Sudan University of Science and Technology

College of Graduate Studies

Wireless Embedded System for Anti-Theft and Anti-Forget for Personal Belonging Using Sensor Network

نظام الحماية من السرقة والنسيان للمقتنيات الشخصية اللاسلكي باستخدام الانظمة المدمجة وشبكة الحساسات

A Thesis submitted in Partial Fulfillment of Requirements for

The Degree of Master in Electrical Engineering

(Control and Microprocessor)

Presented By:

NADEEN YASSEN JAFER OSMAN

Supervised by:

Dr. OMER ABDEL RAZAG SHARIF ABUBAKER

March 2015

الآية

سورة البقرة الآية (254-255)

ACKNOWLEDGMENT

Praise be to Allah for his mercy and guidance with strength to complete this thesis. I would like to express the thanks to my supervisor, Dr. Omer Abdel Razag Sharif Abubaker for his support, guidance and advices. I will never forget his kindness in helping me to finish my project. I would like to express my sincere thanks to the College of Engineering, AL-Neeline University for providing some of the most important components of the circuit. Especial thanks to Eng. Walide Babeker for his technical assistant and facilitation. I would like to express my sincere thanks to the Industrial Research and Consultancy Centre (IRCC), Sudan for their good hosting during the research period. There are a lot of people would be thanked, hence grateful acknowledgement to my brother Eng. Albeder Yaseen ,Eng .Mohamed Awad and Eng .Osman Abdellah for their support, help, kindness and true cooperation. A special thank can be extended to my friends and colleagues for their support in all aspects of life. Finally, I would like to thanks anyone who helped me take the high step in life.

DEDICATION

To my lovely mother, and my light sight father,

To my beautiful angles sisters, and to my beautiful angles brothers,

To the researchers in this field,

To my all friends.

ABSTRACT

In recent years, the lost of personnel properties such as vehicle, laptops, briefcase and smart mobiles were increasing at an alarming rate around the world. People have started to use the monitoring and theft control systems installed in their properties based on Global System Mobile (GSM) and/or Global Positioning System (GPS). Unfortunately, the cellular network and GPS are not available in all places throughout the country such as forests, deserts and uninhabited areas, and inside the tunnels as well as the high power consumption and installation cost. Therefore, the wireless sensor network techniques were proposed to overcome theses shortages. The main aim in this thesis is to design and realize a simple, low cost and low power consumption anti-theft and anti-forget system for personal belongings using an inbuilt microcontroller and wireless sensor network means. The proposed system was designed, simulated and implemented. The obtained results show an acceptable performance level, even the estimated distance from the received signal strength was crucially affected by the surrounding environment.

المستخلص

فـي الأعوام الأخيـرة تزايـدت حـالات التبليـغ عن فقدان المقتنيـات الشخصـية حـول العـالم مثـل السـيارات ,الحواسيب المحمولـة , الحقائـب وأجهزة الـهواتـف الذكيـة فظـهرت العديـد مـن أنظمـة المراقبـة والتحكم عن بعد معتمـدة علـى نظـام الـهاتـف النقـال العـالمي أو/و نظـام تحديـد المواقـع العـالمي . للأسـف فـإن شبكة الـهاتـف النقـال او نظام تحديد المواقع العـالمي قد لا تكون متاحـة فـي كل الأمـاكن مثـل الغابـات والصـحراء والأماكن الغير مأهولـة او داخل الأنفاق عوضـا عن الطاقة المستهلكة الكبيرة والتكلفة العالية. لـهذا تكمن الحوجة إلى تقنية لاسلكية أقل كلفة تشغيل وتركيب مثل شبكة الحساسات اللاسلكية لحل هذه المشاكل. الـهدف الرئيسي من هذا البحث هو تصميم وتنفيذ نظام بسيط وغير مكلف ذو طاقة مستهلكة قليلـة ضـد سرقة الحقائب والأمتعة باستخدام متحكم دقيق مدمج ووسيلة تحكم لاسلكية. تم تصـميم الـدوائر الخاصـة بالنظام ومحاكاتها و تنفيذها في نموذج للحقيبة حيث كانت النتائج مرضية لتقدير المسافة للحقيبة المفقودة تبعا للإشارة الواردة من أجهزة شبكة الحساسات اللاسلكية بالرغم من تأثرها بالبيئة المحيطة.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AC	Alternative Current
ADC	Analog to Digital Converter
API	Application Programming Interface
AT	Attention
ATmega	Atmel mega
ATtiny	Atmel tiny
AVR	Advanced Virtual RISC
BASCOM	Basic Compiler
dBm	Decibel –milliwatt
DC	Direct Current
DTMF	Dual Tone Multi Frequency
EEPROM	Electrically Erasable Programmable Read Only Memory
ENES	Electrical and Nuclear Engineering School
FFD	Full Function Device
GHz	Giga Hertz
GND	Ground
GPS	Global Positioning System
GSM	Global System of Mobile
GTS	Guaranteed Time Slot
GUI	Graphic User Interface
I/O	Input /Output
ICSP	In- Circuit Serial Programming
ID	Identification Data
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers

IRCC	Industrial Research and Consultancy Centre
ISM	Industrial, Scientific & Medical
ISO	International Organization for Standardization
JMF	Java Media Framework
KB	Kilo Byte
LBS	Location-based service
LED	Light Emitted Diode
Li-Ion	Lithium-Ion
m	meter
MAC	Medium Access Control
μA	Microampere
mAh	milli Amber /hour
MATLAB	Matrix Laboratory
MHz	Mega Hertz
mW	milli Watt
OSI	Open Systems Interconnection
PHY	physical layer
PWM	Pulse Width Modulated
RF	Radio Frequency
RFD	Reduced Function Device
RFID	Radio-frequency identification
RSS	Received signal strength
RSSI	Received Signal Strength Indication
SMS	Short Message Service
SRAM	Static Random Access Memory
St.Deviat.	Stander Deviation
UART	Universal A synchronous and an synchronous Receiver

	Transmitter
USB	Universal Serial Bus
V	Volt
WPANs	Wireless Personal Area Networks
WSN	Wireless Sensor Network

LIST OF ABBREVIATIONS

AVR	Advanced Virtual RISC
AC	Alternative Current
ADC	Analog to Digital Converter
API	Application Programming Interface
AT mega	Atmel mega
AT tiny	Atmel tiny
AT	Attention
BASCOM	Basic Compiler
dBm	Decibel –milliwatt
DC	Direct Current
DTMF	Dual Tone Multi Frequency
ENES	Electrical and Nuclear Engineering School
EEPROM	Electrically Erasable Programmable Read Only Memory
FFD	Full Function Device
GHz	Giga Hertz
GPS	Global Positioning System
GSM	Global System of Mobile
GUI	Graphic User Interface
GND	Ground
GTS	Guaranteed Time Slot
ID	Identification Data
ICSP	In- Circuit Serial Programming
IRCC	Industrial Research and Consultancy Centre
ISM	Industrial, Scientific & Medical
I/O	Input /Output

IEEE	Institute of Electrical and Electronics Engineers
IDE	Integrated Development Environment
ISO	International Organization for Standardization
JMF	Java Media Framework
KB	Kilo Byte
LED	Light Emitted Diode
Li-Ion	Lithium-Ion
LBS	Location-based service
MATLAB	Mathematic Laboratory
MAC	Medium Access Control
MHz	Mega Hertz
m	meter
μA	Micro Amber
mAh	milli Amber /hour
mW	milli Watt
OSI	Open Systems Interconnection
PHY	physical layer
PWM	Pulse Width Modulated
RF	Radio Frequency
RFID	Radio-frequency identification
RSS	Received signal strength
RSSI	Received Signal Strength Indication
RFD	Reduced Function Device
SMS	Short Message Service
St.Deviat.	Stander Deviation
SRAM	Static Random Access Memory
UART	Universal A synchronous and an synchronous Receiver

	Transmitter
USB	Universal Serial Bus
V	Volt
WPANs	Wireless Personal Area Networks
WSN	Wireless Sensor Network

CHAPTER ONE

INTRODUCTION

1.1 General Background

In recent years, personnel properties such as vehicle, laptops, briefcase and smart mobiles theft are increasing at an alarming rate around the world. People have started to use the theft control systems installed in their properties especially vehicles. The commercially available anti-theft systems are relatively expensive especially for vehicles. The global system for mobile communications is the most popular standard for mobile phones in the world. Over billion people use GSM service across the world. The usability of the GSM standard makes international roaming very common between mobile phone operators, enabling subscribers to use their phones in many parts of the world. At any given moment, any particular individual can be contacted with the mobile phone. But the application of mobile phone cannot just be restricted to sending messages or starting conversations. New innovations and ideas can be generated from it that can further enhance its capabilities. Moreover, the Short Messaging Service (SMS) can be used in many applications such as remote control and security system and design a tracking systems that effectively allows monitoring for property such as vehicle [1-3], pet animal [4], laptop, and bag.

There are many remote controlled security systems that monitor and remotely control the properties when it is stolen based on GSM and/or global positioning system. Unfortunately, the cellular network is not available in all places throughout the country such as forests deserts and uninhabited areas [5, 6]. Therefore, other wireless techniques should be emerging such as Wireless Sensor Network (WSN) [7].

1.2 Problem Statement

The ability to accurately detect a property's location to save it from lost and theft is the main goal of surveillance systems. These systems can be implemented using several hybrid techniques that include: wireless communication and embedded applications. Considering the disadvantage of the available sophisticated antitheft solution based on GSM and GPS networks, a portable low cost, and simple antitheft and anti-forgot system is needed.

1.3 Objectives

The objective of this thesis is to design and realize a simple, and low cost anti-theft, anti-forget and oblivion property (*e.g.* bag or briefcase) control scheme using an inbuilt microcontroller and wireless sensor network.

1.4 Methodology

To achieve the proposed objective to design an anti-theft and anti-forget system for personal belonging, the following methodology has been followed:

- System analysis: this may include gather the system requirements and its functionality, as well as figure out its components and their interrelation/interaction.
- System design: this may include design, simulate and fine tuning all the required system components (hardware and software) based on the gathered requirements and specifications.
- System realization and testing: this may include implementation, and testing of the designed system components individually and integrally.

1.5 Thesis Layout

The rest of the thesis have organized as follow: chapter two presents the main concepts of monitoring and tracking system for some related applications where some available literature was reviewed. In chapter three, the proposed tracking system was designed, developed and implemented using wireless sensor network technology as a part of an embedded system. Chapter four showed the obtained results and their related comments. The conducted research was concluded in chapter five, where some recommendations for future work was suggested.

CHAPTER TWO

MONITORING AND TRACKING SYSTEM OF BELONGINGS

2.1 Introduction

The concept of tracking goods, people and animals has long been used by humans as a method to control their surroundings. Early tracking devices such as the astrolabe and compasses were used to help people track the movement of heavenly bodies. Modern technology has given people the ability to track nearly anything, provided it either contains a device or is in certain proximity. In this chapter, the most important techniques used in the field of surveillance and tracking were briefly highlighted.

2.2 Previous Works

In the following will be discussed types of tracking devices that presented in reviewed literature.

2.2.1 GPS and GSM

The global positioning system uses a series of roughly 32 satellites located in medium Earth orbit to triangulate the position of an object. While it was developed by the United State air force, it was put into civilian use in the early 21st century. Since then been integrated into this system in many personal belongings in order to control and determine their exact location has proved highly efficient not only in location but also became possible to control the system to be monitored by integrating the control system which is composed of microcontroller operates in harmony with the monitoring unit as in [2,4] .

Ramadan *et al.* have proposed control vehicle tracking system in order to protect it from theft. The system works to protect the vehicle not only alerting the owner but also the possibility of tracking the location and control to stop it by remotely by an SMS sent by the owner to the security unit in the vehicle to stop after system check the console from the sender's identity matching the sender's phone number with the phone number of the owner of the car tastiest, until locate it completely by GPS built inside and displayed in Google Earth. Start monitoring unit to work as soon as the vehicle is moving, where you send an alert message to the owner of the car, which in turn send a text message to the console compact vehicle engine to stop working and after verifying the occurrence of an attempt to steal his car [2].

Abukovich *et al*. have proposed a wireless animal containment system that uses voice commands instead of shock to contain an animal in a defined area. The microcontroller-based system has the capability of sending tracking information to the user's handheld device using GPS. The designed system is composed of three main components: the hand held device, the remote tracking device, and the GPS receiver module. The hand held device and the remote device are communicating with each other to exchange the pet's location and alerting the user using TI MSP430 ezRF2500, CC2500 and Wi2Wi W2SG0004 for GPS. The integrated GPS on the pet's collar was designed to send a signal to a receiver displaying its position to the user to overcome problem that the pet may not listen to the command without the physical reinforcement [4].

In the process of developing some of the previous techniques, and to improve the efficiency of these systems, short message service has been added to the possibility of remote control of the system [1, 3, 5, and 6].

Sadagopan, Francis and Rajendran have proposed an anti-theft control system design using embedded system for automobiles using GSM-SMS services. The vehicle's owner use password to operate it, and if the password matches then the owner has to insert the key to start the vehicle and if fails automatically alerts the by using GSM modem exact message will be given to the owner of the vehicle.

The system can be transmitting the position to the owner on his mobile phone as a SMS at his request [1].

Dhotre *et al*. have designed and developed an advanced microcontroller-based vehicle locking system in the real time environment using GPS and GSM. The embedded system in the vehicle send an SMS to the owner, followed by call on owners mobile when the owner did not acknowledge the SMS, lastly system will turn off the ignition of the vehicle. The proposed system has three different types of theft detection sensors namely, rider sensor, vibration sensor and battery removal sensor when the targeted devices can be controlled either by sending Dual Tone Multi Frequency (DTMF) or by the voice messages of the owner. More convenient feature can be added to the system is use of satellite modems instead of cell phones as tracking device as the system may fail when there is no network coverage [3].

In [5], a GSM and GPS based vehicle location and tracking system was proposed. In this system AT89S52 microcontroller is used for interfacing to various hardware peripherals. The design is an embedded application, which will continuously monitor a moving vehicle and report the status of the vehicle on demand. This design can use in other applications like: It is used for food delivery and car rental companies.

Al-Khedher, has proposed design a hybrid GPS-GSM localization of Automobile tracking system. In this system, a real-time automobile tracking system via Google Earth is presented. The system included two main components: a transmitting embedded module to interface in-vehicle GPS and GSM devices in order determine and send automobile location and status information via SMS. The second stationary module is a receiving module to collect and process the transmitted information to a compatible format with Google Earth to remotely monitor the automobile location and status online. The transmitted location of the vehicle has been filtered using Kalman filter to achieve accurate tracking [6].

2.2.2 Radio frequency identification

Radio Frequency Identification (RFID) tags is one of the easiest and smallest devices that can be installed in nearly any component to enable tracking. The basic idea is to have each personal belonging equipped with a RFID tag. Recently, the smart phone acts as a RFID reader and keeps track of the presence of the objects. The purpose of the use of this technology not only to property from theft, but the system can be used as an anti-forget system. The tracking system starts to work as soon leave the person's home are automatic so that the phone work a survey on going to verify the existence of the property to be monitored within the allowable range presence in it and once owner exit a property from the scope of surveillance due to the move away owner about in the event of theft or forget the phone sounds an alert owner. There are different types of identification system are present for the detection of animals, students, products and also for transportation such that use RFID reader and passive RFID chips. A single reader can identify many no of chips in very short period of time. So, we can use these properties of RFID reader and tag to monitor the student [8, 9]. Tushar *et al*. [8] have proposed an online student monitoring system using passive RFID. Reader is located on fixed location sends signal to passive RFID chip detected in range of reader. Chip re-transmits the acknowledgement signal with its unique identifier code, hence chip is identified. Also, a single reader can identify many no of chips in very short period of time. So, these properties of RFID reader and tag were used to monitor the student.

2.2.3 Location based service

Location Based Service (LBS) is the type of tracking device used to make a person's cell phone reveal a location. Similar to GPS, this system can be installed in nearly any communications device or vehicle.

2.2.4 Inertia

Inertial tracking devices are mechanical in nature and use gyroscopes to detect a change in orientation. These are fast and accurate, but limited in the distance in which they can identify movement.

2.2.5 Received signal strength indication

Received Signal Strength Indication (RSSI) is a tracking system that uses real time locating technology to identify the position of an object or person. RSSI features wireless technology that can be installed into name badges or items [10]. Benkic *et al*. [10] have proposed a design based on RSSI value for distance estimation in WSN based on ZigBee where the received power should be a function of distance where the RSSI can be used for evaluating distances between nodes.

2.3 IEEE 802.15.4 Protocol

Institute of Electrical and Electronics Engineers 802.15.4 (IEEE 802.15.4) is standardization by IEEE 802.15.4 (Working group IEEE 802.15 Task Group 4 (TG4)), focused on the bottom two layers of International Organization for Standardization/Open Systems Interconnection (ISO/OSI) protocol stack, namely physical layer and Data Link layer, also called the Medium Access Control (MAC) layer. The other layers are normally specified by industrial consortia. See Figure 2.1. IEEE 802.15.4 wireless technology is a short-range communication system for applications with relaxed throughput and latency requirements in Wireless Personal Area Networks (WPANs). The key features of IEEE 802.15.4 wireless technology are low complexity, low cost, low power consumption and low data rate transmissions, supported by cheap fixed or moving devices. The main field of application of this technology is the implementation of WSNs [11].

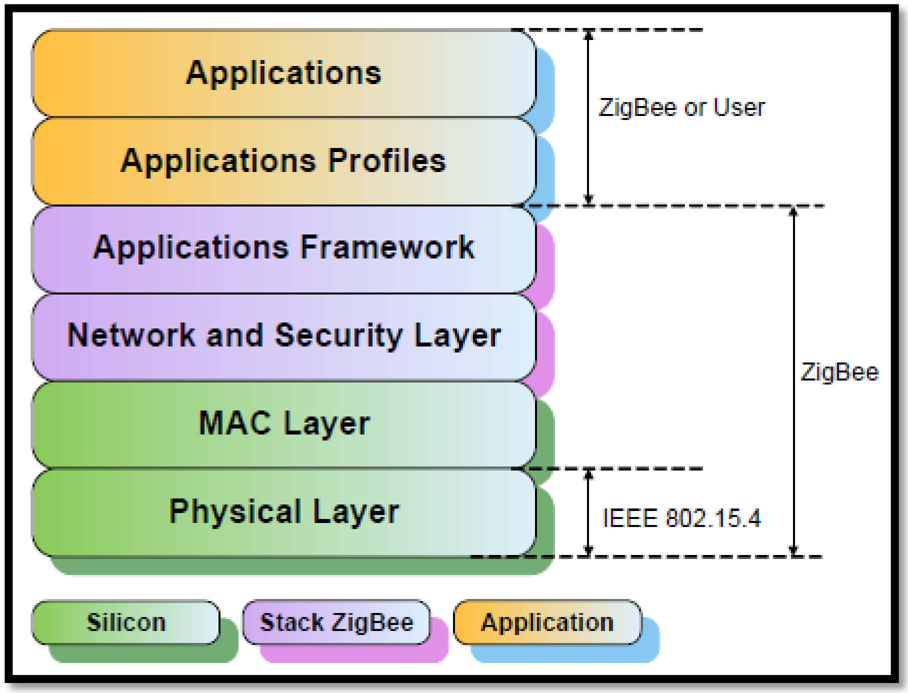


Figure 2.1: IEEE 802.15.4 and ZigBee reference model

2.3.1 ZigBee protocol

ZigBee is an industrial standard to cope with the restrictions of WSN. ZigBee alliance offers a standard based on very low-cost, low power consumption and two-way wireless communications. The name of ZigBee comes from the domestic honeybee which uses a zig-zag type of dance to communicate with other hive members [12]. ZigBee is a speciation of a joint of high level wireless communication protocols based on the wireless Personal Area Network (PAN) standard IEEE 802.15.4. Its goal is the applications that require reliable communications, due to mesh topology, with low data transmission rate and long live batteries.

ZigBee can be used in several types of applications as automation and security control, control of end devices as mouse or keyboards, remote control of electronic devices, monitoring patients or elder lies and in tracking system. But the main and most successful is home automation. The ZigBee operates in the Industrial, Scientific and Medical (ISM) bands: it can operate globally in the 2.4 GHz frequency. Figure 2.2 show the main application of ZigBee.

Figure 2.2: ZigBee main applications

The characteristics that make it so suitable for these purposes are: low-cost and low power consumption. However, three different ZigBee devices can be defined according to its role in the network:

- Coordinator: ZigBee networks always have a single coordinator device for forming the network, handing out the address, securing the network and keeping it healthy.
- Router: A router is a full-featured ZigBee node. It can join existing networks, send information, and receive information and route information. Routers are typically plugged into an electrical outlet because they must be turned on all the time.
- End device: An end device always needs a router or coordinator to be their parent device. The parent helps end devices join the network, and stores messages for them when they sleep.

ZigBee networks can connect together in several different layouts or topologies to give the network its structure. These topologies indicate how the radios are logically connected to each other. Their physical arrangement, of course, is different. There are four major ZigBee topologies pair, star, mesh and cluster tree as show in Figure 2.3.

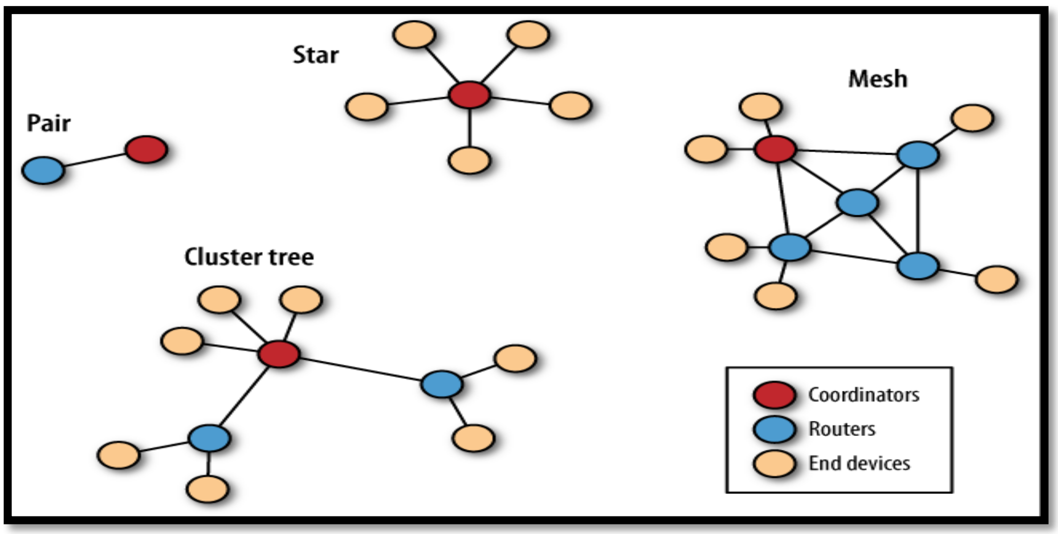


Figure 2.3: ZigBee pair, star, mesh and cluster tree topologies

The technologies to be considered for tracking and monitoring system would be ZigBee (based on IEEE 802.15.4), Bluetooth (based on IEEE 802.15.1) and WiFi (based on IEEE 802.11). Other Radio Frequency (RF) technologies are available but only these three have the technical maturity for this kind of systems and low cost required for consumer products. Table 2.1 shows the main characteristics of each technology.

Table 2.1: Main comparison between Bluetooth, ZigBee and WiFi

	ZigBee	Wi-Fi	Blutooth
Range	10-1000 meters	50-100 meters	10-100 meters
Networking Topology	Ad-hoc, Peer to Peer, Start, or Mesh	Point to Hub	Ad-hoc, Very Small Networks
Operating Frequency	868 MHz (Europe)900-928 MHz(NA),2.4 GHz(Worldwide)	2.4 and 5 GHz	2.4GHz
Complexity (Device & Application Impact)	Low	High	High
Power Consumption (Battery Option and Life)	Very low (Low Power is a Design Goal)	High	Medium
Typical Application	Industrial Control and Monitoring Sensor Network, Home Control and Automation	Wireless LAN Connectivity, Broadband Internet Access	Wireless Connectivity Between Devices such as Phones

2.3.2 XBee

XBee is a brand of radio that supports a variety of communication protocols. XBee is a feature-rich RF module which makes it a very good solution for WSN designers; the implemented protocols on the modules like IEEE 802.15.4 and ZigBee can significantly reduce the work by the programmer for ensuring data communication. Besides the capability of these modules to communicate with microcontroller through UART serial communication, it also has additional pins which can serve for XBee standalone applications. For example, a router node can be built without the need for a microcontroller. The XBee also has a 10-bit Pulse Width Modulated (PWM) output may be sent to another XBee. One important feature is line passing where a digital input on one XBee can be reflected on the digital output of another, thus controlling the output of the second XBee.

In this thesis the conducted experiments were based on the ZigBee protocol which is considered as one of the most common protocols in personal WSN with low bandwidth, low cost, high level of security and low power consumption. Moreover, Digi’s international Inc. Xbee modules were used with ZigBee protocols with variant coverage range up to 1000 m and the ability to use the same hardware for all node types (coordinator, router, and end node). However, each RF data packet sent over-the-air contains a source address and destination address field in its header. XBee modules have a unique and permanent address. This address is a 64-bit serial number assigned by the manufacturer. The XBee modules also have a 16-bit short address assigned within the network. Finally, Node Identifier (NI) can be assigned to each module as a string of text. Table 2.2 shows the type of addresses in ZigBee network.

Table 2.2: Type of address in ZigBee network

Type	Example	Unique
64-bit	0013A200403E0750	Yes, on the Earth
16-bit	23F7	Yes, Only Within a network
Node Identifier	Node1	Uniqueness not Guaranteed

In addition, the PAN address which is another 16-bit address relevant to the personal area network can be recognized in ZigBee network. The16-bit PAN address gives the possibility of 65,536 different PAN addresses along with 65,536 (i.e. 16-bit) module address for each PAN. This addressing scheme gives the capability for more than 4 billion total radios [13]. However, when all the addressing is correct in the network, the communication still may not be established unless the modules work on the same frequency. Thus, all modules in the network must use the same channel. Generally 12 channels, or more based on the type of radio, are available, but the programmer doesn't have to worry about selecting a specific channel since the XBee modules select it automatically. In peer-to-peer network, if any pair of XBee modules wishes to communicate in the same area where a network already exists, they have to choose a different address, PAN ID or frequency channels [14].

In addition, XBee has five modes of operation; the basic one is idle mode. This is the mode when the module is not receiving, transmitting, commanding or sleeping. This mode can consume a significant amount of energy compared to active mode where data is being sent or received. One solution is to put the XBee in sleep mode. However, sleep mode depends on the type of application; whether it is event-driven or periodic application. Moreover, sleeping requires a light scheduling algorithm in order to wake up the nodes in the network in harmony.

An Application Programming Interface (API) is simply a set of standard interfaces created to allow one software program to interact with another. They are generally not designed for direct human interaction. In API mode, the programmer packages the data with necessary information, such as destination address, type of packet, and checksum value. Also, the receiving node accepts the data with information such as source address, type of packet, signal strength, and checksum value. The advantages are the user can build a packet that includes important data, such as destination address, and that the receiving node can pull from the packet information such as source address of the data. While more programming intensive, API mode allows the user greater flexibility and increased reliability in some cases.

2.4 The Received Signal Strength Indicator

Received Signal Strength (RSS) is measured from each received frame as average value measured during the first eight symbols (preamble) and converted to the RSS Indicator (RSSI). This conversion is variable for every type of radio chip. With properly modeled path loss in a given environment, it should be possible to formulate a relation between distance and RSSI value. Problems arise with dynamical changes in environment because of radio signal unstable properties. However, there are many radio propagation models known for wireless communications that predict signal-strength loss with distance - path loss. There are three models widely used for wireless sensor networks:

- Free space propagation model (Friis' Equation) is built on the assumption that the transmitter and receiver are in the line of sight, and there are no obstacles between them.

- Two-ray ground model adds reflection to the previous model. A two-ray ground model receiver receives two rays: direct communication ray and reflected ray.
- Log-distance model (shadowing model) has been derived from analytical and empirical methods. This model comprises two parts:

i. Attenuation model: let d_0 be the reference distance and *P (d0)* the received power at d_0. The following formula can be used to calculate the received power at an arbitrary distance *d*:

$$\frac{P(d_0)}{P(d)} = \left(\frac{d_0}{d}\right)^n \qquad (2.1)$$

Where P (d) is the RSS in distance d [mW], P (d_0) is RSS in the reference distance d_0 [mW], n is path loss exponent [−], d is the distance between transmitter and receiver [m], and d_0 is the reference distance [m]. The attenuation factor n is determined empirically.

ii. Changes in power level at a fixed distance: observation has revealed that a radio wave power's varies through time even when the measurements are taken at the same location. A log-normal probability law is typically used to represent these changes. Considering these two aspects, the complete model is written as:

$$\left[\frac{P(d_0)}{P(d)}\right]_{dB} = RSSI_{d_0} - RSSI_d = nlog\left(\frac{d_0}{d}\right) \qquad (2.2)$$

$$RSSI_d = RSSI_{d_0} - nlog\left(\frac{d}{d_0}\right)[dBm] \qquad (2.3)$$

$$d = d_0.10^{\left(\frac{RSSI_d - RSSI_{d_0}}{10n}\right)}[m] \qquad (2.4)$$

In many systems at short distances, the reference distance is d_0 given as 1 m. The ideal relation between the RSSI [dBm] and distance [m] can be shown in Figure

2.4. Parameter $RSSI_{d0}$ is measured at reference distance in every direction from a transmitter and then averaged, where for the modules following the CC2430 ZigBee module standard, which is belonging to Texas Instrument, the value of $RSSI_{d0}$ is about -40dBm. Path loss exponent n expresses how much radio signal strength decreases with distance where in the free space it is equals to 2. These values can be estimated through measurement of radio signal propagation environment or it can be usually defined as constant for indoor or outdoor environment. Although the Shadowing model is more complex, it is the closest to reality as it does not consider the communication range as a circle or sphere but rather as a surface (volume) which shape changes through time, even without moving the source. In the following experiments to evaluate the values of n and the associated $RSSI_{d0}$ were conducted in different environments.

