

Design Assignment 3

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

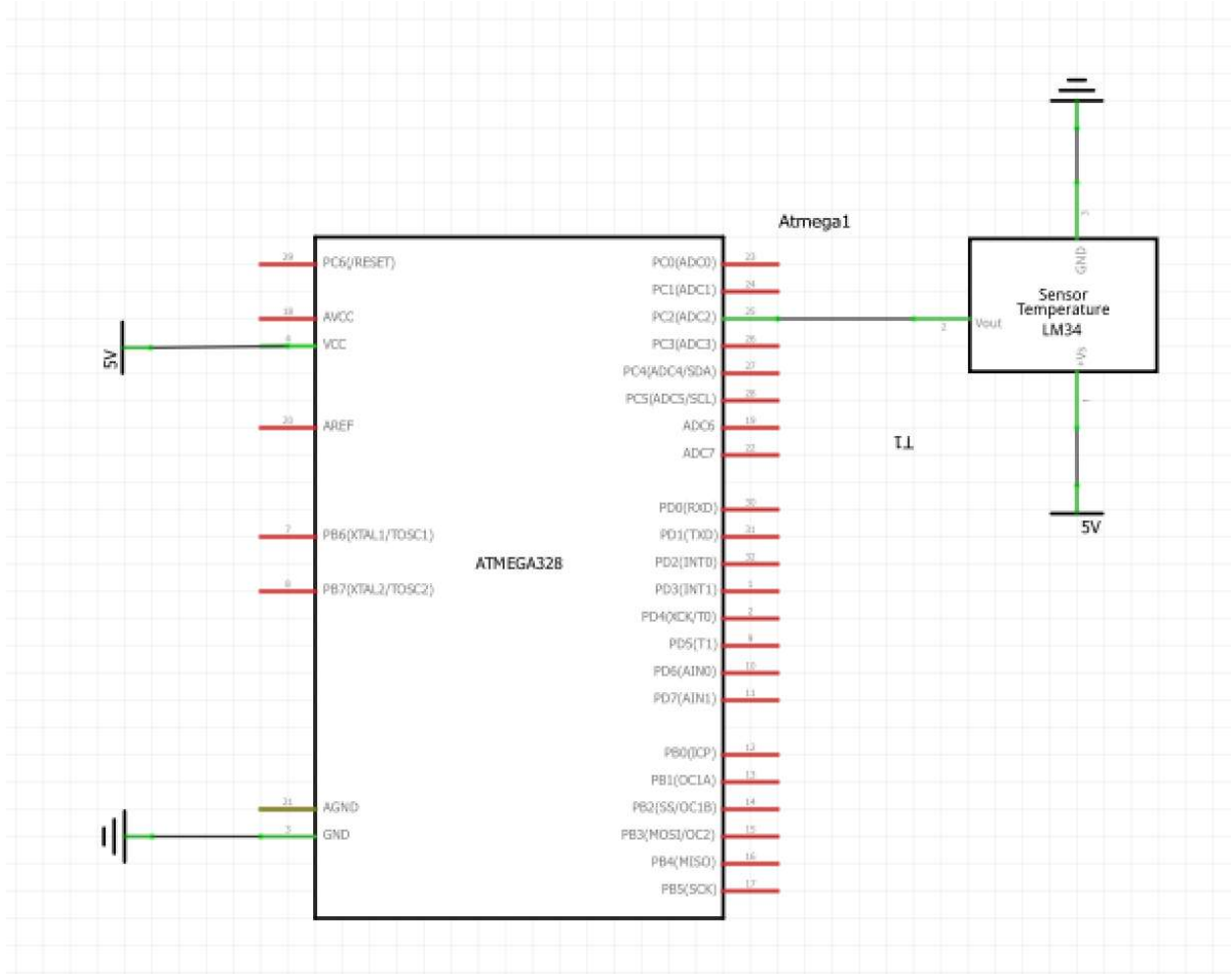
The student understands that all required components should be submitted in complete for grading of this assignment.

