Lab1 Report
Michael Park(mjp2853), Jack Zhao(jz6492)
01/27/16

Lab 1. Fixed Point Output

Objectives:

Our objective for Lab 1 was to experiment with Fixed Point calculations on the microcontroller. Then we would have to demonstrate these values as outputs onto the LCD screen. Specifically, we would have to deal with a wide variety of situations requiring fixed resolutions, to test the robustness of our program in dealing with the situations without the use of floating point calculations. The reason we would need to do this is that floating point processing requires much more resources and time for calculation than fixed point calculations. We would need to deal with both the situations of printing out specific fixed point values onto the screen as well as using fixed point values in the plotting of a graph. We would need to perform these scenarios in different methods than just standard multiplication of a resolution.

Lab1 Report

Michael Park(mjp2853), Jack Zhao(jz6492)
01/27/16

Analysis and Discussion:

For this lab, one of our objectives in design was to minimize the number of arrows in the call graph. This is good design because with a reduced amount of arrows in the call graph will reduce the modules dependence on any portion of the overall program. This allows the program to be used in a wide variety of settings and on several different machines with the same efficiency. In addition, we implement the feature of setting the decimal point to be fixed (ie. In the exact same physical position independent of the number). The reason that we do this is we can pretty-print values on LCD. Because the resolution of the software will be consistent. Since we will have a format for the display values, we can use ST7735_SetCursor function to set the fixed cursor position for the displays. Also the displaying will become flexible in that we can display digits in the value in any order once we have the output string built.

In performing this lab, we reached the option of using fixed point calculations versus floating point calculations. There are set instances that we implement both of these calculation methods. The reason we would need to use fixed point calculations in some cases over floating point calculations is that fixed point can represent integers up to 16 bits with a possible combination of 2^16 possible bit patterns. This becomes handy versus the use of floating point, when the calculations themselves are not too complex. A few key considerations developers make when using fixed point is Cost, Ease of Development and Performance. Fixed point processers can be manufactured at a much lower cost than floating point processors because they are simpler in nature. In addition, the performance speed of fixed point calculations is much quicker than floating point which allows for the mass processing of data for fixed point calculations. On the other hand, floating point calculation is much more complex than fixed point calculations, but allows for more robustness in the processing of data. Since floating point calculations are similar to scientific notations, both the resolutions and the precisions can be altered to reflect better values. This allows for up to 2^32 bit patterns in floating point calculations. Though the cost of manufacturing a floating point processor is much greater than for a fixed point processor, developing an algorithm for floating point processors is much easier as there is less manipulation of the actual values. In addition, floating point can be used to calculate a much wider range of values than fixed point which allows for more precise calculations. All these variables allow for floating point processors to be more specialized in data processing (ie. Addressing specific scenarios that a fixed point processor would not be able to address).

Another consideration that we made is whether we would use binary fixed point or decimal fixed point. In performing this lab, we were exposed to both of these methods and led us to decide when we would use either of these methods. Binary fixed point is preferred in situations where speed of calculations matter. Binary fixed point is processed much quicker because bit shift to rescale numbers into fixed point is much quicker than standard multiplication. But decimal fixed point is preferred for usability especially when the resolution is a multiple of 10 because it is much easier to develop programs using decimal values.

An example of a system in our lives that uses fixed point is a vending machine. A vending machine (that accepts US dollars) uses fixed-point numbers. The resolution only needs to be 0.01. Therefore, the numbers can be represented as integers internally. This leads us to the question as to whether we can use floating point on our board, the TM4C123. The answer as to whether the hardware is implemented is yes. We can use floating point on the ARM Cortex M4 with single precision FPU. The cost of performing floating point calculations is primarily a time cost.
Without FPU: 1.25 / 1.52 / 1.91 DMIPS/MHz
With FPU: 1.27 / 1.55 / 1.95 DMIPS/MHz

Some final considerations as we performed this lab is that there are different instances in using fixed point versus floating point calculations and this can depend based on a variety of factors. At one time, floating point was not possible to be implemented due to hardware constraints and so people developed methods completing various tasks using fixed point implementations. Though our hardware has progressed significantly, there are still many applications of fixed point in our society today.