Representing and Manipulating Information: Floating point normalized and denormalized numbers

Yipeng Huang

Rutgers University

February 27, 2024
Table of contents

Announcements
  Programming assignment 3

Floats: Overview

Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example

Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples

Floats: Special cases

Floats: Summary

Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
Programming assignment 3

1. Due Friday 3/8.
2. Get started early! Plenty of background already for Parts 1 through 3.
Table of contents

Announcements
  Programming assignment 3

Floats: Overview

Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example

Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples

Floats: Special cases

Floats: Summary

Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
Floating point numbers

Avogadro’s number
$+6.02214 \times 10^{23} \text{ mol}^{-1}$

Scientific notation
- sign
- mantissa or significand
- exponent
Floating point numbers

Before 1985

1. Many floating point systems.
2. Specialized machines such as Cray supercomputers.
3. Some machines with specialized floating point have had to be kept alive to support legacy software.

After 1985

2. A floating point standard designed for good numerical properties.
3. Found in almost every computer today, except for tiniest microcontrollers.

Recent

1. Need for both lower precision and higher range floating point numbers.
Floats and doubles

Single precision

31 30 23 22 0

s exp frac

Double precision

63 62 52 51 32

s exp frac (51:32)

31 0

frac (31:0)

Figure: The two standard formats for floating point data types. Image credit CS:APP
## Floats and doubles

<table>
<thead>
<tr>
<th>property</th>
<th>half*</th>
<th>float</th>
<th>double</th>
</tr>
</thead>
<tbody>
<tr>
<td>total bits</td>
<td>16</td>
<td>32</td>
<td>64</td>
</tr>
<tr>
<td>s bit</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>exp bits</td>
<td>5</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>frac bits</td>
<td>10</td>
<td>23</td>
<td>52</td>
</tr>
<tr>
<td>C printf() format specifier</td>
<td>None</td>
<td>&quot;%f&quot;</td>
<td>&quot;%lf&quot;</td>
</tr>
</tbody>
</table>

**Table:** Properties of floats and doubles
The IEEE 754 number line

Figure: Full picture of number line for floating point values. Image credit CS:APP

Figure: Zoomed in number line for floating point values. Image credit CS:APP
Different cases for floating point numbers

Value of the floating point number = \((-1)^s \times M \times 2^E\)

- **E** is encoded the exp field
- **M** is encoded the frac field

1. Normalized

   \(s \neq 0 \& \neq 255\)

2. Denormalized

   \(s 0 0 0 0 0 0 0\)

3a. Infinity

   \(s 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0\)

3b. NaN

   \(s 1 1 1 1 1 1 1 \neq 0\)

**Figure:** Different cases within a floating point format. Image credit CS:APP

Normalized and denormalized numbers

Two different cases we need to consider for the encoding of E, M
Table of contents

Announcements
  Programming assignment 3

Floats: Overview

Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example

Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples

Floats: Special cases

Floats: Summary

Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
Normalized: exp field

For normalized numbers, $0 < \exp < 2^k - 1$

- $\exp$ is a $k$-bit unsigned integer

**Bias**

- Need a bias to represent negative exponents
- $\text{bias} = 2^{k-1} - 1 < 2^{k-1} - 1 = \underbrace{2^{k-1}}_{k-1} - 1 = 1024 - 1 = 1023$
- $\text{bias}$ is the $k$-bit unsigned integer: 011..111

For normalized numbers, $E = \exp - \text{bias}$

In other words, $\exp = E + \text{bias}$

Table: Summary of normalized exp field

<table>
<thead>
<tr>
<th></th>
<th>property</th>
<th>float</th>
<th>double</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>8</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>bias</td>
<td>127</td>
<td>1023</td>
<td></td>
</tr>
<tr>
<td>smallest E (greatest precision)</td>
<td>-126</td>
<td>-1022</td>
<td></td>
</tr>
<tr>
<td>largest E (greatest range)</td>
<td>127</td>
<td>1023</td>
<td></td>
</tr>
</tbody>
</table>