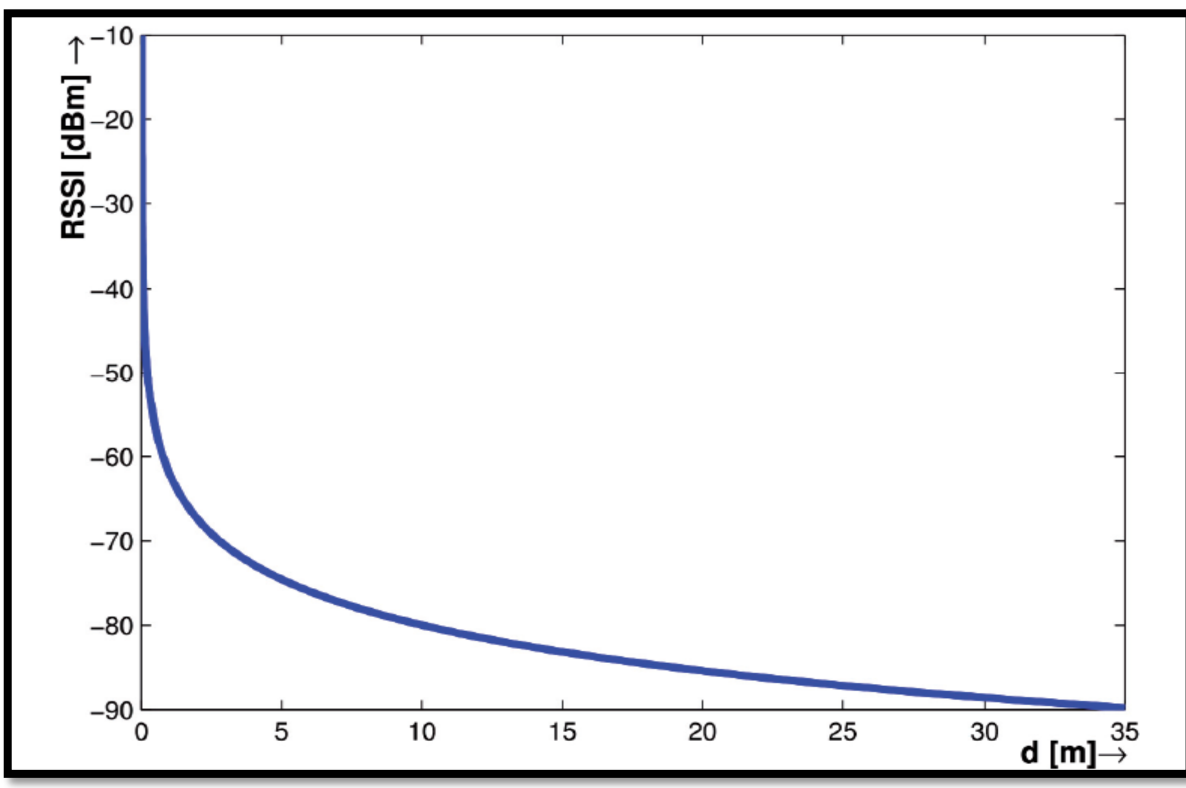


Figure 2.4: Ideal relation between RSSI and distance

CHAPTER THREE

DESIGN AND IMPLEMENTATION OF ANTI THEFT AND ANTI-FORGET SYSTEM

3.1 System Design Overview

The microcontroller-based anti-theft and anti-forget system (smart bag system) consist of processing, communication, power and alarm units as shown in Figure 3.1. For the processing unit, a suitable microcontroller was selected, where a simple buzzer was used to be the alarm unit. The wireless sensor network technology was utilized to fulfill the requirement of the communication unit. A heavy duty rechargeable battery was recommended to power the entire system.
In the following a detailed design, simulation, and implementation for each unit were presented, where the Proteus ISIS environment simulator was utilized.

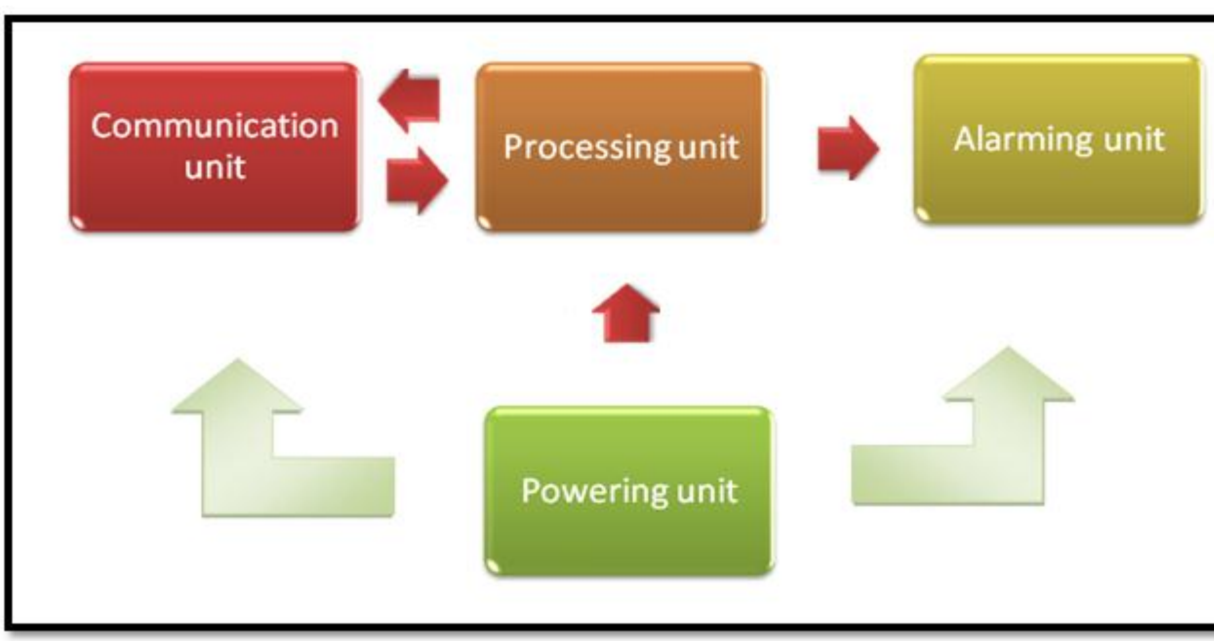


Figure 3.1: The block diagram of the proposed antitheft and anti-forget system

3.2 Processing Unit

An ATtiny2313 Advanced Virtual RISC (AVR) microcontroller was selected for the proposed system due to its low power consumption, availability of an internal oscillator, and availability of an internal Analog to Digital Converter (ADC). The selected ATtiny2313see (appendix G) has 128byte Electrically Erasable Programmable Read Only Memory (EEPROM) and 192 bytes Static Random Access Memory (SRAM) that are very enough to store RSSI values. It is worthy to note that a comparison was made between the Atmel microcontrollers families based on: power consumption, size, internal memory, and available ports and interfaces. Therefore, ATtiny family was selected where its operating voltage is 0.7VDC, and the entire ports and interfaces were meeting the proposed system requirements. Moreover, a comparison was made between the members of the ATtiny microcontroller's family for their availability in the used simulation environment *i.e.* (Basic Compiler) BASCOM and Proteus. For example the BASCOM and Proteus library include ATtiny (12, 13, 15, 22, 24, 25, 26, 44, 45, 84, 85, 461, 861, and 2313) where only the ATtiny2313 has PWM and Universal A synchronous and an synchronous Receiver Transmitter (UART) as can be shown in Figure 3.2, Figure 3.3 and Figure 3.4 show the complete schematic diagram of the proposed transmitter and receiver for anti-theft and anti-forget system, respectively.

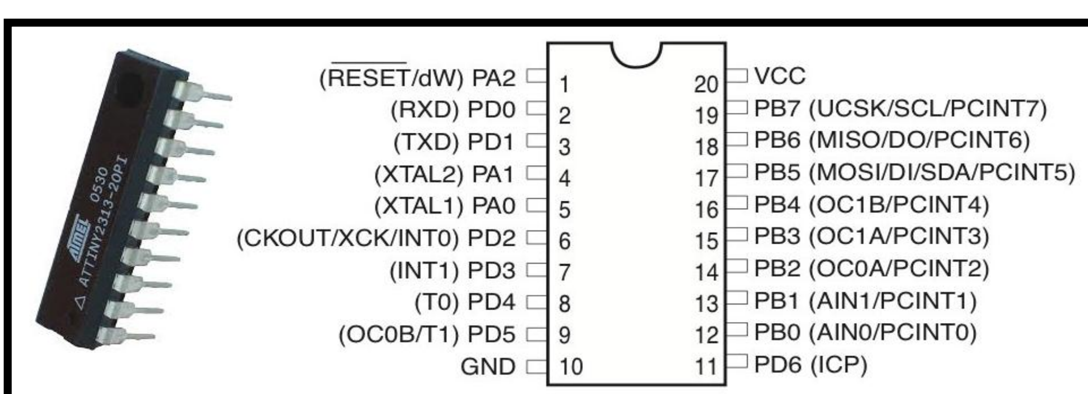


Figure 3.2: The ATtiny 2313 Atmel microcontroller and its pin out

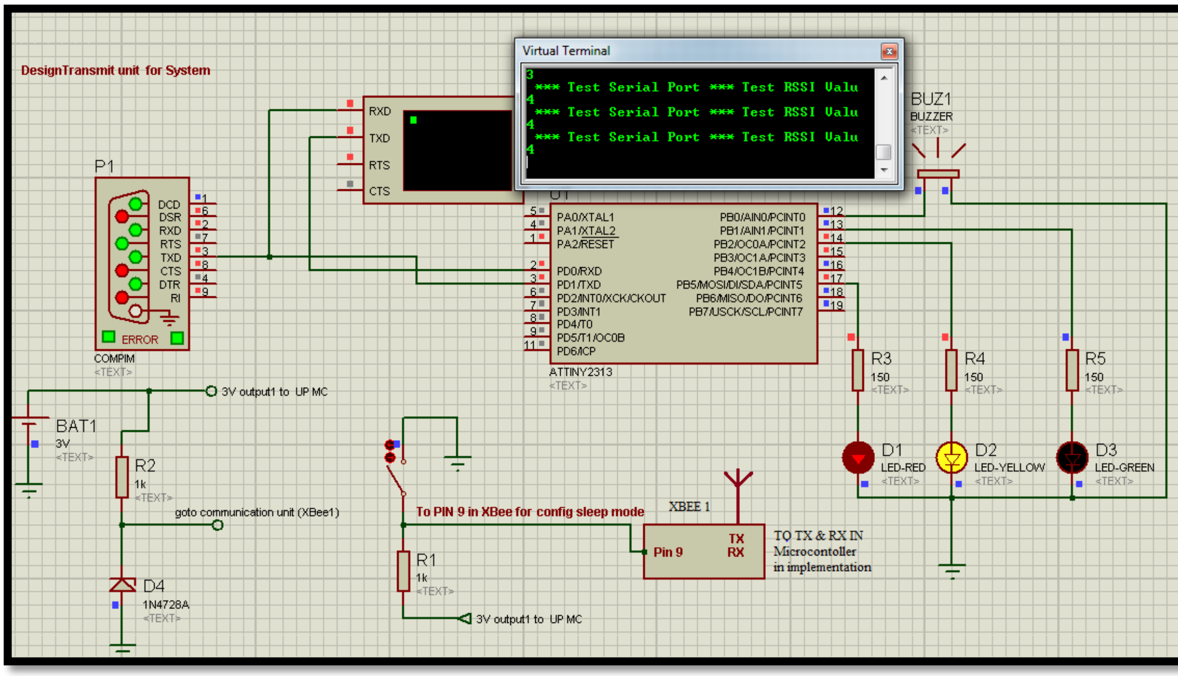


Figure3.3: The complete schematic diagram of the proposed transmitter system

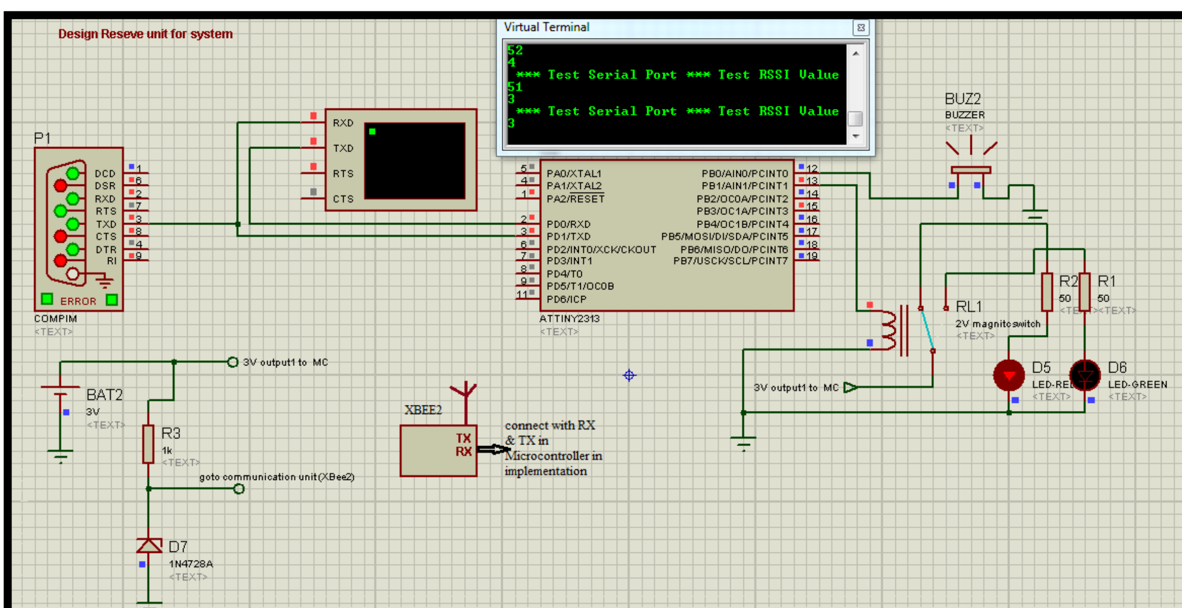


Figure3.4: The complete schematic diagram of the proposed receiver system

3.3 Communication Unit

In the proposed system, the communication unit which based on the WSN *i.e.* XBee (appendix E) is attached to the transmission and the receiver units. The transmission unit is always presenting with the owner (user), while the receiver unit be inside the precious goods e.g. bag that to be monitored. Feed of XBee comes from Arduino board associated with him knowing that the transmitters and receivers are working only when change the owner the status of the work of the XBee from sleep mode to active mode through the existing key that present in transmission system with owner .Figure 3.5 shows the XBee and its pins out description.

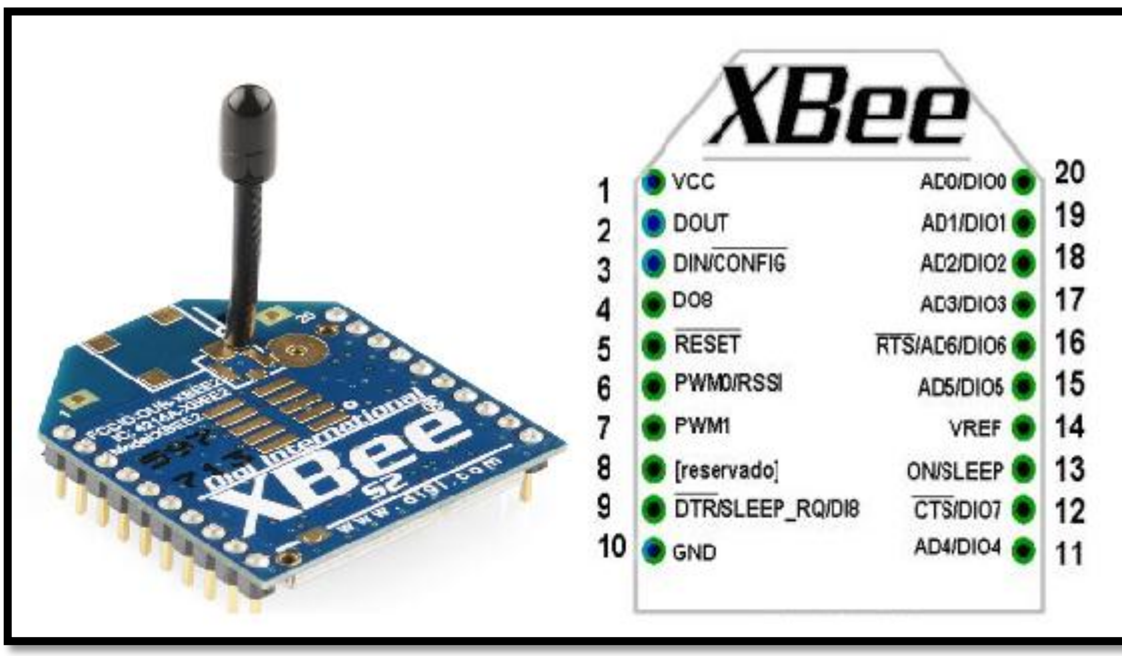


Figure 3.5: The XBee and its pins out description

XBee modules were engineered to meet ZigBee/IEEE 802.15.4 standards and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of critical data between devices. Table 3.1 presents some Xbee module specifications.

Table 3.1: Some specifications of the Xbee module

Radio Frequency Band	ISM 2.4GHz
RF Data Rate	250,000 bps
Indoor/Urban	up to 100’ (30 m)
Outdoor Line-of-Sight	up to 300’ (100 m)
Transmit Power	1 mW (0 dBm)
Receiver Sensitivity	-92 dBm
TX Current	45 mA (@3.3 V)
RX Current	50 mA (@3.3 V)
Power-Down Current	< 10 µA

3.4 Alarm Unit

The alarm unit in the proposed anti-theft and anti-forget system for precious goods can be configured from visual and audio alarming system. The visual alarm consists of a group of Light Emitted Diode (LEDs) allocated at the transmitting side, where the audio alarm, which is consist of a single buzzer, is attached to the both transmitter and receiver. The visual alarm at the transmitter consists of four colored LEDs. A green LED was used to indicate a normal (healthy) powering, booting, and operating system. In addition; green, green-yellow, yellow and red LEDs were configured to indicate safe, risky, recoverable, unrecoverable zones respectively. To interface these LEDs to the microcontroller, resistors with value 300 ohm can be attached serially to each, where 10-20mA can be drown from the microcontroller and the LED voltage is 3V. These resistors, which are marked with R2, R3, R4, R5 as show in Figure 3.6, are LED’s protection resistors.

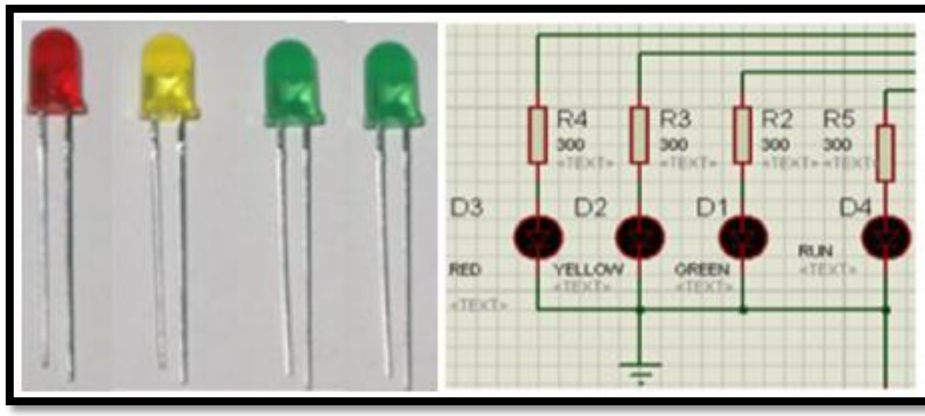


Figure 3.6: The transmitter indicator LEDs and its schematic

However, two buzzers (appendix I) were attached to the transmitter and receiver units to alert the owner and the surrounding people when the precious good has been stolen or forgotten. Figure 3.7 shows the audio alarm that used.

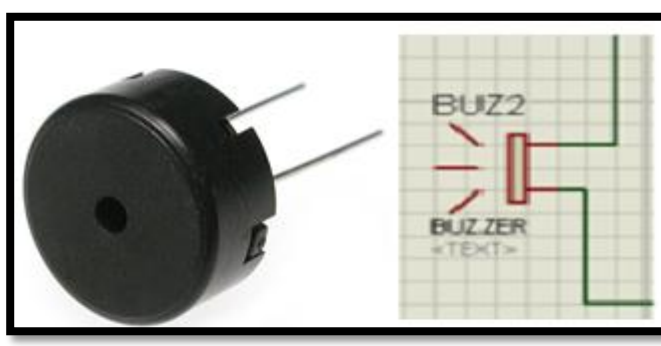


Figure 3.7: The buzzer and its schematic

3.5 The Bag's Lock

To prevent the accessibility of the precious good a simple magnetic lock was proposed, when the owner can monitor the status of the lock through red and green LEDs to indicate open and close lock status. In Figure 3.8, the proposed relay to control the magnetic lock with the indicator LEDs was shown for the simulation purposes.

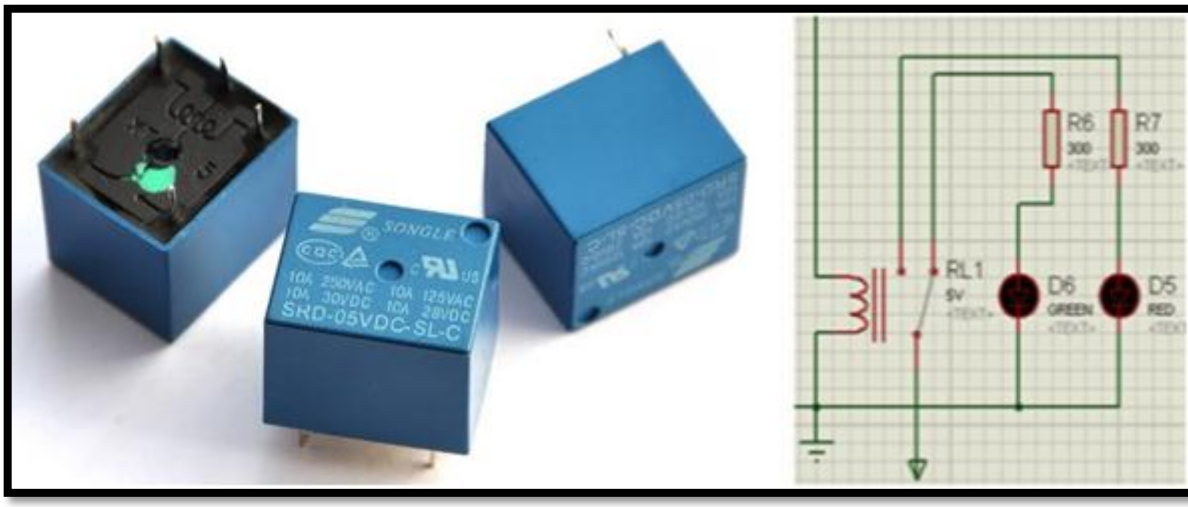


Figure 3.8: The proposed relay and the schematic of the simulation luck circuit

3.6 Power Supply Unit

Table 3.2 represents the power requirements by the proposed system units. Therefore, a Lithium-Ion (Li-Ion) battery, which is a combination of several electrochemical cells used to convert stored chemical energy into electrical energy and vice versa and mostly used in the conventional mobile phones, be the rechargeable portable power supply of transmitter and receiver of the proposed system.

Table 3.2: Power requirement

Elements	Unit Power Consumption	Number of Elements	Total Power Consumption
ZigBee Module	1.5mW	2	3mW
ATtiny Microcontroller	1.4mW	2	2.8mW
Buzzer	175mW	2	350mW
LED	40mW	4	160mW
Resistor	250mW	5	1250mW

These two rechargeable Li-Ion 3.7V- 1020 mAh batteries (*i.e.* Figure 3.9), are serially connected to produce 7.4V with the advantage of high capacity-to-weight ratio, a low self-discharge characteristic and inexpensiveness. Although the proposed system contains elements with different operating voltages, only a Zener diode is used for regulation in the ATtiny-based system, however, the Arduino board is build-in to handle this regulation requirement.

Figure 3.9: Rechargeable 3.7V lithium-ion battery

However, a sleep mode is proposed to the system for power saving and be freely from the monitoring system upon user desired. Figure 3.10 shows the sleep mode switch with the indication lamps and how it is connected to the XBee module.

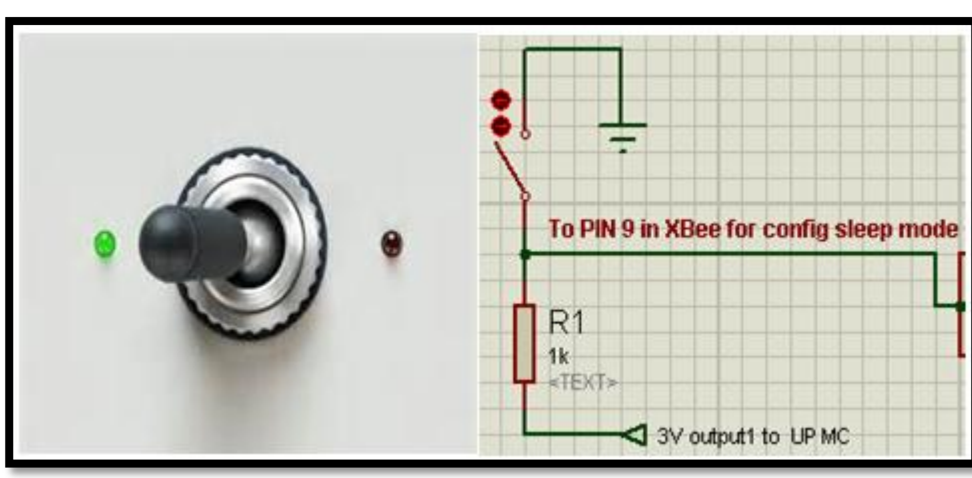


Figure 3.10: The sleep mode switch and its connection to the Xbee module

3.7 Flow Chart of Smart Bag System

Figure 3.11 shows the proposed follow chart of the proposed system the desired goal of the system can be achieved thoroughly. Firstly, the system can verified all its peripherals to ensure smooth operation. A status mode check can be also achieved. Once the system is in a wakeup mode, the values of the RSSI are value recorded and compared with reference values for each zones: safe, risky, recoverable and unrecoverable. The associated alarms can be activated accordingly. These zones (Figure 3.12) can be described as follow:

- The safe zone: where the distance between the bag and its owner is not exceeding a 1 m. In this zone the green LED in the transmitter unit will be on.
- The risky zone: where the distance between the bag and the owner had exceeded the safe zone and still less than 2 m. Therefore, the green and yellow LEDs will turn on concurrently with the buzzer with certain tone frequency (1000, 1000Hz).
- The recoverable region: At this level, the distance between the bag and the owner is between 2-5 m where a yellow LED is turned on and the associated buzzer operates in (500, 900 Hz).
- The unrecoverable zone: in this zone the user belonging be in more than 5 m and where the red LED is turned on and the buzzer be in (1000, 100Hz).

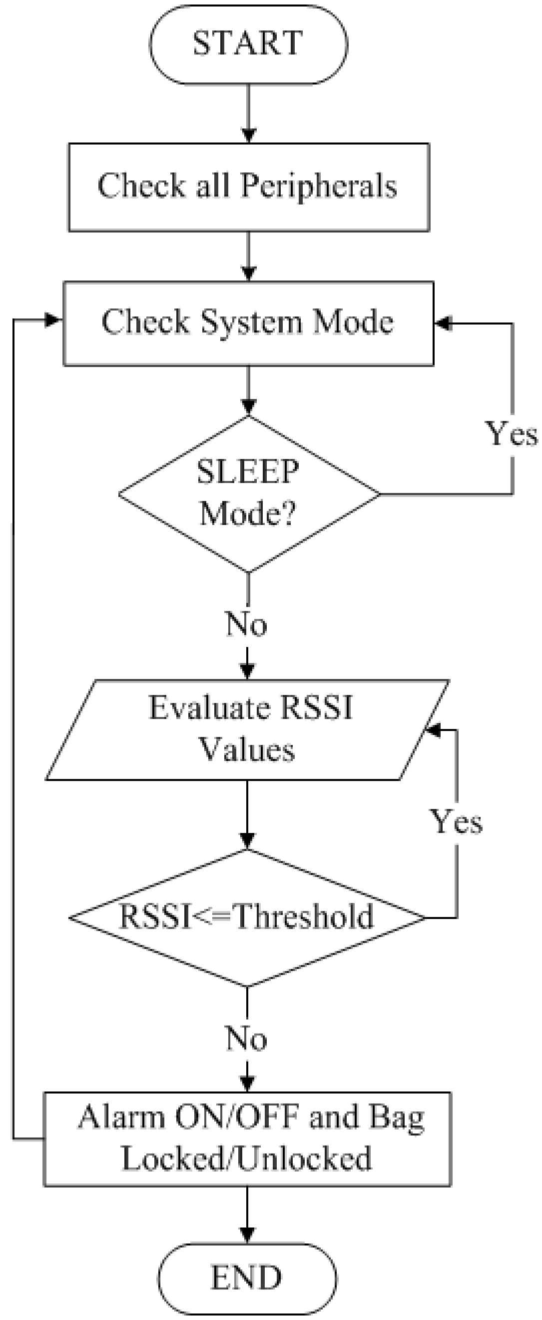


Figure 3.11: Flow chart of smart bag system

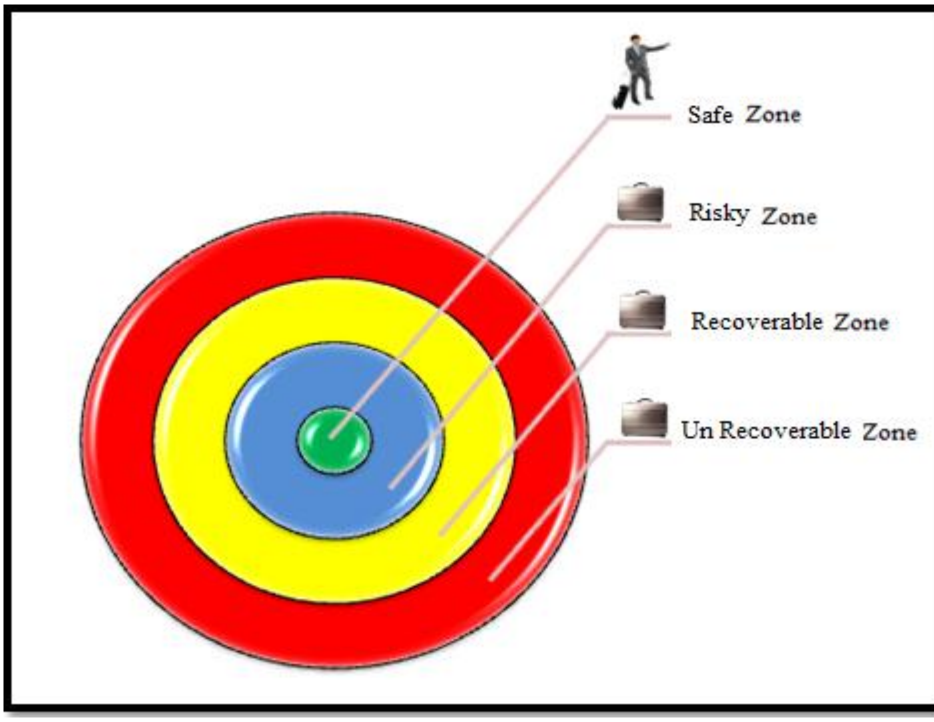


Figure 3.12: The system work areas

3.8 Embedded Software Design

The proposed system can be empowered by embedded software for the microcontroller and for the XBee modules. A BASCOM program was developed for the microcontroller, when XCTU software was utilized for XBee configuration.

