Andrew Wells

CPE301 – SPRING 2016

Design Assignment 3

**DO NOT REMOVE THIS PAGE DURING SUBMISSION:**

The student understands that all required components should be submitted in complete for grading of this assignment.

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **SUBMISSION ITEM** | **COMPLETED (Y/N)** | **MARKS**  **(/MAX)** |
| 0. | COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS |  |  |
| 1. | INITIAL CODE OF TASK 1/A |  |  |
| 2. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B |  |  |
| 3. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C |  |  |
| 4. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D |  |  |
| 5. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E |  |  |
| 6. | SCHEMATICS |  |  |
| 7. | SCREENSHOTS OF EACH TASK OUTPUT |  |  |
| 8. | SCREENSHOT OF EACH DEMO |  |  |
| 9. | VIDEO LINKS OF EACH DEMO |  |  |
| 10. | GOOGLECODE LINK OF THE DA |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 0. | COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS |  |  |

Xplained Mini: Atmega328P Micro Controller

LM-34 Temperature Sensor

USB to Serial Adaptor

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | INITIAL CODE OF TASK 1/A |  |  |

#define *F\_CPU* 8000000UL // Set clock to 8MHz

#define UBRR\_9600 51 //For 8Mhz

#include <avr/io.h>

#include <stdio.h>

#include <util/delay.h>

#include <avr/interrupt.h>

void ADCstart(); // Function to initialize the ADC

void read\_adc(); // Function to read from the ADC

void USARTstart(unsigned int ubrr); // Funciton to initialize the USART port

void USART\_tx\_string(char \*data); // Function to transmit the value ove the serial port

void TIMER1\_init(); // Function to initialize timer 1

volatile unsigned int ADCvalue; // Variable to dave the temperature value

volatile unsigned int X; // variable to keep track of how many values have been read

char outs[20]; // output buffer

float Ratio; // variable for calculating the temperature

int main(void)

{

Ratio = 500.0/1024.0; // Ratio to calcuate the actual temperature

ADCstart(); // initialize the ADC port

USARTstart(UBRR\_9600); // initialize the USART port

TIMER1\_init(); // Initialize timer 1

sei (); // enable global interuupts

while (1) // infinite loop to transmite the temperature over serial port

{

X = 4; // number of values taken before transmission

ADCvalue = 0; // reset temperature value

while(X) // stay here until 4 reading are taken

{

}

ADCvalue = ADCvalue/4;// average of values recieved

*snprintf*(outs, sizeof(outs),"%3d\r\n", ADCvalue); // convert value to a string

USART\_tx\_string(outs); // transmit value over serial connection

}

return 0;

}

void read\_adc() // function to read the value from the sensor

{

ADCSRA |= (1<<ADSC);// write one to ADSC bit to start conversion

while(ADCSRA & (1<<ADSC))// wait until conversion is complete

{

}

ADCvalue += (Ratio\*ADC);// save temperature

return;

}

void ADCstart()

{

ADMUX = (0<<REFS1)| // Reference Selection Bits

(1<<REFS0)| //use AVcc as reference

(0<<ADLAR)| //ADC Left adjust Result

(0<<MUX2)| // Analog Channel Selection Bits

(1<<MUX1)| // ADC2 (PC2, PIN25)

(0<<MUX0);

ADCSRA = (1<<ADEN)| //ADC Enable

(0<<ADSC)| //ADC Start Conversion

(0<<ADIF)| //ADC Auto Trigger Enable

(0<<ADIE)| //ADC Interrupt Flag

(0<<ADATE)| //ADC Interrupt Enable

(1<<ADPS2)| //ADC Prescaler Bits

(0<<ADPS1)|

(1<<ADPS0);

return;

}

void TIMER1\_init()

{

OCR1A = 0x7A12;

TIMSK1 |= (1<<OCIE1A); //enable Compare A interrupt

TCCR1B |=(1<<WGM12)|// CTC Mode

(1<<CS11)| // Prescaler = 64

(1<<CS10);

