Nathan Hanuscin

CPE301 – SPRING 2018

Design Assignment 3

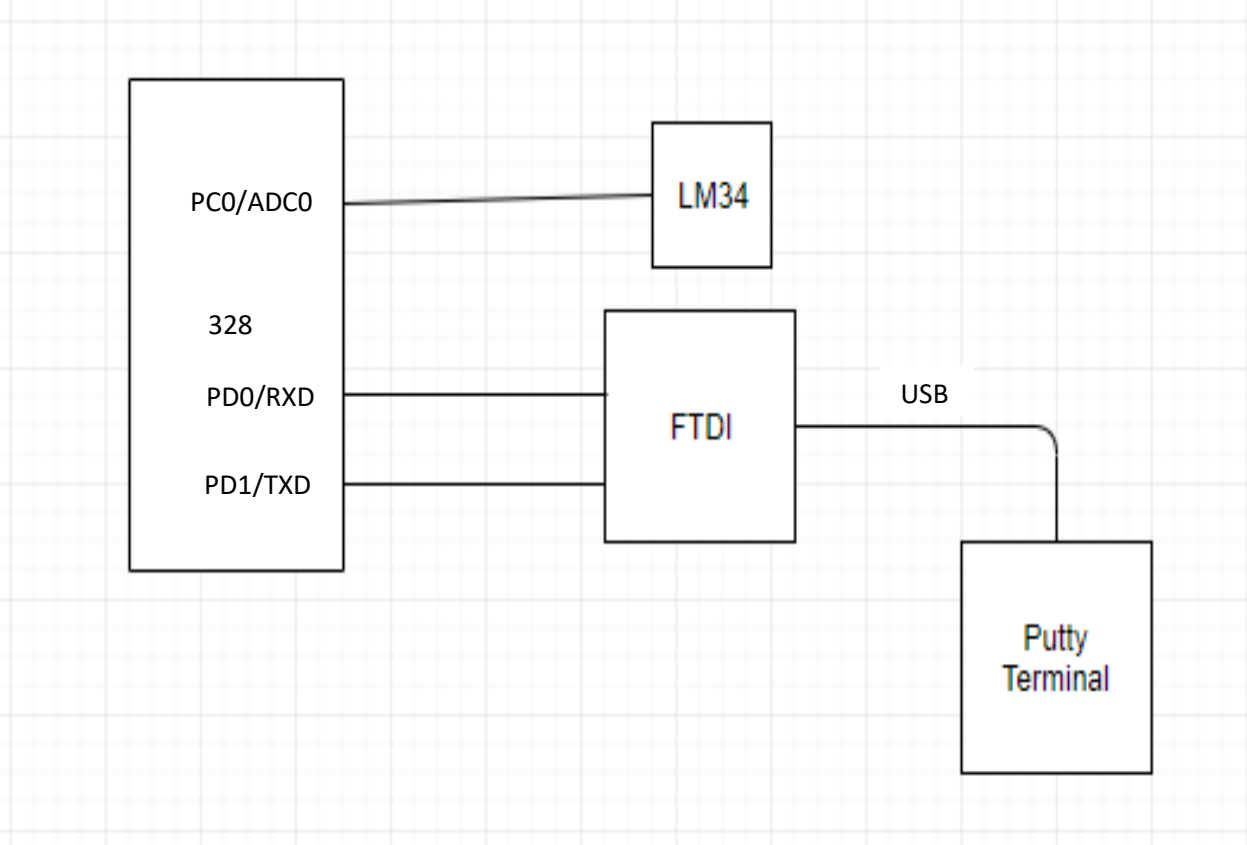
DO NOT REMOVE THIS PAGE DURING SUBMISSION:

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

FTDI R232
LM34 Temperature sensor



2. INITIAL/DEVELOPED CODE OF TASK 1/A

```
#define F_CPU 1000000UL
#define UBRR_1200 51
#define UBRR_2400 25 // for 1Mhz
// #define UBRR_2400 207 // for 8Mhz with .2% error
// #define UBRR_9600 51 // for 8Mhz with .2% error
// #define UBRR_19200 25 // for 8Mhz with .2% error
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>
void read_adc(void); // Function Declarations
void adc_init(void);
void USART_init( unsigned int ubrr );
void USART_tx_string( char *data );
volatile unsigned int adc_temp;
char outs[20];

int main(void) {
    adc_init(); // Initialize the ADC (Analog / Digital Converter)
    USART_init(UBRR_2400); // Initialize the USART (RS232 interface)
    USART_tx_string("Connected!\r\n"); // we're alive!
    _delay_ms(125); // wait a bit

    while(1)
    {

        read_adc();
        snprintf(outs,sizeof(outs),"%3d\r\n", adc_temp);// print it
        USART_tx_string(outs);
        _delay_ms(125); // wait a bit

    }
}

void adc_init(void)
{
    /** Setup and enable ADC **/
    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

    (0<<ADSC)| // ADC Start Conversion
    (0<<ADATE)| // 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<<CS11)|
    (1<<CS10); // native clock

}
```

```
/* READ ADC PINS */
void read_adc(void) {
    unsigned char i =4;
    adc_temp = 0;
    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;
    UCSR0B = (1 << TXEN0); // Enable receiver, transmitter & RX interrupt
    UCSR0C = (3 << UCSZ00); //asynchronous 8 N 1
}
/* SEND A STRING TO THE RS-232 */
void USART_tx_string( char *data ) {
    while ((*data != '\0')) {
        while (!(UCSR0A & (1 <<UDRE0)));
        UDR0 = *data;
        data++;
    }
}
```

