Zig-Bee based Wireless Sensor Network for an Agricultural Environment

Dissertation submitted towards the partial fulfillment of
the requirements for the award of the degree of

Master of Engineering

In

Electronics and Communication Engineering

Submitted by:

Rahul Sharma

Roll No. 821186008

Under the guidance of:

Dr. Surbhi Sharma

Assistant Professor, ECED

Dr. Rajesh Khanna

Professor, ECED

ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT

THAPAR UNIVERSITY

PATIALA-147004, INDIA

CERTIFICATE

I hereby certify that the thesis entitled "**Zig-Bee based Wireless Sensor Network for an Agricultural Environment**" is an authentic record of my study carried out as requirement for the award of degree of Master of Engineering in Electronics and Communication at Thapar University, Patiala under the supervision of Dr. Surbhi Sharma, Assistant Professor and Dr. Rajesh Khanna, Professor, Electronics and Communication Engineering Department, Thapar University, Patiala.

I have not submitted the matter presented in the Thesis for the award of any other degree to any other university.

Date: 17-July-2014

Rahul Sharma
Reg. No. 821186008

It is certified that above statement made by the student is correct to the best of my knowledge and belief.

Dr. Rajesh Khanna
Professor, ECED
Thapar University
Patiala - 147004

Dr. Surbhi Sharma
Assistant Professor, ECED
Thapar University
Patiala - 147004

Dr. Sanjay Sharma
Professor and Head, ECED
Thapar University
Patiala - 147004

Dr. S.K. Mohapatra
Dean Academic Affairs
Thapar University
Patiala - 147004

ii

ACKNOWLEDGEMENT

With great pleasure and sense of obligation I express my heartfelt gratitude to my supervisor **Dr. Surbhi Sharma** , Assistant Professor, Department of Electronics and Communication, Thapar University Patiala. I am proud to say that I have successfully completed my dissertation under her eminent guidance. Inspite of her heavy work commitments and a busy schedule, her persistent encouragement, perpetual motivation, everlasting patience and valuable technical inputs in discussion during the process of dissertation have benefited my work which is beyond expression. He helped me a lot with his expert guidance in designing and development of the dissertation.

I wish to acknowledge my deep sense of gratitude towards **Dr. Rajesh Khanna**, Professor, Department of Electronics, Thapar University, Patiala for motivating me and supporting me. He was always there to guide me through all times in spite of his busy schedule.

I would also like to thank **Mr. Ambrish Kela**, MD, Scientech Technologies Pvt. Ltd. Indore, **Mr. Manish Joshi** Marketing Head, Scientech Technologies Pvt. Ltd. and **Mr. Krenal Chauhan** Team Leader of Embedded Department, Scientech Technologies Pvt. Ltd., for providing the infrastructure and facilities to complete the hardware designing work. Also I would like to thanks my team member **Mr. Rajeev Karothia** from Scientech Technologies Pvt. Ltd. for providing innovative ideas during designing work.

The successful completion of dissertation work is not an individual effort but, it is the outcome of the number of persons, each having their own importance and help for completion of objective. I am thankful to all those peoples who are directly or indirectly contributed in their own special way towards to completion of this dissertation work.

Date : **RAHUL SHARMA**

Roll No. 821186008

ABSTRACT

Wireless Sensor Networks underwent great technological advancements in the recent years, as a result of which, its use in monitoring and control of agricultural parameters became possible. Most of the improvements in agriculture technology and increase in the final yield was seen in the last ten years. As the farmers in most regions still rely on rainwater for crop irrigation, it is very important for them to closely monitor and control the distribution of water to crops in accordance with the uneven rainfall. The main reason for the suitability of wireless technology for irrigation is because of different weather conditions, soil composition and various crop types.

Various aspects of agriculture must be thoroughly analyzed so as to make a correct choice of method. The accuracy of weather prediction is often poor, causing improper irrigation and then further leading to the losses on the part of the farmer. With the fast evolution of wireless sensor technologies and sensor devices, it has now become more feasible and economically viable to implement these for the monitoring and controlling of various parameters of agriculture for Precision Agriculture application.

In this project a fully functional wireless sensor platform with Zig-bee and IEEE 802.15.4 modules have been developed directly without using any Controller. This includes design using Eagle, and implementation of Zig-Bee protocol stack, Application interfaces monitoring and control applications.

Software stack is implemented after the hardware development is complete. The final stage is the development of test application.

Index

Chapters	Page No
List of Figures	IX
List of Tables	X
Chapter 1: Introduction	1
1.1 Introduction	2
1.2 Goal	2
1.3 Design Specification	2
1.4 Methodology	4
Chapter 2: Wireless Communication	5
2.1 Communication Medium	6
2.2 Zig-Bee and IEEE 802.15.4	6
2.3 Technical Specifications	9
2.4 PIN Configuration	10
2.5 Electrical characteristics	10
2.6 RF Module Operations	11
2.7 Serial Interface Protocols	13
2.8 Modes of Operation	14
Chapter 3: System Design	19
3.1 Design Modules	20
3.2 Block Diagram	20
3.3 Block Diagram Description	21
Chapter 4: Hardware Design Details	23
4.1 Power Supply Section – Schematic Layout	24
4.2 Power Supply Section – Schematic Description	24
4.3 Power Supply Section- Bill of Material	24
4.4 Receiver Section – Schematic Layout	25
4.5 Receiver Section – Schematic Description	25

4.6 Receiver Section – Bill of Material 26
4.7 Receiver Section – PCB Layout (Bottom Side) 26
4.8 Receiver Section – PCB Layout (Top Side) 26
4.9 Receiver Section – PCB Layout (Top Component Side) 27
4.10 End Device Node Section – Schematic Layout 27
4.11 End Device Node Section – PCB Layout 28
4.12 End Device (Transmitter) Section – Schematic Description 28
4.13 End Device (Transmitter) Section – Bill of Material 28
4.14 Sensor Section – Schematic Layout 28
4.15 Sensor Section – Schematic Description 29
4.16 Sensor Section – Bill of Material 29
4.17 Sensor Section – PCB Layout 29
Chapter 5: Tools, Firmware and Software 30
5.1 Tools used for designing Hardware and Software 31
5.2 Firmware for Digi- XBee Receiver 33
5.3 Firmware for Digi- XBee End Device Node1 35
5.4 Firmware for Digi- XBee End Device Node2 37
5.5 Application Interface Software Source Code 39
Chapter 6: Result, Analysis and Conclusion 49
6.1 Result 50
6.2 Main Window 51
6.3 Monitor Window 52
6.3 Graph Window 53
6.4 Reading Window 55
6.5 Analysis of Sensor Node1 data 56
6.6 Analysis of Sensor Node2 data 57
6.7 Conclusion 58
6.8 Future Scope 58

LIST OF FIGURES

Figure	Description	Page No
1.	Flow of Design	4
2.	Wireless Module Dimensions	8
3.	Module Air to Air Interface	11
4.	Serial Data Packet	11
5.	Wireless Module Internal Architecture	12
6.	Transparent and API Mode Comparison	14
7.	Modes of Operation	14
8.	Transmit Flow	16
9.	AT Command Format	17
10.	System Design Blocks	20
11.	Block Diagram of Single Sensor Node	20
12.	Block diagram of Complete system	21
13.	Block diagram of Device at Control Room	22
14.	Schematic Layout of Power Supply	24
15.	Schematic Layout of Receiver Section	25
16.	PCB of Receiver Section (Bottom Side)	26
17.	PCB of Receiver Section (Top Side)	26
18.	PCB of Receiver Section Component Side	27
19.	Schematic Layout of Transmitter Section	27
20.	Schematic Layout of Sensor Section	28
21.	Application Software Home Window	51
22.	Application Software Monitoring Window	52
23.	Application Software Graph Window (Day)	53
24.	Application Software Graph Window (Week)	53
25.	Application Software Graph Window (Month)	54
26.	Application Software Graph Window (Year)	54

LIST OF TABLES

Table	Description	Page No
1.	Technical Specifications	9
2.	PIN Configuration	10
3.	Electrical Characteristics	10
4.	Power Supply Section BOM	24
5.	Receiver Section BOM	26
6.	Transmitter Section BOM	28
7.	Sensors BOM	29
8.	Analysis Data of Sensor Node 1	56
9.	Analysis Data of Sensor Node 2	57

CHAPTER 1

INTRODUCTION

CHAPTER 1: INTRODUCTION

1.1 Introduction

In Precision Agriculture (PA) there are various techniques which allow us to monitor the environmental parameters and then control them as per the need of a particular crop. Proper environment is the primary requirement for a good yield and therefore it is important to analyze the various methods for this purpose. It is now becoming a commonplace to use wireless sensor network for precision agriculture. Precision agriculture helps to maintain favorable environment for the desired crop.[1] As a result of this, farmers can produce different crops in different climate of any season.

1.2 Goal

The goal of this project is to develop fully functional wireless sensor platform with Zig-Bee modules without using any Microcontroller to support the wireless application requirement in the industrial scenario. The power supply is applied directly by using the external power source and will provide the continual operation of the sensor node. The design aim is to develop a prototype which provides the analysis and comparison of the sensor node. The design also describes the implementation of temperature and humidity sensors for monitoring applications. Since a sensor is a device that detects and responds to some type of input from the physical environment and convert it into analog voltage form, so for our system we are collecting these values to monitor and control Agricultural parameters [2].

1.3 Design Specifications

The complete system design specifications are as under [3]

1. Communicate with Monitoring Station
2. Support Multiple Interconnectivity
3. Easy addition of new Sensor Node
4. Self Configurable

Temperature & Humidity Sensor Specification [5]

1. Operating Voltage 5V Supply
2. Supply Current 20 mA
3. Relative Humidity from 1%-100%
4. Temperature from +2 deg C to +60 deg C

Wireless Communication Specification [4]

1. Indoor/Urban Range	:	up to 100 ft. (30m)
2. Outdoor RF line-of-sight range	:	up to 300 ft. (100m)
3. Transmit Power Output	:	1 mW (0dbm)
4. RF Data Rate	:	250 Kbps
5. Supply Voltage	:	2.8 - 3.4 V
6. Transmit Current (typical)	:	45 mA (@ 3.3 V)
7. Idle/Receive Current (typical)	:	50 mA (@ 3.3 V)
8. Frequency	:	ISM 2.4 GHz
9. Dimensions	:	0.0960" x 1.087"
10. Operating Temperature	:	-40 to 85 C

Relay Control Circuit Specification

1. Operating Voltage 5V DC
2. Current 20 mA
3. Voltage handle capacity 250VAC / 7A

1.4 Methodology

The modular design methodology is used in designing the project. The project design approach is bottom up i.e. individual sections are tested separately and then finally added to work as a complete system. This report presents a complete design and technology details used in **"Zig-Bee based Wireless Sensor Network for an Agricultural Environment".** The methodology used in designing the project is described in following flow diagram [6].

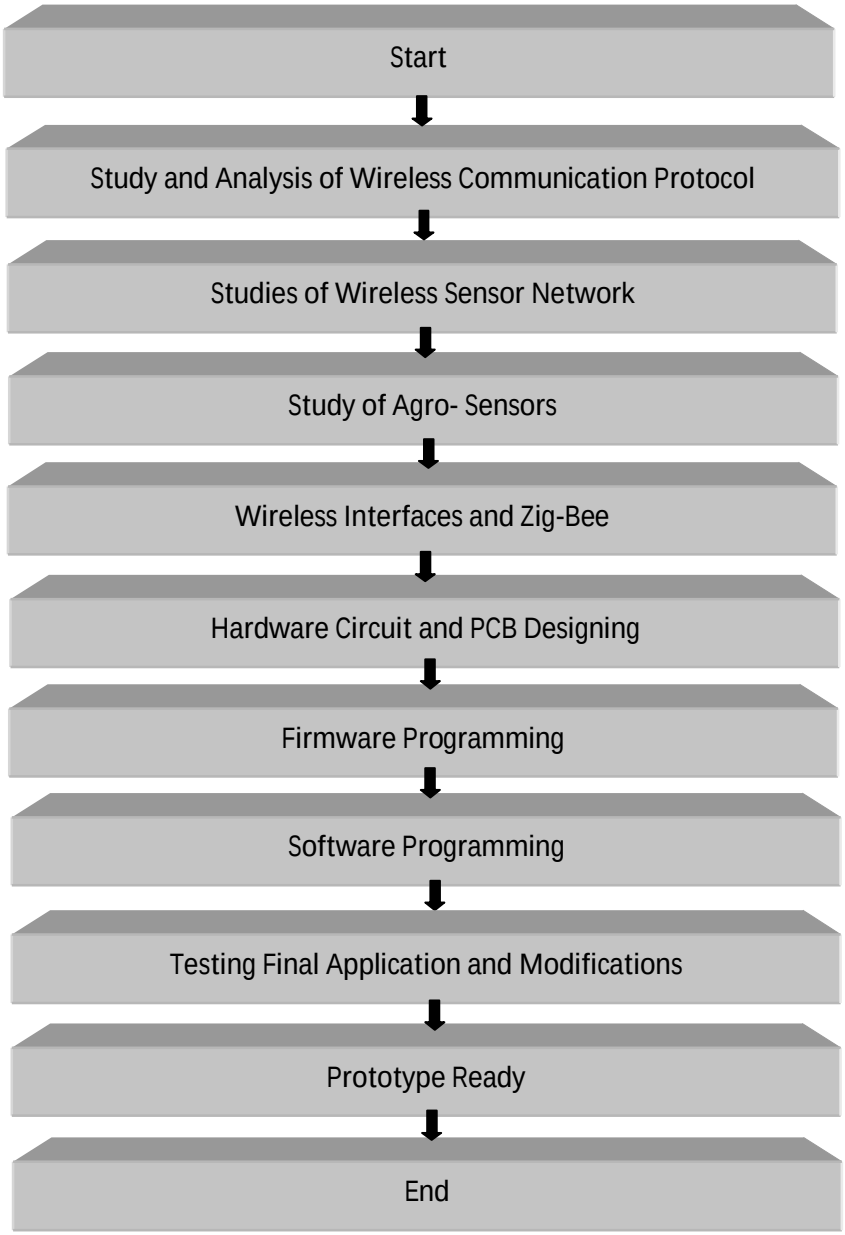


Figure 1: Flow of Design

CHAPTER 2

WIRELESS COMMUNICATION

CHAPTER 2 WIRELESS COMMUNICATION

2.1 Communication Medium

There are various communication mediums and ways are available by which the data from a sensor node can be communicated to a remote PC where the data can be recorded for analysis [7].

