Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 1 Homework

Assignment:

3.1 thru 3.8

3.9 thru 3.11 (151 and 214 as 8-bit decimal numbers in the 2's complement format have negative values)

3.32 thru 3.34

3.1 [5] <§3.2> What is 5ED4 - 07A4 when these values represent unsigned 16-bit hexadecimal numbers? The result should be written in hexadecimal. Show your work.

 5ED4
-07A4 =
______
 5730

3.2 [5] <§3.2> What is 5ED4 - 07A4 when these values represent signed 16-bit hexadecimal numbers stored in sign-magnitude format? The result should be written in hexadecimal. Show your work.

5ED4 = 0101111011010100
07A4 = 0000011110100100

 0101111011010100
-0000011110100100=
____
 0101011100110000 = 5730

3.3 [10] <§3.2> Convert 5ED4 into a binary number. What makes base 16 (hexadecimal) an attractive numbering system for representing values in computers?

5ED4 = 0101111011010100

Hexadecimal (base 16) is an attractive numbering system for computers because it takes advantage of the fact that 16 is 2^4. Using base 16 is the most space efficient way to store 4 bits of information. 16 is also a nicer human readable format than say, 32, since 32 would require 0123456789ABCDEFGHIJKLMNOP notation.

3.4 [5] <§3.2> What is 4365 - 3412 when these values represent unsigned 12-bit octal numbers? The result should be written in octal. Show your work.

 4365
-3412 =
______
 0753

3.5 [5] <§3.2> What is 4365 - 3412 when these values represent signed 12-bit octal numbers stored in sign-magnitude format? The result should be written in octal. Show your work.

4365 = 100011110101 (negative)
3412 = 011100001010

 100011110101
-011100001010 =
______________ (subtracting negatives is analogous to adding positives)
 111111111111

3.6 [5] <§3.2> Assume 185 and 122 are unsigned 8-bit decimal integers. Calculate 185 – 122. Is there overflow, underflow, or neither?

 185
-122 =
______
 063

This is neither underflow or overflow

3.7 [5] <§3.2> Assume 185 and 122 are signed 8-bit decimal integers stored in sign-magnitude format. Calculate 185 + 122. Is there overflow, underflow, or neither?

185 = 10111001 = -071
122 = 01111010 = 122

 -071
+122 =
_____
  051

Neither overflow or underflow, value within range of -128 to 127

3.8 [5] <§3.2> Assume 185 and 122 are signed 8-bit decimal integers stored in sign-magnitude format. Calculate 185 - 122. Is there overflow, underflow, or neither?

-071
-122 =
______
-193

This value is outside of the range of -128 to 127, therefore there is underflow

3.9 [10] <§3.2> Assume 151 and 214 are signed 8-bit decimal integers stored in two’s complement format. Calculate 151 + 214 using saturating arithmetic. The result should be written in decimal. Show your work.

151 = 10010111

Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 1 Homework

214 = 11010110

-10010111
-11010110 =
____
101101101

This is underflow. Using saturated arithmetic, this is held at the highest decimal value, 255

3.10 [10] <§3.2> Assume 151 and 214 are signed 8-bit decimal integers stored in two's complement format. Calculate 151 - 214 using saturating arithmetic. The result should be written in decimal. Show your work.

151 = 10010111
214 = 11010110

 -10010111
+11010110 =

  11010110
 -10010111 =
____
   10101011 = -85

3.11 [10] <§3.2> Assume 151 and 214 are unsigned 8-bit integers. Calculate 151 + 214 using saturating arithmetic. The result should be written in decimal. Show your work.

151 = 10010111
214 = 11010110

  10010111
+11010110 =
____
101101101

This is overflow. Using saturated arithmetic, this has the decimal value of 255

3.32 [20] <§3.9> Calculate (3.984375 x 10_{-1} + 3.4375 x 10_{-1}) + 1.771 x 10_3 by hand, assuming each of the values are stored in the 16-bit half precision format described in Exercise 3.27 (and also described in the text). Assume 1 guard, 1 round bit, and 1 sticky bit, and round to the nearest even. Show all the steps, and write your answer in both the 16-bit floating point format and in decimal.

.3984375 = 51/128 = 0110011/2^7 = 1.10011 x 2^{-2}
.34375 = 11/32 = 01011/2^5 = 1.011 x 2^{-2}
1771 = 1771/2^0 = 11011101011 x 2^0 = 1.1011101011 x 2^{10}

Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 1 Homework

1.10011×2^{-2}
$+1.011 \quad \times 2^{-2} =$

$10.11111 \times 2^{-2} = 1.011111 \times 2^{-1} = 0.0000000001 \times 2^{10}$

$0.0000000001 \times 2^{10}$
$+1.1011101011 \times 2^{10} =$

$1.1011101100 \times 2^{10}$

$= 1772 = 1.772 \times 10^{3} = 0\ 01010\ 1011101100$

3.33 [20] <§3.9> Calculate $3.984375 \times 10_{-1} + (3.4375 \times 10_{-1} + 1.771 \times 10_{3})$ by hand, assuming each of the values are stored in the 16-bit half precision format described in Exercise 3.27 (and also described in the text). Assume 1 guard, 1 round bit, and 1 sticky bit, and round to the nearest even. Show all the steps, and write your answer in both the 16-bit floating point format and in decimal.

$.3984375 = 51/128 = 0110011/2^{7} = 1.10011 \times 2^{-2}$
$.34375 = 11/32 = 01011/2^{5} = 1.011 \times 2^{-2}$
$1771 = 1771/2^{0} = 11011101011 \times 2^{0} = 1.1011101011 \times 2^{10}$

$1.011 \times 2^{-2} = 0.0000000000 \times 2^{10}$

$0.0000000000 \times 2^{10}$
$+1.1011101011 \times 2^{10} =$

$1.1011101011 \times 2^{10}$

$1.10011 \times 2^{-2} = 0.0000000000 \times 2^{10}$

$0.0000000000 \times 2^{10}$
$+1.1011101011 \times 2^{10} =$

$1.1011101011 \times 2^{10}$

$= 1771 = 1.771 \times 10^{3} = 0\ 01010\ 1011101011$

3.34 [10] <§3.9> Based on your answers to 3.32 and 3.33, does $(3.984375 \times 10_{-1} + 3.4375 \times 10_{-1}) + 1.771 \times 10_{3} = 3.984375 \times 10_{-1} + (3.4375 \times 10_{-1} + 1.771 \times 10_{3})$?

No, the answers to not match. Floating point addition is not necessarily associative. This is because floating point values are often just close approximations to the real value.