Notebook 0

ELEC 3040/3050 Spring 2021

By Jacob Howard, Undergraduate Computer Engineering

Objective

The objective of this lab was very similar to the previous lab. We had 2 decade counters, but instead of being controlled by switches, they were controlled by buttons. What was mainly new about this lab was to learn and program interrupts.

Pre-Lab

Before Lab, we were supposed to write the code that we would be using. The code consisted of two “decade” counters (counters that count up to 9 and then reset). Counter A, which was to have the ability to count up to 9, and resetting, but also would switch direction and count down from 9 if a button was pushed. Counter B was just supposed to count up to 9 and reset; it did not change directions. The next part, which was the most challenging, was to write interrupts for the microprocessor. We had 2 interrupts that would light up a Green LED when the start button was pushed, and a Red LED when the direction button was pushed. I wrote my entire code before the Lab to ensure time for testing and other tasks the lab required.

In-Lab

In the lab, we were supposed to set up two buttons with our code. I had set up the pins in my code in the Pre-Lab. button 1 (microcontroller I/O port pin PA1) was supposed to turn on and off the entire code. Button 2 would be the button that would switch the direction. Once we had our buttons wired up, we were supposed to wire the outputs to 8 LEDs.

Once everything was set up, we were asked to compile and debug the code. This is where I ran into my first issue. For some reason, the uVision 5 software was giving me errors when trying to run the debugger. This was not because of my code or anything I did incorrectly. In fact, another student in the lab was encountering the same problem. The problem for her persisted until 30 minutes left in the lab and the problem for me just fixed itself without any changes from me with 15 minutes left in the lab. This gave me very little time to debug and test my code.

With about 15 minutes left in the lab, I tried to test my code for functionality and do all the tasks required in the lab. When debugging I encountered some more problems, but these problems were in my code. The first problem was that I could not get the physical buttons on the board to function. Only virtual buttons seemed to work. The next problem was that I had to hold the buttons down in order for the code to run. Lastly, it seems my delay was not functioning correctly. I was unable to fix these problems since I had limited time in the lab left because of the original software error. Other than these problems, my code did seem to function. I do plan on fixing these issues before the next lab. In Figure 1, I captured an instance of my code running on the virtual LEDs.

