Jose Capestany

Sayaf Almeri

EE215

4/14/2016

LAB7 Report

The purpose of this lab was to use the accelerometer inside the MSP430 and hook it up to an A/D convertor. We are then supposed to use the output of the convertor to control when two different LEDs are on and what their flash frequency is supposed to be set to.

**Flowchart:**

The following flowchart is a simple representation of what we intend our code to follow. The accelerometer is connected to an A/D convertor. Then depending on how the accelerometer is tilted the LEDs will turn on and off as shown in our flowchart.

C:\Users\Jose Capestany\Downloads\Untitled Diagram.png

**Conclusion:**

This program was pretty hard to figure out. We had to first figure out how to enable the A/D convertor, and that took a while by itself. We were able to get the LEDs to turn on and off appropriately, but not change their frequency. We then realized we should be changing the Timer A1 module as that controls our frequency. We then set the frequency to a formula using the A/D convertor. Overall, it was a useful lab in order to familiarize with the A/D convertor.

**APPENDIX: CODE**

#include <msp430.h>

int main(void){

    WDTCTL=WDTPW|WDTHOLD; //Turn off watchdog timer

    P1DIR=0X03; // Various Bits and pin selection

    P6OUT|=BIT0;

    P6DIR|= BIT0;

    P6SEL|= 0X0E;

    P6DIR &= ~0X0E;

            TA1CTL = TASSEL\_1 | TACLR | MC\_1; //Initialize Timer A1, used to refresh and toggle LEDs

        TA1CCR0 = 65000;

            ADC12CTL0 |= ADC12ON | ADC12MSC; // Initialize the ADC accelerometer

            ADC12CTL1 |= ADC12CSTARTADD\_1 | ADC12CONSEQ\_1;

            ADC12CTL1 |= ADC12SHS\_0 | ADC12SHP | ADC12SSEL\_0;

            ADC12MCTL0 = ADC12INCH\_0; //3 outputs (X Y Z)

            ADC12MCTL1 = ADC12INCH\_1;

            ADC12MCTL2 = ADC12INCH\_2 | ADC12EOS;

            ADC12CTL0 |= ADC12ENC;

            while(1) { //Runs program forever and ever and ever

                ADC12CTL0|=ADC12SC; //Set sampling bits

                //Not tilted (small margin of error)

                if((ADC12MEM1 >= 0x07D4)&&(ADC12MEM1 <= 0x07EA)){

                    P1OUT |= BIT0 | BIT1;

                }

                //Tilted Left

                else if((ADC12MEM1 > 0x07EA) && ((TA1CTL & BIT0) ==1 )){

                    P1OUT ^= BIT0;

                    P1OUT &= ~BIT1;

                    TA1CTL &= ~BIT0;

                }

                //Tilted Right

                else if((ADC12MEM1 < 0x07D0) && ((TA1CTL & BIT0) ==1 )){

                    P1OUT ^= BIT1;

                    P1OUT &= ~BIT0;

                    TA1CTL &= ~BIT0;

                }

                //Change the frequenzy of Timer A1 as long as it is busy

                while((ADC12CTL1 & ADC12BUSY) == 1) {}

                               TA1CCR0 = 40000 - 100\*abs(0x07EA-ADC12MEM1);

            }

}