ENGR 498: Design for the Internet of Things – Wireless Communications

Dr. Jason Forsyth
Department of Engineering
James Madison University

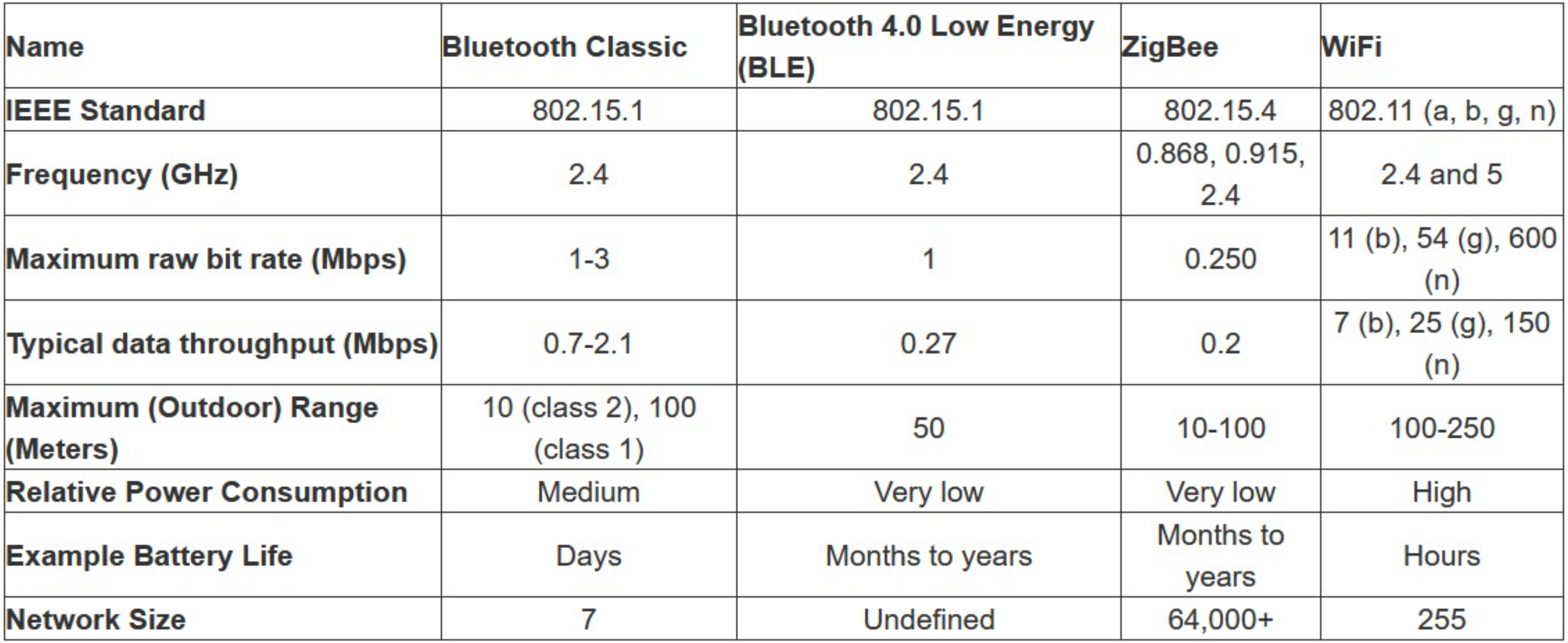
Comms

- Some applications may require data to be transmitted to a remote station. Any data link will have some *transmission rate* (bits per second) and *range* (meters).

- Communications (especially radios) are energy expensive to operate so it may be useful to have a faster transmissions rate. However, a faster radio may require more energy to operate.
    - Example: WIFI is much faster than Bluetooth, but requires more energy.

- Also, it may be more efficient to transmit information in batches rather than whenever data is received.
    - The energy to power up the transmitter may be so large that it is not worth turning on/off frequently. Send a large dataset at once.

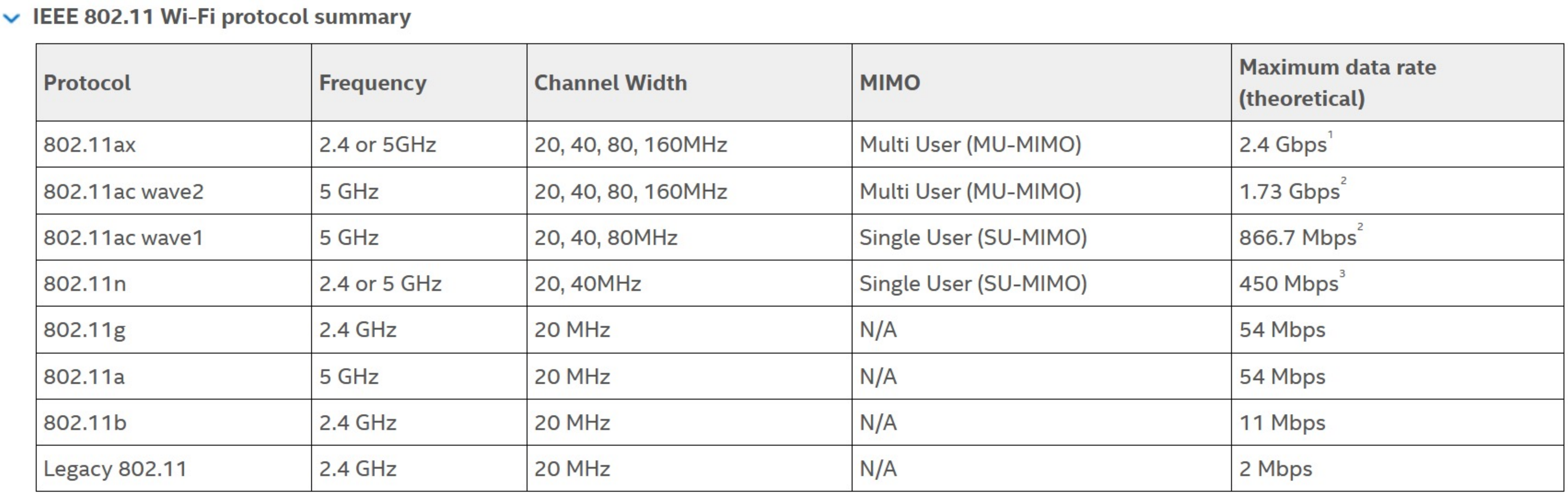
More power means faster comms and longer range. (*Not an exhaustive list*)



Name	Bluetooth Classic	Bluetooth 4.0 Low Energy (BLE)	ZigBee	WiFi
IEEE Standard	802.15.1	802.15.1	802.15.4	802.11 (a, b, g, n)
Frequency (GHz)	2.4	2.4	0.868, 0.915, 2.4	2.4 and 5
Maximum raw bit rate (Mbps)	1-3	1	0.250	11 (b), 54 (g), 600 (n)
Typical data throughput (Mbps)	0.7-2.1	0.27	0.2	7 (b), 25 (g), 150 (n)
Maximum (Outdoor) Range (Meters)	10 (class 2), 100 (class 1)	50	10-100	100-250
Relative Power Consumption	Medium	Very low	Very low	High
Example Battery Life	Days	Months to years	Months to years	Hours
Network Size	7	Undefined	64,000+	255

https://learn.sparkfun.com/tutorials/bluetooth-basics/wireless-comparison

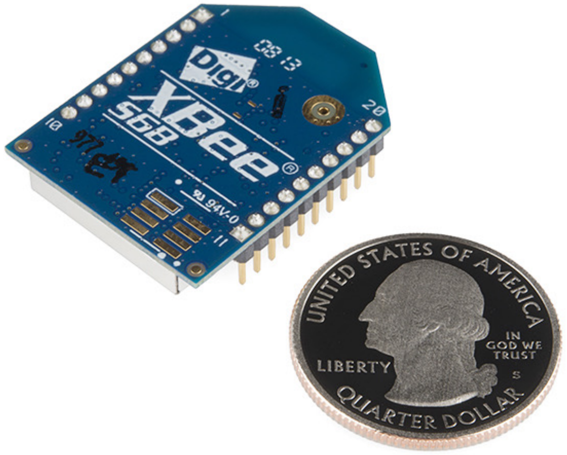
Speeds for various Wifi protocols



IEEE 802.11 Wi-Fi protocol summary

Protocol	Frequency	Channel Width	MIMO	Maximum data rate (theoretical)
802.11ax	2.4 or 5GHz	20, 40, 80, 160MHz	Multi User (MU-MIMO)	2.4 Gbps[1]
802.11ac wave2	5 GHz	20, 40, 80, 160MHz	Multi User (MU-MIMO)	1.73 Gbps[2]
802.11ac wave1	5 GHz	20, 40, 80MHz	Single User (SU-MIMO)	866.7 Mbps[2]
802.11n	2.4 or 5 GHz	20, 40MHz	Single User (SU-MIMO)	450 Mbps[3]
802.11g	2.4 GHz	20 MHz	N/A	54 Mbps
802.11a	5 GHz	20 MHz	N/A	54 Mbps
802.11b	2.4 GHz	20 MHz	N/A	11 Mbps
Legacy 802.11	2.4 GHz	20 MHz	N/A	2 Mbps

Xbee RF Module



- “Drop in” RF module built by Digi (previously Maxstream) that creates “personal area networks” (PANs).
- Can operate a variety of standard (IEEE 802.15.4) or proprietary (Zigbee, DigiMesh) protocols.
- Configuration can be complex (sample data, sleep, encryption) or simple (send direct message to location).

Images © SparkFun. Used with Creative Commons 2.0

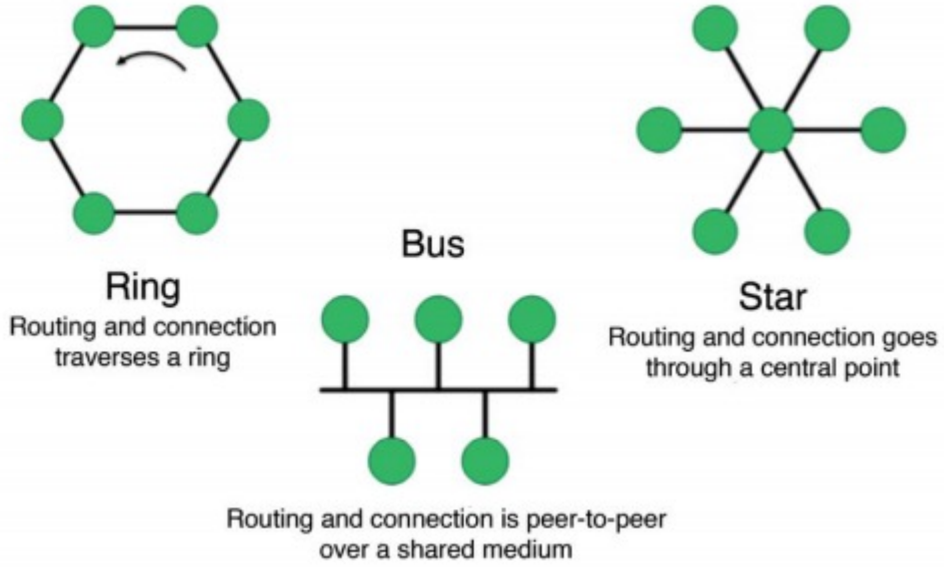
Challenges with RF Networks

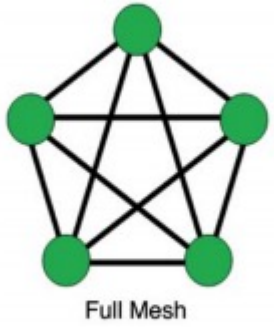
- Our Xbee uses radio frequencies (RF) as the transmission medium in the 2.4GHz band. Your laptop uses the same band for Wifi.

