High-level .NET Software Implementations of the Unum Type I and Posit Floating-Point Number Types with Simultaneous FPGA Implementation Using Hastlayer

Zoltán Lehóczky, Álmos Szabó @ Lombiq

GPU Day 21 June 2018


lombiq

be the hardware

Warning

Hastlayer is currently in alpha stage and posit work is ongoing!

Introducing Hastlayer

computer program → hardware (FPGA) logic

FPGAs (Field-Programmable Gate Array)

- Can behave like any other chip (with limitations)
- Can dynamically be „re-wired”
- Power efficient and highly parallelized

Image by SparkFun Electronics, Boulder, USA

.NET (C#, VB, C++, F#, Python, PHP, JavaScript…) → FPGA logic

The benefits of FPGAs for software developers

- Performance increase for parallel compute-bound algorithms
- Higher power efficiency
- Still only software development

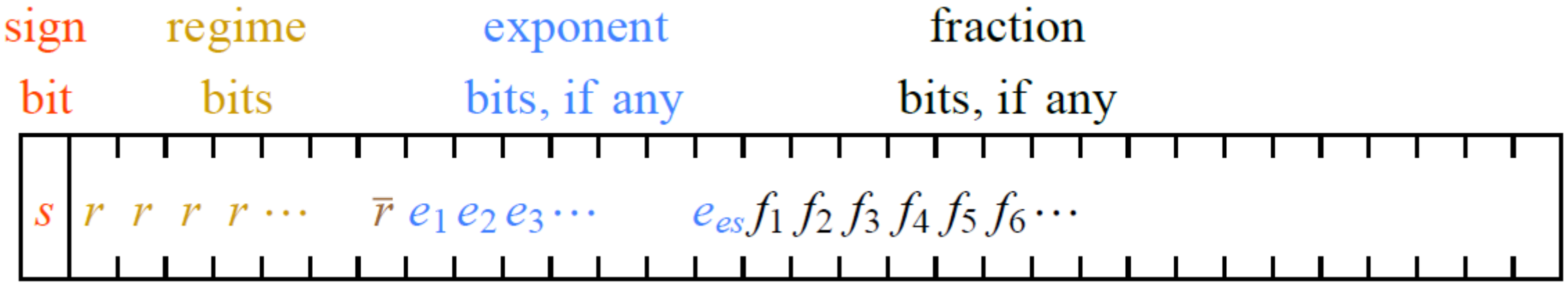
Demo: Hands-on Hastlayer

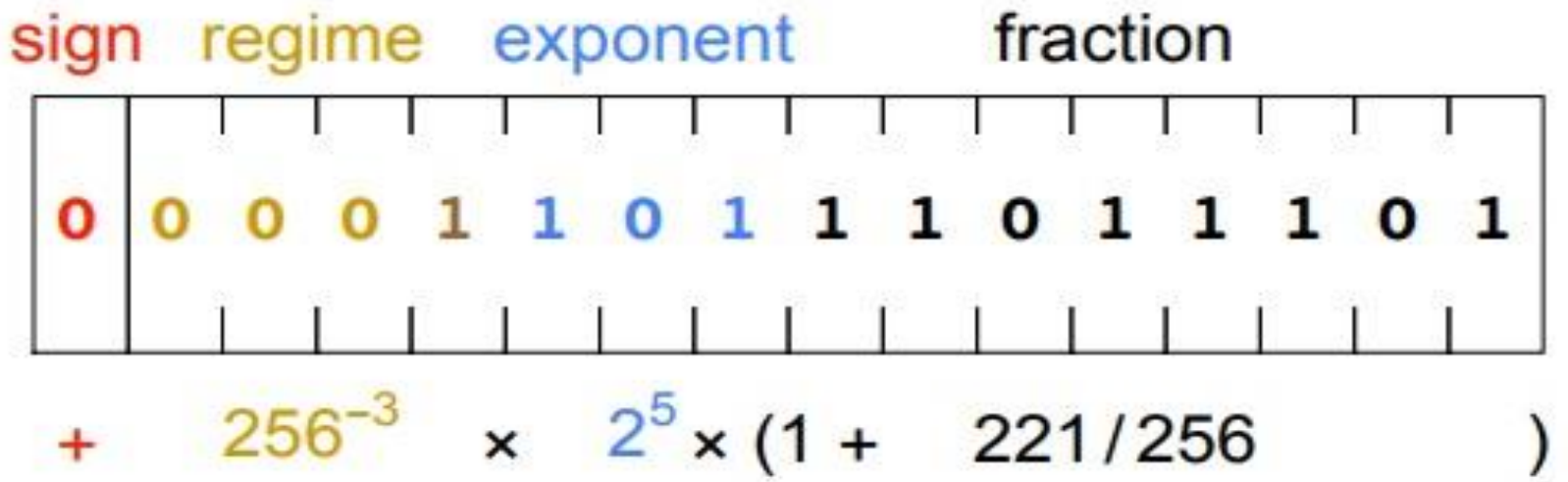
Next-Generation Arithmetics

Problems with IEEE floats

- Wasteful – over 16 million bit patterns mean NaN
- Special cases require complex logic
- Value distribution is not very user-friendly
- Different results on different computers

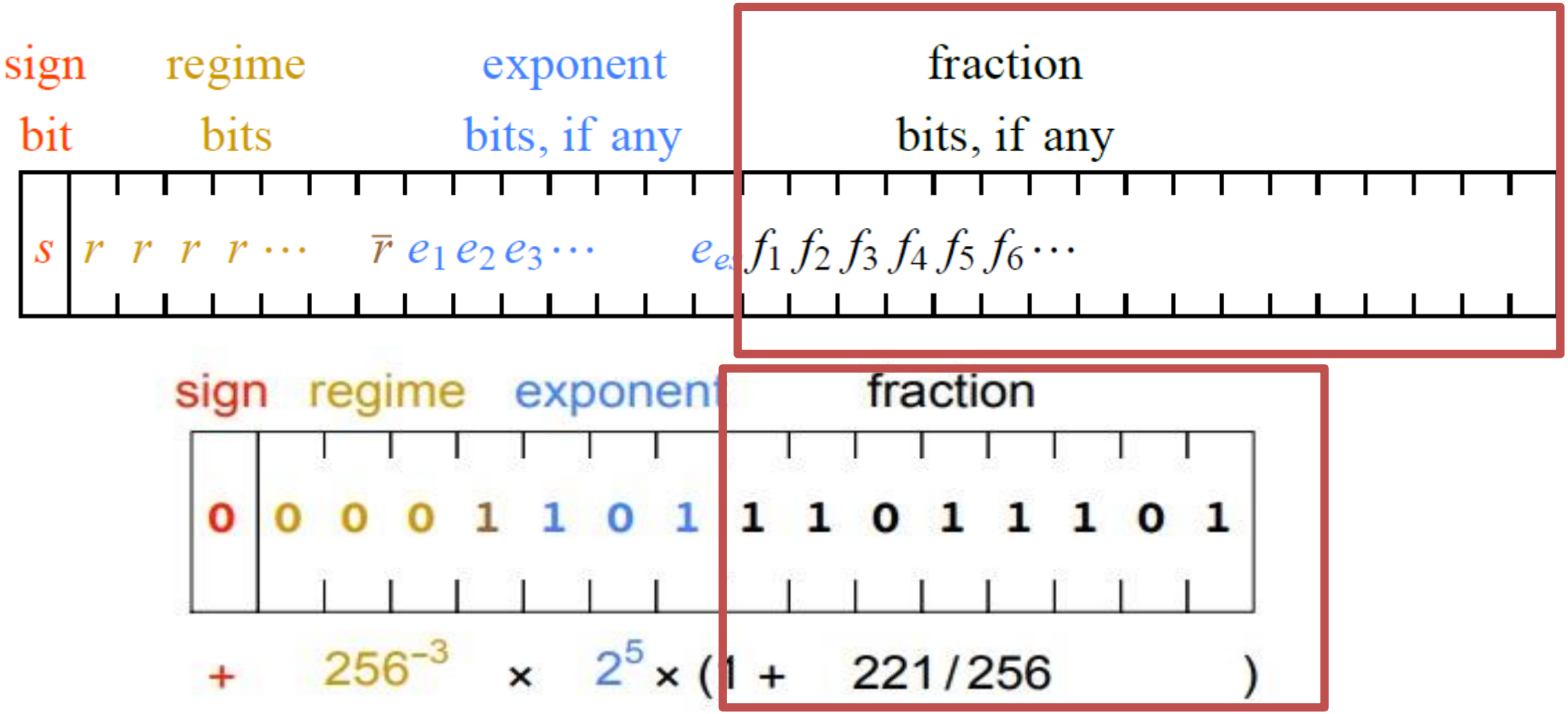
How does a posit look like?





