DESIGN AND CONSTRUCTION OF A DIGITAL CLOCK AND CALENDAR WITH REMINDER FUNCTIONS

BY

OBIDIKE CHINENYE IJEOMA

2004/18992EE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, FEDERAL UNIVERSITY OF TECHNOLOGY MINNA, NIGER-STATE, NIGERIA.

DECEMBER, 2009.

DESIGN AND CONSTRUCTION OF A DIGITAL CLOCK AND CALENDAR WITH REMINDER FUNCTIONS

BY

OBIDIKE CHINENYE IJEOMA

2004/18992EE

A THESIS SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, FEDERAL UNIVERSITY OF TECHNOLOGY MINNA, NIGER-STATE, NIGERIA.

DECEMBER, 2009.

DEDICATION

I hereby dedicate this project to the Almighty God , my father above, the creator of Heaven and Earth, the one who moulds and shapes the destines of men; it is him that deserves all thanks and praise. Also to my lovely dearest parents, Mr. and Mrs. Chris Obidike, for the solid foundation they gave to me and to my lovely siblings.

DECLARATION

I, Obidike Chinenye Ijeoma, declare that this work was done by me and has never been presented elsewhere for the award of a degree. I also hereby relinquish the copyright to the Federal University of Technology, Minna.

OBIDIKE CHINENYE I.

Name of Student

18/12/2009

Signature and Date

MR. USMAN GALADIMA

Name of Supervisor

18/12/09

Signature and Date

Dr. Y.B ADEDIRAN

Name of H.O.D

(May 6, 2010)

Signature and Date

Name of External Examiner

Signature and date

ACKNOWLEDGEMENT

To my father in heaven, the creator of Heaven and Earth, the one who moulds and shapes the destinies of men; it is you that deserve all the praise and glory, for the favor, protection and honor you have bestowed upon me at this point in my life…God, I bless your name. Jesus, I thank you.

My parents; Mr. & Mrs. Obidike, words are definitely not enough to appreciate and thank you for your prayers, support and the trust you both have given to me. I look at both of you and you give me every reason to want to excel in life, you both undoubtedly are the best. I love you both. My prayer; is that I continue to be a source of joy to you.

What can I do without my siblings; Ugochukwu, Ofodile, Ifeoma, Uche and Emeka, the five of you are my source of inspiration, my joy and my hope. I love you guys eternally. Together, the six of us are going to affect our generation positively and we are definitely bound to be at the top.

To my amiable, principled and incorruptible Head of Department, Dr. Yunusa Adediran, you are one in a million, I thank you sir for the discipline and sense of responsibility you have instilled in my colleagues and I. God bless you and your family sir.

My project supervisor, Mr. Usman Nuhu Galadima, meeting you has taken me step closer to my destiny; you are a friend and a brother. I thank you for your assistance, support and encouragement throughout the course of executing this project. God bless you sir.

ABSTRACT

The design and construction of a microcontroller-based digital clock and calendar with reminders function as described in this project report is intended to produce a simple means of keeping and displaying Reminder functions in a better and More accurate way. The project is divided into six main different modules, namely: Power unit, Control unit, Keypad unit, Memory unit, Display unit and Alarm unit. Power unit consist of dual source, the control unit has a microcontroller that was Programmed in assembly language, the keypad was interfaced with the Microcontroller for inputting information, Memory unit is for storing the text Information. Display unit is used for visual interaction and the alarm unit is used to alert the user when the reminder is set.

Table of contents

5 CHAPTER FIVE: CONCLUSION AND RECOMMENDATION

LIST OF FIGURE

CHAPTER ONE

GENERAL INTRODUCTION

The design and construction of a microcontroller based digital alarm clock and calendar with reminder function is used to provide a simple modern way of recalling a fixed and specific function of a particular day and time with low power consumption, reduce complexity and an at affordable price. This model is designed in such a way that any individual can employ its services because of its user friendliness.

Due to various activities of man in our modern and busy world today, it will be quite difficult for man to always recall his time schedules and appointment, but with this project, such difficulties could easily be eliminated. Time telling is an essential aspect of human life. Human beings generally live by time because all his activities are based on time passage. The idea of the time passage that was calculated and recorded in the past was based on rotational characteristics of some useful astronomical or celestial bodies [1]. Celestial bodies include the sun, moon, planets and stars. They have provided us a reference for measuring the passage of time throughout our existence.

Calendar has its traces down to the measurement of seconds in the time measurement [2]. According to the oxford advanced learner's dictionary, Calendar is a system by which time is divided into fixed periods showing the beginning and end of a year. A calendar is a system of measuring time for the needs of civil life by dividing time into days, weeks, months, and year.

Calendar division was completely based on the movement of the sun and moon. In that respect, a day is the average time required for one complete rotation of the earth about its axis. In a similar vein, a month is the average time measurement of the month to revolve completely round the sun in one cycle. One year measurement was based one complete revolution of the earth in cycle around the sun. Such an account which results to a complete one year seasonal count is called tropical, solar, or seasonal year.

To achieve this, the earliest human being used the position of the sun in the sky to approximate the time of the day. If the sun was on the horizon, it means that either day was starting or ending. If the sun was high in the sky, it was about midday. However, this system of measuring the time had obvious flaws as listed below;

i. One could not tell the exact time due to indistinct position of the sun in the sky.

ii. It was difficult to see the sun at night.

iii. The sun could be hard to see if there is heavy cloud coverage.

About five thousand years ago, Sumerian in Tigris- Euphrates valley in today's Iraq had a calendar that helped to divide the year into (30) thirty days to four months each. The day was divided into (12) twelve periods. [3].

As earlier mentioned, the concept of year was based on the earth's motion around the sun. The time from one fixed point, such as solstice or equinox, to the next is called a Tropical year. Therefore, one complete solar year contains 365 days, 5 hours, 48 minutes, and 46 seconds.

In the same respect, the concept of a month was based on the moon's motion around the earth. The length of the month is approximately one twelfth of a year. (28 to 31 days) and is justified to fit the 12 months into a solar year [4]. In modern calendars, however, the number of days in a month is not based on the phase of the moon. To avoid any form of difficulties our present civilization has adopted a 365 days solar calendar with a leap year occurring every fourth year.

Along side with the calendar is a clock that keeps track of time in its own form. The clock is also a digital Alarm clock that apart from keeping track of time it also helps to alert an individual of a particular time in the form of hours, minutes, and seconds, with a complete cycle of 24 hours making one (1) day. These seconds, minutes and hour are all meaningful units of calendar. Before any substantial changes are recorded in calendar, the changes of these units of calendar, which result in the calendar changes, must be kept track of. Once it is a new day, then we have a complete event of a new calendar day. The calendar and clock work together and serve the purpose of making us to be aware of time in the form of seconds, minutes, days, months and years as they pass by.

The digital Alarm clocks and calendars also serve as reminders. The clocks that are in use up till today are of different format or classes. By the nature in which clocks are made of, they are classified broadly into analogue and digital clocks. The digital time which is adopted in this project was further classified, by the mode of their counting, into 12-hour clocking count and 24-hour clocking count. The 24-hour counting clock or the 24-hour based timing system is the model used in this project. A complete circle 0f 24 hours counted makes one (1) day. The clock counts from 0 o'clock (the beginning of a new day) through 23 o'clock (which signifies the ending of the day). Once it is new day we have a complete event of a new calendar day.

Most of clocks today make use of batteries and due to the advancement in the electronics technology, integrated circuit ranging from a simple counter IC up to as high as microcontrollers are being embedded in the electronics design of digital alarm clocks and calendar.

A microcontroller with specification AT89C51 was used in this project work. For enhancement and more reliability, a dual power source was used to supply the system energy. Power from 9v rechargeable battery and that of power holding company of Nigeria (P H C N) were used. The device uses microcontroller and quartz crystal for determining the frequency of the microcontroller [5]. The controller was used because of its compatibility with MCS-51™ series products which makes it easy to be handled. The controller also has some useful characteristics as follows;

i. It has a 4KB of reprogrammable flash memory.

ii. It has fully static operation of 0Hz-to-24MHz and

iii. It is mad with an endurance of 1000 write/erase cycle.

The usefulness of the third characteristics is that, if the software program burn onto the controller is not yielding the expected result, the program can be erased off the controller and reprogram it as many times as 1000 times.

The construction of the main project, consist of six (6) different electronic sub circuits as explained in detail in chapter three. The names of these sub circuit are listed below.

i. Alarming circuit-unit.

ii. Power supply circuit unit.

iii. Display unit.

iv. Control circuit unit and

v. The memory unit.

Once the program is written and programmed onto the microcontroller to keep track of time, months, years, leap years and alarm output and also carry out all specified instructions, the device will be able to inform and provide an individual with present date and time. The digital alarm clock and calendar has the features of dates, real time clock chip, months, hours, year, leap year, reminder text and an external memory, and an alarm output. The alarm module consists of a sound device, using speaker or any other sound device.

Multiplexed output signal is being taken through port zero (P0) of the controller into the display unit. The device would be able to serve the purpose of automated calendar/clock since it is digital, software based, and using microcontroller. It properly combines the functionalities of the microcontroller and the efficiency and effectiveness of a liquid crystal display (L C D) to provide a flexible and effective electronic device.

1.1 AIMS AND OBJECTIVES.

1. The design and construction of the microcontroller-based digital alarm clock and calendar with reminder functions is aimed at solving the problems of cost, complexity, portability and accuracy because the earlier clocks/calendars were very large in size, complex in structure and expensive to build.
2. The project is also aimed at providing a longer life span digital calendar/clock that have several years guaranty.
3. The project helps you recall a particular appointment or special functions easily.
4. The project is aimed to realize a digital alarm clock and calendar using microcontroller and acquaints one with the way chips are programmed and handled.

1.2 METHODOLOGY:

The microcontroller-based digital Alarm clock and Calendar with Reminder function system shall use a programmable digital device for system control purposes. An 8bit microcontroller shall be utilized for this purpose.

The controller shall interface with;

1. A 256-byte non-volatile memory for keeping the reminder text date and time settings.
2. A real time clock chip for maintaining the date and time information.
3. A 2 line by 16 characters L C D.

4. A 3*4 ASCII keypad (0-9, A-Z. Other symbols) with (4) four functions keys, (Delete, Escape, Menu and Ok/Enter).
5. An audio sounder for providing available alert when a reminder set date/time matches the system time.

1.3 OVERWIEW:

The system shall provide a means of inputting a maximum of 16 characters into the non-volatile memory via the keypad. A reminder's setting shall comprise the 16 character text, the date and the alarm time. In all, a reminder takes up 22 bytes with a system memory capacity of 256bytes; a maximum of 11 reminders to be stored. A menu-driven approach shall be adopted.

