# TABLE OF CONTENTS

# ABSTRACT

Real Time Clock (RTC) ICs (DS3231) measure time and temperature , even when the power of the Main device is off. During these times , RTC ICs draw power from an auxiliary battery or supercapacitor. As expected , power consumption is a key factor in most RTC designs, but accuracy and small package size are also important . We need to generalize our knowledge with the best available things and redesign the entire thing into a small compact cluster which can fulfil the basic needs of livelihood with the use of Arduino MEGA 2560 which interfaced with the small electronic integrated unit with temperature sensor and real time clock will be fixed on any system that need real time analysis i.e.DS3231 . The data and commands for the device will be preinstalled during its manufacture. This will be automatically update as the time passes by and will show the temperature at that time. Generally implemented in house hold appliance, industrial appliance and many more .This whole setup works with the I2C protocol. The purpose of the present research is to construct a kind of real time based clock with temperature. It is intended to use as a desktop appliance. The system design consists of Arduino UNO, DS 3231 real time clock module, input switches, and a 16 characters 4 lines LCD display. The goal of the circuit is to display a current, date and time, temperature with alarm using buzzer.



**Keyword:**

Real Time Clock , DS3231 , I2C ,Arduino , Buzzer ,MEGA 2560

# LIST OF FIGURES

# CHAPTER-I

# INTRODUCTION

A real time clock (RTC) is a battery powered clock that measures time even when there is no external power or the microcontroller is reprogrammed by using Arduino. Arduino based real time clock is a digital clock to display real time using a RTC IC DS 3231 which works on I2C protocol. In this circuit we have used a 16×4 LCD module to display the time in (hours, minutes, seconds, dates, months and years) format. Real time clock is commonly used in our computers, houses, offices and electronics device for keeping them update with real time. It is not only a real time clock but also a temperature and humidity sensing circuit. The temperature and humidity are major concern as global warming. In this circuit, the weather data measuring system is designed to investigate temperature and humidity intensities. The measurement values are also displayed on LCD. The basic structure of the system is illustrated with a block diagram in figure 1.

