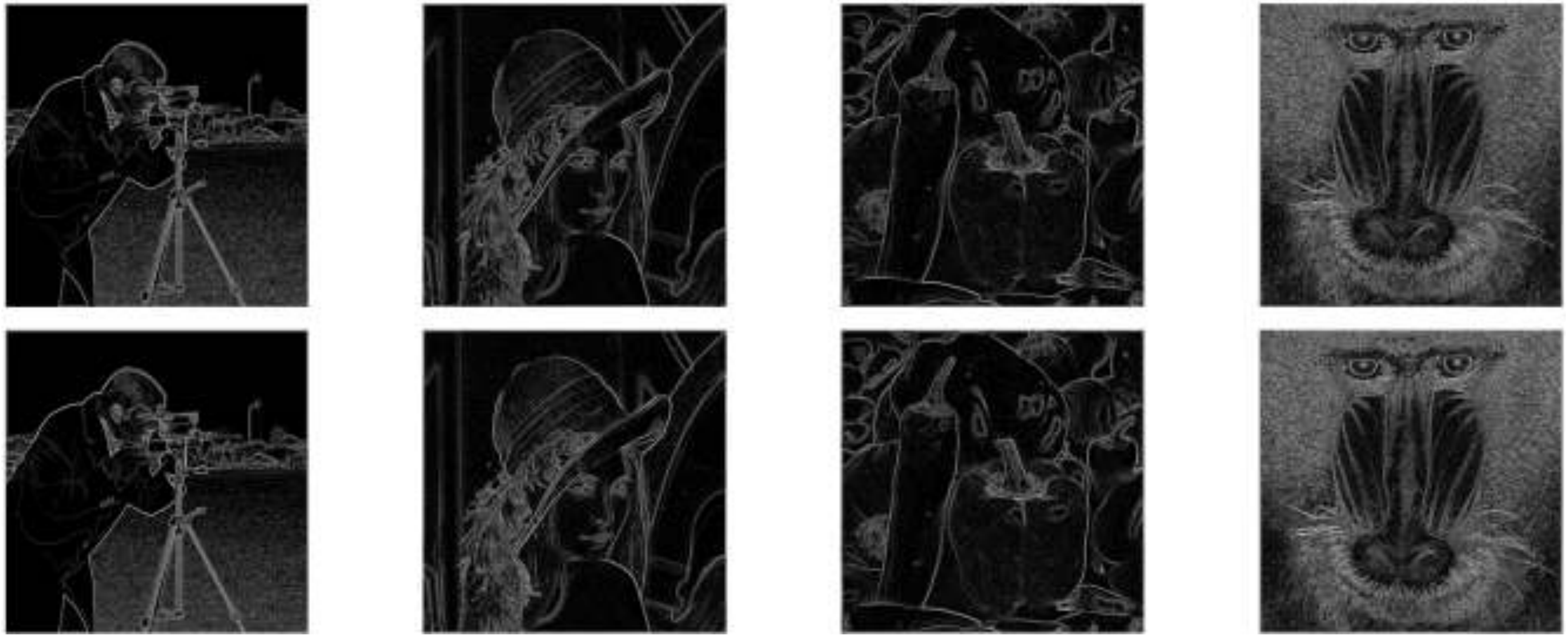
PSNR: 27.6019	PSNR: 29.0023	PSNR: 28.2892	PSNR: 28.7468
SSIM: 0.98217	SSIM: 0.98281	SSIM: 0.98318	SSIM: 0.97396

Applications – Computer Vision

- Sobel filter for edge detection (mul, div, sqrt)

Exact

Approximate



PSNR: 51.0439	PSNR: 44.5427	PSNR: 45.6212	PSNR: 38.2694
SSIM: 0.99909	SSIM: 0.99629	SSIM: 0.99629	SSIM: 0.99503

Applications – Machine Learning

- *K*-Nearest Neighbors (mul, sqrt)

