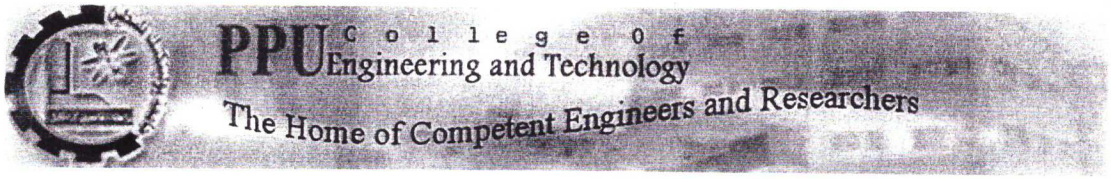


Palestine Polytechnic University

College of Engineering and Technology

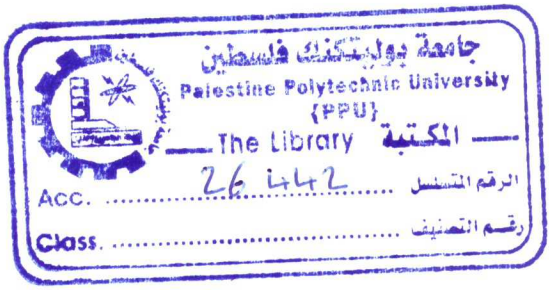
Electrical and Computer Engineering Department

Graduation Report

Building Management System

Project Team
Hamza Abu Ajamia
Fouad Abu Eisheh
Sarah Haddad
Amnah Al Masri

Project Supervisor
Eng. Yousef Salah
Project Co-Supervisor
Dr.Osama Ata



Hebron – Palestine
May, 2012

جامعه بوليتكنيك فلسطين
الخليل – فلسطين
كليه الهندسه والتكنولوجيا
دائره الهندسه الكهربائيه والحاسوب

اسم المشروع
Building Management System

اسماء الطلبه
حمزه خالد ابو عجميه
فؤاد نايف ابو عيشه
امنه ابراهيم مصري
ساره رشيد حداد

بناء على نظام كليه الهندسة والتكنولوجيا واشراف ومتابعة المشرف المباشر على المشروع وموافقة اعضاء اللجنه الممتحنة تم تقديم هذا المشروع الى دائرة الهندسة الكهربائيه والحاسوب وذلك للوفاء بمتطلبات درجة البكالوريوس في الهندسة تخصص هندسة انظمة الحاسوب وتخصص هندسة الاتصالات والكترونيات

توقيع المشرف

_____

توقيع اللجنه الممتحنه

_____ _____

_____

توقيع رئيس الدائره

_____

Project Awards

هذا المشروع حاصل على منحة رابطة خريجي فورد لدعم البحث العلمي

This project holds the grant of "Ford Alumni Association" to support scientific research

هذا المشروع حاصل على دعم من مركز الخدمات الميكانيكيه في وحده التكامل مع الصناعة في جامعة بوليتيكنيك فلسطين

This project is funded by the Mechanical Service Project in the Industrial Synergy Center at the Palestine Polytechnic University

Dedication

To our families and our parents whose constant inspiration and encouragements drives us to become all that we can be.

Our appreciation goes to our teachers and all staff of the faculty of engineering.
We give thanks for them for being our mentors, and our friends; it is indeed a privilege for us to be their students.

Hamza Abu Ajamia
Fouad Abu Eishah
Sarah Haddad
Amnah Al Masri

Acknowledgement

We would like to thank our mentors Eng.Yousef Salah and Dr.Osama Ata who read our numerous revisions of this document, and who were generous enough to give us the support we needed to make this project.

We must not forget to thank all the instructors in the Electrical and Computer Engineering Department for their great impact in our education.

Special thanks to Eng.Sami Al Salameen for all his care and valuable advices, for he was there for us with every time we need help.

Our thanks go to Dr.Raed Amro, Dr. Ramzi Qawasmi, Dr. Murad Abu Subaih for their support and advice with our project and graduation report.

We thank the Industrial Synergy Center for its financial and logistical support, and belief in our abilities.
We thank Ford Alumni Association for their generous grant, which greatly helps in the development of this project.

We acknowledge the Palestine Polytechnic University for giving us the opportunity to experience real engineering and exhibit some of the things we've been taught over the course of the last 5 years.

Finally we can't forget to acknowledge our great parents who gave so much to offer us the education we wanted. We truly cannot ever repay them for the love and assurance they entrusted in us.

Hamza Abu Ajamia
Fouad Abu Eishah
Sarah Haddad
Amnah Al Masri

Table of Contents

List of Tables

List of Figures

Abbreviation and Acronyms

3D	Three-dimensional space
AI	Artificial Intelligence
AI	Artificial Intelligence
API	Application Programming Interface
CO2	Carbon dioxide
DPDT	Double -Pole Double-Throw switch
etc.	Etcetera
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HID	Human Interface Device
HW	Hardware
IC	Integrated Circuit
ID	Identification
IDE	Integrated development environment
IDE	Integrated Drive Electronics
IP	Internet protocol
LAN	Local Area Network
LCD	Liquid Crystal Display
MC	Microcontroller
MS	Microsoft
OS	Operating System
PIC	Peripheral Interface Controller
PPI	Parallel Peripheral Interface
PPU	Palestine Polytechnic University
SDK	Software Development Kit
SMS	Short Message Service
SPDT	Single-Pole Double-Throw switch
SQL	Structured Query Language
SW	Software
TTL	Transistor-Transistor Logic
UI	User Interface
UML	Unified Modeling Language
USB	Universal Serial Bus
VGA	Video Graphics Array
WPF	Windows Presentation Foundation
WSN	Wireless Sensor Network
XAML	Extensible Application Markup Language
DCS	Digital Cellular System
GPRS	General Packet Radio Service
WAP	Wireless Application Protocol
MHz	Mega Hertz

ISM	(Industrial, Scientific and Medical
DTR	Data Terminal Ready
RTS	Request-to-Send
PAN	Personal Area Network
TX	Transmission
RX	Reception
GND	Ground
ID	Identity
API	Application Programming Interface
AT	Attention
ACK	Acknowledgement
DH	Destination high
DL	Destination low
RSSI	Received signal strength indicator
NLOS	non-line-of-sight
LOS	line-of-sight
MY	My network
ND	Node Discover
dBm	Power level in decibels relative to 1mW
+++	Entering command mode

Chapter One

Introduction

1.1. Problem Overview

1.2. Project Goals

1.3. Literature Review

1.4. Project Importance

1.5. Work Methodology

1.6. Budget Study

1.7. Time Plan

1.8. Needed Technologies

1.9. Risk Identification and Planning

1.10. Report Overview

1.1 Problem Overview

With the growing number of electrical appliances for household use, a major problem arises, which is the lack of management plus the economic and high environmental cost of running. Those appliances constantly consume electricity as long as they're plugged in, an occurrence known as energy vampires; and more specifically identified as the electrical consumption of electrical appliance when they're not in use.

It's proposed to build a cost-effective system, that has the capability to optimize and properly manage those appliances, and effectively cut the cost of running using monitoring, control protocols, and software algorithms; a system that properly and effectively transfers the living experience towards the 21st century.

With the proper implementation, we will hopefully reduce the amount of money users pay for electrical bills, and insure that they're fully aware of their environmental blueprint, and how they can properly save resources to be responsible human beings.

1.2 Project Goals

Building flexible and reliable automation system that can be optimized for home layouts and multiple environments needs to achieve these objectives:

1. Make the software as flexible as possible
2. Design and implement a controller capable of running a smart grid.
3. Design and implement a small scale sensor network.
4. Design and implement software algorithms capable of making efficient decisions in absence of the user.
5. Design and implement an attractive graphical interface for the software components to focus on the entertainment aspects of the system.
6. Implement a voice recognition mechanism for easier control.
7. Interface and implement a remote control mechanism with a wireless remote control device.

8. Extend the system's control and sensingwirelessly.
9. Extend the system's control by using the SMS messaging service
10. Design and implement a database for storing the data state of the system.
11. Design a testing mechanism to insure software and hardware components viability.

1.3 Literature Review

The following related projects and papers were researched:

- **Measurement of standby power for selected electrical appliances in Australia**[1]

This paper has highlighted the importance of reducing the standby power used in a range of common electrical appliances. In particular, consumer behavior and product choice can have a considerable impact on annual energy use and the associated greenhouse gas emissions. The minimum values of measured standby power also suggest that it is technically feasible to achieve standby power of less than 1W. It is believed that international coordinated action, such as the 'One Watt' initiative, can have a strong impact on the reduction of energy use by appliances globally.

The methodology used for analysis and calculating standby power is the following:

- Types and sample size of products in the study, a group of products were selected for the study, covering a wide range of electrical appliances.
- Instrument used for field measurements, the instrument used for the field measurements in this study was the Energy Monitor 3000, which is able to measure, with over 99% accuracy.
- Data analysis.

The results showed the corresponding annual standby energy estimated for the measured electrical appliances (kWh/appliance) from comparing the difference between minimum value and maximum value of standby power, it can be seen that considerable amounts of energy could be saved by using more energy efficient

appliances. For example, up to 170kWh of electricity could be saved annually in home theatre systems if an appliance with maximum value was replaced with one having minimum value. This would be equivalent to saving $32 per year and 177.7 kg of CO2 emissions. This shows the importance of consumer behavior and product choice, which could have a serious impact on energy demand and greenhouse gas emissions.

In our project we will solve even the standby power consumption from the poor energy efficient appliances by using the methodology of completely cutting of the power from the appliances so that no standby power consumption would occurs.

- **Monitoring and Controlling home by cellular technology**[2]

This project is based on GSM network technology for transmission of message (MMS, SMS) between system and owners.

The system maintain the security alert which is achieve in a way that on the detection of any abnormal events such that (gas leaking, fire and outer threats), the system allows automatic generation messages thus alerting the user against security risks.

The system was implemented using Java language and PIC18f4550, Smoke sensor, Motion sensor, Temperature sensor, Camera, and a Mobile.

Our project is more developed than this project, this project represents one of our project's units, which is the emergency unit, except that they use the MMS that we don't use in our project but the rest tasks is included in our project, and also they don't have a monitoring on electricity and water consumption, voice control or smart grid.

- **Smart House**[3]

In this project they design and implement warning and protection system against disasters that happen in urban house such as fire and theft, the system react to fire and theft crimes lighting a red color in the place to indicate the dangerous situation, lunching sound alarm and cut off the electricity of the house, and power on

a water pump working to extinguish the fire, also the system will automatically dial a previously saved phone number and send voice message indicate the theft crime.

The sound alarm is protected by a password so that no one is allowed to stop it unless the password entered and also the user can monitor the sensor status via the web interface.

This project uses an Atmel mega32 microcontroller as its core, keypad, visonic speech dialer, smoke sensor and motion sensor.

In our project we don't have internet access to the house status nor voice messages, but we have alarm system against fires and theft crimes as this project; also we employ a magnetic sensor in addition to the motion sensor in case of theft crimes.

Our project in more developed than this project, we have a smart grid, water and electricity consumption smart meters, in addition to the voice control.

- **Automated Home System**[4]

In this project they design stand-alone system that controls some house applications automatically such as opening or closing a door, a window, turning on/off a heat system.

The project uses PIC Microcontroller (18f4520), lights, temperature and motion sensor using a wireless technology. It was implemented using VB.net.

Our project is to be more developed compared than this project, it has a lot of other controlling options such (water and electricity consumption, emergency unit, SMS and voice control), and smart grid, also we use C #programming language.

- **Standby power requirements of household appliances in Canada**[5]

In this paper a study of Standby electricity use, or leaking electricity, which is the electrical energy consumed by household appliances when they are turned off, or not in use; In this study, standby power requirement data of household appliances were measured, analyzed, and presented for 268 new stock appliances and 1393 existing stock appliances in 75 households. The methodology used is that all power

requirement measurements were taken with a true RMS power Extech 380803 power analyzer, it was found that 222 of the 268 new stock appliances and 999 of the 1393 existing stock appliances have a standby power requirement. It was found that the power requirements of most appliances have a wide range indicating that manufacturers are capable of producing energy-efficient and cost-competitive appliances.

Based on the findings of this study, annual average standby energy consumption per household was determined to be 427 kWh. This could possibly be reduced by 59% to 177 kWh if the standby power requirement of all appliances were reduced to 1 W. while saving over US$ 2 billion in annual US energy costs and significantly reducing carbon emissions.

In our project we will solve the standby power consumption without reducing the standby power requirement of all appliances were reduced to 1 W, instead we will use a current transducer and a set of software rules that will completely cut off the power from the appliances so that no standby power will be consumed even if the appliances are turned off, or not in use.

1.4 What is the Importance of the Project?

Our project importance originates from the ability to manage a building, and so reduce the effort and time required to do daily tasks with regards to dealing with it. By doing that, it is clear that the proper implementation will have the following benefits:

- Increase awareness about normal living experiences consumption-wise.
- Provide the ability to manage the building using users' voice.
- Helps elderly and disabled people to maintain their independence and safety.
- Helps Getting rid of standby power consumption also known as "Vampire Energy", which expectedly reduces the running costs, expectedly a 10% to 40% decrease.

- Providing smart meters that calculate the amount of power and water consumed and so the equivalents payments to be paid.
- Reduce the CO2 emissions produced by the standby power which decrease users' environmental blueprint.
- Maintain a safe environment with regards to fires, gas leaks, and unauthorized access to the building area.
- Improve building security by introducing locking mechanisms activated via code or voice.
- Reducing time and efforts to do daily tasks that require control over the electrical grid.
- Provide a high-tech, easy to use, low-cost product, and sophisticated software to utilize the hardware at use.
- Provide the ability to Interface different types of wireless control nodes across the building for multiple uses, such as:

  - Gate control.

  - Shutter Control.

  - Provide a suitable interface with multiple speaker levels and multiple screen monitors.

1.5 Work Methodology

This section presents the work methodology in which this project will be processed with, it also defines the general process and characteristics of the work plan.

1.5.1 Development Process

Using the "Component-Based" Development Process model, the software will be defined as a set of modules or sub-components that address specific and small well-defined issues.

After all of the sub-components required for the project are fully developed, the integration process begins to form the final product.

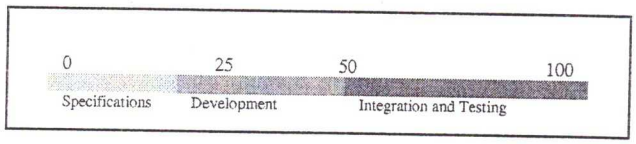


Figure 1.1 Component Based Developement Model

The main components required for software development are:

- **Main Hardware Driver module,** which is the Universal Serial Bus (USB) software component used to interface with the main microcontroller.
- **Secondary Hardware Driver module,** which is the serial interface software components used to interface with the wireless transceiver.
- **Voice Recognition Engine and Speech Synthesizer Interface**. A component used to convert voice commands over the microphone into an integer code.
- **Database and Database interface** for saving users information and personal configurations.
- **Windows Presentation Foundation Graphical User Interface module**, which acts as the main interface for the program, it displays a 3D character, consumption charts, an interactive building plan, and environmental information regarding the state of the building.
- **Main software controller and Event System.** Responsible for decision making and interfacing all of the software modules.

As for Hardware development, a "Spiral" Development Model will be used, which means that the development process will first address the basic requirements, then undergo evolution by redefining the problem and upgrading the hardware gradually to get the final product.

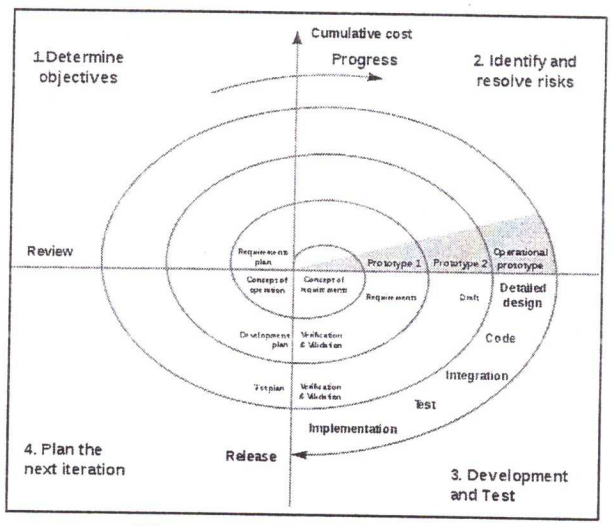


Figure 1.2 Spiral Model

An abstraction of the system's hardware components is the following:

- Main Controller, a device with sensory inputs and control ports.
- Secondary Controller, a device that interacts with the wireless extensions of the system.
- Wireless nodes, wireless components capable of interfacing 4 sensor/control ports.
- Control Nodes, which are computer controlled switches.
- Peripherals, such as the Keypad, Microphone array … etc.

1.5.2 Proposed Technologies

The project will use the following list of technologies, characterized into two categories, Hardware and Software:

Hardware technologies

- XBee Wireless Sensor Network (WSN).
- A GSM Cellular module.
- Programmable Microcontrollers.
- Universal Serial Bus Interfacing.

Software technologies

- .Net Framework, which includes:
  - Visual C#, as the main language for software development. Since it easily offers easy code integration and a .Net Speech Engine.
  - Windows Presentation Foundation (WPF), for interface design.
- MPLAB C18 language API for microchip programming.

- Google's Android Development Environment

1.6 Tasks and Time Plan

The project goes through 3 stages, a preliminary research stage, First Semester Stage, and the Second Semester Stage.

1.6.1 Stage 1 –Preliminary Research

In the preliminary research stage, the problem was researched and characterized, then an initial understanding of the problem was reached and an understanding of the system was achieved. This paved the way for the rapid development of the project's components, and allowed some isolation in functionalities for easier tasks deployment

Table 1.1 Stage 1 - Preliminary Research Schedule Chart

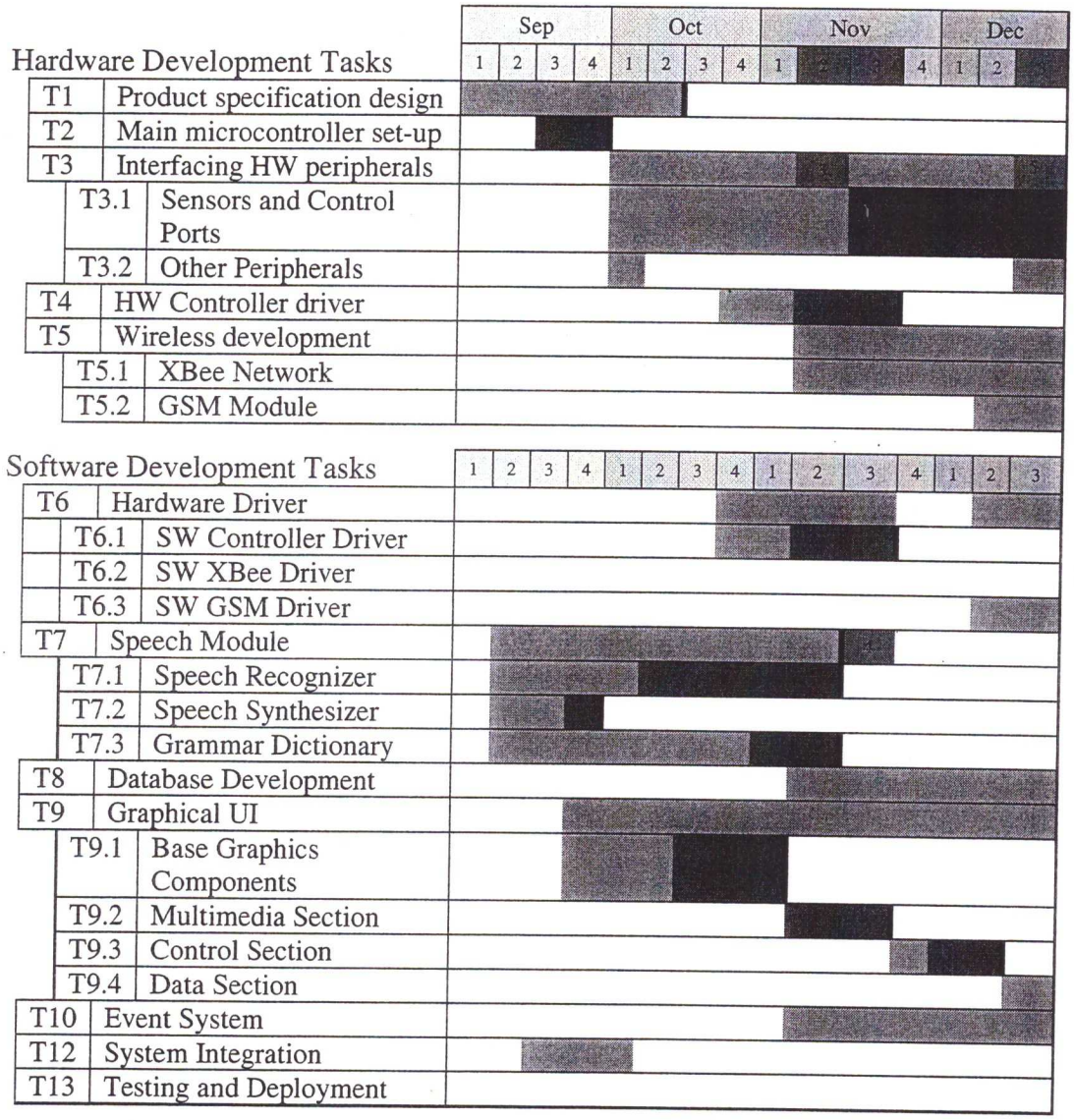
	Preliminary Research	May				Jun				Jul				Aug			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
R1	Resea ch Problem Parameters																
R	In ial non-technical des gn of the system																
R3	Research in Possible solutions and available technologies																
R4	Research financial viability and business plan																
R5	Development Plan																

Research Milestones	
M1	Basic understanding of the problem's parameters
M2	Finished business plan

1.6.2 Stage 2 – First Semester

In this stage, the system hardware's main structure is designed and implemented. Most of the software's technical modules are designed then implemented independently, so that later they can be assembled within the main logic. The following Tasks include both a design stage, and an implementation stage:

Table 1.2 Stage 2 - First Semester Schedule Chart



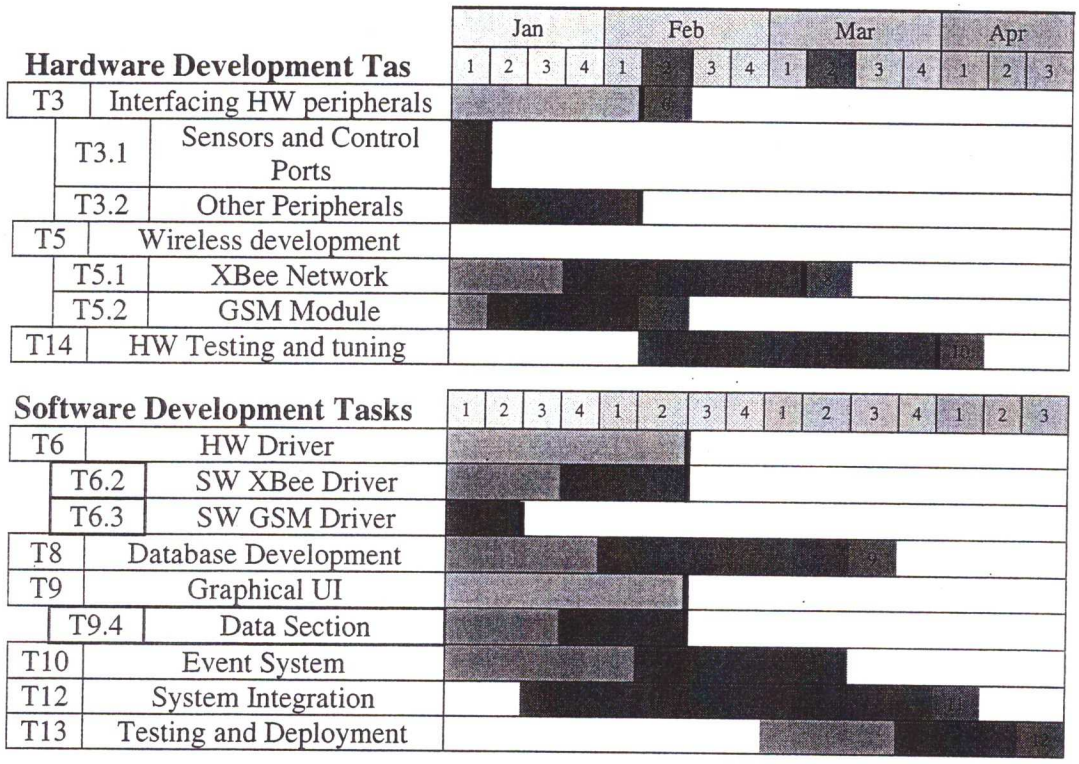
Design	
Implementation	
Milestone	
M3	Arrival of the first batch of Hardware components
M4	Speech Module Ready for integration
M5	Main Controller Operational and Prototype ready for integration

1.6.3 Stage 3 – Second Semester

In this stage, the system begins to shape in its final form, the software components are finalized and assembled, the wireless network is implemented, and system is tested and then deployed.

The following table follows the one in Stage 2:

Table 1.3 Stage 3 - Second Semester Schedule Chart



Design	
Implementation	
Milestone	
M6	Main Controller Ready
M7	GSM Module Ready
M8	XBee Network Ready
M9	Database Ready
M10	HW Tuned for Operation
M11	Software Integrated
M12	Project Finished

1.7 Project Budget Analysis

Listed and characterized as following:

1.7.1 Software Tools

This project uses the following set of software

Table 1.4 Software Tool Set

Software	Provider	Costs
MS Windows 7 OS	PPU	None
Microsoft Office 2010	PPU	None
MPLAB IDE	PPU	None
C18 Package	PPU	None
Cadence OrCAD 9.2	PPU	None
MS SQL Server 2008	MS Dreamspark	None
MS Visual Studio 2010	MS Dreamspark	None
Dia Software	Open Source	None

1.7.2 Hardware Apparatus

The project estimated costs were calculated and found to be around 2800$, this table provides a detailed list of the components needed for the project

Table 1.5 Hardware Budget

Components	**Price (NIS)**	**Quantity**	**Total (NIS)**	**Description**
PIC Programmer	None	1	None	Provided by the Projects Lab
PIC18F4550 MC	57.25	3	171.75	Microcontroller
8255A PPI	28.625	4	114.5	Peripheral Interface
74LS139	5.725	4	22.9	2x4 Decoder/Demultiplexer
XB24-Z7WIT-004 Module	206.1	3	618.3	ZigBee module Series 2 Antenna
XBee Explorer Module	148.85	1	148.85	XBee USB Serial adapter
Breakout board with regulator	34.35	3	103.05	Interface XBee module with TTL voltage
Other Electronics	343.5	1	343.5	Resistors, Capacitors, Connectors,

				Wires, Transistors …
Extension unit	3.435	64	219.84	OptoIisolator, Transistor, and 2 bit Connector
Power Supply	137.4	1	137.4	+5 +12 Power Supply
Breaker	103.05	3	309.15	40A Rating
Contactor	148.85	3	446.55	24V/220V 40A Switch
Relays	51.525	8	412.2	30A Rating
Presence Sensor	68.7	6	412.2	(IR / Ultrasonic)
Current Transducer	286.25	3	858.75	40A Rating
Temperature Sensor	171.75	2	343.5	
CO2 Sensor MQ-811	400.75	1	400.75	CO2 Sensor/Smoke Detector
Sensor MQ-7	103.05	2	206.1	CO Sensor
Sensor MQ-4	103.05	3	309.15	Methane Sensor
Smoke Detector	171.75	4	687	Smoke Sensor
Water Flow Sensor	435.1	1	435.1	Water Consumption Measurement
2.4GHz HID Key Input Device	503.8	1	503.8	for wireless Remote Control
Portable Stand	343.5	1	343.5	To hold and display components
IP Camera	687	1	687	
LCD Screen	64.12	1	64.12	64 Character LCD Display
NetBox-nT330 Minicomputer	1832	1	1832	20W Power Rating
Total	**5752.48 (NIS)**		**10,131 (NIS)**	**2,814.16 (US Dollars)**

On the assumption that 1 USD is 3.6 NIS.

1.8 Risk identification and Planning

Risks is a major factor to consider while going through any project, so it's important to anticipate the risks that might affect the project schedule progress and to take action to avoid these risks is very important for the success of the project.

The following table lists the types of risk which may impact the project development process, showing the risk and its type – Technology, people, organizational, tools, requirements, and estimation – and a short description about it.

Table 1.6 Risks Identification and Characterization

Type of Risk	Possible Risks	
Technology	R1	Hardware which is essential for the project may not be delivered on schedule.
	R2	Malfunction of hardware parts (IC's, Microcontroller, Sensors).
	R3	The wireless sensor network and the related hardware parts may not be constructed by its team.
	R4	Radio Frequency interface with the wireless sensor network may get a wrong reading on the base station receiver.
	R5	Problem with Interface the voice engine with the system and its sensitivity
	R6	Problems during constructing the graphical user interface (GUI).
	R7	The inability to build a valid Artificial Intelligence (AI) in the system.
Requirements	R8	There may be a larger number of changes to the requirements than anticipated.
Estimations	R9	The time required developing the software and the hardware is underestimated.
	R10	The size of the software and the hardware is underestimated.
	R11	The Budget is not sufficient.
Tools	R12	Code of the software is damaged or deleted suddenly.
	R13	Malfunction in the Team's IC's Programmer.
Organizational	R14	Time of delivery of the project changed.
People	R15	Illness of one or more team member.

The following table lists the probability and effect of each risk:

Table 1.7 Risk Probability and Analysis

Risk ID	Risk	Probability	Effects
R1	Hardware which is essential for the project may not be delivered on schedule.	Moderate	Catastrophic
R2	Malfunction of hardware parts (IC's, Microcontroller, Sensors).	Low	Serious
R3	The wireless sensor network and the related hardware parts may not be constructed by its team.	Very low	Serious
R4	Frequency interface with the wireless sensor network may get a wrong reading on the base station receiver.	Low	Serious
R5	Problem with the voice engine sensitivity and its actual usability	Low	Tolerable
R6	Problems during constructing the graphical user interface (GUI).	Very low	Tolerable
R7	Building the Artificial Intelligent (AI).	Moderate	Serious
R8	There may be a larger number of changes to the requirements than anticipated.	Low	Tolerable
R9	The time required developing the software and Hardware is underestimated.	Low	Tolerable
R10	The size of the software and Hardware is underestimated.	Low	Tolerable
R11	The Budget is not sufficient.	Moderate	Catastrophic
R12	Code of the software is damaged or deleted suddenly.	Very low	Catastrophic
R13	Malfunction in the Team's IC's Programmer.	Moderate	Tolerable
R14	Time of delivery of the project changed.	Very low	Tolerable
R15	Illness of one or more team member.	Low	Tolerable

The following table shows the strategies that address each risk in this project:

Table 1.8 Risk Management Strategies

Risk ID	Strategy
R1	Identify the needed hardware as fast as possible and order them.
R2	Continue the project development and simulate the network operation.
R3	Continue the project development and simulate the network operation.
R4	Prevent that by trying the methods of reducing radio frequency interference
R5	Produce protocols for smart listening
R6	Build a number variation of the GUI.
R7	Simplifying the needed AI algorithms and total required AI tasks.
R8	Derive traceability information to assess requirements change impact, and maximize information hiding in the design.

R9	Understand almost the exact time needed in developing the software and hardware.
R10	Understand almost the exact size for the software and hardware.
R11	Borrow the needed money.
R12	Always save backup copies of the software (on flash, CD's, hard disks)
R13	Depend on another team's Programmer, or buy a new one.
R14	Refer to University Regulation
R15	Reorganize team so that there is more overlap in the work

1.9 Report overview

The following chapters will be included in this report:

Chapter One - Introduction: an introduction to the project, describing the problem, project idea, and its aims and the software engineering methodology followed.

Chapter Two - Background: A theoretical background study on the components and technologies needed for this project.

Chapter Three - Design: The system's design is laid out, put in two categories hardware and software, then it is dissected into a group of subsystems, each with its functionality described and detailed. Finally giving a complete overview of how the system works starting from

Chapter Four - Implementation: This section of the report will be filled in the 2nd stage of the project.

Chapter Five - Testing: This section of the report will be filled in the 2nd stage of the project.

Chapter Six - Conclusion: This section of the report will be filled in the 2nd stage of the project.

Chapter Two

Theoretical Background

2.1 Microchip Microcontrollers

2.2 Windows Presentation Foundation

2.3 Extensible Application Markup Language

2.4 Microsoft Speech SDK

2.5 Universal Serial Bus

2.6 GSM Modem

2.7 XBee Radios

2.8 Arduino

2.9 8255A Programmable Peripheral Interface Sensors

2.10.1 Presence Sensor

2.10.2 Temperature Sensor

2.10.3 CO2 Sensor

2.10.4 Current Transducers

2.10.5 Water flow Sensors

2.10.6 Smoke Detectors

2.1 Microchip Microcontrollers [6]

Microcontrollers are a small and low cost computers existing on a single integrated circuit. More specifically, a microcontroller contains a Central Processing Unit (CPU), Random Access Memory (RAM), Read Only Memory (ROM), Input /Output Lines (I/O), serial and parallel ports, timers and other peripherals such as Analog To Digital (A/D) and (D/A) converters.

In this project the PIC18F4550 8-bit microcontroller is chosen as the main controller unit. It is one of the best performers in the PIC18F family, and is based on 16-bit instruction set architecture. Usually, it is used for (nanoWatt) low power applications, and interfacing applications because of the availability of three serial ports. It's also well regarded as an advanced microcontroller that is equipped with enhanced communication protocols such as the Universal Serial Bus 2.0 protocols.

PIC18F4550 has a 32KB flash memory, a non-volatile computer storage chip that can be electrically erased and reprogrammed. It has large amounts of RAM memory for buffering and Enhanced Flash program memory, which makes it ideal for embedded control and monitoring applications that require periodic connection with a (legacy free) personal computer via USB for data upload/download and/or firmware updates.

The chip's most widely available form factor is a 40 pin silicon chip consisting of 5 I/O ports (PORTA, PORTB, PORTC, PORTD and PORTE). PORTB and PORTD have 8 pins to receive/transmit 8-bit I/O data. The remaining ports have different numbers of pins for I/O data communication.

PIC18F4550 can work on different internal and external clock sources. It can work on a varied range of frequency from 31 KHz to 48 KHz. it has four in-built timers. And it contains various inbuilt peripherals like ADC, comparators … etc.

The main features of PIC18f4550:

- 2 KB RAM.
- 32 KB Flash memory.
- 256 Byte EEPROM.
- CPU Speed (12 MIPS).
- 2 Comparators.
- 4 Timers.
- 5 Digital/IO PORTS, 35 Pins in total.
- 14 Analog input channels
- USB interface

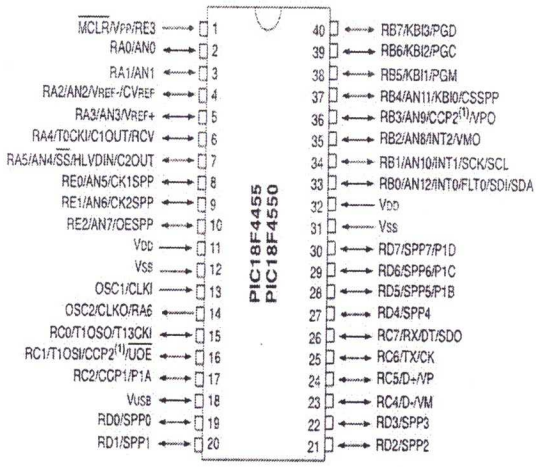


Figure 2.1 PIC18F4550 Pin Layout

2.2 Windows Presentation Foundation (WPF) [7]

Windows Presentation Foundation is a computer software graphical subsystem developed by Microsoft for rendering user interfaces in windows based application. It is a part of .Net framework 3.0 and higher and it combines application UIs, 2D graphics, 3D graphics, documents, typography, vector graphics, runtime animation and multimedia into one single framework. The core of WPF is a resolution-independent and vector-based rendering engine that is built to take advantage of modern graphics hardware

WPF separates the appearance of a user interface from its behavior. The appearance is generally specified in the Extensible Application Markup Language (XAML), the behavior is implemented in a managed programming language like C# or Visual Basic. The two parts are tied together by data binding, events and commands. The advantages of this separation are appearance and behavior are loosely coupled, Designers and developers can work on separate models, and graphical design tools can work on simple XML documents instead of parsing code.

Microsoft provides two development tools for WPF applications. One is Visual Studio, made for developers and the other is Expression Blend made for designers. While Visual Studio is good in code and XAML editing, it has a rare support for all the graphical stuff like gradients, template editing, animation, etc. This is the point where Expression Blend comes in. Blend covers the graphical part very well but it has (still) rare support for code and XAML editing, so both will be needed for the development of this project.

2.3 Extensible Application Markup Language (XAML) [8]

Extensible Application Markup Language (XAML), Pronounced 'ZAMMEL', is a declarative markup and simple language based on XML, created by Microsoft originally for WPF to create and initialize .NET objects with hierarchical relation, and to create user interfaces in WPF, Silverlight, declare workflows in WF and for electronic paper in the XPS standard.

The advantages of using XAML code in this project are as following: XAML code is short and clear to read, separation of designer code and logic and allows developers to separate the roles of designer and programmer. Microsoft Expression Blend will be used in the development of XAML required for our application.

2.4 Microsoft Speech SDK [9]

Microsoft Speech SDK is a software development kit for building speech engines and applications for Microsoft Windows, it is not an end user application, it is only intended to allow the programmers to write applications incorporating speech recognition and speech synthesis into them.

Speech Synthesis is a technology known as Text-To Speech (TTS) converts textual information into synthetic speech output, but speech recognition is a container of

language rules that define a set of known language constraints that a speech recognizer can use to perform recognition

The SDK contains conceptual information about adding speech to the applications , the Microsoft continuous speech recognition engine and Microsoft concatenated speech synthesis (or text-to-speech) engine, and a reference manuals for tools to test and optimize the voice applications , sample application and tutorials that demonstrate the use of Speech with other engine technologies, sample speech recognition and speech synthesis engines for testing with speech-enabled applications, and documentations for the managed code which is based on Microsoft .Net framework.

2.5 Universal Serial Bus [10]

The major goal of USB was to define an external expansion bus which makes adding peripherals to a PC low cost and as easy as hooking up a telephone to a wall-jack. USB connections allow data to flow both ways between the PC and peripheral. This means you can use your PC to control peripherals in new and creative ways. USB is the first cross-platform "hot-swappable" interface- no more operating system incompatibility, no more restarting before unplugging or plugging in, no more mess. USB has benefits for everyone. It helps to reach the first time PC consumer by overcoming usability barriers through easy peripheral installation.
USB has numerous benefits, listed as the following:

- Significant user benefit USB 2.0 is fast provides full bandwidth for everyone.
- Simplicity "user obvious" technology. Small connectors, easy installation via plug-and-play, and only one cable type all contribute to the new standard's simplicity and ease of use.
- Compatibility. USB 2.0 is fully backward- and forward-compatible with USB 1.1 .
- Transparency. USB 2.0 uses the same cables as USB 1.1. The connector shapes are the same. The topology is the same. So if the user doesn't know the difference between USB 1.1 and USB 2.0, the system will still work.

- Cost effectiveness. because a number of variables can influence the final cost of the product. Another I/O standard, 1394 (FireWire), achieves similar speeds to USB 2.0 and thus is a good model for comparison.

USB2 has several benefits it is fats ;it will be transfer 480 Mb/sec and this 40 times faster than the old one USB 1.1 , its cable is 5 meters length for single cable and it could be extending to 30 meters , it connect up to 127 devices to a single host using 5 hubs.

2.6 GSM Modem [11]

Global system for mobile communication (GSM) is a globally accepted standard for digital cellular communication. GSM is the name of a standardization group established in 1982 to create a common European mobile telephone standard that would formulate specifications for a pan-European mobile cellular radio system operating at 900 MHz it is estimated that many countries outside of Europe will join the GSM partnership.

The GSM system is a frequency- and time-division system; each physical channel is characterized by a carrier frequency and a time slot number. GSM system frequencies include two bands at 900 MHz and 1800 MHz commonly referred to as the GSM-900 and DCS-1800 systems. For the primary band in the GSM-900 system, 124 radio carriers have been defined and assigned in two sub-bands of 25 MHz each in the 890–915 MHz and 935–960 MHz ranges, with channel widths of 200 kHz. Each carrier is divided into frames of 8 time slots (for full rate), with frame duration of about 4.6 ms. For DCS 1800, there are two sub-bands of 75 MHz in the 1710–1785 MHz and 1805–1880 MHz ranges.

Today, GSM is used mainly for speech communication, but its use for mobile data communication is growing steadily. The GSM Short Message Service (SMS) is a great success story: several billion text messages are being exchanged between mobile users each month. The driving factor for new (and higher bandwidth) data services is the wire- less access to the Internet. The key technologies that have been introduced in GSM, the General Packet Radio Service (GPRS) and the Wireless Application Protocol (WAP).

2.7 XBee Radios [12]

XBee is a wireless communication module that is built to the 802.15.4/ZigBeestandard, XBee modules are bidirectional 2.4 GHz digital radios targeted towards hobbyists. They cost relatively cheap and are advertised to use small amounts of power. It allows multi-point networks. That means you can have more than two modules communicate with each other have a typical range of about 100-200 feet in a normal home or office. There are high-power XBee radios called XBee Pro that can achieve farther ranges (up to 2km in open space) and several 900Mhz models that have a range of several miles in open space. The modules have selectable baud rates (1200-230400 bps) and a 3.3 volt serial interface. [12]

The XBee modules use 128 bit AES encryption to encrypt data ,this mean give a good security for data is used for viewing sensor data, remote control, and uploading scripts. It's really handy having a wireless connection and no configuration needed for out-of-the-box RF communications. (12)

The XBee modules offer good reliability. So when you send a byte, chances are that it will arrive at the other end intact. Another plus is that the newer XBee modules also have built-in ADC's and digital I/O's so you can communicate simple information (like the output of a sensor) without bothering with a microcontroller, it can form self-healing mesh networks. [13]

2.8 Arduino [14]

Arduino is an open-source electronics prototyping platform it can sense environment by receiving input from a variety of sensor,it provides your robot the intelligence you want by using a variety of sensors, Arduino can affect your robot's behavior by controlling motors, actuators, LCDs or any other robot components.

Arduino can be stand-alone, or they can be communicate with software running on your computer ,it has digital and analog input/output pins ,serial communication pins, In-system programming pins (ISP), Compatibility with Arduino software.

Arduino also simplifies the process of working with microcontrollers, but it offers some advantage like the following:

- Inexpensive - the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
- Cross-platform - The Arduino software runs on Windows, Macintosh OSX, and Linux operating systems. Not like other microcontroller systems are limited to Windows.
- Simple, clear programming environment - is easy-to-use for beginners, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with the look and feel of Arduino
- Open source and extensible software- The Arduino software and is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries.

2.9 8255A Programmable Peripheral Interface[16]

8255 is a general-purpose I/O device that connecting peripherals equipment to a microprocessor system bus . It is programmable , flexible , versatile , economical and parallel I/O device . it could be programmed to transfer data under various conditions from simple I/O to interrupt I/O .

8255 is a 40 pins chip consisting of 24 I/O pins that can be grouped primarily in two 8-bit parallel ports: A and B, with the remaining eight bits as port C. The eight bits of port C can be used as individual bits or be grouped in two 4-bit ports: CUPPER (CU) and CLOWER (CL).

8255 operates in two different modes:

- Input /Output Mode : There is three types of I/O modes :

1. **Mode 0**: Port A, B and C can be independently configured as input or output to read or write data. Ports A and B are used as two simple 8-bit I/O ports and port C as two 4-bit ports , and the ports do not have handshake signals. In this Mode Outputs are latched but the inputs are not latched.
2. **Mode 1 :** Port A and B can be configured as input or output ports , and Port C can be used as handshake signals 3 lines for Port A and 3 Lines For Port B and the remaining two lines can be used for simple I/O functions .In this mode the input and output are latched.
3. **Mode 2**: Port A can be configured as the bidirectional port and Port B either in Mode 0 or Mode 1.Port A uses five signals from port C as handshakesignals for data transfer , the remaining three signalsfrom Port C can be used either as simple I/O or a handshake for Port B.

- Bit Set/Reset (BSR) Mode :

The BSR mode is concerned only with the eight bits of port C which canbe set or reset by writing an appropriatecontrol word in the control register.

The I/O operations of ports A and B are notaffected by a BSR control word

The Main features Of 8255 PPI are:

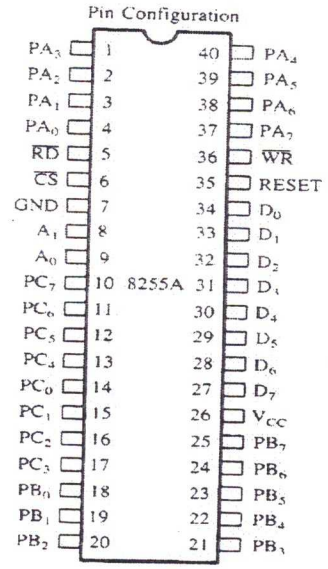


Figure 2.2 8255A Pin Layout

- 3 I/O Ports, 24 Programmable Pins in total.
- Can be used with any microprocessor.
- 8-bit bidirectional data bus.
- Three Operation Modes; Mode 0, 1 and 2.
- RESET input clears the Control Register and all 24 programmable I/O lines set toinput mode.
- High Speed (zero wait state).
- Low Power.
- Direct Bit Set/Reset Capability.

2.10 Sensors

A sensor (detector) is a device that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument, sensor is a device which receives and responds to a signal.

The main sensors used in this project are:

2.10.1 Presence Sensor [17]

Motion detectors respond to walking movements. They perceive these in the selected detection zone and respond to them, this means light is switched 'ON' when a movement is detected once ambient light levels fall below a preselected threshold. Light switches 'OFF again after the period you set. Their use is recommended for detecting moving objects outdoors or in thoroughfares indoors.

presence sensors use different technologies — including passive infrared (PIR), detect occupants' presence by sensing the difference between heat emitted by moving people and background heat ,ultrasonic sensors detect the presence of people by sending out ultrasonic sound waves into a space and measuring the speed at which they return. They look for frequency changes caused by a moving person.

PIR sensors require a direct line of sight between the sensor and occupants in a space, PIR sensors are very suitable for enclosed spaces, wall-switch replacements, and areas with direct line-of-sight viewing, and spaces in which it is necessary to mask unwanted detection in certain areas.

Ultrasonic sensors, meanwhile, are highly suitable for spaces in which a line of sight is not possible, such as partitioned spaces, and in spaces requiring a higher level of sensitivity.

2.10.2 Temperature Sensor [18]

A thermocouple is a pair of junction formed from two dissimilar metals, once at a reference temperature and the other at the temperature to be measured , use the fact as temperature increases, the voltage across a diode increases at a known rate. (Technically, this is actually the voltage drop between the base and emitter - the Vbe -

of a transistor. By precisely amplifying the voltage change, it is easy to generate an analog signal that is directly proportional to temperature.

2.10.3 MQ-gas Sensor [19]

The MQ series of gas sensors use a small heater inside with an electro-chemical sensor. They are sensitive for a range of gasses and are used indoors at room temperature. They have high sensitivity, low cost and suitable for different application.

The voltage for the heater is very important. The heater may not be connected directly to an output-pin, since it uses too much current for that.
Some sensors use 5V for the heater, others need 2V. The 2V can be created with a PWM signal, using analogWrite() and a transistor or logic-level mosfet.
Some sensors need a few steps for the heater. This can be programmed with an analogWrite() function and delays. A transistor or logic-level mosfet should also in this situation be used for the heater.

2.10.4 Current Transducers

A current sensor is a device that detects electrical current (AC or DC) in a wire, and generates a signal proportional to it. The generated signal could be analog voltage or current or even digital output. It can be then utilized to display the measured current in an ammeter or can be stored for further analysis in a data acquisition system or can be utilized for control purpose. [20]

In this Project, the sensors to be used are built on using current transformer, and an interfacing circuit, as it offers the following advantages: [21]

- Electrical isolation
- Capable of handling high current
- Low power consumption
- Low temperature shift
- Light weigh
- Low Cost

2.10.5 Smoke Detectors [23]

A smoke detector is a device that detects smoke, typically as an indicator of fire. Commercial, industrial, and mass residential devices issue a signal to a fire alarm system, while household detectors, known as smoke alarms, generally issue a local audible and/or visual alarm from the detector itself.

Chapter Three

System Design

3.1 System Abstraction

This section handles a proposed system at a very abstract level, exhibiting requirements, design concepts and other criteria.

3.1.1 User requirements and non-technical objectives

The following requirements are summarized in (but not limited to) the following points:

- Turn on or off any outlet or light at any given moment.
- Be able to extend Control via wire or wirelessly.
- The user must be able to talk with the computer and give voice commands.
- Know the current power consumption of the grid.
- Know the amount of water being consumed.
- Know the temperature inside and outside the building.
- Have alarms for detecting fires and smoke.
- Have alarms for Gas leaks and unhealthy air.
- Be notified via SMS at certain events.
- Have extended control via SMS.
- Set rules for automatic operation and nodes control
- View data and interface on TV screen.
- Watch movies using the interface on a TV screen.
- Lock and unlock the house via an electronic signature.
- To utilize power consumption
- Have the room turn off after becoming vacant.
- Have custom events and notifications for the sensory services
- Have activity tracking services, especially in a business or a shopping environment.

3.1.2 Other Design Criteria

The design aspires to fulfill the functionalities and requirements assumed for the user, but it should also fulfill the following Criteria:

- It must be relatively affordable to serve a larger number of users.

- It must be flexible enough to allow all the functionalities, or a sub group of them should the user not require all.
- The system existence should not affect the house existing infrastructure when disabled.

3.1.3 Basic System Abstraction

By looking at the most basic understanding of the system, and by addressing the design from a purely non-technical users' point of view, it is clear that the system must have the following components to achieve the required functionalities and user requirements, summarized in figure 3.1 below

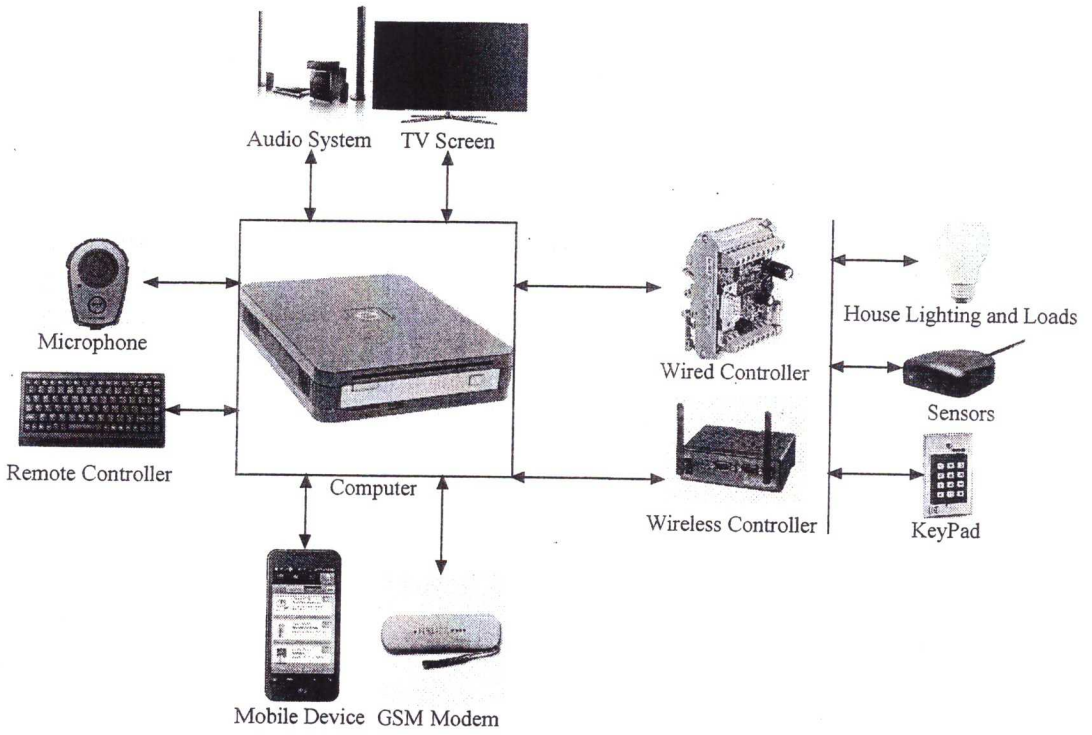


Figure 3.1 Basic System Abstraction

To achieve a good understanding of the proposed system from a technical point of view, the system must be characterized in different levels of abstraction and structure; thus the system is to be presented as two interacting systems, one deal with the computer software and the other one with hardware.

3.2 System Hardware Design

This section deals with hardware, where on a very basic level, the hardware system is divided to fulfill the wired and wireless functionalities.

3.2.1 Functional Requirements

The functional requirements expected from the hardware system are the following:

- The system must handle a large number of digital control ports to be interfaced with the control nodes laid within the control grid.
- The system must interface an adequate digital LCD screen.
- The system must provide ports for future expansion.
- The system should provide wireless electrical ports at request for control and monitoring.
- Required variation of the sensory node are: current sensor, water flow sensor, indoor temperature sensors, outdoor temperature sensor, water temperature sensor, CO sensor, CO^2 sensor, smoke detectors, and presence detectors.
- It must implement SMS functionality with a mobile network
- The system must interface a microphone or microphone array
- The system must interface an audio system.
- The system must interface one or multiple monitors.

3.2.2 Non- Functional Requirements

The nonfunctional requirements expected from the hardware system are listed as the following:

- It should consume very little power and be able to withstand long hours of operation.
- The hardware must operate under any windows OS starting from Windows XP or Higher.
- It should be able to interface with any local mobile network
- The system must be relatively easy to integrate with the Palestinian buildings' electrical systems.

- The system must provide flexibility in terms of required functionalities, and require only the minimum amount of dependency.
- The system must not have any negative effect on the electrical grid should it be turned off or disabled at any moment.

3.2.3 Hardware System Design Abstraction

The system can be modeled as a set of subsystems and the relationships between them, each subsystem must have the least amount of dependency, so in general it can be present as illustrated in figure 3.2 below:

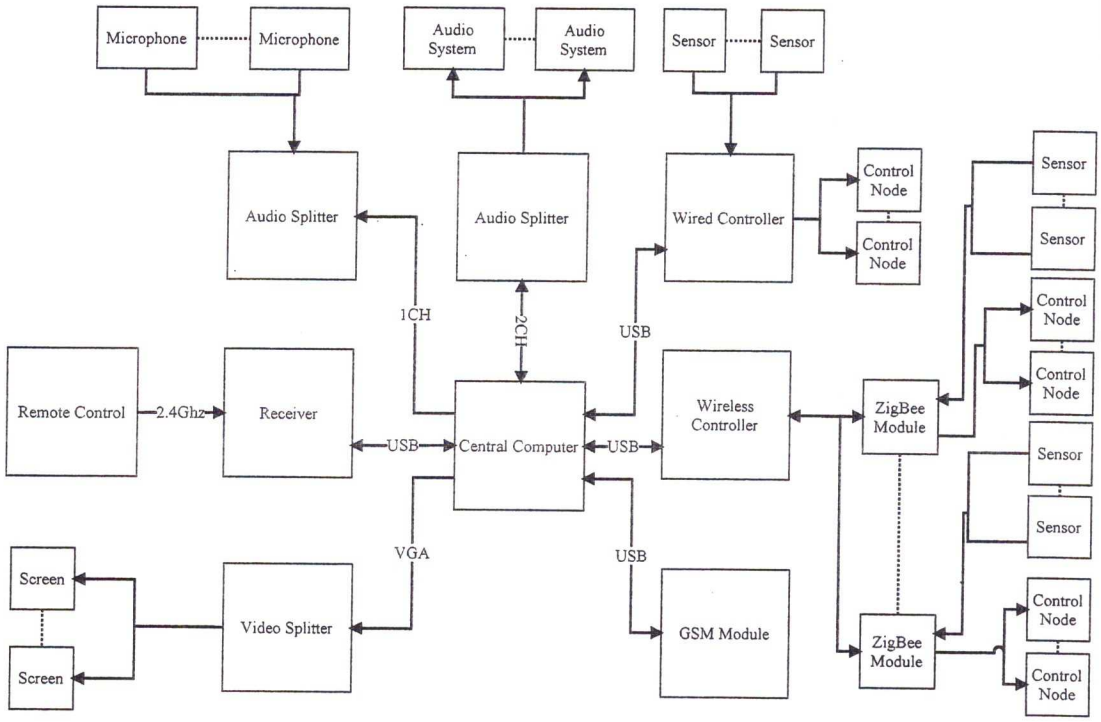


Figure 3.2 Hardware System Block Diagram

Decisions and commands are made in the central computer, produced using the software as a response to the user's needs, and then sent via the corresponding connection to the wanted module.

Some of the modules are interfaced directly from the computer, as it offers support for multitude of peripherals, and some require controllers and other mediating components and modules, such as the LCD which requires a controller.

The hardware assembly was intentionally designed in a way that conserves flexibility and the ability to change any component for upgrading and maintenance purposes, somewhat similar to Object Oriented Design, each component has its specific requirements, inputs and outputs with disregard to implementation.

3.2.4 Hardware Components and Design

The hardware system consists of the following components, starting from the center towards the peripherals:

3.2.4.1 Central Computer

The central computer operates a Windows OS, runs the software designed for this project and issues the commands for all the system. This computermust meet certain criteria to fit the application of this system, best summarized in the following requirements:

- It must be able to operate any windows OS starting from Windows XP or Higher.
- It must provide a minimum of 2GB of RAM for smooth operation.
- It must contain a minimum of 80GB Disk space to contain the OS, the system's software and database, and multimedia files.
- It must contain USB card.
- It must contain a standard audio card.
- It must contain a VGA port.
- It should consume very little power and be able to withstand long hours of operation.
- It must be small in physical size (form factor).

Other than that, the central computer behavior is addressed in section 3, as it runs the software algorithms and handles users' commands.

3.2.4.2 Wired Controller

The wired controller is one of the main hardware components of the system, as it is responsible for interfacing the actual function of automation and sensory network. The specifics of this component require the following functional requirements:

- It must provide 64 digital output ports to be interfaced with control nodes laid within the control grid.
- It must provide sufficient voltage levels required for the control array.
- It should be able to interface 16 analog sensors.

- It should be able to interface 16 digital sensors.
- It should be able to communicate with a computer via the universal serial bus, to transfer control and sensory data back and forth.
- It must be able to interface with an adequate digital LCD screen.
- It must provide a free 8 bit port with suitable control lines for future expansion.

The wired controller was designed with clear regard to the popular PIC microcontroller's capabilities and operation, and the following design structure was proposed to fill in the system requirements; the wired controller is interfaced from the computer side with a USB Connection, and then it interfaces the other peripherals via pin connectors.

Figure 3.3 illustrates the block diagram for the wired controller, where two microcontrollers take the command for operation, and the rest of the components exhibit interfacing elements.

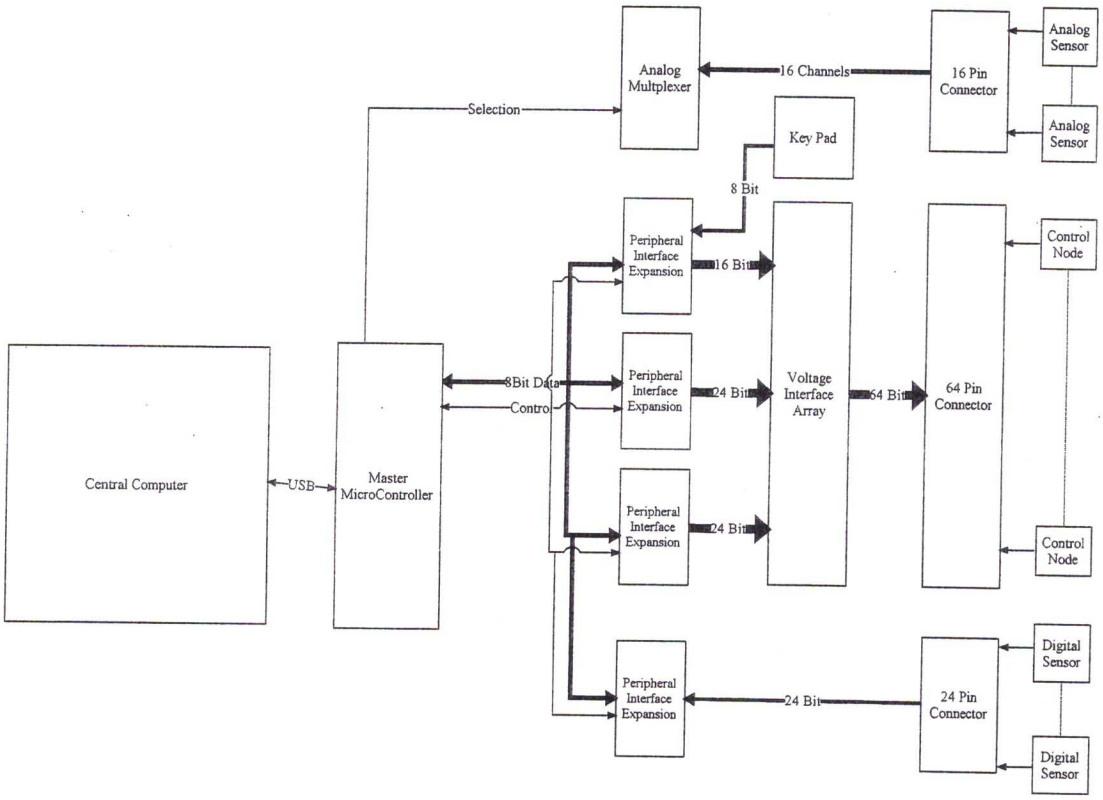


Figure 3.3Wired Controller Block Diagram

Basically, the subsystem in the previous block diagram is a set of 2 PIC Microcontrollers with a Master/Slave relationship. The master microcontroller's responsibility is to connect to the central computer via USB, and operate as a medium to the other circuitry; it receives commands from the Central Computer, decodes them, then sends the corresponding command to the required Peripheral via the slave microcontroller.

However, even with the two controllers' ports and capabilities; the system physical interfacing functionalities cannot be satisfied. So, peripheral expanders and analog multiplexers were interfaced.

Each module requires data transfer lines and control lines, so they were put with the design structure.

The components of the wired controller are the following:

3.2.4.2.1 Master Program

The software used to drive the master microcontroller is divided into a main logic and a set of subroutines, and two data buffer arrays, one for incoming data from the PC, and one for the data to be sent from the microcontroller.

- USB Communication :

With USB communication, the computer sends blocks of data each with 64 bytes of data, so with each USB transmission step, the microcontroller can send or receive 64 bytes.

In this system, the microcontroller is expected to receive a command key at the first byte of the received buffer array, and the following two bytes to identify additional information regarding that command. The rest of the array will not be used.

Table 3.1 Control Command Mapping

Index 0	Index 1	Identification
0	Don't Care	No Command
1 to 64	State	Set Control Node(Index0) to State

After each operation, the microcontroller is expected to send data regarding the state of the control nodes and the analog nodes, in an array called ToSendDataBuffer, with the following table explaining its structure:

Table 3.2 Echoback Data Structure

Echoback Data Array Mapping	
Index	Identification
0	ReceivedBuffer[0]
1	Not Used
2	AnalogPort 0 [0]
3	AnalogPort 0 [1]
(i+1) * 2	AnalogPort i [0]
(i+1) * 2 +1	AnalogPort i [0]
34	AnalogPort 16 [0]
35	AnalogPort 16 [1]
36	DigitalPorts 1 to 8
....	
44	ControlPorts 56 to 64
45	Digital ports 1 to 8
46	Digital Ports 9 to 16
47	Digital Ports 17 to 24

- Main program logic flowcharts :

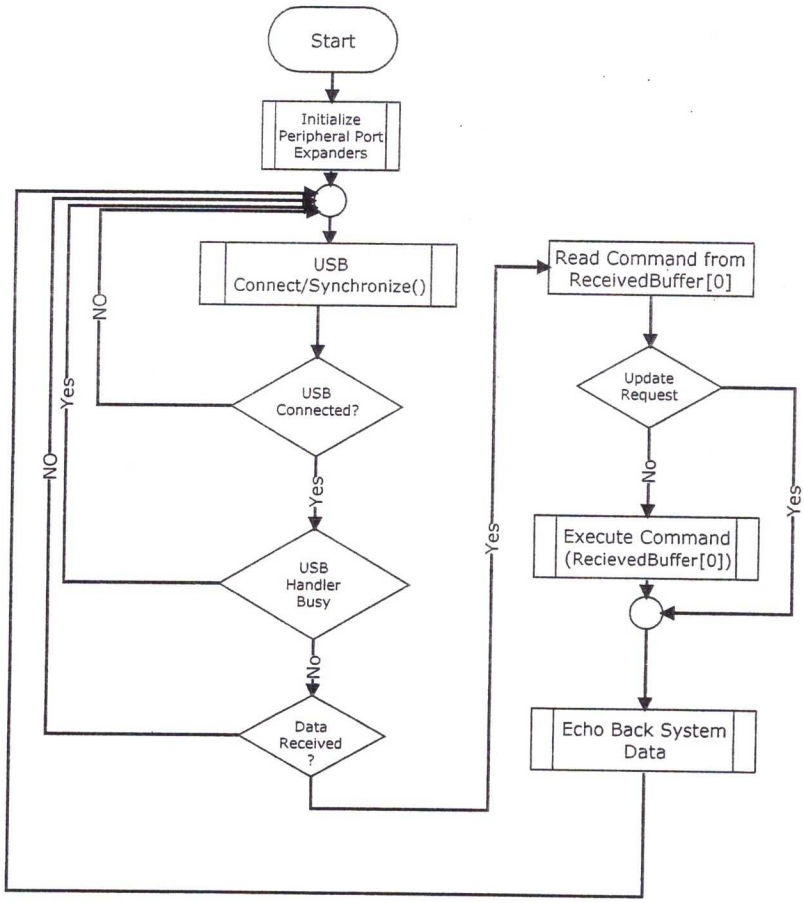


Figure 3.4 Main program logic flowchart

3.2.4.2.2 Peripheral functions

The software used to drive the slave microcontroller is also divided into a main logic and a set of subroutines, its primary objective is to deal directly with the ports and constantly read analog ports to have up to date information.

The slave microcontroller has an array of bytes that is identical to the one in the microcontroller. The program consists of the following main and subroutines:

- Execute command function Logic as shown in the figure 3.5:

```
Begin


if Command == Control
call SetPin()
else
if Command == Update
call SensoryChannelsUpdate()


end
```

Figure 3.5Execute Command Logic

- Sensory Channels Update subroutine

This subroutine updates the values of the sensory nodes and saves them within the data buffer as shown in the figure 3.6.

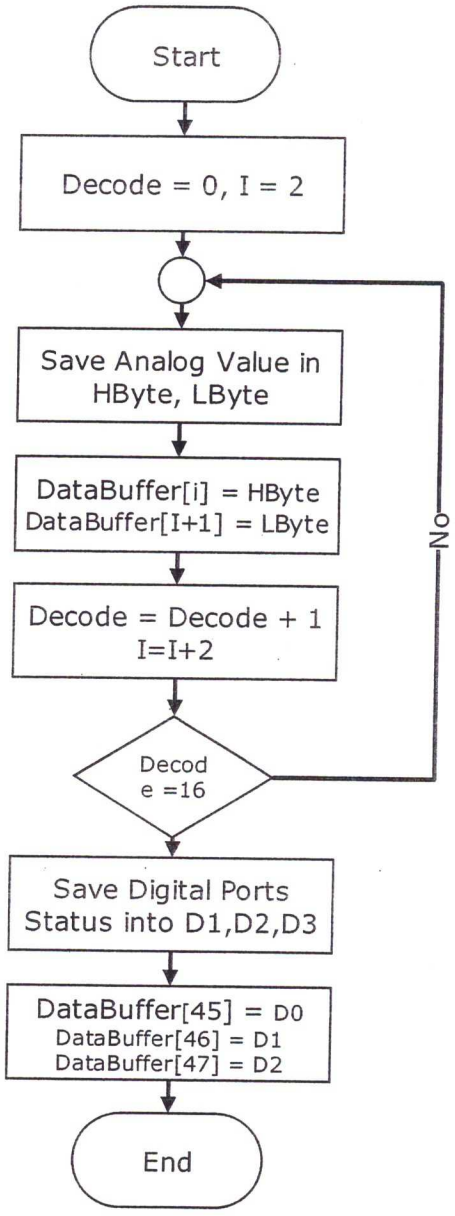


Figure 3.6 Sensory Update Flow Chart

- SetPin subroutine
  Due to the nature of this subroutine, a pseudo code would represent it better than a flow chart. It goes through the following steps:

Table 3.3 SetPin Subroutine pseudo code

```
BEGIN,

LOCATION = Databuffer[0]
STATE = Databuffer[1]

LOCATION = LOCATION – 1
TEMP = LOCATION / 8
If TEMP = 0
Get the first byte from the first Output Expander
If TEMP = 1
Get the Second byte from the first Output Expander
If TEMP = 2
Get the third byte from the first Output Expander
…
If TEMP = 7
Get the third byte from the third Output Expander
Set byte retrieved to WORD

TEMP = LOCATION % 8
WORD [ TEMP] = STATE

END
```

It should be mentioned that the combination of PIC18F4550 and 8255a was chosen for many reasons, most important of which is the availability of these chips in the local market, and the availability of programming and debugging tools for them. So the controller could be duplicated with minimal cost and effort.

