Deliverables

Objectives:

Our objectives for this lab are to develop a solid understanding of how to use the TM4C123 Microcontroller, the SATSET LCD, and the Keil IDE. In addition to this fundamental knowledge of interfacing with the lab components, we also aim to program a set of functions to output fixed point conversion for base10 and base2. Familarizing ourselves with the MicroController, LCD and Keil will allow us to complete the subsequent labs without spending time on the “setup” phase of the project. A mastery\* of Keil’s processes and modes also enables us to debug and test our program efficiently.

Furthermore, we will manipulate input data for a set of points to draw a particular shape in order to fit our LCD parameters. This will require scaling the data appropriately and offsetting it to achieve the desired image.

These goals together encompass our objectives for Lab1.

Analysis and Discussion:

1) In what way is it good design to minimize the number of arrows in the call graph for your system?

It is faster.

2) 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.

3) When should you use fixed-point over floating point? When should you use floating-point over fixed-point?

We should use fixed-point numbers when we want to express values in our computer that have non-integer values. We can use fixed point when the range of values is small and known. Therefore we will not need floating point operations for most embedded systems applications because fixed point is sufficient. Furthermore, if the processor does not have floating point instructions then a floating point implementation will run much slower than the corresponding fixed point implementation.

4) When should you use binary fixed-point over decimal fixed-point? When should you use decimal fixed-point over binary fixed-point?

When adding or subtracting fixed-point numbers with different fixed parts, then we must first convert the two inputs to the format of the result before adding or subtracting. This is where binary fixed point is more efficient, because the conversion process involves shifting rather than multiplication or division. Many instructions on the ARM allow a data shift operation to be performed at no added execution time.

When interacting with a human operator, it is usually convenient to use decimal fixed point. Decimal Fixed will be easy to display, while binary fixed point will be easier to use when performing mathematical calculations. The ARM processor is very efficient performing left and right shifts. Because the Shift operator logic is processed before the input is sent to the Arithmetic Logic Unit (ALU).

5) 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).

6) Can we use floating point on the ARM Cortex M4? If so, what is the cost?