TO: Profs. J. Hung

FROM: “Blue” Jacob Howard

SECTION: Tuesday 1:00

DATE: 3/12/21

SUBJECT: Lab 3: Interrupt service routines in C

**Objective:**

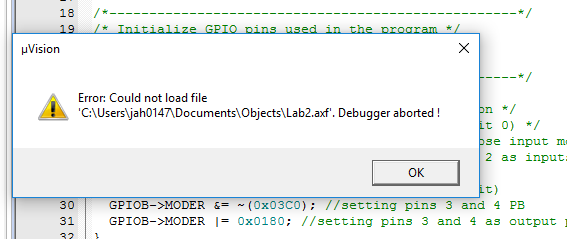
This lab reviews the concept of Interrupt Service Routines. The goal of the lab is to design a C program that will be programmed onto the microcontroller’s GPIO ports as inputs connected to physical buttons. These buttons will start/stop the decade counters and reverse one of the counters. Other GPIO ports are to be programmed as output connected to physical and virtual LEDs. The LEDs will display the status of interrupts and status of counters respectively. You may use the code desiged from the first two labs and edit them for this lab.

**Data:**

When testing and debugging code, you will want to observe the two sets of 4-bit binary counters and that the two external interrupts correctly control the counting. Also, set up the logic analyzer to capture and display the counting sequence of the two 4-bit binary counters and also the state of the two interrupts. An optional part of the lab is to set up the oscilloscope channels to capture the state of the buttons and the LED that is toggled by the interrupt service routine. You may also check the other button with the oscilloscope in order to confirm that the interrupt signals have occurred. The inmortant information to gather here is to verify that both counters are working properly and that the interrupts are being called.

**Procedure:**

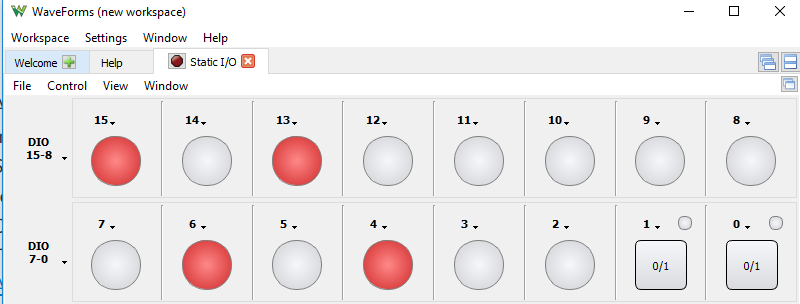
After everything was set up properly, we were to wire up our board and test the code. I ran into a major problem when everything was set up in the lab. For some reason, the uVision 5 software was giving me errors when trying to run the debugger. The error had nothing to do with my code and the TA did not know how to fix the error. Another person in my lab group was having the same issue, but the error went away with 30 minutes left in the lab for her. She did not change anything; the error just finally went away. At one point, I received another type of error when running the debugger, so I decided to screenshot the new error to provide in the report. I do not have a screenshot of the first error. The error for me went away without me changing anything with 15 minutes left in the lab. This gave me very limited time to run and test my code.



**Testing:**

Since I had very little time left in the lab, I was unable to completely test and fix any issues during lab. My code did seem to function, but I did encounter some errors that I did not have time to fix. The first thing I noticed was that the physical buttons were not working for the lab. I had to proceed with virtual buttons to test the rest of the code. The next issue I noticed was that I had to hold down the virtual buttons for them to signal on. I could not press them for a toggle. The last issue I noticed was that my delay() function did not seem to work.

Although I ran into some errors which I could have probably fixed within the lab time, because of a software error that persisted almost all of the lab time, I was not able to make many corrections to fix these issues. It did seem that the counters were functioning fine and the major problems had to do with the delay function and having the buttons toggle instead of hold. I was unable to proceed with the lab from here as the lab time was up. I did screenshot the 8 Leds functioning below at one instance.



**Conclusion:**

In conclusion, I felt that this lab was not as difficult as the last lab, as we already had most of the code from Lab 2 after we corrected any issues. I do wish the Lab Manuel showed and described how to set up interrupts because that was the most challenging part of the lab. I also wish that since this year is different, meaning we do not have lab partners, I wish we had more time or a make-up lab to fix and test any corrections we need to make. I know one of the TAs holds a make-up lab on Fridays for his section. Sadly, my TA does not hold any make-up lab or provide more time. A make-up lab would have been very helpful this week, considering I only had 15 minutes to do all of the lab work and fix any issues. The code I wrote for this lab can be seen below.

|  |
| --- |
| /\*====================================================\*/  /\* 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*