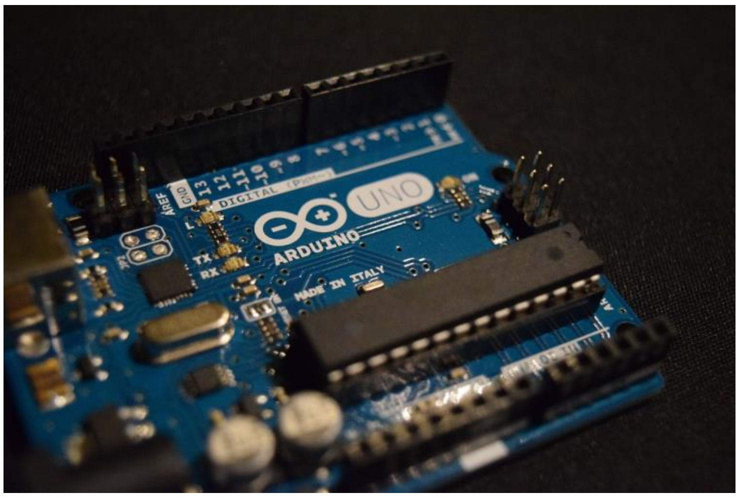


TEMPERATURE SENSING USING ARDUINO AND XBEE

Analog Communication (EC2L009)

Group Project

Group members

Karan Pattanaik
22EC01009

N. Jay Roshan Devankar:
22EC01005

Atharva Tol
22EC01003

Nihar Ranjan Jena
22EC01001

Abstract

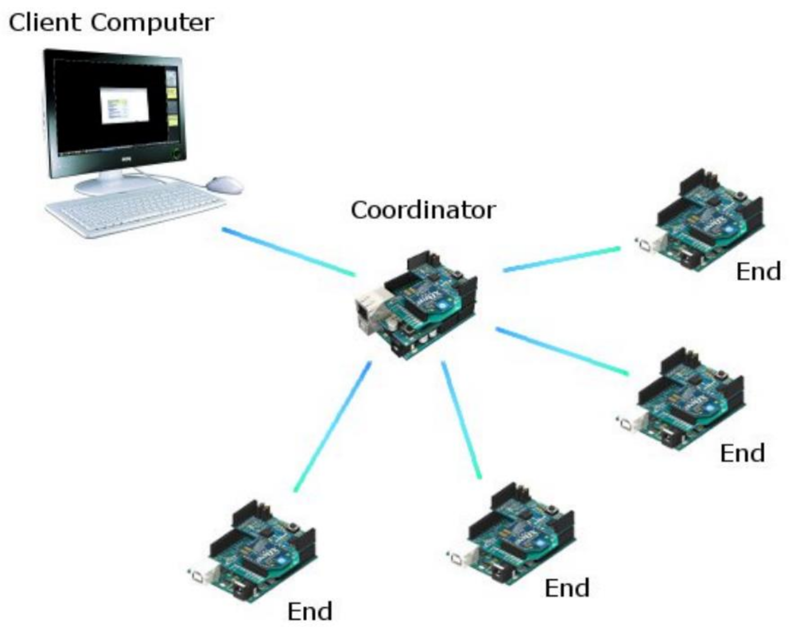
Wireless networks consists of nodes communicate with each other wirelessly.A wireless sensor network (WSN) is a special class of ad-hoc networks that integrates sensing, processing and communications in small, battery-powered motes. These sensor nodes typically collaborate on a global sensing task and deliver required data to one or more hubs. This paper presents experimental setup up of creating wireless sensor network using Arduino and Xbee module. This is to create setup which will allow to read temperature value form inexpensive temperature sensor placed apart at various locations.

2. Introduction

Nowadays, the world is facing many challenges in reducing energy consumption and global warming. In the same time, there are many technologies that can be used to resolve these problems and more over support better living. Wireless Sensor networks (WSNs) are the technology that could provide ubiquitous computing. WSNs deploy many small sensor devices to detect environmental properties .

