Numbers for Computer Scientists

Chapter 1 Generic numbers

§1.1 Famous constants

π = 3.14159265358979323846264338327950288419716939937510582097494459230	(base 10)
π = 11.001001000011111101101010100010001000010110100011 …	(base 2)
π = 4000 C90F DAA2 2168 C235	(base 16, IEEE754 extended format)
π = 4009 21FB 5444 2D18	(base 16, IEEE754 double format)
e = 2.71828182845904523536028747135266249775724709369995 …	[natural logarithm]
e = 4000 ADF8 5458 82BB 4A9B	(base 16, IEEE754 extended format)
γ = 0.57721566490153286060651209008240243104215933593992 …	[gamma]

§1.2 Integers, Twos complement, n = number of bits

Largest unsigned integer = 2^n-1

Largest signed integer = $2^{n-1}-1$

Negative one = $(-1)_{10}$ = $(\text{n ones})_2$

Smallest signed integer = $(-2^{n-1})_{10}$

§1.3 Integers, Twos complement, n = 64

Largest unsigned integer = FFFF FFFF FFFF FFFF

Largest signed integer = 7FFF FFFF FFFF FFFF = $+2^{63}-1$

Negative one (signed) = FFFF FFFF FFFF FFFF = -1

Smallest signed integer = 8000 0000 0000 0000 = -2^{63}

§1.4 Integers, Twos complement, n = 128

Largest unsigned integer:
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF (8 groups) = 2^{128}-1.

Largest signed integer:
7FFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF (8 groups) = 2^{127}-1.

Negative one (signed):
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF (8 groups) = -1.

Smallest signed integer:
8000 0000 0000 0000 0000 0000 0000 0000 (8 groups) = - 2^{127} .

Chapter 2 Definitions

Definition: The base number b is equal to 1 minus the bias number.

Definition: Biased exponent and stored exponent are synonymous.

Definition: The biased exponent equals the true exponent plus the bias number.

Terminology: A group of hex digits consists of four digits, which equals one word. Thus, it can be said that one group equals one word.

Chapter 2½   IEEE 754 Extended numbers – 10-byte floats – Graphical View

The classical number line modified to show extended numbers (not shown in true scale):

```
[ Negative nans ] – ¥ [ Negative normals //////] [ Negative denormals ] 0.0 [ Positive denormals ] [////// Positive normals ] + ¥ [ Positive nans ]
a               b   c d                  e     f g                    h  I  j                    k l      m                n   o p                q
```

Names of specific values:
a: smallest negative nan
b: largest negative nan
c: negative infinity
d: smallest negative normal
e: smallest negative pseudo denormal number
f: largest negative normal number = largest negative pseudo denormal number
g: smallest negative denormal number
h: largest negative denormal number
i: zero
j: smallest positive denormal number
k: largest positive denormal number
l: smallest positive normal number = smallest positive pseudo denormal number
m: largest positive pseudo denormal number
n: largest positive normal number
o: positive infinity
p: smallest positive nan
q: largest positive nan

The slash marks, //////, indicate where pseudo denormal numbers overlap with ordinary normal numbers. Specifically, the positive pseudo denormal numbers overlap with the low-end of the positive normal numbers. Also, the negative pseudo denormal numbers overlap with the high-end of the negative normal numbers. All pseudo denormal numbers lie in the two regions indicated by slash marks.

Chapter 3 IEEE 754 Extended numbers – 10-byte floats

§3.1 Important numbers

Extended numbers are often called 10-byte float numbers.

Bias number = 3FFF = 16383 = $(2^{14} - 1)$

Base number = 4002 = -16382 = $-(2^{14} - 2)$

§3.2 Exponents

Range of true exponents for normal numbers: [-16382 .. +16383]

Range of stored exponents for normal numbers: [+1 .. +32766]

Stored exponent of 0.0 = 0_{10} = $(000\ 0000\ 0000\ 0000)_2$

Stored exponent of denormals = 0_{10} = $(000\ 0000\ 0000\ 0000)_2$ = $(0000)_{16}$

Effective exponent of denormals = base number = -16382

Stored exponent of both infinities and all nans = +32767 = 7FFF

Effective exponent of both infinities and all nans = +16384 = $(4000)_{16}$

§3.3 Significands

The stored significand holds 64 bits. The complete mathematical significand including the single bit on the left side of the radix point will be stored in the significand. The bit on the left side of the radix point will be stored in position number 63. That bit has a special status as follows.

For zero bit #63 is 0.

For ordinary denormal numbers bit #63 is 0.

For pseudo denormal numbers bit #63 is 1. Pseudo denormal numbers are not generated by the CPU, and furthermore, they overlap the low end of the normal numbers.

For normal numbers bit #63 is 1.

If bit #63 of a normal number is artificially changed to 0 then that number is an error.

For infinities and nans bit #63 is 1. If bit #63 is 0 then the number is an error.

Source: The famous tutorial by author Ray Filiatreault

§3.4 The denormal numbers – more recently called subnormal numbers

§3.4.1 Basics

These are genuine denormals and include a 0 in bit position #63.

§3.4.2 Smallest positive denormal number

Smallest positive denormal = 0000 0000 0000 0000 0001

Since this is a denormal number the base number, -16382, is the initial exponent.

Significand = 0.000 0000 … 0001 = $\left(\frac{1}{2}\right)^{63}$. There are 62 consecutive zeros between the point and the 1 on the right. All 64 binary digits are stored.

