Damian Cisneros

CPE301 – SPRING 2018

Midterm

**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

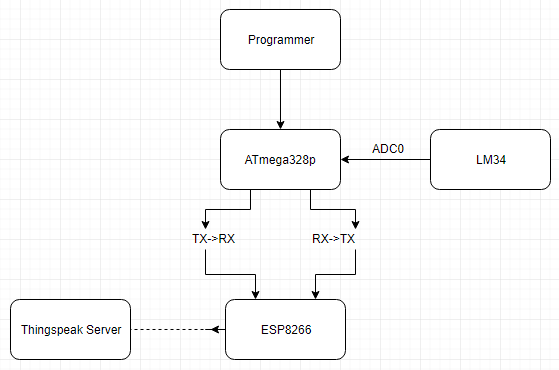
LM34

Breadboard

10kΩ resistor

100nF capacitor

Nodemcu with ESP8266



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

\*/

/\*

\* Midterm.c

\*

\* Created: 3/20/2018 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 <util/delay.h>

//Used to enable printf for use in USART

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 timer **=** 0**;** //send data every so often to Thingspeak based on this value

//AT Commands

char ATCWMODE**[]** **=** "AT+CWMODE=1\r\n"**;**

char ATCWJAP**[]** **=** "AT+CWJAP=\"TEST\",\"\"\r\n"**;**

char CIPMUX**[]** **=** "AT+CIPMUX=1\r\n"**;**

char CIPSTART**[]** **=** "AT+CIPSTART=0,\"TCP\",\"api.thingspeak.com\",80\r\n"**;**

char CIPSEND**[]** **=** "AT+CIPSEND=0,48\r\n"**;**

char SENDDATA**[]** **=** "GET /update?key=SFMSNLTJWNDH5Y6L&field1="**;**

char CIPCLOSE**[]** **=** "AT+CIPCLOSE\r\n"**;**

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();

\_delay\_ms(300);

printf("AT\r\n");

//USART\_tx\_string(AT); //test AT command

\_delay\_ms(2000);

USART\_tx\_string(ATCWMODE);//change to client mode

\_delay\_ms(2000);

USART\_tx\_string(ATCWJAP); //connect to wifi

\_delay\_ms(8000);

USART\_tx\_string(CIPMUX); //multiple connection set

\_delay\_ms(6000);

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

if(timer==3){

timer = 0; //reset timer

\_delay\_ms(1000);

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

USART\_tx\_string(CIPSTART);

\_delay\_ms(2000);

USART\_tx\_string(CIPSEND);

\_delay\_ms(4000);

USART\_tx\_string(SENDDATA);

USART\_tx\_string(c);

USART\_tx\_string("\r\n");

\_delay\_ms(2000);

USART\_tx\_string(CIPCLOSE);

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

}

timer++;

}

}

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)

{

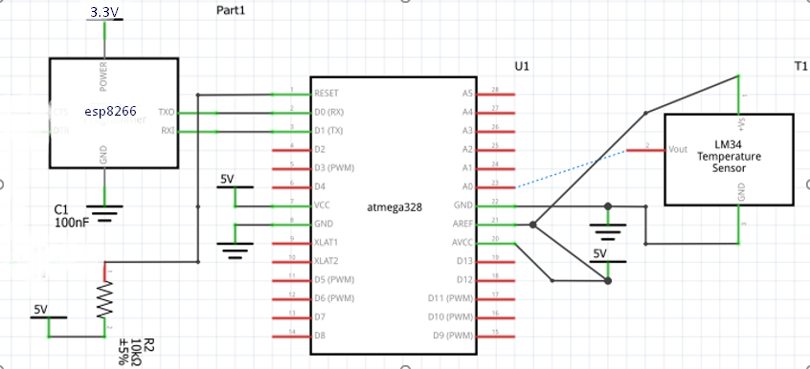
while(!(UCSR0A &(1<<UDRE0))); // wait for UDR to be clear

UDR0 = c; //send the character

return 0;

