Ryan Comer – rnc525
Connor White – bcw824

Lab 1 Report

Objectives

The objective of this lab was to familiarize us with the Keil programming environment and the Stelaris LCD screen by creating a software module that has fixed point and graphical capabilities. This software module consisted of 2 primary files: fixed.c and fixed.h. The fixed.h file had the function definitions for 4 functions, and the fixed.c file had the implementation details for those functions. The functions include: convert an integer to decimal fixed point, covert an integer to binary fixed point, initialize the LCD display, and draw pixels defined by their (x, y) coordinates. Ideally, a user should only look at the .h file to see how to use the functions thus abstracting the actual implementation.

Another objective of this lab was to learn about engineering tradeoffs. The started code has a fputc method which calls the LCD function that outputs a character to the screen. The printf method calls fputc for every character in the input string, so using the printf function will print to the screen. The tradeoff here is how to output text to the screen. If you use printf for this task, the code can be greatly simplified at the cost of the overhead of the printf function. Alternatively, you could use fputc directly which would speed up the execution but make the code more complex.

Analysis and Discussion

In what way is it good to minimize the number of arrows in the call graph for your system?
More arrows in a call graph means in increase in the coupling between modules of the system. In general, it is best to decrease coupling as minimal coupling will increase clarity and decrease the extent to which each of the modules rely on eachother.

Why is it important for the decimal point to be in the exact same physical position independent of the number being displayed? Think about how this routine could be used with the ST7735_SetCursor command.
The decimal point needs to be in the same location regardless of the number being displayed so that the screen is properly formatted, with each number lined up correctly. This could be done by using the ST7735_SetCursor command by resetting the cursor to a set location each time the decimal is printed that way the alignment would still be the same.

When should you use fixed-point over floating point? When should you use floating-point over fixed-point?
Fixed-point should be used over floating point when the range of values is known and small. In this situation, fixed point will have greater resolution than floating point. Floating point should be used over fixed-point when the range of possible values is large, because floating point can store a dynamic range of numbers.

When should you use binary fixed-point over decimal fixed point? When should you use decimal fixed-point over binary fixed-point?
When exact decimal fractions are needed, it is best to use decimal fixed-point. Otherwise, binary fixed point can be better because it will be easier to use when performing mathematical calculations since the bits can simply be left and right shifted.

Ryan Comer – rnc525
Connor White – bcw824

Give an example application (not mentioned in the book) for fixed-point. Describe the problem, and choose an appropriate fixed-point format. (no software implementation required).
An application of the fixed point format would be a digital display of the speed of a vehicle. The problem here is to display a vehicle's speed in MPH with a resolution of 0.1. The fixed point format would be I * (1/10), which would convert 553 to 55.3 MPH.

Can we use floating point on the ARM Cortex M4? If so, what is the cost?
We can use floating point on the ARM Cortex M4, but the tradeoffs here are with speed and power. The M4 comes with FPUs which can be enabled. These FPUs can potentially do floating point operations faster than the ALU can do fixed point, but the power cost is higher.