CPE301 – SPRING 2019

Midterm 2

Student Name: Allis Hierholzer

Student #: 2000160356

Student Email: hierholz@unlv.nevada.edu

Primary Github address: https://github.com/acexhp/submission\_da.git

Directory: Repository/cpe301/DesignAssignment/Midterm2

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 I2C of ATmega328/p to read RGB/Ambient Light data from APDS 9960 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 Lux sensor value to ESP-01/ESP32 through UART port using AT Commands.

6. Display the Lux 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**

* Breadboard
* Wires
* USB Cables
* Multi-functional shield
* ATMEL STUDIO 7.0
* APDS9960 Sensor
* ESP-32

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

/\*

\* Midterm2.c

\*/

#define F\_CPU 16000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <stdio.h>

#include "i2c\_master.h"

#include "uart.h"

#include "apds.h"

#define BAUD 9600

#define BRGVAL (F\_CPU/16/BAUD) - 1

#ifndef APDS\_H

#define APDS\_H

#include <avr/io.h>

#include "i2c\_master.h"

#include "apds.h"

#define APDS\_WRITE (0x39 << 1) | 0

#define APDS\_READ (0x39 << 1) | 1

//APDS-9960 I2C address

#define APDS9960\_I2C\_ADDR 0x39

#define ERROR 0xFF

//Device ID

#define APDS9960\_ID\_1 0xAB

#define APDS9960\_ID\_2 0x9C

//Parameters

// Wait -- FIFO reads

#define FIFO\_PAUSE\_TIME 30

//APDS-9960 register addresses

#define APDS9960\_ENABLE 0x80

#define APDS9960\_ATIME 0x81

#define APDS9960\_WTIME 0x83

#define APDS9960\_PERS 0x8C

#define APDS9960\_CONFIG1 0x8D

#define APDS9960\_PPULSE 0x8E

#define APDS9960\_CONFIG2 0x90

#define APDS9960\_ID 0x92

#define APDS9960\_RDATAL 0x96

#define APDS9960\_RDATAH 0x97

#define APDS9960\_GDATAL 0x98

#define APDS9960\_GDATAH 0x99

#define APDS9960\_BDATAL 0x9A

#define APDS9960\_BDATAH 0x9B

#define APDS9960\_POFFSET\_UR 0x9D

#define APDS9960\_POFFSET\_DL 0x9E

#define APDS9960\_CONFIG3 0x9F

//Bit fields

#define APDS9960\_PON 0x01

#define APDS9960\_AEN 0x02

#define APDS9960\_PEN 0x04

#define APDS9960\_WEN 0x08

#define APSD9960\_AIEN 0x10

#define APDS9960\_PIEN 0x20

#define APDS9960\_GEN 0x40

#define APDS9960\_GVALID 0x01

//Power On/Off

#define OFF 0

#define ON 1

//Parameter SetMode

#define POWER 0

#define AMBIENT\_LIGHT 1

//Proximity

#define WAIT 3

#define AMBIENT\_LIGHT\_INT 4

#define ALL 7

//LED Drive values

#define LED\_DRIVE\_100MA 0

#define LED\_DRIVE\_50MA 1

#define LED\_DRIVE\_25MA 2

#define LED\_DRIVE\_12\_5MA 3

//LED Boost values

#define LED\_BOOST\_100 0

#define LED\_BOOST\_150 1

#define LED\_BOOST\_200 2

#define LED\_BOOST\_300 3

//Default values

#define DEFAULT\_ATIME 219

#define DEFAULT\_WTIME 246

#define DEFAULT\_PROX\_PPULSE 0x87

#define DEFAULT\_POFFSET\_UR 0

#define DEFAULT\_POFFSET\_DL 0

#define DEFAULT\_CONFIG1 0x60 //wait (WTIME) factor

#define DEFAULT\_LDRIVE LED\_DRIVE\_100MA

#define DEFAULT\_PGAIN PGAIN\_4X

#define DEFAULT\_AGAIN AGAIN\_4X

#define DEFAULT\_AILT 0xFFFF //Force interrupt for calibration

#define DEFAULT\_AIHT 0

#define DEFAULT\_PERS 0x11 //Two consecutive prox or ALS for int.

