CELL PHONE CONTROLLED ELECTRICAL APPLIANCES
(GSM CONTROLLER)

TEO CHE SHEN

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

MAY 2011

DECLARATION

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

Signature : ______________________

Name : TEO CHE SHEN

ID No. : 07UEB05101

Date : 10th May 2011

APPROVAL FOR SUBMISSION

I certify that this project report entitled **“CELL PHONE CONTROLLED ELECTRICAL APPLIANCES”** was prepared by **TEO CHE SHEN** 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 : Prof. Dr. Lim Yun Seng

Date : ___________________________

Specially dedicated to

my beloved grandmother, mother, father and friends

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, Prof. Dr. Lim Yun Seng for his invaluable advice, guidance, tolerance, acceptance and his enormous patience throughout the development of the research.

PROJECT TITLE IN CAPITAL LETTER TITLE TO BE THE SAME AS FRONT COVER

ABSTRACT

The initial idea of this project is to create a low cost remote house automation system. By implementing both hardware and software components, the author and his colleague will able to create an efficient equipment that provides safety and convenient to home users. Mobile phones can be used to send Short Message Services (SMS) and it will be receive by a housed GSM mobile phone which functions as a GSM modem. The message which is received will be in PDU format. The product uses GSM mobile phone to communicate with a microcontroller that will control the instruction for the home automation modem, ultrasonic switch and burglary notification system. The GSM modem and the microcontroller are connected via serial RS232 cable. There will be some standard instruction in messages that can be interpreted by the microcontroller that is connected to the housed GSM mobile phone. Programming will be done in order to gives priority to the latest message instruction whereby it will be executed. The interpreted message instruction will allow certain outcome to be sent from the microcontroller to a home automation modem, ultrasonic switch and burglary notification system. Then, home automation modem and ultrasonic switch control switching of home electrical appliances by mean of frequencies. The PLC modem will be a form of simple circuit that can be plugged in to any power line outlet in a building. As for ultrasonic switch, it can be connected to anywhere of a house. Same goes to the burglary notification system. This product allows user who is away from home, to close or on desired home electrical appliances remotely using any mobile phones and also provide a security system to user. Apart from that, this product even provides convenient for disable individual who cannot move freely to switch off some particular household

appliances. Moreover this is environmentally friendly where unused energy can be turned off rather than being wasted when nobody is at home.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

ADC	analogue to digital converter
AT	attention
BSS	base station subsystem
DC	direct current
GPRS	general packet radio service
GSM	global system for mobile communication
IC	integrated circuit
ISR	interrupt subroutine
MAP	mobile application part
OSS	operations support system
PC	personnel computer
PDU	protocol description unit
PLC	power line communication
PSTN	public switched telephone network
RS232	recommended standard 232
SCR	silicon controlled rectifier
SMS	short message service
SMSC	short message service centre
TTL	transistor-transistor logic
UART	universal asynchronous receiver/transmitter
USB	universal serial bus

CHAPTER 1

INTRODUCTION

1.1 Background

Due to busy lifestyle, people seem to being forgetful on small alpha matter such as switching off household electrical appliances. This is an act of wasting energy. Apart from that, electrical apparatus such as iron, microwave and oven can be dangerous as kept on for long period especially when nobody around. Moreover, house break in seems to be a serious crime in this era. Hence, House Automation System is vey efficient solution towards above problem. This is a simply low cost and only simple installation is required.

The major problem of the current issue is how to switch on or off those household appliances remotely. For instant, closing an electronic gate but having an external remote control seems to be inconvenient and bulky. It will be more practical if there is a modem that can receive certain specified messages via message from any mobile phones to perform the controlling on switching. Hence, GSM modem (GSM mobile phone) can be introduced.

The GSM controller provides mobility to remote controlling and the PLC modem provides direct access to power line without rewiring. Apart from PLC, ultrasonic switch is also alternative to perform remote switching. Moreover, a burglary notification system can increase the security level at any resident area. By combining these technologies, Remote Household Automation System is very useful and efficient because it is very user-friendly.

1.2 Objectives, Aims and Motivation

The objective of the project is to build GSM modem that can communicate with PLC modem and ultrasonic switches to serve as a Remote House Automation System. Apart from that burglary notification system is also integrated to the project.

The aim of this assessment is to provide controlling of home appliances remotely and will also enable home security against intrusion in the absence of home owner.

The motivation is to facilitate the users to automate their homes having ubiquitous access with availability for development of a low cost system. The home appliances control system provides remote access to electrical appliances. In addition there was a need to automate home so that user can take advantage of the technological advancement in such a way when an individual is out from home. Therefore this paper proposes a system that allows user to be control home appliances SMS using GSM technology.

1.3 Focus and Task Distribution

This project consists of GSM Controller, Power Line Communication, ultrasonic switch and burglary notification system . Teo Che Shen will focus on GSM Controller, Burglary notification system details and some of the ultrasonic switch meanwhile teammate, Chew Kean Chai will focus on Power Line Communication details and some of the ultrasonic switch. During the accomplishment of this project project participants will help each other to ensure the outcome can be achieved.

1.4 Project Outcome

- To develop hardware design skills.
- To develop software design and writing skills

- To build teamwork among teammate

CHAPTER 2

LITERATURE REVIEW

2.1 Global System for Mobile Communication (GSM)

In 1982, the European Conference of Postal and Telecommunications Administrations (CEPT) created the Groupe Spécial Mobile (GSM) to develop a standard for a mobile telephone system that could be used across Europe. GSM (Global System for Mobile Communications: originally from Groupe Spécial Mobile) is the most popular standard for mobile telephony systems in the world.

2.1.1 The Structure of GSM Network

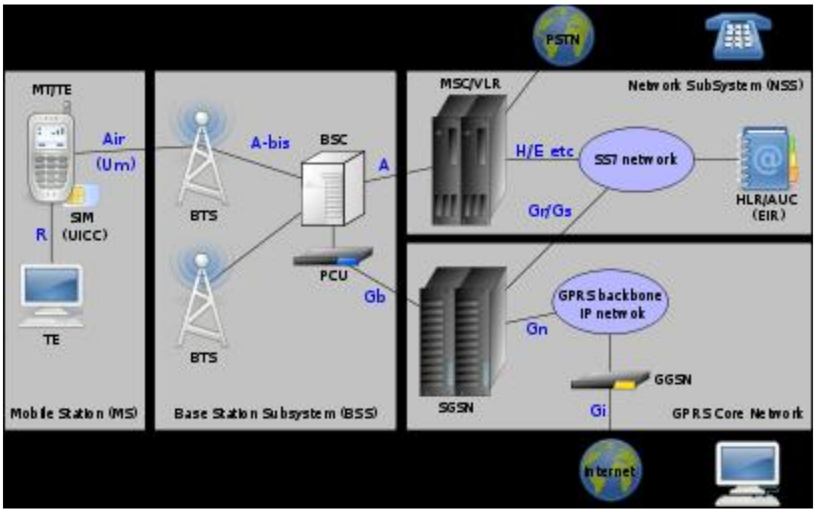


Figure 2.1: The Structure of GSM Network

The network is structured into a number of discrete sections:

- The Base Station Subsystem (the base stations and their controllers).

  It handles handling traffic and signalling between a mobile phone and the network switching subsystem. The BSS carries out trans-coding of speech channels, allocation of radio channels to mobile phones, paging, transmission and reception over the air interface and tasks that are related to the radio network.

- The Network and Switching Subsystem (the part of the network most similar to a fixed network).

  It is also referred as core network. It provides functions such as out call switching and mobility management for mobile phones roaming on the network of base stations. It allows mobiles devices to communicate with each other and as well to telephones through wider Public Switched Telephone Network (PSTN).

- The General Packet Radio Service (GPRS) Core Network (the optional part which allows packet based Internet connections).

  GSM mobile phones use GPRS system for transmitting IP packets. The GPRS core network is the centralized part of the GPRS system.

- The Operations Support System (OSS) for maintenance of the network.

  It is used by telecommunications service providers by supporting processes such as maintaining network inventory, provisioning services, configuring network components, and managing faults.

  Apart from that, the complementary term Business Support Systems or BSS is a newer term and typically refers to "business systems" dealing with customers, supporting processes such as taking orders, processing bills, and collecting payments. The two systems together are often abbreviated as BSS/OSS.

Nevertheless in general, an OSS covers at least the application areas:

- Network Management Systems
- Service Delivery
- Service Fulfilment, including the Network Inventory, Activation and Provisioning
- Service Assurance
- Customer Care
- Billing

2.1.2 GSM Service - Short Message Service (SMS)

In this assessment, only the main GSM service which is the Short Message Service (SMS) will be discussed.

Short Messages is the most used data application on mobile phones. User can use mobile phones or other GSM related devices to SMS text message to other user. The messages are usually sent from mobile devices via the Short Message Service Centre (SMSC) using the MAP protocol.

Basically, SMSC is a central routing hub for Short Messages. Many mobile service operators use their SMSCs as gateways to external systems, including the Internet, incoming SMS news feeds, and other mobile operators.

SMSC functioned by practicing "store and forward" mechanism. Whenever a message is sent but the recipient is not achievable the SMSC queues the message for later retry. However, there are some SMSCs provide a "forward and forget" option where transmission is tried only once.

2.1.3 Cellular Radio Network

Mobile phones can connect to GSM (cellular network) via cells in the immediate vicinity which each cells vary in coverage area by according to the implantation environment. Basically, there are five different cell sizes in the GSM network,

- Macrocells are cells where the base station antenna is installed on a mast or a building above average roof top level
- Micro cells are cells whose antenna height is usually under average roof top level and are typically used in urban areas.
- Picocells are small cells whose coverage diameter is a few dozen meters and mainly used in indoors.
- Femtocells are cells designed for use in residential or small business environments. It can connect to the service provider’s network via a broadband internet connection.
- Umbrella cells are used to cover shadowed regions of smaller cells and fill in gaps in coverage between those cells.

2.1.4 Hayes Command Set (AT Command)

The Hayes command set is a specific command-language originally developed for the Hayes Smartmodem 300 baud modem in 1977. The command set consists of a series of short text strings which combine together to produce complete commands for operations such as dialing, hanging up, and changing the parameters of the connection. Most dialup modems follow the specifications of the Hayes command set. Nowadays, it is also called as AT command sets.

2.2 Transistor–Transistor Logic (TTL)

TTL is digital circuits that built from bipolar junction transistors and resistors. It is called TTL because transistors perform function both functions as the logic gating and the amplifying. It is widely used in many applications such as computers, controls equipment, test equipment and instrumentation, electronics, synthesizers, and many more.

2.3 MAX232

The MAX232 is a type of integrated circuit that converts signals from an RS-232 serial port to signals suitable for use in TTL compatible digital logic circuits. It has dual driver and receiver that typically convert the RX, TX, CTS and RTS signals. It is helpful to understand what occurs to the voltage levels of the MAX232.

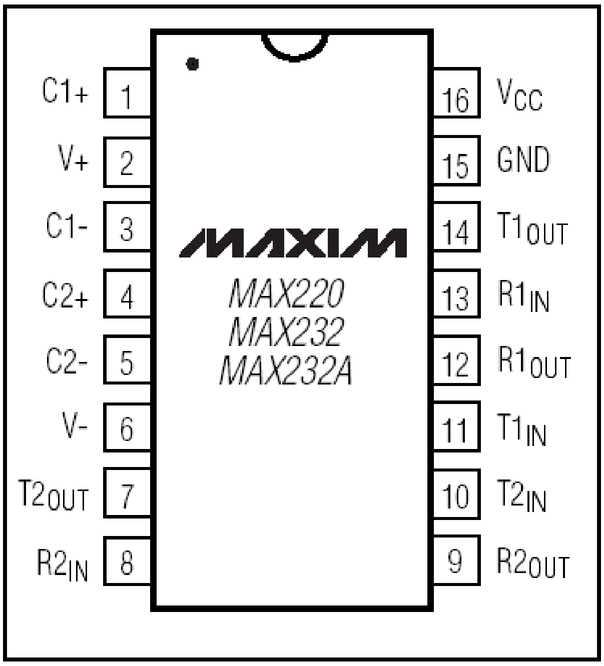


Figure 2.2: MAX232 IC

RS232 Line Type & Logic Level	**RS232 Voltage**	**TTL Voltage to/from MAX232**
Data Transmission (Rx/Tx) Logic 0	+3V to +15V	0V
Data Transmission (Rx/Tx) Logic 1	-3V to -15V	5V
Control Signals (RTS/CTS/DTR/DSR) Logic 0	-3V to -15V	5V
Control Signals (RTS/CTS/DTR/DSR) Logic 1	+3V to +15V	0V

Table 2.1: RS232 Line Types and Logic Level

2.4 Universal Asynchronous Receiver/ Transmitter (UART)

Western Digital made the first single-chip UART WD1402A around 1971. A universal asynchronous receiver/transmitter is a type of asynchronous receiver/transmitter that translates data between parallel and serial forms. UARTs are commonly used in conjunction with other communication standards such as RS-232.

Usually, UART is an individual integrated circuit used for serial communications throughout a computer or peripheral device serial port and it is commonly included in microcontrollers. As for dual UART (DUART), two UARTs will be combined into a single chip.

2.4.1 Transmitting and Receiving Serial Data

UART controller is the key component of the serial communications subsystem of a computer. The UART takes bytes of data and transmits the individual bits in a sequential trend. At the end of destination, a second UART reassembles the bits into complete bytes. Serial transmission of digital information (bits) through a single wire or other medium is much more cost effective than parallel transmission through multiple wires. A UART is used to convert the transmitted information between its sequential and parallel form at each end of the link. Every single UART contains a shift register which is the fundamental method of conversion between serial and parallel forms.

Typically, UART usually does not directly generate or receive the external signals used between different items of equipment. Hence, separate interface devices are used to convert the logic level signals of the UART to and from the external signaling levels. External signals may be of many different forms and examples of standards for voltage signaling are RS-232, RS-422 and RS-485. For embedded systems communications, UARTs are commonly used with RS-232. Many chips provide UART functionality in silicon, and low-cost chips exist to convert logic level signals (such as TTL voltages) to RS-232 level signals (for example, Maxim's MAX232).

2.5 Recommended Standard 232 (RS-232)

Basically, RS-232 is a standard in telecommunication especially in computer serial ports. It allows serial binary single ended data and control signal that connect in between data terminal equipment and a data circuit terminating equipment.

Figure 2.3: RS232 Serial Cable

2.5.1 Connector

Data Terminal Equipment (DTE) or Data Communication Equipment (DCE) define at each device which wires will be sending and receiving each signal. The standard recommended male connectors with DTE pin functions, and modems have female connectors with DCE pin functions.

2.5.2 Pin-Outs for RS232

Signal			Origin		DB-25 pin
Name	Typical purpose	Abbreviation	DTE	DCE	
Data Terminal Ready	OOB control signal: Tells DCE that DTE is ready to be connected	DTR	-		20
Data Carrier Detect	OOB control signal: Tells DTE that DCE is connected to the telephone line	DCD		-	8
Data Set Ready	OOB control signal: Tells DTE that DCE is ready to receive commands or data	DSR		-	6
Ring Indicator	OOB control signal: Tells DTE that DCE has detected a ring signal on the telephone line	RI		-	22
Request To Send	OOB control signal: Tells DCE to prepare to accept data from DTE	RTS	-		4
Clear To Send	OOB control signal: Acknowledges RTS and allows DTE to transmit	CTS		-	5

Transmitted Data	Data signal: Carries data from DTE to DCE	TxD	-		2
Received Data	Data signal: carries data from DCE to DTE	RxD		.	3
Common Ground		GND	common		7
Protective Ground		PG	common		1

Table 2.2: Pin-Out for RS232

2.5.3 Serial Cables

Basically, there are no distinct standard to define the maximum length of the cable. However, there is standard that defines maximum capacitance that a compliant circuit can tolerate. Based on particular research, cables which are longer than 15 meters will have unwanted excessive capacitance unless special cables are used. Low capacitance cable can provide full speed communication which can be maintained over larger distances up to 1,000 feet. Hence for longer distances, other signal standards are preferred in order to maintain high speed.

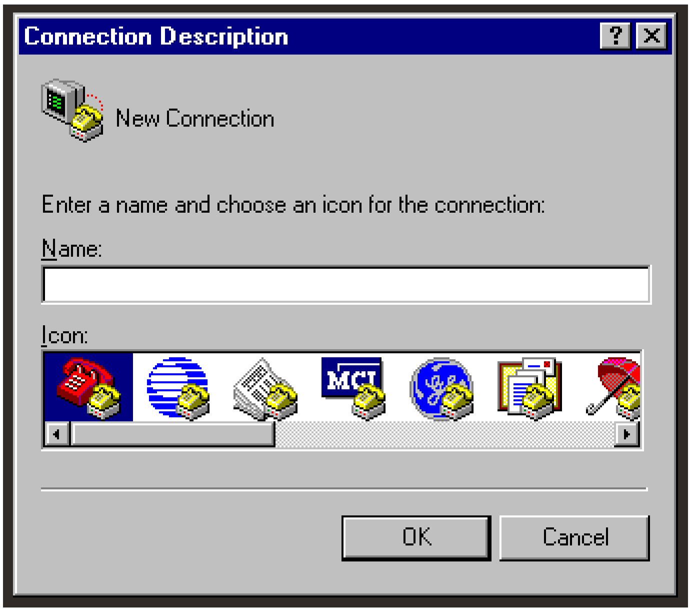
2.5.4 Limitation of Standard

- The power consumption increases due to large voltage swings and requirement for positive and negative supplies increases and hence leads to complicated power supply design.
- Due to the single-ended signaling referred to a common signal ground, it limits the noise immunity and transmission distance.
- Usually, multi-drop connection is not a normal practice that is predefined. Even with multi-drop is defined, there are limitation in speed and compatibility.

- The use of handshake lines for flow control is not really reliable in many other devices because it is initiated for setup and takedown of a dial-up communication circuit purposes.
- There is no method for power sending which is specified. There will be only a small amount of current will be extracted from DTR and RTS lines. Hence, it is only preferable for low power devices

2.6 Hyperterminal

HyperACCESS is the name for a number of successive computer communications software, made by Hilgraeve. In 1995, Hilgraeve licensed a low-end version of HyperACCESS, known as HyperTerminal to Microsoft for use in their set of communications utilities. Nowadays, Microsoft HyperTerminal is a small program that comes with Microsoft Windows. It can be used to send AT commands to your mobile phone or GSM/GPRS modem. By using some AT commands, some desired data from the GSM phone can be retrieved and analyzed.



Connection Description

New Connection

Enter a name and choose an icon for the connection:

Figure 2.4 Screen Shot of Hyperterminal

CHAPTER 3

METHODOLOGY

3.1 Hardware Implementation

3.1.1 Overall Concept of the Design

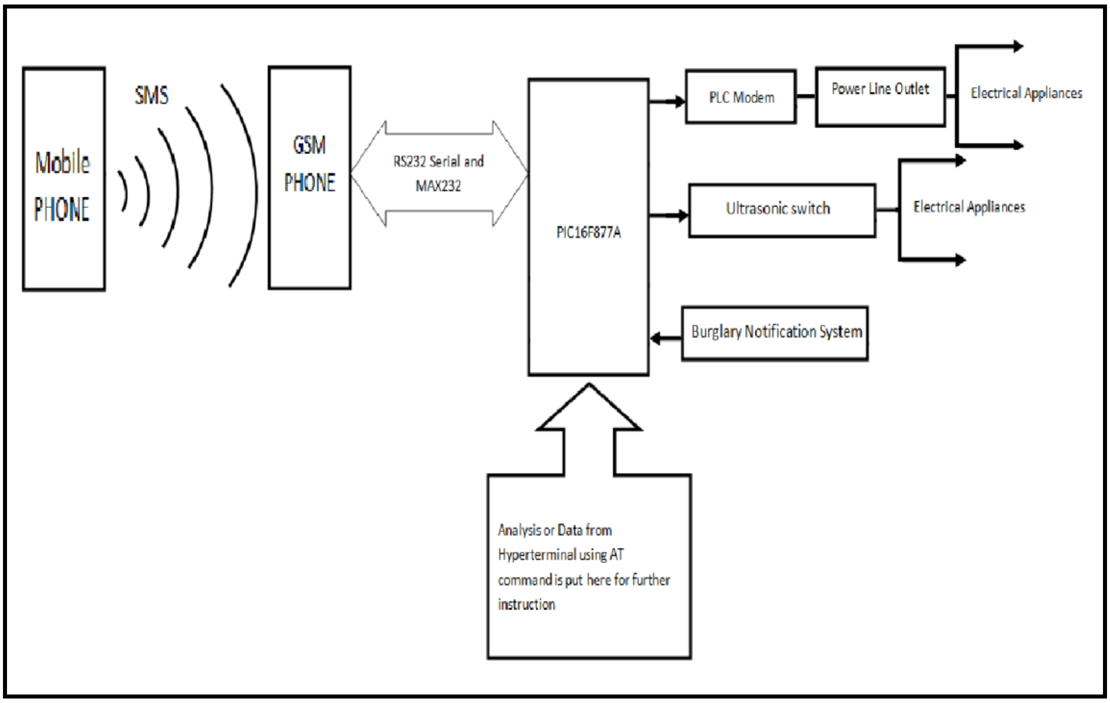


Figure 3.1: The Diagram of Overall Design

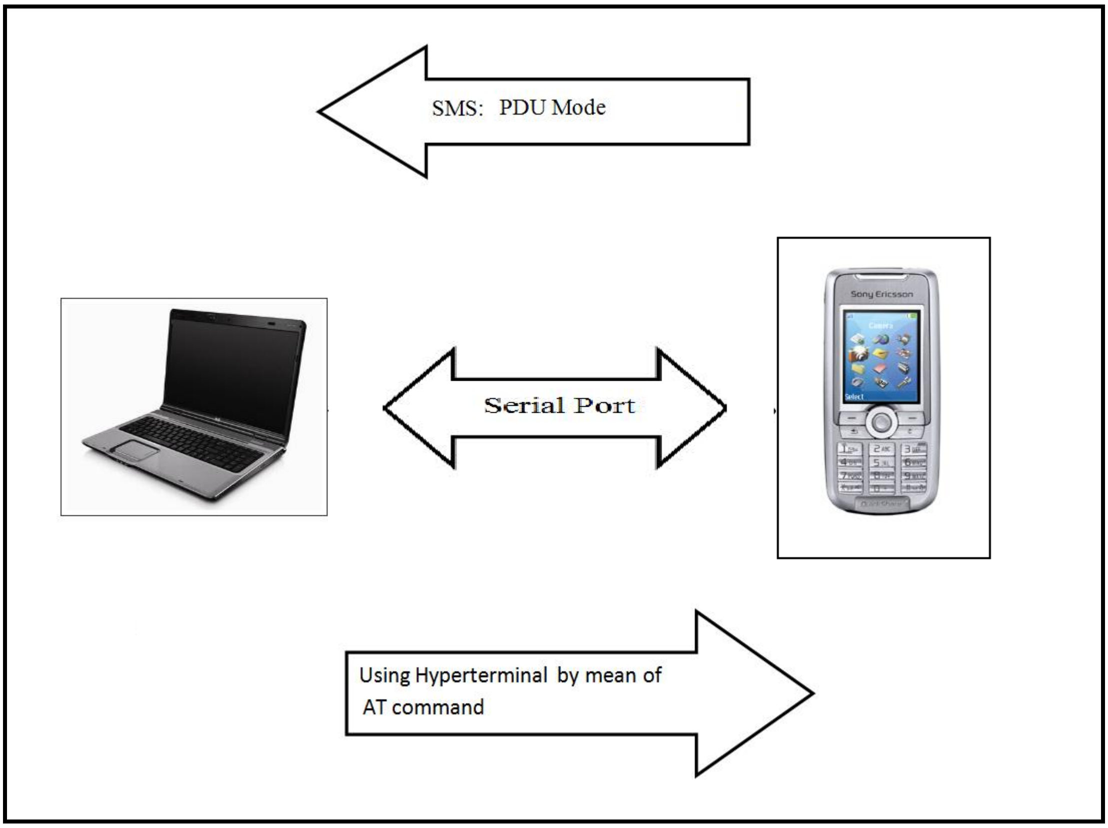


Figure 3.2: The Communication between PC and Mobile Phone

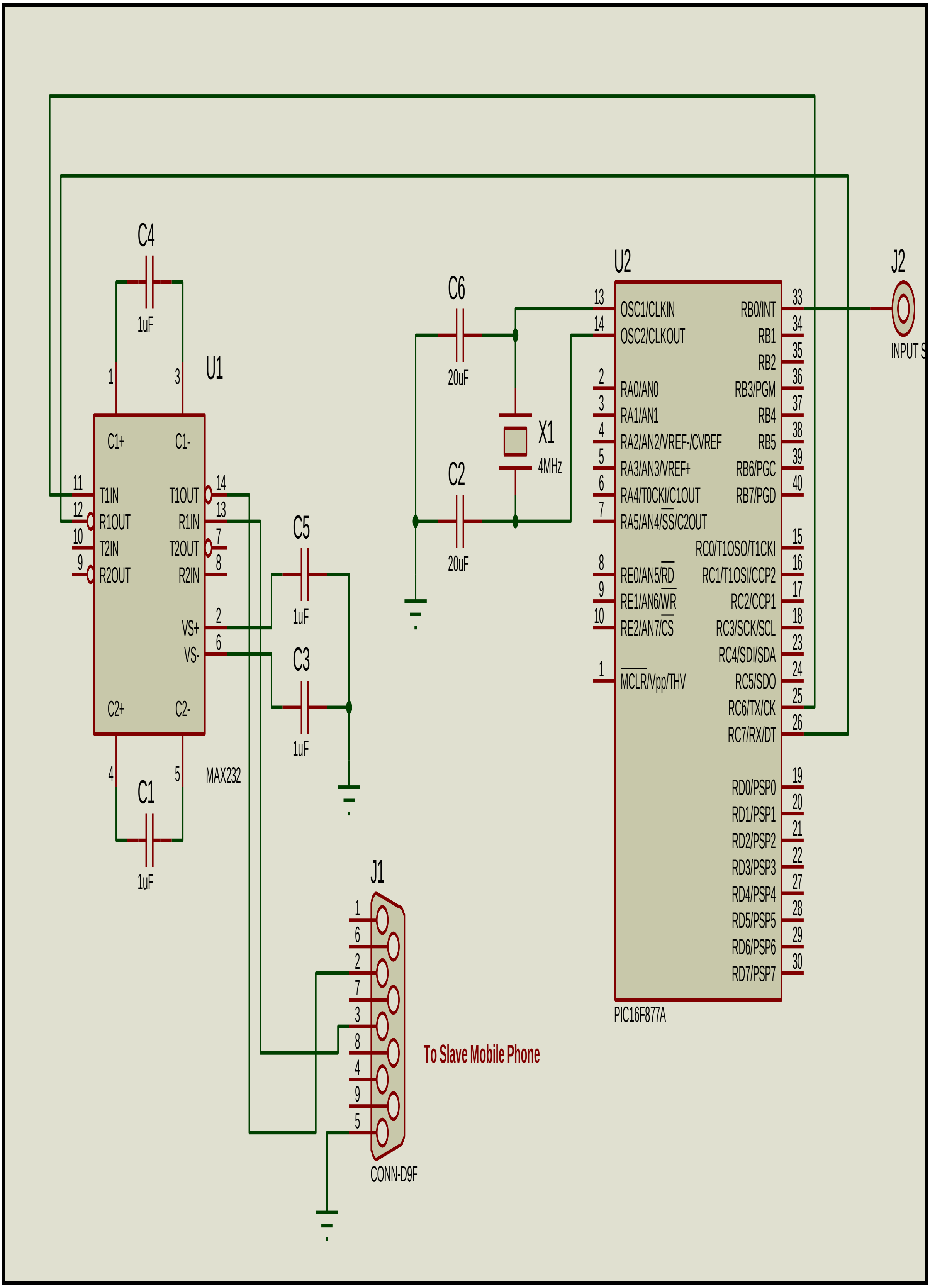


Figure 3.3: Connection between PIC16F877A and MAX232

3.1.2 GSM Mobile Phone – Sony Ericsson K700

Figure 3.4: Sony Ericsson K700

The reason for choosing Sony Ericsson K700 as a GSM modem because this phone has a built-in modem which can be used to connect a PC or PDA many alternatives:

Infrared,

As the phone and a PC running Windows 2000 or XP are first connected via infrared, a generic modern driver is automatically installed by checking the modem properties and Sony Ericsson modem driver is installed.

Bluetooth™ ,

Phone and PC can be connected using wireless technology. This can be done by adding the PC to the phone's list of Bluetooth devices. This procedure is called as pairing or bonding in other product documentation. Hence, the devices are connected within the range limitation of 10 once the Bluetooth is turned on.

USB cable ,

This Sony Ericsson phone provides USB cable phone to PC connection. Initially, the cable must be installed by a accesing to the software driver for the cable.

RS-232 serial

The RS-232 allows connection between the phone and PC.

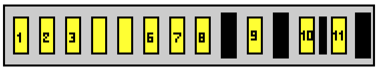


Figure 3.5: Pin-Out for Sony Ericsson K700

Bottom Pin	Name	Direction	Description
1	ATMS	⇐	Audio to mobile
2	AFMS/RTS	⇒	Audio from mobile/RTS
3	CTS/ONREQ	▬	CTS/ Mobile Station **O**n **RE**quest
4	Data in	⇐	Data to mobile (Rx)
5	Data out	⇒	Data from mobile (Tx)
6	ACC in	⇐	Accessory control to mobile. Used as Rx in some models (i.e. T68) for flashing.
7	ACC out	⇒	Accessory control from mobile/handsfree sense. Used as Tx in some models (i.e. T68) for flashing.
8	AGND	▬	Audio signal ground + 0V reference
9	flash	▬	Flash memory voltage + Service (shorted to pin 11 in service cable)
10	DGND	▬	Digital ground
11	Vcc	▬	DC + for battery charging + External accessory powering

Table 3.1: Information of Sony Ericsson K700 Pin-Outs

3.1.3 PIC16F877A

This microcontroller is very powerful with 200 nanosecond instruction execution with yet easy-to-program which stated only 35 word instructions.

This CMOS FLASH-based 8-bit microcontroller packs Microchip's powerful PIC® architecture into a 40- or 44-pin package and is upwards compatible with the PIC16C5X, PIC12CXXX and PIC16C7X devices.

The PIC16F877A features 256 bytes of EEPROM data memory, self programming, an ICD, 2 Comparators, 8 channels of 10-bit Analog-to-Digital (A/D) converter, 2 capture/compare/PWM functions, the synchronous serial port can be configured as either 3-wire Serial Peripheral Interface (SPI™) or the 2-wire Inter-Integrated Circuit (I²C™) bus and a Universal Asynchronous Receiver Transmitter (USART). All of these features make it ideal for more advanced level A/D applications in automotive, industrial, appliances and consumer applications.

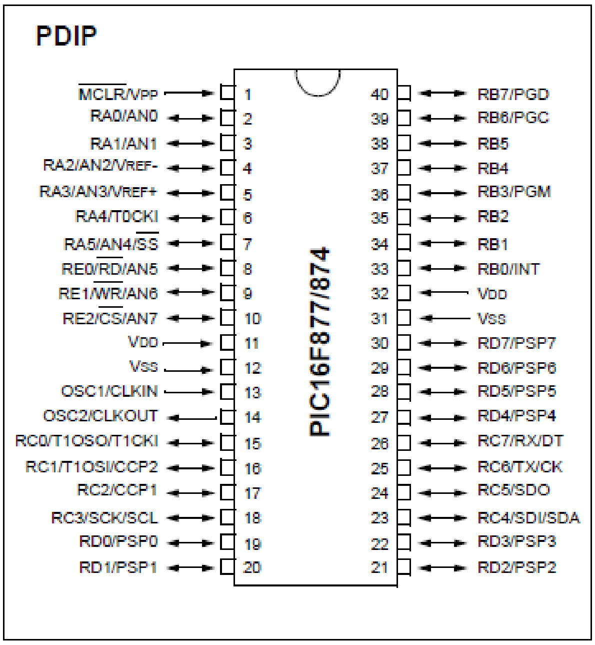


Figure 3.6: PIC16F877A IC

3.2 Software Implementation

3.2.1 ATtention Command (AT Command)

A GSM modem can be controlled my AT command where every command line will starts with “AT” or “at”. Basically, AT commands consist of two types which are basic commands and extended commands. Basic commands are commands that do not start with "+" and it is for controlling wired dial-up modems, for instant

- ATD (Dial)
- ATA (Answer)
- ATH (Hook control)
- ATO (Return to online data state),

Usually, GSM or GPRS modems and mobile phones support an AT commands set (extended commands) that is specific to GSM technology, for instant, SMS-related commands.

AT Command	**Function**
AT+CMGS	Send SMS message
AT+CMSS	Send SMS message from storage
AT+CMGL	List SMS messages
AT+CMGR	Read SMS messages

Table 3.2: SMS Related Commands (Extended Command)

There are four types of AT command operations:

- Test operation.
    - A test operation that is used to check whether certain AT command is supported by the GSM/GPRS modem or mobile phone.
- Set operation.
    - A set operation is used to change the settings used by the GSM/GPRS modem or mobile phone for certain tasks.
- Read operation.

  - A read operation is used to retrieve the current settings used by the GSM/GPRS modem or mobile phone for certain tasks.
- Execution operation.
  - An execution operation is used to perform an action or retrieve information/status about the GSM/GPRS modem or mobile phone.

3.2.2 Operating Mode: SMS Text Mode and SMS Protocol Description Unit (PDU) Mode

There are two ways of sending and receiving SMS messages it is either by text mode or by PDU mode.

3.2.2.1 Text Mode

Basically, text mode is just an encoding of bit stream represented by PDU mode. In SMS Text mode, a single short message can have text length of 160 characters which is 7 bit coded or 140 characters which is 8 bit coded. This application is bounded to the set of present coding options.

3.2.2.2 Protocol Description Unit (PDU) Mode

The PDU mode offers to send binary information either in 7 bit or 8 bit format. It can be used to send compressed data, binary data or own encoding of the characters in the binary bit stream. Any encoding can be implemented by using PDU format.

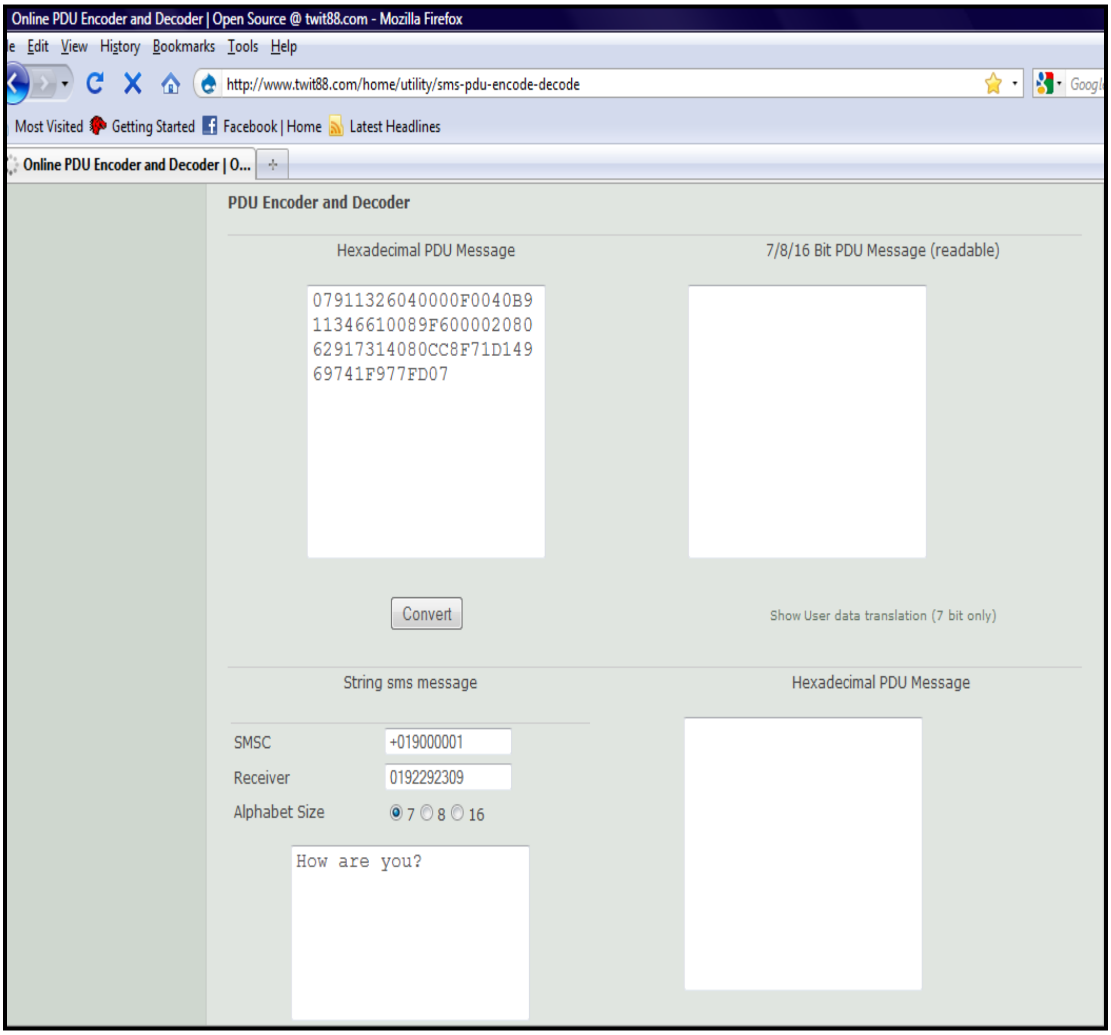
Given an example:
07911326040000F0040B911346610089F60000208062917314080CC8F71D149697
41F977FD07

The above PDU string contains the message "How are you?" . The string is built from hexadecimal-octets and semi decimal-octets. The following table describes and explains the code.

Octet(s)	Description	format	In this example
07	Length of the SMSC information	hex-octet	7 octets
91	Type of address of SMSC	hex-octet	internation format
13 26 04 00 00 F0	SMSC number	decimal semi-octets b	31624000000
04	First octet of this SMS-DELIVER message.	hex-octet	TP-MMS
0B	Length of the sender address	hex-octet	11 (decimal)
91	Type of address of the sender number	hex-octet	...
13 46 61 00 89 F6	Sender number	decimal semi-octets	31641600986
00	Protocol identifier	hex-octets	...
00	Data encoding scheme	hex-octets	...
20 80 62 91 73 14 08	Time stamp c	decimal semi-octets	06-08-02 29:17:31
0C	Length of User data (SMS message)	hex-octets	12 (decimal)
C8 F7 1D 14 96 97 41 F9 77 FD 07	User data	8-bit octets respresenting 7-bit data	How are you?

Table 3.3: PDU Format Information

3.2.3 Comparison of SMS Text Mode and SMS PDU Mode


Online PDU Encoder and Decoder | Open Source @ twit88.com - Mozilla Firefox
e Edit View History Bookmarks Tools Help
http://www.twit88.com/home/utility/sms-pdu-encode-decode
Googl
Most Visited
Getting Started
Facebook | Home
Latest Headlines
Online PDU Encoder and Decoder | O...
PDU Encoder and Decoder
Hexadecimal PDU Message
7/8/16 Bit PDU Message (readable)
Convert
Show User data translation (7 bit only)
String sms message
Hexadecimal PDU Message

```
07911326040000F0040B9
11346610089F600002080
62917314080CC8F71D149
69741F977FD07
```

SMSC: +019000001

Receiver: 0192292309

Alphabet Size: ● 7 ○ 8 ○ 16

How are you?

Figure 3.7 Online PDU Encoder and Decoder

3.2.3.1 Short Message Service Center (SMSC)

In SMS text mode, the only way to specify the service center address to be used by +CMGW and +CMGS is through +CSCA. However, in SMS PDU mode, it is possible to specify the service center address to the AT commands +CMGW and +CMGS directly as a parameter value.

3.2.3.2 Syntax of SMS AT Commands and Responses

The syntax of certain SMS AT commands and the return responses after command execution is different with GSM modem operates at different modes.

Given a situation where using SMS text mode to send SMS message "It is easy to send text messages." The command lines are as follow.

```
AT+CMGS="+85291234567"<CR>It is easy to send text messages.<Ctrl+z>
```

However, if in SMS PDU mode, executing the above command line will cause an error to occur. This is due to the syntax of the +CMGS AT command is different in SMS PDU mode. The command lines should be as below.

```
AT+CMGS=42<CR>07915892000000F001000B915892214365F7000021493A283
D0795C3F33C88FE06CDCB6E32885EC6D341EDF27C1E3E97E72E<Ctrl+z>
```

3.2.3.3 Defined Values for Certain Parameters

When the GSM/GPRS modem or mobile phone is operating in different modes, the defined values for certain parameters are also different. Usually string values are defined for text mode while numeric values are defined for PDU mode.

Message status	**Defined values in text mode**	**Defined values in PDU mode**
Received unread	"REC UNREAD"	0
Received read	"REC READ"	1
Stored unsent	"STO UNSENT"	2
Stored sent	"STO SENT"	3

Message status	Defined values in text mode	Defined values in PDU mode
All messages	"ALL"	4

Table 3.4: Particular Instructions which are defined in Text Mode and PDU Mode

3.2.3.4 Input/Output Format of SMS Messages Used by SMS AT Commands

The input/output format of SMS messages used by SMS AT commands is different when different mode is used. In SMS text mode, headers and body of SMS messages are inputted/outputted as separate parameters or fields. In SMS PDU mode, TPDUs (Transport Protocol Data Units) in hexadecimal format are inputted and outputted. Headers and body of SMS messages are encoded in the TPDUs.

For instant, when sending the SMS message "It is easy to send text messages." to the mobile phone number +85291234567 in SMS text mode,

```
AT+CMGS="+85291234567"<CR>It is easy to send text messages.<Ctrl+z>
```

However, when it is need to send the same SMS text message as SMS PDU mode,

```
AT+CMGS=42<CR>07915892000000F001000B915892214365F7000021493A283
D0795C3F33C88FE06CDCB6E32885EC6D341EDF27C1E3E97E72E<Ctrl+z>
```

3.2.3.5 Ease of Use

Based on previous examples, it is easier to use AT commands in SMS text mode where it is more direct. Moreover the structure of different types of TPDUs in the bit level and the encoding and decoding of the hexadecimal sequence is not required.

3.3 Alternative to Avoid Rewiring Other Than using Power Line Communication – (Ultrasound)

Other than using the existing power for home automation, communication in the air is also functional. For instant, ultrasound can be used. Ultrasound is the cyclic sound where its frequency is greater than sound frequency that can be heard by human being. Ultrasonic is widely used in the industry such as to penetrate a certain medium, to measure the reflection signature or even supply focused energy.

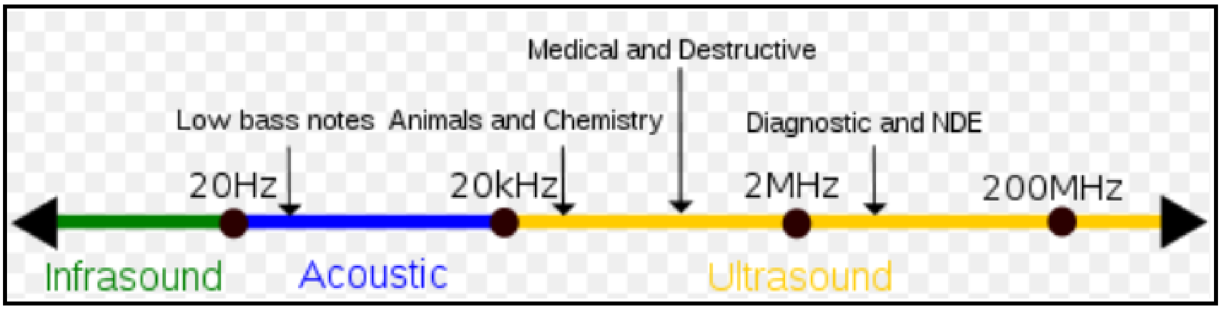


Figure 3.8 The Range of Sound

3.3.1 Ultrasonic Sensor

Ultrasonic sensor (transmitter and receiver) evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively. It can generate high frequency sound waves and as well to evaluate echo that is received by the sensor. Basically, it performs calculation of time interval between sending the signal and receiving the echo to determine the distance to an object.

3.3.2 Ultrasonic switch

An ultrasonic switch is implemented by using a transmitter and a receiver that communication to each other in order to perform a switching task.

3.3.2.1 Ultrasonic Transmitter

It is a need for create a circuit that is able to transfer signals through air. Hence, the circuit is designed in such a way that ultrasonic waves are transmitted via airs. By using a 555 IC which is configured as an astable multivibrator (oscillator), a continuous signal of a set frequency (frequency of 40 to 50kHz) will be generated as the reset pin (pin 4) is held high. The signal will be passed through a ultrasonic transmitter. IC555 Timer is an integrated circuit to provide pulse generation, oscillation and variety of time. It has three operating modes, which are Monostable, Astable and Bistable. In this project, astable will be discussed because it works as oscillator to provide pulse generation.

