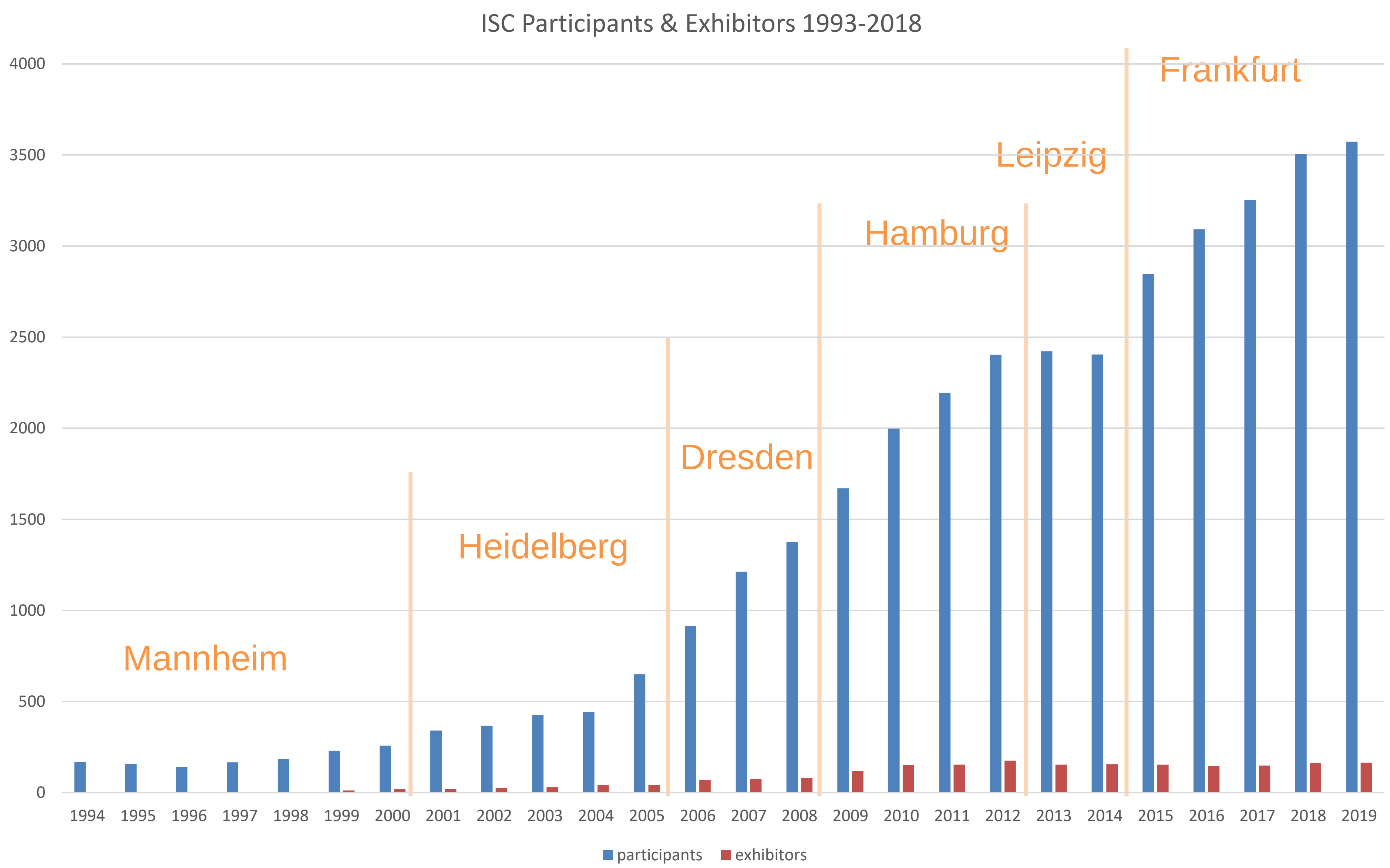
Highlights of the International Supercomputing Conference (ISC) 2019

Michael Fink ZID

Obergurgl Feb 2020

International Supercomputing Conference - Attendance History



ISC 2019 Topics

Top 500

Hardware - Post Moore Heterogeneity, New Architectures (ARM), GPU Computing

Middleware - OpenMP 5.0, MPI and Hybrid - Consolidating Multiple Parallelization Methods