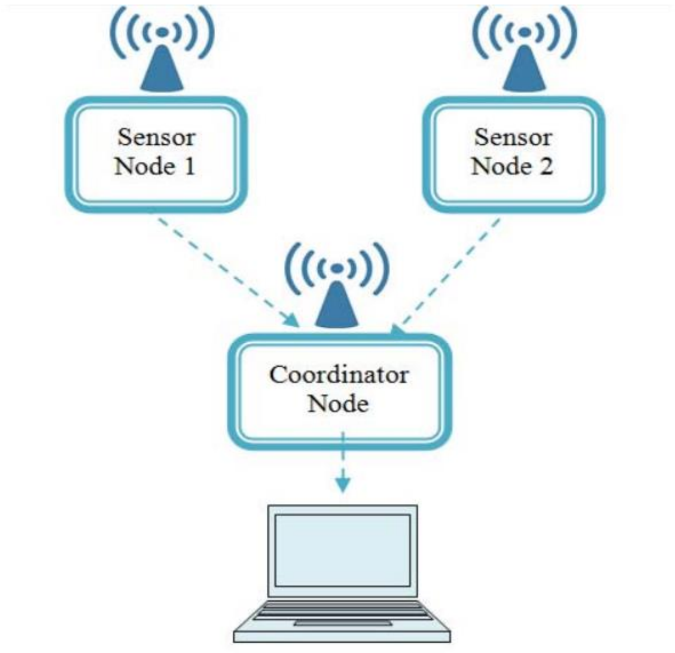
Recent advances in the field of electronics and communication leads to development of tiny battery enabled sensor nodes. These sensor nodes are randomly deployed as a wireless sensor network for sensing environment. Over the years, the most important problem in wireless sensor network is to develop a routing protocol that maximize the life time of network. Each node in a sensor network becomes useless after wasting its energy completely because its power totally depends on the embedded battery.

3. Network Design



System Design (We have used 1 end Device)

For this experimental study, we propose network of which is based on the Arduino development platform with mesh topology. A setup of proposed network is shown in Figure 1. We will be considering this as a static network. Sensor Node 1 and sensor Node 2 will sense the temperature value and send this information to coordinator radio. Coordinator node is very important and powerful node of network like sink node, which gather all data that is transmitted. Routers are intermediate device, which relay/forward packets. Sensor node is simply ending devise which sleep often to save energy.



System Design

4. Hardware specification

The demonstration will require minimum basic components as **Arduino Uno Board**, **Xbee Module**, temperature sensor **LM 35**.

4.1 Arduino Uno Board

Arduino Uno Board

Arduino is an open-source micro-controller system based on simple input output board. Arduino is typically used for creating prototype as well as to develop standalone interactive objects [1] Arduino was developed with requirements as easy to learn and use, flexible, reliable. They are widely used in a wireless sensor network as a portable device [2].

There are number of sensors and actuators that work with Arduino. Popular sensors like temperature, air pollution, light, GPS modules and sound and actuators like LEDs, speakers and digital/analogue outputs are common actuators.

