Wireless Communication With Arduino

Using the RN-XV to communicate over WiFi

Seth Hardy
shardy@asymptotic.ca

Last Updated: Nov 2012

Overview

- Radio: Roving Networks RN-XV
  - "XBee replacement": fits in the same socket
  - This method can be used for XBee as well, but they are much harder to configure
- Four connections: power, ground, transmit (TX), receive (RX)
- 3.3V - do not use directly with 5V Arduino!

RN-XV + Arduino

- The easy way:
  - Step 1: Configure the radio
  - Step 2: Use as a standard serial port
- The hard way:
  - Step 1: Find an Arduino library that configures the radio
  - Step 2: Run a program and hope it works
- Most people seem to think these ways are reversed before giving it a try

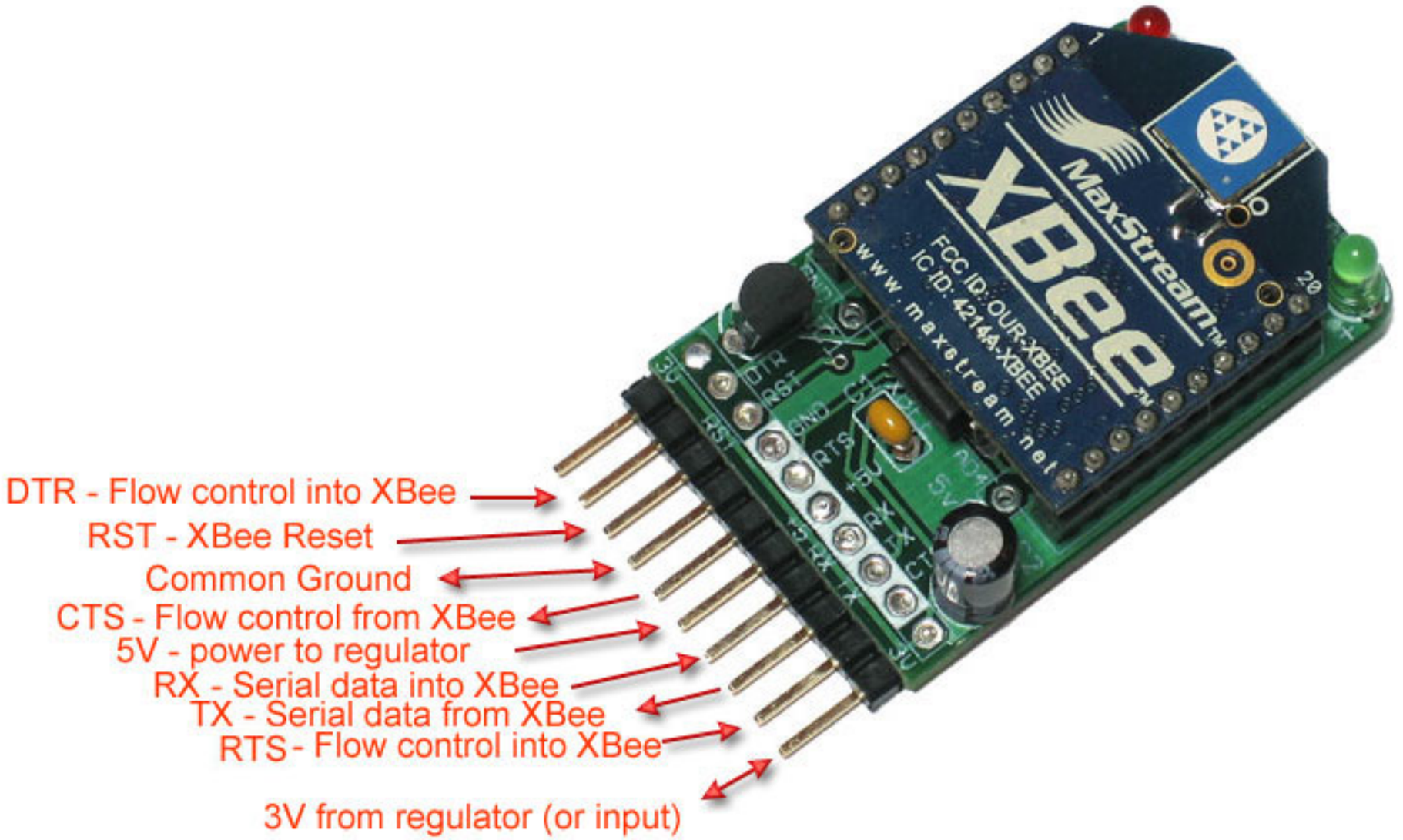
