CPE301 – SPRING 2019

MIDTERM 2

Student Name: Daniel Senda
Student #: 5002362633
Student Email: sendad1@unlv.nevada.edu
Primary Github address: https://github.com/dsenda/Smiles
Directory: Midterm2

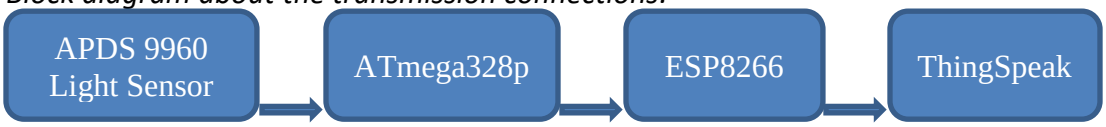
Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

List of Components used:
Atmel Studio 7
ATmega328P Xplained Mini
FTDI Basic chip
ESP8266 ESP-01 Serial Wireless WiFi Transceiver Receiver Module
OPEN SMART USB to ESP8266 ESP 01 Wi Fi Adapter Module
APDS 9960 sensor (light sensor).
Breadboard
Jumper wires

Block diagram about the transmission connections:



2. INITIAL CODE OF TASK FOR MIDTERM II

Not applicable, there was no initial code for midterm II assignment.

3. DEVELOPED/MODIFIED CODE FOR MIDTERM1

The following is the code that accomplishes the required tasks for the Midterm II assignment.

```
// midterm2_c_code.c
// Daniel Senda

// Definitions
#define F_CPU 16000000UL
#define BAUD 9600
#define FOSC 16000000
#define UBRREQ FOSC/16/BAUD -1
#define APDS9960_WRITE 0x72
#define APDS9960_READ 0x73

// Libraries
#include <avr/io.h>
#include <stdio.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdint.h>
#include <stdlib.h>
#include "i2c_master.h"
#include "SparkFun_APDS9960.h"

// Functions
void usart_init(void);
void at_wifi_cmd(int cmd);
void light_sensor_cmd(int cmd1);
```

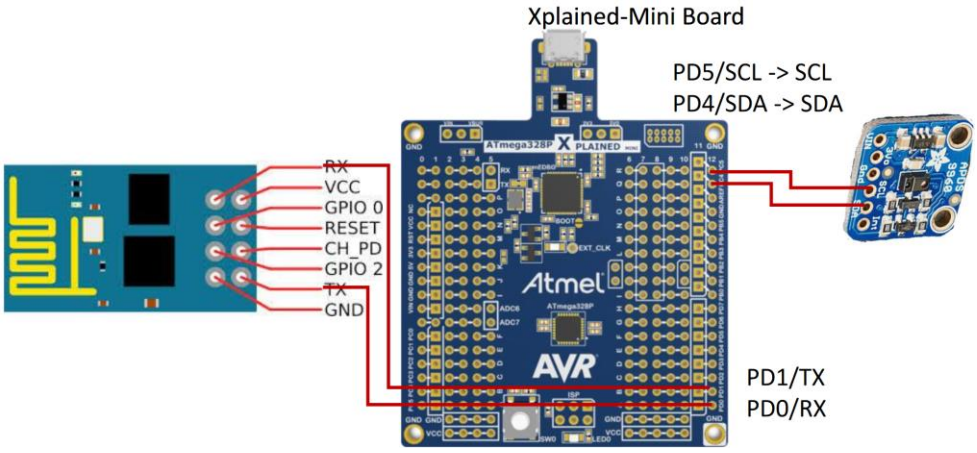
```
int usart_send(char ch, FILE *stream);
FILE str_uart = FDEV_SETUP_STREAM(usart_send, NULL , _FDEV_SETUP_WRITE);

// Variables
uint8_t red_high, red_low;         // Variables used to capture raw light data.
uint8_t green_high, green_low;
uint8_t blue_high, blue_low;
uint16_t red_light = 0;            // Variables used to store processed light data.
uint16_t green_light = 0;
uint16_t blue_light = 0;

int main(void)
{
       usart_init();               // Initializes USART.
       i2c_init();                 // Initializes I2C communication.
       light_sensor_cmd(1);        // Initializes light sensor (APDS9960).
       stdout = &str_uart;
       at_wifi_cmd(1);             // Initializes WiFi module.

       while(1)
       {
              light_sensor_cmd(2); // Reads data from light sensor.
              at_wifi_cmd(2);      // Sends processed light data to ThingSpeak channel.
       }
}

void usart_init(void)       // Initializes USART.
{
       // Sets baud rate.
       uint16_t baud_rate = UBRREQ;
       UBRR0H = baud_rate >> 8;
       UBRR0L = baud_rate & 0xFF;

       // Enables receiver and transmitter.
       UCSR0B = ( 1 <<RXEN0)|( 1 <<TXEN0);

       // Sets frame format: 8data, 1stop bit.
       UCSR0C = (3 <<UCSZ00);
}

int usart_send(char ch, FILE *stream)
{
       // Waits until buffer empty.
       while ( !( UCSR0A & ( 1 <<UDRE0)) );

       // Puts data into buffer.
       UDR0 = ch;
       return 0;
}

void at_wifi_cmd (int cmd)
{
       if (cmd == 1) // Initializes WiFi module.
       {
              // Checks AT connect.
              _delay_ms(2000);
              printf("AT\r\n");
```

