CPE301 – SPRING 2019

MIDTERM 2

Student Name: Geovanni Portillo
Student #: 8000603824
Student Email: portig1@gmail.com
Primary Github address: https://github.com/portig1/submissions_E
Directory: portig1/submissions_E/tree/master/Midterms/Midterm_2

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/Midterm, 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

Atmel Studio 7
ATmega328PB Xplained mini

Figure 1-1. ATmega328P Xplained Mini Headers and Connectors

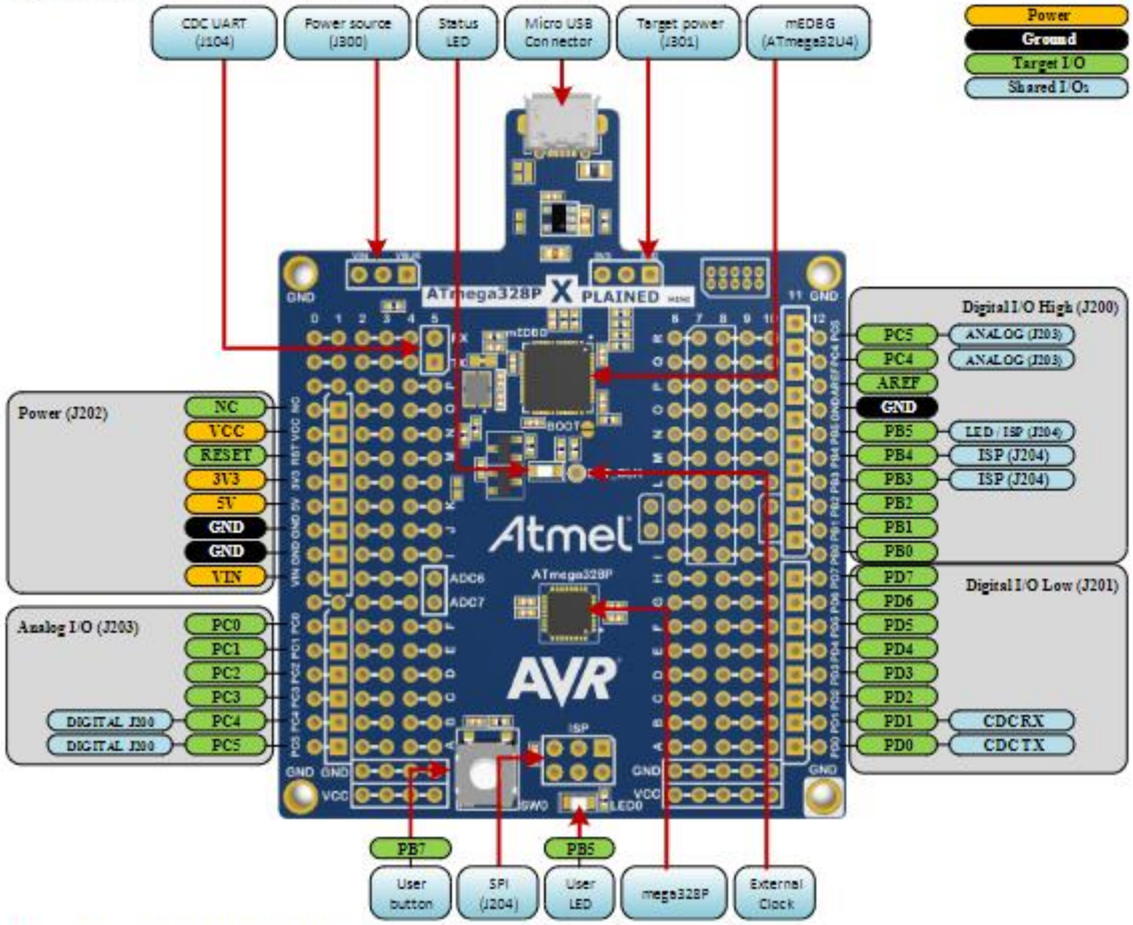
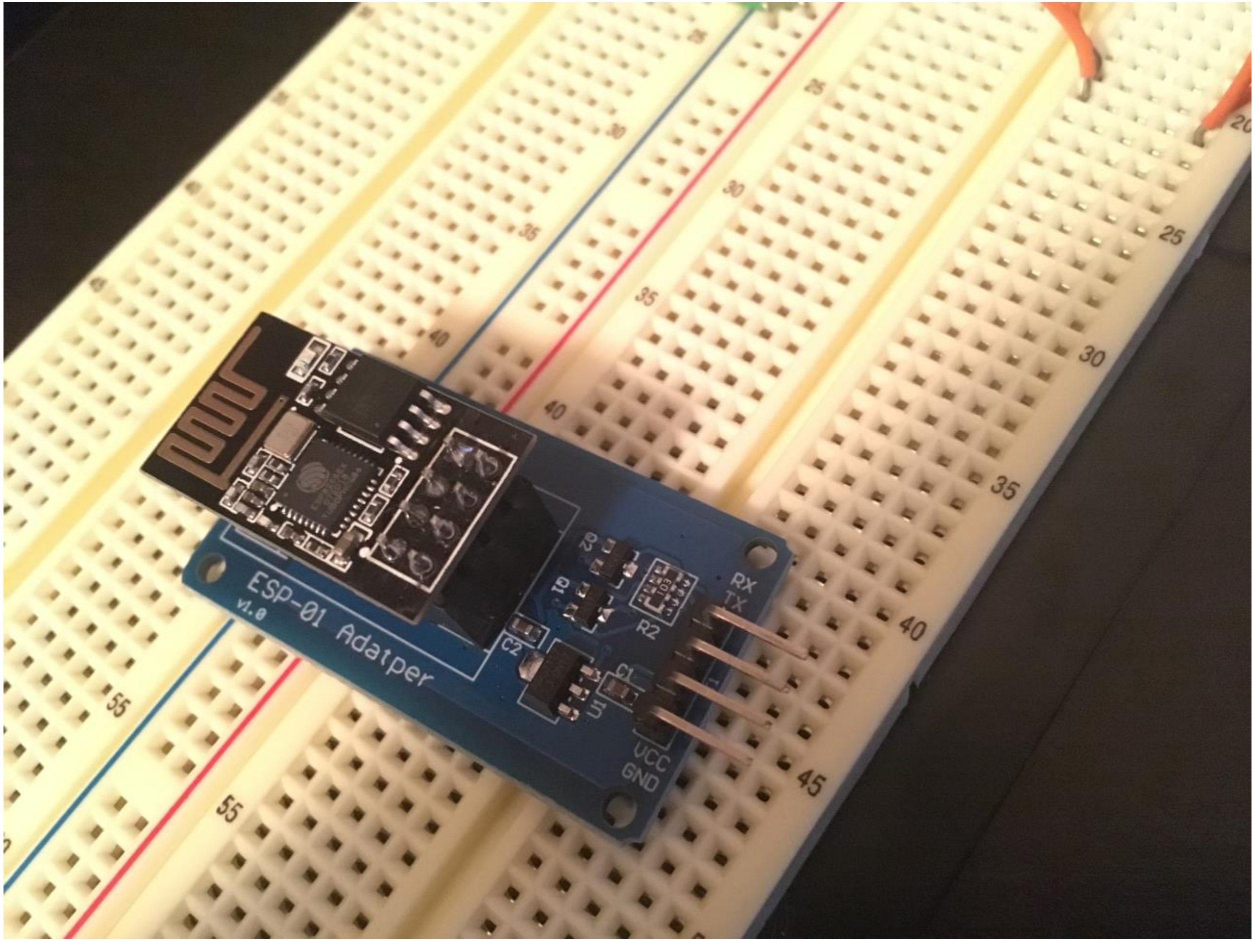
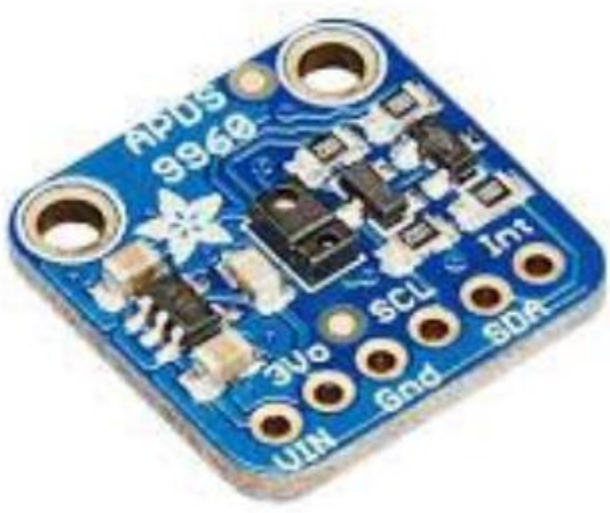


