

CAR SPEED CONTROLLER USING BLUETOOTH

(Wireless Communication-BLUETOOTH APPLICATION)

V.BALAJI

Department Of Computer Science

I.F.E.T College Of Engineering

vbbalajir@gmail.com

ABSTRACT:

In the current scenario the world is plagued by accidents which are primarily due to human errors in judgment and hence thousands of lives are lost. These accidents can be avoided if only there was a mechanism to alert the driver of approaching danger. This can be done by monitoring the distance between nearby cars and alerting the driver whenever the distance becomes too short. This is precisely the aim of this paper. In this paper we propose the use of Bluetooth Technology by which we can check the speed of the car whenever it comes dangerously close to any other vehicle up front, thereby saving very many lives.

1) INTRODUCTION

Since Bluetooth devices are capable of communicating with eight other devices simultaneously we can monitor and check the speeds of up to eight cars simultaneously, thus preventing accidents. Thus if we have two Bluetooth enabled devices in two cars the devices automatically communicate with each other when they come in the range of up to 100 meters of each other. The range is dependent on the power class of the product. Power transmission rates vary in many Bluetooth devices depending upon the power saving features available in a particular unit, bandwidth requirements, transmission distance. The statistics of road accidents is tremendous and highlights the need for such a system. The following is a statistic on the number of road accidents occurring each year.

2) OPERATION

Because of traveling at high speeds there is a possibility of having accident. The figure 2 shows that when two cars or more come within the distance of 10 km at high speeds there is a possibility of having accidents. The Bluetooth radio is a short distance, low power radio operating in the unlicensed spectrum of 2.4 GHz and using a nominal antenna power of 20 dB m.

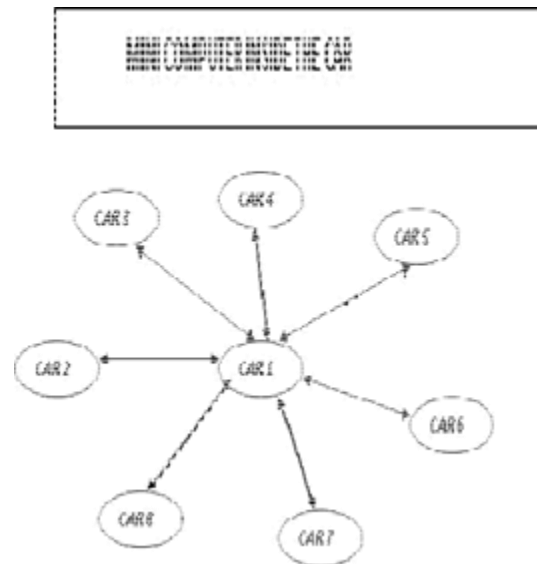


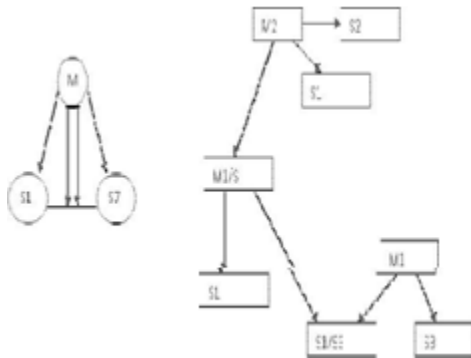
Fig 2

At the 20 dB the range is 100 meters, meaning equipment must be Within 100 meters to each other (about 328 feet) to communicate using the Bluetooth standard. With the help of this technology we can send data to the eight

devices. The group of eight devices is known as piconet. Here we have a piconet and a scatternet, in the piconet M is the master and S1 to S7 are the slaves

Fig 3

Pico net formed by Bluetooth devices

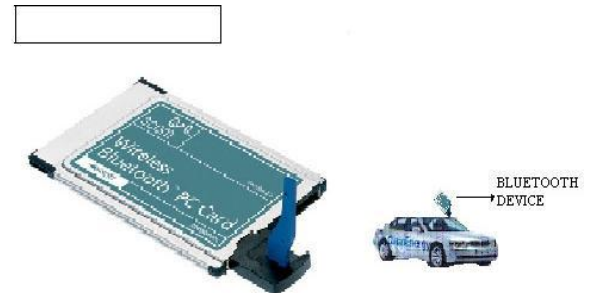


Radio communication is subjected to noise and interference, as the 2.4 GHz frequencies are shared between the all device in piconet. So the Bluetooth specification has solved this problem by employing what is called as spectrum spreading, in which the Bluetooth radio hops among different frequencies very quickly. There are 79 hops starting at 2.402 GHz and stopping at 2.480 GHz, each of which is displaced by 1 MHz. The Bluetooth avoids interference by hopping around these 79 frequencies 1600 times per second. So in order to avoid it we use bluetooth equipped car, in which each car have Bluetooth transmitter and receiver. And the every car should have minicomputer to monitor the relative position of the car with the other car.

When any car comes close together bluetooth device sends warning signal to the car. Based on the type of warning signal received the computer sends signal to the brake control system to slow down the speed of the car. There are two types of control signals. First type of signal control the speed of the car and the second type of signal is to overtake the car which is moving forward.

Bluetooth device

Fig 4



3) AUTOMTIC BRAKE SYSTEM:

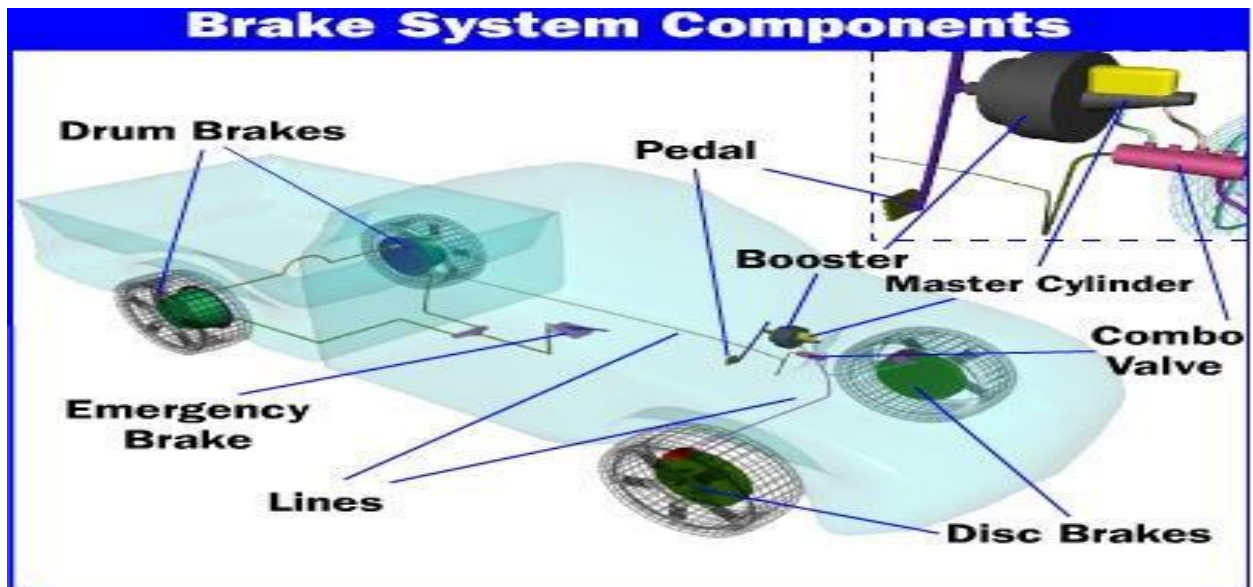
The automatic brake system is the next generation braking system for controlling the speed of the car. On receiving the control signal from the traveling car the computer inside the car manipulates the signal and gives control signal to the braking system.

There are four main components to an automatic braking system:

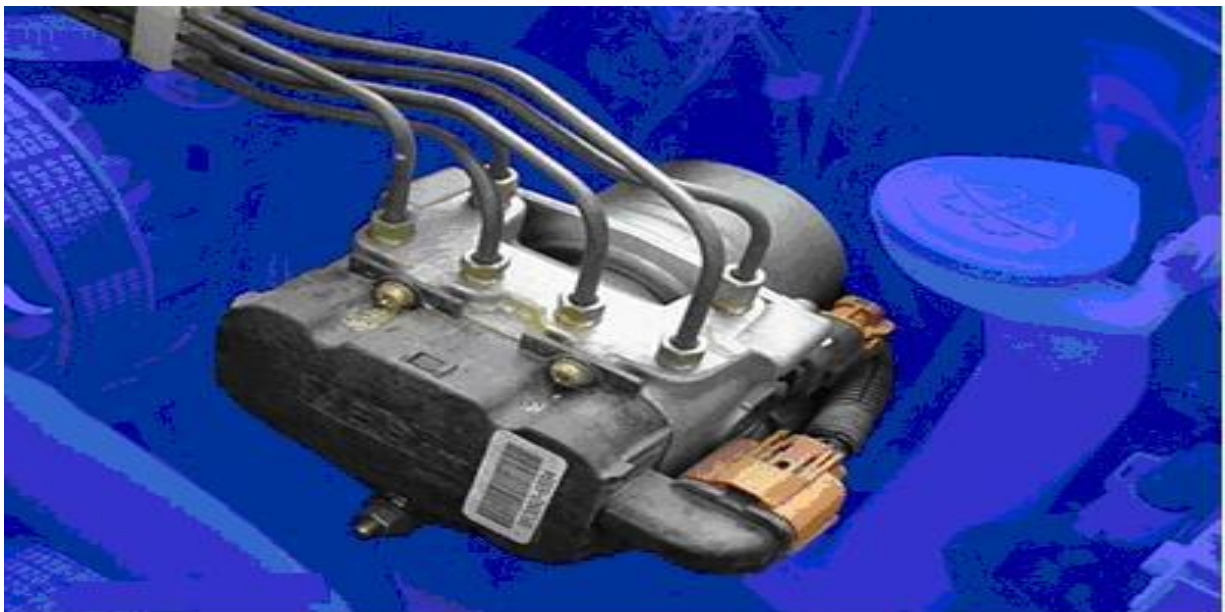
- 1) speed sensors
- 2) pump
- 3) valves
- 4) controller

The computer constantly monitors the distance between each of these cars and when it senses that the car is getting too close it moves the hydraulic valves to increase the pressure on the braking circuit, effectively increasing the braking force on the wheels. If the distance between two vehicles is within the 100m the Bluetooth devices get enabled and if the distance comes closer within 10m the automatic braking system takes the control. After the speed of the car is reduced and distance increased the hydraulic valves decreases the pressure on the braking circuit, thus effectively decrease the braking force on the wheels.

Diagram of the car(Fig 5)



Automatic braking pump and
Valves(Fig 6)



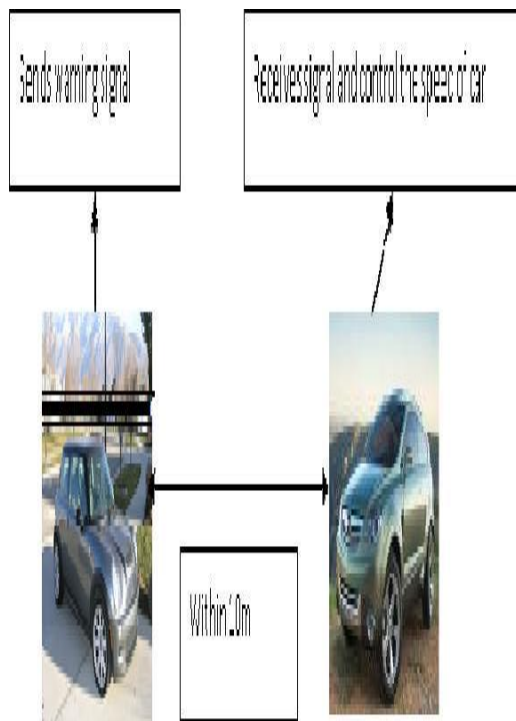
The following steps show the various functions of the hydraulic valves:

- In position one, the valve is **open**; pressure from the master cylinder is passed right through to the brake.
- In position two, the valve **blocks** the line, isolating that brake from the master cylinder. This prevents the pressure from rising further should the driver push the brake pedal harder.
- In position three, the valve **releases** some of the pressure from the brake.

PROPOSED MODEL

In the figure 7 when car A and car B come within the range of 100m both the Bluetooth devices get enabled and if any one of the car comes too fast then the bluetooth device sends a warning signal to the other car and it processes the signal and gives it to the automatic braking system.

Representing Proposed Model
(Fig 7)



4) CONCLUSION

This technology is being widely adopted by the Industry leaders. The possibility for new applications is very exciting with this versatile technology. It provides a simple, logical answer to all the Problems-which is built a single common radio into every mobile computer, then neither do companies have to worry about WAN, nor do communication companies need to worry about building external cables. The Bluetooth communication device will thus be a small, low powered radio in a chip that will talk to other Bluetooth enabled products. Bluetooth has been designed to solve a number of connectivity problems experienced by the mobile workers & consumers. Thus, this technology helps make the electronic devices more users friendly and helps address various other problems like accidents.

REFERENCES

1. <http://Bluetoothtechnologies.in>
2. <http://Applicationsofbluetooth.in>
3. [http://Accident Prevention Techniques](http://AccidentPreventionTechniques)

Project Report

On

Remote Control Mouse using TV Remote

By

Pijus Kumar Saha	(ID No.: 083820001)
Md. Majadul Haque Rana	(ID No.: 082820002)
S.M. Mortaza Shakil	(ID No. : 083820005)
Amirul Islam Mahmud	(ID No. : 082820005)

Supervised By :

Dr. Md. Abu Bakar Siddiquei

Assistant Professor, Department
of ETE

Faculty of Engineering & Technology
Eastern University, Dhaka, Bangladesh.



DEPARTMENT OF ELECTRONIC AND TELECOMMUNICATION ENGINEERING
EASTERN UNIVERSITY
DHAKA, BANGLADESH

September, 2012

Remote Control Mouse using TV Remote

A Project Submitted to the Eastern University in Partial Fulfillment of the
Requirements for the Degree of Bachelor of Science in Electronics and
Telecommunication Engineering

Submitted By

Pijus Kumar Saha
ID: 083820001

Md. Majadul Haque Rana
ID: 082820002

S.M. Mortaza Shakil
ID : 083820005

Amirul Islam Mahmud
ID : 082820005

**DEPARTMENT OF ELECTRONIC AND TELECOMMUNICATION ENGINEERING
EASTERN UNIVERSITY**

September, 2012

DECLARATION

It is, hereby, declared that the work presented in this project report is done by the authors under the supervision of **Dr. Md. Abu Bakar Siddiquei**, Assistant Professor, Department of Electronic and Telecommunication Engineering, Eastern University. It is also declared that neither this report nor any part of it has been submitted elsewhere for the award of any kind of degree.

External

(Ms. Farzana Rahmat Zaki)

Senior lecturer

Department of EEE

Faculty of Engineering & Technology

Eastern University, Dhaka, Bangladesh.

(Pijus Kumar Saha)

ID: 083820001

(Md. Majadul Haque Rana)

ID: 082820002

Supervisor

(Dr. Md. Abu Bakar Siddiquei)

Assistant Professor,

Department of ETE

Faculty of Engineering & Technology

Eastern University, Dhaka, Bangladesh.

(S.M. Mortaza Shakil)

ID: 083820005

(Amirul Islam Mahmud)

ID : 082820005

Approval

The thesis report on “Remote control mouse using TV remote” has submitted to the following respected member the board of examiners of the faculty of engineering in partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical and Electronic Engineering on September 2012 by following students and has accepted as satisfactory.

- | | |
|----------------------------------|-----------------------|
| 1. Pijus Kumar Saha | ID: 083820001 |
| 2. Md. Majadul Haque Rana | ID: 082820002 |
| 3. S.M. Mortaza Shakil | ID : 083820005 |
| 4. Amirul Islam Mahmud | ID : 082820005 |

Supervisor

(Dr. Md. Abu Bakar Siddiquei)

Assistant Professor
Department of ETE
Eastern University
Dhaka, Bangladesh.

Dean

(Prof. Dr. Nurul Islam)

Dean
Faculty of Engineering & Technology
Eastern University
Dhaka, Bangladesh.

CERTIFICATION

A project titled

Remote Control Mouse using TV Remote

Submitted by Pijus Kumar Saha (Student ID.: 083820001), S.M. Mortaza Shakil (Student ID.: 083820005), Majadul Haque Rana (Student ID.: 082820002), Amirul Islam Mahmud (Student ID.: 082820005) students of B.Sc. Engineering has been accepted as satisfactory in partial fulfillment for the degree of Bachelor Electronic & Telecommunication Engineering on August 2012.

(Dr. Md. Abu Bakar Siddiquei)

Assistant Professor

ETE, Eastern University

ACKNOWLEDGEMENTS

First, we would like to express our deepest gratitude to our supervisor **Dr. Md. Abu Bakar Siddiquei** for his excellent guidance and continual support during the course of our degree and the project. Second, we would like to thank **Prof. Dr. Mirza Golam Rabbani** for his valuable reference, continuous encourage and suggestions during this thesis of our studies. For their valuable teaching in different levels of the course which has helped us a lot in this thesis work, we would like to thank all the faculty and stuff of Electrical and Electronic Engineering (EEE) department of Eastern University (EU).

The financial support provided by our family for the project entitled "**Remote Control Mouse using TV Remote**"

We would like to thank our senior S. M. Anayetullah and our friend Md. Al-Amin for providing valuable time, which greatly help us to complete the project work.

Finally, we are deeply indebted to our family for their love and support throughout this degree and our life.

ABSTRACT

This final report is established to fulfill the Final year Project/Thesis requirement for Engineering for final semester student, a standard guideline is prepared to standardize various documentations and this report is done mainly to list down all the important aspect of the project or the electrical and electronic device that designed.

We introduce a method of how electronics devices and machines can be controlled remotely from an operator/user. As a practical model we design a microcontroller Atmega8 L-8PU based receiver circuit that may be connected to Personal computer. This project is an implementation of Sony remote reception on an Atmega8 L-8PU microcontroller. The received code is decoded and sent to the PC IR remote software developed using high level graphical programming language (Visual Basic). The cursor position is moved according to the keys pressed.

The project report focuses on the system which is designed by using the Microcontroller Atmega8 L-8PU. The infrared receiver have three pins in which two pins are for +5v supply and ground while the third pin is for data output. The Infrared Receiver is designed for demodulating the frequency of 30 kHz to 40 kHz, for example, TSOP1738 is designed for demodulating frequency of 38 kHz which is used in our project.

The IR receiver module receives the data sent by remote handset, amplifies, demodulates and converts it to MCU compatible voltage format and outputs it on its data output pin. The microcontroller decodes the infrared signal data and the microcontroller will sent the infrared Key code to the PC through the USB port.

After getting this key code of the Remote control the program compare the key code with the code present in the program and if they are equal then the corresponding function was done.

Table of Contents

	Page
DECLARATION	ii
APPROVAL	iii
CERTIFICATION	iv
ACKNOWLEDGEMENT	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	ix
LIST OF TABLES	x
LIST OF SYMBOLS AND COLLABORATION	xi
 Chapter-01 : Overview	
1.1 Introduction	01
1.2 Aim of the Project	01
1.3 Components and Methodology	01
1.3.1 Hardware Components	01
1.3.2 Overview Block Diagram	02
1.4 Significance and Applications	02
1.5 Organization of the Report	03
 Chapter-02 : Hardware	
2.1 Algorithm Flowchart	04
2.2 Used Components and Description	05
2.2.1 List of Component Used	05
2.2.1.1 Description of the Circuit Components	06
2.2.1.1.1 : Resistor	06
2.2.1.1.2 : Zener Diode	07
2.2.1.1.3 : Capacitor	08
2.2.1.1.4 : 555 Timer to Modulate Infrared (IR) Light	08
2.2.1.1.5 : IR Receiver	09
2.2.1.1.6 : Crystal Oscillator (XTAL – 12.00 MHz)	10
2.2.1.1.7 : Atmega8 L-8PU Microcontroller	11

2.3	Circuit Operation	13
2.3.1	Transmitting Data via TV Remote	13
2.3.2	Decoding Data	15
2.3.3	Decoder data transmit to the pc via USB port	15

Chapter-03 : Methodology

3.1	Introduction to Embedded System	16
3.2	Applications of Embedded System	17
3.3	Microcontrollers for Embedded Systems	18
3.4	Circuit Diagram	19
3.5	PCB Layout	21
3.5.1	Top Silk	21
3.5.2	Bottom Copper	21
3.5.3	PCB Board with Components	22
3.5.4	PCB Board with Bottom copper	22
3.6	Project Board with Components	23
3.7	Initialization	23
3.8	Generating Delay	23
3.8.1	Defined Mode	24
3.8.2	Compare with SIRC Protocol	24
3.8.3	Transmit	24

Chapter-04 : Results & Discussion

4.1	Operating Manual	25
4.2	Specifications of the Project	25
4.3	Device Specification	26
4.4	Advantage	26
4.5	Limitation	27
4.6	Cost of Practical Work	27

Chapter-05 : Conclusion

5.1	Conclusion	28
5.2	Future Work	28

Reference	29
------------------	-----------

Appendix	30-39
-----------------	--------------

List of Figures

Figure no.	Title	Page
Figure 1.1	Overview Block Diagram	02
Figure 2.1	Algorithm Flowchart	04
Figure 2.2	Resistor	05
Figure 2.3	Zener Diode	07
Figure 2.4	Symbol of Capacitor	08
Figure 2.5	Modulated IR Transmitter	08
Figure 2.6	Modulated IR Receiver	09
Figure 2.7	IR Receiver	09
Figure 2.8	Symbol of Crystal Oscillator	10
Figure 2.9	Atmega8 L-8PU Microcontroller	11
Figure 2.10	Pin Configuration Diagram of Atmega8 L-8PU Microcontroller	12
Figure 2.11	Sony TV Remote	13
Figure 2.12	SIRC Protocol	14
Figure 2.13	USB Extension Cable	15
Figure 2.14	USB port	15
Figure 3.1	Circuit Diagram	20
Figure 3.2	Top Silk of PCB Layout	21
Figure 3.3	Bottom Copper of PCB Layout	21
Figure 3.4	PCB board with Components	22
Figure 3.5	PCB Board with Bottom copper	22
Figure 3.6	Project Board with components	23

List of Tables

Table no.	Title	Page
Table 2.1	List of Components Used	05
Table 2.2	Resistor Color Code	07
Table 2.3	SIRC Function	14
Table 2.4	USB Port Pin Configuration	15
Table 6.1	Device Specification	26
Table 8.1	Cost of practical work	27

LIST OF SYMBOLS AND COLLABORATION

RAM	Random Access Memory
ROM	Read Only Memory
I/O	Input Output
XTAL	Crystal
AM	Amplitude Modulation
SIRC	Serial Infra-Red Control
USB	Universal Serial Bus
IR	Infer-Red Receiver
LED	Light Emitting Diode
PCB	Printed Circuit Board
FB	Feedback
PC	Personal Computer

Chapter-01

OVERVIEW

1.1 : Introduction

The objective of our project is to develop a microcontroller based system to control computer mouse with an infrared remote. To accomplish the task we have used a Sony[®] TV remote, an Atmega8 microcontroller, a 38 KHz IR receiver with an external circuit. For connecting the system with PC we have chosen USB interfacing. The program code is developed in AVR Studio. Our circuit is very simple and easy to understand. We have tried to make the device user friendly as much as we can. It is compatible in almost all PC with operating system Microsoft[®] Windows XP, Windows Vista or Windows 7. We hope anyone will have fun using our device.

1.2 : Aim of the Project

The main goal of the project is to control mouse cursor with TV remote. This is done with the implementation of Sony remote on an Atmega8 L-8PU microcontroller. Here the IR receiver is connected to the microcontroller. The microcontroller is connected to the pc through USB port. When a certain key is pressed in the remote, it sends infrared signal through its IR transmitter to the IR receiver which is connected to the Atmega8 L-8PU microcontroller the received infrared signal is decoded by using the program written on the ROM of the microcontroller. Hence the operations of cursor are performed according to the key pressed.

1.3 : Components and Methodology

1.3.1 : Hardware Components

- Microcontrollers
- IR Transmitter
- IR Receiver
- USB Extension Cable
- Computer

1.3.2 : Overview Block Diagram

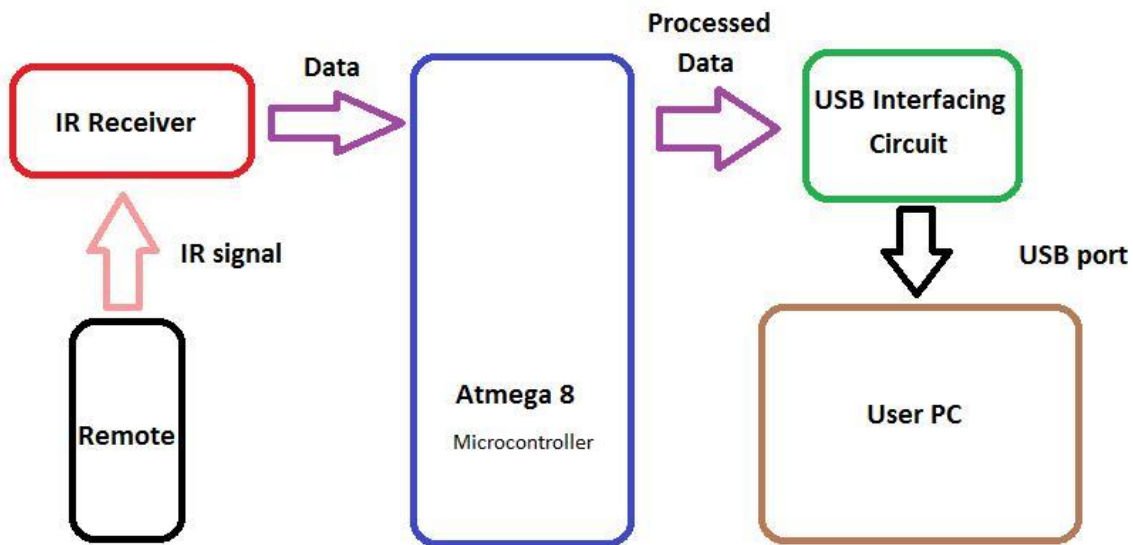


Figure 1.1 : Overview Block Diagram

The Figure 1.1 shows the block diagram of the controlling pc using TV remote. From this figure we know that in at first the data are transmitted through the remote. Then the receiver received the signal. After this data are decoded with the help of Atmega8 L-8PU microcontroller and the decoded value is sent through the pc via USB extension cable. Then according this the function is processed through the computer.

1.4 : Significance and Applications

The use of some wireless mouse that are having the disadvantages like occupying more space, more complex system, high power consumption, slow speed can be overcome by the use of this application.

1.5 : Organization of the Report

The chapters are arranged in the following manner

- Chapter 1 deals with introduction, aim, methodology, overview block diagram, significance & application of the project.
- Chapter 2 represents with project hardware.
- Chapter 3 contains the methodology used in our project. Embedded Systems and its applications, circuit diagram and operating principle of the project has also been included in this chapter.
- In chapter 4, we included results and discussions. Here specifications about our project, advantages, limitations, cost estimations and user guide has also been discussed.
- Chapter 5 deals with Conclusion.

Chapter-02

HARDWARE

2.1 : Algorithm Flowchart

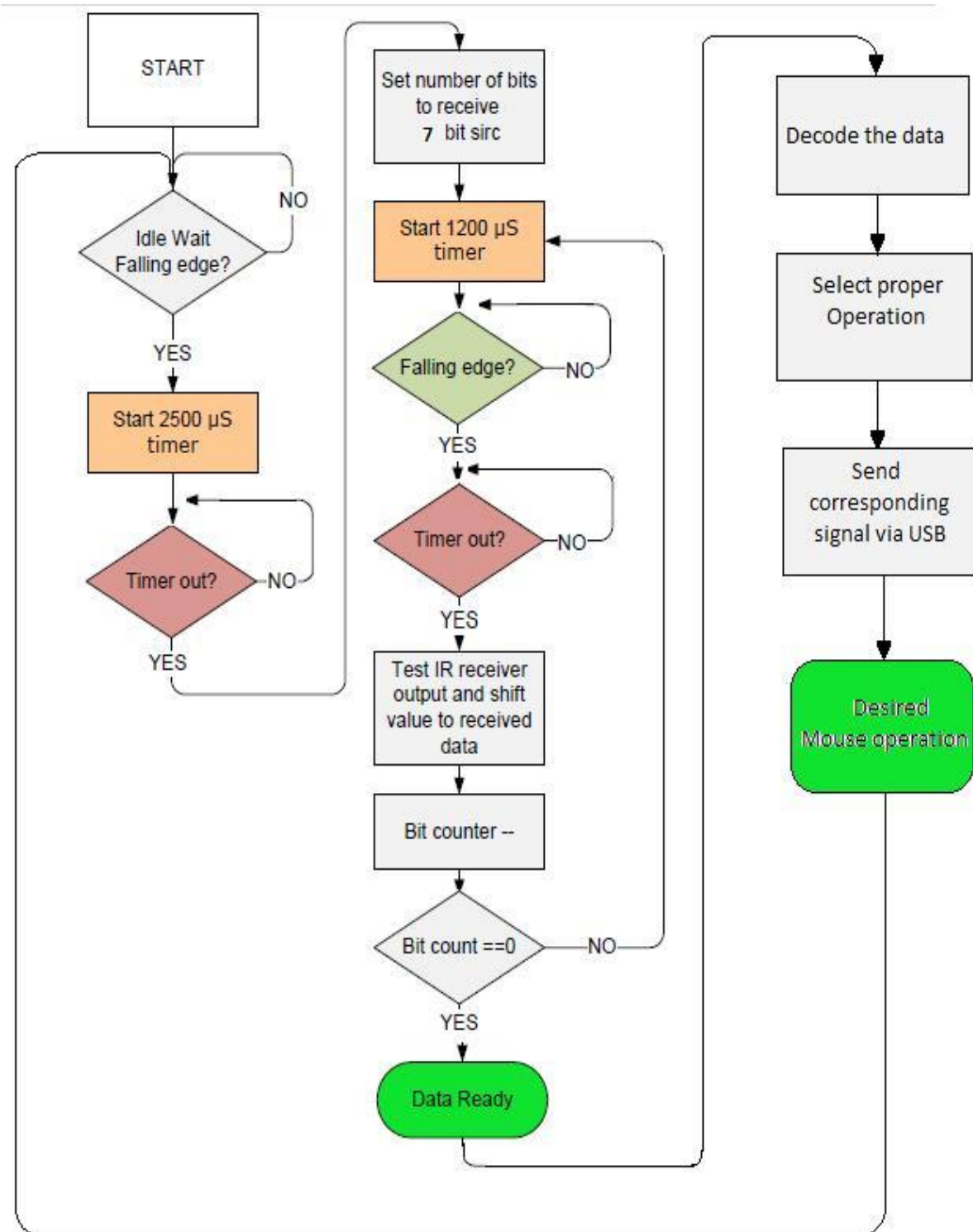


Figure 2.1 : Algorithm Flowchart

As from Sony remote, we send a 12-bit along digital signal modulated in 38 KHz with a starting bit. The IR receiver demodulates the received signal and sends Interrupt signal to the microcontroller. We set a timer (initial value is 0) to count the time of two adjacent 'high level' and 'low level' and count how many '1' and '0' are in the 7 command bits. From this counting, microcontroller recognizes the button. And in the main program, different mouse commands are assigned for the different buttons. Then data are sends to the user PC via USB. We've used HID mouse Protocol to generate different mouse signals. Here user PC can recognize the device by its pull up resistors at D+ and D- pins of USB port. On low and full speed devices, a differential '1' is transmitted by pulling D+ over 2.8V with a 68Ω ohm resistor pulled to ground and D- under 0.3V with a 68Ω ohm resistor pulled to 3.6V. A differential '0' on the other hand is a D- greater than 2.8V and a D+ less than 0.3V with the same appropriate pull down/up resistors.

2.2 : Used Components and Description

2.2.1 : List of Components Used

Name of the component	Number used
Atmega8 L-8PU	1
Sony TV remote	1
IR Receiver	1
12 MHz Crystal Osc.	1
4.7uF Capacitor	1
2.7pF Capacitor	2
LED	1
3.6 V Zener Diode	2
68 Ohm Resistors	3
1.5K Ohm Resistors	1
USB male port	1
USB extension cord	1
PCB Board	1

Table 2.1 : List of Components Used

2.2.1.1 : Description of the Circuit Components

2.2.1.1.1 : Resistor

Resistor is a component that resists the flow of direct or alternating electric circuit. Resistors can limit or divide the current, reduce the voltage, protect an electric circuit, or provide large amount of heat or light. An electric current is the movement of charged particles called electrons from one region to another. Resistors are usually placed in electric circuits.

Physicists explain the flow of current through a material, such as a resistor, by comparing it to water flowing through a pipe. Resistors are designed to have a specific value of resistance. Resistors used in electric circuits are cylindrical. They are often color coded by three or four color bands that indicate the specific value of resistance. Resistors obey the ohm's law, which states that the current density is directly proportional to the electric field when the temperature is constant.

How to identify the resistance value from color bands:

One should hold the resistor as shown below:

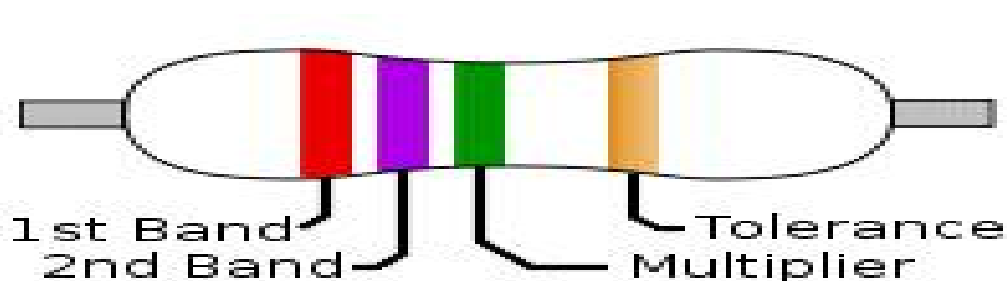


Figure 2.2 : Resistor

Three bands that are close together are to the left.

Then colors of;

Band no.1 signifies the 1st digit

Band no.2 signifies the 2nd digit

Band no.3 the multiplier.

Band no.4 the tolerance.

Band Color	Significant Digit	Multiplier	Tolerance
Black	0	1	
Brown	1	10	
Red	2	100	
Orange	3	1k	
Yellow	4	10k	
Green	5	100k	
Blue	6	1M	
Violet	7	10M	
Grey	8	-	
White	9	-	
Gold	-	0.1	5%
Silver	-	0.01	10%
No Band	-	-	20%

Table 2.2 : Resistor Color Code

2.2.1.1.2 : Zener Diode

In our project we used 3.6V zener diode (Zn). If the output voltage drops due to an overload, etc., the clamp voltage shifts from 3.6V to 8.0V. As a result, the CS pin voltage rises to 8.0V. The CS pin is also, at 3.6V by zener diode Zn. With Q off, the clamp is released and the 10uA constant-current source, forcing the CS pin voltage below 7.0V. During normal operation, the CS pin is clamped by a 3.6V zener , FA5305A FA5305A CS pin voltage (3.6V) DT voltage Oscillator output OUT pin output FB pin voltage.



Figure 2.3 : Zener Diode

2.2.1.1.3 : Capacitor

Capacitors store electric charge. They are used with resistors in the timing circuits because it takes time for a capacitor to fill with charge. They are used to smooth varying DC supplies by acting as a reservoir of charge. They are also used in filter circuits because capacitors easily pass AC signals but they block DC signals.

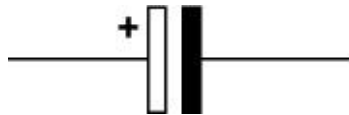


Figure 2.4 : Symbol of Capacitor

Capacitance is a charge of a capacitor's ability to store charge. A large capacitance means that more charge can be stored. Capacitance is measured in farads, symbol F. However 1F is very large, so prefixes are used to show the smaller values.

Three prefixes (multipliers) are used, μ (micro), n (nano) and p (pico):

- μ means 10^{-6} (millionth), so $1000000\mu\text{F} = 1\text{F}$
- n means 10^{-9} (thousand-millionth), so $1000\text{nF} = 1\mu\text{F}$
- p means 10^{-12} (million-millionth), so $1000\text{pF} = 1\text{F}$

2.2.1.1.4 : 555 Timer to Modulate Infrared (IR) Light

An IR emitter is going to be modulated using an astable 555 timer in this electronics exercise. The IR emitter needs to be modulated by a frequency of 38 kHz since the detector used in this exercise only detects 38 kHz modulated IR. The detector is set to only see 38 kHz modulated IR because there are random IR sources such as overhead lights, the sun, heaters, etc. in most environments that can cause interference if using un-modulated IR.

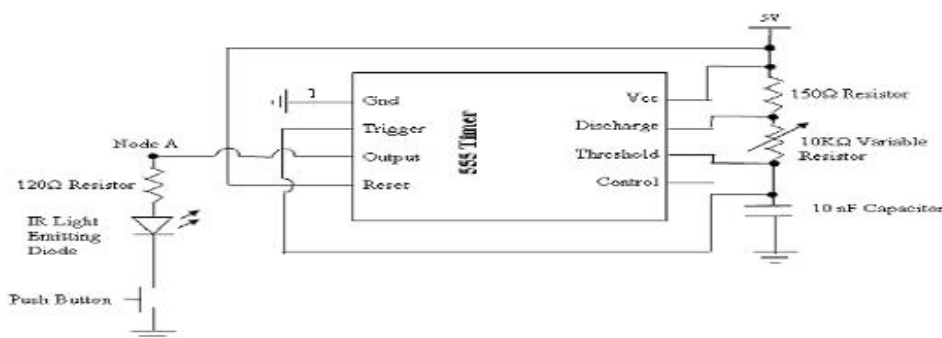


Figure 2.5 : Modulated IR Transmitter

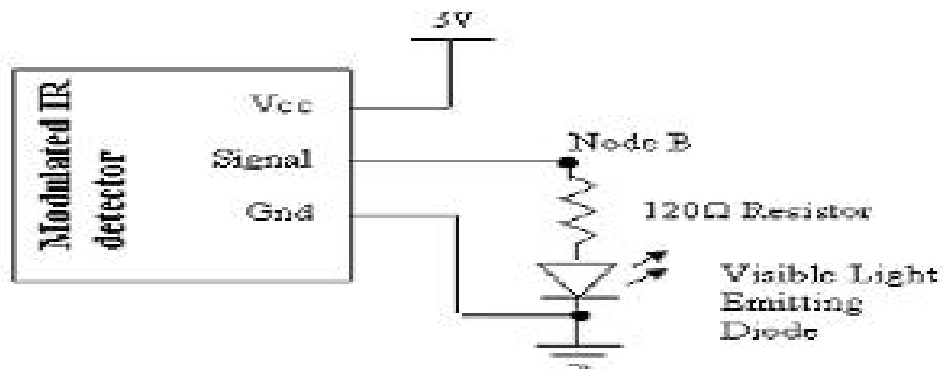


Figure 2.6: Modulated IR Receiver

Then applying power to the transmitter circuit. Using an oscilloscope to observe the signal at node A. Adjusting the 10kΩ variable resistor until the signal at node A is a 38 kHz series of pulses. Applying power to the receiver circuit Point the IR light emitting diode (LED) on the transmitter to the detector on the receiver. When the pushbutton is depressed the visible LED on the receiver should blink. If the visible led is blinking randomly, putting exposed 35 mm camera film around the IR detector.

2.2.1.1.5 : IR Receiver

Infrared receivers pick up infrared signals within line-of-sight, and within 30 feet or so, and turn the signal into electrical impulses. These electrical impulses can be carried around the home on wires, and then turned back into infrared signals by emitters. Due to their complexity and sensitivity, infrared receivers tend to be the most expensive part of an infrared distribution system.

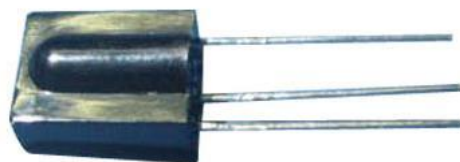


Figure 2.7 : IR Receiver

2.2.1.1.6 : Crystal Oscillator (XTAL – 12.00 MHz)

A crystal oscillator is an electronic circuit that uses the mechanical resonance of a vibrating crystal of piezoelectric material to create an electric signal with a very precise frequency. This frequency is commonly used to keep track of time, to provide a stable clock signals for digital integrated circuits, and to stabilize frequencies for radio transmitters and receivers. The most common type of piezoelectric resonator used is the quartz crystal, so oscillator circuits designed around them were called “crystal oscillator”. Quartz crystals are manufactured for frequencies from tens of megahertz.

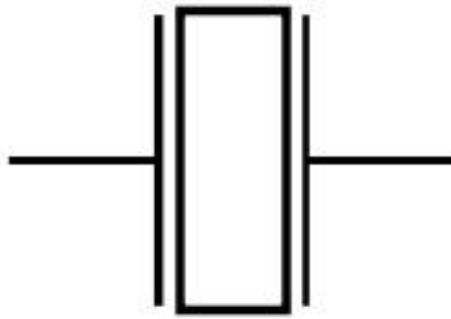


Figure 2.8 : Symbol of Crystal Oscillator

The crystal used in the circuit is AB-12.00 MHz-B2 often called Oscillators, which have mounting type is through hole. Its Oscillating Frequency is 12.00 MHz and its operating temperature is $\pm 20\text{ppm}$. It has operating mode fundamentally and its operating temperature is -20°C (minimum) to 70°C (maximum).

Crystal 1: Input to inverting oscillator amplifier and input to the internal clock generator circuits.

Crystal 2: Output from the inverting oscillator amplifier.

Oscillator characteristics: XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured of use an On-chip oscillator. To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected except that two PSEN activations are skipped during each access to external data memory.

2.2.1.1.7 : Atmega8 L-8PU Microcontroller

The ATmega8 is a low-power CMOS 8-bit microcontroller based on the AVR RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega8 achieves throughputs approaching 1 MIPS per MHz, allowing the system designer to optimize power consumption versus processing speed.