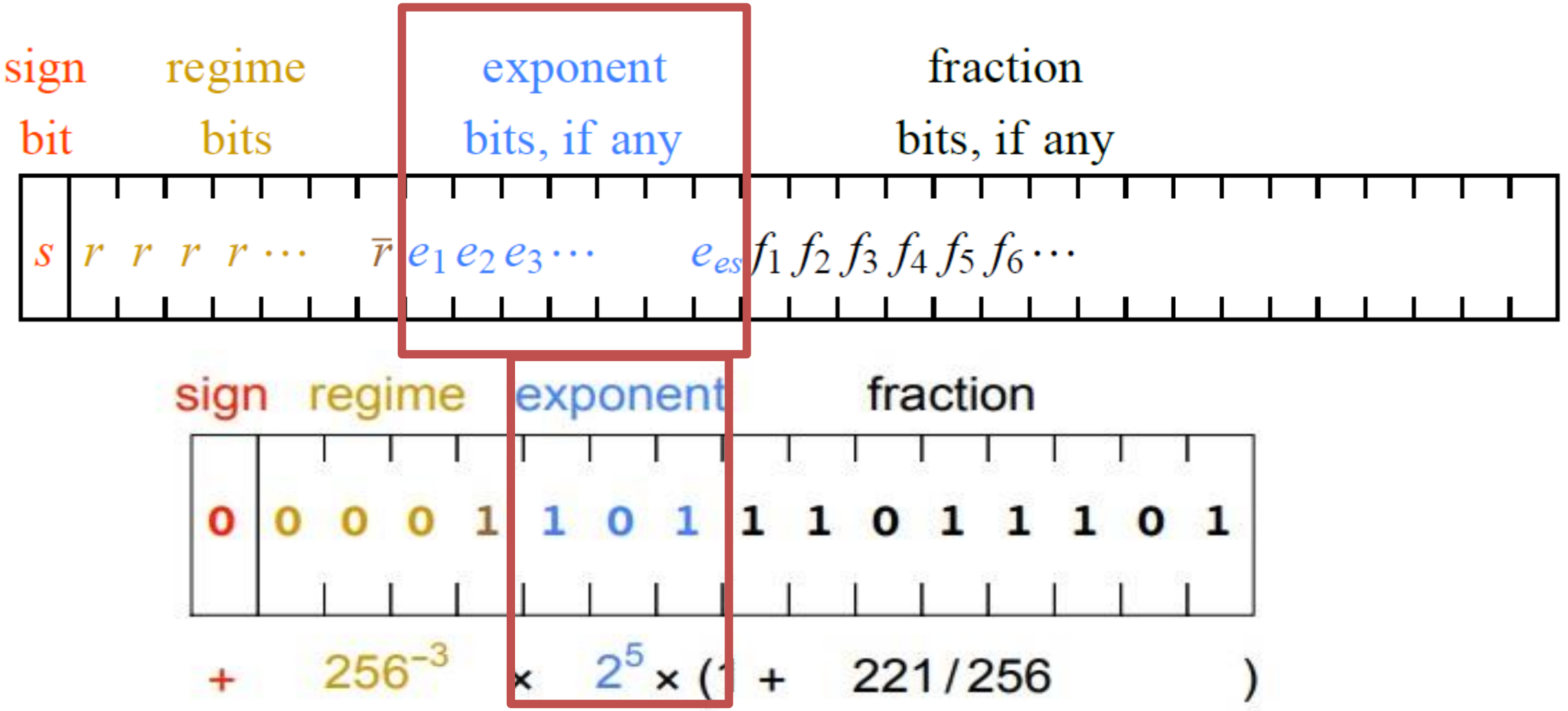
Source: posithub.org

How does a posit look like?



