CPE301 – FALL 2019

Midterm 1

Student Name: Moriah Wingrove

Student #: 5000156247

Student Email: Wingrove@unlv.nevada.edu

Primary Github address: https://github.com/windew/Tiny\_Dragons/

Directory: midterm

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**

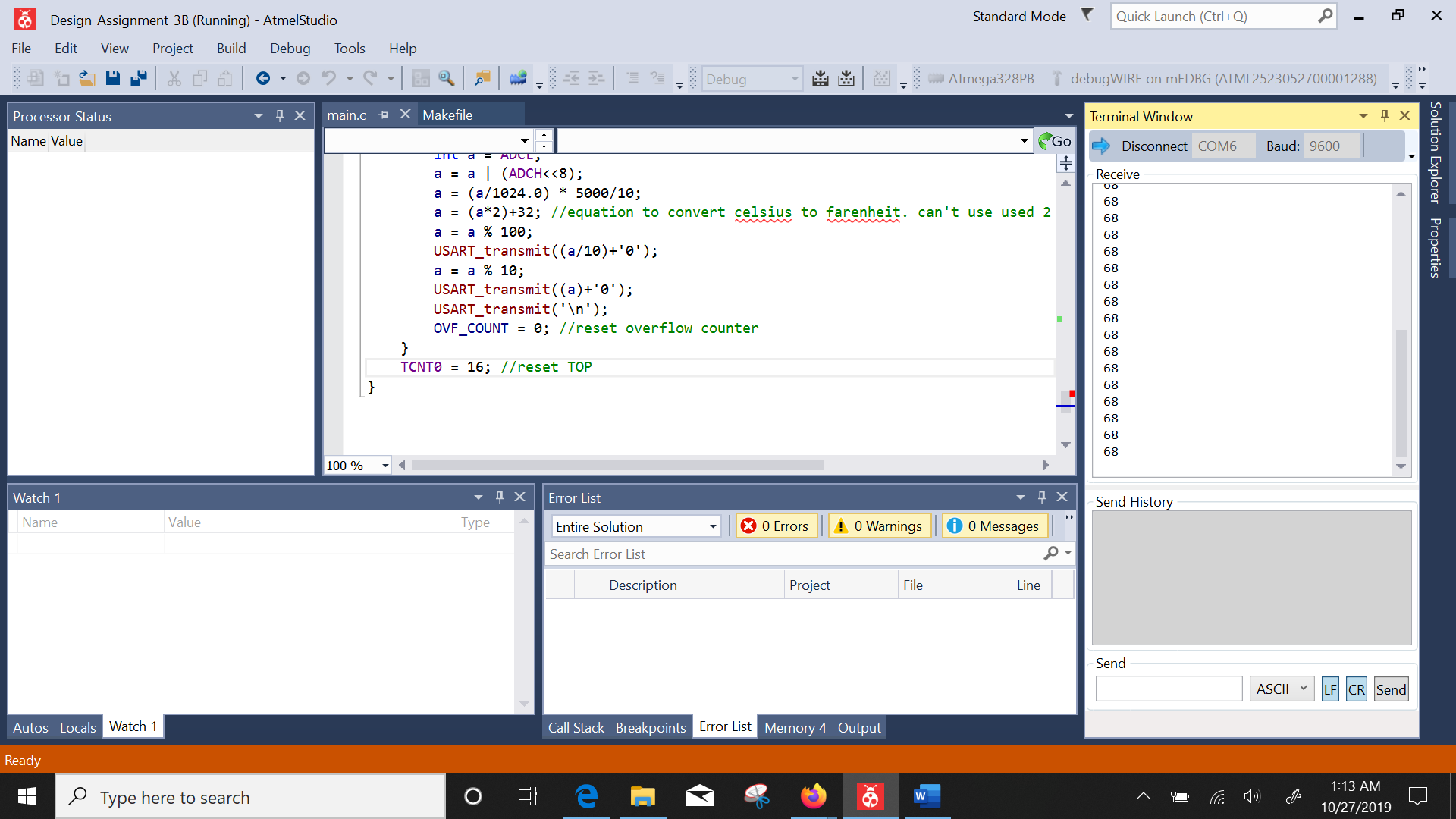
Atmega328PB, FTDI, ESP01, LM35

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A**
2. #define *F\_CPU* 16000000UL //16 Mhz.
3. #define BAUDRATE 115200 //Set Baud rate 9600.
4. #define BAUD\_PRESCALLER (((*F\_CPU* / (BAUDRATE \* 8UL))) - 1)
5. #include <avr/io.h>
6. #include <avr/interrupt.h>
7. #include <stdio.h>
8. #include <stdlib.h>
9. #include <util/delay.h>
10. #include <math.h>
11. void read\_adc(void); // Function Declarations
12. void adc\_init(void); //function that initializes ADC
13. void USART\_init(void); //initializes USART
14. unsigned char USART\_receive(void); //fn that receives USART
15. void USART\_transmit(unsigned char data); //function to send through USART
16. void USART\_tx\_string( char \*data );
17. volatile unsigned int adc\_temp;
18. char outs[20];
19. *uint8\_t* OVF\_COUNT = 0; //initialize the overflow count for interrupt
20. *uint8\_t* OVF\_LIMIT = 250; //set the limit the count can reach to set 1 sec delay
21. int main(void) {
22. adc\_init(); // Initialize the ADC (Analog / Digital Converter)
23. USART\_init(); // Initialize the USART (RS232 interface)
25. USART\_tx\_string("AT\r\n"); //
26. *\_delay\_ms*(500); // wait a bit
27. USART\_tx\_string("AT+CWMODE=3\r\n");
28. *\_delay\_ms*(500); // wait a bit
30. USART\_tx\_string("AT+CWJAP=\"WIFI\_NAME\_HERE\",\"WIFI\_PASSWORD\_HERE\""); // // Log in WiFi
31. *\_delay\_ms*(500);
33. USART\_tx\_string("AT+CIPMUX=0\r\n");
34. *\_delay\_ms*(500);
36. /////////////////////////////////////////////////
37. TCCR0A = 0x00; //normal operation
38. TCCR0B |= (1 << CS02); //set prescalar to 256
39. TCNT0 = 16; //TOP = 256-240 = 16
40. TIMSK0 |= (1 << TOIE0);
41. sei(); //enable interrupt
42. ////////////////////////////////////////////////////////
43. while(1)
44. {
45. //keep waiting
46. }
47. }
48. void adc\_init(void)
49. {
50. /\*\* Setup and enable ADC \*\*/
51. ADMUX = (0<<REFS1)| // Reference Selection Bits
