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.

5730

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.

753

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

Neither (63)

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?

Neither (65)

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?

Overflow (result = −179, which does not fit into an SM 8-bit format)

3.20 [5] <§3.5> What decimal number does the bit pattern 0×0C000000 represent if it is a two's complement integer? An unsigned integer?

201326592 in both cases.

3.22 [10] <§3.5> What decimal number does the bit pattern 0×0C000000 represent if it is a floating point number? Use the IEEE 754 standard.

0×0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000

= 0 0001 1000 0000 0000 0000 0000 0000 000

sign is positive

exp = 0×18 = 24 − 127 = −103

there is a hidden 1

mantissa = 0

answer = 1.0×2^{-103}

3.23 [10] <§3.5> Write down the binary representation of the decimal number 63.25 assuming the IEEE 754 single precision format.

$63.25 \times 10^0 = 111111.01 \times 2^0$

normalize, move binary point 5 to the left

1.1111101×2^5

sign = positive, exp = 127+5=132

Final bit pattern: 0 1000 0100 1111 1010 0000 0000 0000 000

= 0100 0010 0111 1101 0000 0000 0000 0000 = 0x427D0000

3.24 [10] <§3.5> Write down the binary representation of the decimal number 63.25 assuming the IEEE 754 double precision format.

$63.25 \times 10^0 = 111111.01 \times 2^0$

normalize, move binary point 5 to the left

1.1111101×2^5

sign = positive, exp = 1023+5=1028

Final bit pattern:

0 100 0000 0100 1111 1010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000
= 0x404FA00000000000

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.

$-1.5625 \times 10^{-1} = -.15625 \times 10^0$

$= -.00101 \times 2^0$

move the binary point 3 to the right, $= -1.01 \times 2^{-3}$

exponent = −3 = −3+15 = 12, fraction = −.0100000000

answer: 1011000100000000

3.41 [10] <§3.5> Using the IEEE 754 floating point format, write down the bit pattern that would represent −1/4. Can you represent −1/4 exactly?

Answer	sign	exp	Exact?
1 01111101 00000000000000000000000	–	–2	Yes