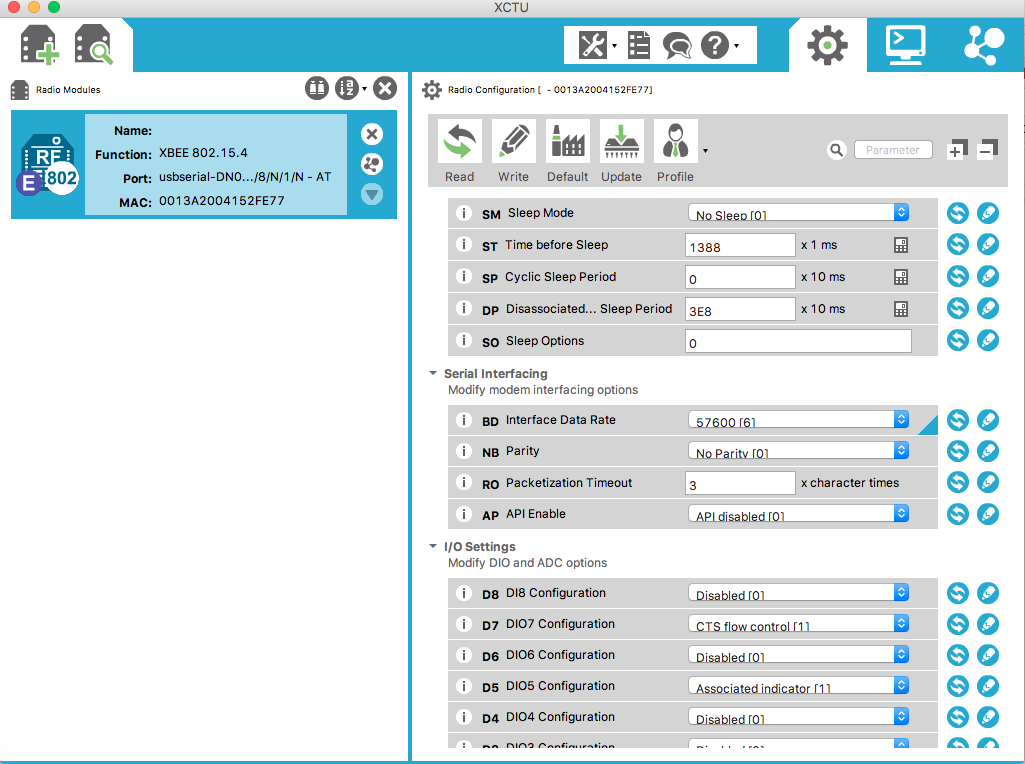
Wilson Conley - [jconley@g.clemson.edu](mailto:jconley@g.clemson.edu)

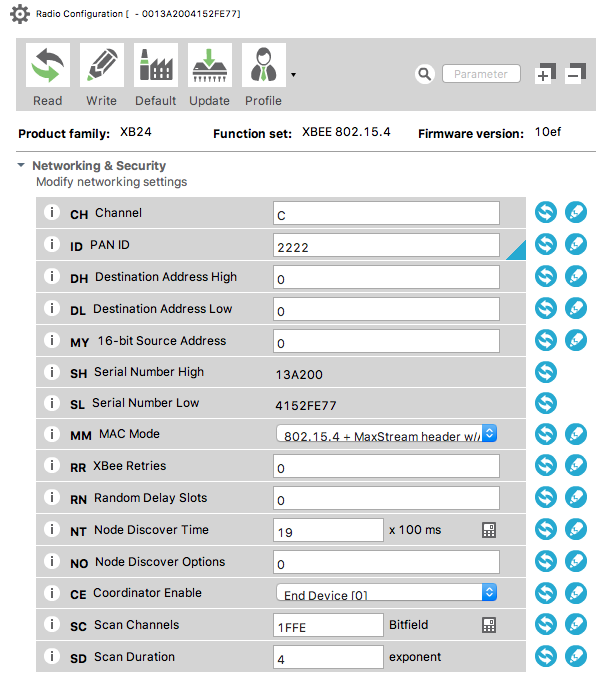
Michaela Loar - [mloar@clemson.edu](mailto:mloar@clemson.edu)

