IEEE standard: extended precision

- It is a format that offers just a little extra precision and exponent
- Motivation for extended precision: some operations benefit from using more digits internally
- Example: some calculators display 10 digits but use 13 internally for calculation
- Example: binary-decimal conversion
  Think about writing/reading numbers to/from files
  When writing a binary number to a decimal number and read it back, can we get the same binary number?
Writing 9 digits is enough for short
Though $10^8 > 2^{24}$, 8 digits are not enough (details not discussed)

From Section 2.1.2 of Goldberg [1990], in reading the 9-digit number, if extended precision is available, an efficient algorithm can be done so that the original binary representation is recovered (details not shown)

17 digits for double precision (proof not provided).

Example:
numbers in a data set from Matrix market:
IEEE standard: extended precision III

> tail s1rmq4m1.dat

8.2511736085618438E+01  2.5134528659924950E+01
-6.0042951255041466E+00  8.6599442206615524E+04
1.0026197619563723E+01  -1.3136837661844502E+04
-1.5108331040361231E+01  5.1423173996955084E+04
-1.1690286345961363E+03  1.6250726655807816E+03
8.2511736074473220E+01  1.5108331040361227E+01

- Matrix market:

http://math.nist.gov/MatrixMarket/

A collection of matrix data
IEEE standard: exactly rounded operations

- Operations: IEEE standard requires results of addition, subtraction, multiplication and division exactly rounded.
- Exactly rounded: an array of words or floating-point numbers, expensive
- Goldberg [1990] showed using 2 guard digits and one sticky bit the result is the same as using exactly rounded operations
- Only little more cost
IEEE standard: exactly rounded operations II

- Reasons to specify operations run on different machines ⇒ results the same
- IEEE: square root, remainder, conversion between integer and floating-point, internal formats and decimal are correctly rounded (i.e. exactly rounded operations)
- IEEE does not require transcendental functions to be exactly rounded
- Transcendental numbers:
IEEE standard: exactly rounded operations III

- e.g., exp, log

  - Reason: cannot specify the precision because they are arbitrarily long
On some computers (e.g., IBM 370) every bit pattern is a valid floating-point number.

For IBM 370, $\sqrt{-4} = 2$ and it prints an error message.

IEEE: NaN, not a number.

Thus not every bit pattern is a valid number.

Special value of IEEE:

$+0$, $-0$, denormalized numbers, $+\infty$, $-\infty$, NaNs

(more than one NaN).

A summary.
Special quantities II

<table>
<thead>
<tr>
<th>Exponent</th>
<th>significand</th>
<th>represents</th>
</tr>
</thead>
<tbody>
<tr>
<td>$e = e_{\text{min}} - 1$</td>
<td>$f = 0$</td>
<td>$\pm 0, -0$</td>
</tr>
<tr>
<td>$e = e_{\text{min}} - 1$</td>
<td>$f \neq 0$</td>
<td>$0.f \times 2^{e_{\text{min}}}$</td>
</tr>
<tr>
<td>$e_{\text{min}} \leq e \leq e_{\text{max}}$</td>
<td>$f$</td>
<td>$1.f \times 2^e$</td>
</tr>
<tr>
<td>$e = e_{\text{max}} + 1$</td>
<td>$f = 0$</td>
<td>$\pm \infty$</td>
</tr>
<tr>
<td>$e = e_{\text{max}} + 1$</td>
<td>$f \neq 0$</td>
<td>NaN</td>
</tr>
</tbody>
</table>

- Why IEEE has NaN

  Sometimes even $0/0$ occurs, the program can continue

- Example: find $f(x) = 0$, try different $x$’s, even $0/0$ happens, other values may be ok.
Special quantities III

- If \( b^2 - 4ac < 0 \)

\[
\frac{-b + \sqrt{b^2 - 4ac}}{2a}
\]

returns NaN

- \(-b + \text{NaN}\) should be NaN

In general when a NaN is in an operation, result is NaN

- Examples producing NaN:
Special quantities IV

<table>
<thead>
<tr>
<th>Operation</th>
<th>NaN by</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>$\infty + (-\infty)$</td>
</tr>
<tr>
<td>$\times$</td>
<td>$0 \times \infty$</td>
</tr>
<tr>
<td>$/$</td>
<td>$0/0, \infty/\infty$</td>
</tr>
<tr>
<td>REM</td>
<td>$x \text{ REM } 0, \infty \text{ REM } y$</td>
</tr>
<tr>
<td>$\sqrt{}$</td>
<td>$\sqrt{x}$ when $x &lt; 0$</td>
</tr>
</tbody>
</table>
\[ \beta = 10, \ p = 3, \ e_{\text{max}} = 98, \ x = 3 \times 10^{70}, \]
x\(^2\) overflow and replaced by \(9.99 \times 10^{98}\)??
In IEEE, the result is \(\infty\)

- Note \(0/0 = \text{NaN}, \ 1/0 = \infty, \ -1/0 = -\infty\)

\[ \Rightarrow \text{nonzero divided by 0 is } \infty \text{ or } -\infty \]
Similarly, \(-10/0 = -\infty\), and \(-10/-0 = +\infty\)
(\(\pm 0\) will be explained later)

- \(3/\infty = 0, \ 4 - \infty = -\infty, \ \sqrt{\infty} = \infty\)

- How to know the result?
  replace \(\infty\) with \(x\), let \(x \to \infty\)
Example:

$$\frac{3}{\infty} : \lim_{x \to \infty} \frac{3}{x} = 0$$

If limit does not exist ⇒ NaN

- $\frac{x}{(x^2 + 1)}$ vs $\frac{1}{(x + x^{-1})}$

  - $\frac{x}{(x^2 + 1)}$: if $x$ is large, $x^2$ overflow, $\frac{x}{\infty} = 0$ but not $\frac{1}{x}$.
  - $\frac{1}{(x + x^{-1})}$: $x$ large, $\frac{1}{x}$ ok

  - $\frac{1}{(x + x^{-1})}$ looks better but what about $x = 0$?

  - $x = 0$, $\frac{1}{(0 + 0^{-1})} = \frac{1}{(0 + \infty)} = \frac{1}{\infty} = 0$
If no infinity arithmetic, an extra instruction is needed to test if $x = 0$. This may interrupt the pipeline.