Moreover, it's a cost effective combination that could produce a very large number of ports, working at high speed, and operate under the Universal Serial Bus.

3.2.4.3 Wireless Controller and ZigBee Network Modules

The system is supposed to be implemented wirelessly, and the ZigBee technology was chosen for this particular purposed; with that in mind, the network design becomes relevant on these two axis, where it is need to:

1) Design and implement wireless sensor network (WSN) using ZigBee technology with tree topology.
2) Design the process to gather data and send it to the computer.

3.2.4.3.1 Network Architecture

ZigBee wireless network technology will be used for the system, where the XBee series 2.0 radios represent the modules for the main networks nodes

The block diagram below shows the architecture of the network which consists of only one central node called coordinator surrounded by router nodes that are connected to end devices.

The topology used is a tree topology as shown in the block diagram in figure 3.7.

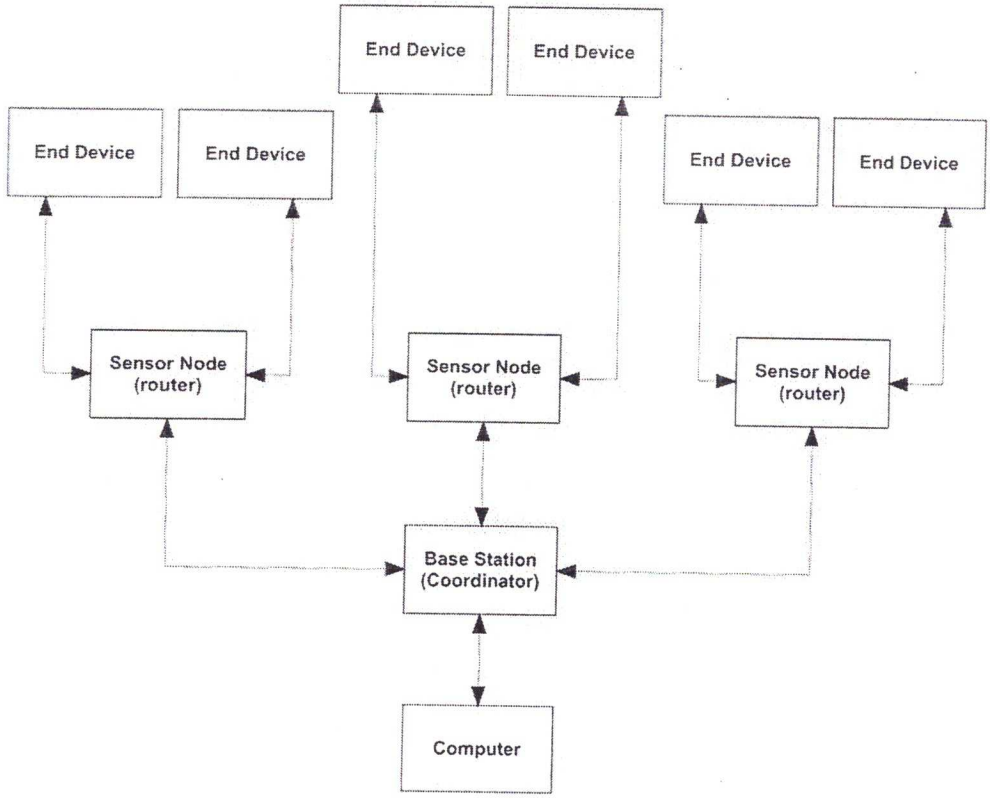


Figure 3.7 Network Topolgy and Structure

3.2.4.3.2 End Device Operation

The end device such as a sensor detects the surrounding environment status every certain periodic time and send the data to the processor, the processed data will be transfer through the wireless radio nodes to the computer.

Ad-hoc multi-hop routing will be used to support the ZigBee wireless communication. It uses a shortest-path-first algorithm with a single destination node and active two-way link estimation. The multi-hop router is essentially transparent to us and easy to transplant into our system; with the frame check and re-send mechanisms, we could make sure that all the commands and the monitoring data will be sent to the target device successfully.

3.2.4.3.3 Modules Operation

The XBee Modules interface to the microcontroller through a logic-level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible; or through a level translator to any other serial device or serial communication standard.

USB will be used to interface the coordinator (Tree Root) with the central computer, where it will behave as a standard communication port.

On the other hand, the wireless router modules are interfaced with microcontrollers using the standard RS-232 protocol as show in figure 3.8.

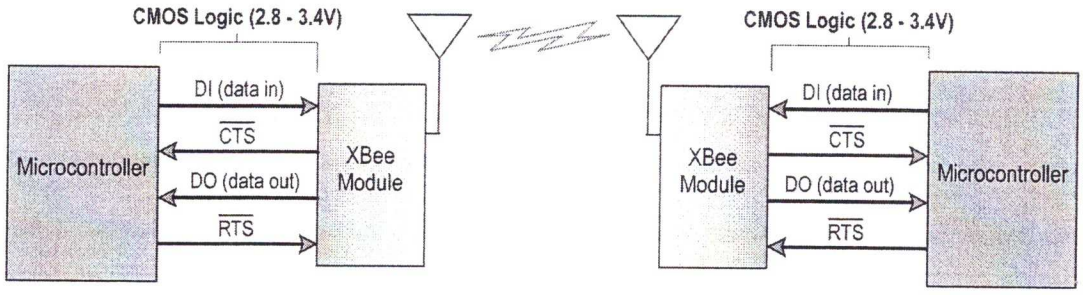


Figure 3.8 Data Flow Diagram for the XBee routers

Data enter and leave the XBee module through "Data in" and "Data out" pins as an asynchronous serial signal. Other signals such as "Clear-to-Send" and "Request-to-Send" are used to control the data flow. Then the XBee module exchange data through an RF link at a certain RF channel.

3.2.4.3.4 Serial Buffers

The XBee modules maintain small buffers to collect received serial and RF data flow, which is illustrated in the figure below. The serial receiver buffer collects incoming serial characters and holds them until they can be processed. The serial transmit buffer collects data that is received via the RF link that will be transmitted out the DOUT.

Figure 3.9 illustrates the internal data flow within the XBee module prior to send or after receive.

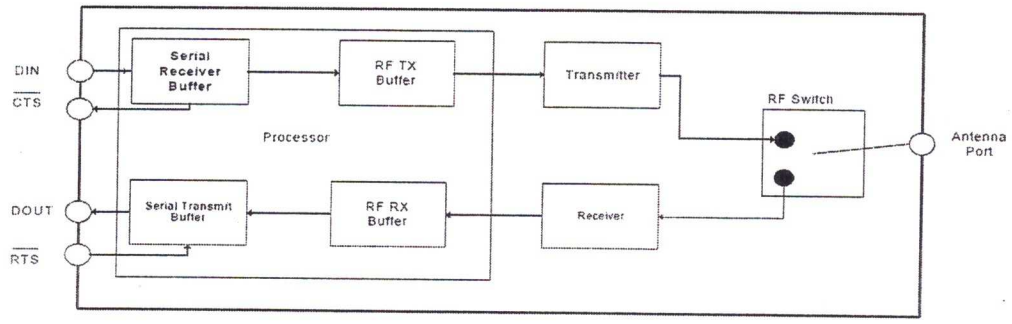


Figure 3.9 Internal Data Flow Diagram

The following flow chart in figure 3.10 illustrates the sensing process of the XBee module:

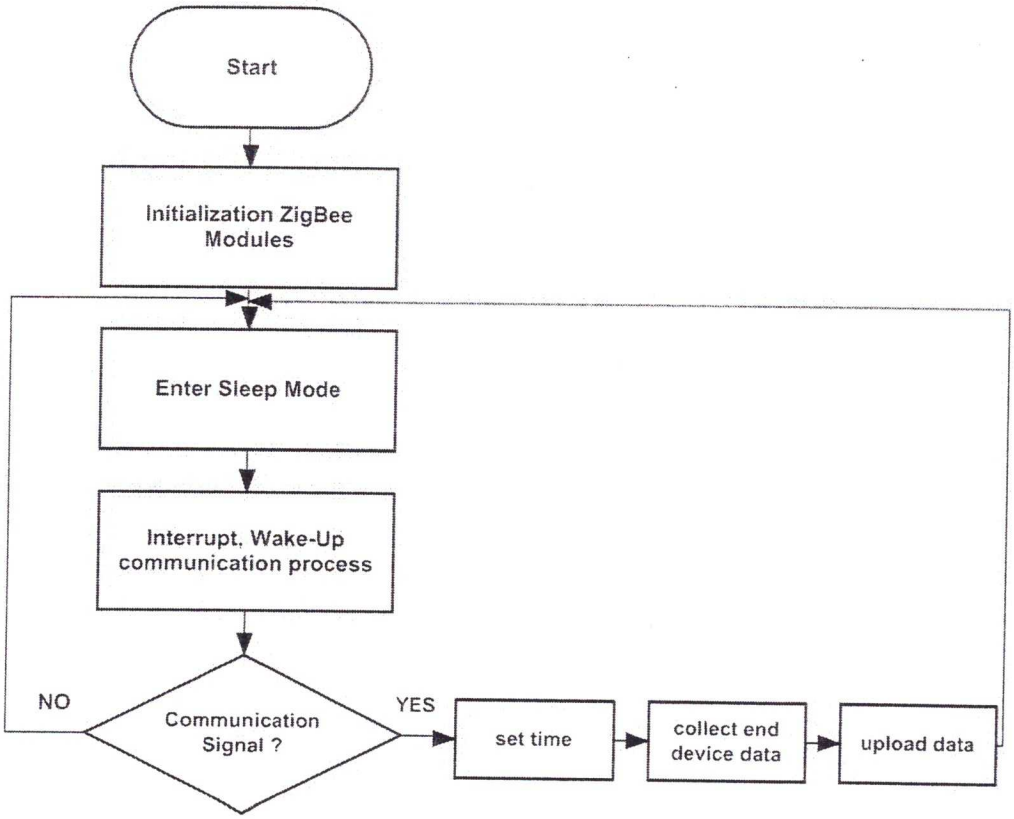


Figure 3.10 XBee Modules Sensing Flow Chart

3.2.4.3.5 Setting the RF Channel

The coordinator is responsible for starting a ZigBee networkas a PAN, the coordinator performs a series of scans to discover the level of RF activity on different channels (energy scan) and any other nearby operating PANs (PAN scan). It performs an energy scan on multiple channels (frequencies) to detect energy levels on each channel to avoid starting on channels with high energy levels. Channels with excessive detected energy levels are removed from its list of potential channels to start on.

Figure 3.11 illustrates a list of channels potential for operation and illustrates some eliminated channels.

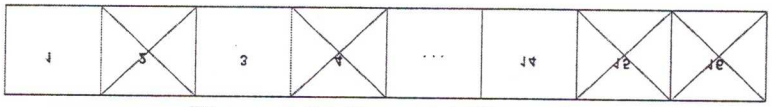


Figure 3.11 Potential Channels

3.2.4.4 GSM

GSM modem connects with the computer through a serial USB port, and the computer sends AT commands to the GSM Modem to control its data transmission; it is used in our application to send and receive SMS Messages.

The following diagram in figure 3.12 illustrates the structure of the GSM Module:

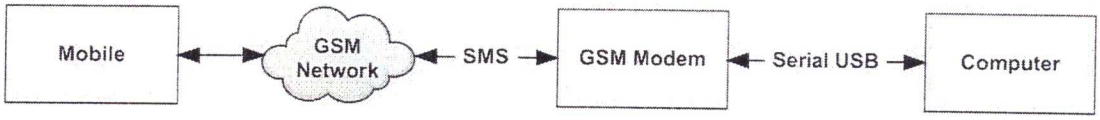


Figure 3.12 GSM Modem Structure

The system will send SMS using the GPRS modem about the building status; it will send SMS automatically in emergency situation such as fires.

The following general block diagram illustrates the data flow form the computer or the mobile through the GPRS modem

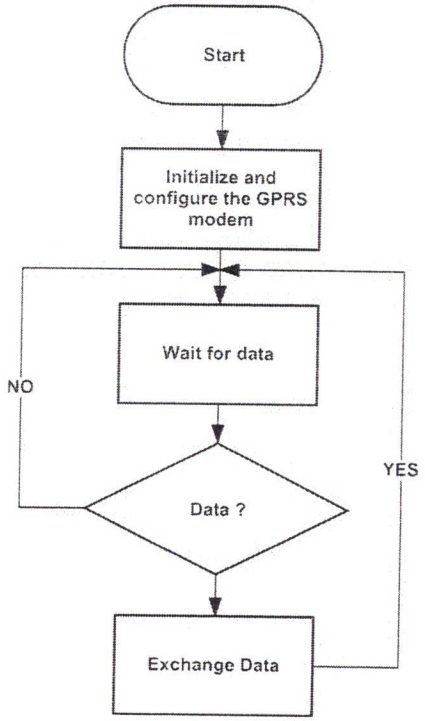


Figure 3.13 GSM Module GeneralOperation Flow Chart

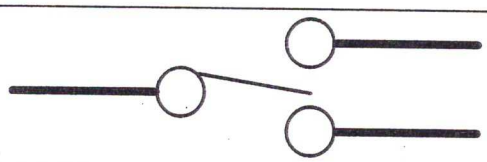
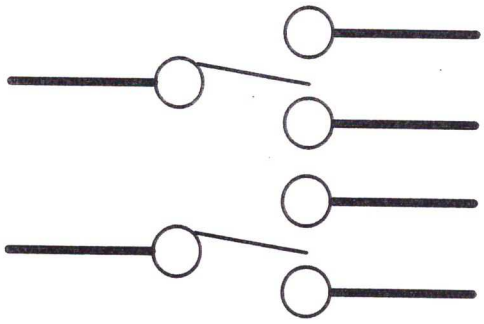
3.2.4.5 Control Node

The control node must operate as a switch that operates based on a computer signal, it should satisfy the following requirements:

- It must operate as a digital switch on a high voltage line.
- Its existence should not affect the normal behavior of the voltage line when disabled.
- It must be able to receive digital commands from the central controller.
- It must withstand a suitable operational current.

Picking from a multitude of candidate technologies, the most plausible candidate for the following function is a power relay. The specifications for a power relay that satisfies the design requirements are depicted in table 3.4..

Table 3.4 Design Specifications for Relays

Criteria	Suitable Values	
Type	 Figure 3.14 SPDT – Single Pole Double Throw Relay  Figure 3.15 DPDT – Double Pole Double Throw Relay	
Connection State	Normally Closed	
Primary Voltage/Current	Lighting Loads Main Lines	220VAC/10A 220VAC/20A 220VAC/40A
Controlling Voltage	5V or 12V or 24V	

Any relay with these criteria and values can be used in this application; other criteria for a relay include energy consumption, affordability.

The user can use relays such as low power Mechanical Relays, and SSR Solid State Relays.

The block diagram in figure 3.14 illustrates the mechanism a relay is controlled and located.

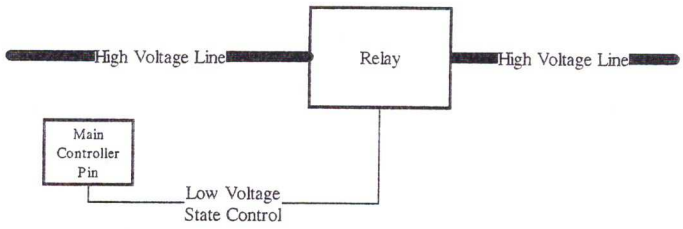


Figure 3.14 Control Node Block Diagram

The control node will be used to control:

- House lighting
- Loads and Main Voltage Lines
- Window shutters
- Doors and gate lock
- Any other digital devices operated with a switch

3.2.4.6 KeyPad

An interrupt driven 12 Key Keypad will be used in this subsystem; data will be transferred via an 8 bit line to identify the corresponding pressed key. They keypad must have the following specifications presented in table 3.5:

Table 3.5 Key Pad Specifications

Criteria	Value
Operation Voltage	5 Volts
Number of Keys	12 Keys, 12 Keys
Other	Interrupt driven operation

The internal structure for this component requires the presence of a line scan mechanism, which will be utilized using either an external chip, or by using a main component within the main controller.

3.2.4.7 Digital Sensors

The digital sensors addressed in this section, are those sensors that produce a value of 0 or 1, mostly they're motion detectors, presence detectors, and Hall Effect sensors and Smoke detectors.

The following design diagram fits all the digital sensors, however, the internal structure of the conditioning circuitry depends on sensor type, and thus more will be written when implementing these sensors in chapter 4.

Figure 3.15 illustrates a block diagram for any digital sensor:

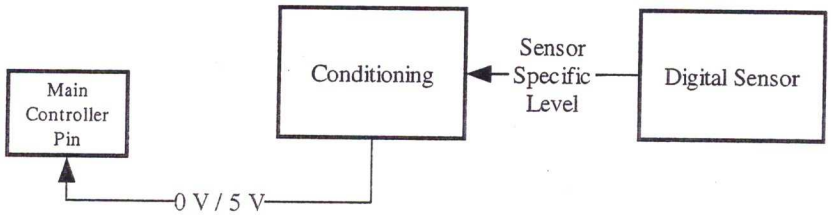


Figure 3.1516 Digital Sensors (Presence)

3.2.4.8 Analog Sensors

The Analog sensors addressed in this section, are those sensors that produce an analog range of values, with a max and min value, mostly temperature sensors, Gas sensors, current transducers and water flow sensors.

3.2.4.8.1 Current Transducers

Current Transducers provide real-time data about the amount of current going through an active voltage line, it is used to calculate the power consumption of loads on a selected wire.

One major advantage of using the selected current transducers is that they provide a 0V to 5V analog signal suitable for a microcontroller, so no conditioning is required as shown in figure 3.16.

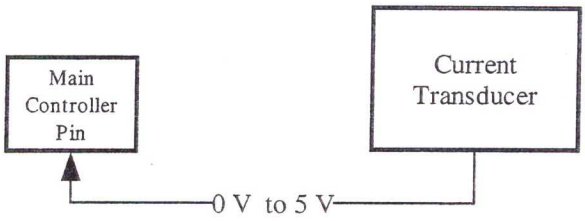


Figure 3.176 Current Transducer Block Diagram

The signal is assumed to be linear, and is conditioned in higher systems to represent its real value.

3.2.4.8.2 Water flow sensors

Water flow sensors produce values that are proportional to the flow of water within a water tube; it is used to calculate the amount of water going through a buildings water supply line.

Water flow sensors, are frequency generation sensors, they generatedigital pulses with a frequency that represents the actual environmental value,so it has to be conditionedto an analog signal so it can be handled by the main controller.

The following block diagram illustrated in figure 3.17 presents the interfacing structure of the water flow sensor:

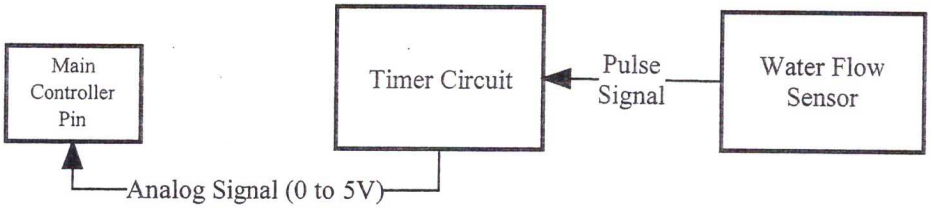


Figure 3.17 Water Flow Sensor Block Diagram

3.2.4.8.3 Temperature Sensors

Temperature sensors will be used to calculate the temperature within certain rooms and on the outdoors;this application will use standard temperature sensors that provide real-time information in the shape of analog voltage value:
Figure 3.18 exhibits the block diagram for the Temperature Sensor.

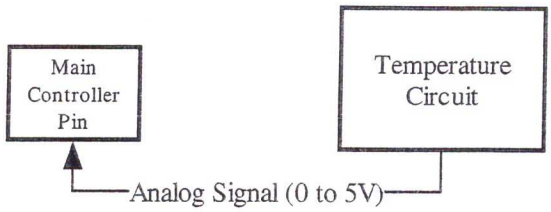


Figure 3.18 Temperature Sensor Interface Block Diagram

3.2.4.8.4 Gas Sensors

Gas Sensors, can be abstracted as a varying resistance with regards to the amount of gas in the air, in this application, a simple voltage divider method will allow us to calculate the amount of gas.

Figure 3.19 exhibits the general block diagram for the Gas sensor.

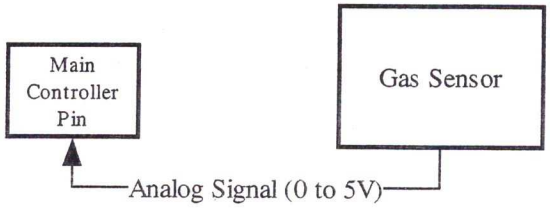


Figure 3.19 Gas Sensor Interface Block Diagram

3.2.4.9 Remote Control

The system will use wireless mini keyboard as a remote controller, the main advantage of this choice is the fact that it is an off-the-shelf component with huge benefits with regards to usability, the number of keys, and its ability to be used as a normal keyboard for other applications.

Figure 3.20 illustrates the general structure for the Remote control:

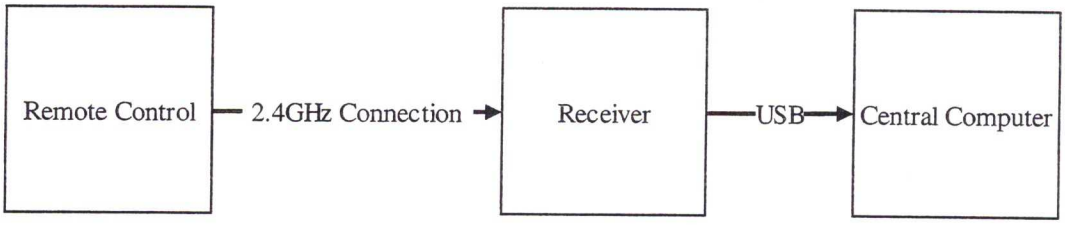


Figure 3.18 Remote Control Interface

3.3 Software System Design

This section explains the expected design of the software components, classified and characterized according to function, and with the intention to keep dependency to the minimum.

3.3.1 Functional Requirements

The software within the project has the following functional requirements:

- The software must be able to communicate with the Central Controller, the wireless module, and the Mobile module.
- The software must be able to process data taken from the hardware, and convert it into suitable understandable data.
- The software must be able to take actions independently within certain events such as a suspected unauthorized entry, fire or any other predetermined case.
- The software must provide an event driven architecture (Event, Condition, Action) as a set of rules for dealing with the ports values.
- The set of rules should be dynamically set and can differ depending on the state of the system.
- The software must have an attractive graphical user interface suitable for TV and home deployment.
- The software, being meant for a TV, must provide video and multimedia support.
- The software must be able to adequately present the data taken from the hardware analog input.
- The software must be able to recognize voice commands
- The software must be able to synthesize speech at certain at certain events.
- The software should provide a universal interface for commands related to the system hardware, which can be accessed by any type of controller.
- The software must keep track of and organize all of the systems data and keep and be able to restore them at the user's request.
- The software must have a higher level of understanding for each port (Control/Sensory) within the controller, with regards to its static parameters (Default Status/Physical attributes), and dynamic parameters (Current Status and value).

- Display data in graph formation and distinguish between the different classes of sensory data.

3.3.2 Nonfunctional Requirements

The software also has the following nonfunctional requirements:

- The software should be as reliable, and as robust as possible.
- The software must hold object oriented design principles, so that components can be independently built.
- Objects must have a limited amount of dependency.

3.3.3 Software System Design Abstraction

The software can be modeled as a set of objects each performing a specific task with disregard to the main logic. it is taken in regards that each subsystem must have the least amount of dependency, so in general it can be can present as shown in figure 3.21.

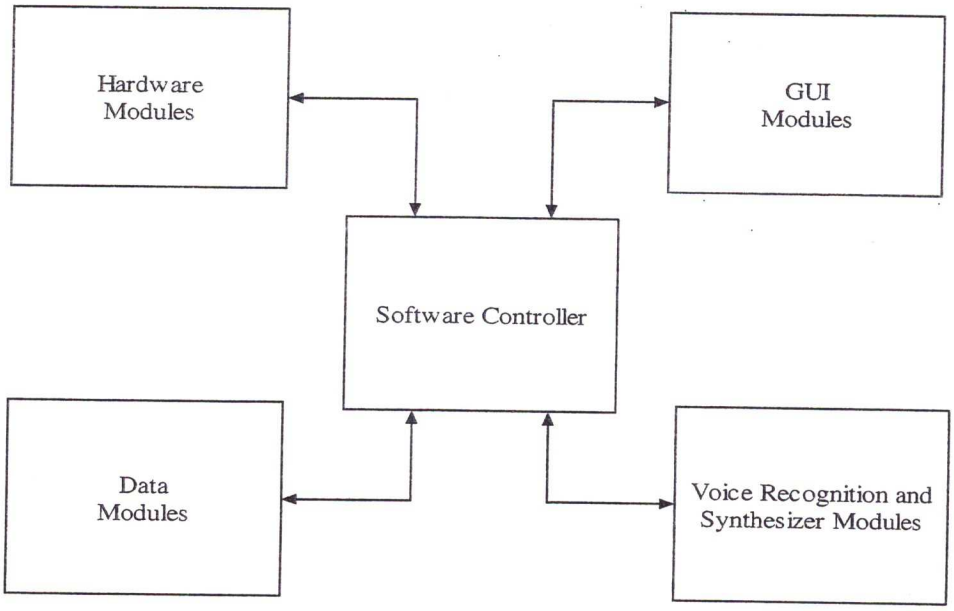


Figure 3.21 System Abstraction Block Diagram

Basically, the software controller takes control of the interfacing requirements of the system, and is the main front to handle all commands, tasks, and data transactions. The rest of the system's modules are classified in 4 module types: GUI Modules, Hardware Modules, Data Modules, Voice Modules

3.3.4 Software Components and Design

The following components and subcomponents represent the individual functionalities required from the software:

3.3.4.1 Hardware Modules

3.3.4.1.1 PIC USB Driver

This component is a serial communication module; by having a device id, it can scan the local windows system for an attached device with a matching id, the commence communication with it. This will be used to communicate with our wired controller, which uses a PIC microcontroller to interface with the computer.

This module will use the "Generic Human Interface Device" open source framework available from Microchip Technology Inc., more commonly known as "Generic HID". It is chosen for this project because it allows a very general specification for the communication process, and it can be configured so that any computer can read it.

Using a UML Diagram, this component can be represented as shown in figure 3.22.

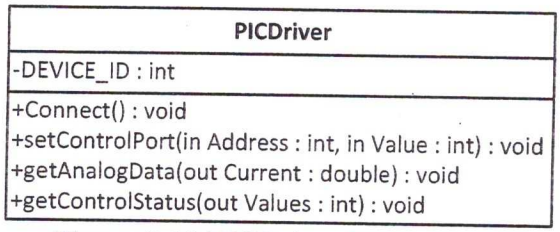


Figure 3.19 PICDriver UML Class Diagram

Mainly, It's an object contain a Device ID, and 4 main function:

- Connect: This function scans the windows registry, finds the device with the corresponding device ID, then establishes a connection with it.
- setControlPort: This function sets a control ports with the corresponding address to the accompanying value.
- getAnalogData: Retrieves a list of float type data, that represent the status of the analog ports.
- getControlStatus: Retrieves a list of int type data, representing the status of each control port.

3.3.4.1.2 SMS Serial Module

This component implements a standard serial communication program, it contains information regarding the port number, parity bits, stop bits, the number of bits for data transfer. In addition to that it contains functions specific to the SMS functionality, and functions to send and receive SMS as shown in figure 3.23 .

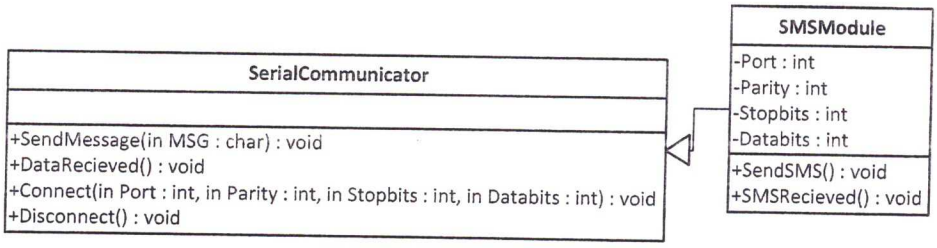


Figure 3.23 SMS Module UML Class Diagram

The main components for this module are:

- Connect: This function will instantiate a new port connection and sets its configurations, then establish a connection.

  It will also send the following command to put the Hardware SMS module in text mode:

  AT+CMGF=1

- Disconnect: This function will end the serial communication across the specified port.
- SendMessage: This function will take a string as an argument, and then send it across the port connection.
- DataReceived: this function will be called whenever a message is received and notify the object.
- SendSMS: This function will mainly use AT Commands to communicate with the GSM Modem, the following command will be used within the software to send an SMS to a phone number:

  AT+CMGS="<Number>", <MessageString>→

- SMSReceived: this function will be called to identify whether a valid message has been received or not.It will then process the message and send the data to a higher level object.

  It will begin by sending the following command:

  AT+CMGL = "REC_UNREAD"

This will retrieve all unread messages, so the function can process the most recent one.

3.3.4.1.3 ZigBee Serial Module

This component implements a standard serial communication program, it contains information regarding the port number, parity bits, stop bits, the number of bits for data transfer. In addition to that, it contains a list of device IDs , addresses, and a list of functions for to provide accessibility to the wireless modules.

The figure 3.24 represent UML diagram for the ZigBee Module:

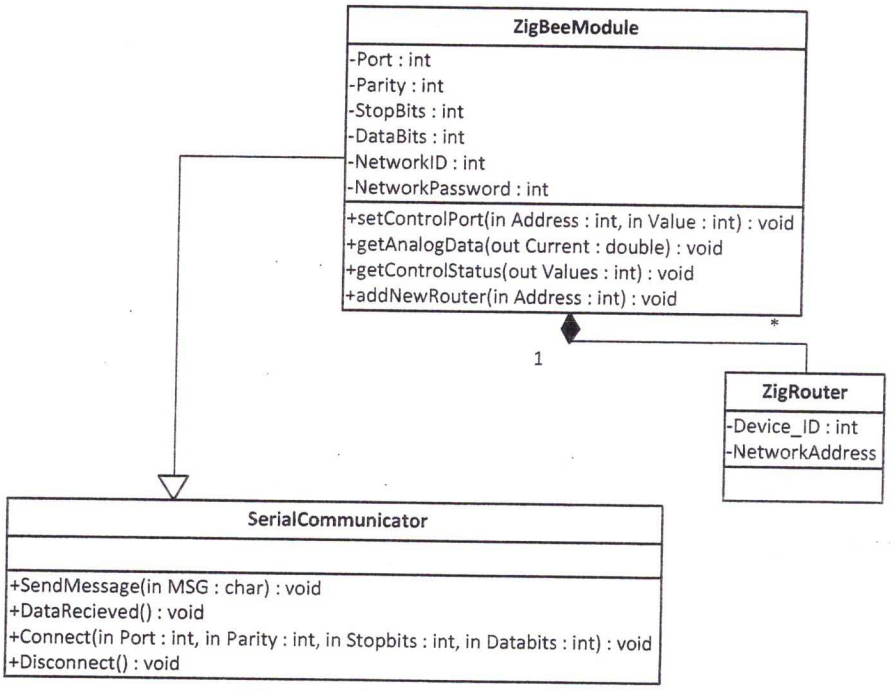


Figure 3.24 ZigBee Module UML Class Diagram

The main components in this UML diagram are:

- NetworkID : the ZigBee network has a specific ID that must be specified in the software.
- NetworkPassword: the system can provide a password to prevent alterations and any unwanted intervention
- setControlPort: This function sets a control ports with the corresponding address to the accompanying value.
- Identify: This function identifies received data and categorizes the information received in data structure representing them

3.3.4.2 *Socket Communication Module*

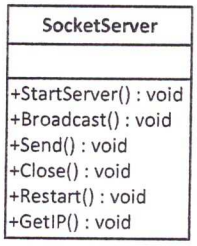
This component implements a socket communication application; it registers devices that request any connection over an Ethernet network, where this allows control from other computers and smart devices on WiFi or any other LAN connections.

To implement control using sockets, the system will have to act as a server and await commands from clients. This means that a client application will also be implemented with regards to compatibility with the server.

3.3.4.2.1 Server module

The server module will have to present a unified multi-threaded interface for various clients where they can query and control the system. Each client must be registered and dealt with upon exit. Moreover, simple security procedure should be implemented as a proof of concept, so they can pave the way for further development in the future.

The Server will have to include the following methods, with the following functions described in the UML diagram in figure 3.25 and the description:



The figure 3.25 represent UML diagram for the ZigBee Module:

- StartServer: This function starts the server and awaits for connection requests from clients, it should monitor each client in a thread, and register it's
- GetIP: This function should return the IP of the server device so it can be used within a client
- Void Broadcast: This function sends a message to all connected clients, in string format.

- Send: This function sends a single message to a single client identified by its index.
- DataReceived: This function is called whenever a block of data is received, it parses it to a string, and also it passes an index identifying the client sending the message. This function is to be overloaded above the object's level, so it can execute other functions.
- SendAnalogData: This function provide an interface to report analog data for sensors of the system.

3.3.4.2.2 Client module

The Client module will have to be implemented on a android environment that uses the JAVA programming languages and the Android mobile API, for that a android device is needed; the program needs to implement the basic client communication functions.

The following figure 3.26 is the UML diagram for the client module implemented on the android environment:

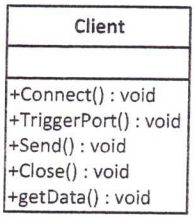


Figure 26 client module

The functions are described as following:

- Connect: This function starts the server and awaits for connection requests from clients, it should monitor each client in a thread, and register it's
- TriggerPort :this function sends a message to the server triggering a control node to a specific state
- Send: This function sends a message to the server
- getData: This function retrieves analog data

3.3.4.3 Graphical User Interface Modules

The following components must be built using any graphical user interface API, and they require the following specification and design:

3.3.4.3.1 Home Screen

This window screen must show the current environmental status of the building, provide a navigation interface, and provide the ability to do some commands.

Essentially, this object described in this UML, must have functions that update the GUI according the status of the environment in the building.

The figure 3.27 represent UML diagramfor the HomeSecreen module:

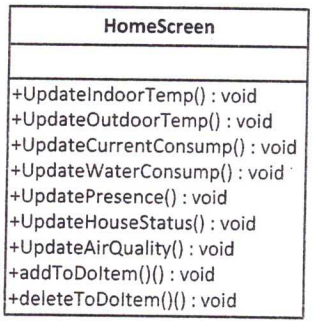


Figure 3.27 HomeScreen Window UML Class Diagram

3.3.4.3.2 Movie and Picture Screen

These windows screens must show a list of movies and pictures for the entertainment of the user, and provide a user friendly interface capable of adding new films and picture to the current list of movies or pictures.

This modules is represented in figure 3.28 as a UML Class Diagram:

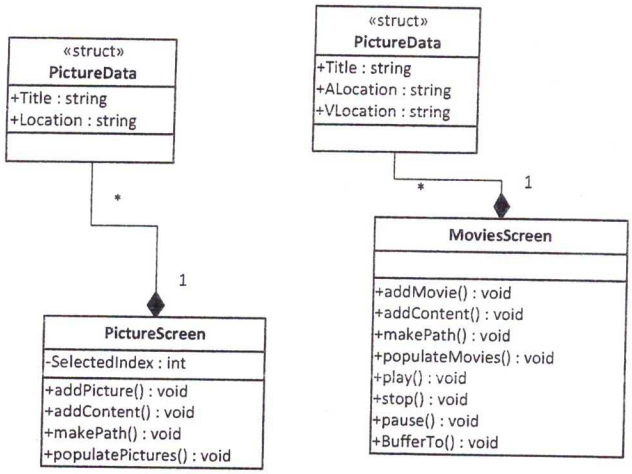


Figure 3.28 MovieScreen and PictureScreen UML Class Diagram

3.3.4.3.3 Settings Screen

This window screen provides an interface to show, add, and change the event system of the software; moreover, it must provide an interface to allocate the ports so they can be accessed, whether it be a control port or a sensor port.

In Figure 3.29, Settings Screen UMLmodel is represented as a set of functions:

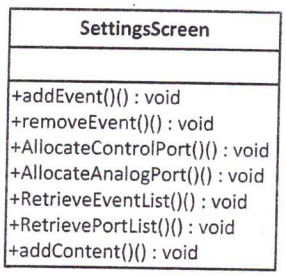


Figure 3.92 SettingsScreen UML Class Diagram

The functions in the settings Screen should access the GUI components and produce the function they're supposed to do:

- addEvent: Creates a new event, then saves it
- removeEvent: Deletes an existing event
- AllocateControlPort: allows accessibility for an existing control port
- AllocateAnalogPort: allows accessibility and monitoring for an analog port.
- RetrieveEventList: fills the GUI with a list of events.
- RetrievePortList: fills the GUI with the full list of the systems port, both control and analog ports.

3.3.4.3.4 Data Screen

This window displays all the data the user may require for the sensors and analog ports in both Graph and Table form on multiple scales, meaning it can show the varying values over different timespans, using different time units.

The Figure 3.30 describes the structure of this window:

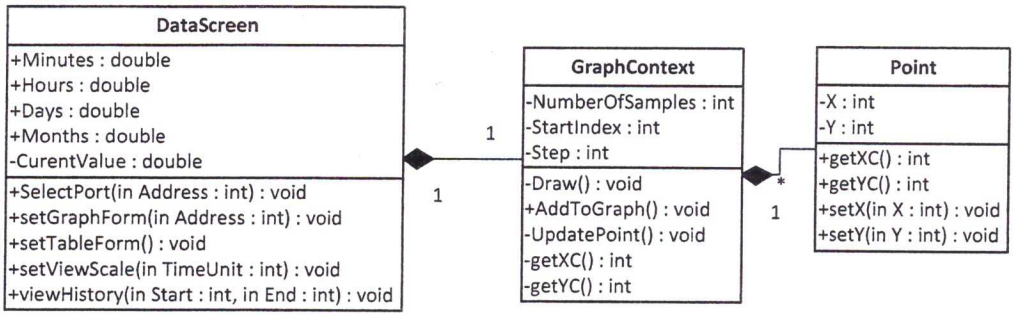


Figure 3.30 DataSceen UML Class Diagram

DataScreen contains the following functions to fulfill its behavior:

- SelectPort: displays data specific to the selected port
- setGraphForm: changes the GUI to Graph form
- setTableForm: changes the GUI to Table form
- setViewScale: sets the scale of data with respect to time, whether it be months, days, or even hours.
- viewHistory: takes two date representing arguments, then displays a range of data within that time period.

For the tables, use any graphical table component provided by the implantation language will be used, as for the Graph, a specific implementation must be included.
The DataScreen contains the following lists:

- Minutes: 60 Elements
- Hours: 24 Elements
- Days: 60 Elements
- Months : 12 Elements

3.3.4.3.5 Control Screen

This window should allow the user to access the control ports within the house, it characterizes outlets and switches within rooms and aorders the according to class and function.

Figure 3.31 represents the general structure of the window, where each control port is allocated according to level and function.

Level 0 List	Level 1 List			Level 2 List		
	Currently On	Currently Off	All	Currently On	Currently Off	All
Lighting Loads Heating System "Custom"	Bedroom Living Room Custom			Outlet 1 Outlet 2 Custom		

Figure 3.31 Window Structure

3.3.4.4 Voice Modules

The following modules introduce the intended design for all functionalities regarding voice recognition and synthesis:

3.3.4.4.1 Voice Synthesizer

This object generates speech, and allows modifications on the voice characteristic such as culture, age, and gender. As a composite of the voice module object, the voice module receives text from voice listener and sends to voice synthesizer to speak it.

Figure 3.32 is the UML class diagram of this module:

Voice Synthesizer
-Text : void
+Speak(in Text : string) : void +SetPitch(in Pitch : int) : void

Figure 3.32 UML Class Diagram

The following functions are required within this module:

- Speak : To generate speech from a string, the string was taken from voice module object .
- SetPitch: set the characteristics of the voice synthesizer or speaker.

3.3.4.4.2 Voice Listener

This module is listening to the user continuously and it works simultaneously with the Voice Module .it contains a dictionary that define all the system commands .

The dictionary works as a series of valid alternatives, , each alternative is set of choices, and each choice is set of strings. This composition allows recognition for complex voice commands, and allows the generation of integers representing each specific command.

Figure 3.33 is the UML diagram of this module:

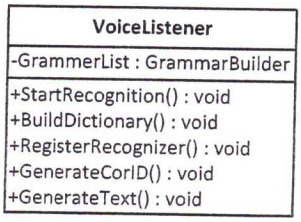


Figure 3.3320 Voice Listener UML Class Diagram

This window require the presence of the following functions:

- GrammerList: a complex set of strings for the voice commands.
- StartRecognition: initiates the voice engine with the built dictionary.
- BuildDictionary: Generate the grammar required for the voice command set and bind it with the voice engine.
- RegisterRecognizer: Binds the voice engine with an audio input stream.
- GenerateCoreID : this function generates an integer to represent a corresponding voice command.
- GenerateText : this function returns the full text for the recognized voice command.

3.3.4.4.3 Voice Module

This module is the link between the Voice Listener, Voice Synthesizer, and any higher tier object as it provides a proper interface for all the required voice recognition and synthesis commands.

Figure 3.34 shows the UML diagram and lists the components of this module :

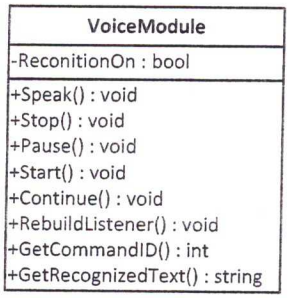


Figure 3.34 UML Class Diagram for the Voice Module

These components can be abstracted as the following:

- RecognitionOn: an attribute that identify if the system listening to the user or not .
- Start: a function that initiate the voice listener and voice synthesizer module and then start listening to the user .
- Speak: a function that speaks the entered text .
- Pause: a function that pauses the system from listening to the user while it speak the last command entered by the user .
- Continue: a function that allows the system to continue listening to the user after complete speaking the command .
- GetCommandID: a function that take the string command and make a level of analysis by comparing it to the commands that stored in the dictionary , and then return the ID of the commands .
- GetRecognizedText: a function that return the voice command as a string .

Tabel 3.6 showsthe list of commands required for voice recognition and their proposed location for various commands:

Table 3.6 Voice Command Lists

Weight	x 10^5	x 10^4	x 10^3	x 10^2	x 10^1	x 10^0
1	Turn	On	Light	One	Room	<Number>
2		Off	Socket	Two	Hall	
3			Heating	Three		
4			TV	Four		
5	Play		music	Five		
6	Stop			Six		
7	Night		mode	Seven		
8	Day			Eight		
9	Morning			Nine		
10	surveillance			Ten		
11	Home			Eleven		
12	Movies			Twelve		
13	picture			Thirteen		
14	Home	Lock		Fourteen		
15	Data	Unlock		Fifteen		
16	settings			Sixteen		

17	control			Seventeen		
18	volume	Up		Eighteen		
19	shutters	Down		Nineteen		
20	weather			Twenty		

Each word has a single decimal value that is weighted according to location within the voice command.

For example, the command "Turn On Socket Three Room One" would generate an integer like 112311 as shown in figure 3.37

Turn	On	Socket	<Number3>	Room	<Number1>
1	1	2	3	1	1

Figure 3.37 Voice Command Example

3.3.4.4.4 Voice Interface

This object is meant to provide accessibility to the voice module in a way that is compatible with the action list within the main logic, it inherits the voice module , and then provides general voice interface functions that would control the voice module, and it maps an internal commandID to the appropriate action.

Figure 3.38 show the UML Diagram and lists the components of this module:

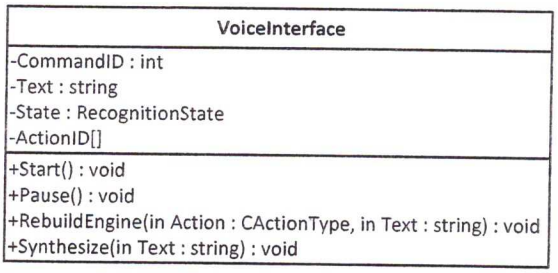


Figure 3.218 Voice Interface UML Class Diagram

An abstraction of the required functions is listed as the following:

- CommandID and Text: storage for the most recent command recognized by the module
- State: an attribute used to identify the current status of the module, whether its on, off, or paused.
- RebuildEngine: a function used to build the dictionary within the voice engine at startup and at any new voice command admittance, and tie it with the current action list.
- Synthesize : a function that provide an interface to the voice synthesizer, it tells the module to speak a certain text.

3.3.4.5 Data Modules

The following modules provide a the data warehouse for all information and command flow within the software. The access to this information is regulated by the main software controller.

3.3.4.5.1 Control Ports Data

The system implements a multitude of hardware device, each with different configurations that are not always utilized by the user; so, only ports that are registered and properly configured are allowed for user access.

Each port is given an ID and a logical address, so it can be properly referenced from anywhere within the software. Moreover, there are other attributes are given for each port such as the associated rooms, state, class, level, and the parent port (which has to be one level lower than the current node).
Figure 3.39 is the UML Diagram for this object and it summarizes the attribute list:

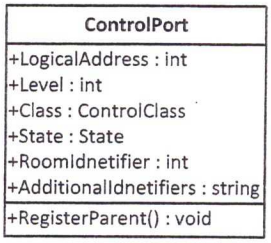


Figure 3.39 ControlPort Data Structure UML Diagram

3.3.4.5.2 Sensor Port Data

The sensor nodes store data regarding each sensor nodes, such as the logical address of the sensor port, and other information regarding its type and classifications. Moreover, the sensor port data nodes, are inherited by two other types of object that differ by classification, one is a standard sensor, and the other one is a detector that holds only one value that is useful at the current moment.

Figure 3.40 shows the Sensor Data Structure UML diagram and the relationship between tis different classes.

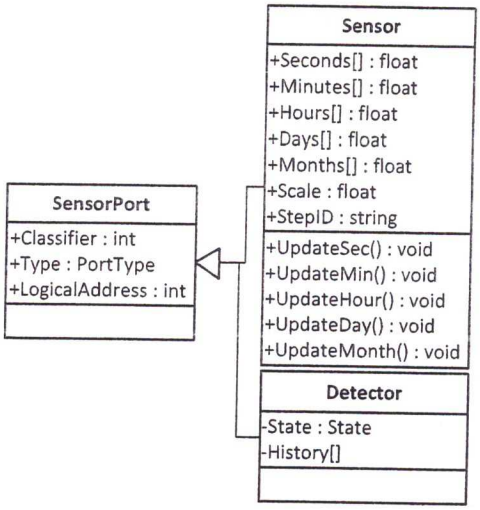


Figure 3.22 Sensor Data Structure UML Class Diagram

3.3.4.5.3 Events System Data

The software implements an event system for independent operation; the presence of such events allows the user to customize and specify the operation of the hardware as a set of rules that is saved and executed should their conditions be meet.

Figure 3.41 shows the sequence diagram that explains the expected functionality of the event system:

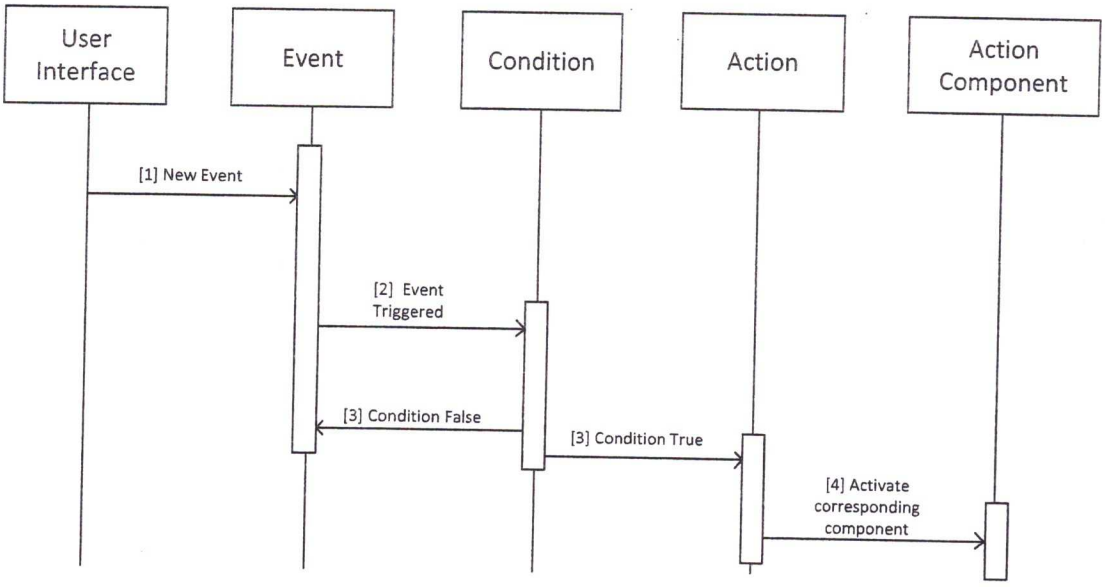


Figure 3.41 Event Sequence Diagram

3.3.4.5.3.1 Events

Events occur through the runtime of the system, these events must be identified, characterized, and used to implement certain actions under certain conditions. For this specific software design, table 3.9 shows the type of events occur the system, with the following corresponding types of conditions:

Table 3.9 Events and Conditions Characterization

EventType		
Event		Condition
Control Port	Triggered	Equals {0,1,ANY}
Sensor Port	Checked	Equals <float>
		Larger than <float>
		Smaller than <float>
SMS	Received	Equals <String>
		has <String>
Key	Pressed	Equal <Char>
Voice Command	Recognized	Equals <String>

Each one of these events is triggered at a certain point, for example, sensor ports events are triggered when the system retrieves data from the hardware; then if the conditions are meet, an action is implemented.

Basically, each event has an automatically generated identifier, a type, a condition, and an action. Moreover, an event is inherited by other events that have customized information regarding that event.

Figure 3.42 shows the UML class diagram which represents the structure of the event data type:

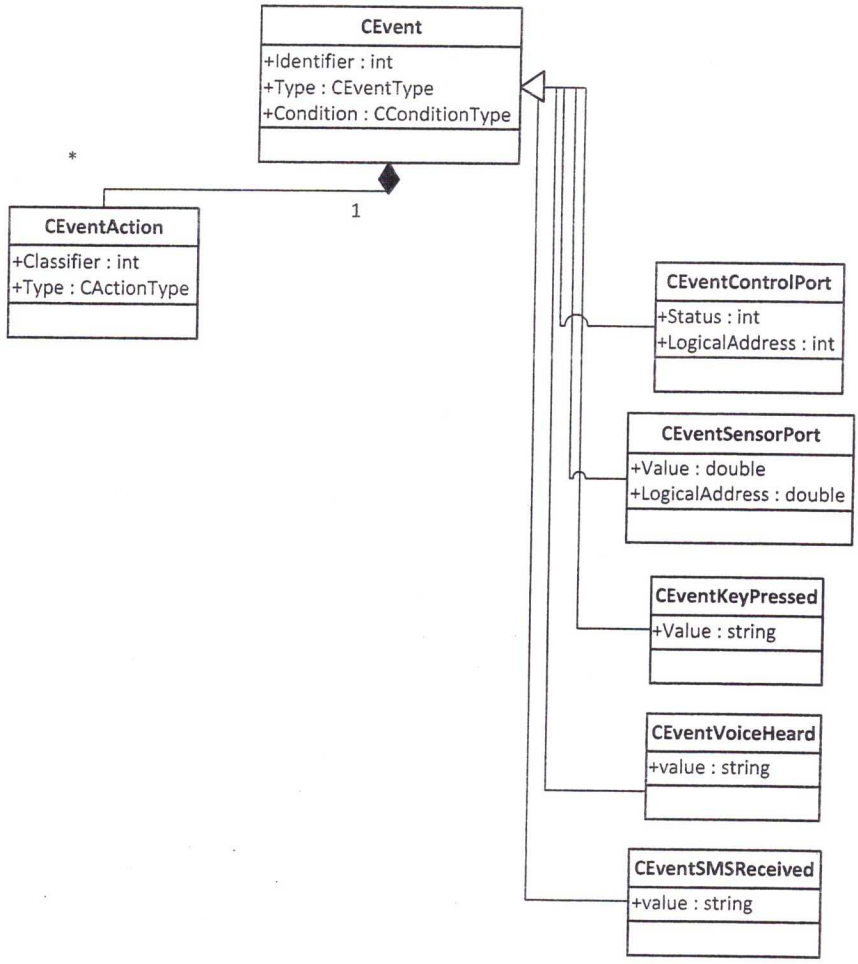


Figure 3.23 Events Architecture UML Class Diagram

3.3.4.5.3.2 Actions

Should an event occur, the system must be able to implement a set of actions; these actions are characterized and grouped by type, so they can be easily implemented by the main software controller.

Tabel 3.7 show the action types and corresponding data:

Table 3.7 Types of Actions

ActionType	Corresponding Data
SetControlPort	ControlPort
Speak	Text String
SendSMS	Text String
ShowMessage	Text String
PlayAlarm	AlarmID

The figure 3.43 show UML Class diagram explains the structure of the design:

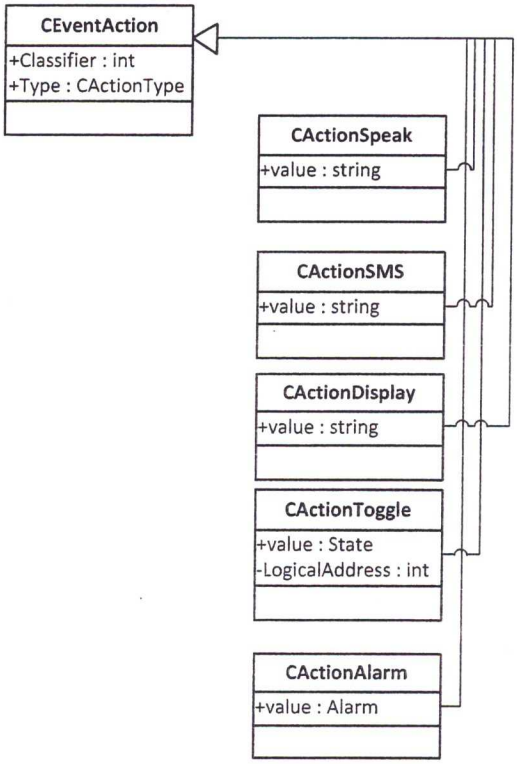


Figure 3.24 Actions UML Class Diagram

3.3.4.5.4 Database Interface

Provides an interface for connecting to and interacting with a database like as pictures ,movies..etc , and encapsulates the process of creating structured SQL statements ; the connection is through the login object and this object As a composite of the data access object .

Figure 3.44 shows the UML diagram of this module:

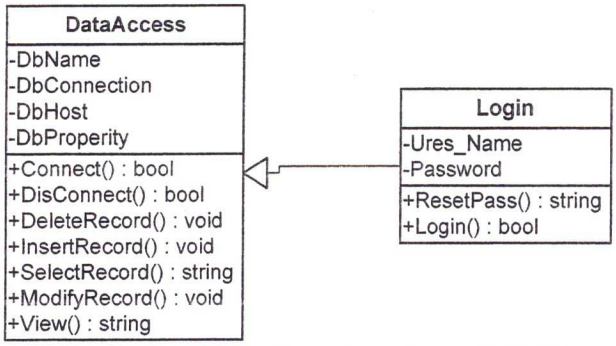


Figure 3.44 DataBase Interface UML Diagram

These are the required functions for this module:

- DbName: Contains the name of the database that we want to connect with it.
- DbConnection: Contains connection code to database.
- DbHost: Contains the address of the server that contain the database system or the server is local host.
- DbProperity: as the root name and other property that use to connect to database.
- Connect, Disconnect: use the code connection to connect or disconnect to database and attending the data.
- Delete,Insert,Select,Modify,View: SQL statement use to retrieve the data or delete or modify or insert to or from database.
- User_Name, password: information the of the user required for access.
- Login: enter to database.

3.3.4.5.5 Database

A database is employed within the software to insure that the environmental data is saved, and to insure that the user can properly retrieve the history of any of the sensory nodes.

It is also used to save the current configurations and operation data. The Database UML Diagram and design is presented in the UML Class diagram in figure 3.47 at the end of chapter 3.

3.3.4.6 Main Software Logic

The main software Logic implements and interfaces all software elements of the system, also, it contains the main algorithm which defines the general behavior of the system.

Figure 3.45 shows the UML Class diagram for some of the required interfacing functions:

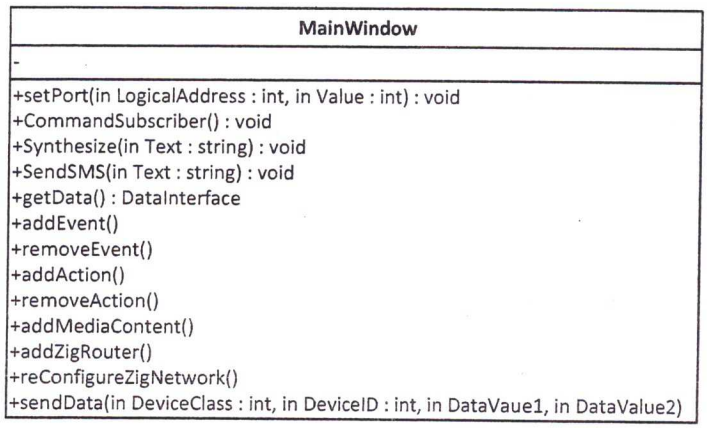


Figure 3.45 DataBase Interface UML Diagram

These functions are interfacing functions for the rest of the components; its internal operation is summarized in activity diagram in figure 3.46.

Figure 3.46 at the end of chapter 3 shows the UML Diagram for the entire software package and exhibits the relationship between its components.

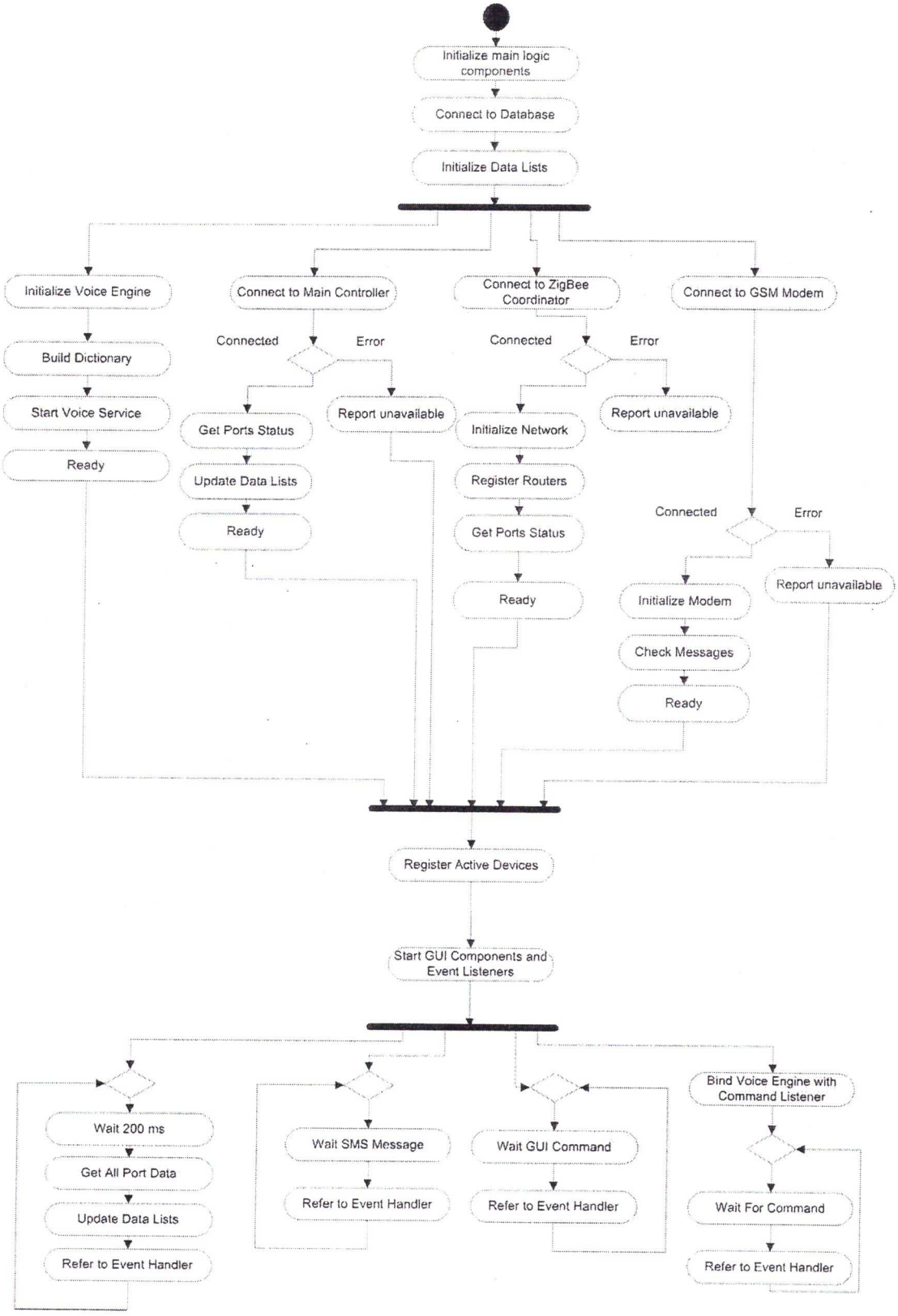
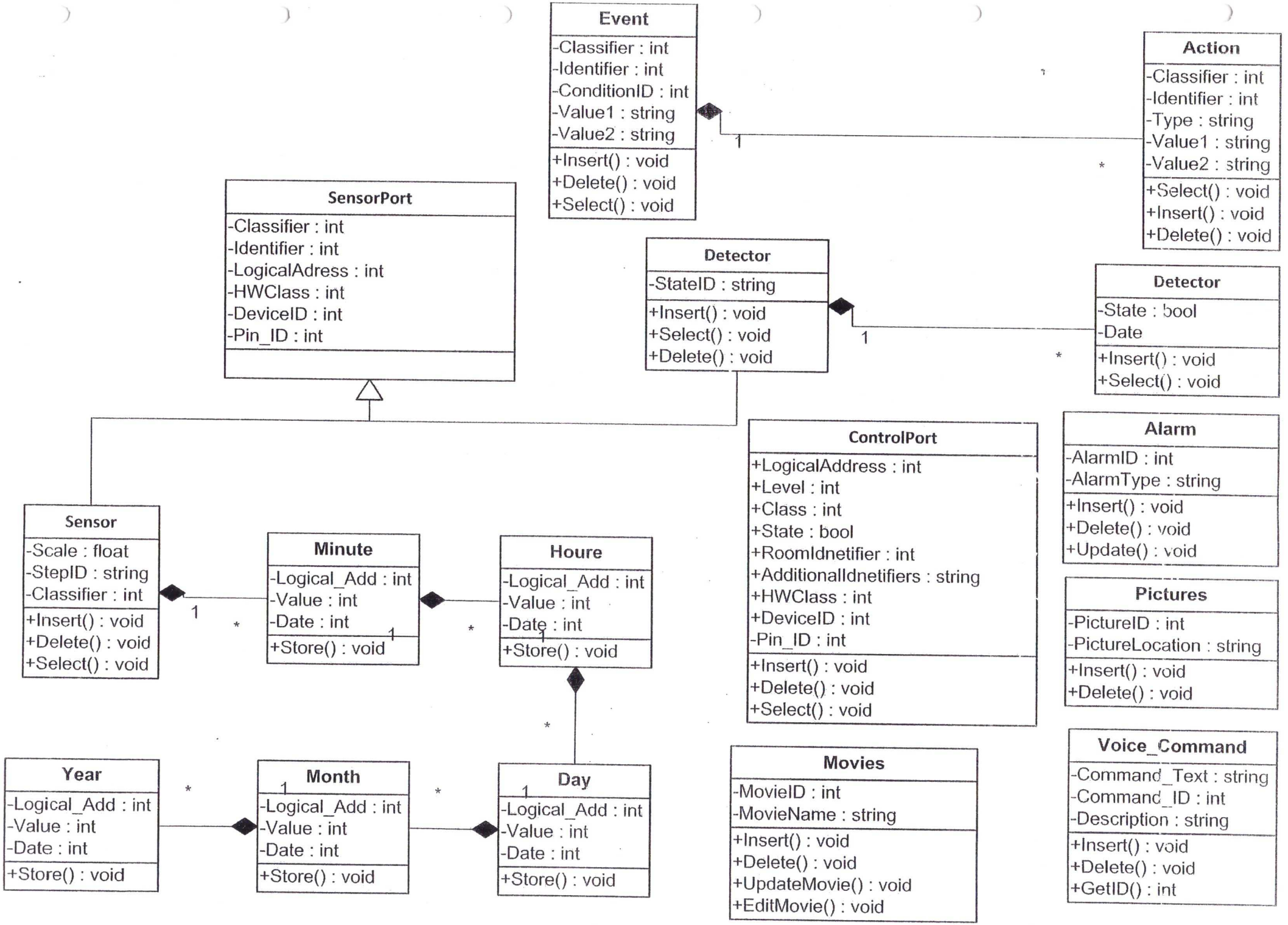


Figure 3.46 Activity Diagram for the Main Logic


Event
-Classifier : int
-Identifier : int
-ConditionID : int
-Value1 : string
-Value2 : string
+Insert() : void
+Delete() : void
+Select() : void
Action
-Classifier : int
-Identifier : int
-Type : string
-Value1 : string
-Value2 : string
+Select() : void
+Insert() : void
+Delete() : void
SensorPort
-Classifier : int
-Identifier : int
-LogicalAdress : int
-HWClass : int
-DeviceID : int
-Pin_ID : int
Detector
-StateID : string
+Insert() : void
+Select() : void
+Delete() : void
Detector
-State : bool
-Date
+Insert() : void
+Select() : void
Sensor
-Scale : float
-StepID : string
-Classifier : int
+Insert() : void
+Delete() : void
+Select() : void
Minute
-Logical_Add : int
-Value : int
-Date : int
+Store() : void
Houre
-Logical_Add : int
-Value : int
-Date : int
+Store() : void
ControlPort
+LogicalAddress : int
+Level : int
+Class : int
+State : bool
+RoomIdnetifier : int
+AdditionalIdnetifiers : string
+HWClass : int
+DeviceID : int
-Pin_ID : int
+Insert() : void
+Delete() : void
+Select() : void
Alarm
-AlarmID : int
-AlarmType : string
+Insert() : void
+Delete() : void
+Update() : void
Pictures
-PictureID : int
-PictureLocation : string
+Insert() : void
+Delete() : void
Year
-Logical_Add : int
-Value : int
-Date : int
+Store() : void
Month
-Logical_Add : int
-Value : int
-Date : int
+Store() : void
Day
-Logical_Add : int
-Value : int
-Date : int
+Store() : void
Movies
-MovieID : int
-MovieName : string
+Insert() : void
+Delete() : void
+UpdateMovie() : void
+EditMovie() : void
Voice_Command
-Command_Text : string
-Command_ID : int
-Description : string
+Insert() : void
+Delete() : void
+GetID() : int
1
*

Chapter Four

System Implementation

4.1 Development Environment Overview

4.2 Hardware Implementation
- 4.2.1 Central Computer
- 4.2.2 Main Controller
- 4.2.3 ZigBee Modules
- 4.2.4 GSM Modem
- 4.2.5 Control Nodes
- 4.2.6 KeyPad
- 4.2.7 System Sensors
- 4.2.8 Remote Control

4.3 Software Implementation
- 4.3.1 Hardware Drivers Module
- 4.3.2 Graphical Modules
- 4.3.3 Voice Modules
- 4.3.4 Data Module
- 4.3.5 Main Logic

4.1 Development Environment Overview

The following is the set of programming tools used to develop the components specified in the design phase as they are presented here, with a small description to illustrate their use and their relevance to this project:

- **Microsoft Visual Studio 2010**

An integrated development environment (IDE) available from Microsoft is used to in the development of C# Code, and the GUI application. As shown in Figure 4.1:

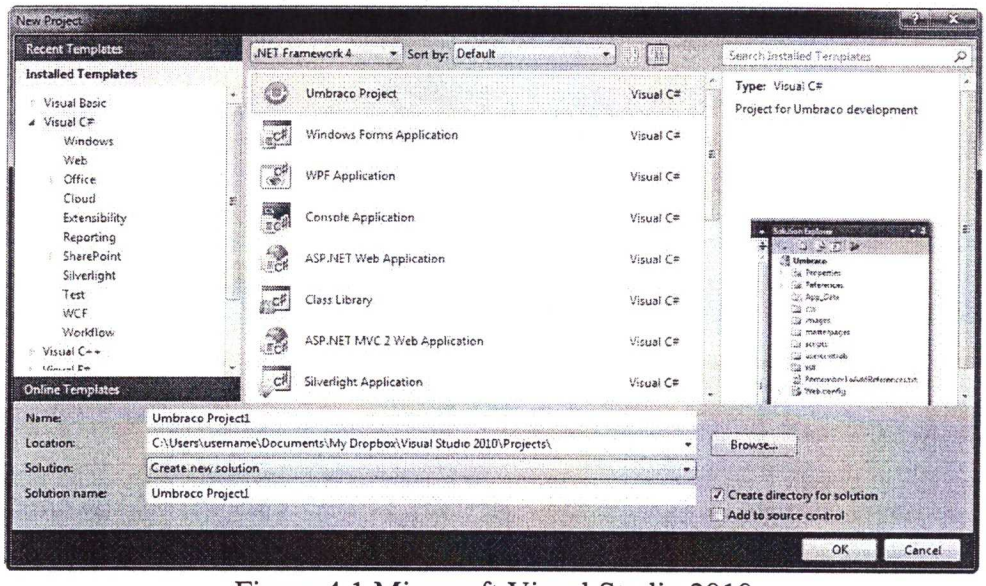


Figure 4.1 Microsoft Visual Studio 2010

The following are its main components:

- **Windows Forms Designer**

  Used to build GUI applications using Windows Forms, where Data-bound controls can be created by dragging items from the Data Sources window onto a design surface .The UI is linked with code using an event-driven programming model.

