Real Numbers and Floating-Point Arithmetic

A real consists of an integer and a fractional part. The number of digits employed for the fractional part determines the precision, while the number of digits of the integer part determines the range of the representation. The transcendental numbers such as

- Integers work for many, but not all situations
- Often need to represent real numbers
  - $\pi = 3.14159265…$
  - $e = 2.71828…$
  - 0.0000000031

- An alternate way to write numbers is in scientific notation where the number is normalized
  - $3.1 \times 10^{-9}$
- Several advantages
  - Simplifies exchange since has common form
  - Increases accuracy since unnecessary 0’s are replaced

Ex:

$0.1 \times 10^{-5}, 10.4 \times 10^{-4}$ are not normalized.

Similarly, we can represent binary numbers in normalized form:

$(0.1)_2 \equiv (1.0)_2 \times 2^{-1}$ (normalized)

Computer arithmetic that supports such numbers is called floating point.

So, in binary:

$(1.xxxxxxxxx)_2 \times 2^{yyyy}$

The representation of real numbers with a pre-determined number of integer and fractional digits is called the fixed point notation (FixP).

Example: The real number $\pi$ can be represented in a 16-bit register using 8-bit for the integer, and the remaining 8-bit for the fractional part (8.8-bit FixP) in the following form:
The disadvantage of the **FixP notation** is the loss of relative or percent resolution ($U\%$) in representing small numbers.

Example: Consider the numbers $A$= 152.005 and $B$=0.025 in 8.8-bit FixP.

$A_{(8.8\text{-bit FixP})} = 10011000.00000001_2$ and

$B_{(8.8\text{-bit FixP})} = 00000000.00000110_2$.

The absolute resolution or precision is in both cases

$U(A) = U(B) = 1/256 \approx 0.004$.

The percent resolution or precision in the representation of $A$ is

$U\% (A) = \frac{\text{absolute-precision}}{\text{significand-magnitude}} \times 100 \%$

$= (1/256) / 152.005 \times 100 \% \approx 0.0025 \%$.

However, for $B$ it is

$U\% (B) = (1/256) / 0.025 \times 100 \approx 15.6 \%$.

The loss of percent resolution of $B$ is a result of decreased number of significant bits in the register.

The **scientific notation** uses a fixed number of significant-figures (significant-digits) and it provides an almost constant percent resolution for a much larger range of numbers.
Example: Write the numbers $A = 152.005$ and $B = -0.025$ in the scientific notation with 6 significant digits and find the percent resolution for each case.

<<solution

$A = 1.52005 \times 10^2$, and $B = -2.50000 \times 10^{-2}$.

The percent resolution for the numbers $A$ and $B$ are:

$U\% (A) = 0.00001 / 1.52005 \times 100 = 0.0006 \%$

$U\% (B) = 0.00001 / 2.50000 \times 100 = 0.0004 \%$

>>

Floating point (FP) notation is the scientific notation of the binary numbers. The format of FP notation depends on the width and sign convention of the significand and exponent fields.

A floating point format is composed of three binary-fields:

1. a sign-bit field $S$,
2. a binary-exponent $E$, and
3. a binary-significand-fraction $F$.

The value of a non-zero number $R$ is:

$R = (-1)^S \times F \times 2^E$,

The leading bit of $F$ of a non-zero number is always one. (i.e., $1 \leq F < 2$).

Example Find the sign $S$, fraction $F$, and exponent $E$ of the binary numbers $A=152.005$ and $B=-0.025$. Use 16-bit for $F$, and 8-bit $E$ fields.

<<solution

$A = 152.005 = 10011000.00000001_2$,

$A = (-1)^0 \times 1.0011000000000001_2 \times 2^7$,

$S = 0$; (positive).

$F$ must have a leading-one, and a fraction.

$F = 1.001100000000001_2 \approx 152.005 / 2^7 \approx 1.187539...$;

In the binary form of $A$ the binary point is shifted right 7 bits to have a leading-one. Each shift is equivalent to dividing by 2, total effect is divide by $2^{-7} = 1/128$. This effect is compensated by the exponent term $2^7$, giving that

$E = 00000111_2 = 7$. 
B = -0.025 = -00000000.00000110 0110 0110 0110 0110 0110 0110 0110 2
B = (-1)^1 × 1. 1001 1001 1001 1001 1002 × 2^{-6},

S = 1; (negative).  E = -6 = 1111 10102.
F = 1.1001 1001 1001 1002 ≈ 0.025/2^{-6} ≈ 1.6

Example: A non-standard 24-bit floating point format has 1-bit sign S, 15-bit significand F, 8-bit exponent E in signed-binary notation, and represents \( r = (-1)^S \times F \times 2^E \). The significand F is normalized to have both a leading-one and a fraction.

Find the binary-FP form of \( A = 152.005 \):

\[ S = 0; \quad F = 1.0011 0000 0000 002; \quad \text{and} \quad E = 7 = 0000 01112. \]

\[ A_{(24-bit-FP)} = \begin{array}{cccccccccc}
S & F & E \\
0 & 100 & 1100 & 0000 & 0000 & 0000 & 01112.
\end{array} \]

In this example, the least significant figure of A is not fitting into the significand field of this FP format. Because of the loss of one bit the represented value is 152.

