ENT269

Embedded System and Interfacing

Laboratory 3: Serial Communication

Name:

Matrix No.:

Laboratory Group:

Date:

Lab Demonstration

Task 1

Demo	Verification
	Time :
	Sign :

Task 2.1

Demo	Verification
	Time :
	Sign :

Task 2.2

Demo	Verification
	Time :
	Sign :

OBJECTIVES

1. To understand serial communication in PIC18.
2. To program the serial in C18 programming language.
3. To demonstrate serial communication program in hardware application.

EQUIPMENT/COMPONENTS

1. PIC Microcontroller (SK40C) – 1 Unit
2. ESP8266 Module – 1 Unit
3. Resistor 1kΩ – 1 Unit
4. Resistor 2.2kΩ – 1 Unit

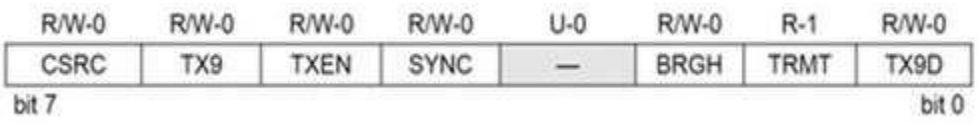
INTRODUCTION

USART (Universal Synchronous Asynchronous Receiver Transmitter) located within the PIC. It is a universal communication component (Synchronous/Asynchronous), which can be used as transmitter or as receiver. When using the synchronous communication – the information is transmitted from the transmitter to the receiver: in sequence, bit after bit, with fixed baud rate and the clock frequency is transmitted along with the bits.That means that the transmitter and the receiver are synchronized between them by the same clock frequency. The clock frequency can be transmitted along with the information, while it is encoded in the information itself, or in many cases there is an additional wire for the clock. This type of communication is faster compare to the asynchronous communication since it is "constantly transmitting" the information, with no stops.

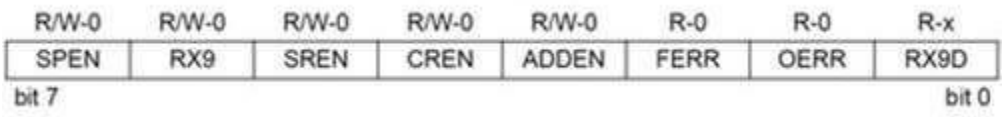
When using the asynchronous communication - the transmitter and the receiver refraining to transmit long sequences of bits because there isn't a full synchronization between the transmitter, that sends the data, and the receiver, that receives the data.

To enable the serial communication with PIC micro the different parameters need to set within two registers:

TXSTA – Transmit Status and Control Register



RCSTA – Receive Status and Control Register



ACTIVITY/TASK

PREPARE a structured program in C language to the tasks given below:

Task 1:

Continuously send "AT\r\n" command to ESP8266 (embedded in ESP12 board) every 5 second. The ESP8266 should reply "OK" if everything is good. Otherwise it will reply "ERROR". If the reply from ESP8266 is "OK", you have to toggle RB6; otherwise, toggle RB7. Refer Figure 1.0 on how to connect PIC board and ESP8266 breakout board.