Cluster Architectures - Networks and Interconnects, Exascale, Energy Efficiency

Compute Environments - Containers, Clouds & Virtualization in HPC

HPC, AI and Deep Learning - Integration, Neuromorphic Computing

Architectural Challenges - Data Movement, Memory, Storage

Numerical Representations - Mixed + Low precision, Number formats beyond IEEE

Software - Compilers (LLVM), Software Deployment and Execution Environments

Quantum Computing - Applications

Performance Tuning + Modelling, Numerical Reproducibility at Exascale

TOP 500 - 53rd + 54th Editions

TOP 500 - top500.org

- Collection of leading 500 HPC installations
- Released 2x / yr → 27 years of history
- Benchmark: HPL (High Performance Linpack)
  Solve dense $N \times N$ system of linear equations $Ax = b$ with random coefficients A in time t_{solve}:
  Number of Floating Point Operations $Flop = \frac{2}{3}N^3 + 2N^2$
  Rate = $\frac{Flop}{10^9 t_{solve}} \left[\frac{GFlop}{s}\right]$ reported for arbitrarily selected problem size
- Exponential growth since beginning (Moore)
- Upper limit of practically achievable performance
  - Many other algorithms slower by factor 10...100

- Criticism
  - HPL not relevant for real life performance - leads to suboptimal procurement decisions
    ⇒ use HPCG as complementary benchmark
  - self - reported: multiple entries for identical machines + omissions of important installations

  BUT:
  - Valuable overview of dominant HPC technology: Architecture: Vector → SIMD → SMP → Cluster → GPU
    Power efficiency
    Vendors & customers (countries, regions, segments) etc.
  - Only source of long-term HPC performance data

Dennard Scaling

Dennard (1974)

Power dissipation for MOSFETs:

$$P \propto f\, C\, U^2$$

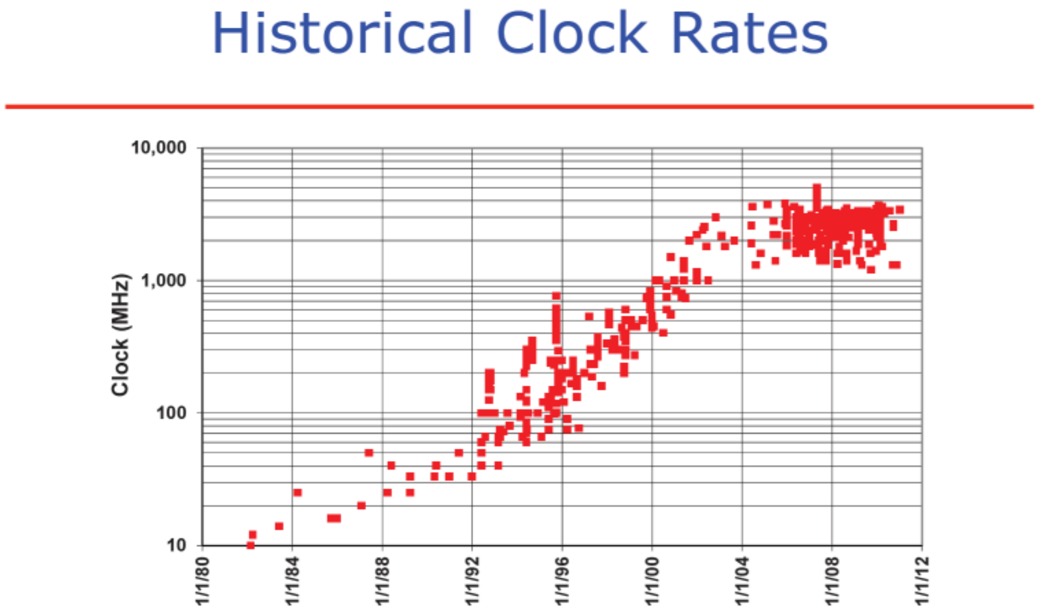
f	Frequency
C	Capacitance, proportional to area = (linear dimension)^2
U	Voltage

Dennard scaling: decrease voltage, shrink physical size
$\Rightarrow$ able to increase frequency at constant power density

Ignored: leakage currents and switching thresholds

Around 2006: leakage currents started to dominate
$\Rightarrow$ frequency stalled at <= 4 GHz

Further limitations: manufacturing tolerances at small scale
$\rightarrow$ error rates, inhomogeneous clock frequencies



Reproducibility - Conclusions

What can I do?