The communication medium can be one of the following

1. Wired Medium:
    a. Through dedicated line communication via internet
2. Wireless Medium
    a. Through Blue-tooth technology
    b. Through Zig-Bee technology
    c. Through GSM Technology

Each of the above has a different advantages and limitations. Communication through Internet require Internet connection in every sensor node hence increases installation cost and also running maintenance cost of meter itself. The wireless system has its added advantage of reduced installation cost and no usage charges as in case of Internet. The Blue-tooth device has less range as compared to the Zig-Bee modems and GSM Modems. The best optimized solution is to use Zig-Bee modems for sensor nodes. This prototype design the whole communication medium is through Zig-Bee modems i.e. XB24 from Digi Internati2onal [8].

2.2 Zig-Bee and IEEE 802.15.4

Zig-Bee technology is economical in terms of money as well as power consumption. It is a wireless networking remote control application protocol targeted towards monitoring, IEEE 802.15.4 started that working on a low data rate standard and later on the Zig-Bee and the IEEE recommeded to join forces and Zig-Bee is the name for this technology [9].

Zig-Bee is required to provide cost an power effectiveness. Its battery life may range from several months to several years whereas high data transfer rates may not be a requirement. Zig-Bee can be implemented in mesh networks. Zig-Bee compliant wireless devices can transmit 10- 100 meters, depending on the RF environment and the power output consumption required for a given application, and will operate in the unlicensed RF worldwide (2.4 GHz global, 40kbps at 915 MHz Americas or at 868MHz, Europe). The data rate is 250kbps at 2.4GHz, 40kbps at 915MHz and 20kbps at 868MHz, IEEE and Zig-Bee Alliance is on a continuous process for building the entire protocol stack. IEEE 802.15.4 lays main stress on the bottom two layers of protocol i.e. physical and data link layer. On the other contrary, Zig-Bee Alliance is focused on providing the upper layers of the protocols stack i.e. from network to the application layer for inter operable data networking. In addition it also provides security services and a range of wireless home and building control solutions which enables interoperability compliance testing, marketing of the standard, advanced engineering for the evolution of the standard. This will result in the compatibility of the products so that users may buy products from different manufacturers without the need to worry about their inter-operability. Zig-Bee network layer provides communication redundancy. This unique feature helps in eliminating “single point of failure” in mesh network [10].

XBee RF Communication Module used in prototype

XBee are designed to meet Zig-Bee/IEEE 802.15.4 standards and support the unique needs of low power consumption, low cost wireless sensor networks. The apparatus require minimum power for delivered of critical data between devices [12].

Features:

High Performances, Low Cost

Indoor/Urban: up to 30 meter

Outdoor line of sight: up to 100 meter

Transmit Power: 1mW (0 dBm)

Receiver Sensitivity: -92 dBm

RF Data Rate: 250,000 bps

Advanced Networking & Security [11]

Retries and Acknowledgements

DSSS (Direct Sequence Spread Spectrum)

Each direct sequence channels has over 65,000 unique network address available

Point-to-point, Point-to-Multipoint and Peer-to-Peer topologies supported

128 bit Encryption (downloadable firmware version coming soon)

Self-routing/Self-healing mesh networking

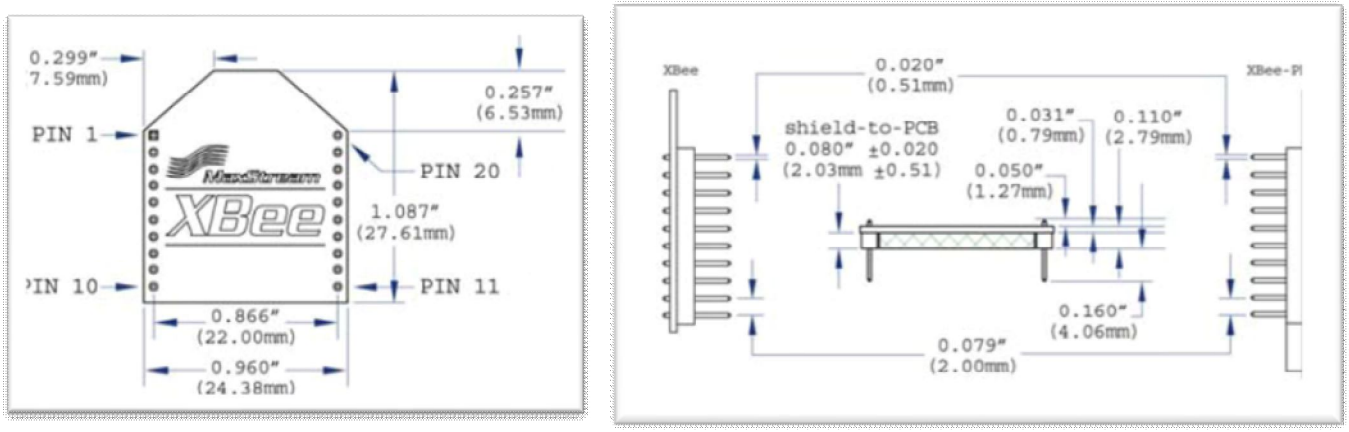
Low Power

TX Current: 45mA @ 3.3V

RX Current: 50mA @ 3.3V

Power-down Current: <10uA

Dimensions [13]



Top View Side View

Figure 2: Wireless Module Dimensions

2.3 Technical Specification

Specification	XBee	XBee-PRO
Performance		
Indoor/Urban Range	Up to 100 ft (30 m)	Up to 300 ft. (90 m), up to 200 ft (60 m) International variant
Outdoor RF line-of-sight Range	Up to 300 ft (90 m)	Up to 1 mile (1600 m), up to 2500 ft (750 m) international variant
Transmit Power Output (software selectable)	1mW (0 dBm)	63mW (18dBm)* 10mW (10 dBm) for International variant
RF Data Rate	250,000 bps	250,000 bps
Serial Interface Data Rate (software selectable)	1200 bps - 250 kbps (non-standard baud rates also supported)	1200 bps - 250 kbps (non-standard baud rates also supported)
Receiver Sensitivity	-92 dBm (1% packet error rate)	-100 dBm (1% packet error rate)
Power Requirements		
Supply Voltage	2.8 – 3.4 V	2.8 – 3.4 V
Transmit Current (typical)	45mA (@ 3.3 V)	250mA (@3.3 V) (150mA for international variant) RPSMA module only: 340mA (@3.3 V) (180mA for international variant)
Idle / Receive Current (typical)	50mA (@ 3.3 V)	55mA (@ 3.3 V)
Power-down Current	< 10 µA	< 10 µA
General		
Operating Frequency	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960" x 1.087" (2.438cm x 2.761cm)	0.960" x 1.297" (2.438cm x 3.294cm)
Operating Temperature	-40 to 85° C (industrial)	-40 to 85° C (industrial)
Antenna Options	Integrated Whip Antenna, Embedded PCB Antenna, U.FL Connector, RPSMA connector	Integrated Whip Antenna, Embedded PCB Antenna, U.FL Connector, RPSMA connector
Networking & Security		
Supported Network Topologies	Point-to-point, Point-to-multipoint & Peer-to-peer	
Number of Channels (software selectable)	16 Direct Sequence Channels	12 Direct Sequence Channels
Addressing Options	PAN ID, Channel and Addresses	PAN ID, Channel and Addresses
Agency Approvals		
United States (FCC Part 15.247)	OUR-XBEE	OUR-XBEEPRO
Industry Canada (IC)	4214A XBEE	4214A XBEEPRO
Europe (CE)	ETSI	ETSI (Max. 10 dBm transmit power output)*
Japan	R201WW07215214	R201WW08215111 (Max. 10 dBm transmit power output)* Wire, chip, RPMSA, and U.FL versions are certified for Japan. PCB antenna version is not.
Australia	C-Tick	C-Tick

Table 1: Technical Specifications [14]

2.4 PIN Configuration

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / $\overline{\text{CONFIG}}$	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	$\overline{\text{RESET}}$	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	$\overline{\text{CTS}}$ / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / $\overline{\text{SLEEP}}$	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	$\overline{\text{RTS}}$ / DIO6	Either	Request-to-Send Flow Control, or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

Table 2: PIN Configurations [15]

2.5 Electric characteristics

Symbol	Characteristic	Condition	Min	Typical	Max	Unit
V_{IL}	Input Low Voltage	All Digital Inputs	-	-	0.35 * VCC	V
V_{IH}	Input High Voltage	All Digital Inputs	0.7 * VCC	-	-	V
V_{OL}	Output Low Voltage	I_{OL} = 2 mA, VCC >= 2.7 V	-	-	0.5	V
V_{OH}	Output High Voltage	I_{OH} = -2 mA, VCC >= 2.7 V	VCC - 0.5	-	-	V
II_{IN}	Input Leakage Current	V_{IN} = VCC or GND, all inputs, per pin	-	0.025	1	µA
II_{OZ}	High Impedance Leakage Current	V_{IN} = VCC or GND, all I/O High-Z, per pin	-	0.025	1	µA
TX	Transmit Current	VCC = 3.3 V	-	45 (XBee) / 215, 140 (PRO, Int)	-	mA
RX	Receive Current	VCC = 3.3 V	-	50 (XBee) / 55 (PRO)	-	mA
PWR-DWN	Power-down Current	SM parameter = 1	-	< 10	-	µA

Table 3: Electrical Characteristics [18]

2.6 RF Module Operations

Serial Communications

The RF Modules connects to a host device through a logic level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART or through a level translator to any serial device (For example: RS-232/485/422 or USB interface board) [16]

UART Data Flow

Devices that have a UART interface can connect directly to the pins of the RF module as shown in the figure below [19].

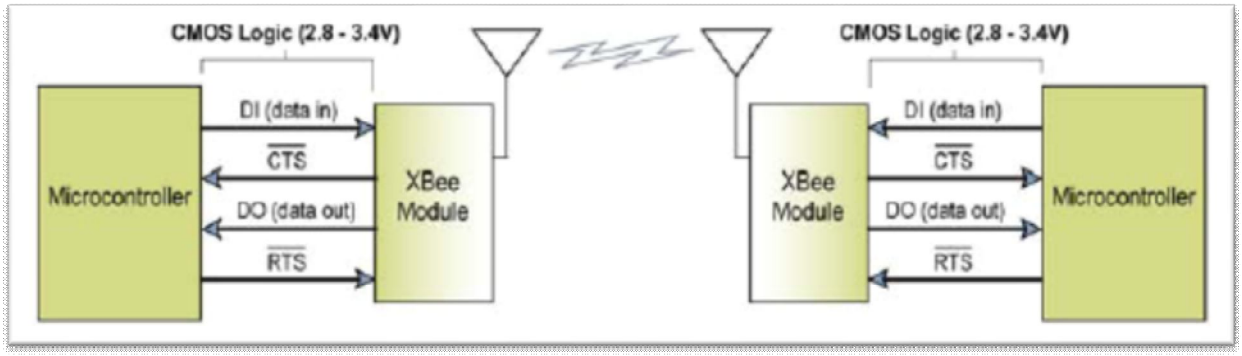


Figure 3: Module Air to Air Interface

Serial Data

Data in the form of asynchronous serial signal enters the module UART through the pin 3 which is the DI pin. When no data is being transmitted, idle signal is transmitted. Every data byte consists of two parts. First one is the start bit which is the low signal and a stop bit which is high signal. The serial bit pattern of data passing through the module is shown in the following figure. The figure below shows the UART data packet 0x1F (decimal 31) as transmitted through RF module [17].

Figure 4: Serial Data Packet

The RF module UART performs operations like timing and parity checking. These tasks are the ones that are needed for data communications. Serial communications relies on the two UARTs to be configured with compatible settings. These settings include baud rate, parity, start bits, stop bits, data bits.

Internal Flow diagram

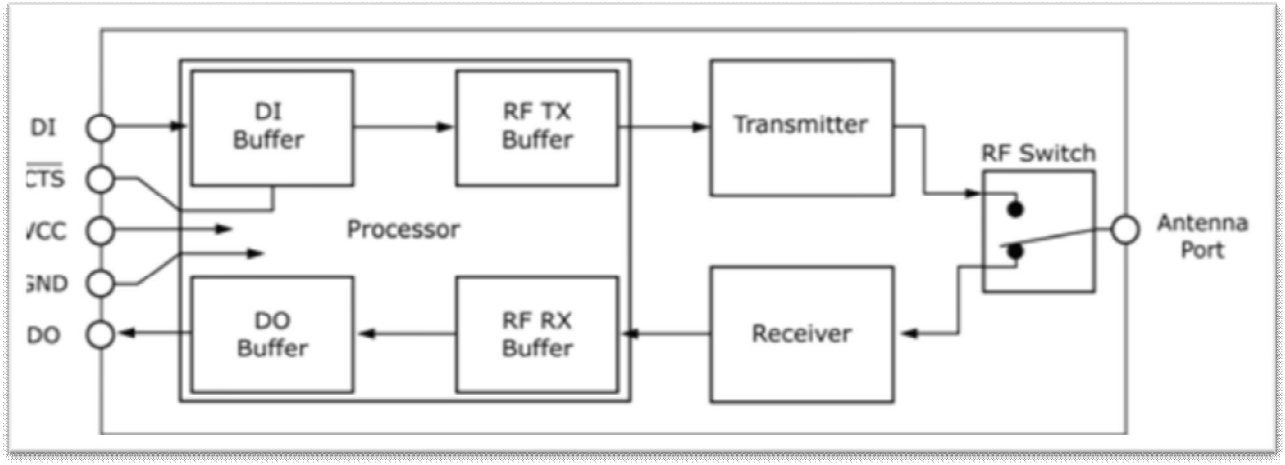


Figure 5: Wireless Module Internal Architecture [20]

DI (Data In) Buffer

The data before being further processed, is stored in the DI Buffer as soon as it enters the RF Module through the Data In pin.

Hardware Flow Control (CTS)

The module automatically de-asserts CTS to signal to the host device when only 17 bytes worth of memory is left in the buffer so that no more data is sent. The data is resumed as soon as 34 bytes of memory are available whence the CTS is re-asserted [24].

DO (Data Out) Buffer

Upon receiving the RF data, it enters the DO buffer and is relayed to the serial port of a host device. Once the DO Buffer fills to its capacity, any additional incoming RF data is lost.

