Communication with XBee using UART protocol

Experiment

1 Aim:

Establish Communication between 2 micro-controllers using XBee modules. Print the received data on Arduino Serial Monitor.

2 Problem Statement:

Configure one XBee module to send data, and another to receive data. Send data to sender XBee using serial port of one Arduino, and read data from receiver XBee using serial port of another Arduino. Print the received data on Arduino Serial Monitor so that it can be verified.

3 Procedure:

3.1 Configuring XBee:

Install XCTU by follwing the instructions given in PDF. Open XCTU, and connect your XBee to computer using data cable to configure it.

- Setting up a new device:
  - To add new device, click on **Discover Radio Device** button. Corresponding window will open.
  - In this window, select the COM port where XBee is connected. You can use device manager in Windows to get the COM port number. Let other settings remain default, and click on **Finish**.
  - Updating XBee Firmware: XCTU will detect your XBee module. Click on the icon to show Radio Configuration Properties. Click on Update button. Select **XB24C, ZIGBEE TH Reg, Newest** and click on Update. Firmware update may take a while.

  Device is now ready to be used as a sender or a receiver.
- In every XBee network, there needs to be one and only one *Coordinator*. Rest devices can be configured as *Routers* or *End Devices*. Routers can relay messages from end devices to the coordinator. We are going to use router to send messages to coordinator.

  XBee devices work in 2 modes, Transparent(AT) mode, and API mode. In AT mode, communication between 2 non-Xbee devices happens through XBees. Router transmits whatever data it has received from serial directly, and coordinator writes the received data on the serial port. For more advanced networks, API mode is required. Multiple devices can be connected in single network using API mode. Here, we're using only 2 modules, so we keep both in transparent(AT) mode. Steps to configure XBee:
  - Sender will be configured as **Router**, and the receiver will be configured as **Coordinator**.
  - There are 5 settings that need to be done:

i. **PAN ID** (ID): This number can have any value from 0x0000 to 0xFFFF. PAN ID for the router and the coordinator *needs to be same* for successful communication.

ii. **Node Identifier** (NI): A label can be given to a Xbee module. It can be anything that makes us easier to recognize the device.

iii. **Channel Enable** (JV): This needs to be enabled only for the router, not for coordinator.

iv. **Coordinator Enable** (CE): This needs to be enabled only for the coordinator, and disabled for the router.

v. **API Enable** (AP): Set this as transparent for both router and coordinator.

To apply these changes, click on **write** button. Changes won't take effect till they're written. You can also click write corresponding to each option to apply it.

- We have kept destination address on router as 0, which means it will be sending messages to the coordinator only.
- Go to console view. There will be a button labeled **open**. Click on that button, then the label will change to **close**. This connects the two modules.

3.2 Connecting Arduino with XBee:

- To send data to router, it needs to be connected to Arduino.
- To print data on Arduino serial monitor, receiver XBee, i.e. coordinator, needs to be connected to an Arduino micro-controller.
- Do these connections for both XBee-Arduino pairs:
  i. 5V of Arduino to 5V of XBee
  ii. GND of Arduino to GND of XBee
  iii. Rx of Arduino to Tx of XBee
  iv. Tx of Arduino to Rx of XBee
- Upload the code "sender.ino" to sender Arduino, and "receiver.ino" to receiving Arduino. "sender.ino" sends the given data to serial port, and "receiver.ino" receives data from serial port and prints it on the serial monitor.
  (**Note**: if serial 0 is being used to communicate with XBee, remove the Rx and Tx connections while uploading a code.)

3.3 Testing the setup:

- Connect both Arduinos to your PC using data cables to power them up.
- Open Arduino IDE, select COM port where receiver Arduino is connected. Open the serial monitor.
- The message "Sending Successful!" will be printed on the monitor.
- Message in the sender code can be modified, that same message should get printed on the monitor on receiving side.