

ECE 218

EXERCISE 2. SERIAL INTERFACE WITH UART

This exercise will guide you through the process of creating a serial link between a computer and the PIC24 chip on the Microstick II. You will use a serial cable connected to the header on the end of the board, and the PUTTY terminal emulator on the computer to access the serial port. You will use a library of functions for the serial UART2 peripheral to write a program that communicates with a computer through the PIC24 serial interface.

Figure 1 shows the major components of the hardware: the PIC24, serial cable with FTDI interface, and computer (PC or Mac).

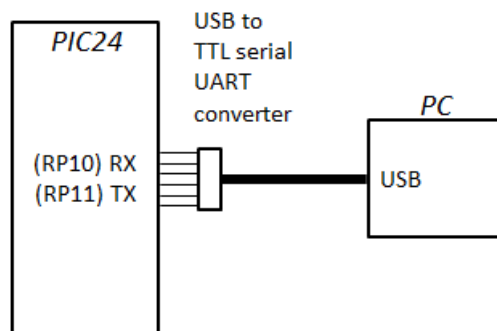


Figure 1. Block Diagram of Hardware

HARDWARE SETUP

Connect the 6-pin, 90° header on the Microstick II board to the serial cable in the proper orientation. The pins on the terminal are color coded and defined as shown in Figure 4.1 from the cable datasheet. On the Microstick II, pin 1 (GND) is indicated with a square pad and white pattern around the pin.

Once in place, connect the FTDI serial cable to a free USB port on the computer.

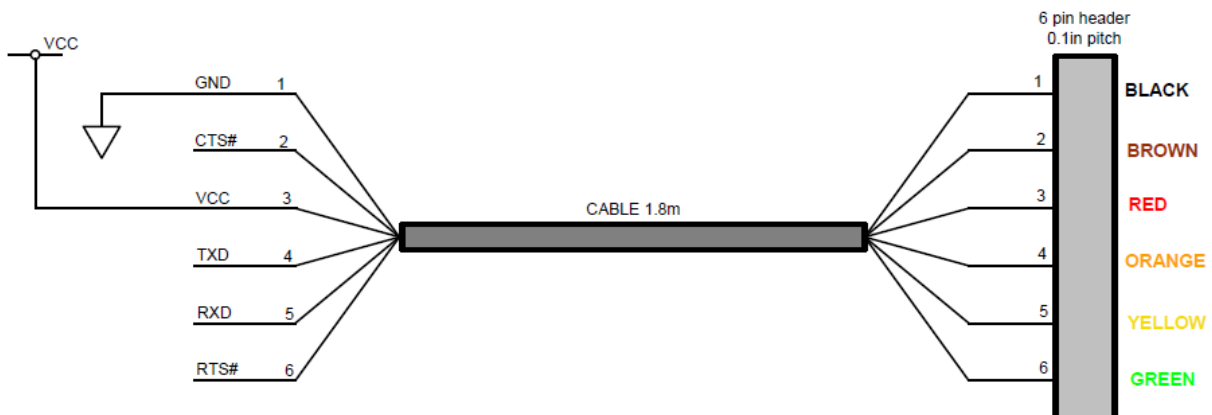


Figure 4.1 TTL-232R-5V and TTL-232R-3V3, 6 Way Header Pin Out

SOFTWARE DEVELOPMENT

In this exercise we will use the code library that comes with the textbook¹ which includes files with important parameters and functions for serial communication between the PIC24 and the PC. We will use the PIC24 UART1 for serial communication.

Navigate to the folder, *pic24_code_examples*, in your Exercises folder. Navigate to the following subfolder: *Exercises/pic24_code_examples/lib* and find the */include* and */src* library folders. Examine the *pic24_uart.h* and *pic24_uart.c* files and find the function for configuring UART1.

function name: __configUART1

parameter: __unit32_t u32_baudrate

Creating New Project

For your convenience, the list of steps to start a new project are repeated below.

Create a new project (File-New Project).

- Choose category “Microchip Embedded”, and Project type “Standalone Project”.

In the next screen, you will be asked to select a microcontroller family, and a particular device.

- Family: 16-bit MCUs (PIC24)
- Device: PIC24HJ128GP502



For hardware tools, you will choose the Starter Kit (PKOB) - Microstick II starter kit. Note that it will not show up in this window if it is not connected to the USB port.

For the compiler, you will choose the XC16.


In the next window you will select a project name, location, and folder.