Results

- If possible, test if code: correctly renders known analytic results
  correctly reflects theoretical symmetries (e.g. geometric, gauge, scaling...)
  ...
- Assess sensitivity of your code to changes: mathematical model (e.g. truncation errors)
  numerical model (e.g. discretization errors - change grid)
  arithmetic (e.g. rounding modes - use fesetenv(3)
  FP Unit: -mfpmath=sse|387 - caution: false positives)
  parallelization (e.g. change domain boundaries or number of processes)
  optimization (e.g. -O2 vs. -O3)

Performance

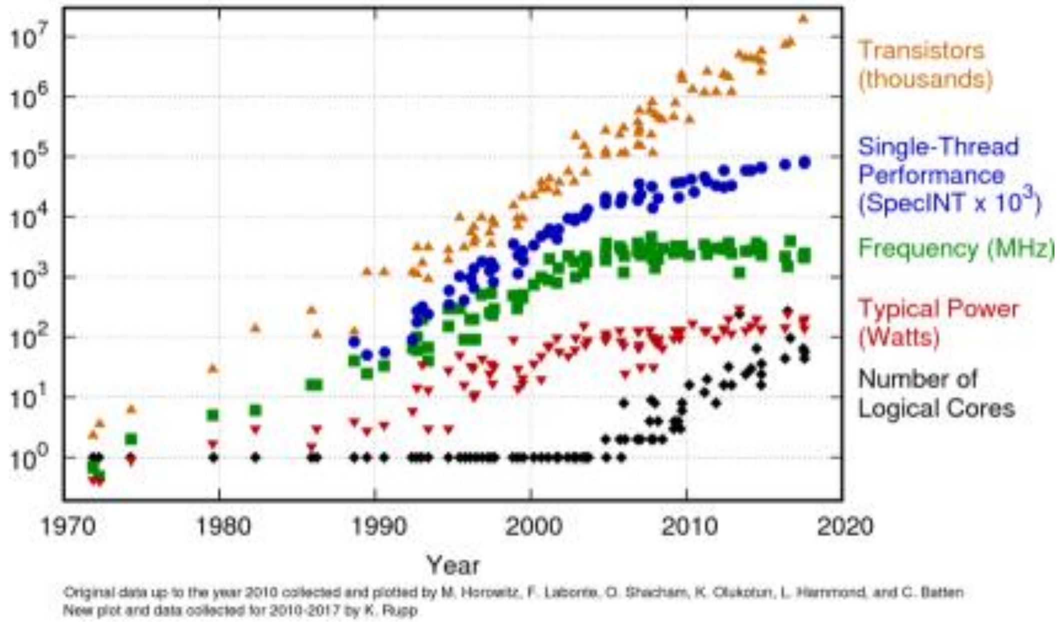
- Contact system administration early before making extensive measurements (e.g. reservation of nodes)
- Plan, control and report every aspect of your environment
- Make sure your experiments are not disturbed by other activities
- Monitor your programs during execution for unexpected patterns
- Use adequate statistics to assess quality of your measurements and to describe your results
- Read the Hoefler paper ☺

Mixed Precision and New Floating Point Formats

Moore's law ending

- Processor speed stagnates
- Data movement = primary bottleneck

} sacrifice accuracy for speed



AI / deep learning drives hardware diversity

GPU performance (e.g. Tesla V100)