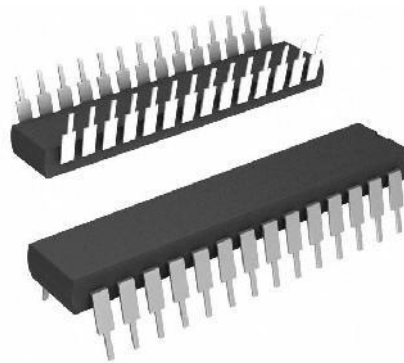


Figure : 2.9 : Atmega8 L-8PU Microcontroller

The ATmega8 provides the following features: 8 Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes of EEPROM, 1 Kbyte of SRAM, 23 general purpose I/O lines, 32 general purpose working resistors, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte oriented Two wire Serial Interface, a 6-channel ADC (eight channels in TQFP and QFN/MLF packages) with 10-bit accuracy, a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power down mode saves the resistor contents but freezes the Oscillator, disabling all other chip functions until the next Interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator

Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The Flash Program memory can be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash Section will continue to run while the Application Flash Section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega8 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

Pin Configuration :

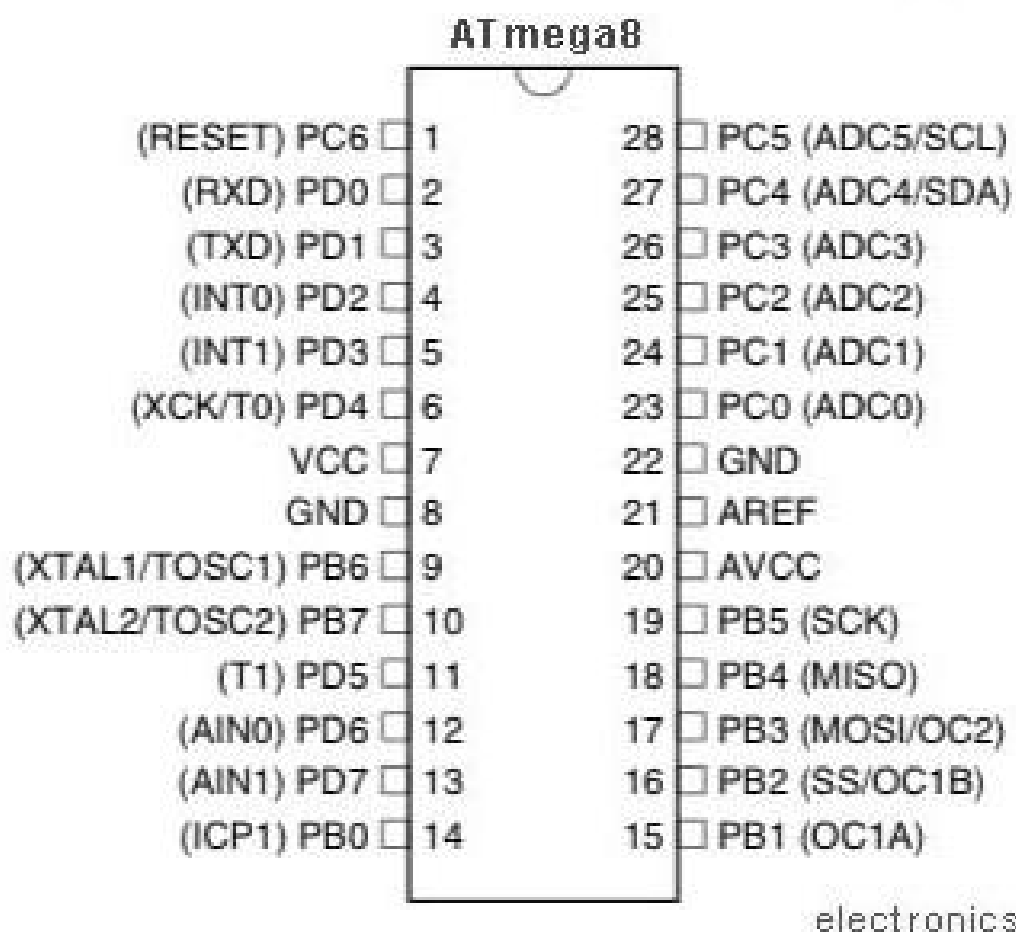


Figure 2.10 : Pin Configuration Diagram of Atmega8 L-8PU Microcontroller

2.3 : Circuit Operation

In this circuit mainly operation is divided into three different parts. The different parts are listed below.

- Transmitting data via remote
- Decoding data
- Decoder data transmit to the pc via USB port

2.3.1 : Transmitting Data via TV Remote

In this circuit, for transmitting data we are using Sony remote. Because Sony remote are working on the SIRC protocol. In this project we are working on the SIRC protocol.



Figure 2.11 : Sony TV Remote

Sony SIRC Protocol :

- ✓ 5-bit address and 7-bit command length (12-bit protocol)
- ✓ Pulse width modulation
- ✓ Carrier frequency of 40kHz
- ✓ Bit time of 1.2ms or 0.6ms

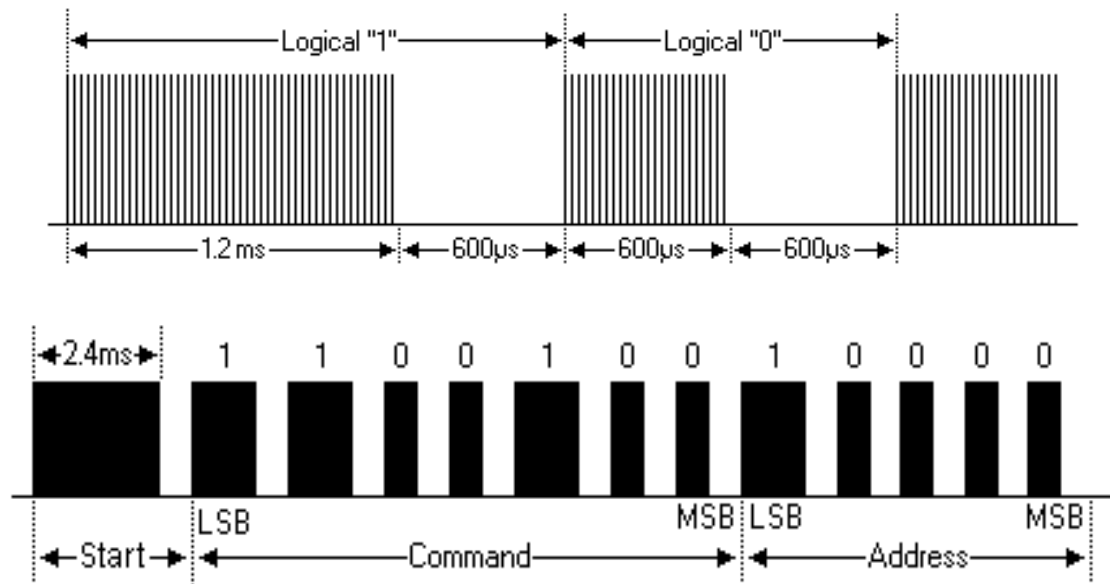


Figure 2.12 : SIRC Protocol

Command	Function
0	Digit key 1
1	Digit key 2
2	Digit key 3
3	Digit key 4
4	Digit key 5
5	Digit key 6
6	Digit key 7
7	Digit key 8
8	Digit key 9
9	Digit key 0
16	Channel +
17	Channel -
18	Volume +
19	Volume -

Table 2.3 : SIRC Function

2.3.2 : Decoding Data

With the help of TV remote we can send the data to the receiver circuit. The infrared sensor will take the data circuit is working on the 36 kHz frequency. So the decoding process is done through two different parts.

- Infrared Sensor
- Microcontroller AT89C2051
- Decode data transmitted via serial communication

2.3.3 : Decoder data transmit to the pc via USB port

The decoded data will be transmitted from the microcontroller and received by the USB port. The data will be transmitted to the PC via USB port. For this project we are using USB Extension Cable.



Figure 2.13 : USB Extension Cable

USB Port Pin Configuration :

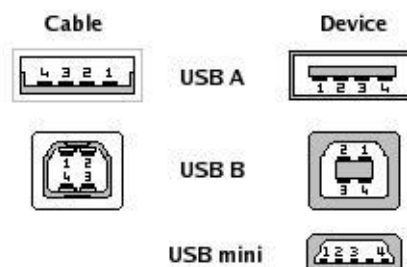


Figure 2.14 : USB port

Pin	Signal	Color	Description
1	VCC	Red	+5V
2	D-	White	Data -
3	D+	Green	Data +
4	GND	Black	Ground

Table 2.4 : USB Port Pin Configuration

Chapter-03

Methodology

3.1 : Introduction to Embedded System

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, sometimes with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, Handheld computers share some elements with embedded systems — such as the operating systems and microprocessors which power them — but are not truly embedded systems, because they allow different applications to be loaded and peripherals to be connected.

An embedded system is some combination of computer hardware and software, either fixed in capability or programmable, that is specifically designed for a particular kind of application device. Industrial machines, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines, and toys (as well as the more

obvious cellular phone and PDA) are among the myriad possible hosts of an embedded system. Embedded systems that are programmable are provided with a programming interface, and embedded systems programming is a specialized occupation.

Certain operating systems or language platforms are tailored for the embedded market, such as Embedded Java and Windows XP Embedded. However, some low-end consumer products use very inexpensive microprocessors and limited storage, with the application and operating system both part of a single program. The program is written permanently into the system's memory in this case, rather than being loaded into RAM (random access memory), as programs on a personal computer are.

3.2 : Applications of Embedded System

We are living in the Embedded World. We are surrounded with many embedded products and our daily life largely depends on the proper functioning of these gadgets. Television, Radio, CD player of our living room, Washing Machine or Microwave Oven in our kitchen, Card readers, Access Controllers, Palm devices of our work space enable us to do many of our tasks very effectively. Apart from all these, many controllers embedded in our car take care of car operations between the bumpers and most of the times we tend to ignore all these controllers.

In recent days, we are showered with variety of information about these embedded controllers in many places. All kinds of magazines and journals regularly dish out details about latest technologies, new devices; fast applications which make us believe that our basic survival is controlled by these embedded products. Now we can agree to the fact that these embedded products have successfully invaded into our world. We must be wondering about these embedded controllers or systems.

The computer we use to compose our mails, or create a document or analyze the database is known as the standard desktop computer. These desktop computers are manufactured to serve many purposes and applications.

We need to install the relevant software to get the required processing facility. So, these desktop computers can do many things. In contrast, embedded controllers carryout a specific work for which they are designed. Most of the time, engineers design these embedded controllers with a specific goal in mind. So these controllers cannot be used in any other place.

Theoretically, an embedded controller is a combination of a piece of microprocessor based hardware and the suitable software to undertake a specific task.

These days designers have many choices in microprocessors/microcontrollers. Especially, in 8 bit and 32 bit, the available variety really may overwhelm even an experienced designer. Selecting a right microprocessor may turn out as a most difficult first step and it is getting complicated as new devices continue to pop-up very often.

In the 8 bit segment, the most popular and used architecture is Intel's 8031. Market acceptance of this particular family has driven many semiconductor manufacturers to develop something new based on this particular architecture. Even after 25 years of existence, semiconductor manufacturers still come out with some kind of device using this 8031 core.

3.3 : Microcontrollers for Embedded Systems

In the Literature discussing microprocessors, we often see the term Embedded System. Microprocessors and Microcontrollers are widely used in embedded system products. An embedded system product uses a microprocessor (or Microcontroller) to do one task only. A printer is an example of embedded system since the processor inside it performs one task only; namely getting the data and printing it. Contrast this with a Pentium based PC. A PC can be used for any number of applications such as word processor, print-server, bank teller terminal, Video game, network server, or

Internet terminal. Software for a variety of applications can be loaded and run. Of course the reason a pc can perform myriad tasks is that it has RAM memory and an operating system that loads the application software into RAM memory and lets the CPU run it.

In an Embedded system, there is only one application software that is typically burned into ROM. An x86 PC contains or is connected to various embedded products such as keyboard, printer, modem, disk controller, sound card, CD-ROM drives, mouse, and so on. Each one of these peripherals has a Microcontroller inside it that performs only one task. For example, inside every mouse there is a Microcontroller to perform the task of finding the mouse position and sending it to the PC.

3.4 : Circuit Diagram

In this project, wireless mouse system is mainly based on the 8- bit micro controller. Here we are using the micro controller named as Atmega8 L-8PU. Crystal oscillator is connected to the 9th and 10th pin of the microcontroller. Crystal oscillator is connected with two 27pF capacitor. And two capacitor's common point connected with ground. In IR receiver vs point is connected to the USB vcc point. And IR receiver vo point is connected with 5th pin of the microcontroller. For USB port vcc point is connected with a 4.7 μ F capacitor then connection goes to 5th pin of microcontroller via zener diode and resistor R4. USB D+ point connected with 4th pin of microcontroller via zener diode and resistor R3. USB vcc point connected with two resistors R1 and R2. Resistor R1 goes to input of the zener diode D1 and another resistor R2 goes to USB D- point.

Mainly the project is used to move the cursor wirelessly using TV remote. IR receiver is connected to the micro controller though the port 3. IR receiver is connected to the PD3. IR transmitter is fixed in the remote. So pressing the key2 in the remote, cursor move in upper direction, pressing the key4 in the remote, cursor move in left direction, pressing the key6 in the remote, cursor move in right direction, pressing the key8 in the remote, cursor move in lower direction. The system is connected to the PC using USB port.

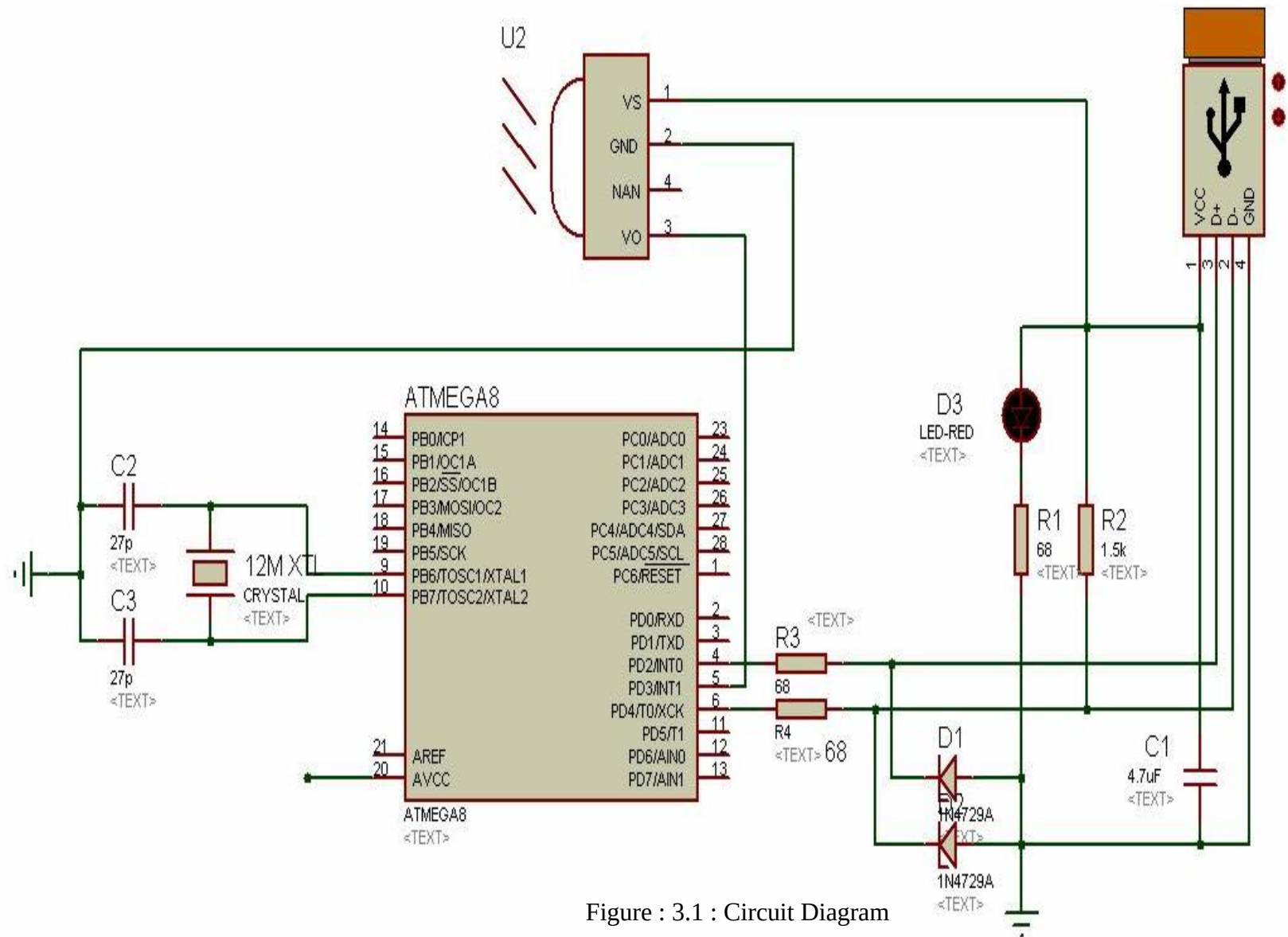


Figure : 3.1 : Circuit Diagram

3.5.1 : Top Silk

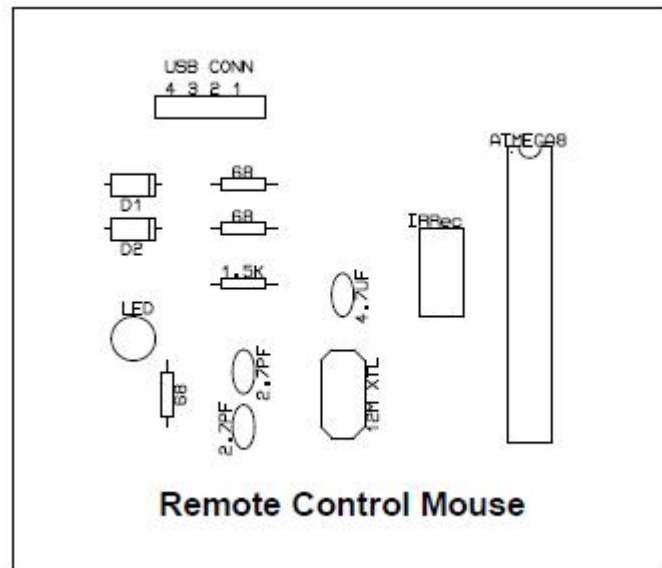


Figure 3.2 : Top Silk of PCB Layout

3.5.2 : Bottom Copper

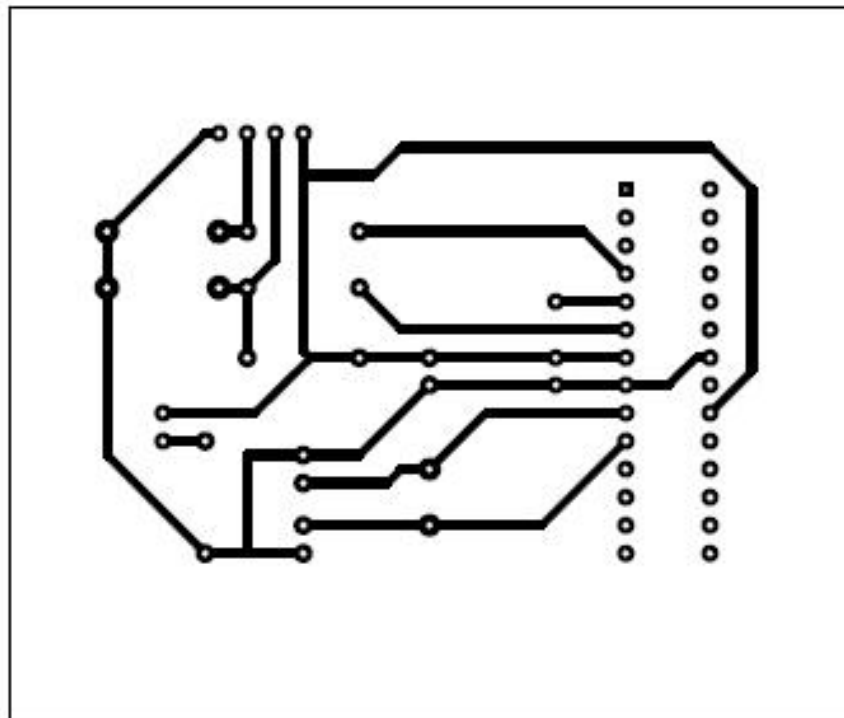


Figure 3.3 : Bottom Copper of PCB Layout

3.5.3 : PCB Board with Components



Figure 3.4 : PCB board with Components

3.5.4 : PCB Board with Bottom copper

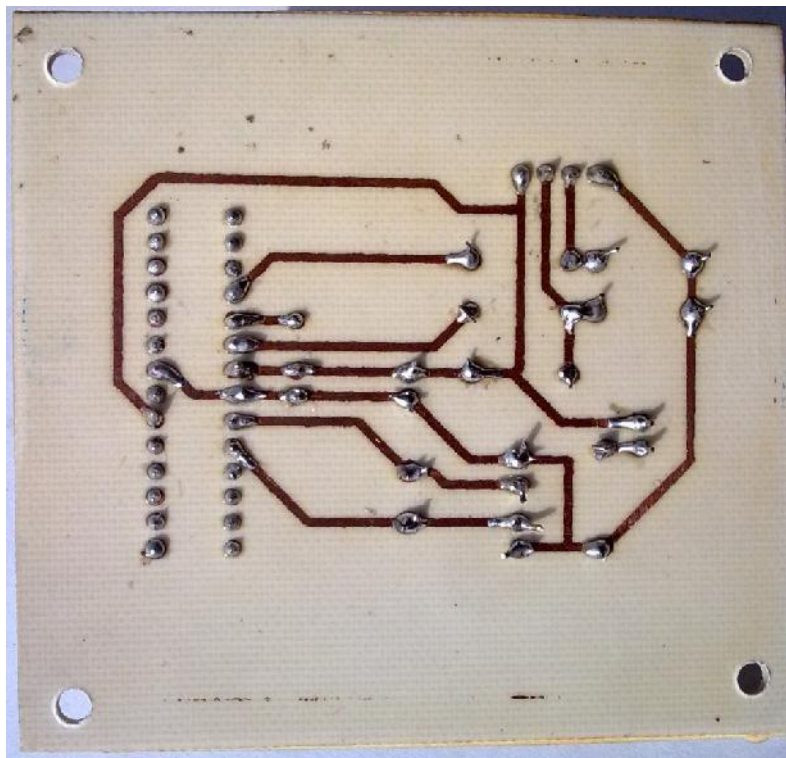


Figure 3.5 : PCB Board with Bottom copper

3.6 : Project Board with Components

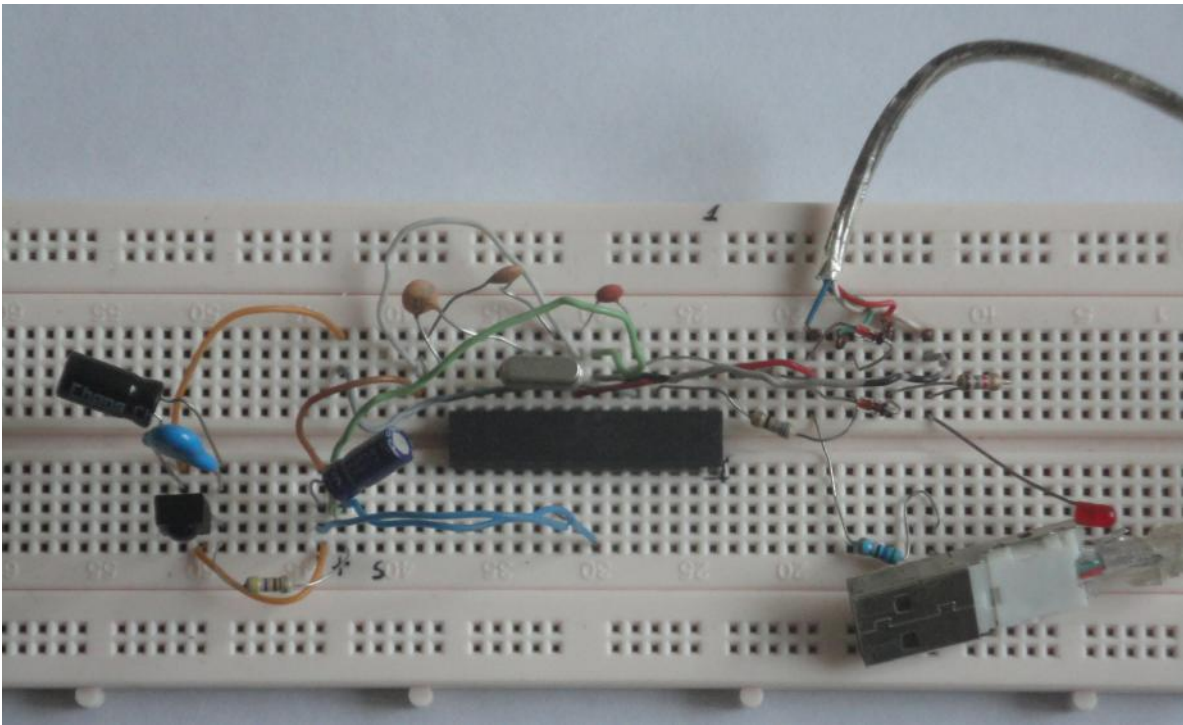


Figure 3.6 : Project Board with components

3.7 : Initialization

In this we are taking some variables for initialization. We are also initialization for the mode. In this we know about that as it uses in the mouse mode. We also initialize for the flip bit and toggle bit. We also initialize for the IR receiver pin.

3.8 : Generating Delay

In this project we are using SIRC protocol. In SIRC protocol we know that first two bit is automatic gain control bit and another one bit is toggle bit. So for that we have to generate the delay. After than five bits are the address bits. So first we have to wait for 4.572ms. After then we can generate the delay. For generating delay we are terminating the loop.

3.8.1 : Defined Mode

After generating delay we have to define the mode in which the computer will be operated like keyboard mode or mouse mode. For that we have to define the mode bit. If mode bit is equals to 1 then it will be operating in a mouse mode. That means TV remote will work like mouse. If mode bit equals to 0 then it will be operated in a keyboard which means TV remote will work like keyboard. If it is operated in the mouse mode then in the program will jump to the mouse loop. After then it will compare with the different addresses. If it is operated in the keyboard mode then in the program will jump to the keyboard loop. After then it will compare with the different addresses of the SIRC protocol.

3.8.2 : Compare with SIRC Protocol

SIRC protocol is used widely in the Sony TV remote. Now with the different inputs are given by the TV remote. So that all different inputs have different inputs have different hexadecimal addresses. Now given by these different addresses are compared with the SIRC protocol addresses value. If the given addresses are compared with SIRC protocol addresses then it will jump to another loop. Now it will be compared with the another address. If the addresses match then it will perform the function and it will return to the main loop. This process will perform until the address will be compared.

3.8.3 : Data Transmission

After comparison of the addresses pc will select the addresses which are given as an input. Now this will be transmitted to the pc remote control software via serial port. For this we use RS232 port. Now bit will be transmitted to the pc serial communication. Now transmitted bit will be performed according to the pc remote software. In this manner the whole software were designed. These are the basics steps of the flow designed of the software.

Chapter-04

Results & Discussion

4.1 : Operating Manual

This is the user manual to operate the hardware portion of the PC remote control project. First of all we have to check the hardware layout that the pin connections are as given in schematic diagram and also no schematic diagram and also no schematic connections were lost of layout. After then check that I.C. is perfectly burnt. If software is not perfectly loaded in I.C. then the execution of the hardware is not possible.

If the output is not correct then check the connections of the circuit. If the connections are perfect then watch out for the component. Due to high temperature any components are damaged or not. For that component see the operating range in data sheet. If operating range is above the given range of the component.

User must follow the data sheets of components for avoiding limitations and damages occur in testing of circuit. All the data sheets are given in appendix.

The interfacing of microcontroller and P.C. is necessary for controlling P.C. using TV remote. For interfacing of microcontroller and P.C. we must have to load the assembly language code into the I.C. With the help of this code we can connect the TV remote to the P.C. and we can perform the function. Without loading this assembly language code into the I.C. we cannot perform the function.

4.2 : Specifications of the Project

- Range : 2.7-6V
- Frequency : 0-24 MHz
- Modulation : AM
- Data Rate : 100kbps

4.3 : Device Specification

Remote Button	Mouse Operation
Volume +	Move right
Volume -	Move left
Channel+	Move up
Channel -	Move down
Key 4	Left Click
Key 6	Right Click
Key 5	Scroll up
Key 8	Scroll down
Key 1	Move up left
Key 3	Move up right
Key 7	Move down left
Key 9	Move down right
Key 2	Click and drag
Key 0	Double click

Table 4.1 : Device Specification

4.4 : Advantages

- a. Low cost facility then any other mouse.
- b. Easy to operate.
- c. Two in one facility (TV, PC mouse)
- d. Higher range facility than any other wireless mouse.
- e. Multi function operation in PC.
- f. Easy to construct and use.
- g. Use internal power supply.

4.5 : Limitations

- Our device is only compatible in Windows XP, Vista and Windows 7.
- As we used USB instead of PS2 port, so our mouse is little slow.
- The circuit is very much dependent on the sensitivity of IR receiver.
- The TV remote available in market is not good enough, so sometimes it may be needed to press a button repeatedly.
- Also some buttons in our remote don't work properly.

4.6 : Cost of Practical Work

Name of the component	Number used	Price (Tk)
Atmega8 L-8PU	1	200
Sony TV remote	1	150
IR Receiver	1	70
12 MHz Crystal Osc.	1	70
4.7uF Capacitor	1	2
2.7pF Capacitor	2	2 x 0.5
LED	1	2
3.6 V Zener Diode	2	2 x 1
68 Ohm Resistors	3	3 x 0.25
1.5K Ohm Resistors	1	0.25
USB male port	1	30
USB extension cord	1	120
PCB Board	1	150
Total		796

Table 4.1 : Cost of practical work

Chapter-05

CONCLUSION

5.1 : Conclusion

Using this project, there need not be any wire interface between the PC and mouse. Here we can control the PC using TV remote. The project is mainly based on the SIRC protocol using IR sensors.

From this project we can perform various computer functions through TV remote. In this age we can operate many digital devices through the remote control. We have to use different types of remote control to operate the P.C. using the remote which we are using for the TV. TV remote is working or the mouse for P.C.

With the help of this project we can overcome many complexities in daily life. We don't need to learn different remote to operate different devices. Also, we can operate the computer from the distance. But this distance should be equal to the infrared range.

From this project we can conclude that with the help of the Atmega8 L-8PU microcontroller family we can make a universal remote. With the help of this remote we can control both TV and P.C.

5.2 : Future Work

- Just changing the code we can also interface the Keyboard using the same circuit and remote.
- The code can be transformed to work with any TV remote.
- Modifying the code other commands like “Shut down”, “Sleep”, “Volume up”, “Volume down”, etc. can be controlled by remote too.
- Instead of IR other signal like RF, Bluetooth[®] can be used modifying the receiver circuit.

REFERENCE

- [1] Mr. A.V. Deshmuk, “Microcontrollers (Theory & Applications)”, WTMH, 2005
- [2] Mr. Daniel W Lewis, “Fundamentals of Embedded Software.”
- [3] Sony SIRC infrared protocol, <http://picprojects.org.uk/>, February 2010
- [4] USB COMPLETE ,FORTH EDITION, JON AXELSON
- [5] DRIVER API –V-USB , <http://vusb.wikidot.com/driver-api>
- [6] Universal Serial Bus (USB) HID Usage Tables, 10/28/2004, Version 1.12
- [7] Universal Serial Bus (USB) Device Class Definition for Human Interface Devices (HID), Firmware Specification—6/27/01, Version 1.11
- [8] The PS/2 Mouse Interface, <http://www.Computer-Engineering.org>
- [9] USB in a nutshell , making sense of the USB standard
- [10] <http://www.dharmanitech.com/2009/01/ir-remote-controlled-car-pwm-motor>
- [11] <http://szczuka.eu/en/avr/irmouse>
- [12] <http://www.scribd.com>

APPENDIX

Main Code:

```
/****** MOUSE CONTROL with IR *****/
```

```
/Controller: ATmega8 (12MHz external Crystal)
```

```
/Compiler: AVRStudio
```

```
/Date: August 2012
```

```
/****** HEADER FILE main.c *****/
```

```
/*
```

This example should run on most AVR's with only little changes. No special hardware resources except INT1 are used. We may have to change usbconfig.h for different I/O pins for USB. Please note that USB D+ must be the INT0 pin, or at least be connected to INT0 as well.

We use VID/PID 0x046D/0xC00E which is taken from a Logitech mouse. Don't publish any hardware using these IDs! This is for demonstration only!

```
*/
```

```
#include <avr/io.h>
```

```
#include <avr/wdt.h>
```

```
#include <avr/interrupt.h>
```

```
#include <util/delay.h>
```

```
#include <avr/pgmspace.h> /* required by usbdrv.h */
```

```
#include "usbdrv.h"
```

```
#include "oddebug.h" /* This is also an example for using debug macros */
```

```
#define OSCILLATOR 12
```

```
#define dig_key_0 9 //for double click
```

```
#define dig_key_1 0 //for moving left-up
```

```
#define dig_key_2 1 //for drag and move
```

```
#define dig_key_3 2 //for moving right-up
```

```
#define dig_key_4 3 //for left click
```

```
#define dig_key_5 4 //for wheel up
```

```
#define dig_key_6 5 //for right click
```

```
#define dig_key_7 6 //for moving left-down
```

```

#define dig_key_8 7           //for wheel down
#define dig_key_9 8           //for moving right-down
#define channel_plus 16       //for moving up
#define channel_minus 17      //for moving down
#define volume_plus 18        //for moving right
#define volume_minus 19       //for moving left
//define mute 20              //for middle click

/*-----USB Interfacing -----*/
PROGMEM char usbHidReportDescriptor[52] = { /* USB report descriptor, size must match
usbconfig.h */
    0x05, 0x01,                // USAGE_PAGE (Generic Desktop)
    0x09, 0x02,                // USAGE (Mouse)
    0xa1, 0x01,                // COLLECTION (Application)
    0x09, 0x01,                //  USAGE (Pointer)
    0xA1, 0x00,                //  COLLECTION (Physical)
    0x05, 0x09,                //   USAGE_PAGE (Button)
    0x19, 0x01,                //   USAGE_MINIMUM
    0x29, 0x03,                //   USAGE_MAXIMUM
    0x15, 0x00,                //   LOGICAL_MINIMUM (0)
    0x25, 0x01,                //   LOGICAL_MAXIMUM (1)
    0x95, 0x03,                //   REPORT_COUNT (3)
    0x75, 0x01,                //   REPORT_SIZE (1)
    0x81, 0x02,                //   INPUT (Data,Var,Abs)
    0x95, 0x01,                //   REPORT_COUNT (1)
    0x75, 0x05,                //   REPORT_SIZE (5)
    0x81, 0x03,                //   INPUT (Const,Var,Abs)
    0x05, 0x01,                //  USAGE_PAGE (Generic Desktop)
    0x09, 0x30,                //  USAGE (X)
    0x09, 0x31,                //  USAGE (Y)
    0x09, 0x38,                //  USAGE (Wheel)
    0x15, 0x81,                //  LOGICAL_MINIMUM (-127)
    0x25, 0x7F,                //  LOGICAL_MAXIMUM (127)
    0x75, 0x08,                //  REPORT_SIZE (8)
    0x95, 0x03,                //  REPORT_COUNT (3)
    0x81, 0x06,                //  INPUT (Data,Var,Rel)
    0xC0,                      // END_COLLECTION
    0xC0,                      // END_COLLECTION
};

```

```

/* This is the same report descriptor as seen in a Logitech mouse. The data
 * described by this descriptor consists of 4 bytes:
 *   . . . . B2 B1 B0 .... one byte with mouse button states
 *   X7 X6 X5 X4 X3 X2 X1 X0 .... 8 bit signed relative coordinate x
 *   Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0 .... 8 bit signed relative coordinate y
 *   W7 W6 W5 W4 W3 W2 W1 W0 .... 8 bit signed relative coordinate wheel
 */
typedef struct{
    uchar  buttonMask;
    char   dx;
    char   dy;
    char   dWheel;
}report_t;

static report_t reportBuffer;

static uchar  idleRate;
unsigned char flag = 0;
unsigned char flag1=0;

void initialize(void);
ISR (INT1_vect);
unsigned char IR_reader(void);

usbMsgLen_t usbFunctionSetup(uchar data[8])
{
    usbRequest_t *rq = (void *)data;

    if((rq->bmRequestType & USBRQ_TYPE_MASK) == USBRQ_TYPE_CLASS)    {
        /* class request type */

        DBG1(0x50, &rq->bRequest, 1); /* debug output: print our request */

        if(rq->bRequest == USBRQ_HID_GET_REPORT){ /* wValue: ReportType (highbyte),
        ReportID (lowbyte) */
            /* we only have one report type, so don't look at wValue */

            usbMsgPtr = (void *)&reportBuffer;

```

```

        return sizeof(reportBuffer);
    }
    else if(rq->bRequest == USBRQ_HID_GET_IDLE)
    {
        usbMsgPtr = &idleRate;
        return 1;
    }
    else if(rq->bRequest == USBRQ_HID_SET_IDLE)
    {
        idleRate = rq->wValue.bytes[1];
    }
}
else
{
    /* no vendor specific requests implemented */
}
return 0;
}

/* -----Initialization----- */

void initialize(void)
{
    DDRD &= ~(0x08);

    GICR |= 0x80;
    MCUCR |= 0x08;//Enable Falling Edge

    TCCR0 =0x04;//Prescaler 256
    TCNT0 = 0;
    TIMSK &=~(0x01);
}

/* -----Interrupt Service Routine ----- */

ISR (INT1_vect)
{
    unsigned char count;
    TCNT0 = 0;

```

```

while(!(PIND&0x08));
count = TCNT0;
if(count<8*OSCILLATOR)
{
    _delay_ms(20*OSCILLATOR);
    TCNT0 = 0;
    flag1 = 0;
}
else
    flag1 = 1;
//PORTC |= 0x20;    //for IR indication
}

/* ----- IR decoder ----- */

unsigned char IR_reader(void)
{
    unsigned char i=0, command=0, count;

    while(i<7)
    {
        while(PIND&0x08);
        TCNT0 = 0;

        while(!(PIND&0x08));
        count = TCNT0;

        if(count>4*OSCILLATOR)
        {
            command = command | (1<<i);
        }
        else
        {
            command = command & ~(1<<i);
        }
        i++;
    }
}

```

```

        return command;
    }

void remote_control (void)
{

    char remote_command = 0;
    if(flag1 == 1)
    {
        remote_command = IR_reader();
    }
    if(remote_command == volume_plus) //for moving right
    {
        reportBuffer.dx = 10;
        reportBuffer.buttonMask &= ~(0x10);
        DBG1(0x03, 0, 0);
        remote_command = -1;
        reportBuffer.dx = 0;
    }
    else if(remote_command == volume_minus) //for moving left
    {
        reportBuffer.dx = -10;
        reportBuffer.buttonMask &= ~(0x10);
        DBG1(0x03, 0, 0);

        remote_command = -1;
        reportBuffer.dx = 0;
    }
    else if(remote_command == channel_plus) //for moving up
    {
        reportBuffer.dy = 10;
        reportBuffer.buttonMask &= ~(0x20);
        DBG1(0x03, 0, 0);
        remote_command = -1;
        reportBuffer.dy = 0;
    }
    else if(remote_command == channel_minus) //for moving down
    {

```

```

        reportBuffer.dy = -10;
        reportBuffer.buttonMask &= ~(0x20);
        DBG1(0x03, 0, 0);
remote_command = -1;
        reportBuffer.dy = 0;
    }
    else if(remote_command == dig_key_5) //for moving wheelup
    {
        reportBuffer.dWheel = 10;
        DBG1(0x03, 0, 0);
remote_command = -1;
        reportBuffer.dWheel = 0;
    }
    else if(remote_command == dig_key_8) //for moving wheeldwn
    {
        reportBuffer.dWheel = -10;
        DBG1(0x03, 0, 0);
remote_command = -1;
        reportBuffer.dWheel = 0;
    }
    else if(remote_command == dig_key_4) //for left click
    {
        reportBuffer.buttonMask |= 0x01;
        DBG1(0x03, 0, 0);
        remote_command = -1;
        reportBuffer.buttonMask &= ~(0x01);
    }
    else if(remote_command == dig_key_6) //for right click
    {
        reportBuffer.buttonMask |= 0x02;
        DBG1(0x03, 0, 0);

        remote_command = -1;
        reportBuffer.buttonMask &= ~(0x02);
    }
    else if(remote_command == dig_key_1) //for moving leftup
    {
        reportBuffer.dx = -10;
        reportBuffer.dy = -10;

```

```

        reportBuffer.buttonMask &= ~(0x30);
        DBG1(0x03, 0, 0);
remote_command = -1;
        reportBuffer.dx = 0;
        reportBuffer.dy = 0;
    }
    else if(remote_command == dig_key_3) //for moving rightup
    {
        reportBuffer.dx = 10; reportBuffer.dy
        = -10; reportBuffer.buttonMask &=
        ~(0x30);
        DBG1(0x03, 0, 0);
remote_command = -1;
        reportBuffer.dx = 0;
        reportBuffer.dy = 0;
    }
    else if(remote_command == dig_key_7) //for moving leftdwn
    {
        reportBuffer.dx = -10; reportBuffer.dy
        = 10; reportBuffer.buttonMask &=
        ~(0x30);
        DBG1(0x03, 0, 0);
remote_command = -1;
        reportBuffer.dx = 0;
        reportBuffer.dy = 0;
    }
    else if(remote_command == dig_key_9) //for moving rightdwn
    {
        reportBuffer.dx = 10; reportBuffer.dy
        = 10; reportBuffer.buttonMask &=
        ~(0x30);
        DBG1(0x03, 0, 0);
        remote_command = -1;
        reportBuffer.dx = 0;
        reportBuffer.dy = 0;
    }
    else if(remote_command == dig_key_2) //for drag & move
    {
        reportBuffer.buttonMask |= 0x01;

```

```

        DBG1(0x03, 0, 0);
    }
    else if(remote_command == dig_key_0) //for double click
    {
        reportBuffer.buttonMask |= 0x01;
        _delay_ms(25);
        reportBuffer.buttonMask |= 0x01;
        DBG1(0x03, 0, 0);
        remote_command = -1;
        reportBuffer.buttonMask &= ~(0x01);

    }
    /*
    else if(remote_command == mute) //for drag & move
    {
        reportBuffer.buttonMask |= 0x04;
        DBG1(0x03, 0, 0);
    }*/

}

/* ----- Main ----- */

int main(void)
{
    uchar i;

    wdt_enable(WDTO_1S);
    DBG1(0x00, 0, 0);    /* debug output: main starts */

    odDebugInit();
    usbInit();
    usbDeviceDisconnect(); /* enforce re-enumeration, do this while interrupts are disabled!
    */
    i = 0;
    while(--i){          /* fake USB disconnect for > 250 ms */
        wdt_reset();
        _delay_ms(1);
    }
}

```

```

}
usbDeviceConnect();      /*device is now connected*/
    initialize();
sei();                    /*enable interrupt*/
DBG1(0x01, 0, 0);        /* debug output: main loop starts */
for(;;){                  /* main infinite loop */
    DBG1(0x02, 0, 0);     /* debug output: main loop iterates */
    wdt_reset();
    usbPoll();            /* need for data transmission*/
    if(usbInterruptIsReady()){
        /* called after every poll of the interrupt endpoint */
        remote_control();
        usbSetInterrupt((void *)&reportBuffer, sizeof(reportBuffer));

    }
}
return 0;
}

/* ----- */

```

Automatic Vehicle Speed Control With Wireless In-Vehicle Road Sign Delivery System Using ARM 7

Gummarekula Sattibabu^{*} B.V.V.Satyanarayan, VV Satyanarayana Kona

Assistant Professor, Electronics & Communication Engineering Department, Regency Institute of Technology, Yanam, India.

