Digital Signal Processing

DSP Number Systems
Floating Point Numbers

Today's Topics

- Floating Point Number Representations
  - IEEE format
- Round off error in floating point numbers
  - Can look like quantization noise
- Floating Point Dynamic Range and Precision
  - Range of values
  - Resolution of values

Fixed Point Numbers Review

- Recall that fixed point numbers can represent unsigned or signed (2's complement) values
- Arduino has two lengths of fixed point numbers
  - INT – 16 Bits
  - LONG – 32 Bits
- Fixed Point numbers can also represent fractions with limited range – QM.N values and using scaling

Floating Point

- To represent numbers with greater precision and a wider range of values, floating point representation developed
- Floating point representation is like scientific notation
- Two levels of precision are accommodated
  - Single and Double Precision
- Single precision numbers can range from as large as $\pm 3.4 \times 10^{38}$ to as small as $\pm 1.2 \times 10^{-38}$

Floating Point and Scientific Notation

- In scientific notation numbers are normalized to a value between 1 and 9.9999 then multiplied by 10 raised to an exponent

$$-5.3782 \times 10^{8}$$

Sign

Value between 1 and 9.999 (MANTISSA)

10 raised to an EXPONENT

Floating Point and Scientific Notation

- In a similar way floating point numbers consist of a sign, a value normalized to between 1 and 1.999999 multiplied by 2 raised to an exponent

$$-1.32456 \times 2^{19} = 6.9445 \times 10^{5}$$

Sign

2 raised to an EXPONENT

Value between 1 and 1.999
(MANTISSA)

Single Precision Floating Point

- The value represented by the number is

Sign Bit

Mantissa

Exponent (E)

Note that the exponent is the value E.
2 is raised to E-127

$$v = (-1)^{S} \times M \times 2^{(E-127)}$$

0	10010100	10010100100101001001010

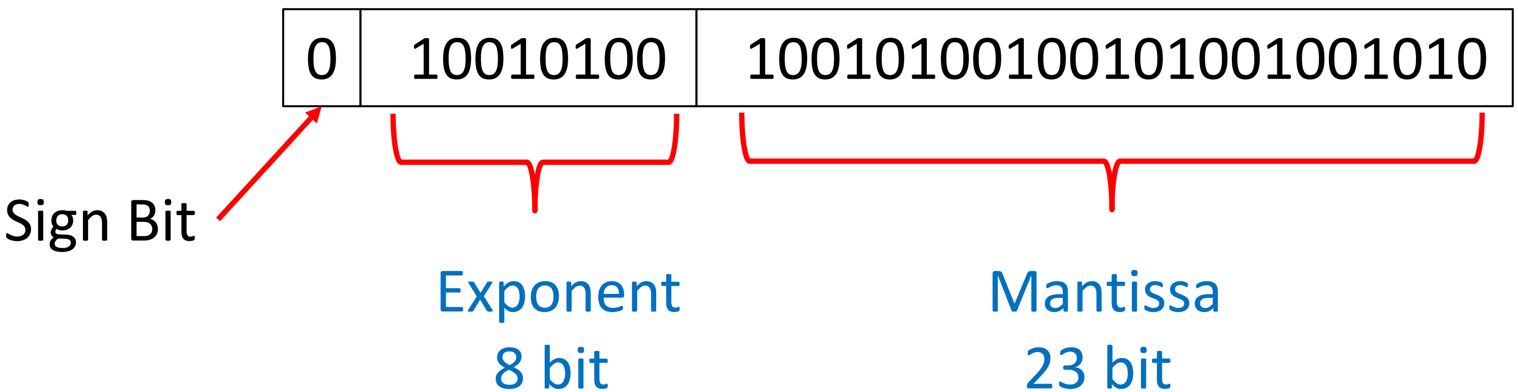
Sign Bit

Exponent
8 bit

Mantissa
23 bit

Single Precision Floating Point

- A single precision floating point number is made up of 32 bits (4 Bytes)



$$32\ Bits = 1 + 8 + 23 = 4\ Bytes$$

Looking at the Mantissa

- Considering just the mantissa

$$M = 1 + m_{22}2^{-1} + m_{21}2^{-2} + m_{20}2^{-3} + \cdots$$

$$1.32456 = 1 + (0)2^{-1} + (1)2^{-2} + (0)2^{-3} + (1)2^{-4} + \cdots$$

$$1.\underbrace{0101...0011000101101100101}_{\substack{\text{Mantissa} \\ \text{23 bits}}}$$

The leading bit is always 1 so we don't need to store it

Looking at the Exponent

- The stored exponent is an unsigned integer of 8 bits.
- 127 is subtracted from that value to allow positive and negative values from +128 to -127

Sign Bit

Mantissa

Exponent

$$v = (-1)^S \times M \times 2^{(E-127)}$$

Power of 2 ranges from 2^{-127} to 2^{128}

Single Precision Floating Point Range

- Theoretically, the largest signed number that can be represented is:

$$\pm(2 - 2^{-23}) \times 2^{128} = \pm 6.8 \times 10^{38}$$

- Theoretically, the smallest signed number that can be represented is:

$$\pm 1.0 \times 2^{-127} = \pm 5.9 \times 10^{-39}$$

IEEE Standard

- The IEEE defines the standard for floating point values
  - ANSI/IEEE Std. 754-1985

- The theoretical range is reduced to free up bit patterns to represent special meanings.
  - Largest -- $\pm 3.4 \times 10^{38}$
  - Smallest -- $\pm 1.2 \times 10^{-38}$
- Special values $\pm 0, \pm\infty, NAN$, etc..

