# CMPE 443 PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

## LAB #008

## "Serial Communication - UART"

## Motivation

In this experiment, we will study getting information from a remote center over a wireless access point. Serial Communication provides connectivity of an embedded system to the other computing systems. Universal Asynchronous Transmitter and Receiver (UART) is the most basic form of the serial communication. AT commands are used in the industry to communicate with and set up a modem. These represent commands for the Rockwell chipsets. In this experiment, we introduce the basic usage of these modules. Hence, you will learn:

- configuring UART ports of a microcontroller
- configuring serial port of a PC.
- using AT commands to control and communicate with a wireless module, ESP8266.
- displaying data on the Computer Screen.
- controlling an embedded system from a computer by entering comments from the keyboard.

## 1) Initialize Serial Communication *5 pts*

Serial communication is the common method of transmitting data between a microcontroller and a peripheral device. LPC4088 includes 5 UARTs (Universal Asynchronous Receiver and Transmitter). In order to use serial communication, P0_2 and P0_3 pins and UART0 will be used. Therefore find and write into **Library/Serial.h** file the following information:

- Write the base address of the UART0. **(*?)** ___________ *0,25 pts*
- Write the IOCON address of TX Pin. **(*?)** ___________ *0,25 pts*
- Write the IOCON address of PX Pin **(*?)** ___________ *0,25 pts*

In order to use, P0_2 and P0_3 as UART TX or RX, function of these pins should be changed. Find and write into **Library/Serial.c** file:

- Change the function of TX (P0_2) and RX (P0_3) pins for UART. **(*?)**
___________ *0,25 pts*

In order to use UART0 for serial communications, UART0 should be configured correctly. In **Library/Serial.c** file:

- Turn on UART0. **(*?)** ___________ *0,25 pts*
- Enable FIFO for UART0. **(*?)** ___________ *0,25 pts*

In order to change the baudrate of serial communication, DLM, DLL and FDR values should be correctly assigned. In order to change these values for 9600 baudrate (60 MHz PCLK) in **Library/Serial.c** file:

- Enable access to Divisor Latches. **(*?)** ___________ *0,25 pts*
- What is DIVADDVAL value? **(*?)** ___________ *0,25 pts*
- What is MULVAL value? **(*?)** ___________ *0,25 pts*
- Write correct DLM, DLL and FDR values. **(*?)** ___________ *0,5 pts*
- Disable access to Divisor Latches. **(*?)** ___________ *0,25 pts*

For Serial Communication, you will use 8-bit character transfer, 1 stop bits and even parity configuration. In **Library/Serial.c** file:

- Change LCR register value for 8-bit character transfer, 1 stop bits and Even Parity. **(*?)** ___________ *0,5 pts*

Receiving and Transmitting data for serial communication are made via interrupt. Therefore in **Library/Serial.c** file:

- Enable the Receive Data Available and THRE Interrupt. **(*?)** ___________ *0,25 pts*
- Enable UART0_IRQn Interrupt. **(*)** ___________ *0,25 pts*
- Change Priority of UART0_IRQn Interrupt to 5. **(*)** ___________ *0,25 pts*

When you enable UART0_IRQn Interrupt, UART0_IRQHandler will be called. In UART0_IRQHandler, you should separate Receive Data Available and THRE Interrupt from each other. In **UART0_IRQHandler** method:

- Change if statements for Receive Data Available interrupt and THRE interrupt. (First if statement is for Receive Data Available interrupt. Second if statement is for THRE interrupt.) **(*)** ___________ *0,25 pts*

Answer these questions:

- Which UART register contains the next received character to be read. **(?)**

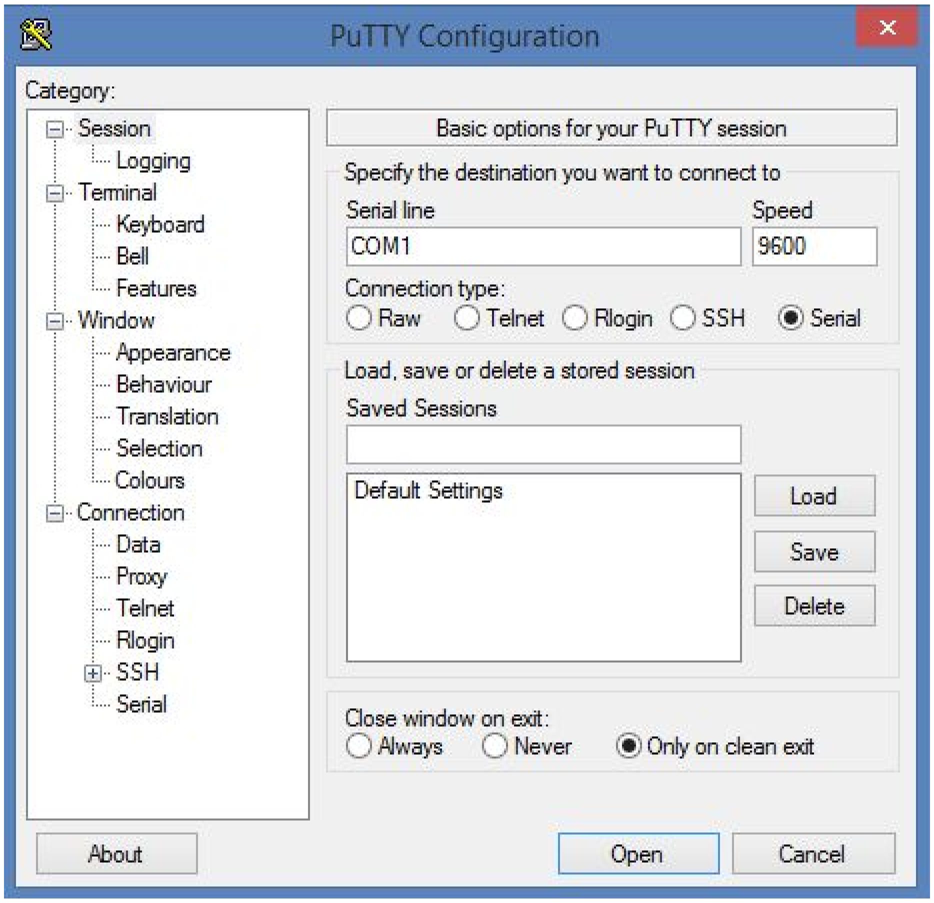
  ____________ *0,25 pts*

- In order to transmit a character via serial communication, this character should be written into a UART register. What is the name of this register? **(?)**

  ____________ *0,25 pts*

## 2) Communication with PC *1 pt*

There are a lot of Serial Communication applications for PCs. We will use PuTTy ([http://www.chiark.greenend.org.uk/~sgtatham/putty/](http://www.chiark.greenend.org.uk/~sgtatham/putty/)):



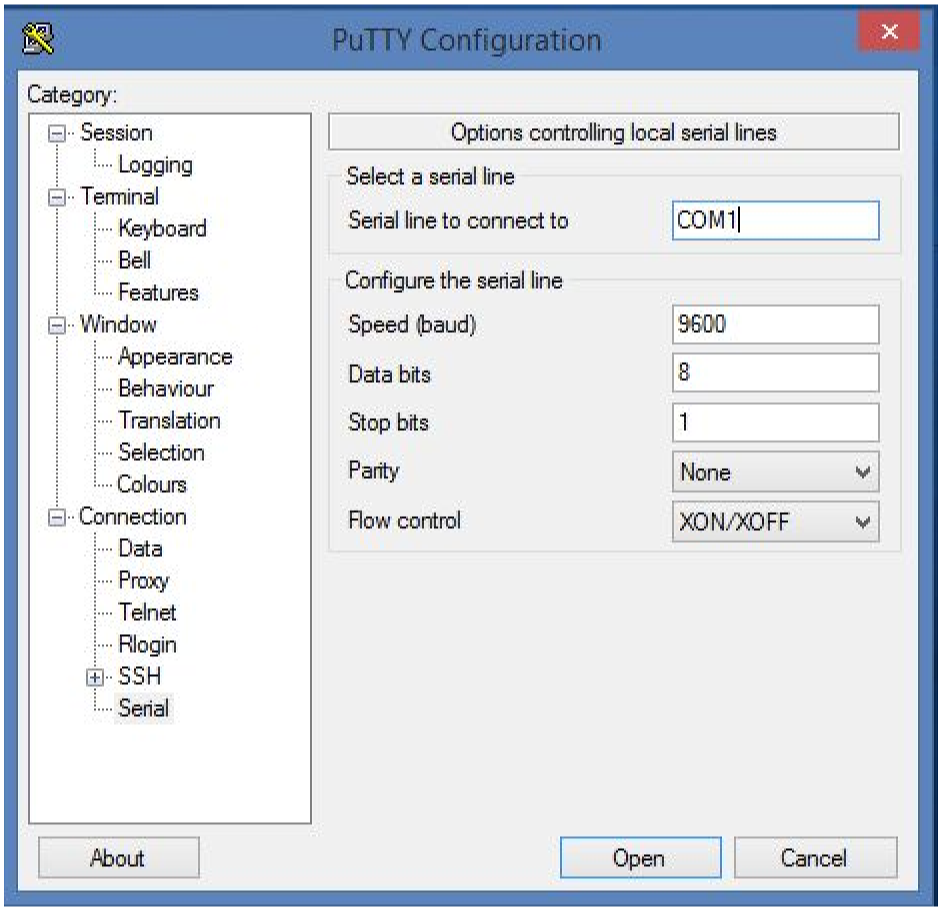
Specify the destination you want to connect to