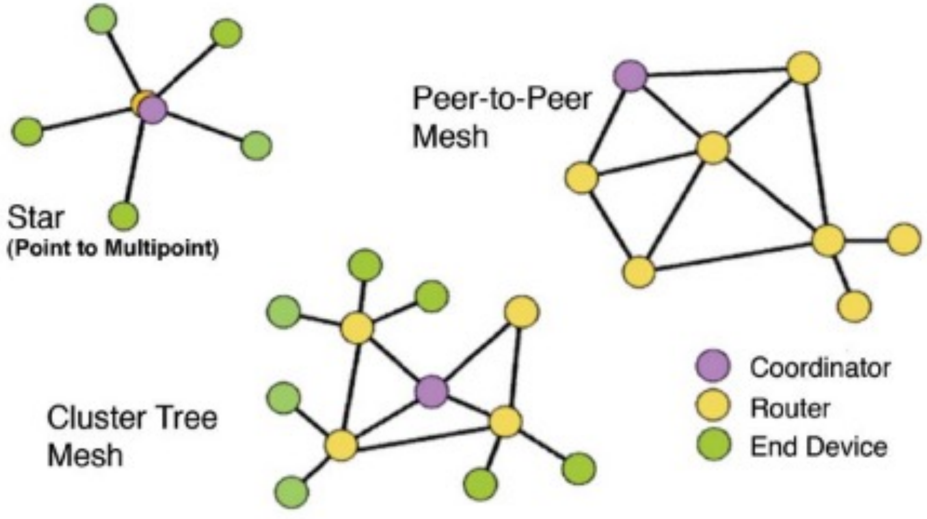
- As RF is a *broadcast* medium, each device in the network can “hear” the other messages. If the data is important then use encryption.

- Because each node hears all message a good configuration/organization is needed so nodes don’t deal with “bogus” messages.

Network Topology


Ring
Routing and connection traverses a ring
Bus
Routing and connection is peer-to-peer over a shared medium
Star
Routing and connection goes through a central point





- Communication networks can have different structures called topologies.

- Each topology has tradeoffs such as *cost* (how much material is needed), *bandwidth/latency* (how fast information is sent), *redundancy* (how resilient is the network to failures), and *scalability* (how easy is it add nodes and what happens when you do).

Digi, “Untangling the Mesh” (link)

Our Peer to Peer Xbee Network

- Current network is *peer-to-peer* where the devices establish their own network without a central/coordinating node. There is no hierarchical relationship between nodes.

- All your devices can see each other but are configured to only “talk” with one other device. Network is not *mesh* cannot self-heal or forward messages if a node drops out.

- Selected network for simplicity. This is the default configuration for the devices. There are only three settings to change on each device.

Transmitting Data to the Ground

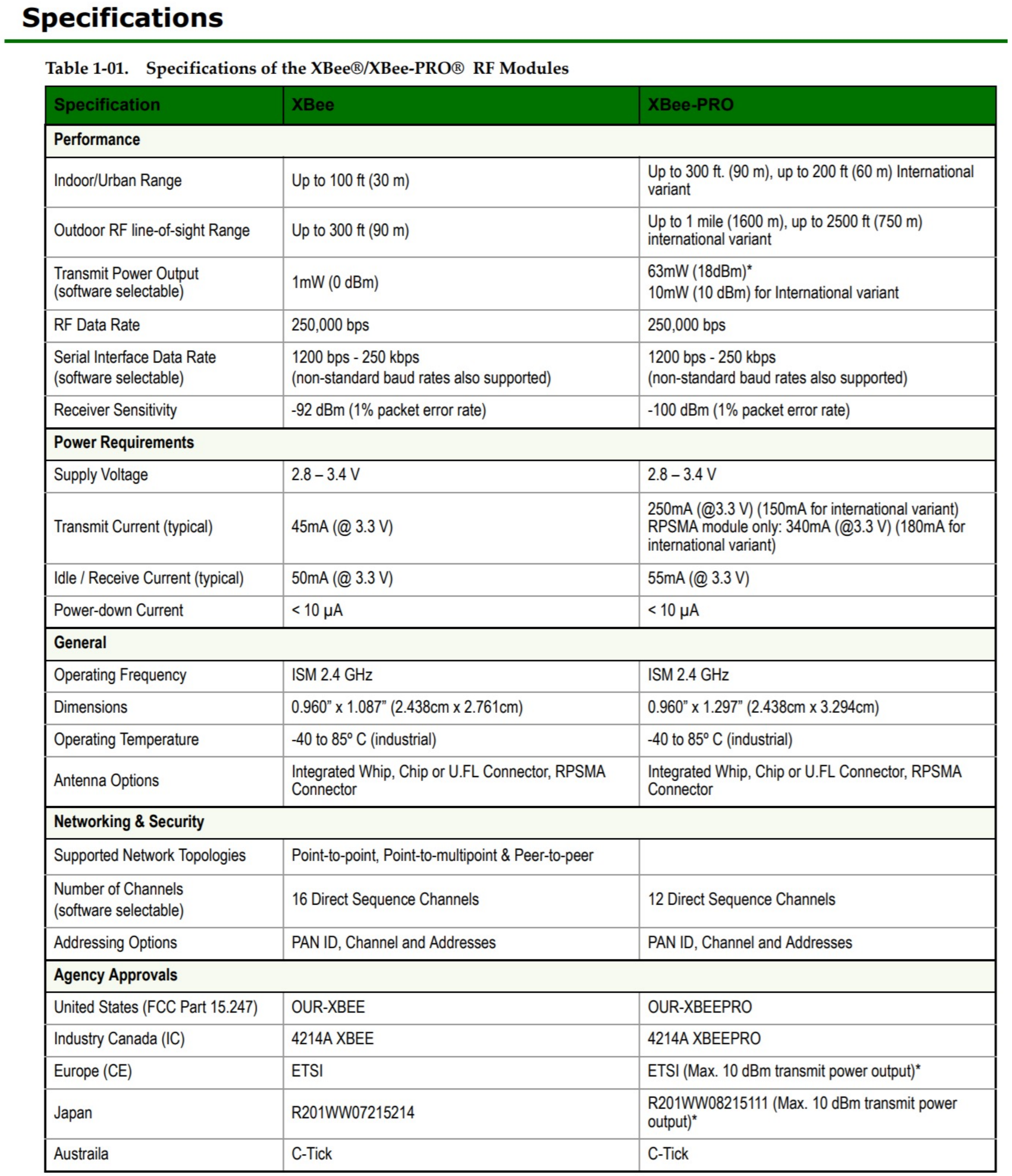
- Each team will have a “payload” and a matching “base station” Xbee. Any data sent from the payload will be received by the base station and vice versa.

- Your payload Xbee is more powerful (63 mW versus 1mW) so that it can transmit farther. Base station is less powerful and may not be able to respond to or acknowledge messages.

- Manufacturer spec says 1 mile range with outdoor line of sight.
    - Sure, we’ll see. Never trust these measurements completely.

Which parameters are important for our design?

Can you estimate their impact?



Specifications

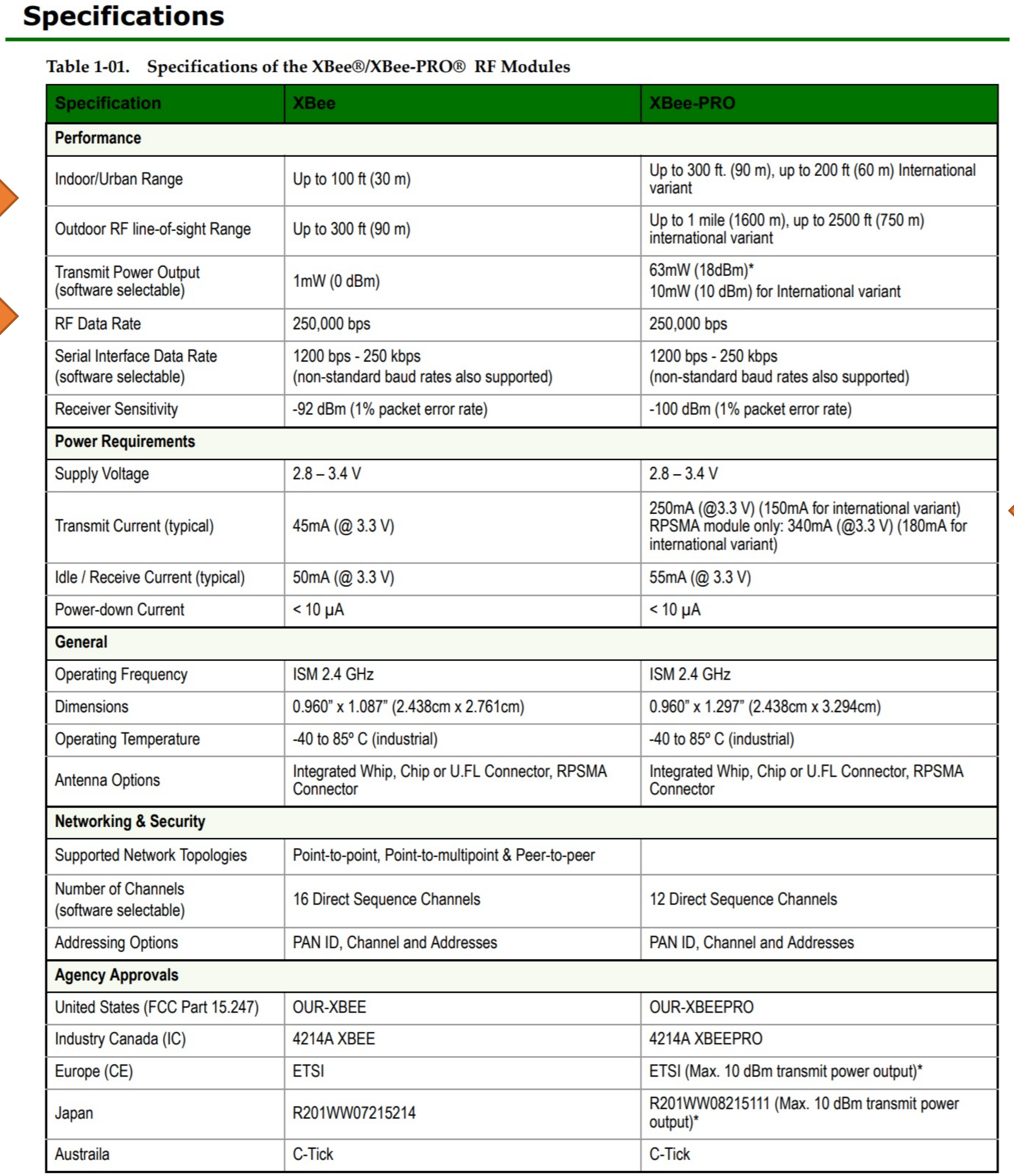
Table 1-01. Specifications of the XBee®/XBee-PRO® RF Modules

Specification	XBee	XBee-PRO
Performance		
Indoor/Urban Range	Up to 100 ft (30 m)	Up to 300 ft. (90 m), up to 200 ft (60 m) International variant
Outdoor RF line-of-sight Range	Up to 300 ft (90 m)	Up to 1 mile (1600 m), up to 2500 ft (750 m) international variant
Transmit Power Output (software selectable)	1mW (0 dBm)	63mW (18dBm)* 10mW (10 dBm) for International variant
RF Data Rate	250,000 bps	250,000 bps
Serial Interface Data Rate (software selectable)	1200 bps - 250 kbps (non-standard baud rates also supported)	1200 bps - 250 kbps (non-standard baud rates also supported)
Receiver Sensitivity	-92 dBm (1% packet error rate)	-100 dBm (1% packet error rate)
Power Requirements		
Supply Voltage	2.8 – 3.4 V	2.8 – 3.4 V
Transmit Current (typical)	45mA (@ 3.3 V)	250mA (@3.3 V) (150mA for international variant) RPSMA module only: 340mA (@3.3 V) (180mA for international variant)
Idle / Receive Current (typical)	50mA (@ 3.3 V)	55mA (@ 3.3 V)
Power-down Current	< 10 µA	< 10 µA
General		
Operating Frequency	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960" x 1.087" (2.438cm x 2.761cm)	0.960" x 1.297" (2.438cm x 3.294cm)
Operating Temperature	-40 to 85° C (industrial)	-40 to 85° C (industrial)
Antenna Options	Integrated Whip, Chip or U.FL Connector, RPSMA Connector	Integrated Whip, Chip or U.FL Connector, RPSMA Connector
Networking & Security		
Supported Network Topologies	Point-to-point, Point-to-multipoint & Peer-to-peer	
Number of Channels (software selectable)	16 Direct Sequence Channels	12 Direct Sequence Channels
Addressing Options	PAN ID, Channel and Addresses	PAN ID, Channel and Addresses
Agency Approvals		
United States (FCC Part 15.247)	OUR-XBEE	OUR-XBEEPRO
Industry Canada (IC)	4214A XBEE	4214A XBEEPRO
Europe (CE)	ETSI	ETSI (Max. 10 dBm transmit power output)*
Japan	R201WW07215214	R201WW08215111 (Max. 10 dBm transmit power output)*
Austraila	C-Tick	C-Tick

SparkFun, “Xbee Datasheet” (link)

How far we can go.

