

**Improvement on Ultrasonic Level Detector  
For Solid and Liquid Applications**

by

Md Afifuddin bin Daud @ Ab Aziz

Dissertation Report submitted in partial fulfillment of  
the requirements for the  
Bachelor of Engineering (Hons)  
(Electrical and Electronics Engineering)

DECEMBER 2004

Universiti Teknologi PETRONAS  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan

*This Great gratitude goes to GOD the Almighty.*

*To my beloved father and mother,*

*Hj Daud @ Ab Aziz bin Ab Rahman*

*Hjh Norizan bin Ab Rahmin*

*Also to my beloved sister,*

*Norizzati binti Hj Daud @ Ab Aziz, and*

*Nursofwati binti Hj Daud @ Ab Aziz*

*To all my friends, and then last but not least to my precious, Nany.*

*All your supports will be kept in my soul.*

*Thank you!*

# CERTIFICATION OF APPROVAL

**Improvement on Ultrasonic Level Detector for Solid and Liquid Application**

by

MD AFIFUDDIN BIN DAUD @ AB AZIZ

A project dissertation submitted to the  
Electrical and Electronics Engineering Programme  
Universiti Teknologi PETRONAS  
in partial fulfillment of the requirement for the  
BACHELOR OF ENGINEERING (Hons)  
(ELECTRICAL AND ELECTRONICS ENGINEERING)

Approved by,



(Associate Professor Dr Mohammad bin Awan)

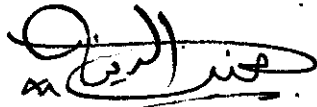
UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

December 2004

## CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done unspecified sources or persons.



---

MD AFIFUDDIN BIN DAUD @ AB AZIZ

## **ABSTRACT**

This report describes the design for a small scale of Ultrasonic Level Detector (ULD). The main parts of the ULD are Ultrasonic Transmitter and Receiver, Level Processor, Serial Communicator and Visual Basic 6.0 software. A microcontroller; PIC16F877 and PIC16F84A are used for the data processing and serial communication. All the circuits used for ULD and the source codes are included in this thesis. Some experiments are carried out in order to verify the design. Further improvements are suggested for future works.

## ACKNOWLEDGEMENT

First and foremost, the author would like to thank ALLAH the Almighty for all HIS blessing that made all things possible for while doing this research.

The author also would like to convey his highest gratitude to *Associate Prof Dr Mohammad bin Awan* for his guidance and assistance through the period of this project as supervisor. Without his advices and helps, the project may not be able to be completed within the given period of time. Then, this gratitude also dedicated to *Mr Wan Ab Aziz bin Wan Jusoh*, Senior Instrument Engineer in PETLIN (M) SDN BHD, on his idea for initiating this project during mid year of 2003. The author also would like to thanks Associate Prof Pham for his suggestions to improve this project.

The compliment should also go to all Electrical and Electronics Engineering Laboratory technicians for their assistance in laboratory works especially *Miss Siti Hawa*.

Then, the author also dedicated his gratitude to all Electrical and Electronics Engineering Final Year 2004 students for their contributions in this project especially to *Mr Azizan, Mr Rizaudin and Mr Ahmad Rizal*.

Last but not least, the author would like to thank all the people that have contributed in completing this project especially to all laboratory mates and friends for their support and comments.

# TABLE OF CONTENTS

|  |             |
|--|-------------|
| <b>CERTIFICATION OF APPROVAL.....</b>                | <b>i</b>    |
| <b>CERTIFICATION OF ORIGINALITY.....</b>             | <b>ii</b>   |
| <b>ABSTRACT.....</b>                                 | <b>iii</b>  |
| <b>ACKNOWLEDGEMENT.....</b>                          | <b>iv</b>   |
| <b>TABLE OF CONTENTS.....</b>                        | <b>v</b>    |
| <b>LIST OF TABLES.....</b>                           | <b>viii</b> |
| <b>LIST OF FIGURES.....</b>                          | <b>viii</b> |
| <b>LIST OF ABBREVIATION.....</b>                     | <b>x</b>    |
| <b>CHAPTER 1 : INTRODUCTION.....</b>                 | <b>1</b>    |
| 1.1    BACKGROUND OF STUDY.....                      | 1           |
| 1.2    PROBLEM STATEMENT.....                        | 2           |
| 1.3    OBJECTIVES AND SCOPE OF STUDY.....            | 2           |
| <b>CHAPTER 2 : LITERATURE REVIEW AND THEORY.....</b> | <b>4</b>    |
| 2.1    OVERALL DESIGN OF THE SYSTEM.....             | 4           |
| 2.2    ULTRASONIC WAVE CHARACTERISTICS.....          | 4           |
| 2.3    SOLID AND LIQUID CHARACTERISTICS.....         | 5           |
| 2.4    OPERATION OF ULTRASONIC SENSOR.....           | 6           |
| 2.4.1    The Transmitter and Receiver.....           | 6           |
| 2.4.2    The Basic Features of the Beam Light.....   | 7           |
| 2.4.3    Ultrasonic Sensor's Insertion Loss.....     | 7           |

|   |  |           |
|---|--|-----------|
| 2.5   | THE OPERATION OF PIC16F877 MICROCONTROLLER.....                        | 8         |
| 2.5.1   | The Overview on the PIC16F877 Microcontroller<br>File Register.....    | 9         |
| 2.5.2   | Overview of 8-Channel to 10-Bit Analog-Digital<br>Converter (ADC)..... | 10        |
| 2.5.3   | ADC Module in PIC16F877 Microcontroller.....                           | 11        |
| 2.6   | THE OPERATION OF PIC16F84A MICROCONTROLLER.....                        | 12        |
| 2.6.1   | The Overview on the PIC16F84A Microcontroller<br>File Register.....    | 13        |
| 2.6.2   | Common Features of PIC16F84A Microcontroller.....                      | 14        |
| <b>CHAPTER 3 : METHODOLOGY AND PROJECT WORKS.....</b> |  | <b>16</b> |
| 3.1   | HARDWARE.....  | 16        |
| 3.1.1   | The Transmitter Circuit.....   | 16        |
| 3.1.2   | The Receiver Circuit.....  | 17        |
| 3.1.3   | PIC16F877 Microcontroller Circuit.....                                 | 18        |
| 3.1.4   | The Personal Computer (PC) Serial Communicator Circuit.....            | 19        |
|   | 3.1.4.1 PIC16F84 Microcontroller Circuitry.....                        | 19        |
|   | 3.1.4.2 MAX232 Circuitry.....  | 19        |
| 3.1.5   | The Other Additional Features for the Instrument.....                  | 21        |
| 3.2   | SOFTWARE.....  | 21        |
| 3.2.1   | PIC16F877 Microcontroller Assembly Language.....                       | 21        |
| 3.2.2   | PIC16F84A Microcontroller C programming.....                           | 22        |
| 3.2.3   | Visual Basic 6.0 Interface Programming.....                            | 23        |
| 3.3   | EXPERIMENT AND DEVELOPMENT.....  | 25        |
| 3.3.1   | The Transmitter Circuit Construction.....                              | 25        |
| 3.3.2   | The Receiver Circuit Construction.....                                 | 26        |
| 3.3.3   | The Microcontroller PIC16F877 Circuit Construction.....                | 29        |
| 3.3.4   | PC Serial Communicator Circuit Construction.....                       | 31        |
| 3.3.5   | Power Supply to the Circuit.....                                       | 32        |
| 3.3.6   | Visual Basic 6.0 Serial Interface Development.....                     | 32        |



|  |   |           |
|--|---|-----------|
| 3.4  | COMSSIONING AND VERIFICATION.....                               | 33        |
| 3.4.1  | Components Assembly.....  | 33        |
| 3.4.2  | Problem Raised and Troubleshoot.....                            | 33        |
| <b>CHAPTER 4 : RESULTS AND DISCUSSION.....</b> |   | <b>36</b> |
| 4.1  | RESULTS.....  | 36        |
| 4.1.1  | Distance Variation of the Transmitter and Receiver Circuits.... | 36        |
| 4.1.2  | Microcontroller PIC16F877 Clock Set Up.....                     | 39        |
| 4.1.3  | PC Serial Communicator with Visual Basic 6.0.....               | 40        |
| 4.2  | DISCUSSION.....   | 42        |
| 4.2.1  | Real World Application and Small Scale Model.....               | 42        |
| <b>CHAPTER 5 : CONCLUSION.....</b>             |   | <b>43</b> |
| 5.1  | CONCLUSION.....   | 43        |
| 5.2  | RECOMMENDATION.....   | 44        |
| <b>REFERENCES.....</b>                         |   | <b>45</b> |
| <b>APPENDICES.....</b>                         |   | <b>46</b> |

## LIST OF TABLES

|           |   |
|-----------|---|
| Table 3.1 | Range of the input voltage in binary and hexadecimal.                             |
| Table 3.2 | Microsoft Comm Control 6.0 Properties   |
| Table 3.3 | Summarization of the faulty and method taken to overcome                          |
| Table 4.1 | Mean voltage versus distance measured   |
| Table 4.2 | The instruction set and assignment of the variables.                              |
| Table 4.3 | Summary of the level display on VB when the assigned pin on PIC16F84 is activated |
| Table 4.4 | Result from the experiment on pilot plant vessel                                  |

## LIST OF FIGURES

|           |  |
|-----------|--|
| Figure 1  | The application of ULD   |
| Figure 2  | Overall concept of the design  |
| Figure 3  | A Simple Continuous Ultrasonic Wave  |
| Figure 4  | Monostatic and bistatic transducers (a) Monostatic transmitter and monostatic receiver (b) Bistatic transmitter/receiver |
| Figure 5  | Interference effects (a) The basic feature of the beam profile (b) Interference effect in the near field.                |
| Figure 6  | Microcontroller 16F877   |
| Figure 7  | PIC16F877 register map file  |
| Figure 8  | Simplified block diagram of the PIC16F877 ADC module   |
| Figure 9  | ADCON1 Register  |
| Figure 10 | ADCON0 Register  |
| Figure 11 | PIC16F84A pin out  |
| Figure 12 | Simplified Block Diagram for PIC16F84A   |
| Figure 13 | Components of a Microcontroller  |
| Figure 14 | Preliminary ultrasonic transducer transmitter schematic  |
| Figure 15 | Ultrasonic receiver schematic diagram  |
| Figure 16 | Microcontroller PIC16F877 circuitry  |

|           |   |
|-----------|---|
| Figure 17 | Microcontroller PIC16F84A circuitry                                     |
| Figure 18 | MAX232 circuitry  |
| Figure 19 | The overall circuit of the serial communicator                          |
| Figure 20 | The flow chart of function for C programming for PIC16F84A              |
| Figure 21 | VB window interface   |
| Figure 22 | A 40 kHz square-wave signal from the signal generator via oscilloscope. |
| Figure 23 | A filter of the receiver circuit.                                       |
| Figure 24 | The output from the filter for a source of 1 V <sub>p-p</sub> , 40 kHz  |
| Figure 25 | A half-wave rectifier circuit with a capacitance effect                 |
| Figure 26 | Half-wave rectifier effects and capacitance effects                     |
| Figure 27 | The amplifier circuit configuration                                     |
| Figure 28 | Signal from output of the complete receiver circuit                     |
| Figure 29 | The PIC16F877 microcontroller card.                                     |
| Figure 30 | Flow chart of the PIC16F877 microcontroller function                    |
| Figure 31 | The Serial Communicator card.   |
| Figure 32 | The serial communicator's circuit work's flow                           |
| Figure 33 | Basic connection for IC voltage regulator                               |
| Figure 34 | VB MSComm application environments                                      |
| Figure 35 | Graph of voltage versus distance before regulate by 5V                  |
| Figure 36 | The vessel's dimension in centimeter                                    |
| Figure 37 | The plotted graph of product level versus the voltage                   |

## LIST OF ABBREVIATION

|          |   |
|----------|---|
| AC       | Alternating Current                                 |
| ADC      | Analog-Digital-Converter                            |
| ADRESH   | Address HIGH  |
| ADRESL   | Address LOW   |
| ADCON    | Analog-to-Digital Converter                         |
| BOR      | Brown-Out-Reset                                     |
| CCP      | Command Control Processor                           |
| DC       | Direct Current                                      |
| EEPROM   | Electrically Erasable Programmable Read-Only Memory |
| EWB      | Engineering Workbench                               |
| FYP      | Final Year Project                                  |
| GUI      | Graphical User Interface                            |
| LED      | Light Emitting Diode                                |
| LCD      | Liquid Crystal Display                              |
| MSComm   | Microsoft Communication                             |
| PETRONAS | Petroleum Nasional Berhad                           |
| PWM      | Pulse Width Modulation                              |
| RAM      | Random Access Memory                                |
| RX       | Receiver  |
| TX       | Transmitter   |
| UART     | Universal Asynchronous Receiver/Transmitter         |
| ULD      | Ultrasonic Level Detector                           |
| UTP      | University Technology PETRONAS                      |
| VB       | Visual Basic  |

# CHAPTER 1

## INTRODUCTION

### 1.1 BACKGROUND OF STUDY

*Ultrasonic Level Detector* (ULD) has been used for everyday-life applications as well as in industrial applications. In industry, the ULD is mounted on the top of the tank. So that it can sense the product level via the voltage different to the microcontroller which is also known as PIC. The sonar transducer would emit a sound wave to the product surface in the tank. The sound wave will be reflected by the surface of the solid or liquid. This type of sonar is technically known as *echo*. The receiver transducer will receive this echo as shown in Figure 1. From the output intensity of the receiver, the PIC would be able to signal the operator when the surface level is at HIGH, MEDIUM or LOW. The display using Light Emitting Diode (LED)'s on the instrument are used as indicators and the operators can observe the level via software display.

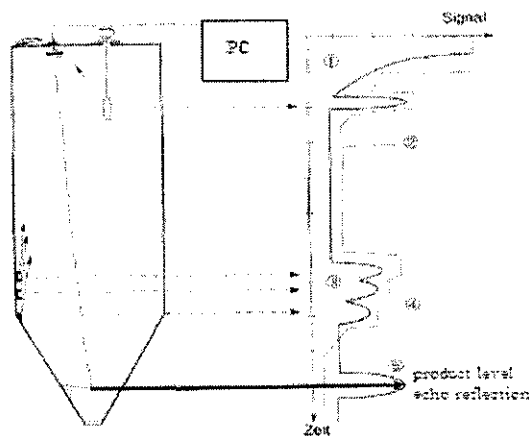


Figure 1: The application of ULD

This project is carried out in three main steps. The understanding on the circuit diagram of ULD is very crucial. This is followed by the development of the C programming or assembly language for PIC16877 microcontroller to indicate the detected level using LED's. The communication to a computer is done using Visual Basic program via a serial communicator.

## **1.2 PROBLEM STATEMENT**

ULD is one of the devices that are used to detect the level of solid or liquid in a storage vessel. Current application is already can fulfill the plant requirement to be a product references especially for solid or grain product. However, for the sake of research, more improvements need to be done for some crucial circumstances during production. In addition, this device is also used as one of the control loop element for vessel like a buffer tank or product storage silo.

The study on the ultrasonic level detector and the improvement needed was started with the development of the device. Then, the further studies on the previous and demand for the current ULD requirement. Therefore, by improving the existing system and hopefully that this new version of ULD can give the better result according to its trustworthiness, reliability and accuracy.

## **1.3 OBJECTIVE AND SCOPE OF STUDY**

The objectives to improve the ULD are as follows:

- i. To increase the efficiency of the signal transmission from the ultrasonic driver to the receiver according to solid and liquid applications.
- ii. To increase the accuracy of level detection in order to make it as a reference point for plant production estimation.

Therefore, this instrument needs to be developed by following the commercial aspect for the plant requirement. As a result, the understanding on the operation of plant

management system and the structure of PIC microcontroller are required. The study on the computer-device communication is also required to complete this project.

After completing the ULD prototype, the commissioning procedure needs to be carried out to verify the reliability of the ULD.

There are several development steps in completing this project. They are as follows:

- i. The operation of the overall system.
- ii. The electrical circuitry and the lay out of the instrument.
- iii. The development of assembly language or C code for microcontroller.
- iv. The communication with a computer for monitoring purposes.

The Gantt chart shown in *Appendix A* is a guide in completing this project.

## CHAPTER 2

### LITERATURE REVIEW AND THEORY

#### 2.1 OVERALL DESIGN OF THE SYSTEM

This instrument should fulfill the current plant requirement. This will enable the plant operation and production run smoothly. The block diagram in Figure 2 shows the overall design of the system.

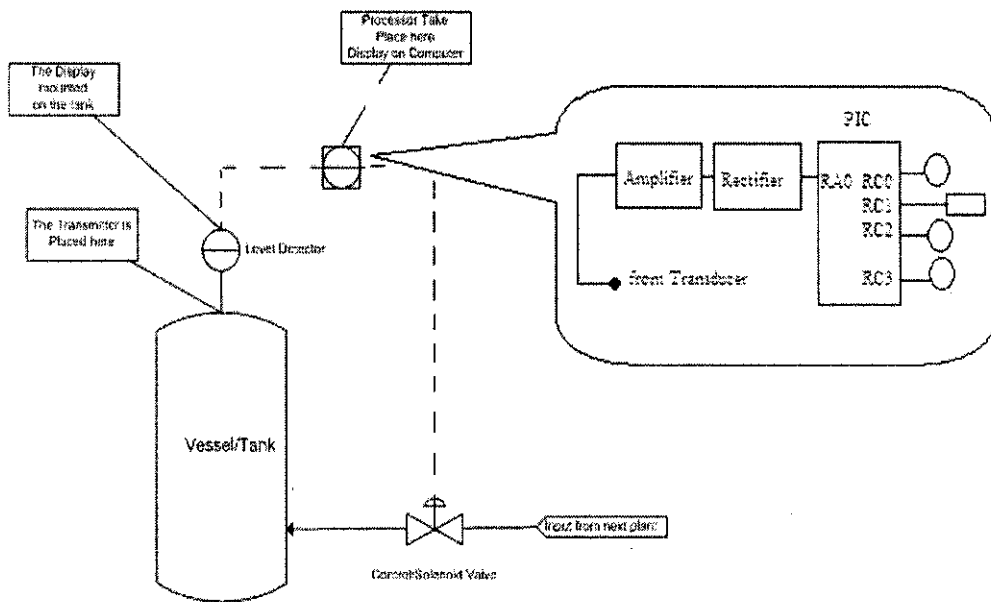


Figure 2: Overall concept of the design

#### 2.2 ULTRASONIC WAVE CHARACTERISTICS

The ultrasound velocity is 380 meter per second (m/s). The echo is the reflection of the sound from the transmitter. The mean frequency for the ultrasound is 20 kHz.



It also involves with the refraction and interference characteristics. The sound direction of motion is in longitudinal or transverse modes.

The frequency,  $f$ , in Hz can be defined as the number of oscillation per second and it is illustrated using a simple continuous wave shown in Figure 3 [1].

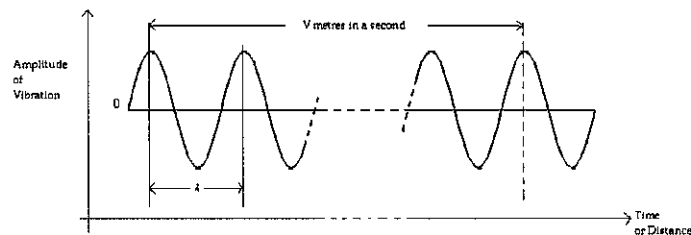


Figure 3: A simple continuous wave

The relationship between the wavelength,  $\lambda$ , velocity,  $v$ , and frequency,  $f$ , of a signal is shown in equation 2.1

$$\lambda = \frac{V}{f} \quad (2.1)$$

This equation is a simple physics equation and it is used as a fundamental concept for this project.

### 2.3 SOLID AND LIQUID CHARACTERISTICS

The solid is considered to be polyethylene pallets from the process area. This pallet's size is 0.4 cm in diameter. The pallets are white in color. It may also contain a small amount of ethylene concentration during storage in a vessel. The liquid is considered as a less concentration of ethylene gas and other chemical gases (less than 1 mol). This is because; the liquid vapor and temperature characteristics can influence the result.

## 2.4 OPERATION OF ULTRASONIC SENSOR

An ultrasonic either generates ultrasound from another form of energy is a transmitter or converts ultrasound into another form of energy is a receiver. Compressional ultrasound can be produced and detected using many techniques. Some of the detectors' technique can be very simple and some of them are quite complex. Therefore, depend on these techniques, it could be used for various type of application especially for the level and flow measurement. The techniques are *monostatic* and *bistatic* mode. These two techniques will be discussed later in the next section.

### 2.4.1 The Transmitter and Receiver

The term of *monostatic* and *bistatic* are sometimes used in relation to transducers. If the transducer is being used only as a transmitter or only as receiver it is known as monostatic. The term of bistatic is used for a transducer that is used for both actions which are transmitter and receiver. However, in this project, a monostatic transducer is modified to be a bistatic transducer in order to meet the plant requirements. These terms can be clearly viewed in Figure 4 [1].



Figure 4: Monostatic and bistatic transducers (a) Monostatic transmitter and monostatic receiver (b) Bistatic transmitter/receiver

Other ultrasonic sensors have separate transducers. This might be very costly but it has several advantages; for example it becomes easier to test and characterize the transducer before installation and for maintenance purposes or replacement during the services. However, it depends on the applications that are requested by the users. Commonly, the monostatic mode is used in order to measure the flow in a pipeline. This is because, the flow is measure by measuring the voltage different from the transmitter to the receiver. However, for level measurement, the bistatic mode is more applicable.