}

void USARTstart(unsigned int ubrr)

{

UBRR0H = (unsigned char)(ubrr>>8);// set baud rate to 9600

UBRR0L = (unsigned char)ubrr;

UCSR0B |= (1<<RXEN0) | (1<<TXEN0); // Enable Transmit and Receive

UCSR0C |= (1<<UCSZ01) | (1<<UCSZ00); // Set Frame: 8bit, 1 Stop

return;

}

void USART\_tx\_string(char \*data)// transmit value function

{

while ((\*data != '\0'))// loop to send each character over the serial connection

{

while (!(UCSR0A & (1<<UDRE0)))// wait until transmit buffer is clear

{

}

UDR0 = \*data; // load character into transmit buffer

data++; // go to next character

}

}

ISR (TIMER1\_COMPA\_vect)// Timer 1 compare interrupt

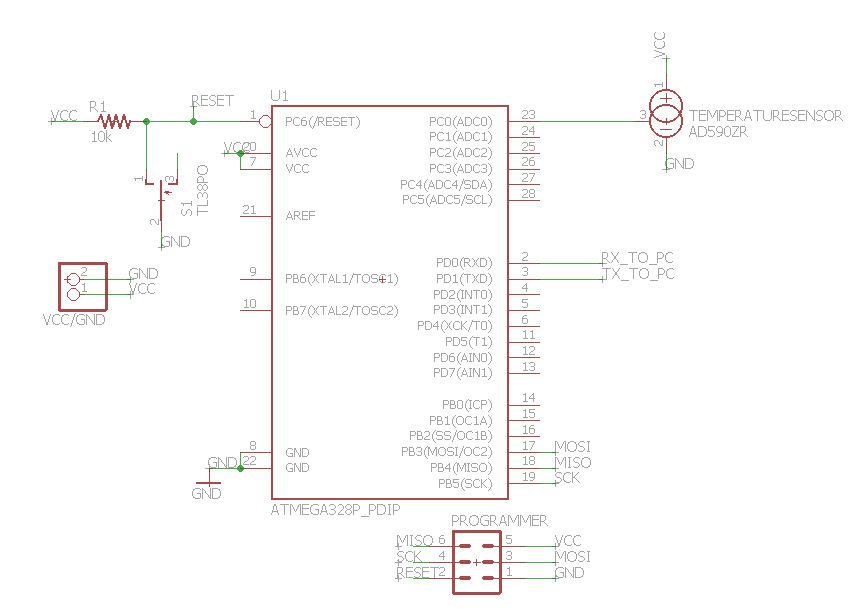
{

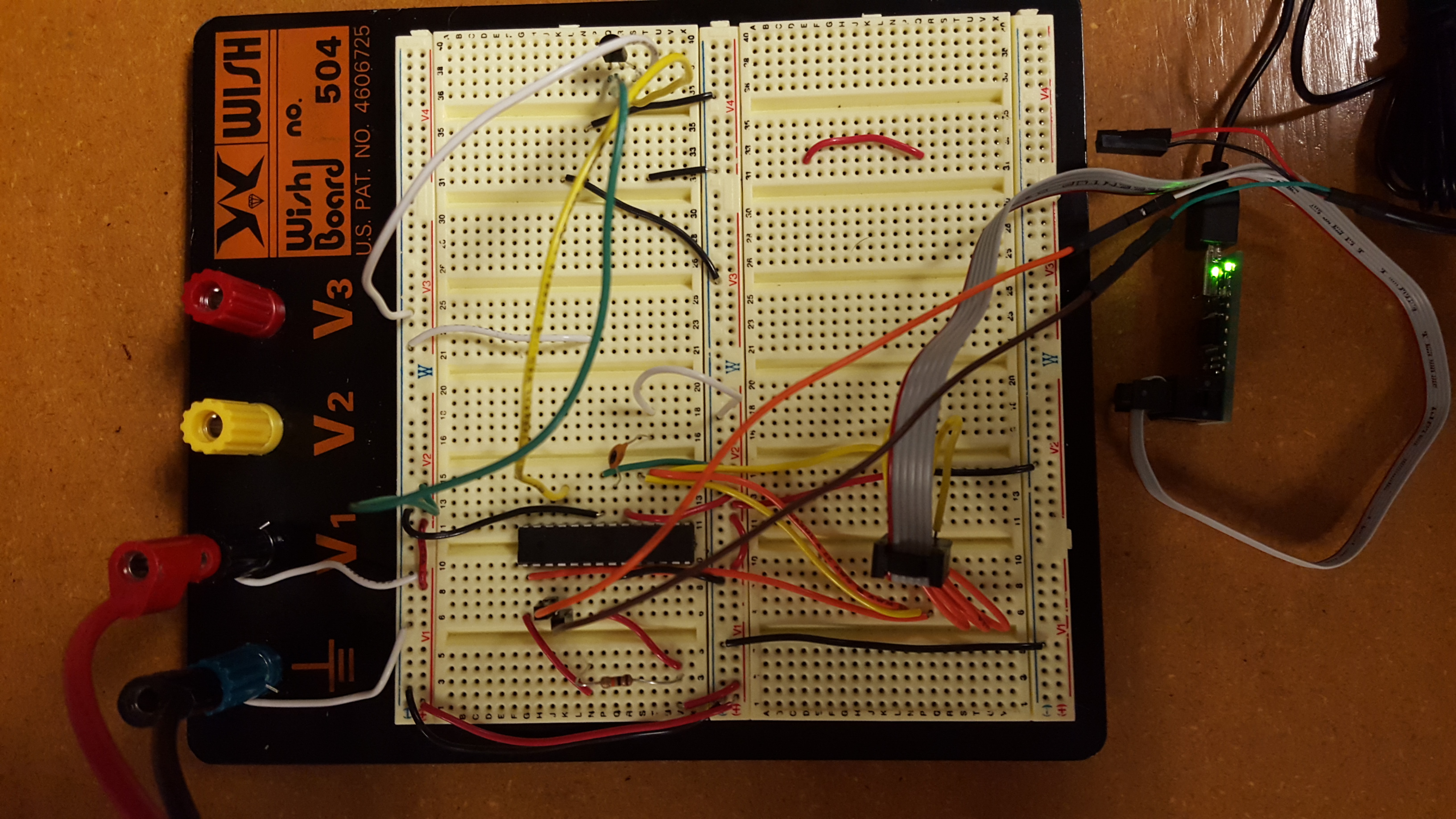
read\_adc();// read value from sensor

X--; // decrease X

}

|  |  |  |  |
| --- | --- | --- | --- |
| 6. | SCHEMATICS |  |  |

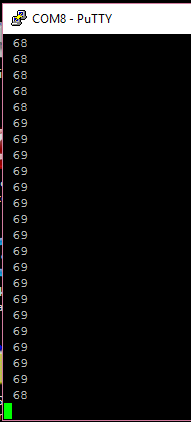




|  |  |  |  |
| --- | --- | --- | --- |
| 7. | SCREENSHOTS OF EACH TASK OUTPUT |  |  |

TASK 1:

Write a C AVR program that will monitor the LM34/35 connected to an Analog pin to display the temperature in F on the serial terminal every 1 sec. Use a timer with interrupt for the 1 sec delay.



**FLOWCHART**

START

Initialize ADC

Initialize USART

Initialize Timer 1

Enable Interrupts

X = 4

ADCvalue = 0

FALSE

X = 0

TRUE

ADCvalue = ADCvalue/4

Convert ADCvalue to string called outs

Transmit string outs using USART over serial connection

|  |  |  |  |
| --- | --- | --- | --- |
| 9. | VIDEO LINKS OF EACH DEMO |  |  |
| No video taken | | | |
| 10. | GOOGLECODE LINK OF THE DA |  |  |
| https://github.com/Wellsa15/wellsa\_unlv | | | |

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

Andrew Wells