- **WPF Designer**

  The WPF designer, it supports the drag and drop metaphor. It is used to author user interfaces targeting Windows Presentation Foundation. It supports all WPF functionality including data binding and automatic layout management. It generates XAML code for the UI.

- **Class designer**

  The Class Designer is used to author and edit the classes (including its members and their access) using UML modeling. The Class Designer can generate C# code outlines for the classes and methods. It can also generate class diagrams from hand-written classes.

- **Data designer**

  The data designer can be used to graphically edit database schemas, including typed tables, primary and foreign keys and constraints. It can also be used to design queries from the graphical view.

- **Arduino IDE Software**

Arduino is a popular open-source , designed to make the process of using electronics in multidisciplinary projects more accessible it includes a code editor and is also capable of compiling and uploading programs to the Arduino board with a single click.

There is typically no need to edit make files or run programs on a command-line interface. Although building on command-line is possible if required with some third-party tools, Arduino programs are written in C/C++, although users only need define two functions to make a run able program. As shown in Figure 4.2.

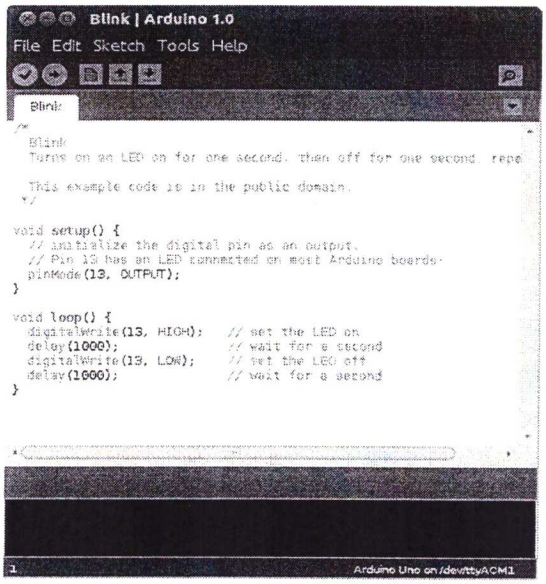


Figure 4.2 Arduino

- **HyperTerminal**

HyperTerminal is a serial communication program used in this application to test and communicate with the serial devices, more specifically the GSM Modem and the ZigBee Coordinator. As shown in Figure 4.3.

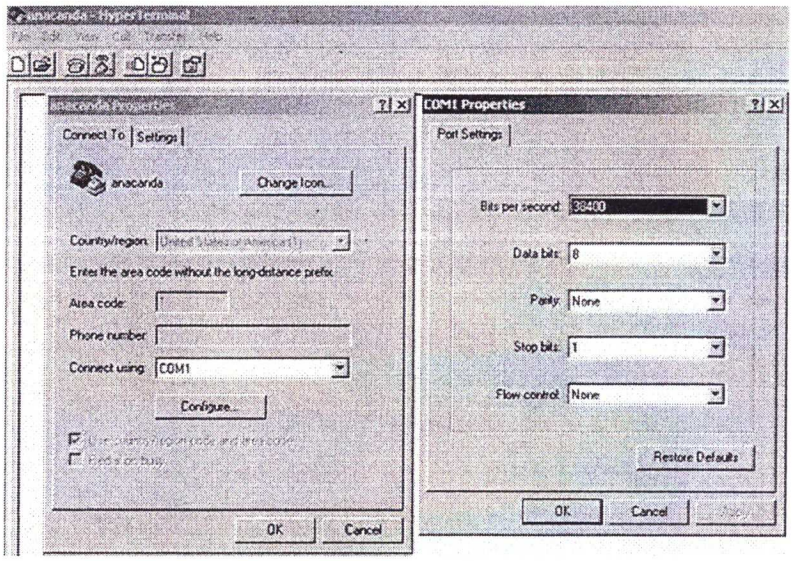


Figure 4.3 Hyper Terminal

- **MPLAB X**

An Integrated Development Environment (IDE), and a toolset for the development of embedded applications on Microchip's PIC and dsPIC microcontrollers includes several free software components for application development, hardware emulation and debugging. As shown in Figure 4.4.

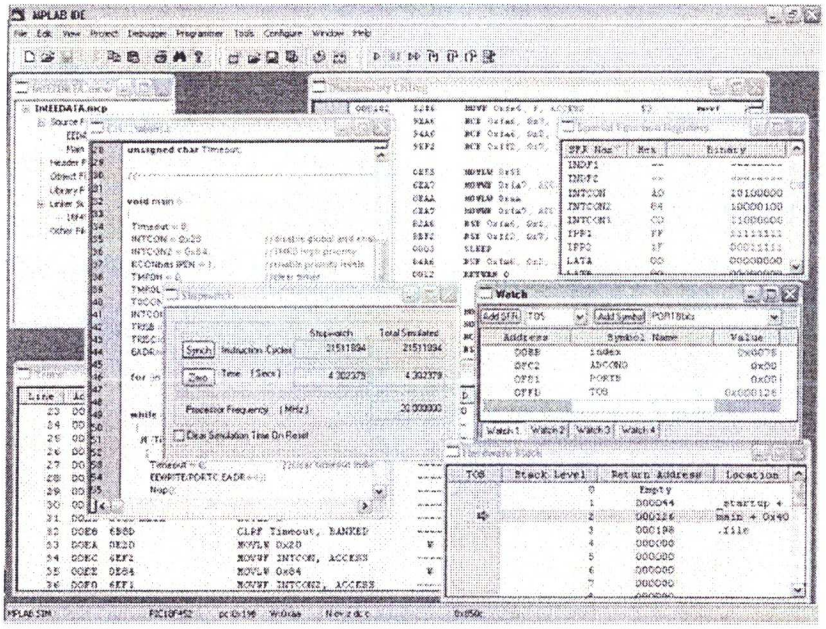


Figure 4.4 MPLAP

- **Eclipse IDE**

Eclipse is a Java Development Integrated Enviroment, and defines the main development environment for android applications, where plugins such as the Google ADT (Android Development Tools), the Android SDK Manager (Software Development Kit Manager), and the Android AVD Manager (Android Development Machine). Figure 4.5 presents a screenshot of the Eclipse IDE:

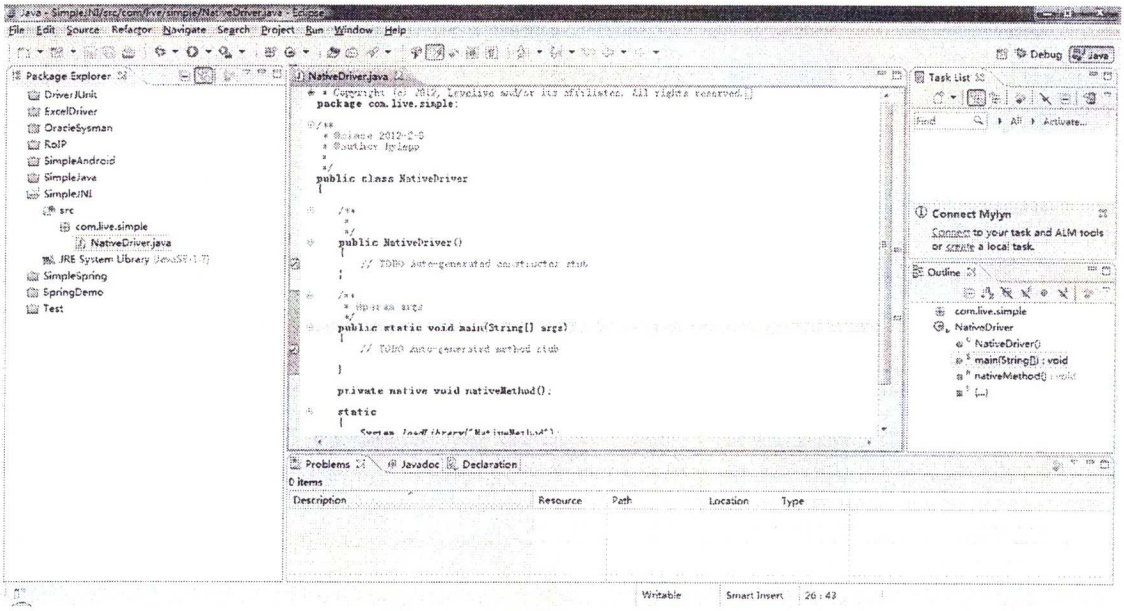


Figure 4.5 Eclipse Classic

- **XCTU**

The program used to configure XBee modules is X-CTU as the official configuration program for XBee radios, and it is available only for the Microsoft Windows operating system. On top of X-CTU we will use other serial terminal programs to change many of the settings and testing but the X-CTU is the only program that can update the firmware of the XBee. Figure 4.6 illustrate the main UI of XCTU

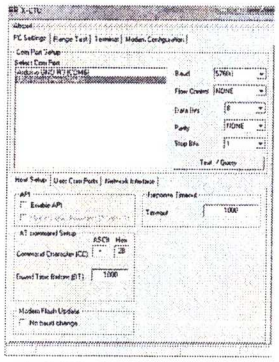


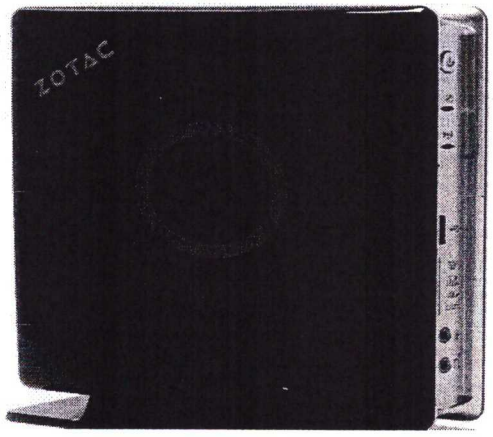
Figure 4.6 X-CTU starting screen

4.2 Hardware Implementation

This section describes the implementation of the hardware components for this project.

4.2.1 Central Computer

Any type of personal computer running a windows system can operate the software of the system; however, since this is considered a real time system it is encouraged and desired to have a computer with the minimal power consumption, and for that, mini computers area a perfect example. At the time of this research, the following computers proved to be available on the market and proved to have a very low power rating of less than 20 watts, they are presented in Figures 4.7 and 4.8:



Intel® Atom™ D525 (dual-core) (1.8 GHz)
Up to 4GB of DDR3 RAM
250GB HDD
4 USB 3.0 Ports
NVIDIA® ION™ graphics processor offering Dual DVI and HDMI video Support

Figure 4.7 Zotac ZBox-ID41-E

AMD E-350 (dual-core) (1.60 GHz)
Up to 8GB of DDR3 RAM
250GB HDD
3 USB 3.0 Ports
AMD Radeon HD 6310 video card, offering Dual DVI and HDMI video Support

Figure 4.8 Zotac ZBox-AD03BR

4.2.2 Wired Controller

The wired controller is implemented with direct regards to the design specification of the project where a set of PPI and other "extensions" are connected to a USB enabled PIC18f4550.The function of the wired controller is to divert a computer command into a digital On/Off command, and retrieve analog and digital signals for the computer.

With this implementation, and this version of the controller, it should provide 72 ports for digital control, 24 ports for digital input, and 16 ports for analog input. It could be argued that using the same implementation, the number of ports could be increased or decreased.

The following are the output and input pin configurations for the wired microcontroller design:

Table 4.1 Wired Controller Pin IDs

Chip	Pin	Description
MC14067BCP	X_0-X_{15}	16 Channel Analog Input
NEC8255A -1	$PA_0 - PA_7$	Output 1 to 8
NEC8255A -1	$PB_0 - PB_7$	Output 9 to 15
NEC8255A -1	$PC_0 - PC_7$	Output 17 to 24
NEC8255A -2	$PA_0 - PA_7$	Output 25 to 32
NEC8255A -2	$PB_0 - PB_7$	Output 33 to 40
NEC8255A -2	$PC_0 - PC_7$	Output 41 to 48
NEC8255A -3	$PA_0 - PA_7$	Output 49 to 56
NEC8255A -3	$PB_0 - PB_7$	Output 57 to 64
NEC8255A -3	$PC_0 - PC_7$	Output 65 to 72
NEC8255A -4	$PA_0 - PA_7$	Digital Input 1 to 8
NEC8255A -4	$PB_0 - PB_7$	Digital Input 9 to 15
NEC8255A -4	$PC_0 - PC_7$	Digital Input 17 to 24

The following schematics and PCB Blueprints illustrate the implementation of the wired controller:

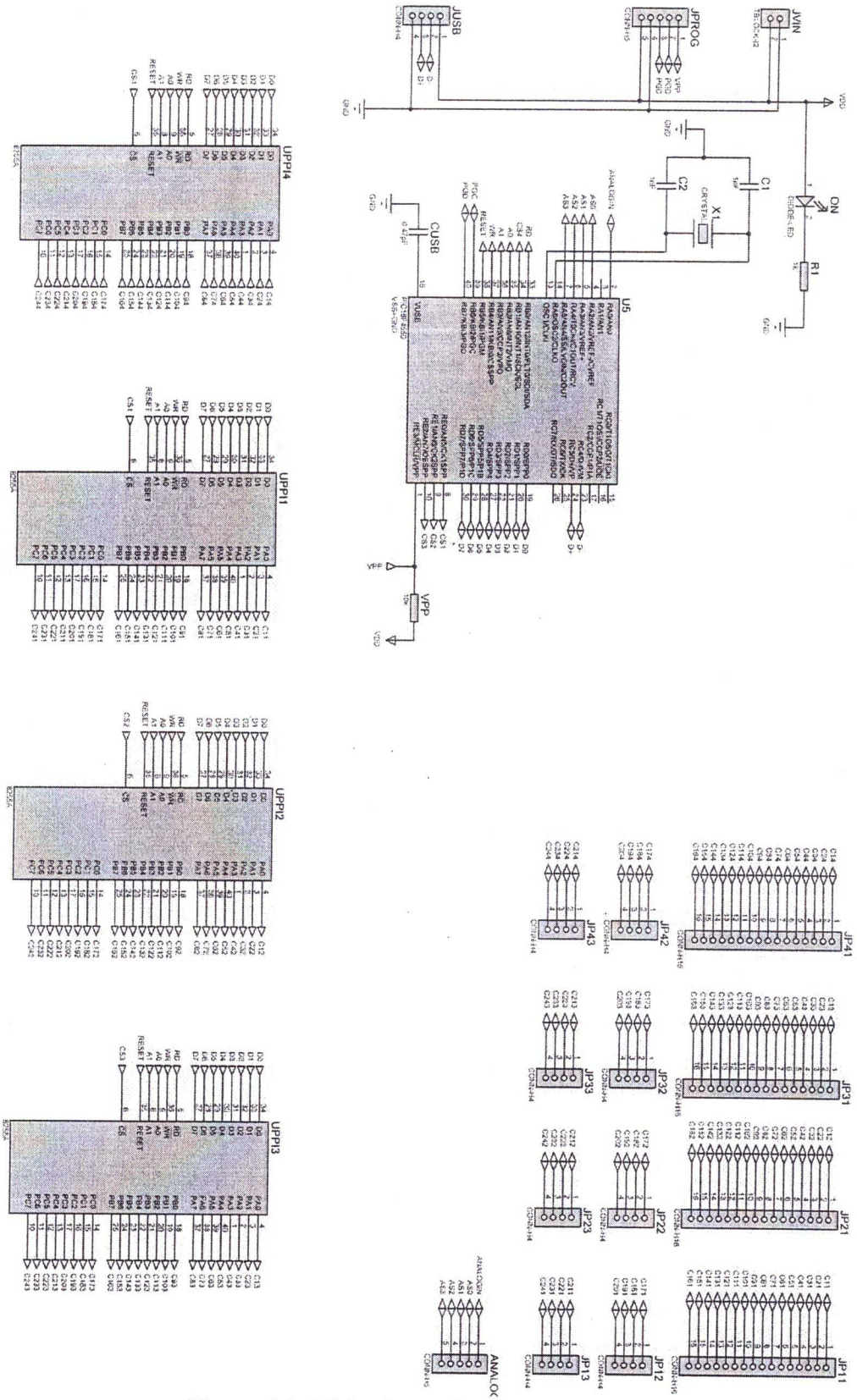


Figure 4.9 Main Controller Schematics

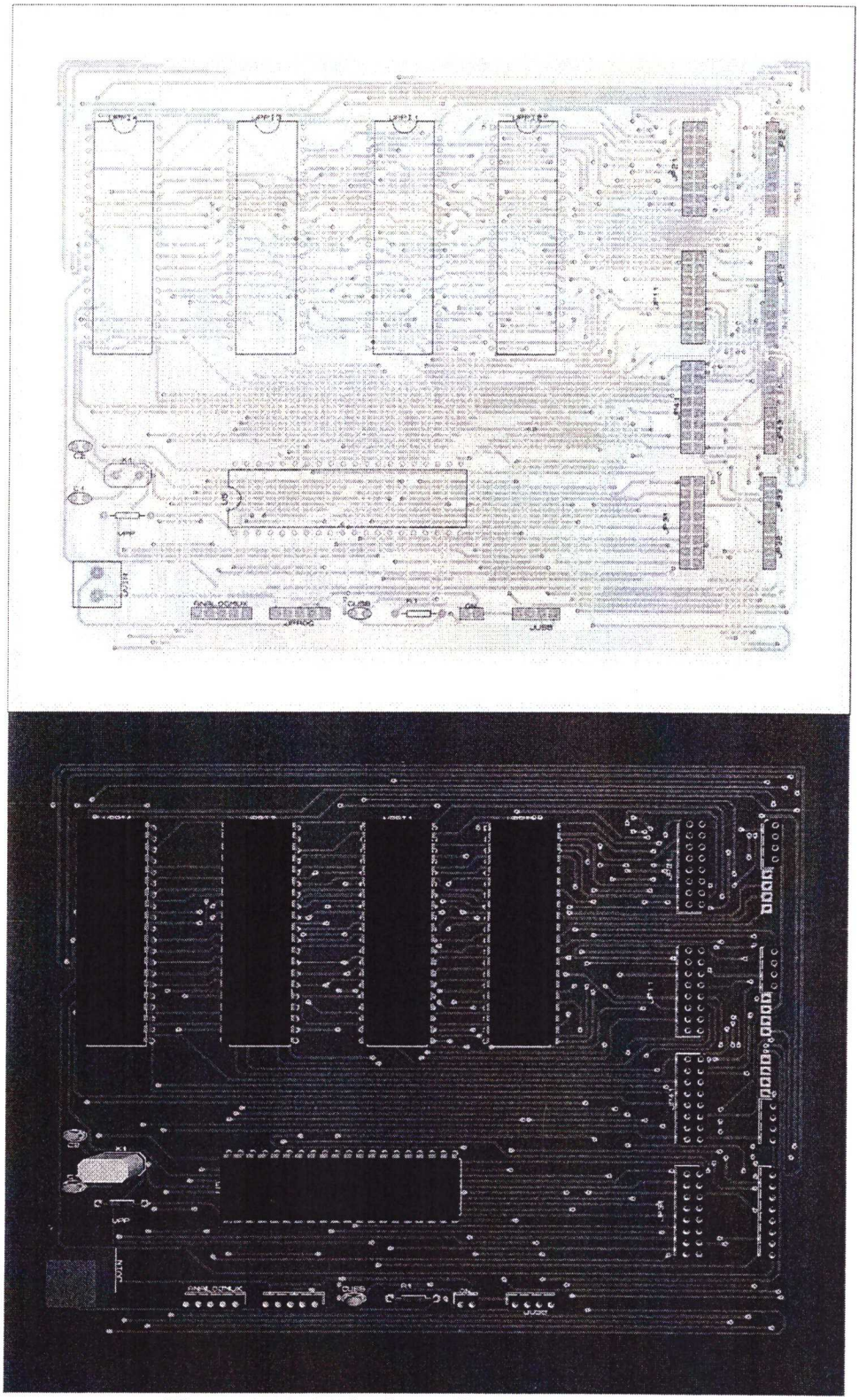


Figure 4.10 Main Controller PCB Blueprints

4.2.2.1 Analog Ports

16 Channels Mux/Demux is used to take in analog signals from various types of sensors, and they are interface with pins from port A within the PIC18F4550, one line for the analog input and 4 lines represent selection lines.

Table 4.2 PIC18F4550 Selection lines

PIC18F4550	Description
AN0	Analog Input
RA1	Selection Line 0
RA2	Selection Line 1
RA3	Selection Line 2
RA4	Selection Line 3

The 5 Lines are taken as an input for an analog board that provides input for 16 channels of analog sensors.

The analog board is consisted of an MC14067BCP and the corresponding connections required interfacing the various analog sensors. Pin X represents the common output analog pin, A,B,C,D represent the selection lines, INHIBIT is put to 0, and X0 To X15 represent the analog inputs. The following is the schematics and PCB blueprints for the analog board:

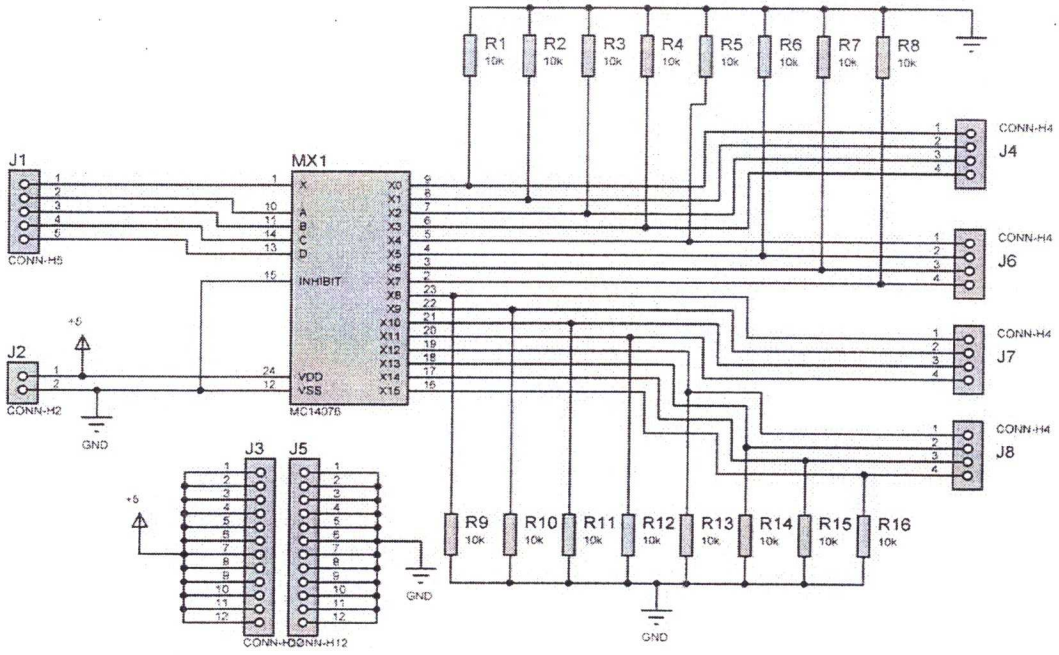


Figure 4.11 Analog Board Schematics

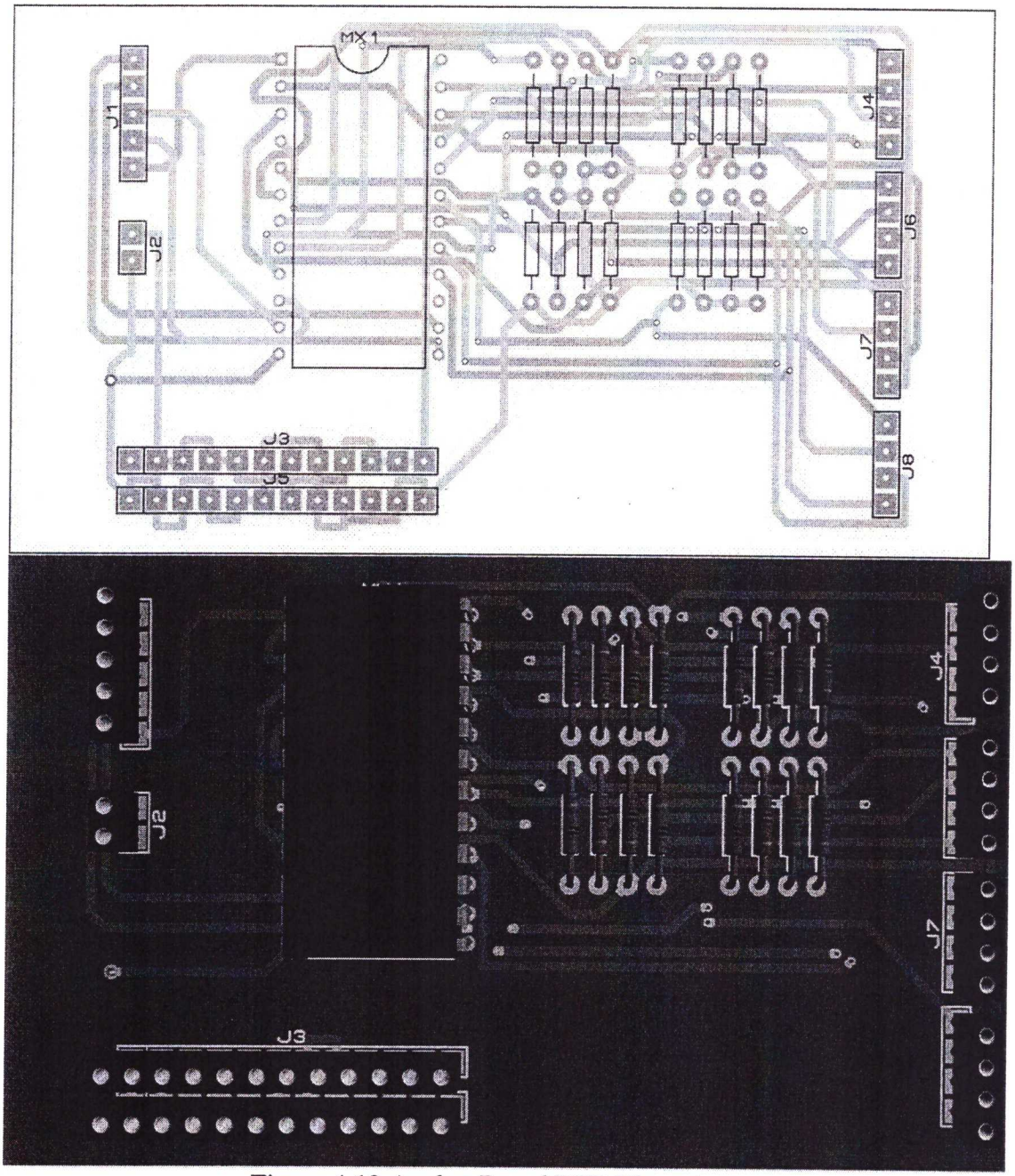


Figure 4.12 Analog Board PCB Blueprints

4.2.2.2 Digital Ports

Four 8255a chips are used to extend the PIC18F4550 ports, each pin is specified an address for access control.

To control a PPI chip, 8 Lines must be allocated as a data bus, 5 Lines for access control, and 1 line as a chip select, and so, the following pins of the PIC18f4550 were used for implementation:

Table 4.3 PPI Control Lines

PIC18F4550	8255a	Description
RD0 – RD7	D0 – D7	Data bus
RB0	RD	Read signal
RB4	WR	Write signal
RB2	A0	Port selection lines
RB3	A1	
RB5	RESET	Reset configurations

The following Pins were allocated as selection lines for 4 8255a chips:

Table 4.4 8255 Selection Lines

PIC18F4550	Line	Description
RE0	CS (PPI1)	Select PPI output 1
RE1	CS (PPI2)	Select PPI output 2
RE2	CS (PPI3)	Select PPI output 3
RB1	CS (PPI4)	Select PPI input

Each set of ports is interfaced to an either digital output board, or a digital input board.

4.2.2.3 Output Board

The digital output board contains chips that convert voltage levels, from the 5V logic to 12V or 24V so it can control relays and other electrical devices. The ULN2803A transistor array chip is used for that purpose, where it accepts a 5V input and then outputs a secondary voltage from a secondary line.

The following is the schematics for the output board:

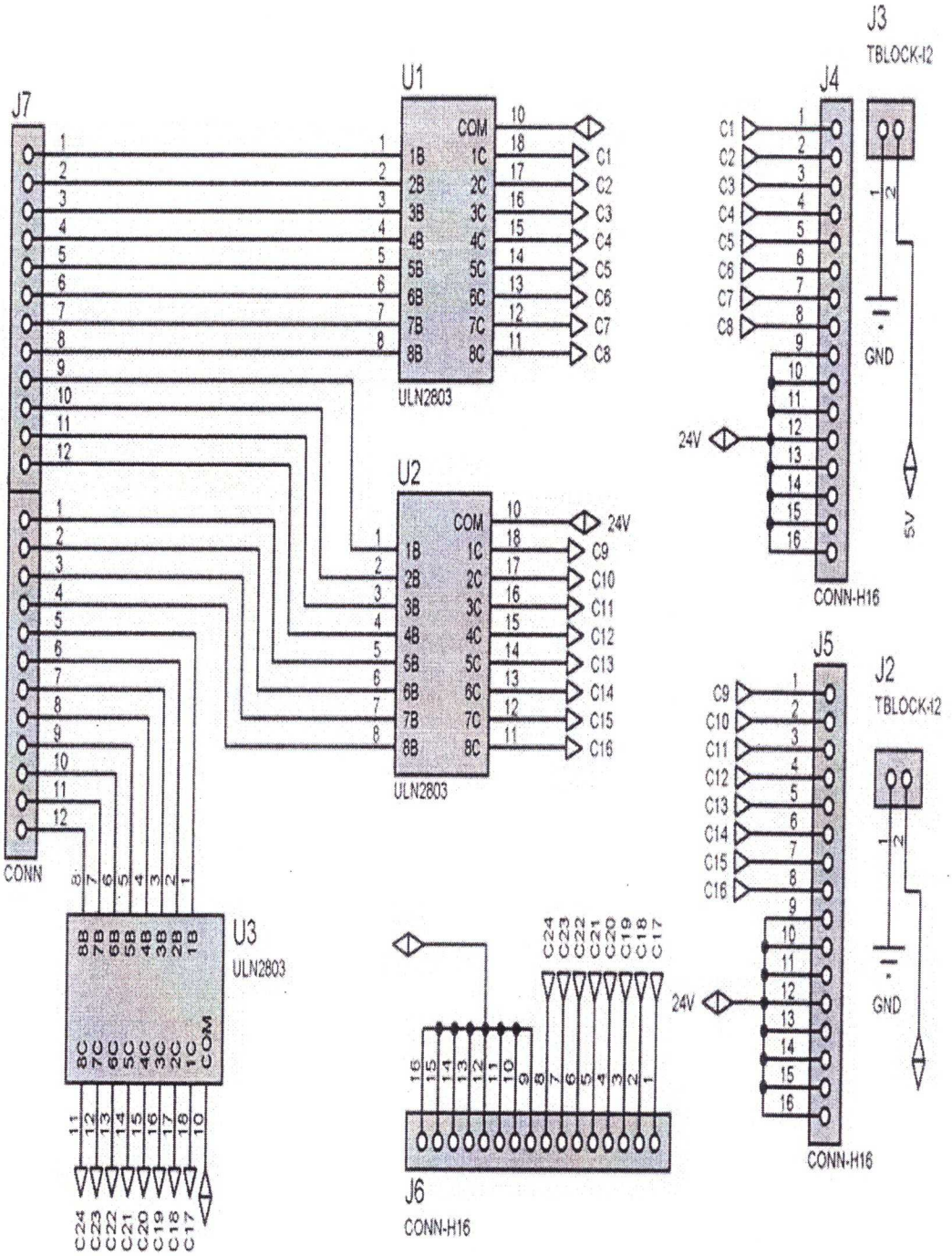


Figure 4.13 Output Board Schematics

Figures 4.14 ,4.14 are the PCB Blueprints for the output board :

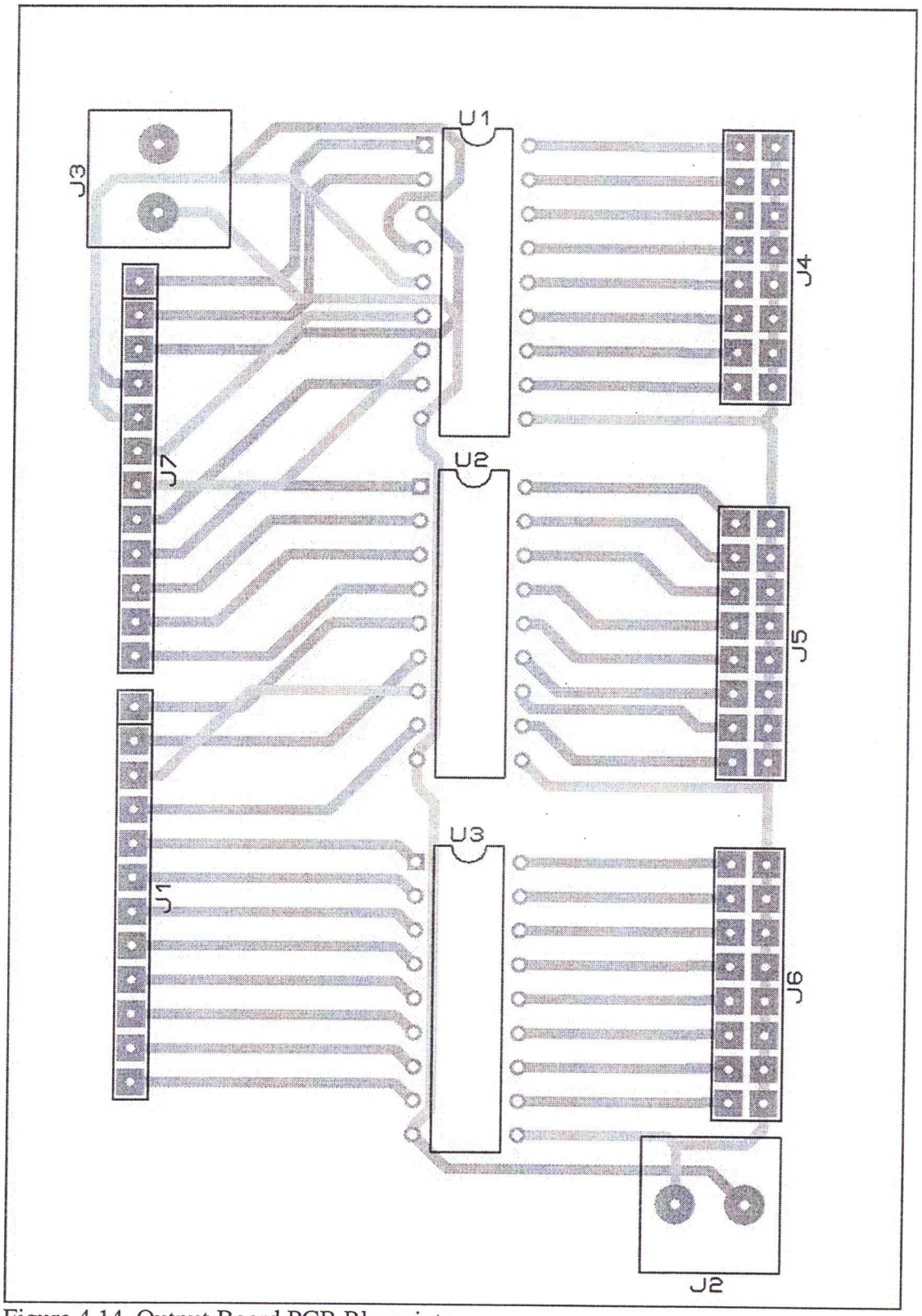


Figure 4.14 Output Board PCB Blueprints

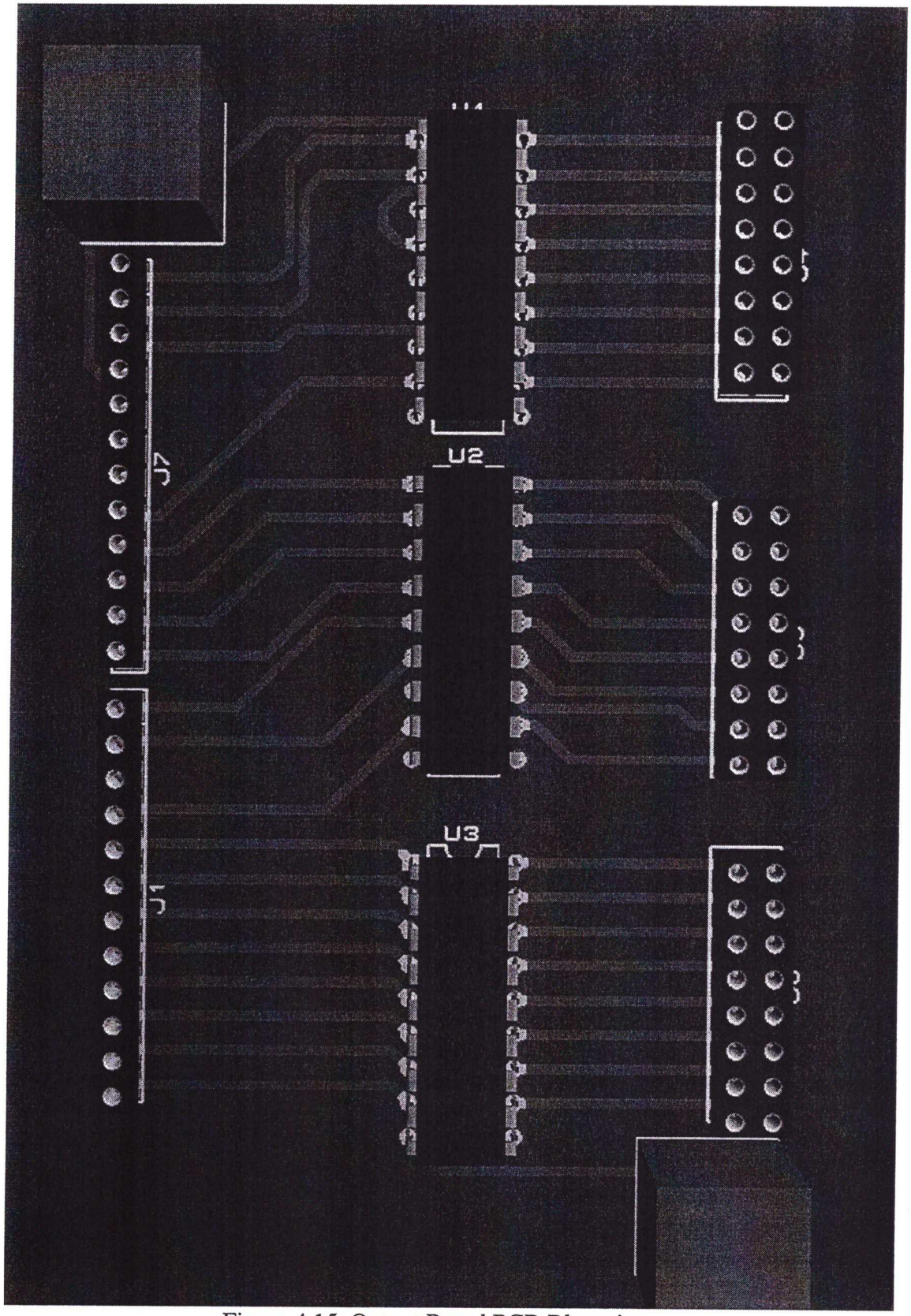


Figure 4.15 Output Board PCB Blueprints

4.2.2.4 Input Board

The digital input board provides input for one of the PPIs, the sole purpose of this board is to introduce a 0V input in the absence of an incoming signal. It is used mainly to interface digital sensors such as IR sensors and motion sensors.

The following are the schematics and PCB Blueprints for the Input board :

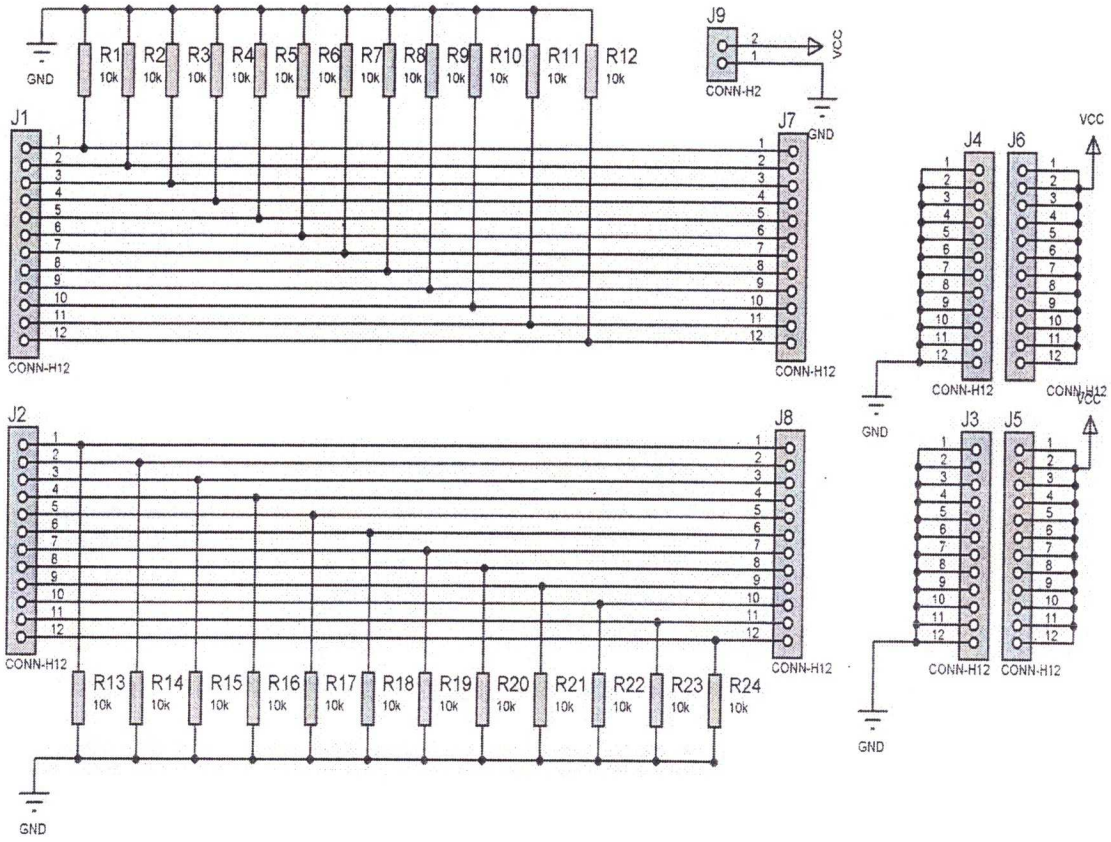


Figure 4.16 Input Board Schematics

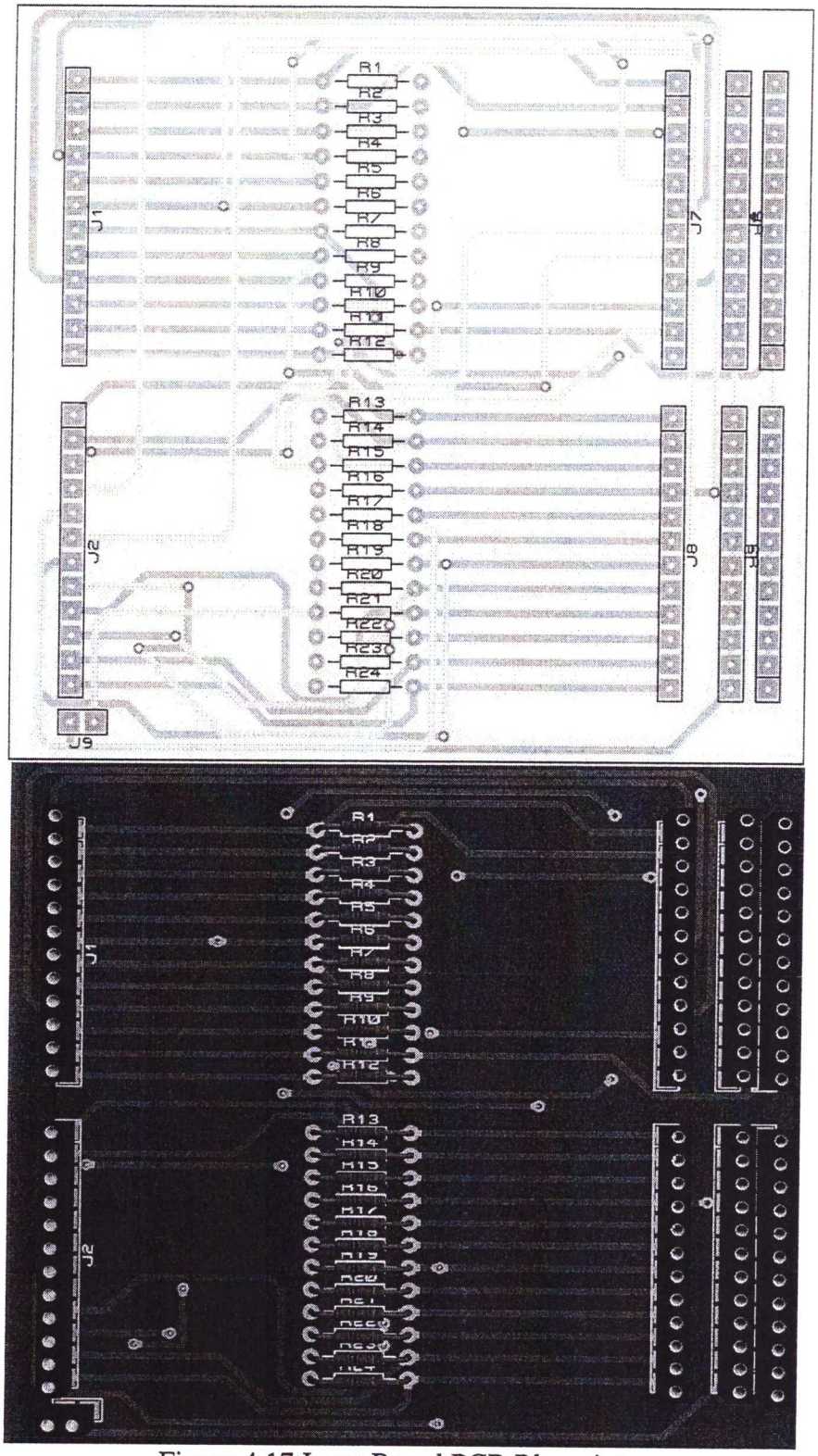


Figure 4.17 Input Board PCB Blueprints

4.2.2.5 Main Controller Software

The following software code is required to operate the PIC in the main microcontroller; it is compiled in MPLab and then uploaded to the chip. These software snippets illustrate the main functions required for it to run, the other software statements and functions are stated in the code appendix (Appendix E) of this document.

4.2.2.5.1 Peripheral Chips Functions

Three function have been made to control the 8255a chips :

- Reset function : it resets the four chips, the sets the control word according to the argument provided:

  The following is the function declaration, where it supplied 4 arguments specifying the control words for the four 8225a

```
void ResetPPIs(char CW1,char CW2,char CW3,char CW4);
```

- Get functions : these function are responsible for reading a specific port of one of the 8255a chips, so there are 3 functions, one for each port taking an argument specifying the corresponding chip number:

```
int getA(char PPI);
int getB(char PPI);
int getC(char PPI);
```

- Set functions : these function are responsible for setting a specific port of one of the 8255a chips, so there are 3 functions, one for each port taking an argument specifying the corresponding chip number, and the port's byte:

```
void setA(charPPI,char a);
void setB(charPPI,char a);
void setC(charPPI,char a);
```

- SetPin functions: this function map a specific pin address to the specified state, so it provides accessibility to the pins rather than the 8 bit ports, the following is the function declaration:

```
void setPin(unsignedchar Address, char state);
```

The body for these function, and their internal implementation is included and well commented in the appendix

4.2.2.5.2 Analog Read Functions

To read the 16 analog channels provided in the analog input board, the following justify functions are used:

- ReadAnalog1function : it is responsible for fetching the analog value of the AN0 and convert it into an 10bit digital value stored in a union structure containing two bytes named word_val, the following is the function declaration:

```
WORD_VAL ReadAnalog1(void);
```

- getADCfunction : it sets the corresponding pins value to the argument specified in the function call, then calls the ReadAnalog1() function, and thus effectively reading a specific numbered analog port, the following is the function declaration:

```
void getADC(char ANX);
```

4.2.2.5.3 Placement functions

These functions are responsible for filling the 64 bytes array used to transfer data to the computer, it calls the function defined above in the defined arrangement specified in the design chapter:

- ConvertAllANX function : which retrieves the status of all 16 analog channels, and then map it to the array starting with index 1 and ending with index 32, the following is the function declaration:

```
Void ConvertAllANX();
```

- ReportPPIStatus function : which retrieves the current value of all the 8255a ports, it is used to verify the correct status of each output port, and get the value of each input port. The following is the function declaration:

```
Void ReportPPIStatus()
```

4.2.2.5.4 USB Communication functions

There are a lot of functions dedicated for the Plug and Play USB communication and they are provided in the source code from Microchip Corporation. They are categorized into device specific functions which should stay unchanged and unaltered, and user specific functions where one may alter and insert one's own code and system implementation.

Device specific functions contain instructions to handle interrupts, and to set the internal control registers for USB Handling, also they manage all the USB communication details which involve tasks such as registering the device, reading the bus and waiting for it to be in an available state, plus sending and receiving data.

The following are the functions most important to this project propose, and they are specified as user functions:

- InitializeSystem function : which sets all the required ports to the desired state, some are output and some are input, the following is the function declaration:

```
static void InitializeSystem();
```

- ProcessIO function : which is called within a loop in the main function, where it handles the USB application tasks more specifically here as reading data and setting digital ports to a specific state, the following is a code segment representing part of the function:

```
voidProcessIO(void)
  {
     //Blink the LEDs according to the USB device status
     if(blinkStatusValid)
       {
         BlinkUSBStatus();
       }
     // User Application USB tasks
     if((USBDeviceState< CONFIGURED_STATE)||(USBSuspendControl==1))
          return;
     if(!HIDRxHandleBusy(USBOutHandle))
       {   switch(ReceivedDataBuffer[0])   //Look at the data the host sent
             {
             case 0x37://0x37 represents the Command Identifier responsible for
                          // getting the analog values
                    ToSendDataBuffer[0] = 0x37;  //Echo back to the host 0x37
                    PORTAbits.RA1 = !PORTAbits.RA1;
                    ConvertAllANX();
```

```
if(!HIDTxHandleBusy(USBInHandle))
  {
      USBInHandle = HIDTxPacket(HID_EP,(BYTE*)&ToSendDataBuffer[0],64);
  }
```

The code checks the USB Device state, and ends the function if it is in an unconnected state, then it reads the incoming array of bytes by calling a function that returns a flag specifying whether data has been received or not, if data is read, it returns a non-zero argument and goes into a switch statement on the first byte of the array.

In the switch statement, one of the cases is illustrated, which is an inquiry of the analog values coded with the number 0x37, the function lists the incoming command for echo back and validation, and then calls the ConvertAllANX function which effectively fills an array with the read analog values .Then it prepares a packet for the USB Firmware functions containing the array of bytes so it can be sent.

The full details of the code are listed in the code appendix (Appendix E) .

4.2.3 Wireless Controller

4.2.3.1 XBee Wireless Technology

The technology selected to achieve the wireless networking required for this project is the ZigBee Wireless standard using Digi® XBee® ZB modules. The project design essentially uses a single coordinator, and two router nodes. XBees were designed as a standard for low cost, low power mesh networks components with the main application areas being wireless sensor networks.

The XBee modules, shown in Figure 4.18, were chosen as they seemed to be the suitable choice to implement the ZigBee network as needed.

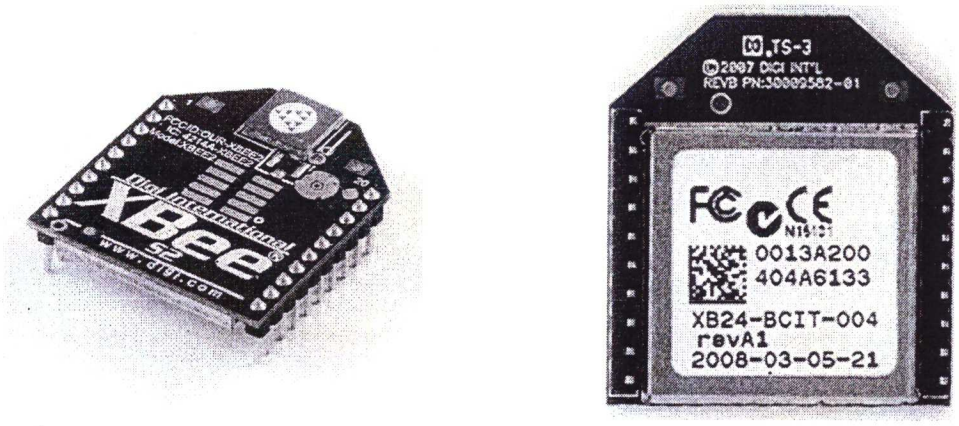


Figure 4.18 This module is the 2mW Chip Antenna Series 2 version

The XBee modules contain a microcontroller loaded with firmware that implements all ZigBee networking/routing features and the RF circuitry and antenna.

Figure 4.19 below shows the XBee's pins numbering and their names, XBee's datasheet illustrates the specification of every pin:

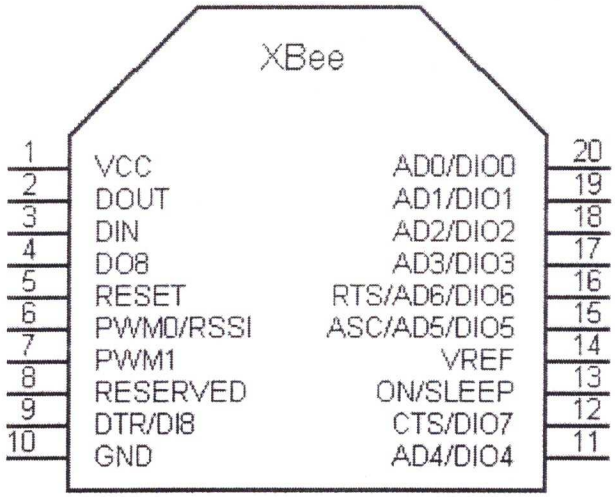


Figure: XBee physical pin numbering

Figure 4.19 XBee physical pin numbering

Only four pins for the XBee modules will be used; with the following table illustrating the pins details.

Table 4.5 Pin Assignments for the XBee Modules

Pin #	Name	Direction	Description
1	VCC	-	Power supply, 3.3 V power supply
2	DOUT	Output	UART Data Out (TX)
3	DIN	Input	UART Data In (RX)
10	GND	-	Ground

The DOUT and DIN will be used to transmission and reception and the VCC and GND for powering the XBee.

In addition we will use another three pins in case of updating the firmware; the following table illustrates the pins and their details. As Shown in table 4.6.

Table 4.6 : Pin Assignments for the XBee Modules

Pin #	Name	Direction	Description
9	$\overline{\text{DTR}}$	Either	Control Line, Data Terminal Ready
12	$\overline{\text{CTS}}$	Either	Clear-to-Send Flow Control
16	$\overline{\text{RTS}}$	Either	Request-to-Send Flow Control

To configure the XBee radios we used the adapter or the explorer; it is a USB to serial base unit for the XBee line.

Figure 4.20 shows the actual Explorer module, which is represented as the seat of the XBee module on which the female pins of the explorer which are matched with the XBee's pins as shown.

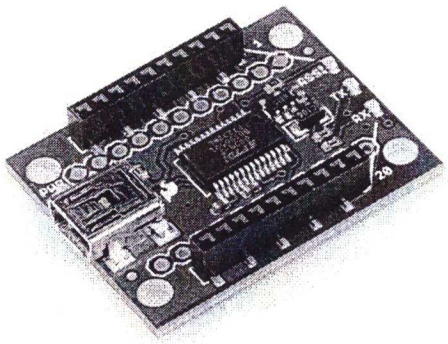


Figure 4.20 XBee Explorer board

Figure 4.21 illustrates the schematic of explorer USB interface, which illustrate the connection of the D-, D+, VBUS and GND of USB with the explorer circuit to GND, VCC, DIN, DOUT, CTS, RTS and DTR.

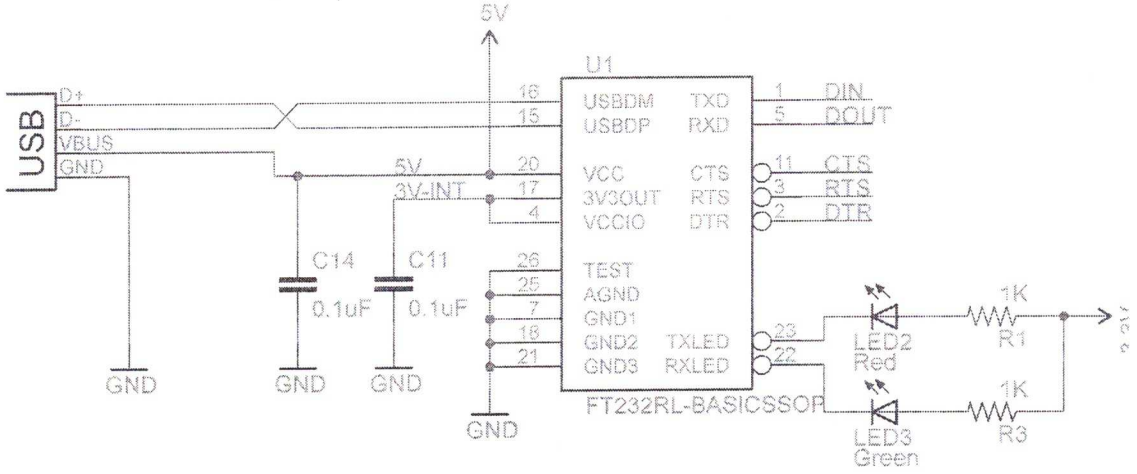


Figure 4.21 Explorer schematic, USB interface

The XBee radio has 2mm spacing between their pins to make it compact size. The breadboard has 2.54mm spacing, we use the XBee breakout board to fit the XBee into breadboard; the below figures illustrate the XBee breakout board and its pins.

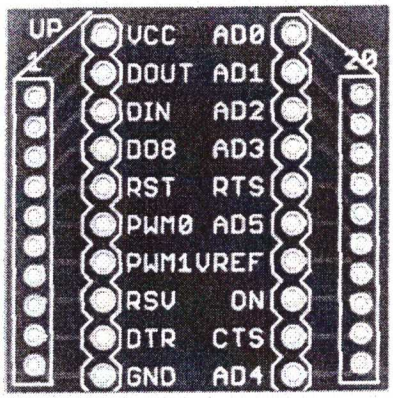


Figure 4.22 XBee breakoutboard

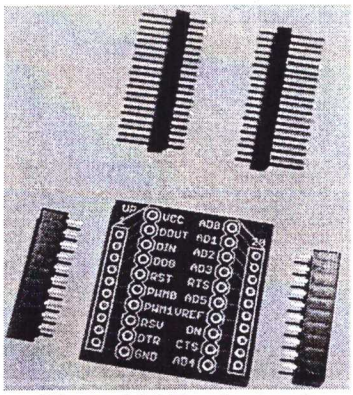


Figure 4.23 Breakout and connection pins

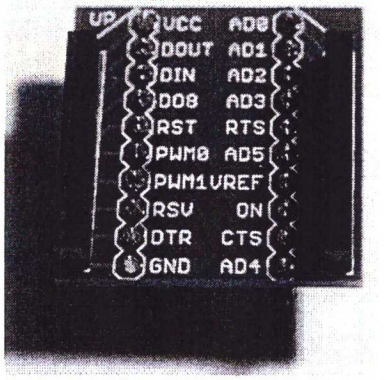


Figure 4.24 Soldered breakoutboard

To configure the coordinator and routers we need to use the X-CTU but before that we need to get the XBee radio address, every XBee radio has a 64-bit serial number address printed on the back, figure 4.25 shows the XBee address. The high part of the address will be 0013A200, Digi's pre-assigned range of address space. The low part of the address will be different and unique for every radio.

Also an ID is needed for a personal area network PAN; each ZigBee network has a PAN address, another 16-bit address.

For every ZigBee network it must have only one coordinator radio to be properly defined and managed.

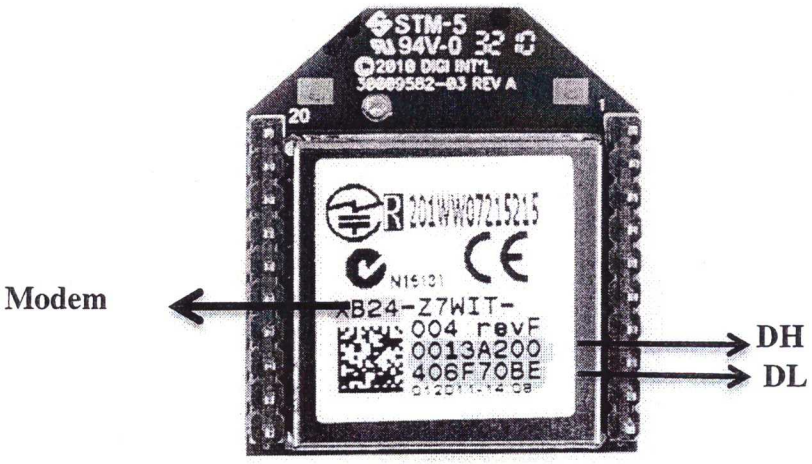


Figure 4.25 Back of XBee showing 64-bit address and Modem type

4.2.3.2 XBee Modes of Communication

The following figure 4.26 illustrates the mesh network topology, in which this application is built with; it is consisted of one coordinator and multiple routers:

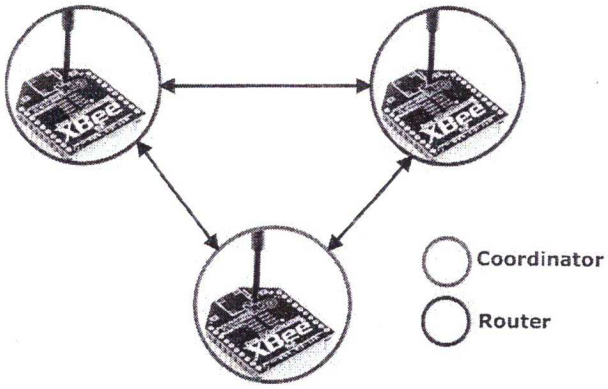


Figure 4.26 Mesh Network Topology

XBee radio has different modes for communications and configurations which are AT, API, transparent and command modes; which are described as the following:

- Transparent or AT mode: the default state for XBee radios using AT firmware, where the radio simply passes information along exactly as it receives it as identified by the DH and DL address.
- Command mode: allows talks directly to the local radio for configuration and its issues, where in command mode it is talked to the XBee itself.
- API or Packet Mode, it requires that communication with the module be done through a structured interface (data is communicated in frames in a defined order). It has many features such as I/O Samples, ACK and Retries, receive packets contain the source address of transmitting radio which is the most important one in this project and other features such as obtaining RSSI (signal strength) of an RX packet.

The API Mode will be chose as the best way to interface the nodes as it allows single and direct communication for each router, and can control it state.

The following table illustrates the API frame structure which includes:

- **Start Delimiter**: which is a unique number that indicates we are at the beginning of the data frame.

- **Length**: Number of bytes between the length and the checksum.
- **Frame Data**: The frame data is specific to each type of message we receive from the XBee radio. Table 4.7 illustrates the structure of the frame data which is consists of identifier and data. The API-identifier indicates which API messages will be contained in the Identifier-specific data. The XBee modules support many API frames and we will use only two frames, table 4.7 illustrate the two frames details.
- **Checksum**: The last byte of the frame is always a checksum, it is used at the receiving end to check and see if there was a transmission error.

To calculate the checksum: Not including frame delimiters and length, we add all bytes keeping only the lowest 8 bits of the result and subtract the result from 0xFF. And To verify it we add all bytes (include checksum, but not the delimiter and length). If the checksum is correct, the sum will equal 0xFF.

Start Delimiter (Byte 1)	Length (Byte 2)	(Byte 3)	Frame Data (Byte 4 -to- Byte *n*)	Checksum (Byte n+1)
0x7E	MSB	LSB	API-specific	1 Byte

Figure 4.27 Basic API frame structure

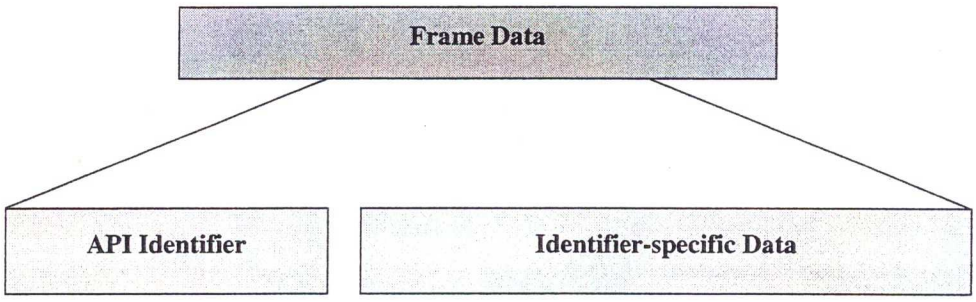


Figure 4.28 Frame Data

Table 4.7 shows an API packet example with detailed, it illustrates the structure of the packet that we will be sent and received.

Table 4.7 API PACKET

	Frame Fields		Offset	Hex	Description	
API Packet	Start Delimiter		0	0x7E		
	Length		MSB 1	0x00	Number of bytes between the length and the	
			LSB 2	0x00		
	Frame-specific Data	Frame Type	3	0x90 or 0x10	ZigBee Transmit Request	0x10
					ZigBee Receive Packet	0x90
		Frame ID	4	0x00	Identifies the UART data frame for the host to correlate with a subsequent ACK	
		64-bit Source Address	MSB 5	0x13	64-bit address of sender	
			6	0xA2		
			7	0x00		
			8	0x00		
			9	0x00		
			10	0x00		
			LSB 11	0x00		
		16-bit Source Network Address	MSB 12	0x00	16-bit address of sender	
			LSB 13	0x00		
		Receive Options	14	0x00	0x01 - Packet Acknowledged 0x02 - Packet was	
		Received Data	15	0x00	Received RF data	
			16	0x00		
			17	0x00		
			18	0x00		
			19	0x00		
			20	0x00		
	Checksum		21	0x00	0xFF - the 8 bit sum of bytes from offset 3 to this	

4.2.4 Wireless Controller Modules

4.2.4.1 Wireless Coordinator

Every XBee networks have a single coordinator device. It is responsible for forming the network, handing out addresses, and managing the other functions that define the network, secure it, and keep it healthy.

The coordinator is connected to the pc via USB port, and it opens a connection to control the various system routers, as illustrated in figure 4.29.

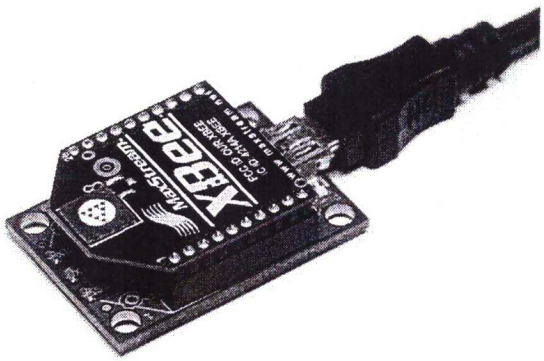


Figure 4.29 The XBee aligned and seated in its adapter

It is responsible for forming the network, handing out addresses, and managing the other functions that define the network, secure it, and keep it healthy. XBee radio into explorer and connect it to the computer USB port. We open the X-CTU application to start the configuration; the figure below shows the X-CTU starting screen which has a lot of options and details, in the right side there is the serial terminal options such as the baud rate, also the available com ports connected to the computer.

Figure 4.30 shows the X-CTU application screenshot, where the connection configuration is highlighted in yellow in the screenshot to the left, and the modem configuration to the right.

