CPE201 HW 2 (100 points)

Answer all questions completely. Put a box around the final solution. Put your name on it. Show your work.

By hand:

1. Convert the following numbers to floating point representation (represented as 4 hex bytes).
   a. 123.9876

Sign is positive = 0

Convert whole number to binary

| | | | |
|---|---|---|---|
| 123 – 64 | = | 59 | $2^6$ |
| 59 – 32 | = | 27 | $2^5$ |
| 27 – 16 | = | 11 | $2^4$ |
| 11 – 8 | = | 3 | $2^3$ |
| 3 – 2 | = | 1 | $2^1$ |
| 1 – 1 | = | 0 | $2^0$ |

Then the whole number can be represented as:
0111 1011$_2$
This is 7 bits (subtract 1 for the leading 1), so the max amount of bits for the fractional part is 23 – 6 = 17

Convert fractional number to binary

| | | | |
|---|---|---|---|
| 0.9876 – 0.5 | = | 0.4876 | $2^{-1}$ |
| 0.4876 – 0.25 | = | 0.2376 | $2^{-2}$ |
| 0.2376 – 0.125 | = | 0.1126 | $2^{-3}$ |
| 0.1126 – 0.0625 | = | 0.0501 | $2^{-4}$ |
| 0.0501 – 0.03125 | = | 0.01885 | $2^{-5}$ |
| 0.01885 – 0.015625 | = | 0.003225 | $2^{-6}$ |
| .003225 - .001953125 | = | 0.001271875 | $2^{-9}$ |
| 0.001271875 – 0.0009765625 | = | 0.0002953125 | $2^{-10}$ |
| 0.0002953125 – 0.000244140625 | = | 0.000051171875 | $2^{-12}$ |
| 0.000051171875 – 0.00030517578125 | = | 0.000020654296875 | $2^{-15}$ |
| 0. 000020654296875– 0.0000152587890625 | = | 0.0000053955078125 | $2^{-16}$ |
| 0. 0000053955078125– 0.000003814697265625= | | 0.000001580810546875 | $2^{-18}$ |

Then the fractional number can be represented as:
0.1111 1100 1101 0011 0$_2$

Then the entire number can be represented as
111 1011.1111 1100 1101 0011 0

Convert this to an exponent representation

1.1110 1111 1111 0011 0100 110 x $2^6$

E – 127 = 6
E = $133_{10}$ = $2^7$ + $2^2$ + $2^0$ = 1000 0101

Create the entire floating-point representation
0 1000 0101 1110 1111 1111 0011 0100 110

Regroup to 4 bit groups to convert to hex
0100 0010 1111 0111 1111 1001 1010 0110
0x4 0x2 0xF 0x7 0xF 0x9 0xA 0x6

Then the floating-point number is 0x42 F7 F9 A6

b. 3.141

Sign is positive = 0

Convert whole number to binary

| | | | |
|---|---|---|---|
| 3 – 2 | = | 1 | $2^1$ |
| 1 – 1 | = | 0 | $2^0$ |

Then the whole number can be represented as:
$0000\ 0011_2$
This is 2 bits (subtract 1 for the leading 1), so the max amount of bits for the fractional part is 23 – 1 = 22

Convert fractional number to binary

| | | | |
|---|---|---|---|
| 0.141 – 0.125 | = | 0.016 | $2^{-3}$ |
| 0.016 – 0.015625 | = | 0.000375 | $2^{-6}$ |
| 0. 000375 – 0.000244140625 | = | 0.000130859375 | $2^{-12}$ |
| 0. 000130859375– 0.0001220703125 | = | 0.0000087890625 | $2^{-13}$ |
| 0. 0000087890625– 0.00000762939453125 | = | 0.00000115966796875 | $2^{-17}$ |
| 0. 00000115966796875– 0.00000095367431640625 | = | 0.00000020599365234375 | $2^{-20}$ |
| 0. 00000020599365234375– 0.00000011920928955078125= | | 0.00000008678436279296875 | $2^{-23}$ |

Then the fractional number can be represented as:
$0.0010\ 0100\ 0001\ 1000\ 1001\ 00_2$

Then the entire number can be represented as
11. 0010 0100 0001 1000 1001 00

Convert this to an exponent representation
1. 1001 0010 0000 1100 0100 100x $2^1$

E – 127 = 1

E = $128_{10}$ = $2^7$ = 1000 0000

Create the entire floating-point representation
0 1000 0000 1001 0010 0000 1100 0100 100

Regroup to 4 bit groups to convert to hex
0100 0000 0100 1001 0000 0110 0010 0100
0x4 0x0 0x4 0x9 0x0 0x6 0x2 0x4

Then the floating-point number is 0x40 49 06 24

2. Convert the following English statements to Boolean statements after picking variables.
   a. I’ll go to dinner with Terry as long as Terry brings his wife, otherwise George has to come too.

F = I go to dinner
T = Terry goes to dinner
W = Terry’s wife goes to dinner
G = George goes to dinner

F = TW + TG
OR
F = T(W + G)

   b. I’ll go to the store if I need to buy milk and eggs, or tp, or ice (as long as I have soda).

M = I need to buy milk
E = I need to buy eggs
T = I need to buy tp
I = I need to buy ice
S = I have soda

F = ME + T + IS

3. Evaluate the following Boolean statement with the given inputs
   F = AB’ + A’C + BD
   a. A = 0, B = 0, C = 1, D = 0

Substituting the values into the equation
00’ + 0’1 + 00
Simplifying
01 + 11 + 00
0 + 1 + 0
1

b. A = 1, B = 1, C = 0, D = 0

Substituting the values into the equation
11’ + 1’0 + 10
Simplifying
10 + 00 + 10
0 + 0 + 0
0

c. A = 1, B = 0, C = 0, D = 1

Substituting the values into the equation
10’ + 1’0 + 01
Simplifying
11 + 00 + 01
1 + 0 + 0
1

4. What inputs are needed in order to get the following Boolean statement to evaluate to TRUE?
   F = B’C + AD + CD’ + B

| A | B | C | D | B’C | AD | CD’ | F |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |

There are 13 out of 16 possible input combinations that will make the output TRUE.

With a calculator:

5. Convert the following message to ASCII encoding (using an ASCII table), then calculate the 8-bit checksum for the message: “It’s under the sauce”.

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| I | t | ‘ | s | ␣ | u | n | d | e | r | ␣ | t | h | e | ␣ | s | a | u | c | e |
| 0x49 | 0x74 | 0x27 | 0x73 | 0x20 | 0x75 | 0x6E | 0x64 | 0x65 | 0x72 | 0x20 | 0x74 | 0x68 | 0x65 | 0x20 | 0x73 | 0x61 | 0x75 | 0x63 | 0x65 |

Use a hex calculator to get the sum of all 20 numbers
Sum = 0x727

Calculate the checksum by taking the 2’s complement of 0x2C
0010 1100 → 1101 0011 → 1101 0100 → 0xD9

6. Show that the checksum from problem 5 is correct.

Take the sum (truncate it to 8-bits) and add the checksum and the result should be zero
0x727 + 0xD9 → 0x27 + D9

```
  0010 0111
+ 1101 1001
1 0000 0000
```

Throw away the carry bit that is in the position for the 9th bit to get an 8-bit results, which is zero.