How fast we can send back information.



Specifications

Table 1-01. Specifications of the XBee®/XBee-PRO® RF Modules

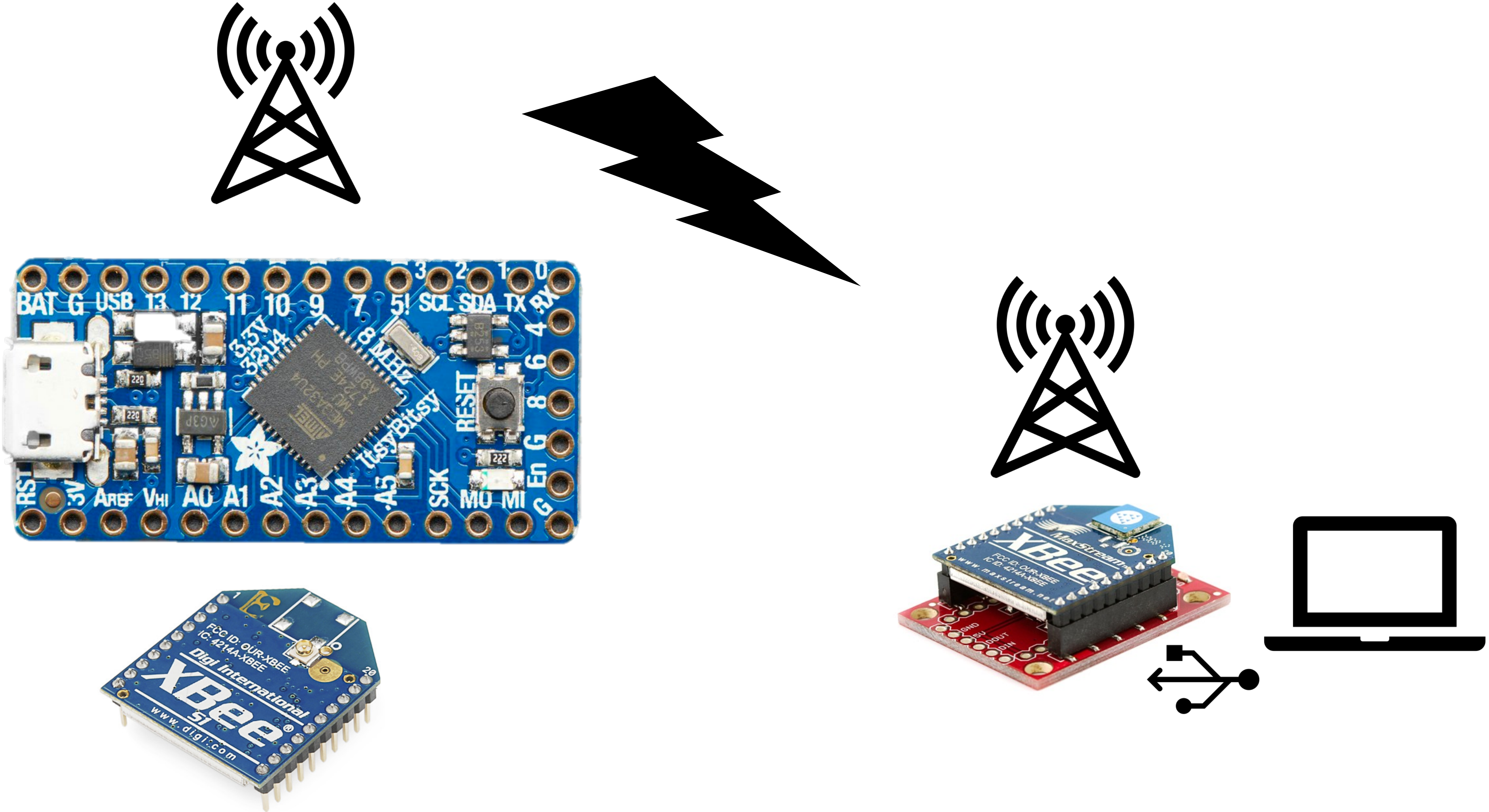
Specification	XBee	XBee-PRO
Performance		
Indoor/Urban Range	Up to 100 ft (30 m)	Up to 300 ft. (90 m), up to 200 ft (60 m) International variant
Outdoor RF line-of-sight Range	Up to 300 ft (90 m)	Up to 1 mile (1600 m), up to 2500 ft (750 m) international variant
Transmit Power Output (software selectable)	1mW (0 dBm)	63mW (18dBm)* 10mW (10 dBm) for International variant
RF Data Rate	250,000 bps	250,000 bps
Serial Interface Data Rate (software selectable)	1200 bps - 250 kbps (non-standard baud rates also supported)	1200 bps - 250 kbps (non-standard baud rates also supported)
Receiver Sensitivity	-92 dBm (1% packet error rate)	-100 dBm (1% packet error rate)
Power Requirements		
Supply Voltage	2.8 – 3.4 V	2.8 – 3.4 V
Transmit Current (typical)	45mA (@ 3.3 V)	250mA (@3.3 V) (150mA for international variant) RPSMA module only: 340mA (@3.3 V) (180mA for international variant)
Idle / Receive Current (typical)	50mA (@ 3.3 V)	55mA (@ 3.3 V)
Power-down Current	< 10 µA	< 10 µA
General		
Operating Frequency	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960" x 1.087" (2.438cm x 2.761cm)	0.960" x 1.297" (2.438cm x 3.294cm)
Operating Temperature	-40 to 85° C (industrial)	-40 to 85° C (industrial)
Antenna Options	Integrated Whip, Chip or U.FL Connector, RPSMA Connector	Integrated Whip, Chip or U.FL Connector, RPSMA Connector
Networking & Security		
Supported Network Topologies	Point-to-point, Point-to-multipoint & Peer-to-peer	
Number of Channels (software selectable)	16 Direct Sequence Channels	12 Direct Sequence Channels
Addressing Options	PAN ID, Channel and Addresses	PAN ID, Channel and Addresses
Agency Approvals		
United States (FCC Part 15.247)	OUR-XBEE	OUR-XBEEPRO
Industry Canada (IC)	4214A XBEE	4214A XBEEPRO
Europe (CE)	ETSI	ETSI (Max. 10 dBm transmit power output)*
Japan	R201WW07215214	R201WW08215111 (Max. 10 dBm transmit power output)*
Austraila	C-Tick	C-Tick

How long we can run on a battery.

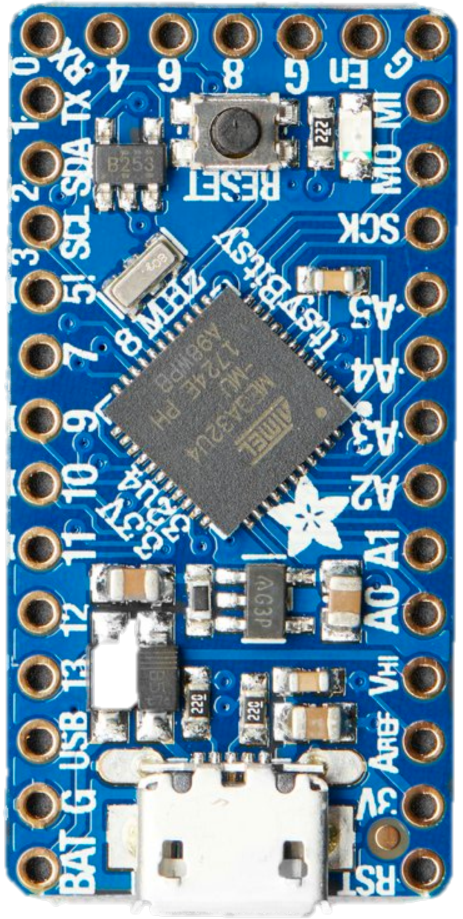
How big our box can be.

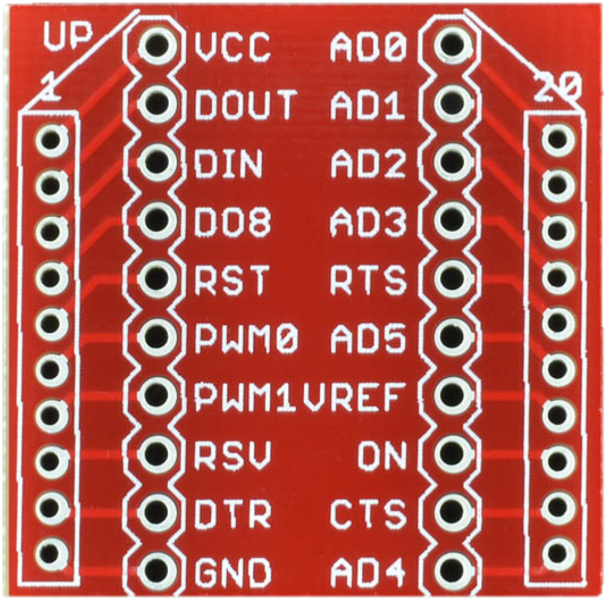
SparkFun, "Xbee Datasheet" (link)

