CPE301 – SPRING 2019

Midterm 1

Student Name: Prachi Patel

Student #: 5002380222

Student Email: patelp3@unlv.nevada.edu

Primary Github address: <https://github.com/prachi173/da_sp18>

Directory: <https://github.com/prachi173/da_sp18/tree/master/Midterms>

Task:

Q: Write, simulate, and demonstrate using Atmel Studio 7 a C code for the AVR ATMEGA328p microcontroller that performs the following functions:

1. Program the ADC of ATmega328/p to read the LM34/35 temperature sensor.

2. Display the value to UART.

3. Make sure the AT Firmware is downloaded into the ESP-01/ESP32 module.

4. Register for a free Thingspeak account with MATHWORK. Setup and get the channel Key.

5. Transmit temperature sensor value to ESP-01/ESP32 through UART port using AT Commands.

6. Display the temperature sensor value as a graph in Thingspeak

Submission:

The following are required for successful completion of the design assignment:

a. AVR C code that has been compiled and working.

b. The C code should be well documented with explanation of every instruction.

c. A word document that contains the flow chart of the assembly code along with the screenshots/snapshot of the Atmel Studio 7 and/or live connections during debugging at the beginning and end of Task 1-6.

d. Submit one solution folder, with doc and video/snapshot file.

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

* ATMEGA328P
* LM35
* ESP8266 ESP01S

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

#ifndef F\_CPU

#define F\_CPU 8000000UL

#endif

#ifndef BAUD

#define BAUD 9600

#endif

#define ubrr (F\_CPU/(16UL\*BAUD))-1

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#include <stdio.h>

#include <stdlib.h>

//defining ADC0 as 0 to initialize

//using PC0 to read in temperature

#define ADC0 0x00

#define ADC1 0x01

#define AVCC 5000

volatile int OVFCount; //declare global overflow counter

//initializing and calling functions

void USART\_init(void);

void USART\_send(char \*data);

void ADC\_init(void);

uint16\_t read\_adc(uint8\_t adc\_temp);

char outs[256];

volatile unsigned int temp; //define unsigned 16 bit value for voltage reading from adc

volatile unsigned int t3; //define unsigned 8bit value for three digits

void main(void)

{

USART\_init(); //initialize the USART

ADC\_init(); //setup analog to digital converter

\_delay\_ms(500);

while(1){

/\* Initialize AT commands \*/

unsigned char AT[] = "AT\r\n";

//Set device mode to Station mode

unsigned char AT\_CWMODE[] = "AT+CWMODE=1\r\n";

//Wifi connection, (replaced mine with SSID and Password)

unsigned char AT\_CWJAP[] = "AT+CWJAP=\"ssid\",\"password\"\r\n";

//single IP Address Mode

unsigned char AT\_CIPMUX[] = "AT+CIPMUX=0\r\n";

//Start TCP connection to Thingspeak.com at port 80

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

//string length

unsigned char AT\_CIPSEND[] = "AT+CIPSEND=100\r\n";

//send to USART

\_delay\_ms(200);

USART\_send(AT);

\_delay\_ms(1000);

USART\_send(AT\_CWMODE);

\_delay\_ms(5000);

USART\_send(AT\_CWJAP);

\_delay\_ms(5000);

USART\_send(AT\_CIPMUX);

\_delay\_ms(5000);

USART\_send(AT\_CIPSTART);

\_delay\_ms(5000);

USART\_send(AT\_CIPSEND);

\_delay\_ms(5000);

temp = read\_adc(ADC0); //set temp as ADC value read from read\_adc function

temp = ((temp\*500)/1024); //convert temp using prescaler of 1024

t3 = (temp\*1.8)+32;

//transfer from ASCII to decimal value and send to print

//

snprintf(outs,sizeof(outs), "GET https://api.thingspeak.com/update?api\_key=WI421PP7666NY2VZ&field1=0 \r\n", t3);

USART\_send(outs);//send data

\_delay\_ms(5000);

}

}

//commented out from previous code since it's not necessary

//USART\_send(t1+48);

//USART\_send(t2+48);

//USART\_send(t3+48);

//USART\_send('C');

//USART\_send('\n');

//USART\_send('\r');

//delay for a second

void USART\_init(void){

//initialize the USART FTDI Basic

//Set Baud Rate and frame format 8N1

//enable receiver and transmitter

UBRR0H = (uint8\_t)(ubrr >> 8);