Therefore, putting it all together, the smallest positive denormal number is

$\left(\frac{1}{2}\right)^{63}$ x 2^{-16382} = 2^{-63} x 2^{-16382} = 2^{-16445}.

§3.4.3 Largest denormal number (positive)

Largest positive (genuine) denormal = 0000 7FFF FFFF FFFF FFFF
This (genuine) denormal has a 0 in position of bit #63 from the right.

This number is 0.111111 … 1 x 2^{-18362} where there are 63 consecutive ones

$$= \sum_{k=1}^{63}\left(\frac{1}{2}\right)^{k} \text{x } 2^{-16382} = \left(\frac{1}{2}\right)\sum_{k=0}^{62}\left(\frac{1}{2}\right)^{k} \text{x } 2^{-16382} = \left(\frac{1}{2}\right)\frac{1-\left(\frac{1}{2}\right)^{63}}{\left(\frac{1}{2}\right)} \text{x } 2^{-16382} = \left[1-\left(\frac{1}{2}\right)^{63}\right] \text{x } 2^{-16382}$$

§3.4.4 Smallest denormal number (negative)

The smallest denormal number is the arithmetic negative of the largest denormal number (§3.4.3). Therefore, the smallest denormal number is
8000 7FFF FFFF FFFF FFFF = $-\left[1-\left(\frac{1}{2}\right)^{63}\right]$x 2^{-16382}.

Another way to arrive at the same value is the following.

Let N be the smallest denormal number. Then N is the negative denormal number furtherest from zero. Such a number has these properties:

sign bit = 1
stored exponent = 0 .. 0 (15 zeros)
true exponent = -16382
stored significand = 0111 … 1 (zero followed by 63 ones)

Therefore, N = 1000 0000 0000 0000 0111 1111 1111 … 1111 (63 consecutive ones)

= 8000 7FFF FFFF FFFF FFFF

= – 0.111 … 1 x 2^{-16382} (63 ones right of the point)

= – 1.111 … 1 x 2^{-16383} (62 ones right of the point)

$= -\sum_{k=0}^{62} \left(\frac{1}{2}\right)^k \times 2^{-16383} \quad = -\frac{1-\left(\frac{1}{2}\right)^{63}}{1-\left(\frac{1}{2}\right)} \times 2^{-16383}$

$= -\left[1 - \left(\frac{1}{2}\right)^{63}\right] \times 2 \times 2^{-16383} \quad = -\left[1 - \left(\frac{1}{2}\right)^{63}\right] \times 2^{-16382}$. Again, the result is the same.

§3.4.5 Largest negative denormal number

This number is the arithmetic negative of smallest positive denormal number (§3.4.2). Therefore, the largest negative denormal number is

8000 0000 0000 0000 0001 = -2^{-16445}.

Another way to arrive at the same value is the following.

Let N be this number. Then N is the negative denormal number closest to zero. Such a number has these properties:

sign bit = 1
stored exponent = 0 .. 0 (15 zeros)
true exponent = -16382
stored significand = 0000 … 0001 (63 zeros followed by a single one)

Therefore, N = 1000 0000 0000 0000 … 0001 (a pair of ones with 78 zeros between them)

= 8000 0000 0000 0000 0001

= – 0.0000000 … 01 x 2^{-16382} (62 zeros between the point and the 1)

= – 1.0 x $2^{-16382-63}$ = – 2^{-16445}. Again, the result is the same.

§3.5 The pseudo denormal numbers

§3.5.1 Basics

A pseudo denormal number is a number that meets the criteria for classification as an ordinary denormal except that bit 63 is 1 rather than the standard 0. The remaining bits of the significand may be any values. Pseudo denormal numbers are not generated by modern X86 CPUs, but may be introduced by software. The CPU will correctly interpret the value of a pseudo denormal number, but, as stated before, will not create a new pseudo denormal number as a result of executing an instruction.

§3.5.2 Smallest positive pseudo denormal

The properties of the smallest positive pseudo denormal number are these:

sign bit = 0
stored exponent = 000 0000 0000 0000 (15 zeros)
stored significand = 1 followed by 63 zeros = 8000 0000 0000 0000
true exponent = -16382

Therefore, the number is 0000 8000 0000 0000 0000, which equals 1.0 x 2^{-16382}, and the latter value is exactly the smallest positive normal number. This facts supports the assertion that the pseudo denormal numbers are a subset of the normal numbers.

§3.5.3 Largest positive pseudo denormal

A pseudo denormal has a 1 in position of bit #63. All CPUs after the 80386 will not generate a pseudo denormal number from a math operation, but all such CPUs will properly interpret such a number if is received from an external source such as user input.

Largest positive pseudo denormal = 0000 FFFF FFFF FFFF FFFF

Clearly, the largest positive pseudo denormal has significand =

1.111 … 1111 (64 ones) = $\sum_{k=0}^{63}\left(\frac{1}{2}\right)^k = \frac{1-\left(\frac{1}{2}\right)^{64}}{1-\left(\frac{1}{2}\right)} = \frac{1-\left(\frac{1}{2}\right)^{64}}{\left(\frac{1}{2}\right)} = \left[1-\left(\frac{1}{2}\right)^{64}\right]$ x 2. Therefore, the largest positive pseudo denormal is

$$\left[1-\left(\frac{1}{2}\right)^{64}\right] \text{x } 2 \text{ x } 2^{-16382} = \left[1-\left(\frac{1}{2}\right)^{64}\right] \text{x } 2^{-16381}$$

§3.5.4 Smallest pseudo denormal number (negative)

Under development

§3.5.5 Largest negative pseudo denormal number

Under development

§3.5.6 Pseudo denormals overlap the normal numbers

Demonstration: Any pseudo denormal number has a general form: 1.yyy...yyyyy x 2^{-16382} where the y's are 63 arbitrary bits. From this it follows that
1.yyy...yyyyy x $2^{-16382} \geq 1.0 \times 2^{-16382} = 2^{-16382}$

The value on the right is the smallest normal number. Hence, all pseudo denormal numbers fall in the range of normal numbers, which is clearly undesirable. There is no surprise that modern CPUs do not generate pseudo denormal numbers because they already exist as normal numbers.