#define DEFAULT\_CONFIG2 0x01 //No Saturation/LED boost

#define DEFAULT\_CONFIG3 0 //Enable all photodiodes

#define DEFAULT\_GLDRIVE LED\_DRIVE\_100MA

#define DEFAULT\_GWTIME GWTIME\_2\_8MS

void initialize\_APDS();

void Color\_detector();

#endif

*FILE* str\_uart = *FDEV\_SETUP\_STREAM*(uart\_char, *NULL* , *\_FDEV\_SETUP\_WRITE*);

char results[256];

void initialize\_UART();

int uart\_char( char c, *FILE* \*stream);

int main(void)

{

*uint16\_t* red = 0, green = 0, blue = 0;

i2c\_init();

initialize\_UART();

*stdout* = &str\_uart;

initialize\_APDS();

*\_delay\_ms*(2000);

*printf*("AT\r\n");

*\_delay\_ms*(3000);

*printf*("AT+CWMODE=1\r\n");

*\_delay\_ms*(3000);

*printf*("AT+CWJAP=\"Higbee\_Pixel\",\"Higbee19\"\r\n");

while (1)

{

*\_delay\_ms*(3000);

*printf*("AT+CIPMUX=0\r\n");

*\_delay\_ms*(3000);

*printf*("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n");

*\_delay\_ms*(3000);

Color\_detector(&red, &green, &blue);

*printf*("AT+CIPSEND=104\r\n");

*printf*("GET https://api.thingspeak.com/update?api\_key=8J8ALHG02UEU5SVN&field1=%05u&field2=%05u&field3=%05u\r\n", red, green, blue);

*\_delay\_ms*(3000);

}

}

void initialize\_APDS(){

*uint8\_t* setup;

i2c\_readReg(APDS\_WRITE, APDS9960\_ID, &setup,1);

if(setup != APDS9960\_ID\_1) while(1);

setup = 1 << 1 | 1<<0 | 1<<3 | 1<<4;

i2c\_writeReg(APDS\_WRITE, APDS9960\_ENABLE, &setup, 1);

setup = DEFAULT\_ATIME;

i2c\_writeReg(APDS\_WRITE, APDS9960\_ATIME, &setup, 1);

setup = DEFAULT\_WTIME;

i2c\_writeReg(APDS\_WRITE, APDS9960\_WTIME, &setup, 1);

setup = DEFAULT\_PROX\_PPULSE;

i2c\_writeReg(APDS\_WRITE, APDS9960\_PPULSE, &setup, 1);

setup = DEFAULT\_POFFSET\_UR;

i2c\_writeReg(APDS\_WRITE, APDS9960\_POFFSET\_UR, &setup, 1);

setup = DEFAULT\_POFFSET\_DL;

i2c\_writeReg(APDS\_WRITE, APDS9960\_POFFSET\_DL, &setup, 1);

setup = DEFAULT\_CONFIG1;

i2c\_writeReg(APDS\_WRITE, APDS9960\_CONFIG1, &setup, 1);

setup = DEFAULT\_PERS;

i2c\_writeReg(APDS\_WRITE, APDS9960\_PERS, &setup, 1);

setup = DEFAULT\_CONFIG2;

i2c\_writeReg(APDS\_WRITE, APDS9960\_CONFIG2, &setup, 1);

setup = DEFAULT\_CONFIG3;

i2c\_writeReg(APDS\_WRITE, APDS9960\_CONFIG3, &setup, 1);

}

void readColor(*uint16\_t* \*red, *uint16\_t* \*green, *uint16\_t* \*blue){

*uint8\_t* redl, redh;

*uint8\_t* greenl, greenh;

*uint8\_t* bluel, blueh;

i2c\_readReg(APDS\_WRITE, APDS9960\_RDATAL, &redl, 1);

i2c\_readReg(APDS\_WRITE, APDS9960\_RDATAH, &redh, 1);

i2c\_readReg(APDS\_WRITE, APDS9960\_GDATAL, &greenl, 1);

i2c\_readReg(APDS\_WRITE, APDS9960\_GDATAH, &greenh, 1);

i2c\_readReg(APDS\_WRITE, APDS9960\_BDATAL, &bluel, 1);

i2c\_readReg(APDS\_WRITE, APDS9960\_BDATAH, &blueh, 1);

