Jeffrey Razon

CPE301 – SPRING 2018

Design Assignment 03

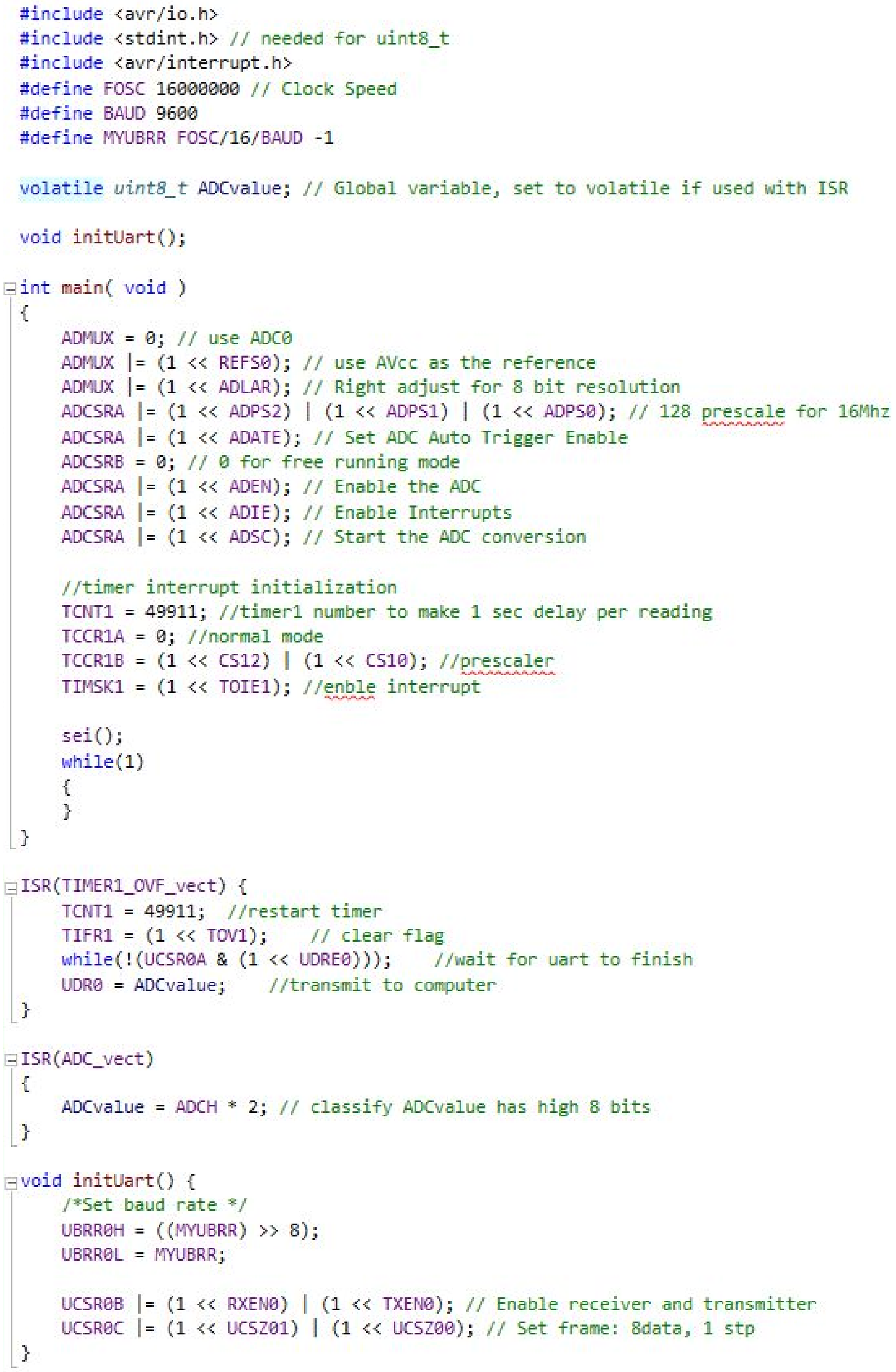
DO NOT REMOVE THIS PAGE DURING SUBMISSION:

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.	INITIAL CODE OF TASK 1/A		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E		
4.	SCHEMATICS		
5.	SCREENSHOTS OF EACH TASK OUTPUT		
5.	SCREENSHOT OF EACH DEMO		
6.	VIDEO LINKS OF EACH DEMO		
7.	GOOGLECODE LINK OF THE DA		

Task 1/A: Write a C AVR program that will monitor the LM34/35 connected to an Analog pin to display the temperature in F 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.