- FP 64: 7.1 TFlops
- FP 32: 14.1 TFlops
- FP 16: 28.3 TFlops

For AI, low precision is often sufficient

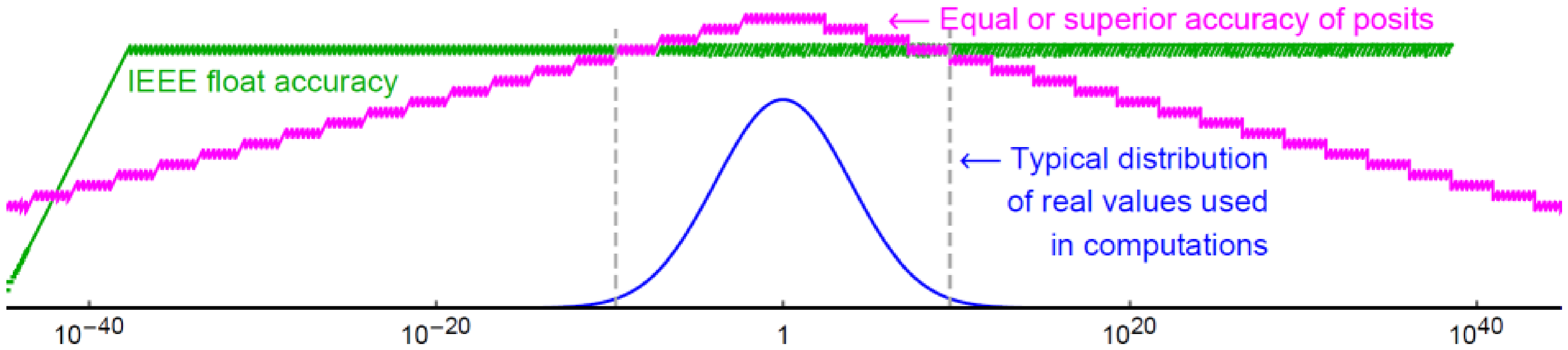
however: IEEE FP 16: insufficient dynamic range ⇒

- Google: bfloat (16 bit FP w/ IEEE FP 32 # exponent bits)
- Intel: flexpoint (scheme w/ 16 bit mantissa + 5 bit shared exponent) - specific for neural networks

- Gustafson: Posit number format plugin replacement for IEE
  tapered floats increased precision (middle of dynamic range)
  increased dynamic range - general purpose

Gustafson's Claims

New floating point format:



Posit

- Drop-in replacement for IEEE floating point
- Tapered accuracy division between exponent and fraction depends on exponent
  + higher accuracy than IEEE in range where needed
  + higher dynamic range
- Simpler, smaller, faster circuits (no software traps for exceptions)
- Matches what languages support (no hidden flags e.g. rounding direction, inexact, ...)
- Fast addition, simpler multiplication, no subnormals
- No overflow & underflow (gradual under/overflow, finally *maxpos* or *minpos* - *explicit handling needed*)
- Entire 2-complement integer range monotonically mapped to reals - integer comparisons possible
- Extension: exact sums and dot products ("*Quine*" = wide accumulator - based on ideas by Kulisch)
- Bitwise reproducible answers on all systems

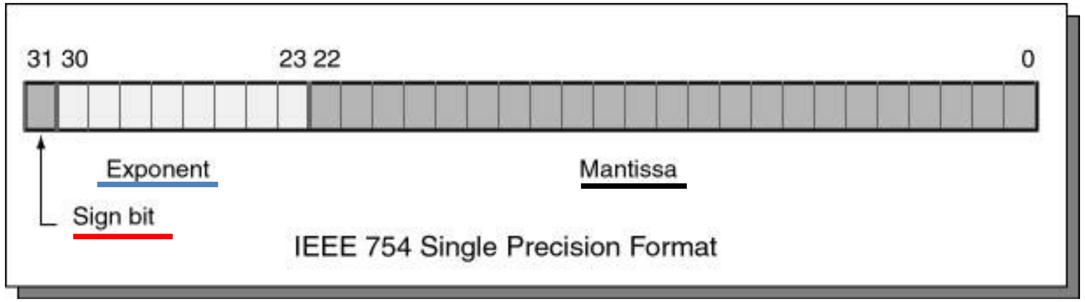
Note

- various precursors (Type I & II unums) with variable word sizes and lookup tables impractical and expensive - not feasible in hardware