The menu screen offers the user the option of:

i. Viewing any set Reminder
ii. Adding new Reminder if the maximum number presettable is not exceeded.
iii. Setting the system time and date
iv. Deleting one/all the reminders.

For speed and easy manipulations, assembly language was used for the coding, since it allows bit-level operation in a single execution cycle. Alarm modification shall be effected by the system controller comparing the date/time information stored along with each reminder in memory. When the reminder date/time matches the system timer, the Reminder text message is read from memory and displayed on the L C D. The date/time compare operation is extended every 1 minute. The user can there after delete the reminder if so desired.

1.4 SCOPE OF WORK:

This project work is going to be limited and useful in the research and development fields, industrial, laboratory and domestic work at large. The design is targeted at providing a device that could be presented as a final year project. However a well finished design can profitably be marketed to the general public. The study is concentrated on the accuracy of time keeping and date providing a visible character display with the barest minimal reminder function and alarm.

1.5 SOURCES OF MATERIAL:

Datasheet of all the IC's used in the project were of good help because it gave a good insight about the property of the IC's and the requirement for good working condition. Studies of text books/ past project from department and school libraries was very necessary in the design of the power supply and the display unit that displays the operation of the digital Alarm clock/Calendar, magazines and materials from the internet were also employed to make the project a good reference material. My colleagues were also good sources during times of sharing idea.

1.6 PROJECT LAYOUT:

This thesis comprises of five chapters. Chapter one contains the project objectives, methodology, scope of work and project outline. Chapter two dealt with literature review and theoretical background of every important component. Chapter three contains design and implementation of the thesis. Chapter four contains testing and result. Chapter five contains conclusion and recommendation.

CHAPTER TWO

LITERATURE REVIEW/THEORITICAL BACKGROUND

2.1 Historical Background

The quest for the more appropriate means and method of keeping time has evolved greatly from the earliest to the modern clock age. There has been scientific development in the improving of the clock from the primitive Egyptian sundials to the much modern development. Scientists have always worked to make clocks that are more accurate than the last generation of time keeping devices. [7]

Then, in around 3500 B.C, the Egyptians began to construct huge pillars which served as primitive sundials. These pillars told time by casting shadows on the ground which change positions depending on the time of the day. This was a huge improvement over simply looking at the sun, because it was much easier to tell the time by looking at a shadow on the ground than by hurting your eyes looking at the sun.

By 1500 B.C, smaller more improved sundials began to appear, although like previous sundials, they had limitations; they did not work at night or on cloudy days, and were not accurate.

The next great advance in timekeeping occurred in about 3450 B.C when a primitive hour glass was invented. This device was basically a bowl with a hole on the bottom. Water dripped through the hole slowly as the day wore on, and grooves cut into the sides of the hole measured the passage of time. They had many advantages over the sundials because they would work on cloudy days and night. However, these devices required careful calibrations because the water poured out faster when the bowl was full since the water pressure was greater. Also the device would not work in freezing weather.

A huge advancement occurred in the 1300's when mechanical clocks which use weights or springs, began to appear. At first, they had no faces and no hour or minute hands; rather they struck a bell every hour. Later, clocks with hour and then minute hands began to appear. These early mechanical clocks worked by using, a lever that pivoted with and meshed with a toothed wheel at certain intervals. These controlled the movements or "escape "of either the weight or the springs that were powering the clocks, in other to regulate the speed at which the gears and wheels were measured the time turned.

Then, in 1656, Christian Huygens invented the pendulum clocks, which use weight and a swinging pendulum. These clocks were much more accurate than previous. Bigger pendulums made the clocks accurate.

In 1714, the British parliament offered a cash reward to anyone who could invent a clock accurate enough for use in navigation at sea. Thousands of sailors died because they were unable to find their position and pendulum clocks would not work at sea. For every minute lost by a clock meant that there would be a navigational error of 15° and sailors died because they were lost or smashed against rocks as they were unable to figure out their exact position. Then in 1761, after four attempts, John Harrison finally succeeded at inventing a small clock accurate enough to use for navigation at sea. The tiny pocket watch lost only five seconds in six and half weeks.

During the mid late eighties, many countries saw the need to create standard time zones so that everyone could agree on the time and nations could work more efficiently. In 1852, Great Britain implemented a telegraph network that transmitted "Greenwich mean time" (GMT), so the whole country would be running on exactly the same time. In 1884 delegates from many countries met and agreed on worldwide time zones.

At the dawn of the 20th century, women wore wrist watches. However, in the First World War, soldiers wore wrist watches because taking out a pocket watch to check the time was difficult or impossible in battle. After the war was over, it was considered to wear wrist watches and thereafter became popular. The next advancement in time keeping was in 1967, when the atomic clock which used the oscillators of calcium-133 atoms to tell time was invented. This clock had an error of one seconds for every 1.4 million years. Recently in 1999, scientist developed the calcium fountain atomic clock, which is off by only one seconds every twenty million years. The clock is the most accurate in the world.

Throughout the last 5000 years, many advances have been made in the field of time keeping. Time keeping has continuously evolved, and will keep evolving in the future. Perhaps one day humans will invent a clock that accomplishes that long sought after, seemingly unreachable goal; a time keeping device which is one hundred percent [100%] accurate and never gains or loses a second, not even in billions of years. Also as time progresses, everyday clock will gain more features, such as the ability to automatically adjust for daylight saving time, and the ability to synchronize with atomic clocks through radio waves. Watches will gain new features as well, such as integrated radios and displays that shows altitude, temperature and heart rate. Apart from that who knows what the future will bring?

2.1.1 CALENDAR

The common theme for calendar making is the desire to organize units of time to satisfy the needs and preoccupation of society. In addition to solving practical purposes, the process of organization providing a sense of understanding and controlling time itself. [9]

These calendars serve as a link between mankind and the universe. It is little wonder that calendars have provided the basis for planning agricultural, hunting and migration cycles of religious and civil events.

According to a recent estimate by Fraser in 1987 that there are about forty (40) calendars used in the world today [9]. This chapter will only discuss few of them. The emphasis will be placed on their function rather than culture.

2.1.1.0 THE GREGORIAN CALENDAR.

The Gregorian calendar today serves as an international standard for civil use. As Part of its usefulness, it regulates the ceremonial cycle of the Roman Catholic and protestant churches. In fact, its original purpose was ecclesiastical. Although a variety of other calendars are in use today, they are restricted to a particular religion or culture.

He Gregorian calendar resulted from a perceived need to reform the method of calculating dates of Easter. Under the Julian calendar, the dating of Easter has become standardized, using March 21st as the date of the equinox and the me-tonic cycle as the basis for calculating lunar phases [10].

By the 13th century, it was realized that the true equinox had regressed from March 21st (its supposed date at the time of council of Nicea.) to a date earlier in the month [10]. As a result Easter, was drifting away from its springing position and was loosing its relation with the Jewish Passover.

Over the next four centuries, scholars debated the "correct" time for celebrating Easter and means of regulating this time calendrically [10].

By the 16th century the equinox had shifted by ten days, the astronomical New Moons were occurring four days before the ecclesiastical New moons. At the request of Council of Trent, Pope Pius V introduced a Breviary in 1568 and Missal in 1570, both of which included adjustments to the lunar table and the leap year system Pope Gregory xiii, who succeeded Pope Pius in 572, soon convened a commission to consider reform of the calendar, since he considered his predecessor's measures inadequate.

The recommendations of Pope Gregory's calendar commission were instituted by the papal bull "inter gravissimus" signed on 1582 February 24th. Ten days were deleted from the calendar, so that 1582 October 4th was followed by 1582 October 15, thereby causing the Vienna equinox of 1583 and subsequent years to occur about March 21st. and a new table of New Moon was introduced for determining the dates of Easter[10].

The finally reformed modern time adopted for Gregorian calendar in today life to aid in taking note of date and time passage is shown in the table 2.1

TABLE 2.1: DAYS PER EACH MONTH IN GREGORIAN CALENDAR

Months	Number of days
JANUARY	31
FEBUARY	28 or 29
MARCH	31
APRIL	30
MAY	31
JUNE	30
JULY	31
AUGUST	31
SEPTEMBER	30
OCTOBER	31
NOVEMBER	30
DECEMBER	31

2.1.1.ITHE ISLAMIC CALENDAR

The Islamic calendar is a purely lunar calendar in which Months correspond to the lunar phase cycle. As a result, the cycle of the twelve lunar months regresses through the season over a period of about 33 years. For Religious purposes, Muslims begins the Month with the first visibility of the lunar crescent after conjunction. For civil purposes, a tabulated calendar that approximates the lunar phase cycle is often used.

The seven-day week is observed with each day beginning at sunset. Number, with day 1 (one) beginning at sunset on Saturday and ending as sunset on Sunday, specifies weekdays. Day

5, which is called Juma'a is the day of congregational worship. Unlike the Sabbath day of the Christians and Jews, however, Juma'a is not a day of rest. Juma'a begins at sunset on Thursday and ends at sunset on Friday [11].

The form of the Islamic calendar, as a lunar calendar was laid down by the prophet Muhammad in the Qur'an (Sur ix 36-37) and in his farewell pilgrimage. This was a departure from the lunisolar calendar commonly used in the Arab world, in which months were based on the first sighting of the lunar crescent, but an intercalary month was added as deemed necessary. The Islamic calendar starts with the first month called "Muhammad" and ends with the last month named "Zhul Hajj" [11].

2.1.1.2 THE CHINESE CALENDAR

This is one of the special calendars with some special features in the world. The Chinese calendar is a lunisolar calendar based on calculations of the position of the sun and moon. Months of 29 or 30 days begin at days at astronomical New Moons, with an intercalary month being added every two or three years.

Since the calendar is based on the true position of the sun and moon, the accuracy of the calendar depends on the accuracy of the astronomical theories and calculations [12].

Although the Gregorian calendar is used in the people's republic of China for administrative purposes, the traditional Chinese calendar is used for setting traditional festivals and for timing agricultural activities in the countryside.

In China, the calendar was a sacred document, sponsored and propagated by the reigning monarch. For more than two millennia, a bureau of astronomy made astronomical observations, calculated astronomical events such as eclipses, prepared astrological predictions, and maintained the calendar (Needham in 1959). Analysis of surviving astronomical records inscribe on oracle bones reveals a Chinese lunisolar calendar, with intercalation of lunar months, dating back to the Shang dynasty of the

Fourteenth century B.C. various intercalation schemes were developed for the early calendars, including the nineteen-year and 76-year lunar phase cycles that came to be known in the west as the Metonin cycle and callipic cycle.

