


“I / We admit that I/we have read this literature work through my/our observation which has fulfilled the scope and quality in order to be qualified for the conferment of Bachelor Degree in Electronic Engineering (Electronic Industry).”

Signature : 
Supervisor's name : SANI / ROHAN SAJIM
Date : 1 / 4 / 05

REAL –TIME CLOCK DISPLAY PART II


TOH YEN YIN

**This Report Is Submitted In Partial Fulfillment Of Requirements For The
Bachelor Degree of Electronic Engineering (Industrial Electronic)**

**Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer
Kolej Universiti Teknikal Kebangsaan Malaysia**

March 2005

“I admit that this is done by my self except the discussion and extracts taken from other sources that I explained each in detail.”

Signature : 
Author's Name : TOH YEN YIN
Date : 18th March 2005

This literature piece is dedicated to my beloved father and mother.

APPRECIATION

In this section, I would like to take an opportunity to thank my faculty dean, Professor Hamid Hamidon and my supervisor, En. Sani Irwan Bin MD Salim, who assist and guide me a lot in executing my project. I would also like to thank all my family members and course mates for giving moral supports.

ABSTRACT

This is a final year project report that describes about the project of Real-Time Clock Display (PART-2). This project is to implements a program for a digital clock model with LED dot matrix display which is using a PIC16F877 microcontroller and a DS1307 Real Time chip (RTC) as the main part of the circuitry. PART-2 involves more on PIC programming side as well as the configuration of microcontroller. The advantage of using real time chip instead of normal programming to create a digital clock is the RTC use crystal frequency count to set the time counter that will avoid count time problem. Furthermore, RTC also provides seconds, minutes, hours, day, date, month, and year information that initialized by writing the appropriate register bytes. The project program flow start from reading input from setting buttons, follow by data processing for i²c driving of RTC chip from a PIC microcontroller and character generation with an i²c EEPROM lastly display the output via LED dot matrix. C Language as chosen programming language to write the program.

ABSTRAK

Ini merupakan sebuah laporan bagi Projek Sarjana Muda yang menerangkan tentang projek bertajuk Paparan Jam Masa Nyata (Bahagian-2). Projek ini adalah untuk melaksanakan aturcara bagi sebuah jam digit yang memaparkan keluaran di LED dot matrix dengan menggunakan PIC 16F877 pengawal micro dan DS1307 chip masa nyata (RTC) dalam litar utamanya. Bahagian-2 lebih bertumpu kepada bahagian aturcara PIC dan juga perancangan microcontroller. Kelebihan menggunakan RTC untuk menggantikan pengaturcaraan biasa bagi mencipta sebuah jam digit adalah RTC menggunakan pengiraan frekuensi kristal untuk menentukan pengiraan masa dimana ia dapat mengabaikan masalah pengiraan masa. Selain itu, RTC juga menyediakan maklumat tentang saat, minit, jam, hari, tarikh dan tahun yang dapat dikenalpasti dengan menulis kepada *register bytes* yang sesuai. Aliran aturcara untuk projek ini bermula dengan membaca masukan dari putang penyelarasan, diikuti dengan pemprosesan data bagi i²c yang dituju ke RTC chip dari PIC pengawal micro dan penjanaan perkataan dengan i²c EEPROM. Akhir sekali keluaran dipaparkan melalui LED dot matrix. Bahasa C merupakan bahasa pengaturcaraan untuk menulis aturcara projek ini.

TABLE OF CONTENTS

TITLE	PAGES
PROJECT TITLE	i
ADMIT	ii
DEDICATION	iii
APPRECIATION	iv
ABSTRACT	v
ABSTRAK	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	ix
LIST OF FIGURES	x
1 INTRODUCTION	1
1.1 Introduction of Project	1
1.2 Objectives of Project	2
2 BACKGROUND RESEARCH	4
2.1 Review of Related Literature	4
2.1.1 Real-time clock	4
2.1.2 Digital Clock	5
2.2 Problem Statement based on Digital Clock Literature Review	6
2.2.1 Clock was slow	6
2.2.2 The time setting of units of hour in PM1x was abnormal	6
2.2.3 Time setting mode error	7
2.2.4 It is delayed about 1 second in a day.	8
2.3 Solution	8
2.3.1 PIC Microcontrollers	8
2.3.2 Real-Time Clock (RTC)	12
2.3.3 MPLAB IDE	15
2.3.4 LED Dot Matrix	28
3 PROJECT METHODOLOGY	38
3.1 Procedures	38
3.2 The Development Environment	41
3.2.1 The Text Editor	41
3.2.2 The Translator	41
3.3 The Development Cycle	42

