Palestine Polytechnic University
College of Engineering and Technology
Electrical Engineering Department

Graduation Project

Computerized Queuing System for Al_Ahli Clinics

Project Team

Fatima Nabeel Amwas   Muna Adel Al_Hanini

Project Supervisor

Eng. Elayan AbuGarbyeh

A graduation project Submitted to the Department of Electrical and Computer Engineering in the College of Engineering and Technology

Hebron – Palestine
2007/2008

جامعة بوليتكنك فلسطين

الخليل – فلسطين

كلية الهندسة والتكنولوجيا

دائرة الهندسة الكهربائية والحاسوب

Computerized Queuing System for Al_Ahli Clinics

فاطمة نبيل أمواس　　　　منى عادل الهنيني

بناء على نظام كلية الهندسة والتكنولوجيا وإشراف ومتابعة المشرف المباشر على المشروع و موافقة أعضاء اللجنة الممتحنة تم تقديم هذا المشروع إلى دائرة الهندسة الكهربائية والحاسوب وذلك للوفاء بمتطلبات درجة البكالوريوس في الهندسة تخصص أنظمة حاسوب.

توقيع المشرف

....

توقيع اللجنة الممتحنة

....　　................................　　................................

توقيع رئيس الدائرة

....

Dedication

To our parents who
spent nights and days doing their best
to give us the best....

To our children and husbands

To whom who carry candle of science
To light his avenue
Of life ...

To our beloved country Palestine...

To all of our friends…

To our Supervisor Eng. Elayan AbuGardyeh for his supports and advices.

Acknowledgments

To our great supervisor, who offered his best for this project to see light through his instructions and advices, Eng. Elayan AbuGarbyeh with all his kindness and wisdom we thank him.

Great thanks to Eng. Sami Salamin and Khalid Etmaizy for their support, help and bright ideas.

Our great thanks to our mothers, fathers, sisters, brother, friends and every person who offered anything to success this work; we sincerely believe that this work wouldn't exist without his inspiration. A great thanks to our college for his support and help, and any one who help us in our project.

We also thankful for the Al_Ahli hospital engineering team, especially engineer Ala' Ezghayer.

Abstract

This project aims to construct a Computerized Queuing System For Al_Ahli Clinics, which would be able to order the patient entrance to the desired clinic in a quiet arrangable way, this system will replace the non-computerized traditional existing system.

ملخص المشروع

هذا المشروع يهدف إلى بناء نظام الطوابير المحوسب للعيادات الخارجية في مستشفى الأهلي لكي يكون قادرا على تنظيم دخول المرضى إلى العيادة المطلوبة بطريقة هادئة ومنتظمة,هذا النظام سوف يكون بديلا عن النظام التقليدي غير المحوسب المستخدم حاليا.

Table of Contents

Chapter Three: Project Conceptual Design

Chapter Four: Detailed Technical Project Design

Chapter Five: Software System Design

Chapter Six: System Implementation and Testing

List of Figure

List of Tables

1

Introduction

1.1 General idea about the project and its importance

1.2 Project Objective

1.3 Literature Review

1.4 Time Plane /Project Schedule

1.5 Estimated Cost and Budget Breakdown

1.6 Project Risk Management

1.7 Report Contents (Road map)

Chapter One
Introduction

This chapter introduces the general idea of the project, its importance and discusses some of the related projects.

1.1 General Idea about the Project and its Importance

The project came as an idea from Al_Ahli Hospital so it basically depends on the user demand and how to satisfy him. The project is in general a computerized stand alone system that displays the patient number on a 7-segment display accompanied with a simple sound in order to arrange the patient entrance to the desired clinic.

The project consists of two parts: the first one is connecting the Network PC with the system in order to displays the patients' total number inside the desired clinic, this is done using Visual Basic.NET. The second part is displaying the patient number for both the doctor and the patients in the reception so as the patient hears a soft sound, checks if his number appeared and as a result enter the clinic.

1.2 Project Objective

The objectives of the project are:

1. Create an organized way for the patient's entrance to the clinic.
2. To prevent the noise that occurs when calling the patient by his name.
3. Organize the patients order in a fair way.
4. Give the doctor a clear view of the patients' number, so he/she can manage his/her time to serving as much as he can.
5. Give the registration nurse an indication about the number of served patients.

1.3 Literature Review

The queuing system has been used in some of the companies such as Jawwal Company.

By visiting Jawwal Company there have been found a lot of deference's between this system and their.

Jawwal Queuing system is done by using a ticket provider which gives the customer his/her number automatically, then when his/her turn comes he/she saw the same number in the 7-segmant display hanged above the employee. Their system depends on the personal computer and there is no need for any microcontroller.

In this system we plane to use Keypad and Microcontroller instead of the personal computer for every clinic and one personal computer for the registration and network communication, so as to reduce the cost.

1.4 Time Plane /Project Schedule

The project activities depend on each other, so the task durations and dependencies are as the following:

T1: Preparing the project: find the suitable project by searching the internet and the library then asking an advisor to initialize the project, as the project been suggested by the user the challenge was to get the approval to take it, then prepare the group and evaluate the project tasks cost and levels.

T2: Understand the problem: find the requirements, the constraints. This is done by meeting the user and preparing a certain questions to ask him for in order to get the best project understanding.

T3: The project searching and analysis: analysis the project and allocate information and data about the project levels and sublevels, tasks and subtasks.

T4: The project requirements analysis: the project has many types of equipment that must be provided and explained in order to implement the final project and achieve the system requirements. The system has a hardware and software requirements which must be achieved through the simulation and final presentation.

T5: Introduction to project and study the 18F4520 PIC microcontroller system.

T6: Study and find the type of displays and the keypad that want to be used and other hardware required.

T7: Theoretical background about the system. Find the hypothesis and study environment.

T8: Design concepts, modeling the system, design the block diagram and find the design options.

T9: Writing the software. Draw the flowcharts, write the algorithms and the code listing.

T10: Implementation then testing the system: the project will be tested and implemented to insure that the system and user requirements levels are achieved or not, to adjust the problems and errors in the system to maintain it, then try to test and execute it again until it works in the best way.

T11: Reanalyze and re-implement the system if any thing goes wrong.

T12: Final Project and presentation: as a result the final project will be implemented completely without any problem to meet the objectives.

T13: Writing the documentation: the writing begins from the first step to the last one in parallel.

1.4.1 Timeline Chart:

The time chart shows all the project tasks, the duration of each task and the concurrency between the tasks.

The following two tables show the timeline for the first semester table 1-1, and the second semester table 1-2

Table 1-1: Project Activity Bar Chart (First Semester)

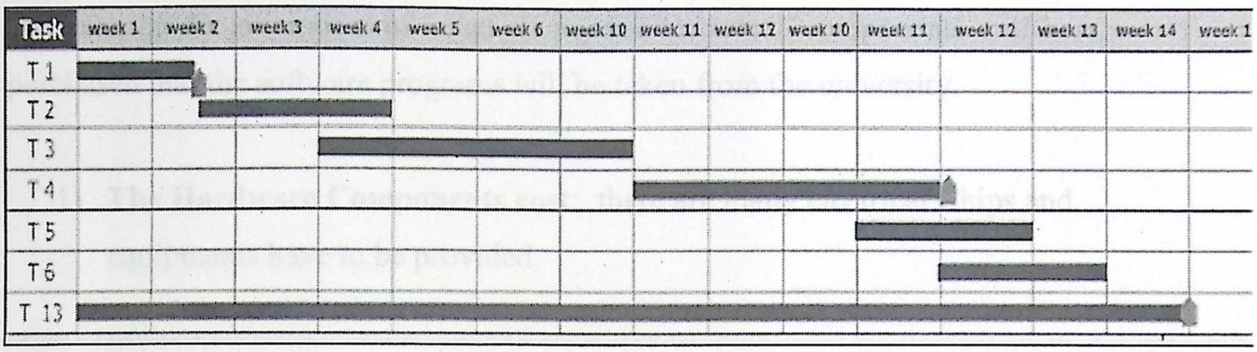
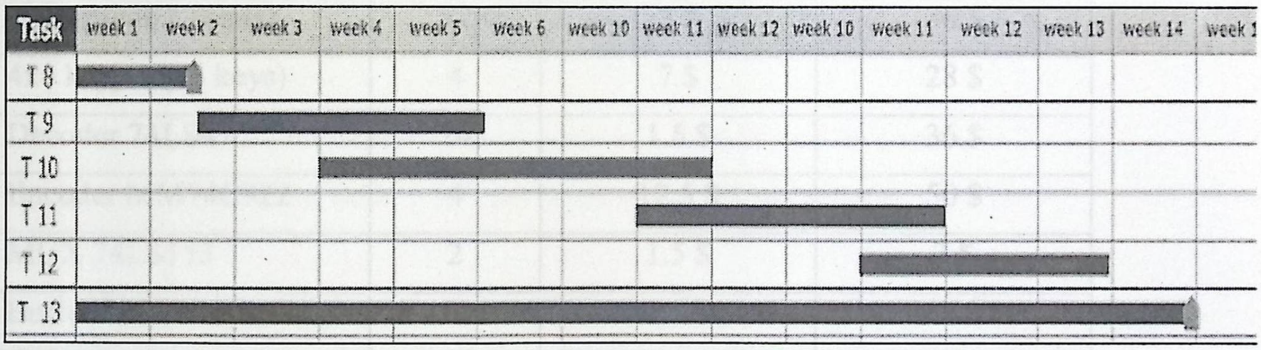


Table 1-2: Project Activity Bar Chart (Second Semester)



As seen from both timeline that there are some dependency between some tasks, such as the relation between the first two tasks (after looking at the suitable project and deciding what to do then the problem understanding start). And also there is a concurrency between the second and the third task (Understand the problem and the project searching and analysis can be worked simultaneously).

1.5 Estimated Cost and Budget Breakdown

The project need both of hardware equipments and software programs that runs on the microcontroller, so all needed electronic components will be purchased and the software programs will be taken from the university.

1) **The Hardware Components cost**: there are many electrical Chips and equipments have to be provided.

Table 1-3:Tthe Project Hardware Cost

Components	*Number*	*Cost*	*Total cost*
PIC18F4520	1	17.5 $	17.5 $
7-Segment(2-digits)	12	3 $	36 $
4x4 keypad(16 keys)	4	7 $	28 $
Decoder 74LS47	24	1.5 $	36 $
Encoder MM74C922	4	12.5 $	50 $
MUX 74LS153	2	1.5 $	3 $
Buffer 74LS244	12	1.5 $	18 $
Decoder 74LS138	2	1 $	2 $
MAX232	1	2 $	2 $
Serial Port Cable(9-pins)	1	2.5 $	2.5 $
Reset Switch	1	0.5 $	0.5 $
Capacitors 1uF	11	0.25 $	2.75 $
Capacitors0.1uF	4	0.25 $	1 $
Resistors (150ohm)	24	0.25 $	6 $
Resistors (1-100)kohm	5	0.25 $	1.25 $
Diodes	2	1 $	2 $
Transistors	2	1 $	2 $
Speaker	1	4 $	4 $

Bases	62	(1.25 - 1.5) $	87.5 $
Wire-rapping board	1	17.5 $	17.5 $
Inverter 7404	1	0.5 $	0.5 $
OR Gate 7432	1	0.5 $	0.5 $
Total Cost		320.5 $	

2) Human Effort Cost

The system group consists of two undergraduate students:

Fatima Amwas
Muna Al_Hanini

The group work five days at week and take 6$ per days so the estimated work cost for each is 30$ per week and 120$ per month.

The total cost contains the hardware equipments, software programs and human effort is approximately (1280.5$).

1.6 Project Risk Management

The project risk management to avoid the project from being suddenly threatened by occurred risk\problem that might terminate the project, so by studying the project from its all site the project came with some risk which can be avoided in particular case.

There are three categories of risk which are:

1. Project risk: the risks that affect the project schedule or resources which are:
    a) One or more essential hardware for the project will not be delivered on schedule.
    b) Delivering the project will be delayed.
2. Product risk: the risks that affect the quality or performance of the software and the hardware for the project which are:
    a) Large number of requirements changed than anticipated
    b) The ability to improve the system will be difficult.
    c) The database size is underestimated.

To avoid these risks and managing them, the following was done:

1. Looking in the market for the needed components before starting the project.
2. Make a deal with the hospital in order not to give the project to other group and not to buy it from other company until the project delivered to them in the schedule time.
3. Understand the project from all its different phases so no sudden changes occurred.
4. Save the written works in more than one resources (Computer, flash memory, Internet Email, and CD Rome)

1.7 Report Contents (Road map)

The following is a brief description of the topics that are covered in each chapter.

Chapter 2: Theoretical Background

This chapter talks in more details about the basic component used in the project, discuss the hypothesis, show the project integrity and theoretical background about the system components.

Chapter 3: Project Conceptual Design

This chapter describes in details the design concepts, introduces project objectives, shows the general block diagram of the system and explains how the system will works, discuss design options and justify those chosen for the project. Show how the system interacts with the surrounding environment.

Chapter 4: Detailed Technical Project Design

This chapter presents detailed description of the project phases, views the subsystem design, shows the schematic diagram and discusses the user system interface.

Chapter 5: Software

This chapter handles the software related to the system, depicts flowcharts about system operation and the code listing.

Chapter Six: System Implementation and Testing

This chapter includes the implementation phases with the testing of these phase. General hardware and software component are tested and shown in this chapter.

Chapter Seven: Conclusion and Future Work

This chapter provides the conclusions, suggestion and developments for future work.

2

Theoretical Background

2.1 Hypothesis, Hardware, and Software Related to the Project.

2.2 Project Integrity.

2.3 Theoretical Background about Project Components.

Chapter Two

Theoretical Background

This chapter focuses on system requirements, theories that are related to this system and components used in the system.

2.1 Hypothesis, Hardware, and Software Related to the Project

Hypotheses

After studying the project some modifications were added to the project to make it an efficient system. The project is design to serve four clinics in Al_Ahli Hospital as a test.

Hardware

The project requires an electronic devises such as the microcontroller, interfacing ICs, and contain hardware devices such as the 7-segment display and 4×4 keypad.

It also needs some interconnectivity ports such as the serial port between the hospital network PC and the 7-segment display.

Software

The project needs some computer programs, data structures, and related documentation in order to make the modeling, simulation, implementation, and programming, and testing for the project.

The needed programming languages are:

- C language: to programming PIC18f4520.
- Visual Basic language: to transfer data from hospital network computer to 7-segment display. A bridge between these two hardwires is needed. Visual Basic language which has the ability to send the data to the serial port in the network computer then to seven-segment display would do the job.

The documentation programs needed are:

- The Microsoft Office Package.
- The SmartDraw Program.

2.2 Project Integrity

The project is an integrated system that serves the clinic in the hospital as a whole. The doctor will have a clear vision for the patients numbers by providing him/her with two different displays one for the total number of patients and other for the served number. The assistant nurse will have an easy way to manage the patient's entrance by just pressing the suitable button on the keypad. And the patients will have a comfortable way for waiting their order. As a result the system will make the clinic of the hospital more desired for the patients because it offers a developed, comfortable, easy and fair way of patients serving.

This project is designed to be worked on four clinic as a test. In the future it will be completed to conclude all the clinic rooms in Al_Ahli hospital which are twelve, the design for the 12 clinic is done in this project but the implementation is done for four clinic as mentioned because of the high components cost.

2.3 Theoretical Background about Project Components

The project has two inputs: first input comes from network PC in order to display the patients' total number in the desired clinic on a small 7-segment at the doctor room.

The second input comes from four 4×4 keypads, which is displayed on large 7-segment above the entrance door as well as a small 7-segment and a sound inside the clinic.

The basic unit is the controlling unit which controls all the system functions.

The following sections will give an explanation of each component (hardware device) that will be used in this system.

2.3.1 PIC 18F4520 Microcontroller

2.3.1.1 Introduction to Microcontroller

A controller is used to control some process or aspect of the environment. At one time, controllers were built exclusively from logic components, and were usually large, heavy boxes (before this, were the even bigger, more complex analog). Later on, microprocessors were used and the entire controller could fit on a small circuit board. This is still common –the user can find many good controllers powered by one of the many common microprocessors (including Intel 8088, Motorola 6809, and others).

As the process of miniaturization (small size) continued, all of the components needed for a controller were built right onto one chip. A one chip computer or microcontroller was born. A microcontroller is a high integrated chip which includes, on one chip, all or most of the parts needed for a controller. The microcontroller could be called a "one-chip solution". It typically includes:

- CPU (central processing unit)
- RAM (Random Access Memory)

- EPROM/PROM/ROM (Erasable Programmable Read Only Memory)
- I/O (input/output)
- DAC \ ADC ports.
- Interrupt controller.

By only including the features specific to the task (control), cost is relatively low. A typical microcontroller has bit manipulation instructions, easy and direct access to I/O (input/output), and quick and efficient interrupt processing. Microcontrollers are a "one-chip solution" which drastically reduces parts count and design costs.

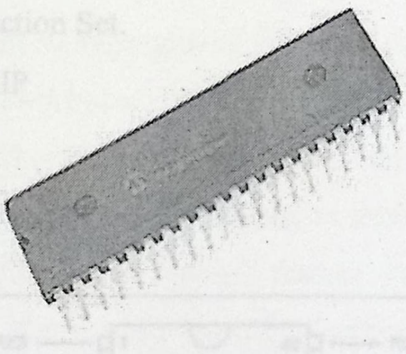


Figure 2-1: PIC 18F4520 Microcontroller[1]

2.3.1.2 PIC 18F4520 Features:

- DC - 40 MHz Operating Frequency.
- 32 K Program Memories (Bytes).
- 16384 Program Memory (Instructions).
- 1536 Data Memory (Bytes).
- 256 Data EEPROM Memory (Bytes).
- 18 Interrupt Sources.
- A, B, C, D, E I/O Ports.
- Master Synchronous Serial Port (MSSP) module, Addressable USART Serial Communications.
- Programmable Low Voltage Detect.
- Programmable Brown-out Reset.
- 75 Instruction Set.
- 40-pin DIP

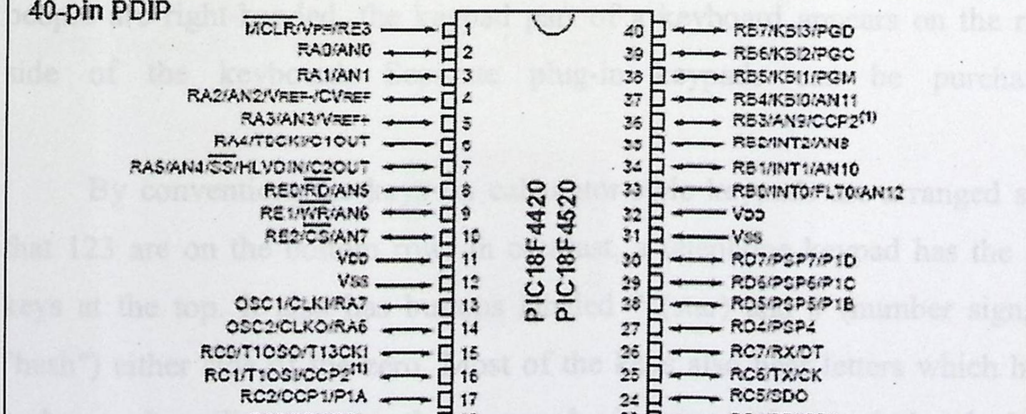


Figure 2-2: PIC 18F4520 Pin Layout[1]

2.3.2 Keypad

2.3.2.1 Introduction

A keypad (or "numeric keypad") specifically refers to a set of buttons similar to an alphanumeric keyboard that bears numbers and possibly other mathematical features.

The keypad of a calculator contains the digits 0 through 9, together with the four arithmetic operations, the decimal point and other more advanced functions.

The term keypad can also refer to the part of a computer keyboard that contains a calculator-style arrangement of buttons - many of them duplicating existing keys on the main keyboard - allowing efficient entry of numerical data. On most laptops, special function keys have to be depressed to turn part of the alphabetical keyboard into a numerical keypad as there is insufficient space to allow a keypad to be built into the laptop's chassis. Presumably because most people are right-handed, the keypad part of a keyboard appears on the right side of the keyboard. Separate plug-in keypads can be purchased.

By convention, the keys on calculator-style keypads are arranged such that 123 are on the bottom row. In contrast, a telephone keypad has the 123 keys at the top. It also has buttons labeled * (star) and # (number sign, or "hash") either side of the zero. Most of the keys also bear letters which have had several auxiliary uses, such as remembering area codes or whole telephone numbers.

A keypad can also refer to the series of numbered buttons, similar to a telephone keypad, used as part of a combination lock. This is often used to allow multiple entries to doors, such as that found at the main entrance to some offices.

'Keypad' is a PIC based system for decoding switch matrix type numeric keypads with up to 4 rows and columns. The keypad switch matrix is read, and if a key is pressed, it is converted into an equivalent binary value (0-0fh) for output. Full debounce logic is included to suppress mechanical switch bounce effects.

Output can either be a 4-bit parallel word or a serial clocked output. Both serial and parallel outputs support a "latching" pulse to drive external interface timing.

2.3.2.2 4×4 Keypad

4×4 keypad, this is a standard device with 16 keys connected in a 4x4 matrix, giving the characters 0-9, A-D, * and # symbols.

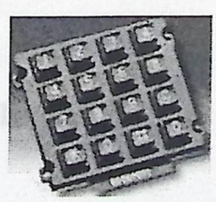


Figure 2-3: 4×4 keypad[2]

2.3.3 7-Segment Display

2.3.3.1 Introduction

One common requirement for many different digital devices is a visual numeric display. Individual LEDs can of course display the binary states of a set of latches or flip-flops. However, we're far more used to thinking and dealing with decimal numbers. To this end, we want a display of some kind that can clearly represent decimal numbers without any requirement of translating binary to decimal.

This requires just seven LEDs (plus an eighth one for the decimal point, if that is needed). A common technique is to use a shaped piece of translucent plastic to operate as a specialized optical fiber, to distribute the light from the LED evenly over a fixed bar shape. The seven bars are laid out as a squared-off figure "8". The result is known as a 7-segment LED.
All 7-segment displays can be seen in a wide range of applications. Clocks, watches, digital instruments, and many household appliances already have such displays.

2.3.3.2 7-Segment Display Layout

The illustration to the right shows the basic layout of the segments in a 7-segment display. The segments themselves are identified with lower-case letters "a" through "g," with segment "a" at the top and then counting clockwise. Segment "g" is the center bar.

Most 7-segment digits also include a decimal point ("dp"), and some also include an extra triangle to turn the decimal point into a comma. This improves readability of large numbers on a calculator, for example. The decimal point is shown here on the right, but some display units put it on the left, or have a decimal point on each side.

In addition, most displays are actually slanted a bit, making them look as if they were in italics. This arrangement allows us to turn one digit upside down and place it next to another, so that the two decimal points look like a colon between the two digits. The technique is commonly used in LED clock displays.

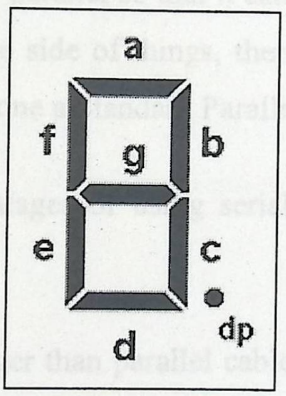


Figure 2-4: 7-Segment Display[3]

There is no automatic advantage of the common-cathode 7-segment unit over the common-anode version, or vice-versa. Each type lends itself to certain applications, configurations, and logic families.

2.3.4 Serial Port

In computing, a serial port is a serial communication physical interface through which information transfers in or out one bit at a time (contrast parallel port). Throughout most of the history of personal computers, data transfer through serial ports connected the computer to devices such as terminals or modems. Mice, keyboards, and other peripheral devices also connected in this way.

In serial I/O technique, data can be transmitted as either current or voltage, the commonly used standard is known as RS-232. This standard governs the physical dimensions of the connectors, the number and configuration of ports and several electrical parameters.

The Serial Port is harder to interface than the Parallel Port. In most cases, any device you connect to the serial port will need the serial transmission converted back to parallel so that it can be used. This can be done using UART. On the software side of things, there are many more registers that you have to attend to than one a Standard Parallel Port(SPP).

So what are the advantages of using serial data transfer rather than parallel

- Serial cable can be longer than parallel cable. The serial port transmits '1' as -3 to -25 volts and a '0' as +3 to +25 volts where as a parallel port transmits a '0' as 0 V and a '1' as 5V. Therefore the serial port can have a maximum swing of 50V compared to the parallel port which has a maximum swing of 5V. Therefore cable loss is not going to be as much of a problem for serial cables than they are for parallel.

- Microcontrollers have proven to be quite popular recently. Many of these have in built SCI (Serial Communication Interfaces) which can be used to talk to the outside world. Serial Communication reduces the pin count of these microcontrollers. Only two pins are commonly used, Transmit Data (TXD) and Receive Data (RXD) compared with at least 8 pins if you use a 8 pit parallel method (also may require a Strobe).

2.3.5 MAX232

2.3.5.1 Introduction

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept (+/-) 30-V inputs. Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels.

2.3.5.2 Features

- Operates from a single +5V power supply with 1.0-µF charge-pump capacitors.
- Contain two transceivers.
- Operates up to 120 Kbit/s.
- Two drivers and two receivers
- (+/-) 30-V input levels.
- Low supply current…8 mA typical.

2.3.5.3 Applications

- TIA/EIA-232-F.
- Battery-powered systems.
- Terminals.
- Modems.
- Computers.

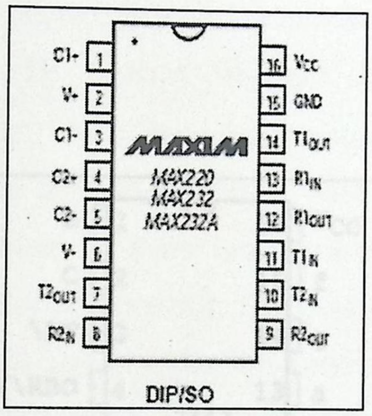


Figure 2-5: MAX232 pin[4]

2.3.6 7447 Decoder

2.3.6.1 General Description

The 7447 decoder – DM74LS47 decoder – is BCD to 7-segment decoder/driver with open-collector outputs.
The 7447 decoder accepts four line of BCD (8421) input data, generates their complements internally and decodes the data with seven AND/OR gates having open-collector outputs to drive indicator segments directly. Each segment output is guaranteed to sink 24 mA in the ON (LOW) state and withstand 15 V

in the OFF (HIGH) state with a maximum leakage current of 250 μA. Auxiliary inputs provided blanking, lamp test and cascadable zero-suppression functions.

2.3.6.2 Features

- Open-collector outputs.
- Drive indicator segments directly.
- Lamp test input.

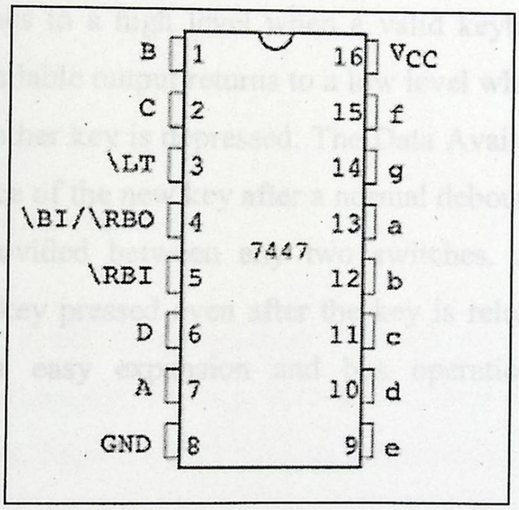


Figure 2-6: 7447 Decoder Pin[5]

2.3.7 MM74C922 Encoder

2.3.7.1 General Description

The MM74C922 and MM74C923 CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 kohm on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this two-key roll-over is provided between any two switches. An internal register remembers the last key pressed even after the key is released. The 3-STATE outputs provide for easy expansion and bus operation and are LPTTL compatible.

2.3.7.2 Features

- 50 kohm maximum switch on resistance
- On or off chip clock
- On-chip row pull-up devices
- 2 key roll-over
- Keybounce elimination with single capacitor
- Last key register at outputs
- Wide supply range: 3V to 15V

- Low power consumption

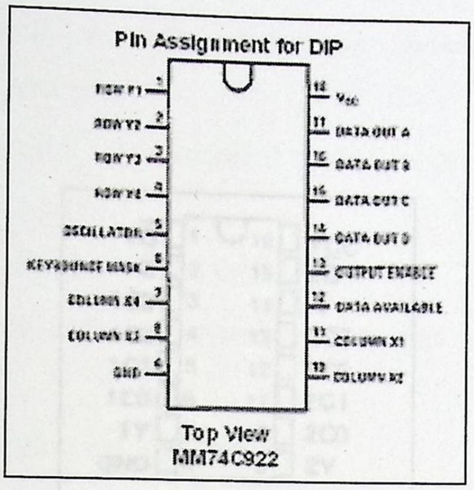


Figure 2-7: 74922 Encoder Pin[6]

2.3.8 74153 MUX

2.3.8.1 General Description

Each of these data selectors/multiplexers contains inverters and drivers to supply fully complementary, on-chip, binary decoding data selection to the AND-OR-invert gates. Separate strobe inputs are provided for each of the two four-line sections.

2.3.8.2 Features

- Permits multiplexing from N lines to 1 line
- Performs parallel-to-serial conversion
- Strobe (enable) line provided for cascading (N lines to n lines)
- High fan-out, low-impedance, totem-pole outputs
- Typical average propagation delay times

  From data 11 ns

  From strobe 18 ns

From select 20 ns

- Typical power dissipation 170 mW

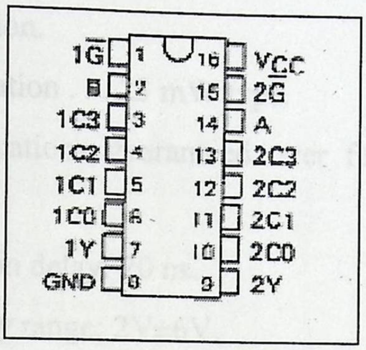


Figure 2-8: 74153 Mux Pin[7]

2.3.9 74138 Decoder

2.3.9.1 General Description

The 74LS138 decodes one-of-eight lines, based upon the conditions at the three binary select inputs and the three enable inputs. Two active-low and one active-high enable inputs reduce the need for external gates or inverters when expanding. A 24-line decoder can be implemented with no external inverters, and 32-line decoder requires only one inverter. An enable input can be used as a data input for demultiplexing applications.

This decoder/demultiplexer features fully buffered inputs, presenting only one normalized load to its driving circuit. All inputs are clamped with high performance Schottky diodes to suppress line-ringing and simplify system design.

2.3.9.2 Features

- Designed specifically for high speed:Memory decoders Data transmission systems.
- 3- to 8-line decoder incorporates 3 enable inputs to simplify cascading and/or data reception.
- Low power dissipation . . . 23 mW type.
- Switching specifications guaranteed over full temperature and VCC range.
- Typical propagation delay: 20 ns.
- Wide power supply range: 2V±6V.
- Low input current: 1 mA maximum.

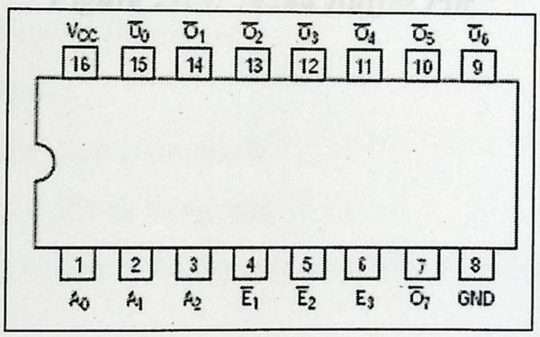


Figure 2-9: 74138 Decoder pin[8]

3.10 74244 Buffer

2.3.10.1 General Description

The SN74LS244 is Octal Buffers and Line Driver designed to be employed as memory address driver, clock driver and bus-oriented transmitter/receiver which provide improved PC board density.