UBRR0L = (uint8\_t)ubrr;

UCSR0B = (1<<RXEN0) | (1<< TXEN0);

UCSR0C |= (1<<UCSZ00) | (1<<UCSZ01);

}

void ADC\_init(void)

{

ADMUX |= (1<<REFS0); //AVcc - external cap at AREF

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

//ADC prescaler and ADC enable

}

uint16\_t read\_adc(uint8\_t adc\_temp){

ADMUX = (ADMUX & 0xF0) | (adc\_temp & 0x0F);

ADCSRA |= (1<<ADSC); //start conversion

while(ADCSRA & (1<<ADSC)); //wait for conversion to finish

return ADC; //return the value in ADC to where the function is called (in temp)

}

void USART\_send(char \*data ) {

while ((\*data != '\0')) {

while (!(UCSR0A & (1 <<UDRE0)));

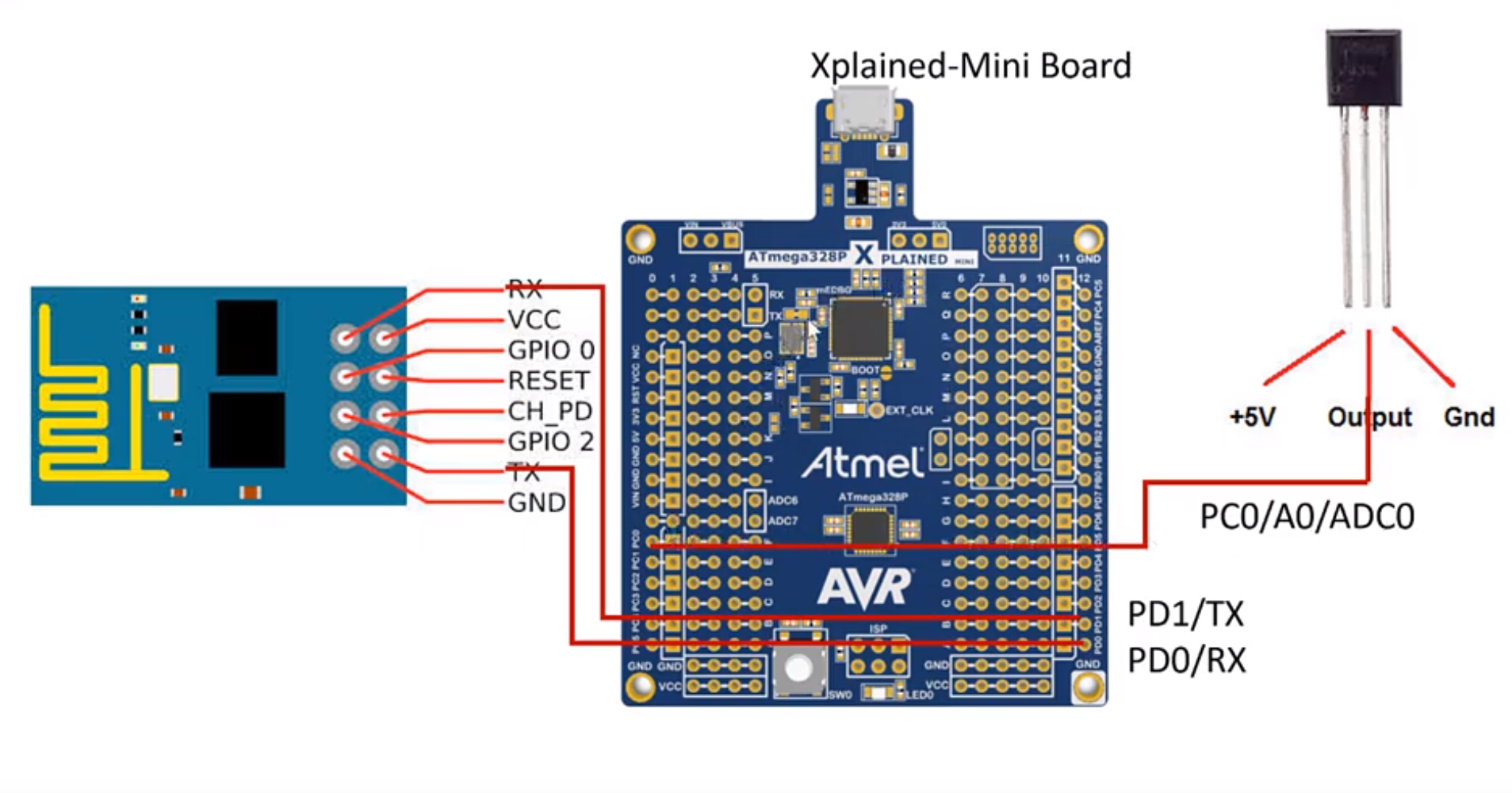
UDR0 = \*data;

