INTELLIGENT HOME AUTOMATION SYSTEM

MAK KWAN WUEY

A project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor (Hons.) of Electronics and Communications Engineering

Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

May 2011

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : ___________________________

Name : ___________________________

ID No. : ___________________________

Date : ___________________________

APPROVAL FOR SUBMISSION

I certify that this project report entitled **“INTELLIGENT HOME AUTOMATION SYSTEM** was prepared by **MAK KWAN WUEY** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor (Hons.) of Electronics and Communications Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : ___________________________

Supervisor : Dr Goi Bok Min

Date : ___________________________

Specially dedicated to

My beloved mother and father

ACKNOWLEDGEMENTS

Throughout this project, I would like to thank Dr Goi Bok Min, my supervisor for guiding me in this project for the difficulties I faced, and Liu Chee Wei, who has given me valuable feedbacks in designing this system as a whole. My gratitude extends to my families and my housemate that has supports me in this project.

INTELLIGENT HOME AUTOMATION SYSTEM

ABSTRACT

Intelligent Home Automation System (IHAS) is a computer based wireless home automation system that uses Adobe Flash for the user interface, enabling users to control physical devices on their computer. IHAS uses 802.15.4 in controlling various types of devices. IHAS is developed as a low cost alternative in home automation technology as it doesn't need a dedicated hardware console. The aim of this project is to reduce the cost associate to home automation, which will encourage more people to install home automation technology in their home. IHAS has achieved success in implementing core features such as plug and play coordinator setup, RFID control and integration, AES128 encrypted communication, communication fault detect and control extension to an Android smartphone device. Also, IHAS control interface is optimized for both mouse control and touch input. This is possible with the use of Adobe Flash which is a web based multimedia platform. One of the major challenges in this project is how to send data over COM port using Adobe Flash, as it does not natively support such feature. Also, setting up a secure connection between coordinator and end devices is another challenge that has been solved in IHAS. In short, IHAS is successful and have achieved the aims and objectives. Improvement has to be made to the overall stability of IHAS and integration of IHAS and commercial products could be further investigated. Also, IHAS could expand to control various types of devices, from power monitoring to security in the future.

TABLE OF CONTENTS

DECLARATION **ii**

APPROVAL FOR SUBMISSION **iii**

ACKNOWLEDGEMENTS **vi**

ABSTRACT **vii**

TABLE OF CONTENTS **viii**

LIST OF TABLES **xii**

LIST OF FIGURES **xiii**

LIST OF SYMBOLS / ABBREVIATIONS **xvi**

LIST OF APPENDICES **xvii**

CHAPTER

1 INTRODUCTION 18

1.1 Background 18

1.2 Justification and Motivation 19

1.3 Aims and Objectives 20

1.4 Methodology 20

1.5 Project Scopes 22

1.6 Thesis Outline 22

2 LITERATURE REVIEW 23

2.1 Communication protocol 23

2.1.1 Communications Protocol Background 23

2.1.2 Number of supported devices 24

2.1.3 Network Topology 24

2.1.4 Power Requirement 25

2.1.5 Security 26

2.1.6 Hardware requirements 26

2.1.7 Other Considerations 27

2.1.8 Summary 27

2.2 XBEE 28

2.2.1 XBEE Hardware 28

2.2.2 XBEE Setup and Command Modes 29

2.3 RFID in Intelligent Home Automation System 33

2.4 User Interface Design 34

2.4.1 User Interface Design Guidelines 34

2.4.2 Comparison between Visual Basic and Adobe Flash 35

2.4.3 Adobe Flash with Serial Port Communication 36

2.5 Software 36

2.5.1 Adobe Flash Professional CS5 36

2.5.2 MDM Zinc 3.0.22 37

2.5.3 Eterlogic Virtual Serial Ports Emulator 0.938.4.846 38

2.5.4 AGGsoftware Com Port Data Emulator 2.7.1 build 622 38

2.5.5 XCTU 32-bit ver. 5.1.4.1 39

2.5.6 TightVNC version 2.0 39

3 INTELLIGENT HOME AUTOMATION SYSTEM 40

3.1 Overview 40

3.2 Management Console 41

3.3 User Interface for Main Menu 41

3.4 User Interface for Lights 42

3.5 Automated Plug and Play Coordinator Setup 42

3.6 Automated Scanning for End Devices Status 44

3.7 Remote Encryption Settings for End Devices 45

3.8 IHAS Communications Format and Protocol 46

3.9 RFID Integration in IHAS 47
3.10 Modular Design 48
3.11 Smartphone Controlled IHAS 49
3.12 Compatibility of IHAS with Commercial Products 50

4 IMPLEMENTATION OF IHAS AND OUTCOMES 53
4.1 Sending Data through a Com Port 53
4.1.1 Implementation 53
4.1.2 Outcome 57
4.2 Design a User Friendly UI 61
4.2.1 Implementation 61
4.2.2 Outcome 61
4.3 Automated Setup for XBEE Coordinator 64
4.3.1 Implementation 64
4.3.2 Outcome 65
4.4 Automated Scanning for End Devices Status 66
4.4.1 Implementation 66
4.4.2 Outcome 67
4.5 Remote Encryption Settings for End Devices 68
4.5.1 Implementation 68
4.5.2 Outcome 70
4.6 RFID Integration with IHAS 70
4.6.1 Implementation 70
4.6.2 Outcome 72
4.7 Smartphone Controlled IHAS 73
4.7.1 Implementation 73
4.7.2 Outcome 73
4.8 Design Solution based on commercial products 75
4.8.1 Implementation 75
4.8.2 Outcome 81

5 CONCLUSION AND FUTURE IMPLEMENTATION 83

REFERENCES **84**

LIST OF TABLES

TABLE	TITLE	PAGE
Table 2.1: Summary of Zigbee and ONE-NET specifications		27
Table 3.1: IHAS Communication Protocol		46
Table 4.1: Bill of Material for Lighting and HVAC control prototype		76
Table 4.2: Preset RFID Configuration		78

LIST OF FIGURES

FIGURE	TITLE	PAGE
Figure 1.1: Intel Intelligent Home Management Proof of Concept (Kanellos, 2010)		19
Figure 2.1: Zigbee Network Topologies		24
Figure 2.2: A Pair of XBEE Starter Kit		28
Figure 2.3: Entering AT Command Flow Chart		30
Figure 2.4: AT Command format		30
Figure 2.5: API for AT command		31
Figure 2.6: Example of API Mode Frame Data		32
Figure 2.7: RFID and RFID tags		33
Figure 2.8: Adobe Flash (Adobe Flash Professional CS5)		36
Figure 2.9: MDM Zinc 3.0.22 (Multidmedia Ltd, 2002)		37
Figure 2.10: Eterlogic Virtual Serial Ports Emulator 0.938.4.846 (ETERLOGIC, 2007)		38
Figure 2.11: AGGsoftware Com Port Data Emulator (AGGsoftware, 1999)		38
Figure 2.12: XCTU 32-bit ver. 5.1.4.1 interface		39
Figure 2.13 : TightVNC Server Version 2.0		39
Figure 3.1: Overall Block Diagram for Intelligent Home Automation System		40
Figure 3.2: Block Diagram for Management Console		41
Figure 3.3: Management Console Main Menu		41
Figure 3.4: Lighting Control Page		42

Figure 3.5: Automated Setup of Coordinator for XBEE 43

Figure 3.6: IHAS Communication Protocol Example 46

Figure 3.7: When IHAS Detects RFID 47

Figure 3.8: Registering new RFID card to IHAS database 48

Figure 3.9: RFID Hardware Modular Design 49

Figure 3.10: Design Solution for Energy Savings On Campus 51

Figure 3.11: Design Concept on management console 52

Figure 4.1: Zinc 3.0 54

Figure 4.2: Virtual Serial Com port 55

Figure 4.3: Com Port Emulator Setup 55

Figure 4.4: Configuring Data Emulator to Send Data 56

Figure 4.5: Scanning Ports 57

Figure 4.6: Started Com Port Emulator Before Starting Up Application 58

Figure 4.7: Result from Application Testing (Send) 59

Figure 4.8: Result from Application Testing (Receive) 60

Figure 4.9: Main Menu 61

Figure 4.10: Lights Control Page (full) 62

Figure 4.11: Lights Control (desktop mode) 63

Figure 4.12: Flow chart to configure XBEE settings automatically 64

Figure 4.13: Plug and Play XBEE Coordinator Easy Setup 65

Figure 4.14: Steps to Detect and Save End Devices to Database 66

Figure 4.15: End Device Scanning 67

Figure 4.16: endnodes.xml Datafile 67

Figure 4.17: Remote Encryption Setting 68

Figure 4.18: API Frame Example 69

Figure 4.19: Security Setup 70

Figure 4.20: Modular design of RFID Conversion Circuit 71

Figure 4.21: RFID Conversion Circuit and Power Conversion and Regulation Circuit 72

Figure 4.22: Complete RFID Hardware 72

Figure 4.23: Smartphone (left) and IHAS Application(right) 73

Figure 4.24: Smartphone Streaming IHAS Application 74

Figure 4.25: Electrical Wiring diagram for HVAC and Lighting Control System Prototype 75

Figure 4.26(A): Lighting Control Prototype 77

Figure 4.27: Different Preset Modes, from left clockwise (OHP mode, Projector mode and All On) 79

Figure 4.28: HVAC Prototype 80

Figure 4.29: The Lighting and HVAC Control System Prototype 81

Figure 4.30: IEEE STUDENT Conference 2010 Best Exhibitor Award Winner, From left, MAK KWAN WUEY, LIU CHEE WEI, GAN YU HAN. 82

LIST OF SYMBOLS / ABBREVIATIONS

RFID	Radio Frequency Identification
IHAS	Intelligent Home Automation System
MICS	Modular Intelligent Control System
DALI	Digital Addressable Lighting Interface
UI	User Interface
WSA	Wide Sensor Array
AES	Advance Encryption Standard
XTEA-2	eXtended Tiny Encryption Algorithm version 2
ROM	Read-Only Memory
RAM	Random Access Memory
GUI	Graphical User Interface
EXE	Executable File
PC	Personal Computer
USB	Universal Serial Bus
XML	Extensible Markup Language
FLA	Flash Movie Authoring File
RS-232	Recommended Standard 232
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
AT	A prefix to enter the command mode of XBEE
API	Application Programming Interface (XBEE)
PLC	Programmable Logic Controller
AHU	Air Handling Unit
HVAC	Heating, Ventilation and Air Conditioning

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
Appendix A	XBEE Command List	84
Appendix B	Baseline Data and Energy Savings Calculation	91

CHAPTER 1

INTRODUCTION

1.1 Background

Home automation system existed for decades but due to its high cost, it remains as a niche product for the high end consumers. With energy crisis looming in the horizon, home automation devices' popularity has seen an uptake to reduce unnecessary electricity consumption. However, mass adoption of home automation technology is not possible unless there is a significant cost reduction in installation of home automation system. Therefore, we have to identify and implement ways to reduce the installation cost of home automation systems. Wireless system appeals to us as we could cut down wiring cost in conventional home automation system.

Home automation system could not be complete without a control and monitoring console. Figure 1.1 shows an example energy management console from Intel, which integrates answering machines and household clocks on the home energy management console (Kanellos, 2010). Intel tries to make energy management a part of our daily lives, by requiring us to interact with it daily. Instead of a dedicated console, we believe we could do the same thing on the computer as we interact with the computer most of the time, not to mention we could save the cost of installing a dedicated energy management console.

Figure 1.1: Intel Intelligent Home Management Proof of Concept (Kanellos, 2010)

In this project, IHAS will offer lighting control as a proof of concept. Also, IHAS also provides multiple methods in lighting control such as central console from computer, RFID control and control via smartphone.

1.2 Justification and Motivation

The reason that motivates us to develop such system is that there is a need for cheap and easy to implement home automation system. By using an open protocol, we could save license fee for proprietary and close protocol. By choosing a wireless communication medium, we could save wiring and installation cost as we could reuse existing infrastructure with minimal modification. By developing an application that enables users to control devices from the computer instead of a dedicated console, we could save the cost for the need of a dedicated automation console.

1.3 Aims and Objectives

The aim of this project is to build a cost effective wireless home automation system that is based on an open protocol, easy to configure and setup home automation system, RFID control and secured communication. It is hoped that our project will spark the interest of home automation manufacturer in low cost home automation technology that is affordable to mass market.

Therefore, the objectives in this project are:

- Build a robust and scalable wireless home automation system that is suitable for multiple devices such as lights.
- Implement wireless security to avoid potential malicious attacks to the home automation system.
- Design a home automation management console that is controlled from a personal computer, which is both intuitive and appealing to all kinds of home users.
- Evaluate and implement various intelligent features that could make IHAS stand out among competing solutions such as automation control from smartphones, RFID lighting control and plug and play easy setup.

1.4 Methodology

The project starts from a comprehensive study on the communications protocol, effective User Interface (UI) design and implementation of various planned features for IHAS. We have to research on the alternatives, evaluate the performance of each and choose the best solution for our project. The sources of information could be obtained but not limited to website, journals, books, electronics news source and company’s newsletter.

After choosing the technology and related method, we have to test out systematically towards the aims and objectives of the proposed system. For example, we have examined how to send a simple data through Zigbee after we have chosen Zigbee as our communication protocol. Also, we have to examine if the chosen software to create the home automation console is capable of transmitting and receiving data from a Zigbee transceiver. All tests conducted are detailed in this report.

When communications is proved to be working, planned features such as easy plug and play setup, RFID lighting control, wireless security and automation control from smartphone are implemented and tested individually.

Finally, when individual features are working, all features are put together as a complete system. IHAS is tested with a lighting control system in collaboration of Smart Lighting System: Modular Intelligent Control System (Wei, 2010).

1.5 Project Scopes

This project comprises mainly 2 sections, the Intelligent Home Automation System (IHAS) and Smart Lighting System: Modular Intelligent Control System (Wei, 2010). IHAS presents the devices and control panel in a logical and attractive manner to the users. Also, IHAS also handles setup of communication network between the users and Smart Lighting System: Modular Intelligent Control System. IHAS also provides an alternative way to control groups of lights, which is through personalized lighting profiles. Smart Lighting System: Modular Intelligent Control System on the other hand, deals with physical devices which can be controlled by IHAS.

1.6 Thesis Outline

This thesis will start off with a discussion of home automation communication protocols that are available in the market and the hardware that are needed for IHAS. Also, effective user interface design concept is also studied and analysed, which is needed in designing IHAS application.

In the succeeding chapter, IHAS features and concept will be discussed in detail, while exploring the possibilities of using IHAS with commercial products.

The next chapter that come after will discuss how to achieve and implement some of the features that are mentioned in the preceding chapter. Outcome of the implementations are also presented in the same chapter.

Finally, a brief conclusion on the current work on IHAS and the future implementation and direction of IHAS will be available at the last chapter.

CHAPTER 2

LITERATURE REVIEW

2.1 Communication protocol

Currently, there are quite a number of communication protocols for home automation system such as INSTEON, BACnet, X10, KNX, LonWorks, C-Bus, CEBus, SCS BUS with OpenWebNet, Universal powerline bus (UPB), ZigBee and Z-Wave. However, in this project, wireless communication protocols are chosen to reduce rewiring of current systems. Therefore, the list is shortened to INSTEON, Zigbee, Z-Wave and ONE-NET. These four protocols are further compared to find which protocol is best suited for IHAS.

2.1.1 Communications Protocol Background

Insteon is a closed protocol established by Smarthome since 1992 for home automation networks (Technology, 2005). This means that one could not design a solution based on their protocol outside the company. On the other hand, Zigbee is an open alliance protocol that is established to become a global control/sensor network. (Kinney, 2003) Z-wave however, is an open consortium between 160 manufacturers who create products based on Z-wave, which is a protocol design for controls of residential and light commercial system (Alliance, 2004). Lastly, ONE-NET is a multi vendor, open source wireless control protocol for home automation

(ONE-NET, 2008). From here, we can see that ONE-NET and Zigbee is an open protocol and not Z-Wave and INSTEON, and for this project, Z-Wave and INSTEON is eliminated as equipment design that is based on a closed protocol requires additional licensing fee.

2.1.2 Number of supported devices

Zigbee could be addressed up to 18,450,000,000,000,000,000 devices (64 bit IEEE address) (Kinney, 2003), which is practically unlimited addresses, whereas ONE-NET could only support up to 4096 unique devices (Cotter, ONE-NET, 2008). Therefore, Zigbee is preferred over ONE-NET in terms of maximum supported devices per network as more supported devices will be an advantage in system expansion in the future.

2.1.3 Network Topology

Zigbee could be configured to run star, mesh, cluster tree or even hybrid modes as shown in Figure 2.1 (MeshNetics Team, 2009).

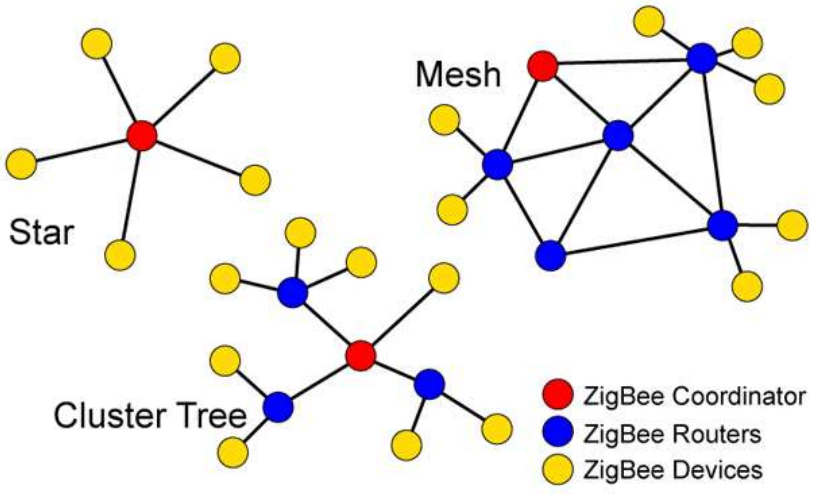


Figure 2.1: Zigbee Network Topologies

ONE-NET on the other hand supports star, mesh and peer to peer configuration. (ONE-NET Topology, 2008). Referring to Figure 2.1, the star network is the easiest kind of network implementation as it is controlled by a single hub. However, it has one huge disadvantage, which is a single point of failure could cripple the whole network. Cluster tress network make use of some local processing in the nodes, in Zigbee, we call it Zigbee routers. These nodes has the ability to pass information to and from central point (normally a Zigbee coordinator) and they are slightly more powerful in terms of processing power compared to Zigbee Devices, which are the end-nodes so that we could introduce low level algorithm in the routers for local processing. Mesh network on the other hand is the hardest to implement but it offers the most robust way in a Wide Sensor Array (WSA) network. It offers Zigbee routers to pass information to each another so that a single point of failure would not cripple the whole network. Peer to peer network is a single point to point communication, which is not used useful in a WSA network.

From a practicality standpoint, both Zigbee and ONE-NET offer mesh network, therefore neither communications protocol has advantages over the other in this section.

2.1.4 Power Requirement

Both Zigbee and ONE-NET protocol is designed from ground-up for home automation, therefore both protocols have battery life up to years (ONE-NET FAQ, 2008) (Meshnetics Team, 2009). However, Zigbee Green power initiatives provides upcoming Zigbee protocols with self-energy harvesting for Zigbee devices (Harrop, 2010) whereas ONE-NET has no intention on self-energy harvesting for the foreseeable future. Therefore in this segment, Zigbee is a better choice as compared to ONE-NET, as battery less sensors could be implemented in the future.

2.1.5 Security

Zigbee is an 802.15.4 wireless standard hence it uses AES (advance encryption standard) with 128b key length whereas ONE-NET uses XTEA-2 (eXtended Tiny Encryption Algorithm) with 32 rounds encoding. Not all encryption methods are created equal. In this case, using a genetic implementation of differential attack, it produces the same output regardless of how many rounds XTEA uses (Hernandez & lsasi, 2004). This poses a security risk with a known theoretical attack method to the system. AES – 128 does not suffer from such attacks hence it is more secured than XTEA-2. Based on this fact, Zigbee has the upper hand in terms of security, as compared to ONE-NET.

2.1.6 Hardware requirements

Zigbee requires an 8 bit microprocessor, <32kB of Read Only Memory (ROM) for full protocol stack, with simple nodes only stack require ~6kB of ROM. Zigbee coordinators require extra Random Access Memory (RAM) for node device database, transaction table and pairing table. On the other hand, ONE-NET also requires at least an 8 bit microprocessor, but it only needs 16kB ROM, 1kB of RAM and 64B of non-volatile memory for a complete ONE-NET devices. In terms of pure hardware requirements, ONE-NET edges Zigbee as it is a less complicated protocol than Zigbee.

2.1.7 Other Considerations

Zigbee has a throughput of 250kbps using 2.4GHz band, which is higher than 230kbps of ONE-NET. Also, although 2.4 GHz is an unlicensed band, the performance will not degrade with the presence of other 2.4Ghz devices such as Wi-Fi network (Thonet, Allard-Jacquin, & Colle, 2008). Zigbee devices are readily available and support documents are much comprehensive as compared to ONE-NET.

2.1.8 Summary

Table 2.1: Summary of Zigbee and ONE-NET specifications

Protocols	Maximum supported devices	Network topology	Power requirement	Security	Hardware requirement	Other Considerations
Zigbee	practically unlimited	all types	low	AES128	More	More support documents
ONE-NET	limited	all types	low	X-TEA2	Less	-

After some careful and detailed comparison, Zigbee is used as the communication protocol for this project.

2.2 XBEE

2.2.1 XBEE Hardware

The hardware that is chosen for this project is XBEE starter kit from Cytron technologies model name SKXBEE. This starter kit is based on XBEE OEM XB24 version 1, which unfortunately is not based on full Zigbee specification; rather it is using the same underlying radio standard, 802.15.4. This XBEE supports AES encryption, star and point to point communication, but it does not support mesh configuration. However, XBEE has all the features for the proposed system's prototype. We shall refer Zigbee module as XBEE from this point onwards.

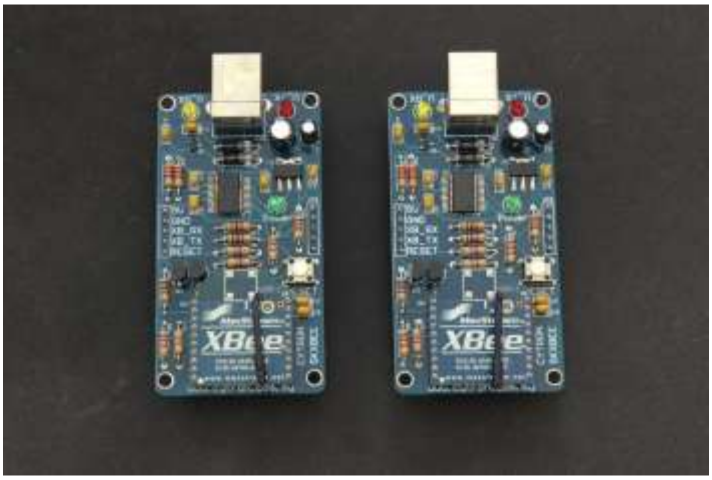


Figure 2.2: A Pair of XBEE Starter Kit

Figure 2.2 shows a pair of XBEE module starter kit. As our system focuses on features that can be achieved by using XBEE starter kit rather than XBEE transceiver only based on the following reasons:

- XBEE transceiver uses 3.3V for all connections, which complicates design of integration with microcontroller and other hardware. XBEE starter kit accepts 5V for power and data communications.
- XBEE starter kit has USB connection that we could use together with X-CTU software for debugging purposes. X-CTU is covered in chapter 2.5.5.

- There are a set of LED indicator lights for USB transmission that helps us to learn how to transmit information with Rx and Tx indicator lights for each module.

2.2.2 XBEE Setup and Command Modes

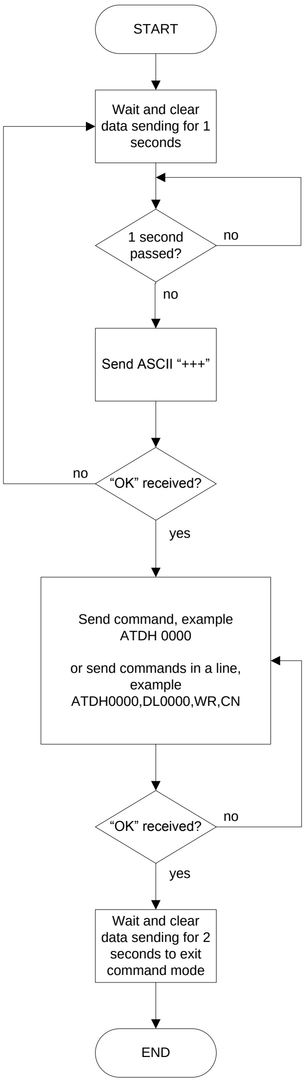
XBEE must be configured it could be used. There are 2 ways to configure XBEE, the 2 methods will be discussed in detail in chapter 2.2.2.1.

2.2.2.1 AT command mode

If the XBEE is configured to operate as transparent mode, where it functions as a serial replacement after configuration, there are specific conditions to be met to enter command mode. The flow of entering command mode is shown in figure 2.3.

Figure 2.3 shows how to enter AT command mode to modify XBEE settings. This mode is easier to configure as compared to API command mode. Once entered the command AT command mode, XBEE configurations could be altered using the format as shown in Figure 2.4. For a full list of commands, please refer to Appendix A.

To check a certain XBEE settings, for example to know the unique address of XBEE, the “Paramater” is omitted in at command, and XBEE will response the parameter requested instead of replying “OK”.


START
Wait and clear data sending for 1 seconds
1 second passed?
no
no
Send ASCII "+++"
"OK" received?
no
yes
Send command, example ATDH 0000

or send commands in a line, example ATDH0000,DL0000,WR,CN
"OK" received?
no
yes
Wait and clear data sending for 2 seconds to exit command mode
END

Figure 2.3: Entering AT Command Flow Chart

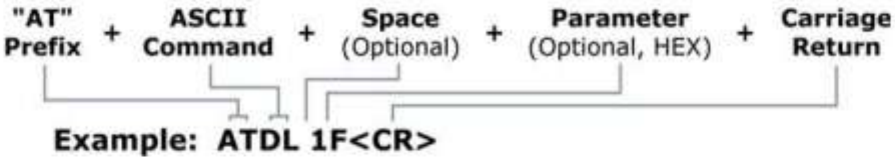


Figure 2.4: AT Command format

2.2.2.2 API Command mode

Instead of operating in transparent operation, which functions as a serial replacement, XBEE could be configured to communicate using a structured order. This mode of operation is called API mode. Using API mode has more freedom in controlling XBEE locally or remotely, however, the cost is increased complexity.

IHAS uses both transparent mode and API mode to communicate, to cut down unnecessary complexity tied to API mode. Figure 2.5 shows an example of how to enter AT command using API mode.

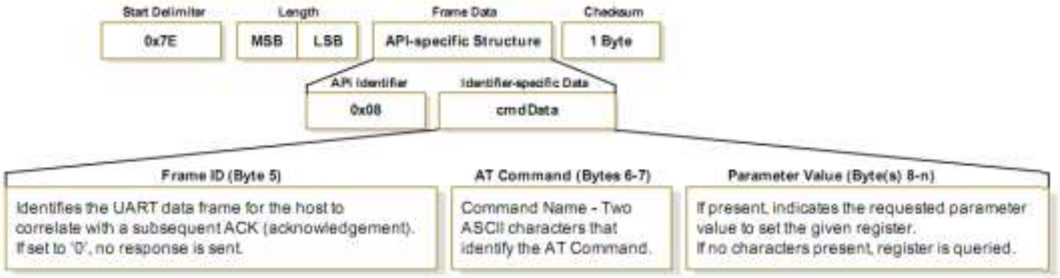

Start Delimiter
0x7E
Length
MSB
LSB
Frame Data
API-specific Structure
Checksum
1 Byte
API Identifier
0x08
Identifier-specific Data
cmdData
Frame ID (Byte 5)
Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to '0', no response is sent.
AT Command (Bytes 6-7)
Command Name - Two ASCII characters that identify the AT Command.
Parameter Value (Byte(s) 8-n)
If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.

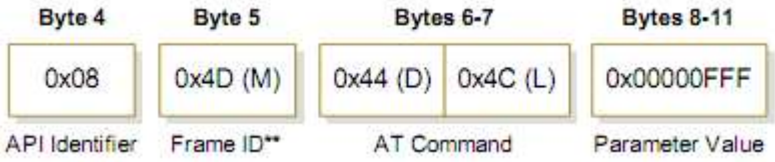
Figure 2.5: API for AT command

As shown in Figure 2.5, every command is send in hexadecimal, not ASCII as in transparent operation. Figure 2.5 shows how to enter AT command using API mode.

1) **Start Delimiter**
   One byte HEX value 7E. This is the starting number of *any* API mode frames. Sending data to other XBEE, entering AT commands, remote configuring a XBEE must start with this.

2) **Length**
   Two byte HEX value. This is the number on the length of the frame data. This length is varied and needs to compute on case by case basis.

3) **Frame data**
   Frame data carries information data, from control information to data frames. Based on different uses, for example for entering AT command mode, frame data starts with 0x08.