$E = 0000_2 - 0001_2 - \text{bias}$

$E = 1 - (127_{10}) = -26_{10} = 2046_{10} - 1 - 1023 = 1023$
Normalized: frac field

\[
\text{Number} = (-1) \times M \times 2^E
\]

\[
1.01110
\]

\[M = 1.\text{frac}\]
Normalized: example

\[ 12.375_{10} = (-1)^{3} \times M \times 2^{E} \]

\[ = (-1)^{3} \times (s_{6} + q_{6} \times \frac{1}{4} + \frac{1}{2}) \]

- 12.375 to single-precision floating point
- sign is positive so s=0
- binary is 1100.0112
- in other words it is 1.1000112 \times 2^{3}
- exp = E + bias = 3 + 127 = 130 = 1000_0010_{2}
- M = 1.1000112 = 1.frac
- frac = 100011

\[ 0.6022 \times 10^{-2} \]
\[ 6.022 \times 10^{-2} \]
\[ 0.6022 \times 10^{2} \]

\[ 0 \leq M < 2 \]
\[ 0.10000010 - 1000 \ldots \]

\[ \frac{1}{2} \frac{1}{2} \frac{1}{2} \frac{1}{2} \frac{1}{2} \ldots \]
16-bit float
1-bit sign
5-bit exp field
10-bit frac.

1.01100_00101_00000

Number = (-1)^s \times M \times 2^E
= (-1)^0 \times 1.15625 \times 2^{-3}

M = \text{frac.} = 1.0010100000
= 1.0 \times 2^{-1} + 0 \times 2^{-4} + 1 \times 2^{-6} + 0 \times 2^{-3} + 1 \times 2^{-2}
= 1.00125 + 0.03125
= 1.15625

E: exp - bias
= 011002 - 1011_2
= (2^4 + 2^2) - (2^4 - 1)
= 12 - 15
= -3
Table of contents

Announcements
  Programming assignment 3

Floats: Overview

Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example

Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples

Floats: Special cases

Floats: Summary

Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
The IEEE 754 number line

Figure: Full picture of number line for floating point values. Image credit CS:APP

Figure: Zoomed in number line for floating point values. Image credit CS:APP
Denormalized: exp field

For denormalized numbers, exp = 0

Bias

- need a bias to represent negative exponents
- bias = $2^{k-1} - 1$
- bias is the $k$-bit unsigned integer: 011..111

For denormalized numbers, $E = 1$-bias

<table>
<thead>
<tr>
<th>property</th>
<th>float</th>
<th>double</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>bias</td>
<td>127</td>
<td>1023</td>
</tr>
<tr>
<td>$E$</td>
<td>-126</td>
<td>-1022</td>
</tr>
</tbody>
</table>

Table: Summary of denormalized exp field
Denormalized: frac field

$M = 0.frac$
value represented leading with 0
Denormalized: examples
Table of contents

Announcements

Programming assignment 3

Floats: Overview

Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example

Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples

Floats: Special cases

Floats: Summary

Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
# Floats: Special cases

<table>
<thead>
<tr>
<th>number class</th>
<th>when it arises</th>
<th>exp field</th>
<th>frac field</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0 / -0</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>+infinity / -infinity</td>
<td>overflow or division by 0</td>
<td>$2^k - 1$</td>
<td>0</td>
</tr>
<tr>
<td>NaN not-a-number</td>
<td>illegal ops. such as $\sqrt{-1}$, inf-inf, inf*0</td>
<td>$2^k - 1$</td>
<td>non-0</td>
</tr>
</tbody>
</table>

**Table:** Summary of special cases
Table of contents
Announcements
  Programming assignment 3
Floats: Overview
Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example
Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples
Floats: Special cases
Floats: Summary
Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
## Floats: Summary

<table>
<thead>
<tr>
<th></th>
<th>normalized</th>
<th>denormalized</th>
</tr>
</thead>
<tbody>
<tr>
<td>value of number</td>
<td>((-1)^s \times M \times 2^E)</td>
<td>((-1)^s \times M \times 2^E)</td>
</tr>
<tr>
<td>E</td>
<td>(E = \text{exp-bias})</td>
<td>(E = \text{-bias + 1})</td>
</tr>
<tr>
<td>bias</td>
<td>(2^{k-1} - 1)</td>
<td>(2^{k-1} - 1)</td>
</tr>
<tr>
<td>exp</td>
<td>(0 &lt; \text{exp} &lt; (2^k - 1))</td>
<td>(\text{exp} = 0)</td>
</tr>
<tr>
<td>M</td>
<td>(M = 1.\text{frac})</td>
<td>(M = 0.\text{frac})</td>
</tr>
<tr>
<td></td>
<td>(M) has implied leading 1</td>
<td>(M) has leading 0</td>
</tr>
<tr>
<td>greater range</td>
<td>greater precision</td>
<td></td>
</tr>
<tr>
<td>large magnitude numbers</td>
<td>small magnitude numbers</td>
<td></td>
</tr>
<tr>
<td>denser near origin</td>
<td>evenly spaced</td>
<td></td>
</tr>
</tbody>
</table>

