Posits: A Summary

Alexandra Poulos

Holcombe Department of Electrical and Computer Engineering
Clemson University
Clemson, SC 29634
alpoulo@clemson.edu

06 October 2021

Outline

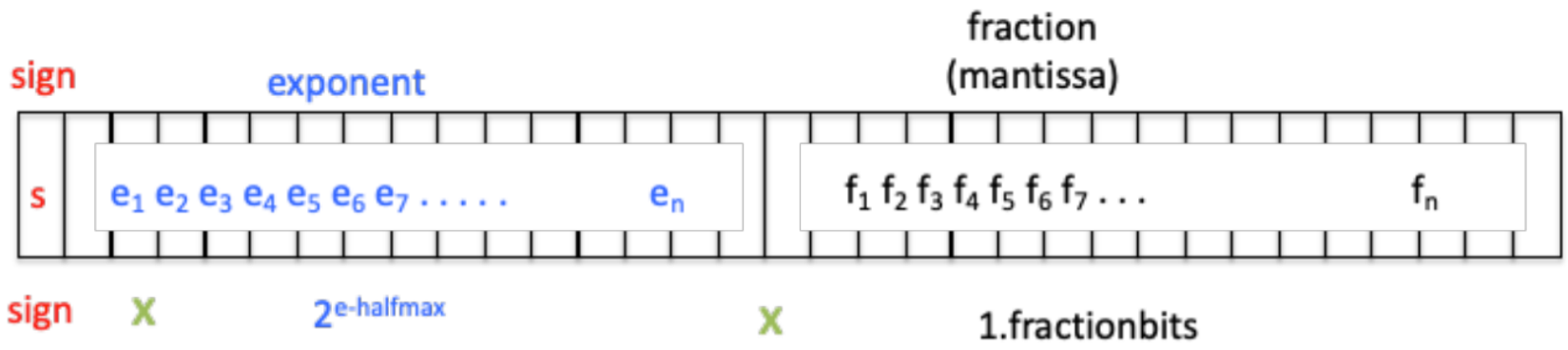
1. Introduction/Background
2. Posits
3. Applications of Posits

Outline

1. Introduction/Background
2. Posits
3. Applications of Posits

IEEE-754 floating point standard

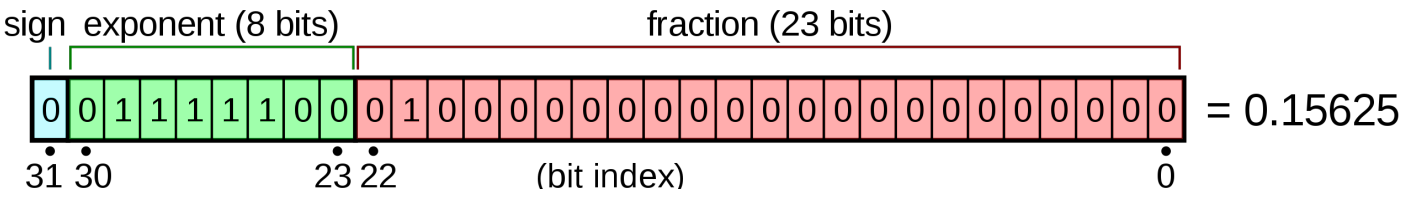
An *IEEE-754 float* is structured as follows:



and is numerically interpreted as

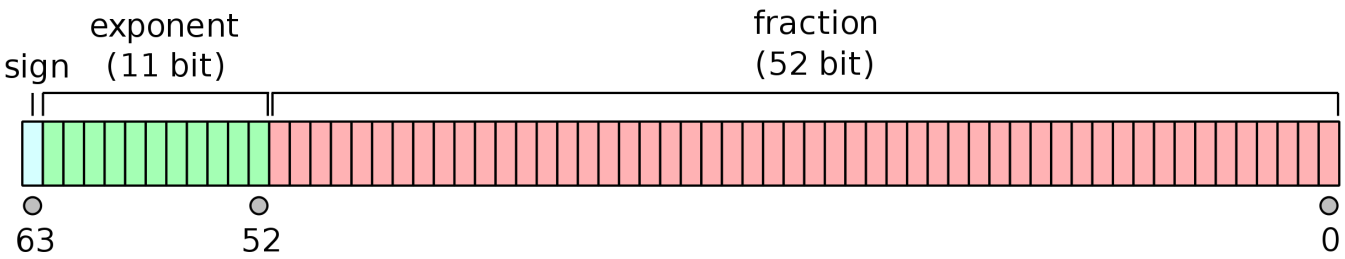
$$x = (-1)^{sign} \times 2^{exponent-bias} \times 1.mantissa$$

IEEE-754 floating-point formats



FP32

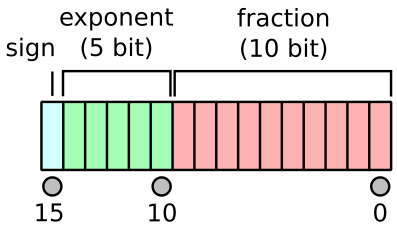
Source: Source: https://commons.wikimedia.org/w/index.php?curid=3357169



FP64

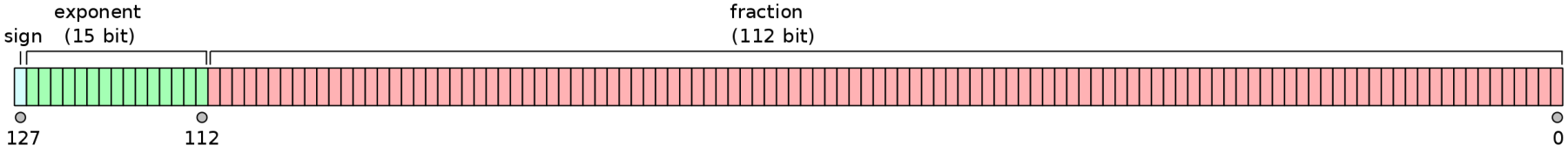
Source: https://en.wikipedia.org/wiki/Double-precision_floating-point_format

IEEE-754 floating-point formats



FP16

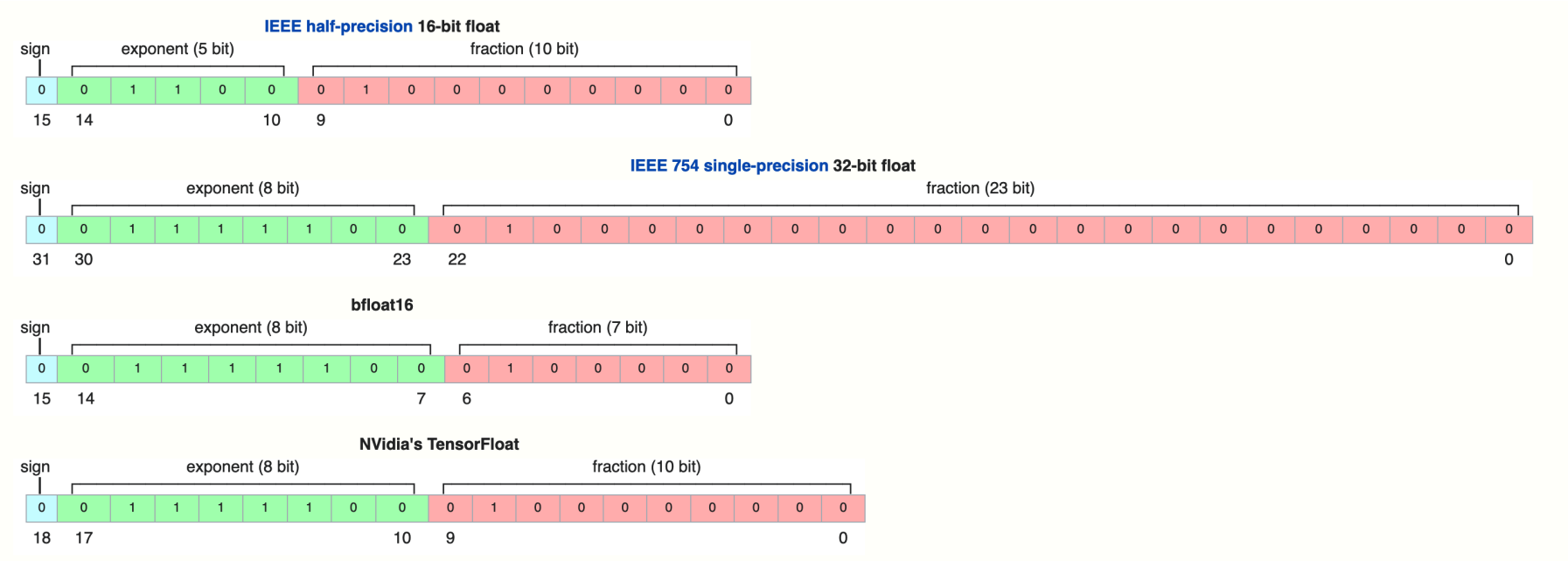
Source: https://en.wikipedia.org/wiki/Half-precision_floating-point_format



