CPE301 – SPRING 2019

Midterm 2

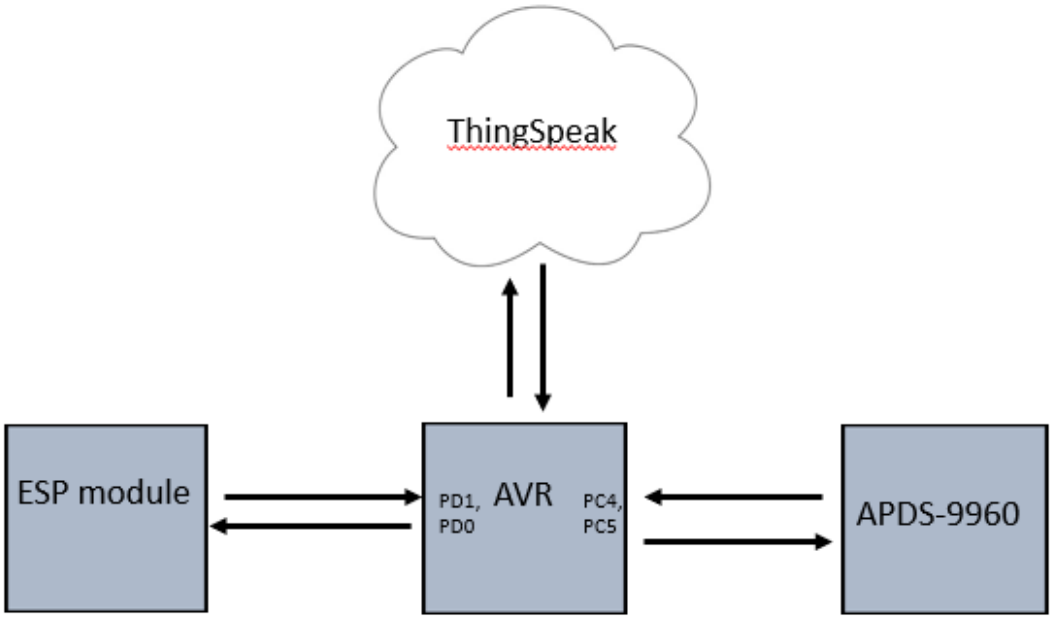
Student Name: Francisco Mata Carlos
Student #: 1012593607
Student Email: matacarl@unlv.nevada.edu
Primary Github address: https://github.com/chicosisco/da_sub.git
Directory: repository/cpe301/DesignAssignments/midterm2

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

The components used for this assignment are the next:

a. Atmega328p Xplained Mini
b. Atmel Studio 7
c. ESP8266 with module
d. APDS-9960 (Digital Proximity, Ambient Light, RGB and Gesture Sensor)

