HW 3

Problem 1

How would you test for overflow, the result of an addition of two 8-bit operands if the operands were

- unsigned
- signed with 2s complement representation.

Add the following 8-bit strings assuming they are *(i) unsigned (ii) signed* and represented using 2’s complement. Indicate which of these additions overflow.

A. 0110 1110 + 1001 1111

B. 1111 1111 + 0000 0001

C. 1000 0000 + 0111 1111

D. 0111 0001 + 0000 1111

Problem 2

One possible performance enhancement is to do a shift and add instead of an actual multiplication. Since *9×6*, for example, can be written *(2×2×2+1)×6*, we can calculate *9×6* by shifting *6* to the left three times and then adding *6* to that result. Show the best way to calculate $0\mathrm{xAB}_{\mathrm{hex}} \times 0\mathrm{xEF}_{\mathrm{hex}}$ using shifts and adds/subtracts. Assume both inputs are 8-bit unsigned integers

Problem 3

What decimal number does the 32-bit pattern *0×DEADBEEF* represent if it is a floating-point number? Use the IEEE 754 standard

Problem 4

Write down the binary representation of the decimal number 78.75 assuming the IEEE 754 single precision format. Write down the binary representation of the decimal number 78.75 assuming the IEEE 754 double precision format

Problem 5

Write down the binary representation of the decimal number 78.75 assuming it was stored using the single precision IBM format (base 16, instead of base 2, with 7 bits of exponent).

Problem 6

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 (fractional field) is 10 bits long. A hidden 1 is assumed.

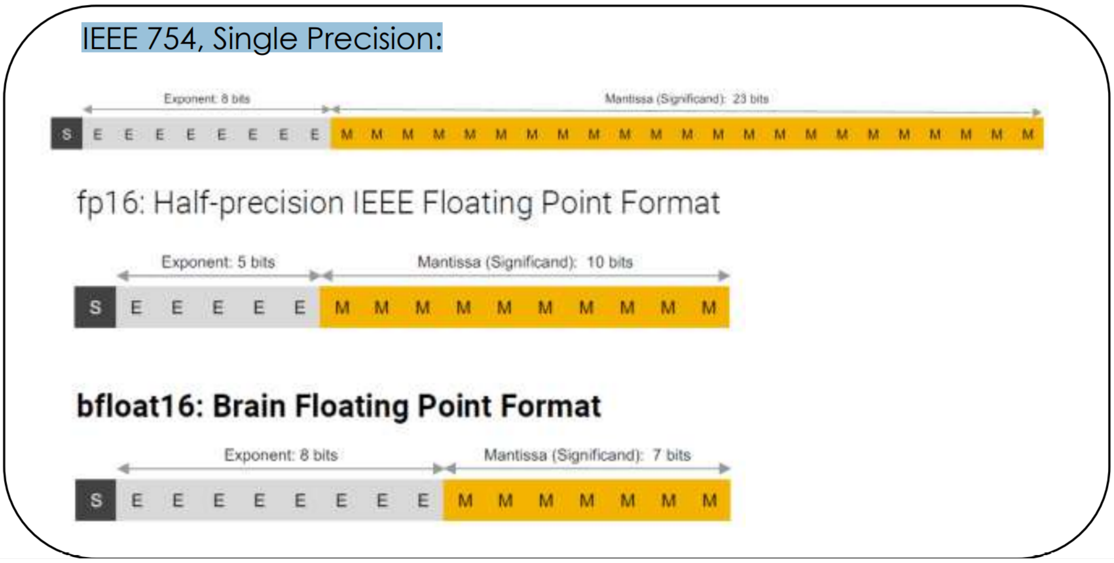
- Write down the bit pattern to represent −1.3625 ×10−1 Comment on how the range and accuracy of this 16-bit floating point format compares to the single precision IEEE 754 standard.
- Calculate the sum of 1.6125×10^{1} (A) and $3.150390625 \times 10^{-1}$ (B) by hand, assuming operands A and B are stored in the 16- bit half precision described in problem a. above

Assume 1 guard, 1 round bit, and 1 sticky bit, and round to the nearest even. Show all the steps

Problem 7

What is the range of representation and relative accuracy of positive numbers for the following 3 formats:

- IEEE 754 Single Precision
- IEEE 754 – 2008 (described in Problem 6 above) and
- 'bfloat16' shown in the figure below



Problem 8

Suppose we have a 7-bit computer that uses IEEE floating-point arithmetic where a floating point number has 1 sign bit, 3 exponent bits, and 3 fraction bits. All of the bits in the hardware work properly. Recall that denormalized numbers will have an exponent of 000, and the bias for a 3- bit exponent is

$2^{3-1} - 1 = 3$

- For each of the following, write the *binary value* and *the corresponding decimal value* of the $\underline{\text{7-bit floating point number that is the closest available representation of the requested number}}$. If rounding is necessary use round-to-nearest. Give the decimal values either as whole numbers or fractions. The first few lines are filled in for you.

Number	Binary	Decimal
0	0 000 000	0.0
-0.125	1 000 000	-0.125
Smallest positive normalized number		
largest positive normalized number		
Smallest positive denormalized number > 0		
largest positive denormalized number > 0		

- The associative law for addition says that $a+(b+c)=(a+b)+c$. This holds for regular arithmetic, but does not always hold for floating-point numbers. Using the 7-bit floating-point system described above, give an example of three floating-point numbers a, b, and c for which the associative law does not hold, and show why the law does not hold for those three numbers.