**Assignment 5 – MSP430 - Implementing Fixed Point Math**

Due: Due Before the start of MONDAY’s class (February 17, 2020)

Honor Code: \_\_\_\_\_\_\_I have neither given or received, nor have I tolerated others’ use of unauthorized aid. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name: \_\_\_\_\_Joe Leveille\_\_\_\_\_\_\_ Signature: \_\_\_\_Joseph Leveille\_\_\_\_\_

Work through the Lesson - Implementing Fixed Point Math with TI’s Qmath and IQmath Libraries, and answer the following.

1. Record the data.

|  |  |  |
| --- | --- | --- |
| **Operation** | **Clock Cycles** | **Result** |
| **out1** | 202 | 9 |
| **out2** | 406 | 9 |
| **out3** | 45 | 9 |

1. Which outx operation took the most clock cycles? Why?

Out2 b/c it had the most calculation steps to complete.

1. Which took the least clock cycles? Was it the one you expected? Was the number of clock cycles saved significant?

Out3 b/c it only needed 2 bit shifts and an addition, all of which are easily handled by an ALU. This was a very significant improvement in clock cycles.

1. Why was the Q4.12 format chosen for this operation instead of the more precise Q1.15 fixed point format? Hint look at Table 2.1 in the Qmath and IQmath Library User’s Guide.

Q1.15 is not as precise, since it only has a resolution of one digit used for the fractional part of a number.

1. Record the data.

|  |  |  |
| --- | --- | --- |
| **Operation** | **Clock Cycles** | **Result** |
| **Floating point** | 418 | z1=0.142857149 |
| **Fixed point** | 122 | z2= 0.142822266 |
| **Conversion** | 84 | Zz=585 |

1. Write the equation for converting this number to floating point and compute on your calculator. Record your result.

Zz= 0.142822266

zz\*2^4 = 585 = z2

84 clock cycles

1. How much faster is the fixed point conversion in this example?

This is the only fixed point conversion, I’m not sure what I’m supposed to compare to, so I’ll assume the float -> fixed point conversions, which took fewer than 122 cycles to complete 2 of them plus a division, so fixed to floating point is slower.

1. How far off is the fixed point result from the floating point result (z1 vs. z2)?

0.142857149 = z1

0.142822266 = z2

So about .00003 off

1. How many bits is a float data type?

32 bits

1. How many bits was the fixed point data type?

16 bits

1. Record the data.

|  |  |  |
| --- | --- | --- |
| **Operation** | **Clock Cycles** | **Result** |
| **Floating point** | 423 | z1=0.142857149 |
| **Fixed point** | 249 | z2=0.142857149 |
| **Conversion** | 98 | Zz=38347922 |

1. First, why was the Q4.28 fixed point format chosen for this example rather than the more precise Q2.30 format?

In order to hold the value of 7.0, there needs to be at least 3 bits on the integer side, so 2.30 would not be a valid option.

1. How much faster is the fixed point conversion than the floating point conversion in this example?

One floating point to fixed point conversion took 5 clock cycles (measures separately from the table), whereas the fixed to floating point took 98 cycles, so almost 20 times faster.

1. How far off is the fixed point result from the floating point result (z1 vs. z2)?

They are precisely the same.

1. Is this a more fair comparison with the floating point conversion?

Yes, since it will hold the same information exactly.

That is it for this brief introduction to fixed point calculations on microcontrollers. Go back to your projects!