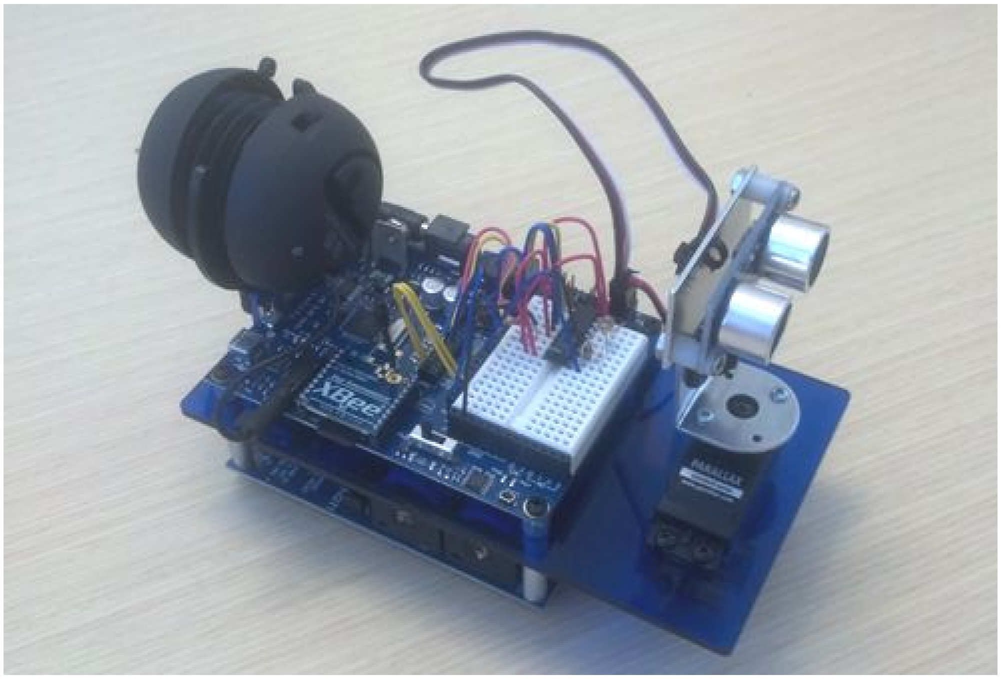
Spin Project: XBee® 802.15.4 to PC Demo

LEVEL: Intermediate
SKILLS REQUIRED: Spin Programming, circuit building.
HOURS TO COMPLETE: 2



The XBee 802.15.4 to PC Demo provides an example of communication between any Series 1 XBee Module and the PC, using the XStick 802.15.4 (Parallax #28186, discontinued). This demo is a modified version of the RN-42 Bluetooth to PC Demo using the same components except the radio transceiver. The RN-42 to PC project is also included in the download zip for this project.

The demo provides a two-way communication channel for a remote device which has two functions: one is to send system telemetry from on-board sensors, the other is to receive and act upon known commands from a remote serial terminal or command interface; in this case a PC.

While this demo application is small, it represents a scalable real-world application where you might be receiving weather or sensor telemetry from a farm or industrial control system and need to be able to then send commands to such a system, perhaps to open or close a valve in response to data received.

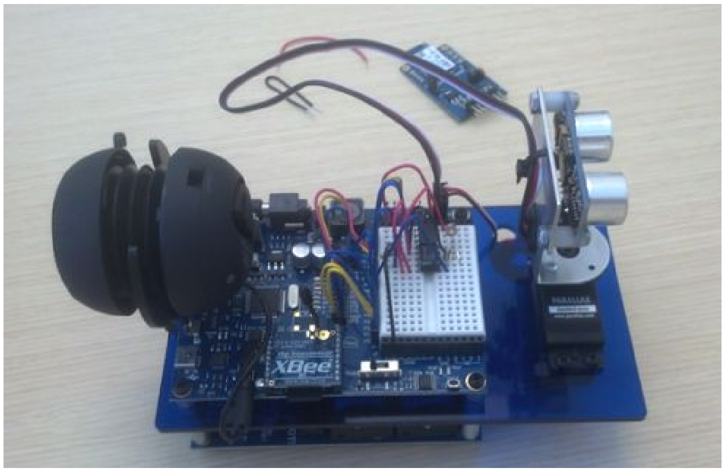
What's Required:

- (1) Propeller Board of Education (#32900, discontinued)
- (1) Any Series 1 XBee Module (#32405 used in this demo, discontinued)
- (1) XStick 802.15.4 (#28186, discontinued)
- (1) ADC0831 8-bit A/D Converter (#ADC0831)
- (1) DS1620 Digital Thermometer (#604-00002, discontinued)
- (1) PING))) Ultrasonic Sensor + Mounting Bracket (#910-28015A)
- (1) Photoresistor (#350-00009, discontinued)
- (1) 10K ohm, ¼ W Resistor (#150-01030)
- (1) 4.7K ohm, ¼ W Resistor (#150-04720)
- (1) Li-ion Power Pack Charger (#28986)
- (2) High-Capacity Li-ion Cell (#28987)
- (1) Veho 360 Speaker (#900-00018)
- (1) Veho Speaker Stand for Propeller BOE (#725-32900, discontinued)
- (1) 2 GB microSD Card (#32319)
- Connecting wires
- USB Adapter for connecting microSD card to PC and load WAV files
- Mounting plate or chassis for PING))) Mounting Bracket and PCB

Note: Many of these parts are discontinued for sale or manufacture by Parallax, but may be found through other retailers, or have suitable replacements available through Parallax or other retailers.

XBee and USB Connections

Since this demo is a modified version of the RN-42 Bluetooth to PC Demo Project, please refer to that project for build details. For this demo we'll just cover the installation and connection of the XBee Module and USB interface, since that is the only difference.



- ✓ If you're converting the RN-42 Bluetooth to PC Demo, remove the RN-42 Bluetooth module from the breadboard and all the wires used to connect it.
- ✓ Install an XBee Series 1 Module in the XBee Socket on the Propeller Board of Education.

Since power is already provided to the socket you only need to connect the signal pins. In this case we're only going to connect DI and DO. You may connect RST, RTS and CTS but they won't be used in the code. P1 should connect to DI on the socket in the center of the board. P2 should connect to DO.

Now for the PC side. Shown below is the XStick 802.15.4 dongle, which resembles a USB memory stick.

- ✓ Be sure you have the latest USB drivers from FTDI installed on your PC.

If you have any Parallax USB devices you probably already have these. If not, go to the downloads section on Parallax.com and there will be a USB Drivers download link in the most frequently downloaded items at the top of the page. Insert the XStick into a USB port on your PC. Once driver installation has completed a new COM port will be assigned to the XStick.

- ✓ As you add each component to the Propeller BOE you should run the associated test code for that device. For example, once the DS1620 is connected, run the Test_DS1620.spin code.
- ✓ Once the ADC and photoresistor are connected run the Test_ADC0831.spin code.

