1. Introduction

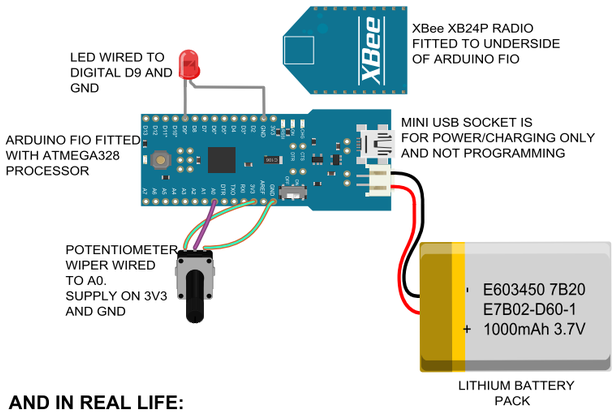
This tutorial is an extension of the article published at http://www.ladyada.net/make/xbee/arduino.html
The object of this tuorial is to show how to set up a wireless connection between an Arduino Fio using two XBee 24 Pro radios.
The tutorial will explain how to load the Fio bootloader to accept wireless uploads, how to configure the XBee radios to communicate, and then upload a program to the Arduino.

This partucular project uses an Arduino Fio fitted with an ATMega 328P processor which allows the communications baud rate to run at 57600 bps..

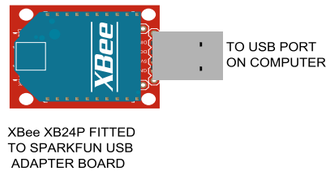
There are basically 3 components which require configuration, namely

1.1 The Arduino board
1.2 The XBee radios
1.3 The PC/Laptop

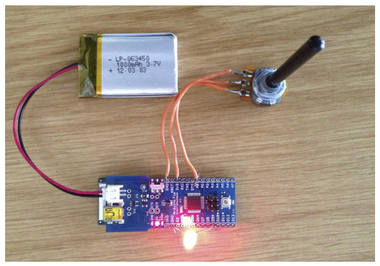
REMOTE SIDE (RECEIVER)



COMPUTER SIDE (TRANSMITTER)



AND IN REAL LIFE:

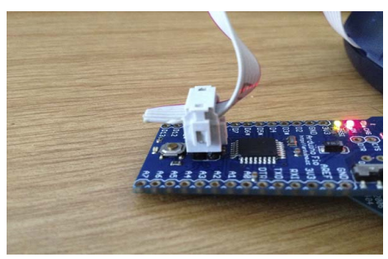


Remote FIO Rx side - (XBee underneath)

PC Tx side

2. Arduino FIO setup

The following description is relevant for an Arduino FIO fitted with an ATMega 328P processor. It will work with a 168 processor but then the maximum baud rate shouldbe 19200 bps and this baud used for the XBee and Arduino communication.
The FIO comes pre-programmed out-of-the-box with the correct bootloader but if this has been overwritten or corrupted it will need reflashing to enable wireless programming.
The first step is to flash the FIO with the correct bootloader program. The correct bootloader program is ATmegaBOOT_168_atmega328_pro_8MHz.hex. This is located in your Arduino directory under ....\hardware\arduino\bootloaders\atmega
To burn the bootloader for a FIO, I used an AVRISPmkII programmer. It is possible to use an FTDI breakout interface board and the Arduino IDE as aprogrammer but this will require some soldering of connections on the FIO to access the TX and RX channels.
With the AVR it uses the 6 pin ISP header on the FIO as shown below:

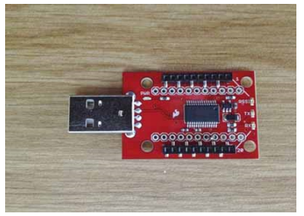


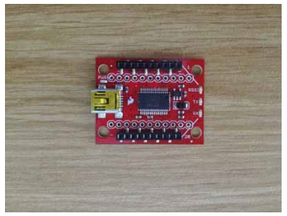
With the AVR connected and powered on and power on the FIO (either from the battery or mini USB connected to a PC), open a blank sketch in the Arduino IDE. Under 'Tools' select the board type from the dropdown list (Arduino Fio)
Next, select serial port used for connecting the AVRISPmkII. Now choose the programmer type under the 'Programmer' selection. In my case it is the AVRISP mk II.
Once the above selections are made, click on 'Burn Bootloader' and the bootloader should upload to the Arduino. The on board LED on the Fio (above the reset pushbutton) will start to blink at the end of the burn routine. This is part of the bootloader program to inform that the upload was succesful and will continue to blink until an application sketch is uploaded. The Fio is now ready to accept uploads wirelessly once an XBee is plugged into the header socket on the underside of the board.

3. XBee radio setup

The next task is to configure the XBee radios. My example uses XBee Pro units, type XBP24. You can mix and match XB24 and XBP24 and I have successfully tried a combination of both. The configuration is easier with both types being the same as the parameter set and firmware are identical if they are the same type.
There are a number of methods which allow parameterizing ranging from command line methods using the Hayes AT command sets but I chose a GUI method utilizing a software package from Digi International called X-CTU. This package allows parameterizing and firmware updating in one interface and permits saving configurations to file. It also has a number of utilities including Range Test and network discovery.
At the time of writing, the latest version of X-CTU is 6.1.0 for both Windows and Mac and can be downloaded from Digi International's web site at http://www.digi.com

The XCT-U software package also contains firmware libraries fot the XBee radios. At the time of writing, the latest version for the XB24 and XBP24 is version 10ed. The first step in preparing the radios is to flash them with the latest firmware. Now, to do this the XBee's need to be connected to the computer running the XCT-U software. The software can update firmware wirelessly to remote radios but at least on base radio must be connected to the computer. For this project I will only describe the process using the radios connected directly to the computer to keep things simple. I used Sparkfun's USB XBee breakout boards to do this. Both the standard USB and the mini USB can be used.
http://www.sparkfun.com





Software Required for this system setup:

1. X-CTU ver. 6.1.0 available from http://www.digi.com

2. Arduino IDE available from http://www.arduino.cc

3. The "Blink.ino" Arduino example file. (Included with IDE)

4. The Rx and Tx XBee config files (*.pro), included in the GITHUB repository... https://github.com/aetcnc/Arduino_FIO_XBee_Programming)

Arduino FIO - Wireless programming Prepared By: AC Date: 2412/2013	Affordable•Engineering•Technologies
Arduino Fio Wireless Programming - Sheet 1/2	