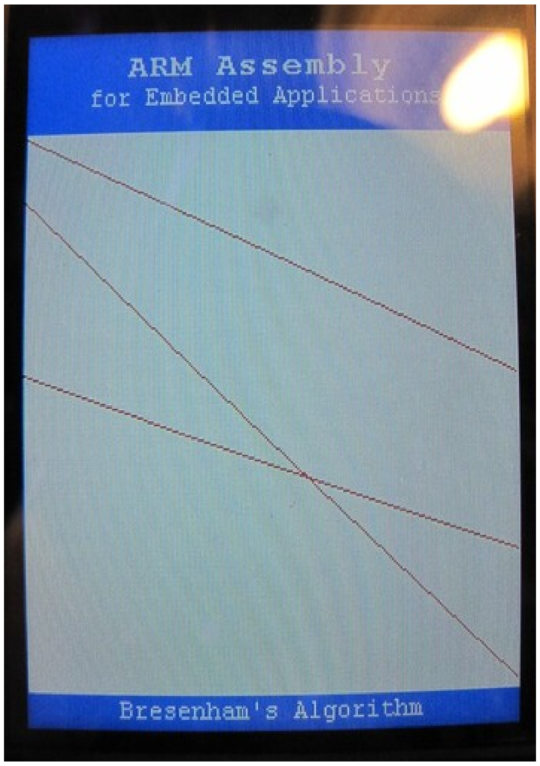
CS2240 Fall 2022
Programming Project #2 – The Bresenham Line Drawing Algorithm
Due: Friday, October 28, 2022, 11:59pm

Hint: Remember the helpful ‘`SVC #0`’ debugging instruction available with your Discover Board library! That can be inserted anywhere in your assembly code any time you need to clarify exactly what is in the registers!

In the Lab #6 exercise you developed assembly code to draw horizontal and vertical lines on the Discovery Board display. We now explore an efficient algorithm for drawing a line in a bitmap graphics system at any arbitrary downward slope, not just straight horizontal or vertical. First study the basics of the Bresenham algorithm at https://en.wikipedia.org/wiki/Bresenham%27s_line_algorithm . In the section titled “Algorithm for integer arithmetic”, the short algorithm in the text box under the line “This results in an algorithm that uses only integer arithmetic” is what we will be coding in assembly. Download the C code file Project2- BresenhamMain.c and the template assembly source file Project2-bresenham.s. Your project code will resemble the lab exercises with two source code modules, a main program in C and an assembly module. For this project the assembly module will provide one function called by the C code. Note that the coordinate variables *x* and *y* in this Wikipedia article are the ‘column’ and ‘row’ coordinates we have been using in the labs writing graphics to the display screen.



You will complete the ‘bresenham’ function in the assembly source file to follow the Bresenham algorithm for integer arithmetic and draw a line. There should be four function arguments accepted, in the order x0, y0, x1, y1. These are the starting coordinates (x0,y0) and ending coordinates (x1,y1) of the line to be drawn, as in the algorithm description. Note in the main C program three calls are made to a function ‘bresenham’ in the assembly language module to draw three lines on your display screen with different starting and ending (column,row) coordinates. The coordinates in the main program function calls are consistent with the downward slope in the octant assumed in the Wikipedia article. Make use of your code in Lab #7 that generated a display pixel array index $i=(y \cdot 240)+x$ and then an STR instruction with the [Rn, Rm, LSL #2] addressing mode to set the screen pixel. Choose whatever pixel color you would like for your lines. Be sure to push onto the stack at the entry point of your function any registers other than R0-R3 and R12 you use and pop them off the stack before returning.

Run your code and verify that it produces a display similar to that shown above. Upload your code to Canvas and a screen photo. Make sure, however, that any debugging breakpoints you inserted for debugging your code have been removed in the version you submit.