Talking to the RN-XV

- The radio module has a built-in serial port (UART) - the TX/RX pins
- You can talk to it through a terminal program with a USB->Serial converter (often called FTDI cable/chip)
  - FTDI is the manufacturer of the chip
  - The "FTDI cable" has a chip in the cable
- Use a set of commands to configure how the radio works
  - The RN-XV manual has all the documentation you'll need to do this

FTDI Cable Time

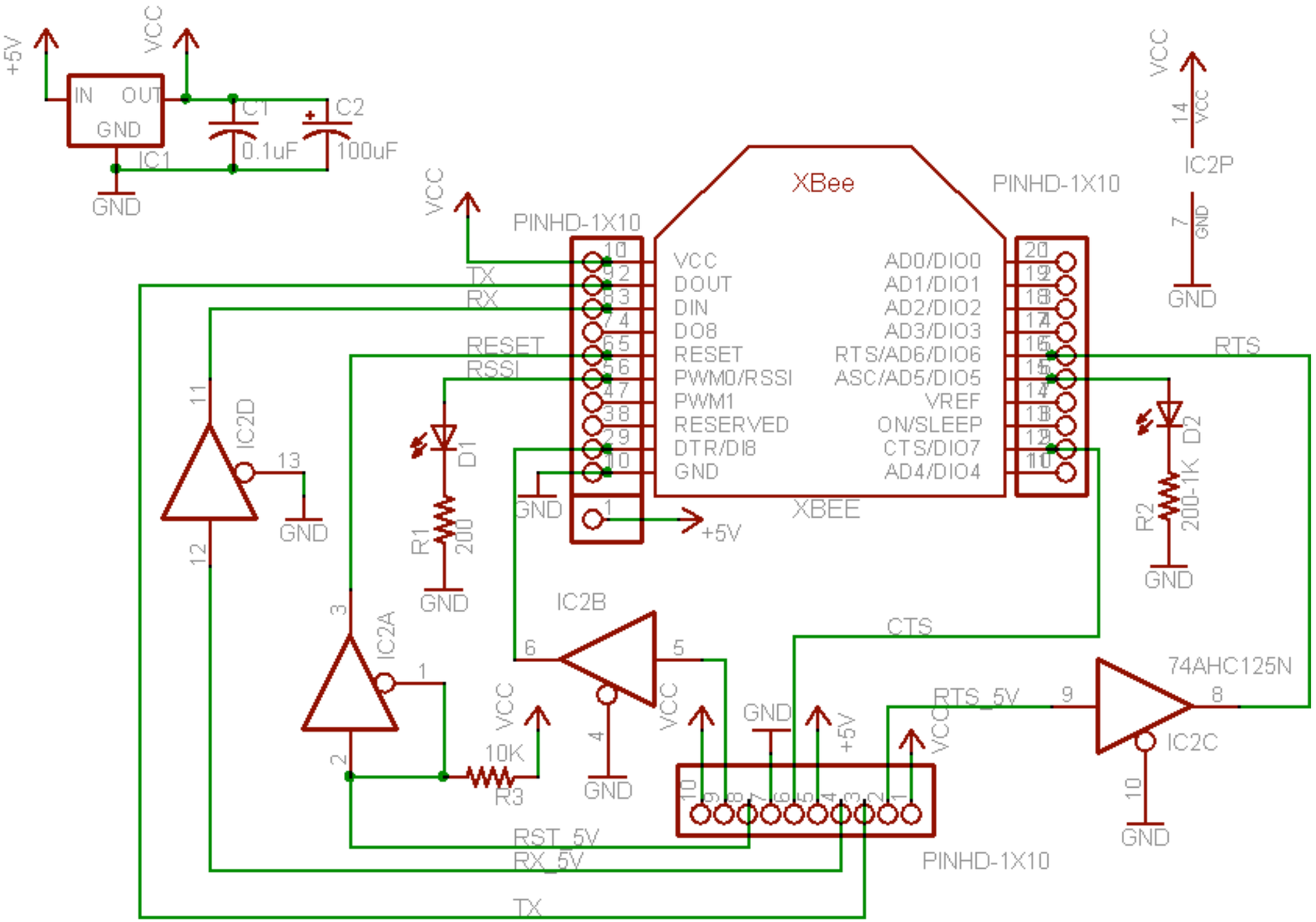
- To make this easier, we'll use the following hardware:
  - Adafruit XBee Adapter: provides a "FTDI header" and 5V to 3.3V level shifting
  - FTDI breakout board: like a cable, but lets you reuse a Mini-USB cable
- You get to assemble the XBee adapter.