3.8.1 XCTU software

The XCTU software is the official configuration program for XBee modules provided by Digi International Inc. The XCTU is available only for Windows operating system; if the user has Linux then XCTU should be running under WINE windows emulator. XCTU is used mainly for setting the firmware to XBee module. Figure 3.13 shows the main configuration screen of XCTU. The XCUT software includes the list of available firmware for various types of XBee module, and modes of the XBee module. For configuration the proposed system a firmware XB24-ZB, which supports the full ZigBee functionality, can be used to set up the module as coordinator, router or end device. Moreover, each function is supported with the ability to work in ATtention (AT) or API mode.

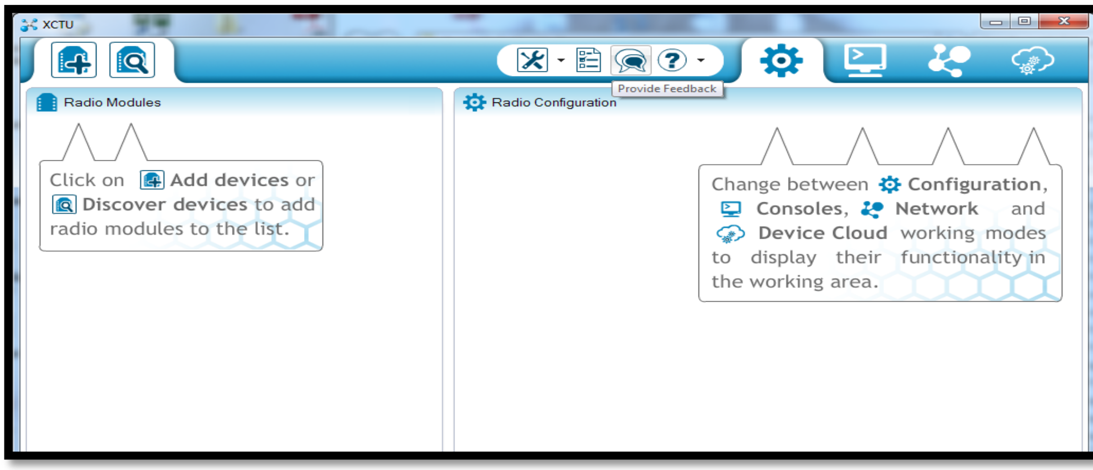


Figure 3.13: The main configuration screen of XCTU

In Figure 3.14, the fully commented setup commands of XCTU with the ability to change the values of these commands were presented. The XCTU has a terminal window which can be used to read the data being received by the connected XBee module. Additionally, this terminal can be used to enter AT command mode and write the setting to the module as shown in Figure 3.15.

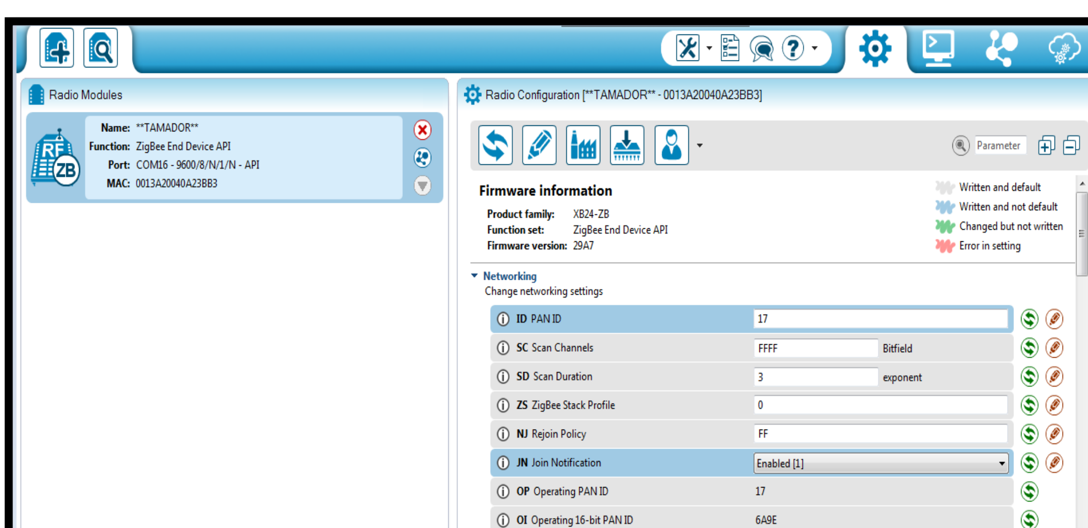


Figure 3.14: Setup window for XCTU

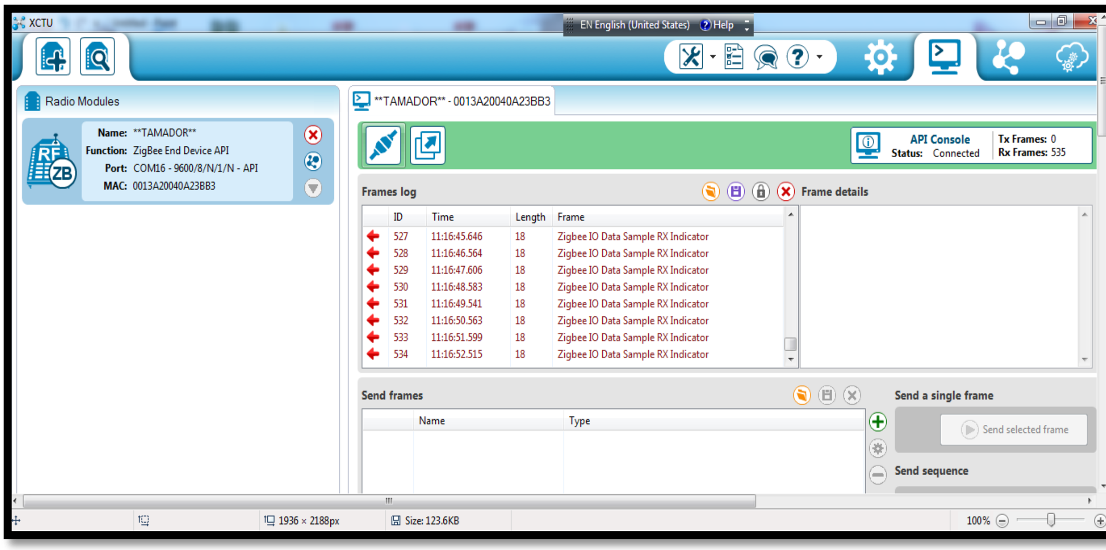


Figure 3.15: The terminal window for XCTU

Moreover, range test is another feature provided by XCTU which was utilized in this thesis. Figure 3.16 shows how the range test is appeared, where it provides the RSSI in dBm where -40 dBm represents the strongest signal received by the module. Range test procedure can be achieved as follows:

- The local XBee module which is connected to PC/laptop sends a packet of data to a remote XBee module.
- The remote XBee module acknowledges the packet back to the local module.
- X-CTU reads the value of the received signal strength by and shows it in dBm as shown in Figure 3.16.

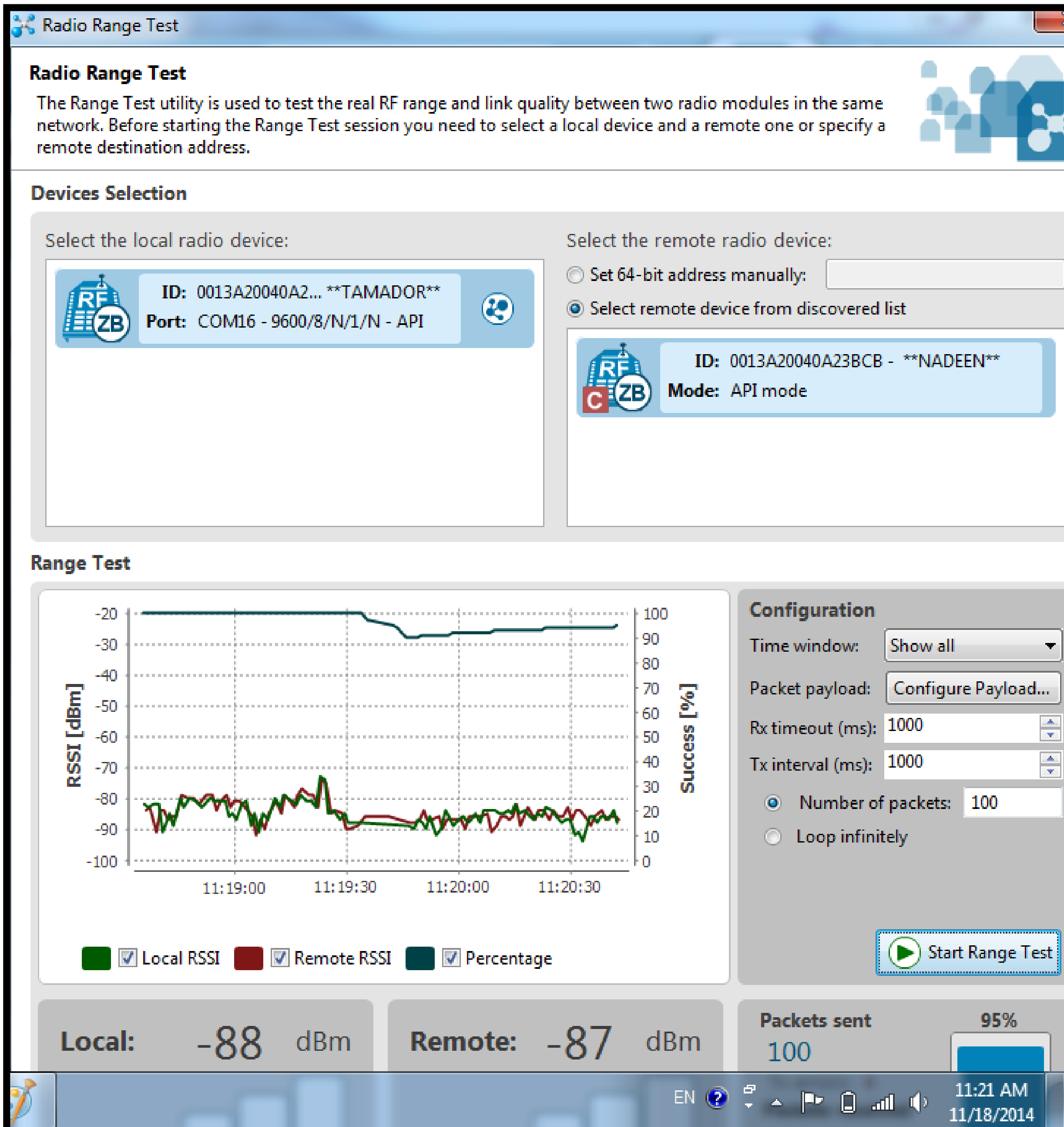


Figure 3.16: Range test screen for XCTU

3.8.2 BASCOM AVR software

The ATtiny2313 microcontroller can be programmed by the BASCOM and it can download it for free directly from the AVR website's. The BASCOM environment, as shown in Figure 3.17 is splitted into three areas. The grey area, which is located at the top of the window, is features toolbar and buttons that control program behavior. The white area in the middle is where the code can be entered and modified. The blue section in the right of the window is where the microcontroller chip pins out can be showed.

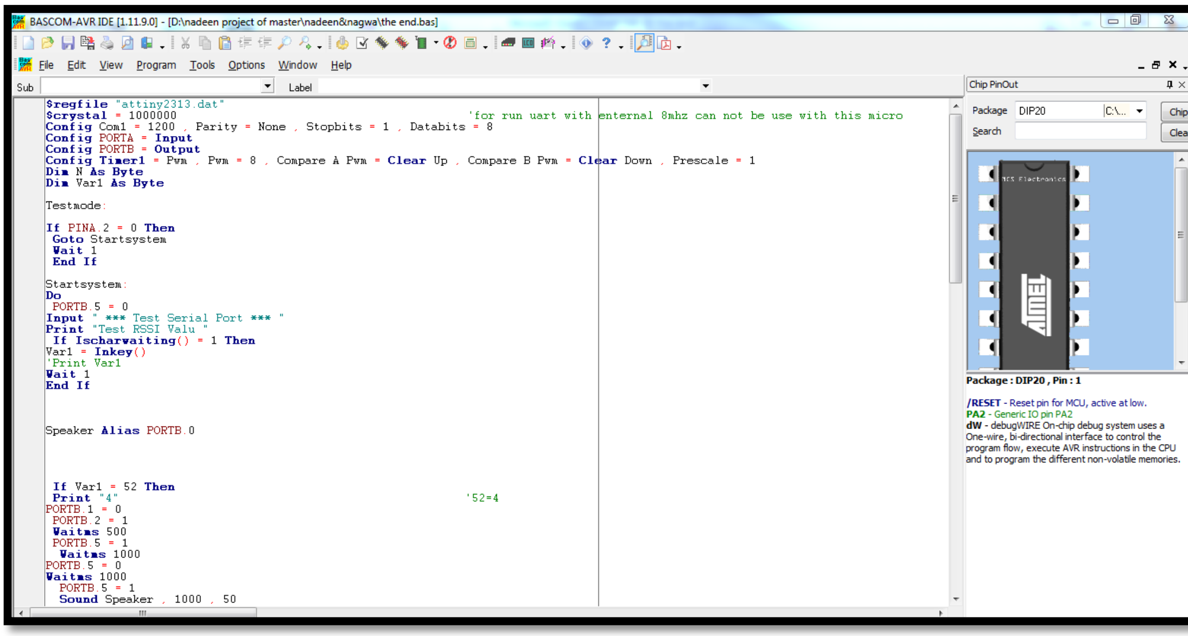


Figure 3.17: BASCOM software for programming ATtinny

3.9 Arduino Uno

For implementation purposes only, an Arduino Uno see (appendix F)was utilized. The unavailability of the Xbee explorer with the provided Xbee module encourages the researcher to find an alternative platform to implement the proposed design with the available incomplete XBee module. An Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital Input/Output (I/O) pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a Universal Serial Bus (USB) connection, a power jack, an In-Circuit Serial Programming (ICSP) header, and a reset button. The most important specifications of Arduino Uno can be found in Table 3.3.

Table 3.3: The most important specifications for Arduino Uno

Microcontroller	ATmega328	Operating Voltage	5V
Input Voltage (recommended)	7-12V	Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)	Analog Input Pins	6
DC Current per I/O Pin	40 mA	DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by boot loader	SRAM	2 KB
EEPROM	1 KB	Clock Speed	16 MHz

Figure 3.18 shows the most important components of the Arduino board.

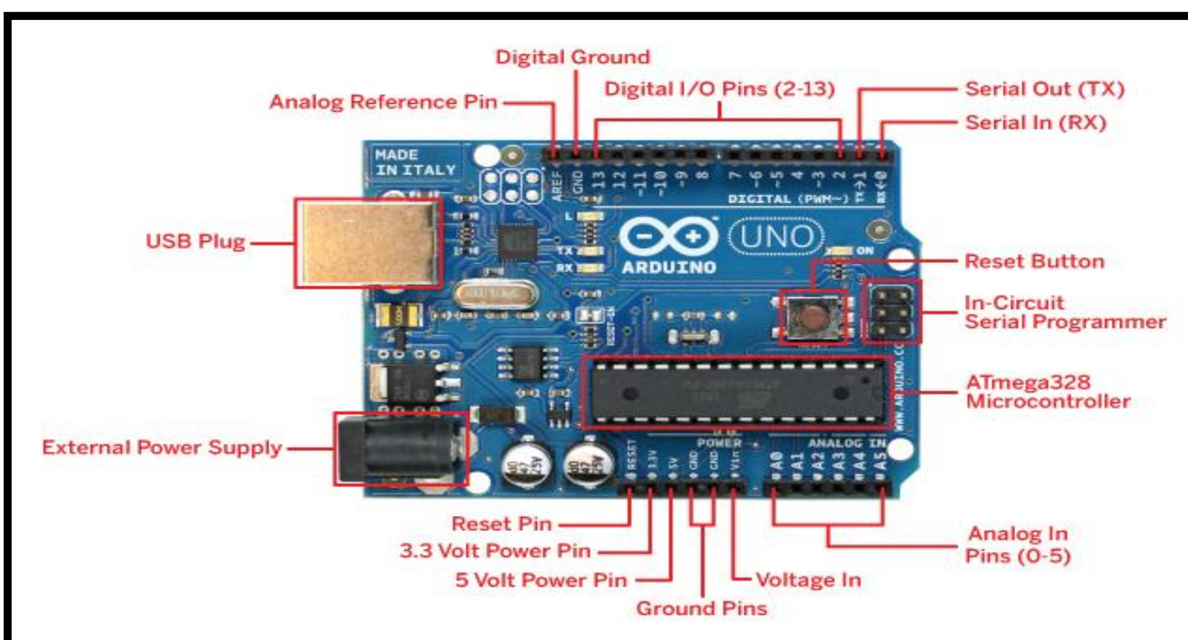


Figure 3.18: The board of Arduino Uno

And because of alternatively using Arduino Uno, the implemented system can be shown in Figure 3.19.

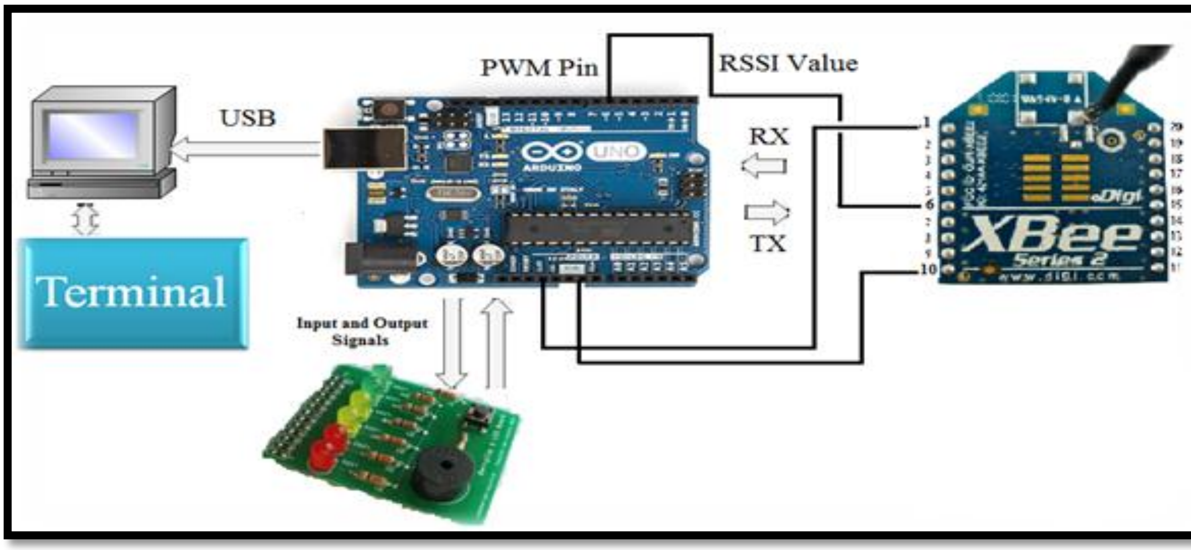


Figure 3.19: The block diagram of the implemented smart bag system using Arduino uno

Figure 3.20 and Figure 3.21 show the schematic of the with Arduino Uno based transmitter and receiver unit,respectively.

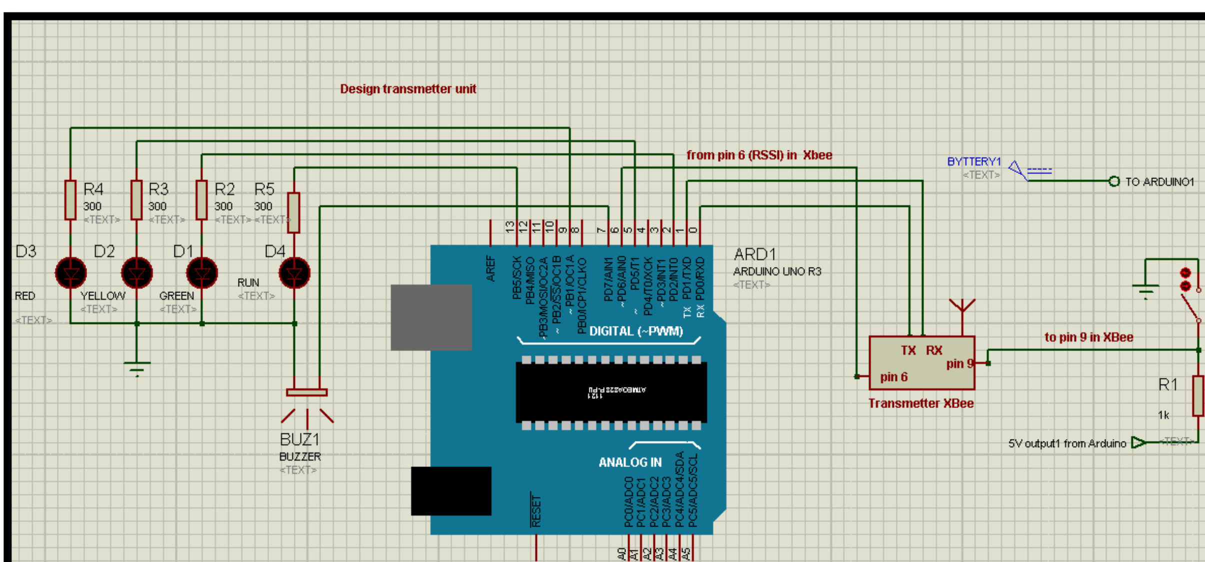


Figure 3.20: Schematic of the Arduino Uno-based transmitter unit

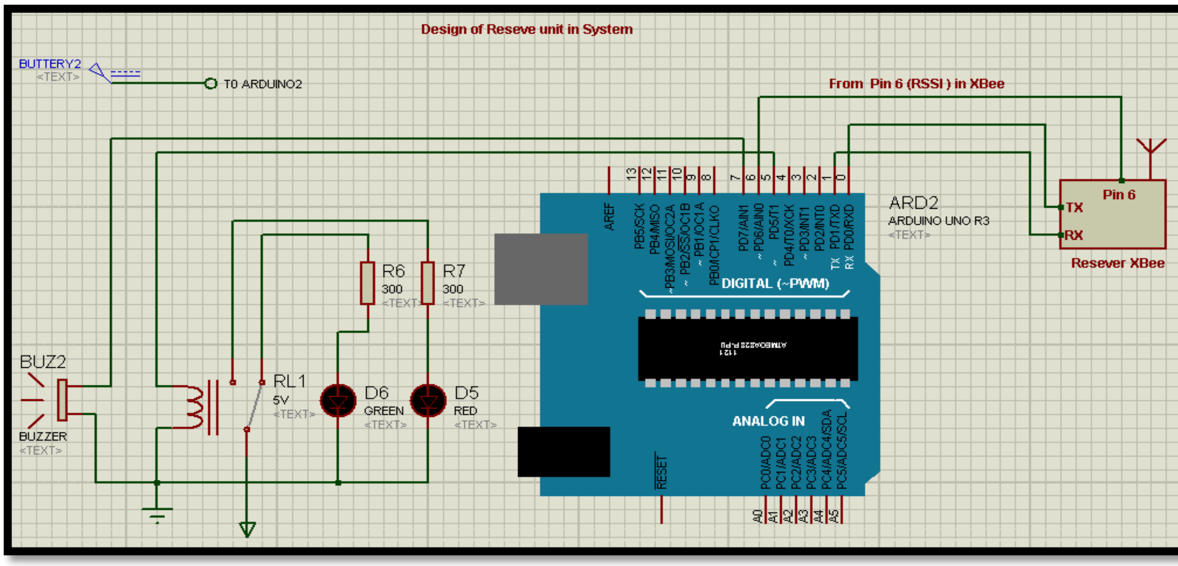


Figure 3.21: Schematic of the Arduino Uno-based receiver unit

However, the Arduino Uno can be programmed using an open source application that is known as the Integrated Development Environment (IDE) and it can download freely and directly from the Arduino website. The Arduino IDE environment is splitted into three areas as shown in Figure 3.22. The blue area at the top of the window includes the control program behavior through toolbar and associated buttons, the middle white area where the code can be developed, and the black section at the bottom of the window is where status messages appear.

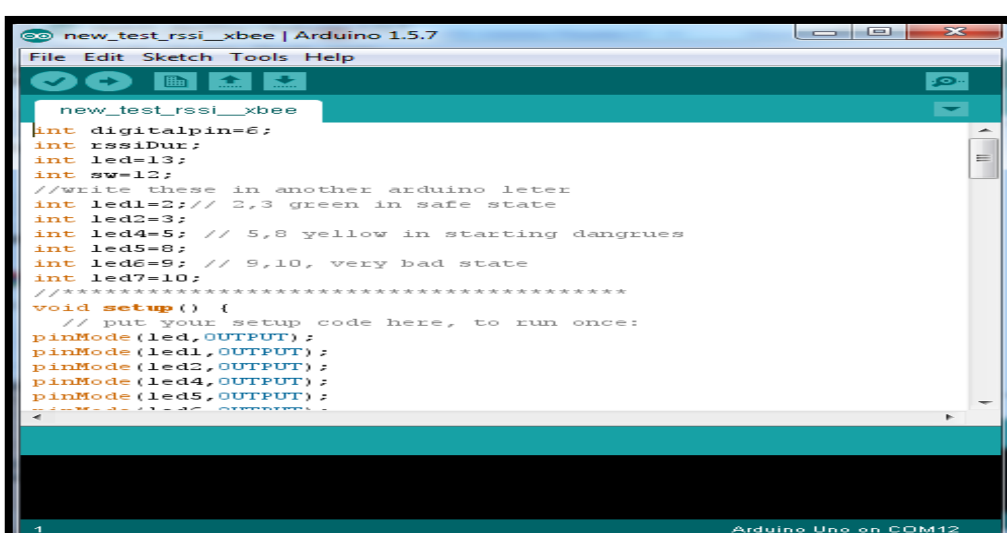


Figure 3.22: Arduino IDE environment

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

In order to determine the required conditions and thresholds that required determining the accurate distance of the bag from its owner, a correction to the ideal RSSI model was achieved through a real indoor and outdoor experiment. Moreover, validation experiments have been conducted using the tuned values of the logarithmic model of the RSSI (dBm)-d (m). In addition, another two model: power and polynomial were also tested. The standard deviation of the above all model was used to reflect the accuracy.

4.2 Experiment Environment Characteristics

For experiment purposes, two low-power RF modules (XBee ZB-24) from Digi International Inc., with 2.4GHz IEEE 802.15.4 transceiver supporting ZigBee protocols were used to figure out the XBee network. Moreover, the RF transceiver modules were configured and interfaced to a microcontroller over serial communication. Unfortunately, the explore module of the available ZigBee was unclosed; therefore, an Arduino Uno-kit was utilized alternatively where each ZigBee node was connected to a laptop. However, the Arduino UnoR3 includes microcontroller ATmega326, and some other peripherals

In this experiment, readings reflect the amount of change in signal intensity with the changing distance was recorded. The experiment environment has chosen to be similar to the expected operation environment (application environment) in terms of presence of the factors that affecting the signal quality/strength during the reception such as the existence of reflective surfaces or moving objects or even the presence of another source of the same operating frequency. A 500 line-of-sight RSSI reading values for each meter up to twenty meters have been recorded and saved to assist the estimation of the distance between the transmitter and receiver.

4.3 Determine the Values of *n* and $RSSI_0$ Empirically

For the purposes of determining the values of *n* and $RSSI_0$ empirically using Equation (page 16), an experiments were designed to realized the proposed anti-theft and anti-forgot system (smart bag system). The measurements were conducted in real indoor and outdoor environment, where the signal paths were violating, at three different locations: at the Industrial Research and Consultancy Centre (IRCC) i.e. at the corridor of the first floor which will be shortly as IRCC, in front of the Electrical and Nuclear Engineering School, Sudan University of Science and Technology will be known shortly as (ENES); and at the researcher home witch will be shortly as home. The procedure of the conducted experiment can be as follow:

1. To establish the WSN network, one of the XBee nodes was setup as a coordinator, and the other as a router and then checks the integrity of the connection between each as shown in Figure 4.1. Then, the cool terminal was configured to display the RSSI values and saved these measures in text files for each distance from 1 to 20m as shown in Figure 4.2.