## 2.4.2 The Basic Features of the Beam Light

The beam profile generated by an ultrasonic transducer is determined by interference (diffraction) phenomena between the Huygens' wavelets. The generation of a beam of compressional ultrasound by a circular plane pistons radiator excited by a signal constant amplitude. This situation is particularly relevant to many piezoelectric transducers and also good for many others [4]. The basic features of the beam profile are illustrated in Figure 5.



Figure 5: Interference effects (a) The basic feature of the beam profile  
(b) Interference effect in the near field.

The simple model shows that the near field changes into the far field at a distance  $N$  from the transducers face given by

$$N = \frac{D^2 - \lambda^2}{4\lambda} = \frac{D^2 f}{4V} \quad (2.2)$$

where;  $N$  is distance of field from the face,  $D$  is diameter (meter) of the face,  $f$  is frequency (Hz),  $V$  is velocity (m/s) and  $\lambda$  is the wavelength (meter) [1].

## 2.4.3 Ultrasonic Sensor's Insertion Loss

The insertion loss is a relative measure of a transducer's energy conversion efficiency. The insertion loss [2] is the decreasing in dB, of the amplitude of the amplitude of the received signal compared with amplitude of the input signal and it is given by

$$InsertionLoss = 20 \log \frac{E_i}{E_r} dB \quad (2.3)$$

where;  $E_i$  is the input voltage of the transducer and  $E_r$  the voltage of the received pulse.

## 2.5 THE OPERATION OF PIC16F877 MICROCONTROLLER

PIC16F877 shown in Figure 6 is a high-performance FLASH microcontroller that provides engineers with the highest design flexibility. In addition to  $8192 \times 14$  words of FLASH program memory, 256 data memory bytes, and 368 bytes of user Random Access Memory (RAM), PIC16F877 also features an *integrated 8-channel 10-bit Analogue-to-Digital converter*. Peripherals include two 8-bit timers, one 16-bit timer, a Watchdog timer, Brown-Out-Reset (BOR), In-Circuit-Serial Programming™, RS-485 type Universal Asynchronous Receiver/Transmitter (UART) for multi-drop data acquisition applications, and I2C™ or SPI™ communications capability for peripheral expansion. Precision timing interfaces are accommodated through two Command Control Processor (CCP) modules and two Pulse Width Modulation (PWM) modules [6].

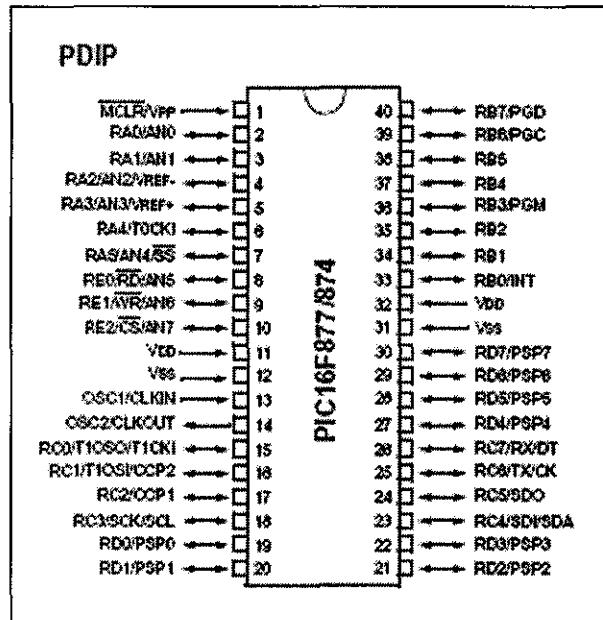


Figure 6: Microcontroller 16F877

## 2.5.1 The Overview on the PIC16F877 Microcontroller File Register

The data memory is partitioned into multiple banks which contain the general purpose registers and the special function registers. Bits RP1 and RP0 are the bank select bits, these bits are found in the STATUS register (b6 & b5).

As shown in Figure 7, each bank extends up to size of 7Fh (128 bytes). The lower locations of each bank are reserved for the *special function registers*. Above the special function registers are *general purpose registers*, implemented as static RAM. All implemented banks contain special function registers. Some “high use” special function registers from one bank may be mirrored in another bank for code reduction and quicker access. Noted that there are 16 *general purpose global registers*, these registers can be accessed from the assignment of the pins [6].

| File Address                      | File Address                      | File Address                           | File Address                           |
|-----------------------------------|-----------------------------------|--|--|
| Indirect addr. <sup>(1)</sup> 00h | Indirect addr. <sup>(1)</sup> 80h | Indirect addr. <sup>(1)</sup> 100h     | Indirect addr. <sup>(1)</sup> 180h     |
| TMRD 01h                          | OPTION_REG 81h                    | TMRD 101h                              | OPTION_REG 181h                        |
| PCL 02h                           | PCL 82h                           | PCL 102h                               | PCL 182h                               |
| STATUS 03h                        | STATUS 83h                        | STATUS 103h                            | STATUS 183h                            |
| FSR 04h                           | FSR 84h                           | FSR 104h                               | FSR 184h                               |
| PORTA 05h                         | TRISA 85h                         |  |  |
| PORTB 06h                         | TRISE 86h                         | PORTE 105h                             | TRISS 185h                             |
| PORTC 07h                         | TRISC 87h                         |  |  |
| PORTD <sup>(1)</sup> 08h          | TRISD <sup>(1)</sup> 88h          |  |  |
| PORTE <sup>(1)</sup> 09h          | TRISE <sup>(1)</sup> 89h          |  |  |
| PCLATH 0Ah                        | PCLATH 8Ah                        | PCLATH 10Ah                            | PCLATH 18Ah                            |
| INTCON 0Bh                        | INTCON 8Bh                        | INTCON 10Bh                            | INTCON 18Bh                            |
| PIR1 0Ch                          | PIE1 8Ch                          | EEDATA 10Ch                            | EEOCN1 18Ch                            |
| PIR2 0Dh                          | PIE2 8Dh                          | EEADR 10Dh                             | EEOCN2 18Dh                            |
| TMR1L 0Eh                         | PCON 8Eh                          | EEDATH 10Eh                            | Reserved <sup>(2)</sup> 18Eh           |
| TMR1H 0Fh                         |                                   | EEADRH 10Fh                            | Reserved <sup>(2)</sup> 18Fh           |
| T1CON 10h                         |                                   |  |  |
| TMR2 11h                          | SSPCON2 90h                       |  |  |
| T2CON 12h                         | PR2 91h                           |  |  |
| SSPBUF 13h                        | SSPAD 92h                         |  |  |
| SSPCON 14h                        | SSPSTAT 93h                       |  |  |
| CCPR1L 15h                        |                                   |  |  |
| CCPR1H 16h                        |                                   |  |  |
| CCP1CON 17h                       |                                   |  |  |
| RCSTA 18h                         | TXSTA 97h                         | General Purpose Register 16 Bytes 117h | General Purpose Register 16 Bytes 197h |
| TXREG 19h                         | SPBRG 98h                         |  |  |
| RCREG 1Ah                         |                                   |  |  |
| CCPR2L 1Bh                        |                                   |  |  |
| CCPR2H 1Ch                        |                                   |  |  |
| CCP2CON 1Dh                       |                                   |  |  |
| ADRESH 1Eh                        | ADRESL 9Eh                        |  |  |
| ADCON0 1Fh                        | ADCON1 9Fh                        |  |  |
|                                   | ADh                               |  |  |
| General Purpose Register 56 Bytes | General Purpose Register 80 Bytes | General Purpose Register 80 Bytes      | General Purpose Register 80 Bytes      |
|                                   | accesses 70h-7Fh                  | accesses 70h-7Fh                       | accesses 70h-7Fh                       |
| Bank 0 7Fh                        | Bank 1 FFh                        | Bank 2 17Fh                            | Bank 3 1FFh                            |

Figure 7: PIC16F877 register map file

## 2.5.2 Overview of 8-Channel to 10-Bit Analog-Digital Converter (ADC)

At first, it appears that the PIC16F877 has 8 built-in ADCs, but this is not the case. Figure 8 shows a simplified block diagram of the analogue-to-digital converter module, clearly there is only one 10-bit ADC which can be connected to only one of eight input pins at any one time.

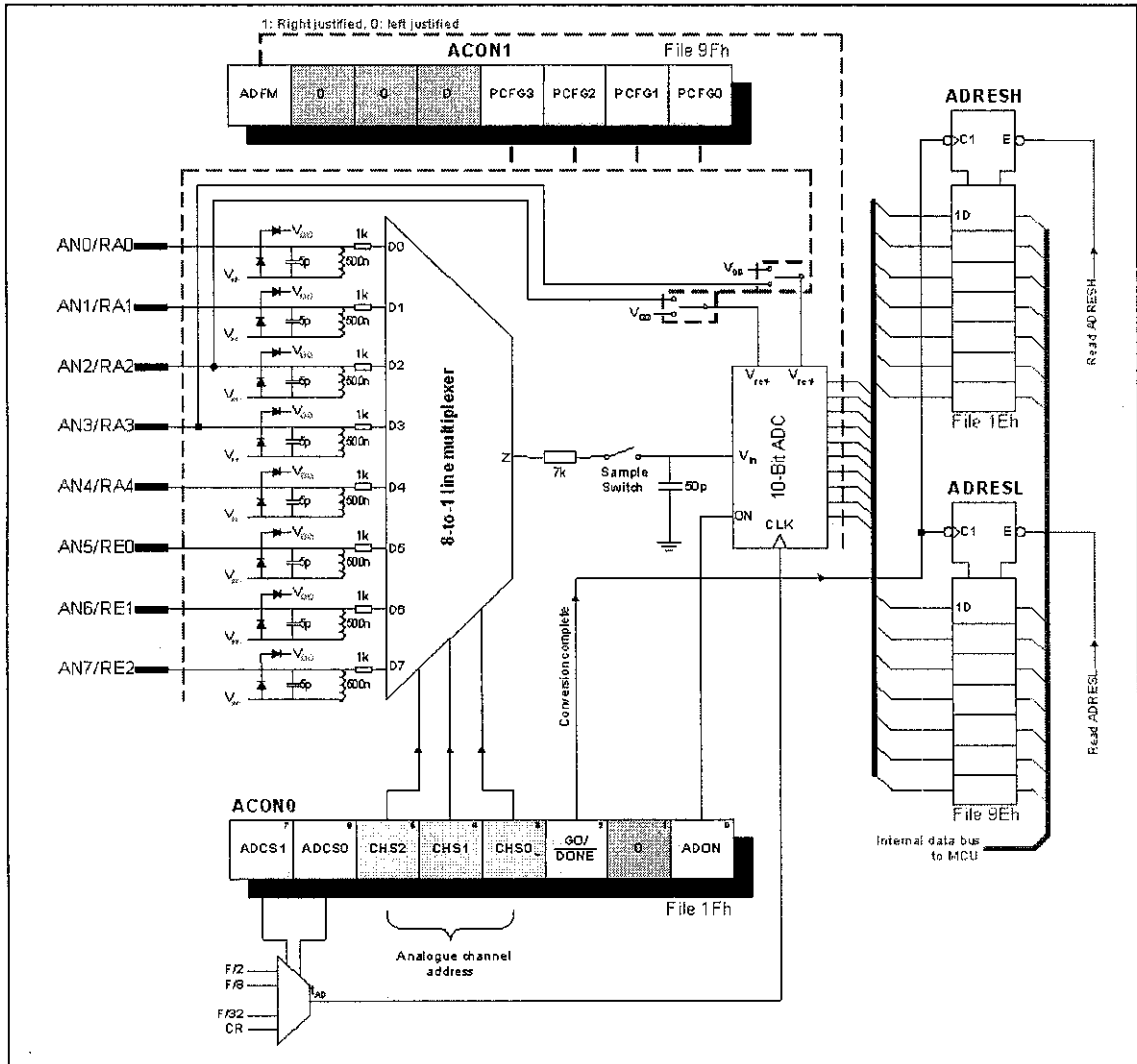


Figure 8: Simplified block diagram of the PIC16F877 ADC module

As shown in Figure 8, the input analogue channels AN4.0 are shared with port A, and channels AN7.5 are shared with port E. If less than eight analogue channels are required, then some of the pins can be assigned as digital I/O port lines using PCFG3.0 bits. For example, if PCFG3.0 = 0010 then AN4.0 are configured as analogue inputs,

while AN7.5 are digital (port E free), with  $V_{DD}$  used as the reference. The ADCON1 register is shown in Figure 9 to configure the functions of the port pins. These port pins can be configured as analog inputs (RA3 can also be the voltage reference), or as digital I/O.

### 2.5.3 ADC Module In PIC16F877 Microcontroller

To set up the PIC16F877's analogue to digital channel, A/D module which has four registers need to be configured which are as follows:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register0 (ADCON0)
- A/D Control Register1 (ADCON1)

The ADCON0 register, shown in Figure 10, controls the operation of the A/D module.

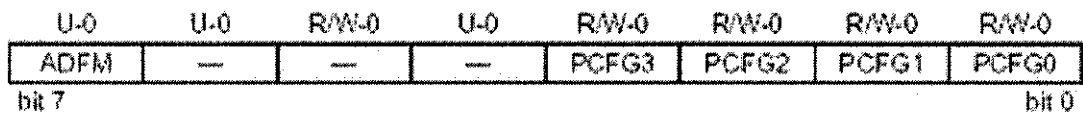


Figure 9: ADCON1 Register

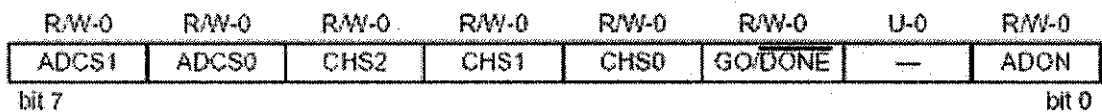


Figure 10: ADCON0 Register

The detail of the ADC module for this microcontroller is given in *Appendix 2*.

## 2.6 THE OPERATION OF PIC16F84A MICROCONTROLLER

The PIC16F84A belongs to the family of a low-cost, high-performance, CMOS, fully-static, 8-bit microcontrollers. All PICmicro™ microcontrollers employ an advanced RISC architecture. PIC16F84A [7] have enhanced core features, eight-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with a separate 8-bit wide data bus. The two stage instruction pipeline allows all instructions to execute in a single cycle, except for program branches (which require two cycles). A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set is used to achieve a very high performance level.

PIC16F84A microcontrollers typically achieve a 2:1 code compression and up to a 4:1 speed improvement (at 20 MHz) over other 8-bit microcontrollers in their class. The PIC16F84A has up to 68 bytes of RAM, 64 bytes of Data EEPROM memory, and 13 I/O pins. A timer/counter is also available.

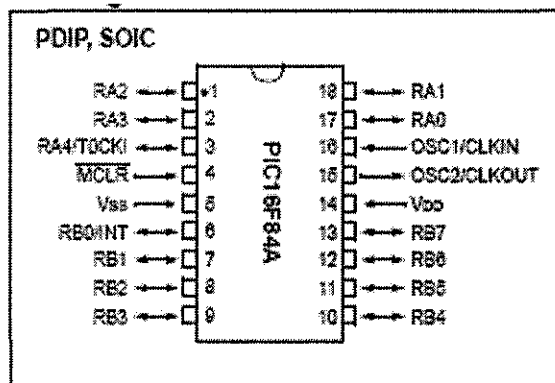


Figure 11: PIC16F84A pin out

There are four oscillator options, of which the single pin Resistor, Capacitor (RC) oscillator provides a low-cost solution, the Low Power (LP) oscillator minimizes power consumption, External (XT) is a standard crystal, and the HS is for High Speed crystals.



## 2.6.1 The Overview on the PIC16F84A Microcontroller File Register

Generally, the high performance of the PIC16F84A can be attributed to a number of architectural features commonly found in RISC microprocessors. The Figure 12 below shows the architecture of the PIC16F84A microcontroller.

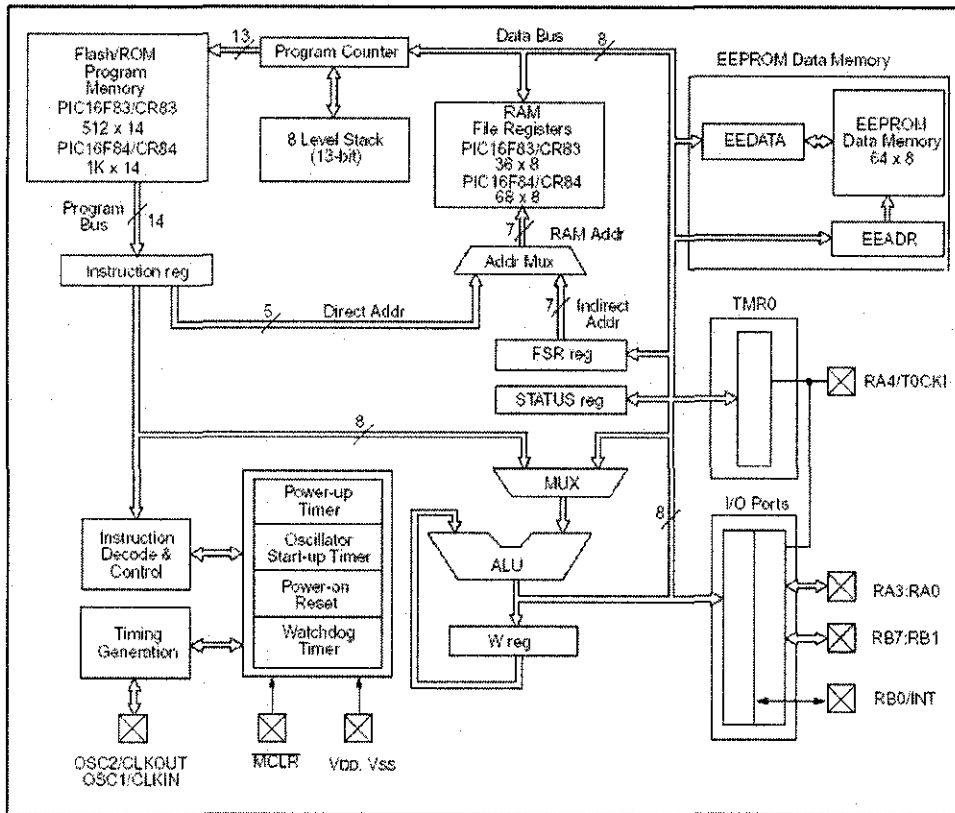


Figure 12: Simplified Block Diagram for PIC16F84A

To begin with, the PIC16F84A uses Harvard architecture. This architecture has the program and data accessed from separate memories. The device has a program memory bus and a data memory bus. This improves the bandwidth over traditional Von Neumann architecture where program and data are fetched from the same memory, which accesses over the same bus. Separating program and data memory further allows instructions to be sized differently than the 8-bit wide data word.

The PIC16F84A can directly or indirectly address its register files or data memory. All special function registers including the program counter are mapped in the data memory. An orthogonal (symmetrical) instruction set makes it possible to carry out any operation on any register using any addressing mode.

### **2.6.2 Common Features of PIC16F84A Microcontroller**

The interesting features in the PIC16F84A are:

- The SLEEP (power-down) mode offers power saving. The user can wake the chip from sleep through several external and internal interrupts and resets.
- A highly reliable Watchdog Timer with its own on-chip RC oscillator provides protection against software lockup.

The devices with Flash program memory allow the same device package to be used for prototyping and production. In-circuit re-programmability allows the code to be updated without the device being removed from the end application. This is useful in the development of many applications where the device may not be easily accessible, but the prototypes may require code updates. This is also useful for remote applications where the code may need to be updated (such as rate information).

The lists of the features of the PIC16F84A are attached to *Appendix C*. A simplified block diagram of the PIC16F84A is shown in Figure 13.

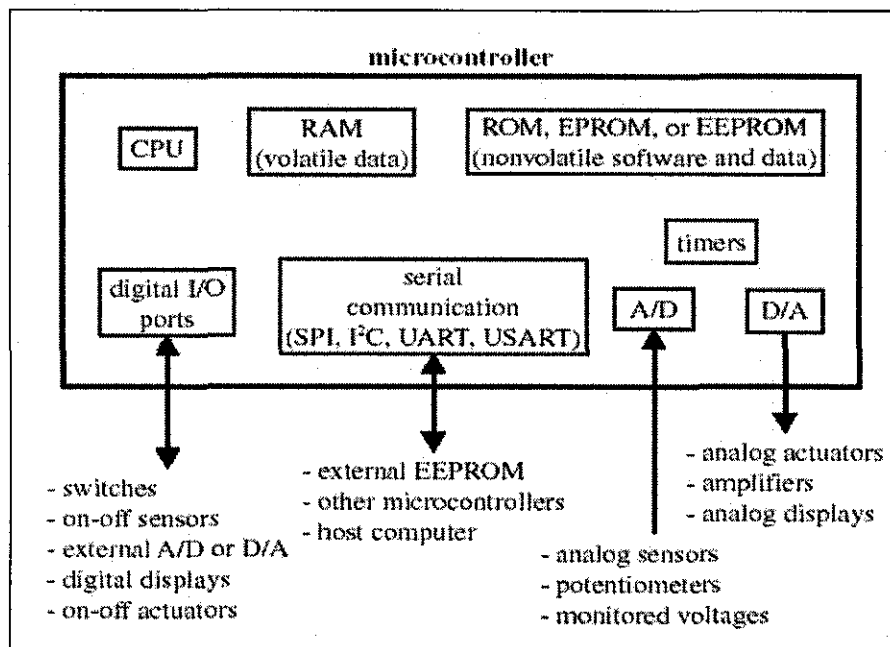


Figure 13: Components of a Microcontroller

The PIC16F84A fits perfectly in applications ranging from high speed automotive and appliance motor control to low-power remote sensors, electronic locks, security devices and smart cards. The Flash or EEPROM technology makes customization of application programs such as transmitter codes, motor speeds, receiver frequencies and security codes extremely fast and convenient. The small footprint packages make this microcontroller series perfect for all applications with space limitations. Low-cost, low-power, high performance, ease-of-use and I/O flexibility make the PIC16F84A very versatile even in areas where no microcontroller use has been considered before (example: timer functions; serial communication; capture, compare and PWM functions; and co-processor applications).