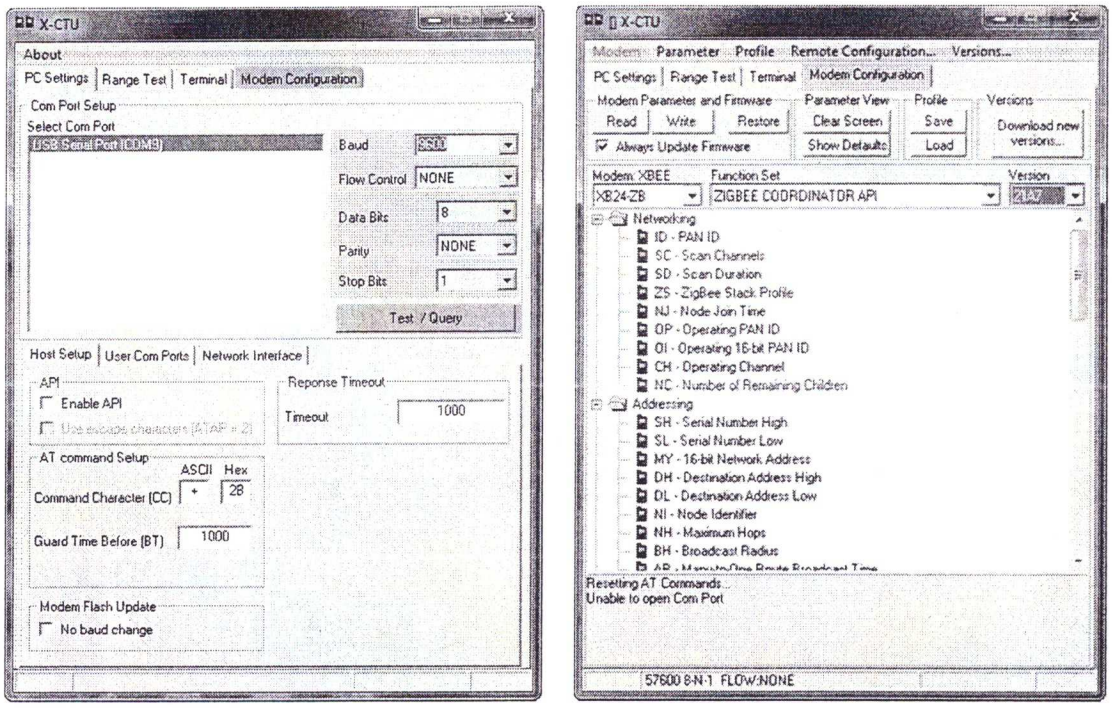


Figure 4.30 X-CTU Screenshots

Next is to confirm that everything is set up correctly is to click on the Test/Query button after selecting the COM port. If all goes well, a message will appear such in the figure below tells that the communication with the modem is OK and the message will gives us the modem type, the serial number or the address and firmware version as shown in figure 4.30

Under the Function Set, a list of firmware is listed for the class of radio modem such as coordinator and router/end device firmware, and under Version the latest version should be chosen for the Function Set to avoid errors.

Under each Function Set we have a list of configuration that needed to configure to certain values, the following table 4.8 illustrates the ones we will use:

Table 4.8 Coordinator AT command

AT Command	Name and Description	Parameter Range
ID	Used to determine the 64-bit PAN ID.	0 - 0xFFFFFFFFFFFFFF
JN	Set/read the join notification setting. If enabled, the module will transmit a broadcast node identification frame on power up and when joining.	0 - 1
DH	Destination Address High. Set/Get the upper 32 bits of the 64-bit destination address.	0 - 0xFFFFFFFF
DL	Destination Address Low. Set/Get the lower 32 bits of the 64-bit destination address.	0 - 0xFFFFFFFF
NI	Node Identifier. Stores a string identifier. The register only accepts printable ASCII data. In AT Command Mode, a string cannot start with a space.	20-Byte printable ASCII string
BR	Set/read the serial interface baud rate for communication between modem serial port and host.	1200 - 115200
SH	Read high 32 bits of modems unique IEEE 64-bit Extended Address.	0 - 0xFFFFFFFF
SL	Read low 32 bits of modems unique IEEE 64-bit Extended Address.	0 - 0xFFFFFFFF
RE	Restore Defaults. Restores module parameters to factory defaults.	--
WR	Writes parameter values to nonvolatile memory so that parameter modifications persist through subsequent resets.	--

This Table shows the configuration put for the ZigBee Coordinator

Table 4.9 Coordinator XBee Configuration window for the router

Modem	XB24-ZB
Function Set	ZIGBEE COORDINATOR AT
VERSION	22A7
ID	ABC555
DH	0
DL	FFFF
BD	6 (57600)
JN	1
NI	COORDINATOR

The XBee Coordinator modules serial terminal settings configured for both coordinator and routers are as following in table 4.10

Table 4.10 XBee settings used for serial terminal software

Baud	57600
Data	8 bit
Parity	None
Stop bits	1
Flow control	None
Local echo	On

4.2.4.2 Wireless Router

The wireless router is seen as the extension of the wireless controller, so a network could be established from 1 coordinator, and a large number of wireless routers,

The wireless router has two main components, the first is an XBee module responsible for communication with the coordinator, and the second is an Arduino microcontroller that performs various operations such as ADC conversions.

4.2.4.2.1 XBee Router

Prior to connecting the XBee in the wireless router, it has to be configured using the X-CTU application; the following is the general configurations for the router XBee as described in table 4.11

Table 4.11 Router XBee Modem Configuration window for the router

Modem	XB24-ZB	
Function Set	ZIGBEE ROUTER AT	
VERSION	22A7	
ID	ABC555	
DH	0 (for the Coordinator)	
DL	0 (for the Coordinator)	
BD	6 (57600)	
JN	1	
NI	For the first Router NI: ROUTER1	For the second Router NI: ROUTER2

Since the coordinator always receives a 16-bit address of 0x0000, a 64-bit address of 0x0000000000000000 is defined as the coordinator's address, for this we assign '0' for both DH and DL.

4.2.4.2.2 Arduino

The second component of the wireless router is the Arduino programmable board, and it was chosen as an external microcontroller partner for several important advantages to such as:

- Local logic, XBee radios can't be programmed to perform logical information processing.
- Additional input/output lines.
- Fast prototyping.
- Lots of connection options: such interacting with GPS modules, drive LCD display screens, store data in local memory banks, and interact directly with the Internet via WiFi or your mobile phone.
- Relatively inexpensive compared to other microcontroller platforms.
- Open source and extensible software and hardware.

The following is the schematics in figure 4.31 and the actual board in figure 4.31 for the Arduino Uno, the one used in pair with the XBee router:

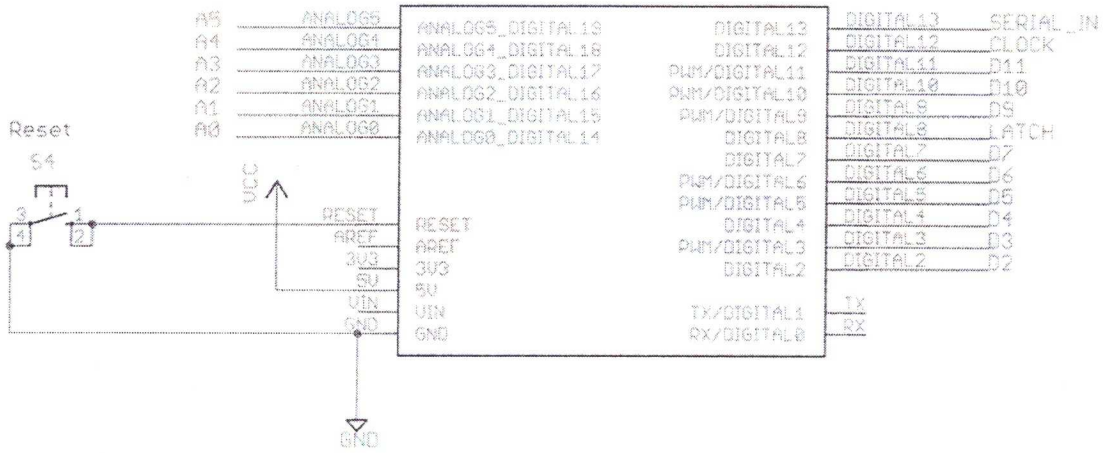


Figure 4.31 Arduino Basic Schematic

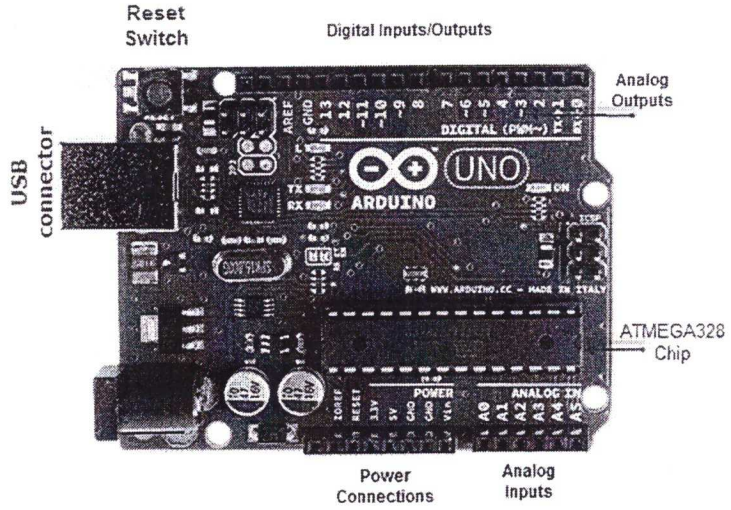


Figure 4.32 Arduino Uno

The Arduino Uno configurations and specifications are listed in the following table 4.12, where it is clear that the board is quite powerful and suitable for this application:

Table 4.12 Arduino UNO specifications

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 (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

4.2.4.2.3 Connecting the XBee to the Arduino

It takes only four pins of XBee to create a working connection so that the Arduino can communicate wirelessly, using its built-in serial communications protocol. The table 4.13 below illustrates the connection between Arduino and XBee.

Table 4.13 Pin connections between Arduino and XBee

Arduino	XBee
3.3 Volt	VCC
Pin# 0 (RX)	TX
Pin# 1 (TX)	RX
GND	GND

Arduino uses serial commands to send information out via the XBee, and to read in any information that's received.

Figure 4.33 illustrates the schematic connection of Arduino and XBee, the connected pins are described in table 4.13.

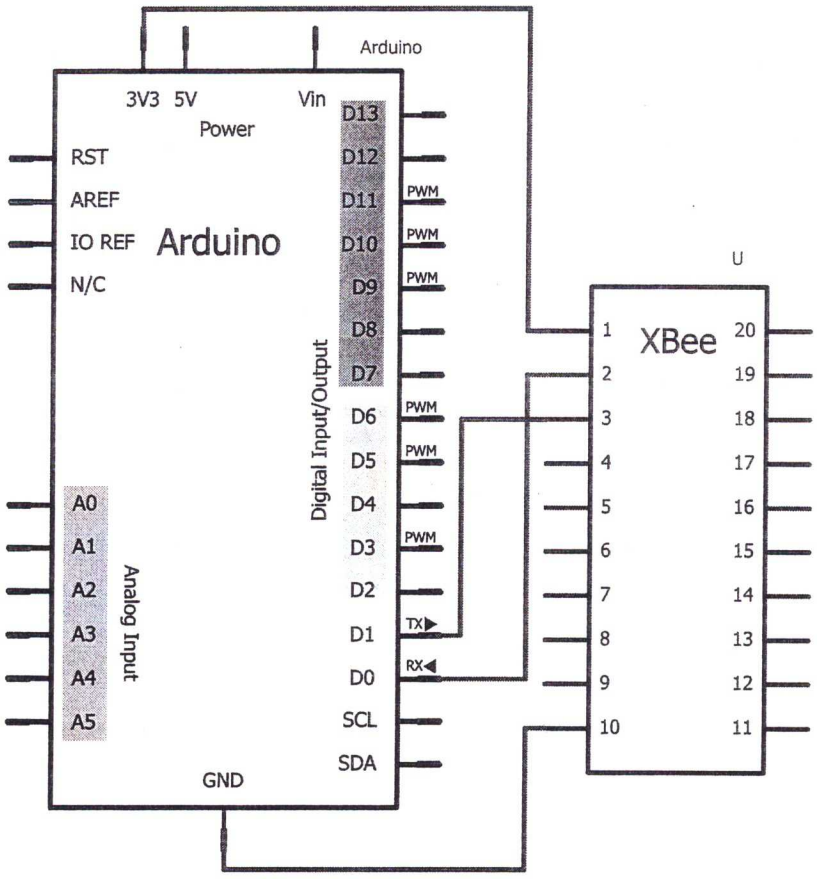


Figure 4.33 Arduino-XBee schematic connection

Table 4.14 Wireless controller connectors

	Wireless controllers pins	Pins description
	A0-A5	Analog inputs (for Analog sensors)
	D2-D6	Digital inputs (for Digital sensors)
	D7-D13	Digital outputs (for Control nods)

4.2.5 Keypad

A keypad is a matrix of switches used to provide input for numeric digits. The buttons on this particular keypad are setup in a 3X4 matrix format so we only need 7 pins to detect the pressing of 12 keys. It is setup like this to minimize the number of pins needed to control the keypad.

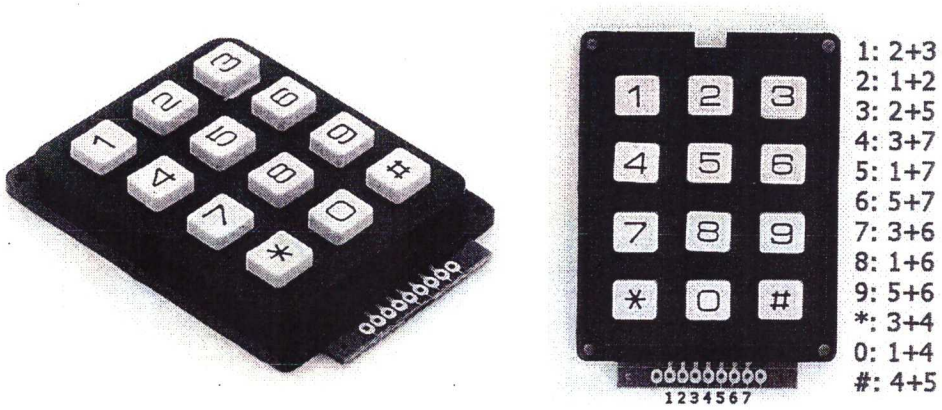


Figure 4.34 3X4 matrix keypad

A secret code must be entered on the keypad, and if it is correct, a lock will open; otherwise, the lock will stay closed. In the following figure 4.35: the illustration to the row column pins assignment is shown.

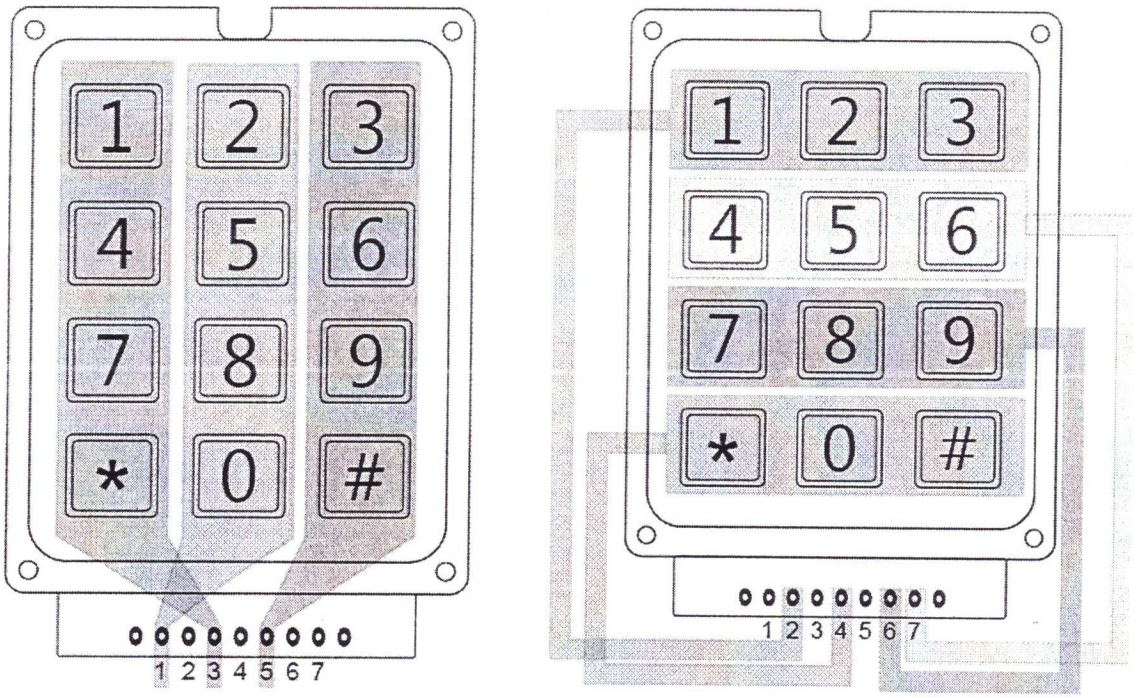


Figure 4.35 3x4 matrix keypad column row pin assignment

The buttons on this particular keypad are setup in a 3X4 matrix format so we only need 7 pins to detect the pressing of 12 keys. For example, when you hit the number 2 pins 1&2 are connected, 6 connects pins 5&7 and 9 connects 5&6. So in code we just look for the combination and we know what button is being pressed. So in simple terms, if pins 5 and 2 are signaling the microcontroller, it means that button 3 has been pressed it is setup like this to minimize the number of pins needed to control the keypad.

In the following figure a schematic diagram for keypad, it illustrates the pins of columns and rows of key pad

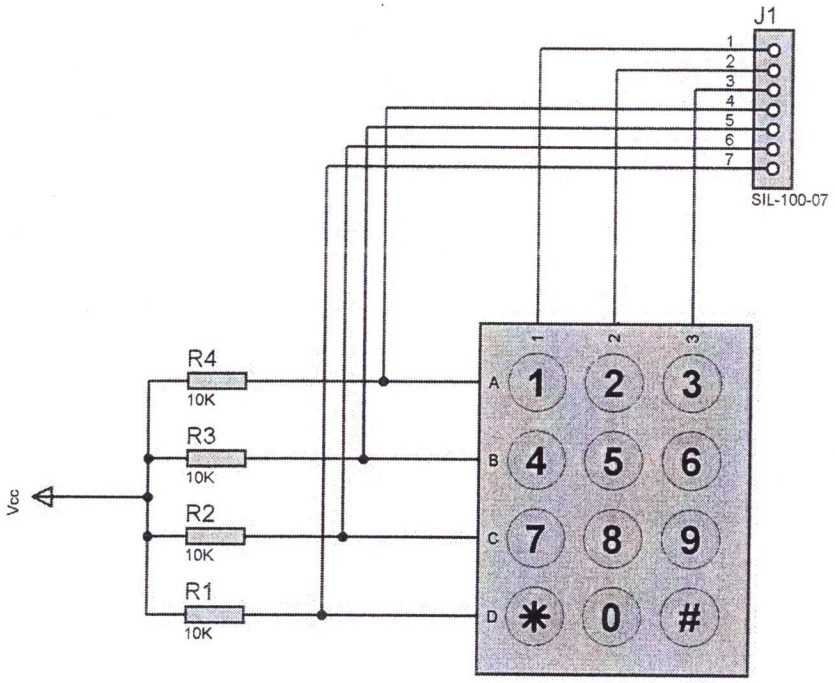


Figure 4.36 Keypad Schematic

Pins 3, 5, 6 & 7 all connected through a resistor to 5v.The resistors can range from 1K to 10Kohm, we are using 10Kohm. After that we connect the pins to digital ports in Arduino.

The source code for the keypad is attached in the code appendix (Appendix E).

4.2.6 Light Intensity Sensor

The choice for the light intensity sensor is the photocell sensor, as shown in figure 4.37 below; the output voltage will be sampled using an ADC channel on the microcontroller. Since the resistor and the voltage reference are connected to the same voltage source, variations in the supply voltage will not influence the readings. The range and threshold values of the photocell circuit were determined through thorough testing in various lighting environments. Basically, the testing consists of gathering data in various lighting environments to characterize the circuit, the data was then correlated with various light intensity levels.

The photocell sensor can be configured to give continuous updates at a user specified rate, or to send an alert when a threshold is reached.

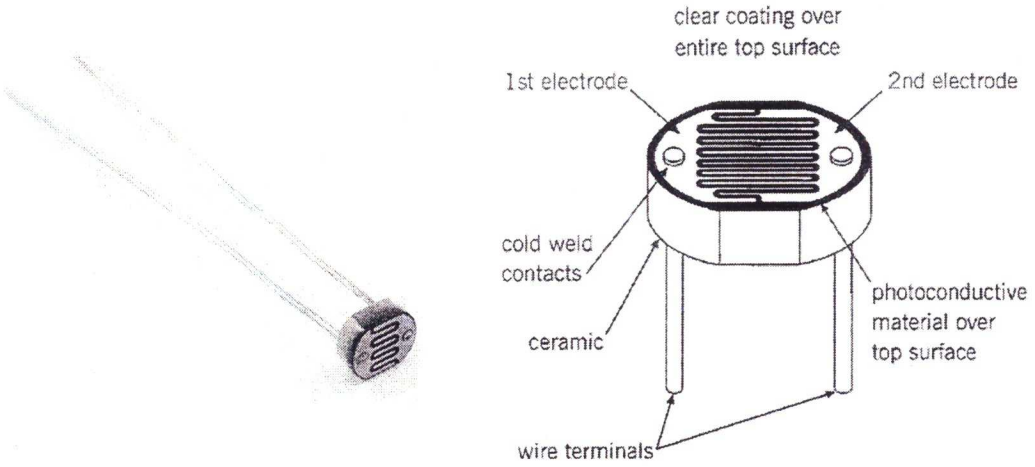


Figure 4.37 Photocell Sensor

When light strikes the photocell the internal resistance decreases. When there is no light striking on the photocell, then resistance increases. The resistance varies between 1K ohms and 10M ohms. So using a simple connection as described in figure 4.x, an analog signal can be generated with the following value as the output voltage:

Equation 4.1 Photo sensor Output Voltage

$$Vo = Vcc\left(\frac{R}{R + Photocell}\right); R = 10k\Omega$$

The output is not linear voltage with respect to brightness. That is, the voltage is proportional to the inverse of the photocell resistance which is, in turn, inversely proportional to light levels. Figure 4.38 illustrates the connection schematics for the photocell:

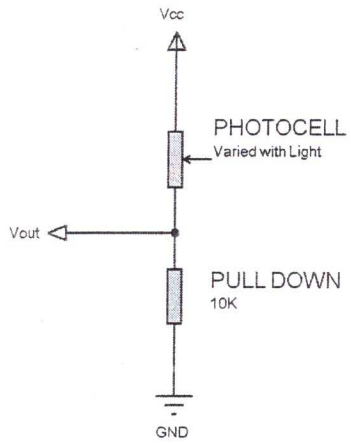


Figure 4.38 Photocell Sensor

The following table 4.15 illustrates the values of the photocell to be calculated by the microcontroller:

Table 4.15 Photocell values

Lighting	Ambient light (lux)	Photocell resistance	Expected Output
Moonlit night	1 lux	70 KΩ	0.1V
Dark room	10 lux	10 KΩ	0.6V
Bright room	100 lux	1.5 KΩ	2.5V
Overcast day	1000 lux	1.3 KΩ	4.3V
Full daylight	10,000 lux	1.1 KΩ	5V

4.2.7 Current Sensor

The choice for current Transducer sensing element is CE-IJ03 sensor shown in figure 4.39 below. This sensor produces output current proportional to the input average RMS AC current. The output current is translated into a 0-5V DC Output directly proportional to a line carrying 0-40A current over 220VAC. And It is used when the AC current waveforms are not purely sinusoidal and it is more precise and accurate than other transducers. So it is perfect for standard electrical Grids.

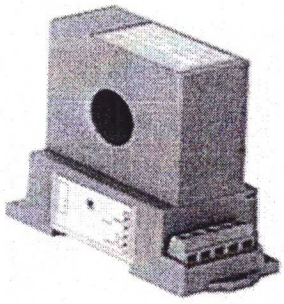


Figure 4.39 CE-IJ03 Current transducer

CE-IJ03 sensor has a 4-pin connection one pin is ground, another is output voltage, another is output current and the final one is power. The power used with this sensor is 12V DC inputs .the figure 4.40 below shows the current transducer pins.

Equation 4.2 Conversion ratio of current transducer equation

$$The\ conversion\ ratio\ = \frac{5-0}{40-0} = 0.125\ V/A$$

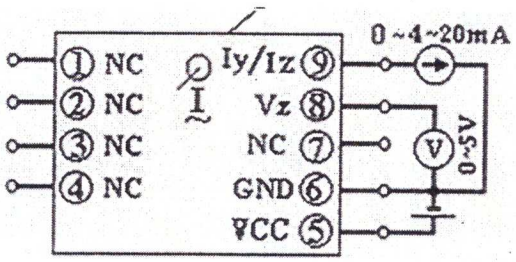


Figure 4.40 CE-IJ03 Pins

4.2.8 Water Flow

The choice for water flow sensing element is JXL11H94 sensor. JXL11H94 consist of a base, impeller assembly and the speed sensor measurement unit. So it detects the water flow from the water flowing through the base and the rotations of the driven impeller assembly, the hall element sensor installed in impeller components on the ring changes in the magnetic field and convert it into electrical impulses. So the frequency pulse reflects the flow of the water.

JXL11H94 sensor has a 3-pin connection one pin is ground, another is output frequency signal and the final pin is the input voltage signal, as shown in the figure below:

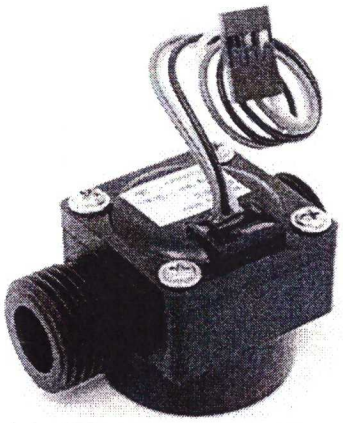


Figure 4.41 JXL11H94 Water Flow Sensor

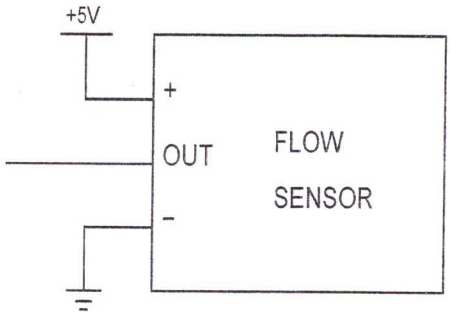


Figure 4.42 Water flow Schematic

Equation 4.3 The output of JXL11H94 sensor is a frequency pulse

$$Q = \frac{f + 3}{7.0}$$

Where f: frequency.
Q: flow rate (L/min).

4.2.9 Temperature sensors

4.2.9.1 LM35 Temperature Sensor

The temperature sensing element is LM35 temperature sensor showing in figure 4.43 . The LM35 temperature sensor produces an analog voltage directly proportional to temperature with an output of 1 millivolt per 0.1°C (10 mV per degree). It draws only 60 micro amps from its supply and possesses a low self-heating capability. The sensor self-heating causes less than 0.1 °C temperature rise in still air.

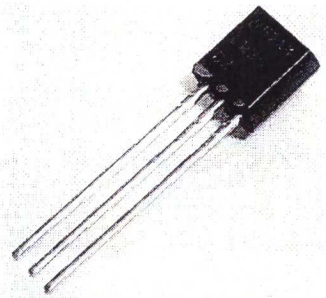


Figure 4.43 LM35 Temperature Module

It has 3-pins one is the power (4V to 20V), another one in the analog output and the final one is the ground as show in figure. The temperature can be calculated using the following equation:

Equation 4.4 Temperature Equation

$$T_C = 2 + V_{out}x\ (100V/°C)$$

Where Tc is the temperature coefficient of the sensor (10mV/deg C), and Vout is the output voltage of the sensor.

The following figure illustrates the schematic diagram of the LM35 sensor which has 3-pins the VCC which is equal to 5 volt, ground and the analog output to the microcontroller as shown.

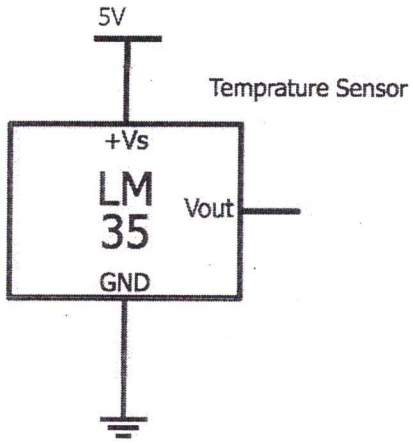


Figure 4.44 schematic

4.2.8.2 TSC-8218 TEMPERATURE SENSOR

TSC-8218 temperature sensor series of NTC or Negative Temperature Coefficient that consists in a simple component that provides a variable resistance those changes with temperature, which we can read into the Arduino microcontroller board as an analog value.

It can be used to sense the water temperature within a tank. Figure 4.45 shows the TSC-8218, it has inductive element with NTC 10K ohm, with working range 0 to 50C.

Figure 4.45 TSC-8218

The following is the schematics of the TSC-8218 Temperature Sensor connection:

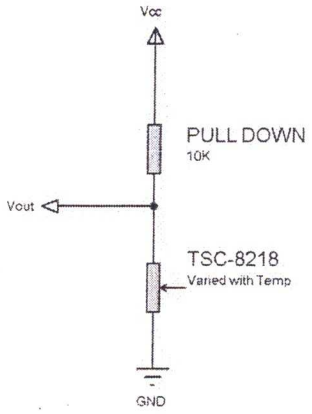


Figure 4.46 TSC Temperature Sensor

The resistance of the sensor can be calculated and compared to the following table 4.16 where it illustrates the value of the calculated resistance:

Table 4.16 Resistance Values

TEMPERATURE ℃	RESISTANCE Ω	TEMPERATURE ℃	RESISTANCE Ω	TEMPERATURE ℃	RESISTANCE Ω
0	32600	17	14318.0	34	6810.50
1	30985	18	13676.9	35	6534.00
2	29459	19	13068.1	36	6270.00
3	28017	20	12489.8	37	6018.00
4	26654	21	11940.3	38	5778.00
5	25365	22	11418.0	39	5548.70
6	24145.2	23	10921.4	40	5330.00
7	22991.4	24	10449.2	41	5120.00
8	21899.2	25	10000.0	42	4920.00
9	20865.2	26	9572.60	43	4729.00
10	19885.8	27	9165.80	44	4547.00
11	18957.9	28	8778.50	45	4372.00
12	18078.6	29	8409.60	46	4205.00
13	17245.0	30	8058.30	47	4045.00
14	16454.5	31	7723.60	48	3893.00
15	15704.7	32	7404.60	49	3746.30
16	14993.2	33	7100.58	50	3606.00

The resistance is calculated using the following equation:

Equation 4.5 Thermostat resistance equation

$$R_{therm} = \frac{10k}{(\frac{1023}{ADC} - 1)}$$

ADC represents the 10bit digital equivalent to the output of the sensor output voltage.

4.2.10 Motion Sensor

The choice for the motion sensing element is the parallax PIR sensor. Passive Infrared Sensors allow us to sense motion and detect a human has moved in the range of sensors. It is a pyroelectric device that detects motion by measuring changes in the infrared levels emitted by surrounding objects. They are small, inexpensive, low-power and easy to use.

The PIR sensor module is a prepared circuit that includes PIR sensor. It gives a simple on/off signal. When it detects a motion it gives logic high to its output pin.

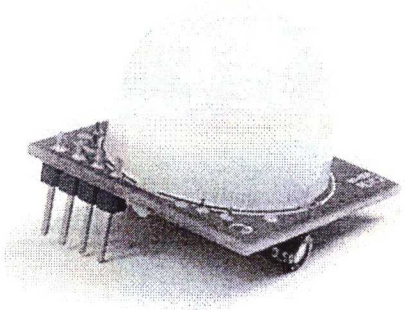


Figure 4.47 PIR sensor module

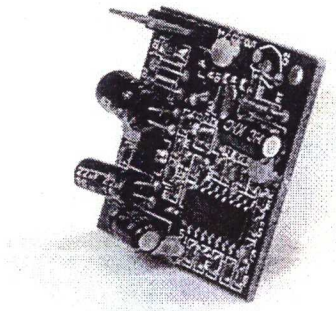


Figure 4.48 PIR sensor module

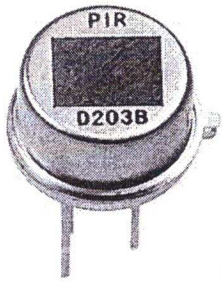


Figure 4.49 PIR sensor core

PIR sensor module has a 4-pin connection at the bottom as shown in the Figure one pin is ground, another is output signal, another is CDS and the final one is power. Power is usually between 3.5 - 12 V DC inputs. CDS pin is a CMOS Schmitt trigger input pin. It is used to distinguish between day time and night time. This pin is connected to VCC to disable it.

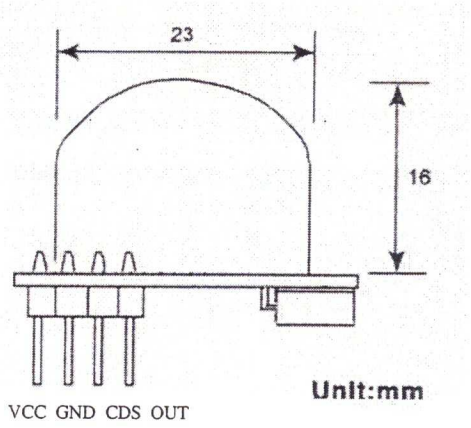


Figure 4.50 PIR Pins

4.2.11 MQ gas sensors

The MQ series of gas sensors use a small heater inside with an electro-chemical sensor. They are sensitive for a range of gasses.

They was chosen because they have high sensitivity and low cost. The following figure illustrates the MQ gas sensors schematic

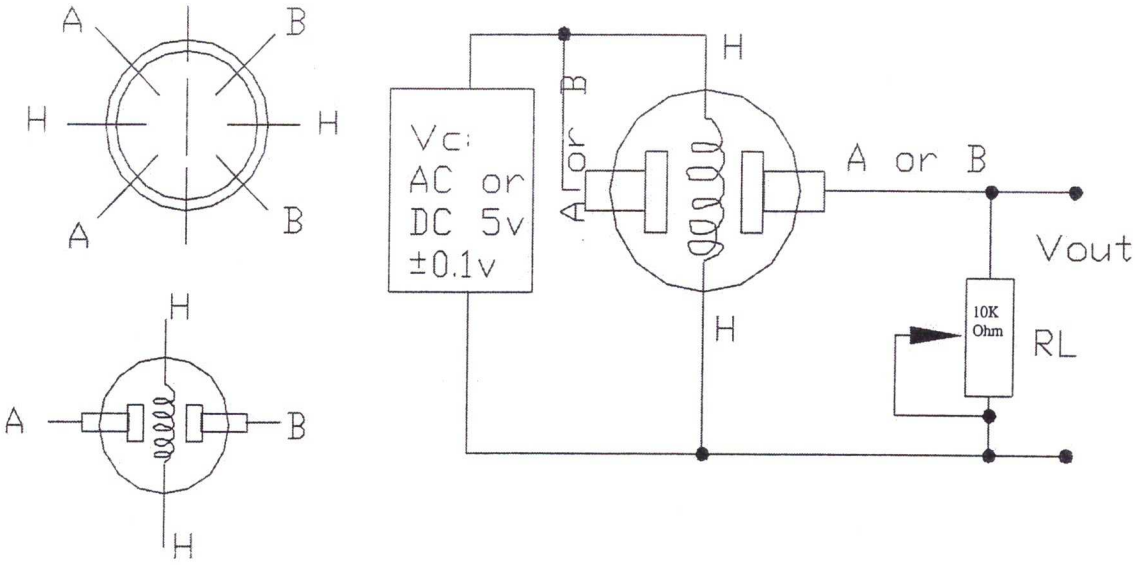


Figure 4.51 MQ gas sensor schematic

To hook up MQ gas sensors with breadboard, it should be noted that they have 6-legs that are not compatible with the breadboard spacing, so gas breakout boards were used, also it was used to simplifying the interface from 6 to 3 pins which are ground, VCC, and an analog voltage output. The following figures show breakout board.

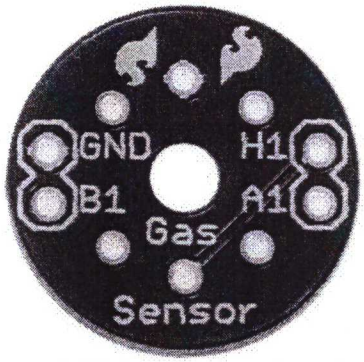


Figure 4.52 Gas breakout board, top view

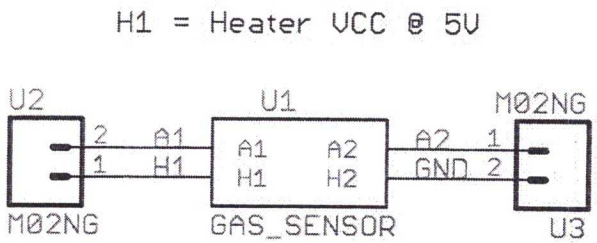


Figure 4.53 Gas breakout board schematic

4.2.10.1 Connections of MQ gas sensors

With regard to breakout connection, H1 and A1 connected to 5 V, GND to ground, and B1 to an analog input of Arduino microcontroller.

A 10K ohm RL variable resistor is connected between B1 and the ground. The MQ gas sensors has to no polarity set, so the gas sensor will work with the board any way it fits into the 6 pins.
The potentiometer value could be changed to adjust the sensitivity of the sensor and thus one could change the level of warning of the sensor's outputs.

4.2.10.2 MQ-2

MQ-2 gas sensor in Figure 4.54 is used to detect the presence of a dangerous flammable gas leaking and air quality. It has high sensitivity to LPG, Propane and Hydrogen, also could be used to Methane and other combustible steam.

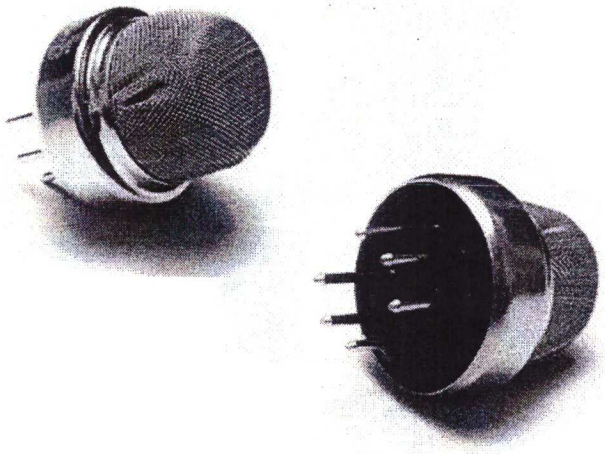


Figure 4.54 MQ-2 gas sensor

MQ-2 could be used in ketchin to detect cooking gas, aslo industerial buildings to the flammable gas leaking and air quality.

4.2.10.3 MQ-3

MQ-3 gas sensor in Figure 4.55 and Figure 4.56 are used to detect the presence of alcohol gas. The sensor's conductivity is higher along with the gas

concentration rising. MQ-3 gas sensor has high sensitivity to Alcohol, Ethanol and has good resistance to disturb of gasoline, smoke and vapor.

MQ-3 could be used in some industrial buildings. The potentiometer of the sensor used to adjust the sensitivity to the suitable concentration of gas depending on the environment surrounding the sensor.

Figure 4.55 MQ3, bottom view

Figure 4.56 Figure: MQ3, top view

4.2.10.4 MQ-4

MQ-4 gas sensor in Figure 4.57 is used to detect the presence of natural gas composed of mostly Methane CH4 gas and Small sensitivity to alcohol, smoke. The MQ-4 can detect natural gas concentrations anywhere from 200 to 10000ppm.

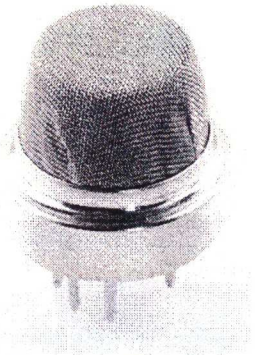


Figure 4.57 MQ-4

MQ-4 also could be used in ketchin to detect cooking gas, aslo industerial buildings to the flammable gas leaking and air quality.

4.2.10.5 MQ-7

MQ-7 gas sensor in Figure 4.58 is used to detect the carbon monoxide (CO) concentrations in the air from 20 to 2000ppm.

Figure 4.58 MQ-7

Carbon monoxide is colorless and odorless gas and is a normal constituent of exhaust gases from incomplete combustion. Levels of carbon monoxide inside buildings should not exceed 9 ppm. Exposure to carbon monoxide at levels even as low as 35 ppm could cause bodily fatigue.

If levels inside a building are detected greater than 100 ppm, the building should be evacuated. The Department of public health guidelines for indoor air quality recommends that all buildings with indoor combustion sources install carbon monoxide

A variable resistor RL with 10K ohm is to be connected and adjusted to minimum which is adjust the MQ-7 sensitivity to suit the indoor building environment.

4.2.11.1 Sensor Values

It should be mentioned that its extremely difficult to translate an analog signal to a real world meaning in ppm regarding gas intensity, because too many variables control the output of the sensors, variables such as temperature and humidity.

However; theses sensors are adjustable, and they work by providing a reference to a specific acceptable value, all of which is described thoroughly in the testing phase.

4.2.12 GSM Modem

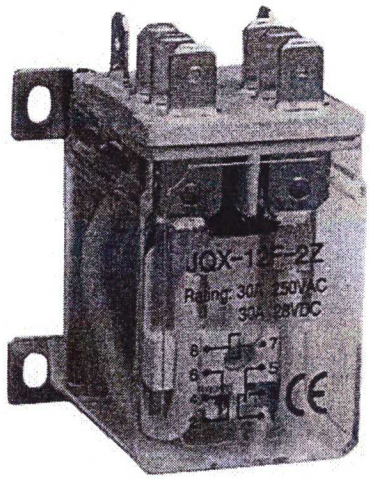
For this particular component, the widely available GSM modem, advertised as a USB HSDPA modem was chosen to provide the required tasks. It is extremely affordable, small in size, and provides all the necessary actions required for the short messaging service access:

Figure 4.59 HSDPA GSM Modem

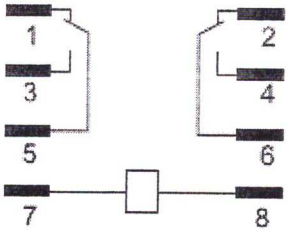
The GSM Modem implementation is handled directly via the C# Software Code directly, without any special wiring for this component.

4.2.13 Control Node

The control node represents the actual electrical switch which manages a high power line, for this project ,they vary in Current rating and each one is set to a specific application, the following relays were used:

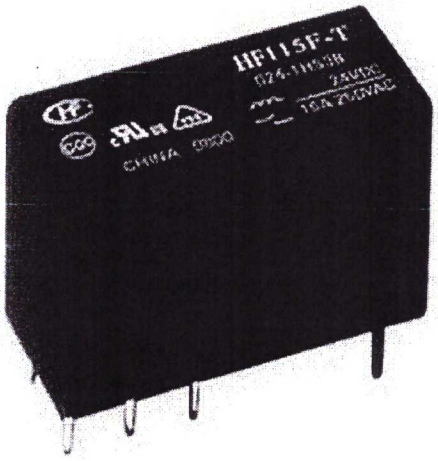


Rated Coil Power: 1.5W
Contact Capacity: 30A 250VAC/ 12VDC
Terminal Type: Screw
Wiring Diagram:



Application: Main Lines, and High Power Load Lines.

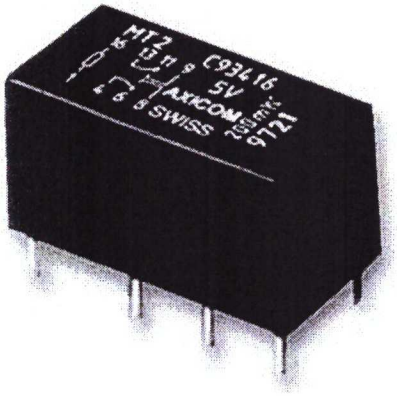
Figure 4.60 JQX-12F Relay, 30A Rating



Rated Coil Power: 0.4W
Contact Capacity: 12A 250VAC/ 5VDC
: 12A 250VAC/ 12VDC
Wiring Diagram:

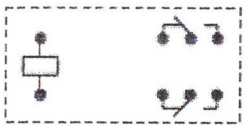
Application: Used for load lines and wireless Controllers (5V)

Figure 4.61 HF115F Relay



Rated Coil Power: 0.15W
Contact Capacity: 2A 250VAC/ 5VDC
: 2A 250VAC/ 12VDC

Wiring Diagram:



Application: Used for Lighting and wireless Controllers (5V)

Figure 4.62 C93402 Relay

These relays are considered DPDT relays, which means they work as either normally open nodes or normally closed, they are connected according to the following schematics:

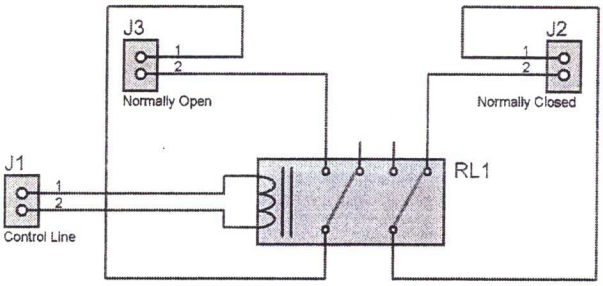


Figure 4.63 DPDT Relays Schematics

4.2.14 Remote Control

For remote control, we used a small form wireless 2.4GHz Keyboard and mouse, where the following are it components.

Figure 4.64 Remote Control

This approach uses a widely available control mechanism that is available in many forms, such as the widely popular Logitech remotes and controllers.

4.2.15 Android Mobile

The Galaxy S2 phone shown in figure 4.65 has been chosen to implement mobile control over the system, due to the fact that it's available with the students responsible for the implementation of this project, most of the implementation is done within the software.

Figure 4.65 Galaxy S2

4.3 Software Implementation

This section describes the implementation of the software modules in this project.

4.3.1 Hardware Drivers Modules

Includes the code used to operate the hardware as required to perform the required tasks .

4.3.1.1 PIC USB Driver

For this component, the code is quite large and thus it is included with the appendix of this document. The following code is the implementation of the key function specified in chapter 3:

- Connect Function, this function establishes a connection the microcontroller.

```
Public void Connect()
 {
//Step 1
  DEV_BROADCAST_DEVICEINTERFACEDeviceBroadcastHeader =
newDEV_BROADCAST_DEVICEINTERFACE();
  DeviceBroadcastHeader.dbcc_devicetype = DBT_DEVTYP_DEVICEINTERFACE;
  DeviceBroadcastHeader.dbcc_size =
(uint)Marshal.SizeOf(DeviceBroadcastHeader);
  DeviceBroadcastHeader.dbcc_reserved = 0;       //Reserved says not to use...
  DeviceBroadcastHeader.dbcc_classguid = InterfaceClassGuid;
//Step 2
  IntPtrpDeviceBroadcastHeader = IntPtr.Zero;  //Make a pointer.
  pDeviceBroadcastHeader =
Marshal.AllocHGlobal(Marshal.SizeOf(DeviceBroadcastHeader));
  Marshal.StructureToPtr(DeviceBroadcastHeader, pDeviceBroadcastHeader,
false);  //Copies

RegisterDeviceNotification(this.Handle,pDeviceBroadcastHeader,DEVICE_NOTIFY_WI
NDOW_HANDLE);
//Step 3
  if(CheckIfPresentAndGetUSBDevicePath())
   {
    uintErrorStatusWrite;
    uintErrorStatusRead;
    WriteHandleToUSBDevice = CreateFile(DevicePath, GENERIC_WRITE,
FILE_SHARE_READ |
                            FILE_SHARE_WRITE, IntPtr.Zero, OPEN_EXISTING, 0,
IntPtr.Zero);
    ErrorStatusWrite = (uint)Marshal.GetLastWin32Error();
    ReadHandleToUSBDevice = CreateFile(DevicePath, GENERIC_READ,
FILE_SHARE_READ |
                           FILE_SHARE_WRITE, IntPtr.Zero, OPEN_EXISTING, 0,
IntPtr.Zero);
    ErrorStatusRead = (uint)Marshal.GetLastWin32Error();
```

```
//Scenario 1
  if((ErrorStatusWrite == ERROR_SUCCESS) && (ErrorStatusRead ==
ERROR_SUCCESS))
      {
        AttachedState = true;
        AttachedButBroken = false;
        StatusBox_txtbx.Text = "Device Found, AttachedState = TRUE";
      }
  else//Scenario 2
    {
      AttachedState = false;
      AttachedButBroken = true;
     if(ErrorStatusWrite == ERROR_SUCCESS)
         WriteHandleToUSBDevice.Close();
     if(ErrorStatusRead == ERROR_SUCCESS)
         ReadHandleToUSBDevice.Close();
    }}
  else //Scenario 3
   {
     AttachedState = false;
     AttachedButBroken = false;
   }
  if (AttachedState == true)
   {
    StatusBox_txtbx.Text = "Device Found, AttachedState = TRUE";
   }
  else
  {
   StatusBox_txtbx.Text = "Device not found, verify connect/correct firmware";
  }
```

Register for WM_DEVICECHANGE notifications. This code uses these messages to detect plug and play connection/disconnection events for USB devices

Need to get the address of the DeviceBroadcastHeader to call RegisterDeviceNotification , but can't use "&DeviceBroadcastHeader". Instead, using a roundabout means to get the address by making a duplicate copy using Marshal.StructureToPtr().

Allocate memory for a new DEV_BROADCAST_DEVICEINTERFACE structure, and return the address theDeviceBroadcastHeader structure into the memory already allocated at DeviceBroadcastHeaderWithPointer .

Now make an initial attempt to find the USB device, if it was already connected to the PC and enumerated prior to launching the application.

If it is connected and present, we should open read and write handles to the device so we can communicate with it later.

If it was not connected, we will have to wait until the user plugs the device in, and the WM_DEVICECHANGE callback function can process the message and again search for the device.

After all this has been implemented, the next step is to check and make sure at least one device is attached with the matching VID/PID.

Then the code checks for multiple scenarios and reports them using global Boolean flags, the first describes successful device connectivity, the second describes a successful connection but reports an error with data communication, the third describes a lack of connectivity.

4.3.1.2 XBee Serial Module

The following functions are implemented to deal with XBee Module :

- `Connect(String COM)` : this function initializes the serial port, and gives it the connection properties for the serial modules, which are the following:

    - Baud Rate: 57600 .
    - Data units Bits: 8 .
    - Stop bits: 1 .
    - Parity: No parity .

    That sets other configurations for data such as the handshaking mode, the data shape, and binds an event to interrupt execution should data be received the function then starts the connection and reports in its status .

- `setControlPort(String RouterID, String Pin, int state)` : this function sets a specific control port within a wireless node to a specific state the state is a digital (0 or 1), however, the ports can be configured to output an analog value(0 to 255) .

- `Identify(String IncomingString)` : this function is called when the router receives data in a packet format contain information about the analog sensor then this function packs each value, and put the most recent value within a data structure repository concerning each wireless node, which contain a list of connected routers .

data received are shaped to this packet format :

```
S:[NodeAddress]:[PinAddress]:[Value]     //Sensor Packet starting with an S
C:[NodeAddress]:[PinAddress]:[Value]     //Control Packet starting with an S
```

```
public void Connect(String COM)
        {
            try
            {
                port = new SerialPort(COM, 57600, Parity.None, 8,
StopBits.One);
                port.Handshake = Handshake.RequestToSend;
                port.NewLine = System.Environment.NewLine;
                port.DataReceived += new
SerialDataReceivedEventHandler(DataReceived);
                port.ReadTimeout = 6000;
                port.WriteTimeout = 5000;
                port.Open();
                if (port.IsOpen)
                {
                    port.WriteLine("AT");
                    connected = true;
                    MessageBox.Show("Connected Successfully");
                }

                InitializeTimer(2000);
            }
            catch (Exception ex)
            {
                MessageBox.Show("Connection Error");
            }}

public void setControlPort(String RouterID, String Pin, int state)
        {
            str = "-W" + RouterID + Pin + state.ToString();
            port.WriteLine(Header+str+CalcFooter(str));
        }

private void Identify(String IncomingString)
        {
            String[] dataSegments = IncomingString.Split(":");
            int PinID, Value;
            String NodeID;
            RouterNode myRouter;
            try
            {
```

```
            if (dataSegments.Length == 4)    //Correct Format of 4
Strings
             {
                 NodeID = dataSegments[1];
                 PinID = int.Parse(dataSegments[2]);
                 Value = int.Parse(dataSegments[3]);
                 myRouter = (RouterNode)Routers[NodeID];

                 if (dataSegments[0] == "S")
                 {
                     myRouter.SensorPin[PinID] = Value;
                 }
                 else if (dataSegments[0] == "C")
                 {
                     myRouter.ControlPin[PinID] = Value;
                 }}}
            catch
              {
                  //Disregard, Invalid Data Format
              }}}
```

4.3.1.3 SMS Module

In this module we use a GSM modem to provide a messaging service as a means of mobile control; the following functions are implemented to deal with the SMS Module:

- Send (): Use this function to write data to the serial port output buffer.
- Data Received (): This function will be executed when data is received through the serial port as an implementation of the C# internal "DataReceived" event

```
private void btnsend_Click(object sender, EventArgs e)
    {
      port.WriteLine(AText.Text);
    }

void DataReceived(object sender, SerialDataReceivedEventArgs e)
    {
      String Incoming;
      Incoming = port.ReadExisting();
      segment(das);
    }
```

- Segment (string data) :This function is used to performs comparison operation to distinguish between the type of reply the GSM responds with; each type of

response is tied with a particular function that handles the incoming message. The following is the implementation of this function:

```
public void  segment(string data)
        {
            char[] delimeter={',',"",':','\n'};
            sms1.Clear();
            string[] words = data.Split(delimeter);
            foreach (string s in words)
                {
                     sms1.Add(s);
                }
            for (int i = 0; i <sms1.Count; i++)
                {
                    string inputString = sms1[i];
                    char[] charsToRemove = new char[] { '\r', '\t', '\n', ' ' };
                    string[] results = inputString.Split(charsToRemove);
                    StringBuilder transformedString = new StringBuilder();
                    foreach (string s in results)
                       {
                         transformedString.Append(s);
                       }
                    switch (transformedString.ToString())
                      {
                          case "+CMGL":
                          SMS_Received(sms1, i);
                          Add_MSG(sms1, i);
                          break;
                     case "+CMTI": SMS_Read(sms1, i);
                                   break;
                     case "+CMGR" :MessageBox.Show("Check CMGR  "+data);
                                   Add_MSG(sms1, i);
                                   break;
                        } } }

void SMS_Received(List<string> message,int start)
           {
              string Index, Num, Msg;
              if (message.Count > start + 17)
                 {
                     Index = message[start + 1];
                     Num = message[start + 6];
                     Msg = message[start + 17];
                 }
            }

void SMS_Read(List<string> message, int start)
        {
             string num = message[start + 4];
             port.WriteLine("AT+CMGR=" + num + "\"");
        }
```

- Send message (): this function is used to send SMS messages to mobile using at commands. It begins with "AT" to initiate the communication, then

"AT+CMGF=1" to change the mode of data, and finally it sends "AT+CMGS ..." with the message and mobile number.

```
private void Send_SMS(object sender, EventArgs e)
    {
        port.WriteLine(AText.Text);
        MethodInvoker action = delegate
        { txtsms.Text += AText.Text; };
        txtsms.BeginInvoke(action);
         string msg = AText.Text;
         string mobilenum = textBox1.Text;
         port.WriteLine("AT" + (char)(13));
         port.WriteLine("AT+CMGF=1" + (char)(13));
         port.WriteLine("AT+CMGS=\"" + mobilenum + "\"");
         port.WriteLine(msg + (char)(26));
    }
```

4.3.1.4 Socket Communication Modules

To develop the socket communication modules, TCP sockets were chosen for that ask, as they present a reliable method of communication between two nodes on a LAN network. Moreover, the following protocol format is chosen as method of communication between the server and clients, it contains the following fields:

Description	Packet Type	Password	Controller ID	Pin ID	State (Optional)
Size	1 Char	6 -18 Chars	4 Chars	2 Chars	1 Char
Command	"C"	<Any Value>	<Numerical>	<Numerical>	"0" or "1"
Query	"Q"	<Any Value>	<Numerical>	<Numerical>	None

It should be mentioned that each of those fields is separated by a delimiter symbol ';' for easier handling in the code.

4.3.1.4.1 Socket Server Module

The server module implements a TCP socket module, where it awaits connections, and start asynchronously communicating with every single client that has requested communication, each client is assigned a socket from within a list to identify its IP Address and port number, then an Asynchronous Call Back function is called, and is put under an infinite loop to await input from the client.

These are the main code segments that represent the socket server module, they include but not limited to the following:

- Traffic control data: Lists and data items containing control data, incoming messages, and a list of clients kept in the server so it can enable it to have one to one communication with each client.

```
…
private int numberOfClients = 0;
private String CurrentIP = "IP Not Assigned";
public AsyncCallback pfnWorkerCallBack;
private Socket ServerSocket;
private int port=-1;
SystemPacket MostRecentPacket;
List<Socket> ClientSockets = new List<Socket>();

//------- Data and Control
String ReceivedString = "";
int ReceiveCount = 0;
bool newMessage = true;
…
```

- Functions:

All of the functionalities specified in the design phase have been implemented, and the following are an example of them:

- `public void StartListening(int PortNumber)` : this function initiates the server on the given Port.
- `public void Broadcast(String Text)`: sends a packet in string format to all clients.
- `public void OnClientConnect(IAsyncResult asyn)`: this function is called whenever a new client request is made, it establishes a connection, then it register it, after that it initiates other functions that work asynchronously for each client as it receives data from it.
- `public void OnDataReceived(IAsyncResult asyn)`: this function handles incoming data and notifies the module of the text received.
- `public String GetIP()`: this function retrieves the IP of the Device running the server so it can be passed.

The following is a code segment from the function `OnClientConnect` where it handles a new connection:

```
Socket CurrentClient = ServerSocket.EndAccept(asyn);
ClientSockets.Add(CurrentClient);
NewClientAdded(ClientSockets.Count.ToString(), ClientSockets.Count);
WaitForData(CurrentClient);
// Now increment the client count
numberOfClients++;

ServerSocket.BeginAccept(new AsyncCallback(OnClientConnect), null);
```

A client is instantiated on a new request, added to a list of clients, function handlers then are instantiated to handle the data exchange specific to that client. More information is included within the Code Appendix.

4.3.1.4.2 Android Socket Client

This module is implemented on the android environment which implements Java code, it implements basic socket client functionalities, and it is presented as the foundation an infrastructure for a more user friendly application.

The following is the interface of the app implemented on the Galaxy S2, it is shown as the following:

Figure 4.66 Android Application GUI

The code behind the graphical interface implements several functions, most important of which are the following:

- `public void SendData(String str)` : This function is responsible for sending strings to the server, it acts as a port of communication for the server.
- `public void StartConnection(String ServerIP,int ServerPort)`: This function initiates the connection, and sends a TCP connection request to the server, identified by its IP and Port.
- `public void closeConnection()` : this function ends communication with the server and successfully disposes of the reserved port.
- `class DataReader extends Thread` : This class extends the java thread class and is instantiated when a connection is started; where it continues asynchronously waiting for data from the server, and displaying it should the code require to do so .

The code is well commented and explained in the code appendix (Appendix E).

4.3.2 Graphical Modules

The following modules were written in the c# language and using the windows presentation foundation XAML language, which is very similar to XML:

4.3.2.1 Interface Style Sheet

This component contains the key graphical components included with all the other GUI components; it contains a number of moving vectors and a light blue gradient in the background. It looks like this:

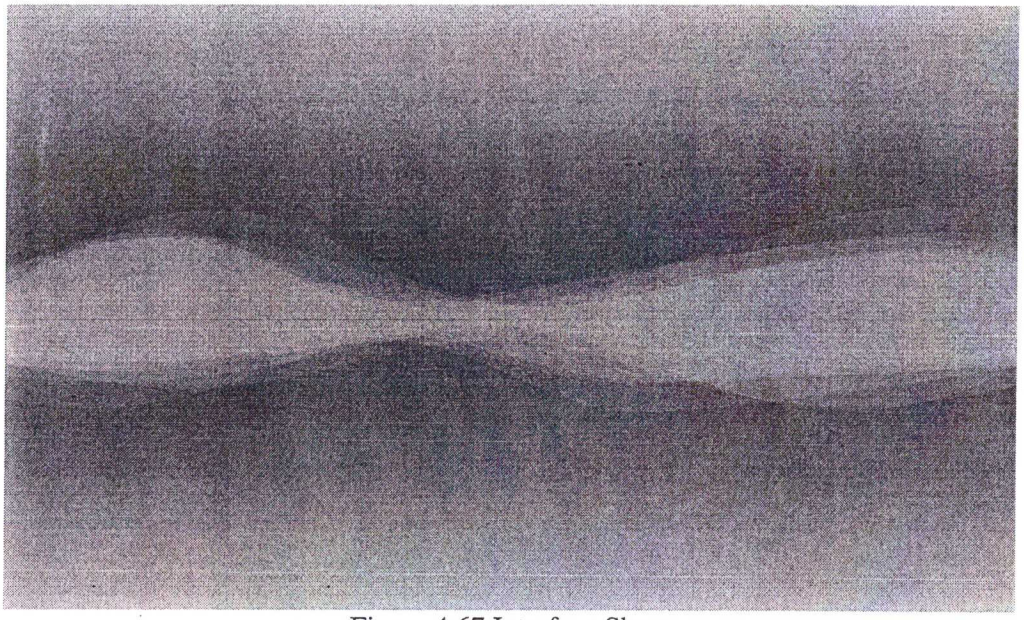


Figure 4.67 Interface Sheet

The graphics and animation are made in the XAML code included with the appendix, this is an example of some of the parts of the code shown below :

Object definition in XAML

```
<Page x:Class="HomeSystem.StyleSheet"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:mc="http://schemas.openxmlformats.org/markup-
compatibility/2006"
       xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
       mc:Ignorable="d"
        d:DesignHeight="300" d:DesignWidth="300"
       Title="Sheet" Width="1360" Height="768">
```

Gradient Code

```
<LinearGradientBrushEndPoint="0.5,1"StartPoint="0.5,0">
             <GradientStop Color="#FF92A0A9" Offset="0"/>
             <GradientStop Color="#FF0C4D76" Offset="1"/>
```

```
</LinearGradientBrush>
```

Definition for vector graphics, declared by the Path tag

```
<Path x:Name="path2" Fill="#16FFFFFF" Margin="-4.5,-16.719,-
354,0"StrokeThickness="5"
Height="154.553"Grid.Row="1"VerticalAlignment="Top">

<Path.Data>
<PathGeometry>
<PathFigureIsClosed="True"StartPoint="1364.452,46.1433588574747">

<BezierSegment Point3="691.649101190476,136.373335493092"
Point2="861.705601190476,148.136645967069"
Point1="1094.35234077381,2.52106499981046"/>
<BezierSegment Point3="262.458886904762,0.0258163642699142"
Point2="421.714099845238,1.63009396391074"
Point1="521.592601190476,124.610015019115"/>
<BezierSegment Point3="4,72.1206789041735" Point2="4,72.1206789041735"
Point1="103.203673964286,-1.57846116537092"/>
<LineSegment Point="4,154.439746221039"/>
<LineSegment Point="1364.452,154.552996225602"/>

</PathFigure>
</PathGeometry>
</Path.Data>
</Path>
```

In this code, the path's name is defined along with all the visual properties for it, such as the fill type, stroke type, and the points and segments describing its path.

4.3.2.2 Home Screen

The home screen adds components to show the main components of the system; moreover it displays images regarding the current status of the building. This is a picture of the home screen:

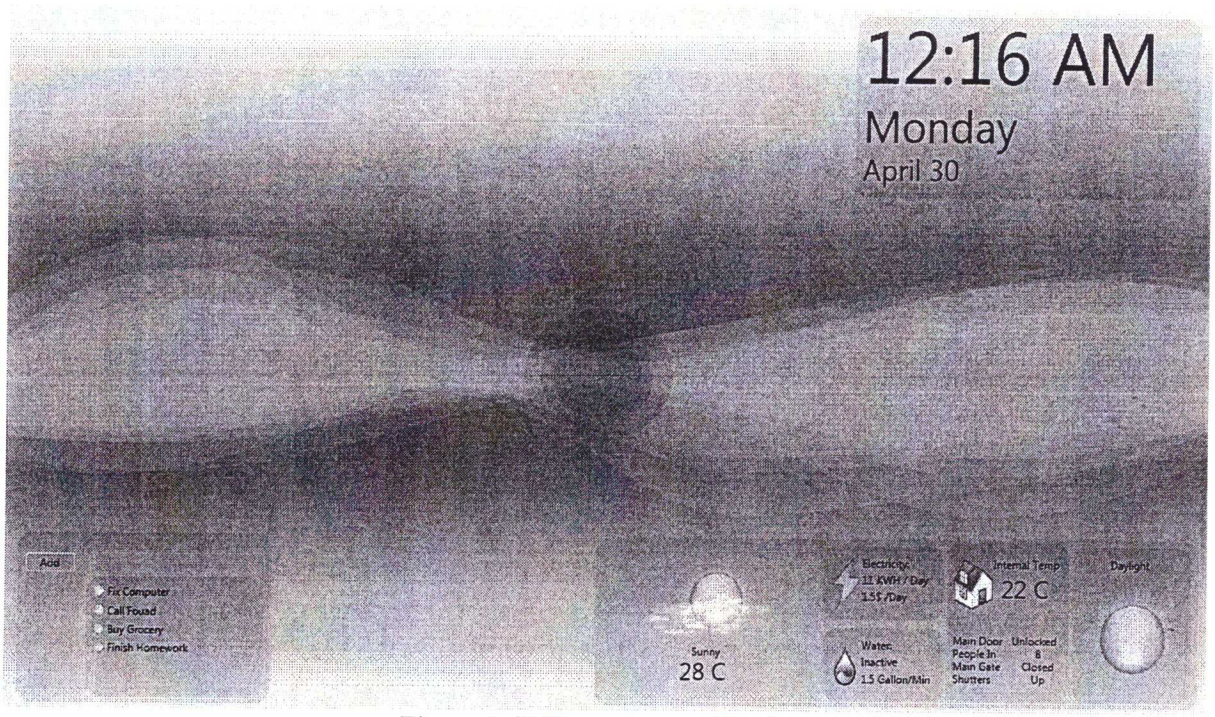


Figure 4.68 Home Screen Window

The following are code examples for some of the home screen functions:

```
publicvoidUpdateInternTemperature(int s)
  {
    Internal_Temp.Text = s.ToString() + "C";
  }

publicvoidUpdateOutsideTemperature(int s)
  {
    External_Temp.Text = s.ToString() + "C";
    Uri uri;
    if(s>20)
      uri =newUri("/HomeSystem;component/Status-weather-clear-
icon.png",UriKind.Relative);
    elseif(s>10)
      uri =newUri("/HomeSystem;component/Status-weather-few-clouds-icon.png",
                  UriKind.Relative);
    else
     uri=newUri("/HomeSystem;component/Status-weather-many-clouds-icon.png",
                UriKind.Relative);
   BitmapImageimgSource = newBitmapImage(uri);
   OutSideImage.Source = imgSource;
 }
```

This code segments updates some of the pictures and text blocks with incoming data, so it can be used to update the GUI from elsewhere.

4.3.2.3 Movies and Picture Screen

The Movies and Pictures Screen displays media content within a grid of graphical components, and since the implementation is quite similar, only the Movies screen is described and taken as an example of the picture screen.

This is a screenshot of the Movies Screen:

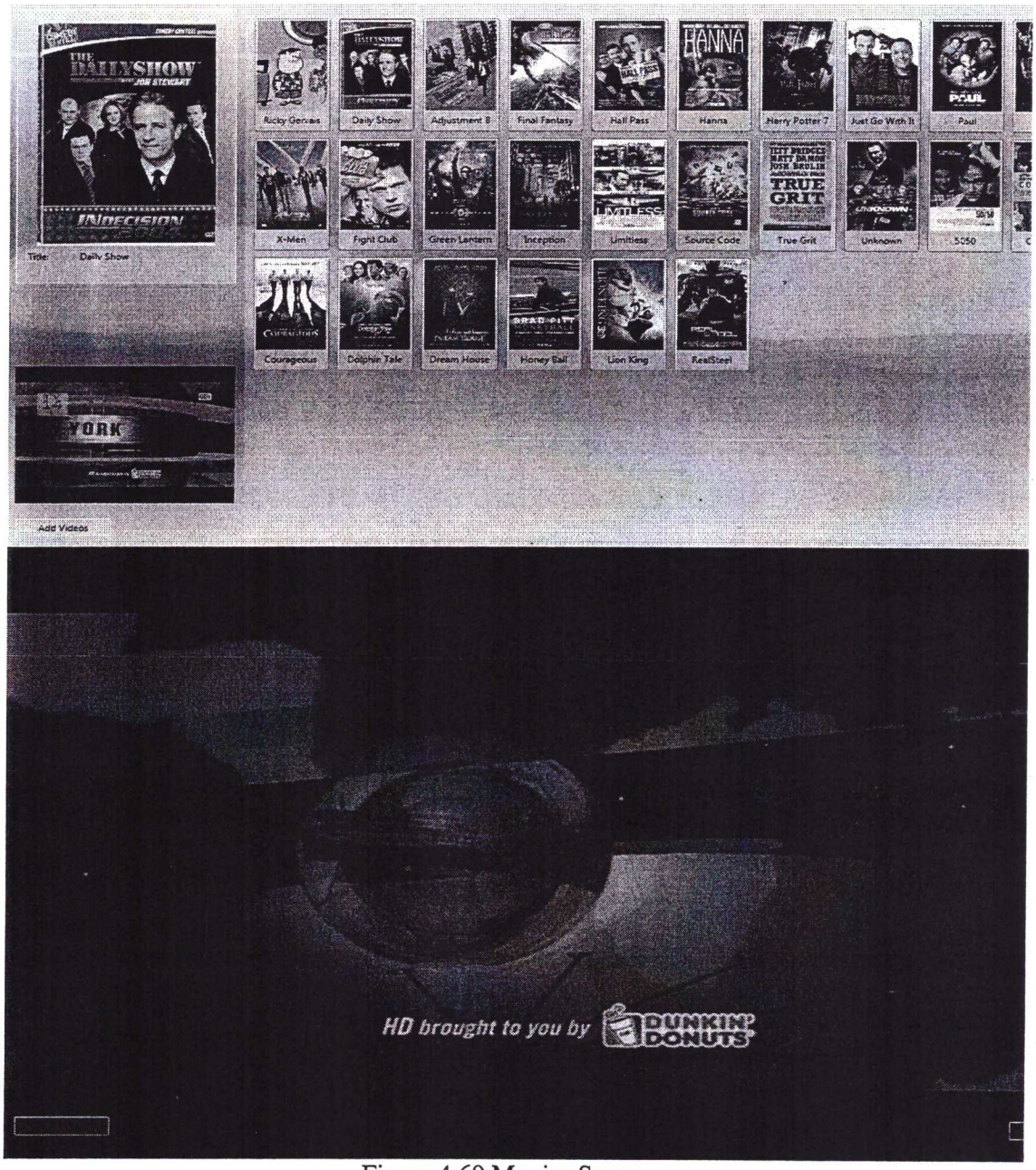


Figure 4.69 Movies Sreen

Along with the methods required to display and navigate the video, the most important methods were the ones that implemented the dynamic behavior of the GUI.