§3.6 The normal numbers

§3.6.1 Basics

These numbers must have a 1 in bit position #63. There are no hidden bits with EFP numbers. A “normal” number with a 0 in bit position #63 is an error, and should not occur.

§3.6.2 The smallest positive normal number

The stored exponent is the smallest integer greater than the exponent used by the denormal numbers, namely: 0001.

The stored significand is the smallest 64 bit integer provided that bit #63 is 1. Therefore, the smallest positive normal = 0001 8000 0000 0000 0000 = 1.0 x $2^{-16382} = 2^{-16382}$

§3.6.3 The largest normal number (positive)

Largest positive normal = 7FFE FFFF FFFF FFFF FFFF

The stored exponent is 7FFE = 0111 1111 1111 1110 = $2^{15} - 2 = 32766$
By subtracting the bias number one obtains the true exponent: 16383

The significand is 1.111 1111 … 1111 [64 ones] = $\sum_{k=0}^{63} \left(\frac{1}{2}\right)^k$

Now put the two parts together and find that the largest positive normal number is

$$\sum_{k=0}^{63} \left(\frac{1}{2}\right)^k \times 2^{16383} = \frac{1-\left(\frac{1}{2}\right)^{64}}{1-\left(\frac{1}{2}\right)} \times 2^{16383} = \left[1 - \left(\frac{1}{2}\right)^{64}\right] \times 2 \times 2^{16383} = \left[1 - \left(\frac{1}{2}\right)^{64}\right] \times 2^{16384}$$

§3.6.4 The smallest normal number (negative)

This number is simply the negative of the number in §3.6.3. Therefore, simply change the sign bit to be 1. The result is:

Smallest negative normal number = FFFE FFFF FFFF FFFF FFFF = $-\left[1 - \left(\frac{1}{2}\right)^{64}\right]$ x 2^{16384}.

§3.6.5 The largest negative normal number

This number is simply the negative of the number in §3.6.2. Therefore, simply change the sign bit to be 1. The result is:

Largest negative normal number = 8001 8000 0000 0000 0000 = -2^{-16382}.

§3.6.6 What normal number equals the largest pseudo denormal number?

In section §3.5.3 it is computed that the largest positive pseudo denormal number is
0000 FFFF FFFF FFFF FFFF = $\left[1 - \left(\frac{1}{2}\right)^{64}\right]$x 2^{-16381} .

Now construct the following normal number:

sign bit = 0
significand = 1.1111111111 … 1 (63 ones right of the point)
stored exponent = 000 0000 0000 0001, and therefore, the true exponent equals
stored exponent minus bias number = 1 – 16383 = –16382

This normal number is 0001 FFFF FFFF FFFF FFFF, which equals this:

1.111111111111 … 1 x 2^{-16382} = $\sum_{k=0}^{63}\left(\frac{1}{2}\right)^{k}$x 2^{-16382} = $\frac{1-\left(\frac{1}{2}\right)^{64}}{1-\left(\frac{1}{2}\right)}$x 2^{-16382} = $\frac{1-\left(\frac{1}{2}\right)^{64}}{\left(\frac{1}{2}\right)}$x 2^{-16382} = $\left[1 - \left(\frac{1}{2}\right)^{64}\right]$ x 2 x 2^{-16382} = $\left[1 - \left(\frac{1}{2}\right)^{64}\right]$ x 2^{-16381} , which is exactly the value of the largest pseudo denormal number. In hex we express this as

0000 FFFF FFFF FFFF FFFF = 0001 FFFF FFFF FFFF FFFF.

The value on the left is pseudo denormal and the value on the right is ordinary normal. Try to explain that equality to a beginner.

§3.6.7 What normal number equals the smallest positive pseudo denormal number?

According to §3.6.2 the smallest positive normal number is
0001 8000 0000 0000 0000 = 2^{-16382}.

According to §3.5.2 the smallest positive pseudo denormal number is
0000 8000 0000 0000 0000 = 2^{-16382}.

Since both hex values are equal to the common value 2^{-16382} they must be equal to each other. Therefore, 0000 8000 0000 0000 0000 = 0001 8000 0000 0000 0000.

§3.7 The Nans

§3.7.1 Basics

A nan is a pseudo-numeric entity with these defining properties.

- The stored exponent is 7FFF
- Bit 63 of the significand is 1 as is required of all efp #s except denormals
- Bits 62-0 may assume any binary value
- The sign bit may be any binary value

The information presented here about nans is heavily borrowed from the tutorial “Simply FPU” by Raymond Filiatreault.

§3.7.2 Infinity

There are two infinities and they are both nans. An infinity satisfies the additional property that bits 62-0 are all zeros.