Byte 4	Byte 5	Bytes 6-7		Bytes 8-11
0x08	0x4D (M)	0x44 (D)	0x4C (L)	0x00000FFF
API Identifier	Frame ID**	AT Command		Parameter Value

Figure 2.6: Example of API Mode Frame Data

In Figure 2.6, we use an example of changing destination address (low) of the XBEE where API identifier of 0x08 defines AT command, 0x4D is acknowledge byte that the receiver will respond upon receiving the command, AT command of DL (Destination Low) with Hex equivalent, and with the optional parameter value to edit the destination low parameter, or read the current destination low parameter if absent.

4) Checksum

One Byte data for error checking. To calculate checksum, not including frame delimiters and length, add all bytes, keeping only the lowest 8 bits of the result and subtract from 0xFF.

2.3 RFID in Intelligent Home Automation System

As IHAS can be implemented in various situations such as commercial and home, RFID can offer personalized settings to different environment. For example, if IHAS is installed to control lightings in a house, a RFID could detect the different user to the room, and adjust the lighting in the room accordingly. More detail usage on the application of RFID in IHAS will be discussed on chapter 3.10. The RFID hardware that is chosen for IHAS prototype is Cytron Technologies RFID-IDR-232N. Figure 2.7 shows the hardware used in this project and the RFID cards.

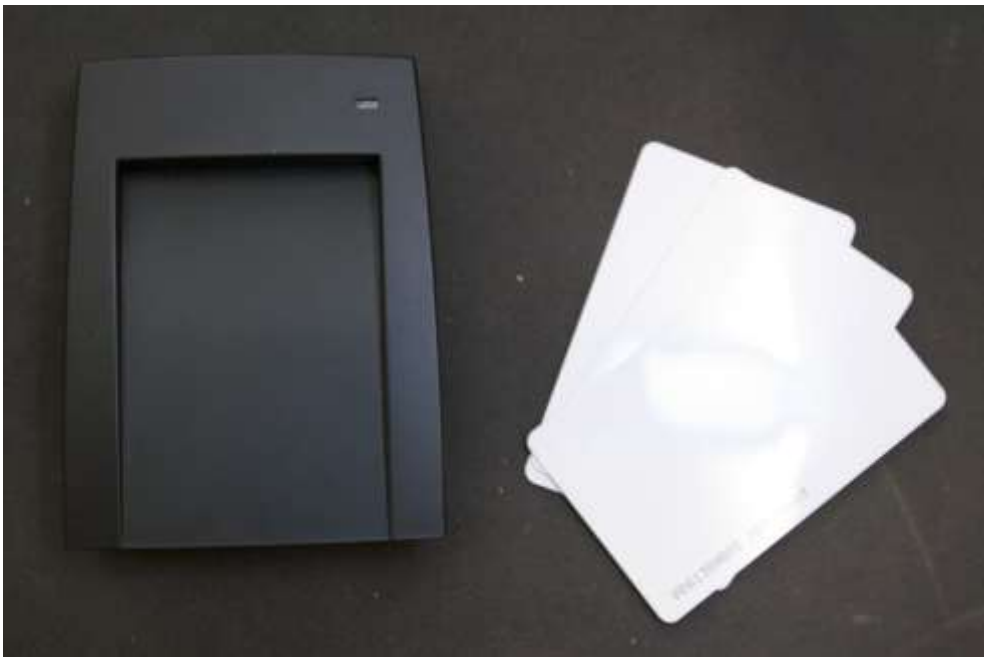


Figure 2.7: RFID and RFID tags

The RFID hardware from Cytron Technology has the following features: (RFID Reader RS232, 2010)

- Support 125KHz RFID passive tag
- 9600 baud rate through RS232 standard
- Fully powered from 5V of USB connection
- Buzzer as audio indicator for read operation
- Red and Green LED as visual indicator for power and read operation

- Come with USB cable (for 5V power) and DB9 standard RS232 socket (female), ready to be plug in to desktop computer with serial port
- 2cm reading range

Since this RFID reader from Cytron Technology is built to send data through RS232, care must be taken if it is used to interface with other embedded devices as the voltage level for communication is 8.5-10.5V, way above the threshold of embedded devices.

2.4 User Interface Design

For IHAS, all the controls and information is presented on the computer screen so that the users could navigate and control at the ease of their computers. However, there are design guidelines that have to be followed to ensure the designed user interface is easy to understand by all levels of users.

2.4.1 User Interface Design Guidelines

There are 6 principles in the fundamentals of user interface design, which are structure principle, simplicity principle, visibility principle, feedback principle, tolerance principle and reuse principle (Lucy A. D. Lockwood & Larry L.Constantine).

The structural principle states that the UI should be designed consistently, recognizable and apparent to users, grouping related things together and separate unrelated things. The structure principle is the overall presentation of the user interface architecture. The simplicity principle says an effective UI design should make simple task easy to do and providing good shortcut that are meaningfully related to longer procedure. Visibility principle emphasis on keeping the user interface clean and simple, and try not to cramp too many unnecessary data in the same screen. Feedback principle touch on the issue of notifying users about the changes on user’s action that are relevant and of interest to the users through a clear and concise and unambiguous language familiar to users. Tolerance principle gives a certain flexibility that could reduce cost of mistakes by allowing undoing and redoing, while also covers an input tolerance that is within user’s reasonable interaction with the UI. Reuse principle states that we should design our UI to be modular and could be reuse throughout the program, maintaining consistency with purpose thus reducing the need the users to rethink and remember. (Scott W. Ambler, 1998)

2.4.2 Comparison between Visual Basic and Adobe Flash

Visual Basic and Adobe Flash is both suitable to design a Graphical User Interface (GUI) for IHAS application. Visual Basic .NET is an object oriented computer programming language by Microsoft. This differs from other object oriented programming such as C++ with the ability to design a GUI at a short time. However, the GUI is extremely simplistic and monotone that hardly attracts users to interact with the program.

Adobe Flash on the other hand is created as a multimedia platform to add interactivity, video and animation to webpages. More recently, it has evolved to a crossover of multimedia and application over the net, making it rich internet applications such as AIR Flash that is used to create web applications that does not require webpages. Adobe flash contains an object oriented language call Actionscript.

Between these 2 applications, one that was built to entertain and designed to capture people’s attention and one that offers a simple GUI to work with a certain code, it is clear that Adobe Flash fits the need of an attractive and intuitive user interface for IHAS application.

2.4.3 Adobe Flash with Serial Port Communication

Adobe flash is not without its drawback to use as the GUI for the IHAS. Natively, Actionscript 3.0 does not support communication via serial port. However, a workaround is found that is using a 3rd party application call mdm Zinc 3.0. Zinc 3.0 offers Actionscript 3.0 extensions that allow a Flash to communicate with the serial port.

Also, Zinc 3.0 also wraps a flash application into an executable file (EXE) that will allow the designed product to run as a standalone application. Therefore, by using Zinc 3.0, we have solved the issues of communicating with serial port that was present on the native Actionscript 3.0 codes.

2.5 Software

2.5.1 Adobe Flash Professional CS5

Figure 2.8: Adobe Flash (Adobe Flash Professional CS5)

Adobe Flash Professional 5 is the latest version of flash authoring tool from Adobe (Adobe, 2011). Originally established as a multimedia platform to add animation, video, interactivity to webpages, it has positioned itself as a tool for “Rich Internet Application” as it supports XML based FLA source files, actionscript editor, wide content distribution, code snippets panel, a new text engine and video improvements (Adobe Flash, 2011). For future expansion, it is a bonus that the same application developed in Adobe Flash can be deployed in multiple areas, such as desktop application, mobile phones, other electronic devices that run on Adobe Flash.

Actionscript 3.0 is a powerful object oriented programming that is highly suitable to build rich internet application. Actionscript 3.0 is based on ECMAscript, the international standardized programming language for scripting. We will be using Actionscript 3.0 for smart control as it is flexible and powerful enough for our usage.

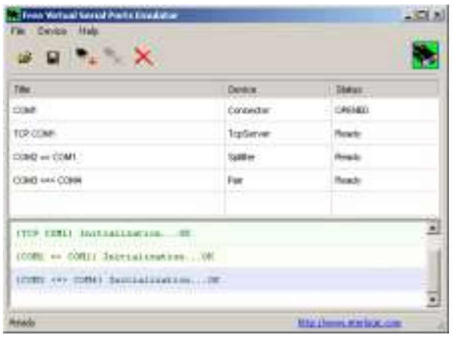
2.5.2 MDM Zinc 3.0.22

Figure 2.9: MDM Zinc 3.0.22 (Multidmedia Ltd, 2002)

Figure 2.9 shows MDM Zinc 3.0.22, which is used in the development of IHAS application. This application is used to create, build and deploy real desktop applications based upon Adobe flash, allowing us to turn an internet based application into powerful desktop application (Multidmedia Ltd, 2002).

Zinc 3.0 has some of the following unique features, including extending Actionscript with over 500 new events, methods and property (Multidmedia Ltd, 2002). For example, native Actionscript does not have the ability to retrieve/send data through a communication, write files to local drives and exit an application easily, by using mdm Zinc 3.0, these problems are solved.

2.5.3 Eterlogic Virtual Serial Ports Emulator 0.938.4.846



Title	Device	Status
COM1	Connector	OPENED
TCP COM1	TcpServer	Ready
COM2 <= COM1	Splitter	Ready
COM3 <=> COM4	Pair	Ready

```
[TCP COM1] Initialization...OK
[COM2 <= COM1] Initialization...OK
[COM3 <=> COM4] Initialization...OK
```

Ready http://www.eterlogic.com

Figure 2.10: Eterlogic Virtual Serial Ports Emulator 0.938.4.846 (ETERLOGIC, 2007)

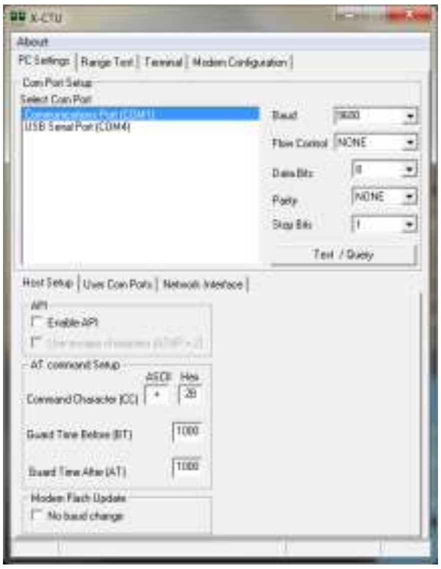
Figure 2.10 shows the Eterlogic Virtual Serial Port Emulator that creates virtual serial ports that could be use to debug the software before try to communicate directly with the hardware. Using Eterlogic Virtual Serial Port, 2 actual serial ports is not needed to test the system.

2.5.4 AGGsoftware Com Port Data Emulator 2.7.1 build 622

Figure 2.11: AGGsoftware Com Port Data Emulator (AGGsoftware, 1999)

Figure 2.11 shows AGGsoftware Com Port Data Emulator which is a software that generates a serial stream of data. The program can create a data flow, wrap it to data packets (RS232, TCP/IP or UDP) and send to a port (AGGsoftware, 1999). This software is used to receive/send data to IHAS application to emulate the data in/out to the XBEE.

2.5.5 XCTU 32-bit ver. 5.1.4.1



Communications Port (COM1)
USB Serial Port (COM4)

Baud 9600
Flow Control NONE
Data Bits 8
Parity NONE
Stop Bits 1

□ Enable API
□ Use escape characters (ATAP = 2)

Command Character (CC) ASCII + Hex 2B
Guard Time Before (BT) 1000
Guard Time After (AT) 1000

□ No baud change

Figure 2.12: XCTU 32-bit ver. 5.1.4.1 interface

Figure 2.12 shows X-CTU Software. X-CTU is the XBEE diagnostics software which is invaluable in learning to communicate with XBEE. It offers web update to the latest revision of the connected XBEE, and support for API and AT command mode for XBEE. Also, it can change XBEE configuration without typing command, although a built in terminal is available.

2.5.6 TightVNC version 2.0

Figure 2.13 : TightVNC Server Version 2.0

Figure 2.13 shows TightVNC version 2.0. TightVNC is a Virtual Network Computing (VNC) server that allows remote connection into the computer. It will stream the desktop to the connected client, providing real time control of the host computer on a remote site.

CHAPTER 3

INTELLIGENT HOME AUTOMATION SYSTEM

3.1 Overview

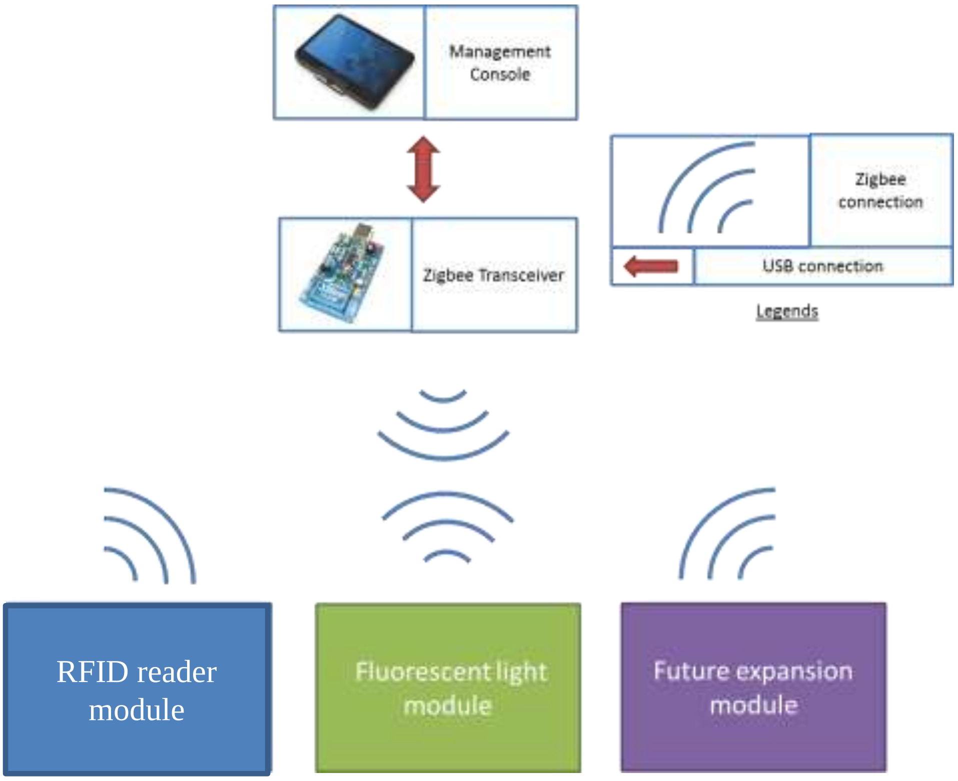


Figure 3.1: Overall Block Diagram for Intelligent Home Automation System

Figure 3.1 shows the overall block diagram for IHAS. The IHAS Application which resides in the PC connects with a XBEE through USB connection and communicates wirelessly with end devices. For example, the light module receives command from the IHAS to dim the lights to 50 %. IHAS Application also receives sensor data from other modules such as occupancy sensor and light sensors.

3.2 Management Console

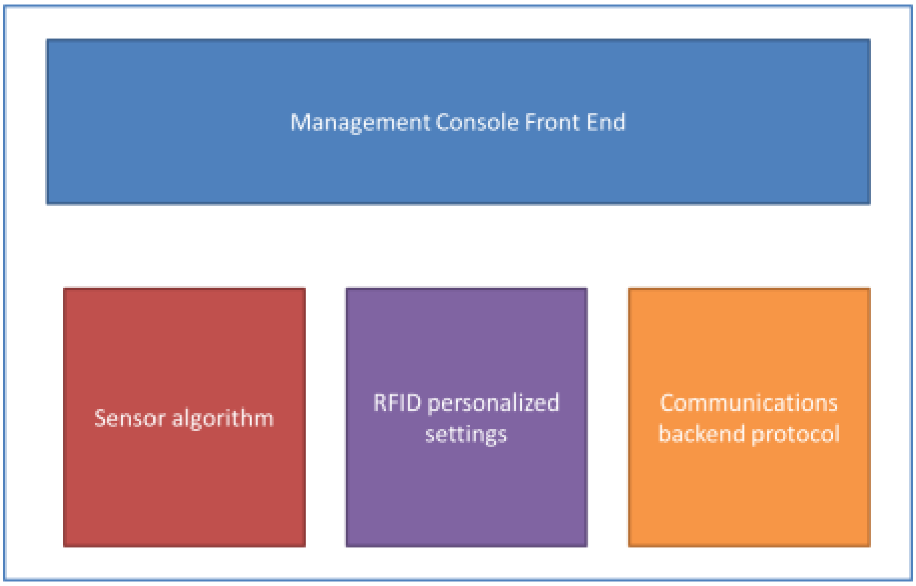

Management Console Front End
Sensor algorithm
RFID personalized settings
Communications backend protocol

Figure 3.2: Block Diagram for Management Console

Figure 3.2 shows the management console block diagram. In the management console, front end consists of UI to let users interact with IHAS easily. Sensor algorithm is how IHAS will respond when sensor data is received by IHAS, whereas communication backend protocol deals with how to communicate with the actual devices. Data from personalized RFID settings could be stored on another subsystem.

3.3 User Interface for Main Menu

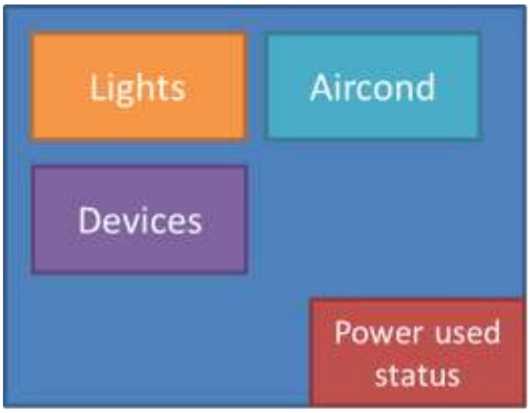


Figure 3.3: Management Console Main Menu

Figure 3.3 shows the management console homepage. Users could launch the application and interact it as how they would with other application. The main menu consists of different categories, each with their own subpage. Users could navigate through different types of devices easily. If there is a power meter attached, the power consumption could also be shown in the main menu.

3.4 User Interface for Lights

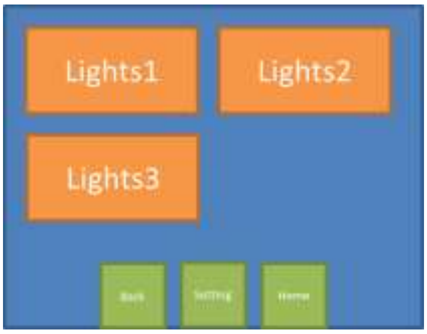


Figure 3.4: Lighting Control Page

The lighting control sub panel is shown in Figure 3.4. Users could see the status of the lights currently, and turn on/off or dim up/down the lights in this panel. Through the setting button the users could add lights, form lighting groups, delete lighting from groups etc. Home button redirect the users back to the main page while back button redirect users from the last page the users came from.

3.5 Automated Plug and Play Coordinator Setup

IHAS is designed to be user friendly and therefore all XBEE configurations should be automated and requires no user intervention. First of such automated setup is to configure the XBEE become a coordinator when it is plugged in to the computer.

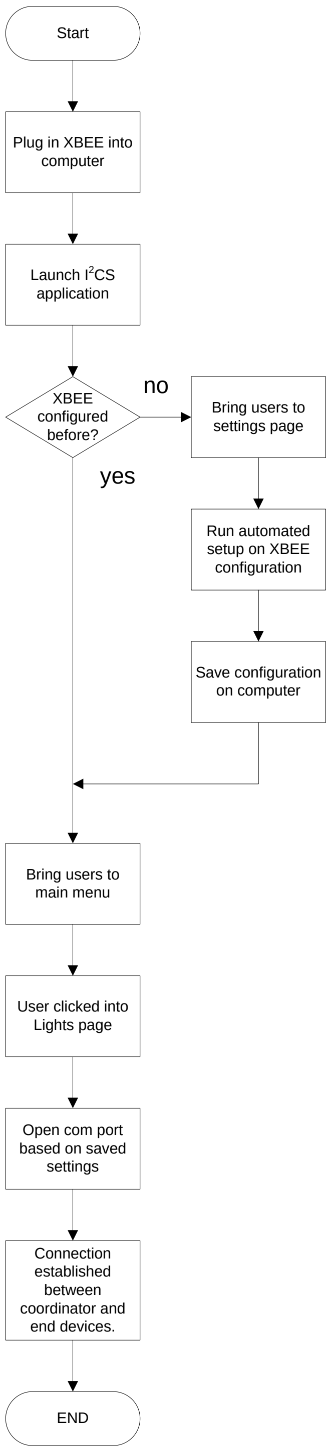

Start
Plug in XBEE into computer
Launch I2CS application
XBEE configured before?
no
yes
Bring users to settings page
Run automated setup on XBEE configuration
Save configuration on computer
Bring users to main menu
User clicked into Lights page
Open com port based on saved settings
Connection established between coordinator and end devices.
END

Figure 3.5: Automated Setup of Coordinator for XBEE

Figure 3.5 shows how IHAS application is able to detect new XBEE devices and bring users directly to settings page. After configuring the XBEE, IHAS application should save the details of the configured XBEE into a file, and return users to main page.

There are few challenges that are presented; one of them being how would the computer know which COM ports is the XBEE? Different XBEE is attached and listed as different COM ports in the computer's device manager. Furthermore, if user has more than one COM devices attached to computer, the IHAS have to find XBEE from the attached devices.

As IHAS application is written in Actionscript 3.0, another hurdle that we have to overcome is that Actionscript 3.0 does not permit file writing to local drive, which is to computer. This is a design decision to avoid flash banners/ web games to install malicious software into computer. Therefore, we have to find our way around this limitation to enable this feature in IHAS.

3.6 Automated Scanning for End Devices Status

Similarly, if users have multiple end devices, IHAS application have to identify the end devices automatically. For example, if the user has been using IHAS setup for some time, and the user wishes to install new light fixture into the kitchen. IHAS have to recognize the newly installed end device and its services provided. IHAS has to show different settings and profile for different "services" offered by end devices. For example, a ceiling fan is different from lights in terms of service provided and configuration method to that particular device.

The steps IHAS should take to scan end devices:

I. Make sure coordinator and end devices are powered up
II. IHAS send "identify yourself" to all end devices in broadcasting mode
III. Wait incoming replies from all end devices.
IV. Time out after 5 seconds, Save all identified end devices to a file
V. Assigned different groups for different kinds of end devices, dimmable lights, normal lights to lights, switches and sensors to inputs, ceiling fan to fan etc.
VI. Show the different groups and function in the IHAS application in an easy to navigate and attractive manner.

During active operations, IHAS has to monitor the connection between end devices and coordinator. For example, if the study room lights controller experience communication breakdown, IHAS has to notify users that particular end devices experience error and should stop users from interacting with that particular end device until communications has been restored.

3.7 Remote Encryption Settings for End Devices

Security is especially important in wireless communications. XBEE supports AES128 encryption; however, it is disabled at default. Therefore, steps have to be taken to re-enable AES128.

I. Make sure coordinator and end devices are powered up
II. Prompt users to enter an encryption key
III. Send encryption key to end devices, enables AES128 encryption and restart end devices at a delayed time* to take effect.
IV. Send command to coordinator to enable AES128 encryption and restart coordinator.
V. Communication re-establish with AES128 encryption

* Coordinator has to restart first to enable end devices to connect to coordinator after end devices restart.

One of the challenges in implementing this feature is how to remote configure all connected XBEE, simultaneously. As mentioned in chapter 2.2.2.2, instead of transparent operation, we have to use API mode to remote configure XBEE.

Another downside of this method is the master encryption key is sent out to XBEE unencrypted, which means the encryption key is prone to be intercepted. This is unavoidable as XBEE does not support session key.

3.8 IHAS Communications Format and Protocol

To communicate effectively between IHAS and end devices, a communication format is established. The criteria of the protocol are compact and flexibility. Since error detecting and correction is handled by the 802.15.4, the protocol does not need error correction which increases complexity and processing time.

Table 3.1: IHAS Communication Protocol

Device	Type	Zone	Number	Data	End Byte
3 bytes	2 bytes	2 bytes	2 bytes	3 bytes	1 byte

Table 3.1 shows the IHAS communication protocol. The first 3 byte dedicates to what device type it is communicating, For example, Lights and Fans have different Device parameters. Type is a communication type. For example, it could be type of sensor/device type or message type. Zone defines how the devices are group, Number is specified devices in a specific zone where data is the information payload and end byte signifies end of packet.

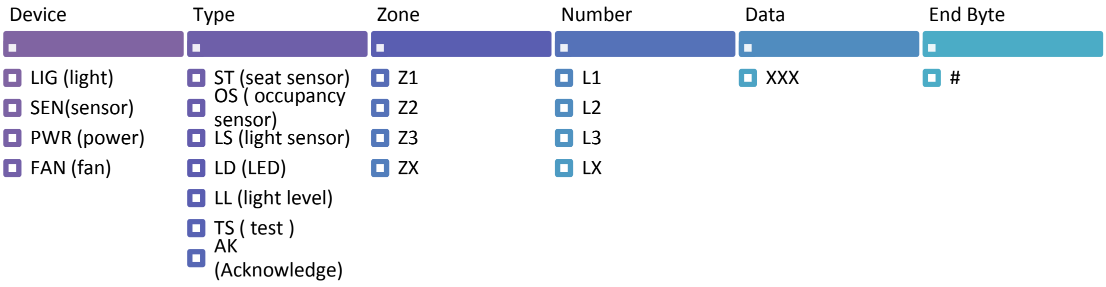


Figure 3.6: IHAS Communication Protocol Example

Figure 3.6 shows an example of IHAS Communication protocol. The protocol could be expended for future expansion. For example, user wanted to turn on kitchen light which is on zone 3 to maximum intensity, IHAS application will send the following frame to end device, “LIGLDZ3L1255#”.

IHAS also receive command from end devices in the same format. This enables IHAS to know exactly which device is requesting/reporting data to IHAS.

3.9 RFID Integration in IHAS

RFID integration with IHAS is another method for users to change settings. Different users with their own RFID tag could trigger different set of devices. For example, imagine IHAS is implemented in an office floor with multiple rooms and cubicles. When the office employee enters the front door, the RFID reader detects his employee RFID tag, it triggers the light and air conditioner for his room and this would cut down the need for human occupancy sensor. When more employees started to come in, IHAS will detects and turn on the devices registered to the RFID respectively. The same applies when the RFID is detected second time, instead of turning on, IHAS will turn off the devices automatically. Figure 3.7 shows how IHAS response when it detects a RFID card.

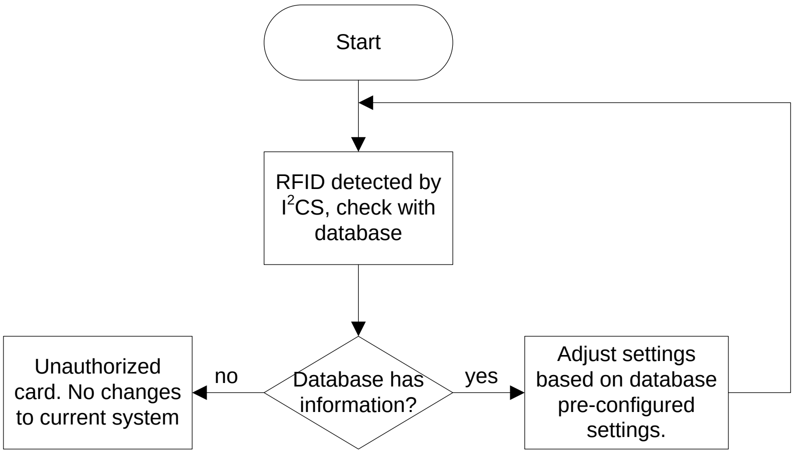

Start
RFID detected by I^{2}CS, check with database
Database has information?
no
yes
Unauthorized card. No changes to current system
Adjust settings based on database pre-configured settings.

Figure 3.7: When IHAS Detects RFID

Before RFID cards can be used with IHAS, it has to first register with IHAS RFID database. The database will store RFID number and the settings it would change to the end devices. Figure 3.8 shows the flow of how a new RFID card could be registered with IHAS.

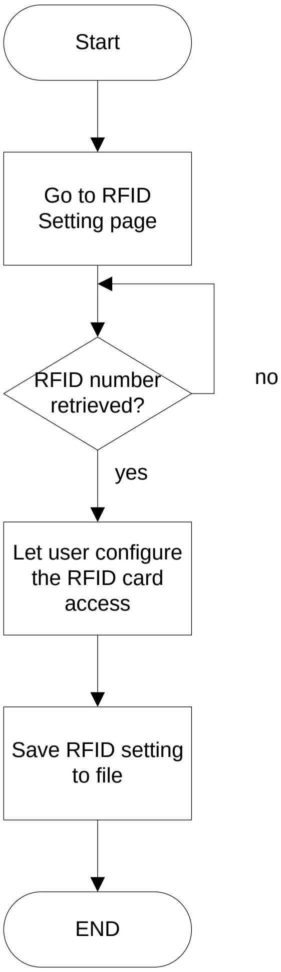

Start
Go to RFID Setting page
RFID number retrieved?
no
yes
Let user configure the RFID card access
Save RFID setting to file
END

Figure 3.8: Registering new RFID card to IHAS database

3.10 Modular Design

IHAS is built with modularity in mind. This concept is shown in Chapter 3.2 for software design; while modular design is used when design with the RFID reader hardware. This practice is good to implement so that we could increase flexibility in implementing IHAS without rework of the whole system. Figure 3.9 shows the modular design of RFID hardware.

