



NUSPACE

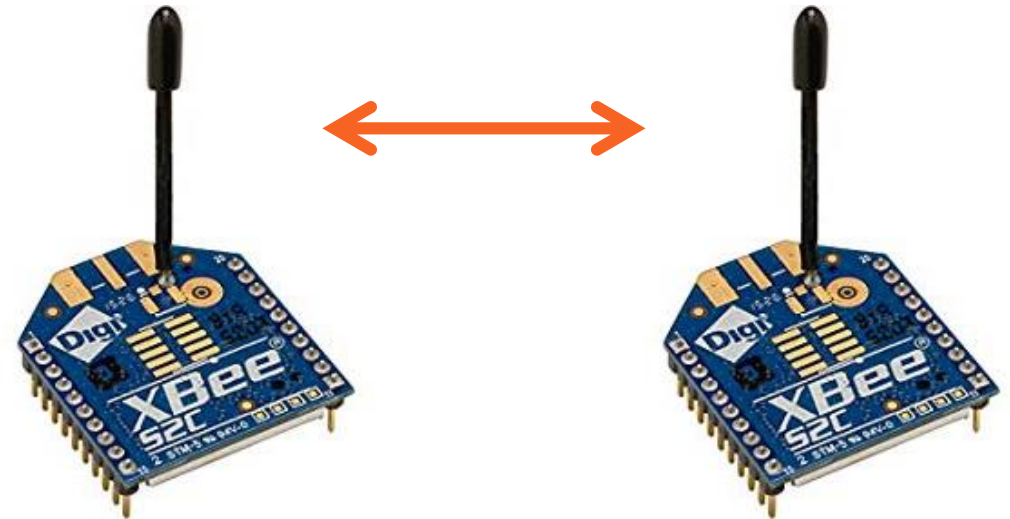
The NUSPACE logo features the word "NUSPACE" in a white, sans-serif font. An orange swoosh underline starts under the "N", curves around the "U" and "S", and ends under the "E". A small satellite icon is positioned above the "S".

Balloon Satellite – Lesson 5
Communications

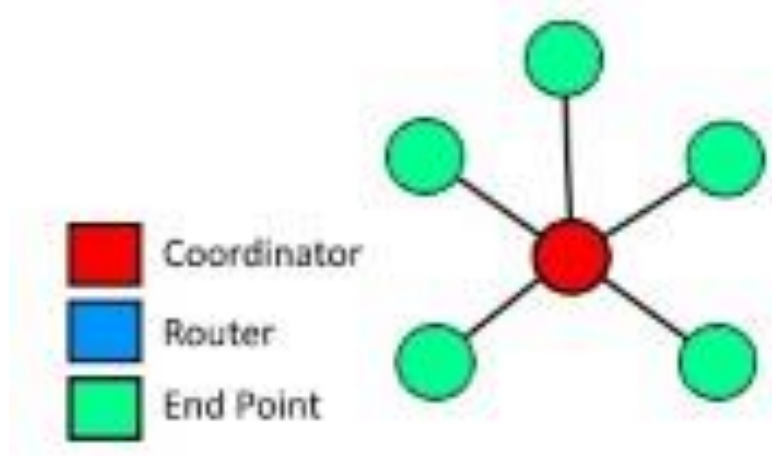


▷ Wireless Communication – Xbee

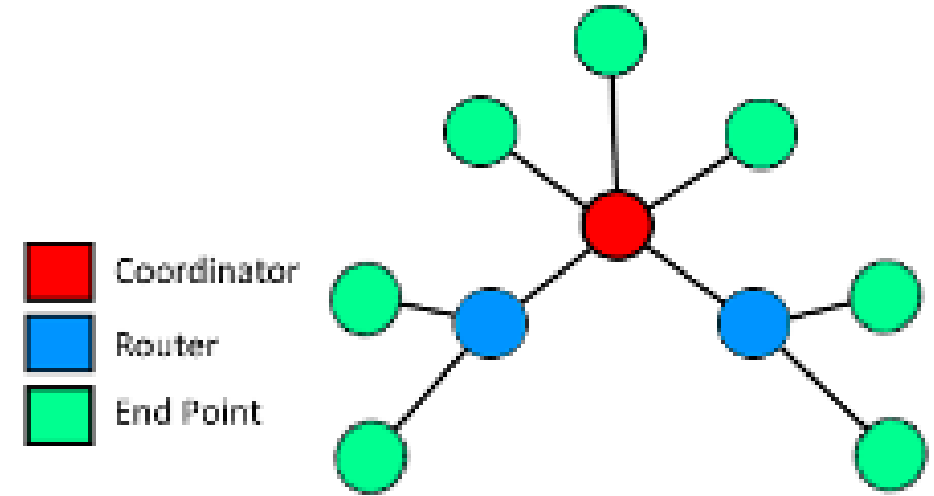
- XBee provides users a way to perform wireless communication
- Operates over UART port (Tx & Rx)
- Xbee modules are 3.3V modules
- They can form multiple network topologies