Double Precision Floating Point

- A double precision floating point is made from 64 bits or 8 Bytes

Sign Bit

0	10010100111	110010100100101001001010100101001001010010010100111

Exponent
11 bit

Mantissa
52 bit

Double Precision Floating Point

- Largest Value -- $\pm 1.8 \times 10^{308}$
- Smallest Value -- $\pm 2.2 \times 10^{-308}$

Sign Bit

0	10010100111	110010100100101001001010100101001001010010010100111
	Exponent 11 bit	Mantissa 52 bit

Floating Point Example

- Find the decimal number that corresponds to the following floating point bit pattern.
  - 1 01110001 01010100000000000000000

Recall that the value is computed using

Sign Bit

Mantissa

Exponent

$$v = (-1)^S \times M \times 2^{(E-127)}$$

Floating Point Example

- Find the decimal number that corresponds to the following floating point bit pattern.
  - 1 01110001 01010100000000000000000

First Identify the components

1	01110001	01010100000000000000000

Sign Bit

Exponent
8 bit

Mantissa
23 bit

$$\boxed{v = (-1)^S \times M \times 2^{(E-127)}}$$

Find the Sign Bit

- Find the value of the sign bit

1	01110001	01010100000000000000000

Sign Bit

Exponent
8 bit

Mantissa
23 bit

Sign Bit = 1 -- Negative number

$$v = (-1)^{S} \times M \times 2^{(E-127)}$$

Find the Exponent

- Find the value of the exponent

1	01110001	01010100000000000000000

Sign Bit

Exponent
8 bit

Mantissa
23 bit

$$\text{Exponent} = 01110001b - 127d$$

$$\text{Exponent} = 113d - 127d = -14$$

$$\boxed{v = (-1)^S \times M \times 2^{(E-127)}}$$

Find the value of the Mantissa

- Considering just the mantissa

1	01110001	01010100000000000000000

Sign Bit

$\underbrace{01110001}_{\text{Exponent 8 bit}}$ $\underbrace{01010100000000000000000}_{\text{Mantissa 23 bit}}$

Exponent
8 bit

Mantissa
23 bit

$$M = 1 + m_{22}2^{-1} + m_{21}2^{-2} + m_{20}2^{-3} + \cdots$$

$$M = 1 + (0)2^{-1} + (1)2^{-2} + (0)2^{-3} + (1)2^{-4} + (0)2^{-5} + (1)2^{-6}$$

$$M = 1 + 0 + (1/4) + 0 + (1/16) + 0 + (1/64) = 1.328125$$

Putting it all together

- Apply the equation to the component parts

$$S = 1 \qquad M = 1.328125 \qquad E - 127 = -14$$

Sign Bit

Mantissa

Exponent

$$v = (-1)^S \times M \times 2^{(E-127)}$$

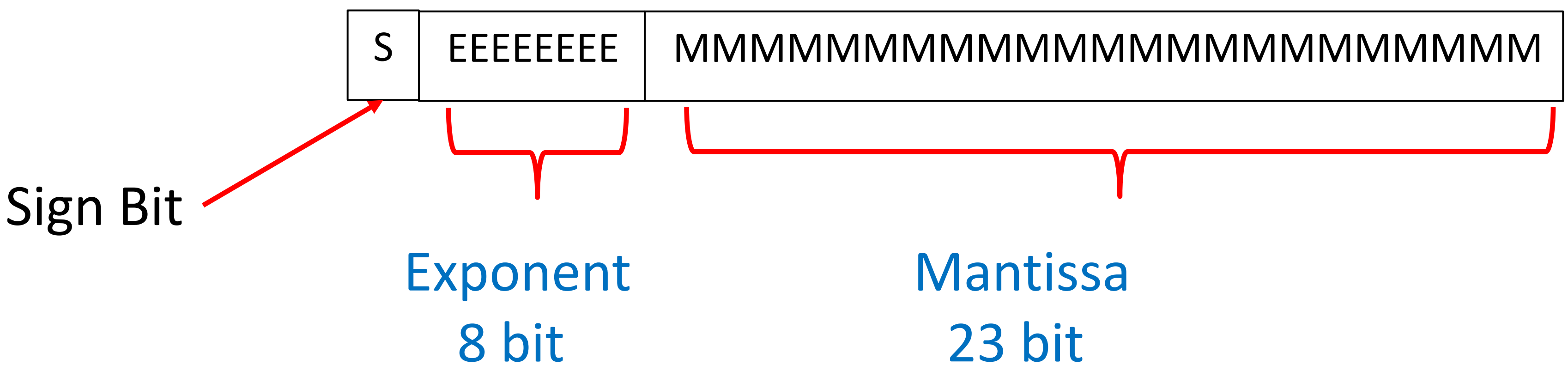
$$v = (-1)^1 \times 1.328125 \times 2^{-14}$$

$$v = -8.10623 \times 10^{-5} = -.0000810623$$

Floating Point
In Class Problem (1)

- Find the decimal number that corresponds to the floating point bit pattern.

