CPE301 – SPRING 2019

Midterm 2

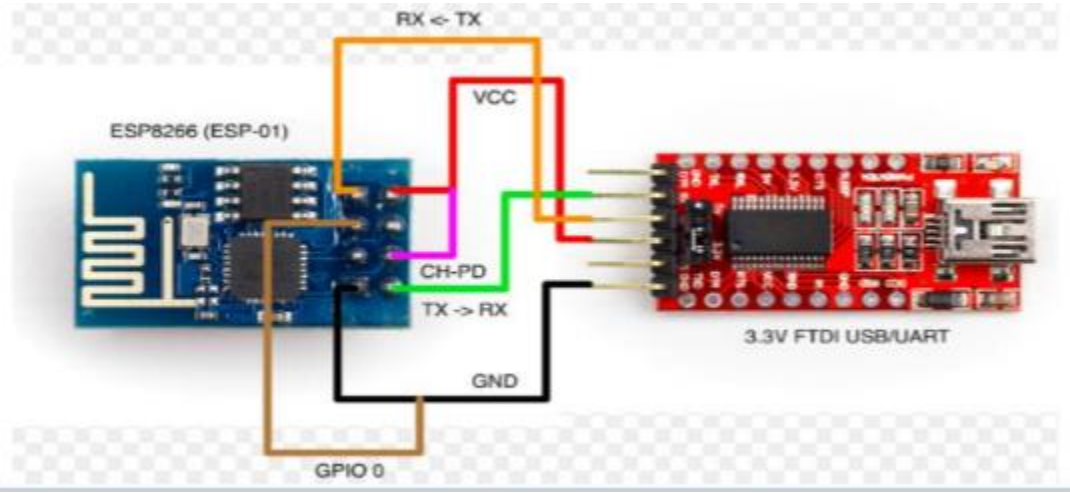
Student Name: Tyler Gardenhire
Student #: 8000450294
Student Email: gardenhi@unlv.nevada.edu
Primary Github address: gardenhi@unlv.nevada.edu
Directory: https://github.com/tylergardenhire/submission_projects.git

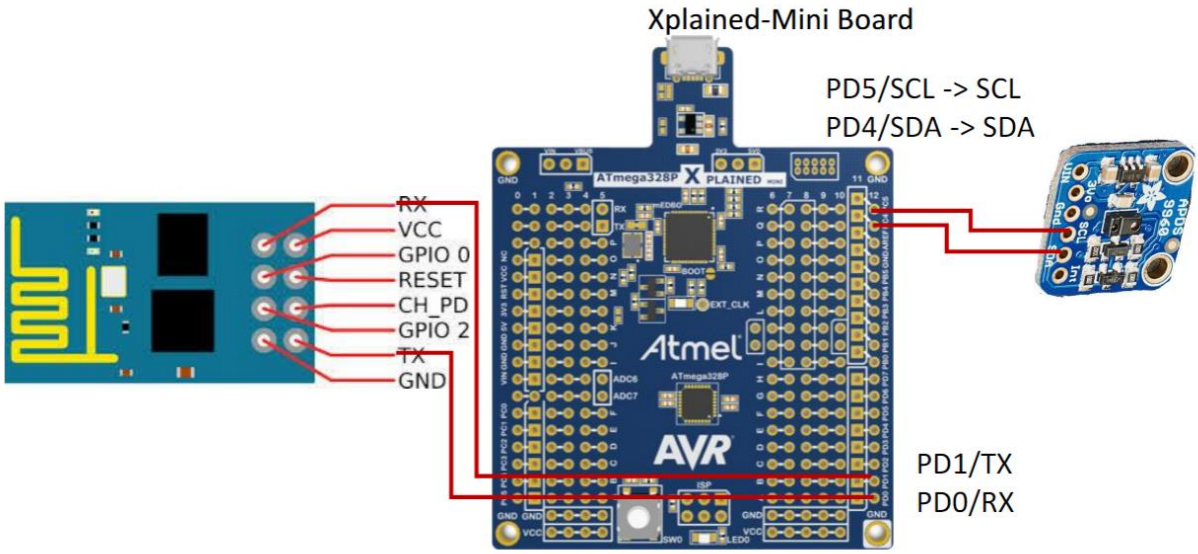
Submit the following for all Labs:

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

Atmel Studio 7 w/ AVR assembly, Atmega328p board, FTDI chip, and ESP module used.