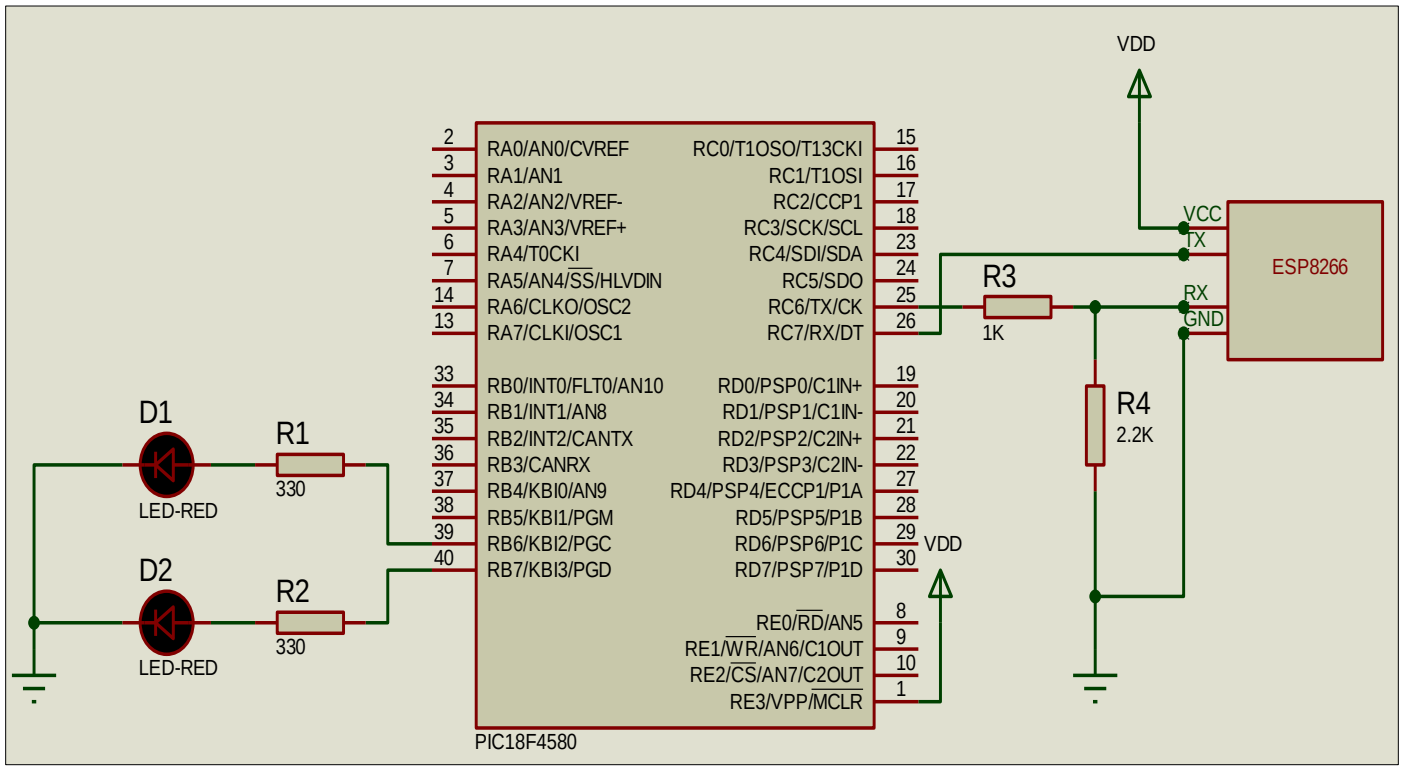


Figure 1.0

Task 2:

Task 2.1: Write a PIC C18 program to access ESP8266 using AT command and send the data to Thingspeak server. Please refer an Appendix A as your reference.

Task 2.2: Write a PIC C18 program to access ESP8266 using AT command and send the data (randomly) to Thingspeak server. Please refer an Appendix A and Appendix B as your reference.

PROGRAM CODE:

DISCUSSION:

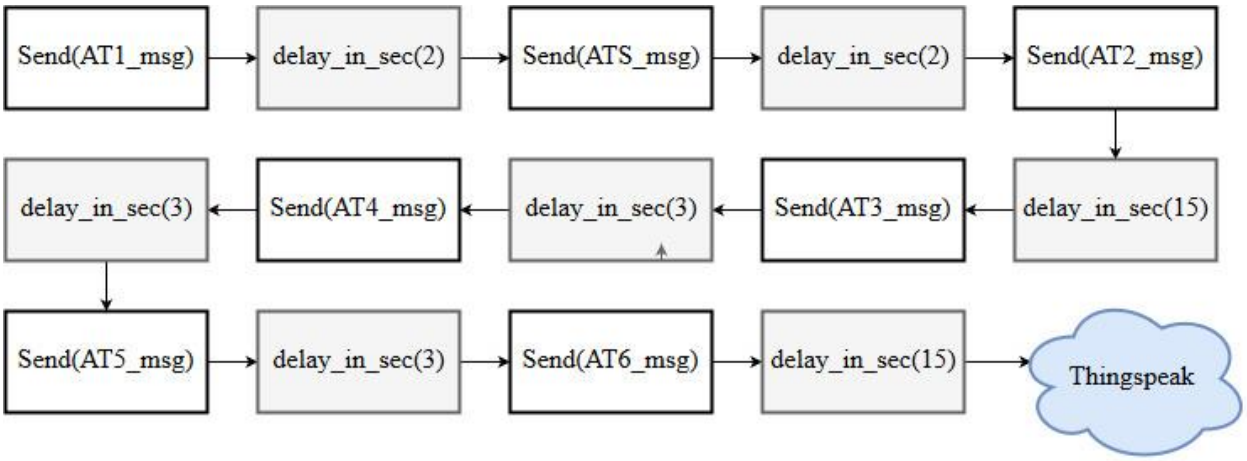
CONCLUSION:

Appendix A:

```
unsigned char AT1_msg[]="AT\r\n";                                   //Hi ESP8266
unsigned char ATS_msg[]="AT+CWMODE=3\r\n";                          //AP + Station mode
unsigned char AT2_msg[]="AT+CWJAP=\"SSID\",\"PASSWORD\"\r\n"; //Change the SSID with your wifi ID
                                                                    //PASSWORD with your wifi password
unsigned char AT3_msg[]="AT+CIPMUX=1\r\n";
unsigned char AT4_msg[]="AT+CIPSTART=4,\"TCP\",\"184.106.153.149\",80\r\n";
unsigned char AT5_msg[]="AT+CIPSEND=4,44\r\n";
unsigned char AT6_msg[]="GET /update?key=KSY4UIPM8ERGDZPW&field1=30\r\n\r\n";
```

Change your API Keys

Data to Transmit to Cloud (field1)



Appendix B:

Please include the library below:

```
#include <stdio.h>
#include <stdlib.h>
```

Declaration:

```
unsigned char x;
unsigned char y;
unsigned char buf1[50];
```

Data to Send:

```
x = rand() % 50;          //Random between 0 to 50
itoa(x,buf1);
send(buf1);               //field1 data
```