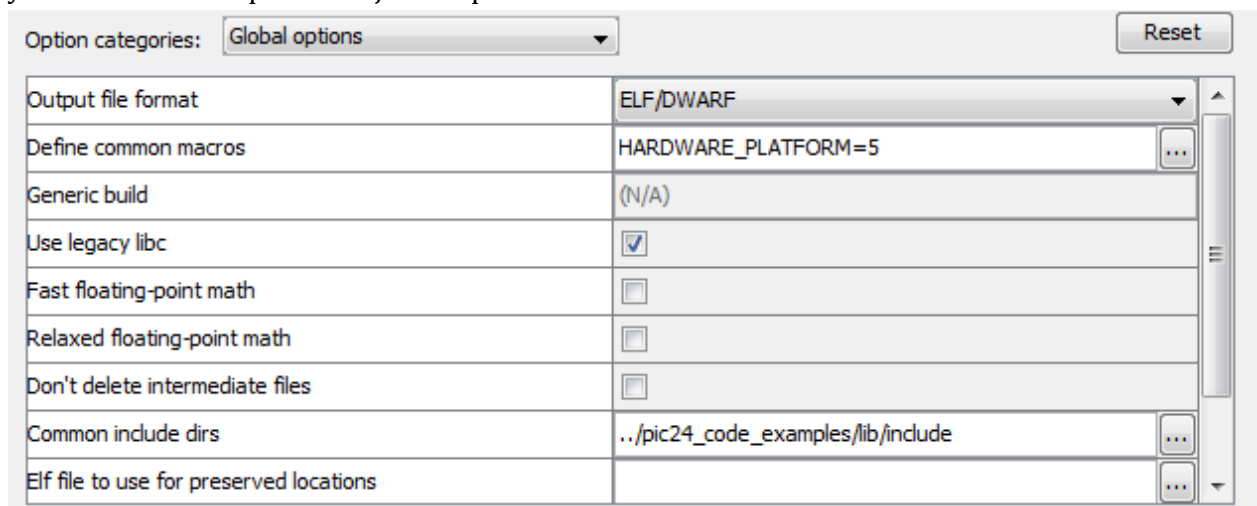
- Project name: *serial*
- Project location: Browse to your flashdrive folder *Exercises*.

Extra steps for textbook library are needed to specify the Microstick II platform and to set the path to the installed library location:

1. Once the project is created, click on the **Project Properties** icon  in the dashboard window (lower left).
2. Select the XC16 (Global Options) in the Categories window.
3. Use the “Define common macros” to set `HARDWARE_PLATFORM=5` (click on  and then double-click on the “Enter text here” box and type `HARDWARE_PLATFORM=5`).

¹ *Microcontrollers: From Assembly Language to C using the PIC24 Family*, Jones, Reese, Bruce, Second Edition, Cengage Learning, 2015. ISBN-13: 978-1-305-07655-6

4. Use the “Common include dirs.” to specify the path to the downloaded library. Click on  and then browse to the library *include* folder **../pic24_code_examples/lib/include**. When you finish these steps the Project Properties window should look like this:



If so, then Apply the changes and click OK to close the window.

5. Add the library source folder to the project by right clicking on “Source Files” in the Projects window and selecting “Add Existing items from Folder”. In the pop-up window, click “Add Folder” and navigate to the **pic24_code_examples/lib/src** folder and add this to the project.

Main Program – main_serial.c

The purpose of the main program is to communicate with a terminal emulator application on the computer. The first program we will write will capture characters typed on the keyboard and echo them back to the terminal application.

We will start with template c file that will be used for all new source files. This file is on Nexus in the Resources area, and called **main_template.c**. Download this file and save it in your Exercises folder. Make a copy and name it **main_serial.c**. Add this file to your project within MPLAB X.

Starting with the template, fill in the meta data at the top (file name, author, etc) and then write code for the **main_serial.c** program. Hints follow for each section.

- a. Include files –the following include file will include all other textbook library files.
 - a. *pic24_all.h*
- b. Definitions (none needed for this program)
- c. Global functions and variables (none needed for this program)
- d. Main program:
 1. Declare one local variable (of type `uint8_t`) to hold the key character
 2. Call configuration functions used with textbook library
 - a. `configClock();` //configures the PIC24 to run with a 40MHz clock
 - b. `configUART1(230400);` //parameter is the baudrate for serial communication
 3. No port initialization needed here because the serial port initialization is done in the `configUART1()` function.