From the earliest records, the beginning of the year occurred at new moon nears the winter solstice. The choice of month for beginning the civil year varied with time and place. However, in the late second century B.C., a calendar reform established the practice, which continues today, of requiring the winter solstice to occur in month 11. this reform also introduced the intercalation system in which dates of the New Moons are compared with the 24 solar terms. However, calculations were based on the mean motion resulting from the cyclic relationship. Inequalities in the Moon's motions were incorporated as early as the seventh century A.D (Sivin, 1969), but the sun's mean longitude was used for calculating the solar terms until 1644 (Liu and Stephenson, in press)

2.2 THEORETICAL BACKGROUND

2.2.1 Power Supply

In order to avoid frequent power interruption of PHCN, dual power sources was made available for this project work. The project was powered by a 9v rechargeable battery and 220Va.c of PHCN main source.

2.2.1.0 STEP-DOWN TRANSFORMER

The transformer used was a step down transformer to help step down the 220V to 12V. It is shown below.

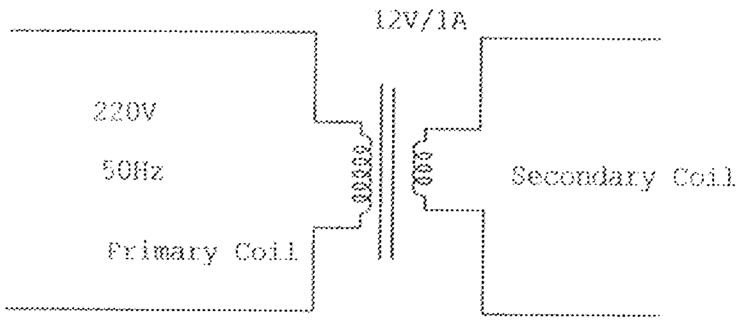


Fig 2.1: Step down Transformer

2.2.1.1 REGULATOR

The regulator was used to regulate the voltages that are to be used for each system. It is shown below

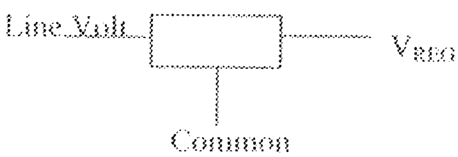


Fig 2.2: Regulator

2.2.1.2 CAPACITORS

The capacitors were used for smoothening, and stability of the voltages.

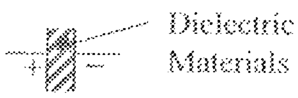


Fig 2.3: Capacitor

2.2.1.3 BRIDGE RECTIFICATION

The full wave rectifier was used to rectify the A.C voltage to a D.C voltage. It is shown below

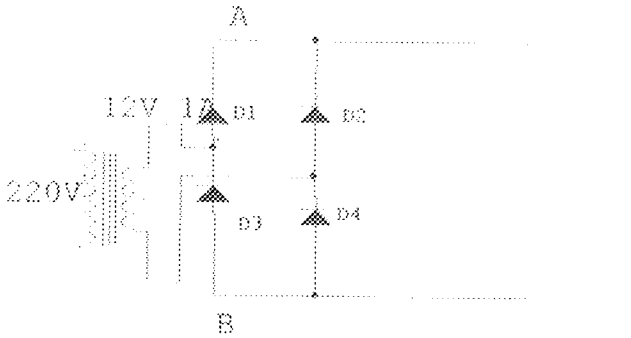


Fig 2.4: Full wave rectifier

2.2.1.4 DIODE

Diode normally conducts fully in one direction. The current flows from anode terminal to cathode terminal. It is shown below

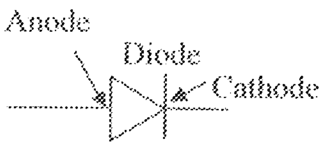


Fig 2.5: Diode

CHAPTER THREE

3.0 DESIGN AND IMPLEMENTATION

The digital reminder unit was built around these under listed sub units;

1. Power supply
2. 8-bit 8957 system controller
3. 256-byte 12C EPROM ($24C0_2$)
4. 2×16 character liquid capital display
5. DS 1307 real time clock chip (RTCC)
6. Audible alert generator
7. 4×4 matrix keypad

Below is the systems block diagram

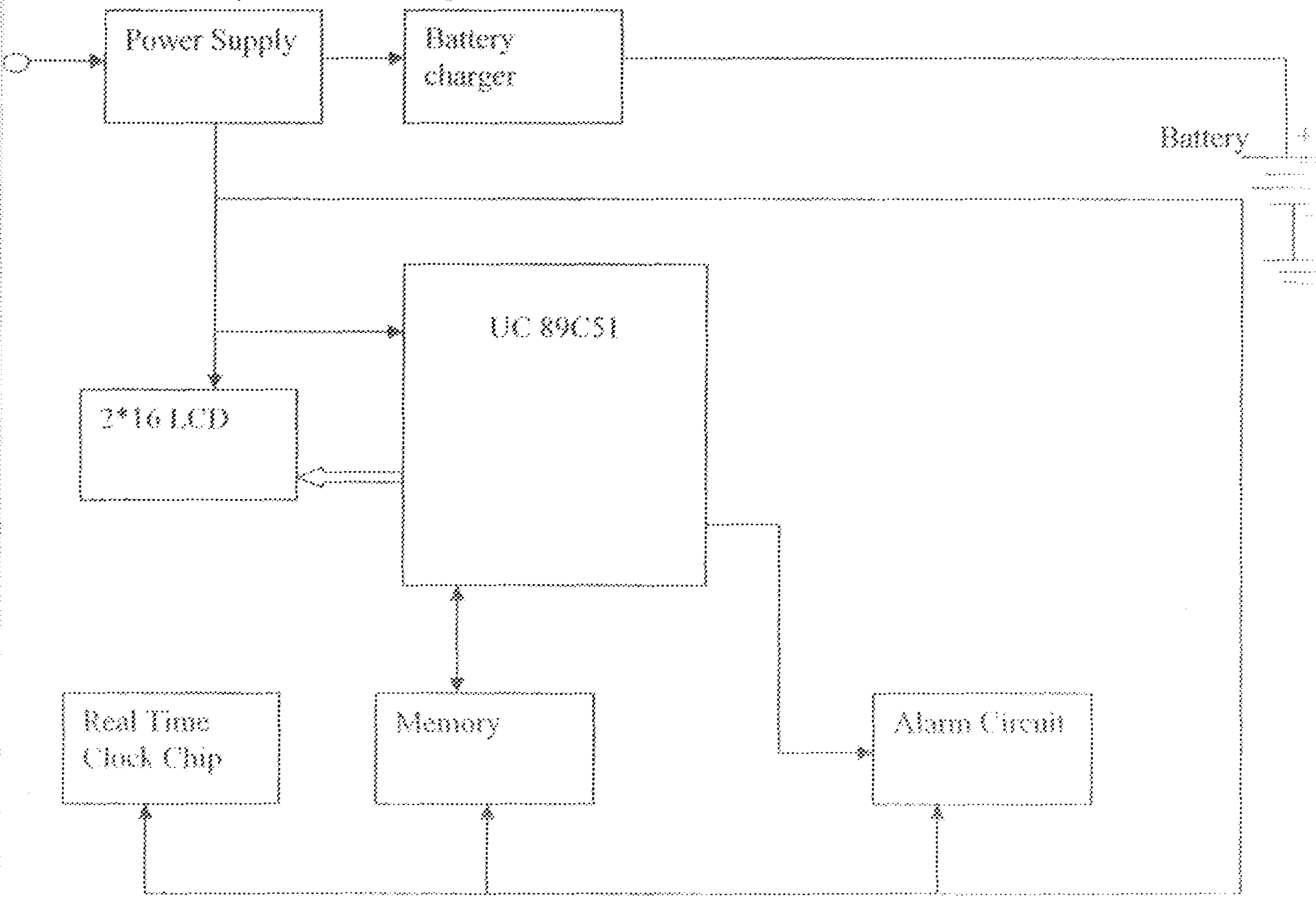


Fig 3.1: THE SYSTEM BLOCK DIAGRAM

3.1 Power supply

A dual source supply was used for system operation.

1. A mains power source and
2. A battery power source

The mains source was obtained using a 240v step down transformer wired to a bridge rectifier as shown below.

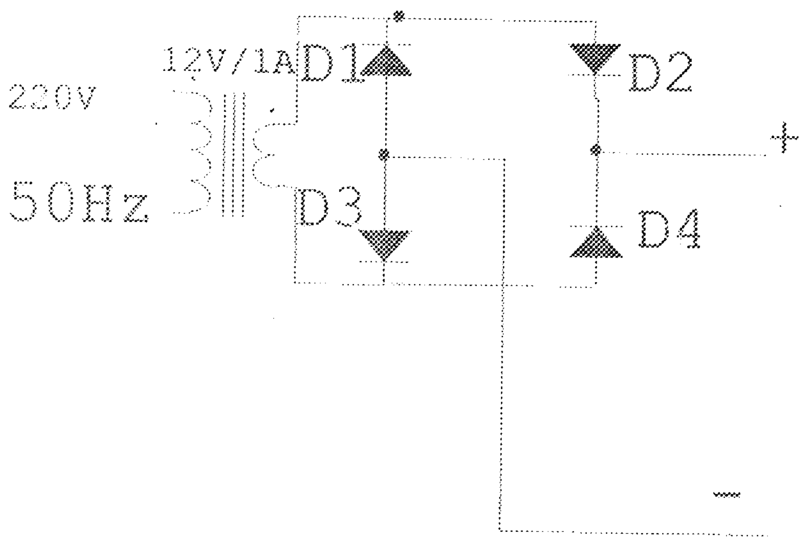


Fig 3.2 SYSTEM POWER SOURCE

The 12V AC voltage was connected to a bridge rectifier to produce DC voltage of amplitude.

$V_{dc} = V_{RMS}\sqrt{2} - 1.4$

V_{RMS} = Transformer secondary output RMS Voltage = 12V

$\sqrt{2}$ RMS-to-peak scaling factor

1.4 = Voltage drop across two alternate diodes with rectifier.

For a 12V RMS input voltage

$V_{dc} = 12\sqrt{2} - 1.4 \cong 15.5V$

Note that the above DC value can only be obtained on a 240V line input voltage. The pulsating DC voltage (at $100H_z$) was smoothened by a capacitance deduced from the expression.

CV = It

C = It/V

Where

C = Smoothening capacitance

I = Maximum load current

$t = \frac{1}{f} = 1/2f$ (FWR) full wave rectifier

V = maximum allowable peak-to-peak ripple voltage on the DC before regulation.

For a 5-Volts system output supply, the minimum input into the 7805 was 7v = (5v+2v drop in the pass transistors). On a 15.5v DC voltage, this translates into a maximum ripple voltage of (15.5 – 1) = 8.5v

The maximum current drain was fixed at a modest 1A, deduced from a conservative summation of the current drawn by the different sub-systems; this value was dominated by the 1A charging current for the incorporated battery.