If the sign bit is 0 the infinity is called positive infinity. Specifically, positive infinity is
+¥ = 7FFF 8000 0000 0000 0000 with a theoretical decimal value of $1.0 \times 2^{16384} = 2^{16384}$.

If the sign bit is 1 the infinity is called negative infinity. Specifically, negative infinity is
-¥ = FFFF 8000 0000 0000 0000 with a theoretical decimal value of $– 1.0 \times 2^{16384} = -2^{16384}$.

An infinity value may be generated by the CPU in cases such as

- An attempt to divide a value number by 0.0
- A computation that results in an overflow, eg $(1.0 \times 2^{15383}) \times 5.0$
- Attempt to store a value that overflows the destination

§3.7.3 Nans other than infinity

§3.7.3.1 Qnans, which are also called Quiet Nans

This is a class of nans with these properties:

- The sign bit may be either 0 or 1
- The stored exponent is 7FFF
- Bit 63 is 1
- Bit 62 is 1
- Bits 61-0 may assume any value.

An example of a qnan: 7FFF C100 0000 0000 0000

Special case of qnan: the indefinite nan. There is a single qnan that receives special identification and is known as the indefinite nan. The properties of the indefinite nan are all the properties of a qnan and the additional requirements:

The sign bit is 1
Bits 61-0 are all zeros.

The value of the indefinite nan is FFFF C000 0000 0000 0000 = 1.5x2^{16384}. The indefinite nan is generated by the FPU in various conditions. One such condition is the attempt to compute the square root of a negative number. There are other conditions also.

§3.7.3.2 Snans or signaling nans

This is a class of nans with these properties:

The sign bit may be either 0 or 1
The stored exponent is 7FFF
Bit 63 is one
Bit 62 is zero
At least one bits among bits 61-0 is a one.

An example of an snan: FFFF A000 0000 0000 0000 = 1.25 x 2^{16384}.

§3.7.3.3 The successor of positive infinity

This nan could be called the first positive nan or the smallest positive nan. It is described by the following:

The sign is 0
The exponent is all ones
Bit number 63 is 1 because that bit is mandated to be 1.
Bits 62-1 are all zeros
Bit number 0 is 1.

If bit number 0 were not 1 then we would have here positive infinity, but we want the next value beyond positive infinity, and therefore, bit number 0 is 1. Put all the facts together to obtain the following number:

0 111 1111 1111 1111 1000 0000 0000 0000 0000 ……. 0001 = 7FFF 8000 0000 0000 0001.

§3.7.3.4 The last nan

It is not proper to call this nan the “largest nan” because officially there is no order among nans. However, it is the last nan on the far right side of the number line. Look at the properties of this number:

The sign is 0
The exponent is all ones
Bit number 63 is 1
Bits 62-0 are all 1's

Therefore, the number is 7FFF FFFF FFFF FFFF FFFF.

§3.8 Zero

There exists both a positive zero and a negative zero. The zeros have their own special representation, namely: the exponent is all zeros and the significand is all zeros. Hence,

+0.0 = 0000 0000 0000 0000 0000
-0.0 == 8000 0000 0000 0000 0000

§3.9 Distance between successive extended fp numbers

Let's pose the question: What is the distance between a normal efp number and its immediate successor? Implicit in the question is the assumption that the starting number does have a successor that is normal. An answer is given in the following paragraphs.
Answer:
Let N be a positive normal efp number with an immediate successor M. We want to find D = M – N.

N can be represented in a canonical exponential form: $N = S \times 2^E$, where S is the significand satisfying $1 \le S \quad 2$, and E is an integer satisfying $-16382 < E < 16383$.

Therefore, $M = (S + 0.0000 \ldots 01) \times 2^E$, and $D = M - N = 0.0000 \ldots 01 \times 2^E = \left(\frac{1}{2}\right)^{63} \times 2^E = 2^{E-63}$.

Starting with $N = S \times 2^E$, we arrive at $\log_2(N) = \log_2(S) + E$. The inequality $1 \le S \quad 2$ implies that $\left(\frac{1}{2}\right) < \left(\frac{1}{S}\right) \le 1$.
Therefore, $D = 2^{E-63} = 2\text{^}(\log_2(N) - \log_2(S) - 63) = N \times \left(\frac{1}{S}\right) \times 2^{-63}$.

We can place bounds around D in the following manner:
$D = N \times \left(\frac{1}{S}\right) \times 2^{-63} \le N \times 2^{-63}$ and $D = N \times \left(\frac{1}{S}\right) \times 2^{-63} > N \times \left(\frac{1}{2}\right) \times 2^{-63} = N \times 2^{-64}$. Combining inequalities yields: $N \times 2^{-64} < D \le N \times 2^{-63}$.

Example 1.
Suppose $N = 2^{128}$. Then S = 1.0 . Then the distance from N to its successor is
$D = N \times \left(\frac{1}{1}\right) \times 2^{-63} = 2^{128} \times 2^{-63} = 2^{65}$, which is a large distance between successive numbers.

Example 2.
Suppose $N = 2.5 = 1.25 \times 2^1$. Then the distance $D = N \times \left(\frac{1}{1.25}\right) \times 2^{-63} = 2.5 \times 0.8 \times 2^{-63} = 2.0 \times 2^{-63} = 2^{-62}$, which is a small distance.

Conclusion. As a normal efp number becomes larger in absolute value so does the distance between that number and its successor. That is to say, floating point numbers are farther apart at the extreme ends of the number line than they are toward the center of the number line.