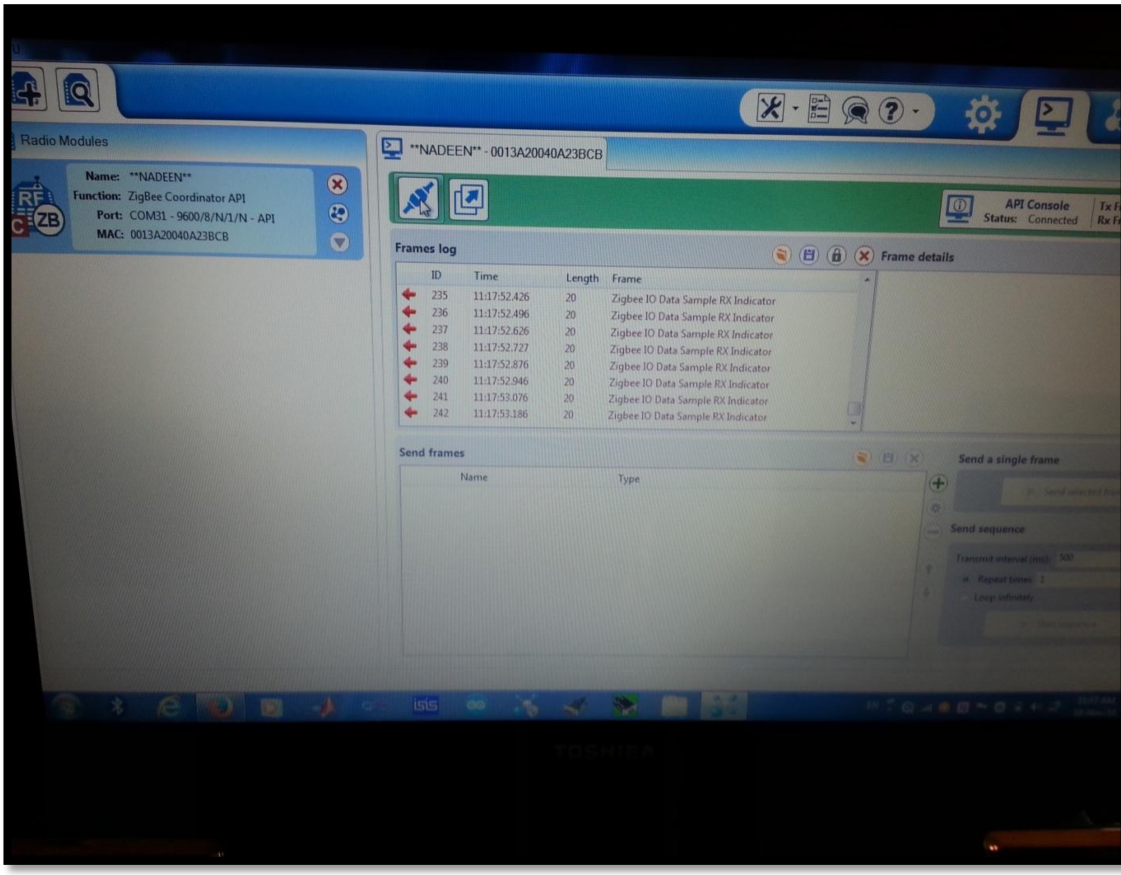


Figure 4.1: A snapshot represents show terminal for coordinator xbee

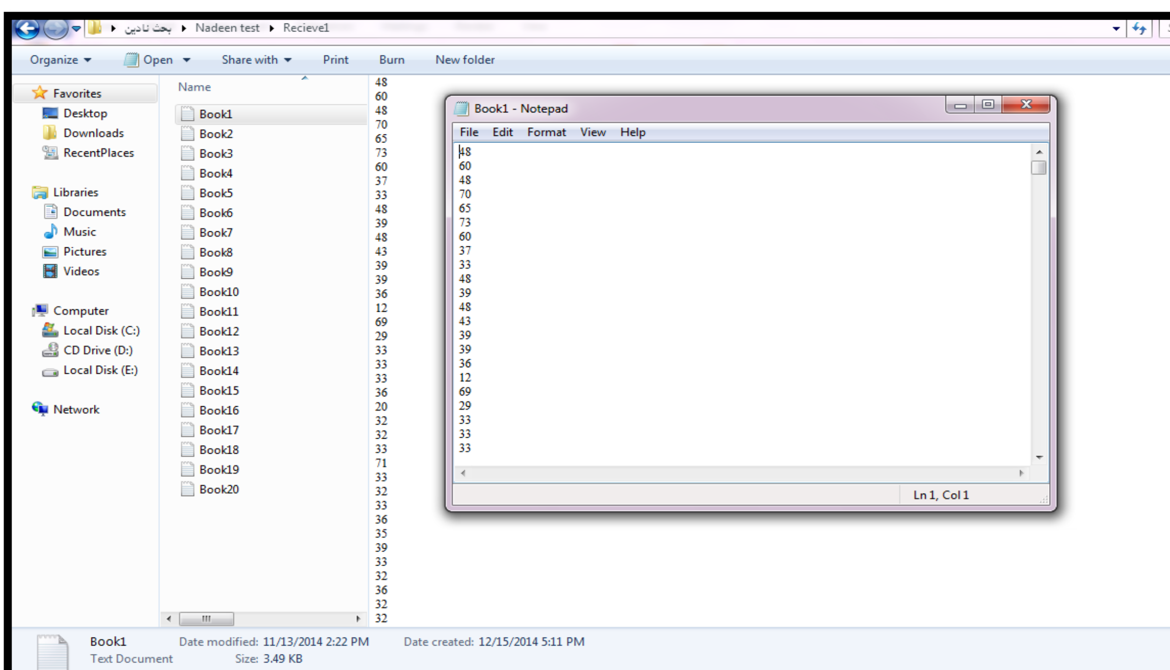


Figure 4.2: Sample of the recorded RSSI measurements

2. The mean, and the mode of the recorded RSSI values were obtained for 10, 20, 50, 100, and 200 samples to study the influence of the number of samples on the log-distance model *i.e.* Equation (2.3) using MATLAB environment as shown in Figure 4.3. In addition, Figure 4.4- 4.7 show the mean and mode of the RSSI measures for 10, 20, 50, 100, and 200 samples according to the distance *d*[m] at IRCC and ENES locations, respectively. It is worthy to note that the RSSI measure at the pin 6 of the ZigBee module was unsigned decimal number therefore, a conversion to dBm should be achieved by converting these unsigned decimals to signed on, in dBm, using the 2's complement.

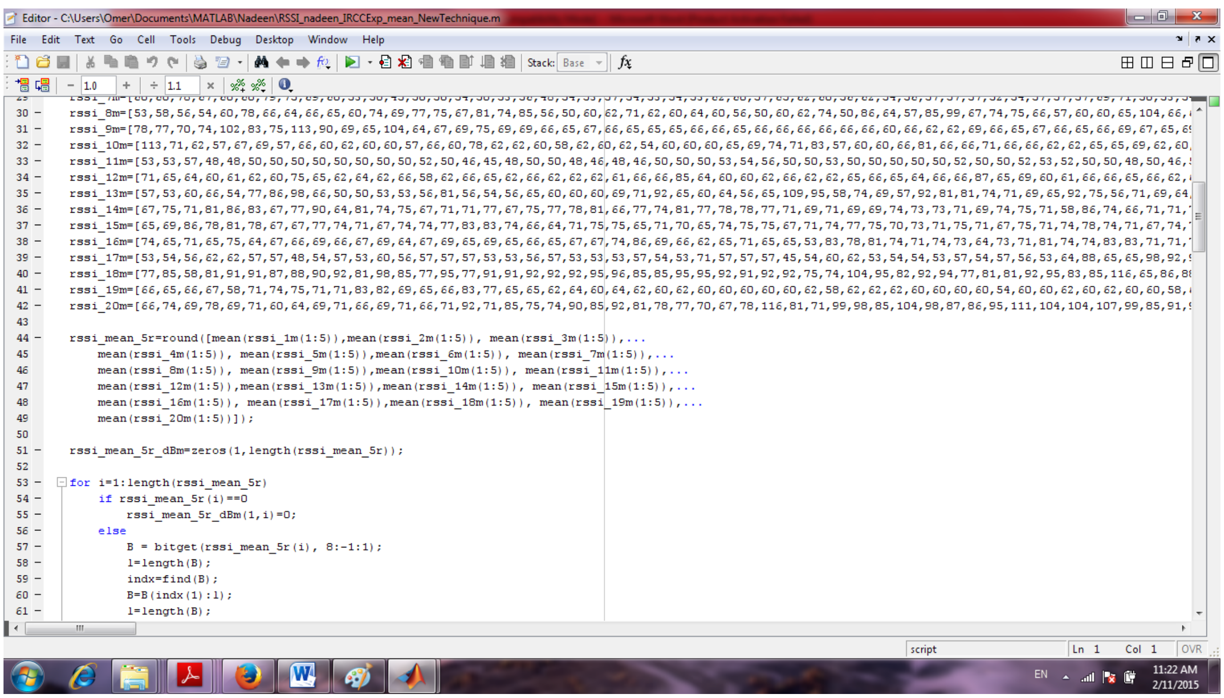


Figure 4.3: A Snapshot of analysis RSSI measures using MATLAB

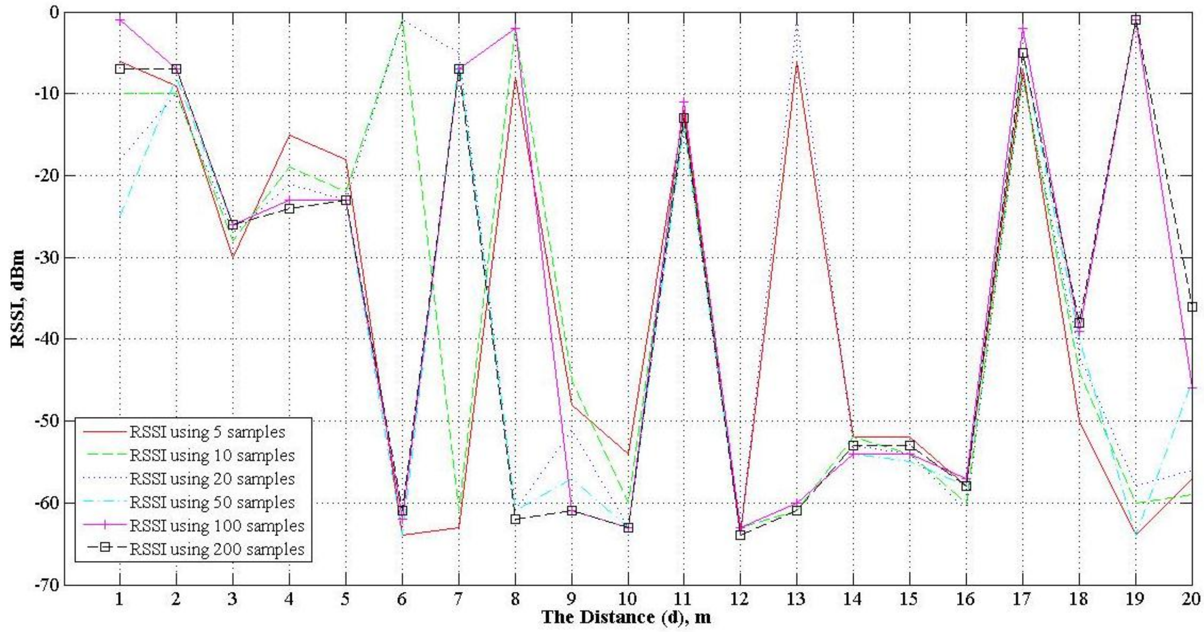


Figure 4.4: The variation of the mode RSSI for 10, 20, 50, 100, and 200 samples at the IRCC

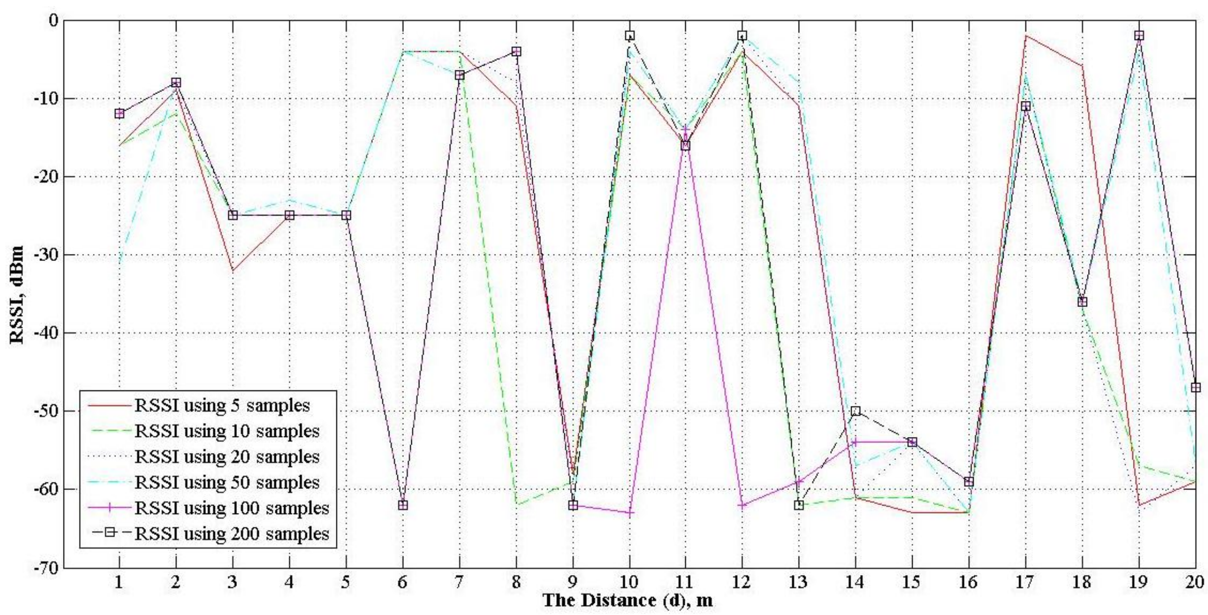


Figure 4.5: The variation of the mean RSSI for 10, 20, 50, 100, and 200 samples at the IRCC

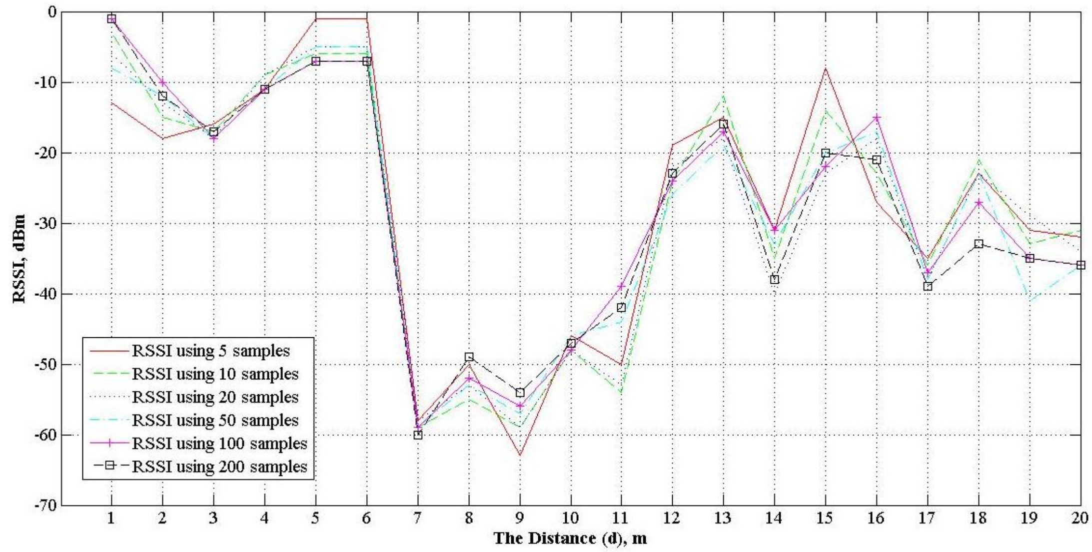


Figure 4.6: The variation of the mode RSSI for 10, 20, 50, 100, and 200 samples at the ENES.

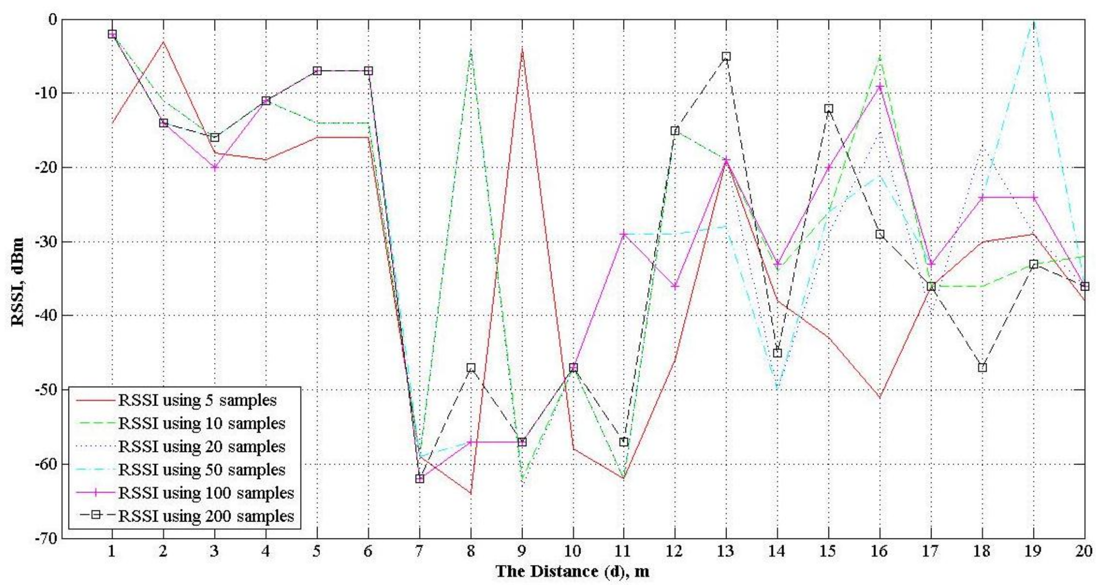


Figure 4.7: The variation of the mean RSSI for 10, 20, 50, 100, and 200 samples at ENES

Observably, the incensement of the samples of the RSSI has a minimal effect on the resultant model i.e. n and $RSSI_0$ where the caused fluctuation on the obtained RSSI (dBm) against the distance d (m) as shown in Figures 4.5-4.7 is due to the instability of the surrounding test environment. This also can explain the coincident occurrence of the RSSI (dBm) for 5 – 200 samples such as in 3, 4, 7 and 10m in Figure 4.7. Table 4.1 shows the comparison between the statistical mean and mode of the first 20 samples RSSI measures at ENES.

Table 4.1: The comparison between the statistical mean and mode of the first 20 samples RSSI measures at ENES

d(m)	The RSSI measures for 20 samples				
	Mean (Unsigned Decimal)	Mean (dBm)	Mode (Unsigned Decimal)	Mode (dBm)	Std.
1	10	-6	6	-2	6.24078267696069
2	51	-13	53	-11	7.93659080992655
3	46	-18	48	-16	3.09668753415816
4	55	-9	53	-11	6.54538809110338
5	59	-5	50	-14	9.76621458746535
6	59	-5	50	-14	9.76621458746535
7	70	-58	69	-59	8.02233723654410
8	75	-53	60	-4	12.8631460504726
9	69	-59	65	-63	5.71793116801044
10	80	-48	81	-47	6.36830143428330
11	75	-53	66	-62	9.31481558989142
12	106	-22	113	-15	10.8453239984603
13	110	-18	109	-19	11.2160599142480
14	88	-40	78	-50	9.89896326310036
15	105	-23	99	-29	23.9944620803700
16	110	-18	113	-15	11.1042903041075
17	90	-38	88	-40	9.49944596722412
18	105	-23	111	-17	9.17247025213578
19	99	-29	99	-29	8.35700147685822
20	94	-34	92	-36	5.52005911486272

From Table 4.1, the values of n and $RSSI_0$ are vitally affected by the change of the environment. This can be clearly recognized when the standard deviation

(St.Deviat) of the RSSI measures over the distance are reviewed. The relative large value of the standard deviation reflecting the fluctuation of the RSSI measures which is due to the real environment. It is worthy to note that for an ideal test environment, the deviation of the RSSI measures can be minimized. However, using either the statistical mean or mode, as shown in Table 4.1, will not cause a numerous change in the values of the resultant RSSI and inheritably the values of n and $RSSI_0$. Refereeing to the reviewed literature, the deviation of the value of the RSSI can be along ±4dBm to ±8dBm, which is quite high deviation for distance measurement [17]. This can concluded in a general belief that in wireless sensor networks, the RSSI is a bad estimator of link quality. This conclusion was tested during this research.

3. The curve fitting tool in the MATLAB environment was utilized to obtain the values of n and $RSSI_0$ in Equation (2.3) were using the recorded RSSI measures as can be seen in Figure 4.8.

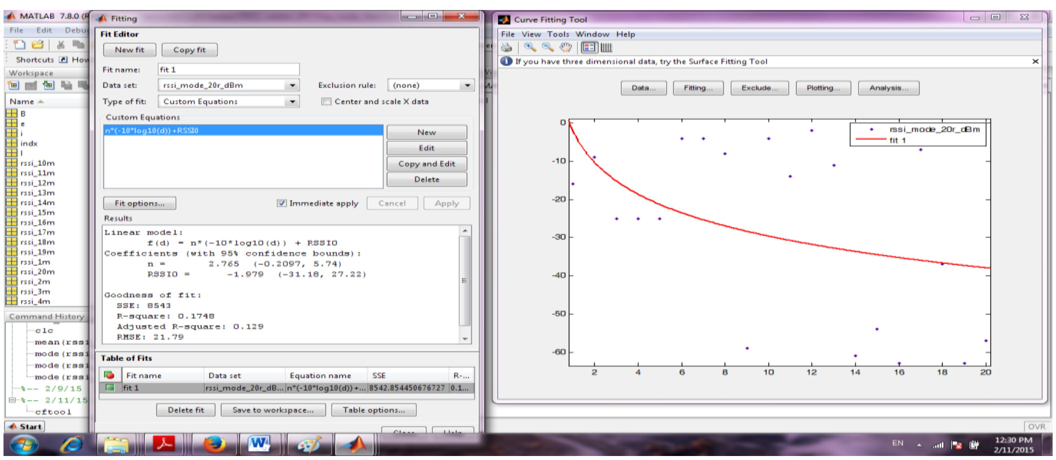


Figure 4.8: A snapshot of the curve fitting process for the RSSI measures

Table 4.2 and Table 4.3 show the values of n and $RSSI_0$ for the measures at the three experiment locations when the statistical mean and mode were used respectively.

Table 4.2: The values of *n* and $RSSI_0$ for the mean RSSI (dBm) measures using Equation (2.3)

Loc.	Mean RSSI Measures (dBm)					
	For 5 samples	For 10 samples	For 20 samples	For 50 samples	For 100 samples	For 200 samples
Home	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.164 (0.009102, 4.319) RSSI0 = -19.36 (-40.51, 1.792) Goodness of fit: SSE: 4482 R-square: 0.1983 Adjusted R-square: 0.1537 RMSE: 15.78	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.682 (-0.6663, 4.03) RSSI0 = -21.49 (-44.54, 1.564) Goodness of fit: SSE: 5323 R-square: 0.1118 Adjusted R-square: 0.06241 RMSE: 17.2	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.629 (1.276, 5.982) RSSI0 = -0.6868 (-23.79, 22.41) Goodness of fit: SSE: 5345 R-square: 0.3684 Adjusted R-square: 0.3333 RMSE: 17.23	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.582 (1.203, 5.962) RSSI0 = -1.167 (-24.53, 22.19) Goodness of fit: SSE: 5466 R-square: 0.3572 Adjusted R-square: 0.3215 RMSE: 17.43	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.62 (0.3549, 4.885) RSSI0 = -14.41 (-36.65, 7.818) Goodness of fit: SSE: 4952 R-square: 0.247 Adjusted R-square: 0.2052 RMSE: 16.59	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.683 (-0.3527, 3.719) RSSI0 = -24.82 (-44.81, -4.84) Goodness of fit: SSE: 4001 R-square: 0.1436 Adjusted R-square: 0.09598 RMSE: 14.91

ENES	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.589 (-0.8599, 4.037) RSSI0 = -12.79 (-36.83, 11.24) Goodness of fit: SSE: 5787 R-square: 0.09357 Adjusted R-square: 0.04321 RMSE: 17.93	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.066 (-0.3538, 4.485) RSSI0 = -9.011 (-32.76, 14.74) Goodness of fit: SSE: 5650 R-square: 0.1516 Adjusted R-square: 0.1045 RMSE: 17.72	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.161 (-0.1588, 4.48) RSSI0 = -8.736 (-31.5, 14.03) Goodness of fit: SSE: 5194 R-square: 0.1755 Adjusted R-square: 0.1296 RMSE: 16.99	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.161 (-0.034, 4.356) RSSI0 = -8.682 (-30.23, 12.86) Goodness of fit: SSE: 4651 R-square: 0.192 Adjusted R-square: 0.1471 RMSE: 16.08	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.336 (0.1968, 4.475) RSSI0 = -6.125 (-27.12, 14.87) Goodness of fit: SSE: 4417 R-square: 0.2263 Adjusted R-square: 0.1833 RMSE: 15.67	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.525 (0.4625, 4.587) RSSI0 = -5.192 (-25.43, 15.05) Goodness of fit: SSE: 4104 R-square: 0.2688 Adjusted R-square: 0.2281 RMSE: 15.1
IRCC.	Lognormal model: f(d) = n*(-	Lognormal model: f(d) = n*(-	Lognormal model: f(d) = n*(-	Lognormal model: f(d) = n*(-	Lognormal model: f(d) = n*(-	Lognormal model: f(d) = n*(-

	10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.27 (0.3609, 6.179) RSSI0 = -6.791 (-35.34, 21.76) Goodness of fit: SSE: 8168 R-square: 0.2365 Adjusted R-square: 0.1941 RMSE: 21.3	10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.833 (1.126, 6.54) RSSI0 = -1.364 (-27.93, 25.2) Goodness of fit: SSE: 7071 R-square: 0.3296 Adjusted R-square: 0.2924 RMSE: 19.82	10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.092 (0.1473, 6.037) RSSI0 = -6.125 (-35.03, 22.78) Goodness of fit: SSE: 8370 R-square: 0.2128 Adjusted R-square: 0.1691 RMSE: 21.56	10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.803 (0.04895, 5.557) RSSI0 = -14.93 (-41.96, 12.1) Goodness of fit: SSE: 7321 R-square: 0.2026 Adjusted R-square: 0.1583 RMSE: 20.17	10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.784 (-0.4411, 6.008) RSSI0 = -7.51 (-39.16, 24.14) Goodness of fit: SSE: 1.004e+004 R-square: 0.1545 Adjusted R-square: 0.1075 RMSE: 23.61	10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.383 (-0.7437, 5.51) RSSI0 = -14.24 (-44.93, 16.45) Goodness of fit: SSE: 9436 R-square: 0.1247 Adjusted R-square: 0.07605 RMSE: 22.9

Table 4.3: The values of n and $RSSI_0$ for the mean RSSI (dBm) measures using Equation (2.3)

Loc.	Mode RSSI measures (dBm)					
	For 5 samples	For 10 samples	For 20 samples	For 50 samples	For 100 samples	For 200 samples
Home	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.893 (1.655, 6.132) RSSI0 = -5.607 (-27.58, 16.37) Goodness of fit: SSE: 4839 R-square: 0.4258 Adjusted R-square: 0.3939 RMSE: 16.4	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.597 (1.343, 5.85) RSSI0 = -5.087 (-27.2, 17.03) Goodness of fit: SSE: 4901 R-square: 0.3845 Adjusted R-square: 0.3503 RMSE: 16.5	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.459 (1.013, 5.905) RSSI0 = -2.549 (-26.56, 21.46) Goodness of fit: SSE: 5776 R-square: 0.329 Adjusted R-square: 0.2918 RMSE: 17.91	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.332 (0.9009, 5.763) RSSI0 = -2.37 (-26.23, 21.49) Goodness of fit: SSE: 5704 R-square: 0.3154 Adjusted R-square: 0.2773 RMSE: 17.8	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.524 (1.358, 5.691) RSSI0 = -2.901 (-24.17, 18.36) Goodness of fit: SSE: 4530 R-square: 0.3936 Adjusted R-square: 0.3599 RMSE: 15.86	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.625 (1.283, 5.967) RSSI0 = -1.277 (-24.27, 21.71) Goodness of fit: SSE: 5295 R-square: 0.37 Adjusted R-square: 0.335 RMSE: 17.15

ENES	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.701 (0.3464, 5.055) RSSI0 = -8.322 (-31.43, 14.79) Goodness of fit: SSE: 5350 R-square: 0.244 Adjusted R-square: 0.202 RMSE: 17.24	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.286 (-0.1502, 4.721) RSSI0 = -5.888 (-29.8, 18.02) Goodness of fit: SSE: 5727 R-square: 0.1776 Adjusted R-square: 0.1319 RMSE: 17.84	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.968 (-0.4616, 4.398) RSSI0 = -9.555 (-33.41, 14.3) Goodness of fit: SSE: 5700 R-square: 0.1386 Adjusted R-square: 0.09074 RMSE: 17.79	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.968 (-0.4616, 4.398) RSSI0 = -9.555 (-33.41, 14.3) Goodness of fit: SSE: 5700 R-square: 0.1386 Adjusted R-square: 0.09074 RMSE: 17.79	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.85 (-0.4726, 4.172) RSSI0 = -10.35 (-33.14, 12.45) Goodness of fit: SSE: 5205 R-square: 0.1346 Adjusted R-square: 0.08655 RMSE: 17	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.59 (0.1206, 5.059) RSSI0 = -5.44 (-29.68, 18.8) Goodness of fit: SSE: 5886 R-square: 0.2124 Adjusted R-square: 0.1687 RMSE: 18.08

IRCC	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.274 (-0.9282, 5.476) RSSI0 = -5.995 (-37.43, 25.44) Goodness of fit: SSE: 9898 R-square: 0.1101 Adjusted R-square: 0.06061 RMSE: 23.45	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 3.11 (0.06309, 6.157) RSSI0 = -4.609 (-34.52, 25.3) Goodness of fit: SSE: 8962 R-square: 0.2035 Adjusted R-square: 0.1592 RMSE: 22.31	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.765 (-0.2097, 5.74) RSSI0 = -1.979 (-31.18, 27.22) Goodness of fit: SSE: 8543 R-square: 0.1748 Adjusted R-square: 0.129 RMSE: 21.79	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.26 (-1.815, 4.335) RSSI0 = -13.16 (-43.35, 17.02) Goodness of fit: SSE: 9126 R-square: 0.03956 Adjusted R-square: -0.0138 RMSE: 22.52	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 2.435 (-0.6167, 5.487) RSSI0 = -12.16 (-42.12, 17.8) Goodness of fit: SSE: 8991 R-square: 0.135 Adjusted R-square: 0.08699 RMSE: 22.35	Lognormal model: f(d) = n*(-10*log10(d)) + RSSI0 Coefficients (with 95% confidence bounds): n = 1.819 (-1.3, 4.937) RSSI0 = -11.83 (-42.44, 18.78) Goodness of fit: SSE: 9386 R-square: 0.07699 Adjusted R-square: 0.02571 RMSE: 22.84

4. The corresponding values of the n and $RSSI_{d0}$ along the different test environment were averaged, therefore the empirical lognormal model in Equation (2.3) can be updated as:

$$RSSI_d = -5.18 - 2.961log(d)[dBm] \qquad (4.1)$$

$$RSSI_d = -4.69 - 2.73log(d)[dBm] \qquad (4.2)$$

For the mean and mode RSSI measures, respectively. Figures (4.9-4.12) are taken during the tests in the three test environments.

