Arduino / Xbee Wifi Communications (Basics)

Introduction:

This project covers basic Arduino / Xbee (Zigbee) radio frequency (RF) communications. We are merely scratching the surface here on what Xbee and Arduino controllers can do; for now, the tutorial aims to get the Arduino and Xbee communication system set up, and transmit and receive a stack frame of data from one Arduino to another. In order to keep the tutorial simple we will only be interpreting a single bit of all the data sent per transmission.

*Important side note: there's a lot of information for Xbees but most explanations and detailed reference material to your Xbee chips can be found here:
https://www.digi.com/resources/documentation/digidocs/pdfs/90000976.pdf

Procedure:

SETUP

To begin, the Xbee communication chip needs to be setup. If you have taken ECE 36200 microprocessors you might remember universal synchronous asynchronous receiving transmission USART communication. Luckily the Arduino / Xbee interface simplifies a lot of the setup procedure for your micro controllers. The Xbee chips operate in UART/USART communication in several modes AT / API; the easiest way to set them up is with a free software made by the manufacturers of Xbee called XCTU.

XCTU download:
https://www.digi.com/resources/documentation/digidocs/90001526/tasks/t_download_and_install_xctu.htm

Luckily, they have links to download in windows, Linux, and osx!!!

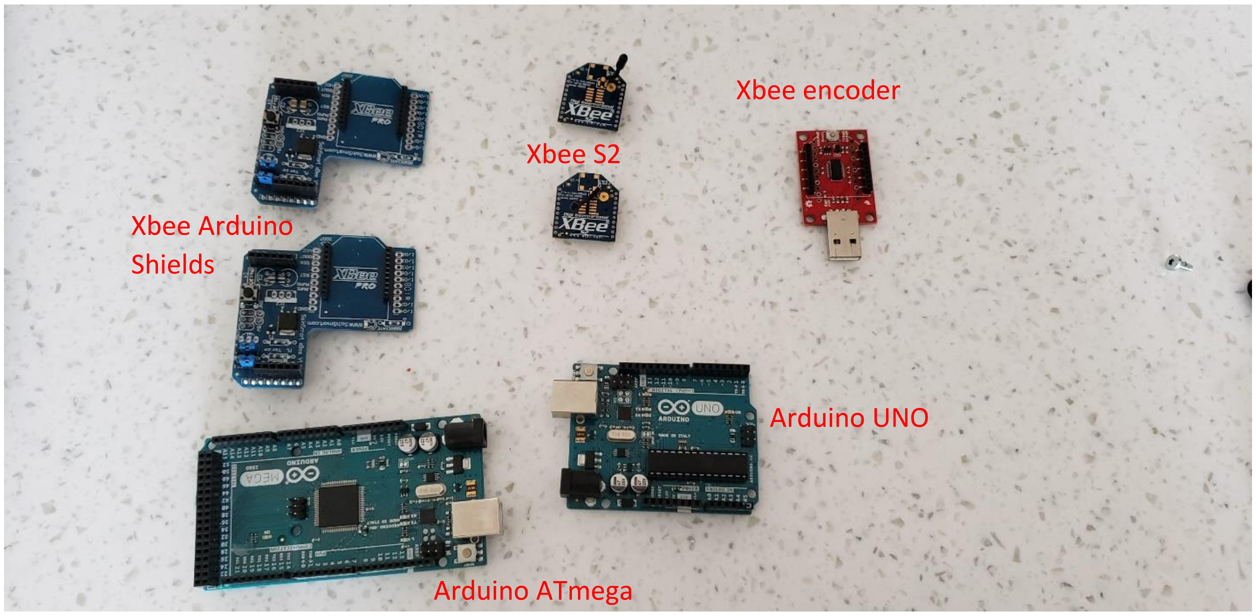


Figure 1: Parts (2 arduino units / 2 arduino Xbee shields / 2 Xbee chips / 1 xbee encoder)
Side note There are a lot of ways to connect an Xbee to an Arduino. The Xbee chip's pins don't align with a normal breadboard so an adapter will be needed in one way or another. This tutorial uses Xbee, Arduino shields, because of their ease of connection and quick switching between programming and running; shown later.