Block diagram with pins used



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

1. Task 1

Write, simulate, and demonstrate using Atmel Studio 7 a C code for the AVR ATMEGA328p microcontroller that performs the following functions:
Program the I2C of ATmega328/p to read RGB/Ambient Light data from APDS 9960 sensor. Display the value to UART. Make sure the AT Firmware is downloaded into the ESP-01/ESP32 module. Register for a free Thingspeak account with MATHWORK. Setup and get the channel Key. Transmit Lux sensor value to ESP-01/ESP32 through UART port using AT Commands. Display the Lux sensor value as a graph in Thingspeak

```
#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 info
#define APDS9960_I2C_ADDR       0x39

//code for returned values with errors
#define ERROR                   0xFF

// device IDs
#define APDS9960_ID_1           0xAB
#define APDS9960_ID_2           0x9C

//parameters
#define FIFO_PAUSE_TIME         30

//APDS-9960 register addresses info
#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 info
#define APDS9960_PON            1
#define APDS9960_AEN            2
#define APDS9960_PEN            4
#define APDS9960_WEN            8
#define APSD9960_AIEN           16
#define APDS9960_PIEN           32
#define APDS9960_GEN            64
#define APDS9960_GVALID         1

#define OFF                     0
#define ON                      1

//parameters for set-modes
#define POWER                   0
#define AMBIENT_LIGHT           1

#define WAIT                    3
#define AMBIENT_LIGHT_INT       4
#define ALL                     7

//LED Drive values for LED driver
#define LED_DRIVE_100MA         0
#define LED_DRIVE_50MA          1
#define LED_DRIVE_25MA          2
#define LED_DRIVE_12_5MA        3

//values for LED boost
#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     // 27ms
#define DEFAULT_PROX_PPULSE     0x87    // 16us, 8 pulses
#define DEFAULT_POFFSET_UR      0
#define DEFAULT_POFFSET_DL      0       // 0 offset
#define DEFAULT_CONFIG1         0x60    // No 12x wait (WTIME) factors
#define DEFAULT_LDRIVE          LED_DRIVE_100MA
```

```
#define DEFAULT_PGAIN           PGAIN_4X
#define DEFAULT_AGAIN           AGAIN_4X
#define DEFAULT_AILT            0xFFFF
#define DEFAULT_AIHT            0
#define DEFAULT_PERS            0x11    // 2 consecutive prox
#define DEFAULT_CONFIG2         0x01    // No saturation interrupts or LED boost
#define DEFAULT_CONFIG3         0       // Enable all photo-diodes
#define DEFAULT_GLDRIVE         LED_DRIVE_100MA
#define DEFAULT_GWTIME          GWTIME_2_8MS

void reading_colors();
void ini_APD();
#endif


void uart_ini();
int uart_putchar( char c, FILE *stream);

FILE str_uart = FDEV_SETUP_STREAM(uart_putchar, NULL , _FDEV_SETUP_WRITE);
char results[256];


int main(void)
{
	uint16_t red = 0, green = 0, blue = 0;

	i2c_init();
	uart_ini();
	stdout = &str_uart;
	ini_APD();

	_delay_ms(2000);
	printf("AT\r\n");

	_delay_ms(3000);
	printf("AT+CWMODE=1\r\n");

	_delay_ms(3000);
	printf("AT+CWJAP=\"Wifi\",\"password\"\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);
		reading_colors(&red, &green, &blue);
		printf("AT+CIPSEND=104\r\n");
		printf("GET
https://api.thingspeak.com/update?api_key=L8RI7VYEFL3ZID5U&field1=%05u&field2=%05u&field3
=%05u\r\n", red, green, blue);

		_delay_ms(3000);
	}
}
```

```
void uart_ini(void){
	//setting baud rate
	uint16_t baud_rate = BRGVAL;
	UBRR0H = baud_rate >> 8;
	UBRR0L = baud_rate & 0xFF;

	//Enable in order to receive and transmit
	UCSR0B = ( 1 <<RXEN0)|( 1 <<TXEN0);

	// Seting the frame format: 8bitdata
	UCSR0C = (3 <<UCSZ00);
}

void ini_APD(){
	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);
}


int uart_putchar(char c, FILE *stream){
	//wait until buffer empty
	while ( !( UCSR0A & ( 1 <<UDRE0)) );

	//Put data into buffer
	UDR0 = c;
	return 0;
}

void reading_colors(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);
```