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

```
/*
 * Midterm 2
 * Created: 5/13/2019 3:21:23 PM
 * Author : Tyler Gardenhire
 */

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

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

void UART_init (void);
void APDS_init (void);

int uart_putchar( char c, FILE *stream);
FILE str_uart = FDEV_SETUP_STREAM(uart_putchar, NULL , _FDEV_SETUP_WRITE);
void getreading(void);

uint16_t red;
uint16_t green;
uint16_t blue;

int main(void)
{
	UART_init(); //initialize uart values
	APDS_init(); //initialize APDS9960
	i2c_init();  //initialize I2C
	stdout = &str_uart;

	red = 0;
	green = 0;
	blue = 0;

	_delay_ms(5000);
	printf("AT+CWMODE=3\r\n");  //set AP's info

	_delay_ms(5000);
	printf("AT+CWJAP=\"eos2f8\",\"122gfh48\"\r\n"); //connect to Internet

	while(1)//send values through the cloud
	{

		_delay_ms(5000);
		printf("AT+CIPMUX=0\r\n");  //enable single connection
```

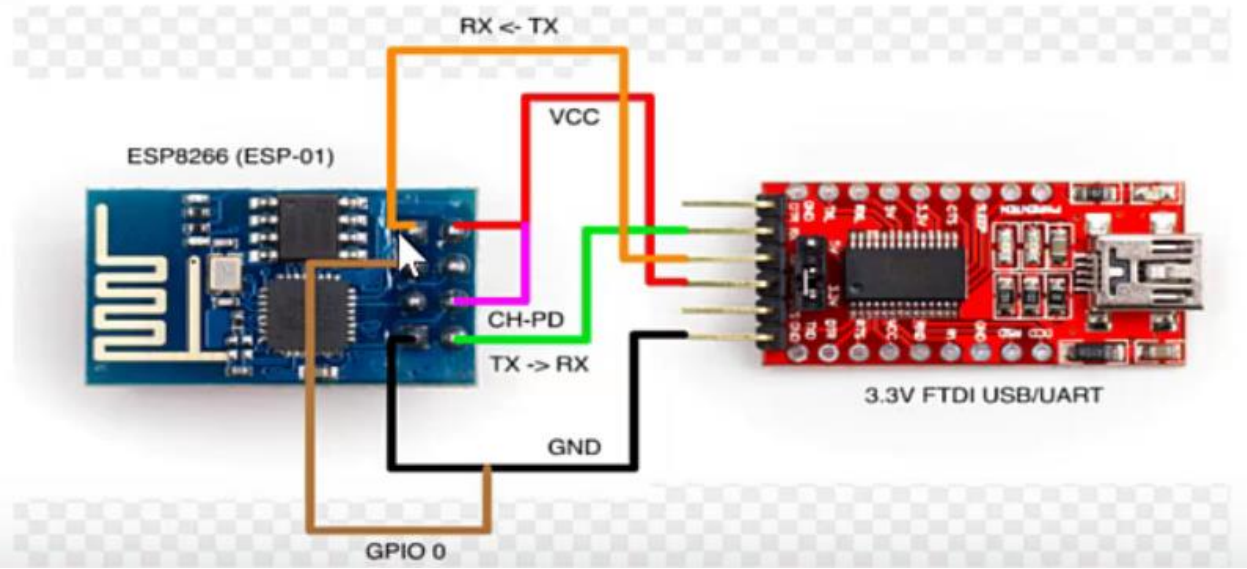
```
              _delay_ms(5000);
              printf("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n"); //start the
connection to the cloud

              //set length of data to be sent
              _delay_ms(5000);
              getreading();
              printf("AT+CIPSEND=104\r\n");
              printf("GET
https://api.thingspeak.com/update?api_key=thingspeakkey&field1=0%05u&field2=%05u&field3=%
05u\r\n", red, green, blue);
              _delay_ms(3000);

       }
}

void getreading()
{

       uint8_t redH, redL;
       uint8_t greenH, greenL;
       uint8_t blueH, blueL;

       //red
       i2c_readReg(APDS9960_WRITE, APDS9960_RDATAH, &redH, 1);
       i2c_readReg(APDS9960_WRITE, APDS9960_RDATAL, &redL, 1);

       //green
       i2c_readReg(APDS9960_WRITE, APDS9960_GDATAH, &greenH, 1);
       i2c_readReg(APDS9960_WRITE, APDS9960_GDATAL, &greenL, 1);

       //blue
       i2c_readReg(APDS9960_WRITE, APDS9960_BDATAH, &blueH, 1);
       i2c_readReg(APDS9960_WRITE, APDS9960_BDATAL, &blueL, 1);

       red = (redH << 8) | redL;
       green = (greenH << 8) | greenL;
       blue = (blueH << 8) | blueL;

       // THRESHOLD
       if (red > 255)
              red = 255;

       if (green > 255)
              green = 255;

       if (blue > 255)
              blue = 255;
}

void APDS_init(void)
{
       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;
```

