```
/*
 * simonDisplay.h
 *
 *  Created on: Jun 4, 2015
 *      Author: Taylor Cowley
 */
#ifndef SIMONDISPLAY_H_
#define SIMONDISPLAY_H_

#include "supportFiles/display.h"
#include "supportFiles/utils.h"
#include <stdbool.h>
#include <stdint.h>
#include <stdio.h>

#ifndef SIMON_H_
#define SIMON_H_

// Width, height of the simon "buttons"
#define SIMON_DISPLAY_BUTTON_WIDTH 60
#define SIMON_DISPLAY_BUTTON_HEIGHT 60
#define SIMON_DISPLAY_BUTTON_WIDTH_HALF  (SIMON_DISPLAY_BUTTON_WIDTH  / 2)
#define SIMON_DISPLAY_BUTTON_HEIGHT_HALF (SIMON_DISPLAY_BUTTON_HEIGHT / 2)

//width, height of the screen
#define SIMON_DISPLAY_HEIGHT display_height()
#define SIMON_DISPLAY_WIDTH  display_width()
#define SIMON_DISPLAY_HALF_HEIGHT   (SIMON_DISPLAY_HEIGHT / 2)
#define SIMON_DISPLAY_HALF_WIDTH    (SIMON_DISPLAY_WIDTH  / 2)
#define SIMON_DISPLAY_FOURTH_HEIGHT (SIMON_DISPLAY_HEIGHT / 4)
#define SIMON_DISPLAY_FOURTH_WIDTH  (SIMON_DISPLAY_WIDTH  / 4)
#define SIMON_DISPLAY_THREE_FOURTH_HEIGHT (SIMON_DISPLAY_HEIGHT * 3 / 4)
#define SIMON_DISPLAY_THREE_FOURTH_WIDTH  (SIMON_DISPLAY_WIDTH  * 3 / 4)

// Given coordinates from the touch pad, computes the region number.

// The entire touch-screen is divided into 4 rectangular regions, numbered 0 - 3.
// Each region will be drawn with a different color. Colored buttons remind
// the user which square is associated with each color. When you press
// a region, computeRegionNumber returns the region number that is used
// by the other routines.
/*
|----------|----------|
|          |          |
|    0     |     1    |
|  (RED)   | (YELLOW) |
------------------------
|          |          |
|     2    |    3     |
|  (BLUE)  |  (GREEN) |
------------------------
*/

// These are the definitions for the regions.
#define SIMON_DISPLAY_REGION_0 0
#define SIMON_DISPLAY_REGION_1 1
#define SIMON_DISPLAY_REGION_2 2
```

```
58 #define SIMON_DISPLAY_REGION_3 3
59
60 // These are the definitions for the buttons.
61 #define SIMON_DISPLAY_BUTTON_0 0
62 #define SIMON_DISPLAY_BUTTON_1 1
63 #define SIMON_DISPLAY_BUTTON_2 2
64 #define SIMON_DISPLAY_BUTTON_3 3
65
66 int8_t simonDisplay_computeRegionNumber(int16_t x, int16_t y);
67
68 // Draws a colored "button" that the user can touch.
69 // The colored button is centered in the region but does not fill the region.
70 void simonDisplay_drawButton(uint8_t regionNumber);
71
72 // Convenience function that draws all of the buttons.
73 void simonDisplay_drawAllButtons();
74
75 // Draws a bigger square that completely fills the region.
76 // If the erase argument is true, it draws the square as black background to "erase" it.
77 void simonDisplay_drawSquare(uint8_t regionNo, bool erase);
78
79 // Runs a brief demonstration of how buttons can be pressed and squares lit up to implement the
   user
80 // interface of the Simon game. The routine will continue to run until the touchCount has been
   reached, e.g.,
81 // the user has touched the pad touchCount times.
82
83 // I used a busy-wait delay (utils_msDelay) that uses a for-loop and just blocks until the time
   has passed.
84 // When you implement the game, you CANNOT use this function as we discussed in class.
   Implement the delay
85 // using the non-blocking state-machine approach discussed in class.
86 void simonDisplay_runTest(uint16_t touchCount);
87
88 #endif /* SIMON_H_ */
89
90 #endif /* SIMONDISPLAY_H_ */
91
```