Today:

- floating point
- Numerical linear alg.

Announcements:

- HW0 due
- HW1 out tomorrow
- Quiz deadlines
- Exam lot 0 starts Thu

Wanted: Real Numbers... in a computer

Computers can represent *integers*, using bits:

$$23 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = (10111)_2$$

How would we represent fractions?

$23.625 = (\ldots) + .625$

$= (\ldots) + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3}$

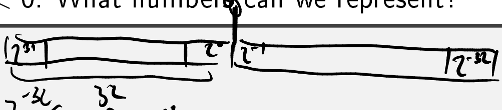
a fixed # of bits

allocate:

- # bits for integer part
- # bits for the fractional

Fixed-Point Numbers

Suppose we use units of 64 bits, with 32 bits for exponents $\geqslant 0$ and 32 bits for exponents < 0. What numbers can we represent?



$2^{32} - 2^{-32} \leftarrow$ largest

$2^{-32} \leftarrow$ smallest

How many 'digits' of relative accuracy (think relative rounding error) are available for the smallest vs. the largest number?

for largest: rel. error $\dfrac{2^{-32}}{2^{32}} \approx$ → 19 digits

smallest: $\dfrac{2^{-32}}{2^{-32}} \approx 1$ → 0 digits

Floating Point Numbers

$0.000\,0\,\underline{1}\,0\,0\,1$

1110.110

Convert $13 = (1101)_2$ into floating point representation.

$$13 = (1.101)_2 \cdot 2^{3}$$

What pieces do you need to store an FP number?

significand (fraction) sign bit

exponent

IEE 854

Floating Point: Implementation, Normalization

Previously: Consider *mathematical* view of FP.

Next: Consider *implementation* of FP in hardware.

Do you notice a source of inefficiency in our number representation?

$$\underbrace{3}_{\text{math. exponent}} = \underbrace{-1023}_{\text{implicit offset}} + \underbrace{1026}_{\text{actually stored exponent}}$$

Unrepresentable numbers?

Can you think of a somewhat central number that we cannot represent as

$$x = (1.\_\_\_\_\_\_\_\_\_\_\_)_2 \cdot 2^{-p}?$$

Hack: Label an exponent value special
namely -1023 (stored as 0)
↪ net effect: turn off the leading 1

Demo: Picking apart a floating point number

Subnormal Numbers

What is the smallest representable number in an FP system with 4 stored bits in the significand and an exponent range of $[-7, 7]$?

Subnormal Numbers II

What is the smallest representable number in an FP system with 4 stored bits in the significand and an exponent range of $[-7, 7]$? (Attempt 2)

Why learn about subnormals?

Underflow

- FP systems without subnormals will *underflow* (return 0) as soon as the exponent range is exhausted.
- This smallest representable *normal* number is called the *underflow level*, or *UFL*.
- Beyond the underflow level, subnormals provide for *gradual underflow* by 'keeping going' as long as there are bits in the significand, but it is important to note that subnormals don't have as many accurate digits as normal numbers.
- Analogously (but much more simply–no 'supernormals'): the overflow level, *OFL*.

Rounding Modes

$(1.1101010\overset{1}{0}11)_2$

How is rounding performed? (Imagine trying to represent π.)

- chop ('round to zero')
- round-to-nearest

$(1.1101011)_2$

What is done in case of a tie? $0.5 = (0.1)_2$ ("Nearest"?)

$(1.1101010\overset{1}{0}10)_2$

"round-to-even"

Demo: Density of Floating Point Numbers

Demo: Floating Point vs Program Logic

Smallest Numbers Above...

- What is smallest FP number > 1? Assume 4 bits in the significand.

$$(1.\underline{0}\,\underline{0}\,\underline{0}\,\underline{1})_2 \cdot 2^0 = \underline{1} \cdot (1+\varepsilon)$$

What's the smallest FP number > 1024 in that same system?

$$(1.\underline{0}\,\underline{0}\,\underline{0}\,\underline{1})_2 \cdot 2^{10} = 1024 \cdot (1+\varepsilon)$$

Can we give that number a name?

Unit Roundoff

Unit roundoff or *machine precision* or *machine epsilon* or $\varepsilon_{\text{mach}}$ is the smallest number such that

$$\text{float}(1+\varepsilon) > 1.$$

ignoring round-to-even

- Assuming round-to-nearest, in the above system, $\varepsilon_{\text{mach}} = (0.\underline{00001})_2$.
- Note the extra zero.
- Another, related, quantity is *ULP*, or *unit in the last place*. ($\varepsilon_{\text{mach}} = 0.5\,\text{ULP}$)

FP: Relative Rounding Error

What does this say about the relative error incurred in floating point calculations?

$$\frac{|x \cdot (1+\varepsilon_{mach}) - x|}{|x|} = \varepsilon_{mach}$$

FP: Machine Epsilon

What's that same number for double-precision floating point? (52 bits in the significand)

Demo: Floating Point and the Harmonic Series

Implementing Arithmetic

How is floating point addition implemented?
Consider adding $a = (1.101)_2 \cdot 2^1$ and $b = (1.001)_2 \cdot 2^{-1}$ in a system with three bits in the significand.