FP128

Source: https://en.wikipedia.org/wiki/Quadruple-precision_floating-point_format

Other floating-point representations



Comparison of different floating-point types
Source: https://handwiki.org/wiki/Bfloat16_floating-point_format

Outline

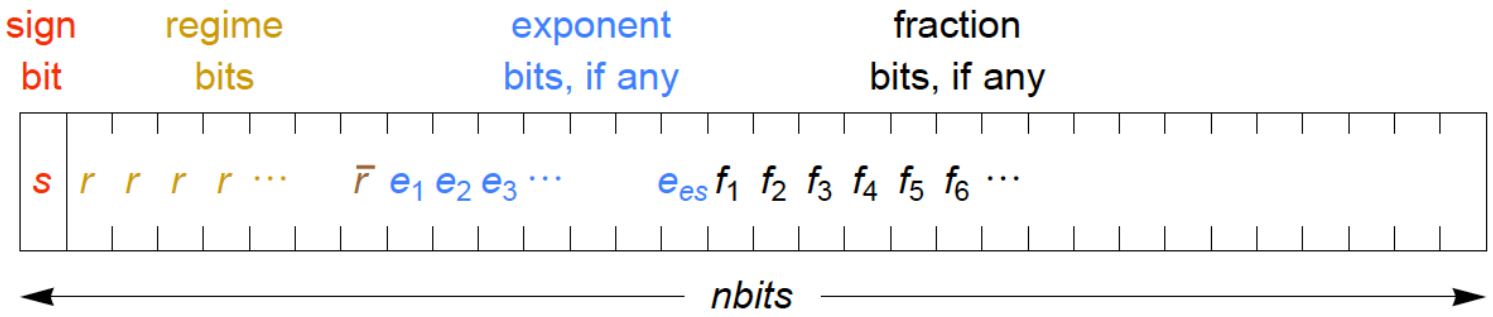
1. Introduction/Background
2. Posits
3. Applications of Posits

Posits

A *posit type* is denoted $\langle n, es \rangle$, where

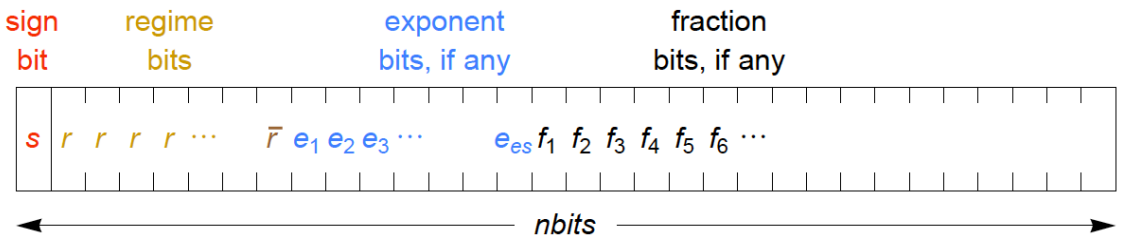
- n is the width of the posit, in bits
- es is the *maximum* exponent size, in bits

A posit is structured as follows:



(Gustafson and Yonemoto, 2017)

Posits

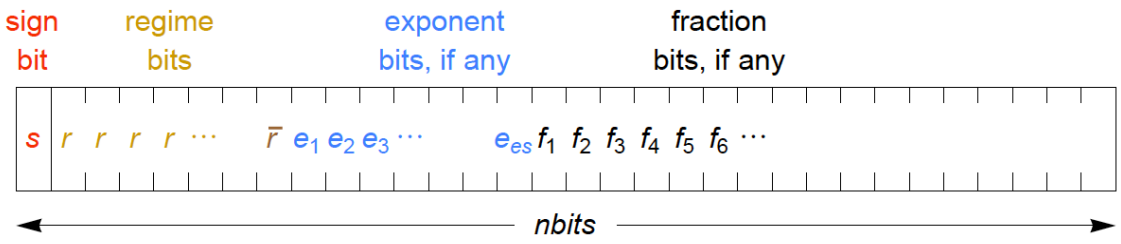


A posit is numerically interpreted as

$$x = \begin{cases} 0 & p = 0 \\ \pm\infty & p = -2^{n-1} \\ (-1)^{sign} \times useed^{k} \times 2^{exponent} \times 1.fraction & \text{all other } p \end{cases}$$

where $useed = 2^{2^{es}}$ and k is the number of regime bits.

Posits



A posit is numerically interpreted as

$$x = \begin{cases} 0 & p = 0 \\ \pm\infty & p = -2^{n-1} \\ (-1)^{sign} \times useed^{k} \times 2^{exponent} \times 1.fraction & \text{all other } p \end{cases}$$

where $useed = 2^{2^{es}}$ and k is the number of regime bits.

Binary	0000	0001	001x	01xx	10xx	110x	1110	1111
Numerical meaning, k	−4	−3	−2	−1	0	1	2	3

Posits

Example: $p = \langle 16, 3 \rangle$

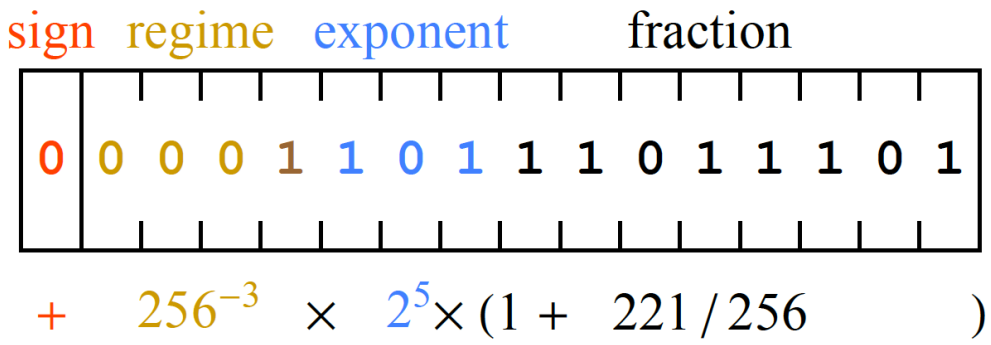
sign	regime			exponent				fraction							
0	0	0	0	1	1	0	1	1	1	0	1	1	1	0	1

$$+ \quad 256^{-3} \times 2^{5} \times (1 + 221/256)$$

(Gustafson and Yonemoto, 2017)