2.3.10.2 Features

- Hysteresis at Inputs to Improve Noise Margins.
- 3-State Outputs Drive Bus Lines or Buffer Memory Address Registers.
- Input Clamp Diodes Limit High-Speed Termination Effects.

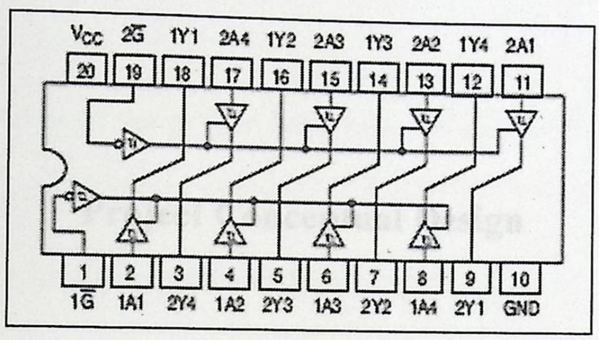


Figure 2-10: 74244 Buffer Pin[9]

3

Project Conceptual Design

Chapter Three
Project Conceptual Design

3.1 Detailed Project Objective

The objectives of the project have been determined from the first place. To understand the project objectives here is a detailed description for them.

The following are the objectives of the project:

1. Create an organized way for the patient's entrance to the clinic. The patient first register his/her name on the reception office from which the registration number is a sequence number depending on the registered patients, the patient take his/her registration number and wait in front of the desired clinic until he/she see his/her number on the display.

2. Produce a sound attend patients for their turn. To prevent the noise that occurs when calling the patient by his name. This objective rest the nurse from calling the patient by his name many times until he/she respond, it also allows a comfortable and quiet way for the whole hospital environment.

3. Organize the patients order in a fair way. The sequence number for each patient is unique so no two patients could have the same number, on the other hand this numbers are ordered according to the patients' reservation; the first one takes number one, the second takes the second number and so on.

4. Give the doctor a clear vision for the patient number, so he can manage his time to serve as much as he can. By supporting him with two different displays one for the total number and other for the next patient to be served.

5. Give the assistance an easy way to manage the patients' entrance by just pressing the suitable button on the keypad. The assistance could press one of the following buttons on the keypad:

   a) Next: order the next patient in order.
   b) Back: to decrease the display number in case it increased by mistake.
   c) Clear: to clear both displays and reset them to zero.
   d) No answer: to store the no responding patient number in the memory in order to be recalled later.
   e) Recall no answer: when the doctor needs to recall no answer patient the system allows the user to recall the no answering patient by entering the number of him directly or restore it from the memory.
   f) Enter: ensure the input key.
   g) Numeric keypad: to enter the exact number of the patient order to enter the clinic room

3.2 Design Options

This project has several design options, such as:

2) Using Microprocessor unit and memories.

3) Using microcontroller unit.

This project chooses the third option which is using microcontroller unit (PIC18f4520) and the option of choosing PIC18F4520 refers to:

• High computational performance.
• Economical price.
• High endurance.
• Enhanced Flash program memory.

Special Features:

• C compiler optimized architecture: Optional extended instruction set designed to optimize re-entrant code.
• 100,000 erase/write cycle Enhanced Flash program memory typical.
• 1,000,000 erase/write cycle Data EEPROM memory typical.
• Flash/Data EEPROM Retention: 100 years typical.
• Self-programmable under software control.
• Priority levels for interrupts.
• Wide operating voltage range: 2.0V to 5.5V.

Other components used in this project are available and easy to use.

3.3 Design Realization Approach

3.3.1 Implementation

Implementation of the system will not be performed until making sure that every thing is working efficiently. After designing, simulation and testing the project the implementation stage start by connecting the system components and interfacing them then programming the over all system.

3.3.2 Modeling

In order to understand the system clearly graphical representation is made. There are many modeling graph one of them is relationship graph which show the primary data objects to be processed by the system and the relations between these objects.

The following figure 3-1 shows that the system has five objects with their relations

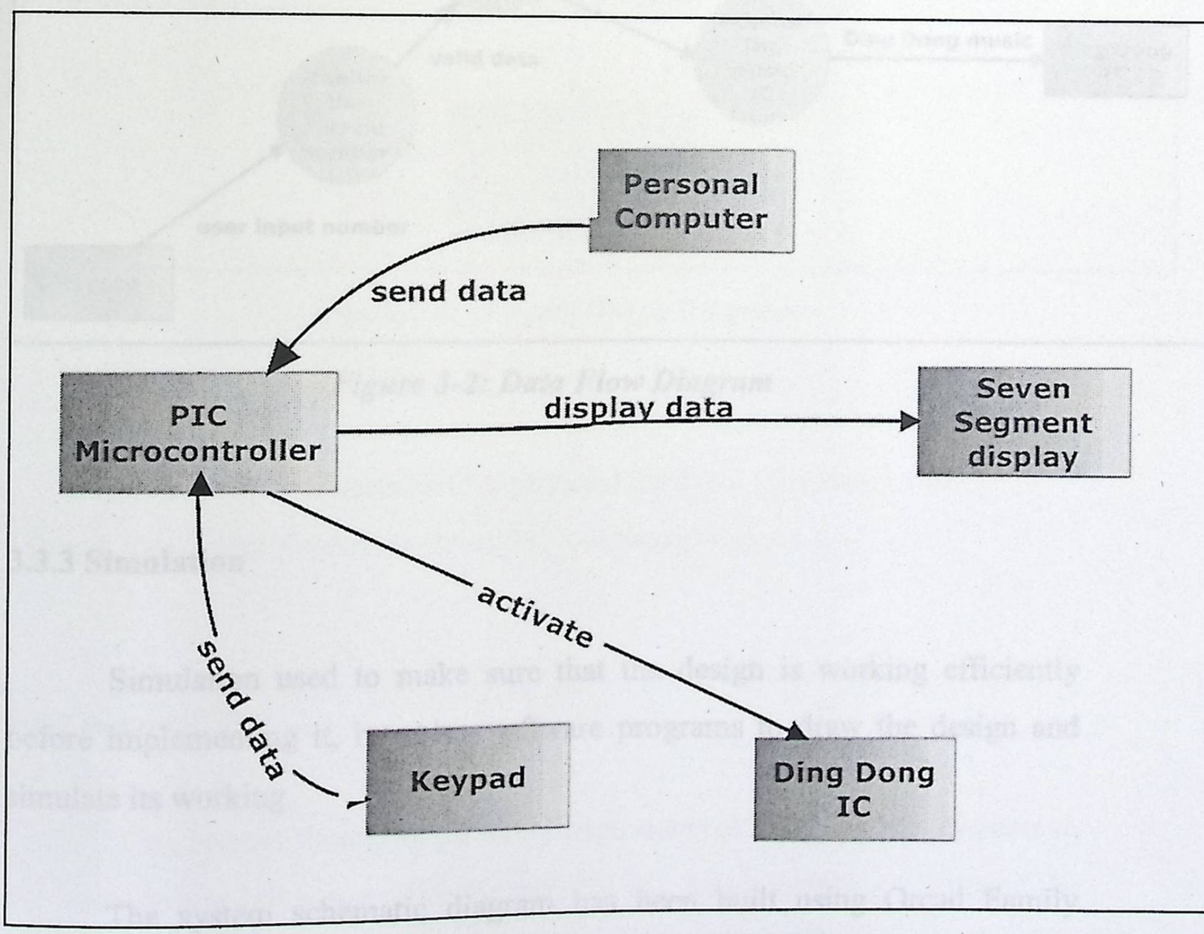


Figure 3-1: System Relationship Modeling Diagram

the project the implementation stage start by connecting the system components and interfacing them then programming the over all system.

3.3.2 Modeling

In order to understand the system clearly graphical representation is made. There are many modeling graph one of them is relationship graph which show the primary data objects to be processed by the system and the relations between these objects.

The following figure 3-1 shows that the system has five objects with their relations

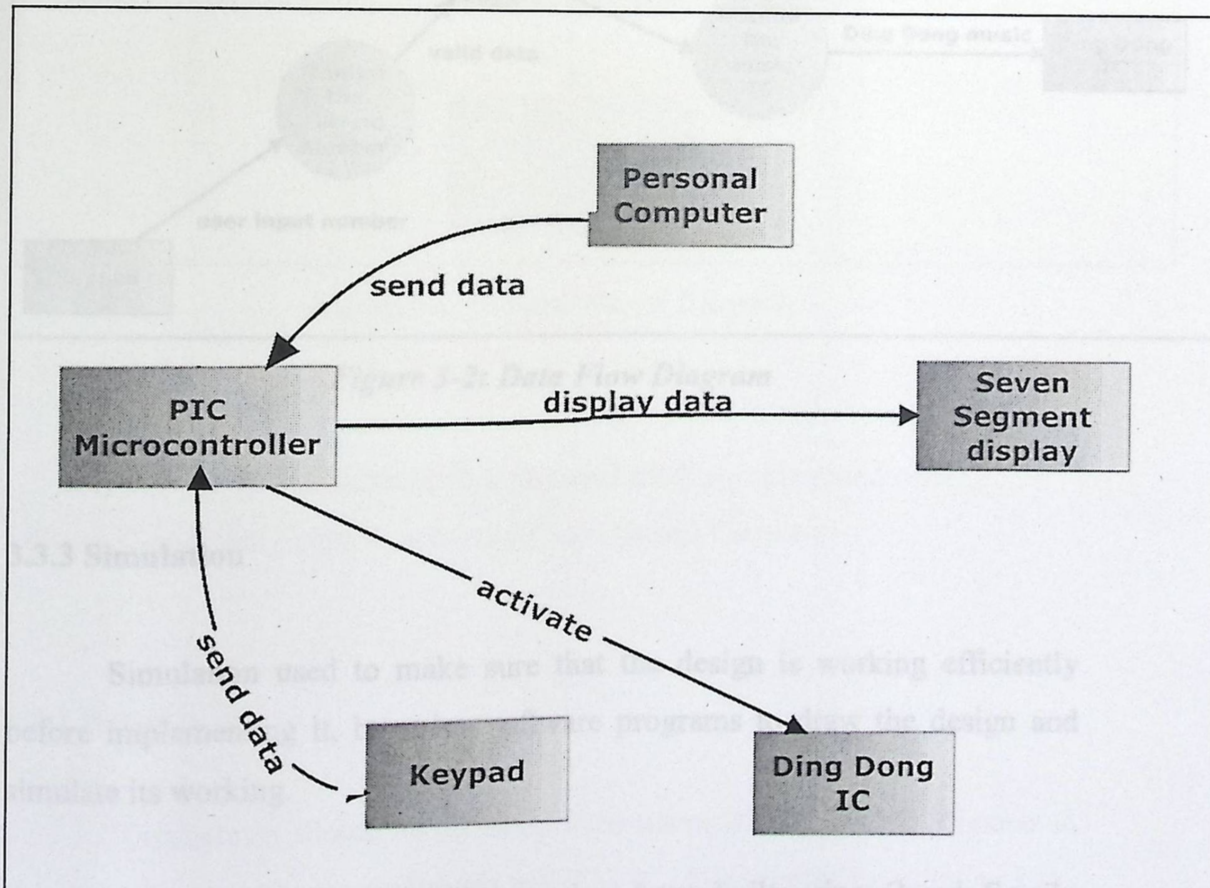


Figure 3-1: System Relationship Modeling Diagram

To develop model for the information and functional domain at the same time the Data Flow Diagram DFD is presented. The DFD is refined into greater level of details and analyst. Figure 3-2 shows the data flow diagram.

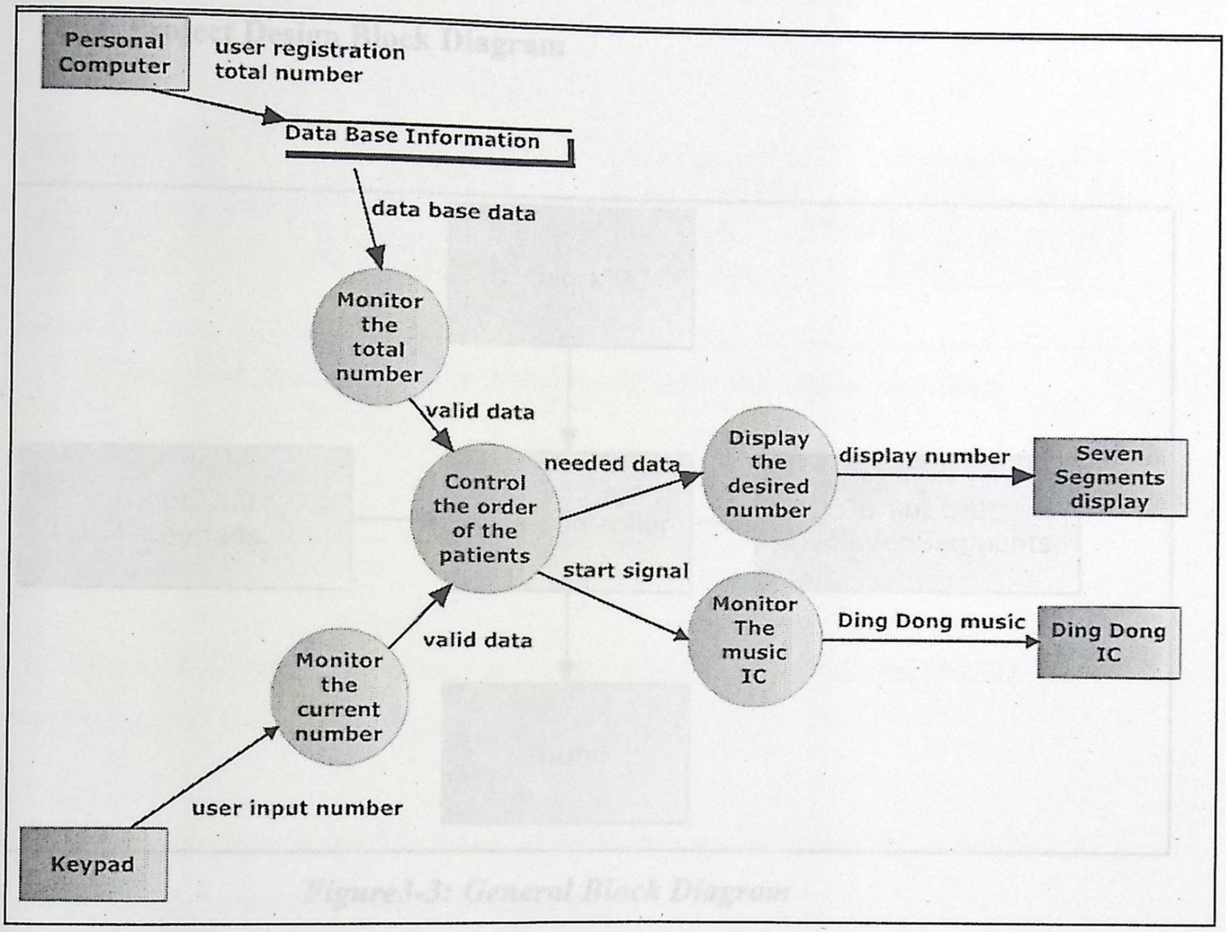


Figure 3-2: Data Flow Diagram

3.3.3 Simulation

Simulation used to make sure that the design is working efficiently before implementing it, by using software programs to draw the design and simulate its working.

The system schematic diagram has been built using Orcad Family Release 9.2 in the Capture program.

In the next chapter the simulation details will be explained.

3.4 Project Design Block Diagram

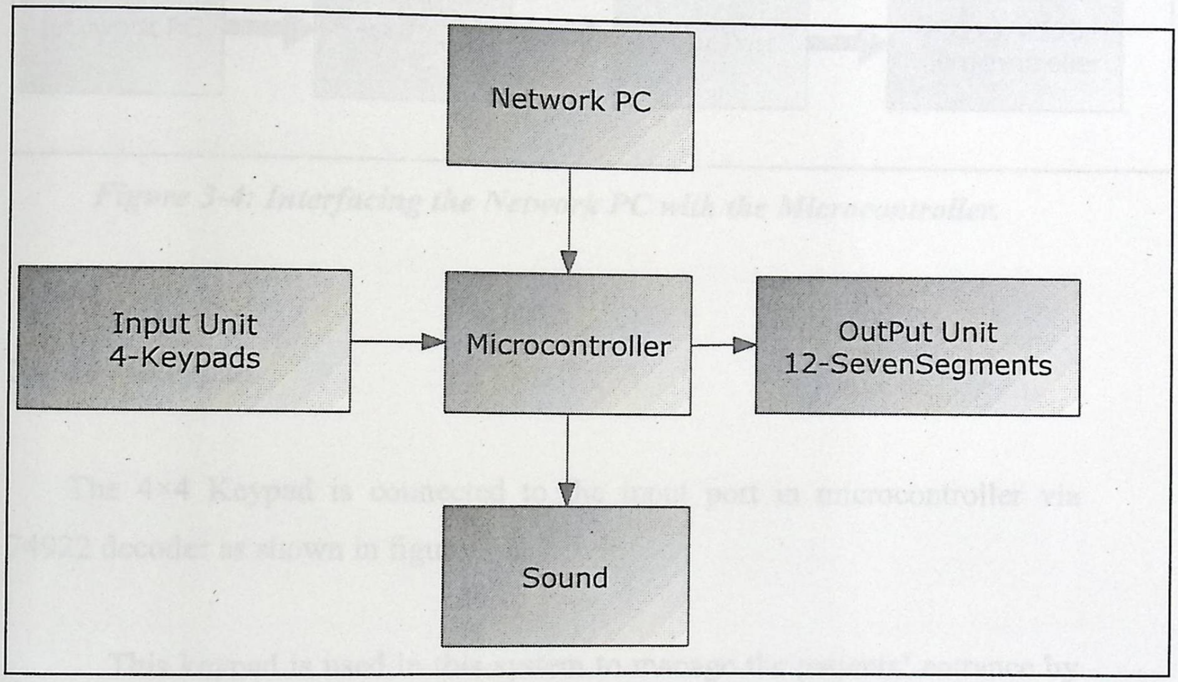


Figure3-3: General Block Diagram

This system consists of four physical modules (microcontroller, keypad, 7-segment display, network PC and Sound Circuit).

3.4.1 Network PC

This system should be provided from network PC. The PC connects to the input port in microcontroller via max232 IC, this interfacing to change the output voltage to +5volt instead of the 12 PC voltages. This part is the first input in this system which comes from hospital network PC in order to display the patients' total number in the desired clinic at the doctor room; display the total patient number who wants to be served, this outputs on small seven-segment display.

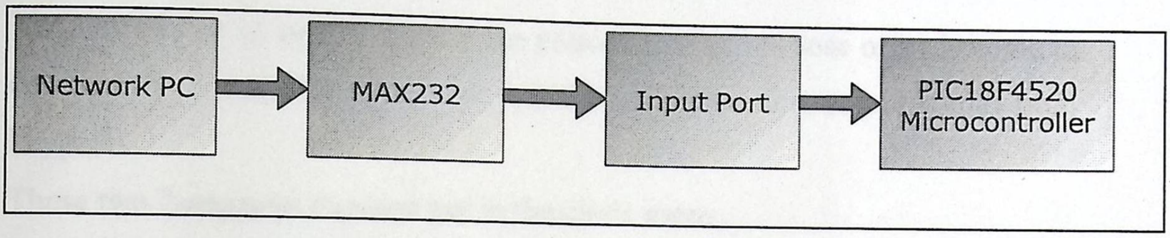


Figure 3-4: Interfacing the Network PC with the Microcontroller.

3.4.2 4×4Keypad

The 4×4 Keypad is connected to the input port in microcontroller via 74922 decoder as shown in figure 3-5.

This keypad is used in this system to manage the patients' entrance by just pressing the suitable button on the keypad.

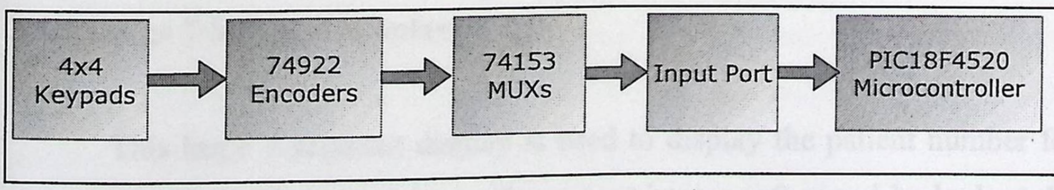


Figure 3-5: Interfacing the keypad with the Microcontroller.

3.4.3 Small 7-Segment Display

This system use two small 7-segment displays, the first one is used to display the patients' total number in the desired clinic at the doctor room; display the total patient number who wants to be served, this output comes from network PC.

The second one is used to display the patient number - whose order is come to entrance the room - for the doctor and the assistance, this output comes from keypad.

These two 7-segment displays put in the clinic room.

The display is connected to the Microcontroller via7447 decoder, as shown in Figure 3-6.

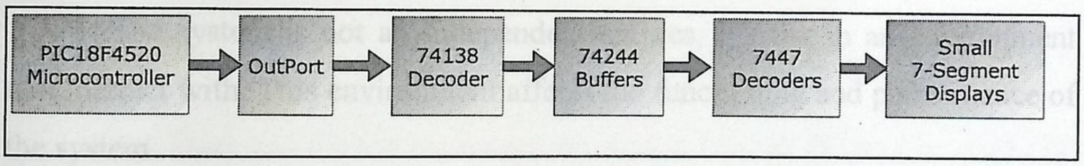


Figure 3-6: Interfacing the Small 7-Segment Display with the Microcontroller

3.4.4 Large 7-Segment Display

This large 7-segment display is used to display the patient number for the patients in the reception, so as the patient hear a soft sound he look at the display screen and check if his number appeared. This display is place above the entrance door of clinic room.

The display is connected to the Microcontroller via7447 decoder, as shown in figure3-7.

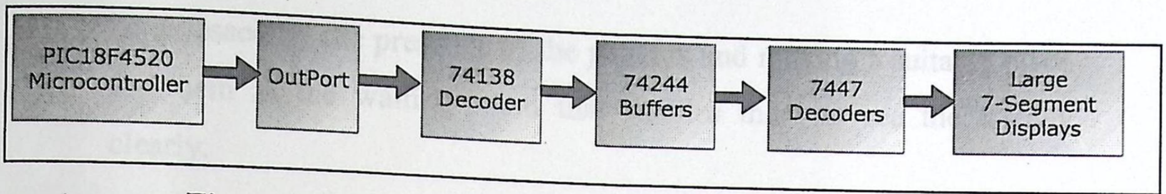


Figure 3-7: Interfacing the large 7-Segment Display with the Microcontroller.

4.5 Microcontroller (PIC18F4520)

This project use PIC18F4520 that is responsible for controlling and processing operations of the system. It performs the functions in the system.

3.5 Project Interaction with the Surrounding Environments

The system is not an independent entities, it exist in an environment that interact with. This environment affects the functioning and performance of the system.

The system environment for the project is other systems that might be incorporation the hospital which are: the power supply system and the security system; they are defined as a local environment. The people that interact with the system may also be considered as a local environment. All other systems that are outside the system location is an overall environment which are the hospital building itself, the outside building and the town.

The study of the environment came as a result of two main reasons:

1. The system intends to make changes in the environment through ordering the patient. The correct functioning of the system can therefore

be assessed by the presence of the patients and making a suitable place for them in the waiting room that enables them to see the display clearly.

2. The functioning of the system can be affected by changes in its environment difficult ways to predict. For example a virus to the computer network may affect the system to work differently in undesirable way, or power fail which may reset the system.

If the hospital environment is not properly understood, system may not meet business needs and may bee rejected by the user and hospital manager. These human and organization affect the system in the following manner:

Process change: the system requires changes to the work processes in the environment. That is because the assistant and the register secretary will take small training lessons in order to use the system.

Job changes: the system causes the user to change the way he work. As what happen to the assistant instead of calling the patients she has just to press some buttons.

Organizational changes: the assistant may lost her job because the doctor can use the system easily with out affecting his main job. So the doctor will take more power than he used to have.

4

Detailed Technical Project Design

4.1 Detailed Description of the Program Phases

4.2 Subsystem Detailed Design

4.3 Over all System Design

4.4 User -System Interface

Chapter Four

Detailed Technical Project Design

4.1 Detailed Description of the Program Phases

The system goes through three main phases: the input, the processing and the output. These phases are explained as the following:

4.1.1 The Input Phase

The system has two different inputs from two different places:

1) The first input comes from the serial port: the secretary who works on the network computer opens the Clinics Table program, resets it and start to register the patients in their order to the desired clinic. The data after that goes to the Access database and then to the serial port in the PC. The serial data then goes to the input of MAX232 which convert the input PC into acceptable output to the PIC. The serial data enters the PIC through the RX pin.

2) The second input comes from the assistant nurse: the nurse switch on the system, reset the data by pressing clear button on the keypad then enter the first patient number, the second, the third until all the patients been served. The data form the keypad is taken through eight line _four rows and other four columns_ to the inputs of the MM74C922 encoder. The encoder outputs are then connecter to the PIC through four lines to portA and one line to the interrupt pin 33.

4.1.2 The Processing Phase

The main processing operations occurred in the microcontroller. This system uses PIC 18F4520 Microcontroller. The data from the serial port are come through pin 25 (TX) and pin26 (RX), this data are processed in order to be shown on the desired small 7-segment display inside the clinics as the total number for the patients.

The other data from the keypad are comes as input to the PIC through four lines from portA (RA0, RA1, RA2, RA3) and four data available line to three interrupts pins (INT0, INT1, INT2). Then the microcontroller processes this data to show the particular patient in the large outside the clinic 7-segment display and the small 7-segment display inside the doctor clinic.

4.1.3 The Output Phase

The output phase start from taking the data from the output port of the PIC to the 7-segment displays.

There are three 7-segment displays in every clinic; two small 7-segment displays inside the clinic one for displaying the total number and the other for displaying the patient number, the third 7-segment display is a large one hanged above the clinic door so the patients can see, it displays the number to the patient who has the turn to enter the clinic. These 7-segment displays are connected through by the 7447 decoder which take four output lines from the one port and send seven output lines to the 7-segment display.

4.2 Subsystem Detailed Design

4.2.1 The Serial Port to the PIC Interface Circuit

The serial port is connected to the PIC using the MAX232 as in the following figure 4-1.

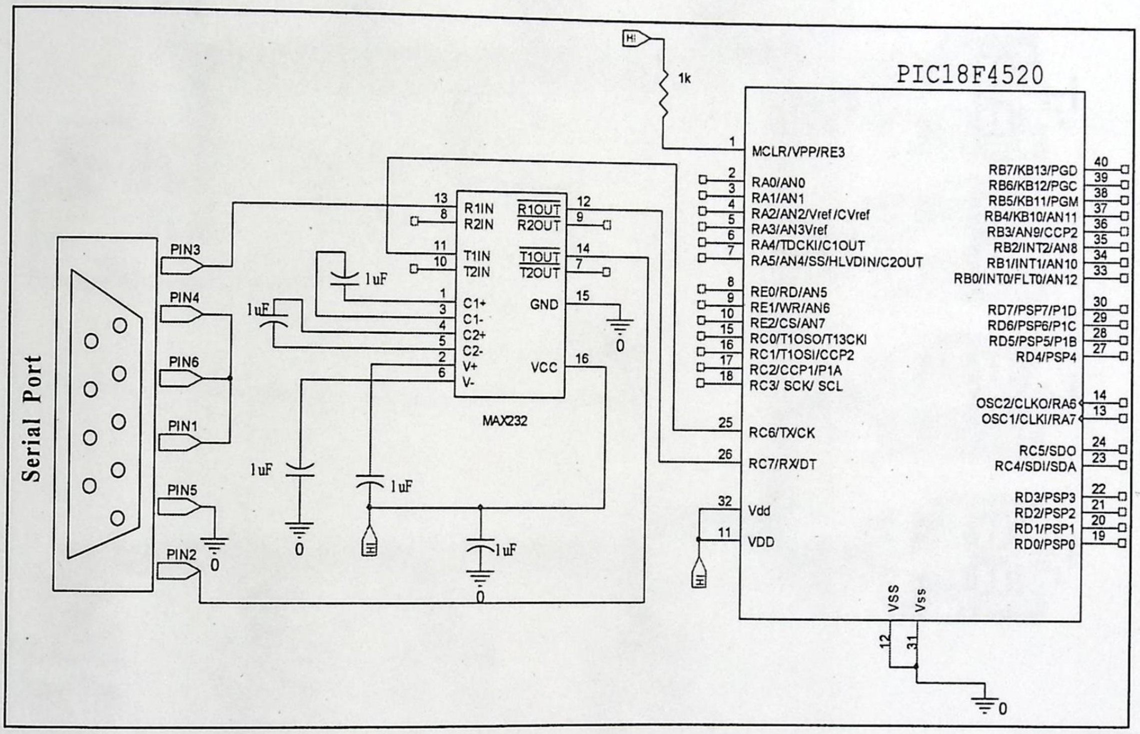


Figure 4-1: The Serial port to the PIC Interface Circuit

4.2.2 The 7-segment Display to the PIC Interface Circuit

7-segment display connects to the PIC through the 7447decoder. The decoder take four output line from one port in the PIC then send seven output lines (À, B̀, C̀, D̀, È, F̀, and G̀) to the 7-segment display, since the 7-segment displays two digits it needs two decoders with eight lines from the PIC output port. The following figure shows the interfacing circuit for the 7-segment display.