ABSTRACT: Nowadays accidents are occurring frequently, causing demise of many people by making modest mistakes while driving (in school zone, hills area, and highways). But sometimes it may not be possible to view the signboards placed by the Highway Department to alert the drivers in such kind of places and there is a chance for accident. The advancement in the processor technology and microcontrollers has opened a new system designed to prevent the accidents caused due to negligence of drivers in seeing traffic signals alongside the road and other anomalies on the roads. So to intimate the driver about the zones and to automatically maintain the speed is accomplished by means RF technology. The main objective is to design a Electronic Display controller meant for vehicle's speed control and monitors the zones, which runs on an embedded system and can be custom designed to fit into a vehicle's dashboard to display information on the vehicle. This system if adopted by some state can effectively reduce the number of road accidents caused by speeding vehicles losing control of the vehicle at speed breakers or by driver's negligence towards traffic signals. This paper presents a new design to control the speed of the automobiles at remote places for fixed time. The project is composed of two separate units: Zone status transmitter unit and Electronic Display and Control unit. Once the road-sign signal is received from the zones, the vehicle's Electronic Display Controller Unit warns the driver, to reduce the speed according to the zone, it waits for driver's response and reduces the speed of vehicle automatically.

Keywords: Automobile, RF, embedded system.

1 INTRODUCTION

It is evident that road accidents are increasing day by day. Recent studies show that one third of the number of fatal accidents are associated with excessive speeds in places where sharp turnings and junctions exist, as well as changes in the roadway like the presence of road-work or unexpected obstacles. This is due to having to wait hours together in traffic jams, taking tortuous detours due to on-road works, trying to spot speed breakers, navigating blind turns, one-ways and so on. Forked roads, railway crossings, sudden reverse bends and steep ascents and descents are just few of the road oddities that one may encounter on the average drive. Such road oddities are indicated by road-signs. Mandatory road-signs enforce traffic laws; Cautionary road-signs are installed in hazardous areas to avert accidents. Informative road-signs provide directions, locations and other information that is potentially useful to drivers in that locality. However, most vehicle drivers miss road signs more often than not. It is difficult to keep an eye out for road signs when one should be focused on driving. Many Driver Assistance systems for speed control have been developed so as to prevent accidents. One of them is Cruise control system (CC) that is capable of maintaining pre-defined speed and its later evolution version Adaptive Cruise Control (ACC) which keeps the automobile at pre-defined safer distance from the preceding vehicle. But these systems fail to detect the curved roads where the speed of the vehicles have to be reduced to avoid the accidents. Later Curve Warning Systems (CWS) came into existence to detect the curved roads by using Global Positioning System (GPS) and the digital maps accessed from the Geographical Information Systems (GIS) to warn driver of approaching the curved road. But these maps need to be updated regularly and are not useful if there are unpredictable road diversions or accidents. Here we propose a dynamic model where the system controls the vehicle according to the data frame that is transmitted by the RF transmitter fixed to the nearby road signs. The data frame is received by the microcontroller in automobile which controls the speed of vehicle. This is a RFID-Based Intelligent vehicle speed controller system where passive RF transceivers are arranged in the road close to the position of real traffic

signals. This model can also be better utilized to improve the fuel efficiency by imposing the maximum speed limit on the automobiles at which the mileage will be more.

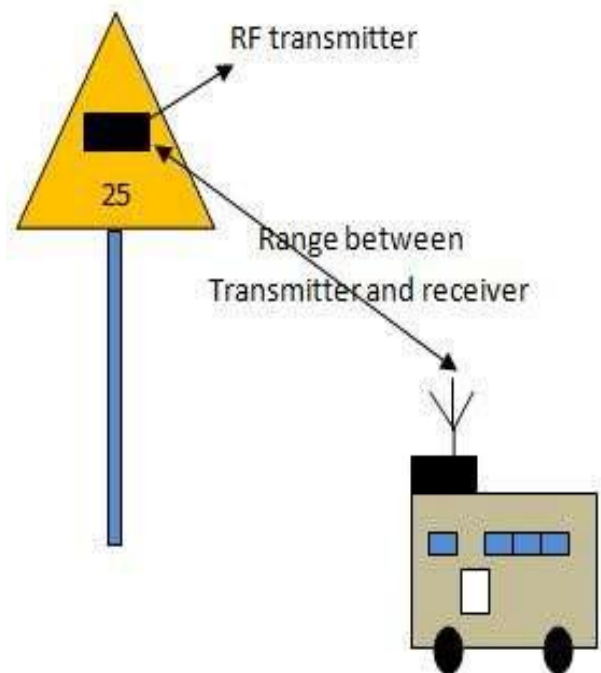


Figure 1: Traffic Signal posts equipped with RF Transmitter (left) Automobile equipped with the RF receiver (right)

2 BLOCK DIAGRAMS

Transmitter RF Tag:

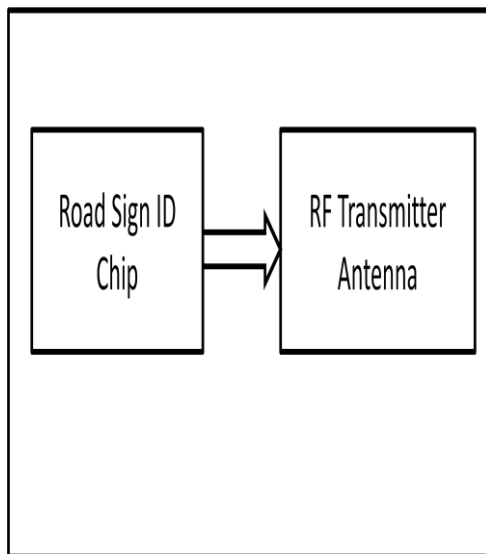


Figure 2.1: Passive RF Tag

Automobile Receiver Unit equipped with Display:

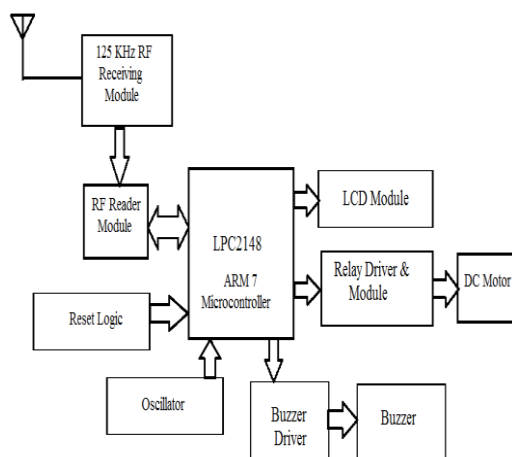


Figure 2.2: Electronic Controller and Display unit embedded in automobile

3 SYSTEM MODEL

The system model for Electronic Controller and Display Unit consists of following Hardware Modules:

3.1 Microcontroller Module:

LPC 2148 microcontroller is based on ARM7TDMI-S CPU with real-time emulation and embedded trace support. It combines microcontroller with embedded high speed flash memory and wide memory interface. Due to their low power consumption and tiny size, LPC 2148 are ideal for applications where miniaturization is a requirement, such as point-of-sale and access control. In this application the ARM 7 controller is the heart of operations of Electronic Controller and Display Unit. It consists of 2 ports for interfacing with various devices and has 40kb of RAM and 512kb of Flash memory and high speed performance at a speed of 60MHz.

3.2 Wireless Module:

3.2.1 RFID Reader: RFID reader connected to a wire antenna which demodulates the Manchester RF 32/64 bit signal and decodes it automatically. The data retrieved from the transponder is ready to be processed inside the device or to be sent over I2C/ UART / SPI or custom protocols. The user can control chip/module with an external device such as microcontroller, PC or handheld device with UART/I2C.

3.2.2 RF Tag:

RF identification is the wireless non-contact use of radio-frequency electromagnetic fields to transfer data, for identifying and tracking tags attached to objects. A Passive Clamp shell type tag is used for transmitting the road-sign which consists of memory chip and antenna.

3.3 Speed Control Module:

3.3.1 DC Motor:

DC motor is a mechanically commutated electric motor powered from direct current (DC). DC motors can operate directly from batteries which are rechargeable, providing the motive power for the first electric vehicles.

3.3.2 Relay Module:

Electromechanical relays are devices that join or break a circuit by physically moving electrical contacts into contact with each other.

3.4 Warning Module:

- LCD Module
- Buzzer

4. WORKING AND PROTOTYPE

The main objective is to replace road signs with **RFID** tags, and use in-vehicle RFID Reader-enabled modules to sense them, and provide useful information to the driver and design Electronic Display Controller meant for vehicle's speed control which is an embedded system. The RF tag can be placed on an existing road sign to transmit the information provided by signals placed on the road to adapt the vehicle's speed. Once the information is received from the RF tags, the vehicle's Electronic Display Controller automatically warns the driver, to reduce the speed according to the traffic sign indicated by the tag. It waits for few seconds for the driver's response to the information received, otherwise vehicle's EDC unit automatically reduces the speed. The Zone between two tags where the speed is controlled or reduced is called as Sensitive Zone. This process can be used not only to indicate Sensitive zones but also provide additional information to the drivers. Along with the primary objective of road safety, a plethora of other information can be provided to the commuter. Tags could disseminate additional information such as locations of nearby hospitals, fuel stations and food centres, by serving as Navigator. If there is road work or a construction in progress in a locality, installment of a tag a few km before the distressed area can be used to suggest suitable detours, thereby averting potential traffic jams and blockades. The possibilities are numerous. Care should be taken to provide the alerts on a priority basis. The entire operation of the proto-type is demonstrated in the flow chart given below:

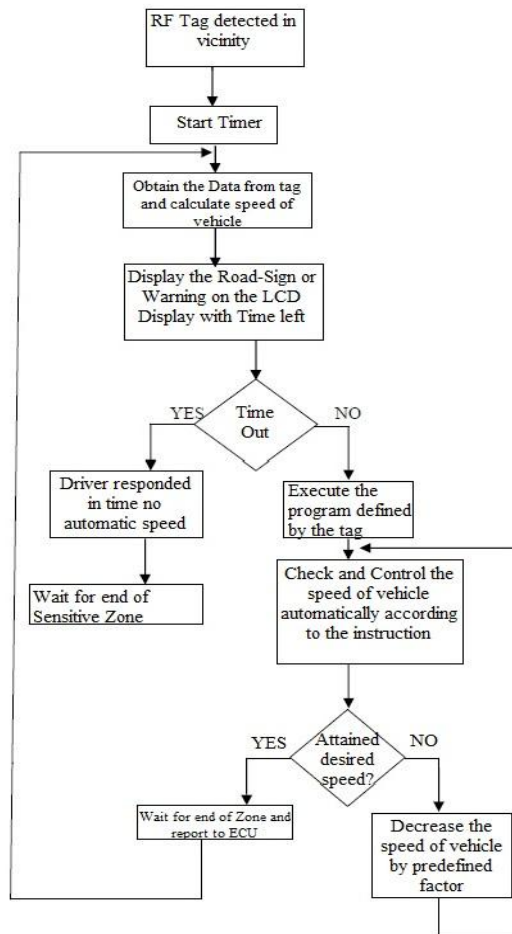
Flow chart for operation:

Figure 4.1: Demonstration of process through flow-chart

Different oddities would require notifications at different distances, which must be predetermined beforehand, to ensure that the driver is able to maneuver. Depending on the nature of the oddity, the optimal distance must be fixed at the time of tag placement. So far it has elucidated the concept with one-way traffic only. In the real scenario, however, it is imperative to consider traffic bound in both oncoming and outgoing directions, with respect to the tag. This poses a potential problem. What if the tag meant to be read on one side of the road is read by traffic bound in the opposite direction? For example, a tag meant to be read by a vehicle travelling in the A $\rightarrow$ B direction may possibly be read by one travelling from B $\rightarrow$ A on the same road. Using of better directional Algorithms in programming of the micro controller and using of Active RF tags solves this problem with this prototype.

5 CONCLUSION

In this paper the prototype design of a system that can deliver road signs to commuters' vehicles and can control the speed of the automobile has been demonstrated. This project is very simple which is durable and is of low cost. This project consumes less power. This system is easy to implement on present system which ensures maximum safety for drivers, passengers and pedestrians. The driver can get the information without any kind of distraction. This proto-type works even in bad weather conditions while the technology of artificial vision-based recognition of traffic signals might fail if visibility is poor and GPS Navigation

system may not work due to the sheer distance of satellites or weak signals. Radio Frequency signals might still be transmitted reliably through all the conditions. Driving safety will be enhanced and offers a positive cost differential to the government. We are trying to work with the all-pros and cons related to this project. In this prototype only one vehicle is considered. Practically other vehicles moving nearby can block or attenuate RF signals. This project may be enhanced further by establishing vehicle to vehicle microcontrollers' communication through Radio frequency so as to avoid vehicle collisions and to prevent deaths.

6. REFERENCES

- [1]. A design model for Automatic vehicle speed Control, International Journal for Computer Applications, volume-35, No.9, December 2011.
- [2]. Design and Prototype of an In-Vehicle Road Sign Delivery System using RFID, 2012 12th International Conference on ITS Telecommunications.
- [3]. Design of RF based Speed control system for vehicles, International Journal of Advanced Research in Computer and Communication Engineering Vol. 1, Issue 8, October 2012.
- [4]. Vehicle Speed Control using R.F. Technology, 5704017, June 2006, SRM institute of Science and Technology.
- [5]. Jerry Banks, David Hanny, Manuel Panchano, Les G. Thompson – RFID Applied – 2007, John Wiley and sons.
- [6]. "Follmer, W. C." (1974). "Electronic Speed Control", SAE Tech. Paper # 740022.
- [7]. LPC2141/42/44/46/48 Data Sheet, Rev 0.1, 7 September 2005
- [8]. <http://youthworldcom.blogspot.in/p/automatic-vehicle-over-speed.html>
- [9]. <http://www.engineersgarage.com/electronic-circuits/dc-motor-control-circuit-wireless-rf>
- [10]. Gangadhar, S.; R N shetty Inst. Of Technol, An intelligent road traffic control system, IEEE conference publication Kharagpur (2010).

Control of Electrical Lights and Fans using TV Remote

Group No. D07

Anshu Jain (03d07005) <anshuj@ee.iitb.ac.in>

Sandeep Gupta (03d07014) <sandeepg@ee.iitb.ac.in>

Durga Prasad (03d07033) <dprasad@ee.iitb.ac.in>

Supervisor: Prof. P.C.Pandey

Course Instructors: Prof. P.C.Pandey & Prof. V.K.Tandon

Abstract

The objective of this design project is to build a system for controlling light intensity of an electrical bulb or speed of fan using a TV remote or using serial port interface. The system provides regulation against input power supply. Intensity/speed of the load is controlled by changing the RMS voltage across the load using triac circuitry that is by changing the corresponding firing angle at the gate of the triac. An ADC circuit monitors the unregulated input voltage, followed by software to check for voltage variation and provide regulation.

1. Introduction

The main aim of this project is to design an electronic circuit which can control intensity of an electric bulb and speed of a fan using infrared TV remote or using serial port interface.

RMS voltage across a triac is related to the RMS voltage of the mains according to the following relation

$$V_{\text{rms}(\text{load})} = V_{\text{rms}(\text{mains})}(1 - \theta/\pi)^{0.5}$$

Here, θ is the firing angle of the triac measured from the zero crossing point. Thus by changing the firing angle we can regulate the voltage across the load. Hence we can vary the intensity/speed of the light/fan. If θ is zero i.e. if triac is fired at the zero crossing, then maximum power is transferred to the load similarly if triac is fired at the peak of the cycle (i.e. $\theta = \pi/2$) then only half of the power is transferred to the load. Thus our main job was to generate these pulses according to the signal from the TV remote or serial port. We are also monitoring the attenuated unregulated DC input voltage using an ADC, so as to provide voltage regulation against the mains voltage. For regulation against the mains frequency we are changing the instance of firing of the triac.

2. Design approach: -

2.1 Block Diagram:

The block diagram (fig. 1) consists of:

- Power supply consisting of 230V/9V step down transformer followed by a bridge rectifier, a low pass filter and a 5V voltage regulator (7805).
- IR receiver for receiving the signals from TV remote.
- RS 232 level shifter for serial port interfacing.
- Two microcontrollers one for decoding the signals (40 pin) and other (20 pin) for generating the trigger pulse for triac.
- A zero crossing detector.
- ADC for voltage regulation.
- LCD for displaying the current status of the loads.
- Triac circuitry.

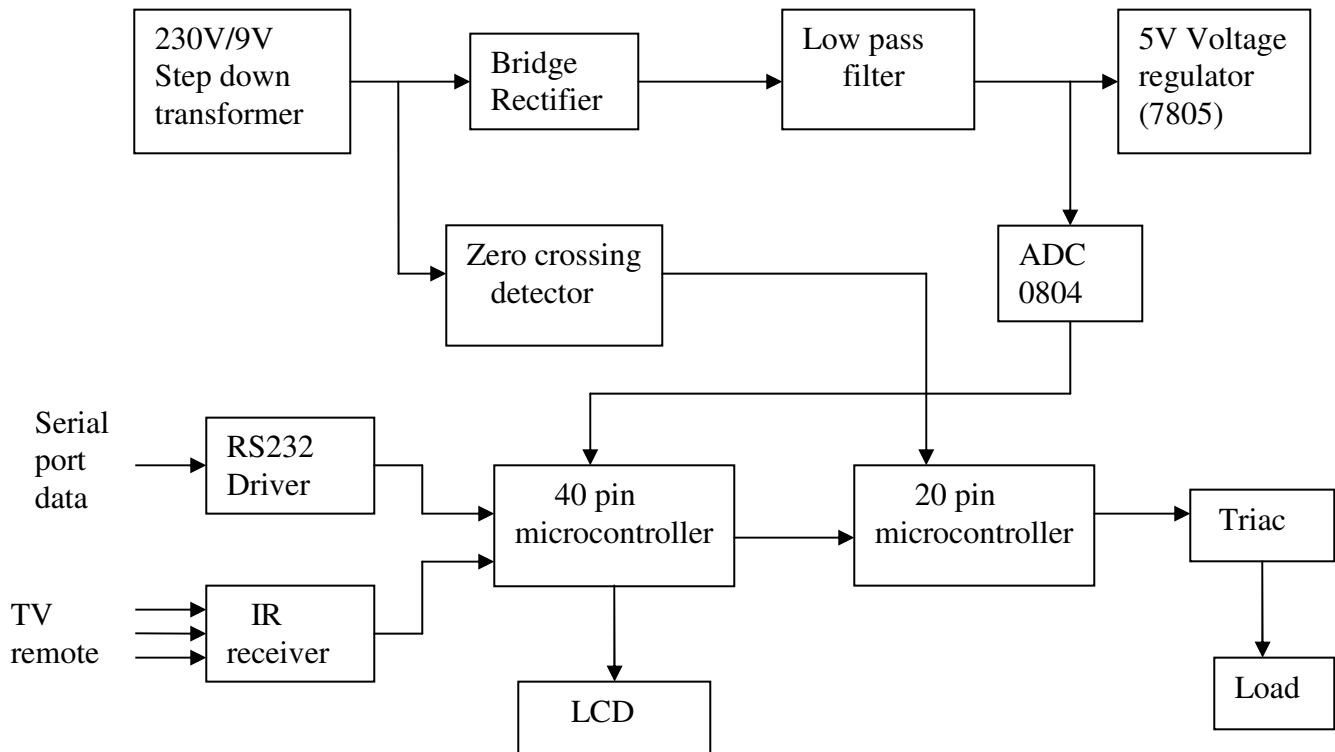


Fig. 1 Block diagram

2.2 Description of the various blocks:

a) 5V Power supply

Fig. 2 shows the circuit to generate 5V DC supply from mains supply. For this purpose we used 230V/9V step down transformer followed by a bridge rectifier consisting of IN4001 diodes, low pass filter and 5V voltage regulator LM7805 (IC1). Output from transformer is also used to detect zero crossing. Capacitor C1 is used to filter the output

from bridge rectifier to generate unregulated DC supply (V_x) which serves as input for ADC and voltage regulator. C2 and C3 are used for frequency stabilization. This 5V DC supply is used for power supply for various chips in the whole circuit.

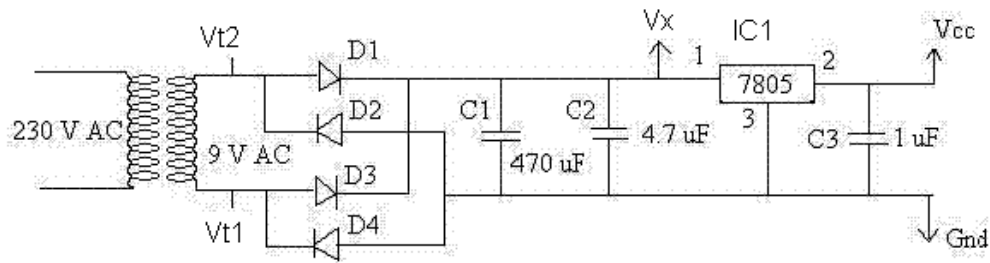


Fig. 2: DC power Supply Circuit

b) IR Receiver

We are using RC5 encoded remote. Every time a button is pressed at the remote control, it sends a train of 14 bits, 1.728ms per bit, the whole train is repeated every 130 ms if the button is kept pressed.

TSOP 1738 (IC2) is used as an infrared receiver. It's a 3 pin module with active low output. When no key is pressed on remote, it shows high output. Fig. 3 shows the pin diagram of TSOP. Output pin of TSOP is connected to pin 3.2 of 40 pin microcontroller AT89c42 (IC5).

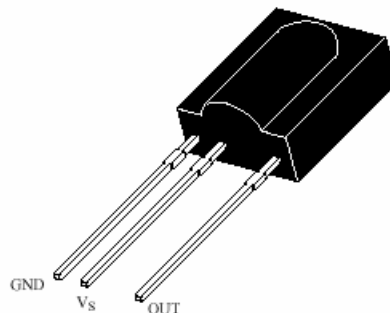


Fig. 3: IR receiver TSOP 1738 (IC2)

Fig. 4 shows the output signal from TSOP. This type of coding is called Manchester coding technically known as return to zero (RZ) format. In this coding each bit is divided into 2 half bits one as left bit and second one right bit. Bit value is determined by the transition at the center of the bit. If there is downward transition i.e. bit is changing from high to low at the center then bit value is one (Logic 1) and if there is upward transition at the center then value of that bit is zero (Logic 0).

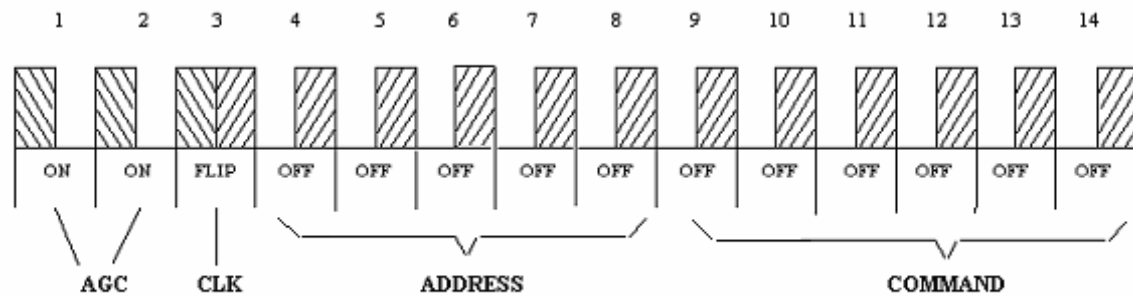


Fig. 4: Output from TSOP

The first two bits in the bit train, #1 and #2, are called Auto Gain Control (AGC) calibration. They are ON on the left side or takes transition from 1 to 0 at the center (Logic 1), and serve to calibrate the IR receiver's AGC. The bit #3 is the CHECK bit, every time a key is pressed at the remote (even repeatedly the same key) this bit flips state. The next 5 bits, #4 to #8, are used for SYSTEM ADDRESS, or to identify which kind of device should execute the COMMAND bits. The next 6 bits, #9 to #14, are used for COMMAND information to the device selected by the ADDRESS bits. Bit #14 is the LSB.

c) RS232 Driver

As we have to receive data so we are using only receiver part of RS232 driver. The level shifting of RS232 input for interfacing to the microcontroller has been carried out using IC3 (MAX232) as shown in Fig. 5. Pin 3 (TXD) of Male D9 connector is connected to pin 13 of MAX232 and pin 12 in connected to pin 3.0 of microcontroller 89c52.

d) IR Decoder

40-pin microcontroller AT89C52 is used to decode the IR signal from the TV remote and RS232. Here we are using a crystal of 11.0562 MHz so as to get a baud rate of 9600. The output of the TSOP is connected to pin P3.2 (external interrupt 0) and the serial data from Max232 is connected to pin P3.0 (serial interrupt). Whenever an interrupt comes at pin P3.2 the microcontroller first give a delay of 0.433ms so that we can read the mid point of any particular bit. Now as we don't need first 6 bits we skip the first 6 bits and read the next 8 bits by sampling the command bits (14 bit train) receive from the TV remote. Flow chart for decoding signal from TV remote is shown in Fig. 12.

This microcontroller is also being interrupted whenever computer sends a command over the serial port. The command consists of a train of 11 bits including the start bit and the stop bit with baud rate 9600. Microcontroller decodes this data by using the serial receiver (Rx) of the microcontroller.

After decoding the signal from the TV remote or from serial port this microcontroller also sends the instruction to the next microcontroller. This is done by setting the port 2 pins accordingly. First 3 bits P2.0, P2.1 and P2.2 denotes which load

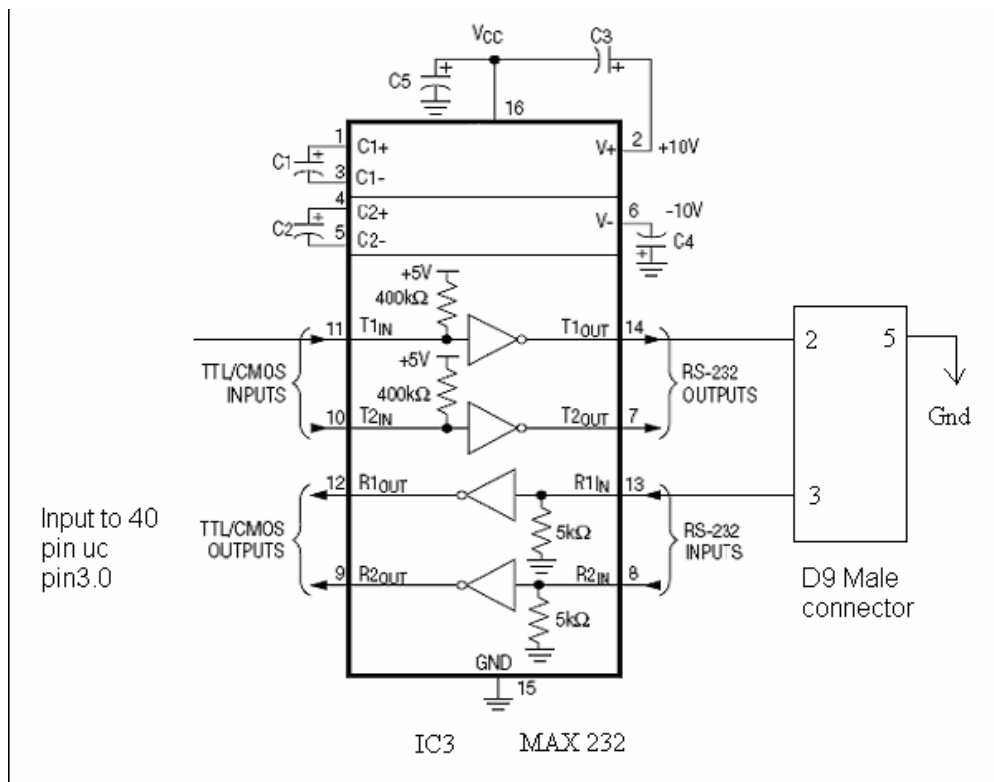


Fig. 5: RS232 Driver

to be executed. In our case we are using only 2 loads thus we need only 1 bit (P2.2). P2.2 is set if load is fan and is reset if load is light. But we can control 8 loads using this scheme. Next two bits are for the instruction to be executed which is coded as follows:

	P2.3	P2.4
OFF	0	0
ON	0	1
Decrease	1	0
Increase	1	1

Pin P2.5 is used to send a signal to the next microcontroller to show that a new key is pressed. P2.5 is complemented whenever a key is pressed on a TV remote or whenever a new data comes at the serial port. Remaining two bit i.e. P2.6 and P2.7 are not used thus can be used for some other purpose.

e) Zero Crossing Detector

Fig. 7 is circuit for zero crossing. Zero crossing is used as a reference for measuring firing angle of the triac. Output from transformer serves as input for low power quad OPAMP LM324 (IC4) (Fig. 6).

Connection Diagrams

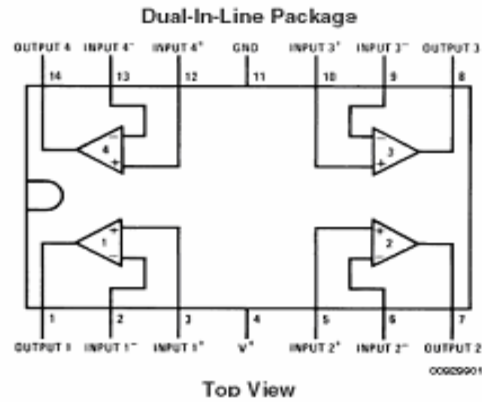


Fig. 6: Low power quad Opamp LM324 (IC4)

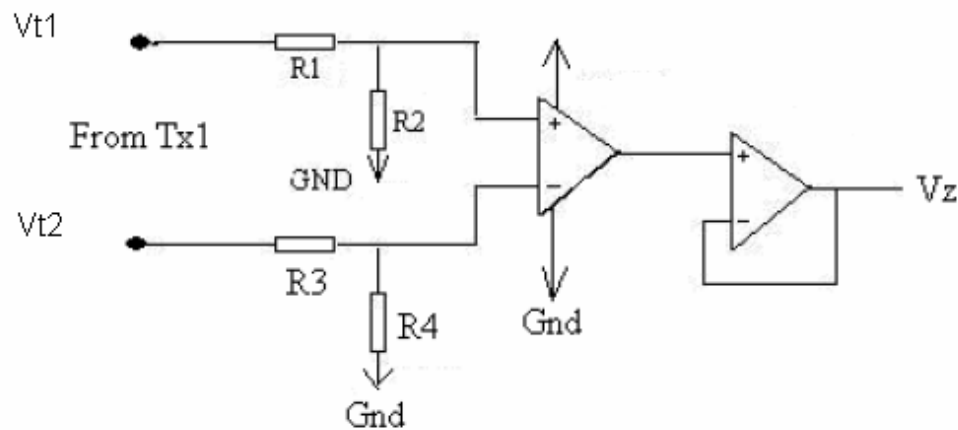


Fig. 7: Zero crossing detector circuit

LM 324 has four opamps. One of them was used as comparator with input as output from transformer followed by potential divider while 2nd opamp was used as a buffer which avoids comparator output getting loaded by microcontroller. Because the output of LM741 is bipolar and thus it can damage microcontroller so we used LM324.

Output from the buffer V_z is connected to pin 1.0 of 20 pin microcontroller AT89c2051(IC6).

$R_1 = R_2 = R_3 = R_4 = 1K \text{ ohm}$.

f) Voltage Regulation against mains power supply

For regulation against voltage supply we used ADC0804 which is one channel input ADC. It's a 20 pin chip with differential input voltage range 0V-5V. Unregulated DC voltage (V_x) followed by a potential divider is used as input for ADC (pin 6). A register R of 10K and capacitor C of 150uf are used to generate a clock frequency of 640

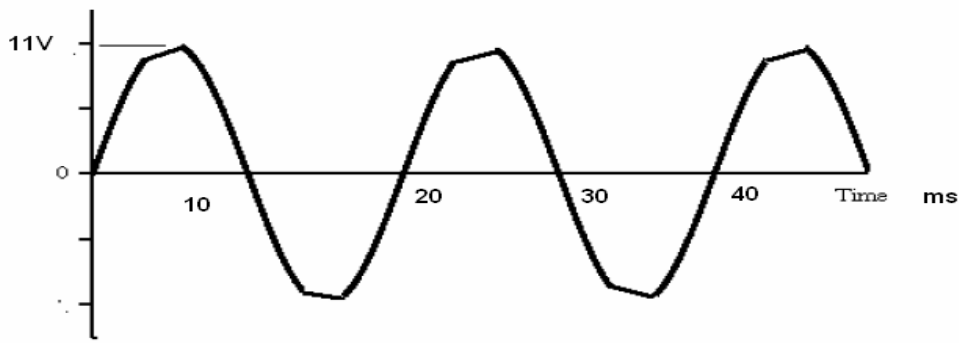


Fig 8 (a): Oupput waveform across transformer



Fig. 8 (b): Output waveforms of zero crossing detector

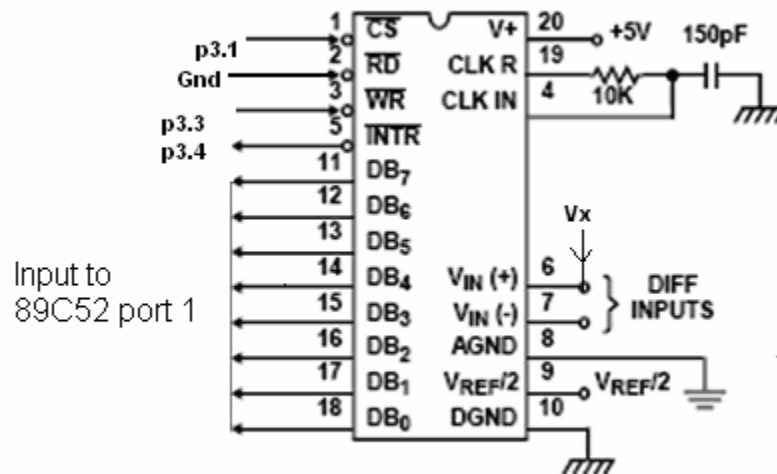


Fig. 9: ADC0804

KHz. Because V_x has magnitude about 10V so we used potential divider to make the input for ADC with in chip's input voltage range.

Voltage span of ADC is made 2.2V to 7.2 volts by providing constant 2.2V to the negative input pin of ADC. Here we are assuming voltage variation from 200V to 250V with 230V normal voltage. So our input voltage to ADC changes from 2.2V to 2.9V. We used a limiter circuit followed by a zener diode of 2.2V breakdown voltage for this purpose. Chip select low pin of ADC is connected to pin 3.1 of 89c52, write low to pin 3.4 and INTR low to pin 3.5. Output from ADC pin 11 to 18 are multiplexed with LCD and connected to pin p1.0 to p1.7 of the microcontroller 89c52. When ADC subroutine is

called in the program p3.1 is made active low. Thus ADC chip is selected. p3.4 is made active low for some time interval (about 10 us) and then made active high. ADC starts conversion after 1 to 8 clock periods and when conversion is completed it interrupts microcontroller. Interrupt pin is connected to p3.5 of 89c52. We are comparing output of ADC (pin 11 to pin 18) with output corresponding to 230V and changing the firing angle of triac accordingly.

g) Firing pulse generation

20 pin microcontroller AT89C2051 processes the instruction from the first microcontroller and generates the triac firing pulses accordingly. Output from the zero crossing detector is connected to pin p1.0 and is used to get the count for the half cycle. This count is then used to get the counts for the various levels. Zero crossing detector output is also used to synchronize the triac pulse with the input supply frequency. Fig. 13 shows the flowchart for its working.

h) Triac:

A triac is a three terminal semiconductor device that is triggered by a low-energy signal applied to its gate. The supply to the triac is an AC signal and it will always turn off when the applied voltage reaches zero at the end of the current half cycle. If a turn on pulse is applied, the triac goes from a high-impedance state to a conductive state, and it starts conducting current through the load. So the position at which this turn on pulse is fired (firing angle) directly control the percentage power of that half-cycle transmitted to the load.

$$P_{\text{load}} = P_{\text{main}} (1 - \theta/\pi)$$

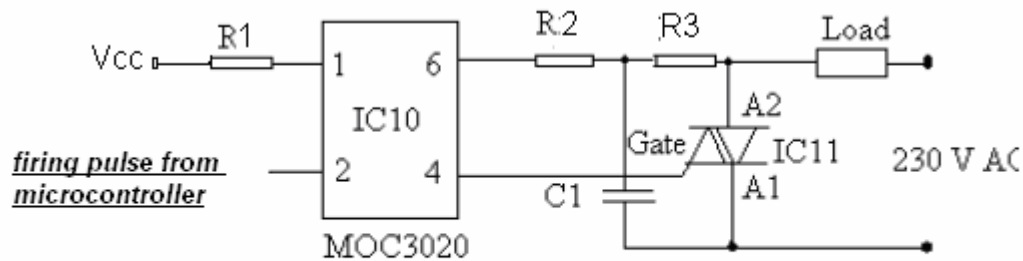


Fig. 10: Triac circuit

The pulses from the microcontroller are connected to the pin 1(the anode of the photodiode) of the opto-coupler MOC3020 and the cathode (pin 2) is connected to the ground. On the pulses, the photodiode radiates and drives the triac inside MOC3020. In this part the pulses are in optical form hence providing the optical isolation between the triac load circuit and the electronic circuit. The diac of the opto-coupler further drives the triac with the load connected between a power supply terminal and A2 terminal of the triac and the other power supply terminal and A1 are shorted.

R1 = 560 Ω; R2 = 250 Ω; R3 = 1k Ω; C1 = 0.22μF.

i) LCD Display:

The LCD display contains two internal byte wide registers, one for commands (RS=0) and the second for characters to be displayed (RS=1). Pins 1 and 3 are connected to the ground, and pins 7 to 14 are connected to Port 1 of the microcontroller AT89C52 to furnish the command or data byte, and pins 4,5 and 6 (register select, read/write and enable high signals respectively) are connected to 3.5,3.6 and 3.7. Pin 14 of LCD is monitored for logic high (busy) to ensure the display is not overwritten.

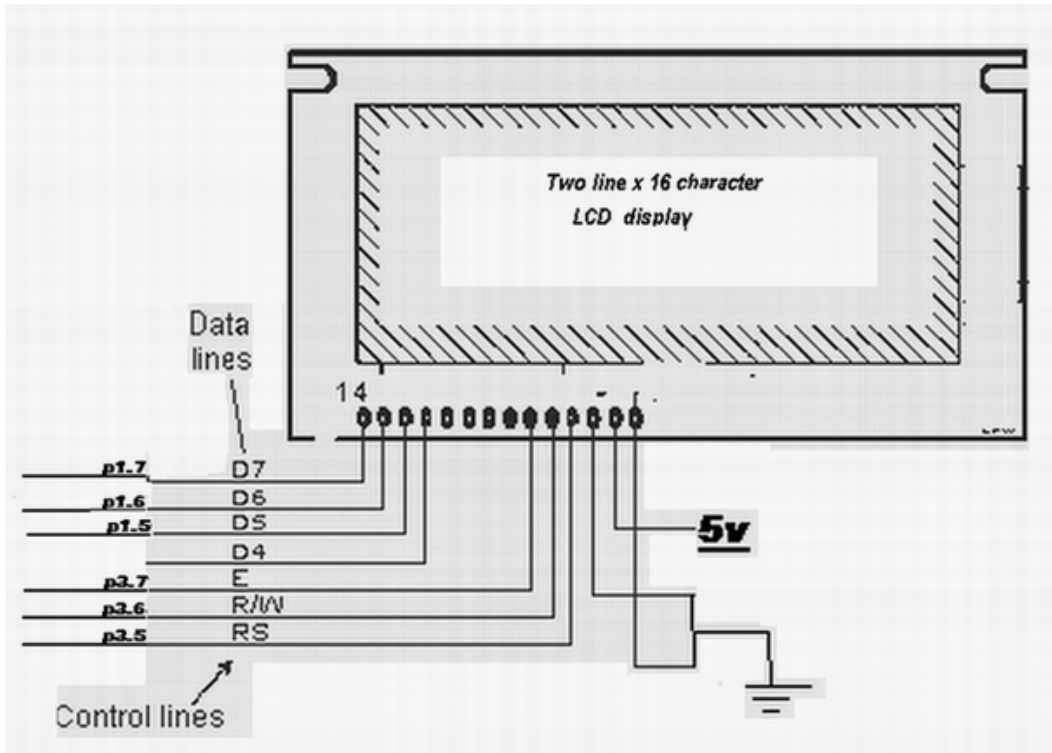


Fig. 11: LCD display

Each time a remote key is pressed it displays the respective levels of each load and holds till a new signal is received by an IR receiver.

