CIS 2107. Practice Problem Solutions        due: Friday, October 14

1. (5 points) For this question, we’re doing 6-bit two’s complement representation of integers. Fill in the empty boxes in the following table. Addition and subtraction should be performed based on the rules for 6-bit, two’s complement arithmetic. Recall that in your book’s notation, TCMin is defined to be the smallest negative two’s complement number that we can represent, and TCMax is the largest positive one. UMin and UMax are the smallest and largest unsigned numbers that we can represent.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>0</td>
<td>0</td>
<td>00 0000</td>
</tr>
<tr>
<td>n/a</td>
<td>13</td>
<td>13</td>
<td>00 1101</td>
</tr>
<tr>
<td>n/a</td>
<td>22</td>
<td>22</td>
<td>01 0110</td>
</tr>
<tr>
<td>n/a</td>
<td>58</td>
<td>-6</td>
<td>11 1010</td>
</tr>
<tr>
<td>n/a</td>
<td>38</td>
<td>-26</td>
<td>10 0110</td>
</tr>
<tr>
<td>n/a</td>
<td>9</td>
<td>9</td>
<td>00 1001</td>
</tr>
<tr>
<td>n/a</td>
<td>57</td>
<td>-7</td>
<td>11 1001</td>
</tr>
<tr>
<td>n/a</td>
<td>25</td>
<td>25</td>
<td>01 1001</td>
</tr>
<tr>
<td>n/a</td>
<td>59</td>
<td>-5</td>
<td>11 1011</td>
</tr>
<tr>
<td>TCMax</td>
<td>31</td>
<td>31</td>
<td>01 1111</td>
</tr>
<tr>
<td>TCMin+2</td>
<td>34</td>
<td>-32</td>
<td>10 0010</td>
</tr>
<tr>
<td>UMax</td>
<td>63</td>
<td>-1</td>
<td>11 1111</td>
</tr>
<tr>
<td>UMin</td>
<td>0</td>
<td>0</td>
<td>00 0000</td>
</tr>
<tr>
<td>TCMax+2</td>
<td>33</td>
<td>-31</td>
<td>10 0001</td>
</tr>
</tbody>
</table>
2. (3 points) What’s printed by the following?

```c
#include <stdio.h>

int main(int argc, char **argv) {
    char x;
    unsigned char y;
    x=65;
    y=x;
    x*=2;
    y*=2;
    printf("x=%d, y=%u\n", x, y);
    return 0;
}
```

Solution:
The program prints

```
x=-126, y=130.
```

3. (3 points) What’s printed by the following?

```c
#include <stdio.h>

int main(int argc, char **argv) {
    char x;
    unsigned char y;
    x=60;
    y=x;
    x*=4;
    y*=4;
    printf("x=%d, y=%u\n", x, y);
    return 0;
}
```

Solution: It’s the same idea as the previous problems.
60 * 4 = 240_{10}, which is represented by the bit string 1111 0000₂. If it’s interpreted as an unsigned integer, we get 240₁₀. If it’s interpreted as a signed value, we have $-128₁₀ + 112₁₀ = -16₁₀$

The program prints

```
x=-16, y=240.
```
4. (3 points) How would \( 139.5625_{10} \) be represented as a 32-bit float?

Solution:

- \( 139_{10} = 1000\,1011_2 \)
- \( 0.5625_{10} = 0.1001_2 \)
- \( 139.5625_{10} = 1000\,1011.1001_2 \)
- \( 1000\,1011.1001_2 \) normalized is \( 1.0001\,0111\,001_2 \times 2^7 \)
- For 32-bit floats, 8 bits are used for the exponent field, so the bias value is \( 2^{8-1} - 1 = 127 \).
- What’s stored in the exponent field is the actual exponent of 7 + the bias value of 127 = \( 134_{10} \), which is \( 1000\,0110_2 \).

The final result is

<table>
<thead>
<tr>
<th>sign</th>
<th>exponent</th>
<th>mantissa</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1000 0110</td>
<td>0001 0111 001 followed by 12 zeros</td>
</tr>
</tbody>
</table>
5. (3 points) **Recognizing the value of a Floating-point variable.** In this question, consider 7-bit floating-point numbers. What number is represented by 0 011 000, where:

- 0 is the sign bit
- 011 is stored in the mantissa field
- 000 is stored in the exponent field

**Solution:**

- The sign bit is a 0 so the number is positive
- The mantissa field 011 represents the fraction $1/4 + 1/8$
- The exponent field is 3-bits wide, so the bias is $2^{3-1} - 1 = 3$
- The exponent field is all zeros so it’s denormalized, and the exponent represented is $1 - 3 = -2$
- The number represented is $(1/4 + 1/8) \times 2^{-2} = (1/4 + 1/8) \times 1/4 = 3/32 = 0.093750$.

6. (3 points) **Recognizing the value of a Floating-point variable.** In this question, consider 8-bit floating-point numbers. What number is represented by 1 101 0000, where:

- 1 is the sign bit
- 101 is stored in the mantissa field
- 0000 is stored in the exponent field

**Solution:**

- The sign bit is a 1 so the number is negative
- The mantissa field 101 represents the fraction $1/2 + 1/8$
- The exponent field is 4-bits wide, so the bias is $2^{4-1} - 1 = 7$
- The exponent field is all zeros so it’s denormalized, and the exponent represented is $1 - 7 = -6$
- The number represented is $-(1/2 + 1/8) \times 2^{-6} = -(1/2 + 1/8) \times 1/64 = -5/512 = -0.009766$. 