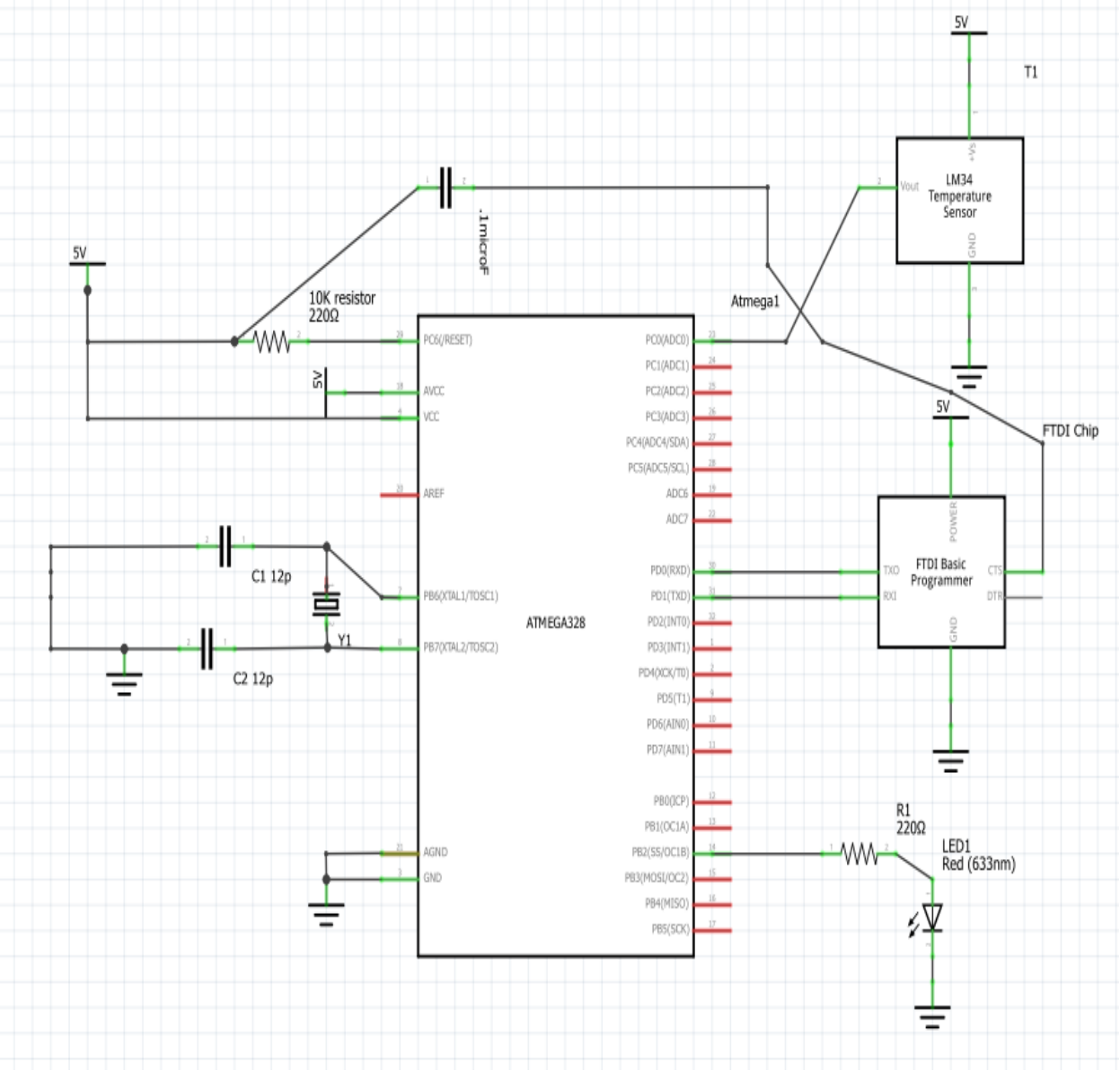
3. COMPLETE/MODIFIED CODE

```
#define F_CPU 8000000UL
#define UBRR_9600 51 // for 8Mhz with .2% error

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

// Function Declarations
void read_adc(void);
void adc_init(void);
void USART_init( unsigned int ubrr );
void USART_tx_string( char *data );
volatile unsigned int adc_temp;
char outs[20];


int main(void)
{
    adc_init();                         //Initialize the ADC (Analog / Digital Converter)
    USART_init(UBRR_9600);              //Initialize the USART (RS232 interface)
    USART_tx_string("Connected!\r\n");  //Display connected
    _delay_ms(125);                     //wait a bit
    TIMSK1 = (1<<TOIE1);                //enable timer overflow interrupt
    TCNT1 = 57723;                      //set counter value
    TCCR1A = 0;                         //normal mode
    TCCR1B = (1<<CS12) | (1<<CS10);     //pre-scaler of 1024
    sei();                              //enable interrupts
    while(1)
    {
        // wait for interrupt
    }
}
```

```
ISR (TIMER1_OVF_vect)
{
    TCCR1B = 0;                                         //turn timer off
    read_adc();                                         //read the adc values
    snprintf(outs,sizeof(outs),"%3d\r\n", adc_temp);    //print it
    USART_tx_string(outs);
    _delay_ms(125);                                     // wait a bit
    TCNT1 = 57723;                                      //reset timer value
    TCCR1B = (1<<CS12) | (1<<CS10);                     //restart the clock

}

void adc_init(void)
{
    /** Setup and enable ADC **/
    ADMUX = 0;              //select ADC0 Pin as input
    ADMUX = (0<<REFS1)|     //Reference Selection Bits
    (1<<REFS0)|             //AVcc - external cap at AREF
    (1<<ADLAR);             //ADC left Adjust Result

    ADCSRA = (1<<ADEN)|     //ADC ENable
    (1<<ADSC)|              //ADC Start Conversion
    (1<<ADATE)|             //ADC Auto Trigger Enable
    (0<<ADIF)|              //ADC Interrupt Flag
    (0<<ADIE)|              //ADC Interrupt Enable
    (1<<ADPS2)|             //ADC Pre-scaler of 64
    (1<<ADPS1)|
    (0<<ADPS0);

}
```