A new question. What is the distance between two successive denormal numbers?
Answer:

Let N be a (positive) denormal number. Then N can be represented as $N = S \times 2^E$ where E =

-16382 and $0 < S < 1$.

The successor of N is M = (S+0.00000 ... 1) x 2^E., and so, the difference is
D = M-N = 0.00000 … 1 x 2^E = $\left(\frac{1}{2}\right)^{63}$ x 2^E = 2^{-63} x 2^{-16382} = 2^{-16445}, a value which is independent of the original number N.

Conclusion. Denormal numbers are equally spaced, and the distance between two adjoining numbers is 2^{-16445}.

Chapter 4 IEEE 754 Double precision numbers–8-byte floats

Hiding Principle: The single bit of the significand left of the radix point is hidden (omitted from storage) for all 8-byte floating point numbers.

§4.1 Important numbers

Bias number = 3FF = 1023 = $(2^{10} - 1)$

Base number = 402 = -1022 = $-(2^{10} – 2)$

§4.2 Exponents

All exponents contain 11 bits.

Range of true exponents for normal numbers: [-1022 .. +1023]

Range of stored exponents for normal numbers: [+1 .. +2046]

Stored exponent of 0.0 = 0_{10} = $(000\ 0000\ 0000)_2$

Stored exponent of denormals = 0_{10} = $(000\ 0000\ 0000)_2$ = $(000)_{16}$

True exponent of denormals = base number = -1022

Stored exponent of both infinities and all nans = +2047 = 2^{11}-1 = 7FFF

True exponent of both infinities and all nans = +1024 = $(400)_{16}$

§4.3 Significands

The true significand is the significand used in arithmetic operations. In 8-byte float numbers the true significand has 53 bits: 1 bit on the left of the radix point and 52 bits on the right of the radix point. The stored significand holds the 52 bits of the right side of the true significand. The left-most bit, which is not in the stored significand, is commonly called the hidden bit.

§4.4 The subnormal numbers – formerly called denormal numbers.

§4.4.1 Basics

The denormal numbers lie in these ranges:

largest negative normal < negative denormals < zero < positive denormals < smallest positive normal.

The leading bit of the significand, a zero bit, is not stored. The concept of a pseudo denormal number does not apply to these 8-byte numbers.

§4.4.2 Smallest positive denormal number

Smallest positive denormal = 0000 0000 0000 0001 = 2^{-1074}.

The stored significand is 000 … 01 (51 zeros followed by a single 1.
The true significand is 0.000 … 01 (the hidden zero appears on the left of the point)
The true exponent is -1022.
Therefore, the smallest positive denormal number = 0.00 … 01 x 2^{-1022} =
$\left(\frac{1}{2}\right)^{52}$ x $2^{-1022} = 2^{-1074}$

§4.4.3 Largest denormal number (positive)

The largest denormal number has these properties:

The sign bit is 0.
The stored significand is 111 … 1 (52 ones)
The true significand is 0.111 … 1 (the hidden zero is on the left of the point)
The stored exponent is 000 0000 0000.
The true exponent is -1022.
Therefore, the largest positive denormal number = 000F FFFF FFFF FFFF =

$$0.111 \ldots 1 \times 2^{-1022} = \sum_{k=1}^{52} \left(\frac{1}{2}\right)^{k} \times 2^{-1022} = \left(\frac{1}{2}\right)\sum_{k=0}^{51} \left(\frac{1}{2}\right)^{k} \times 2^{-1022} = \left(\frac{1}{2}\right)\frac{1-\left(\frac{1}{2}\right)^{52}}{\left(1-\frac{1}{2}\right)} \times 2^{-1022} =$$

$$\left[1 - \left(\frac{1}{2}\right)^{52}\right] \times 2^{-1022}$$

Summary: 000F FFFF FFFF FFFF = $\left[1 - \left(\frac{1}{2}\right)^{52}\right] \times 2^{-1022}$

§4.4.4 Smallest denormal number (negative)

The smallest denormal number is the negative of the largest denormal number (§4.4.3).
Therefore, the smallest denormal number is 800F FFFF FFFF FFFF = $-\left[1 - \left(\frac{1}{2}\right)^{52}\right] \times 2^{-1022}$.

§4.4.5 Largest negative denormal number

The largest negative denormal number is the negative of the smallest positive denormal number (§4.4.2).

Therefore, the largest negative denormal number is 8000 0000 0000 0001 = -2^{-1074}.

§4.5 Pseudo denormal numbers in 8 bytes

The concept does not exist in 8 bytes because of the principle of hiding the leading bit of the significand. The hidden bit is always 1 for normal numbers and is always 0 for denormal numbers – no exceptions. Bit number 51 is undistinguished; it may assume either 0 or 1 as a value

§4.6 The normal numbers

§4.6.1 Basics

The lead bit of the stored significand is bit number 51. It is undistinguished since it may received either 0 or 1 as a valid value. The true significand must have a 1 on the left side of the radix point and no other digits will be on the left of the point. That single 1 is not stored, and is commonly referred to as the hidden bit.

§4.6.2 The smallest positive normal number.

The smallest exponent in the allowed range of exponents is 000 0000 0001. The smallest stored significand for normals is 0000 … 0 (52 zeros).

Now put the pieces together: the smallest positive normal number is
0000 0000 0001 0 … 0 with 52 zeros on the right side of the single one bit. Therefore, the smallest positive normal number in hex is 0010 0000 0000 0000.