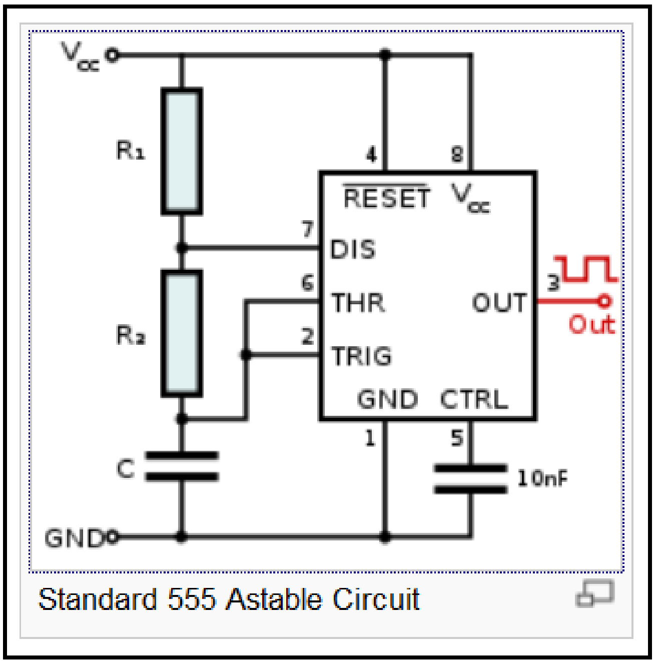


Figure 3.9 IC555 Timer Circuit for Astable Mode

In order to get the desired frequency from ic555 timer, the following equation can be applied.

$$f = \frac{1}{\ln(2) \times C \times (R1 + R2)}$$

3.3.2.2 Ultrasonic Receiver

An ultrasonic receiver will be used to sense an ultrasonic signal as the switch of transmitter is pressed. It will convert the ultrasonic waves into electrical variation of the same frequency. Outcome from the receiver will be fed to inverting pin of a operational amplifier. The outcome from the operational amplifier will be used to turn on a relay that can be used to control any house electrical appliances.

3.4 Extended Project (Home Security System – Burglary Notification System)

This idea came from the ultrasonic switch where ultrasonic sensors are used. The idea is to notify the user through SMS, whenever there is someone break-in (for instant window) where the sensor is placed. The system used two pairs of ultrasonic switches to implement the data sensing. One ultrasonic which is connected to a normally off relay is used to detect any trespasser. If an object goes through the first ultrasonic switch, the transmitter's signal will not be received by the receiver. Hence the relay that connected to the receiver will be triggered on. This will as well turn on the second ultrasonic switch. The second ultrasonic switch will feedback to the microcontroller. Once the microcontroller detects the desired voltage level, it will fulfils certain criteria and a notification message will be sent to user.

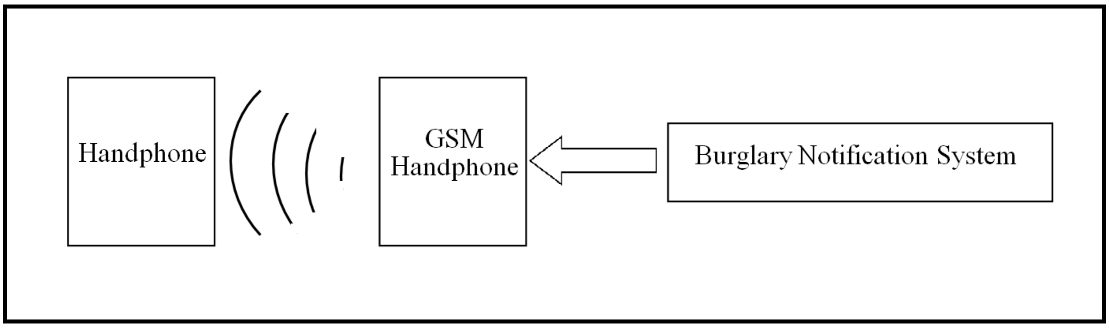


Figure 3.10 Rough Idea of Burglary Notification System

3.5 A Rough Work Distribution Graph

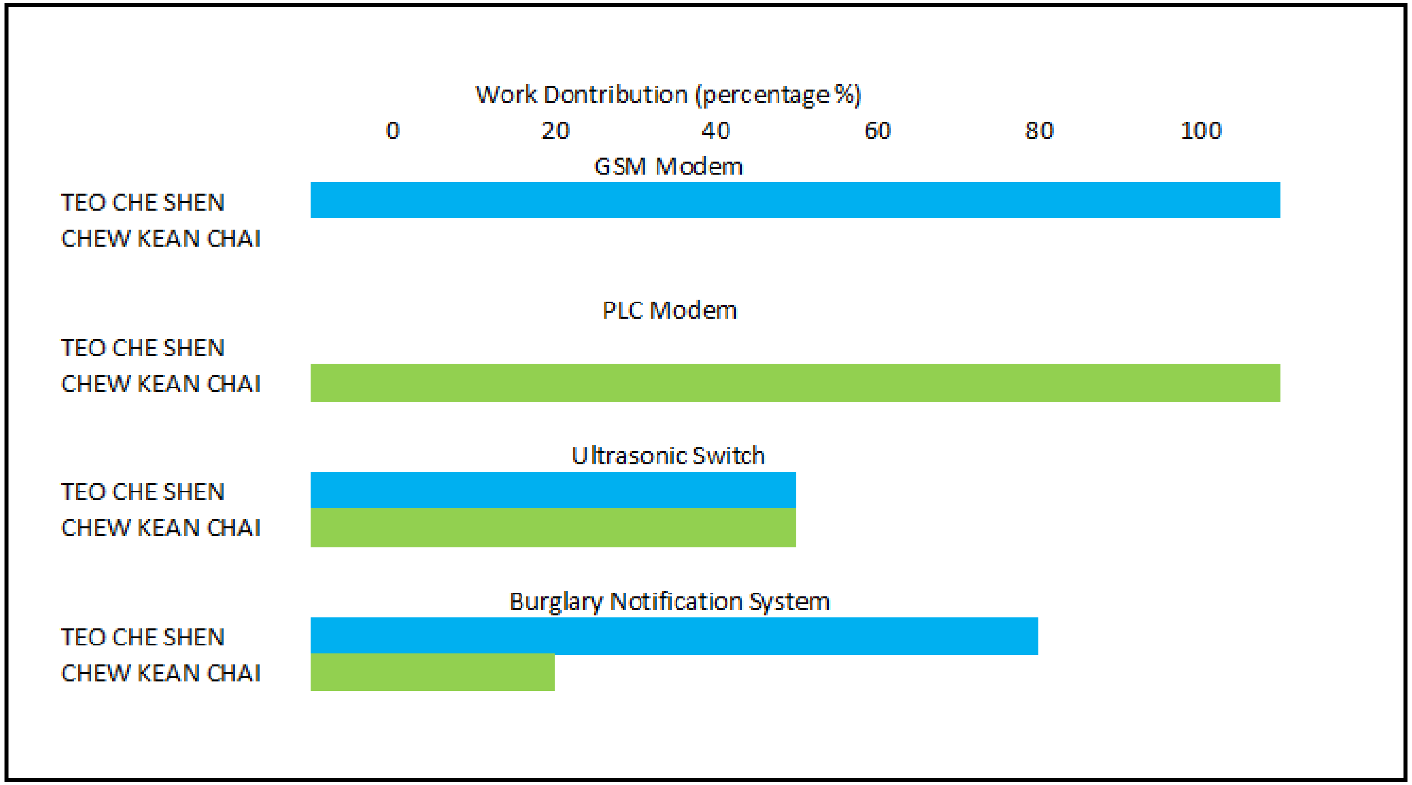
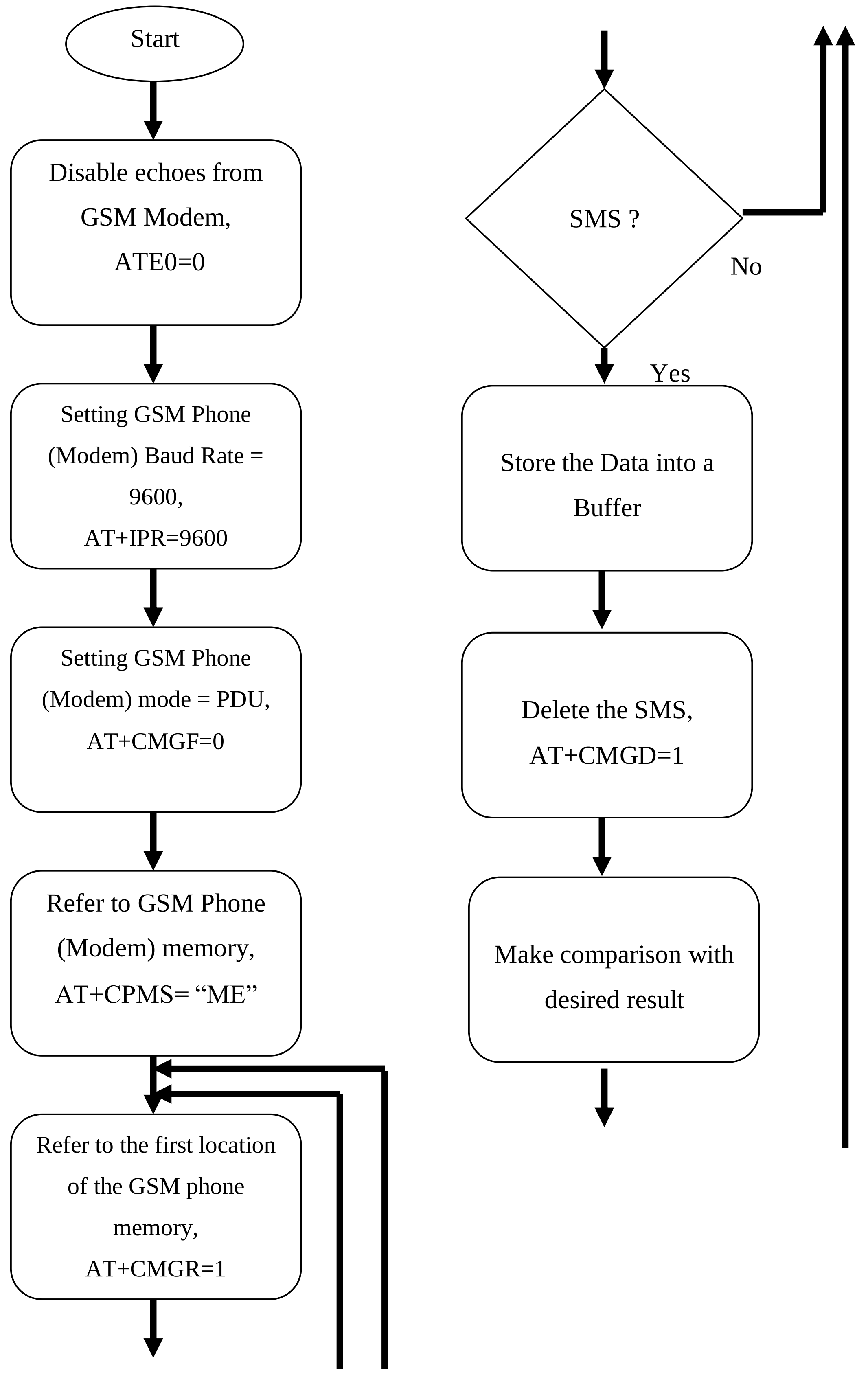
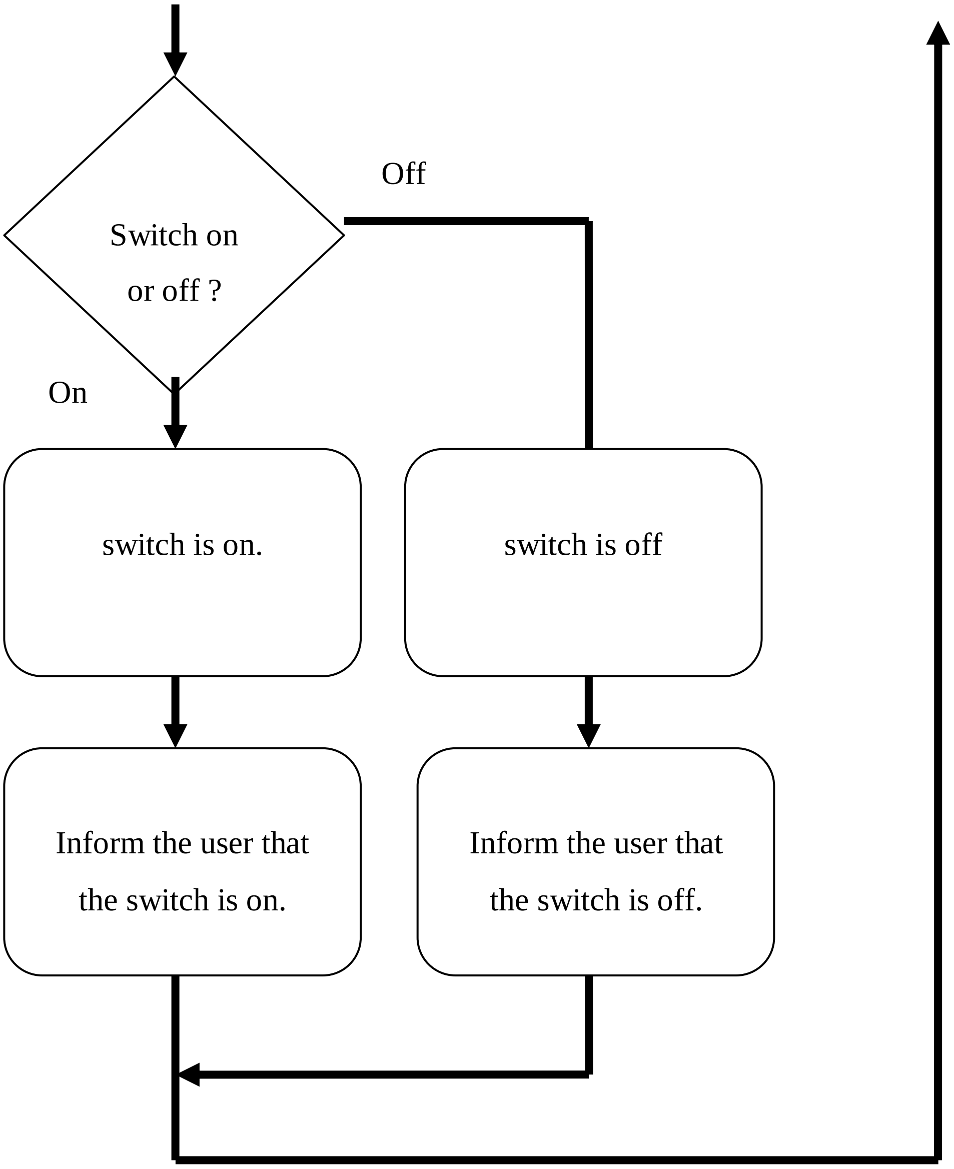


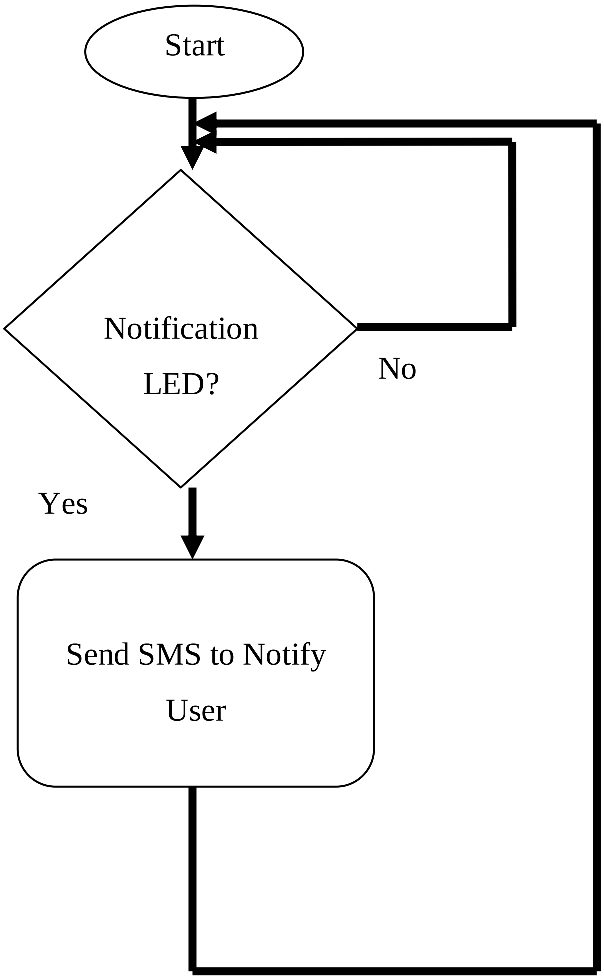
Figure 3.11 Work Distribution Graph

3.6 Flow Chart for GSM Modem to Switch On Ultrasonic Switch


Start
Disable echoes from GSM Modem, ATE0=0
Setting GSM Phone (Modem) Baud Rate = 9600, AT+IPR=9600
Setting GSM Phone (Modem) mode = PDU, AT+CMGF=0
Refer to GSM Phone (Modem) memory, AT+CPMS= "ME"
Refer to the first location of the GSM phone memory, AT+CMGR=1
SMS ?
No
Yes
Store the Data into a Buffer
Delete the SMS, AT+CMGD=1
Make comparison with desired result


Off
Switch on
or off ?
On
switch is on.
switch is off
Inform the user that
the switch is on.
Inform the user that
the switch is off.

3.7 Flow Chart for Burglary Notification System



CHAPTER 4

Discussion: Problems and Solutions; Result: Testing

4.1 LCD as Monitoring Tool

Initially, a combination of LED and a 330 Ohms resistor is used to check present of input signal. It is done by comparing the received data with a predefined value. Once the received signal tally with the predefined value, a port is set to be on and the LED will be lighted up. However, this idea can only be applied if the correct comparison is made. By using this way, the wrong incoming data cannot be identified. Hence, a LCD display is needed. The LCD will display all the incoming data which can be used for correction and investigation of the problem root.

4.1.1 Sample Code to Turning On a LED

Whenever there is a “sw on” message is receive, a LED a port RD2 will be turn on and the incoming data from GSM phone will be displayed on LCD as well.

```
if(received_data1[70]=='F' && received_data1[71]=='3'
&& received_data1[72]=='3' && received_data1[73]=='B'
&& received_data1[74]=='E'&& received_data1[75]=='8'
&& received_data1[76]=='E' && received_data1[77]=='D'
&& received_data1[78]=='0' && received_data1[79]=='6')
```

```
{
        if(LED==1)
                {
                        send_message4();
                }
        else
                {
                        LED=1;
                        __delay_ms(500);
                        send_message4();
                        CHECK=1;
                }
}
```

Whenever there is a “sw off” message is receive, a LED a port RD2 will be turn off and the incoming data from GSM phone will be displayed on LCD as well.

```
if(received_data1[70]=='F' && received_data1[71]=='3'
&&received_data1[72]=='3' &&received_data1[73]=='B'
&& received_data1[74]=='E'  &&  received_data1[75]=='8'
&& received_data1[76]=='6'  &&  received_data1[77]=='D'
&& received_data1[78]=='3' &&  received_data1[79]=='6' &&
received_data1[80]=='0' && received_data1[81]=='3')
{
        if(LED==0)
                {
                        send_message5();
                }

        else
                {
                        LED=0;
                        __delay_ms(500);
                        send_message5();
                        CHECK=0;


                }
        }
```

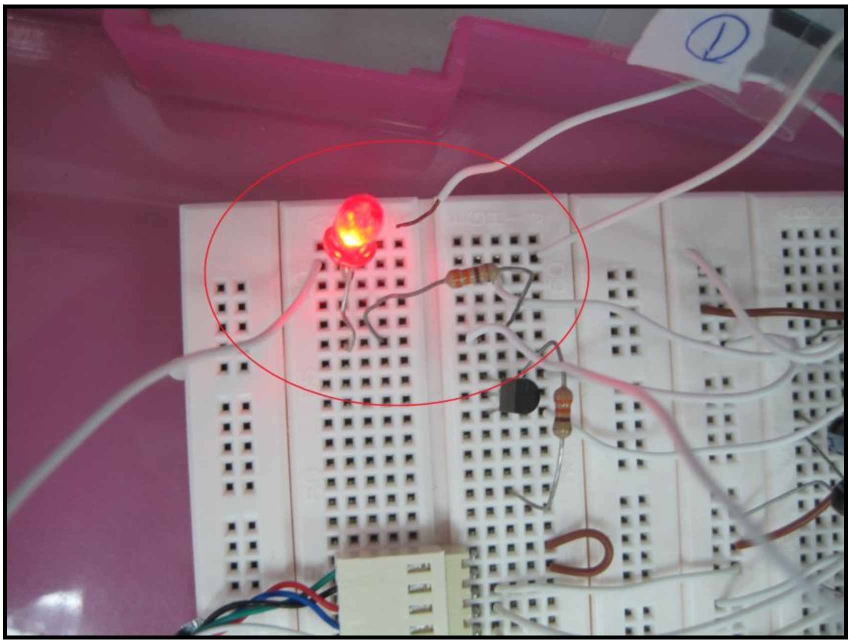


Figure 4.1 LED is Lighted Up to Show the Correct Input is Detected

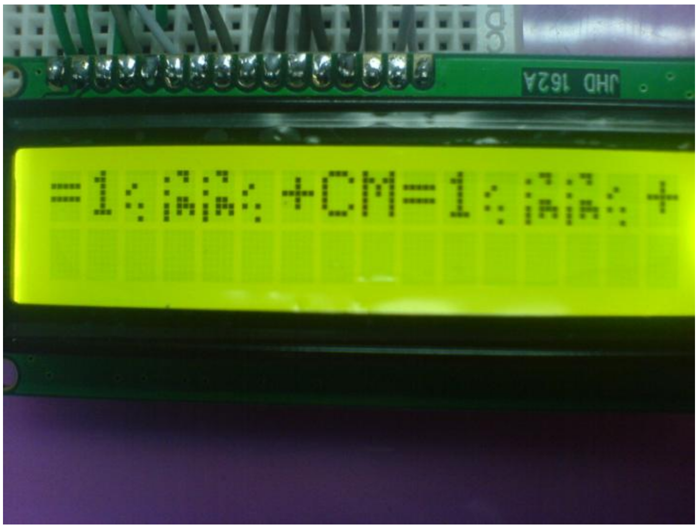


Figure 4.2 LCD Displays Received Data

LCD can only display ascii codes, whereby unidentified data will be replaced by other symbol.