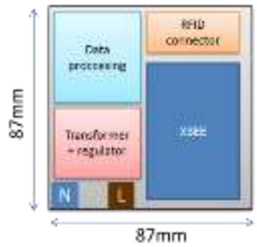


Figure 3.9: RFID Hardware Modular Design

3.11 Smartphone Controlled IHAS

Apart from using IHAS on the computer, users could also use extend their home automation control using their smartphones. There are 2 ways of implementing this method, which is using:

I. A dedicated smartphone client. This method requires a separate development for the smartphone that enables the best user experience as it is coded specifically for the smartphone. The smartphone application connects to IHAS application through Wi-Fi network which enables users to seamlessly control devices from their phone.

II. A VNC client. This method is inferior as compared to dedicated smartphone client. This method requires a VNC server setup on the IHAS Application computer and a VNC client on the smartphone. The VNC then streams the desktop environment from the computer to the smartphone, enabling users to control devices from their phone.

Although method 1 is superior as compared to method 2, method 2 is chosen as the development time is shorter for the current version of IHAS.

3.12 Compatibility of IHAS with Commercial Products

IHAS is not just a proprietary method that only works with designated devices. IHAS is built to works with both commercial systems and proprietary devices. We shall discuss a potential design solution on energy savings for a university campus. The design solution consists of the following features:

- Air conditioner could be controlled by central management console and/or by sensor deployed around the campus

- Lighting system could be intelligently controlled by schedule based control, occupancy based control or central management controlled.

- Central monitoring system is needed to track energy usage and to fine tune energy usage profile, which could be achieved by using XBEE and an easy to manage control panel which integrates Heating Ventilation and Air Conditioning (HVAC), lighting and device control.

The potential design solution could be implemented with commercial products such as Schneider Electric range of energy management solution with IHAS. The proposed design solution is strongly collaborated with (Wei, 2010).

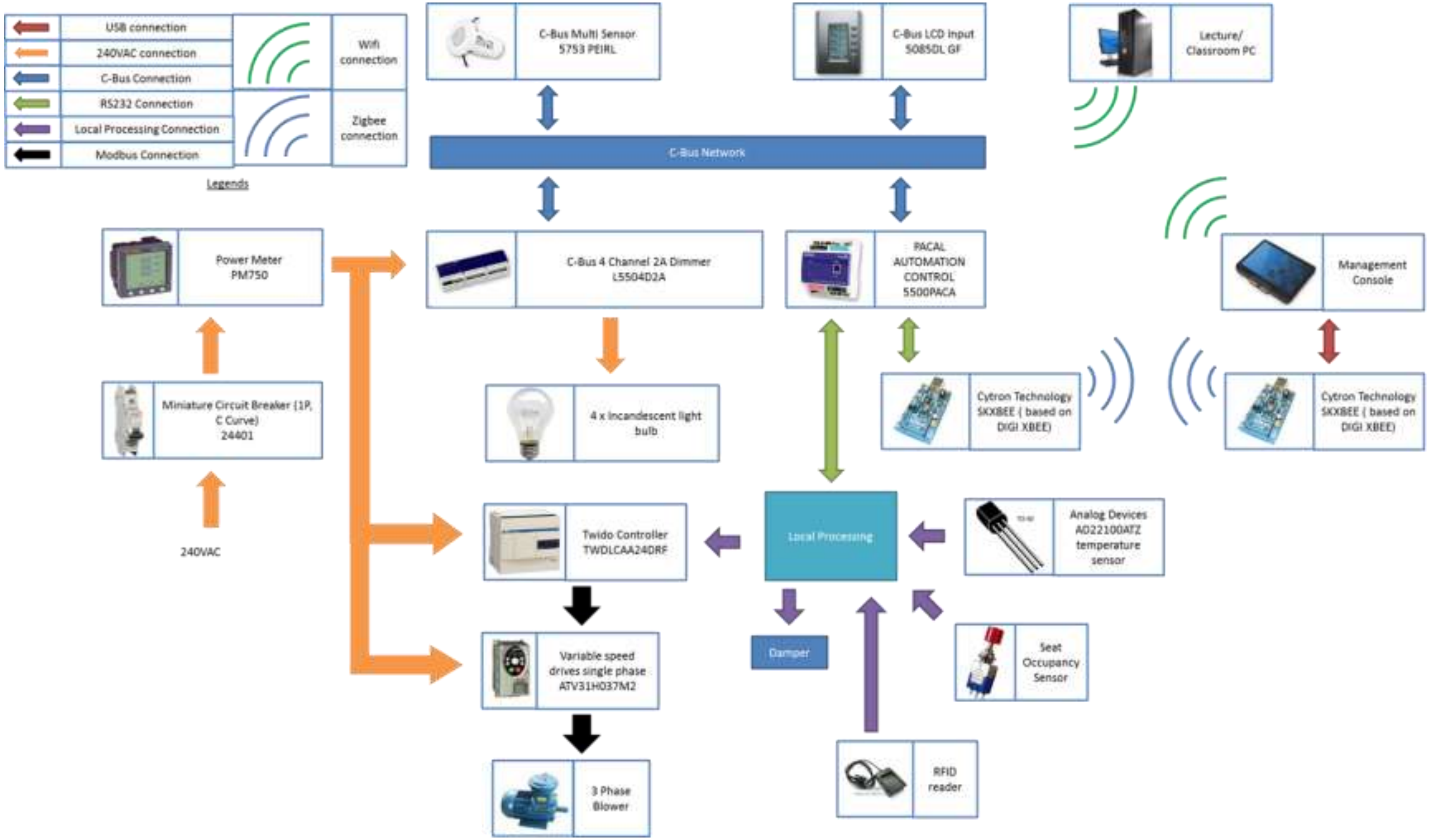

USB connection
240VAC connection
C-Bus Connection
RS232 Connection
Local Processing Connection
Modbus Connection
Wifi connection
Zigbee connection
Legends
C-Bus Multi Sensor 5753 PEIRL
C-Bus LCD Input 5085DL GF
Lecture/ Classroom PC
C-Bus Network
Power Meter PM750
C-Bus 4 Channel 2A Dimmer L5504D2A
PACAL AUTOMATION CONTROL 5500PACA
Management Console
Miniature Circuit Breaker (1P, C Curve) 24401
4 x Incandescent light bulb
Cytron Technology SKXBEE (based on DIGI XBEE)
Cytron Technology SKXBEE (based on DIGI XBEE)
240VAC
Twido Controller TWDLCAA24DRF
Local Processing
Analog Devices AD22100ATZ temperature sensor
Variable speed drives single phase ATV31H037M2
Damper
Seat Occupancy Sensor
3 Phase Blower
RFID reader

Figure 3.10: Design Solution for Energy Savings On Campus

Referring to Figure 3.10, the power enters the circuit breaker and passes through the power meter, which measured the power consumption of all the devices. Then, the power is distributed to four channels 2A dimmer to power up the lights, and to power up the motor and PLC. The C-bus multi sensor sense human occupancy and ambient light level and relays the information to C-bus network while C-Bus LCD input is used for local control for lighting and HVAC control. The Pacal automation control also communicates with the local processing module via another RS232 connection. All sensor data and device status is transmitted via a XBEE transceiver for further processing.

The PACAL automation controller enables the integration of IHAS with C-Bus network. From there, we are able to get sensor data, power usage from power meter, lights status from respective dimmer and display on the same screen, which is on the IHAS management console.

If the system is integrated with classroom time management system, teachers can book time for specific lecture class that reduces conflicts among lecturer regarding extra class’s venue. Also, the management console not only be able to control lighting and HVAC system, it could also remotely shutdown, restart or sleep unused pc through Wi-Fi. Figure 3.11 shows how could the management console for this energy management console looks like.

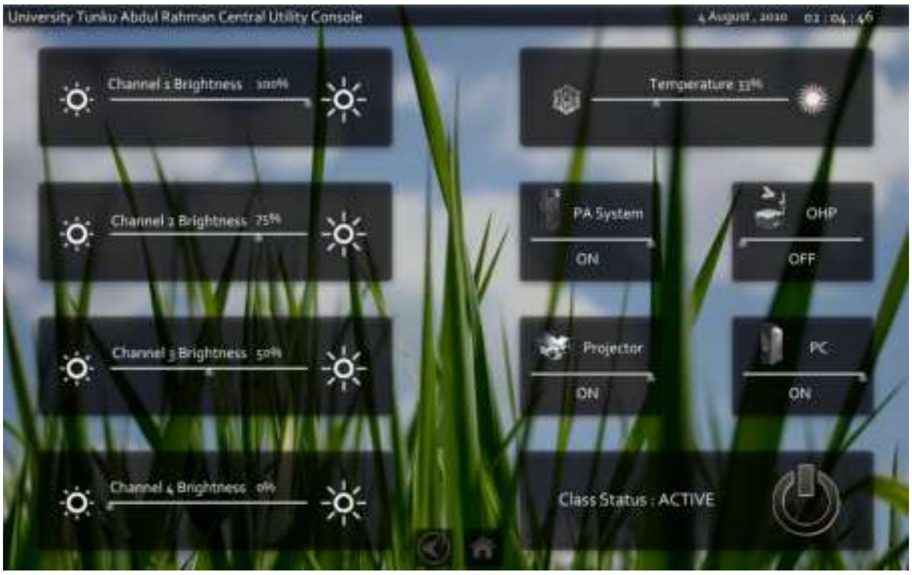


Figure 3.11: Design Concept on management console

CHAPTER 4

IMPLEMENTATION OF IHAS AND OUTCOMES

4.1 Sending Data through a Com Port

4.1.1 Implementation

The first step in this system is to send a data through Actionscript 3.0 using mdm zinc 3.0 scripts. Since IHAS uses sliders frequently, this test uses sliders to test. An Adobe Flash file with the following code is created:

```
//Actionscript initialization
import fl.controls.Slider;
import fl.events.SliderEvent;
import mdm.*;
mdm.Application.init(this);
//initialization ends
```

Initialization of actionscripts, importing relevant library and extensions

```
//setup of event listener
s.addEventListener(SliderEvent.CHANGE,
announceChange);
//end of event listener
```

Event listener that will be triggered if slider value has change

```
//Initialization of com ports
var availablePorts:String = mdm.COMPort.ports;
mdm.Dialogs.prompt(availablePorts);
mdm.COMPort.open(2, 9600, 8, "N", 1, "OFF");
//Initialization of com ports end
```

MDM Zinc 3.0 scripts that initializes the com port

```
//Com port functions
mdm.COMPort.onCOMPortData =
function(myObject:Object){
  mdm.Dialogs.prompt(myObject.data);
};
//Com port functions end
```

Com port function call in the event of data received

```
//functions calls
function announceChange(v:SliderEvent):void {

        mdm.COMPort.send(v.target.value);
}
//end function call
```

Sends the data out to the com port when the slider value has change

After setting up the flash, this application is compiled to a native flash file, .SWF. After that, the SWF file is rebuilded using Zinc 3.0 into an EXE excutable file that includes all the neccesary mdm scripts library file, as shown in Figure 4.1.

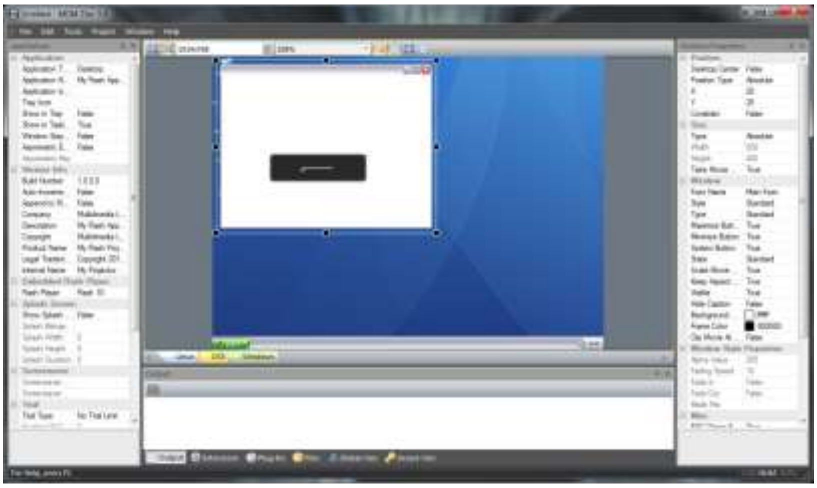
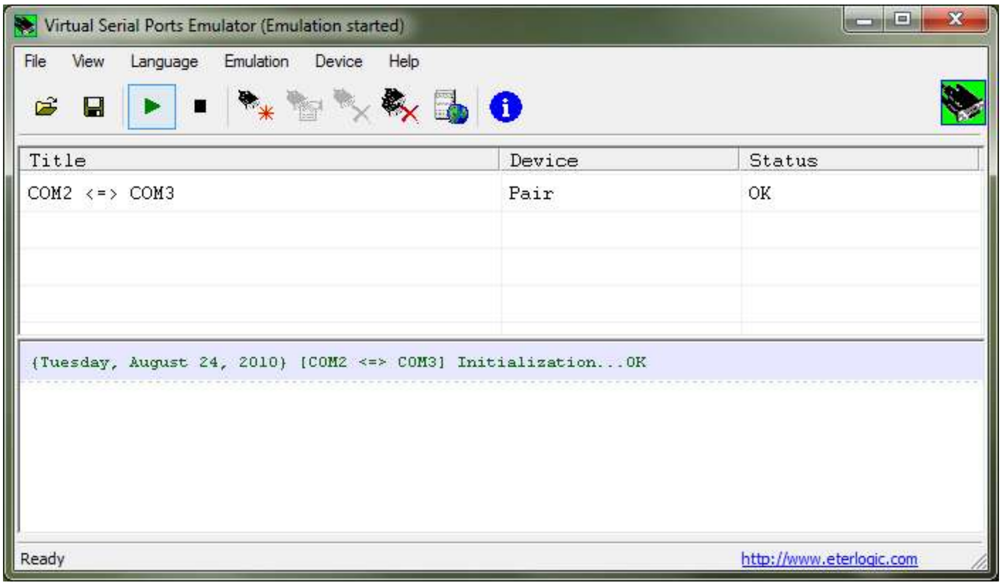


Figure 4.1: Zinc 3.0

When the build is completed, virtual com port and com port data emulator is setup as shown in Figure 4.2 and Figure 4.3.

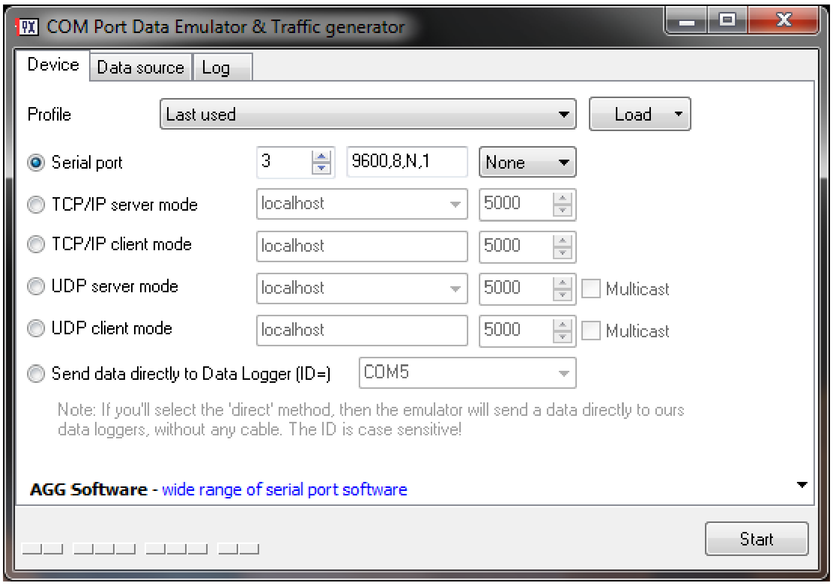


Title	Device	Status
COM2 <=> COM3	Pair	OK

```
(Tuesday, August 24, 2010) [COM2 <=> COM3] Initialization...OK
```

Figure 4.2: Virtual Serial Com port

Figure 4.2 shows that COM2 and COM3 ports has been opened and is linked together. Note that our test application is connected to COM2.


COM Port Data Emulator & Traffic generator
Device
Data source
Log
Load
AGG Software - wide range of serial port software
Start

Profile: Last used

● Serial port: 3, 9600,8,N,1, None

○ TCP/IP server mode: localhost, 5000

○ TCP/IP client mode: localhost, 5000

○ UDP server mode: localhost, 5000, □ Multicast

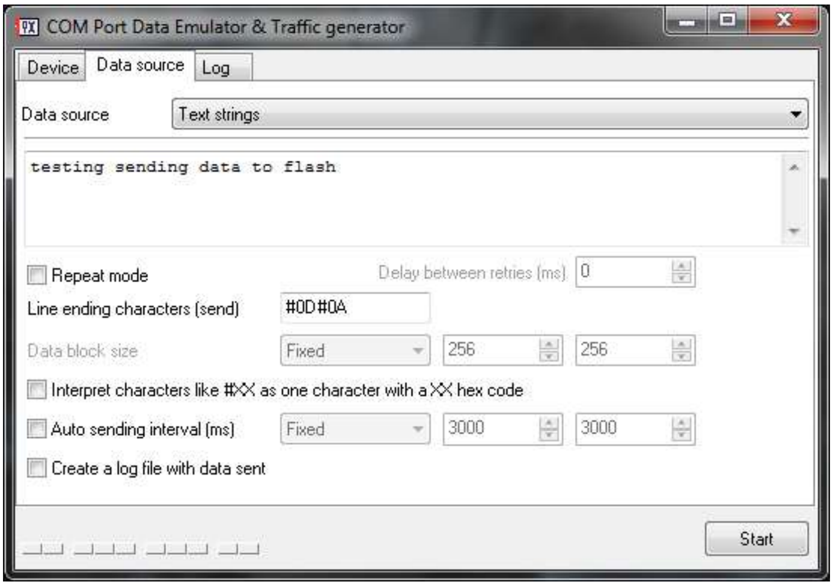
○ UDP client mode: localhost, 5000, □ Multicast

○ Send data directly to Data Logger (ID=): COM5

Note: If you'll select the 'direct' method, then the emulator will send a data directly to ours data loggers, without any cable. The ID is case sensitive!

Figure 4.3: Com Port Emulator Setup

Figure 4.3 shows the setup of Com port emulator which is connected to COM3 and is running at the same baud rate, same encoding scheme with the program.



Data source: Text strings

```
testing sending data to flash
```

□ Repeat mode — Delay between retries (ms): 0

Line ending characters (send): #0D#0A

Data block size: Fixed 256 256

□ Interpret characters like #XX as one character with a XX hex code

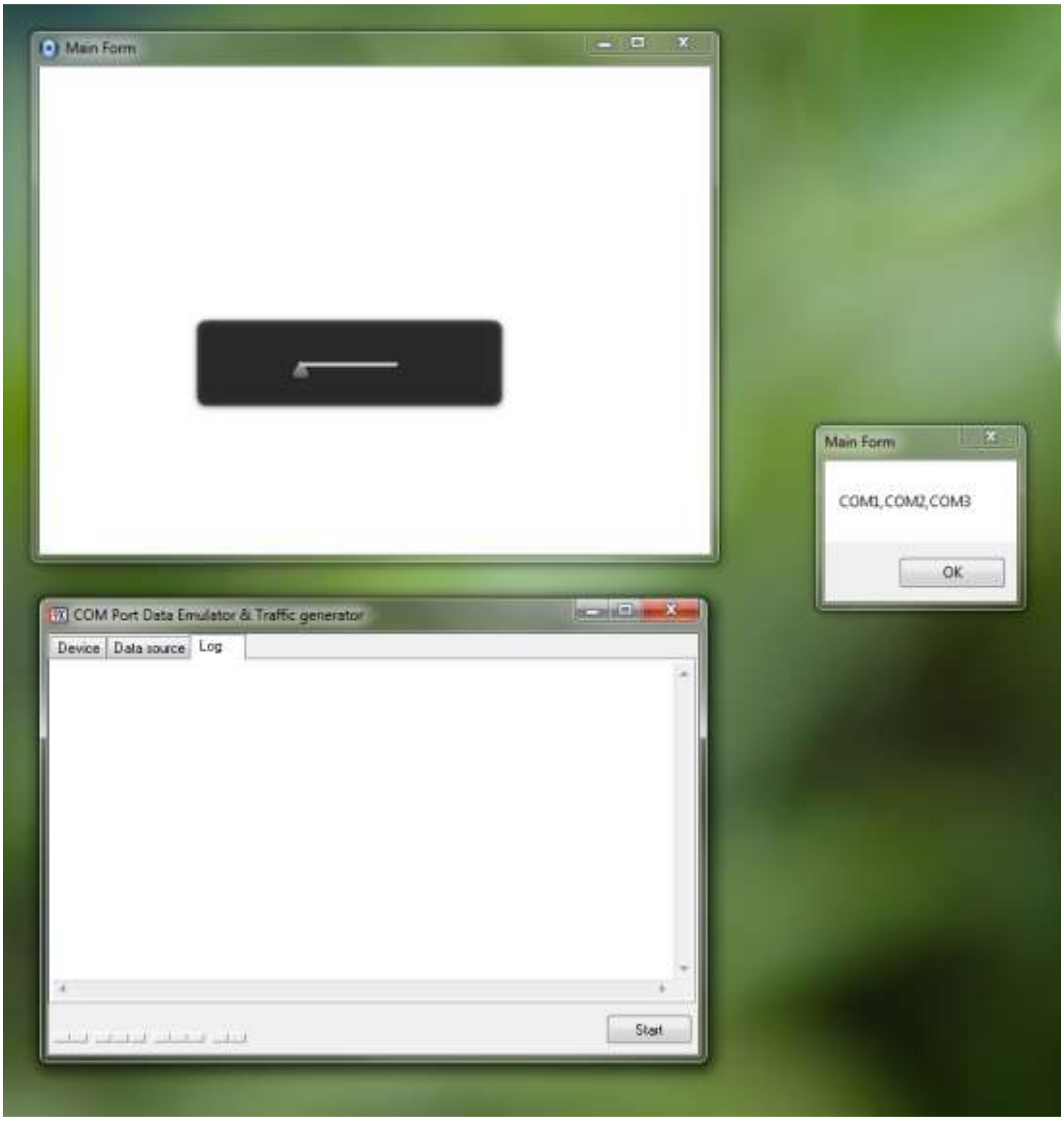
□ Auto sending interval (ms): Fixed 3000 3000

□ Create a log file with data sent

Figure 4.4: Configuring Data Emulator to Send Data

Figure 4.4 shows how to configure a com port data emulator to send text data to the test application.

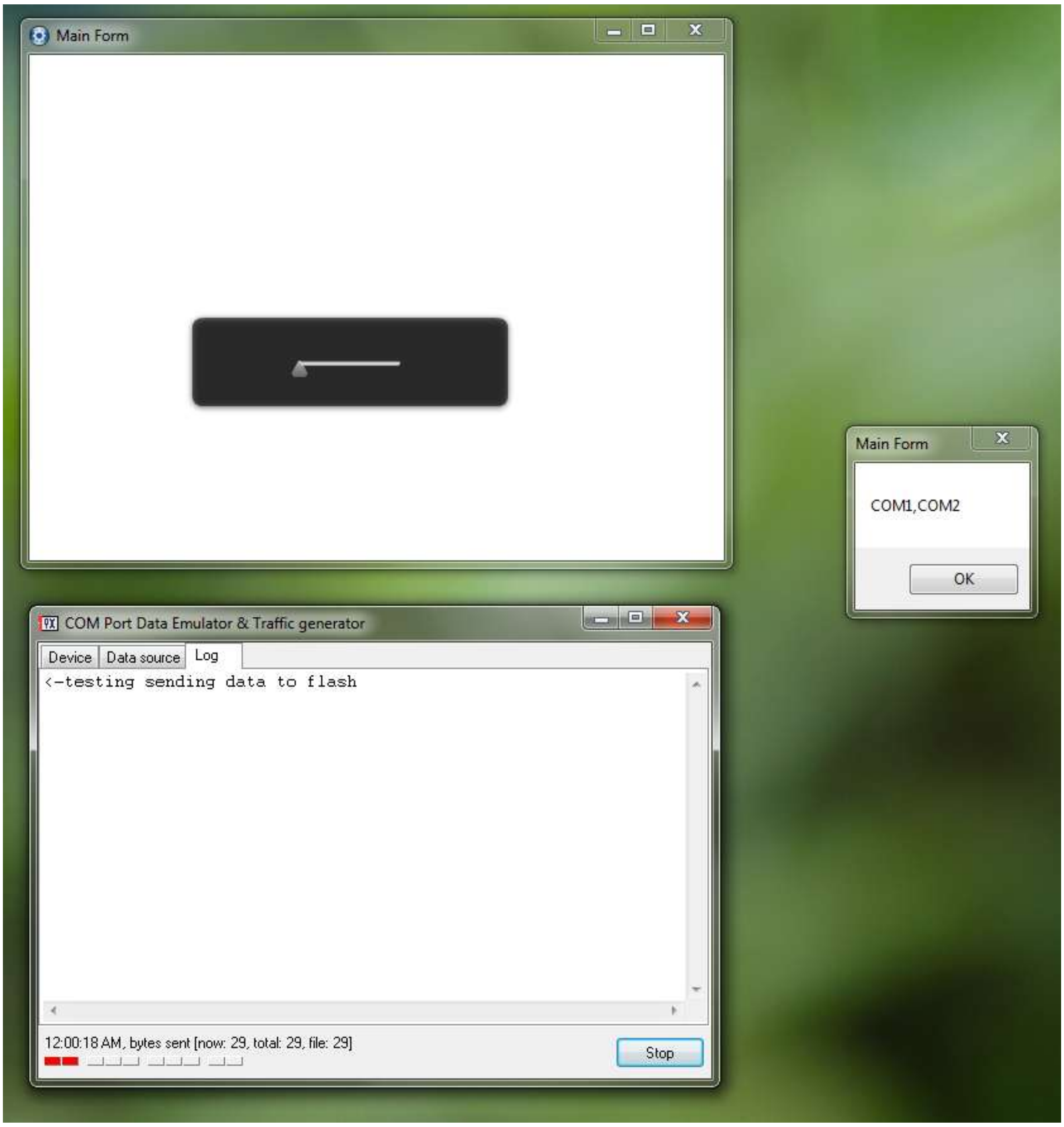
4.1.2 Outcome



COM1,COM2,COM3

Figure 4.5: Scanning Ports

Figure 4.5 shows initial port scan of free com ports before starting of com port emulator returns 3 com ports, 1 physical com port and 2 virtual com ports, COM2 and COM3.



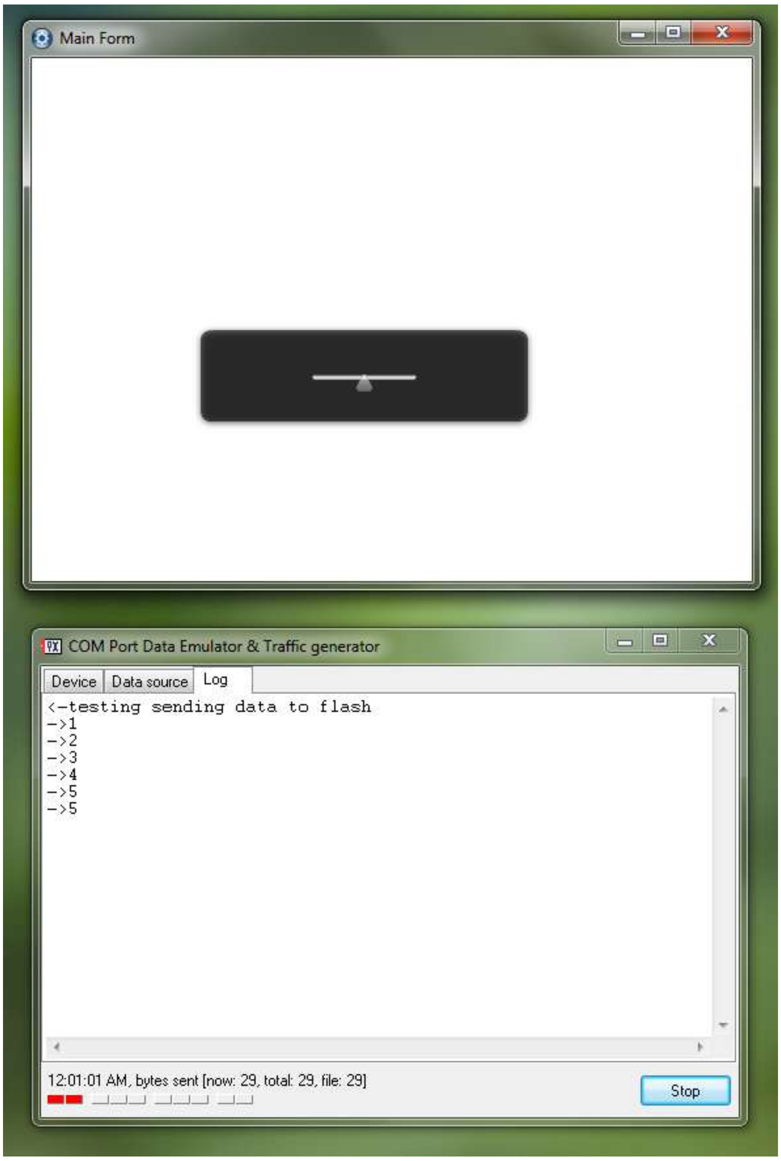
COM1,COM2

```
<-testing sending data to flash
```

```
12:00:18 AM, bytes sent [now: 29, total: 29, file: 29]
```

Figure 4.6: Started Com Port Emulator Before Starting Up Application

Figure 4.6 shows that COM3 has been connected to Com Port Data Emulator before starting up the application, resulting a returned of free com port of only COM1 and COM2.