**Xbee:**

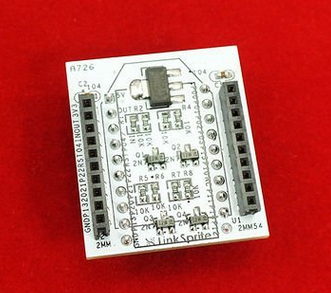
For this project, we are using the Xbee modules to send and receive data from one Arduino board to another. Xbee is a type of radio frequency communication that is more reliable that what was being used before. The previous RF transmitters that were being used would not always receive the data that was being sent, so we chose to implement the Xbees to fix this problem. The Xbee is full duplex, allowing for both sending and receiving of data across multiple devices at the same time. The Xbees should work reliably up to about 100 ft indoors and use a 2.4GHz frequency.

**Configuring the Xbee:**

1. The first thing that you need to do to get the Xbees working is setting them up to baud rate that you want to use. To do this, download the program XCTU from this link: <https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu#productsupport-utilities>
2. Then plug the Xbee into the Xbee Explorer, which looks like this: Plug that into your computer and open up XCTU. Hit the discover radio modules button in the top left corner of the XCTU window. It looks like a folder with a magnifying glass over it. Then select the usb port and search for devices. If the Xbee does not show up at first, select to search for more baud rates in the search options.
3. Once you have found the Xbee, select it. The XCTU window should then look like this:
4. Scroll down to the Serial Interfacing section and find the Interface Data Rate option. Change it to whatever baud rate you want the Xbees to communicate with, and then click the write button at the top of the window. The 115200 baud rate is the only one that doesn’t seem to work for the Xbee. The highest one that we got to work was 57600.
5. You also need to make sure the the Channel and Pan ID of the Xbees are all the same. In the picture below, it shows the current settings of the Xbees. Make sure any new Xbees are set to these values by changing them and then clicking the write button (as you did with the baud rate).



1. Once you are done configuring the Xbee (make sure to click write to save the changes), you can just unplug the Explorer. You can use the same Explorer to configure all the Xbees.



**Hooking up Xbee to Arduino Board:**

1. The first thing you need to do to connect the Xbee to the Arduino is attach the Xbee to the Xbee breakout board.
2. Put the breakout board into the breadboard.
3. Pin 1 of the Xbee should be hooked up to the 5 V power supply of the Arduino board, and pin 10 should be attached to the ground of the Arduino board.
4. To use the Xbee to communicate, we will use the SoftwareSerial function. SoftwareSerial uses two pins of the Arduino board as a new serial communication port. In our code, we are using pins 11 (RX of Arduino) and 12 (TX of Arduino).

More info on SoftwareSerial function can be found here:

<https://www.arduino.cc/en/Tutorial/SoftwareSerialExample>

1. Pin 2 of the Xbee (the TX of the Xbee) should be attached to whatever pin you are using as the RX of the Arduino (We are using pin 11) and pin 3 of the Xbee (the RX of the Xbee) should be attached to the TX pin of the Arduino (we are using pin 12).
2. The schematic of the Xbee modules can be found here:

<https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf>

1. These four connections are the only ones we need to make to have the Xbees communicate.

**Arduino code for Using Xbee modules:**

1. First step is to include the SoftwareSerial library and create the Xbee SoftwareSerial label. Code to do this is:

#include <SoftwareSerial.h>

SoftwareSerial XBee(11, 12); // RX, TX - (11 and 12 can be any pins you want, but those are the ones we are using)