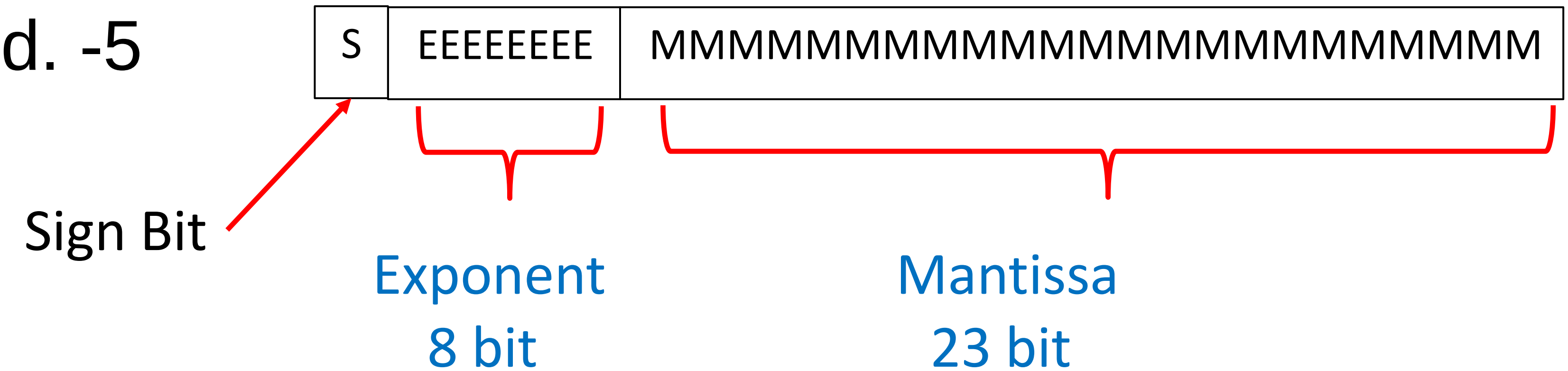
0 11110011 11110010000000000000000



Floating Point
In Class Problem (2)

- Convert the following decimal numbers into their IEEE floating point bit patterns:
- a. 1
- b. 2
- c. 4
- d. -5

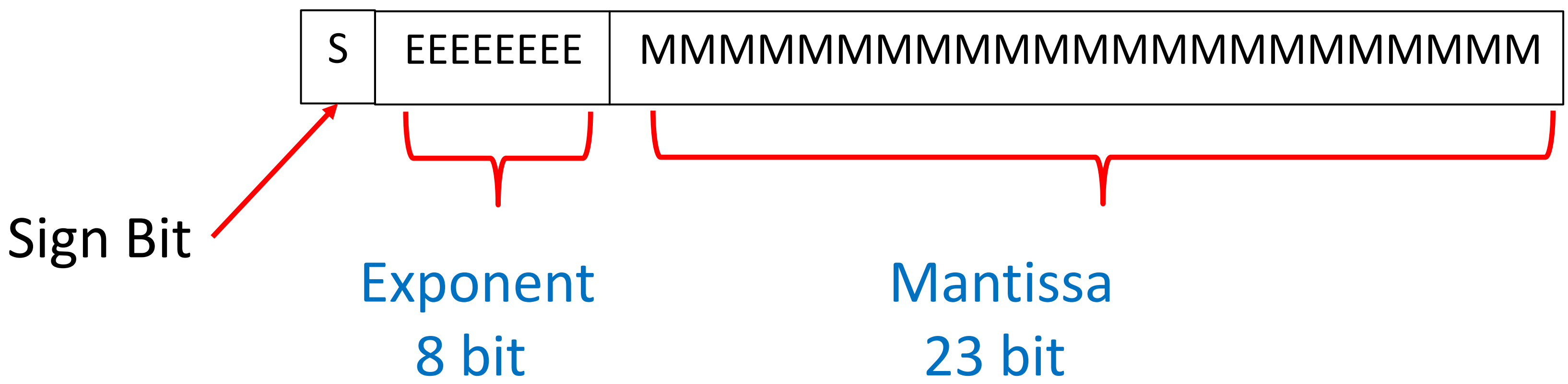
$$v = (-1)^S \times M \times 2^{(E-127)}$$



Floating Point
In Class Problem

- Find the decimal number that corresponds to the floating point bit pattern.

