Problem 1. “Bits of Floating-Point”

Represent the following in single-precision IEEE floating point. Give your answers in hexadecimal:

a) 2007.0
b) -4014.0
c) -0.00390625 (Hint: -2^{-8})
d) 16777215.0 (Hint: 2^{24} - 1)
e) 16777216.0 (Hint: 2^{24})

Convert the following single-precision floating point numbers (given in hexadecimal) to decimal:

f) 0x3e800000
g) 0xbec00000
h) 0x46800000
i) 0xc67ffc00
j) 0x46800200

Problem 2. “Floating-Point Arithmetic”

Given the following two single-precision IEEE floating-point numbers:

\[ x = 0x3c800000 \quad \text{and} \quad y = 0x48800000 \]

Compute the following showing all work:

a) \( x + y \)
b) \( x \times y \)
c) \( x - y \)
Problem 3. “Half-Precision”

Certain modern graphics cards have adopted a new floating format called “half precision”. Half-precision numbers are stored in 16-bits (2-bytes), and they use the following format:

\[ S \text{ EEEEE FFFFFFFFFF} \]

Where S is the sign-bit, E indicates bits of the exponent, which are represented in bias-15, and F is a 10-bit fractional component of an 11-bit significand with a hidden “1”. The exponent values of 00000\(_2\) and 11111\(_2\) are reserved, and the values 0x0000 and 0x8000 are interpreted as 0.0 and -0.0 respectively.

a) Represent the number -0.03125 in half-precision. Give your answer in hexadecimal.

b) What value does 0x2bad represent?

c) What is the largest positive number representable as a normalized number in half-precision format?

d) What is the smallest number greater than zero that is representable as a normalized number in this format?

Problem 4. “Floating-Point Division”

Most floating-point units perform division iteratively using the following process. The expression, \( A / B \) is computed by multiplying \( A \) times \( X = 1/B \), which is approximated using the following iteration formula.

\[ X_{i+1} = X_i (2 - B X_i) \]

This approach requires an initial guess, \( X_0 \). If we assume that \( B \) is between 1.0 and 2.0, as would be the case for the significand in IEEE floating point, we can start with \( X_0 = (3 - B) / 2 \).

a) Compute \( X_0 \), \( X_1 \), \( X_2 \), and \( X_3 \) when \( B = 1.25 \)

b) Compute \( X_0 \), \( X_1 \), \( X_2 \), and \( X_3 \) when \( B = 1.5 \)

c) The number 10 can be represented as \( 1.25 \times 2^3 \), and the inverse of the significand can be computed as in part a). How should one adjust the exponent? Compute \( 1/10.0 \) using 3 iterations. Make sure your result is of the form, a number between 1.0 and 2.0 multiplied by a power of 2.

d) An alternative method for initializing \( X_0 \) is to use a look-up table based on a few of the most significant bits of \( B \)’s significand (Note: Errors in a table like this one were the cause of the Intel Pentium FDIV bug). Assume the following table for the first 2 bits (note the B values are given in binary, whereas \( X_0 \) is given in decimal):

<table>
<thead>
<tr>
<th>B</th>
<th>( X_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00xxx</td>
<td>0.888888</td>
</tr>
<tr>
<td>1.01xxx</td>
<td>0.727272</td>
</tr>
<tr>
<td>1.10xxx</td>
<td>0.615384</td>
</tr>
<tr>
<td>1.11xxx</td>
<td>0.533333</td>
</tr>
</tbody>
</table>

Repeat part b), with \( X_0 \) initialized from this table. Compare the accuracy of the results after one iteration.
**Problem 5. The Real “Y2K”**

Immediately following an unusually useful Comp 411 lecture, Lee Hart races back to his part-time job at the upstart Mod-3 Calendar Corporation. Central to all products in the Mod-3’s product line is a patented circuit that determines if a given calendar year (e.g. 2007) is divisible by 3. The year is entered into Mod-3’s products as an unsigned integer, one bit at a time. In Mod-3’s prototype system, this function was accomplished with a shift register and a look-up table using the circuit shown below.

A) Lee has been told that the look-up table has been carefully determined and programmed into a ROM. Based on the circuit shown above, how many bits are in this ROM?

B) If each register has the following timing specs, \( t_{pd} = 3 \text{ ns} \), \( t_s = 2 \text{ ns} \), and \( t_h = 1 \text{ ns} \), and the ROM’s specs are \( t_{pd} = 11 \text{ ns} \), what is the minimum value of \( t_{cd} \) that allows the circuit to operate properly?

C) What is the fastest rate which Mod-3’s patented circuit can be clocked and still operate as intended?

Lee has warned the management at Mod-3 of a real looming Y2K problem associated with the current design, and in an effort to humor him they have assigned him the task of improving the circuit’s design. Lee recalls a simple state machine specification from Comp 411. The state transition diagram of the circuit is shown below.

D) Verify the operation the behavior of the state transition diagram given in class by determining its final state for each of the following input sequences following a reset

\{11111010011, 11111010001, 10111010100, 11011110000, 11110101010\}

Convinced that the circuit performs as specified, Lee hastily sets out to implement the state machine. He cleverly assigns the following state encodings, \( S_0 = 10 \), \( S_1 = 00 \), and \( S_2 = 01 \). This allows the most significant bit of the state encoding to serve double duty as the divisible by three output, \( D_3 \).
E)  Recreate Lee’s state machine using a ROM and a 2-bit state register. What is the size of the ROM. Write out a table showing the new ROM’s contents.

F)  After completing his FSM design, Lee closely examines the behavior of the original circuit only to realize that the original circuit expects the year input to be entered *most significant bit* first. Feeling slightly demoralized, he decides to go ahead and substitute his version on the circuit into a working Mod-3 prototype. Much to his surprise, the prototype system operates just as before. Explain how this is possible.

G)  Why is this new “divisible-by-3” circuit immune to the Y2K problem foreseen by Lee? Is it likely to be faster than the original design? Is it likely to be less expensive than the original (explain why)?