Guillermo Gálvez

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. | TASK 1 – C AVR Program |  |  |
| 3. | Schematic |  |  |
| 4. | TASK 2 – Atmel Studio Visualization |  |  |
| 5. | Flow Chart |  |  |
| 6. | Bread Board Setup |  |  |
| 7. | Vid Links |  |  |
| 8. | Git links |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

LM34 temperature sensor

ATmega XPlained Mini 328P

Jumper wires

Bread Board

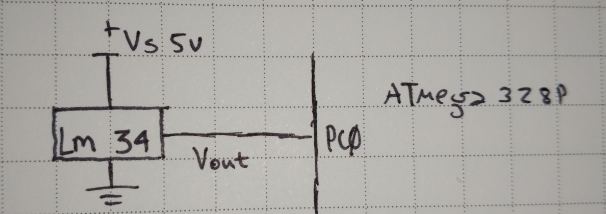
1. **DEVELOPED CODE OF TASK 1/A**
2. /\*
3. \* da3.c
4. \*
5. \* Created: 3/25/2018 9:45:22 AM
6. \* Author : Guillermo Gálvez
7. \*/
8. #include <stdlib.h>
9. #include <avr/io.h>
10. #include <avr/interrupt.h>
11. #include <util/delay.h>
12. #include <stdio.h>
13. #include <stdint.h>
14. #define *F\_CPU* 8000000UL
15. #define FOSC 16000000 //Frequency
16. #define BAUD 9600 //Baud Rate
17. #define MYUBRR FOSC/16/BAUD-1 //Automatic BAUD rate calculation
18. volatile *uint8\_t* adcValue;
19. /\*Function Declarations\*/
20. void USART\_Init();
21. void ADC\_Init();
22. void outputChr(unsigned char c);
23. void outputStr(char \*c);
24. void outputNum(float t);
25. void readTemp();
26. /\*Interrupt Service Routines\*/
27. ISR(ADC\_vect)
28. {
29. ADCSRA |= (1 << ADIF); //Reset flag
30. adcValue = ADCH; //MSB 8-bits of ADC form left shift of ADLAR
31. }//end ISR(ADC\_vect(
32. ISR(TIMER1\_OVF\_vect)
33. {
34. TIFR1 |= (1 << TOV1); //Clr Flag
35. readTemp();
36. }//end ISR(TIMER0\_OVF
37. //Attempted Didn't work
38. // static int outChr(char t, FILE \*stream); //Output function
39. // static FILE usartout = FDEV\_SETUP\_STREAM(outChr, NULL, \_FDEV\_SETUP\_WRITE); //Stream pointer
40. int main(void)
41. {
42. //stdout = &usartout; //Attempt at output stream
44. ADC\_Init(); //initialize ADC
45. USART\_Init(); //UART initialization
47. TCCR1B = (1 << CS12); //Set prescale 256
48. OCR1A = 31094; //TCNT1 for 1s real time 31250 reduced to account for
49. //additional delays in print functions
50. TIMSK1 |= (1 << TOIE1); //enable OVF interrupt