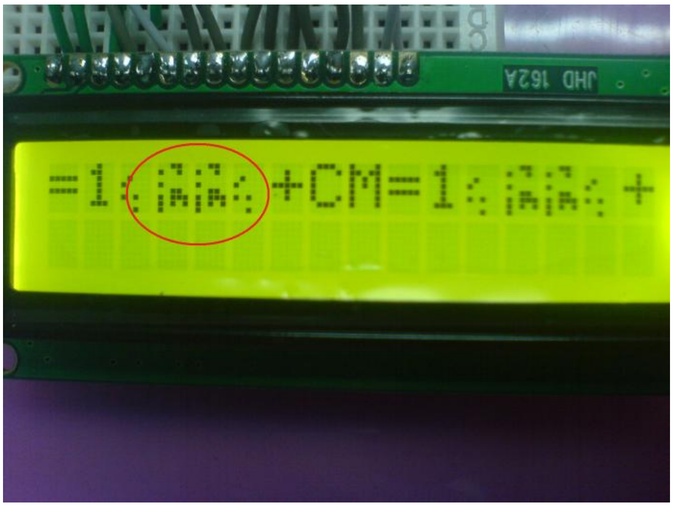


Figure 4.3 LCD Displays Symbol for Enter and New Row.

Basically, LCD just serve as a monitoring circuit, hence, it does have main role in the overall circuit.

4.2 Communication between PIC16F877A and Hyperterminal

In order to translate TTL signal to signal that PC recognize, one max232 IC is used (this is due to GSM phone and PC have different voltage levels). Apart from that, a serial to USB converter is used for proper communication.

Figure 4.4 Serial to USB converter

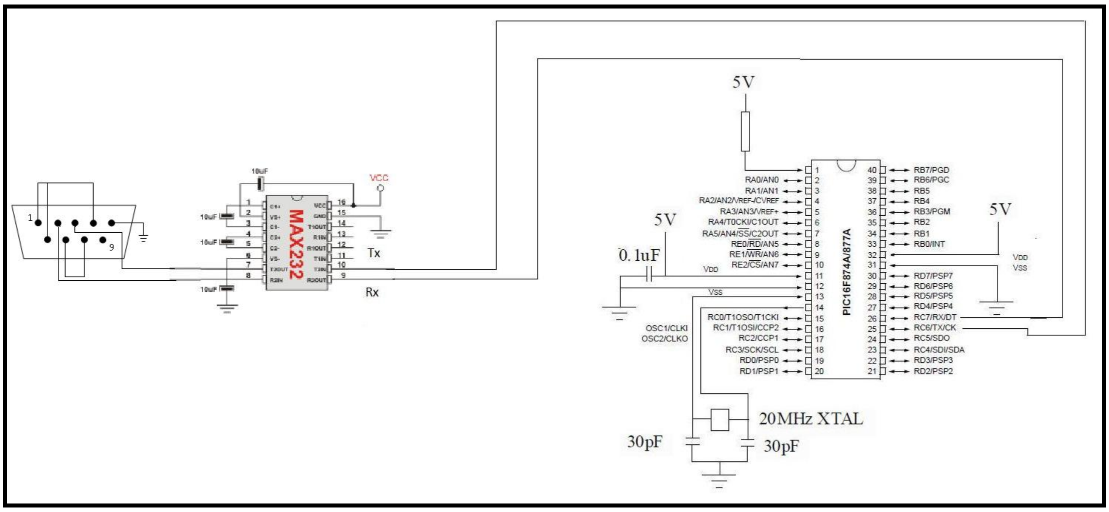


Figure 4.5 Circuit Connection for PIC16F877A to Serial to USB Converter

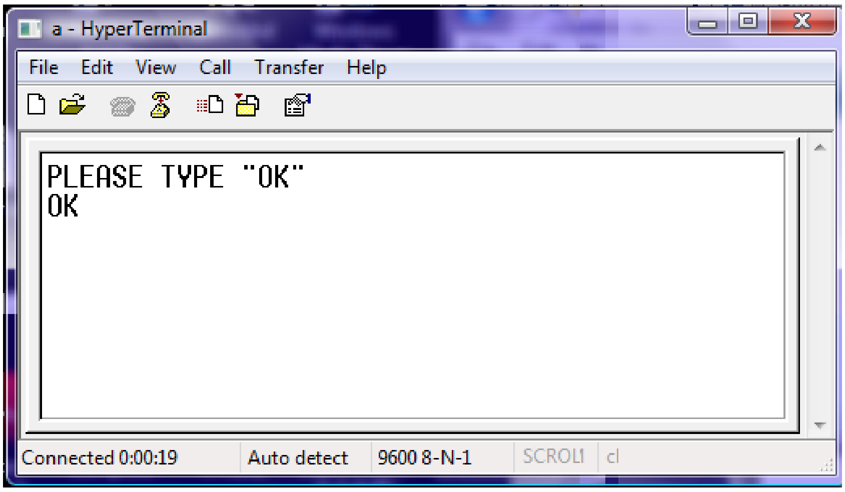
4.2.1 Code Testing for Checking PIC16F877A and Hyperterminal Communication

In order to check for the communication, a sample code is written whereby a string of character is send by the PIC16F877A to the hyperterminal (the data is showed on the hypertrminal interface) and any feedback (typed in hypeterminal using keyboard) will be displayed on the LCD.

Sample code:

```
void main()
{
uart_send(‘P’);
uart_send(‘L’);
uart_send(‘E’);
uart_send(‘A’);
uart_send(‘S’);
uart_send(‘E’);
uart_send(‘ ’);
uart_send(‘T’);
uart_send(‘Y’);
uart_send(‘P’);
uart_send(‘E’);
uart_send(‘ ’);
uart_send(‘“’);
uart_send(‘O’);
uart_send(‘K’);
uart_send(‘”’);//Display Command on Hyperterminal

send_char(data); //Display command on LCD



}
void uart_send(unsigned char data)
{
        while(TXIF==0);                         //only send the new data after
        TXREG=data;                             //the previous data finish sent
}
```

```
void uart_receive(unsigned char data)
{
        while(RXIF==0);                         //only send the new data after
        RXREG=data;                             //the previous data finish sent
}
```



```
PLEASE TYPE "OK"
OK
```

Figure 4.6 Hyperterminal Displays Command from Microcontroller

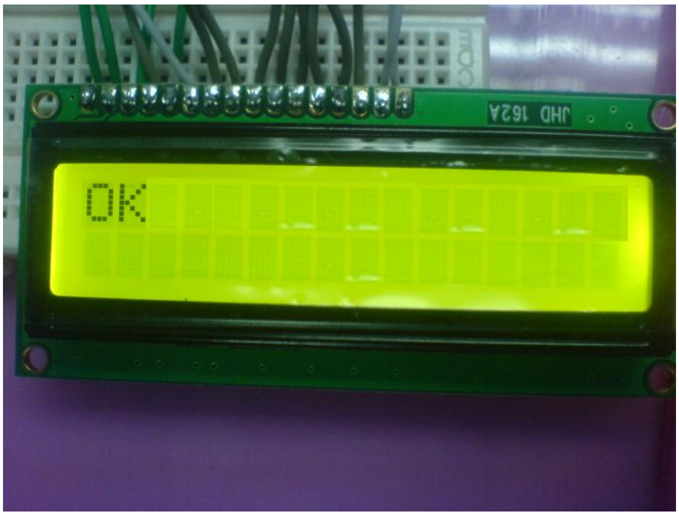
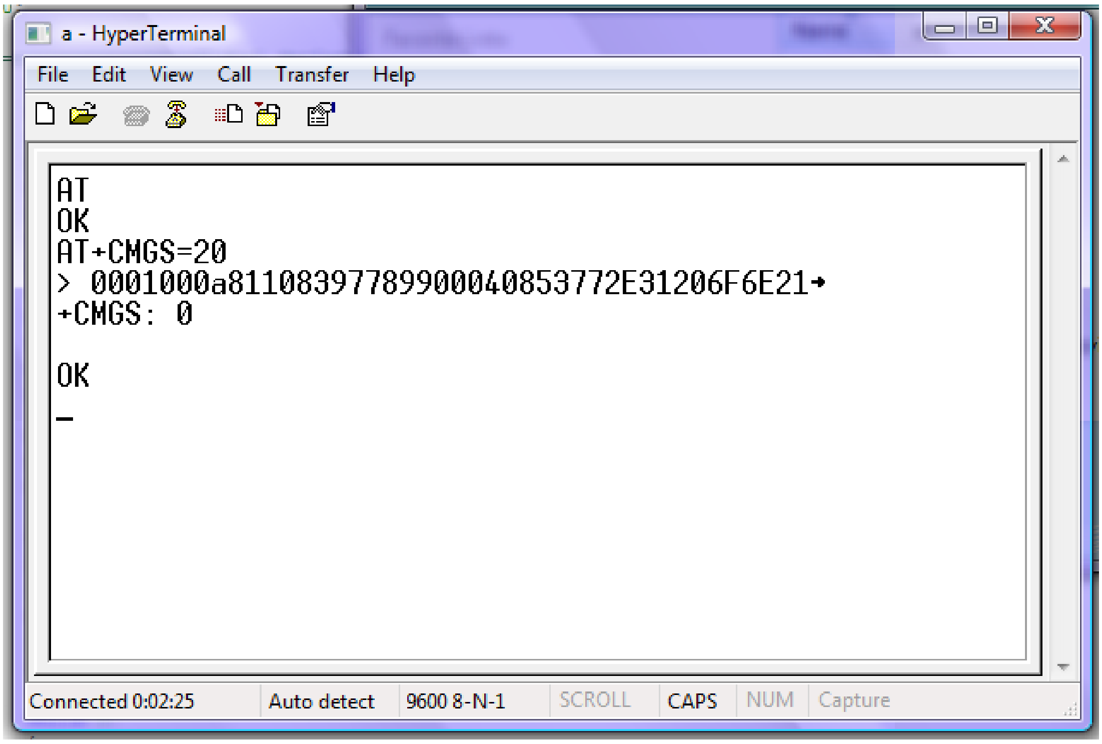


Figure 4.7 LCD Displays “key-in” Data from Hyperteminal

If the entered data is showed on the LCD, this shows that the hyperterminal and microcontroller are well – communicated.

4.3 Communication between SE K700i (GSM Modem) and Hyperterminal

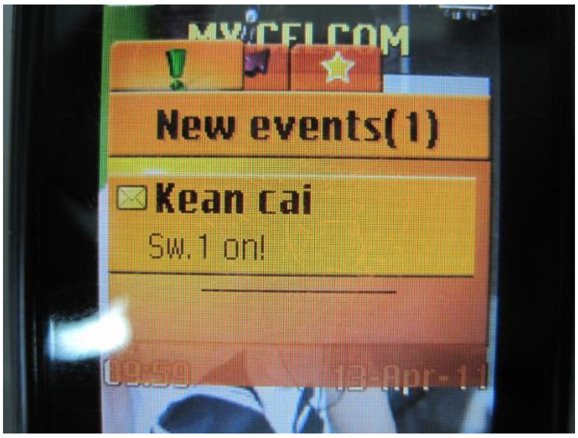
SE K700i is connected to laptop by using a serial to USB cable. This test is to check the communication between hyperterminal and the GSM phone. By typing certain AT-command in hyperterminal, the GSM phone will respond by providing some reply.



```
AT
OK
AT+CMGS=20
> 0001000a81108397789900040853772E31206F6E21→
+CMGS: 0

OK
_
```

Figure 4.8 Display on Hyperterminal (to Send a Message to User)

A set of AT-command is typed in order to instruct the GSM phone to send certain message to other user.



MY CELCOM

New events(1)

Kean cai

Sw.1 on!

09:59 13-Apr-11

Figure 4.9 Display on User’s Hand Phone

4.4 Communication between SE K700i (GSM Modem) and PIC16F877A

Since serial to USB cable is the only choice available in the market, USB hub is removed and the following circuit is connected. Two max232 ICs are used to trigger the output signal from the GSM phone to 5V.

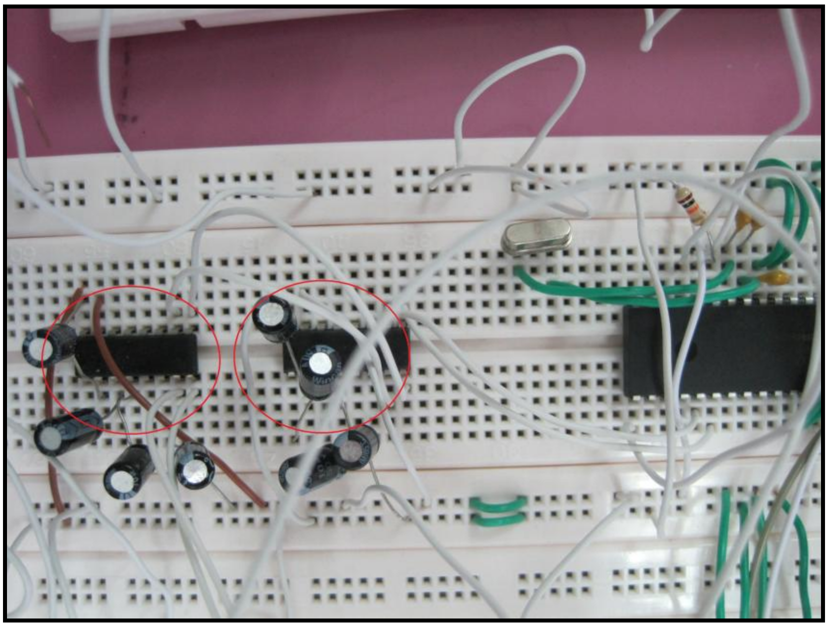


Figure 4.10 Circuit for Two MAX232 ICs

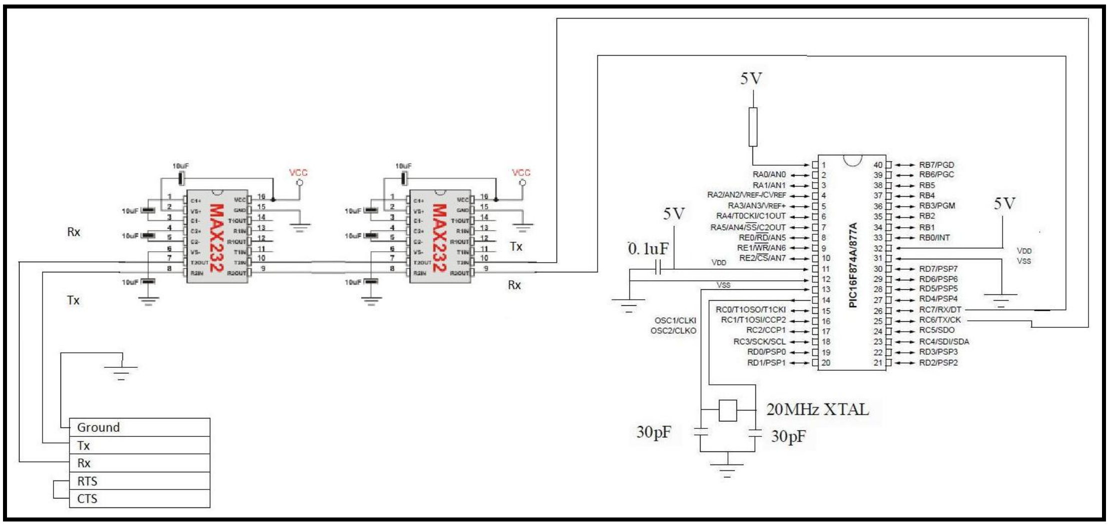


Figure 4.11 Schematic Circuit for PIC16F877A to GSM Phone Connection

4.5 Receive Error from GSM Phone

Basically, the transmit and the receive pin-outs of the GSM phone can be connected to the RX (receive) and TX (transmit) of microcontroller pic16F877A directly. However, the LCD displays black squares or "bug" characters whenever there is an input. This means there are some errors.

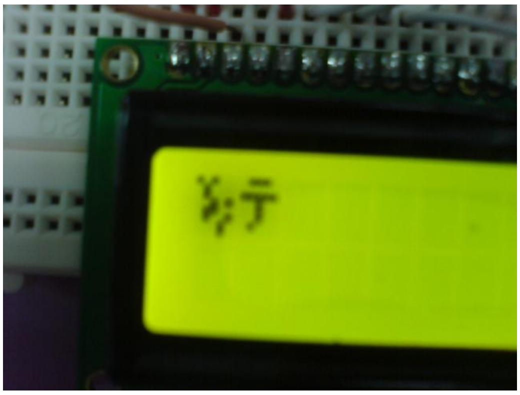


Figure 4.12 LCD Displays Some Bug Characters

4.5.1 Investigation and Solution on Receiving an Error

In order to investigate the causes of the problem, digital oscilloscope is used to check for the incoming data from the GSM phone (Probe is more preferable than usual cable for more accurate outcome because probe provides lowest noise).

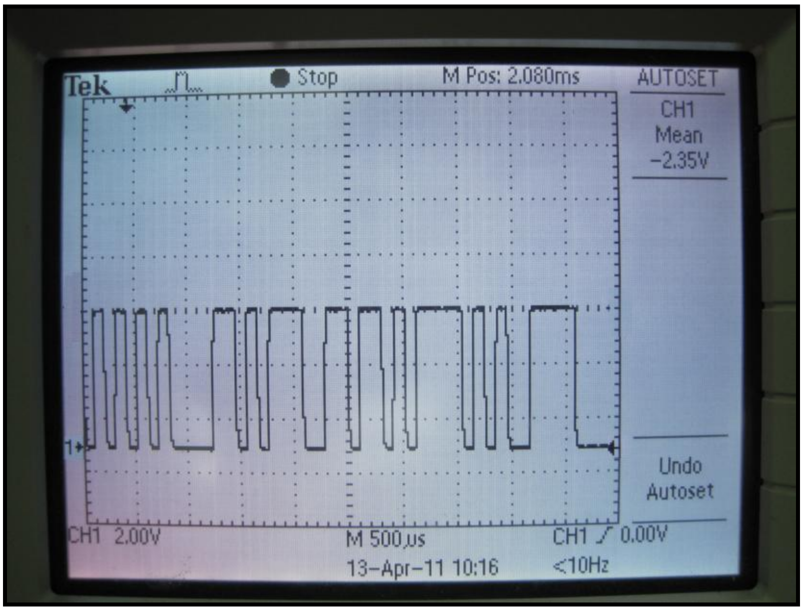


Figure 4.13 Transmitted Data from Microcontroller with Signal of 5V Peak to Peak

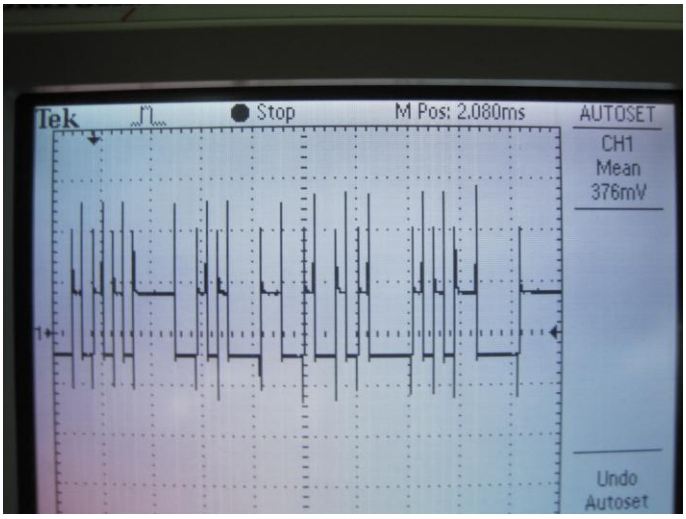


Figure 4.14 Transmitted Data from GSM Phone with 3V Peak to Peak Signals (with Noises)

Based on the oscilloscope, the incoming signal from the GSM phone has amplitude of 3V peak to peak rather than voltage input of 5V peak to peak. This is the major cause for receiving an error input because of the failure of the microcontroller to detect the weaker signals. Hence, two 5V powered-up max232 ICs are used to trigger the signals to 5V which is the appropriate input signal for pic16F877A.

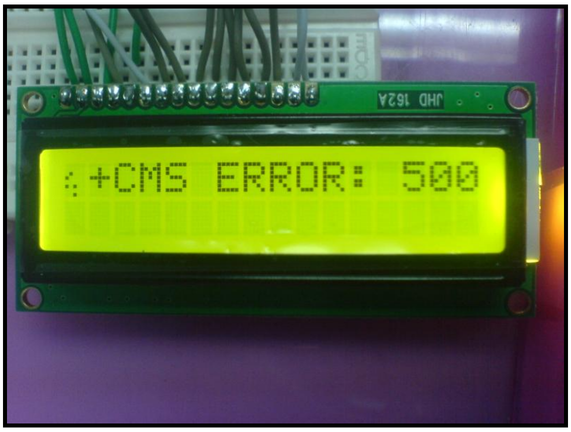


Figure 4.15 LCD Displays Correct Input Data when Data from GSM Phone is Passed to Two MAX232 ICs

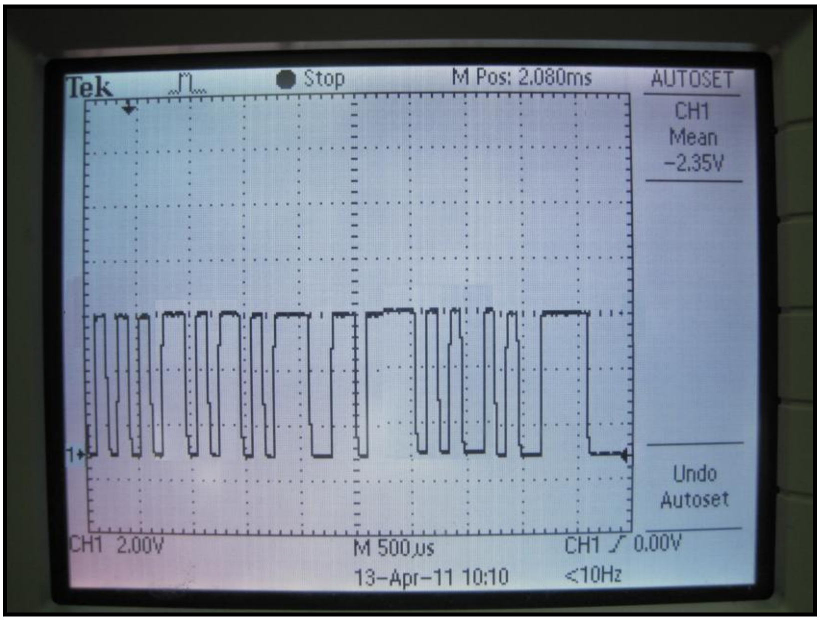


Figure 4.16 Input Signals for Microcontroller after being Pass through Two MAX232 ICs

4.6 Coding Errors

4.6.1 Problems Caused by Polling in Coding Structure

The mean of polling is by calling the transmit function and the receive function in algorithm.