Posits

Example: $p = \langle 16, 3 \rangle$

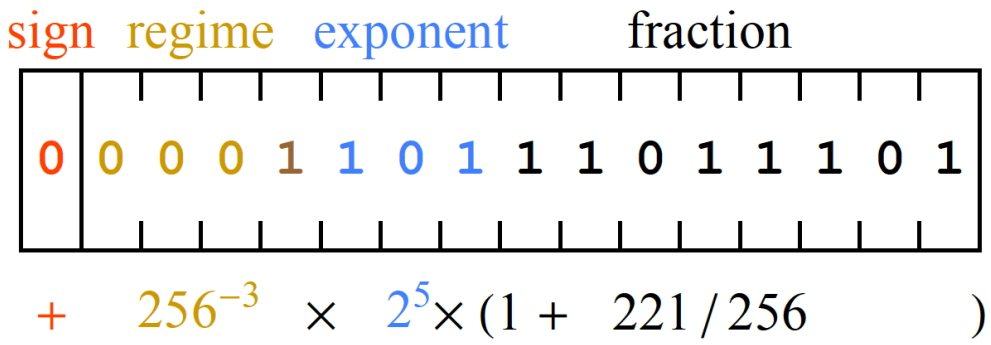


(Gustafson and Yonemoto, 2017)

- $useed = 2^{2^3} = 256$

Posits

Example: $p = \langle 16, 3 \rangle$

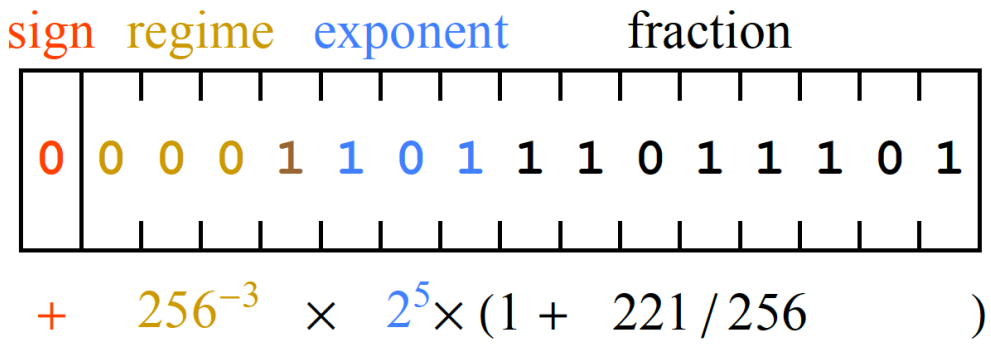


(Gustafson and Yonemoto, 2017)

- $useed = 2^{2^3} = 256$
- regime is three 0's terminated by 1, so $k = -3$

Posits

Example: $p = \langle 16, 3 \rangle$

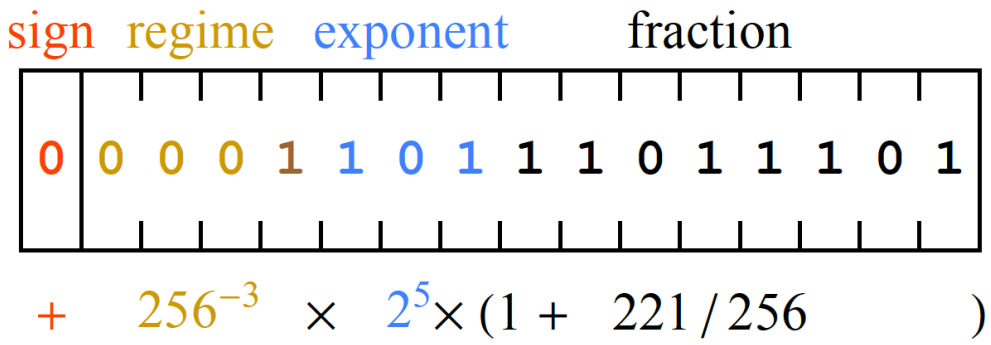


(Gustafson and Yonemoto, 2017)

- $useed = 2^{2^3} = 256$
- regime is three 0's terminated by 1, so $k = -3$
- exponent is interpreted as an unsigned integer

Posits

Example: $p = \langle 16, 3 \rangle$

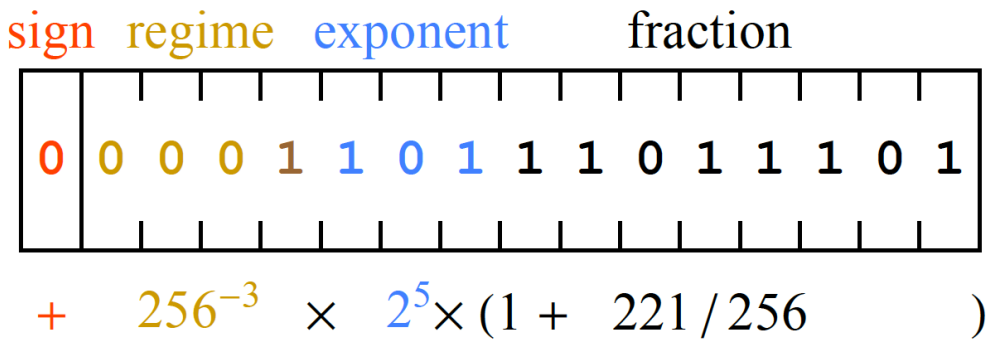


(Gustafson and Yonemoto, 2017)

- $useed = 2^{2^3} = 256$
- regime is three 0's terminated by 1, so $k = -3$
- exponent is interpreted as an unsigned integer
- fraction interpreted same as IEEE floats

Posits

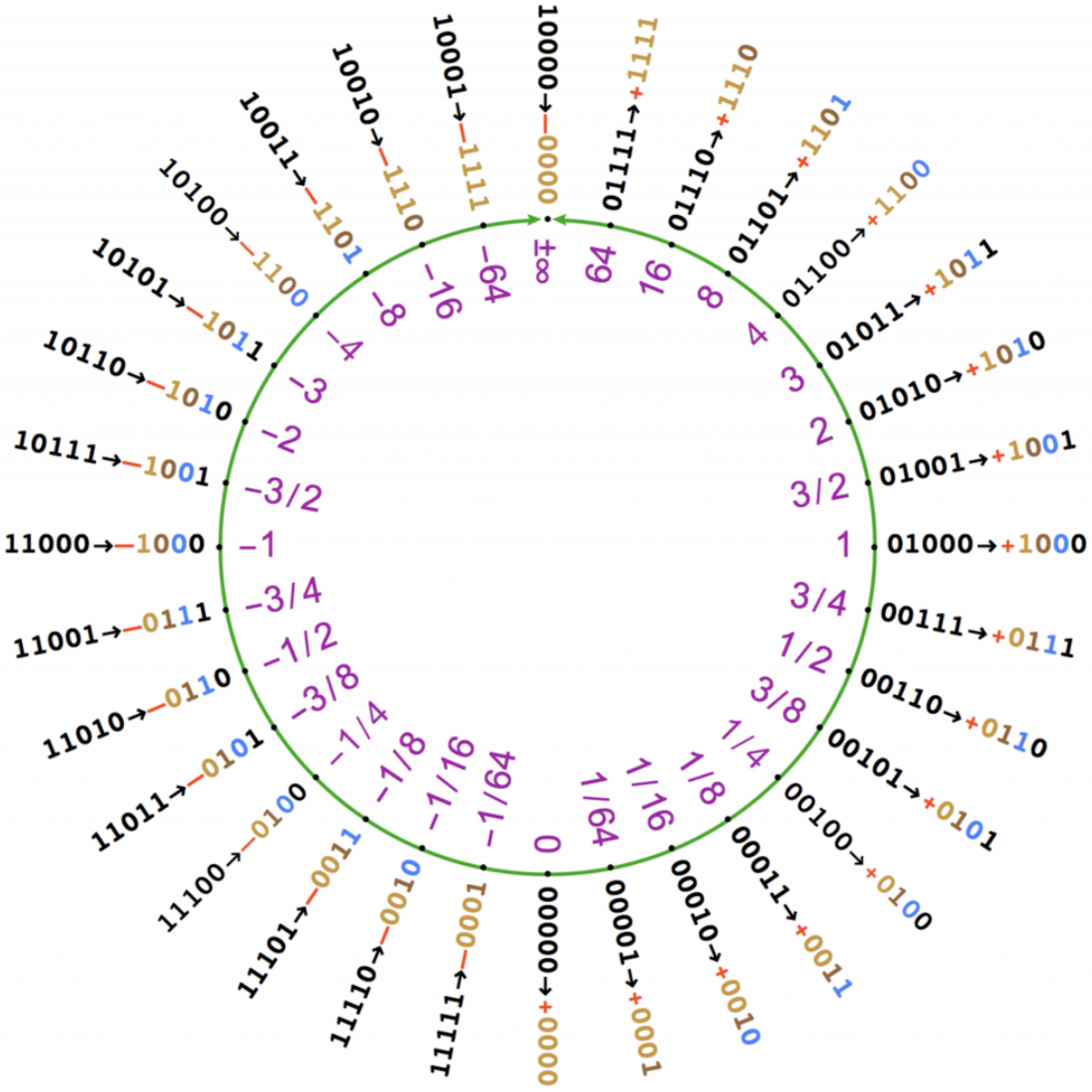
Example: $p = \langle 16, 3 \rangle$



(Gustafson and Yonemoto, 2017)