Some connection methods:
-https://www.amazon.com/Arduino-org-FBA_01ARD00007-Arduino-Xbee-Shield/dp/B004L6PNLA
**Recommended

-https://www.adafruit.com/product/126 *Recommended

-https://www.amazon.com/SparkFun-XBee-Breadboard-Adapter/dp/B077DRMP5Z

-https://www.digikey.com/product-detail/en/dfrobot/DFR0015/1738-1230-ND/7087127?utm_adgroup=Evaluation%20Boards%20-%20Expansion%20Boards%2C%20Daughter%20Cards&utm_source=google&utm_medium=cpc&utm_campaign=Shopping_Development%20Boards%2C%20Kits%2C%20Programmers_NEW&utm_term=&utm_content=Evaluation%20Boards%20-%20Expansion%20Boards%2C%20Daughter%20Cards&gclid=EAIaIQobChMI2dCf4vn16AIVzP_jBx2MwgATEAQYFCABEgKnjfD_BwE

For this example, we will set up our Xbee chips in the following ways.

Real quick,
A short explanation for Xbee chip operation:
In a simple communication network, Xbee uses a designated master chip (Coordinator in Xbee talk) and one or more slaves (Routers in Xbee talk). A Coordinator can send and receive

data from the Routers depending on the mode the chips are configured to API or AT. The Routers send data to the Coordinator.

- After the XCTU software is downloaded, put a chip into your USB programmer and connect to the XCTU software.

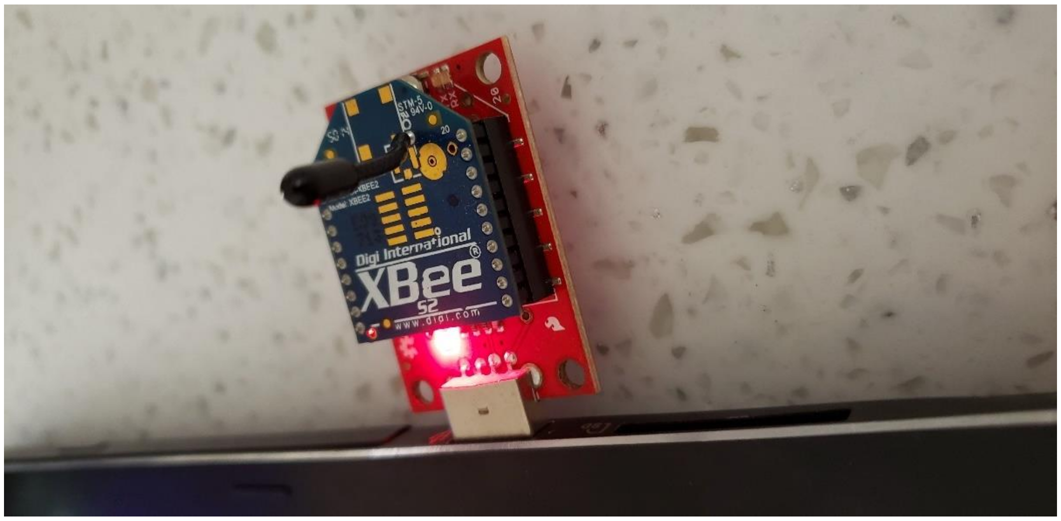
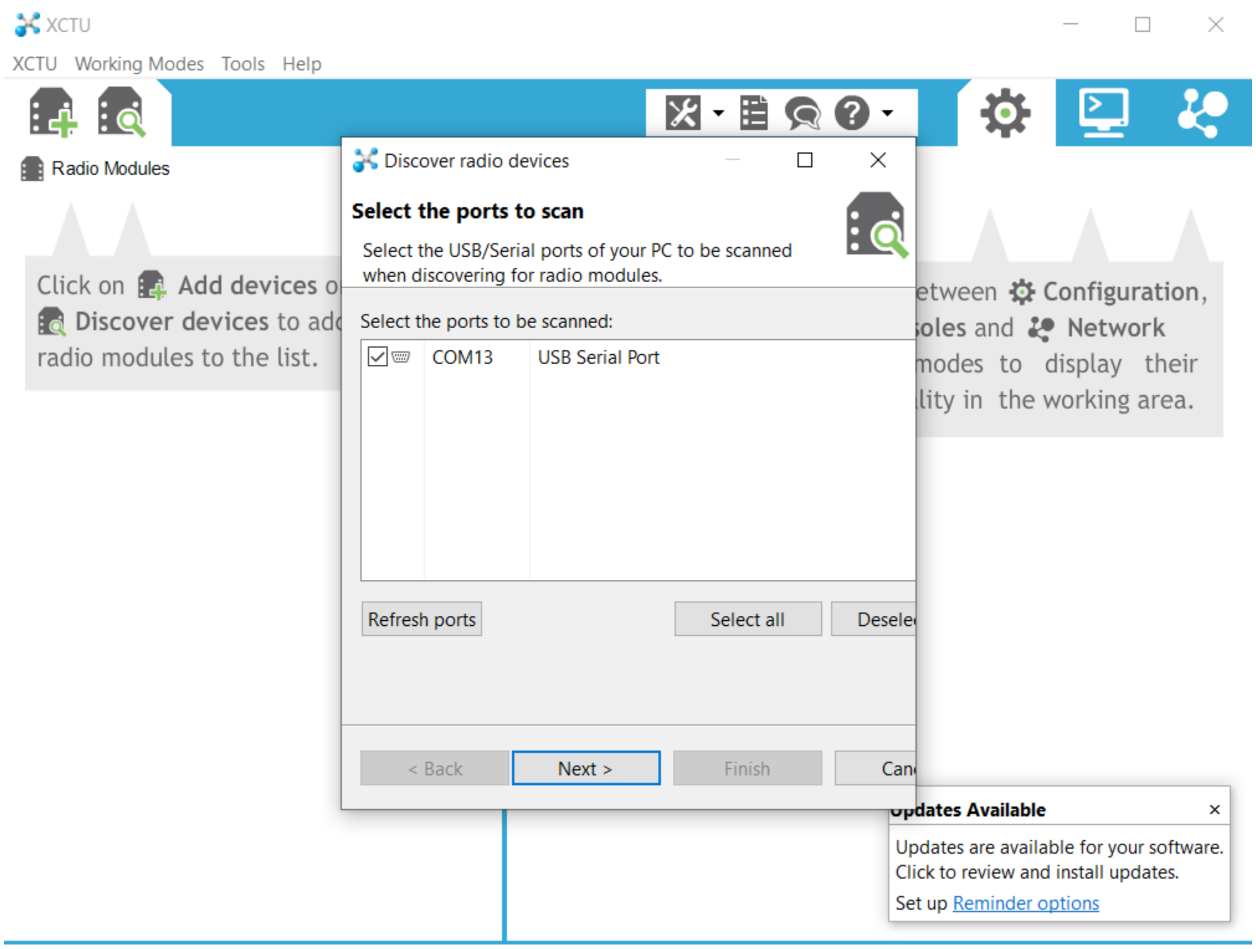