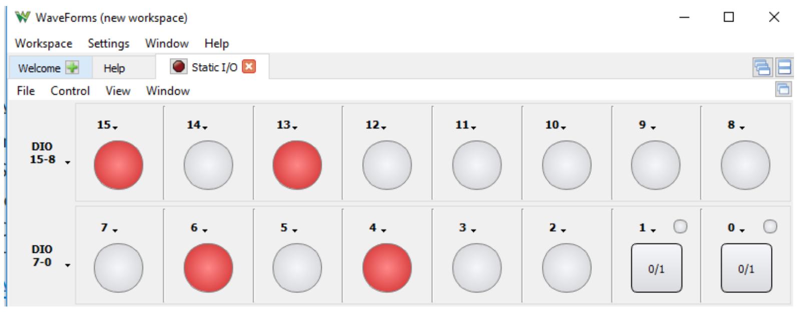

Figure 1

Summary

In conclusion, I felt that this lab was less difficult than the previous lab. Since we had most of the code from the previous lab, if we fixed the errors, then all this lab required was to write interrupts. I do wish the lab manual gave examples and explained how to write interrupts in the code better. Also, I wish that since we are not working with partners like usual in this lab, I wish that this lab section had a make-up lab to fix any errors and test their code. One of the Tas provides a make-up session on Fridays to receive full credit for the lab, but sadly my section does not have a make-up section. This would have been very helpful this week, as the software error, I ran into this week only gave me 15 minutes to debug and test my code in the lab. Maybe for future labs, the professor and my TA will consider this, as it would be very beneficial to all of us. The code I wrote for this lab can be seen below in *Code 1*.

```
/*====================================================*/
/* Jacob Howard */
/* Toggle LED1 while button pressed, with short delay inserted */
/*====================================================*/

#include "stm32l4xx.h" /* Microcontroller information */

/* Define global variables */
//int toggles; /* number of times LED state toggled */
static int counter1, counter2;
unsigned char run, up, blueLED, greenLED;

static uint16_t sw1; //declare 16-bit variable that matches IDR size (Switch 1 to start and stop counters)
static uint16_t sw2; //Switch 2 to reverse counter 1



/*---------------------------------------------------*/
/* Initialize GPIO pins used in the program */
/* PA11 = push button */
/* PB4 = LDR, PB5 = green LED */
/*---------------------------------------------------*/
void PinSetup () {
 /* Configure PA0 as input pin to read push button */
		RCC->AHB2ENR |= 0x01; /* Enable GPIOA clock (bit 0) */
		GPIOA->MODER &= ~(0x03FFFC3C); // General purpose input mode */		//THIS may be wrong. Might be ~0x0000003C.
  GPIOA->MODER |= 0x01555400; // Sets pins 1 and 2 as inputs and pins 5-12 as outputs, and all others as 00.

		RCC->AHB2ENR |= 0x03; //enable GPIOB clock (2bit)
		GPIOB->MODER &= ~(0x03C0); //setting pins 3 and 4 PB
		GPIOB->MODER |= 0x0180; //setting pins 3 and 4 as output pins
}

/*----------------------------------------------------------*/
/* Interrupt settup
/*----------------------------------------------------------*/
void InterruptSetup(){
		SYSCFG->EXTICR[0] &= ~(0x000F); //Clear EXTI0 (set for PA0)
		SYSCFG->EXTICR[0] &= ~(0x00F0); //Clear EXTI1 (set for PA1)

		EXTI->RTSR1 |= 0x0001; //Set rising trigger for PA0
		EXTI->RTSR1 |= 0x0002; //Set rising trigger for PA1

		EXTI->IMR1 |= 0x0001; //Enable interrupt PA0
		EXTI->IMR1 |= 0x0002; //Enable interrupt PA1

		EXTI->PR1 |= 0x001; //Clear pending
		EXTI->PR1 |= 0x002; //Clear pending

		NVIC_EnableIRQ(EXTI1_IRQn); //Enable interrupt for PA0
		NVIC_EnableIRQ(EXTI2_IRQn); //Enable interrupt for PA1

		NVIC_ClearPendingIRQ(EXTI1_IRQn); //Clear NVIC pending for PA0
		NVIC_ClearPendingIRQ(EXTI2_IRQn); //Clear NVIC pending for PA1
}

/*----------------------------------------------------------*/
/* Interrupt Service Routine                              */
/*----------------------------------------------------------*/
void EXTI1_IRQHandler() {
   if (run == 1) {
      run = 0;
   }
   else {
      run = 1;
   }
  GPIOB->ODR = (GPIOB->ODR & ~(0x08)) | (run << 3);//Set to value of run
  NVIC_ClearPendingIRQ(EXTI1_IRQn);
  EXTI->PR1 |= 0x0002;
}

void EXTI2_IRQHandler(){
          if (up == 1) {
                    up = 0;
          }
               else {
                    up = 1;
               }
               GPIOB->ODR = (GPIOB->ODR & ~(0x04)) | (up << 2);//Set to value of run
  NVIC_ClearPendingIRQ(EXTI2_IRQn);
  EXTI->PR1 |= 0x0001;
          /*
          up = 1;
          if(greenLED == 0){
          GPIOB->BSRR = 0x0008;      //Toggle PB4
          greenLED = 1;
          }else{
          GPIOB->BSRR = 0x0008 << 16;   //Toggle PB4
          greenLED = 0;
          }
          EXTI->PR1 |= 0x002;                              //Clear pending
          NVIC_ClearPendingIRQ(EXTI2_IRQn);  //Clear pending
          */
}


/*--------------------------------------------------------*/
/* Function to Count up to 9 and then back to zero
/*--------------------------------------------------------*/

void countUpandDown () {
 if (up == 1) {
               if (counter1 < 9) {
                              counter1++;
                    }
               else {
                         counter1 = 0;
                    }
               }
 else {
               if (counter1 > 0) {
                         counter1--;
                    }
               else {
                    counter1 = 9;
                    }
 }

}

/*--------------------------------------------------------*/
/* Function to Count up to 9 indefinitely
/*--------------------------------------------------------*/

          void count () {
               if (counter2<9) {
                    counter2++;

               }
  else {
  counter2 = 0;
  }
```

```
                }
/*----------------------------------------------------------*/
/* Delay function - 0.5 second delay */
/*----------------------------------------------------------*/
void delay (){
 int a,b;
 for (a=0; a<100; a++) { //outer loop
 for (b=0; b<12000; b++) { //inner loop
 }
 }
}

/*------------------------------------------------*/
/* Main program */
/*------------------------------------------------*/
int main(void) {
                counter1 = 0; //initializing counter values
                counter2 = 0;
                run = 0; //setting initial run to 0
                up = 1; //setting direction as counting up


                //setting blue and green leds as off
                blueLED = 0;
  greenLED = 0;

                PinSetup(); //Configure GPIO pins
                InterruptSetup(); //Configure Interrupts
                __enable_irq(); //Enable CPU interrupts

                while (1) {

                run = (GPIOA->IDR & PWR_PUCRA_PA1) >> 1; //sets switch 1 to PA1 and shifts right 1 bit
  up = (GPIOA->IDR & PWR_PUCRA_PA2) >> 2; //sets switch 2 to PA2 and shifts right 2 bits

                if (run == 1) {
                countUpandDown();
  count();

                delay(); //delay

  // Writes counter1 and counter2 to the output pins
   GPIOA->ODR = (GPIOA->ODR & ~(0x1FE0)) | (((counter2 << 4) + counter1) << 5);
  }
}

}
```
Code 1