The largest positive number which can be stored in this 24-bit notation is obtained as follows:

\[ N_{\text{maximum}} = 1.1111 1111 1111 1112 \times 2^{011111112} \]
\[ = 2 \times 2^{127} - 1 \times 2^{127-15} \]
\[ \approx 2^{128} \]
\[ \approx 3.4 \times 10^{38} \]

IEEE-754 Standard of Floating Point Numbers

In IEEE-754 standard, the exponent field is written in biased-signed-number format, IEEE-754 defines two floating point notation for two different precision.
The \textit{single-precision-format} requires 32-bit, and it is shortly called \textit{float}. The single precision FP format has the following fields:

\begin{tabular}{|c|c|c|}
\hline
\textbf{b_{31}} & \text{1-bit} & \textbf{S} & \text{sign-bit (1:negative, 0:positive)} \\
\hline
\textbf{b_{23}}-\textbf{b_{30}} & \text{8-bit} & \textbf{Es} & \text{biased-exponent ( \( Es = E+127 \))} \\
\hline
\textbf{b_{0}}-\textbf{b_{22}} & \text{23-bit} & \textbf{Fs} & \text{binary-significand ( \( Fs = F-1 \))} \\
\hline
\end{tabular}

\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|}
\hline
\text{bits:} & 31 & 30 & 29 & 28 & 27 & 26 & 25 & 24 & 23 & 22 & 21 & 20 & \ldots & \ldots & 2 & 1 & 0 \\
\hline
\text{fields:} & \textbf{S} & \text{Biased-Exponent} & \textbf{Es} = E + 127 & \text{Binary-Significand} & \textbf{Fs} = F - 1 & \text{(8 bits)} & & & & & & & & & \text{(23 bits)} \\
\hline
\end{tabular}

where,
\begin{itemize}
\item \textbf{S} is set if the number is negative,
\item \textbf{Es} = \( E+127 \) is the \textit{biased-exponent} to represent the exponents from \(-126\) to \(128\) ,
\item ( \( Es=0 \) is reserved for number-zero).
\item \textbf{Fs} = \( F-1 \) is the binary-significand of the number, after the leading-one bit in the normalized fraction of the number is removed,
\end{itemize}

The value of the represented number is \( R = (-1)^{S} \times (Fs + 1) \times 2^{Es-127} \)

<table>
<thead>
<tr>
<th>IEEE-754 single-precision-FP-fields</th>
<th>value of the number</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{S} (1-bit), \textbf{Es} (8-bit), \textbf{Fs} (23-bit)</td>
<td>((-1)^{S} \times (Fs + 1) \times 2^{Es-127})</td>
</tr>
</tbody>
</table>

In IEEE-754 float format, \textbf{Es}=0 (or \( E= -127 \)) is reserved for a zero number, so that a number that contains zeros in all fields is encountered to be zero.

The smallest positive number \( R_{\text{min}} \) has \textbf{Es}=1 (\( E= -126 \)) and \( Fs=0 \) (\( F=1 \)). It corresponds to \( 1 \times 2^{-126} = 1.17 \times 10^{-38} \).
Any number under this value causes to an \textit{underflow} condition, because \( Es=0 \) is reserved to represent \( R=0 \).

The largest number \( R_{\text{max}} \) has \textbf{Es}=255, (or \( E= +128 \)) and \( Fs=1-2^{-23} \approx 1 \) (\( F\approx2 \)). It corresponds to \( 2 \times 2^{128} \approx 6.8 \times 10^{38} \).
Any number over this value causes to an \textit{overflow} condition, because 8-bit exponent field is not wide enough for \textbf{Es}.
Example: Write the numbers $A = 152.005$ and $B = -0.025$ in IEEE-754 float form.

$A = 10011000.0000001_2$, 
$A = -1^0 \times 1.00110000000001_2 \times 2^7$, 
$S = 0$; 
$F = 1.00110000000001_2 \approx 152.005/2^7 \approx 1.187539...$;
$Fs = .00110000000001_2 \approx F–1 = 0.187539...$.
$E = 7$ ; $Es = 7 + 127 = 128 + 6 = 1000 0110_2$ 
$|S| Es | Fs | A = 0 1000 0110 0011 0000 0000 0010 0000 000$

$B = -000000000.00000110 01100110 01100110_2$
$B = -1^1 \times 1.1001100110011001100_2 \times 2^{-6}$, 
$S = 1$; 
$F = 1.100110011001100_2 \approx 0.025/2^{-6} \approx 1.6$; 
$Fs = .100110011001100_2 \approx 1.6 - 1 \approx 0.6$; 
$E = -6$ ; $Es = -6 + 127 = 121 = 0111 1001_2$

$|S| Es | Fs | B = 1 0111 1001 100 1100 1100 1100 1100 1100$

**IEEE 754 double-precision-float format extends**

the precision to **53-bits** (16-significant-figures in decimal), and the range of the floating point numbers to 11-bit ($R_{max} \approx 10^{308}$).

From MSB to LSB, the fields of the double precision format are:

<table>
<thead>
<tr>
<th>W1: Most Significant Word</th>
<th>$b_{31}$</th>
<th>1-bit</th>
<th>$S$</th>
<th>sign-bit (0:positive, 1:negative)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{20} - b_{30}$</td>
<td>11-bit</td>
<td>$Ed$</td>
<td>biased Exponent ($Ed = E + 1023$)</td>
<td></td>
</tr>
<tr>
<td>$b_{0} - b_{19}$</td>
<td>52-bit</td>
<td>$Fd$</td>
<td>binary-significand ($Fd = F–1$)</td>
<td></td>
</tr>
</tbody>
</table>

| W0: L.S.Word | $b_{31} - b_{0}$ | 52-bit | $Fd$ | binary-significand ($Fd = F–1$)   |