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

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 2/C TASK1 2A.1**

**C code**

/\*

\* Design\_Assignment\_3A.c

\*

\* Created: 10/13/2019 4:11:17 AM

\* Author : Moriah Wingrove

\*/

//Write a C AVR program that will display a string, random integer and floating point values on the serial terminal

// every 1 sec. Use a timer with interrupt for the 1 sec delay. Use a FTDI chip for serial to USB conversion

#define BAUD 9600 //Defines baud rate

#define *F\_CPU* 16000000UL //Defines clock of 16MHz

#include <avr/io.h>

#include <stdio.h>

#include <util/delay.h>

#include <avr/interrupt.h>

void USART\_send(char data); // Used to send integer to terminal

void USART\_putstring(char\* StringPtr); // Used to take in every character in the string and sends it to the terminal

void USART\_init(void); // Initializes the analog to digital functions, as well as OVF interrupt

char stringtype[] = "String: "; // Declaring the string value on screen

char inttype[] = "Integer: "; // Declaring the integer value on screen

char floattype[] = "Floating Point: "; // Declaring the floating point value on screen

char Space[] = "\n"; // Used to create the next line

char String[] = "String for assignment. Working."; //String[] is the variable to output into terminal

char outs[20]; // Allocating memory space to contain the float value

volatile float adc\_temp = 95; // Sets the float value

int main(void)

{

USART\_init(); // Initializes the analog to digital functions as well as OVF interrupt

while(1)

{

// main loop

}

}

ISR (TIMER1\_OVF\_vect)

{

USART\_putstring(Space); // creates next line

USART\_putstring(stringtype); // LABEL PRINT "String: "

USART\_putstring(String); // prints string "String for assignment. Working."

USART\_putstring(Space); // creates next line

USART\_putstring(inttype); // LABEL PRINT "Integer: "

USART\_send('3'); // prints value 3

USART\_putstring(Space); // creates next line

USART\_putstring(floattype); // LABEL PRINT "floating point: "

*snprintf*(outs, sizeof(outs), "%f\r\n", adc\_temp); // the floating point characters are stored in outs

USART\_putstring(outs); // transmits outs to UART

USART\_putstring(Space); // creates next line

TCNT1 = 49911; // Reset timer

}

void USART\_init( void )

{

UBRR0H = 0;

UBRR0L = *F\_CPU*/16/BAUD - 1; // Used for the BAUD prescaler

UCSR0C = \_BV(UCSZ01) | \_BV(UCSZ00); /\* 8-bit data \*/

UCSR0B = \_BV(RXEN0) | \_BV(TXEN0); /\* Enable RX and TX \*/

TCCR1B |= 5; //(1 << CS12) | (1 << CS10); // Sets prescaler to 1024

TIMSK1 = (1 << TOIE1); // Enables overflow flag

TCNT1 = 49911; // 1 second delay = (0xFFFF) - TCNT = 65535 - 15624 = 49911

sei();

}

void USART\_send(char data)

{

while (!(UCSR0A & (1 << UDRE0))); // Until UDRE0 goes high, it will keep looping

UDR0 = data; // UDR0 register grabs the value given from the parameter

}

void USART\_putstring(char \*StringPtr)

{

while ((\*StringPtr != '\0')){ // Until it reaches the end of the line, it will keep looping