Calculating

$$C = \frac{It}{V} = \frac{1 \times 1/(2\times 50)}{8.5} = \frac{0.01}{8.5} f$$

This value of capacitance was increased to $2200\mu f$ for improved system performance. The smoothened DC voltage was fed into a current-limited charger and the 7805 regulator.

The 7805 regulator was configured as shown below;

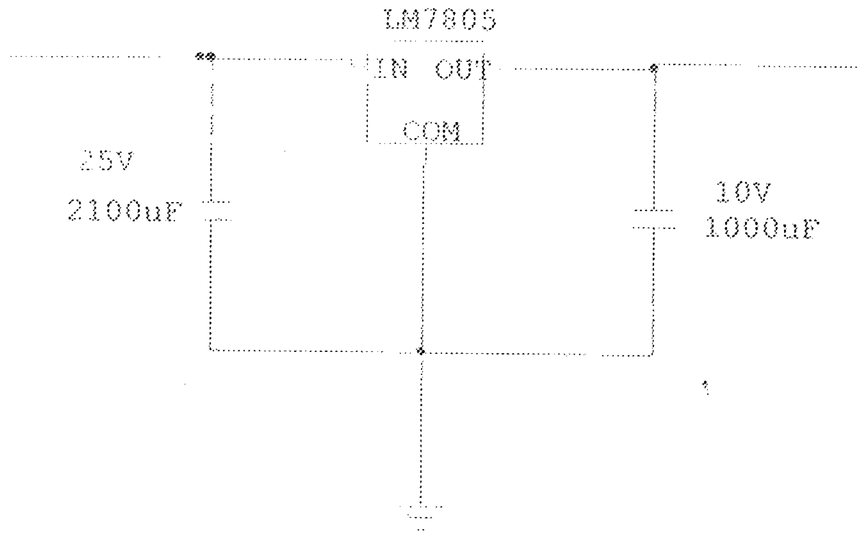


Fig 3.3:5 VOLTS SUPPLY

CURRENT LIMITED CHARGER.

A battery was incorporated into the design. The battery was charged by a constant - current constant – voltage charger system to prevent overcharging. It is shown below,

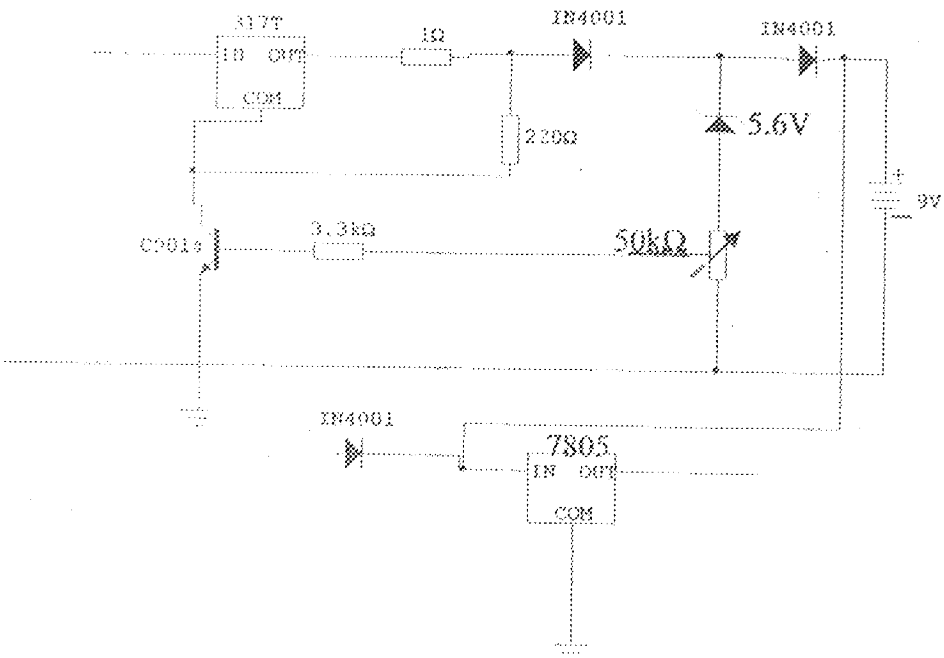


Fig 3.4: CURRENT LIMITED CHARGER.

The charging current was fixed by a resistance connected between pin2 and pin3 of the 317 regulator. The charging current was computed from the expression

$$I = \frac{Vref}{Rs} = \frac{1.25}{Rs} = \frac{1.25}{1} = 1.25A$$

At the commencement of the charging cycle, the charger operates in the constant –current mode. As the battery thermal voltage rises, the voltage of the transistor between 317's pin3 and ground rises. to a value of battery terminal voltage determined by the setting of the 50k

$$V_{batt\,term} = V_z + V_{BE} + V_{f\,diode}$$

$$\doteq 5.6 + 0.7 + 0.7 = 7.0$$

The battery terminal voltage can be altered by adjusting the $50k\Omega$ resistance.

3.2 8951 system controller

For system configuration and control, an 8bit 8951 microcontroller was embedded in the design realization. The device was run of a $12MH_z$ clock, providing an effective system operational speed of $1MH_z$. The controller was programmed to execute the following functions.

1 Accept user input from the 4×4 matrix keypad

2 Process the user input to extract the user request and execute the appropriate operations.

3 Write to the $2line \times 16character\ LCD$

4 Read/write from and to the 256byte memory

5 Read/write from and to the DS1307 clock chip

6 Activate an audible alarm when the system date matches with the alarm time stored against each reminder.

The device was coded with assembly language due to the low level bilateral implementation of the I^2C interface with software. The microcontroller is interfaced with different subsystems as shown below.

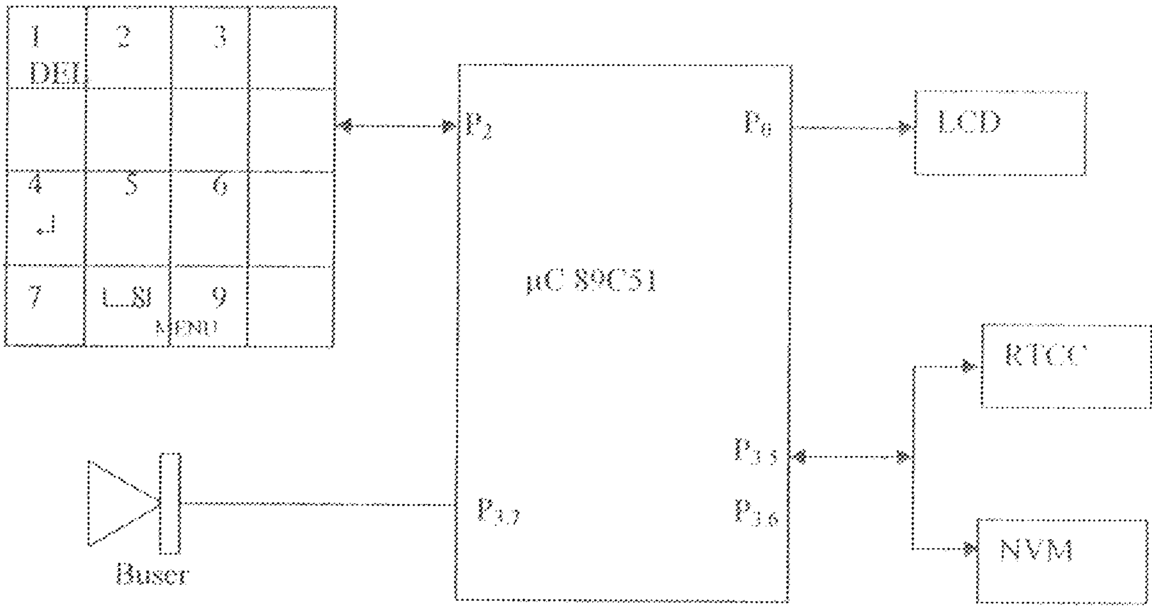


Fig 3.5: Controller Interface

At startup, the LCD display s the system date and time while awaiting the menu button press. When the menu button is pressed, the user is presented with the following message.

VIEW -1 NEW – 2
CLEAR – 3 DATES – 4

When any of the designated buttons is pressed, the software executes the routines associated with such.

3.21 MENU DESCRIPTION

VIEW

This button allows the reminders in memory to be viewed reluctantly.

All the eleven reminders can be viewed by pressing the key after each displayed remainder.

NEW

This option adds a new remainder to memory, assuming the memory is not already full. If the memory is filled up, the message "memory full!" is displayed.

CLEAR

This option clears the remainders in memory one at a time. By selecting the "CLEAR ONE" options, the desired remainder can be deleted by advancing to the designated reminder. Selecting the "CLEAR ALL" option deletes all the reminders in memory.

DATE

The date option enables the system time and date to be set and saved in the real time clock chip. The reminder is stored in the 256byte 24C02 memory device interfaced to port3 over p3.5 and p3.6, since the generic 8051 device does not have a hardware I^2C port. Software emulations were used to implement the proprietary Philips (R) Communications.

The software also compares the different menu time and date with the current system time and date. When there is a match between a reminders' time and date, the appropriate reminder is loaded from memory and displayed on LCD. This reminder is loaded and displayed every one minute until it is cleared by the user via the MENU selection on interface.

KEYPAD MATRIX

A 4 X 4 keypad was interfaced with the controller over p2.0. The keypad was used such that a key press was detected by software without scanning the row/column. This was achieved by the arrangement shown below.