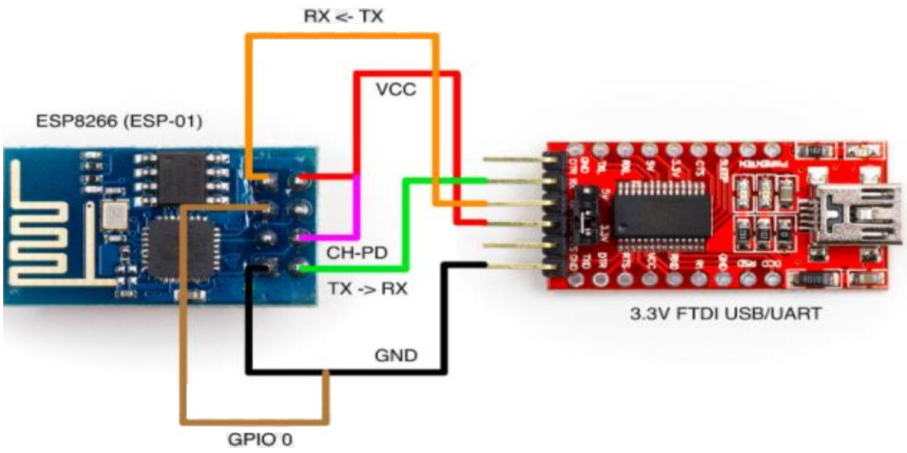
```
                // Sets device mode, 1 = Station mode.
                _delay_ms(5000);
                printf("AT+CWMODE=3\r\n");

                // Connects to WiFi using SSID and Password.
                _delay_ms(5000);
                printf("AT+CWJAP=\"{SSID}\",\"{PASSWORD}\"\r\n");
        }

        if (cmd == 2) // Sends data to ThingSpeak channel.
        {
                // Sets device for single IP Address Mode.
                _delay_ms(3000);
                printf("AT+CIPMUX=0\r\n");

                // Starts TCP connection to Thingspeak.com at port 80.
                _delay_ms(3000);
                printf("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n");

                // Gives upcoming string length.
                _delay_ms(3000);
                printf("AT+CIPSEND=110\r\n");

                // Send processed light data to field 1 (red_light), field 2 (green_light),
                and field 3 (red_light).
                _delay_ms(3000);
                printf("GET
https://api.thingspeak.com/update?api_key={INSERT_API_KEY}&field1=0%05u&field2=%05u&field
3=%05u\r\n", red_light, green_light, blue_light);
        }
}

void light_sensor_cmd(int cmd1)
{
        if (cmd1 == 1)       // Initializes light sensor (APDS9960).
        {
                uint8_t setup;

                i2c_readReg(APDS9960_WRITE, APDS9960_ID, &setup,1);
                if(setup != APDS9960_ID_1) while(1);
                setup = 1 << 1 | 1<<0 | 1<<3 | 1<<4;

                i2c_writeReg(APDS9960_WRITE, APDS9960_ENABLE, &setup, 1);
                setup = DEFAULT_ATIME;

                i2c_writeReg(APDS9960_WRITE, APDS9960_ATIME, &setup, 1);
                setup = DEFAULT_WTIME;

                i2c_writeReg(APDS9960_WRITE, APDS9960_WTIME, &setup, 1);
                setup = DEFAULT_PROX_PPULSE;

                i2c_writeReg(APDS9960_WRITE, APDS9960_PPULSE, &setup, 1);
                setup = DEFAULT_POFFSET_UR;

                i2c_writeReg(APDS9960_WRITE, APDS9960_POFFSET_UR, &setup, 1);
                setup = DEFAULT_POFFSET_DL;

                i2c_writeReg(APDS9960_WRITE, APDS9960_POFFSET_DL, &setup, 1);
```

```
            setup = DEFAULT_CONFIG1;

            i2c_writeReg(APDS9960_WRITE, APDS9960_CONFIG1, &setup, 1);
            setup = DEFAULT_PERS;

            i2c_writeReg(APDS9960_WRITE, APDS9960_PERS, &setup, 1);
            setup = DEFAULT_CONFIG2;

            i2c_writeReg(APDS9960_WRITE, APDS9960_CONFIG2, &setup, 1);
            setup = DEFAULT_CONFIG3;

            i2c_writeReg(APDS9960_WRITE, APDS9960_CONFIG3, &setup, 1);
        }

        if (cmd1 == 2)       // Reads data from light sensor.
        {
            // Reads red_light data.
            i2c_readReg(APDS9960_WRITE, APDS9960_RDATAH, &red_high, 1);
            i2c_readReg(APDS9960_WRITE, APDS9960_RDATAL, &red_low, 1);

            // Reads green_light data.
            i2c_readReg(APDS9960_WRITE, APDS9960_GDATAH, &green_high, 1);
            i2c_readReg(APDS9960_WRITE, APDS9960_GDATAL, &green_low, 1);

            // Reads blue_light data.
            i2c_readReg(APDS9960_WRITE, APDS9960_BDATAH, &blue_high, 1);
            i2c_readReg(APDS9960_WRITE, APDS9960_BDATAL, &blue_low, 1);

            // Stores processed data.
            red_light = (red_high << 8) | red_low;
            green_light = (green_high << 8) | green_low;
            blue_light = (blue_high << 8) | blue_low;

            // Sets Max values to prevent spikes in data.
            if (red_light > 255){red_light = 255;}
            if (green_light > 255){green_light = 255;}
            if (blue_light > 255){blue_light = 255;}
        }
}
```