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

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

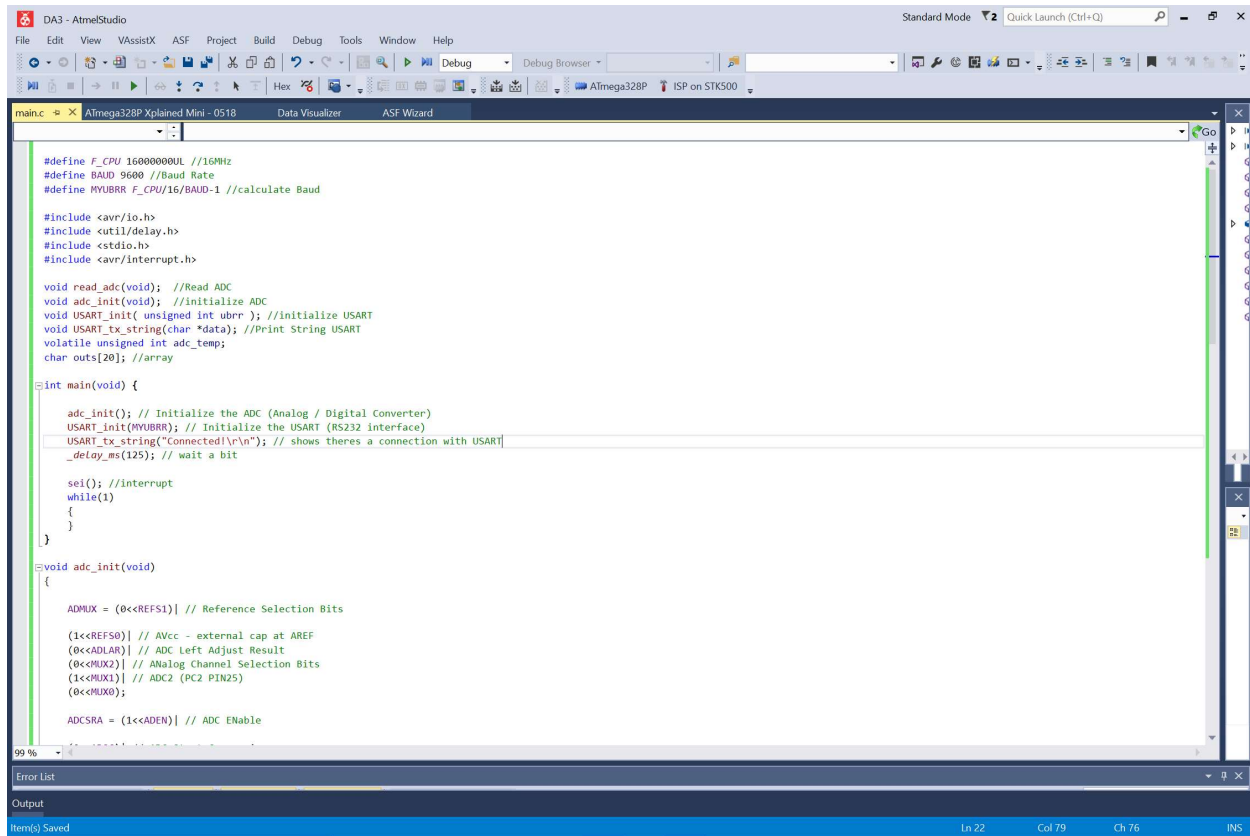
Components:

- LM34 Temperature Sensor
- ATmega328P Xplained Mini Board
- Breadboard
- 5V Power Supply



This is the schematic/block diagram showing the pins used on the ATmega328P Board. I used PC2/ADC2 as the port to read from the LM34 Temperature Sensor.

