ECE 461/561 –   
Embedded System Design:  
Speed Profiling and Optimization Project

# Overview

In this project you will port and write code and then analyze and optimize its run-time performance (i.e. speed). Profile the resulting code to find the slow parts, and then optimize them as much as you can. Feel free to optimize ***any*** code in the system which will help your demonstration program run faster. 10% of your grade will be based on your final program’s speed.

Extend the Glyph graphics API to include these functions:

* Draw\_Line(x1, y1, x2, y2, color)
* Draw\_Circle(x1, y1, radius, color). The circle does not need to be filled. Note that the pixels on the LCD are not square, so you will actually need to scale either the x or y coordinate of each point.
* Demonstration program: figure starts centered in display, moves in display, bounces off sides/top/bottom.
  + First, 100 iterations of bouncing NCSU logo (logo is half as wide as the display). With each iteration, change the logo’s x position and y position by one pixel. Reverse the change direction if that will start moving the logo off the screen.
  + Second , 100 iterations of bouncing circle (circle is half as wide as the display). With each iteration, change x position and y position by one pixel. Reverse the change direction if that will start moving the circle off the screen.
  + Third, 100 iterations of rotating square (40x40 pixels). With each iteration, change x position and y position by one pixel, and angle of rotation by 1 degree. Reverse the change direction if that will start moving the square off the screen.

Deliverables:

* Ported source code
* Report with documentation, and stepwise performance analysis and description of optimizations (with their quantitative impact).

Resources:

* Wiki page on Bresenham’s Line Algorithm: <http://en.wikipedia.org/wiki/Bresenham%27s_line_algorithm>
* <http://free.pages.at/easyfilter/bresenham.html>
* Wiki page on Midpoint Circle Algorithm: <http://en.wikipedia.org/wiki/Midpoint_circle_algorithm>

# Demonstration Requirements

You must demonstrate your program to the instructor or a TA.

# Submission Requirements

Submit the following items in a zip archive.

1. Project workspace, source and object files.
2. Project Report (PDF)
   1. Introduction
   2. Changes needed to port code to RL78.
   3. Execution Time Analysis and Optimization
      1. Initial execution time profile. Break down for bitmap, circle and square.
      2. Narrative description of each optimization you tried, with measurements of performance before and after. Break down for bitmap, circle and square.
      3. Final execution time profile. Break down for bitmap, circle and square.
      4. Analysis and discussion of results
   4. Lessons learned in this project, and how you might do things differently next time
      1. Technical issues (processor, peripherals, compiler, assembly code, etc.)
      2. Your own software development process
   5. Conclusions

# Grading

You grade will be based upon the submitted report and the speed of your fully optimized program.