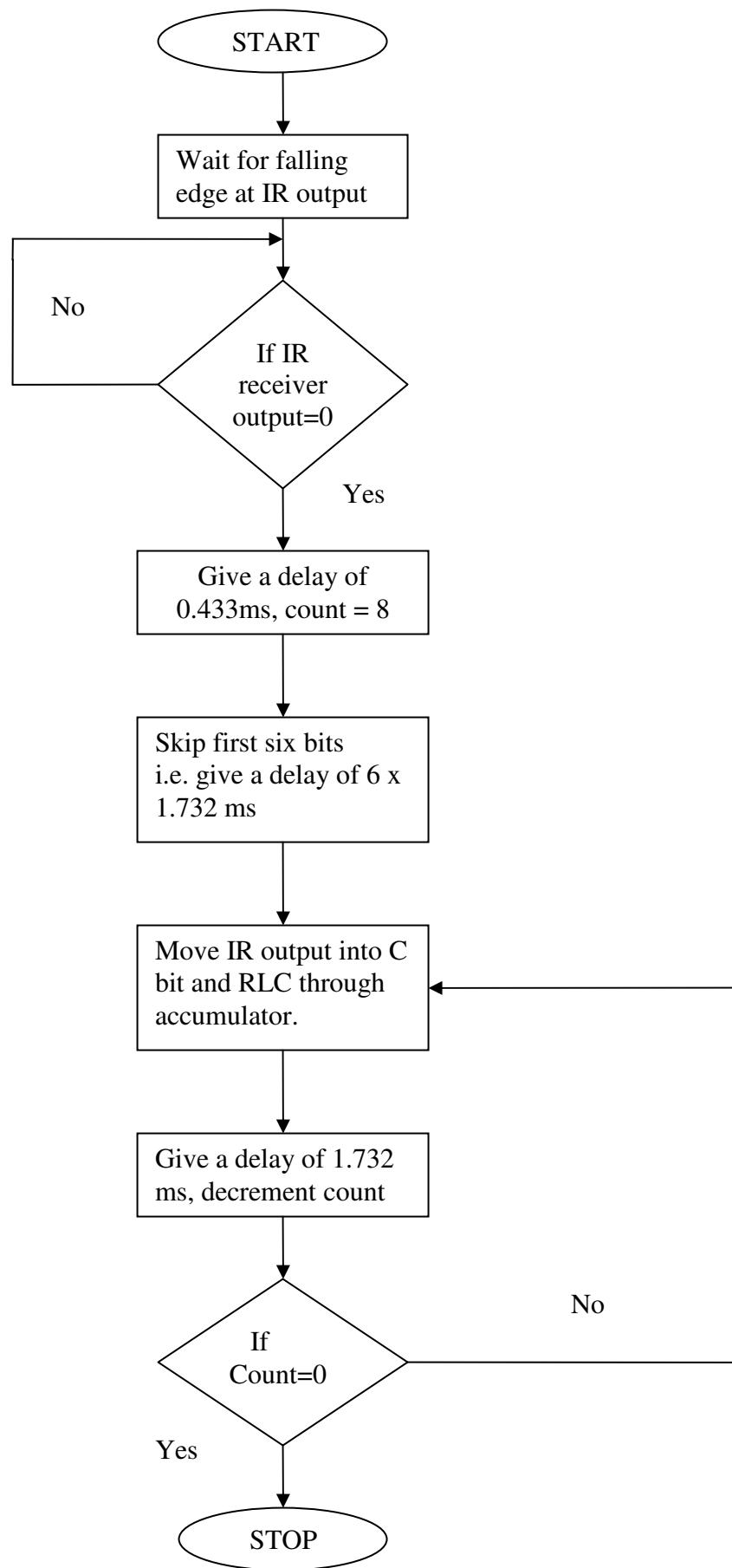


Fig. 12: Flowchart for IR decoding

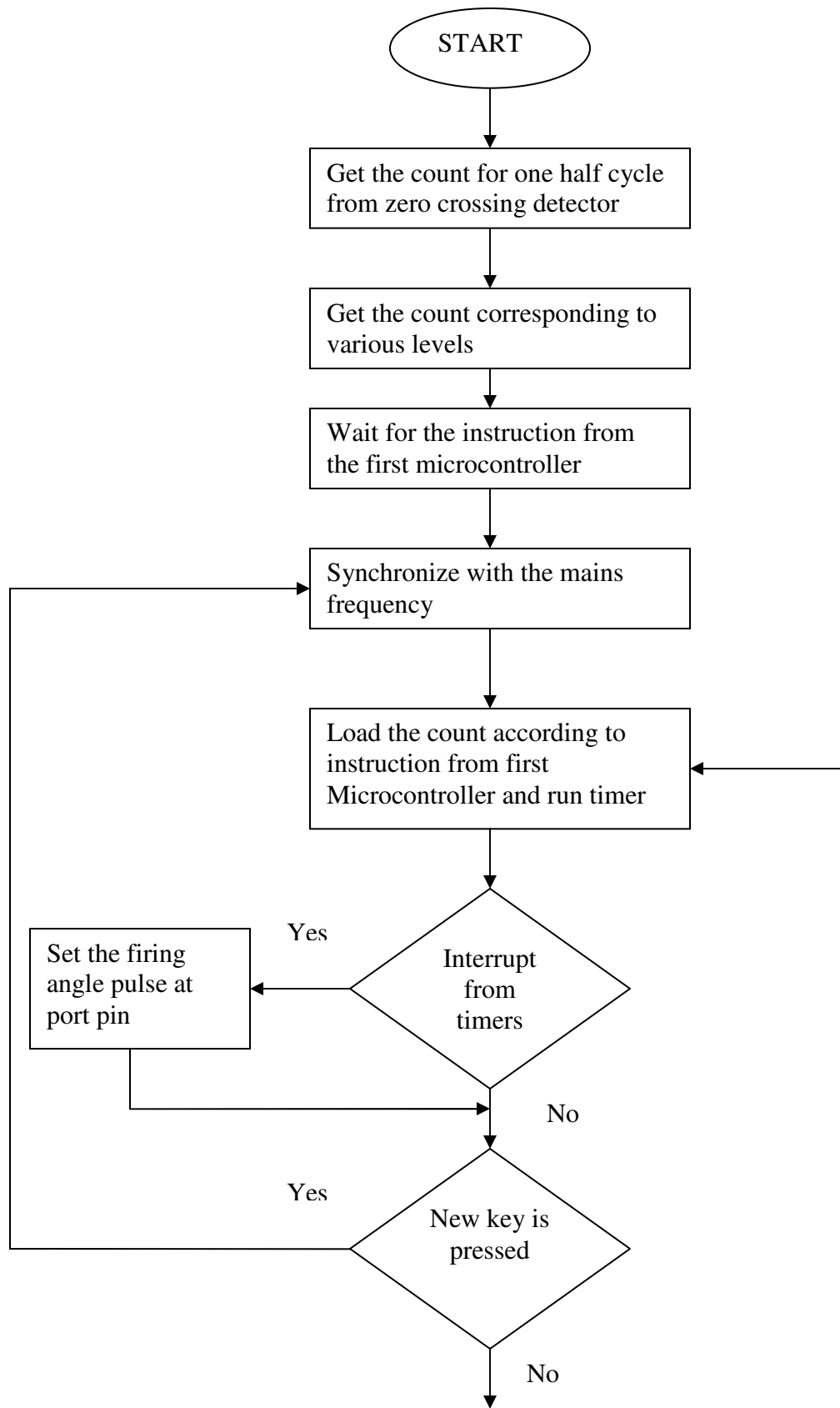


Fig. 13: Flowchart for the generation of triac pulses

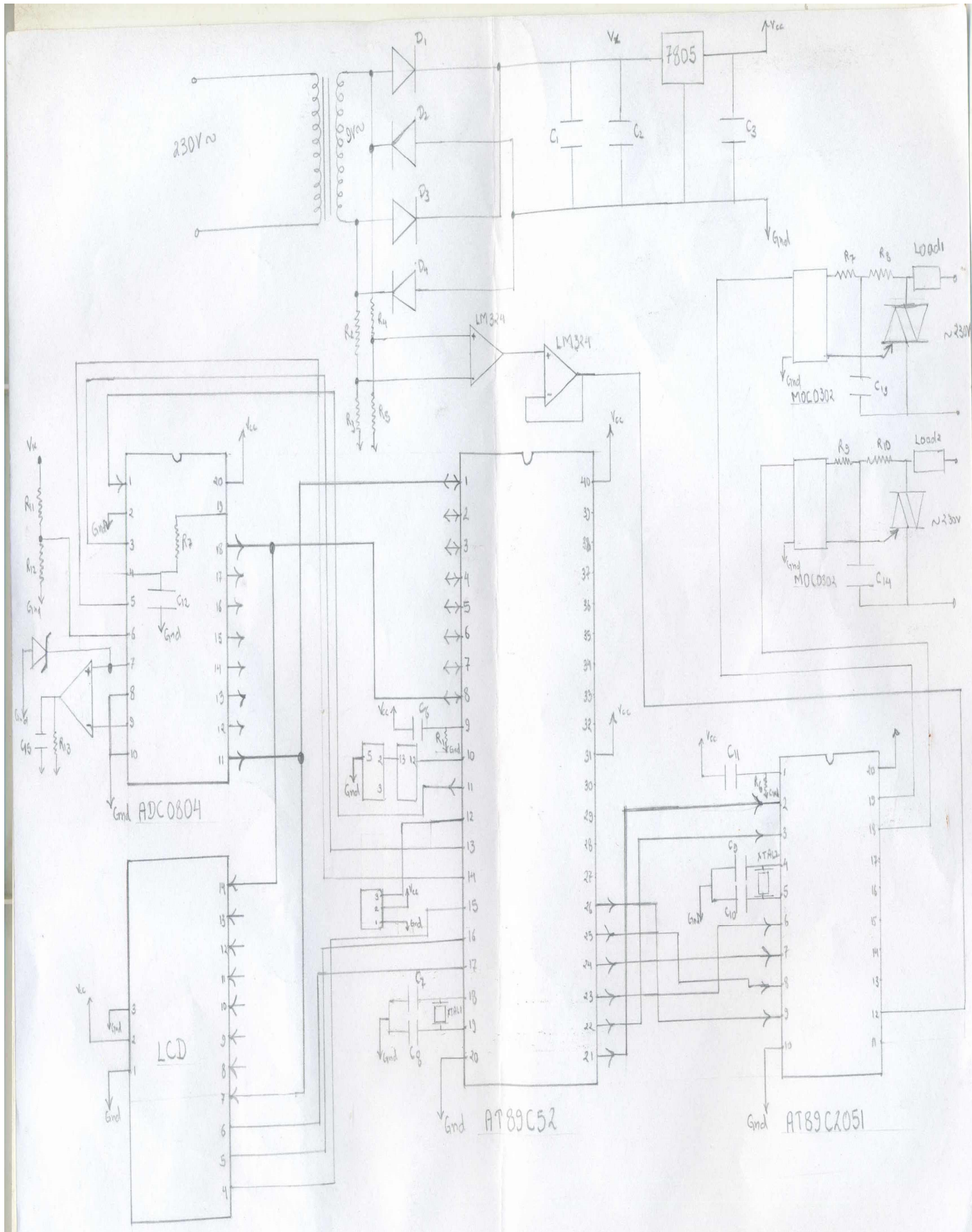


Fig. 14: Complete Circuit Diagram

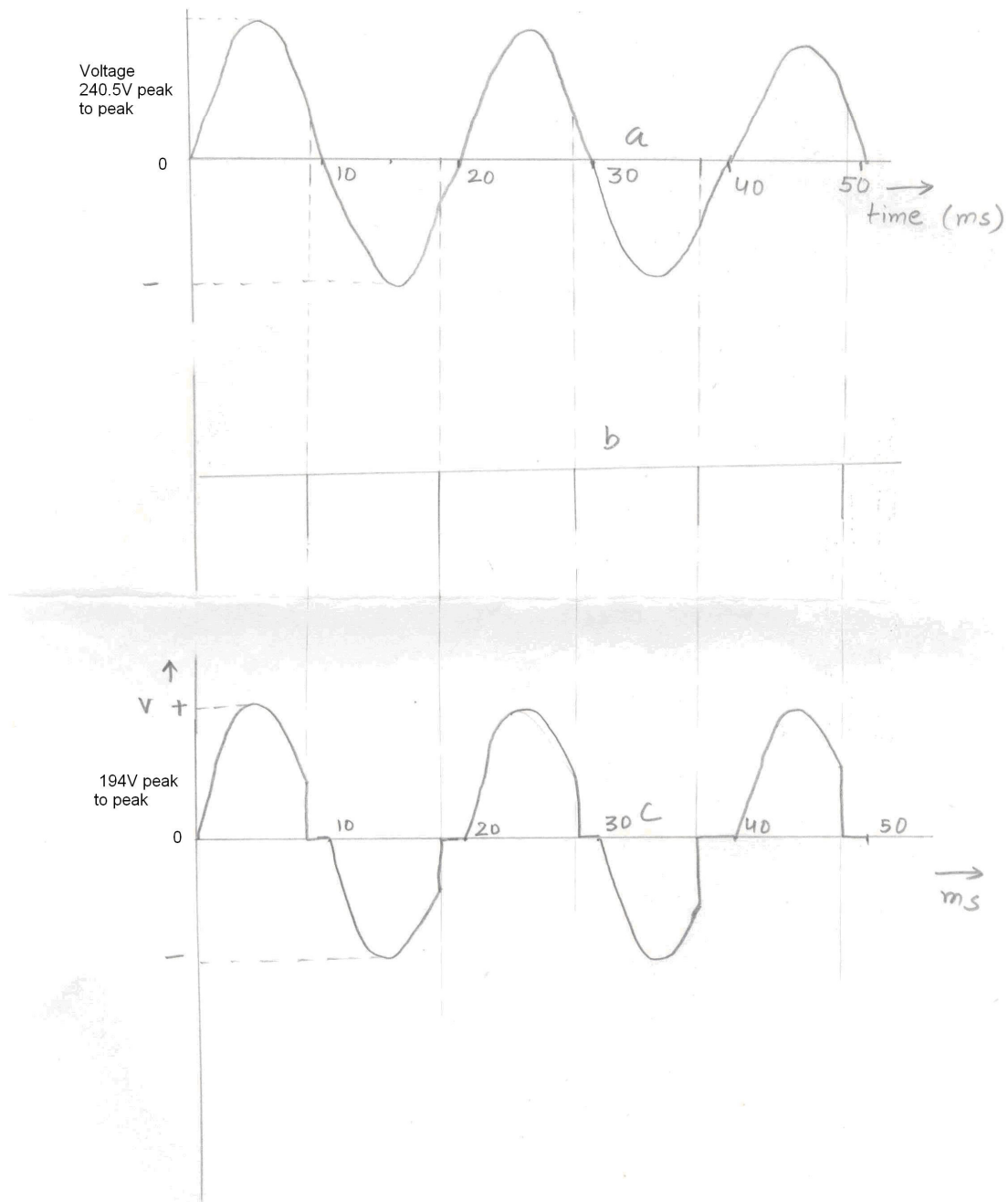


Fig. 15: a) Voltage waveform of mains supply
 b) Pulse pattern at the gate of triac at level 1
 c) Voltage across triac at level 1

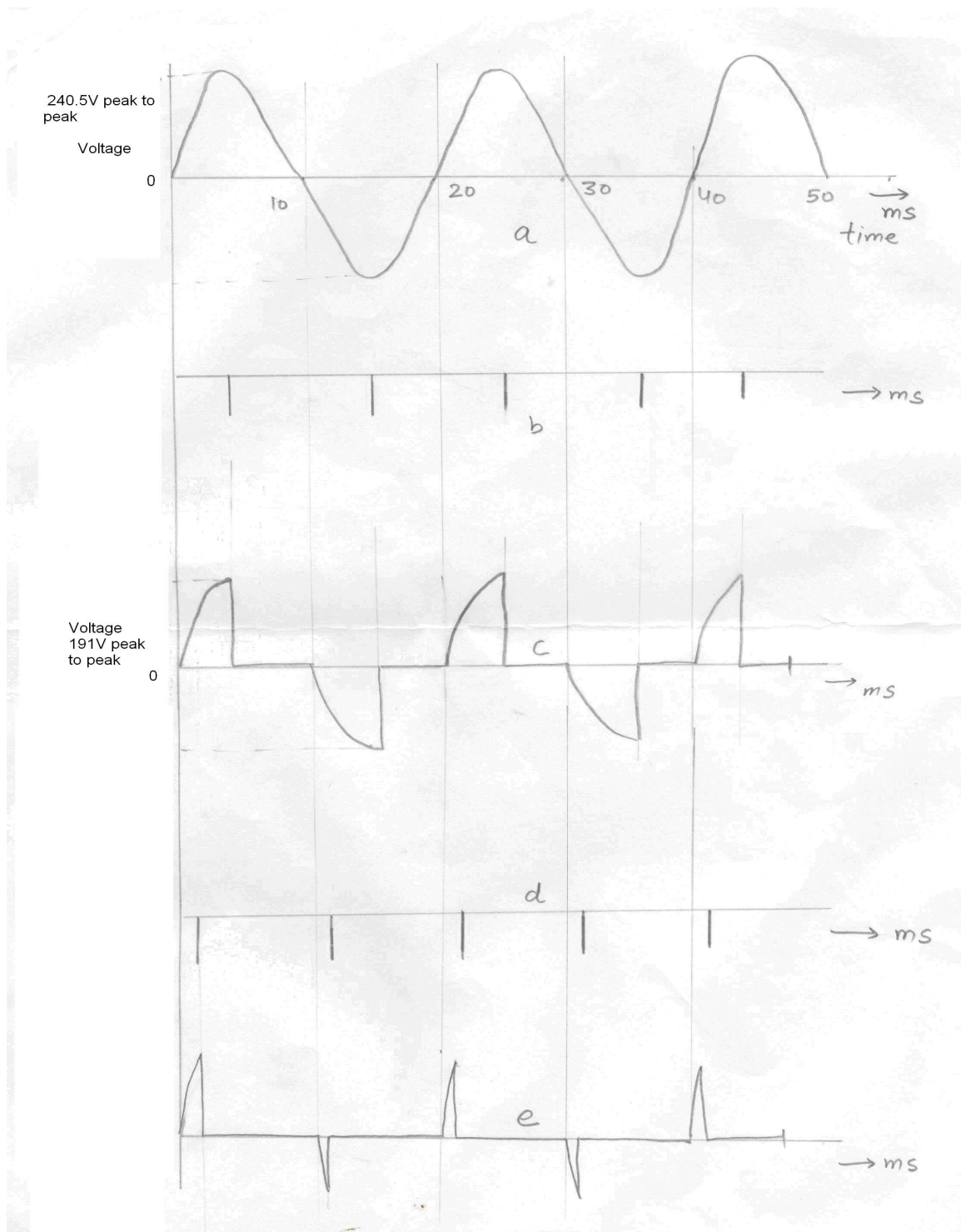


Fig. 16: a) Voltage waveform of mains supply
 b) Pulse pattern at the gate of triac at level 4
 c) Voltage across triac at level 4
 d) Pulse pattern at the gate of triac at level 7
 e) Voltage across triac at level 7

3. Implementation

Fig. 14 shows the complete circuit diagram. Figs. 15 and 16 show voltage waveforms for mains supply, pulse at triac and voltage across triac for level 1 (ON), 4 and 7 (maximum speed/intensity).

4. Conclusion

Intensity of a bulb and speed of the fan can be controlled in 8 different levels including ON and OFF using a TV Remote or a Serial port interface. The level at which different loads are operating is displayed on a LCD.

5. Further Scope

As one of the ports of AT89c52 is still unused, a keypad can be provided to control this system or this port can also be used to provide internet connectivity to the system. A little beyond this can also be thought of, where the device is instructed through the instructions, sent in an SMS, from a cellular phone.

References

- [1] A. S. Sedra, K. C. Smith, "Microelectronic Circuits", Fourth edition, 1982, Oxford University Press (2003).
- [2] K. J. Ayala, "8051 Microcontroller, Architecture, Programming and Applications", Penram International Publishing (India) (2001).
- [3] National Semiconductors, "National Analog and Interface Products Databook", National Semiconductors (2001).
- [4] <http://symlink.dk/electro/hd44780/>
- [5] 8052 Resources, Vault Information Services. [Online]. Available: <http://www.8052.com>
- [6] Infrared Remote Control Tutorial, Universal Solution Technology Research Inc. - Orlando, FL USA 32837-5314. [Online]. Available: <http://www.ustr.net/infrared/infrared1.shtml>
- [7] AT89C51 Datasheet, Atmel Corporation, [Online]. Available: http://www.atmel.com/dyn/resources/prod_documents/doc0265.pdf
- [8] TRIAC tutorial, American Microsemiconductor Inc., [Online]. Available: <http://www.americanmicrosemi.com/tutorials/triac.htm>
- [9] LM324 Datasheet, National Semiconductor Corporation, [Online]. Available: <http://www.national.com/pf/LM/LM324.html>
- [10] Building a 5 Volt power supply, Iguana Labs, [Online]. Available: <http://www.iguanalabs.com/7805kit.htm>
- [11] IN4001 Datasheet, ChipDocs, [Online]. Available: <http://www.chipdocs.com/pndecoder/datasheets/ZOWIE/IN4001.html>
- [12] TSOP1738 IR Receiver Datasheet, Vishay, [Online]. Available: <http://www.vishay.com/docs/tsop17.pdf>

6. APPENDIX:

User Manual

a) For Remote

1	Light ON
2	Light OFF
3	Fan ON
4	Fan OFF
Ch up	Increase Light Intensity
Ch down	Decrease Light Intensity
Vol up	Increase Fan speed
Vol down	Decrease Fan speed

b) For Computer

1	Light ON
2	Light OFF
3	Fan ON
4	Fan OFF
5	Increase Light Intensity
6	Decrease Light Intensity
7	Increase Fan Speed
8	Decrease Fan Speed

Design of Automatic Speed Control System in 4 – Wheelers for Avoiding Accidents

R.Prajit¹, V.Santhosh Kumar², S.Srivatsan³, R.Anantha Narayanan⁴
^{1,2,3,4}Mechanical Department, SRM University, India

ABSTRACT : Many accidents are at least partially caused because of rash driving. This can happen due to many reasons: poor judgment on the part of the driver, poor driving by the driver. Once the driver has lost control it is very difficult to drive the vehicle. To avoid rash driving and to prevent losing of valuable property we need some safety systems in the vehicles. We can make this true by controlling the flow of fuel to the Engine using sensors and other mechanical arrangements and this can bring the vehicle to the intended course. This simple system provides safety, and avoids rash driving in highly populated regions.

Keywords: formatting, insert, style, styling, sensors.

I. INTRODUCTION

For avoiding rash driving by drivers, we provide safety systems within the vehicles especially for 4-wheelers.. The main conclusion of their report is the crucial importance of seat belts and padded dashboards. Since then many important inventions such as automatic braking, electronic stability control, adaptive cruise control, traction control systems, antilock braking, etc. Our work is on a same pathway. In this system there is one power source (battery) as an input, one controlling element (toroidal coil with controlling rod arrangement), micro controller and small modifications in the already existing conventional designs of S.I & C.I engines.

1.1 Description

It is the system, which mainly uses photoelectric sensor, electrical power source and micro-controller for controlling the vehicle speed. The arrangement of the components used in this system. By using this simple system we can control automatically the speed of the vehicle. The vehicle will move with designated speed (low speed) of the control system even though the driver wants to drive the vehicle with high velocities.

1.2 Incremental Rotary Encoder

It is an incremental rotary encoder, also known as a quadrature encoder or a relative rotary encoder. In the optical type there are two gray coded tracks. The fact that incremental encoders use only two sensors does not compromise their accuracy. There can be an optional third output reference, which happens once every turn. This is used when there is the need of an absolute reference, such as positioning systems. The optical type is used when higher RPMs are encountered. Incremental encoders are used to track motion and can be used to determine position and velocity.

1.3 Photo Electric Sensor

A photoelectric sensor is a device used to detect the distance, absence, or presence of an object by using a light transmitter, often infrared. A retro-reflective arrangement places the transmitter and receiver at the same location and uses a reflector to bounce the light beam back from the transmitter to the receiver^[1]. An object is sensed when the beam is interrupted by the same^[1]. Diffuse reflection is the reflection of light from an uneven or granular surface such that an incident ray is seemingly reflected at a number of angles. It is the complement to specular reflection. If a surface is completely non-specular, the reflected light will be evenly spread over the hemisphere surrounding the surface.

1.4 Microcontrollers

A microcontroller (also microcomputer, MCU or μC) is a small computer on a single integrated circuit consisting internally of a relatively simple CPU, clock, timers, I/O ports, and memory. Program memory in the form of ROM is also often included on chip, as well as a typically small amount of RAM. Microcontrollers are designed for small or dedicated applications^[1]. Embedded systems are very sophisticated; many have minimal requirements for memory and program length, with no operating system, and low software complexity. Typical input and output devices include switches, relays, solenoids, LEDs, small or custom LCD displays, radio frequency devices, and sensors for data such as temperature, humidity, and light level.

1.5 Interrupt

Microcontrollers must provide real time (predictable, though not necessarily fast) response to events in the embedded system they are controlling. When certain events occur, an interrupt system can signal

the processor to suspend processing the current instruction sequence and to begin an interrupt service routine^[1]. In this Interrupt function the photoelectric sensor output is connected. Thus Microcontroller gives priority to the object detection.

1.6 Relay

A relay is an electrically operated switch. Many relays use an electromagnet to operate a switching mechanism. Relays find applications where it is necessary to control a circuit by a low-power signal, or where several circuits must be controlled by one signal.

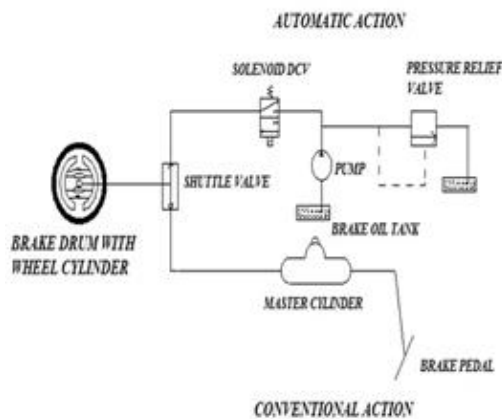


Figure 1: Automatic Action

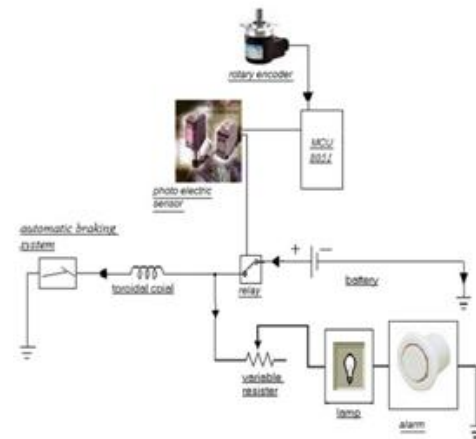


Figure 2: Conventional Action

1.7 Solid State Relays

Solid-state relays (SSR) control power circuits with no moving parts, instead using a semiconductor device triggered by light to perform switching. A SSR is a solid state electronic component that provides a similar function to an electro mechanical relay but doesn't have any moving components, increasing long-term reliability.

1.8 Toroidal Coil

It is an electromagnetic coil, which is used for pulling the controlling bar. It is energized by the electrical energy, which is gained from battery. It stores the energy and generates magnetic field around it. This attracts the controlling bar with some force. This movement of bar is used for controlling other components. The stored energy will be consumed by a lamp, which is connected to this toroidal coil which is in circular shape, which will generate magnetic field in circular manner, which is easy to pull the controlling rod.

1.9 Lamp and Alarm system

It is used for giving indication to the driver. It acts as a freewheeling path for the discharge of energy stored in the coil.

1.10 Battery

It is the main power source, which will give the required power supply to the toroidal coil for activating it and also to the sensor system.

1.11 Controlling rod

It is the plain metal rod of some diameter, which is used for controlling the fuel flow rate in S.I. engine carburetor or C.I. fuel induction pump. It is just attracted by the magnetic field in toroidal coil with some force. This force will give a movement to the controlling rod. This controls the other equipment.

II. MODIFICATIONS IN FUEL CONTROL SYSTEMS

We are giving small modifications to the already existing fuel supply system in S.I. engine and also in C.I. engine. But these modifications are very small.

2.1 In S.I engine

In this fuel supply system I am just using the control rod movement to force a metering rod element in to the main jet pipe that will restrict the flow area. So for given pressure depression in the carburetor a decrease in flow area implies the amount of fuel entering also decreases which eventually reduces the speed of the vehicle.

[2]

To achieve this we are providing a needle through a hole in to the main jet pipe for which we have to make a hole in the main jet pipe and we have to provide sealing arrangements.

2.2 In C.I engine

All of us know that how helical groove will help us in fuel induction pump I am just using the same helical groove with another circular ring with a hole. This ring is provided on the upper side of the spill port ^[2]. This ring is connected to the controlling rod. Controlling rod movement will help us to rotate this circular ring.

In this circular ring we will provide a hole which will act as same as spill port by rotating this ring the position at which the helical groove will open will be varied and at the same time the amount of fuel injected also varied

III. WORKING

The system will start working only when we applying the brake a number of times. The brake will act like a switch. When the brake is pressed the circuit will be completed by the battery which will start conducting the charge, which will be stored in the toroidal coil. Due to high valued resistor in the parallel circuit and low resistance of coil, 95% of charge will be stored within the coil. This charge will create a magnetic field around the coil that will pull the controlling rod.^[4] This controlling rod movement will be used for controlling the fuel supply system for varying the fuel rate entering into the combustion chambers.

In case of S.I. engine the controlling rod will force or move another needle, which will obstruct the flow area in main fuel nozzle so with this effect the quantity of fuel entering will be decreased. It cannot affect the airflow rate or any other thing; it only reduces the quantity of fuel entered.^[5] In case of C.I. engine the controlling rod movement will be used for rotating the circular ring on fuel induction pump, which will control the opening of hole through the port, when the hole is opened the fuel will be spilled out.

Due to opening of spill port in circular ring before the actual spill port (due to helical groove arrangement) the quantity of fuel injected will be varied. It will rotate in opposite direction to that of plunger rotation. If the controlling rod is moved to a full extent then the quantity of fuel injected will be very small irrespective of the actual spill port position. Due to opening of brake the circuit will be disconnected with the battery. During this energy that is stored in the toroidal coil will be discharged through the variable resistor and the lamp ^[3]. So there is no power or energy input to the toroidal coil. To discharge all the energy stored within the coil will take some time due to higher resistance in the path. During this period the system will continuously control the fuel flow. This time period we can vary by varying the resistance value of the resistor.

The lamp will be used as power consuming device and also for providing indicating signal for driver that he is in the highly populated road so he has to move slowly. So after some time the vehicle will be coming back to its normal state due to discharging of the stored power in toroidal coil. It will run as it runs normally ^[6].

IV. MATHEMATICAL INTERPRETATION FOR FORCE ACTING ON CONTROL ROD

R_v = Variable resistor R_c = Coil resistance

R_l = Lamp resistance V = Battery voltage

4.1 While Charging

Current output values from the battery when switch is in on condition.

I_1 is current flowing in main circuit; I_2 is current entering in to the toroidal coil;

I_3 is current entering in to the variable resistor

We know that $I_1 = I_2 + I_3$ ----- (1)

Equivalent resistance:

$$\frac{1}{R_{eq}} = \frac{1}{R_c} + \frac{1}{(R_v + R_l)} \Rightarrow \frac{1}{R_{eq}} = \frac{(R_c + R_v + R_l)}{(R_c (R_v + R_l))} \text{----- (2)}$$

$$I_1 = \frac{V}{R_{eq}} = \frac{V (R_v + R_c + R_l)}{(R_c (R_v + R_l))} \text{----- (3)}$$

$$I_2 = \frac{(R_l + R_v)}{(R_l + R_v + R_c)} I_1 \text{----- (4)}$$

$$I_3 = \frac{R_c}{(R_v + R_c + R_l)} I_1 \text{----- (5)}$$

Charge which is stored in the coil = -----.

Force generated = $N \cdot I_2$, where, N is Number of turns.

4.2 While Discharging

During discharging the switch will be in off state so the battery will be disconnected from the coil, now coil will start discharging.

I_4 is discharging current

$$R_{eq} = R_c + R_v + R_i; V = I_2 * R_c \text{-----} (7)$$

$$I_4 = V / R_{eq} = (I_2 * R_c) / (R_i + R_c + R_v) \text{-----} (8)$$

This gives the discharge time.

V. SALIENT FEATURES OF THE SYSTEM

- We used the low voltage battery which is safe to operate and easy to charge
- Controlling of force acting on the controlling rod is very easy. It is simply by varying the variable resistor.
- Energy consuming device which we are using i.e. a lamp is a less expensive device and it is also used for giving indication to the driver.
- The modifications given for the conventional carburetor or fuel injection pump are very small and simple.
- The costs of the system are of very less expense varying from Rs. 1000/- to Rs. 3000/- only.
- This system can be more effectively used for heavy duty locomotives and automobiles, which are the main cause for accidents.

VI. LIMITATIONS

There is a chance of the battery getting disconnected from the system when driver wants. So to reduce this there is a requirement of frequent checking of the complete system by some external government employees. In case of C.I. engine the system will be little complicated which is difficult to manufacture. The system will start working only when there is application of brake otherwise it will not start working.

VII. CONCLUSION

Finally we can use this system for S.I. engines, C.I. engines and also for gas engines. We can vary the value of variable resistance as per our requirements. For buses and Govt. vehicles we will set the resistance to a low value. So for these vehicles the time for which the system is in an active condition, is very less. So the time for which the vehicle will move slowly will also be very less. But for Lorries and autos we can use higher values of variable resistance. So it will take more time to consume the charged energy. These vehicles need more time to get back the system to its normal state.

So by using this system we can reduce rash driving and accidents up to 35% and we can save many lives and many valuable properties. We can reduce the rash driving within cities, within the regions of school zones, villages that are located at the highways and beside the highways.

VIII. ACKNOWLEDGEMENT

This is to acknowledge that the following people without whom this producing paper wouldn't have been successful. We would like to express our sincere gratitude to Mr. RAJESH B.Tech, M.Tech, (Ph.D.) our guiding staff, Head of Mechanical Department Mr. C.K. LAKSHMY NARAYANAN B.Tech, M.E and our dean Mr. N. VASUDEVAN, B.E, M.Tech for providing us with all necessary facilities, encouragement and their support. We also place record, our sense of gratitude to one and all who directly or indirectly have lent a helping hand in this venture.

REFERENCES

- Edward C. Jordan, Keith G. Balmain, *Electro magnetic waves and radiating system* (Prentice Hall (1968)).
V. Ganeshan, *Internal combustion engine* (New Delhi, Tata McGraw-Hill, 2007).
R.K. Rajput, *Thermal engineering* (New Delhi, Laxmi Publications, 2010).
Kripal Singh, *Automobile engineering* (Standard Publishers Distributors, 2002).
John B Heywood, *Internal combustion engine function*. (McGraw-Hill, 1988).
Richard Fischer, Rolf Gscheidle, Uwe Heider, *Modern automatic technology* (Verlag Europa-Lehrmittel Nourney, Vollmer GmbH & C; English edition (2006)).

**DIGITAL SECURITY LOCKING SYSTEM –
USB BASED GSM SECURITY SYSTEM**

CHAI CHONG YI

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

**Faculty of Engineering and Science
Universiti Tunku Abdul Rahman**

April 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 : CHAI CHONG YI

ID No. : 07UEB06518

Date : 15 APRIL 2011

APPROVAL FOR SUBMISSION

I certify that this project report entitled “**DIGITAL SECURITY LOCKING SYSTEM– USB BASED GSM SECURITY SYSTEM**” was prepared by **CHAI CHONG YI** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons.) Electrical and Electronic Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : _____

Supervisor: Dr. Yong Thian Khok

Date : _____

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of University Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2011, Chai Chong Yi. All right reserved.

Specially dedicated to
my beloved family,
supervisor Dr. Yong Thian Khok
and partner Miss Chan Sieu Chen

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Dr. Yong Thian Khok for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, I would like to express my gratitude to my loving parent and friends who had helped and given me encouragement during the development of the project as well as providing guidance on the project.

In addition, I would like to express my sincere thanks to my partner Miss Chan Sieu Chen for providing me guidance and help me in completing this project as well as providing me the opportunity to work as a team to complete a project.

Lastly, I would like to thanks UTAR for providing me an opportunity to complete a project as a part of fulfilment of the requirement for the Bachelor of Engineering Programme.

DIGITAL SECURITY LOCKING SYSTEM

ABSTRACT

Security threat has become an important issue to most of the people especially in urban area. With the increase in criminal rate, having a security system is desirable to ensure home is a safe place to live in. However, security alarm system may not be cheap depending on the features available on the security system and most of the security system can only provide local alarm which property owner cannot know what is happening to their property at the time emergency case happens. This problem has become the motivation of developing a low cost, low power, simplicity of operation, small and standalone security system. The developed project has to make use of mobile phone that use USB cable as GSM terminal instead of GSM modem which is not cost effective. Mobile phones that use USB cable can be found everywhere and some of the old model may potentially become GSM terminal to lower down the cost of the whole project. In the project, PIC24FJ64GB002 is used as the core processor of the project to act as a USB host as well as processing some task required by the security system. Mobile phone Sony Ericsson C902 is the main experimental GSM terminal in the system. A LCD and a 4x4 keypad are used to change setting for the security system. This system can then integrate with RFID based locking system to become a complete digital security locking system which is more useful and can be used in home, office or automotive.

TABLE OF CONTENTS

DECLARATION	ii
APPROVAL FOR SUBMISSION	iii
ACKNOWLEDGEMENTS	vi
ABSTRACT	vii
TABLE OF CONTENTS	viii
LIST OF TABLES	xi
LIST OF FIGURES	xii
LIST OF SYMBOLS / ABBREVIATIONS	xiv
LIST OF APPENDICES	xv

CHAPTER

1	INTRODUCTION	1
	1.1 Background and Motivation	1
	1.2 Aims and Objectives	3
	1.3 Outline of Thesis	3
2	LITERATURE REVIEW	4
	2.1 Introduction	4
	2.2 Alarm System	4
	2.2.1 History of Alarm System	4
	2.2.2 Types of Alarm System	5
	2.3 USB Based Embedded System	6
	2.4 Short Message Service (SMS)	7
	2.4.1 History of SMS	7

	2.4.2	Advantages of Short Service Message	7
2.5		AT Command	8
	2.5.1	History of AT Command	8
	2.5.2	Types of AT Command	9
	2.5.3	AT Command Mode	10
2.6		Comparison and Critic	12
3		METHODOLOGY	13
	3.1	Introduction	13
	3.2	Overall System Setup	13
	3.3	USB Based GSM Alarm System Setup	15
	3.4	GSM terminal	16
	3.4.1	GSM/GPRS Modems	16
	3.4.2	Cellular Phones	17
	3.4.3	Choosing GSM terminal	17
	3.5	Modem Commands	18
	3.6	Sensor	18
	3.7	Microcontroller	19
	3.8	Matrix Keypad	21
	3.9	LCD Display	22
	3.10	MPLAB IDE	24
	3.11	EAGLE Layout Editor	25
	3.12	Schedule of FYP 2	25
4		RESULTS AND DISCUSSIONS	26
	4.1	Introduction	26
	4.2	Software Implementation	26
	4.2.1	Software Language	26
	4.2.2	USB	27
	4.2.3	Emulating Data EEPROM	37
	4.2.4	Decoding keypad	39
	4.2.5	Program on LCD Display	42
	4.3	Program overview	44

		x
4.3.1	Option menu	46
4.3.2	Detailed Operation on Changing Password	50
4.3.3	Detailed Operation on Changing Target Phone Number	51
4.3.4	Detailed Operation on Changing SMSC Number	52
4.4	Results of Tested Possible Mobile Phone as GSM Terminal	53
4.5	Hardware implementation	54
4.5.1	Hardware Placement	54
4.5.2	Circuit Diagram	54
4.6	Problem Encounter	56
5	CONCLUSION AND RECOMMENDATIONS	58
5.1	Conclusion	58
5.2	Limitation and Future Enhancement	59
	REFERENCES	60
	APPENDICES	61

LIST OF TABLES

TABLE	TITLE	PAGE
Table 2.1:	PDU String Decoding	11
Table 3.1:	PIC24FJ64GB002 Specification	20
Table 3.2:	LCD Pin Table	23
Table 4.1:	USB Device Classes	29
Table 4.2:	Enumeration Steps	30
Table 4.3:	USB Files in Programming Part of the Project	31
Table 4.4:	Effect of USBHostTasks Calling Interval on USB Enumeration	32
Table 4.5:	USB Result Code	37
Table 4.6:	Data Memory Structure	38
Table 4.7:	User Data Storage Location	38
Table 4.8:	List of Option Menu	47
Table 4.9:	Detail Operation on Changing Password	50
Table 4.10:	Detail Operation on Changing Target Phone Number	51
Table 4.11:	Detail Operation on Changing SMSC Number	52
Table 4.12:	Test Result of Mobile Phones	53

LIST OF FIGURES

FIGURE	TITLE	PAGE
Figure 2.1:	PDU Encoding Method	11
Figure 3.1:	Digital Security Locking System Setup	14
Figure 3.2:	Project Division of Digital Security Locking System	15
Figure 3.3:	USB Based GSM Alarm System Setup	16
Figure 3.4:	GSM Modem from Wavecom	17
Figure 3.5:	Snap Action Switch	19
Figure 3.6:	Magnetic Proximity Sensor	19
Figure 3.7:	PIC Microcontroller (SPDIP Package)	20
Figure 3.8:	Pin Diagram of PIC24FJ64BG002	21
Figure 3.9:	A 4x4 Matrix Keypad	22
Figure 3.10:	Internal Connection of a 4x4 Matrix Keypad	22
Figure 3.11:	A 2x16 LCD Display	23
Figure 3.12	Gantt Chart of FYP Part 2	25
Figure 4.1:	Example of TPL List	31
Figure 4.2:	Transfer Types of Different Endpoint	34
Figure 4.3:	Code to Test Transfer Types and Initiating Data Transfer	34
Figure 4.4:	Flow Chart of Testing USB Transfer Type and Initializing Data Transfer	35
Figure 4.5:	Generic Host Driver Write Function	36

Figure 4.7: Flow Chart of Decoding Keypad Part 1	40
Figure 4.8: Flow Chart of Decoding Keypad Part 2	41
Figure 4.9: LCD Initiation Process	42
Figure 4.10: Program Code for Function Set and Entry Mode Set of LCD	43
Figure 4.11: Program Code of Writing Data to LCD	44
Figure 4.12: Program Flow Chart of USB Based GSM Security System	45
Figure 4.13: LCD Display on (a) Start up, (b) Supported Terminal Attached, (c) Unsupported Terminal Attached, (d) Terminal Detach	46
Figure 4.14: Menu Flow Chat Part One	48
Figure 4.15: Menu Flow Chat Part Two	49
Figure 4.16: Circuit Diagram of Digital Security Locking System	55
Figure 4.17: PCB Layout of Digital Security Locking System	56

LIST OF SYMBOLS / ABBREVIATIONS

ASCII	American Standard Code for Information Interchange
AT	ATtention
CDC	Communications Device Class
CPU	Central Processing Unit
EEPROM	Electrically Erasable Programmable Read-Only Memory
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
HID	Human Interface Device
I/O	Input / Output
LCD	Liquid Crystal Display
LED	Light-emitting Diode
OTG	On-The-Go
PCB	Printed Circuit Board
PDU	Protocol Data Unit
PIC	Peripheral Interface Controller
PID	Product Identification
RAM	Random-access Memory
RFID	Radio Frequency Identification
SMS	Short Message Service
SMSC	Short Message Service Center
SPDIP	Skinny Dual In-line Package
SPI	Serial Peripheral Interface
TPL	Target Peripheral List
USB	Universal Serial Bus
VID	Vendor Identification
WLAN	Wireless Local Area Network

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
	APPENDIX A: Main Program Code	61

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

In this modern age, security has become an important issue for most of the people especially in the urban area. With the living standard of people improved, people tend to have more valuable property inside their house. Some people will try to rob or steal the property which may endanger the safety of property owner or people inside the house. To overcome the security threat, most people will install bunch of locks or alarm system.