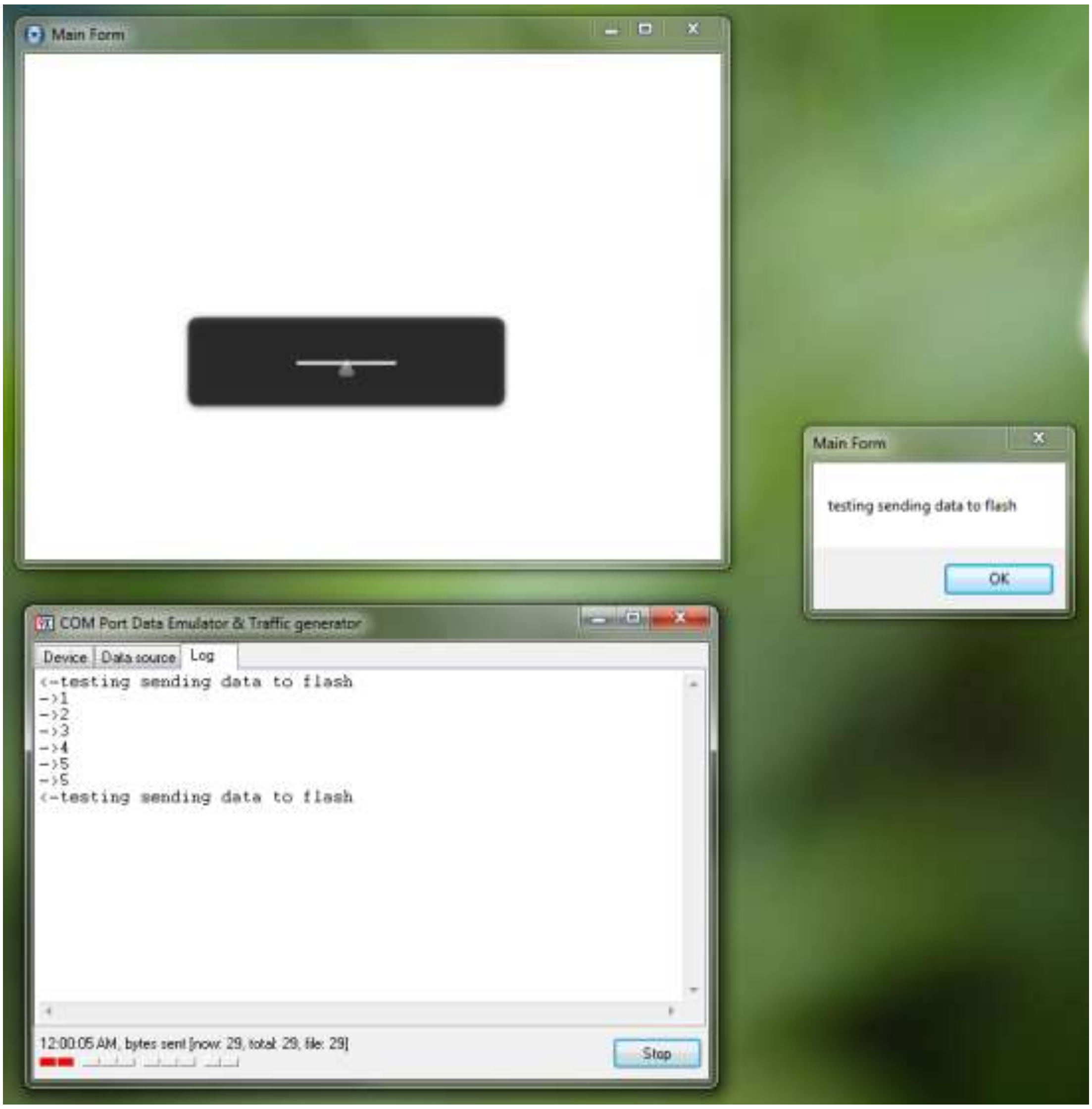


```
<-testing sending data to flash
->1
->2
->3
->4
->5
->5
```

12:01:01 AM, bytes sent [now: 29, total: 29, file: 29]

Figure 4.7: Result from Application Testing (Send)

Figure 4.7 show the application is able to send a continuous stream of change value through a com port. Since the slider value has a maximum value of 10 and minimum value of 0, the Figure show a transition from 0 to 5 translate into half the slider bar.



```
<-testing sending data to flash
->1
->2
->3
->4
->5
->5
<-testing sending data to flash
```

12:00:05 AM, bytes sent [now: 29, total: 29, file: 29]

testing sending data to flash

Figure 4.8: Result from Application Testing (Receive)

Figure 4.8 shows that when the data emulator sends a string of text file to the application, the application responds by showing out on a dialog, which is a preprogram actions.

4.2 Design a User Friendly UI

4.2.1 Implementation

Using Adobe Flash, we try to create a UI following the design guidelines that has been discussed on Chapter 2.4.

4.2.2 Outcome

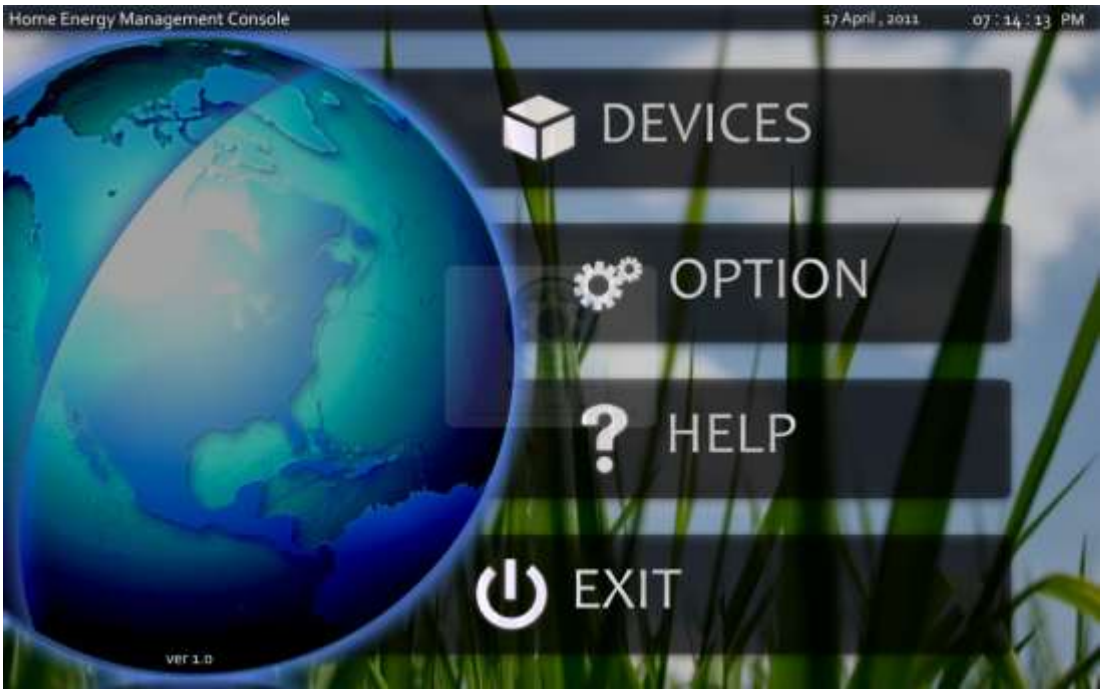


Figure 4.9: Main Menu

Figure 4.9 shows the main menu of IHAS application which is named "Home Energy Management Console". The main menu has 4 options, which are Devices, Option, Help and Exit. Devices will bring users to device control such as lighting control. Option will bring users to setup the XBEE. Help is to assist user in using this program while Exit exits the program.

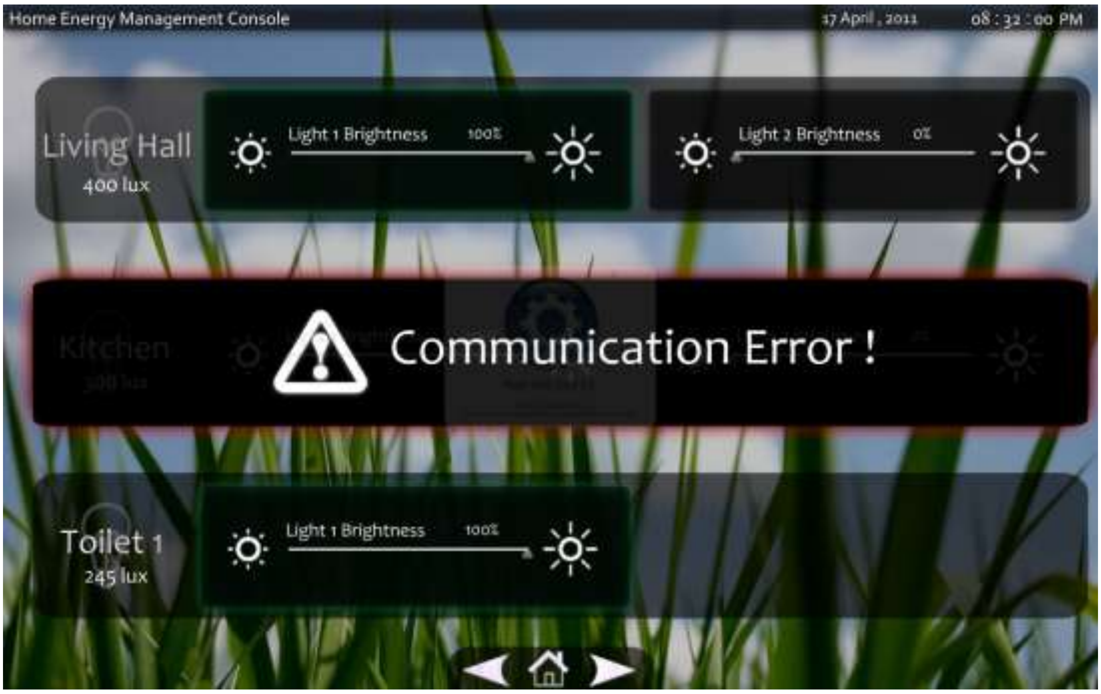


Figure 4.10: Lights Control Page (full)

Figure 4.10 shows the lights control page. The lights are divided into sections according to the zones. Instead of naming the sections Zone 1 or Zone 2, users could rename them into meaningful phrases to them, such as Kitchen, Living Hall and Toilet. Below the zone description is the ambient light level detected by the light level sensors. The slider controls individual lighting where green glow around the border of the light is an indication that the light is on.

When the communication breakdown between a particular zone and the coordinator, warning is displayed as showed in Figure 4.10 and changes could not be made on the particular zone until communications has been restored.

The buttons on the bottom of the page are back, home and next. These buttons help users to navigate through devices.

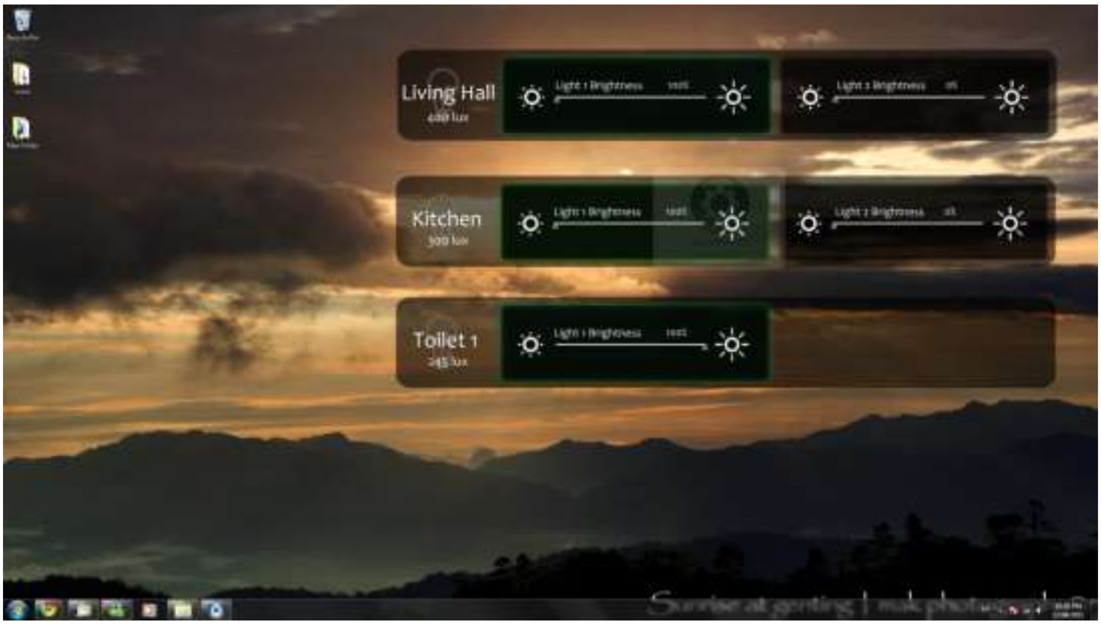
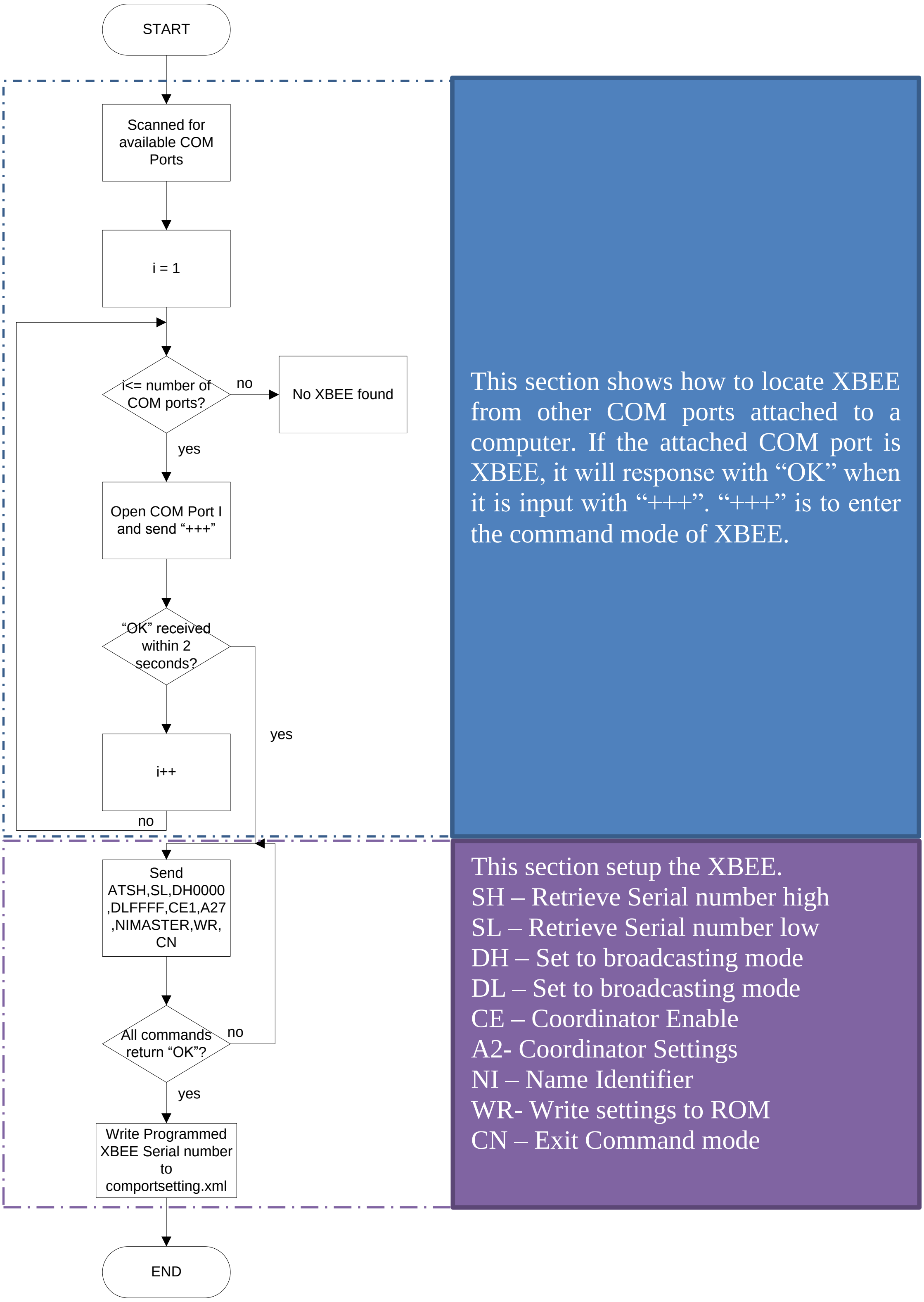


Figure 4.11: Lights Control (desktop mode)

Figure 4.11 shows the lights control on desktop mode, where users could use their computer and interact with the devices at the same time. In desktop mode, IHAS lose the ability to configure the XBEE. Therefore, users have to run IHAS full application once before they could launch IHAS desktop mode.

4.3 Automated Setup for XBEE Coordinator

4.3.1 Implementation


START
Scanned for available COM Ports
i = 1
i<= number of COM ports?
no
No XBEE found
yes
Open COM Port I and send "+++"
"OK" received within 2 seconds?
yes
i++
no
Send ATSH,SL,DH0000 ,DLFFFF,CE1,A27 ,NIMASTER,WR, CN
All commands return "OK"?
no
yes
Write Programmed XBEE Serial number to comportsetting.xml
END

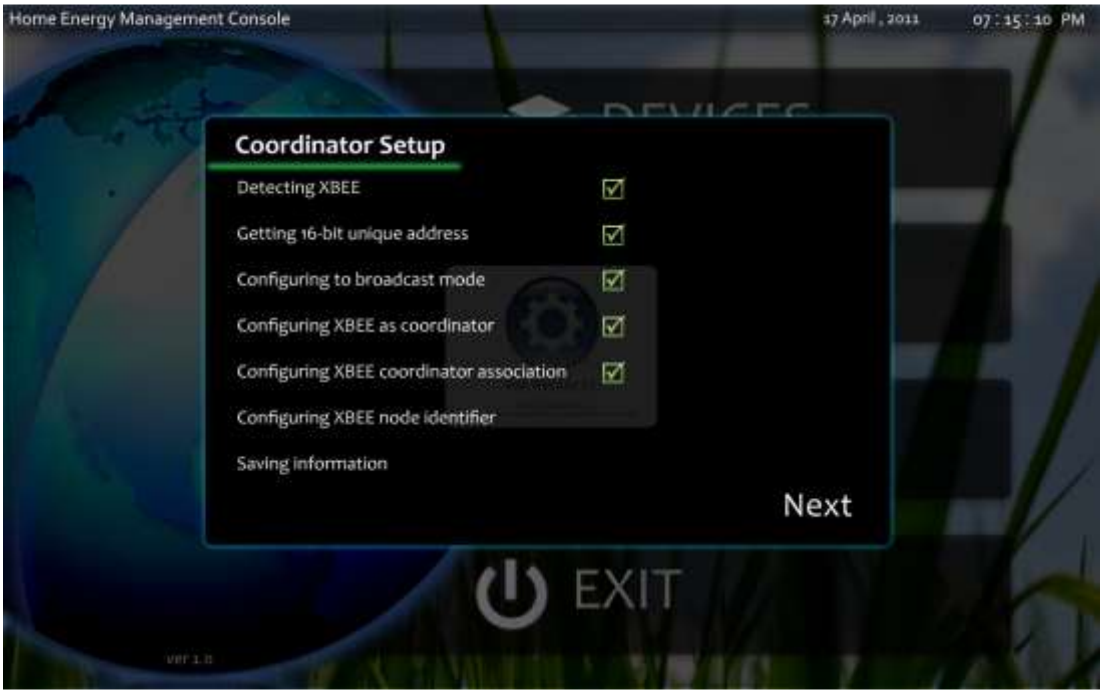
This section shows how to locate XBEE from other COM ports attached to a computer. If the attached COM port is XBEE, it will response with "OK" when it is input with "+++". "+++" is to enter the command mode of XBEE.

This section setup the XBEE.
SH – Retrieve Serial number high
SL – Retrieve Serial number low
DH – Set to broadcasting mode
DL – Set to broadcasting mode
CE – Coordinator Enable
A2- Coordinator Settings
NI – Name Identifier
WR- Write settings to ROM
CN – Exit Command mode

Figure 4.12: Flow chart to configure XBEE settings automatically

Figure 4.12 shows how to scan for an attached XBEE and related settings to program a XBEE into a coordinator. As discussed in Chapter 2.2.2.1, “+++” is used to enter AT command mode of the XBEE. If the com port attached is polled with “+++”, XBEE will return “OK”, which is used to identify that specific port is attached to a XBEE.

4.3.2 Outcome


Home Energy Management Console

17 April , 2011 07 : 15 : 10 PM

Coordinator Setup

Detecting XBEE ☑

Getting 16-bit unique address ☑

Configuring to broadcast mode ☑

Configuring XBEE as coordinator ☑

Configuring XBEE coordinator association ☑

Configuring XBEE node identifier

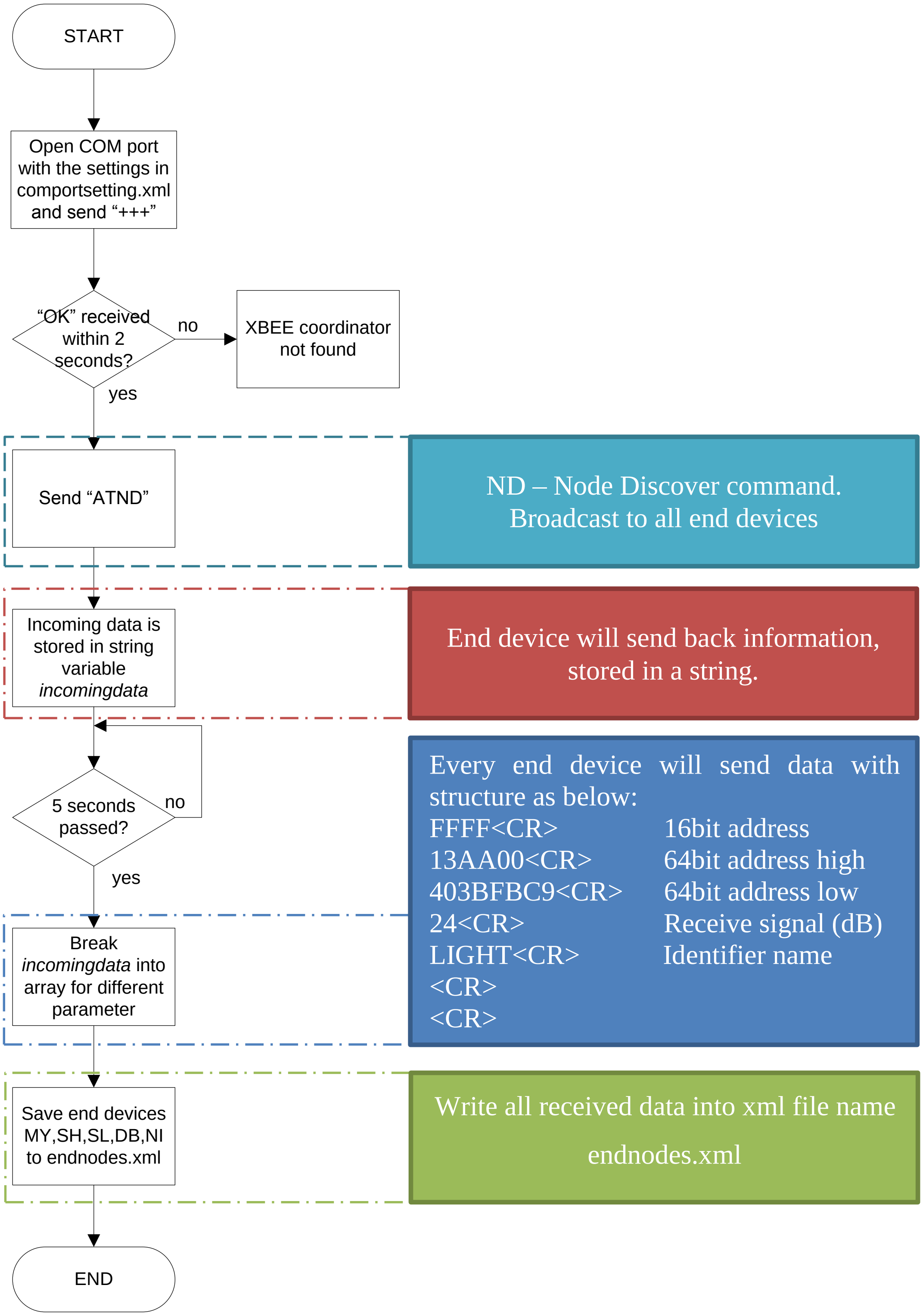
Saving information

Figure 4.13: Plug and Play XBEE Coordinator Easy Setup

Figure 4.13 shows the visual indicator of the XBEE configuring process, where IHAS get the 16-bit unique address from XBEE, configure it to broadcast mode, enable coordinator mode in XBEE, change coordinator settings to allow channel assignment, allow panID assignment and allows association. These settings are used to allow end device to connect to the coordinator.

4.4 Automated Scanning for End Devices Status

4.4.1 Implementation


START
Open COM port with the settings in comportsetting.xml and send "+++"
"OK" received within 2 seconds?
no
XBEE coordinator not found
yes
Send "ATND"
Incoming data is stored in string variable *incomingdata*
5 seconds passed?
no
yes
Break *incomingdata* into array for different parameter
Save end devices MY,SH,SL,DB,NI to endnodes.xml
END

ND – Node Discover command. Broadcast to all end devices

End device will send back information, stored in a string.

Every end device will send data with structure as below:

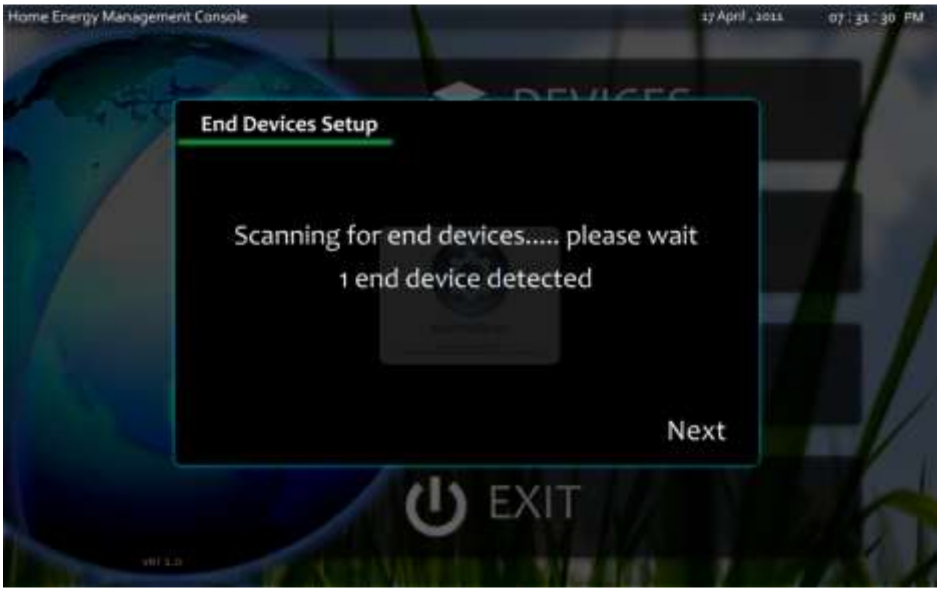
FFFF<CR>	16bit address
13AA00<CR>	64bit address high
403BFBC9<CR>	64bit address low
24<CR>	Receive signal (dB)
LIGHT<CR>	Identifier name
<CR>	
<CR>	

Write all received data into xml file name endnodes.xml

Figure 4.14: Steps to Detect and Save End Devices to Database

Figure 4.14 shows the flow chart on how to scan end devices and keep the data into a data file. When the Node Discover is sent to the end devices, all the end devices will return their 16 bit short address, 64 bit unique address and also the identifiers name. The coordinator also reports the received signal strength of that particular end device. All these data is stored in a data file that is used for IHAS to recognize which XBEE controls which device from the node identifier name.