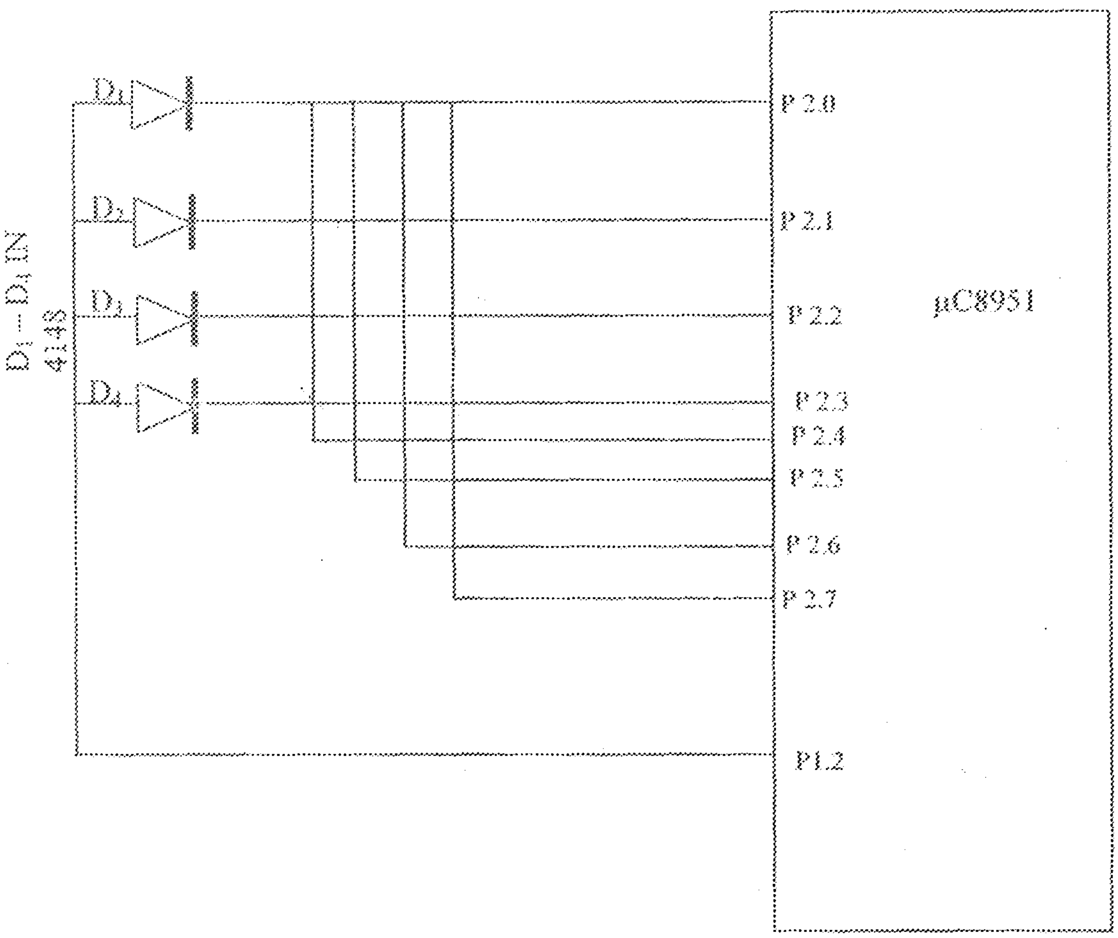


FIG 3.6: KEY PAD CONTROLLER INTERFACE

The keypad port was set to 00001111 at power up, biasing D1-D4, when any key is pressed the diode on the associated row is forward biased, pulling P1.2 to ground. The software detects this high to low transition and does a keypad scan to get the key pressed.

To enable full text inputting via the 4×4 keypad, the buttons were encoded with alternate values. For example, key1 also generated a, b, c; key 2 generated d, e, f, etc. The alternate keys were displayed if the software detects the key has been pressed for longer than $0.5s$.

3.2 24Co2 ELECTRONICALLY ERASABLE PROGRAMMABLE READ ONLY MEMORY

For storing the text information, a 256byte I^2C eprom device ($24C0_2$) was incorporated in the design. The ($24C0_2$) has the electrical specifications stated below.

The device was placed on the I^2C bus formed on P3.5 and P3.6. It was configured for read/ write access at address 10100000B. The memory was positioned into eleven (11) 23-byte pages. The data structure of the stored messages is presented below.

		Address
Text message area	BYTE 0	$x+0$
	BYTE 1	$x+7$
	BYTE 15	$x+15$
	Alarm Day	$x+16$
	Alarm Month	$x+17$
	Alarm year	$x+18$
	Alarm seconds	$x+19$
	Alarm Minute	$x+20$
	Alarm Hour	$x+21$
	Alarm Mask	$x+22$

FIG 3.7: DATA STRUCTURE OF REMINDER

For a 256-byte device, the maximum number of reminders that can be stored is thus;

$$256/23 = 11$$

The $24C0_2$ was interfaced with the controller as shown below

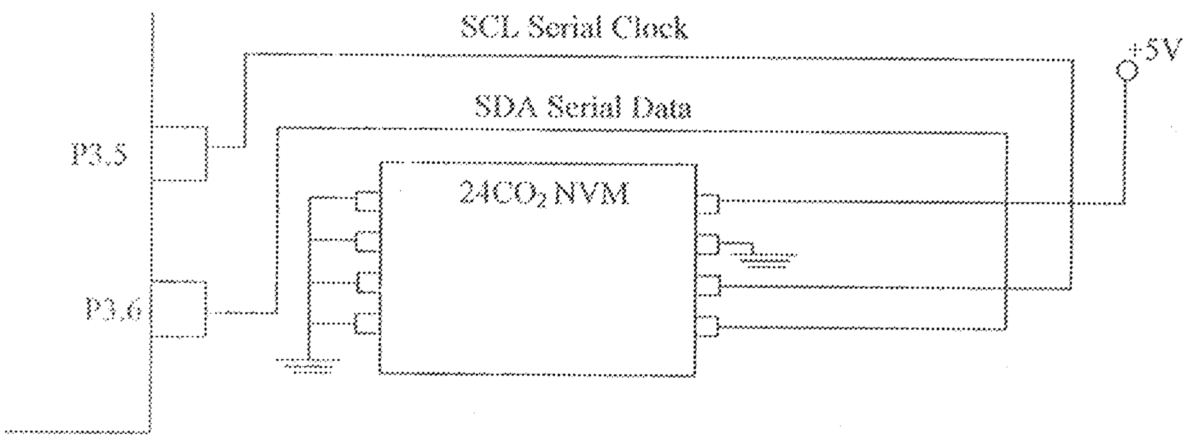


FIG 3.8 $24C0_2$ INTERFACE WITH CONTROLLER

DS1307 REAL TIME CLOCK CHIP

For calendar information, a hardware clock was used. A DS1307 device was selected as it has a I^2C bus in hardware. It was configured for read/write access at address 11010000B.

The device has eight registers with 56-bytes of Ram as shown below

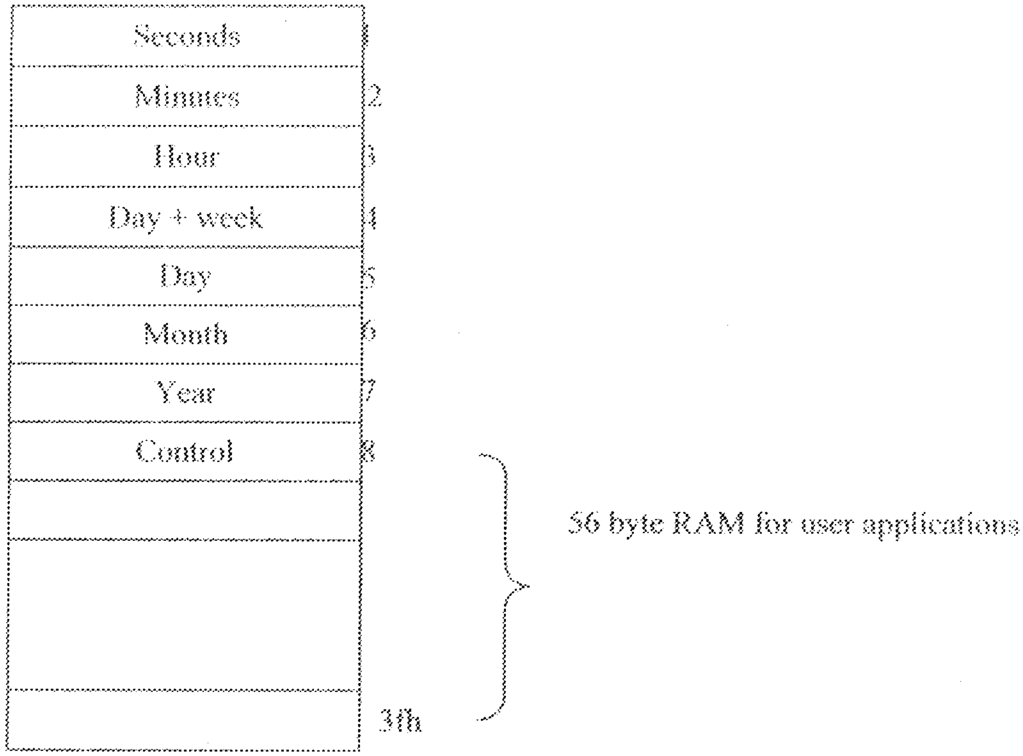


FIG 3.9 DS1307 INTERNAL MEMORY SPACE

Software detection of a newly installed RTCC was effected by reading the string "DS1307" from RAM location 8 through 13. If the power to the real time clock has been disturbed, the signature bytes are lost, otherwise they are retained. A new RTCC is written with the default system given below

Seconds : ooh

Mute : ooh

Hour : ooh

Day-week : ooh

Day : ooh

Month : ooh

Year : ooh

This setting is maintained until its over written by the user via the menu option. The device was programmed to generate a $1Hz$ output pulse on pin7. The $1Hz$ output interrupts the controller every 1s, initiating a read of the calendar registers prior to updating the displayed time/date information on the LCD.

3.3 2×16 CHARACTER LCD

For visual interaction a liquid crystal display was utilized. The display was interfaced with the controller over P0 as shown below

