CPE301 – SPRING 2019

MIDTERM 2

Student Name: Eric Smith

Student #: 2000543940

Student Email: smithe4@unlv.nevada.edu

Primary Github address: <https://github.com/smitheas95>

Directory: C:\Users\eas7w\OneDrive\Documents\Repository\cpe301\Midterms\Midterm\_2\Midterm2

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

**List:**

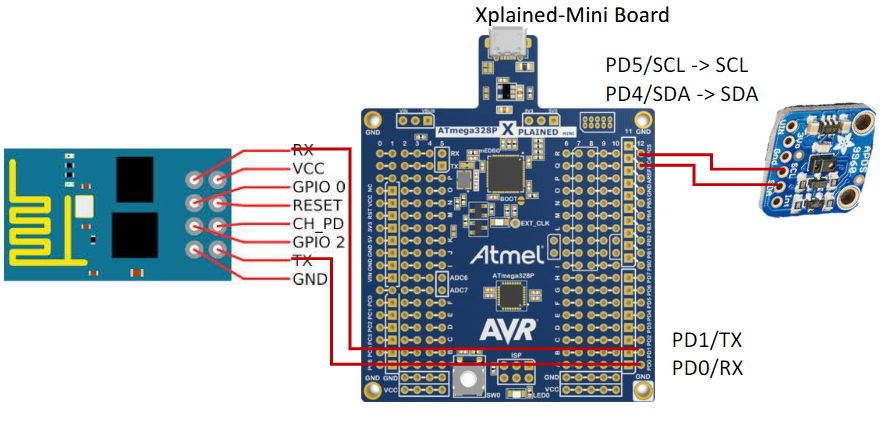
ATMega328P XPLAINED MINI

APDS-9960

Wires (male-male, male-female)

WIFI – ESP8266MOD

Breadboard (provided from cpe310L lab kit)



1. **INITIAL/MODIFIED/DEVELOPED CODE OF MIDTERM 2**



/\*

midterm2.c

Created: 5/11/2019 3:37:39 PM

Author : Eric Smith

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

\*/

#define F\_CPU 16000000UL // DEFINE 16MHz

// NEEDED LIBRARIES

#include <avr/io.h>

#include <stdio.h>

#include <util/delay.h>

#include "uart.h"

#include "apds.h"

#include "i2c\_master.h"

// BAUDRATE DEFINITIONS

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

#define BAUD 9600

// APDS DEFINITIONS

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

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

#ifndef APDS\_H

#define APDS\_H

#define APDS9960\_I2C\_ADDR 0x39 // I2C ADDRESS FOR APDS-9960

#define ERROR 0xFF // RETURNED VALUES ERROR CODE

// MISC PARAMETERS

#define FIFO\_PAUSE\_TIME 30 // FIFO READS, WAIT PERIOD IN ms

// ACCEPTABLE DEVICE IDs

#define APDS9960\_ID\_1 0xAB

#define APDS9960\_ID\_2 0x9C

// BIT FIELDS

#define APDS9960\_PON 0b00000001

#define APDS9960\_AEN 0b00000010

#define APDS9960\_PEN 0b00000100

#define APDS9960\_WEN 0b00001000

#define APSD9960\_AIEN 0b00010000

#define APDS9960\_PIEN 0b00100000

#define APDS9960\_GEN 0b01000000

#define APDS9960\_GVALID 0b00000001

// 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

// ON/OFF DEFINITIONS

#define OFF 0

#define ON 1

// ACCEPTABLE PARAMTERS FOR SETMODE

#define POWER 0

#define AMBIENT\_LIGHT 1

// DEFINE PROXIMITY

#define WAIT 3

#define AMBIENT\_LIGHT\_INT 4

#define ALL 7

// LED BOOST VALUES

#define LED\_BOOST\_100 0

#define LED\_BOOST\_150 1

#define LED\_BOOST\_200 2

#define LED\_BOOST\_300 3

// LED DRIVE VALUES

#define LED\_DRIVE\_100MA 0

#define LED\_DRIVE\_50MA 1

#define LED\_DRIVE\_25MA 2

#define LED\_DRIVE\_12\_5MA 3

// DEFAULT VALUES

#define DEFAULT\_ATIME 219 // 103 ms VALUE

#define DEFAULT\_WTIME 246 // 27 ms VALUE

#define DEFAULT\_PROX\_PPULSE 0x87 // 16 us, PULSE OF 8

#define DEFAULT\_POFFSET\_UR 0 // OFFSET 0

#define DEFAULT\_POFFSET\_DL 0 // OFFSET 0

#define DEFAULT\_CONFIG1 0x60 // WAIT FACTOR FOR NO 12x

#define DEFAULT\_LDRIVE LED\_DRIVE\_100MA

#define DEFAULT\_PGAIN PGAIN\_4X

#define DEFAULT\_AGAIN AGAIN\_4X

#define DEFAULT\_AILT 0xFFFF // CALIBRATION FORCE INTERRUPT

#define DEFAULT\_AIHT 0

#define DEFAULT\_PERS 0x11 // 2 CONSECUTIVE PROX OR ALS FOR INT.

