# Table of Contents

## Chapter-11

**Interfacing Zigbee Module with Microcontroller**

11.1 Introduction to Zigbee 123

11.2 Zigbee Module Interfacing to the Microcontroller 124

11.3 Circuit Connections 124

11.4 Zigbee Protocol Working Procedure 124
Chapter-11

Interfacing a Zigbee Module with the Microcontroller

11.1 Introduction to Zigbee
In this present communication world, there are numerous high data rate communication standards, but none of these meet the sensors’ and controlling devices’ communication standards. These high-data rate communication standards require low-latency and low-energy consumption even at lower bandwidths. The available proprietary wireless systems’ Zigbee technology's low-cost and low-power consumption and its excellent and superb characteristics makes this communication best suited for several embedded applications, industrial controlling devices, and home automation systems, and so on.

Zigbee Technology

Zigbee communication is specially built for control and sensor networks on IEEE 802.15.4 standard for Wireless Personal Area Networks (WPANs), and it is the product from Zigbee alliance. This communication standard defines physical and Media Access Control (MAC) layers to handle many devices at low-data rates. These Zigbee's WPANs operate at 868 MHz, 902-928MHz and 2.4 GHz frequencies. The data rate of 250 kbps is best suited for periodic as well as intermediate two way transmission of data between sensors and controllers.
11.2 Interfacing Zigbee Module to the Microcontroller
The Zigbee is a transceiver module constructed with transmitter and receiver. The Zigbee module can be directly interfaced to the microcontroller of Rx and Tx pins and doesn't require any external middleware devices. The Zigbee is half-duplex protocol wherein transmitting and receiving the data is not possible at the same time.

In manufacturing and production industries, a communication link continually monitors various parameters and critical equipment. Hence, Zigbee considerably reduces this communication cost as well as optimizes the control process for greater reliability.

11.3 Circuit Connections
The Zigbee module is connected to the microcontroller with Rx and Tx pins as a transmitting and receiving module for the information from the designation.

11.4 Zigbee Protocol’s Working Procedure
Zigbee is particularly designed locally for networks in home environment, and it does not directly communicate with the servers on the internet.

Zigbee devices are needed to send and collect the data back to the managing server on the internet with an additional mechanism.

For example, a gateway can be placed to connect a ZigBee network to the Internet. In a ZigBee network, end devices collect data and send data to the gateway, which then translates the data from the ZigBee protocol format to Internet Protocol format, and vice versa. This allows ZigBee devices to communicate with the servers on the Internet.
ZigBee is a popularly adopted communication technology in smart-grid systems. There are three types of devices in a ZigBee network: coordinator, routers, and end devices. A coordinator is responsible for establishing, maintaining, and controlling a ZigBee network. It allocates network addresses to other nodes that join the network successively. Routers, sometimes called as Relay nodes, take care of data transmission and have the capability to extend the scope of a ZigBee network. End devices collect data and transmit it to routers or coordinators.
Programming:
The Zigbee's transmitting and receiving data as a corresponding program is given here.

1. Transmitting single character
2. Transmitting word
3. Receiving char

```c
#include<reg51.h>
Void init_RS_232();
Void Tx_Char(unsigned char ch);
Void Tx_String(unsigned char *str);
Void Rx_Char();

Void init_RS_232()
{
    TMOD |= 0x20;  //Timer 1 in mode 2 (Auto Reload mode)
    TH1 = 0xFD;    //0xFD for 9600bps
    SCON = 0x50;   //Enable TI and RI pins using Serial control Register
    TR1 = 1;       //Start Timer 1;
}
Void Tx_Char(unsigned char ch)
{
    SBUF = ch;     //Load the character into SBUF register to transmit.
    while (!TI);   //wait for TI flag to raise high
    TI = 0;        //clear TI for further transmission.
}
Void Tx_String(unsigned char *str)
{
    while (*str)
        Tx_Char(str++);
}
Void Rx_Char()
{
    while (RI == 1);  //wait for RI flag to receive any character
    Ch = SBUF;       //capture the character from SBUF into Ch variable
    RI = 0;          //clear RI flag for further reception
}
```