4.2 XBee Radio

Xbee Series 1

Xbee Series 2

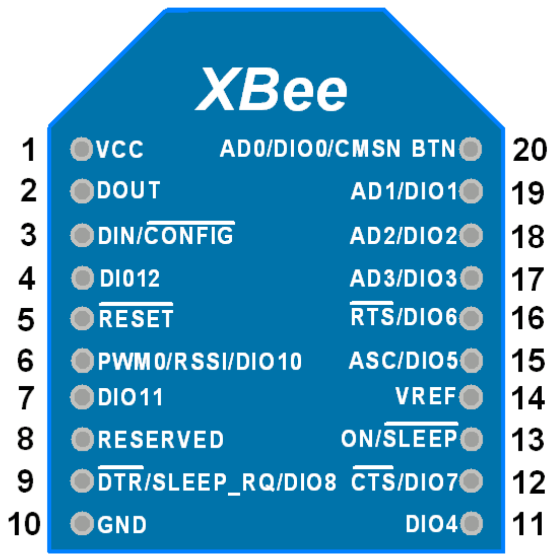
In WSN every node is wirelessly communicate with each other. Xbee is basically used for this purpose. Many people do believe that ZigBee and Xbee are same. But that is not true. XBee is a Zigbee compliant hardware [3]. ZigBee is a standard communications protocol for low-power, low-throughput, low-cost wireless mesh networking applications.

1. XBee Series 1 hardware: These radios use a microchip made by Free scale to provide simple, standards-based point-to-point communications, as well as a proprietary implementation of mesh networking .

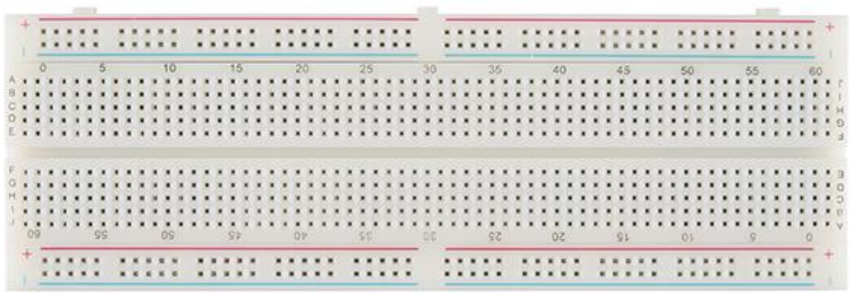
2. XBee Series 2 hardware: The Series 2 uses a microchip from Ember Networks that enables several different flavours of standards-based ZigBee mesh networking.

Both the Series 1 and Series 2 radios are available in two different transmission powers, regular and the regular version is called simply an XBee.

Here we have used an XBee S2C Module which operates over the frequency 2.4 GHz

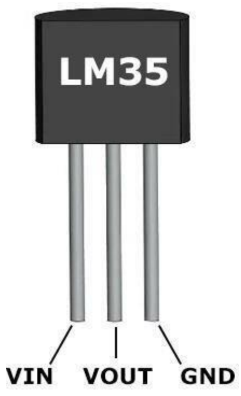


C. Breadboard



Solder less breadboards provide an easy test bed for hooking up electronic circuits without worrying about electrical hazards. They consist of a plastic housing riddled with small holes. Metal clips that lurk beneath the holes in the breadboard provide a way to hold and connect components.

D. LM35 Temperature Sensor



The LM35 Temperature Sensor detects temperature from a range of -55^0 C to 150^{0}C. It gives the temperature output in form of Voltage through the Vout pin. The output varies linearly with temperature and has a factor of 10mV/^{0}C i.e. for every 1^{0}C increase in temperature voltage increases by 10mV

5. Software specification

5.1 Terminal Software to configure Xbees

Each XBee radio has a microcontroller running a configurable program called as firmware. The firmware for the XBee must be downloaded manually [4]. This firmware performs necessary information for addressing, communication, security and utility functions [5].

This firmware is configurable to change different settings like: local address, security setting, destination address and read the analogue sensors connected. The official way to change or upgrade this firmware is through a program called X-CTU and can be downloaded for free from the Digi website [4]. The X-CTU program is the official configuration program for XBee radios which runs only on windows. X-CTU initially used to load the proper firmware onto your XBee radio.

5.2 Arduino IDE

The Arduino is programmed using an open-source application that runs on computer called as IDE and freely downloaded from the Arduino website's software area [6]. The Arduino language is based on C/C++. This IDE simply translate script into C language and compile it using avr-gcc, which makes it understandable to the micro-controller [3].