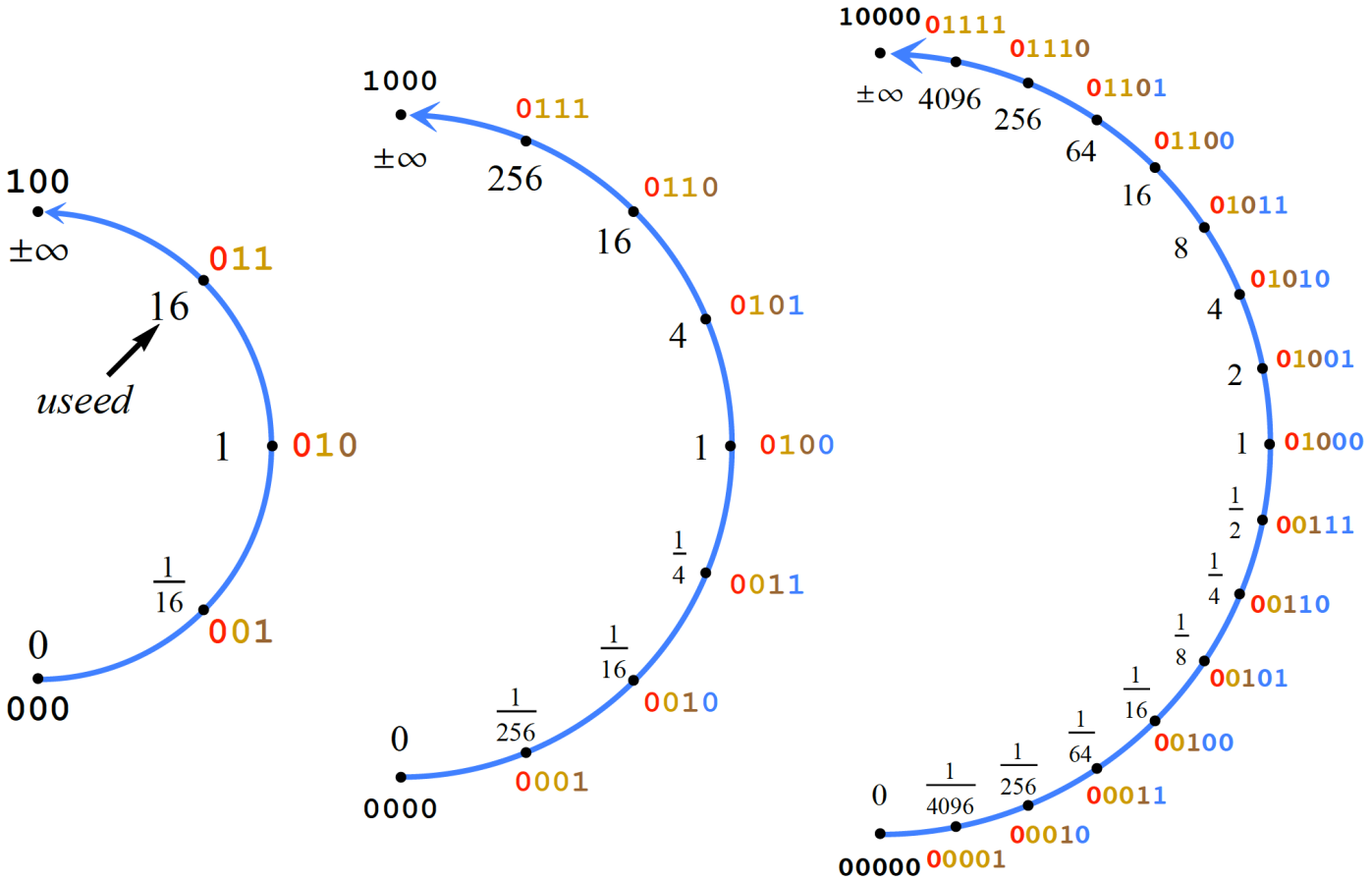
- $useed = 2^{2^3} = 256$
- regime is three 0's terminated by 1, so $k = -3$
- exponent is interpreted as an unsigned integer
- fraction interpreted same as IEEE floats
- So our posit evaluates to $477/134217728 \approx 3.55393 \times 10^{-6}$

Posit properties



Posit $\langle 5, 1 \rangle$ ring plot (Gustafson, 2017)

Posit properties



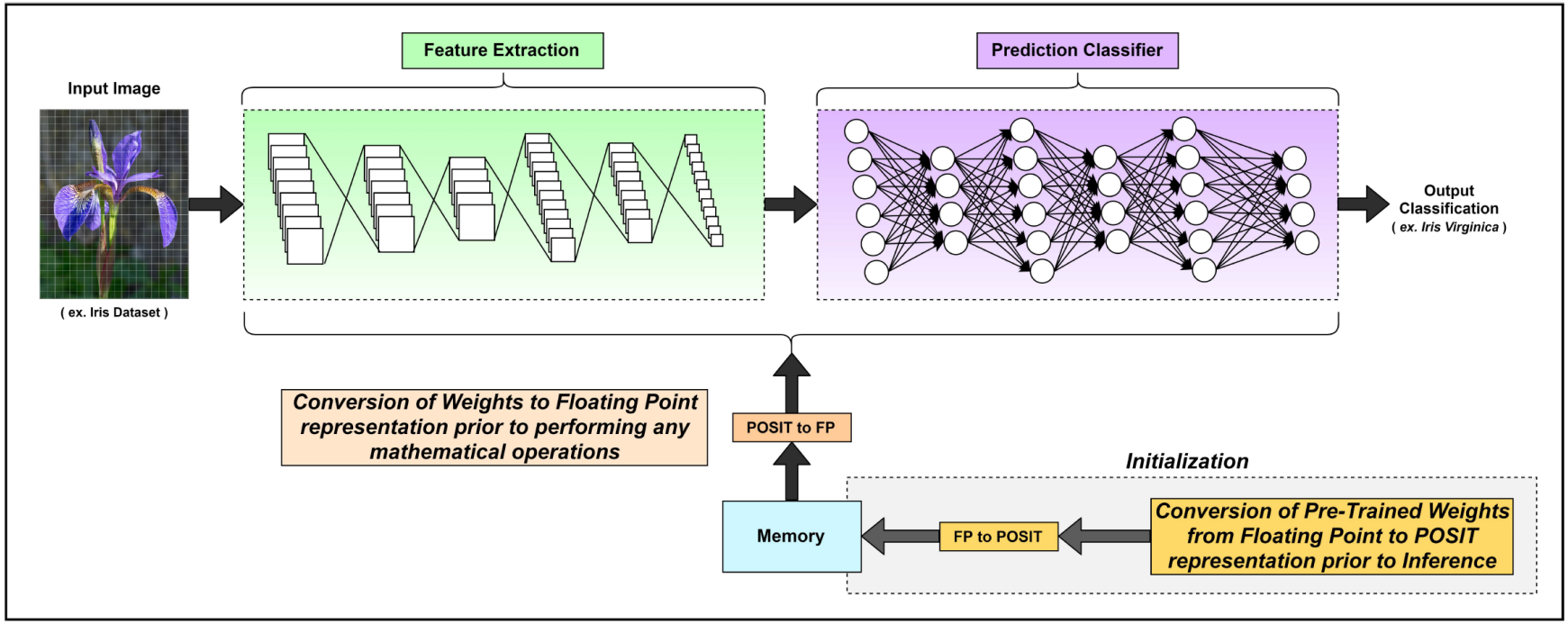
Posit construction with two exponent bits, *es*=2, *useed* $= 2^{2^{es}} = 16$
(Gustafson and Yonemoto, 2017)