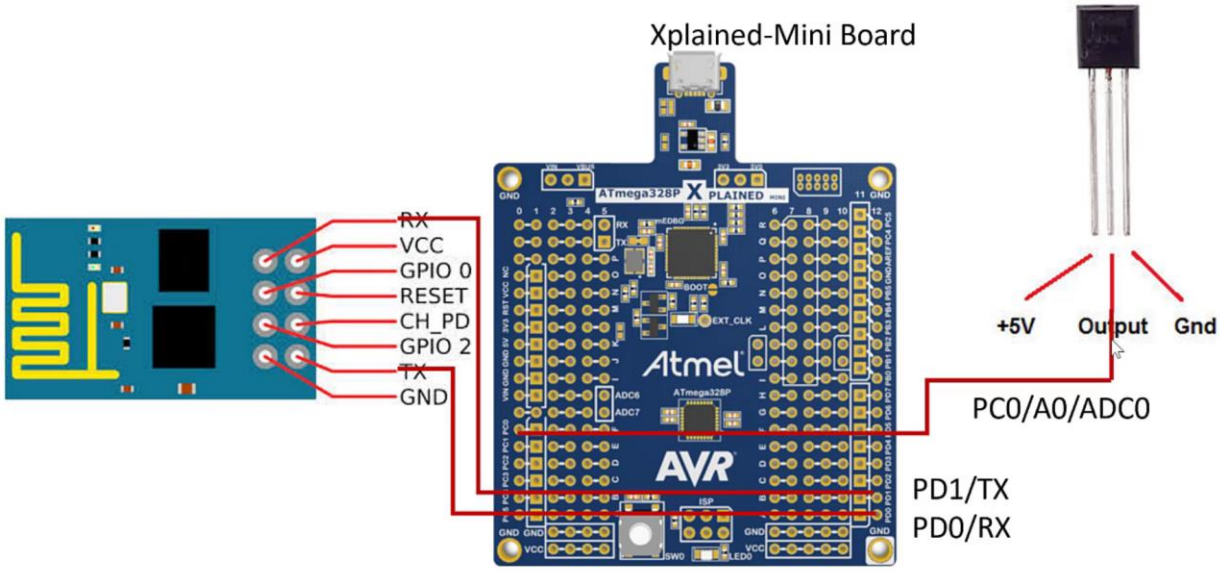
```
	i2c_writeReg(APDS9960_WRITE, APDS9960_ENABLE, &setup, 1);
	setup = DEFAULT_ATIME;
	i2c_writeReg(APDS9960_WRITE, APDS9960_ATIME, &setup, 1);
	setup = DEFAULT_WTIME;
	i2c_writeReg(APDS9960_WRITE, APDS9960_WTIME, &setup, 1);
	setup = DEFAULT_PROX_PPULSE;
	i2c_writeReg(APDS9960_WRITE, APDS9960_PPULSE, &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);
}

void USART_putstring(char *StringPtr)
{
	while ((*StringPtr != '\0'))
	{
		while (!(UCSR0A & (1 << UDRE0)));
		UDR0 = *StringPtr; //UDR0 register grabs the value given from the parameter
		StringPtr++;
	}
}

void UART_init(void)
{
	uint16_t baud_rate = UBRREQ;
	UBRR0H = baud_rate >> 8;
	UBRR0L = baud_rate & 0xFF;

	UCSR0B = ( 1 <<RXEN0)|( 1 <<TXEN0);	//enable receiver and transmitter
	UCSR0C = (3 <<UCSZ00);	// Set frame format: 8data, 1stop bit
}

int uart_putchar(char c, FILE *stream)
{
	while ( !( UCSR0A & ( 1 <<UDRE0)) ); //wait until buffer is empty
	UDR0 = c; //data goes into buffer
	return 0;
}
```

