CPE 325: Intro to Embedded Computer System

Lab03

Laboratory Assignment #3,

LEDs, Debouncing, Switches.

Submitted by: Nolan Anderson

Date of Experiment: 09/13/2020

Report Deadline: 09/14/2020

Introduction

This lab covers how to interface with the switches and LEDs that are on the MSP430 microcontroller. We also learn how to debounce and delay for the LEDs to essentially toggle on an off in a specific time frame. We do this by using the _delay_cycles(num); and a while loop.

Theory

Debouncing: Debouncing is using a for loop to perform operations on a specific time range. Coupling this with software delay, every time you go through the for loop, you are waiting the amount of time in _delay_cycles(num), and then toggling the LED, whether it be on or off. Essentially, you use debouncing so that the software delay works properly for a given period of time.

Software Delay: In this lab, I used the _delay_cycles(num); function in C to delay the toggle of the LEDs. Essentially, 50000 in my code is similar to 500ms, or 5hz. So the light is blinking in the debounced for loop every 500ms. Similarly, for the 2hz I used 20000, or 200ms to toggle in the for loop every 200ms, or 2hz.

Results & Observation

Copy the question from the assignment here:

1. **How do you handle debouncing?**

Debouncing is done with decrementing for loop, such as: `for (i = 10000; i > 0; i--);`

2. **How do you create the required delay?**

I created the required delay by using the _delay_cycles(num) function.

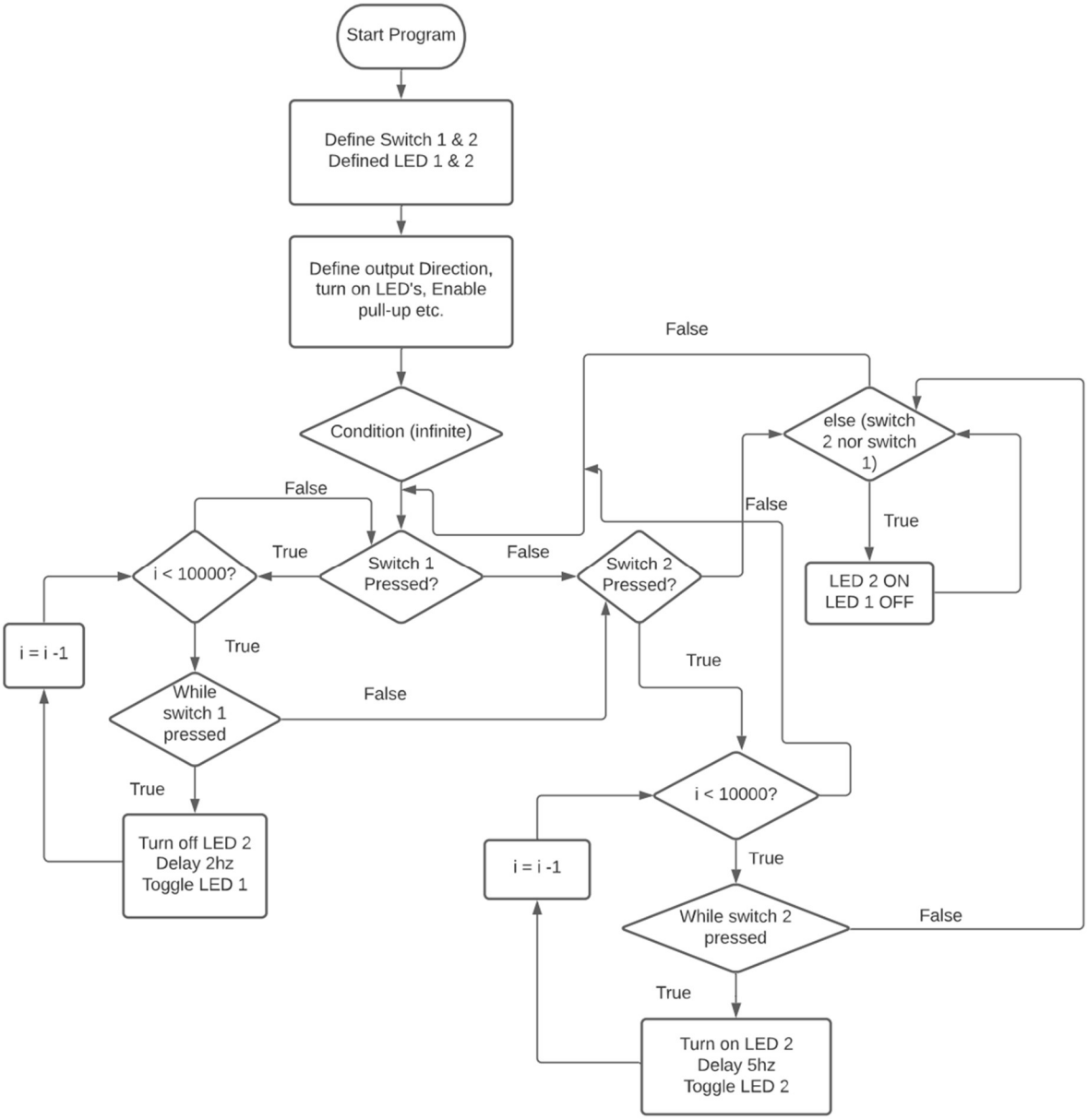
3. **How does your code handle both the switches?**

To handle both of the switches, I used an infinite loop, if statements, a debouncing for loop, and a while loop to hold on while the switches are pressed. I used the initial if statements to scan for the push initially, basically to find which button was pressed. And as long as you are still holding down the button, it will continue in that if statement because there is a for and while loop inside of it.