0 11110011 11110010000000000000000



Sign Bit = 0 – Positive Number

NOPRINT

Floating Point
In Class Problem

Find the exponent

0 11110011 11110010000000000000000

0	11110011	MMMMMMMMMMMMMMMMMMMMMMM

Sign Bit

Exponent
8 bit

Mantissa
23 bit

Exponent = $1111011b\ - 127d$

Exponent = $243d\ - 127d = 116d$

NOPRINT

Find the value of the Mantissa

Find the mantissa

0 11110011 11110010000000000000000

0	11110011	11110010000000000000000

Sign Bit

Exponent
8 bit

Mantissa
23 bit

$$M = 1 + m_{22}2^{-1} + m_{21}2^{-2} + m_{20}2^{-3} + \cdots$$

$$M = \ 1 + (1)2^{-1} + (1)2^{-2} + (1)2^{-3} + (1)2^{-4} + (0)2^{-5} + (0)2^{-6} + (1)2^{-7}$$

$$M = \ 1 + \left(\frac{1}{2}\right) + \left(\frac{1}{4}\right) + \left(\frac{1}{8}\right) + \left(\frac{1}{16}\right) + 0 + 0 + \left(\frac{1}{128}\right) = 1.9453125$$

NOPRINT

Putting it all together

- Apply the equation to the component parts

$$S = 0 \qquad M = 1.9453125 \quad E - 127 = 116$$

Sign Bit

Mantissa

Exponent

$$v = (-1)^S \times M \times 2^{(E-127)}$$

$$v = (-1)^0 \times 1.9453125 \times 2^{116}$$

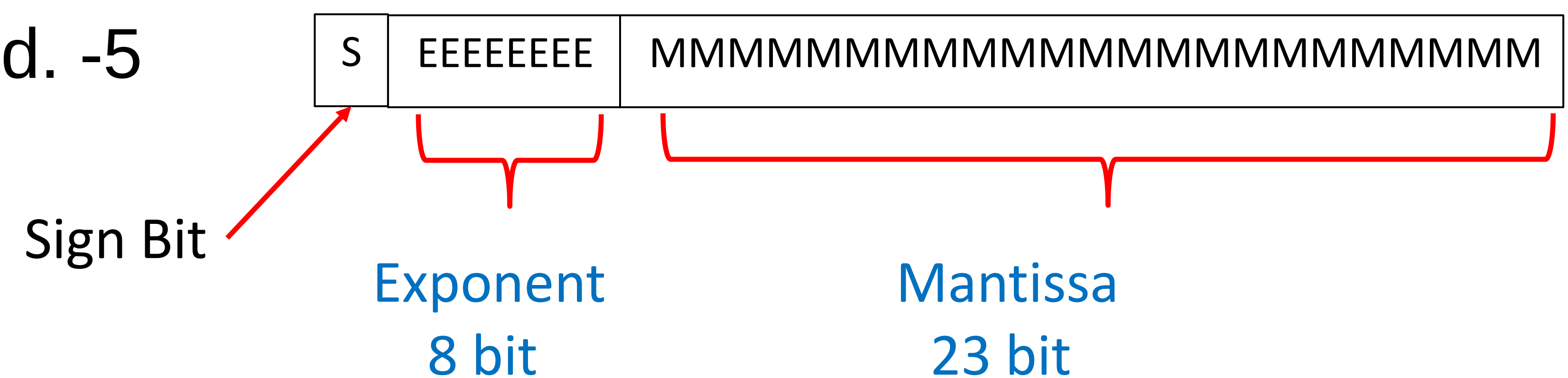
$$v = 1.61610239 \times 10^{35}$$

NOPRINT

Floating Point
In Class Problem

- Convert the following decimal numbers into their IEEE floating point bit patterns:
- a. 1
- b. 2
- c. 4
- d. -5

$$v = (-1)^S \times M \times 2^{(E-127)}$$



Floating Point
In Class Problem

- A. 1

$$v = (-1)^S \times M \times 2^{(E-127)}$$

NOPRINT

$$v = 1 = (-1)^0 \times 1 \times 2^{(0)}$$

Positive Value -- Sign Bit = 0

Total exponent = 0 +127 = 127d = 01111111

Mantissa = $1 = \ 1.m_{22}m_{21}m_{20} \ldots = 1 + \mathrm{m}_{22}\left(\frac{1}{2}\right) + \mathrm{m}_{21}\left(\frac{1}{4}\right) + \cdots$

Mantissa Bits = 00000000000000000000000

0	01111111	00000000000000000000000	= 1

Floating Point
In Class Problem

- B. 2

$$v = (-1)^S \times M \times 2^{(E-127)}$$

NOPRINT

$$v = 2 = (-1)^0 \times 1 \times 2^{(1)}$$

Positive Value -- Sign Bit = 0

Total exponent = 1 +127 = 128d = 10000000

Mantissa = 1 $= \ 1.m_{22}m_{21}m_{20} \ldots = 1 + \mathrm{m}_{22}\left(\frac{1}{2}\right) + \mathrm{m}_{21}\left(\frac{1}{4}\right) + \cdots$