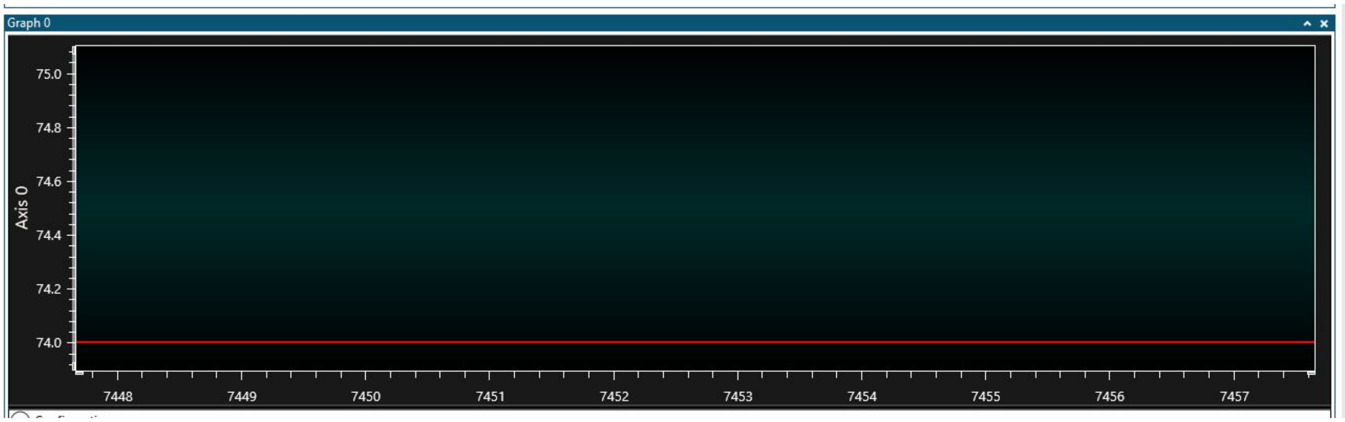
a) Full code (in C)