3.4	Design Process	43
3.5	Project System Block Diagram	44
4	PROJECT FINDING	46
4.1	Output Display	46
4.2	Circuit Design	47
4.2.1	Display Clock In 5x7 LED Dot Matrix	47
4.2.2	RTC, EEPROM, Crystal, and Switch Circuit	48
4.2.3	Counter Circuit	49
4.3	Program Development	52
4.3.1	Main Program Flow Chart	54
4.3.2	Display Subroutine Flow Chart	54
4.3.3	DS1307 Read routine.	55
4.3.4	i ² c EEPROM to dot matrix display	58
4.4	Simulation	61
4.4.1	Software Simulation - Proteus 6.3	61
4.4.2	On Board Simulation	62
5	DISCUSSION AND CONCLUSION	63
5.1	Discussion	63
5.1.1	Future Works	63
5.2	Conclusion	64
	REFERENCE	65
APPENDIX A	LIBRARY FUNCTIONS REFERENCE	
APPENDIX B	DS1307 DATA SHEET	
APPENDIX C	PIC 16F877 BLOCK DIAGRAM	
APPENDIX D	PIC16F877 PINOUT DESCRIPTION	
APPENDIX E	PIC16F877 REGISTER FILE MAP	
APPENDIX F	PIC 16F877 INSTRUCTION SET SUMMARY	
APPENDIX G	5 x 7 DOT MATRIX DISPLAYS DATA SHEET	
APPENDIX H	4017 DATA SHEET	
APPENDIX I	24LC256 EEPROM DATA SHEET	
APPENDIX J	PROJECT PROGRAM	
APPENDIX K	PSM PLANNING GANTT CHART	

LIST OF TABLES

TITLE	PAGES
Table 1: Display Pattern for the Character A*	32
Table 2: The table consists of 128 characters (128x5 bytes x 8 bits=5120 bits total)	33

LIST OF FIGURES

TITLE	PAGES
Figure 1: Real-time clock	5
Figure 2: Digital Clock	5
Figure 3: Inside microcontroller	10
Figure 4: PIC16F877 Pin Diagram	11
Figure 5: DS1307 Block Diagram	13
Figure 6: DS1307 Address Map	14
Figure 7: DS1307 Timekeeper Registers	15
Figure 8: Components of MPLAB IDE	16
Figure 9: Project Manager	18
Figure 10: LED Matrix with Common Anode Arrangement	30
Figure 11: Illustration of the Characters A and B	31
Figure 12: Character Scanning	36
Figure 13: Scanning Duty-Cycle	37
Figure 14: Project Methods Flow Chart	39
Figure 15: The Development Cycle	43
Figure 16: Project System Block Diagram	44
Figure 17: Real-Time Clock Display via LED Dot Matrix Circuit Diagram	47
Figure 18: RTC, EEPROM, Crystal And Switch Circuit Diagram	49
Figure 19: Counter circuit	51
Figure 20: Timing Diagram Of Counter	51
Figure 21: Program development flow	52
Figure 22: Program writes in MPLAB IDE workspace	53
Figure 23: Program simulation output	53
Figure 24: Main Program Flow Chart	54
Figure 25: Display Subroutine	55
Figure 27: i ² c EEPROM to dot matrix display	59
Figure 28: Simulation using Proteus 6.3	61
Figure 29: On board simulation testing hardware	62

CHAPTER 1

1 INTRODUCTION

1.1 Introduction of Project

The Real-Time Clock Display project is searching around the development of a digital clock that displays the real-time through LED dot matrix display which includes the seconds, minutes, hours, day, date, month, and year, by using a DS1307 Real-Time Chip and a PIC16F877 microcontroller as main part of the circuitry.

The Real-Time Clock Display project is divided into two parts which is for PART-1 consists of designing the clock circuit, including setting up the connection of dot matrix scanning. PART-2 involves more on PIC programming side as well as the configuration of microcontroller.

The project of Real-Time Clock Display PART-2 is involve on PIC programming which is create program for digital clock model using PIC microcontroller and Real-Time chip that display output via LED dot matrix.