Hardware Flow Control (RTS)

Data will not be sent out the DO Buffer as long as RTS (pin 16) is de-asserted, if RTS is enabled for flow control [21].

2.7 Serial Interface Protocols

The RF modules support two interfaces:

1. Transport Interface
2. Application Program Interface

Transparent Operation

In transparent mode of operation, the modules replace lines serially. All UART data received through the DIN pin is queued up for RF transmission. The module receives the RF data and sends it out on DOUT pin. AT command mode interface is used to configure module.Buffering of the data takes place which later initiates packetization by either of the following means:

- Packetization Timeout occurs i.e. no serial characters are received for the amount of time determined by the RO parameter.
- If RO = 0, packetization begins when a character is received.
- The Command Mode Seque2nce (GTb+ CC + GT) is received.
- The maximum numbers of characters that will fit in an RF packet is received [22].

API Operation

An alternative solution to using Transparent Operation is the API Operation. API allows a host application to interact with the network capabilities of the module.In the API mode, all data entering and leaving the module is contained in frames. These frames define operations or events within the module.

- Transmit Data Frames. Equivalent to AT commands.
- Receive Data Frames. Sent out the DOUT pin (pin 2).

Event notifications such as reset, associate, disassociate, etc are included [23].

Alternative means for configuring the modules and routing of the data at the host application layer are taken care of by API. A data frame is sent by the host application to the module that contain address and payload information rather than using command mode to modify

address. The frames sent by the module contains status packet, source and payload information from the data packets received.

Data is transmitted to several destinations without entering Command Mode. Success/failure status of each transmitted RF Packet is received. The identification of source address of each data received are among the various operations allowed by the API.

Transparent and API Operation Comparison [25]

Transparent Operation Features	
Simple Interface	All received serial data is transmitted unless the module is in command mode.
Easy to support	It is easier for an application to support transparent operation and command mode.
API Operation Features	
Easy to manage data transmissions to multiple destinations	Transmitting RF data to multiple remotes only requires changing the address in the API frame. This Process is much faster than transparent operation where the application must enter AT command mode, change the address, exit command mode, and then transmit data. Each API transmission can return a transmit status frame indicating the success or reason for failure
Received data frames indicate the sender's address	All received RF data API frames indicate the source address.
Advanced Networking diagnostics	API frames can provide indication of IO samples from remote devices, transmission status messages, and local radio status messages.
Remote Configuration	Set/read configuration commands can be sent to remote devices to configure them as needed using the API.

Figure 6: Transparent and API Mode Comparison

2.8 Modes of Operation

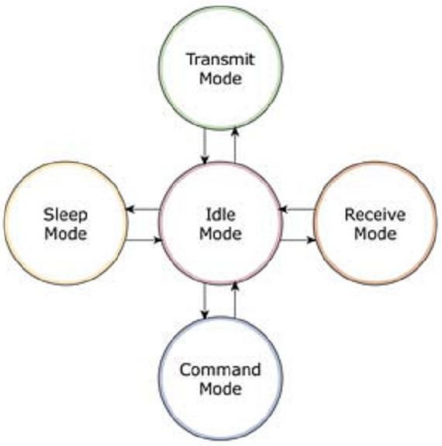


Figure 7: Operation Modes [27]

Idle Mode

Idle mode is the one in which the RF module is neither receiving nor transmitting any data. The various conditions in which the module shifts into other modes of operation are as follows [26]:

- Transmit Mode
- Receive Mode
- Sleep Mode
- Command Mode

Transmit Mode

Initially, the RF module is in idle mode but as soon as serial data is received and ready for packetization, it exits the idle mode and tries to transmit the data. The node whose destination address is there, receives the data. The module first checks the 16 bit network address and makes sure that a route to destination node is established. Network discovery is done in case the network address is unknown.

In case the route is not known, route discovery will take place so as to establish a route to the destination node. If a match in the network address of the module does not occur, the packet is discarded.

The data is transmitted upon the establishment of the route. If route discovery is not able to establish a route, the packet will be discarded. Upon successfully receiving data, the network layer of the destination node sends acknowledgement packet to the source node through the established route. In case the source does not receive the acknowledgement from the destination it assumes that the packet was not received and re transmits the data [28].

A rare case may be that the destination receives the data but the acknowledgement it sends gets lost. In such case the source will re-transmit the data which will unfortunately be accepted by the destination rather than discarding. It is so because the Xbee modules do not filter out duplicate packets. The application should include provisions to address this potential issue.

Transmit Mode Flow Sequence

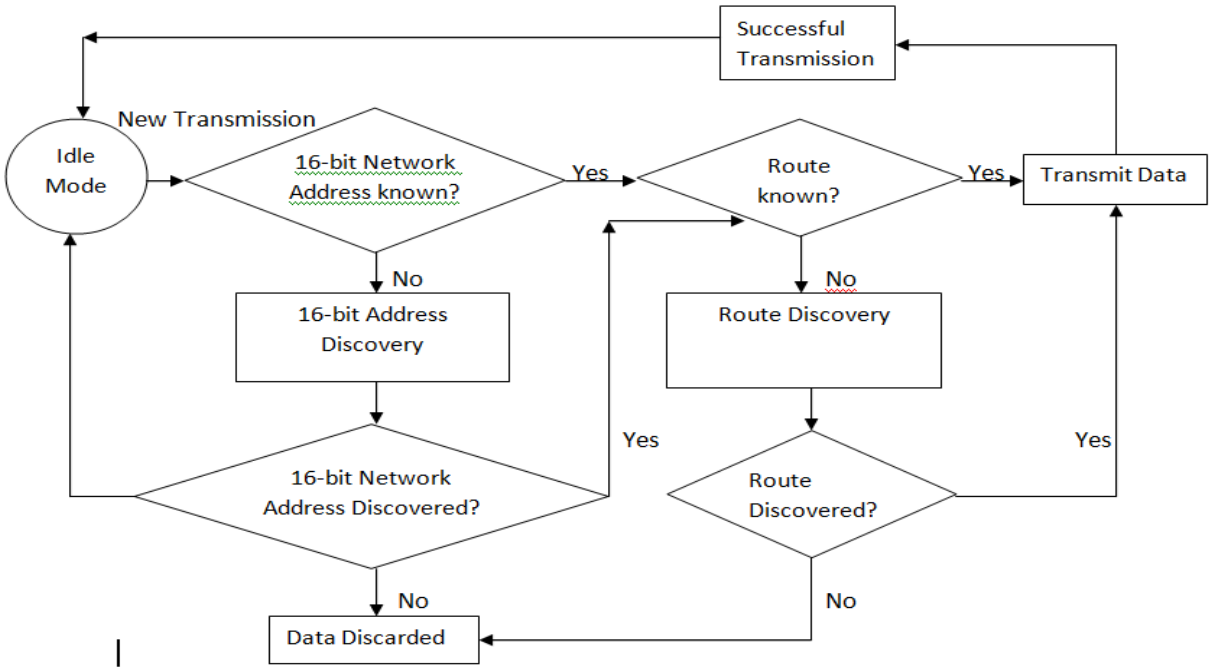


Figure 8: Transmit Flow [30]

Receive Mode

If the RF packet received is valid, the data is transferred to the serial transmit buffer.

Command Mode

The Command Mode needs to be entered by the module before it can interpret the incoming serial characters as commands.

AT Command Mode

To Enter AT Command Mode Send the 3 character command sequence "+++" and observe guard times before and after the command characters

- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]
- Input three plus characters ("+++") within one second (CC(Command Sequence Character)) parameter = 0x2B]
- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]

Once the AT command mode sequence had been issued, the module sends an “OK\r” out the DOUT pin. The “OK\r” characters can be deslayed if the module has not finished transmitting received serial data.

When command mode has been entered, the command mode timer is started (CT command), and the module is able to receive AT commands on the DIN pin. All of the parameter values in the sequence can be modified to reflect user preferences.

By default, the BD (baud rate) parameter = 3 (9600 bps)j. To Send AT Commands: Send AT commands and parameter using the syntax shown below.

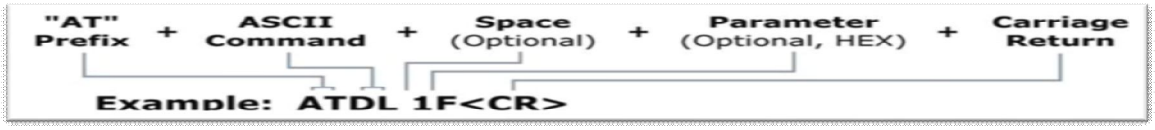


Figure 9: AT Command Format [29]

Sending the WR (Write) command will store new value to non-volatile memory. Changes must be saved to non-volatile memory using the WR (Write) Command to persist in the module’s registry after a reset. Else, parameters are restored to previous values after the module is reset.

Command Response

When we send a command to the module, it will parse and execute the command. Upon successful execution of a command, the module returns an “OK” message. If execution of a command results in an error, the module returns an “ERROR” message.

Applying Command Changes

The changes made to the configuration command registers through AT commands will not take effect until the changes are applied. Changes can be applied in one of the following ways:

- Issue the AC (Apply Changes) command.
- Exit the AT command mode.

To exit AT Command Mode:

To exit AT Command mode, first we send the ATCN command. If no valid AT commands are received within the time specified by CT (Command Mode Timeout) Command, the RF module automatically returns to Idle Mode.

Sleep Mode

When not in use, Sleep modes allow the RF module to enter states of low power consumption. The Xbee RF modules supports two types of sleep:

- Pin sleep
- Cyclic sleep

CHAPTER 3

SYSTEM DESIGN

3.1 Design Modules

The total System is combination of different sections like Temperature & Humidity Sensor , sprinkler controlling hardware and Zig-Bee module Embedded On Single PCB.

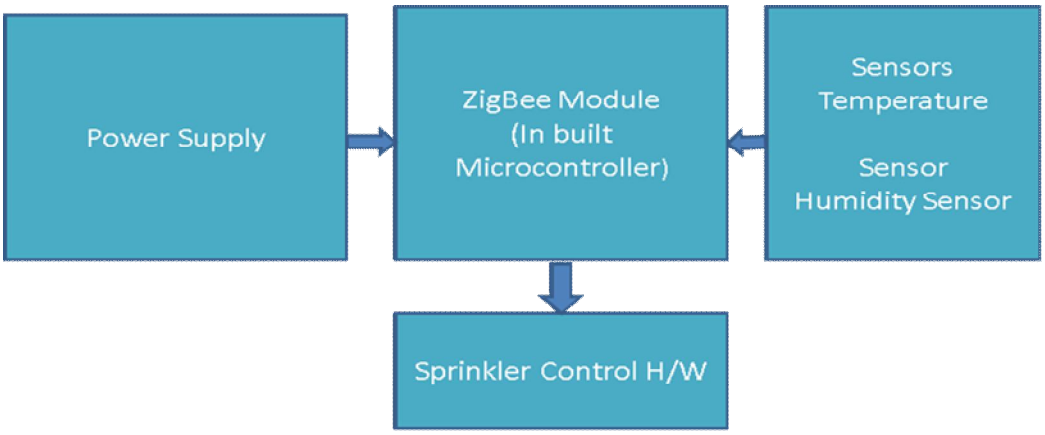


Figure 10: System Design Blocks [31]