The serial in-system programming feature (via two pins) offers flexibility of customizing the product after complete assembly and testing. This feature can be used to serialize a product, store calibration data, or program the device with the current firmware before shipping.

## CHAPTER 3

### METHODOLOGY AND PROJECT WORK

#### 3.1 HARDWARE

This project requires the design of the system hardware and the improvement on the ULD applications. It consists of a transmitter circuit, the receiver circuit and a PIC microcontroller, LED's and an adapter card to a computer via Visual Basic 6.0.

##### 3.1.1 The Transmitter Circuit

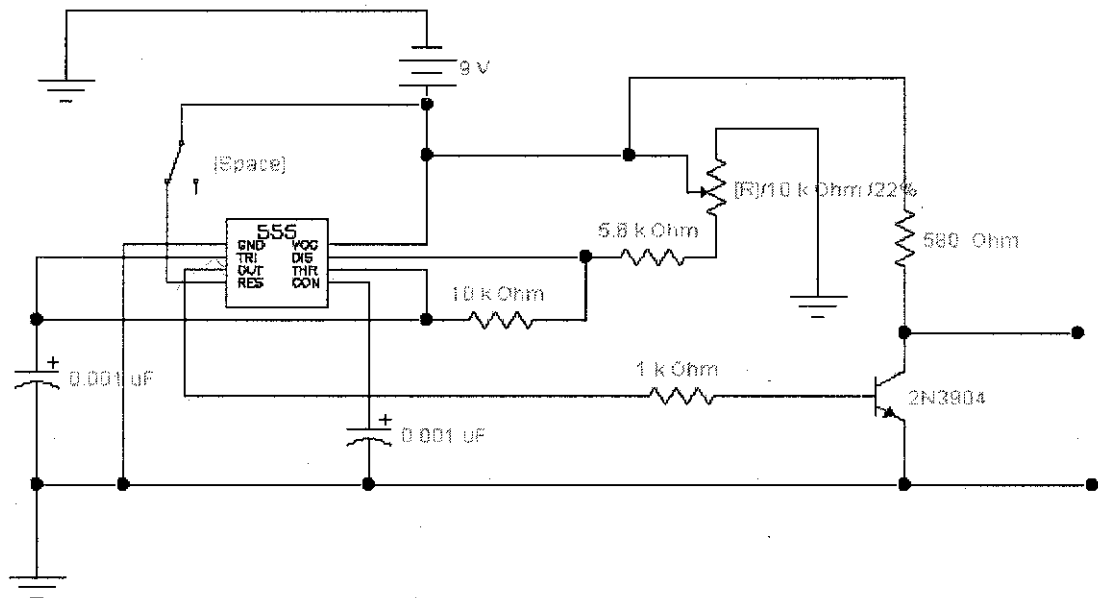


Figure 14: Preliminary ultrasonic transducer transmitter schematic

The preliminary transmitter circuit contains a 555 timer integrated circuit (IC) that can generate 40 kHz square-wave signal. This signal is used to 'latch' the transistor which receives +9V supply to drive the ultrasonic transducer that is able to work on a 40 kHz

frequency. The signal is then amplified using a 2N3904 transistor. The amplified signal will be transmitted into the container through the ultrasonic transducer.

### 3.1.2 The Receiver Circuit

The input part as shown in Figure 15 is connected to the ultrasonic receiver transducer. It receives the sinusoidal signal from the receiver. This signal will be re-amplified using an LM741 Op-Amp. Besides that, this part also acts as a filter to make sure that only a 40 kHz signal wave is received. The amplified signal is converted to DC voltage by a half-wave rectifier circuit. The DC signal is amplified by using LM741 Op-Amp. Then, this signal is sent to the PIC for voltages-distances comparison.

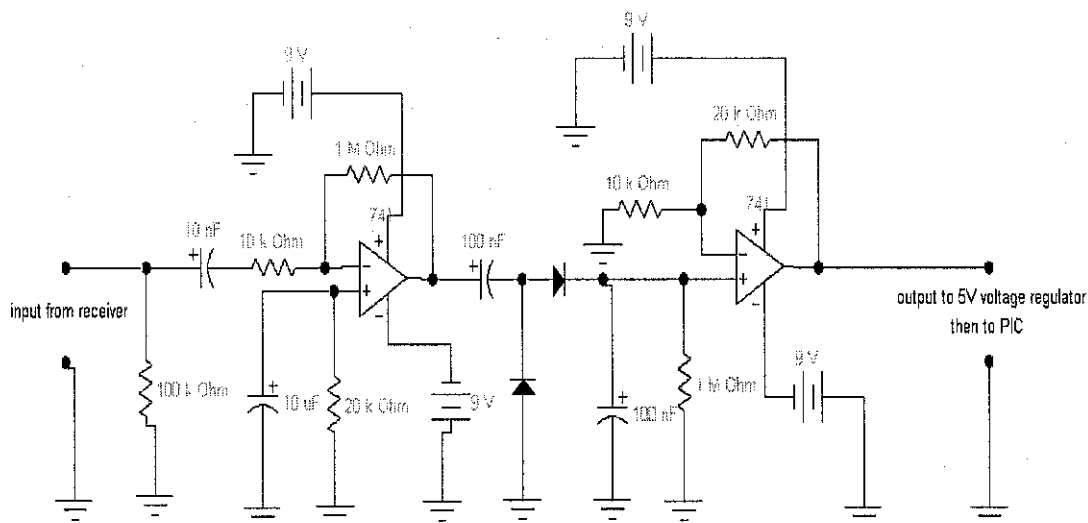


Figure 15: Ultrasonic receiver schematic diagram

### 3.1.3 PIC16F877 Microcontroller Circuit

The microcontroller is connected with other external components such as crystal oscillator, capacitors, resistors and LED's. The purpose of this connection is to make sure that the microcontroller can work properly according to the requirement. The LED purpose is to display the level status as shown in Table 4.3.

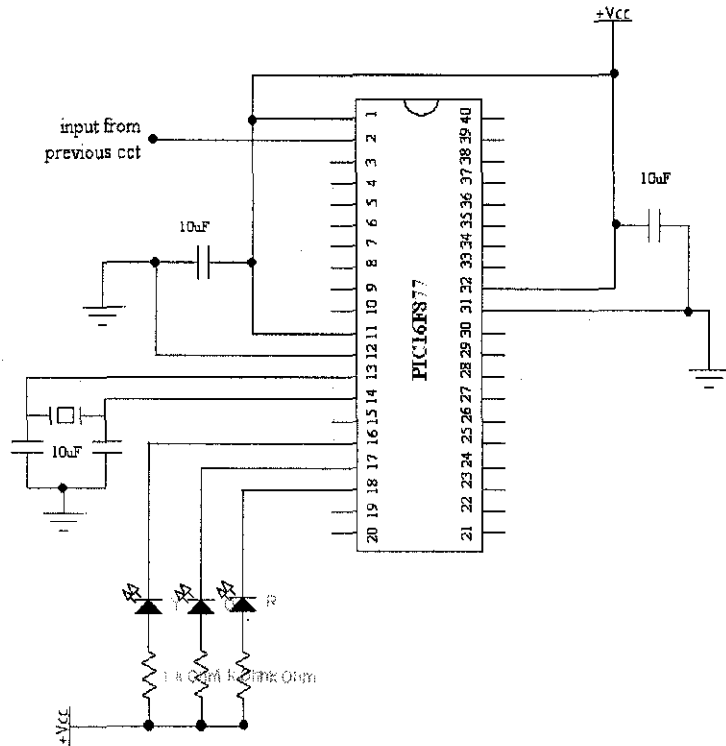


Figure 16: Microcontroller 16F877 circuitry

After completing the connection, the microcontroller needs to be programmed by using assembly language. The programmed codes are downloaded via MPLab software by using WARP13 external device.

### 3.1.4 The Personal Computer (PC) Serial Communicator Circuit

This part actually consists of two major parts which are PIC16F84A circuit and MAX232 circuit.

#### 3.1.4.1 PIC16F84A Microcontroller Circuitry

This circuit was built in order to establish the communication between the instrument and the workstation (computer). The microcontroller was programmed to read the level from the ULD instrument. A complete programming language used for this circuit is given in *Appendix D*. The 'RX' and 'TX' were connected to the next circuit which consists of MAX232 integrated circuit. Therefore, the communication could be established.

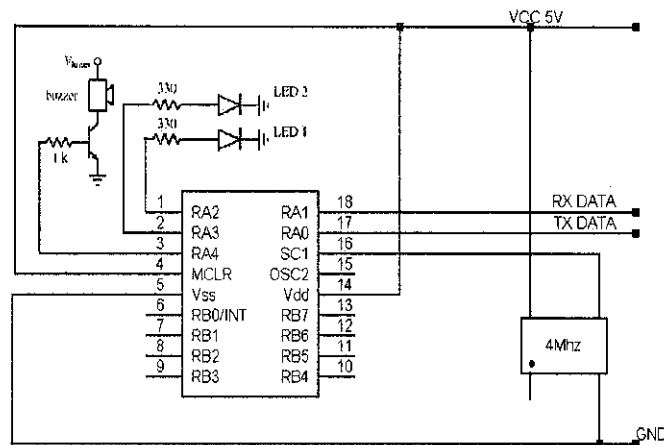


Figure 17: Microcontroller PIC16F84A circuitry.

#### 3.1.4.2 MAX232 Circuitry

MAX232 is an integrated circuit which controls the communication of the serial communication. It either can be as a driver or receiver to the serial port of the computer. In this project, it was used to be a driver to the serial port. The easiest way to get these values is to use the MAX232. The MAX232 acts as a buffer driver for the processor. It accepts the standard digital logic values of 0 and 5 volts and converts them to the RS232 standard of +10 and -10 volts. It also helps protect the processor from possible

damage from static that may come from people handling the serial port connectors. Therefore, it is needed to establish the communication between the device, processor and RS232 serial port. The connection of the MAX232 circuit is as shown in Figure 18.

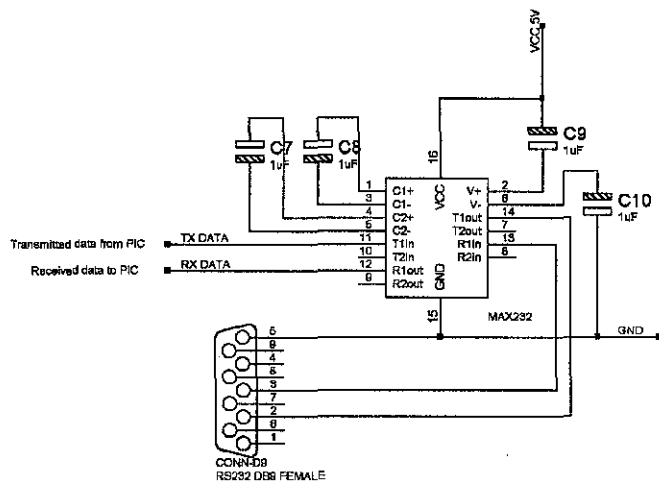


Figure 18: MAX232 circuitry

The 'TX' and 'RX' terminal was connected to the point 'TX' and 'RX' from PIC16F84A microcontroller part. The data will send to workstation via Visual Basic 6.0 which was programmed to display the current status.

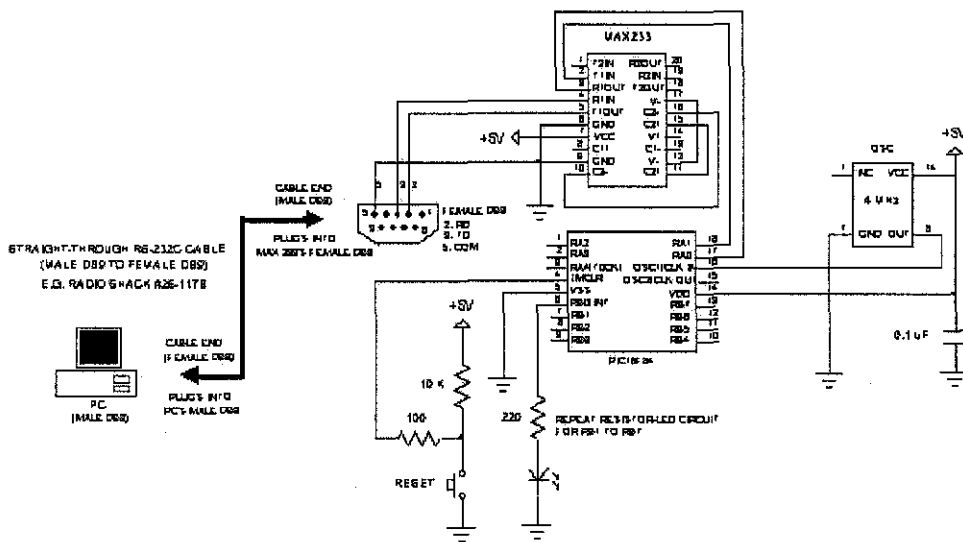


Figure 19: The overall circuit of the serial communicator



### **3.1.5 The Other Additional Features for the Instrument**

There are some additional features that can be added to this ULD. Currently, the added features are as follows:

- Light Emitter Diode (LED)
- Potentiometer

In the future, other possible additional features that would be added are as follows:

- Liquid Crystal Display (LCD)
- Digital Control Devices

## **3.2 SOFTWARE**

The software part consists of assembly language for the microcontrollers and the interface programming. In this project, there are three major types of programming are used namely the assembly language, the C language for microcontrollers and the Visual Basic 6.0 programming.

### **3.2.1 PIC16F877 Microcontroller Assembly Language**

This PIC actually works on a 10-bit digital number. The 10-bit A/D data is loaded onto the register pair ADRESH:ADRESL which is a 16-bit wide register pair. The A/D module gives the programmer a choice whether to left or right justify the 10-bit result into the 16-bit register (more details available in PIC16F877 manual).

The A/D module has a high reference voltage of 5V and a low reference voltage of 0V, therefore, the analog input must be varied within the range of 0 to 5V. An example to convert the corresponding voltage to a hexadecimal number is as shown below:-

Since the maximum voltage allowed is 5V the corresponding 10-bit binary number would:-

11 1111 1111<sub>2</sub> is equal to 1024<sub>10</sub>.

This is the base ration for the next voltage value.

e.g.: (a) Input voltage is 3.75 V.

$$3.75 \text{ V} \times \frac{1024_{10}}{5\text{V}} = 768_{10}$$

Therefore, 768<sub>10</sub> is equal to 11 0000 0000<sub>2</sub>

Examples of another range of value are shown on the Table 3.1. This value can be used for PIC programming.

Table 3.1: Range of the input voltages in binary and hexadecimal.

| Voltage (V) | Binary<br>(ADRESH:ADRESL) | Hexadecimal | Output       |
|-------------|---------------------------|-------------|--------------|
| 5.0         | 0000 0011 1111 1111       | 03FF        | High range   |
| 3.75        | 0000 0011 0000 0000       | 0300        | (red)        |
| 3.74        | 0000 0010 1111 1111       | 02FF        | Medium range |
| 1.26        | 0000 0010 0000 0010       | 0200        | (green)      |
| 1.25        | 0000 0001 1111 1111       | 01FF        | Low range    |
| 0.0         | 0000 0000 0000 0000       | 0000        | (yellow)     |

The flow chart of this sequence is given in Figure 22.

**3.2.2 PIC16F84A Microcontroller C programming**

This microcontroller is used as a part of serial communication interface. It is used to determine the status of the condition since the data was arrived into 16F84A. The programming was designed to select a condition since the state of either PIN B1, B2 or B3 are in HIGH state. The C programming that is used for this project is available in *Appendix D*. Figure 20 shows the flow chart of that C programming.

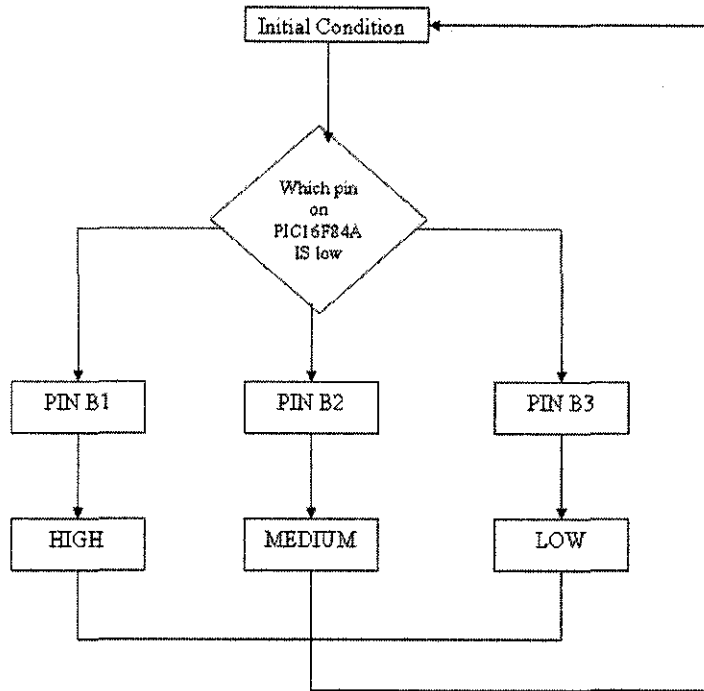


Figure 20: The flow chart of function for C programming for PIC16F84A

Then, this data will be sent to the computer via the serial port by using Visual Basic 6.0. The communication was done by applying MSComm application. This feature is already available in the Visual Basic 6.0.

### 3.2.3 Visual Basic 6.0 Interface Programming

The program that is used in the ULD needed to be installed in the microcontroller before it is ready to be used. The Microsoft Visual Basic (VB) is known to be among the most popular choice to create Windows Graphical User Interface (GUI). In Visual Basic, new windows created are called forms. Elements (such as text boxes and buttons) that are placed inside a form are called controls. The Visual Basic allows event-driven programming, where the user's actions cause events, and each event in turn triggers a procedure that is associated with it. By applying the features in VB, the communication between the serial port of the computer and the device can be established. This VB

interface will display the data that is sent by the serial communicator is shown in Figure 21.

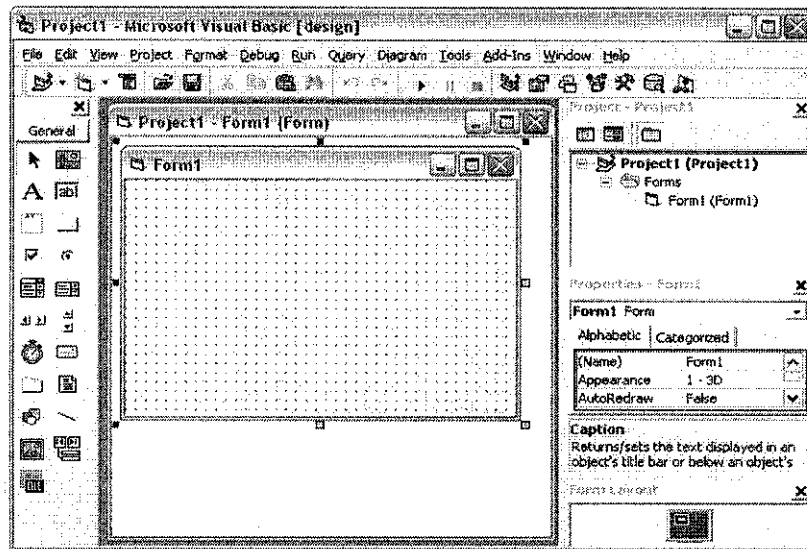


Figure 21: VB window interface

A component known as the Microsoft Comm Control 6.0 is applied to allow the connection between Visual Basic and the serial port. Some of the properties of MSComm that used are listed in Table 3.2.

Table 3.2: Microsoft Comm Control 6.0 Properties

| Properties | Description   |
|------------|---|
| CommPort   | Sets and returns the communications port number   |
| Settings   | Sets and returns the baud rate, parity, data bits and stop bits as a string                     |
| PortOpen   | Sets and returns the state of a communication port. Also opens and closes a port                |
| Input      | Returns and remove character from the receive buffer  |
| Output     | Writes a string of characters to the transmit buffer  |
| InputLen   | Sets the maximum number of characters that will be returned when the input property is accessed |

### 3.3 EXPERIMENT AND DEVELOPMENT

This part is very useful in order to get the result. Therefore, some improvement and development can be done base on this experimental result.

#### 3.3.1 The Transmitter Circuit Construction

First of all, the scratch of the design was constructed in Engineering Workbench (EWB). This software helped the author to construct and simulate the circuit before constructing it on a board. The circuit used is shown in Figure 14. When the circuit is completed, it needs to be tested to ensure it is working as required. The comparison between the simulation waveform using EWB and the waveform obtained from the oscilloscope. Therefore, the graph of 40 kHz square wave is as shown in Figure 22.