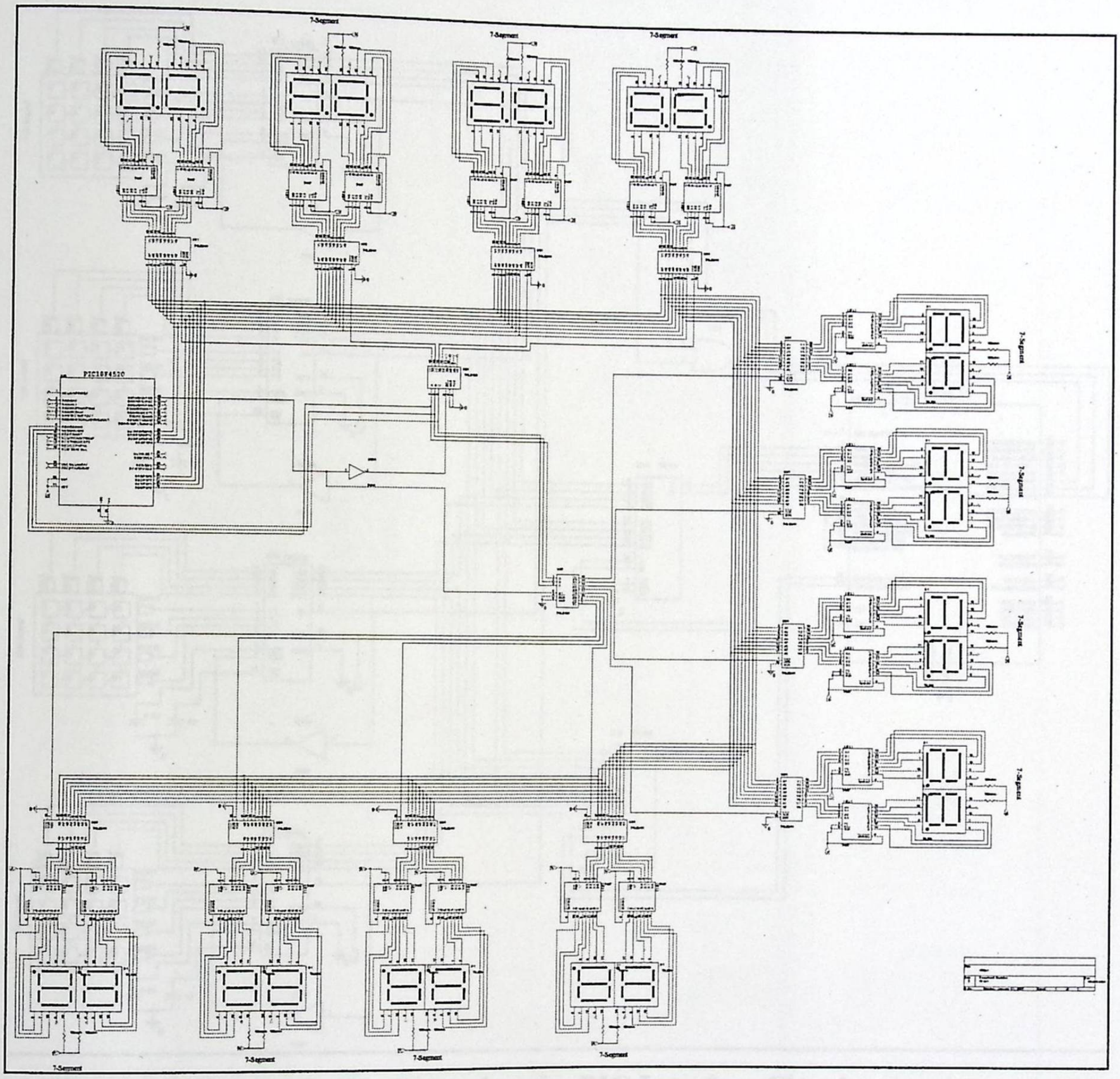


Figure 4-2: The 7-Segment Display to the PIC Interface Circuit

4.2.3 The Keypad to the PIC Interface Circuit

The system uses 4x4keypad which consists of four rows and four columns. The most suitable encoder been used to interface the keypad with the PIC was the MM74C922 which accepts eight inputs from the keypad and forwards four output to the PIC . The following figure shows the interfacing circuit for the keypad.

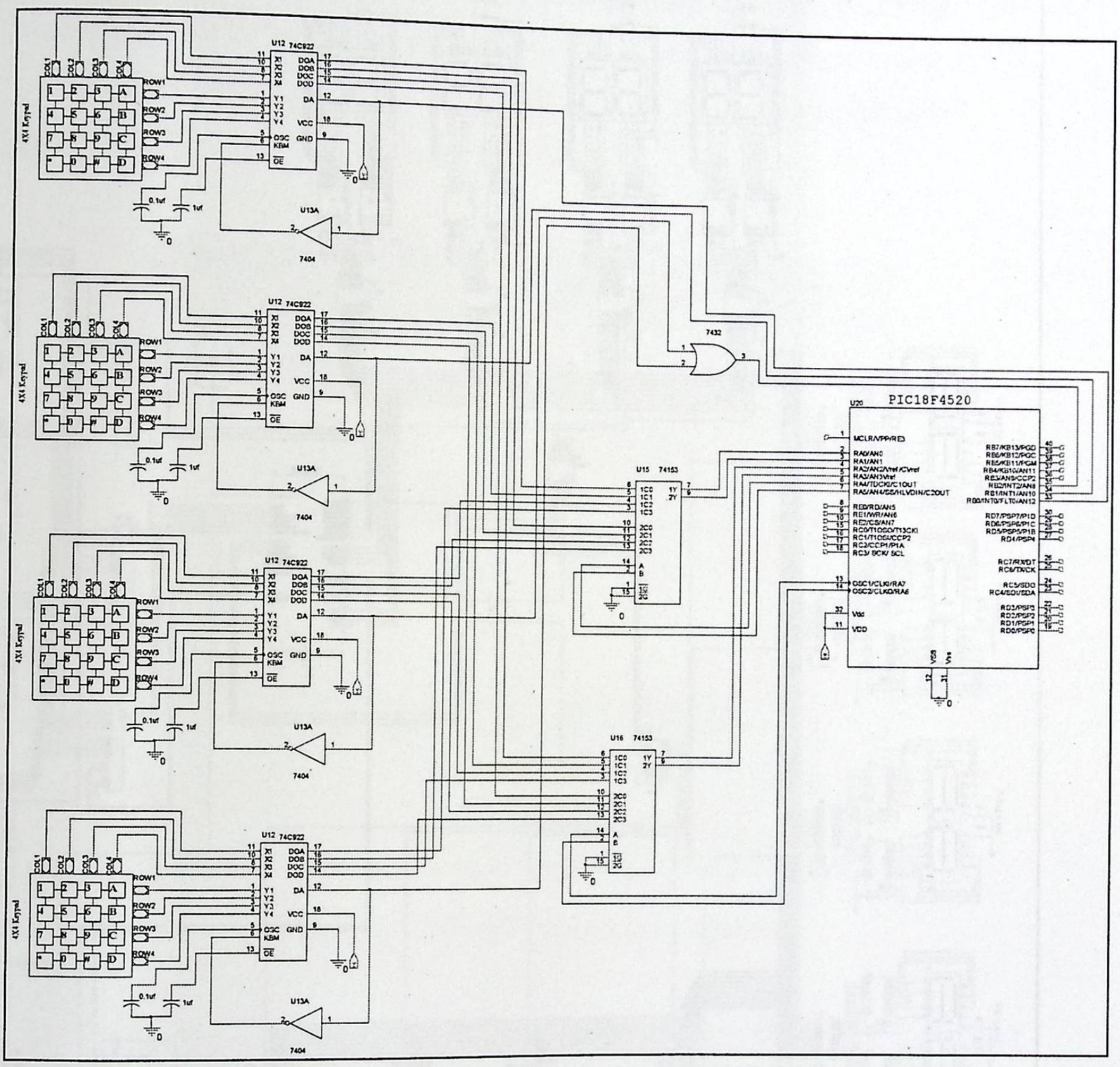


Figure4-3: The keypad to the PIC Interface Circuit

4.3 Over all System Design

The overall system design is described in the following schematic diagram.

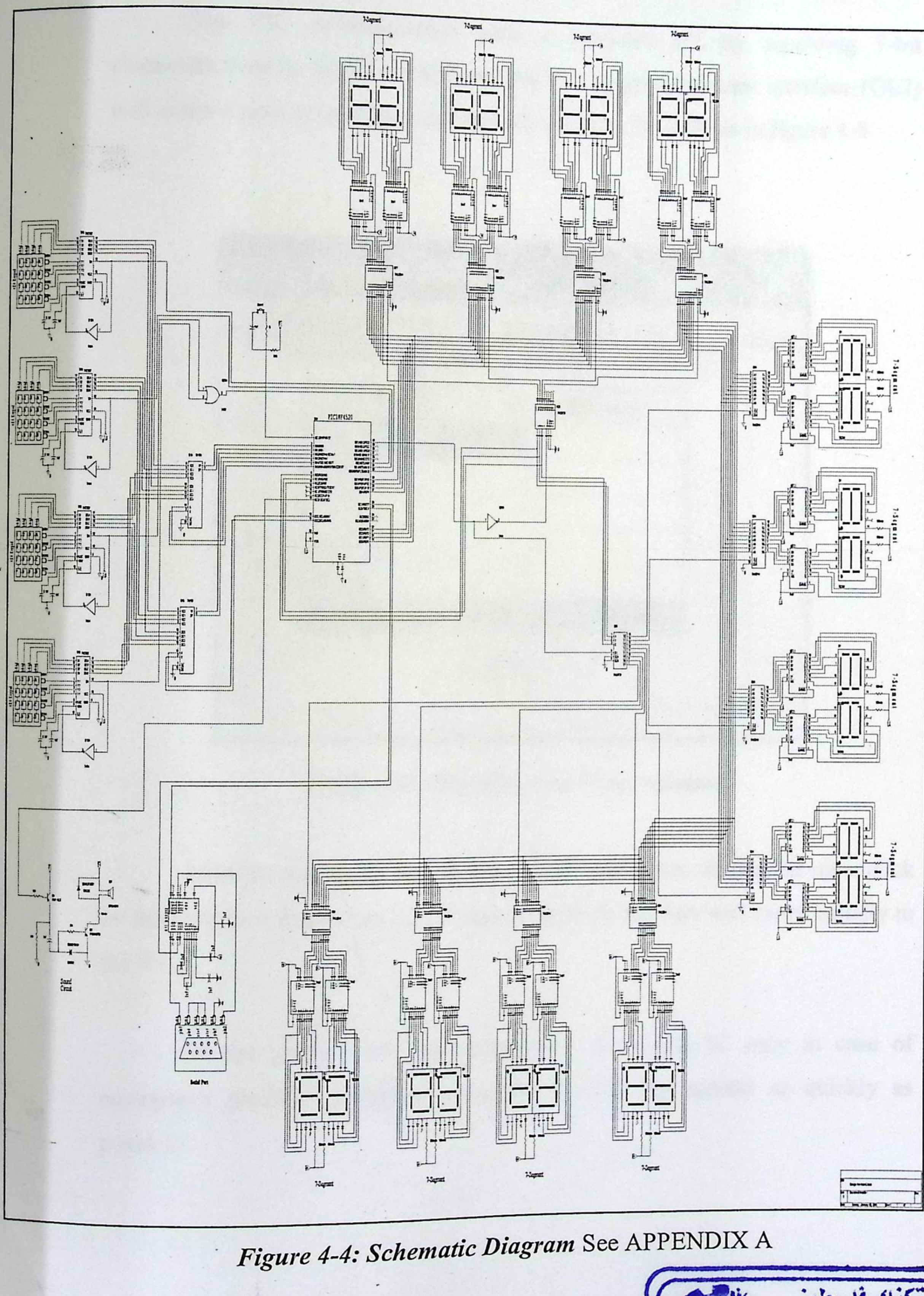


Figure 4-4: Schematic Diagram See APPENDIX A

4.4 User-System interface

The PIC communicates with a Network PC by receiving 8-bit characters over an RS-232 line From the PC. A graphical user interface (GUI) will allow a user to send the total number and the clinic ID as in figure 4-5

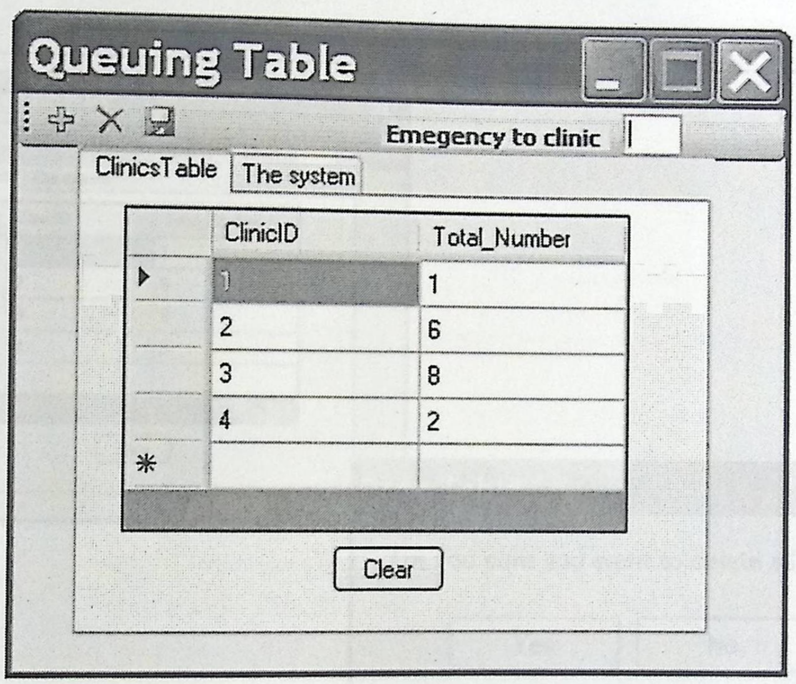


Figure 4-5: The Queuing Table window

After the user made any change on the total clinic number he must click on the save icon to confirm his change, as a result the data will move serially to the PIC.

On the textbox the user must enter the clinic IC only in case of emergency patients' situation, in order to serve the patient as quickly as possible.

When the user click the clear button which clear the whole database a warning massage box appears to worn him that this will clear the database and according to the user choice the database will be cleared or not.

The following figure shows the message box that appears and the window before and after clearing the data base.

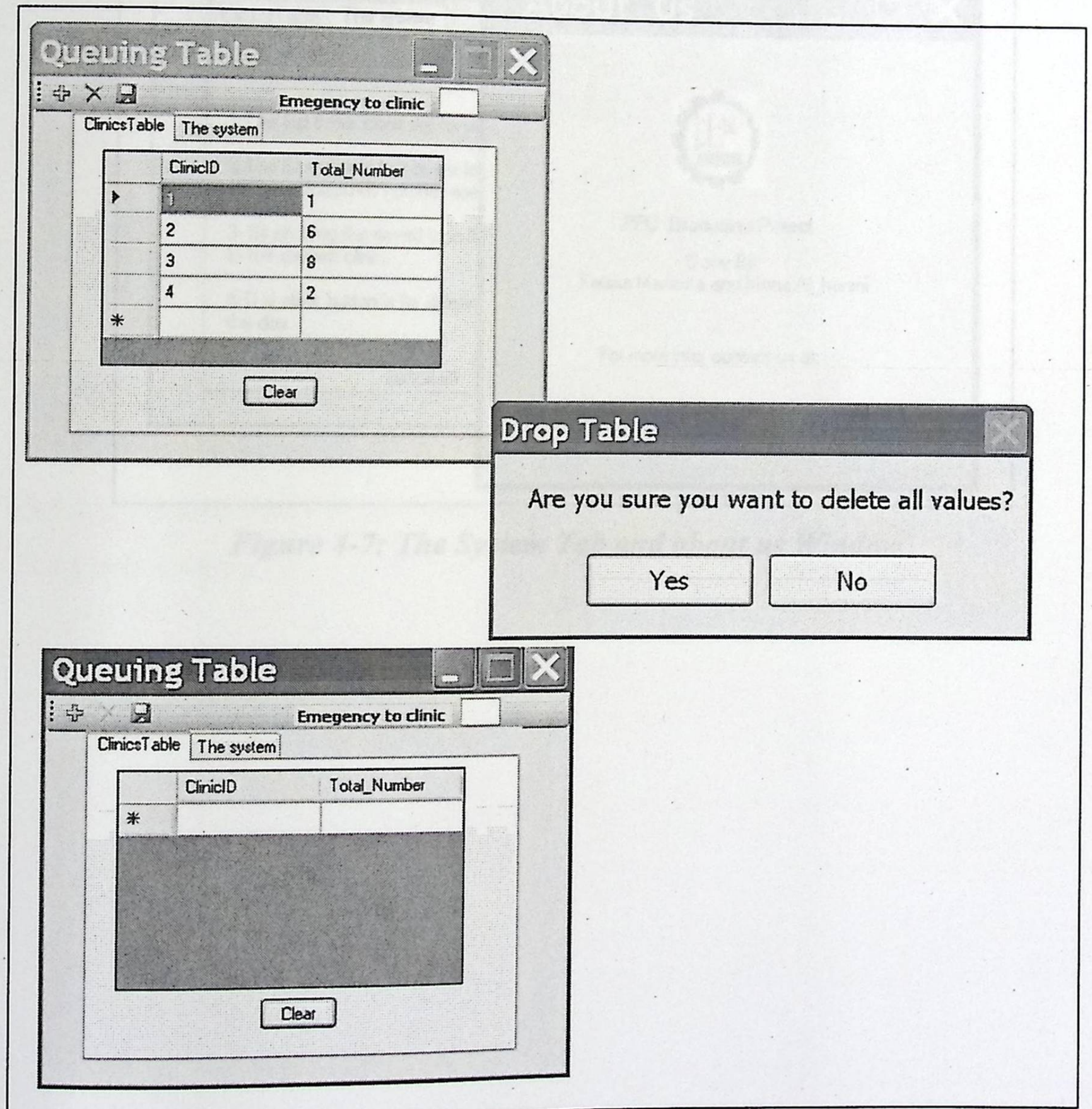


Figure 4-6: The Clearing Button Function

Other thing added to the GUI which is the system tab which has an information about using the system. Also has a link to other window which talks about the designers.

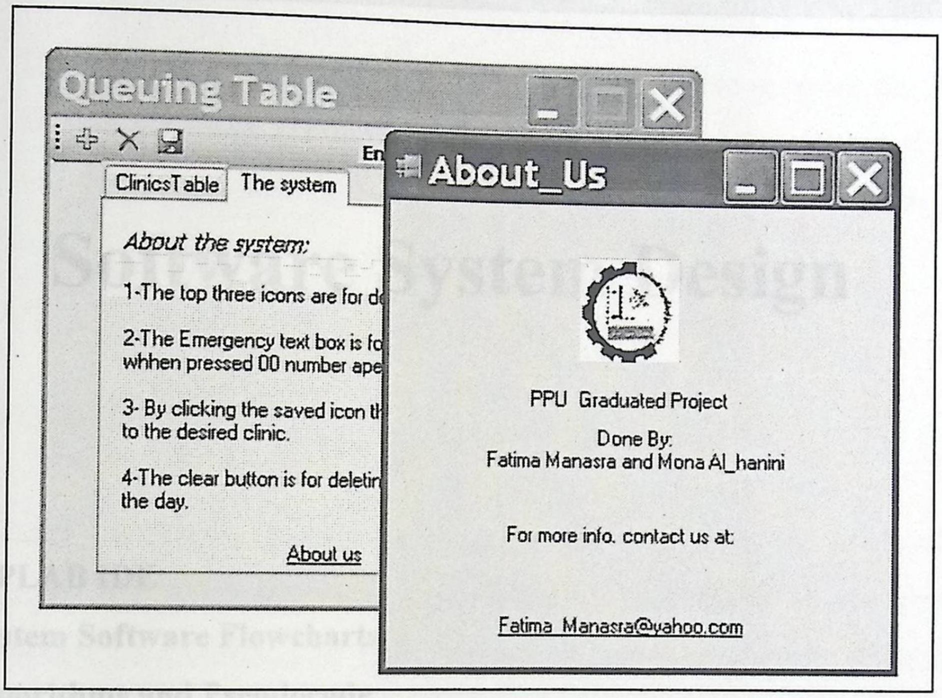


Figure 4-7: The System Tab and about us Window

5

Software System Design

Chapter Five
Software System Design

This chapter describes the basic program we used to program our PIC "MPLAB IDE". It also contains the general psudocode, flowcharts of the programs, algorithms used in the system and the general programming algorithms.

5.1 MPLAB IDE

MPLAB IDE is a software program that runs on a PC to develop applications for Microchip microcontrollers. It is called an Integrated Development Environment, or IDE, because it provides a single integrated "environment" to develop code for embedded microcontrollers.

5.1.1 MPLAB's Language Tools

Language tools are programs such as cross-assemblers and cross-compilers.

Most people are familiar with language tools that run on a PC such as Visual Basic or C compilers. When using language tools for embedded systems, a "cross-assembler" or "cross-compiler" is used. These tools differ from typical compilers in that they run on a PC but produce code to run on another microprocessor, hence they "cross-compile" code for a microcontroller that uses an entirely different set of instructions from the PC.

The language tools also produce a debug file that MPLAB IDE uses to correlate the machine instructions and memory locations with the source code. This bit of integration allows the MPLAB editor to set breakpoints, allows watch windows to view variable contents, and lets you single step through the source code, watching the application execute.

MPLAB IDE supports many language toolsuites. Integrated into MPLAB IDE is the Microchip MPASM Toolsuite, but many others can be used, including the Microchip C17, C18 and C30 Toolsuites, as well as language tools from HI-TECH, IAR, CCS, microEngineering Labs and Byte Craft. These are integrated into MPLAB IDE in two ways: using "plug ins" designed by the manufacturer, and by older style ".MTC" files that can be customized for any language toolsuite.

5.1.2 Application Debugging and Programming

There are two types of hardware that can be used with MPLAB IDE: programmers and hardware debuggers. A programmer simply transfers the machine code from the PC into the internal memory of the target microcontroller. The microcontroller can then be plugged into the application and, hopefully, it will run as designed.

5.2 System Software Flowcharts

These flowcharts shows the functions of the programs and algorithms written to make the system work properly, for each part of the system there is an algorithm written to show the function of its' parts and all these algorithms are joined to control the overall behavior of the system.

5.2.1 Main program flowcharts

These flowcharts describe how the basic program should work. Their is two main algorithm.

The first one for connecting the network PC with the PIC to display the total patients' number on the small 7-segments inside the clinic or to display "00" on the large 7-segments outside the clinic in case of emergency situation.

As seen from the flowchart bellow all the 7-segments are set to zero when the system start then as the number of the patients increase in each clinic the number on the 7-segment for the clinic increase.

When ever an emergency situation happened _ patient whose situation is in danger and can't wait_ the system will send "00" on the large 7-segments outside the indicated clinic. The system constantly checks if there is any changes on the total patients' number to inform the doctor about all the changes.

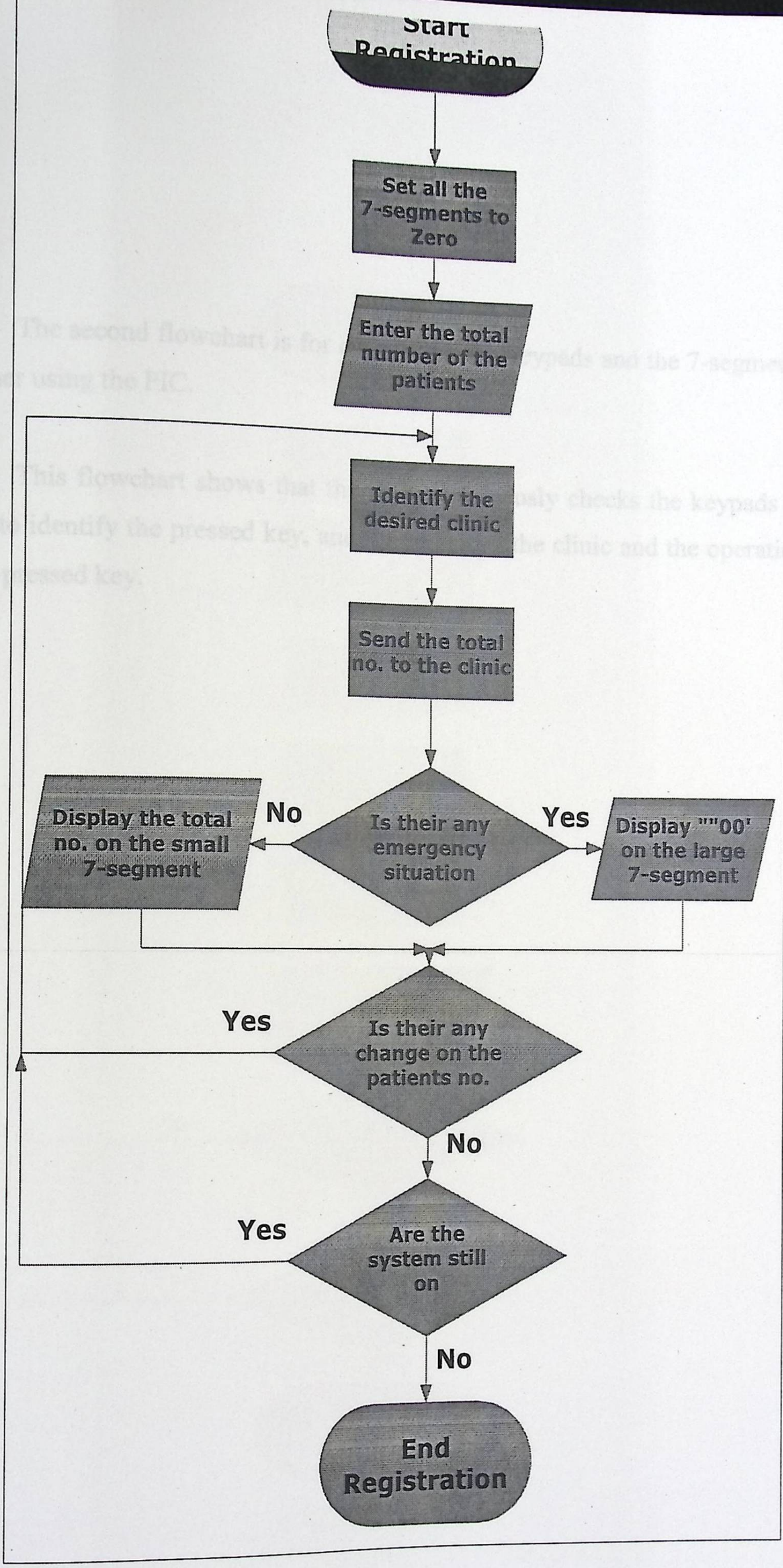


Figure 5-1: The First General Flowchart "Connecting the Network PC with the PIC"

The second flowchart is for connecting the keypads and the 7-segments together using the PIC.

This flowchart shows that the PIC continuously checks the keypads in order to identify the pressed key, and then identify the clinic and the operation of the pressed key.

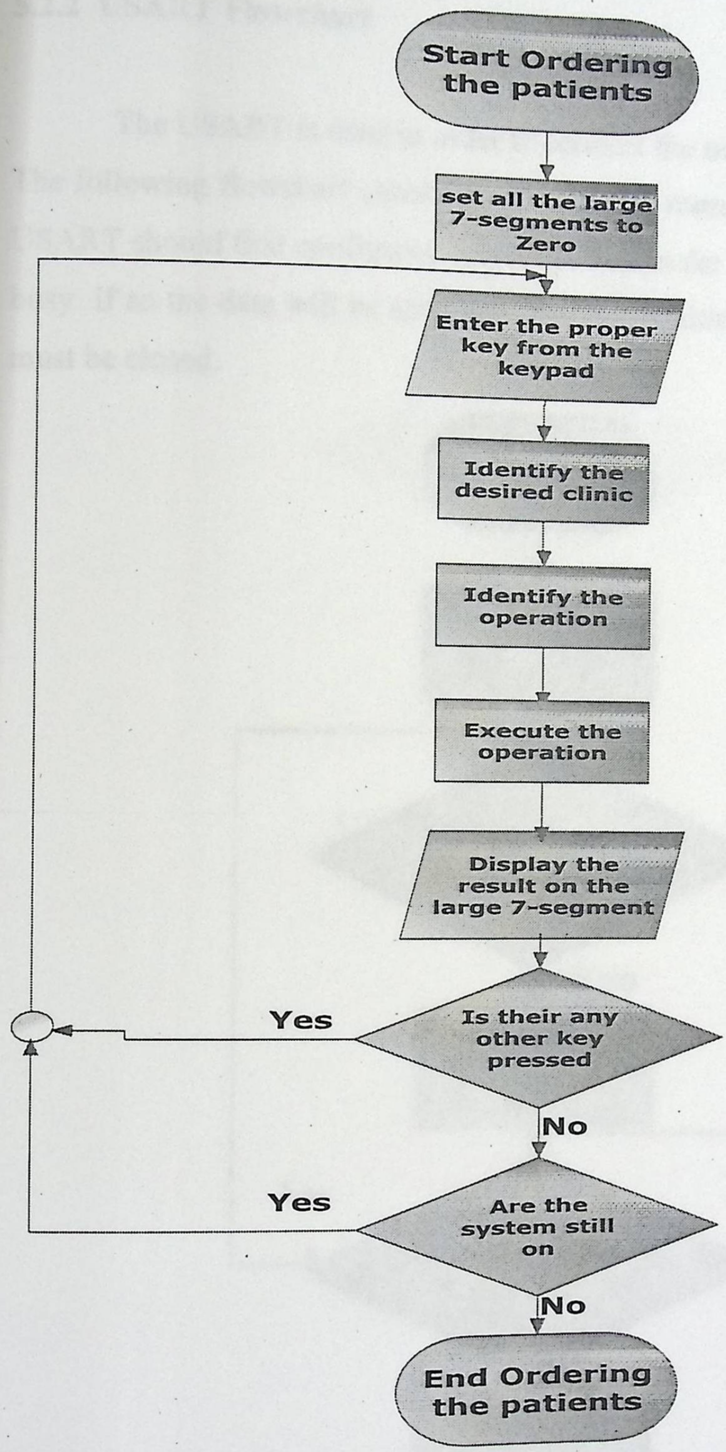


Figure 5-2: The Second General Flowchart "Connecting the 7-Segments with the Keypads"

5.2.2 USART Flowchart

The USART is used in order to connect the network PC with the PIC. The following flowchart shows that in order to transmit data from the PC ,the USART should first configured correctly, then make sure that the USART isn't busy if so the data will be send and after transmitting all the data the USART must be closed.

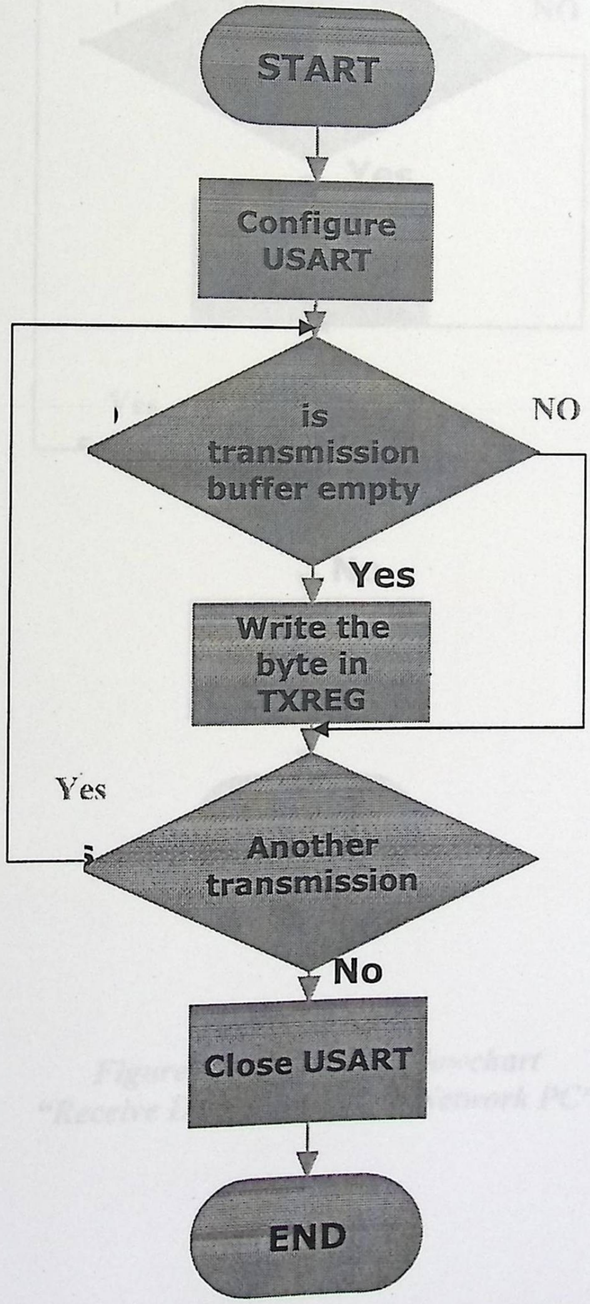


Figure 5.3: Transmission Flowchart
"Transmit Data From Network PC to PIC"

At the receiving side the same thing will happened reversely. First asking if the receiving buffer is full or not (is USART busy) then read the data one by one.

