CPE301 – SPRING 2019

Design Assignment 3A

Student Name: Moriah Wingrove

Student #: 5000156247

Student Email: Wingrove@unlv.nevada.edu

Primary Github address: <https://github.com/windew/Tiny_Dragons.git>

Directory:

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

Atmega 328pb

FTDI 232 chip

HiLetgo Multi-Function Shield ProtoShield Multi-functional Expansion Board Sensor Shield Module witn Four Digital Display for Arduino

Lm35

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 3B**

**C code**

/\*

/\*

\* Design\_Assignment\_3B.c

\*

\* Created: 10/16/2019 12:05:36 PM

\* Author : Moriah Wingrove

\*/

//TASK ONE: Write a C AVR program that will monitor the LM34/35 connected to an Analog pin (PC5) to display

// the temperature in F on the serial terminal every 1 sec. Use a timer with interrupt delay for the

// 1 sec delay. Use a FTDI chip for serial to USB conversion.

//TASK TWO: Use the ATMEL Studio Data Visualizer or any charting program to display the values

#define *F\_CPU* 16000000UL //16 Mhz.

#define BAUDRATE 9600 //Set Baud rate 9600.

#define BAUD\_PRESCALLER (((*F\_CPU* / (BAUDRATE \* 16UL))) - 1)

#include <avr/io.h>

#include <avr/interrupt.h>

#include <stdio.h>

#include <stdlib.h>

#include <util/delay.h>

#include <math.h>

void read\_adc(void); // Function Declarations

void adc\_init(void);

void USART\_init(void);

unsigned char USART\_receive(void);

void USART\_transmit(unsigned char data); //function to send through USART

void USART\_tx\_string( char \*data );

volatile unsigned int adc\_temp;

char outs[20];

*uint8\_t* OVF\_COUNT = 0; //initialize the overflow count for interrupt

*uint8\_t* OVF\_LIMIT = 250; //set the limit the count can reach to set 1 sec delay

int main(void) {

adc\_init(); // Initialize the ADC (Analog / Digital Converter)

USART\_init(); // 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*(1000); // 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

(1<<MUX2)| // ANalog Channel Selection Bits

(0<<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

TCNT1 = 49911;

}

/\* 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 /3; // Average a few samples

//int tempf= (adc\_temp\*5\*100);

*snprintf*(outs,sizeof(outs),"%3d\r\n", adc\_temp);// print it

USART\_tx\_string(outs);

}

void USART\_init(void)

{

UBRR0H = (*uint8\_t*)(BAUD\_PRESCALLER >> 8);

UBRR0L = (*uint8\_t*)(BAUD\_PRESCALLER);

UCSR0B = (1 << RXEN0) | (1 << TXEN0);

UCSR0C = (3 << UCSZ00);

}

/\* SEND A STRING TO THE RS-232 \*/

unsigned char USART\_receive(void)

{

while(!(UCSR0A & (1 << RXC0)));

return UDR0;

}

void USART\_transmit(unsigned char data)

{

while(!(UCSR0A & (1 << UDRE0)));

UDR0 = data;

}

void USART\_tx\_string(char\* data)

{

while(\*data != 0x00)

{

USART\_transmit(\*data);

data++;

}

}

ISR(TIMER1\_OVF\_vect)

{

OVF\_COUNT++; //increment the overflow counter

if (OVF\_COUNT == OVF\_LIMIT) //check to see if the limit was reached

{

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 = (a/1024.0) \* 5000/10;

a = (a\*2)+32; //equation to convert celsius to farenheit. can't use used 2 instead of 9/5

a = a % 100;

USART\_transmit((a/10)+'0');

a = a % 10;

USART\_transmit((a)+'0');

USART\_transmit('\n');