Adding Radios to Arduinos

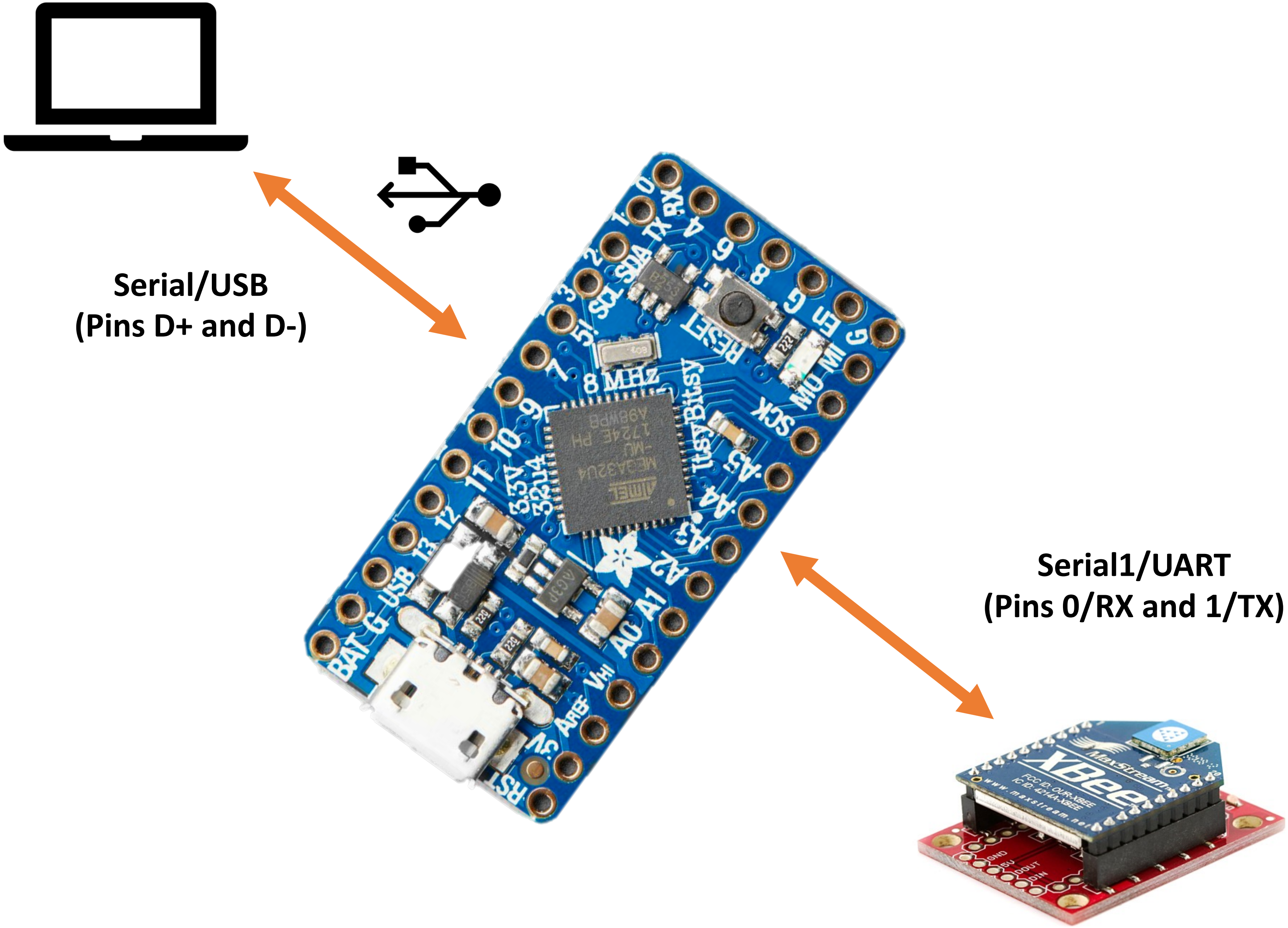

ItsyBitsy
XBee
Digi International
www.digi.com
MaxStream
www.maxstream.net

Required Connections



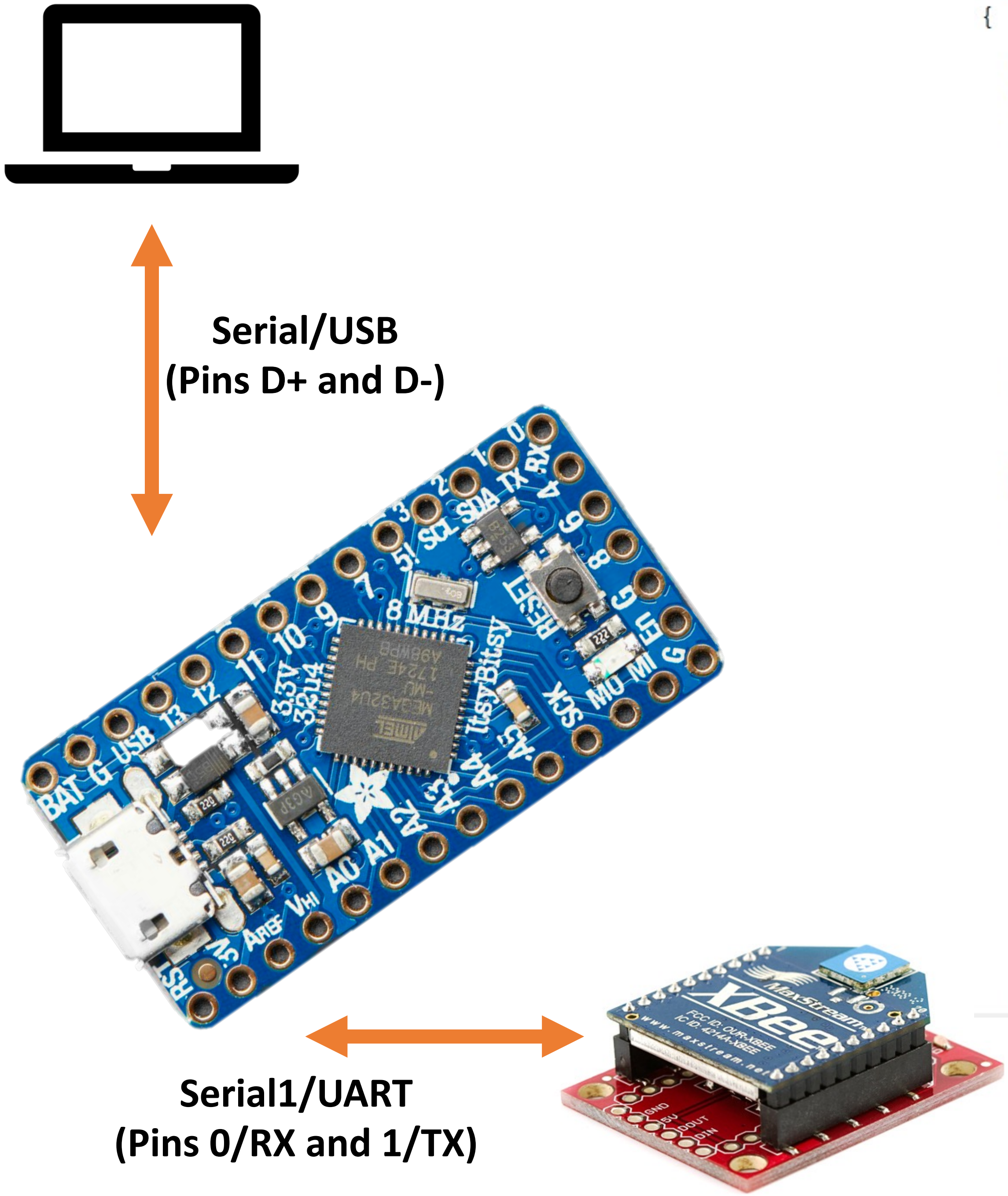


Xbee Pin	Arduino Pin
VCC	3V
GND	G
DOUT	RX
DIN	TX



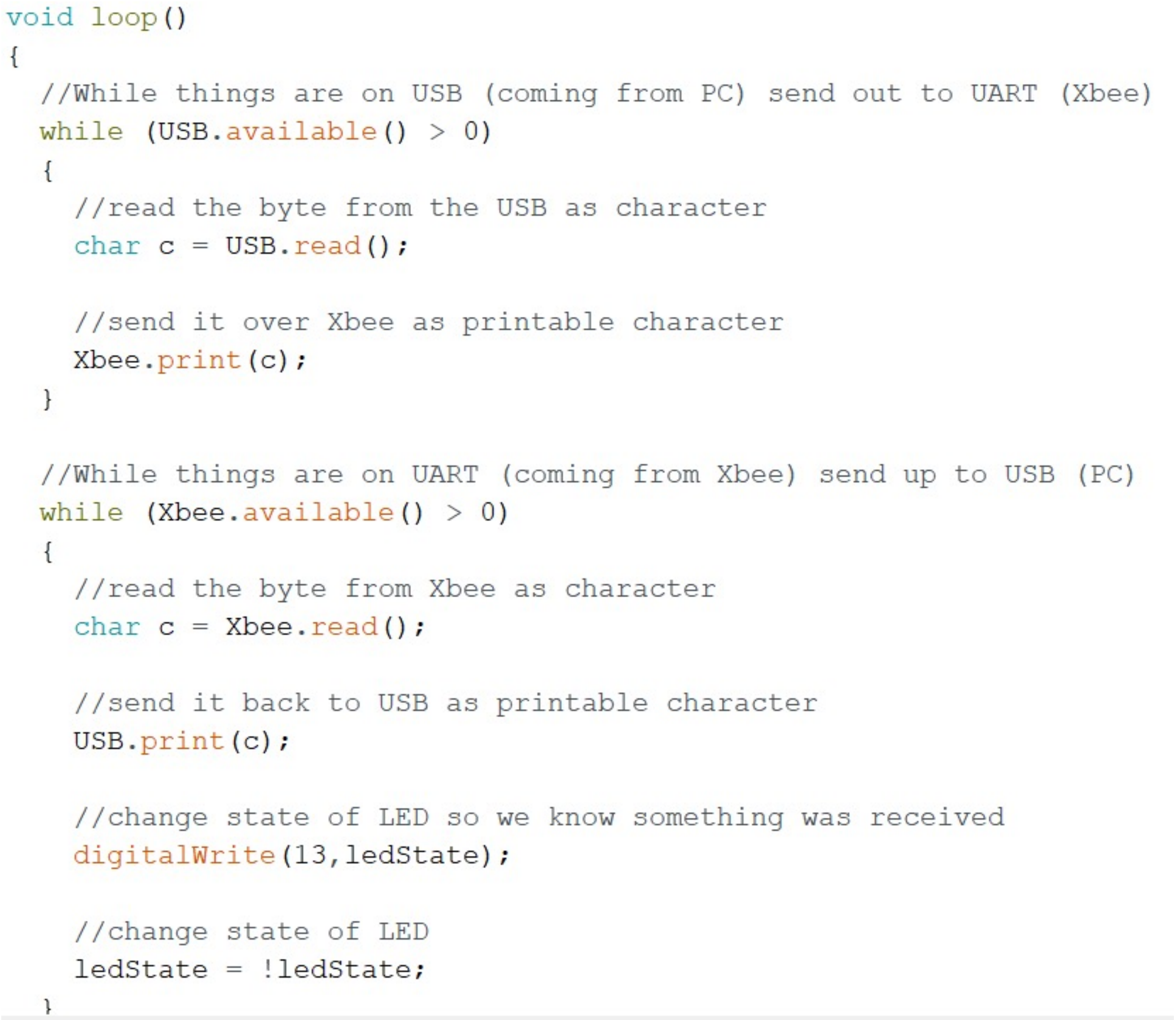
Serial/USB
(Pins D+ and D-)

Serial1/UART
(Pins 0/RX and 1/TX)



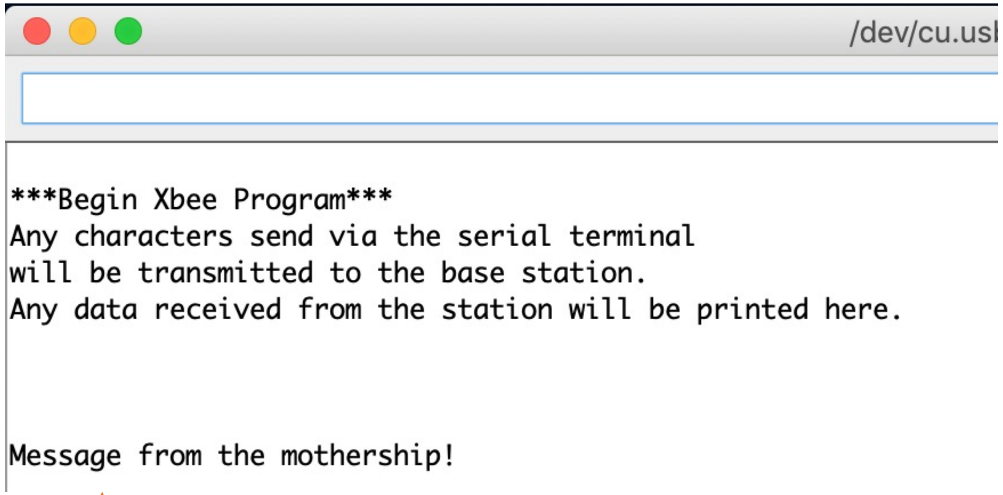
Serial/USB
(Pins D+ and D-)

Serial1/UART
(Pins 0/RX and 1/TX)



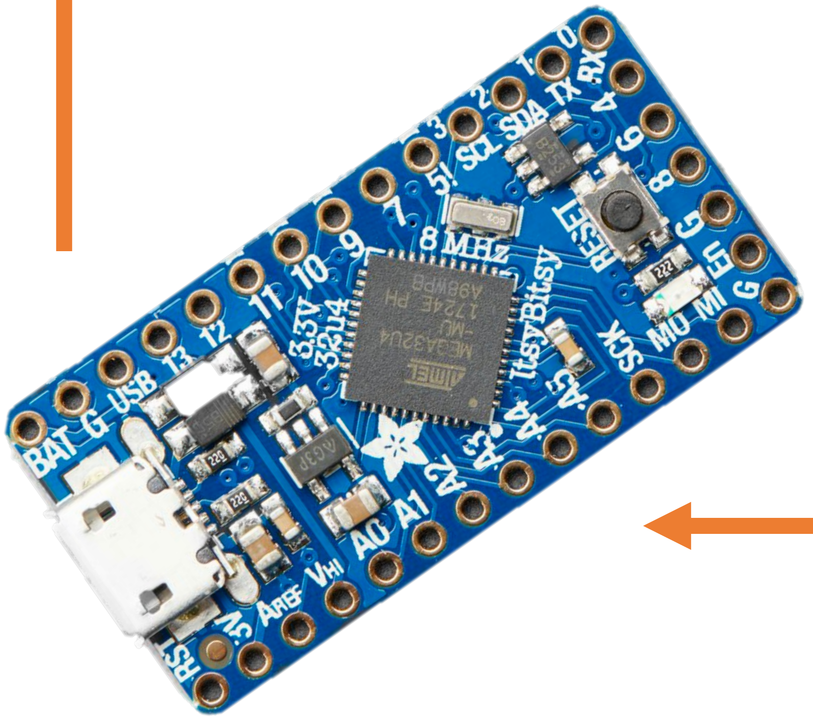
```
void loop()
{
  //While things are on USB (coming from PC) send out to UART (Xbee)
  while (USB.available() > 0)
  {
    //read the byte from the USB as character
    char c = USB.read();

    //send it over Xbee as printable character
    Xbee.print(c);
  }

  //While things are on UART (coming from Xbee) send up to USB (PC)
  while (Xbee.available() > 0)
  {
    //read the byte from Xbee as character
    char c = Xbee.read();

    //send it back to USB as printable character
    USB.print(c);

    //change state of LED so we know something was received
    digitalWrite(13,ledState);

    //change state of LED
    ledState = !ledState;
  }
```

