Representation of Positive & Negative Integral and Real Values

• A representation for both positive and negative integral values is needed

• Objectives
  • Easy to create the negative of a value
  • Easy to perform arithmetic with both positive and negative values
  • Easy to convert to and from decimal

• A representation for real numbers is needed

• Objectives are similar
Difference Between Numbers Represented on Computers and in Mathematics

• Range
  • The scope of numbers from the smallest possible to the largest possible that can be represented

• Precision
  • The number of bits (digits) of accuracy available to approximate a real value

• Integral numbers in computers are limited in range
• Floating-point numbers in computers are limited in range and precision
Integral Number Representation

- Integers
  - Unsigned
  - Sign and magnitude
  - One’s-complement
  - Two’s-complement
  - Excess notation

- Range
**Unsigned**

- The simplest representation allows for only positive values

- There is no way to represent negative values
Sign and Magnitude

• Perhaps the next simplest representation has a sign bit followed by the value
  • Sign bit of 1 indicates a negative value
  • Sign bit of 0 indicates a positive value

• The MSB is the sign bit
  • Value = \(-1^{\text{Sign-bit}} \times \text{Magnitude}\)

• Difficult to perform arithmetic

• Two representations for zero
One’s-Complement

• Given a value, form its one’s-complement by inverting each of the bits
• The MSB will still be used to indicate a negative value
  • Sign bit of 1 indicates a negative value
  • Sign bit of 0 indicates a positive value

- Still difficult to perform arithmetic
- Still two representations for zero
Two’s-Complement

• Given a value, form its two’s-complement by inverting each of the bits and then adding one
  • Complement then increment
• The MSB will still be used to indicate a negative value
  • Sign bit of 1 indicates a negative value
  • Sign bit of 0 indicates a positive value

• Easy to perform arithmetic
  • Conventional addition works with positive and negative numbers
• Only one representation for zero
• One more negative number than positive number
  • Zero has a sign bit of 0
• Two’s-complement is the most common representation for signed integral numbers
Two’s-Complement and Our Adder (1 of 2)

• If we have an adder that can perform $A + B + \text{carryIn}$,
• Then, to perform addition, we set $\text{carryIn}$ to 0 and the adder will add $A$ and $B$

• However, if we have the option to complement either $A$ or $B$ and also the option to set the $\text{carryIn}$ to 1,
• Then, using two’s-complement representations, our adder will perform subtraction!
  • $A + \overline{B} + 1 = A - B$
  • $\overline{A} + B + 1 = B - A$
Two’s-Complement and Our Adder (2 of 2)

• Furthermore, if we have the option to complement either A or B and set the other to zero and also the option to set the carryIn to 1,
• Then, using two’s-complement representations, our adder will perform negation!
  • \(0 + \sim B + 1 = -B\)
  • \(\sim A + 0 + 1 = -A\)
Excess Notation

• Value = Representation - Bias
• For example, using 8 bits,
  • If the representation is $64_{10}$ with a bias of $64_{10}$, then the value is 0
  • If the representation is $65_{10}$ with a bias of $64_{10}$, then the value is $1_{10}$
  • If the representation is $63_{10}$ with a bias of $64_{10}$, then the value is $-1_{10}$

• Although not easy to perform arithmetic, allows the demarcation point between positive and negative numbers to be set
• Only one representation for zero
• Used within floating-point numbers
Range of Values Represented

- Assume 8-bit word size
- 256 different bit representations

<table>
<thead>
<tr>
<th>Representation</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsigned</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>One’s-complement</td>
<td>-127</td>
<td>127</td>
</tr>
<tr>
<td>Two’s-complement</td>
<td>-128</td>
<td>127</td>
</tr>
<tr>
<td>Excess Notation, Bias=$64_{10}$</td>
<td>-64</td>
<td>191</td>
</tr>
</tbody>
</table>
Floating-Point Number Representation

- $s$: sign bit (0 for positive, 1 for negative)
- $b$: base or radix of the representation
- $e$: exponent value (represented using excess notation with a bias)
- $p$: number of base-$b$ digits in the significand
- $f_k$: significand digits
- $x = -1^s \cdot b^e \cdot \left( \sum_{k=1}^{p} f_k \cdot b^{-k} \right)$,  
  $e_{\text{min}} \leq e \leq e_{\text{max}}$
Floating-Point Bit Configuration

• The sign bit is the MSB
• Followed by the exponent value
• The significand digits are in the LSBs
IEEE 754 Floating-Point

- Size = 32 bits (float), 64 bits (double)
- Radix = 2
- Sign bit field
- Exponent field = 8 bits (float), 11 bits (double)
- Fraction field = 23 bits (float), 52 bits (double)
- Bias = 127 (float), 1023 (double)
- Zero value representation has exponent field = 0, fraction field = 0
  - Can be positive or negative
Normalization

• A normalized number has $f_1 > 0$, if $x$ (i.e., the value) is not 0
• A subnormal (denormalized) number is non-zero, has $e = e_{\text{min}}$ and $f_1 = 0$
  • Exponent is -126 (float), -1022 (double)
• An unnormalized number is non-zero, has $e > e_{\text{min}}$ and $f_1 = 0$
• A subnormal number is too small to be normalized
• Hidden bit
  • For normalized numbers, there is an assumed single 1 bit to the left of the binary point
  • Gives one more significant bit
Special Values

• Infinities
  • Positive
  • Negative
  • $sign = 0$ for positive infinity, $1$ for negative infinity; $biased \ exponent = \text{all 1 bits}$; $fraction = \text{all 0 bits}$

• NaN’s
  • Quiet
  • Signaling
  • $sign = \text{either 0 or 1}$; $biased \ exponent = \text{all 1 bits}$; $fraction = \text{anything except all 0 bits (because all 0 bits represents infinity)}$
Range and Precision of Values Represented

<table>
<thead>
<tr>
<th>Representation</th>
<th>Closest to Zero</th>
<th>Furthest from Zero</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>±1.18 × 10^{-38}</td>
<td>±3.4 × 10^{38}</td>
<td>~7 decimal digits</td>
</tr>
<tr>
<td>double</td>
<td>±2.23 × 10^{-308}</td>
<td>±1.80 × 10^{308}</td>
<td>~15 decimal digits</td>
</tr>
</tbody>
</table>