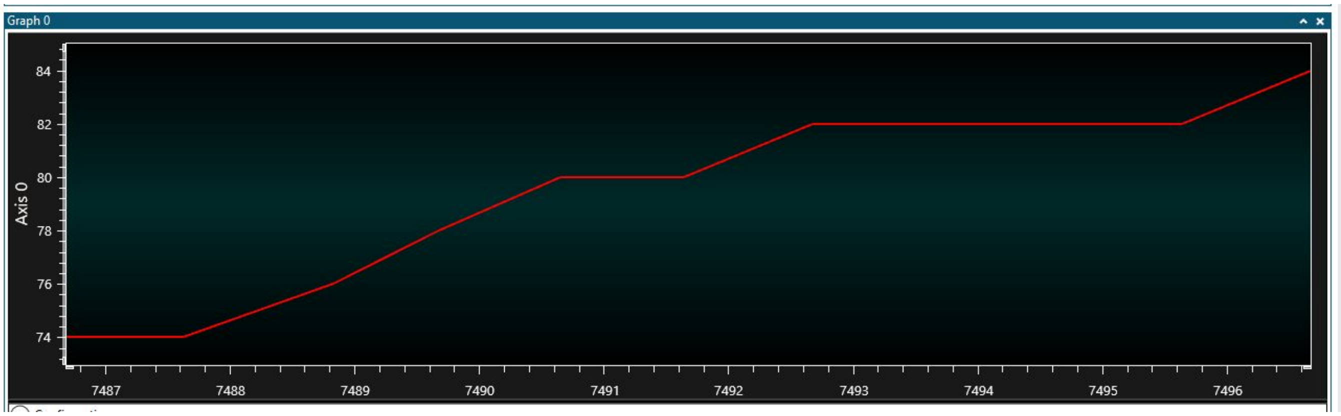
```c
#include <avr/io.h>
#include <stdint.h> // needed for uint8_t
#include <avr/interrupt.h>
#define FOSC 16000000 // Clock Speed
#define BAUD 9600
#define MYUBRR FOSC/16/BAUD -1

volatile uint8_t ADCvalue; // Global variable, set to volatile if used with ISR

void initUart();

int main( void )
{
    ADMUX = 0; // use ADC0
    ADMUX |= (1 << REFS0); // use AVcc as the reference
    ADMUX |= (1 << ADLAR); // Right adjust for 8 bit resolution
    ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // 128 prescale for 16Mhz
    ADCSRA |= (1 << ADATE); // Set ADC Auto Trigger Enable
    ADCSRB = 0; // 0 for free running mode
    ADCSRA |= (1 << ADEN); // Enable the ADC
    ADCSRA |= (1 << ADIE); // Enable Interrupts
    ADCSRA |= (1 << ADSC); // Start the ADC conversion

    //timer interrupt initialization
    TCNT1 = 49911; //timer1 number to make 1 sec delay per reading
    TCCR1A = 0; //normal mode
    TCCR1B = (1 << CS12) | (1 << CS10); //prescaler
    TIMSK1 = (1 << TOIE1); //enble interrupt

    sei();
    while(1)
    {
    }
}

ISR(TIMER1_OVF_vect) {
    TCNT1 = 49911;  //restart timer
    TIFR1 = (1 << TOV1);    // clear flag
    while(!(UCSR0A & (1 << UDRE0)));    //wait for uart to finish
    UDR0 = ADCvalue;    //transmit to computer
}

ISR(ADC_vect)
{
    ADCvalue = ADCH * 2; // classify ADCvalue has high 8 bits
}

void initUart() {
    /*Set baud rate */
    UBRR0H = ((MYUBRR) >> 8);
    UBRR0L = MYUBRR;

    UCSR0B |= (1 << RXEN0) | (1 << TXEN0); // Enable receiver and transmitter
    UCSR0C |= (1 << UCSZ01) | (1 << UCSZ00); // Set frame: 8data, 1 stp
}
```

Task 2/B: Use the ATMEL Studio Data Visualizer or any Charting program to display the values in time.
- For this assignment, I used ATMEL Studio Data Visualizer to display the ADCvalue, which is assigned the UDR0 register.

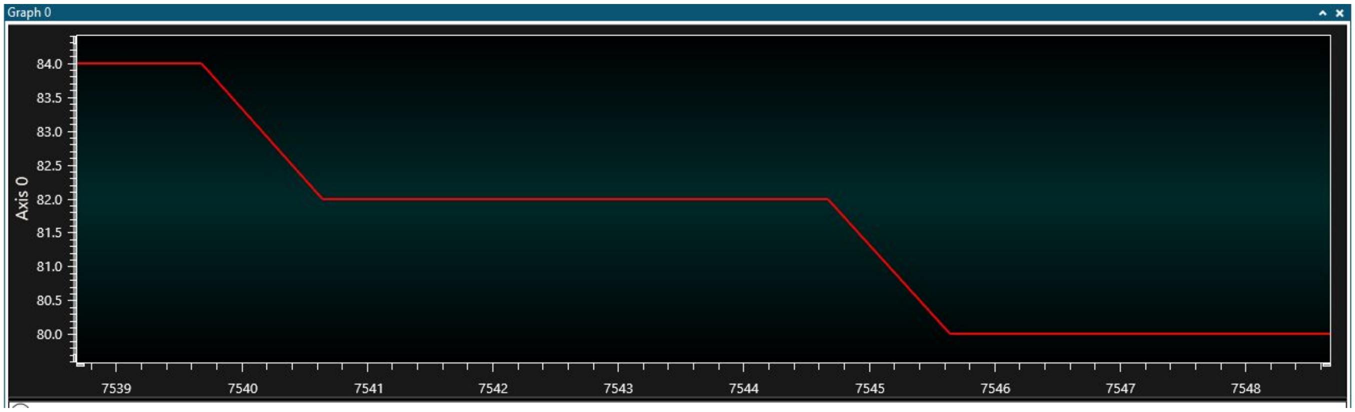
a) Constant temperature at 74^{o}F (w/o finger contact to LM34)



b) Temperature raising to the $80s^{o}$F (Pinching the LM34)