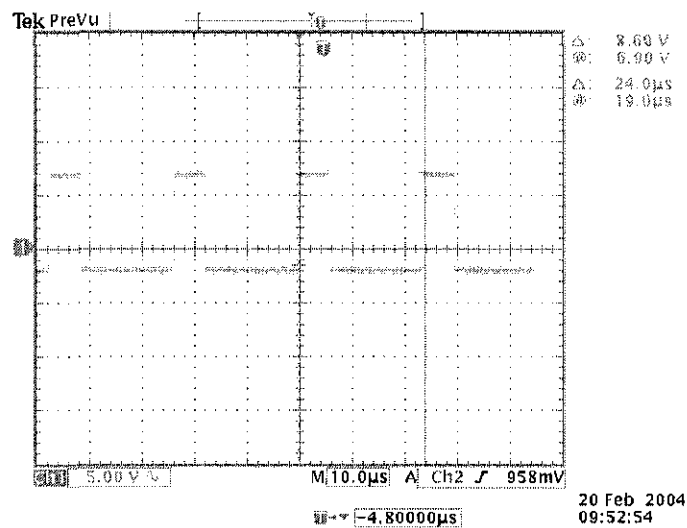


Figure 22: A 40 kHz square-wave signal from the signal generator via oscilloscope.

This figure shows the 555 timer circuit is able to generate a 40 kHz square wave signal accurately. However, there is a potentiometer to control the frequency of the signal depends on the type of sensor and material that are dealing with. This is the point where the adjustment is done according to the vessel conditions.

### 3.3.2 The Receiver Circuit Construction

A receiver circuit is very important part of the whole project. This is because, since it was exposed to the environment, it needs to identify the signal that it will receive. The circuit is constructed part-by-part. Therefore, the experiment is done in part by part too. The result obtained from simulation is compared with the waveform obtained from the oscilloscope. The first part of the circuit is a *filter* [8]. The filter is needed in order to identify the frequency that it should receive in this work is 40 kHz signal. The frequency of the filter is determined by the equation 3.1.

$$f = \frac{1}{2\pi C_1 R_1} \quad (3.1)$$

The circuit configuration for the filter is shown in Figure 23.

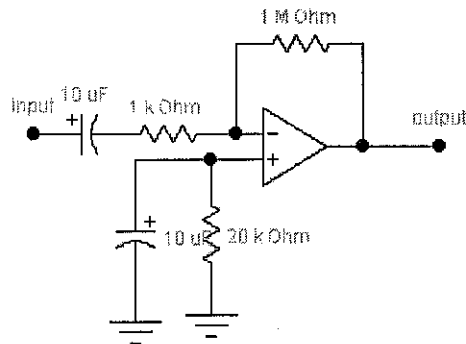


Figure 23: A filter of the receiver circuit.

This circuit is tested with a sinusoidal waveform signal at frequency of 40 kHz. The output of the filter measured is in Figure 24.

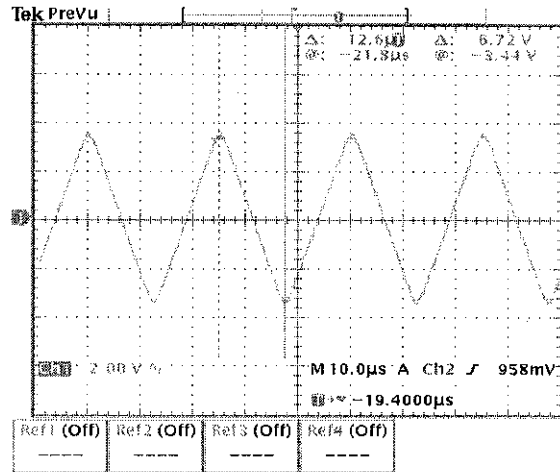


Figure 24: The output from the filter for a source of 1 Vp-p, 40 kHz

The next part is the rectifier circuit. The rectification is required to convert an AC voltage to DC voltage. Therefore, for this small scale model, a simple half-wave rectifier is used. The circuit is as shown in Figure 25.

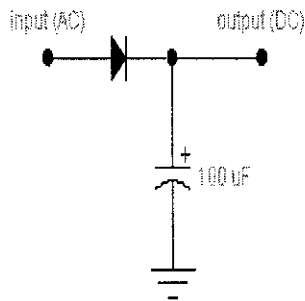


Figure 25: A half-wave rectifier circuit with a capacitance effect

The AC voltage is connected to the input and then the diode will do the rectification. The result of the rectification is as shown in Figure 25. The capacitor is used to smooth the half wave out part to DC signal as shown on Figure 26.

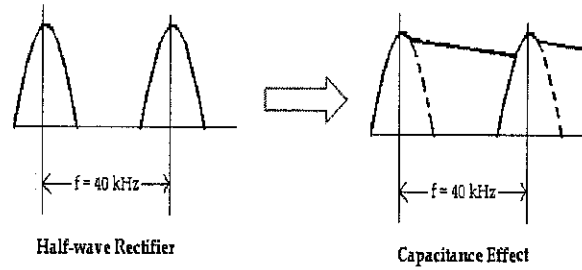


Figure 26: Half-wave rectifier to DC signal using a capacitance

The signal shown in Figure 27 will enter the third part of the circuit which is an amplifier. The amplification is carried out using LM741 Op-Amp. The circuit configuration is to non-inverting input. This is because, the signal just needs to be amplified and not to be inverted. The circuit configuration for this amplifier is as shown in Figure 27.

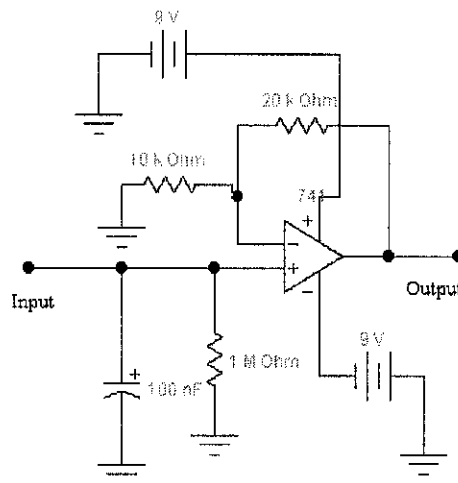


Figure 27: The amplifier circuit configuration

As a result, the output voltage from the amplifier is a DC voltage which is proportional to the input voltage. This voltage will be regulated to a 5 VDC. The voltage is used as an input to the microcontroller and processed based on the voltage different. The waveform is as shown in Figure 28.



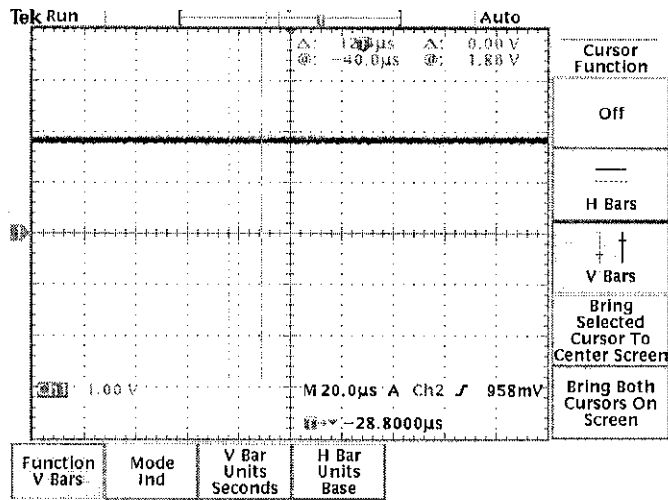


Figure 28: Signal from output of the complete receiver circuit

### 3.3.3 The Microcontroller PIC16F877 Circuit Construction

A miniature processor is needed for an instant result. So, PIC16F87 is used as a processor or voltage ranger. The circuit is simple because the microcontroller itself is already come with the features. The complete circuit of the microcontroller part is shown in Figure 14. The circuit that is assembled on a board is shown in Figure 29. The assembly language for this microcontroller is given in *Appendix F*.

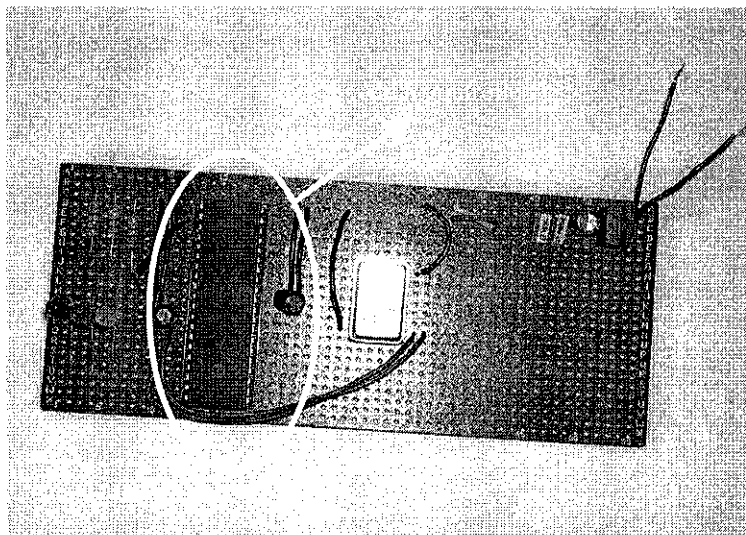


Figure 29: The PIC16F877 microcontroller card.

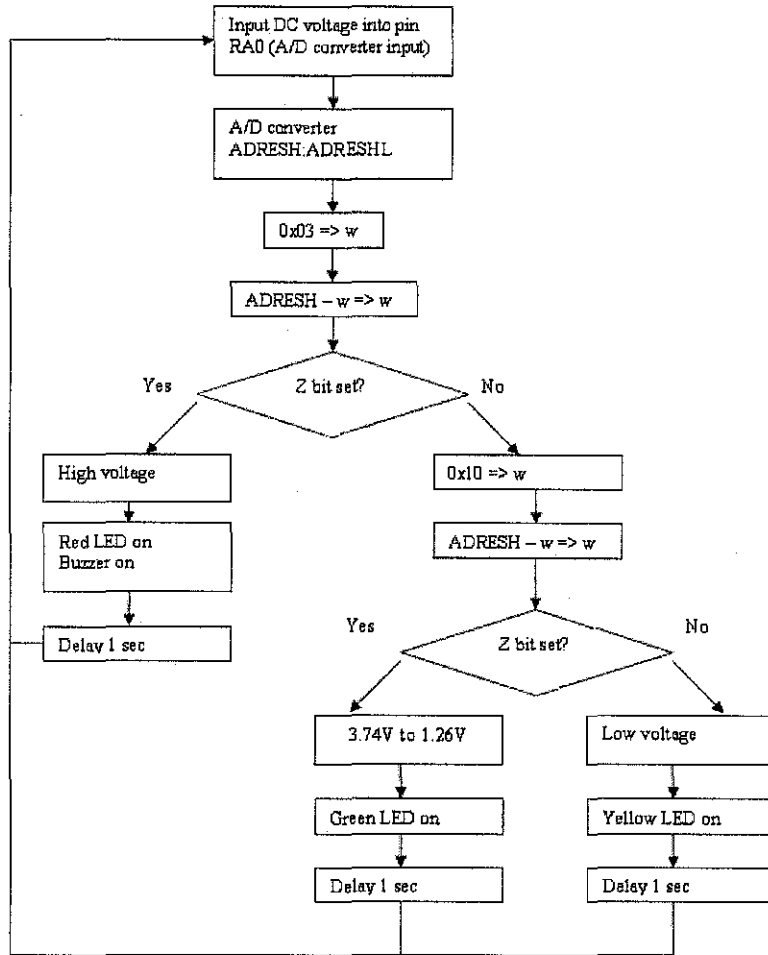


Figure 30: Flow chart of the PIC16F877 microcontroller function.

The circuit in Figure 29 works under the flow chart in Figure 30. The input which is the DC voltage is detected by PINA0. Then, the analog-to-digital converter will convert it to the binary value according to the values in Table 3.1. After that, the microcontroller does some comparison with a pattern value which is 0x03 or in binary is 0000 0011<sub>2</sub>. If the value after the subtraction is bigger than 0x03, the microcontroller decides either to set the Z bit or not. If the decision is 'yes', the microcontroller will light up the RED LED. However, if the decision is 'no', the comparison is done for the second time. Now, the value is 0x10 or in binary is 0001 0000<sub>2</sub>. The microcontroller needs to decide again either to set the Z bit or not. If the decision is 'yes', the microcontroller will light up the GREEN LED to indicate the MEDIUM level. Otherwise, the microcontroller will light up the YELLOW LED to indicate a LOW level.

### 3.3.4 PC Serial Communicator Circuit Construction

The circuit is used to communicate with the serial port of the computer. The cards is consists of 16F84A microcontroller and MAX232 parts. This serial communicator card is used to communicate with the Visual Basic 6.0. The complete circuit is constructed using the circuit in Figure 3.4 and Figure 3.5. However, the microcontroller needs to be programmed in order to act as required. The final proposed circuit is as shown in Figure 31.

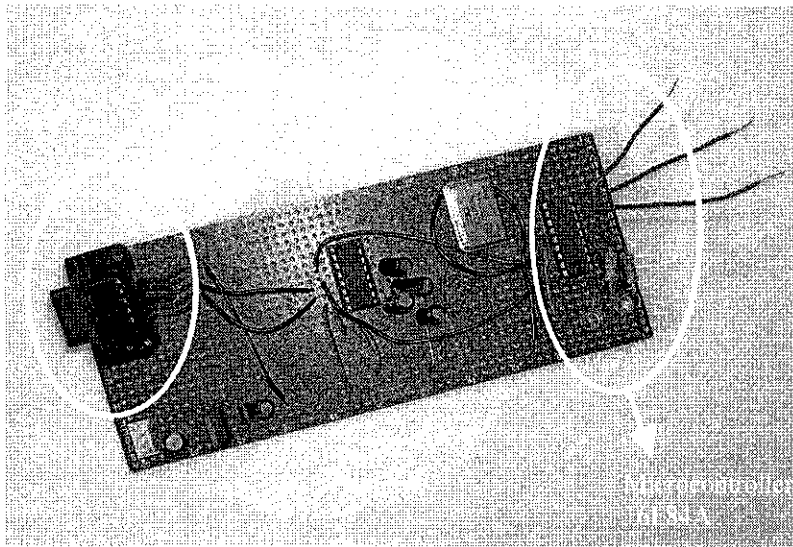


Figure 31: The Serial Communicator card.

The circuit works under the flow chart in Figure 32. The inputs from PIC16F877's pins are connected to the PIC16F84A's pins. Then, according to the flow chart in Figure 32, the level decision is sent to MAX232 and then to computer's serial port. Lastly, the display is on the VB application.

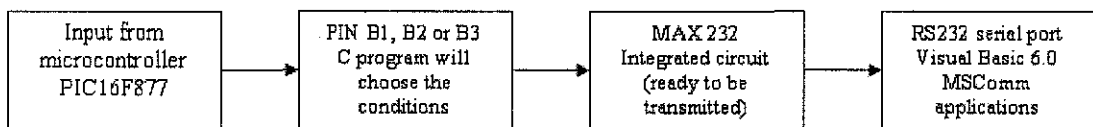


Figure 32: The serial communicator's circuit work's flow

### 3.3.5 Power Supply to the Circuit

In the early stage of this project, the power supply for the LM741 Op-Amp is connected directly to the power supply in the circuit. The first idea is just to connect the IC's directly to the main input (+9V) from the voltage regulator output. However, it was found that the current not enough. Therefore, each voltage regulator LM7809/05 is connected to every Op-Amp voltage supply.

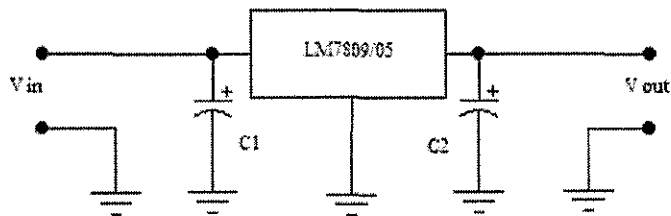


Figure 33: Basic connection for IC voltage regulator

### 3.3.6 Visual Basic 6.0 Serial Interface Development

The design is started with a study on the VB application especially in the MSComm application program. The literature review and other information are obtained from the internet and the books in UTP's library [11].

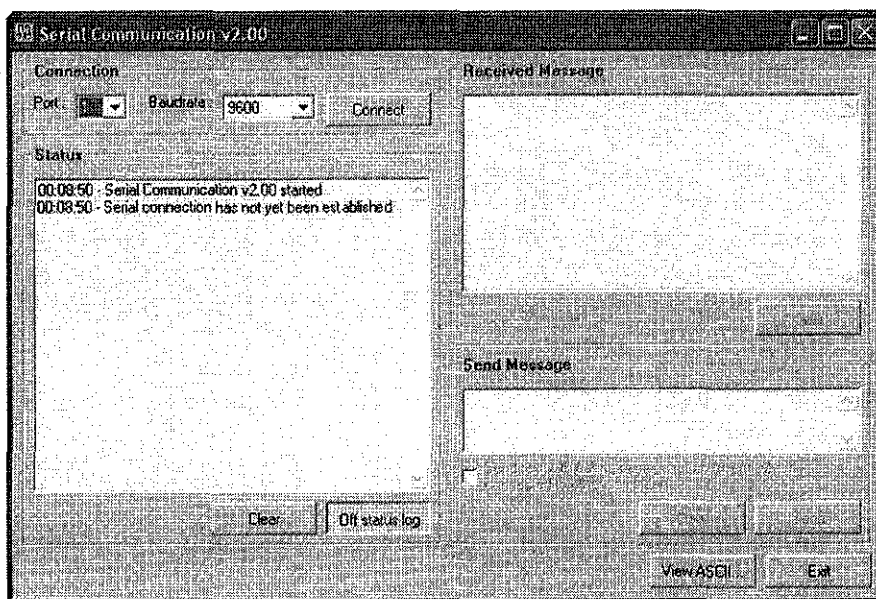


Figure 34: VB MSComm application environment

The serial port is connected by using a serial cable. Since the device was ready, the 'Connect' button was pressed. Then, status will show the current condition as shown in 'Status'. Then, the 'Received Message' box will show the level. The VB programming is inserted in the *Appendix G*.

### **3.4 COMMISSIONING AND VERIFICATION**

After completing all those procedures, the next one is commissioning. During commissioning, the verification and device 'test run' are carried out. These will ensure that the device can work properly and can withstand various operating conditions. The steps that are taken include components assembly, test run, trouble shoot and followed by preparing the user manual.

#### **3.4.1 Components Assembly**

At the early stage of this work, each sub-circuit is built independently. At the end of this project, all these sub-circuits are combined to be the ULD. All the circuits are assembled according to its function. The first sub-circuit is the transmitter and receiver circuit. This is followed by the microcontroller circuit, serial communicator circuit and a cable which is used to connect the serial communicator circuit to a computer or a workstation.

#### **3.4.2 Problem Raised and Troubleshoot**

In a project, the problem cannot be prevented. However, the way of tackling those problems are more important.

An ultrasound transmitter/receiver pair was selected that operated in the 40kHz range. After testing, the theory on susceptibility of ambient conditions was proven true. It is possible that sensors in a different frequency range might prove better, but time and resource constraints forced the use of these sensors.

Then, the Noise to Signal Ratio problems also occurred. After the sensor selection was made and the amplifier circuit constructed on the breadboard a problem with a high amount of noise in the circuit was discovered. The first thing that was modified was the selection of a switching transistor with a narrow switching region. The transistor selected was a NPN-type with a switching region of 0.72 - 0.83 volts. This narrow switching band transistor served to make the circuit less susceptible to minor noise spikes. After the implementation of the new transistor, measurements of distances of up to two feet were possible.

Even with the change in the transistor there was still a noise problem. The next step that was taken was to experiment with some RC high pass filter circuits. This did not help enough to make a difference in the circuit. This is probably due to the required placement of isolating capacitors in the LM358 op-amps that skewed the RC ratios. No attempt was made with active filters. A solution was found before resorting to that. It is possible that an active band-pass filter constructed from an op-amp may have alleviated the problem. It is possible that the addition of an active filter could extend the current range of the device.

The final solution that was used was to solder the circuit to a generic copper traced board. This helped to eliminate a large portion of the circuit noise. Once this was completed measurements were possible of distances up to 15 ft.

The other problems also raised and needed to be solved. The summary of the problem and method to solve are included in Table 3.3.

Table3.3: Summarization of the faulty and method taken to overcome

| Problem and faulty   | Method to overcome   |
|--|--|
| Signal lost during the first time connection.              | The connection should <i>be centralized</i> and is taken from one point. The device cannot use a different power supply. So, it makes the electrical system become simpler.  |
| Signal becomes unstable for a long time running operation. | The trouble shoot is done on the <i>signal generator</i> part which is suspected to vary according to the time. This is because, both transmitter and receiver work at 40 kHz. Then, the <i>potentiometer part is being set to a fix value</i> , so that it can give the best performance. |

## CHAPTER 4

### RESULT AND DISCUSSION

#### 4.1 RESULTS

During the project commissioning, some experiments were carried out to get the experimental result. According to the requirement, the experiment is performed on the distance's variation versus the voltage output at the receiver circuit. The next one is on microcontroller and followed by Visual Basic 6.0 interface.

##### 4.1.1 Distance Variation of The Transmitter And Receiver Circuits

After completing all the hardware part, the verification of the distance with output voltage is carried out. The ultrasonic transmitter and receiver are connected to the

- |               |  |
|---------------|--|
| Title         | : Voltage (V) versus Distance (cm) .   |
| • Procedure   | : i – The circuit is constructed<br>ii – The transmitter and receiver are placed side-by-side.<br>iii – The distance is marked on 5 cm scale and the measurement was made.<br>iv – The data were taken and recorded it in a table. |
| • Result      | : The data obtained are recorded into a table as shown in Table 4.1.   |
| • Observation | : This signal generator is able to drive the transmitter and this system can work for more than 0.5 meter.   |



The result that obtained from the above experiment is given in Table 4.1.

