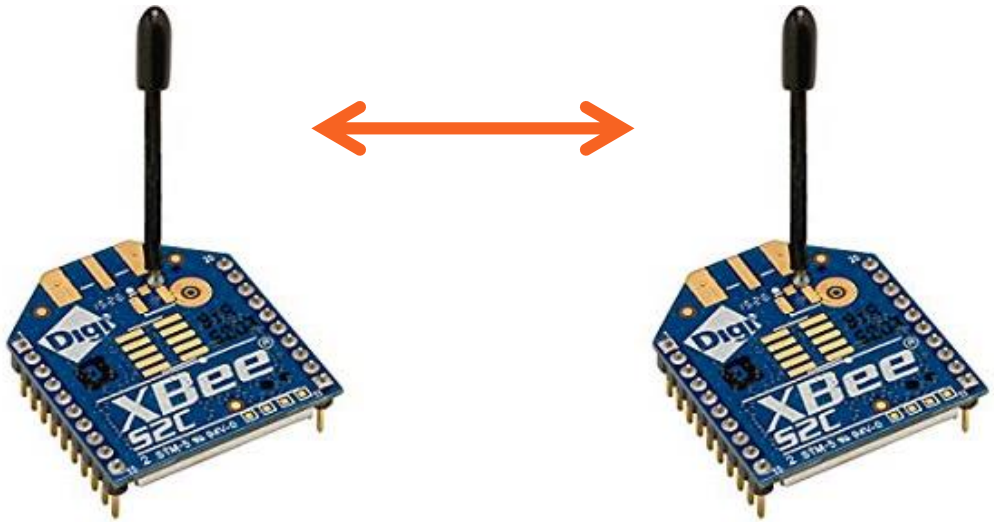
Balloon Satellite – Lesson 5

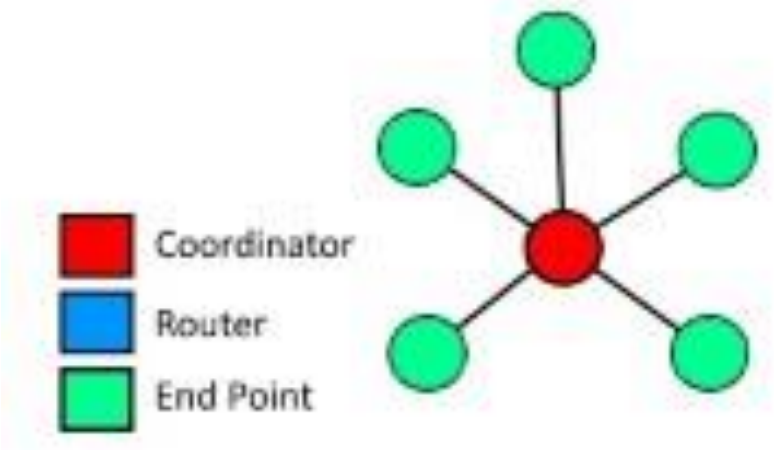
Communications

Wireless Communication – Xbee

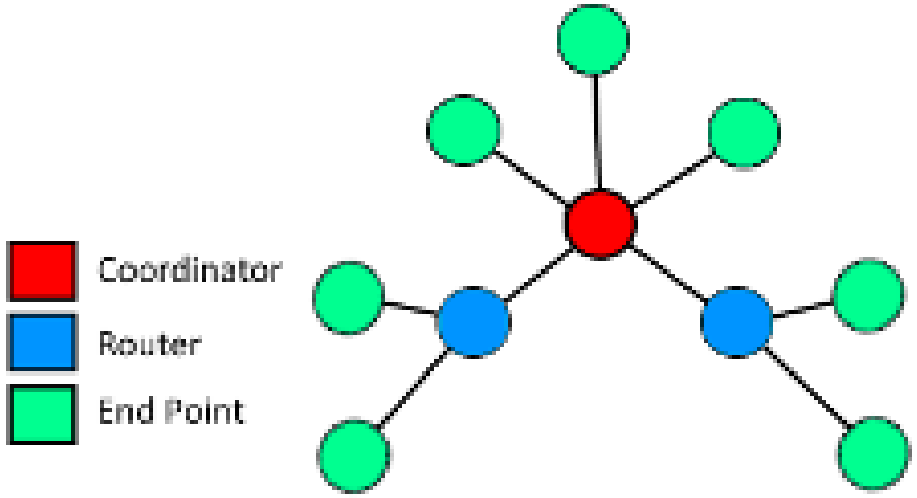
- XBee provides users a way to perform wireless communication
- Operates over UART port (Tx & Rx)
- Xbee modules are 3.3V modules
- They can form multiple network topologies