Table 1-1. Default Configurations

ESP-01 to USB Programmer/Adapter



ESP-01 Adapter with ESP8266 plugged in



APDS9960 with pin labels

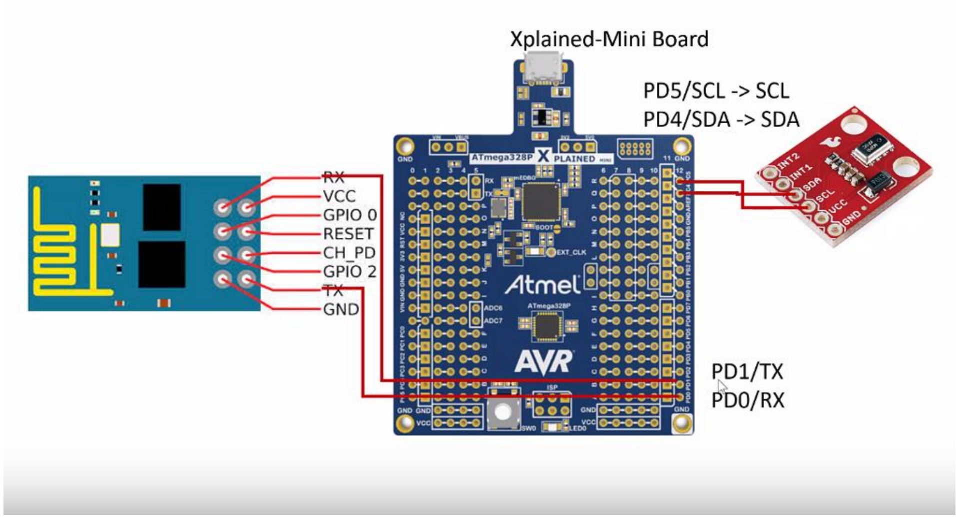
2. DEVELOPED CODE OF TASK 1/2/5/6

```
#ifndef F_CPU
```