Figure 4.9: A snapshot represents the conducted real experiment at the IRCC for 1m between the two Xbee modules

Figure 4.10: A snapshot represents the conducted real experiment at the IRCC for 15m between the two Xbee modules

Figure 4.11: A snapshot represents the conducted real experiment at the researcher home for 7m between the two Xbee modules

Figure 4.12: A snapshot represents the conducted real experiment at the ENES for 3m between the two Xbee modules

However, the conversion of the RSSI to the dBm encounter a problem for the value that contains all ones in their binary format whenever a two's complement is required such as 1, 3, 7, 15, 31, 63, 127, and 255 can be converted to -1dBm. Therefore, a model based on the direct reading of the RSSI values is required to minimize this encountered error. A polynomial and power model were suggested respectively as follow:

$$F\ (RSSI) = p1*RSSI^{2} + p2*RSSI + p3 \quad \text{(4.3)}$$

$$F\ (RSSI) = a*RSSI^{b} \quad \text{(4.4)}$$

Where p1, p2, p3, a, and b are constants can be obtained from the curve fitting procedure for recorded results of the conducted experiment as described earlier.

Table 4.4 and Table 4.5 show the value of p1, p2, p3, a, and b when the mean and mode where used respectively to described the 5, 10, 20, 50, 100 and 200 samples of the RSSI. It is worthy to note that only the values of p1, p2, p3, a, and b at the 20 RSSI samples were utilized in validation experiment.

Table 4.4: The values of p1, p2, p3, a, and b for the mean RSSI measures using Equations (4.3-4.4)

Loc./ Mean	For 5 samples	For 10 samples	For 20 samples	For 50 samples	For 100 samples	For 200 samples
IRCC Exp.	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1454 (-0.3245, 0.03362) p2 = 4.631 (0.7597, 8.501) p3 = 33 (15.35, 50.65) Goodness of fit: SSE: 2149 R-square: 0.485 Adjusted R-	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1664 (-0.3441, 0.01126) p2 = 5.249 (1.407, 9.09) p3 = 29.77 (12.26, 47.29) Goodness of fit: SSE: 2117 R-square: 0.5445 Adjusted R-	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1459 (-0.3067, 0.01498) p2 = 5.046 (1.569, 8.524) p3 = 27.8 (11.94, 43.66) Goodness of fit: SSE: 1735 R-square: 0.6327 Adjusted R-	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1343 (-0.2828, 0.01414) p2 = 5.064 (1.855, 8.274) p3 = 26.2 (11.57, 40.83) Goodness of fit: SSE: 1478 R-square: 0.7126 Adjusted R-	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1528 (-0.2841, -0.02144) p2 = 5.604 (2.764, 8.443) p3 = 22.78 (9.838, 35.73) Goodness of fit: SSE: 1156 R-square: 0.7851 Adjusted R-	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1421 (-0.3002, 0.016) p2 = 5.575 (2.156, 8.993) p3 = 21.71 (6.123, 37.3) Goodness of fit: SSE: 1676 R-square: 0.7418 Adjusted R-square: 0.7114

	square: 0.4245 RMSE: 11.24	square: 0.4909 RMSE: 11.16	square: 0.5894 RMSE: 10.1	square: 0.6788 RMSE: 9.323	square: 0.7599 RMSE: 8.248	RMSE: 9.93
	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 37.62 (25.19, 50.05) b = 0.2196 (0.08354, 0.3556) Goodness of fit: SSE: 2361 R-square: 0.4344 Adjusted R-square: 0.403 RMSE: 11.45	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 35.55 (23.47, 47.62) b = 0.2467 (0.1079, 0.3855) Goodness of fit: SSE: 2364 R-square: 0.4914 Adjusted R-square: 0.4632 RMSE: 11.46	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 31.97 (21.73, 42.2) b = 0.2852 (0.1555, 0.4148) Goodness of fit: SSE: 1859 R-square: 0.6064 Adjusted R-square: 0.5845 RMSE: 10.16	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 29.44 (20.6, 38.28) b = 0.3232 (0.2025, 0.4438) Goodness of fit: SSE: 1521 R-square: 0.7042 Adjusted R-square: 0.6878 RMSE: 9.192	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 27.5 (19.96, 35.05) b = 0.3501 (0.2405, 0.4596) Goodness of fit: SSE: 1185 R-square: 0.7799 Adjusted R-square: 0.7677 RMSE: 8.112	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 25.81 (17.37, 34.26) b = 0.3788 (0.2489, 0.5088) Goodness of fit: SSE: 1600 R-square: 0.7536 Adjusted R-square: 0.74 RMSE: 9.427
Electric Dep. Exp.	Linear model Poly2: f(x) = p1*x^2 + p2*x	Linear model Poly2: f(x) = p1*x^2 + p2*x	Linear model Poly2: f(x) = p1*x^2 + p2*x	Linear model Poly2: f(x) = p1*x^2 + p2*x	Linear model Poly2: f(x) = p1*x^2 + p2*x	Linear model Poly2: f(x) = p1*x^2 + p2*x

	+ p3 Coefficients (with 95% confidence bounds): p1 = -0.2987 (-0.4575, -0.1399) p2 = 10.13 (6.695, 13.56) p3 = 16.32 (0.661, 31.97) Goodness of fit: SSE: 1691 R-square: 0.8713 Adjusted R-square: 0.8562 RMSE: 9.973	+ p3 Coefficients (with 95% confidence bounds): p1 = -0.2805 (-0.4455, -0.1154) p2 = 9.846 (6.278, 13.41) p3 = 15.26 (-1.009, 31.53) Goodness of fit: SSE: 1826 R-square: 0.8659 Adjusted R-square: 0.8501 RMSE: 10.37	+ p3 Coefficients (with 95% confidence bounds): p1 = -0.2812 (-0.4417, -0.1206) p2 = 9.764 (6.293, 13.24) p3 = 15.62 (-0.2079, 31.45) Goodness of fit: SSE: 1728 R-square: 0.8673 Adjusted R-square: 0.8517 RMSE: 10.08	+ p3 Coefficients (with 95% confidence bounds): p1 = -0.3448 (-0.497, -0.1926) p2 = 11.01 (7.722, 14.3) p3 = 11.69 (-3.309, 26.7) Goodness of fit: SSE: 1553 R-square: 0.8815 Adjusted R-square: 0.8675 RMSE: 9.558	+ p3 Coefficients (with 95% confidence bounds): p1 = -0.35 (-0.5045, -0.1954) p2 = 11.21 (7.866, 14.55) p3 = 10.95 (-4.28, 26.19) Goodness of fit: SSE: 1601 R-square: 0.8827 Adjusted R-square: 0.8689 RMSE: 9.705	+ p3 Coefficients (with 95% confidence bounds): p1 = -0.3606 (-0.5135, -0.2076) p2 = 11.28 (7.969, 14.58) p3 = 10.95 (-4.128, 26.02) Goodness of fit: SSE: 1568 R-square: 0.8791 Adjusted R-square: 0.8649 RMSE: 9.604
	General model Power1: f(x) = a*x^b Coefficients (with 95%	General model Power1: f(x) = a*x^b Coefficients (with 95%	General model Power1: f(x) = a*x^b Coefficients (with 95%	General model Power1: f(x) = a*x^b Coefficients (with 95%	General model Power1: f(x) = a*x^b Coefficients (with 95%	General model Power1: f(x) = a*x^b Coefficients (with 95%

	confidence bounds): a = 30.72 (21.41, 40.03) b = 0.4281 (0.3088, 0.5475) Goodness of fit: SSE: 2220 R-square: 0.8311 Adjusted R-square: 0.8217 RMSE: 11.11	confidence bounds): a = 28.81 (19.74, 37.89) b = 0.448 (0.3245, 0.5716) Goodness of fit: SSE: 2228 R-square: 0.8364 Adjusted R-square: 0.8273 RMSE: 11.13	confidence bounds): a = 29.21 (20.35, 38.07) b = 0.4391 (0.3199, 0.5582) Goodness of fit: SSE: 2070 R-square: 0.8411 Adjusted R-square: 0.8322 RMSE: 10.72	confidence bounds): a = 29.99 (20.43, 39.55) b = 0.4285 (0.3029, 0.554) Goodness of fit: SSE: 2343 R-square: 0.8212 Adjusted R-square: 0.8112 RMSE: 11.41	confidence bounds): a = 29.71 (20.05, 39.37) b = 0.4354 (0.3076, 0.5633) Goodness of fit: SSE: 2439 R-square: 0.8213 Adjusted R-square: 0.8113 RMSE: 11.64	confidence bounds): a = 30.37 (20.47, 40.26) b = 0.4218 (0.2933, 0.5503) Goodness of fit: SSE: 2466 R-square: 0.8099 Adjusted R-square: 0.7993 RMSE: 11.71
HomeExp.	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2345 (-0.4683, -	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2356 (-0.4667, -	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2343 (-0.4673, -	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.236 (-0.4651, -0.007017)	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2455 (-0.4612, -	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2319 (-0.4254, -

	0.0007479) p2 = 6.638 (1.584, 11.69) p3 = 36.7 (13.66, 59.74) Goodness of fit: SSE: 3664 R-square: 0.4434 Adjusted R-square: 0.3779 RMSE: 14.68	0.004558) p2 = 6.754 (1.758, 11.75) p3 = 34.75 (11.97, 57.53) Goodness of fit: SSE: 3580 R-square: 0.4674 Adjusted R-square: 0.4047 RMSE: 14.51	0.001275) p2 = 6.802 (1.764, 11.84) p3 = 33.75 (10.78, 56.72) Goodness of fit: SSE: 3640 R-square: 0.4769 Adjusted R-square: 0.4154 RMSE: 14.63	p2 = 6.88 (1.929, 11.83) p3 = 33.13 (10.55, 55.71) Goodness of fit: SSE: 3517 R-square: 0.4943 Adjusted R-square: 0.4348 RMSE: 14.38	0.02982) p2 = 7.122 (2.46, 11.79) p3 = 32.34 (11.08, 53.61) Goodness of fit: SSE: 3119 R-square: 0.5379 Adjusted R-square: 0.4836 RMSE: 13.55	0.03848) p2 = 6.929 (2.747, 11.11) p3 = 32.22 (13.15, 51.3) Goodness of fit: SSE: 2509 R-square: 0.5999 Adjusted R-square: 0.5528 RMSE: 12.15
	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 45.5 (29.79, 61.21) b = 0.2159 (0.07362,	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 43.67 (28.4, 58.95) b = 0.2286 (0.085, 0.3723)	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 42.58 (27.42, 57.74) b = 0.2381 (0.09223,	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 42.14 (27.25, 57.02) b = 0.2424 (0.09791,	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 42.03 (27.88, 56.19) b = 0.246 (0.1083, 0.3837)	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 41.01 (28.57, 53.45) b = 0.2556 (0.1319,

	0.3581) Goodness of fit: SSE: 3737 R-square: 0.4323 Adjusted R-square: 0.4007 RMSE: 14.41	Goodness of fit: SSE: 3635 R-square: 0.4592 Adjusted R-square: 0.4292 RMSE: 14.21	0.3839) Goodness of fit: SSE: 3656 R-square: 0.4746 Adjusted R-square: 0.4454 RMSE: 14.25	0.387) Goodness of fit: SSE: 3559 R-square: 0.4883 Adjusted R-square: 0.4598 RMSE: 14.06	Goodness of fit: SSE: 3246 R-square: 0.5192 Adjusted R-square: 0.4925 RMSE: 13.43	0.3794) Goodness of fit: SSE: 2563 R-square: 0.5914 Adjusted R-square: 0.5687 RMSE: 11.93

Table 4.5: The values of p1, p2, p3, a, and b for the mode RSSI measures using Equations (4.3-4.4)

Loc./Mode	For 5 samples	For 10 samples	For 20 samples	For 50 samples	For 100 samples	For 200 samples
IRCC Exp.	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1049 (-0.2412, 0.03134) p2 =	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.0941 (-0.2561, 0.06794) p2 =	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.07325 (-0.2344, 0.08785) p2 =	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1352 (-0.2823, 0.01199) p2 =	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1887 (-0.3465, -0.03088) p2 =	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1724 (-0.3368, -0.008055) p2 =

	3.827 (0.8814, 6.773) p3 = 29.57 (16.14, 43) Goodness of fit: SSE: 1245 R-square: 0.61 Adjusted R-square: 0.5641 RMSE: 8.558	3.964 (0.4609, 7.467) p3 = 29.88 (13.91, 45.85) Goodness of fit: SSE: 1760 R-square: 0.6126 Adjusted R-square: 0.567 RMSE: 10.18	3.478 (-0.004847, 6.961) p3 = 31.39 (15.51, 47.27) Goodness of fit: SSE: 1740 R-square: 0.5987 Adjusted R-square: 0.5515 RMSE: 10.12	4.948 (1.766, 8.129) p3 = 24.3 (9.791, 38.8) Goodness of fit: SSE: 1452 R-square: 0.6931 Adjusted R-square: 0.657 RMSE: 9.242	6.469 (3.057, 9.881) p3 = 17.41 (1.847, 32.96) Goodness of fit: SSE: 1670 R-square: 0.7419 Adjusted R-square: 0.7115 RMSE: 9.912	6.128 (2.575, 9.682) p3 = 18.24 (2.042, 34.45) Goodness of fit: SSE: 1811 R-square: 0.722 Adjusted R-square: 0.6892 RMSE: 10.32
	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 31.6 (22.55, 40.65) b = 0.2504 (0.1335, 0.3674)	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 30.54 (20.43, 40.64) b = 0.2912 (0.1574, 0.4251)	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 30.68 (20.53, 40.84) b = 0.2845 (0.1504, 0.4186)	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 27.97 (19.22, 36.71) b = 0.3214 (0.1957, 0.447) Goodness of fit:	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 25.2 (16.44, 33.96) b = 0.3779 (0.2398, 0.516) Goodness of fit:	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 24.89 (15.93, 33.84) b = 0.3803 (0.2373, 0.5232)

	Goodness of fit: SSE: 1340 R-square: 0.5803 Adjusted R-square: 0.557 RMSE: 8.627	Goodness of fit: SSE: 1838 R-square: 0.5955 Adjusted R-square: 0.573 RMSE: 10.11	Goodness of fit: SSE: 1827 R-square: 0.5787 Adjusted R-square: 0.5552 RMSE: 10.08	SSE: 1482 R-square: 0.6868 Adjusted R-square: 0.6693 RMSE: 9.073	SSE: 1718 R-square: 0.7346 Adjusted R-square: 0.7198 RMSE: 9.769	Goodness of fit: SSE: 1807 R-square: 0.7227 Adjusted R-square: 0.7073 RMSE: 10.02
Electric Dep. Exp.	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1886 (-0.3325, -0.04468) p2 = 7.746 (4.635, 10.86) p3 = 14.98 (0.7908, 29.17) Goodness of fit: SSE: 1389	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2717 (-0.4926, -0.05076) p2 = 9.772 (4.995, 14.55) p3 = 12.28 (-9.495, 34.06) Goodness of fit: SSE: 3273	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2281 (-0.4507, -0.005563) p2 = 8.841 (4.029, 13.65) p3 = 15.06 (-6.884, 36.99) Goodness of fit: SSE: 3321	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.5189 (-0.879, -0.1587) p2 = 13.48 (5.692, 21.26) p3 = 4.88 (-30.62, 40.38) Goodness of fit: SSE: 8697 R-square:	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.3178 (-0.4704, -0.1652) p2 = 10.89 (7.595, 14.19) p3 = 9.868 (-5.175, 24.91) Goodness of fit: SSE: 1561	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.3737 (-0.5921, -0.1553) p2 = 11.49 (6.764, 16.21) p3 = 9.771 (-11.76, 31.3) Goodness of fit: SSE: 3199

	R-square: 0.8797 Adjusted R-square: 0.8655 RMSE: 9.038	R-square: 0.7897 Adjusted R-square: 0.765 RMSE: 13.88	R-square: 0.7807 Adjusted R-square: 0.7549 RMSE: 13.98	0.513 Adjusted R-square: 0.4557 RMSE: 22.62	R-square: 0.8971 Adjusted R-square: 0.885 RMSE: 9.584	R-square: 0.7787 Adjusted R-square: 0.7526 RMSE: 13.72
	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 22.97 (16.03, 29.91) b = 0.4912 (0.3734, 0.6089) Goodness of fit: SSE: 1475 R-square: 0.8723 Adjusted R-square: 0.8652 RMSE: 9.051	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 25.99 (14.82, 37.17) b = 0.4782 (0.3103, 0.6461) Goodness of fit: SSE: 3683 R-square: 0.7634 Adjusted R-square: 0.7502 RMSE: 14.3	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 25.52 (14.69, 36.34) b = 0.4816 (0.3159, 0.6472) Goodness of fit: SSE: 3491 R-square: 0.7695 Adjusted R-square: 0.7567 RMSE: 13.93	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 35.07 (10.75, 59.39) b = 0.3272 (0.04884, 0.6055) Goodness of fit: SSE: 1.162e+004 R-square: 0.3491 Adjusted R-square: 0.313 RMSE: 25.41	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 27 (18.41, 35.58) b = 0.4776 (0.3535, 0.6018) Goodness of fit: SSE: 2168 R-square: 0.8572 Adjusted R-square: 0.8492 RMSE: 10.97	General model Power1: f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 30.24 (17.25, 43.24) b = 0.4189 (0.2495, 0.5884) Goodness of fit: SSE: 4217 R-square: 0.7082 Adjusted R-square: 0.692 RMSE: 15.31

HomeExp.	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2213 (-0.4073, -0.03539) p2 = 6.659 (2.639, 10.68) p3 = 29.25 (10.91, 47.58) Goodness of fit: SSE: 2319 R-square: 0.6048 Adjusted R-square: 0.5583 RMSE: 11.68	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2626 (-0.4909, -0.03434) p2 = 7.617 (2.681, 12.55) p3 = 28.36 (5.855, 50.87) Goodness of fit: SSE: 3495 R-square: 0.5427 Adjusted R-square: 0.4889 RMSE: 14.34	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2339 (-0.4633, -0.004527) p2 = 6.891 (1.931, 11.85) p3 = 32.47 (9.851, 55.08) Goodness of fit: SSE: 3529 R-square: 0.5024 Adjusted R-square: 0.4439 RMSE: 14.41	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.2761 (-0.5681, 0.01578) p2 = 7.79 (1.479, 14.1) p3 = 30.43 (1.654, 59.21) Goodness of fit: SSE: 5714 R-square: 0.4102 Adjusted R-square: 0.3409 RMSE: 18.33	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.3067 (-0.5773, -0.03605) p2 = 8.484 (2.633, 14.33) p3 = 28.43 (1.75, 55.11) Goodness of fit: SSE: 4911 R-square: 0.4742 Adjusted R-square: 0.4123 RMSE: 17	Linear model Poly2: f(x) = p1*x^2 + p2*x + p3 Coefficients (with 95% confidence bounds): p1 = -0.1951 (-0.4859, 0.09558) p2 = 6.94 (0.6548, 13.23) p3 = 22.93 (-5.727, 51.59) Goodness of fit: SSE: 5667 R-square: 0.5159 Adjusted R-square: 0.459 RMSE: 18.26
	General model Power1:	General model Power1:	General model Power1:	General model Power1:	General model Power1:	General model Power1:

	f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 37.78 (25.9, 49.65) b = 0.2647 (0.1367, 0.3926) Goodness of fit: SSE: 2384 R-square: 0.5936 Adjusted R-square: 0.5711 RMSE: 11.51	f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 39.4 (24.69, 54.11) b = 0.267 (0.1151, 0.419) Goodness of fit: SSE: 3681 R-square: 0.5183 Adjusted R-square: 0.4916 RMSE: 14.3	f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 41.28 (26.65, 55.92) b = 0.25 (0.1052, 0.3948) Goodness of fit: SSE: 3502 R-square: 0.5062 Adjusted R-square: 0.4788 RMSE: 13.95	f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 42.06 (23.05, 61.06) b = 0.2502 (0.06565, 0.4348) Goodness of fit: SSE: 5909 R-square: 0.3901 Adjusted R-square: 0.3562 RMSE: 18.12	f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 42.15 (24.21, 60.09) b = 0.2548 (0.08117, 0.4284) Goodness of fit: SSE: 5319 R-square: 0.4305 Adjusted R-square: 0.3988 RMSE: 17.19	f(x) = a*x^b Coefficients (with 95% confidence bounds): a = 30.04 (13.71, 46.37) b = 0.3671 (0.1507, 0.5835) Goodness of fit: SSE: 5800 R-square: 0.5046 Adjusted R-square: 0.4771 RMSE: 17.95

Table 4.6 and Figure 4.13 present the average stander deviation of the measured distance using RSSI measures for the three different models when the mean and the mode were used to represent the RSSI values. From Table 4.6, it is clearly that the log linear model performs the polynomial and power model, where the mode and mean were used. In addition, the used statistical measure has a minimal effect on the performance of the associated model which can be recognized from Table 4.6 and Figure 4.13.

Table 4.6: The values of average of St.Deviat for the mean and the mode for RSSI (measures using ENES experiment) when using the three models

Statistical Parameter	Model/Distance	1m	2m	3m
Mode	Average of St.Deviat lognormal	0.3046	0.4158	1.0864
	Average of St.Deviat poly	1.9158	2.1103	5.4116
	Average of St.Deviat power	2.5737	2.5712	2.6898
Mean	Average of St.Deviat lognormal	0.2237	0.1846	0.7386
	Average of St.Deviat poly	1.8559	2.0329	5.0678
	Average of St.Deviat power	2.5598	2.5555	2.6681

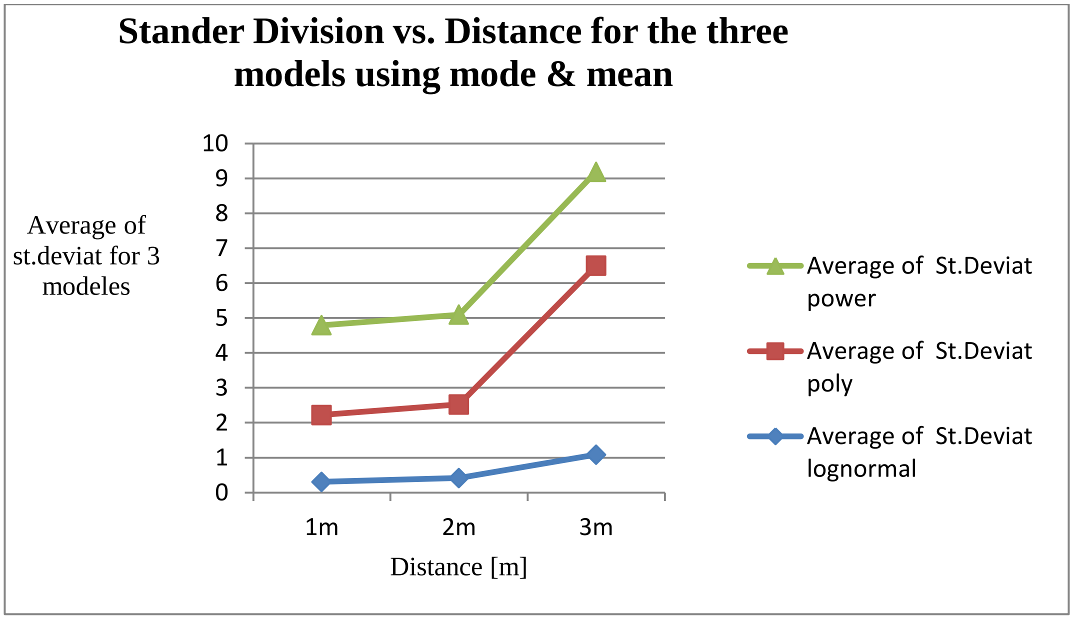


Figure 4.13: Relation between stander division for the three models using mean or mode for ENES experiment.

Table 4.7 represents the real distance vs. the average of distance and the stander division for the lognormal model using mean and mode for ENES experiments.

Table 4.7: The real distance vs. the average of distance in ENES

Average of Distance	form/Distance	1m	2m	3m	7m
	mean	1.76	7.03	9.15	19.17
	mode	6.23	11.74	23.86	42.39

Table 4.8: The stander division for calculation distance from real distance

Average of st.Deviat. lognormal	form/Distance	1m	2m	3m	7m
	mean	1.29	11.63	15.1	13.83
	mode	12.91	17.53	34.5	37.76

Figures 4.14 and 4.15 show the Relation between stander deviation vs. real distance using mode and mean for ENES experiment, respectively.

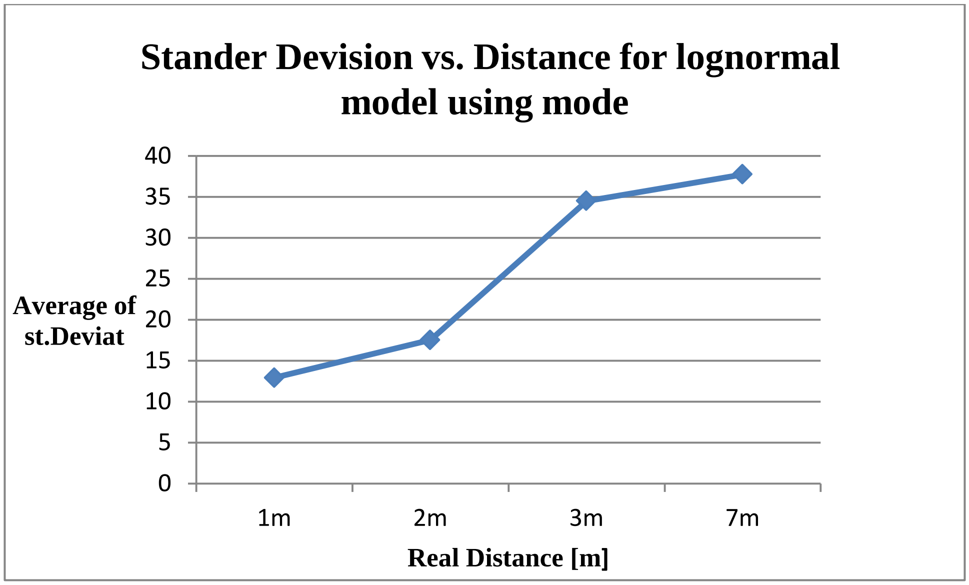


Figure 4.14: Relation between stander deviations vs. real distance using mode for ENES experiment

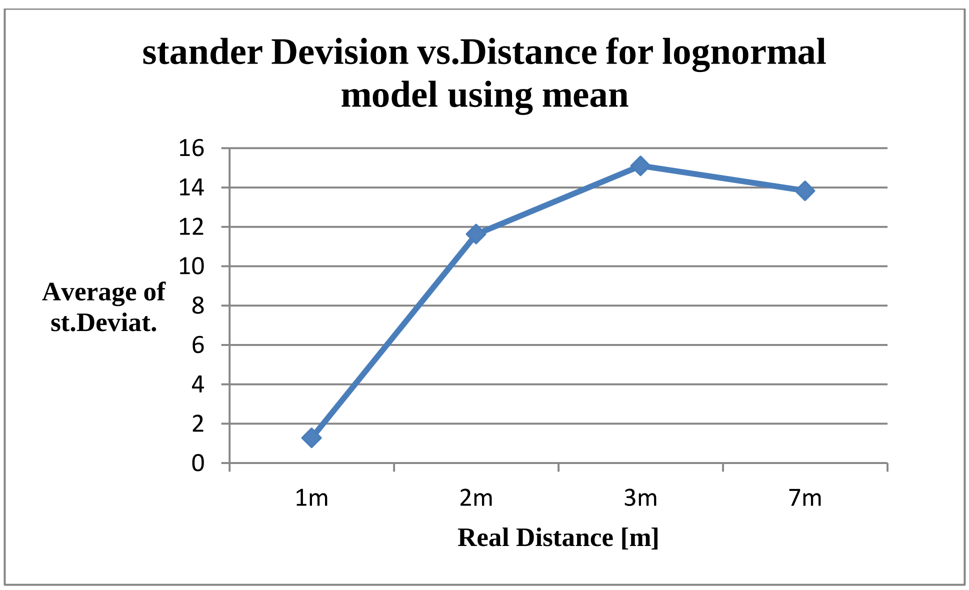


Figure 4.15: Relation between stander deviations vs. real distance using mean for ENES experiment

Table 4.9 shows the variation of the standard deviation for the RSSI values (unsigned decimal) recorded along the two conducted experiments at the same location. It can conclude that the RSSI values were crucially affected by the environment.

Table 4.9: The stander division for fresh RSSI from terminal indifferent distance for the two experiments at ENES

Distance	St. Deviat. of the 1st Experiment	St. Deviat. of the 2nd Experiment
1m	3.067658	2.139629
2m	6.560061	3.547143
3m	4.350814	3.716775
7m	5.476555	11.73664

CHAPTER FIVE
CONCLUSION AND RECOMMENDATIONS
5.1 Conclusion

The techniques that used in the monitoring and tracking systems vary depending on the object to be followed by the system and the surrounding environment conditions. In this thesis, an anti-theft and anti forget system for personal valuable holdings based on XBee technology was designed. Taking into account the factors affecting such system such as: size, consumed power, and number of integrated sensors. The designed system measures the intensity of the signal depending on the distance between the bag and its owner. The received signal strength from the XBee module was improved using the conventional log-linear model based on real experiment. Another two models based on polynomial and power were tested. It can be concluded that the log-linear model is more suitable to estimate the distance (m) based on RSSI (dBm) values. In addition, the estimated distance is crucially affected by the operation environment.

5.2 Recommendations

This system can be developed in several ways; it can be allocated to protect and keep track of the bags containing hazardous laboratory samples and this requires adding sensors and some other elements that achieve an appropriate environment for the sample. It can also be allocated to the system for the transfer of confidential documents or even with bank transactions, here is only in this case linking system with the GPS to locate the bag in the worst conditions. To sense the battery level there is need for mechanism to sense the voltage level of battery. In addition, a solar cell can be integrated to the system for powering unit.

REFERENCES

[1] V. K., Sadagopan, A. J., Francis and U. Rajendran, "Anti-theft Control System Design Using Embedded System", *In the proceeding of IEEE International Conference Vehicular Electronics and Safety* (ICVES), pp.1-5, 2011.

[2] M. N., Ramadan, M. A., Al-Khedher, Senior Member, IACSIT, and S. A., Al-Kheder, "Intelligent Anti-Theft and Tracking System for Automobiles", Proceeding of International Journal of Machine Learning and Computing, Vol. 2, No. 1, pp. 88-92, 2012.

[3] A. S., Dhotre, S., Abhishek, Chandurkar and S. S., Jadhav, "Design of a GSM Cell – Phone based Vehicle Monitoring & Theft Security System", In Proceeding of International Journal of Electrical and Electronics Engineering, Vol. 1, No. 3, pp.101-105, 2012.

[4] A., Abukovich, C., Castillo, C., Mitchell, A., Tomicic, and S., Elkhammas, "Wireless Animal Containment", Report for Texas Instrument (TI) Competition, The Ohio State University, 2009.