Table 4.1: Mean voltage versus distance measured

| Distance (cm) | Voltage Mean (V) |
|---------------|------------------|
| 5             | 8.42             |
| 10            | 8.41             |
| 15            | 8.39             |
| 20            | 8.28             |
| 25            | 8.11             |
| 30            | 7.10             |
| 35            | 5.60             |
| 40            | 5.30             |
| 45            | 5.00             |
| 50            | 5.00             |
| 55            | 4.85             |

The values in Table 4.1 are used to plot a graph as shown in Figure 35. The graph shows that the voltage changes rapidly within the range of 20 cm to 35 cm of the ULD and the product's surface or level.

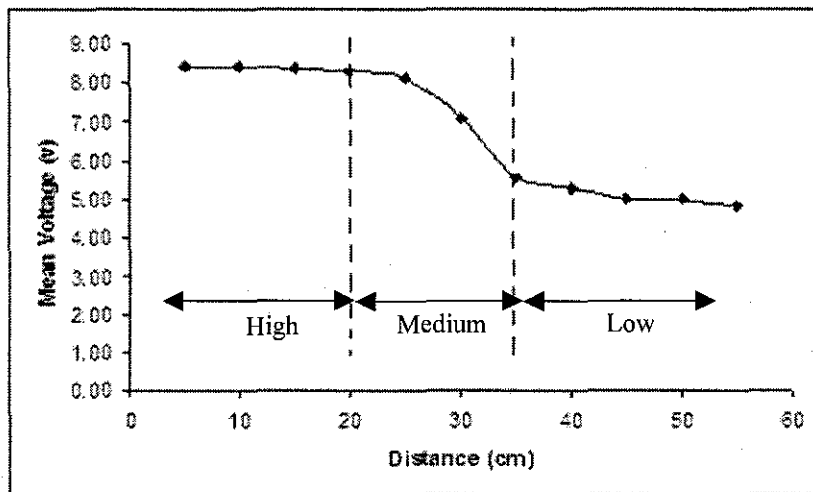


Figure 35: Graph of voltage versus distance before regulate by 5V

### 4.1.2 Microcontroller PIC16F877 Clock Set Up

The prototype of the ULD is assembled on a bread-board, it is found that the LED and the display have a delay. In *Appendix 6*; PIC16F877 assembly language, the delay is provided in order to allow the microcontroller to process the data. The delay is defined by the designer. Table 4.2 shows the calculation used to obtain the number of instruction cycles needed to create a 1 second delay.

|  |  |
|--|--|
| * Clock frequency                                  | = 4 Mhz (frequency of crystal oscillator used) |
| * Working frequency                                | = 1 Mhz (4Mhz/4 quadrant)                      |
| * Period of a working cycle                        | = 1/1Mhz = 1µsecond                            |
| * Number of cycles needed to create 1 second delay | = 1 second / Period of a working cycle         |
|  | = 1 second / 1 microsecond                     |
|  | = 1,000,000 cycles                             |

Table 4.2: The instruction set and assignment of the variables.

| Instruction    | Total number of cycles |
|----------------|------------------------|
| call           | 2                      |
| movlw 0x64     | 1                      |
| movwf DCOUNT1  | 1                      |
| movlw 0x64     | 1*B                    |
| movwf DCOUNT2  | 1*B                    |
| movlw 0x20     | 1*C                    |
| movwf DCOUNT3  | 1*C                    |
| decfsz DCOUNT3 | 1*C*B*A                |
| goto \$ - 1    | 2*C*B*A                |
| decfsz DCOUNT2 | 1*B*A                  |
| goto \$ - 5    | 2*B*A                  |
| decfsz DCOUNT1 | 1*A                    |
| goto \$ - 9    | 2*A                    |
| return         | 2                      |

Thus, the total number of cycles to execute *ALL* instructions in delay subroutine is:

$$\text{Total Cycle} = 6 + 2B + 2C + 3ABC + 3AB + 3A$$

A = 0xf, B = 0xff & C = 0x57 were chosen, which will give a total of 1,010,535 cycles. Since each cycle corresponds to 1  $\mu$ s, the total delay that this subroutine provides is 1,010,535  $\mu$ s, which is approximately 1 second.

This delay must be inserted into the assembly language. Therefore, the delay needed can be defined by the designers according to the project requirements.

#### 4.1.3 PC Serial Communicator with Visual Basic 6.0

As far as interfacing between the PIC16F84A and PC is concerned, the initialization is the first considered. This carried out by modifying the settings on both ends to accommodate each other. The settings required are:

- Baud rate: 9600
- Parity bit: None
- Data bits: 8
- Stop bits: 1

These initializations will coincide with the PIC16F84A. The two pins from PIC16F84A that interacts with the serial port are:

- Transmit pin (A0) – TX
- Receive pin (A1) – RX

Then, a test is performed on the programming to check the function of the serial communicator. The procedure is started by applying 5VDC to every pin; pin B1, B2 and B3. The result of the test is shown in the Table 4.3.

Table 4.3: Summary of the level display on VB when the assigned pin on PIC16F84 is activated

| Pin No. | Word Transmitted | VB Display |
|---------|------------------|------------|
| B1      | HIGH             | HIGH       |
| B2      | MEDIUM           | MEDIUM     |
| B3      | LOW              | LOW        |

The test is also carried out on a liquid vessel that contained of water in UTP pilot plant. The experiment was done by varying the location of the instrument on the top of the vessel. The vessel length is 150 cm and the diameter is 70 cm. The result of the experiment is as shown in Table 4.4.

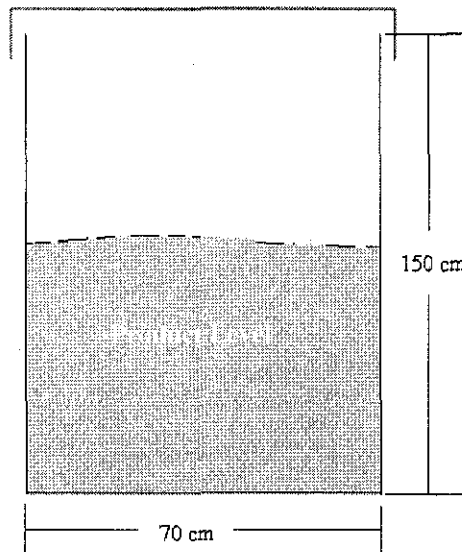


Figure 36: The vessel's dimension in centimeter

Table 4.4: Result from the experiment on pilot plant vessel distance between the sensor and product level.

| Distance (cm) | Voltage (V) |
|---------------|-------------|
| 0             | 2.01        |
| 15            | 2.57        |
| 30            | 3.13        |
| 45            | 3.45        |
| 60            | 4.75        |
| 75            | 5.00        |
| 90            | 5.30        |
| 105           | 5.60        |
| 120           | 7.11        |
| 135           | 8.40        |
| 150           | 8.43        |

The values in Table 4.4 are used to plot a graph as shown in Figure 36. The graph shows that the voltage changes unstable according to increment in the product level. This is because of the additional factors like the material of the vessel and ambient condition in the vessel.

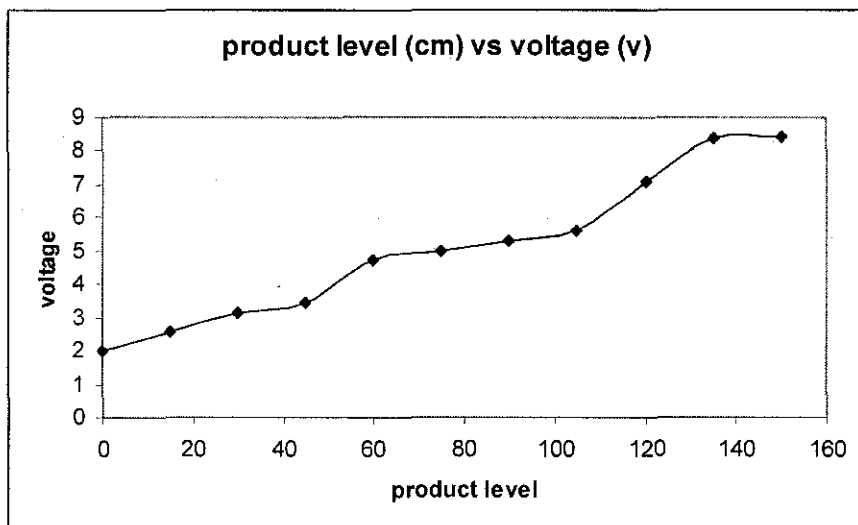


Figure 37: The plotted graph of product level versus the voltage.

## **4.2 DISCUSSION**

According to the result in Table 4.1 and Figure 35, the device can indicate the status of HIGH, MEDIUM and LOW as required. However, the status that it was indicated still can be improved. This is because in Figure 35, the status for HIGH and LOW levels' slope were not too steep. Since that, the levels in these regions are hard to be defined according to the voltage different. However, the medium region can be easily determined because the slope in this region was already steep enough.

### **4.2.1 Real World Application and Small Scale Model**

As stated in the title, the application is on the solid and liquid level detection. According to Table 4.4, the instrument or device is tested on the liquid (water) vessel in UTP pilot plant. The problem raised is the suitable location to mount the device and the echo signal-to-noise ratio received by the receiver.

Although the device can work at the UTP pilot plant, the device still needs to be improved in order to fulfill the plant requirement. A small scale model and concept was produced from this project and the manufacturer can use the user friendly and simple concept to be implemented in the industries.

However, for an industrial scale the changes are on the type of sensor and circuit assembly. In this project, due to the cost constraint, a simple ultrasonic sensor was used. Therefore, the sensor could be replaced by a Polaroid ultrasonic sensor or a pulse ultrasonic transmitter.

## **CHAPTER 5**

### **CONCLUSION**

#### **5.1 CONCLUSION**

The preliminary goal of this project is to design and test the concept of an Ultrasonic Level Detector (ULD) and it was achieved. In the implementation of the PIC16F877 microprocessor is used and the first generation of a ULD can be justified. According to the experimentation that is carried out, the ultrasonic sensor can detect for a distance of more than 50 centimeter (cm). This specification is acceptable for a mini ULD. Then, after this justification, the focus is more on achieving a secondary goal which are to establish a communication between the instrument and a computer for monitoring purpose. Anyway, the constraint on the budget and time must be considered.

According to the results of experiments, obtained that the device can work properly followed the suggested target in the early day in this year. By including other additional features like serial communicator and Visual Basic 6.0 GUI, the system becomes more reliable instead of the transmitter and receiver only.

To put in a nut shell, the target was achieved in this frame of time. Now, it depends on the next inventors to continue this project and improve it for the better features in the future.

## 5.2 RECOMMENDATION

As a recommendation, the transmitter circuit that is used is commonly used to generate a 40 kHz signal by using a 555 timer. However, this project still needs a lot of improvement from the other skill innovators for the better features. Although, the experiments that were carried out shown an acceptable result, but the range is still large. In order to detail the range, more samples will be taken. Then, the range can be simply patterned the data in the PIC microprocessor's program. However, the most important target is to come out with the basic prototype first.

Other recommendation is on the software that established the serial communication. The other student or innovators after this still can improve the system of the software. For example, the other features that should be added is the ULD can be controlled from the workstation instead of just display the data. Therefore, some additional digital components may be needed to be added to fulfill this new requirement.

A documentation of the study must be clearly written and compiled. Therefore, the project will be easier to be developed by the other innovators. As advised by the supervisor, the detail drawing on the dimension of the prototype must be also included.



## REFERENCES

- [1] R .C. Asher, *Ultrasonic Sensors for Chemical and Process Plant*, Institute of Physics Publishing Bristol and Philadelphia, 1997, pp. 14-18, 65-69.
- [2] J. David and N. Cheeke, *Fundamentals and Applications of Ultrasonic Waves*, CRC Press, 2002.
- [3] J. P. Charlesworth and J.A.G. Temple, *Engineering Application of Ultrasonic Time-of-Flight Diffraction – Second Edition*, Research Studies Press Ltd, 2001.
- [4] D. C. Giancoli, *Physics- Fifth Edition*, Prentice Hall, 1998, pp. 362-370.
- [5] Final Year Project Guidelines for Supervisors and Students, Semester January 2004, Universiti Teknologi PETRONAS.
- [6] PIC16F87x Data Sheets 28/40-Pin 8-Bit CMOS FLASH Microcontrollers, Microchips Technology Inc. , 2001
- [7] PIC16F84A Data Sheet 18-pin Enhanced FLASH/EEPROM 8-bit Microcontroller, Microchips Technology Inc. , 2001
- [8] R. L .Boylstead and L. Nashelsky, “*Op-Amp Applications*”, *Electronics Devices and Circuit Theory – Fifth Edition*, Prentice Hall, 2002, pp. 715-719
- [9] R. J. Tocci and N. S. Widmer, “*Number system and Codes*”, *Digital System Principal and Applications – Eight Edition*, Prentice Hall, 2002, pp. 24-33.
- [10] C. D. Johnson, “*Mechanical Sensor – Level control and sensors*”, *Process Control Instrumentation Technology – Seventh Edition*, Prentice Hall, 2003, pp. 220-222.
- [11] D. Zak, *Visual Basic 6.0 Enhanced Edition*, “*An Introduction to Visual Basic – Visual Basic Startup Screen*”, Course Technology, Thomson Learning, pp. 19-28.
- [12] [www.electronics-ee.com/electronics/measuring](http://www.electronics-ee.com/electronics/measuring)
- [13] [www.murata.com](http://www.murata.com)
- [14] [www.ednmag.com](http://www.ednmag.com)

## **APPENDICES**

- APPENDIX A : GANTT CHARTS**
- APPENDIX B : ULTRASONIC SENSOR**
- APPENDIX C : PIC 16F84A DATA SHEET - FEATURES**
- APPENDIX D : C PROGRAMMING (16F84A) - SERIAL  
COMMUNICATION**
- APPENDIX E : 555 TIMER DATA SHEET**
- APPENDIX F : ASSEMBLY LANGUAGE FOR PIC16F877**
- APPENDIX G : VISUAL BASIC 6.0 PROGRAMMING – GUI  
INTERFACE**
- APPENDIX H : PIC16F877 DATA SHEET (ADC CONVERSION)**
- APPENDIX I : MAX232 DATA SHEET**
- APPENDIX J : FLOW CHART OF PIC16F877**





# APPENDIX B

Data Pack E

Issued March 1991 003-065



## Ultrasonic transducers

RS stock numbers 307-351, 307-367

A range of two transducers operating at 40kHz approximately and designed for ultrasonic transmission and reception. The ultrasonic transmitter, 307-351 is capable of emitting 106dB (0dB =  $2 \times 10^{-4}$ µbar) and the receiver 307-367 has a sensitivity of -65dB (0dB = 1µbar√metre).

These units can be used for the transmission of continuous wave ultrasonic sound or for pulsed sound applications

### Characteristics

| Item                                      | Unit             | 307-351     | 307-367     |
|---|------------------|-------------|-------------|
| Transmitting sensitivity                  | Sv               | dB*1        | -           |
| Receiving sensitivity                     | Mv               | dB*2        | -65         |
| Resonant frequency (transmitting)         | F <sub>0T</sub>  | kHz*3       | 40±1        |
| Resonant frequency (receiving)            | F <sub>0R</sub>  | kHz*4       | -           |
| Structural angle                          | °                | -           | 30          |
| Maximum input voltage                     | V <sub>rms</sub> | 20          | -           |
| Impedance                                 | Ω                | Approx. 300 | Approx. 30k |
| Capacitance                               | pF               | 1100±20%    | -           |
| Pulse rise time                           | msec.            | 2.0         | 0.5         |
| Maximum input voltage for pulse operation | V <sub>pp</sub>  | 30          | -           |
| Temperature range                         | °C               | -20 to +60  | -           |
| Transmitting selectivity                  | Cov              | Approx. 20  | -           |
| Receiving selectivity                     | Cov              | -           | Approx. 20  |

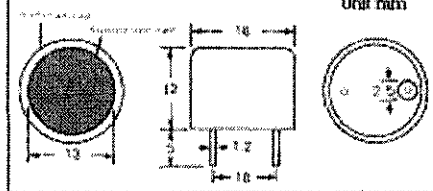
\*1 0dB =  $2 \times 10^{-4}$ µbar

\*2 0dB = 1V/µbar

\*3 Frequency where transmitting sensitivity is maximum

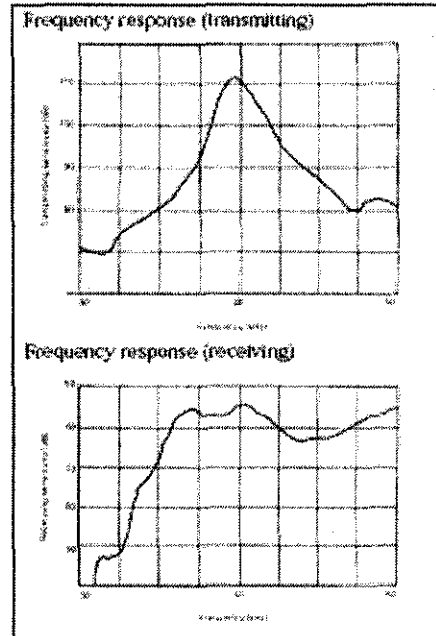
\*4 Frequency where receiving sensitivity is maximum

### Shape and dimensions



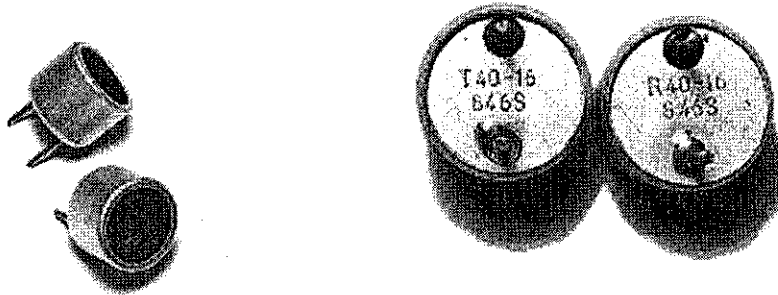
### Applications

- Burglar alarm systems
- Proximity switches
- Liquid level meters
- Anti-collision devices
- Counters for moving objects
- TV remote control systems.



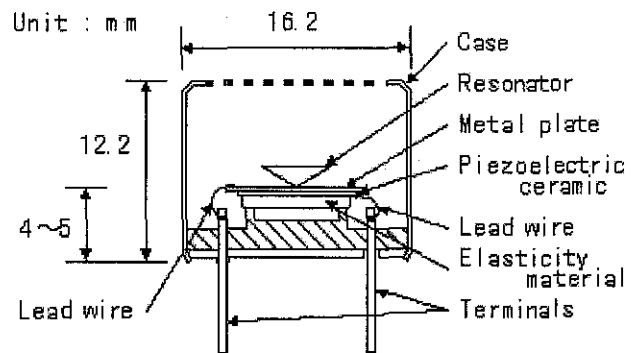
## Ultrasonic sensor

I used the ultrasonic sensor for the air which is made by the Nippon Ceramic company. This sensor separates into the two kinds for the transmitter and the receiver. For the transmitter, it is T40-16 and for the receiver, it is R40-16. T shows the thing for the transmitter and R shows the thing for the receiver. 40 shows the resonant frequency of the ultrasonic.(40kHz) 16 shows the diameter of the sensor. Because the one of the terminal is connected with the case, when grounding, the terminal on the side of the case should be used.



The brief specification of the ultrasonic sensor is shown below.

| Item                      | Spec     |      |
|---------------------------|----------|------|
| Frequency(KHz)            | 40       |      |
| Sound pressure level (dB) | 115 <    |      |
| Sensitivity (dB)          | -64 <    |      |
| Size (mm)                 | Diameter | 16.2 |
|                           | Height   | 12.2 |
|                           | Interval | 10.0 |



As for the detail specification, refer to "[Air Transmission Ultrasonic Sensor](#)".

