Lab 9
Jacob Hillebrand
CEE-345 Microprocessor System Design
Switch Controlled RGB LEDs

This lab was an exploration into direct physical control of the FreedomBoard's LEDs, as well as a look into taking button input. 3 pushbutton switches were wired up to the FreedomBoard, which were used to control the Red, Green, and Blue LEDs on the Freedomboard.

In order to control this, the PORTE registers were set up to take input from the pushbuttons, the PORTD and PORTB registers were set up for LED output. In addition several header files were set up with LED pin and register values for use in prepping the outputs. When it was detected that the button corresponding to the proper LED was pushed, that LED's register was set to output, turning on the LED. When the button was released, the LED was then turned off. The code for this program was split into 7 different files, shown below.

```c
/*CEE-345 Microprocessor System Design
This code is to demonstrate the use of General Purpose Input and Output
ports on the Kinetic ARM processor. Three Tactile switches are used to control
a three color (RGB) LED on the Freedom board. By pressing each tact switch, users
will see RED, GREEN, and BLUE light on the LED.
 *-----------------------------------------------------------------------------*/
#include <MKL25Z4.H>
#include "gpio_defs.h"
#include "LEDs.h"
#include "switches.h"

void Test_Switches(void) {
        unsigned int switch_code;            // declare a parameter/variable: unsigned

        while (1) {
                switch_code = ~READ_SWITCHES;                    //read values from RE



                Control_RGB_LEDs( (switch_code & MASK(SW_UP_POS)),
                                  (switch_code & MASK(SW_LT_POS)),
                                  (switch_code & MASK(SW_RT_POS)) );
        }
}

/*-----------------------------------------------------------------------------
  MAIN function
 *-----------------------------------------------------------------------------*/
int main (void) {

         Init_5way_Switch();                  //add code here to initialize the tactil
         Init_RGB_LEDs();                                 //add code here to initi
         Test_Switches();                                 //Read the switch value

}
```

Figure 1: Main.c

```
#ifndef SWITCHES_H
#define SWITCHES_H

// All switches are on port E
#define SW_UP_POS (21)
#define SW_LT_POS (30)
#define SW_RT_POS (23)

// Macro to read switches returns state switches, active low
#define READ_SWITCHES  (PTE->PDIR)

// Function prototypes
extern void Init_5way_Switch(void);

#endif
// ********************************************************************
~
```

Figure 2: switches.h

```
#include <MKL25Z4.H>
#include "switches.h"
#include "gpio_defs.h"

void Init_5way_Switch(void) {
  SIM->SCGC5 |=  SIM_SCGC5_PORTE_MASK; /* enable clock for port E */

  /* Select GPIO and enable pull-up resistors for pins connected to switches */
  PORTE->PCR[SW_UP_POS] |= PORT_PCR_MUX(1) | PORT_PCR_PS_MASK | PORT_PCR_PE_MASK;
  PORTE->PCR[SW_LT_POS] |= PORT_PCR_MUX(1) | PORT_PCR_PS_MASK | PORT_PCR_PE_MASK;
  PORTE->PCR[SW_RT_POS] |= PORT_PCR_MUX(1) | PORT_PCR_PS_MASK | PORT_PCR_PE_MASK;

  /* Set port E bits 0-3, 7 to inputs */
  PTE->PDDR &= ~( MASK(SW_UP_POS) |
                  MASK(SW_LT_POS) |
                  MASK(SW_RT_POS));
}
```

Figure 3: switches.c

```
#ifndef LEDS_H
#define LEDS_H

// Freedom KL25Z LEDs
#define RED_LED_POS (18)              // on port B
#define GREEN_LED_POS (19)      // on port B
#define BLUE_LED_POS (1)              // on port D

// function prototypes
void Init_RGB_LEDs(void);
void Control_RGB_LEDs(unsigned int red_on, unsigned int green_on, unsigned int blue_on);

#endif
// ********************************************************************
~
```

Figure 4: LEDs.h

```c
#include <MKL25Z4.H>
#include "LEDs.h"
#include "gpio_defs.h"

void Init_RGB_LEDs(void) {
	// Enable clock to ports B and D
	SIM->SCGC5 |= SIM_SCGC5_PORTB_MASK | SIM_SCGC5_PORTD_MASK;;

	// Make 3 pins GPIO
	PORTB->PCR[RED_LED_POS] &= ~PORT_PCR_MUX_MASK;
	PORTB->PCR[RED_LED_POS] |= PORT_PCR_MUX(1);
	PORTB->PCR[GREEN_LED_POS] &= ~PORT_PCR_MUX_MASK;
	PORTB->PCR[GREEN_LED_POS] |= PORT_PCR_MUX(1);
	PORTD->PCR[BLUE_LED_POS] &= ~PORT_PCR_MUX_MASK;
	PORTD->PCR[BLUE_LED_POS] |= PORT_PCR_MUX(1);

	// Set ports to outputs
	PTB->PDDR |= MASK(RED_LED_POS) | MASK(GREEN_LED_POS);
	PTD->PDDR |= MASK(BLUE_LED_POS);
}

void Control_RGB_LEDs(unsigned int red_on, unsigned int green_on, unsigned int blue_on) {
	if (red_on) {
			PTB->PCOR = MASK(RED_LED_POS);
	}
	else {
		PTB->PSOR = MASK(RED_LED_POS);
	}
	if (green_on) {
		PTB->PCOR = MASK(GREEN_LED_POS);
	}
	else {
		PTB->PSOR = MASK(GREEN_LED_POS);
	}
	if (blue_on) {
			PTD->PCOR = MASK(BLUE_LED_POS);
	}	else {
			PTD->PSOR = MASK(BLUE_LED_POS);
	}
}
// ********************************************************
```

Figure 5: LEDs.c

```c
#ifndef GPIO_DEFS_H
#define GPIO_DEFS_H

// basic light switch
#define LED1_POS (1)    // on port A
#define LED2_POS (2)    // on port A
#define SW1_POS (5)             // on port A

// Define a Macro to do left shifting where UL is unsigned long - 32 bits
#define MASK(x) (1UL << (x))

// Speaker output   //speaker is not used in this lab
#define SPKR_POS (0)    // on port C

#endif
// ********************************************************************
```

Figure 6: gpio_defs.h

```c
#include <MKL25Z4.H>

void Delay (uint32_t dly) {
  volatile uint32_t t;

        for (t=dly*10000; t>0; t--)
                ;
}
// ************************************************
```

Figure 7: delay.h