2. INITIAL/DEVELOPED CODE OF TASK 1/A



```
#define F_CPU 16000000UL //16MHz
#define BAUD 9600 //Baud Rate
#define MYUBRR F_CPU/16/BAUD-1 //calculate Baud

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

void read_adc(void); //Read ADC
void adc_init(void); //initialize ADC
void USART_init(unsigned int ubrr); //Initialize USART
void USART_tx_string(char *data); //Print String USART
volatile unsigned int adc_temp;
char outs[20]; //array

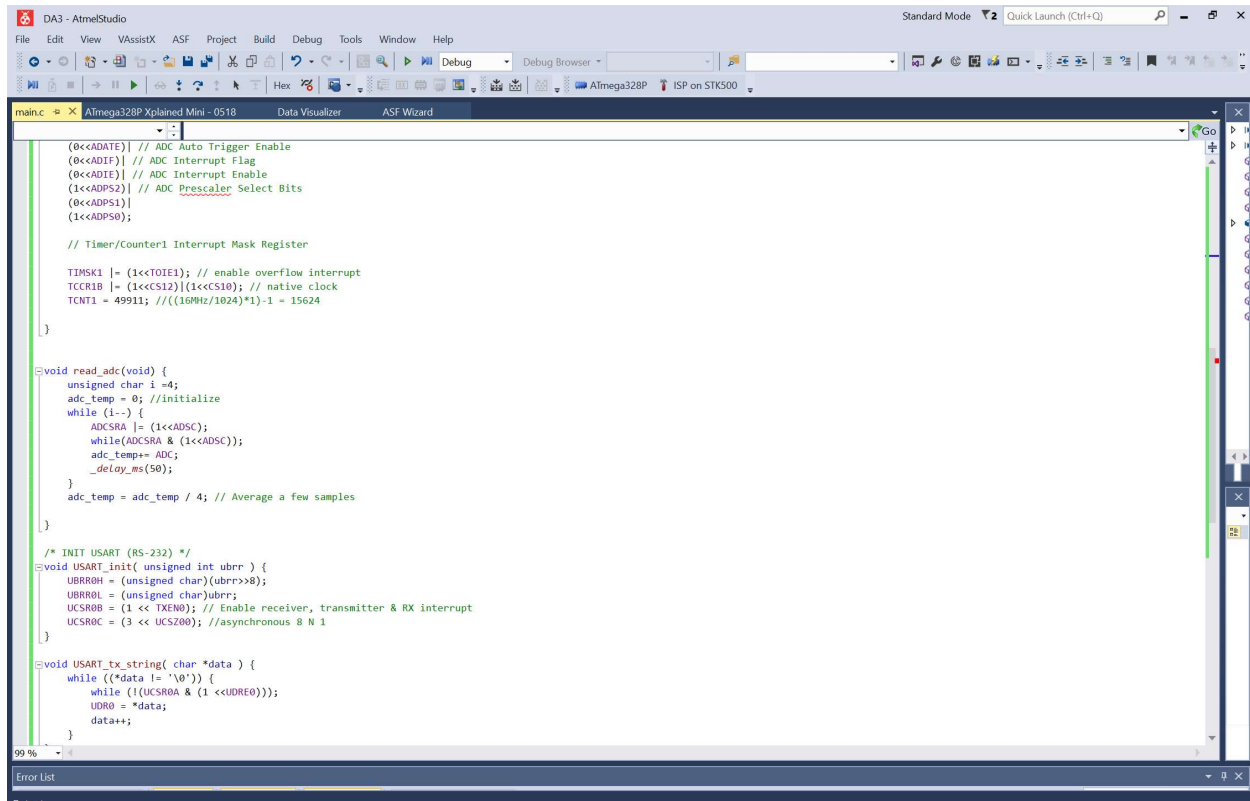
int main(void) {
    adc_init(); // Initialize the ADC (Analog / Digital Converter)
    USART_init(MYUBRR); // Initialize the USART (RS232 interface)
    USART_tx_string("Connected\r\n"); // shows theres a connection with USART
    _delay_ms(125); // wait a bit

    sei(); //interrupt
    while(1)
    {
    }
}

void adc_init(void)
{
    ADMUX = (0<<REFS1) // Reference Selection Bits
    (1<<REFS0) // AVcc - external cap at AREF
    (0<<ADLAR) // ADC Left Adjust Result
    (0<<MUX2) // Analog Channel Selection Bits
    (1<<MUX1) // ADC2 (PC2 PIN25)
    (0<<MUX0);

    ADCSRA = (1<<ADEN) // ADC Enable
}
```