Dataset	Instances	Attributes	Classes	*K*	Exact	Approx.
Iris	150	3	3	5	95.55%	95.55%
Wine	178	13	3	7	67.92%	67.92%
Glass	214	9	7	5	59.38%	59.38%
Breast Cancer	569	30	2	13	94.74%	94.74%

Outline

- Background
- Posit Logarithmic Approximate Units (PLAUs)
- Implementation analysis
- Applications
- **Conclusions**

Conclusions

- Posit arithmetic is raising interest for its properties…
- But still has a high implementation cost
- Posit Logarithmic Approximate Units (PLAUs)
  - Multiplication, division, and sqrt
  - Much faster circuits
  - Outstanding savings in area, power and energy
  - Relative error of -11.11%, 12.5% and 6.06%, respectively
  - Useful in error-tolerant applications (CV, ML, …)

2023 Conference on Next Generation Arithmetic (CoNGA)

Thank you!

PLAUs: Posit Logarithmic Approximate Units to implement low-cost operations with real numbers

Raul Murillo – ramuri01@ucm.es

David Mallasén, Alberto A. Del Barrio and Guillermo Botella

Complutense University of Madrid, Spain

https://github.com/RaulMurillo/CoNGA_23

UNIVERSIDAD COMPLUTENSE MADRID

ArTeCS

Fundación BBVA

Comunidad de Madrid

UNIÓN EUROPEA
Fondos Estructurales
Invertimos en su futuro

UNIÓN EUROPEA
Fondo Social Europeo
El Fondo Social Europeo invierte en tu futuro

Backup slides

Example Posit16

1 bit	4 bits				1 bit	2 bits		8 bits							
1	1	1	1	1	0	1	0	1	0	0	1	0	1	1	0
Sign	Regime					Exponent		Fraction							

$$p = ((1-3s)+f) \times 2^{(1-2s)\times(4r+e+s)}, \quad r = \begin{cases} -k & \text{if } R_0 = 0 \\ k-1 & \text{if } R_0 = 1 \end{cases}$$

$$p = ((1-3\cdot 1) + \frac{150}{2^8}) \times 2^{(1-2\cdot 1)\times(4\cdot 3+2+1)}, \quad r = 4-1 = 3$$