The advantage of using real time chip instead of normal programming to create a digital clock is the RTC use crystal frequency count to set the time counter that will avoid count time problem. Beside that, RTC also provides seconds, minutes, hours, day, date, month, and year information that initialized by writing the appropriate register bytes. With the advantage of RTC in count time and information provided, the programming step for 24hrs/12hrs format with AM/PM indicator and create the program for calendar and time can be skip by using RTC.

The program design of the digital clock is implementing using MPLAB-IDE as the project development tool and C Language as programming language. Program compile by using HI-TECH PICC compiler to convert PIC instruction from program language into machine language. Software and hardware devices are chose depending on target processor (PIC 16F877 microcontroller). Debuggers and emulators are needed to use for help debug and find errors with code. Programmers as devices that use to program the processors with source code that burned in Flash ROM inside PIC.

LED dot matrix programming will route the output display. Programming scanning through row driver and column driver to transfers the output to exactly coordinate.

1.2 Objectives of Project

1. To carry out a project for Real Time Clock Display.
2. To accomplish program writing for Real Time Clock Display.
3. To perceive about PIC microcontroller and RTC chip.
4. To experience implement an embedded system design with MPLAB IDE.

5. To practice and become familiar with the components in development tools of MPLAB IDE.
6. To learn the C Language for PIC Programming (PIC instruction code) to apply in the project.
7. To study Data EEPROM and Flash Program Memory configuration.

CHAPTER 2

2 BACKGROUND RESEARCH

2.1 Review of Related Literature

2.1.1 Real-time clock

This real time clock is build using PIC16F84 (16C84) CPU for control and display via seven segment and Dallas DS12887A RTC for clock. In fact this clock is testing prototype for more complicated clock design. This device is build for testing the programs on PIC microcontroller. There are two buttons, one to increase and another to decrease RTC memory address.

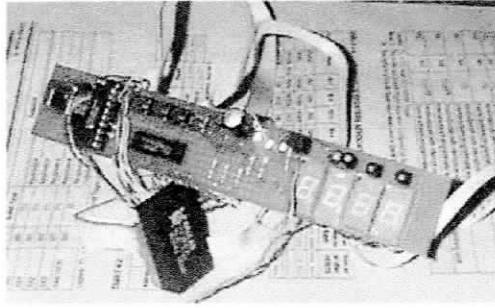


Figure 1: Real-time clock

2.1.2 Digital Clock

This digital clock is built using PIC16F873 CPU for control and display via seven segments. 10MHz clock which has ultra precision is used for the input clock. This clock is converted into 50Hz which is $1/200000$ by CPLD (XC9536). It is to make time setting correct that it didn't change into 1Hz. When making 1Hz, the setting of a time becomes a second unit. So, it isn't possible to do setting in the second correctly. In case of 50Hz, because it is adjusted in the 20-millisecond precision, there is no problem in case of practical use. A 50Hz clock is connected with the RB0 port of PIC. The RB0 port can make have the function to do interruption in the change of the input signal. This time, it detects the rising edge of the input signal and the interruption occurs. It counts this interruption 50 times and recognizes 1 second. This frequency converter can be made with the circuit using three 74HC390 to be introducing "Frequency divider".

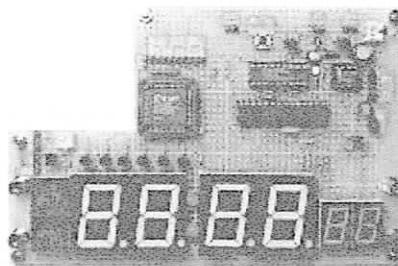


Figure 2: Digital Clock

4-MHz resonator is used for the operation clock oscillation by PIC. The precision of this oscillation frequency doesn't influence the precision of the clock. The precision of the clock is decided by the precision of the frequency. Because it doesn't need the high-speed operation of PIC at the circuit this time, 4MHz is used.

2.2 Problem Statement based on Digital Clock Literature Review

2.2.1 Clock was slow

When measuring count time of 1 second, it had been delayed compared with the correct clock. There was a mistake in the processing which makes time of 1 second as a result of the investigation. 1 second is made by counting 20 milliseconds 50 times. At first, if being equal to or less than 50, it was doing the processing which updates a counter. In this case, it counts in the extra of 1 count because it is increasing a count after checking. 1 correct second could be made by correcting checking data to 49.