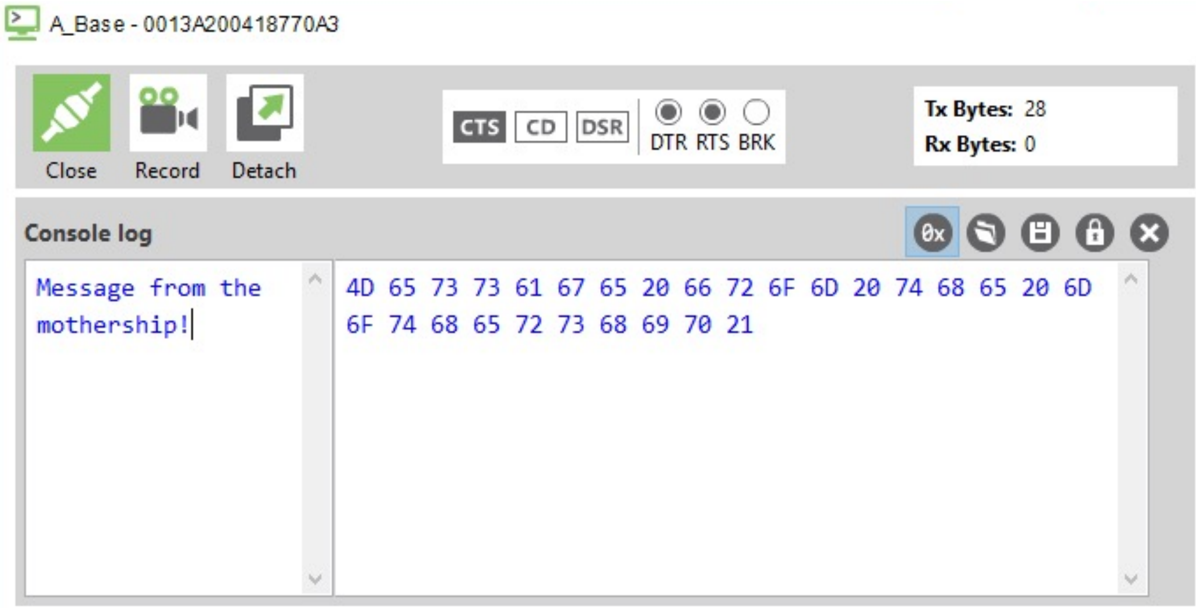
Transmission from the Base station to the Arduino.



```
***Begin Xbee Program***
Any characters send via the serial terminal
will be transmitted to the base station.
Any data received from the station will be printed here.



Message from the mothership!
```

Received in Serial Terminal





A_Base - 0013A200418770A3

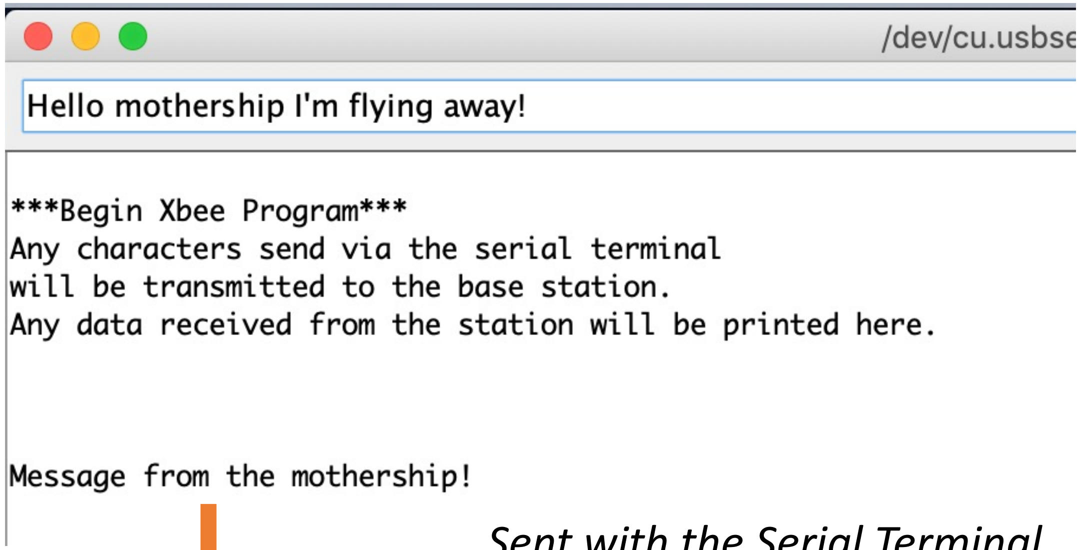
Tx Bytes: 28
Rx Bytes: 0

Console log

```
Message from the     4D 65 73 73 61 67 65 20 66 72 6F 6D 20 74 68 65 20 6D
mothership!          6F 74 68 65 72 73 68 69 70 21
```

Message sent from the base station to the Arduino using the XCTU Software (we'll cover that in a second).

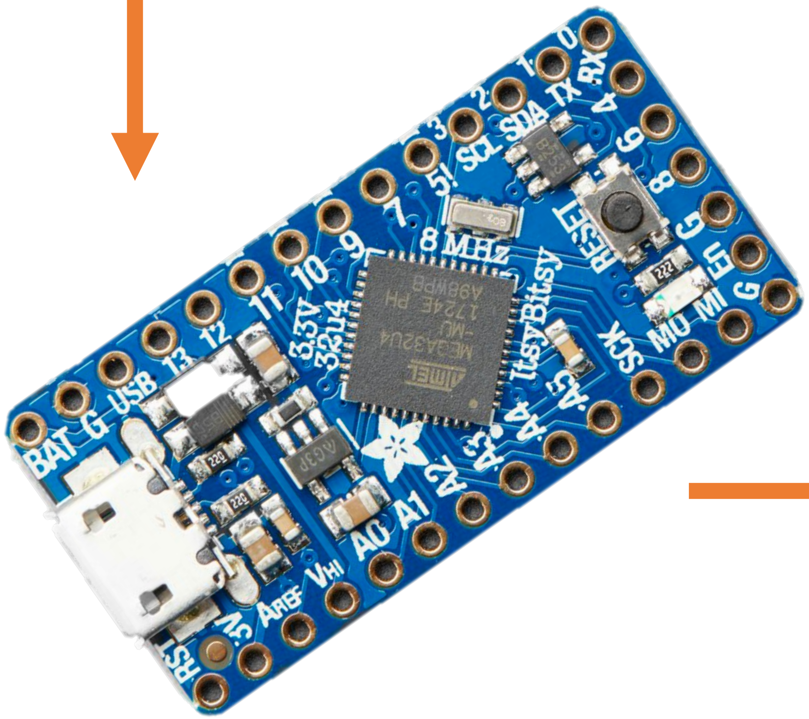
Transmission from the Arduino to the Base Station.

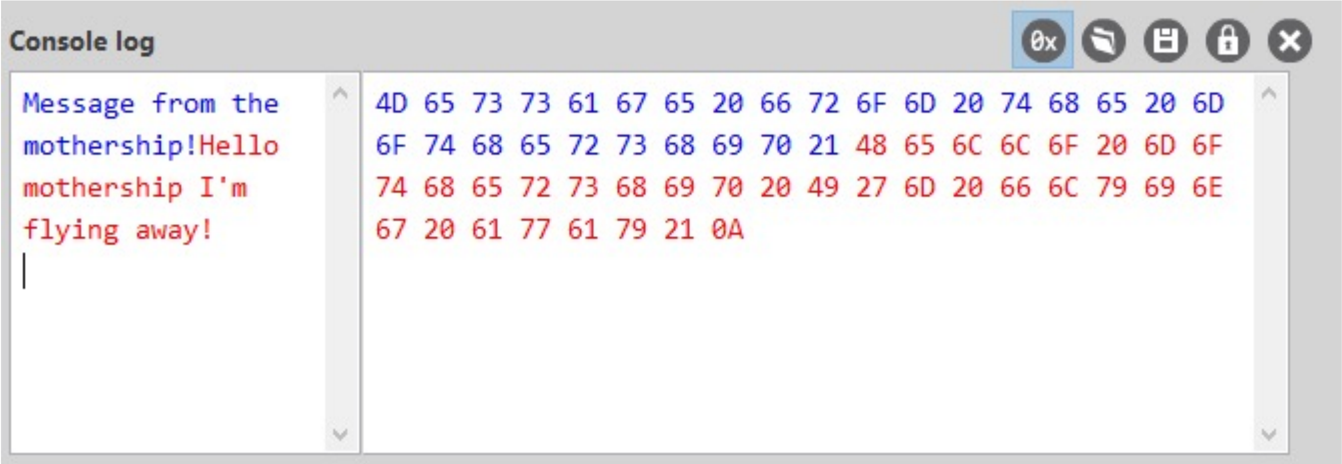


```
Hello mothership I'm flying away!
```

```
***Begin Xbee Program***
Any characters send via the serial terminal
will be transmitted to the base station.
Any data received from the station will be printed here.



Message from the mothership!
```

Sent with the Serial Terminal

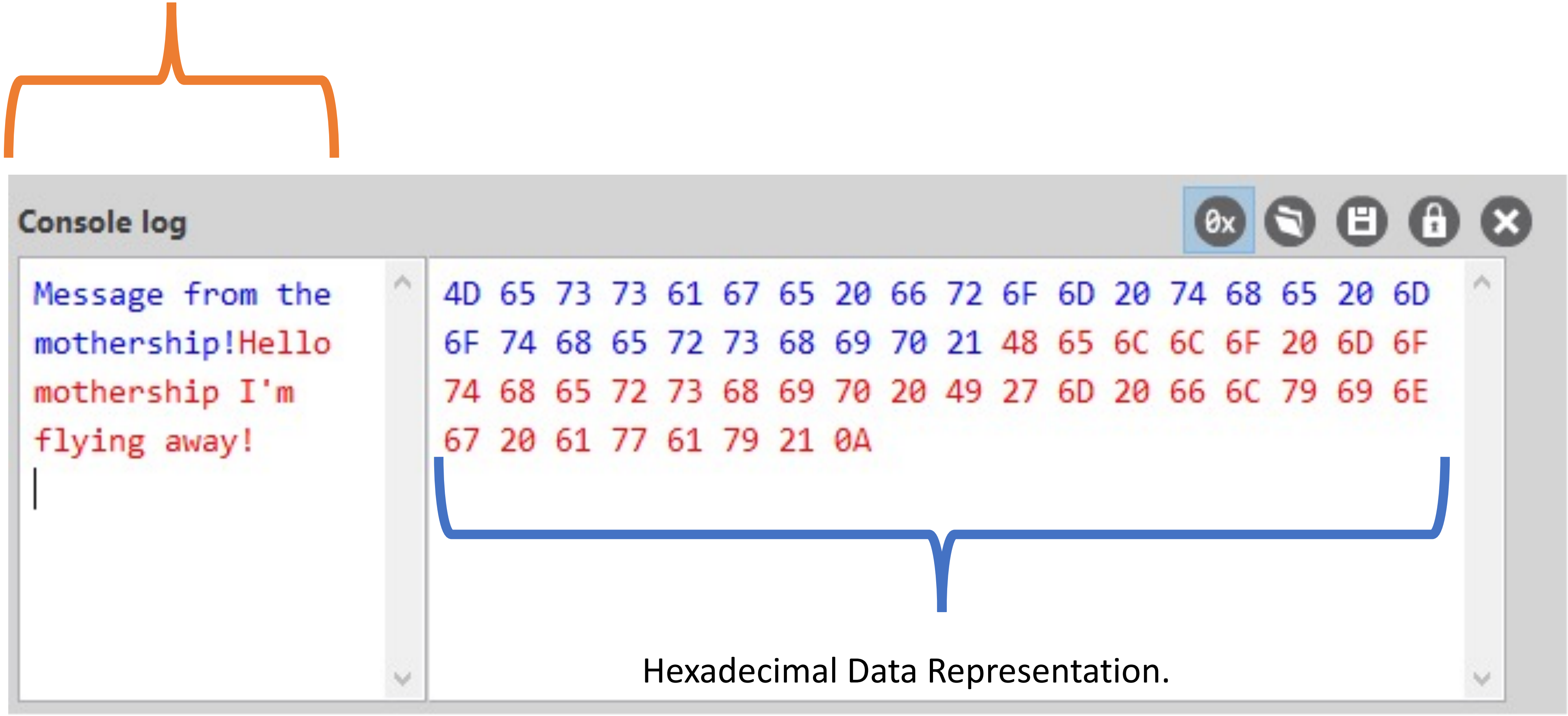




```
Message from the        4D 65 73 73 61 67 65 20 66 72 6F 6D 20 74 68 65 20 6D
mothership!Hello        6F 74 68 65 72 73 68 69 70 21 48 65 6C 6C 6F 20 6D 6F
mothership I'm          74 68 65 72 73 68 69 70 20 49 27 6D 20 66 6C 79 69 6E
flying away!            67 20 61 77 61 79 21 0A
```

Message received from the Arduino. Sent/TX data is Blue. Received/RX data is Red.