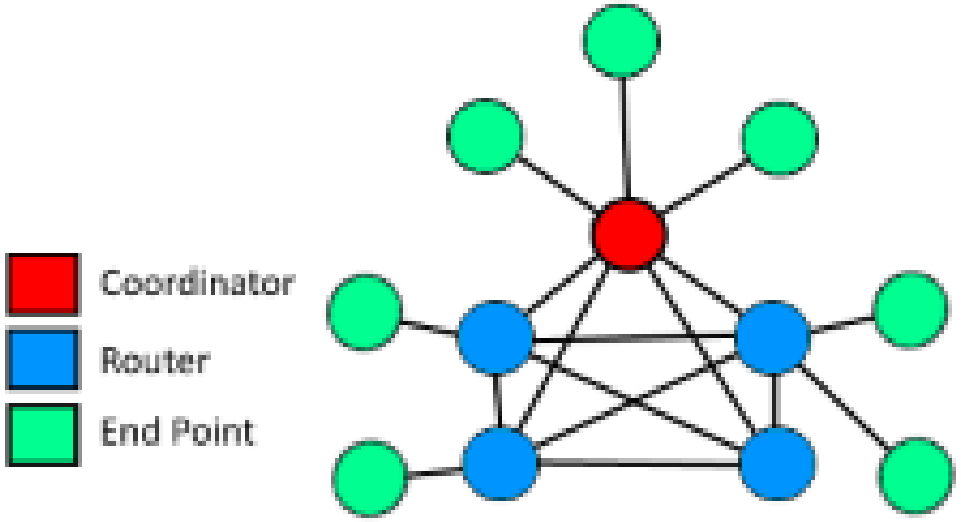
Network Topologies



Star

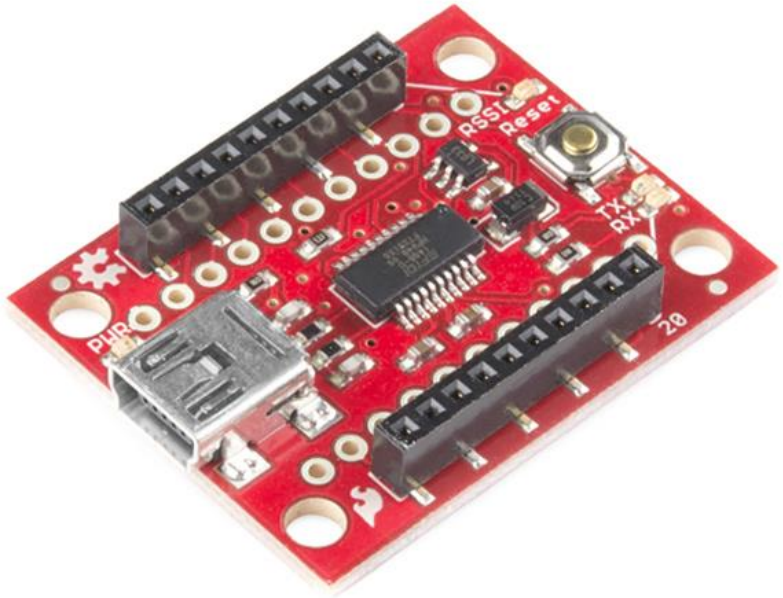


Tree



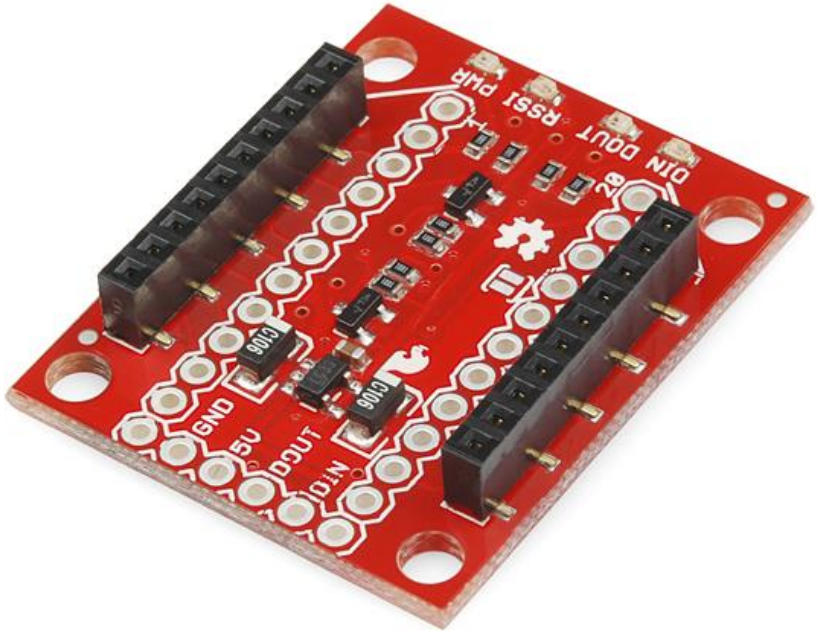
Mesh

Adapters



Xbee Explorer USB

- Connect to PC for configuring of Xbee module
- Onboard regulator and level shifters to adapt to Xbee module