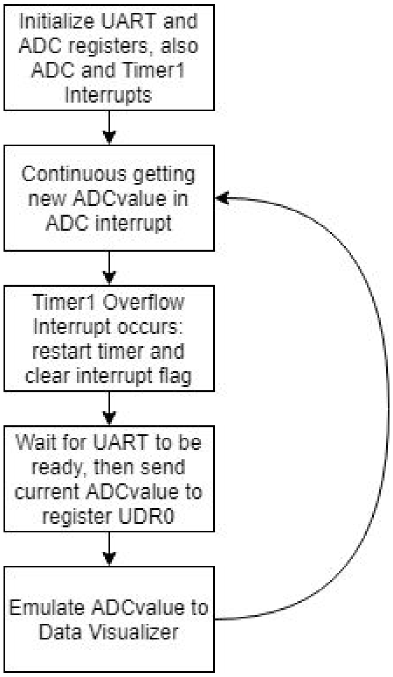


c) Temperature decreasing (no longer



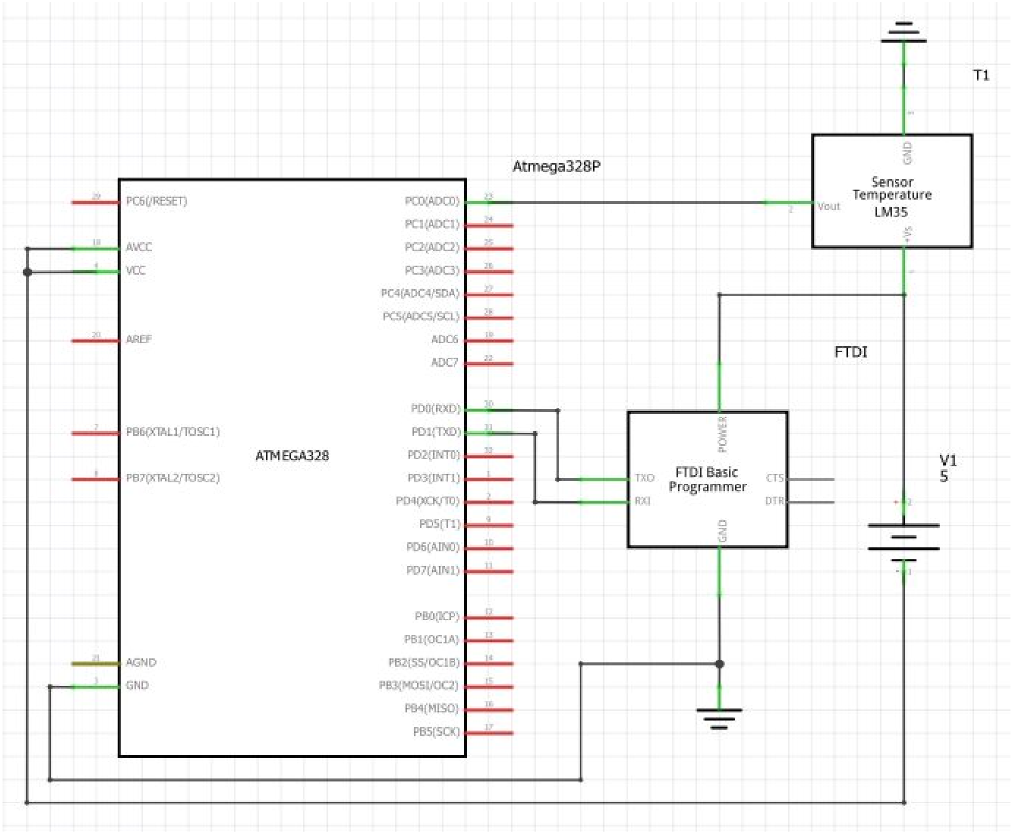
Flowchart:

a) Flowchart


Initialize UART and ADC registers, also ADC and Timer1 Interrupts
Continuous getting new ADCvalue in ADC interrupt
Timer1 Overflow Interrupt occurs: restart timer and clear interrupt flag
Wait for UART to be ready, then send current ADCvalue to register UDR0
Emulate ADCvalue to Data Visualizer

