CPE 301 – SPRING 2015

DESIGN ASSIGNMENT 3

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **SUBMISSION ITEM** | **COMPLETED (Y/N)** | **MARKS**  **(/MAX)** |
| 0. | AVR C Code that compiles and works | Y |  |
| 1. | Schematics | Y |  |
| 2. | Snapshot of board with connected components | Y |  |
| 3. | Link to YouTube Video | Y |  |

**0 - AVR C Code**

//Emmanuel Sanchez

//CPE 301 - DA3

//USART, ADC, temp sensor

#define F\_CPU 16000000UL //16 MHz clock

#include <avr/io.h>

#include <avr/interrupt.h>

void usart\_init (void) //initialize USART

{

UCSR0B = (1<<TXEN0); //transmit enable

UCSR0C = ((1<<UCSZ01)|(1<<UCSZ00)); //asynch mode

UBRR0L = 0x67; //baud rate 9600 at 16 MHz

}

void usart\_tx\_string(char \*data) //takes a string and sends it serially

{

while(\*data != '\0'){ //send chars until NULL is found

while(!(UCSR0A & (1<<UDRE0))); //wait for UDRE0 to be 1

UDR0 = \*data; //send char serially

data++; //point to next char

}

}

int main(void)

{

DDRC &= ~(1<<PORTC0); //PC0 is analog input

usart\_init(); //initialize USART

ADCSRA = 0x87; //set ADEN, ADC prescaler 128

ADMUX = 0xC0; //select channel 0 (PC0)

sei(); //enable interrupts (needed for timer overflow interrupt)

//configure timer 1 to interrupt every second

TCNT1 = 65536 - ((double)F\_CPU/256); //overflow in 1 sec

TCCR1A = 0; //normal mode

TCCR1B = 4; //prescaler = 256

TIMSK1 |= (1<<TOIE1); //interrupt on overflow

while(1); //wait for interrupts

return 0;

}

ISR(TIMER1\_OVF\_vect) //timer1 overflow ISR

{

TCCR1B = 0; //stop timer 1

TIFR1 = 1; //clear overflow flag

int adc\_temp; //stores ADC temporarily

float adc\_tempf; //float for calculations

int adc\_tempi; //integer part

int adc\_tempd; //decimal part

//read ADC

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

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

adc\_temp = ADC; //save ADC value

adc\_tempf = (float)adc\_temp \* (1.1 / 1024) / 0.01; //ADC value \* resolution / scale factor of LM34 (10mV/deg F)

adc\_tempi = (int)adc\_tempf; //integer part

adc\_tempf = adc\_tempf - adc\_tempi; //subtract integer part

adc\_tempd = (int)(adc\_tempf \* 10); //decimal part

char intpart[10]; //stores temperature to be displayed

sprintf(intpart, "%d", adc\_tempi); //integer part to string

strcat(intpart, "."); //concatenate dot

char decpart[4]; //stores decimal part

sprintf(decpart, "%d\r\n", adc\_tempd); //decimal part to different string

strcat(intpart, decpart); //concatenate integer and decimal parts

usart\_tx\_string(intpart); //send temperature serially

//re-configure timer 1 to interrupt every second

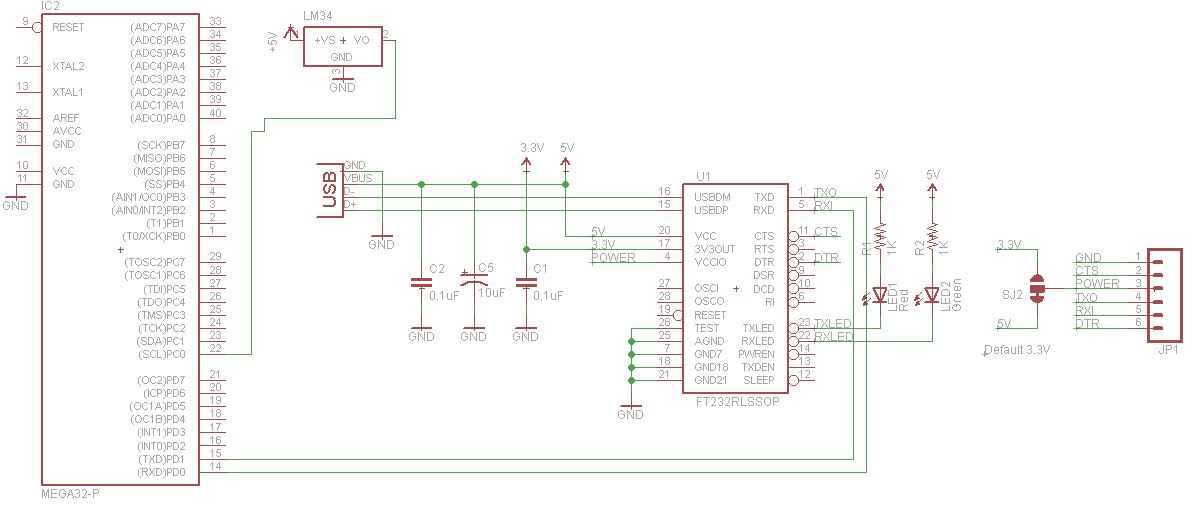
TCNT1 = 65536 - ((double)F\_CPU/256); //overflow in 1 sec

TCCR1A = 0; //normal mode

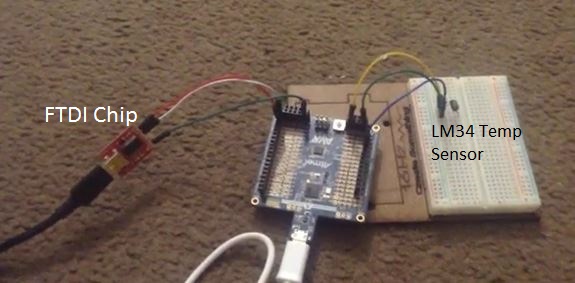
TCCR1B = 4; //prescaler = 256

return;

}

**1 - Schematics**

**2 – Snapshot of board with connected components:**

****

**3 - Link to YouTube Video:**

[**https://www.youtube.com/watch?v=ebXu3cF0Vjg**](https://www.youtube.com/watch?v=ebXu3cF0Vjg)