

```

1  #include <stdio.h>
2  #include "system.h"
3  #include <unistd.h>
4  #include <inttypes.h>
5  #include <stdlib.h>
6  #include "lcdAPI.h"
7
8  volatile uint16_t backBuffer[X_SIZE*Y_SIZE];
9
10 volatile unsigned int *frontBufferControl = (unsigned int*)FRONT_BUFFER_CONTROLLER;
11 volatile unsigned int *backBufferControl = (unsigned int*)BACK_BUFFER_CONTROLLER;
12 volatile unsigned int *resolutionControl = (unsigned int*)RESOLUTION_CONTROLLER;
13 volatile unsigned int *statusControl = (unsigned int*)STATUS_CONTROLLER;
14
15 volatile unsigned int *touchStatus = (unsigned int*)TOUCH_STATUS;
16 volatile unsigned int *touchPosition = (unsigned int*)TOUCH_COORDINATES;
17
18 /**
19  * lcd_init
20  * Purpose: initializes the LCD screen
21  * args:
22  *     nothing
23  * returns:
24  *     nothing
25  */
26 void lcd_init(){
27
28     *backBufferControl = backBuffer;
29     //swap front and back buffer
30     *frontBufferControl = backBuffer;
31     while((( *statusControl)&BUFFER_SWAP_STATUS)&NOT_SWAPPED){
32
33     }
34     *backBufferControl = backBuffer;
35
36 }
37
38 /**
39  * touch_init
40  * Purpose: initializes the touch capabilities of the screen
41  * args:
42  *     nothing
43  * returns:
44  *     nothing
45  */
46 void touch_init(){
47     *touchStatus |= TOUCH_ENABLE;
48 }
49
50 /**
51  * poll_position
52  * Purpose: To poll for new coordinates
53  * args:
54  *     coords - an array to hold both coordinates
55  * returns:
56  *     X and Y coordinates in the coords array
57  */
58 void poll_position(int* coords){
59     coords[0] = (*touchPosition & (X_COORD));
60     coords[1] = (*touchPosition & (Y_COORD))>>Y_CORRECTION;
61     usleep(100);
62 }
63
64 /**
65  * lcd_fill
66  * Purpose: Fills the entire LCD screen
67  * args:
68  *     color - the 24-bit color code
69  * returns:

```

```

70      *      nothing
71      */
72 void lcd_fill(unsigned int color){
73     unsigned int blue = (color&BLUE_MASK)>>3;
74     unsigned int green = (color&GREEN_MASK)>>10;
75     unsigned int red = (color&RED_MASK)>>19;
76     unsigned int adjColor = blue|(green<<5)|(red<<11);
77
78     for(int i = 0; i < (X_SIZE*Y_SIZE); i++){
79         backBuffer[i]=adjColor;
80     }
81
82 }
83
84 /**
85  * set_pixel
86  * Purpose: Colors one pixel at the given coordinates with the given 24-bit color
87  * args:
88  *     x - the X coordinate
89  *     y - the Y coordinate
90  *     color - the 24-bit color code
91  * returns:
92  *     nothing
93  */
94 void set_pixel(unsigned int x, unsigned int y, unsigned int color){
95     unsigned int blue = (color&BLUE_MASK)>>3;
96     unsigned int green = (color&GREEN_MASK)>>10;
97     unsigned int red = (color&RED_MASK)>>19;
98     unsigned int adjColor = blue|(green<<5)|(red<<11);
99     backBuffer[(y*320)+x]=adjColor;
100 }
101
102 /**
103  * draw_rectangle
104  * Purpose: Draws a rectangle starting at a given x and y of given length and width. It
105  * will either fill or outline
106  *     the rectangle in a given 24-bit color
107  * args:
108  *     x - the starting X coordinate
109  *     y - the starting Y coordinate
110  *     width - the width of the rectangle
111  *     height - the height of the rectangle
112  *     color - the 24-bit color code
113  *     fill - 1 to fill, else outline
114  * returns:
115  *     nothing
116  */
117 void draw_rectangle(unsigned int x, unsigned int y, unsigned int width, unsigned int
height, unsigned int color, int fill){
118     unsigned int blue = (color&BLUE_MASK)>>3;
119     unsigned int green = (color&GREEN_MASK)>>10;
120     unsigned int red = (color&RED_MASK)>>19;
121     unsigned int adjColor = blue|(green<<5)|(red<<11);
122     if(fill == 1){
123         for(int j = 0; j < height; j++){
124             for(int i = (((y+j)*320)+x); i < (((y+j)*320)+x)+width; i++){
125                 backBuffer[i]=adjColor;
126             }
127         }
128     } else {
129         for(int j = 0; j < height; j++){
130             for(int i = (((y+j)*320)+x); i < (((y+j)*320)+x)+width; i++){
131                 if(j == 0 | i == (((y+j)*320)+x) | i == (((y+j)*320)+x)+width-1 | j
== (height-1)){
132                     backBuffer[i]=adjColor;
133                 }
134             }
135         }
136     }
137 }

```