OVF\_COUNT = 0; //reset overflow counter

}

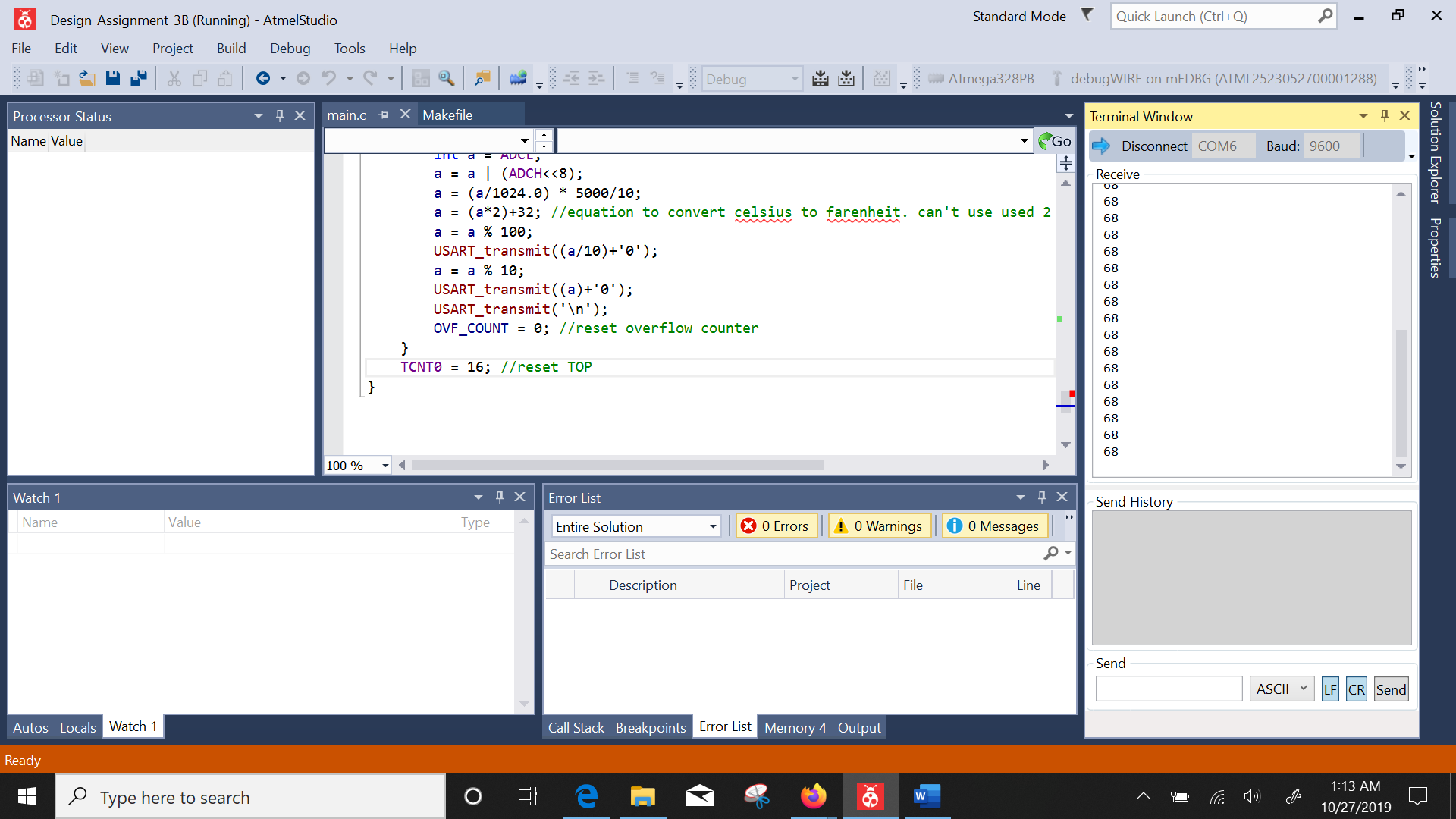
TCNT0 = 16; //reset TOP

}

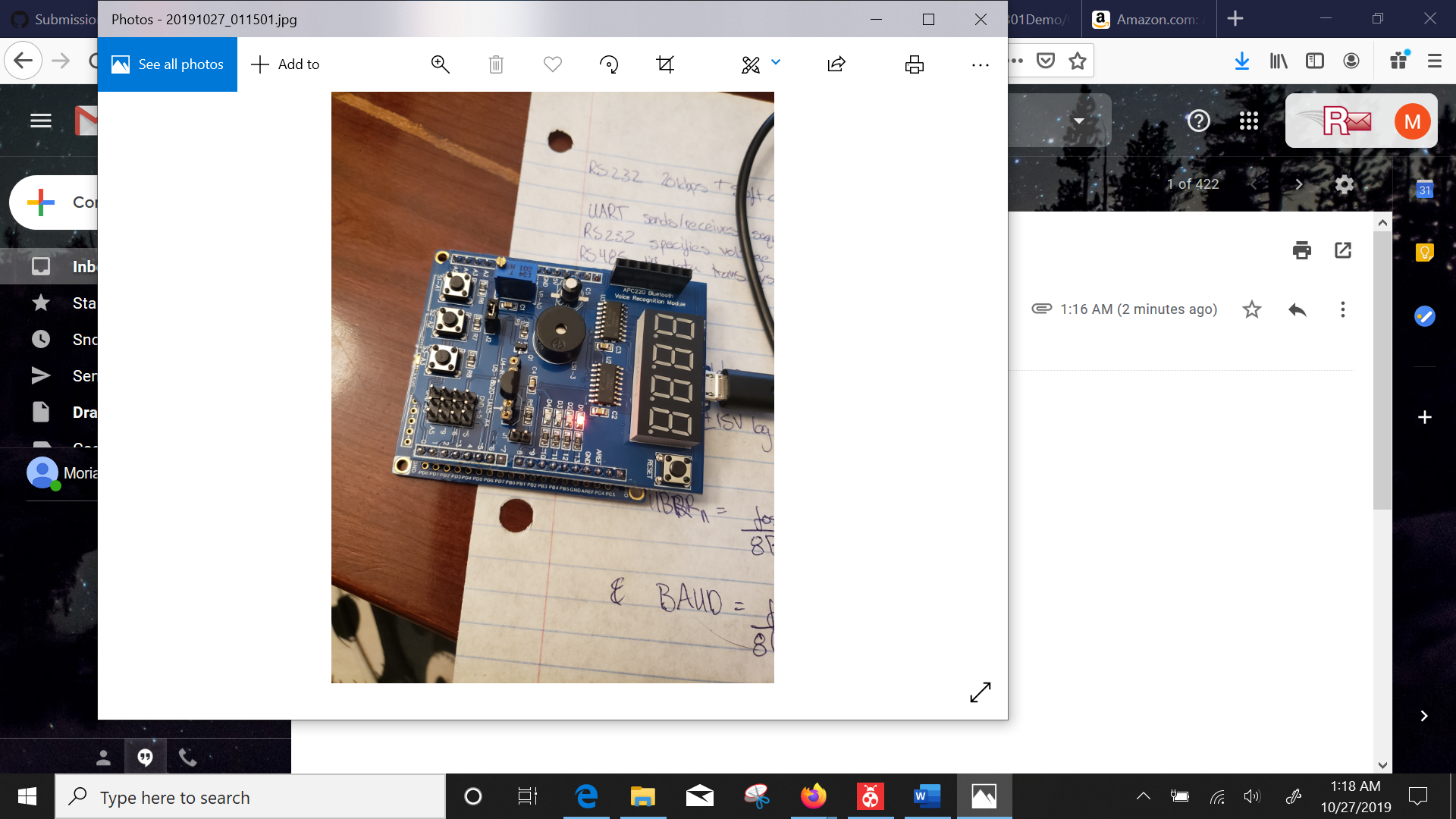
1. **SCHEMATICS**

Use fritzing.org

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**



1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**



1. **VIDEO LINKS OF EACH DEMO**
2. **GITHUB LINK OF THIS DA**

<https://github.com/windew/Tiny_Dragons>

**Student Academic Misconduct Policy**

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

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

NAME OF THE STUDENT