Outline

1. Introduction/Background
2. Posits
3. Applications of Posits

Applications of posits

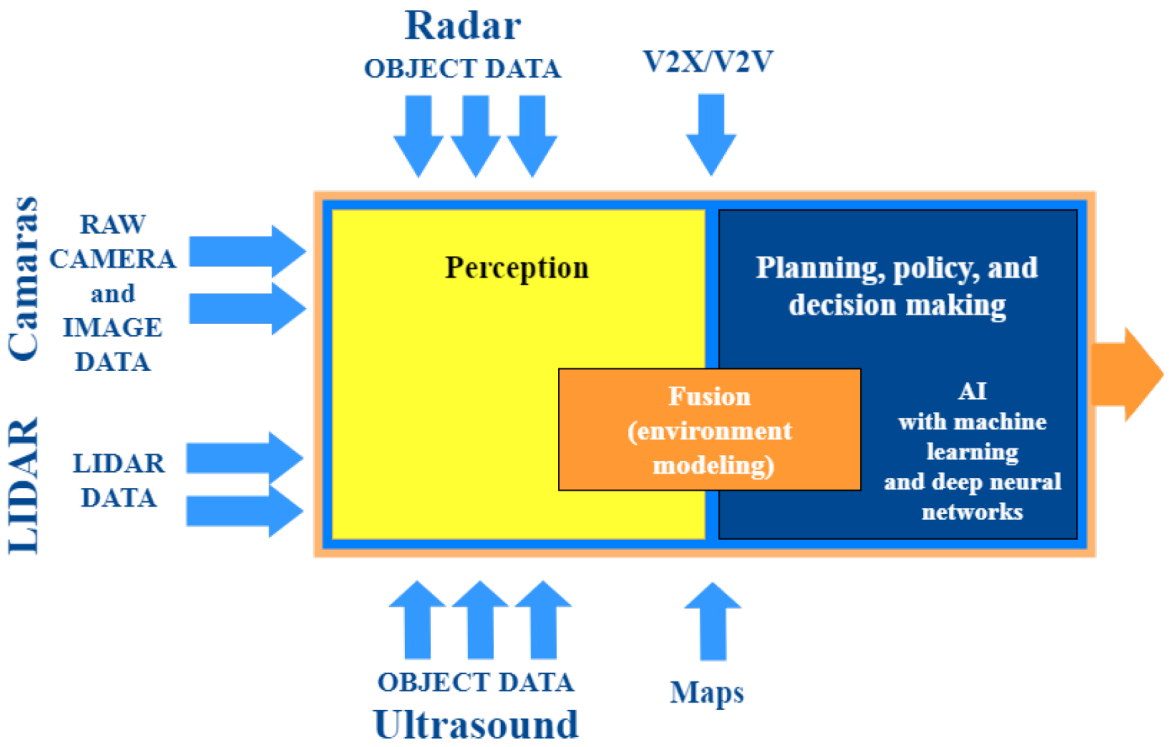
Deep Learning Inference on Embedded Devices: Fixed-Point vs Posit (Langroudi, Pandit, and Kudithipudi, 2018)



High level overview of the DCNN architecture implementation which uses the posit number system to represent the weights

Applications of posits

Exploiting Posit Arithmetic for Deep Neural Networks in Autonomous Driving Applications (Cococcioni, Ruffaldi, and Saponara, 2018)



AD architecture with perception (left) based on sensor data fusion and planning (right). Both functions can take advantage of AI approaches in particular deep learning techniques

Applications of posits

Deep PeNSieve: A deep learning framework based on the posit number system (Murillo, Del Barrio, and Botella, 2020)

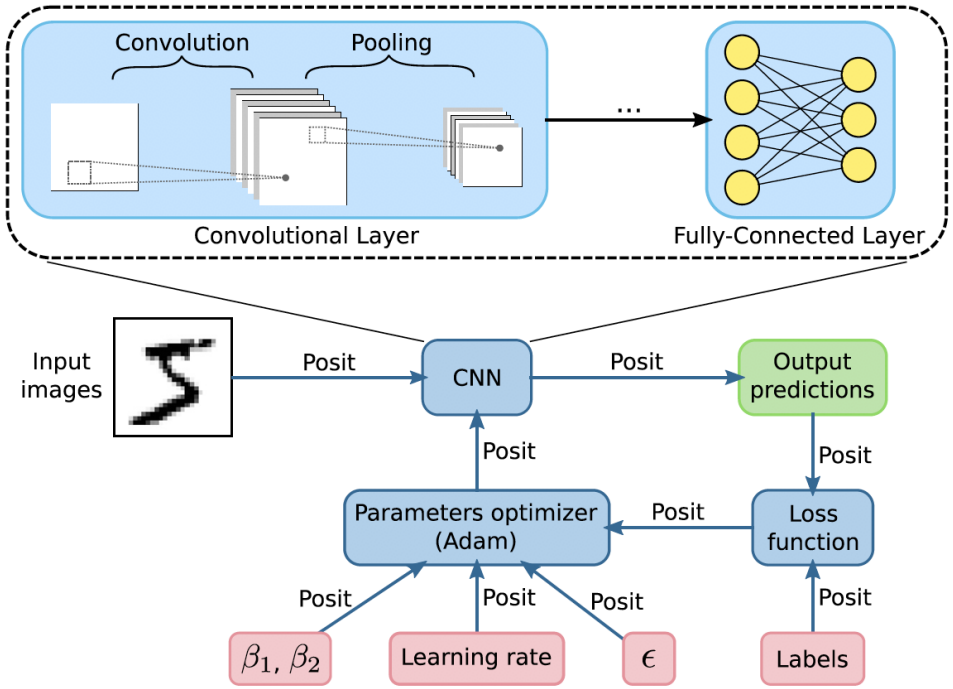
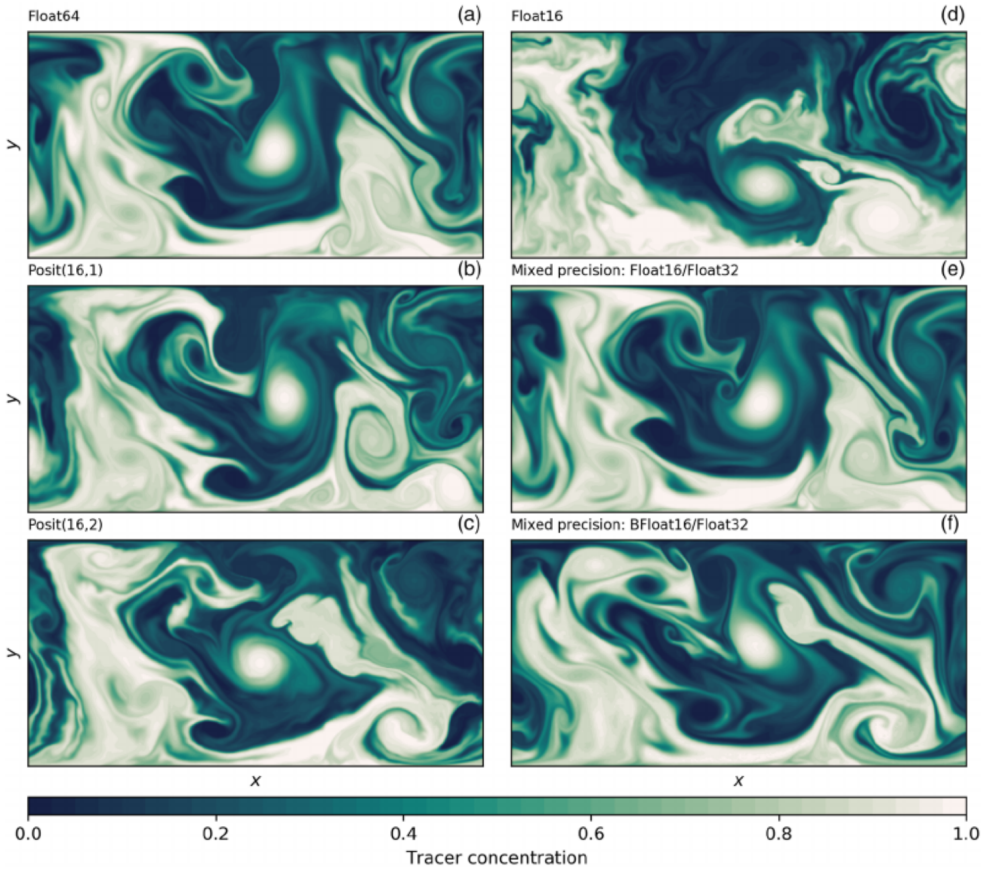