2.2.2 The time setting of units of hour in PM1x was abnormal

First, it set units of hour to "9" in the condition of PM0x. After, it set tens of hour to PM1x. In this condition, when increasing the count of units of hour, an abnormally display was done to units of hour. In case of PM1x, only three kinds (0,1

and 3) are made to be able to be set to units of hour. In the previous processing, it judged in the condition which is equal to 2 as the upper limit judgment. Therefore, when units of hour were 9, it was increasing a count simply because it was not 2. As a result, it referred to the data which exceeded 7 segment tables and an abnormally display was done. It changed the processing into "equal to or more than 2" judgment and it became able to be normally set. The count-down processing with PM1x was similar too. It wasn't possible to do normal time setting because the unit of hour was only the judgment of "0". The normal setting became possible by putting "equal to or more than 3" judgment. This clock can set to AM1x or PM1x after setting the units of hour to 9(The figure which is not 0, 1, 2). So, the setting of AM19 or PM17 has become possible. It isn't checking about this.

2.2.3 Time setting mode error

When changed with the time setting mode, tens of hours are set as the setting position. However, it had become the position of units of hour.

When the time setting mode ends, the error occurs to the setting in 0 seconds. Even if a setting switch in 0 seconds was pushed according to 0 seconds when the time setting ends, the error occurred to the setting in the second.(Becoming 1 second immediately so on) The cause was to be because when becoming a clock mode from the time setting mode the counter which counts 1 second wasn't cleared. It solved in clearing a counter in 1 second when becoming a time setting mode. The checking of setting in 0 seconds is done in the 20-millisecond interval. So, the maximum of 20-millisecond error occurs.

2.2.4 It is delayed about 1 second in a day.

The clock was delayed about 1 second in a day. However, it was doing normal operation in 10000kHz when measuring by the frequency counter. The cause was to do the mistake being of the logic of CPLD which makes 50Hz from 10MHz. The 1/200000 counter is composed of 1/100000 counter and 1/2 counter. It is judged 1/100000 count at 100000. The right value is 99999. Therefore, it became the counter of 1/200002. In case of the wrong logic, to be delayed in 1 second is caused by 100000 seconds. Because a day is 86400 seconds, it is delayed 0.86 seconds in a day. The output of the counter of CPLD becomes 49.9995 Hz in the calculation. According to the detailed measurement, there is possibility to have found a mistake.

2.3 Solution

2.3.1 PIC Microcontrollers

There are several families of microcontrollers currently available from different manufacturers and each family has varied ranges of ICs which have difference function. PIC series, manufactured by Arizona Microchip as the chooses microcontroller for this project which are now the most popular microcontrollers because they are cheap in price, small size and offers many several functions such as general input/output ports with individual direction control, clock/counter, serial ports and A to D Converters and many more. The PIC devices have assemblers and simulators available for the PC. The Microchip PIC microcontrollers are a convenient and cost-effective solution for a lot of home-made applications, because:

- PIC microcontrollers are widely available (Farnell, Conrad, Disterelec)
- PIC controllers are cheap and do not require a lot of prerequisites before running in an application (i.e. additional HW blocks such as external EPROM for program memory, external A/D converter)
- the PIC development environment is freely available and up-to-date
- there is a broad range of different PIC microcontrollers, very likely to fit into your target application
- the hardware programming itself is easy and does not require a lot of additional equipment (like JTAG programmer, EPROM/flash programmer)
- the various packages available (DIP, SSOP, SOIC, TQFP, PLCC), even with low pin counts, allow for convenient prototyping without big initial efforts

The PICmicro MCU has program memory for the firmware, or coded instructions, to run its program. It also has "file register" memory for storage of variables that its program will need for computation or temporary storage. It has a number of peripheral device circuits on the same chip. Some peripheral devices are called I/O ports. These are pins on the microcontroller that can be driven high or low to send signals, blink lights, drive speakers - just about anything that can be sent through a wire. Often these pins are bidirectional and can also be configured as inputs allowing the program to respond to an external switch, sensor, or to communicate with some external device.

In order to design such a system, it need to decide which peripherals are needed for the application. Analog to Digital converters allow to connect the microcontroller to sensors and receive changing voltage levels. Serial communication peripherals, allow to stream communications over a few wires to another microcontroller, to a local network, or to the internet. Peripherals on the PICmicro MCU called "timers" accurately measure signal events and generate and capture communications signals, produce precise waveforms, even automatically reset the microcontroller if it gets "hung" or lost due to a power glitch or hardware malfunction. Other peripherals detect if the external power is dipping below

dangerous levels so the micro can store critical information and safely shut down before the power is completely lost.