5] B., Kodavati, et al., “GSM and GPS based vehicle location and tracking system”, International Journal of Engineering Research and Applications (IJERA), pp. 2248-9622, 2011.

[6] M. A., Al-Khedher, "Hybrid GPS-GSM Localization of Automobile Tracking System", Proceeding of International Journal of Computer Science & Information Technology, Vol. 3 , No. 6, pp.75-85, 2011.

[7] S., Hwang and D. Yu., "Remote Monitoring and Controlling System Based on ZigBee Networks", Proceeding of International Journal of Software Engineering and Its Applications, Vol. 6,No. 3, pp. 35-42, 2012.

[8] T. T., Tushar , S. S., Harshad, R. S., Chaitanya , V. O., Priyanka and B. M., Vikas, “Online Student Monitoring System Using Passive RFID “, proceeding

of international Journal of Innovative Research in Computer and Communication Engineering, Vol. 1, Issue 2, pp. 409-1027, 2013.

[9] M. H., Abdul Kadir, T., Zarina and A., Ariffin, "Tracking Student Movement using Active RFID herd Await", ISSN: 1790-5117 ,MALAYSIA.

[10] K., Benkic, M., Malajner, P., Planinsic and Z., Cucej, "Using RSSI Value for Distance Estimation in Wireless Sensor Networks Based on ZigBee", SPaRC Laboratory, UM-FERI Maribor, Smetanova 17, Maribor, Slovenia, 2000.

[11] M., Srbinovska, C., Gavrovski, V., Dimcev, "Localization Estimation System Using Measurement of RSSI Based on ZIGBEE Standard'', Electronics, Sozopol, BULGARIA, pp. 46, 2008.

[12] R., Faludi , "Building Wireless Sensors Networks", USA, 2011.

[13] M., Hebel, G., Bricker, D., Harris, "A tutorial, Getting Start with Xbee RF Modules ", Parallax Inc , 2010.

[14] IEEE Computer Science: Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specification for low-rate Wireless Personal Area Networks (WPANs), 2007.

[15] G., horvat, D., Sostaric, and D., Zagar, "Power Consumption and RF Propagation Analysis on ZigBee XBee Modules for ATPC", IEEE, 2012.

A. APPENDIX A

BASCOM Programmes

This programme is trying to Simulate XBee modules as serial communication for transmit unit and receiver unit

Transmitter unit:

```
'*******************************
'name                   : Simulate XBee modules as serial communication
'unit                   : transmitter
'purpose                : code shows the difference between 3 zones acordding distance
'micro                  : attiny2313
'*******************************
$regfile "attiny2313.dat"
$crystal = 1000000                    'for run UART with eternal 8mhz cannot be use with
this micro
Config Com1 = 1200 , Parity = None , Stopbits = 1 , Databits = 8
Config Porta = Input
Config Portb = Output
Dim Var1 As Byte
Testmode:
If Pina.2 = 0 Then                            'swich to control of sleep mode
Goto Startsystem                              'systm be run when Pina.2 = 0 (=pin 9 in Xbee
module)
Wait 1
End If
Startsystem:
Do
Portb.5 = 0
Input " *** Test Serial Port *** "
Print "Test RSSI Valu "
If Ischarwaiting() = 1 Then
Var1 = Inkey()
Print Var1
Wait 1
End If
Speaker Alias Portb.0
If Var1 = 52 Then
Print "4"                          '52=4
Portb.1 = 0
Portb.2 = 1
Waitms 500
Portb.5 = 1
Waitms 1000
Portb.5 = 0
```

```
Waitms 1000
Portb.5 = 1
Sound Speaker , 1000 , 50
Wait 1
Sound Speaker , 1000 , 50
Wait 1
Sound Speaker , 1000 , 50
 '***************************************************************************"
End If
If Var1 = 51 Then                              '51=3
Print "3"
Portb.1 = 0
Portb.2 = 1
Waitms 500
Portb.5 = 0
Sound Speaker , 100 , 50
Wait 1
Sound Speaker , 100 , 50
Wait 1
Sound Speaker , 100 , 50
Wait 1
Sound Speaker , 100 , 50
End If
If Var1 = 50 Then                              '45=2
Print "2"
Portb.1 = 1
Waitms 500
Portb.1 = 0
Waitms 1000
Portb.1 = 1
Portb.2 = 0
Waitms 500
Portb.5 = 0
Sound Speaker , 1000 , 50
Waitms 50
Sound Speaker , 1000 , 50
Waitms 50
Sound Speaker , 1000 , 50
Waitms 50
Sound Speaker , 1000 , 50
End If
If Var1 = 49 Then
Print "1"
Portb.0 = 0
Portb.1 = 1
Portb.2 = 0
Portb.5 = 0
End If
```

```
Goto Testmode
Loop
```

Receiver unit:

```
'*******************************
'name                  : Simulate XBee modules as serial communication
'unit                 : receiver
'purpose               : code shows the difference between 3 zones according distance
'micro                 : attiny2313
'*******************************
$regfile "attiny2313.dat"
$crystal = 1000000                     'for run UART with internal 8mhz cannot be use
with this micro
Config Com1 = 1200 , Parity = None , Stopbits = 1 , Databits = 8
Config Portb = Output
Dim N As Byte
Dim Var1 As Byte
Do
Portb.5 = 0
Input " *** Test Serial Port *** "
Print "Test RSSI Value"
If Ischarwaiting() = 1 Then
Var1 = Inkey()
Print Var1
Wait 1
End If
Speaker Alias Portb.0                        'aduio alarming
If Var1 > 49 Then
Portb.1 = 1                            ' visual alarming Red LED
End If
If Var1 = 52 Then
Print "4"                              'Ascii code of 4=52 (zone 3=5m)
Sound Speaker , 1000 , 50
Wait 1
Sound Speaker , 1000 , 50
Wait 1
Sound Speaker , 1000 , 50
End If
'*****************************************************************
If Var1 = 51 Then                           'Ascii code of 3=51 (zone 2=2m)
Print "3"
Sound Speaker , 100 , 50
Wait 1
Sound Speaker , 100 , 50
Wait 1
Sound Speaker , 100 , 50
Wait 1
Sound Speaker , 100 , 50
```

```
End If
'******************************************************************
If Var1 = 50 Then                                'Ascii code of 2=45 (zone 1=1m)
Print "2"
Sound Speaker , 1000 , 50
Waitms 50
Sound Speaker , 1000 , 50
Waitms 50
Sound Speaker , 1000 , 50
Waitms 50
Sound Speaker , 1000 , 50
End If
If Var1 <= 49 Then
Print "1"
Portb.0 = 0
Portb.1 = 0
End If
Loop
```

B. APPENDIX B

Arduino Programme for the System

The following code for transmitter unit (with the owner)

```
int digitalpin=11;
double rssiDur;
double rssiDur0;
double rssiDur1;
double rssiDur2;
double rssiDur3;
double rssiDur4;
double rssiDur5;
double rssiDur6;
double rssiDur7;
double rssiDur8;
double rssiDur9;
double rssiDurF;
int led=12;
double  a ;
int  a1 ;
int  a2 ;
int  a3 ;
int  a4 ;
int  a5 ;
int  a6 ;
int  a7 ;
int  a8 ;
int b ;    int c  ;
int L  ;  int z  ;
int x  ;
double k;
int led1=8;
```

```
int led4=9;
int led6=10;
int counterloop;
double distanseF;double distanse1;double distanse2;int mmmm;
double y1;double y2;double y3;double y4;double y5;double y6;double y7;double y8;double
y9;double y10;
void setup() {
Serial.begin(9600);
pinMode(led,OUTPUT);
pinMode(led1,OUTPUT);
pinMode(led4,OUTPUT);
pinMode(led6,OUTPUT);
pinMode(digitalpin,INPUT);
}
//*******************START PROGRAM**************************************
void loop() {
bigen:

 if(Serial.available() >= 21) {
if(Serial.read() == 0x7E){
  for(int i=1; i<19; i++){
for (int jk=0; jk<2; ++jk){
   for(int j=0; j<2; ++j){
getdata:
if(distanse1>0 || mmmm==1 ){
  jk=1;
}
       for(int x=0; x<11; ++x){
     if (x=1){
 byte discardbyte =Serial.read();
      rssiDur =pulseIn(digitalpin, LOW, 2000);
   rssiDur0=rssiDur;
```

```
  Serial.println(rssiDur0);
  }

//*************************************
  if (x=2){
    byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur1=rssiDur;
      Serial.println(rssiDur1);
  }

//**************************************
 if (x=3){
   byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur2=rssiDur;
     Serial.println(rssiDur2);
  }

//****************************************
   if (x=4){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur3=rssiDur;
       Serial.println(rssiDur3);
  }

//********************************************
   if (x=5){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur4=rssiDur;
```

```
     Serial.println(rssiDur4);
  }

  //****************************************
  if (x=6){
    byte discardbyte =Serial.read();
    rssiDur =pulseIn(digitalpin, LOW, 2000);
     rssiDur5=rssiDur;
      Serial.println(rssiDur5);
  }
  //****************************************
  if (x=7){
    byte discardbyte =Serial.read();
    rssiDur =pulseIn(digitalpin, LOW, 2000);
     rssiDur6=rssiDur;
      Serial.println(rssiDur6);
  }
  //****************************************
if (x=8){
    byte discardbyte =Serial.read();
    rssiDur =pulseIn(digitalpin, LOW, 2000);
     rssiDur7=rssiDur;
      Serial.println(rssiDur7);
  }
  //****************************************
 if (x=9){
    byte discardbyte =Serial.read();
    rssiDur =pulseIn(digitalpin, LOW, 2000);
     rssiDur8=rssiDur;
      Serial.println(rssiDur8);
  }
  //****************************************
```

```
    if (x=10){
      byte discardbyte =Serial.read();
      rssiDur =pulseIn(digitalpin, LOW, 2000);
       rssiDur9=rssiDur;
        Serial.println(rssiDur9);
    }
  //**************************************************
if (j==0){
rssiDurF=(rssiDur0+rssiDur1+rssiDur2+rssiDur3+rssiDur4+rssiDur5+rssiDur6+rssiDur7+rssi
Dur8+rssiDur9) ;
  j=1;
    goto getdata;
}
  if (j==1){

rssiDur=(rssiDur0+rssiDur1+rssiDur2+rssiDur3+rssiDur4+rssiDur5+rssiDur6+rssiDur7+rssi
Dur8+rssiDur9+rssiDurF)/20 ;
 Serial.println("*********************");
 Serial.println(rssiDur);
  Serial.println("*********************");
    }
}
  }
 if (rssiDur==0){
   digitalWrite(led6,LOW);
digitalWrite(led1,LOW);
  delay(500);
digitalWrite(led4,LOW);
    goto bigen;
 }
if (rssiDur > 0){
    goto xx;
```

```
}
//************END READ DATA AND CALCULATE THE MEEN
VALUE*****************************

xx:
counterloop  = 1+counterloop;
//if (counterloop > 10){
//  counterloop=1;
//}
for (int i=7;i>=0;--i){
  b=rssiDur;
delay(20);
z= pow(2, i)+1;
c=b & z;
//*******************************
if(c==128){
    L=7;
   goto bailout;
   }
if(c==64){
    L=6;
   goto bailout;
   }
if(c==32){
    L=5;
   goto bailout;
   }
if(c==16){
    L=4;
   goto bailout;
   }
if(c==8){
```

```
    L=3;
   goto bailout;
   }
if(c==4){
    L=2;
   goto bailout;
   }
if(c==2){
    L=1;
   goto bailout;
   }
if(c==1){
    L=0;
   goto bailout;
   }
//*******************************
}
bailout:
a1=0;
a2=0;
a3=0;
a4=0;
a5=0;
a6=0;
a7=0;
a8=0;
Serial.println("L=  ");
Serial.println(L);
delay(20);
for (int i=0;i<8;++i){
z=00000001;
int t=z  << i;
```

```
x=b & t;
if(x==1){
  a1=1 ;
  }
if(x==2){
   a2=2 ;
  }
if(x==4){
   a3=4 ;
  }
if(x==8){
    a4=8 ;
  }
if(x==16){
    a5=16 ;
  }
if(x==32){
  a6=32;
  }
if(x==64){
    a7=64 ;
  }
if(x==128){
    a8=128;
  }
}
a=-(1)* pow(2, L)+(a1+a2+a3+a4+a6+a5+a7+a8-(pow(2, L)));
Serial.println(" RSSI dB= ");
Serial.println(a);
delay(20);
k=(-5.18-a)/(10*2.961);
double y=pow(10, k);
```

```
Serial.println(" THE REAL Distance:  " );
Serial.println(y);
Serial.println("**********************");
if (counterloop ==1){
  y1=y;
   goto bigen;
}
if (counterloop ==2){
  y2=y;


}
if (counterloop >=3){
  y3=y;
}
//if (counterloop ==4){
//  y4=y;
//   goto bigen;
//}
//if (counterloop ==5){
//  y5=y;
//   goto bigen;
//}
//if (counterloop ==6){
//  y6=y;
//   goto bigen;
//}
//if (counterloop ==7){
//  y7=y;
//   goto bigen;
//}
//if (counterloop ==8){
//  y8=y;
```

```
//   goto bigen;
//}
//if (counterloop ==9){
//  y9=y;
//   goto bigen;
//}
//if (counterloop ==10){
//  y10=y;
//}
if (counterloop ==2 && jk ==0 ){
  distanse1=(y1+y2)/2;
jk=1;
mmmm=1;
distanseF=distanse1;
 goto compare;
 }
if (counterloop >=3 && jk==1){
  distanse2=y3;
Serial.println("*********************");
 Serial.println(distanseF);
 Serial.println(distanse2);
distanseF=(distanseF+distanse2)/2;
 Serial.println(distanseF);
Serial.println("*********************");
  goto compare;
 }
}
//*********************************************
compare:
if(distanseF >= 0 && distanseF <=1){
digitalWrite(led,HIGH);
  delay(500);
```

```
  digitalWrite(led6,HIGH);
digitalWrite(led1,LOW);
  delay(500);
digitalWrite(led4,LOW);
  delay(500);
  }
//*******************************************
if(distanseF > 1 && distanseF <=2){
  digitalWrite(led,HIGH);
   for(int toon=0;toon<4; ++toon){
tooo:
  tone(6, 1000, 100);
  delay(2000);
  // tone(6, 0, 0);
if(toon ==3){
  goto led2;
}
  toon=++ toon;
  goto tooo;
  }
led2:
 digitalWrite(led6,HIGH);
  delay(500);
digitalWrite(led4,HIGH);
  delay(500);
digitalWrite(led1,LOW);
 delay(500);
 }
//*********************************************
if(distanseF > 2 && distanseF <=5){
   Serial.println("444444444444444444444444");
  digitalWrite(led,HIGH);
```

```
  for(int toon=0;toon<4; ++toon){
too:
 tone(6, 1000, 1000);
 delay(5000);
// tone(6, 0, 0);
if(toon ==3){
 goto led1;
}
 toon=++ toon;
 goto too;
 }
led1:
digitalWrite(led1,LOW);
 delay(500);
digitalWrite(led4,HIGH);
 delay(500);
digitalWrite(led6,LOW);
 delay(500);
}
//************************************************
if(distanseF >5){
  Serial.println("w55555555555555555555t");
 digitalWrite(led,HIGH);
 for(int toon=0;toon<16; ++toon){
to:
  tone(6, 1000, 100);
 delay(250);
   //tone(6, 0, 0);
if(toon ==15){
 goto leds;
}
 toon=++ toon;
```

```
  goto to;
  }
leds:
  digitalWrite(led6,LOW);
  delay(500);
digitalWrite(led4,LOW);
  delay(500);
digitalWrite(led1,HIGH);
  delay(500);
  }
  delay(500);
 Serial.println("wait wait wait wait");
// mmmm=0;
 distanse1=0;
    distanse2=0;
 // distanseF=0;
goto bigen;
   }
   }
 }
 }
```

The following code for receiver unit (in the Bag)

```
int digitalpin=11;int led=13;
double rssiDur;double rssiDurF;
double rssiDur0;double rssiDur1;
double rssiDur2;double rssiDur3;
double rssiDur4;double rssiDur5;
double rssiDur6;double rssiDur7;
```

```
double rssiDur8;double rssiDur9;
double  a ;int b ;int c  ;
int  a1 ;int  a2 ;
int  a3 ;int  a4 ;
int  a5 ;int  a6 ;
int  a7 ;int  a8 ;
int L  ;int z  ;
int x  ;double k;
int motorp=8;int motorn=9;
int counterloop;
double distanseF;double distanse1;double distanse2;int mmmm;
double y1;double y2;double y3;double y4;double y5;double y6;double y7;double y8;double y9;double y10;
void setup() {
Serial.begin(9600);
pinMode(led,OUTPUT);
pinMode(motorp,OUTPUT);
pinMode(motorn,OUTPUT);
pinMode(digitalpin,INPUT);
}
//******************START PROGRAM*************************************
void loop() {
bigen:
 if(Serial.available() >= 21) {
if(Serial.read() == 0x7E){
  for(int i=1; i<19; i++){
```

```
for (int jk=0; jk<2; ++jk){
   for(int j=0; j<2; ++j){
getdata:
if(distanse1>0 || mmmm==1 ){
  jk=1;
}
       for(int x=0; x<11; ++x){
     if (x=1){
 byte discardbyte =Serial.read();
      rssiDur =pulseIn(digitalpin, LOW, 2000);
   rssiDur0=rssiDur;
   Serial.println(rssiDur0);
   }
//***********************************
   if (x=2){
     byte discardbyte =Serial.read();
      rssiDur =pulseIn(digitalpin, LOW, 2000);
       rssiDur1=rssiDur;
       Serial.println(rssiDur1);
   }
//************************************
  if (x=3){
    byte discardbyte =Serial.read();
      rssiDur =pulseIn(digitalpin, LOW, 2000);
       rssiDur2=rssiDur;
      Serial.println(rssiDur2);
```

```
  }
//****************************************
   if (x=4){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur3=rssiDur;
       Serial.println(rssiDur3);
  }
//********************************************
   if (x=5){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur4=rssiDur;
       Serial.println(rssiDur4);
  }
  //****************************************
  if (x=6){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur5=rssiDur;
       Serial.println(rssiDur5);
  }
  //****************************************
  if (x=7){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
```

```
      rssiDur6=rssiDur;
       Serial.println(rssiDur6);
  }
  //****************************************
 if (x=8){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur7=rssiDur;
       Serial.println(rssiDur7);
  }
  //****************************************
  if (x=9){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur8=rssiDur;
       Serial.println(rssiDur8);
  }
  //****************************************
   if (x=10){
     byte discardbyte =Serial.read();
     rssiDur =pulseIn(digitalpin, LOW, 2000);
      rssiDur9=rssiDur;
       Serial.println(rssiDur9);
  }
  //*************************************************
if (j==0){
```

```
rssiDurF=(rssiDur0+rssiDur1+rssiDur2+rssiDur3+rssiDur4+rssiDur5+rssiDur6+rssiDur7+rssiDur8+rssiDur9) ;
  j=1;
    goto getdata;
}
  if (j==1){

rssiDur=(rssiDur0+rssiDur1+rssiDur2+rssiDur3+rssiDur4+rssiDur5+rssiDur6+rssiDur7+rssiDur8+rssiDur9+rssiDurF)/20 ;
 Serial.println("********************");
 Serial.println(rssiDur);
  Serial.println("********************");
   }
}
  }
 if (rssiDur==0){
   digitalWrite(motorp,HIGH);
digitalWrite(motorn,LOW);
  delay(2000);
  digitalWrite(motorp,LOW);
digitalWrite(motorn,LOW);
  delay(1000);
    goto bigen;
 }
if (rssiDur > 0){
    goto xx;
}
```

```
//************END READ DATA AND CALCULATE THE MEEN VALUE*****************************

xx:

counterloop  = 1+counterloop;

for (int i=7;i>=0;--i){

  b=rssiDur;

delay(20);

z= pow(2, i)+1;

c=b & z;

//*******************************

if(c==128){

    L=7;

   goto bailout;

   }

if(c==64){

    L=6;

   goto bailout;

   }

if(c==32){

    L=5;

   goto bailout;

   }

if(c==16){

    L=4;

   goto bailout;

   }
```

```
if(c==8){
    L=3;
   goto bailout;
   }
if(c==4){
    L=2;
   goto bailout;
   }
if(c==2){
    L=1;
   goto bailout;
   }
if(c==1){
    L=0;
   goto bailout;
   }
//*******************************
}
bailout:
a1=0;
a2=0;
a3=0;
a4=0;
a5=0;
a6=0;
a7=0;
```

```
a8=0;
Serial.println("L=  ");
Serial.println(L);
delay(20);
for (int i=0;i<8;++i){
z=00000001;
int t=z  << i;
x=b & t;
if(x==1){
   a1=1 ;
   }
if(x==2){
    a2=2 ;
   }
if(x==4){
    a3=4 ;
   }
if(x==8){
     a4=8 ;
   }
if(x==16){
     a5=16 ;
   }
if(x==32){
   a6=32;
   }
```

```
if(x==64){
    a7=64 ;
  }
if(x==128){
    a8=128;
  }
}
a=-(1)* pow(2, L)+(a1+a2+a3+a4+a6+a5+a7+a8-(pow(2, L)));
Serial.println(" RSSI dB= ");
Serial.println(a);
delay(20);
k=(-5.18-a)/(10*2.961);
double y=pow(10, k);
Serial.println(" THE REAL Distance:  " );
Serial.println(y);
Serial.println("**********************");
if (counterloop ==1){
  y1=y;
   goto bigen;
}
if (counterloop ==2){
  y2=y;
}
if (counterloop >=3){
  y3=y;
}
```

```
if (counterloop ==2 && jk ==0 ){
  distanse1=(y1+y2)/2;
jk=1;
mmmm=1;
distanseF=distanse1;
 goto compare;
 }
if (counterloop >=3 && jk==1){
  distanse2=y3;
Serial.println("********************");
 Serial.println(distanseF);
 Serial.println(distanse2);
distanseF=(distanseF+distanse2)/2;
 Serial.println(distanseF);
Serial.println("********************");
  goto compare;
 }
}
//**********************************************
compare:
if(distanseF >= 0 && distanseF <=1){
digitalWrite(led,HIGH);
  delay(500);
digitalWrite(motorp,HIGH);
digitalWrite(motorn,LOW);
  delay(2000);
```

```
  digitalWrite(motorp,LOW);
digitalWrite(motorn,LOW);
  delay(1000);
  }
//*********************************************
if(distanseF > 1 && distanseF <=2){
  digitalWrite(led,HIGH);
   for(int toon=0;toon<4; ++toon){
tooo:
  tone(6, 1000, 100);
  delay(2000);
   //tone(6, 0, 0);
if(toon ==3){
  goto led2;
}
  toon=++ toon;
  goto tooo;
  }
led2:
digitalWrite(motorp,LOW);
digitalWrite(motorn,HIGH);
 delay(9000);
 digitalWrite(motorp,LOW);
digitalWrite(motorn,LOW);
  delay(1000);
 }
```

```
//*********************************************
if(distanseF > 2 && distanseF <=5){
   Serial.println("4444444444444444444444444");


  digitalWrite(led,HIGH);
    for(int toon=0;toon<4; ++toon){
too:
  tone(6, 1000, 1000);
  delay(5000);
  //tone(6, 0, 0);
if(toon ==3){
  goto led1;
}
  toon=++ toon;
  goto too;
  }
led1:
digitalWrite(motorp,LOW);
digitalWrite(motorn,HIGH);
 delay(9000);
 digitalWrite(motorp,LOW);
digitalWrite(motorn,LOW);
  delay(1000);
 }
//*************************************************
if(distanseF >5){
```

```
   Serial.println("w55555555555555555555555t");
  digitalWrite(led,HIGH);
  for(int toon=0;toon<16; ++toon){
to:
    tone(6, 1000, 100);
  delay(250);
     // tone(6, 0, 0);
if(toon ==15){
  goto leds;
}
  toon=++ toon;
  goto to;
  }
leds:
  digitalWrite(motorp,LOW);
digitalWrite(motorn,HIGH);
 delay(9000);
 digitalWrite(motorp,LOW);
digitalWrite(motorn,LOW);
  delay(1000);
  }
  delay(500);
 Serial.println("wait wait wait wait");
 distanse1=0;
    distanse2=0;
goto bigen;
```

```
    }
    }
  }
  }
```