52. (1<<REFS0)| // AVcc - external cap at AREF
53. (0<<ADLAR)| // ADC Left Adjust Result
54. (1<<MUX2)| // ANalog Channel Selection Bits
55. (0<<MUX1)| // ADC2 (PC2 PIN25)
56. (0<<MUX0);
57. ADCSRA = (1<<ADEN)| // ADC ENable
58. (0<<ADSC)| // ADC Start Conversion
59. (0<<ADATE)| // ADC Auto Trigger Enable
60. (0<<ADIF)| // ADC Interrupt Flag
61. (0<<ADIE)| // ADC Interrupt Enable
62. (1<<ADPS2)| // ADC Prescaler Select Bits
63. (0<<ADPS1)|
64. (1<<ADPS0); // Timer/Counter1 Interrupt Mask Register
65. }
66. /\* READ ADC PINS \*/
67. void read\_adc(void)
68. {
69. unsigned char i =4;
70. adc\_temp = 0;
71. while (i--) {
72. ADCSRA |= (1<<ADSC);
73. while(ADCSRA & (1<<ADSC));
74. adc\_temp+= ADC;
75. *\_delay\_ms*(50);
76. }
77. adc\_temp = adc\_temp /3; // Average a few samples
78. //int tempf= (adc\_temp\*5\*100);
79. *snprintf*(outs,sizeof(outs),"%3d\r\n", adc\_temp);// print it
80. USART\_tx\_string(outs);
81. }
82. void USART\_init(void)
83. {
84. UBRR0H = (*uint8\_t*)(BAUD\_PRESCALLER >> 8);
85. UBRR0L = (*uint8\_t*)(BAUD\_PRESCALLER);
86. UCSR0A = (1 << U2X0); // double mode for more accuracy
87. UCSR0B = (1 << RXEN0) | (1 << TXEN0);
88. UCSR0C = (3 << UCSZ00);
89. }
90. /\* SEND A STRING TO THE RS-232 \*/
91. unsigned char USART\_receive(void)
92. {
93. while(!(UCSR0A & (1 << RXC0)));
94. return UDR0;
95. }
96. void USART\_transmit(unsigned char data) { // Function to transmit ASCII value into UDR0
97. while (!(UCSR0A & (1 << UDRE0))); // Keep Checking until UDRE0 data register 'High' to break loop
98. UDR0 = data; // Store unsigned char serial data into UDR0
99. }
100. void USART\_tx\_string(char\* data)
101. {
102. while(\*data != 0x00)
103. {
104. USART\_transmit(\*data);
105. data++;
106. }
107. }
108. ISR(TIMER1\_OVF\_vect)
109. {
110. OVF\_COUNT++; //increment the overflow counter
111. if (OVF\_COUNT == OVF\_LIMIT) //check to see if the limit was reached
112. {
113. ADCSRA|=(1<<ADSC); //start conversion
114. while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish
116. ADCSRA |= (1<<ADIF);
117. int a = ADCL;
118. a = a | (ADCH<<8);
119. a = (a/1024.0) \* 5000/10;
120. a = (a\*2)+32; //equation to convert celsius to farenheit. can't use used 2 instead of 9/5