4. SCHEMATICS

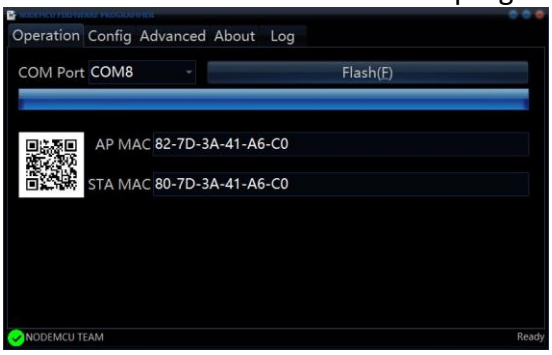
The following schematic from the slides describe the connections that are made relatively well.



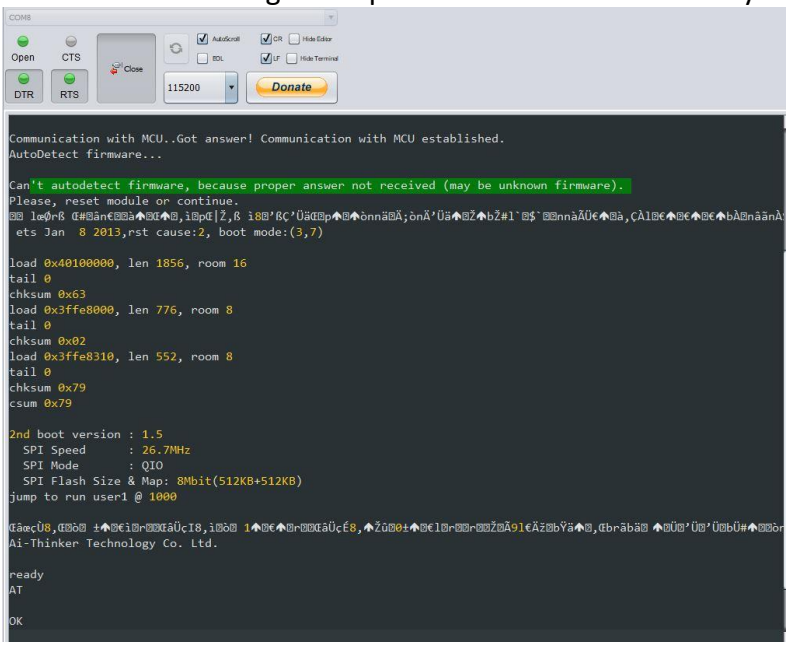


5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

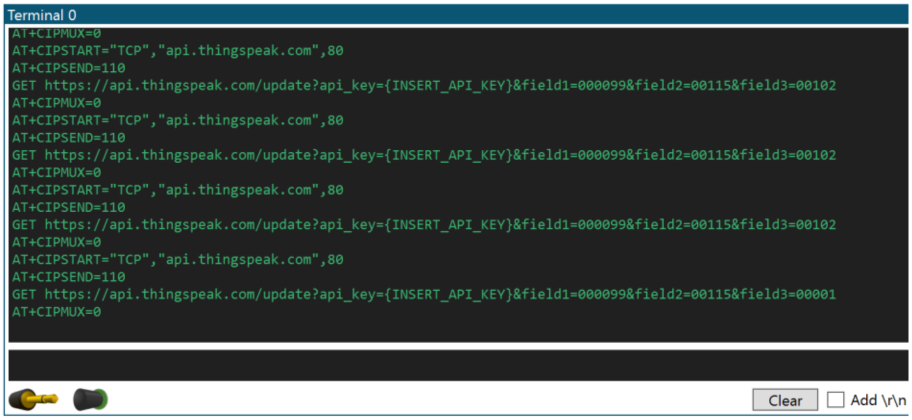
The following is a screenshot shows that the ESP module was programmed with the firmware.