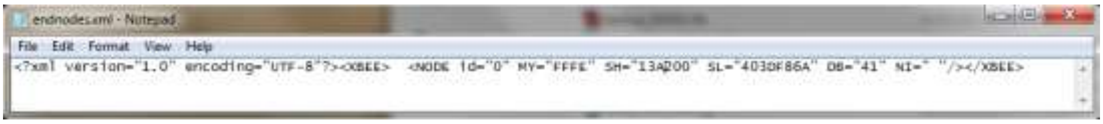
4.4.2 Outcome



Scanning for end devices..... please wait
1 end device detected

Figure 4.15: End Device Scanning

Figure 4.15 shows the IHAS will report how many detected XBEE after a 5 seconds timeout. At the same time, IHAS will write to a xml file with the acquired data from the end devices. An example is shown in Figure 4.16.



```
<?xml version="1.0" encoding="UTF-8"?><XBEE>  <NODE id="0" MY="FFFE" SH="13A200" SL="403DF86A" DB="41" NI=" "/></XBEE>
```

Figure 4.16: endnodes.xml Datafile

4.5 Remote Encryption Settings for End Devices

4.5.1 Implementation

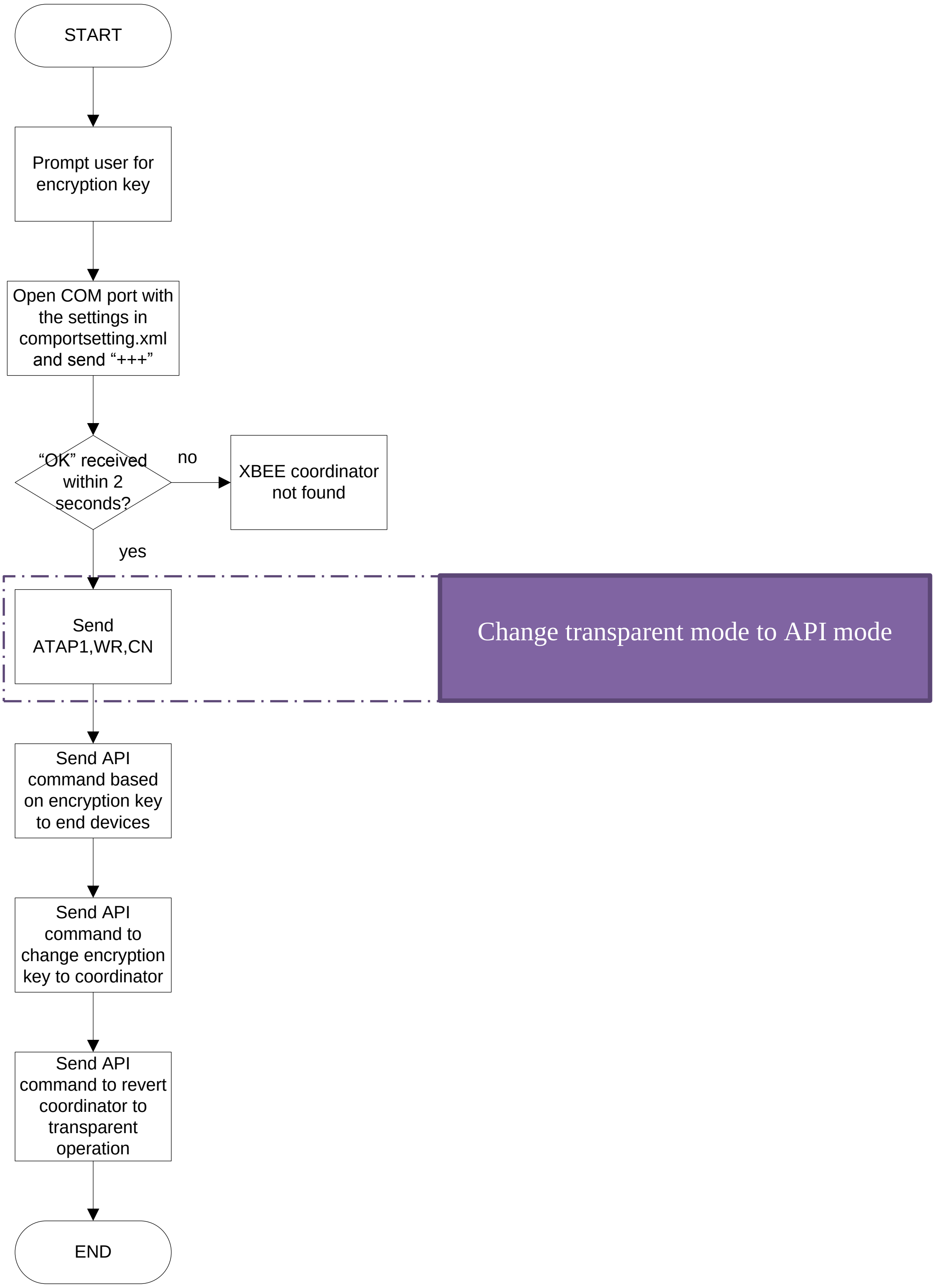

START
Prompt user for encryption key
Open COM port with the settings in comportsetting.xml and send "+++"
"OK" received within 2 seconds?
no
XBEE coordinator not found
yes
Send ATAP1,WR,CN
Change transparent mode to API mode
Send API command based on encryption key to end devices
Send API command to change encryption key to coordinator
Send API command to revert coordinator to transparent operation
END

Figure 4.17: Remote Encryption Setting

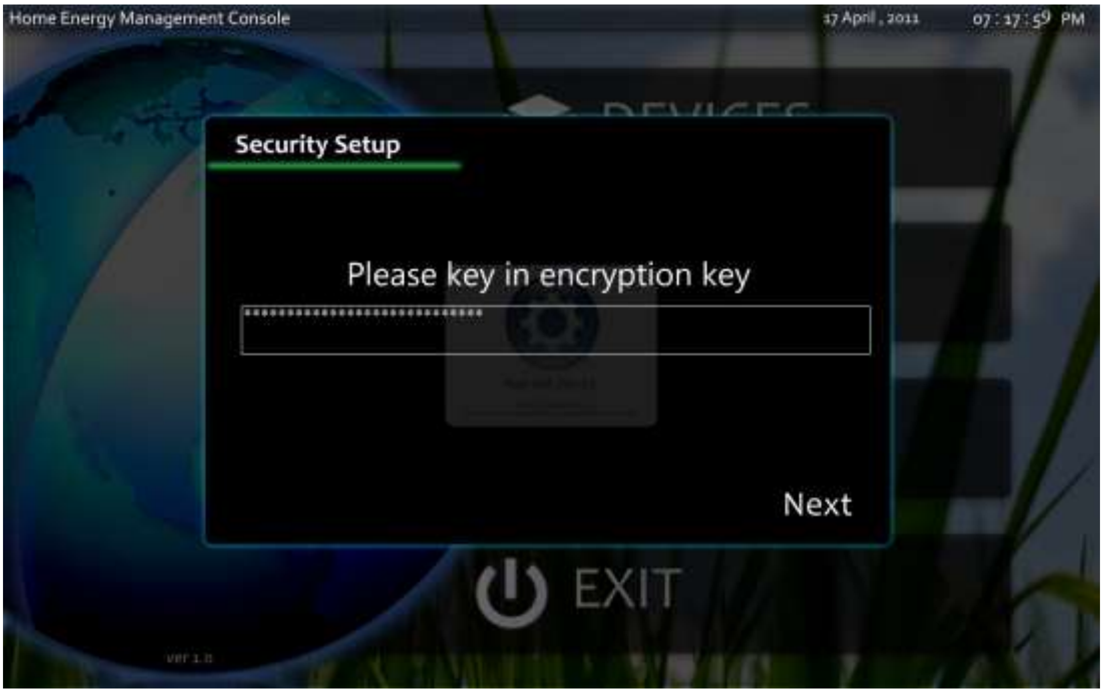
Figure 4.17 shows how to remotely change an encryption key and enable AES128 encryption. API mode is needed as transparent operation does not support remote configuration. Once the user keys in the encryption key, IHAS will process an API frame before it could be sent out as listed in Chapter 2.2.2.2. An example of sending an API command of enabling an encryption key of end devices having a short address of FFFE will be as shown in Figure 4.18:

7E	00	10	17	01	00	00	00	00	00	00	FF	FF	FF	FE	02	45	45	01	61

🟦	**Start Delimiter**
🟪	**Frame length in Hexadecimal**
🟦	**API identifier for remote configuration**
🟫	**Acknowledgment**
🟥	**64 bit unique address (configured to use 16 bit short address in this case)**
🟩	**16 bit short address**
🟧	**Apply changes after received command flag**
🟪	**AT command in hexadecimal**
🟦	**Parameter**
⬜	**Checksum**

Figure 4.18: API Frame Example

4.5.2 Outcome



Please key in encryption key

****

Figure 4.19: Security Setup

Users are prompted to key in the 32 Hex character encryption key as shown in Figure 4.19. However, when the encryption key is sent, it is unencrypted. Potential interception of encryption key is possible within the timeframe of sending encryption key to the end devices.

4.6 RFID Integration with IHAS

4.6.1 Implementation

Using RFID reader as mentioned in Chapter 3.9, modifications have to be made to interface with XBEE. Voltage level conversion chip is required as discussed in Chapter 2.3.

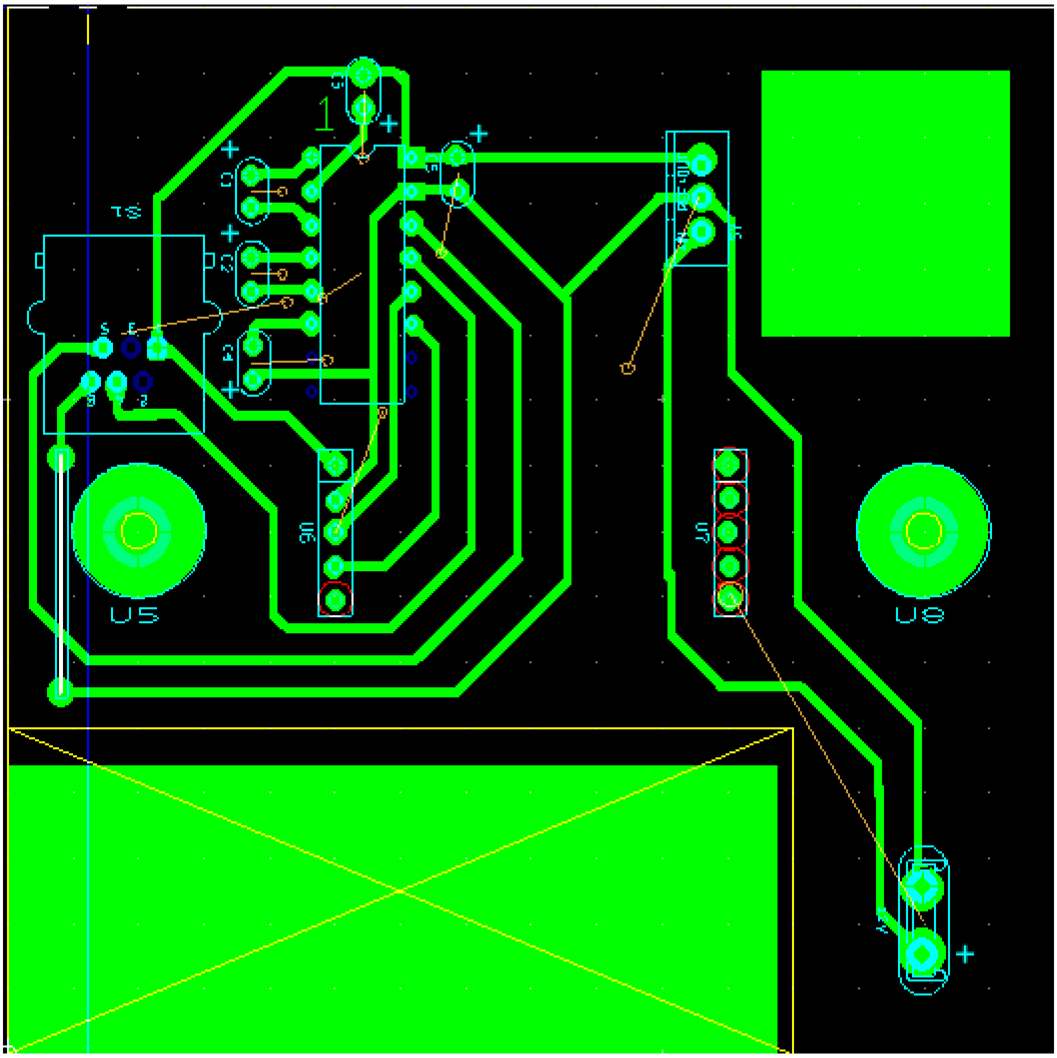


Figure 4.20: Modular design of RFID Conversion Circuit

Figure 4.20 shows a modular design in RFID conversion circuit design. 24V input header is located on the lower right of the circuit. This RFID is powered by 240V and the voltage conversion and regulation circuit is a commercially available power supply. XBEE can be detached quickly as it is attached to the header on the board.

4.6.2 Outcome

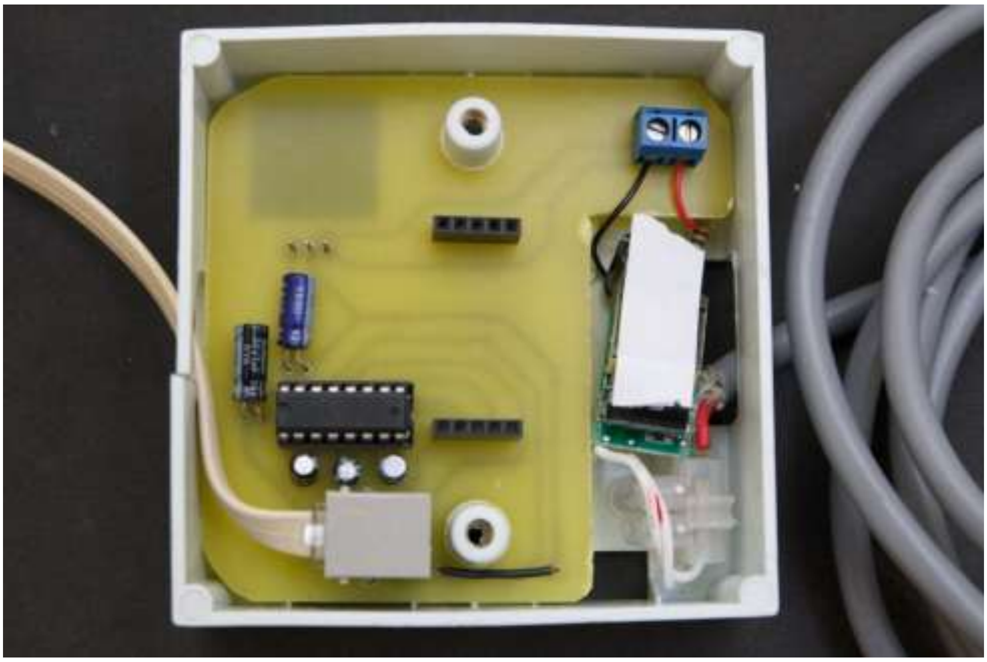


Figure 4.21: RFID Conversion Circuit and Power Conversion and Regulation Circuit

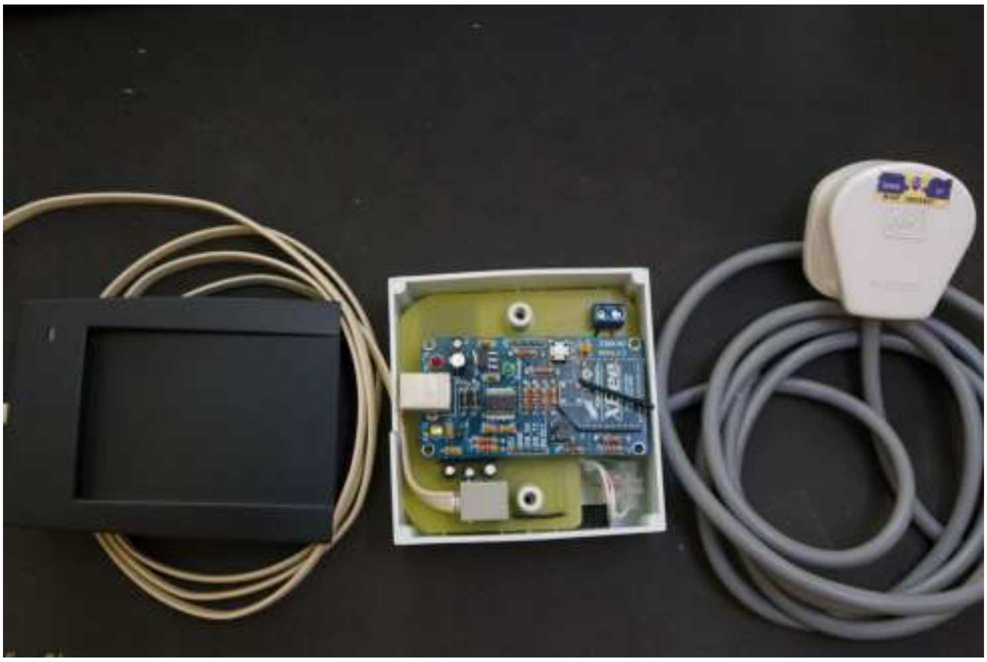


Figure 4.22: Complete RFID Hardware

Figure 4.21 and 4.22 shows the completed RFID hardware. The RFID hardware is powered by 240VAC and transmits RFID tag number wirelessly to IHAS management console. Figure 4.21 shows the 240VAC to 24VDC conversion circuit on bottom right, under an electrical insulating tape.

4.7 Smartphone Controlled IHAS

4.7.1 Implementation

Using a VNC server, IHAS could stream the desktop to the smartphone, which is running a VNC client.

4.7.2 Outcome

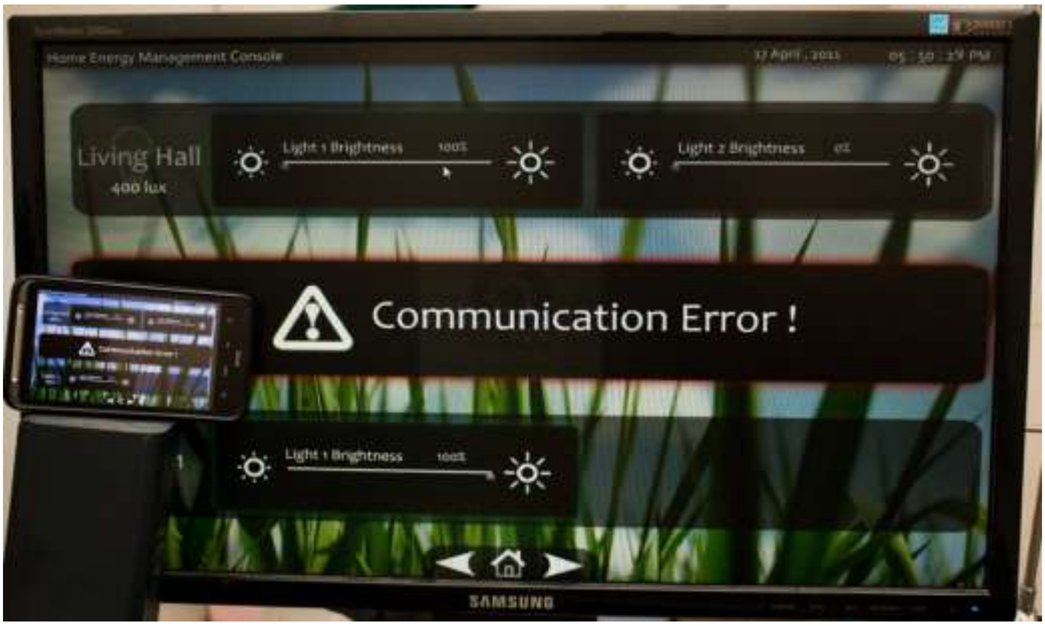
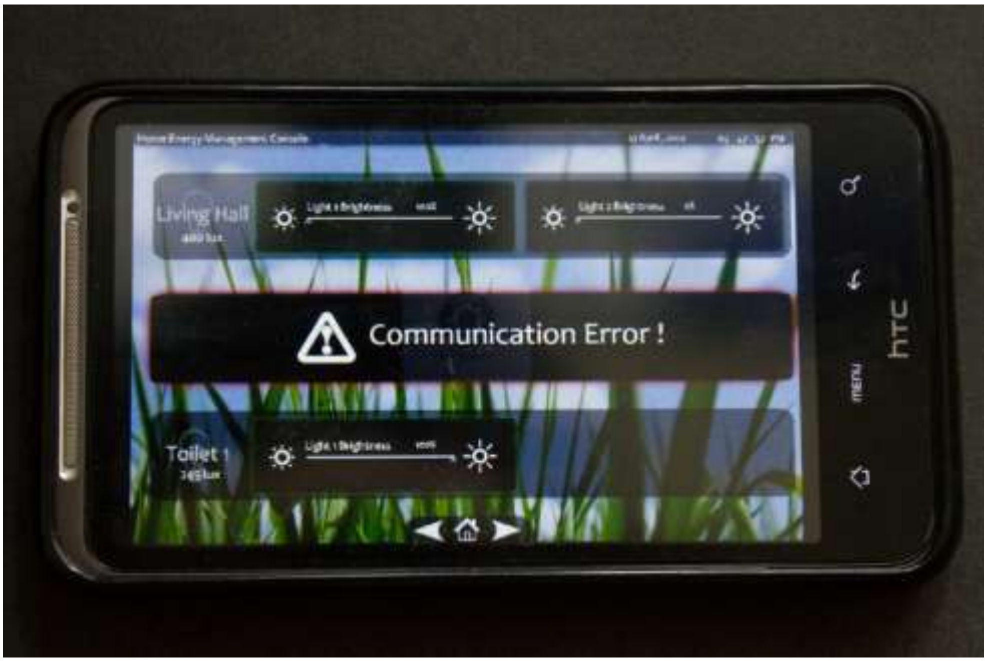


Figure 4.23: Smartphone (left) and IHAS Application(right)



Communication Error !

Figure 4.24: Smartphone Streaming IHAS Application

Figure 4.23 and Figure 4.24 shows how we could extend IHAS control to a smartphone. The screen resolution and refresh rate is not optimized. This problem could be fixed if native IHAS client is developed for the smartphone instead of running VNC through a Wi-Fi network.

4.8 Design Solution based on commercial products

4.8.1 Implementation

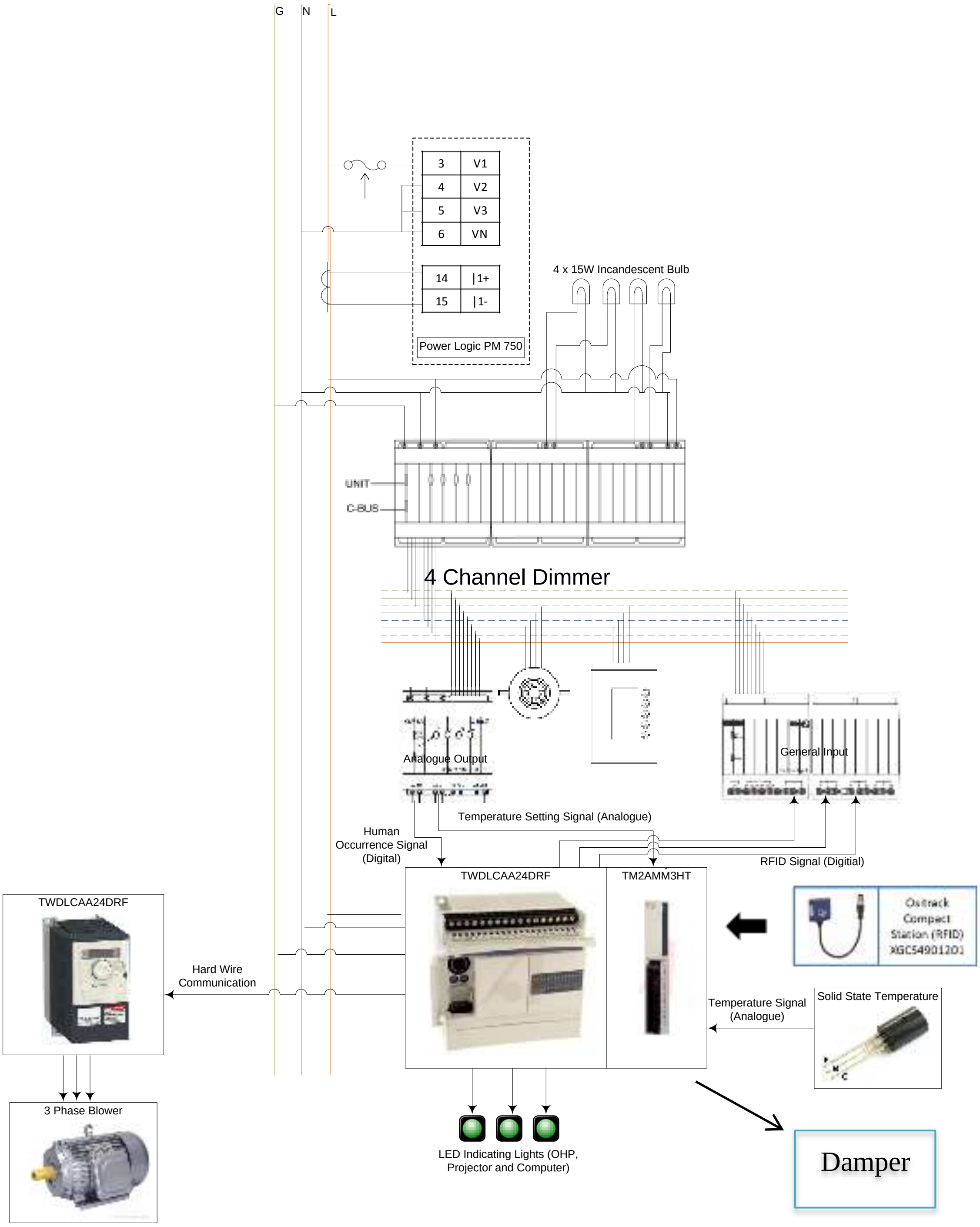

G
N
L
3
V1
4
V2
5
V3
6
VN
14
|1+
15
|1-
Power Logic PM 750
4 x 15W Incandescent Bulb
UNIT
C-BUS
4 Channel Dimmer
Analogue Output
General Input
Temperature Setting Signal (Analogue)
Human Occurrence Signal (Digital)
RFID Signal (Digitial)
TWDLCAA24DRF
TM2AMM3HT
Ositrack Compact Station (RFID) XGCS4901201
TWDLCAA24DRF
Hard Wire Communication
Temperature Signal (Analogue)
Solid State Temperature
3 Phase Blower
LED Indicating Lights (OHP, Projector and Computer)
Damper

Figure 4.25: Electrical Wiring diagram for HVAC and Lighting Control System Prototype

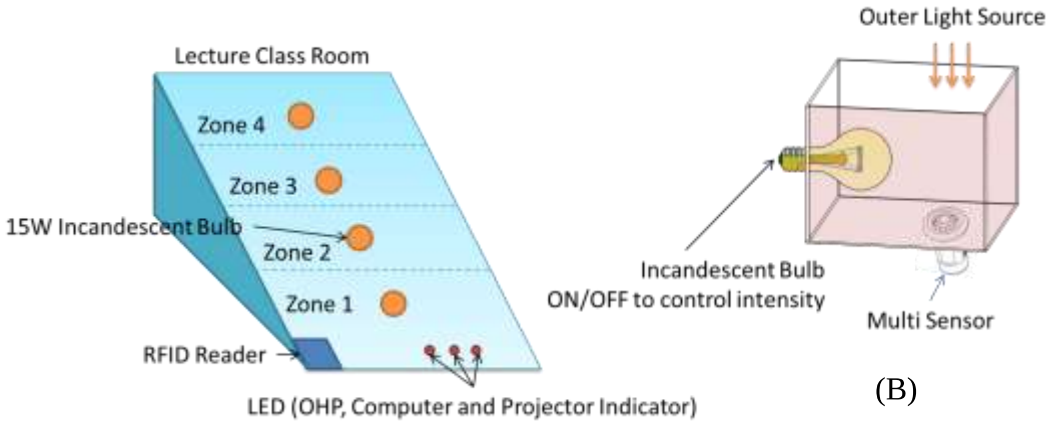
Figure 4.25 shows the Electrical Wiring diagram for HVAC and Lighting Control System Prototype. As this is a prototype of the design solution on integrated control system, there is some modification on the initial design solution to focus the attention on integrating HVAC and Lighting Control. Hence, the central management console, XBEE and PACAL automation controller is not included in this first prototype as it is a later stage integration to the HVAC and Lighting Control System. The bill of material that is shown in Figure 4.25 is listed in Table 4.1.

Table 4.1: Bill of Material for Lighting and HVAC control prototype

Item	**Part**
C-BUS 5G KEY INPUT, SATURN DLT,LCD	5085DL GF
C-BUS GENERAL INPUT	E5504GI
4CHANNEL DIMMER 240V-200mA POWER SUPPLY	L5504D2A
COMBINED PIR & LIGHT LEVEL SENSOR	5753PEIRL
C-BUS ANALOGUE OUTPUT	L5504AMP
ATV312 ALTIVAR	ATV312H037M2
PM750 POWER METER	PM750MG
RJ45 - MINI DIN CABLE	TSXCRJMD25
USB - RS485 CONVERTER	TSXCUSB485
TWIDO ANALOGUE MODULE	TM2AMM3HT
RFID OSISENSE STATION XG	XGCS4901201
RFID CARDS	XGHB90E340
RFID CABLE	TCSMCN1F2
BASE UNIT AC,14 IN DC,10 OUT RLY,TBK	TWDLCAA24DRF
MOD-BUS ADAPTER	TWDNAC485T

The prototype is divided in 2 parts, lighting control and HVAC and RFID control.