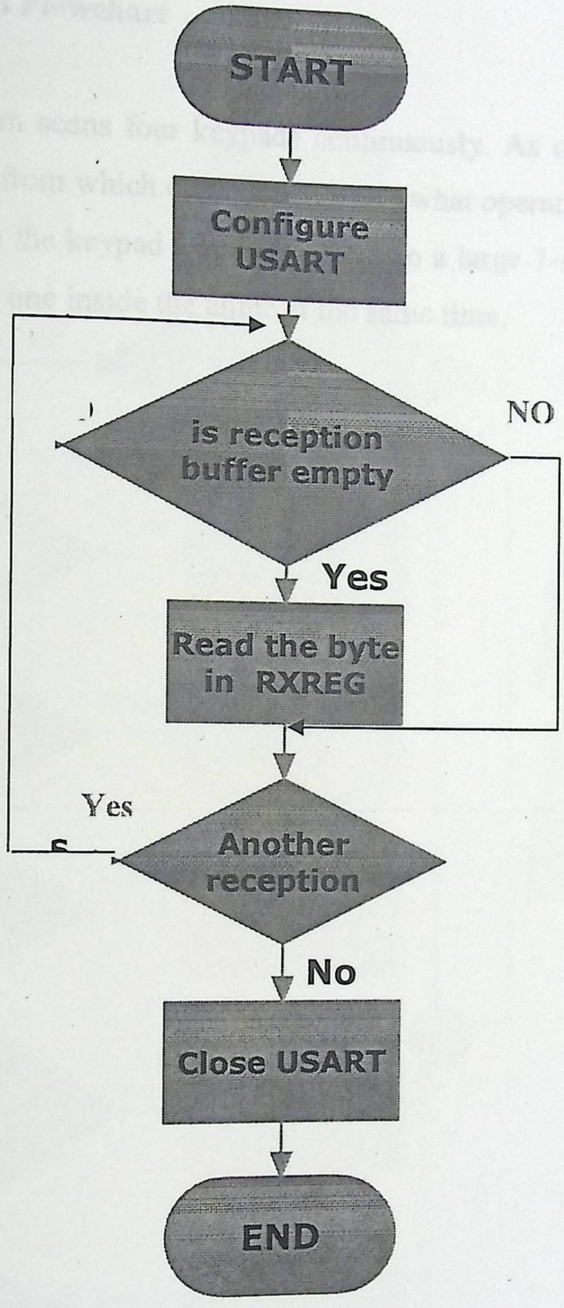


Figure 5.4: Receiving Flowchart
"Receive Data from PIC to Network PC"

5.2.3 The keypad Flowchart

The system scans four keypads continuously. As one key pressed the program identify from which clinic it come and what operation should perform. The number from the keypad will be shown into a large 7-segment outside the clinic and a small one inside the clinic at the same time.

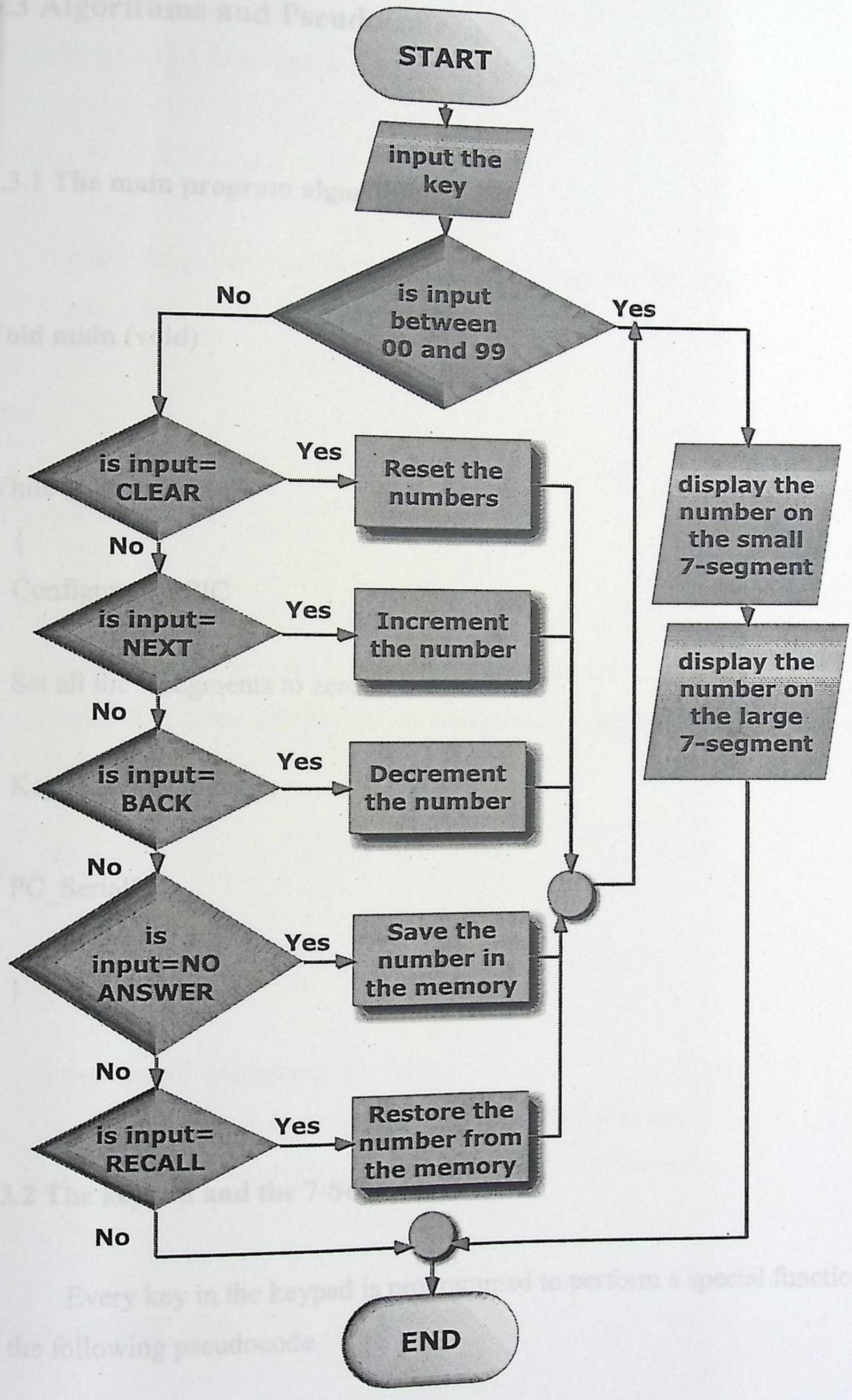


Figure 5.5: The Keypad Flowchart

5.3 Algorithms and Pseudocode

5.3.1 The main program algorithm:

Void main (void)

```
{
While (system is on)
  {
  Configure the PIC

  Set all the 7-segments to zero;

  Keypad_7-segment ();

  PC_Serial();

  }
}
```

5.3.2 The keypad and the 7-Segment ()

Every key in the keypad is programmed to perform a special function as in the following pseudocode.

Void keypad_7-segment ()

```
{
```

```
Check the input from the keypad;
Identify the clinic from which the key was pressed;
If input >=00 AND input <=99 then
   {
      Display the patient number on the inside 7-segment display;
      Display the patient number on the outside 7-segment display;
   }
Else

 If input=clear then
   {
      Reset the number;
      Display two zeros on the inside 7-segment display;
      Display two zeros on the outside 7-segment display;
      Clear the memory;
   }
 Else

 If input=NEXT then
   {
      Increment the patient number;
      Display patient number on the inside 7-segment display;
      Display patient number on the outside 7-segment display;
   }

If input=BACK then
   {
      Decrement the patient number;
      Display patient number on the inside 7-segment display;
      Display patient number on the outside 7-segment display;
```

}

If input=NOANSWER then

{

Save the patient number in the memory;

Increment the patient number;

Display patient number on the inside 7-segment display;

Display patient number on the outside 7-segment display;

}

If input=RECALL then

{

Restore the patient number from the memory;

Display patient number on the inside 7-segment display;

Display patient number on the outside 7-segment display;

Play sound; // Ding Dong sound

}

5.3.3 Network PC and the PIC Serially Connection

In order to create the connection between the network PC and the PIC the network PC must program to send data and the PIC to receive data. So there is two code side as following:

Void PC_Serial()

{

Void Sender_side()

{

```
Configure the USART
Open USART();
Busy USART(); // While busy do nothing
Send data();
}
```

Void Receiver_side()

```
{
 Configure the USART
 Open USART();
 Busy USART(); // While busy do nothing
 Receive data();
 If total_no= 0 then
        display_Emergency(clinicID,total);
 Else
        display_total(clinicID,total);
 }
}
```

5.3.4 The Display Functions

There is two display function, the first is to display the total number on the inside 7-segment only and the second is to display zero in case if emergency patient situation on both inside and outside 7-segment.

Void display_total(int clinicID,int total)

```
{
```

Identify the clinic;

Show the total number on the small 7-segment inside the clinic;

}

Void display_Emergency(int clinicID,int total)

{

Identify the clinic;

Save the previous number;

Show the zero on the small 7-segment inside the clinic and the large 7-segmentoutside the clinic;

}

5.4 Code Listing

In order to see the whole code refer to appendix B.

6

System Implementation and Testing

6.1 Implementation

6.2 Component testing

6.3 Subsystem Testing

6.4 System Software Testing

Chapter Six
System Implementation and Testing

Preface

To build a successful project, drawing the whole schematic of the project is needed, then begin with building and testing each circuit individually and finally connect these circuits together to perform the desired result of the project.

In the hardware work, testing is considered to be the most important phase and crucial step in implementing a system. Testing must be applied in away that makes it easy to perform, and can detect error directly. So after finishing the design of the system, and drawing the system schematic, the next step was to test it. At the beginning tested each chip individually, (as shown below).

This system has more than one issue to be tested. Some testing parts reflect a software, hardware .Also, testing procedures concentrate on a single device independent from the over whole system.

So, in this project first began to implement project circuits using breadboard and after testing each circuit and insure that the output as desired, then go to implemented it wire rapping connection.

6.1 Implementation

6.1.1 Building Clock and Reset Circuit

As an essential step is to build The Clock and Reset circuits for the PIC system, then connected the PIC to the other components.

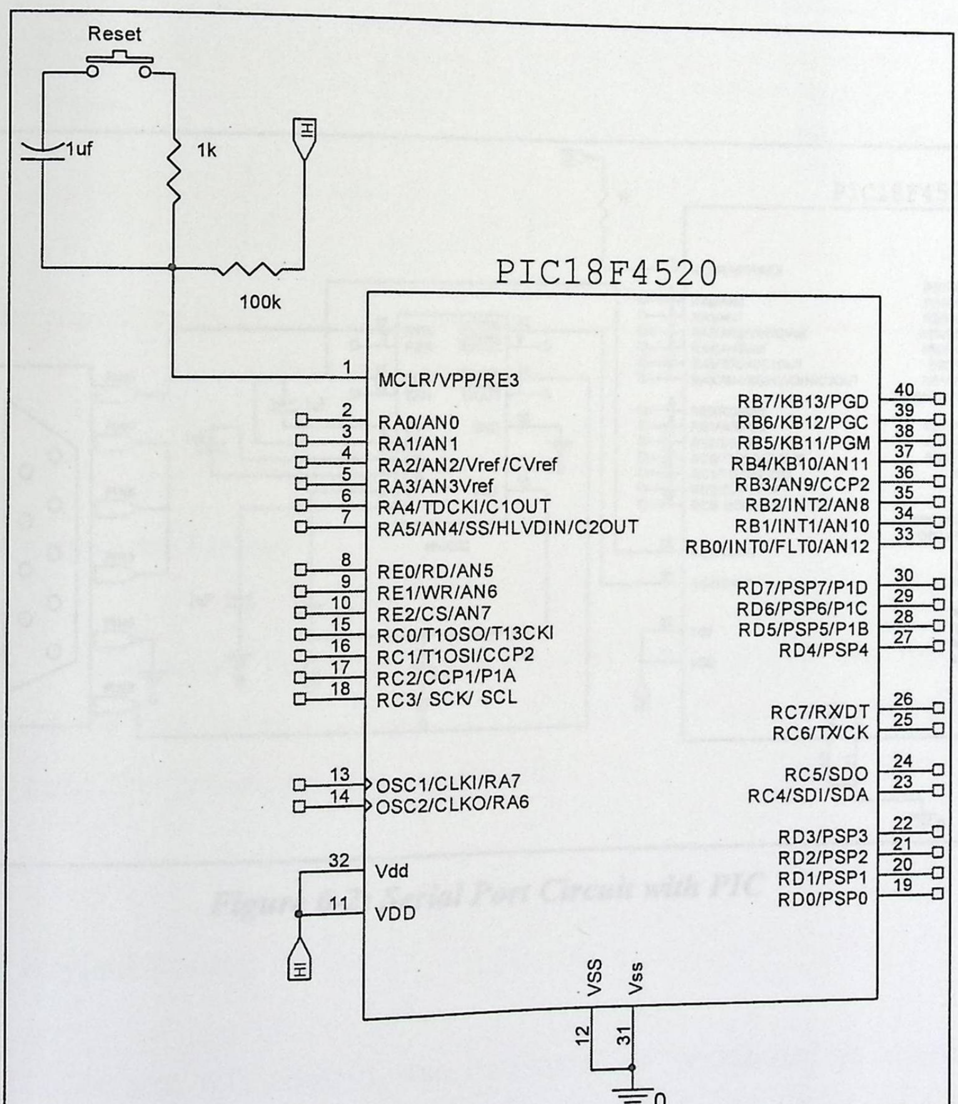


Figure 6-1: Clock and Reset Circuit

6.1 Implementation

6.1.1 Building Clock and Reset Circuit

As an essential step is to build The Clock and Reset circuits for the PIC system, then connected the PIC to the other components.

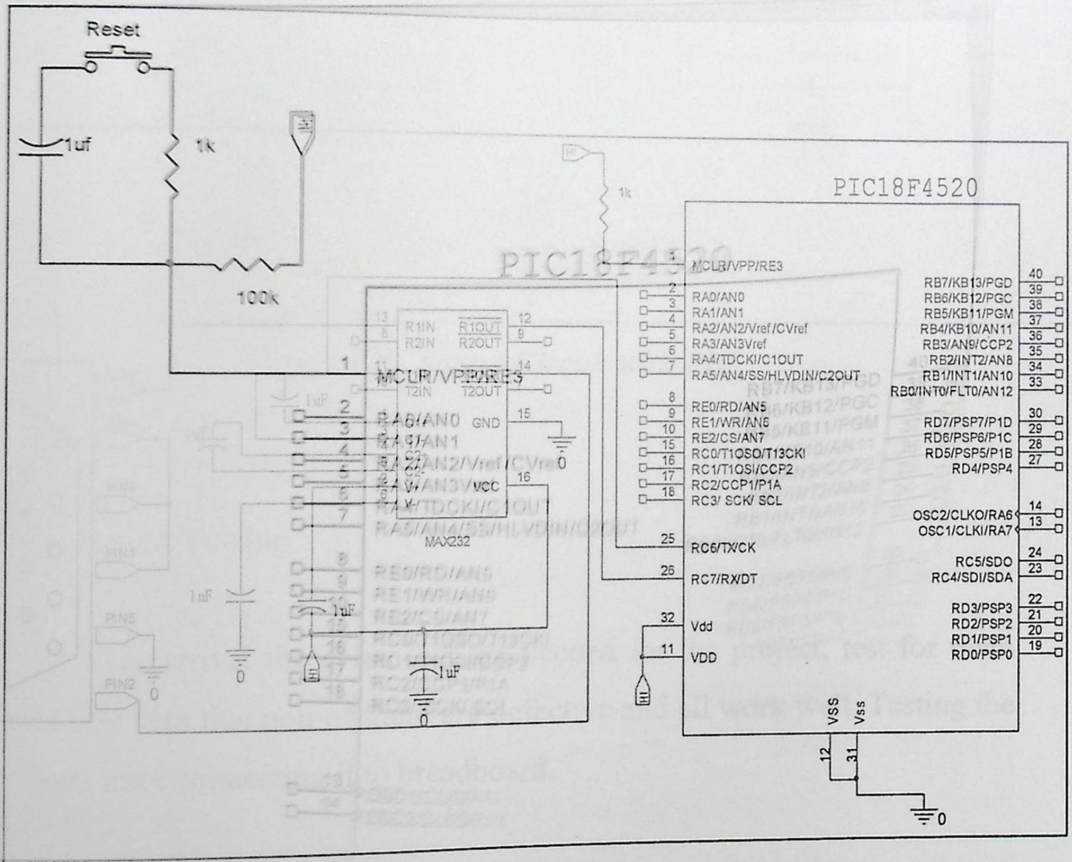


Figure 6-2: Serial Port Circuit with PIC

6.1.3 Building sound Circuit

This system includes sound circuit connected to the PIC with one pin.

6.1.2 Building the Serial Port (PC Interface) Circuit with PIC

Interfacing this system with the PC needed Serial port cable and MAX232.After that, Visual Basic.Net was chosen to make program for sending data from PC through cable to the PIC to output this data on 7-segments.

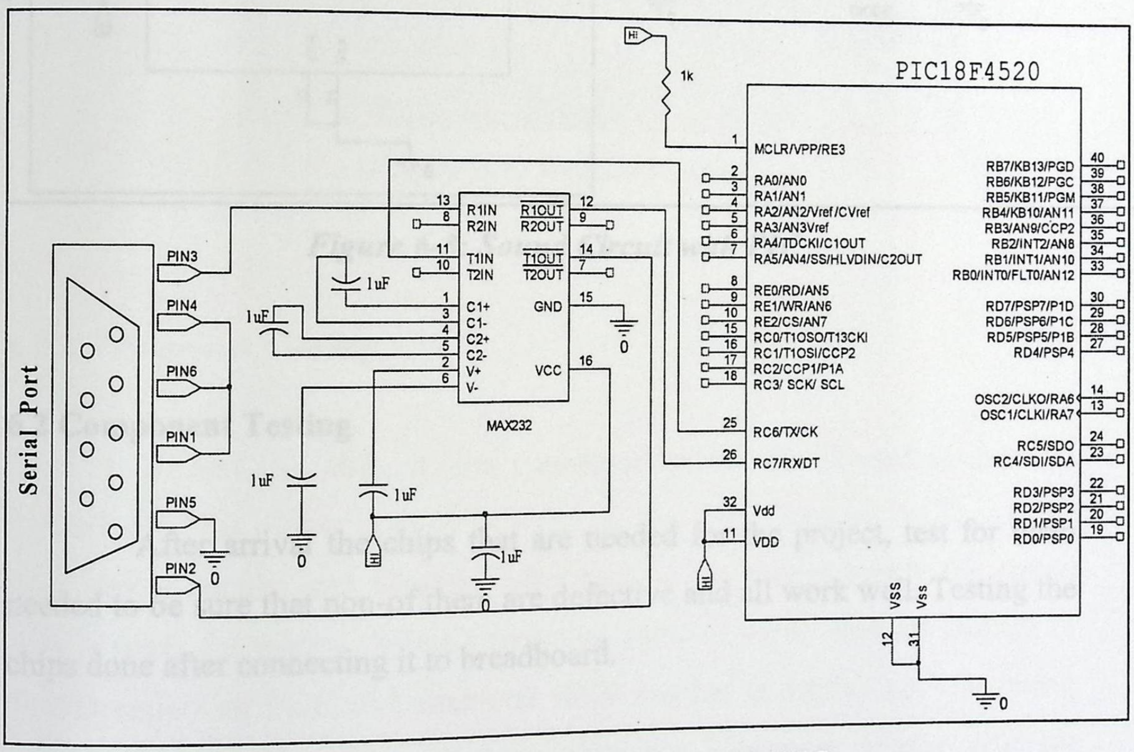


Figure 6-2: Serial Port Circuit with PIC

6.1.3 Building sound Circuit

This system includes sound circuit connected to the PIC with one pin.

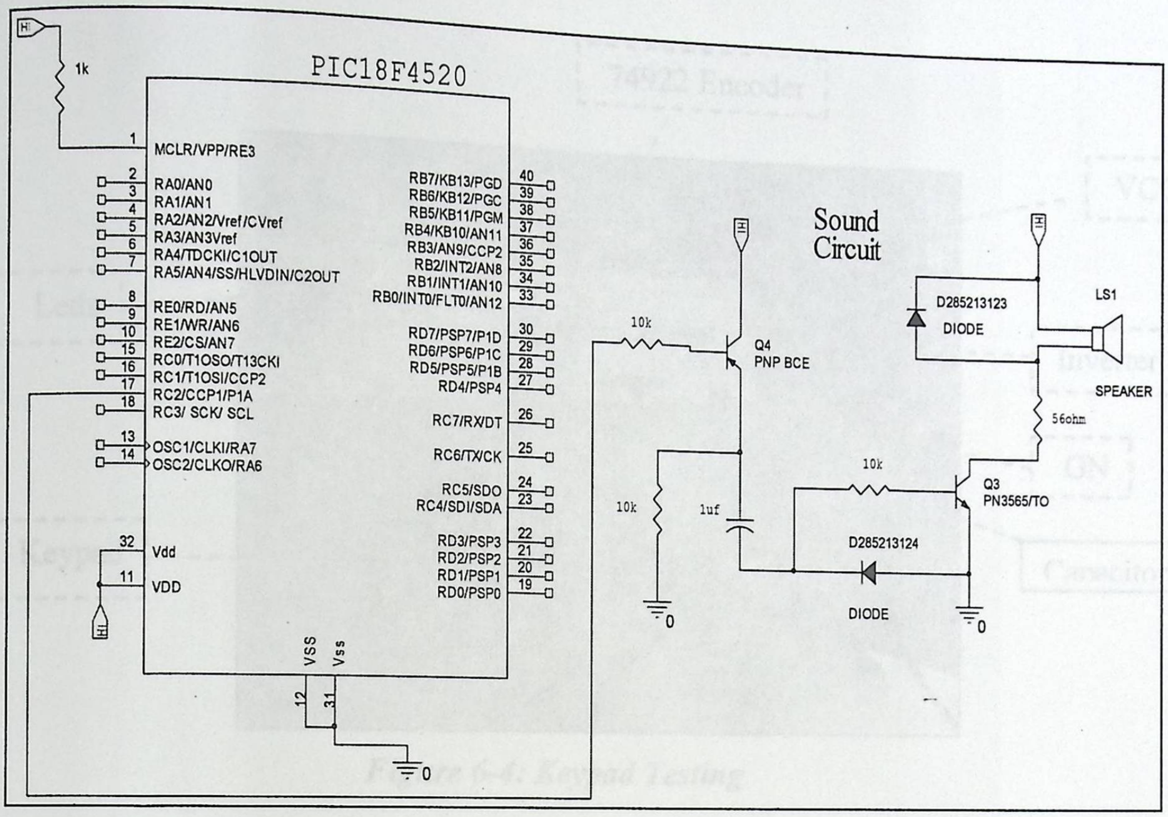


Figure 6-3: Sound Circuit with PIC

6.2 Component Testing

After arrival the chips that are needed for the project, test for them needed to be sure that non-of them are defective and all work well. Testing the chips done after connecting it to breadboard.

6.2.1 Keypad Testing

To test this chip, it was connected on breadboard as shown in figure(6-4). This circuit contain keypad, 74922Encoder, four Leds, and VCC and GND from power supply connected to the circuit. After turn on the power supply the circuit now is ready for testing. When any of the keys is pressed one Led or more are turns on.

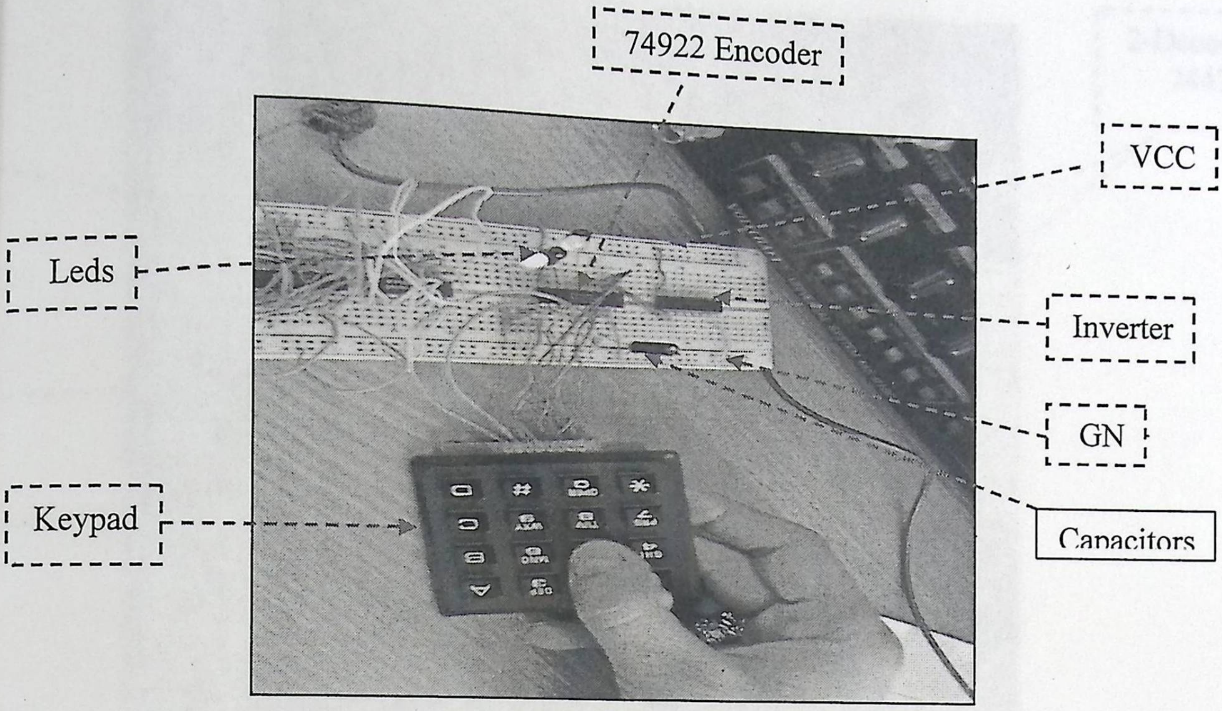


Figure 6-4: Keypad Testing

6.2.2 7-Segment Testing

To test this chip, it was connected on the breadboard as show in figure(6-5). This circuit contain 7-Segment, two 7447Decoder, VCC , GND from power supply, and 8-wires. After turn on the power supply the circuit now is ready for testing. When connecting some of wires -that are outputs of 7447Decoder- on VCC and others on GND number is display on 7-Segment, and by changing the places of these wires the 7-Segment display different numbers.

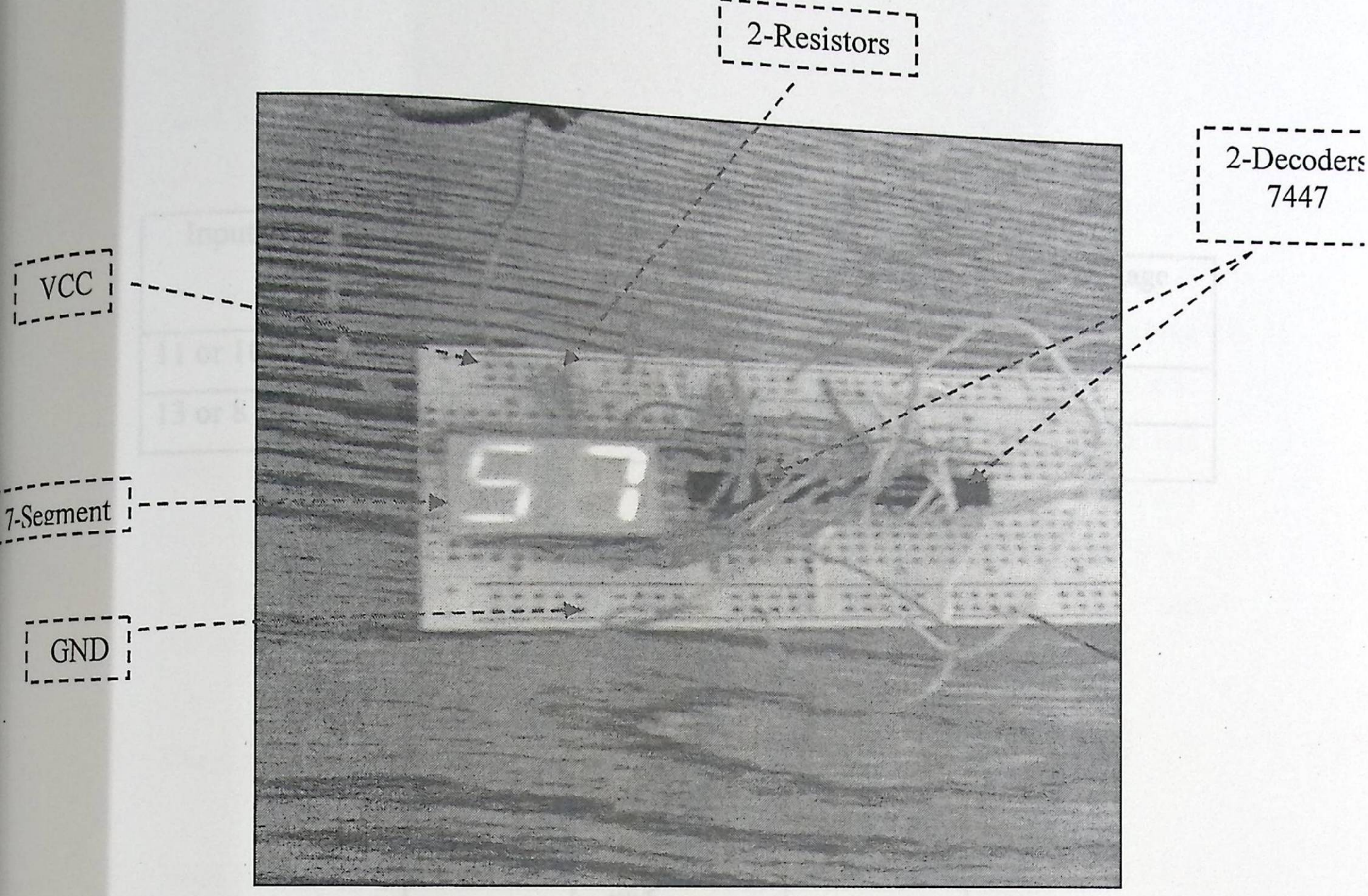


Figure 6-5: 7-Segment Testing

6.2.3 Sound Testing

To test this chip, it was connected on the breadboard as show in the circuit in figure (6-3). This circuit contains speaker, resistors, capacitor, and two diodes, two transistors VCC, and GND from power supply. After turn on the power supply the circuit now is ready for testing and the sound is audible from the speaker.