Figure 2: Xbee encoder



Select the ports to scan

Select the USB/Serial ports of your PC to be scanned when discovering for radio modules.

Select the ports to be scanned:

✓	COM13	USB Serial Port

Click on Add devices o Discover devices to add radio modules to the list.

etween Configuration, oles and Network modes to display their lity in the working area.

Updates Available

Updates are available for your software. Click to review and install updates.

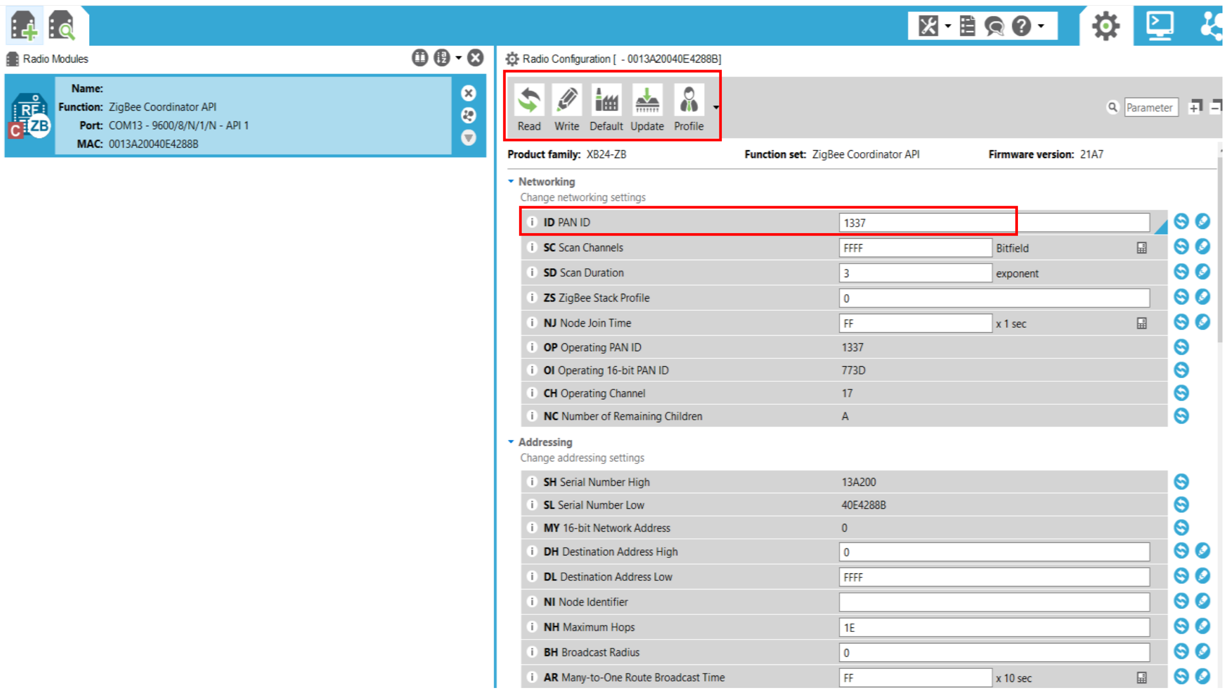
Set up Reminder options

Figure 3: Xbee XCTU terminal connection

- After you hit next on the chip you want to encode you will be prompted with the device settings: data bits, baud rate, parity etc… For this tutorial and most basic applications the default settings are enough to accomplish simple projects. For this project we used 9600 baud. Continue with the basic settings.

- You will then need to specify the chip as the Coordinator or the Router. We will start with the Coordinator in API mode.

API stands for application programming interface and in this example we will be using it to scan a stack frame of data from the reception end and output it via the Arduino interface.



Name:
Function: ZigBee Coordinator API
Port: COM13 - 9600/8/N/1/N - API 1
MAC: 0013A20040E4288B

Radio Configuration [- 0013A20040E4288B]

Product family: XB24-ZB   Function set: ZigBee Coordinator API   Firmware version: 21A7

Networking
Change networking settings

ID PAN ID	1337	
SC Scan Channels	FFFF	Bitfield
SD Scan Duration	3	exponent
ZS ZigBee Stack Profile	0	
NJ Node Join Time	FF	x 1 sec
OP Operating PAN ID	1337	
OI Operating 16-bit PAN ID	773D	
CH Operating Channel	17	
NC Number of Remaining Children	A	