A lot of code has been put within the XAML and C# portion of this screen :

- Public void AddContent(Stringimg, String title) : this function serves as an example of the process of populating a single cell within the movie grid , so it adds a single cell into a pre-configured component described as a Warp Grid, where it initializes a new label the a movie title, then it initializes an image corresponding to the movie, having these two components; the function then initializes a button and binds it with the two components to later add it to the warp grid .

```
Public void AddContent(Stringimg, String title)
  {
   Label s = newLabel();
   s.Content = title;
   Image i = newImage();
   BitmapImagesrc = newBitmapImage();
   src.BeginInit();
   src.UriSource = newUri(img,UriKind.RelativeOrAbsolute);
   src.EndInit();
   i.Source = src;
   i.Stretch = Stretch.Uniform;
   i.StretchDirection = StretchDirection.DownOnly;
   i.Width = 86;
   i.Height = 126;
   StackPanel d = newStackPanel();
   d.HorizontalAlignment = HorizontalAlignment.Center;
   d.Children.Add(i);
   d.Children.Add(s);
   Button BS = newButton();
   SolidColorBrushscb = newSolidColorBrush(Colors.White);
   scb.Opacity = 0.3;
   BS.Background = scb;
   BS.Content = d;
   BS.Click += newRoutedEventHandler(MovieHandler);
   BS.Margin = newThickness(5);
   myWrapGrid.Children.Add(BS);
  }
```

4.3.2.4 Settings Screen

This screen provides an interface to add and configure the current settings of the system and the hardware components ; it defines the used system nodes, and connects with the hardware implemented with the system . It also defines rules and events, and other information.

The tabs visuals were used to implement the different functionalities of the system, and this is a screenshot of one of the tabs:

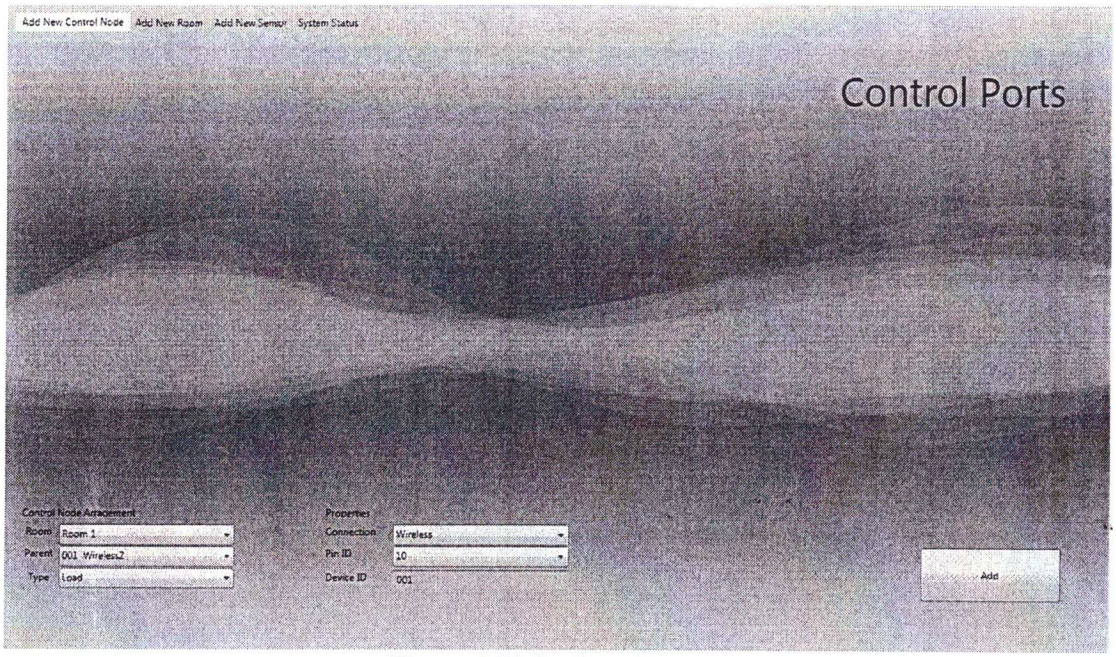


Figure 4.70 Settings Screen

All of the function mentioned in the design stage has been implemented, and this is an example on one which defines a new node for the system:

```
publicvoidAddNewControlNode(intRoomIndex, intPinID, DeviceCLASSControllerType,
ControlCLASS Type, StringDeviceID)
{
   ControlNodetNode = newControlNode(PinID, DeviceID, ControllerType, Type,
RoomIndex);
   CNodes.Add(tNode);
   Rooms[RoomIndex].Children.Add(tNode);
   notifyParentOfChange(ChangeType.Control);
}
```

Rooms and CNodes are shared lists of ControlNode objects containing information for the used control nodes, this function defines a new control node called tNode and adds it to the CNode list and Rooms list. Then finally it calls a function called notifyParentOfChange which makes sure that this data is immediately added to the database.

4.3.2.5 Data Screen

In data screen there is a graph that displays the readings for each of sensors by choosing the sensor name, the first date, and the second date needed for readings between them. this is a screen shot of Data screen in figure 4.x:

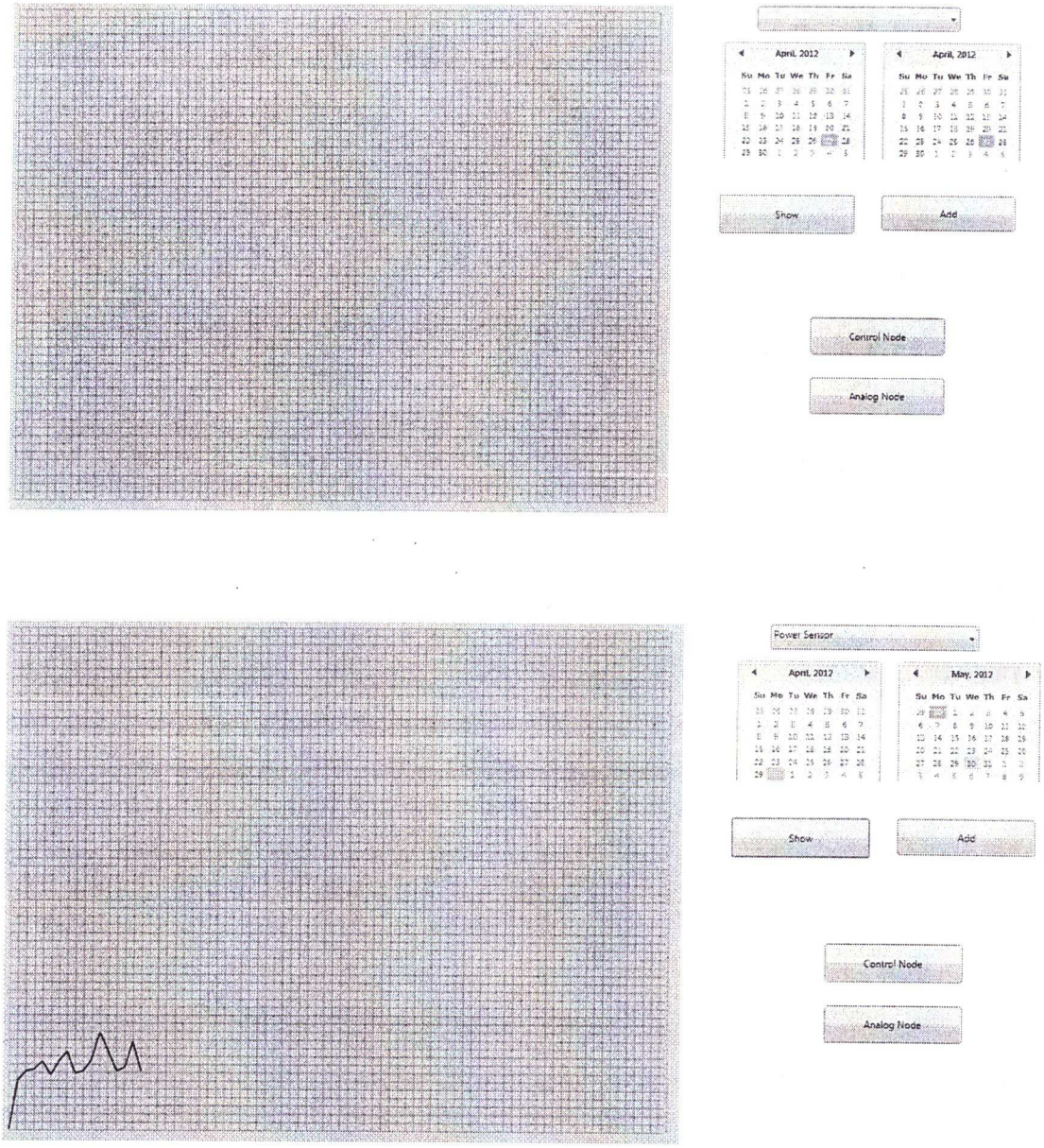


Figure 4.71 Data Screen

There is a lot of function used to take the values from the database and then draws them on the graph , all of them included in the appendix and the following functions is a part of these functions :

```
private void button3_Click(object sender, RoutedEventArgs e)
      {
        int[] HA ; // Hours Array
        int LogicalAddressNum; // Logical Address Number
        string SensorName = comboBox1.Text; // get the sensor name from combo
box
        string firstDate = calendar1.SelectedDate.ToString(); // get the first
date
        string secondDate = calendar2.SelectedDate.ToString(); // get the
second date

   switch (SensorName)
      {
    case "Power Sensor": LogicalAddressNum = 1;
                         HA
=DatabseObject.getValues(LogicalAddressNum,firstDate, secondDate);
                         Draw(HA);
                         break;
       case "CO Sensor": LogicalAddressNum = 2;
                         HA
=DatabseObject.getValues(LogicalAddressNum,firstDate, secondDate);
                         Draw(HA);
                         break;
      case "CO2 Sensor": LogicalAddressNum = 3;
                         HA
=DatabseObject.getValues(LogicalAddressNum,firstDate, secondDate);
                         Draw(HA);
                         Break;
 case "Temperature Sensor": LogicalAddressNum = 4;
                         HA
=DatabseObject.getValues(LogicalAddressNum,firstDate, secondDate);
                         Draw(HA);
                         break;
      case "Presence Sensor": LogicalAddressNum = 5;
                         HA
=DatabseObject.getValues(LogicalAddressNum,firstDate, secondDate);
                         Draw(HA)
                         break;
             }

       }

public void Draw(int[] arr)
        {
            for (int l = 0; l < arr.Length; l++)
            {
                Graph.AddToGraph(arr[l]);
            }
        }
```

The first function checks the selected sensor and then take its values from the database by its logical address and send them to the Draw function to draw them on the graph.

4.3.2.6 Control Screen

The control screen displays the control nodes according to room assignment; the following figure 4.72 is a screen shot of the Control screen:

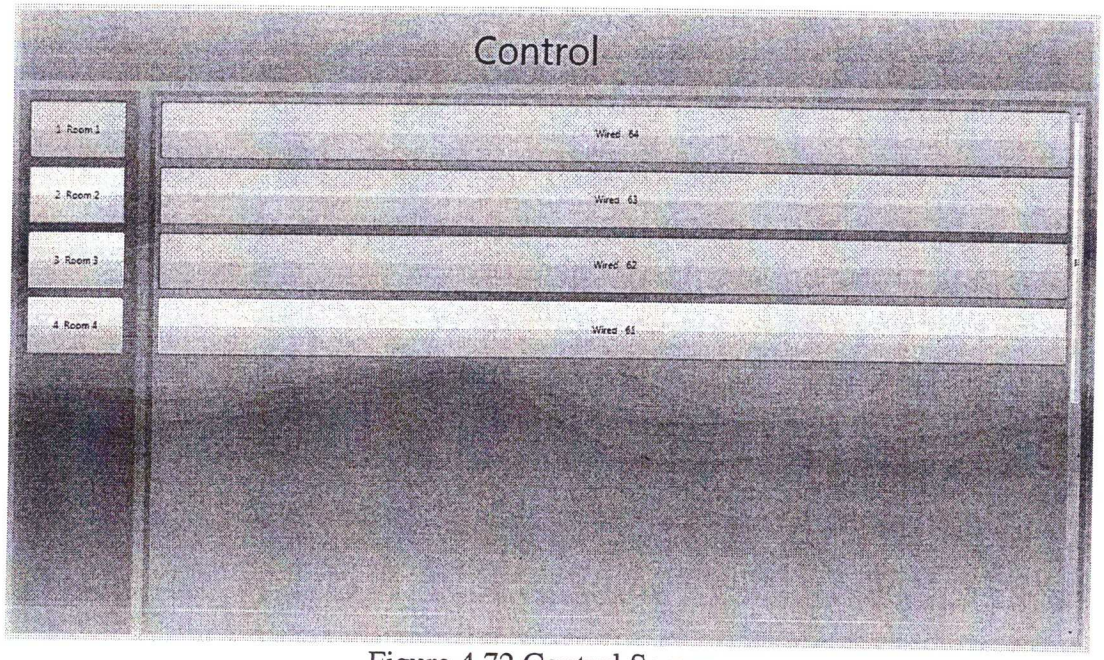


Figure 4.72 Control Screen

This function is the implementation of the event bound to all the buttons of the control screen, where the correct node is identified and toggled according to status:

```
Private void ControlAction_Click(object sender, RoutedEventArgs e)
  {
     int index = LEVEL2.Children.IndexOf((sender) asUIElement);
     ToggleButton s = (ToggleButton) sender;
     ControlNodetNode = Rooms[CurrentRoom].Children[index];

     if (tNode!=null&&tNode.DeviceType == DeviceCLASS.Wired){
       try{
          if (s.IsChecked == true){
               Controller.SetPin(tNode.PinAddress, 1);}
          else{
               Controller.SetPin(tNode.PinAddress, 0);}}
      catch { }}
    else if (tNode != null&&tNode.DeviceType == DeviceCLASS.Wireless){
         if(ZigController.connected) {
         if (s.IsChecked == true)
             {
               ZigController.SetPin(tNode.DeviceID, tNode.PinAddress, 1);}
      else{
             ZigController.SetPin(tNode.DeviceID,tNode.PinAddress, 0);}}}}
```

4.3.3 Voice Modules

In this section we have two classes, these classes work at the same time to get the best results in the system response to the voice commands.

4.3.3.1 Voice Module Class

This class is the main interface for the voice engine, and it contain the following functions to operate the voice module:

- `public void Speak(String s)`: this function repeat the entered command after the user.
- `public int GetCommandID(RecognitionResult MyWords)`: this function return the ID number for the entered command .
- `public int GetShuttersID(RecognitionResult MyWords, int Startwith)`: this function return the shutter ID number depending on the shutter status which is up or down .
- `public int GetVolumeID(RecognitionResult MyWords, int Startwith)`: this function return the volume ID number depending on the volume status which is up or down .
- `public int GetHouseID(RecognitionResult MyWords, int Startwith)`: this function return the house ID number depending on the second word which is lock or unlock .
- `public int GetFacebookID(RecognitionResult MyWords, int Startwith)`: this function return the facebook ID number .
- `public int GetTurnOnOrOff(RecognitionResult MyWords, int Startwith)`: this function return the light ID number depending on the light status which is on or off.
- `public int GetLightNum(RecognitionResult MyWords, int Startwith)`: this function return ID depending on the light number .
- `public int GetRoomNum(RecognitionResult MyWords, int Startwith`: this function return ID depending on room number.

```
public void Speak(String s)
        {
            SayThis = s;
            SpeakingThread = new Thread(DoSpeaking);
            SpeakingThread.Start();
        }

public int GetCommandID(RecognitionResult MyWords)
        {
            RecognizedWordUnit word;
            string Temp = "";
            for (int i = 0; i < MyWords.Words.Count; i++)
            {
                word = MyWords.Words[i];
                Temp = word.Text;
                switch (Temp)
                {
                    case "shutters":
                        return 10000 + GetShuttersID(MyWords, i);
                    case "volume":
                        return 20000 + GetVolumeID(MyWords, i);
                    case "house":
                        return 30000 + GetHouseID(MyWords, i);
                    case "weather":
                        return 40000;
                    case "facebook":
                        return 50000 + GetFacebookID(MyWords, i);
                    . . . .
                    case "switch":
          return 100000 + GetTurnOnOrOff(MyWords, i) + GetObjectID(MyWords, i + 1);
                }}
            return 1;}
public int GetShuttersID(RecognitionResult MyWords, int Startwith)
        {
            for (int i = Startwith; i < MyWords.Words.Count; i++)
            {
                word = MyWords.Words[i];
                Temp = word.Text;
                switch (Temp)
                {
                    case "up":return 1;
                    case "down":return 0;}}
            return 1;}
public int GetVolumeID(RecognitionResult MyWords, int Startwith)
        {
            for (int i = Startwith; i < MyWords.Words.Count; i++)
            {
                word = MyWords.Words[i];
                Temp = word.Text;
                switch (Temp)
                {
                    case "up":return 1;
                    case "down":return 0;
                    case "mute":return 2;
                }}
            return 1; }
public int GetHouseID(RecognitionResult MyWords, int Startwith)
      {
       for (int i = Startwith; i < MyWords.Words.Count; i++)
            {
                word = MyWords.Words[i];
                Temp = word.Text;
                switch (Temp)
```

```
                {
                    case "lock":return 1;
                    case "unlock":return 0;
                }}
            return 1; }

public int GetFacebookID(RecognitionResult MyWords, int Startwith)
        {
         for (int i = Startwith; i < MyWords.Words.Count; i++)
            {
                word = MyWords.Words[i];
                Temp = word.Text;
                switch (Temp)
                {
                    case "status":return 1;
                }}
            return 1; }
```

4.3.3.2 Voice Listener Class

This class contain the following functions to operate the voice module :

- `public void Listen()` : this function contain the definition for the all commands used in our system ,the grammar builder and it start the speech recognized event .
- `void recognizer_SpeechRecognized(object sender , SpeechRecognizedEventArgs e)` : this function send the command text to the voice module class to analyze it .

```
public void Listen()
        {
recognizer = new SpeechRecognitionEngine(new
System.Globalization.CultureInfo("en- US"));

            Choices UpDownCommands = new Choices("up", "down", "mute");
            Choices HouseCommands = new Choices("lock", "unlock");
            Choices OnOffCommands = new Choices("on", "off");
            Choices TvCommand = new Choices("tv");
            Choices ObjectCommands = new Choices("light");

            GrammarBuilder VolumeUpOrDown_Builder = new
GrammarBuilder("volume");
            VolumeUpOrDown_Builder.Append(UpDownCommands);

            GrammarBuilder Music_Builder = new GrammarBuilder();
            Music_Builder.Append(PlayStopCommands);
            Music_Builder.Append(MusicCommand);

            GrammarBuilder Surveillance_Builder = new GrammarBuilder();
            Surveillance_Builder.Append(Commands11);

            GrammarBuilder Facebook_Builder = new
GrammarBuilder("facebook");
```

```
            Facebook_Builder.Append(StatusCommand);

            Choices choicesOfBuilders = new Choices();
            choicesOfBuilders.Add(VolumeUpOrDown_Builder, TvOnOff_Builder,
Facebook_Builder, HouseLockOrUnlock_Builder, OnOrOff_Builder,
Surveillance_Builder, ShuttersUpOrDown_Builder, Weather_Builder,
Mode_Builder, Music_Builder);

            GrammarBuilder gb = choicesOfBuilders.ToGrammarBuilder();
            Grammar g = new Grammar(gb);
            recognizer.LoadGrammar(g);
            recognizer.SetInputToDefaultAudioDevice();
            recognizer.SpeechRecognized += new
EventHandler<SpeechRecognizedEventArgs>(recognizer_SpeechRecognized);
            RecognitionStarted = true;
          }

void recognizer_SpeechRecognized(object sender, SpeechRecognizedEventArgs
e)
        {
            TextGeneratedFromSpeech(e.Result, this.v);
        }
```

It should be noted that there are also other functions that rebuild the voice engine, and add new commands to it, that are exhibited in the Code Appendix of this document.

4.3.4 Data Modules

The following code components represent the data modules within the system:

4.3.1.1 Shared Data Structure

All data structures specified in the Software design were implemented in the code, and instantiated within lists within a single object of type "DataRepositry", as it contains all the necessary data, and function handlers required for data movement between objects.

The following are examples of the UML design implementation:

```
public classSensorNode
    {
public String Name;
public intPinNumber;
public intHardwareNumber;

  public SensorNode(int pin,int HardwareNumber, String Name)
      {
       this.Name = Name;
       PinNumber = pin;
       this.HardwareNumber = HardwareNumber;
      }
    }
.....
```

These are instantiated within the Data Repository in lists or objects to handle other objects requests; these are example of those lists:

```
public List <ControlNode> CNodes = new List<ControlNode>();
public List <SensorNode> ANodes = new List<SensorNode>();
public List <Room> Rooms = new List<Room>();
```

4.3.1.2 Database and Database Interface

In this system the database tables are built through the visual studio program and the WPF forms developed in C#.Net, these forms are linked to the database using SQL Connections and commands, and the main objects and their methods used to make this connection are listed in the table below :

Table 4.4 Connection Object

Object Name	Attributes	Methods
SqlConnection	ConnectionString	Open() Close()
SqlDataAdapter	Select	Fill() Update()
DataSet	Tables[]	AcceptChanges()

The database is divided to several parts each part has its own tables, connections, forms and functions, as listed below:

1. Sensors Data: this section contains a group of tables associated with each other to do an important operation to enable the user to show a graph explain to it the readings of each sensor. these operations based on the following functions:
    - `public void connect()` : this function initialize and start a connection to the database and start a timer to take values from the sensors every minute .
    - `public void AddRecord()` : this function is used to take value from each sensor every minute and then store it in Minutes table in the database ,and after 60 minutes which means 60 value for each sensor take the average value for each sensor and send it to Hours Table and delete all the values from the Minutes table.
    - `public void AddToHoursTable(string count, int logicaladdress)` : this function is used to add the value from the previous step in Hours Table ,this step repeated every hour . so after 24 hour which means

each sensor has a 24 value , take the average value for each sensor and send it to Days table and delete all the values from Hours table .

- `public void AddToDayTable(string count, int logicaladdress)` : this function is used to add the value from the previous step in Days table , this step repeated every day . so after 30 day which means each sensor has a 30 value take the average value for each sensor and send it to the Months table .
- `public void AddToMonthsTable(string count, int logicaladdress)` : this function is used to add the value from the previous step in Months table this step repeated every month . so after 12 month which means each sensor has a 12 value take the average value for each sensor and send it to the Years table .
- `public void AddToYearTable(string count, int logicaladdress )` : this function is used to add the value from the previous step in Years table this step repeated every Year .

```
public void connect()
{
  ConnectToDatabase.ConnectionString=@"DataSource                    =.\SQLEXPRESS;
AttachDbFilename=C:\

Users\user\Documents\VisualStudio2010\Projects\

TestApplication\TestApplication\Database1.mdf;
                                        Integratedsecurity=True       ;       User
Instance=True";
   ConnectToDatabase.Open();
}

public void AddRecordToMinuteTable()
  {
    if (number <= 60) // number is the number of minutes values for each sensor
        {
           DR2[0] = logicaladdress
           DR2[1] = CurrentTime;
           DR2[2] = random.Next(0, 255);
        } else if(number >= 60)
            {
              object s = dataset.Tables["mins"].Compute("SUM(Value)", "
                        LogicalAddress ='" + logicaladdress + "'");
              for (int i = 0; i < number; i++)
                 {
                 if (dataset.Tables["mins"].Rows[i]["LogicalAddress"].ToString()
== d)
                   {
                      dataset.Tables["mins"].Rows[i].Delete();
                   }}
                    AddToHoursTable(avg, logicaladdress);
                  }}
```

```
public void AddToHoursTable(string avg , int logicaladdress)
        {
        if (number <= 24) // number is the number of hours values for each
sensor
           {
            DR2[0] = logicaladdress ;
            DR2[1] = CurrentTime; ;
            DR2[2] = avg;
                . . . .
           } else if (number > 24)
            {
              object s = dataset.Tables["hrs"].Compute("SUM(Value)", "
LogicalAddress
                          = '" + logicaladdress + "'");
             for (int i = 0; i < number; i++)
               {
               if (dataset.Tables["hrs"].Rows[i]["LogicalAddress"].ToString()
== d)
                 {
                   dataset.Tables["hrs"].Rows[i].Delete();
                  }}
                   AddToDayTable(avg, logicaladdress);
                }}


public void AddToDayTable(string avg , int logicaladdress)
     {
      if (number <= 30) // number is the number of days values for each sensor
         {
            DR2[0] = logicaladdress ;
            DR2[1] = CurrentTime; ;
            DR2[2] = avg;
         }
      else if (number > 30)
            {
              object s = dataset.Tables["days"].Compute("SUM(Value)", "
                        LogicalAddress = '" +  logicaladdress + "'");
              for (int i = 0; i < number; i++)
                {
                if (dataset.Tables["days"].Rows[i]["LogicalAddress"].ToString()
== d)
                  {
                     dataset.Tables["days"].Rows[i].Delete(); }}
                     AddToMonthsTable(avg, logicaladdress); } }


public void AddToMonthsTable(string count , int logicaladdress)
   {
    if (number <= 12) // number is the number of month values for each sensor
         {
            DR2[0] = logicaladdress ;
            DR2[1] = CurrentTime; ;
            DR2[2] = avg;
         }
    else if (number > 12)
           {
           object s = dataset.Tables["months"].Compute("SUM(Value)", "
LogicalAddress
                              ='" +  logicaladdress + "'");
          for (int i = 0; i < number; i++)
              {
             if (dataset.Tables["months"].Rows[i]["LogicalAddress"].ToString()
==d)
                 {
                   dataset.Tables["months"].Rows[i].Delete();}}
```

```
                AddToYearTable(avg, logicaladdress);
               }}

public void AddToYearTable(string count, int logicaladdress)
           {
            DR2[0] = logicaladdress;
            DR2[1] = CurrentTime; ;
            DR2[2] = count;
            }
```

2. **Control and Analog Ports**: this section contain three tables(AnalogPort Table, AnalogPortValue Table and Control Port Table). To deal with these tables we make a connection to each table and functions to make a set of operations on them.

   - **Control Port :** we have the following functions to deal with Control Port Table :

     - `public void ConnectToControlTable()` : this function initialize and start a connection to the ControlNode Table in the database .

     - `public void AddToControlNode(string additionalID, bool state,. . .)` : this function is used to add a new control port to ControlNode Table .

```
public void ConnectToControlTable()
        {
            try
            {
             ConnectToControlNode.ConnectionString=@"Data Source=.\
SQLEXPRESS;
             AttachDbFilename=C:\Users\user\Documents\Visual Studio …." ;
             ConnectToControlNode.Open();
             command.CommandText = "SELECT * FROM ControlNodeTable";
             DataAdapter = new SqlDataAdapter(command.CommandText,
             ConnectToControlNode);
             DataAdapter.Fill(dataset, "CN");
            } catch
            {
                MessageBox.Show("Can not access the Control Node table
??");
            }}


public void AddToControlNode(string additionalID, bool state, . . . . )
     {
       try
        {
```

```
        SqlCommandBuilder CommandBuilder = new
SqlCommandBuilder(DataAdapter);
        rt = dataset.Tables["CN"].Select("PinID = '" + pinID + "' ");
        int count = rt.Count();
        if (count == 0)
           {
              DataRow DR2 = dataset.Tables["CN"].NewRow();
              DR2[0] = pinID;
              DR2[1] = logicalAddress;
              . . . .
              DataAdapter.Update(dataset, "CN");
                } else
                { MessageBox.Show("Pin ID Exist ??"); }}
            catch (SqlException ex)
            { MessageBox.Show(ex.ToString());}}
```

- **Analog Port And Analog Port Values :** for analog ports we have two tables for them the first is AnalogNode table that store each analog node in our system , the second is AnalogNodeValues that store the values for each analog node . so to deals with these tables we declare the following functions :
  - `public void connctToAnalogTable()` : this function is initialize and start a connection to AnalogNode Table.
  - `public void AddToAnalogNode(int PinID, int LogicalAddress, int Level, int Class, int HWClass)` : this function is used to add a new analog port to AnalogNode Table .
  - `public void ConnectToAnalogValueTable()` : this function is initialize and start a connection to AnalogNodeValue Table.
  - `public void AddToAnalogValuesTable(int PinID, int Value)` : this function is used to add a new value to AnalogNodeValue Table for some analog node depending on its PinID.

```
public void connctToAnalogTable()
    {
     try
      {
       ConnectToAnaloglNode.ConnectionString=@"Data Source=.\SQLEXPRESS; AttachDbFilename
                                              =C:\Users\user\Documents\Visual Studio. .";
       ConnectToAnaloglNode.Open();
       command.CommandText = "SELECT * FROM AnalogNodeTable";
       DataAdapter.Fill(dataset, "AN");
      } catch
           {
```

```
                MessageBox.Show("Can not access the Analog node table ??");
            }}


public void AddToAnalogNode(int PinID, int LogicalAddress, int Level, int Class, int
HWClass)
    {
      try
       {
          SqlCommandBuilder CommandBuilder = new SqlCommandBuilder(DataAdapter);
          DR2[0] = PinID;
          DR2[1] = LogicalAddress;
          DR2[2] = Level;
          DR2[3] = Class;
          DR2[4] = HWClass;
          dataset.Tables["AN"].Rows.Add(DR2);
          DataAdapter.Update(dataset, "AN");
        } catch (SqlException ex)
              {
                  MessageBox.Show(ex.ToString());
              }
              finally
              {
                  ConnectToAnaloglNode.Close();
              }}

public void ConnectToAnalogValueTable()
   {
     try
      {
      ConnectToAnalogValue.ConnectionString=@"Data Source=.\SQLEXPRESS; AttachDbFilename
                                          =C:\Users\user\Documents\Visual Studio. . .";
      ConnectToAnalogValue.Open();
      command.CommandText = "SELECT * FROM AnalogNodeValues";
      DataAdapter = new SqlDataAdapter(command.CommandText, ConnectToAnalogValue);
      DataAdapter.Fill(dataset, "ANV");
            } catch
            {
                MessageBox.Show("Can not access the Analog Values table ??");
            }}

public void AddToAnalogValuesTable(int PinID, int Value)
     {
       try
        {
         SqlCommandBuilder CommandBuilder = new SqlCommandBuilder(DataAdapter);
         DR2[0] = PinID;
         DR2[1] = DateTime.Now;
         DR2[2] = Value;
         DataAdapter.Update(dataset, "ANV");
        } catch
            {
                MessageBox.Show("Foreign Key Constraint : the Pin ID does not exist !");
            } finally
            {
                ConnectToAnalogValue.Close();
            }}

public void AddToAnalogValuesTable(int PinID, int Value)
   {
     try
      {
       SqlCommandBuilder CommandBuilder = new SqlCommandBuilder(DataAdapter);
       DR2[0] = PinID;
       DR2[1] = DateTime.Now;
       DR2[2] = Value;
```

```
        DataAdapter.Update(dataset, "ANV");
      } catch
            {
                MessageBox.Show("Foreign Key Constraint : the Pin ID does not exist !");
            }
            finally
            {
                ConnectToAnalogValue.Close();}}
```

4.3.5 Main Logic

The main logic for this program is implemented in the main window of this WPF application, and it contains all of the objects required for the system functions. Moreover, it contains implementation of various system test points and events, such as the key down event, the voice recognition event, the SMS received event, and other various events for the implementation of the software.

The following is an example of the system's data composition and declarations:

```
Public MainWindow()
{
#region Data Initialization
DataRepo = newDataRepositry();
#endregion
#regionVoiceEngine Code Binding
DataRepo.VoiceEngine.TextGeneratedFromSpeech +=
newVoice_Module.InformTextRecognized(GotText);
DataRepo.VoiceEngine.IDGeneratedFromSpeech +=
newVoice_Module.InformIDRecognized(GotId);
StartEngine();
#endregion
this.Closing +=
newSystem.ComponentModel.CancelEventHandler(MainWindow_Closing);
HomeScreen = newHome();
MoviesScreen = newMovies(DataRepo);
DataScreen = newDataRep(DataRepo);
ControlScreen = newControl(DataRepo);
DataRepo.BuildList = ControlScreen.BuildLists;
DataRepo.BuildList();
PicturesScreen = newPictures();
SettingsScreen = newSettings(DataRepo);
...
}
```

This component also includes implementation of the system's events, most important of which are the following:

- `Private void Window_KeyDown(object sender, KeyEventArgs e) :`

This function is called when a button on the keyboard is pressed, it identifies it and execute the corresponding command for it.

- `public void GotText(String TextSaid, EventArgs e)`

  `public void GotID(intID, EventArgs e)` : These two functions are called when a voice is recognized and a text is presented and passed to these two functions so they can be handled as commands.

- `Public void TextMessageReceived(String Message, EventArgs e)` :This function is called to notify a new incoming message, and it is passed a string containing a the message body.

There are more functions that are included and well commented in the appendix of this document; the following is an example, showing the GotText event handler implementation:

```
Public void GotText(StringTextSaid, EventArgs e)
    {
      DataRepo.VoiceEngine.Speak(TextSaid);
      this.Dispatcher.Invoke(
      System.Windows.Threading.DispatcherPriority.Normal,
      newAction(delegate()
                {
      switch(TextSaid)
          {
          case"home": ExecuteCommand("H");
                      break;
          case"movies": ExecuteCommand("M");
                      break;
          case"pictures": ExecuteCommand("P");
                       break;
     ....
   }
```

The main logic also includes a timed function responsible for updating the clock of the system and it calls the methods responsible for updating the lists and tables of the system, as shown in the following code segment:

```
private void SynchronousCycle(Object state)
    {
      ...
     if (!this.Dispatcher.CheckAccess())
       {
        this.Dispatcher.Invoke(
```

```
System.Windows.Threading.DispatcherPriority.Normal,newAction(delegate()
          {
           HomeScreen.TimeText.Text = DateTime.Now.ToShortTimeString();
           HomeScreen.UpdateInternTemperature(A++);
           HomeScreen.UpdateOutsideTemperature(2);
           HomeScreen.DateText.Text =
           ConvertMonth(DateTime.Now.ToShortDateString().Substring(3,3))+" " +
          DateTime.Now.ToShortDateString().Substring(0, 2);
          DateTemp = DateTime.Now.ToLongDateString();
          DateTemp = DateTemp.Substring(0, DateTemp.IndexOf(','));
          HomeScreen.DayText.Text = DateTemp;
         }));  }
...
       }
```

Other functions are implemented in the main window that handle the graphical switching mechanism and graphics within the software graphical user interface, and they are well commented in this document's appendix.

4.3.6 Event System

The event system implementation closely follows the design criteria specified in chapter 3, where a set of system objects and enumerations guide a path for a list of objects so they can execute certain functions according to certain values.

Events in the system are a list of objects saved in a list in the DataRepositry object, and the following are code segments showing the functionality of the event system within the system code:

- Event System Classes and Definitions:

All classes specified in the UML diagram have been implemented, and this is an example:

```
public enum CConditionType
    {
        On,Off,More,Less,Has,Equal,Pressed
    }
    public enum CEventType
    {
        Control,Senor,Text,Key
    }
    public class CEventControlPort : CEvent
    {
        public DeviceCLASS DevClass;
        public String DevID;
        public int State;
        public int Index;
    }
    public class CEventSensorPort : CEvent
```

```
    {
        public DeviceCLASS DevClass;
        public String DevID;
        public int State;
        public int Value;
    }
```

- Actions

Actions are all abstracted into a single function called Execute, where it identifies the type of action and acts on it, this function segment shows its functionality:

```
public void Execute()
        {
            if(Type==null) return;
            else if (Type == ActionType.Display)
                MessageBox.Show(Text);
            else if (Type == ActionType.SMS)
            {
                MyRepo.MySMSModule.SendSMS(Text2, Text);
            }
            else if (Type == ActionType.Speak)
            {
                MyRepo.VoiceEngine.Speak(Text);
            }
            else if (Type == ActionType.Toggle)
            {
                MyRepo.SetControlNode(DevClass, DevID, Index, State);
            }
        }
```

- Test Points

The events are checked in the system at predefined points, where certain changes occur to the system. First for every test point, events corresponding to the type of test point are picked, then the condition is checked, and if it is true, the associated action is executed.

The following are the Test Points in the system, which are represented by function calls that could possibly change the state of the system:

- `HomeSystem.SMSModule.SMSReceived`: Test Point for SMS's received on the system, characterized by event objects of type `CEventSMSRecieved`.
- `HomeSystem.Server.DataReceived`: Test Point for text received on the system, characterized by event objects of type `CEventTextRecieved`
- `HomeSystem.MainWindow.GotText`: Test Point for voice recognition of user defined words, characterized by event objects of type `CEventVoiceRecognized`.

- HomeSystem.MainWindow.Window_KeyDown: Test Point for keys pressed on the remote control or keyboard, characterized by event objects of type CEventKeyPressed.
- HomeSystem.DataModule.AnalogUpdated: Test Point for analog input values should they change CEventSensorPort.
- HomeSystem.DataModule.DigitalUpdated: Test Point for digital values should they change either as an output or an input CEventControlPort.

The following is an example of a Test Point for a key pressed:

```
private void Window_KeyDown(object sender, KeyEventArgs e)
{
      //System Keys
      If (MainEventHandling == true)
      {
         if (Startup)
         {
           Startup = false;
           Switch = HomeScreen.FindResource("StartupEnd") as Storyboard;
           Switch.Begin();
         }
         ExecuteCommand(e.Key.ToString());

      //Check for Key Events
         for (int i = 0; i < EventLists; i < EventLists.Count)
         {
             if (EventLists[i].Type == CEventType.Key)
                 EventLists[i].Action.Execute();
         }
      }
}
```

Chapter Five

System Testing and Deployment

5.1.1 Wired Controller
5.1.2 Wireless Controller
5.1.3 Testing serial communication
5.1.4 Transducers and Control Nodes
5.1.5 Hardware System Assembly

5.1 Wired Controller

Despite the availability of PCB Blueprints for the main hardware controller, it was not implemented due to the lack of an available and operational PCB Printing Machine. And so, it was to be hand wired and soldered using prototyping circuit boards which allowed the making of all the necessary components for the main controller.

The testing started with the implementation of the controller on bread boards, where outputs and inputs were put under a test and the system's behavior was observed. The following is a picture of the bread board half way done, with the all the necessary components:

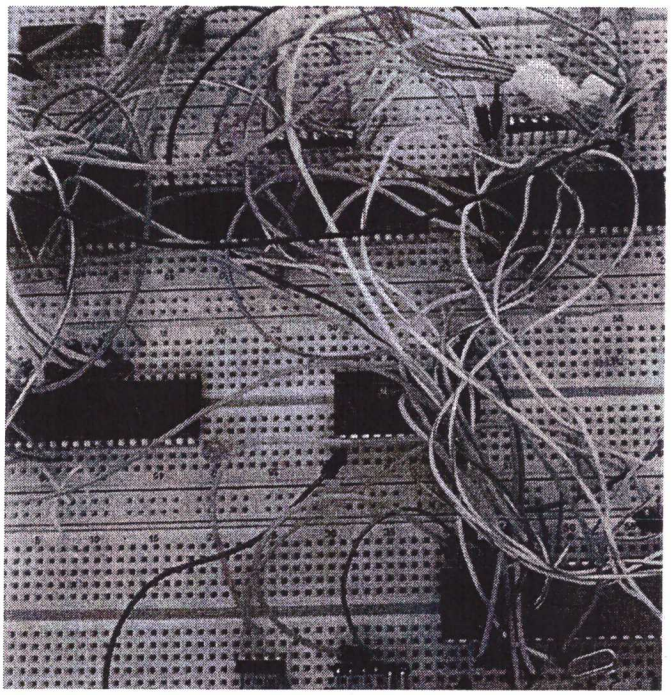


Figure 5.1 Controller Bread Board

After the basic functionality tests, the controller was implemented on soldering prototyping boards, and the following boards were the results of that implementation:

Wired controller core: This board is the core of the wired controller, where 4 PPI's and an 8255a are connected and interfaced for the other boards, Figure 5.2 shows the prototype half way soldered.

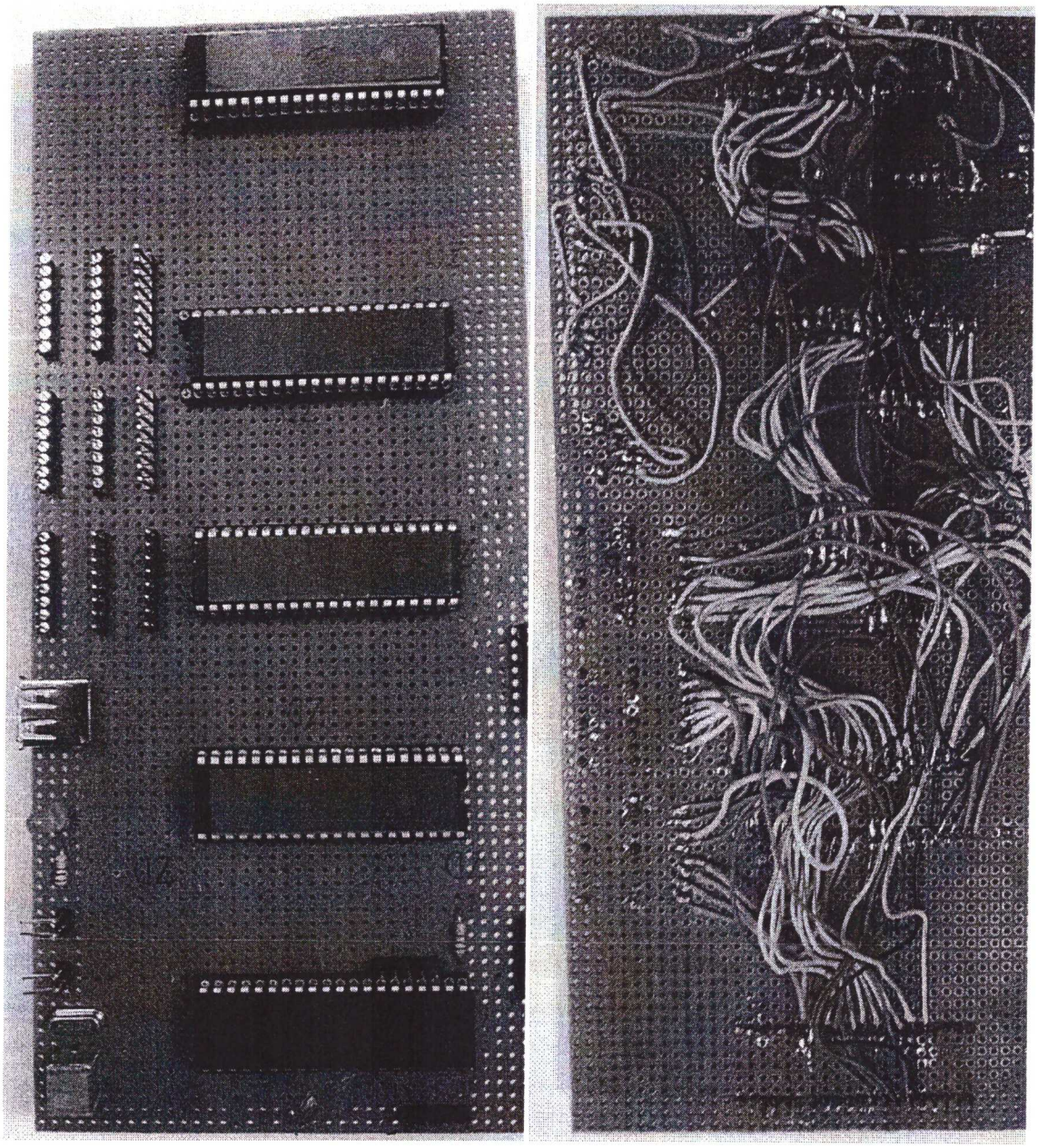


Figure 5.2 Wired Controller Core

Analog Input board: This is the implementation for the analog input circuit as shown in figure 5.3.

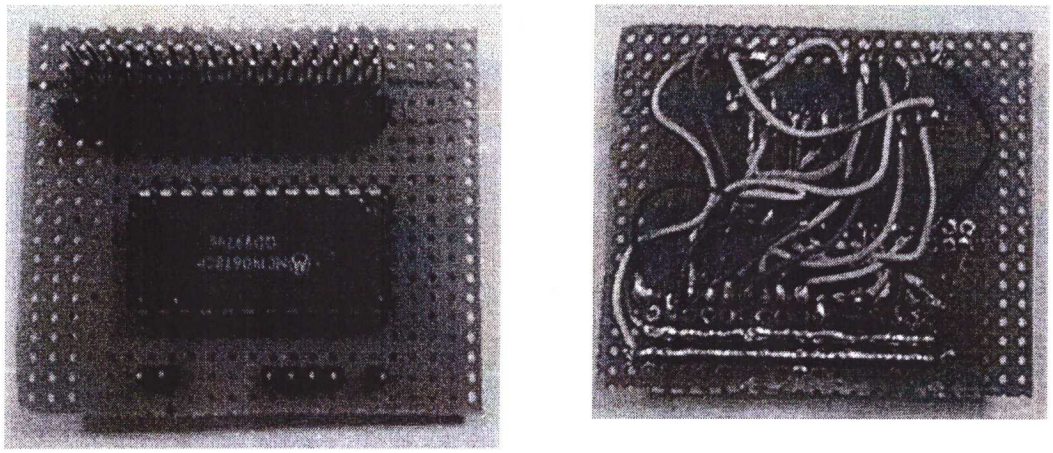


Figure 5.3 Analog Input Circuit

Digital Input board: This is the implementation for the digital input circuit shown in Figure 5.4, it inputs 0, otherwise it inputs 1.

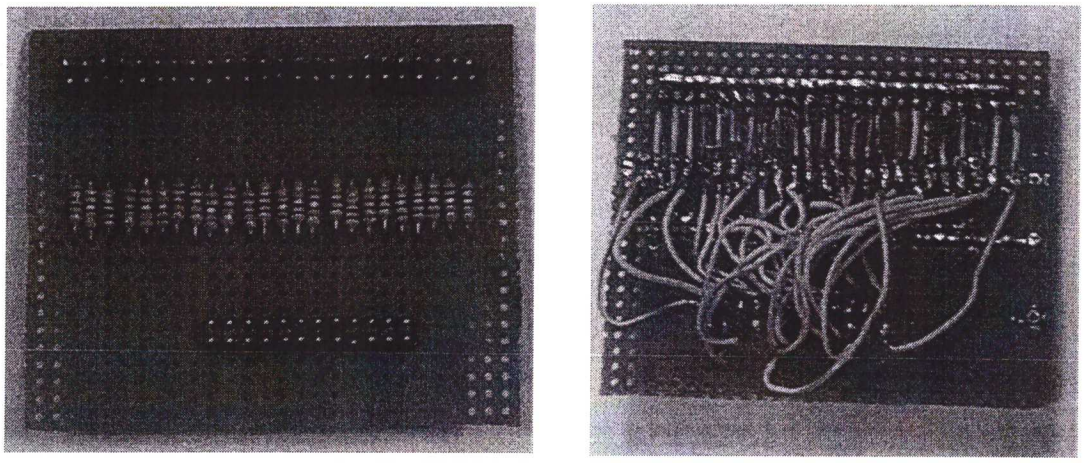


Figure 5.4 Digital Input Board

Output Circuit : Figure 5.5 Shows the Output circuit implemenation, where only one 8255a is soldered. This circuit is duplicated 8 times to cover 64 outputs.

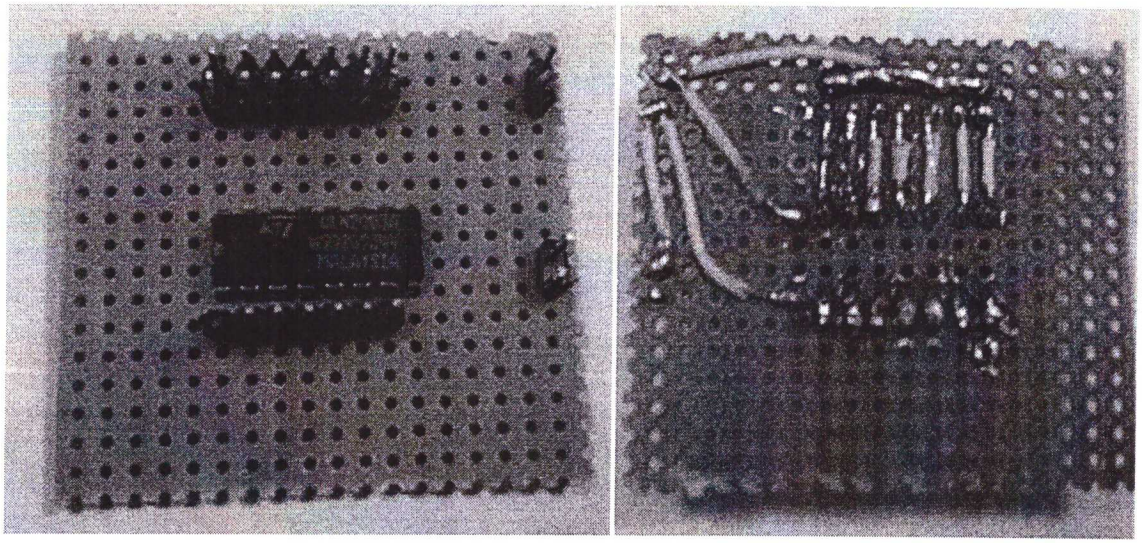


Figure 5.5 Output Board

All of these were connected to the main controller using their respective pins, so they can be interfaced to the computer via the USB port, and tested using the prepared hardware testing GUI.

After the physical implementation of the hardware, tests were made to confirm whether the output is as expected or not, and to test other criteria such as response time, and hardware stability, and so The following interface was used to test the hardware functionality:

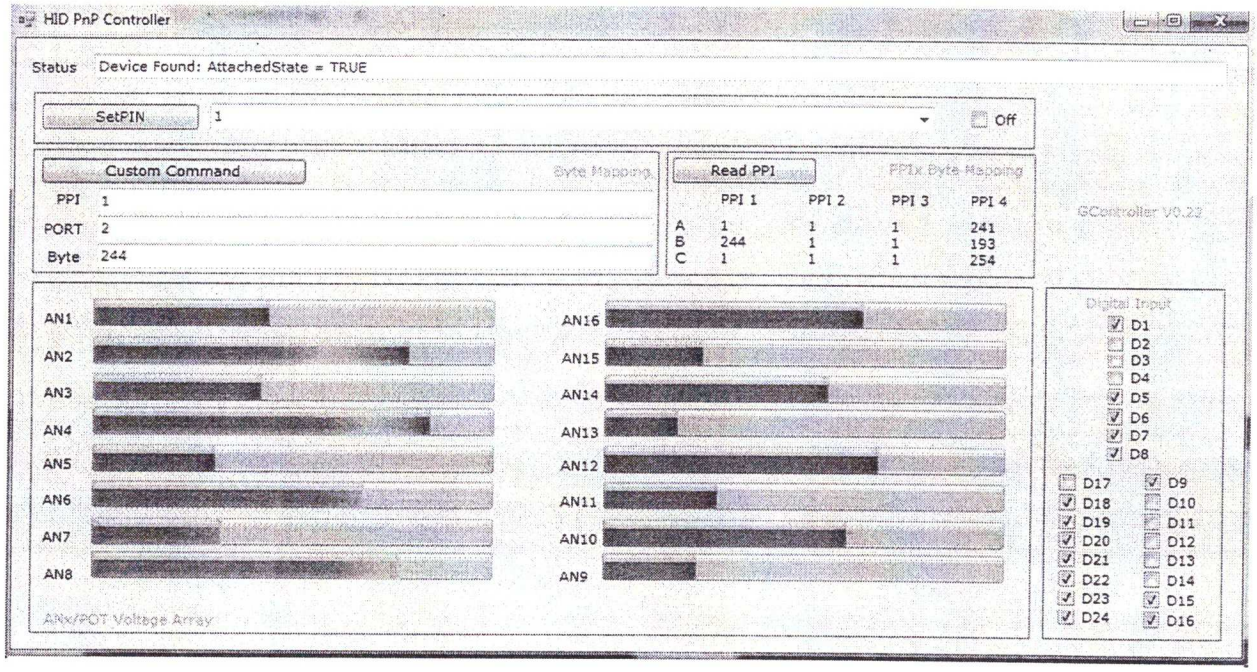


Figure 5.6 Wired Controller Testing GUI

The following tests were executed on the main controller:

- Device Output:

Action	Results
SetPin(X,1) :X{1,72}	With X representing a digital pin number, all corresponding pins were tested and effectively triggered to state 1
SetPin(X,0): X{1,72}	With X representing a digital pin number, all corresponding pins were tested and effectively triggered to state 0
getAnalog(Y): Y{1,16}	Y represents an analog channel, they were tested using 3 DC signals to test the validity of ADC module in the controller. The signals were a 5V signal, a 3.3V signal and a 0V signal. And the it outputted 1023, 674, and 0 which are the correct corresponding 10bits values.
ReadPPIArray()	16 8-bits values are read when this function is called representing the status of the 4 ports of the 4 PPI Values. The values reflected the correct status of the registers for all the ports.

Testing was done using a LED Array, a Potentionmeter, and a DIP switch, there are no software errors found, and any percentage of error will be due to physical failure or general hardware failure.

- Response Time

Tests were made using a loop to calculate the time between each consecutive hardware request, using the following function, where time was calculated between each function call.

Each function call sends 1000 64-bytes packet and awaits for response with a 64-bytes packet, the time is then calculated as an average time for a single two way packet send and receive; The following is the output for that function:

Test Iteration	Results	Test Iteration	Results
1	3.075 ms	5	3.00 ms
2	2.98 ms	6	2.97 ms
3	2.99 ms	7	3.075 ms
4	3.07 ms	8	2.98 ms

The tests proved a very fast response time, meaning that it takes an average of 3ms to trigger ports and read their status.

Even though this represents the speed of the controller working on the USB Bus, it should be mentioned that it is not always the case, and it is somewhat affected by type of the computer running it.

5.1.1 Wireless Controller Testing

Regarding to XBee radios testing, it has been build single-hop and multi-hop XBee networks with several XBee wireless.

Communication with XBee modules is done either via Arduino or via a USB dongle which is connected to a computer.

5.1.1.1.1 RSSI Measurement

RSSI (Received Signal Strength Indicator) is the signal level (in -dBm) of last good packet received. API commands were used to read the value of RSSI.

The RSSI value reported for standard XBee module is between -23 to -92 dBm. However, the XBee manual says that the reported value is accurate between -40 dBm and the sensitivity of Xbee module's receiver .

5.1.1.1.2 RSSI Measurement in an open area

We connect two XBee radio (Coordinator and Router) and then vary the distance between them to measure the relationship between RSSI values and distances. We make RSSI measurements in line-of-sight (LOS) and non-line-of-sight (NLOS) settings in a large roof area.

For the LOS test we put the two radios in two places on the roof with a direct view to each other and vary the separation distance between them by moving the coordinator away from the router keeping the direct communication between them.

With regards to the NLOS setting the two nodes has put in roof area which has some obstructions from near objects such metals, cars, bricks. The results of the RSSI measurement at various distances are shown in Figure below.

In Figure 5.7, the x-axis is the distance measured in meters and y-axis is the RSSI value measured in dBm. RSSI values for LOS and NLOS setting decrease with the distance as expected and the NLOS RSSI values are much lower than the LOS values due to obstructions.

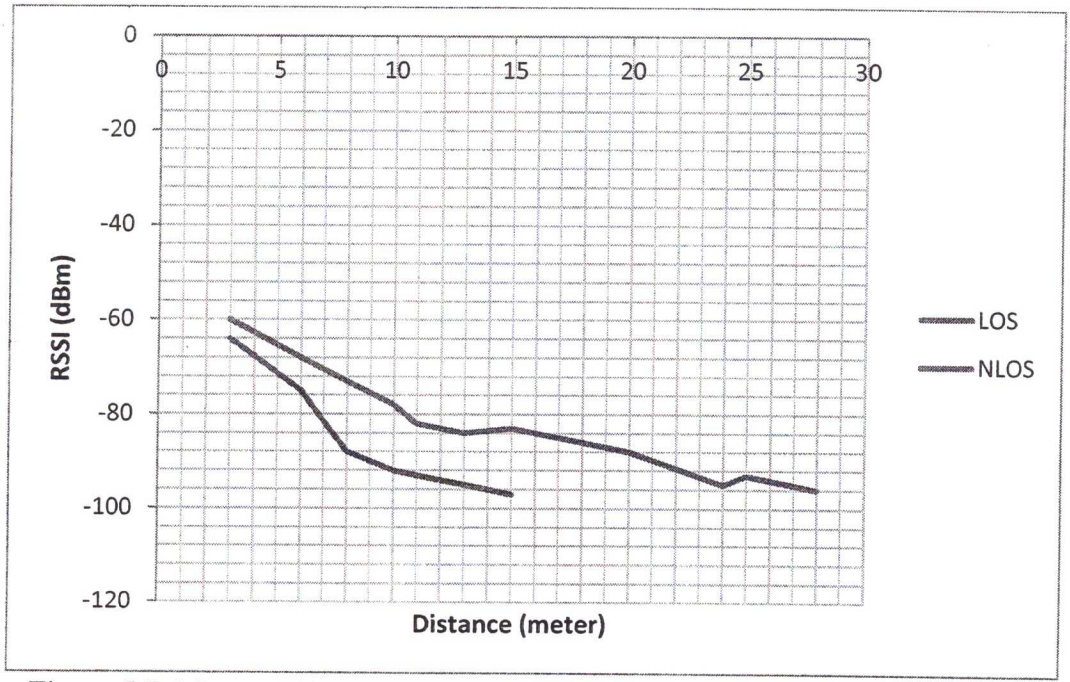


Figure 5.7: Measured RSSI values versus distance, in LOS and NLOS settings.

5.1.1.1.3 RSSI Measurement in a Building

A standard residential building was the other location in which RSSI measurements were made. Performing the RSSI measurements in a NLOS (non-line of sight) setting, where the XBee coordinator sets to fix position in one room and move the router device node away from the coordinator into the room next to the coordinator's room and measured the RSSI.

The RSSI measurement result is shown in figure below. From the figure, we can see that the RSSI decreases rapidly with distance for the first three rooms but slightly decreases in the fourth and fifth room. The RSSI decays almost linearly with distance.

The x-axis is the distance measured in meters and y-axis is the RSSI value measured in dBm.

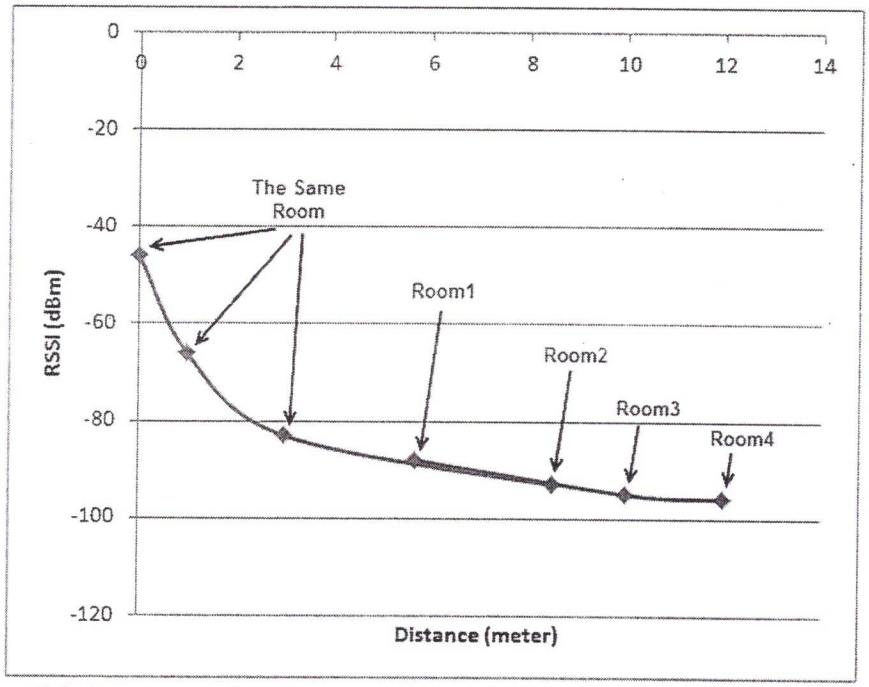


Figure 5.8: Measured RSSI versus distance from the transmitting node to the receiving node located in five rooms of a building.

5.1.1.1.4 Second floor testing

In this test, the router set at the first floor and the coordinator at the second floor, as shown in the figure below the RSSI value decreases due to distance and obstructions.

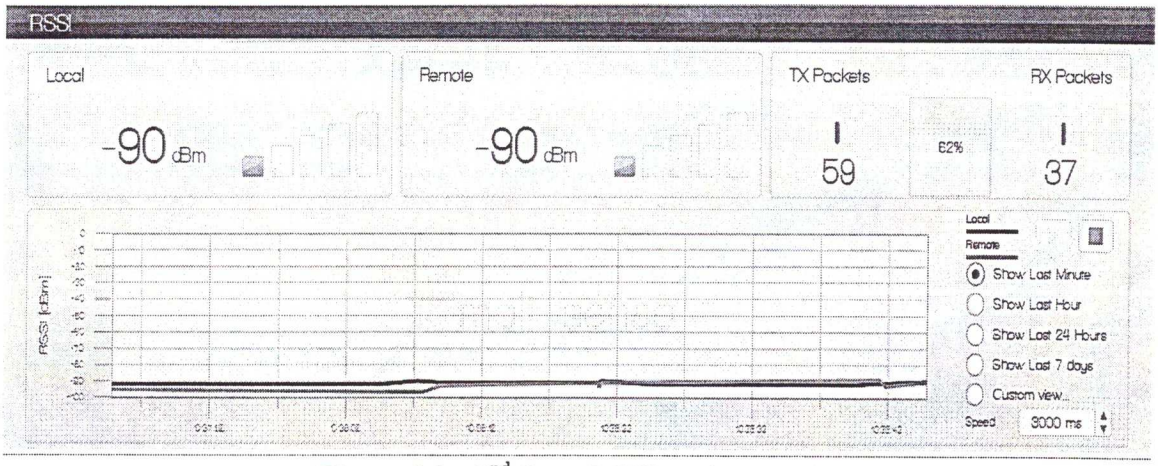


Figure 5.9 : 2nd floor RSSI testing results

5.1.1.1.5 Discover nodes test

This test involves discovering nodes of the PAN network. Node discovery is done through X-CTU and AT command.

After entering AT command by typing (+++), ATND (Node Discover) command which is performed to discovers and reports all modules on its current operating channel and PAN ID, all the nodes that have joined the network will respond to the command with their information.

The figure below illustrates the test results which has done by router1, from the result we have the node address, node identifier and the MY (16-bit Network Address) value.

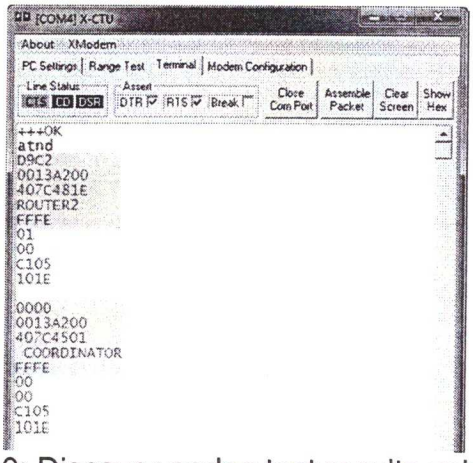


Figure 5.10: Discover nodes test results using X-CTU

5.1.1.1.6 Testing Serial Communications

This test has been done using X-CTU application, after connecting the three modules with the right serial configurations, one click on Test/Query button will perform the test and shows its result at a new window as shown in the figure below.

The results show that the communication with modem is success and gave the serial number, modem firmware version and the modem type for the tested radio.

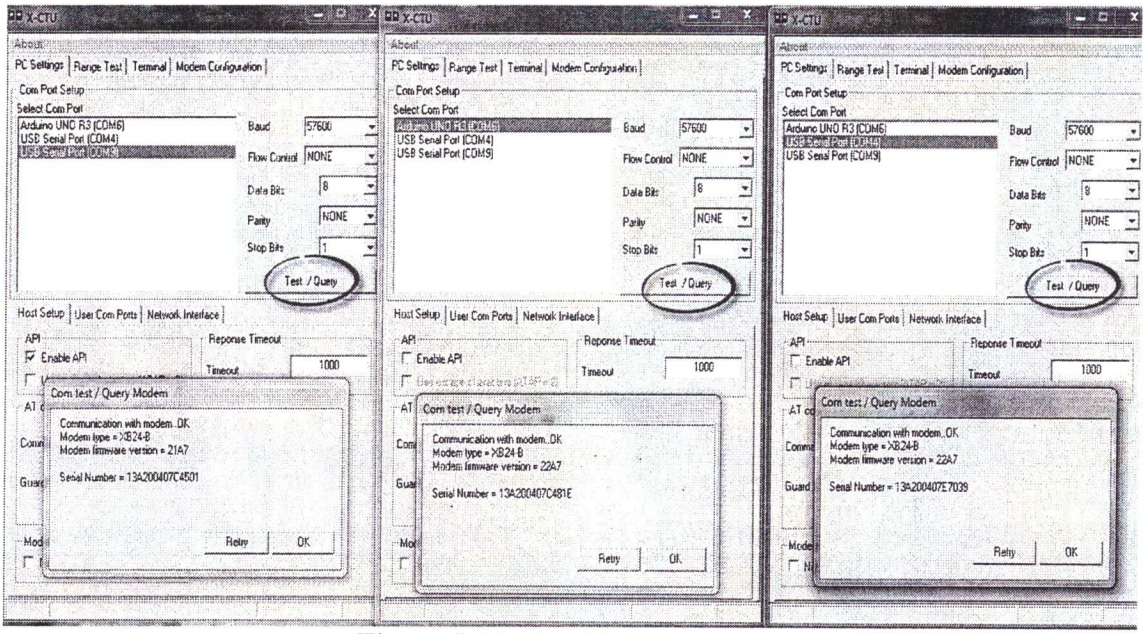


Figure 5.11: Test/Query results

5.1.1.1.7 Chatting test

Chatting between the network modules has been tested using the terminal of the X-CTU, the two routers send assembled packet to coordinator, and the coordinator sends API packets for each route. And the chatting was successfully done without any error or missing packets as shown in the following figure.