Serial line: COM1

Speed: 9600

Connection type: Raw Telnet Rlogin SSH [✓ Serial]

Close window on exit: Always Never [✓ Only on clean exit]


PuTTY Configuration

Category:

- Session
  - Logging
- Terminal
  - Keyboard
  - Bell
  - Features
- Window
  - Appearance
  - Behaviour
  - Translation
  - Selection
  - Colours
- Connection
  - Data
  - Proxy
  - Telnet
  - Rlogin
  - SSH
  - Serial

Options controlling local serial lines

Select a serial line

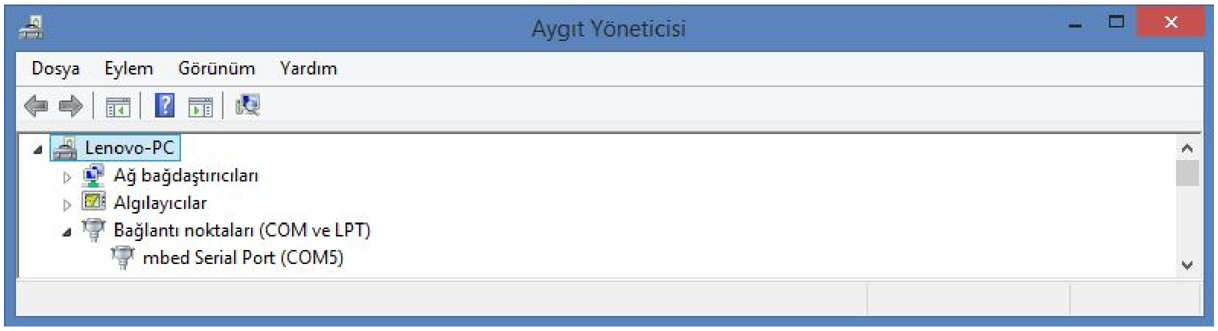
| | |
|---|---|
| Serial line to connect to | COM1 |

Configure the serial line

| | |
|---|---|
| Speed (baud) | 9600 |
| Data bits | 8 |
| Stop bits | 1 |
| Parity | None |
| Flow control | XON/XOFF |

As you see, there are 6 fields, you should fill:

- The COM name of the LPC4088.
- Speed which is the baudrate of the Serial Communication.
- Data Bit number.
- Stop Bit Number.
- Parity Type.
- Flow Control.


Aygıt Yöneticisi

- Lenovo-PC
  - Ağ bağdaştırıcıları
  - Algılayıcılar
  - Bağlantı noktaları (COM ve LPT)
    - mbed Serial Port (COM5)

Firstly, you should find the COM name by using Device Manager in Windows. In Device Manager:

- What is the COM name next to the *mbed Serial Port* in your computer? **(?)**
  ___________ *0,5 pts*

In Serial Communication, there are standard baudrates which are 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200. You configured UART0 as 9600. Also you configured Serial Communication as 8 Data bits, 1 Stop bit and Even Parity.

You can check the Serial Communication by using this code in update: *serialTransmitData = "Hello World"; Serial_WriteData(*serialTransmitData++); while(!serialTransmitCompleted);* (Do not forget to remove command before the methods in the **init** method.)

In PuTTY, in **Session** tab, choose **Serial** option, write COM and Speed, and then click to **Serial** which is in the left menu, write the configuration numbers, press **Open**:

- What is written on the PuTTY screen? **(?)** ___________ *0,5 pts*

## 3) Turning on and off the LEDs According to Data From PC *2 pts*

In this part, you will send data from PC to LPC4088 and according to the received data corresponding LED will be turned on:

- E - Turn on LED1
- N - Turn on LED2
- W -Turn on LED3
- S - Turn on LED4
- Other values - Turn off all the LEDs