Configuring Xbee for basic communication

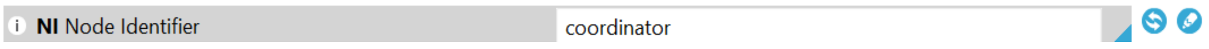
We are using Xbee series 2 radio for this study. Each Xbee has 64-bit address printed on back side of radio. The beginning part of address is assigned by Digi (0013A200) and lower part is actually address of radio.

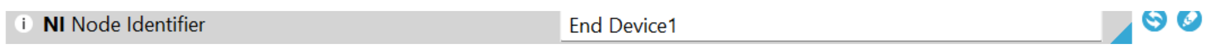
- Plug Xbee series 2 radio on Xbee Explorer board, then plug into one of your USB ports of computer.
- Launch X-CTU, switch to modem configuration tab, it will populate window which will show all kinds of useful information.
- Xbee should be automatically detected and model type should be listed needs to be XB24-ZB.
- Under function set, list of different firmware that can be loaded for this class of radio is shown. You can select ZigBee Coordinator AT, ZigBee Router AT, etc depending on function of modem in network.
- There are other important parameters need to be changed like PANID, destination address high, destination address low.

- Click Write button to program the radio button.
- You can even check radio addresses using AT commands.

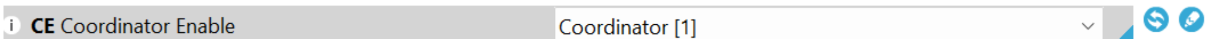
Configuration settings of Xbee

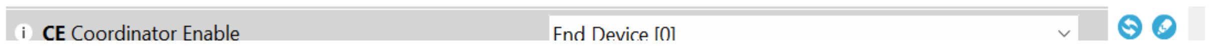
- Naming of both Xbee is done.


NI Node Identifier: coordinator


NI Node Identifier: End Device1

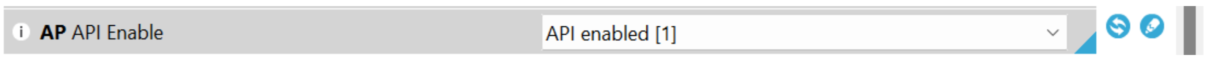
Xbee is set to either coordinator mode or End device mode.

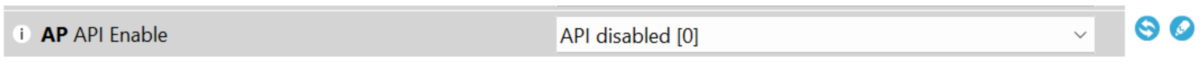

CE Coordinator Enable: Coordinator [1]


CE Coordinator Enable: End Device [0]

- API mode enabled for coordinator and disabled (AT mode) for End device.

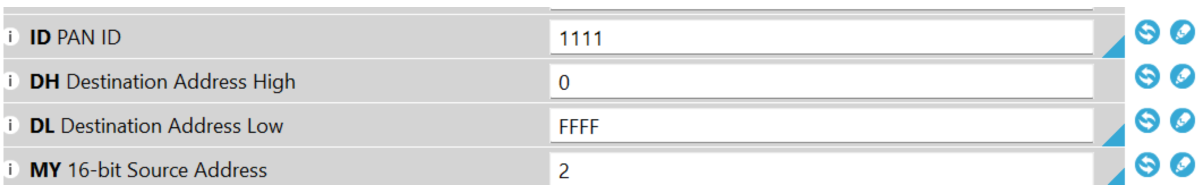
  - <u>Transparent mode (AT command)-</u>In Transparent mode, whatever data is available on DIN pin is directly transmitted to receiver (in case of point to point) or receivers (in case of point to multipoint).
  - <u>Application Programming Interface (API) mode -</u>In API mode, data is wrapped in frame. Frame consists of Start Delimiter, Frame Length, Frame Type, Data, Checksum etc. Parameter setting and packet delivery feedback can be viewed in API mode.