1. In the setup function, set the baud rate of the XBee to whatever baud rate you want to use. We are using 57600 right now. This baud rate can be different than the Arduino baud rate (the Serial.begin() baud rate), but make sure that all the Xbees are using the same baud rate. You must configure the Xbee modules to the baud rate you want to use before it will work. The steps to do this are outlined in the previous section. The code to set the baud rate is:

XBee.begin(57600); //57600 is the current baud rate

1. The Xbee module then basically functions as its own serial port. The commands to send and receive data are the same as they would be as if you were using the serial monitor. Now, to send information via Xbee, use the following code:

XBee.write();

1. To read in information via Xbee, use the following code:

XBee.read();

**Sample Code: (can also be found on Github)**

Once you have everything configured on the Xbee modules and have them hooked up to the Arduino boards, you should be able to run the following test code and have the Xbees talk to each other. Make sure the baud rate of the Serial monitor matches the baud rate of the Arduino set by the Serial.begin() command. The XBee baud rate should be set to whatever you configured the modules to (all Xbees should be the same). Hook up each Arduino board equipped with an Xbee to its own computer. Upload the code. Data sent to the Arduino board via the Serial monitor on one computer should be printed out to the Serial monitor of the other computers hooked up to Arduino boards with Xbees.

// We'll use SoftwareSerial to communicate with the XBee

// Input data via the serial monitor and it will send the data you inputted out to the other xbees

// Data sent from other xbees will be received and printed out to the serial monitor

#include <SoftwareSerial.h>

// XBee's DOUT (TX, pin 2 of xbee) is connected to pin 11 of Arduino board (Arduino's Software RX)

// XBee's DIN (RX, pin 3 of xbee) is connected to pin 12 of Arduino board (Arduino's Software TX)

SoftwareSerial XBee(11, 12); // RX, TX

void setup()

{

// set up baud rates for XBee and Serial monitor

XBee.begin(57600);

Serial.begin(115200);

}

void loop()

{

if (Serial.available())

{ // If data comes in from serial monitor, send it out to XBee

XBee.write(Serial.read()); // send data to other xbee

Serial.write("sent"); // print to serial monitor to confirm data was sent

}

if (XBee.available())

{ // If data comes in from XBee, write that data out to serial monitor

Serial.write(XBee.read());

}

}

**Why we chose Xbee:**

We decided to go with the Xbee modules for a number of reasons. We thought about doing bluetooth or wifi as a new form of communication, but we decided that both of those would not work for our purposes. Bluetooth requires one device to be paired to another, but we want all of the Roombas to communicate with each other the same time. So because we want multiple devices communicating at the same time, we decided not to go with Bluetooth. We decided not to go with wifi because we didn’t want to have the Arduinos have to communicate through a router or anything like that. Also with wifi, we would have to be operating the Roombas in an area with wifi and have all the Arduinos connect to it, and we felt that this would make it too complicated and unreliable. We wanted a direct form of communication that would be reliable and work for multiple devices at the same time. We chose to go with Xbee because of that. It still uses RF, but it is much more reliable than the RF transmitters we were previously using. It is also relatively cheap, only about $20 per module. The Xbees are full duplex, allowing for two-way communication across multiple devices. They integrate nicely with the Arduino boards and the code we are using.

**What we did this semester:**

At the start of the semester, we were given the task of finding a new form of communication between the Roombas that would be more reliable than the current RF transmitters. For the first couple weeks, we researched different forms of communication. We looked at bluetooth and wifi, but ultimately we chose to go with the Xbee modules. The first thing we did was order 3 Xbee modules and 3 Wireless SD Shields to come along with them. We are able to get these Wireless SD Shields to get the Xbees to communicate, but there was a problem. The Wireless SD Shields we got used pins 0 and 1 as the RX and TX between the Arduino and the Xbee module. These pins are reserved for the Serial monitor, and you had to use the Serial monitor to communicate with the other Xbees. Everything written to the other Xbees would be written to the Serial monitor, and everything written to the Serial monitor would be written to the other Xbees. This worked for the purposes of getting the Xbees to communicate with each other, but for our project we really needed to have pins 0 and 1 open for just the Serial monitor.