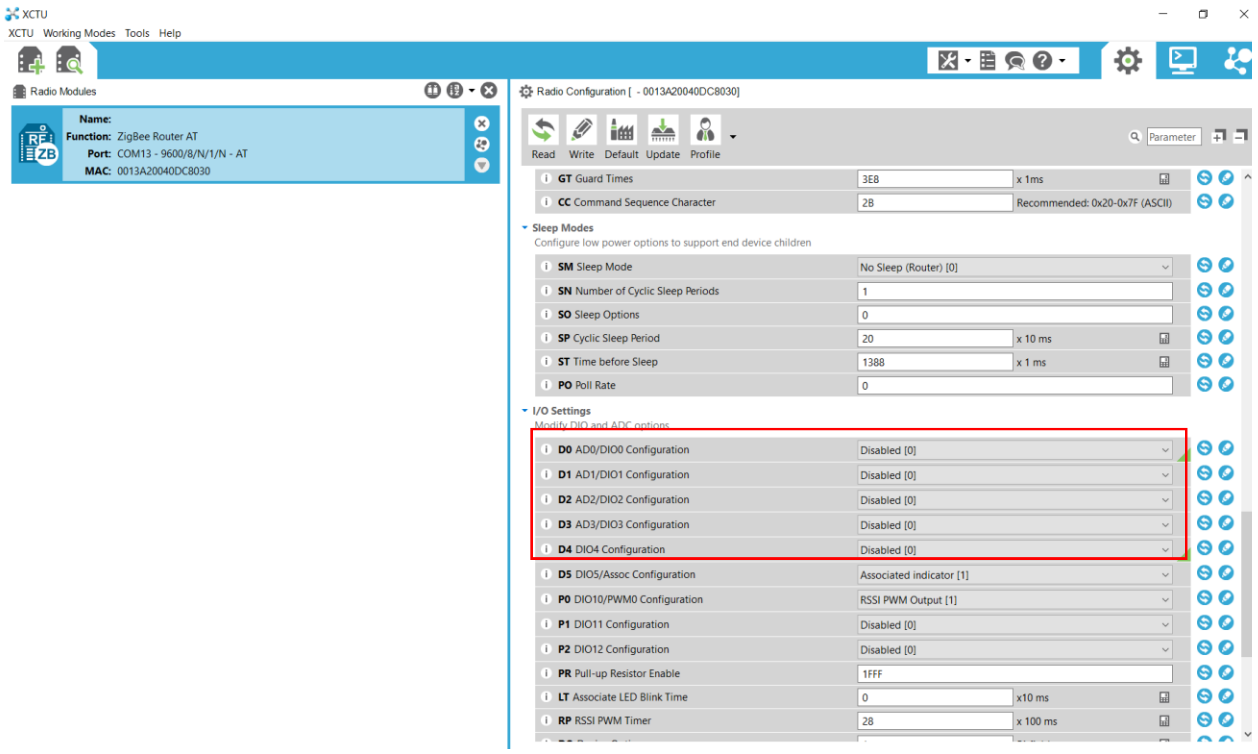
Addressing
Change addressing settings

SH Serial Number High	13A200	
SL Serial Number Low	40E4288B	
MY 16-bit Network Address	0	
DH Destination Address High	0	
DL Destination Address Low	FFFF	
NI Node Identifier		
NH Maximum Hops	1E	
BH Broadcast Radius	0	
AR Many-to-One Route Broadcast Time	FF	x 10 sec

Figure 4: Xbee cordinator settings

- Red Box Explinations
  - Update: is used to update the firmware the chip uses and should be selected before doing any of the following. Read and Write: begin by reading. This will load any previous chip settings already on your Xbee. After all the changes that needed to be made are done write them to the chip.
  - Pan ID: Pick a random number and remember to set every Xbee in the network to this number!! It is used to check to make sure that it is receiving and sending the information to the place it needs to go to.

- We will now setup the important settings for the Router. We will be using it in AT mode. For more info on AT operations: https://www.gme.cz/data/attachments/dsh.772-148.2.pdf. Remember to keep the settings the same as those you chose for the Coordinator, baud rate, data size, and PAN ID are the most important!!!


XCTU
XCTU Working Modes Tools Help
Radio Modules
Radio Configuration [- 0013A20040DC8030]
Read Write Default Update Profile
Parameter

Name:
Function: ZigBee Router AT
Port: COM13 - 9600/8/N/1/N - AT
MAC: 0013A20040DC8030

GT Guard Times	3E8	x 1ms
CC Command Sequence Character	2B	Recommended: 0x20-0x7F (ASCII)

Sleep Modes
Configure low power options to support end device children

SM Sleep Mode	No Sleep (Router) [0]	
SN Number of Cyclic Sleep Periods	1	
SO Sleep Options	0	
SP Cyclic Sleep Period	20	x 10 ms
ST Time before Sleep	1388	x 1 ms
PO Poll Rate	0	

I/O Settings
Modify DIO and ADC options

D0 AD0/DIO0 Configuration	Disabled [0]	
D1 AD1/DIO1 Configuration	Disabled [0]	
D2 AD2/DIO2 Configuration	Disabled [0]	
D3 AD3/DIO3 Configuration	Disabled [0]	
D4 DIO4 Configuration	Disabled [0]	
D5 DIO5/Assoc Configuration	Associated indicator [1]	
P0 DIO10/PWM0 Configuration	RSSI PWM Output [1]	
P1 DIO11 Configuration	Disabled [0]	
P2 DIO12 Configuration	Disabled [0]	
PR Pull-up Resistor Enable	1FFF	
LT Associate LED Blink Time	0	x10 ms
RP RSSI PWM Timer	28	x 100 ms

Figure 5: Xbee/ XCTU Router settings

- Red Box Explanations
  - These are the DIO pins on your physical Xbee. See figure 6. These can be configured for several different modes but for this example we will select a digital input. DIO/ 0 was used for this experiment.