ASCII/Text of Data



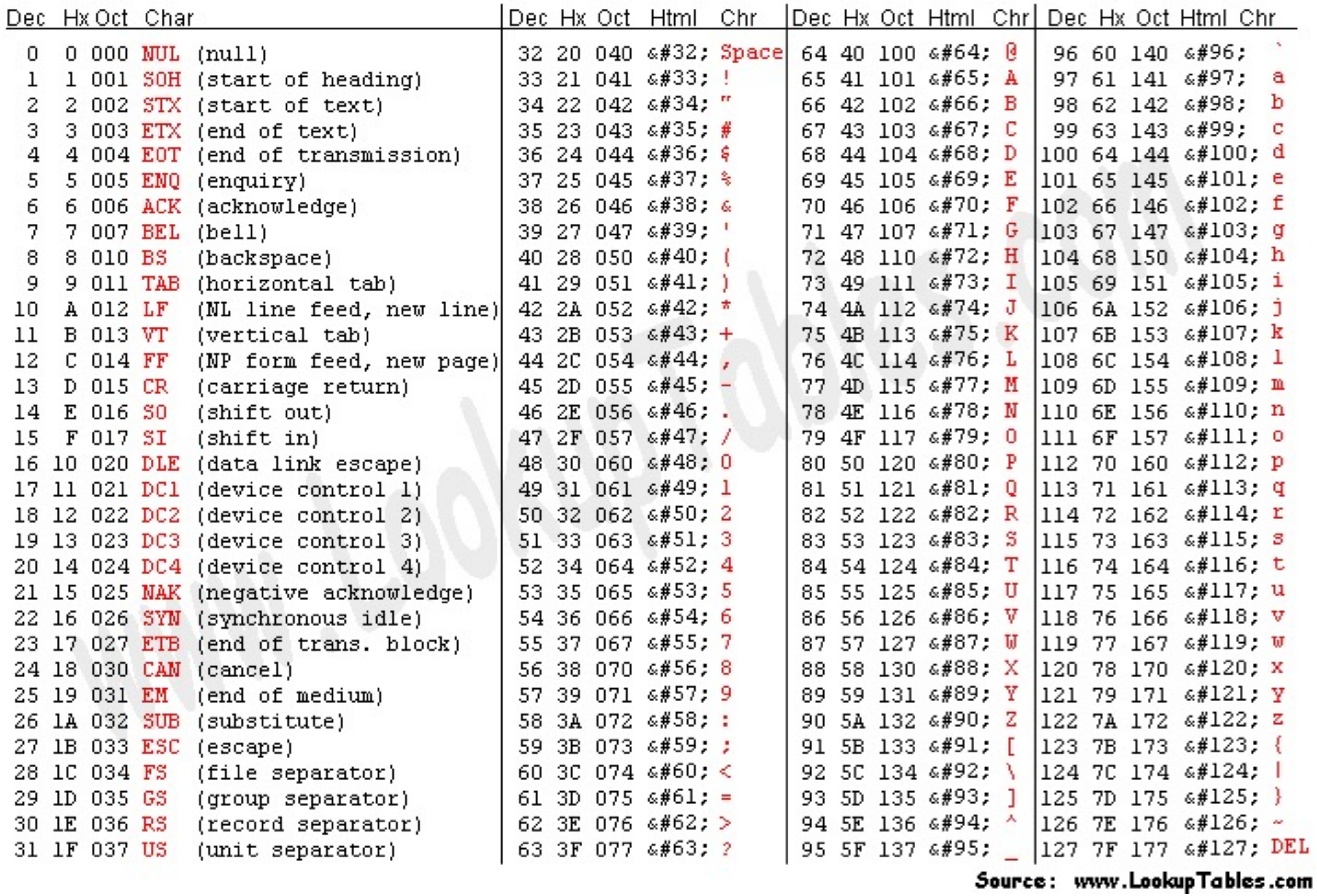
```
Message from the
mothership!Hello
mothership I'm
flying away!
```

```
4D 65 73 73 61 67 65 20 66 72 6F 6D 20 74 68 65 20 6D
6F 74 68 65 72 73 68 69 70 21 48 65 6C 6C 6F 20 6D 6F
74 68 65 72 73 68 69 70 20 49 27 6D 20 66 6C 79 69 6E
67 20 61 77 61 79 21 0A
```

Hexadecimal Data Representation.

When information is transmitted on the network it is represented/encoded in a binary (1001010) format called Hexademical. In hex all letters [A-F] and numbers [0-9] represent a 4-bit binary value. Thus 0x1B is 0001 1011_2

Each character your send has an ASCII encoding that specifies its hexadecimal equivalent. The letter ‘a’ is 0x61 and ‘M’ is 0x4D.



Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH	(start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX	(start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX	(end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT	(end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL	(bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS	(backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR	(carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO	(shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI	(shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE	(data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1	(device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2	(device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3	(device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4	(device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK	(negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN	(synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB	(end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN	(cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM	(end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB	(substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC	(escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS	(file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS	(group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS	(record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US	(unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

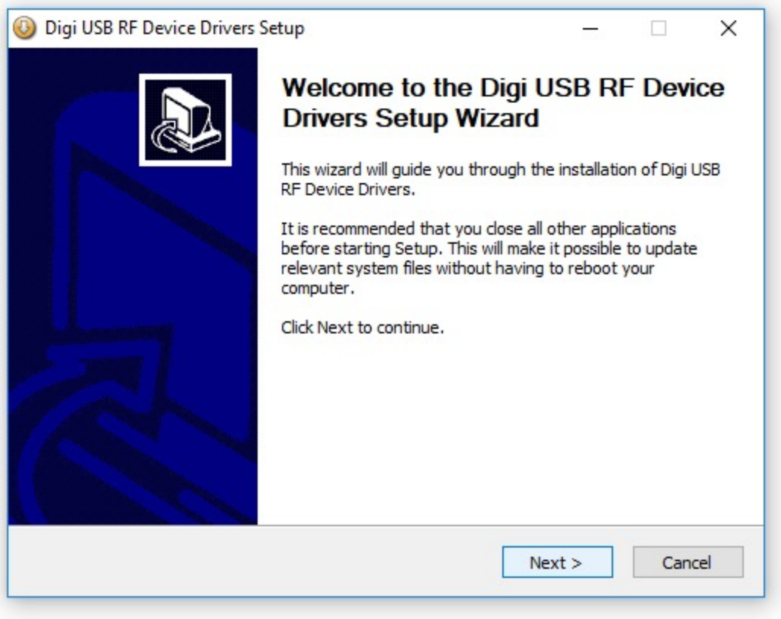
Source: www.LookupTables.com

www.asciitable.com

Setting Up Your Xbee and Running Demos

Drivers, Software, and All That Jazz…

- Select a member of your group to be the "base station" for the Xbee. They will need to install the Xbee Drivers and the XCTU software. Both are available on Canvas under Files/Software/



Digi USB RF Device Drivers Setup

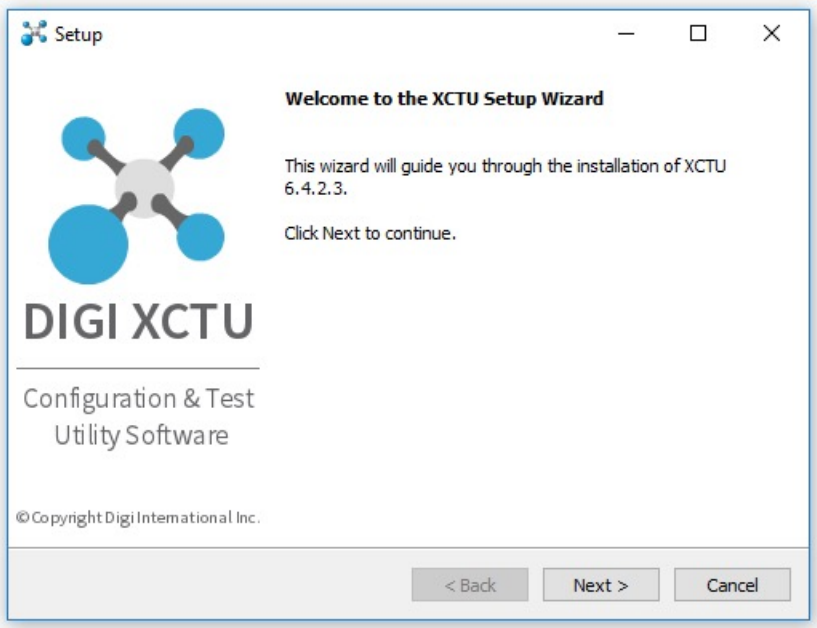
Welcome to the Digi USB RF Device Drivers Setup Wizard

This wizard will guide you through the installation of Digi USB RF Device Drivers.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Installing Xbee/Digi Drivers



Setup

Welcome to the XCTU Setup Wizard

This wizard will guide you through the installation of XCTU 6.4.2.3.

Click Next to continue.

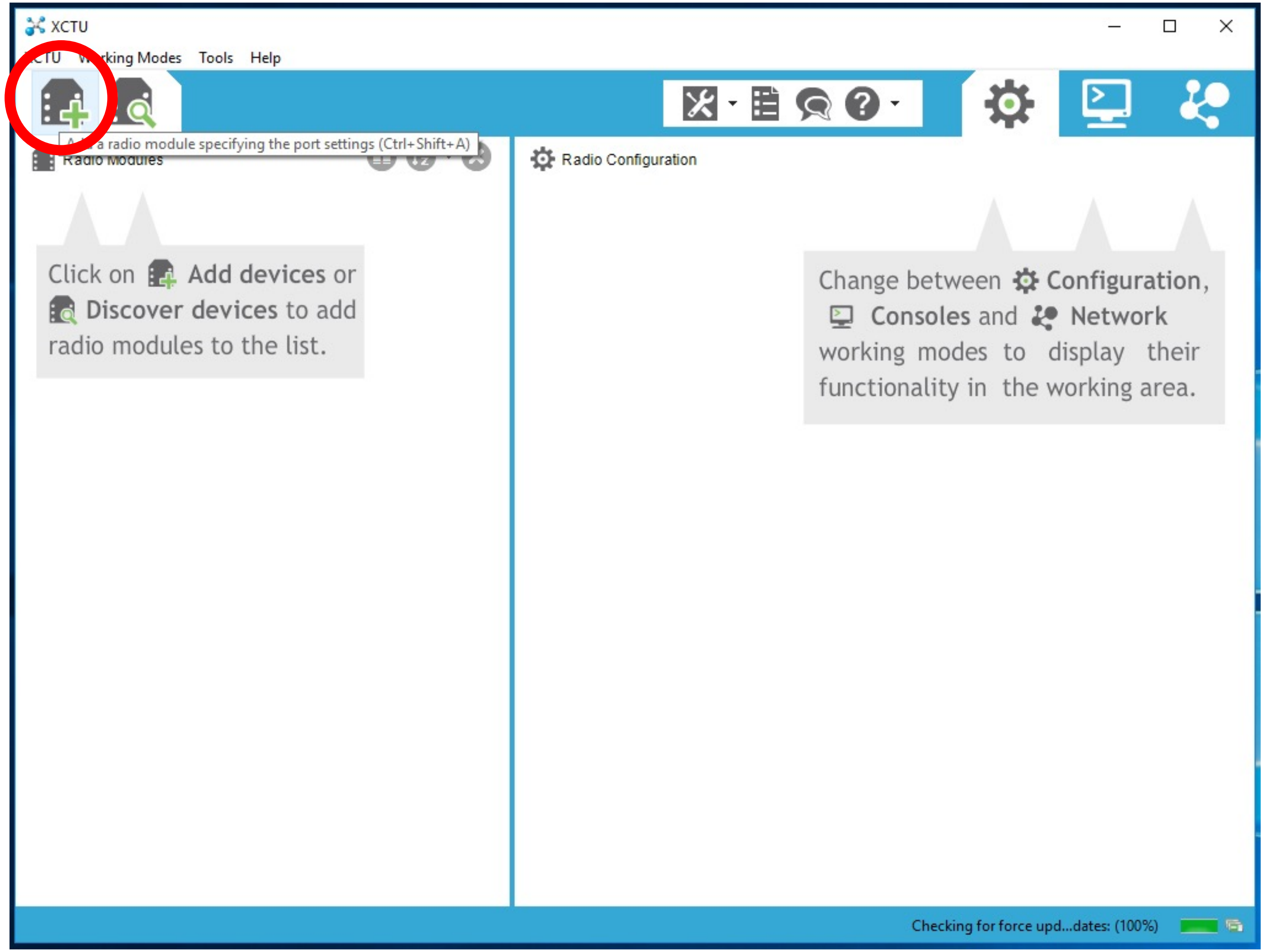
DIGI XCTU
Configuration & Test Utility Software
©Copyright Digi International Inc.
< Back
Next >
Cancel