## APPENDIX C



# PIC16F84A

## 18-pin Enhanced FLASH/EEPROM 8-Bit Microcontroller

### High Performance RISC CPU Features:

- Only 35 single word instructions to learn
- All instructions single-cycle except for program branches which are two-cycle
- Operating speed: DC - 20 MHz clock input  
DC - 200 ns instruction cycle
- 1024 words of program memory
- 68 bytes of Data RAM
- 64 bytes of Data EEPROM
- 14-bit wide instruction words
- 8-bit wide data bytes
- 15 Special Function Hardware registers
- Eight-level deep hardware stack
- Direct, indirect and relative addressing modes
- Four interrupt sources:
  - External RB0/INT pin
  - TMR0 timer overflow
  - PORTB<7:4> interrupt-on-change
  - Data EEPROM write complete

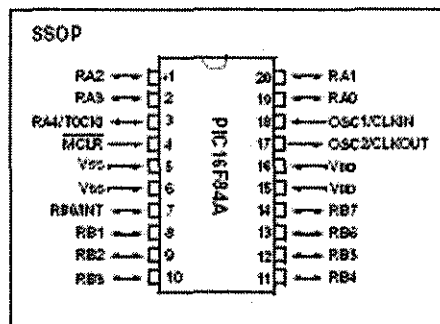
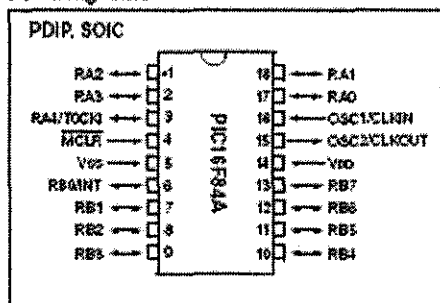
### Peripheral Features:

- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
  - 25 mA sink max. per pin
  - 25 mA source max. per pin
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler

### Special Microcontroller Features:

- 10,000 erase/write cycles Enhanced FLASH Program memory typical
- 10,000,000 typical erase/write cycles EEPROM Data memory typical
- EEPROM Data Retention > 40 years
- In-Circuit Serial Programming™ (ICSP™) - via two pins
- Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own On-Chip RC Oscillator for reliable operation
- Code protection
- Power saving SLEEP mode
- Selectable oscillator options

### Pin Diagrams



### CMOS Enhanced FLASH/EEPROM Technology:

- Low power, high speed technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.0V to 5.5V
  - Industrial: 2.0V to 5.5V
- Low power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 15 µA typical @ 2V, 32 kHz
  - < 0.5 µA typical standby current @ 2V



## APPENDIX D

```
#include <16F84A.H>
#fuses XT,NOPROTECT,NOWDT
#use delay(clock=4000000)

#use rs232(baud=9600,xmit=PIN_A0,rcv=PIN_A1)

void main()
{
    int is_high=0;
    int is_med=0;
    int is_low=0;

    while (true)
    {
        if (!input(PIN_B1))
        {
            if (is_high==0)
            {
                printf("HIGH ");

                is_high=1;
                is_med=0;
                is_low=0;
            }
        }

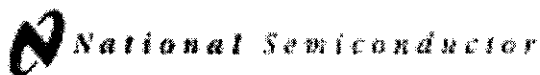
        if (!input(PIN_B2))
        {
            if (is_med==0)
            {
                printf("MEDIUM ");

                is_high=0;
                is_med=1;
                is_low=0;
            }
        }

        if (!input(PIN_B3))
        {
            if (is_low==0)
            {
                printf("LOW ");

                is_high=0;
                is_med=0;
                is_low=1;
            }
        }
    }
}
```

## APPENDIX E



February 2000

LM555 Timer

### LM555 Timer

#### General Description

The LM555 is a highly stable device for generating accurate time delays or oscillation. Additional terminals are provided for triggering or resetting if desired. In the time-delay mode of operation, the time is precisely controlled by one external resistor and capacitor. For astable operation as an oscillator, the free-running frequency and duty cycle are accurately controlled with two external resistors and one capacitor. The circuit may be triggered and reset on falling waveforms, and the output circuit can source or sink up to 200mA or drive TTL circuits.

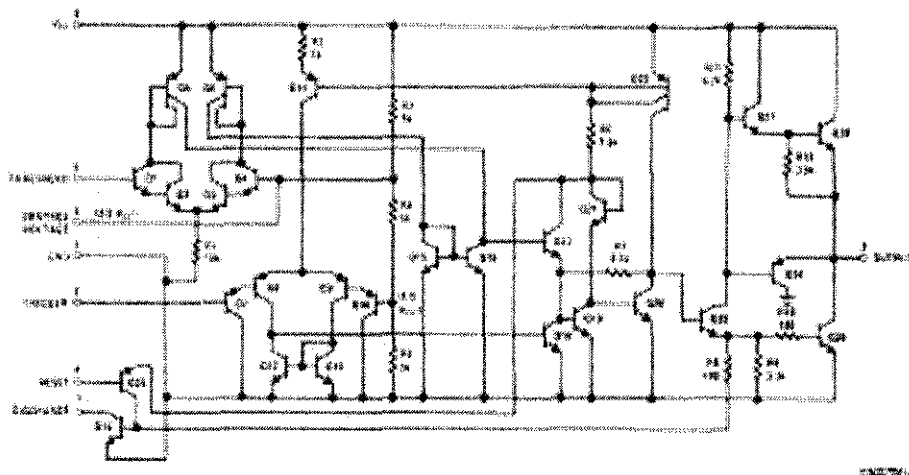
#### Features

- Direct replacement for SE555/NE555
- Timing from microseconds through hours
- Operates in both astable and monostable modes
- Adjustable duty cycle
- Output can source or sink 200 mA
- Output and supply TTL compatible
- Temperature stability better than 0.005% per °C
- Normally on and normally off output
- Available in 8-pin MSOP package

#### Applications

- Precision timing
- Pulse generation
- Sequential timing
- Time delay generation
- Pulse width modulation
- Pulse position modulation
- Linear ramp generator

#### Schematic Diagram



## Applications Information

### MONOSTABLE OPERATION

In this mode of operation, the timer functions as a one-shot (Figure 1). The external capacitor is initially held discharged by a transistor inside the timer. Upon application of a negative trigger pulse of less than  $1/3 V_{CC}$  to pin 2, the flip-flop is set which both releases the short circuit across the capacitor and drives the output high.

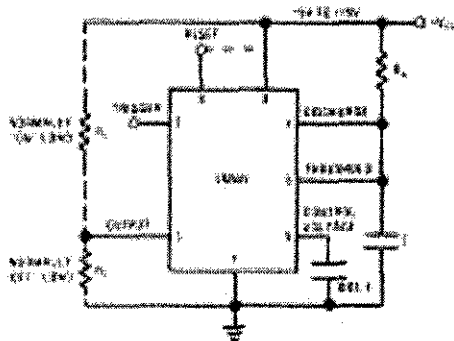
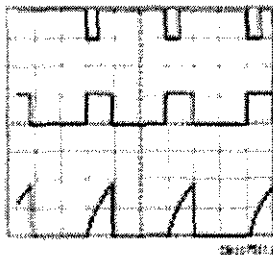


FIGURE 1. Monostable

The voltage across the capacitor then increases exponentially for a period of  $t = 1.1 R_A C$ , at the end of which time the voltage equals  $2/3 V_{CC}$ . The comparator then resets the flip-flop which in turn discharges the capacitor and drives the output into low state. Figure 2 shows the waveforms generated in this mode of operation. Since the charge and the threshold level of the comparator are both directly proportional to supply voltage, the timing interval is independent of supply.



$V_{CC} = 5V$   
 TIME = 0.1 ms/DIV.  
 $R_A = 0.11k\Omega$   
 $C = 0.01\mu F$

Top Trace: Input Trigger.  
 Middle Trace: Output Pulse.  
 Bottom Trace: Capacitor Voltage 2X/DIV.

FIGURE 2. Monostable Waveforms

During the timing cycle when the output is high, the further application of a trigger pulse will not affect the circuit so long as the trigger input is returned high at least 10µs before the end of the timing interval. However, the circuit can be reset during this time by the application of a negative pulse to the reset terminal (pin 4). The output will then remain in the low state until a trigger pulse is again applied.

When the reset function is not in use, it is recommended that it be connected to  $V_{CC}$  to avoid any possibility of false triggering.

Figure 3 is a nomograph for easy determination of  $R, C$  values for various time delays.

NOTE: In monostable operation, the trigger should be driven high before the end of timing cycle.

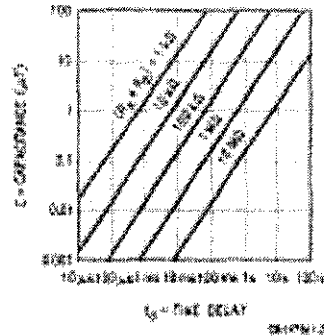


FIGURE 3. Time Delay

### ASTABLE OPERATION

If the circuit is connected as shown in Figure 4 (pins 2 and 6 connected) it will trigger itself and free run as a multivibrator. The external capacitor charges through  $R_A + R_B$  and discharges through  $R_C$ . Thus the duty cycle may be precisely set by the ratio of these two resistors.

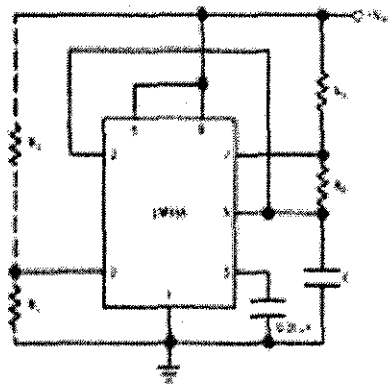


FIGURE 4. Astable

In this mode of operation, the capacitor charges and discharges between  $1/3 V_{CC}$  and  $2/3 V_{CC}$ . As in the triggered mode, the charge and discharge times, and therefore the frequency are independent of the supply voltage.

## APPENDIX F

```
*****
;
; The file was originally started with use of the 16F877 code
; template supplied by Microchip.
;
*****
;
; Last Revision Date      :      March 18, 2004
; File Version           :      1.0
;
; Company                :      University Technology Petronas
;
*****
;
; Files required: P16F877.inc
;                16F877.lkr
;
*****
list      p=16f877                ; List directive to define processor
          #include <p16f877.inc>   ; Processor specific variable definitions

          errorlevel -302         ; These remove unwanted warnings
          errorlevel -305         ; from the compile information listing
          errorlevel -306

          _CONFIG_CP_OFF & _WDT_OFF & _BODEN_OFF & _PWRTE_OFF & _XT_OSC & _WRT_ENABLE_ON
          & _LVP_OFF & _CPD_OFF

; '_CONFIG' directive is used to embed configuration data within .asm file.
; The labels following the directive are located in the respective .inc file.
; See respective data sheet for additional information on configuration word.

;**** VARIABLE DEFINITIONS

;**** Variables
input     EQU      0x60           ; Buffer used up to 0x2C+D'21' = 0x41
offsetLow EQU      0x22
offsetHigh EQU     0x23
length   EQU      0x24
char     EQU      0x25
tableOffset EQU    0x26
expectedLength EQU  0x27
hexByteLow EQU     0x28
hexByteHigh EQU    0x29
temp     EQU      0x2A

;**** Delay
DCOUNT1  EQU      0x2F           ; Used in Delay subroutines
DCOUNT2  EQU      0x30
DCOUNT3  EQU      0x31
DCOUNT4  EQU      0x32

; ** General
BANK0    EQU      0x0000        ; Data Memory start positions
BANK1    EQU      0x0080
BANK2    EQU      0x0100
BANK3    EQU      0x0180
```

```

PAGE0          EQU    0x0000          ; Program Memory start positions
PAGE1          EQU    0x0800

;*****
;*** On reset these are the first commands performed
;*****
        ORG     0x000          ; processor reset vector
        clr    PCLATH         ; ensure page bits are cleared
        goto   main           ; go to beginning of program

;*****
;*** This is the start of the main code for receiving input
;*****
main

start

        call   initializeChip    ; Initialize the entire Chip to known state

        call   initADC           ; initialize the ADC

mainLoop

        banksel BANK0
        movlw  0x64
        call   delayWx10ms

        call   readADC

        banksel BANK0

        movlw  0x03             ;binary'0000 0011'
        subwf  ADRESH,w
        btfs  STATUS,Z         ;check if high byte equal to h'03' (high voltage)
        goto  redlighton
        movlw  0x10             ;check if high byte equal to h'10' (low voltage)
        subwf  ADRESH,w
        btfs  STATUS,Z
        goto  greenlighton
        goto  yellowlighton

redlighton          ; High voltage
        call   redlight
        goto  resume

greenlighton       ; 3.74V to 1.26V
        call   greenlight
        goto  resume

yellowlighton     ; Low voltage
        call   yellowlight
        goto  resume

resume

        banksel BANK0
        movlw  0x64

        call   delayWx10ms

        goto  mainLoop

```

```

;*****
;      Function subroutine to turn on an LED at PORTC bit0
;
; receives      : nothing
; uses         : nothing
; returns      : nothing
;*****
redlight
    banksel BANK0
    movlw 0xFB
    movwf PORTC
    return

;*****
;      Function subroutine to turn on an LED at PORTC bit1
;
; receives      : nothing
; uses         : nothing
; returns      : nothing
;*****
greenlight
    banksel BANK0
    movlw 0xF7
    movwf PORTC
    return

;*****
;      Function subroutine to turn on RED LED at PORTC bit2
;                        and on buzzer PortC bit4
;
; receives      : nothing
; uses         : nothing
; returns      : nothing
;*****
yellowlight
    banksel BANK0                ;turn on yellow led and buzzer
    movlw 0xFD
    movwf PORTC
    return

;*****
;      Function subroutine to set specific registers on
;      initialization of software.
; receives      : nothing
; uses         : W
; returns      : nothing
;*****
initializeChip                ; Set all Pins to output (low) (high impedance)

    banksel BANK0                ; Move to Bank 0
    clrf PORTA                ; Clear Port A
    clrf PORTB                ; Clear Port B
    clrf PORTC                ; Clear Port C

    banksel BANK1                ; Move to Bank 1
    movlw 0x86                ;
    movwf ADCON1                ; Set all Port A to Digital I/O

    clrf TRISA                ; Set Port A to digital out
    clrf TRISB                ; Set Port B to digital out
    clrf TRISC                ; Set Port C to digital out

    banksel BANK0                ; Move to Bank 0
    clrf ADCON0                ; Shutoff all ADC function and set available
                                ; channel to AN0

```

```

        clr    PORTA                ; Clear Port A
        clr    PORTB                ; Clear Port B
        movlw 0xff
        movwf PORTC                ; set Port C

        return
;*****
;      ADC Module Subroutines
;*****

;*****
;      Function subroutine to initialize RAO as an ADC input.
;      Sets the (right justified) data flag.
; receives      : nothing
; uses          : W
; returns       : nothing
;*****
initADC
        banksel BANK1              ; Move to Bank 1
        movlw 0x82                  ; Set all AN pins on Port A as analog
        movwf ADCON1               ; and right justify data in ADRESH and ADRESL

        banksel BANK0              ; Move to Bank 0
        movlw 0x01                  ; Shutoff all ADC function and set available
        movwf ADCON0               ; channel to A0
                                        ; Also selects: A/D clock = Fosc/2

        movf  PORTA,W               ; Initialize Port A values to 0
        andlw 0x10
        movwf PORTA

        banksel BANK1              ; Set to bank1
        bcf   PIE1,ADIE            ; Disable A/D interrupt

        movf  TRISA,W               ; Initialize Port A Analog ANs to inputs
        iorlw 0x2f
        movwf TRISA

        banksel BANK0              ; Move to Bank 0
        bcf   PIR1,ADIF            ; Clear A/D interrupt flag

        return
;*****
;      Function subroutine to initiate a read of ADC port pins
;      which stores its value in ADRESH, and ADRESL (right justified)
; receives      : W - <0 to 4> pin to read
; uses          : W, char
; returns       : nothing
;*****
readADC

        movwf  char

        swapf  char                  ; Move input value into the ADC
        bcf   STATUS,C              ; pin selection bits
        rrf   char
        bsf   char,0                ; Set the enable bit 0
        movf  char,W

        movwf  ADCON0

```

```

bsf    ADCON0,GO           ; Start A/D conversion (sets bit 2)

htfsc  ADCON0,GO           ; Repeat until the GO bit is automatically
goto   $ - 1               ; set low by hardware

bcf    PIR1,ADIF           ; Clear A/D interrupt flag
; Values are now stored in ADRESH and ADRESL

return

```

```

;*****
;      Function subroutine to perform multiple 10ms Delays
; receives      : W - a multiplier of the 1ms time
; uses         : W,DCOUNT1,DCOUNT2, DCOUNT3
; returns      : nothing
;*****
delayWx10ms

```

```

movwf  DCOUNT1           ; DCOUNT1 = W
movlw  0x64
movwf  DCOUNT2
movlw  0x20
movwf  DCOUNT3
decfsz DCOUNT3           ; delay of > (((3 * DCOUNT3)+3)*DCOUNT2)+3*W)
goto   $ - 1             ; * instruction cycle time (1us for 4MHz)
decfsz DCOUNT2
goto   $ - 5             ; so currently W x 10ms
decfsz DCOUNT1
goto   $ - 9

return

```

```

END                       ; Directive 'end of program'
; Code reaching this point may perform irrationally.

```



## APPENDIX G

```
'Utk form resize property  
Public tinggi_form_lama  
Public lebar_form_lama  
Public tinggi_form_baru  
Public lebar_form_baru  
Public constraint_tinggi  
Public constraint_lebar
```

```
'Utk connection  
Public currentport  
Public currentbaudrate  
Public is_connected
```

```
Private Sub chkvber_Click()
```

```
End Sub
```

```
Private Sub disText_Change()
```

```
End Sub
```

```
Private Sub Form_Load()
```

```
'Constant utk form resize property  
tinggi_form_lama = Height  
tinggi_form_baru = 0  
lebar_form_lama = Width  
lebar_form_baru = 0  
constraint_tinggi = 5220  
constraint_lebar = 5910
```

```
Call FORM_INITIALIZATION  
Call ENABLE_OBJECT(False)
```

```
textmessage = Me.Caption & " started"  
WRITE_STATUS (textmessage)  
textmessage = "Serial connection has not yet been established"  
WRITE_STATUS (textmessage)
```

```
'inputText.Text = MSComm1.Settings
```

```
End Sub
```

```
Private Sub cmbport_Click()  
    currentport = cmbport.List(cmbport.ListIndex)
```

```
End Sub
```

```
Private Sub cmbbaudrate_Click()  
    currentbaudrate = cmbbaudrate.List(cmbbaudrate.ListIndex)
```

```
End Sub
```

```
Private Sub cmdexit_Click()  
  
    Call SERIAL_DISCONNECT  
    Unload Me
```

```
End Sub
```

```
Private Sub frmconnection_DragDrop(Source As Control, X As Single, Y As Single)
```

```
End Sub
```

```
Private Sub frmdisplay_DragDrop(Source As Control, X As Single, Y As Single)
```

```
End Sub
```

```
Private Sub frminput_DragDrop(Source As Control, X As Single, Y As Single)
```

```
End Sub
```

```
Private Sub frmstatus_DragDrop(Source As Control, X As Single, Y As Single)
```

```
End Sub
```

```
Private Sub inputText_Change()
```

```
End Sub
```

```
Private Sub Label1_Click()
```

```
End Sub
```

```
Private Sub Label2_Click()
```

End Sub

Private Sub MSComm1\_OnComm()

Dim strInput As String

Select Case MSComm1.CommEvent

Case comEvReceive

'display incoming event data to displaying textbox

strInput = MSComm1.Input

disText.SelText = strInput

End Select

End Sub

Private Sub ENABLE\_OBJECT(status\_object As Boolean)

If status\_object = True Then

disText.Enabled = True

inputText.Enabled = True

cmdsend.Enabled = True

chkvbc.Enabled = True

ElseIf status\_object = False Then

disText.Enabled = False

inputText.Enabled = False

cmdsend.Enabled = False

chkvbc.Enabled = False

End If

End Sub

Private Sub cmdsend\_Click()

'send the data

'if sending to other computer, include a trailing carriage return

MSComm1.Output = inputText.Text & vbCrLf

'place the focus back to the textbox

inputText.SetFocus

'select the current text to be overwritten

inputText.SelStart = 0

inputText.SelLength = Len(inputText.Text)

End Sub

```
Private Sub FORM_INITIALIZATION()
```

```
    'Title initialization
```

```
    Me.Caption = App.Title & " v" & App.Major & "." & App.Minor & App.Revision
```

```
    'Status initialization
```

```
    txtstatus.Text = ""
```

```
    'Combo box initialization
```

```
    'Put value in the combo box
```

```
    arrayport = Array("1", "2")
```

```
    arraybaudrate = Array("110", "300", "600", "1200", "2400", "9600", "14400",  
"19200", "28800", "38400", "56000", "128000", "256000")
```

```
    For count1 = 0 To 1
```

```
        cmbport.List(count1) = arrayport(count1)
```

```
    Next
```

```
    For count1 = 0 To 12
```

```
        cmbbaudrate.List(count1) = arraybaudrate(count1)
```

```
    Next
```

```
    'Set initial value
```

```
    cmbport.ListIndex = 0
```

```
    cmbbaudrate.ListIndex = 5
```

```
    'Set initial global value
```

```
    currentport = cmbport.List(cmbport.ListIndex)
```

```
    currentbaudrate = cmbbaudrate.List(cmbbaudrate.ListIndex)
```

```
    is_connected = False
```

```
End Sub
```

```
Private Sub SERIAL_DISCONNECT()
```

```
    'Close port
```

```
    If is_connected = True Then
```

```
        MSComm1.PortOpen = False
```

```
        is_connected = False
```

```
    End If
```

```
    'Write status
```

```
    textmessage = "Serial connection has been closed"
```

```
    WRITE_STATUS (textmessage)
```

```
End Sub
```

```

Private Sub SERIAL_CONNECT()
    'set the active serial port
    MSComm1.CommPort = currentport
    'MSComm1.CommPort = 1

    'set the baudrate,parity,databits,stopbits for the connection
    MSComm1.Settings = currentbaudrate & ",N,8,1"
    'MSComm1.Settings = "2400,N,8,1"

    'set the DTR and RTS flags
    MSComm1.DTREnable = True
    MSComm1.RTSEnable = True

    'enable the oncomm event for every received character
    'RThreshold=1,comEvReceive=enabled
    'RThreshold=0,comEvReceive=disabled
    MSComm1.RThreshold = 1

    'disable the oncomm event for send characters
    'SThreshold=1,comEvSend=enabled
    'SThreshold=0,comEvSend=disabled
    MSComm1.SThreshold = 0

    'Write status
    textmessage = "Connecting..."
    WRITE_STATUS (textmessage)

    On Error GoTo errorhandler
    'open the serial port
    MSComm1.PortOpen = True
    is_connected = True

    'Write status
    textmessage = "Serial connection has been made successfully"
    WRITE_STATUS (textmessage)
    textmessage = "Port = " & currentport
    WRITE_STATUS (textmessage)
    textmessage = "Baudrate = " & currentbaudrate
    WRITE_STATUS (textmessage)

    'This exit sub is to prevent the normal flow (without error) goes into errorhandler
    Exit Sub

errorhandler:
    a1 = MsgBox(Err.Description & vbCrLf & "[Error no.=" & Err.Number & "]",
vbExclamation, "Error")

```

```
is_connected = False
```

```
'Write status
```

```
textmessage = Err.Description & ". Serial connection attempt failed"
```

```
WRITE_STATUS (textmessage)
```

```
End Sub
```

```
Private Sub WRITE_STATUS(textmessage)
```

```
txtstatus.Text = txtstatus.Text & Format(Now, "hh:mm:ss") & " - " & textmessage &  
vbCrLf
```

```
End Sub
```

```
Private Sub tglconnect_Click()
```

```
If tglconnect.Value = True Then
```

```
'Connect
```

```
    tglconnect.Caption = "Disconnect"
```

```
    Call SERIAL_CONNECT
```

```
    If is_connected = True Then
```

```
        'If succesful
```

```
            Call ENABLE_OBJECT(True)
```

```
    ElseIf is_connected = False Then
```

```
        'If still not connected
```

```
            Call ENABLE_OBJECT(False)
```

```
            tglconnect.Caption = "Connect"
```

```
            'Turn OFF button back
```

```
            tglconnect.Value = False
```

```
    End If
```

```
    ElseIf tglconnect.Value = False Then
```

```
        'Disconnect
```

```
            Call ENABLE_OBJECT(False)
```

```
            tglconnect.Caption = "Connect"
```

```
            Call SERIAL_DISCONNECT
```

```
    End If
```

```
End Sub
```

```

Private Sub Form_Resize()
    'CONSTANT
    lebar_form = Width
    tinggi_form = Height

    'Set minimum constraint for height
    If tinggi_form >= constraint_tinggi Then
        'Renew value
        tinggi_form_baru = tinggi_form
    Else If tinggi_form < constraint_tinggi Then
        'Renew value
        tinggi_form_baru = constraint_tinggi
    End If

    bezatinggi_form = tinggi_form_baru - tinggi_form_lama
    tinggi_form_lama = tinggi_form_baru

    'Reset object
    disText.Height = disText.Height + bezatinggi_form
    frmdisplay.Height = frmdisplay.Height + bezatinggi_form
    frminput.Top = frminput.Top + bezatinggi_form

    'Set minimum constraint for width
    If lebar_form >= constraint_lebar Then
        'Renew value
        lebar_form_baru = lebar_form
    ElseIf lebar_form < constraint_lebar Then
        'Renew value
        lebar_form_baru = constraint_lebar
    End If

    bezalebar_form = lebar_form_baru - lebar_form_lama
    lebar_form_lama = lebar_form_baru

    'Reset object
    disText.Width = disText.Width + bezalebar_form
    frmdisplay.Width = frmdisplay.Width + bezalebar_form
    frminput.Width = frminput.Width + bezalebar_form
    inputText.Width = inputText.Width + bezalebar_form
    cmdsend.Left = cmdsend.Left + bezalebar_form

End Sub

Private Sub txtstatus_Change()

End Sub

```

## APPENDIX H

# PIC16F87X

### 11.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the other devices.

The analog input charges a sample and hold capacitor. The output of the sample and hold capacitor is the input into the converter. The converter then generates a digital result of this analog level via successive approximation. The A/D conversion of the analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low voltage reference input that is software-selectable to some combination of  $V_{DD}$ ,  $V_{SS}$ , RA2, or RA3.