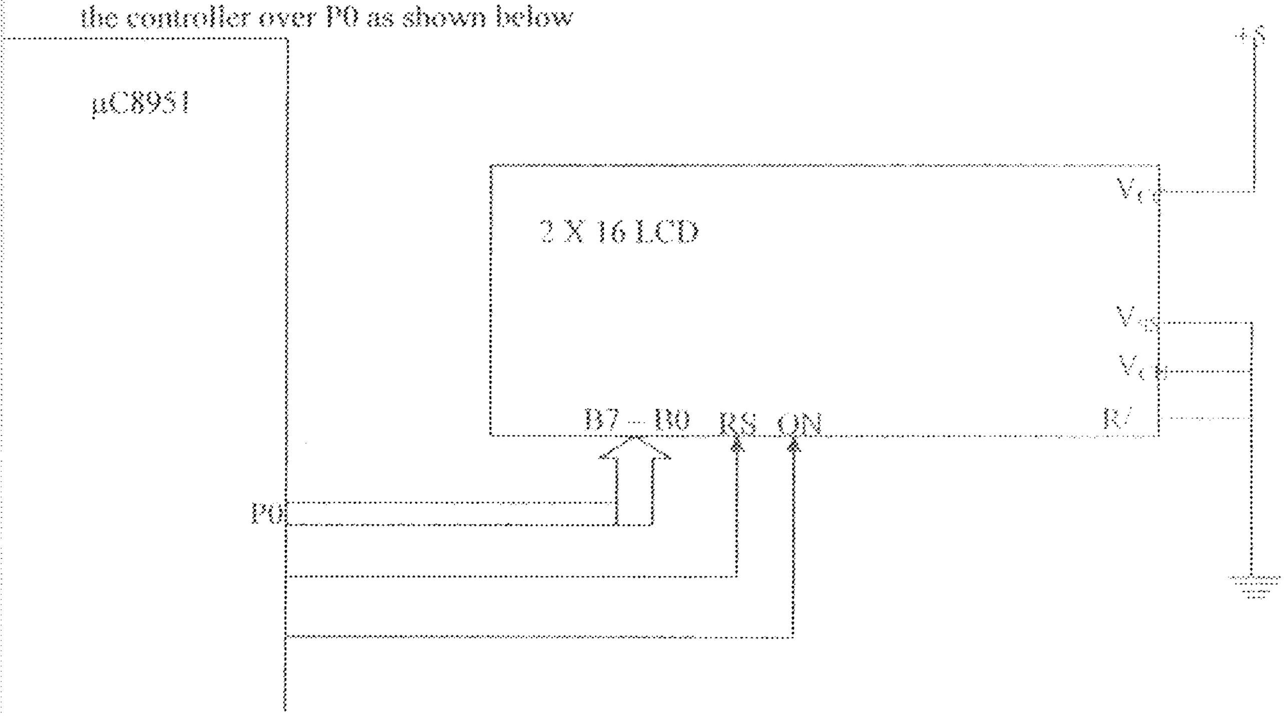


FIG 3.10 CONTROLLER LCD INTERFACE

The LCD was configured in the byte mode with P0 utilized for data command transfer. The register selects (RS) and enabled (EN) control signals were generated by P1.0 and P1.2 respectively. Since the display was only written to and not read from, the read/write input was grounded permanently. The LCD was used to display different messages congruent with the selection currently designated via the menu selection.

3.4 AUDIBLE ALERT GRENERATOR

When the system time matches the calendar information saved against any reminder, the system generates a tune over a sounder wired to the sounder drivers as shown below

3.5 AUDIO SOUNDER

The pnp transistors are forward biased by P3.0 going low. Current flows into the based emitter junction of $Q2$ forward biasing it and generating an audio tune over the sounder.

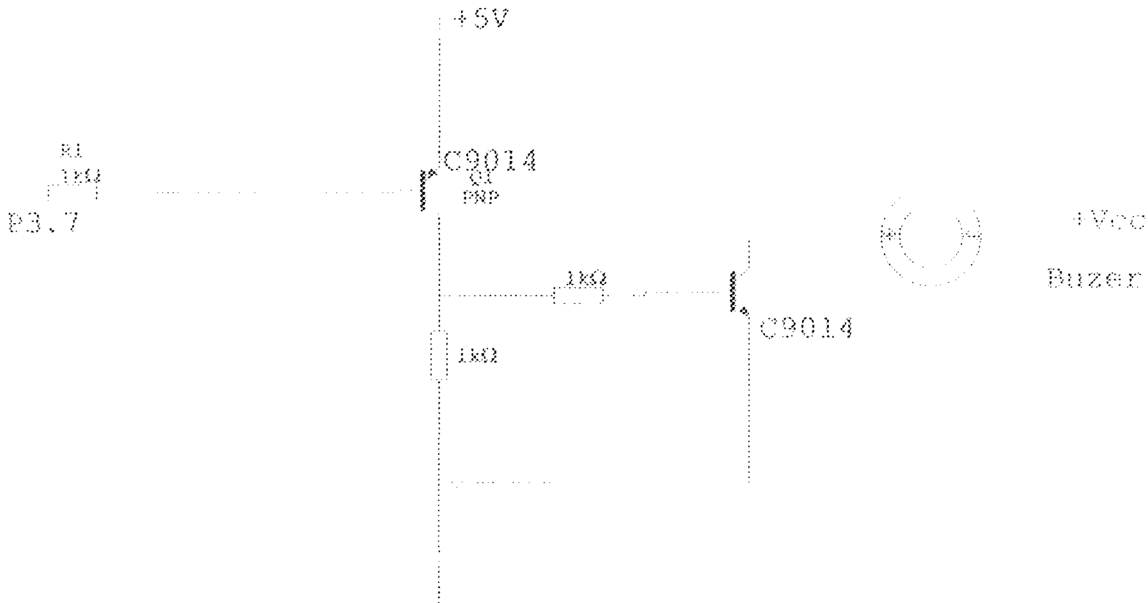
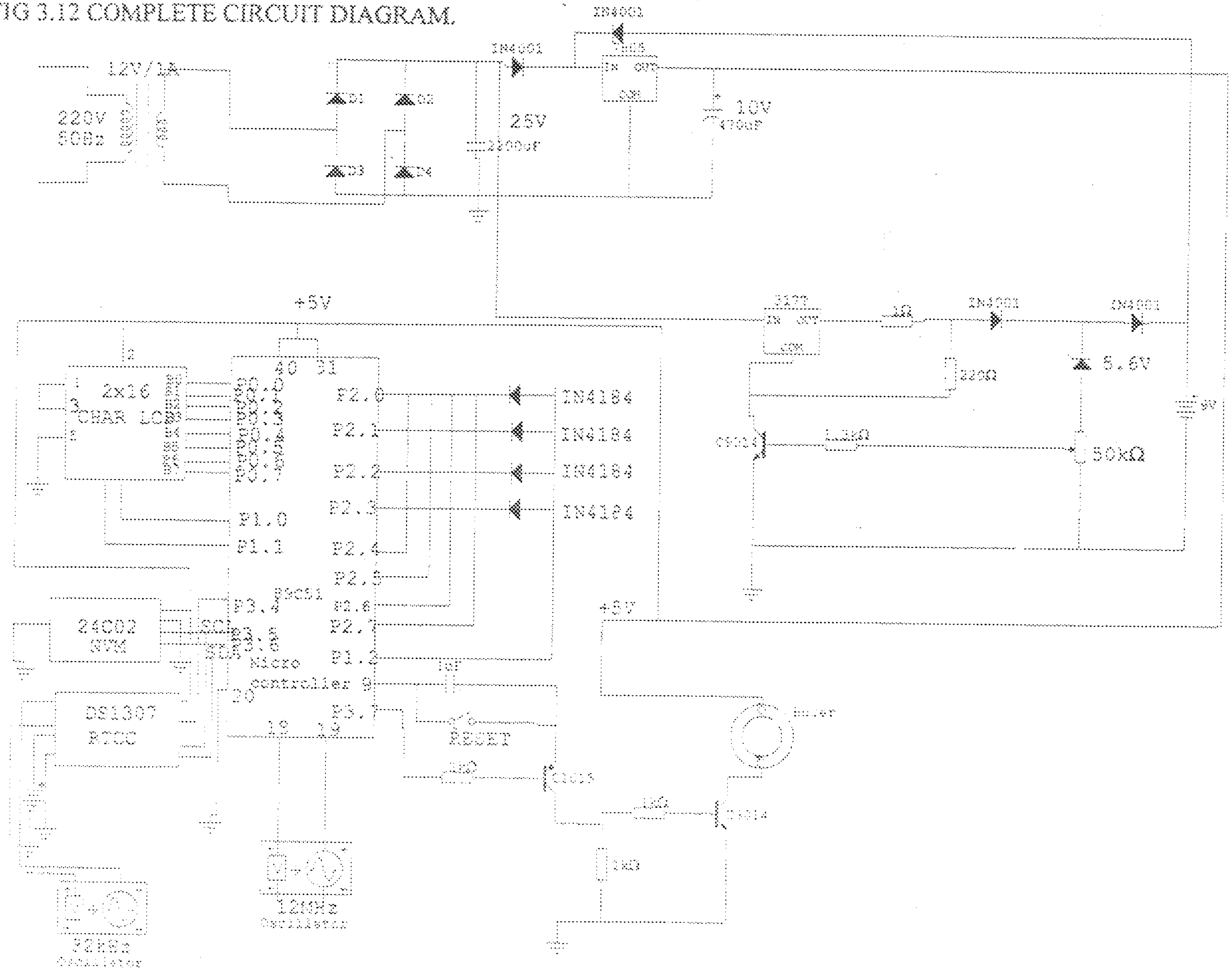


FIG 3.11 Audio Sounder

FIG 3.12 COMPLETE CIRCUIT DIAGRAM.



CHAPTER FOUR

CONSTRUCTION, TESTING AND RESULT

4.1 Construction

This project consists of two main parts namely:

i) Circuit Construction

ii) Input/Output Case Construction

4.1.1 Circuit Construction

This part of the project involved practical exercise on making the circuit diagram on the paper into a real working hardware. The specified components in the design were carefully connected together under the guide of the circuit diagram. The circuit construct was made of ceramics.

Some of the equipment tools used during construction work is listed below.

1) Vero board
2) Copper wire and a pliers
3) Soldering iron, lead and suction
4) Glue and razor blade
5) Plastic and metal sheath
6) Digital multimeter.

During the circuit construction process, each components terminal and functionality are verified.

The power supply unit was quite delicate and was properly checked for short circuit and unwanted bridges.

Fig 4.1 shows the component on Vero board used in the design of this project.

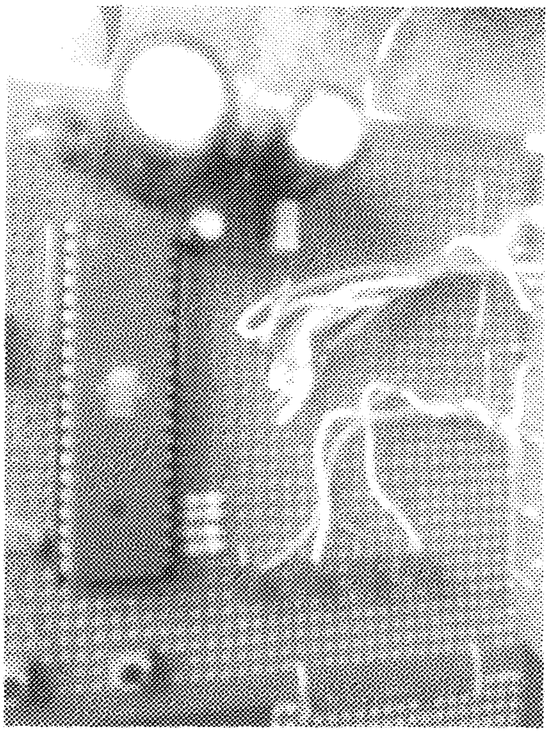


Fig 4.1 Constructed work on a Vero board.

4.1.2 Input/Output case construction.

The input output casing consists of the keypad unit and the LCD which is the output unit.

The casing unit is shown below ;

Fig 4.2 Construction of the Input/Output Case unit.