C. APPENDIX C

MATLAB Programme

This programme is trying to study the measurements of the SRRI for Zigbee modules in 20 meters

```
clear;
clc;
rssi_1m=[24,20,18,14,18,10,4,11,6,6,15,6,6,6,4,8,12,3,6,3,6,8,12,2,2,4,8,15,8,8,6,8,6,4,0,2,3,6
,6,8,8,6,3,6,8,3,2,6,6,6,6,8,8,11,12,6,2,1,1,0,6,6,6,6,11,11,3,8,8,6,6,6,10,11,8,6,6,6,10,12,11,4,
4,6,8,3,6,6,6,12,2,2,6,6,8,3,6,15,3,8,12,4,6,11,6,8,10,6,6,7,6,8,6,8,6,6,6,8,8,6,6,8,6,6,6,6,3,8,4,
6,6,6,6,3,6,6,3,6,6,8,6,6,6,6,6,8,8,8,6,6,8,8,6,10,8,6,6,8,8,3,12,8,6,6,6,6,6,6,4,8,10,8,3,6,4,6,6,6
,6,8,6,6,8,8,6,6,6,6,4,3,0,2,0,1,0,4,0,1,2,0];

RSSI_1m_dBm=zeros(1,length(rssi_1m));
for i=1:length(rssi_1m)
 if rssi_1m(i)==0
RSSI_1m_dBm(1,i)=0;
else
B = bitget(rssi_1m(i), 8:-1:1);
l=length(B);
indx=find(B);
 B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_1m_dBm(1,i)=y;
end
end
%---------------------------------------------------------------------
rssi_2m=[29,66,44,39,53,50,53,50,56,53,43,60,57,53,53,50,53,57,53,56,54,53,58,46,46,50,52,
60,57,50,50,50,48,45,50,50,52,48,50,49,53,50,54,57,53,60,52,62,57,57,60,56,57,57,60,74,60,6
5,53,54,57,50,54,56,53,50,56,57,57,57,56,48,60,54,54,57,56,78,66,52,54,56,48,57,45,46,48,50
,56,53,52,50,48,58,46,58,57,50,48,48,46,48,50,57,50,50,50,50,50,50,50,50,50,50,50,53,61,85,
50,48,45,52,50,53,50,50,53,54,41,48,46,52,46,46,50,50,50,54,48,50,54,43,83,48,41,50,48,44,5
7,33,0,48,53,50,50,53,41,54,50,60,54,45,50,48,50,83,50,56,50,54,65,44,53,60,50,50,45,53,54,
50,50,41,41,50,50,53,53,53,41,53,52,52,48,46,52,50,48,50,53,50];

RSSI_2m_dBm=zeros(1,length(rssi_2m));
for i=1:length(rssi_2m)
if rssi_2m(i)==0
 RSSI_2m_dBm(1,i)=0;
else
 B = bitget(rssi_2m(i), 8:-1:1);
```

```
l=length(B);
indx=find(B);
 B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
 for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_2m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_3m=[56,48,46,46,43,45,41,48,48,44,45,45,48,44,45,48,44,45,49,48,48,41,44,44,48,45,41,
50,46,46,46,50,48,41,45,48,44,44,41,39,48,44,44,48,48,45,45,44,44,48,44,44,48,57,48,48,53,5
0,48,48,44,46,45,48,41,50,46,60,48,45,50,36,53,41,44,44,40,45,41,41,41,44,44,43,45,45,44,45
,44,45,44,44,46,44,50,45,53,45,60,78,40,44,48,48,41,48,43,44,48,45,57,48,46,48,48,45,46,46,
45,45,53,41,44,41,39,48,54,45,48,36,48,48,48,48,50,41,56,48,56,67,52,48,41,48,48,48,64,41,5
2,48,48,46,48,44,44,54,53,48,48,46,48,46,45,48,48,45,48,44,36,45,44,48,45,48,48,48,48,46,48
,48,46,48,48,48,44,44,48,57,45,45,48,45,45,48,45,48,43,45,48,46];

RSSI_3m_dBm=zeros(1,length(rssi_3m));
for i=1:length(rssi_3m)
if rssi_3m(i)==0
RSSI_3m_dBm(1,i)=0;
else
B = bitget(rssi_3m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_3m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_4m=[56,48,62,53,45,54,53,53,56,67,74,57,53,50,48,53,53,57,54,54,50,54,54,53,54,53,50,
53,50,53,53,53,54,54,54,66,53,53,53,50,45,48,48,53,46,48,60,41,48,53,53,53,50,48,49,53,53,5
6,57,54,53,57,53,53,54,56,53,54,57,50,54,50,52,50,54,54,48,48,0,56,52,53,62,50,49,54,53,54,
60,46,39,60,50,50,56,53,57,53,57,54,57,57,56,57,56,48,53,57,54,54,57,54,86,50,57,57,48,50,5
4,41,57,53,54,54,50,46,57,57,53,54,54,57,54,53,65,44,57,53,48,57,50,54,53,50,54,50,65,45,48
,50,60,43,48,43,50,66,62,53,54,57,50,54,53,66,48,52,60,53,65,50,50,54,50,50,60,54,52,57,48,
50,53,45,60,57,50,53,48,60,56,56,75,44,48,54,53,53,53,54,54,53];

RSSI_4m_dBm=zeros(1,length(rssi_4m));
for i=1:length(rssi_4m)
```

```
 if rssi_4m(i)==0
RSSI_4m_dBm(1,i)=0;
else
B = bitget(rssi_4m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
 RSSI_4m_dBm(1,i)=y;
end
end
%------------------------------------------------------------------------
rssi_5m=[48,60,65,81,62,50,58,54,56,50,50,56,86,54,54,57,57,66,53,57,56,60,50,65,53,71,54,
60,48,54,99,65,57,57,54,60,60,62,90,57,50,50,56,60,66,57,48,50,54,57,57,57,48,53,54,53,54,6
2,54,49,53,52,54,52,53,54,50,50,53,56,54,56,60,54,57,87,48,52,46,50,57,50,73,60,57,57,60,60
,57,60,57,56,48,53,57,57,62,74,46,53,90,60,56,52,54,57,50,53,48,50,56,57,54,54,53,56,60,57,
53,57,54,60,60,57,57,56,57,56,53,57,50,50,57,56,57,60,54,57,60,57,65,62,62,57,60,57,54,57,6
0,54,52,57,53,60,57,60,57,60,57,50,56,53,56,50,66,91,57,57,57,54,54,50,49,53,48,56,60,57,56
,53,53,57,74,83,54,57,57,60,54,57,57,57,57,50,56,56,57,60,60,56];

RSSI_5m_dBm=zeros(1,length(rssi_5m));
for i=1:length(rssi_5m)
if rssi_5m(i)==0
RSSI_5m_dBm(1,i)=0;
else
B = bitget(rssi_5m(i), 8:-1:1);
 l=length(B);
 indx=find(B);
B=B(indx(1):l);
 l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_5m_dBm(1,i)=y;
 end
end
%------------------------------------------------------------------------
rssi_6m=[48,60,65,81,62,50,58,54,56,50,50,56,86,54,54,57,57,66,53,57,56,60,50,65,53,71,54,
60,48,54,99,65,57,57,54,60,60,62,90,57,50,50,56,60,66,57,48,50,54,57,57,57,48,53,54,53,54,6
2,54,49,53,52,54,52,53,54,50,50,53,56,54,56,60,54,57,87,48,52,46,50,57,50,73,60,57,57,60,60
,57,60,57,56,48,53,57,57,62,74,46,53,90,60,56,52,54,57,50,53,48,50,56,57,54,54,53,56,60,57,
53,57,54,60,60,57,57,56,57,56,53,57,50,50,57,56,57,60,54,57,60,57,65,62,62,57,60,57,54,57,6
```

```
0,54,52,57,53,60,57,60,57,60,57,50,56,53,56,50,66,91,57,57,57,54,54,50,49,53,48,56,60,57,56
,53,53,57,74,83,54,57,57,60,54,57,57,57,57,50,56,56,57,60,60,56];

RSSI_6m_dBm=zeros(1,length(rssi_6m));
for i=1:length(rssi_6m)
if rssi_6m(i)==0
RSSI_6m_dBm(1,i)=0;
else
B = bitget(rssi_6m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_6m_dBm(1,i)=y;
end
end
%---------------------------------------------------------------------------
rssi_7m=[69,71,69,69,71,77,66,60,69,66,73,69,69,66,69,71,65,91,58,90,69,69,66,66,65,65,66,
65,65,78,74,69,67,65,66,71,71,65,69,66,69,69,66,65,66,83,81,66,64,69,65,69,65,65,65,67,67,6
9,66,69,69,81,67,69,60,64,66,66,62,81,69,78,69,66,66,66,66,65,71,69,71,74,64,71,67,65,65,66
,66,71,74,102,69,66,66,69,64,66,78,66,65,65,65,66,69,69,69,70,65,66,67,77,74,66,69,66,65,74
,56,65,65,57,58,66,69,62,81,75,74,81,78,65,69,73,79,70,77,69,75,74,78,65,66,65,65,60,62,62,
62,74,65,66,78,71,74,70,74,71,66,62,62,62,62,62,58,60,71,74,65,66,65,65,57,62,65,78,66,66,6
6,66,66,57,66,65,66,65,65,65,65,58,57,60,60,65,69,62,65,62,69,65];

RSSI_7m_dBm=zeros(1,length(rssi_7m));
for i=1:length(rssi_7m)
   if rssi_7m(i)==0
      RSSI_7m_dBm(1,i)=0;
   else
      B = bitget(rssi_7m(i), 8:-1:1);
      l=length(B);
      indx=find(B);
      B=B(indx(1):l);
      l=length(B);
      y=-B(1)*2^(l-1);
      for e=2:l
         y=y+B(e)*2^(l-e);
      end
      RSSI_7m_dBm(1,i)=y;
   end
end
%---------------------------------------------------------------------------
```

```
rssi_8m=[85,94,64,73,75,60,60,71,64,83,60,62,79,95,104,85,69,69,77,66,71,75,56,69,71,71,7
1,71,81,69,86,81,96,96,74,77,65,81,71,71,75,78,85,81,81,66,71,71,75,73,74,77,78,75,83,94,81
,78,64,74,81,81,81,75,75,71,74,62,77,66,75,71,98,75,71,64,81,95,81,88,71,74,74,75,75,67,90,
65,71,75,71,61,66,83,81,79,87,77,73,71,83,128,71,85,77,83,77,83,81,81,64,111,71,81,83,78,8
1,75,87,92,86,69,75,88,104,90,79,88,87,85,102,107,83,81,81,92,90,92,78,81,78,79,81,86,81,8
6,108,73,66,107,74,73,73,81,78,75,81,77,78,74,78,74,81,73,73,74,71,100,71,74,74,81,79,81,8
3,90,92,82,86,87,78,88,83,83,86,90,75,74,73,79,88,90,78,70,81,79,81,73,77,99];

RSSI_8m_dBm=zeros(1,length(rssi_8m));
for i=1:length(rssi_8m)
if rssi_8m(i)==0
RSSI_8m_dBm(1,i)=0;
 else
B = bitget(rssi_8m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
 y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
 RSSI_8m_dBm(1,i)=y;
 end
end
%-----------------------------------------------------------------------
rssi_9m=[65,71,66,62,60,69,66,87,74,73,67,71,74,65,71,69,65,67,69,65,71,69,75,73,65,75,67,
71,73,75,74,71,75,75,67,75,65,73,67,71,71,66,67,71,83,66,99,69,71,71,81,71,71,71,73,71,65,6
9,74,71,67,69,67,74,74,73,75,91,74,75,74,73,69,69,71,71,69,71,75,71,69,74,85,66,98,65,79,83
,62,71,66,69,69,74,74,77,70,81,75,70,71,73,70,75,71,54,65,71,79,79,70,78,70,71,75,65,73,74,
74,74,74,74,67,71,74,74,107,92,65,69,67,71,66,75,71,71,65,71,71,71,62,69,70,69,69,74,71,71,
64,71,69,71,71,71,79,73,71,75,74,71,74,113,74,71,78,74,65,88,78,81,86,125,108,107,104,102,
94,77,99,64,74,73,69,74,71,75,71,71,66,75,67,71,71,71,78,74,71,83,66,94];

RSSI_9m_dBm=zeros(1,length(rssi_9m));
for i=1:length(rssi_9m)
if rssi_9m(i)==0
RSSI_9m_dBm(1,i)=0;
 else
B = bitget(rssi_9m(i), 8:-1:1);
 l=length(B);
 indx=find(B);
 B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
```

```
RSSI_9m_dBm(1,i)=y;
 end
end
%-------------------------------------------------------------------------
rssi_10m=[96,70,85,86,73,79,81,77,81,74,78,81,81,85,81,71,74,83,81,90,75,81,83,74,83,81,7
9,86,85,86,86,81,81,71,96,83,87,86,87,92,79,90,79,86,73,79,78,79,92,74,74,78,79,88,90,83,92
,78,77,83,83,78,79,85,75,81,73,81,74,81,94,85,78,81,70,74,74,86,81,81,79,81,75,81,75,74,70,
69,79,67,69,75,71,69,67,83,99,75,83,78,79,67,74,75,62,74,78,74,74,75,74,74,78,77,73,78,78,7
5,75,74,86,66,116,90,83,65,78,69,66,75,78,75,74,74,73,69,81,75,74,71,79,85,74,90,66,81,98,9
4,86,81,94,87,85,104,90,108,86,81,78,69,83,83,83,78,102,88,90,86,102,102,102,99,79,106,10
2,87,96,102,92,96,70,107,87,75,81,81,73,83,78,71,103,96,77,81,81,87,92,79,82,90];

RSSI_10m_dBm=zeros(1,length(rssi_10m));
for i=1:length(rssi_10m)
if rssi_10m(i)==0
RSSI_10m_dBm(1,i)=0;
 else
B = bitget(rssi_10m(i), 8:-1:1);
  l=length(B);
indx=find(B);
B=B(indx(1):l);
 l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_10m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_11m=[66,67,83,81,92,74,69,66,71,69,78,71,66,73,74,67,71,69,87,99,70,70,70,77,95,88,8
3,88,109,85,92,98,96,82,88,102,90,85,99,99,99,91,96,102,98,100,91,98,83,91,98,91,74,107,98
,86,87,99,119,95,109,99,98,83,99,92,100,98,95,99,95,104,95,107,108,102,88,79,87,88,82,95,9
2,92,96,82,83,90,108,92,95,99,86,90,102,102,88,99,94,91,88,78,79,75,98,95,94,113,96,90,95,
81,88,86,91,102,115,108,100,88,74,86,65,79,74,79,75,83,90,69,75,81,67,71,77,77,90,74,81,71
,70,69,90,88,71,78,102,74,77,78,83,71,71,71,71,67,65,69,66,87,86,78,102,91,82,70,74,91,79,9
2,90,106,119,90,107,88,79,75,78,81,104,79,86,81,81,81,71,70,78,74,85,79,82,92,87,70,92,69,
85,78];

RSSI_11m_dBm=zeros(1,length(rssi_11m));
for i=1:length(rssi_11m)
if rssi_11m(i)==0
RSSI_11m_dBm(1,i)=0;
else
B = bitget(rssi_11m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
```

```
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_11m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_12m=[113,120,82,112,117,91,107,90,113,99,108,123,113,104,115,104,107,100,102,92,1
20,99,99,98,95,95,88,109,107,98,116,88,100,87,92,81,92,77,113,99,107,99,104,108,99,109,10
3,113,83,108,112,112,95,111,109,129,102,125,98,91,92,88,98,100,113,106,116,106,100,100,1
11,92,92,91,92,95,81,119,124,112,129,108,117,109,104,106,102,98,100,113,107,99,111,116,1
25,107,123,92,119,111,115,107,119,119,112,106,117,125,113,120,127,100,104,96,98,98,104,
92,104,106,119,102,94,92,116,113,99,116,102,115,120,99,123,111,79,99,116,112,104,111,11
6,120,116,109,88,119,103,104,85,87,87,95,92,113,98,104,99,98,113,98,104,107,100,116,116,
96,102,99,107,99,119,119,98,107,113,120,104,121,100,119,102,107,109,91,98,113,119,113,1
25,119,100,107,98,96,108,107,96,109,117,109];

RSSI_12m_dBm=zeros(1,length(rssi_12m));
for i=1:length(rssi_12m)
if rssi_12m(i)==0
RSSI_12m_dBm(1,i)=0;
else
B = bitget(rssi_12m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_12m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_13m=[121,109,109,116,109,123,128,125,120,100,125,115,91,102,104,100,104,99,95,99,
108,109,123,102,119,99,96,92,116,100,104,108,116,108,107,121,96,123,119,108,116,112,123
,106,111,115,104,107,100,98,123,125,125,111,91,116,91,116,109,106,115,120,113,108,120,1
07,120,125,125,109,116,111,117,115,106,113,119,98,119,111,99,109,95,103,120,123,113,106
,99,108,116,108,111,120,112,123,123,109,125,121,96,125,120,107,117,113,120,119,115,115,
123,100,120,121,117,104,117,108,116,111,116,113,111,109,98,113,116,107,123,120,123,119,
108,103,123,115,125,127,116,120,128,112,123,128,117,116,111,111,112,107,108,123,111,10
2,102,113,104,102,108,109,113,104,102,109,113,112,116,119,117,119,125,120,112,111,111,1
23,98,99,120,117,123,109,120,115,123,100,120,115,113,100,111,116,119,120,95,94,99,125,1
15,116];
```

```
RSSI_13m_dBm=zeros(1,length(rssi_13m));
for i=1:length(rssi_13m)
if rssi_13m(i)==0
 RSSI_13m_dBm(1,i)=0;
else
B = bitget(rssi_13m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_13m_dBm(1,i)=y;
end
end
%------------------------------------------------------------------------
rssi_14m=[90,95,116,92,94,85,94,96,78,91,87,78,90,77,85,99,83,73,78,81,83,82,78,90,75,82,
96,95,87,91,109,91,102,102,102,98,94,125,106,123,120,109,113,108,113,86,95,115,107,113,1
19,104,111,74,98,88,95,107,119,92,102,81,92,87,79,99,98,86,125,116,113,113,107,119,115,1
16,96,94,88,95,92,92,85,102,95,92,83,95,98,99,91,107,108,91,113,83,99,90,99,119,99,88,92,1
11,92,91,83,83,87,78,86,83,79,85,85,83,85,81,78,81,81,81,78,0,78,81,83,78,79,77,78,81,66,79
,81,79,79,81,75,81,85,81,79,81,79,82,87,99,86,96,83,83,90,83,87,87,88,73,86,83,74,82,85,87,
83,86,77,83,87,90,81,85,83,87,86,78,90,79,94,90,86,86,86,99,81,92,92,74,90,83,73,81,81,83,8
3,87,86,78,92,99];

RSSI_14m_dBm=zeros(1,length(rssi_14m));
for i=1:length(rssi_14m)
if rssi_14m(i)==0
RSSI_14m_dBm(1,i)=0;
else
 B = bitget(rssi_14m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
 y=y+B(e)*2^(l-e);
end
RSSI_14m_dBm(1,i)=y;
end
end
%------------------------------------------------------------------------
rssi_15m=[199,102,85,99,115,95,120,116,102,104,98,108,99,102,92,99,98,90,91,87,102,94,1
02,102,106,104,116,116,99,96,98,99,108,108,113,115,116,111,120,115,128,103,125,112,119,
112,116,100,111,116,111,107,116,117,119,111,108,111,100,116,117,104,111,100,107,107,11
```

```
1,111,115,0,108,117,113,107,99,108,107,106,102,98,107,91,98,108,104,96,102,104,103,108,9
5,100,111,115,113,116,96,108,109,107,109,111,107,125,108,116,96,100,98,102,100,111,0,94,
123,109,125,123,108,123,98,96,108,109,119,111,99,91,109,117,103,99,106,125,123,108,120,
123,95,121,116,119,109,108,115,127,104,100,96,94,104,120,116,121,112,123,109,111,123,10
7,113,96,116,107,119,128,116,115,117,120,113,116,117,116,111,109,102,107,106,120,95,104
,111,103,116,100,120,111,116,107,106,106,96,111,119,113,100,96,103,119];

RSSI_15m_dBm=zeros(1,length(rssi_15m));
for i=1:length(rssi_15m)
if rssi_15m(i)==0
RSSI_15m_dBm(1,i)=0;
else
B = bitget(rssi_15m(i), 8:-1:1);
l=length(B);
indx=find(B);
 B=B(indx(1):l);
 l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
 RSSI_15m_dBm(1,i)=y;
end
end
%--------------------------------------------------------------------------
rssi_16m=[107,95,77,104,123,108,113,103,123,98,111,117,119,109,116,113,120,119,117,116
,123,109,120,109,123,115,108,107,111,107,111,115,125,112,98,109,123,119,117,107,108,95,
107,100,113,119,100,111,107,104,106,111,95,109,125,112,123,119,116,123,120,112,123,109,
121,115,120,120,125,112,108,103,111,106,120,109,116,119,120,115,112,125,115,119,115,11
9,107,119,119,125,109,108,119,115,109,116,112,116,113,109,113,112,0,111,103,116,102,113
,119,113,125,115,119,116,112,113,112,116,113,112,104,112,113,106,107,116,100,113,100,10
7,104,99,104,102,104,94,102,94,104,100,95,123,95,95,88,107,95,87,88,98,100,99,99,92,98,95
,98,99,102,99,99,92,100,99,99,102,96,102,94,94,99,98,99,98,92,99,99,92,102,83,99,94,99,104
,98,102,102,94,102,104,96,99,91,91,102,98,99,102,87,92];

RSSI_16m_dBm=zeros(1,length(rssi_16m));
for i=1:length(rssi_16m)
 if rssi_16m(i)==0
 RSSI_16m_dBm(1,i)=0;
 else
B = bitget(rssi_16m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
 l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
```

```
 end
RSSI_16m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_17m=[86,88,92,92,108,95,83,96,82,94,92,88,81,90,88,95,79,108,67,99,95,83,95,95,86,95
,92,74,113,81,79,102,99,92,86,88,112,108,98,98,107,100,87,82,90,91,85,88,0,90,90,91,86,98,
102,83,92,90,88,95,95,87,88,95,85,88,92,91,92,92,98,102,95,91,98,107,102,104,92,104,96,10
4,98,99,102,99,95,104,94,94,102,91,95,102,96,92,95,91,90,0,104,90,90,90,77,95,95,79,92,95,
87,98,104,88,95,92,90,92,88,96,90,99,92,98,85,102,90,75,74,87,99,83,77,85,0,88,79,83,83,83,
74,92,92,79,90,86,82,86,77,71,83,87,87,95,92,90,85,75,99,81,81,86,81,81,98,92,77,92,90,81,9
0,94,85,95,86,92,92,86,87,83,96,81,92,88,92,90,86,92,88,82,90,90,92,88,90,85,95,92,91,88];

RSSI_17m_dBm=zeros(1,length(rssi_17m));
for i=1:length(rssi_17m)
 if rssi_17m(i)==0
RSSI_17m_dBm(1,i)=0;
else
B = bitget(rssi_17m(i), 8:-1:1);
l=length(B);
 indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_17m_dBm(1,i)=y;
end
end
%-------------------------------------------------------------------------
rssi_18m=[99,104,103,120,98,111,92,106,123,113,96,99,109,108,111,111,107,112,85,100,11
9,108,113,104,109,104,98,98,111,98,107,95,111,108,104,103,102,107,108,99,104,111,99,106,
95,106,104,104,96,102,104,94,111,106,108,109,129,107,113,100,117,109,104,104,109,113,12
0,113,102,111,106,113,108,96,107,100,111,73,85,83,79,81,78,74,83,95,91,74,83,100,95,82,95
,96,104,90,98,78,78,81,79,83,79,86,85,81,81,83,78,81,81,83,73,70,75,90,85,85,81,81,90,104,8
6,87,102,86,90,92,92,107,100,99,85,83,79,85,74,86,86,77,86,75,103,96,82,104,96,92,95,104,7
5,81,83,73,83,86,73,75,92,102,104,81,88,81,81,81,75,83,74,79,81,74,83,81,98,113,113,85,92,
108,103,96,83,95,117,106,83,92,95,95,100,91,85,102,91,90,95,85,86,86];

RSSI_18m_dBm=zeros(1,length(rssi_18m));
for i=1:length(rssi_18m)
if rssi_18m(i)==0
RSSI_18m_dBm(1,i)=0;
else
B = bitget(rssi_18m(i), 8:-1:1);
 l=length(B);
 indx=find(B);
```

```
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_18m_dBm(1,i)=y;
end
end
%------------------------------------------------------------------------
rssi_19m=[99,99,107,95,86,94,95,108,87,77,98,111,104,104,99,104,106,102,100,106,104,100
,102,102,92,116,100,104,103,99,112,112,123,120,120,117,0,0,0,0,0,0,0,98,104,103,109,103,
107,103,103,100,111,104,104,96,103,90,95,86,111,92,94,99,102,94,85,113,103,91,113,107,90
,87,103,113,87,95,111,98,94,107,111,96,87,92,104,102,95,100,102,85,91,98,100,99,100,91,98
,94,91,95,92,91,98,96,98,99,95,95,99,98,99,95,99,95,91,91,90,95,99,87,99,95,77,102,98,77,95
,92,90,95,92,82,92,92,95,92,92,79,90,95,92,92,95,96,95,96,95,94,95,86,95,96,90,85,95,92,87,
88,95,104,87,81,90,98,86,100,102,92,77,102,81,77,86,102,92,99,91,98,99,88,104,100,102,86,
95,95,95,88,96,91,92,96,86,91,95,92,88];

RSSI_19m_dBm=zeros(1,length(rssi_19m));
for i=1:length(rssi_19m)
if rssi_19m(i)==0
RSSI_19m_dBm(1,i)=0;
else
 B = bitget(rssi_19m(i), 8:-1:1);
 l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_19m_dBm(1,i)=y;
end
end
%------------------------------------------------------------------------
rssi_20m=[90,92,95,91,113,102,98,94,96,96,92,92,92,86,92,92,92,92,92,92,91,90,92,82,92,92
,86,92,92,90,87,85,92,90,90,92,86,91,82,91,95,81,99,95,102,104,104,92,95,85,92,87,95,99,95,
83,95,94,91,107,109,102,95,92,92,92,85,92,92,92,96,94,88,91,96,88,95,95,95,82,96,92,90,85,
91,92,87,88,92,90,96,92,92,92,92,86,83,92,96,92,92,90,92,86,92,92,86,83,92,90,85,92,90,92,9
2,95,92,88,88,96,95,96,87,92,92,92,90,92,87,92,90,94,92,91,86,88,91,81,92,92,92,82,92,90,90
,92,88,82,92,92,86,92,92,86,92,92,82,92,90,82,92,90,82,92,87,92,91,92,86,92,90,83,91,90,92,
92,86,87,88,82,92,90,83,92,90,104,113,103,96,100,106,95,106,125,117,117,117,119,113,109]
;

RSSI_20m_dBm=zeros(1,length(rssi_20m));
for i=1:length(rssi_20m)
```

```
if rssi_20m(i)==0
RSSI_20m_dBm(1,i)=0;
 else
B = bitget(rssi_20m(i), 8:-1:1);
l=length(B);
indx=find(B);
B=B(indx(1):l);
l=length(B);
y=-B(1)*2^(l-1);
for e=2:l
y=y+B(e)*2^(l-e);
end
RSSI_20m_dBm(1,i)=y;
end
end
%------------------------------------------------------------------------

%-----------------------study the effect of ---------------------------------------------------
RSSI_dBm_MEAN_5r=[mean(RSSI_1m_dBm(1:5)),mean(RSSI_2m_dBm(1:5)),
mean(RSSI_3m_dBm(1:5)),...
   mean(RSSI_4m_dBm(1:5)), mean(RSSI_5m_dBm(1:5)),mean(RSSI_6m_dBm(1:5)),
mean(RSSI_7m_dBm(1:5)),...
   mean(RSSI_8m_dBm(1:5)), mean(RSSI_9m_dBm(1:5)),mean(RSSI_10m_dBm(1:5)),
mean(RSSI_11m_dBm(1:5)),...
   mean(RSSI_12m_dBm(1:5)),mean(RSSI_13m_dBm(1:5)),mean(RSSI_14m_dBm(1:5)),
mean(RSSI_15m_dBm(1:5)),...
   mean(RSSI_16m_dBm(1:5)), mean(RSSI_17m_dBm(1:5)),mean(RSSI_18m_dBm(1:5)),
mean(RSSI_19m_dBm(1:5)),...
   mean(RSSI_20m_dBm(1:5))];

RSSI_dBm_MEAN_10r=[mean(RSSI_1m_dBm(1:10)),mean(RSSI_2m_dBm(1:10)),
mean(RSSI_3m_dBm(1:10)),...
   mean(RSSI_4m_dBm(1:10)), mean(RSSI_5m_dBm(1:10)),mean(RSSI_6m_dBm(1:10)),
mean(RSSI_7m_dBm(1:10)),...
   mean(RSSI_8m_dBm(1:10)), mean(RSSI_9m_dBm(1:10)),mean(RSSI_10m_dBm(1:10)),
mean(RSSI_11m_dBm(1:10)),...

mean(RSSI_12m_dBm(1:10)),mean(RSSI_13m_dBm(1:10)),mean(RSSI_14m_dBm(1:10)),
mean(RSSI_15m_dBm(1:10)),...
   mean(RSSI_16m_dBm(1:10)),
mean(RSSI_17m_dBm(1:10)),mean(RSSI_18m_dBm(1:10)),
mean(RSSI_19m_dBm(1:10)),...
   mean(RSSI_20m_dBm(1:10))];

RSSI_dBm_MEAN_20r=[mean(RSSI_1m_dBm(1:20)),mean(RSSI_2m_dBm(1:20)),
mean(RSSI_3m_dBm(1:20)),...
   mean(RSSI_4m_dBm(1:20)), mean(RSSI_5m_dBm(1:20)),mean(RSSI_6m_dBm(1:20)),
mean(RSSI_7m_dBm(1:20)),...
```

```
    mean(RSSI_8m_dBm(1:20)), mean(RSSI_9m_dBm(1:20)),mean(RSSI_10m_dBm(1:20)),
mean(RSSI_11m_dBm(1:20)),...

mean(RSSI_12m_dBm(1:20)),mean(RSSI_13m_dBm(1:20)),mean(RSSI_14m_dBm(1:20)),
mean(RSSI_15m_dBm(1:20)),...
    mean(RSSI_16m_dBm(1:20)),
mean(RSSI_17m_dBm(1:20)),mean(RSSI_18m_dBm(1:20)),
mean(RSSI_19m_dBm(1:20)),...
    mean(RSSI_20m_dBm(1:20))];

RSSI_dBm_MEAN_50r=[mean(RSSI_1m_dBm(1:50)),mean(RSSI_2m_dBm(1:50)),
mean(RSSI_3m_dBm(1:50)),...
    mean(RSSI_4m_dBm(1:50)), mean(RSSI_5m_dBm(1:50)),mean(RSSI_6m_dBm(1:50)),
mean(RSSI_7m_dBm(1:50)),...
    mean(RSSI_8m_dBm(1:50)), mean(RSSI_9m_dBm(1:50)),mean(RSSI_10m_dBm(1:50)),
mean(RSSI_11m_dBm(1:50)),...

mean(RSSI_12m_dBm(1:50)),mean(RSSI_13m_dBm(1:50)),mean(RSSI_14m_dBm(1:50)),
mean(RSSI_15m_dBm(1:50)),...
    mean(RSSI_16m_dBm(1:50)),
mean(RSSI_17m_dBm(1:50)),mean(RSSI_18m_dBm(1:50)),
mean(RSSI_19m_dBm(1:50)),...
    mean(RSSI_20m_dBm(1:50))];

RSSI_dBm_MEAN_100r=[mean(RSSI_1m_dBm(1:100)),mean(RSSI_2m_dBm(1:100)),
mean(RSSI_3m_dBm(1:100)),...
    mean(RSSI_4m_dBm(1:100)),
mean(RSSI_5m_dBm(1:100)),mean(RSSI_6m_dBm(1:100)),
mean(RSSI_7m_dBm(1:100)),...
    mean(RSSI_8m_dBm(1:100)),
mean(RSSI_9m_dBm(1:100)),mean(RSSI_10m_dBm(1:100)),
mean(RSSI_11m_dBm(1:100)),...

mean(RSSI_12m_dBm(1:100)),mean(RSSI_13m_dBm(1:100)),mean(RSSI_14m_dBm(1:100)
), mean(RSSI_15m_dBm(1:100)),...
    mean(RSSI_16m_dBm(1:100)),
mean(RSSI_17m_dBm(1:100)),mean(RSSI_18m_dBm(1:100)),
mean(RSSI_19m_dBm(1:100)),...
    mean(RSSI_20m_dBm(1:100))];

RSSI_dBm_MEAN_200r=[mean(RSSI_1m_dBm(1:200)),mean(RSSI_2m_dBm(1:200)),
mean(RSSI_3m_dBm(1:200)),...
    mean(RSSI_4m_dBm(1:200)),
mean(RSSI_5m_dBm(1:200)),mean(RSSI_6m_dBm(1:200)),
mean(RSSI_7m_dBm(1:200)),...
    mean(RSSI_8m_dBm(1:200)),
mean(RSSI_9m_dBm(1:200)),mean(RSSI_10m_dBm(1:200)),
mean(RSSI_11m_dBm(1:200)),...
```

```
mean(RSSI_12m_dBm(1:200)),mean(RSSI_13m_dBm(1:200)),mean(RSSI_14m_dBm(1:200)
), mean(RSSI_15m_dBm(1:200)),...
   mean(RSSI_16m_dBm(1:200)),
mean(RSSI_17m_dBm(1:200)),mean(RSSI_18m_dBm(1:200)),
mean(RSSI_19m_dBm(1:200)),...
   mean(RSSI_20m_dBm(1:200))];

figure
plot(RSSI_dBm_MEAN_5r,'-r')
hold on
plot(RSSI_dBm_MEAN_10r,'--g')
hold on
plot(RSSI_dBm_MEAN_20r,':b')
hold on
plot(RSSI_dBm_MEAN_50r,'-.c')
hold on
plot(RSSI_dBm_MEAN_100r,'-m+')
hold on
plot(RSSI_dBm_MEAN_200r,'--ks')
hold on
```