6.2.4 MAX232 Testing

To test this chip, it was connected with capacitors, VCC, and GND as shown in figure (6-6). Testing this chip done by entering a specific voltage level to one of the transceivers and then watching their results on digital multi-meter. The following table contains the testing results:

Table 6-1: Results of the MAX232 Circuit

Input Pin #	Output Pin #	Input Pin Voltage (V)	Out Pin Voltage (V)
11 or 10	14 or 7(respectively)	5	10
13 or 8	12 or 9(respectively)	10	5

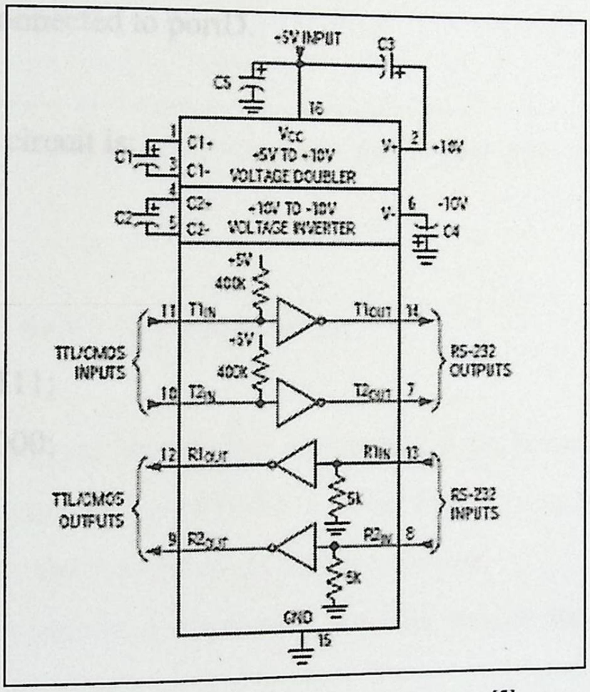


Figure 6-6: MAX232 Circuit [5]

6.3 Subsystem Testing

In this phase each individual circuit that performs a special function was tested and the result as following:

6.3.1 Testing PIC and Keypad Circuit

This circuit was tested after connecting it on breadboard as shown in the schematic diagram in figure (6-7). A code was written by C Programming Language to enable the keypad to be an output unit that enables the keys to output data. The program worked correctly and the PIC programmed with this code, and by using digital multi-meter the results watched -after pressing any key- on the output pins of the 74922Encoder and these results are either 0volt or 5volt according to which key is pressed. The result of pressing keys are shown on 8 lids connected to portD.

The Code for this circuit is:

```
Void main (void)
{
TRISA=0b11111111;
TRISD=0b00000000;
ADCON1=15;
PORTD=PORTA;
}
```

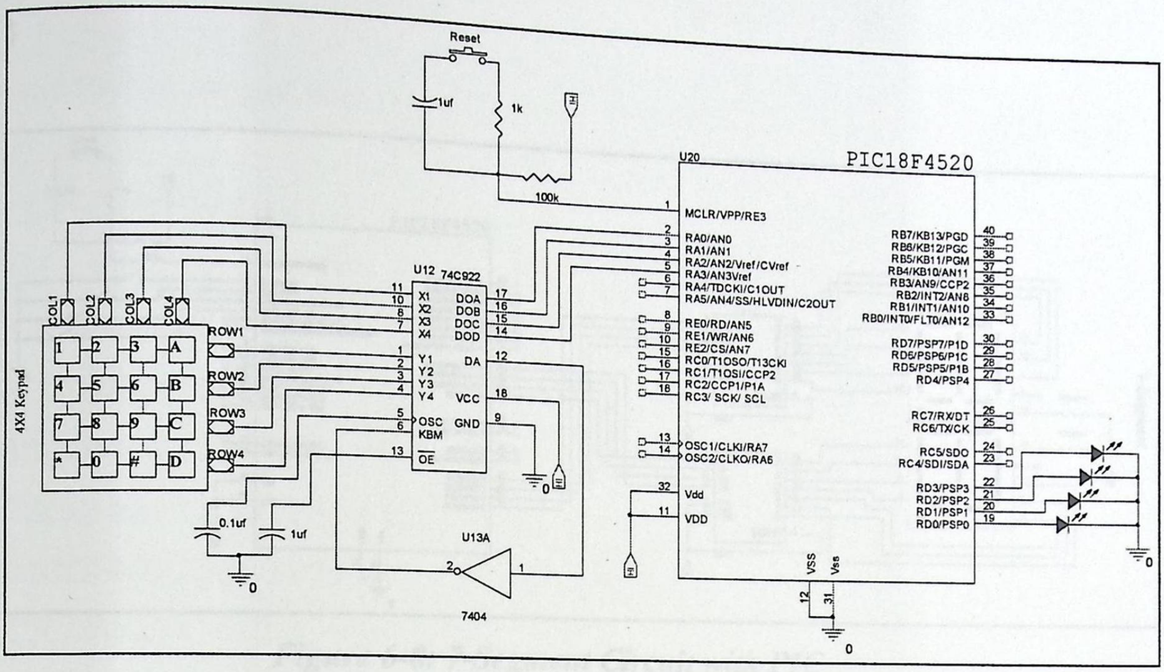


Figure 6-7: Keypad Circuit with PIC

6.3.2 Testing PIC and 7-Segment Circuit

This circuit was tested after connecting it on breadboard as shown in the schematic diagram in figure (6-8). A code was written by C Programming Language to enable the 7-Segment to display number.

Figure(6.) show breadboard connection for this circuit and the output on 7-Segment.

The code for this circuit is:

```
Void main (void)
{
TRISA=0b11111111;
TRISD=0b00000000;
ADCON1=15;
PORTD=PORTA;
```

}

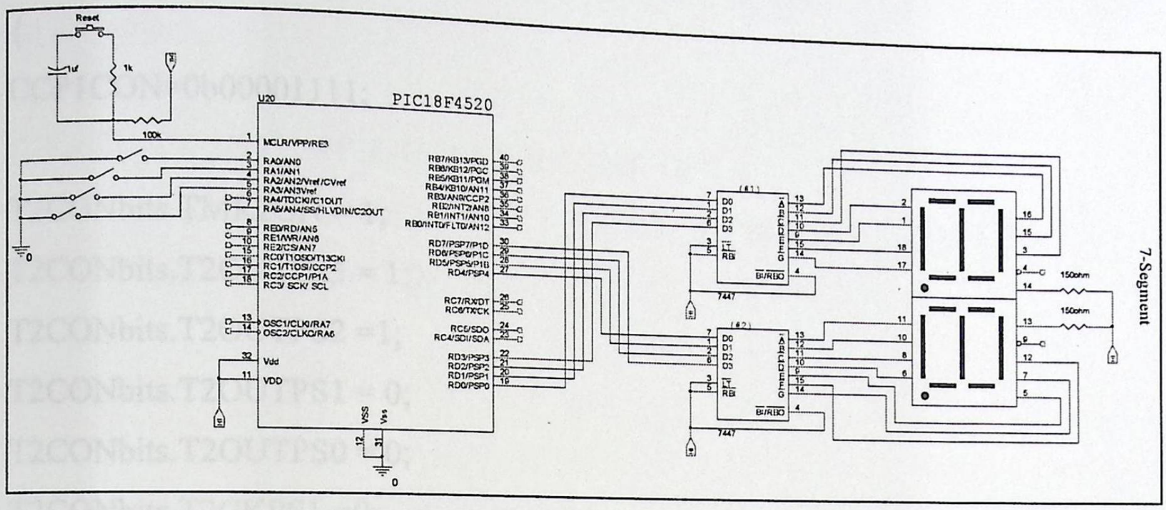


Figure 6-8: 7-Segment Circuit with PIC

6.3.3 Testing PIC and Sound Circuit

This circuit was tested after connecting it on breadboard as shown in the schematic diagram in figure(6-3). A code was written by C Programming Language to enable the PIC to output sound on the speaker that is audible to human.

The code for this circuit is:

```
void sound(void)
{
PWM(1);
Delay1KTCYx(100);
PWM(5);
Delay1KTCYx(100);
CCP1CON=0b00000000;
```

```
}
void PWM(unsigned char i)
{
CCP1CON=0b00001111;

T2CONbits.TMR2ON = 1;
T2CONbits.T2OUTPS3 = 1;
T2CONbits.T2OUTPS2 =1;
T2CONbits.T2OUTPS1 = 0;
T2CONbits.T2OUTPS0 = 0;
T2CONbits.T2CKPS1 =0;
T2CONbits.T2CKPS1 =0;

PR2 = 255;
TRISCbits.TRISC2=0;
T2CONbits.TMR2ON = 1;
CCPR1L = 5*i;
}
```

6.3.4 Testing PIC with Serial Port (PC Interface) Circuit

This circuit was tested after connecting it on breadboard as shown in the schematic diagram in figure (6-2). A code was written by C Programming Language to enable the PIC to receive data from serial port and output this data

The function for the PIC is:

```
void USARTResieving ()
{
```

```
int counter=0;
int TotalPatientNo;
int clinicID;
OpenUSART
(USART_TX_INT_OFF & USART_RX_INT_OFF &
USART_ASYNCH_MODE & USART_EIGHT_BIT & USART_CONT_RX
& USART_BRGH_LOW,77);

while(!counter)
    {
        while (BusyUSART()); //do nothing
        clinicID=ReadUSART();
        PORTE=clinicID;
        !counter;
        }//while

while (BusyUSART()); //do nothing
TotalPatientNo=ReadUSART();
if (TotalPatientNo==0)
{
PORTBbits.RB7=1; //enable the small 7-segments

switch(clinicID)
        {
case( 0x00): //Enable the 1st 7-segment
                    PORTEbits.RE0=0;
                    PORTEbits.RE1=0;
                    PORTEbits.RE2=0;

case( 0x01): //Choosing the second clinic
```

```
                    //Enable the 2nd 7-segment
                    PORTEbits.RE0=1;
                    PORTEbits.RE1=1;
                    PORTEbits.RE2=0;

case(0x02): //Choosing the third clinic

                    PORTEbits.RE0=1;
                    PORTEbits.RE1=0;
                    PORTEbits.RE2=1;

case(0x03)://Choosing the fourth clinic
                    PORTEbits.RE0=1;
                    PORTEbits.RE1=1;
                    PORTEbits.RE2=1;

if (TotalPatientNo==0)DisplayEmergency();
 DisplayTotal(TotalPatientNo);
 }//switch
CloseUSART();
}
}
```

6.3.5 Testing One Clinic Design

The system also tested on one clinic design. Figure 6-9 shows the schematic diagram and the following one shows the implementation on the breadboard.

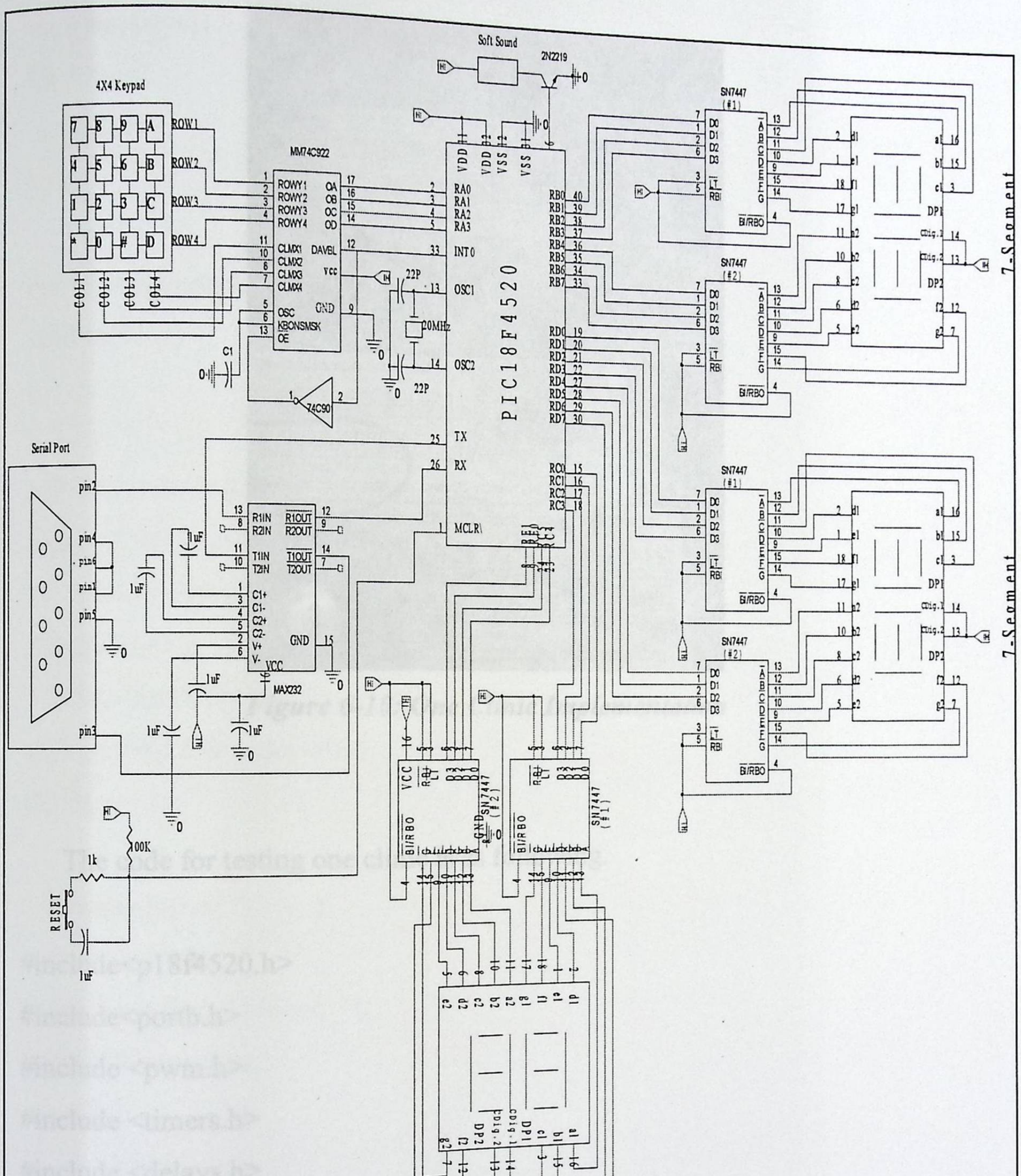


Figure6-9: One Clinic Design

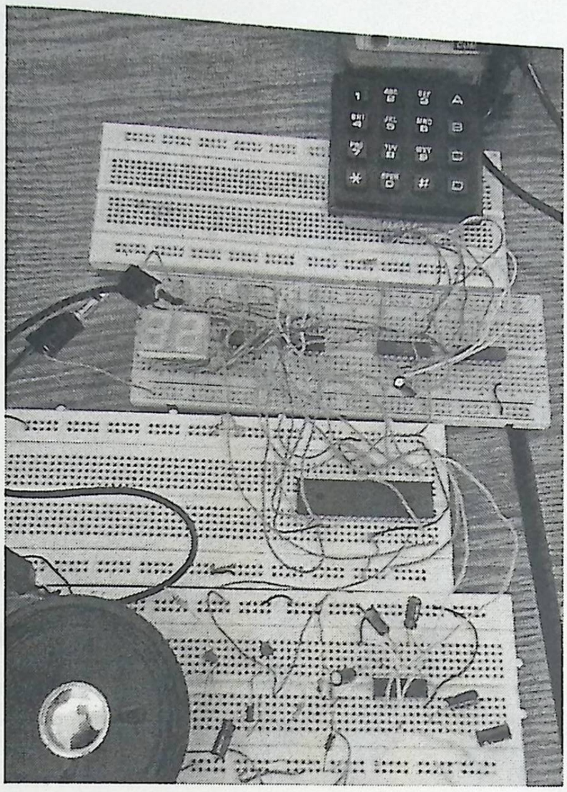


Figure 6-10: One Clinic Implementation

The code for testing one clinic is as following.

```
#include<p18f4520.h>
#include<portb.h>
#include <pwm.h>
#include <timers.h>
#include <delays.h>

void DiplayKeyPressed(int);
void arrayputNoAnswer(int);
void arraygetRecall(void);
int Input,first,second,number;
int Input;
int flag=0;
```

```
int Arraykeys1[20];
int Arin=0;
int Arout=0;
int i;
#pragma interrupt aa
void aa(void)
{
if(PORTBbits.RB0==1){
Input=PORTA & 0B00001111;
TRISD=00;
ADCON1=15;
TRISA=0b00001111;
if (Input>=0 && Input<=2){
if(flag == 1)
 {
 second=Input;
 first=first<<4;
 number= first + second +1;
 PORTD=number;
 ++flag;
 }

if(flag == 0)
 {
 first=Input;
 ++flag;
 Delay1KTCYx(1);
 number= first +1;
 PORTD=number;
```

```
 }
}//if (Input>=0 && Input<=2)

if (Input>=4 && Input<=6){
if(flag == 1)
 {
 second=Input;
 first=first<<4;
 number= first + second;
 PORTD=number;
 ++flag;
 }
if(flag == 0)
 {
 first=Input;
 ++flag;
 Delay1KTCYx(1);
 number= first;
 PORTD=number;
 }
}//if (Input>=4 && Input<=6)

if (Input>=8 && Input<=10){
 if(flag == 1)
 {
 second=Input;
 first=first<<4;
 number= first + second -1;
 PORTD=number;
```

```
  ++flag;
  }

if(flag == 0)
  {
  first=Input;
  ++flag;
  Delay1KTCYx(1);
  number= first -1;
  PORTD=number;
  }
}//if (Input>=8 && Input<=10)
if (flag==2)flag=0;
if(Input==15){++number;PORTD=number;}//NEXT
if(Input==14){--number;PORTD=number;}//BACK
if(Input==11)arrayputNoAnswer(number);//NO ANSWER
if(Input==7)arraygetRecall();   //RECALL
if(Input==3){PORTD=0b00000000;for(i=0;i<20;++i)Arraykeys1[i]=0;}//CLE
AR

if(Input==12){if (flag==0)PORTD=first;PORTD=number;} //ENTER

}//if(PORTBbits.RB0==1)

}//Interrupt

#pragma code high_vector=0x08
void high_vector (void)
{ _asm goto aa _endasm }
```

```
#pragma code

void main(void)
{
INTCON = 0b10010000;
INTCON3=0B0011100;
ADCON1=15;
}

void arrayputNoAnswer(int KeyArray)
        {
//Saving in The first clinic Array

        if(Arin<18)
        {
        Arraykeys1[Arin]=KeyArray;

        ++Arin;
        }
        }


void arraygetRecall()
        {
        //Recalling The first clinic Array data
        int KeyArray;

        if(Arout<18)
        {
        KeyArray=Arraykeys1[Arout];
```

```
PORTD=KeyArray;
//Delay1KTCYx(1);
++Arout;
}
}
```

6.4 System Software Testing

The software that controls the system was tested alone without hardware by using PIC18 Simulator IDE. This process was done to be sure that the problems generated are only software problems.

To see the system software refers to appendix B.

7

Conclusions and Future Work

7.3 Future Works
7.2 Problems
7.1 Conclusions

Chapter Seven
Conclusions and Future Work

This chapter introduces some significant points about the way of continuing do more and more in the field of the system concepts or tools. Also, it represents the conclusions extracted during designing and implementing it. The chapter illustrates the system implementation achievements and output.

7.1 Conclusions

Many conclusions can be stated here, but only significant and important ones are described here:

**This project challenged us as engineers and it was very demanding as the team spent 30-40 hours a week in the university lab working on this project in the last month. We learned a lot and used everything we had learned in our classes to solve the problems and come up with solutions to make this system work.

**In This project, we've navigated through many experiences that we've never gone through before. We've learned different approaches and experiences, especially the way of thinking and how to develop an approach to solve problems.

**There were different problems that we've faced and solved in the implementation phase in which we learned how to trace the different signals step by step, chip by chip, and module by module, and how to use different tools and utilities.

**For programming thePIC18f4520 microcontroller, we used MPLAB IDE program with MPLAB ICD2 debugging and programming device.

**The microcontroller can be programmed in different languages using MPLAB IDE. The language we used is C so all programs are written in C.

**Each device was tested individually in its own circuit to study its behavior and make sure it works properly and can do its expected job.

**The subsystems we defined were implemented each in its own circuit and tested by means of Hardware and Software.

**In the next testing stage two or more subsystems were combined together to check the influence of their outputs on each other.

**Then the whole system will be upgraded to check its work and test the complete system program on it.

7.2 Problems

As we go on the project a new problems appears and new way of thinking comes to solve any obstacle try to stop the program.

7.2.1 Hardware Problems

The size of the project was a problem itself. It led to the need of many ports from the PIC so we use port expansion techniques by using the decoders and the muxes. It also leads to high cost and less availability.

7.2.2 Software problems:

**The software was much difficult than we thought. We face many hard issues. Those enforce us to change a written program many times. The first code we wrote was about 1024 line for the keypads and the 7-segments only, now we come up with 318 line that perform the same function more efficiently.

**Dealing with portB interrupt was not that easy. We are forced to learn every thing about the interrupt registers and how to configure it.

**The serial programming also faced many problems that comes to an end.

7.3 Future Works

**This project will be complete to conclude all the clinic rooms in Al_Ahli hospital.

** Companies, banks and institutions may be using this system.

References

Book references

GAONKAR Microprocesser Archeticture, Programming and Applications with 8085/8080. *S.Gaonkar*. 1984

The 80386DX Microprocessor Hardware Software and Interfacing. *A.Triebel*.1992

Internet References

http://en.wikipedia.org

www.alldatasheet.com

www.microchip.com

PIC datasheet

[1]http://ww1.microchip.com/downloads/en/DeviceDoc/39631a.pdf

[2]http://users.tpg.com.au/gramo/Site/proton_keypad1.htm

[3]http://www.electronics-project-design.com/electronics-design-contest.html

[4] http://www.datasheetarchive.com/preview/2190358.html

[5] http://www.datasheetarchive.com/preview/456208.html

[6]http://www.ee.pucrs.br/~lep/ftp/inicio/datasheets/circ_int/ttl/TEXAS/74922.

[7]http://www.ee.pucrs.br/~lep/ftp/inicio/datasheets/circ_int/ttl/TEXAS/74153.pdf

[8] http://www.datasheetarchive.com/preview/455149.html

[9]http://www.ee.pucrs.br/~lep/ftp/inicio/datasheets/circ_int/ttl/motorola/74244.pdf

