iRobot Create: wireless connectivity

(1) Using XBee for wireless serial communication with a computer:

I followed the example from: [http://www.ladyada.net/make/xbee/arduino.html](http://www.ladyada.net/make/xbee/arduino.html) I was able to establish a wireless serial connection between a Mac and the Arduino for sending serial data back and forth (in case of the iRobot Create) for debugging and testing sensor readings without having to chase the robot with my laptop tied to it via a USB cable. It wouldn’t work for uploading Arduino code onto the board wirelessly (I tried the Diecimilia without success).

Setting up XBee via terminal program:
Here is how I configured the ID numbers for my XBee modules on the Mac using CoolTerm ([http://coolterm.softpedia.com/](http://coolterm.softpedia.com/)) based on the PC instructions from Adafruit: [http://www.ladyada.net/make/xbee/configure.html](http://www.ladyada.net/make/xbee/configure.html)

Using a terminal, you can change the baud rate using the ATBD command with a number afterwards that selects which baud rate to use:

```
• 0 = 1200
• 1 = 2400
• 2 = 4800
• 3 = 9600
• 4 = 19200
• 5 = 38400
• 6 = 57600
• 7 = 115200
```

```text
+++ (get into AT mode)
< OK
> AT (check if xbee modem is responding)
< OK
> ATBD (get current baud rate as above)
< 3 (9600)
> ATBD 4 (set baud rate to 19200)
< OK
> ATBD (check again)
< 4
> ATWR (write the baud rate change to flash)
< OK
```
Follow these commands in CoolTerm **after** setting up CoolTerm (see next page on how to set up CoolTerm).

Here's more info on CoolTerm with a nice tutorial: [https://learn.sparkfun.com/tutorials/terminal-basics/coolterm-windows-mac-linux](https://learn.sparkfun.com/tutorials/terminal-basics/coolterm-windows-mac-linux)

The next step (setting the XBee modules' PAN ID numbers) should be done in MoltoSenso network manager IRON v. 1.01 (Sorry this section is still a bit rough...)

---

**Serial Port Options**
- **Port:** serial-0000 (default)
- **Baudrate:** 57600
- **Data Bits:** 8
- **Parity:** None
- **Stop Bits:** 1
- **Flow Control:** RTS
- **Initial Line States when Port opens:** DTR On, RTS Off

**Terminal Options**
- **Terminal Mode:** Raw Mode
- **Enter Key Emulation:** CR+LF
- **ASCII View Options:** Convert Non-printable Characters

**Receive Options**
- **Loop back received data**
- **Ignore receive signal errors**
- **Receive Buffer Size:** 10240
- **Capture Text Options**
  - **Capture Format:** Raw Data
  - **Add timestamps to received data**
  - **Wait for termination string**
    - **Termination String (Hex):** 0D 0A
    - **Type:** Absolute Date and Time
  - **Capture Local Echo**
  - **Leave File open while capturing**

**Transmit Options**
- **Use transmit character delay**
  - **Delay (ms):** 3
  - **Use transmit line delay**
  - **Delay (ms):** 3
  - **Delay characters (Hex):** 0A

**Send String Options**
- **Terminate Send String Data**
  - **Termination String (Hex):** 0D 0A
- **Send Text Options**
  - **Notify after sending text file**
Open IRON with your XBee connected to your computer:

Set the correct baud rate and open the port:

Hit “Get All” in the user interface to get all current XBee module settings. Use “Write Permanently” to write setting to XBee module.

Set a unique PAN ID (do this for each of the XBee modules):
The next settings are trying to duplicate what is mentioned in the Adafruit workshop, setting up the XBees in such a way that you can program an Arduino Diecimilia wirelessly (however, this hasn’t worked yet for me…) – these settings are not necessary if you only need to transfers serial data (i.e. from sensors) back and forth.
### Module Configuration

<table>
<thead>
<tr>
<th>Module</th>
<th>Node ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NCK020K64F</td>
<td></td>
</tr>
</tbody>
</table>

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Current</th>
<th>New</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT Command</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>I/O Settings</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BP</td>
<td>26</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>PR</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>P1</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>P2</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>P3</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D1</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D2</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D3</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D4</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D5</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D6</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D7</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
<tr>
<td>D8</td>
<td>FF</td>
<td>FF</td>
<td>FF</td>
</tr>
</tbody>
</table>

**Description:**
- **BP** (Battery Power): Setting for the battery power supply of the module.
- **PR** (Power Reserve): Setting for the power reserve function of the module.
- **P1** to **P8**: Various configuration settings related to the module's operation.
- **D1** to **D8**: Various configuration settings related to the module's digital outputs.

---

**Notes:**
- Configuration settings can be adjusted through the software interface.
- Each parameter can be set to a specific value as per the software interface instructions.
- The default values are provided for reference.

---

**Detailed Parameters:**

- **BP (Battery Power):** Setting for battery power supply.
  - FF: Full power supply.
- **PR (Power Reserve):** Setting for power reserve function.
  - FF: Full reserve.
- **P1** to **P8**: Various configuration options:
  - FF: Default option.
- **D1** to **D8**: Various digital output configurations:
  - FF: Default configuration.

---

**Additional Information:**
- **I/O Settings:** Additional interface options for configuring the module's I/O functionalities.
- **AT Command:** At commands for module control.
- **Current/Default:** Current and default values for each parameter.

---

**Software Interface:**
- The interface provides a graphical representation of the configuration settings.
- Users can adjust settings by clicking on the respective parameters in the interface.
- The interface also includes a help section for detailed instructions on each parameter.

---

**Technical Support:**
- For any issues or questions, please refer to the module's user manual or contact the technical support line for assistance.
(2) Using the Electric Imp for connecting the iRobot's Arduino board to the internet:

I am following the instructions in this helpful tutorial: [https://learn.sparkfun.com/tutorials/electric-imp-breakout-hookup-guide/improduction](https://learn.sparkfun.com/tutorials/electric-imp-breakout-hookup-guide/improduction)

You can use the test account setup for this class (or create your own at: [https://ide.electricimp.com/login](https://ide.electricimp.com/login))

Test account information:
username: ********
password: ********* (get this from me in class)

There were just a couple of minor things I found unclear in the tutorial (or that didn't work right away) – here is what I found out:

When you commission the IMP via the BlinkUp app, make sure to be in a rather dark space (i.e. no direct sunlight/daylight on the IMP) when you hold the IMP close to the cell phone screen.

Assign a device:
In the new IDE on the IMP's website make sure to click on the little gear next to the initial device name to assign the device/enter the device settings (give it a new name and write new code):

Example 2 (Web Control – Request) and example 3 (Web Response) could be especially interesting connecting the iRobot Create to the internet (either allow users to control it through a web browser, or give people access to sensor information from the robot via a web browser).

This tutorial has some ideas for how to link an Arduino and an IMP via a serial connection: [https://www.sparkfun.com/tutorials/397](https://www.sparkfun.com/tutorials/397)