CSCI 2150 – Binary Representations Homework

Name: ________________________________

1. Fill in the blank cells of the table below with the correct numeric format. ***For cells representing binary values, only 8-bit values are allowed!*** If a value for a cell is invalid or cannot be represented in that format, write "X".

Decimal	2's complement binary	Signed magnitude binary	Unsigned binary
101	**01100101**	**01100101**	**01100101**
-13	**11110011**	**10001101**	**X**
109	**01101101**	**01101101**	**01101101**
130	**X**	**X**	**10000010**

First row:

- First, note that the most significant digit of the 2's complement value is 0. This means that it is a positive value and that ALL three binary representations will be identical. Therefore, the answer is 01100101_2 for signed magnitude and unsigned binary.
- Decimal: Since we have a positive value, simply add the powers of 2 for each position that has a 1 in the unsigned binary representation: $2^6 + 2^5 + 2^2 + 2^0 = 64 + 32 + 4 + 1 = 101_{10}$.

Second row:

- Unsigned binary: Since the MSB of the signed magnitude value is a one, it is a negative number and not capable of being represented in unsigned binary. Put an X there.
- Decimal: Since the value is negative, we need to first convert the signed magnitude representation to its corresponding positive value to determine the magnitude. This is done by flipping the MSB from a 1 to a zero. This gives us 00001101_2. Converting this to decimal gives us $2^3 + 2^2 + 2^0 = 8 + 4 + 1 = 13_{10}$. Therefore, the decimal value is -13_{10}.
- To compute the 2's complement value, take the positive binary representation of 13_{10} which is 00001101 and do the "bit flippy thing." Starting on the right side, copy all of the zeros up to and including the first 1 you come to. Well, the first bit you come to on the right side is a 1, so copy only it down. Invert the remaining bits. This gives us 11110011_2.

Third row:

- I'm not sure what I was thinking (or not thinking) when I put together this homework, but this line is very much like the top row. The MSB of the unsigned value (which must be positive in the first placed) is zero, which means that it is small enough to be represented with any of the signed representations, i.e., it is less than $2^7 - 1$. Therefore, the answer is 01101101_2 for signed magnitude and two's complement.
- Decimal: Since we have a positive value, simply add the powers of 2 for each position that has a 1 in the unsigned binary representation: $2^6 + 2^5 + 2^3 + 2^2 + 2^0 = 64 + 32 + 8 + 4 + 1 = 109_{10}$.

Fourth row:

- Unsigned binary: The first step is to make the conversion from decimal to unsigned binary. We do this by determining the powers of two that make up 130. It actually isn't too bad – there is a 128 $= 2^7$ and a $2 = 2^1$. This gives us a binary value with a 1 in the 7th position and a 1 in the 1st position. (Note that the least significant bit (LSB) is referred to as the 0th position.) This gives us the binary value 10000010_2.
- 2's complement and signed magnitude: Basically, since the unsigned binary value is using the MSB for a magnitude bit, 128, it cannot be used for a sign bit. Therefore, you need to put an 'X' in both of these columns.

2. What is the most negative value that can be represented using 10-bit 2's complement representation?

Remember that for 2's complement, the most negative value has the MSB set to a one (this makes it negative) with all the remaining bits equal to zero. This means that the most negative 10-bit 2's complement value is 1000000000_2. To convert this to a decimal value, we begin by doing the "bit flippy thing." Copying down all of the zeros starting from the LSB along with the first 1 gives us the ***unsigned value*** 1000000000_2. (Yes, I know it's odd, but it's how 2's complement is done.) Converting the unsigned value 1000000000_2 to decimal gives us $2^9 = 512$. But remember, this was a negative value! Therefore, the answer is –29 = –512.

If you wanted to, you could also have used the expression for the most negative 2's complement value for an n-bit number is $-2^{(n-1)}$. Substituting 10 for n gives us $-2^{(10-1)} = -2^9 = -512$.

3. Convert 101010101101011111110_2 to hexadecimal.

To the right is a copy of the hexadecimal to binary conversion table. Simply partition the binary value above into nibbles ***starting on the right side.***

1 0101 0101 1010 1111 1110

As you can see, the left most nibble was not a full four bits long. In fact, it only had 1 bit. Therefore, you need to pad it with three zeros added to the left side. Below is the re-partitioned binary value.

0001 0101 0101 1010 1111 1110

By replacing each nibble with its corresponding hexadecimal digit from the table, we get:

1 5 5 A F E

Binary				Hexadecimal
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	B
1	1	0	0	C
1	1	0	1	D
1	1	1	0	E
1	1	1	1	F

4. Convert the binary value 101.1001 to decimal. (Note the binary point!)

Remember that binary digits to the right of the point continue in descending integer powers relative to the 2^0 position. Therefore, the powers of two are in order to the right of the point $2^{-1} = 0.5$, $2^{-2} = 0.25$, $2^{-3} = 0.125$, $2^{-4} = 0.0625$, and $2^{-5} = 0.03125$.

2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
0	1	0	1	1	0	0	1	0

Therefore, the answer is: $2^2 + 2^0 + 2^{-1} + 2^{-4} = 4 + 1 + 1/5 + 1/16 =$ **5.5625**. You could have left your answer in any of these forms in order to receive full credit.

It is interesting to note that if you moved the binary point all the way to the right side of the 1 that occurs at the 2^{-4} position, this is the same as multiplying by 16, i.e., shifting left 4 positions which is the same as multiplying by 2 four times. The resulting value would be $01011001_2 = 2^6 + 2^4 + 2^3 + 2^0 = 64 + 16 + 8 + 1 = 89_{10}$. If you divide 89 by 16, i.e., shift it right 4 times to turn it back into what we started with, you get 89/16 = 5.5625. Okay, maybe it wasn't that interesting.

5. Convert the 32-bit floating-point number 00111101001010100000000000000000 to its binary exponential format, e.g., 1.1010110×2^{-12}, (which, by the way, is not even close to the right answer).

Begin by dividing up the floating-point number into its components.

<table>
<tr><th>S</th><th>E</th><th>F</th></tr>
<tr><td rowspan="3">0</td><td>01111010</td><td rowspan="3">01010100000000000000000</td></tr>
<tr><td>64 + 32 + 16 + 8 + 2</td></tr>
<tr><td>122</td></tr>
</table>

Substituting into the expression $\pm 1.F \times 2^{(E-127)}$ gives us our answer. (Notice the trailing zeros from the F or fraction part have been removed in the sake of space.)

$$\begin{aligned}\text{Answer} &= \pm 1.F \times 2^{(E-127)}\\ &= 1.010101 \times 2^{(122-127)}\\ &= 1.010101 \times 2^{-5}\\ &= 0.00001010101\\ &= 0.04150390625\end{aligned}$$

Okay, you didn't even remotely have to worry about the last value in that expression, but you should have been able to come up with the third or fourth one.