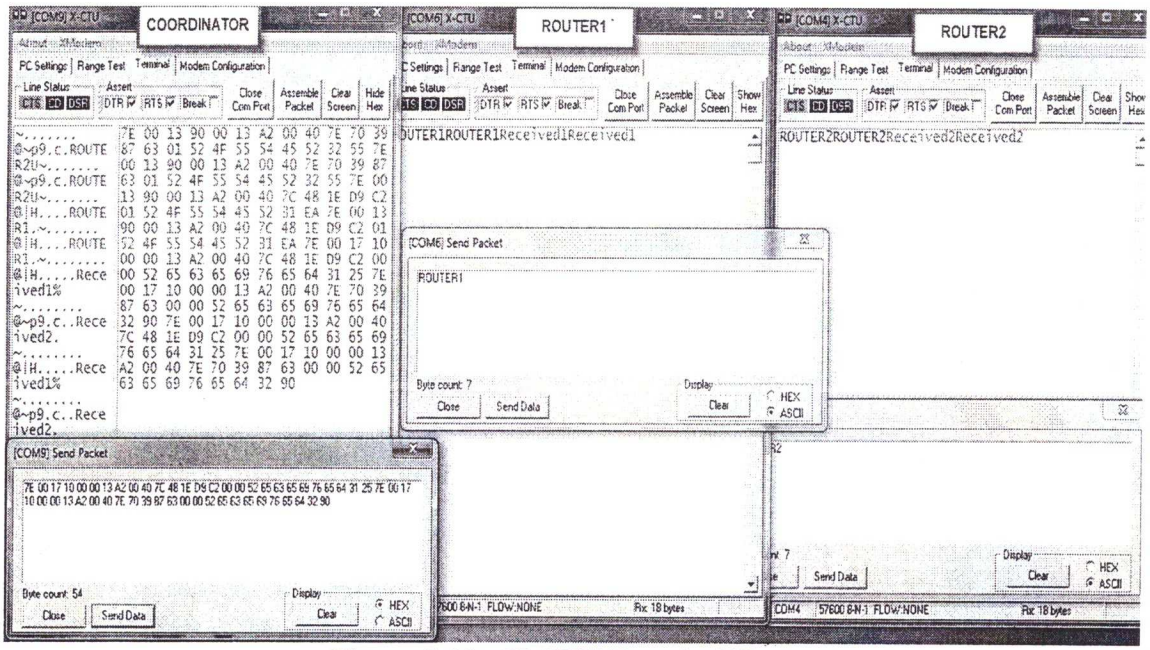


Figure 5.12 : X-CTU Terminal Test

5.1.2 SMS Module

To test the functionality of the SMS Module, the following interface is built as shown in figure 5.13 , and it contains four functions: connect, data received, send message and display last message as the following graphical interface:

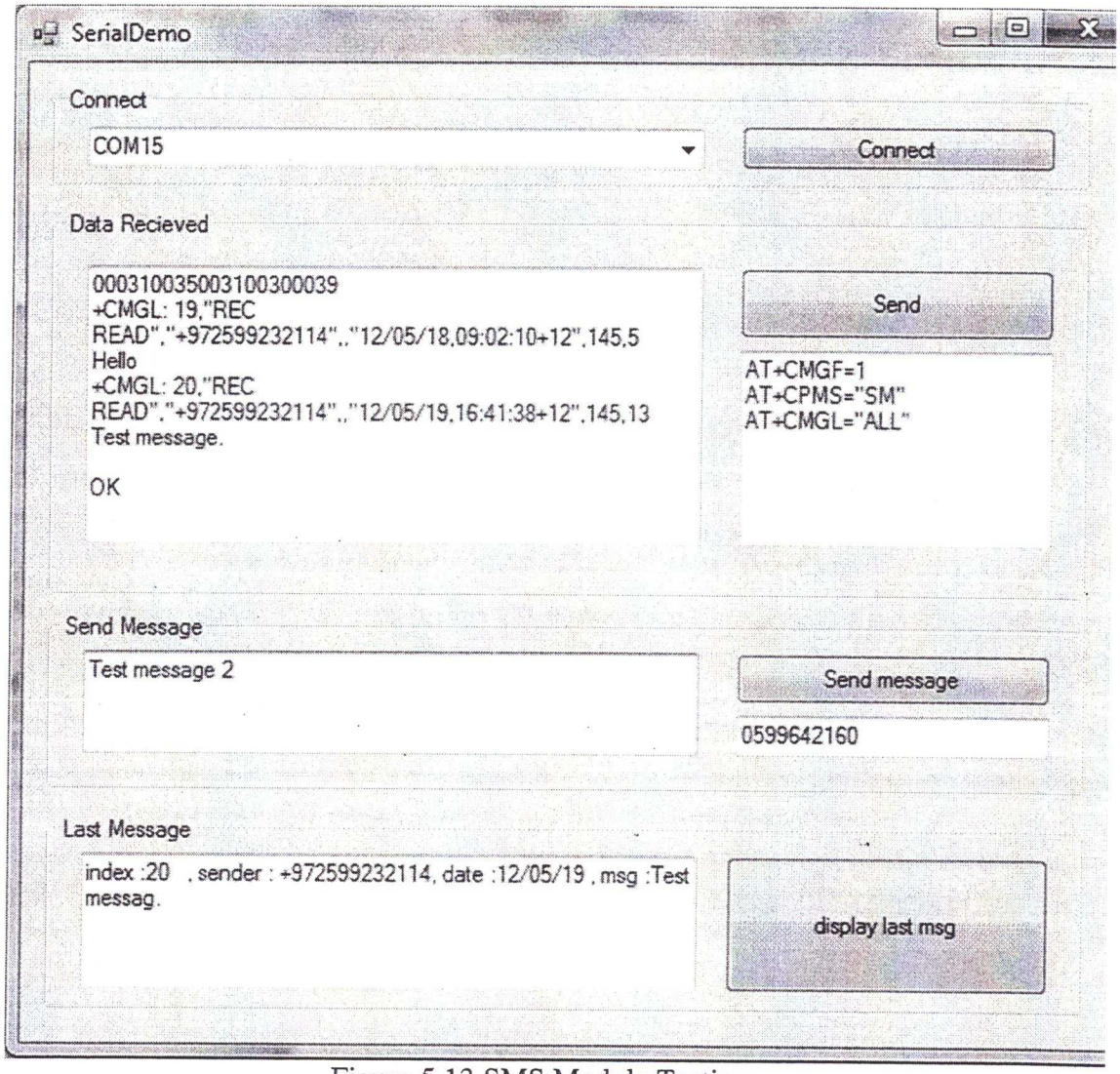


Figure 5.13 SMS Module Testing

The table 5.1 shows the tests that were executed on the GSM Module:

Table 5.1 GSM Tests		
Action	Description	Result
Connect("COM15")	COM15 used in this function is the port name of the GSM modem connected with the computer. After all the properties are set and the function is called,	a message is displayed to show a successful connection.
Send("AT+CMGF=1") Send ("AT+CPMS=SM") Send("AT+CMGL=ALL")	Use AT command to communicate with GSM by write on the send buffer and then to the COM15 port,at first send "AT+CMGF=1" to set sms to text mode, "AT+CPMS=SM" to browse the sms memory, then "AT+CMGL=ALL" is used to display all messages in mobile	All Messages were displayed, in the following format: +CMGL: 19,"REC READ","+0599642160",,"12/05/18,09:02:10+12" Hello
DataReceived()	This event is triggered whenever the GSM modem sends data, and here a new message is received from another mobile and analyzed.	"index :20 , sender : +0599642160, date :12/05/19 , msg : Test messag."
SendMessage("0599642160", "Test")	Using this function to send to the mobile number 0599642160 a message "Test"	The mobile did receive a message from the SIM number containing the text "Test"
DisplayLastMessage()	Return the last message from the archive and Display it in textbox .	Displays the message components in a new window.

all of the tests mentioned in table 5.1 were done multiple times throughout the development of the project, and all proved to provide successful results.

5.1.3 Transducers and Control Nodes

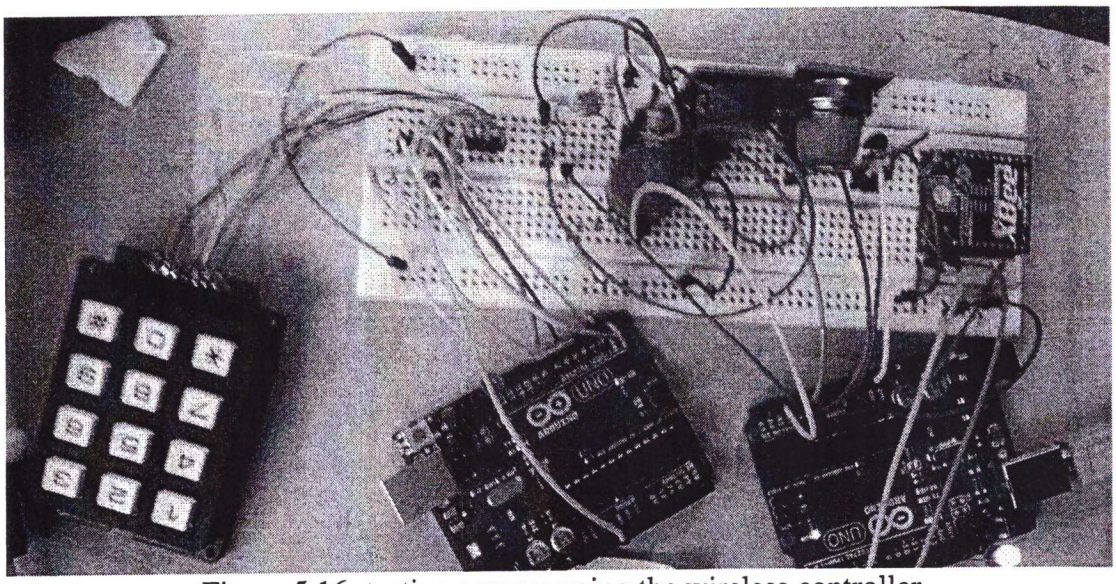


Figure 5.16: testing sensors using the wireless controller

5.1.3.1 Current Transducer

The output for the current sensor was tested using a few known loads, and the following were its output:

Table 5.2 : Testing the current Transducer

Load	Rating	Theoretical Output (Volts)	Actual Output (Volts)	Calculated Power (Watts)
Heater	2000W	1.13636	1.109 V	1950 W
TV	400W	0.227272	0.219 V	
Automatic Washer	900W	0.511363	0.485V	

Theoretical Output = (Rating/220V) * Conversion Coefficient

The actual output is somewhat different than the theoretical output, but it is assumed that this is due to variation in electrical voltage and non-constant load rating.

5.1.3.2 Gas sensor sensitivity test

In this test gas sensitivity was tested, by changing the value of sensor's potentiometer to appropriate value.

5.1.3.2.1 MQ2 test

Regarding to MQ-2 sensor testing, different outputs were taken at different sensitivity related to potentiometer value to determine the suitable sensitivity.

After exposed the sensor to flammable gas such as Butane, the outputs below show that 2.4kohm has better sensitivity response to gas concentration with less than level 200 for the normal concentration at indoor environment.

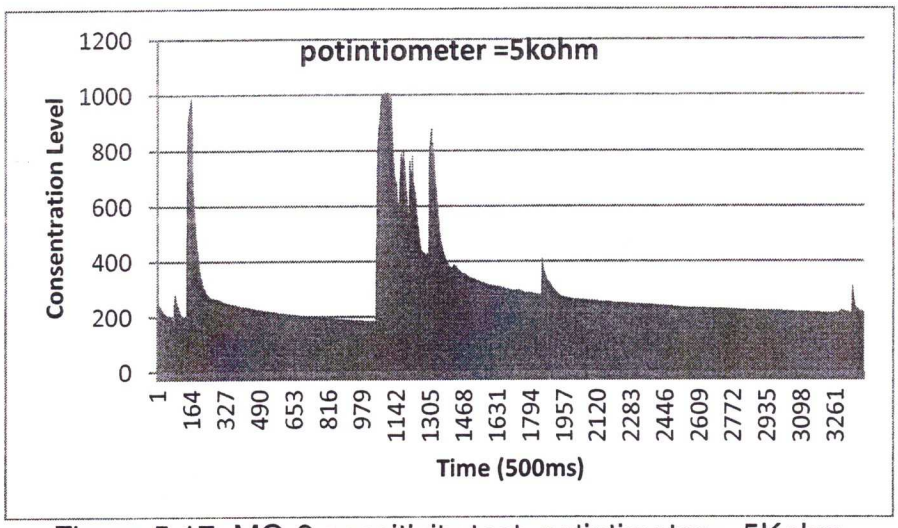


Figure 5.17: MQ-2 sensitivity test, potintimeter = 5Kohm

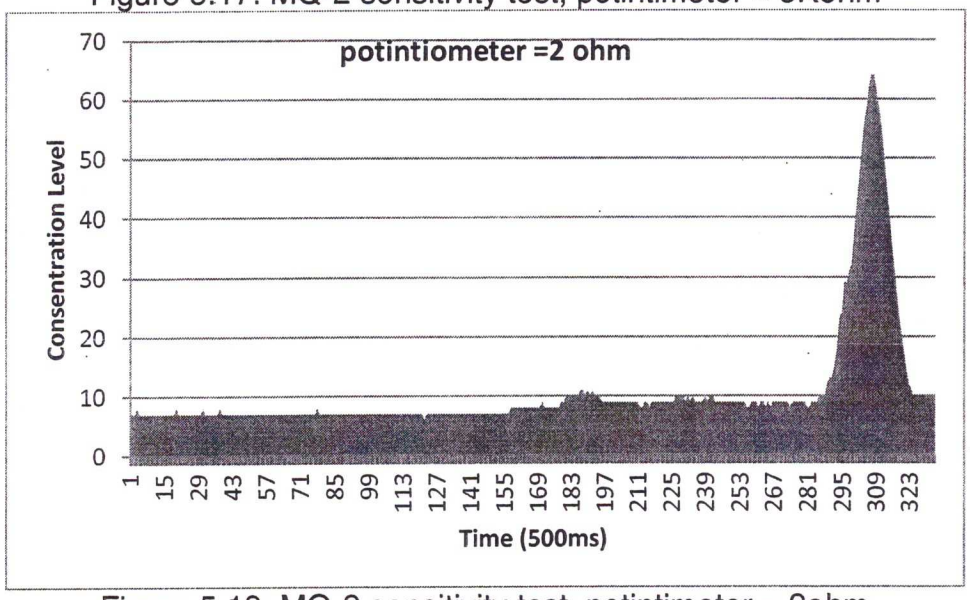


Figure 5.18: MQ-2 sensitivity test, potintimeter = 2ohm

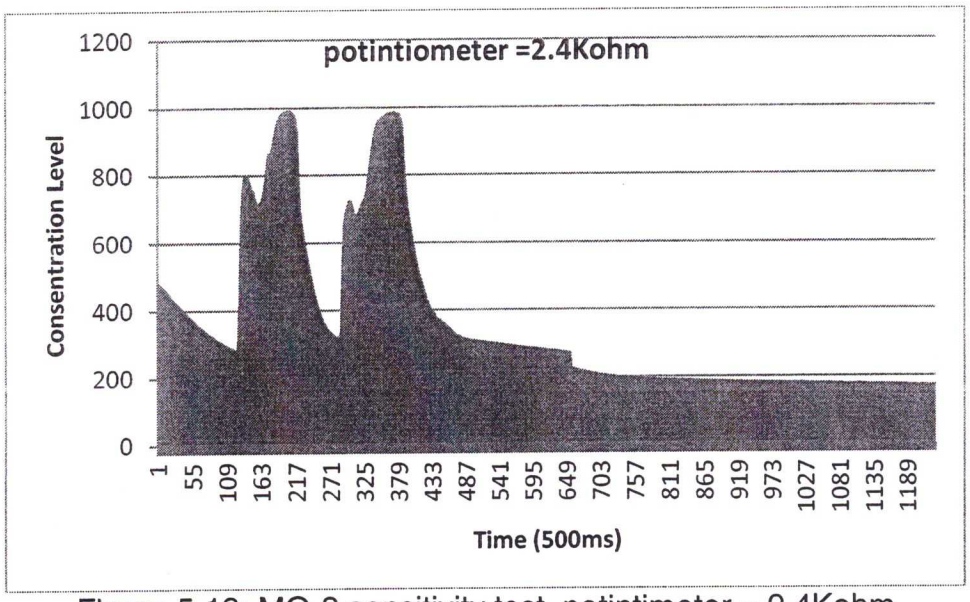


Figure 5.19: MQ-2 sensitivity test, potintimeter = 2.4Kohm

5.1.3.2.2 MQ3 test

Regarding to MQ-3 sensor testing, different outputs were taken at different sensitivity related to potentiometer value to determine the suitable sensitivity.

After exposed the sensor to pure medical alcohol, the outputs below show that 1.27kohm has better sensitivity response to gas concentration with less than level 180 for the normal concentration at indoor environment.

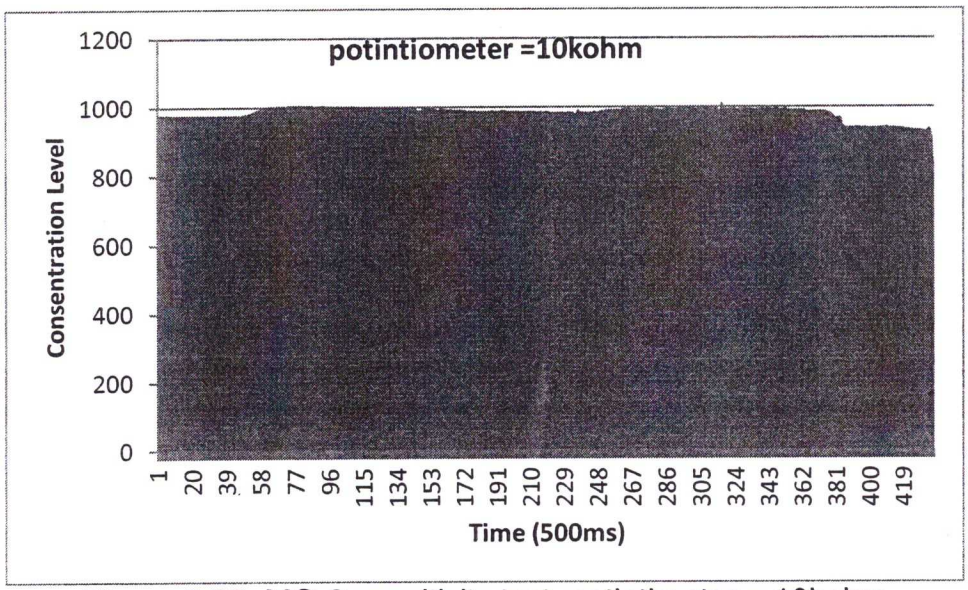


Figure 5.20: MQ-3 sensitivity test, potintimeter = 10kohm

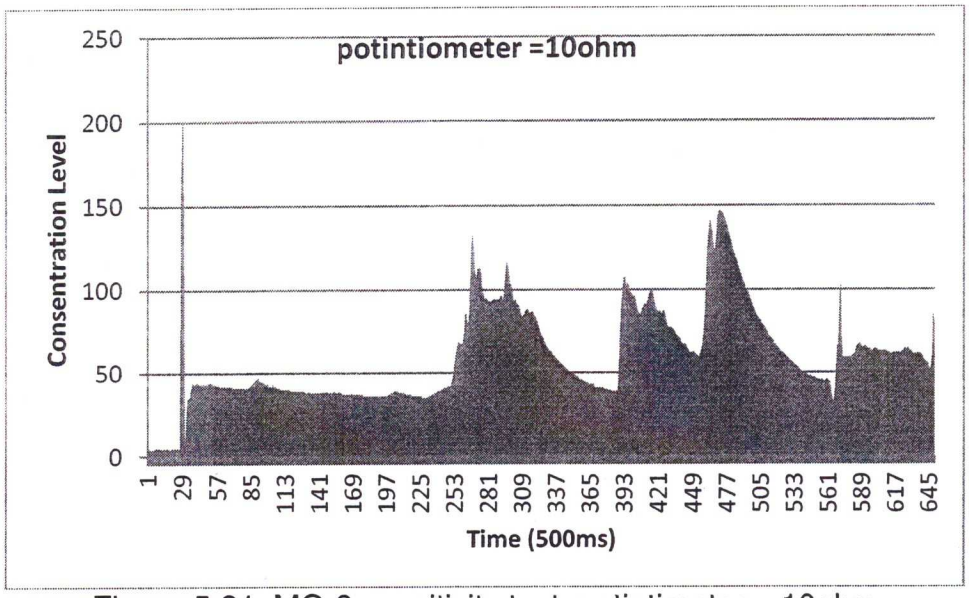


Figure 5.21: MQ-3 sensitivity test, potintimeter = 10ohm

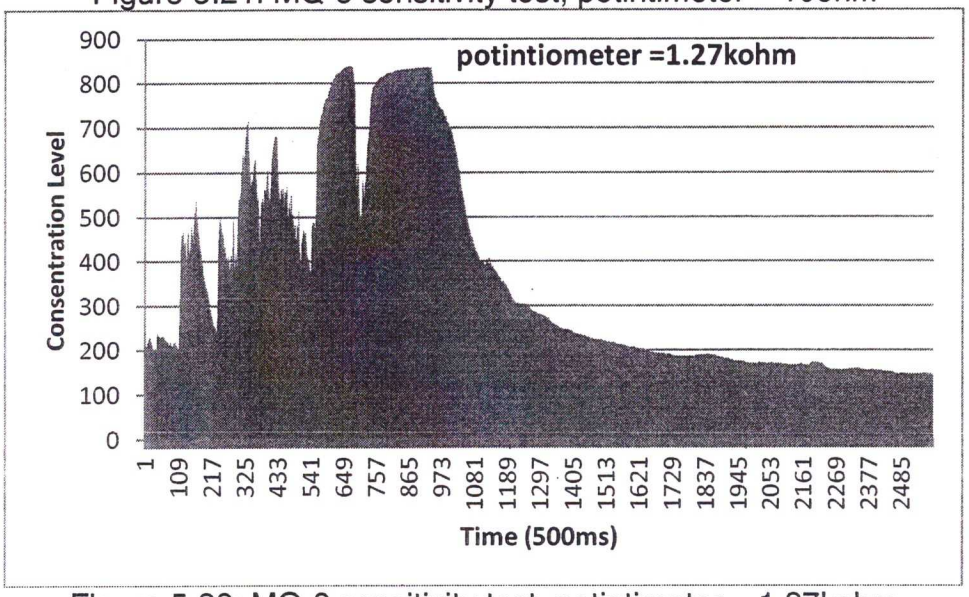


Figure 5.22: MQ-3 sensitivity test, potintimeter = 1.27kohm

5.1.3.2.3 MQ4 test

Regarding to MQ-4 sensor testing, different outputs were taken at different sensitivity related to potentiometer value to determine the suitable sensitivity.

After exposed the sensor to flammable gas such as cooking gas, the outputs below show that 1kohm has better sensitivity response to gas concentration with less than level 50 for the normal concentration at indoor environment.

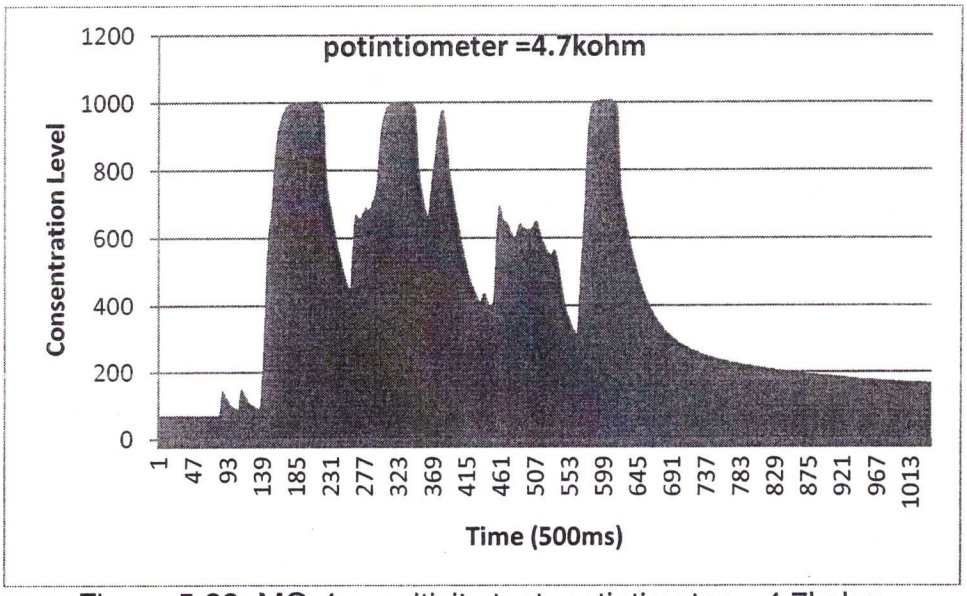


Figure 5.23: MQ-4 sensitivity test, potintimeter = 4.7kohm

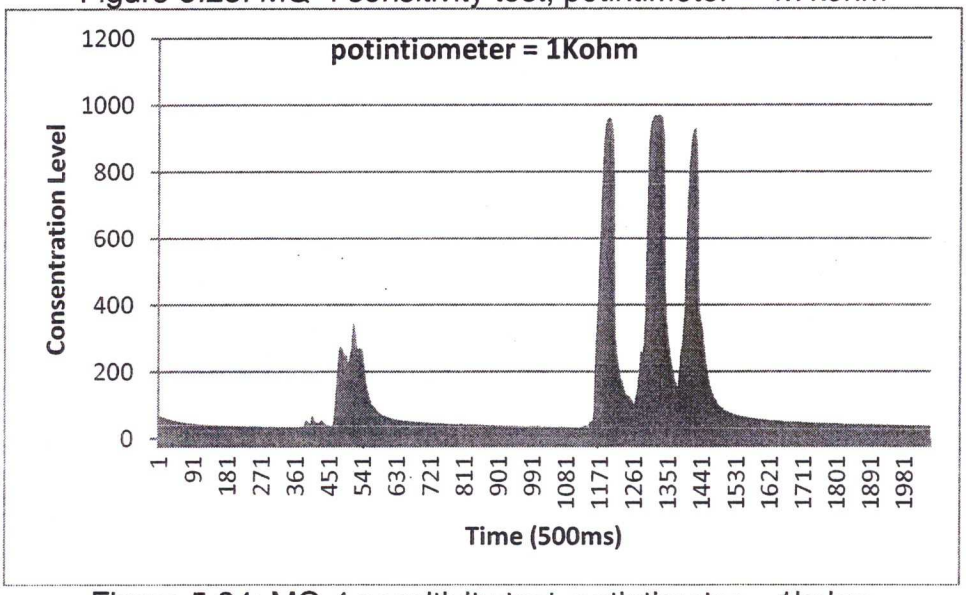


Figure 5.24: MQ-4 sensitivity test, potintimeter = 1kohm

5.1.3.2.4 MQ7

Regarding to MQ-7 sensor testing, different outputs were taken at different sensitivity related to potentiometer value to determine the suitable sensitivity.

After exposing the sensor to some gases such as cigar smoke, flammable gases and burning smoke, the outputs below show that 330ohm has better sensitivity response to gas concentration with less than level 6 for the normal acceptable concentration at indoor environment.

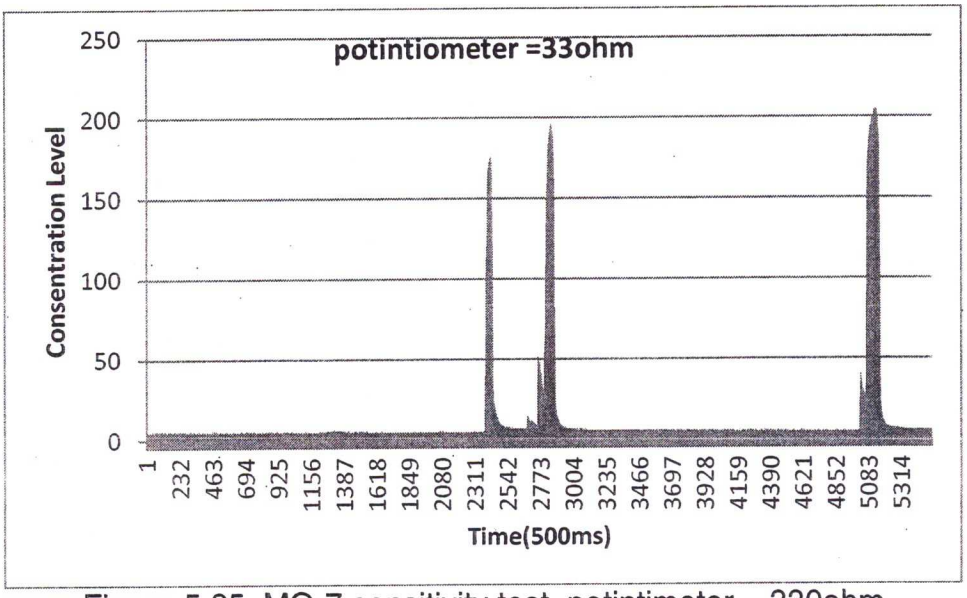


Figure 5.25: MQ-7 sensitivity test, potintimeter = 330ohm

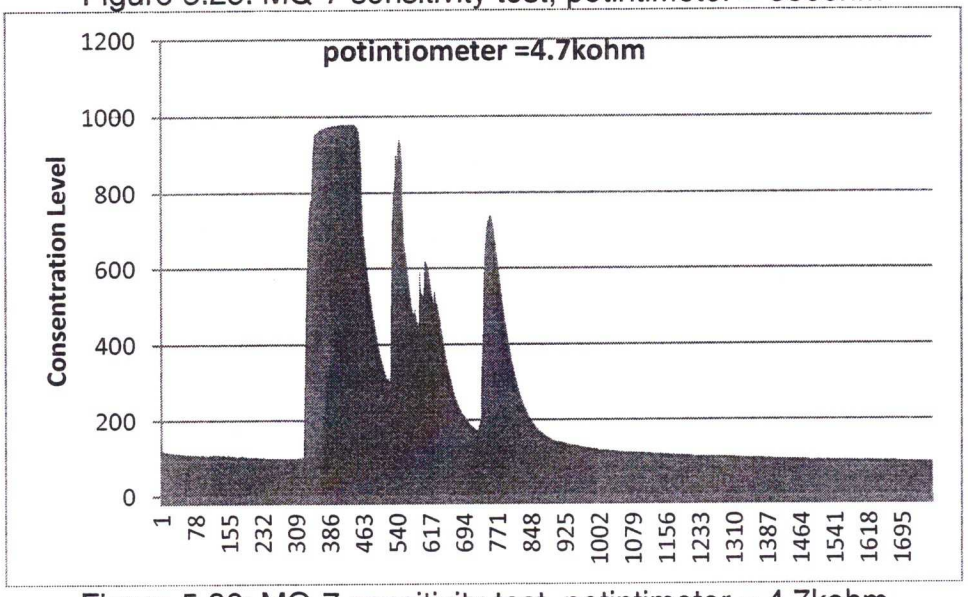


Figure 5.26: MQ-7 sensitivity test, potintimeter = 4.7kohm

5.1.3.3 Photocell

In this test, the light intensity was taken at different level inside a standard residential building, the output divided to five variables which are dark, dim, light, bright and very bright; the following table shows the test results.

Analog reading = 809 - Very bright Analog reading = 808 - Very bright Analog reading = 633 - Bright Analog reading = 53 - Dim Analog reading = 43 - Dim Analog reading = 396 - Light Analog reading = 414 - Light Analog reading = 485 - Light Analog reading = 823 - Very bright Analog reading = 825 - Very bright Analog reading = 823 - Very bright Analog reading = 824 - Very bright Analog reading = 823 - Very bright Analog reading = 969 - Very bright Analog reading = 968 - Very bright Analog reading = 981 - Very bright Analog reading = 995 - Very bright Analog reading = 1009 - Very bright Analog reading = 947 - Very bright Analog reading = 809 - Very bright
Figure 5.27 : Photocell light intensity test

5.1.3.4 Range Of PIR Sensor

In this test PIR sensors placed at the door frame to detect motion and then placed inside the room to detect the motion inside the room.

Normally, if there is no obstacle in front of the sensor the tested range is as shown in figure below.

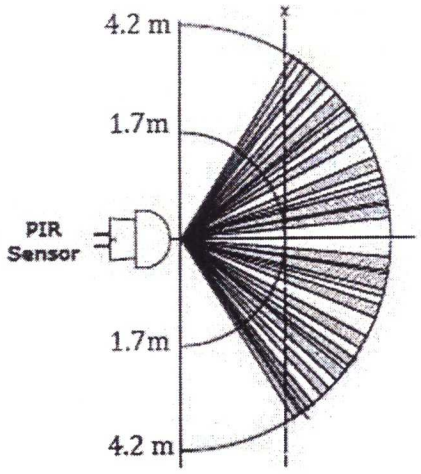


Figure 5.28: Range of PIR sensor if there is no obstacle

As shown in the Figure 5.28, the detection range of PIR sensor is approximately 4.2 meters to forward, 1.7 meters to left and right side of the sensor and these results took from inside the room.

With regard to door frame the range should be reduced to detect only the motion that occurs too close from the door. So PIR placed adjacent to the wall covered front of the sensor eye with a small cap. Thus, the range reduced to approximately 1 meter as shown in Figure 5.29.

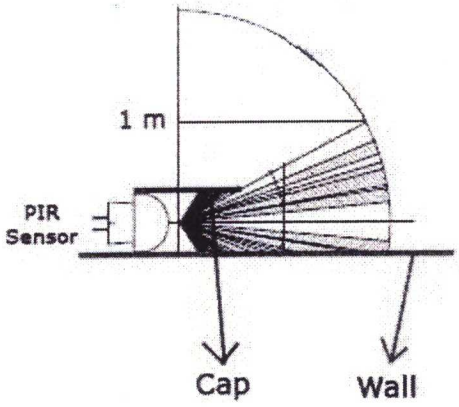


Figure 5.29 Range of PIR sensor is reduced with obstacles

At the beginning of the operation, PIR sensor took some seconds to calibrate, so after calibration it will detect motion, the output of the sensor was tested in multiple distances and angles after calibration ended and sensor is activated, the following outputs was taken at the beginning of the motion till the end in seconds.

Motion detected	Motion ended	Range
motion detected at 33 sec	motion ended at 75 sec	0.3 meter
motion detected at 94 sec	motion ended at 122 sec	3.5 meter
motion detected at 223 sec	motion ended at 256 sec	1 meter
motion detected at 328 sec	motion ended at 366 sec	1.8 meter
motion detected at 395 sec	motion ended at 453 sec	4 meter
motion detected at 473 sec	motion ended at 638 sec	2 meter
motion detected at 647 sec	motion ended at 695 sec	0.5meter

5.1.3.5 Water flow testing

The output of water flow was testing using tap water as the following table:

Input quantity (liters)	Output quantity (liters)
1.5	1.4
2	1.8
3	2.8
4	3.7

The actual output is somewhat different than the theoretical output, but it is assumed that this is due to sensor tolerance and reading resolution.

5.1.3.6 LM35 testing

The output of the LM35 temperature sensor was tested in a standard residential building; sensor was exposed to variable temperature at the building, in addition it was exposed to other effects to vary its temperature; the table below illustrates the output results achieved.

Analog output (volt)	Temperature degrees C
3.4	35.88
3.3	34.8
3.1	32.44
0.29	30.81
0.27	28.86
0.26	28.37
0.26	27.88
0.15	16.79
0.10	11.85

5.1.4 Hardware System Assembly

The hardware system will connect the following control nodes and sensors:

Wired Controller		4 Control Nodes
		LM35 Temprature Sensor
		PIR motion detector Sensor
Wireless controller	Node 1	Arduino microcontroller
		XBee series 2
		MQ2, MQ3, Photocell
		One control node
	Node 2	Arduino microcontroller
		XBee series 2
		Keypad, MQ4, MQ7
		One control node

These will be presented on a presentation wood stand contained a 32" TV screen, computer, lighting spots, wooden drawers, locks.

Chapter Six

System Applications Future Works

6.1 system savings
6.2 System Applications and Scenarios
6.3 Conculsion

6.1 System Savings

Following the implementation of the system, a study on the feasibility of the system is to be made, and how much it is expected to cost and save.

The general cost of the system does not exceed 3000$, and while it provides a lot of results, it is expected that the system is will also introduce savings on the electrical bill that varies from one environment to another. The following is an example for a standard Palestinian household environment.

Using the data obtained from Palestinian Central Bureau of Statistics, it is concluded that the average house-hold consumption for Palestinian families in 2010 ranges between (200KWH/Month and 400KWH/Month). Thus, it is assumed that the average consumption is around 300KWH/Month.

Applying this data to the cost of KWH in Hebron which is 0.58 NIS, we can deduce that on average, Families in Hebron pay electric bills somewhat around 174 NIS/Month

It is also expected that this system will serve a minimum lifetime of 10 years before the need of maintenance. Evidently, with the all these figures mentioned, and basing on the fact that the system can save anywhere from 10% to 40% the following figures can be stated:

Electricity minimal costs for 10 years $= 20 \times 12 \times 174 = 20880$ NIS
(Worst Case) With 10% Saving, Savings = 2088 NIS
(Average) With 20% Saving, Savings= 4176 NIS
(Best Case) With 40% Saving, Savings= 8552 NIS

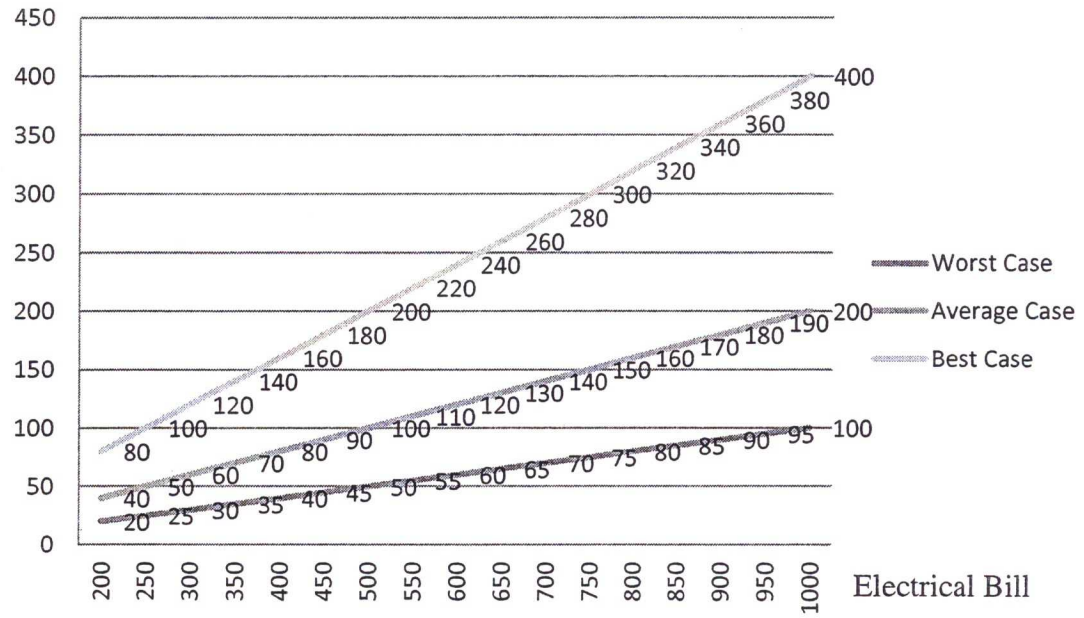


Figure 6.1 savings

For houses that consume more than 1000KWH per month, savings can reach on average 200NIS per month, which translates to savings on average up to 24000NIS for the lifetime of the system.

6.2 System Applications and Scenarios

- Scenario 1: Gas Detection

  This scenario is prepared to warn residents about gas leaking, toxic gas sand air quality, even if they are away from the building it will notify them on mobile .

- Scenario 2: Lighting management

  This scenario could be used in standard residential buildings and commercial building such molls, shops, offices …etc.

  If the resident leave the room and forgot to turn off the lights thy system will manage that and turn off the lights, and it will turn on when resident came back.

  Another example, at moll's gallery the lights will turns on when the customers are close to gallery and it will turn off if there is no close customer.

- Scenario 3: Security

  This scenario is prepared to ensure building security by using electronic lock, motion detectors.

- Scenario 4: Power monitoring

  This scenario is prepared to monitor the building power consumption and billings.

- Scenario 5: Controlling grid outlets

This scenario is prepared to control building's outlets in the electrical grid.

- Scenario 6: Management of heating system

  This scenario is prepared to control building's temperature to maintain on suitable degree of temperature.

- Scenario 7: Entertainment

  This scenario is prepared to provide entertainment through movie, music and pictures databases with in the system.

6.3 Conclusion

The project lays the foundation for multiple future projects where it can be developed in multiple fields and directions, such as SCADA systems and distributed control systems, artificial intelligence home systems, and even automation in farms and industrial locations.

This project lays the technology behind simple affordable automation in buildings, and it proved a successful tool in this task.

References

1. Measurement of standby power for selected electrical appliances in Australia. **Lisa Guana, Trevor Berrill b, Richard J. Browna.** 2-3, Brisbane, Australia : School of Engineering Systems, Queensland University of Technology, February 2011, Sustainable Energy Systems Consultant & Educator, Vol. 43, pp. 485-490.

2. **Manar Ikhlayil, Sabha Abu Sabha, Yasmin Al-Shamesti.** Monitoring and Controlling Home by Cellular Technology. Hebron : Palestine Polytechnic University, 2010.

3. **Insherah Badawi Abu Soloom, Rana Jamal Abd Alla, Reem Ahmad Abu Sharar.** Smart Home. Hebron : Palestine Polytechnic University, 2009.

4. **Ahmad Abdelsameh Abu-Sneineh, Shurooq Farid Al-Qawasmi, Wedad Wajeeh Al-Hadad.** Automated Home System. Hebron : Palestine Polytechnic University, 2007.

5. Standby power requirements of household appliances in Canada. **Alan S. Fung, Adam Aulenback, Alex Ferguson, V. Ismet Ugursal.** 2 , Halifax, NS, Canada : Department of Mechanical Engineering, Canadian Residential Energy End-Use Data and Analysis Centre (CREEDAC), 21 April 2002, Vol. 35, pp. 217-228.

6. Microchip Corp. [Online] October 10, 2009. [Cited: July 21, 2011.] http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010300.

7. **Mosers, Christian.** Introduction to Windows Presentation Foundation. WPF Tutorials. [Online] January 17, 2010. [Cited: December 10th, 2011.] http://www.wpftutorial.net/WPFIntroduction.html.

8. XAML Extensible Application Markup Language. Tech Target. [Online] August 2006. [Cited: December 7, 2011.] http://searchwindevelopment.techtarget.com/definition/XAML.

9. **Microsoft.** Microsoft Speech Platform SDK Documentation. December : Microsoft Corporation, 2010.

10. Universal Serial Bus. Silicon Imaging. [Online] [Cited: September 7, 2011.] http://www.siliconimaging.com/universal_serial_bus.htm.

11. **Telecom Space.** GSM Overview. TelecomSpace.com. [Online] [Cited: November 5, 2011.] http://www.telecomspace.com/gsm.html.

12. XBee Wireless Module. Open Circuit. [Online] September 21, 2010. [Cited: October 10, 2011.] http://www.opencircuits.com/Xbee_wireless_module.

13. XBee Starter Guide. Tristan Tech. [Online] October 10, 2010. [Cited: October 10, 2011.] http://www.tristantech.net/articles/xbee_tutorial/1.php.

14. Arduino: Getting Started. Arduino. [Online] [Cited: October 10, 2011.] http://www.arduino.cc/en/Guide/Introduction.

15. Introduction to Liquid Crystal Displays. Case Western Reserve University. [Online] August 2009. [Cited: November 31, 2011.] http://plc.cwru.edu/tutorial/enhanced/files/lcd/intro.htm.

16. **Ajamia, Hamza Abu.** The 8255A Programmable Peripheral Interface. Scribd. [Online] April 3, 2011. [Cited: December 14, 2011.] http://www.scribd.com/doc/49985106/3-General-Purpose-Programmable-Peripheral-Devices.

17. **DiLouie, Craig.** Occupancy Sensors: Passive Infrared, Ultrasonic And Dual-Technology. Facilities Net. [Online] September 2008. [Cited: November 4, 2011.] http://www.facilitiesnet.com/lighting/article/Occupancy-Sensors-Passive-Infrared-Ultrasonic-and-DualTechnology--9608.

18. Temperature Sensor Tutorial. Instructables. [Online] June 15, 2009. [Cited: October 1, 2011.] http://www.instructables.com/id/Temperature-Sensor-Tutorial/.

19. Air Quality Sensor. Wiring. [Online] October 4, 2010. [Cited: December 5, 2011.] http://arduino.cc/playground/Main/MQGasSensors

20. Current sensor. Wikipedia. [Online] May 3, 2011. [Cited: December 5, 2011.] http://en.wikipedia.org/wiki/Current_sensor.

21. **Koon, William.** Current Sensing for Energy Metering. Analog. [Online] [Cited: December 5, 2011.] http://www.analog.com/static/imported-files/recommended_reading/688920132Current_sensing_for_meteringNew.pdf.

22. G1/2 Water Flow sensor. Seed Studio. [Online] December 14, 2011. [Cited: December 19, 2011.] http://seeedstudio.com/wiki/G1/2_Water_Flow_sensor.

23. Smoke Detector. Wikipedia. [Online] December 13, 2011. [Cited: December 20, 2011.] http://en.wikipedia.org/wiki/Smoke_detector.

24. PIC18F4550. Engineering Garage. [Online] [Cited: August 5, 2011.] http://www.engineersgarage.com/electronic-components/pic18f4550-microcontroller.

25. **Kharga, Bineeta.** Microcontrollers. Scribd.com. [Online] August 4, 2011. [Cited: December 8, 2011.] http://www.scribd.com/doc/52605786/Microcontroller.

26. **Koch, Frans H. "Emissions Produced by 1 Kilowatt-hour of Electricity Based on Life-Cycle Analysis."** Lecture. Hydropower-Internalized Costs and

Externalized Benefits. Canada, Ottawa. *Nuclear Energy Institute*. International Energy Agency (IEA)-Implementing Agreement for Hydropower Technologies and Programs, 21 July 2007. Web. [Cited: 5 Aug. 2011]. http://www.nei.org/resourcesandstats/documentlibrary/protectingtheenvironment/graphicsandcharts/lifecycleemissions

Appendix A

Survey

نموذج استبيان

نرجو من حضرتكم التكرم بتعبئة بيانات الاستبيان المرفق والذي تم إعداده للتعرف على مرئياتكم وآرأءكم حول إستخدام البيوت الذكية في مدينة الخليل، ونتطلع لتعاونكم الكرم معنا، علماً بأن الآراء التي سنحصل عليها خاصة بأغراض البحث والدراسة فقط.

شاكرين ومقدرين لكم كرم تعاونكم
ونفضلوا بقبول أطيب التحية والتقدير

الإسم :.....

العنوان :.....

الهاتف /البريد الإلكتروني :.....

مساحة البيت :.....

- هل أنت مهتم بالمنتجات التكنولوجية بشكل عام؟

☐ نعم ☐ لا ☐ إلى حد ما

- هل تثق بمثل هذه المنتجات؟

☐ نعم بالكامل ☐ نعم إلى حد ما ☐ لا ☐ لا على الإطلاق

- هل سمعت من قبل بالبيوت الذكية ؟ ماذا يعني لك مصطلح "البيت الذكي" Smart Home ؟

.....

.....

.....

.....

.....

.....

المنازل الذكية هي منازل بسيطة متواضعة تختلف عن المنازل التقليدية العادية بأنها منازل مُمَيكنة (مُأتمتة) بالكامل،حيث تعد أنظمة المنازل الذكية وسيلة لتخفيض فواتير الطاقة بشكل كبير نتيجة التحكم بشكل أفضل في معدلات الاستهلاك مع استمرار الفعالية في أداء الأجهزة مما يطيل عمر الأجهزة الافتراضي كنتيجة لعملها بصورة منتظمة وعند الحاجة لها فقط. بمعنى أنها تعتمد على التقنية في جميع مرافق المنزل، حيث تستطيع بضغطة زر أن تتحكم في كل شيء في المنزل، بدءاً من الستائر، الإضاءة، البوابات، مكيفات الهواء ،أنظمة المراقبة والحماية، البيئة الترفيهية، وغيرها من الأجهزة الأخرى.

- لو كانت تكلفة البيت الذكي قريبة من تكلفة البيت العادي أيهما تفضل؟

.....

- إذا كنت *مهتم* بأتمتة بيتك، فما هو السبب؟

☐ بسبب توفيرها للأمان.
☐ بسبب تسهيلها للعمل وتوفيرالوقت.
☐ لحبي لمنتجات التكنولوجيا.
☐ بسبب توفير المال (توفير الطاقة).
☐ أسباب أخرى : .....

- إذا كنت *غير مهتم* بأتمتة بيتك، فما هو السبب؟

☐ لأنه ليس ضرورياً.
☐ لأنه هدر للمال.
☐ لأني لا أعرف الكثير عنه.
☐ أسباب أخرى: .....

يا ترى متى كانت آخر مرة كنت في طريقك إلى عملك وبدأت تتساءل هل نسيت المكواة تعمل أم أنك أغلقتها ؟ أو تتساءل هل قمت بإغلاق المدفئة، أو المصابيح أو التلفاز أو الستائر؟ في الغالب مثل هذه الأسئلة لا يمكن تجنبها، لأنها تحدث بإستمرار، ولكن عندما تدخل التقنية في الموضوع، فإن باستطاعتك نسيان هذه الأسئلة، وتسليمها لها لتجيب بدلاً عنك.

- عند خروجك من البيت هل حصل معك ونسيت إطفاء أو إغلاق أحد الأمور التالية؟

☐ أضواء البيت (الداخلية أو الخارجية).
☐ التلفاز ،الراديو ،الحاسوب ،المكواة ،نظام الترفيه.
☐ أبواب البيت.
☐ حنفية المياه.
☐ الطباخ.
☐ المكيف.
☐ غير ذلك: .....

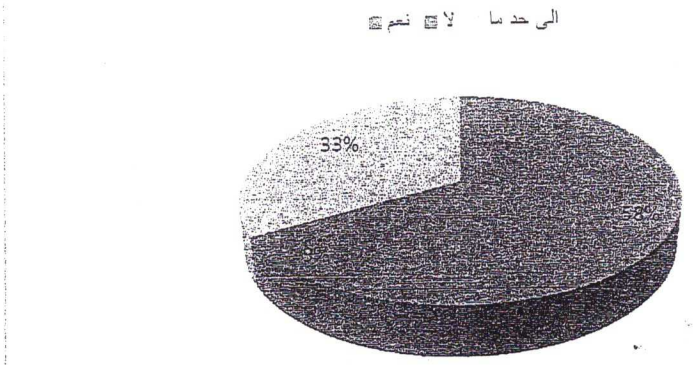
تحتوي المنازل الذكية على شبكة معلومات متكاملة تُنبه الساكن لدرجة الحرارة ، وكميات ثاني أكسيد الكربون الموجودة في البيت، والكميات المستهلكة من الطاقة والمياه ،وتنبهه من مخاطر الحرائق تسربات الغاز والحالات الطارئة، وتساعد ذوي الإحتياجات الخاصة وكبار السن من التحكم في البيت دون بذل مجهود.وبهذا فإن المنازل الذكية هي منازل تفهم ما تريده منها، وتقوم بمعظم الأشياء بدلاً عنك، بالاعتماد على التقنية.

- من هذه المهام أي منها هو الأهم بالنسبة لك؟ ضع اشارة X في المكان المناسب من 1-5 حيث أن الرقم 1 هو للأعلى أهمية والرقم 5 هو الأدنى أهمية.

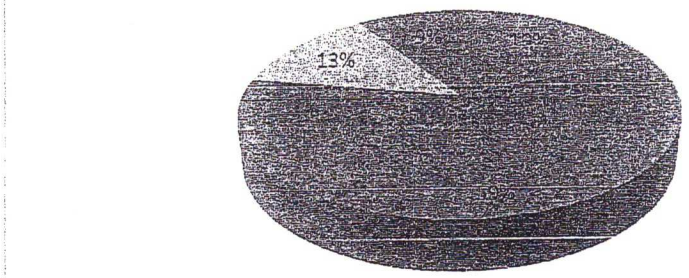
الرقم	المهمة	1	2	3	4	5
1	التحكم المُأتمة بالإضاءة.					
2	التحكم المُأتمة بالستائر.					
3	التنبية والتحذير في الحالات الطارئة (مثل تسربات الغاز أو الحرائق ...).					
4	إمكانية التحكم في البيت من الإنترنت.					
5	التحكم في بيئة البيت (التهوية ،الحرارة ،جو البيت ...).					
6	نظام اتصال بين غرف المنزل.					
7	نظام الترفيه (الموسيقى،الأفلام ،التلفاز ...).					
8	التحكم بالأبواب.					
9	مساعدة ذوي الإحتياجات الخاصة أو كبار السن في التحكم في البيت.					
10	المراقبة الدورية للمنزل بواسطة كاميرات التسجيل.					
11	توفير الطاقة والمياه.					
12	التحكم بالبيت عن طريق الصوت والريموت كنترول عن بعد					
13	مراقبة مجريات الأمور في البيت من استهلاك للطاقة والمياه ودرجات الحرارة وغيرها					

نتائج تحليل الإستبيان

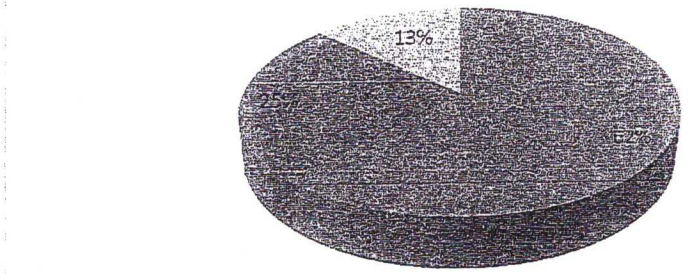
هل أنت مهتم بالمنتجات التكنولوجية بشكل عام؟



هل تثق بمثل هذه المنتجات؟



هل سمعت من قبل بالبيوت الذكية ؟



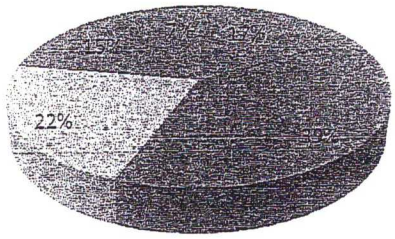
لو كانت تكلفة البيت الذكي قريبة من تكلفة البيت العادي أيهما تفضل؟

البيت العادي ▨ البيت الذكي ▨

إذا كنت مهتم بأتمتة بيتك, فما هو السبب؟

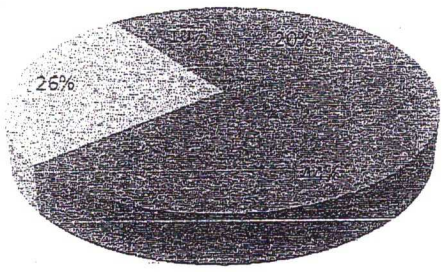
لحبي لمنتجات التكنولوجيا. ▨ بسبب تسهيلها للعمل وتوفير الوقت. ▨ بسبب توفيرها للأمان. ▨

أسباب أخرى ▨ بسبب توفير المال (توفير الطاقة). ▨

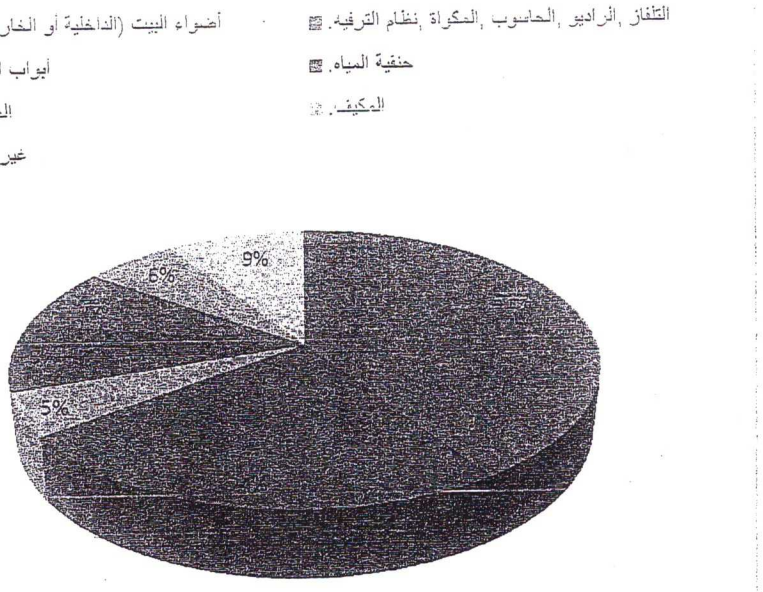


إذا كنت غير مهتم بأتمتة بيتك, فما هو السبب؟

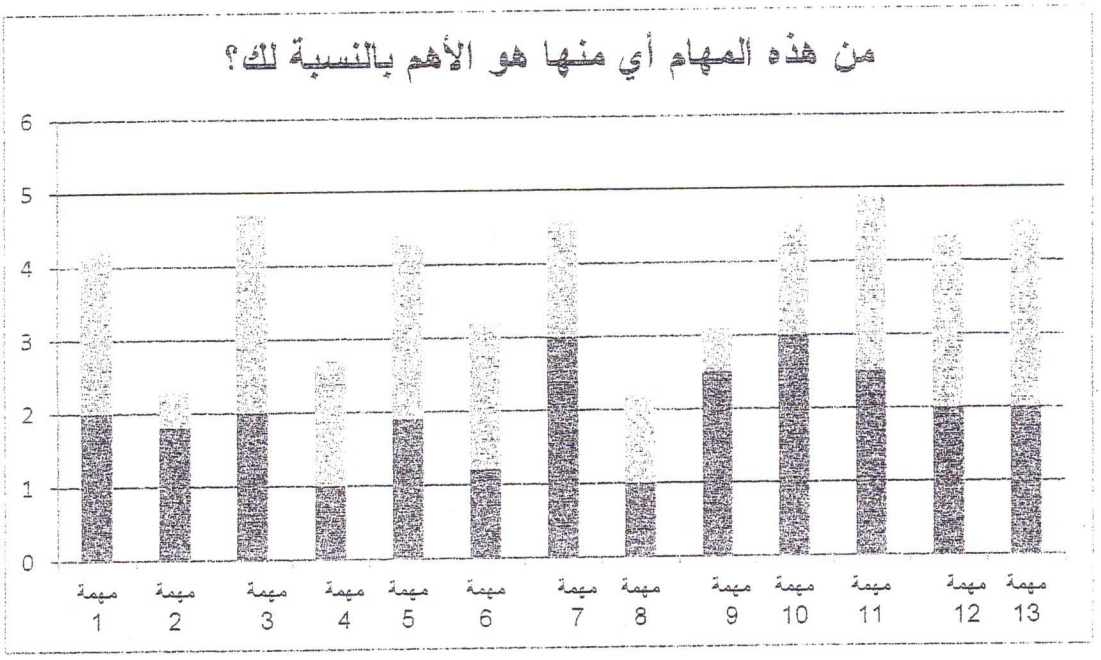
أسباب أخرى ▨ لأني لا أعرف الكثير عنه. ▨ لأنه هدر للمال. ▨ لأنه ليس ضروريا. ▨



عند خروجك من البيت هل حصل معك ونسيت إطفاء أو إغلاق أحد الأمور التالية؟



الرقم	المهمة
1	التحكم المُأتمة بالإضاءة.
2	التحكم المُأتمة بالستائر.
3	التنبية والتحذير في الحالات الطارئة (مثل تسربات الغاز أو الحرائق ...).
4	إمكانية التحكم فى البيت من الإنترنت.
5	التحكم في بيئة البيت (التهوية ,الحرارة ,جو البيت ...).
6	نظام اتصال بين غرف المنزل.
7	نظام الترفيه (الموسيقى,الأفلام ,التلفاز ...).
8	التحكم بالأبواب.
9	مساعدة ذوي الإحتياجات الخاصة أو كبار السن في التحكم في البيت.
10	المراقبة الدورية للمنزل بواسطة كاميرات التسجيل.
11	توفير الطاقة والمياه.
12	التحكم بالبيت عن طريق الصوت والريموت كنترول عن بعد
13	مراقبة مجريات الأمور في البيت من استهلاك للطاقة والمياه ودرجات الحرارة وغيرها



Appendix B

Power consumption facts, figures, and charts

انبعاثات كيلو واط ساعة

Emissions Produced by 1 Kilowatt-hour of Electricity Based on Life-Cycle Analysis

Generation Option	Greenhouse gas emissions gram equiv. (in CO2/kWh)	Sulfur dioxide emissions (in milligrams/kWh)	Nitrogen oxide emissions (in milligrams/kWh)	NMVOC (in milligrams /kWh**)	Particulate matter (in milligrams /kWh)
Hydropower	2 – 48	5 – 60	3 – 42	0	5
Nuclear	2 – 59	3 – 50	2 – 100	0	2
Wind	7 – 124	21 – 87	14 – 50	0	5 – 35
Solar photovoltaic	13 – 731	24 – 490	16 – 340	70	12 – 190
Biomass forestry waste combustion	15 – 101	12 – 140	701 – 1,950	0	217 – 320
Natural gas (combined cycle)	389 – 511	4 – 15,000[*]	13 – 1,500	72 – 164	1 – 10
Coal – modern plant	790 – 1,182	700 – 32,321	700 – 5,273	18 – 29	30 – 663

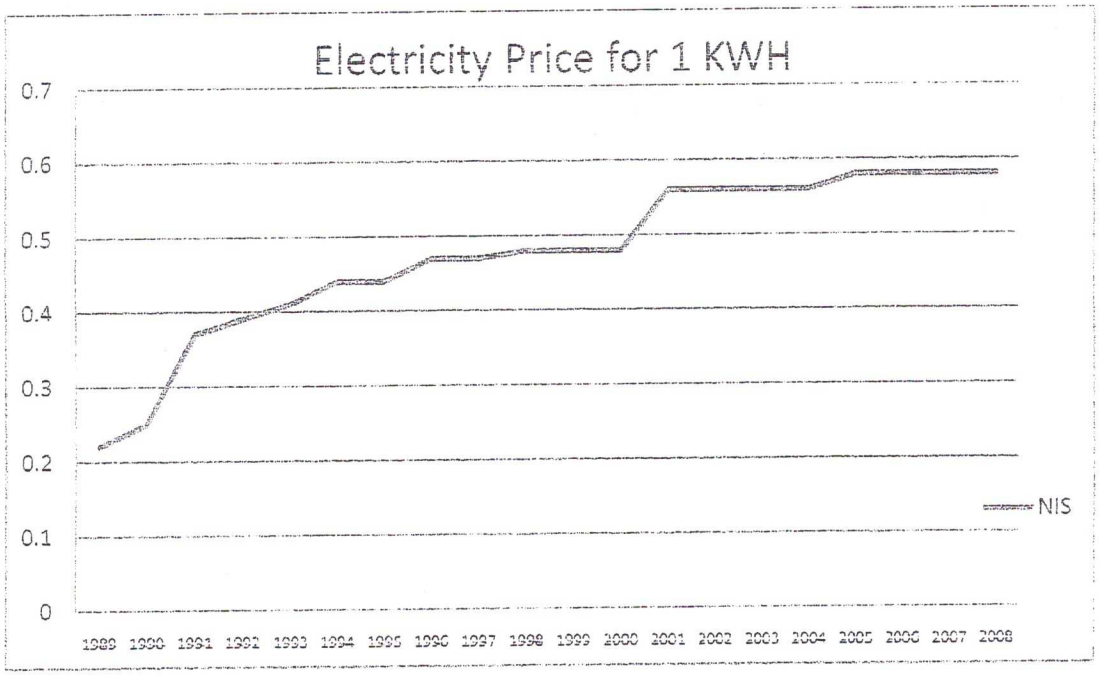
[*] The sulfur content of natural gas when it comes out of the ground can have a wide range of values. When the hydrogen sulfide content is more that 1 percent, the gas is usually known as "sour gas." Normally, almost all of the sulfur is removed from the gas and sequestered as solid sulfur before the gas is used to generate electricity. Only in the exceptional case when the hydrogen sulfide is burned would the high values of sulfur dioxide emissions occur.