The A/D converter has a unique feature of being able to operate while the device is in SLEEP mode. To operate in SLEEP, the A/D clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register0 (ADCON0)
- A/D Control Register1 (ADCON1)

The ADCON0 register, shown in Register 11-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 11-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference), or as digital I/O.

Additional information on using the A/D module can be found in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 11-1: ADCON0 REGISTER (ADDRESS: 1Fh)

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0   | U-0   | R/W-0 |
|-------|-------|-------|-------|-------|---------|-------|-------|
| ADCS1 | ADCS0 | CHS2  | CHS1  | CHS0  | GO/DONE | —     | ADCON |
|       |       |       |       |       |         | bit 7 | bit 0 |

|         |  |
|---------|--|
| bit 7-6 | ADCS1:ADCS0: A/D Conversion Clock Select bits<br>00 = Fosc/2<br>01 = Fosc/8<br>10 = Fosc/32<br>11 = Frc (clock derived from the internal A/D module RC oscillator)   |
| bit 5-3 | CHS2:CHS0: Analog Channel Select bits<br>000 = channel 0, (RA0/AN0)<br>001 = channel 1, (RA1/AN1)<br>010 = channel 2, (RA2/AN2)<br>011 = channel 3, (RA3/AN3)<br>100 = channel 4, (RA5/AN4)<br>101 = channel 5, (RE0/AN5) <sup>(1)</sup><br>110 = channel 6, (RE1/AN6) <sup>(1)</sup><br>111 = channel 7, (RE2/AN7) <sup>(1)</sup> |
| bit 2   | GO/DONE: A/D Conversion Status bit<br><b>!!ADCON = 1:</b><br>1 = A/D conversion in progress (setting this bit starts the A/D conversion)<br>0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)   |
| bit 1   | Unimplemented. Read as '0'   |
| bit 0   | ADON: A/D On bit<br>1 = A/D converter module is operating<br>0 = A/D converter module is shut-off and consumes no operating current  |

Note 1: These channels are not available on PIC16F873/876 devices.

**Legend:**

|                   |                  |                                    |
|-------------------|------------------|------------------------------------|
| R = Readable bit  | W = Writable bit | U = Unimplemented bit, read as '0' |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared               |
|                   |                  | x = Bit is unknown                 |



# PIC16F87X

REGISTER 11-2: ADCON1 REGISTER (ADDRESS 9Fh)

|       |     |       |     |       |       |       |       |
|-------|-----|-------|-----|-------|-------|-------|-------|
| U-0   | U-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
| ADFM  | —   | —     | —   | PCFG3 | PCFG2 | PCFG1 | PCFG0 |
| bit 7 |     |       |     | bit 0 |       |       |       |

- bit 7      ADFM: A/D Result Format Select bit  
1 = Right justified. 8 Most Significant bits of ADRESH are read as '0'.  
0 = Left justified. 6 Least Significant bits of ADRESL are read as '0'.
- bit 6-4    Unimplemented: Read as '0'
- bit 3-0    PCFG3:PCFG0: A/D Port Configuration Control bits

| PCFG3:<br>PCFG0 | AN7 <sup>(1)</sup><br>RE2 | AN6 <sup>(1)</sup><br>RE1 | AN5 <sup>(1)</sup><br>RE0 | AN4<br>RA5 | AN3<br>RA3 | AN2<br>RA2 | AN1<br>RA1 | AN0<br>RA0 | VREF+ | VREF- | Chan#<br>Ref# <sup>(2)</sup> |
|-----------------|---------------------------|---------------------------|---------------------------|------------|------------|------------|------------|------------|-------|-------|------------------------------|
| 0000            | A                         | A                         | A                         | A          | A          | A          | A          | A          | VDD   | VSS   | 8/0                          |
| 0001            | A                         | A                         | A                         | A          | VREF+      | A          | A          | A          | RA3   | VSS   | 7/1                          |
| 0010            | D                         | D                         | D                         | A          | A          | A          | A          | A          | VDD   | VSS   | 5/0                          |
| 0011            | D                         | D                         | D                         | A          | VREF+      | A          | A          | A          | RA3   | VSS   | 4/1                          |
| 0100            | D                         | D                         | D                         | D          | A          | D          | A          | A          | VDD   | VSS   | 3/0                          |
| 0101            | D                         | D                         | D                         | D          | VREF+      | D          | A          | A          | RA3   | VSS   | 2/1                          |
| 011x            | D                         | D                         | D                         | D          | D          | D          | D          | D          | VDD   | VSS   | 0/0                          |
| 1000            | A                         | A                         | A                         | A          | VREF+      | VREF-      | A          | A          | RA3   | RA2   | 6/2                          |
| 1001            | D                         | D                         | A                         | A          | A          | A          | A          | A          | VDD   | VSS   | 6/0                          |
| 1010            | D                         | D                         | A                         | A          | VREF+      | A          | A          | A          | RA3   | VSS   | 5/1                          |
| 1011            | D                         | D                         | A                         | A          | VREF+      | VREF-      | A          | A          | RA3   | RA2   | 4/2                          |
| 1100            | D                         | D                         | D                         | A          | VREF+      | VREF-      | A          | A          | RA3   | RA2   | 3/2                          |
| 1101            | D                         | D                         | D                         | D          | VREF+      | VREF-      | A          | A          | RA3   | RA2   | 2/2                          |
| 1110            | D                         | D                         | D                         | D          | D          | D          | D          | A          | VDD   | VSS   | 1/0                          |
| 1111            | D                         | D                         | D                         | D          | VREF+      | VREF-      | D          | A          | RA3   | RA2   | 1/2                          |

A = Analog Input    D = Digital I/O

- Note: 1: These channels are not available on PIC16F873/876 devices.  
 2: This column indicates the number of analog channels available as A/D inputs and the number of analog channels used as voltage reference inputs.

|                    |                  |                                    |                    |
|--------------------|------------------|------------------------------------|--------------------|
| Legend             |                  |                                    |                    |
| R = Readable bit   | W = Writable bit | U = Unimplemented bit, read as '0' |                    |
| - n = Value at POR | '1' = Bit is set | '0' = Bit is cleared               | x = Bit is unknown |

The ADRESH:ADRESL registers contain the 10-bit result of the A/D conversion. When the A/D conversion is complete, the result is loaded into this A/D result register pair, the GO/DONE bit (ADCON0<2>) is cleared and the A/D interrupt flag bit ADIF is set. The block diagram of the A/D module is shown in Figure 11-1.

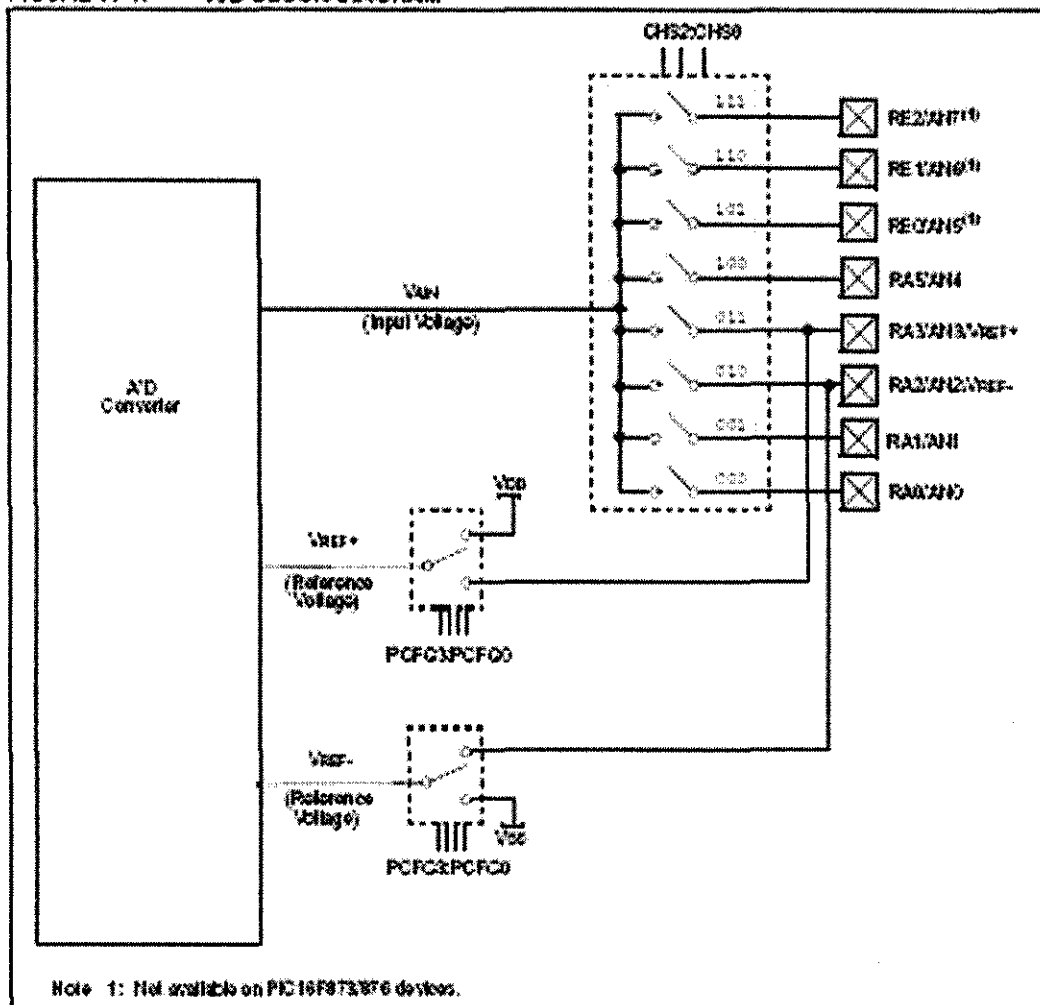
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as input.

To determine sample time, see Section 11.1. After this acquisition time has elapsed, the A/D conversion can be started.

These steps should be followed for doing an A/D Conversion:

1. Configure the A/D module:
  - Configure analog plus/voltage reference and digital I/O (ADCON1)
  - Select A/D input channel (ADCON0)
  - Select A/D conversion clock (ADCON0)
  - Turn on A/D module (ADCON0)
2. Configure A/D Interrupt (if desired):
  - Clear ADFR bit
  - Set ADE bit
  - Set PBE bit
  - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion
  - Set GO/DONE bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
  - Polling for the GO/DONE bit to be cleared (with interrupts enabled); OR
  - Waiting for the A/D interrupt
6. Read A/D result register pair (ADRESH:ADRESL), clear bit ADIF if required.
7. For the next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as  $T_{AD}$ . A minimum wait of  $2T_{AD}$  is required before the next acquisition starts.

FIGURE 11-1: A/D BLOCK DIAGRAM



# PIC16F87X

## 11.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 11-2. The source impedance ( $R_S$ ) and the internal sampling switch ( $R_{SS}$ ) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch ( $R_{SS}$ ) impedance varies over the device voltage ( $V_{DD}$ ), see Figure 11-2. The maximum recommended impedance for analog sources is 10 k $\Omega$ . As the impedance is decreased, the acquisition time may be decreased.

After the analog input channel is selected (changed), the acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 11-1 may be used. This equation assumes that 1/2 LSB error is used (1024 steps for the A/D). The 1/2 LSB error is the maximum error allowed for the A/D to meet its specified resolution.

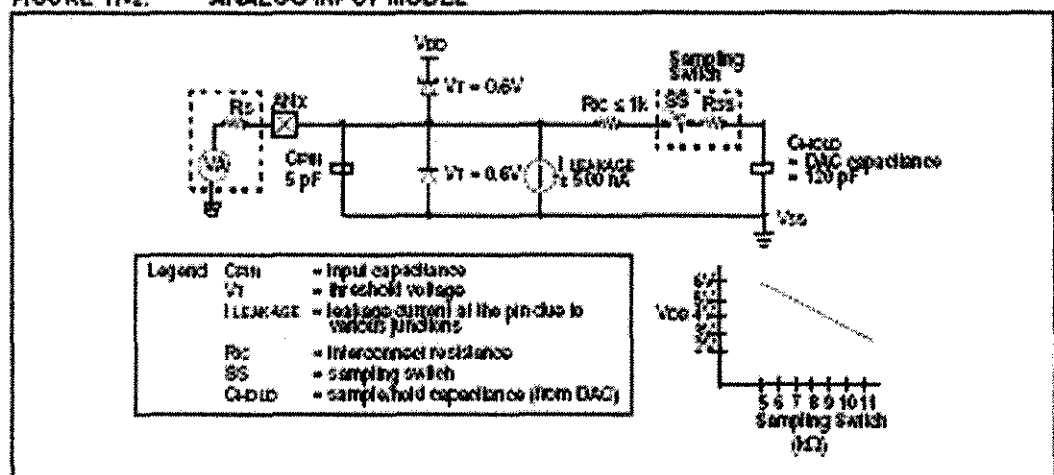
To calculate the minimum acquisition time,  $T_{ACQ}$ , see the PICmicro™ Mid-Range Reference Manual (DS39023).

### EQUATION 11-1: ACQUISITION TIME

|           |  |
|-----------|--|
| $T_{ACQ}$ | = Amplifier Settling Time +<br>Hold Capacitor Charging Time +<br>Temperature Coefficient |
|           | = $T_{AMP} + T_C + T_{COFF}$   |
|           | = $2\mu s + T_C + [(Temperature - 25^\circ C)(0.00\mu s/^\circ C)]$                      |
| $T_C$     | = $CHOLD \cdot (R_S + R_{SS} + R_{SS}) \ln(1/2047)$                                      |
|           | = $130pF (1k\Omega + 7k\Omega + 10k\Omega) \ln(0.0004885)$                               |
|           | = $16.47\mu s$   |
| $T_{ACQ}$ | = $2\mu s + 16.47\mu s + [(50^\circ C - 25^\circ C)(0.00\mu s/^\circ C)]$                |
|           | = $19.72\mu s$   |

- Note 1: The reference voltage ( $V_{REF}$ ) has no effect on the equation, since it cancels itself out.
- 2: The charge holding capacitor (CHOLD) is not discharged after each conversion.
- 3: The maximum recommended impedance for analog sources is 10 k $\Omega$ . This is required to meet the pin leakage specification.
- 4: After a conversion has completed, a 2.0 $\mu s$  delay must complete before acquisition can begin again. During this time, the holding capacitor is not connected to the selected A/D input channel.

FIGURE 11-2: ANALOG INPUT MODEL



## 11.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as  $T_{AD}$ . The A/D conversion requires a minimum  $12T_{AD}$  per 10-bit conversion. The source of the A/D conversion clock is software selected. The four possible options for  $T_{AD}$  are:

- $2T_{osc}$
- $8T_{osc}$
- $32T_{osc}$
- Internal A/D module RC oscillator ( $3-6 \mu s$ )

For correct A/D conversions, the A/D conversion clock ( $T_{AD}$ ) must be selected to ensure a minimum  $T_{AD}$  time of  $1.6 \mu s$ .

Table 11-1 shows the resultant  $T_{AD}$  times derived from the device operating frequencies and the A/D clock source selected.

TABLE 11-1:  $T_{AD}$  vs. MAXIMUM DEVICE OPERATING FREQUENCIES (STANDARD DEVICES (C))

| AD Clock Source ( $T_{AD}$ ) |             | Maximum Device Frequency |
|------------------------------|-------------|--------------------------|
| Operation                    | ADCS1:ADCS0 | Max.                     |
| $2T_{osc}$                   | 00          | 1.25 MHz                 |
| $8T_{osc}$                   | 01          | 5 MHz                    |
| $32T_{osc}$                  | 10          | 20 MHz                   |
| RC (1, 2, 3)                 | 11          | (Note 1)                 |

Note 1: The RC source has a typical  $T_{AD}$  time of  $4 \mu s$ , but can vary between  $3-6 \mu s$ .

2: When the device frequencies are greater than 1 MHz, the RC A/D conversion clock source is only recommended for SLEEP operation.

3: For extended voltage devices (LC), please refer to the Electrical Characteristics (Sections 15.1 and 15.2).

## 11.3 Configuring Analog Port Pins

The ADCON1 and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bit set (input). If the TRIS bit is cleared (output), the digital output level ( $V_{OH}$  or  $V_{OL}$ ) will be converted.

The A/D operation is independent of the state of the CHS2:CHS0 bits and the TRIS bits.

Note 1: When reading the port register, any pin configured as an analog input channel will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.

2: Analog levels on any pin that is defined as a digital input (including the AN7:AN0 pins), may cause the input buffer to consume current that is out of the device specifications.

# PIC16F87X

## 11.4 A/D Conversions

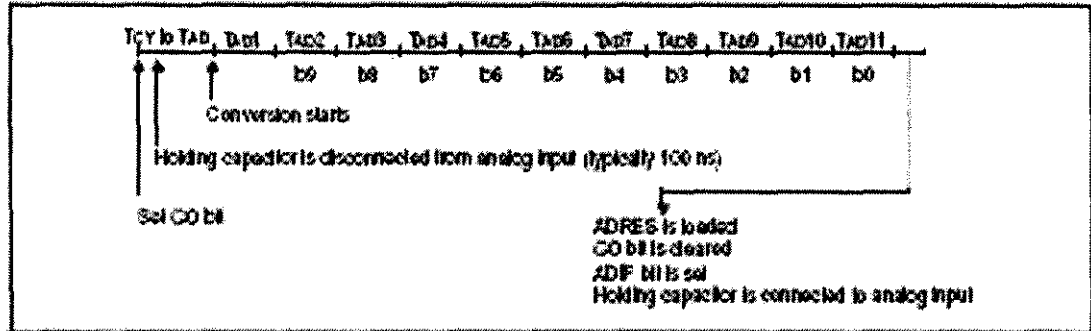
Clearing the GO/DONE bit during a conversion will abort the current conversion. The A/D result register pair will NOT be updated with the partially completed A/D conversion sample. That is, the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers). After the A/D conversion is aborted, a 2T<sub>AD</sub> wait is required before the next

acquisition is started. After this 2T<sub>AD</sub> wait, acquisition on the selected channel is automatically started. The GO/DONE bit can then be set to start the conversion.

In Figure 11-3, after the GO bit is set, the first time segment has a minimum of T<sub>cy</sub> and a maximum of T<sub>AD</sub>.

Note: The GO/DONE bit should NOT be set in the same instruction that turns on the A/D.