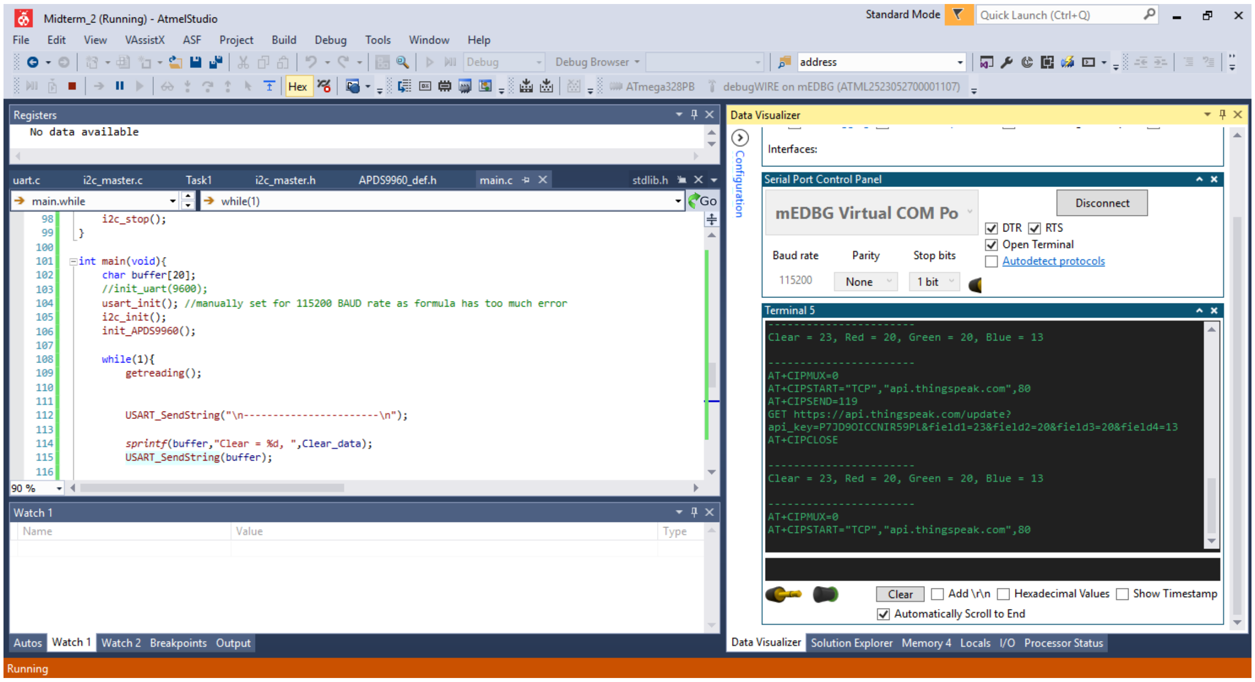
```
#define F_CPU 16000000UL
#endif

#include <avr/io.h>
#include <util/delay.h>
#include <math.h>
#include <stdlib.h>                                                                /*
Include standard library file */
#include <stdio.h>                                                                 /*
Include standard library file */
#include "APDS9960_def.h"                                            /* Include APDS9960
register define file */
#include "i2c_master.h"                                              /* Include I2C
Master header file */
#include "uart.h"                                             /* Include USART header
file */

#define APDS9960_WRITE 0x72
#define APDS9960_READ 0x73

uint16_t Clear_data, Red_data, Green_data, Blue_data;


void init_uart(uint16_t baudrate){

       uint16_t UBRR_val = (F_CPU/16)/(baudrate-1);

       UBRR0H = UBRR_val >> 8;
       UBRR0L = UBRR_val;

       UCSR0B |= (1<<TXEN0) | (1<<RXEN0) | (1<<RXCIE0); // UART TX (Transmit - senden)
einschalten
       UCSR0C |= (1<<USBS0) | (3<<UCSZ00); //Modus Asynchron 8N1 (8 Datenbits, No Parity,
1 Stopbit)
}


void usart_init (void)
{
       //UBRR0H = (uint8_t)(BAUD_PRESCALLER >> 8);
       //UBRR0L = (uint8_t)(BAUD_PRESCALLER);
       UBRR0 = 8;    //Manually setting as the formula give a float value that has too
much error.
       //UCSR0A = (1 << U2X0);
       UCSR0B = (1 << RXEN0) | (1 << TXEN0);
       UCSR0C = (1<< UCSZ01)|(1<<UCSZ00);
}

void uart_putc(unsigned char c){

       while(!(UCSR0A & (1<<UDRE0))); // wait until sending is possible
       UDR0 = c; // output character saved in c
}

void uart_puts(char *s){
       while(*s){
              uart_putc(*s);
              s++;
       }
}

void init_APDS9960(void){
       _delay_ms(150);
       /* Power up time >100ms */
       i2c_start(APDS9960_WRITE); //
       i2c_write(APDS9960_ENABLE); //
       i2c_write(0x00); //Turn off all features
       i2c_stop();

       i2c_start(APDS9960_WRITE);
       i2c_write(APDS9960_ATIME);
       i2c_write(DEFAULT_ATIME); // Set default integration time
       i2c_stop();

       i2c_start(APDS9960_WRITE);
       i2c_write(APDS9960_CONTROL); //Set default gain value
       i2c_write(DEFAULT_AGAIN);
       i2c_stop();

       i2c_start(APDS9960_WRITE);
       i2c_write(APDS9960_ENABLE);
       i2c_write((1 << POWER) | (1 << AMBIENT_LIGHT));        //
       i2c_stop();

}

void getreading(void){