The peripherals and the amount of memory the application needs to run the program largely determine which PICmicro MCU to use. Other factors might include the power consumed by the microcontroller and its "form factor," i.e., the size and characteristics of the physical package that must reside on your target design.

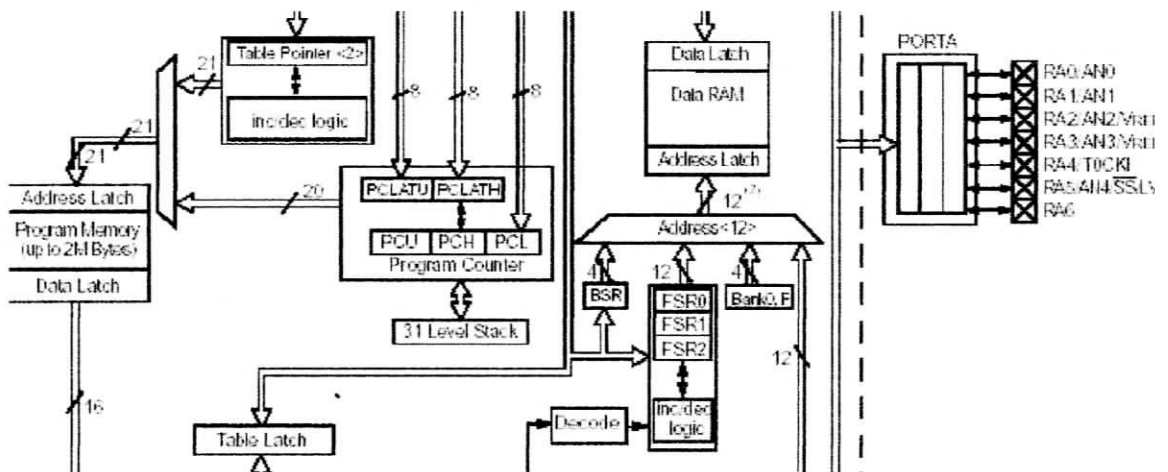


Figure 3: Inside microcontroller

2.3.1.1 PIC16F877 Feature

- easy to program, even with self-made PIC programmers (search the web)
- sufficient performance for ordinary controller-applications, up to 5 MIPS @ 20 MHz
- 8-level deep hardware stack (for calling sub-routines)
- has internal and external interrupt support (e.g. timers, port change), total of 4 interrupt sources
- only 35 RISC instructions

- provides many additional hardware resources, e.g.
 - 10-bit A/D converter (8 channels)
 - synchronous serial port (SPI) with SPI master mode and I²C master mode
 - ability to store non-volatile data in internal EEPROM (256 bytes non-volatile memory)
 - analog comparators
 - universal synchronous asynchronous receiver transmitter (USART) for RS232 or similar interfaces
 - parallel slave port
 - capture/compare/PWM blocks
- total of 15 interrupt sources (internal and external interrupts)
- self-reprogrammable under software control (boot-loader)
- 40 pin package (DIP40)
- offers additional peripheral interfaces, especially to connect I²C peripheral components