AP API Enable: API enabled [1]


AP API Enable: API disabled [0]

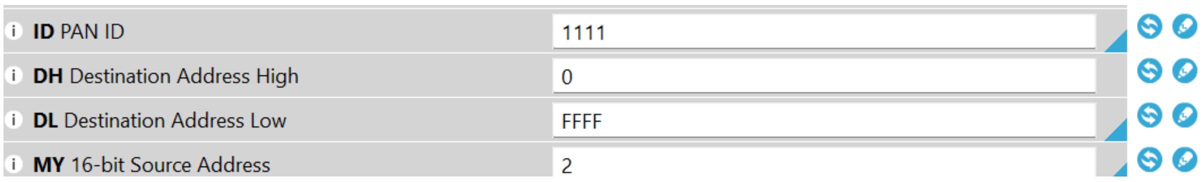
- PAN ID and address of both Xbee are kept same to establish a connection between them.

For coordinator:



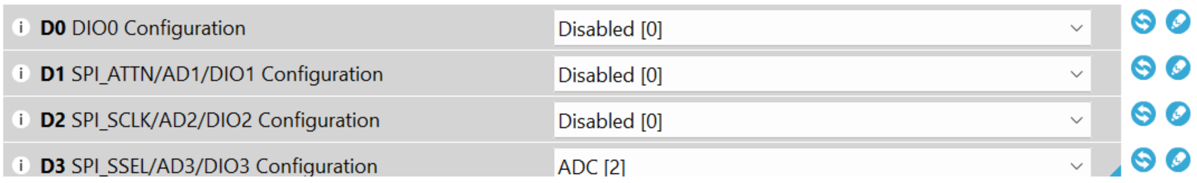
ID PAN ID	1111
DH Destination Address High	0
DL Destination Address Low	FFFF
MY 16-bit Source Address	2

For End device



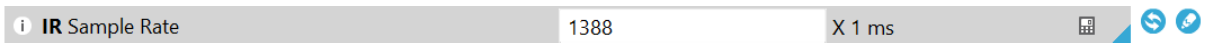
ID PAN ID	1111
DH Destination Address High	0
DL Destination Address Low	FFFF
MY 16-bit Source Address	2

- Pin D3 of End device Xbee is set to ADC(Analog to Digital Converter) to retrieve analog values from LM35



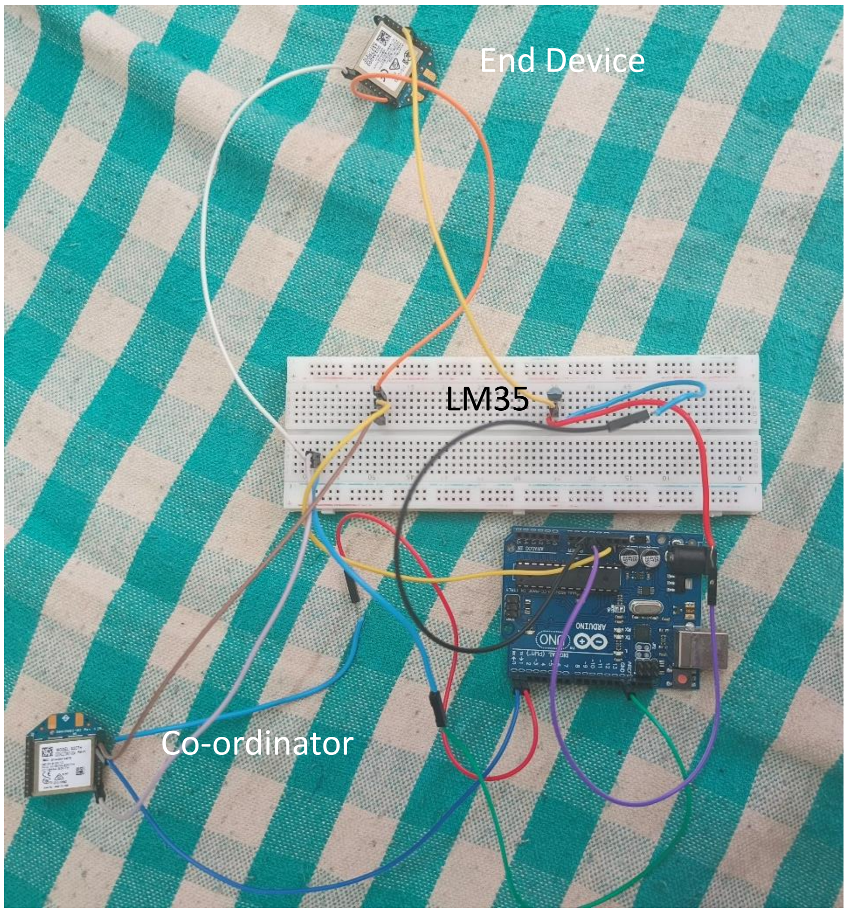
D0 DIO0 Configuration	Disabled [0]
D1 SPI_ATTN/AD1/DIO1 Configuration	Disabled [0]
D2 SPI_SCLK/AD2/DIO2 Configuration	Disabled [0]
D3 SPI_SSEL/AD3/DIO3 Configuration	ADC [2]