## 1.1 BLOCK DIAGRAM

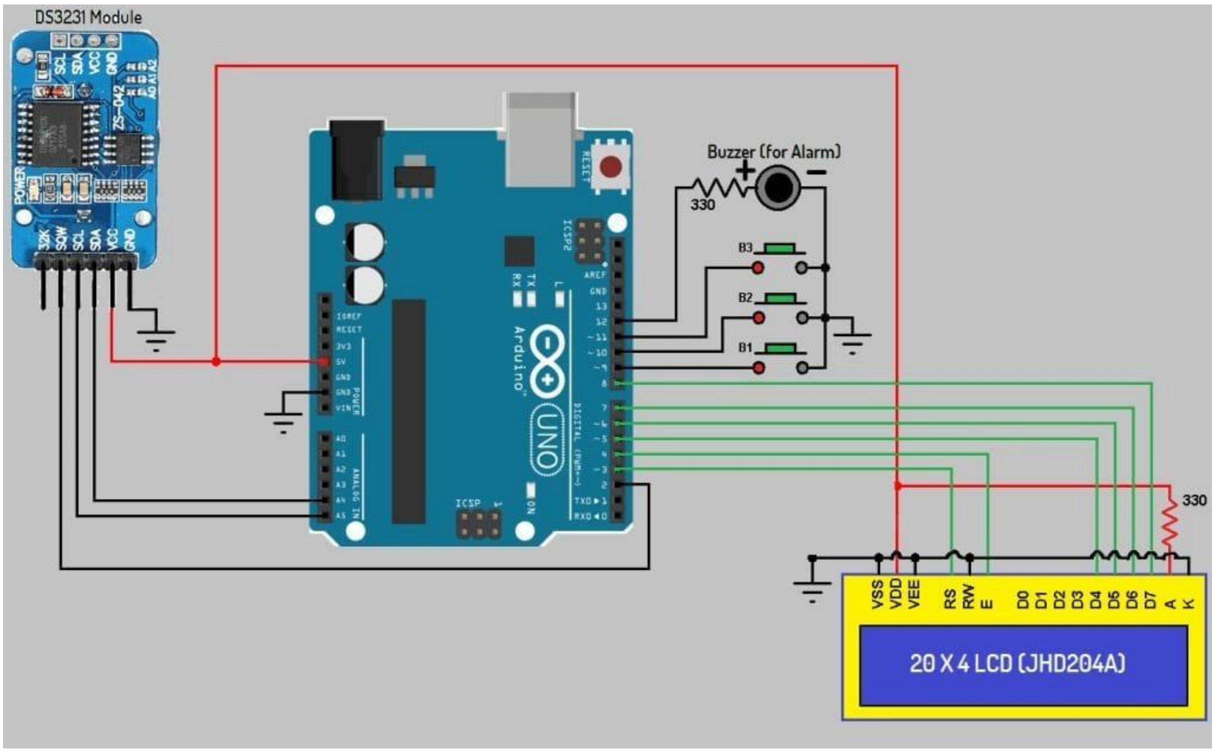


Fig 1.1 Block Diagram

### 1.1 Real Time Clock (RTC):

A real time clock is basically just like a watch - it runs on a battery and keeps time for you even when there is a power outage! Using an RTC, you can keep track of long timelines, even if you reprogram your microcontroller or disconnect it from USB or a power plug. A real-time clock (RTC) is a computer clock, most often in the form of an integrated circuit, that keeps track of the current time. Although the term often refers to the devices in personal computers, servers and embedded systems, RTCs are present in almost any electronic device which needs to keep accurate time. Most microcontrollers, including the Arduino, have a built-in timekeeper called millis() and there are also timers built into the chip that can keep track of longer time periods like minutes or days. So why would you want to have a separate RTC chip? Well, the biggest reason is that millis() only keeps track of time *since the Arduino was last powered* . That means that when the power is turned on, the millisecond timer is set back to 0. The Arduino doesn't know that it's 'Tuesday' or 'March 8th', all it can tell is 'It's been 14,000 milliseconds since I was last turned on'. OK so what if you wanted to set the time on the Arduino? You'd have to program in the date and time and you could have it count from that point on. But if it lost power, you'd have to reset the time. Much like very cheap alarm clocks: every time they lose power they blink 12:00 While this sort of basic timekeeping is OK for some projects, some projects such as data-loggers, clocks, etc will need to have consistent timekeeping that doesn't reset when the Arduino battery dies or is reprogrammed. Thus, we include a separate RTC! The RTC chip is a specialized chip that just keeps track of time. It can count leap-years and knows how many days are in a month, but it doesn't take care of Daylight Savings Time (because it changes from place to place).


Fig:1.2 DS 3231 RTC Module

Although keeping time can be done without an RTC, using one has benefits:

- Low power consumption (important when running from alternate power)
- Frees the main system for time-critical tasks
- Sometimes more accurate than other methods

RTCs often have an alternate source of power, so they can continue to keep time while the primary source of power is off or unavailable. This alternate source of power is normally a lithium battery in older systems, but some newer systems use a super-capacitor, because they are rechargeable and can be soldered. The alternate power source can also supply power to battery backed RAM. Most RTCs use a crystal oscillator, but some use the power line frequency. In many cases, the oscillator's frequency is 32.768 kHz. This is the same frequency used in quartz clocks and watches, and for the same reasons, namely that the frequency is exactly 215 cycles per second, which is a convenient rate to use with simple binary counter circuits. The DS3231 module is made up of the DS3231 chip which is a cheap and accurate RTC chip capable of storing time and data information for years after being set as long as the chip keeps getting power from and attached coin cell battery. The module even automatically makes variations the leap years in its memory once it is set. It works with either 5V or 3.3V voltage level and can be used. The RTC can program square wave output signal. It is automatic power-fail detect and switch circuitry. Real time clock can count seconds, minutes, hours, date of the month, month day of the week and year with leap-year.

## 1.2 ARDUINO UNO

Arduino is an open-source hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL),[1] permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors. Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (for prototyping) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs. The microcontrollers can be programmed using the C and C++ programming languages, using a standard API which is also known as the Arduino language, inspired by the Processing language and used with a modified version of the Processing IDE. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool developed in Go.

Fig 1.3 Arduino UNO

## 1.3 LIQUID CRYSTAL DISPLAY MODULE

Liquid crystal display (LCD) screen is an electronic display module and a wide range of applications. A 16×2 LCD display is a very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. LCD is easily programmable, has no limitation of displaying special and even custom characters (unlike in seven segments), animations and so on. A 16×2 LCD means that it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5×7 pixel matrix. This LCD has two registers, namely, command and data. Pins diagram of 16×2 LCD are shown in figure 4.

