used in AI hardware along with 8-bit and even 4-bit...

IEEE 754

16-bit: Half (binary16)

32-bit: Single (binary32), decimal32

64-bit: Double (binary64), decimal64

128-bit: Quadruple (binary128), decimal128

256-bit: Octuple (binary256)

Extended precision

Standard for scientific computation.

x87
80-bit

traditionally implemented in hardware... since 70's.

Sometimes used to compute residual errors to higher precision in order to make a correction...

Software libraries for higher precision would need to multiply the mantissa's together for two numbers

2 4 5 6 8 n-digits

1 3 5 9 7 n-digits

$n=5$

Usual algorithm multiplies the digits together in pairs one red one blue... n^2 single digit mult. (25).

power causes lots of work for large n.

Karatsuba algorithm

Class Multiplication algorithm

Software for high precision computation...

$$n^{\log_2 3} \approx n^{1.58}$$

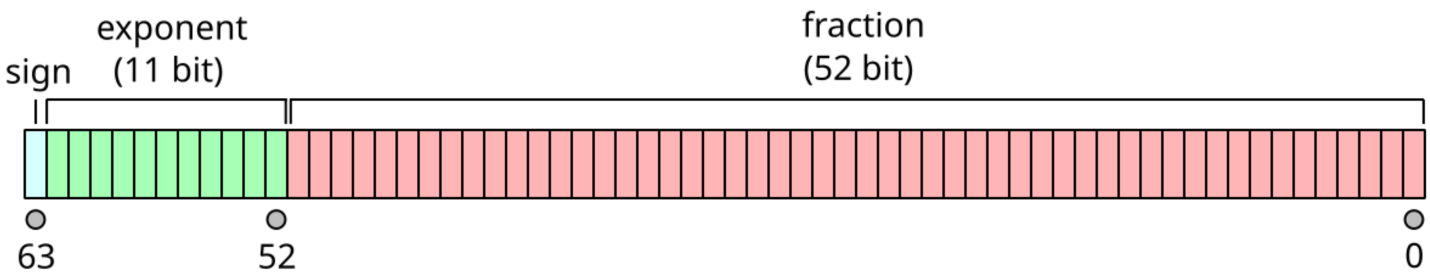
instead of n^2 it takes
$n^{1.58}$ single digit multiplications
to multiply two n-digit numbers...

The **Schönhage–Strassen algorithm** is an asymptotically fast multiplication algorithm for large integers, published by Arnold Schönhage and Volker Strassen in 1971.[1] It works by recursively

$$n \cdot \log n \cdot \log\log n$$

instead of n^2 or $n^{1.58\ldots}$
the work scales linearly
in the size of the numbers
with a logarithmic correction...

How much error is in the arithmetic.



way to store approximations to real numbers...

$$(-1)^{\text{sign}}(1.b_{51}b_{50}\ldots b_0)_2 \times 2^{e-1023}$$

since all floating point mantissas (other than the representation of zero itself) start with a 1 there is no need to store the first binary digit.

Mantissa is 2^{52} different things mathematically.

Precision is like a 53 bit mantissa.

$$2^{52} \approx 10^x$$

solve for x

$$52 \log 2 = x \log 10$$

$$x = \frac{52 \log 2}{\log 10}$$

```
julia> 52*log(2)/log(10)
15.653559774527022
```

double precision arithmetic has about 15 decimal digits of precision...

This is just for intuition...

Solving $Ax = b$

Suppose x^* is the best floating point approximation of x.

Means that the x^* is the closest value to x.

Example.

$\overbrace{315}^{3+1}.2854$ round to 4 digits

$\underbrace{315.2}854$

315.3

$3+2$ is one more

bound on the error $\pm 5 \times 10^{-2}$

$\underbrace{3.152}854 \times 10^{2}$

3.152×10^{2}

max error $= 5 \times 10^{-4} \cdot 10^{2} = 5 \times 10^{-2}$

$x \in \mathbb{R}$ round to 15 digits

$x = d.ddd\ldots. \times 10^{e}$

max error $\pm 5 \times 10^{-15} \cdot 10^{e} = 5 \times 10^{e-15}$

relative error divide by x

$$\frac{\text{max error}}{x} \quad \pm \frac{5\times 10^{-15}\cdot \cancel{10^{e}}}{d.ddd \times \cancel{10^{e}}} = \frac{5\times 10^{e-15}}{d.ddd \times \cancel{10^{e}}} = \frac{5}{d.ddd\ldots} 10^{-15}$$

$$\frac{1}{2}\times 10^{-15} \leq \max \frac{|x-x^*|}{|x|} \leq 5\times 10^{-15}$$

Solving $Ax=b$ Even if I knew the exact answer the best I could store it is with relative error about 10^{-15} in each component of the vector.

So if x^* is the closest (by rounding) to the exact answer it could be off by about 10^{-15}.

$$\frac{\|x-x^*\|}{\|x\|} \approx 10^{-15}$$

$$\frac{\|x - x^*\|}{\|x\|} \le \kappa(A) \frac{\|b - b^*\|}{\|b\|}$$

The best this could be is 10^{-15}

factor might also be a little wrong. 10^k

$$\frac{\|x - x^*\|}{\|x\|} \le 10^k \cdot 10^{-15} = 10^{k-15}$$

If $\kappa(A) \approx 10^k$ then we can only see $15-k$ significant digits of x using residual error...