3.2 Block Diagram

Single Sensor Node

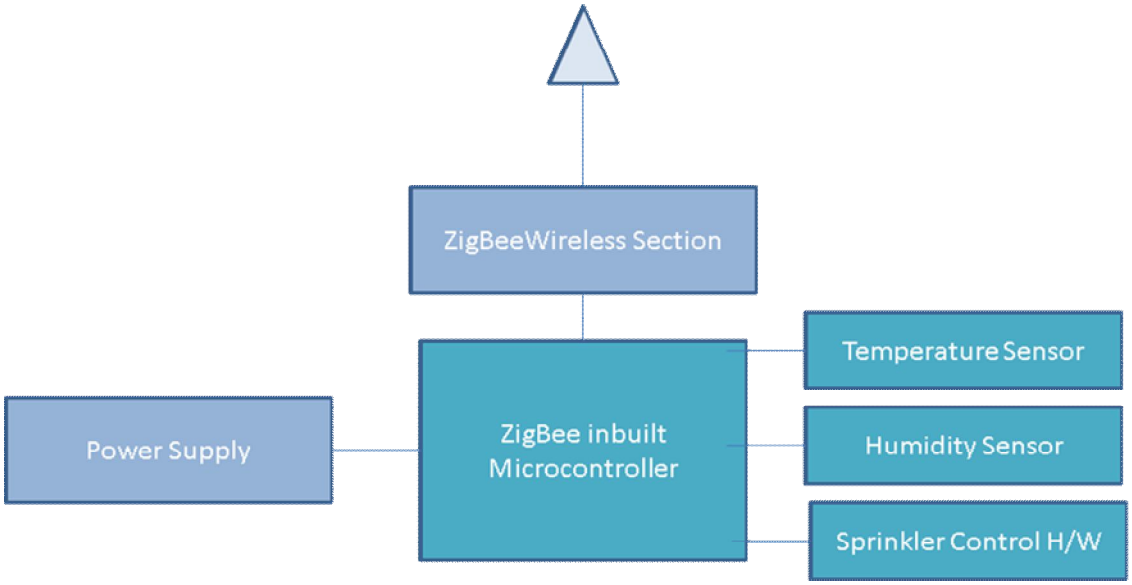


Figure 11: Block diagram of Single Sensor Node

Complete System

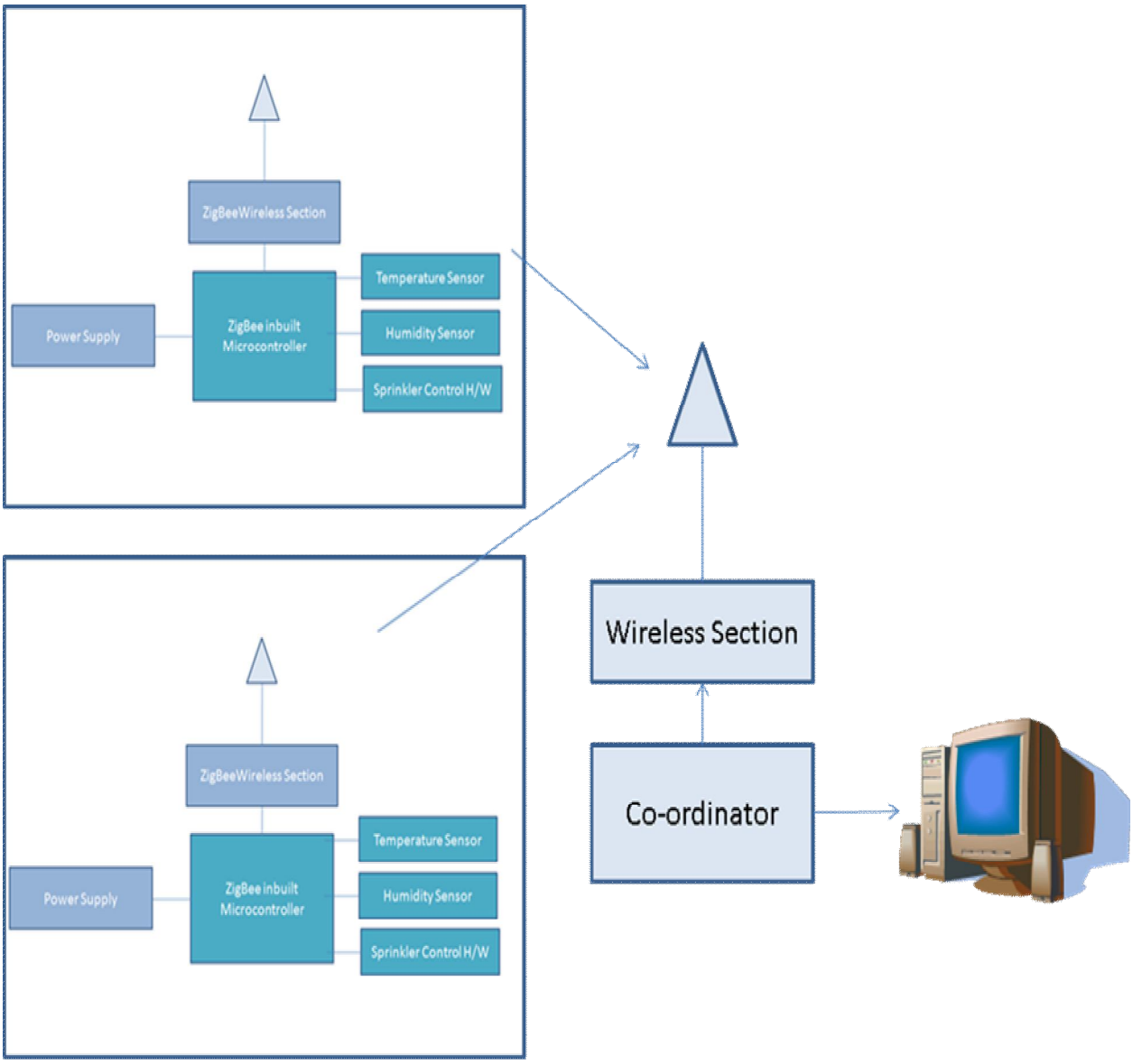


Figure 12: Block diagram of Complete System [31]

3.3 Block Diagram Description

The Sensor Node is connected with sensor which provides Temperature and Humidity parameters of present place. We can provide supply externally or via battery depends upon availability. The inbuilt Microcontroller of Zig-Bee module reads the Temperature and Humidity from the sensors using serial communication and sends via Zig-Bee wireless section to base station.

Temperature and Humidity Sensor

Here we are using Humidity and Temperature Sensor which provide direct serial out which we read through XBee Transmitter.

Wireless Receiver

Sensor data is received by wireless receiver in digital format, receiver section is based on Digi XBee wireless modem. It receives data and send to Host PC through Serial Port.

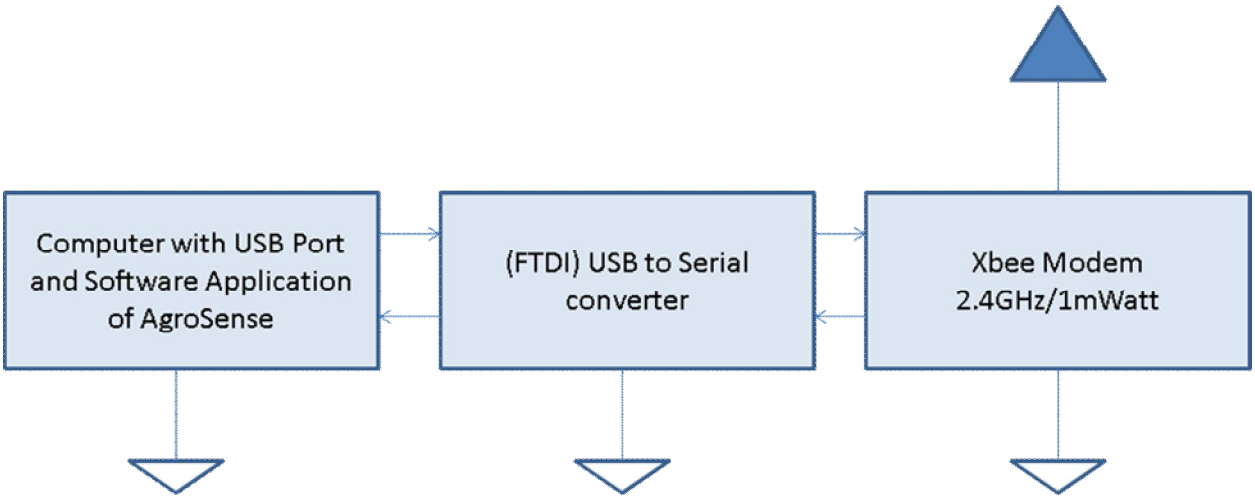


Figure 13: Block Diagram of Device at Control Room [32]

CHAPTER 4

HARDWARE DESIGN DETAILS

4.1 2Power Supply Section – Schematic Layout

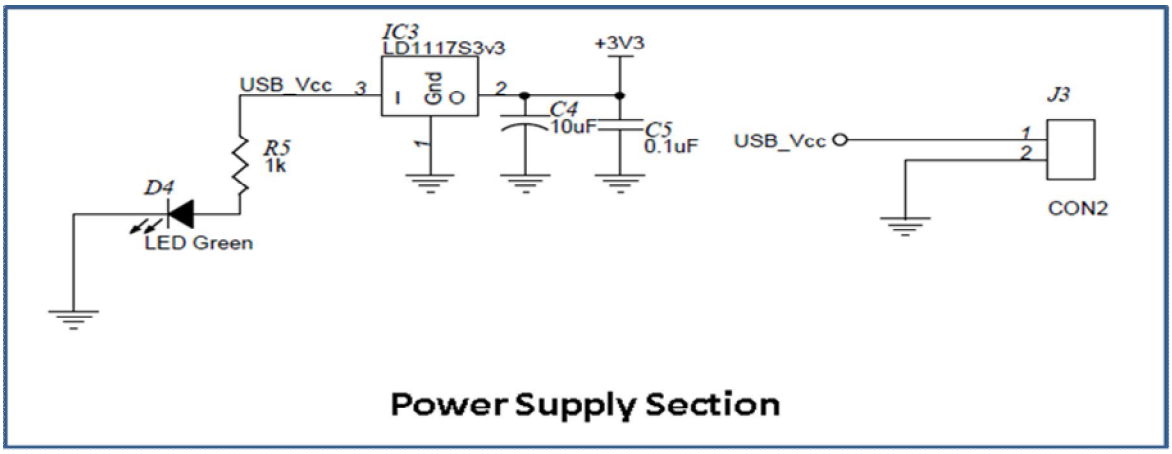


Figure 14: Schematic Layout of Power Supply [35]

4.2 Power Supply Section – Schematic Description

Module uses a power +5V directly from USB connector of PC or some external +5V. IC LD1117S3v3 is used to regulate power supply of 3.3V which is requirement of Xbee Modem. Capacitors are used as filters and LED is used for power indication purpose.

4.3 Power Supply-Bill of Material

Name of Item	**Legend**
Voltage Regulator LD1117S3v3	IC3
LED	D4
Resistance 1K Ohm	R5
Capacitor 10uF/25V	C4
Capacitor 0.1uF/25V	C5
2 Pin Connector	J3

Table 4: Power Supply Section BOM

4.4 Receiver Section – Schematic Layout

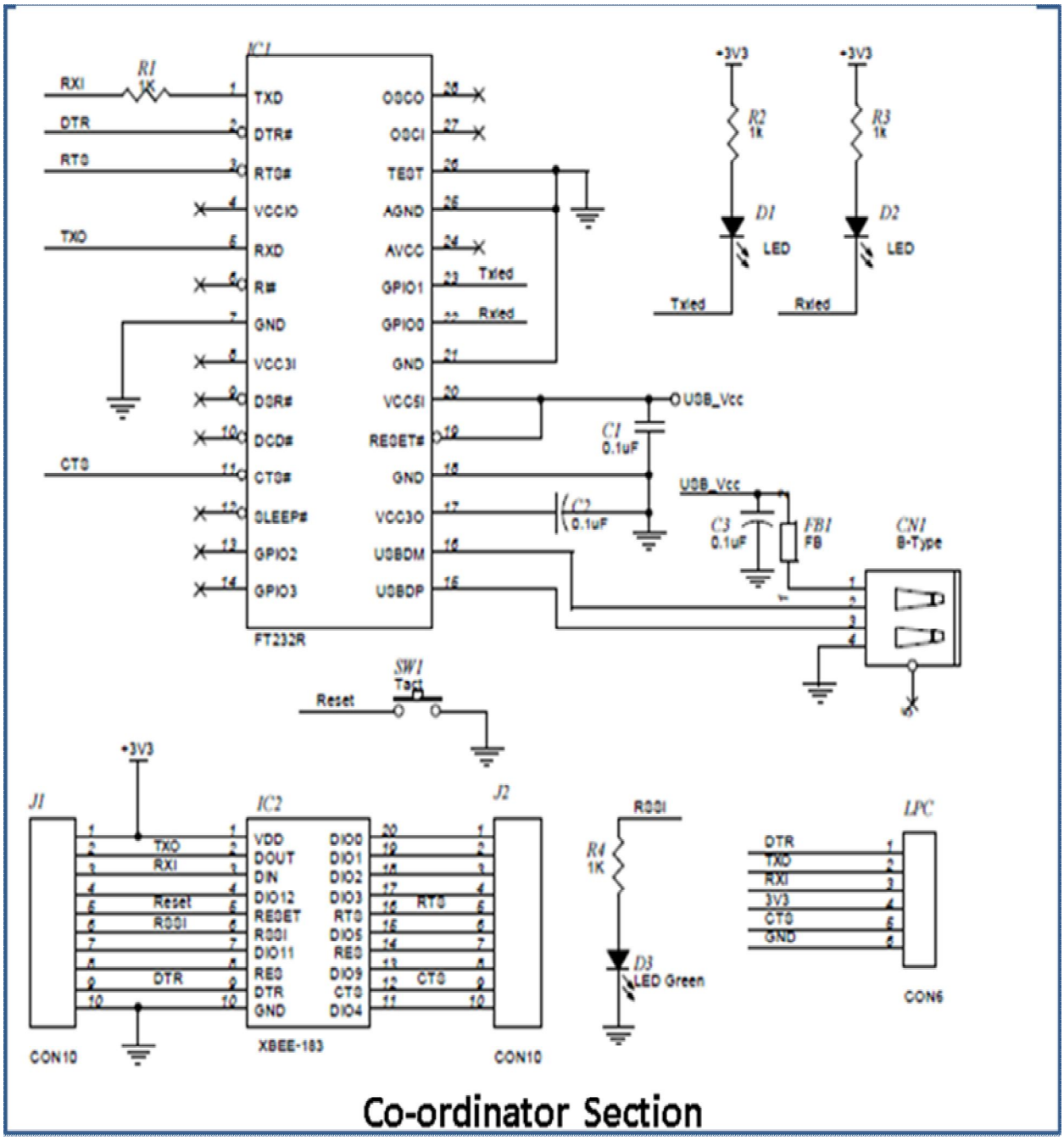


Figure 15: Schematic Layout of Receiver Section

4.5 Receiver Section – Schematic Description

The Receiver section system is the heart of the Wireless Network. It collects data from various nodes and sends to Control and Monitoring base station.

4.6 Receiver Section – Bill of Material [33]

Name of Item	Legend
IC FT232R	IC1
LED SMD	D1, D2, D3
B-Type Female USB Connector	CN1
Digi XB24 Module	XBEE-183

Table 5: Receiver Section BOM

4.7 Receiver Section – PCB Layout (Bottom Side)

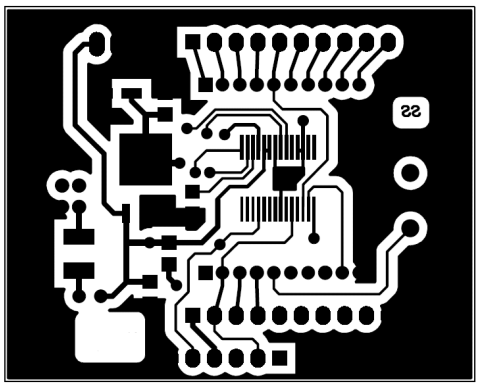


Figure 16: PCB of Receiver Section (Bottom Side)

4.8 Receiver Section – PCB Layout (Top Side)

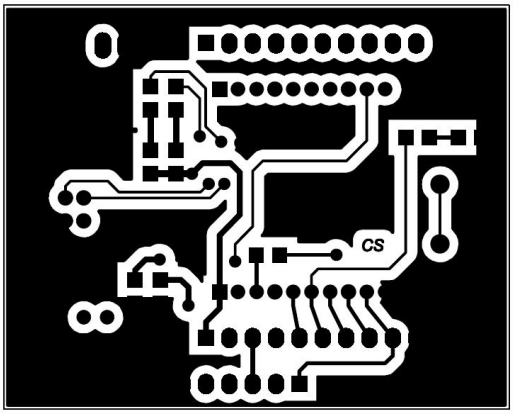


Figure 17: PCB of Receiver Section (Top Side)

4.9 Receiver Section – PCB Layout (Top Component Side)

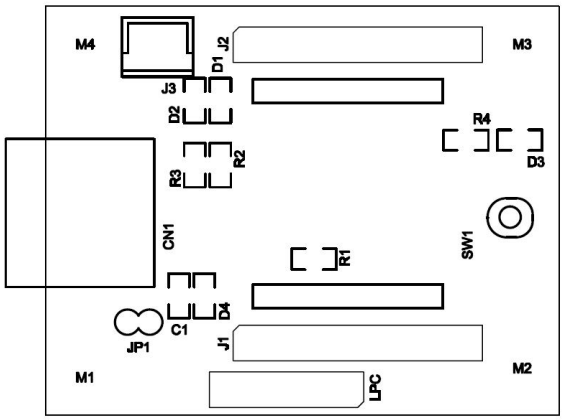


Figure 18: PCB of Receiver Section Component Side

4.10 End Device Node Section – Schematic Layout

It is same as Coordinator Schematic, additionally contains Sensor and Relay control circuitry [34].