Alarm system is useful in reducing the security threat at home regardless it is a high technology or a simple alarm system. Robbers and thieves will tend to make house with no alarm system as a target rather than those with alarm system. Even a dummy alarm system can scare off some of the thieves and robbers due to human nature where most people tend to stay out of trouble.

With alarm system, property owner do not have to always look after their property or having people guard at the property. This can reduce the time and cost on guarding property. Moreover, security system can guard the property all the time without have to worried getting tired or boring.

There are many types of alarm system in the market which utilize different types of sensor. Each types of sensor can detect different types of changes occur in

the surrounding and given out alert when the value is exceed the pre-set value. When combining few types of sensor, a place can me monitor very well.

Alarm system typically consists of sensor, alarm siren and relay. For home alarm system, alarm siren will sound whenever the sensor detects the door or window is open. This kind of security system can only provide local alarm and property can never know when emergency case happen. Some of the new alarm system includes a modem which will send alert to property owner when any security threat occurs. Property owner can know what is going on and taking any further action as soon as possible to minimize any loses.

When comes to security product in the market, the price for security product is usually high. Security manufacturer have to come out with new security technique form time to time to ensure their product has high level of security level. There are some low cost security systems which offer limited security. Thus, it is desirable to build a low cost, low power, simple operation, small and standalone digital security alarm system.

For digital security alarm system, the cost of fixed line security alarm system is fair enough. However, the prices for wireless security alert system are generally high in the market due to the cost of GSM/GPRS modem are high. Even the module of the GSM/GPRS modem is costly. There are many old models of cellular phones in the market. This cellular phone can be use as a low cost terminal to replace GSM/GPRS modem. This cellular phone can send alert to particular person when emergency case happened. By integrating these phones in the digital security alert system, the cost would be much more lowered. When integrate to the digital locking system, we can create a low cost digital security locking system.

1.2 Aims and Objectives

The aims and objective for this project is to develop a security system which is based on GSM network and USB host terminal. The security system is able to send SMS to user when emergency case occurs. The cost of the developed project should be low cost, low power, simple operation, small and standalone. This project is then able to integrate with RFID based locking system to form digital security locking system.

1.3 Outline of Thesis

This thesis starts with introduction of the digital security locking system and the focused part USB based GSM security system. In this chapter, the motivation as well as objective of the project development is stated.

Second chapter state the review on the currently available security system and possible method to develop a security system using GSM terminal. If possible, USB connection is used to connect between GSM terminal and microcontroller. Next chapter will be discussing on which potential component will be used in the project and how the project development progress is identified.

Chapter four is about how the project is developed and the results of each experiment are stated. After the project is developed, it will then combine with RFID based locking system to form a complete digital security locking system. The results of integration as well as problems encounter are stated in this chapter. The final chapter will be on the conclusion as well as future enhancement can be made on the project to develop a more powerful digital security locking system.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

To understand better how a security system should work, a review on others people work is important to develop a better solution. In this chapter, history of alarm system as well as types of alarm system will be reviewed. GSM history as well as AT command is reviewed to develop a GSM based security system. Furthermore, working principle of USB and embedded host can provide cheaper solution for GSM based security system.

2.2 Alarm System

2.2.1 History of Alarm System

Long before the invention of electronic alarm system, people use animal such as dogs to alert them when someone is trying to steal the belonging from them or enter certain area. In the middle age, people uses large gong to alert people when there is invasion. Things started to change when Tildesley invented the first home alarm system, which contain a series of bells connecting together and link to the door (Gregory, 2007). When someone is open the door, the bells will ring and alerting the people inside. Soon after that, Edwin Homes invented first electro-mechanical alarm system (Gregory, 2007). The alarm system contains a spring connected to the switch

which attach to door and whenever the door is open, the circuit will closed and the bell will ring. This alarm system only uses simple electrical circuit such as switch, relay and bells. With the invention of integrated circuit, the size of alarm system has becomes smaller and the features has increased. The alarm system nowadays is no longer work only on switch, but also can work on infrared, ultrasonic, microwave, photoelectric and more. Some alarm system even has integrated circuit which can send alert to property owner when there is emergency case occurs (Gregory, 2007).

2.2.2 Types of Alarm System

Alarm system can be categorized into two main types. First type is the alarm system that uses wired sensor nodes and another type uses wireless sensor nodes. Wired alarm system has hard wire connected between the sensor network and the main processing unit while wireless alarm system utilize wireless local area network (WLAN), wireless USB and Bluetooth for communication between sensor and the main processing unit.

Research done by Ahmed, A., Ali, J., Raza, A. & Abbas, G. (2007) shows that wireless sensor network has advantages over wired sensor network in the sense that the sensor can be placed at any terrain without have to worry about the wiring problem. However, most of the wireless sensors run on battery supply. Battery lifetime becomes the most deadly disadvantages to the wireless network especially the distance is long. Longer distance require stronger signal which also means higher power require to drive the signal. In another words, power efficiency of wireless sensor determines the operating lifetime of sensor network. Once battery is drained up, it can no longer provide any protection to the perimeter and are vulnerable to any attack.

On the other hand, wired sensor network does not suffer power source problem but it does has disadvantages where the quantity of sensors is limited due to the wiring problems. The network becomes complicated with the number of sensor build up. However, the transmission speed of wired network is fast when compare

with wireless network. Additionally, the wired sensor network can provide more reliable security over wireless sensor network as wired sensor network does not suffer signal lost where wireless sensor network did. Besides, wired sensor network is easier to setup compare with wireless sensor network (Ahmed, 2007).

Based on the survey done by Zhao & Ye (2008) indicate the cost for wireless sensor network vary by technology used. Both WLAN and Bluetooth are high cost and high power consumption technology while wireless USB offer low on both cost and power consumption. To inform the user when any event occur, it can be done via internet or GSM network. In order for user get the information as soon as possible, GSM network is preferable as user will carry phone almost all the time compare to staying in front of computer all the time. According to research done by Huang, H. P., Xiao, S. S., Meng, X. Y. & Xiong, Y. (2010), GSM is a mature technology and hence, high security, wide coverage and can transmit over long distance.

2.3 USB Based Embedded System

With the growth of USB technology, USB has been gradually replaces traditional serial protocol and parallel protocol. Due the ease of use, fast transfer rate and high stability, USB becomes a favourite choice of most people. USB can support control transfer, bulk transfer, interrupt transfer and isochronous transfer. Control transfer enable USB host read the configuration of device connected to it. Bulk transfer can transfer non-fixed amount of data with error correction while interrupt transfer is use to transfer small amount of data. For isochronous transfer, the data is transfer at a constant rate and no retransmission if error occurs (Kim, Y. S., Kim, H. S. & Lee, 2005). Most electronic appliances in the market nowadays uses USB interface.

2.4 Short Message Service (SMS)

SMS is a technology which enables a short message to send and receive between mobile phone. SMS can send up to 140 bytes or equivalent to 1120 bits of data. If these 1120 bits is fully utilize, SMS can include 160 characters of 7-bits character encoding which is the English character and 70 characters of 16-bit Unicode used for many language. SMS is compatible for all GSM phones. GSM 07.05 is a standard for short message rules for SMS (Bodic, 2005).

2.4.1 History of SMS

SMS was first appeared in 1992 in Europe. SMS is included in Global System for Mobile Communication (GSM). Both SMS and GSM standard was developed by European Telecommunication Standard Institute (ETSI). The development and maintenance of GSM and SMS standard was then turn to Third Generation Partnership Project (3GPP) (Bodic).

2.4.2 Advantages of Short Service Message

According to the survey done by CTIA Wireless Association in year 2009, there are about 4.1 billion messages being sent daily. According to Bodic, the core reason why SMS is so successful is mainly due to its conveniences.

i. SMS can be sent or read at anytime

When a SMS is received, the message will store into the memory. User can either read it immediately or read it later. Nowadays, most people have their mobile phone with them most of the time. SMS can be read or sent anytime regardless the user is in office, home, street or even in the bus.

ii. SMS is less disturbing than phone call

If user is on a phone call, every word the user says will immediately pass to another side of the phone. Sometimes, user may require some thinking before replying the question. Furthermore, SMS will not create load noise as talking on phone. This enable user to send or receive SMS while they are in library or theater without have to going outside. This enable users to keep in touch all the time without causing much disturbing.

iii. SMS can be sent to offline mobile phone

The benefits of SMS is if the destination phone is offline, the user can still receive the message after he or she is turn on the phone. Most of the time, the network signal may be weak on some area especially rural area. The message can still be receive by user if the user moves to a place with better network coverage.

2.5 AT Command

AT Command is the modem command code and it is define in GSM 07.07 standard. GSM 07.07 provides AT Command which enable mobile platform to communicate with Data Terminal Equipment in serial communication (Cao, 2009).

2.5.1 History of AT Command

In early 1980s, a company named Hayes has developed a modem called Hayes Smart modem 1200. Shortly after that, the company releases another modem call Hayes smart modem 2400. The model number simply represents the baud rate of the modem. They call the modem as smart modem because of the ability of the modem to auto-dial a number. Furthermore, this modem can send morse mode, work with Radioteletype (RTTY) and amateur radio repeaters. These two modems accept same programming command with only the operating speed is different. This has a great

advantage where the old driver can be use by new model of modem. Few modems makers has follow what Hayes did, where they develop modem that accept same command set as previous modem and called it “Hayes Compatible”. Soon after that, Hayes sued these modems makers for using the word “Hayes” in their product. Subsequently, these modem makers has change to “AT Command Set compatible”. AT command is taken from the words Attention, where the first two letters is used. AT command is the programming command that modem receive from the input and execute it. However, mobile usually do not implement full AT command and some of the command used in mobile phone may varied from manufacturer to manufacturer and model to model. Overall, the GSM/GPRS modem will have better support on AT command (Bodic).

2.5.2 Types of AT Command

There are two types of AT commands, which categorize as basic commands and extended commands. Basic commands are the commands that do not start with “+”. For example ATA stands for Answer, ATD stands for dial and ATH stands for hook control. On the other hand, extended command are the command that start with “+”. As for example, AT+CMGS will send the message and AT+CMSS will send message from storage. Following is some example of extended AT command (Developer's Home, 2004-2010).

+CSMS	Select message service
+CPMS	Preferred message storage
+CMGS	Message format
+CESP	Enter SIM block mode protocol
+CMS ERROR	Message failure code
+CSCA	Service center address
+CSMP	Set text mode parameter
+CSDS	Show text mode parameter
+CSCB	Select cell broadcast message type
+CSAS	Save setting

+CRES	Restore setting
+CNMI	New message indication to TE
+CMGL	List message
+CMRG	Read message
+CNMA	New message acknowledgement
+CMGS	Send message
+CMSS	Send message from storage
+CMGW	Write message to memory
+CMGD	Delete message
+CMGC	Send command
+CMMS	More message to send

2.5.3 AT Command Mode

Control of SMS function can be made via AT command in three mode which is Block mode, Text mode, and Protocol Data Unit (PDU) mode. AT stands for Attention. Block mode is a binary communication protocol including error protection. Text mode is a character based protocol suitable for high level software application while PDU mode is a character based protocol with hexadecimal-encoded binary transfer of command. PDU mode is usually for low level software driver that do not understand the content of command while text mode is suitable for high level programming language (Bodic).

PDU is harder for people to read it but it has advantages where it has lots more features compare to the simple text mode. Thus, PDU mode can support better on GSM modem and some of the mobile phone will only understand PDU mode. PDU mode has three encoding mode namely 7-bit code, 8-bit code, and 16-bit code. 7-bit and 8-bit code is use for normal ASCII code which is English message while 16-bit code is use for Unicode character which supports not only English but Chinese and other language as well. It is clear that to use other language in sending message, PDU mode in 16-bit code must be adopted since text mode does not support 16-bit mode (Cao, 2009).

When come to encoding in PDU mode, we need to know the meaning of each of the string. Since the syntax is fixed, the PDU string can easily created with the reference of the meaning for each words. The only tricky part of the PDU string is the semi-octets string. For example an international number +60123456789. To encode it into PDU string, we need to remove the “+” sign and add “F” to the end of the string. Then, each of the pair is twisted to convert to PDU string. +60123456789 will become 0621436587F9 (Wan, Zhang & Fang, 2008).

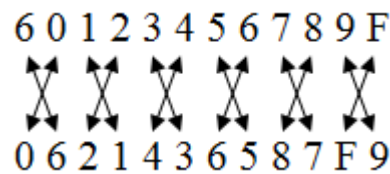


Figure 2.1: PDU Encoding Method

One example of PDU string contains the following information. 07910621000010F511000B910621436587F90000AA08C7F79B0C2287F3 will have the meaning as shown in the table below.

Table 2.1: PDU String Decoding

Octet(s)	Description	Example
07	Length of the SMSC information	7 octets
91	Type of address of SMSC	International format
0621000010F5	SMSC number	+60120000015
00	First Octets of SMS-DELIVER message	-
0B	Length of sender address	11
91	Type of address of the sender number	International format
0621436587F9	Sender number	+60123456789
00	Protocol identifier	-
00	Data encoding scheme	-
08	Length of User data	8
C7F79B0C2287F3	User data	Good Day

2.6 Comparison and Critic

The security system purpose by Zhoa & Ye and Huang et al. has a disadvantage where it runs on battery supply. User will never know when the power of battery will drained up and causing the security system failed to work. Besides, they uses RS-232 serial interface which normally found on GSM modems and rarely on cellular phones. The system built by Kim et al. is based on USB which is widely used by cellular phones and most electronic appliances today.

As for the modem command, Cao and Wan, Zhang el al. recommend PDU mode as Chinese text is required and compatibility issue on GSM modem and most of the phones. Text mode is easier to apply but is hardware limited. PDU mode is better option since it can be supported by most of the phone and GSM modem.

CHAPTER 3

METHODOLOGY

3.1 Introduction

To start the development of a project, the component needed must first be identified clearly. In this chapter, the list of the component required as well as technical specification of the component is discussed. The software needed for project development is also discussed in this chapter.

3.2 Overall System Setup

Figure 3.1 shows the core components of digital security locking system and how the components are related to each other. Direction of arrow represent flow of data and each block represent one component. Digital security locking system consist of seven core component which is PIC microcontroller, RFID reader, keypad, magnetic lock, alarm siren, door opening sensor, LCD display and GSM terminal.

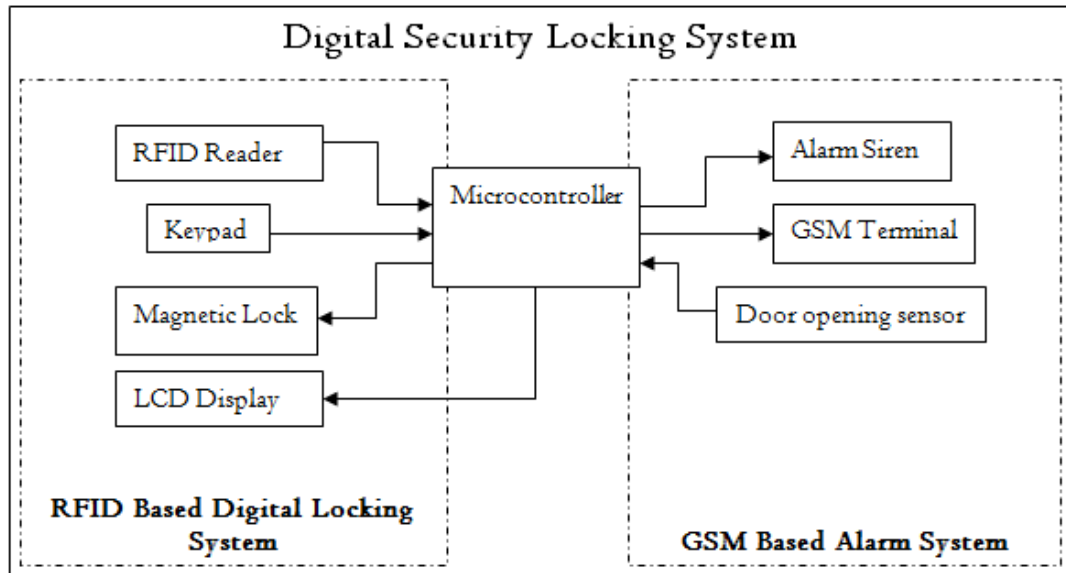


Figure 3.1: Digital Security Locking System Setup

The RFID reader will read a RFID tag and sent the data to the microcontroller. If the tag ID is in the access list, a signal will be sent to magnetic lock to release the lock. When there is no card present, microcontroller will keep on reading from keypad for password input. If the password matches the current password, lock will be release. User can choose to use either way to release lock or both of them depend on user setting. However, if the door is force to open without entering password or scanning tag, the microcontroller will send out a signal to the alarm and GSM terminal. The alarm will sound and GSM terminal will send a message to inform the property owner. Password is needed to turn off the alarm.

Digital security locking system consist two main parts which is the RFID based locking system and USB based GSM alarm system. These parts will be developed separately and combined at the final stage. After combination of both parts, optimization of project will be carry on. Figure 3.2 shows the project division of digital security locking system.

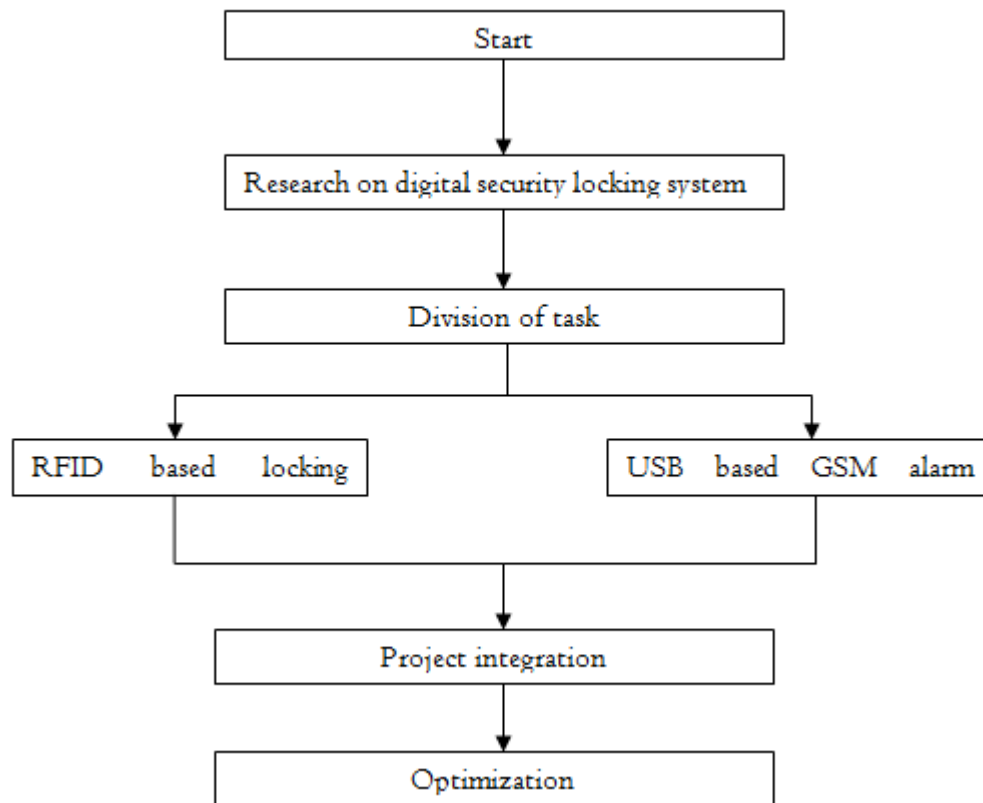


Figure 3.2: Project Division of Digital Security Locking System

3.3 USB Based GSM Alarm System Setup

USB based GSM alarm system consist of six core component which are door opening sensor, alarm siren, GSM terminal, LCD display, keypad and microcontroller. When the system is turn on, door opening sensor will keep on checking the door. If the door is open, microcontroller will send signal for the alarm siren to sound. At the same time, microcontroller will send a message to the property owner through GSM terminal. Figure 3.3 show the structure of USB based GSM alarm system.

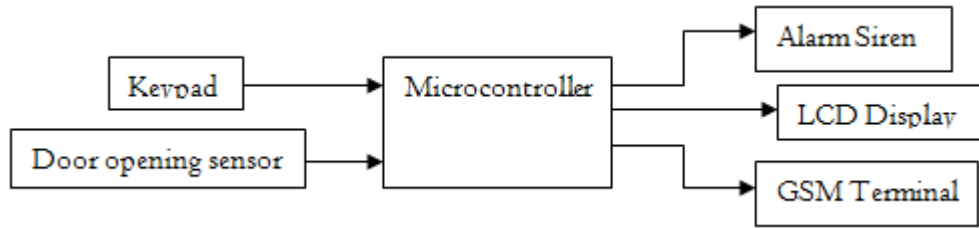


Figure 3.3: USB Based GSM Alarm System Setup

There are few major components that are needed to develop a digital security locking system. These components are PIC microcontroller PIC24FJ64GB002, 125 kHz RFID card reader, 4x4 matrix keypad, LCD display, magnetic lock and GSM terminal.

3.4 GSM terminal

3.4.1 GSM/GPRS Modems

GSM/GPRS modem is the easiest way to build the security alert system. The best thing about the GSM/GPRS modem is that this modem can support AT commands very well and come in two types on connectivity interface, serial RS-232 and USB. However, the choice of these modems on the market is limited and the prices are expensive. USB has advantages on the ease of use but the way to program it is slightly difficult. The deep understanding in USB is a must to work with programming with USB.



Figure 3.4: GSM Modem from Wavecom

3.4.2 Cellular Phones

Cellular phones can serve as a cheaper terminal for sending SMS. However, not all cellular phones can work with AT Commands. Only certain models of phones are compatible with AT Command. Another issue will be the availability of data cable of old cellular phones. To find an old model of cell phone, it will be very easy. However, to get the data cable will be the hardest work as old model of cell phone mostly did not come with data cable. New model of cell phones usually come with cable which is in USB interface. Old model phones use RS-232 interface which is easier to work with compared to new model of phone which utilize USB cable.

3.4.3 Choosing GSM terminal

After doing some research, it is better to use a mobile phone as a GSM terminal for lower the cost of the system. Since the price of mobile phone is very much depends on the model numbers, old model of mobile phones will be used in the project as long as the phones support AT Command. Next, USB interface phone will be used as the availability is very much higher than those phones using RS-232 serial interface cable. Mobile phone Sony Ericsson C902 is the main experimental GSM terminal for this project and few other phones will be taken to test the performance of the project.

3.5 Modem Commands

AT command will be the best choice among the modem commands as it has a standard command set. AT command is applicable in all AT Commands compatible modem while other modem command only work on certain model of phones only and usually the model of phones uses same modem commands are limited. Most of the phone in the market supports AT command. PDU mode will be adopted as it support on most of the GSM modem as well as mobile phone. Text mode is easier to implement but only very little phone does support text mode while almost all the phone that support AT command will support PDU mode.

3.6 Sensor

Sensor can be either in wired or wireless. Since the number of sensor is very small and to lower the cost, wired sensor is recommended. Wired sensor is also easier to configure than wireless sensors. To get a stable signal from the sources, switch is the best choice. Switch can give a very accurate signal to feed to the microcontroller as an input to indicate the door is closed or opened. This is important as good signal can ensure the happening of false alarm is minimized. Since signal is required when the door touches the switch, snap action switch or proximity sensor will be the best available choice. Snap action switch may be good but due to it mechanical switch properties makes the device cannot last for long runs. There are many types of proximity sensor available and cheapest solution with high stability will be magnetic sensor.



Figure 3.5: Snap Action Switch

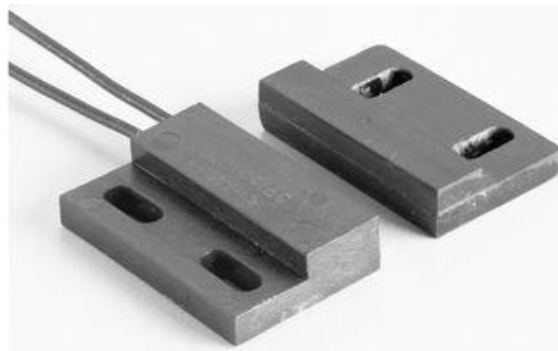


Figure 3.6: Magnetic Proximity Sensor

3.7 Microcontroller

There are several types of major microcontroller available in the market for our choice. They are 68HC08/68HC11 from Freescale or formally known as Motorola, 8051 from Intel, AVR from Atmel, Z8 from Zilog and PIC from Microchip Technology. Among the microcontroller, PIC microcontroller from Microchip is recommended. PIC stands for Peripheral Interface Controller. Microchip is the top supplier of 8 bits microcontroller in the world.

Since USB interface is chosen, a microcontroller with host capability is a must. USB require a host to control the speed of buses and manage the device connect to it which is called slave. Majority of the PIC microcontroller do not have USB capability and even they have USB capability, the only serve as a device or USB slave. USB On-The-Go (OTG) is a technology which enables a PIC

microcontroller to serve as a USB host. This technology was developed few years ago and until recently, the technology is included in some new PIC microcontroller.

There are two way to getting USB work on microcontroller. One is using USB host peripheral control together with another microcontroller and another is using microcontroller with USB OTG capability. The price of USB host peripheral is not cheap and deep knowledge on the device is required for it to work. Thus, microcontroller with USB OTG capability will be the choice.

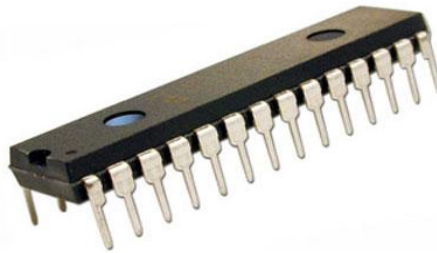


Figure 3.7: PIC Microcontroller (SPDIP Package)

PIC24FJ64GB002 is a 16 bits microcontroller with 64k flash memory. It has USB-OTG capability and 28 pins Shrink Plastic Dual In-Line Package (SPDIP) which is suitable to fit into our design. Besides, it is the cheapest and only PIC with USB-OTG capability in SPDIP. PIC24FJ64GB002 has the following features. PIC24 has advantage over earlier PIC family in the pin assignment. Most of the pin are remappable to easier the design of the whole circuit. Furthermore, more communication module can be use compared to earlier family of PIC if the pins are available. It comes with 5 timer and internal Real-Time Clock.

Table 3.1: PIC24FJ64GB002 Specification

Architecture	16-bit
Program Memory	64KB
CPU Speed	16 MIPS
Timer	5 x 16 bit

USB	1, Full Speed, USB OTG 2.0
Digital Communication Peripherals	2-UART, 2-SPI, 2-I2C
I/O Pins	21
RAM bytes	8192

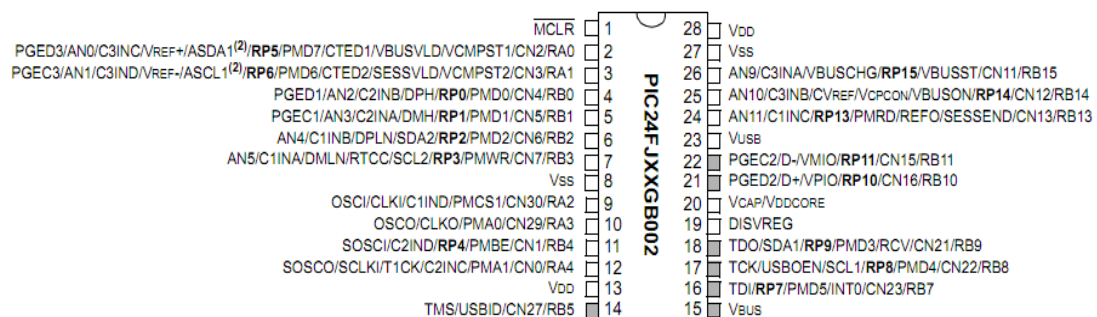


Figure 3.8: Pin Diagram of PIC24FJ64BG002

3.8 Matrix Keypad

To enter phone number and change setting, it is essential to have a keypad for this project. Matrix keypad is the most common keypad available due to its cheap and easy production. Compared with a SPI output keypad, the price of a SPI output keypad can be as high as five times of normal matrix keypad. Figure 3.8 and Figure 3.9 show the figure of 4x4 matrix keypad and internal connection respectively.



Figure 3.9: A 4x4 Matrix Keypad

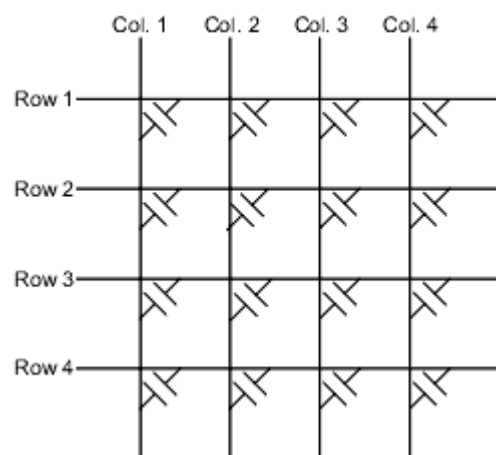


Figure 3.10: Internal Connection of a 4x4 Matrix Keypad

To read from the keypad, microcontroller has to do scanning through rows and column to read the key. Another option is using a keypad decoder which will be easier but a more costly solution. To stick with low cost solution, decode using microcontroller is preferable choice.

3.9 LCD Display

LCD is useful to display setting available so that setting can be changed on the project. LCD makes the project more user friendly and easier to understand on which operation is currently going on. A 2x16 LCD display is more than enough to display the information and changing setting. Figure 3.10 shows a 2x16 LCD display and table 3.1 shows the connection pin of the LCD.



Figure 3.11: A 2x16 LCD Display

Table 3.2: LCD Pin Table

Pin Number	Name	Description
1	V _{SS}	Negative supply for LCD
2	V _{CC}	Positive supply for LCD
3	V _{EE}	Contrast Adjustment for LCD
4	RS	Register Select (0: Command, 1: Data)
5	R/W	Read or write selection (0: Write, 1: Read)
6	E	Enable Signal
7	DB0	Data Bus 0
8	DB1	Data Bus 1
9	DB2	Data Bus 2
10	DB3	Data Bus 3
11	DB4	Data Bus 4
12	DB5	Data Bus 5
13	DB6	Data Bus 6
14	DB7	Data Bus 7
15	LED+	Positive supply for LCD back light

16	LED-	Negative supply for LCD back light
----	------	------------------------------------

3.10 MPLAB IDE

MPLAB IDE is a free programming tool which is use together with PIC microcontroller developed by Microchip. This software has the ability to build the files require to program into microcontroller for project development as well as in circuit debugging.

MPLAB IDE will be use to write the program for digital security locking system. In this project, program of USB based GSM security system is developed using MPLAB IDE C30 compiler which the program is written using C language. Assemble language may have faster and more compact code but due to the complexity of the code in large program, it is hard to handle and C language is preferred over assembly language. There will be few major functions inside the program which is reading and decoding keypad, write to LCD display, read from sensor when interrupt occurs and send SMS using GSM terminal when anything happen.

MPLAB will be using together with PICKIT 2 to program the microcontroller. PIC24FJ64GB002 can only perform in circuit debugging when use with newer version of PICKIT which is PICKIT 3. However, the cost of PICKIT 3 cost around RM360 which is very expensive and it is beyond the budge. Thus, PICKIT 2 will be use for programming purpose. Debugging of the program will be done manually using test and trial method.

3.11 EAGLE Layout Editor

Eagle Layout Editor simply means Easily Applicable Graphical Layout Editor. It is software developed by Cadsoft. Eagle Layout Editor will be use to develop PCB layout for digital security locking system. Eagle Layout Editor has a lot of component to choose from and it has auto route function to express the PCB layout development. However, auto route function may not be suitable in this project but it may be a good guidance for new user.

3.12 Schedule of FYP 2

The schedule of FYP 2 starts with development of USB based GSM security system. After the focused part is done, the project will then integrate with RFID based locking system to create a digital security locking system. After that, whole project after integration will be built on PCB and a model is built to represent the house.

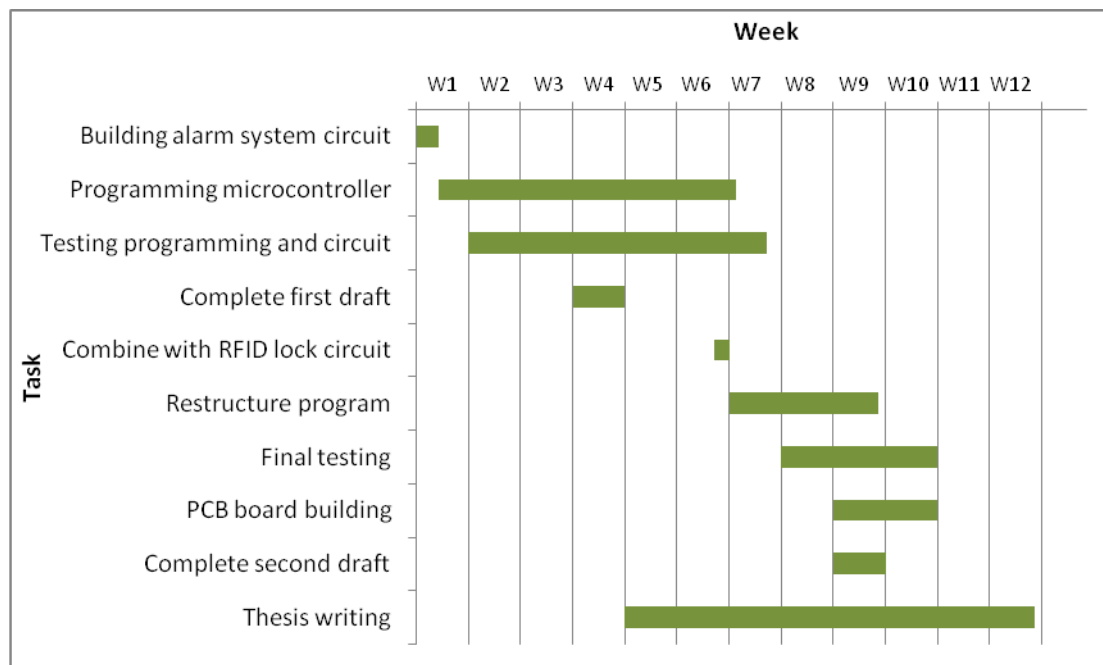


Figure 3.12 Gantt Chart of FYP Part 2

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

In this chapter, there will be two major parts which is the software part and the hardware part. The software part includes the discussion of program codes as well as step by step system configuration while the hardware part includes the schematic and PCB drawing. Discussion of program codes include embedded generic USB host, LCD as well as matrix keypad.

4.2 Software Implementation

4.2.1 Software Language

Due to complexity of the program and the effectiveness of C language in MPLAB C30 compiler, C language is preferred over assembly as the program code is expected to be very long. There will be some long string appear in the program and with the advantages of C language in comparing the string, we can easily writing out the code and combine few of the part together using C language program.

4.2.2 USB

4.2.2.1 USB System

USB is a type of serial communication which is more intelligent than previous serial communication such as RS-232. In RS-232 interface, the content of data is treated as a piece of data regardless what types of data it is. In the USB, each data is categorized and the types of transfer depending of the data types. However, the data is transferred between host and device(s). Typically a host is a computer with host controller hardware and software and a device is a peripheral with device controller hardware and firmware.

Host is responsible to detect attachment of device and assign each device attached a driver. These drivers manage the communication between host and devices. When device is attached, host will perform enumeration in which host and device are exchanging information. Host will request device description which acts like an identity of the device. During the enumeration process, the device is assign with a device address 0.

4.2.2.2 USB Embedded Host System

USB does not allow direct communication of host to host and device to device. When data transfer between two devices such as camera and printer is needed, it is inconvenience to have a computer to do the job. An USB embedded host is needed to solve the problem. Unlike USB host of computer, USB Embedded host only has limited capability and support certain device classes. Most of the embedded host do not support Hubs structure as computer USB host does which is part of the limitation of USB embedded host. USB embedded host is design for specific purpose and to meet the minimum requirement of data transfer.

4.2.2.3 USB On-The-Go (USB OTG)

To perform data transfer between two USB devices, one of them must be able to perform the role as the host and another as the device. On PC, the driver of USB host is complicated and with many function. However, the Embedded USB host has only limited function due to its memory size. USB has many classes and one of them is CDC (Communication Device Classes). Communication device classes are suitable for communication devices that utilized AP command.

There are few USB host driver provided by Microchip which is CDC, charger, mass storage, HID, printer and generic type driver. There are two types of driver we can choose from which is CDC and generic host. However, CDC embedded host is not suitable as mobile phone is not just a communication device but also a mass storage device. To deal with this problem, a generic embedded host driver is used instead of CDC host. Apart from combination of CDC device and mass storage device, mobile phone may fall in other category which is neither CDC nor mass storage.

When these devices do not falls to one of the class stated above, a generic host will be the last hope. A generic host has the basic ability of USB host which is managing the flow the data and at the same time, enabling both devices to communicate with each other. After some experiment, CDC classes host driver cannot support the mobile phone that is use in the experiment. On the other hand, generic driver can support most of the mobile phone in which the mobile phone can act like a GSM terminal.

4.2.2.4 USB Device Classes

Unlike older serial communication interface, USB categorized types of data transfer according to device classes. These device classes increase with the version of USB increase. There are some popular classes which are commonly used and Table 1.2 shows the popular device classes.

Table 4.1: USB Device Classes

Classes	Description	Example
0x01h	Audio	Speaker
0x02h	Communication Device Classes (CDC)	Modem
0x03h	Human Interface Device (HID)	Mouse
0x07h	Printer	-
0x08h	Mass Storage	Flash Drive
0x09h	Hub	-

When some of the composite device which has overlapped classes that does not belongs to any of the group above, typical a generic host driver is needed to solve the problem. Generic drivers enables the use of control, isochronous, bulk and interrupt transfer using driver specified application programming interface (API). A mobile phone may not be belongs to CDC classes since it includes modem and mass storage.

4.2.2.5 USB Enumeration

There are six steps in enumeration process which is Powered, Default, Address, Configuration, Attach and Suspend process. During a device attached to the host, the typical process in list in the table.

Table 4.2: Enumeration Steps

No	Step	Description
1	Power up device	The device attach is drawing power from the host up to 100mA.
2	Detect new device	The host detects the device by monitors the D+ and D- signal lines.
3	Host learn new device	Interrupt occurs and host check the new device attached.
4	Detect device speed	Host determine the speed by monitoring both D+ and D- line.
5	Reset device	Port is reset using the new configuration.
6	Detect if support high speed	If the device supports full speed, host will then try to use high speed if available.
7	Establish signal path	After port resume from reset, the device is now in default state. The device is ready to response from the host.
8	Learn Descriptor	Host request device descriptor from the device using endpoint 0.
9	Assign address	After getting device descriptor, the device is reset and gives a new address.
10	Learn device ability	The host now request for full device descriptor such as maximum packet size, number of configuration and some basic information.
11	Assign device driver	The host will search for any match in the device management and assign to it.
12	Select configuration	If device configuration is available, the device driver request configuration from the device.

4.2.2.6 USB Files in Programming Part of the Project

Microchip provided USB host driver demo project and USB client driver project in their website for use with their development board. These files contain host driver file with can be use with other device if proper modification are made. Writing an own USB host driver may be time consuming and great knowledge in USB structure, as well as USB operation is needed. Thus, several files are taken from the demo which is then include in the project to make the USB host driver project easier. There are many categories of host driver files and the most suitable host driver will be included in the project. Apart from choosing host driver, defining types of transfer and targeted peripheral list (TPL) must be done to make the project working. Some

major files included in the project are as following. Table 4.1 shows some important file in the project and Figure 4.1 shows an example of target peripheral list.

Table 4.3: USB Files in Programming Part of the Project

File name	Comment
usb_config.c	This file defining types of transfer allowed, current needed by USB client and attach debounce interval. Types of transfer allowed must be select correctly or else the project cannot work properly or fail to work. This file configures the USB stack.
usb_config.h	This file contain target peripheral list (TPL) in which defining the client supported by the host. This file configures the USB stack.
usb_host_generic.c	This is generic USB embedded host driver file in which the operation of generic host is enabling by this file. Each type of host driver has different operation allowed which defining by this file.
usb_host.c	This file is the main USB embedded host driver file which use to control most of the USB operations. These operations include detection of device attached and detached, configuring speed of transfer, performing USB enumeration, and start of transferring data between host and client. All USB embedded host driver must have this file included in the project. This file does not provide class support.

```
// *****
// USB Embedded Host Targeted Peripheral List (TPL)
// *****

USB_TPL usbTPL[] =
{
    { INIT_VID_PID( 0x0fceul, 0xd016ul ), 0, 0, {0} } // k750
,
    { INIT_VID_PID( 0x0fceul, 0xd0d4ul ), 0, 1, {0} } // C902
,
    { INIT_VID_PID( 0x0fceul, 0xd039ul ), 0, 2, {0} } // K800
,
    { INIT_VID_PID( 0x0fceul, 0xd0b7ul ), 0, 3, {0} } // k770
,
    { INIT_VID_PID( 0x0421ul, 0x0025ul ), 0, 4, {0} } // 5300
};
```

Figure 4.1: Example of TPL List

4.2.2.7 USB Operation

To use the USB function inside microcontroller, the function USBHostInit must be called before any USB operation can be performed. USBHostInit will initiate the USB host stack for USB operation. As soon as USBHostInit is successfully performed, USBHostTasks should be called to perform hardware initialization as well as perform USB host task. USBHostTasks include checking on device attach or detach and perform enumeration. USBHostTasks must be called regularly to make sure USB operation is working correctly. If USBHostTasks did not call regularly, device detachment or attachment will not be detected by microcontroller and USB operation might fail. Table 4.2 shows the experimental result of time for enumeration for different calling interval of USBHostTasks.