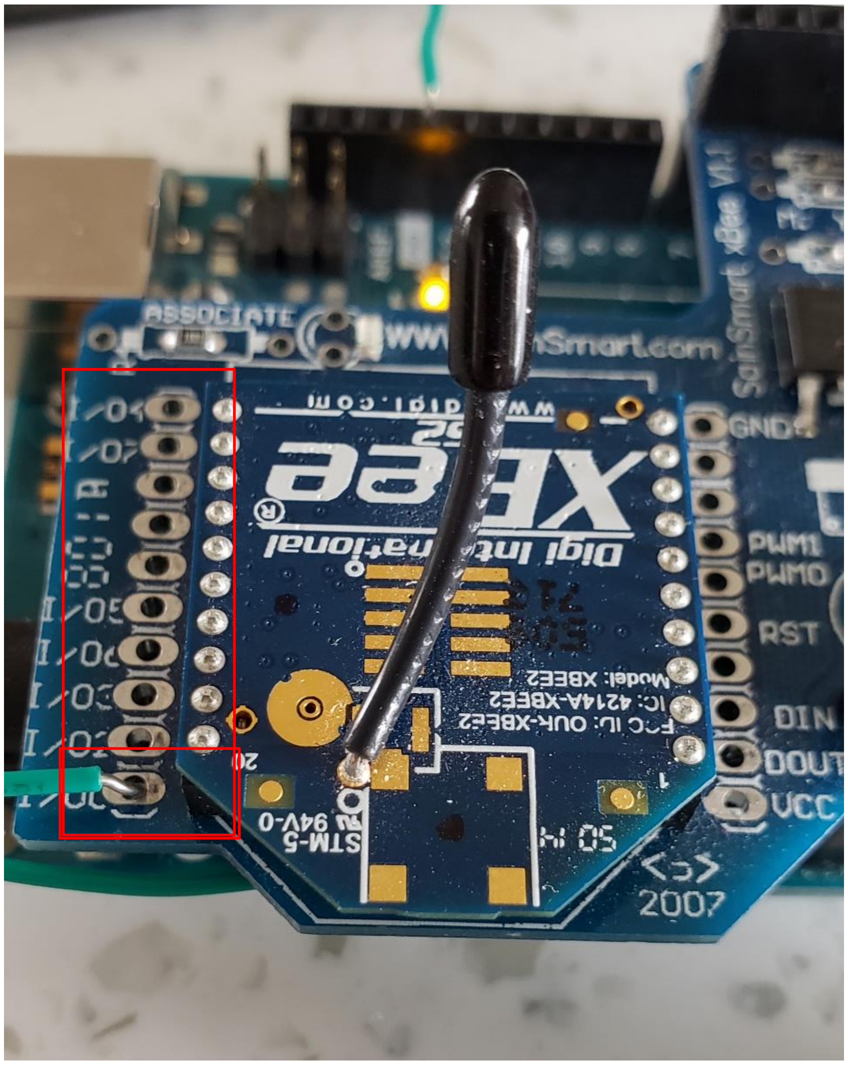


Figure 6: Xbee interface to Router (DIO pins highlighted / DIO 0)