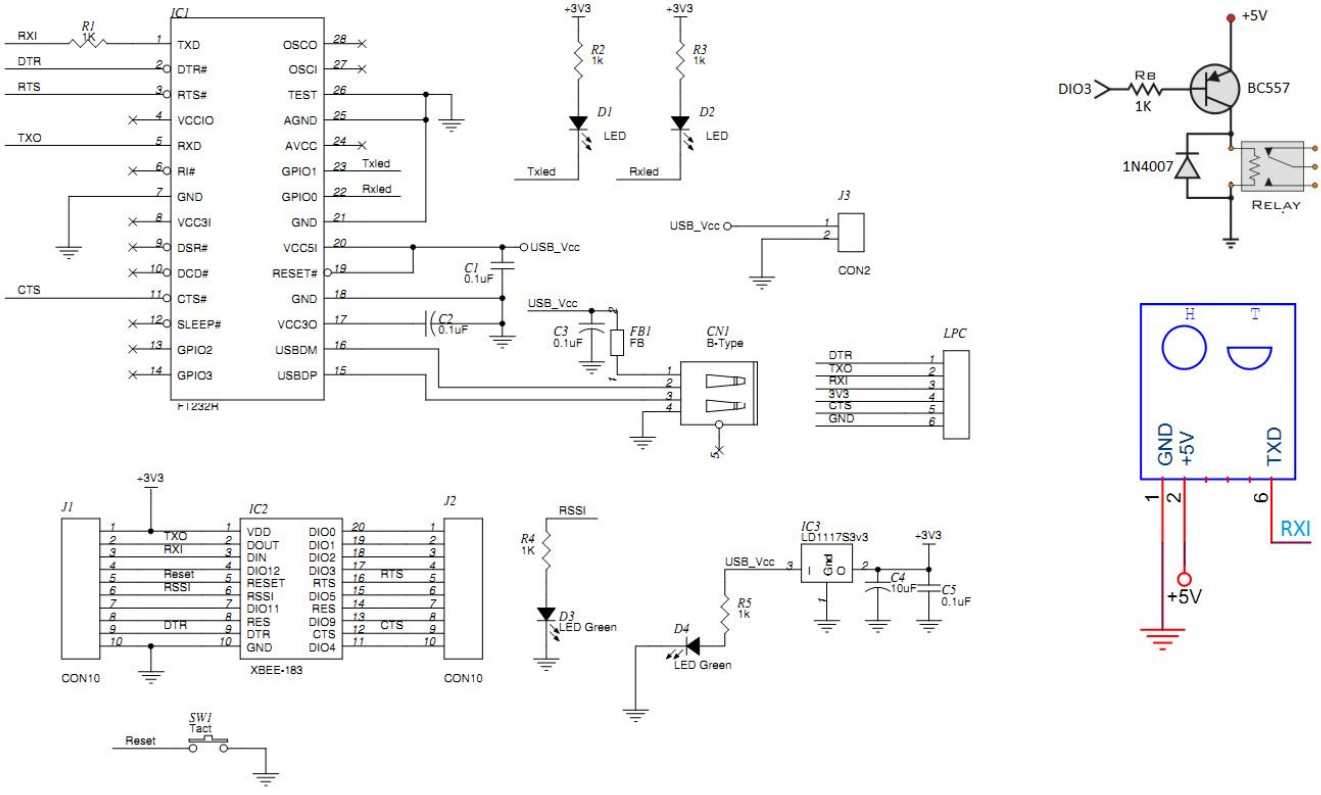


Figure 19: Schematic Layout of Transmitter Section

4.11 End Device Node Section – PCB Layout

It is same as Coordinator Layout

4.12 End Device (Transmitter) Section – Schematic Description

The end Device work as a transmitter for Wireless Network. It sends data from Sensors connected to it to Coordinator Section (Receiver) Monitoring base station.

4.13 End Device (Transmitter) Section – Bill of Material

Name of Item	Legend
IC FT232R	IC1
LED SMD	D1, D2, D3
B-Type Female USB Connector	CN1
Digi XB24 Module	XBEE-183

Table 6: Transmitter Section BOM

4.14 Sensor Section – Schematic Layout [36]

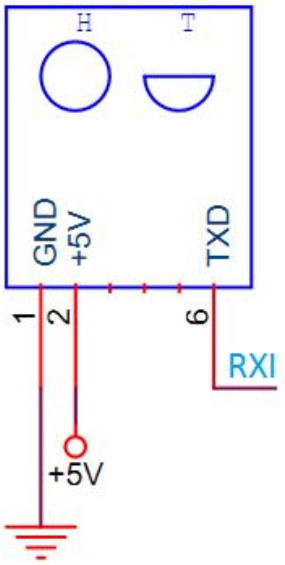


Figure 20: Schematic Layout of Sensor Section

4.15 Sensor Section – Schematic Description

This module sends data on Serial Terminal which we receive directly on Transmitter mode and send wirelessly.

4.16 Sensor Section – Bill of Material

Name of Item	**Legend**
LM35	IC1
Humidity Sensor	IC2

Table 7: Sensors BOM

CHAPTER 5

TOOLS, FIRMWARE AND SOFTWARE

5.1 Tools used for designing Hardware and Software

The following are the tools that are used for designing the prototype of Zig-Bee based Wireless Sensor Network.

Circuit Designing

EAGLE Evaluation Version with limited functionality

PCB Designing

Zuken: CADStar 8.1

Microcontroller IDE

Arduino

DigiXBee Firmware Burning Tool

X-CTU Terminal Software

Operating System Platform

UBUNTU 12.04(Linux)

Man Machine Interface Platform (Software on PC)

Qt 4.7

Graph Monitoring Tool

Meters used for Testing and Analysis of Prototype

1. Multimeter 4011 from Scientech Technologies Pvt, Ltd., Indore
2. Oscilloscope: Caddo 831 from Scientech Technologies Pvt, Ltd., Indore
3. Multimeter 4011 from Scientech Technologies Pvt, Ltd., Indore
4. Digital Storage Scope: CD100i from Scientech Technologies Pvt, Ltd., Indore

5.2 Firmware for Digi- XBee Receiver

Below are the settings done for Receiver Firmware

xb24_15_4_10e6.mxi	->	Modem_Function_Version
[A]CH=C	->	Set/read channel number (Uses 802.15.4 channels).
[A]ID=1234	->	PAN ID
[A]DH=0	->	Destination Address High
[A]DL=FFFF	->	Destination Address Low
[A]MY=5678	->	Own Address
[A]MM=0	->	MAC Mode
[A]RR=3	->	XBee Retries
[A]RN=0	->	Random Delay Slots
[A]NT=19	->	Node Discovery Time
[A]NO=0	->	Node2 Discovery Options
[A]CE=1	->	Coordinator Enable
[A]SC=1FFE	->	Scan Channels
[A]SD=4	->	Scan Duration
[A]A1=0	->	End Device Association
[A]A2=0	->	Coordinator Association
[A]D8=0	->	DIO8 Disabled
[A]D7=1	->	DIO7 – CTS Flow Control
[A]D6=0	->	DIO6 Disabled
[A]D5=0	->	DIO5 Diabled

[A]D4=3	->	DIO4 – Digital Input
[A]D3=3	->	DIO3 – Digital Input
[A]D2=3	->	DIO2 – Digital Input
[A]D1=3	->	DIO1 – Digital Input
[A]D0=3	->	DIO0 – Digital Input
[A]IU=1	->	Input/Output Enable
[A]IT=2	->	Samples before TX
[A]IC=18	->	DIO Change Detect
[A]IR=32	->	Sample Rate

5.3 Firmware for Digi- XBee End Device Node1

xb24_15_4_10e6.mxi->		Modem_Function_Version
[A]CH=C	->	Set/read channel number (Uses 802.15.4 channels).
[A]ID=1234	->	PAN ID
[A]DH=0	->	Destination Address High
[A]DL=5678	->	Destination Address Low
[A]MY=4321	->	Own Address
[A]MM=0	->	MAC Mode
[A]RR=0	->	XBee Retries
[A]RN=1	->	Random Delay Slots
[A]NT=19	->	Node Discovery Time
[A]NO=0	->	Node Discovery Options
[A]CE=0	->	Coordinator Disable
[A]SC=1FFE	->	Scan Channels
[A]SD=4	->	Scan Duration
[A]A1=0	->	End Device Association
[A]A2=0	->	Coordinator Association
[A]D8=0	->	DIO8 Disabled
[A]D7=1	->	DIO7 – CTS Flow Control
[A]D6=0	->	DIO6 Disabled
[A]D5=0	->	DIO5 Disabled
[A]D4=4	->	DIO4 Digital Out - Low

[A]D3=4 -> DIO3 Digital Out - Low

[A]D2=0 -> DIO2 Disabled

[A]D1=0 -> DIO1 Disabled

[A]D0=0 -> DIO0 Disabled

[A]IU=1 -> Input/Output Enable

[A]IT=1 -> Samples before TX

[A]IC=0 -> DIO Change Detect

[A]IR=400 -> Sample Rate

5.4 Firmware for Digi- XBee End Device Node2

xb24_15_4_10e6.mxi->		Modem_Function_Version
[A]CH=C	->	Set/read channel number (Uses 802.15.4 channels).
[A]ID=1234	->	PAN ID
[A]DH=0	->	Destination Address High
[A]DL=5678	->	Destination Address Low
[A]MY=1234	->	Own Address
[A]MM=0	->	MAC Mode
[A]RR=0	->	XBee Retries
[A]RN=0	->	Random Delay Slots
[A]NT=19	->	Node Discovery Time
[A]NO=0	->	Node Discovery Options
[A]CE=0	->	Coordinator Disable
[A]SC=1FFE	->	Scan Channels
[A]SD=4	->	Scan Duration
[A]A1=0	->	End Device Association
[A]A2=0	->	Coordinator Association
[A]D8=0	->	DIO8 Disabled
[A]D7=1	->	DIO7 – CTS Flow Control
[A]D6=0	->	DIO6 Disabled
[A]D5=0	->	DIO5 Disabled
[A]D4=4	->	DIO4 Digital Out - Low