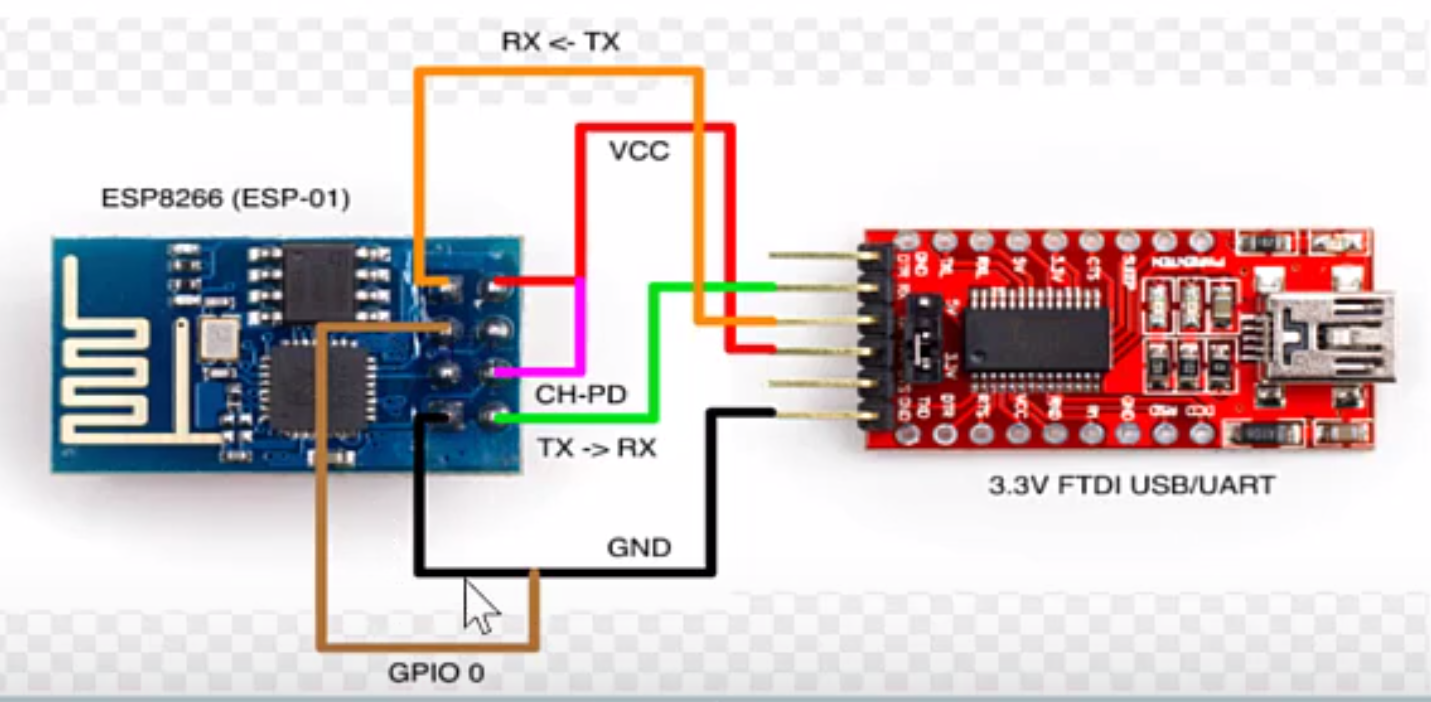
data++;

}

}

1. **SCHEMATICS**





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

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

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

**A picture containing indoor, wall

Description automatically generated**

1. **VIDEO LINKS OF EACH DEMO**

NA – The AT didn’t connect.

1. **GITHUB LINK OF THIS DA**

<https://github.com/prachi173/da_sp18/tree/master/Midterms>

**Student Academic Misconduct Policy**

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

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

Prachi Patel