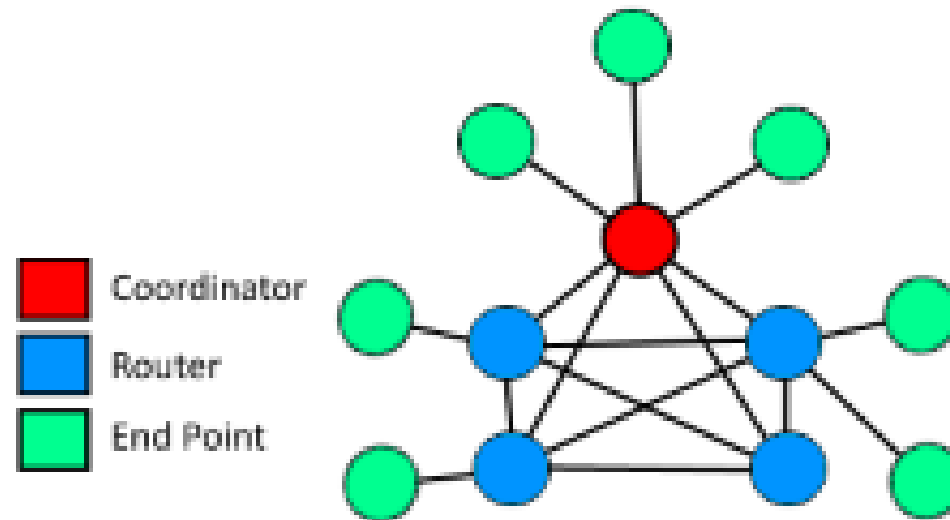
➤ Network Topologies



Star

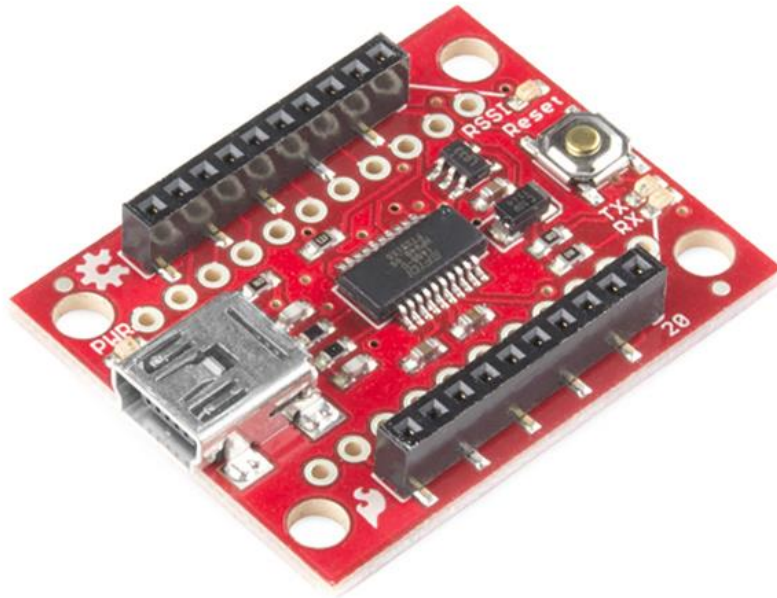


Tree



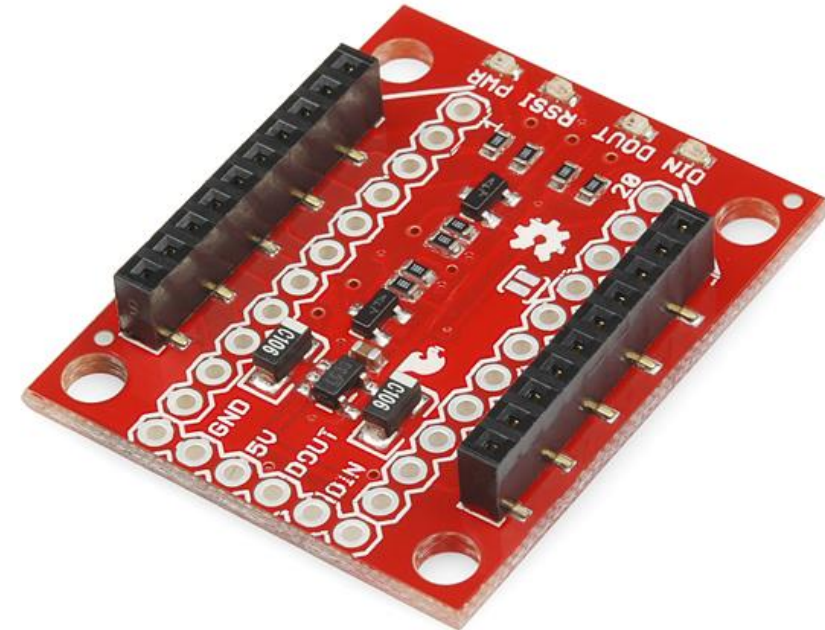
Mesh

▷ 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

 Discover radio devices

Set port parameters
Configure the Serial/USB port parameters to discover radio modules.



Baud Rate:
☐ 1200
☐ 2400
☐ 4800
☒ 9600
☐ 19200
☐ 38400
☐ 57600
☐ 115200

Data Bits:
☐ 7
☒ 8

Parity:
☒ None
☐ Even
☐ Mark
☐ Odd
☐ Space

Stop Bits:
☒ 1
☐ 2

Flow Control:
☒ None
☐ Hardware
☐ Xon/Xoff

Select all
Deselect all
Set defaults

Estimated discovery time: 00:10

< Back

Next >


Finish

Cancel

➤ Configure Xbee Modules

Discovering radio modules...


Search finished. 1 device(s) found

 Estimated remaining time: 00:03

1 device(s) found ⌵ Stop

Devices discovered:

☒



Port: COM3 - 9600/8/N/1/N - AT
Name:
MAC Address: 0013A20041944BF6

Select all Deselect all

Your device was not found? [Click here](#)


Cancel Add selected devices

Configure Xbee Modules

The screenshot displays the XCTU (XBee Configuration Tool) application window. The interface includes a menu bar (XCTU, Working Modes, Tools, Help) and a toolbar with icons for adding modules, searching, and other functions. The main area is divided into two panes: 'Radio Modules' on the left and 'Radio Configuration' on the right. The 'Radio Modules' pane contains a list of modules, with one module selected and highlighted by an orange rectangle. The 'Radio Configuration' pane is currently empty, showing a message that prompts the user to select a radio module from the list to display its properties and configure it. A status bar at the bottom indicates the progress of checking for radio firmware updates (40%).