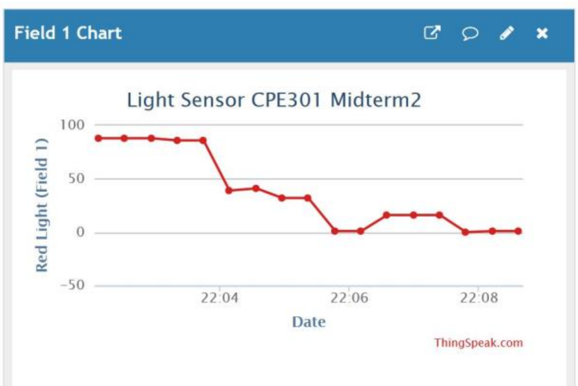
The WiFi module was then tested using the ESplorer and it worked correctly.

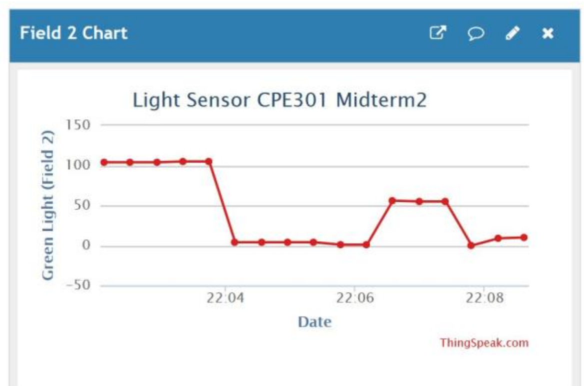


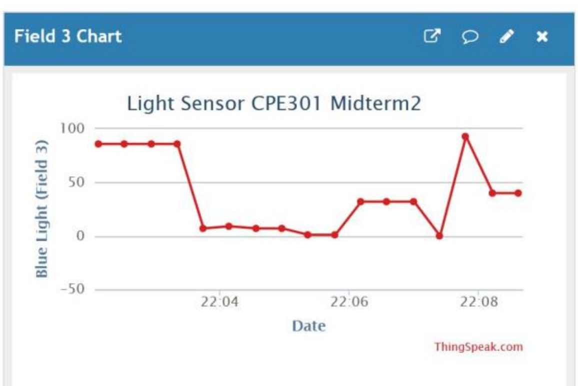
The following is a screenshot of the output on the terminal that shows what is being sent to the WIFI module and ThingSpeak.