Smallest positive normal number in decimal. The true exponent is the stored exponent minus the bias number, which is 1 – 1023 = -1022. Therefore, the smallest positive normal number is 1.00000 x 2^{-1022} = 2^{-1022}.

Therefore, it is correct to write 0010 0000 0000 0000 = 2^{-1022}.

§4.6.3 The largest normal number (positive).

The largest positive normal number has these properties.

The sign bit is 0.

The stored exponent is 111 1111 1110. = 2046.
The true exponent is 2046 – 1023 = 1023.
The stored significand is 111 … 1 (52 ones).

Therefore, the largest positive normal number is

0111 1111 1110 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 =
7FEF FFFF FFFF FFFF. = 1.111 … 1 x 2^{1023} (52 ones right of the radix point) =

$$\sum_{k=0}^{52}\left(\frac{1}{2}\right)^{k} \text{ x } 2^{1023} = \frac{1-\left(\frac{1}{2}\right)^{53}}{1-\left(\frac{1}{2}\right)} \text{ x } 2^{1023} = \left[1-\left(\frac{1}{2}\right)^{53}\right] \text{ x } 2 \text{ x } 2^{1023} = \left[1-\left(\frac{1}{2}\right)^{53}\right] \text{ x } 2^{1024} .$$

Summary: 7FEF FFFF FFFF FFFF = $\left[1-\left(\frac{1}{2}\right)^{53}\right]$ x 2^{1024} .

§4.6.4 The smallest normal number (negative).

The number is the arithmetic negative of the largest normal number of §4.6.3. Therefore, this number is FFEF FFFF FFFF FFFF = $-\left[1-\left(\frac{1}{2}\right)^{53}\right]$ x 2^{1024} .

§4.6.5 The largest negative normal number.

This number is the arithmetic negative of the smallest positive normal number (§4.6.2). Therefore, it is 8010 0000 0000 0000 = -2^{-1022}.

§4.7 The Nans

§4.7.1 Basics

A nan is a pseudo-numeric entity with one defining property, namely: the stored exponent is 7FF. The sign bit may be either 0 or 1, and thus, there are both positive and negative nans. The bits of the significand may assume either 0 or 1 as a value.

§4.7.2 Infinity

There are two infinities: one positive and one negative. Both infinities are special nans. An infinity is characterized by all the bits of the significand being zeros. Therefore,

+¥ = 7FF0 0000 0000 0000 with a theoretical decimal value of +1.0 x 2^{1024} = $+2^{1024}$, and
-¥ = FFF0 0000 0000 0000 with a theoretical decimal value of – 1.0 x 2^{1024} = -2^{1024}.

§4.7.3 Nans other than infinity

If at least one bit of the significand is 1 and the stored exponent is 7FF then the number is a nan not equal to one of the two infinities.

§4.7.3.1 Successor of positive infinity

Since positive infinity is 7FF0 0000 0000 0000 its successor must be 7FF0 0000 0000 0001 = 1.0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 x 2^{1024} = $\left[1+\left(\frac{1}{2}\right)^{52}\right]$ x 2^{1024} .

§4.7.3.2 The last positive nan

It is not proper to to think of nans as having an ordering scheme because they are not true numbers. The X86 instruction sets does not provide an instruction that compares a nan with a number or a nan with a nan. However, humans can imagine an ordering among nans where the larger nan has a significand larger than the significand of the smaller nan. Using that ordering scheme we find that the last or largest positive nan is 7FFF FFFF FFFF FFFF =

1.1111 1111 1111 …. 1111 x 2^{1024} (52 ones on the right of the point) =

$$\sum_{k=0}^{52}\left(\frac{1}{2}\right)^k \text{ x } 2^{1024} = \frac{1-\left(\frac{1}{2}\right)^{53}}{1-\left(\frac{1}{2}\right)} \text{x } 2^{1024} = \left[1-\left(\frac{1}{2}\right)^{53}\right] \text{ x } 2 \text{ x } 2^{1024} = \left[1-\left(\frac{1}{2}\right)^{53}\right] \text{ x } 2^{1025} .$$

§4.8 Zero

There are two zeros: one is positive and one is negative. In both cases the stored exponent is all zeros and the significand is all zeros. Hence, the two zeros are:

+0 = 0000 0000 0000 0000 and -0 = 8000 0000 0000 0000

This leads to an interesting dilemma. Use the standard algorithm to convert quad number to decimal. This is the result.

+0 = 0000 0000 0000 0000 = 1.0000 …. 0000 x 2^{-1022} = 2^{-1022}, or simply stated:

+0 = 2^{-1022}.

In a similar fashion one can show that – 0 = – $2^{-1022.}$

§4.9 Distance between successive double precision numbers.

//Under development

4.10 Tabular summary of 8-byte non-negative floating point numbers.

Numeric Type	Stored exponent unsigned binary int	Stored exponent unsigned hex int	True exponent signed decimal int	Hidden bit	Significand requirements	Min value decimal	Max value decimal
Positive zero	000 0000 0000	000	-1022	0	52 zeros	+0.0	+0.0
Subnormal	000 0000 0000	000	-1022	0	At least 1 bit of 52 must be nonzero	2^{-1074}	$\left[1-\left(\frac{1}{2}\right)^{52}\right]$ x 2^{-1022}
Normal	000 0000 0001 thru 111 1111 1110	001 thru 7FE	-1022 thru +1023	1	Any 52 bits: ones or zeros	2^{-1022}	$\left[1-\left(\frac{1}{2}\right)^{53}\right]$ x 2^{1024}
Positive infinity	111 1111 1111	7FF	+1024	1	52 zeros	2^{1024}	2^{1024}
Positive nan	111 1111 1111	7FF	+1024	1	At least 1 bit of 52 must be nonzero	$\left[1+\left(\frac{1}{2}\right)^{52}\right]$ x 2^{1024}	$\left[1-\left(\frac{1}{2}\right)^{53}\right]$ x 2^{1025}

Chapter 5 IEEE 754 Quadruple precision – 16-byte numbers

§5.1 Important numbers

Quadruple precision numbers may be called 16-bytes floats

The important numbers are the same as those of the Extended Precision type. Refer to §3.1.