Unfortunately, there was nothing we could do to change the RX and TX pins of the Wireless SD Shield to something other than 0 and 1. We realized that we needed to use SoftwareSerial in order to create a new serial port that used two pins other than 0 and 1. We tried soldering wires directly to the RX and TX pins of the Xbee via the breadboard on the Wireless SD Shield and wiring them to the pins we specified as our new serial port via SoftwareSerial. This still did not work, however. We then thought that if we manually configured the Xbee modules with the Xbee Explorer to the same channel, ID, baud rate, etc that we could still get the Wireless SD Shields to work. We got the Explorers and configured the Xbees all to the same settings, and then tried to use them with our soldered Wireless SD Shields, but using pins other than 0 and 1 as a serial port still did not work. It was at this point that we realized that the Wireless SD Shields were not going to work and that we were going to have to just wire everything directly from the Arduino to the Xbee in order to get it working the way we wanted.

We then had to order the breakout boards for the Xbee modules because the pin spacing on the actual Xbees is 2 mm, whereas the breadboard pin spacing is 2.5 mm, and the breakout boards allowed the Xbees to fit onto the breadboards. We then were able to wire any two pins on the Arduino board that we wanted to use as a new serial port with SoftwareSerial to the RX and TX pins of the Xbee (pins 2 and 3 of the Xbee). The Xbee runs on 3.3V, but the breakout takes 5V and converts to 3.3V. We just had to wire the 5V power supply from the Arduino board and the Ground from the Arduino board to the power and ground of the Xbee (pins 1 and 10). We were then able to get the Xbees all communicating without having to use pins 0 and 1 (the Serial monitor). We figured out that the Xbees still had to all be configured to the proper baud rates with the Explorer before they would work together. We learned that 57600 was the highest baud rate we could get to work with the Xbees. We then implemented the sample code provided earlier to test the communication between the Xbees, and it worked. We went in the hallway and tested the max range of them, finding it to be about 75-100 ft. A majority of the problems we encountered this semester had to do with the Wireless SD Shield, and once we stopped using that and wired directly to the Xbee from the Arduino we were able to get them working.

**Links to Xbee components:**

Xbee module (need 1 of these for every board):

<https://www.amazon.com/XBee-1mW-Wire-Antenna-802-15-4/dp/B004G4ZHK4/ref=sr_1_1?ie=UTF8&qid=1487812484&sr=8-1&keywords=XBee+1mW+Wire+Antenna+-+Series+1+%28802.15.4%29>

Xbee breakout board (need 1 of these for every Xbee module):

<https://www.amazon.com/LinkSprite-Breakout-Board-Module-interface/dp/B0191IJD36/ref=sr_1_1?rps=1&ie=UTF8&qid=1491666095&sr=8-1&keywords=xbee+breakout+board&refinements=p_85%3A2470955011>

Xbee Explorer (should only need 1 of these, just configure the Xbees one by one):

<https://www.amazon.com/SparkFun-LYSB011QJIHY4-ELECTRNCS-Sparkfun-Explorer-Dongle/dp/B011QJIHY4/ref=sr_1_1?ie=UTF8&qid=1489611403&sr=8-1&keywords=xbee+explorer+dongle>

XCTU program download:

<https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu#productsupport-utilities>

**Websites we found useful:**

<http://cs.smith.edu/dftwiki/index.php/Tutorial:_Arduino_and_XBee_Communication>

<https://www.sparkfun.com/pages/xbee_guide>

<https://learn.sparkfun.com/tutorials/exploring-xbees-and-xctu>

<http://cs.smith.edu/dftwiki/index.php/Tutorial:_Arduino_and_XBee_Communication>

<http://www.instructables.com/id/Changing-Xbee-Baud-Rates/>

<https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf>

<https://www.arduino.cc/en/Tutorial/SoftwareSerialExample>