After everything was connected and the program was running, the following graph was created on ThingSpeak using the supplied data from the light sensor. Field 1 is the red light data, field 2 is the green light data, and field 3 is the blue light data.

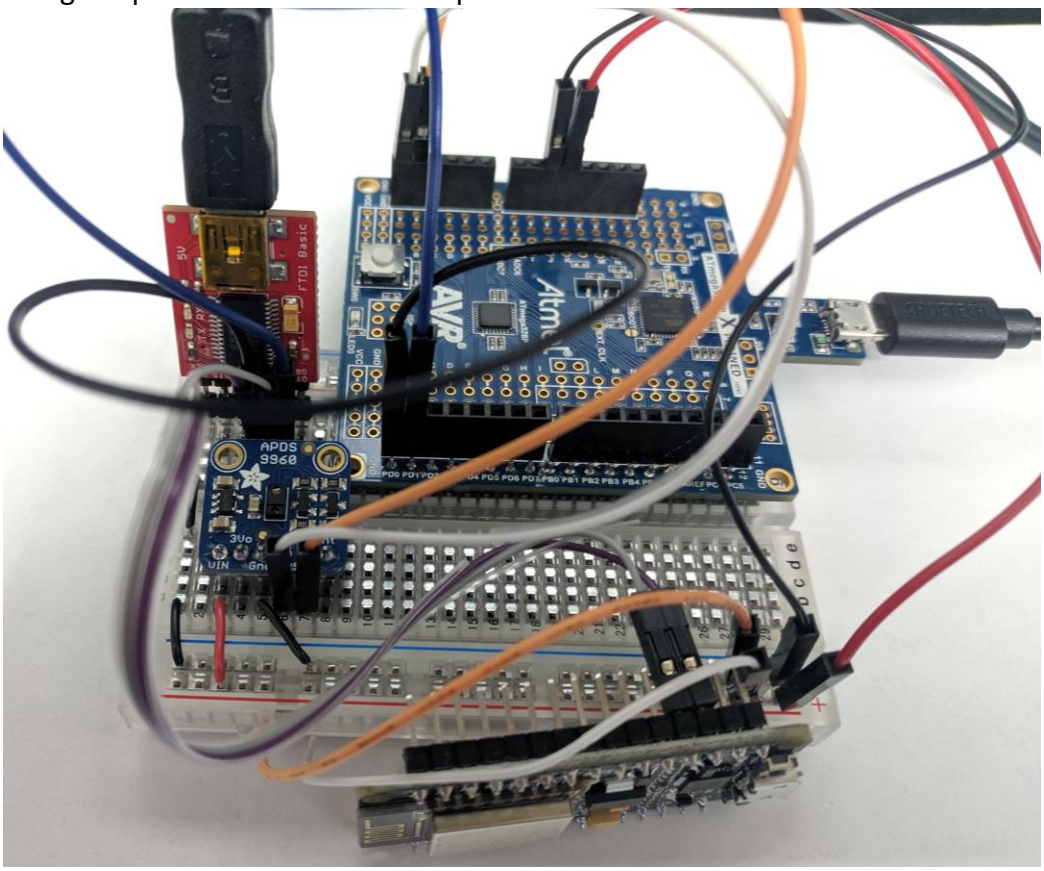






6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

The following is a picture of the board setup:



7. VIDEO LINKS OF EACH DEMO

https://youtu.be/FZFqvjO_EVw

8. GITHUB LINK OF THIS DA

https://github.com/dsenda/Smiles/tree/master/Midterm2

Student Academic Misconduct Policy

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

"*This assignment submission is my own, original work*".

Daniel Senda