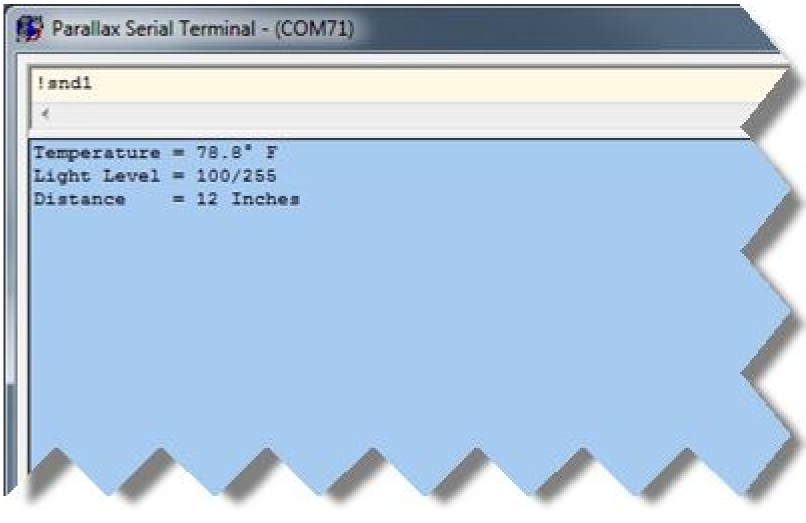
The Parallax Serial Terminal will display the results @ 9600 bps connected to the COM Port used to program the Propeller BOE. There are test programs included with this project for all sub-systems connected to the Propeller BOE.

Test the Communications

- ✓ Load the XBee PC_Prop-BoE Demo V1.0 program.

Communication with the demo unit is done by connecting via the XStick from your PC. Open the Parallax Serial Terminal (or any terminal) to the XStick COM Port created by your O/S. The port speed should be set to 9600 bps. Be sure the Propeller BOE power switch is in the second position (2) so the PING))) Sensor and servo are powered up.

Once connected the demo unit should start sending telemetry. At this point you should be seeing this data from the demo unit as shown in the example screenshot below. Temperature, light level and distance as determined by the PING))) Sensor.



```
!snd1
```

```
Temperature = 78.8° F
Light Level = 100/255
Distance    = 12 Inches
```

You can also send commands to the demo unit. The demo unit understands the following commands sent from the terminal:

- !clr - clear the terminal screen, and refresh the telemetry data
- !svo1 – move the servo to position one (90 degrees right)
- !svo2 – move the servo to position two (center position)
- !svo3 – move the servo to position three (90 degrees left)

- !snd# - where # is a number from 0-9 corresponding to a WAV file to play
- !inr0 – turn Red LED off
- !inr1 – turn Red LED on
- !ing0 – turn Green LED off
- !ing1 – turn Green LED on
- !inb0 – turn Blue LED off
- !inb1 – turn Blue LED on

Each command starts with an exclamation point. The next three letters are the command. Three letters were chosen to keep things simple for the parser code. Some commands have a parameter that follows. Once the full command is understood it is executed, no carriage return and/or linefeed are required. All commands are in lowercase.

The only command that would affect the telemetry is the 'svo' command, since the direction the PING))) Sensor faces may change the distance seen. Watch the Youtube video for the RN-42 Bluetooth to PC Demo on the Parallaxinc channel for an example of how everything works. That video shows all commands, as well as disconnecting and reconnecting.