4. Call a function to send an initialization message of your choice (it should be a string, surrounded with quotes) to the screen.
5. In the main while loop, write code that forever receives a character from the keyboard and outputs the character to the terminal screen. Use the variable you declared.

The library functions that will be used to send the initialization message, and input and output the key character are declared and defined in the *pic24_serial.h* and *pic24_serial.c* library files, respectively. Documentation for these files can be found here (bookmark this location):

http://courses.ece.msstate.edu/ece3724/main_pic24/docs/index.html

Go to this link and click on the “Files” tab, and find the *pic24_serial.h* file reference. You will need to use the following functions in your program, so read about them in the documentation.

- `outString();`
- `inChar();`
- `outChar();`

Write the *main_serial.c* program and make sure it builds successfully before moving on to the next section.

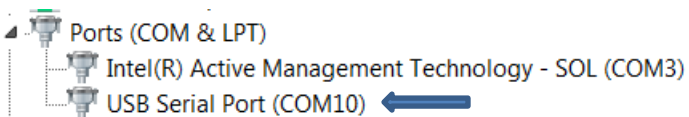
TESTING

MAC users will use a terminal window to discover the USB port attached to the serial cable, and to start a new “screen” with this terminal.

1. In a terminal, type `ls /dev/tty.*` to list all of the serial devices. One of them should be something like `/dev/tty.usbserial-FTA3...` this is your *serial_port_name*.
2. Type `screen serial_port_name 230400` to start a new terminal communicating on *serial_port_name* at 230400 baud.

PC users will use a terminal emulator program called “putty” to interact with the PIC24 through a serial port. If putty is not installed, install it (you want the SSH and Telnet client itself) on the desktop from here: <http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

To set this up you must know which USB port the serial cable is connected to. This can be determined, once the cable is plugged in, using the “Control Panel” – “Hardware and Sound” - “Device Manager” on a PC. On a Mac, you may be able to find it using the “Apple” menu – “About this Mac” – “System Report”. On a PC, the serial COM port used will be under the “Ports” section.



Once you know which COM port you are using, open putty and you will first see a configuration window where you can define your session.

In putty, the “Host Name” will be COMx, where x is the port number you discovered. The connection type is Serial, and the Speed is 230400 baud. You may want to name and save this session (something like PIC24) so that you don’t have to specify all of this the next time you run putty.

Once you have saved and opened the session, and downloaded your main serial program, you should see whatever welcome message you wrote in your initialization code, and you should be able to type some characters and see them displayed.

If you would like to tidy up the display, the following useful characters can be added to the end of your initial message string (inside the quotes) to go to the beginning of a new line:

```
\n - new line
\r - return
```

And if you really want to get fancy, you can output this string of control characters first to clear the screen.

```
"\e[1;1H\e[2J"
```

Remember that you can always start your program at the beginning once it is downloaded by pressing the reset button on the Microstick board.

Password Program

Now modify your program so that it accepts a 2-character password (that you will set in your program) and if the password is correct, outputs a string “ – Passed” and if not, outputs a string “- Failed”. Echo the characters back to the terminal as they are typed. For example, with a password of 12, the screen will look something like this after testing:

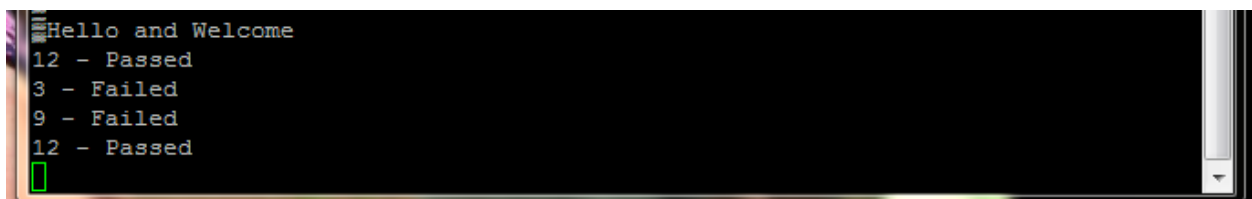


Figure 2. Screen Shot of Sample Password Program Output

Hint: The compiler will convert things in single quotes (like ‘1’) to 8-bit characters, and things in double quotes (like “ – Passed”) to strings.

The exercise is completed when you have commented your code completely, demonstrated this last program to your instructor, and uploaded your final C code to Nexus.

SAVE YOUR WORK!