53. while (1)
54. {
56. }//end while(1)
58. }//end main
59. void USART\_Init()
60. {
61. /\*Attempt at stream didn't work
62. unsigned int BR;
63. BR = ((F\_CPU/16)/BAUD)-1;
64. UBRR0H = (unsigned char) (BR>>8);
65. UBRR0L = (unsigned char) BR;
66. \*/
68. /\*Set Baud Rate\*/
69. UBRR0H = (MYUBRR>>8); //Shift MSB "top" of UBRR0H
70. UBRR0L = MYUBRR; //Shift LSB "bot" of UBRR0L
72. UCSR0B |= (1 << RXEN0) | (1 << TXEN0); //Enable Rec and Trans
73. UCSR0B |= (1 << RXCIE0); //Enable Rec INT
74. UCSR0C |= (1 << UCSZ01) | (1 << UCSZ00); //Set frame 8-bit, 1 STP
75. }//end USART\_int
76. void ADC\_Init()
77. {
78. DDRC = 0; //Set PORTC as input for adc
79. DIDR0 = 0x01; //Disable Digi input on ADC0
81. ADMUX = 0; //
82. ADMUX |= (1 << REFS0); //Use Vcc Ref voltage
83. ADMUX |= (1 << ADLAR); //Left adjust ADC Reg, 8-bit Resolution
85. //Enable ADC, Start Conv, Set Auto Trig Enable
86. ADCSRA |= (1 << ADEN); //Enable ADC
87. ADCSRA |= (1 << ADATE); //Set ADC Auto Trig
88. ADCSRA |= (1 << ADIE); //Enable Interrupts
89. ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (0 << ADPS0); //8MHz, Pre-Scale 64 = 125KHz
90. ADCSRA |= (1 << ADSC); //Start Conversion
92. ADCSRB = 0; //Free running mode
93. sei(); //Enable interrupts
95. }//end ADC\_Init
96. void outputChr(unsigned char c)
97. {
98. UDR0 = c; //Display Char on Serial
99. *\_delay\_ms*(800);
100. }//end outputChr
101. void outputNum(float t)
102. {
103. UDR0 = t;
104. *\_delay\_ms*(500);
105. }
106. void outputStr(char \*c)
107. {
108. unsigned int i = 0; //loop control
109. while(c[i] != 0)
110. outputChr(c[i++]);
111. }//end outpuStr
112. void readTemp()
113. {
114. char seeTemp[8];
115. float lm34\_0; //For ASCII Temp output
116. float lm34\_1; //For showing valued of ADC
118. ADCSRA |= (1 << ADSC); //Start conversion
119. while((ADCSRA & (1 << ADIF)) == 0); //Wait for conversion to finish

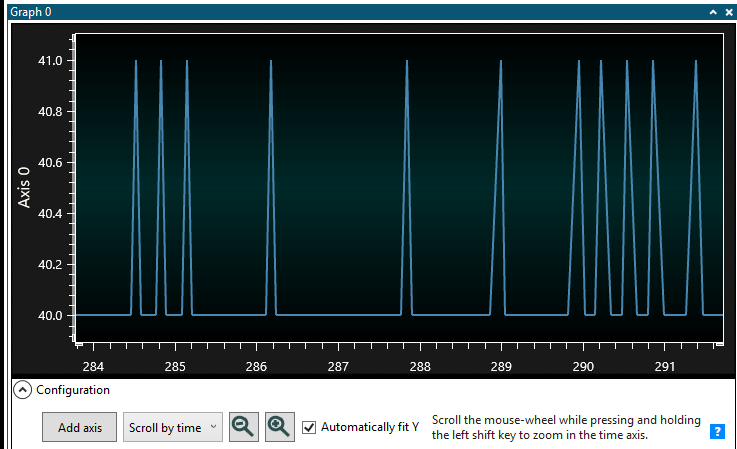

123. /\*To show ADC value as float on graph at ~78°F it's ~40\*/
124. /\*On terminal it will show up as some ASCII character depending on value\*/
125. lm34\_1 = adcValue;
126. outputNum(lm34\_1);

129. /\*Conversion to °F\*/
130. lm34\_0 = (adcValue \* 5.0 / 0x100) \* 100.0; //(ADC \* 5 = 200 /256) \* 100
132. *dtostrf*(lm34\_0, 5, 2, seeTemp); //Float to char conversion
134. outputStr(" Temp is: "); //Count
135. outputStr(seeTemp);
137. outputStr("\n\r");
139. //Prints non-sense
140. /\*
141. printf("Temp is: ");
143. outputStr(seeTemp);
145. printf("°F");
147. printf('\n');
148. printf('\r');
149. \*/
150. }
151. //Attempt at Stream File output Prints non-sense \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
152. /\*
153. void outputStr(char \*c)
154. {
155. while((\*c != '\0'))
156. {
157. while(!(UCSR0A & (1 << UDRE0))); //Wait until Tx buffer is ready
158. UDR0 = \*c; //Send char to Tx
159. c++; //Next char
160. }
161. }//end outpuStr
162. static int outChr(char t, FILE \*stream)
163. {
164. while(!(UCSR0A & (1 << UDRE0))); //Wait for UDR to clr
165. UDR0 = t; //Send Char to stream Tx
166. return 0;
167. }//End OutChr
168. \*/
169. **SCHEMATICS**