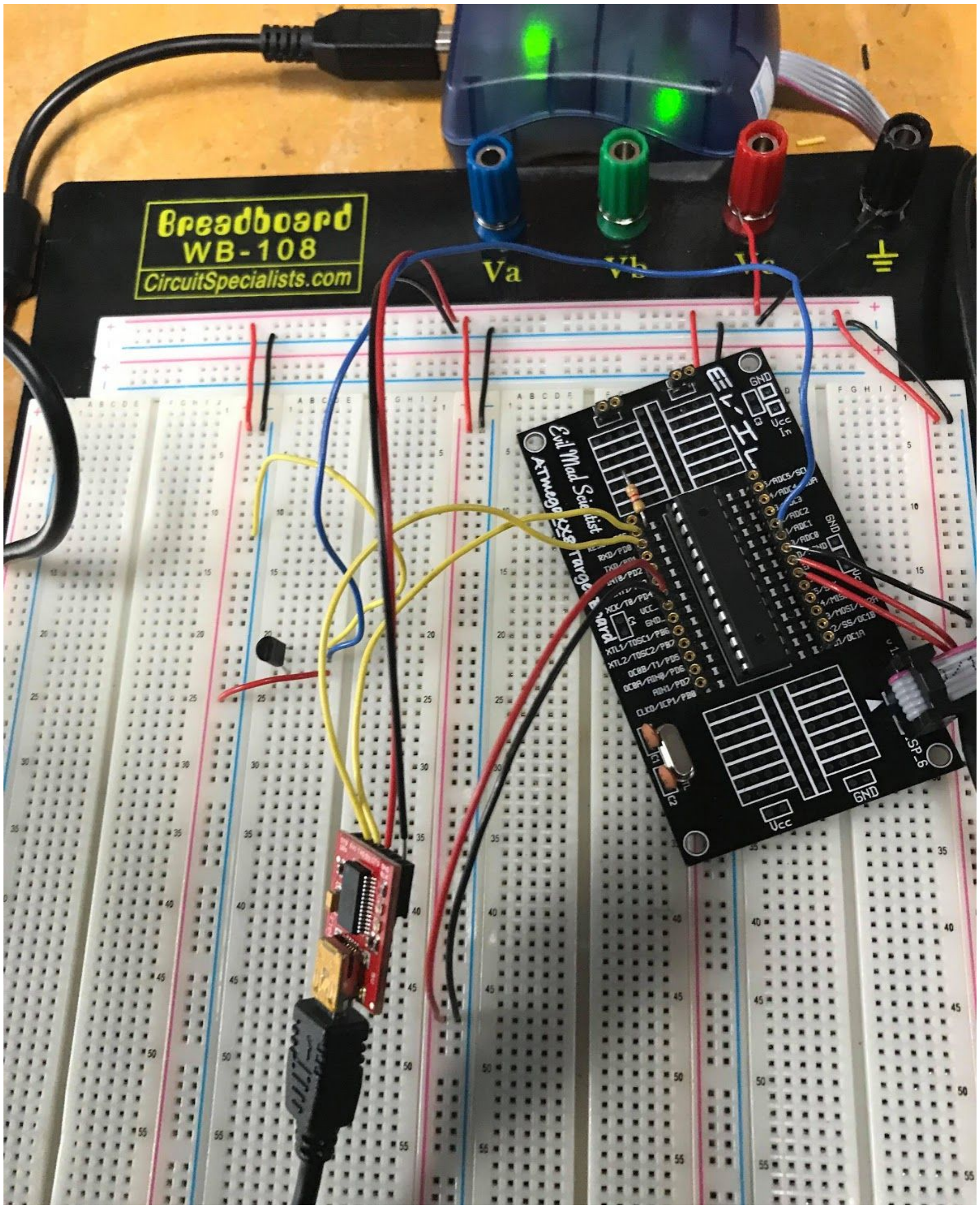
Schematics:

a) Fritzing schematics



Physical Set-up:

a) Breadboard circuit (ATMega328P and FTDI chip)



GITHUB LINK: https://github.com/JeffinVegas/EmbSys.git

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Jeffrey Razon