4.2 Testing

All the components were tested using digital multimeter before being used in this project work. The resistors, diode, capacitor, cables, transistor, switches and LCD were tested for short circuits and open circuits. The software program for the AT89C51 microcontroller was tested by test running it and debugged where necessary, after which it was burnt into the microcontroller. Further testing was carried out on the project by examining the accuracy of the digital clock against a standard clock and by also checking its correctness for keeping of the February of the leap year.

4.2 Result

The result obtained at the end of repeated test was found to be consistent and match up to expected results. Thus, following the construction of this project, the desired aim of the project was achieved.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

The performances of the design and construction of this project met the design specification. It accomplished the aim and objectives of the project, which is to design and construct a prototype that can measure the time in Hour, minutes and seconds and gives the date, day and year on a display in digital form. Reminder function text was also used to make it more users friendly.

5.2 Recommendations

The following recommendations were made based on some areas of the design that could be improved upon, namely:

i. Hourly beep can be introduced to indicate the top of the hour.

ii. The latest sixteen-segment display can be used to increase the number of the alpha-numeric characters on the display and also enlarge the digits display.

iii. A voice alarm can be used to make it more interactive and user friendly.

REFERENCE

1. R. Boylested and L. Nashelsk, [online, Available]: http://physics.nist.gov/GenInt/Time.early.html, 25 June 2008
2. Ancient calendar. [online], Available http://www.calendarzone.com/Janice Mclean 1998-2004. Oct 3rd 2007
3. S.W. Amos, Digital devices [online], Available, http://webxhibits.or/calendars/year.html
4. Danis Russian, [online], Available,http://www.aztecalendar.com Danis Russian (1997-2005) Oct 12th 2007.
5. Tom Duncan: Success in electronics. S. Chand and Company Ltd. New Delhi. (1999)
6. L. Jones, Early physical time. [online] (2003) Available: http://physics.nist.gov/GenInt/Time.early.html
7. Electronic Devices, Microsoft Encarta 2007. 1993-2006 Microsoft Corporation.
8. Edward Hughes, Hughe Electrical and Electronic Technology.
9. http://www.Obliquity.com/calendar/html.David Hapaper and Lynne Marie Stockman 1998, Oct 16th 2007.
10. http://www.Obliquity.com/calendar/georgian/html.David Hapaper and Lynne Marie Stockman.

11. http://www.Obliquity .com/calendar/Islamic/html. David Hapaper and Lynne Marine Stockman 1998, Oct 16th 2007.

12. .Obliquity .com/calendar/Islamic/html. David Hapaper and Lynne Marine Stockman 1998, Oct 16th 2006.