       i2c_start(APDS9960_WRITE);
       i2c_write(APDS9960_CDATAL); // set pointer
       i2c_stop();

       i2c_start(APDS9960_READ);
       Clear_data = (((int)i2c_read_ack()) | (int)i2c_read_ack() << 8);
       Red_data = (((int)i2c_read_ack()) | (int)i2c_read_ack() << 8);
       Green_data = (((int)i2c_read_ack()) | (int)i2c_read_ack() << 8);
       Blue_data = (((int)i2c_read_ack()) | (int)i2c_read_ack() << 8);

       i2c_stop();
}

int main(void){
       char buffer[20];
       //init_uart(9600);
       usart_init(); //manually set for 115200 BAUD rate as formula has too much error
       i2c_init();
       init_APDS9960();

       while(1){
              getreading();


              USART_SendString("\n-----------------------\n");

              sprintf(buffer,"Clear = %d, ",Clear_data);
              USART_SendString(buffer);
```

```
                sprintf(buffer,"Red = %d, ",Red_data);
                USART_SendString(buffer);

                sprintf(buffer,"Green = %d, ",Green_data);
                USART_SendString(buffer);

                sprintf(buffer,"Blue = %d\n",Blue_data);
                USART_SendString(buffer);

                USART_SendString("\n-----------------------\n");
                _delay_ms(1000);


                char thingSpeakUpdate[150];

                char setMUX[] = "AT+CIPMUX=0\r\n"; //Repeating the steps ESPlorer used
                USART_SendString(setMUX);
                _delay_ms(1000);

                char thingSpeakStart[] = 
"AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n" ;
                USART_SendString(thingSpeakStart);
                _delay_ms(1000); //add in delay to allow for proper interactions

                char thingSpeakSend[] = "AT+CIPSEND=119\r\n"; //saying we'll send more data 
than we actually will
                USART_SendString(thingSpeakSend);
                _delay_ms(1000);

                snprintf(thingSpeakUpdate, sizeof(thingSpeakUpdate), "GET 
https://api.thingspeak.com/update?api_key=P7JD9OICCNIR59PL&field1=%d&field2=%d&field3=%d&
field4=%d\r\n", Clear_data, Red_data, Green_data, Blue_data);
                USART_SendString(thingSpeakUpdate);
                _delay_ms(1000);

                char thingSpeakClose[] = "AT+CIPCLOSE\r\n";
                USART_SendString(thingSpeakClose);
                _delay_ms(1000);

                _delay_ms(1500);     //delay of 15s
        }

        return 0;
}
```