XBee Adapter



We use this instead of the SparkFun XBee Explorer because it uses a real level shifter, not a diode.

Level Shifting



Connect to the RN-XV

- Put the radio in the adapter board
- Plug the FTDI cable into the board so that the “ground” markings on pin 1 line up
- Open your terminal program and connect at 115200 bps

Talking to the RN-XV

- Two modes: data and command
  - Data mode: data is sent “through” the radio - it is transmitted depending on how it is configured
  - Command mode: data is sent “to” the radio - this is where the radio is configured
- To enter command mode, type in “$$$”
- If the radio responds with “CMD”, you are in command mode

RN-XV Command Mode

- Two important commands: "get" and "set"
  - "get" shows configuration options
  - "set", uh, sets them
- Different categories and configuration options within each category
- Try typing "get wlan" to see current configuration for the wireless network

RN-XV Configuration Options

Adhoc controls the adhoc parameters

Broadcast controls the broadcast hello/heartbeat UDP message

COMM communication and data transfer, timers, matching characters

DNS DNS host and domain

FTP FTP host address and login information

IP IP settings

Option optional and not frequently used parameters

Sys system settings such as sleep and wake timers

Time timer server settings

UART serial port settings such as baudrate and parity

WLAN wireless interface settings, such as ssid, chan, and security options

Note: none of the options are case sensitive, and can often be shortened.

Set Device Name

set opt deviceid NameOfYourDevice

sets the device name

save

writes the new options to memory

get opt

shows the configuration

Configure Wireless

set wlan ssid [networkname]

NOTE: use $ instead of a space, e.g. "Free$candy" - see "get opt replace" for which character is the substitute

set wlan passphrase [passphrase]

again, use $ instead of a space

save

reboot

This will use DHCP (automatic IP address assignment) and will be able to determine what encryption method is being used.

Configure Wireless

(the easy way)

scan

```
SCAN:Found 5


Num             SSID   Ch  RSSI    Sec    MAC Address.Suites
1               America 01 -79 WPA2PSK 1c:7e:e5:3c:0b:9c AES/TKIPM-TKIP  WPSPB 3104
2             Elements 08 -75 WPA2PSK 30:85:a9:3a:37:88 AESM-AES   1104
3            Sabsburger 10 -79   WPAv1 00:26:f3:31:01:dc TKIPM-TKIP   7104
4               BURKASS 11 -71     WEP 00:30:4f:70:8e:e4    1104
5               BELL271 11 -71     WEP 3c:ea:4f:71:3e:81    3104
```

join

save

reboot

Note: doesn’t always work.

Test Connection

Reboot, and it should look like this:

```
*Reboot*.WiFly Ver 2.32, 02-13-2012 on RN-171
MAC Addr=00:06:66:71:e2:66
Auto-Assoc OMG HAX chan=6 mode=WPA2 SCAN OK
Joining OMG HAX now..
*READY*
Associated!
DHCP: Start
DHCP in 2939ms, lease=14400s
IF=UP
DHCP=ON
IP=10.0.1.16:2000
NM=255.255.255.0
GW=10.0.1.1
Listen on 2000
```

Check Settings

- Enter command mode (\$\$\$)
- “get ip” to show network config
  - Remember this IP!
- “ping [IP]” to test connectivity

Test Connectivity

- From your laptop, try to connect to the radio:
  - Windows: open a command prompt (run cmd.exe)
    - Mac: open a Terminal window (under Utilities)
    - Linux: you already know how to do this
- Run “telnet [IP] 2000” and see if it connects
- This is directly connecting to the radio itself!