**Table:** Summary of normalized and denormalized numbers
Table of contents

Announcements
  Programming assignment 3

Floats: Overview

Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example

Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples

Floats: Special cases

Floats: Summary

Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
Deep understanding 1: Why is exp field encoded using bias?

exp field needs to encode both positive and negative exponents. Why not just use one of the signed integer formats? 2’s complement, 1s’ complement, signed magnitude?
Deep understanding 1: Why is exp field encoded using bias?

exp field needs to encode both positive and negative exponents. Why not just use one of the signed integer formats? 2’s complement, 1s’ complement, signed magnitude?

Answer: allows easy comparison of magnitudes by simply comparing bits.
Deep understanding 1: Why is exp field encoded using bias?

exp field needs to encode both positive and negative exponents. Why not just use one of the signed integer formats? 2’s complement, 1s’ complement, signed magnitude?

Answer: allows easy comparison of magnitudes by simply comparing bits.

Consider hypothetical 8-bit floating point format (from the textbook)
1-bit sign, $k = 4$-bit exp, 3-bit frac.

What is the decimal value of $0b1_0110_111$? What is the decimal value of $0b1_0111_000$?
Deep understanding 1: Why is exp field encoded using bias?

exp field needs to encode both positive and negative exponents. Why not just use one of the signed integer formats? 2’s complement, 1s’ complement, signed magnitude?

Answer: allows easy comparison of magnitudes by simply comparing bits.

Consider hypothetical 8-bit floating point format (from the textbook)
1-bit sign, \( k = 4 \)-bit exp, 3-bit frac.

What is the decimal value of 0b1_0110_111?
\(-1.875 \times 2^{-1}\)

What is the decimal value of 0b1_0111_000?
\(-2.000 \times 2^{-1}\)
Table of contents
  Announcements
    Programming assignment 3
  Floats: Overview
  Floats: Normalized numbers
    Normalized: exp field
    Normalized: frac field
    Normalized: example
  Floats: Denormalized numbers
    Denormalized: exp field
    Denormalized: frac field
    Denormalized: examples
  Floats: Special cases
  Floats: Summary
  Deep understanding 1: Why is exp field encoded using bias?
  Deep understanding 2: Why have denormalized numbers?
  Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?
Deep understanding 2: Why have denormalized numbers?

Why not just continue normalized number scheme down to smallest numbers around zero?
Answer: makes sure that smallest increments available are maintained around zero.

Suppose denormalized numbers NOT used.

What is the decimal value of 0b0_0000_001?
1.125 \times 2^{-7}

What is the decimal value of 0b0_0000_111?
1.875 \times 2^{-7}

What is the decimal value of 0b0_0001_000?
2.000 \times 2^{-7}
Deep understanding 2: Why have denormalized numbers?

Why not just continue normalized number scheme down to smallest numbers around zero?
Answer: makes sure that smallest increments available are maintained around zero.

Suppose denormalized numbers ARE used.

What is the decimal value of 0b0_0000_001? 0.125 \times 2^{-6}
What is the decimal value of 0b0_0000_111? 0.875 \times 2^{-6}
What is the decimal value of 0b0_0001_000? 1.000 \times 2^{-6}
Table of contents
Announcements
  Programming assignment 3
Floats: Overview
Floats: Normalized numbers
  Normalized: exp field
  Normalized: frac field
  Normalized: example
Floats: Denormalized numbers
  Denormalized: exp field
  Denormalized: frac field
  Denormalized: examples
Floats: Special cases
Floats: Summary
Deep understanding 1: Why is exp field encoded using bias?
Deep understanding 2: Why have denormalized numbers?
Deep understanding 3: Why is bias chosen to be $2^{k-1} - 1$?