Homework 2

(Due date: February 9th @ 5:30 pm)
Presentation and clarity are very important!

Problem 1 (10 pts)

- Complete the following table. Use the fewest number of bits in each case.
  You MUST show your conversion procedure. **No procedure ≡ zero points**.

REPRESENTATION			
Decimal	**Sign-and-magnitude**	**1's complement**	**2's complement**
-129			
			1010010
		010011010	
			1000000
	1001010		

Problem 2 (16 pts)

a) Perform the following additions and subtractions of the following unsigned integers. Use the fewest number of bits n to represent both operators. Indicate every carry (or borrow) from c_0 to c_n (or b_0 to b_n). For the addition, determine whether there is an overflow. For the subtraction, determine whether we need to keep borrowing from a higher bit. (6 pts)

Example (n=8):

✓ 54 + 210

	c_8=1	c_7=1	c_6=1	c_5=1	c_4=0	c_3=1	c_2=1	c_1=0	c_0=0
54 = 0x36 =		0	0	1	1	0	1	1	0 +
210 = 0xD2 =		1	1	0	1	0	0	1	0
Overflow! →	1	0	0	0	0	1	0	0	0

✓ 77 - 194

	Borrow out! →	b_8=1	b_7=0	b_6=0	b_5=0	b_4=0	b_3=0	b_2=1	b_1=0	b_0=0
77 = 0x4D =			0	1	0	0	1	1	0	1 -
194 = 0xC2 =			1	1	0	0	0	0	1	0
			0	0	0	0	1	0	1	1

✓ 244 + 267
✓ 39 + 218
✓ 251 - 126
✓ 169 - 201

b) Perform the following operations, where numbers are represented in 2's complement arithmetic: (10 pts)

✓ -70 + 63
✓ -257 + 256
✓ 490 + 47
✓ -127 - 183

- For each case:
  - ✓ Determine the minimum number of bits required to represent both summands. You might need to sign-extend one of the summands, since for proper summation, both summands must have the same number of bits.
  - ✓ Perform the binary addition in 2's complement arithmetic. The result must have the same number of bits as the summands.
  - ✓ Determine whether there is overflow by:
    i. Using c_n, c_{n-1} (carries).
    ii. Performing the operation in the decimal system and checking whether the result is within the allowed range for n bits, where n is the minimum number of bits for the summands.
  - ✓ If there is overflow and we want to avoid it, what is the minimum number of bits required to represent both the summands and the result?

Problem 3 (27 pts)

a) Calculate the result of the additions and subtractions for the following signed fixed-point numbers. Use the minimum number of bits for both operands and result so that overflow is avoided. (6 pts.)

0.1110 + 1.010111	11.11101 - 0.001101	1.0001 + 1.001001	0.00101 - 101.01101

b) Calculate the result of the multiplication of the following signed fixed-point numbers: (9 pts.)

10.0101 × 0.10111	101.1101 × 110.011	01.101 × 100.01011

c) Calculate the division result (with $x = 4$ fractional bits) for the following signed fixed-point numbers: (12 pts.)

10.0101 ÷ 01.011	01.0111 ÷ 01.11	01.01110 ÷ 1.011	1.1101 ÷ 10.001

PROBLEM 4 (10 PTS)

a) We want to represent numbers between -128.7 and 179. What is the fixed point format that requires the fewest number of bits for a resolution better or equal than 0.0005? (3 pts.)

b) We want to represent numbers between -255.9 and 234.5. What is the fixed point format that requires the fewest number of bits for a resolution better or equal than 0.0025? (3 pts.)

c) Represent these numbers in Fixed Point Arithmetic (signed numbers). For each case, select the minimum number of bits

-127.3125	232.21875

PROBLEM 5 (9 PTS)

a) Complete the table for the following fixed point formats (signed numbers):

Fractional bits	Integer Bits	FX Format	Range	Dynamic Range (dB)	Resolution
7	5				
12	4				
17	7				

b) Complete the table for the following floating point formats (which resemble the IEEE-754 standard) with 16, 24, 48 bits. Only consider ordinary numbers.

Exponent bits (E)	Significant bits (p)	Min	Max	Range of e	Range of significand
6	9				
7	16				
10	37				

PROBLEM 6 (28 PTS)

a) Calculate the decimal values of the following floating point numbers represented as hexadecimals. Show your procedure.

Single (32 bits)		Double (64 bits)	
✓ F8000378	✓ 7FFCDEAC	✓ 8009DECADE080000	✓ 7FF0000000000000
✓ 801DECAF	✓ B300D959	✓ FFFDECAFC0FFEE90	✓ FACADEDECADE1990

b) Calculate the result of the following operations with 32-bit floating point numbers. Truncate the results when required. When doing fixed-point division, use 8 fractional bits. Show your procedure. (20 pts.)

✓ 40B00000 + C2FA8000	✓ 10DAD000 – 90FAD000	✓ 7AB80000 × 81800000	✓ FA390000 ÷ 48400000
✓ 42FA8000 + C0E00000	✓ 3DE38866 – B300D959	✓ FA09D300 × 4D080000	✓ FF800000 ÷ 09FE0090