Illustration of the training flow of the proposed framework. Red boxes represent input data, including hyperparameters, while the blue boxes correspond with the functionalities performed by the TensorFlow framework.

Applications of posits

Number Formats, Error Mitigation, and Scope for 16-Bit Arithmetics in Weather and Climate Modeling Analyzed With a Shallow Water Model (Klöwer, Düben, and Palmer, 2020)



Shallow water simulation of vigorous turbulence interacting with a zonal current

Current Limitations and Open Problems

- No finalized standard for posit implementations
- Need further research into the use of posits in different domains
- Hardware implementations need to be further developed

Conclusion

- Posit is a new data type proposed as an alternative to IEEE floats
- Algorithms/use-cases for posits need to be further developed/explored
- Can offer superior numerical performance when used appropriately at a lower cost

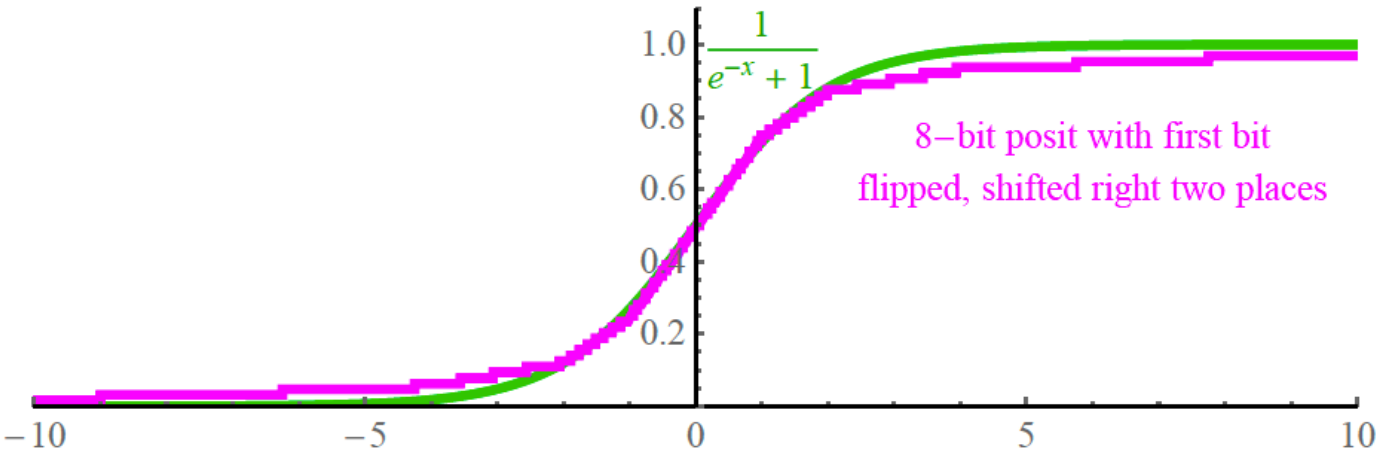
Thank you

Questions?

Backup Slides

Applications of posits

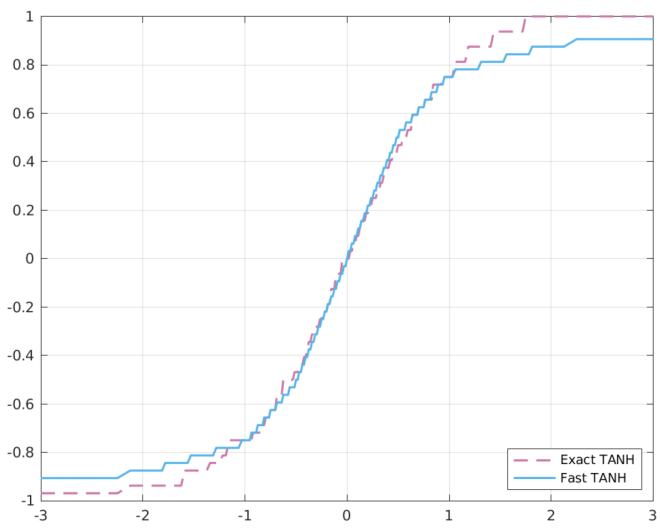
Beating Floating-point at it's Own Game: Posit Arithmetic (Gustafson and Yonemoto, 2017)

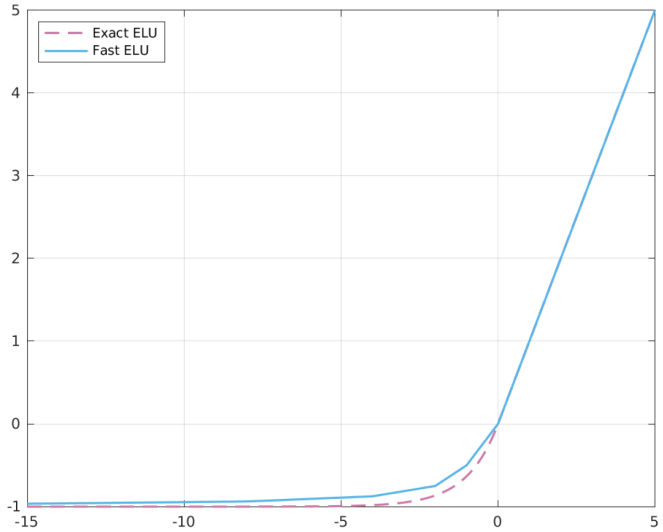


Fast sigmoid function with $\langle 8, 0 \rangle$ posit for neural network training

Applications of posits

Fast Approximations of Activation Functions in Deep Neural Networks when using Posit Arithmetic (Cococcioni et al., 2020)