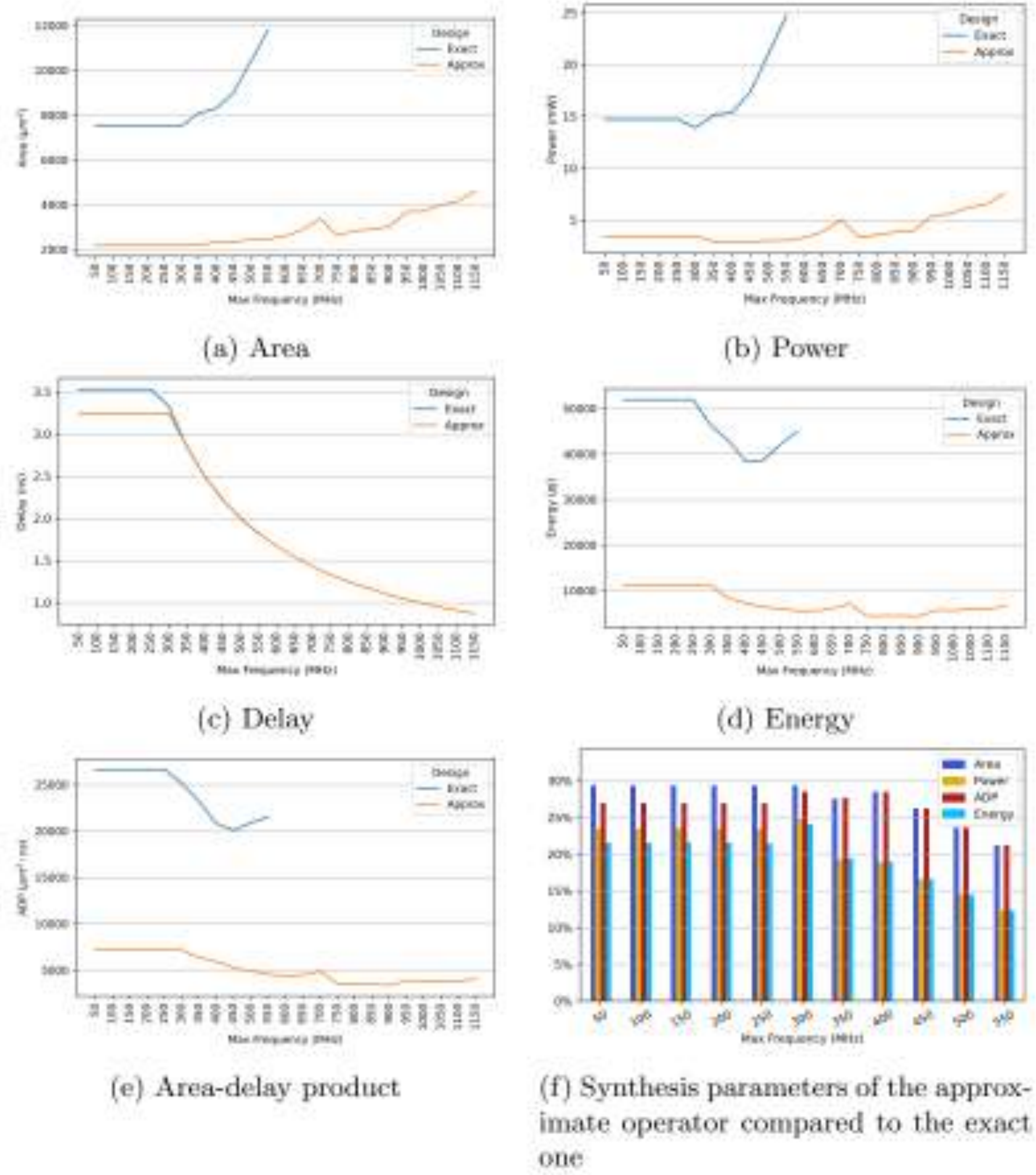
$$p = -0.000043154$$

Relative Error

$$E_{mul} = \begin{cases} \dfrac{1 + f_A + f_B}{(1 + f_A)(1 + f_B)} - 1 & \text{if } f_A + f_B < 1, \\[2ex] \dfrac{2(f_A + f_B)}{(1 + f_A)(1 + f_B)} - 1 & \text{if } f_A + f_B \geq 1. \end{cases}$$

$$E_{div} = \begin{cases} \dfrac{(1 + f_A - f_B)(1 + f_B)}{(1 + f_A)} - 1 & \text{if } f_A - f_B \geq 0, \\[2ex] \dfrac{(2 + f_A - f_B)(1 + f_B)}{2(1 + f_A)} - 1 & \text{if } f_A - f_B < 0. \end{cases}$$

