Damian Cisneros

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. | 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 |  |  |
|  |  |  |  |
|  |  |  |  |

Damian Cisneros

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

List of Components used:

ATmega328p

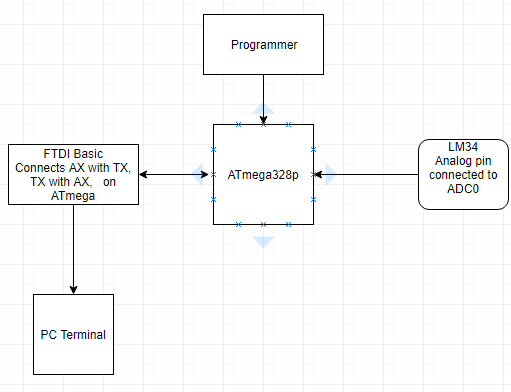
LM34

Breadboard

10kΩ resistor

100nF capacitor

FIDI232R



Block diagram with pins used in the Atmega328P

1. **INITIAL/DEVELOPED CODE OF TASK 1/C Code**

/\*

\* DA3\_Task1.c

\*

\* Created: 3/18/2018 1:54:58 PM

\* Author : Damian Cisneros

\* Description : This program monitors temperature using an LM34 sensor. It

\* reads the temperature every 1s and displays it on a serial

\* terminal. Using 8Mhz clock

\*/

#define BAUD 9600

#define F\_CPU 8000000UL

#include <stdlib.h>

#include <avr/io.h>

#include <avr/interrupt.h>

#include <stdio.h>

//#include <stdint.h> //needed for uint8\_t

#include <util/delay.h>

static int put\_char**(**char c**,** FILE **\***stream**);**

static FILE mystdout **=** FDEV\_SETUP\_STREAM**(**put\_char**,** **NULL,** \_FDEV\_SETUP\_WRITE**);**

void init\_ADC**();**

void init\_USART**();**

void USART\_tx\_string**(**char**\*);**

//void writeData(unsigned char);

int main**(**void**)**

**{**

float ADCvalue**;** //holds converted ADC value

char c**[**9**];** //holds converted value in string

stdout **=** **&**mystdout**;** //set the output stream

init\_USART**();**

init\_ADC**();**

**while(**1**){**

ADCSRA **|=** **(**1 **<<** ADSC**);** //start the conversion. while in free running mode it will

**while((**ADCSRA**&(**1 **<<** ADIF**))==**0**);** //check if conversion done

ADCSRA **|=** **(**1 **<<** ADIF**);** //reset flag

ADCvalue **=** ADC **&** 0x03ff**;** //grab all 10 bits from ADC

ADCvalue **=** **((**ADCvalue **\*** 5**)/**1024**)\***100**;** //convert to degrees Fahrenheit

//itoa(ADCvalue, c, 10); //convert int to string

dtostrf**(**ADCvalue**,**3**,**1**,**c**);** //convert double to string

printf**(**"Temperature: "**);**

USART\_tx\_string**(**c**);** //print value to terminal

printf**(**" F\r\n"**);**

\_delay\_ms**(**1000**);** //wait 1s

**}**

**}**

void init\_USART**(){**

unsigned int BAUDrate**;**

//set BAUD rate: UBRR = [F\_CPU/(16\*BAUD)]-1

BAUDrate **=** **((**F\_CPU**/**16**)/**BAUD**)** **-** 1**;**

UBRR0H **=** **(**unsigned char**)** **(**BAUDrate **>>** 8**);** //shift top 8 bits into UBRR0H

UBRR0L **=** **(**unsigned char**)** BAUDrate**;** //shift rest of 8 bits into UBRR0L

UCSR0B **|=** **(**1 **<<** RXEN0**)** **|** **(**1 **<<** TXEN0**);** //enable receiver and trasmitter

// UCSR0B |= (1 << RXCIE0); //enable receiver interrupt

UCSR0C **|=** **(**1 **<<** UCSZ01**)** **|** **(**1 **<<** UCSZ00**);** //set data frame: 8 bit, 1 stop

**}**

void init\_ADC**(){**

ADMUX **=** 0**;** //use ADC0

ADMUX **|=** **(**1 **<<** REFS0**);** //use AVcc as the reference (5V)

//ADMUX |= (1 << ADLAR); //set to right adjust for 8-bit ADC

//ADCSRA |= (1 << ADIE); //ADC interrupt enable

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

//set pre-scale to 128 for input frequency

ADCSRA **|=** **(**1 **<<** ADPS2**)** **|** **(**1 **<<** ADPS1**)** **|** **(**1 **<<** ADPS0**);**

ADCSRB **=** 0**;** //free running mode

**}**

void USART\_tx\_string**(**char**\*** data**){**

**while((\***data**!=**'\0'**)){** //print until null

**while(!(**UCSR0A **&(**1**<<**UDRE0**)));** //check if transmit buffer is ready for new data

UDR0**=\***data**;** //print char at current pointer

data**++;** //iterate char pointer

**}**

**}**

static int put\_char**(**char c**,** FILE **\***stream**)**

**{**

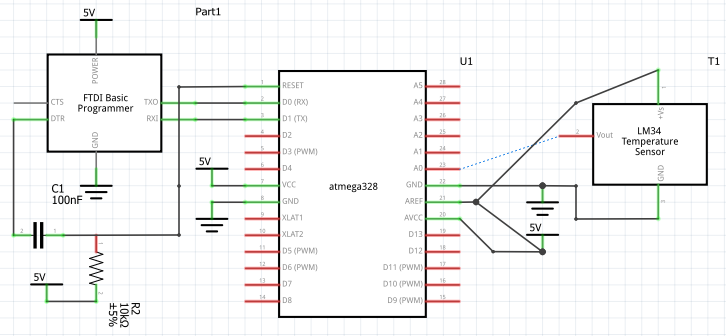
**while(!(**UCSR0A **&(**1**<<**UDRE0**)));** // wait for UDR to be clear