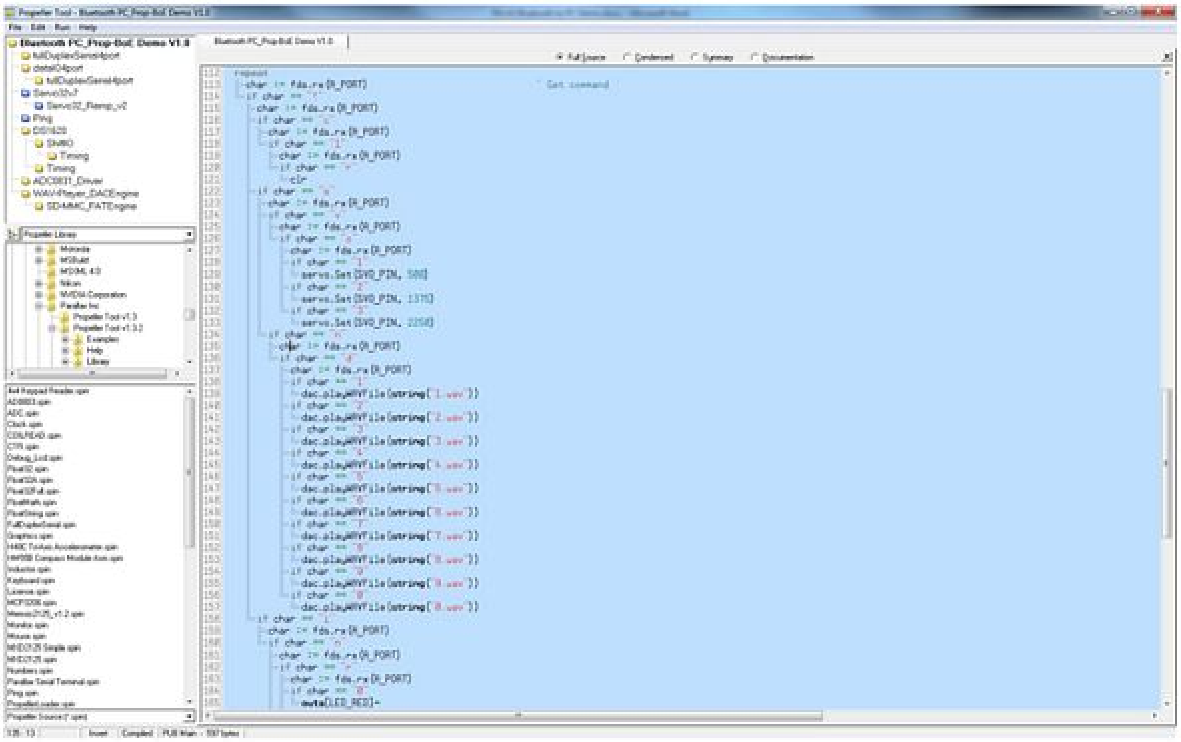
More About the XBee to PC Code

All of the objects used for the various functions were obtained from the Propeller Library (built into the IDE), the OBEX (Object Exchange), or the Parallax Forums. Everything works together with very little work to balance out resources. All required objects are included in the ZIP file you downloaded for this project.

The telemetry code is launched into a new cog. Its function is to read the sensors and send the data, wait one second and repeat. It runs completely independent of the other functions. Each of the three telemetry sections calls a method in the appropriate object setting a variable and then formatting that data out to the terminal.

The command parser evaluates incoming commands and executes them when a complete command is detected. It does this by first evaluating every character starting with the "!" character. Once this character is received the next character is checked against possible matches. If at any point the next character is not a match the whole evaluation resets. In the parser section the various levels are indented in such a way that commands are evaluated on through until completion or a wrong character.

For example, once the "!" is received the next character is compared against "c", "s" or "i". If "s" is received the next character is compared against "v" or "n". If "v" is received the next character is compared against "o". If it does not match the parser won't evaluate again until "!" is received again. This level-nested evaluation is an easy way to parse commands, even if they have similar characters at the beginning as the "svo" and "snd" commands do.



For more information about our currently offered XBee Modules please go to the Parallax website (www.parallax.com) and search XBee.

Did You Know?

This project is made up by adding code from more than seven (7) different objects and modified portions of the demos to get the desired results. To make things easier, constants for each object were placed in multiple `CON` sections, keeping them isolated. The start methods were placed as needed and initialization code was placed at the beginning to ensure everything started up properly. In spite of the apparent complexity of having over seven (7) objects running in the demo code there are only three (3) `PUB` and one (1) `PRI` method and two of these contain only one line of code.

Try This

- ✓ Change the commands used to activate various functions. You can do this by studying the structure of the parser. Indentation is important to how things are executed (and evaluated) in SPIN, so watch the level of indentation of each sub-section of the parser evaluations.
- ✓ You could also add commands, or add/change the WAV files loaded on the microSD card. The Test_microSD.spin file makes use of a command to detect when a WAV file has stopped playing. You could use this to enhance the “snd” commands in this demo.

XBee is a registered trademark of Digi International.