$$E_{sqrt} = \frac{1 + f/2}{\sqrt{1 + f}} - 1.$$

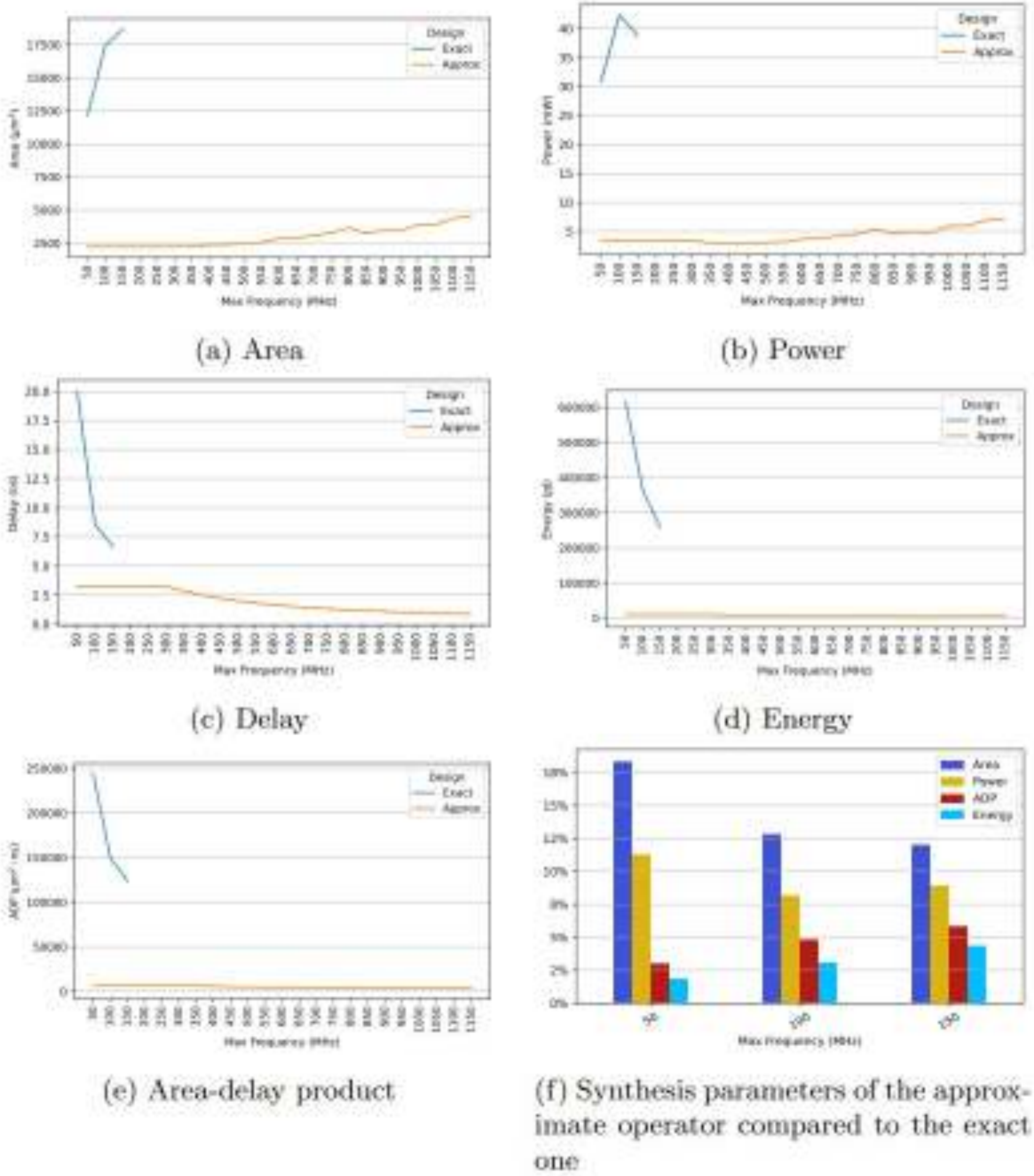


(a) Area (b) Power

(c) Delay (d) Energy

(e) Area-delay product (f) Synthesis parameters of the approximate operator compared to the exact one

Fig. 4: Comparison of multiplication implementations for Posit⟨32, 2⟩.



(a) Area (b) Power

(c) Delay (d) Energy

(e) Area-delay product (f) Synthesis parameters of the approximate operator compared to the exact one

Fig. 5: Comparison of division implementations for Posit⟨32, 2⟩.

Comparison of Posit Unit and FPU

- FPGA synthesis targeting Genesys II (Vivado 2020.2) [4]

Arithmetic unit (LUT, FF)	Posit⟨32,2⟩	+Single IEEE 754	+Double IEEE 754
PAU Area	(11879, 2985)	(11796, 2979)	(11810, 2979)
FPU Area	–	(3726, 1008)	(6352, 1905)
Total Area	(44693, 23636)	(50318, 25727)	(55900, 27652)

[4] D. Mallasén, R. Murillo, et al. "PERCIVAL: Open-Source Posit RISC-V Core with Quire Capability." *arXiv preprint arXiv:2111.15286* (2021). Source code available at github.com/artecs-group/PERCIVAL