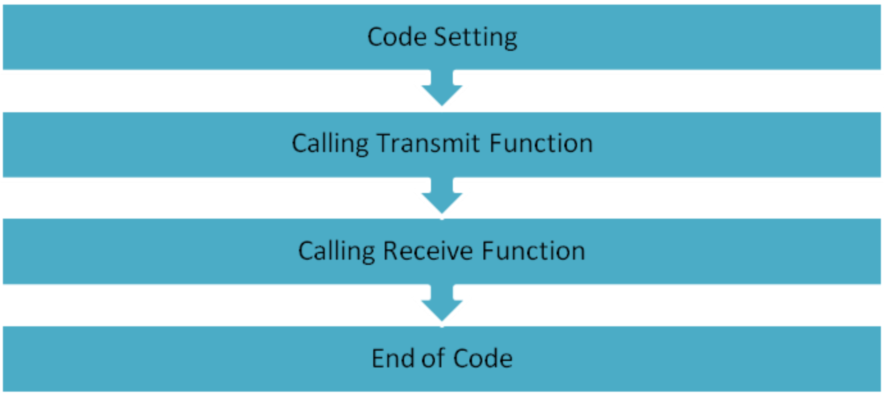


Figure 4.17 The Algorithm of Polling

4.6.1.1 Sample Code for Using Polling Structure

```
void main()
{
uart_send();
uart_receive();
}
void uart_send(unsigned char data)
{
        while(TXIF==0);                                //only send the new data after
        TXREG=data;                                    //the previous data finish sent
}

void uart_receive(unsigned char data)
```

```
{
        while(RXIF==0);                          //only send the new data after
        RXREG=data;                              //the previous data finish sent
}
```

4.6.1.2 Explanation for Sample Code

When polling method is used, some of the received data will be corrupted and might not able to be delivered to PIC16F877A fully. This is because the speed of data transfer is too fast where receiving function is too late to be called out and causing some of the data is not able to be stored in the buffer on time.

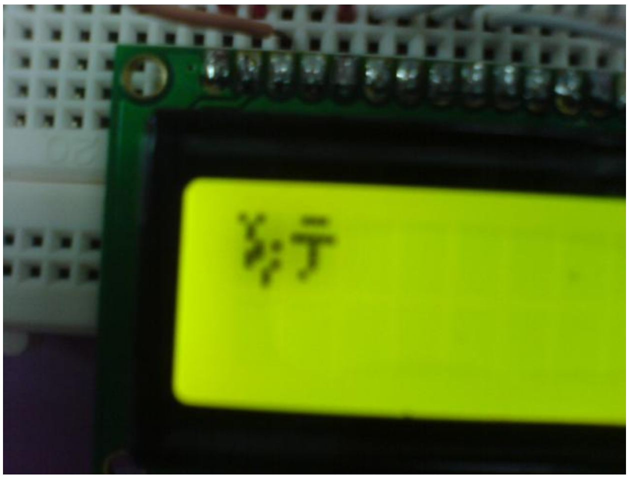


Figure 4.18 Screen Shoot from LCD with Receiving an Error

4.6.2 Problem Caused by Interrupts Sub-Routine (ISR) Function

Interrupt functions are used whenever an interrupt flag is enabled, the main code will be paused and jump to the interrupt function. This is because Interrupt function has higher priority. After the task in interrupt function is completed, the main coding will continue to execute.

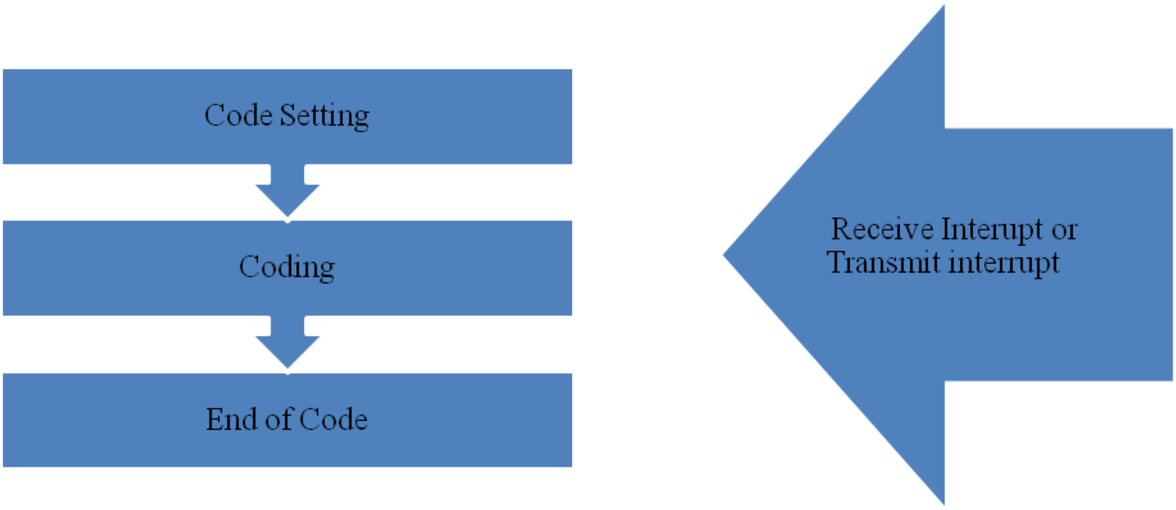


Figure 4.19 The Structure of Using ISR Functions

4.6.2.1 Sample Code for Using ISR Functions

```
void main()
{}
void uart_send(unsigned char data)
{
        while(TXIF==0);                          //only send the new data after
        TXREG=data;                              //the previous data finish sent
}

void interrupt ISR (void)
{
unsigned char data1;
if(TXIF==1)
{
send_uart();
}
if(RCIF==1)
{
```

```
data1=RCREG;
}
}
```

4.6.2.2 Explanation for Sample Code

However, if the transmit function is used as transmit interrupt, it will cause error too whereby the microcontroller will be receiving black square (as shown in LCD). This can be proven based on experiences and countless testing.

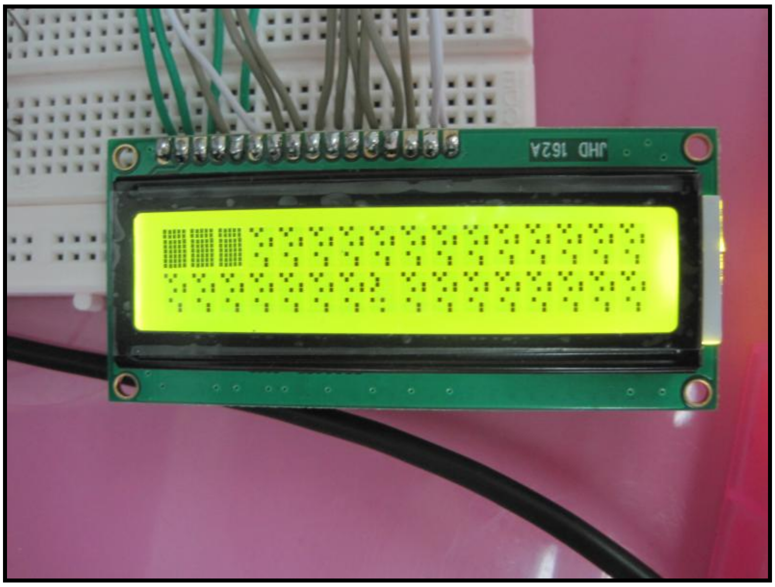


Figure 4.20 LCD Displays Black Square Whenever Transmit Interrupt is Used

4.6.3 The Problem is Solved by Fuse-in Polling Structure and Interrupt Function

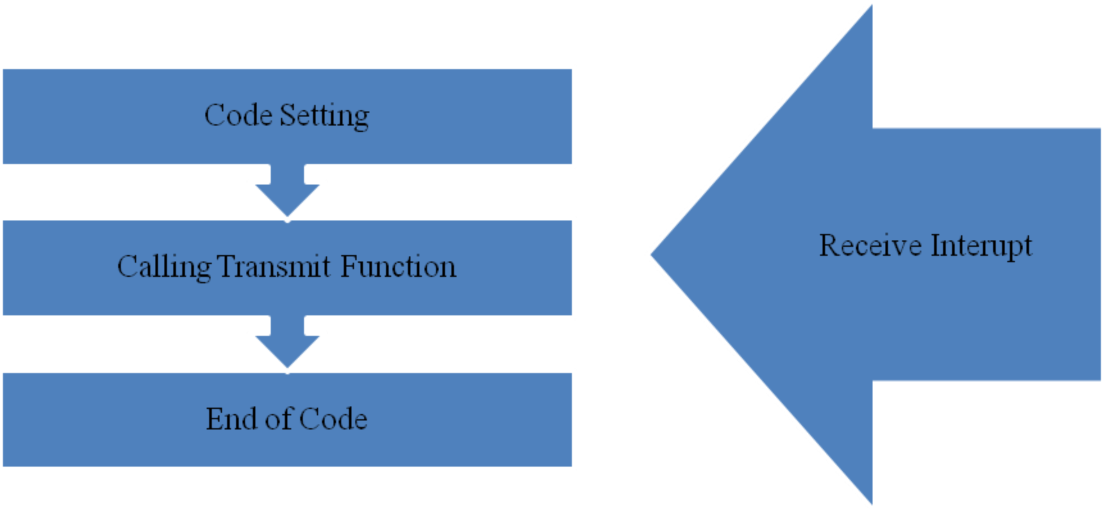


Figure 4.21 Code Structure of Calling Transmit Function and Using Receive Interrupt Function

4.6.3.1 Sample Code of Calling Transmit Function and Using Receive Interrupt Function

```
void main()
{
send_uart();
}
void interrupt ISR (void)
{
unsigned char data1;
if(RCIF==1)
{
data1=RCREG;
}
}
```

4.6.3.2 Explanation for the Sample Code

By this idea, receive interrupt can be activate throughout the coding as needed. Once input is detected, it will be stored into a buffer without the need of waiting the transmit function to complete. Hence, this guarantee no data loss at the highest possible.

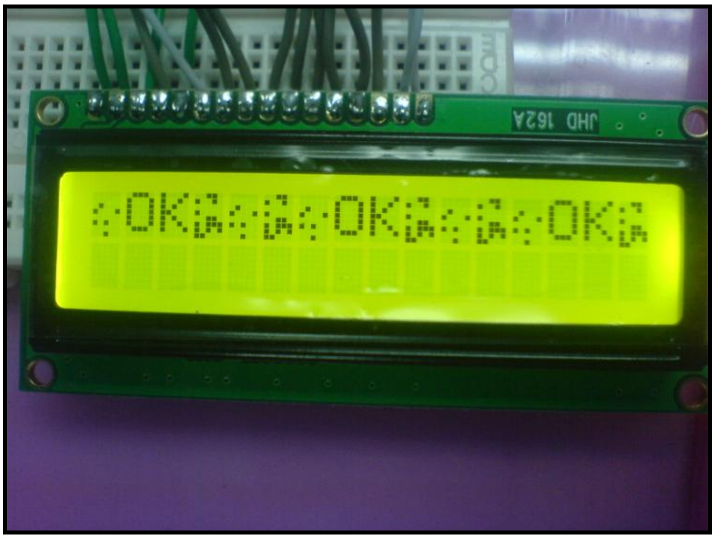


Figure 4.22 Screen Shot from LCD which Shows No Problem from the Coding

4.6.4 Special Remark for Using Receive Interrupt Function

Apart from that, the ISR function must be totally free from any instruction other than saving the received data to a buffer. Otherwise, it may cause partial of the data will be loss.

Wrong coding:

```
void interrupt ISR (void)
{
unsigned char data1;

if(RCIF==1)
{
data1=RCREG;
send_char(data1);
__delay_ms(2000);
}
}
```

Correct coding:

```
void interrupt ISR (void)
{
unsigned char data1;

if(RCIF==1)
{
data1=RCREG;
}
}
```

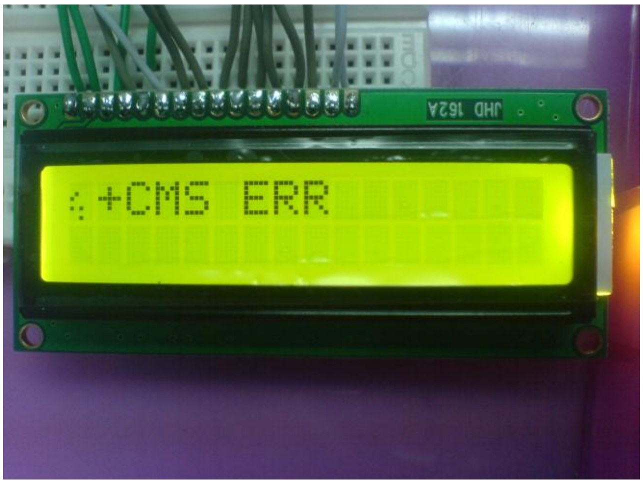


Figure 4.23 From LCD (Only Some of the Data is Retrieved)

4.6.5 The Limited Size of Array

The size of the array only up to 95 characters (surpassingly should be up to 125 characters but due to memory allocation of the microcontroller, there is only 95 character space left). The contain of the array will be used to do comparison and to perform some switching tasks.

4.6.5.1 Sample Code to Clear the Array

Hence the array must be cleared from time to time.

```
for(i=0; i<=94; i++)
        {
                received_data1[i]=0;
        }
```

4.6.5.2 Echoes from GSM Phone

Moreover, the echoes from the GSM phone must be turn off. Echoes mean whatever sent At-command will be reflected back to the input before replies for the command is received. The algorithm for a GSM phone to react with AT-Commands:

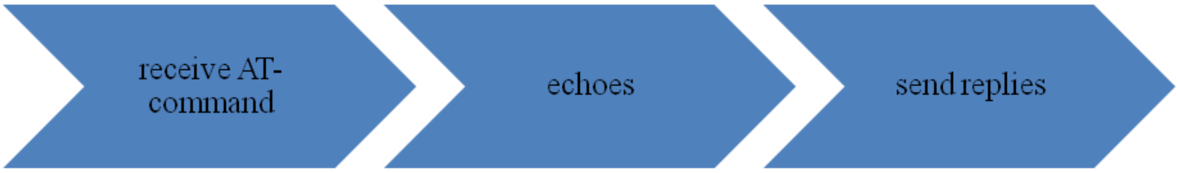


Figure 4.24 The Algorithm of Getting a Reply From GSM Phone

Echoes can be turned off by send AT-command, ATE0 to the GSM via microcontroller. The purpose of turning off the echoes is to save places (arrays) for the replies (data).

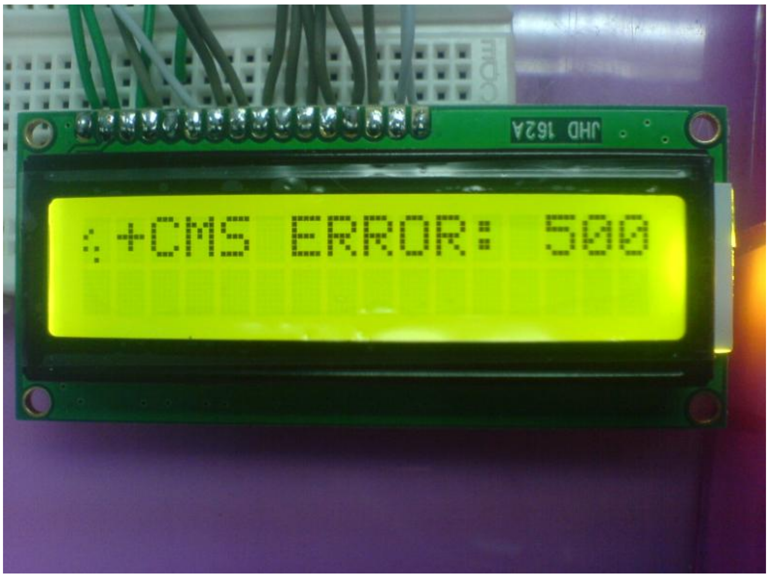


Figure 4.25 Before Echoes is Turned Off

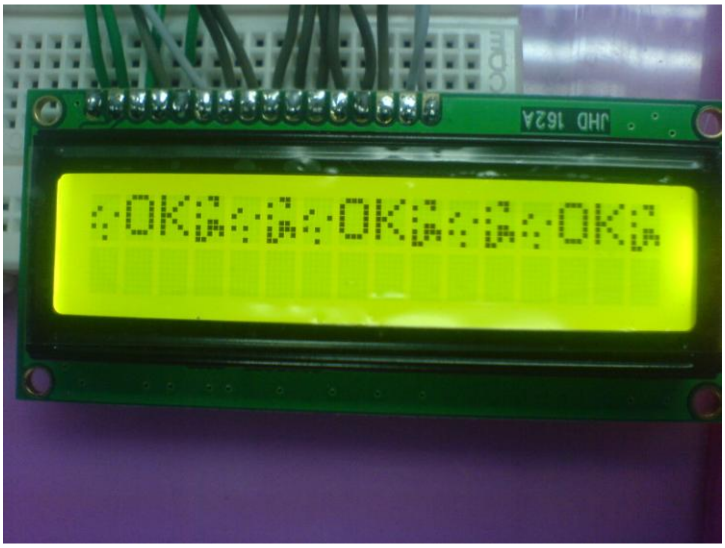


Figure 4.26 After Echo is Turned Off

4.7 The Analysis of PDU Mode for Sending SMS

Example:
00|01|00|0a|81|1083977899|00|04|12|537769746368204E6F2E31206973206F6E21

1083977899 = the user’s (receiver) phone number
12=the hex number of the total characters for the message
37769746368204E6F2E31206973206F6E21= the contain of the message

The arrangement of receiver number is arranged in a way that two numbers are switched among themselves.

10 83 97 78 99 = 01 38 79 87 99

The massage: Switch No.1 is on!

The total characters for the message which include the space is 18 and the hexadecimal number for 18_{10} is 12_{16}. Every character in ascii is represented 2 hexadecimal number.

4.8 Ultrasonic Transmitter

The signal produced by 555 IC is too small, it is required to be amplified by using a pair of PNP and NPN transistors.

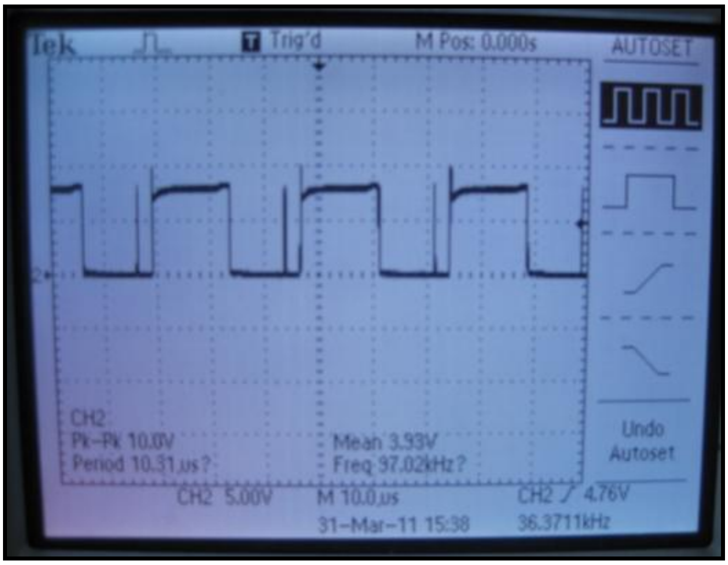


Figure 4.27 Output from IC 555 Timer

The transistors will drive the ultrasonic transmitter which will vibrate at 40kHz that generate ultrasonic waves.

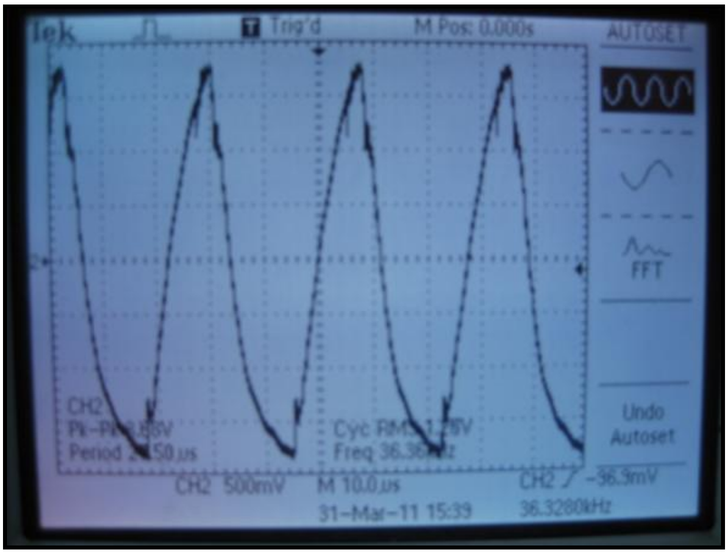


Figure 4.28 Output from Transistors is Fed to the Ultrasonic Transducer

4.8.1 Checking the Transmitted Signal Using Spectrum Analyzer

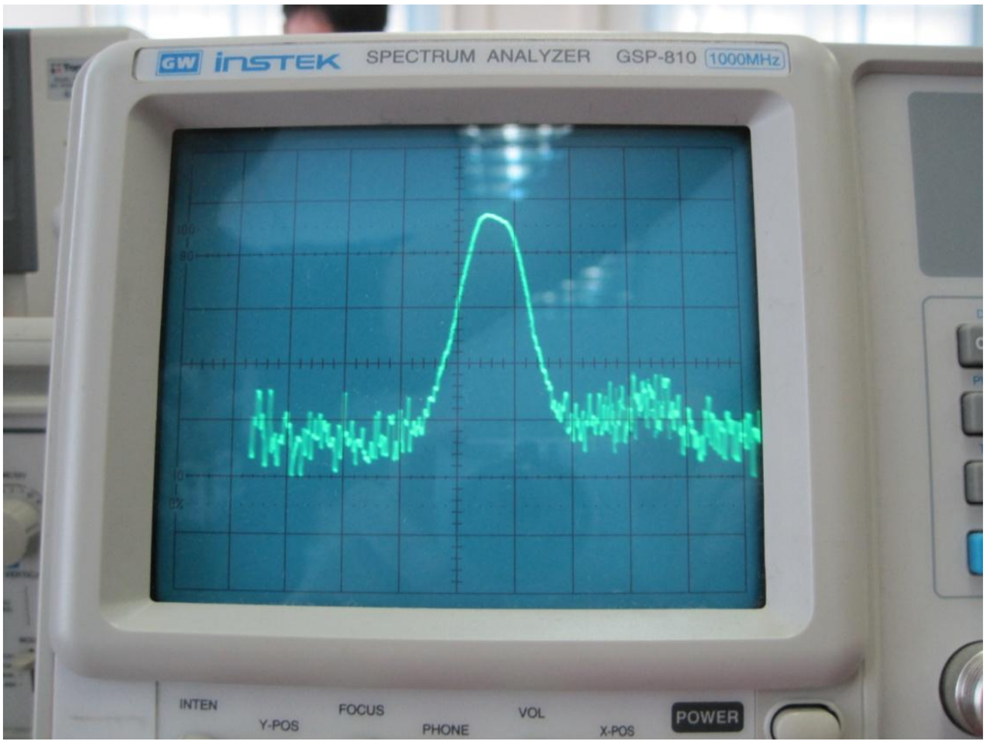


Figure 4.29 The Transmitted Signal is Captured by the Spectrum Analyzer

The signal can be adjusted by adjusting the variable resistor on the transmitter circuit in order to obtain required transmitted signal frequency.