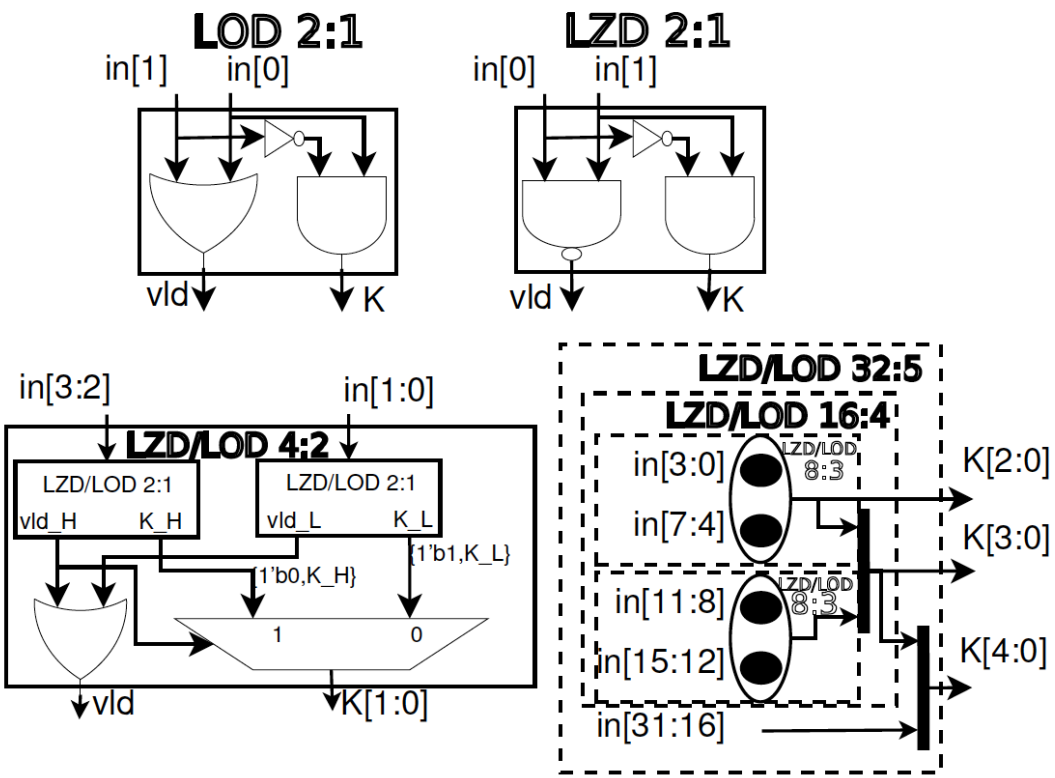




Comparison between the exact and approximated versions of hyperbolic tangent (TANH) and extended linear unit (ELU).

Hardware implementations of posits

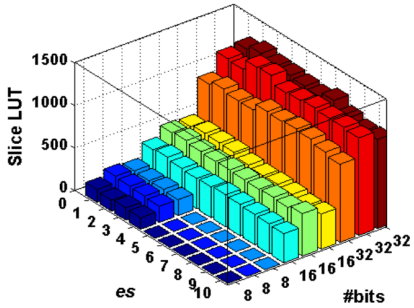
Architecture Generator for Type-3 Unum Posit Adder/Subtractor (Jaiswal and So, 2018)



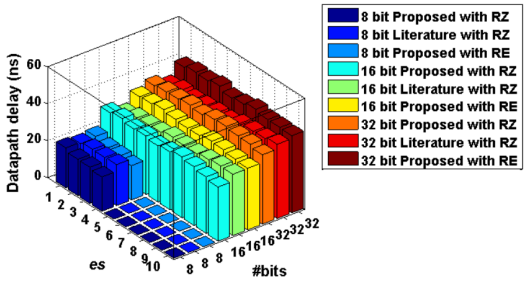
LOD and LZD architectural design

Hardware implementations of posits

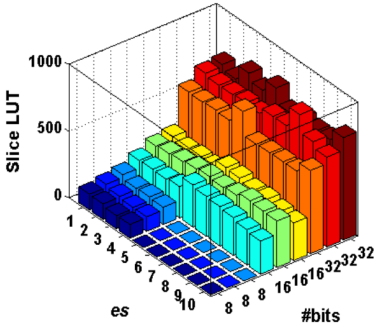
Parameterized Posit Arithmetic Hardware Generator (Chaurasiya and Gustafson, 2018)



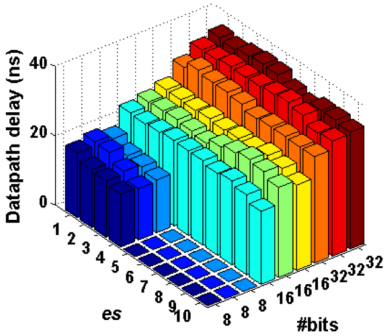
(a) Posit Adder Resource Utilization (Proposed Vs. Literature)



(b) Posit Adder Datapath Delay (Proposed Vs. Literature)



(c) Posit Multiplier Resource Utilization (Proposed Vs. Literature)



(d) Posit Multiplier Datapath Delay (Proposed Vs. Literature)

Comparison of posit adder and multiplier resource utilization and data path delay in FPGA synthesis (proposed vs. Jaiswal and So., 2018)

Hardware implementations of posits

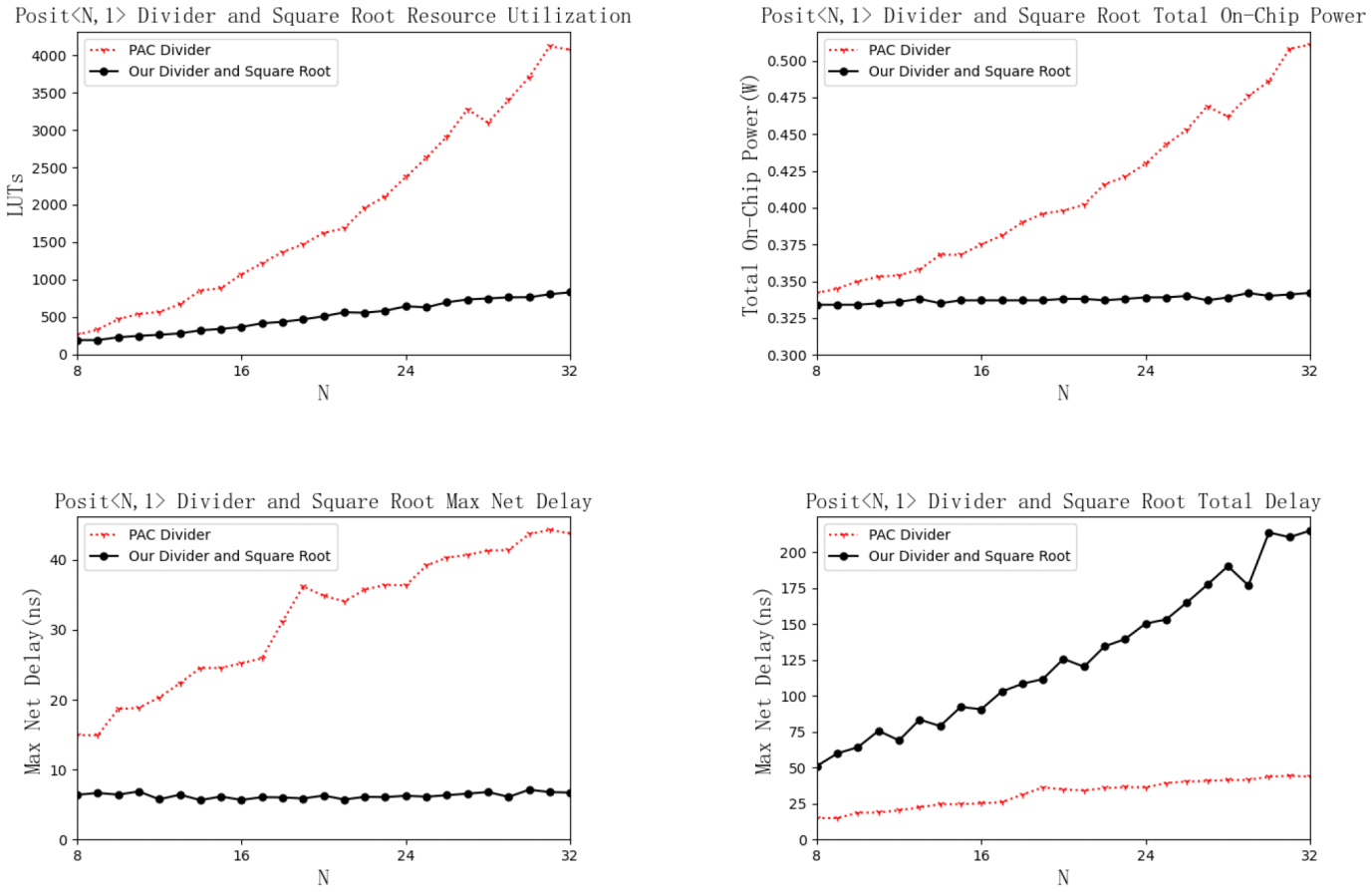
PACoGen: A Hardware Posit Arithmetic Core Generator (Jaiswal and So, 2019)