Bias number = 3FFF = 16383 = $(2^{14} - 1)$

Base number = 4002 = -16382 = $-(2^{14} - 2)$

§5.2 Exponents

The facts about exponents are identical to those in §3.2. For convenience those facts are restated here.

Range of true exponents for normal numbers: [-16382 .. +16383]

Range of stored exponents for normal numbers: [+1 .. +32766]

Stored exponent of 0.0 = 0_{10} = $(000\ 0000\ 0000\ 0000)_2$

Stored exponent of denormals = 0_{10} = $(000\ 0000\ 0000\ 0000)_2$ = $(0000)_{16}$

Effective exponent of denormals = base number = -16382

Stored exponent of both infinities and all nans = +32767 = 7FFF

Effective exponent of both infinities and all nans = +16384 = $(4000)_{16}$

§5.3 Significands

The stored significand holds 112 bits and hides 1 bit giving an effective precision of 113 bits.

The hidden bit is the single bit on the left of the radix point of the true significand.

§5.4 Denormal numbers – more recently called subnormal numbers.

§5.4.1 Basics

The denormal numbers lie in the two ranges on opposite sides of zero:

largest negative normal # < denormals < 0.0 < denormals < smallest positive normal #

The leading bit of the true significand, a zero, is not stored, but it is hidden.

The stored exponent of all denormal numbers is 000 0000 0000 0000 (15 zeros).

The true exponent of all denormal numbers is -16382.

The absolute value of a denormal number will be less than the smallest positive normal number: see §5.6.2.

§5.4.2 Smallest positive denormal number

The number is positive, therefore, the sign bit = 0.
The number is denormal, therefore, its stored exponent must be zero.
The true exponent is the base number = -16382.
The stored significand is 0000 … 001 (111 zeros followed by a single 1)

Combine the three components, sign, exponent, significand, to obtain:

Smallest positive denormal number = 0000 … 001 (127 zeros followed by 1)

= 0000 0000 0000 0000 0000 0000 0000 0001
(8 groups of hex digits with 1 appearing at the far right).

The hex number above is converted to base 10 as follows. Note these facts.
sign bit = 0
stored exponent = 0
true exponent = base number = -16382
true significand = 0.0000 … 0001 (111 zeros between the point and the 1)

Therefore, the smallest positive denormal number in decimal is

$0.0000 \ldots 0001 \times 2^{-16382} = \left(\frac{1}{2}\right)^{112} \times 2^{-16382} = 2^{-16494}$, which is considered to be a valid expression for a number in base 10. In summary,

0000 0000 0000 0000 0000 0000 0000 0001 = 2^{-16494}.

§5.4.3 The largest denormal number (positive)

The largest positive denormal number has these properties.
sign bit = 0
stored exponent = 000 0000 0000 0000
true exponent = -16382
hidden bit = 0
stored significand = 1111 1111 1111 … 1111   (112 ones)

Combine the information above to obtain the number:

0000 0000 0000 0000 1111 1111 1111 … 1111 =

0000 FFFF FFFF FFFF FFFF FFFF FFFF FFFF   (8 groups) =

0.1111111 … 1 x 2^{-16382} = $\sum_{k=1}^{112}\left(\frac{1}{2}\right)^k$ x 2^{-16382} = $\left(\frac{1}{2}\right)\sum_{k=0}^{111}\left(\frac{1}{2}\right)^k$ x 2^{-16382} =
$\left(\frac{1}{2}\right)\frac{1-\left(\frac{1}{2}\right)^{112}}{\left(\frac{1}{2}\right)}$ x 2^{-16382} = $\left[1-\left(\frac{1}{2}\right)^{112}\right]$ x 2^{-16382} .

§5.4.4 Smallest denormal number (negative)

§5.4.5 Largest negative denormal number

§5.11 Distance between successive double precision numbers.

§5.12 Useful facts

A. Proposition. If a floating point octoword number has true exponent f then the distance from the number to its successor is 2^{f-112}.

Demonstration. Let X = any floating point octoword number with true exponent f. Then X = signifcand x 2^f and 1.0 ≤ signifcand < 2.0 .

To obtain the successor float number we add to the significand of X the smallest increment possible, namely: 0.0000 …. 0001 (111 zeros on the right of the point) = $\left(\frac{1}{2}\right)^{112}$ = 2^{-112} .
Therefore, the successor of X is succ(X) = (signifcand + 2^{-112}) x 2^f . From here we can easily compute distance = succ(X) – X = (signifcand + 2^{-112}) x 2^f – signifcand x 2^f = 2^{-112} x 2^f = 2^{f-112}.

End of proposition.

B. Exercise. Find the smallest floating point octoword number X such that the distance from X to its successor is exactly 1.