Above shows my main function that calls the ADC and USART functions. Furthermore, it shows the ADC function in the above image and below image.



```
(0<<ADSC) // ADC Auto Trigger Enable
(0<<ADIF) // ADC Interrupt Flag
(0<<ADIE) // ADC Interrupt Enable
(1<<ADPS2) // ADC Prescaler Select Bits
(0<<ADPS1)
(1<<ADPS0);

// Timer/Counter1 Interrupt Mask Register

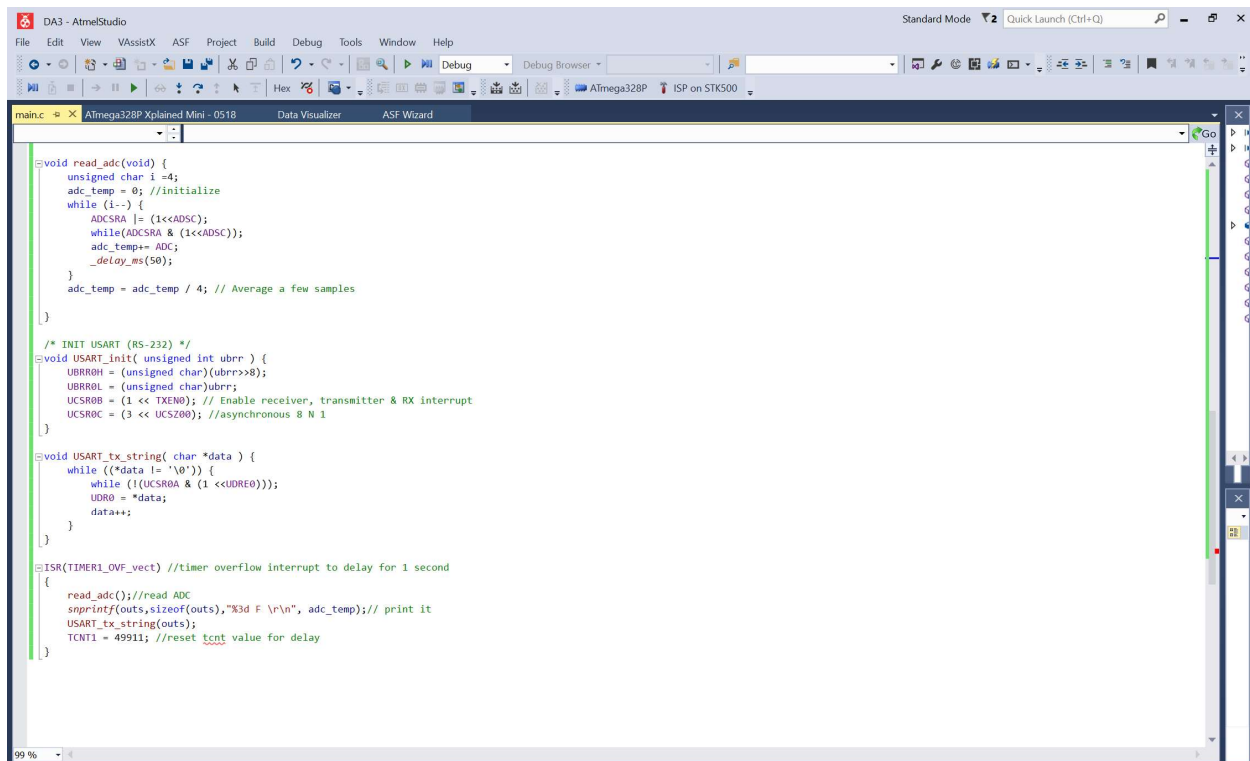
TIMSK1 |= (1<<TOIE1); // enable overflow interrupt
TCCR1B |= (1<<CS12)|(1<<CS10); // native clock
TCNT1 = 49911; //((16MHz/1024)*1)-1 = 15624
}

void read_adc(void) {
    unsigned char i = 4;
    adc_temp = 0; //initialize
    while (i--) {
        ADCSRA |= (1<<ADSC);
        while(ADCSRA & (1<<ADSC));
        adc_temp += ADC;
        _delay_ms(50);
    }
    adc_temp = adc_temp / 4; // Average a few samples
}

/* INIT USART (RS-232) */
void USART_init( unsigned int ubrr ) {
    UBRR0H = (unsigned char)(ubrr>>8);
    UBRR0L = (unsigned char)ubrr;
    UCSRB0 = (1 << TXEN0); // Enable receiver, transmitter & RX interrupt
    UCSRC0 = (3 << UCSZ00); //asynchronous 8 N 1
}

void USART_tx_string( char *data ) {
    while ((*data != '\0')) {
        while (!(UCSR0A & (1 << UDRE0)));
        UDR0 = *data;
        data++;
    }
}
```