4.8.1.1 Lighting Control



(B)

(A)

Figure 4.26(A): Lighting Control Prototype

Figure 4.26(B): Enclosure for Multi Sensor

Figure 4.26 (A) shows the lighting control prototype while Figure 4.26 (B) shows the enclosure for multi sensor. The prototype is divided into two parts as they are two very different systems. The Lighting Control system is based on commercial products produces by Clipsal, whereas the HVAC and RFID control is an industrial solution by Telemacanique. However, we managed to integrate two different solutions working as one system. We shall begin to discuss our system with lighting control system.

The lecture hall is divided into four zones, where zone one is the teaching area. Also, there are three LED that represents OHP, Computer and Projector respectively. There is also and RFID reader that is located at the entrance of the lecture hall, which is shown in the bottom left corner of the lighting control prototype. Each zone is light up by an incandescent bulb. There is also a control box to simulate daylight and shows how the light and occupancy sensor reacts to sudden change in light intensity.

Clipsal four channel dimmer L5504D2A, Clipsal Analogue Output L5504AMP, Clipsal C-Bus 5G Key Input 5085DLGF, Clipsal C-Bus general Input and Clipsal C-Bus Occupancy and Light level Multi Sensor 5753PEIRL is connected to C-Bus network. By connecting C-Bus network, the connected devices could communicate with each other.

The four channel light dimmer activates the lights according to preset luminance based on ambient light level when the multi sensor detects human occupancy. Simultaneously, the analogue output will sends signal to the HVAC control which will turn on the air conditioner system for the particular lecture hall. C-Bus general input unit is used to accept RFID signals from the PLC as there is no RFID reader available from Clipsal.

RFID settings not only control the light's intensity, it also control the devices that is able switched on based on lecturer teaching style. In this prototype, three types of teaching style are created, OHP, Projector and All on. Therefore, based on the RFID detected upon lecturer entering the lecture hall, changes will be made automatically to suit that particular lecturer.

Table 4.2: Preset RFID Configuration

Preset modes	Lights configuration	Projector and screen	OHP
OHP mode	Front row off, 2nd row 50 % intensity, Others on	Off	On
Projector mode	Front row 50 %, 2nd row 70 % intensity, Others on	On	Off
All on	All lights full intensity	Off	Off

Table 4.2 shows preset RFID Configurations. In OHP modes, the front rows of lights will be switched *off* to increase the clarity of the OHP projector. At the same time, the projector is turned *off* to save power. The lecture PC is configured to sleep, minimizing the power consumed when it is not in used. For lecturer who uses projector, the PLC will control the motorized projector screen and turn *on* the

projector while keeping the OHP switched *off* to save power. For lecturer who teaches using white board, both projector and OHP is switched *off* but lights will be *on* at full intensity to provide a good teaching environment. Figure 4.27 shows the outcome of the different preset configurations where OHP mode is on the left, Projector mode on upper right and All on mode on lower right. Besides preconfigured light and device settings, lecturers could also override the lights setting from the input panel. Lecturer could also change the air conditioner setting from the input panel.

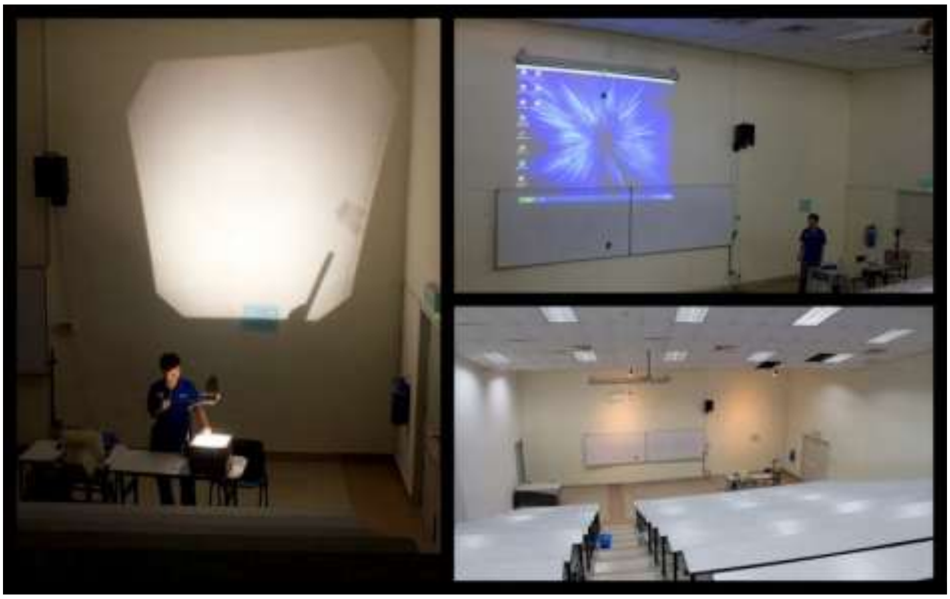


Figure 4.27: Different Preset Modes, from left clockwise (OHP mode, Projector mode and All On)

4.8.1.2 HVAC and RFID Control

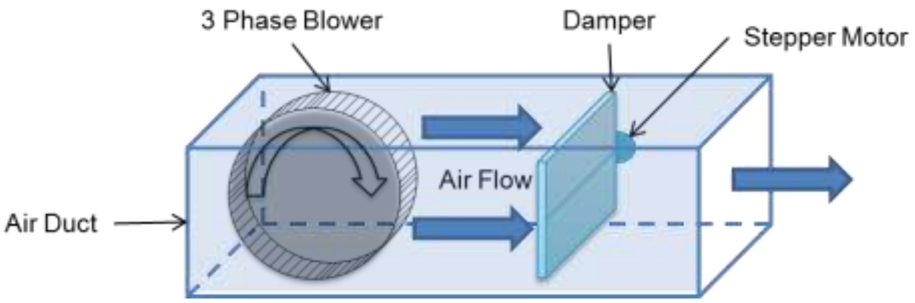


Figure 4.28: HVAC Prototype

The HVAC prototype consists of 3 sections, the air duct, blower, and a damper as shown in Figure 4.28. Air duct is the air channel that the cold air passes from the Air Handling Unit (AHU) to every lecture hall, and each individual lecture hall controls the air flow via a damper. Figure 4.28 shows the HVAC prototype that is use to show the blower in AHU and damper control in the HVAC system. The blower speed and the damper opening will be determined by the PLC which gets its signal from C-Bus network.

The respective hardware needed in HVAC control is Telemacanique ATV312 Altivar ATV312H037M2 variable speed drive 3 phase motor driver, Twido Base unit TWDLCAA24DRF which is a PLC, Twido Analogue module TM2AMM3HT to expand the PLC unit to accept analogue input.

From Figure 4.25, PLC receives signals from C-Bus analogue output unit on the following occasions:

- C-Bus Multi sensor detects human occupancy, C-Bus analogue sends signal to turn on air conditioner.
- C-Bus input panel changes temperature settings, C-Bus analogue sends signal for air flow regulation.
- C-Bus Multi sensor detects human occupancy, C-Bus analogue sends signal to turn off air conditioner after certain period.

Also, RFID receiver is also connected to the PLC in this prototype. When the RFID is scanned and detected, the PLC will send signal to the C-Bus General Input to switch between various modes.

To keep track of the energy savings, continuous monitoring is essential. Therefore, PM750MG Power Meter is installed in the mains to measure and track energy consumption of our prototype.

4.8.2 Outcome

Through various experiment and research, it is found that the HVAC and Lighting Control System is able to save 44 % in lighting control and 60 % in HVAC control. This translates into massive savings if it is implemented UTAR FES campus. By implementing the proposed HVAC and Lighting Control System, UTAR FES will achieve return on investment in 3 years with a total savings up to RM250,000 a year. Detailed calculations on the acclaimed savings are available in Appendix B.

The massive savings, simple solution of the HVAC and Lighting Control System is awarded 1st Runner Up in Schneider Electric Green the World @ My Campus University Challenge 2010 is as shown in Figure 4.29.

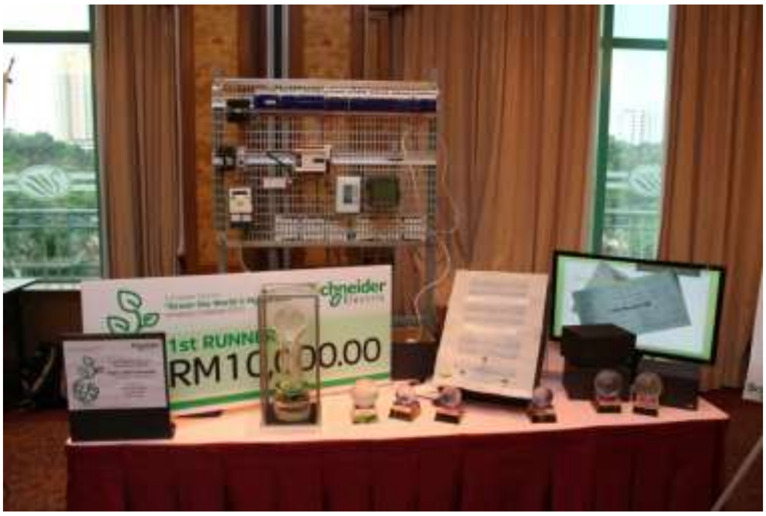


Figure 4.29: The Lighting and HVAC Control System Prototype

Figure 4.30 shows the Lighting and HVAC Control System with updated prototype, showcasing in IEEE STUDENT Conference 2010. The HVAC and Lighting Control System is awarded Best Exhibitor award.

Figure 4.30: IEEE STUDENT Conference 2010 Best Exhibitor Award Winner, From left, MAK KWAN WUEY, LIU CHEE WEI, GAN YU HAN.

CHAPTER 5

CONCLUSION AND FUTURE IMPLEMENTATION

As a conclusion, Integrated Home Automation System allows users to interact with physical devices from their computer. Also, IHAS has established a common method of communication for device manufacturer. IHAS is able to automate coordinator setup, which is a plug-and-play feature to the end users. This simplifies the home automation process as IHAS does not require users to manually configure the XBEE.

Also, IHAS has the option to run on desktop mode, which integrates IHAS to the desktop of a Windows machine, enabling users to interact with IHAS easily and seamlessly while using other applications. Both full application mode and desktop mode features an attractive and intuitive design that aims to being home automation technology easy to operate by anyone.

IHAS operates at AES128 encryption mode, ensuring the security of the users' home automation system. Also, users could also use RFID to control their devices using IHAS. Furthermore, users are also given the option to remote control IHAS via Wi-Fi network using a smartphone with VNC client installed.

Integration of IHAS with commercial system is planned for future development, as well as developing a dedicated smartphone client for IHAS that offers better user experience. Overall stability of IHAS also needs to be enhanced. Also, IHAS could also provides time management in future releases. Lastly, more devices support and updated and more streamlined user interface is also planned for IHAS.

REFERENCES

Adobe. (2011). Retrieved August 27, 2010, from Adobe Flash Professional CS5: http://www.adobe.com/products/flash/

Adobe Flash. (2011). Retrieved August 27, 2010, from Flash Professional for web application development: http://www.adobe.com/products/flash/web/

AGGsoftware. (1999). *AGGsoftware, TECHNOLOGY AT YOUR SERVICE*. Retrieved August 27, 2010, from COM PORT DATA EMULATOR: http://www.aggsoft.com/com-port-emulator.htm

Alliance, Z.-W. (2004). *Z-Wave Alliance*. Retrieved August 27, 2010, from Z-Wave Alliance: http://www.z-wavealliance.org/modules/AboutUs/

Cotter, K. (2008, November). *ONE-NET*. Retrieved from ONE-NET: http://www.one-net.info/index.php?option=com_content&view=article&id=25:nodes&catid=16:tidbits&Itemid=20

Cotter, K. (2008, November). *ONE-NET FAQ*. Retrieved August 27, 2010, from ONE-NET: http://www.one-net.info/index.php?option=com_content&view=article&id=24:power&catid=16:tidbits&Itemid=20

Cotter, K. (2008, November). *ONE-NET Topology*. Retrieved August 27, 2010, from ONE-NET: http://www.one-net.info/index.php?option=com_content&view=article&id=27:topology&catid=16:tidbits&Itemid=20

ETERLOGIC. (2007). *ETERLOGIC.COM*. Retrieved August 27, 2010, from Virtual Serial Port Emulator: http://www.eterlogic.com/Products.VSPE.html

Harrop, P. (2010). ZigBee for Energy Harvesting and Very Low Power. *IDTechEx Energy Harvesting and Storage conference,* (pp. 1-2). France.

Hernandez, J. C., & lsasi, P. (2004, June). New results on the genetic cryptanalysis of TEA and reduced-round versions of XTEA. *2004 Congress on Evolutionary Computation CEC2004, 2,* 2128.

Kanellos, M. (2010, July 3). *Intel's Cure for Home Energy Management: Answering Machines*. Retrieved August 25, 2010, from WIRED: http://www.wired.com/epicenter/2010/07/intel%E2%80%99s-cure-for-home-energy-management-answering-machines/?intcid=postnav

Kinney, P. (2003). Zigbee Technology: Wireless Control that Simply Works. *Communication Design Conference,* (pp. 2-4).

(n.d.). In Lucy A. D. Lockwood, & Larry L.Constantine, *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design* (pp. 45-46). US: Addison-Longman, Inc.

Meshnetics Team, A. C. (2009). *Atmel Corporation*. Retrieved from ZIGBEE FAQ: http://www.meshnetics.com/zigbee-faq/#14

MeshNetics Team, A. C. (2009). *MeshNetics*. Retrieved April 10, 2011, from ZIGBEE FAQ: http://www.meshnetics.com/zigbee-faq/#33

Multidmedia Ltd. (2002). *Multidmedia Ltd*. Retrieved August 26, 2010, from Zinc™ 3.0 - Rapid Application Development for Adobe® Flash: http://www.multidmedia.com/software/zinc/

ONE-NET. (n.d.). Retrieved from About ONE-NET: http://www.one-net.info/index.php?option=com_content&view=section&layout=blog&id=1&Itemid=2

RFID Reader RS232. (2010). Retrieved April 10, 2011, from Cytron Technologies, Robot Head To Toe: http://www.cytron.com.my/viewProduct.php?pid=EyQkLxQPGT0JJicFBzk4E5Kg2NOpoGCSzAPYwnQ/jLQ=

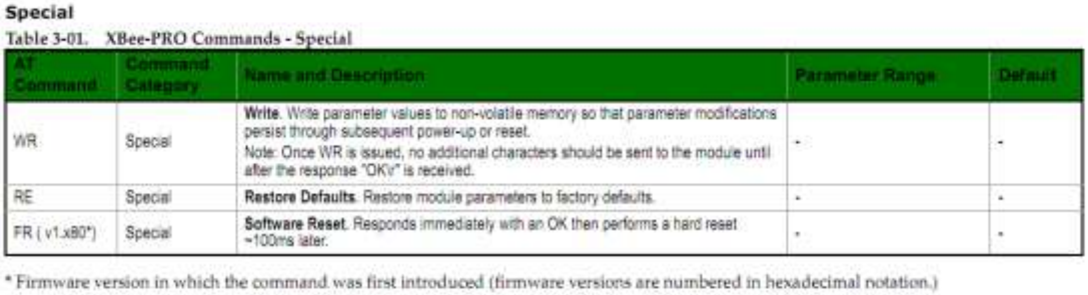
Scott W. Ambler. (1998). *Best Practices for Software Development*. Retrieved August 27, 2010, from User Interface Design Tips, Techniques, and Principles: http://www.ambysoft.com/essays/userInterfaceDesign.html

Technology, S. (2005, August). Retrieved from Insteon, a smartlabs technology: http://www.insteon.net/about-whitepapers.html

Thonet, G., Allard-Jacquin, P., & Colle, P. (2008). *ZigBee – WiFi Coexistence.* White paper and Test Report, Innovation Department, France.

Wei, L. C. (2010). *Smart Lighting System: Modular Intelligent Control System.*

Appendix A XBEE Command List

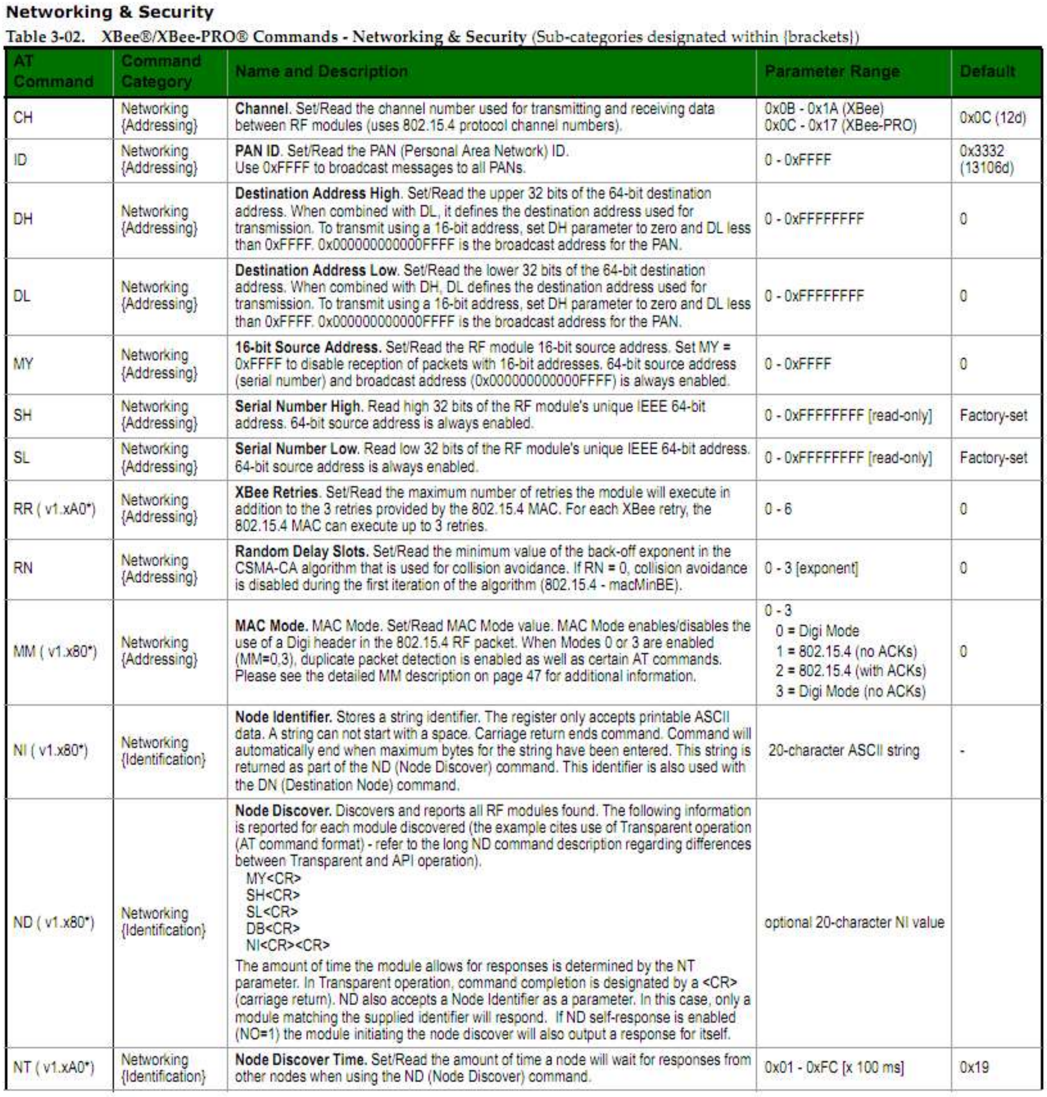


Special

Table 3-01. XBee-PRO Commands - Special

AT Command	Command Category	Name and Description	Parameter Range	Default
WR	Special	**Write**. Write parameter values to non-volatile memory so that parameter modifications persist through subsequent power-up or reset. Note: Once WR is issued, no additional characters should be sent to the module until after the response "OK\r" is received.	-	-
RE	Special	**Restore Defaults**. Restore module parameters to factory defaults.	-	-
FR (v1.x80*)	Special	**Software Reset**. Responds immediately with an OK then performs a hard reset ~100ms later.	-	-

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

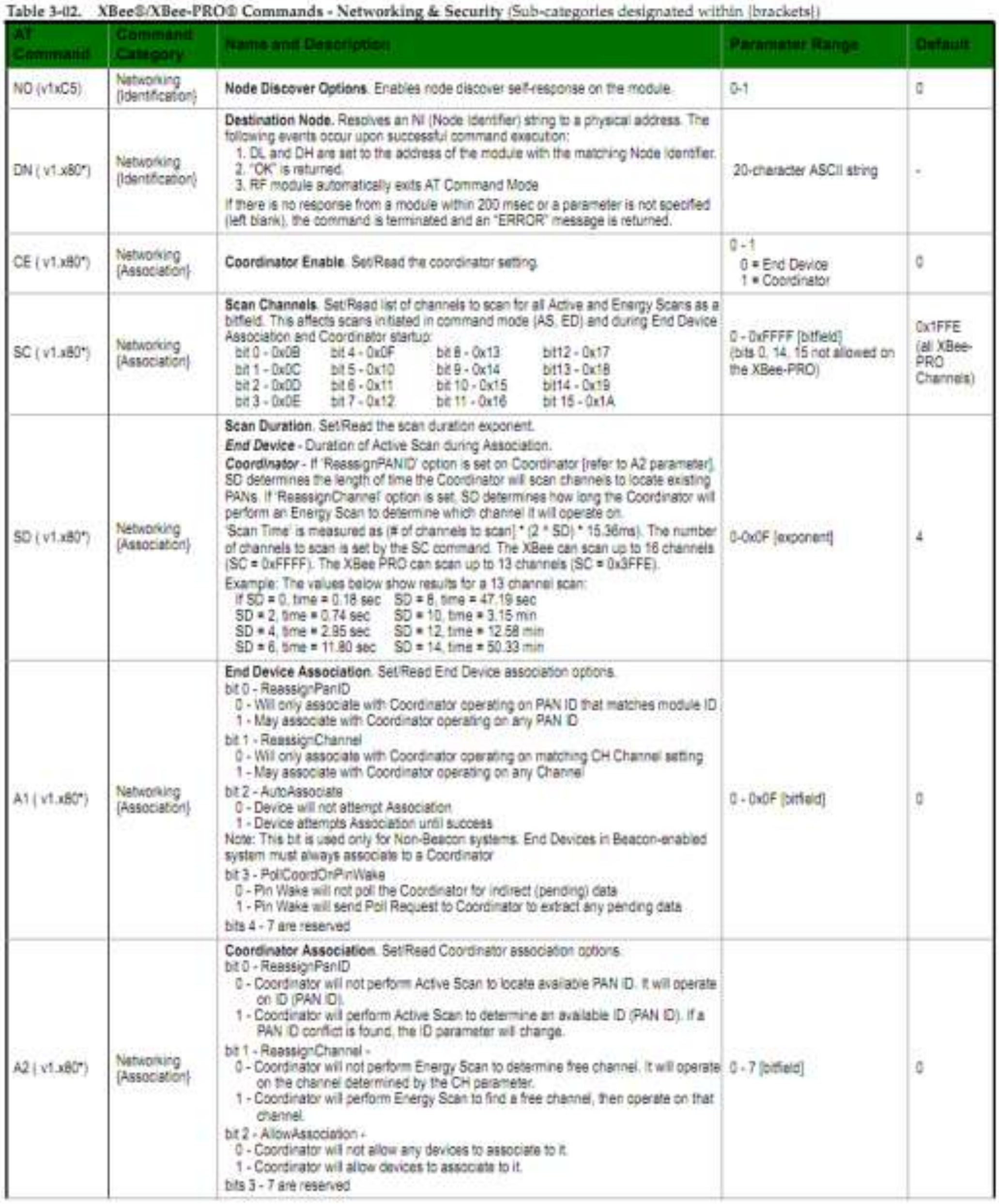


Networking & Security

Table 3-02. XBee®/XBee-PRO® Commands - Networking & Security (Sub-categories designated within {brackets})

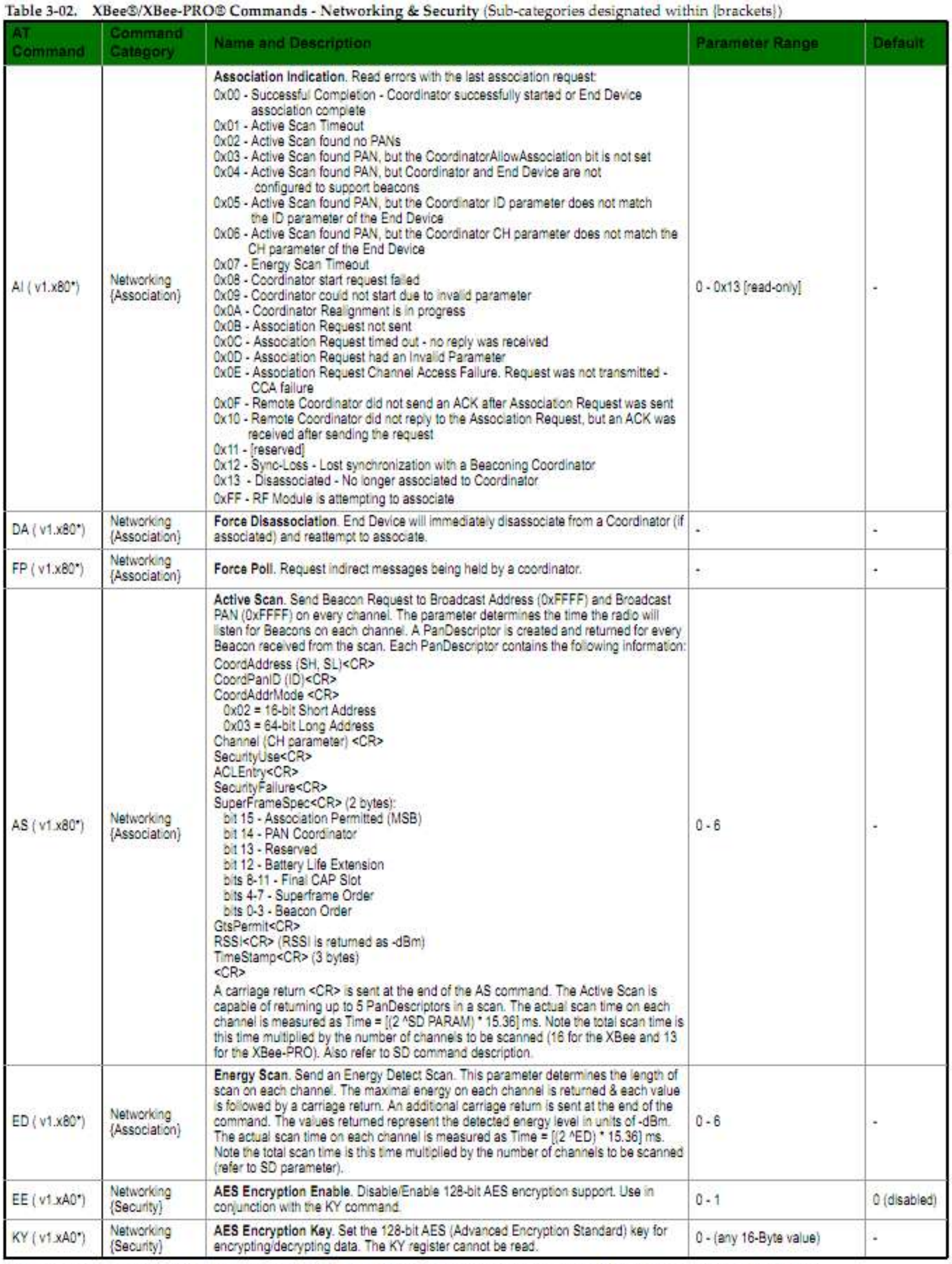
AT Command	Command Category	Name and Description	Parameter Range	Default
CH	Networking {Addressing}	**Channel**. Set/Read the channel number used for transmitting and receiving data between RF modules (uses 802.15.4 protocol channel numbers).	0x0B - 0x1A (XBee) 0x0C - 0x17 (XBee-PRO)	0x0C (12d)
ID	Networking {Addressing}	**PAN ID**. Set/Read the PAN (Personal Area Network) ID. Use 0xFFFF to broadcast messages to all PANs.	0 - 0xFFFF	0x3332 (13106d)
DH	Networking {Addressing}	**Destination Address High**. Set/Read the upper 32 bits of the 64-bit destination address. When combined with DL, it defines the destination address used for transmission. To transmit using a 16-bit address, set DH parameter to zero and DL less than 0xFFFF. 0x000000000000FFFF is the broadcast address for the PAN.	0 - 0xFFFFFFFF	0
DL	Networking {Addressing}	**Destination Address Low**. Set/Read the lower 32 bits of the 64-bit destination address. When combined with DH, DL defines the destination address used for transmission. To transmit using a 16-bit address, set DH parameter to zero and DL less than 0xFFFF. 0x000000000000FFFF is the broadcast address for the PAN.	0 - 0xFFFFFFFF	0
MY	Networking {Addressing}	**16-bit Source Address**. Set/Read the RF module 16-bit source address. Set MY = 0xFFFF to disable reception of packets with 16-bit addresses. 64-bit source address (serial number) and broadcast address (0x000000000000FFFF) is always enabled.	0 - 0xFFFF	0
SH	Networking {Addressing}	**Serial Number High**. Read high 32 bits of the RF module's unique IEEE 64-bit address. 64-bit source address is always enabled.	0 - 0xFFFFFFFF [read-only]	Factory-set
SL	Networking {Addressing}	**Serial Number Low**. Read low 32 bits of the RF module's unique IEEE 64-bit address. 64-bit source address is always enabled.	0 - 0xFFFFFFFF [read-only]	Factory-set
RR (v1.xA0*)	Networking {Addressing}	**XBee Retries**. Set/Read the maximum number of retries the module will execute in addition to the 3 retries provided by the 802.15.4 MAC. For each XBee retry, the 802.15.4 MAC can execute up to 3 retries.	0 - 6	0
RN	Networking {Addressing}	**Random Delay Slots.** Set/Read the minimum value of the back-off exponent in the CSMA-CA algorithm that is used for collision avoidance. If RN = 0, collision avoidance is disabled during the first iteration of the algorithm (802.15.4 - macMinBE).	0 - 3 [exponent]	0
MM (v1.x80*)	Networking {Addressing}	**MAC Mode.** MAC Mode. Set/Read MAC Mode value. MAC Mode enables/disables the use of a Digi header in the 802.15.4 RF packet. When Modes 0 or 3 are enabled (MM=0,3), duplicate packet detection is enabled as well as certain AT commands. Please see the detailed MM description on page 47 for additional information.	0 - 3 0 = Digi Mode 1 = 802.15.4 (no ACKs) 2 = 802.15.4 (with ACKs) 3 = Digi Mode (no ACKs)	0
NI (v1.x80*)	Networking {Identification}	**Node Identifier.** Stores a string identifier. The register only accepts printable ASCII data. A string can not start with a space. Carriage return ends command. Command will automatically end when maximum bytes for the string have been entered. This string is returned as part of the ND (Node Discover) command. This identifier is also used with the DN (Destination Node) command.	20-character ASCII string	-
ND (v1.x80*)	Networking {Identification}	**Node Discover.** Discovers and reports all RF modules found. The following information is reported for each module discovered (the example cites use of Transparent operation (AT command format) - refer to the long ND command description regarding differences between Transparent and API operation). MY<CR> SH<CR> SL<CR> DB<CR> NI<CR><CR> The amount of time the module allows for responses is determined by the NT parameter. In Transparent operation, command completion is designated by a <CR> (carriage return). ND also accepts a Node Identifier as a parameter. In this case, only a module matching the supplied identifier will respond. If ND self-response is enabled (NO=1) the module initiating the node discover will also output a response for itself.	optional 20-character NI value	
NT (v1.xA0*)	Networking {Identification}	**Node Discover Time.** Set/Read the amount of time a node will wait for responses from other nodes when using the ND (Node Discover) command.	0x01 - 0xFC [x 100 ms]	0x19