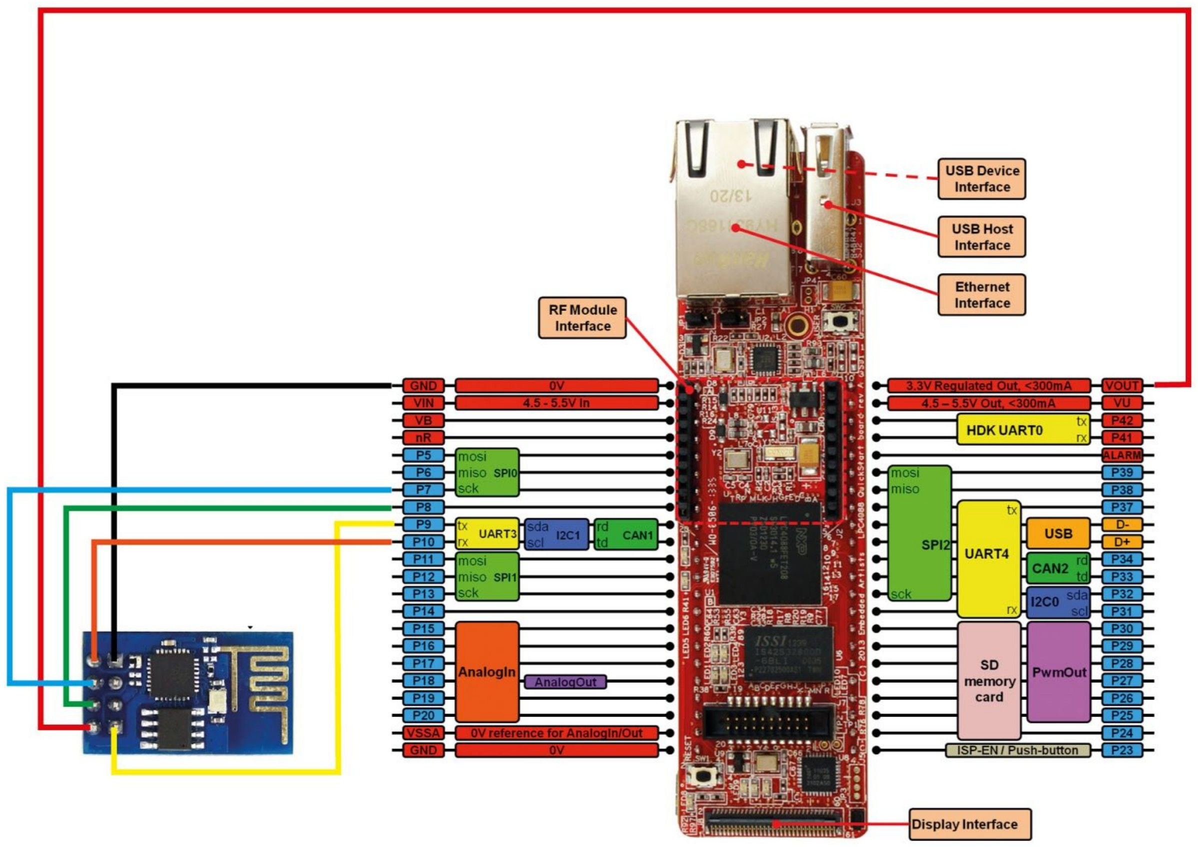
(Hint: You can use *serialReceivedCharacter* for detecting which character is received.)

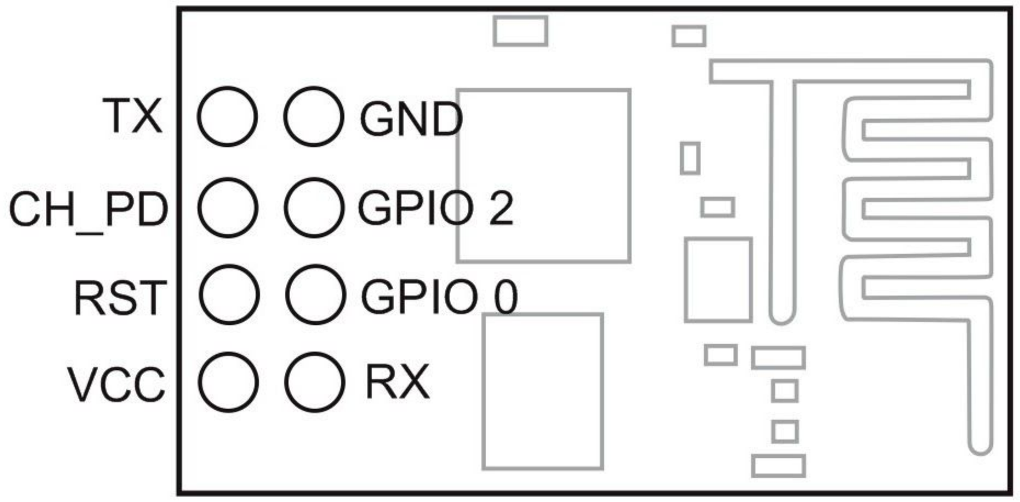
## 4) Power Consumption *0.5 pts*

In this part, you need measure the power consumption of the current system.