Test Connectivity

On the telnet session:

```
Macintosh:~ shardy$ telnet
10.0.1.16 2000
Trying 10.0.1.16...
Connected to 10.0.1.16.
Escape character is '^]'.
*HELLO*
[type something in]
^]
```

In the serial terminal:

```
*OPEN*
[what you typed]
*CLOS*
```

Radio Lights

- Not connected to a network:
  - Red and green lights blink
- Connected to a network:
  - Connected: green light on
  - Connected, data mode active: green light blinks slowly
  - Connected, command mode active: green light blinks quickly
- Data being sent:
  - Amber light blinks

Turn off remote admin

- What this means: anyone can connect to your radio over the network
- We need to turn off remote administration
  - Go back to command mode in the serial terminal and type in:
    - “set ip tcp-mode 0x10”
    - To turn back on: “set ip tcp-mode 0”
    - Don’t forget to “save” afterwards

Now what?

- The radio acts as a tunnel: it echoes anything from the serial console to a connection made to it
- This would be more useful if the radio connects outbound and acts as a tunnel
- This is easy to set up!

Outbound UDP

- Any data the radio receives on the UART, it bundles in a UDP packet and sends.
- Any data the radio receives on its listening port (2000 by default, change with “set ip port”) gets sent on the UART.
- UDP is connectionless: it just sends data packets. No verification of receipt.
- Very useful for simple data streams.

Configuring UDP

On the radio:

```
set ip host [address]
set ip remote [port]
set ip protocol 0x1
save
exit
```

On your laptop:

```
nc -u -l [port]
```

Windows users may have to download netcat.

Outbound TCP

- TCP makes a full connection, which it can keep open to keep sending more data.
- Guaranteed delivery of data.
- May not be as fast as UDP, and less suitable for when you are “streaming” something and don’t care about losing occasional data.

Configuring TCP

On the radio:

```
set ip host [address]
set ip remote [port]
set ip protocol 0x2
save
open
```

On your laptop:

```
nc -l [port]
```

“open” creates the connection to the server (your laptop).

Outbound HTTP

- The radio can talk HTTP (web page requests) easily as well.
- Two ways of doing this:
  - GET: request a web page and put data in the request URL
  - POST: request a web page and send data in the body
- We'll just look at GET requests here.
- The radio can also receive data from a webpage and use it locally.

Configuring HTTP

On the radio:

```
set ip host [address]
set ip remote [port]
set ip protocol 18
set com remote [url]
save
exit
```

On your laptop:

```
nc -l [port]
```

This will append data to the URL you specify.

HTTP Example

Configure the radio:

```
set com remote GET$/testing.php?data=
save
```

Listen for requests:

```
nc -l [port]
```

Type in “test” and you see this:

```
GET /testing.php?data=t HTTP/1.0

Host: server1

GET /testing.php?data=e HTTP/1.0

Host: server1

GET /testing.php?data=s HTTP/1.0

Host: server1

GET /testing.php?data=t HTTP/1.0

Host: server1
```

Buffers and Flushing

- Why are there four requests with one character, instead of one request with four characters?
- The output buffer is "flushed" (sent) when certain thresholds are hit: time, size, or a matching character (e.g. carriage return or line feed)
- You can control these buffers:
  - set comm time [seconds]
  - set comm size [size]
  - set comm match [char]
- The radio can also connect automatically:
  - set sys auto [every N seconds, from 2 - 254]

So... what now?

- Configure the radio
- Connect radio TX to Arduino RX (usually pin 0), radio RX to Arduino TX (pin 1)
- Use the serial port as normal:

```
Serial.begin(115200)
Serial.println("hello");
if( Serial.isAvailable() > 0 ) { }
```

Advanced Topics

- Want to talk to multiple IPs?
  - Libraries exist to configure the radio from Arduino
  - Serial.print(“$$$”);
    delay(500);
    Serial.println(“set ip host 192.168.1.1”);
    etc.

Advanced Topics

- UDP broadcast packets
  - Sent automatically every N seconds
  - By default, to broadcast (255.255.255.255), UDP port 55555
  - Contains radio name, time, sensor data, signal strength, battery voltage

Advanced Topics

- Don't want to use an Arduino?
  - RN-XV has digital GPIO and analog sensor inputs
  - The radio can be configured to automatically read and send sensor data