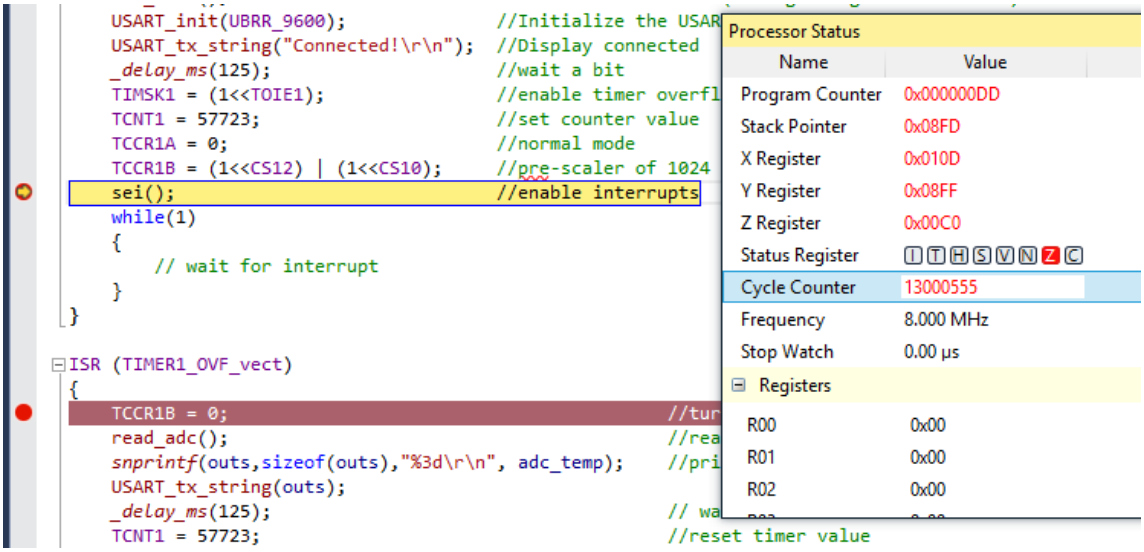
```c
/* READ ADC PINS*/
void read_adc(void)
{
    unsigned char i = 4;                    //set for 4 ADC reads
    adc_temp = 0;                           //initialize temp to 0
    while (i--)
    {
        ADCSRA |= (1<<ADSC);                //start the conversion
        while((ADCSRA & (1<<ADIF)) == 0);   //wait for conversion to finish
        adc_temp += ADCH*2;                 //get temp value
        _delay_ms(50);                      //wait a bit
    }
    adc_temp = adc_temp / 4;                // Average a few samples

}

/* INIT USART (RS-232) */
void USART_init( unsigned int ubrr )
{
    UBRR0H = (unsigned char)(ubrr>>8);          //set baud rate
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << TXEN0) | (1 <<RXEN0);        // Enable receiver, transmitter
    UCSR0C = (1 << UCSZ00) | (1 << UCSZ01);     //asynchronous 8-bit data 1 stop bit
}

/* SEND A STRING TO THE RS-232*/
void USART_tx_string( char *data )
{
    while ((*data != '\0'))
    {
        while (!(UCSR0A & (1 <<UDRE0)));    //wait for the transmit buffer to empty
        UDR0 = *data;                       //put the data into the empty buffer, which sends the data
        _delay_ms(125);                     // wait a bit
        data++;
    }
}
```