FIGURE 11-3: A/D CONVERSION T<sub>AD</sub> CYCLES

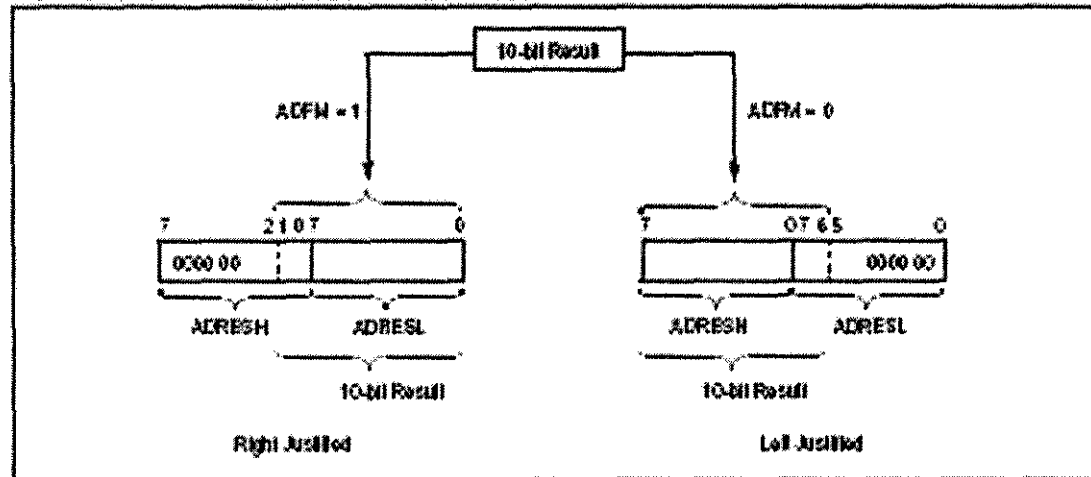


### 11.4.1 A/D RESULT REGISTERS

The ADRESH:ADRESL register pair is the location where the 10-bit A/D result is loaded at the completion of the A/D conversion. This register pair is 16-bit wide. The A/D module gives the flexibility to left or right justify the 10-bit result in the 16-bit result register. The A/D

Format Select bit (ADFM) controls this justification. Figure 11-4 shows the operation of the A/D result justification. The extra bits are loaded with 0's. When an A/D result will not overwrite these locations (A/D disable), these registers may be used as two general purpose 8-bit registers.

FIGURE 11-4: A/D RESULT JUSTIFICATION



## 11.5 A/D Operation During SLEEP

The A/D module can operate during SLEEP mode. This requires that the A/D clock source be set to RC (ADCS1:ADCS0 = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the `sleep` instruction to be executed, which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/DONE bit will be cleared and the result loaded into the ADRES register. If the A/D interrupt is enabled, the device will wake-up from SLEEP. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a `sleep` instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

**Note:** For the A/D module to operate in SLEEP, the A/D clock source must be set to RC (ADCS1:ADCS0 = 11). To allow the conversion to occur during SLEEP, ensure the `sleep` instruction immediately follows the instruction that sets the GO/DONE bit.

## 11.6 Effects of a RESET

A device RESET forces all registers to their RESET state. This forces the A/D module to be turned off, and any conversion is aborted. All A/D input pins are configured as analog inputs.

The value that is in the ADRESH:ADRESL registers is not modified for a Power-on Reset. The ADRESH:ADRESL registers will contain unknown data after a Power-on Reset.

TABLE 11-2: REGISTER BITS ASSOCIATED WITH A/D

| Address              | Name   | Bit 7                         | Bit 6 | Bit 5   | Bit 4   | Bit 3 | Bit 2                     | Bit 1  | Bit 0  | Value on POR, BOR | Value on MCLR, WDT |
|----------------------|--------|-------------------------------|-------|---|---------|-------|---------------------------|--------|--------|-------------------|--------------------|
| 06h, 06h, 106h, 106h | INTCON | OE                            | PEE   | TOE   | INTF    | RFIF  | TOF                       | INTF   | RIF    | 0000 000x         | 0000 000x          |
| 0Ch                  | PIR1   | PSPIF <sup>(1)</sup>          | ADIF  | RCF   | TOIF    | SSPIF | OCPIF                     | TMR3IF | TMR1IF | 0000 0000         | 0000 0000          |
| 8Ch                  | PIE1   | PSPIE <sup>(1)</sup>          | ADE   | RCE   | TOE     | SSPIE | OCPIE                     | TMR3IE | TMR1IE | 0000 0000         | 0000 0000          |
| 1Eh                  | ADRESH | A/D Result Register High Byte |       |   |         |       |                           |        |        | xxxxxx xxxxxx     | xxxxxx xxxxxx      |
| 0Eh                  | ADRESL | A/D Result Register Low Byte  |       |   |         |       |                           |        |        | xxxxxx xxxxxx     | xxxxxx xxxxxx      |
| 1Fh                  | ADCON0 | ADCS1                         | ADCS0 | CHS2  | CHS1    | CHS0  | GO/DONE                   | —      | ADON   | 0000 0000         | 0000 0000          |
| 9Fh                  | ADCON1 | ADFM                          | —     | —   | —       | PCFG3 | PCFG2                     | PCFG1  | PCFG0  | 0000 0000         | 0000 0000          |
| 85h                  | TRISA  | —                             | —     | PORTA Data Direction Register                       |         |       |                           |        |        | 0000 0000         | 0000 0000          |
| 0Ch                  | PORTA  | —                             | —     | PORTA Data Latch when written; PORTA pins when read |         |       |                           |        |        | 0000 0000         | 0000 0000          |
| 9Ah <sup>(1)</sup>   | TRISE  | BF                            | CBF   | BCF   | PSPNODE | —     | PORTE Data Direction bits |        |        | 0000 0000         | 0000 0000          |
| 0A <sup>(1)</sup>    | PORTE  | —                             | —     | —   | —       | —     | RE2                       | RE1    | RE0    | 0000 000x         | 0000 000x          |

Legend: x = unknown, 0 = unchanged, - = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

Note 1: These registers/bits are not available on the 28-pin devices.

# APPENDIX I

154223 Rev. 05/11/97

# MAXIM +5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

## General Description

The MAX220-MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where +12V is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245/MAX346/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

## Applications

- Portable Computers
- Low-Power Modems
- Interface Translation
- Battery-Powered RS-232 Systems
- Multi-Drop RS-232 Networks

## Features

### Superior to Bipolar

- Operate from Single +5V Power Supply (+5V and +12V—MAX231,MAX239)
- Low-Power Receive Mode in Shutdown (MAX223,MAX242)
- Meet All EIA/TIA-232E and V.28 Specifications
- Multiple Drivers and Receivers
- 3-State Driver and Receiver Outputs
- Open-Line Detection (MAX243)

## Ordering Information

| PART       | TEMP. RANGE     | PIN-PACKAGE  |
|------------|-----------------|--------------|
| MAX220-EE  | 0°C to +70°C    | 16 Basic DIP |
| MAX220-CE  | 0°C to +70°C    | 16 Narrow SO |
| MAX220-CNE | 0°C to +70°C    | 16 Wide SO   |
| MAX220-CD  | 0°C to +70°C    | Dice*        |
| MAX220-EPE | -40°C to +85°C  | 16 Basic DIP |
| MAX220-ESE | -40°C to +85°C  | 16 Narrow SO |
| MAX220-EWE | -40°C to +85°C  | 16 Wide SO   |
| MAX220-EJE | -40°C to +85°C  | 16 CERDIP    |
| MAX220-ELE | -55°C to +125°C | 16 CERDIP    |

Ordering information continued at end of data sheet.  
\*Contact factory for size specifications.

## Selection Table

| Part Number      | Power Supply (V)    | No. of RS-232 Drivers/Rx | No. of Ext. Caps. | Normal Cap. Value (µF) | SHDN & Three-State | For Active In SHDN | Data Rate (bps) | Features   |
|------------------|---------------------|--------------------------|-------------------|------------------------|--------------------|--------------------|-----------------|--|
| MAX220           | +5                  | 1/2                      | 4                 | 1.0 (0.1)              | No                 | —                  | 120             | Low-power shutdown   |
| MAX221           | +5                  | 1/2                      | 4                 | 0.1                    | Yes                | —                  | 200             | Low-power shutdown   |
| MAX221 (MAX213)  | +5                  | 4/2                      | 4                 | 1.0 (0.1)              | Yes                | ✓                  | 120             | MAX241 and receiver's mode in shutdown   |
| MAX221           | +5                  | 1/2                      | 0                 | —                      | Yes                | ✓                  | 120             | Available in SO  |
| MAX2210 (MAX220) | +5                  | 1/2                      | 4                 | 1.0 (0.1)              | Yes                | —                  | 120             | 1 driver with shutdown   |
| MAX2211 (MAX220) | +5 and +12 or +13.1 | 1/2                      | 2                 | 1.0 (0.1)              | No                 | —                  | 120             | Standard +5V or battery supplied same functions as MAX220                        |
| MAX2211 (MAX220) | +5                  | 1/2                      | 4                 | 1.0 (0.1)              | No                 | —                  | 120 (24)        | Industry standard  |
| MAX2211A         | +5                  | 1/2                      | 4                 | 0.1                    | No                 | —                  | 200             | High speed rate, small caps  |
| MAX2211 (MAX220) | +5                  | 1/2                      | 0                 | —                      | No                 | —                  | 120             | No external caps   |
| MAX2213A         | +5                  | 1/2                      | 0                 | —                      | No                 | —                  | 200             | No external caps, high speed rate  |
| MAX2214 (MAX220) | +5                  | 4/2                      | 4                 | 1.0 (0.1)              | No                 | —                  | 120             | 3-state 14V  |
| MAX2211 (MAX220) | +5                  | 1/2                      | 0                 | —                      | Yes                | —                  | 120             | No external caps   |
| MAX2216 (MAX220) | +5                  | 4/2                      | 4                 | 1.0 (0.1)              | Yes                | —                  | 120             | Shutdown, three state  |
| MAX2217 (MAX220) | +5                  | 1/2                      | 4                 | 1.0 (0.1)              | No                 | —                  | 120             | Complements 384 FC serial port   |
| MAX2216 (MAX220) | +5                  | 4/2                      | 4                 | 1.0 (0.1)              | No                 | —                  | 120             | Replaces 1449 and 1447   |
| MAX2219 (MAX220) | +5 and +12 or +13.1 | 1/2                      | 2                 | 1.0 (0.1)              | No                 | —                  | 120             | Standard +5V or battery supplied, single-package solution for 384 FC serial port |
| MAX2240          | +5                  | 1/2                      | 4                 | 1.0                    | Yes                | —                  | 120             | QFN package  |
| MAX2241 (MAX220) | +5                  | 4/2                      | 4                 | 1.0 (0.1)              | Yes                | —                  | 120             | Complements 384 FC serial port   |
| MAX2241          | +5                  | 1/2                      | 4                 | 0.1                    | Yes                | ✓                  | 200             | Separate shutdown and enable   |
| MAX2241          | +5                  | 1/2                      | 4                 | 0.1                    | No                 | —                  | 200             | Operates shutdown, simplified setting  |
| MAX2244          | +5                  | 4/2/0                    | 4                 | 1.0                    | No                 | —                  | 120             | High speed rate  |
| MAX2243          | +5                  | 4/2/0                    | 0                 | —                      | Yes                | ✓                  | 120             | High speed rate, no caps, and shutdown mode                                      |
| MAX2240          | +5                  | 4/2/0                    | 0                 | —                      | Yes                | ✓                  | 120             | High speed rate, no caps, three shutdown modes                                   |
| MAX2241          | +5                  | 4/2                      | 0                 | —                      | Yes                | ✓                  | 120             | High speed rate, no caps, three shutdown modes                                   |
| MAX2246          | +5                  | 4/2                      | 4                 | 1.0                    | Yes                | ✓                  | 120             | High speed rate, separate half-chip enables                                      |
| MAX2247          | +5                  | 4/2/0                    | 4                 | 1.0                    | Yes                | ✓                  | 120             | Available in quad package  |

MAXIM

Maxim Integrated Products 1

For free samples & the latest literature: <http://www.maxim-ic.com>, or phone 1-800-998-8800.  
For small orders, phone 408-737-7600 ext. 3468.

## +5V-Powered, Multichannel RS-232 Drivers/Receivers

### Detailed Description

The MAX220-MAX249 contain four sections: dual charge-pump DC-DC voltage converters, RS-232 drivers, RS-232 receivers, and receiver and transmitter enable control inputs.

#### Dual Charge-Pump Voltage Converter

The MAX220-MAX249 have two internal charge pumps that convert +5V to  $\pm 10V$  (unloaded) for RS-232 driver operation. The first converter uses capacitor C1 to double the +5V input to +10V on C3 at the  $V_+$  output. The second converter uses capacitor C2 to invert +10V to -10V on C4 at the  $V_-$  output.

A small amount of power may be drawn from the +10V ( $V_+$ ) and -10V ( $V_-$ ) outputs to power external circuitry (see the Typical Operating Characteristics section), except on the MAX225 and MAX245-MAX247, where these pins are not available.  $V_+$  and  $V_-$  are not regulated, so the output voltage drops with increasing load current. Do not load  $V_+$  and  $V_-$  to a point that violates the minimum +5V EIA/TIA-232E driver output voltage when sourcing current from  $V_+$  and  $V_-$  to external circuitry.

When using the shutdown feature in the MAX222, MAX225, MAX230, MAX235, MAX236, MAX240, MAX241, and MAX245-MAX249, avoid using  $V_+$  and  $V_-$  to power external circuitry. When these ports are shut down,  $V_-$  falls to 0V, and  $V_+$  falls to +5V. For applications where a +10V external supply is applied to the  $V_+$  pin (instead of using the internal charge pump to generate +10V), the C1 capacitor must not be installed and the SHDN pin must be tied to  $V_{CC}$ . This is because  $V_+$  is internally connected to  $V_{CC}$  in shutdown mode.

#### RS-232 Drivers

The typical driver output voltage swing is  $\pm 8V$  when loaded with a nominal 3k $\Omega$  RS-232 receiver and  $V_{CC} = +5V$ . Output swing is guaranteed to meet the EIA/TIA-232E and V.28 specification, which calls for  $\pm 5V$  minimum driver output levels under worst-case conditions. These include a minimum 3k $\Omega$  load,  $V_{CC} = +4.5V$ , and maximum operating temperature. Unloaded driver output voltage ranges from ( $V_+ - 1.5V$ ) to ( $V_- + 0.5V$ ).

Input thresholds are both TTL and CMOS compatible. The inputs of unused drivers can be left unconnected since 400k $\Omega$  input pull-up resistors to  $V_{CC}$  are built in. The pull-up resistors force the outputs of unused drivers low because all drivers invert. The internal input pull-up resistors typically source 12 $\mu A$ , except in shutdown mode where the pull-ups are disabled. Driver outputs turn off and enter a high-impedance state—where leakage current is typically microamperes (maximum 25 $\mu A$ )—when in shutdown mode, in three-state mode, or

when device power is removed. Outputs can be driven to  $\pm 15V$ . The power-supply current typically drops to 8 $\mu A$  in shutdown mode.

The MAX230 has a receiver three-state control line, and the MAX223, MAX225, MAX235, MAX236, MAX240, and MAX241 have both a receiver three-state control line and a low-power shutdown control. Table 2 shows the effects of the shutdown control and receiver three-state control on the receiver outputs.

The receiver TTL/CMOS outputs are in a high-impedance, three-state mode whenever the three-state enable line is high (for the MAX225/MAX235/MAX236/MAX239-MAX241), and are also high-impedance whenever the shutdown control line is high.

When in low-power shutdown mode, the driver outputs are turned off and their leakage current is less than 1 $\mu A$  with the driver output pulled to ground. The driver output leakage remains less than 1 $\mu A$ , even if the transmitter output is backdriven between 0V and ( $V_{CC} + 0V$ ). Below -0.5V, the transmitter is diode clamped to ground with 1k $\Omega$  series impedance. The transmitter is also zener clamped to approximately  $V_{CC} + 0V$ , with a series impedance of 1k $\Omega$ .

The driver output slew rate is limited to less than 30V/ $\mu s$  as required by the EIA/TIA-232E and V.28 specifications. Typical slew rates are 24V/ $\mu s$  unloaded and 10V/ $\mu s$  loaded with 3 $\Omega$  and 250pF.

#### RS-232 Receivers

EIA/TIA-232E and V.28 specifications define a voltage level greater than 3V as a logic 0, so all receivers invert. Input thresholds are set at 0.8V and 2.1V, so receivers respond to TTL level inputs as well as EIA/TIA-232E and V.28 levels.

The receiver inputs withstand an input overvoltage up to  $\pm 25V$  and provide input terminating resistors with nominal 3k $\Omega$  values. The receivers implement Type 1 interpretation of the fault conditions of V.28 and EIA/TIA-232E.

Table 2. Three-State Control of Receivers

| PART                       | SHDN               | SHDN                | EN               | EN(R)            | RECEIVERS                                  |
|----------------------------|--------------------|---------------------|------------------|------------------|--|
| MAX223                     | --                 | Low<br>High<br>High | X<br>Low<br>High | --               | High Impedance<br>Active<br>High Impedance |
| MAX225                     | --                 | --                  | --               | Low<br>High      | High Impedance<br>Active                   |
| MAX235<br>MAX236<br>MAX240 | Low<br>Low<br>High | --                  | --               | Low<br>High<br>X | High Impedance<br>Active<br>High Impedance |



# +5V-Powered, Multichannel RS-232 Drivers/Receivers

**MAX220-MAX249**

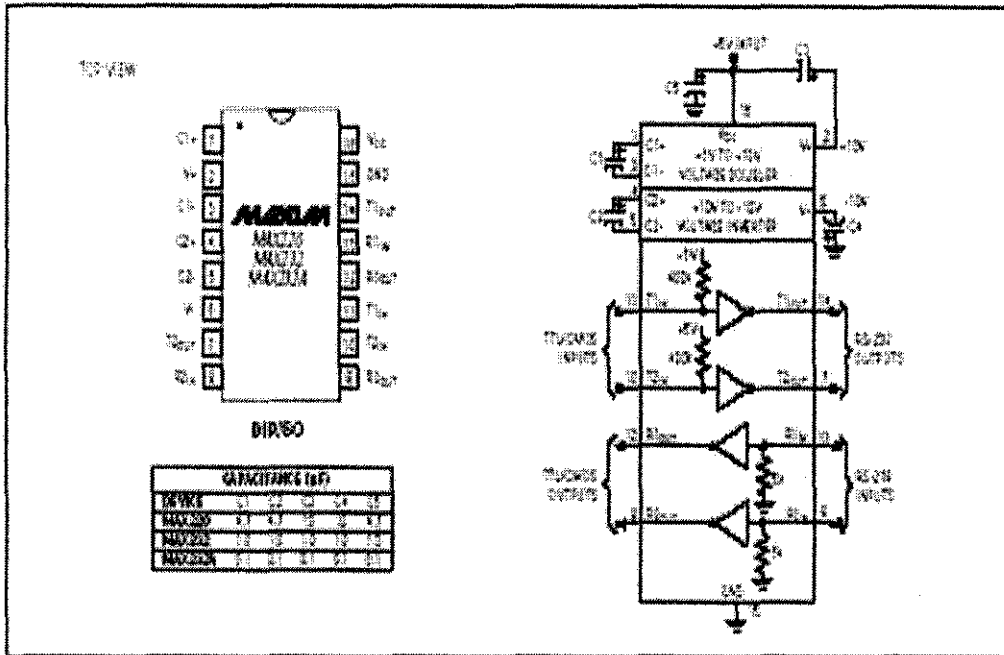


Figure 5. MAX220/MAX222/MAX228 Pin Configuration and Typical Operating Circuit

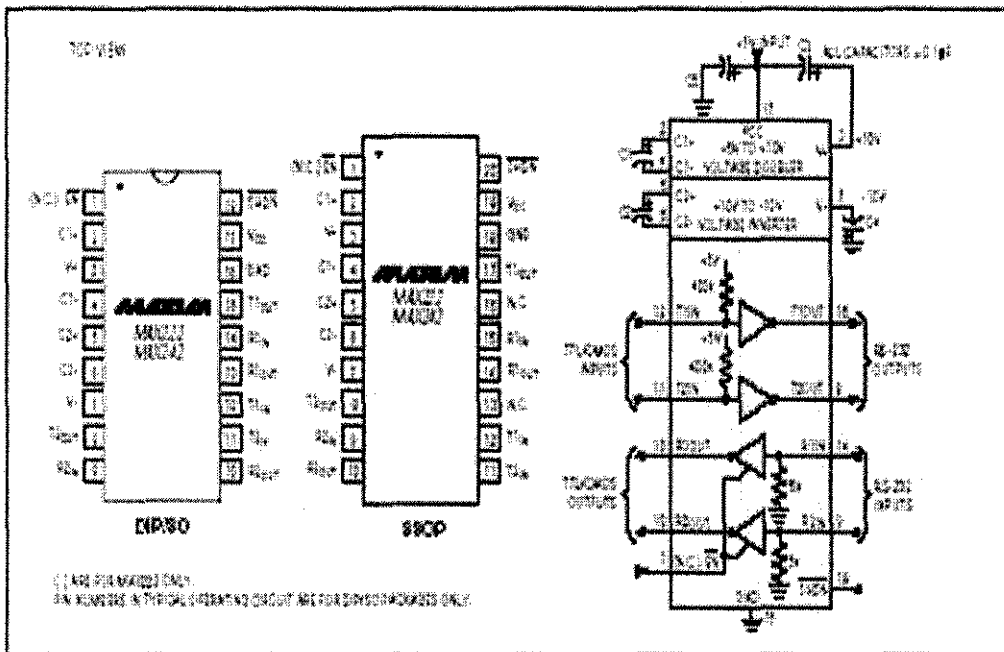


Figure 5. MAX222/MAX242 Pin Configuration and Typical Operating Circuit

# APPENDIX J