Xbee Explorer Regulated

Connect to microcontrollers for interfacing with Xbee module

- Onboard regulator and level shifters to adapt to Xbee module

Configuration Parameters

Setting	Explanation	Range
Channel (CH)	Uses the channel of wireless network specified (a,b,g,n,etc.) you want this to match for your two modules.	16
PAN ID (ID)	The ID of the network the module will broadcast on. This must match for XBee modules you want to communicate.	0x0 - 0xFFFF
Destination Address High (DH)	This value provides one way to communicate directly to a particular module (hence destination). We typically set this to 0 for point-to-point communication.	0x0 - 0xFFFF
Destination Address Low (DL)	This is the My Address of the board you are trying to communicate with.	0x0 - 0xFFFF
My Address (MY)	This is the address you pick for the board you are currently programming. If multiple boards have the same MY, they will all receive data sent to that address.	0x0 - 0xFFFF

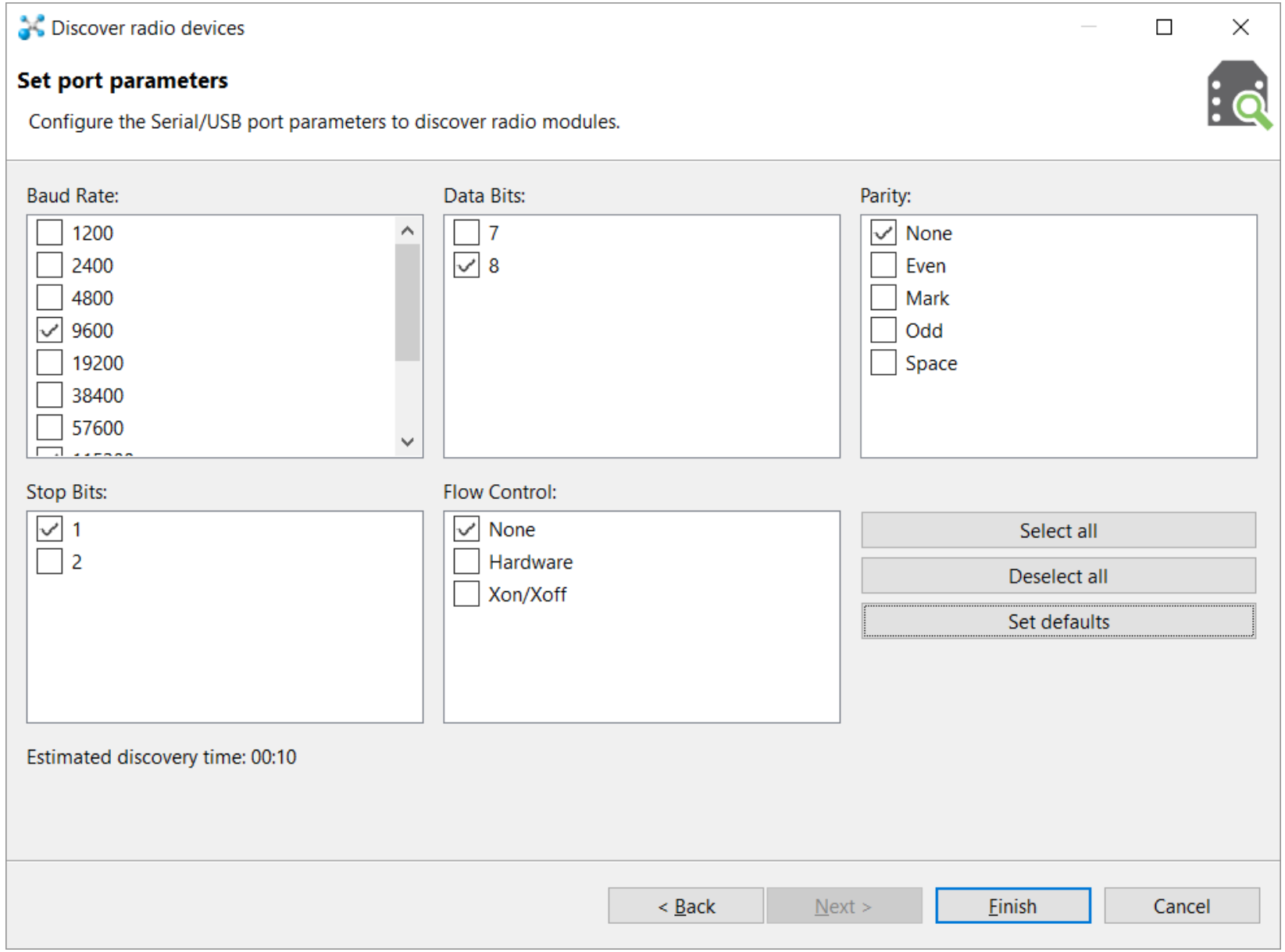
Download XCTU

- https://www.digi.com/products/iot-platform/xctu#productsupport-utilities

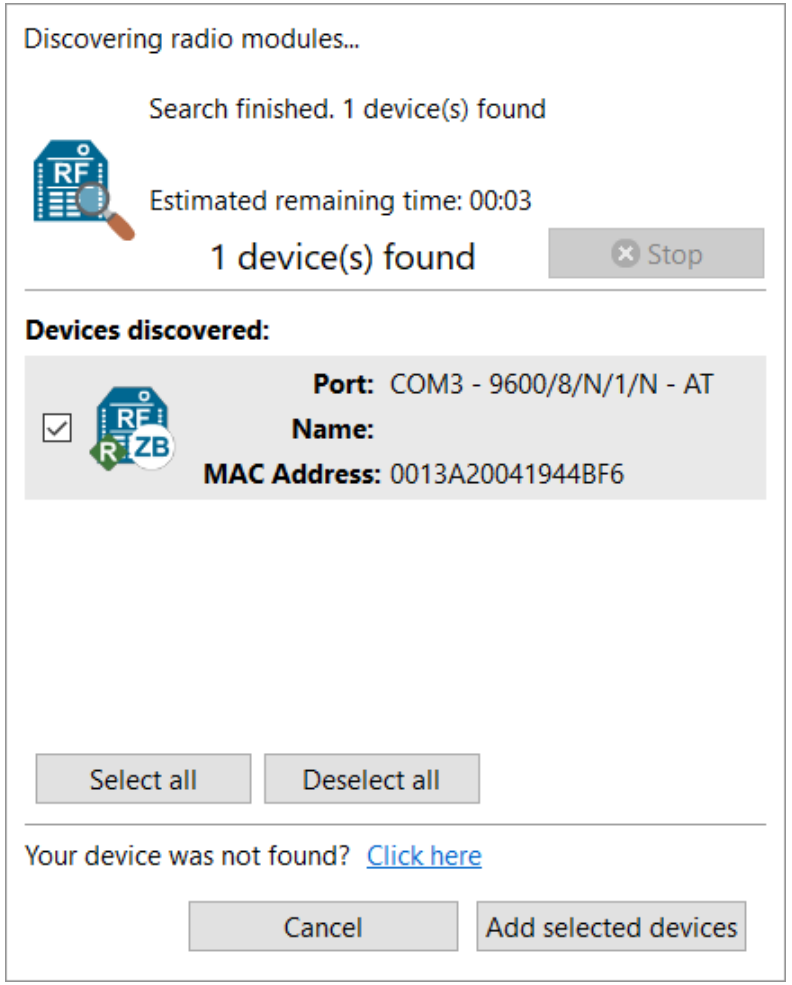
Recommended Parameters in XCTU

Setting	Acronym	XBee Node1	XBee Node 2
Channel	CH	0x0B – 0x1A	
PAN ID	ID	0x0 - 0xFFFF	
Destination Address High	DH	0x0	0x0
Destination Address Low	DL	0x1	0x0
16-bit Source Address	MY	0x0	0x1
Coordinator Enable	CE	1	0