3. SCHEMATICS



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


Midterm_2 (Running) - AtmelStudio
Standard Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
address
Hex
ATmega328PB
debugWIRE on mEDBG (ATML2523052700001107)
Registers
uart.c i2c_master.c Task1 i2c_master.h APDS9960_def.h main.c stdlib.h
main.while
while(1)
Go
90 %
Watch 1
Name Value Type
Autos Watch 1 Watch 2 Breakpoints Output
Data Visualizer
Configuration
Interfaces:
Serial Port Control Panel
mEDBG Virtual COM Po
Disconnect
Terminal 5
Clear
Data Visualizer Solution Explorer Memory 4 Locals I/O Processor Status
Running

```
No data available
```

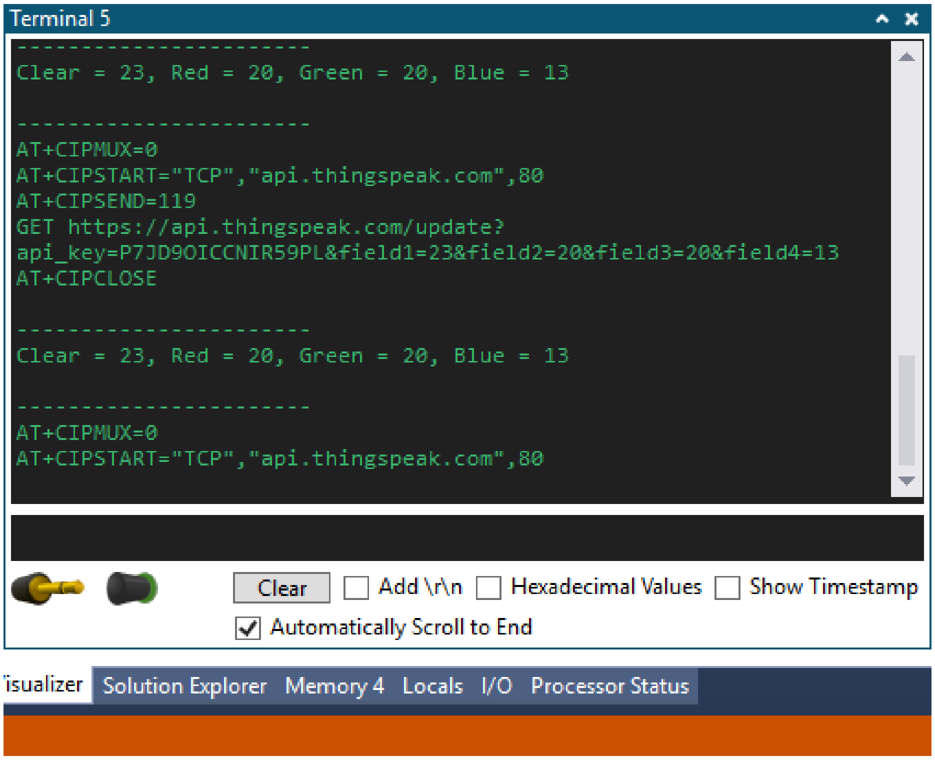
```c
    i2c_stop();
}

int main(void){
    char buffer[20];
    //init_uart(9600);
    usart_init(); //manually set for 115200 BAUD rate as formula has too much error
    i2c_init();
    init_APDS9960();

    while(1){
        getreading();


        USART_SendString("\n----------------------\n");

        sprintf(buffer,"Clear = %d, ",Clear_data);
        USART_SendString(buffer);