APPENDIX A: Schematics

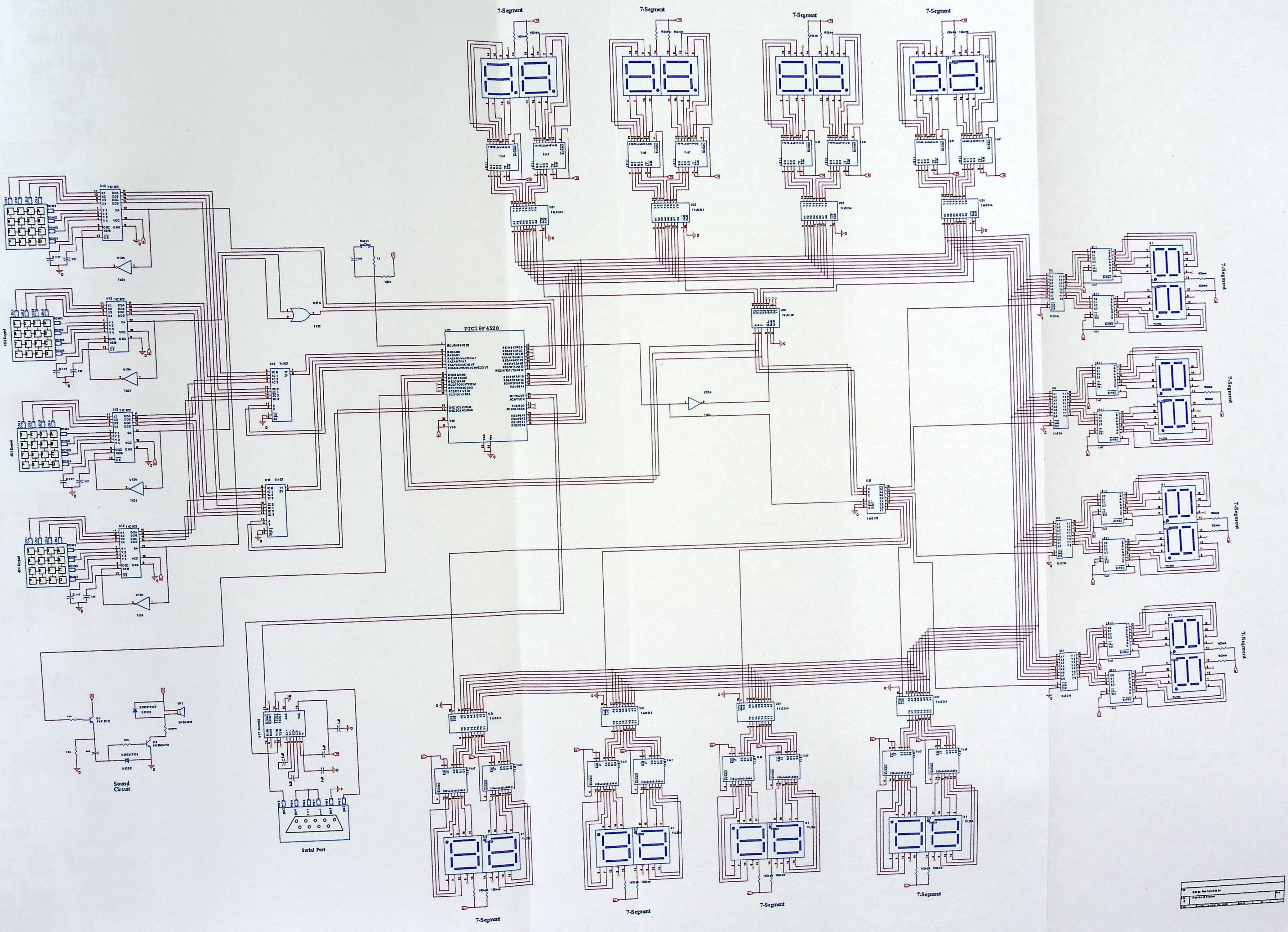
APPENDIX B: Code

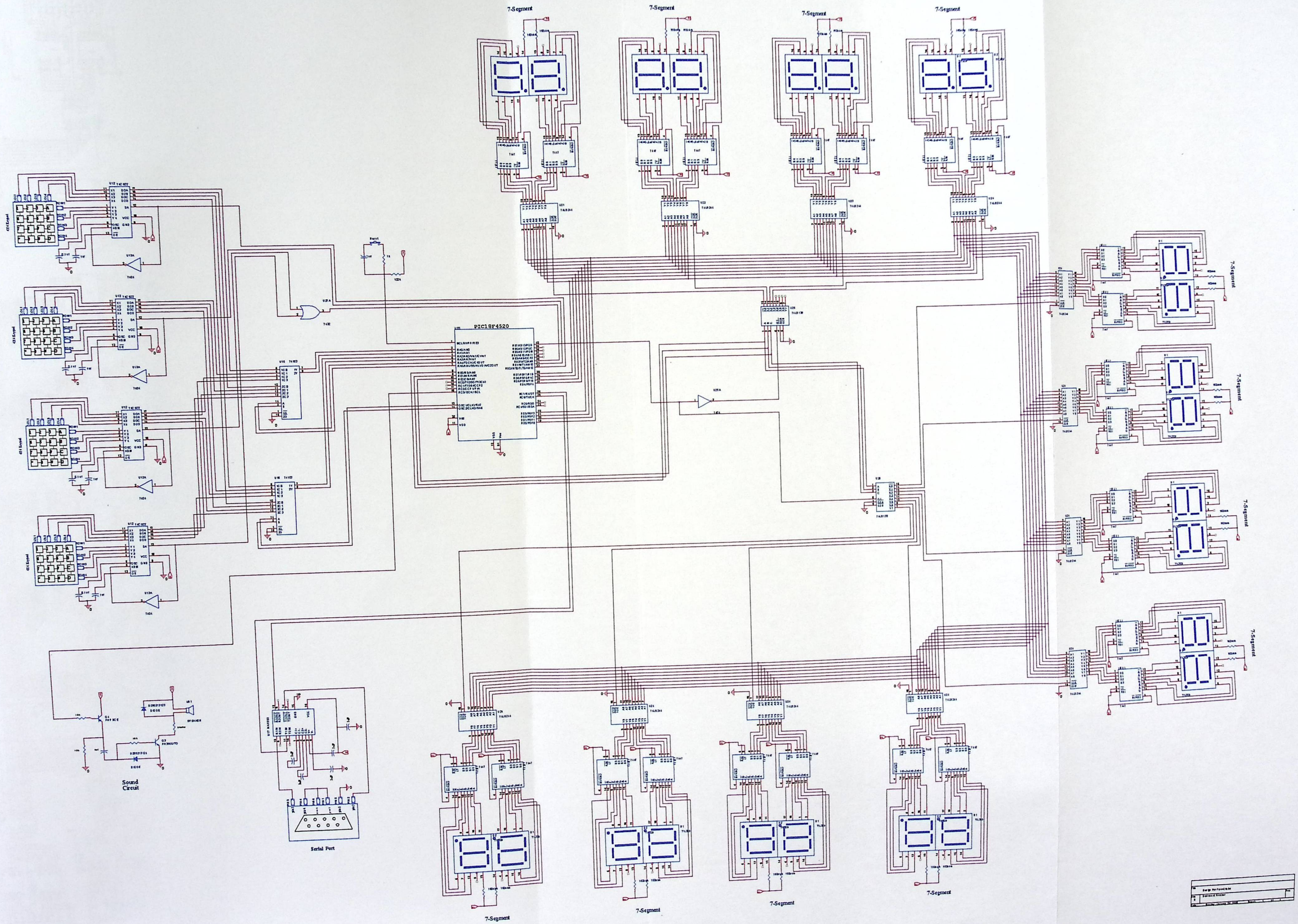
APPENDIX C: PIC18f4520 Datasheets

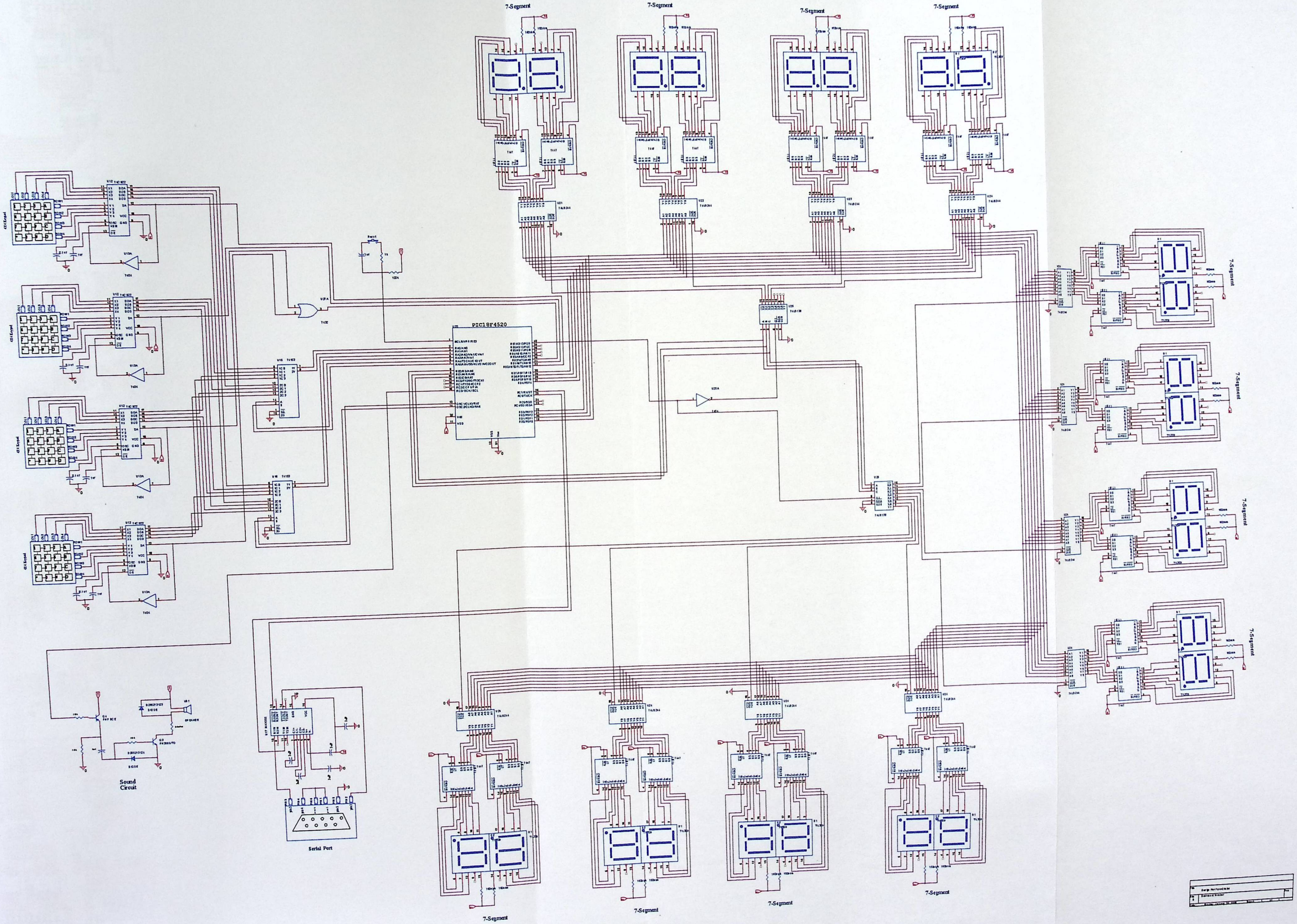
APPENDIX D: Components Datasheets

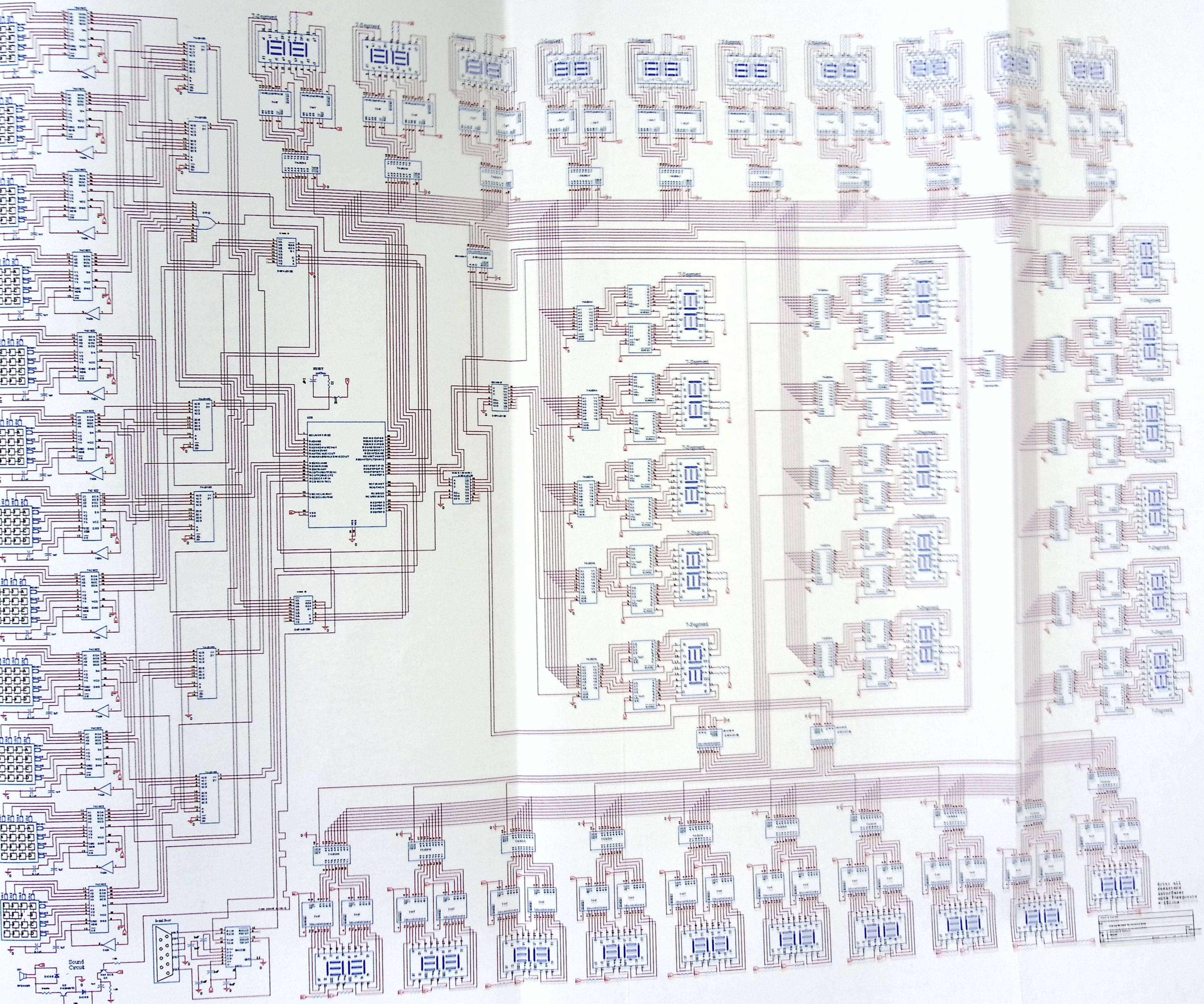
APPENDIX A

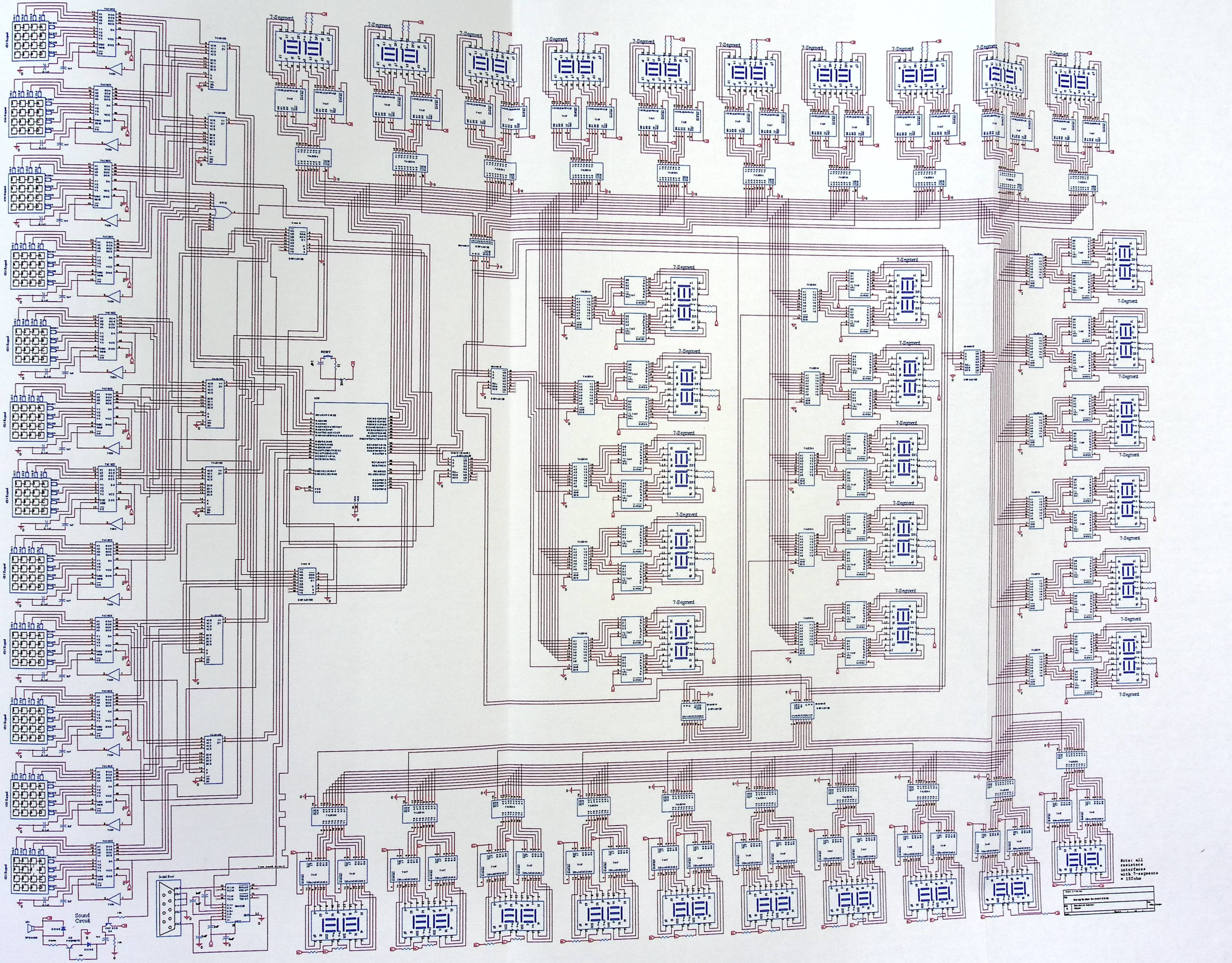
Schematics


7-Segment
7-Segment
7-Segment
7-Segment
PIC18F4520
7-Segment
7-Segment
7-Segment
7-Segment
Sound Circuit
Serial Port
7-Segment
7-Segment
7-Segment
7-Segment


7-Segment
PIC18F4520
Sound Circuit
Serial Port


7-Segment
7-Segment
7-Segment
7-Segment
PIC18F4520
Sound Circuit
Serial Port
7-Segment
7-Segment
7-Segment
7-Segment
7-Segment
7-Segment
7-Segment
7-Segment


7-Segment
Serial Port
Sound
Circuit


7-Segment
RESET
Serial Port
Sound
Circuit
Note: all
resisters
interfaces
with 7-segments
= 150ohm

APPENDIX B

Code

Apendix B

The keypads and the 7-segments code

```
#include<p18f4520.h>
#include<portb.h>

int Input,first,second,number;
int Input;
int flag=0;

void Display (void);
void NoAnswer(int);
void Recall(void);

int Arraykeys1[20];
int Arraykeys2[20];
int Arraykeys3[20];
int Arraykeys4[20];
int ClinicID;
int Arin=0;
int Arout=0;
int i;

#pragma interrupt aa
void aa(void)
{

if(PORTBbits.RB0==1){      //Choosing the FIRST clinic
                      ClinicID=1;
                       PORTAbits.RA4=0;
                      PORTAbits.RA5=0;
                      PORTAbits.RA6=0;
                      PORTAbits.RA7=0;
                       Input=PORTA & 0B00001111;

                      Display();
                    }//if(PORTBbits.RB0==1)

if(PORTBbits.RB1==1){ //Choosing the SECOND clinic
                      ClinicID=2;
                        PORTAbits.RA4=1;
                      PORTAbits.RA5=0;
```

```
                    PORTAbits.RA6=1;
                    PORTAbits.RA7=0;
                    Input=PORTA & 0B00001111;
                    Display();
                }//if(PORTBbits.RB1==1


if(PORTBbits.RB2==1){ //Choosing the THIRD clinic
                      PORTAbits.RA4=0;
                    PORTAbits.RA5=1;
                    PORTAbits.RA6=0;
                    PORTAbits.RA7=1;

                    Input=PORTA & 0B00001111;

                if (Input==0)
                        { //Choosing the FOURTH clinic
                        ClinicID=4;
                          PORTAbits.RA4=1;
                        PORTAbits.RA5=1;
                        PORTAbits.RA6=1;
                        PORTAbits.RA7=1;
                        Input=PORTA & 0B00001111;
                        Display();
                            }

                    ClinicID=3;
                    Display();

}//if(PORTBbits.RB3==1)

}//Interrupt


#pragma code high_vector=0x08
void high_vector (void)
{ _asm goto aa _endasm }
#pragma code


void main(void)
{

INTCON = 0b10010000;
INTCON3=0B0011100;
ADCON1=15;
TRISD=00;
```

```
TRISA=0b00001111;

}

/******DISPLAY FUNCTION*******/

void Display (void)
{

if (Input==13)PORTD=0b00000000;
if (Input>=0 && Input<=2){
                    if(flag == 1)
                    {
                    second=Input;
                    first=first<<4;
                    number= first + second +1;
                    PORTD=number;
                    sound();
                    ++flag;
                    }
                     if(flag == 0)
                    {
                    first=Input;
                    ++flag;
                    Delay1KTCYx(1);
                    number= first +1;
                    PORTD=number;
                    sound();
                    }
                }//if (Input>=0 && Input<=2)


 if (Input>=4 && Input<=6){
                    if(flag == 1)
                    {
                    second=Input;
                    first=first<<4;
                    number= first + second;
                    PORTD=number;
                    sound();
                    ++flag;
                    }
                     if(flag == 0)
                    {
```

```
            first=Input;
            ++flag;
            Delay1KTCYx(1);
            number= first;
            PORTD=number;
            sound();
            }
        }//if (Input>=4 && Input<=6)


if (Input>=8 && Input<=10){
            if(flag == 1)
            {
            second=Input;
            first=first<<4;
            number= first + second -1;
            PORTD=number;
            sound();
            ++flag;
            }

             if(flag == 0)
            {
            first=Input;
            ++flag;
            Delay1KTCYx(1);
            number= first -1;
            PORTD=number;
            sound();
            }
        }//if (Input>=8 && Input<=10)

        if (flag==2)flag=0;

if(Input==15){++number;PORTD=number;}//NEXT

if(Input==14){--number;PORTD=number;}//BACK

if(Input==11)NoAnswer(number);//NO ANSWER

if(Input==7)Recall();     //RECALL

if(Input==3){//CLEAR

      PORTD=0b00000000;

      if (ClinicID=1)for(i=0;i<20;++i)Arraykeys1[i]=0;
```

```
    if (ClinicID=2)for(i=0;i<20;++i)Arraykeys1[i]=0;
    if (ClinicID=3)for(i=0;i<20;++i)Arraykeys1[i]=0;
    if (ClinicID=4)for(i=0;i<20;++i)Arraykeys1[i]=0;

    }

if(Input==12){if (flag==0)PORTD=first;
            PORTD=number;
            sound();
            }  //ENTER

}


//display
/***********NO ANSWER FUNCTION**********/

   void NoAnswer(int KeyPressed)
        {
        //save the no. into the Queu
   if (ClinicID=1)
        {          if(Arin<18)
   {
   Arraykeys1[Arin]=KeyPressed;
   sound();
   ++Arin;
   }

   }


   if (ClinicID=2)
        {
   if(Arin<18)
   {
   Arraykeys2[Arin]=KeyPressed;
   sound();
   ++Arin;
   }

   }
   if (ClinicID=3)
        {
```

```
if(Arin<18)
{
Arraykeys3[Arin]=KeyPressed;
sound();
++Arin;
}

}

if (ClinicID=4)
    {
if(Arin<18)
{
Arraykeys4[Arin]=KeyPressed;
sound();
++Arin;
}

}

}
/**********RECALL***************/

    void Recall()
        {
        //Pop the two no. from the queu
        if (ClinicID=1)
        {

    if(Arout<18)
    {
    PORTD=Arraykeys1[Arout];
    sound();
    ++Arout;
    }

        }


    if (ClinicID=2)
        {

    if(Arout<18)
    {
    PORTD=Arraykeys2[Arout];
```

```
sound();
++Arout;
}
     }

if (ClinicID=3)
     {
     if(Arout<18)
{
PORTD=Arraykeys3[Arout];
sound();
++Arout;
}

     }

if (ClinicID=4)
     {

if(Arout<18)
{
PORTD=Arraykeys4[Arout];

++Arout;

}
     }



}


void sound(void)
{
  PWM(1);
  Delay1KTCYx(100);
  PWM(5);
  Delay1KTCYx(100);
  CCP1CON=0b00000000;
}

void PWM(unsigned char i)
{
  CCP1CON=0b00001111;

  T2CONbits.TMR2ON = 1;
  T2CONbits.T2OUTPS3 = 1;
```

```
T2CONbits.T2OUTPS2 =1;
T2CONbits.T2OUTPS1 = 0;
T2CONbits.T2OUTPS0 = 0;
T2CONbits.T2CKPS1 =0;
T2CONbits.T2CKPS1 =0;

PR2 = 255;
TRISCbits.TRISC2=0;
T2CONbits.TMR2ON = 1;
CCPR1L = 5*i;
}
```

The USART code

```
#include <p18f4520.h>
#include <usart.h>
#include <portb.h>
#include <pwm.h>
#include <timers.h>

#pragma config OSC = INTIO67
#pragma config PBADEN = OFF
#pragma config WDT = OFF
#pragma config MCLRE = ON

//void CloseUSART(void);
void DisplayTotal(int);
void DisplayEmergency(void);
void USARTResieving (void);
char Busy2USART( void );
//void DisplayTotal(int);

void main (void)
{
TRISD=0b00000000; //Configure PORTD I/O as output
TRISE=0b00001111; //Configure PORTE I/O as input
ADCON1=0b00001111; // Enable digital I/O
USARTResieving ();

}

void USARTResieving ()
{
int counter=0;
int TotalPatientNo;
int clinicID;
OpenUSART
(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE &
USART_EIGHT_BIT & USART_CONT_RX & USART_BRGH_LOW,77);

while(!counter)
    {
    while (BusyUSART());    //do nothing
    clinicID=ReadUSART();
    PORTE=clinicID;
!counter;

}//while
```

```
while (BusyUSART());    //do nothing

TotalPatientNo=ReadUSART();
if (TotalPatientNo==0)
{
 PORTBbits.RB7=1; //enable the small 7-segments

 switch(clinicID)
     {
 case( 0x00): //Enable the 1st 7-segment
                PORTEbits.RE0=0;
                PORTEbits.RE1=0;
                PORTEbits.RE2=0;

case( 0x01): //Choosing the second clinic

                //Enable the 2nd 7-segment
                PORTEbits.RE0=1;
                PORTEbits.RE1=1;
                PORTEbits.RE2=0;

case(0x02): //Choosing the third clinic

                PORTEbits.RE0=1;
                PORTEbits.RE1=0;
                PORTEbits.RE2=1;

case(0x03)://Choosing the fourth clinic

                PORTEbits.RE0=1;
                PORTEbits.RE1=1;
                PORTEbits.RE2=1;


 if (TotalPatientNo==0)DisplayEmergency();

  DisplayTotal(TotalPatientNo);

  }//switch
 CloseUSART();
 }
 }


  void DisplayEmergency(void)
 {
           PORTD=0;
```

```
    sound();
}
```

The GUI

```
Public Class Form1

    Dim currentTotalsArray As Integer() = {0, 0, 0, 0}
    Dim WithEvents serialPort As New IO.Ports.SerialPort

    Private Sub ClinicsBindingNavigatorSaveItem_Click(ByVal
sender As System.Object, ByVal e As System.EventArgs)
Handles ClinicsBindingNavigatorSaveItem.Click
        Me.Validate()
        Me.ClinicsBindingSource.EndEdit()

Me.ClinicsTableAdapter.Update(Me.ClinicDBDataSet.Clinics)

        If Me.ClinicsBindingSource.Count >= 1 Then
            Me.checkUpdated()
        End If

    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        'TODO: This line of code loads data into the
'ClinicDBDataSet.Clinics' table. You can move, or remove
it, as needed.

Me.ClinicsTableAdapter.Fill(Me.ClinicDBDataSet.Clinics)
        SerialPort1.Open()
    End Sub

#Region "Update functions"
    Private Sub checkUpdated()
        Dim row As DataRowView
        Dim rowIndex As Integer
        Dim totalCurrentNum As Integer
        Dim newCurrentNum As Integer

        For rowIndex = 0 To Me.ClinicsBindingSource.Count -
1

            row = Me.ClinicsBindingSource.Item(rowIndex)
            totalCurrentNum = currentTotalsArray(rowIndex)
            newCurrentNum = row("Total_Number")

            If (totalCurrentNum <> newCurrentNum) Then
```

```
                currentTotalsArray(rowIndex) =
newCurrentNum
                Me.outputSerial(newCurrentNum,
row("ClinicID"))
            End If
        Next

    End Sub

#End Region

#Region "Output functions"
    Private Sub outputSerial(ByVal outTotal As Integer,
ByVal clinicID As Integer)
        'OUTPUT SERIALLY TO MAX
        MsgBox("Transmiting: ClinicID=" & clinicID & "
Total=" & outTotal)

        Try

            SerialPort1.BaudRate = 9600
            SerialPort1.Parity = IO.Ports.Parity.None
            SerialPort1.DataBits = 8
            SerialPort1.StopBits = IO.Ports.StopBits.One


        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
        Try

            'Send the clinic ID
            SerialPort1.Write(clinicID)


        Catch ex As Exception
            MsgBox(ex.Message)
        End Try


    End Sub
#End Region

    Private Sub clrBtn_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles clrBtn.Click
```

```
        Dim userCh As MsgBoxResult

        userCh = MsgBox("Are you sure you want to delete
all values?", MsgBoxStyle.YesNo, "Drop Table")
        Dim row As DataRowView

        If userCh = MsgBoxResult.Yes Then
            For i As Integer = 0 To
Me.ClinicsBindingSource.Count - 1
                row = Me.ClinicsBindingSource.Item(i)

Me.ClinicsTableAdapter.Delete(row("ClinicID"),
row("Total_Number"))
            Next

Me.ClinicsTableAdapter.Fill(Me.ClinicDBDataSet.Clinics)
        End If
    End Sub

    Private Sub TextBox1_TextChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
EmergencyTB.TextChanged


        Dim EmergencyID As Integer

        EmergencyID = Val(EmergencyTB.Text)
        If EmergencyID <= 4 Then
            If EmergencyID >= 0 Then

                Try

                    SerialPort1.BaudRate = 9600
                    SerialPort1.Parity =
IO.Ports.Parity.None
                    SerialPort1.DataBits = 8
                    SerialPort1.StopBits =
IO.Ports.StopBits.One


                Catch ex As Exception
                    MsgBox(ex.Message)
                End Try
                Try
```

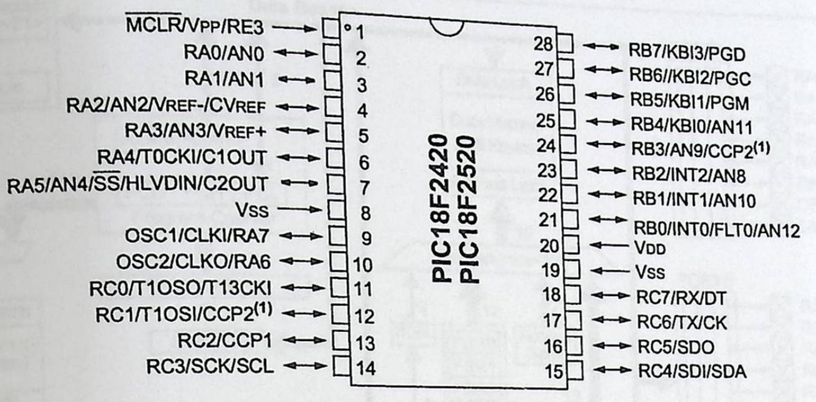
```
                    'Send the clinic ID
                    SerialPort1.Write(EmergencyID)
                    'SerialPort1.Write(0)
                    MsgBox("clinic emergency=" &
EmergencyID)
                Catch ex As Exception
                    MsgBox(ex.Message)
                End Try
            End If
        Else : MsgBox("You have enter a wrong clinic ID")

        End If


    End Sub

    Private Sub LinkLabel1_LinkClicked(ByVal sender As
System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles
LinkLabel1.LinkClicked
        About_Us.Show()

    End Sub

End Class
```

APPENDIX C

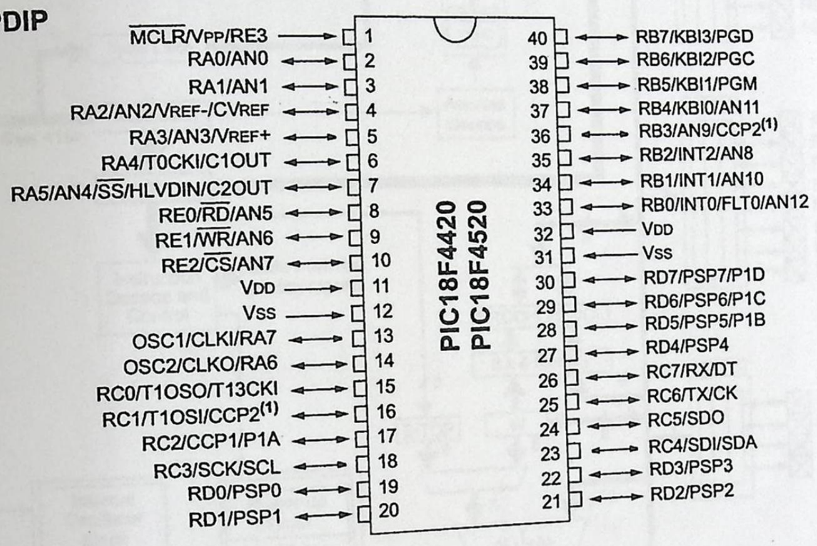
PIC Datasheets

Pin Diagrams

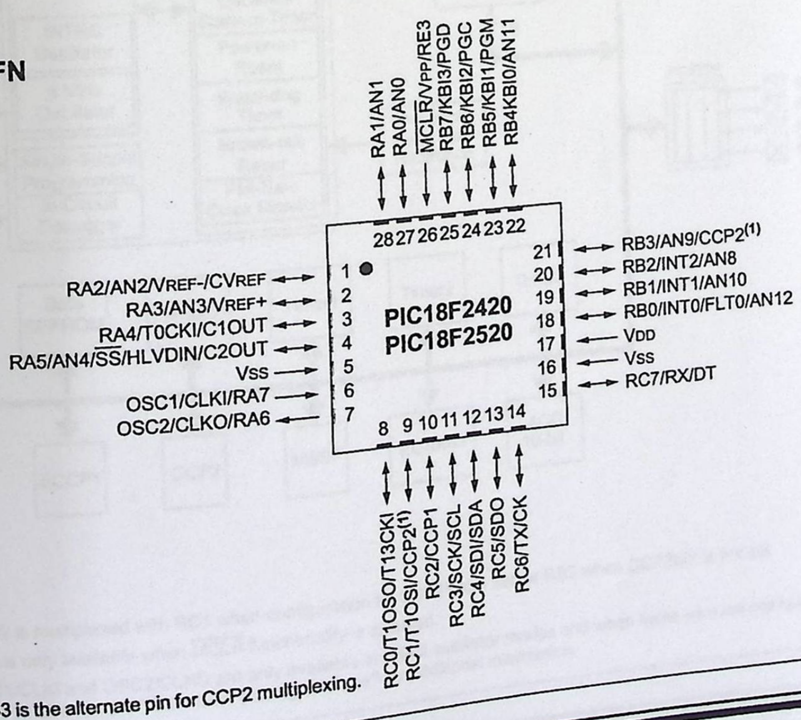
28-pin PDIP, SOIC



40-pin PDIP

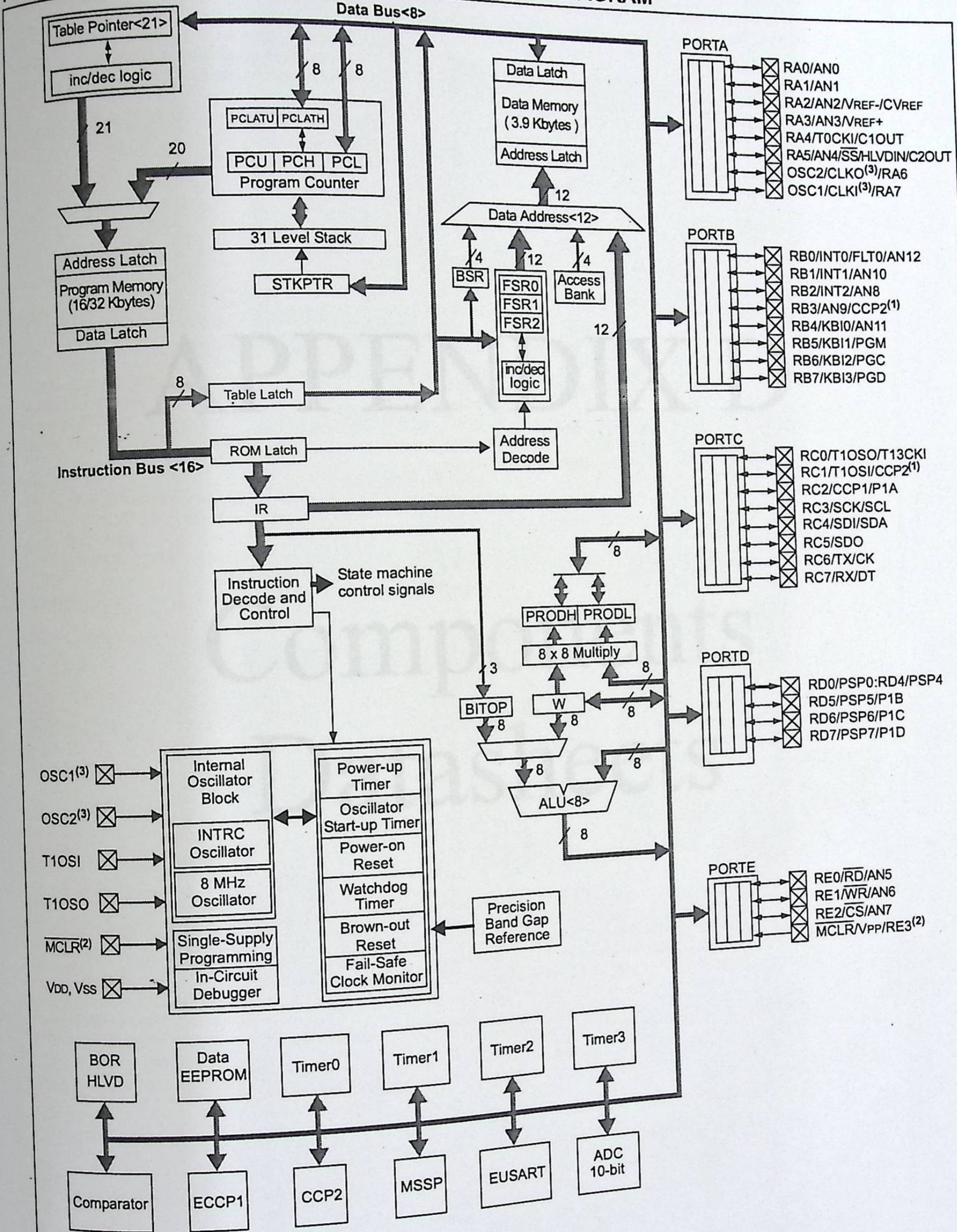


28-pin QFN



Note 1: RB3 is the alternate pin for CCP2 multiplexing.

FIGURE 1-2: PIC18F4420/4520 (40/44-PIN) BLOCK DIAGRAM



Note 1: CCP2 is multiplexed with RC1 when configuration bit CCP2MX is set, or RB3 when CCP2MX is not set.

2: RE3 is only available when $\overline{MCLR}$ functionality is disabled.

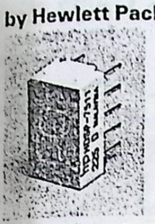
3: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.

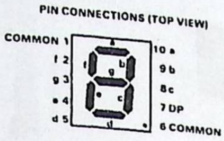
APPENDIX D

Components Datasheets

7·6mm/0·3in. Low Current

by Hewlett Packard





W. 7·62 H. 12·7 D. 5·08
Pin spacing 2·54 Row spacing 5·08

7 segment displays with right-hand decimal points. The AlGaAs versions have a very bright output making them ideal for use in high ambient light conditions. The low current displays have very low power consumption and are compatible with both TTL and CMOS circuitry.

technical specification

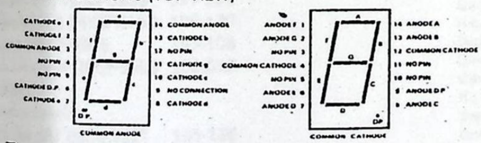
	Hi eff. Red	AlGaAs Red	Green	Low Current Red	Units
V_F (typ.)	2	2	2·1	1·6	V
I_F (typ.)	20	20	20	2	mA
V_R (min.)	3	3	3	3	V
Intensity	5·4	14	3·4	0·27	mcd
View angle ±	50	50	50	50	deg
P_D	105	96	105	52	mW
Operating temperature			-40°C to +85°C		

type	stock no.
common anode	
red (hi eff) HDSP-7501	195-170
AlGaAs red HDSPA151	195-215
green HDSP-7801	195-192
low current HDSP-7511	589-086
common cathode	
red (hi eff) HDSP-7503	195-186
AlGaAs red HDSP-A153	195-221
green HDSP-7803	195-209
low current HDSP-7513	589-092

7·6mm/0·3in.

