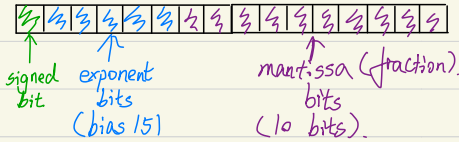
- Due on April 9 23:55pm
- 3.27, 3.29, 3.30.

4/7/2020

张佳宸

3.27 [20] <§3.5> IEEE 754-2008 contains a half precision that is only 16 bits wide. The leftmost bit is still the sign bit, the exponent is 5 bits wide and has a bias of 15, and the mantissa is 10 bits long. A hidden 1 is assumed. Write down the bit pattern to represent -1.5625×10^{-1} assuming a version of this format, which uses an excess-16 format to store the exponent. Comment on how the range and accuracy of this 16-bit floating point format compares to the single precision IEEE 754 standard.



Decimal: $-1.5625 \times 10^{-1} = -0.15625$ $(-1)^1 = -1$ the signed bit is 1

$(0.15625)_{10} = (0.00101)_2$.

Binary: $0.00101 = 1.01 \times 2^{-3}$

The exponent is -3, as the bias is 15, the exponent bits is 01100

And the mantissa is 0100000000

⇒ The required bit pattern is 1 01100 0100000000

As the exponent bits 00000 and 11111 reserved.

1° smallest value

Exponent bits: 00001. Actual exponent is $(1-15) = -14$

Mantissa: 0000000000 ⇒ Significand: 1.0

⇒ $\pm 1.0 \times 2^{-14} \approx \pm 6.1 \times 10^{-5}$

2° largest value

Exponent bit: 11110. Actual exponent is $(30-15) = 15$

Mantissa: 1111111111 ⇒ significand ≈ 2.0

⇒ $\pm 2.0 \times 2^{15} = \pm 6.5536 \times 10^{4}$

⇒ Accuracy: $\frac{\Delta A}{|A|} = \frac{2^{-10} \times 2^{exp}}{1 \times 2^{exp}} = 2^{-10}$

when equivalent to $10 \lg 2 \approx 3$ decimal digits of precision.

		16-bit floating point format	Single precision
Range	Smallest	$\pm 1.0 \times 2^{-14} \approx \pm 6.1 \times 10^{-5}$	$\pm 1.0 \times 2^{-126} \approx \pm 1.2 \times 10^{-38}$
	Largest	$\approx \pm 2.0 \times 2^{15} = \pm 65536$	$\approx \pm 2.0 \times 2^{127} \approx \pm 3.4 \times 10^{38}$
Accuracy		2^{-10} / 3 decimal digits	2^{-23} / ≈6 decimal digits

3.29 [20] <§3.5> Calculate the sum of 2.6125×10^{1} and $4.150390625 \times 10^{-1}$ by hand, assuming A and B are stored in the 16-bit half precision described in Exercise 3.27. Assume 1 guard, 1 round bit, and 1 sticky bit, and round to the nearest even. Show all the steps.

Decimal: $2.6125 \times 10^{1} = 26.125$ >0, the signed bit is 0

$(26.125)_{10} = (11010.001)_2$

Binary: $11010.001 = 1.1010001 \times 2^{4}$ the mantissa is 1010001000

The exponent is 4, as the bias is 15, the exponent bit is 10011

The 16-bit half precision of $A = 0\,10011\,1010001000$

Decimal: $4.150390625 \times 10^{-1} = 0.4150390625$ >0 signed bit is 0

$(0.4150390625) = (0.01101001)_2$

Binary: $0.01101001 = 1.1010010 \times 2^{-2}$ the mantissa is 1010010000

The exponent is -2, as the bias is 15, the exponent bit is 01101

The 16-bit half precision of $B = 0\,01101\,1010010000$

$A: 1.1010001 \times 2^{4}$ $B = 1.1010001 \times 2^{-2}$

1. Align binary points

$1.1010001 \times 2^{-2} = 0.0000011010001 \times 2^{4}$

2. Add significands

$1.1010001 \times 2^{4} + 0.0000011010001 \times 2^{4} = 1.1010100010001 \times 2^{4}$

3. Normalize result & check for over/underflow

• $1.1010100010001 \times 2^{4}$, with no overflow/underflow.

4. Round and renormalize if necessary.

1.1010100010001

Guard, Round Bit, Sticky Bit

Guard is 0, RS is 10, after round to nearest even

finally get 1.1010100010 and it's normalized.

So the result is $1.1010100010 \times 2^{4}$ $0\,10011\,1010100010$

3.30 [30] <§3.5> Calculate the product of -8.0546875×10^{0} and $-1.7993164625 \times 10^{-1}$ by hand, assuming A and B are stored in the 16-bit half precision format described in Exercise 3.27. Assume 1 guard, 1 round bit, and 1 sticky bit, and round to the nearest even. Show all the steps; however, as is done in the example in the text, you can do the multiplication in human-readable format instead of using the techniques described in Exercises 3.12 through 3.14. Indicate if there is overflow or underflow. Write your answer in both the 16-bit floating point format described in Exercise 3.27 and also as a decimal number. How accurate is your result? How does it compare to the number you get if you do the multiplication on a calculator?

Decimal: $-8.0546875 \times 10^{0} = -8.0546875$.

$(-8.0546875)_{10} = (-1000.0000111)_2$

Binary: $-1000.0000111 = -1.0000000111 \times 2^{3}$

Decimal: $-1.7993164625 \times 10^{-1} = -0.17993164625$

$(-0.17993164625)_{10} = (-0.00101111000001)_2$

Binary: $-0.00101111000001 = -1.0111100001 \times 2^{-3}$

Step 1. Adding the exponents without bias:

$3 + (-3) = 0$

1.0111100010
$\times\ 1.0000000111$
10111100010
10111100010
10111100010
1011110000110

Step 2. Multiplying the significands:

1.0111100010
1.0111100110000101001110

The product is 1.0111001100010100111

We need to keep to 12 bits (10 bits + 1 round bit + 1 sticky bit)

1.0111001100010100111

G R S

The result is $1.0111001100010 \times 2^{0}$

Step 3. The result is normalized and the exponent, which is 0, is not overflow or underflow.

Step 4. Rounding the product

RS = 01, Truncate the result by discarding RS.

The result is $1.0111001100 \times 2^{0}$

Step 5. Since the signs of the original operands are same. make the sign of the product positive.

Hence, the product is $1.0111001100 \times 2^{0}$

Converting to decimal to check our result:

$(1.0111001100)_2 \times 2^{0} = (1.0111001100)_2$

$= 1 + 115/2^{8} = 1.44921875$

The accuracy of my result is 2^{-10}, which is about 3 decimal digits.

Compared with the result calculated by calculator, the difference is less than 0.0001.