<u>**CIRCUIT**</u>

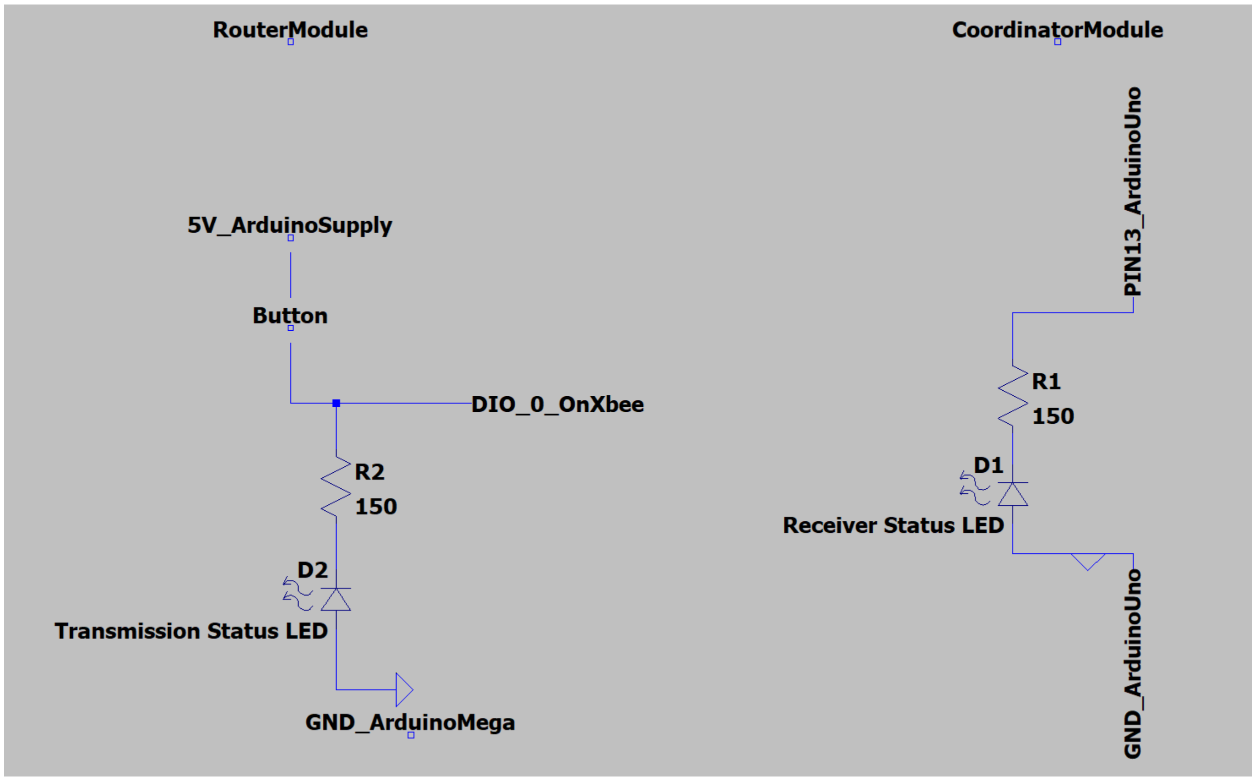


Figure 7: LT spice diagram

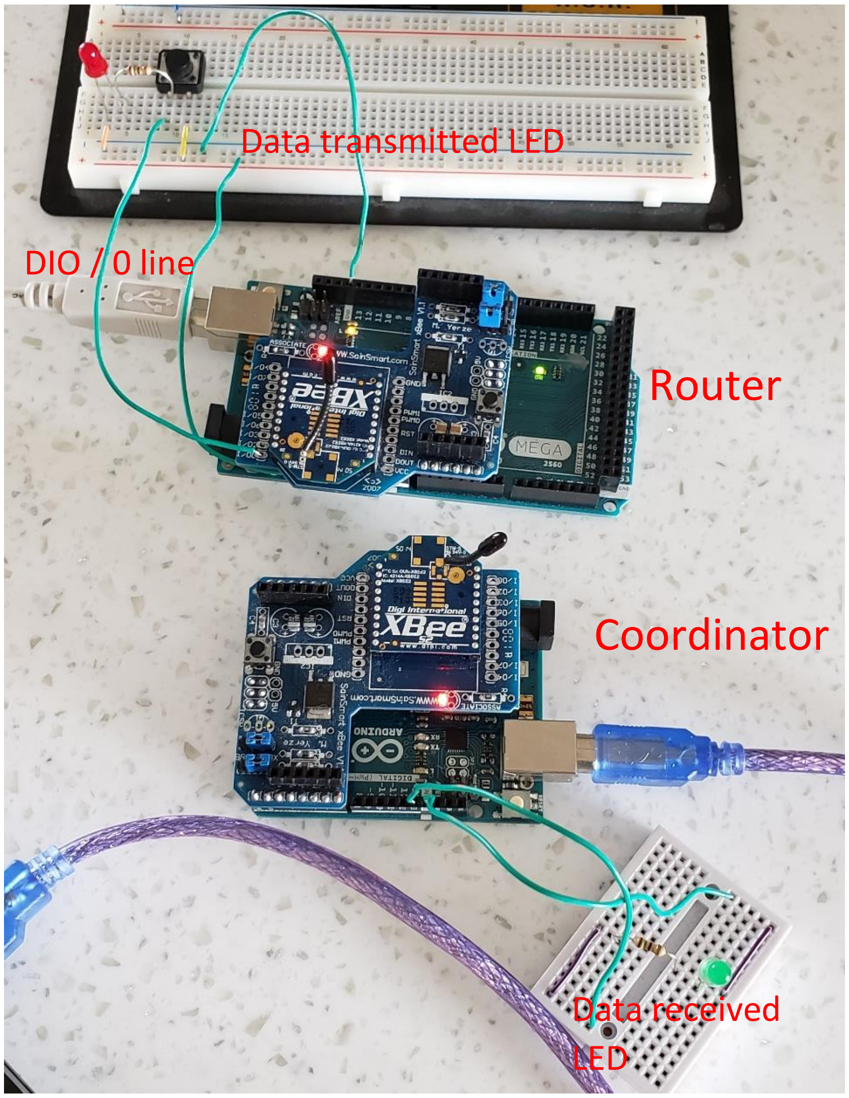


Figure 8: Circuit overview

CODE

This tutorial assumes you have basic experience in coding C, and basic knowledge of using the Arduino IDE. Therefor the code files used in this tutorial are posted on this tutorial's webpage in the documents section. The following describes some helpful tips for getting this to run on your Arduino boards.