[A]D3=4 -> DIO3 Digital Out - Low

[A]D2=0 -> DIO2 Disabled

[A]D1=0 -> DIO1 Disabled

[A]D0=0 -> DIO0 Disabled

[A]IU=1 -> Input/Output Enable

[A]IT=1 -> Samples before TX

[A]IC=0 -> DIO Change Detect

[A]IR=400 -> Sample Rate

5.5 Application Interface Software

Main Window

```
#include <QtGui/QApplication>
#include "mainwindow.h"

class I : public QThread
{
public:
      static void sleep(unsigned long secs) {
            QThread::sleep(secs);
      }
};

int main(int argc, char *argv[])
{
   QApplication a(argc, argv);

   QPixmap pixmap("./image/Splash.png");
   QSplashScreen splash(pixmap);
   splash.show();
//   splash.showMessage("Wait...");
   splash.showMessage(QObject::tr("Loading Programs"),
                  Qt::AlignCenter , Qt::magenta);
   I::sleep(1);
   splash.showMessage(QObject::tr("Loading Program*"),
                  Qt::AlignCenter , Qt::magenta);

   I::sleep(1);
   splash.showMessage(QObject::tr("Loading Progra**"),
                  Qt::AlignCenter , Qt::magenta);
   I::sleep(1);
   splash.showMessage(QObject::tr("Loading Progr***"),
                  Qt::AlignCenter , Qt::magenta);

   splash.showMessage(QObject::tr("Loading Prog****"),
                  Qt::AlignCenter , Qt::magenta);

   splash.showMessage(QObject::tr("Loading Pro*****"),
                  Qt::AlignCenter , Qt::magenta);

   splash.showMessage(QObject::tr("Loading Pr******"),
                  Qt::AlignCenter , Qt::magenta);

   splash.showMessage(QObject::tr("Loading P*******"),
                  Qt::AlignCenter , Qt::magenta);

   splash.showMessage(QObject::tr("Loading ********"),
```

```
                Qt::AlignCenter , Qt::magenta);

    splash.showMessage(QObject::tr("Loadin* ********"),
                Qt::AlignCenter , Qt::magenta);

    splash.showMessage(QObject::tr("Loadi** ********"),
                Qt::AlignCenter , Qt::magenta);

    splash.showMessage(QObject::tr("******* ********"),
                Qt::AlignCenter , Qt::magenta);
    I::sleep(1);
    splash.showMessage(QObject::tr("*****************"),
                Qt::AlignCenter , Qt::magenta);

//    splash.showMessage(QObject::tr("                Loading Programs....."),
//                  Qt::AlignLeft | Qt::AlignTop, Qt::black);

    qApp->processEvents();
    QMainWindow window;
    window.setWindowTitle("Qt Application"); //Set the title of your main Qt
Application
       window.resize(600, 500);
       I::sleep(1);            // Splash page is shown for 5 seconds
    window.setStyleSheet("* { background-color:rgb(199,147,88); padding: 7px}");

    MainWindow w;
    w.show();

    splash.finish(&window);

    return a.exec();
}



#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <serialportinfo.h>
#include <serialport.h>
#include <QtCore/QVariant>
#include <QApplication>
#include <QHBoxLayout>
#include <QPushButton>
#include <QtGui>
#include <QDialog>
#include <QMessageBox>
#include <QtCore/QCoreApplication>
#include <QTimer>
```

```
Q_DECLARE_METATYPE(SerialPortInfo)

MainWindow::MainWindow(QWidget *parent) :
   QMainWindow(parent),
   ui(new Ui::MainWindow)
{
   ui->setupUi(this);
   procUpdateAvailablePorts();
   port = new SerialPort(this);
   connect(port, SIGNAL(readyRead()), this, SLOT(slotRead()));
   data123 = "0";
   data234 = "0";
   ADC0="0";
   ADC1="0";
2   ADC2="0";
   ADC3="0";
   maindata="";
   monitor_flag = false;
   sprinkler_flag = false;
   cnt = 0;
   t1 = new QTimer(this);
   connect(t1, SIGNAL(timeout()), this, SLOT(Start_t()));
   t1->setInterval(1500);
}

void MainWindow::Start_t()
{
   Proceed(maindata);
   maindata="";
}

MainWindow::~MainWindow()
{
   delete ui;
   t1->stop();
}

/*****************************************************/
/*********Enumerate available Serial Ports***********/
/*****************************************************/
void MainWindow::procUpdateAvailablePorts()
{
   ui->comboBox->clear();
   foreach (SerialPortInfo info, SerialPortInfo::availablePorts())
   {
      QVariant v;
      v.setValue(info);
```

```
        ui->comboBox->addItem(info.portName(), v);
    }
}

/*******************************************************/
/*********Connect/Disconnect Serial Port**************/
/*******************************************************/
void MainWindow::on_Connect_Disconnect_PB_clicked()
{
    QVariant v = ui->comboBox->itemData(ui->comboBox->currentIndex());
    SerialPortInfo info= v.value<SerialPortInfo>();
    port->setPort(info);
    if(ui->Connect_Disconnect_PB->text()=="Connect")
    {
        if(!port->open(SerialPort::ReadWrite | SerialPort::Unbuffered ))
        {
            QMessageBox::information( this, "Information", "Port Open Fail" );
            return;
        }
        else
        {
            port->clearBreak(true);
            port->setRate(SerialPort::Rate9600);
            port->setDataBits(SerialPort::Data8);
            port->setParity(SerialPort::NoParity);
            port->setStopBits(SerialPort::OneStop);
            port->setFlowControl(SerialPort::NoFlowControl);
            QByteArray ba;
            ba.resize(8);
            ba[0]=0x7B;
            ba[1]=0x00;
            ba[2]=0x00;
            ba[3]=0x00;
            ba[4]=0xFF;
            ba[5]=0x00;
            ba[6]=0x00;
            ba[7]=0x7D;
            //port->write(ba);
            t1->start();
            ui->Connect_Disconnect_PB->setText("Disconnect");
            ui->stackedWidget->setCurrentIndex(0);
        }
    }
    else if (ui->Connect_Disconnect_PB->text() == "Disconnect")
    {
        port->close();
        if (!port->isOpen())
            ui->Connect_Disconnect_PB->setText("Connect");
```

```
        else
            QMessageBox::information( this, "Information", "  Close Fail" );
    }
}
/***********When Connect Icon is clicked it opens connection
Window**************************/




/*******************************************************/
/****Read Received data and convert into Hex**********/
/*******************************************************/
void MainWindow::slotRead()
{
    port->clearBreak(true);
    QByteArray ba = port->readAll();
    QString ss(ba.toHex().toUpper());
    maindata=maindata + ss;
}


/*******************************************************/
/*********Manipulate Received String for Values************/
/*******************************************************/
void MainWindow::Proceed(QString Str)
{
    //7E 00 0A 83 43 21 3A 00 01 00 18 00 18 AD 0A 48 3A 30 36 37 20 54 3A 30
33 31 0D
    if(Str.startsWith("7E000A834321") && (Str.mid(28,2) == "0A"))
    {
        sprinkler_status = Str.mid(24,1);
        if(sprinkler_status == "09")
        {
            ui->Manual_LEd_Node1_ON->setValue(1);
            ui->Manual_LEd_Node1_OFF->setValue(0);
        }
        else if(sprinkler_status == "1")
        {
            ui->Manual_LEd_Node1_OFF->setValue(1);
            ui->Manual_LEd_Node1_ON->setValue(0);
        }
        data123=Str.mid(28,25);
        if(data123.isEmpty())
        {
            ADC0_Data = ADC0_Data_Temp;
            ADC1_Data = ADC1_Data_Temp;
        }
```

```
        else
        {
            ADC0=data123.mid(6,6);
            ADC0_MSB = ADC0.mid(1,1);
            ADC0_MidSB = ADC0.mid(3,1);
            ADC0_LSB = ADC0.mid(5,1);
            ADC0_Data = ADC0_MSB + ADC0_MidSB + ADC0_LSB;
            ADC0_Data_Temp = ADC0_Data;

            ADC1=data123.mid(18,6);
            ADC1_MSB = ADC1.mid(1,1);
            ADC1_MidSB = ADC1.mid(3,1);
            ADC1_LSB = ADC1.mid(5,1);
            ADC1_Data = ADC1_MSB + ADC1_MidSB + ADC1_LSB;
            ADC1_Data_Temp = ADC1_Data;
        }
    }

    if(Str.startsWith("7E000A831234") && (Str.mid(28,2) == "0A"))
    {
        sprinkler_status_Node2 = Str.mid(25,1);
        if(sprinkler_status_Node2 == "0")
        {
            ui->Manual_LEd_Node2_ON->setValue(1);
            ui->Manual_LEd_Node2_OFF->setValue(0);
        }
        else if(sprinkler_status_Node2 == "8")
        {
            ui->Manual_LEd_Node2_OFF->setValue(1);
            ui->Manual_LEd_Node2_ON->setValue(0);
        }
        data234=Str.mid(28,25);
        if(data234.isEmpty())
        {
            ADC2_Data = ADC2_Data_Temp;
            ADC3_Data = ADC3_Data_Temp;
        }
        else
        {
            ADC2=data234.mid(6,6);
            ADC2_MSB = ADC2.mid(1,1);
            ADC2_MidSB = ADC2.mid(3,1);
            ADC3_Data = ADC3_MSB + ADC3_MidSB + ADC3_LSB;
            ADC3_Data_Temp = ADC3_Data;
        }
    }
}
```

```
/*******************************************************/
/*********Wrapper to return Sensor Values*************/
/*******************************************************/
int MainWindow::GetADC(int ADCChannel)
{
   if(ADCChannel==0)
   {
      int ADC0_dec = ADC0_Data.toInt();
      return ADC0_dec;
   }
   else
      if(ADCChannel==1)
      {
         int ADC1_dec = ADC1_Data.toInt();
         return ADC1_dec;
      }
      else
         if(ADCChannel==2)
         {
            int ADC2_dec = ADC2_Data.toInt();
            return ADC2_dec;
         }
         else
            if(ADCChannel==3)
            {
               int ADC3_dec = ADC3_Data.toInt();
               return ADC3_dec;
            }
}

/*******************************************************/
/**Open Monitor Window to Obsweerve Sensor output*****/
/*******************************************************/
void MainWindow::on_Monitor_clicked()
{
   ui->stackedWidget->setCurrentIndex(2);
   monitor_flag = true;
   monitor_timer_id=startTimer(2000);
}

/*******************************************************/
/*Timer to continuous Read Values and write into File*/
/*******************************************************/
void MainWindow::timerEvent(QTimerEvent *event)
{
   cnt = cnt + 1;
   ti = QTime::currentTime();
```

```
   da = QDate::currentDate();
   if(monitor_flag == true)
   {
      ui->Humidity_Node_1->setValue(GetADC(0));
      ui->lcdNumber_Humi_Node1->display(GetADC(0));

      ui->Thermo_Node_1->setValue(GetADC(1));
      ui->lcdNumber_Temp_Node1->display(GetADC(1));

      ui->Humidity_Node_2->setValue(GetADC(2));
      ui->lcdNumber_Humi_Node2->display(GetADC(2));

      ui->Thermo_Node_2->setValue(GetADC(3));
      ui->lcdNumber_Temp_Node2->display(GetADC(3));

   }

   if(cnt == 60)
      {
         QFile sensor_log_file("Sensor_Readings.txt");
         sensor_log_file.open(QIODevice::WriteOnly | QIODevice::Text |
QIODevice::Append);
         QTextStream out1(&sensor_log_file);
         out1 << "Sensors Output on " + da.toString() + " " + ti.toString() + "
Temperature Node1:"+ ADC0_Data + " Humidity Node1:" + ADC1_Data + " " +
"\n\t\t\t\t\t " + " Temperature Node2:"+ ADC2_Data + " Humidity Node2:" +
ADC3_Data + "\n\n";
         sensor_log_file.close();

         QFile htn1n2("/home/girish/htn1n2.txt");
         htn1n2.open(QIODevice::WriteOnly | QIODevice::Text);
         QTextStream out2(&htn1n2);
         out2 << ADC0_Data + " " + ADC1_Data + " " + ADC2_Data + " " +
ADC3_Data;
         htn1n2.close();

         cnt = 0;
      }
/**************When Monitor Icon is clicked it opens Monitor
Window*******************************/


   if(sprinkler_flag == true)
      {
         ui->Manual_Done_Temp_LCD_Node1->display(GetADC(1));
         ui->Manual_Done_Humi_LCD_Node1->display(GetADC(0));
         ui->Manual_Done_Temp_LCD_Node2->display(GetADC(3));
         ui->Manual_Done_Humi_LCD_Node2->display(GetADC(2));
```

```
    }
/*****When Manual Control icon is clicked it opens Status
window***************/
}

/*****************************************************/
/*********Return Back to Home Window*****************/
/*****************************************************/
void MainWindow::on_Back_pg2_clicked()
{
   killTimer(monitor_timer_id);
   monitor_flag = false;
   ui->stackedWidget->setCurrentIndex(0);

   ui->Thermo_Node_1->setValue(0);
   ui->Humidity_Node_1->setValue(0);
   ui->Thermo_Node_2->setValue(0);
   ui->Humidity_Node_2->setValue(0);
}

/*****************************************************/
/******Show Sprinkler Staus Window********************/
/*****************************************************/
void MainWindow::on_Manual_Control_clicked()
{
   ui->stackedWidget->setCurrentIndex(3);
   sprinkler_flag = true;
   sprinkler_timer_id = startTimer(2000);
}

/*****************************************************/
/*********Open Serial Connection Window***************/
/*****************************************************/
void MainWindow::on_Serial_Connect_clicked()
{
   ui->stackedWidget->setCurrentIndex(1);
}

/*****************************************************/
/*********Return Back to Home Window*****************/
/*****************************************************/
void MainWindow::on_Manual_Done_Home_PB_clicked()
{
   killTimer(sprinkler_timer_id);
   sprinkler_flag = false;
   ui->stackedWidget->setCurrentIndex(0);
}
```

```
/*******************************************************/
/*********Open Sensor Log Text file******************/
/*******************************************************/
void MainWindow::on_Reading_PB_clicked()
{
   system("/usr/bin/gedit ./Sensor_Readings.txt &");
}

/*************When Reading Icon is clicked it opens Log file for Sensor
Nodes************/


/*******************************************************/
/*****************Open Graph Window********************/
/*******************************************************/
void MainWindow::on_Graph_clicked()
{
   system("./fancybrowser &");
}
/*****When Graph button is clicked it opens Browser window which display last
status of Sensor Nodes***/
```

CHAPTER 6

RESULT ANALYSIS AND CONCLUSION

CHAPTER 6 RESULT ANALYSIS AND CONCLUSION

6.1 Result

This system is working as required, the results is as follows

- The two different parameters Temperature and Humidity are measured.
- Temperature Sensor continuously monitors temperature of surroundings.
- Temperature changes immediately as changes occur at Temperature Sensor.
- Humidity Sensor continuously monitors humidity of surroundings.
- Humidity changes immediately as changes occur at Humidity Sensor.
- Transmitter nodes send sensors value in digital format continuously at receiver end.
- Receiver receives value of Temperature and Humidity Sensors from different wireless nodes and forwards it to Application Software in PC.
- Application Software show status of sensor values according to value received.
- Sensor value is monitored remotely on PC via Software developed and controlled manually using switches from receiver end, remote controlling is working as required.
- Two type of Monitoring is provided, first by continuous monitoring of Sensor nodes and second by plotting sensor values characteristics in graph.
- Graph plot characteristics in day, week, month and yearly format, both monitoring methods are working as required.
- We also provided database facility to maintain database of values of sensors of different nodes with respect to date and time, data is updated regularly with current time stamp.

6.2 Main Window

We have provided three different ways to display the results.

Following figure shows how to connect the master receiver module with software

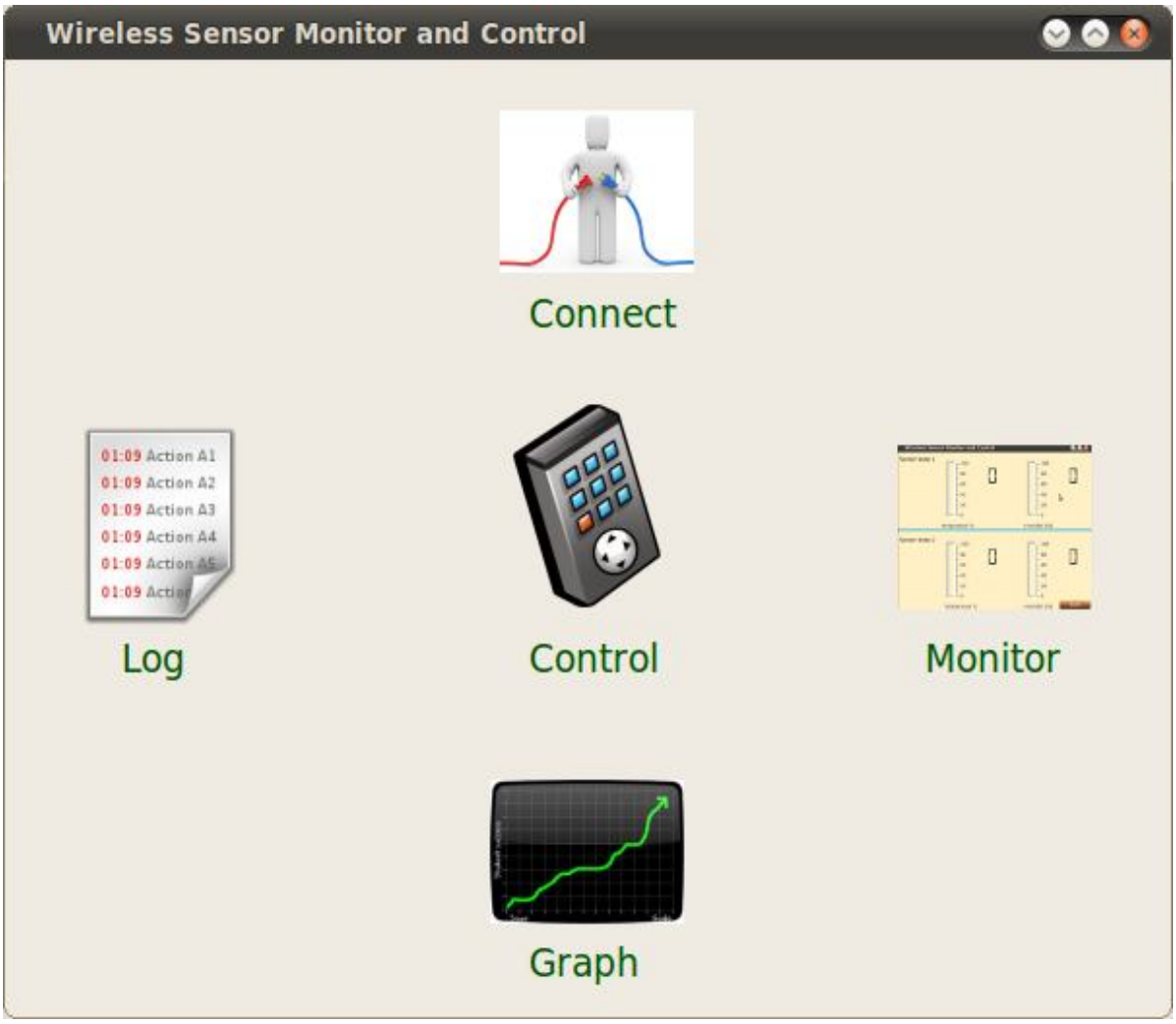


Figure 21: Application Software Home Window

- User has to Click Connect Icon, it opens a window where user have to select com port to which master station is connected, once the connection is established between PC and Master unit the Software is ready to receive data from Sensor Nodes.
- Now user has to connect Sensor Nodes from external supply or via battery and Software starts receiving sensors data
- The measured and monitored data is represented in three different ways on Software