- Sampling rate is in hex value (1388 in hex =5000 in decimal)



IR Sample Rate 1388 X 1 ms

6. Circuit



Here we have used Arduino only as a power source for the end device and LM 35 sensor. We have used the 5V pin and ground pin of Arduino for LM35. And the Vout pin is given to the D3 pin of End Device XBee. Also the end device is given 3.3V supply and ground.

This XBee is wirelessly communicating with the Coordinator Xbee and sending the abalog vakue in form of Packets.

The coordinator XBee also has its supply and ground connected with Arduino. It also has 2 more pins i.e. Dout And Din pins connected to Rx(0) and Tx(1) pins of Arduino respectively. This Rx pin receives data from XBee and displays it in the serial monitor of Arduino IDE.

7. Code

```
float temp;
void setup ( ) {
      Serial.begin(9600);
} ;
void loop ( ) {
  byte discardByte;
     if   (Serial.available( ) >=13)  {
          if  (Serial.read ( )  ==  0x7E)   {
             for  (int i = 1; i < 11; i++)  {
                  discardByte = Serial.read( );
             }
             int analogMSB = Serial.read( );
             int analogLSB = Serial.read( );
             int analogReading =analogLSB + (analogMSB * 256);
             temp =analogReading / 1023.0;

             temp = temp / 10;


             Serial.println(String(temp) + " degreesC ");

       }
    }
}
```

8. Conclusion

Wireless sensor network is a communication network wherein sensor nodes deployed at various location senses the data and send to base station. We consider a temperature monitoring application to demonstrate the proof-of-concept of our system. Xbee connected to Arduino board are able to sense data and wirelessly send the data to coordinator radio. Using X-CTU and Arduino IDE we are able to synchronize Xbee and Arduino.

7. References

[1] Stefan Poslad, Ubiquitous Computing: Smart Devices, Environments and Interactions, Wiley, 2009

[2] L. Malathi and R. K. Gnanamurthy, ”Cluster Based Hierarchical Routing Protocol for WSN with Energy Efficiency ”, International Journal of Machine Learning and Computing , Vol.4 No.5, October 2014.

[3] R. Faludi, ”Building Wireless Sensor Networks: with Zig-Bee, XBee, Arduino, and Processing”,O’Reilly Media, Incorporated, 2010

[4] Banzi, M. ,Getting Started with arduino, Make Books,2008.

[5] http://www.digi.com/products/xbee-rf-solutions/xctusoftware.

[6] Arduino homepage,http://www.arduino.cc, October 2012.