CPE301 – SPRING 2019

Design Assignment 3B

Student Name: Bruce Moquete

Student #: 1013123266

Student Email: moquete@unlv.nevada.edu

Primary Github address: https://github.com/dcon99/CMX2345.git

Directory: All Design Assignments/DA3B

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

Atmega328P

LM35DZ Temperature Sensor

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A**

N/A

1. **DEVELOPED MODIFIED CODE OF TASKS**
2. /\*
3. This program is meant to demonstrate the ADC (Analog-to-Digital Converter) of the
4. ATmega328P alongside its UART and Timer1 capabilities. The purpose of this code is
5. to convert the analog value provided by an LM35DZ temperature sensor using the
6. ADC and then transmit the newly converted value onto a terminal through UART
7. every second (approximately a second) using Timer1.
8. \*/
9. #define *F\_CPU* 8000000UL //Sets the clock speed of the MCU
10. #define BAUD 9600 //Sets the baud rate for the UART to transmit
11. #define BAUDRATE ((*F\_CPU*) / (BAUD \* 8UL)-1) // Set Baud Rate Value for UBRR
12. /////////////////////////////////////////////////////////////////////////////////////////////////////
13. #include <avr/io.h>
14. #include <util/delay.h>
15. #include <avr/interrupt.h>
16. #include <stdlib.h>
17. #include <stdio.h>
18. /////////////////////////////////////////////////////////////////////////////////////////////////////
19. char String[]="The Current Temperature Is: "; //Character array for string
20. char LineBreak[]="\r\n"; //New line and return string array for a neater transmission
21. char String\_num[]; //String array to hold an int after converting to string for int
22. char String\_flt[]; //String array to hold a double value after converting to string
23. volatile *uint8\_t* time\_ovf; //Integer to hold the amount of times the timer overflows
24. float temperature; //Float variable used to hold the measured analog value from the LM35DZ
25. char Degrees[]="° F"; //Used to make the output temperature value readable as Fahrenheit
26. /////////////////////////////////////////////////////////////////////////////////////////////////////
27. //A function used to initialize Timer1 of the ATmega328P with a prescale of 64
28. void timer1\_init()
29. {
30. //Set up timer with a prescale of 64
31. TCCR1A |= (0<<COM1A1) | (0<<COM1A0);
32. TCCR1B |= (1<<CS11)|(1 << CS10);

35. //Initialize counter
36. TCNT1 = 0;
38. //Enable overflow interrupt
39. TIMSK1 |= (1 << TOIE1);
41. //Initialize overflow counter variable
42. time\_ovf = 0;
43. }
44. //Used to initialize the UART of the ATmega328P with a baud rate of 9600 and enables global interrupts.
45. void initialize\_UART(){
46. UBRR0H = (*uint8\_t*)(BAUDRATE>>8);
47. UBRR0L = (*uint8\_t*)(BAUDRATE);
48. UCSR0B = (1<<RXEN0)|(1<<TXEN0);
49. UCSR0C = ((1<<UCSZ00)|(1<<UCSZ01));
50. sei();
52. }
53. //Function used to send one character (8-bits) at a time through USART
54. void USART\_send( unsigned char data){
56. while(!(UCSR0A & (1<<UDRE0)));
57. UDR0 = data;
59. }
60. //Function used to send converted numbers through a string array through USART
61. void USART\_send\_num( unsigned int data){
63. while(!(UCSR0A & (1<<UDRE0)));
64. UDR0 = data;
66. }
67. //Function used to receive data from USART
68. unsigned char USART\_receive(void){
70. while(!(UCSR0A & (1<<RXC0)));
71. return UDR0;
73. }

76. //Function used to transmit an array of characters (string) through USART one character at a time.
77. void USART\_putstring(char\* StringPtr){
79. while(\*StringPtr != 0x00){
80. USART\_send(\*StringPtr);
81. StringPtr++;}
83. }
84. //Function used to transmit an array of characters of an integer converted into a string through USART one character at a time.
85. void USART\_putnumber(char\* String\_num, unsigned int temperature){
87. String\_num = (*utoa*(temperature,String\_num,10)); //Converts an int to a string
88. while(\*String\_num != 0x00){
89. USART\_send\_num(\*String\_num);
90. String\_num++;}
92. }
93. //Function used to transmit an array of characters of an double converted into a string through USART one character at a time.
94. void USART\_putflt(char\* String\_flt, float temperature){
96. String\_flt = *dtostrf*(temperature,0,2,String\_flt); //Converts a double to a string
97. while(\*String\_flt != 0x00){
98. USART\_send\_num(\*String\_flt);
99. String\_flt++;}
101. }
102. //Function used to initialize the ADC with an adc-prescaler of 64 and using a reference voltage of 5V
103. void ADC\_init(){
105. ADCSRA |= ((1<<ADPS2)|(1<<ADPS1)|(0<<ADPS0)); //Prescaler at 64 so we have an 125Khz clock source
106. ADMUX |= (0<<REFS1)|( 1 << REFS0 );
107. }
108. //Interrupt Service Routine used for Timer1 Overflow.
109. ISR(TIMER1\_OVF\_vect)
110. {
111. //Keep track of number of overflows
112. time\_ovf++;
114. }
115. /////////////////////////////////////////////////////////////////////////////////////////////////////
116. //This function is used to start a single-conversion from an ADC of the user's choice and store the converted value in the temperature variable.
117. void ADC\_Read( *uint8\_t* channel )
118. {
119. // Select the ADC channel to be read.
120. ADMUX |= channel;
121. // Turn on the ADC.
122. ADCSRA |= ( 1 << ADEN );
123. // Start the conversion.
124. ADCSRA |= ( 1 << ADSC );
126. while( ADCSRA & ( 1 << ADIF ) );
128. temperature= ADC; //read upper 8bits
130. temperature = ((temperature/2)\*1.8)+32; //Convert converted value to Fahrenheit.
131. }
132. int main(void)
133. {
134. initialize\_UART(); //Initializes UART
135. ADC\_init(); //Initialize ADC
136. ADC\_Read(0); //Do one conversion (slowest conversion) for a table conversion process during the main loop.
137. timer1\_init(); //Initializes Timer1
139. while (1)
140. {
142. if(time\_ovf >= 40){ //Uses increments of 4 for a timer1 overflow interrupt of roughly 1 second.
144. ADC\_Read(0); //Using ADC 0
145. //USART\_putstring(String);
146. //USART\_putflt(String\_num, temperature);
147. //USART\_putstring(Degrees);
148. USART\_putstring(LineBreak);
149. //USART\_putstring(LineBreak);


153. TCNT1 = 0;
154. time\_ovf = 0;
156. }
158. else;
160. }
161. }
162. **SCHEMATICS**
163. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**
164. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**
165. **VIDEO LINKS OF EACH DEMO**
166. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

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

Bruce Moquete