4.9 Ultrasonic Receiver

Since the signal from the receiver is very small, it will be amplified by using a two stages amplifier. However, in order to feed in DC voltage in to the operational amplifier (which work as comparator), amplified signal need to be rectified and filtered. The filtered DC voltage is given to inverting pin of the operational amplifier

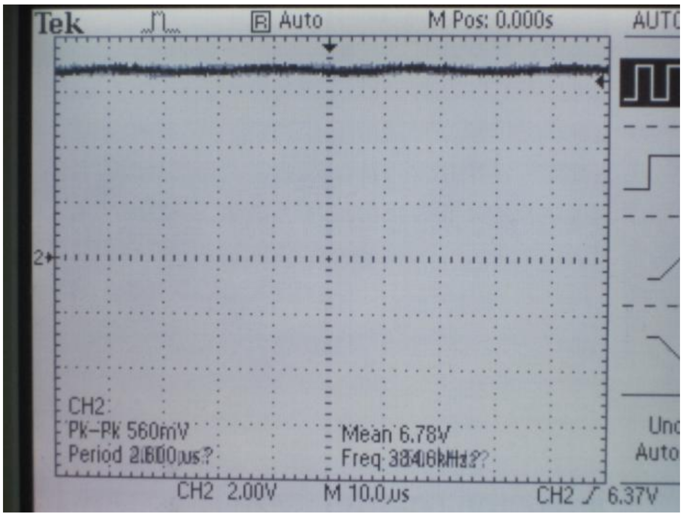


Figure 4.30 Output from Transistors before Relay is On

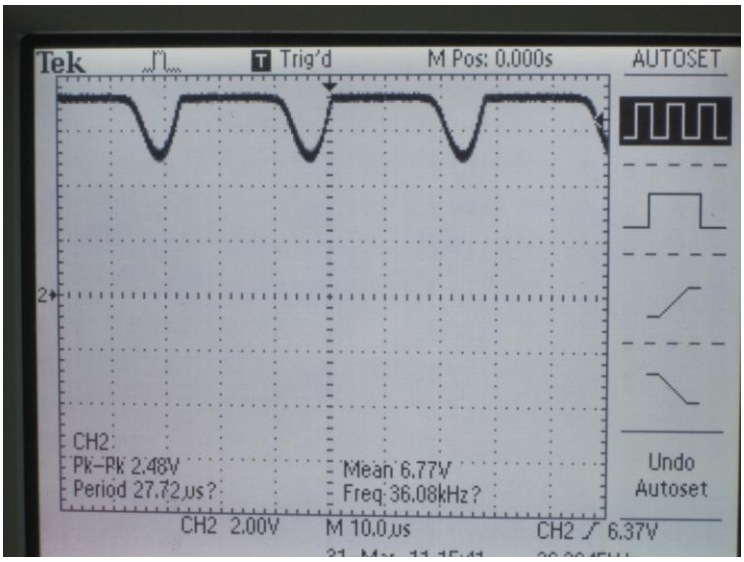


Figure 4.31 Output from Transistor after Relay is On

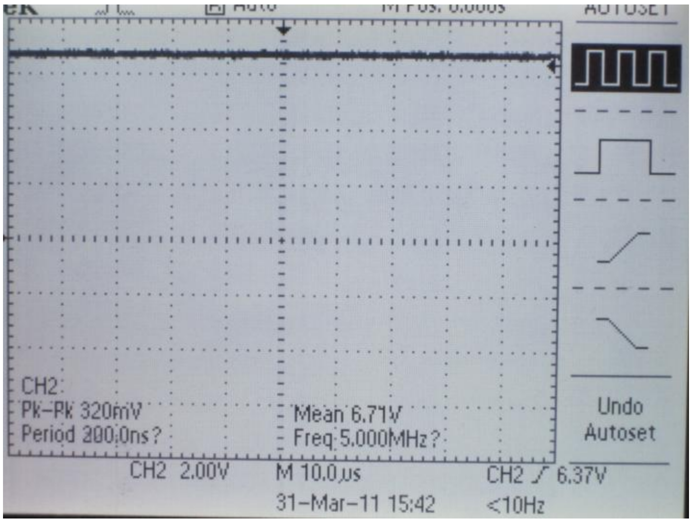


Figure 4.32 Output from Rectifier before the Relay is On

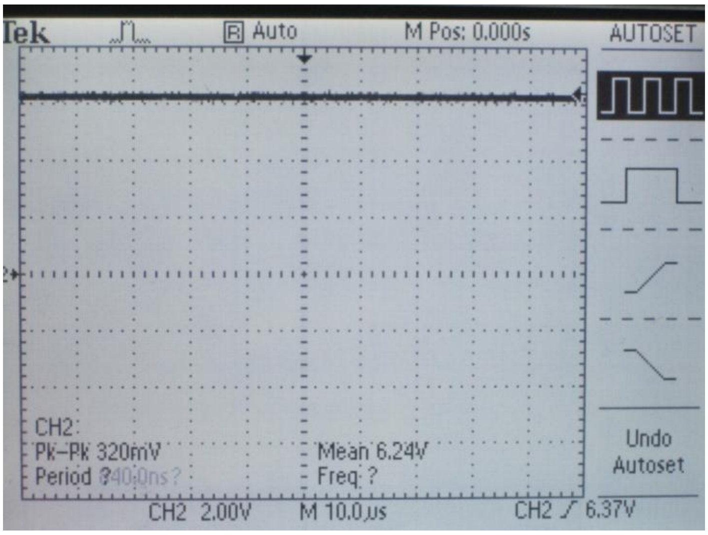


Figure 4.33 Output from Rectifier after the Relay is On

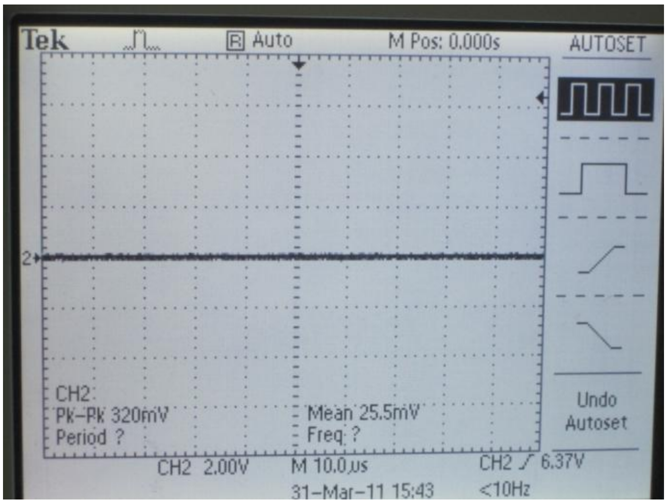


Figure 4.34 Output from Comparator before the Relay is On

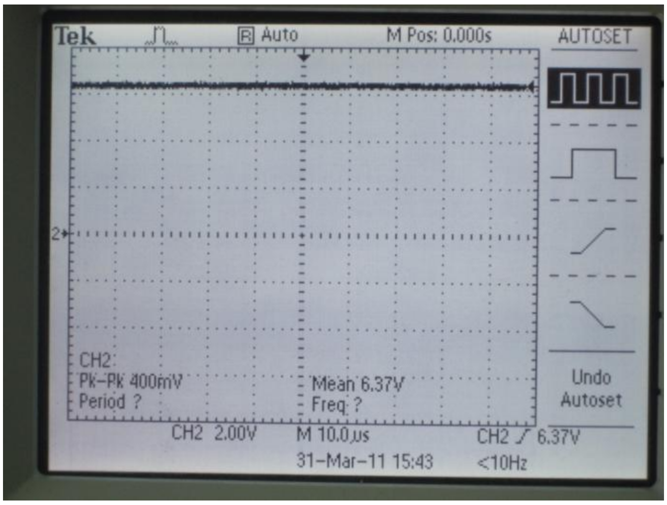


Figure 4.35 Output from the Comparator after the Relay is On

4.10 Table of Signal Sensitivity for the Ultrasonic Switch

The sensitivity of the ultrasonic switch varies with the distance between the transmitter and receiver.

Distance (CM)	Signal Sensitivity
0 to 10	Very strong
10 to 20	Strong
30 to 75	Moderate
75 to 100	Weak
>100	Very weak

Table 4.1 Sensitivity of the Ultrasonic Signal Versus Distance

4.11 Encountered Problems for the Ultrasonic Switch

Ultrasonic sounds are highly directional. Hence, the ultrasonic transmitter should be placed towards ultrasonic receiver for proper functioning during operational. For in-house installation, the transmitter and receiver should be placed “face-to-face” for higher transmission efficiency.

Apart from that, Ultrasonic waves are emitted by many natural sources. Therefore, the circuit might get falsely triggered, especially when a flip-flop is used with the circuit. Hence, it should be installed where the possibility of ultrasonic source exposure is minimized. Apart from that, ultrasonic signal can be reflected from hard surface.

4.12 Home Security System (Burglary Notification System)

The idea comes from ultrasonic switch. Whenever an intruder passes through the sensors (transmitter and receiver), the transmitted signal will be reflected. This will causes the receiver receiving no signal. Hence, it will turn on the relay. However, usually the trespasser took little time to pass through the signal path. This signal is short and unable to notify the microcontroller (due to delay in coding).

4.12.1 Remedy-SCR circuit

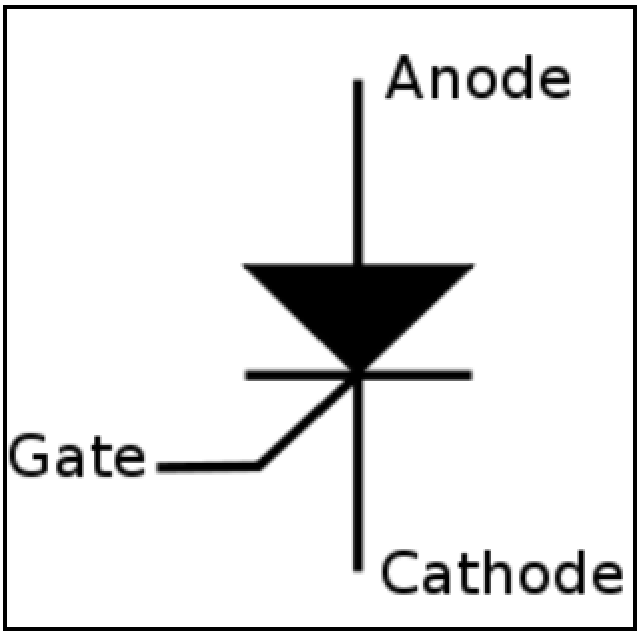


Figure 4.36 Silicon Controlled Rectifier (SCR)

SCR is a four-layer solid state device that controls current. Once the gate of SCR is triggered on (minimum 1 second) , it will turn on constantly unless the power supply is unplug. Hence, it can provide sustainable voltage level to the microcontroller

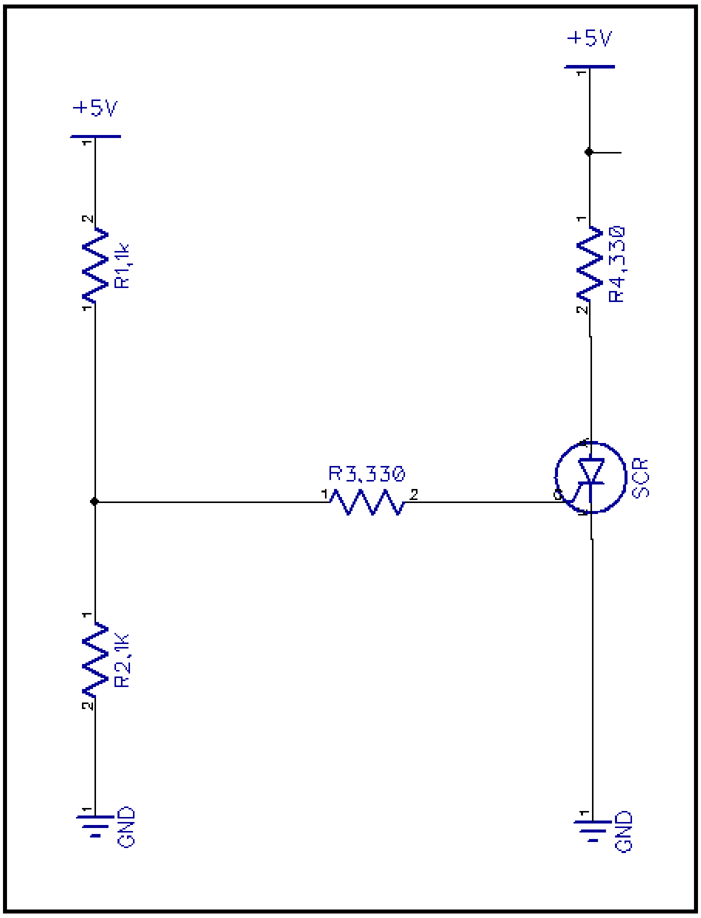


Figure 4.37 SCR Circuitry

4.12.2 SCR Unable to Notify Microcontroller

Before the SCR is triggered, it supplies 5V to output. However, when it is triggered, it only can supply a output voltage from the SCR which is approximately 0.7V. This is too low to trigger the microcontroller.

4.12.2.1 Solution- NAND Gate

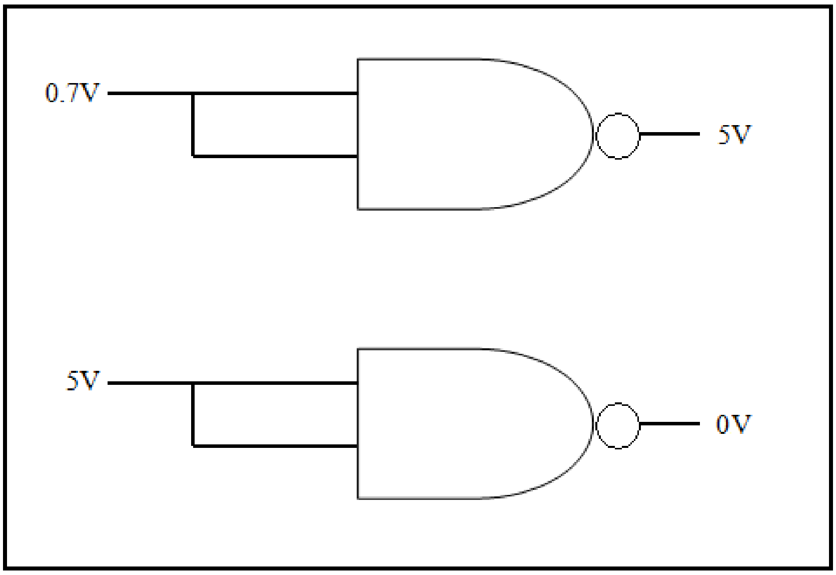


Figure 4.38 NAND gate

When SCR is not triggered, it provides a 5V to output the 5V will be fed into a NAND gate which will output a 0V. This will not notify the microcontroller to turn on the LED (LED is used to notify that someone break-in). On the other hand, when the SCR is triggered, it will provide a 0.7V at the output, the 0.7V will be fed into the NAND gate to outcome a 5V. This will notify the microcontroller to turn on the LED.

4.12.3 Coding to Trigger the Burglary Notification System

```
if(IN==1)
{

LED2=1;
//send_message6();
m++;
}
else
{
LED2=0;
m=0;
}

if(m==1)
{
send_message6();
}
```

If LED2 is on (react from SCR circuit), a notification will be sent to user.

CHAPTER 5

Conclusion and Further Recommendation

5.1 Conclusion

In conclusion, the Final Year Project is a success where the participants are able to complete the project on time. Although, the Power Line Communication (PLC) for data transmission and reception done by Chew Kean Chai is not fully applicable, the final project students still manage to fulfil the requirement for the FYP title “Home Automation” by replacing the PLC with ultrasonic swithes. Apart from that, the project is integrated with a extended project which is home security system. Student, TEO CHE SHEN is able to achieve the objective whereby able to design a hardware by integrating several available electronic devices. Moreover, the programming skills are greatly improved and proper coding algorithm for this project is acknowledged. Along with that, the participants are able to build up a good working team by cooperate and tolerate to each other.

5.2 Potential

Nowadays, people are rushing for daily tasks and being busy, sometimes, they being forgetful and careless. These might be a dangerous scenario where electrical appliances such as microwave, heater and so on might not turned off. This product allows user to turn off the house appliances through SMS. Apart from that it also help to save energy where energy wasting can be prevented. Moreover, security issues are of concerned in current society. House break-in seems to terrify most of the house residents. This system can notify security guards and police officer whenever break-in occurs but without signify the burglar.

5.3 Limitation

Nowadays, normal or old-designed handphones are graduating decrease in demand as smartphones are widely in-used; hence, it will be harder to obtain those handphones for this design. This scenario also happens to serial cable for GSM handphones.

Sometimes, the message that has been sent will not be processed by the microcontroller due to duty cycle error.

The ultrasonic transmitter and receiver are sensitive to environment because the environment may contain ultrasonic sources such as, handphone, radio and so on. These may interrupt the sensor by causing some trigger. Apart from that, the ultrasonic switches seem to be more vulnerable to noise or interrupt as the transmitter and receiver is getting apart.

5.4 Future Recommendation

The GSM circuit can be integrated with more appliances rather than just ultrasonic switches and security system only. For instant, stove turn on or off, energy saving, monitor circuit and so on.

Moreover, the switches can be designed in such a way that, the transmitter and receiver communicate through FM or AM transmission where the sensors does not required to be placed face-to-face. This will eventually stabilise the signal transmission. By the way, the frequency can be adjusted according for different switches.

Nowadays, USB is more popular and efficient for data transmission, the project can be modified to use USB transmission rather than serial transmission.

REFERENCES

Edaboard . (2010). GSM Problem.Retrieved July 18, 2010,
from http://www.edaboard.com/ftopic380629.html

Embedtronic. (2010). Nokia Fbus Connection.Retrieved July 18, 2010,
from http://www.embedtronics.com/nokia/fbus.html

Gsm Server. (2010). Pinout for Sony Erricson. Retrieved July 18, 2010,
http://gsmserver.com/pinout/ericsson/erics_t28.php

Microchip. (2010). HI-TECH C for the PIC10/12/16 MCU Family. Retrieved July 18, 2010, from http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_ PAGE &nodeId=1406&dDocName=en542849

Microchip. (2010). MPLAB Integrated Development Environment . Retrieved July 18, 2010, from http://www.microchip.com/stellent/idcplg?IdcService=SS_GET _PAGE&nodeId=1406&dDocName=en019469&part=SW007002NowMobile.com Limited. (2010).