\*red = redh << 8 | redl;

\*green = greenh << 8 | greenl;

\*blue = blueh << 8 | bluel;

}

void initialize\_UART(void){

//Set baud rate

*uint16\_t* baud\_rate = BRGVAL;

UBRR0H = baud\_rate >> 8;

UBRR0L = baud\_rate & 0xFF;

//Enable receiver/transmitter

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

//Set frame format

UCSR0C = (3 <<UCSZ00);

}

int uart\_char(char c, *FILE* \*stream){

//Empty buffer

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

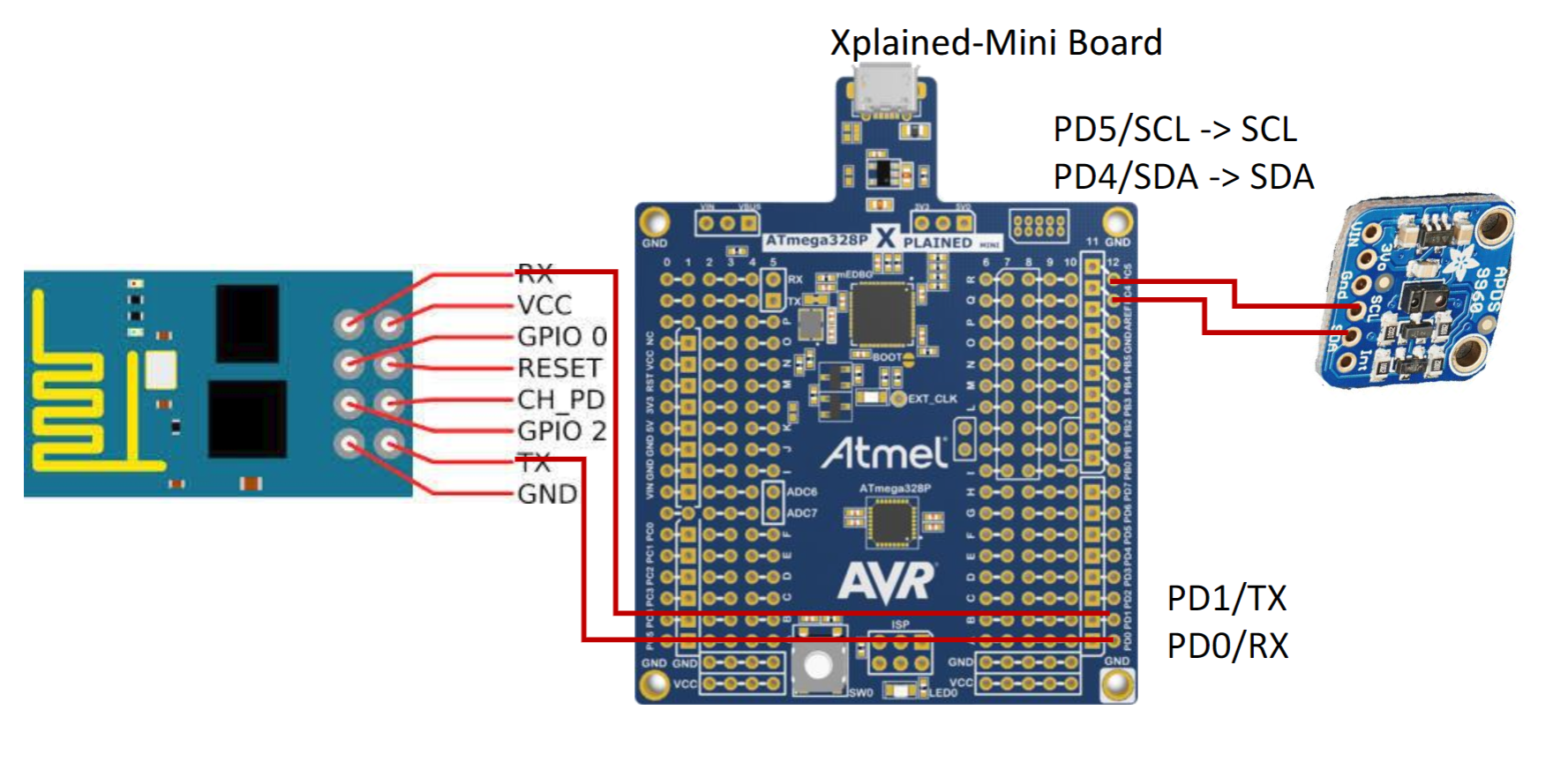
//Data to buffer

UDR0 = c;