Table 3-02. XBee®/XBee-PRO® Commands - Networking & Security (Sub-categories designated within {brackets})



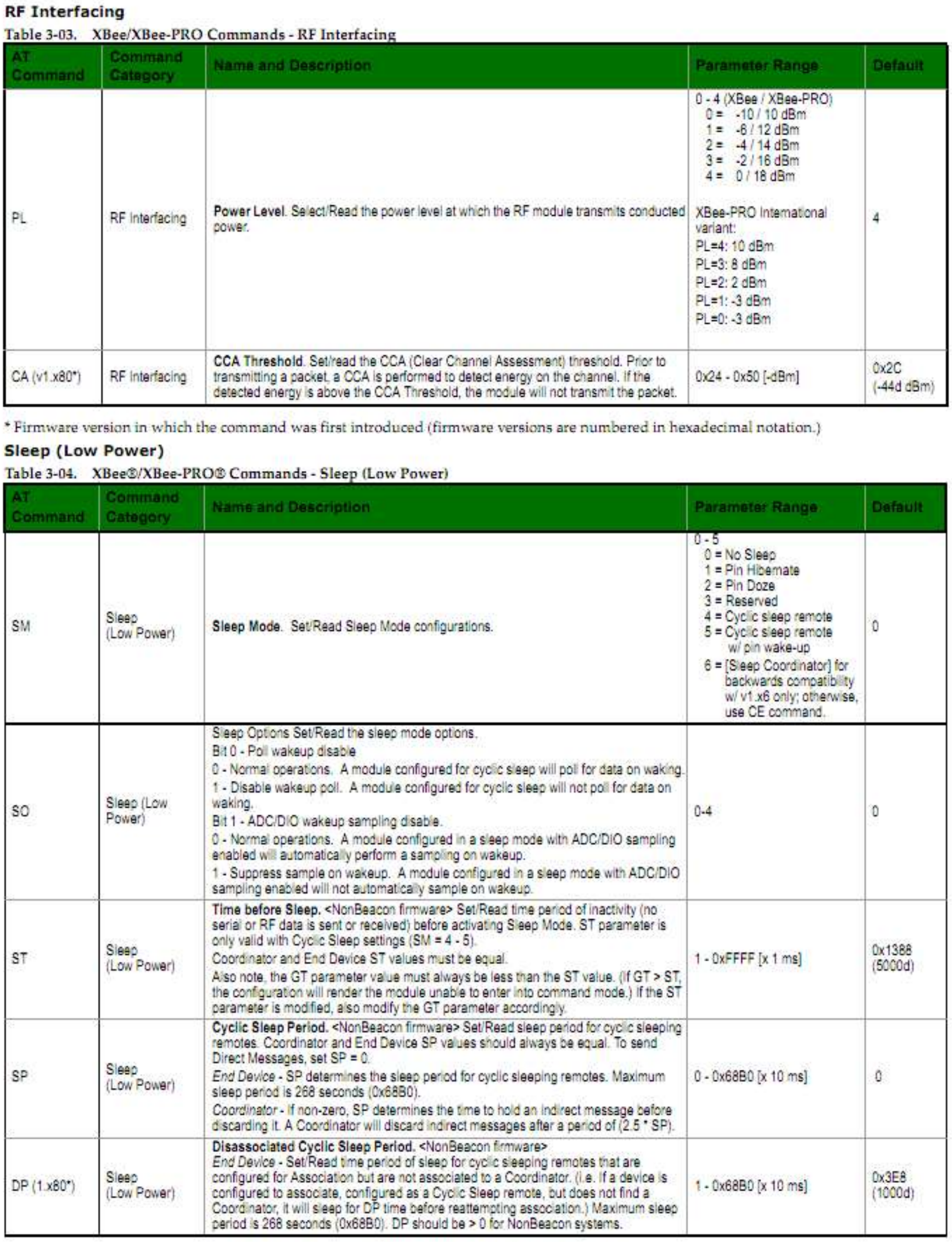
AT Command	Command Category	Name and Description	Parameter Range	Default
NO (v1xC5)	Networking {Identification}	**Node Discover Options**. Enables node discover self-response on the module.	0-1	0
DN (v1.x80*)	Networking {Identification}	**Destination Node.** Resolves an NI (Node Identifier) string to a physical address. The following events occur upon successful command execution: 1. DL and DH are set to the address of the module with the matching Node Identifier. 2. "OK" is returned. 3. RF module automatically exits AT Command Mode If there is no response from a module within 200 msec or a parameter is not specified (left blank), the command is terminated and an "ERROR" message is returned.	20-character ASCII string	-
CE (v1.x80*)	Networking {Association}	**Coordinator Enable**. Set/Read the coordinator setting.	0 - 1 0 = End Device 1 = Coordinator	0
SC (v1.x80*)	Networking {Association}	**Scan Channels**. Set/Read list of channels to scan for all Active and Energy Scans as a bitfield. This affects scans initiated in command mode (AS, ED) and during End Device Association and Coordinator startup: bit 0 - 0x0B bit 4 - 0x0F bit 8 - 0x13 bit12 - 0x17 bit 1 - 0x0C bit 5 - 0x10 bit 9 - 0x14 bit13 - 0x18 bit 2 - 0x0D bit 6 - 0x11 bit 10 - 0x15 bit14 - 0x19 bit 3 - 0x0E bit 7 - 0x12 bit 11 - 0x16 bit 15 - 0x1A	0 - 0xFFFF [bitfield] (bits 0, 14, 15 not allowed on the XBee-PRO)	0x1FFE (all XBee-PRO Channels)
SD (v1.x80*)	Networking {Association}	**Scan Duration**. Set/Read the scan duration exponent. ***End Device*** - Duration of Active Scan during Association. **Coordinator** - If 'ReassignPANID' option is set on Coordinator [refer to A2 parameter], SD determines the length of time the Coordinator will scan channels to locate existing PANs. If 'ReassignChannel' option is set, SD determines how long the Coordinator will perform an Energy Scan to determine which channel it will operate on. 'Scan Time' is measured as (# of channels to scan] * (2 ^ SD) * 15.36ms). The number of channels to scan is set by the SC command. The XBee can scan up to 16 channels (SC = 0xFFFF). The XBee PRO can scan up to 13 channels (SC = 0x3FFE). Example: The values below show results for a 13 channel scan: If SD = 0, time = 0.18 sec SD = 8, time = 47.19 sec SD = 2, time = 0.74 sec SD = 10, time = 3.15 min SD = 4, time = 2.95 sec SD = 12, time = 12.58 min SD = 6, time = 11.80 sec SD = 14, time = 50.33 min	0-0x0F [exponent]	4
A1 (v1.x80*)	Networking {Association}	**End Device Association**. Set/Read End Device association options. bit 0 - ReassignPanID 0 - Will only associate with Coordinator operating on PAN ID that matches module ID 1 - May associate with Coordinator operating on any PAN ID bit 1 - ReassignChannel 0 - Will only associate with Coordinator operating on matching CH Channel setting 1 - May associate with Coordinator operating on any Channel bit 2 - AutoAssociate 0 - Device will not attempt Association 1 - Device attempts Association until success Note: This bit is used only for Non-Beacon systems. End Devices in Beacon-enabled system must always associate to a Coordinator bit 3 - PollCoordOnPinWake 0 - Pin Wake will not poll the Coordinator for indirect (pending) data 1 - Pin Wake will send Poll Request to Coordinator to extract any pending data bits 4 - 7 are reserved	0 - 0x0F [bitfield]	0
A2 (v1.x80*)	Networking {Association}	**Coordinator Association**. Set/Read Coordinator association options. bit 0 - ReassignPanID 0 - Coordinator will not perform Active Scan to locate available PAN ID. It will operate on ID (PAN ID). 1 - Coordinator will perform Active Scan to determine an available ID (PAN ID). If a PAN ID conflict is found, the ID parameter will change. bit 1 - ReassignChannel - 0 - Coordinator will not perform Energy Scan to determine free channel. It will operate on the channel determined by the CH parameter. 1 - Coordinator will perform Energy Scan to find a free channel, then operate on that channel. bit 2 - AllowAssociation - 0 - Coordinator will not allow any devices to associate to it. 1 - Coordinator will allow devices to associate to it. bits 3 - 7 are reserved	0 - 7 [bitfield]	0

Table 3-02. XBee®/XBee-PRO® Commands - Networking & Security (Sub-categories designated within {brackets})



AT Command	Command Category	Name and Description	Parameter Range	Default
AI (v1.x80*)	Networking {Association}	**Association Indication**. Read errors with the last association request: 0x00 - Successful Completion - Coordinator successfully started or End Device association complete 0x01 - Active Scan Timeout 0x02 - Active Scan found no PANs 0x03 - Active Scan found PAN, but the CoordinatorAllowAssociation bit is not set 0x04 - Active Scan found PAN, but Coordinator and End Device are not configured to support beacons 0x05 - Active Scan found PAN, but the Coordinator ID parameter does not match the ID parameter of the End Device 0x06 - Active Scan found PAN, but the Coordinator CH parameter does not match the CH parameter of the End Device 0x07 - Energy Scan Timeout 0x08 - Coordinator start request failed 0x09 - Coordinator could not start due to invalid parameter 0x0A - Coordinator Realignment is in progress 0x0B - Association Request not sent 0x0C - Association Request timed out - no reply was received 0x0D - Association Request had an Invalid Parameter 0x0E - Association Request Channel Access Failure. Request was not transmitted - CCA failure 0x0F - Remote Coordinator did not send an ACK after Association Request was sent 0x10 - Remote Coordinator did not reply to the Association Request, but an ACK was received after sending the request 0x11 - [reserved] 0x12 - Sync-Loss - Lost synchronization with a Beaconing Coordinator 0x13 - Disassociated - No longer associated to Coordinator 0xFF - RF Module is attempting to associate	0 - 0x13 [read-only]	-
DA (v1.x80*)	Networking {Association}	**Force Disassociation**. End Device will immediately disassociate from a Coordinator (if associated) and reattempt to associate.	-	-
FP (v1.x80*)	Networking {Association}	**Force Poll**. Request indirect messages being held by a coordinator.	-	-
AS (v1.x80*)	Networking {Association}	**Active Scan**. Send Beacon Request to Broadcast Address (0xFFFF) and Broadcast PAN (0xFFFF) on every channel. The parameter determines the time the radio will listen for Beacons on each channel. A PanDescriptor is created and returned for every Beacon received from the scan. Each PanDescriptor contains the following information: CoordAddress (SH, SL)<CR> CoordPanID (ID)<CR> CoordAddrMode <CR> 0x02 = 16-bit Short Address 0x03 = 64-bit Long Address Channel (CH parameter) <CR> SecurityUse<CR> ACLEntry<CR> SecurityFailure<CR> SuperFrameSpec<CR> (2 bytes): bit 15 - Association Permitted (MSB) bit 14 - PAN Coordinator bit 13 - Reserved bit 12 - Battery Life Extension bits 8-11 - Final CAP Slot bits 4-7 - Superframe Order bits 0-3 - Beacon Order GtsPermit<CR> RSSI<CR> (RSSI is returned as -dBm) TimeStamp<CR> (3 bytes) <CR> A carriage return <CR> is sent at the end of the AS command. The Active Scan is capable of returning up to 5 PanDescriptors in a scan. The actual scan time on each channel is measured as Time = [(2 ^SD PARAM) * 15.36] ms. Note the total scan time is this time multiplied by the number of channels to be scanned (16 for the XBee and 13 for the XBee-PRO). Also refer to SD command description.	0 - 6	-
ED (v1.x80*)	Networking {Association}	**Energy Scan**. Send an Energy Detect Scan. This parameter determines the length of scan on each channel. The maximal energy on each channel is returned & each value is followed by a carriage return. An additional carriage return is sent at the end of the command. The values returned represent the detected energy level in units of -dBm. The actual scan time on each channel is measured as Time = [(2 ^ED) * 15.36] ms. Note the total scan time is this time multiplied by the number of channels to be scanned (refer to SD parameter).	0 - 6	-
EE (v1.xA0*)	Networking {Security}	**AES Encryption Enable**. Disable/Enable 128-bit AES encryption support. Use in conjunction with the KY command.	0 - 1	0 (disabled)
KY (v1.xA0*)	Networking {Security}	**AES Encryption Key**. Set the 128-bit AES (Advanced Encryption Standard) key for encrypting/decrypting data. The KY register cannot be read.	0 - (any 16-Byte value)	-

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)



RF Interfacing

Table 3-03. XBee/XBee-PRO Commands - RF Interfacing

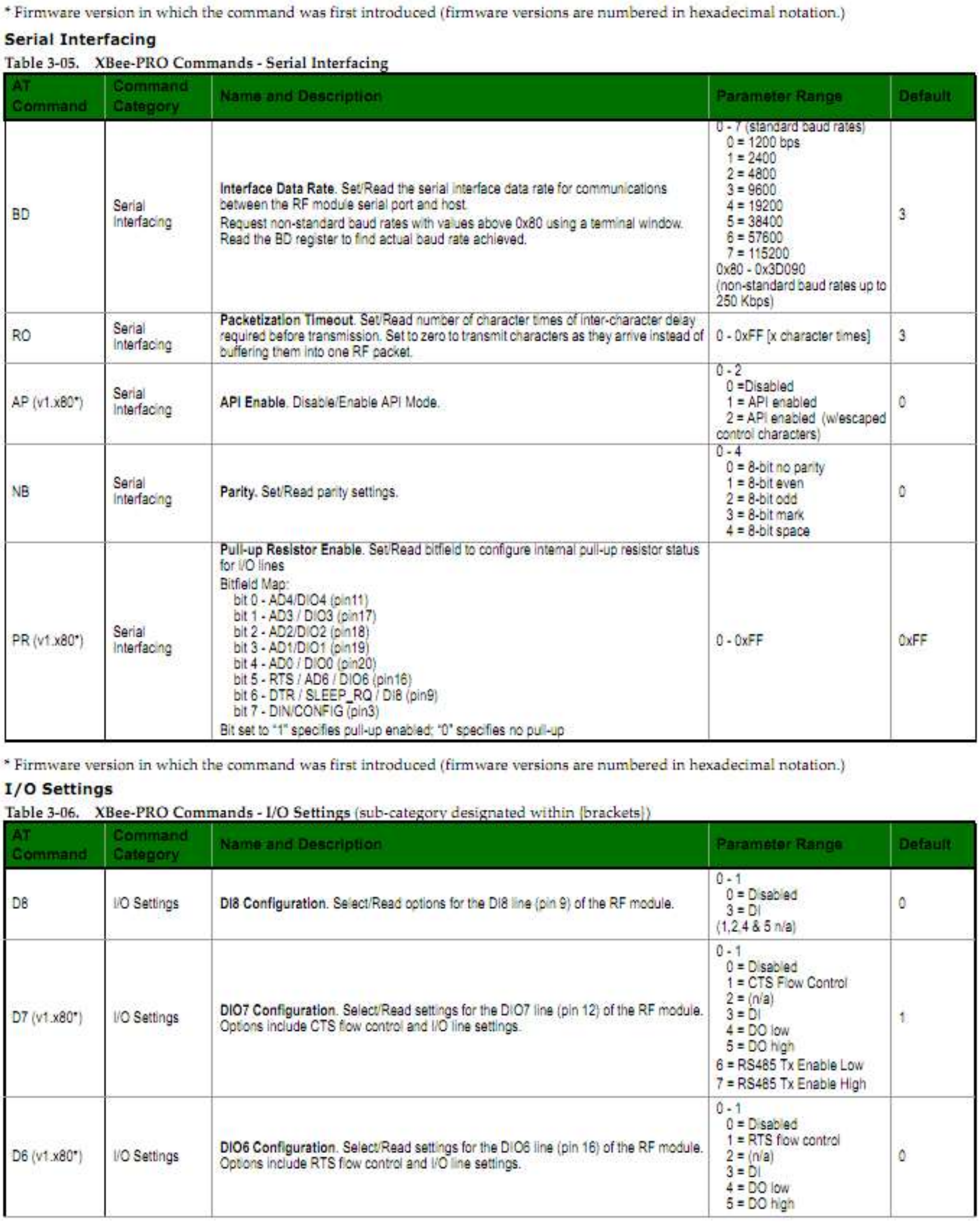
AT Command	Command Category	Name and Description	Parameter Range	Default
PL	RF Interfacing	**Power Level**. Select/Read the power level at which the RF module transmits conducted power.	0 - 4 (XBee / XBee-PRO) 0 = -10 / 10 dBm 1 = -6 / 12 dBm 2 = -4 / 14 dBm 3 = -2 / 16 dBm 4 = 0 / 18 dBm XBee-PRO International variant: PL=4: 10 dBm PL=3: 8 dBm PL=2: 2 dBm PL=1: -3 dBm PL=0: -3 dBm	4
CA (v1.x80*)	RF Interfacing	**CCA Threshold**. Set/read the CCA (Clear Channel Assessment) threshold. Prior to transmitting a packet, a CCA is performed to detect energy on the channel. If the detected energy is above the CCA Threshold, the module will not transmit the packet.	0x24 - 0x50 [-dBm]	0x2C (-44d dBm)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Sleep (Low Power)

Table 3-04. XBee®/XBee-PRO® Commands - Sleep (Low Power)

AT Command	Command Category	Name and Description	Parameter Range	Default
SM	Sleep (Low Power)	**Sleep Mode**. Set/Read Sleep Mode configurations.	0 - 5 0 = No Sleep 1 = Pin Hibernate 2 = Pin Doze 3 = Reserved 4 = Cyclic sleep remote 5 = Cyclic sleep remote w/ pin wake-up 6 = [Sleep Coordinator] for backwards compatibility w/ v1.x6 only; otherwise, use CE command.	0
SO	Sleep (Low Power)	Sleep Options Set/Read the sleep mode options. Bit 0 - Poll wakeup disable 0 - Normal operations. A module configured for cyclic sleep will poll for data on waking. 1 - Disable wakeup poll. A module configured for cyclic sleep will not poll for data on waking. Bit 1 - ADC/DIO wakeup sampling disable. 0 - Normal operations. A module configured in a sleep mode with ADC/DIO sampling enabled will automatically perform a sampling on wakeup. 1 - Suppress sample on wakeup. A module configured in a sleep mode with ADC/DIO sampling enabled will not automatically sample on wakeup.	0-4	0
ST	Sleep (Low Power)	**Time before Sleep.** <NonBeacon firmware> Set/Read time period of inactivity (no serial or RF data is sent or received) before activating Sleep Mode. ST parameter is only valid with Cyclic Sleep settings (SM = 4 - 5). Coordinator and End Device ST values must be equal. Also note, the GT parameter value must always be less than the ST value. (If GT > ST, the configuration will render the module unable to enter into command mode.) If the ST parameter is modified, also modify the GT parameter accordingly.	1 - 0xFFFF [x 1 ms]	0x1388 (5000d)
SP	Sleep (Low Power)	**Cyclic Sleep Period.** <NonBeacon firmware> Set/Read sleep period for cyclic sleeping remotes. Coordinator and End Device SP values should always be equal. To send Direct Messages, set SP = 0. *End Device* - SP determines the sleep period for cyclic sleeping remotes. Maximum sleep period is 268 seconds (0x68B0). *Coordinator* - If non-zero, SP determines the time to hold an indirect message before discarding it. A Coordinator will discard indirect messages after a period of (2.5 * SP).	0 - 0x68B0 [x 10 ms]	0
DP (1.x80*)	Sleep (Low Power)	**Disassociated Cyclic Sleep Period.** <NonBeacon firmware> *End Device* - Set/Read time period of sleep for cyclic sleeping remotes that are configured for Association but are not associated to a Coordinator. (i.e. If a device is configured to associate, configured as a Cyclic Sleep remote, but does not find a Coordinator, it will sleep for DP time before reattempting association.) Maximum sleep period is 268 seconds (0x68B0). DP should be > 0 for NonBeacon systems.	1 - 0x68B0 [x 10 ms]	0x3E8 (1000d)



* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Serial Interfacing

Table 3-05. XBee-PRO Commands - Serial Interfacing

AT Command	Command Category	Name and Description	Parameter Range	Default
BD	Serial Interfacing	**Interface Data Rate**. Set/Read the serial interface data rate for communications between the RF module serial port and host. Request non-standard baud rates with values above 0x80 using a terminal window. Read the BD register to find actual baud rate achieved.	0 - 7 (standard baud rates) 0 = 1200 bps 1 = 2400 2 = 4800 3 = 9600 4 = 19200 5 = 38400 6 = 57600 7 = 115200 0x80 - 0x3D090 (non-standard baud rates up to 250 Kbps)	3
RO	Serial Interfacing	**Packetization Timeout**. Set/Read number of character times of inter-character delay required before transmission. Set to zero to transmit characters as they arrive instead of buffering them into one RF packet.	0 - 0xFF [x character times]	3
AP (v1.x80*)	Serial Interfacing	**API Enable**. Disable/Enable API Mode.	0 - 2 0 =Disabled 1 = API enabled 2 = API enabled (w/escaped control characters)	0
NB	Serial Interfacing	**Parity**. Set/Read parity settings.	0 - 4 0 = 8-bit no parity 1 = 8-bit even 2 = 8-bit odd 3 = 8-bit mark 4 = 8-bit space	0
PR (v1.x80*)	Serial Interfacing	**Pull-up Resistor Enable**. Set/Read bitfield to configure internal pull-up resistor status for I/O lines Bitfield Map: bit 0 - AD4/DIO4 (pin11) bit 1 - AD3 / DIO3 (pin17) bit 2 - AD2/DIO2 (pin18) bit 3 - AD1/DIO1 (pin19) bit 4 - AD0 / DIO0 (pin20) bit 5 - RTS / AD6 / DIO6 (pin16) bit 6 - DTR / SLEEP_RQ / DI8 (pin9) bit 7 - DIN/CONFIG (pin3) Bit set to "1" specifies pull-up enabled; "0" specifies no pull-up	0 - 0xFF	0xFF

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

I/O Settings

Table 3-06. XBee-PRO Commands - I/O Settings (sub-category designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
D8	I/O Settings	**DI8 Configuration**. Select/Read options for the DI8 line (pin 9) of the RF module.	0 - 1 0 = Disabled 3 = DI (1,2,4 & 5 n/a)	0
D7 (v1.x80*)	I/O Settings	**DIO7 Configuration**. Select/Read settings for the DIO7 line (pin 12) of the RF module. Options include CTS flow control and I/O line settings.	0 - 1 0 = Disabled 1 = CTS Flow Control 2 = (n/a) 3 = DI 4 = DO low 5 = DO high 6 = RS485 Tx Enable Low 7 = RS485 Tx Enable High	1
D6 (v1.x80*)	I/O Settings	**DIO6 Configuration**. Select/Read settings for the DIO6 line (pin 16) of the RF module. Options include RTS flow control and I/O line settings.	0 - 1 0 = Disabled 1 = RTS flow control 2 = (n/a) 3 = DI 4 = DO low 5 = DO high	0