This code above shows how USART is being implemented and initialized. The code below is showing the Interrupt Service Routine for the delay of 1 second.



```
void read_adc(void) {
    unsigned char i = 4;
    adc_temp = 0; //initialize
    while (i--) {
        ADCSRA |= (1<<ADSC);
        while(ADCSRA & (1<<ADSC));
        adc_temp += ADC;
        _delay_ms(50);
    }
    adc_temp = adc_temp / 4; // Average a few samples
}

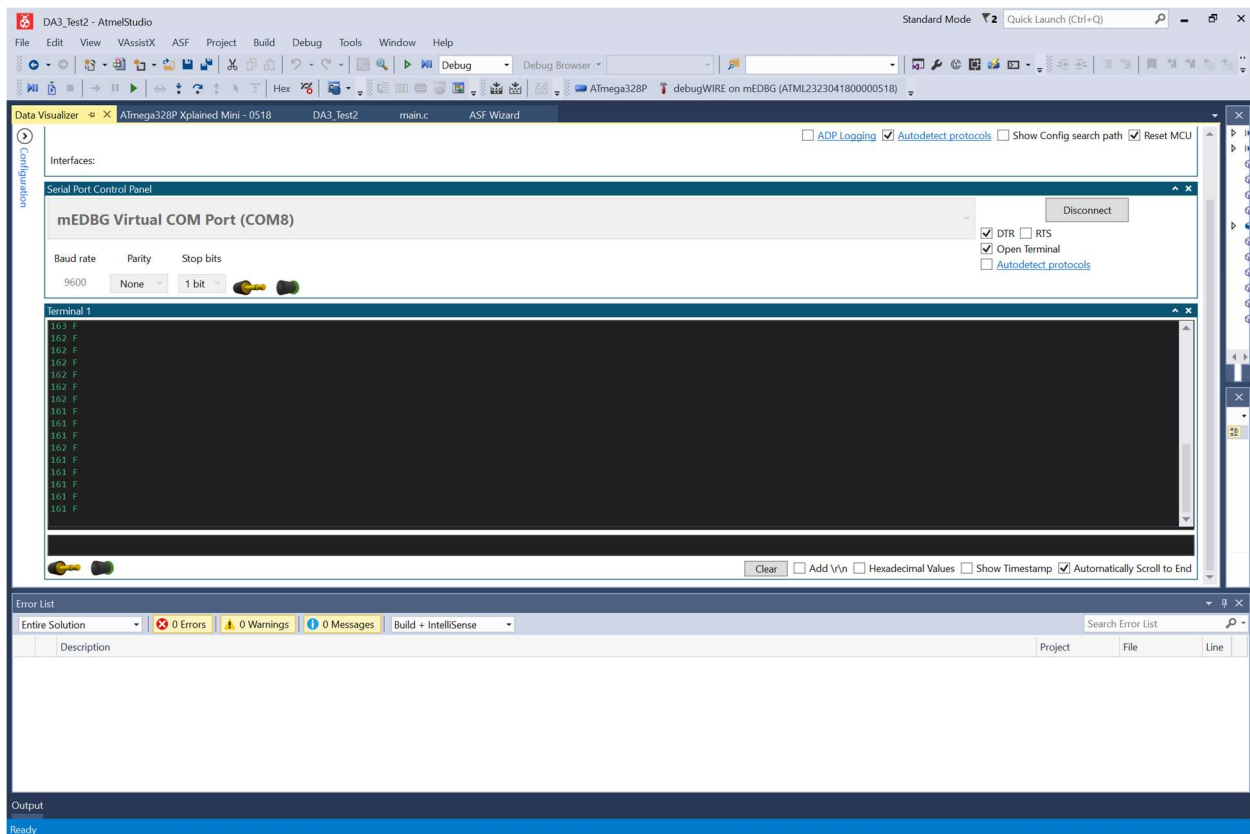
/* INIT USART (RS-232) */
void USART_init( unsigned int ubrr ) {
    UBRR0H = (unsigned char)(ubrr>>8);
    UBRR0L = (unsigned char)ubrr;
    UCSRB0 = (1 << TXEN0); // Enable receiver, transmitter & RX interrupt
    UCSRC0 = (3 << UCSZ00); //asynchronous 8 N 1
}

void USART_tx_string( char *data ) {
    while ((*data != '\0')) {
        while (!(UCSR0A & (1 << UDRE0)));
        UDR0 = *data;
        data++;
    }
}

ISR(TIMER1_OVF_vect) //timer overflow interrupt to delay for 1 second
{
    read_adc(); //read ADC
    snprintf(outs, sizeof(outs), "%3d F \r\n", adc_temp); // print it
    USART_tx_string(outs);
    TCNT1 = 49911; //reset tcnt value for delay
}
```

