CPE301 – SPRING 2019

Design Assignment 3B

Student Name: Tanner Tindall

Student #: 8000733733

Student Email: tindat1@unlv.nevada.edu

Primary Github address: <https://github.com/TannerTindall51>

Directory: <https://github.com/TannerTindall51/tindalltannerm_submission/tree/master/Design_Assignments/DA3B>

Design Assignment 3B:

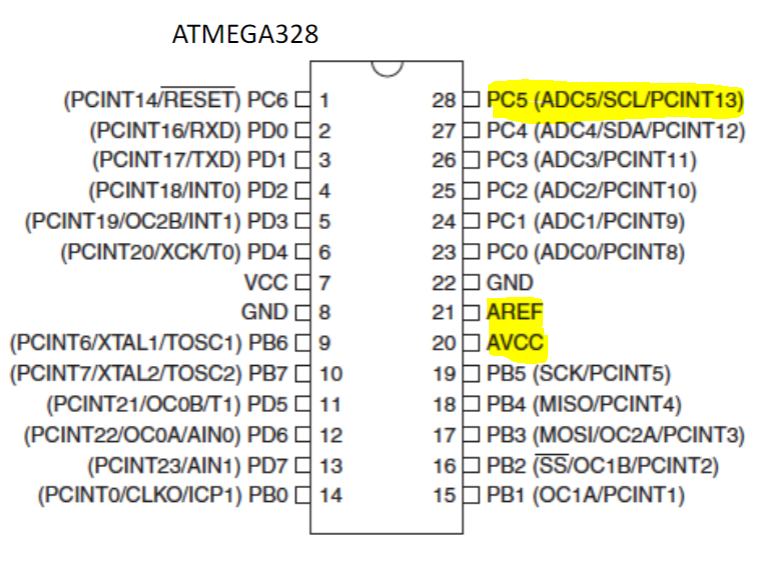
The goal of the assignment is to modify the above codes to do the following:

1. Write a C AVR program that will monitor the LM34/35 connected to an Analog pin (PC5 or PC0) to display the temperature in Centigrade (C) and Fahrenheit (F) on the serial terminal every 0.5 sec. Use a timer with interrupt for the 0.1 sec delay and use a counter variable to implement the 0.5 sec delay. Use a FTDI chip for serial to USB conversion and display the values in the terminal.

2. Use the ATMEL Studio Data Visualizer or any Charting program to display the values in time

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

* Atmel Studio 7.0 (Assembler, Simulator, & Debugger)
* Atmega328PB-Xmini
* Micro USB
* LM35 Temperature Sensor



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

|  |
| --- |
|  |

//

//Design Assignment 3B

//Tanner Tindall

//

#define BAUD 9600

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/setbaud.h>

#include <avr/interrupt.h>

#include <stdlib.h>

#include <stdio.h>

volatile *uint16\_t* delay = 6249; //initializing a global variable to be used as primary overflow for timer

volatile *uint8\_t* counter = 5; //initializing a global variable to be used as counter for delay

volatile *uint8\_t* adcTemp\_C; //initializing a global variable to be used as Celsius value

volatile *uint8\_t* adcTemp\_F; //initializing a global variable to be used as Fahrenheit value

void USART\_init(void) //initialize UART

{ //the baud rate consists of 16 bits therefore a low and high is needed

UBRR0H = *UBRRH\_VALUE*; //high end of baud rate

UBRR0L = *UBRRL\_VALUE*; //low end of baud rate

UCSR0C = \_BV(UCSZ01) | \_BV(UCSZ00); //assigning bit values to UCSR0C

UCSR0B = \_BV(RXEN0) | \_BV(TXEN0); //enabling Reception Complete Interruption & turning on RX/TX

}

void USART\_tx(char\*data) //outputs data to terminal

{

while((\*data!='\0')) //loop until all data is emptied

{

while(!(UCSR0A & (1 << UDRE0))); //wait until data register in emptied

UDR0 = \*data; //once emptied, import into data register UDR0

data++; //increments pointer position for data

}

}

void timer(){

OCR1A = delay;

TCNT1 = 0x00; //initializes the 16 bit timer1 register

TCCR1A = 0x00; //sets the timer to "CTC" Mode

TCCR1B |= (1<<WGM12) | (1 << CS12); //sets the timer to "CTC" Mode & prescaler value to 256

TIMSK1 = (1<<OCIE1A); //enable timer0 overflow interrupt

sei(); //enable interrupts

}

ISR(TIMER1\_COMPA\_vect){ //timer interrupt function

TIFR1 |= 0x02; //reset/clear interrupt

counter++; //increment delay counter

}

void ADC\_init (void)

{

ADMUX = (0<<REFS1)| (1<<REFS0)| //set Vref

(0<<ADLAR)| //set ADC LEFT adjust

(1<<MUX2)|(0<<MUX1)|(0<<MUX0); //set ADC/PC5

ADCSRA = (1<<ADEN)| //enable ADC

(0<<ADSC)| //enable ADC conversion

(0<<ADATE)| //enable ADC trigger

(0<<ADIF)| //enable ADC interrupt flag

(0<<ADIE)| //enable ADC interrupt

(1<<ADPS2)| (0<<ADPS1)|(1<<ADPS0); //set ADC prescaler

}

void adcRead(void){

for(int i = 0; i < 10; i++)

