1. **OBJECTIVES**

Lab 1 aimed to revive our EE 319K skills and develop a set of useful fixed point output routines. The fixed-point routines used to display graphics onto the LCD will be used in subsequent labs. Another objective of this lab was to compare the engineering tradeoffs between fixed point and floating point numbers.

1. **ANALYSIS AND DISCUSSION**
   1. **In what way is it good design to minimize the number of arrows in the call graph for your system?**

The more arrows there are, the more complex the call graph will look and the harder it will be to read/understand (which would ultimately defeat the purpose of using a call graph)

* 1. **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.**

Without a fixed location, you would not know where to put the decimal point (otherwise you would have to some other kind of notation like floating point).

* 1. **When should you use fixed-point over floating point? When should you use**

**floating-point over fixed-point?**

You can use fixed point if you know the range of values you need to represent ahead of time and if this range is small. If you don’t know the range of values ahead of time or if the range is too large, use floating point. If you need to optimize for speed, then you have to see if your processor has the necessary floating point hardware support (otherwise the floating point operations will run very slowly). For systems that don’t optimize for floating point operations, fixed point will be (generally) faster. In general, floating point operations consume more power due to the extra hardware support and overhead. Floating point operations also push extra registers onto the stack, which consumes space and results in more overhead. For this class, we will generally be using fixed point.

* 1. **When should you use binary fixed-point over decimal fixed-point? When should you use decimal fixed-point over binary fixed-point?**

You should use binary fixed point if you need faster mathematical operations (multiplications and divisions will be carried out through shifts, which are very efficient and take zero clock cycles according to Dr. Valvano). You should use decimal fixed point if you want to work with human readable numbers.

* 1. **Give an example application (not mentioned in the book) for fixed-point. Describe the problem and choose an appropriate fixed-point format.**
  2. **Can we use floating point on the ARM Cortex M4? If so, what is the cost?**

Yes but you have to enable target options to enable the floating point unit and uncomment the appropriate lines in startup.s. The cost would be pushing extra registers onto the stack for floating point operations, increased overhead, and probably extra power usage.