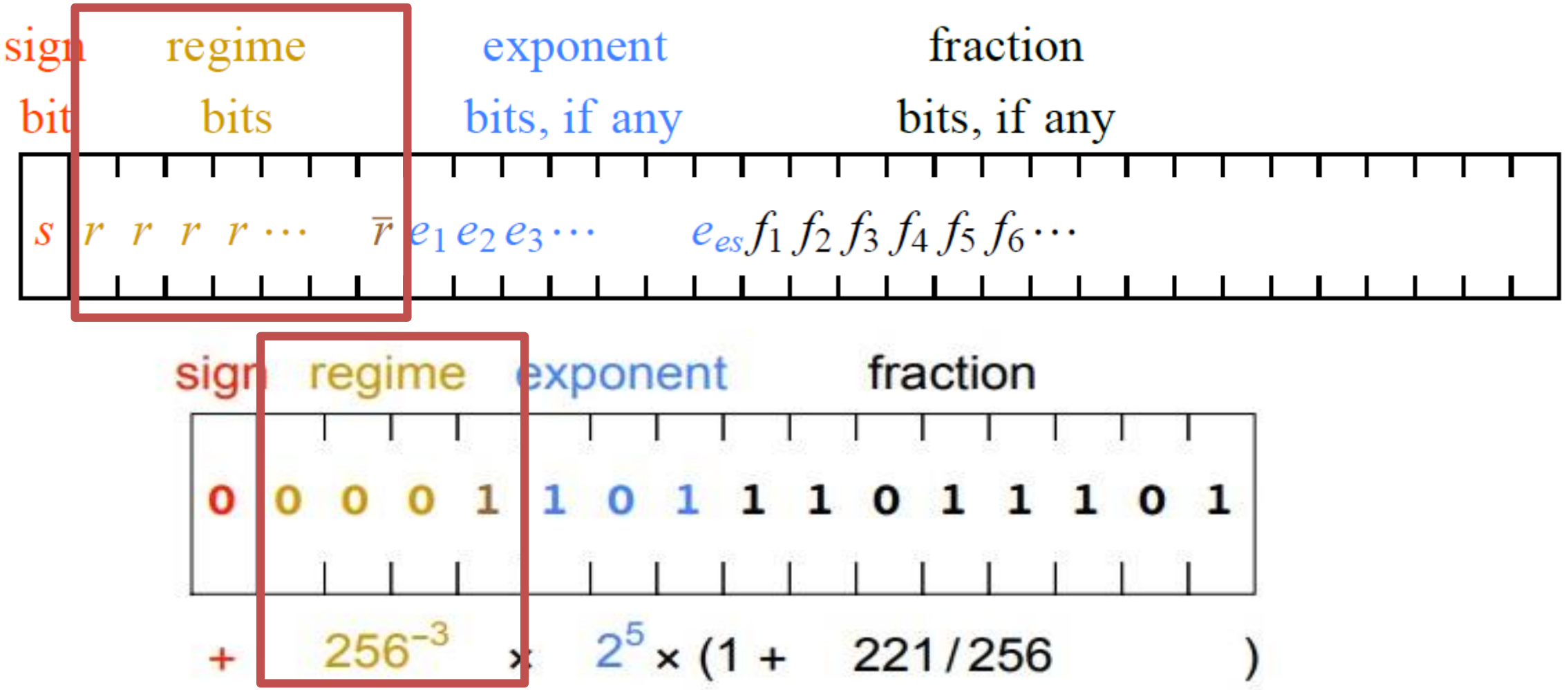
Source: posithub.org

How does a posit look like?



