Microprocessor-Based Systems (E155)

D. Money Harris **Fall 2009**

Lab 6: Wireless Communication

Requirements

Build a wireless calculator. Use a PIC linked by Bluetooth to a PC. The PIC should perform the computations, while the PC provides the keyboard and display. The calculator should be capable at least of addition of 3-digit decimal numbers. Write the PIC code in C. Use a BlueSMiRF wireless module connected to the PIC's SCI port and a Bluetooth dongle connected to the PC's USB port. Use HyperTerm as the PC terminal client.

For example, the user should be able to type

173 + 349

into HyperTerm, then press Enter and observe

522

on the HyperTerm console.

Alternatively, you may build an RPN calculator that accepts inputs such as

173 349 +

C Compiler

To learn to use the C compiler, examine the Hitchiker's Guide document handed out in class. Also, browse the Getting Started and Libraries manuals for the C18 compiler (on the web site).

The C compiler generates a .lis file with the assembly and machine language code that your program compiles into. Open the file (from the File menu) and look at it to see how the code generation works. You should understand the assembly language generated from your C code. If any of your C code produces unusually lengthy or ugly assembly language, consider simplifying.

Bluetooth Link

The Bluetooth link consists of a USB Bluetooth dongle on the PC and the BlueSMiRF module on your breadboard. The dongle emulates a serial port on the PC and the BlueSMiRF uses a serial link to the PIC.

Insert the USB Bluetooth module to a USB port on your PC. Windows XP should automatically recognize it.

Mount your BlueSMiRF module on your breadboard. The pinout is labeled on the underside of the module. Connect 3.3 V power and ground, and connect CTS to RTS so that the module performs its own handshaking. Once the module is powered, you should see a blinking red STAT LED indicating that the module is looking for a connection.

Now, double click on the Bluetooth icon in your system tray and add the BlueSMiRF device. (The device should read: "FireflyXXXX", where XXXX are the last four hexadecimal digits of the MAC number printed on the BlueSMiRF; be sure to talk to the correct device if several are in use in the lab.) The default passkey for your module should be '1234'. Do not change this passkey! Make note of the output COM port to which the device is assigned (e.g. COM11).

HyperTerm is a terminal client that is often used to run a remote session on a distant computer. For our purposes, we will be using HyperTerm to communicate with our BlueSMiRF Bluetooth module over the COM port mentioned before. The COM port uses the following settings:

Speed:	9600 baud
Data bits:	8
Stop bits:	1
Parity:	none
Flow control:	none

HyperTerm automatically recognizes these values, so you do not need to change them.

HyperTerm can be found under Accessories • Communications from the Windows Start Menu. Under File • Properties • Settings, choose ASCII Setup and check "Echo typed characters locally" so that the keys you press appear in the terminal as you type them.

When you begin typing in HyperTerm, the red status LED on the BlueSMiRF should stop blinking and the green connection LED should turn ON.

The BlueSMiRF connects to the PIC over another serial link running at 3.3 V (not the RS-232 standard, but more convenient). The BlueSMiRF expects the following settings:

Speed:	115.2k baud
Data bits:	8

Stop bits: 1
Parity: none
Flow control: none

On the PIC, the USART pins are RC6 and RC7 for TX and RX respectively. They should connect to the opposite pin on the BlueSMiRF module (TX to RX, and vice versa.) Configure the PIC USART to have the correct settings.

Hints

- You may want to configure MPLAB to automatically program the PIC after a successful build. To do this, choose Debugger • Settings • Program and check "Program after successful build."

- You may also want to configure MPLAB to begin debugging at the start of your C `main()` function rather than at the start of the initialization code. To do this, choose Configure • Settings • Debugger and check "Reset device to the beginning of main function."

- Once the serial port is initialized, you can print to the HyperTerm console over the serial port using the `printf` statement in the `stdio` library. The C18 compiler generates a warning on the format string in the printf statement, but (unlike most warnings), it can be ignored.

- The Enter key corresponds ot a carriage return character called `'\r'` in C (ASCII code 0x0D).

- If you want to advance to the next line when printing, send both the `'\n '` and `'\r '` (new line and carriage return) characters.

- You may wish to build a set of functions to receive data from the serial port. The solutions used the following functions:

```
char getcharserial(void)
// waits to receive a character from the serial port, then returns it

getstrserial(char *buf)
// receives a string over the serial port by calling getcharserial
// until a carriage return is received
```

- If you are having trouble determining if you are receiving data from the serial port, consider writing each character to the LEDs as it comes in.

Credits

This lab was originally developed in 2008 by Sam Gordon `09 and Trevor Ashley `09.

What to Turn In

When you are done, have your lab checked off by the instructor. You should thoroughly understand how it works and what would happen if any changes were made. You should also understand the assembly code produced by the C compiler. Turn in your lab writeup including the following information:

- C code for the lab
- Schematics of the breadboarded circuit
- How many hours did you spend on the lab? This will not count toward your grade.