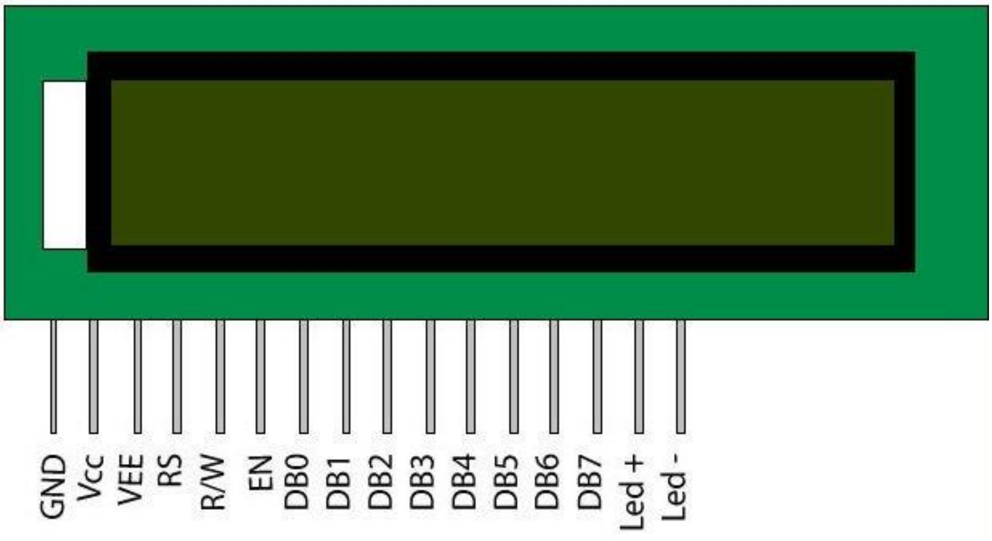


Fig 1.4 LCD 16 x4

## 1.4 PUSH BUTTON

The DS3231 module is supplied with 5V as the 20*4 LCD (JHD204). There are 3 data lined connected between the DS3231 Module & Arduino board. SCL line is connected to analog pin 5, SDA is connected to analog pin 4 and the INT line is connected to digital pin 2 which is the external interrupt pin of the Arduino. The DS3231 interrupts the microcontroller when there is an alarm (alarm1 or alarm2).

In the circuit, there are 3 push buttons: B1, B2, and B3. These buttons are used to set time, calendar & alarms.

**For setting Time & Date:**

Button B1 = Button B1 selects time or date parameter like an hour, minute, day, date, month.

Button B2 = B2 increments the selected parameter.

**For setting Alarm:**

Button B3 = Button B3 selects the parameter

Button B2 = Button B2 increments the selected parameter.

Also, there is a buzzer connected to Arduino pin 12, this buzzer

Fig 1.5 Push Button

## 1.5 CONCLUSION

Real time programming using I2C protocol with RTC was successfully implemented, and it helps to maintain real time clock necessary for various real time running systems. This circuit describes hardware design and implementation of low-cost smart clock based on Arduino. Arduino is used for reading from DS1307 and DHT11 then display it on 16 × 2 LCD. DS1307 sends time/date using 2 lines to Arduino. The advantages of these RTC circuit are high accuracy at a relatively low cost price and the external supply of batteries to store the current time even after disconnecting or main power failure. For connecting module to Arduino is also used I2C bus. For sensing the temperature in the room was used DHT11 sensor which has a range 3.5V to 5.5V of input voltage and it uses 5V TTL over only one wire. The sensor is accurate and the accuracy of measuring the humidity is 2% - 5%. The sensor has a maximum power consumption of 1.5 mA during measurement and it is able to measure and differentiate the value every two seconds. The photograph of real time with temperature and humidity sensing circuit operation are shown in figure 9. Therefore, it is an essential household appliance. It can be developed to control other real time based operating systems such as regular warming the rice cooker, filling the water tank, checking the mail box and so on.

