Nick Harris

CPE 301

Design Assignment 3B

#define *F\_CPU* 8000000UL

#define BAUD\_RATE 9600

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#include <stdlib.h>

#include <stdio.h>

void usart\_init ();

void usart\_send (unsigned char ch);

void USART\_putstring(char\* StringPtr);

char buffer[5];

int main (void)

{

INITI\_USART();

INITI\_Timer1();

ADCSRA= 0x86; //make ADC enable and select ck/64

ADMUX= 0xC8; //1.1V Vref, temp, right-justified, internal temp. sensor

DDRB = (1<<5);

DDRD = 0x00;

ADCSRA |= (1<<ADSC);

sei();

while (1);

return 0;

}

//intialize the USART

void INITI\_USART(void)

{

UCSR0B = (1<<TXEN0);

UCSR0C = (1<< UCSZ01)|(1<<UCSZ00);

UBRR0L = *F\_CPU*/8/BAUD\_RATE-1;

}

void usart\_send(unsigned char ch)

{

while (! (UCSR0A & (1<<UDRE0))); //wait until UDR0 is empty

UDR0 = ch; //transmit ch

}

//organize string into bits for sending

void USART\_putstring(char\* StringPtr)

{

while (\*StringPtr != 0x00) {

usart\_send(\*StringPtr);

StringPtr++;

}

}

void INITI\_Timer1()

{

//Set up timer with a prescale of 64

TCCR1A |= (0<<COM1A1) | (0<<COM1A0);

TCCR1B |= (1<<CS11)|(1 << CS10);

//Initialize counter

TCNT1 = 0;

//Enable overflow interrupt

TIMSK1 |= (1 << TOIE1);

//Initialize overflow counter variable

time\_ovf = 0;

}

ISR(ADC\_vect)

{

PORTD = ADCL;

PORTB = ADCH;

ADCSRA|=(1<<ADSC); //start conversion

while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish

ADCSRA |= (1<<ADIF);

int a = ADCL;

a = a | (ADCH<<8);

a -= 289;

*itoa*(a, buffer, 10);

USART\_putstring(buffer);

USART\_putstring("\n");

*\_delay\_ms*(1000);

}

ISR(TIMER1\_OVF\_vect)

{

//Keep track of number of overflows

time\_ovf++;

}