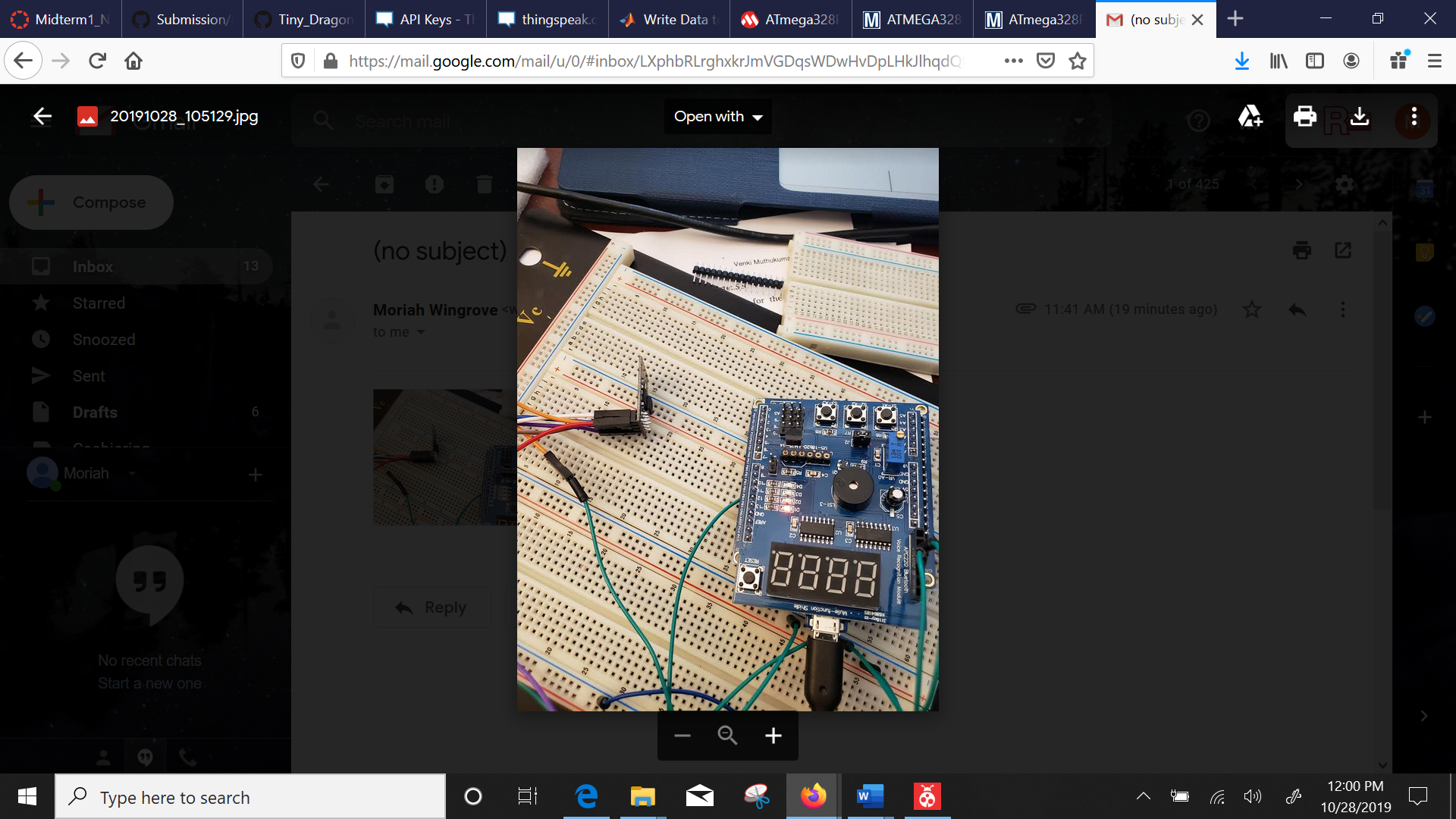

124. USART\_tx\_string("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n"); // Connect API KEy
125. *\_delay\_ms*(500);
126. USART\_tx\_string("AT+CIPSEND=51\r\n");
127. *\_delay\_ms*(500);
128. USART\_tx\_string("GET /update?key=QTJ31HQI0KR0Q5AP&field1=");// Send Value
129. *\_delay\_ms*(500);
130. a = a % 100;
131. USART\_transmit((a/10)+'0');
132. a = a % 10;
133. USART\_transmit((a)+'0');
134. *\_delay\_ms*(500);
135. USART\_transmit('\n');
136. *\_delay\_ms*(500);
138. OVF\_COUNT = 0; //reset overflow counter
139. }
140. TCNT0 = 16; //reset TOP
141. }
142. **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**

**Student Academic Misconduct Policy**

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

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