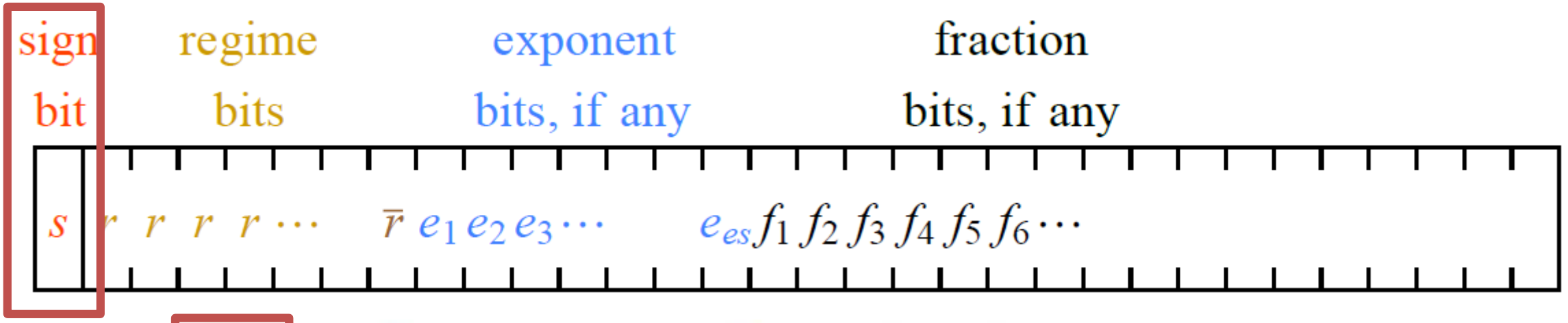
Source: posithub.org

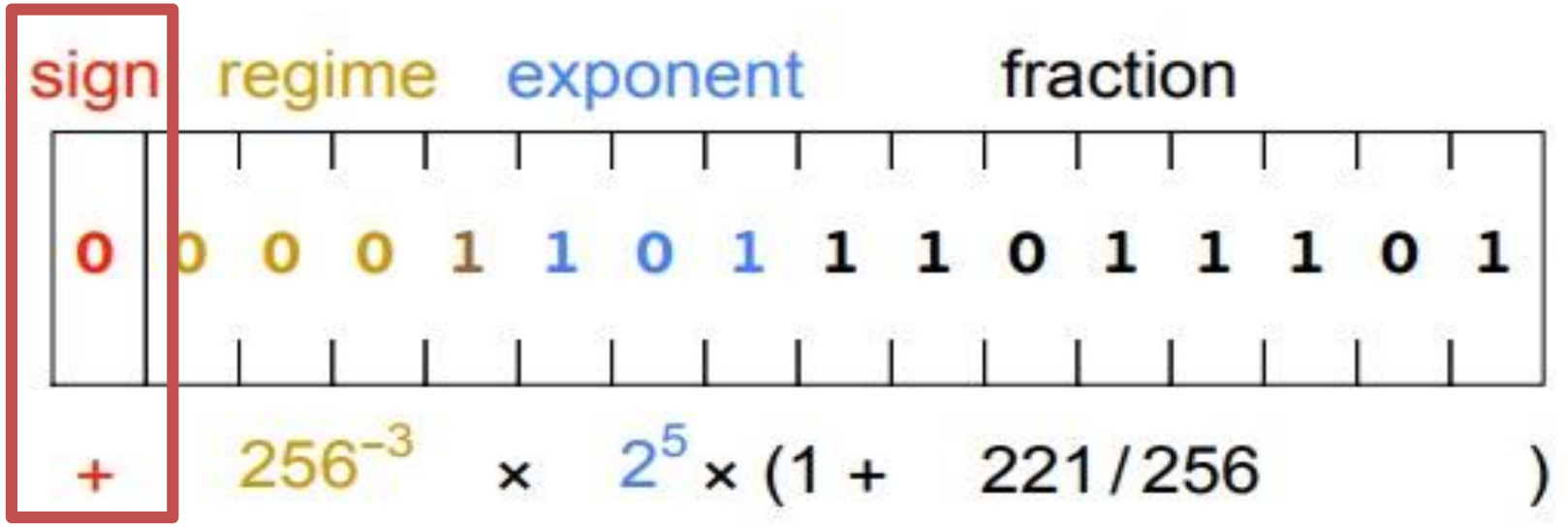
How does a posit look like?



Source: posithub.org

How does a posit look like?