13. The Art of Electronics pages176-145.

PROGRAM CODE

```
;code interfaces a 2*16 lcd with a controleer and 24c02
;256-byte memory to form A digital reminder with 11 alarm settings.
; time and date setting for all reminders, with a 16-character alphanumeric
;text input from an ascii 3*4 kepad. 4 function keys:
;DELETE, ENTER, MENU, ESCAPE.
; USE AT YOUR OWN RISK!!!!!!!!
;***********************************

INCLUDE 89c51.mc
;*********************************
stack EQU 90
;****************************
seconds DATA 8
minutes DATA 9
hours DATA 10
day_week DATA 11
day DATA 12
month DATA 13
year DATA 14
control DATA 15
;*********************************
alarm_flag DATA 16
slave_Address DATA 17
KEY_CODE DATA 18
count4 DATA 19
address DATA 20
data_read DATA 21
```

```
data_2_write DATA 22
count DATA 23
COUNT1 DATA 24
COUNT2 DATA 25
POINTER DATA 26
KEYCODE DATA 27
TEMP1 DATA 28
TEMP2 DATA 29
TEMP3 DATA 30
;****************************************
seconds_address EQU 00h
minutes_Address EQU seconds_Address+1
hour_Address EQU seconds_Address+2
day_week_address EQU seconds_Address+3
day_address EQU seconds_Address+4
month_address EQU seconds_Address+5
year_address EQU seconds_address+6
control_Address EQU seconds_Address+7
;*************************************
sig_byte_Address EQU 08h
sig_byte_lenght EQU 6
sig_byte_offset equ 8
year_offset EQU 18
DAY_OFFSET EQU 16
;*************************************
alarm_lenght EQU 22
alarm1_address EQU 00h
alarm2_Address EQU alarm1_Address+alarm_lenght
```

```
alarm3_address EQU alarm2_address+alarm_lenght
alarm4_address EQU alarm3_address+alarm_lenght
alarm5_address EQU alarm4_address+alarm_lenght
alarm6_address EQU alarm5_address+alarm_lenght
alarm7_address EQU alarm6_address+alarm_lenght
alarm8_address EQU alarm7_address+alarm_lenght
alarm9_address EQU alarm8_address+alarm_lenght
;*********************************************
nvm_Address EQU 10100000b
rtcc_address EQU 11010000b
WRITE_FLAG EQU 00000000B
READ_FLAG EQU 00000001B
;*******************************
KEYPAD_PORT EQU P2
row_1 BIT KEYPAD_PORT.3
row_2 BIT KEYPAD_PORT.2
row_3 BIT KEYPAD_PORT.1
row_4 BIT KEYPAD_PORT.0
col_1 BIT KEYPAD_PORT.4                     ;7
col_2 BIT KEYPAD_PORT.5                     ;6
col_3 BIT KEYPAD_PORT.6                     ;5
col_4 BIT KEYPAD_PORT.7                     ;4
lcd_rs BIT p1.0
lcd_port EQU p0
lcd_en BIT p1.1
Key_in BIT p1.2
SCL BIT P3.5
SDA BIT p3.6
```

```
;***************************************
time_ok BIT 00h
key_set BIT 01h
SAVE_CLOCK BIT 02H
SAVE_OK BIT 03H
compare_ok BIT 04h
no_noise BIT 05h
timeout BIT 06h
;***************************************
row_1 BIT keypad_port.3
row_2 BIT keypad_port.2
row_3 BIT keypad_port.1
row_4 BIT keypad_port.0
col_1 BIT keypad_port.4                 ;7
col_2 BIT keypad_port.5                 ;6
col_3 BIT keypad_port.6                 ;5
col_4 BIT keypad_port.7                 ;4
;********************************
buffer data seconds+50
sounder_dx BIT p3.7
;********************************
DELETE_KEY EQU 0
ENTER_KEY EQU 1
ok_key equ enter_key
MENU_KEY EQU 2
ESC_KEY EQU 3
;********************************
control_mask EQU 00010000b
```

```
MAX_REMINDER EQU 12

;********************************
org 0000h

LJMP start_up
;********************************
org 0003h                              ; enable 1-sec interrupt here from RTCC
JMP rtcc_isr
;********************************
org 000bh
JMP tf0_isr                            ; timeout interrupt for keypad scanning routines
;********************************
org 0013h
JMP alarm_kill_isr                     ; user disable alarm blowfuck!
;********************************
org 001bh
RETI
;********************************
org 0023h
RETI
;********************************
```

```
org 0030h
start_up:                        clr ea
                                        mov sp,#stack
                                        call sys_init

;********************************
mainloop:                        call show_time
                                        call chk_alarm     ; CHK  ALARM  EVERY
1 MINUTE!!!
                                        CALL GET_KEY
                                        JMP mainloop

;********************************
sys_init:                        CALL LONG_DELAY2
                                        MOV tcon, #00000101b
; int0, int1 falling edge.
                                        MOV TMOD,#22H
                                        MOV TH0,#06H
                                        MOV TL0,#06H
                                        MOV count1,#200
                                        MOV count2,#100
                                        MOV count4,#60
                                        CLR compare_ok
                                        CLR no_noise
                                        CLR timeout
                                        SETB key_in
                                        setb SOUNDER_DX
                                        clr lcd_rs
                                        setb lcd_en
                                        CALL INIT_LCD
```

```
                CALL SHOW_ID
                MOV keypad_port,#00001111b
                setb sda
                setb scl
                setb time_ok
                call loading
                call init_timer
                MOV IE,#10000111B
                RET

;********************************
init_timer:     MOV R0,#buffer
                        MOV address,#seconds_Address
init_loop:      call read_rtcc
                        JC error_init_timer
                        MOV @R0, data_Read
                        INC R0
                        INC address
                        cjne r0,#buffer+15, init_loop
                        CALL COMPARE_SIG_BYTE
                        JC GO_INIT
exit_init:       RET

error_init_timer:     MOV DPTR,#timer_error_msg
                        call write_string
                        call long_delay2
                        RET
```

```
timer_error_msg:                DB 01h, 80h, "TIMER INIT ERROR",0

;*********************************************
go_init:                        MOV seconds,#00H
                                MOV minutes,#00H
                                MOV hours,#00H
                                MOV day_Week,#03H
                                MOV day,#01h
                                MOV month,#01H
                                MOV year,#09h
                                MOV control,#control_mask
                        ; reconfigure RTCC to enerate 1hz output!!!
                                MOV address,#seconds_Address
                                MOV R0,#seconds
                                MOV count, #8
                                call write_time
                                call write_sig_byte
                                call zero_alarm
                                RET
;************************************************
zero_alarm:                     MOV DATA_2_WRITE,#0
                                MOV TEMP1,#1
ZERO_LOOP:                      MOV B,TEMP1
                                MOV A,#ALARM_LENGHT
                                MUL AB
                                DEC A
                                MOV ADDRESS,A
                                call write_nvm
```

```
                    JC ERROR_ZERO_ALARM
                    INC TEMP1
                    MOV A, TEMP1
                    CJNE     A,#MAX_REMINDER,
ZERO_LOOP
                    RET
ERROR_ZERO_ALARM:   MOV DPTR,#ZERO_ALARM_ERROR
                    CALL WRITE_STRING
                    CALL LONG_DELAY2
                    RET

ZERO_ALARM_ERROR:   DB 01H, 80H, "ZERO ALARM ERROR",0
;*********************************************
write_time:         MOV data_2_write,@R0
                    call write_rtcc
                    JC ERROR_WRITE_TIME
                    INC address
                    INC R0
                    DJNZ count,write_time
                    RET
ERROR_WRITE_TIME:   MOV DPTR,#TIME_wRITE_ERROR
                    CALL WRITE_sTRING
                    CALL LONG_DELAY2
                    RET

TIME_wRITE_ERROR:   DB 01H, 80H, "WRITE TIME ERROR",0

;*********************************************
```

```
write_sig_byte:                 MOV DPTR,#sig_byte_msg
                                        MOV address,#sig_byte_Address
                                        call write_Sig
                                        RET

;**************************************************
write_Sig:                      CLR A
                                        MOVC A,@a+dptr
                                        JZ exit_write_Sig
                                        MOV data_2_write, A
                                        call write_rtcc
                                        JC ERROR_WRITE_SIG
                                        INC address
                                        INC DPTR
                                        SJMP write_Sig
exit_Write_Sig:                 RET

ERROR_WRITE_SIG:                MOV DPTR,#ERROR_SIG
                                        CALL WRITE_sTRING
                                        CALL LONG_DELAY
                                        RET

ERROR_SIG:                      DB 01H, 80H, "ERROR WRITE SIG!  ",0

;**************************************************
sig_byte_msg:                   DB "DS1307",0
;**************************************************
compare_sig_byte:               MOV R0,#buffer+sig_byte_offset
                                        MOV A,@R0
```

```
        XRL A,#"D"
        JNZ EXIT
        INC R0
        MOV A,@R0
        XRL A,#"S"
        JNZ EXIT
        INC R0
        MOV A,@R0
        XRL A,#"1"
        JNZ EXIT
        INC R0
        MOV A,@R0
        XRL A,#"3"
        JNZ EXIT
        INC R0
        MOV A,@R0
        XRL A,#"0"
        JNZ EXIT
        INC R0
        MOV A,@R0
        XRL A,#"7"
        JNZ EXIT
        CLR C
        RET

EXIT:   SETB C
        RET

;*****************************************
```

```
CHK_ALARM                                   MOV TEMP1,#1
CHK_LOOP:                                   MOV B,TEMP1
                                                CALL
LOAD_ALARM_FLAG
                                                MOV A, ALARM_FLAG
                                                JZ SKIP_CHK_ALARM
                                                MOV A, TEMP1
                                                DEC A
                                                MOV
B,#ALARM_LENGHT
                                                MUL AB
                                                MOV ADDRESS, A
                                                call load_alarm
                                                call load_time
                                                call compare_alarm
                                                call beep_alarm
SKIP_CHK_ALARM:                             INC TEMP1
                                                MOV A, TEMP1
                                                CJNE
A,#MAX_REMINDER, CHK_LOOP
exit_Chk_alarm:                             RET
;**************************************************
load_alarm:                                   mov r0,#buffer
load_alarm_loop:                            call read_nvm
                                                jc error_load_alarm
                                                mov @r0, data_read
                                                inc r0
                                                inc address
```

```
cjne
r0,#buffer+alarm_lenght, load_alarm_loop
RET

error_load_alarm:   MOV DPTR,#ALARM_ERROR
                    CALL WRITE_sTRING
                    CALL LONG_DELAY2
                    RET

ALARM_ERROR:        DB 01H, 80H, "ALARM LOAD FAIL!",0

;*****************************************************
load_time:          mov r0,#seconds
                    mov
address,#seconds_address
load_time_loop:     call read_Rtcc
                    jc error_load_time
                    mov @r0, data_read
                    inc r0
                    inc address
                    cjne    r0,#seconds+8,
load_time_loop
                    RET

error_load_time:    MOV DPTR,#TIME_LOAD_ERROR
                    CALL WRITE_STRING
                    CALL LONG_DELAY2
```

```
                                        RET

TIME_LOAD_ERROR:                        DB 01H, 80H, "TIME LOAD ERROR!",0

;**************************************************

compare_alarm:                          MOV R0,#buffer+year_offset
                                                MOV A, @R0
                                                CJNE    A,      year,
compare_alarm2
                                                DEC r0
                                                MOV A,@r0
                                                CJNE    A,      month,
compare_alarm2
                                                DEC r0
                                                MOV A,@r0
                                                CJNE    A,      day,
compare_alarm2
                                                INC R0
                                                INC R0
                                                INC R0
                                                MOV A,@r0
                                                CJNE    A,      hours,
compare_alarm2
                                                INC R0
                                                MOV A,@r0
                                                CJNE    A,      minutes,
compare_alarm2
                                                SETB c
```

```
                RET

compare_alarm2:         JC alarm_ok
                        CLR C
                        RET

alarm_ok:               SETB C
                        RET

;*****************************************************
;INSERT A KEY READ HERE TO STOP ALARM WAILING WHEN ESC PRESSED!!!
beep_alarm:             JNC EXIT_BEEP_ALARM
                        CLR sounder_dx
                        CALL ENTER_SHOW
we_sound:               call long_Delay2

back_chk_alarm:         CPL sounder_dx
                        JnB key_in,chk1_beep
                        sjmp we_sound

chk1_beep:              call read_keycode
                        CJNE
A,#delete_key,we_sound
                        call delete_one
                        setb sounder_dx
exit_beep_alarm:        RET
```

```
;chk1_beep:                                   call read_keycode
;                                                   CJNE
A,#delete_key,chk2_beep
;;                                                  call delete_one
;                                                   setb sounder_dx
;                                                   ret
;
;chk2_beep:                                   cjne a,#ok_key,chk3_beep
;                                                   call stop_timer
;                                                   SETB sounder_dx
;                                                   ret
;
;chk3_beep:                                   cjne a,#esc_key,back_chk_alarm
;                                                   call stop_timer
;                                                   SETB sounder_dx
;                                                   ret
;
;exit_beep_alarm:                        RET

;***********************************************************
load_alarm_Flag:                          MOV A, #alarm_lenght
                                                    MUL ab
                                                    DEC A
                                                    MOV address,A
                                                    call read_nvm
```

```
                                MOV         alarm_flag,
data_read
                                ret
;*************************************************
show_time:          JNB time_ok, exit_show_time
                        CLR TIME_OK
                        mov r0,#seconds
                        MOV address,#seconds_address
time_loop:          call read_rtcc
                        jc error_show_time
                        mov @r0, data_read
                        inc r0
                        inc Address
                        cjne r0,#buffer+8, time_loop
;*************************************************
                        call line1
                        MOV A,#" "
                        call write_lcd_Data
                        call write_lcD_Data
                        call write_lcd_Data
                        call write_lcd_Data
                        MOV R0,#day
                        call convert_time
                        mov a,#"/"
                        call write_lcd_Data
                        mov r0,#month
                        call CONVERT_TIME
                        mov a,#"/"
```

```
                call write_lcd_Data
                mov r0,#year
                call convert_time
                call line2
                MOV A, #" "
                call write_lcd_Data
                call write_lcD_Data
                call write_lcd_Data
                call write_lcd_Data
                mov r0,#hours
                call convert_time
                mov a,#":"
                call write_lcd_Data
                mov r0,#minutes
                call convert_time
                mov a,#":"
                call write_lcd_Data
                mov r0,#seconds
                call convert_time
                MOV A, #0d0h
                call write_lcd_cmd
exit_Show_time:             RET

error_show_time:            MOV DPTR,#ricc_fail
                    call write_string
                    call long_delay2
                    RET
```

```
rtcc_Fail:                              DB 01h, 80h, "  RTCC FAILURE!  ",0

;****************************************
convert_time:Call hex_2_Ascii
                                        call write_lcd_Data
                                        mov a, b
                                        call write_lcd_data
                                        ret
;****************************************
hex_2_ascii:mov A,@R0
                                        anl a,#11110000b
                                        swap a
                                        call convert_2_ascii
                                        PUSH acc
                                        MOV A,@R0
                                        anl a,#00001111b
                                        call convert_2_Ascii
                                        mov b,a
                                        pop acc
                                        RET
;****************************************
SHOW_ALARM:                             MOV R0,#buffer
show_alarm_loop:                    call read_nvm
                                              JC error_show
                                              MOV @R0, data_read
                                              INC R0
                                              INC address
```

```
                                        CJNE     R0,#buffer+alarm_lenght,
show_alarm_loop
ENTER_SHOW:                             CALL LINE1
                                              MOV R0,#buffer
show1:      *                           MOV A,@R0
                                              call write_lcd_Data
                                              inc r0
                                              cjne r0,#buffer+16, show1
                                              call line2

show2:                                  CALL CONVERT_TIME
                                              MOV A,#"/"
                                              CALL WRITE_LCD_dATA
                                              INC R0
                                              CALL CONVERT_TIME
                                              MOV A,#"/"
                                              CALL WRITE_LCD_DATA
                                              INC R0
                                              CALL CONVERT_TIME
                                              MOV A,#" "
                                              CALL WRITE_LCD_DATA
                                              call write_lcd_data
                                              call write_lcd_data
                                              INC R0
                                              CALL CONVERT_TIME
                                              MOV A,#":"
                                              CALL WRITE_LCD_dATA
                                              INC R0
```

```
                        CALL CONVERT_TIME
                        CALL SHOW_REMINDER_ID
                        RET

error_Show:             MOV DPTR,#ALARM_SHOW_ERROR
                        CALL WRITE_sTRING
                        CALL LONG_DELAY2
                        RET

ALARM_sHOW_ERROR:       DB 01H, 80H, "ALARM SHOW FAIL!",0
;*****************************************
LINE1:                  MOV A,#01H
                        CALL WRITE_LCD_CMD
                        MOV A,#80H
                        CALL WRITE_LCD_CMD
                        RET
;*****************************************
LINE2:                  MOV A,#0C0H
                        CALL WRITE_LCD_cmd
                        RET
;*****************************************
long_delay2:            CALL LONG_DELAY
                                CALL LONG_DELAY
                                RET
;*****************************************
LOADING:                        MOV DPTR,#LOADING_MSG
                                CALL WRITE_sTRING
                                CALL LONG_DELAY2
```

```
        RET
LOADING_MSG:    DB 01H, 80H," LOADING...     ",0

;*****************************************************
settle_delay:   MOV R7,#0
        DJNZ R7,$
        MOV R7,#0
        DJNZ R7,$
        RET
;*****************************************************
START_TIMER:    MOV COUNT1,#200
        MOV COUNT2,#100
        MOV TL0,#06H
        CLR TF0
        CLR timeout
        SETB TR0
        RET
;*****************************************************
stop_timer:     CLR tr0
        CLR TF0
        RET
;*****************************************************
KEY_DELAY:      MOV R2,#30
KEY2:           CALL dly_2ms
        CALL dly_2ms
        CALL dly_2ms
        CALL dly_2ms
        DJNZ R2, KEY2
```

```
                RET
;*********************************************
CLEAR_SCREEN:           MOV A,#01H
                CALL
WRITE_LCD_CMD
                MOV A,#80H
                CALL
WRITE_LCD_CMD
                RET
;*************************************************
get_key:                JB key_in, exit_getkey
            call READ_keyCODE
            CALL DECODE_KEYCODE
            JNB key_in,$
            CALL LONG_DELAY
            SETB time_ok
exit_getkey:        RET
;***********************************************
DECODE_KEYCODE:     CJNE A,#2, exit_decode
            call show_menu
exit_decode:        RET
;***********************************************
show_menu:              MOV DPTR,#menu_msg
            call write_string
            CALL WRITE_sTRING
            CALL GET_MENU
            RET
```