6.2 Monitor Window

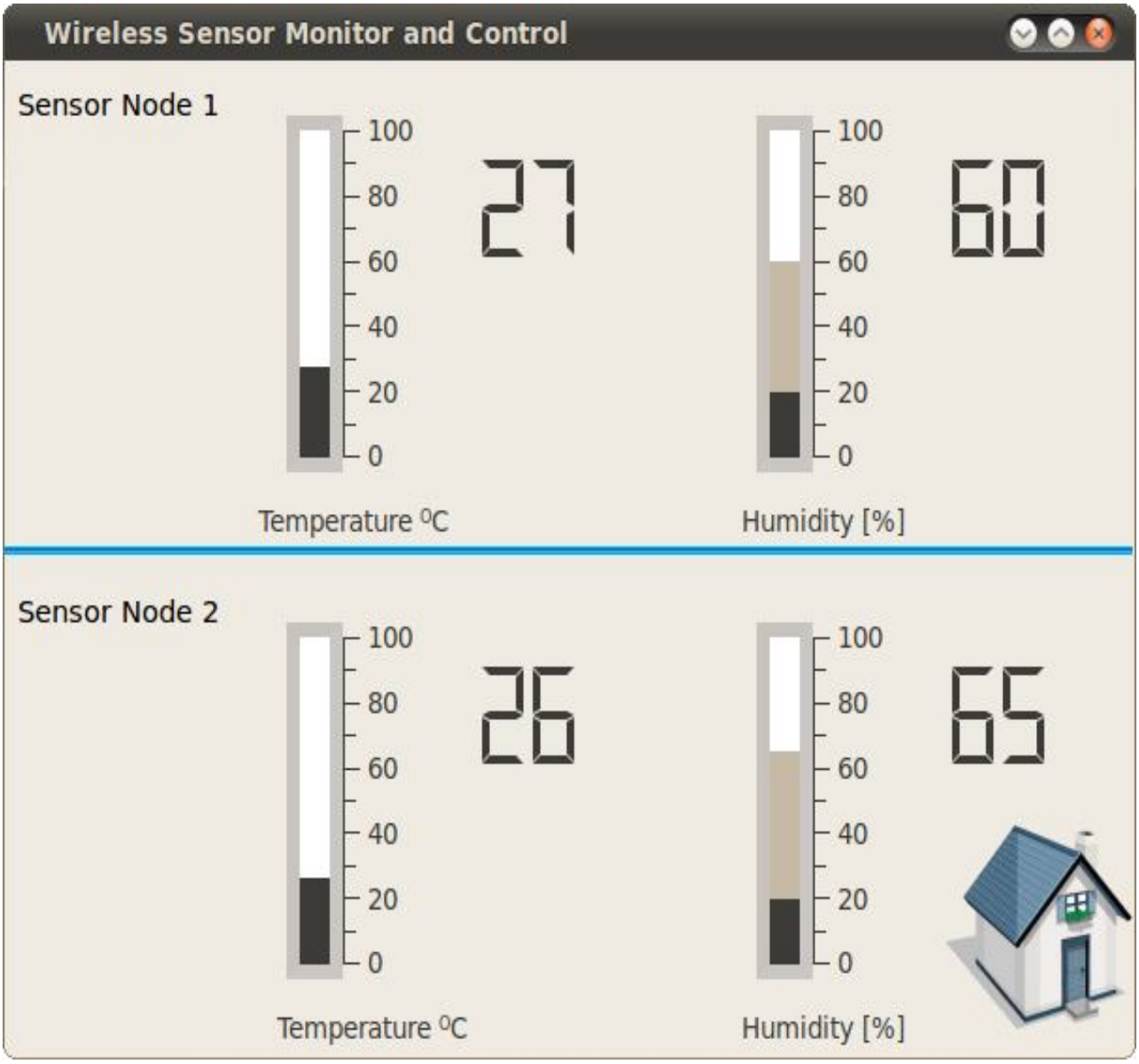


Figure 22: Application Software Monitoring Window

- Values of two wireless sensor nodes are displayed on application software. Values upper from line are Temperature and Humidity values received from sensor node one.
- Values displayed below line are values of sensor node two from remote location.
- A home button is given at bottom right corner to go to home screen at any time.

6.3 Graph Window

Sensor data plots on graph on day, week, month and yearly basis.

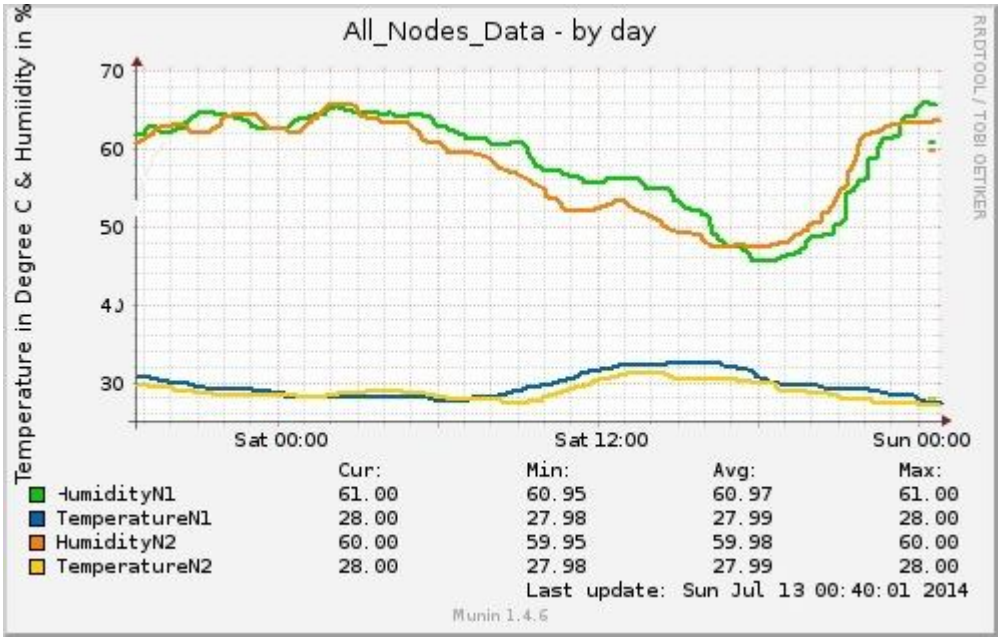


Figure 23: Application Software Graph Window (Day)

Below graph show sensors value of last week

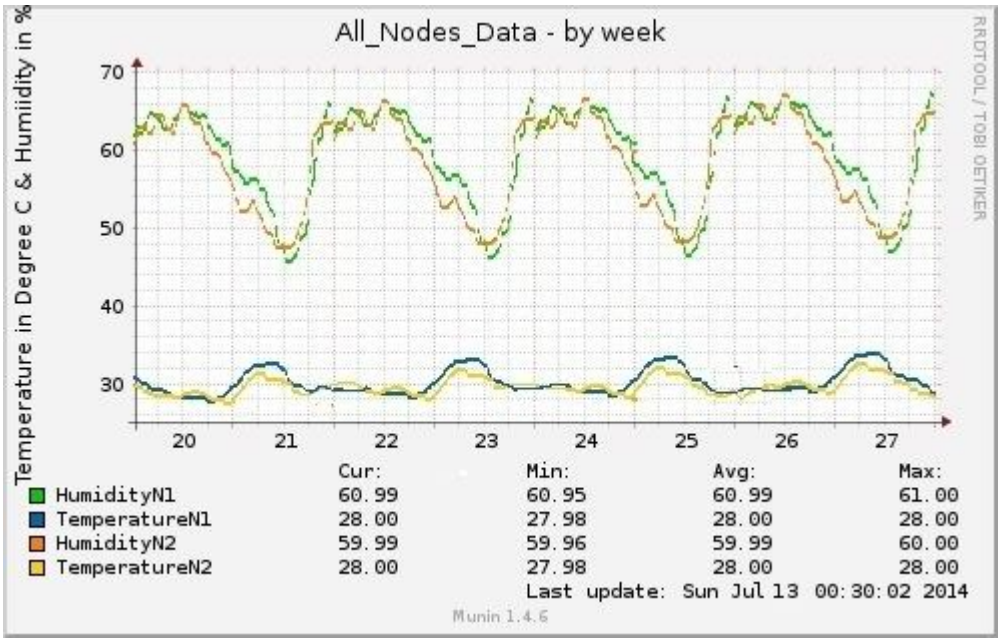


Figure 24: Application Software Graph Window (Week)

Below graph show values of sensors on monthly basis

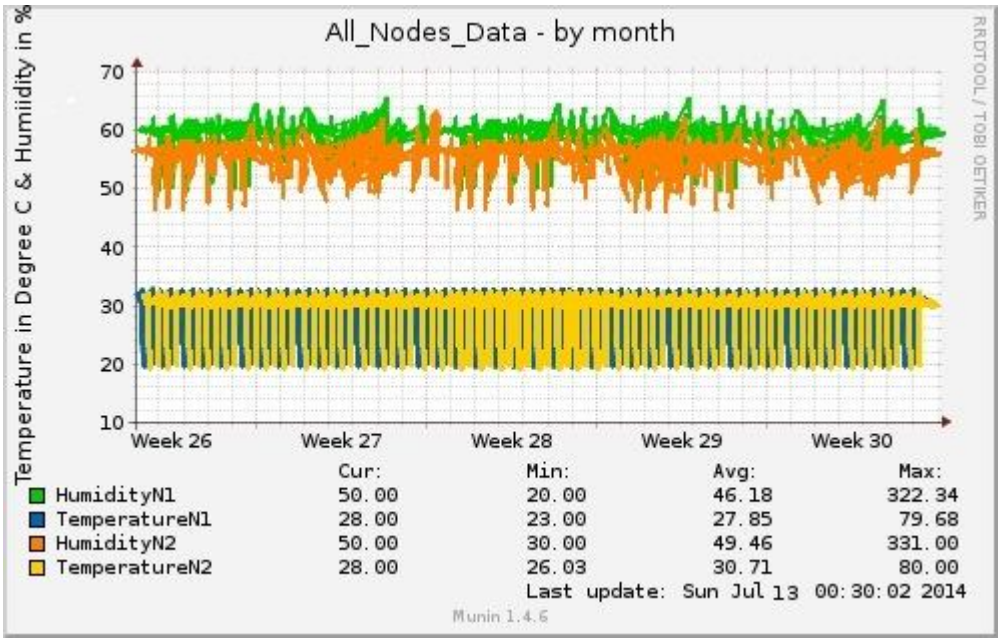


Figure 25: Application Software Graph Window (Month)

Below graph show last year status of sensor outputs

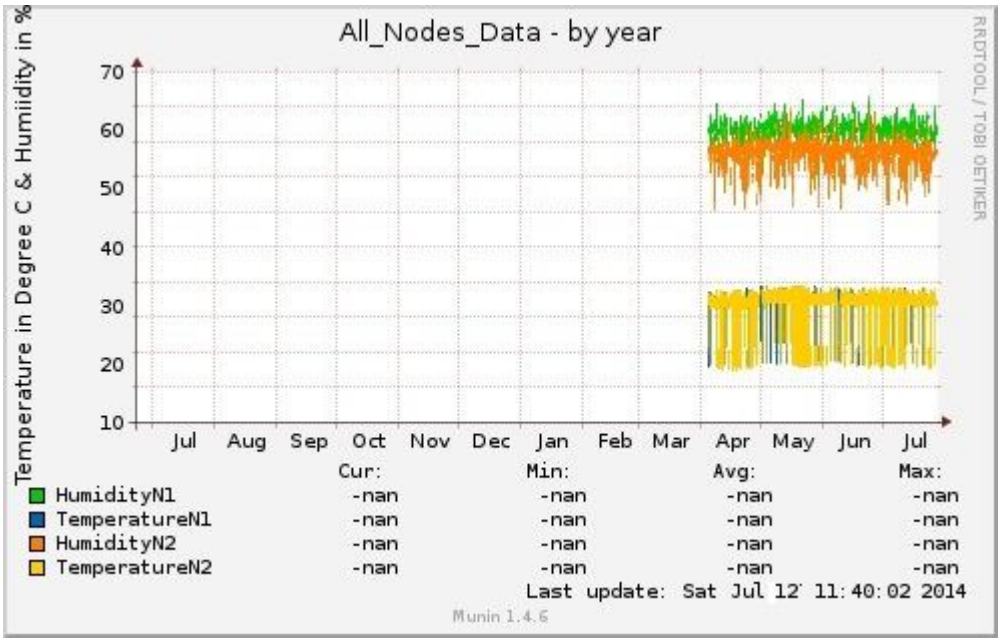


Figure 26: Application Software Graph Window (Year)

6.4 Reading Window

This window shows the value of sensors from different nodes with respect to time and date. File is updated continuously in every two minutes. By using this database user can find out any previous conditions of climate for particular date and time.