Mantissa Bits = 00000000000000000000000

0	10000000	00000000000000000000000	= 2

Floating Point
In Class Problem

- C. 4

$$v = (-1)^S \times M \times 2^{(E-127)}$$

NOPRINT

$$v = 4 = (-1)^0 \times 1 \times 2^{(2)}$$

Positive Value -- Sign Bit = 0

Total exponent = 2 +127 = 129d = 10000001

Mantissa = 1 $= \ 1.m_{22}m_{21}m_{20} \ldots = 1 + \mathrm{m}_{22}\left(\frac{1}{2}\right) + \mathrm{m}_{21}\left(\frac{1}{4}\right) + \cdots$

Mantissa Bits = 00000000000000000000000

0	10000001	00000000000000000000000	= 4

Floating Point
In Class Problem

- D. -5

$$v = (-1)^S \times M \times 2^{(E-127)}$$

NOPRINT

$$v = -5 = (-1)^1 \times 1.25 \times 2^{(2)}$$

Negative Value -- Sign Bit = 1

Total exponent = 2 +127 = 129d = 10000001

Mantissa = 1.25 $= \ 1.m_{22}m_{21}m_{20} \ldots = 1 + \mathrm{m}_{22}\left(\frac{1}{2}\right) + \mathrm{m}_{21}\left(\frac{1}{4}\right) + \cdots$

Mantissa Bits = 01000000000000000000000

1	10000001	01000000000000000000000	= -5

Number Precision

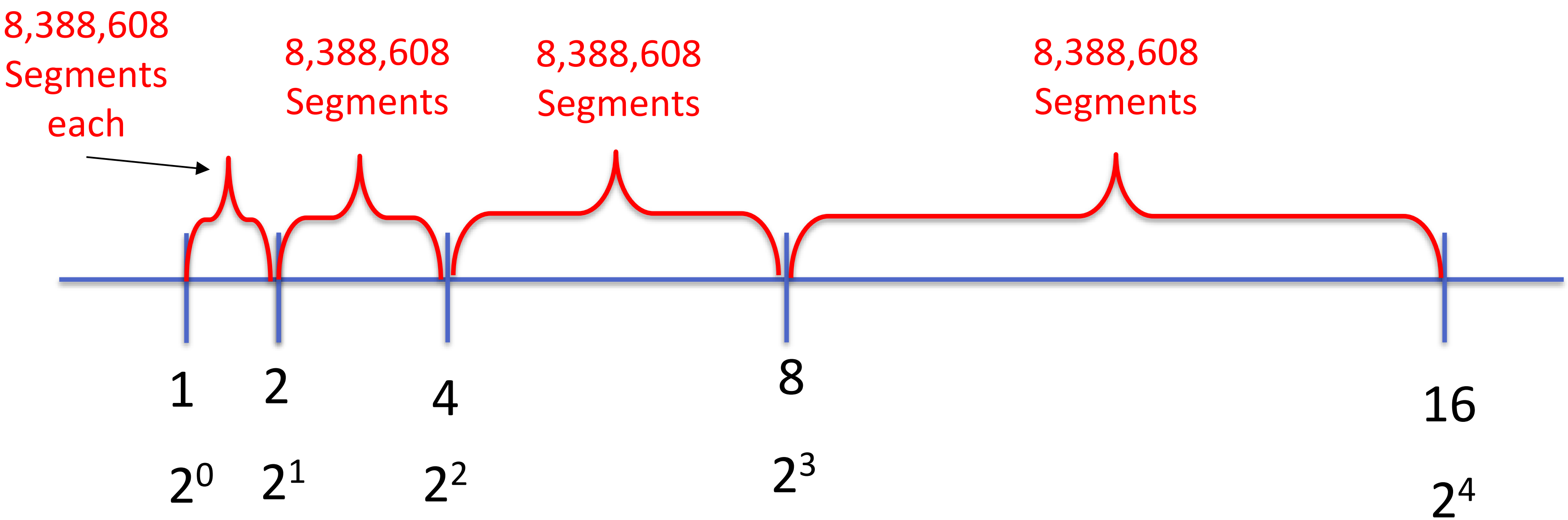
- In integer math, the spacing between numbers is always 1.

- In floating point math, the spacing between numbers varies over the number range.
  - Large numbers have large gaps number to number
  - Small numbers have small gaps number to number

- The spacing between two floating point numbers is about one 10 millionth of the number.