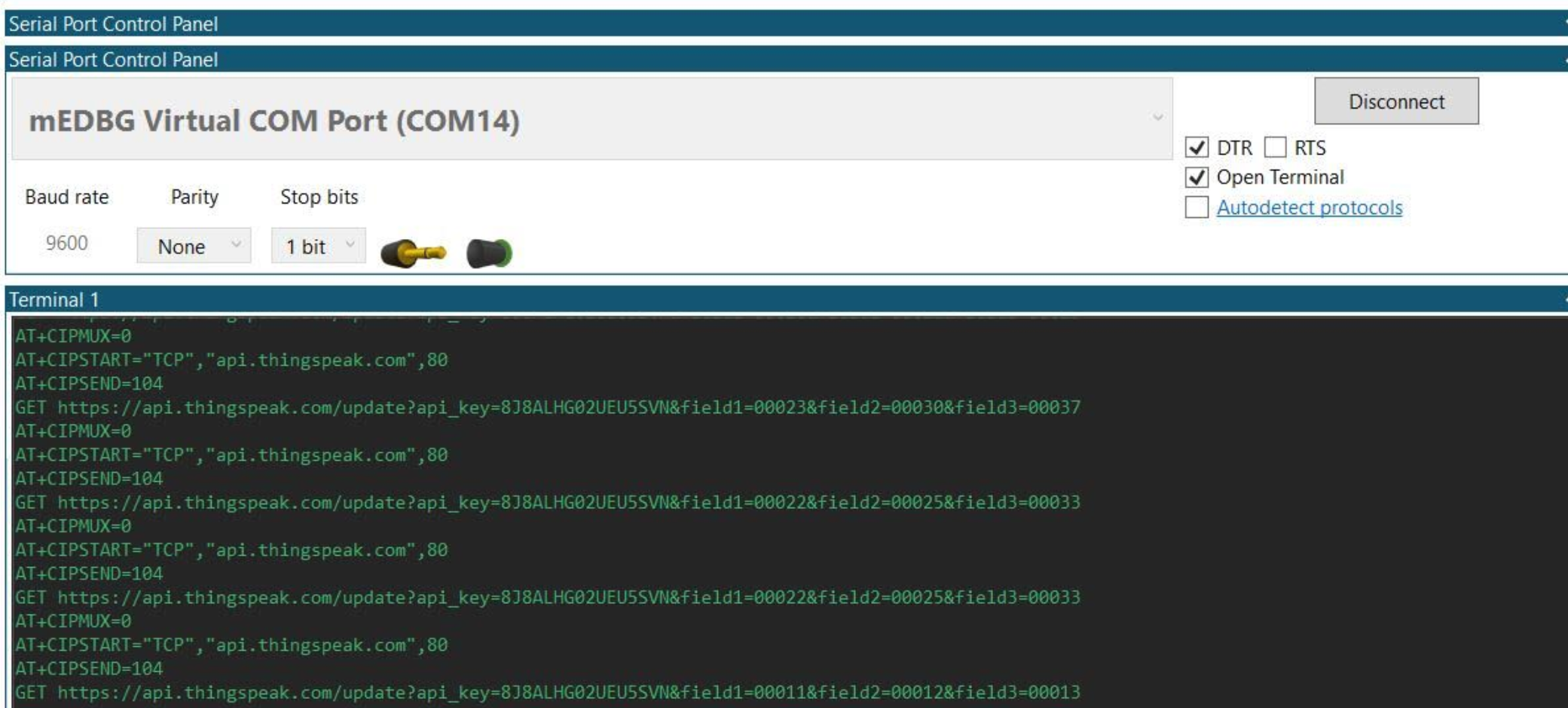
return 0;

