CPE301 – FALL 2019

Midterm 1

Student Name: Cameron Kirk

Student #: 2000660135

Student Email: kirkc1@unlv.nevada.edu

Primary Github address: <https://github.com/kirkster96/submission_da>

Directory: <https://github.com/kirkster96/submission_da/tree/master/Midterms/Midterm1>

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

Atmega328PB

ESP-01

Hilelgo Multi-Function Shield Module for Arduino

1. **AVR C DEVELOPED CODE OF TASK 1**

/\*\*

\* \DA3\_a Cameron Kirk

\*

\* \This is a AVR C program that will use the LM35 to print temperatur to the terminal

\*

\*/

#define *F\_CPU* 16000000UL

#define BAUD\_RATE 9600

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

#define TRUE 1

#define FALSE 0

#define CHAR\_NEWLINE '\n'

#define CHAR\_RETURN '\r'

#define RETURN\_NEWLINE "\r\n"

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

#include <stdlib.h>

#include <string.h>

#include <stdio.h>

void usart\_init ();

void usart\_send (char ch);

void UART\_sendString(const char \*str);

void check\_OK();

volatile unsigned char data\_in[8];

volatile unsigned char command\_in[8];

volatile unsigned char data\_count;

volatile unsigned char command\_ready;

int main (void)

{

//initialize the usart

usart\_init ();

ADMUX = (0<<REFS1)| //Reference selection bits

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

(0<<ADLAR)| //Left adjust most significant bit

(1<<MUX2)| //Analog channel selection bits

(0<<MUX1)| //ADC4 (PC4 PIN27)

(0<<MUX0);

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

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

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

(0<<ADIF)| //ADC Interrupt Flag

(0<<ADIE)| //ADC Interrupt Enable

(1<<ADPS2)| //ADC prescaler bits

(0<<ADPS1)|

(1<<ADPS0);

UART\_sendString("AT\r\n");

*\_delay\_ms*(3000);

check\_OK();

// Select WIFI mode

UART\_sendString("AT+CWMODE=1\r\n");

*\_delay\_ms*(3000);

// Connect to local WIFI

UART\_sendString("AT+CWJAP=\"\_-\_-\",\"\_-\_-\_\"\r\n");

*\_delay\_ms*(3000);

// Enable connection

UART\_sendString("AT+CIPMUX=0\r\n");

*\_delay\_ms*(3000);

volatile unsigned char TEMP [4];

while (1){

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

while((ADCSRA&(1<<ADIF))==0); //wait for conversion to complete

ADCSRA |= (1<<ADIF);

int c = ADCL;

c = c | (ADCH<<8);

c = (c/1024.0)\*5000/10;

*itoa*(c,TEMP,10);

// Start a connection as client to Thingspeak

UART\_sendString("AT+CIPSTART=\"TCP\",\"184.106.153.149\",80\r\n");

*\_delay\_ms*(2000);

// Specify the size of the data

UART\_sendString("AT+CIPSEND=45\r\n");

*\_delay\_ms*(2000);

// Send temperature

UART\_sendString("GET /update?key=2NB8QCW9WK5PJNS9&field1=");

*\_delay\_ms*(2000);

UART\_sendString(TEMP);

*\_delay\_ms*(2000);

UART\_sendString("\r\n\r\n");

*\_delay\_ms*(15000);

}

return 0;

}

void usart\_init(void){

//turn on Rx, TX

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

//8bit char sizes

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

//set baud rate

UBRR0H = (BAUD\_PRESCALE >> 8);

UBRR0L = BAUD\_PRESCALE;

//Enable the USART Receive interrupt

UCSR0B |= (1<<RXCIE0);

//globally enable interrupts

//sei();

}

void usart\_send(char ch){

while(! (UCSR0A & (1<<UDRE0))); //wait until UDR0 is empty

UDR0 = ch; //transmit ch

}

void check\_OK(){

volatile unsigned char i;

volatile unsigned char result[2];

int k = 0;

/\*

// Wait for the response to finish

while(result != "OK"){

while(!(UCSR0A & (1<<RXC0)));

i = UDR0;

usart\_send(i);

result[(k%2)]=i;

k++;

}

\*/

}

void UART\_sendString(const char \*str){

while(\*str){

usart\_send(\*str++);

}

}

1. **VIDEO LINKS OF EACH DEMO**

AVR C Task 1 Demo

<https://drive.google.com/file/d/1EWtPwc__oKJmCTWCSmY1LDAjTSGODB1q/view?usp=sharing>

1. **GITHUB LINK OF THIS DA**

<https://github.com/kirkster96/submission_da/tree/master/Midterms/Midterm1>

**Student Academic Misconduct Policy**

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

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

Cameron Kirk