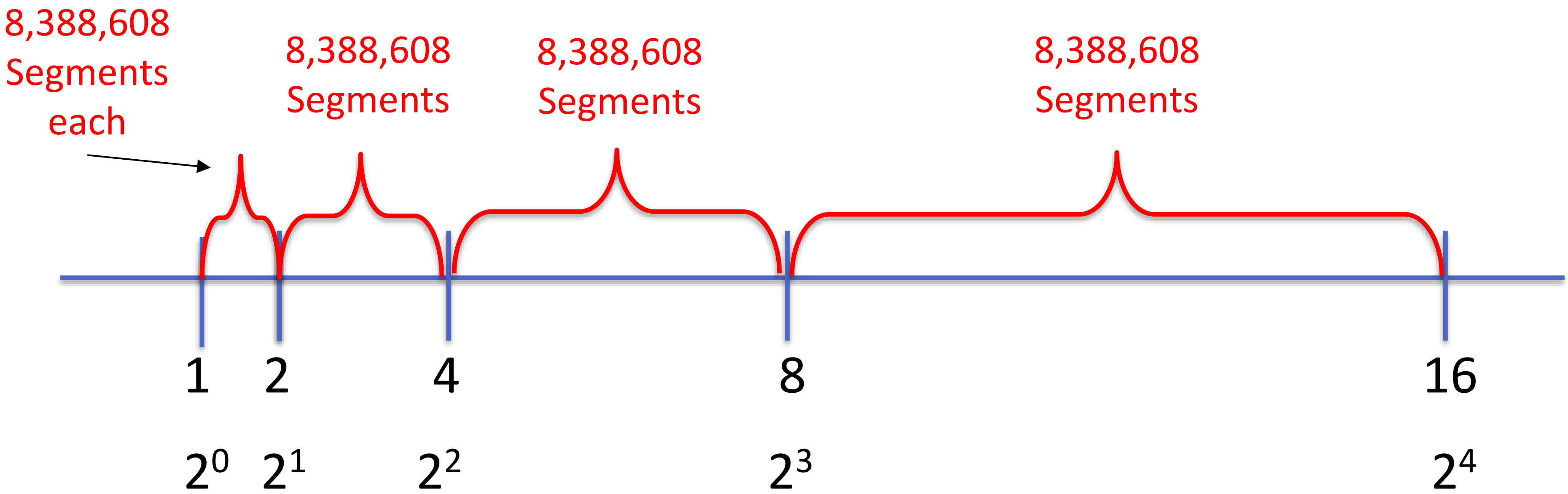
Floating Point Interval Between Numbers

- The mantissa has $2^{23} = 8{,}388{,}608$ unique values
- Each exponential interval is broken into 8,388,608 segments



Floating Point Interval Between Numbers

- The size of each segment becomes larger as the exponent increases.
- The gap between values increases as the exponent increases



Round off Error Due to Finite Precision

- Calculations with floating point numbers can accumulate round-off error.
- Each time a calculation is performed the result must be rounded to the nearest value represented
- The errors associated with finite precision are very similar to quantization errors.

Why is there limited precision?

- Floating point values are represented by 32 bits
- With 32 bits there are $2^{32} = 4.29 \times 10^{9}$ possible values
- However, floating point numbers represent values from $\pm 6.8 \times 10^{38}$ to $\pm 5.9 \times 10^{-39}$

Why is there limited precision?

- Floating point numbers represent values from $\pm 6.8 \times 10^{38}$ to $\pm 5.9 \times 10^{-39}$

- Some values in this range will not be represented

- An <u>approximate</u> value of the round off error is one 40 millionth of the number for each operation.

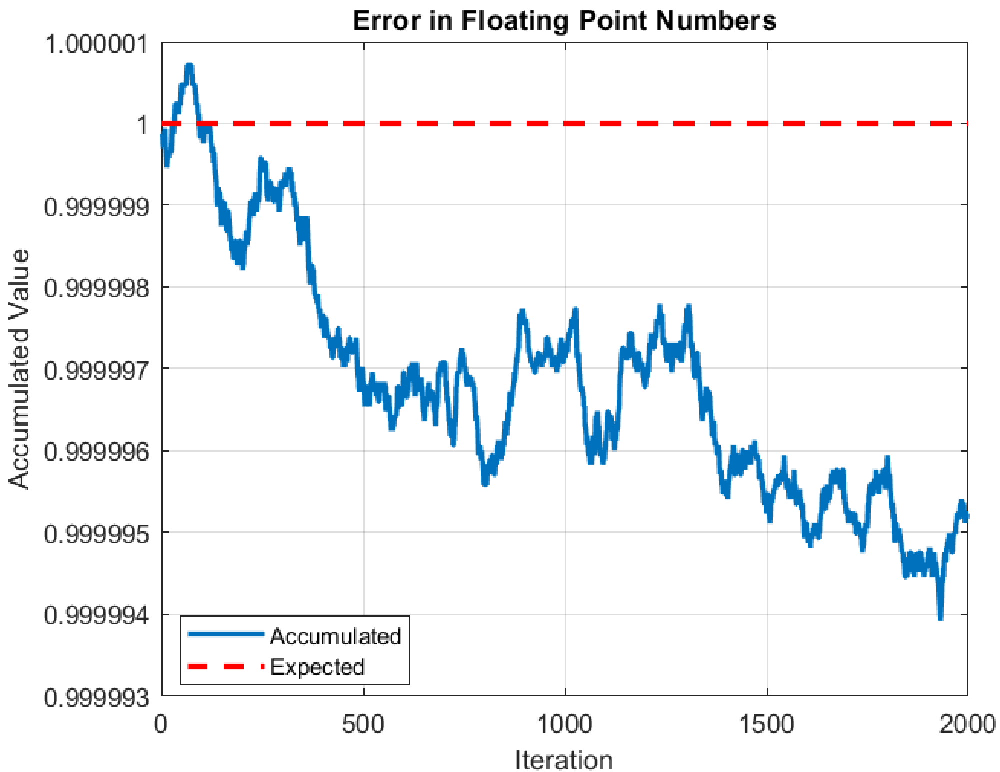
Error Accumulation Example

- Start with the value of 1
- Add a random value A
- Add a random value B
- Subtract A
- Subtract B
- Repeat 2000 times

```
x = single(1);

for i = 1:2000
  a = single( rand );
  b = single( rand );

  %  Add the two random values
  x = x + a;
  x = x + b;

  %  Subtract the two random values
  x = x - a;
  x = x - b;

end
```