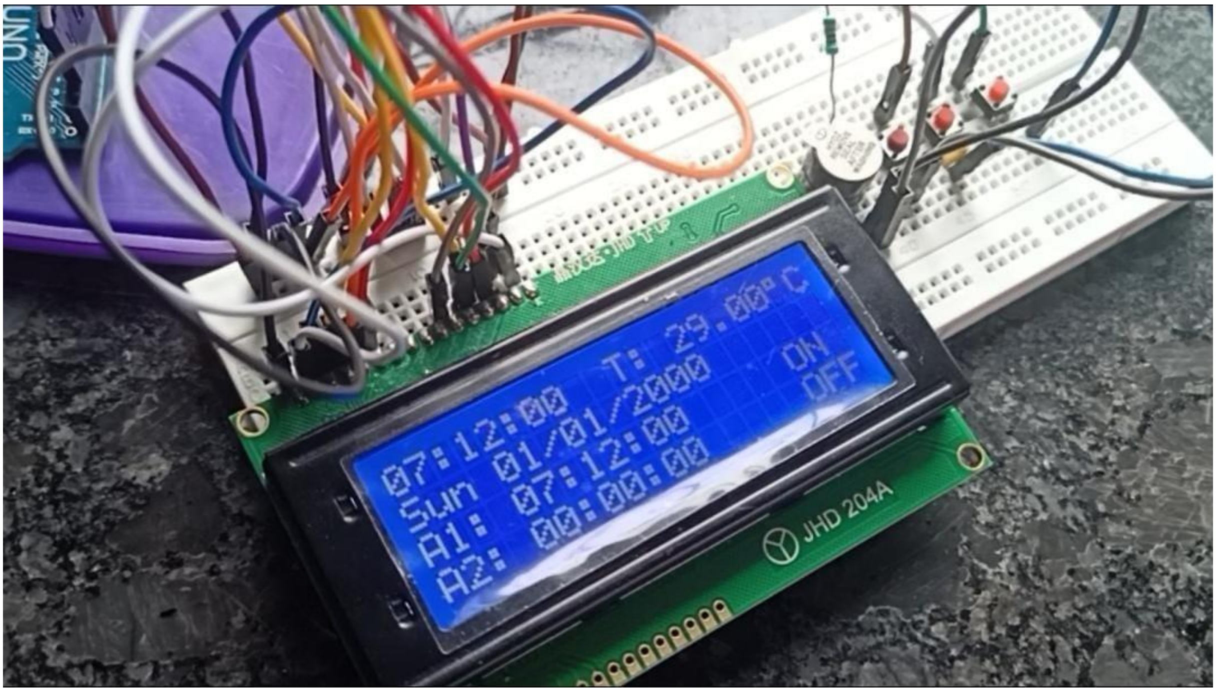


Fig 1.6 Implementation of the Project

# CHAPTER-II

## 2.1 LITERATURE AND REVIEW

Every 1Hz counter, it will represent 1 second of a real time. This increment will continue up until years. The seconds counter will count to 60 second then the minutes will increment by one. After the second reach exactly 60 second, the second counter will reset to 0. Then minutes counter will continue increase for 60 minutes. Then the hours will increment. After 24 hours, the date counter will increase. This increment flow will continue exactly as a real time until it reaches 1 years. After that it will continue for the next year. The date will be depending on month and year as the month has different amount of days in every month and for leap years that will decide by the amount of days in February. This are the basic counter increment flow used to design this research RTC.

## 2.2 PURPOSE RTC BLOCK DIAGRAM

The block diagram consists of bus interface, frequency divider, function controller, RTC, alarm and out data module. The RTC module is the module that will do all the counting and store the real-time value. The function of every module in the block diagram.