Quality Technologies

H. 18·75 W. 9·8 D. 5·08
Pin spacing 2·54
Row spacing 7·62
PIN CONNECTIONS (TOP VIEW)



These displays are housed in 14-pin dil packages. The common anode version has a single left-hand decimal point whilst the common cathode version has a single right-hand decimal point.

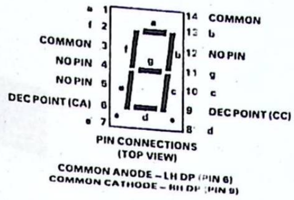
technical specification

	Red	Green	Units
V_F(typ)	1·6	2·2	V
V_R(max)	6	6	V
I_F(typ)	10	10	mA
I_F(max)	30	30	mA
Intensity(per segment typ)	2·5	2·0	mcd
View angle ±	75	75	deg
P_D (per segment max)	60	60	mW
Operating temp.	-40°C to	+85°C	

type	stock no.
common anode	
MAN72A red	587-894
MAN3420A green	587-901
common cathode	
MAN74A red	587-917

10·9mm/0·43in. Low Current

Hewlett Packard



W.12·7 H.19·05 D.6·35 (ex.pins). Pin spacing 2·54 Row spacing 7·62

Low current displays with very low power consumption and compatible with TTL and CMOS circuitry.

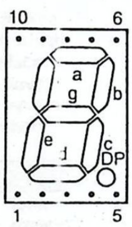
technical specification

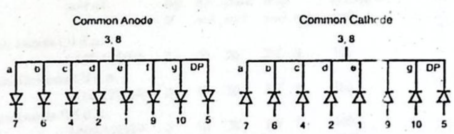
	Hi eff. Red	Green	Low Current Red	Units
V_F (typ.)	2·1	2·1	1·6	V
I_F (typ.)	20	20	2	mA
V_R (max.)	3	3	3	V
Intensity	1115	1750	370	μcd
View angle ±	50	50	50	deg
P_D	105	105	52	mW
Operating temperature		-40°C to +85°C		

type	stock no.
RH decimal point	
common anode	
red (hi eff) 5082-7651	195-158
green HDSP4601	195-164
low current HDSP3351	589-115
common cathode	
red (hi eff) 5082-7653	587-383
green HDSP4603	587-399
low current HDSP3353	589-121
LH decimal point	
common anode	
red (hi eff) 5082-7650	587-175

12·7mm (0·5in.) Light Grey face

Kingbright





H.19 W.12·7 D.8 Pin spacing 2·54 Row spacing 15·24

7 segment LED displays with right hand decimal points. Standard grey face with white segments. Super bright red types have AlGaAs LEDs. For suitable driver ICs refer to the Semiconductors - Drivers and Interface section.

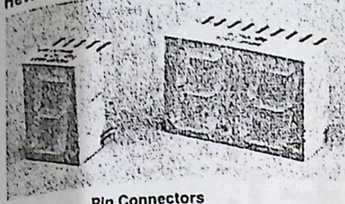
technical specification

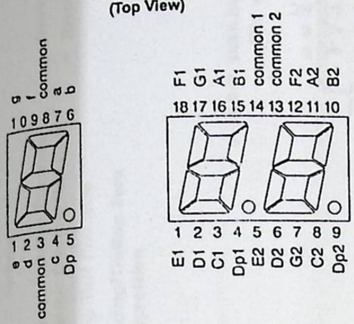
	red (H.E.)	*super-bright red*	*green*	*yellow*	*units*
Kingbright part no.					
Common anode	SA05-11EWA	SA05-11SRWA	SA05-11GWA	SA05-11 WA	
Common cathode	SC05-11EWA	SC05-11SRWA	SC05-11GWA	SC05-11 WA	
V_F (typ.)	2	1·85	2·2	2·1	V
I_F (typ)	20	20	20	20	mA
I_F (max.)	30	30	25	30	mA
V_R (max.)	5	5	5	5	V
Intensity (min.)	2·2	5·6	2·2	2·2	mcd
Intensity (max.)	5·6	21	5·6	5·6	mcd
Power dissipation	105	100	105	105	mW
Operating temperature range -40°C to +85°C					

type	stock no.
common anode	
red (H.E.)	235-8777
super bright red	235-8755
green	235-8799
yellow	235-8828
common cathode	
red (H.E.)	235-8783
super bright red	235-8761
green	235-8812
yellow	235-8834

14·2mm/0·56in. Low Current

Hewlett Packard





Single digit: W.12·57 H.17·2 D.(ex pins)8
Dual digit: W.25·15 H.17·02 D.(ex pins)8

A choice of single and dual digit displays with right-hand decimal points. The AlGaAs versions give a very bright output making them ideal for use in high ambient light conditions. The low current displays have very low power consumption and are compatible with both TTL and CMOS circuitry. Connections are along the top and bottom of the display to simplify wiring in multi-digit applications.

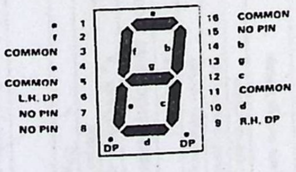
technical specification

	Hi eff. Red	AlGaAs Red	Green	Low Current Red	Units
V_F (typ.)	2·1	1·8	2·1	1·6	V
I_F (typ.)	20	20	20	2	mA
V_R (max.)	3	3	3	3	V
Intensity	2·8	16	2·5	0·37	mcd
View angle ±	50	50	50	50	deg
P_D	105	96	105	52	mW
Operating temperature	-40°C to +85°C				

type	stock no.
single digit	
common anode	
red (hi eff) HDSP5501	587-945
AlGaAs red HDSPH151	195-114
green HDSP5601	195-091
low current HDSP5551	588-623
common cathode	
red (hi eff) HDSP5503	587-951
AlGaAs red HDSPH153	195-120
green HDSP5603	195-108
low current HDSP5553	588-639
dual digit	
common anode	
red (hi eff) HDSP5521	195-136
common cathode	
red (hi eff) HDSP5523	195-142

26

20mm/0·8in.



W. 19·6 D. 8·38 H. 27·69
Pin spacing 2·54 Row spacing 15·24

Two high efficiency, red displays. Both displays have left-hand and right-hand decimal points.

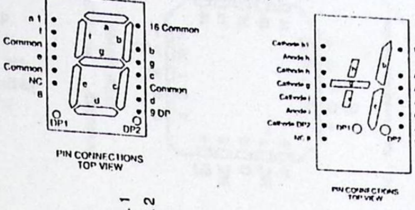
technical specification

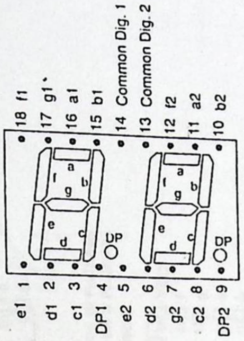
V_F(typ)	1·7V
V_R(max)	3V
I_F(typ)	20mA
I_F(max)	25mA
Intensity/digit (typ)	2·2mcd
P_D/segment (max)	50mW
Operating temperature	-40°C to +85°C

type	stock no.
common anode	
red (Hi eff)	850-653
common cathode	
red (Hi eff)	850-669

20·3mm (0·8in.) Grey Face

Kingbright





PIN CONNECTIONS
Top view

Single digit	*Two digit*
H. 27·7	H. 25·8
W. 20	W. 35·8
D. 8·4	D. 10
Pin spacing 2·54	Pin spacing 2·54
Row spacing 15·24	Row spacing 22

7 segment LED displays, all have right hand decimal points except SA08-12EWA and SC08-12EWA which have left hand decimal points. Standard grey face with white segments. Super bright red types have AlGaAs LEDs. The low current displays have very low power consumption and are compatible with both TTL and CMOS circuitry.

technical specification

Colour	*Kingbright part no.*	*V_F typ. V*	*I_F typ. mA*	*I_F max. mA*	*Intensity mcd min.*	*max.*	*Power dissipation mW*
Common anode LH d.p.							
H.E.red	SA08-12EWA	2	10	30	2·2	9	105
Common cathode LH d.p.							
H.E.red	SC08-12EWA	2	10	30	2·2	9	105
Common anode RH d.p.							
H.E.red	SA08-11EWA	2	10	30	2·2	9	105
Super-bright red	SA08-11SRWA	1·85	10	30	9	21	100
Green	SA08-11GWA	2·2	10	25	1·4	3·6	105
Common cathode RH d.p.							
H.E.red	SC08-11EWA	2	10	30	2·2	9	105
Super-bright red	SC08-11SRWA	1·85	10	30	9	21	100
Green	SC08-11GWA	2·2	10	25	1·4	3·6	105
Yellow	SC08-11YWA	2·1	10	30	1·4	3·6	105
Universal ±1. overflow							
H.E.red	FX08-11EWA	2·2	10	30	2·2	9	105
Low current common anode							
H.E.red	SA08-11LEWA	2	2	7	0·36	0·9	26
Super-bright red	SA08-11LSRWA	1·85	2	30	1·4	5·6	100
Low current common cathode							
H.E.red	SC08-11LEWA	2	2	7	0·36	0·9	26
Super-bright red	SC08-11LSRWA	1·85	2	30	1·4	5·6	100
2-digit common anode							
H.E.red	DA08-11EWA	2	10	30	2·2	9	105
Super-bright red	DA08-11SRWA	1·85	10	30	9	21	100
Common anode							
H.E.red	DE08-11EWA	2	10	30	2·2	9	105
Super-bright red	DC08-11SRWA	1·85	10	30	9	21	100
Green	DC08-11GWA	2·2	10	25	1·4	3·6	105

V_R 5V
Operating temperature range -40°C to +85°C

'THE S.S.M. VALUE EQUALS THE PACK QUANTITY'

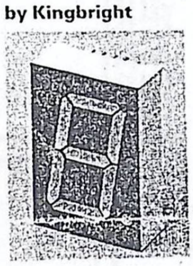
type	stock no.
common anode LH d.p.	
H.E.red	■235-8878
common cathode LH d.p.	
H.E.red	■237-0963
common anode RH d.p.	
H.E.red	235-8862
S-B red *	235-8840
green	■235-8907
common cathode RH d.p.	
H.E.red	235-8884
S-B red *	235-8856
green	■235-8913
yellow	■235-8935
universal	■235-8890
low current common anode	
H.E.red	■247-2722
S-B red *	■247-2738

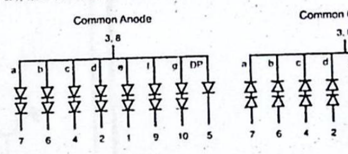
type	stock no.
low current common cathode	
H.E.red	■247-2750
S-B red *	■247-2766
green	■247-2772
2-digit common anode	
H.E.red	■247-2980
S-B red *	■247-2996
2-digit common cathode	
H.E.red	■247-3028
S-B red *	■247-3034
green	247-3040

* S-B = Super-bright

25·4mm (1·0in.) Light Grey Fa

by Kingbright





H. 34 W. 24 D. 10·5
Pin spacing 2·54 Row spacing 30·4

7 segment LED displays with right hand d
points. Each segment has two LED chips
point has one). Standard grey face with w
segments. Super bright red types have A

technical specification

	red (H.E.)	super bright red	green
Kingbright part no.			
Common anode	SA10-21EWA	SA10-21SRWA	
Common cathode	SC10-21EWA	SC10-21SRWA	SC10-21GWA
V_F(typ)	2	1·85	2·2
I_F(typ)	20	20	20
I_F(max)	30	30	25
V_R(max)	5	5	5
Intensity (min)	5·6	14	3·6
Intensity (max)	14	31	9
Power dissipation	105	100	105

Operating temperature range -40°C to +85°C

type	stock no.
Common anode	
red(H.E.)	235-8979
superbright red	235-8941
yellow	■235-9017
Common cathode	
red(H.E.)	235-8985
superbright red	235-8957
green	■235-9001
yellow	■235-9023

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1)	7 V
Input voltage	5.5 V
Maximum current into any output (off-state)	1 mA
Operating free-air temperature range: SN5445 Circuits	−55°C to 125°C
SN7445 Circuits	0°C to 70°C
Storage temperature range	−65°C to 150°C

NOTE 1: Voltage values are with respect to network ground terminal.

recommended operating conditions

	SN5445 MIN	SN5445 NOM	SN5445 MAX	SN7445 MIN	SN7445 NOM	SN7445 MAX	UNIT
Supply voltage, V_{CC}	4.5	5	5.5	4.75	5	5.25	V
Off-state output voltage			30			30	V
Operating free-air temperature, T_A	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

	PARAMETER	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
V_{IH}	High-level input voltage			2			V
V_{IL}	Low-level input voltage					0.8	V
V_{IK}	Input clamp voltage	V_{CC} = MIN, I_I = −12 mA				−1.5	V
$V_{O(on)}$	On-state output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V	$I_{O(on)}$ = 80 mA		0.5	0.9	V
			$I_{O(on)}$ = 20 mA			0.4	
$I_{O(off)}$	Off-state output current	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V, $V_{O(off)}$ = 30 V				250	µA
I_I	Input current at maximum input voltage	V_{CC} = MAX, V_I = 5.5 V				1	mA
I_{IH}	High-level input current	V_{CC} = MAX, V_I = 2.4 V				40	µA
I_{IL}	Low-level input current	V_{CC} = MAX, V_I = 0.4 V				−1.6	mA
I_{CC}	Supply current	V_{CC} = MAX, See Note 2	SN5445		43	62	mA
			SN7445		43	70	

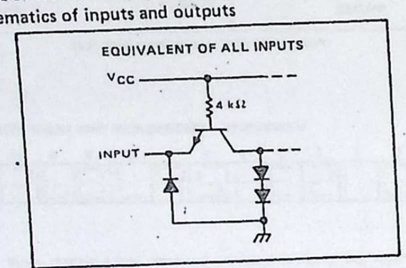
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable type.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
NOTE 2: I_{CC} is measured with all inputs grounded and outputs open.

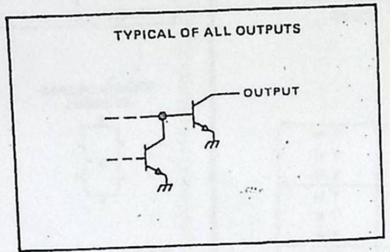
switching characteristics, V_{CC} = 5 V, T_A = 25°C

	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level output	C_L = 15 pF, R_L = 100 Ω, See Note 3			50	ns
t_{PHL}	Propagation delay time, high-to-low-level output				50	ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

schematics of inputs and outputs



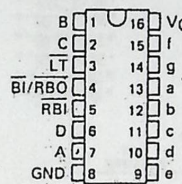


2 TTL Devices

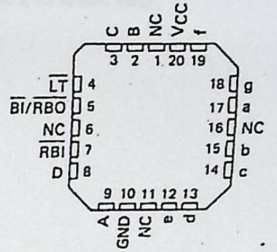
TEXAS INSTRUMENTS

'46A, '47A, 'LS47 feature	'48, 'LS48 feature	'LS49 feature
• Open-Collector Outputs Drive Indicators Directly	• Internal Pull-Ups Eliminate Need for External Resistors	• Open-Collector Outputs
• Lamp-Test Provision	• Lamp-Test Provision	• Blanking Input
• Leading/Trailing Zero Suppression	• Leading/Trailing Zero Suppression	

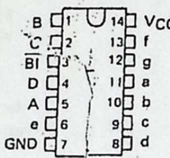
SN5446A, SN5447A, SN54LS47, SN5448, SN54LS48 . . . J PACKAGE
SN7446A, SN7447A, SN7448 . . . N PACKAGE
SN74LS47, SN74LS48 . . . D OR N PACKAGE
(TOP VIEW)



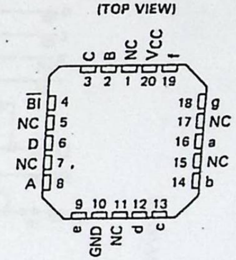
SN54LS47, SN54LS48 . . . FK PACKAGE
(TOP VIEW)



SN54LS49 . . . J OR W PACKAGE
SN74LS49 . . . D OR N PACKAGE
(TOP VIEW)



SN54LS49 . . . FK PACKAGE
(TOP VIEW)



NC – No internal connection

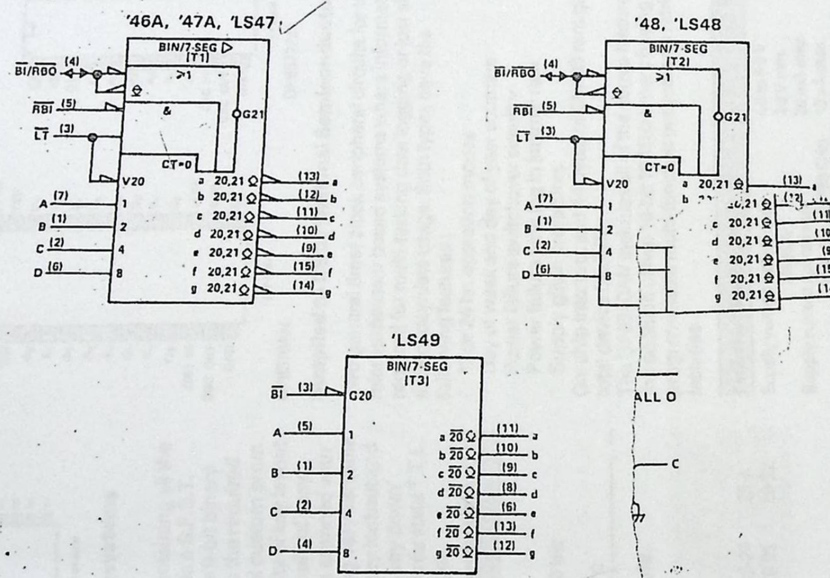
TEXAS INSTRUMENTS

2 TTL Devices

- All Circuit Types Feature Lamp Intensity Modulation Capability

TYPE	DRIVER OUTPUTS				TYPICAL POWER DISSIPATION	PACKAGES
	ACTIVE LEVEL	OUTPUT CONFIGURATION	SINK CURRENT	MAX VOLTAGE		
SN5446A	low	open-collector	40 mA	30 V	320 mW	J, W
SN5447A	low	open-collector	40 mA	15 V	320 mW	J, W
SN5448	high	2-kΩ pull-up	6.4 mA	5.5 V	265 mW	J, W
SN54LS47	low	open-collector	12 mA	15 V	35 mW	J, W
SN54LS48	high	2-kΩ pull-up	2 mA	5.5 V	125 mW	J, W
SN54LS49	high	open collector	4 mA	5.5 V	40 mW	J, W
SN7446A	low	open-collector	40 mA	30 V	320 mW	J, N
SN7447A	low	open-collector	40 mA	15 V	320 mW	J, N
SN7448	high	2-kΩ pull-up	6.4 mA	5.5 V	265 mW	J, N
SN74LS47	low	open-collector	24 mA	15 V	35 mW	J, N
SN74LS48	high	2-kΩ pull-up	6 mA	5.5 V	125 mW	J, N
SN74LS49	high	open-collector	8 mA	5.5 V	40 mW	J, N

logic symbols†



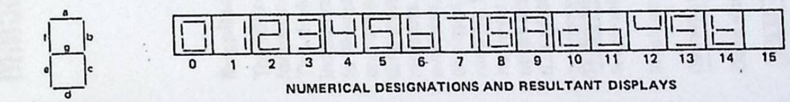
†These symbols are in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12. Pin numbers shown are for D, J, N, and W packages.

description

The '46A, '47A, and 'LS47 feature active-low outputs designed for driving common-anode LEDs or incandescent indicators directly. The '48, 'LS48, and 'LS49 feature active-high outputs for driving lamp buffers or common-cathode LEDs. All of the circuits except 'LS49 have full ripple-blanking input/output controls and a lamp test input. The 'LS49 circuit incorporates a direct blanking input. Segment identification and resultant displays are shown below. Display patterns for BCD input counts above 9 are unique symbols to authenticate input conditions.

The '46A, '47A, '48, 'LS47, and 'LS48 circuits incorporate automatic leading and/or trailing-edge zero-blanking control ($\overline{RBI}$ and $\overline{RBO}$). Lamp test ($\overline{LT}$) of these types may be performed at any time when the $\overline{BI}/\overline{RBO}$ node is at a high level. All types (including the '49 and 'LS49) contain an overriding blanking input ($\overline{BI}$), which can be used to control the lamp intensity by pulsing or to inhibit the outputs. Inputs and outputs are entirely compatible for use with TTL logic outputs.

The SN54246/SN74246 and '247 and the SN54LS247/SN74LS247 and 'LS248 compose the 6 and the 9 with tails and were designed to offer the designer a choice between two indicator fonts.



SEGMENT IDENTIFICATION

NUMERICAL DESIGNATIONS AND RESULTANT DISPLAYS

'46A, '47A, 'LS47 FUNCTION TABLE (T1)

DECIMAL OR FUNCTION	INPUTS						$\overline{BI}/\overline{RBO}$†	OUTPUTS							NOTE
	$\overline{LT}$	$\overline{RBI}$	D	C	B	A		a	b	c	d	e	f	g	
0	H	H	L	L	L	L	H	ON	ON	ON	ON	ON	ON	OFF	
1	H	X	L	L	L	H	H	OFF	ON	ON	OFF	OFF	OFF	OFF	
2	H	X	L	L	H	L	H	ON	ON	OFF	ON	ON	OFF	ON	
3	H	X	L	L	H	H	H	ON	ON	ON	ON	OFF	OFF	ON	
4	H	X	L	H	L	L	H	OFF	ON	ON	OFF	OFF	ON	ON	
5	H	X	L	H	L	H	H	ON	OFF	ON	ON	OFF	ON	ON	
6	H	X	L	H	H	L	H	OFF	OFF	ON	ON	ON	ON	ON	
7	H	X	L	H	H	H	H	ON	ON	ON	OFF	OFF	OFF	OFF	1
8	H	X	H	L	L	L	H	ON	ON	ON	ON	ON	ON	ON	
9	H	X	H	L	L	H	H	ON	ON	ON	OFF	OFF	ON	ON	
10	H	X	H	L	H	L	H	OFF	OFF	OFF	ON	ON	OFF	ON	
11	H	X	H	L	H	H	H	OFF	OFF	ON	ON	OFF	OFF	ON	
12	H	X	H	H	L	L	H	OFF	ON	OFF	OFF	OFF	ON	ON	
13	H	X	H	H	L	H	H	ON	OFF	OFF	ON	OFF	ON	ON	
14	H	X	H	H	H	L	H	OFF	OFF	OFF	ON	ON	ON	ON	
15	H	X	H	H	H	H	H	OFF	OFF	OFF	OFF	OFF	OFF	OFF	
$\overline{BI}$	X	X	X	X	X	X	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	2
$\overline{RBI}$	H	L	L	L	L	L	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	3
$\overline{LT}$	L	X	X	X	X	X	H	ON	ON	ON	ON	ON	ON	ON	4

H = high level, L = low level, X = irrelevant

NOTES: 1. The blanking input ($\overline{BI}$) must be open or held at a high logic level when output functions 0 through 15 are desired. The ripple-blanking input ($\overline{RBI}$) must be open or high if blanking of a decimal zero is not desired.

2. When a low logic level is applied directly to the blanking input ($\overline{BI}$), all segment outputs are off regardless of the level of any other input.

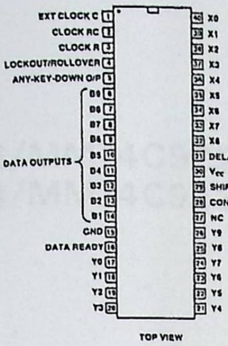
3. When ripple-blanking input ($\overline{RBI}$) and inputs A, B, C, and D are at a low level with the lamp test input high, all segment outputs go off and the ripple-blanking output ($\overline{RBO}$) goes to a low level (response condition).

4. When the blanking input/ripple blanking output ($\overline{BI}/\overline{RBO}$) is open or held high and a low is applied to the lamp-test input, all segment outputs are on.

†$\overline{BI}/\overline{RBO}$ is wire AND logic serving as blanking input ($\overline{BI}$) and/or ripple-blanking output ($\overline{RBO}$).

Keyboard Encoder IC's

9600-PRO



Supplied to RS by Standard Microsystems Corp.

The 9600 is a keyboard encoder i.c. containing all the logic and debounce circuitry to encode a S.P.S.T. keyboard array. The output is a simple 9-bit binary code which can easily be converted to the required code information by use of an external custom prom or microprocessor. Maximum flexibility of key layout and output coding with the added benefit of easy modification is achieved. The keys are scanned with a nine output by ten input matrix giving very versatile keyboard options. Pin selection of N-key lockout and N-key rollover is possible and an 'any-key down' output is also available. Outputs are three state T.T.L. compatible, 40-pin d.i.l. plastic package.

Technical Specification

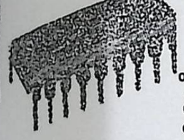
Supply voltage	+5 V d.c.
Supply current	40 mA max.
Clock frequency	10 kHz to 100 kHz
Contact resistance	300 Ω max.
Operating temperature range	0 °C to +70 °C

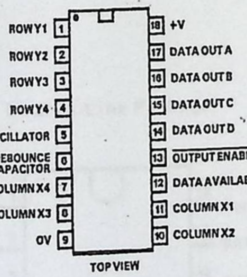
Data sheet 6812 March 91 available.

S.S.M.=1

stock no.	price each 1-9	10-24	25+
633-161	£7·92	£6·91	£6·22

16-Key MM74C922N





Supplied to RS by National Semiconductors

A C-MOS keyboard switch encoder i.c. incorporating all the logic necessary to fully encode an array of up to 16 S.P.S.T. switches (normally open) into a natural binary code. The switches normally arranged in a 4×4 matrix are sequentially scanned at a rate determined by either the internal clock, the frequency of which, being determined by an external capacitor or alternatively an external oscillator, may be used resulting in full synchronisation within a system. The circuit automatically debounces the switches although this function can be disabled if not required. Internal latches store the last entry made even after the key is released. The outputs are tri-state, allowing expansion and bus orientated operation. Internal pull-up resistors on key inputs permit switches with up to 50 kΩ 'on' resistance to be connected directly to the device. All outputs are low power Schottky T.T.L. compatible. Operating supply voltage: +3 to +15 V (18 V max.). Operating temperature range: -40 °C to +85 °C. 18-pin d.i.l. plastic package. See also suitable keyboard 337-100 or 334-410, in the Switches section.

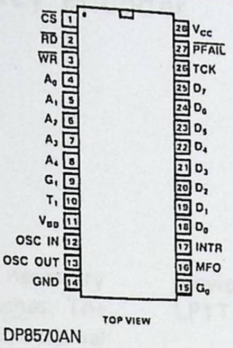
Data sheet 3374 November 83 available.

S.S.M.=1

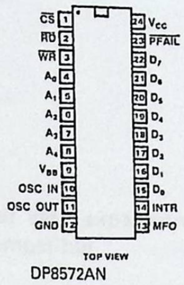
stock no.	price each 1-9	10-24	25+
[illegible]	[illegible]	£4·74	£2·97

Microprocessor Compatible Real-Time Clock ICs

DP8570AN and DP8572AN



DP8570AN



DP8572AN

Supplied to RS by National Semiconductor

Two general timer clock peripheral circuits for use in microprocessor based systems where information is required for multi-tasking data logging or just simple time of day/date usage. Both types have the following features :

- 12 or 24 hr. operation modes
- Day of week and day of year counters
- Power failure switchover circuitry
- Power failure time log in internal ram
- Supply glitch protection.

On-chip interrupt and 44 bytes of CMOS ram give total device flexibility.

The DP8570AN includes all of the above features with the addition of two 16-bit 10 MHz timers having programmable multi-function outputs and retrigger facilities.

Technical Specification

Supply voltage operational	4·5 to 5·5 V
standby	2·2 V min.
Supply current operational	20 mA max.
standby (32·768 KHz)	10 μA max.
Operating temperature range	-40°C to +85°C
Crystal frequencies	32 kHz, 32·768 kHz 4·194304 MHz or 4·9152 MHz

S.S.M.=1

type	stock no.	price each 1-24	25-99	100+
DP8570AN	658-176	£20·10	£13·60	£10·30
DP8572AN	658-182	£14·00	£9·50	£7·15

Need More Information?

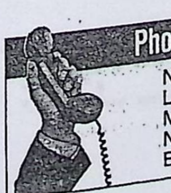
The **RS** Data Library forms a unique source of detailed technical data on thousands of **RS** products.

Containing hundreds of data sheets written by experienced technical engineers, the **RS** Data Library expands greatly on our concise catalogue information. The library combines manufacturers' data with true 'hands on' applications experience and the wealth of facilities in the **RS** laboratory.

Refer to the Technical Books, section 52, for more details.

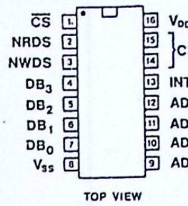
Part 1 Section 52

Phone the Orderline



National	0536 201201
London Area	081-360 8600
Midland	021-359 4900
North West	061-477 8400
Electromail	0536 204555

MM58174AN



Supplied to RS by National Semiconduct

A C-MOS device designed for use as a real tim and calendar in bus orientated microprocesso digital systems. An interrupt timer is included may be programmed to interrupt at 60, 5·0 or 0 second intervals. The timebase is generated fr 32·768 kHz crystal **RS** stock no. 304-447, with timekeeping maintained down to 2·2 V for batt back-up operation. 12 registers contain the foll data: tenths, units and tens of seconds, units a tens of minutes, units and tens of hours, units tens of days, units and tens of months. Autom leap year calculation is also featured. These re may be programmed or read via the 4-bit data when correctly addressed by the 4 register ad lines. 16-pin d.i.l. plastic package.

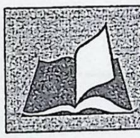
Technical Specification

Supply voltage	
Standby	2·2 V minimum
Operational	4·0 to 6·0 V
Supply current	
Standby	4 μA typ. 2·2 V
Operational	1 mA typ. with 5 V sup
Operating temperature range	0°C to +70°C

Data sheet 4298 March 86 available.

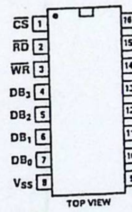
S.S.M.=1

stock no.	price each 1-9	10-24
304-548	£11·80	£9·88



For I²C interface Real Tim Clock, please refer to RS no. 296-683 in the I²C ran

MM58274CN



Supplied to RS by National Semiconduc

The 58274 is a complete real-time clock i.c. d for use in bus orientated microprocessor sys This device is pin compatible with the 58174, incorporating the added features of timekeep to 99 years, faster access times, improved da and extra interrupt periods. The extended timekeeping includes units and tens of years registers, 12 and 24 hour counting is also no available. To simplify data reading a testable changed flag is included allowing error free reading. Basic operation is similar to the 581 however different data and addressing is rec utilise the new features included in the 5827 d.i.l. plastic package.

Technical Specification

Supply voltage: standby	2·2 V minimum
operational	4·5 to 5·5 V
Supply current: standby	4 μA typ. at 2·2 V
operational	1 mA typ. at 5 V
Operating temperature range	-40°C to +85°C

Data sheet 5875 July 85 available.