Table 4.4: Effect of USBHostTasks Calling Interval on USB Enumeration

Calling Interval (ms)	Enumeration Time (s)
1000.000	30.0
500.000	15.0
250.000	7.8
125.000	4.0
62.500	2.6
31.250	2.0
15.625	2.0

USB enumeration will take minimum 2 s to complete the whole process. For calling interval of 31.250 ms, the system will have periodic lag on displaying text on LCD and for 15.625 ms, the system can hardly display text on LCD screen. Calling

interval of 62.500 ms is the better choice for system performance and enumeration time.

4.2.2.8 Device Endpoint

Each of the data is travel from or to device endpoint. Device endpoint is served as a buffer for the data to be transferred to the host and buffer for data received from the host. When USB stack is initializes, each endpoint is assign with transferred types using and Endpoint 0 is especially for enumeration process. Every USB device must at lease have endpoint 0 for the host to obtain information from the device. Each endpoint can have data in both directions. For example endpoint 0 IN means data transfer from device to host in endpoint 0 and endpoint 0 OUT will means data transfer from host to device. Endpoint can have the number range from 0 to 15.

4.2.2.9 USB Transfer Types

As mention of earlier chapter, there are four types of transfer in USB. Control type transfer is supported by all the USB device of else the enumeration will fail. Three other transfer types is use for different purpose and both USB host and USB device must aware which types of transfer will going to take place or the transfer of data will fail.

Referring to embedded generic USB host driver provided by Microchip, generic host will utilize Endpoint 0 for enumeration and Endpoint 1 for data transfer. In the code, each endpoint has different types of data transfer. Figure 4.2 shows the transfer type on different endpoint.

```
// Section: USB Endpoint Transfer Types

#define USB_TRANSFER_TYPE_CONTROL      0x00    // Endpoint is a control endpoint.
#define USB_TRANSFER_TYPE_ISOCHRONOUS  0x01    // Endpoint is an isochronous endpoint.
#define USB_TRANSFER_TYPE_BULK         0x02    // Endpoint is a bulk endpoint.
#define USB_TRANSFER_TYPE_INTERRUPT    0x03    // Endpoint is an interrupt endpoint.
```

Figure 4.2: Transfer Types of Different Endpoint

To use another endpoint other than the default Endpoint 1 for data transfers, another function is written to use Endpoint 2 for data transfer. Besides that, a function is written to test on which transfer types that the device supported. After the function has finish testing on types of transfer supported, the next transfer of data will utilize the type of transfer that is supported. Figure 4.3 shows the coding for testing transfer types available and begin to transfer immediately after the test. Figure 4.4 show the process of testing transfer types.

```
while (USBHostGenericTxIsBusy(deviceAddress));
RetVal = USBHostGenericEP2Write(deviceAddress, buff, 3);
if (RetVal == USB_SUCCESS){
    EP = 1;
}
else{
    EP = 0;
}

if (currstate == 0){
    while(USBHostGenericTxIsBusy(deviceAddress));
    if(EP == 1){
        RetVal=USBHostGenericEP2Write(deviceAddress, buff0, 11);
    }
    else{
        RetVal=USBHostGenericWrite(deviceAddress, buff0, 11);
    }
    currstate = 1;
    delay(100);
}
if (currstate == 1)
{
    while(USBHostGenericRxIsBusy(deviceAddress));
    if (EP == 1){
        RetVal=USBHostGenericEP2Read(deviceAddress,Home, sizeof(Home));
    }
    else{
        RetVal=USBHostGenericRead(deviceAddress,Home, sizeof(Home));
    }
}
```

Figure 4.3: Code to Test Transfer Types and Initiating Data Transfer

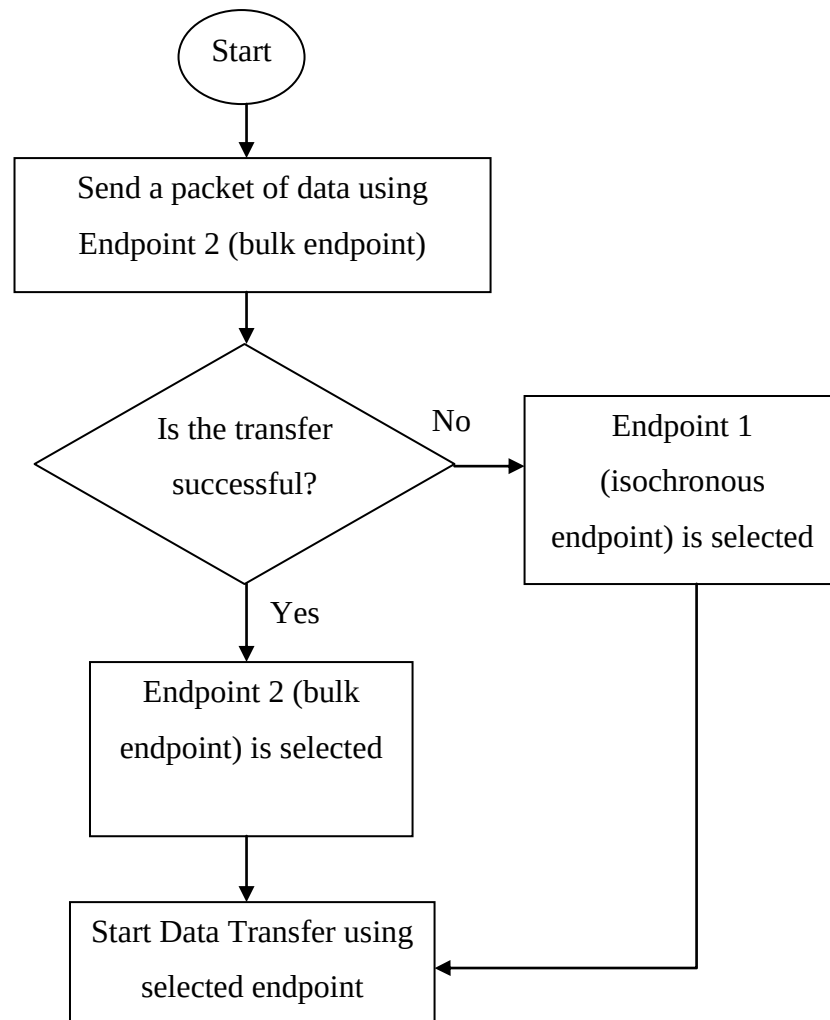


Figure 4.4: Flow Chart of Testing USB Transfer Type and Initializing Data Transfer

4.2.2.10 Generic Host Driver Write Function

Figure 1 shows the default USB write function in the Microchip generic host driver.

```

BYTE USBHostGenericWrite( BYTE deviceAddress, void *buffer, DWORD length )
{
    BYTE RetVal;

    // Validate the call
    if (!API_VALID(deviceAddress)) return USB_INVALID_STATE;
    if (gc_DevData.flags.txBusy) return USB_BUSY;

    // Set the busy flag and start a new OUT transfer.
    gc_DevData.flags.txBusy = 1;
    RetVal = USBHostWrite( deviceAddress, USB_OUT_EP|USB_GENERIC_EP, (BYTE *)buffer, length );
    if (RetVal != USB_SUCCESS)
    {
        gc_DevData.flags.txBusy = 0;    // Clear flag to allow re-try
    }

    return RetVal;
} // USBHostGenericWrite

```

Figure 4.5: Generic Host Driver Write Function

The function will first check for valid machine state as well as any USB host is still busy to transfer data to device. If machine state is not valid or USB transfer is still busy, the function is terminated and returns the error code. When the entire test passes, the write flag will be set high to indicate a write is performing at that time. Next, the write process is started by calling USBHostWrite function in host driver. USBHostWrite function takes four variables which are the device address, endpoint number, pointer to the data and the length of the data to be transferred. The default endpoint is endpoint 1 which is for isochronous transfer. By replacing USB_GENERIC_EP in the figure to other value, different types of transfer can be used as different endpoint has its own transfer types. Another function called USBHostGenericEP2Write is written same as USBHostGenericWrite function with underline part changed to USB_GENERIC_EP2. This function utilizes endpoint 2 which is bulk transfer instead of endpoint 1 isochronous transfer for USB data transfer. Note that USBHostWrite will return result of the USB write function. This result code is defined in USBHost.h and Table 1 shows the result code.

Table 4.5: USB Result Code

Result Code	Description
USB_SUCCESS	Write started successfully.
USB_UNKNOWN_DEVICE	Device with the specified address not found.
USB_INVALID_STATE	We are not in a normal running state
USB_ENDPOINT_ILLEGAL_TYPE	Must use USBHostControlWrite to write to a control endpoint.
USB_ENDPOINT_ILLEGAL_DIRECTION	Must write to an OUT endpoint
USB_ENDPOINT_STALLED	Endpoint is stalled. Must be cleared by the application.
USB_ENDPOINT_ERROR	Endpoint has too many errors. Must be cleared by the application.
USB_ENDPOINT_BUSY	A Write is already in progress.
USB_ENDPOINT_NOT_FOUND	Invalid endpoint.

With the USB result code, we can determine the transfer types which are supported by the device and use the transfer types to communicate between host and device.

4.2.3 Emulating Data EEPROM

This project need memory to store phone number, SMSC number as well as password so that the device is customizable and flexible to use. It is unacceptable for user to program the whole device again if they need to change password or target phone number.

In PIC24, there are no EEPROM which available in past family of PIC such as PIC18, PIC16 or PIC12. PIC uses flash memory as program memory. One of the methods is using table lookup to use the internal flash memory or using external memory storage such as Data EEPROM or memory card. In the design, the pin is

very limited as the PIC in the project has just twenty eight pins and nineteen pins are capable to become I/O pins. To integrate the microcontroller with LCD, keypad, USB connector and RS-232 converter, all the pins become unavailable and hence, external memory is not an option.

As a result, Emulating Data EEPROM which utilized internal flash memory will be a better option for its fast operation. PIC24 has twenty four bits width data size in which emulating data EEPROM will take eight bits as address and sixteen bits as data. This eight bits is not a real address but it act like a tag for the data. First Data EE Address is use store page status and it is not available for storing user data. Table 4.3 shows the data memory structure and Table 4.4 shows the user data storage location of this project.

Table 4.6: Data Memory Structure

Page Address	Data EE Address	Data EE Data
This is the real address in the program memory.	This is tag for data.	User data.

Table 4.7: User Data Storage Location

Data EE Address	User Data
1-110	Card unique ID. Each card uses 10 addresses to store data in word.
200	User password. Uses 6 addresses space.
210	Administrative password. Uses 6 addresses space.
220	SMSC number. Uses 12 addresses space. As for PDU modes, an 'F' is added after the number before it is encoded.
240	Target phone number. Uses 12 addresses space. As for PDU modes, an 'F' is added after the number before it is encoded.

4.2.4 Decoding keypad

Matrix keypad is the most common keypad available due to most cost effective device. However, a hardware decoder is needed to decode the keypad or keypad can simply decode inside microcontroller by writing some function.

When a key is pressed, one wire from a column will touch with another wire from a row. To decode the wire, either row or column will be the input. If column is the input, the row will be the output and vice versa. Now, column is to be the output and row will be the input. Initially the entire row connected to the microcontroller is set to high.

Next, scan all the port connected to column to detect any port is high. Whenever port connected to column is high, entire row is set to low. Then, each row is turn on one by one to scan which row is pressed so that the location of the button can be determined. In between turning on and off, delay is needed to avoid detection of wrong key pressed. After the location of the key is determined, the program will return the value and wait until user leave their hand from the keypad. Again, delay is needed to avoid the function to go inside the loop again to test for key pressed when user leave their hand from keypad. Figure 4.5 and Figure 4.6 show the flow chart of decoding keypad.

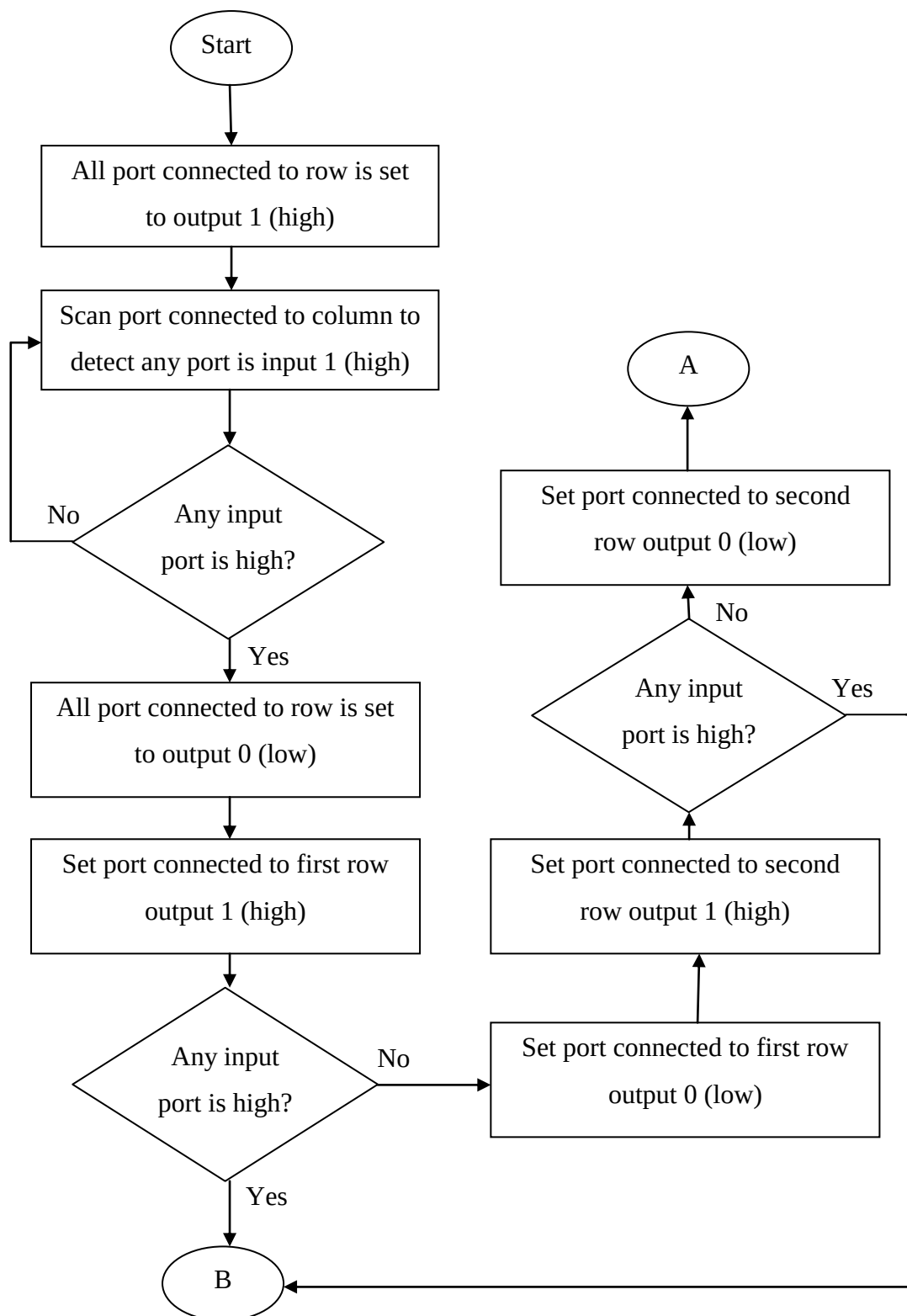


Figure 4.6: Flow Chart of Decoding Keypad Part 1

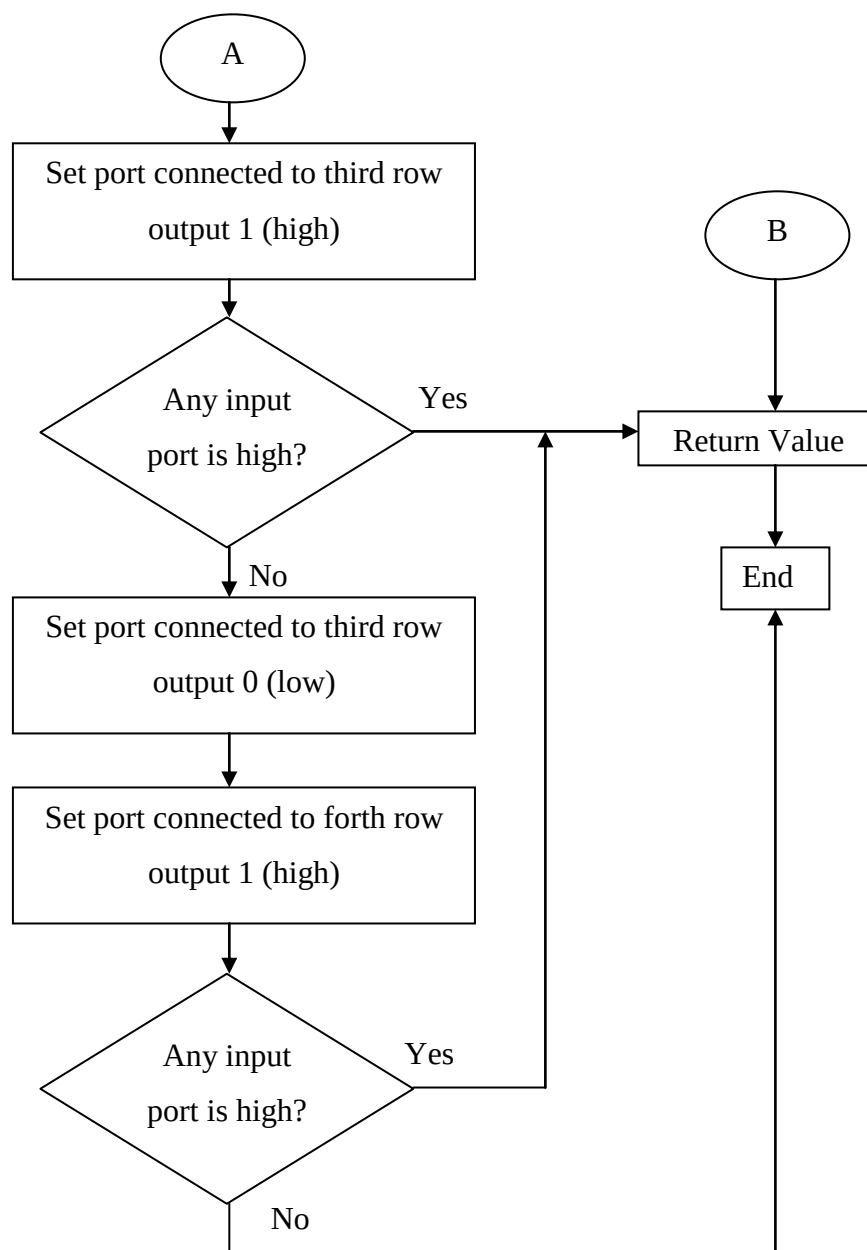


Figure 4.7: Flow Chart of Decoding Keypad Part 2

4.2.5 Program on LCD Display

LCD is useful to show information and change setting. Before an LCD can display words or symbol, it has to be first initialize with proper configuration for it to display correctly. There are some sequences that need to be done for LCD initialization. Figure 4.7 shows initialization process for four bits mode of the LCD in this project.

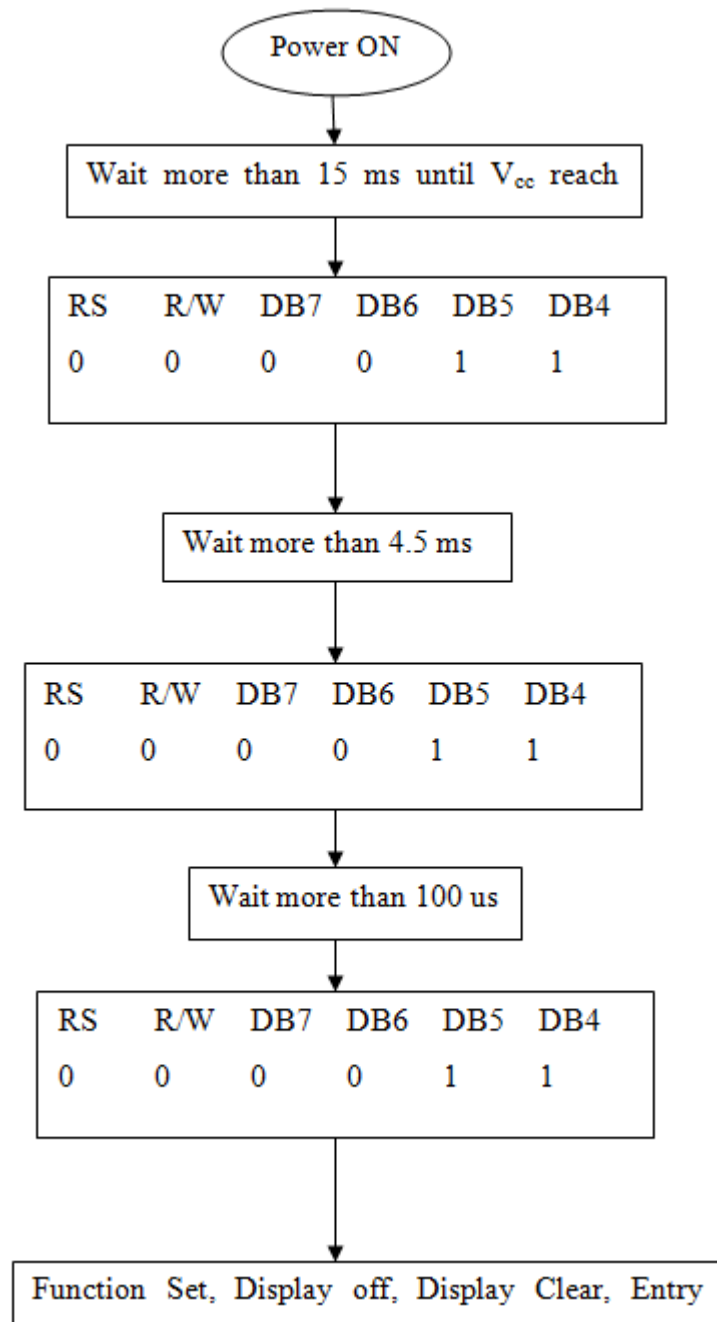


Figure 4.8: LCD Initiation Process

Suppose the LCD is to display two rows and showing the location of the cursor, function and entry mode had to be set to ensure proper operation. The LCD is use in four bits mode to reduce the I/O pins required by the LCD. To do that, LCD is configured to four bits mode operation and Figure 4.8 shows the coding for function set and entry mode set.

```

/* Function:  initlcd
   Overview:  This routine initialize port for lcd and configure
              lcd for system initialization */
void initlcd(void)
{
    TRISB = 0xFFFF0;          // Output RB0-RB3
    TRISBbits.TRISB13 = 0;    // Output RB13
    TRISBbits.TRISB14 = 0;    // Output RB14
    delay(1);
    writelcd4b(0,0x02);        // 4bit mode
    delay(8);
    writelcd8b(0,0x28);        // 4bit mode
    delay(8);
    writelcd8b(0,0x0E);        // turn on display, on cursor
    delay(8);
    writelcd8b(0,0x06);        // Increment mode
    delay(8);
    writelcd8b(0,0x01);        // Clear screen
    delay(8);
}

```

Figure 4.9: Program Code for Function Set and Entry Mode Set of LCD

To access LCD using four bits mode operation, each data bytes sent to LCD has to be done in two times of four bits each of the time. Higher nibble is sent before the lower nibble does. To achieve this operation, the data is first saved into temporary data space. After that, the data are performing with logical conjunction to obtain the lower nibble and higher nibble. When the data are ready, it will send to LCD with higher nibble follow by lower nibble. Register select is to be high when writing data to LCD and low when giving command to LCD. Figure 4.9 shows the program code for sending data to LCD.

```

/* Function:   writelcd8b
   Overview:   This routine writes a byte of data to lcd.
               Data is send to lcd in 4bit mode */
void writelcd8b(unsigned int fnc, char text)
{
    accesslcd();           // Switch to LCD
    delay(1);
    char temp = text;      // Save data inside temp
    temp = temp & 0xF0;    // Take upper 4 bit
    temp = temp >> 4;      // Right shift to output data
    text = text & 0x0F;    // Take lower 4 bit
    PORTB = temp;          // Output upper 4 bit first
    if(fnc)                // Check for intruction or data
        PORTBbits.RB13 = 1;
    else
        PORTBbits.RB13 = 0;
    PORTBbits.RB14 = 1;    // Enable
    delay(1);
    PORTBbits.RB14 = 0;    // Disable
    PORTB = text;
    if(fnc)
        PORTBbits.RB13 = 1;
    else
        PORTBbits.RB13 = 0;
    PORTBbits.RB14 = 1;
    delay(1);
    PORTBbits.RB14 = 0;
}

```

Figure 4.10: Program Code of Writing Data to LCD

4.3 Program overview

The program begins at initiating the USB peripheral. As soon as USB peripheral is started, it will check for any USB device plug into the USB type-A female connector. If there is no device detected, it will continue looping to search for device attached. On the other hand, if a USB device is attached, it will then check whether the device is supported. If the attached device is supported, the program will now ready to read the input from door sensor. Any unauthorized opening of door will cause the alarm to be activated and at the same time, a SMS is sent to warn the user. Figure 4.10 shows the program flow chart of the main program and Figure 4.11 shows the LCD screen of the main program.

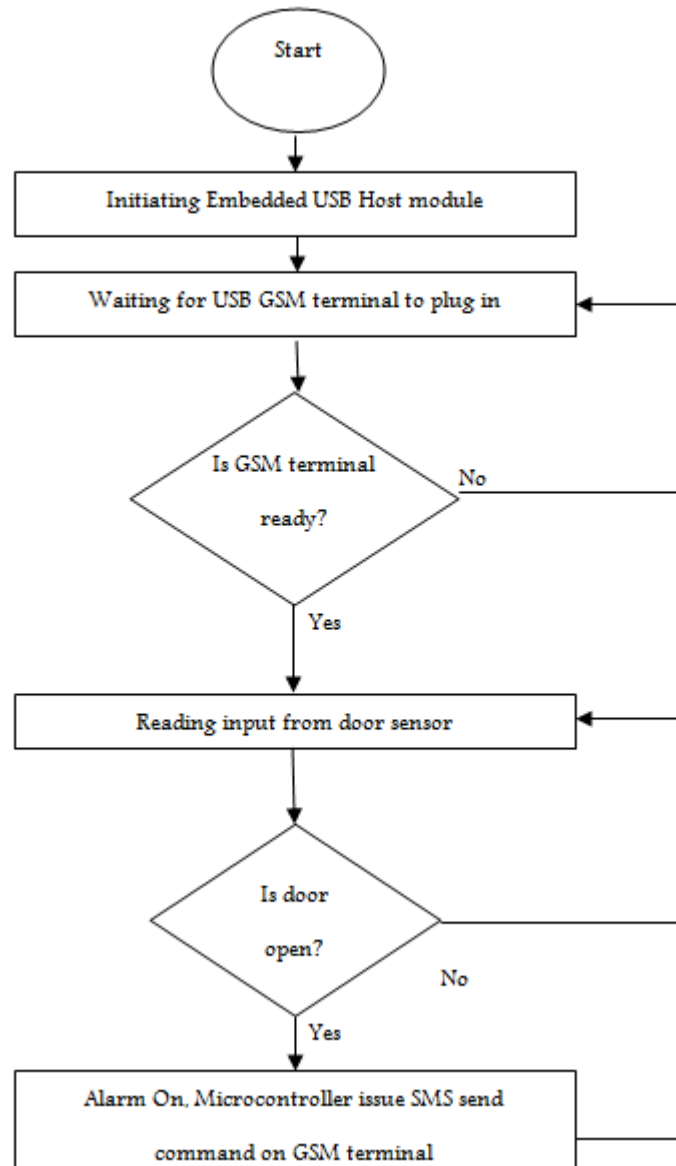


Figure 4.11: Program Flow Chart of USB Based GSM Security System

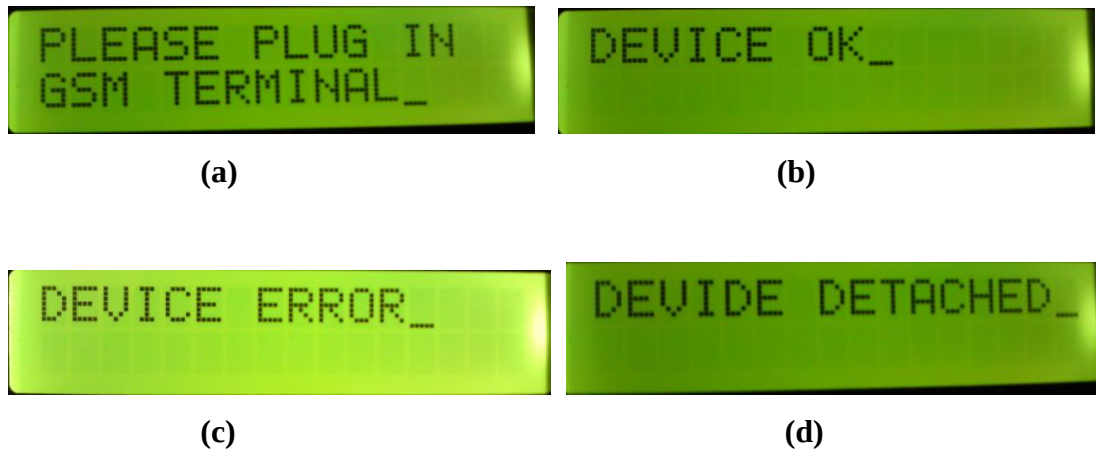


Figure 4.12: LCD Display on (a) Start up, (b) Supported Terminal Attached, (c) Unsupported Terminal Attached, (d) Terminal Detach

4.3.1 Option menu

Option menu can be accessible by pressing 'B' on the keypad. When pressing 'B' on keypad, user will be prompt for two set of password. First password will be user password where it uses to unlock door and second set of password is the administrative password which is use when user change setting. Table 4.5 shows the option menu for the program and Figure 4.12 and Figure 4.13 show the menu flow chart.

Table 4.8: List of Option Menu

Option key	Description
3	This option is use to change user password. User password is in 6 digits length and use for unlocks door. When this option is selected, user will be prompt to enter 6 digits of user password. If user accidentally press the wrong button, the key 'C' is use to re-enter the password again. A message "DONE" will be displayed on LCD for successful changing of user password.
4	This option is use to change administrative password. Administrative password is also in 6 digits length and use when changing setting. When this option is selected, user will be prompt to enter 6 digits of administrative password. If user accidentally press the wrong button, the key 'C' is use to re-enter the password again. A message "DONE" will be displayed on LCD for successful editing of administrative password.
5	This option enable user to change the target phone number for break in case occurs. The length of target phone number is 11 digits which including the country code. Example of phone number in Malaysia is "60123456789". A message "DONE" will be displayed on LCD for successful changing target phone number.
6	This option lets user change SMSC number. Different service provider will have different SMSC number. This is useful when user changes service provider of the GSM terminal. The length of SMSC number is same as phone number which is 11 digits. A message "DONE" will be displayed on LCD for successful editing of SMSC number.

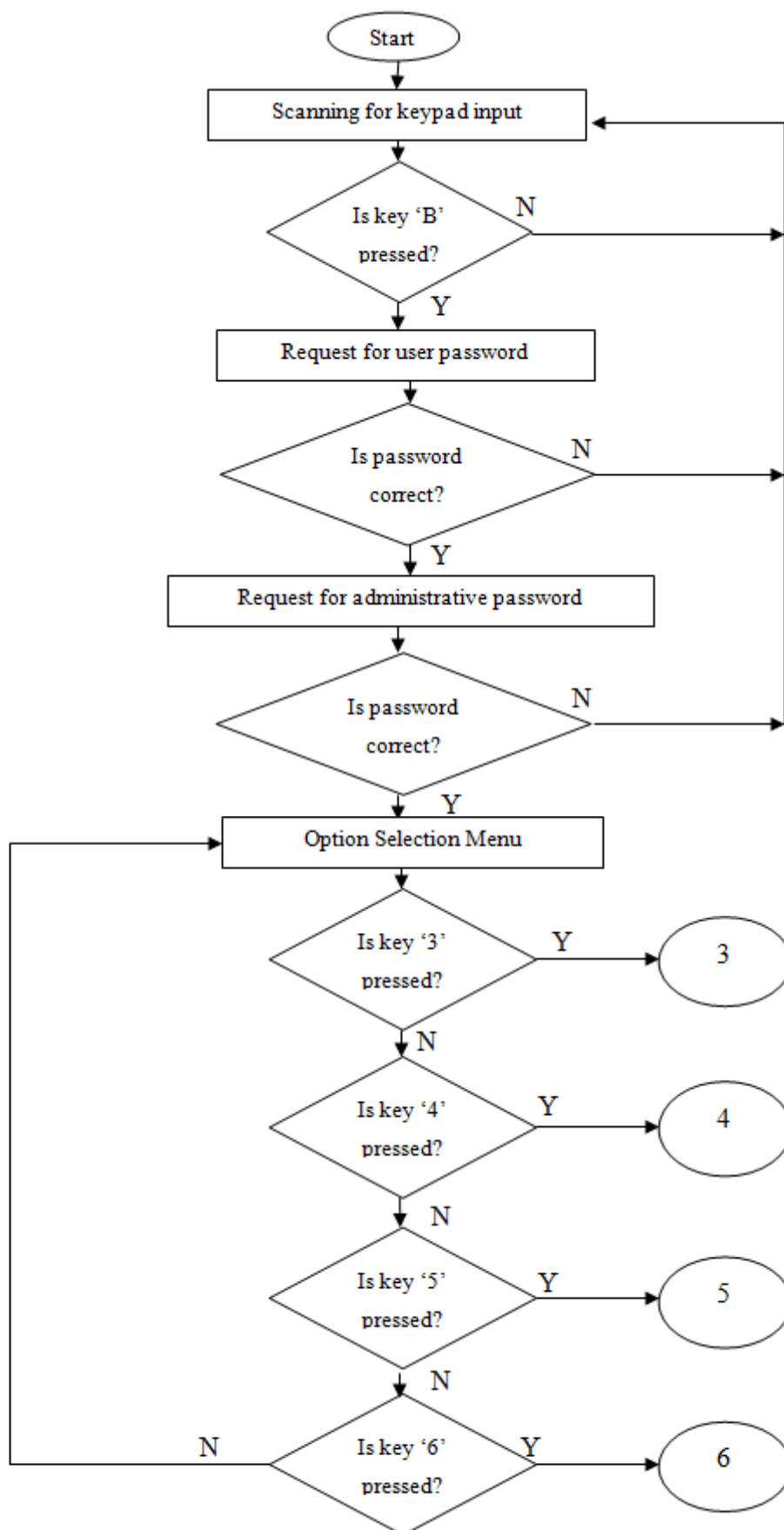


Figure 4.13: Menu Flow Chat Part One

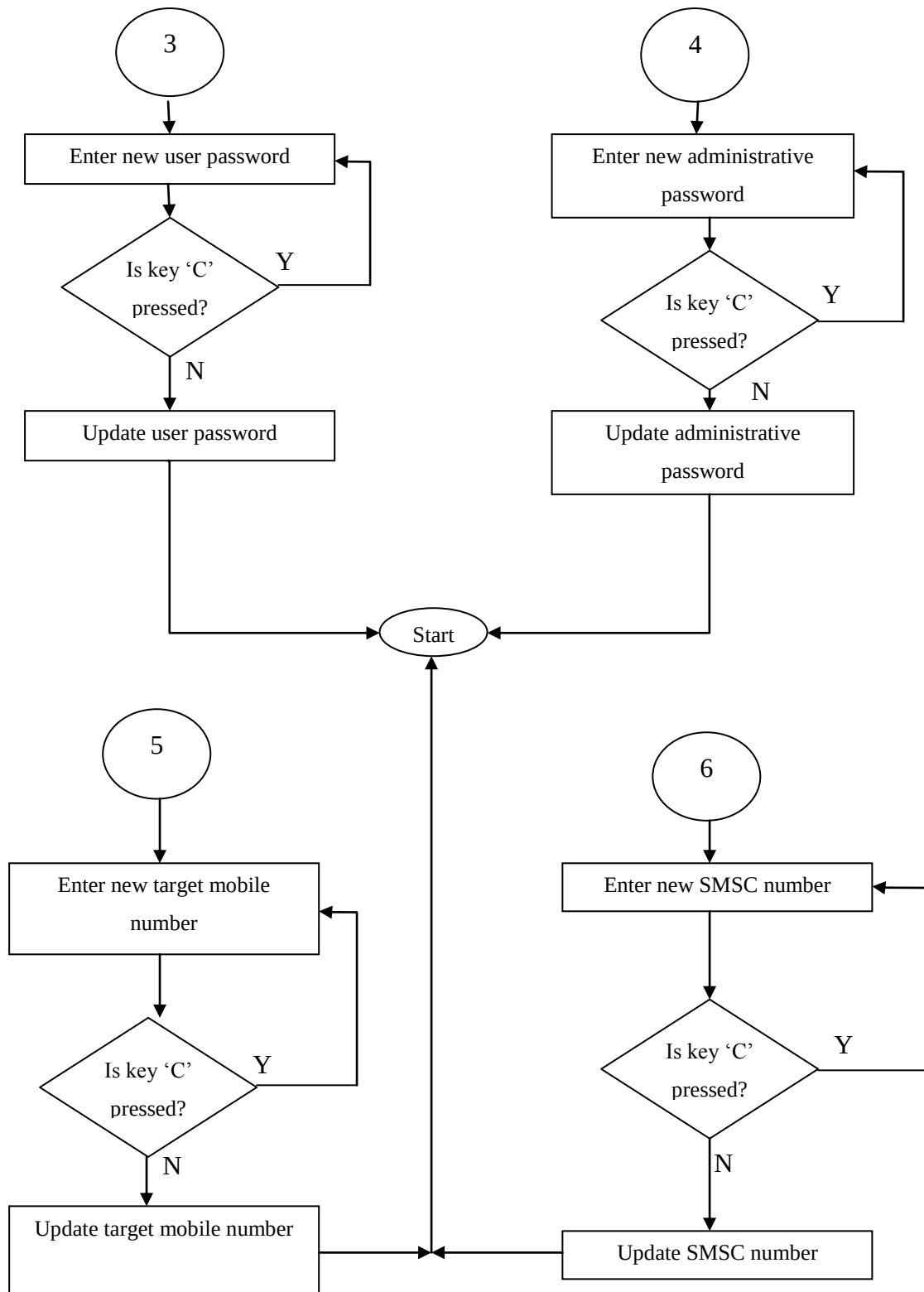
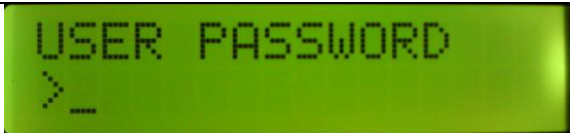
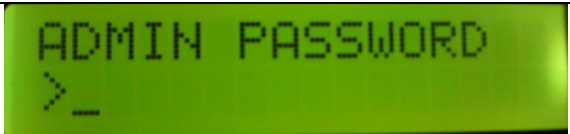


Figure 4.14: Menu Flow Chat Part Two

4.3.2 Detailed Operation on Changing Password

The operation of changing password will be show in detail steps by step together with display on LCD to explain how password changing process is carry on. Table 4.6 shows the detailed steps to change password.

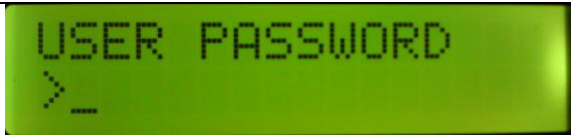
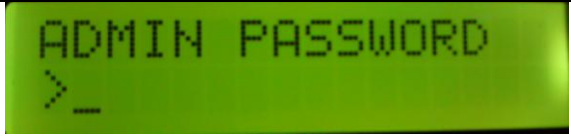
Table 4.9: Detail Operation on Changing Password

Detail Operation	Display shown on LCD
When user presses 'B' during normal operating condition, user will be prompt for user password. Password to be entered is in six digit lengths.	
After user password verification is successful, user will then prompt again for administrative password. The administrative password is in six digits length.	
If both passwords are entered correctly, user will then redirect to option selection menu. There are totally seven options available to select from.	
Press '3' on keypad for changing user password and '4' for changing administrative password. LCD will show enter password after the key is pressed.	
If user accidentally pressed the wrong key, 'C' on keypad is use to enter password from the beginning.	
When user successfully entered all the passwords, a message will show on LCD to indicate operation successful.	

4.3.3 Detailed Operation on Changing Target Phone Number

It is important for a GSM based security system able to change target phone number for emergency case. Detailed operation of changing target phone number is shown in Table 4.7.

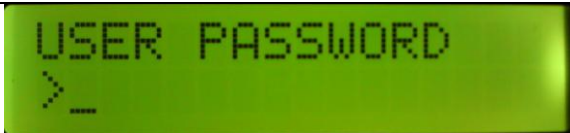
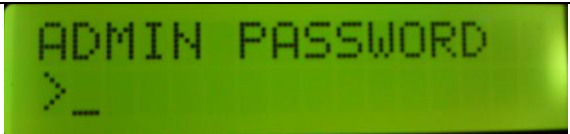
Table 4.10: Detail Operation on Changing Target Phone Number

Detail Operation	Display shown on LCD
When user presses 'B' during normal operating condition, user will be prompt for user password. Password to be entered is in six digit lengths.	
After user password verification is successful, user will then prompt again for administrative password. The administrative password is in six digits length.	
If both passwords are entered correctly, user will then redirect to option selection menu. There are totally seven options available to select from.	
Press '5' on keypad for changing target phone number. Phone number to be entered is in ten digits length and example is 60123456789.	
If user accidentally pressed the wrong key, 'C' on keypad is use to enter phone number from the beginning.	
When user successfully entered all the numbers, a message will show on LCD to indicate operation successful.	

4.3.4 Detailed Operation on Changing SMSC Number

By using PDU mode, SMSC number has to be entering manually by user. Wrong SMSC number can lead to message unable to deliver out and leave to whole system unprotected. Table 4.8 show the step needed to change SMSC number.