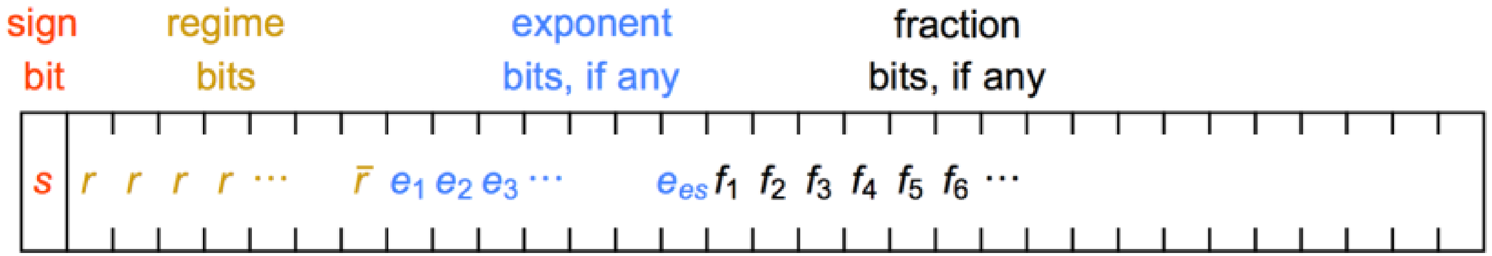
Posit Format

IEEE Format

- Fixed size mantissa (fraction) and exponent
- Biased Exponent (fixed *bias* for negative powers)
- Implicit (1.) in mantissa bits
- value: $(-1)^s \cdot (1.f_1 f_2 f_3 f_4 ...) \cdot 2^{exp-bias}$



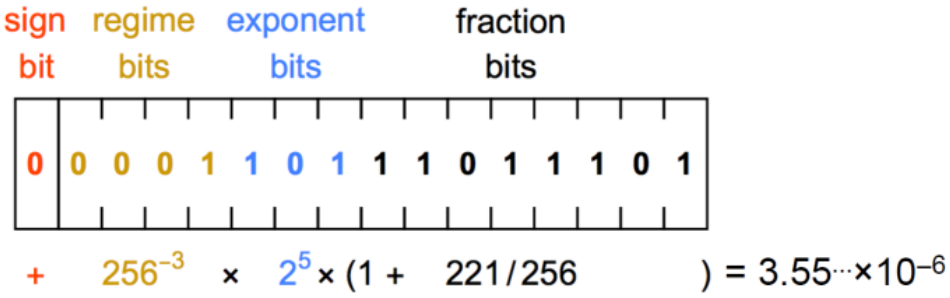
Posit Format: extension of IEEE



- Variable sized fraction and exponent
- Specified by length *n* and maximum exponent length *es*
- Regime bits: run-length *k* encoding variable exponent size (max: *es*) & scale factor $useed^k$ where $useed = 2^{2^{es}}$ $(|k| \leq n-1)$
- variable remaining bits: fraction
- value: $(-1)^s \cdot (1.f_1 f_2 f_3 f_4 ...) \cdot useed^k \cdot 2^{exp}$
- *exponent* bridges gaps between powers of *useed*