** NMVOC means non-methane volatile organic compounds.

الإنبعاثات البيئية الناتجة من واحد كيلو واط في الساعة من الكهرباء[3]

من تاريخ	الى تاريخ	سعر المنزلي (شيكل)
قبل	1989/07/26	0.22
1989/07/26	1990/01/04	0.25
1990/01/04	1990/08/19	0.29
1990/08/19	1990/09/20	0.32
1990/09/20	1990/12/04	0.34
1990/12/04	1991/11/09	0.37
1991/11/09	1992/07/18	0.39
1992/07/18	1992/10/15	0.40
1992/10/15	1993/01/10	0.41
1993/01/10	1993/06/12	0.43
1993/06/12	1995/07/31	0.44
1995/07/31	1997/09/04	0.47
1997/09/04	2000/03/02	0.48
2000/03/02	2004/11/01	0.56
2004/11/01	حتى الان	0.58

تم تزويد البيانات عن طريق بلدية مدينة الخيل قسم الكهرباء

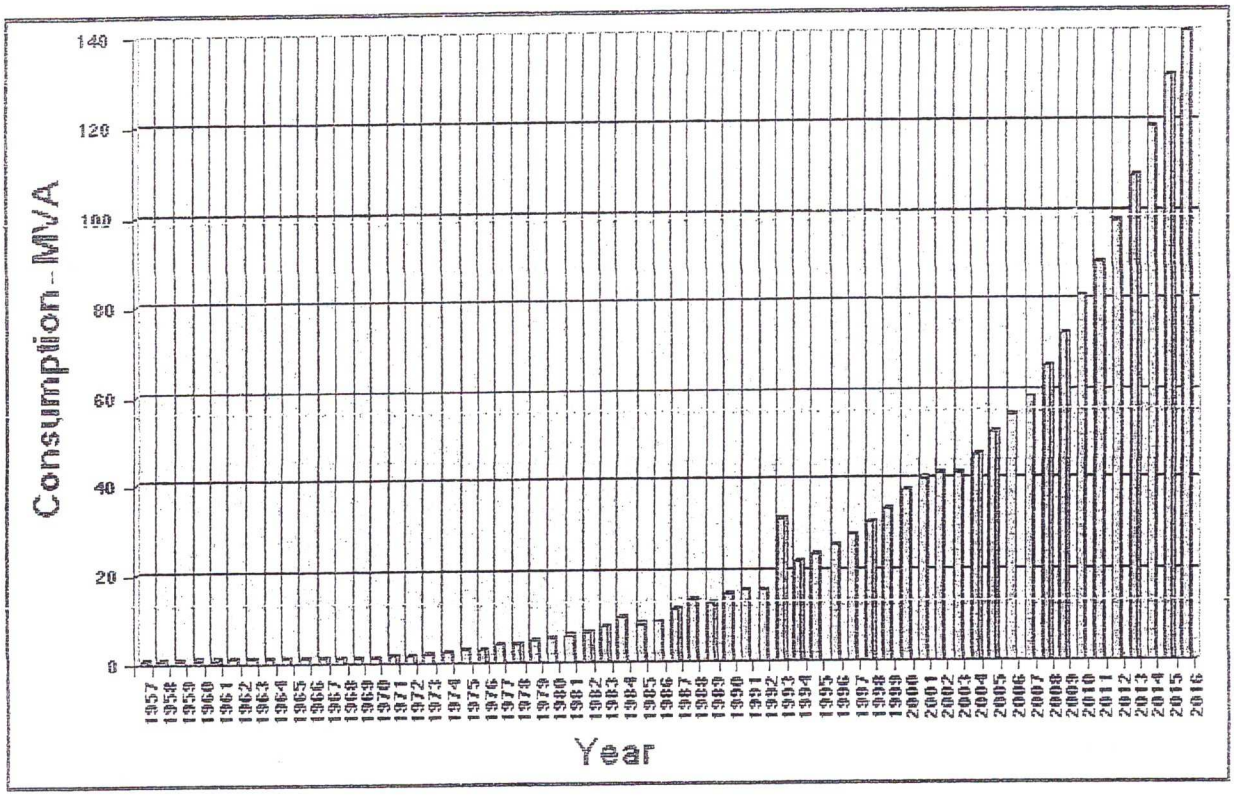


معدل استهلاك الأسرة والفرد من الكهرباء (كيلو واط. ساعة)

السنة	الشهر	الضفة الغربية		شمال الضفة الغربية		وسط الضفة الغربية		جنوب الضفة الغربية	
		الأسرة	الفرد	الأسرة	الفرد	الأسرة	الفرد	الأسرة	الفرد
1999	July	--------	--------	257.4	43	643.1	105	325.1	49
2001	July	277	46	206	36	366	61	280	43
2003	January	279	45	209	35.4	405	64.3	295	44.7
2003	July	283	47.1	201	33.9	439	75.6	226	36.2
2004	January	275.1	45.8	179.6	30.5	418.3	72.1	253.5	40.8
2004	July	269	43.4	180	30.5	414	65.7	232	35.2
2005	January	286	47.7	204	34.0	408	68.0	268	44.7
2005	July	297	54.0	210	38.2	492	89.5	233	42.4
2006	April	200	39.1	184	37.3	278	67.4	209	38.7
2006	July	212	35.3	196	33.8	332	57.2	242	35.4
2008	April	243	44.2	222	41.1	367	70.6	217	36.8
2008	July	233	42.4	229	42.4	301	57.9	242	41.0
2009	January	247	44.9	216	39.3	307	55.8	228	41.5
2009	July	243	49.7	240	49.4	286	69.5	255	48.9
2010	January	233	49.7	225	48.6	262	59.4	214	43.7
2010	July	267	--------	252	--------	294	--------	260	--------
2011	January	256	--------	220	--------	314	--------	244	--------

تم تلخيص هذا الجدول من البيانات الصادرة عن الجهاز المركزي للأحصاء الفلسطيني تحت تقارير شهرية بعنوان المسح الطاقة المنزلي، كما هو مصدر في قائمة المراجع 6 حتى 23

اسعار الكهرباء التاريخية في الخليل



Apparent Power Consumption (MVA) between the years 1957 – 2016 in Hebron district [5]

Standby Consumption

Product/Mode	Average (W)	Min (W)	Max (W)	
Air Conditioner, room/wall				
Off	0.9	0.9	0.9	
Charger, mobile phone				
On, charged	2.24	0.75	4.11	
On, charging	3.68	0.27	7.5	
Power supply only	0.26	0.02	1	
Clock, radio				
On	2.01	0.97	7.6	
Computer Display, CRT				
Off	0.8	0	2.99	
On	65.1	34.54	124.78	
Sleep	12.14	1.6	74.5	
Computer Display, LCD				
Off	1.13	0.31	3.5	
On	27.61	1.9	55.48	
Sleep	1.38	0.37	7.8	
Computer, desktop				
On, idle	73.97	27.5	180.83	
Off	2.84	0	9.21	
Sleep	21.13	1.1	83.3	
Computer, notebook				
Fully on, charged	29.48	14.95	73.1	
Fully on, charging	44.28	27.38	66.9	
Off	8.9	0.47	50	
Power supply only	4.42	0.15	26.4	
Sleep	15.77	0.82	54.8	
Fax, inkjet				
Off	5.31	0	8.72	3
On	6.22	2.89	14	8
Fax, laser				
Off	0	0	0	

On	6.1	6.1	6.1
Ready	6.42	6.42	6.42
Heating, furnace central			
Off	4.21	0	9.8
On	339.71	70.5	796
Hub, USB			
Off	1.44	0.95	1.81
On	2.06	1.06	3.55
Modem, DSL			
Off	1.37	0.33	2.02
On	5.37	3.38	8.22
Modem, cable			
Off	3.84	1.57	6.62
On	6.25	3.64	8.62
Standby	3.85	3.59	4.11
Multi-function Device, inkjet			
Off	5.26	0	10.03
On	9.16	3.9	17.7
Multi-function Device, laser			
Off	3.12	0	4.7
On	49.68	5	175
Night Light, interior			
Off	0.05	0	0.34
On	4.47	0	27.97
Ready	0.22	0	1.2
Phone, cordless			
Ready, handset	2 81	1 05	4 89
Ready, no handset	1.58	0 59	3 09
Active (talking)	1.9	0.59	3.38
Off	0.98	0.54	1.8
Phone, cordless with answering machine			
Ready, handset	4	2.15	7.4
Ready, no handset	2.82	1.72	4.7
Active (talking)	3.53	2.2	6.5

Off	2.92	0.9	7.4
Power Tool, cordless			
Ready, charged	8.34	1.82	14
Active	29.53	1.39	66
Ready	1.74	0	4.7
Printer, inkjet			
Off	1.26	0	4
On	4.93	1.81	22
Printer, laser			
Off	1.58	0	4.5
On	131.07	1.7	481.9
Range, gas			
Ready	1.13	0.7	1.7
Scanner, flatbed			
Off	2.48	0.27	8.2
On	9.6	1.71	15.6
Security Systems, home			
Ready	2.7	2.7	2.7
Set-top Box, DVR			
On, no recording	37.64	25.95	49.2
On, recording	29.29	27.27	31.3
Off	36.68	23.3	48.6
Set-top Box, digital cable with DVR			
Not recording, TV off	44.63	44.38	44.87
Not recording, TV on	44.4	44.2	44.6
Off by remote	43.46	43.3	43.61
Set-top Box, digital cable			
On, TV off	24.65	14.2	74.74
On, TV on	29.64	14.1	102.23
Off by remote	17.83	13.24	30.6
Off by switch	17.5	13.7	26.3
Set-top Box, satellite with DVR			
Not recording, TV off	28.35	25.8	30.9
Not recording, TV on	31.37	24.2	36.3

Off by remote	27.8	22	33.6
Set-top Box, satellite			
On, TV off	15.95	7.69	33.2
On, TV on	16.15	7.69	33.2
Off by remote	15.66	6.58	33.05
Off by switch	15.47	6.58	32.7
Speakers, computer			
On, no sound	4.12	0.69	9.84
Off	1.79	0	5.6
Stereo, portable			
CD, not playing	4.11	1.29	6.83
Cassette, not playing	2.42	1.16	5.92
CD playing	6.8	3.96	9.2
Off	1.66	0.7	5.44
Radio playing	3.3	1.36	8.25
Television, CRT			
Off by remote	3.06	0.3	10.34
Off by switch	2.88	0	16.1
Television, rear projection			
On	186.09	186.09	186.09
Off by remote	6.97	0.2	48.5
Off by switch	6.6	0.2	48.5
Timer, irrigation			
Off	2.75	1.5	5.9
Ready	2.84	1.5	5.9
Tuner, AM/FM			
On, not playing	9.48	5.08	16.4
On, playing	9.92	5.07	17.7
Off	1.12	0	3.37
Amplifier			
On, not playing	33.99	21.4	70.93
On, playing	39.16	21.11	69.3
Off	0.27	0	1.8
Audio Minisystem			

CD, not playing	13.99	1.67	36.95
Cassette, not playing	13.85	1.67	33.14
CD playing	19.09	5.2	41.2
Off	8.32	0.3	24.58
Radio playing	14.41	2.98	38
CD Player			
On, not playing	8.62	4	25.7
On, playing	9.91	5.8	25.6
Off	5.04	2	18.4
Caller ID Unit			
Ready	1.27	1.27	1.27
Cassette Deck			
On, not playing	4.53	4.36	4.7
On, playing	5.72	5.2	6.25
Off	0.54	0	1.08
Clock			
On	1.74	0.99	3.61
Radio playing	2.95	1.7	4.2
Coffee Maker			
Off	1.14	0	2.7
Copier			
Off	1.49	0	2.97
On	9.63	3.6	14
DVD Recorder			
Off	0.75	0	1.5
DVD Player			
On, not playing	7 54	0 24	12.7
On, playing	9.91	5 28	17 17
Off	1.55	0	10.58
DVD/VCR			
On, not playing	13.51	8.48	20.5
On, playing	15.33	9.43	22.37
Off	5.04	0.09	12.7
Game Console			

Active	26.98	5.4	67.68
Off	1.01	0	2.13
Ready	23.34	2.12	63.74
Garage Door Opener			
Ready	4.48	1.8	7.3
Low-voltage Landscape			
Ready	1.13	1.1	1.17
Microwave Ovens			
Ready, door closed	3.08	1.4	4.9
Ready, door open	25.79	1.6	39
Cooking	1433.	966.2	1723.
Musical Instruments			
Off	2.82	1.2	4.2
Receiver (audio)			
On, not playing	37.61	17.1	65.2
Off	2.92	0	19.7
Subwoofer			
On, not playing	10.7	5.8	20.6
On, playing	12.42	5.9	20.6
Surge Protector			
Off	1.05	0	6.3
On	0.8	0	6.92
Telephone Answering Device			
Off	2.01	1.31	2.55
Ready	2.25	1.42	2.83
Television/VCR			
Off by remote	5 15	2.15	13.3
Off by switch	5.99	2 15	13 11
Turntable (audio)			
On, not playing	6.01	1.72	12.8
Off	0.2	0	0.6
VCR			
On, not playing	7.77	3.8	11 62
Off	4.68071	1.2	9.9

Appendix C

Datasheets

LM35 PrecisionCentigradeTemperatureSensors

GeneralDescription

TheLM35seriesareprecisionintegrated-circuittemperature sensors,whoseoutputvoltageislinearlyproportionaltothe Celsius(Centigrade)temperature.TheLM35thushas an advantage overlinear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constantvoltagefromitsoutputtoobtainconvenientCenti- grade scaling. The LM35does not require any external calibrationortrimmingtoprovidetypicalaccuraciesof±¼°C atroomtemperatureand±¾°Covera full−55to+150°C temperaturerange.Lowcostis assuredbytrimmingand calibrationatthewaferlevel.TheLM35'slowoutputimped- ance,linearoutput,and preciseinherentcalibrationmake interfacing toreadoutorcontrolcircuitryespecially easy.It canbeusedwithsingle powersupplies, orwithplusand minussupplies.Asitdrawsonly60µAfromitssupply,ithas verylowself-heating,lessthan0.1°Cinstillair.TheLM35is ratedtooperateovera−55°to+150°Ctemperature range. whiletheLM35Cisratedfora−40°to+110°Crange(−10° withimprovedaccuracy).TheLM35seriesisavailablepack- aged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plasticTO-92transistorpackage.TheLM35Disalsoavail- ableinan8-leadsurfacemountsmalloutlinepackageanda plasticTO-220package.

Features

- Calibrateddirectlyin°Celsius(Centigrade)
- Linear+10.0mV/°Cscalefactor
- 0.5°Caccuracyguaranteeable(at+25°C)
- Ratedforfull−55°to+150°Crange
- Suitableforremoteapplications
- Lowcostduetowafer-leveltrimming
- Operatesfrom4to30volts
- Lessthan60µAcurrentdrain
- Lowself-heating, 0.08°Cinstillair
- Nonlinearity only±¼°C typical
- Lowimpedanceoutput,0.1 for1mAload

TypicalApplications

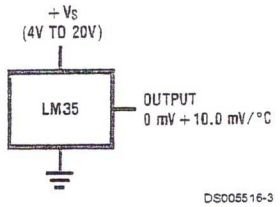
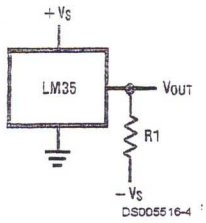


FIGURE1.BasicCentigradeTemperature Sensor (+2°Cto+150°C)

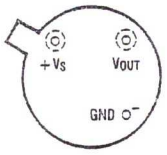


ChooseR1=−VS/50µA
VOUT=+1,500mVat+150°C
=+250mVat+25°C
=−550mVat−55°C

FIGURE2.Full-RangeCentigradeTemperature Sensor

ConnectionDiagrams

TO-46
MetalCanPackage*

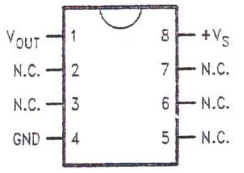


BOTTOM VIEW
DS005516-1

*Caseisconnectedtonegativepin(GND)

OrderNumberLM35H,LM35AH,LM35CH,LM35CAHor LM35DH
SeeNSPackageNumberH03H

SO-8
SmallOutlineMoldedPackage

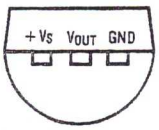


DS005516-21

N.C.=NoConnection

TopView
OrderNumberLM35DM
SeeNSPackageNumberM08A

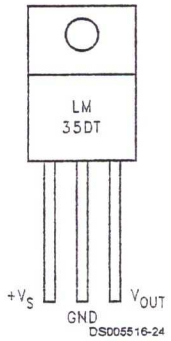
TO-92
PlasticPackage



BOTTOM VIEW
DS005516-2

OrderNumberLM35CZ, LM35CAZorLM35DZ
SeeNSPackageNumberZ03A

TO-220
PlasticPackage*



DS005516-24

*Tabisconnectedtothenegativepin(GND).
Note: TheLM35DTpinoutisdifferentthanthediscontinuedLM35DP.

OrderNumberLM35DT
SeeNSPackageNumberTA03F

AbsoluteMaximumRatings(Note 10)

If Military/Aerospace specified devices are required, pleasecontacttheNationalSemiconductorSalesOffice/ Distributors foravailabilityandspecifications.

SupplyVoltage +35Vto−0.2V
OutputVoltage +6Vto−1.0V
OutputCurrent 10mA
StorageTemp.:
46Package,−60˚Cto+180˚C
−60˚Cto+150˚C
TO-220Package,
LM35C,LM35CA
LM35D0˚Cto+100˚C
SO-8Package,
−65˚Cto+150˚C
LeadTemp.:
TO-46Package,
(Soldering,10seconds) 300˚C
TO-92andTO-220Package,
(Soldering,10seconds) 260˚C
SOPackage(Note12)
VaporPhase(60seconds) 215˚C
Infrared(15seconds) 220˚C
ESDSusceptibility (Note11) 2500V
SpecifiedOperatingTemperature Range:T_{MIN}toT_{MAX}
(Note2)
LM35,LM35A −55˚Cto+150˚C
−40˚Cto+110˚C
TO-92Package,
−65˚Cto+150˚C

ElectricalCharacteristics

(Notes1,6)

Parameter	Conditions	LM35A			LM35CA			Units (Max.)
		Typical	Tested Limit (Note4)	Design Limit (Note5)	Typical	Tested Limit (Note4)	Design Limit (Note5)	
Accuracy (Note7)	T_A=+25˚C	±0.2	±0.5		±0.2	±0.5		˚C
	T_A=−10˚C	±0.3			±0.3		±1.0	˚C
	T_A=T_{MAX}	±0.4	±1.0		±0.4	±1.0		˚C
	T_A=T_{MIN}	±0.4	±1.0		±0.4		±1.5	˚C
Nonlinearity (Note8)	T_{MIN} T_A T_{MAX}	±0.18		±0.35	±0.15		±0.3	˚C
SensorGain (AverageSlope)	T_{MIN} T_A T_{MAX}	+10.0	+9.9, +10.1		+10.0		+9.9, +10.1	mV/˚C
LoadRegulation (Note3)0 I_L 1mA	T_A=+25˚C	±0.4	±1.0		±0.4	±1.0		mV/mA
	T_{MIN} T_A T_{MAX}	±0.5		±3.0	±0.5		±3.0	mV/mA
LineRegulation (Note3)	T_A=+25˚C	±0.01	±0.05		±0.01	±0.05		mV/V
	4V V_S 30V	±0.02		±0.1	±0.02		±0.1	mV/V
QuiescentCurrent (Note9)	V_S=+5V,+25˚C	56	67		56	67		µA
	V_S=+5V	105		131	91		114	µA
	V_S=+30V,+25˚C	56.2	68		56.2	68		µA
	V_S=+30V	105.5		133	91.5		116	µA
Changeof QuiescentCurrent (Note3)	4V V_S 30V,+25˚C	0.2	1.0		0.2	1.0		µA
	4V V_S 30V	0.5		2.0	0.5		2.0	µA
Temperature Coefficientof QuiescentCurrent		+0.39		+0.5	+0.39		+0.5	µA/˚C
MinimumTemperature forRatedAccuracy	Incircuitof Figure1,I_L=0	+1.5		+2.0	+1.5		+2.0	˚C
LongTermStability	T_J=T_{MAX},for 1000hours	±0.08			±0.08			˚C

Datasheet for gas sensor MQ-2

MQ-2 Semiconductor Sensor for Combustible Gas

Sensitive material ofMQ-2gas sensor isSnO_2,which with lower conductivity in clean air. When the targetcombustible gas exist,The sensor'sconductivity is morehigher alongwith the gasconcentration rising. Please use simpleelectrocircuit, Convert change ofconductivitytocorrespond outputsignal of gas concentration.

MQ-2 gas sensor has high sensititytoLPG,Propane and Hydrogen,also could be used toMethane and other combustible steam,it is with lowcostand suitable fordifferentapplication.

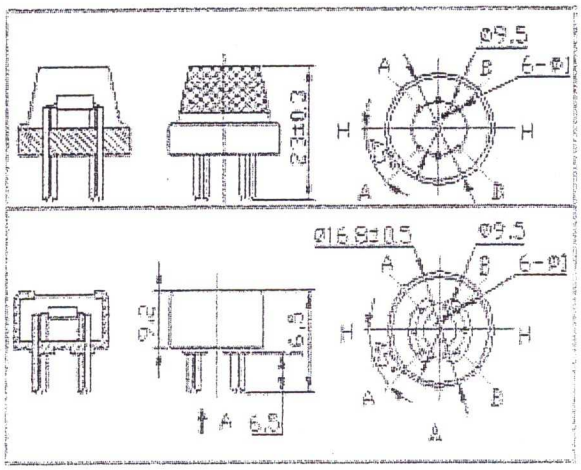
Character

*GoodsensitivitytoCombustiblegasinwiderange
* High sensitivity toLPG,Propane andHydrogen
* Long life and low cost
* Simple drive circuit

Application

* Domestic gas leakage detector
* Industrial Combustible gas detector
* Portablegas detector

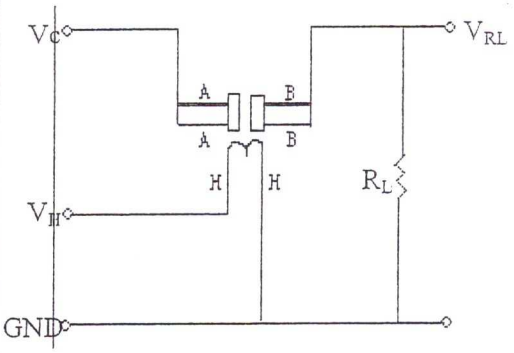
Configuration



TechnicalData

Model No.			MQ-2
SensorType			Semiconductor
StandardEncapsulation			Bakelite(BlackBakelite)
Detection Gas			Combustible gas and smoke
Concentration			300-10000ppm (Combustiblegas)
Circuit	Loop Voltage	V_c	≤24V DC
	Heater Voltage	V_H	5.0V±0.2VACorDC
	Load Resistance	R_L	Adjustable
Character	Heater Resistance	R_H	31Ω±3Ω（RoomTem.）
	Heater consumption	P_H	≤900mW
	Sensing Resistance	R	2KΩ-20KΩ(in2000ppm CH)
	Sensitivity	S	Rs(in air)/Rs(1000ppm isobutane)≥5
	Slope	α	≤0.6($R_{5000ppm}/R_{3000ppm}$ CH_4)
Condition	Tem. Humidity		20℃±2℃；65%±5%RH
	Standardtestcircuit		Vc:5.0V±0.1V； V_H:5.0V±0.1V

Basictestloop



Theaboveisbasictestcircuitofthesensor. Thesensor need tobe put 2 voltage, heatervoltage（VH） andtestvoltage（VC）. VH used tosupply certifiedworking temperature tothe sensor,while VC used todetect voltage (VRL) on load resistance （RL） whomis in serieswith sensor.The sensorhas light polarity, Vcneed DC power. VC andVH could use same power circuitwithprecondition toassure performance of sensor. Inorder tomake the sensorwith better performance, suitable RL value is needed;

SensitivityCharacteristics

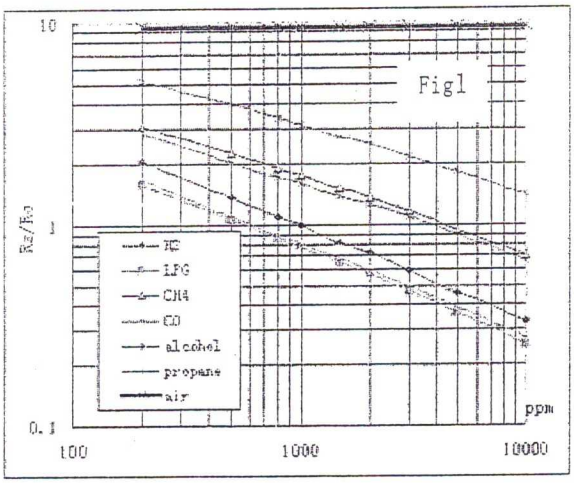


Fig.1shows thetypical sensitivitycharacteristics of the MQ-2,ordinate means resistance ratio of the sensor (Rs/Ro), abscissa is concentrationofgases.Rs means resistancein differentgases,Ro means resistanceof sensorin 1000ppm Hyrogen.All testare under standard testconditions.1000ppm Methane,20℃/65%RH

Influence of Temperature/Humidity

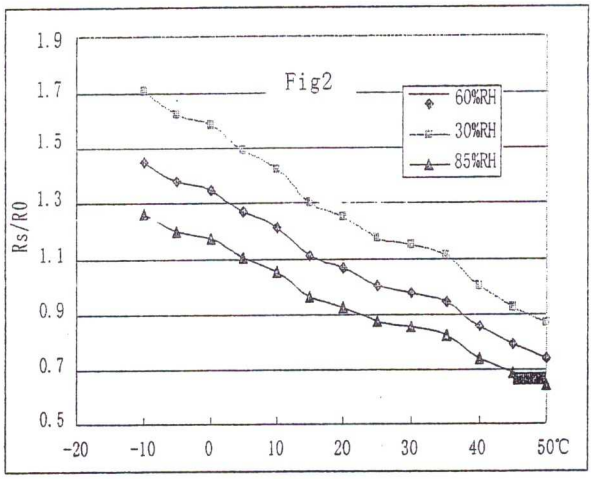
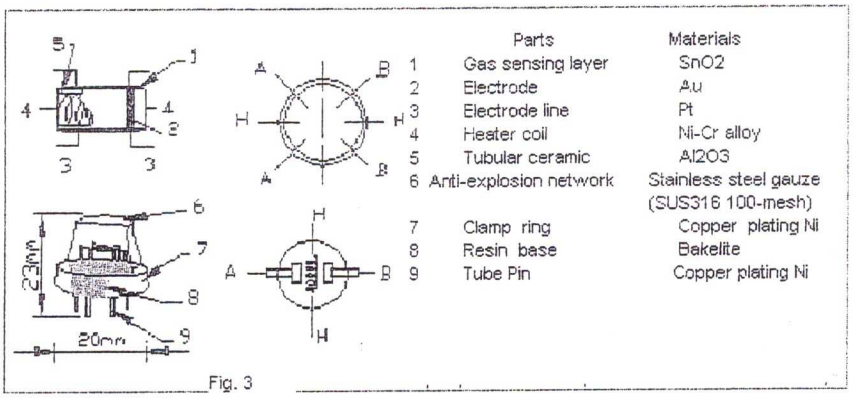


Fig.2shows the typical temperature and humidity characteristics. Ordinatemeans resistance ratio ofthe sensor (Rs/Ro),Rs means resistanceofsensor in 1000ppm Butaneunder different tem.and humidity. Ro means resistance of the sensor inenvironment of

Structureandconfiguration



	Parts	Materials
1	Gas sensing layer	SnO2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al2O3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

Fig. 3

StructureandconfigurationofMQ-2gassensorisshownasFig. 3,sensorcomposedbymicroAL2O3ceramictube,Tin Dioxide(SnO2)sensitivelayer,measuringelectrodeandheaterarefixedintoacrust madebyplasticandstainlesssteel net.Theheater providesnecessaryworkconditionsforworkofsensitivecomponents.TheenvelopedMQ-2have6pin,4 ofthem are used tofetchsignals, and other 2 are used forproviding heating current.

Notification

1 Followingconditionsmust be prohibited

1.1 Exposedtoorganic silicon steam

Organicsiliconsteamcausesensorsinvalid,sensorsmustbeavoidexposingtosiliconbond, fixature, silicon latex, puttyor plasticcontain siliconenvironment

1.2 High Corrosive gas

Ifthesensorsexposedtohighconcentrationcorrosivegas(suchasH_2Sz,SO_X, Cl_2, HCletc),itwill not only result incorrosion ofsensorsstructure,also it causesincere sensitivity attenuation.

1.3Alkali,Alkali metalssalt,halogen pollution

Thesensorsperformancewillbechangedbadlyifsensorsbesprayedpollutedbyalkalimetalssalt especially brine, or be exposed tohalogen such asfluorin.

1.4Touchwater

Sensitivityofthe sensors will be reduced when spattered or dipped in water.

1.5 Freezing

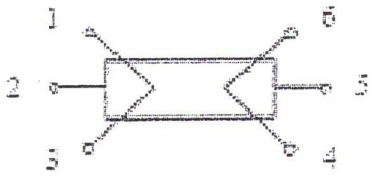
Do avoid icing on sensor'surface, otherwise sensor would lose sensitivity.

1.6Applied voltage higher

Appliedvoltageonsensorshouldnot behigherthanstipulatedvalue,otherwiseitcause down-lineor heater damaged, and bring on sensors'sensitivity characteristicchanged badly.

1.7 Voltageon wrong pins

For6pinssensor,ifapplyvoltageon1、3pinsor4、6pins,it willmakeleadbroken,andwithoutsignalwhenapplyon2、4pins



2 Followingconditionsmust be avoided

2.1 WaterCondensation

Indoorconditions,slightwatercondensationwilleffectsensorsperformancelightly.However,ifwater condensation on sensors surfaceand keep a certainperiod, sensor'sensitivity will be decreased.

2.2 Used in high gasconcentration

Nomatterthesensoriselectrifiedornot,iflongtimeplacedinhighgasconcentration,ifwillaffect sensors characteristic.

2.3 Long time storage

Thesensorsresistanceproducereversibledriftifit'sstoredforlongtime withoutelectrify,thisdriftis relatedwithstorageconditions.Sensorsshouldbestoredinairproofwithoutsilicongelbagwithcleanair. Forthesensorswithlongtimestoragebutnoelectrify,theyneedlongagingtimeforstbilitybeforeusing.

2.4 Long time exposed toadverse environment

Nomatterthesensorselectrifiedornot,ifexposedtoadverseenvironmentforlongtime,suchas high humidity, high temperature, or high pollution etc,it will effectthesensors performance badly.

2.5 Vibration

Continualvibrationwillresultinsensorsdown-leadresponsethenrepture.Intransportationor assembling line, pneumaticscrewdriver/ultrasonicwelding machinecan lead this vibration.

2.6 Concussion

Ifsensors meet strong concussion, itmay lead itslead wire disconnected.

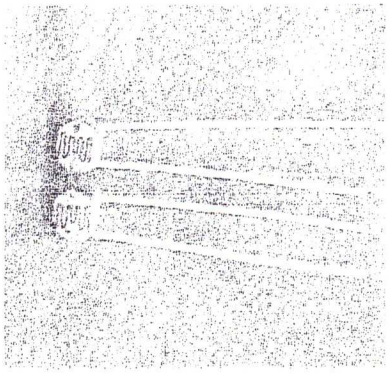
2.7 Usage

Forsensor,handmadeweldingisoptimalway.Ifusewavecrestweldingshouldmeetthefollowing conditions:

2.7.1 Soldering flux:Rosin soldering flux contains least chlorine
2.7.2 Speed:1-2 Meter/Minute
2.7.3 Warm-uptemperature：100±20℃
2.7.4 Weldingtemperature：250±10℃
2.7.5 1 time pass wave crestwelding machine

Ifdisobey the above using terms,sensors sensitivity will be reduced.

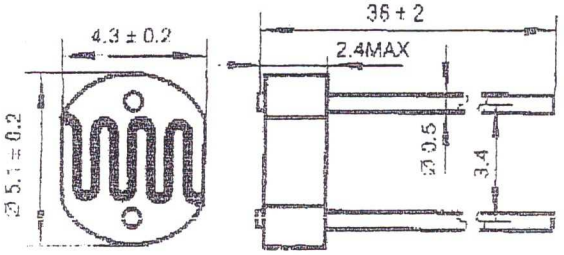
CdSPHOTOCONDUCTIVE GL5528



- ▲Epoxy encapsulated
- ▲Quick response
- ▲Small size
- ▲High sensitivity
- ▲Reliable performance
- ▲Good characteristic of spectrum

Light Resistance at 10Lux (at 25℃)	8~20KΩ
Dark Resistance at 0 Lux	1.0MΩ(min)
Gamma value at 100-10Lux	0.7
Power Dissipation(at 25℃)	100mW
Max Voltage (at 25℃)	150V Spectral
Response peak (at 25℃)	540nm
Ambient Temperature Range:	- 30~+70℃

Outline



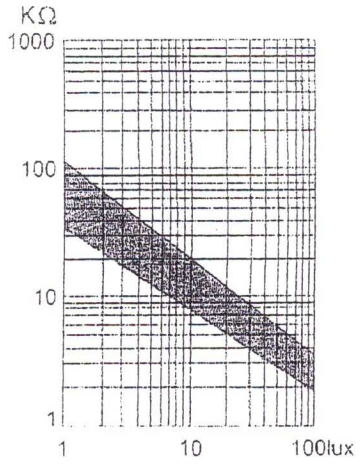
Measuring Conditions

1. Light Resistance:
   measured at 10 lux with standard light A (2854k color temperature) and 2h pre-illumination at 400-600 lux prior to testing.
2. Dark Resistance:
   measured 10 seconds after pulsed 10 lux.
3. Gamma Characteristic:
   between 10 lux and 100 lux and given by

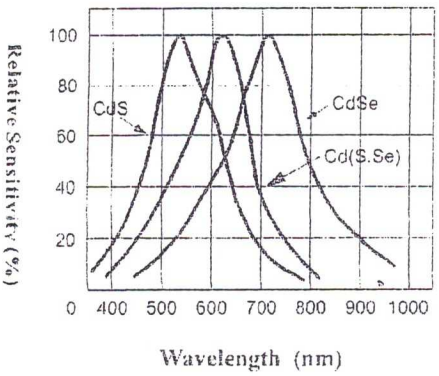
   $$T = \frac{\log (R10/R100)}{\log (100/10)} = \log (R10/R100)$$

   R10, R100 cell resistance at 10 lux and 100 lux.
   The error of T is +0.1.
4. Pmax:
   Max. power dissipation at ambient temperature of 25℃.
5. Vmax:
   Max. voltage in darkness that may be applied to the cell continuously.

IlluminanceVs. Photo Resistance



Spectral Response



LIDA OPTICAL&ELECTRONIC CO., LTD.

254,Zhongzhou road, Nanyang, Henan,P.R.C
TEL: +86-377-6313 0034
FAX: +86-377-6315 2372
E-mail: sale@nylida.com
Http://www.nylida.com

Datasheet for xbee module

XBee®/XBee-PRO®ZBRFModules

ZigBeeRFModulesbyDigiInternational

Firmware versions:

-20xx- Coordinator-AT/TransparentOperation

-21xx- Coordinator-APIOperation

-22xx- Router- AT/TransparentOperation

-23xx- Router- APIOperation

-28xx- EndDevice-AT/TransparentOperation

-29xx- EndDevice-APIOperation

1. Overview

TheXBee/XBee-PROZBRFModulesaredesignedto operatewithintheZigBeeprotocolandsupporttheunique needsoflow-cost,low-powerwirelesssensornetworks. Themodulesrequireminimalpowerandprovidereliable deliveryofdatabetweenremotedevices.

ThemodulesoperatewithintheISM2.4GHzfrequency bandandarecompatiblewiththefollowing:

- XBeeRS-232Adapter
- XBeeRS-232PH(PowerHarvester)Adapter
- XBeeRS-485Adapter
- XBeeAnalogI/OAdapter
- XBeeDigitalI/OAdapter
- XBeeSensorAdapter
- XBeeUSBAdapter
- XStick
- ConnectPortXGateways
- XBeeWallRouter.

TheXBee/XBee-PROZBfirmwarereleasecanbeinstalledonXBeeZNetorZBmodules.TheXBee ZBfirmwareisbasedontheEmberZNet3.xZigBeePROFeatureSetmeshnetworkingstack,while theXBeeZNet2.5firmwareisbasedonEmber'sproprietary"designedforZigBee"meshstack (EmberZNet2.5.x).ZBandZNet2.5firmwarearesimilarinnature,butnotover-the-air compatible.DevicesrunningZNet2.5firmwarecannottalktodevicesrunningtheZBfirmware.

KeyFeatures

HighPerformance,LowCost

XBee

- Indoor/Urban:upto133'(40m)
- Outdoorline-of-sight:upto400'(120m)
- TransmitPower:2 mW(3dBm)
- ReceiverSensitivity:-96dBm

XBee-PRO

- Indoor/Urban:upto300'(90m),200'(60 m)forInternationalvariant
- Outdoorline-of-sight:upto1mile(1600 m),2500'(750m)forInternationalvariant
- TransmitPower:50mW(17dBm),10mW (10dBm)forInternationalvariant
- ReceiverSensitivity:-102dBm

AdvancedNetworking&Security

RetriesandAcknowledgements

DSSS(DirectSequenceSpreadSpectrum)

Eachdirectsequencechannelhasover 65,000uniquenetworkaddressesavailable

Point-to-point,point-to-multipoint andpeer-to-peertopologiessupported

Self-routing,self-healingandfault-tolerant meshnetworking

LowPower

XBee

- TXPeakCurrent:40mA(@3.3V)
- RXCurrent:40mA(@3.3V)
- Power-downCurrent:<1uA

XBee-PRO

- TXPeakCurrent:295mA(170mAfor internationalvariant)
- RXCurrent:45mA(@3.3V)
- Power-downCurrent:<10uA

Easy-to-Use

Noconfigurationnecessaryforout-ofbox RFcommunications

ATandAPICommandModesfor configuringmoduleparameters

Smallformfactor

Extensivecommandset

FreeX-CTUSoftware (Testingandconfigurationsoftware)

Free& UnlimitedTechnicalSupport

WorldwideAcceptance

FCCApproval(USA)RefertoAppendixAforFCCRequirements.

SystemsthatcontainXBee®/XBee-PRO®ZBRFModulesinheritDigiCertifications.
ISM (Industrial,Scientific& Medical)**2.4GHzfrequencyband**

Manufacturedunder**ISO9001:2000**registeredstandards

XBee®/XBee-PRO®ZBRFModulesareoptimizedforuseIn**US,Canada,Australia, IsraelandEurope**(contactMaxStreamforcompletelistofagencyapprovals).

Specifications

Table1-01.SpecificationsoftheXBee®/XBee-PRO®ZBOEMRFModule

[illegible]	XBee	XBee-PRO
Performance		
Indoor/UrbanRange	upto 133ft.(40m)	Upto300ft.(90m),upto200ft(60m)international variant
OutdoorRFline-of-sightRange	upto 400ft.(120m)	Upto1mile(1600m),up to2500ft(750m) internationalvariant
TransmitPowerOutput	2mW(+3dBm),boostmodeenabled 1.25mW(+1dBm),boostmodedisabled	50mW(+17dBm) 10mW(+10dBm)forInternationalvariant
RF DataRate	250,000bps	250,000bps
SerialInterfaceDataRate (softwareselectable)	1200- 230400bps (non-standardbaudratesalsosupported)	1200- 230400bps (non-standardbaudratesalso supported)
ReceiverSensitivity	-98dBm,boostmodeenabled -95dBm,boostmodedisabled	-102dBm

Table1-01.SpecificationsoftheXBee®/XBee-PRO®ZBOEMRFModule

Specification	XBee	XBee-PRO
PowerRequirements		
SupplyVoltage	2.1- 3.6V	3.0- 3.4V
OperatingCurrent(Transmit,max outputpower)	40mA(@ 3.3V, boostmodeenabled) 35mA(@ 3.3V, boostmodedisabled)	295mA(@3.3V),170mA(@3.3V)international variant
OperatingCurrent(Receive))	40mA(@ 3.3V, boostmodeenabled) 38mA(@ 3.3V, boostmodedisabled)	45mA(@3.3V)
IdleCurrent(Receiveroff)	15mA	15mA
Power-downCurrent	<1 uA@25°C	<10uA@ 25°C
General		
OperatingFrequencyBand	ISM2.4GHz	ISM2.4GHz
Dimensions	0.960"x1.087"(2.438cmx 2.761cm)	0.960x1.297(2.438cmx3.294cm)
OperatingTemperature	-40to85°C(industrial)	-40to85°C(industrial)
AntennaOptions	IntegratedWhip,Chip,RPSMA,or U.FLConnector*	IntegratedWhip,Chip,RPSMA,or U.FLConnector*
Networking& Security		
SupportedNetwork Topologies	Point-to-point,Point-to-multipoint,Peer-to-peer,and Mesh	Point-to-point,Point-to-multipoint,Peer-to-peer,and Mesh
NumberofChannels	16DirectSequenceChannels	14DirectSequenceChannels
AddressingOptions	PANIDand Addresses,ClusterIDsandEndpoints (optional)	PANIDandAddresses,ClusterIDsandEndpoints (optional)
AgencyApprovals		
UnitedStates(FCCPart15 247)	FCCID:OUR-XBEE2	FCCID:MCQ-XBEEPRO2
IndustryCanada(IC)	IC:4214A-XBEE2	IC:1846A-XBEEPRO2
Europe(CE)	ETSI	ETSI
Australia	C-Tick	C-Tick
Japan	R201WW07215214	R201WW08215142
RoHS	Compliant	Compliant

MechanicalDrawings

Figure1-01.MechanicaldrawingsoftheXBee®/XBee-PRO®ZBOEMRFModules(antennaoptionsnotshown)

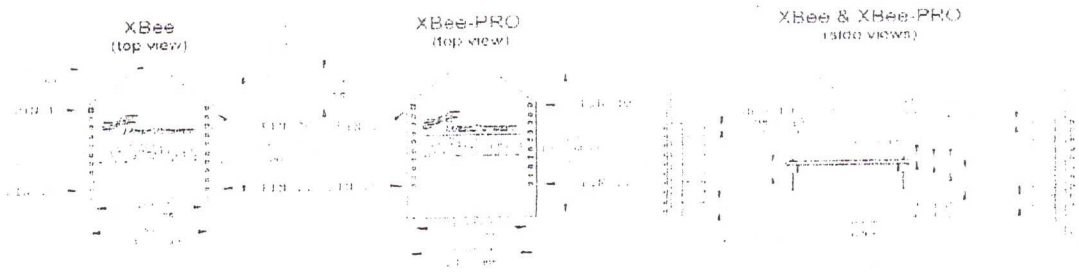
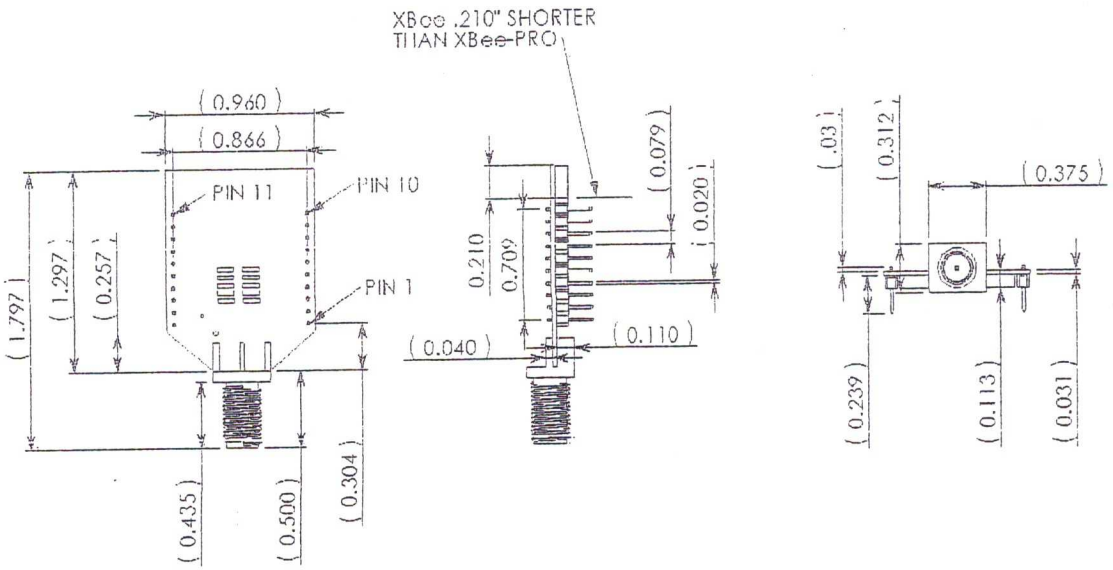


Figure1-02.MechanicalDrawingsfortheRPSMAVariant



Datasheet for PIRSensor

Features

- 2-stageoperationalamplifierasfilter
- Built-innoiserejectioncircuit
- On-chipregulator
- Overridefunction
- SynchronouswithAC220V/50Hzand110V/60Hz
- Pulseoutput(PT8A261)forTRIACdriveor leveloutput(PT8A262)forrelaydrive
- CDStoenable/disableoutput
- Adjustableoutputontimeduration
- ON/AUTO/OFFselectablebyMODEpin
- Auto-resetiftheZCsignaldisappearsover3 seconds
- 40secondswarm-up
- Quickcheckmodeforinitialinstallation
- Operatingvoltage:5V
- Stand-bycurrent:80μA

GeneralDescription

ThePT8A261andPT8A262are lowpower mixedsignal CMOSLSIdesignedfortheautomaticlampcontrolusing PIRsensorasmotiondetector.

Withon chipnoisefilterand voltageregulator,theIC providesstable operationsthroughouttemperature range.CDSinputto disabledaytime operationis desired.

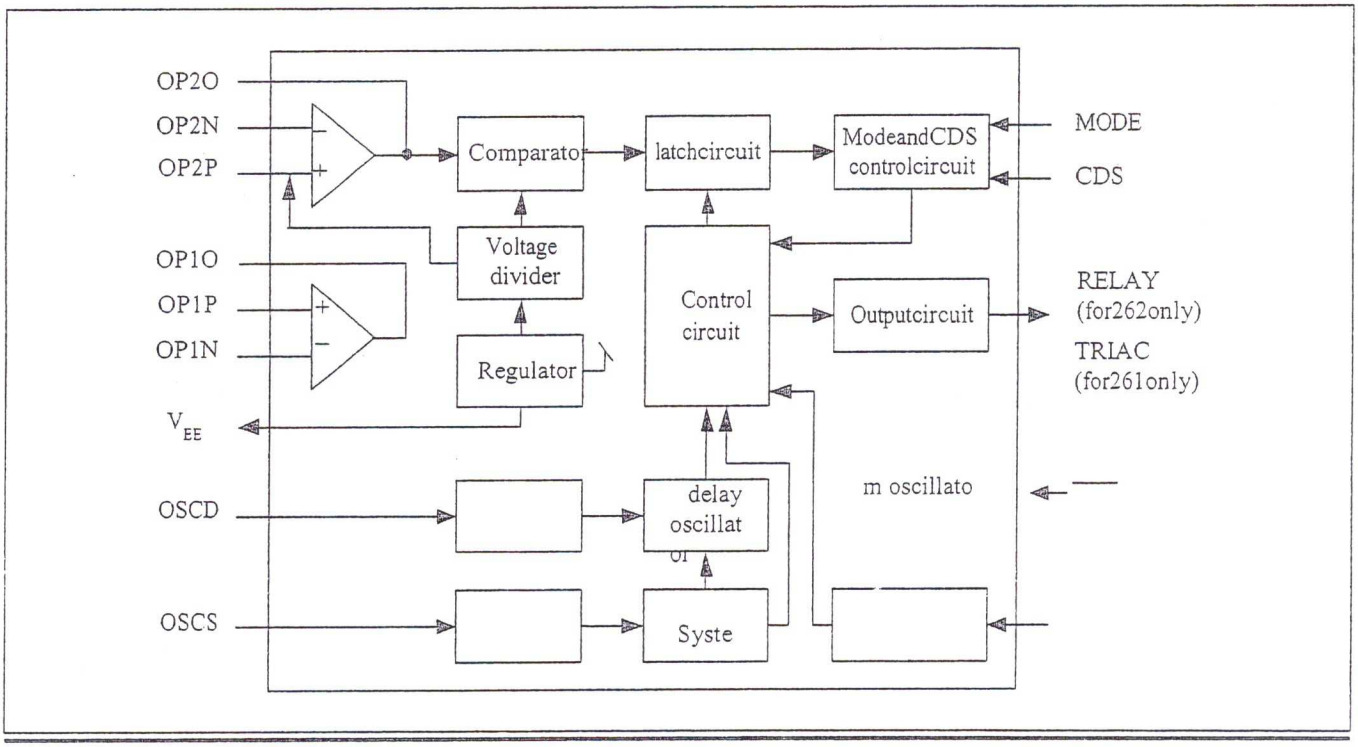
Applications

- Energysavingauto-switchinGarden,kitchen, bathroom,corridor,storageoryard
- Autolightinmeetingroom

OrderingInformation

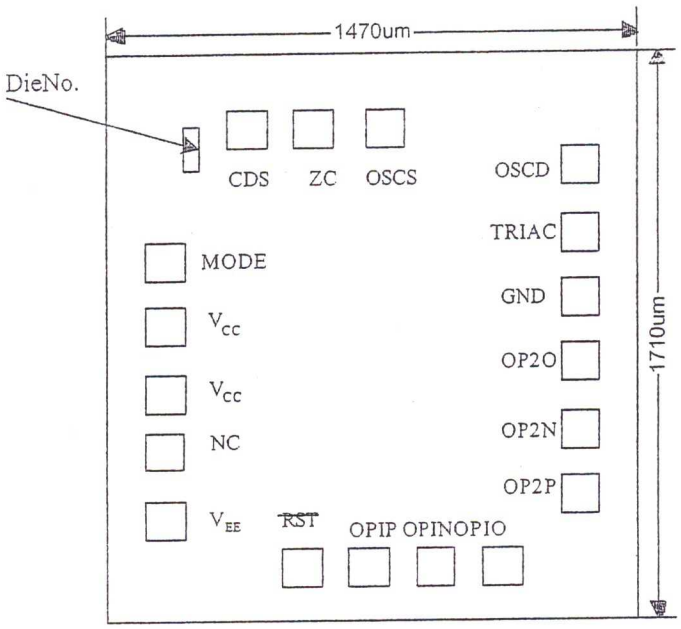
PartNumber	Package
PT8A261P/PT8A262P	16-PinPDIP
PT8A261W/PT8A262W	16-PinSOP
PT8A261DE/PT8A262DE	DieForm

BlockDiagram



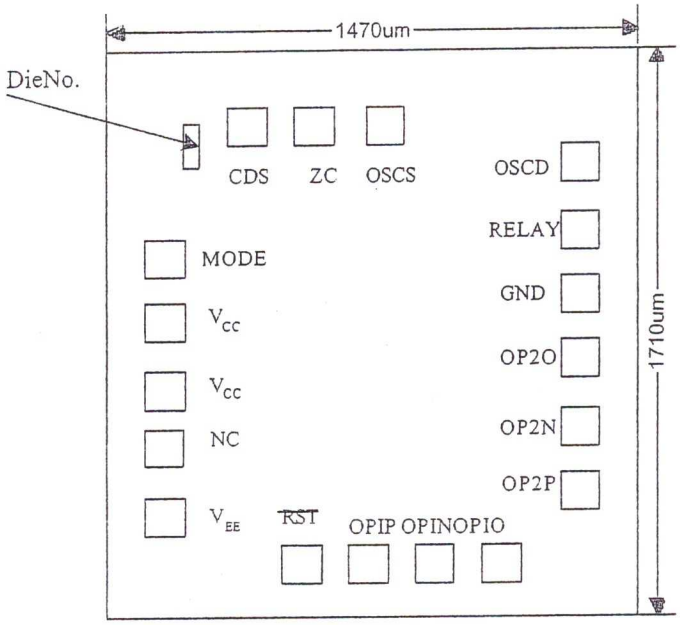
PadLocation

PadLocationofPT8A261DE



PadCoordinate					
PadName	XCoordinate	YCoordinate	PadName	XCoordinate	YCoordinate
GND	570	65.4	NC	570	-268.6
TRIAC	570	-101.6	V_{EE}	-570	232.4
OSCD	570	399.4	RST	303.9	-687.2
OSCS	570	232.4	OP1P	-44.8	689.2
ZC	-378.8	689.2	OP1N	-211.8	689.2
CDS	-30.1	-687.2	OP1O	-570	-435.6
MODE	570	-435.6	OP2P	-570	-268.6
V_{CC}	-197.1	-687.2	OP2N	-570	-101.6
V_{CC}	136.9	-687.2	OP2O	-570	65.4
Note:SubstrateisconnectedtoGND					

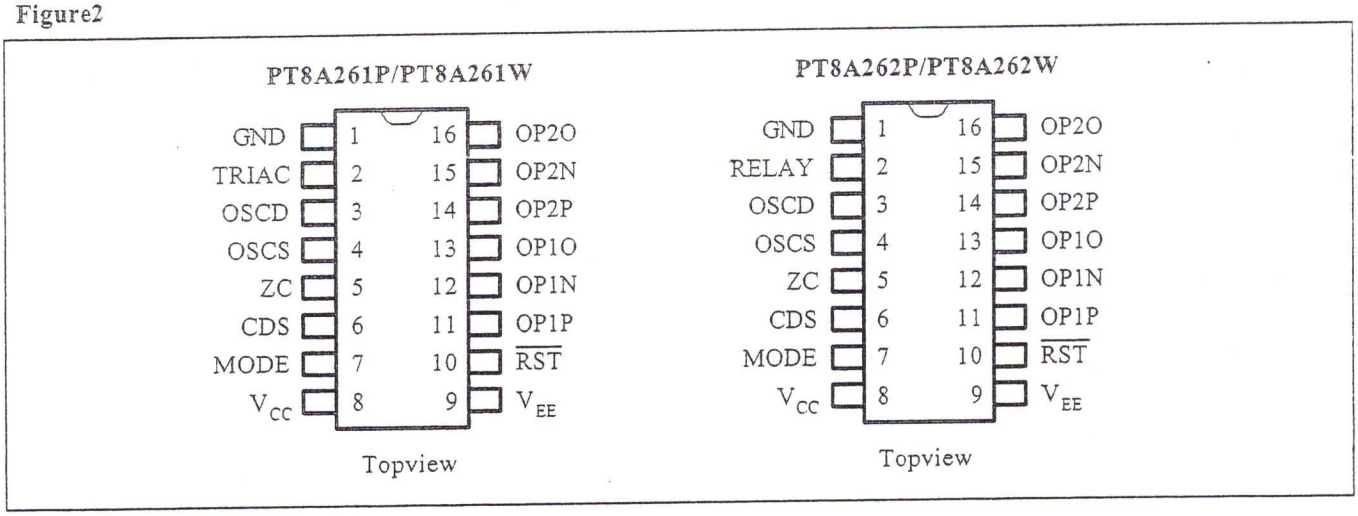
PadLocationofPT8A262DE



PadCoordinate					
PadName	XCoordinate	YCoordinate	PadName	XCoordinate	YCoordinate
GND	570	65.4	NC	570	-268.6
RELAY	570	-101.6	V_{EE}	-570	232.4
OSCD	570	399.4	RST	303.9	-687.2
OSCS	570	232.4	OP1P	-44.8	689.2
ZC	-378.8	689.2	OP1N	-211.8	689.2
CDS	-30.1	-687.2	OP1O	-570	-435.6
MODE	570	-435.6	OP2P	-570	-268.6
V_{CC}	-197.1	-687.2	OP2N	-570	-101.6
V_{CC}	136.9	-687.2	OP2O	-570	65.4
Note:SubstrateisconnectedtoGND					

PinAssignment

Figure2



PinDescription

Table1

Pin/Pad		Name	Type	Description
261	262			
1	1	GND	Ground	Ground
	2	RELAY	O	RELAYdriveoutputthroughanexternalNPNtransistor,activehigh
2		TRIAC	O	TRIACdrivetwo-pulseoutput,activenegativepulse.
3	3	OSCD	I/O	OuputtimingoscillatorI/O,connectedtoanexternalRCtoadjustoutputduration.
4	4	OSCS	I/O	SystemoscillatorI/O,connectedtoanexternalRCtosetthesystemfrequency.Thesystem frequency=16kHzfornormalapplication.
5	5	ZC	I	SchmittinputforACzerocrossingdetection.
6	6	CDS	I	ConnectedtoaCDSvoltagedividerfordaytime/nightauto-detection.Lowinputtothispin candisablePIRinput.CDSisaschmitttriggerinputwith5-secondinputdebouncetime.
7	7	MODE	I	Modeselect,connectingtoV -outputalwayson,connectionGND-outputalwaysoff, open-autodetection
8	8	V_{CC}	Power	Powersupply
9	9	V_{EE}	O	Internalvoltageregulatoroutput,3.6Vwithrespecttoground.Connectedtothedrainof PIRsensor
10	10	$\overline{RST}$	I	Chipresetinput,activelow,keptfloatingorconnectedanRCnetwork
11	11	OP1P	I	Non-invertedinputoffirstoperationalamplifier,connecteddirectlytosourceofPIRsensor
12	12	OP1N	I	Invertedinputoffirstoperationalamplifier
13	13	OP1O	O	Outputoffirstoperationalamplifier
14	14	OP2P	I	Non-invertedinputofsecondoperationalamplifier.
15	15	OP2N	I	Invertedinputofsecondoperationalamplifier
16	16	OP2O	O	Outputofsecondoperationalamplifier

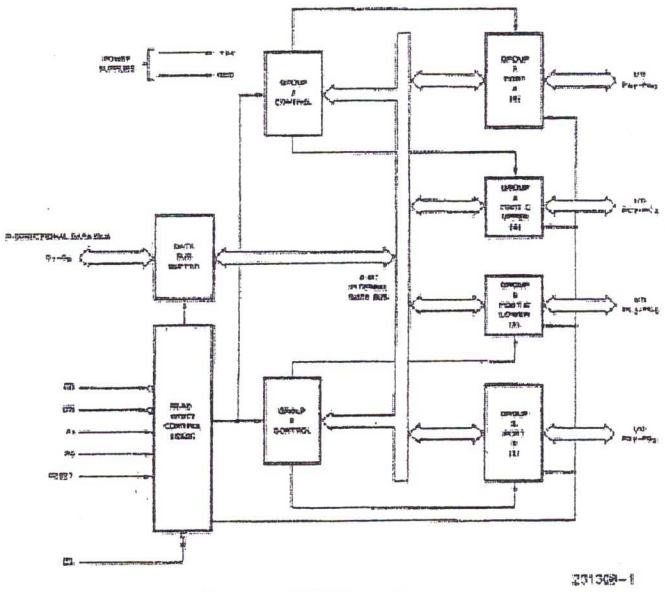
intel

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

- MCS-85™ Compatible 8255A-5
- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with Intel Microprocessor Families
- Improved Timing Characteristics
- Direct Bit Set/Reset Capability Easing Control Application Interface
- Reduces System Package Count
- Improved DC Driving Capability
- Available in EXPRESS
  - Standard Temperature Range
  - Extended Temperature Range
- 40 Pin DIP Package

  (See Intel Packaging: Order Number: 240800-001, Package Type P)

The Intel 8255A is a general purpose programmable I/O device designed for use with Intel microprocessors. It has 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. In the first mode (MODE 0), each group of 12 I/O pins may be programmed in sets of 4 to be input or output. In MODE 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining 4 pins, 3 are used for handshaking and interrupt control signals. The third mode of operation (MODE 2) is a bidirectional bus mode which uses 8 lines for a bidirectional bus, and 5 lines, borrowing one from the other group, for handshaking.



231308-1

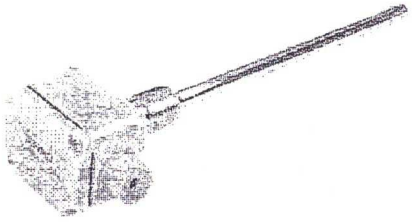
Figure 1. 8255A Block Diagram

231308-2

Figure 2. Pin Configuration

APPLICATION

TSCseriesofNTCtemperature sensorisonekindof TSseries temperaturesensor.Ithastheadvantagesof highsensitivity,high stability, erosionresistance,longservicelifeandconvenientinstallation, etc. It can measure the temperature of air or water quickly and accuratelyinHVACapplication,andsendthesignalsto thecontrol system soastocontrolthetemperatureofairorwateraccurately.Not onlyeachpart butalsothewholeassemblyof thisseriesproducthave precisetechniquesandgoodqualitycontrol,andhave been passed through strictly testing.



(Fig. 1)

CHARACTERISTICS

TEMPERATURESENSITIVEELEMENT:ImportedNTC temperaturesensitive element

WORKINGRANGE:0~50℃

MAX.TEMPERATUREFORTHETERMINALBOX:70℃

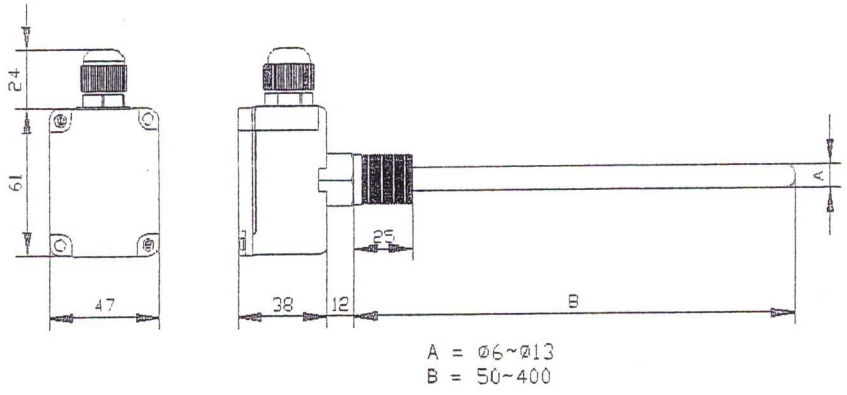
MATERIALOFTHETERMINALBOX:High intensions fireproof PC engineering plastic

INSTALLATIONMODE:Screw connection, inserted mode

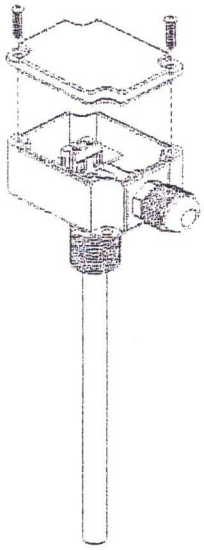
MATERIALOFWIRINGTERMINAL:FireproofABSengineering plastic

PROTECTIONCLASS: IP54

ROD MATERIAL:Brass (Ni-plated on the surface)



(Fig.2)



(Fig.3)

SPECIFICATIONS(SeethefollowingdatasheetandFig.2)

MODEL	SENSITIVE ELEMENT	TEMPERATURE RANGE	ROD DIAMETER **A**	ROD LENGTH **B**
TSC-8112-103B39	NTC 10K	0~120℃	Φ8	120
TSC-8118-103B39				180
TSC-8212-103B39			Φ10	120
TSC-8218-103B39				180

INSTALLATIONINSTRUCTION

Becausethesensitivityofthisseriessensorisveryhigh,thesensormustbeinstalledinthemostsuitable

placesoastogetthebestefficiency. Inordernottointerferethesensor,Itissuggestedthat theconnecting wireshouldbe2-conductorshieldwire,laidinseparatewirechaseandshouldnotexceed50m.Itmust be assuredthat theterminalofthesensorisnotincontactwithanyobjectssoastoavoidthedamageofthe sensor or wrong operation. Moredetails please refer tothe InstallationDiagram (Fig. 3).

RELATIONS BETWEEN TEMPERATUREAND RESISTANCE OF TSC-8×××-103B39

TEMPERATURE ℃	RESISTANCE Ω	TEMPERATURE ℃	RESISTANCE Ω	TEMPERATURE ℃	RESISTANCE Ω
0	32600	17	14318.0	34	6810.50
1	30985	18	13676.9	35	6534.00
2	29459	19	13068.1	36	6270.00
3	28017	20	12489.8	37	6018.00
4	26654	21	11940.3	38	5778.00
5	25365	22	11418.0	39	5548.70
6	24145.2	23	10921.4	40	5330.00
7	22991.4	24	10449.2	41	5120.00
8	21899.2	25	10000.0	42	4920.00
9	20865.2	26	9572.60	43	4729.00
10	19885.8	27	9165.80	44	4547.00
11	18957.9	28	8778.50	45	4372.00
12	18078.6	29	8409.60	46	4205.00
13	17245.0	30	8058.30	47	4045.00
14	16454.5	31	7723.60	48	3893.00
15	15704.7	32	7404.60	49	3746.30
16	14993.2	33	7100.58	50	3606.00

*All the above data will be changedwithout prior notice.
*For other models please refer toSENSOR MODELDESCRIPTION.

TECHNICAL DATA MQ-7 GASSENSOR

FEATURES

*Highsensitivitytocarbon monoxide

*Stableandlonglife

APPLICATION

Theyareusedin gas detectingequipmentforcarbon monoxide(CO)infamilyand industry orcar.

SPECIFICATIONS

A. Standardwork condition

Symbol	Parametername	Technical condition	Remark
Vc	circuitvoltage	5V±0.1	AcorDc
V_H(H)	Heating voltage(high)	5V±0.1	AcorDc
V_H(L)	Heating voltage(low)	1.4V±0.1	AcorDc
R_L	Loadresistance	Can adjust	
R_H	Heatingresistance	33Ω±5%	Roomtemperature
$T_{H(H)}$	Heating time(high)	60±1 seconds	
$T_{H(L)}$	Heating time(low)	90±1 seconds	
PH	Heating consumption	About350mW	

b. Environmentconditions

Symbol	Parameters	Technicalconditions	Remark
Tao	Usingtemperature	-20□-50□	
Tas	Storagetemperature	-20□-50□	Advice usingscope
RH	Relativehumidity	Less than95%RH	
O_2	Oxygen concentration	21%(standcondition) theoxygen concentration can affectthesensitivity characteristic	Minimumvalueisover2%

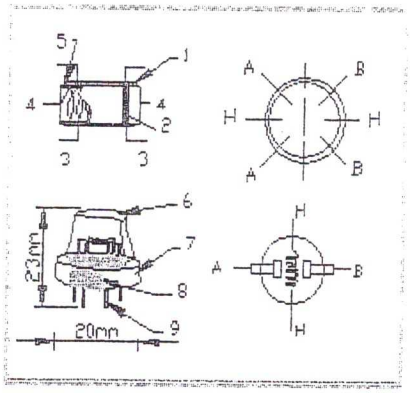
c. Sensitivity characteristic

symbol	Parameters	Technicalparameters	Remark
Rs	Surfaceresistance Ofsensitivebody	2-20k	In100ppm CarbonMonoxide
a(300/100ppm)	Concentrationsloperate	Less than0.5	Rs(300ppm)/Rs(100ppm)
Standardworking condition	Temperature-20□±2□ relativehumidity65%±5% RL:10KΩ±5%		
	Vc:5V±0.1V VH:5V±0.1V VH:1.4V±0.1V		
Preheat time	Noless than48hours	Detectingrange: 20ppm-2000ppmcarbon monoxide	

D. Structureandconfiguration, basicmeasuringcircuit

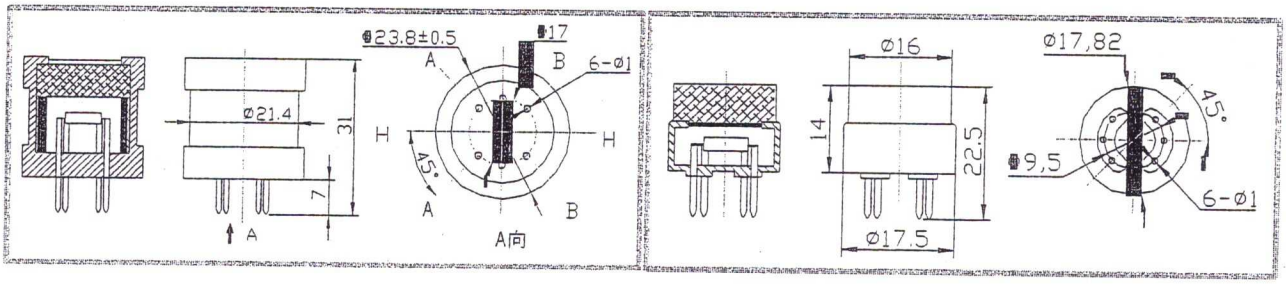
StructureandconfigurationofMQ-7gassensorisshownasFig.1(ConfigurationA orB), sensorcomposedbymicroAL_2O_3 ceramictube,TinDioxide(SnO_2)sensitivelayer,measuring electrodeandheaterarefixed into acrustmadeby plasticand stainlesssteelnet.Theheater providesnecessaryworkconditionsforworkofsensitivecomponents.Theenveloped MQ-7have

6 pin,4 ofthemareusedtofetchsignals, andother2 areused forproviding heatingcurrent.



	Parts	Materials
1	Gassensing layer	SnO_2
2	Electrode	Au
3	Electrodeline	Pt
4	Heatercoil	Ni-Cralloy
5	Tubularceramic	Al_2O_3
6	Anti-explosion network	Stainlesssteel gauze (SUS316100-mesh)
7	Clamp ring	Copper platingNi
8	Resin base	Bakelite
9	TubePin	CopperplatingNi

Fig.1



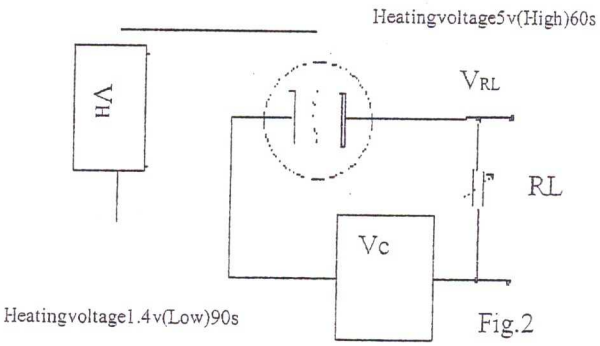


Fig.2

Standardcircuit:

As shown inFig 2,standardmeasuringcircuitof MQ-7sensitivecomponentsconsistsof 2parts.oneisheatingcircuithavingtimecontrolfunction (the high voltage and the low voltage work circularly).Thesecondisthesignaloutputcircuit,itcan accurately respond changesofsurface resistanceofthesensor.

Fig.2

Electricparametermeasurementcircuit is shownas

E. Sensitivitycharacteristiccurve

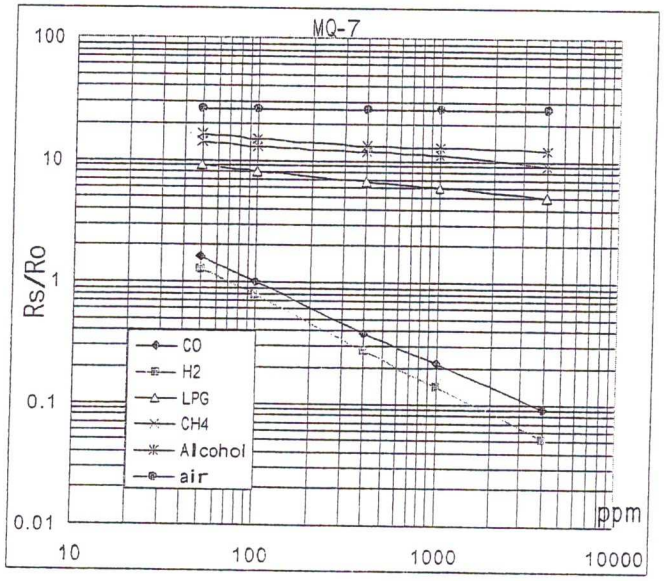


Fig.3 is shows thetypical sensitivitycharacteristicsof theMQ-7 forseveralgases.

intheir: Temp: 20□、

Humidity:65%、

O_2concentration21%

RL=10kΩ

Ro:sensorresistanceat100ppm COinthecleanair.

Rs:sensorresistanceatvarious concentrationsofgases.

Fig.3sensitivitycharacteristicsoftheMQ-7

TEL:86-371-67169070 67169080 FAX:86-371-67169090 Email: sales@hwsensor.com

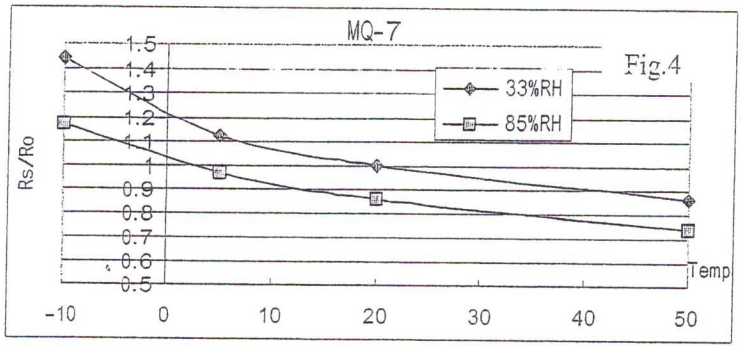


Fig.4

Fig.4 isshows thetypical dependenceoftheMQ-7 on temperatureand humidity. Ro:sensorresistanceat100ppmCOin airat 33%RHand20degree. Rs:sensorresistanceat100ppmCOat differenttemperaturesandhumidities.

OPERATIONPRINCIPLE

.ThesurfaceresistanceofthesensorRsisobtainedthrougheffectedvoltagesignaloutputofthe loadresistanceRLwhichseries-wound. Therelationshipbetweenthemisdescribed:

Rs\RL=(Vc-VRL)/VRL

Fig. 5 shows alterablesituationofRLsignaloutputmeasuredbyusing Fig. 2 circuitoutput

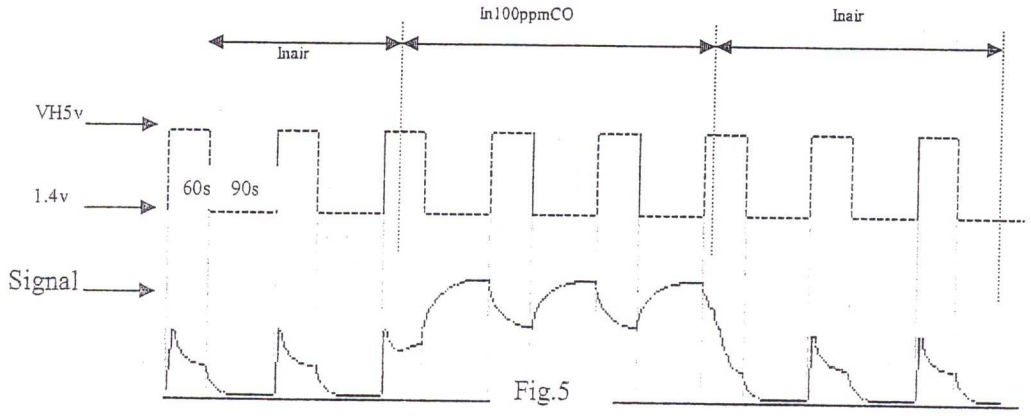


Fig.5

signalwhen thesensoris shiftedfromclean airto carbonmonoxide (CO),outputsignal measurementismadewithin oneortwocompleteheatingperiod(2.5minutefromhighvoltage tolowvoltage).

SensitivelayerofMQ-7gassensitivecomponentsismadeofSnO_2 withstability,So,ithas excellentlongtermstability. Its servicelifecanreach5 years underusing condition.

SENSITVITY ADJUSTMENT

ResistancevalueofMQ-7 is differenceto variouskinds and variousconcentration gases. So, Whenusingthis components, sensitivityadjustment isvery necessary. werecommendthatyou calibratethe detector for200ppmCOinair and usevalueofLoadresistancethat(R_L)about10 KΩ(5KΩ to47 KΩ).