Posit Format	Posit Adder		Posit Multiplier	
	Proposed	[12]	Proposed	[12]
(16,1)	445x16.146 7184.9	391x32.374 12658	273x14.574 3978	218x24.041 5240.9
(16,2)	492x16.410 8073.7	404x33.974 13725.5	273x14.332 3912.6	223x23.680 5280.6
(16,3)	450x16.989 7645	386x32.466 12531.9	280x14.615 4092.2	219x24.078 5273.1
(32,1)	1211x19.150 23190.65	934x38.041 35530.3	668x19.076 12742.77	576x31.013 17863.5
(32,2)	1275x19.984 25480	981x40.032 39271.4	680x18.921 12866.28	572x33.021 18888.1
(32,3)	1272x19.752 25125	951x39.254 37330.554	720x19.203 13826.2	582x32.263 18777.1

Comparison of "Area (LUT) X Period (ns)" product for posit adder and multiplier on Virtex-7 FPGA (proposed vs. Chaurasiya and Gustafson, 2018)

Hardware implementations of posits

Posit Arithmetic Hardware Implementations with The Minimum Cost Divider and SquareRoot (Xiao et al., 2020)



Posit$\langle N, 1\rangle$ divider and square root results compared to PACoGen divider

Hardware implementations of posits

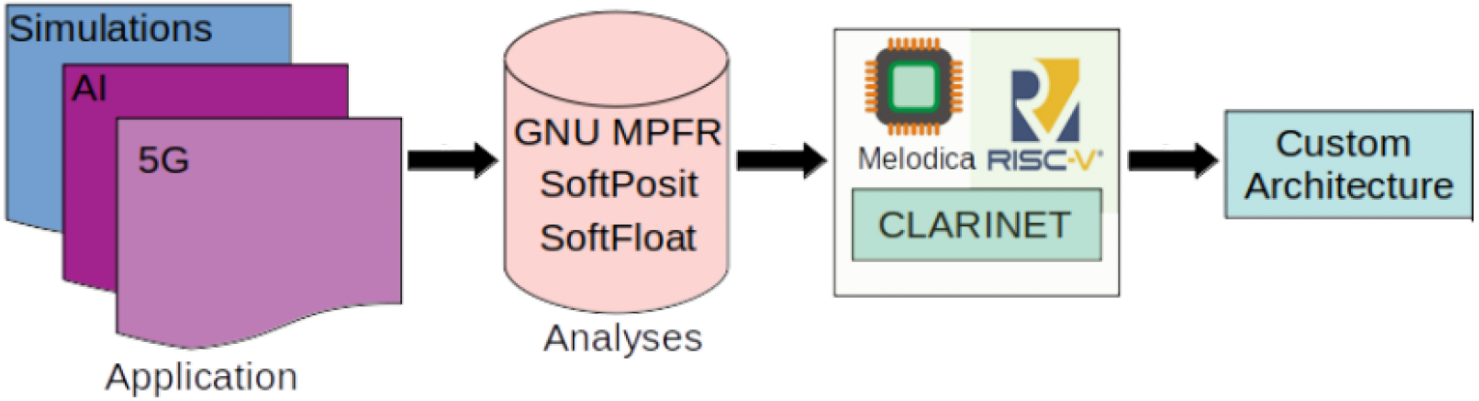
PERI: A Posit Enabled RISC-V Core (Tiwari et al., 2019)

Module	**Posit(es=2)**		**Posit(es=3)**		**Posit(es=2,3)**	
	Slice LUTs	Slice Registers	Slice LUTs	Slice Registers	Slice LUTs	Slice Registers
Fused Multiply-Add	1128	416	1154	415	1176	424
Division	281	270	303	237	279	271
Square root	138	145	135	143	138	147
Integer to Posit	131	37	128	36	131	37
Posit to Integer	163	39	161	39	165	39
Sign Injection	18	0	18	0	18	0
Classify	2	0	2	0	2	0
Decode Posit	554	0	555	0	684	0
Encode Posit	306	0	307	0	371	0
Glue Logic	307	284	281	311	543	376
Total	**3028**	**1191**	**3044**	**1181**	**3507**	**1294**

Module-wise Slice LUTs and Slice Registers with posit FPU of es=2, es=3 and es=2,3 for 32-bit posit size at 100 MHz

Hardware implementations of posits

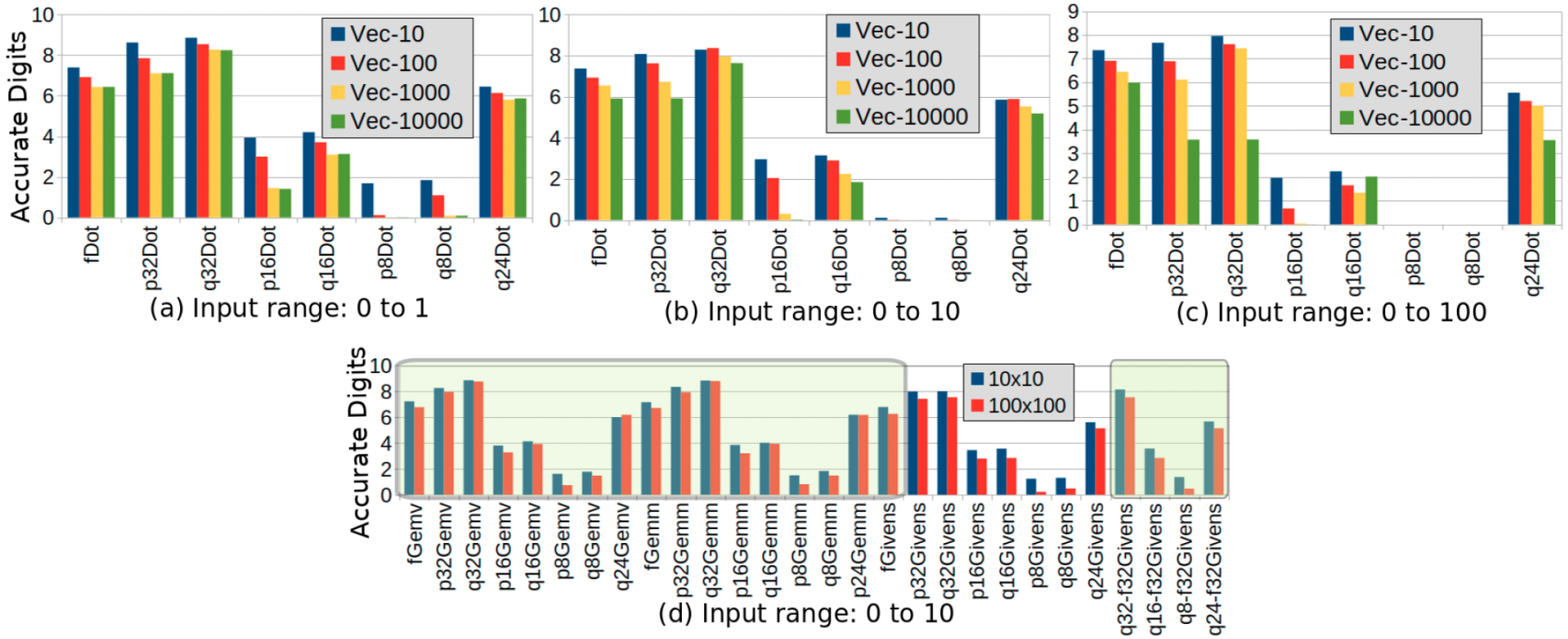
CLARINET: A RISC-V Based Framework for Posit Arithmetic Empiricism (Jain et al., 2020)



Clarinet in the platform design cycle

Hardware implementations of posits

CLARINET: A RISC-V Based Framework for Posit Arithmetic Empiricism (Jain et al., 2020)



Accurate number of digits with different data types in BLAS and LAPACK