Bryan Takemoto

CPE301 – SPRING 2018

Design Assignment Mid-Term

**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 |  |  |
| 2. | C CODE |  |  |
| 3. | FLOWCHART OF THE CODE |  |  |
| 4. | THINGSPEAK GRAPH OF THE TEMPERATURE VIA UART [ESP8266] |  |  |
| 5. | VIDEO LINK EXPLAINING THE BREADBOARD AND OPERATION |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. **COMPONENTS LIST**

Essential components used: Atmega328, LM34, and ESP8266.

1. **C CODE**

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <stdint.h>

#include <avr/interrupt.h>

#include <util/delay.h>

#include <stdlib.h>

#define FOSC 16000000 // Clock speed

#define BAUD 115200 // Desire baud rate

#define MYUBRR FOSC/8/BAUD-1 // Formula to set the baud rate

volatile *uint8\_t* ADCvalue; // Storage for the temperature in F

volatile unsigned char TEMP[5]; // ASCII temp value

// AT Commands

volatile unsigned char CWMODE[] = "AT+CWMODE=3\r\n";

volatile unsigned char WIFI[] = "AT+CWJAP=\"WIFI\",\"PASSWORD\"\r\n";

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

volatile unsigned char FIRM[] = "AT+GMR\r\n";

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

volatile unsigned char CIPSTART[] = "AT+CIPSTART=\"TCP\",\"184.106.153.149\",80\r\n";

volatile unsigned char SIZE[] = "AT+CIPSEND=45\r\n";

volatile unsigned char SEND\_DATA[] = "GET /update?key=54MRLC7ZQ32UD48T&field1=";

volatile unsigned char END[] = "\r\n\r\n";

void send\_AT(volatile unsigned char AT[]);

int main(void)

{

// Port declarations

DDRB = (1<<5); // Set PORTB.5 as output

PORTB = 0x00; // Clear PORTB

// ADC declaration

ADMUX = 0; // Use ADC0

ADMUX |= (1<<ADLAR); // Left justified

ADMUX |= (1<<REFS0); // AVcc is reference

ADCSRA |= (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); // 16 MHz with prescalar of 128

ADCSRA |= (1<<ADATE); // Enable auto trigger

ADCSRB = 0; // Free running settings for auto trigger

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

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

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

// USART declaration

UBRR0H = ((MYUBRR)>>8); // Set baud rate for UPPER Register

UBRR0L = MYUBRR; // Set baud rate for LOWER Register

UCSR0A |= (1<<U2X0); // Double UART transmission speed

UCSR0B |= (1<<TXEN0); // Enable transmitter

UCSR0C |= (1<<UCSZ01) | (1<<UCSZ00); // Frame: 8-bit Data and 1 Stop bit

// ESP8266 settings

*\_delay\_ms*(1000);

send\_AT(AT);

*\_delay\_ms*(2000); // Display firmware

send\_AT(FIRM);

*\_delay\_ms*(2000); // Select WIFI mode

send\_AT(CWMODE);

*\_delay\_ms*(2000); // Connect to local WIFI

send\_AT(WIFI);

*\_delay\_ms*(10000); // Enable connection

send\_AT(CIPMUX);

sei();

// Send temperature to Thingspeak server every 30 seconds

while (1)

{

*\_delay\_ms*(500); // Start a connection as client to Thingspeak

send\_AT(CIPSTART);

*\_delay\_ms*(500); // Specify the size of the data

send\_AT(SIZE);

*\_delay\_ms*(1000); // Send temperature

send\_AT(SEND\_DATA);

send\_AT(TEMP);

send\_AT(END);

*\_delay\_ms*(28000);

}

return 0;

}

// Interrupt subroutine for ADC value

ISR(ADC\_vect)

{

unsigned char i = 0x00;

char temperature[5];

ADCvalue = (ADCH<<1); // Store the decimal value of the converted signal

*itoa*(ADCvalue, temperature, 10);

for(i = 0x00; i < 5; i++)

{

TEMP[i] = temperature[i];

}

}

void send\_AT(volatile unsigned char AT[])

{

volatile unsigned char len = 0;

volatile unsigned char i;

while(AT[len] != 0)

{

len++;

}

for(i = 0x00; i < len; i++){

while(!(UCSR0A & (1<<UDRE0))); // Wait for the transmitter to finish

UDR0 = AT[i]; // Transmit the the new value

}

}

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

Insert only the modified sections here. Use more sections if needed

1. **SCHEMATICS**

Use fritzing.org

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**
2. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**
3. **VIDEO LINKS OF EACH DEMO**
4. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

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

Bryan Takemoto