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

N/A

4. SCHEMATICS

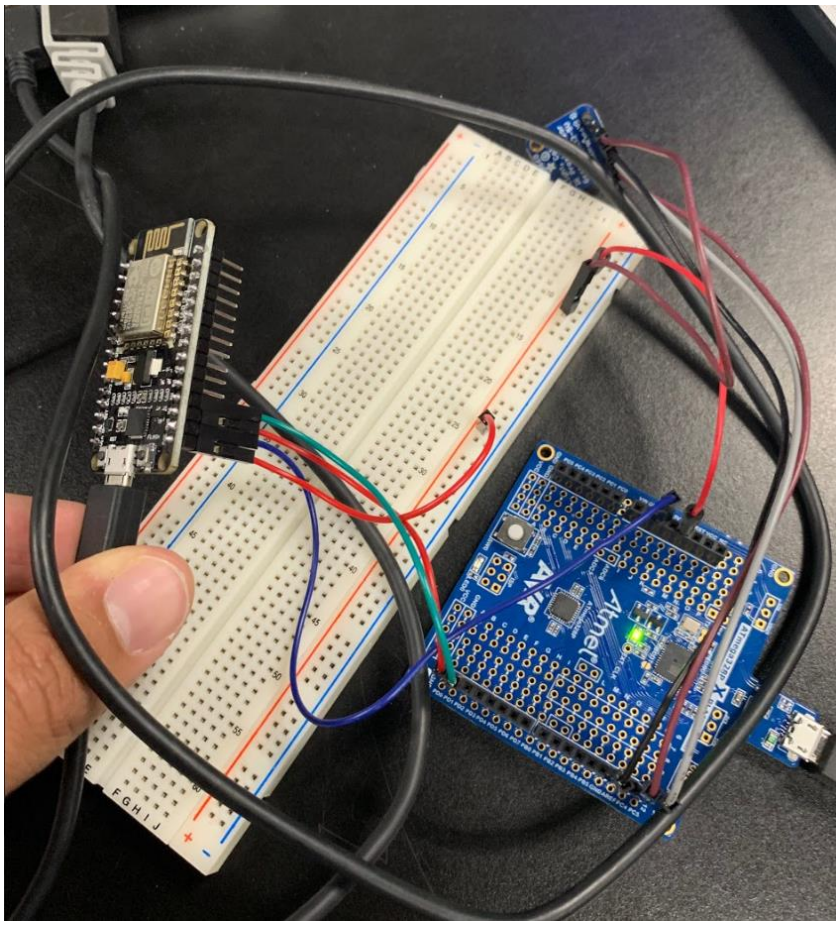




5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

N/A

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)



7. VIDEO LINKS OF EACH DEMO

https://youtu.be/J3iwdaao_Ws

8. GITHUB LINK OF THIS DA

https://github.com/tylergardenhire/submission_projects.git

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

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