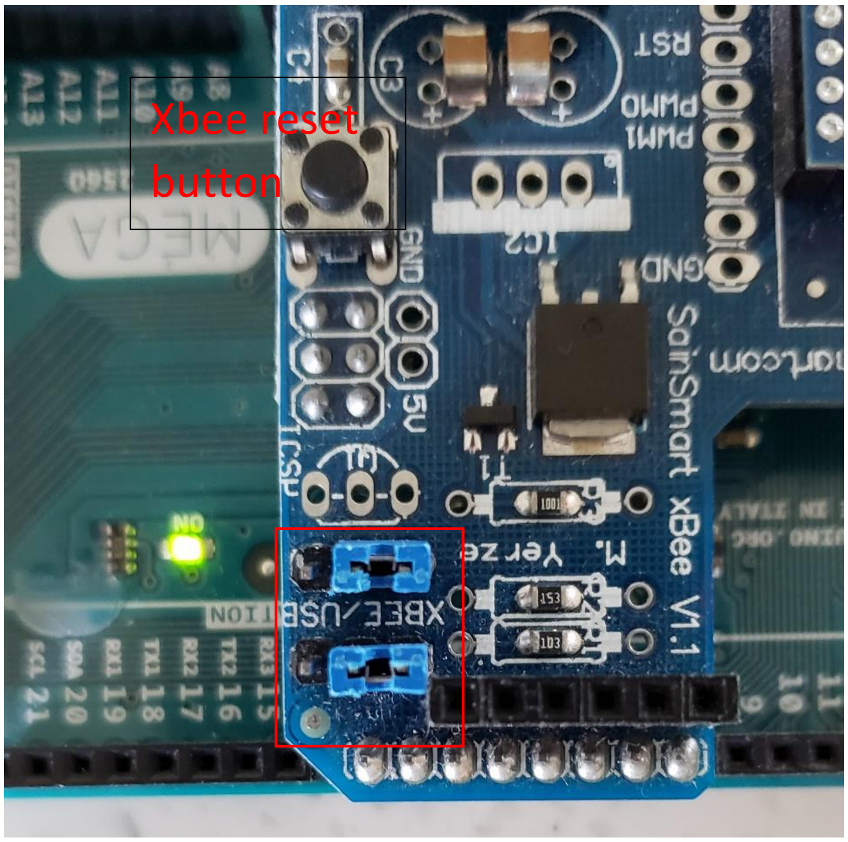


Figure 8: Xbee Jumpers

Coding Tips:

- If the Xbee's aren't syncing after deployed code try resetting them on either the Arduino board or the Xbee shield
- ***IMPORTANT*** when deploying code to the Arduino, the Xbee must be disconnected. But if your using the Xbee shield you can simply move both jumpers from Xbee to USB. This allows code to be deployed to the Arduino chip. Change their position back to Xbee when you want the code to be communicated between the Xbees.
- If not using a Xbee shield be sure to disconnect it from the Arduinos TX and RX, this could result in a deployment error.

OPERATION

The basic functionality is to click the button on the routerr end of the network of Xbee's and have a LED light up on the Coordinator side to show that the button was clicked. This signal is sent wirelessly through the Xbee's communication network. The frame data of the router is displayed through the Coordinator in API mode. That data is parsed through to see if the digital read on the router.

Check the video on the website to make to see what it should be doing!

Data bits description:

Bytes	**Frame Fields**	**Description**
7E_	Start Delimiter	Identifies the first byte of the frame.
0_12_	Length	Total bytes per frame.
92_	Frame Type	Type of API frame and how the information is organized.
0_	Frame ID	Data frame for the host to correlate with a transmit status frame (0 disables response frame).
13_A2_0_40_DC_	Dest. 64-bit Addy	64-bit address of the destination XBee.
80_30_	Dest. 16-bit Addy	16-bit address of the destination XBee.
D2_	Broadcast Radius	Maximum number of hops a broadcast transmission can occur.
74_	Options	Bitfield of supported transmission options.
1_1_0_1_0_0_0_	RF Data	Up to 255 bytes of data that is sent to the receiver XBee.
A3_	Checksum	Last byte of the frame to test data integrity with the sum of frame data bytes.

* "_" is for spacing purposes only.
All values are in hex.