4. SCHEMATICS

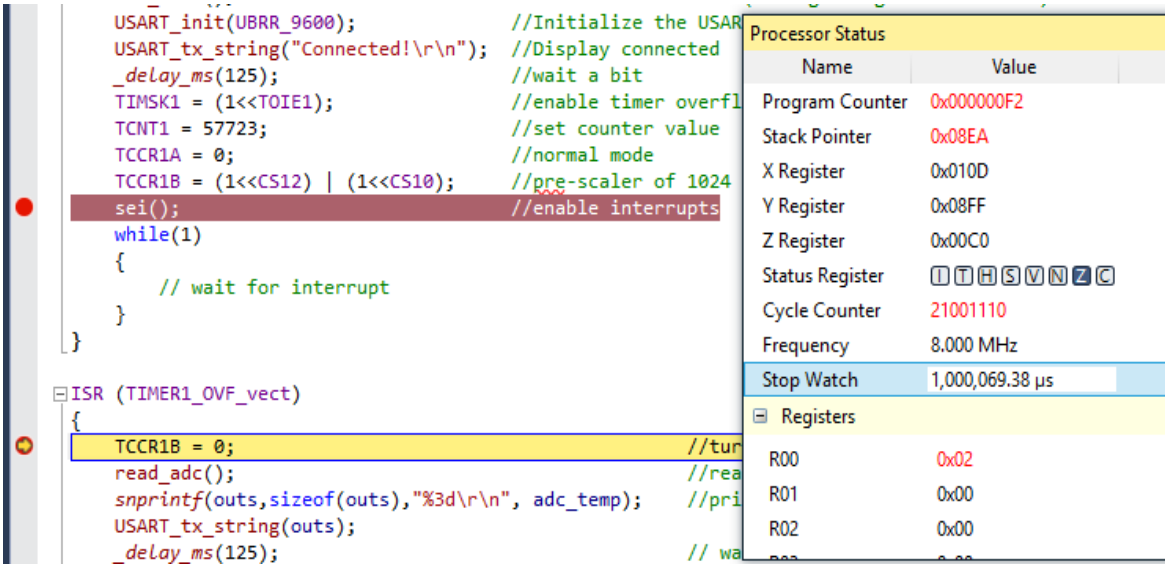


5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

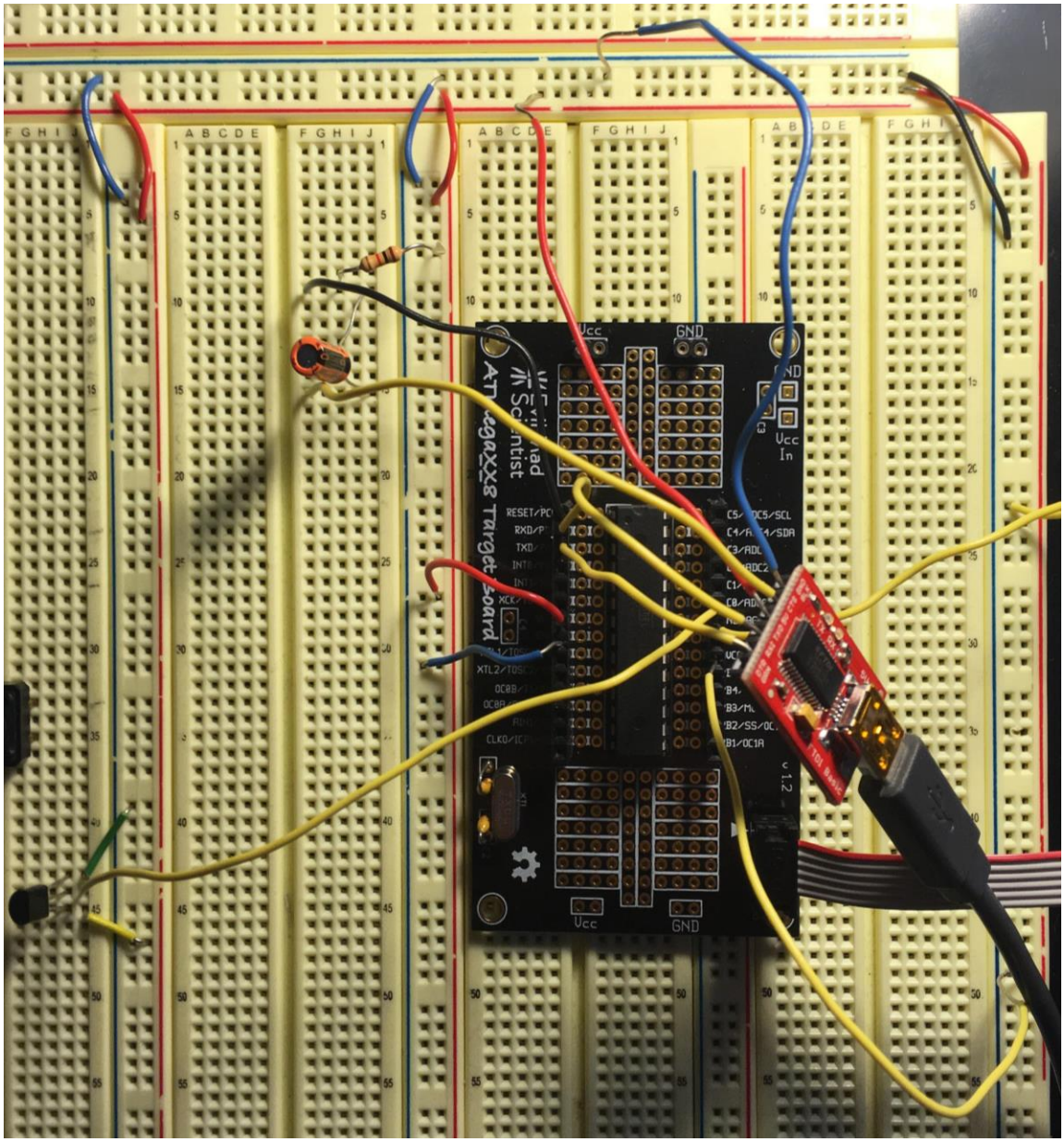
Before Interrupt



After interrupt of 1 second



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)



7. VIDEO LINKS OF EACH DEMO

https://www.youtube.com/watch?v=hYyGTyl2VTc

8. GITHUB LINK OF THIS DA

https://github.com/nhanuscin/submit/tree/master/DA3

Student Academic Misconduct Policy

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

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

Nathan Hanuscin