%This programme is trying to study the measurements of the SRRI for Zigbee modules when using three models by calculate the stander deviation in these models

```
clear
clc
stdDis_poly=[34.7961, 38.0545, 41.1663,44.1317,46.9506];
stdDis_power=[30.6821,37.3697,41.9384,45.5148,48.4976];
% std_poly=0;
% std_power=0;
m=0;
p1 =-0.07325;
p2 =3.478;
p3 =31.39;
%****************************
a = 30.68;
b = 0.2845;
```

```
rssi_1m=[44,45,45,39,44,45,44,45,39,44,39,45,45,44,45,40,44,45,45,44,48,41,48,48,46,46,48,
46,48,48,48,48,45,43,48,48,48,48,40,48,31,29,31,29,22,31,29,29,24,29,29,29,29,29,24,29,24,2
9,29,29,29,31,29,29,31,29,29,29,29,29,29,29,29,29,29,24,27,29,29,29,36,35,36,36,36,35,36,35
,29,36,36,36,35,36,35,36,35,36,36,36,29,29,29,29,24,29,29,31,29,29,29,31,29,29,29,29,29,23,
29,31,33,32,32,32,33,32,32,32,32,32,32,32,33,32,32,32,28,27,32,33,23,23,24,24,24,24,23,24,2
3,24,24,23,24,24,23,24,23,16,24,23,18,18,14,18,18,18,16,18,18,18,18,18,19,18,18,15,18,18,18
,18,28,31,23,29,29,29,29,29,29,29,29,29,31,29,29,29,31,29,29,29,24,24,23,24,24,24,23,24,23,
23,24,22,23,23,23,24,18,22,24,16,29,29,29,29,23,29,31,31,29,29,29,29,29,29,29,29,31,24,31,2
9,29,29,29,29,29,31,23,29,29,29,29,31,31,29,29,29,29,24,27,27,29,29,29,29,29,29,29,29,29,29
,25,31,29,29,24,29,29,29,23,31,33,33,32,32,32,32,33,33,32,32,32,33,33,32,32,33,32,27,32,32,
29,29,29,29,23,31,29,29,29,29,29,29,31,31,29,31,29,24,29,29,27,25,27,25,27,27,25,27,22,27,2
7,27,27,27,29,22,29,29,29,29,27,27,25,27,27,24,27,25,27,27,27,27,27,27,25,25,27,27,27,27,22
,20,22,22,22,20,20,20,20,19,20,20,22,22,20,20,22,20,14,20,25,32,32,28,32,33,32,32,32,33,33,
33,33,32,32,32,32,32,32,32,41,41,41,43,41,41,41,41,41,43,36,41,37,41,41,41,43,43,41,41,48,4
8,48,48,46,48,48,48,48,40,46,48,48,48,46,48,48,46,45,46,50,50,52,50,52,43,50,50,50,50,50,52
,50,45,50,44,50,52,50,50,45,44,44,45,45,45,44,44,44,45,37,44,44,39,45,45,44,37,45,44,48,41,
48,48,48,48,48,46,46,48,48,48,48,48,48,48,48,46,41,48,48,48,48,46,48,41,46,48,48,48,48,46,4
8,48,41,35,41,41,41,41,79,86,87,86,79,87,86,87,87,79,86,82,82,86,86,87,86,86,86,87,65,65,65
,66,61,65,65,65,65,66,66,66,65,65,66,66,65,60,60,66,36,35,29,36,35,36,36,36,36,36,36,36,29,
36,35,35,32,36,35,36,43,41,41,41,37,41,41,43,41,40,41,43,41,43,41,41,39,39,39,37,35,29,36,3
6,36,35,36,36,35,36,36,36,36,35,35,36,35,36,28,36,32,32,32,32,33,32,33,33,27,27,33,32,32,32
,32,32,27,32,28,33,29,31,29,29,31,29,29,29,31,29,31,29,29,29,29,29,29,24,29,29,29,29,29,29,
29,31,29,31,29,29,29,29,29,31,29,29,31,29,29,29,36,35,36,36,36,35,36,36,36,35,36,35,29,36,3
6,36,35,29,36,36,36,28,36,36,36,36,35,36,35,36,35,36,36,32,36,36,35,35,190,35,25,27,27,27,2
4,27,27,27,27,27,27,27,27,27,27,25,27,27,27,27,83,78,83,83,69,83,85,83,85,77,83,83,85,83,83
,83,77,65,65,66,29,29,29,29,23,29,29,29,31,29,29,29,31,29,29,29,29,29,29,29,39,39,39,39,39,
39,39,39,39,39,39,39,39,39,39,39,36,33,39,39,18,18,18,18,18,18,14,18,18,18,18,18,18,18,18,1
8,18,18,15,14,15,7,15,15,15,14,15,14,15,15,15,14,14,15,14,15,14,15,14,11,11,12,12,12,11,11,
12,11,11,11,12,11,12,11,12,11,11,10,11,11,11,11,12,11,11,12,12,11,12,12,11,12,11,12,11,11,1
2,4,4,11,24,23,24,24,24,24,24,23,24,24,24,24,24,24,24,24,24,23,16,23,18,18,18,18,18,18,18,1
8,18,18,18,18,11,18,18,18,18,16,18,18,22,20,20,22,20,22,22,20,22,22,20,20,20,20,22,16,20,22
,14,20];

for i=0:20:length(rssi_1m)-1
   m=m+1;

   RSSI_mode1 = mode(rssi_1m(i+1:20+i));
   std_RSSI_mode1(m)=std(rssi_1m(i+1:20+i));

   if RSSI_mode1==0
      RSSI_1m_mode_dBm=0;
   else
      B = bitget(RSSI_mode1, 8:-1:1);
      l=length(B);
      indx=find(B);
      B=B(indx(1):l);
      l=length(B);
      y=-B(1)*2^(l-1);
```

```
        for e=2:l
            y=y+B(e)*2^(l-e);
        end
        RSSI_1m_mode_dBm=y;

    end

    d_1m_mode(m)=10^((-4.69-RSSI_1m_mode_dBm)/(10*2.73));

    std_lognormal_mode1(m)= sqrt((d_1m_mode(m)-1)^2)/20;

    Poly_RSSI_mode1(m) = p1*RSSI_mode1^2 + p2*RSSI_mode1+ p3;
    std_poly_mode1(m)= sqrt((Poly_RSSI_mode1(m)-stdDis_poly(1))^2)/20;

    Power_RSSI_mode1(m)= a*RSSI_mode1^b;
    std_power_mode1(m)= sqrt((Power_RSSI_mode1(m)-stdDis_power(1))^2)/20;
    %------------------------------------------------------------------
    RSSI_mean1=round(mean(rssi_1m(i+1:20+i)));
    std_RSSI_mean1(m)=std(rssi_1m(i+1:20+i));
        if RSSI_mean1==0
            RSSI_1m_mean_dBm=0;
        else
            B = bitget(RSSI_mean1, 8:-1:1);
            l=length(B);
            indx=find(B);
            B=B(indx(1):l);
            l=length(B);
            y=-B(1)*2^(l-1);
            for e=2:l
                y=y+B(e)*2^(l-e);
            end
            RSSI_1m_mean_dBm=y;

        end
        d_1m_mean(m)=10^((-5.18-RSSI_1m_mean_dBm)/(10*2.961));

    std_lognormal_mean1(m)= sqrt((d_1m_mean(m)-1)^2)/20;


    Poly_RSSI_mean1(m) = p1*RSSI_mean1^2 + p2*RSSI_mean1+ p3;
    std_poly_mean1(m)= sqrt((Poly_RSSI_mean1(m)-stdDis_poly(1))^2)/20;

    Power_RSSI_mean1(m)= a*RSSI_mean1^b;
    std_power_mean1(m)= sqrt((Power_RSSI_mean1(m)-stdDis_power(1))^2)/20;
end

rssi_2m=[45,45,44,39,44,45,45,44,45,44,44,41,44,45,44,44,45,45,44,40,64,60,62,62,62,57,62,
62,56,62,62,64,62,62,62,62,60,57,64,62,60,52,60,58,60,60,60,60,58,54,58,58,60,60,60,60,60,6
```

```
0,60,60,69,69,67,67,64,69,69,57,69,65,69,69,67,69,69,69,69,69,69,69,104,99,104,104,96,104,
98,104,104,98,104,104,104,98,106,104,106,104,98,104,62,62,62,62,64,62,62,62,62,62,57,64,6
2,64,64,62,62,57,62,62,66,65,66,61,66,60,66,58,60,58,65,66,65,60,58,65,65,66,65,66,53,53,46
,54,46,53,53,53,54,53,54,53,46,54,53,53,53,48,54,48,41,41,41,41,41,41,41,41,41,41,43,35,41,
43,41,41,41,41,35,41,39,39,39,39,39,40,39,40,39,40,39,39,39,39,37,39,39,37,39,36,40,32,39,3
9,37,40,39,39,39,37,39,39,39,39,39,37,39,39,31,33,44,44,44,44,45,37,45,45,45,44,44,44,37,44
,45,45,45,44,43,41,33,33,32,32,32,32,32,32,33,33,25,33,32,33,32,33,32,28,32,33,41,43,43,37,
43,41,41,41,41,35,41,40,41,41,41,41,45,45,44,44,48,41,48,48,46,46,46,48,48,46,48,46,48,50,5
6,57,57,56,57,49,44,44,45,45,45,45,45,44,44,44,44,44,45,44,44,44,44,43,45,45,31,31,29,29,29
,29,29,29,31,25,31,29,29,29,29,29,29,24,22,31,54,50,54,53,53,53,53,54,53,53,53,53,54,53,53,
53,54,48,46,53,50,52,50,43,50,50,50,50,50,50,50,50,50,50,50,50,52,49,48,36,35,35,36,36,3
1,36,35,36,35,35,36,35,36,36,35,36,36,36,36,29,29,29,29,23,31,29,29,29,25,29,29,29,31,31,29
,29,31,29,29,41,43,43,41,41,36,43,43,43,41,41,41,43,43,36,41,37,39,39,39,44,44,44,45,37,44,
44,45,39,45,39,44,45,45,40,44,39,43,44,45,53,54,53,54,53,53,53,48,54,46,53,53,53,54,54,53,5
3,53,48,53,46,41,46,48,48,46,48,48,46,48,48,48,48,48,48,48,46,46,44,43,33,32,32,32,33,32,32
,32,33,27,32,32,32,32,33,33,33,32,32,32,39,39,39,40,29,39,39,39,39,39,37,37,39,39,39,39,39,
32,39,39,36,36,36,36,35,35,36,36,35,28,36,35,35,35,35,36,35,35,32,35,39,32,39,39,39,40,39,3
9,39,39,39,39,39,39,39,39,39,39,37,39,29,29,29,29,29,29,31,29,29,24,29,29,29,29,31,29,25,29
,29,29,27,20,27,25,27,27,27,27,27,25,27,25,27,27,27,27,27,27,19,27,23,24,20,18,23,23,24,24,
24,18,24,24,24,23,24,24,23,23,22,24,27,27,27,27,27,27,27,27,27,27,27,27,27,27,25,27,23,22,2
7,27,24,24,24,24,22,24,23,24,23,24,24,23,24,23,23,23,24,19,23,23,23,23,23,24,24,24,23,23,23
,24,24,24,23,24,23,23,23,24,24,20];
m=0;
for i=0:20:length(rssi_2m)-1
   m=m+1;
   RSSI_mode2 = mode(rssi_2m(i+1:20+i));
   std_RSSI_mode2(m)=std(rssi_2m(i+1:20+i));
   if RSSI_mode2==0
      RSSI_2m_mode_dBm=0;
   else
      B = bitget(RSSI_mode2, 8:-1:1);
      l=length(B);
      indx=find(B);
      B=B(indx(1):l);
      l=length(B);
      y=-B(1)*2^(l-1);
      for e=2:l
         y=y+B(e)*2^(l-e);
      end
      RSSI_2m_mode_dBm=y;
   end

   d_2m_mode(m)=10^((-4.69-RSSI_2m_mode_dBm)/(10*2.73));

   std_lognormal_mode2(m)= sqrt((d_2m_mode(m)-2)^2)/20;

   Poly_RSSI_mode2(m) = p1*RSSI_mode2^2 + p2*RSSI_mode2+ p3;
   std_poly_mode2(m)= sqrt((Poly_RSSI_mode2(m)-stdDis_poly(2))^2)/20;
```

```
    Power_RSSI_mode2(m)= a*RSSI_mode2^b;
    std_power_mode2(m)= sqrt((Power_RSSI_mode2(m)-stdDis_power(2))^2)/20;
    %-------------------------------------------------------------------
    RSSI_mean2=round(mean(rssi_2m(i+1:20+i)));
    std_RSSI_mean2(m)=std(rssi_2m(i+1:20+i));
      if RSSI_mean2==0
          RSSI_2m_mean_dBm=0;
      else
          B = bitget(RSSI_mean2, 8:-1:1);
          l=length(B);
          indx=find(B);
          B=B(indx(1):l);
          l=length(B);
          y=-B(1)*2^(l-1);
          for e=2:l
              y=y+B(e)*2^(l-e);
          end
          RSSI_2m_mean_dBm=y;
      end
      d_2m_mean(m)=10^((-5.18-RSSI_2m_mean_dBm)/(10*2.961));
    std_lognormal_mean2(m)= sqrt((d_2m_mean(m)-2)^2)/20;

    Poly_RSSI_mean2(m) = p1*RSSI_mean2^2 + p2*RSSI_mean2+ p3;
    std_poly_mean2(m)= sqrt((Poly_RSSI_mean2(m)-stdDis_poly(2))^2)/20;

    Power_RSSI_mean2(m)= a*RSSI_mean2^b;
    std_power_mean2(m)= sqrt((Power_RSSI_mean2(m)-stdDis_power(2))^2)/20;
end

rssi_3m=[60,60,60,65,65,65,66,53,65,65,65,65,50,65,65,65,66,58,65,66,78,78,77,71,70,78,77,
77,79,74,81,73,81,75,81,75,73,81,81,81,74,81,81,75,75,81,81,79,79,77,81,81,81,81,81,81,81,8
1,75,81,75,75,74,74,74,67,70,75,74,75,62,74,74,74,74,74,74,67,69,75,53,60,60,60,60,60,60,60
,60,60,60,60,60,60,60,60,58,58,58,60,92,94,92,92,85,91,92,92,92,86,81,91,91,92,92,91,92,92,
92,88,92,86,92,92,86,87,92,92,92,85,92,92,92,92,85,92,88,87,91,91,92,87,91,92,85,86,92,92,9
2,86,92,92,86,87,91,90,92,87,87,91,79,79,81,81,77,67,81,81,81,78,79,81,79,81,79,79,78,77,78
,77,92,90,92,86,86,92,92,92,87,85,92,86,92,88,92,85,92,92,86,92,92,86,92,92,85,86,92,92,92,
86,92,92,85,87,92,91,92,87,86,85,87,87,86,86,79,87,86,87,87,78,82,87,87,87,86,85,87,86,87,8
6,98,98,99,99,91,98,92,99,98,92,94,98,98,98,99,91,98,92,91,99,24,23,24,24,23,24,24,24,24,24
,24,24,24,24,23,23,24,24,23,19,19,23,24,24,23,24,24,23,24,24,23,24,22,23,24,24,27,22,25,27,
29,29,29,29,31,24,29,31,29,29,29,29,29,29,29,31,31,29,23,31,18,18,16,18,18,18,18,18,18,18,1
8,12,18,18,18,18,18,19,18,16,24,29,29,29,29,31,29,29,29,29,27,29,29,29,29,29,29,31,29,27,32
,25,32,33,33,32,33,33,32,32,33,33,33,27,32,32,31,31,29,29,14,20,20,20,22,22,20,20,20,22,22,
20,20,20,20,20,20,22,18,20,49,53,46,48,53,48,53,49,54,54,54,53,53,54,53,54,54,54,54,54,45,4
5,44,45,45,45,45,44,44,44,44,45,45,44,44,43,45,44,35,36,53,54,53,53,53,45,53,53,54,53,54,54
,54,53,54,54,54,53,53,50,37,39,45,44,45,44,44,45,44,44,44,40,45,39,44,45,45,44,44,45,44,44,
45,44,45,39,45,39,45,45,45,44,45,44,44,41,45,45,39,45,33,37,39,39,39,39,39,39,39,37,39,39,3
9,39,39,39,39,39,39,39,39,35,33,39,39,39,37,39,37,39,39,33,39,39,40,39,39,39,32,39,29,31,29
```

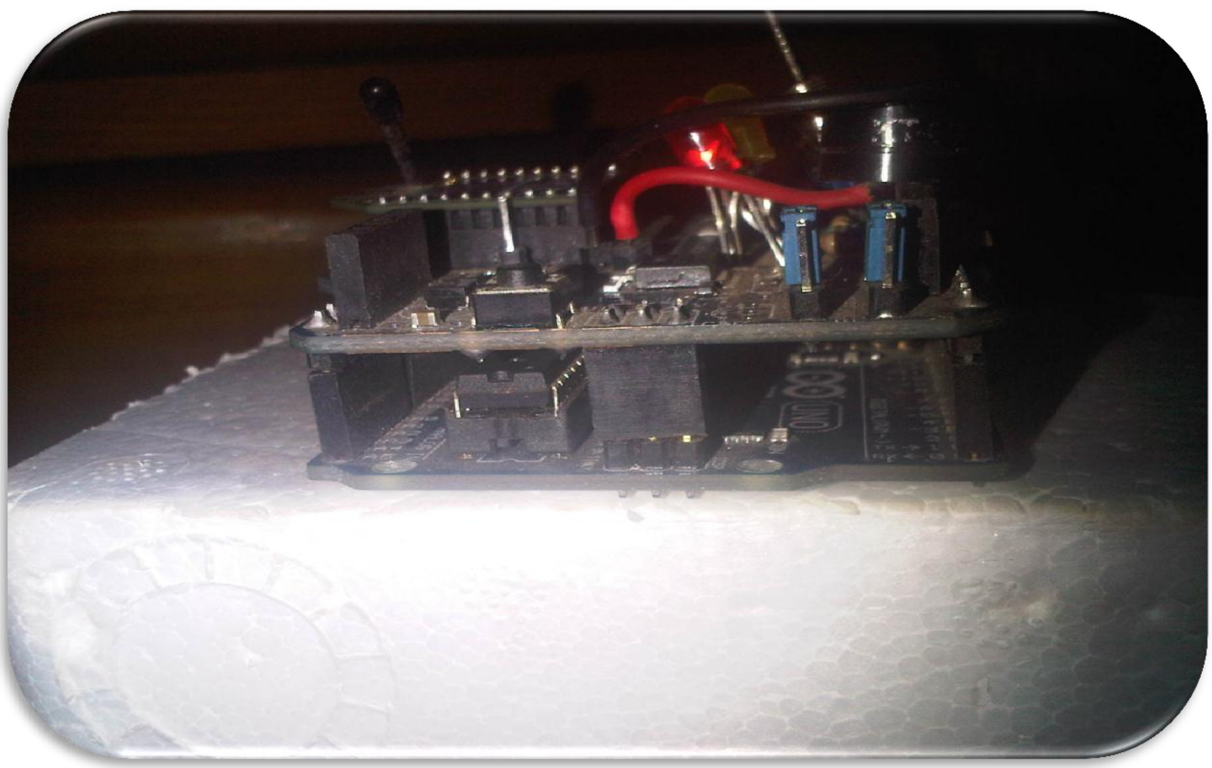
```
,29,31,29,29,29,23,29,29,29,29,29,29,29,29,27,29,29,39,33,39,39,39,37,37,39,39,39,39,39,39,
36,40,37,41,41,43,41,32,32,32,32,33,32,33,33,32,25,32,32,32,32,32,32,33,32,29,32,62,62,62,6
4,62,62,56,58,64,62,62,62,62,62,62,64,57,62,64,62,95,96,95,96,88,91,95,95,90,81,96,95,91,96
,91,95,87,95,95,94,69,69,69,69,62,64,69,69,69,69,69,69,69,67,69,69,69,69,62,69,69,67,67,67,
64,67,62,69,69,67,69,67,69,69,69,69,66,66,58,65,69,69,65,62,67,62,67,69,69,67,61,67,69,67,6
9,64,69,69,67,69,66,69,69,69,69,62,67,69,69,69,69,69,69,67,69,67,66,69,69,69,69,69,67,69,69
,69,69,67,69,62,69,69,62,69,69,69,67,67,69,67,74,74,74,74,66,74,75,74,67,71,74,74,74,74,74,
74,69,74,69,74,78,70,78,77,70,78,73,77,77,66,77,78,78,77,74,78,77,77,74,78,81,81,81,79,75,8
1,69,81,81,77,81,81,81,81,81,81,81,75,73,81,64,57,64,62,62,62,64,64,58,62,64,62,62,57,62,57
,62,62,62,64,43,41,43,41,35,41,43,43,43,41,43,41,41,43,41,41,41,36,43,41,41,41,43,41,41,41,
41,41,43,41,43,41,43,41,43,43,43,35,41,41,41,43,41,41,43,41,41,41,35,43,41,43,41,43,41,41,4
1,41,41,41,46,48,43,48,48,48,48,48,48,48,48,48,46,48,48,48,44,48,48,48,50,50,50,50,50,50,50
,50,45,50,45,50,50,50,50,50,50,44,50,52,48,48,46,46,41,48,48,46,48,48,48,48,48,48,48,46,48,
40,48,46,45,44,44,45,44,45,45,44,45,44,45,45,45,44,45,44,44,45,45,40];
m=0;
for i=0:20:length(rssi_3m)-1
    m=m+1;

    RSSI_mode3 = mode(rssi_3m(i+1:20+i));
    std_RSSI_mode3(m)=std(rssi_3m(i+1:20+i));

    if RSSI_mode3==0
        RSSI_3m_mode_dBm=0;
    else
        B = bitget(RSSI_mode3, 8:-1:1);
        l=length(B);
        indx=find(B);
        B=B(indx(1):l);
        l=length(B);
        y=-B(1)*2^(l-1);
        for e=2:l
            y=y+B(e)*2^(l-e);
        end
        RSSI_3m_mode_dBm=y;
    end

    d_3m_mode(m)=10^((-4.69-RSSI_3m_mode_dBm)/(10*2.73));

    std_lognormal_mode3(m)= sqrt((d_3m_mode(m)-3)^2)/20;

    Poly_RSSI_mode3(m) = p1*RSSI_mode3^2 + p2*RSSI_mode3+ p3;
    std_poly_mode3(m)= sqrt((Poly_RSSI_mode3(m)-stdDis_poly(3))^2)/20;

    Power_RSSI_mode3(m)= a*RSSI_mode3^b;
    std_power_mode3(m)= sqrt((Power_RSSI_mode3(m)-stdDis_power(3))^2)/20;
    %-------------------------------------------------------------------
    RSSI_mean3=round(mean(rssi_3m(i+1:20+i)));
    std_RSSI_mean3(m)=std(rssi_3m(i+1:20+i));
```

```
        if RSSI_mean3==0
            RSSI_3m_mean_dBm=0;
        else
            B = bitget(RSSI_mean3, 8:-1:1);
            l=length(B);
            indx=find(B);
            B=B(indx(1):l);
            l=length(B);
            y=-B(1)*2^(l-1);
            for e=2:l
                y=y+B(e)*2^(l-e);
            end
            RSSI_3m_mean_dBm=y;
        end
        d_3m_mean(m)=10^((-5.18-RSSI_3m_mean_dBm)/(10*2.961));

    std_lognormal_mean3(m)= sqrt((d_3m_mean(m)-3)^2)/20;


    Poly_RSSI_mean3(m) = p1*RSSI_mean3^2 + p2*RSSI_mean3+ p3;
    std_poly_mean3(m)= sqrt((Poly_RSSI_mean3(m)-stdDis_poly(3))^2)/20;

    Power_RSSI_mean3(m)= a*RSSI_mean3^b;
    std_power_mean3(m)= sqrt((Power_RSSI_mean3(m)-stdDis_power(3))^2)/20;
end
```

D. APPENDIX D

Images of finished system

First image of Transmitter unit



Transmitter unit after casing

Receiver unit

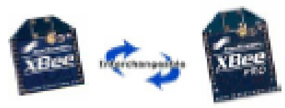
E. APPENDIX E

XBee module Datasheet

1. XBee/XBee-PRO OEM RF Modules

XBee and XBee-PRO Modules were engineered to meet ZigBee/IEEE 802.15.4 standards and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of critical data between devices.

The modules operate within the ISM 2.4 GHz frequency band and are pin-for-pin compatible with each other.



1.1. Key Features

High Performance, Low Cost

XBee

- Indoor/Urban: up to 100' (30 m)
- Outdoor line-of-sight: up to 300' (100 m)
- Transmit Power: 1 mW (0 dBm)
- Receiver Sensitivity: -92 dBm

XBee-PRO

- Indoor/Urban: up to 300' (100 m)
- Outdoor line-of-sight: up to 1 mile (1500 m)
- Transmit Power: 100 mW (20 dBm) EIRP
- Receiver Sensitivity: -100 dBm

RF Data Rate: 250,000 bps

Advanced Networking & Security

Retries and Acknowledgements

DSSS (Direct Sequence Spread Spectrum)

Each direct sequence channels has over 65,000 unique network addresses 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 (downloadable firmware version coming soon)

Low Power

XBee

- TX Current: 45 mA (@3.3 V)
- RX Current: 50 mA (@3.3 V)
- Power-down Current: < 10 µA

XBee-PRO

- TX Current: 270 mA (@3.3 V)
- RX Current: 55 mA (@3.3 V)
- Power-down Current: < 10 µA

Easy-to-Use

No configuration necessary for out-of box RF communications

Free X-CTU Software (Testing and configuration software)

AT Command Mode for simple configuration of module parameters

Small form factor

Network compatible with other ZigBee/802.15.4 devices

Free & Unlimited Technical Support

1.1.1. Worldwide Acceptance

FCC Approval (USA) Refer to Appendix A [p23] for FCC Requirements. Systems that include XBee/XBee-PRO Modules inherit MaxStream's Certifications.

ISM (Industrial, Scientific & Medical) 2.4 GHz frequency band

Manufactured under **ISO 9001:2000** registered standards

XBee/XBee-PRO RF Modules are optimized for use in **US, Canada, Australia, Israel and Europe** (contact MaxStream for complete list of approvals).

A full data sheet can be downloading at
www.libelium.com/.../3/31/**Data-sheet**-max-stream.**pdf**

F. APPENDIX F

Arduino Uno Data Sheet

Arduino UNO

CE

Product Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the Index of Arduino boards.

Index

RS radiospares RADIONICS

A full data sheet can be downloading at
http://www.inhaos.com/uploadfile/otherpic/DOC-BUONO-UNO-R3-V02-EN.pdf

G. APPENDIX G

ATtiny2313/v Microcontroller Data Sheet

Features

- Utilizes the AVR® RISC Architecture
- AVR – High-performance and Low-power RISC Architecture
  - 120 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
- Data and Non-volatile Program and Data Memories
  - 2K Bytes of In-System Self Programmable Flash
    Endurance 10,000 Write/Erase Cycles
  - 128 Bytes In-System Programmable EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 128 Bytes Internal SRAM
  - Programming Lock for Flash Program and EEPROM Data Security
- Peripheral Features
  - One 8-bit Timer/Counter with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Modes
  - Four PWM Channels
  - On-chip Analog Comparator
  - Programmable Watchdog Timer with On-chip Oscillator
  - USI – Universal Serial Interface
  - Full Duplex USART
- Special Microcontroller Features
  - debugWIRE On-chip Debugging
  - In-System Programmable via SPI Port
  - External and Internal Interrupt Sources
  - Low-power Idle, Power-down, and Standby Modes
  - Enhanced Power-on Reset Circuit
  - Programmable Brown-out Detection Circuit
  - Internal Calibrated Oscillator
- I/O and Packages
  - 18 Programmable I/O Lines
  - 20-pin PDIP, 20-pin SOIC, 20-pad QFN/MLF
- Operating Voltages
  - 1.8 – 5.5V (ATtiny2313V)
  - 2.7 – 5.5V (ATtiny2313)
- Speed Grades
  - ATtiny2313V: 0 – 4 MHz @ 1.8 - 5.5V, 0 – 10 MHz @ 2.7 – 5.5V
  - ATtiny2313: 0 – 10 MHz @ 2.7 - 5.5V, 0 – 20 MHz @ 4.5 – 5.5V
- Typical Power Consumption
  - Active Mode
    1 MHz, 1.8V: 230 µA
    32 kHz, 1.8V: 20 µA (including oscillator)
  - Power-down Mode
    < 0.1 µA at 1.8V

8-bit AVR® Microcontroller with 2K Bytes In-System Programmable Flash

ATtiny2313/V

Preliminary

Rev. 2543L–AVR–08/10

A full data sheet can be downloading at
www.atmel.com/images/doc2503.pdf.

H. APPENDIX H

9v buttery Datasheet

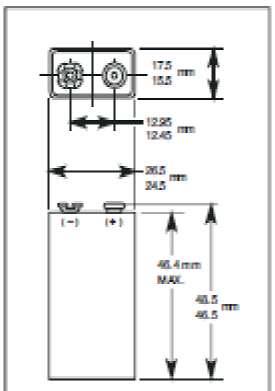
MN1604

Size: 9V (6LR61)

Alkaline-Manganese Dioxide Battery

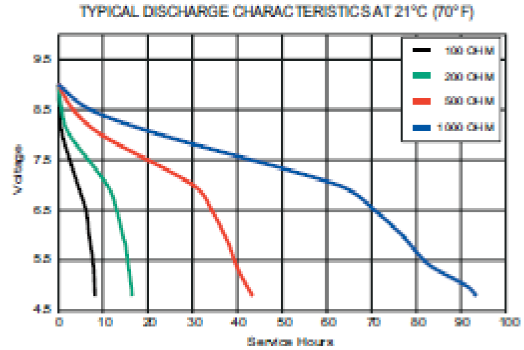
Zn/MnO_2

Nominal Voltage:	9 V
Operating Voltage:	9.6 - 4.8 V
Impedance:	1,700 m-ohm @ 1kHz
Typical Weight:	45 gm (1.6 oz.)
Typical Volume:	22.8 cm^3 (1.4 $in.^3$)
Terminals:	Miniature Snap
Storage Temperature Range:	-20 °C to 35 °C
Operating Temperature Range:	-20 °C to 54 °C (-4 °F to 130 °F)
ANSI: IEC:	1604A 6LR61



Dimensions shown are IEC/ANSI standards

TYPICAL DISCHARGE CHARACTERISTICS AT 21°C (70°F)



DURACELL BATTERIES
Berkshire Corporate Park
Bethel, CT 06801 U.S.A.
Telephone: Toll-free 1-800-544-5454
Internet: www.duracell.com

* Delivered capacity is dependent on the applied load, operating temperature and cut-off voltage. Please refer to the charts and discharge data shown for examples of the energy / service life that the battery will provide for various load conditions.

This data is subject to change. Performance information is typical. Contact Duracell for the latest information

6/08

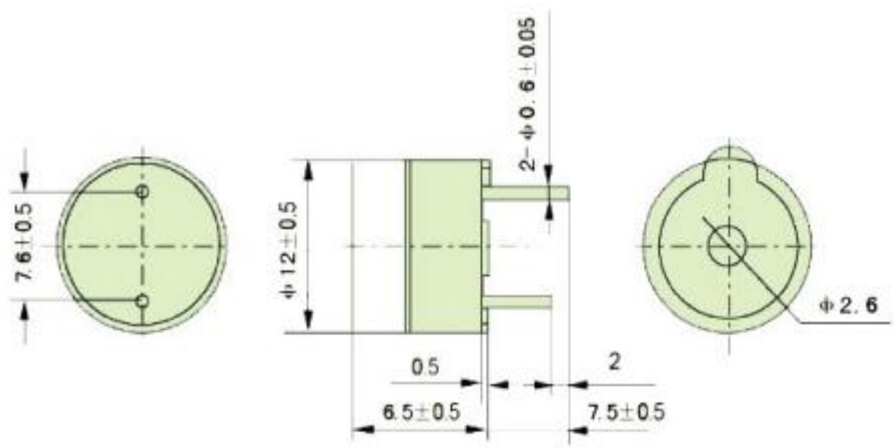
A full data sheet can be downloading at
www.DURACELL.com/datasheet.pdf.

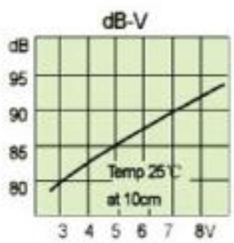
I. APPENDIX I

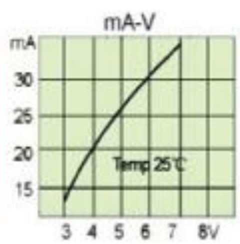
Piezo Buzzer Datasheet

Piezo Buzzer

Dimension







Rated Voltage	Operation Voltage	Rated Current	Sound Output	Resonant Freq	Operating Temp	Storage Temp	Weight
5V	4~7 V	<28mA	>85dB	2300±300	-20~+45	-20~+60	1.5

A full data sheet can be downloading at
Piezo Buzzer datasheet.pdf