Exercise 2 - Serial Debugging

Original Author: Anil Agarwal

Prerequisites

Exercise 1 - Blinky

Objectives

- Learn about the IO communication protocol UART.
- Learn how to create a new Keil µVision project, select the appropriate device, configure the environment and add external source files.
- Learn how to use the USB-to-Serial adapter for UART communication between the PC and the ARM micro-controller.

What is UART?

UART stands for Universal Asynchronous Receiver/Transmitter, and it is a serial IO protocol for communication between different ICs. The word 'Asynchronous' implies that the two communication devices don't share a common (synchronized) clock. Therefore, both communicating devices must be configured for the same bit rate (also known as "baud rate").

Exercise

Step 1 - Connect the USB-to-Serial adapter

The adapter you will be using is based on the FT232R chip, which is a very common interface between UART ICs and USB devices.

1. Open EAGLE, then from the menu, open Projects ecc3852_projects lpc_serial lpc_serial.sch
2. Notice the difference between this schematic and the schematic for lpc_blinky. This schematic shows a "USB Serial" connector
3. Wire up the USB serial adapter by connecting Rx on the adapter to Tx on the LPC1114 and vice-versa. Also connect GND on the adapter to the common GND

Do NOT connect the 5V pin on the serial adapter to the breadboard's 5V power input! The serial adapter is powered through the USB port.

‘Rx’ stands for ‘Receive’ and ‘Tx’ stands for ‘Transmit’. The Rx and Tx labels on a device are from the perspective of that device. Therefore, Tx of one device needs to be connected to Rx of the other device and vice versa.

To-Do Before Next Lab

Additional Resources

Learn More About Serial Communication
Using Multiple UART Devices with the Micro-controller
Other Common IO Protocols
4. Connect the USB serial adapter to the computer through the mini-USB cable

**Step 2 - Create a new Keil project for serial debugging**

1. Open Keil Vision through the Start menu
2. From the top menu bar, select **Project** | **New µVision Project...**
3. A dialog box will open; browse to `C:\ECE395\serial` and then type the name of the project (serial). Then click `Save`.

4. When asked to select a device, change the first drop down to ‘Legacy Device Database [no RTE].’ Then type ‘LPC1114’ in the search box and select `NXP LPC1114/102` from the results. Click `OK`.
5. Click Yes when asked if you want to copy startup_lpc11xx.s to the Project folder.

6. Then from the left pane, right click on ‘Target 1’ and select Manage Project Items...
7. Under Groups, double-click on “Source Group 1” and rename it to ‘Startup’. Then click on Add Files... and select 'system_LPC11xx.c'.
The Manage Project Items window should now look like this:

The Add Files to Group 'Startup' window should now look like this:
8. Create a new group named "Serial" (using the button) and then add the files main.c, Serial.c and Retarget.c to this group.

9. Click OK to return to the main window. From the left pane, right click on 'Target 1' again, and select Options for Target 'Target 1'.
10. Go to the *Debug* tab and select the option button next to ‘Use: ULINK2/ME Cortex Debugger’.

11. Click *OK*
12. Build the project and program the microcontroller

**Step 3 - Open and configure PuTTY**

1. Open PuTTY from the Start menu
2. Open Device Manager and find the COM port for the serial device. In the screenshot shown below, this is COM4. The port may be different on your computer; use the port that is shown on your computer.

Note: It is very important to configure PuTTY with the correct COM port; using the wrong COM port may damage other USB devices connected to the computer.
3. Return to PuTTY and configure the settings for the serial port. Use the COM port from your Device Manager, but configure the speed as shown below.

![PuTTY Configuration](image)

4. Click Open.

5. Reset the microcontroller to run the program. The PuTTY terminal should start printing debug data from the microcontroller.
To-Do Before Next Lab

- Complete this exercise and ensure that you can see debug data on the PuTTY terminal.
- Read and understand the main.c and Serial.c code using the ARM micro-controller's user manual as a reference.

Additional Resources

Learn More About Serial Communication

Sparkfun has a great tutorial [here](#) that goes into the details of serial communication.

Using Multiple UART Devices with the Micro-controller

Although UART by itself doesn't define a master/slave device and there is no "Chip Select" or "Slave Select", it is possible to use a single UART port with multiple chips. One method is to use a decoder and a tri-state buffer.

Other Common IO Protocols

In week 5, you will learn about I²C and SPI, two synchronous IO protocols commonly used for IC-to-IC communication.