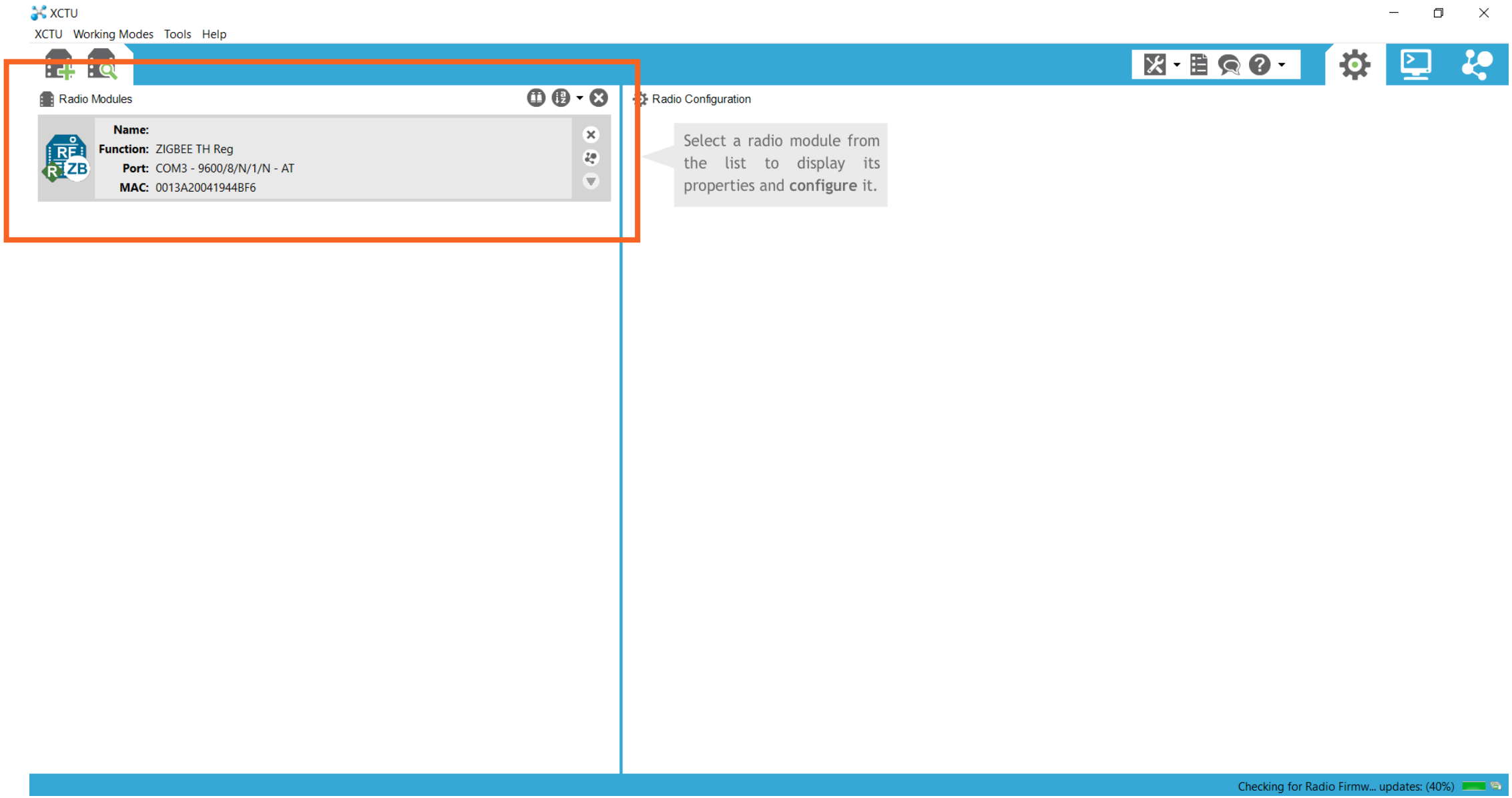
Configure Xbee Modules



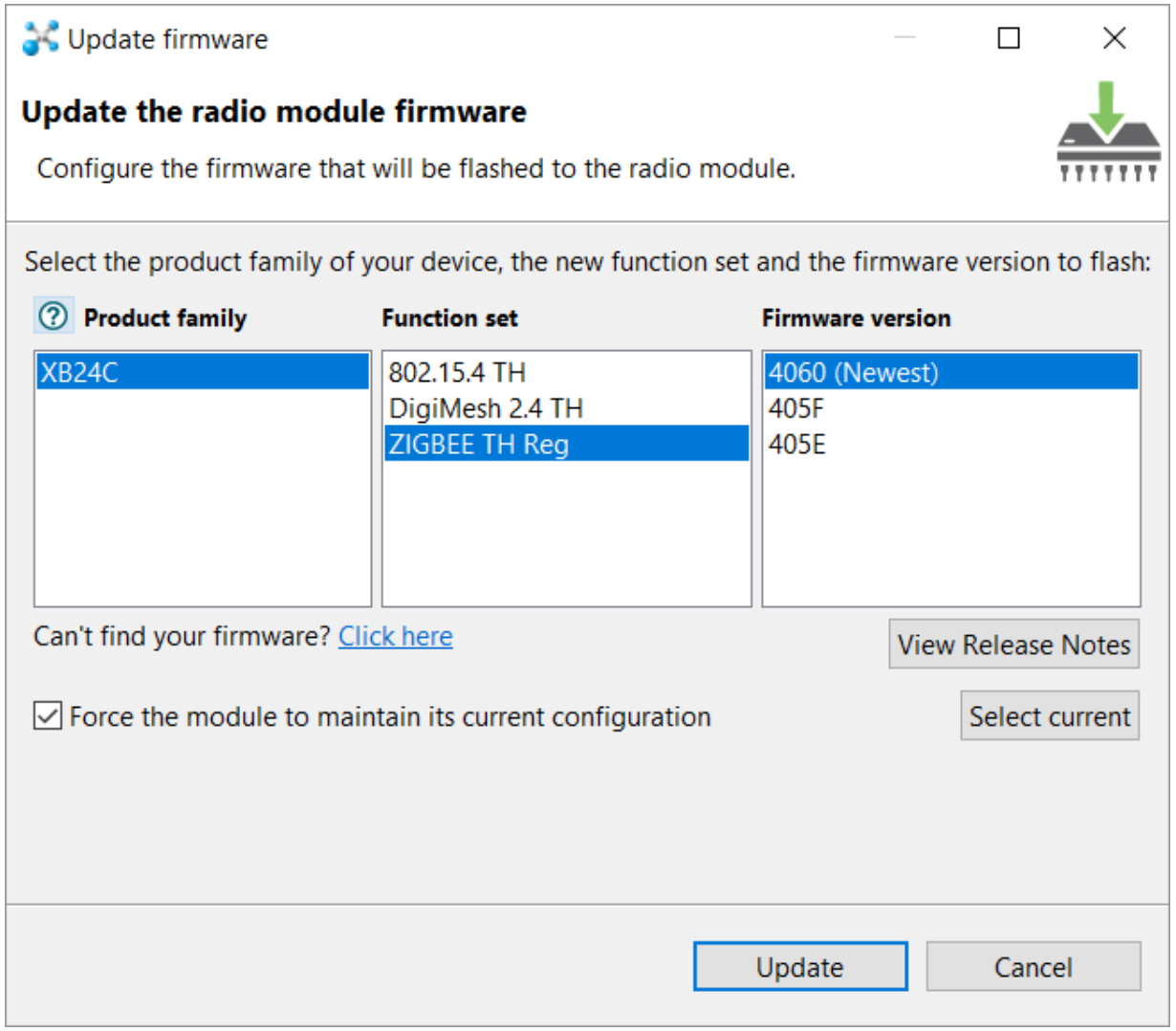
Configure Xbee Modules



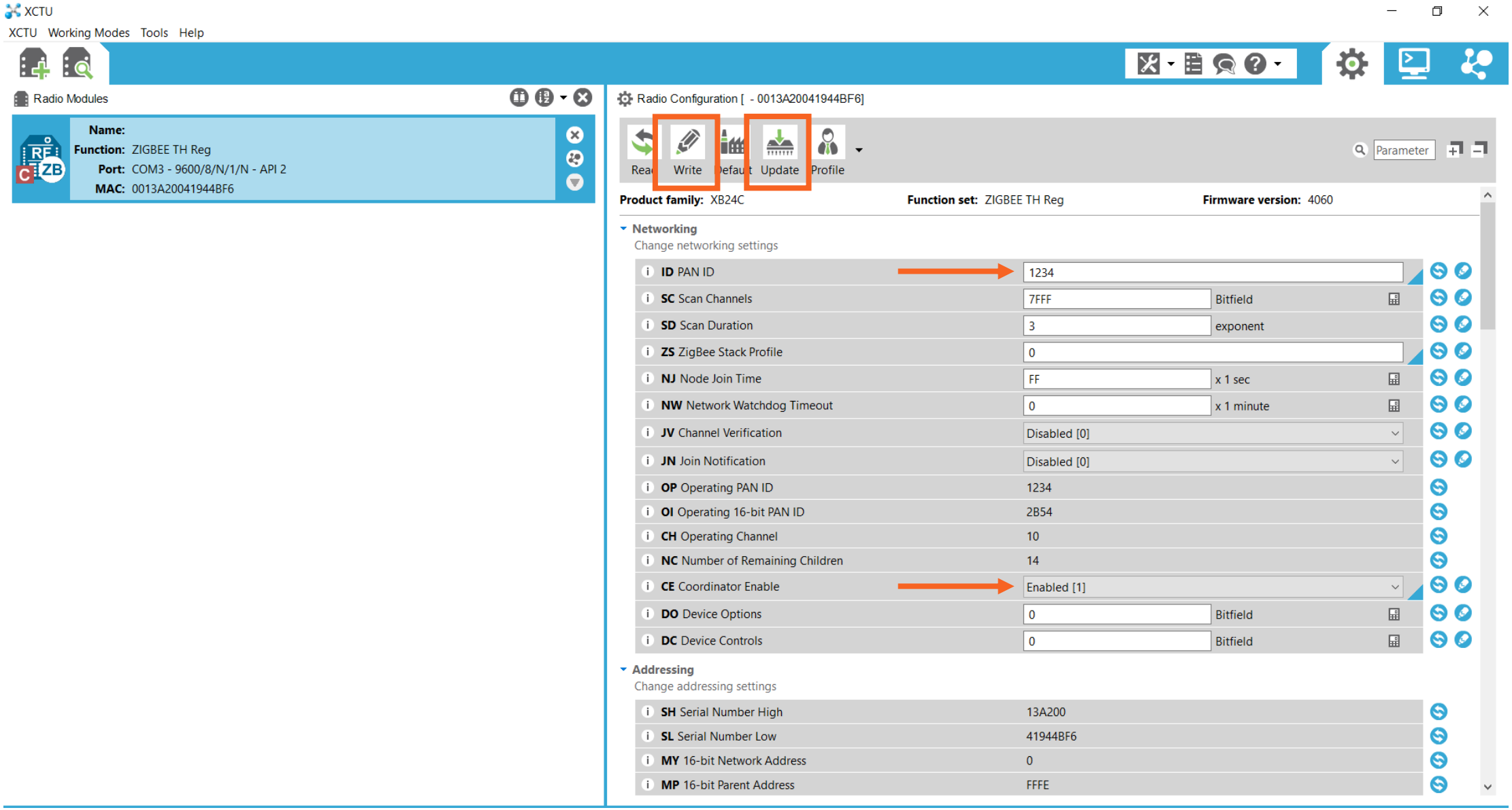
Configure Xbee Modules



Configure Xbee Modules



Configure Xbee Modules



Configure Xbee Modules

XCTU

XCTU Working Modes Tools Help

Radio Modules

Name:
Function: ZIGBEE TH Reg
Port: COM3 - 9600/8/N/1/N - API 2
MAC: 0013A20041944BF6

Radio Configuration [- 0013A20041944BF6]

Read Write Default Update Profile

Parameter

PM Power Mode	Boost Mode Enabled [1]	
PP Power at PL4	9	

Security
Change security parameters

EE Encryption Enable	Disabled [0]	
EO Encryption Options	0	Bitfield
KY Encryption Key		
NK Network Encryption Key		

Serial Interfacing
Change modem interfacing options

BD Baud Rate	9600 [3]	
NB Parity	No Parity [0]	
SB Stop Bits	One stop bit [0]	
RO Packetization Timeout	3	x character times