Use fritzing.org 🡨 Freezes

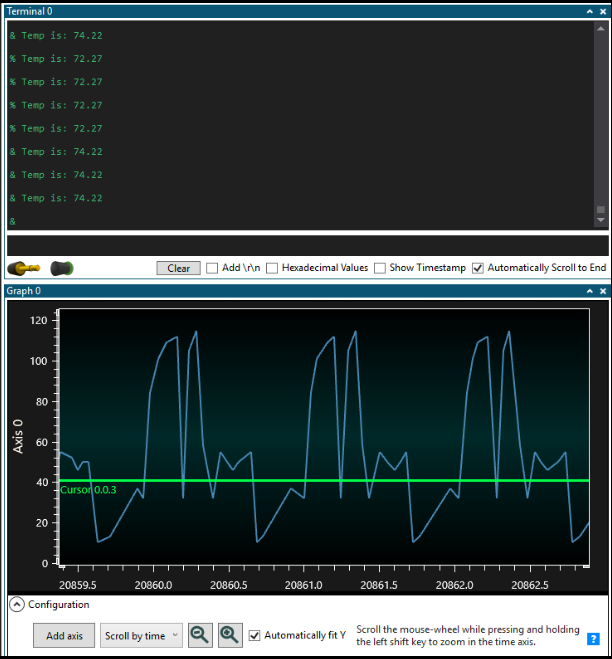


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**



In the picture above, you find the ADC value only, for a temperature of ~78°F the output is 40. The next two pictures you will see the cursor placed at a value of 40. The graph outputs all the characters being printed through the UART however you will see one peak that moves as the value of the ADC changes. This peak corresponds to the peak in the picture above.

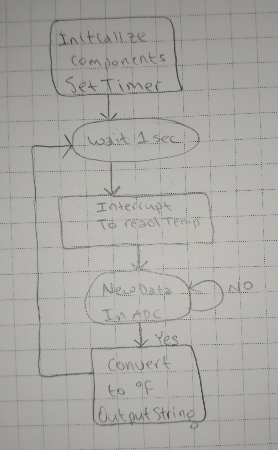




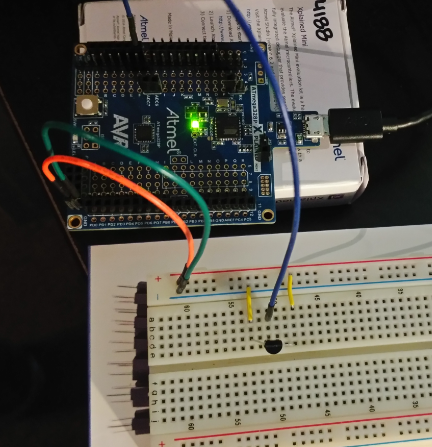
In this last picture you can see that the delay between ADC measurements is approximately 1 second as requested. The code is setup so that an OVF from timer1 will trigger an ISR that will call a function to take an ADC value and display the temperature.



1. **Flow Chart**



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



1. **VIDEO LINKS OF EACH DEMO**

<https://www.youtube.com/watch?v=pim_7S6KkDQ>

1. **GITHUB LINK OF THIS DA**

<https://github.com/galveg1/Design-Assignments/tree/master/DA3>

**Student Academic Misconduct Policy**

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

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

Guillermo Gálvez