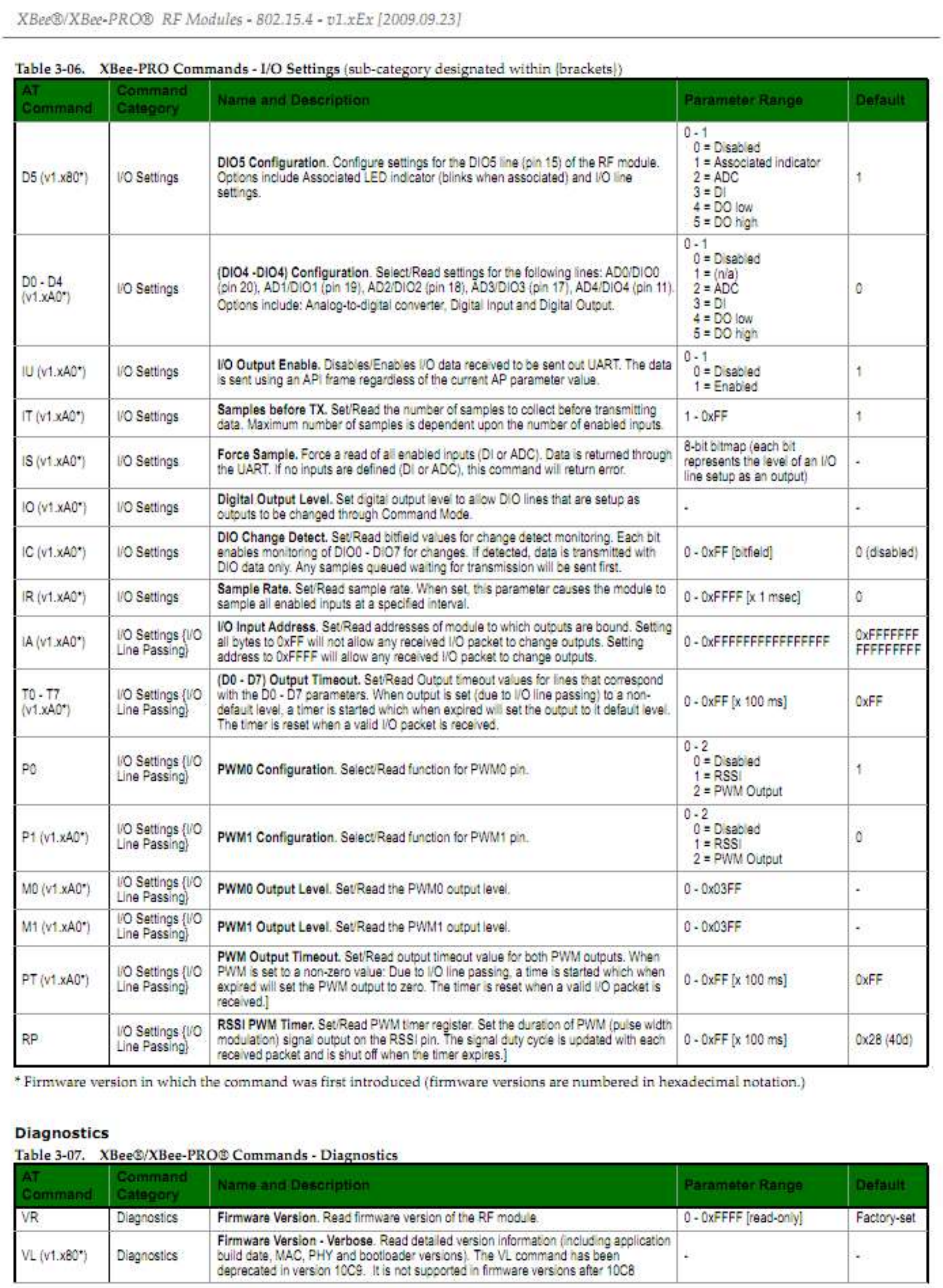


Table 3-06. XBee-PRO Commands - I/O Settings (sub-category designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
D5 (v1.x80*)	I/O Settings	**DIO5 Configuration**. Configure settings for the DIO5 line (pin 15) of the RF module. Options include Associated LED indicator (blinks when associated) and I/O line settings.	0 - 1 0 = Disabled 1 = Associated indicator 2 = ADC 3 = DI 4 = DO low 5 = DO high	1
D0 - D4 (v1.xA0*)	I/O Settings	**(DIO4 -DIO4) Configuration**. Select/Read settings for the following lines: AD0/DIO0 (pin 20), AD1/DIO1 (pin 19), AD2/DIO2 (pin 18), AD3/DIO3 (pin 17), AD4/DIO4 (pin 11). Options include: Analog-to-digital converter, Digital Input and Digital Output.	0 - 1 0 = Disabled 1 = (n/a) 2 = ADC 3 = DI 4 = DO low 5 = DO high	0
IU (v1.xA0*)	I/O Settings	**I/O Output Enable.** Disables/Enables I/O data received to be sent out UART. The data is sent using an API frame regardless of the current AP parameter value.	0 - 1 0 = Disabled 1 = Enabled	1
IT (v1.xA0*)	I/O Settings	**Samples before TX.** Set/Read the number of samples to collect before transmitting data. Maximum number of samples is dependent upon the number of enabled inputs.	1 - 0xFF	1
IS (v1.xA0*)	I/O Settings	**Force Sample.** Force a read of all enabled inputs (DI or ADC). Data is returned through the UART. If no inputs are defined (DI or ADC), this command will return error.	8-bit bitmap (each bit represents the level of an I/O line setup as an output)	-
IO (v1.xA0*)	I/O Settings	**Digital Output Level.** Set digital output level to allow DIO lines that are setup as outputs to be changed through Command Mode.	-	-
IC (v1.xA0*)	I/O Settings	**DIO Change Detect.** Set/Read bitfield values for change detect monitoring. Each bit enables monitoring of DIO0 - DIO7 for changes. If detected, data is transmitted with DIO data only. Any samples queued waiting for transmission will be sent first.	0 - 0xFF [bitfield]	0 (disabled)
IR (v1.xA0*)	I/O Settings	**Sample Rate.** Set/Read sample rate. When set, this parameter causes the module to sample all enabled inputs at a specified interval.	0 - 0xFFFF [x 1 msec]	0
IA (v1.xA0*)	I/O Settings {I/O Line Passing}	**I/O Input Address.** Set/Read addresses of module to which outputs are bound. Setting all bytes to 0xFF will not allow any received I/O packet to change outputs. Setting address to 0xFFFF will allow any received I/O packet to change outputs.	0 - 0xFFFFFFFFFFFFFFFF	0xFFFFFFFF FFFFFFFF
T0 - T7 (v1.xA0*)	I/O Settings {I/O Line Passing}	**(D0 - D7) Output Timeout.** Set/Read Output timeout values for lines that correspond with the D0 - D7 parameters. When output is set (due to I/O line passing) to a non-default level, a timer is started which when expired will set the output to it default level. The timer is reset when a valid I/O packet is received.	0 - 0xFF [x 100 ms]	0xFF
P0	I/O Settings {I/O Line Passing}	**PWM0 Configuration**. Select/Read function for PWM0 pin.	0 - 2 0 = Disabled 1 = RSSI 2 = PWM Output	1
P1 (v1.xA0*)	I/O Settings {I/O Line Passing}	**PWM1 Configuration**. Select/Read function for PWM1 pin.	0 - 2 0 = Disabled 1 = RSSI 2 = PWM Output	0
M0 (v1.xA0*)	I/O Settings {I/O Line Passing}	**PWM0 Output Level**. Set/Read the PWM0 output level.	0 - 0x03FF	-
M1 (v1.xA0*)	I/O Settings {I/O Line Passing}	**PWM1 Output Level**. Set/Read the PWM1 output level.	0 - 0x03FF	-
PT (v1.xA0*)	I/O Settings {I/O Line Passing}	**PWM Output Timeout.** Set/Read output timeout value for both PWM outputs. When PWM is set to a non-zero value: Due to I/O line passing, a time is started which when expired will set the PWM output to zero. The timer is reset when a valid I/O packet is received.]	0 - 0xFF [x 100 ms]	0xFF
RP	I/O Settings {I/O Line Passing}	**RSSI PWM Timer.** Set/Read PWM timer register. Set the duration of PWM (pulse width modulation) signal output on the RSSI pin. The signal duty cycle is updated with each received packet and is shut off when the timer expires.]	0 - 0xFF [x 100 ms]	0x28 (40d)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Diagnostics

Table 3-07. XBee®/XBee-PRO® Commands - Diagnostics

AT Command	Command Category	Name and Description	Parameter Range	Default
VR	Diagnostics	**Firmware Version**. Read firmware version of the RF module.	0 - 0xFFFF [read-only]	Factory-set
VL (v1.x80*)	Diagnostics	**Firmware Version - Verbose**. Read detailed version information (including application build date, MAC, PHY and bootloader versions). The VL command has been deprecated in version 10C9. It is not supported in firmware versions after 10C8	-	-

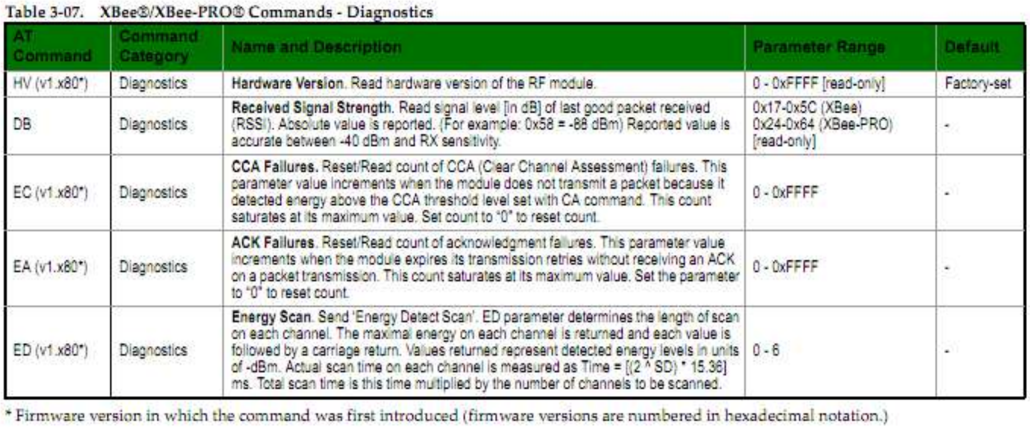
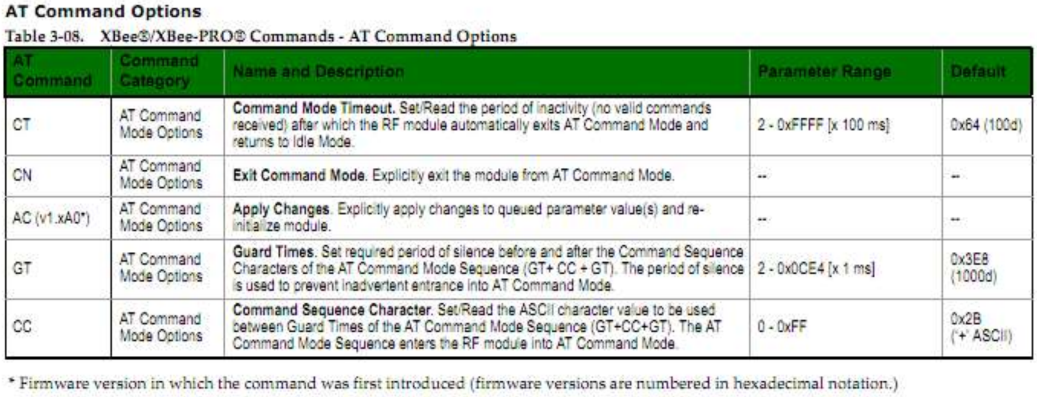


Table 3-07. XBee®/XBee-PRO® Commands - Diagnostics

AT Command	Command Category	Name and Description	Parameter Range	Default
HV (v1.x80*)	Diagnostics	**Hardware Version**. Read hardware version of the RF module.	0 - 0xFFFF [read-only]	Factory-set
DB	Diagnostics	**Received Signal Strength**. Read signal level [in dB] of last good packet received (RSSI). Absolute value is reported. (For example: 0x58 = -88 dBm) Reported value is accurate between -40 dBm and RX sensitivity.	0x17-0x5C (XBee) 0x24-0x64 (XBee-PRO) [read-only]	-
EC (v1.x80*)	Diagnostics	**CCA Failures.** Reset/Read count of CCA (Clear Channel Assessment) failures. This parameter value increments when the module does not transmit a packet because it detected energy above the CCA threshold level set with CA command. This count saturates at its maximum value. Set count to "0" to reset count.	0 - 0xFFFF	-
EA (v1.x80*)	Diagnostics	**ACK Failures**. Reset/Read count of acknowledgment failures. This parameter value increments when the module expires its transmission retries without receiving an ACK on a packet transmission. This count saturates at its maximum value. Set the parameter to "0" to reset count.	0 - 0xFFFF	-
ED (v1.x80*)	Diagnostics	**Energy Scan**. Send 'Energy Detect Scan'. ED parameter determines the length of scan on each channel. The maximal energy on each channel is returned and each value is followed by a carriage return. Values returned represent detected energy levels in units of -dBm. Actual scan time on each channel is measured as Time = [(2 ^ SD) * 15.36] ms. Total scan time is this time multiplied by the number of channels to be scanned.	0 - 6	-

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)



AT Command Options

Table 3-08. XBee®/XBee-PRO® Commands - AT Command Options

AT Command	Command Category	Name and Description	Parameter Range	Default
CT	AT Command Mode Options	**Command Mode Timeout.** Set/Read the period of inactivity (no valid commands received) after which the RF module automatically exits AT Command Mode and returns to Idle Mode.	2 - 0xFFFF [x 100 ms]	0x64 (100d)
CN	AT Command Mode Options	**Exit Command Mode**. Explicitly exit the module from AT Command Mode.	--	--
AC (v1.xA0*)	AT Command Mode Options	**Apply Changes**. Explicitly apply changes to queued parameter value(s) and re-initialize module.	--	--
GT	AT Command Mode Options	**Guard Times**. Set required period of silence before and after the Command Sequence Characters of the AT Command Mode Sequence (GT+ CC + GT). The period of silence is used to prevent inadvertent entrance into AT Command Mode.	2 - 0x0CE4 [x 1 ms]	0x3E8 (1000d)
CC	AT Command Mode Options	**Command Sequence Character**. Set/Read the ASCII character value to be used between Guard Times of the AT Command Mode Sequence (GT+CC+GT). The AT Command Mode Sequence enters the RF module into AT Command Mode.	0 - 0xFF	0x2B ('+' ASCII)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Appendix B Baseline Data and Energy Savings Calculation

Baseline Data

From the UTAR electricity bill that we obtain, we able to obtain the power consumption average of UTAR in one month according to the Tariff C1:

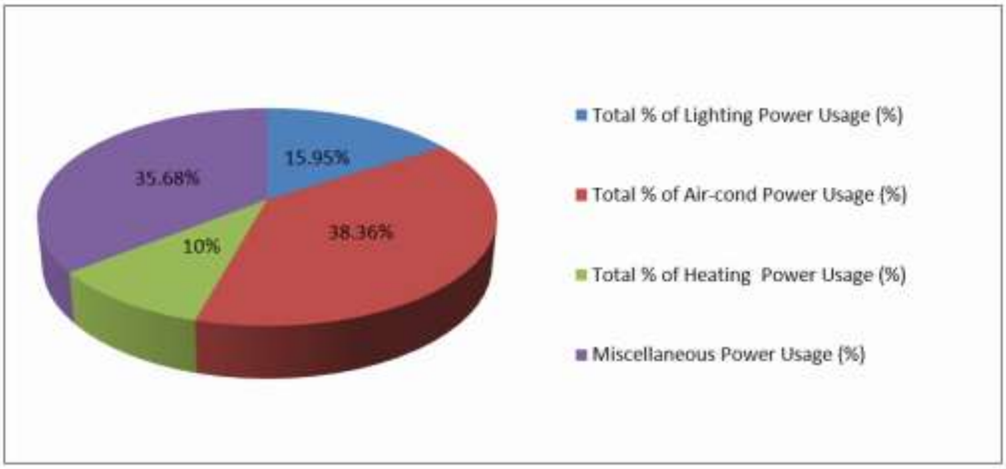
Tariff C1 – Medium Voltage General Commercial Tariff

- For each kW of maximum demand per month, tariff is RM 24.60 / kW
- For all units, tariff is RM0.296 / kWh
- Minimum monthly charge is RM 500.00

Total Energy Usage	**230862 kWh**
Maximum Demand	1009.27 kW
Cost / Month	RM 93 663.31

*Refer to UTAR's Electricity.xlsx for more detail

Also, we also obtain the UTAR floor plan which contains the number of lights in the UTAR. From the data we obtain and the survey made in UTAR for HVAC consumption, we able to obtain a rough data for our UTAR power consumption in Setapak campus.



*Refer to UTAR's Electricity.xlsx for more detail

Thus is will become the main aim for out saving achieve in our UTAR Setapak Campus.

Lighting Control System Savings

As mention previously, we manage to obtain the UTAR floor plans which contain the position of the lights in one floor. Thus, we made counting from there and obtain the number of lights for each block and each room and we manage to obtain the total number of lights and the total power consumption.

Also, we manage to obtain the ratio of the lights between class room and non-class room (such as office, canteen, and corridor).

Total lights consumption in one month = RM 10 880.00

Total power consumption for class room in one month = RM 4 912.44

Total power consumption for non-class room in one month = RM 5 967.97

Ratio of lights (classroom: non-class room) – 1:1.22

On the other hand, we manage to obtain the time table schedule of each class in one week for that particular semester. Hence, another data obtain which is the total power consumption in one class in one day.

As we gather all the data we obtain, we manage to calculate and obtain for some useful information with some assumption made:

Assumption:

SD201											
Time	8-9	9-10	10-11	11-12	12-1	1-2	2-3	3-4	4-5	5-6	6-7
Monday		▨	▨	▨	▨				▨	▨	

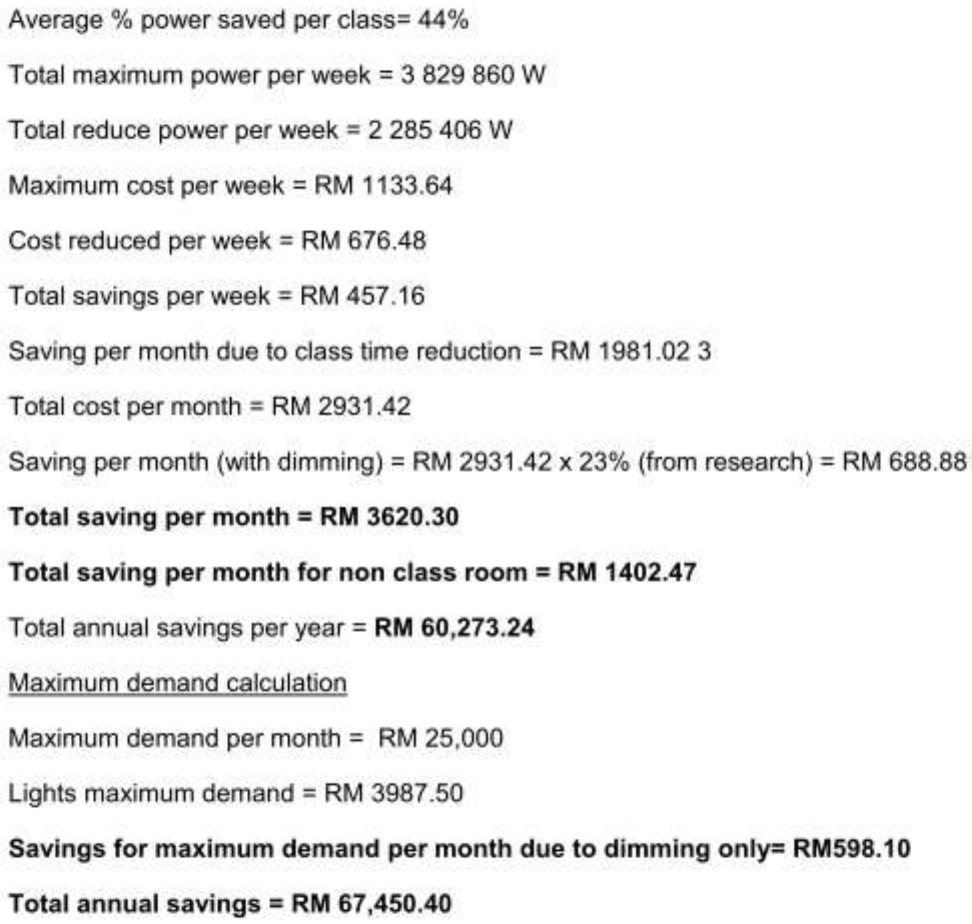
As shown in the table above, it was extract from one of the classes in Wednesday from the time table we obtain. There are 2 assumptions we made, that is:

- Before implementing the system into our class, we made the assumption that on early the morning, the lights of the class will be turn ON at 9a.m. then it would not be turn off until 7p.m. which the security will patrol around the class and turn OFF the lights on that particular class.
- After implement our system into the class, the lights will be automatically turn ON and OFF according to the occurrence of the people in the class which mean following the class time table schedule.

Note that the assumption and calculation made is just based on turning ON and OFF,

Thus, we calculate that if the system is implementing in each class, each class will obtain in average about 44% of saving by just controlling the lights automatically turning ON and OFF (without dimming factor).

Further calculation made based on the data and result we obtain (Calculation is just made on whole lecture and tutorial classes and without dimming factor):



Average % power saved per class= 44%

Total maximum power per week = 3 829 860 W

Total reduce power per week = 2 285 406 W

Maximum cost per week = RM 1133.64

Cost reduced per week = RM 676.48

Total savings per week = RM 457.16

Saving per month due to class time reduction = RM 1981.02 3

Total cost per month = RM 2931.42

Saving per month (with dimming) = RM 2931.42 x 23% (from research) = RM 688.88

Total saving per month = RM 3620.30

Total saving per month for non class room = RM 1402.47

Total annual savings per year = **RM 60,273.24**

Maximum demand calculation

Maximum demand per month = RM 25,000

Lights maximum demand = RM 3987.50

Savings for maximum demand per month due to dimming only= RM598.10

Total annual savings = RM 67,450.40

HVAC savings calculation

We have discussed that HVAC consumes UTAR FES campus 38.36% of electricity cost. We also knew that the AHU in UTAR FES campus is direct-on-line method. By using Schneider Electric ECO2 software, we could estimate how much savings on AHU blower alone.

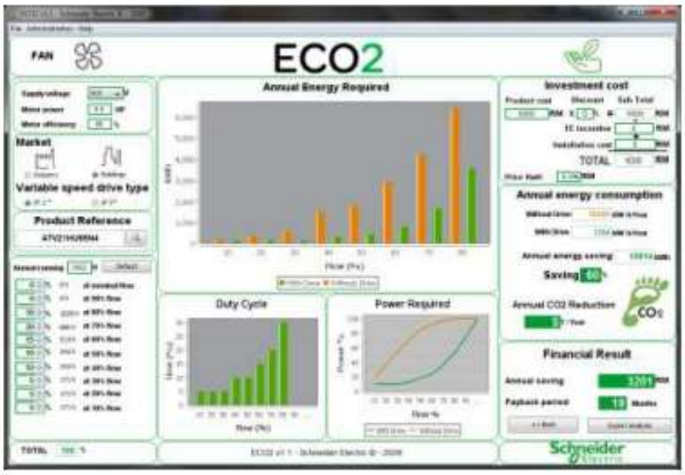


Figure 18: ECO2 on AHU calculation

Parameters in using ECO2:

Market			Buildings						
Type of application			FAN						
Consumption at nominal flow			11.0 A						
Type of mechanical regulation			Outlet damper						
			Motor (nominal values)						
Power			5.5 HP						
Power factor			0.79						
Efficiency			0.88						
Voltage			415 V						
CYCLIC TIME			3432 h						
100%	90%	80%	70%	60%	50%	40%	30%	20%	10%
0 h	0 h	1029 h	686 h	514 h	343 h	343 h	171 h	171 h	171 h

The annual savings are reversed calculated based on the average bill from UTAR. We know that HVAC accounts for 38.36% of the total bill. The ratio between central air conditioner and split unit air conditioner are as below:

BLOCK	Power Consumption (kW)	Air cond types
SA	40.00	Central
SB	40.00	Central
SC	7.50	Split unit
SD	72.70	Split unit
SE	72.70	Split unit
SF	24.30	Split unit
DK	60.00	Central
S	70.00	Split unit
TOTAL SUM:	387.20	

Therefore, the ratio between central air conditioner and split unit air conditioner is 140kW:247.2kW which is then simplify to **1:1.76**.

The calculation for HVAC savings from average electricity bill:

Average monthly bill = RM 93000

Average Maximum Demand = RM 25000

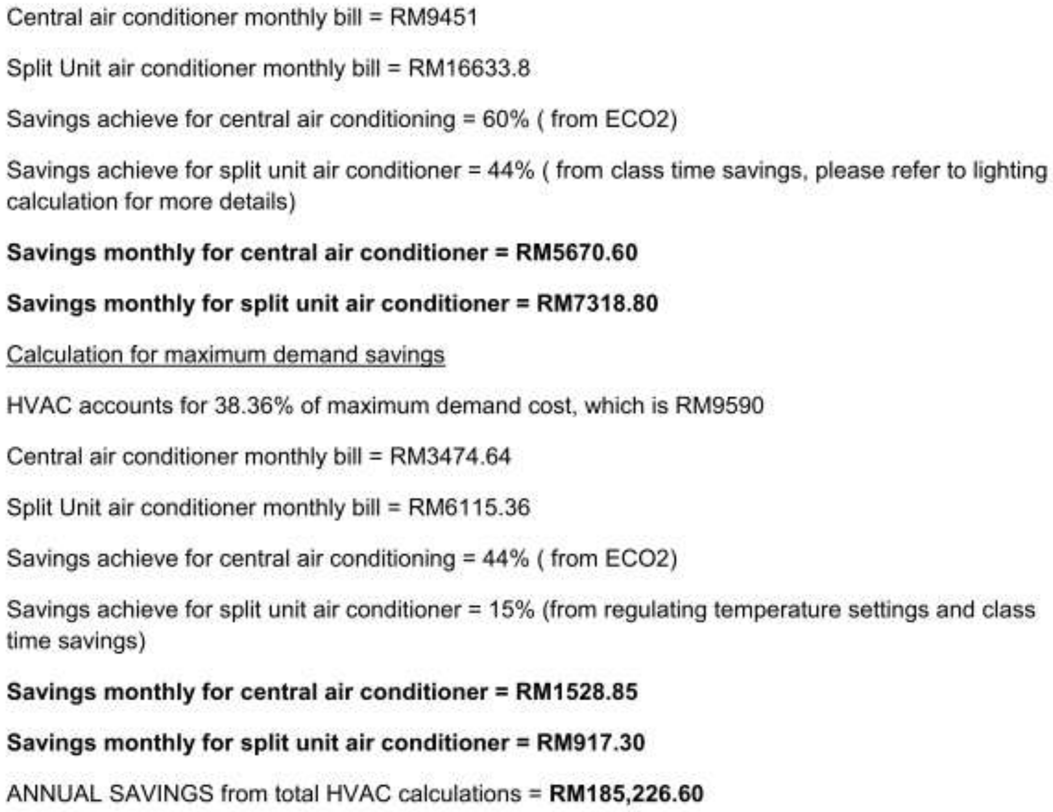
<u>Calculation for monthly savings without maximum demand</u>

Average monthly bill without maximum demand = RM68000

HVAC accounts for 38.36% of average monthly bill, which is RM26084.80

*Central air conditioner : Split unit air conditioner ratio = 1:1.76

Therefore,



Central air conditioner monthly bill = RM9451

Split Unit air conditioner monthly bill = RM16633.8

Savings achieve for central air conditioning = 60% (from ECO2)

Savings achieve for split unit air conditioner = 44% (from class time savings, please refer to lighting calculation for more details)

Savings monthly for central air conditioner = RM5670.60

Savings monthly for split unit air conditioner = RM7318.80

Calculation for maximum demand savings

HVAC accounts for 38.36% of maximum demand cost, which is RM9590

Central air conditioner monthly bill = RM3474.64

Split Unit air conditioner monthly bill = RM6115.36

Savings achieve for central air conditioning = 44% (from ECO2)

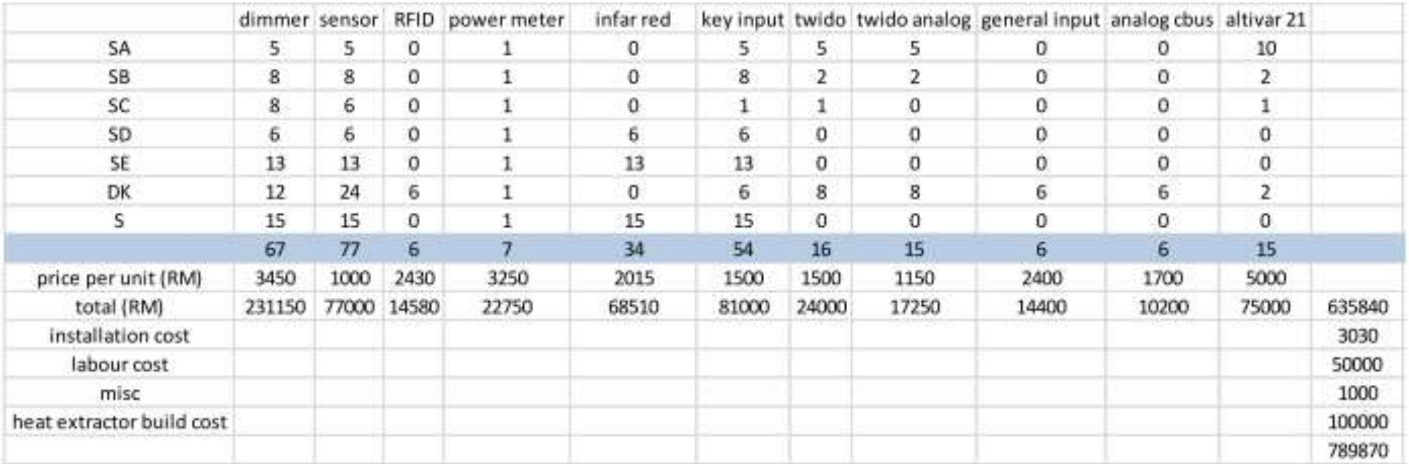
Savings achieve for split unit air conditioner = 15% (from regulating temperature settings and class time savings)

Savings monthly for central air conditioner = RM1528.85

Savings monthly for split unit air conditioner = RM917.30

ANNUAL SAVINGS from total HVAC calculations = **RM185,226.60**

Cost and bill of material



	dimmer	sensor	RFID	power meter	infar red	key input	twido	twido analog	general input	analog cbus	altivar 21	
SA	5	5	0	1	0	5	5	5	0	0	10	
SB	8	8	0	1	0	8	2	2	0	0	2	
SC	8	6	0	1	0	1	1	0	0	0	1	
SD	6	6	0	1	6	6	0	0	0	0	0	
SE	13	13	0	1	13	13	0	0	0	0	0	
DK	12	24	6	1	0	6	8	8	6	6	2	
S	15	15	0	1	15	15	0	0	0	0	0	
	67	77	6	7	34	54	16	15	6	6	15	
price per unit (RM)	3450	1000	2430	3250	2015	1500	1500	1150	2400	1700	5000	
total (RM)	231150	77000	14580	22750	68510	81000	24000	17250	14400	10200	75000	635840
installation cost												3030
labour cost												50000
misc												1000
heat extractor build cost												100000
												789870

- Dimmer : Clipsal C-Bus 4 Channel Universal Dimmer L5504D2UP
- Sensor: Clipsal C-Bus Multi Sensor 5753 PEIRL
- Key input: Clipsal C-Bus LCD Input 5085DL GF
- Infrared red: Clipsal C-Bus IR transmitter unit C5034NIRTWE (for non-centralized air conditioner)
- Analog cbus: Clipsal C-Bus Analog output unit L5504AMP (for centralized air conditioner)
- Power meter: Schneider Electric Power Meter PM750
- Altivar 21: Schneider Electric Altivar ATV21HU55N4
- Twido: Schneider Electric Twido Controller TWDLCAA24DRF
- Twido analog: Schneider Electric Analog Module TM2AMM3HT
- General input: C-Bus General Input unit 5504GL