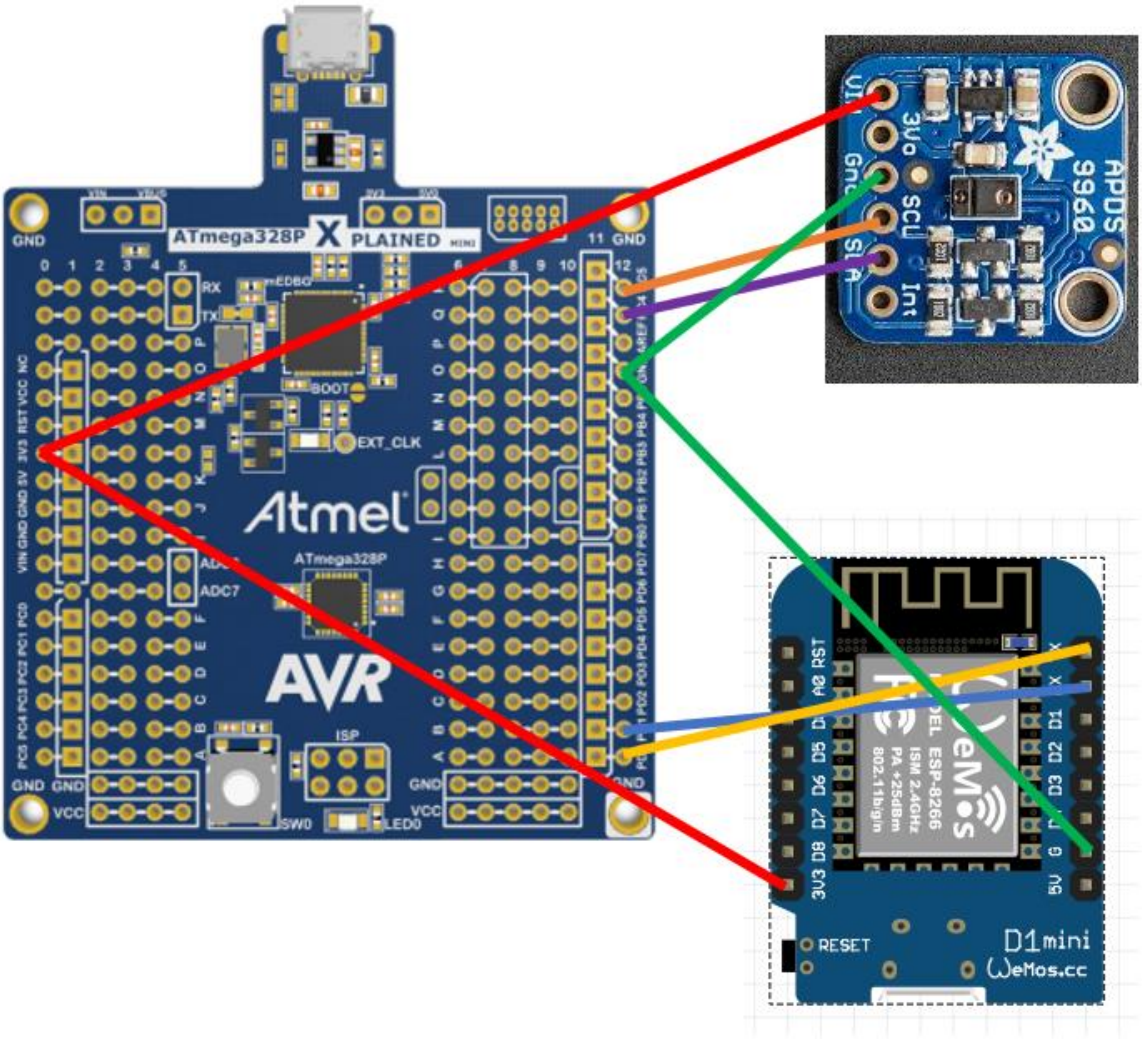
```
        i2c_readReg(APDS_WRITE, APDS9960_GDATAH, &greenh, 1);
        i2c_readReg(APDS_WRITE, APDS9960_BDATAL, &bluel, 1);
        i2c_readReg(APDS_WRITE, APDS9960_BDATAH, &blueh, 1);
        *red = redh << 8 | redl;
        *green = greenh << 8 | greenl;
        *blue = blueh << 8 | bluel;
}
```

3. DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A

Same as above

4. SCHEMATICS

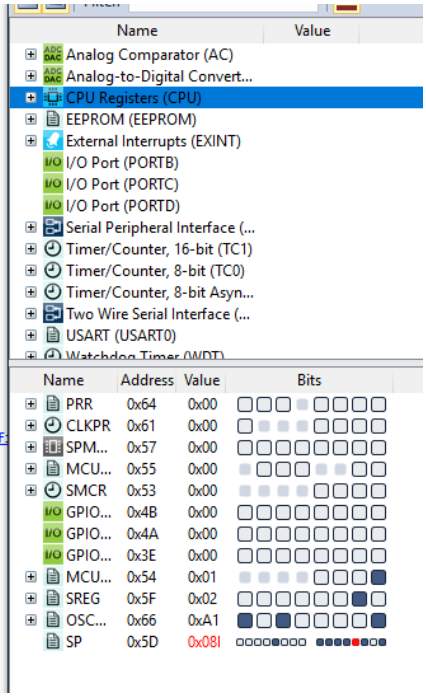
Task 1



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

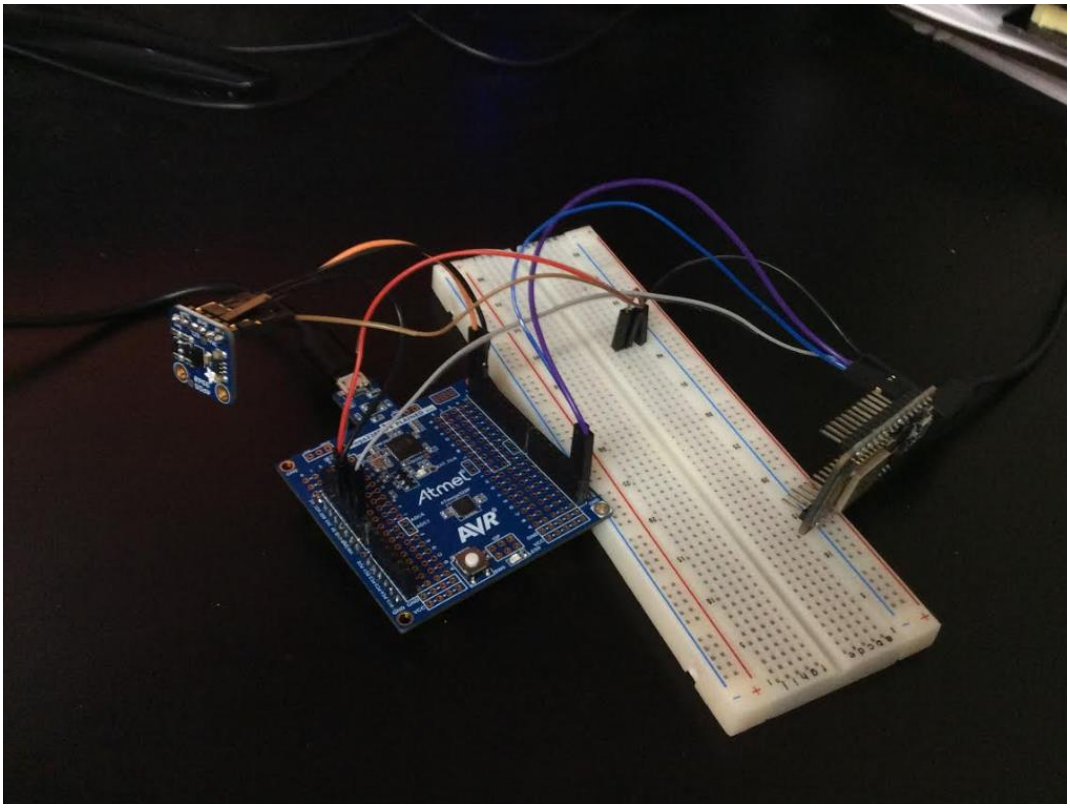
TASK 1

```
  ini_APD();

  _delay_ms(2000);
  printf("AT\r\n");

  _delay_ms(3000);
  printf("AT+CWMODE=1\r\n");

  _delay_ms(3000);
  printf("AT+CWJAP=\"Wifi\",\"password\"\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);
      reading_colors(&red, &green, &blue);
      printf("AT+CIPSEND=104\r\n");
      printf("GET https://api.thingspeak.com/update?api_key=L8RI7VYEFL3ZID5U&field1=%05u&field2=%05u&f

      _delay_ms(3000);
  }


oid uart_ini(void){
  //setting baud rate
  uint16_t baud_rate = BRGVAL;
  UBRR0H = baud_rate >> 8;
  UBRR0L = baud_rate & 0xFF;

  //Enable in order to receive and transmit
```



6. SCREENSHOTS OF ThingSpeak output

7. SCREENSHOT OF EACH DEMO (BOARD SETUP)



8. VIDEO LINKS OF EACH DEMO

Task 1 video:
https://youtu.be/oZrWNseDvTQ

9. GITHUB LINK OF THIS DA

https://github.com/chicosisco/da_sub.git

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

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