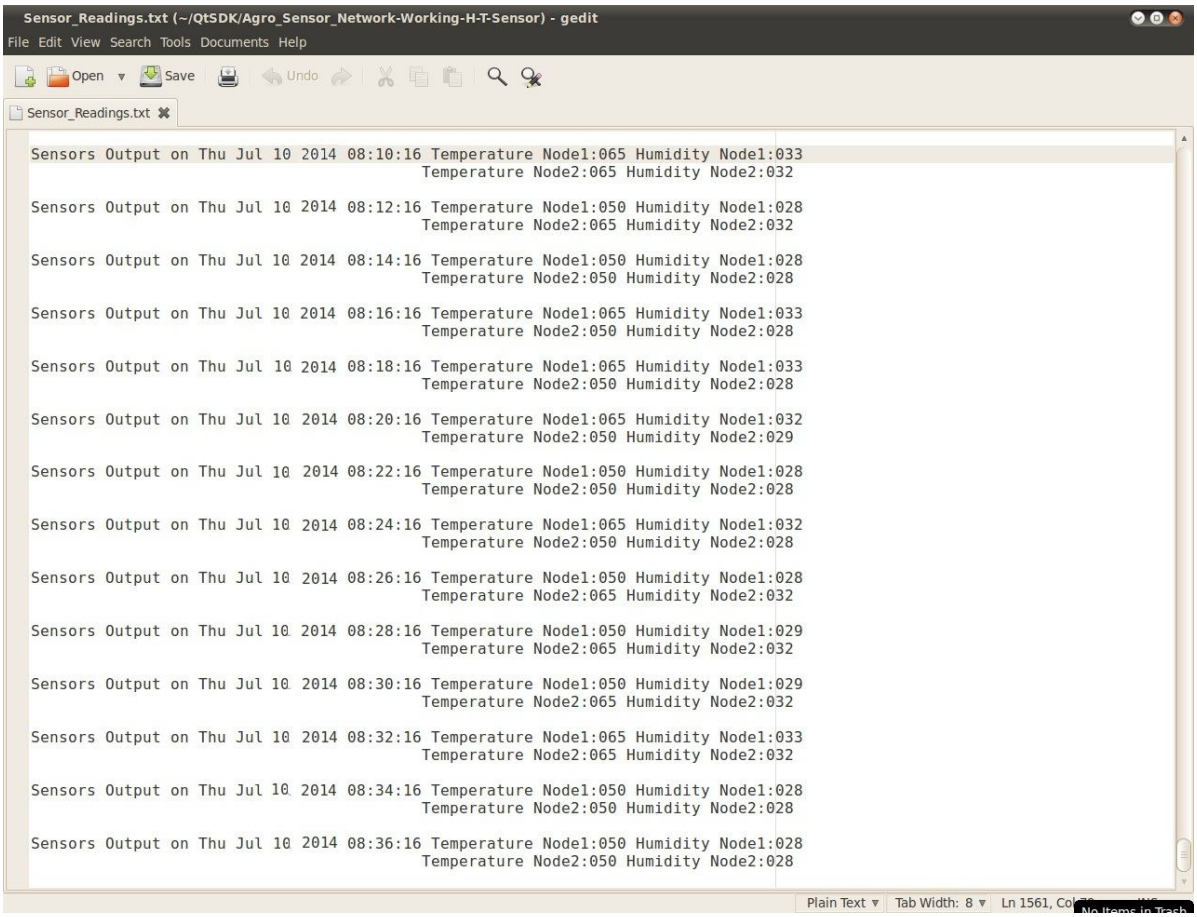


Figure 27: Application Software Reading Window

Analysis:

We have connected two sensor nodes on different places in the range of 20 meter and collected values of Temperature and Humidity from both nodes and found that the nodes are working properly and send values according to climatic condition.

6.5 Analysis of Sensor Node1 data

Time	Temperature Node 1 in O	Humidity Node 1
10:35 AM	25	55
10:55 AM	25	53
11:15 AM	25	51
11:35 AM	26	50
11:55 AM	26	48
12:15 PM	27	45
12:35 PM	27	44
12:55 PM	28	43
01:15 PM	29	42
01:35 PM	30	41
01:55 PM	31	40
02:15 PM	33	37
02:35 PM	33	35
02:55 PM	33	32
03:15 PM	33	30
03:35 PM	33	29
03:55 PM	30	27
04:55 PM	28	25
05:15 PM	28	26
05:55 PM	27	30
06:15 PM	26	32
06:55 PM	26	37
07:15 PM	26	40
07:55 PM	25	45
08:15 PM	24	50
08:55 PM	23	57
09:15 PM	23	60
09:55 PM	23	65

Table 8: Analysis Data of Sensor Node 1

6.6 Analysis of Sensor Node2 data

Time	Temperature Node 2 in OC	Humidity Node 2 in %
10:35 AM	24	57
10:55 AM	24	56
11:15 AM	24	55
11:35 AM	25	50
11:55 AM	25	48
12:15 PM	27	45
12:35 PM	27	44
12:55 PM	28	43
01:15 PM	29	42
01:35 PM	30	41
01:55 PM	31	40
02:15 PM	33	39
02:55 PM	33	34
03:15 PM	33	33
03:35 PM	33	29
03:55 PM	31	27
04:15 PM	29	25
04:55 PM	28	25
05:15 PM	28	26
05:55 PM	26	28
06:15 PM	26	35
06:55 PM	26	37
07:15 PM	26	40
07:55 PM	25	46
08:15 PM	24	50
08:55 PM	23	59
09:15 PM	23	63
09:55 PM	23	66

Table 9: Analysis Data of Sensor Node 2

6.7 Conclusion

Many problems have been encountered during the implementation. Despite all problems we have tested the prototype system successfully. Finally we conclude that

- Parameters of climatic condition for Agricultural environment i.e. Temperature and Humidity are performing its function as desired.
- Monitoring and Controlling operations are also functioning as desired.
- The same can be installed at any work place for monitoring of parameters, controlling them and displaying them on remote PC through wireless communication.
- By the use of relay controlling circuit at transmitter end any type of actuator which runs on high voltage can be controlled wirelessly.
- A prototype system is tested successfully.

6.8 Future Scope

In the future, the scope of this project can be extended as:

- It can be enhanced by assembling more sensor nodes.
- The platform can be used to implement various sensing applications like environment pollution monitoring during peak traffic hours.
- The host computer can be interfaced with GSM modem and the results, alarm can be sends to GSM subscriber.

References

[1] Million Mafuta, Marco Zennaro, Antoine Bagula, Graham Ault, Harry Gombachika and Timothy Chadza “Successful Deployment of a Wireless Sensor Network for Precision Agriculture in Malawi” International Journal of Distributed Sensor Networks Volume 2013,pp 1-7

[2] Manijeh Keshtgary, Amene Deljoo “An Efficient Wireless Sensor Network for Precision Agriculture” Canadian Journal on Multimedia and Wireless Networks, Vol. 3, No. 1, January 2012

[3] M. Keshtgary and A. Deljoo, “An efficient wireless sensor network for precision agriculture,” Canadian Journal on Multimedia and Wireless Networks, vol. 3, pp. 1–5, 2012.

[4] G.H. E. L. de Lima, L. C. e Silva, P.F. R. Neto, “ WSN as a Tool for Supporting Agriculture in the Precision Irrigation,” sixth International conference of Networking and Services (ICNS), 2010, pp. 137 – 142

[5] Ning Wang, Naiqian Zhang, Maohua Wang, “Wireless sensors in agriculture and food Industry —Recent development and future perspective”, published in Computers and Electronics in Agriculture 50 (2010) 1–14.

[6] R. Beckwith, D. Teibel, and P. Bowen, "Unwired wine: sensor networks in vineyards," 2009, pp. 561- 564.

[7] I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2008). “Wireless sensor networks: a survey on Computer Networks”, 38, 393-422.

[8] Rajkamal “Embedded Systems: Architecture, Programming and Design” Tata Mc-Graw Hill Education.

[9] Digi XBee Series 1 Datasheet Product Manual v1.xEx - 802.15.4 Protocol For RF Module Part Numbers: XB24-A...-001, XBP24-A...-001

[10] K. Konstantinos , X .Apostolos, K. Panagiotis , S. George,“Topology Optimization in Wireless Sensor Networks for PrecisionAgriculture Applications, ” SENSORCOMM, March.2007.

[11] T. Wark, P.Corke, P.Sikka, L.Klingbeil,"Transforming Agriculture through Pervasive Wireless Sensor Networks,"Pervasive Computing IEEE, vol. 6, no.2, pp. 50-57, 2007

[12] H. Karl and A. Willig, "Protocols and Architectures for Wireless Sensor Networks," ISBN: 0-470-09510-5, 2005

[13] G.H. E. L. de Lima, L. C. e Silva, P.F. R. Neto, " WSN as a Toolfor Supporting Agriculture in the Precision Irrigation," sixthInternational conference of Networking and Services (ICNS),2010.

[14] A.Baggio," Wireless sensor networks in precision agriculture,"CA: Delft University of Technology, 2009.

[15] W. Cao, G.Xu, E. Yaprak, R. Lockhart, T. Yang and Y. Gao,"Using Wireless Sensor Networking (WSN) to Manage Micro-Climate in Greenhouse", MESA , Oct. 2008

[16] W. Zhang, G. Kantor, and S. Singh."Integrated wireless sensor/actuator networks in agricultural applications". In Second ACM International Conference on Embedded Networked SensorSystems (SenSys), pp. 317, 2004, doi:10.1145/1031495.1031560.man, June 14–19.

[17] R. W. Wall and B. A. King, "Incorporating plug and playtechnology into measurement and control systems for irrigationmanagement," presented at the ASAE/CSAE Annu. Int. Meeting,Ottawa, Canada, pp. 042189 ,Aug. 2004.

[18] S. Yi, "The Short-distance Wireless Communication and NetworkTechnology, " Washington D.C, UAS, Academic , vol. 2, 2009.

[19] Y. Xijun, L. Limei, X. Lizhong , " The Application of Wireless Sensor Network In the Irrigation Area Automatic System, International Conference on Network Security, Wireless Communications and Trusted Computing (NSWCTC), vol.1, pp.21-24, Dec.2009.

[20] "Understanding Humidity Control in Greenhouse," Fact Sheet no. 400-5, BC Ministry of Agriculture, Fisheries and Food, 1994.

[21] H. Liu, Z. Meng, and S. Cui, "A WirelessSensor Network Prototype for Environmental Monitoring in Greenhouse,"Proc. Int'l Conf. Wireless Comm. Networking and Mobi le Computing, WiCOM, 2007, pp. 2344–2347.

[22] K . Sohraby, D. Minoli, and T. Znati,Wireless Sensor Networks Technology,Protocols and Applications, Wiley Interscience, 2007.

[23] J. Burel, T. Brooke, and R. Beckwith, "Vineyard Computing: Sensor Networksin Agricultural Production," IEEE PervasiveComputing, vol. 3, no. 1, 2004, pp. 38–45.

[24] S. Shanmuganthan, A. Ghobakhlou, and P. Sallis, “Sensor Data Acquisition for Climate Change Modeling,” WSEAS Trans.Circuit and Systems, vol. 7, no. 11, 2008, pp. 942–952.

[25] K. Gottschalk, L. Nagy, and I. Farkas,“Improved Climate Control for PotatoStores by Fuzzy Controllers,” Computersand Electronics in Agriculture, vol. 40, nos. 1–3, 2003, pp. 127–140.

[26] S. Chaudhary, V. Sorathia, and Z. Laliwala, "Architecture of sensor based agricultural information system for effective planning of farmactivities," In Proceedings of IEEE International Conference on Services Computing (SCC 2004), pp. 93-100, September 2004.

[27] R. Beckwith, D. Teibel, and P. Bowen, "Report from the field: resultsfrom an agricultural wireless sensor network," In Proceedings of 29th Annual IEEE International Conference on Local Computer Networks, pp. 471-478, November 2004

[28] Albrecht, C.; Hagenau, R.; Doring, A.; , "Cooperative softwaremultithreading to enhance utilization of embedded processors for network applications," In Proceedings of 12th Euromicro Conference on Parallel, Distributed and Network-Based Processing, 300-307, February 2004.

[29] I. A. Aziz, M. H. Hasan, M. J. Ismail, M. Mehat, and N. S. Haron, “Remote monitoring in agricultural greenhouse using wireless sensor and short message service (SMS),” International Journal of Engineering & Technology IJET Vol: 9 No: 9

[30] A. D, S. Roy, and S. Bandyopadhyay, “Agro-sense: precision agriculture using sensor-based wireless mesh networks,” First ITU-T Kaleidoscope Academic Conference.

[31] J. S. Lin, and C. Liu, "A monitoring system based on wireless sensor network and an SoC platform in precision agriculture," 11th IEEE International Conference on Communication Technology Proceedings, 2008.

[32] R.F.M. Inc., “Wireless temperature sensor, application note AN80201, 2009,” http://www.RFM.com/.

[33] T. Chi, M. Chen, and Q. Gao, “Implementation and study of a greenhouse environment surveillance system based on wireless sensor network,” The 2008 International Conference on Embedded Software and Systems Symposia (ICESS2008).

[34] Andrade-Sanchez, P.; Pierce, FJ.; Elliot, T.V perfonnance assessment of wireless sensor networks in agricultural settings. In 2007 ASAB E Annual International Meeting Minneapolis, MN, USA, 2007.

[35] Muhamad Azman Miskam, Azwan bin Nasirudin, Inzarulfaisham Abd. Rahim, "Preliminary Design on the Development of Wireless Sensor Network for Paddy Rice Cropping Monitoring Application in Malaysia", European Journal of Scientific Research , ISSN 1450-216X Vol.37 NO.4 (2009), pp.649-61O

[36] Mahir Dursun and Semih Ozden," A wireless application of drip irrigation automation supported by soil moisture sensors", Academic journals, Scientific Research and Essays Vol. 6(7), 4 April, 2011, pp. 1573-1582