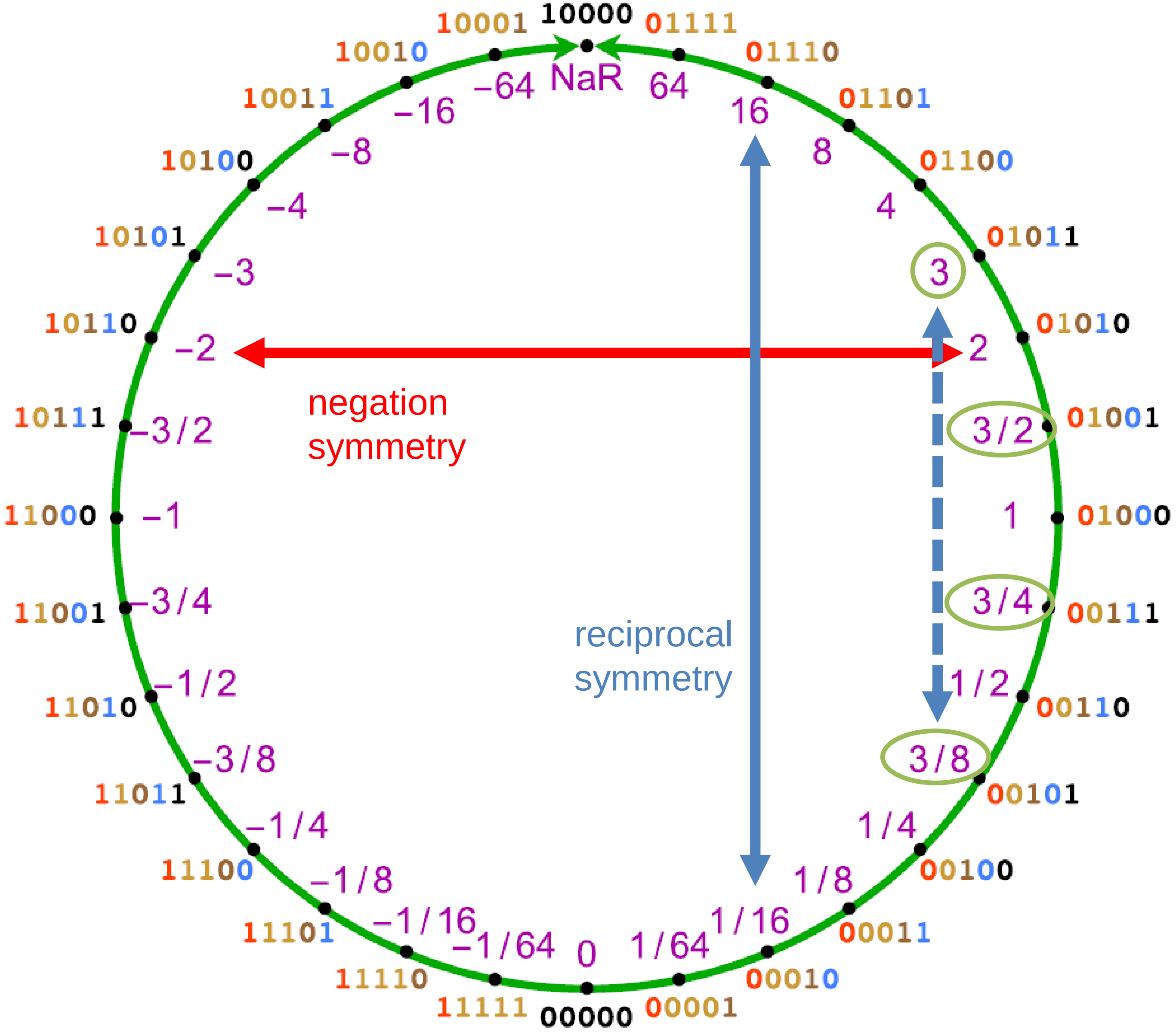
Posit 16 Bit Example

- *es* = 3



for details, see https://posithub.org/docs/BeatingFloatingPoint.pdf

Posits Map Projective Reals to 2’s Complement Integers. Example: 5 bit

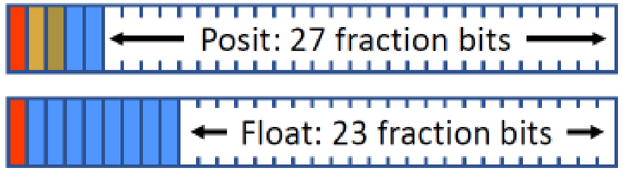


posits monotonically map
2’s complement integers to reals

Integer powers of 2, ±∞ , and 0
have exact inverses

Non-powers of 2: arithmetic mean of
neighboring numbers

Half of all posits have only two regime
bits, leaving more room for fraction
e.g. 32 bit:



Posits: Gustafson's Predictions & My Thoughts

Gustafson

- "Switching will take ten years. It always does"
- Dual mode IEEE+Posit processors, transition default IEEE → Posit; similar to ASCII → Unicode
- Switching floats to posits will be easier than sequential to parallel programming
- Posits will let HPC return to 32 bits as default real size → half cost of data movement
- Deep learning: 16 bits sufficient (8 bit for inference)

My thoughts

- Appears to be significant improvement - for problems that can be scaled to ≈ 1 + inexact workloads (e.g. AI)
- Standard error analysis based on constant relative accuracy over entire range
  - Tapered accuracy ⇒ complete re-assessment of numerical libraries needed
    papers on error analysis exist - old
- Error detection
  - What is worse: termination due to overflow vs. continuation w/ less accurate value
- Empirical error estimates of large codes
  - Verify results by repeated runs under different rounding modes
    (default: nearest & ties to even; towards +∞, -∞, zero)
  - Posits: method not available - only one rounding mode (same as IEEE default)
- Promising LLNL case study - will claims hold up to reality in general?
- Adoption depends on community & market mechanisms

ISC: Software Deployment Coverage

HPC Software deployment and execution environments

- Automated SW deployment in HPC clusters → Spack, Easybuild
- Reproducible and portable SW environments → containers — Singularity ↔ Docker; Sarus (ETHZ)
  Charliecloud, Shifter, Udocker...
- Executing containers - cloud integration in context of
  workload management in HPC clusters → Slurm + Singularity / Kubernetes + Pods } struggling for dominance in HPC — compete or combine?

  increasing support for interactive HPC
  underutilize & overcommit no longer taboo
- Filling containers with software → OS based installers + Spack

Spack talk AHPC'20