Error Accumulation Example

- The value should always be 1
- There is error in each addition and subtraction
- And that error may accumulate depending on the sign



Dealing with Finite Precision

- Use proper typing for variables
  - Loop index should be integer not floating point (termination condition error – see textbook)

- Plan for error
  - Each math operation will result in a round off error of $\approx 1\ in\ 40\ million$.
  - Each number will potentially have that error multiplied by the number of math operations

Dealing with Finite Precision Understand the Number System

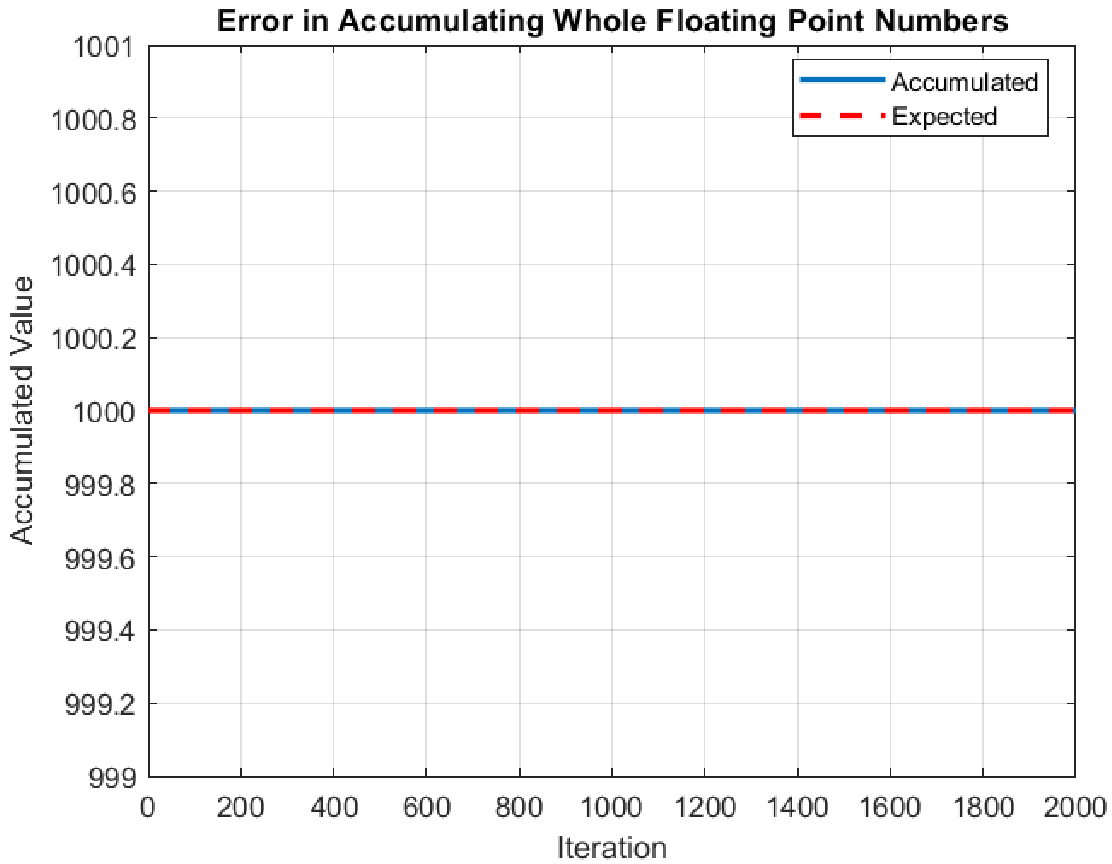
- The entire range of <u>whole</u> numbers are represented exactly

- Values between +/- 16.8 million ($\pm 2^{24}$).

- <u>Whole</u> values staying within this range can be added, subtracted, multiplied without round off error.

Repeat the Round Off Error Test with a Whole Number

- Start with 1000
- Add and subtract a whole number between 0 and 1000



RIT E

Floating Point Dynamic Range

- Dynamic Range is the range of numbers that can be represented
- Floating point numbers have a wide but limited dynamic range.
- The dynamic range is determined by the number of bits in the exponent.