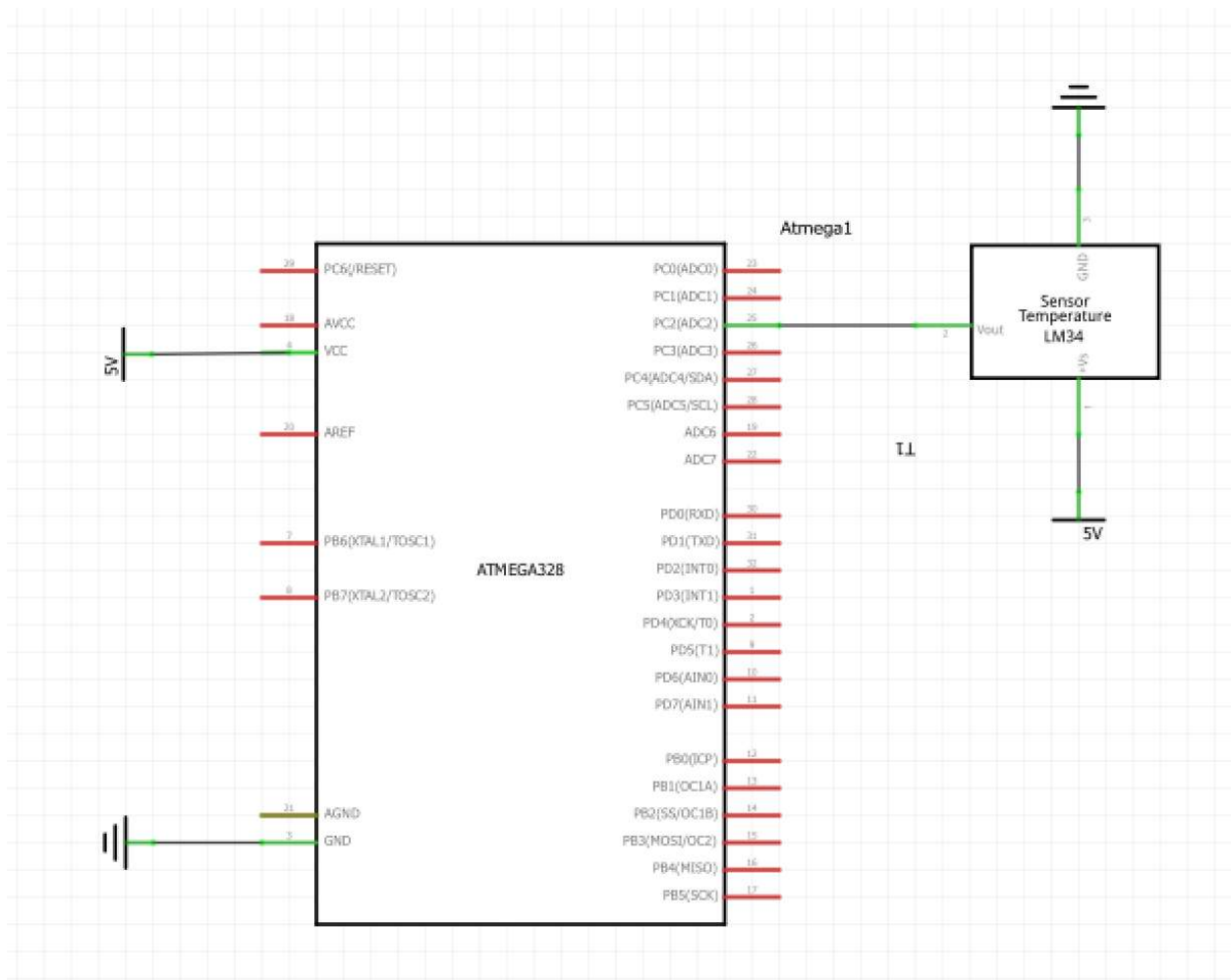
Above is a C Code written to implement USART+ADC by reading the LM34 Temperature Sensor and outputting the temperature to a terminal. The above code is based off the provided resources given on the Lecture Slides. Furthermore, the code is responsible for updating the temperature every second. By calculating this, we update the TCNT1 value to 49911. Once this value reaches its max it will call the Interrupt Service Routine and reset the counter value.