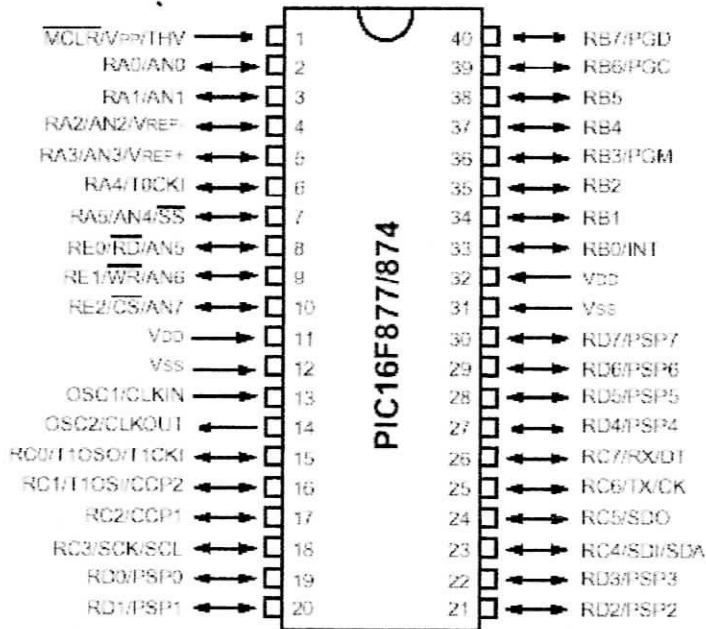


Figure 4: PIC16F877 Pin Diagram

2.3.2 Real-Time Clock (RTC)

Real-Time Clock is the chip that provides the clock/calendar includes seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus. The DS1307 has a built-in power sense circuit that detects power failures and automatically switches to the battery supply.

The advantages of using RTC instead of normal programming to create a digital clock is the programming step for 24hrs/12hrs format with AM/PM indicator and create the program for calendar and time can be skip by using RTC.

2.3.2.1 DS1307 RTC Features

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- RTC use crystal frequency count to set the time counter that will avoid count time problem
- 56-byte, battery-backed, nonvolatile (NV) RAM for data storage
- Two-wire serial interface
- Programmable square wave output signal
- Automatic power-fail detect and switch circuitry

- Consumes less than 500nA in battery backup mode with oscillator running
Optional industrial temperature range:
-40°C to +85°C
- Underwriters Laboratory (UL) recognized

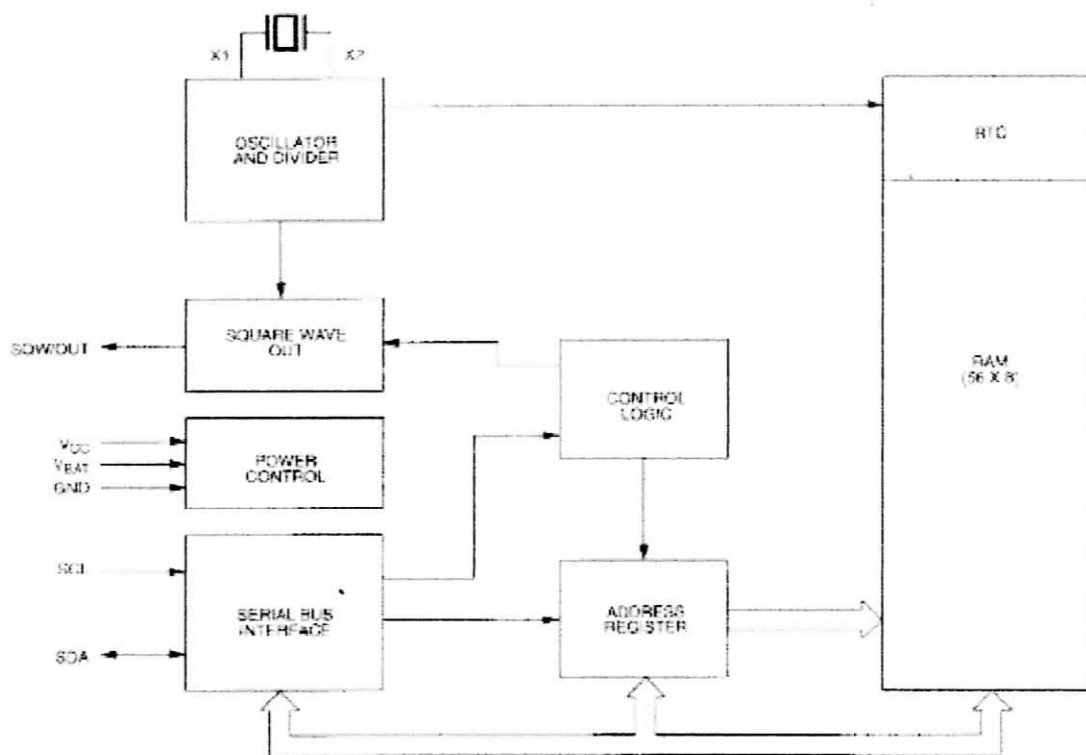


Figure 5: DS1307 Block Diagram

2.3.2.2 RTC and RAM Address Map

The address map for the RTC and RAM registers of the DS1307 is shown in Figure 8. The RTC registers are located in address locations 00h to 07h. The RAM registers are located in address locations 08h to 3Fh. During a multi-byte access,