4. Display the experimenter board to demonstrate the proper functioning on switch presses.

This is in the Demo video at the very end.

Flow Charts:


Start Program
Define Switch 1 & 2
Defined LED 1 & 2
Define output Direction, turn on LED's, Enable pull-up etc.
Condition (infinite)
Switch 1 Pressed?
True
False
i < 10000?
True
False
i = i -1
While switch 1 pressed
True
False
Turn off LED 2
Delay 2hz
Toggle LED 1
Switch 2 Pressed?
True
False
else (switch 2 nor switch 1)
True
False
LED 2 ON
LED 1 OFF
i < 10000?
True
False
i = i -1
While switch 2 pressed
True
False
Turn on LED 2
Delay 5hz
Toggle LED 2

Results Screenshots/Pictures:

Results are not applicable here since there was no console output. It shows the LED's blinking correctly in the demo video.

Observations:

Overall, this lab was similarly pretty tough for me. It was mainly figuring out how to get all the loops working together, but after some time it was pretty straight forward. Essentially grabbing the first click and then using a for loop to keep on going until the user lets go of the button and waits for the next click.

Conclusion

Write your conclusion. (Explain what you have learnt and issues you faced)

This lab really taught me a lot about how to use if, for, and while statements to my advantage. Learning debouncing is a very useful skill and I will probably implement it in other areas. I also learned how to interface with the MSP430 beyond just running code on it. Being able to actually use the board and its functions is very useful.

Video Direct Link:

https://drive.google.com/file/d/1lJpddJacg0gKlmyCM8hDQAR4T9NUCKvp/view?usp=sharing

Videos Folder Link:

https://drive.google.com/drive/folders/1_Y3ABMDhCUxc9phtQ8JKHCM8LDOK7k7J?usp=sharing

Appendix

```
///*-------------------------------------------------------------------------------
//* Student:      Nolan Anderson
//* Program:      Lab_3_1.c
//* Date:         Aug 21, 2121
//* Input:        None
//* Output:       No output, but output of the LED's on the board.
//* Description:  This c program interfaces with switches and LED's on the MSP430
//*               By accessing the pins on the board itself. Also uses debouncing
//*               and program delay in order to blink the leds.
//*-------------------------------------------------------------------------------*/
#include <msp430.h>                 // Including the board directive.
#define S1PUSH ((P2IN&BIT1) == 0)   // Here I am defining the Switch 1 and switch 2 for if they are pushed.
#define S2PUSH ((P1IN&BIT1) == 0)
#define RED 0x01                    // Here I am defining the red and green LED's for their pin values on the board
#define GREEN 0x80                  // LED 1 = RED, LED 2 = GREEN

void main(void)
{
    WDTCTL = WDTPW + WDTHOLD;       // STOP Watchdog timer

    P1DIR |= RED;                   // P1.0 is output direction for 0x01, or red.
    P1REN |= BIT1;                  // Enable the pull-up resistor at P1.1
    P1OUT |= BIT1;                  // LED is on.
    P4DIR |= GREEN;                 // P4.7 is output for 0x80, or green.
    P2REN |= BIT1;                  // Enable pull-up resistor at P2.1
    P2OUT |= BIT1;                  // LED is on
    int i = 0;                      // Loop counter
    for( ; ; )                                  // infinite loop so that it is always reading input
    {
        if (S1PUSH)                             // If switch 1 is pressed
        {
            for (i = 10000; i > 0; i--);               // Loop through until it runs out,
            while (S1PUSH && !S2PUSH)                   // While Switch 1 is pressed
            {
                P4OUT &= ~GREEN;                // Green LED stays off.
                __delay_cycles(500000);         // This is a delay cycle for the hz rate. so every time the loop goes through
                                                // It delays 2hz
                P1OUT ^= RED;                   // Toggle the red LED
            }
        }
        else if (S2PUSH)                        // If switch 2 is pressed.
        {
            for (i = 10000; i > 0; i--);        // For all of these values, loop through from 10,000->0. Basically will never run out.
            while(S2PUSH && !S1PUSH)
            {
                P1OUT |= RED;                   // Here, I turn on the Red LED
                __delay_cycles(200000);         // This is a delay cycle for the hz rate. so every time the loop goes through
                                                // It delays by 5hz
                P4OUT ^= GREEN;                 // And I toggle the green LED
            }
        }
        else                            // Else, for all times that the button is not pressed, do the following:
        {
        P4OUT |= GREEN;                 // Here, the Green LED is on,
        P1OUT &= ~RED;                  // And the Red LED is off
                                        // Until a something changes on the switches.
        }
    }
}
```