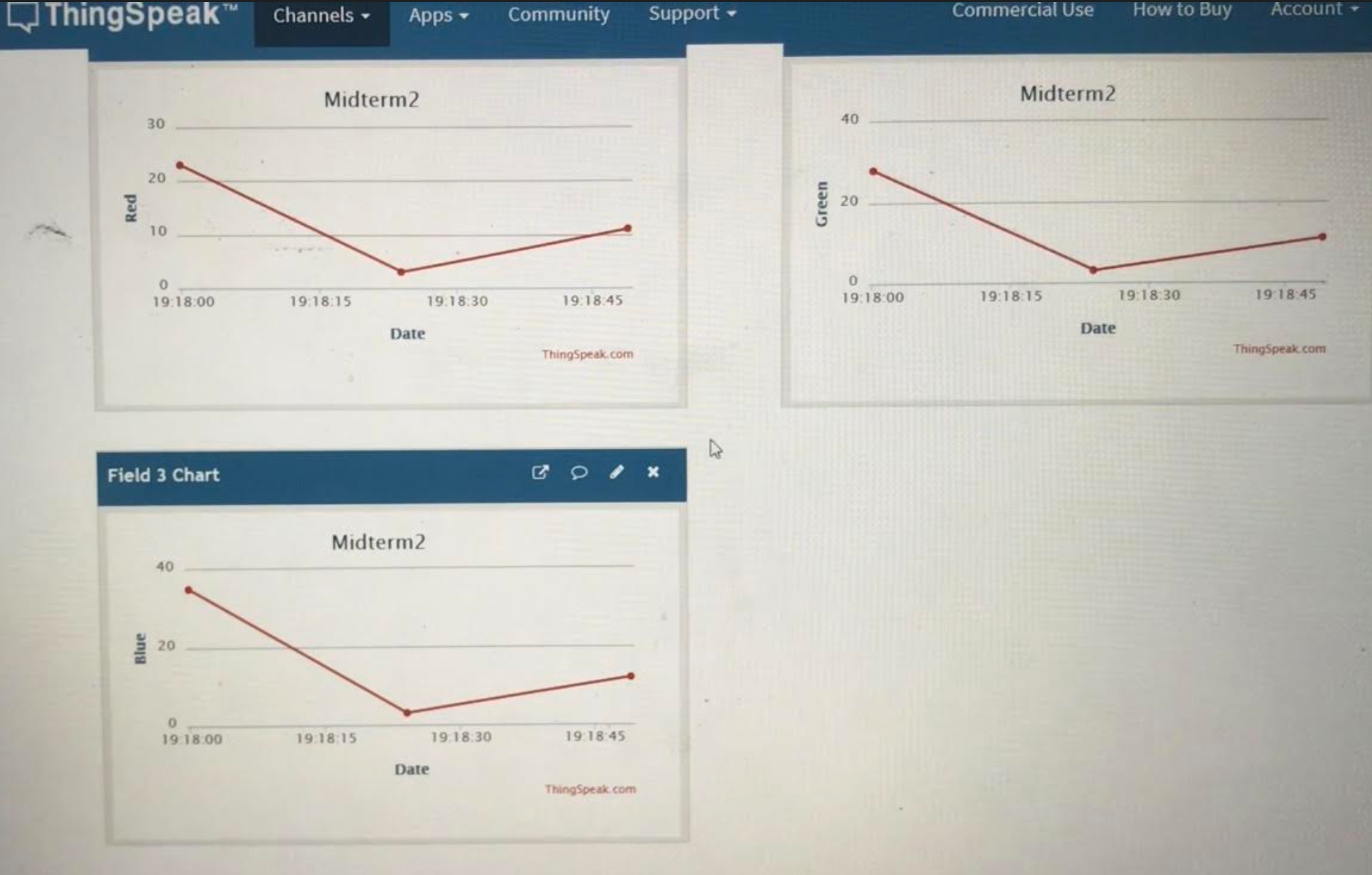
}

1. **SCHEMATICS**

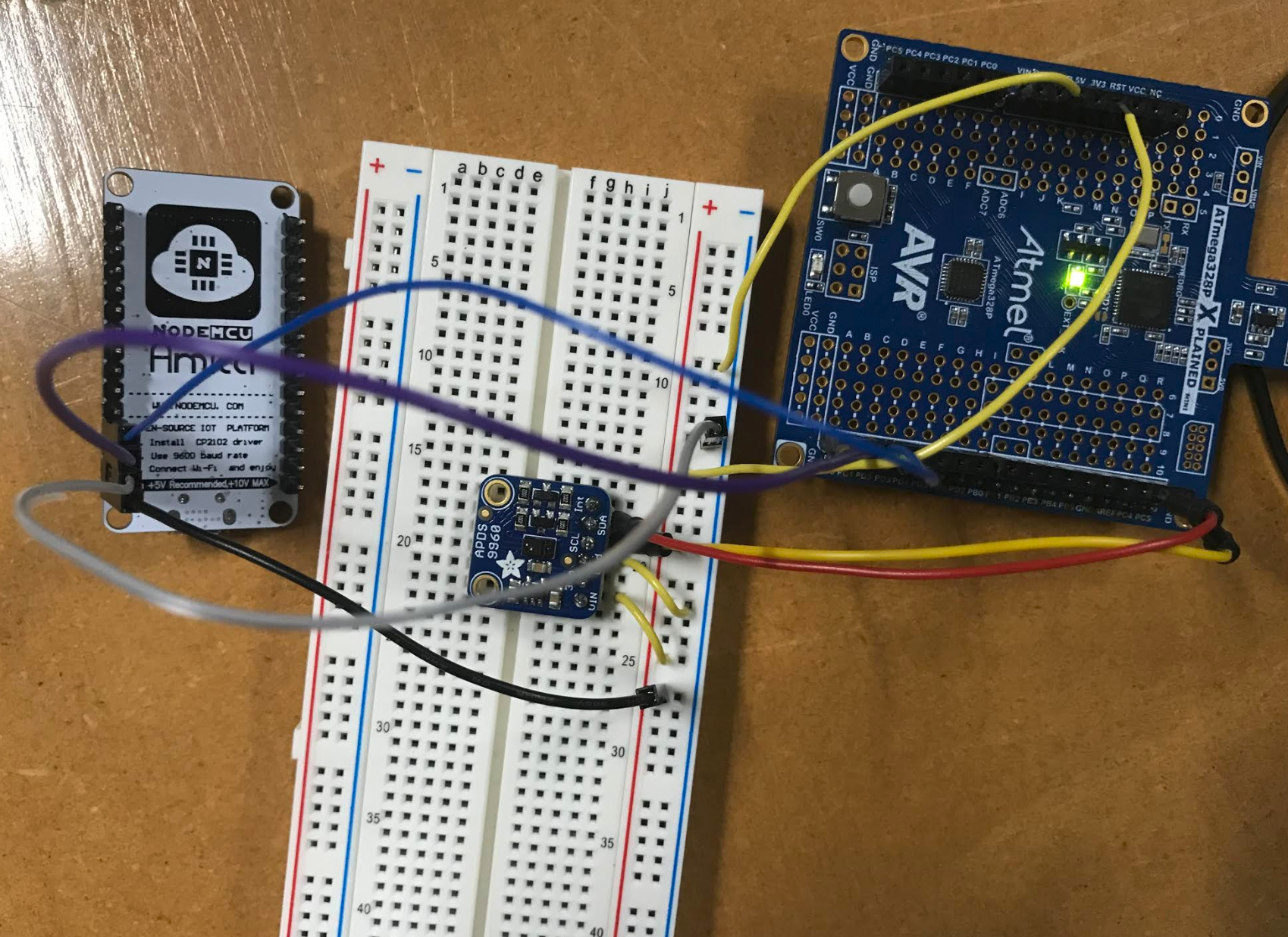


1. **SCREENSHOTS OF EACH TASK OUTPUT (OUTPUT)**





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



1. **VIDEO LINKS OF EACH DEMO**

[**https://youtu.be/dAISH9QtOwM**](https://youtu.be/dAISH9QtOwM)

1. **GITHUB LINK OF THIS DA**

<https://github.com/acexhp/submission_da.git>

**Student Academic Misconduct Policy**

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

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

Allis Hierholzer