Whenaccurately measuring, theproper alarmpointforthegas detectorshouldbedeterminedafter consideringthetemperatureand humidityinfluence. Thesensitivityadjusting program:

a.Connectthesensortotheapplicationcircuit.

b.Turnonthepower,keep preheatingthroughelectricity over48 hours.

c.Adjusttheloadresistance RLuntilyougetasignalvaluewhichisrespondtoacertaincarbon monoxideconcentrationat theend pointof90 seconds.

d. Adjusttheanotherloadresistance RLuntilyou getasignalvaluewhich is respond to aCO concentraticnattheend pointof60 seconds .

TEL:86-371-67169070 67169080 FAX:86-371-67169090 Email: sales@hwsensor.com

Datasheet for gas sensor MQ-4

TECHNICAL DATA MQ-4 GASSENSOR

FEATURES

*HighsensitivitytoCH4, Naturalgas.
*Smallsensitivitytoalcohol, smoke.
*Fastresponse. *Stableandlonglife *Simpledrivecircuit

APPLICATION

Theyareusedin gas leakagedetecting equipmentsinfamilyandindustry, aresuitablefordetecting of CH4,Naturalgas.LNG, avoidthenoiseofalcoholand cookingfumes andcigarettesmoke.

SPECIFICATIONS

A. Standardwork condition

Symbol	Parametername	Technical condition	Remarks
Vc	Circuitvoltage	5V±0.1	ACORDC
V_H	Heatingvoltage	5V±0.1	ACORDC
P_L	Loadresistance	20KΩ	
R_H	Heaterresistance	33Ω±5%	RoomTem
P_H	Heatingconsumption	less than750mw	

B. Environment condition

Symbol	Parametername	Technicalcondition	Remarks
Tao	UsingTem	-10□-50□	
Tas	StorageTem	-20□-70□	
R_H	Relatedhumidity	less than95%Rh	
O_2	Oxygenconcentration	21%(standardcondition)Oxygen concentrationcanaffectsensitivity	minimum valueis over2%

C. Sensitivity characteristic

Symbol	Parametername	Technicalparameter	Ramark2
Rs	Sensing Resistance	10KΩ- 60KΩ (1000ppmCH_4)	Detectingconcentration scope : 200-10000ppm CH_4,naturalgas
α (1000ppm/ 5000ppmCH_4)	Concentrationslope rate	≤0.6	
Standard detecting condition	Temp: 20□±2□ Humidity:65%±5%	Vc:5V±0.1 Vh:5V±0.1	
Preheattime	Over24 hour		

D. Strucyureandconfiguration, basicmeasuringcircuit

	Parts	Materials
1	Gassensing layer	SnO_2
2	Electrode	Au
3	Electrodeline	Pt
4	Heatercoil	Ni Cralloy
5	Tubularceramic	Al_2O_3
6	Anti-explosion network	Stainlesssteel gauze (SUS316100-mesh)
7	Clamp ring	Copper platingNi
8	Resin base	Bakelite
9	TubePin	CopperplatingNi

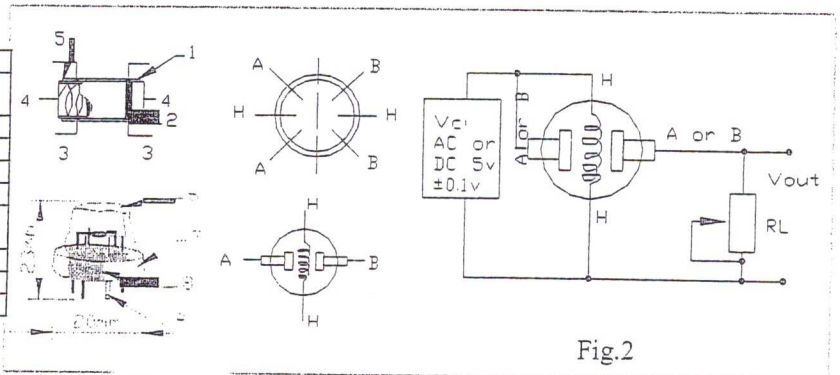
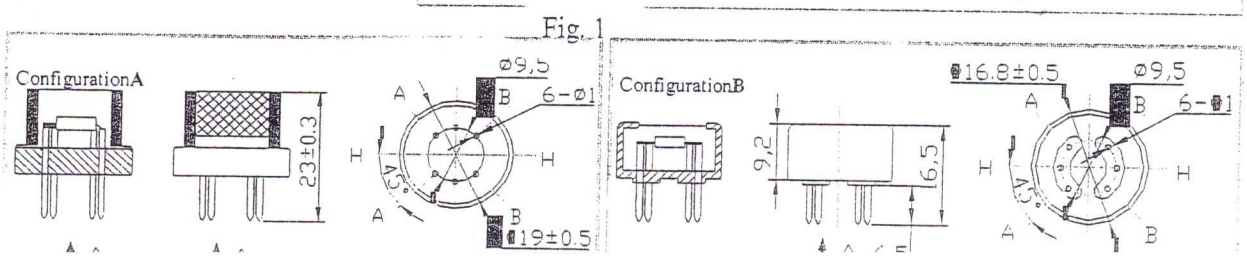


Fig.2

Fig. 1



Structureand configurationofMQ-4gassensorisshown asFig.1(Configuration AorB),sensorcomposed by microAL2O3ceramictube,TinDioxide(SnO2)sensitivelayer,measuringelectrodeandheaterarefixedintoa crust madebyplasticandstainlesssteelnet.Theheaterprovidesnecessaryworkconditionsforworkof sensitivecomponents.TheenvelopedMQ-4have6pin ,4ofthemareusedtofetchsignals,andother2areused forproviding heatingcurrent.

Electricparametermeasurementcircuit is shownas Fig.2

E. Sensitivitycharacteristiccurve

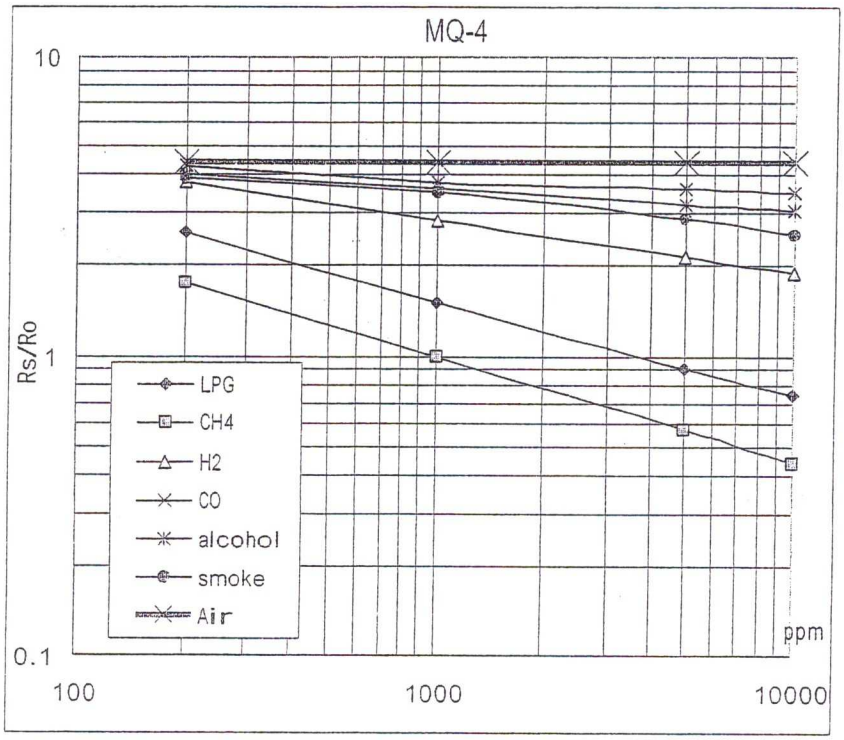


Fig.2sensitivitycharacteristicsoftheMQ-4

Fig.3 is shows thetypical sensitivitycharacteristicsof theMQ-4 forseveralgases.
intheir: Temp: 20□、
Humidity:65%、
O_2concentration21%
RL=20kΩ
Ro:sensorresistanceat1000ppmof CH_4inthecleanair.Rs:sensor resistanceatvarious concentrationsofgases.

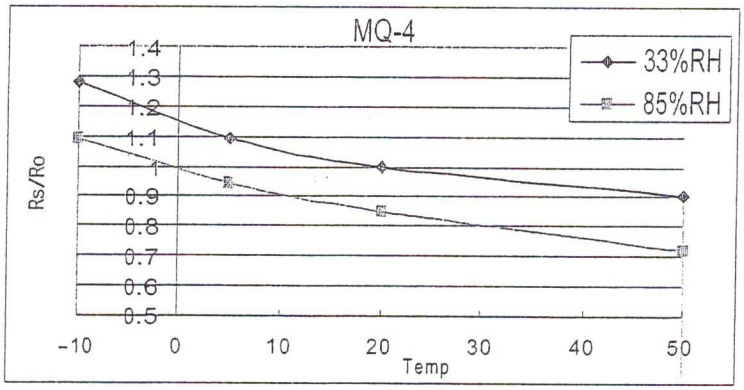


Fig.4 is shows thetypicaldependenceof theMQ-4 ontemperatureand humidity.
Ro:sensorresistanceat1000ppmof CH_4inair at33%RHand20 degree.
Rs:sensorresistanceat1000ppmof CH_4inair atdifferenttemperaturesandhumidities.

SENSITVITYADJUSTMENT

ResistancevalueofMQ-4 is differenceto variouskindsand various concentration gases. So,Whenusing thiscomponents, sensitivityadjustment isvery necessary.werecommend thatyoucalibratethedetector for 5000ppmof CH_4concentrationinair and usevalueofLoadresistance(R_L)about20KΩ(10KΩ to47KΩ).

Whenaccurately measuring, theproper alarmpointforthegas detectorshouldbedeterminedafter consideringthetemperatureand humidityinfluence.

MQ-3 Semiconductor Sensor for Alcohol

Sensitive material ofMQ-3gas sensor isSnO_2,which with lower conductivity in clean air. When the targetalcoholgasexist,Thesensor'sconductivityismorehigheralongwiththegasconcentrationrising. Please use simple electrocircuit,Convert change of conductivity tocorrespond output signal ofgas concentration.

MQ-3gassensorhashighsensitytoAlcohol,andhasgoodresistancetodisturbofgasoline,smoke andvapor.Thesensorcouldbeusedtodetectalcoholwithdifferentconcentration,itiswithlowcostand suitable for differentapplication.

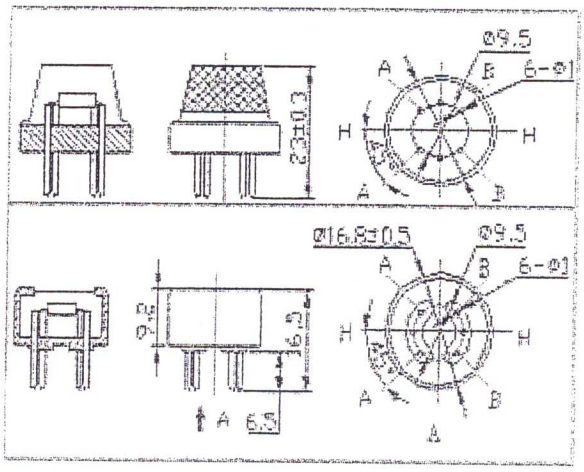
Character

* Good sensitivity toalcohol gas
* Long life and low cost
* Simple drive circuit

Application

* Vehicelalcohol detector
* Portablealcohol detector

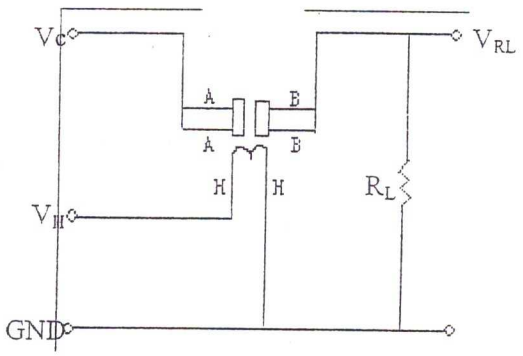
Configuration



TechnicalData

Basictestloop

Model No.			MQ-3
SensorType			Semiconductor
StandardEncapsulation			Bakelite(BlackBakelite)
Detection Gas			Alcohol gas
Concentration			0.04-4mg/l alcohol
Circuit	Loop Voltage	V_C	≤24V DC
	Heater Voltage	V_H	5.0V±0.2VACor DC
	Load Resistance	R_L	Adjustable
Character	Heater Resistance	R_H	31Ω±3Ω（RoomTem.）
	Heater consumption	P_H	≤900mW
	Sensing Resistance	R_s	2KΩ-20KΩ(in0.4mg/l alcohol)
	Sensitivity	S	Rs(in air)/Rs(0.4mg/L Alcohol)≥5
	Slope	α	≤0.6(R_{300ppm}/R_{100ppm}Alcohol)
Condition	Tem. Humidity		20℃±2℃； 65%±5%RH
	Standardtestcircuit		Vc:5.0V±0.1V； V_H:5.0V±0.1V
	Preheattime		Over48 hours



Theaboveisbasictestcircuitofthesensor. Thesensor need tobe put 2 voltage, heatervoltage（VH） andtestvoltage（VC）. VH used tosupply certifiedworking temperature tothe sensor,while VC used todetect voltage (VRL) on load resistance （RL） whomis in serieswith sensor.The sensorhas light polarity, Vcneed DC power. VC andVH could use same power circuitwithprecondition toassure performance of sensor. Inorder tomake the sensorwith better performance, suitable RLvalue is needed:

Power ofSensitivitybody(Ps):

$Ps=Vc^2\times Rs/(Rs+RL)^2$

Resistance ofsensor(Rs): Rs=(Vc/VRL-1)×RL

SensitivityCharacteristics

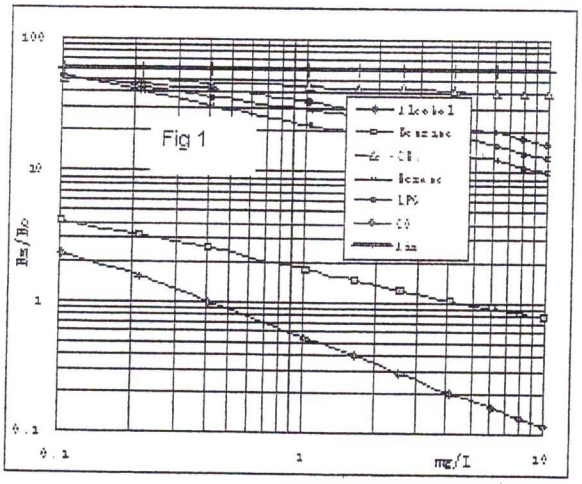


Fig.1shows thetypical sensitivitycharacteristics of the MQ-3,ordinate means resistance ratio of the sensor (Rs/Ro), abscissa is concentrationofgases.Rs means resistancein differentgases,Ro means resistanceof sensorin 0.4mg/l alcohol.All testare understandard testconditions.

P.S.:Sensitivity tosmoke is ignite 10pcs cigarettesin $8m^3$ room, and theoutput equals to0.1mg/l alcohol

Influence of Temperature/Humidity

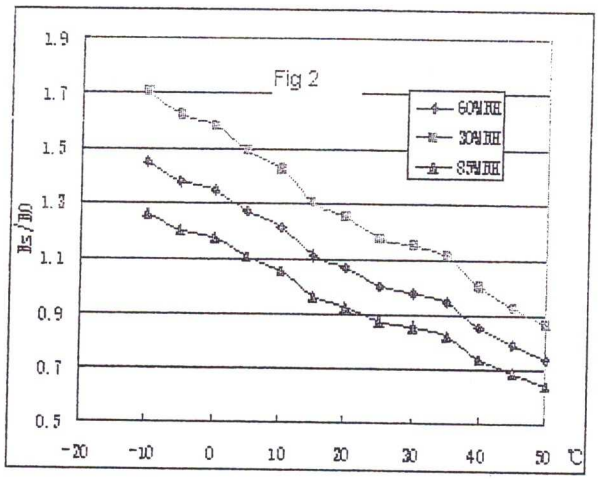


Fig.2shows the typical temperature and humidity characteristics. Ordinatemeans resistance ratio ofthe sensor (Rs/Ro),Rs means resistanceofsensor in 0.4mg/l alcohol under different tem.and humidity. Ro means resistanceofthe sensorin environment of 0.4mg/l alcohol, 20℃/65%RH

Structureandconfiguration

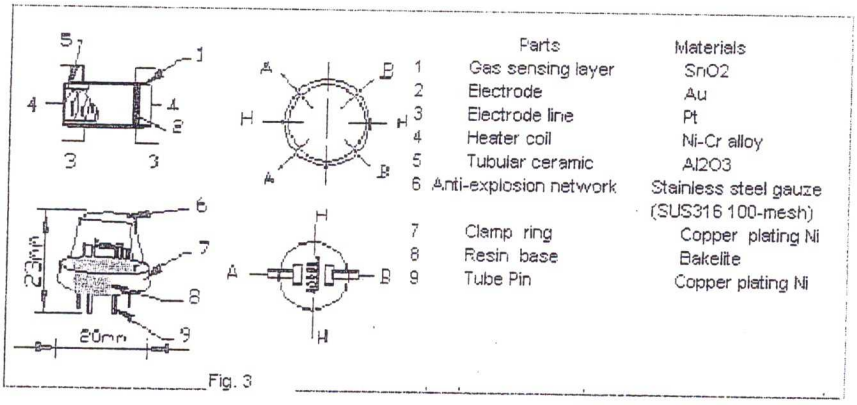


Fig. 3

StructureandconfigurationofMQ-3gassensorisshownasFig. 3,sensorcomposedbymicroAL2O3ceramictube,Tin Dioxide(SnO2)sensitivelayer,measuringelectrodeandheaterarefixedintoacrust madebyplasticandstainlesssteel net.Theheater providesnecessaryworkconditionsforworkofsensitivecomponents.TheenvelopedMQ-4have6pin,4 ofthem are used tofetchsignals, and other 2 are used forproviding heating current.

Notification

1 Followingconditionsmust be prohibited

1.1 Exposedtoorganic silicon steam

Organicsiliconsteamcausesensorsinvalid,sensorsmustbeavoidexposingtosiliconbond, fixature, silicon latex, puttyor plasticcontain siliconenvironment

1.2 High Corrosive gas

Ifthesensorsexposedtohighconcentrationcorrosivegas(suchasH_2Sz,SO_X, Cl_2, HCletc),itwill not only result incorrosion ofsensorsstructure,also it causesincere sensitivity attenuation.

1.3Alkali,Alkali metalssalt,halogen pollution

Thesensorsperformancewillbechangedbadlyifsensorsbesprayedpollutedbyalkalimetalssalt especially brine, or be exposed tohalogen such asfluorin.

1.4Touchwater

Sensitivityofthe sensors will be reduced when spattered or dipped in water.

1.5 Freezing

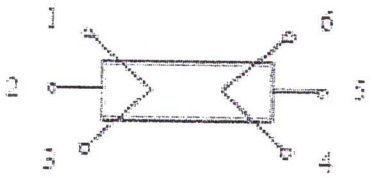
Do avoid icing on sensor'surface, otherwise sensor would lose sensitivity.

1.6Applied voltage higher

Appliedvoltageonsensorshouldnot behigherthanstipulatedvalue,otherwiseitcause down-lineor heater damaged, and bring on sensors'sensitivity characteristicchanged badly.

1.7 Voltageon wrong pins

For6pinssensor,ifapplyvoltageon1、3pinsor4、6pins,it willmakeleadbroken,andwithoutsignalwhenapplyon2、4pins



2 Followingconditionsmust be avoided

2.1 WaterCondensation

Indoorconditions,slightwatercondensationwilleffectsensorsperformancelightly.However,ifwater condensation on sensors surfaceand keep a certainperiod, sensor'sensitivity will be decreased.

2.2 Used in high gasconcentration

Nomatterthesensoriselectrifiedornot,iflongtimeplacedinhighgasconcentration,ifwillaffect sensors characteristic.

2.3 Long time storage

Thesensorsresistanceproducereversibledriftifit'sstoredforlongtime withoutelectrify,thisdriftis relatedwithstorageconditions.Sensorsshouldbestoredinairproofwithoutsilicongelbagwithcleanair. Forthesensorswithlongtimestoragebutnoelectrify,theyneedlongagingtimeforstbilitybeforeusing.

2.4 Long time exposed toadverse environment

Nomatterthesensorselectrifiedornot,ifexposedtoadverseenvironmentforlongtime,suchas high humidity, high temperature, or high pollution etc,it will effectthesensors performance badly.

2.5 Vibration

Continualvibrationwillresultinsensorsdown-leadresponsethenrepture.Intransportationor assembling line, pneumaticscrewdriver/ultrasonicwelding machinecan lead this vibration.

2.6 Concussion

Ifsensors meet strong concussion, itmay lead itslead wire disconnected.

2.7 Usage

Forsensor,handmadeweldingisoptimalway.Ifusewavecrestweldingshouldmeetthefollowing conditions:

2.7.1 Soldering flux:Rosin soldering flux contains least chlorine

2.7.2 Speed:1-2 Meter/Minute

2 7.3 Warm-uptemperature：100±20℃

2.7.4 Weldingtemperature：250±10℃

2.7.5 1 time pass wave crestwelding machine

Ifdisobey the above using terms,sensors sensitivity will be reduced.

MICROCHIP

PIC18F2455/2550/4455/4550

28/40/44-Pin, High-Performance, Enhanced Flash, USB Microcontrollers with nanoWatt Technology

Universal Serial Bus Features:

- USB V2.0 Compliant
- Low Speed (1.5 Mb/s) and Full Speed (12 Mb/s)
- Supports Control, Interrupt, Isochronous and Bulk Transfers
- Supports up to 32 Endpoints (16 bidirectional)
- 1-Kbyte Dual Access RAM for USB
- On-Chip USB Transceiver with On-Chip Voltage Regulator
- Interface for Off-Chip USB Transceiver
- Streaming Parallel Port (SPP) for USB streaming transfers (40/44-pin devices only)

Power-Managed Modes:

- Run: CPU on, peripherals on
- Idle: CPU off, peripherals on
- Sleep: CPU off, peripherals off
- Idle mode currents down to 5.8 µA typical
- Sleep mode currents down to 0.1 µA typical
- Timer1 Oscillator: 1.1 µA typical, 32 kHz, 2V
- Watchdog Timer: 2.1 µA typical
- Two-Speed Oscillator Start-up

Flexible Oscillator Structure:

- Four Crystal modes, including High Precision PLL for USB
- Two External Clock modes, up to 48 MHz
- Internal Oscillator Block:
  - 8 user-selectable frequencies, from 31 kHz to 8 MHz
  - User-tunable to compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Dual Oscillator options allow microcontroller and USB module to run at different clock speeds
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if any clock stops

Peripheral Highlights:

- High-Current Sink/Source: 25 mA/25 mA
- Three External Interrupts
- Four Timer modules (Timer0 to Timer3)
- Up to 2 Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 5.2 ns (TCY/16)
  - Compare is 16-bit, max. resolution 83.3 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Enhanced Capture/Compare/PWM (ECCP) module:
  - Multiple output modes
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
- Enhanced USART module:
  - LIN bus support
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I²C™ Master and Slave modes
- 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D) with Programmable Acquisition Time
- Dual Analog Comparators with Input Multiplexing

Special Microcontroller Features:

- C Compiler Optimized Architecture with optional Extended Instruction Set
- 100,000 Erase/Write Cycle Enhanced Flash Program Memory typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory typical
- Flash/Data EEPROM Retention: > 40 years
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 41 ms to 131s
- Programmable Code Protection
- Single-Supply 5V In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins
- Optional dedicated ICD/ICSP port (44-pin devices only)
- Wide Operating Voltage Range (2.0V to 5.5V)

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EAUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I²C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

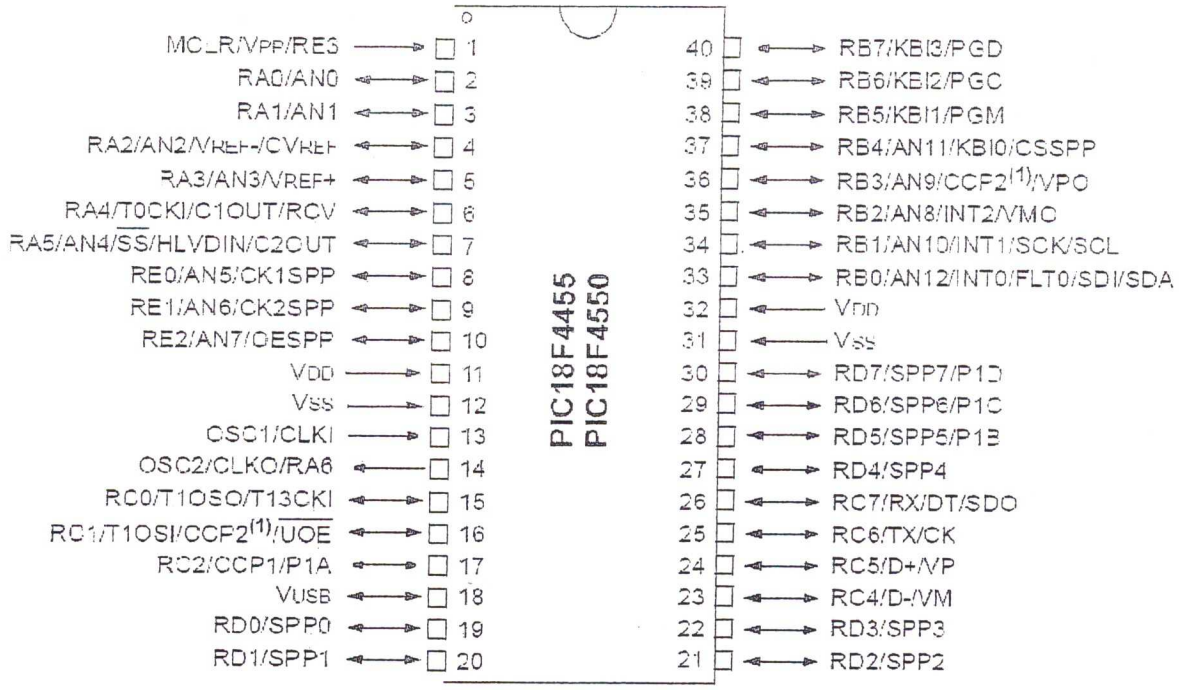


TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
MCLR/VPP/RE3	1	18	18			Master Clear (input) or programming voltage (input).
MCLR				I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
VPP				P		Programming voltage input.
RE3				I	ST	Digital input.
OSC1/CLKI	13	32	30			Oscillator crystal or external clock input.
OSC1				I	Analog	Oscillator crystal input or external clock source input.
CLKI				I	Analog	External clock source input. Always associated with pin function OSC1. (See OSC2/CLKO pin.)
OSC2/CLKO/RA6	14	33	31			Oscillator crystal or clock output.
OSC2				O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO				O	—	In RC mode, OSC2 pin outputs CLKO which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6				I/O	TTL	General purpose I/O pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the DEBUG Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTA is a bidirectional I/O port.
RA0/AN0	2	19	19			
RA0				I/O	TTL	Digital I/O.
AN0				I	Analog	Analog input 0.
RA1/AN1	3	20	20			
RA1				I/O	TTL	Digital I/O.
AN1				I	Analog	Analog input 1.
RA2/AN2/VREF-/CVREF	4	21	21			
RA2				I/O	TTL	Digital I/O.
AN2				I	Analog	Analog input 2.
VREF-				I	Analog	A/D reference voltage (low) input.
CVREF				O	Analog	Analog comparator reference output.
RA3/AN3/VREF+	5	22	22			
RA3				I/O	TTL	Digital I/O.
AN3				I	Analog	Analog input 3.
VREF+				I	Analog	A/D reference voltage (high) input.
RA4/T0CKI/C1OUT/RCV	6	23	23			
RA4				I/O	ST	Digital I/O.
T0CKI				I	ST	Timer0 external clock input.
C1OUT				O	—	Comparator 1 output.
RCV				I	TTL	External USB transceiver RCV input.
RA5/AN4/$\overline{SS}$/HLVDIN/C2OUT	7	24	24			
RA5				I/O	TTL	Digital I/O.
AN4				I	Analog	Analog input 4.
$\overline{SS}$				I	TTL	SPI slave select input.
HLVDIN				I	Analog	High/Low-Voltage Detect input.
C2OUT				O	—	Comparator 2 output.
RA6	—	—	—	—	—	See the OSC2/CLKO/RA6 pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the $\overline{DEBUG}$ Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/AN12/INT0/ FLT0/SDI/SDA	33	9	8			
RB0				I/O	TTL	Digital I/O.
AN12				I	Analog	Analog input 12.
INT0				I	ST	External interrupt 0.
FLT0				I	ST	Enhanced PWM Fault input (ECCP1 module).
SDI				I	ST	SPI data in.
SDA				I/O	ST	I^2C™ data I/O.
RB1/AN10/INT1/SCK/ SCL	34	10	9			
RB1				I/O	TTL	Digital I/O.
AN10				I	Analog	Analog input 10.
INT1				I	ST	External interrupt 1.
SCK				I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL				I/O	ST	Synchronous serial clock input/output for I^2C mode.
RB2/AN8/INT2/VMO	35	11	10			
RB2				I/O	TTL	Digital I/O.
AN8				I	Analog	Analog input 8.
INT2				I	ST	External interrupt 2.
VMO				O	—	External USB transceiver VMO output.
RB3/AN9/CCP2/VPO	36	12	11			
RB3				I/O	TTL	Digital I/O.
AN9				I	Analog	Analog input 9.
CCP2(1)				I/O	ST	Capture 2 input/Compare 2 output/PWM 2 output.
VPO				O	—	External USB transceiver VPO output.
RB4/AN11/KBI0/CSSPP	37	14	14			
RB4				I/O	TTL	Digital I/O.
AN11				I	Analog	Analog input 11.
KBI0				I	TTL	Interrupt-on-change pin.
CSSPP				O	—	SPP chip select control output.
RB5/KBI1/PGM	38	15	15			
RB5				I/O	TTL	Digital I/O.
KBI1				I	TTL	Interrupt-on-change pin.
PGM				I/O	ST	Low-Voltage ICSP™ Programming enable pin.
RB6/KBI2/PGC	39	16	16			
RB6				I/O	TTL	Digital I/O.
KBI2				I	TTL	Interrupt-on-change pin.
PGC				I/O	ST	In-Circuit Debugger and ICSP programming clock pin.
RB7/KBI3/PGD	40	17	17			
RB7				I/O	TTL	Digital I/O.
KBI3				I	TTL	Interrupt-on-change pin.
PGD				I/O	ST	In-Circuit Debugger and ICSP programming data pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the $\overline{\text{DEBUG}}$ Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTC is a bidirectional I/O port.
RC0/T1OSO/T13CKI	15	34	32			
RC0				I/O	ST	Digital I/O.
T1OSO				O	—	Timer1 oscillator output.
T13CKI				I	ST	Timer1/Timer3 external clock input.
RC1/T1OSI/CCP2/UOE	16	35	35			
RC1				I/O	ST	Digital I/O.
T1OSI				I	CMOS	Timer1 oscillator input.
CCP2(2)				I/O	ST	Capture 2 input/Compare 2 output/PWM 2 output.
UOE				O	—	External USB transceiver $\overline{OE}$ output.
RC2/CCP1/P1A	17	36	36			
RC2				I/O	ST	Digital I/O.
CCP1				I/O	ST	Capture 1 input/Compare 1 output/PWM 1 output.
P1A				O	TTL	Enhanced CCP1 PWM output, channel A.
RC4/D-/VM	23	42	42			
RC4				I	TTL	Digital input.
D-				I/O	—	USB differential minus line (input/output).
VM				I	TTL	External USB transceiver VM input.
RC5/D+/VP	24	43	43			
RC5				I	TTL	Digital input.
D+				I/O	—	USB differential plus line (input/output).
VP				I	TTL	External USB transceiver VP input.
RC6/TX/CK	25	44	44			
RC6				I/O	ST	Digital I/O.
TX				O	—	EUSART asynchronous transmit.
CK				I/O	ST	EUSART synchronous clock (see RX/DT).
RC7/RX/DT/SDO	26	1	1			
RC7				I/O	ST	Digital I/O.
RX				I	ST	EUSART asynchronous receive.
DT				I/O	ST	EUSART synchronous data (see TX/CK).
SDO				O	—	SPI data out.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the $\overline{DEBUG}$ Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTD is a bidirectional I/O port or a Streaming Parallel Port (SPP). These pins have TTL input buffers when the SPP module is enabled.
RD0/SPP0	19	38	38			
RD0				I/O	ST	Digital I/O.
SPP0				I/O	TTL	Streaming Parallel Port data.
RD1/SPP1	20	39	39			
RD1				I/O	ST	Digital I/O.
SPP1				I/O	TTL	Streaming Parallel Port data.
RD2/SPP2	21	40	40			
RD2				I/O	ST	Digital I/O.
SPP2				I/O	TTL	Streaming Parallel Port data.
RD3/SPP3	22	41	41			
RD3				I/O	ST	Digital I/O.
SPP3				I/O	TTL	Streaming Parallel Port data.
RD4/SPP4	27	2	2			
RD4				I/O	ST	Digital I/O.
SPP4				I/O	TTL	Streaming Parallel Port data.
RD5/SPP5/P1B	28	3	3			
RD5				I/O	ST	Digital I/O.
SPP5				I/O	TTL	Streaming Parallel Port data.
P1B				O	—	Enhanced CCP1 PWM output, channel B.
RD6/SPP6/P1C	29	4	4			
RD6				I/O	ST	Digital I/O.
SPP6				I/O	TTL	Streaming Parallel Port data.
P1C				O	—	Enhanced CCP1 PWM output, channel C.
RD7/SPP7/P1D	30	5	5			
RD7				I/O	ST	Digital I/O.
SPP7				I/O	TTL	Streaming Parallel Port data.
P1D				O	—	Enhanced CCP1 PWM output, channel D.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the DEBUG Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTE is a bidirectional I/O port.
RE0/AN5/CK1SPP	8	25	25			
RE0				I/O	ST	Digital I/O.
AN5				I	Analog	Analog input 5.
CK1SPP				O	—	SPP clock 1 output.
RE1/AN6/CK2SPP	9	26	26			
RE1				I/O	ST	Digital I/O.
AN6				I	Analog	Analog input 6.
CK2SPP				O	—	SPP clock 2 output.
RE2/AN7/OESPP	10	27	27			
RE2				I/O	ST	Digital I/O.
AN7				I	Analog	Analog input 7.
OESPP				O	—	SPP output enable output.
RE3	—	—	—	—	—	See $\overline{MCLR}$/VPP/RE3 pin.
VSS	12, 31	6, 30, 31	6, 29	P	—	Ground reference for logic and I/O pins.
VDD	11, 32	7, 8, 28, 29	7, 28	P	—	Positive supply for logic and I/O pins.
VUSB	18	37	37	O	—	Internal USB 3.3V voltage regulator output.
NC/ICCK/ICPGC(3)	—	—	12			No Connect or dedicated ICD/ICSP™ port clock.
ICCK				I/O	ST	In-Circuit Debugger clock.
ICPGC				I/O	ST	ICSP programming clock.
NC/ICDT/ICPGD(3)	—	—	13			No Connect or dedicated ICD/ICSP port clock.
ICDT				I/O	ST	In-Circuit Debugger data.
ICPGD				I/O	ST	ICSP programming data.
NC/$\overline{ICRST}$/ICVPP(3)	—	—	33			No Connect or dedicated ICD/ICSP port Reset.
$\overline{ICRST}$				I	—	Master Clear (Reset) input.
ICVPP				P	—	Programming voltage input.
NC/ICPORTS(3)	—	—	34	P	—	No Connect or 28-pin device emulation.
ICPORTS						Enable 28-pin device emulation when connected to Vss.
NC	—	13	—	—	—	No Connect.

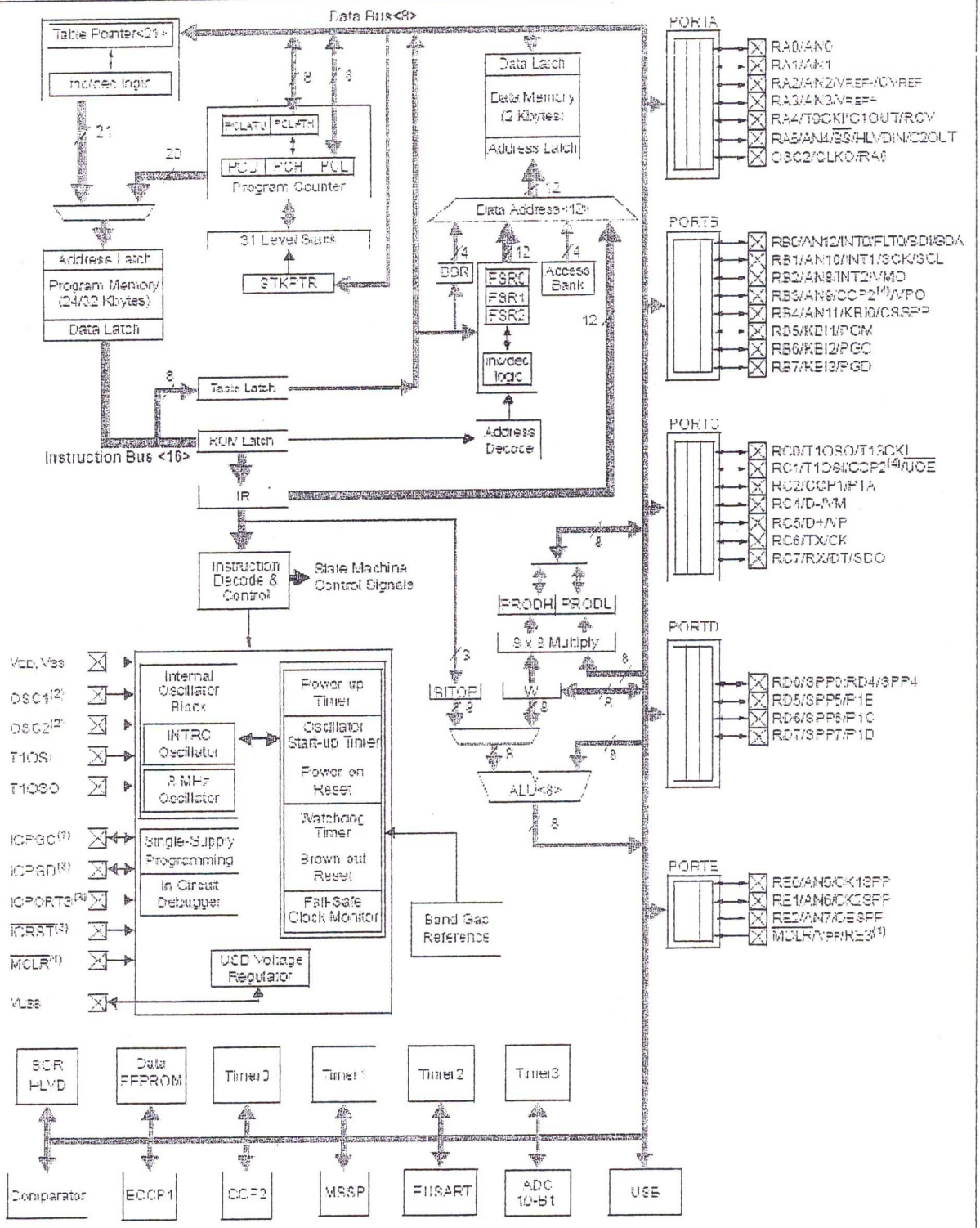
Legend: TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the $\overline{DEBUG}$ Configuration bit is cleared.

FIGURE 1-2: PIC18F4455/4550 (40/44-PIN) BLOCK DIAGRAM



Note 1: RE3 is multiplexed with MCLR and is only available when the MCLR Resets are disabled.

2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to Section 2.0 "Oscillator Configurations" for additional information.

3: These pins are only available on 44-pin TQFP packages under certain conditions. Refer to Section 25.9 "Special ICPORT Features (Designated Packages Only)" for additional information.

4: RB3 is the alternate pin for CCP2 multiplexing.

28.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings(†)

Ambient temperature under bias ..... -40°C to -85°C

Storage temperature ..... -65°C to +150°C

Voltage on any pin with respect to V_{SS} (except V_{DD}, $\overline{MCLR}$ and RA4) ..... -0.3V to (V_{DD} – 0.3V)

Voltage on V_{DD} with respect to V_{SS} ..... -0.3V to +7.5V

Voltage on $\overline{MCLR}$ with respect to V_{SS} (Note 2) ..... 0V to +13.25V

Total power dissipation (Note 1) 1.0W

Maximum current out of V_{SS} pin ..... 300 mA

Maximum current into V_{DD} pin ..... 250 mA

Input clamp current, I_{IK} ($V_I < 0$ or $V_I > V_{DD}$) ..... ±20 mA

Output clamp current, I_{OK} ($V_O < 0$ or $V_O > V_{DD}$) ..... ±20 mA

Maximum output current sunk by any I/O pin ..... 25 mA

Maximum output current sourced by any I/O pin ..... 25 mA

Maximum current sunk by all ports ..... 200 mA

Maximum current sourced by all ports ..... 200 mA

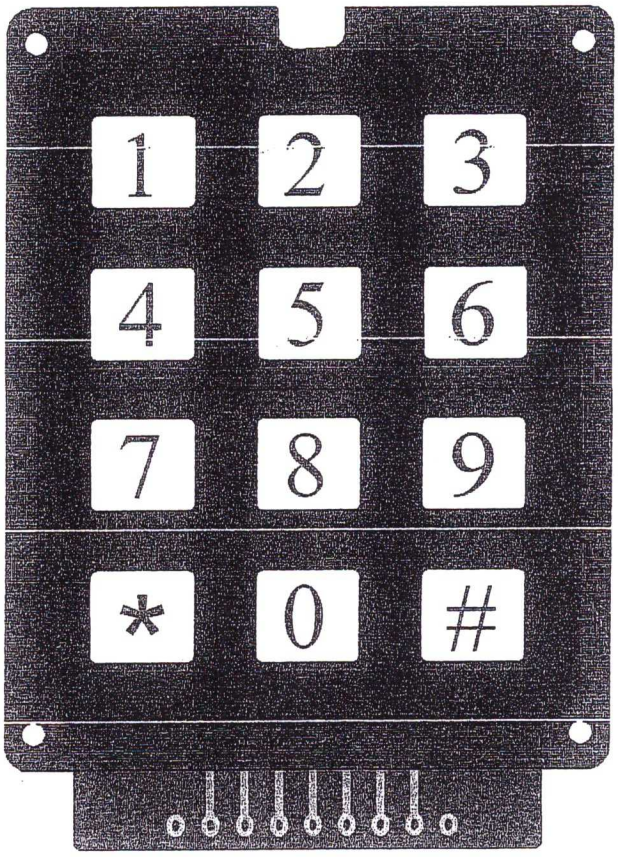
Note 1: Power dissipation is calculated as follows:
$P_{dis} = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} - \Sigma \{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma(V_{OL} \times I_{OL})$

2: Voltage spikes below V_{SS} at the $\overline{MCLR}/V_{PP}$/RE3 pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}/V_{PP}$/RE3 pin, rather than pulling this pin directly to V_{SS}.

† **NOTICE:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

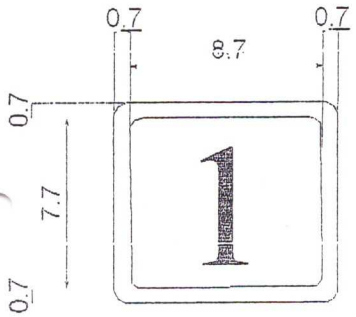
Datasheet for keypad

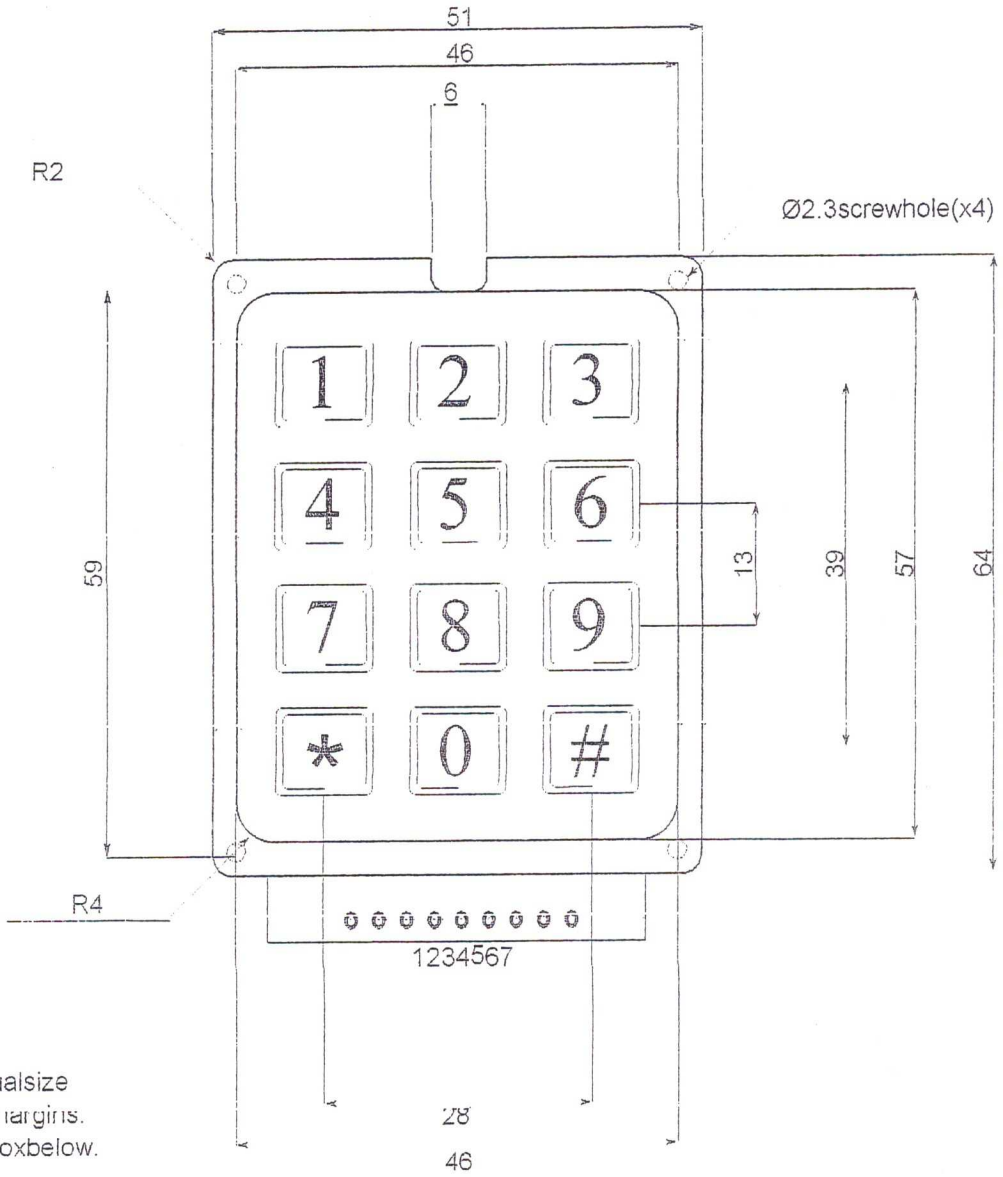
COM-08653



DISCLAIMER:
Thisdocumentistheresultofanobsceneamountof workandresearchbecausel reallywantedtomake anicewoodcase/ bezelusingPonokoforaproject I'mworkingon. Literallyeverythinginthisdocument mightbewrongandifyouwastemoneyorinjure somebodybasedonthisdocument

Mechanical





Allimagesexceptthekeycalloutareactualsize
whenprintedon8.5x11withappropriatemargins.
Youcancalibrateyourprinterusingtheboxbelow.

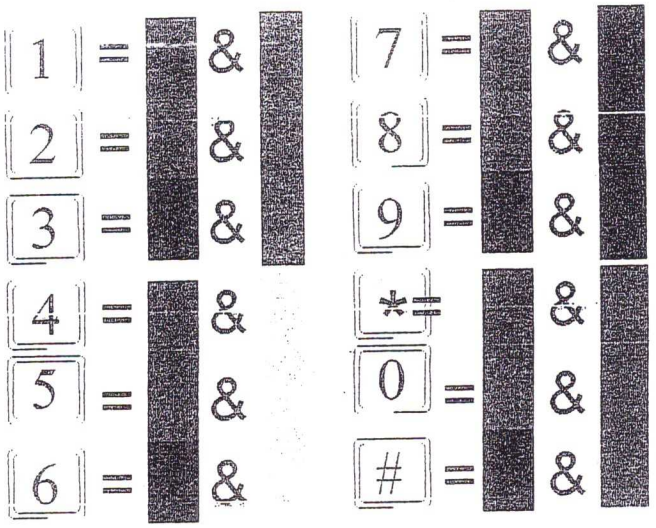
10mm
(1cm)

Electrical

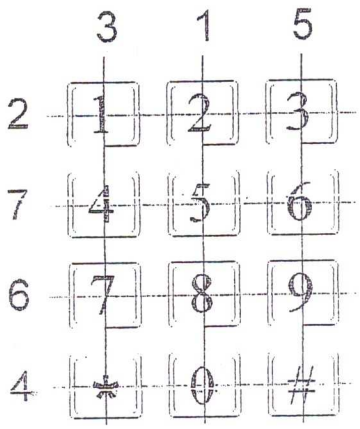
PushButtons:Columns+ Rows

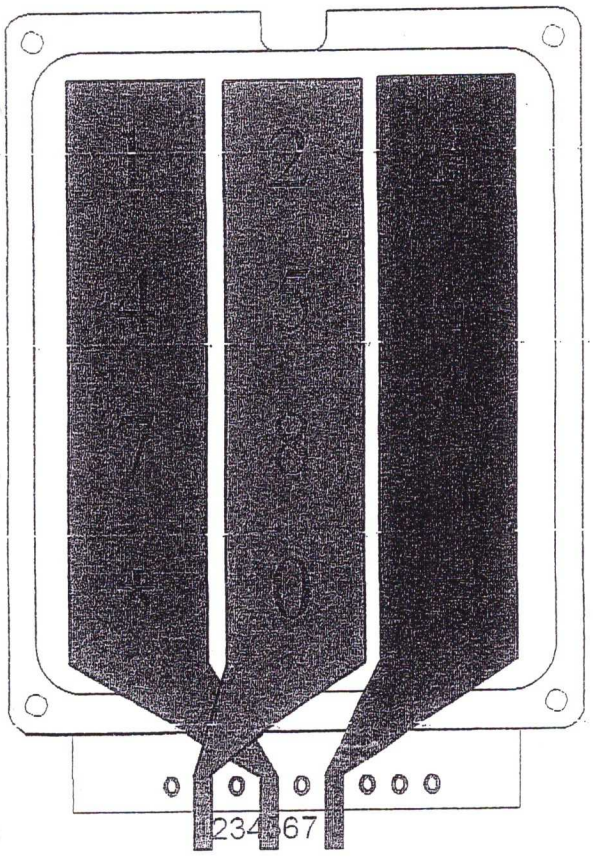
Thisisthesameinformationinseveraldifferentways.
I'vegottenrequeststovisualizeitdifferently,sowhyfightit?

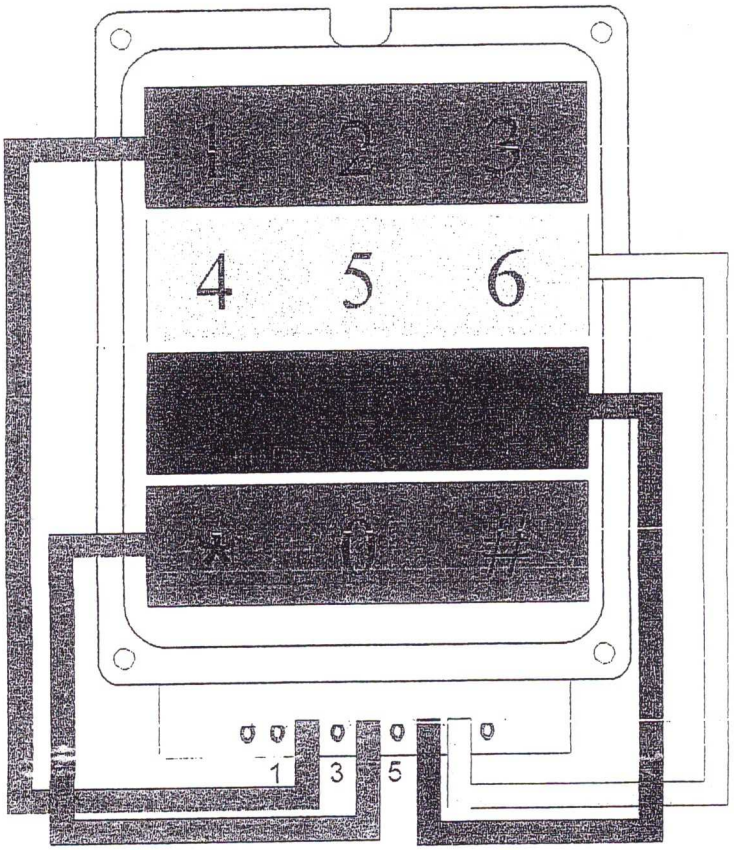
1: 3+2
2: 1+2
3: 5+2
4: 3+7
5: 1+7
6: 5+7
7: 3+6
8: 1+6
9: 5+6
*: 3+4
0: 1+4
#: 5+4



KeypadColumns:3,1,5
KeypadRows:2,7,6,4







Appendix D

Product and Market Analysis from the Project's Business Plan

Business Plan Chapters

Products and Market Analysis for the Building Management System

Market Demand and Competition status

Products and services

Products and services descriptions

Key features of the products and services

Technology and Service Products

Services Operation

Potential Suppliers

Marketing:

- Marketing Strategy
- Marketing Segmentation
- Target Market segments strategy
- Marketing-Mix Strategies

Market Demand and Competition status

Automation services with advanced software technologies that depends on computerized algorithmic control without direct signal intervention is new to local community in Hebron, and it has not been served yet with similar technology and work features and efficiency. However according to the survey:

- There are companies that provide relatively simple services such as gate control, video monitoring and other partial adjustments.
- These companies use off-the-shelf components that are expensive, and have the inefficient use of large amount of resources.

These companies are already partially satisfying the market over the past few years, with the same work characteristics across the entire market; so we have a weak competition from other companies. Because there isn't any company that provides the group of services that our company provides. So it can be classified as indirect competition because its provide very little competition when it comes to automation services.

Those are already controlling the market now because there are no better facilities available .

Depending on information mentioned above and survey results, we have a large opportunity to have a good market share and compete with huge competitive advantages through our service facility and enhancing customer loyalty .

However, the focal points in the market structure that a company selling this product will focus on can be viewed and characterized by:

- Better control quality and precision.
- Better safety solutions.
- Better Media and entertainment Solutions.
- Better solutions for the disabled and those with mobility impairments.
- The appealing economic savings these solutions can enforce on electricity and water.
- Interconnectivity and flexibility of the systems integration. Everything is connected.

The following are questions to be raised when the market characteristics of Hebron are tested:

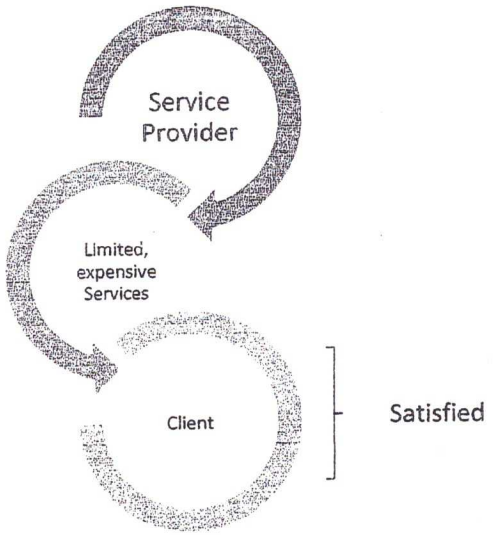
1. Is it possible to invest our product in the Hebron market and the surrounding areas :
2. How can we promote our product idea in the market?
3. What are the incentives to be given?

Selling this product aims to change the structure and the process of demand for automation services through applying the marketing concept, which promote customers satisfaction.

Our role in the market comes at least for the foreseeable three years as a mediator who add value to customers who seek automation features.

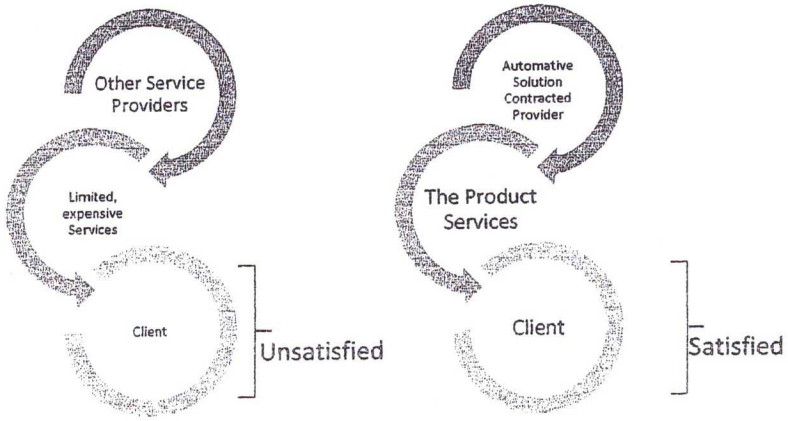
What we are trying to say is as simple as "Safety, higher quality devices, and better economic and environmental work sophistication can be created by working together ".

Before The Product Solutions



- Traditional Services
- Simple technologies
- off-the-shelf components
- Relatively Expensive
- No flexibility
- limited in features

After The Product Solutions



-More Features
-More Flexibility
-Better Integration
-Attractive Design
-Advanced technology
-Suitable for Homes and Businesses
-Cost effective
-Made in Palestine

Figure 7-1 Demand Structure changes as the system is introduced to the market.

Products and services

Products and services descriptions

The system provides customers with the following services and packages:

- Home Package (Digital Home):

  A customizable package that can be installed in any house-held environment. And has an extremely high flexibility in features to price trade-off. The system offer two grades of features:

  - Basic Features:
    - Advanced Smart Grid, an electronic control mechanism that enforces computer controlled switches over the electrical nodes in the buildings electrical grid. Also offers customizable levels of control over the house to provide a good range of price for the customers. It can accomplish tasks such as lights, shutters, and loads control.
    - Basic Sensor Network. It is mainly consisted of two types of sensors, responsible for reporting data on the electric consumption, and water consumption.
    - Computer Software (customized for home use). A bundle of extremely sophisticated Palestinian designed programs with AI capabilities, which are used to run everything in the house.
    - Wireless Remote Control. A device used to control the central computer in the house wirelessly.

  - Extended Features:
    - Advanced Sensor Network, other devices can be installed into the basic sensor network to include data such as:
      - Human Presence
      - Temperature (outdoors and indoors)
      - harmful Gases (CO, CO2, and Petroleum gas)
      - Smoke.
    - Voice Interface: various other tasks can be accomplished using voice commands.
    - Central Computer. If a home doesn't contain a computer, the company can provide one that is relatively cheap with extreme low power consumption abilities

- Wireless extensions, for areas that are disconnected from the main grid.
- Customized Interface. Software interface can be customized to fit within the interior decoration of the house.
- Home central-lock, with keypad and password to gain entry into the house.
- Video monitoring, the system can support a limited number of IP video cameras for monitoring solutions.

- Business Automation Packages:

  Consists of two packages that can be installed in environments suited for business use:

  - Basic control package, which is consisted of the following features:
    - Central Computer with a basic graphical interface
    - Basic smart grid, enforcing control over a limited number of nodes.
    - Limited human presence sensor network, which is used for logging purposes and entrance control.

  - Full control package, includes all of the basic features and includes the following :
    - Central Computer with a basic graphical interface.
    - Advanced Smart Grid.
    - Security and door locking mechanisms.
    - Presence Sensor Network for logging proposes.
    - Cameras and Video Monitoring.
    - Temperature control.

- Customized Solutions:

  Upon demand from a client, a different combination of features and a user-specific solution could be presented at additional costs.

Key features of the products and services

Services provided by the system will be characterized by:

- Flexibility in design when addressing the service; that will provide the customers the choice to customize the solution based on their individual tastes.

- Using Microchip Corporation technology will provide better controller quality and high cost-efficiency.
- Using Microsoft Corporation software technology to provide top of the line software.
- The prices will be lower than the competitors, who perform this type of work by using off-the-shelf components.
- New solutions for the disabled and those with mobility impairments.
- New solutions for providing automated control for the purpose of adopting a green and economic way of living.
- First system to use artificial intelligence as means of control.
- The main products are all developed in the Palestine Polytechnic University, and are a pure representation of Palestine creativity.

Technology and Service Products

The product depends on the latest technology of in terms of software and hardware , to provide users with the best possible combination of features and prices, we currently employ the following products in our services:

- G-Box Home System Software:

  The main controlling software developed by undergraduate engineers in the Palestine Polytechnic University, it is consisted of the following components:

  - Hardware Driver module, which is the USB software component used to interface with the main microcontroller.
  - Voice Recognition Engine and Speech Synthesizer Interface. A component used to convert voice commands over the microphone into an integer code.
  - Home Database and Database interface for saving users information and personal configurations.
  - WPF GUI module, which acts as the main interface for the program, it displays a 3D character, consumption charts, an interactive house plan, and environmental information regarding the state of the house.
  - Main software controller and AI Module. The algorithms responsible for decision making and interfacing all of the software modules.

- AS-1 Control Box

  A device developed by undergraduate engineers in the Palestine Polytechnic University, it offers controlling mechanisms and a computer interface. It offers the following features:

  - USB Interface – to communicate with the central computer
  - 64 Control ports compatible with Control Nodes ASC01
  - 13 Analog Ports compatible with Presence Sensors ASP01
  - 11 Analog Ports compatible with Sensor Node Array ASN01-ASN05
  - Internal LCD Screen
  - Internal main electronic switch
  - Manual grid separation switch

- ASC01 Control Node

  Acts as an electronic switch for a high power line, it operates based on a signal from the AS-1 Control Box.

- ASP01 Presence Node

  It sends a signal whenever a person passes in front of its sight range.

- ASN01 Electrical Sensor Node

  It sends an analog signal in accordance with the electrical power flowing in a power line.

- ASN02 Water flow Sensor Node

  It sends a signal that can estimate the amount of water flowing through a pipe within a time frame.

- ASN03 Temperature Sensor
- ASN04 Smoke Sensor
- ASN05 Gas Sensor
- Z-Box-ID41-E Atom Microcomputer
- Teac SMX-20 Wireless Keyboard and Track Ball Mouse
- Security Keypad
- IP Camera

Services Operation

The services have a specific characteristics in its operation, our services will be held to customers upon their request and as they desire, because most of our marketing strategy concentrate on customization, the first step in our operation will begin of request of customer as demand with specific requirements, the second step is depend on designer who responsible for designing and determining the final shape of the needed works.

The third step is to implement the needed works by the technicians who are responsible for installation of the system in the clients buildings according to the designs and software set up by the system designer.

Potential Suppliers

This Table provides a list of potential suppliers for the raw materials and components needed .

Inputs Needed	Supplier Information	
	Company	Location
Small Electrical Components and Special Electronics	Omega Electronics	Hebron - IbnRushd St.
	Al Basha'er Electronics	Hebron – AinKheer El Din St.
	Badawi Electronics	Nablus – Faysal St.
Larger Electronics and Computer Parts	Al Alameyah	Hebron Duwar Al Manarah
	Al Dawleyah	Hebron Duwar Al Manarah
Special Parts for Customized Requests	Smart Trading	China - Kwanzo
	Electrofun	Jerusalem

Table 1 Potential Suppliers

Marketing

The product market is classified as both a business market (B2B)and a consumer market (B2C).

To clarify the business market, our market is the decoration and construction market, more specifically; businesses concerned with providing Home renovations and buildings contracts.

As we seek to extend our services to involve a large portion of market share we realized the following notes that shaped the way we approach the market :

- There are different businesses and different types of consumers that deal with different aspects of home modeling and renovations, and each one is concerned with specific features.
- Market segmentation and customized products tend to increase effectiveness and efficiency in providing our services.

Marketing Strategy

Since our product is joining the market as a new entrant with new concepts and technology, our marketing strategy will start with discovering customer's needs and is identified as a differentiation strategy.

Our service will be positioned very carefully; this is done through automation services for the customer who understands the need for such services and the innovation they embed. Our marketing strategy is based mainly on making the right information available to right target customer. The marketing has to express the sense of quality in every picture, every promotion, and every publication. We can't afford to appear in second-rate catalogs with poor illustrations that make the services look less than it is. We also need to present our presence using high quality catalogs.

Marketing Segmentation

Based on our market research and survey results we have found that the key market segment of our company which we can cooperate with is divided into four segments which are:

1. High income home-owners.
2. Shops (like clothing stores).
3. Offices and small businesses.

   These market segments are mostly characterized by their lack of understanding for the deep technological aspects of the service and rely mostly on the general appearance and net value of the service.
4. Engineering offices and decoration consultants.

This market segment is characterized by the presence of more technically aware individuals, presentation material and the ability to negotiate further into customization for the products and services.

Target Market segments strategy

Our market segments are defined based on their controlling power in the design and specification of the end product and the reflected segments hold its definition by itself. We are not intended to satisfy all customers in and outside our segments, but, rather only those who are the most demanding and those who address the need for high quality and better work specifications.

In our particular market we seek the buyers who will be pleased and are concerned about the following:

- The quality of work and the excellence of design.
- The green life style.
- Features for the disabled.
- Flexibility in design.

Marketing-Mix Strategies

Each of the four marketing-mix (4 P's) strategies conveys the product differentiation to the target market segments which identified in previous section. It should be noted that due to the special nature of our services.

Pricing Strategy

Prices for our services will depend on the package and amount of features required for installation, which is unstable due to different design specifications and design dimensions. So a customer may have the flexibility of doing a trade-off between additional features and price.

This is one of our strengthening factors, since we use up to date, high technological components with high speed, lower working costs(consumption) and that are relatively at a low production cost, making it a central factor in determining service prices in existing market, and this also reflects the present high prices for similar service.

The estimated selling price for the most basic package ranges between 600$ to 800$.

The estimated selling price for the additional services and features can go from 200$ to 1200$

- Prices are determined to attract desirable channel partners and to gain the desirable market share.
- Hold Stable prices between customers to insure client satisfaction.

The prices range according to the customers' existing requirements and needs.

Promotional Strategy

Promotional campaign focuses on creating awareness for the existence of our company and our product properties , consultative selling is considered as a part of promoting strategy for our business, and to create customer loyalty. Promotional advertisement will position our image in the market.

Promotional tools that will be used for our campaign are:

1. Personal Selling

Personal selling is the most important part in the company promotional strategy, since our business depend mostly on contractual and partnership with clients through professional sales persons, we can introduce the company message clearly to the clients, and increase the company market share, for that; this tool will take a large share of the promotional budget.

2. TV advertisement

The TV advertisement will be placed during the prime time in the period between 7-10 pm at rate of 4 advertisements per night.

We choose that time because we believe that the majority of our target audiences are watching at that time.

Year	2012	2013	2014
Total Cost	₪9,000	₪5,400	₪4,800

Table 2 TV advertisement budget

These prices vary from one TV station to another, so a general yearly allowance is put for this medium, meaning that rates per month may vary,

3. Newspaper advertisements

Which are considered of the most advantageous means in which we can reach our business customers Al- Quds newspaper is selected for this purpose with the following cost:

Year	2012								2013	2014
Month	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug		
Frequency	10	8	10	12	15	15	12	10	92	51
Cost	₪ 250	₪ 250	₪ 250	₪ 250	₪ 250	₪ 250	₪ 250	₪ 250	₪250	₪250
Total Cost	₪ 2,500	₪ 2,000	₪ 2,500	₪ 3,000	₪ 3,750	₪ 3,750	₪ 3,000	₪ 2,500		
TOTAL									₪ 23,000	₪ 12,750

Table 3 Newspaper advertisements budget

4. Brochures

As a company provides service in many forms and ideas, we believe that brochure is a major tool that can hold much of information about the company operations and capabilities on the other hand, displays many picture for environments with our solutions .A sample Brochure is included with Appendix.

These are our promotional campaign in the first year of operation. In the second year we will promote our service through introducing a catalog display a complete set of its finished systems.

Positioning Strategy

The company headquarters will be established in Hebron in Al-Hawouz Althani, with the following characteristics and Key points to take in regard:

	Characteristic	Scores	Details
1	External appearance.	4	The location is on the main street of Second Hawouz that enables everyone to see it even from their cars.
2	Interior design and layout.	4	The design is well organized; every section has its organized offices that distributed on the building to get the maximum benefit from its area to provide a beautiful and attractive layout.
3	Visibility of the store.	4	Perfect visibility for the walkers, cars, and for surrounded shops customers also.
4	Entrance and signage.	4	The entrance which have a beautiful view that attract visitors to enter.
5	Size and adaptability.	3	The size of the building is enough until now, but there's no space for future expansion.
6	Distance from competitors.	3	Most of indirect competitors in the downtown and in other cities.
7	Closeness to customers.	3	It's not near the downtown of Hebron but it's surrounded by many famous and important shops like Sbitany, Shaheen Center, Al-AshkarRestraunt...
8	Customer traffic.	3	The number of customers will increase during the time especially that the people of Hebron love Technology stuff and saving money.
9	Transportation and facilities network.	4	The company locate at the main street of the Second Hawouz which makes it simple to reach from many ways without any difficult.
10	Parking.	1	No parking, and can't build one.

Table 4 Position characteristics and key points