D6 Pin 16 - DIO6/nRTS Configuration	Disable [0]	
D7 Pin 12 - DIO7/nCTS Configuration	nCTS flow control [1]	
AP API Enable	API enabled with escaping [2]	
AO API Output Mode	Native [0]	

AT Command Options
Change AT command mode behavior

CT AT Command Mode Timeout	64	x 100ms
GT Guard Times	3E8	x 1ms
CC Command Sequence Character	2B	Recommended: 0x20-0x7F (ASCII)

Sleep Modes

Xbee Addresses

- Coordinator

SH	
SL	

- End Device

SH	
SL	

Connection

XBee Explorer Regulated	Arduino Mega
DIN	TX1
DOUT	RX1
5V	5V
GND	GND

Arduino Code

Series2_Tx | Arduino 1.8.5

File Edit Sketch Tools Help

Series2_Tx

```
  You should have received a copy of the GNU General Public License
  along with XBee-Arduino.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <XBee.h>

/*
  This example is for Series 2 XBee
  Sends a ZB TX request with the value of analogRead(pin5) and checks the status response for success
*/

// create the XBee object
XBee xbee = XBee();

uint8_t payload[] = { 0, 0, 0, 0, 0 };

// SH + SL Address of receiving XBee
XBeeAddress64 addr64 = XBeeAddress64(0x0013a200, 0x41944BB4);
ZBTxRequest zbTx = ZBTxRequest(addr64, payload, sizeof(payload));
//ZBTxRequest zbTx = ZBTxRequest(0x1, payload, sizeof(payload));
ZBTxStatusResponse txStatus = ZBTxStatusResponse();

int pin5 = 0;

int statusLed = 13;
int errorLed = 13;

void flashLed(int pin, int times, int wait) {

  for (int i = 0; i < times; i++) {
    digitalWrite(pin, HIGH);
    delay(wait);
    digitalWrite(pin, LOW);

    if (i + 1 < times) {
      delay(wait);
    }
  }
}

void setup() {
```

SH SL

Done uploading.

```
Sketch uses 4120 bytes (1%) of program storage space. Maximum is 253952 bytes.
Global variables use 517 bytes (6%) of dynamic memory, leaving 7675 bytes for local variables. Maximum is 8192 bytes.
```