GSM Modems. Retrieved July 18, 2010, from
http://www.nowsms.com/GSM%20Modems.htm

Riccibitti . (2010). GSM Phone.Retrieved July 18, 2010,
from http://www.riccibitti.com/tinyplanet/tiny_intro.htm

Serasidis. (2010). SMS controlled Circuit. Retrieved July 18, 2010,
from http://www.serasidis.gr/circuits/smscontrol/smscontroller.htm

Sony Ericsson Mobile Communications. (2010). K700i Specifications. Retrieved July 18, 2010, from http://www.sonyericsson.com/cws/products/mobilephones /specifications/t610?cc=us&lc=en

Texas Instruments. (2011). MAX232,MAX232I dual EIA-232 drivers/receivers. Retrieved March 2, 2011, from http://focus.ti.com/lit/ds/symlink/max232.pdf

APPENDICES

APPENDIX A: Computer Programme Listing

HI-TECH C Compilers

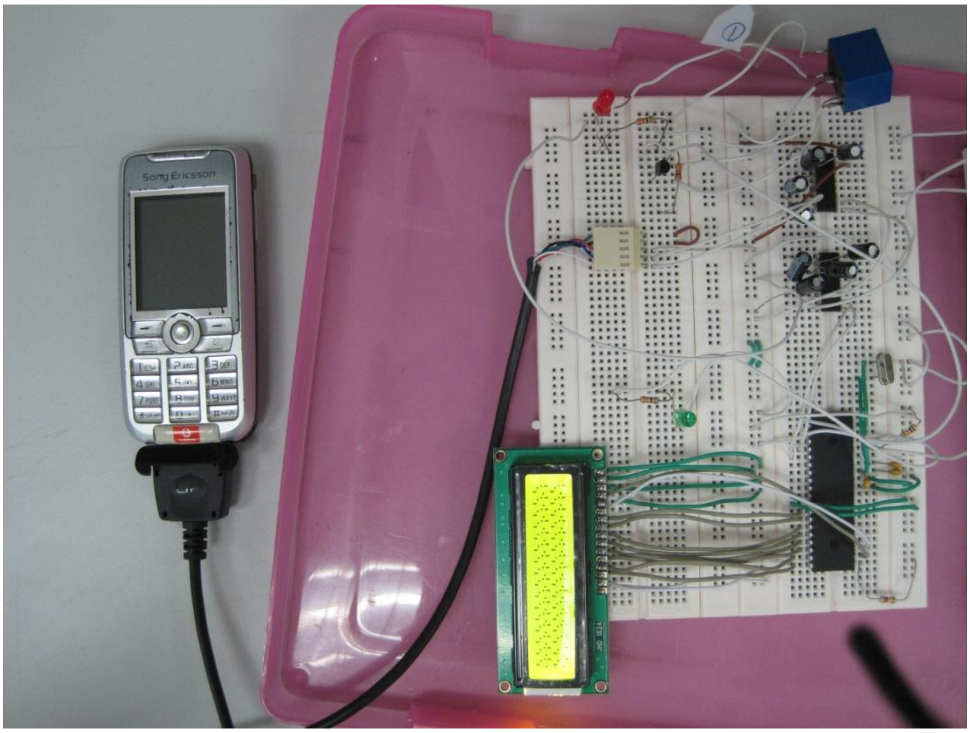
MPLAB C Compiler

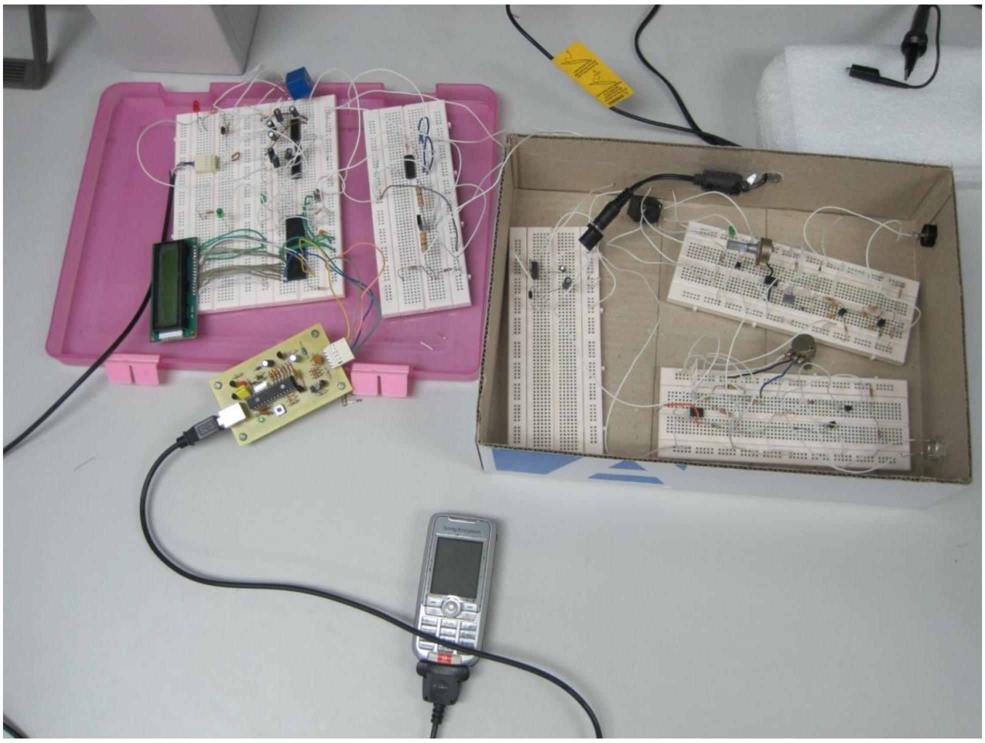
PICkit 2 Development Programmer/Debugger

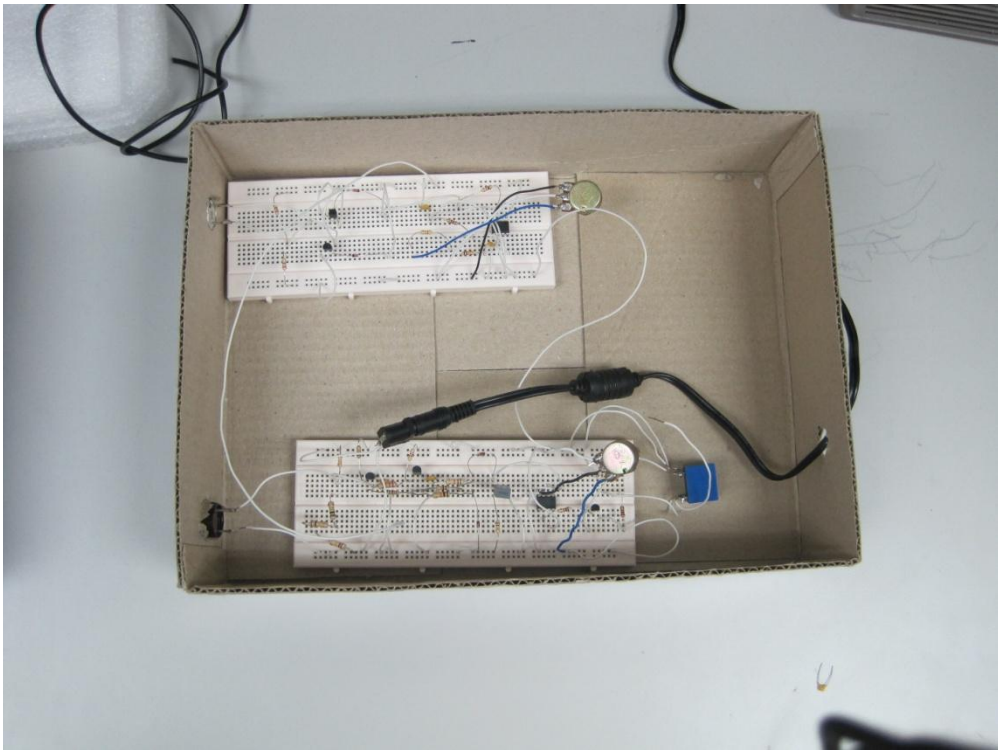
DipTrace

APPENDIX B: Project Planning

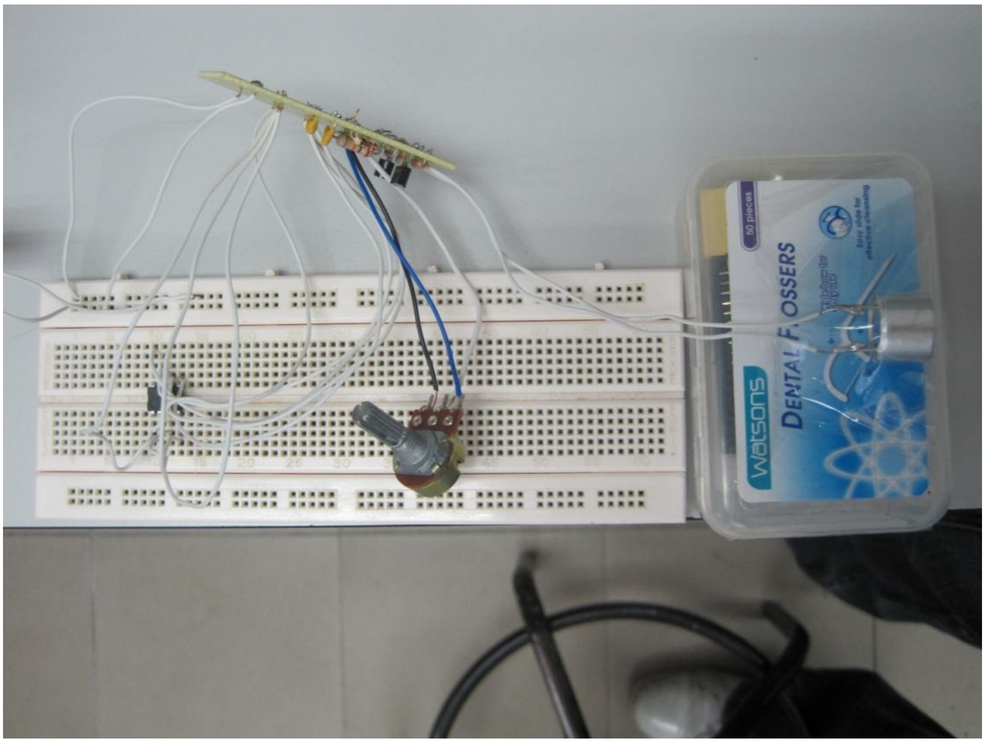
Circuit in Breadboard

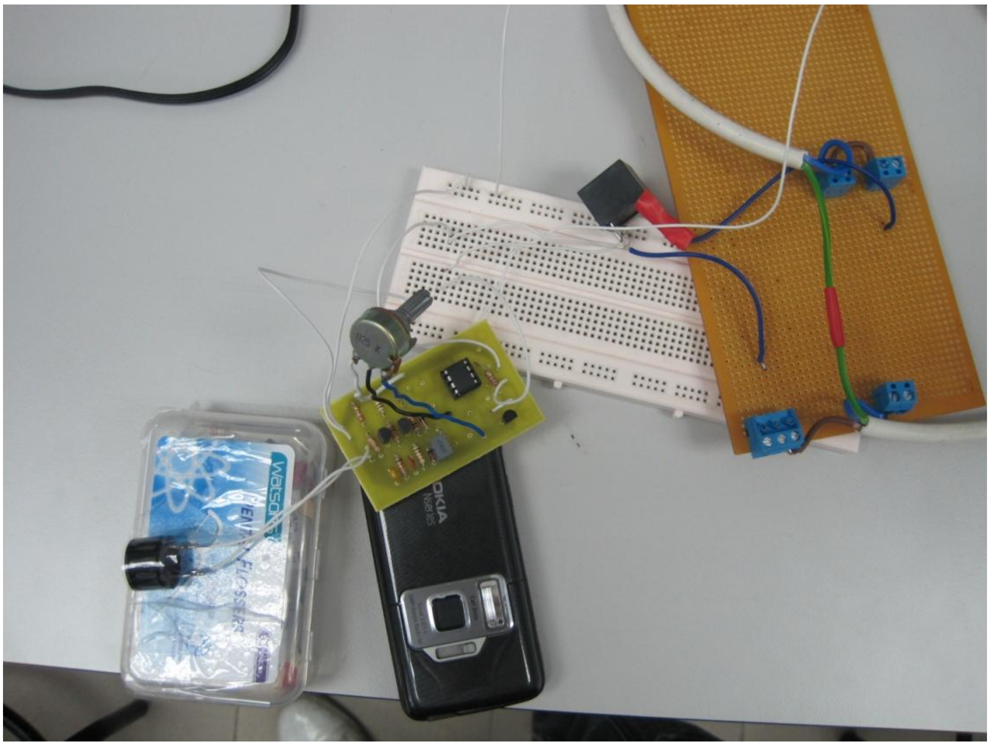






Partial circuit in PCB





Coding:

```
#include<pic.h>
#include<htc.h>
//===============configuration================================
__CONFIG (0x3F32);

#define         lcd                     PORTB
#define         RS                      RA0
#define         E                       RA1
#define         LED                     RD2
#define         LED2            RC2
#define         IN                      RC1
#define _XTAL_FREQ 20000000

//==============FUNCTION
PTOTOTYPE(led)========================
void uart_send(unsigned char data);
unsigned char uart_rec(void);
void long_delay(unsigned short i);
void enable_uart (void);
void send_message1 (void);
void send_message2 (void);
void send_message3 (void);
void send_message4 (void);
void send_message5 (void);
void send_message6 (void);
void e_pulse(void);
void delay(unsigned short i);
void send_char(unsigned char data);
void send_config(unsigned char data);
void lcd_goto(unsigned char data);
void lcd_clr(void);
void dis_num(unsigned long data);
void increment(unsigned long data);
void read_adc(void);
void disp_temp(void);
unsigned short read_temp(void);
int k=0;

//====================Clobal declaration================================
            char uart_data1[]="AT";
            char uart_data2[]="AT+CPMS=\"ME\"";
            char uart_data3[]="AT+CMGR=1";
      //    char uart_data4[]="AT+CMGS=33";
            char uart_data6[]="AT+IPR=9600";
            char uart_data7[]="AT+CMGF=0";
            char uart_data8[]="AT+CMGD=1";
            char uart_data10[]="ATE0";
            char uart_data11[]="AT+CMGS=20";
            char uart_data12[]="0001000a8110839778990004085377 2E31206F6E21";
            char uart_data13[]="AT+CMGS=21";
            char uart_data5[]= "0001000a811083977899000409537 72E31206F666621";
            char uart_data14[]="0001000a811083977899000409496E74727564657221";

            unsigned char received_data1[95];


void main(void)
{


            enable_uart ();
            send_config(0b00000001);        //clear display at lcd
            send_config(0b00000010);        //Lcd Return to home
```

```
        send_config(0b00000110);        //entry mode-cursor increase 1
        send_config(0b00001100);        //diplay on, cursor off and cursor
blink off
        send_config(0b00111000);        //function set

        lcd_clr();
        lcd_goto(0);
        send_message1();
        //LED=1;
        int CHECK = 0;
        int m = 0;
        while(1)
        {
                k=0; //to ask buffer to go back to 1st array

                int i;
                send_message2();
                send_message3();


                for(i=37; i<=45; i++)
                                        {

    send_char(received_data1[i]);

                                        }

                //while(1)
                if(received_data1[70]=='F'                      &&
received_data1[71]=='3'
                &&          received_data1[72]=='3'             &&
received_data1[73]=='B' && received_data1[74]=='E'
                &&          received_data1[75]=='8'             &&
received_data1[76]=='E' && received_data1[77]=='D'
```

```
                                        &&              received_data1[78]=='0'              &&
received_data1[79]=='6')
                                        {

                                              if(LED==1)
                                              {

                                                    send_message4();

                                              }
                                              else
                                              {

                                                    LED=1;
__delay_ms(500);
                                                    send_message4();
                                                    CHECK=1;

                                              }
                                        }

                                        if(received_data1[70]=='F'                           &&
received_data1[71]=='3'
                                        &&              received_data1[72]=='3'              &&
received_data1[73]=='B' && received_data1[74]=='E'
                                        &&              received_data1[75]=='8'              &&
received_data1[76]=='6' && received_data1[77]=='D'
                                        &&              received_data1[78]=='3'              &&
received_data1[79]=='6' && received_data1[80]=='0'
                                        && received_data1[81]=='3')
                                        {

                                              if(LED==0)
                                              {
```

```
                                        send_message5();

                                }

                                else
                                {

                                        LED=0;
__delay_ms(500);
                                        send_message5();
                                        CHECK=0;

                                }
                        }


if(IN==1)
{

LED2=1;
//send_message6();
m++;
}
else
{
LED2=0;
m=0;
}

if(m==1)
{
send_message6();
}
```

```
                                        //To clear all off the arrays
                                        for(i=0; i<=94; i++)
                                                                          {

      received_data1[i]=0;

                                                                          }



                                        __delay_ms(1000);
            }

}



void send_message1 (void)
{
unsigned short i;

//ATE0
      for(i=0; i<=3; i++)
                                                    {
```

```
                                        uart_send(uart_data10[i]);

                                    }

                    uart_send(0x0A);
                    uart_send(0x0D);

__delay_ms(1000);
//AT+IPR=9600
        for(i=0; i<=10; i++)
                                    {
                                        uart_send(uart_data6[i]);

                                    }

                    uart_send(0x0A);
                    uart_send(0x0D);

__delay_ms(1000);



//AT+CMGF=0

                    for(i=0; i<=8; i++)
                                    {
                                        uart_send(uart_data7[i]);

                                    }
                    uart_send(0x0A);
                    uart_send(0x0D);
                        __delay_ms(1000);
```

```
//AT+CPMS=ME

    for(i=0; i<=12; i++)
                        {
                            uart_send(uart_data2[i]);

                        }
        uart_send(0x0A);
        uart_send(0X0D);

        __delay_ms(1000);
}


void send_message2 (void)
{

unsigned short i;
//AT+CNMI=2,1,0,0,0

/*      for(i=0; i<=16; i++)
                    {
                            uart_send(uart_data9[i]);

                        }
        uart_send(0x0A);
        uart_send(0x0D);
            __delay_ms(1000);*/




//AT+CMGR=1
```

```
                    for(i=0; i<=8; i++)
                                        {
                                                  uart_send(uart_data3[i]);

                                        }
                    uart_send(0x0A);
                    uart_send(0x0D);
                              __delay_ms(1000);
}



void send_message3 (void)
{

unsigned short i;



//AT+CMGD=1


                    for(i=0; i<=8; i++)
                                        {
                                                  uart_send(uart_data8[i]);

                                        }
                    uart_send(0x0A);
                    uart_send(0x0D);
                              __delay_ms(1000);
}



void send_message4 (void)
{
    unsigned short i;


//AT+CMGD=1


        for(i=0; i<=9; i++)
                        {
                            uart_send(uart_data11[i]);

                        }
        uart_send(0x0A);
        uart_send(0x0D);
        __delay_ms(1000);

        for(i=0; i<=41; i++)
                        {
                            uart_send(uart_data12[i]);

                        }
        uart_send(0x1A);
        uart_send(0x0D);
        __delay_ms(1000);

}

void send_message5 (void)
{

    unsigned short i;


//AT+CMGD=1
```

```
                    for(i=0; i<=9; i++)
                                        {
                                                uart_send(uart_data13[i]);

                                        }
                    uart_send(0x0A);
                    uart_send(0x0D);
                    __delay_ms(1000);

                    for(i=0; i<=43; i++)
                                        {
                                                uart_send(uart_data5[i]);

                                        }
                    uart_send(0x1A);
                    uart_send(0x0D);
                    __delay_ms(1000);

}


void send_message6 (void)
{

          unsigned short i;


//AT+CMGD=1


                    for(i=0; i<=9; i++)
                                        {
                                                uart_send(uart_data13[i]);
```

```
                        }
        uart_send(0x0A);
        uart_send(0x0D);
        __delay_ms(1000);

        for(i=0; i<=43; i++)
                        {
                            uart_send(uart_data14[i]);

                        }
        uart_send(0x1A);
        uart_send(0x0D);
        __delay_ms(1000);

}
void enable_uart (void)
{
        ADCON1  =   0b00000110;
        TRISB = 0b00000000;
        PORTB   =   0;
        TRISD = 0b00000000;

        TRISD3  =   1;
        PORTD   =   0;
        TRISD2  =   0;

        TRISD4  =   0;
        TRISA = 0b000;
        PORTA   =   0;
        TRISB = 0;
        TRISC6  =   0;
        TRISC7  =   1;
        TRISC1 = 1;
```

```
                TRISC2 =0;
                PORTB       =       0;
                PORTC       =       0;
                SPBRG       =       129;
                BRGH =      1;
                TX9         =       0;
                RX9         =       0;
                SPEN  =     1;
                CREN =      1;
                GIE         =       1;
                PEIE   =    1;
                RCIE   =    1;
                TXEN =      1;

}



// Interrupt ISR
void interrupt ISR (void)
{

        if(RCIF==1)

{
received_data1[k]=RCREG;
}

        k++;

}




// Send Function
```

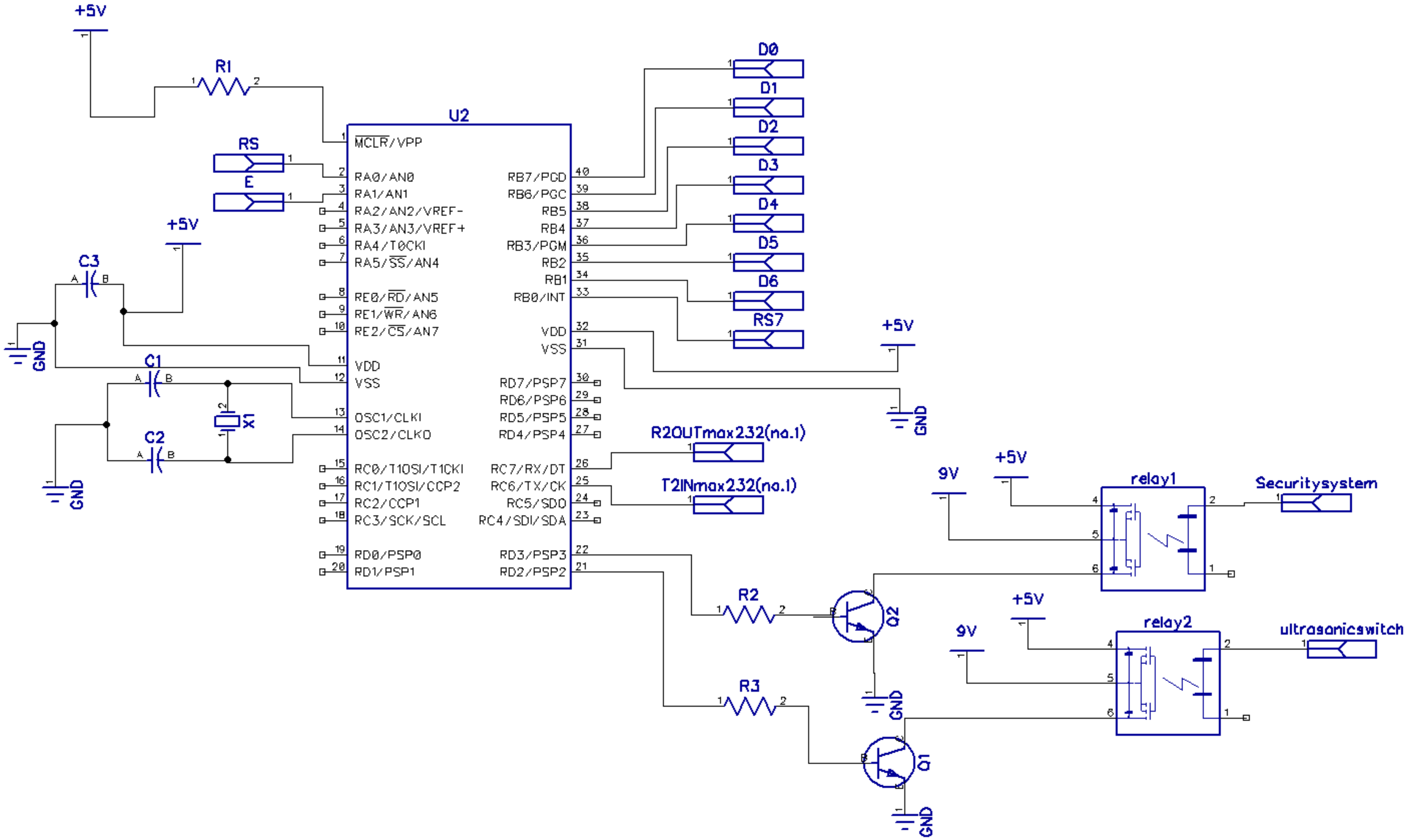
```
void uart_send(unsigned char data)
{
        while(TXIF==0);                         //only send the new data after
        TXREG=data;                             //the previous data finish sent
}


//=================subroutine                    LCD                     setting
=========================

void send_config(unsigned char data)
{
        RS=0;
        lcd=data;
        delay(500);
        e_pulse();
}

void e_pulse(void)
{
        E=1;
        delay(500);
        E=0;
        delay(500);
}

void send_char(unsigned char data)
{
        RS=1;
        lcd=data;
        delay(500);
        e_pulse();
}
```

```
void lcd_goto(unsigned char data)
{
        if(data<16)
        {
                send_config(0x80+data);
        }
        else
        {
                data=data-20;
                send_config(0xc0+data);
        }
}


void lcd_clr(void)
{
        RS=0;
        send_config(0x01);
        delay(600);
}


void dis_num(unsigned long data)
{
        unsigned char hundred_thousand;
        unsigned char ten_thousand;
        unsigned char thousand;
        unsigned char hundred;
        unsigned char tenth;

        hundred_thousand = data/100000;
        data = data % 100000;
        ten_thousand = data/10000;
```