XCTU
XCTU Working Modes Tools Help

Radio Modules


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

Radio Configuration


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

Checking for Radio Firmw... updates: (40%)


➤ Configure Xbee Modules

 Update firmware

Update the radio module firmware
Configure the firmware that will be flashed to the radio module.



Select the product family of your device, the new function set and the firmware version to flash:

 Product family	Function set	Firmware version
<div><div>XB24C</div></div>	<div><div>802.15.4 TH</div><div>DigiMesh 2.4 TH</div><div>ZIGBEE TH Reg</div></div>	<div><div>4060 (Newest)</div><div>405F</div><div>405E</div></div>

Can't find your firmware? [Click here](#)

☒ Force the module to maintain its current configuration

View Release Notes

Select current

Update

Cancel

Configure Xbee Modules

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]

Real Write Default Update Profile

Parameter

Product family: XB24C **Function set:** ZIGBEE TH Reg **Firmware version:** 4060

Networking
Change networking settings

ID PAN ID	1234	
SC Scan Channels	7FFF	Bitfield
SD Scan Duration	3	exponent
ZS ZigBee Stack Profile	0	
NJ Node Join Time	FF	x 1 sec
NW Network Watchdog Timeout	0	x 1 minute
JV Channel Verification	Disabled [0]	
JN Join Notification	Disabled [0]	
OP Operating PAN ID	1234	
OI Operating 16-bit PAN ID	2B54	
CH Operating Channel	10	
NC Number of Remaining Children	14	
CE Coordinator Enable	Enabled [1]	
DO Device Options	0	Bitfield
DC Device Controls	0	Bitfield