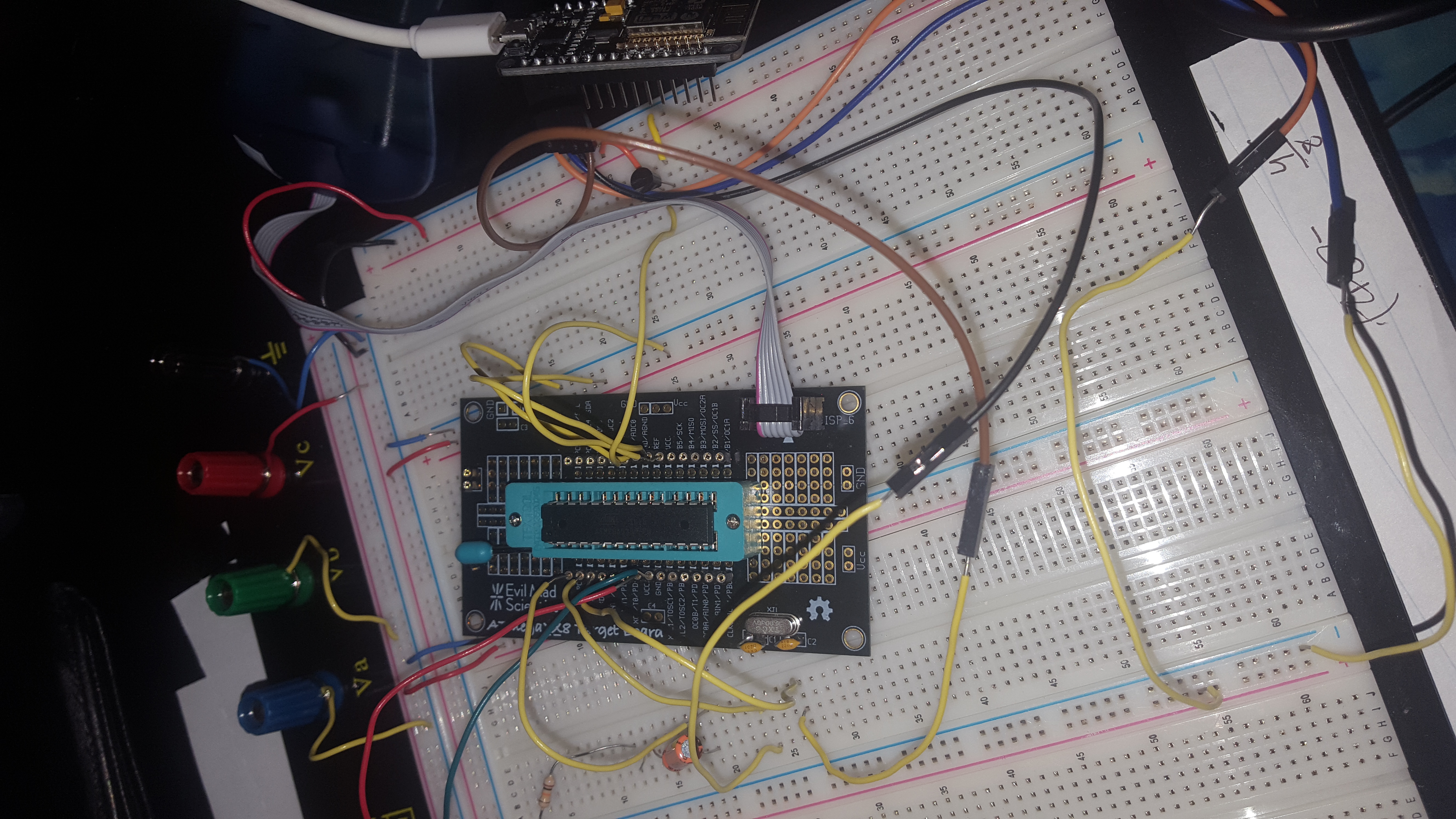
}

1. **SCHEMATICS**



Task 1 C Board Schematic

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



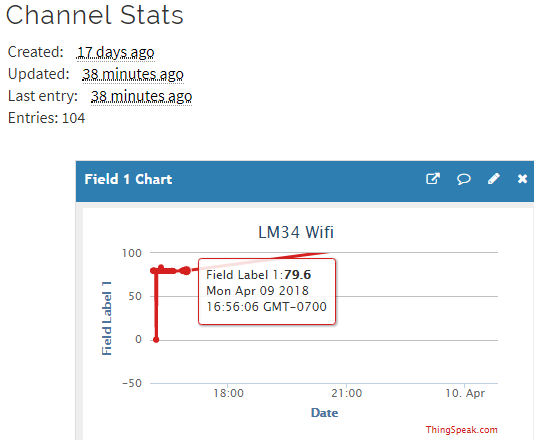
Task 1 C Board Setup



Task 1 Thingspeak data before code



Task 1 AT commands correctly running and sending data



Task 1 Thingspeak data after, heated at end to show value changes

1. **VIDEO LINKS OF EACH DEMO**

Task 1/C Code - <https://youtu.be/0e3gMBXVGo0>

1. **GITHUB LINK OF THIS DA**

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

**Student Academic Misconduct Policy**

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

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