Install XCTU Software

Connecting to your Base Station with XCTU

Click here to ADD an Xbee.

Will need to select the Serial/COM port on your laptop to which it is connected.



A a radio module specifying the port settings (Ctrl+Shift+A)

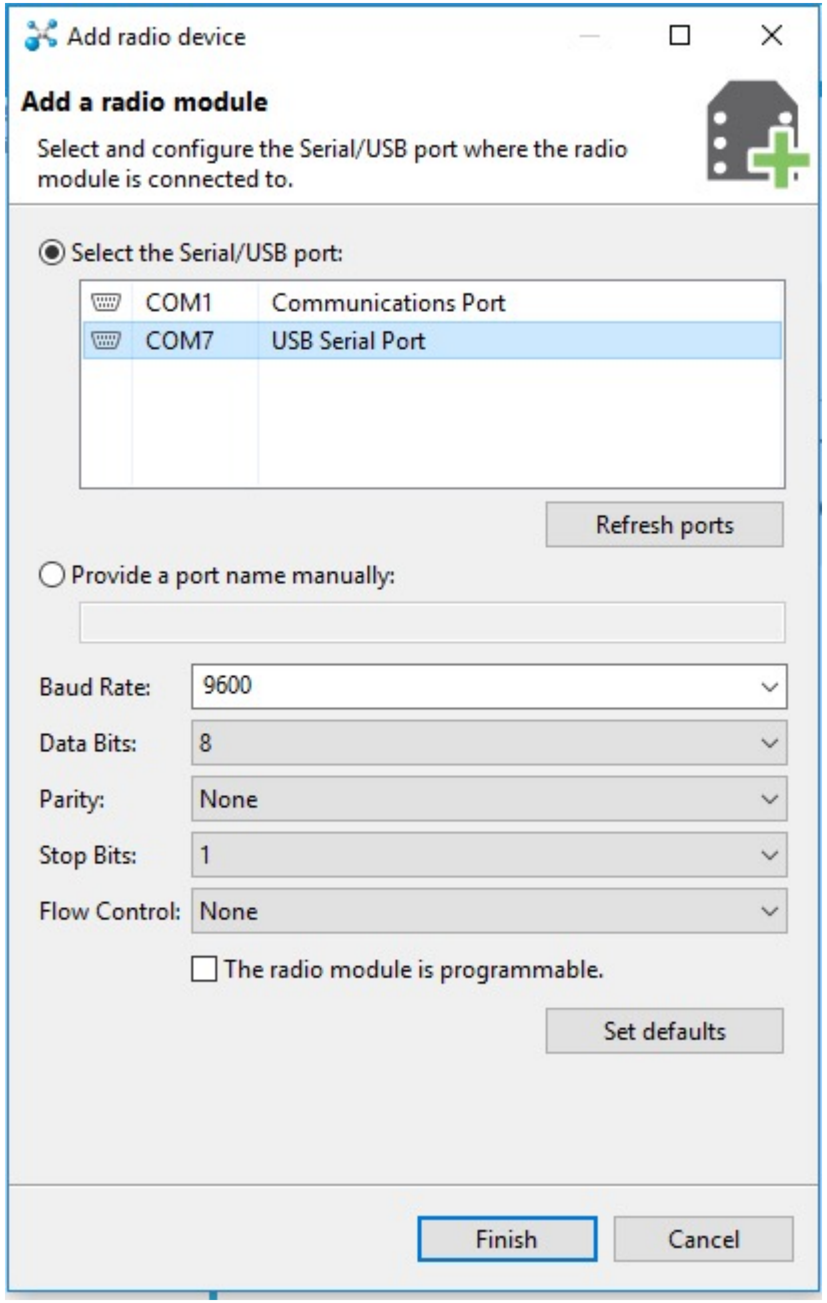
Click on Add devices or Discover devices to add radio modules to the list.

Change between Configuration, Consoles and Network working modes to display their functionality in the working area.

Checking for force upd...dates: (100%)

Pick Your COM Port

- For Windows, select the "highest" number COM port. For OSX, select the one called "usb-serial…", not "Air pods".
- If you have your Arduino connected to this computer (or other devices) there may be multiple COM ports. Try it until it works.
- Leave the communication (Baud Rate, Data Bits…etc.) information alone.



Add radio device

Add a radio module

Select and configure the Serial/USB port where the radio module is connected to.

● Select the Serial/USB port:

COM1	Communications Port
COM7	USB Serial Port

○ Provide a port name manually:

Baud Rate: 9600

Data Bits: 8

Parity: None

Stop Bits: 1

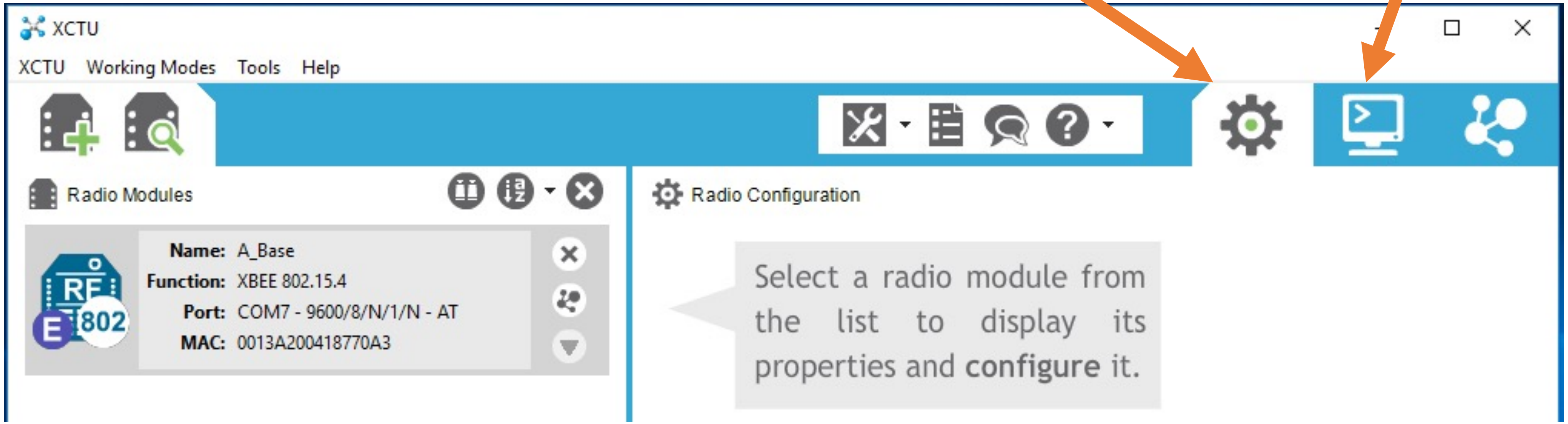
Flow Control: None

□ The radio module is programmable.

Added/Found Xbee

Serial Terminal: how we will send/receive data on the base station

Configuration Button: to show/write network parameters. You can look but don't touch!

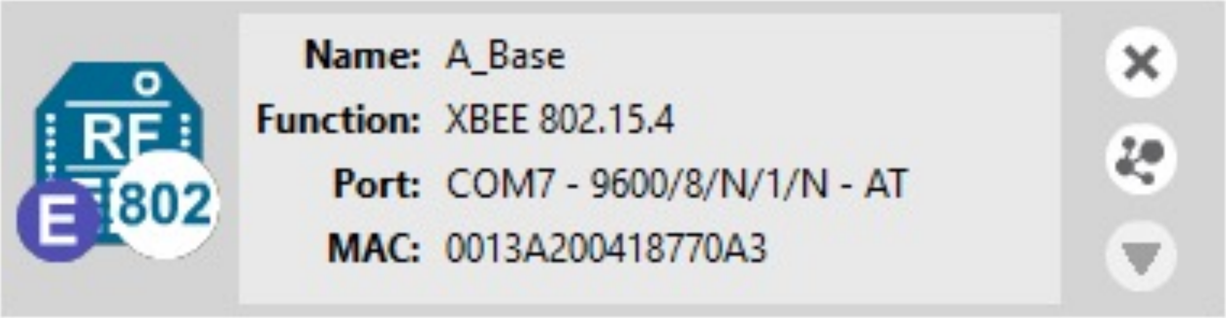


Name: A_Base
Function: XBEE 802.15.4
Port: COM7 - 9600/8/N/1/N - AT
MAC: 0013A200418770A3

Select a radio module from the list to display its properties and **configure** it.

Xbee Detail

- **Name**: A Network Identifier that can be used to “lookup” your Xbee
- **Function**: Currently installed firmware/standard.
- **Port**: Attached Serial port on your computer and associated data rate information
- **MAC**: Media Access Control address. A unique address assigned by the manufacturer.