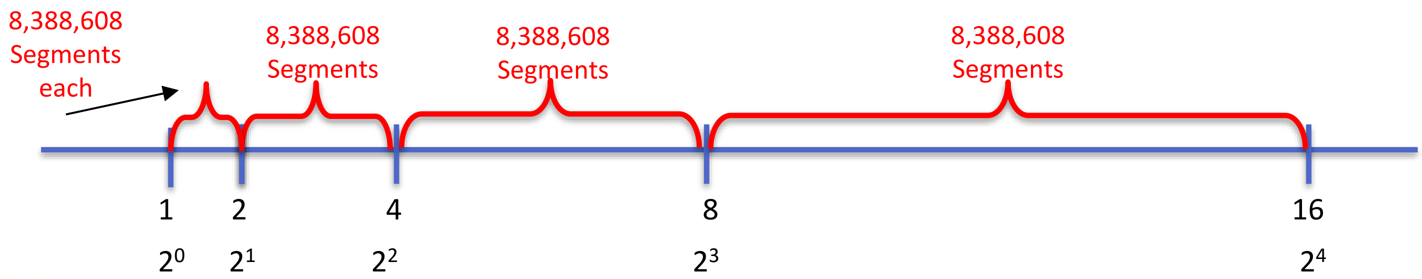
Floating Point Dynamic Range

- For each exponent value we extend the range of values that can be represented by a factor of 2

$$v = (-1)^S \times M \times 2^{(E-127)}$$

- We can express the dynamic range in decibels

$$Dynamic\ Range = 6\ dB \times 2^{N_{exponent\ bits}}$$

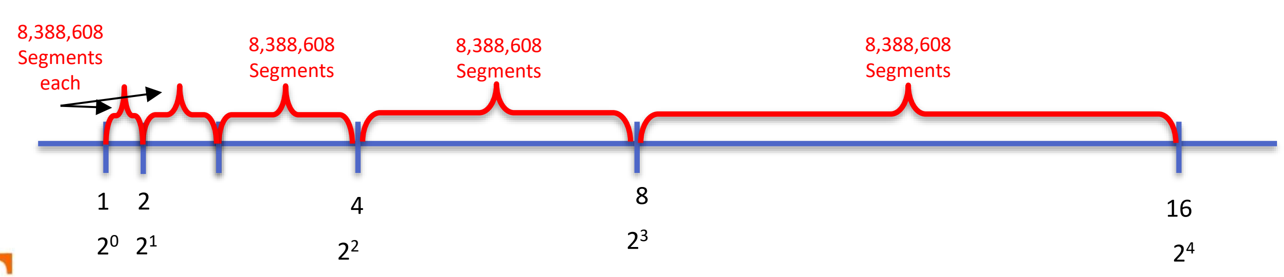


Floating Point Precision and SNR

- Floating point numbers have a finite precision.
- The precision is set by the number of bits in the mantissa and can be expressed as SNR in dB.

$$v = (-1)^S \times M \times 2^{(E-127)}$$

$$SNR = 6dB \times N_{mantissa\ bits}$$



Dynamic Range and SNR for 32 Bit IEEE Floating Point Numbers

- Number of exponent bits = 8+1 for the sign bit
- Number of mantissa bits = 23

$$Dynamic\ Range = 6\ dB \times 2^{N_{exponent\ bits}}$$

$$Dynamic\ Range = 6\ dB \times 2^{8} = 1536\ dB$$

$$SNR = 6dB \times N_{mantissa\ bits}$$

$$SNR = 6dB \times 23 = 138\ dB$$

Round-off Error Computation Example

- In a digital filter, each sample of the output signal is found by multiplying M samples from the input signal by M coefficients and adding the products.

- For this example assume M = 5000, and that single precision floating point math is used.

Example Round-off Error Computation

- How many math operations (# of multiplications plus # additions) need to be conducted to calculate each point in the output signal?

- Approximately 5000 multiplies and 5000 additions for a total of 10,000 operations.

Round-off Error Computation Example

- If the output signal has an average amplitude of about 100, what is the expected error on an individual output sample?
  - Assume that the round-off errors combine by addition.
- Each math operation results in an error of ~1/40 million of the value, so as a fraction of the average value

$$\frac{1}{40\times10^{6}} * 10{,}00\ operations = .00025$$

Round-off Error Computation Example

- Each math operation results in an error of 1/40 million of the value, so as a fraction of the average value

$$\frac{1}{40\times10^{6}} * 10{,}000\ operations = .00025$$

- Then because the average value is 100

$$Error = V_{ave} * .00025 = .025$$

- The approximate error in the output value for each filtering operation is .025

Why All the Attention to Number Formats?

- In the lab, you will be working with a variety of number formats, e.g. short, integer, float, double.
- It is common to get confused if you try to plot a number as a float when it is a 32-bit integer.
- The resulting plot looks like noise. It is important to know exactly how the bit pattern is interpreted
- Pay attention when you are plotting variables in the DSP memory.

Summary

- Floating point numbers can be represented with sign, mantissa and exponent values
- Round off error is created when using floating point numbers for math operations due to their finite precision
- Round off error is a type of noise.

Summary

- Floating point numbers have a huge dynamic range (i.e. the ratio between the biggest number and the smallest number)

- Fixed point numbers have a limited dynamic range but allow for faster processing and cheaper chips.