UDR0 **=** c**;** //send the character

**return** 0**;**

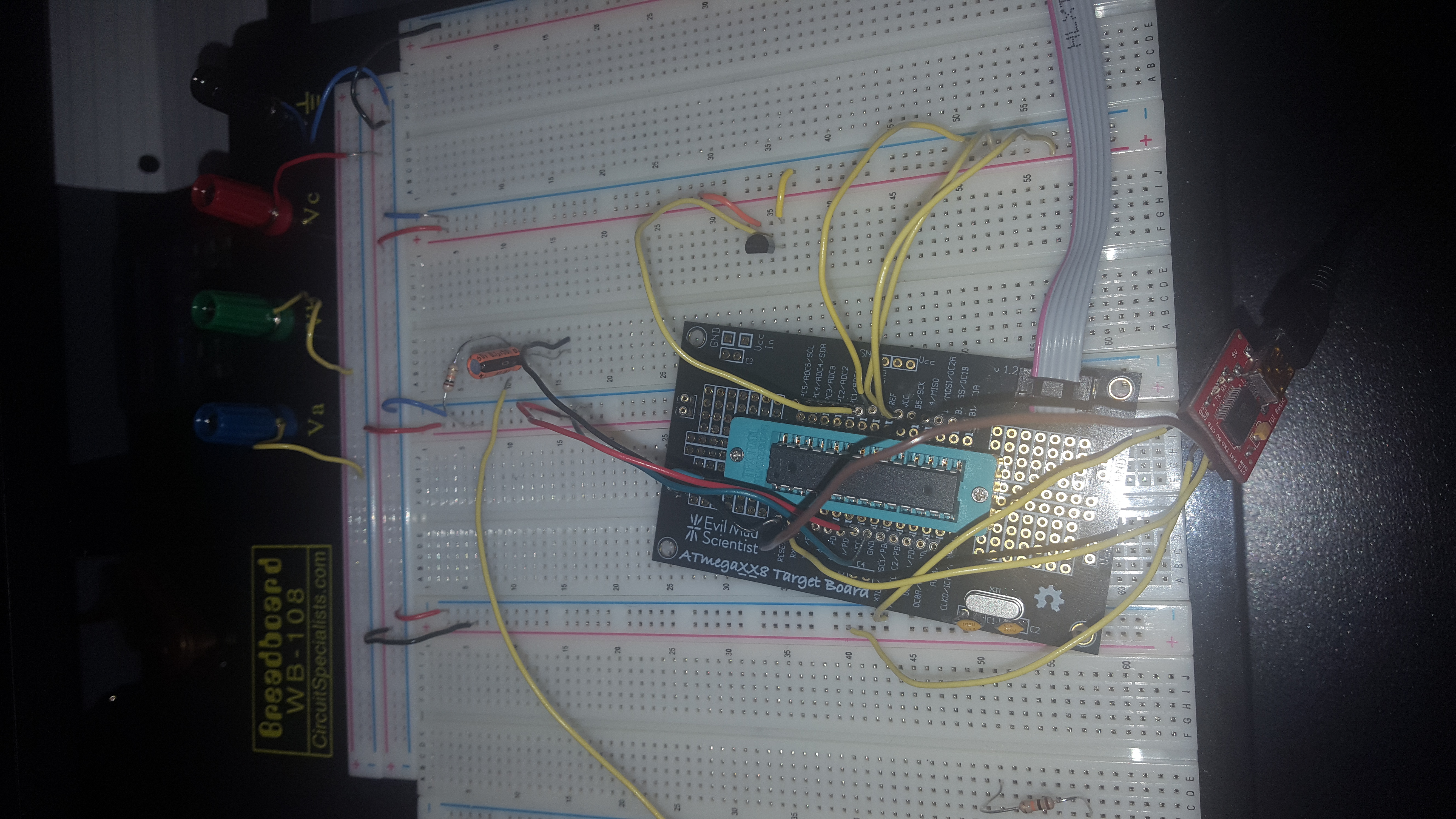
**}**

1. **SCHEMATICS**

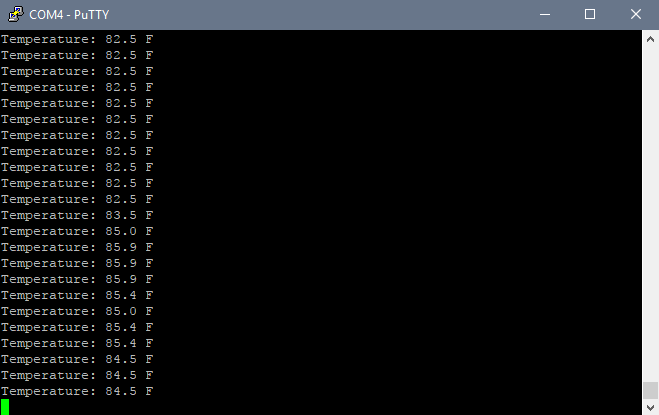


Task 1 C Board Schematic

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



Task 1 C Board Setup



Task 1 Terminal Output (Normal temp then increases as I hold it and decreases as I let go of the LM34).

1. **VIDEO LINKS OF EACH DEMO**

Task 1/C Code - <https://youtu.be/_7blyAvWj9g?list=PL5RuXbzEXwetPnCH9hMMV_BkhptJKt3xD>

1. **GITHUB LINK OF THIS DA**

<https://github.com/cisned2/DA3>

**Student Academic Misconduct Policy**

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

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