```

☑ DTR ☑ RTS
☑ Open Terminal
☐ Autodetect protocols

Baud rate: 115200 Parity: None Stop bits: 1 bit

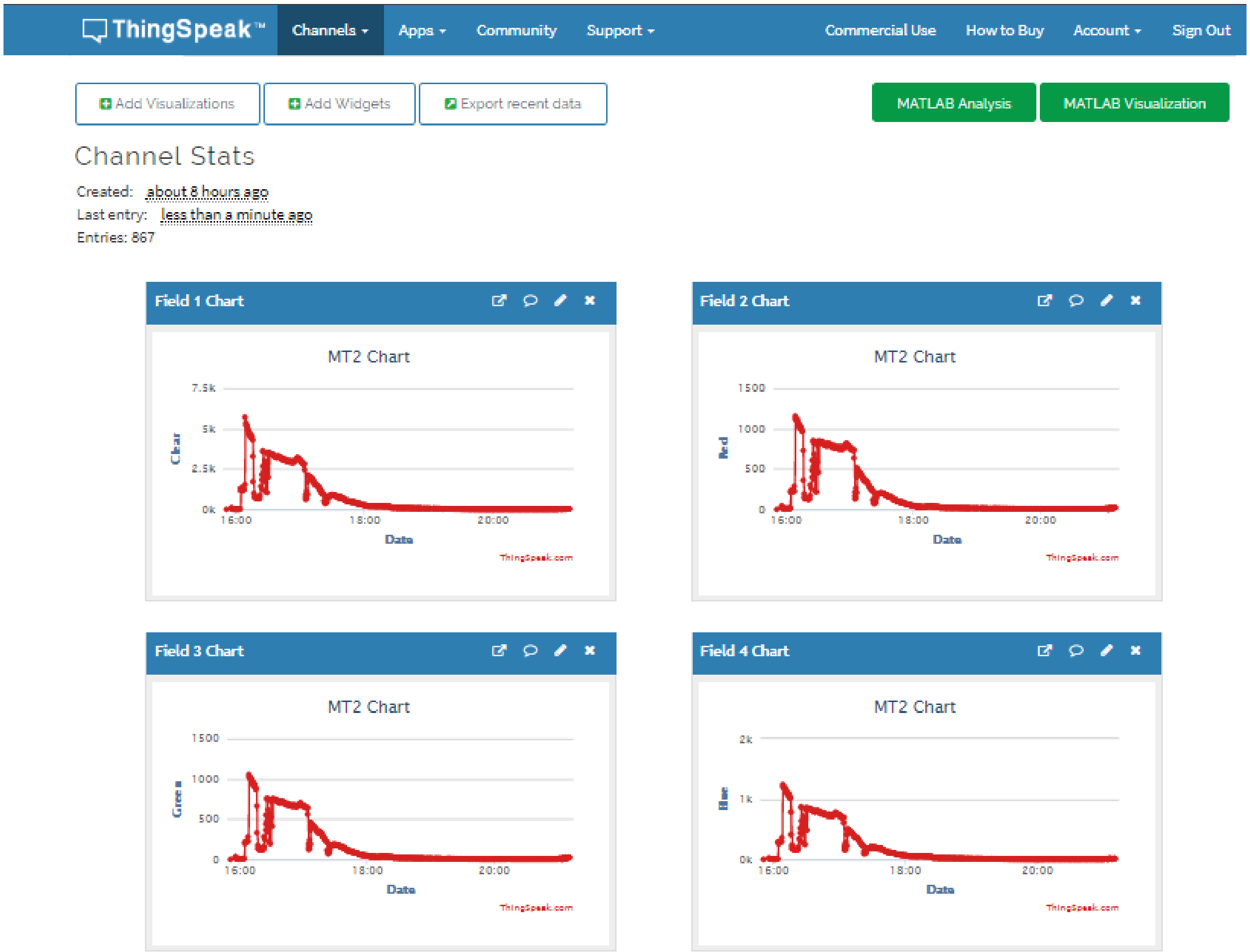
```
-----------------------
Clear = 23, Red = 20, Green = 20, Blue = 13

-----------------------
AT+CIPMUX=0
AT+CIPSTART="TCP","api.thingspeak.com",80
AT+CIPSEND=119
GET https://api.thingspeak.com/update?
api_key=P7JD9OICCNIR59PL&field1=23&field2=20&field3=20&field4=13
AT+CIPCLOSE

-----------------------
Clear = 23, Red = 20, Green = 20, Blue = 13

-----------------------
AT+CIPMUX=0
AT+CIPSTART="TCP","api.thingspeak.com",80
```

☐ Add \r\n ☐ Hexadecimal Values ☐ Show Timestamp
☑ Automatically Scroll to End



```
-----------------------
Clear = 23, Red = 20, Green = 20, Blue = 13

-----------------------
AT+CIPMUX=0
AT+CIPSTART="TCP","api.thingspeak.com",80
AT+CIPSEND=119
GET https://api.thingspeak.com/update?
api_key=P7JD9OICCNIR59PL&field1=23&field2=20&field3=20&field4=13
AT+CIPCLOSE

-----------------------
Clear = 23, Red = 20, Green = 20, Blue = 13

-----------------------
AT+CIPMUX=0
AT+CIPSTART="TCP","api.thingspeak.com",80
```

Output over serial at 115200 baudrate, displays RGBC values to user then sends commands to esp module (write key has since been changed).



