EMBEDDED SYSTEMS LABORATORY

LABORATORY MANUAL

Serial Communication with the MSP432

AY 22/23

OBJECTIVES

- Configure and implement serial communications to interact with MSP432
- Setup the UART to communicate with the ESP01 WiFi module

EQUIPMENT

- Computer or laptop that supports Code Composer Studio (CCS) 9.3 IDE
- Texas Instrument MSP-EXP432P401R LaunchPad Development Kit
- ESP01 WiFi Module
- Micro-USB Cable

NOTE:

- Only students wearing fully covered shoes are allowed in the lab due to safety.
- Bring your laptop with Code Composer Studio installed.
- For your understanding and better quiz preparation, note the given tasks, especially questions or unexpected code behaviour.

Introduction

Embedded systems can communicate with other devices in many ways in what is called digital communication. Typically, serial communication protocols are deployed due to the reduced pin count required as compared to parallel communications.

Serial communications can be divided into two categories, synchronous and asynchronous. Synchronous serial communication dedicates a communication channel for a clock signal that provides the timing between the transmitter and receiver devices. While the asynchronous serial communication does not utilise or require a clock signal. There are several types of both synchronous and asynchronous serial communication that were discussed in the lecture. The MSP432 microcontroller offers three different types of serial communications, SPI, IIC and UART. In this laboratory session, the Universal Asynchronous Receive and Transmit (UART) protocol will be deployed.

As the name implies, the UART does not use a clock (asynchronous) but just an agreed-upon baud rate that is usually very slow. The simplicity of this serial communication has ensured its application for a very long time in embedded applications. Another term, RS232, is often used interchangeably as well, but technically describes the difference in voltage and connectors standard while having the exact same protocol (signalling).

In this lab session, we shall learn how to use the UART module in the MSP432 microcontroller for asynchronous serial communication. A simple transmitter and receiver module in which the MSP432 will exchange 1-byte messages with the PC. The MSP432 microcontroller will trigger different LEDs depending on the received command.

Lastly, we will develop code that incorporates UART to communicate between the **MSP432P401R** and the **ESP01 WiFi** module.

Introduction to UART

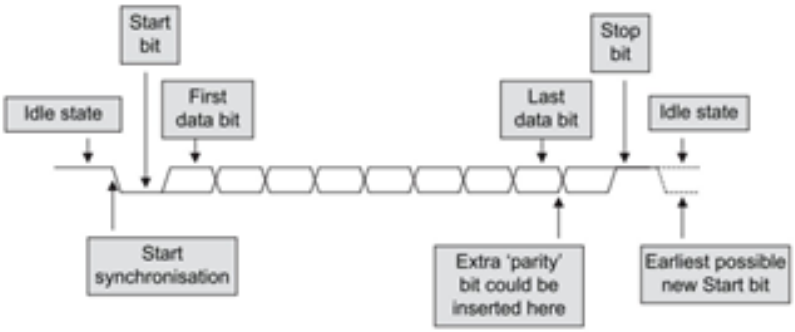

Figure 1: UART Packet Structure.

In UART protocol, the sender and the receiver have to agree on a transmission rate and packet format before any communication can happen. The data is transmitted in packets of 6 to 9 bits per packet. Additionally, a packet consists of a start bit, one or two stop bits and optionally parity bits as shown in Figure 1.

Task 1: Communicate with PC via UART

In this task, we shall develop a simple code to configure the UART and print a statement through it. Printing statement to the console is useful during code debugging. There are a few steps that need to be taken to achieve this.

1. Create an instance of the structure (eUSCI_UART_Config)
2. Configure the GPIO pins for UART mode.
3. Configure the UART module (using the structure above)
4. Enable the UART module
5. Configure what triggers the UART interrupt
6. Enable the UART interrupt
7. Enable master interrupt
8. Define and write code for UART ISR

The code LAB5_UART.c illustrates how to configure the UART to be used as a print statement that can be used on the serial console (e.g. putty). Take note of the configuration and match it with your serial console configuration. For this example, do change your clock to 12Mhz (system_msp432p401r.c). UART is configured as 9600, odd parity with 1 stop bit.

Task 2: MSP432 communicate with ESP01

The project (LAB5_ESP01.zip) configures two UART modules; the serial console and the ESP01 module. The application uses the "ESP8266_Terminal" function that goes into an interactive mode where the user can key in the AT Command to the ESP01 module directly via the serial console. For this example, do note that the clock is modified within the main function to 24Mhz.

To test the entire setup, key in **AT** into the serial console followed by the enter key. If the wiring is correct and the code compiles successfully, the correct response is **OK**.

The datasheet for the module can be found here. Further details on how to utilise the library can be found in the following link. Do note that the ESP01 is the same as the ESP8266 mentioned in the link. The ESP01 is an embedded system platform (called ESP8266) that is configured and programmed to work as a WiFi module.

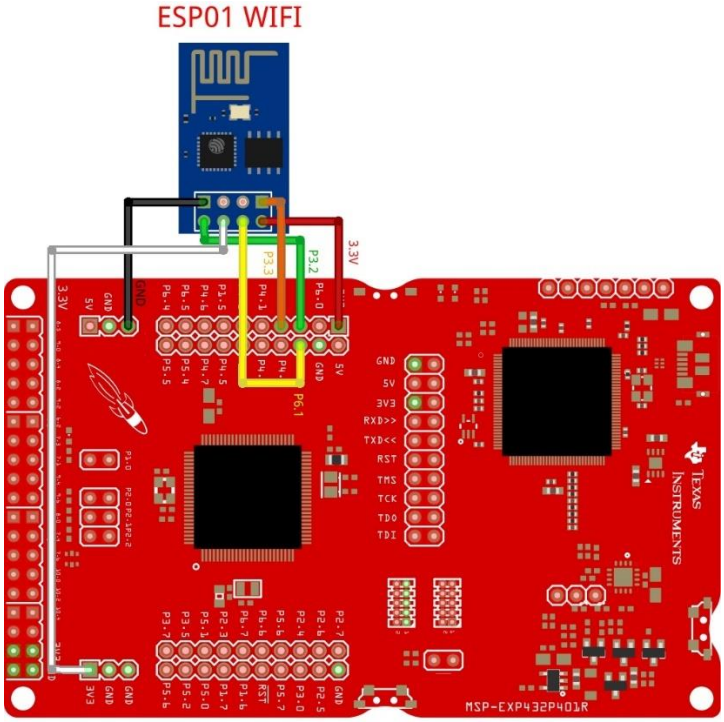

Figure 2: Connecting the ESP01 WiFi module to MSP432.

Submission

Modify the Task 1 code such that the **LED1 will turn on** when **any** of the keys from ‘a’ to ‘m’ is pressed. In addition, the **upper-caps** equivalent (‘A’ to ‘M’) of the character would be displayed on the serial console. In addition, **turn off LED1** and display the character ‘Z’ when the character ‘z’ is keyed in. The MSP432 should be **running at 3Mhz** and the UART configured to the baud rate of **115200bps**, with **one stop bit** and **odd parity**.