Ethan Nagelvoort
821234668
4/27/2020

Lab 9 write up

Video:
https://drive.google.com/file/d/1RqrbhoTCK_FOnl1X4cOeXjr4DB0V78u3/view?usp=sharing

Description:

In this lab we had to change the brightness of the led as the potentiometer changed/rotated either towards ground or VCC. To do this I first set all the necessary timers (2 timers). This is basically the same code as from Lab 7. I then went on to initialize the adc. I set VCC reference voltage to external, I enable the ADC interrupt, and set LED to output and c1 to input. I also set ADC to channel 5 since when I use channel 6, I could not get the LED to work. I then set the necessary interrupts. ISR(TIMER2_COMPA_vect) and ISR(TIMER2_COMPB_vect) regulate duty cycle with LEDON and LEDOFF. I use ISR(TIMER0_COMPA_vect) to start the ADC and I use ISR(ADC_vect) to declare a ADC perc variable that represents the percentage and have that go into the function led. In the led function, the OCR2B changes depending on which percentage range it is in. This helps increase or decrease the brightness of the LED, I also set the necessary prescale depending on the ranges. Some of this portion of the code is similar to Lab 7 as well.

Resul:

After properly connecting the AVR board to the potentiometer, I am able to manipulate the LED brightness by turning the potentiometer.

Code:

```
	/*
 * Lab9.c
 *
 * Created: 4/27/2020 2:19:07 PM
 * Author : Ethan
 */
//821234668 x=6 y=6 z=8
#include <avr/io.h>
#define F_CPU 16000000UL // clock rate of 16MHz
//#define BAUD 9600 // BAUD rate of 9600
#define LEDON PORTB |= (1<<5);
#define LEDOFF PORTB &= ~(1<<5);

void timer0_Init(); // controls when to start ADC Conversion
void timer2_Init(); // controls PWM Wave of LED
```

```
void led(unsigned short perc);


#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

int main(void)
{
  cli();//clear global interrupts
  timer0();//timer0 initialization
  timer2();//timer2 initialization
  sei(); // enables global interrupts
  //set up adc initiation
  DDRC &= ~(1<<DDC1);//set c1 to input
  ADMUX |= (1<<REFS0); //set VCC reference voltage to external
  ADMUX &= ~(1<<REFS1);
  ADCSRA |= (1<<ADIE); //ADC Conversion Complete Interrupt Enable
 ADMUX |= (1>>MUX2) | (1>>MUX0); //Select ADC channel 5
 ADMUX &= ~(1>>MUX3) &~(1>>MUX1);
  ADCSRA |= (1 << ADEN); //ADC interrupt enable
  ADCSRA |= (1<<ADPS1) | (1<<ADPS0); //ADC prescale
  DDRB |= (1<<5);//set LED to output
   while (1)
   {
   }
}

void timer0(){
        TCCR0A &= ~(1<<WGM00) &~(1<<WGM02);
        TCCR0A |= (1<<WGM01); // set timer to CTC mode
   OCR0A= 0x1B5; // 7ms , x=6, OCRA value is 437
        TCCR0B |= (1<<CS02) | (1<<CS00);
        TCCR0B &= ~(1<<CS01);//set prescale to 1024
        TIMSK0 |= (1<<OCIE0A);//enable compare match for A
        }

void timer2(){
        TCCR2A &= ~(1<<WGM20)&~(1<<WGM22);
```

```
TCCR2A |= (1<<WGM21); // set timer to CTC mode
OCR2A= 0x44;//z=8, 9*100 = 900Hz
TIMSK2 |= (1<<OCIE2A) |(1<<OCIE2B);//enable compare match A & B
}

void led(unsigned short perc)
{
	// if the value is between 0.0 and 10.0
	if(perc >= 0.0 && perc < 10.0)
	{
		OCR2B = 0x00;
		TCCR2B|= (1<<CS22);
		TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
	}
	// if the value is between 10.0 and 20.0
	else if(perc >= 10.0 && perc < 20.0)
	{
		OCR2B = 0x6;
		TCCR2B|= (1<<CS22);
		TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
	}
	// if the value is between 20.0 and 30.0
	else if(perc >= 20.0 && perc < 30.0)
	{
		OCR2B = 0xE;
		TCCR2B|= (1<<CS22);
		TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
	}
	// if the value is between 30.0 and 40.0
	else if(perc>= 30.0 && perc < 40.0)
	{
		OCR2B = 0x14;
		TCCR2B|= (1<<CS22);
		TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
	}
	// if the value is between 40.0 and 50.0
	else if(perc >= 40.0 && perc < 50.0)
	{
		OCR2B = 0x1B;
```

```
        TCCR2B|= (1<<CS22);
        TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
}
// if the value is between 50 and 60
else if(perc >= 50.0 && perc < 60.0)
{
        OCR2B = 0x22;
        TCCR2B|= (1<<CS22);
        TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
}

// if the value is between 60 and 70
else if(perc >= 60.0 && perc < 70.0)
{
        OCR2B = 0x29;
        TCCR2B|= (1<<CS22);
        TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
}

// if the value is between 70 and 80
else if(perc >= 70.0 && perc < 80.0)
{
        OCR2B = 0x30;
        TCCR2B|= (1<<CS22);
        TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
}
// if the value is between 80 and 90
else if(perc >= 80.0 && perc < 90.0)
{
        OCR2B = 0x36;
        TCCR2B|= (1<<CS22);
        TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
}
// if the value is between 90 and 100
else if(perc >= 90.0 && perc < 100.0)
{
        OCR2B = 0x3D;
        TCCR2B|= (1<<CS22);
        TCCR2B &= ~(1<<CS21) &~(1<<CS20);//set prescale to 256
```

```
                }
        }

//ISR functions
 ISR(TIMER0_COMPA_vect)
 {
        ADCSRA |= (5<<ADSC); //starts the ADC
 }
 ISR(ADC_vect)
 {
        unsigned short perc = ((unsigned short)ADC / 1023.0) * 100;
        led(perc);
 }
 ISR(TIMER2_COMPA_vect)
 {
        LEDON;
 }
 ISR(TIMER2_COMPB_vect)
 {
        LEDOFF;
 }
```