#define DEFAULT\_CONFIG2 0x01 // SATURATION INTERRUPTS, LED BOOST NONE

#define DEFAULT\_CONFIG3 0 // ENABLE ALL PHOTODIODES, NO SAI ENABLE ALL PHOTODIODES

#define DEFAULT\_GLDRIVE LED\_DRIVE\_100MA

#define DEFAULT\_GWTIME GWTIME\_2\_8MS

void apds\_init(); // VOID FUNCTION FOR APDS INIT

void colorRead(); // VOID FUNCTION FOR READING COLOR

#endif

// FILE

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

char results[256]; // 256

void init\_UART(); // VOID FUNCTION FOR INIT UART

int uart\_putchar( char c, *FILE* \*stream); // VARIABLE INT FOR PUTTING CHARACTERS UART

int main(void) // INT MAIN & CONTAINS THINGSPEAK PORTION

{

*uint16\_t* red = 0, green = 0, blue = 0; // SET RED, GREEN, BLUE TO ZERO

i2c\_init();

init\_UART();

*stdout* = &string\_uart; // STRING UART

apds\_init();

// RESPECTIVE DELAYS PROVIDED

*\_delay\_ms*(2500);

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

*\_delay\_ms*(4500);

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

*\_delay\_ms*(4500);

*printf*("AT+CWJAP=\"n/aVerizon-SM-G950U-748C\",\"n/anifj003)\"\r\n"); // WHERE I PUT MY HOTSPOT AND PASSWORD FOR HOTSPOT (ALTERED FOR SECURITY REASONS)

while (1) // KEY WHILE LOOP

{

*\_delay\_ms*(4500);

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

*\_delay\_ms*(4500);

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

*\_delay\_ms*(4500);

colorRead(&red, &green, &blue);

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

// API KEY WITH RESPECTIVE FIELDS FOR 3 COLORS

*printf*("GET https://api.thingspeak.com/update?api\_key=PRH9WSZ8I5Y3A3LT&field1=%05u&field2=%05u&field3=%05u\r\n", red, green, blue);

*\_delay\_ms*(4500);

}

}

void init\_UART(void){

// BAUD RATE SET

*uint16\_t* baud\_rate = BRGVAL;

UBRR0H = baud\_rate >> 8;

UBRR0L = baud\_rate & 0xFF;

// RECEIVER AND TRANSMITTER ENABLED

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

// FRAME FORMAT SET: 8 DATA, 1 STOP BIT

UCSR0C = (3 <<UCSZ00);

}

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

// BUFFER IS EMPTY, WAIT TIL THEN

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

// BUFFER GETS DATA

UDR0 = c;

return 0;

}

void apds\_init(){

*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

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 colorRead(*uint16\_t* \*red, *uint16\_t* \*green, *uint16\_t* \*blue){

*uint8\_t* redl, redh; // RED LOW, RED HIGH

*uint8\_t* greenl, greenh; // GREEN LOW, GREEN HIGH

*uint8\_t* bluel, blueh; // BLUE LOW, BLUE HIGH

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;

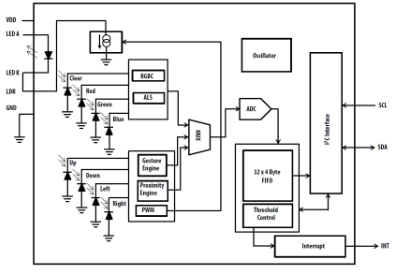
}

1. **DEVELOPED MODIFIED CODE OF MIDTERM 2**

(NOT NEEDED, COVERED IN QUESTION #2)

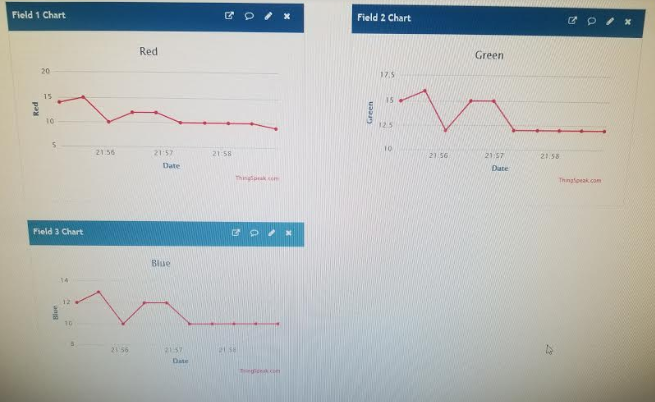
1. **SCHEMATICS**

Key schematic:



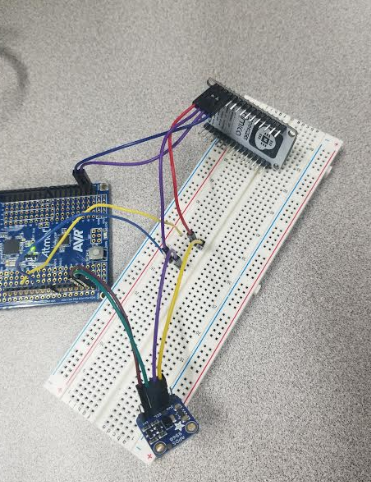
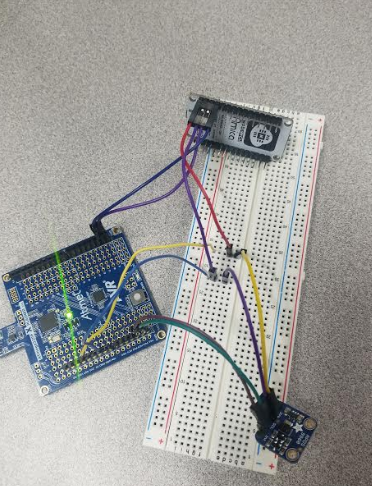
1. **SCREENSHOTS OF EACH TASK OUTPUT (THINGSPEAK OUTPUT)**

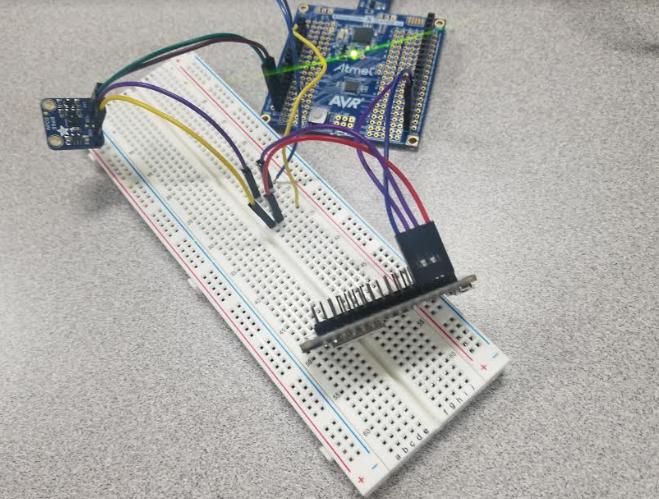
**Note:** At time mark of 21:56, all colors spiked/went up when flashlight via phone was on the APDS sensor (can be observed in video link provided)

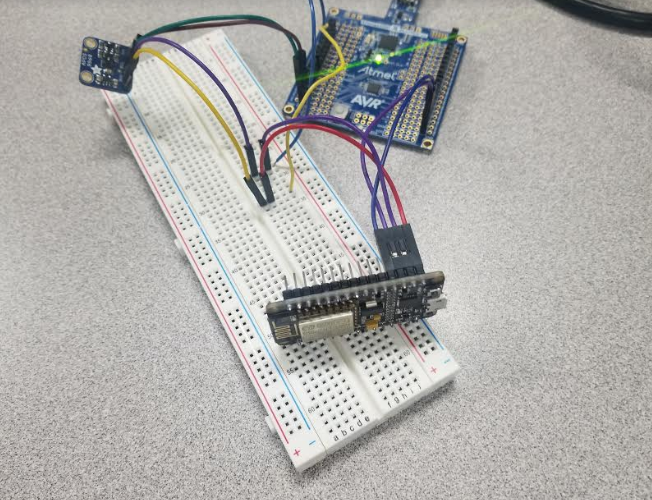


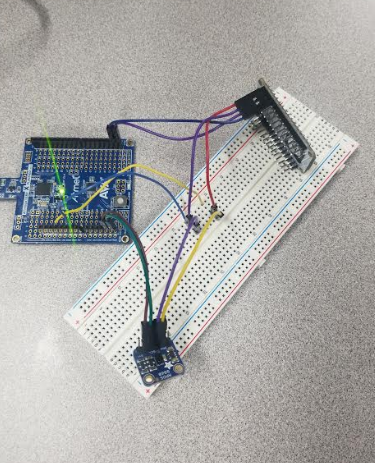
1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

Pictures provided below:











1. **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/Ub53ePHh1fQ>

1. **GITHUB LINK OF THIS DA**

<https://github.com/smitheas95/Midterm2>

**Student Academic Misconduct Policy**

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

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

Eric Smith

Key resources provided by professor (Dr. Venki):