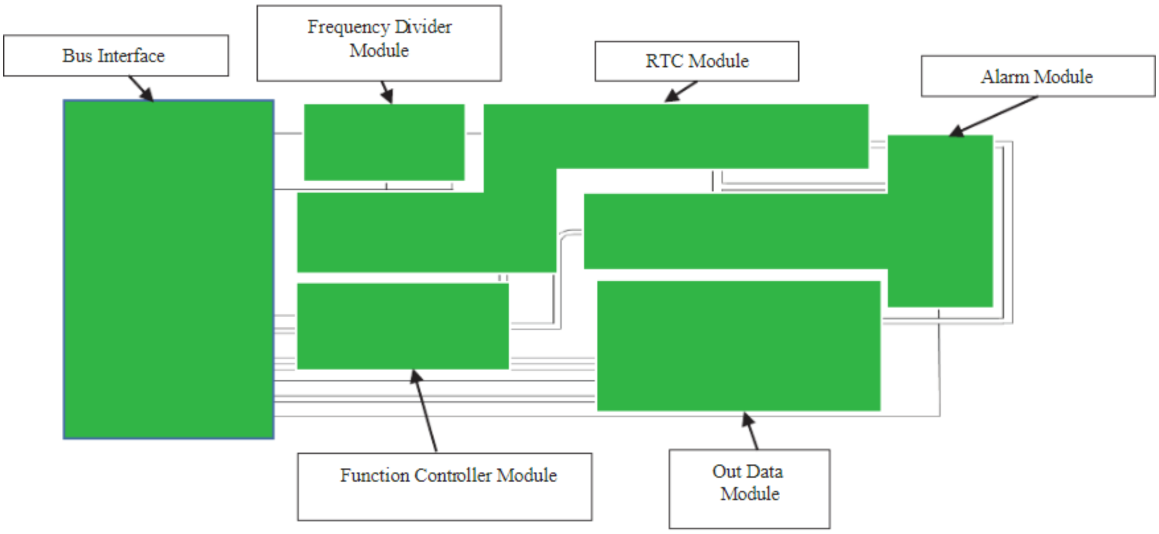


Fig 2.1 RTC Block diagram

## 2.3 FLOWS OF OPREATION

Firstly, the RTC will check for read and write data event. If there is data writing for manual set the time and date, the register inside the RTC module then will be overwritten by a new input value and will continue counting starting from the new value. If there is no writing data event or only the set alarm event occurs, the RTC will start counting from its initial value as stated in the flowchart. If reset is toggle, the RTC will goes into its initial condition that is 00:00:00 (hh:mm:ss) & 01/01/2000 (dd/mm/year). If not the current time and date value will be store into outdata module register. This value will be update and store for every 1 second. To read the data from the RTC, all the condition need to be meet which is the address is call, PSEL and PENABLE is toggle, then the date and time value will be concatenated and transferred to PRDATA (output pin) in form of 32 data bit. For the alarm module, if the alarm is set, the alarm value will be compared with the real-time value. When the value is matched, an interrupt signal will be sent out to CPU as alarm signal.

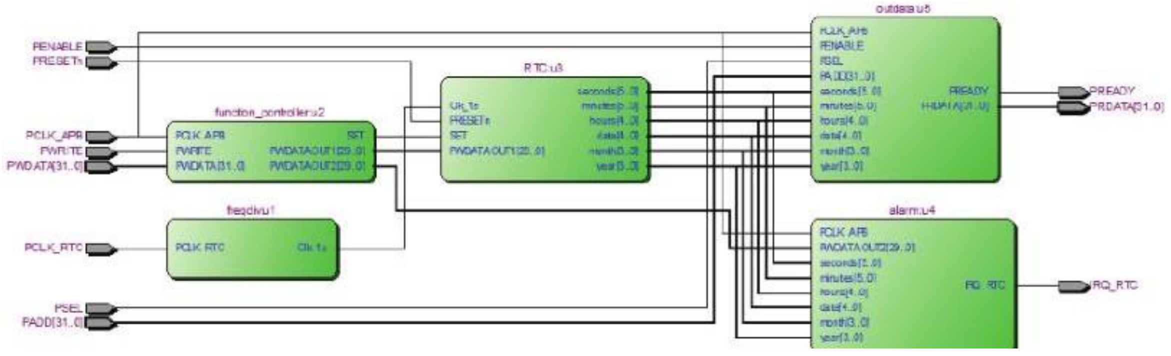


Fig 2.2 Opreation of RTC

# CHAPTER-III

## 3.1 PROJECT PROCESS FLOW

Initially, the current time value is inputted through a keypad. This value is displayed on the LCD screen. The second value is incremented by one; when it reaches 60, the minute is updated by one, and the second is reset. This process continues until the minute reaches 60. When the minute value reaches 60, the hour is updated by one, and the minute and second are reset to zero. Once the hour value reaches 24, the hour value is reset to 00. The process of updating seconds, minutes, and hours is continued as mentioned above. The above operation is repeated infinitely. In certain circumstances like battery failure, the time is to be inputted through Keypad. The program displays appropriate messages to help the user to enter the time value. Once the time value is entered, the program maintains and updates this time successfully.