while (!(UCSR0A & (1 << UDRE0))); // Until UDRE0 goes high, it will keep looping

UDR0 = \*StringPtr; // UDR0 register grabs the value given from the parameter

StringPtr++; // but it does it by every character as shown here

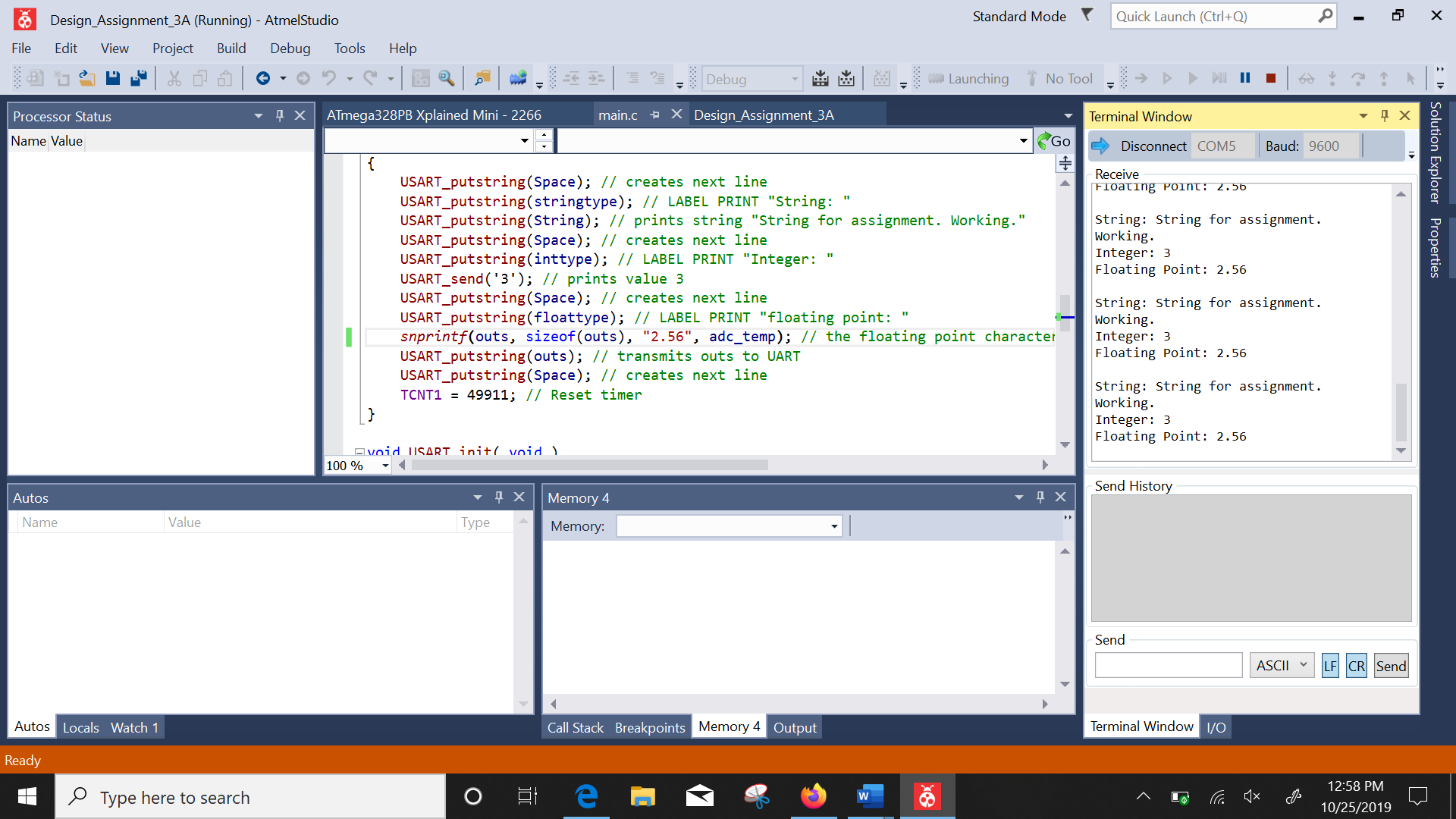
}

}

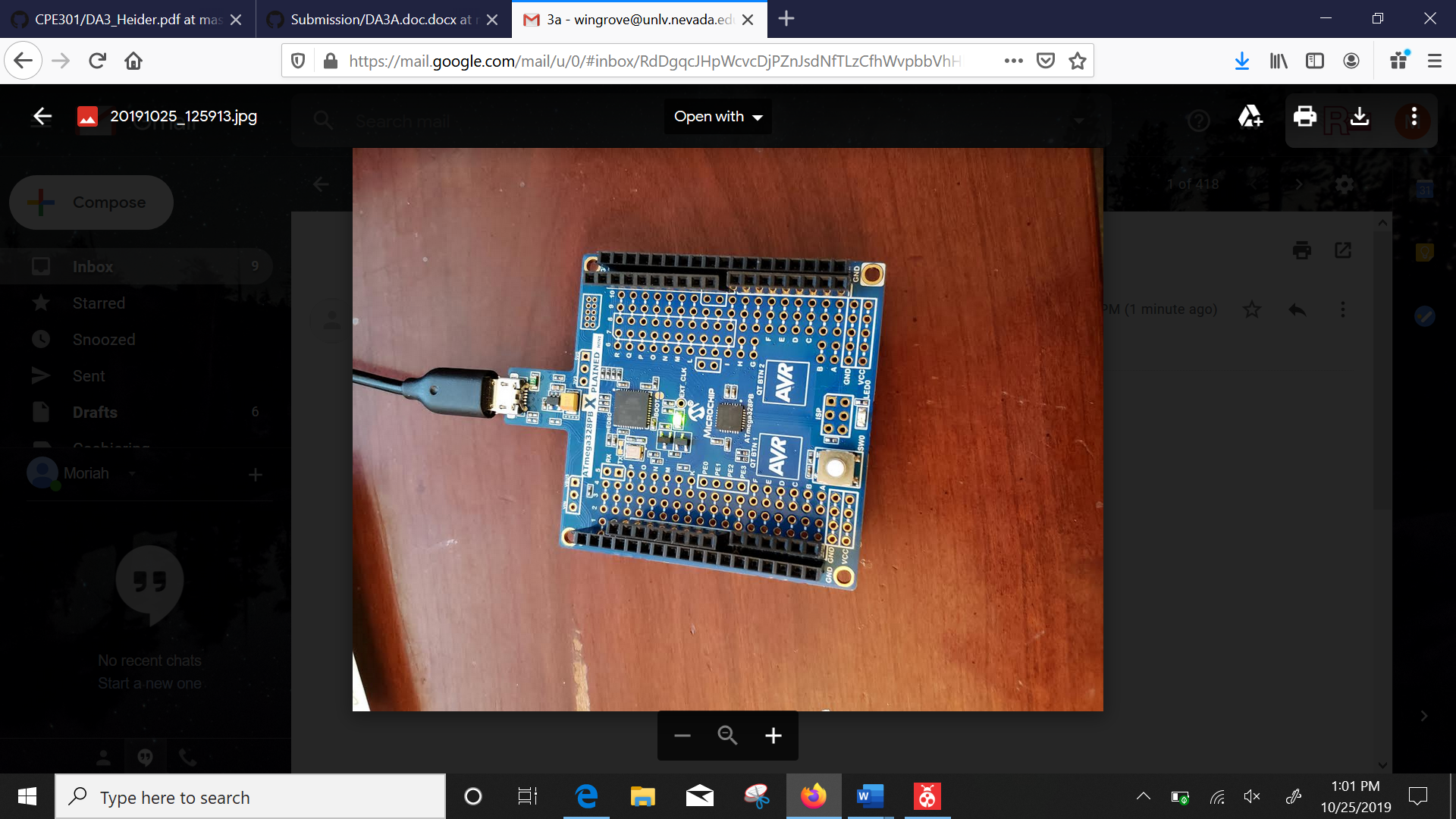
1. **SCHEMATICS**

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