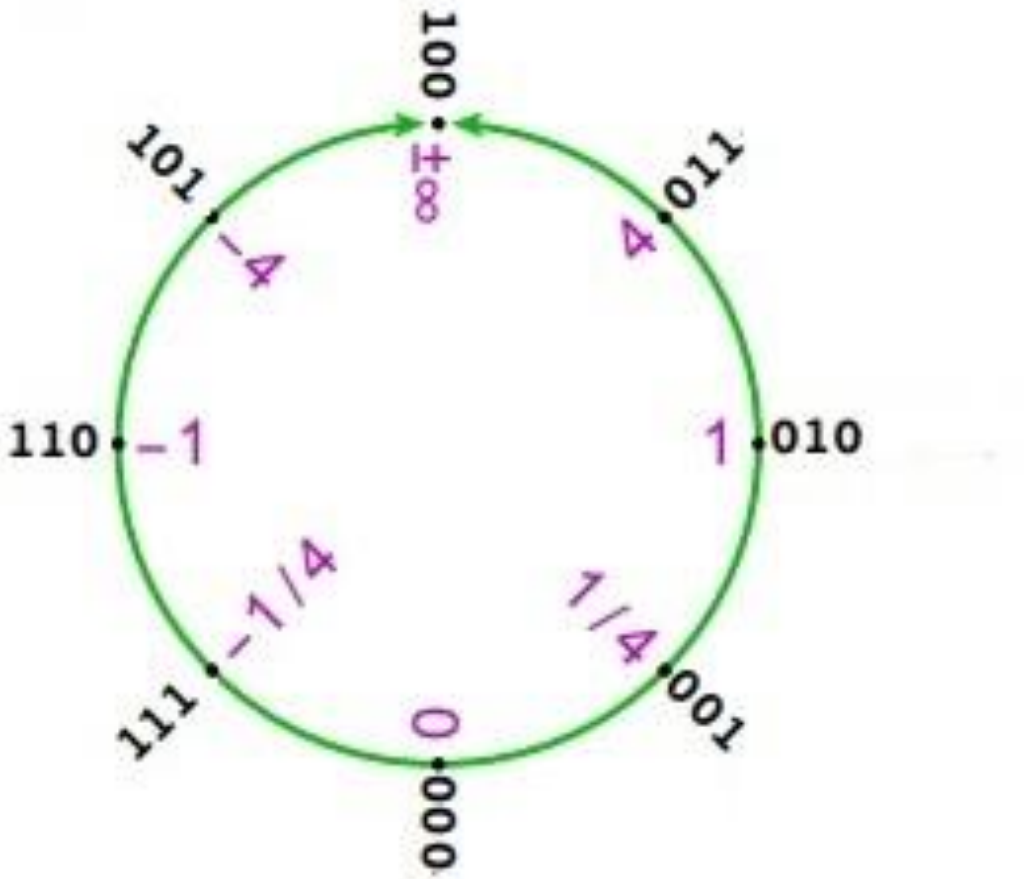




= 0.00000355392

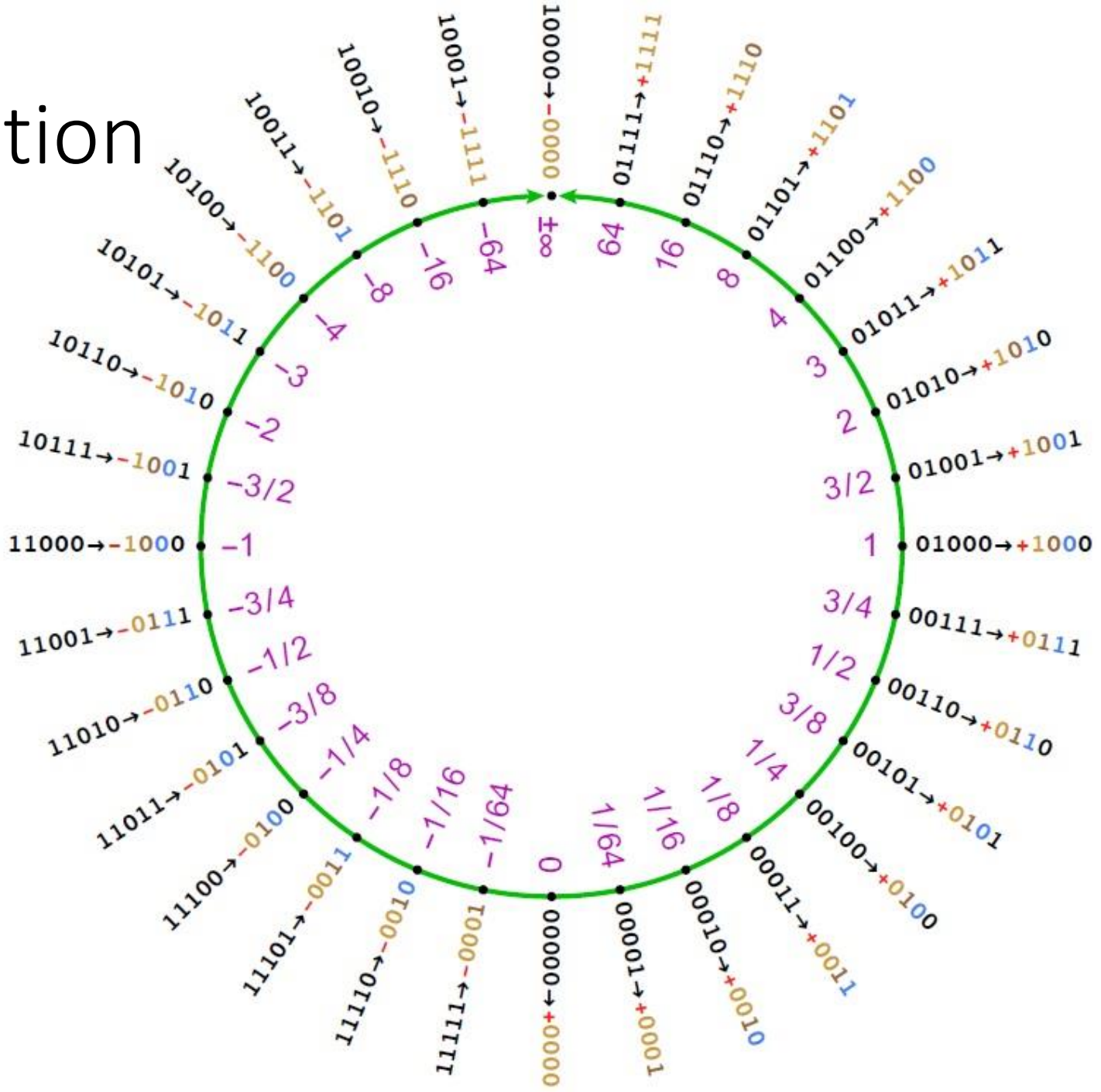
Source: posithub.org

Posit construction



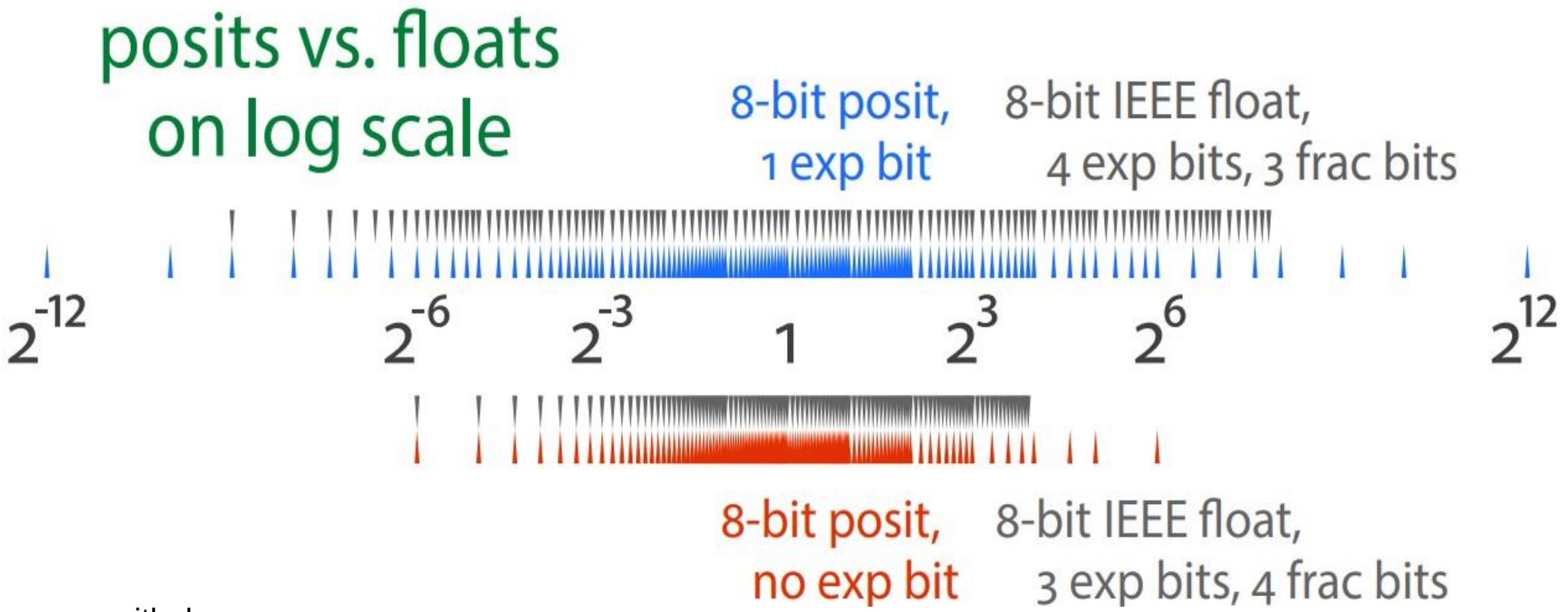
Source: posithub.org

Posit construction



Source: posithub.org

Distribution of values – posits vs floats



Source: posithub.org

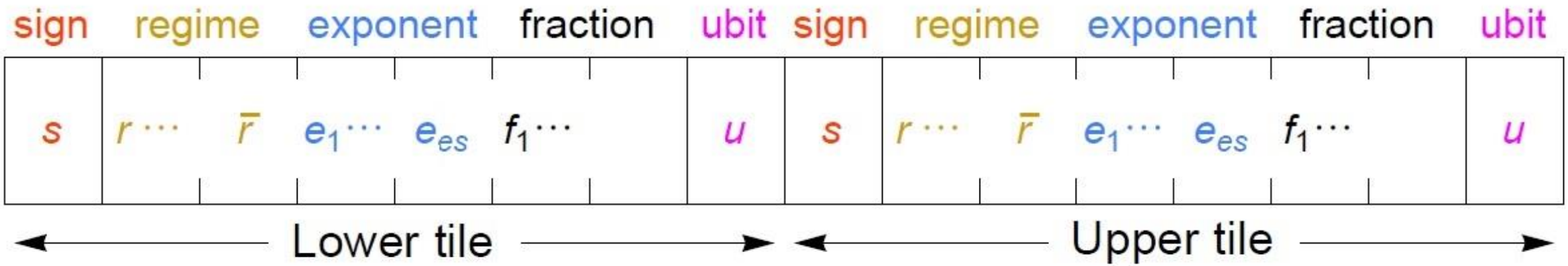
The Quire

- An internal scratchpad for computations required by the standard
- Rounding can be deferred to the last step after multiple operations
- Makes results consistent

Posit positives

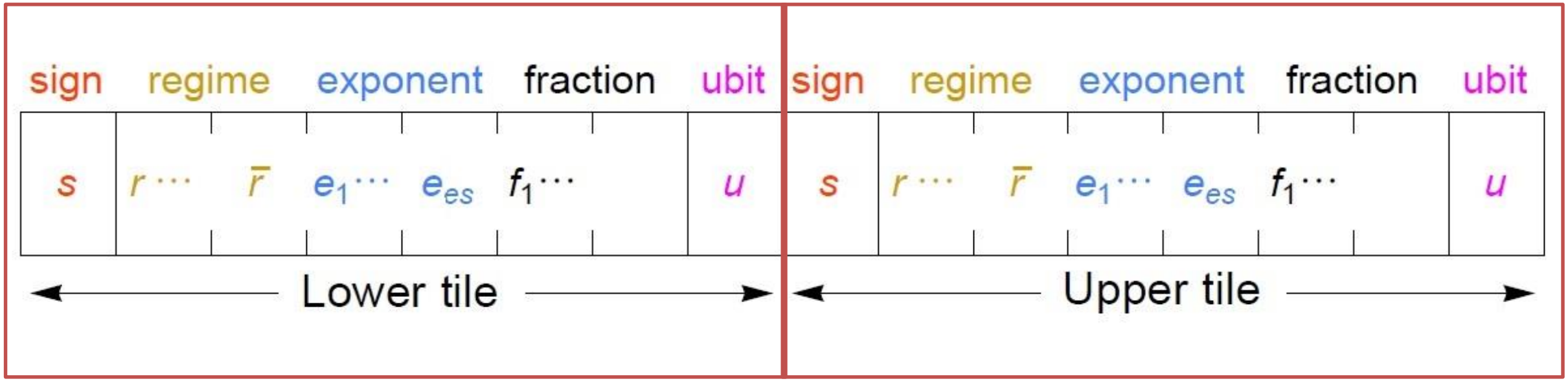
- Fewer bits to store the same information
  - Also affects energy efficiency