S.S.M.=1

stock no.	price each 1-9	10-24
659-337	£6·80	£4·90

MM54C922/MM74C922 16 key encoder
MM54C923/MM74C923 20 key encoder

general description

These CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 kΩ on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this two key roll over is provided between any two switches.

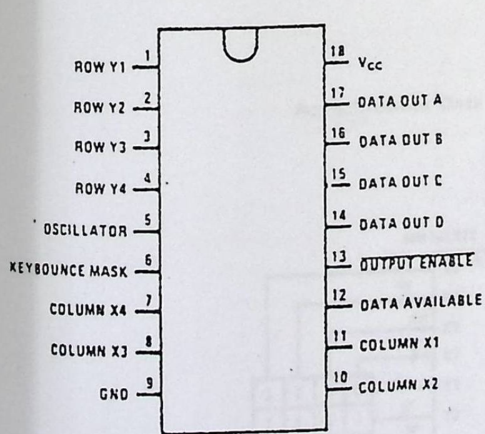
n internal register remembers the last key pressed. en after the key is released. The TRI-STATE® outputs provide for easy expansion and bus operation and are LPTTL compatible.

features

- 50 kΩ maximum switch on resistance
- On or off chip clock
- On chip row pull-up devices
- 2 key roll-over
- Keybounce elimination with single capacitor
- Last key register at outputs
- TRI-STATE outputs LPTTL compatible
- Wide supply range 3V to 15V
- Low power consumption

onnection diagrams

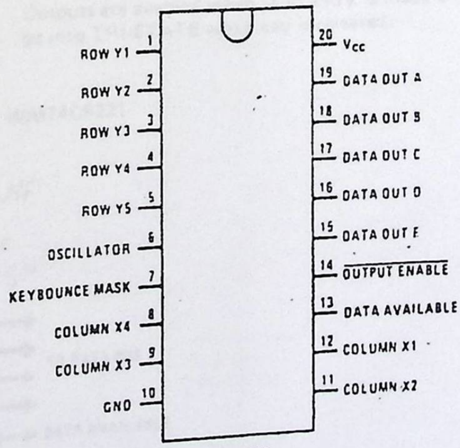
Dual-In-Line Package



TOP VIEW

Order Number MM54C922N or MM74C922N See Package 20

Dual-In-Line Package

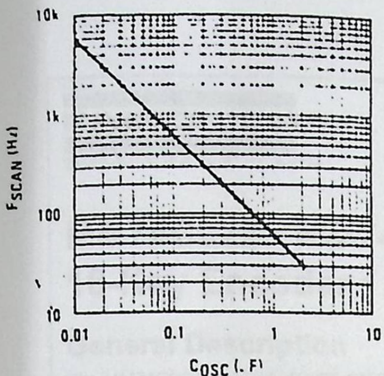


TOP VIEW

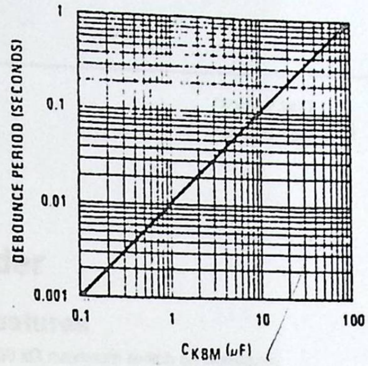
Order Number MM54C923N or MM74C923N See Package 20A

typical performance characteristics (con't)

Typical F_{SCAN} vs C_{OSC}

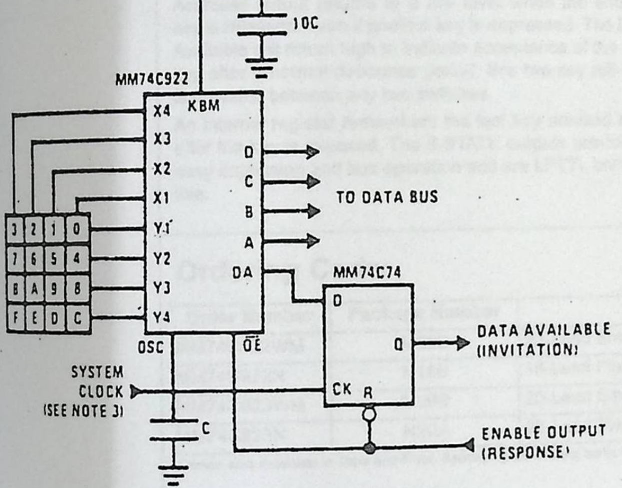


Typical Debounce Period vs C_{KBM}

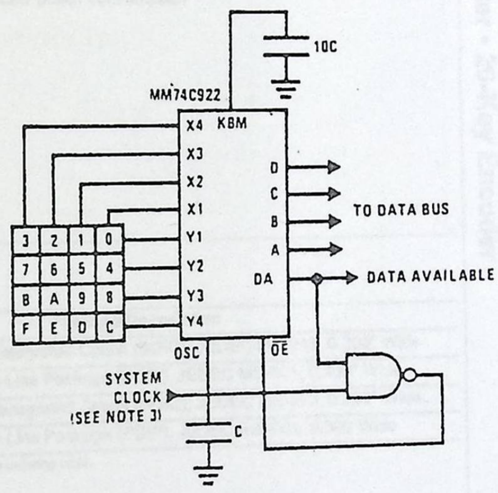


typical applications

Synchronous Handshake (MM74C922)

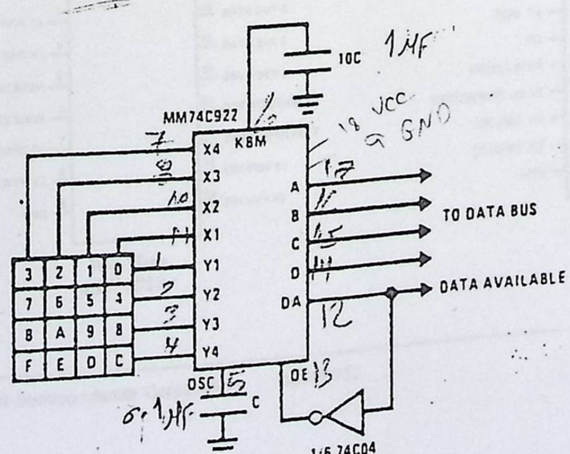


Synchronous Data Entry Onto Bus (MM74C922)



Outputs are enabled when valid entry is made and go into TRI-STATE when key is released.

Asynchronous Data Entry Onto Bus (MM74C922)



Outputs are in TRI-STATE until key is pressed, then data is placed on bus. When key is released, outputs return to TRI-STATE.

Note 3: The keyboard may be synchronously scanned by omitting the capacitor at osc. and driving osc. directly if the system clock rate is lower than 10 kHz.

FAIRCHILD
SEMICONDUCTOR™

October 1987
Revised April 2001

MM74C922 • MM74C923
16-Key Encoder • 20-Key Encoder

General Description

The MM74C922 and MM74C923 CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 kΩ on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this two-key roll-over is provided between any two switches.

An internal register remembers the last key pressed even after the key is released. The 3-STATE outputs provide for easy expansion and bus operation and are LPTTL compatible.

Features

- 50 kΩ maximum switch on resistance
- On or off chip clock
- On-chip row pull-up devices
- 2 key roll-over
- Keybounce elimination with single capacitor
- Last key register at outputs
- 3-STATE output LPTTL compatible
- Wide supply range: 3V to 15V
- Low power consumption

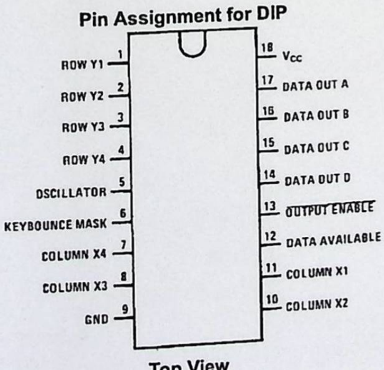
Ordering Code:

Order Number	Package Number	Package Description
MM74C922WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300" Wide
MM74C922N	N18B	18-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide
MM74C923WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300" Wide
MM74C923N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Device also available in Tape and Reel. Specify by appending suffix letter "X" to the ordering code.

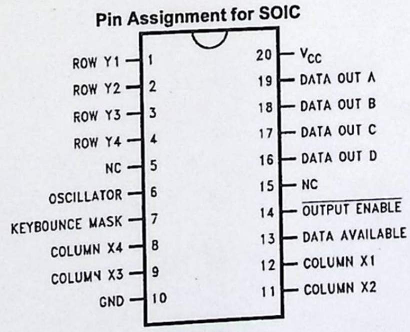
Connection Diagrams

Pin Assignment for DIP



Top View
MM74C922

Pin Assignment for SOIC



Top View
MM74C922

 DS006037

FAIRCHILD
SEMICONDUCTOR™

August 1986
Revised February 2000

DM7404
Hex Inverting Gates

General Description

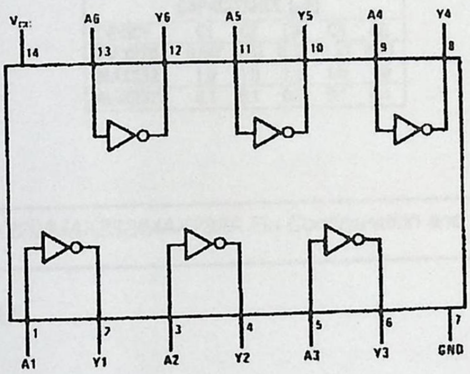
This device contains six independent gates each of which performs the logic INVERT function.

Ordering Code:

Order Number	Package Number	Package Description
DM7404M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-012, 0.150 Narrow
DM7404N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

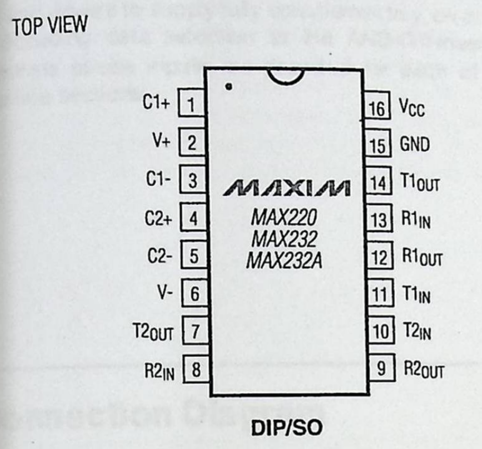
$Y = \overline{A}$

Inputs	Output
A	Y
L	H
H	L

H = HIGH Logic Level
L = LOW Logic Level

 DS006494

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



CAPACITANCE (μF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

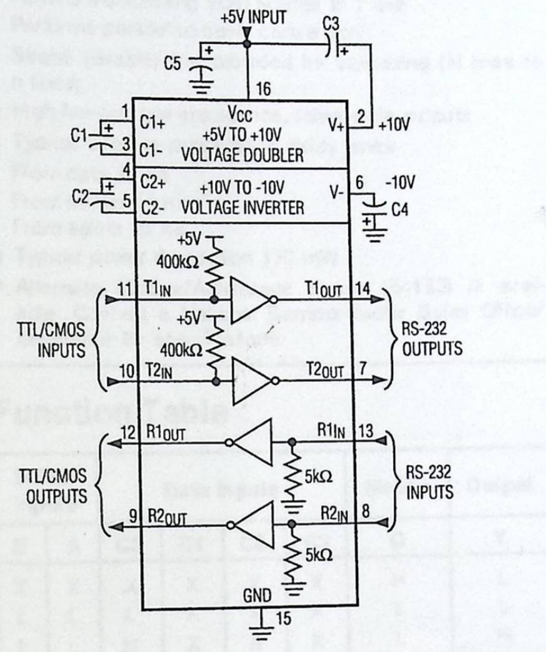
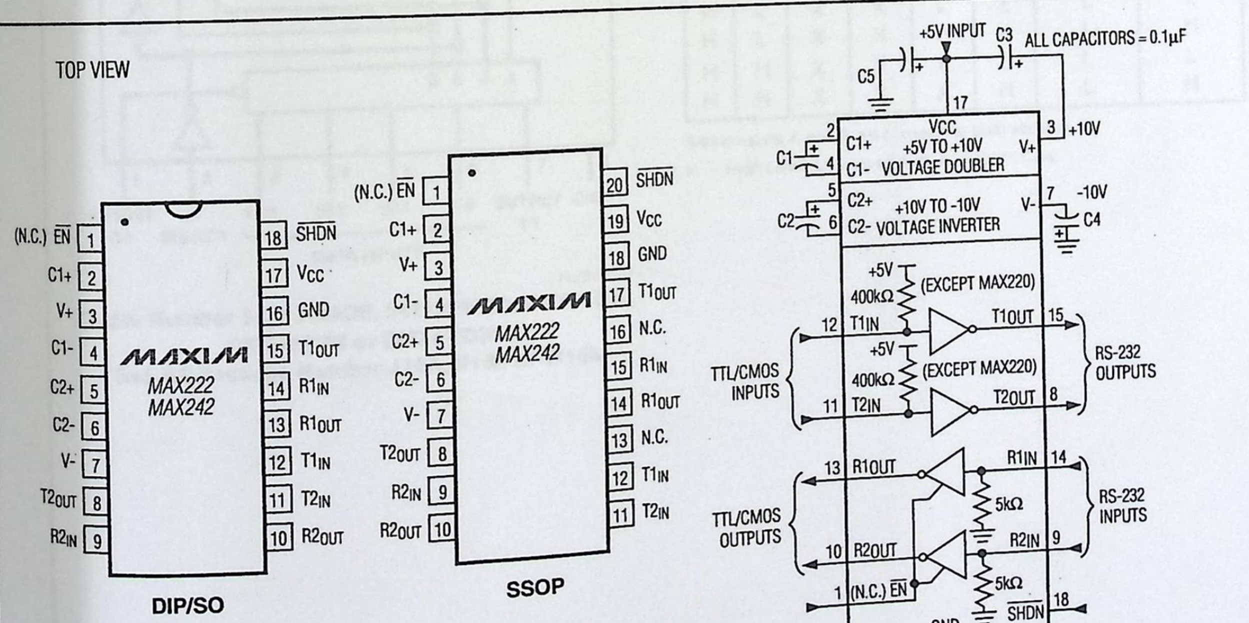


Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit



( ) ARE FOR MAX222 ONLY.
PIN NUMBERS IN TYPICAL OPERATING CIRCUIT ARE FOR DIP/SO PACKAGES ONLY.

Figure 6. MAX222/MAX242 Pin Configurations and Typical Operating Circuit

National Semiconductor

June 1989

54153/DM54153/DM74153 Dual 4-Line to 1-Line Data Selectors/Multiplexers

General Description

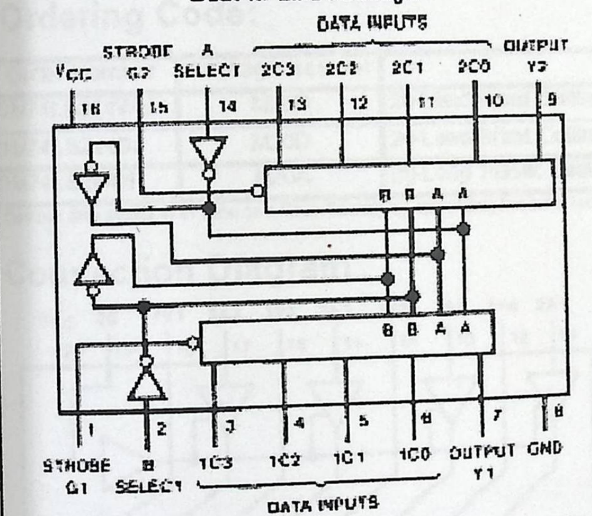
Each of these data selectors/multiplexers contains inverters and drivers to supply fully complementary, on-chip, binary decoding data selection to the AND-OR-invert gates. Separate strobe inputs are provided for each of the two four-line sections.

Features

- Permits multiplexing from N lines to 1 line
- Performs parallel-to-serial conversion
- Strobe (enable) line provided for cascading (N lines to n lines)
- High fan-out, low-impedance, totem-pole outputs
- Typical average propagation delay times
  From data 11 ns
  From strobe 18 ns
  From select 20 ns
- Typical power dissipation 170 mW
- Alternate Military/Aerospace device (54153) is available. Contact a National Semiconductor Sales Office/Distributor for specifications.

Connection Diagram

Dual-In-Line Package



TL/F/6547–1

Order Number 54153DMQB, 54153FMQB, DM54153J, DM54153W or DM74153N
See NS Package Number J16A, N16E or W16A

Function Table

Select Inputs		Data Inputs				Strobe	Output
B	A	C0	C1	C2	C3	G	Y
X	X	X	X	X	X	H	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
L	H	X	L	X	X	L	L
L	H	X	H	X	X	L	H
H	L	X	X	L	X	L	L
H	L	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

Select inputs A and B are common to both sections.

H = High Level, L = Low Level, X = Don't Care

RRD-B30M105/Printed in U.S.A.

FAIRCHILD
SEMICONDUCTOR™

August 1986
Revised March 2000

DM74LS244
Octal 3-STATE Buffer/Line Driver/Line Receiver

General Description

These buffers/line drivers are designed to improve both the performance and PC board density of 3-STATE buffers/drivers employed as memory-address drivers, clock drivers, and bus-oriented transmitters/receivers. Featuring 400 mV of hysteresis at each low current PNP data line input, they provide improved noise rejection and high fanout outputs and can be used to drive terminated lines down to 133Ω.

Features

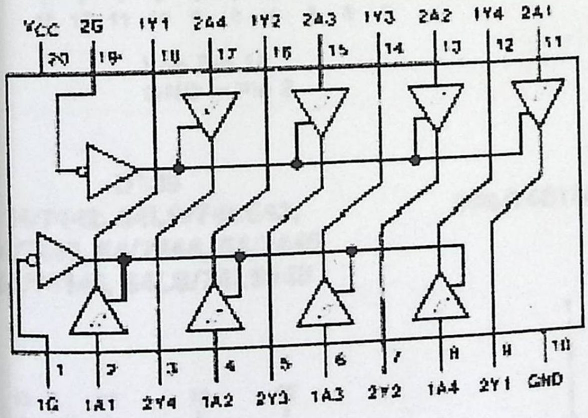
- 3-STATE outputs drive bus lines directly
- PNP inputs reduce DC loading on bus lines
- Hysteresis at data inputs improves noise margins
- Typical I_{OL} (sink current) 24 mA
- Typical I_{OH} (source current) −15 mA
- Typical propagation delay times
  - Inverting 10.5 ns
  - Noninverting 12 ns
- Typical enable/disable time 18 ns
- Typical power dissipation (enabled)
  - Inverting 130 mW
  - Noninverting 135 mW

Ordering Code:

Order Number	Package Number	Package Description
DM74LS244WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS244SJ	M20D	20-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS244N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

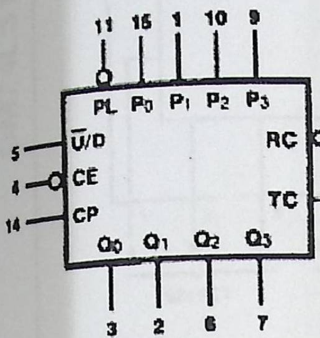
Inputs		Output
$\overline{G}$	A	Y
L	L	L
L	H	H
H	X	Z

L = LOW Logic Level
H = HIGH Logic Level
X = Either LOW or HIGH Logic Level
Z = High Impedance

 DS006442

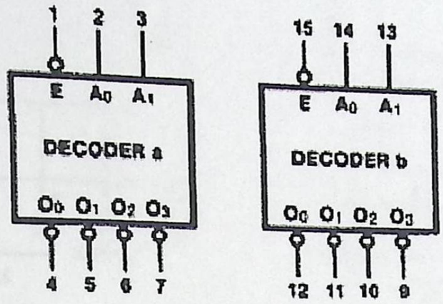
DIGITAL - TTL

D130
54/74190, 74LS190
54/74191, 74LS191



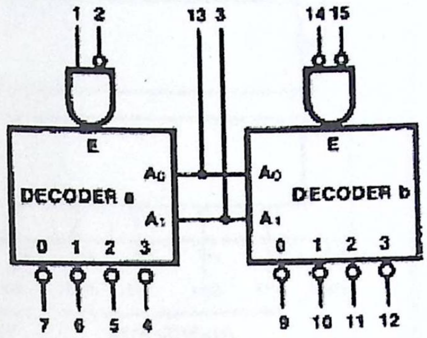
Vcc = Pin 16
GND = Pin 8

D131
9321, 93L21,
54/74S139, 54LS/74LS139



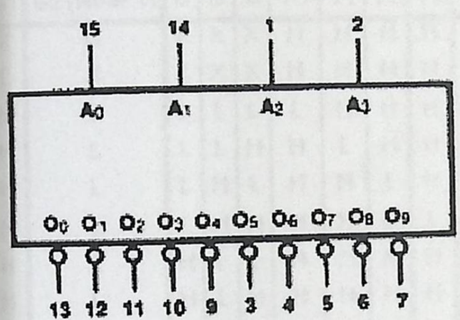
Vcc = Pin 16
GND = Pin 8

D132
54/74155, 54LS/74LS155
54/74156, 54LS/74LS156



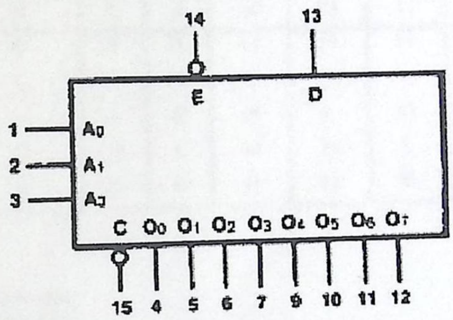
Vcc = Pin 16
GND = Pin 8

D133
9301, 93L01, 9302



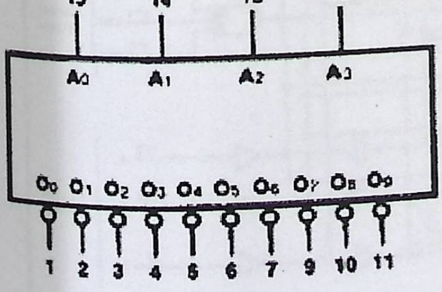
Vcc Pin 16
GND = Pin 8

D134
9334, 93L34, 54LS/74LS259



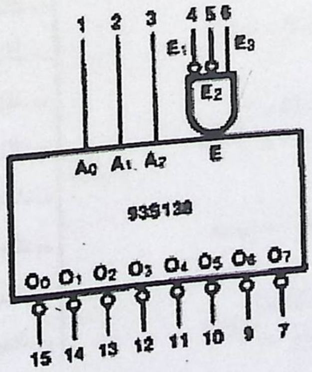
Vcc = Pin 16
GND = Pin 8

D135
54/7442, 54LS/74LS42,
54/7443, 54/7444, 54/7445
54/74145, 54LS/74LS145



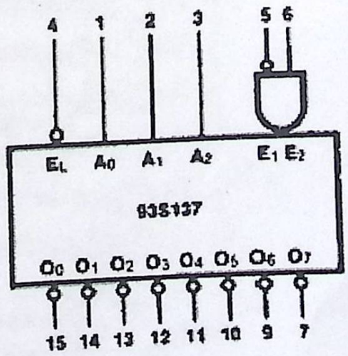
Vcc = Pin 16
GND = Pin 8

D136
54S/74S138, 54LS/74LS138



Vcc = Pin 16
GND = Pin 8

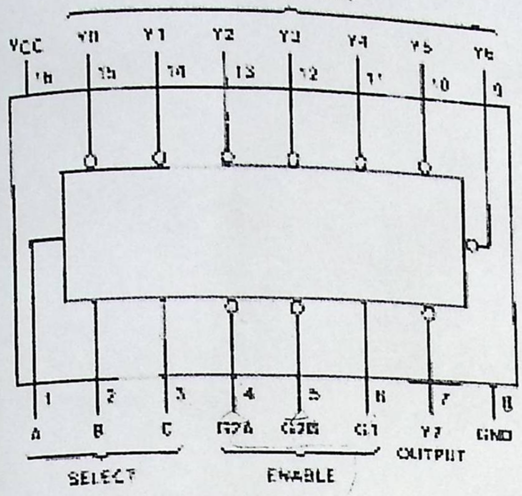
D137
93S137



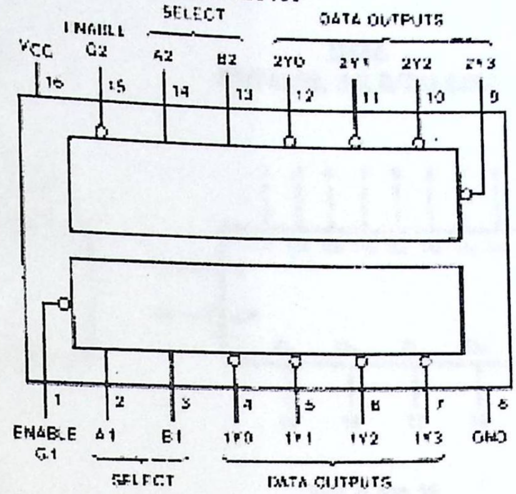
Vcc = Pin 16
GND = Pin 8

Connection Diagrams

DM74LS138



DM74LS139



Function Tables

DM74LS138

Inputs					Outputs							
Enable		Select										
G1	G2 (Note 1)	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

DM74LS139

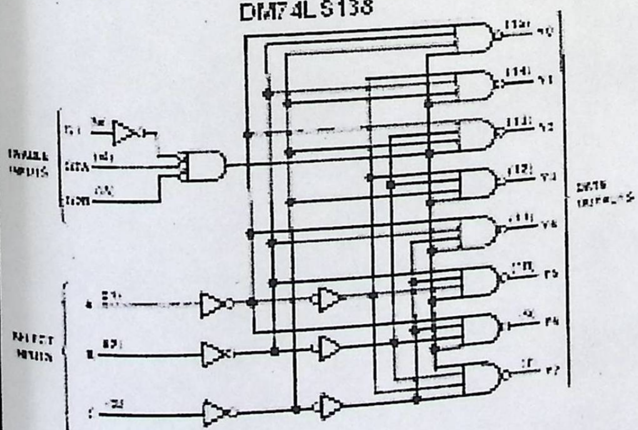
Inputs			Outputs			
Enable	Select					
G	B	A	Y0	Y1	Y2	Y3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

H = HIGH Level
L = LOW Level
X = Don't Care

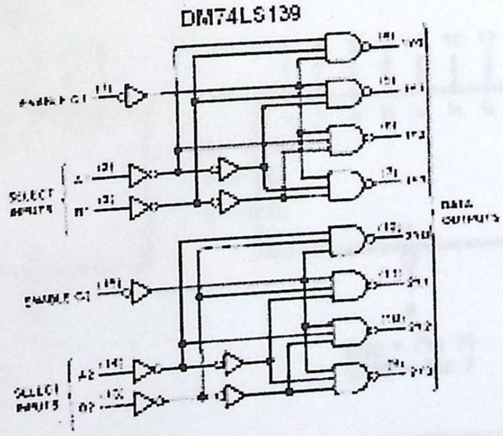
Note 1: G2 = G2A + G2B

Logic Diagrams

DM74LS138

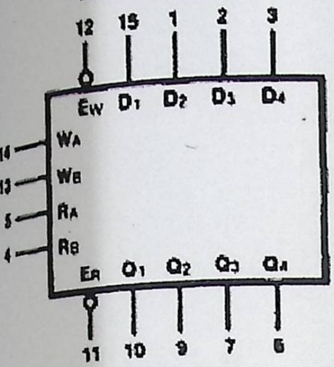


DM74LS139



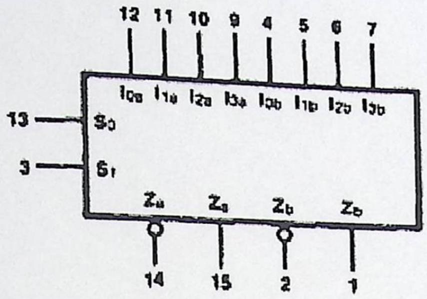
DIGITAL - TTL

D154
54/74170, 54LS/74LS170, 54LS/74LS670



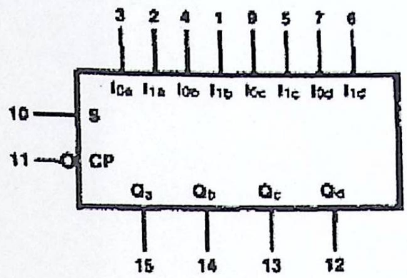
Vcc = Pin 16
GND = Pin 8

D155
9309, 93L09



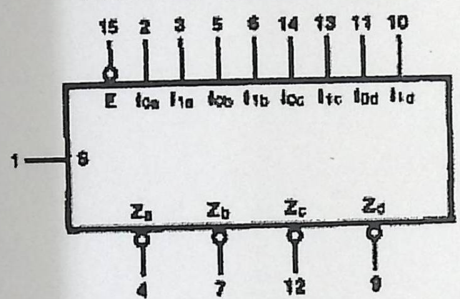
Vcc = Pin 16
GND = Pin 8

D156
54/74298, 54LS/74LS298



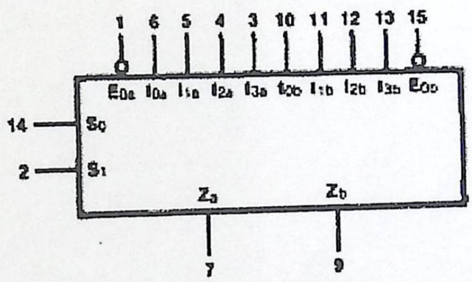
Vcc = Pin 16
GND = Pin 8

D157
9322, 93L22, 54/74157, 54S/74S157, 54LS/74LS157, 54S/74S158, 54LS/74LS158, 54S/74S257, 54LS/74LS257, 54S/74S258, 54LS/74LS258



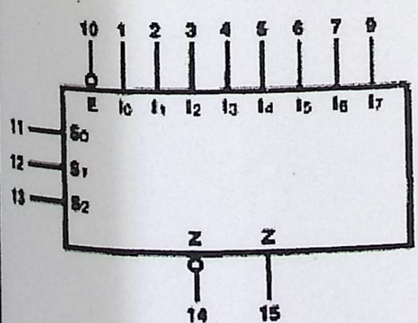
Vcc = Pin 16
GND = Pin 8

D158
54/74153, 54S/74S153, 54LS/74LS153, 54S/74S253, 54LS/74LS253



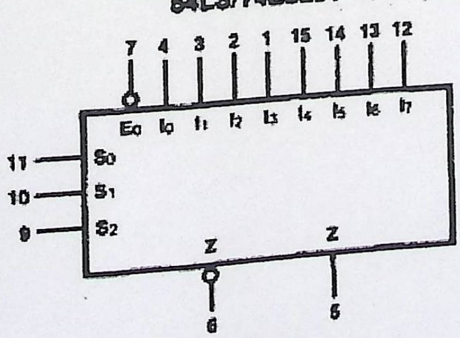
Vcc = Pin 16
GND = Pin 8

D159
9312, 93L12, 93S12, 9313



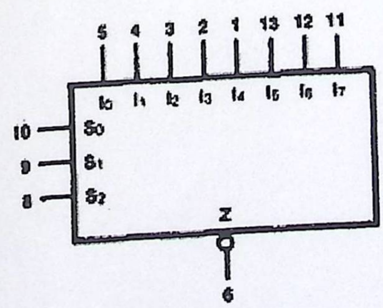
Vcc = Pin 16
GND = Pin 8

D160
54/74151A, 54S/74S151, 54LS/74LS151, 54S/74S251, 54LS/74LS251



Vcc = Pin 16
GND = Pin 8

D161
54/74152A, 54LS/74LS152



Vcc = Pin 14
GND = Pin 7