36 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4

Series2_Rx | Arduino 1.8.5

File Edit Sketch Tools Help

Series2_Rx

```
  if (xbee.getResponse().isAvailable()) {
    // got something

    if (xbee.getResponse().getApiId() == ZB_RX_RESPONSE) {
      // got a zb rx packet

      // now fill our zb rx class
      xbee.getResponse().getZBRxResponse(rx);

      if (rx.getOption() == ZB_PACKET_ACKNOWLEDGED) {
        // the sender got an ACK
        flashLed(statusLed, 10, 10);
      } else {
        // we got it (obviously) but sender didn't get an ACK
        flashLed(errorLed, 2, 20);
      }
      // unpack payload data
      Serial.write(rx.getData(0));
      Serial.write(rx.getData(1));
      Serial.write(rx.getData(2));
      Serial.write(rx.getData(3));
      Serial.write(rx.getData(4));
      Serial.println();

    } else if (xbee.getResponse().getApiId() == MODEM_STATUS_RESPONSE) {
      xbee.getResponse().getModemStatusResponse(msr);
      // the local XBee sends this response on certain events, like association/dissociation

      if (msr.getStatus() == ASSOCIATED) {
        // yay this is great.  flash led
        flashLed(statusLed, 10, 10);
      } else if (msr.getStatus() == DISASSOCIATED) {
        // this is awful.. flash led to show our discontent
        flashLed(errorLed, 10, 10);
      } else {
        // another status
        flashLed(statusLed, 5, 10);
      }
    } else {
      // not something we were expecting
```

11 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4

Arduino Code

Series2_Tx | Arduino 1.8.5

File Edit Sketch Tools Help

Series2_Tx

```
    }
  }
}

void setup() {
  pinMode(statusLed, OUTPUT);
  pinMode(errorLed, OUTPUT);

  Serial.begin(9600);
  Serial1.begin(9600);
  xbee.setSerial(Serial1);

}

void loop() {
  // pack "hello" into the payload
  payload[0] = 72;
  payload[1] = 69;
  payload[2] = 76;
  payload[3] = 76;
  payload[4] = 79;

  xbee.send(zbTx);

  // flash TX indicator
  flashLed(statusLed, 1, 100);

  // after sending a tx request, we expect a status response
  // wait up to half second for the status response
  if (xbee.readPacket(500)) {
    // got a response!

    // should be a znet tx status
    if (xbee.getResponse().getApiId() == ZB_TX_STATUS_RESPONSE) {
      xbee.getResponse().getZBTxStatusResponse(txStatus);

      // get the delivery status, the fifth byte
      if (txStatus.getDeliveryStatus() == SUCCESS) {
        // success.  time to celebrate
        flashLed(statusLed, 5, 50);
      } else {
```

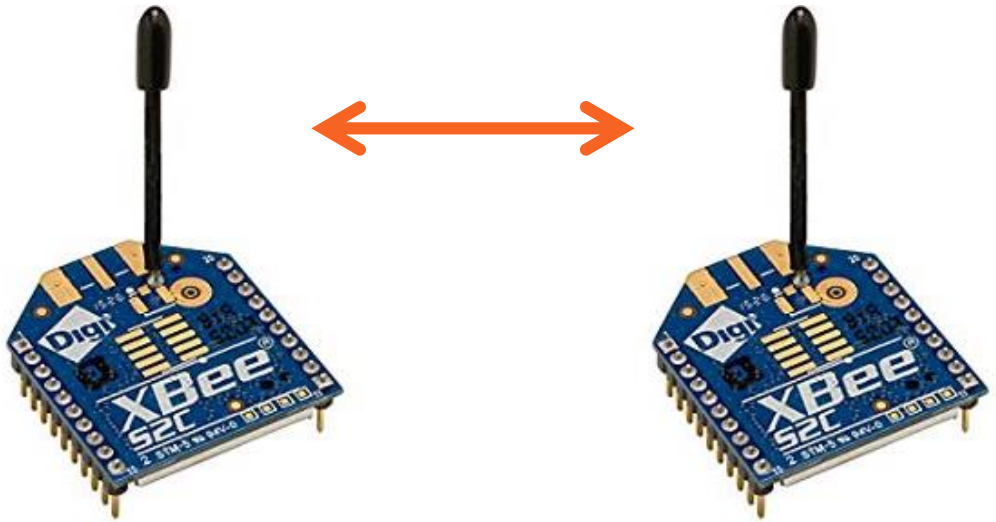
Done uploading.

```
Sketch uses 4120 bytes (1%) of program storage space. Maximum is 253952 bytes.
Global variables use 517 bytes (6%) of dynamic memory, leaving 7675 bytes for local variables. Maximum is 8192 bytes.
```

70 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4

Task 1: Finishing up…

- Change the message that is being transmitted from "**HELLO**" to "**I LOVE TO CODE**"



Task 2: Fusion of Everything

- **Goal:**
  To collect sensor data from humidity sensor and wirelessly transmit the information

- **Task 2.1:**
  Draw up system level block diagram

- **Task 2.2:**
  Setup the system and code (:

Thank You

Contact: zning@nuspace.sg