- Each value uses the same logic → simpler hardware
- Consistent results across platforms

Valids – interval arithmetic



Source: posithub.org

Valids – interval arithmetic



Source: posithub.org

Where can I get it and use it?

- FOSS implementations in multiple languages
- .NET C# implementation by Lombiq
- Transformed to FPGA using Hastlayer

Our posit implementation based on built-in .NET types

- 32-bit version based on unsigned integers
- Addition, subtraction, multiplication
- Quire
- Fused operations

Performance measurements I.

100 000 additions

	CPU	FPGA
32bit posit	13 ms	203 ms

- CPU: 7.69 MPOPS
- FPGA: 0.49 MPOPS

Performance measurements I.

Clock cycles per addition

	CPU	**FPGA**
32bit posit	416	203

Parallelized performance I.

500 000 additions in parallel (5 threads)

	CPU	FPGA
32bit posit	25 ms	203 ms

- CPU: 23.8 MPOPS
- FPGA: 2.46 MPOPS

Parallelized performance II.

Clock cycles per addition

	CPU	FPGA
32bit posit	160	40.6

Next steps

- More operations (division, exponentiation, trigonometric functions, fused operations, etc.)
- 8, 16, 64 bit versions with code generation
- Valids

Demo: 32-bit posits in action

Thank you for your attention!

- crew@hastlayer.com
- https://hastlayer.com
- https://hastlayer.com/arithmetics
- https://github.com/Lombiq/Hastlayer-SDK
- https://posithub.org