Table 4.11: Detail Operation on Changing SMSC Number

Detail Operation	Display shown on LCD
When user presses 'B' during normal operating condition, user will be prompt for user password. Password to be entered is in six digit lengths.	
After user password verification is successful, user will then prompt again for administrative password. The administrative password is in six digits length.	
If both passwords are entered correctly, user will then redirect to option selection menu. There are totally seven options available to select from.	
Press '6' on keypad for changing target phone number. Phone number to be entered is in ten digits length and example of SMSC number for MAXIS is 60120000015.	
If user accidentally pressed the wrong key, 'C' on keypad is use to enter phone number from the beginning.	
When user successfully entered all the numbers, a message will show on LCD to indicate operation successful.	

4.4 Results of Tested Possible Mobile Phone as GSM Terminal

To test the performance of generic host driver with GSM terminal, some phone has been taken into experiment. Before that, product number (PID) and vendor number (VID) must be obtain and insert into target peripheral list (TPL). Generic host will only support the device inside TPL table. Table 4.9 shows some result obtain from the experiment of few mobile devices.

Table 4.12: Test Result of Mobile Phones

Device Name	Enumerations	Send Message
Nokia 5300	Success	Supported, PDU and text mode
Nokia 5800 Express Music	Success	Supported, PDU and text mode
Nokia N95	Success	Supported, PDU and text mode
Sony Ericsson C501	Success	Supported, PDU mode only
Sony Ericsson C902	Success	Supported, PDU mode only
Sony Ericsson K750	Success	Supported, PDU mode only
Sony Ericsson K770	Success	Supported, PDU mode only
Sony Ericsson K800	Success	Supported, PDU mode only

The data in Table 4.9 show that most of mobile phone can communicate effectively with microcontroller using control transfer. Control transfer is use to perform enumerations and most of the device will support this types of data transfer. In the experiment that carried out, all the mobile phone listed in Table 4.9 successfully send out SMS by using either isochronous transfer or bulk transfer. Note that PDU mode is supported by all the phones use in the experiment and only Nokia phones in the experiment support text mode.

4.5 Hardware implementation

4.5.1 Hardware Placement

The main board which microcontroller is on it is to be placed inside the house while LCD and keypad is to be located on outside the house. With this placement, the main board can protected from attack to keep the security system working all the time while providing access to user outside the house. If the whole unit is place outside the house, people can easily damage the board and security system will offline and thus, leaving the house unprotected.

4.5.2 Circuit Diagram

This project includes development of hardware as well as software. A circuit is needed achieve the goal. USB based GSM alarm system is part of the digital security locking system and it consist of a microcontroller, a keypad, a LCD display, a sensor and as well as USB GSM terminal. Microcontroller is the core of the project and all components is connected to it and main task is process here. With limited pin of a 28 pin microcontroller, the project is done by utilize all the available port. Port A has five I/O pins and two will be use for RFID based locking system. Two I/O pins is connected to external crystal for more accurate operation. With only one Port A pins left, the pin is use to connect to magnetic sensor.

Port B has total 16 I/O port and most of the pin will be use for LCD and keypad. Four common pin will be share by LCD and keypad to maximize the utilization of I/O pins and these pins are RB0 to RB3. RB4, RB7, RB8 and RB9 are use for keypad for input detection pin. These pins cannot be shared to avoid any error in decoding keypad. RB13 and RB14 are connected to LCD as the Register Select and Enable bit of LCD. These pins must no share with any other component to avoid error. RB5 is use to connect to buzzer which is use to indicate as an alarm siren and RB15 is connected to a transistor and a relay to drive the magnetic lock on and off.

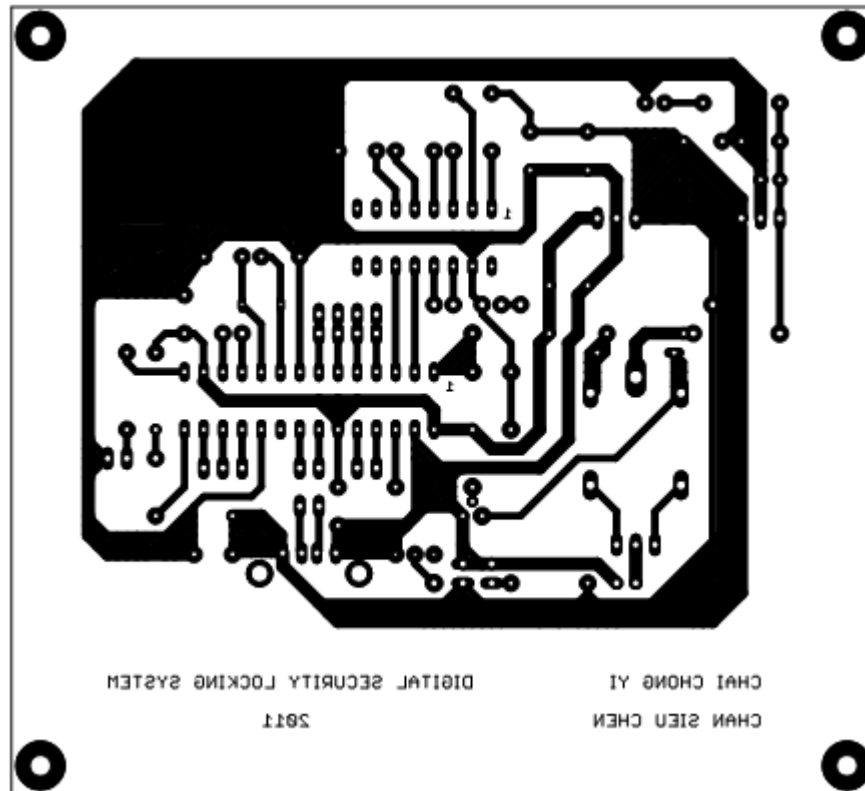


Figure 4.16: PCB Layout of Digital Security Locking System

4.6 Problem Encounter

Most of the project will not go smoothly where some problem may be encounter during the process. The first problem encounter in this project is the USB enumerations. Enumerations process is quite complex in which require deep knowledge of USB structure to make it right. As for the result, USB host driver release by Microchip is used in the project to cope with this problem.

Second problem arise when other GSM terminals is used in the project instead of Sony Ericsson C902. Microcontroller fails to send data to the GSM terminal which results in USB error. This problem is due to the endpoint use by embedded generic host driver provided by Microchip only utilize Endpoint 1 for data transfer which is isochronous transfer. This problem is soon solved by adding another transfer types to the embedded host which is bulk transfer that utilize Endpoint 2.

Third problem is on the input pin on microcontroller that connected to keypad as well as LCD. Some input pin must be properly pulled to ground to avoid any floating state when microcontroller tries to read the state of the input pin. Unexpected input will generated when the input of microcontroller is in floating state. High value of resistor is used to connect between these pin and ground to provide proper grounding.

Forth problem is regarding the voltage regulator. Initially the design is cascading two voltage regulator to provide 5 V and 3.3 V from a 12 V source. The 12 V source is first regulated to 5 V and the further down to 3.3 V. The 12 V supply is use to power magnetic lock and 5 V supply is use for MAX232 IC for RS-232 communication between RFID reader and microcontroller and USB device. As for 3.3 V supply, it is use in USB as well as microcontroller itself. Cascading two voltage regulators may work fine if the current drawn is large enough to avoid too low of dropout voltage which will cause the voltage regulator fail to work. When the circuit is placed on PCB, the voltage regulator fails to operate properly. As such, two different voltage regulators is use in which both of them are connected directly to source.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

As a conclusion, the aim and objective of this final year project had been achieved in which both RFID based digital locking system and USB based GSM security system have successfully integrated to form a digital security locking system.

This digital locking system is a low cost, low power, simple operation, small size, and standalone system. Digital locking system reads RFID tag or reads from keypad which will then unlock the door when the tag or password is matched. This system is able to change password, adding tag to access list, change the period the system is being blocked if password entered wrong for five times, change target phone number and change access types. Even the system is turned off, this information can still keep in memory and these setting will restored when system starts up next time.

Besides, this system can send SMS to user when security is breached or password is being entered wrong for five times. Alarm will be turn on whenever door is forced to open without first enter password or scan tag on the reader. Alarm can be turn off by enter correct password.

5.2 Limitation and Future Enhancement

The number of phone model that support AT command is considerable. However, most of them only support PDU mode instead of both PDU and Text mode. To decode the message in PDU mode, the program is complex. As such, if cannot decode the PDU string, microcontroller cannot response according to what message the GSM terminal has received. To enhance the project, it is desirable to write a program to decode PDU string so that this system can be controlled remotely.

To keep track on when and how door is unlocked, a Real-Time Clock is needed. There are real time clock inside the microcontroller but the precision is the most important thing. This can also be done through connecting the whole project to the computer and let the computer do the tracking.

Besides, this system can further enhance when combining with different types of sensor using wireless sensor such as Bluetooth sensor. All sensors using a single node will save all the pins required by wired sensor and a sensor network can be developed. Many different types of alert can be sent to user through SMS which makes the system even more powerful on monitoring surrounding.

At last, the system can also integrate with home automation system to create more complete home automation system. This system can provide home automotive as well as home security. User can keep track on most of the information at home when they are not around.

REFERENCES

- Ahmed, A., Ali, J., Raza, A. & Abbas, G. (2007). Wired vs wireless deployment support for wireless sensor networks. Retrieved August 9, 2010 from <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4142257&tag=1>
- Bodic, G. L. (2005). Mobile Messaging Technologies and Services SMS, EMS and MMS (2nd ed.). Chichester: John Wiley & Sons, Ltd
- Cao, S. Q. (2009). GSM Modem-Based Mobile Auxiliary Learning System. 2009 International Conference on Computational Intelligence and Software Engineering, 1, 1-4.
- Developer's Home. (2004-2010). Short Message Service / SMS Tutorial. Retrieved July 1, 2010, from <http://www.developershome.com/sms/>
- Gregory, J. (2007). A short history of burglar alarms. Retrieved August 1, 2010 from <http://ezinearticles.com/?A-Short-History-of-Burglar-Alarms&id=475162>
- Huang, H. P., Xiao, S. S., Meng, X. Y. & Xiong, Y. (2010). A remote home security system based on wireless sensor network and GSM technology. 2010 second international conference on networks security, wireless communications and trusted computing, 1, 535-538.
- Jasio, L. D. (2007). Programming 16-Bit Microcontroller in C - Learn to Fly the PIC24. Oxford: Elsevier Inc.
- Kim, Y. S., Kim, H. S. & Lee, C. G. (2005). The development of USB home control network system. 2004 8th international conference on control, automation, robotics and vision, 1, 289-293.
- Wan, S. F., Zhang, Q. Q. & Fang, M. F. (2008). SMS Receiving and Dispatching System Based on Embeded Linux. 2008 ISECS International Colloquium on Computing, Communication, Control, and Management, 2, 475-479.
- Zhao, Y. & Ye Z. (2008). A low cost GSM/GPRS based wireless home security system. IEEE Transactions on consumer electronics, 54 (2), 567-572.

APPENDICES

APPENDIX A: Main Program Code

```
#include <stdlib.h>
#include "GenericTypeDefs.h"
#include "usb_config.h"
#include "USB/usb.h"
#include "USB/usb_host_generic.h"
#include "DEE Emulation 16-bit.h"

// Configuration Bits
// *****
_CONFIG1(WDTPS_PS1 & FWPSA_PR32 & WINDIS_OFF & FWDTEN_OFF & ICS_PGx1 &
GWRP_OFF & GCP_OFF & JTAGEN_OFF)
_CONFIG2(POSCMOD_HS & I2C1SEL_PRI & IOL1WAY_OFF & OSCIOFNC_ON &
FCKSM_CSDCMD & FNOSC_PRIPLL & PLL96MHZ_ON & PLLDIV_DIV2 & IESO_ON)
_CONFIG3(WPFP_WPFP0 & SOSCSEL_IO & WUTSEL_LEG & WPDIS_WPDIS &
WPCFG_WPCFGDIS & WPEND_WPENDMEM)
_CONFIG4(DSWDTPS_DSWDTPS3 & DSWDTOSC_LPRC & RTCOSC_LPRC &
DSBOREN_OFF & DSWDTEN_OFF)

// Global Variables
// *****
#define SLEN 10
BYTE deviceAddress; // Address of the device on the USB
unsigned int USBOn = 0, currstate = 0, retry = 0, alarm = 0, offalarm = 0, halt = 0, wrong = 0,
halttime = 0, cardkey = 0;
char Home[150];
char buff[4] = {'A','T',13,'\0'};
char buff0[12] = "AT+CMGS=52";
char buff1[123] =
"07910621000010F511000B910621520291F10000AA2BD3F2B82E4FD3F3A020BB2CA7835AA0
AA3B5CA7A3DFF2B4BE4C0691DF6F39E80D2FBBD3EE3308";
char buff2[12] = "AT+CMGS=64";
char buff3[147] =
"07910621000010F511000B910621520291F10000AA39D3F2B82E4FD3F3A020BB2CA7835AA0
ABFCED3E83A0E1F9FCFE9693414537BD2C2F9341CDB7BC0CA2A2C36E500D444DB7CB73";

// Function Prototype
void delay(unsigned int msec);
void initt2(unsigned int time);
void initt1(void);
void initU1(void);
void writeU1(char c);
void writesU1(char *str);
```

```

char readU1(void);
char *readsU1(char *str, unsigned int slen);
void accesslcd(void);
void writelcd8b(unsigned int fnc, char text);
void writelcd4b(unsigned int fnc, char text);
void writeslcd (unsigned int fnc, char *string);
void accesskey(void);
char readkey(void);
void initlcd(void);
void DataInit(void);
BOOL InitializeSystem (void);
void readcard(void);
BOOL CheckForNewAttach ( void );
BOOL USB_ApplicationEventHandler ( BYTE address, USB_EVENT event, void *data, DWORD
size );
void SendAlert(void);
void SendAlert1(void);
void RWFlash(unsigned int task);
void GetPass(unsigned int Add);
void GetNum(unsigned int Add);
void GetTime(void);
void setup(void);
void grandaccess (void);
void blockaccess (void);
unsigned int verifykey(unsigned int Add);
unsigned int disarm(unsigned int Add);

```

// Interrupt Routines

```

void __attribute__((interrupt, no_auto_psv)) _U1RXInterrupt(void);
void __attribute__((interrupt, no_auto_psv)) _CNInterrupt(void);
void __attribute__((interrupt, no_auto_psv)) _T3Interrupt(void);
void __attribute__((interrupt, no_auto_psv)) _T1Interrupt(void);

```

```

void _ISR _U1RXInterrupt(void)
{
    IEC0bits.U1RXIE = 0;    // Disable U1RX interrupt
    readcard();
    IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
    IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
}

```

```

void _ISR _CNInterrupt(void)
{
    if (PORTAbits.RA4 == 0 && offalarm == 0){
        TRISBbits.TRISB5 = 0;
        PORTBbits.RB5 = 1;
        alarm = 1;
        SendAlert();
    }

    if (PORTAbits.RA4 == 1 && offalarm == 1){
        offalarm = 0;
    }
    while (alarm){
        if (disarm(200)){
            if (alarm == 1 && PORTAbits.RA4 == 1){
                offalarm = 0;
            }
        }
    }
}

```

```

        else{
            offalarm = 1;
        }
        alarm = 0;
    }
}
IFS1bits.CNIF = 0;
}

void _ISR_T3Interrupt(void)
{
    halt = 0;
    IFS0bits.T3IF = 0;
}

void _ISR_T1Interrupt(void)
{
    USBHostTasks();
    IFS0bits.T1IF = 0;
}
// Local Routines
//*****

/*      Function:      delay
      Overview:      This routine take time in millisecond and cause system to do nothing. */
void delay(unsigned int msec)
{
    unsigned int fac;
    for (;msec > 0;msec--)
    {
        for (fac = 16000; fac > 0; fac --);
    }
}

/*      Function:      initt2
      Overview:      This routine initializes timer 2 in 32bit operation for blockaccess function.
*/
void initt2(unsigned int time)
{
    T2CONbits.T32 = 1;           //32bit operation
    T2CONbits.TCKPS = 3; // 1:256 Prescale
    T2CONbits.TCS = 0;           // Intrnal fosc/2
    T2CONbits.TGATE = 0; // Disable gated timer operation
    PR3 = time;                  //
    PR2 = 0;                     //
    IFS0bits.T3IF = 0;           // Clear interrupt
    IEC0bits.T3IE = 1;           // Enable interrupt
    T2CONbits.TON = 1;
}

/*      Function:      initt1
      Overview:      This routine initializes timer 1 in 16bit operation for usbhosttask. */
void initt1(void)
{
    T1CONbits.TCKPS = 3; // 1:256 Prescale
    T1CONbits.TCS = 0;           // Intrnal fosc/2
    T1CONbits.TGATE = 0; // Disable gated timer operation
    PR1 = 3906;                  //

```

```

        IFS0bits.T1IF = 0;        // Clear interrupt
        IEC0bits.T1IE = 1;        // Enable interrupt
        T1CONbits.TON = 1;
    }

    /*      Function:      initU1
       Overview:      This routine initializes the Tx/Rx pin, setting the baud rate and enabling
interrupt on Rx pin. */
void initU1(void)
{
    U1BRG = 103;
    U1MODE = 0x8000;
    U1STA = 0x8400;
    IFS0bits.U1RXIF = 0;
    RPINR18bits.U1RXR = 5; // Assign U1RX To Pin RR5/RA0
    RPOR3bits.RP6R = 3;    // Assign U1TX To Pin RP6/RA1
    IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
    IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
    IPC2bits.U1RXIP = 7;
}

/*      Function:      writeU1
       Overview:      This routine takes a character as input and writes to Tx port. */
void writeU1(char c)
{
    while (U1STAbits.UTXBF);
    U1TXREG = c;
}

/*      Function:      writesU1
       Overview:      This routine takes a string from input and writes to Tx port. */
void writesU1(char *str)
{
    while (*str)
        writeU1(*str++);
}

/*      Function:      readU1
       Overview:      This routine reads a single character from Rx port and return the character.
*/
char readU1(void)
{
    while (!U1STAbits.URXDA);
    return U1RXREG;
}

/*      Function:      readsU1
       Overview:      This routine reads a string from Rx port and return the string. */
char *readsU1(char *str, unsigned int slen)
{
    char waste;
    char *ptr = str;
    waste = readU1(); // discard 1st unused character
    do{
        *str = readU1();
        str++;
    }

```

```

        slen--;
    }while (slen > 0);
    waste = readU1();           // discard last unused character
    *str = '\0';               // add null terminal to string
    return ptr;
}

/*      Function:      accesslcd
    Overview:      This routine enable write to lcd and disable input from keypad */
void accesslcd(void)
{
    TRISBbits.TRISB0 = 0;
    TRISBbits.TRISB1 = 0;
    TRISBbits.TRISB2 = 0;
    TRISBbits.TRISB3 = 0;
    TRISBbits.TRISB13 = 0; //Output RB13
    TRISBbits.TRISB14 = 0; //Output RB14
    PORTB = 0x0000;
}

/*      Function:      writelcd8b
    Overview:      This routine writes a byte of data to lcd. Data is send to lcd in 4bit mode */
void writelcd8b(unsigned int fnc, char text)
{
    accesslcd();               // Switch to LCD
    delay(1);
    char temp = text;         // Save data inside temp
    temp = temp & 0xF0;       // Take upper 4 bit
    temp = temp >> 4;         // Right shift to output data
    text = text & 0x0F;       // Take lower 4 bit
    PORTB = temp;             // Output upper 4 bit first
    if(fnc)                   // Check for intruccion or data
        PORTBbits.RB13 = 1;
    else
        PORTBbits.RB13 = 0;
    PORTBbits.RB14 = 1;       // Enable
    delay(1);
    PORTBbits.RB14 = 0;       // Disable
    PORTB = text;
    if(fnc)
        PORTBbits.RB13 = 1;
    else
        PORTBbits.RB13 = 0;
    PORTBbits.RB14 = 1;
    delay(1);
    PORTBbits.RB14 = 0;
}

/*      Function:      writelcd4b
    Overview:      This routine writes a 4bit of data to lcd. */
void writelcd4b(unsigned int fnc, char text)
{
    accesslcd();
    delay(1);
    PORTB = text;
    if(fnc)
        PORTBbits.RB13 = 1;

```

```

        else
        PORTBbits.RB13 = 0;
        PORTBbits.RB14 = 1;
        delay(10);
        PORTBbits.RB14 = 0;
        delay(1);
    }

    /*      Function:      writeslcd
    Overview:      This routine writes string to lcd in 4bit mode. */
    void writeslcd (unsigned int fnc, char *string)
    {
        while(*string)
        {
            writelcd8b(fnc,*string);
            string++;
        }
    }

    /*      Function:      accesskey
    Overview:      This routine enable input from keypad and disable write to lcd */
    void accesskey(void)
    {
        TRISBbits.TRISB0 = 0;
        TRISBbits.TRISB1 = 0;
        TRISBbits.TRISB2 = 0;
        TRISBbits.TRISB3 = 0;
        TRISBbits.TRISB4 = 1;
        TRISBbits.TRISB7 = 1;
        TRISBbits.TRISB8 = 1;
        TRISBbits.TRISB9 = 1;
        TRISBbits.TRISB13 = 1;
        TRISBbits.TRISB14 = 1;

        if (alarm)
            PORTB = 0x002F;
        else
            PORTB = 0x000F;
    }

    /*      Function:      readkey
    Overview:      This routine scan matrix keypad for key pressed and return the key pressed
    */
    char readkey(void)
    {
        accesskey();
        delay(1);
        unsigned int col = 0;
        char key = '\0';
        if(PORTBbits.RB4||PORTBbits.RB7||PORTBbits.RB8||PORTBbits.RB9)
        {
            if(PORTBbits.RB4){
                col = 1;
            }
            else if (PORTBbits.RB7){
                col = 2;
            }
            else if (PORTBbits.RB8){

```

```

        col = 3;
    }
    else if (PORTBbits.RB9){
        col = 4;
    }

    if (alarm)
        PORTB = 0x0020;
    else
        PORTB = 0x0000;
    delay(1);

    switch (col)
    {
    case 1 :
    if (alarm)
        PORTB = 0x0021;
    else
        PORTB = 0x0001;
    delay(1);
    if(PORTBbits.RB4)
    {
    key = '1';
    break;
    }

    if (alarm)
        PORTB = 0x0022;
    else
        PORTB = 0x0002;
    delay(1);
    if(PORTBbits.RB4)
    {
    key = '4';
    break;
    }

    if (alarm)
        PORTB = 0x0024;
    else
        PORTB = 0x0004;
    delay(1);
    if(PORTBbits.RB4)
    {
    key = '7';
    break;
    }

    if (alarm)
        PORTB = 0x0028;
    else
        PORTB = 0x0008;
    delay(1);
    if(PORTBbits.RB4)
    {
    key = '*';
    break;
    }

    case 2 :

```

```

    if (alarm)
        PORTB = 0x0021;
    else
        PORTB = 0x0001;
    delay(1);
    if(PORTBbits.RB7)
    {
        key = '2';
        break;
    }

    if (alarm)
        PORTB = 0x0022;
    else
        PORTB = 0x0002;
    delay(1);
    if(PORTBbits.RB7)
    {
        key = '5';
        break;
    }

    if (alarm)
        PORTB = 0x0024;
    else
        PORTB = 0x0004;
    delay(1);
    if(PORTBbits.RB7)
    {
        key = '8';
        break;
    }

    if (alarm)
        PORTB = 0x0028;
    else
        PORTB = 0x0008;
    delay(1);
    if(PORTBbits.RB7)
    {
        key = '0';
        break;
    }

    case 3 :
    if (alarm)
        PORTB = 0x0021;
    else
        PORTB = 0x0001;
    delay(1);
    if(PORTBbits.RB8)
    {
        key = '3';
        break;
    }

    if (alarm)
        PORTB = 0x0022;
    else
        PORTB = 0x0002;

```

```

delay(1);
if(PORTBbits.RB8)
{
key = '6';
break;
}

if (alarm)
    PORTB = 0x0024;
else
    PORTB = 0x0004;
delay(1);
if(PORTBbits.RB8)
{
key = '9';
break;
}

if (alarm)
    PORTB = 0x0028;
else
    PORTB = 0x0008;
delay(1);
if(PORTBbits.RB8)
{
key = '#';
break;
}

case 4 :
if (alarm)
    PORTB = 0x0021;
else
    PORTB = 0x0001;
delay(1);
if(PORTBbits.RB9)
{
key = 'A';
break;
}

if (alarm)
    PORTB = 0x0022;
else
    PORTB = 0x0002;
delay(1);
if(PORTBbits.RB9)
{
key = 'B';
break;
}

if (alarm)
    PORTB = 0x0024;
else
    PORTB = 0x0004;
delay(1);
if(PORTBbits.RB9)
{
key = 'C';

```



```

for( num = 0; num < 10; num++)
{
    for ( i = 1; i < 11 ; i++ )
    {
        str1[i-1]=DataEERead(num*10+i);
    }
    str1[10] = '\0';
    if(!strcmp(str, str1))
        flag = 1;
}

if (flag){
    if (cardkey == 1){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"USER PASSWORD");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        if (verifykey(200)){
            TRISBbits.TRISB15 = 0; //Output RB15
            PORTBbits.RB15 = 1;
            delay(3000);
        }
        else{
            TRISBbits.TRISB15 = 0; //Output RB15
            PORTBbits.RB15 = 0;
        }
    }
    else{
        writelcd8b(0,0x01);
        writeslcd(1,"ACCESS GRANTED");
        TRISBbits.TRISB15 = 0; //Output RB15
        PORTBbits.RB15 = 1;
        delay(3000);
    }
}
else{
    writelcd8b(0,0x01);
    writeslcd(1,"ACCESS DENIED");
    TRISBbits.TRISB15 = 0; //Output RB15
    PORTBbits.RB15 = 0;
}
}

```

```

/*      Function:      CheckForNewAttach
      Overview:      This routine checks to see if a new device has been attached. If it has, it
records the address. */
BOOL CheckForNewAttach ( void )
{
    if (deviceAddress == 0)
    {
        GENERIC_DEVICE_ID DevID;

#ifdef USB_GENERIC_SUPPORT_SERIAL_NUMBERS
        DevID.serialNumberLength = 0;
        DevID.serialNumber = NULL;
#endif
    }
}

```

```

    if (USBHostGenericGetDeviceAddress(&DevID))
    {
        deviceAddress = DevID.deviceAddress;
        #ifdef DEBUG_MODE
        #endif
        return TRUE;
    }
    return FALSE;
}

// USB Support Functions
//*****
/* Function:    USB_ApplicationEventHandler
   Overview:    This routine is called by the Host layer or client driver to notify the
                  application of events that occur. If the event is recognized, it is
                  handled and the routine returns TRUE. Otherwise, it is ignored
(or
                  just "sniffed" and the routine returns FALSE. */
BOOL USB_ApplicationEventHandler ( BYTE address, USB_EVENT event, void *data, DWORD
size )
{
    #ifdef USB_GENERIC_SUPPORT_SERIAL_NUMBERS
    BYTE i;
    #endif

    // Handle specific events.
    switch (event)
    {
        case EVENT_GENERIC_ATTACH:
            writesU1( "Generic demo device attached\r\n" );
            if (size == sizeof(GENERIC_DEVICE_ID))
            {
                deviceAddress = ((GENERIC_DEVICE_ID *)data)->deviceAddress;
                USBOn = 1;

                writelcd8b(0,0x01);
                writelcd8b(0,0x80);
                writeslcd(1,"DEVICE OK");
                writelcd8b(0,0xC0);
                writeslcd(1,"ADDRESS = ");
                writelcd8b(1,deviceAddress+0x30);

                #ifdef USB_GENERIC_SUPPORT_SERIAL_NUMBERS
                for (i=1; i<(((GENERIC_DEVICE_ID *)data)->serialNumberLength; i++)
                {
                    writesU1( ((GENERIC_DEVICE_ID *)data)->serialNumber[i] );
                }
                #endif
                writesU1( "\r\n" );
                return TRUE;
            }
            break;

        case EVENT_GENERIC_DETACH:
            deviceAddress = 0;
            USBOn = 0;

            writelcd8b(0,0x01);
            writeslcd(1,"DEVIDE DETACHED");
            delay(300);
    }
}

```

```

        return TRUE;

case EVENT_GENERIC_TX_DONE:
case EVENT_GENERIC_RX_DONE:
    return TRUE;

case EVENT_VBUS_REQUEST_POWER:
    return TRUE;

case EVENT_VBUS_RELEASE_POWER:
    return TRUE;

case EVENT_HUB_ATTACH:
    writelcd8b(0,0x01);
    writeslcd(1,"HUB DETECTED");

    return TRUE;
    break;

case EVENT_UNSUPPORTED_DEVICE:
    writelcd8b(0,0x01);
    writeslcd(1,"DEVICE ERROR");

    return TRUE;
    break;

case EVENT_CANNOT_ENUMERATE:
    writelcd8b(0,0x01);
    writeslcd(1,"ENUMERATE FAIL");

    return TRUE;
    break;

case EVENT_CLIENT_INIT_ERROR:
    writelcd8b(0,0x01);
    writeslcd(1,"CLIENT ERROR");

    return TRUE;
    break;

case EVENT_OUT_OF_MEMORY:
    writelcd8b(0,0x01);
    writeslcd(1,"OUT OF MEMORY");

    return TRUE;
    break;

case EVENT_UNSPECIFIED_ERROR:
    writelcd8b(0,0x01);
    writeslcd(1,"UNKNOW ERROR");

    return TRUE;
    break;

case EVENT_SUSPEND:
case EVENT_DETACH:
case EVENT_RESUME:
case EVENT_BUS_ERROR:
    return TRUE;
    break;

default:
    break;
}
return FALSE;
}

```

```

/* Function:      SendAlert
   Overview:      This routine send sms to user when door is forced to opened */
void SendAlert(void)
{
    BYTE RetVal;
    unsigned int EP = 0;
    USBHostTasks();
    if (USBOn == 1)
    {
        while (USBHostGenericTxIsBusy(deviceAddress));
        RetVal = USBHostGenericEP2Write(deviceAddress, buff, 3);
        if (RetVal == USB_SUCCESS){
            EP = 1;
        }
        else{
            EP = 0;
        }

        if (currstate == 0){
            while(USBHostGenericTxIsBusy(deviceAddress));
            if(EP == 1){
                RetVal=USBHostGenericEP2Write(deviceAddress, buff0, 11);
            }
            else{
                RetVal=USBHostGenericWrite(deviceAddress, buff0, 11);
            }
            currstate = 1;
            delay(100);
        }
        if (currstate == 1)
        {
            while(USBHostGenericRxIsBusy(deviceAddress));
            if (EP == 1){
                RetVal=USBHostGenericEP2Read(deviceAddress,Home,
sizeof(Home));
            }
            else{
                RetVal=USBHostGenericRead(deviceAddress,Home,
sizeof(Home));
            }
            currstate = 2;
        }

        if (currstate == 2){
            while(USBHostGenericTxIsBusy(deviceAddress));
            if(EP == 1){
                RetVal=USBHostGenericEP2Write(deviceAddress, buff1, 122);
            }
            else{
                RetVal=USBHostGenericWrite(deviceAddress, buff1, 122);
            }
            currstate = 3;
            delay(1000);
        }

        if (currstate == 3)
        {
            while(USBHostGenericRxIsBusy(deviceAddress));
            if (EP == 1){

```

```

RetVal=USBHostGenericEP2Read(deviceAddress,Home,
sizeof(Home));
    }
    else{
RetVal=USBHostGenericRead(deviceAddress,Home,
sizeof(Home));
    }
    currstate = 0;
}
}

/* Function:    SendAlert1
   Overview:    This routine send sms to user when password entered was wrong 5 times */
void SendAlert1(void)
{
    BYTE RetVal;
    unsigned int EP = 0;
    USBHostTasks();
    if (USBOn == 1)
    {
        while (USBHostGenericTxIsBusy(deviceAddress));
        RetVal = USBHostGenericEP2Write(deviceAddress, buff, 3);
        if (RetVal == USB_SUCCESS){
            EP = 1;
        }
        else{
            EP = 0;
        }

        if (currstate == 0){
            while(USBHostGenericTxIsBusy(deviceAddress));
            if(EP == 1){
                RetVal=USBHostGenericEP2Write(deviceAddress, buff2, 11);
            }
            else{
                RetVal=USBHostGenericWrite(deviceAddress, buff2, 11);
            }
            currstate = 1;
            delay(100);
        }
        if (currstate == 1)
        {
            while(USBHostGenericRxIsBusy(deviceAddress));
            if (EP == 1){
                RetVal=USBHostGenericEP2Read(deviceAddress,Home,
sizeof(Home));
            }
            else{
                RetVal=USBHostGenericRead(deviceAddress,Home,
sizeof(Home));
            }
            currstate = 2;
        }

        if (currstate == 2){
            while(USBHostGenericTxIsBusy(deviceAddress));
            if(EP == 1){
                RetVal=USBHostGenericEP2Write(deviceAddress, buff3, 146);
            }
            else{

```

```

        RetVal=USBHostGenericWrite(deviceAddress, buff3, 146);
    }
    currstate = 3;
    delay(2000);
}

if (currstate == 3)
{
    while(USBHostGenericRxIsBusy(deviceAddress));
    if (EP == 1){
        RetVal=USBHostGenericEP2Read(deviceAddress,Home,
sizeof(Home));
    }
    else{
        RetVal=USBHostGenericRead(deviceAddress,Home,
sizeof(Home));
    }
    currstate = 0;
}
}
}
}

```

/* Function: verifykey

Overview: This routine add/remove card unique ID from database */

```

void RWFlash(unsigned int task){
    unsigned int i = 1;
    unsigned int num;
    char str[11] = "*****";
    IEC0bits.U1RXIE = 0;    // Disable U1RX interrupt
    do{
        num = readkey();
        USBHostTasks();
    }
    while(num == '\0' || num == 'B');
    writeU1(num);
    writelcd8b(1,num);
    if (task == 1){
        writelcd8b(0,0x01);
        writeslcd(1,"PLEASE SCAN CARD");
        readsU1(str,SLEN);
    }
    DataEEInit();
    dataEEFlags.val = 0;
    num -= 0x30;
    while(i < 11 && USBOn == 1){
        DataEEWrite(str[i-1],num*10+i);
        i++;
    }
    writelcd8b(0,0x01);
    writeslcd(1,"DONE");
    IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
    IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
}

```

/* Function: GetPass

Overview: This routine get password from user */

```

void GetPass(unsigned int Add){
    unsigned int i = 0;
    char Pass[7];
    char Temp;

```

```

IEC0bits.U1RXIE = 0;    // Disable U1RX interrupt
while( i < 6 ){
    Temp = readkey();
    if(Temp != '\0' && Temp != 'A' && Temp != 'B' && Temp != 'D' ){
        Pass[i] = Temp;
        if (Pass[i]!='C'){
            writeU1(Pass[i]);
            writelcd8b(1,Pass[i]);
            i++;
        }
        else{
            i = 0;
            writelcd8b(0,0x01);
            writelcd8b(0,0x80);
            writeslcd(1,"ENTER PASSWORD");
            writelcd8b(0,0xC0);
            writelcd8b(1,'>');
        }
    }
}
i = 0;
DataEEInit();
dataEEFlags.val = 0;
while(i < 6){
    DataEEWrite(Pass[i],Add+i);
    i++;
}
writelcd8b(0,0x01);
writeslcd(1,"DONE");
DataEEWrite('Y',254);
IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
}

```

```

/* Function:      GetNum
Overview:        This routine get target phone number & SMSC number from user */
void GetNum(unsigned int Add){
    unsigned int i = 0;
    char Num[13];
    char Temp;
    Num[11] = 'F';
    Num[12] = '\0';
    char str1[12];

    IEC0bits.U1RXIE = 0;    // Disable U1RX interrupt
    while( i < 11 ){
        Temp = readkey();
        if(Temp != '\0' && Temp != 'A' && Temp != 'B' && Temp != 'D' ){
            Num[i] = Temp;
            if (Num[i]!='C'){
                writeU1(Num[i]);
                writelcd8b(1,Num[i]);
                i++;
            }
            else{
                i = 0;
                writelcd8b(0,0x01);
                if (Add == 220){

```

```

        writelcd8b(0,0x80);
        writeslcd(1,"ENTER SMSC NUM.");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
    }
    else{
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER PHONE NUM.");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
    }
}

}

for(i = 0; i < 12; i+=2){
    Temp = Num[i];
    Num[i] = Num[i+1];
    Num[i+1] = Temp;
}
i = 0;
DataEEInit();
dataEEFlags.val = 0;
while(i < 13){
    DataEEWrite(Num[i],Add+i);
    i++;
}
writelcd8b(0,0x01);
writeslcd(1,"DONE");
i = 0;
while(i < 12){
    str1[i]=DataEERead(Add+i);
    buff1[24+i] = DataEERead(240+i);
    buff1[4+i] = DataEERead(220+i);
    i++;
}
writesU1(str1);
IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
}

```

```

/* Function:    GetTime
  Overview:    This routine get blocked time from user */
void GetTime(void){
    unsigned int i = 0, t = 0;
    char time[3];
    char Temp;

    IEC0bits.U1RXIE = 0;    // Disable U1RX interrupt
    while( i < 2 ){
        Temp = readkey();
        if(Temp != '\0' && Temp != 'A' && Temp != 'B' && Temp != 'D'){
            time[i] = Temp;
            if (time[i]!='C'){
                writeU1(time[i]);
                writelcd8b(1,time[i]);
                i++;
            }
            else{

```

```

        i = 0;
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"MINUTES TO BLOCK");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
    }
}

t = ((time[0]-0x30) + ((time[1]-0x30)*10))*60;
DataEEInit();
dataEEFlags.val = 0;
DataEEWrite(t,234);
writelcd8b(0,0x01);
writeslcd(1,"DONE");
IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
}

/* Function:      GetType
   Overview:      This routine get access type from user */
void GetType(void){
    unsigned int i = 0;
    char Temp;

    IEC0bits.U1RXIE = 0;    // Disable U1RX interrupt
    while( i < 1 ){
        Temp = readkey();
        if(Temp == '1' || Temp == '0'){
            if (Temp == '1'){
                cardkey = 1;
            }
            else{
                cardkey = 0;
            }
            i++;
        }
    }
    DataEEInit();
    dataEEFlags.val = 0;
    DataEEWrite(cardkey,235);
    writelcd8b(0,0x01);
    writeslcd(1,"DONE");
    IFS0bits.U1RXIF = 0;    // Reset U1RX interrupt register
    IEC0bits.U1RXIE = 1;    // Enable U1RX interrupt
}

/* Function:      setup
   Overview:      This routine configure user password, admin password, target phone number ,
                  add/remove card, setting bloacked time, setting access type and
   user SMSC number*/
void setup(void){

    unsigned int i = 1;
    char keypad;
    writelcd8b(0,0x01);
    writelcd8b(0,0x80);
    writeslcd(1,"SETUP MENU:");
    writelcd8b(0,0xC0);

```

```

writelcd8b(1,'>');

while (i){
    keypad = readkey();

    if (keypad == '1'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER CARD ID");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        RWFlash(1);
        i = 0;
    }
    else if (keypad == '2'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER CARD ID");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        RWFlash(2);
        i = 0;
    }
    else if (keypad == '3'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER PASSWORD");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        GetPass(200);
        i = 0;
    }
    else if (keypad == '4'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER PASSWORD");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        GetPass(210);
        i = 0;
    }
    else if (keypad == '5'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER PHONE NUM.");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        GetNum(240);
        i = 0;
    }
    else if (keypad == '6'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"ENTER SMSC NUM.");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        GetNum(220);
        i = 0;
    }
    else if (keypad == '7'){

```

```

        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"MINUTES TO BLOCK");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        GetTime();
        i = 0;
    }
    else if (keypad == '8'){
        writelcd8b(0,0x01);
        writelcd8b(0,0x80);
        writeslcd(1,"TAG & PASSWORD?");
        writelcd8b(0,0xC0);
        writelcd8b(1,'>');
        GetType();
        i = 0;
    }
    else if (keypad == '*'){
        i = 0;
        writelcd8b(0,0x01);
        writeslcd(1,"CANCELLED");
    }
}

/* Function:    grandaccess
   Overview:    This routine release magnetic door */
void grandaccess (void){
    TRISBbits.TRISB15 = 0; //Output RB15
    PORTBbits.RB15 = 1;
    IEC1bits.CNIE = 0;
    offalarm = 1;
    delay(3000);
    IEC1bits.CNIE = 1;
    if (PORTAbits.RA4 == 1 && offalarm == 1){
        offalarm = 0;
    }
}

/* Function:    blockaccess
   Overview:    This routine block access after 5 times trying */
void blockaccess (void){
    initt2(halttime);
    halt = 1;
    writelcd8b(0,0x01);
    writeslcd(1,"ACCESS BLOCKED");
    SendAlert1();
    while (halt);
    writelcd8b(0,0x01);
    writeslcd(1,"ACCESS UNBLOCKED");
    T2CONbits.TON = 0;
    wrong = 0;
}

/* Function:    verifykey
   Overview:    This routine verify the password entered by user */
unsigned int verifykey(unsigned int Add){
    unsigned int i = 0;
    char temp;

```

```

char pass[7];
char input[7];

while(i < 6){
    temp = readkey();
    if(temp != '\0' && temp != 'A' && temp != 'B' && temp != 'D'){
        input[i] = temp;
        if (input[i] != 'C'){
            writeU1(input[i]);
            writelcd8b(1,input[i]);
            i++;
        }
        else{
            i = 0;
            writelcd8b(0,0x01);
            if (Add == 200){
                writelcd8b(0,0x80);
                writeslcd(1,"USER PASSWORD");
                writelcd8b(0,0xC0);
                writelcd8b(1,'>');
            }
            else{
                writelcd8b(0,0x80);
                writeslcd(1,"ADMIN PASSWORD");
                writelcd8b(0,0xC0);
                writelcd8b(1,'>');
            }
        }
    }
}
input[6] = '\0';
DataEEInit();
dataEEFlags.val = 0;
for( i = 0; i < 6; i++){
    pass[i] = DataEERead(Add+i);
    pass[6] = '\0';
}
if(!strcmp(input, pass)){
    writelcd8b(0,0x01);
    writeslcd(1,"ACCESS GRANTED");
    i = 1;
    wrong = 0;
}
else{
    writelcd8b(0,0x01);
    writeslcd(1,"ACCESS DENIED");
    i = 0;
    wrong += 1;
}
return i;
}

```

/* Function: disarm

Overview: This routine is use to turn off the alarm */

```

unsigned int disarm(unsigned int Add){
    unsigned int i = 0;
    char temp;
    char pass[7];
    char input[7];

```

```

while(i < 6){
    USBHostTasks();
    temp = readkey();
    if(temp != '\0' && temp != 'A' && temp != 'B' && temp != 'D'){
        input[i] = temp;
        if (input[i] != 'C'){
            i++;
        }
        else{
            i = 0;
        }
    }
    input[6] = '\0';
    DataEEInit();
    dataEEFlags.val = 0;
    for( i = 0; i < 6; i++){
        pass[i] = DataEERead(Add+i);
        pass[6] = '\0';
    }

    if(!strcmp(input, pass)){
        i = 1;
        writelcd8b(0,0x01);
        writeslcd(1,"ALARM DISARM");
    }
    else{
        i = 0;
    }
    return i;
}

```

```

/* Function:    main
   Overview:    This is the Application's main entry point */
int main ( void )
{
    char keypad;

    if ( InitializeSystem() != TRUE )
    {
        writelcd8b(0,0x01);
        writeslcd(1, "SYSTEM FAILURE");
        while (1);
    }
    if ( USBHostInit(0) == TRUE )
    {
        writeslcd(1, "BOOTING UP.....");
    }
    else
    {
        writelcd8b(0,0x01);
        writeslcd(1, "USB FAILURE");
        while (1);
    }
    delay(500);
    initt1();
    while (1)

```

```

{
    while (USBOn == 1 || USBOn == 2){
        if (wrong > 4){
            blockaccess();
        }
        keypad = readkey();

        if (keypad == 'B')
        {
            writelcd8b(0,0x01);
            writelcd8b(0,0x80);
            writeslcd(1,"USER PASSWORD");
            writelcd8b(0,0xC0);
            writelcd8b(1,'>');
            if (verifykey(200)){
                writelcd8b(0,0x01);
                writelcd8b(0,0x80);
                writeslcd(1,"ADMIN PASSWORD");
                writelcd8b(0,0xC0);
                writelcd8b(1,'>');
                if (verifykey(210)){
                    setup();
                }
            }
        }
        else if (keypad == 'D'){
            writelcd8b(0,0x01);
            writelcd8b(0,0x80);
            writeslcd(1,"USER PASSWORD");
            writelcd8b(0,0xC0);
            writelcd8b(1,'>');
            if (verifykey(200)){
                grandaccess();
            }
        }
    }
    writelcd8b(0,0x01);
    writelcd8b(0,0x80);
    writeslcd(1,"PLEASE PLUG IN");
    writelcd8b(0,0xC0);
    writeslcd(1,"GSM TERMINAL");
    while(USBOn == 0){
        if (wrong > 4){
            blockaccess();
        }
        keypad = readkey();

        if (keypad == 'A'){
            writelcd8b(0,0x01);
            writelcd8b(0,0x80);
            writeslcd(1,"USER PASSWORD");
            writelcd8b(0,0xC0);
            writelcd8b(1,'>');
            if (verifykey(200)){
                writelcd8b(0,0x01);
                writelcd8b(0,0x80);
                writeslcd(1,"ADMIN PASSWORD");
                writelcd8b(0,0xC0);
                writelcd8b(1,'>');
                if (verifykey(210)){

```

```
        USBOn = 2;
    }
}
}
}
    return 0;
}
```

Mobile Phone Controlling Home Appliances

Ridza Azri Ramlee, Daphne Tang Hui Zyen, Mohd Azlishah Othman, Mohd Muzafar Ismail, Hamzah Asyrani Sulaiman, Mohamad Harris Misran, Maizatul Alice Meor,
Mohamad Zoinol Abidin Abd Aziz

*Faculty of Electronic and Computer Engineering,
Universiti Teknikal Malaysia Melaka (UTeM), Malaysia
ridza@utem.edu.my*

Abstract—Advancement in wireless technology nowadays has allowed the creation of various remote control systems, one of which is the famous Smart Home concept. This project focuses on the development of a smart home system that allows user control of electrical appliances using devices such as laptop or an Android phone via Bluetooth. The system adapts serial connection of the devices with a Bluetooth module and a PIC microcontroller attached on the main circuit board where the microcontroller will then control the home appliances via a relay circuit. The smart home system will be able to ease the effort of physically challenged individuals in controlling their home appliances such as lamps, fan, air-con and etc. Users can trigger the switches anywhere as long as the device is within the vicinity of the Bluetooth signal in the main panel. Moreover, users can also control the appliances via internet connection by sending an email to a specified address. However, this addition requires that the laptop be wirelessly connected to the main board at all times and have internet connection. Generally, the project software application together with its Graphical User Interface was developed using Microsoft® Visual Studio in VB language for the computer platform and Eclipse IDE for the Android platform. The microcontroller, PIC16F877A located on the main circuit board was programmed in C language using MPLAB IDE software and debugged together with the circuit design using Proteus 7 ISIS. The PCB was designed using Proteus 7 ARES. The final prototype of the system was built to demonstrate the proposed functionality of the system.