Solution. Let X be the wanted number. Then X has some true exponent which we call f. From the proposition we know that the distance to the successor of X is 2^{f-112} . However, the exercise says that the distance is 1. Therefore, it must be true that $2^{f-112} = 1$. However, that implies that f = 112.

At this point we know that X = significand x 2^{112} for some number called significand. Since the exercise wants the smallest possible octoword we choose for significand the smallest number allowed as significand, namely: 1.000000 … 0000 = 1. Therefore, X = 1.0 x $2^{112} = 2^{112}$, and that is the wanted number: X = 2^{112} .

Footnote to the exercise: Express X in IEEE754 standard hex. Students enrolled in CPSC240 are very adept at number conversion and they are able to quickly solve this problem:
X = 406F 0000 0000 0000 0000 0000 0000 0000.

C. Comment on the Exercise B. The number 2^{112} is special. Clearly, it is an integer and the next larger octoword number (2^{112} + 1) is also an integer with no floating point number between 2^{112} and (2^{112} + 1) ; the fractions have disappeared. In fact, all the fractions have disappeared for numbers beyond 2^{112} . Look at the next proposition.

D. Proposition. If X is any floating point octoword $\geq 2^{112}$ then X is an integer.

Proof. Let X be any octoword float. Then X = (significand) x 2^w, and 1.0 ≤ significand < 2.0 and w is the true exponent.

The following is a given condition: $2^{112} \leq$ X = (significand) x 2^w < 2.0 x $2^w = 2^{w+1}$.

Now pick out the first and last terms: $2^{112} < 2^{w+1}$. This leads to 112 < w+1, which in turn leads to 111 < w. But w is an integer, therefore, 112 ≤ w. So, we have established that w is at least the size of 112, and perhaps larger.

Now return to X, which can be represented at X = $1.d_1d_2d_3 \ldots d_{112}$ x 2^w where $d_1, d_2, d_3, \ldots d_{112}$ are binary digits and w is the true exponent in the range with the constraint: 112 ≤ w. Now look closely at the X expression:

X = $1.d_1d_2d_3 \ldots d_{112}$ x 2^w = $\left(1 + \sum_{k=1}^{112} d_k\left(\frac{1}{2}\right)^k\right)x2^w = \left(2^w + \sum_{k=1}^{112} d_k 2^{w-k}\right)$. In the latter expression every component part is an integer, and therefore, X must be an integer. End of proof.

E. Think about the result. The proposition says that beyond a certain number, 2^{112} , there are no more fractions. Yes, the true exponents continue through 16383, but after exponent 112 the numbers are all integers.

What should 256-bit floats do to improve on this situation?

Epilogue

This document was begun in July 2015 as an attempt to gather in one place as many facts about numbers used in computing as possible. The project is still continuing although at a much reduced rate of development.

If any reader has ideas, comments, or corrections then send them to me at

holliday@fullerton.edu

Floyd Holliday
Date of last update: 2018-Jan-28.

<u>Beyond this page this document is under development. Do not rely on the calculations that follow.</u>

What is zero point zero?

The stored exponent is 000 0000 0000 0000
The true exponent is -16382
The stored significand is 1111 … 1111 [112 ones]
The true significand is 0.111 … 111 [112 ones right of the point]
Therefore, the largest positive (true) denormal number is

$$0.111 \ldots 111 \times 2^{-16382} = \sum_{k=1}^{112} \left(\frac{1}{2}\right)^k \times 2^{-16382} = \left(\frac{1}{2}\right)\sum_{k=0}^{111} \left(\frac{1}{2}\right)^k \times 2^{-16382} = \left(\frac{1}{2}\right)\frac{1-\left(\frac{1}{2}\right)^{112}}{\left(\frac{1}{2}\right)} \times 2^{-16382}$$

$$= \left[1 - \left(\frac{1}{2}\right)^{112}\right] \times 2^{-16382}.$$

{The above needs a second check for accuracy.}

Smallest positive normal number.

0x0001 0000 0000 0000 0000 0000 0000 0000

The true exponent is 1 minus the bias number, which equals -16382.
The significant is 1.00 … 0 with 112 trailing zeros
Therefore the smallest positive normal number is $1.0 \times 2^{-16382} = 2^{-16382}$

Largest positive normal

0x7FFE FFFF FFFF FFFF FFFF FFFF FFFF FFFF

The stored exponent is 7FFE = 32766, and hence the true exponent is 16383.
The significand is 1.111 … 1111 with a total of 113 one-bits. Therefore, the largest positive normal number is $1.111 \ldots 1111 \times 2^{16383} = \sum_{k=0}^{112} \left(\frac{1}{2}\right)^k \times 2^{16383} = \frac{1-\left(\frac{1}{2}\right)^{113}}{1-\left(\frac{1}{2}\right)} \times 2^{16383} = \left[1 - \left(\frac{1}{2}\right)^{113}\right]$
$\times 2^{16384}$

Positive infinity

Smallest positive nan

Largest positive nan

Distance between the largest positive (true) denormal and the smallest positive normal.

Conversions of decimal numbers to quadruple precision hex and the reverse.This is a work in progress.

For many of the above numbers the corresponding decimal value is yet to be posted.

Eventually there will be sections for all of the following:
    == Extended precision numbers (80 bits)
    == Double precision numbers (64 bits)
    == Quadruple precision numbers (128 bits)

If you have ideas about numbers then send me email:

If you find an error tell me about it. I don't mind being corrected. My goal is scientific correctness.

activeprofessor@yahoo.com