- Current value. ___________ *0.5 pts*

## 5) Connecting ESP8266 to LPC4088 *1 pt*





| Esp8266 Pin | LPC4088 Pin |
|---|---|
| VCC | 3.3V |
| CH_PD | P7 (P1_20) |
| RST | P8 (P0_21) |
| TX | P10 (P0_1) |
| RX | P9 (P0_0) |
| GND | GND |

## 6) Initializing ESP8266 *3 pts*

ESP8266 WiFi Module is working with 115200 baudrate serial communication. In order to change the baudrate of serial communication, DLM, DLL and FDR values should be correctly assigned. In order to change these values for 115200 baudrate (60 MHz PCLK) in **Library/ESP8266.c** file:

- What is DIVADDVAL value? **(*?)** ____________ *0,5 pts*
- What is MULVAL value? **(*?)** ____________ *0,5 pts*
- Write correct DLM, DLL and FDR values. **(*?)** ____________ *1 pt*

ESP8266 WiFi Module communication should not be lost. The received data from this module should be added to buffer immediately. Therefore in **Library/ESP8266.c** file:

- Enable the Receive Data Available Interrupt. **(*?)** ____________ *0,5 pts*
- Enable UART3_IRQn Interrupt. **(*?)** ____________ *0,5 pts*

## 7) Receiving Data from ESP8266 *5 pts*

In order to use ESP8266 WiFi Module for this lab, you should connect ESP8266 WiFi Module the network. In **init** method (In this section only write the AT Commands to the paper):

- Connect the WiFi SSID = **HWLAB** and Password = **12345678** (You can send AT Commands to ESP8266 with *ESP8266_sendCommand*) (Do not forget to use *\r\n* at the end of each command) **(*?)** ____________ *0,5 pts*
- Whenever you use *ESP8266_sendCommand*, wait until ESP8266 response end (You can use *ESP8266_waitResponseEnd*) **(*?)** ____________ *0,5 pts*
- Whenever you use *ESP8266_sendCommand*, send the esp8266Response to the PC via Serial Communication. **(*?)** ____________ *0,5 pts*

- Get IP Address of ESP8266 and send it to the PC. (You do not need to parse the information.) **(?)** ____________ *1 pt*

- Change WiFi Mode to Station Mode and send response to the PC. **(*?)** ____________ *0,5 pts*

In **update** method:

- Connect 192.168.0.100 IP address and 8080 port via TCP. **(*?)**____________ *0,5 pts*
- Send "GET /HWLAB_IoT/GetInformation?ID=? HTTP/1.0\r\n\r\n" packet. (? should be changed as the PC number) (Firstly, you should send the length of packet, after response return OK, you should send the packet. **(*?)** ____________ *0,5 pts*

http://192.168.0.100:8080/HWLAB_IoT/GetInformation?ID= (If your computer ID is 1 the url will be http://192.168.0.100:8080/HWLAB_IoT/GetInformation?ID=1) This address will return a code like ID01W24E20. The first 4 characters for the IDs, if you send ID=1, the data will be started with ID01, if ID=12, starts with ID12. In the data W, E, S and N characters can occur, when:

- E occurs - Turn on LED1
- N occurs - Turn on LED2
- W occurs -Turn on LED3
- S occurs - Turn on LED4

If E and N occurs in the same code, LED1 and LED2 will be turned on at the same time.

(Hint: You can use *strlen* function for calculating the length of the string)
(Hint: You can use *strstr* function for getting the pointer of the first occurrence of the substring)

## 8) Reducing Power Consumption *2.5 pts*

In this part, you need reduce the power consumption of the system. You should write additional codes for reducing the power consumption while getting the same output and explain why it is reduce the power consumption.

| | | |
|---|---|---|
| - Current before and after reducing. | ___________ | *0.5 pts* |
| - What changes after the ESP8266 is added to the system? | ___________ | *0.5 pts* |
| - Power before and after reducing. | ___________ | *0.5 pts* |
| - Explain your method. | ___________ | *1 pt* |

(Hint: You can look at the power saving instructions in the Cortex™-M4 Instructions and you ESP8266 AT Instruction Set)
(Bonus: The group measures the minimum power will get +5 points.)