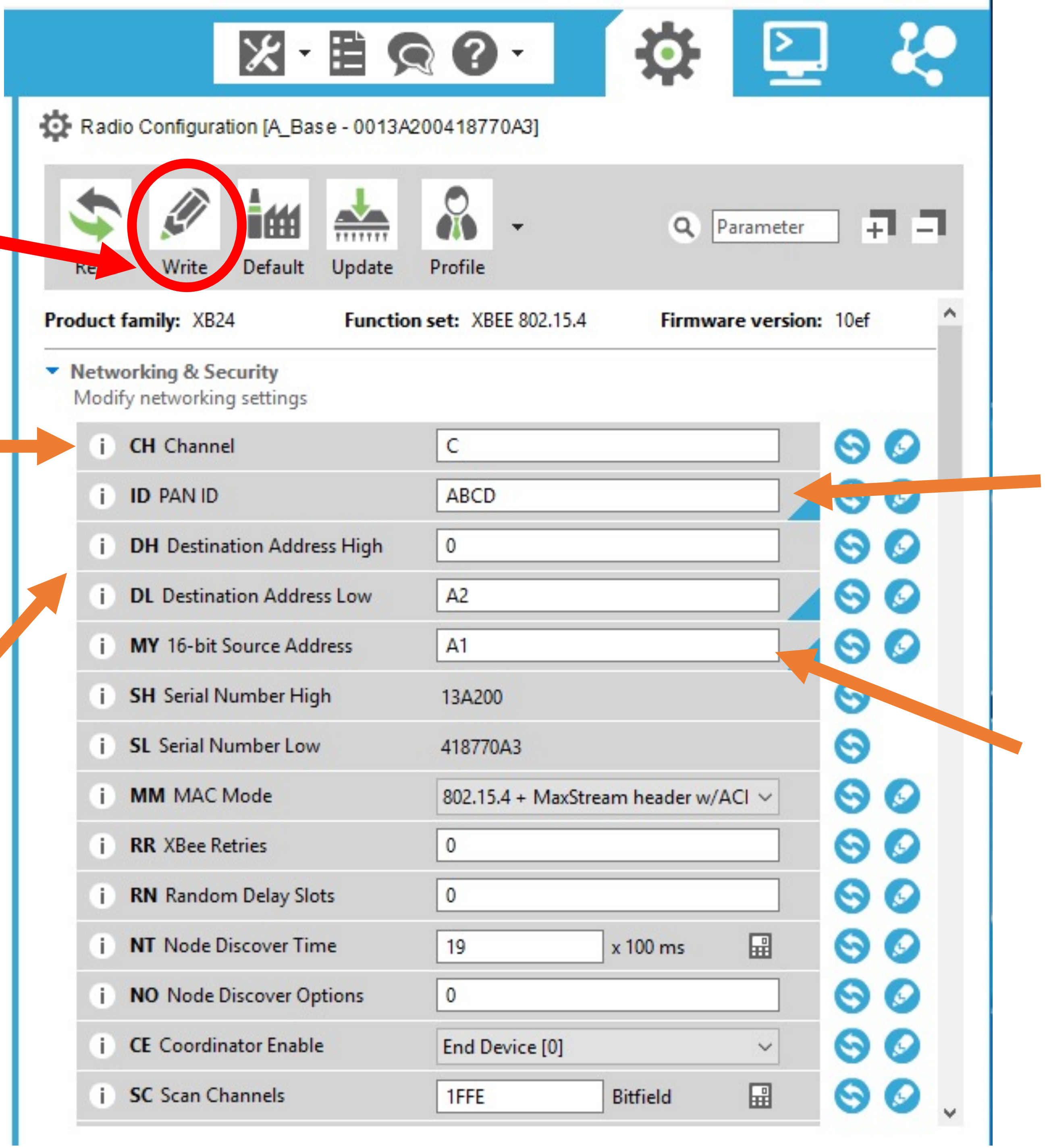


Name: A_Base
Function: XBEE 802.15.4
Port: COM7 - 9600/8/N/1/N - AT
MAC: 0013A200418770A3

Click here to Write new parameters.

RF Channel to use

Upper and Lower address for Destination Node. All your data will go to that node. Should correspond to your Base/Payload connection.



Radio Configuration [A_Base - 0013A200418770A3]

Product family: XB24 **Function set:** XBEE 802.15.4 **Firmware version:** 10ef

Networking & Security
Modify networking settings

CH Channel	C	
ID PAN ID	ABCD	
DH Destination Address High	0	
DL Destination Address Low	A2	
MY 16-bit Source Address	A1	
SH Serial Number High	13A200	
SL Serial Number Low	418770A3	
MM MAC Mode	802.15.4 + MaxStream header w/ACI	
RR XBee Retries	0	
RN Random Delay Slots	0	
NT Node Discover Time	19	x 100 ms
NO Node Discover Options	0	
CE Coordinator Enable	End Device [0]	
SC Scan Channels	1FFE	Bitfield

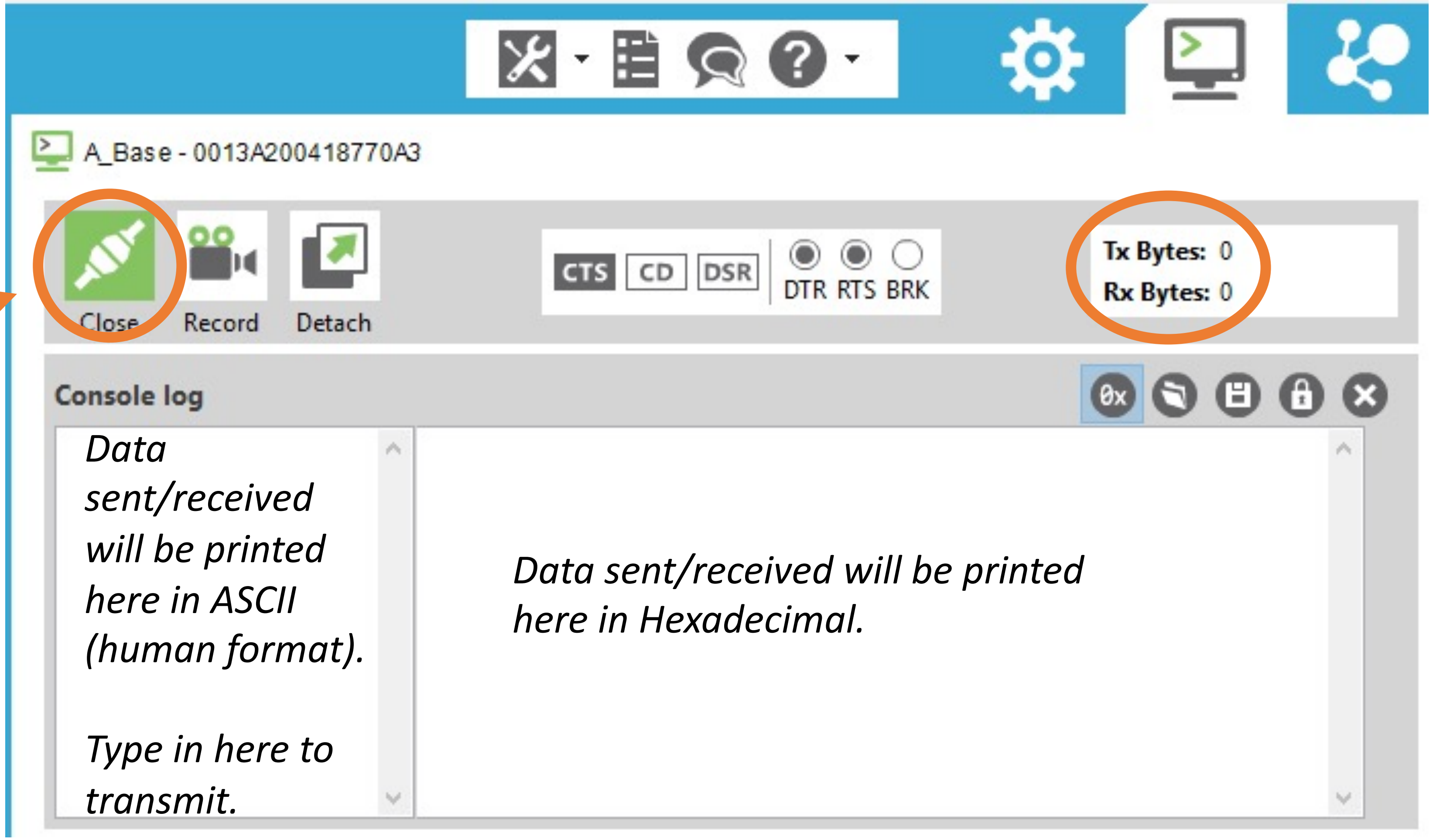
Personal Area Network Name

Your Node’s unique MY address. Other nodes need to know (or lookup) this to send you information.

Xbee Terminal

Open/Close serial communication with the Base station.

Make sure to close when you're done.



Tx Bytes: 0
Rx Bytes: 0

Data sent/received will be printed here in ASCII (human format).

Type in here to transmit.

Data sent/received will be printed here in Hexadecimal.

Bytes sent/received. Useful to see how much data your node requires.

Configuring the Network

- To communicate, all radios must have the same PAN and Channel
- Each device should have its own unique MY address
- To broadcast to all nodes, set Destination Low (DL) to 0xFFFF
- To broadcast to a *specific* node, set DL to its MY address

- Start by seeing if nodes can “broadcast” to one another then configure direct communication

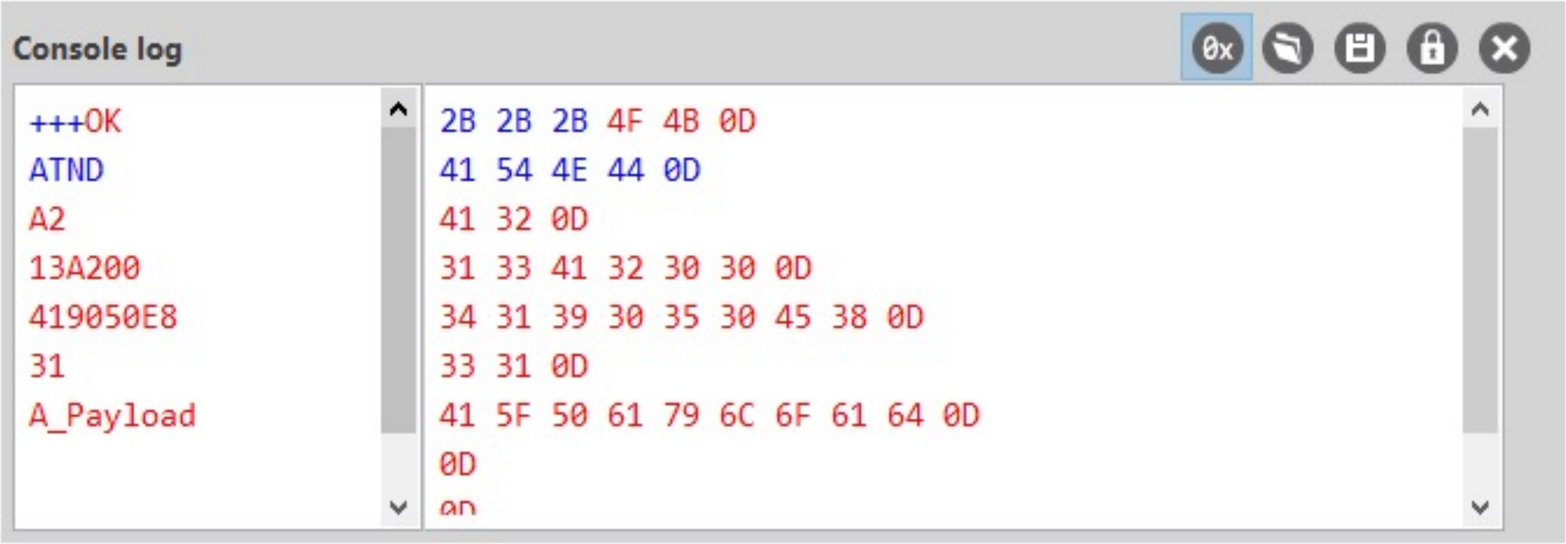
Some Simple Demos

Download them here:

https://github.com/jforsyth/ENGR498-2021/tree/master/assignments/xbee

Network Discovery

- While each Radio only communicates with its paired device, they can still all see each other. Use the XCTU terminal to do a “Network Discovery” and see all nodes in the PAN.
- In the console type “+++” quickly. Then “ATND” and enter.



```
+++OK          2B 2B 2B 4F 4B 0D
ATND           41 54 4E 44 0D
A2             41 32 0D
13A200         31 33 41 32 30 30 0D
419050E8       34 31 39 30 35 30 45 38 0D
31             33 31 0D
A_Payload      41 5F 50 61 79 6C 6F 61 64 0D
               0D
               0D
```

Xbee Beacon

- Download the Xbee beacon program onto your Arduino. It will “ping” the base station each second.

- View the “pings” in the XCTU software. See how far away you can walk.

- Ensure the Arduino Xbee is configured to only talk with your PC radio, otherwise you will spam the network.

Xbee Echo

- Download the Xbee Echo program. Each byte received by the Arduino radio will be “echoed” back.
- You must configure the Arduino and PC radios for direct communication otherwise the “echo” may not go back to the same radio.
- The sent and “echoed” bytes should appear in the XCTU software.

Xbee Serial Monitor

- This implements a “serial monitor” like the XCTU software in the Arduino serial monitor.
- All data received over Xbee is sent back to USB and vice versa.
- Radio behaves as bridge between Xbee and PC connections

Xbee ATND

- An implementation of the ATND (Network Discover) command on the Arduino

- Program asks Xbee to perform Network Discovery and then report back the network nodes.

- Good examples of waiting for feedback. Bonus: why aren’t newlines printed? Would work in OSX but not Windows.

Summary

- Many forms of wireless communication available to extend microprocessor.

- Choices for hardware are driven by range, energy usage, and data rates (probably in that order).

- More modern “microprocessors” available with Wifi and Bluetooth onboard. May be an easier integration with these systems but energy “penalty” will still be present.