{

adcTemp\_F = 0; //initilize/reset value upon reloop

ADCSRA |= (1<<6); //enable ADC conversion

while (!(ADCSRA&(1<<4))); //waits for conversion to complete by ADIF flag

ADCSRA |= (1<<4); //reset when complete

adcTemp\_F += ADC;

i++;

}

adcTemp\_F = (adcTemp\_F/10); //divide by 10 to avg 10 samples from above

adcTemp\_C = (adcTemp\_F - 32) \* 5 / 9; //conver to C

}

int main(void)

{

USART\_init(); //initialize USART

ADC\_init(); //initialize ADC

timer(); //calls timer function

char array1[30];

char array2[30];

while (1)

if (counter >= 5) //since 100ms\*5 = 500ms, this will serve as .5s delay

{

adcRead();

USART\_tx("\n");

*snprintf*(array1,sizeof(array1), "%f\r\n", adcTemp\_C);

USART\_tx("Celsius: ");

USART\_tx(array1);

*snprintf*(array2,sizeof(array2), "%f\r\n", adcTemp\_F);

USART\_tx("Fahrenheit: ");

USART\_tx(array2);

USART\_tx("\n");

counter = 0;

}

}

1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**

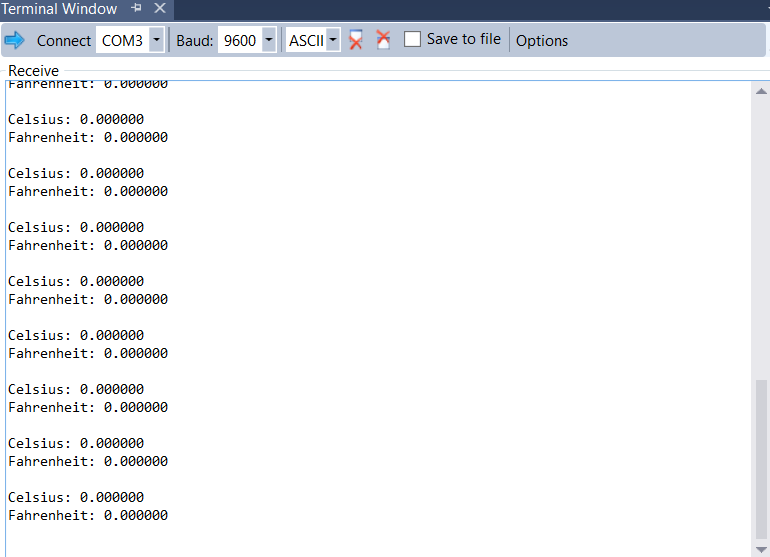
N/A

1. **SCHEMATICS**

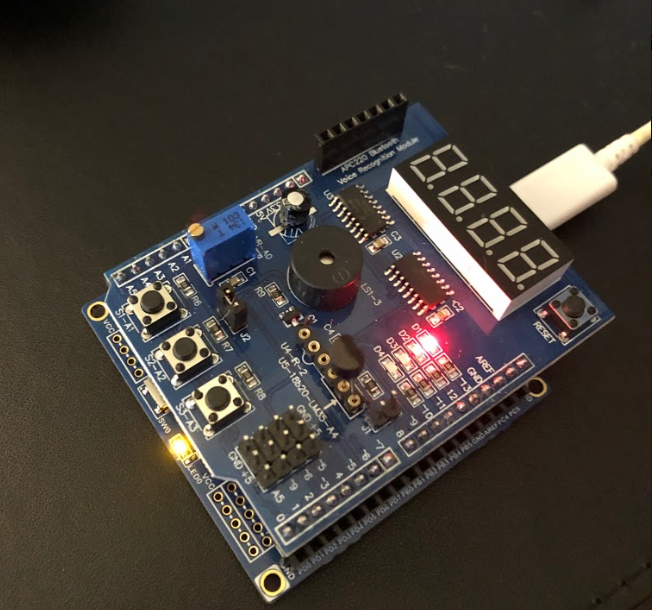
N/A

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

Due to a faulty sensor, I am unfortunately left with 0’s as the output. However, a simple replacement of this sensor (which I do not posess nor have access to another classmates amid the CoronaVirus quaratine) would in fact yeild to correct values as the software remains intact and ready to compile. With faulty hardware, the data visualizer would yield meaningless plots.



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



1. **VIDEO LINKS OF EACH DEMO**

Demo Video: https://youtu.be/eG6yDu9WmEs

1. **GITHUB LINK OF THIS DA**

<https://github.com/TannerTindall51/tindalltannerm_submission/tree/master/Design_Assignments/DA3B>

**Student Academic Misconduct Policy**

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

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

Tanner Tindall