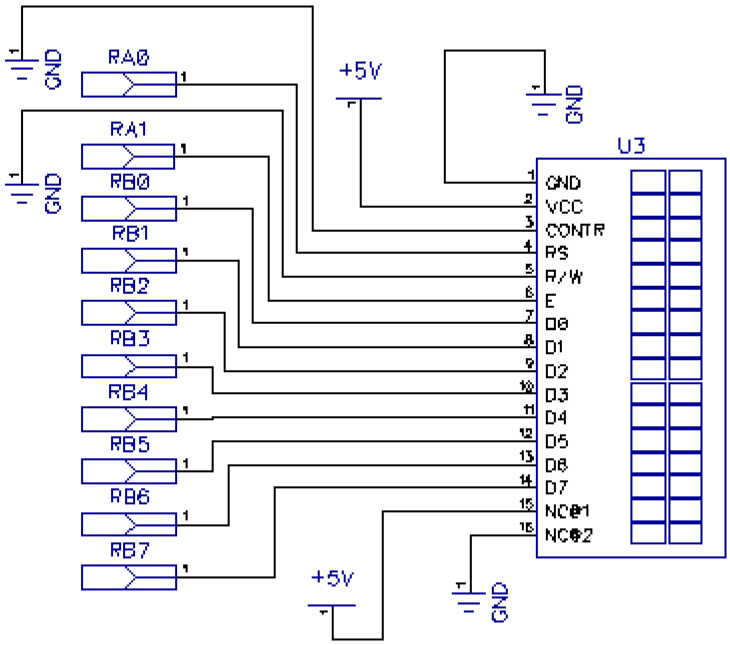
```
    data = data % 10000;
    thousand = data / 1000;
    data = data % 1000;
    hundred = data / 100;
    data = data % 100;
    tenth = data / 10;
    data = data % 10;

    if(hundred_thousand>0)
    {
        send_char(hundred_thousand + 0x30);     //0x30 added to become ASCII code
        send_char(ten_thousand + 0x30);
        send_char(thousand + 0x30);
        send_char(hundred + 0x30);
        send_char(tenth + 0x30);
        send_char(data + 0x30);
    }

    else if(ten_thousand>0)
    {
        send_char(ten_thousand + 0x30);   //0x30 added to become ASCII code
        send_char(thousand + 0x30);
        send_char(hundred + 0x30);
        send_char(tenth + 0x30);
        send_char(data + 0x30);
    }
    else if(thousand>0)
    {
        send_char(thousand + 0x30); //0x30 added to become ASCII code
        send_char(hundred + 0x30);
        send_char(tenth + 0x30);
        send_char(data + 0x30);
```

```
		}
		else if(hundred>0)
		{
			send_char(hundred + 0x30);	//0x30 added to become ASCII code
			send_char(tenth + 0x30);
			send_char(data + 0x30);
		}
		else if(tenth>0)
		{
			send_char(tenth + 0x30);		//0x30 added to become ASCII code
			send_char(data + 0x30);
		}
		else send_char(data + 0x30);  //0x30 added to become ASCII code
}

void increment(unsigned long data)
{
		unsigned short j;
		for(j=10;j>0;j--)
		{	lcd_goto(32);
			data=data+1;
			dis_num(data);
			delay(10000);
		}

}
//==================subroutine DELAY==========================
void delay(unsigned short i)
{
		for(;i>0;i--);
}
```

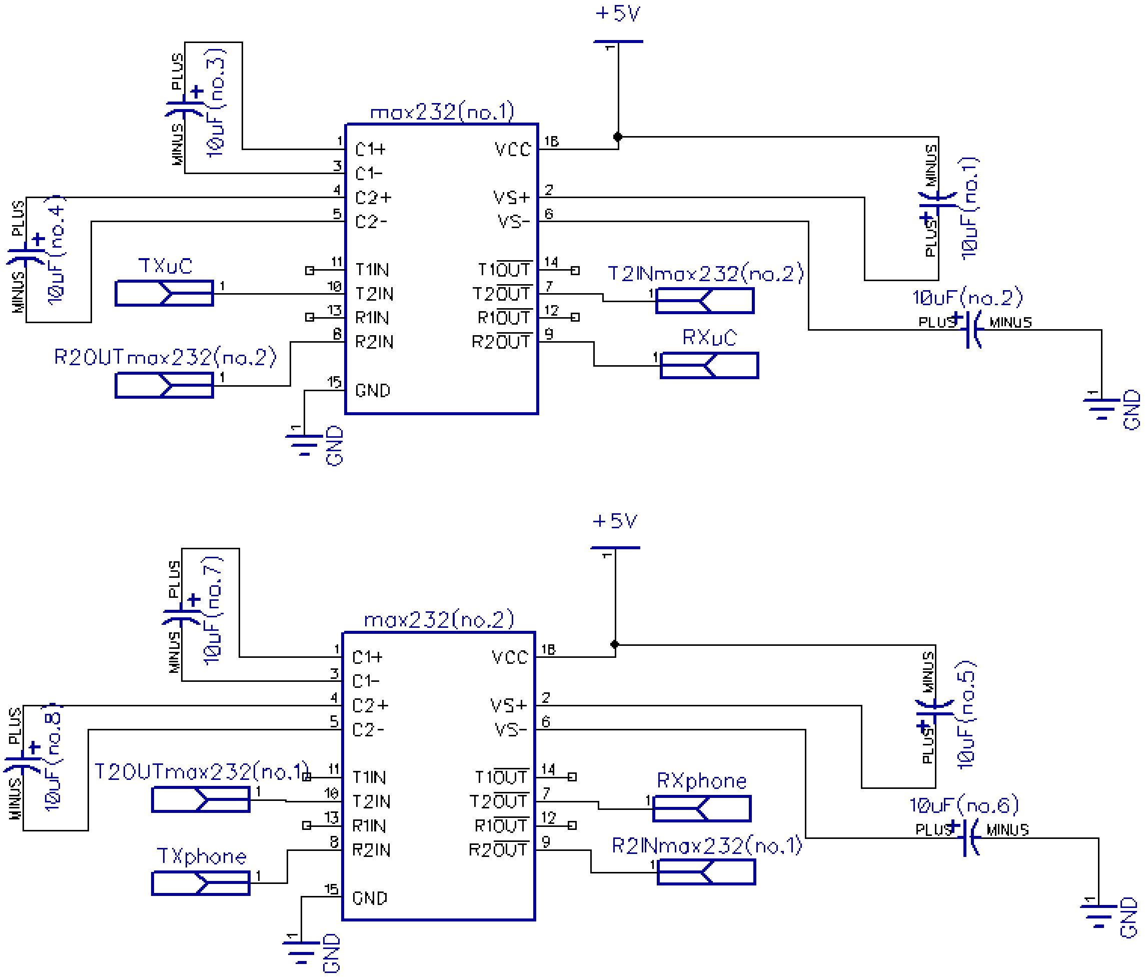
PIF16F877A to relays



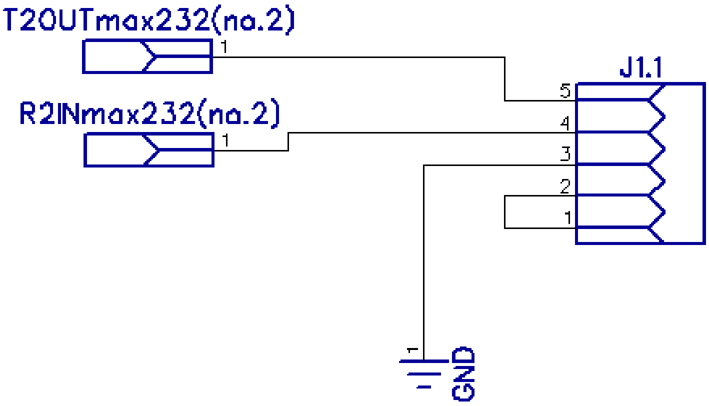
From PIC16F877A to LCD



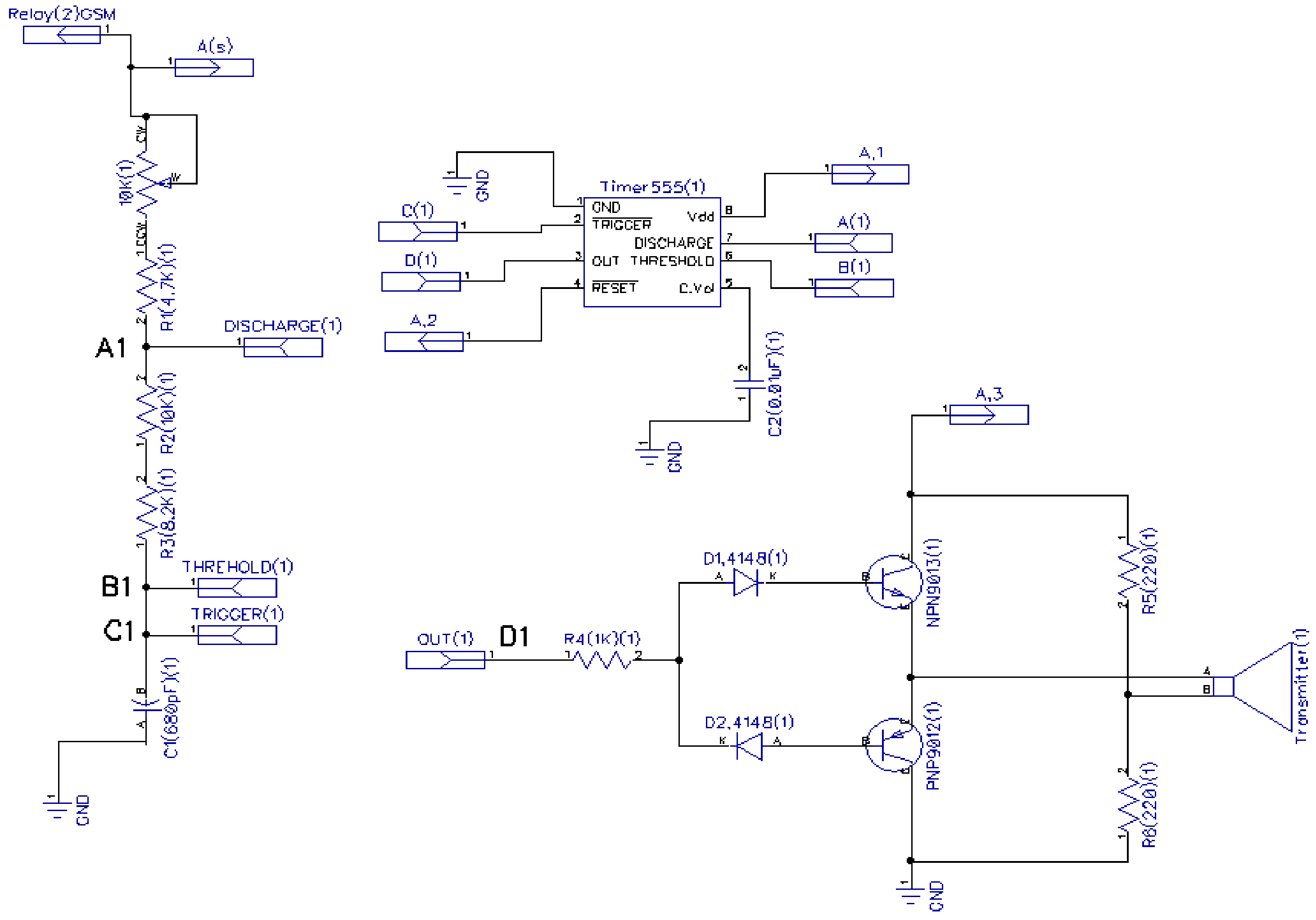
From PIC16F877A to two MAX2332 ICs To Phone Connector



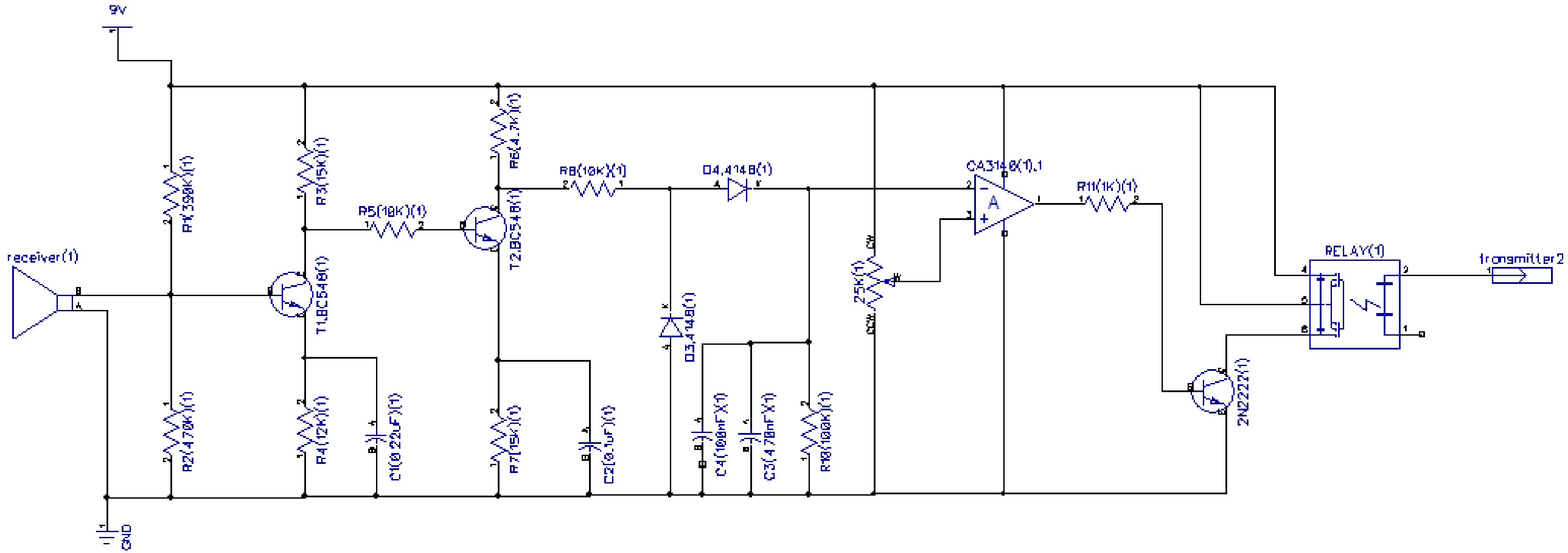
Phone Connector



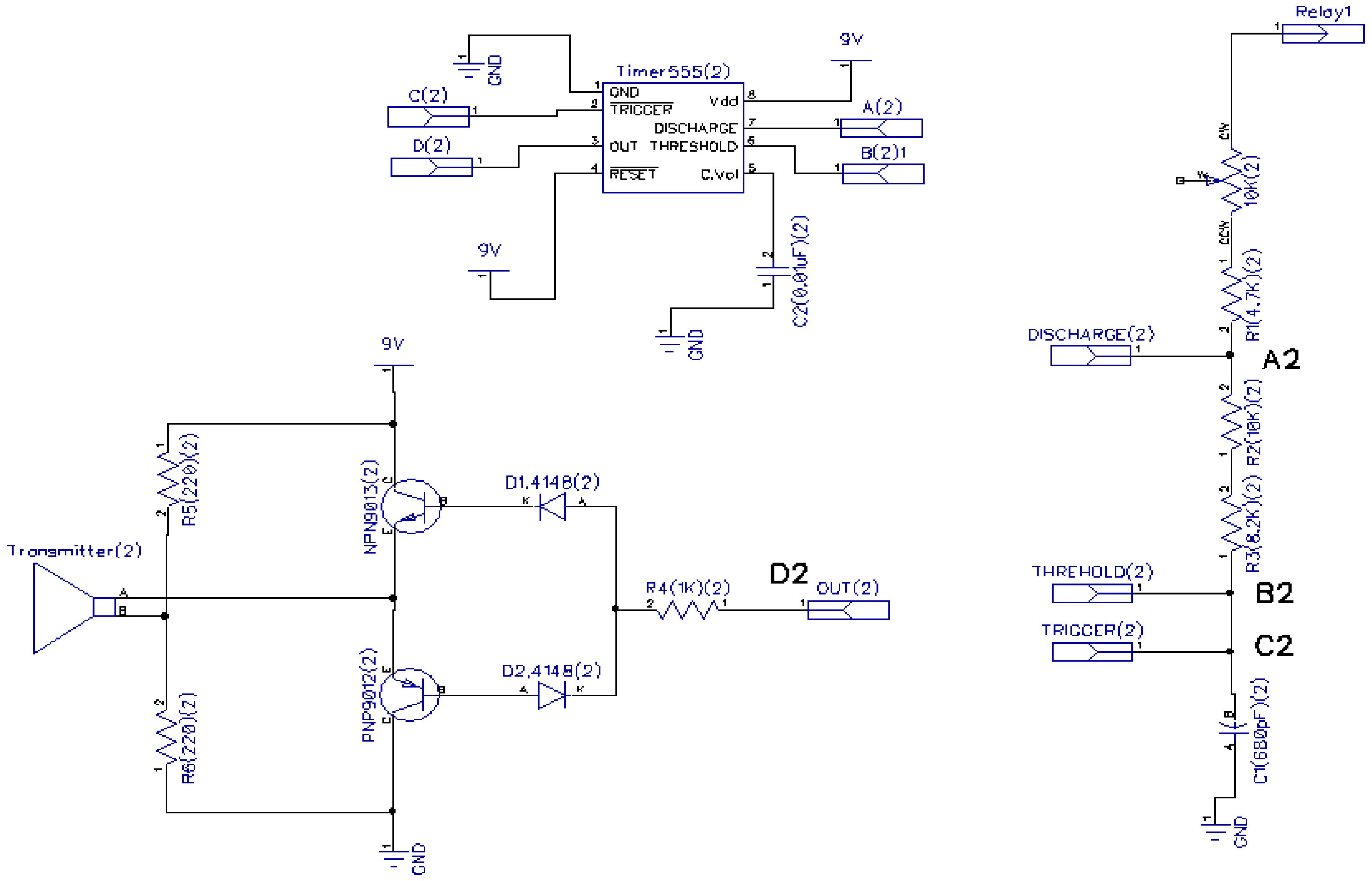
From PIC16F877A to Transmitter no.1 for Burglary Notification system



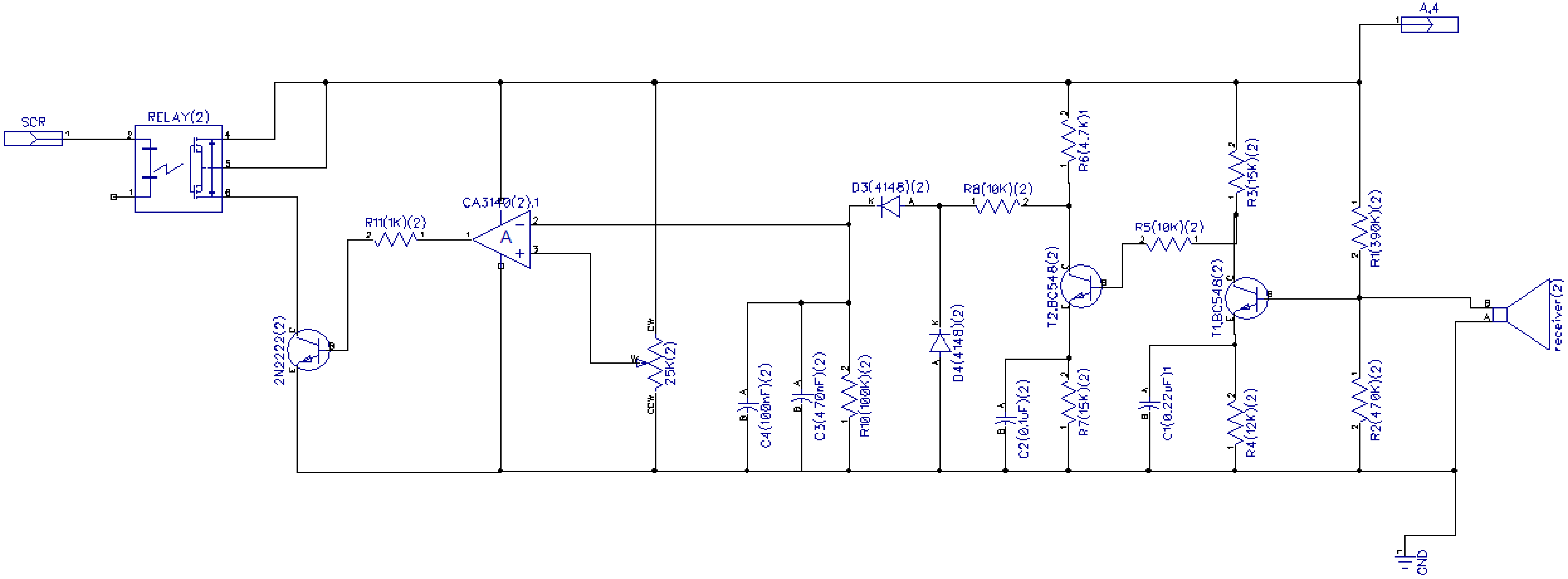
Receiver no.1 for Burglary Notification system



Form Tranmitter no1 to Receiver N0.2



From Receiver No.2 to PIC16f877A


A.4
SCR
RELAY(2)
CA3140(2).1
R11(1K)(2)
2N2222(2)
CW
W
25K(2)
CCW
C4(100nF)(2)
C3(470nF)(2)
R10(100K)(2)
D3(4148)(2)
D4(4148)(2)
R8(10K)(2)
R6(4.7K)1
T2.BC548(2)
C2(0.1uF)(2)
R7(15K)(2)
R5(10K)(2)
R3(15K)(2)
T1.BC548(2)
C1(0.22uF)1
R4(12K)(2)
R1(390K)(2)
R2(470K)(2)
receiver(2)
GND