Index Terms— Android, Bluetooth, Home automation, PIC, Smart home

I. INTRODUCTION

Smart home system is a system that eases user's control over their home appliances and further improves their lifestyle. This project focuses on the adaptation of a low-cost smart home system for domestic purposes. With its target audience set, the project implementations utilize smart technologies such as PIC microcontroller as the main controller for the system. Its user interface is designed for the installation on Windows OS computer and also for the Android platform. Apart from integration of sensors into the system for monitoring purposes, the system integrates Internet technologies where the system can be controlled via messaging system using electronic mail. The system's feedback mechanism allows user to check or prompt for home conditions and further enhances the interactivity of the system. The system is targeted for application in conventional homes for people with disabilities or for the elderly. The nature of this system will provide these people with additive comfort in their own homes and further increase their productivity in their everyday lives.

The main objective of this project is to design a smart system that allows remote control characteristics in easing the controls of home appliances. This will require the integration of the latest wireless technologies such as Bluetooth and Wi-Fi components into the system's concept and application. Also, the other objective of this project is to allow the system to portray a two-way communication system between the house-hold appliances and user's controlling device. This allows the implementation of a feedback mechanism in the system.

Since the invention of mechanical switches, its function has always been simple and direct; which was to connect and disconnect power supply of an electric or electronic device. However, its usage requires that users make direct contact with the switches for it to work. It is deduced from that, direct contact switches will cause problems to arise, such as if the user accidentally touches the switch with wet hands, and if at crucial times, a physically challenged individual or a child could not reach a switch to prevent mishaps. These minor events may lead to accidents or people getting hurt.

Apart from that, conventional switches do not allow control over a distance. With this limitation, users will have less flexibility in controlling their home appliances when they are not at home and prevents remote monitoring of their home devices. In sense, if a house is not occupied, or does not seem occupied for a period of time, it is bound to attract unwanted visitors to our homes such as thieves. Prevention of these mishaps can be further improved by introducing remote controlling of devices.

This project focuses on developing a working model of a remote control smart home system which includes the GUI application for user interaction with a PIC microcontroller attached on a main circuit board. The main circuit board and the GUI will be designed to resemble a working miniature model or prototype of the product. The project will also include adaptation of appliance control over Internet services. However, the research to install this system into existing home circuitries will not be covered.

II. LITERATURE REVIEW

The review was done based on studying proceeding papers and journal articles which were obtained from UTeM's digital library subscription especially from IEEE Xplore website. Some were obtained online from random sources such as university websites.

These sources were found by searching the database using phrases such as 'mobile phone controlling home appliances', 'home automation using 'Bluetooth', 'home

automation for disabled people' and 'smart home for elderly and disabled'. These phrases were also used while looking up journals available through Google search engine. However, only some of the main related journals were cited in this report.

In this section will categorize past systems and literatures according to their cost and further discusses the varieties and concepts of their systems. Discussions will include the analysis of how all these systems are implemented.

A. Low Cost Systems

The latest related journal was produced by [1] which used the open source microcontroller, the Arduino Bluetooth (BT) platform comprising of ATmega 168 and Bluegiga 'WT II' Bluetooth module, paired with smartphone application built using Phytion running on Symbian OS. The journal highlighted its low cost installation but did not offer customization for the user in adding appliances and devices. Its limitation to operate on only Symbian phones has limited the operability of the system. However, [1] provided a useful concept in connecting the appliances to a control board controlled by the ATmega microcontroller that uses relay in controlling power supply to the appliances; the concept can be adapted into the current project.

Another study done by [2] achieved a very complete and moneywise concept by using GPRS as the medium to control and monitor home appliances. They summarized that at low polling rates at 30 seconds, monthly cost of using their system reaches only US\$ 5.10. This tactic however, requires the setup of 4 main systems including the central server, and 3 sub-systems at home, on the web, and on the mobile platform which is tedious. It utilizes Local Area Network (LAN) at home using RabbitCore modules and is capable of alerting, controlling and monitoring premises at the user's home. The special feature covered in this journal is its concept to map the appliances at home from the web which would provide a neat configuration of the household controllable appliances. [2]

B. High Cost Systems

The concept studied by referee [3] can be seen implemented by referee [4], in 1995 where they had conducted a long term research on disability categories and developed the AUTONOMY system applying Environmental Control Systems (ECS) and Alternative and Augmentative Communication (AAC). The unavailability of wireless technologies then, limited the systems interconnection to European Installation Bus (EIB) standards and home installation of this system was tedious and complex. However, the remote control context had been developed to control Infrared (IR) electronics such as television (TV) and CD players, development of a mobile platform to run the UI, video door control, telephone and internet connectivity, and finally the emulation of keyboard and mouse for ease of control for disabled people. The implemented system was highly customizable in sense that the UI could be easily configured by the care-taker of the disabled person to suit each disabled person's capability to control.

Utilizing smart home concept in helping the disabled and elderly people, [5] created a modular system which was easier to install in conventional houses. Their primary

concern was towards the convergence of disability due to accidents and aging effects. As these individuals were not subjected to limitations by birth, their condition required different outlines where the B-live system was made modular so that the system could be implemented to suit different cases. Every module would provide control of different categories of devices. The system designed was complex as its modules can be integrated with several local (SPI, I2C, etc.) and remote (CAN, Zigbee, etc.) communication technologies simultaneously.

C. Research and Analytic Systems

Referee [3] on the other hand, studied on the knowledge of smart home technologies for disabled people and the factors involved in providing better systems and services for the smart home concept of these cases. The research pointed out the increasing numbers of disabled people around the world based on statistics due to increasing population growth, and that improving the lifestyle for this category is crucial in sustaining their independence and comfort. Referee [6] also conducted research in relation to interest as described in [3] by applying Semantic Matching Framework (SMF) to provide personalization of smart home services based on the semantics matching of user's model (capabilities) and linking to environmental model (preference). SMF here is used to identify suitable configurations and relevant smart home adaptations to be implemented into different individual cases.

Several journals such as [7], [8] provided a different approach to home automation for the elderly and disabled. Instead, they developed systems to monitor movements and habits of these groups and through records, depict their habits and recording these information in their activity profiles. The concept applies large amounts of sensor built on to the Human Machine Interaction (HMI) and surrounding.

D. Analysis of Literature

In terms of concept, it was found that the original project proposal resembled the idea of referee [1] in the sense of the appliances connection to the main control board and also the use of relays to control two state switches. Difference can be found in the application's specification of this project to use PIC16F877A instead of the AVR Atmega 168 microcontroller used by [1]. Also, referee [1] only provided UI for Symbian OS phones where other OS users could not use the system. As proposed in this project, there will be two main UI, namely on Windows 7 OS and Android OS to offer a larger platform variant.

From referee [4], the idea to control IR electronics such as television and radio could be used. However, due to time limitations, the implementation of this idea could only be jotted down for future improvements. However, unlike journal [4], this project will only rely on simple design and will not include emulation of user input devices to suit each disabled person's disability. However, the customizability of the system could be studied. This project will also not be a research or analytic based system to monitor human behavior. It will only provide ease of access to control house appliances and also monitor certain areas of the house.

In terms of connection variant, this project proposed mixture of wired and wireless connection, where wired

connection will run from the home appliances to the main control board while wireless connection will only exist in between the main control board and the UI platform, which is the phone or PC connected via Bluetooth.

III. METHODOLOGY

This chapter explains the ideas and methods of this project. Details of how the project will be implemented will be explained under the project components section to describe how each component assists in the development of this project.

A. System Design

Apart from wireless connections, the smart home system will connect to the appliances grid via wired electrical connection. The illustration of the whole system will be provided in block diagram form including the system's functionality flow chart (Figure 1). Figure 2 shows the block diagram for main controller panel.

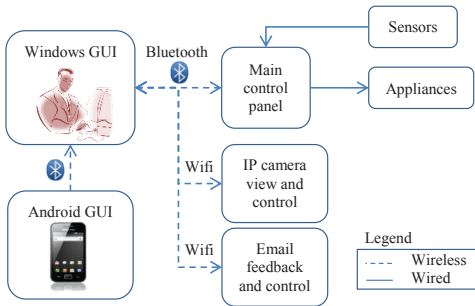


Figure 1: General Block Diagram of Smart Home System. Wireless connections are shown in dashed line while wired ones are shown in completed lines

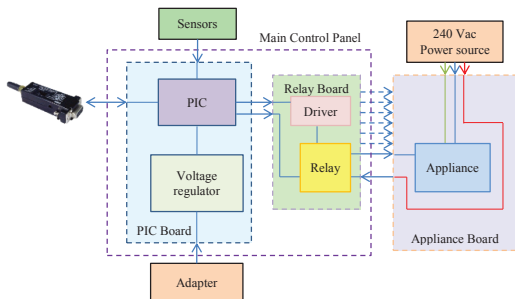


Figure 2: Block Diagram of Main Control Panel and Appliances. Elements in the blocks are important for the hardware design of the system

B. System Flow Chart

To describe the flow of the system's logic, a general flow chart will be included to illustrate the idea of the project (Figure 3). Two parts will be focused in this section which is the interaction of Windows GUI and Android GUI (Figure 4) with the main control panel. The Windows application will serve as a controller and also a server of the system. The flow chart will show how elements of the system interact with each other (Figure 5).

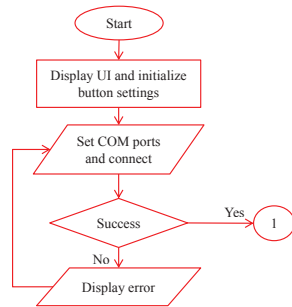


Figure 3: System Flow Chart for Server Application. Red box shows server actions

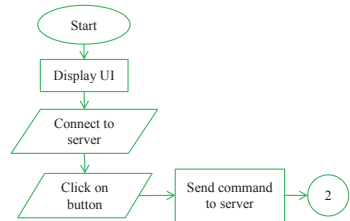


Figure 4: System Flow Chart for Android Application. Green box shows Android application actions

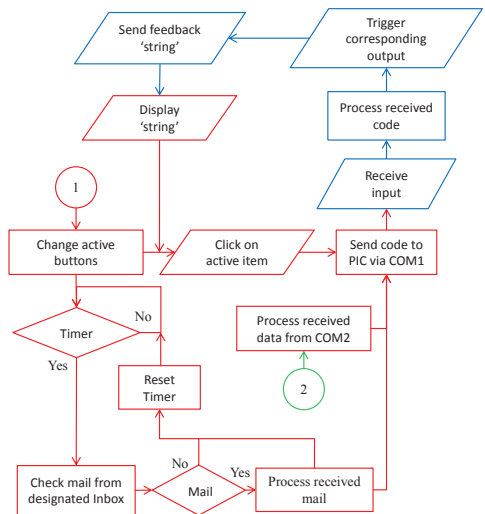


Figure 5: Server to Main Control Panel System Flow Chart. The figure shows the integration of all the devices

C. PIC Board

The PIC board will consist of microcontroller, PIC16F877A together with its basic functional circuit. The design of the board will incorporate the voltage regulator circuit with LM7805 expecting the input of 12Vdc and producing output of 5V to power the PIC, LEDs, and relays. To achieve sufficient amount of current to prevent malfunction of the circuit, a current rating of 2A is preferred. The board will be able to adapt to two input sources which is 12Vdc or 5Vdc. 5Vdc source will not be

feed through the voltage regulator but will directly supply for the components.

Another main requirement for the PIC board is the placing of the RS232 port together with its MAX232 circuit. This driver serves as the communication port between the Bluetooth module and PIC microcontroller. However, the PIC board (Figure 6) will mainly serve as a simple breakout board for interfacing of other components to the board such as the relay board and the appliance mapping circuit. The power supply is not included in the diagram because it only serves as a supply on the PIC board.

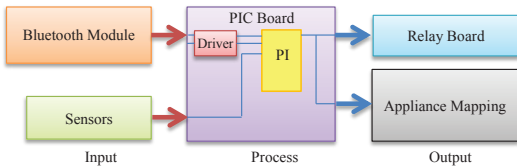


Figure 6: PIC Board Inputs and Outputs. Inputs are shown in red arrows and outputs in blue arrows

The relay board (Figure 7) will be designed based on its function to trigger relays based on the microcontroller pins output. Since that the relay driver, ULN2003, can only control seven relays per chip, the system prototype will be built to contain seven appliances or switches. This means that the relay board will contain a seven pin input, one power supply input and eight connections to the seven relays and their common pin. The power supply's positive terminal serves to energize the relay so the switch will be connected to the 'Normally open' port.

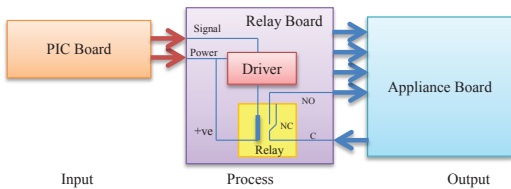


Figure 7: Relay Board Inputs and Outputs. Inputs are shown in red arrows and outputs in blue arrows

The basic plan for the appliances placing is shown below (Figure 8 and Figure 9). The idea to connect the appliance board to the relay board requires the output of eight wires. These eight wires will pass through the relay board and be controlled with an open circuit which connects the active neutral connection (common) to the neutral connection of the appliances. This choice of letting the neutral wire pass through the relay board instead of the live wire was based on the safety issue that live wires carried current to flow through the appliance while the neutral wires were acting as grounded nodes. Green/yellow earth wires serve to discharge unsafe amount of current in case of faults such as fuse blows and tripping of the circuit breaker.

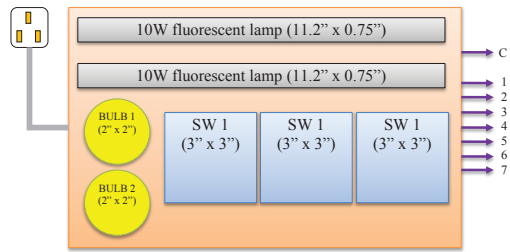


Figure 8: Appliance Board Plan. All wires led out of the board are of neutral connection

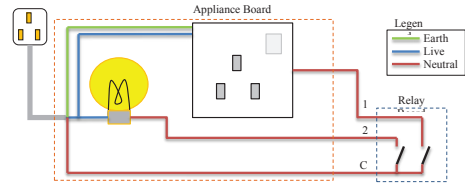


Figure 9: Illustration of Connections from Appliance Board to Relay Board

D. User Interface Design

Application for Windows OS can be easily developed using Microsoft Visual Studio. As the OS is installed on personal computers, screen size is not a matter unless it exceeds the maximum window. For this system, the strategy is to create a GUI that does not differ too much from the existing OS interface.

One of the main limitations of developing an application for Android OS is the small screen size. Since the OS is mainly installed on smartphones, the screen size width and height may barely exceed the size of the human palm. Unless the application is installed in larger Android devices such as tabs, the design of the application must assume an average Android device size such as an Android phone.

IV. RESULTS

A. Connecting PIC Microcontroller to Computer

Bluetooth SPP connections are an emulation of serial cables that provides substitutes for RS232 connections on computers. This concept will create a two way port where the connected devices can exchange data. Using Visual Basic, the GUI application will connect to the PIC microcontroller by opening the port from the computer where the Bluetooth Serial Adapter has been assigned, for example, COM2.

```
// If SerialPort.IsOpen Then
//SerialPort.Close()
// End If
// SerialPort.Open()
```

After establishing a connection successfully, data can be sent through to the other device and vice versa. For the VB application to detect any input data coming from the other device, a handler function must be created. This function may contain logic depending on how the received data is expected to be processed. As the handler is created on a different thread than the main thread, accessing objects from

this thread requires invoke of the function that accesses other thread.

```
// Private Sub SerialPort_DataReceived(ByVal sender As System.Object, ByVal As System.IO.Ports.SerialDataReceivedEventArgs) Handles SerialPort.DataReceived
// Dim temp as String = SerialPort.ReadExisting
// BeginInvoke(New myDelegate(AddressOfoutFunction), New Object() {})
```

B. Mail Parsing in VB

Parsing means the interpretation of data to extract its meaning. In this project, codes sent through the e-mail will be processed based on the limited set of codes with format shown in the next table. Extraction of messages from the inbox of the mail uses Google Atom feed (Table 1).

Table 1
Email to Server Commands

ON	OFF	Explanation
LO	LF	Controls all lights in the house simultaneously
BOxxx	BFxxx	Control bedroom lights
TOxxx	TFxxx	Control toilet lights
VOxxx	VFxxx	Control living room lights
COxxx	CFxxx	Control corridor lights
AOxxx	AFxxx	Control appliances or switches

Characters denoted by 'x' represent numbers to indicate appliances.

To promote security to the user, only codes sent with the user defined verification code will be recognized; thus extending the format to include a four pin number in front, '1234: BO23;VO2;TF1;AF3', where 1234 is the example verification code.

To send multiple codes at a time, codes should be separated by the semi-colon symbol, ';'. Example of valid e-mail code is such as, 'BO23;VO2;TF1;AF3'. Any wrong format or syntax of received data will not be processed and overlooked. However, after processing the e-mail codes, an updated condition of the house will be sent to the user defined address; either an e-mail address or a phone number. The details can only be set via the server program which is the VB application. Any user may check the house condition but the feedback will only be sent to one address or phone number to prevent misuse of the system. This ensures that only the owner of the system will know the condition of the house. To check for appliances condition, just send an e-mail to the server's address with the subject "CHECK".

To promote security to the user, only codes sent with the user defined verification code will be recognized; thus extending the format to include a four pin number in front, '1234: BO23;VO2;TF1;AF3', where 1234 is the example verification code. The figure below shows testing demonstration of using the e-mail control function. Even if the user sends the mail in lower case characters, the program can still understand the command. This is permitted by using the '.ToUpper' function for string manipulation technique in VB before processing the commands.

C. Appliances Mapping

The process of mapping appliances requires the use of an example house map. With that, a graphical map of the house

needs to be created and in the case of this project, is created using Microsoft Visio 2010. The integration of the map into the VB application is as an image.

The final VB GUI (Figure 10 and Figure 11) contains three main tabs including main, security, and settings. The main tab displays the house map where users may control the home appliances. The security tab contains the camera window where a third-party IP camera can be viewed and controlled to view some areas of the house. Settings tab is where settings for the connection and user data may be changed.



Figure 10: VB GUI Main Tab. To control appliances, click on the named objects on the map

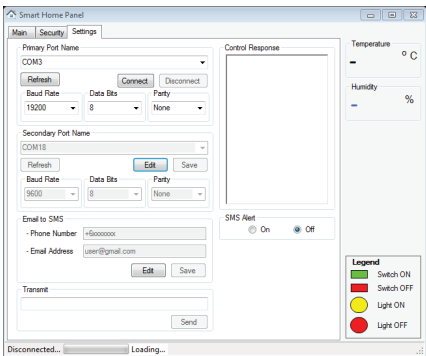


Figure 11: VB GUI Settings Tab. Editable controls for users to control connections and debug

The "Primary Port Name" determines the COM port to connect to the main control panel while the "Secondary Port Name" is the port used for connection from the phone or Android device. For debugging purposes, the "Control Response" box is shown to allow users to observe sent and received data. The additional "Transmit" box is also for debugging purposes to manually send data across the primary port. "Email to SMS" sets the phone number or e-mail address the user wants the system to feedback to. If the "SMS Alert" is turned ON, then the feedback message will be sent to the phone number. Else, the message will be sent to the e-mail address.

D. Android User Interface

Android devices have touch-screen input panels and depending on the device, the sizes of their screens are mostly limited. In this project, the Android User Interface (Figure 12) is designed to suit Android phones. This resulted in a more compact and simple platform for users to navigate and control. Instead of connecting directly to the main control panel, the Android device will control home appliances by connecting to the server.

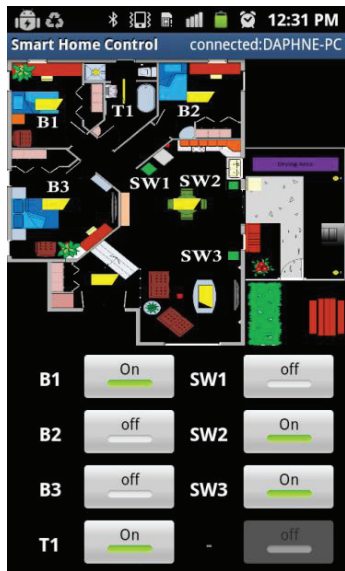


Figure 12: Android Application Main Page. Toggle buttons shows state of home appliances

E. PIC and Relays

The development of the PIC and Relay board was more focused on simulating the electronic circuits using Proteus ISIS and with the successful designs, the PCB design was created using Proteus ARES. MPLAB IDE was also used to create the application for the PIC to provide viable inputs and outputs.

When data is received, the PIC will check, using 'case' statements to check the character received. If the received character is the same as one of the case statements, the program under the statement will be processed.

```
// switch(input) {  
// case 'T':  
//     //get temperature readings;  
//     break;  
// case '1':  
//     //turn on appliance 1;  
//     break; }  

```

For calculation of the sensors, equations were obtained through analysis of the sensors datasheet. The temperature equation was obtained using numerical method's extrapolation technique using data provided from the sensor's manufacturer. From the graph, the data is obtained and calculated using linear regression to obtain A and B for

the general equation shown below.

$$y = A + Bx \tag{1}$$

The calculations for humidity sensor obtained $A = -11.504$, $B = 30.855$; where the humidity in percentage, $y = -11.504 + 30.855x$, where x is the voltage value. Meanwhile, calculations for temperature sensor obtained $A = 5.2346$, $B = 9.1508$; meaning that the temperature in degree Celsius, $y = 5.2346 + 9.1508x$, where x is the voltage.

Figures below (Figure 13, Figure 14, Figure 15, Figure 16, Figure 17 and Figure 18) shows the simulation and design of the PIC and relay board.

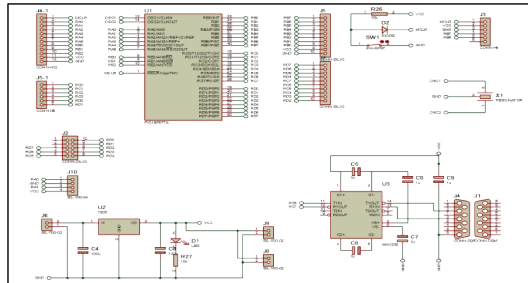


Figure 13: Design of PIC Circuit in Proteus ISIS. The circuit includes a voltage regulator and a MAX232 driver

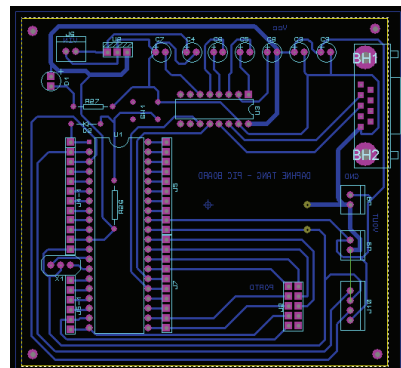


Figure 14: PIC Board PCB Design in Proteus ARES. Separation of the copper paths should meet minimum requirements

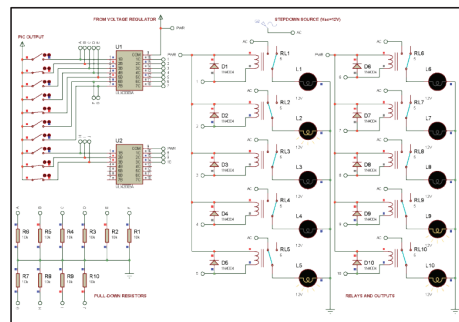


Figure 15: Simulation of Relay Board in Proteus ISIS. Simulation shows circuit of ten outputs. Only seven were used in the actual prototype

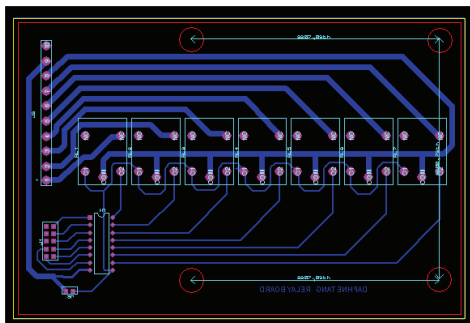


Figure 16: Relay Board PCB Design in Proteus ARES. Land widths for AC wires are of size 80th

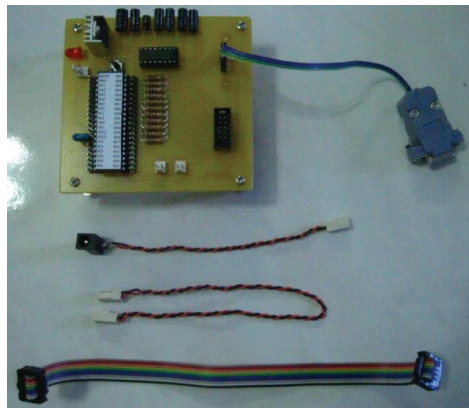


Figure 17: .PIC Board PCB Top View. The set includes RS232 male adapter, 3.5mm adapter jack, 10-way rainbow cable and a 2-pin Molex cable

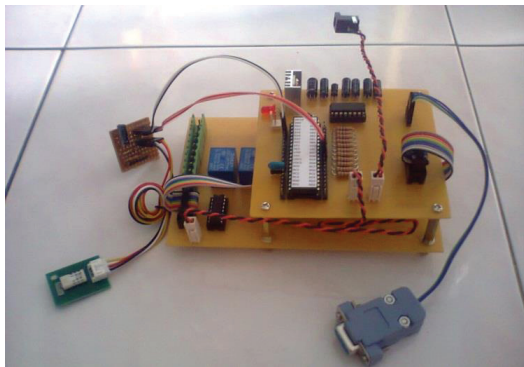


Figure 18: Combined Relay and PIC Board. Connection of the PIC with the relay board was through the 10-way rainbow cable

F. Appliance Board

The base of the appliance board is made of wood. Using half inch screws, all appliances were installed onto the base (Figure 20 and Figure 21). The wiring was done based on the electrical diagram shown below (Figure 19).

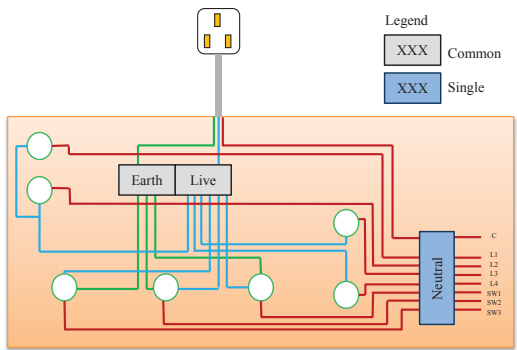


Figure 19: Wiring Schematic. Connections were done using 2.5mm multicore wires

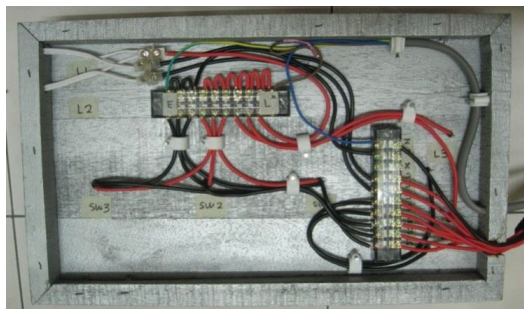


Figure 20: Wiring Diagram of Appliance Board. Wires were properly arranged and clipped on to keep the wires in place



Figure 21: Appliance Board Top View. The board features two 8W fluorescent, two 20W light bulbs and three three-pin switches

G. Prototyping Result

The integration of all the hardware components was done by connecting all the hardware interfaces to the PIC Board. The appliance board was connected to the PIC Board via the relay board and powered up from AC source using three-pin plug. A miniature map was developed to show the appliance state from the PIC using LEDs (Figure 22, Figure 23, Figure 24 and Figure 25).

V. DISCUSSIONS AND ANALYSIS

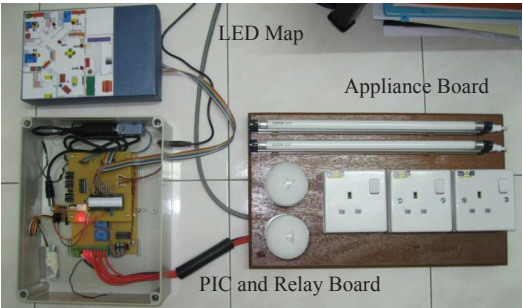


Figure 22: Combined Hardware Components. Connection of all the system's elements for testing

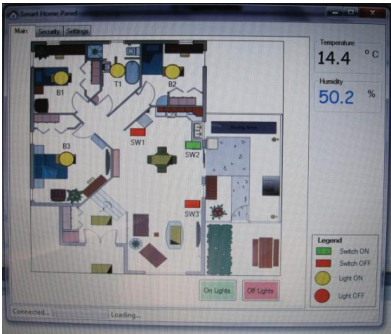


Figure 23: VB Application Testing State. Yellow or Green objects show appliances which have been turned ON



Figure 24: LED Output from PIC on Map. Referring to previous figure, the corresponding outputs are switched ON



Figure 25: Appliance Board Output. Referring to previous figure, the corresponding outputs are switched ON

It is clear that the final system design could be controlled using either an Android device which is wirelessly connected to the server, or using the server computer itself. This feature of the system fulfills the objective to allow remote control characteristics in home appliances. In addition, the server's application was also programmed to retrieve and perform user commands sent from e-mail periodically. This feature was further enhanced to allow the user to query conditions of home appliances with the option for the system to reply to the e-mail address or a phone number via SMS. The use of the SMS service requires the expenditure of RM0.04 per message. With all these features integrated into the smart home system, the problem statement of replacing conventional switches is also being solved thus allowing people with disabilities the option to easily control home appliances.

In terms of system's concept, the project resembled the idea of [1] in the sense that the appliances connection to the main control board was wired and also the use of relays to control switches. However, difference can be found in this system which uses PIC16F877A instead of the AVR Atmega 168 microcontroller used by [1]. The function of the PIC microcontroller acts only as an IO controller where the signals sent to the relays are controlled by the PIC.

Likewise, [1] only provided UI for Symbian OS phones where other OS users could not use the system. This project presented two GUI applications, installed on Windows 7 OS and Android OS to offer larger platform variant. The Window 7 application acts as the main server where full control of the system is dependent on. The Android application was created to allow an Android device to act as an additional controller that allows users to control home appliances by connecting to the main server.

In terms of connection variant, this project used a mixture of wired and wireless connection, where wired connection will run from the home appliances to the main control board while wireless connection exist in between the main control board to the UI platform, and from the Android device to the main server.

A. Software

The algorithms used in developing the main server application features complex levels. To cope with that, the application development utilizes VB.net feature to create multithreading applications. This improves the user interface responsiveness and promotes better functionality. Statements will not be blocked as objects are not tied to one thread. The server had to cope with checking and forwarding of mail, user clicking buttons, periodic communication with the PIC board and also the Android device. This application features a total of ten threads including the main application thread. However, not all functions can be added to individual threads.

On the Android application, the layout was created using XML while the class files to identify action and logic was created in Java class. The Android application connects to the server using Bluetooth SPP to transfer serial data to the server. With limited screen size, the application featured two screens where users can switch between them by

flipping the screen to left or right.

In terms of the PIC microcontroller, the integration of sensors into the circuit used the ADC module of the PIC. Two ports, AN1 and AN2 were used to input voltage readings from the temperature and the humidity sensors. The conversion of the data was based on the calculation as shown using equation 4.1. Even though the sensors were not precise, the integration of the sensors were according to the specification as specified in the sensor's manual and fulfilled the objective to implement two way communication in the system. The server will periodically prompt for sensors reading using specific commands and the PIC microcontroller will reply accordingly in a proper format.

B. Hardware

Integration of the relay board and PIC board was achieved by connection via a ten way rainbow cable. The rainbow cable was used because it is more flexible and durable compared to single core copper wires. Since the relay board, PIC board and the LED outputs shared a same voltage and current source, an adapter of 5 Volts and 2 Amperes was used to power the circuit and via testing, was enough to power up the whole system without heating up. However, the relays showed signs of slight heat. This was one of the disadvantages of using mechanical relays. A way to solve this was to design a new circuit that replaces the relays with transistors. This method could also eliminate the relay switching sound.

When fixing the appliance board and the wirings, caution must be put when cutting the wires. It must be ensured that no copper were accidentally removed as it affects the power rating of the cable. In terms of connection, it is better to not let the live wire be passed through the relays. This is because live wire carries the current for the AC connection. However, doing so disallows users from directly changing any components such as a light bulb when the main switch powering the appliance board is still active. This is a caution that must be taken to prevent electric shock.

VI. PROJECT SIGNIFICANCE AND IMPLICATIONS

Comparing to past literatures, this project integrated valuable ideas in developing a low cost smart home system. With a highly remote control platform, the system added also control via Internet features which improves the functionality of the system. Users are not only allowed to control but also check for appliances condition. The system is capable of providing feedback via e-mail or even through SMS to the user's phone number.

Adding the ideas of the literature [2], the concept of mapping was applied by creating customized house floor maps with indication of the location of the home appliances. This greatly eases the user's control over the appliances instead of a GUI displaying only buttons like the system in literature [1]. Also adopting a different idea from refere [1], another GUI was developed for the popular Android platform. Regardless of that, the microcontroller used created a wider range of applicability for the system. With its lower cost, the system can be integrated in more areas especially in places where it is needed, such as home for the elderly and etc.

VII. CONCLUSIONS

The smart home system was created to allow user's control over home appliances from a remote distance. One of the controllable platforms is the Android device that controls home appliances by connecting to the main server. The main server, which can also control home appliances, is also programmed to perform commands received from the email to control appliances. Apart from that, users may prompt for appliances condition also through email and the reply can be sent to the user's email address or phone number through SMS. All the features stated above has fulfilled the first project objective to design a smart system that allows remote controlling of home appliances. The second objective to portray a two way communication between the GUI and the microcontroller was achieved by adding temperature and humidity sensors to the microcontroller circuit where its values are prompted periodically by the GUI application. By accomplishing the objectives, the problem statement to replace control of home appliances via conventional switches have been solved.

VIII. RECOMMENDATIONS

A. Improvements for Hardware

In terms of installation, the system is designed to be installed in new houses where existing electrical wirings are still not laid out. This creates a disadvantage where there is a limited range of customers that could purchase this product. To solve this issue, further research needs to be done to create or modify this system so that it can be installed into homes with existing electrical wirings.

Also, due to time constraints, the system's development only achieved the control of two state switches at home such as lighting and switches. Improvement of the system can be achieved by integrating techniques to control IR devices such as the television, the air conditioner or even the washing machine. These additions would be able to improve the system's range of control at home. Even further additions could include security features with the installation of sensors and alarms around the house. This would require the design of a flexible PIC board that is easily extensible so that there are enough ports to support the system.

Apart from that, seeming that this project aimed at producing a low cost smart home system, the system's integrity is questionable. However, with a proper power source that can supply enough current, the system can still be applied in domestic homes. To target larger enterprises or locations, the system may be improved to integrate a more industrial applicable controller such as PLC. However, this dramatically increases the cost of the system especially if there is the need to control a large number of appliances.

B. Improvements for Software

In terms of the system's software, this project still utilizes third-party web services such as Google's Atom feed and the SMS mail. If this system was to be improved into an independent product, these two services need to be created so as to cut off the dependencies of the system. If the services were discontinued, the system's server application needs to be reprogrammed to find for replacements of those services. This may be a tedious process but by making the

system independent, this product can be a stable stand-alone system.

C. Applications for Smart Home System

The current smart home system targets domestic homes where ease of control is of major concern. This includes homes for people with disabilities, elderly, or for anyone with difficulties moving around. Nevertheless, these people must still at least have functionality of at least one finger as it is possible to navigate the Android device GUI with a single finger.

Home installed with this system will have a higher sense of controllability over their home appliances even when they are not at home. For instance, if a family went for a vacation, leaving their house vacant for some time, it could easily be targeted by robbers. With this system, home users can monitor and control the appliances in the house from anywhere with Internet connection. Switching on some appliances from time to time may make the house seem occupied and minimize the chances of having unwanted visitors.

ACKNOWLEDGMENT

I would like to thank Universiti Teknikal Malaysia Melaka (UTeM) which has financially supported this research to be accomplished.

REFERENCES

- [1] R. Piyare and M. Tazil, "Bluetooth Based Home Automation System using Cell Phone," in *Consumer Electronics*, 2011, pp. 192-195.
- [2] K. Atukorala, D. Wijekoon, M. Tharugasini, I. Perera, and C. Silva, "SmartEye Integrated Solution to Home Automation, Security and Monitoring through Mobile Phones," in *2009 Third International Conference on Next Generation Mobile Applications, Services and Technologies*, 2009, pp. 64-69.
- [3] J. Ojasalo and N. Suomalainen, "Better Technologies and Services for Smart Homes of Disabled People: Empirical Findings from an Explorative Study among Intellectually Disabled," in *Applied Sciences*, 2010, no. 2, pp. 251-259.
- [4] P. Panek, W. L. Zagler, C. Beck, and G. Seisenbacher, "Smart Home Applications for disabled Persons - Experiences and Perspectives," in *Electronics*, 2001, pp. 1-10.
- [5] J. Fonseca, A. Mota, V. Santos, P. Bartolomeu, M. I. O. Servi, and D. Electrnica, "B-Live - A Home Automation System for Disabled and Elderly People," in *IEEE*, 2007, pp. 333-336.
- [6] R. Kadouche, M. Mokhtari, S. Giroux, and B. Abdulrazak, "Personalization in Smart Homes for Disabled People," in *2008 Second International Conference on Future Generation Communication and Networking*, 2008, pp. 411-415.
- [7] M. Marschollek et al., "Multimodal home monitoring of elderly people – first results from the LASS study," no. 1, 2007.
- [8] M. Chan et al., "Smart House Automation System for the Elderly and the Disabled," *Neural Networks*, no. IEEE, pp. 1586-1589, 1995.