Channel Stats

Created: about 8 hours ago

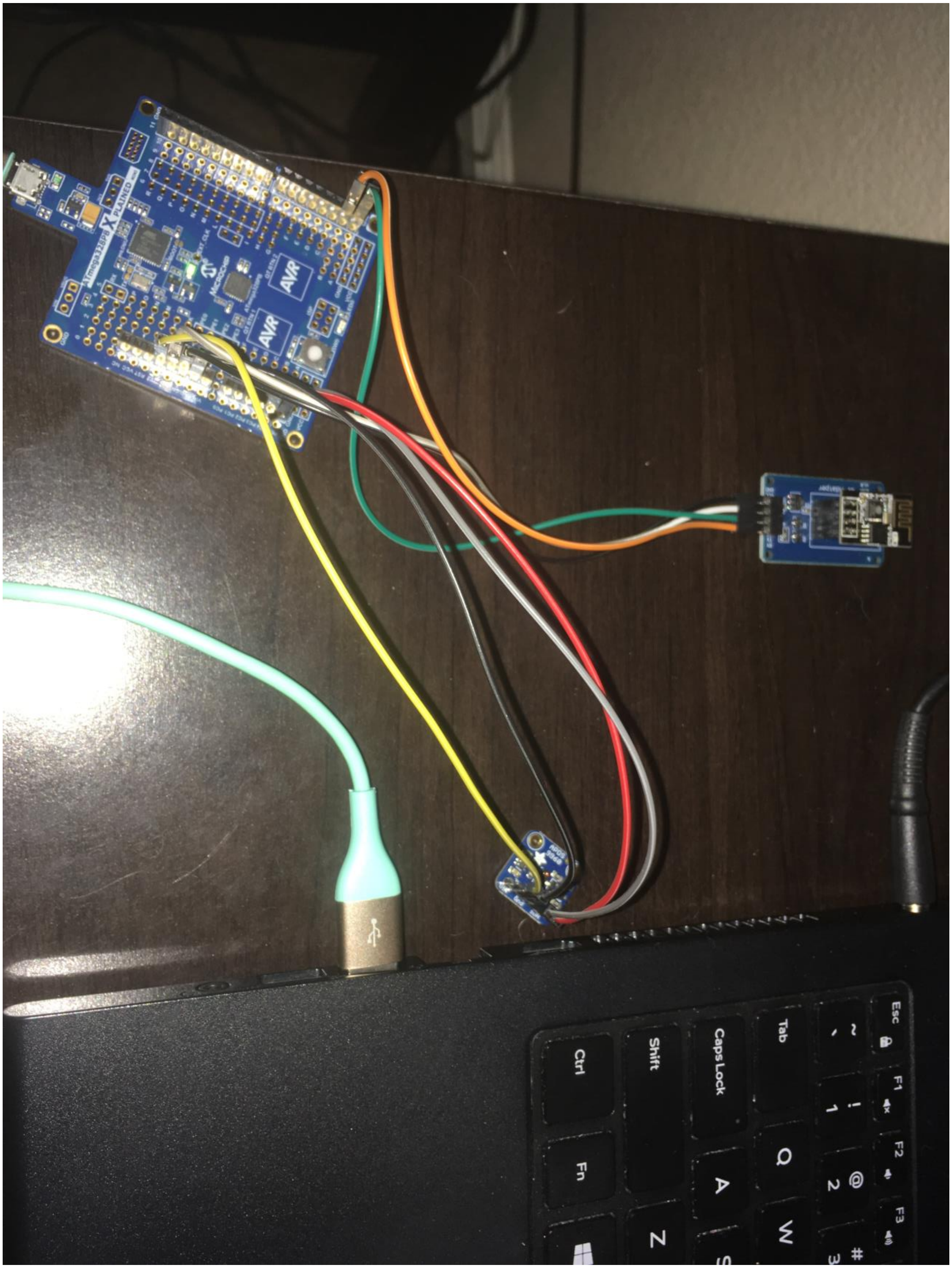
Last entry: less than a minute ago

Entries: 867

ThingSpeak graphs light values from a window starting just before 4pm and ending some time in the night.

Can also be viewed at https://thingspeak.com/channels/778393

5. SCREENSHOT OF EACH DEMO (BOARD SETUP)


ATmega328PB Xplained Mini
AVR
AVR
Esc
F1
F2
F3
Tab
Caps Lock
Shift
Ctrl
Fn
Q
W
A
Z

Board Setup, ESP8266 is plugged into ESP01-Adapter which is then connected to 5V and GND on Xplained mini while the TX line from the adapter is connected to RX (PD1) on the Xplained mini and then RX from the adapter to TX of the mini. The ASP9960 is connected to SCL (PC5) and SDA (PC4) on the mini as well as to 3.3V and GND.

6. VIDEO LINKS OF EACH DEMO

https://youtu.be/EYj7FFeU6o0

7. GITHUB LINK OF THIS DA

https://github.com/portig1/submissions_E/tree/master/Midterms/Midterm_2

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Geovanni Portillo