3. OUTPUT FOR TASK 2



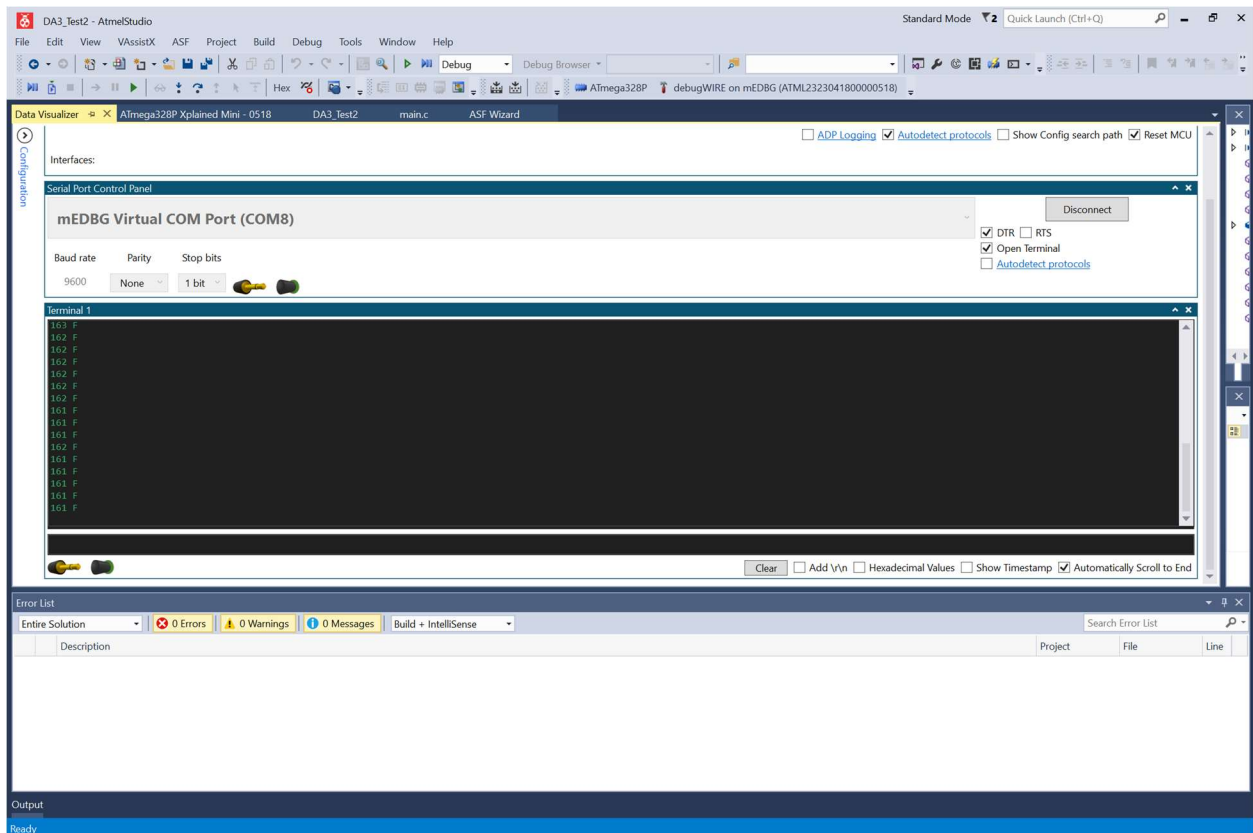
Above is the output for the C Code provided in Task 1. The output shown above is what happens when I grab the LM34 Chip to make it warmer and then let it go to cool down. In the image above we can see that the temperature begins to decrease. The output was displayed on the Data Visualizer using the Micro-USB Port on the ATmega328P Xplained Mini Board using the Virtual COM Port.

4. SCHEMATICS

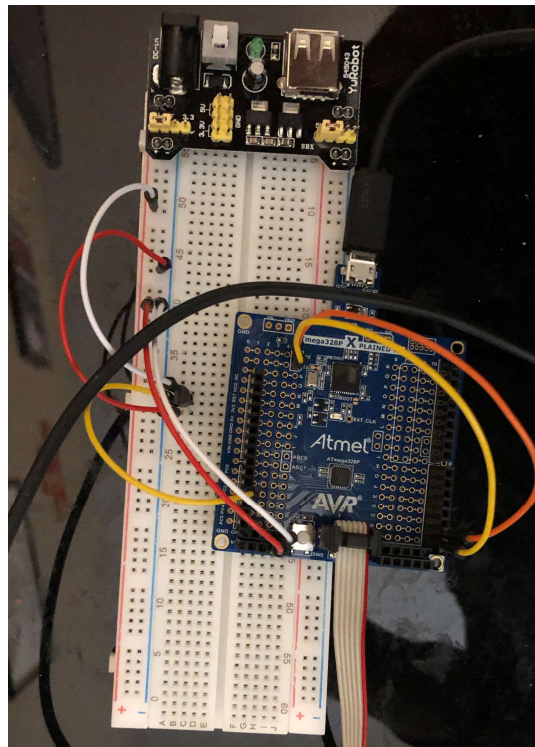


This schematic is what was used to output the LM34 Chip to the Data Visualizer. The board I used for my project was the ATmega328P Xplained Mini; therefore, I did not use the FTDI chip or 16MHz XTAL. I also connected PD0 (RXD) -> TX and PD1 (TXD) -> RX on the Xplained Mini Board.

5. SCREENSHOTS OF EACH TASK OUTPUT (ATEL STUDIO OUTPUT)



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)



7. VIDEO LINKS OF EACH DEMO

<https://youtu.be/2KjxljD6lil>

8. GITHUB LINK OF THIS DA

<https://github.com/bkiaer/DA3>

Student Academic Misconduct Policy

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

"This assignment submission is my own, original work".

Brian Medrano Kiaer