Floating Point Tutorial

In single precision floating point you have:

- A sign (+ or -), represented by 1 bit
- The exponent, represented by 8 bits
- The fraction, represented by 23 bits

1 bit	8 bits	23 bits
sign	**Biased exponent**	**fraction**

So altogether you represent a single precision floating point number in 32 bits.

Example: 25.75 (from slides)

1. **Convert number into “hybrid” binary.**
   Convert 25 into binary → 11001
   Convert .75 into binary… here’s a little trick:

	Trick	Bits
1. Take the decimal (.75) and multiply by 2	0.75 x 2	
2. If you get a 1, this bit is a 1. If you get a 0, this bit is a 0	**1**.50	1
3. Take the decimal part of 1.50 (which is 0.50) and multiply by 2	0.50 x 2	
4. When you have all 0’s for the decimal part, you’re done	1.**00**	1

   So now, 0.75 in binary is 11. You can fill in zeros after the “11”, so it can be “110” or “11000”.
   The complete hybrid binary is: 11001.110
2. **Float the binary point.**
   You need to normalize the mantissa to 1.1001110. Shift the radix point to have **only one 1** to the left of the radix.
   Ex 1: 1110.0001 → 1.1100001 **Ex 2**: 00.111001001 → 1.11001001
   To make the number 1.1001110, you had to move the radix point **4 places to the left**.

When you do so, you need to keep track of how many places you moved it, this is represented in an exponent: $\mathbf{2^4}$. Remember, this is in binary so the base is 2 NOT 10. Now, our value is **1.1001110×2^4**

3. **Floating point representation:**
   Now you need to fill in the fields.

   a. **Sign**: the number initially was positive, so we represent the sign as 0

sign	exponent	fraction
0		

   b. **Exponent:** the value is 4, but the rule of filling in this field is to **add 127 to the initial exponent value** to create the *biased* exponent. 4 + 127 = 131

sign	exponent	fraction
0	**1000 0011**	

   c. **Fraction**: the numbers right of the radix point is the fraction. In the value 1.**1001110**, we care about the 1001110. We then extend the value (add 0's to the right) to 23 bits.

sign	exponent	fraction
0	1000 0011	**1001 1100 0000 0000 0000 000**

Our final representation is 0 1000 0011 1001 1100 0000 0000 0000 000

Example: 1.125

	Trick	Bits
	0.125 x 2	
Whole number is a 0, so bit is a 0	**0**.250	0
	0.250 x 2	
Whole number is a 0, so bit is a 0	**0**.500	0
	0.500 x 2	
Whole number is a 1, so bit is a 1 Decimal is a 0, so done.	**1.000**	1

“Hybrid” binary → 1.001

Example: 2.625

	Trick	Bits
	0.625 x 2	
Whole number is a 1, so bit is a 1	**1**.250	1
	0.250 x 2	
Whole number is a 0, so bit is a 0	**0**.500	0
	0.500 x 2	
Whole number is a 1, so bit is a 1 Decimal is a 0, so done.	**1.000**	1

“Hybrid” binary → 10.101