Addressing
Change addressing settings

SH Serial Number High	13A200
SL Serial Number Low	41944BF6
MY 16-bit Network Address	0
MP 16-bit Parent Address	FFFE

Configure Xbee Modules

The screenshot displays the XCTU (XBee Configuration Tool Utility) software interface. The main window is titled "Radio Configuration [- 0013A20041944BF6]". On the left, a sidebar shows the "Radio Modules" list with a selected module having the following details:

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

The main configuration area is divided into several sections:

- Power Mode:** Includes "PM Power Mode" (Boost Mode Enabled [1]) and "PP Power at PL4" (9).
- Security:** Includes "EE Encryption Enable" (Disabled [0]), "EO Encryption Options" (0), "KY Encryption Key" (empty), and "NK Network Encryption Key" (empty).
- Serial Interfacing:** Includes "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]), and "AO API Output Mode" (Native [0]). An orange arrow points to the "AP API Enable" dropdown.
- AT Command Options:** Includes "CT AT Command Mode Timeout" (64 x 100ms), "GT Guard Times" (3E8 x 1ms), and "CC Command Sequence Character" (2B, Recommended: 0x20-0x7F (ASCII)).
- Sleep Modes:** (Section header visible, details not shown).

The interface includes a top menu bar (XCTU, Working Modes, Tools, Help) and a toolbar with icons for Read, Write, Default, Update, and Profile. A search bar labeled "Parameter" is also present.

▢ 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() {
  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.
}
```

```
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
  }
}
```


➤ 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 {

```

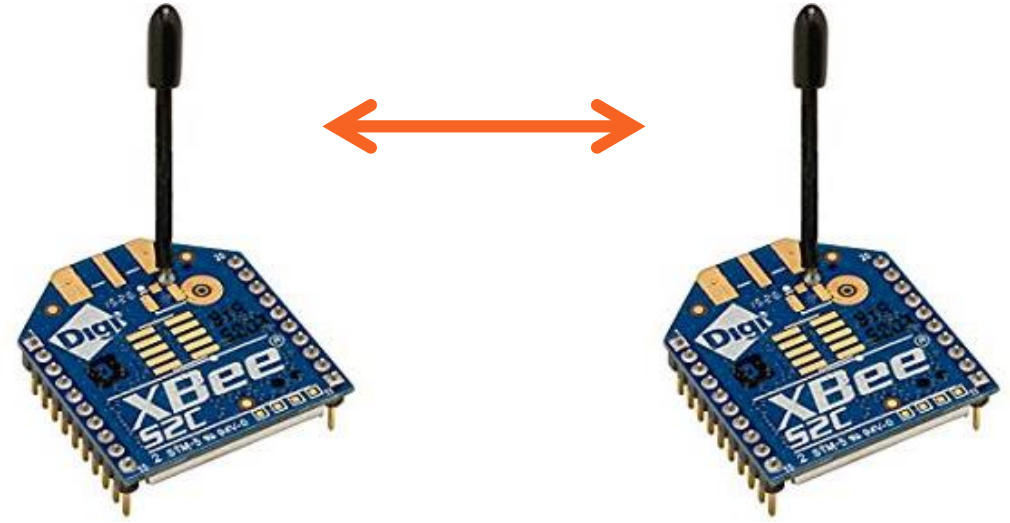
Done uploading.

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

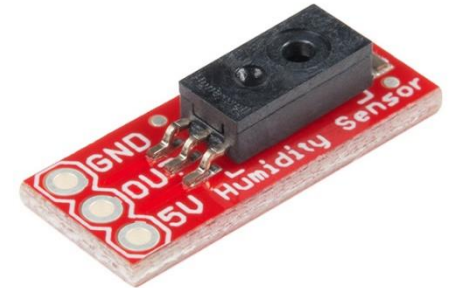
Task 1: Finishing up...

- 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 (:



The NUSPACE logo features the word "NUSPACE" in a bold, white, sans-serif font. A thick orange swoosh curves around the letters "P" and "A". Above the "P" is a small white satellite icon with two rectangular solar panels.

NUSPACE



Thank You

Contact: zning@nuspace.sg