## FLOW DIAGRAM

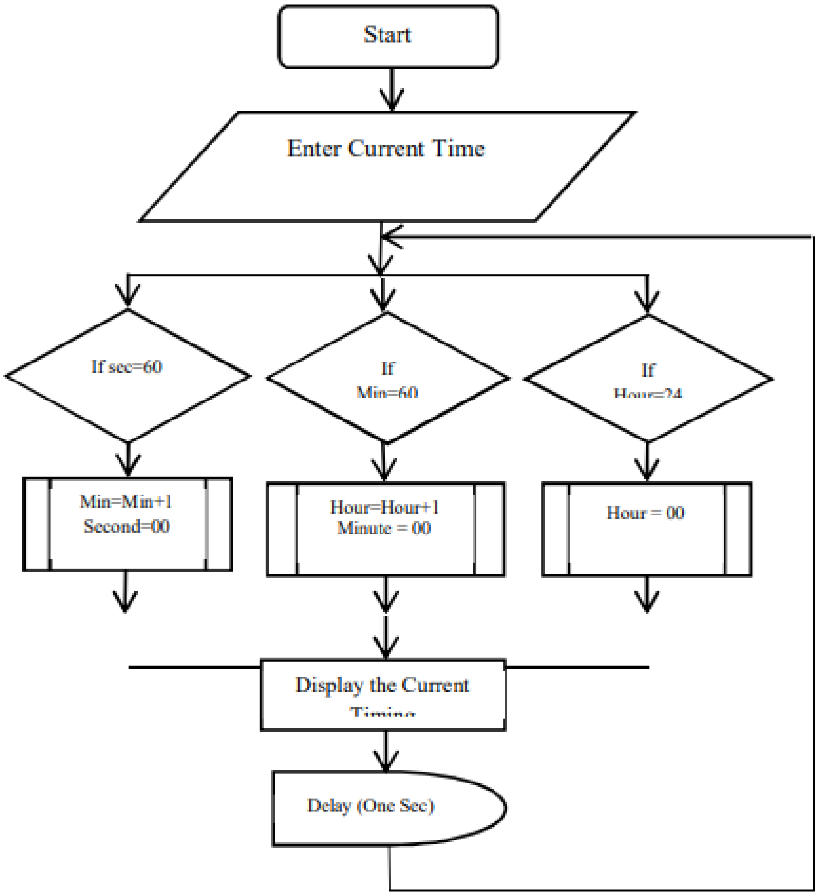


Fig 3.1 Flow graph

## 3.2 BASIC ARCHITECTURE OF RTC

A real-time clock is an integrated circuit that contains a timer generated by a crystal oscillator that supplies the time of day and often, with the date. RTC has a small memory on-chip that stores time and date description values. The time and date values are the years, month, date, hours, minutes, and seconds. In this modern era, an electronic device becomes smaller. The smaller size becomes a concern of electronic technology. By using conventional RTC to provide time and date system to a microcontroller will make the device bigger. So, a dedicated RTC for the customized microcontroller is proposed to provide its own time and date system as a system-on-chip feature on the microcontroller. In this project, a dedicated Real Time Clock (RTC) for Customized Microcontroller unit is developed as an Intellectual Property (IP) and will be covered only the logic part of RTC. The development of the dedicated RTC is to provide the customized microcontroller with its own time and date system for various time-based application to be managed precisely without an external RTC to be used with the microcontroller. This documentation continues the proposed architecture of the RTC on the previous work into the ASIC flow for layout generation and some analysis on the layout stage to optimize the layout in term of size and power consumption. From the literature review, there are many designs of RTC nowadays. Every RTC has their own features according to their specification to be used on any application. Even they have different features, they still have a similar part to make the RTC function correctly such as power management, an oscillation circuit, a communication interface, and memory for time value storage.The color box in Figure 1 shows the conventional RTC basic architecture. The red box shows the oscillation circuit and prescaler or frequency divider of that RTC. The blue box is the power supply or power management for the RTC. The yellow box will be their register which will store the time and date value generated by the counter. Lastly, green box shows their bus interface for the RTC which will be used to interface the RTC with an external application. All four of this basic architecture is essential for RTC to work correctly.

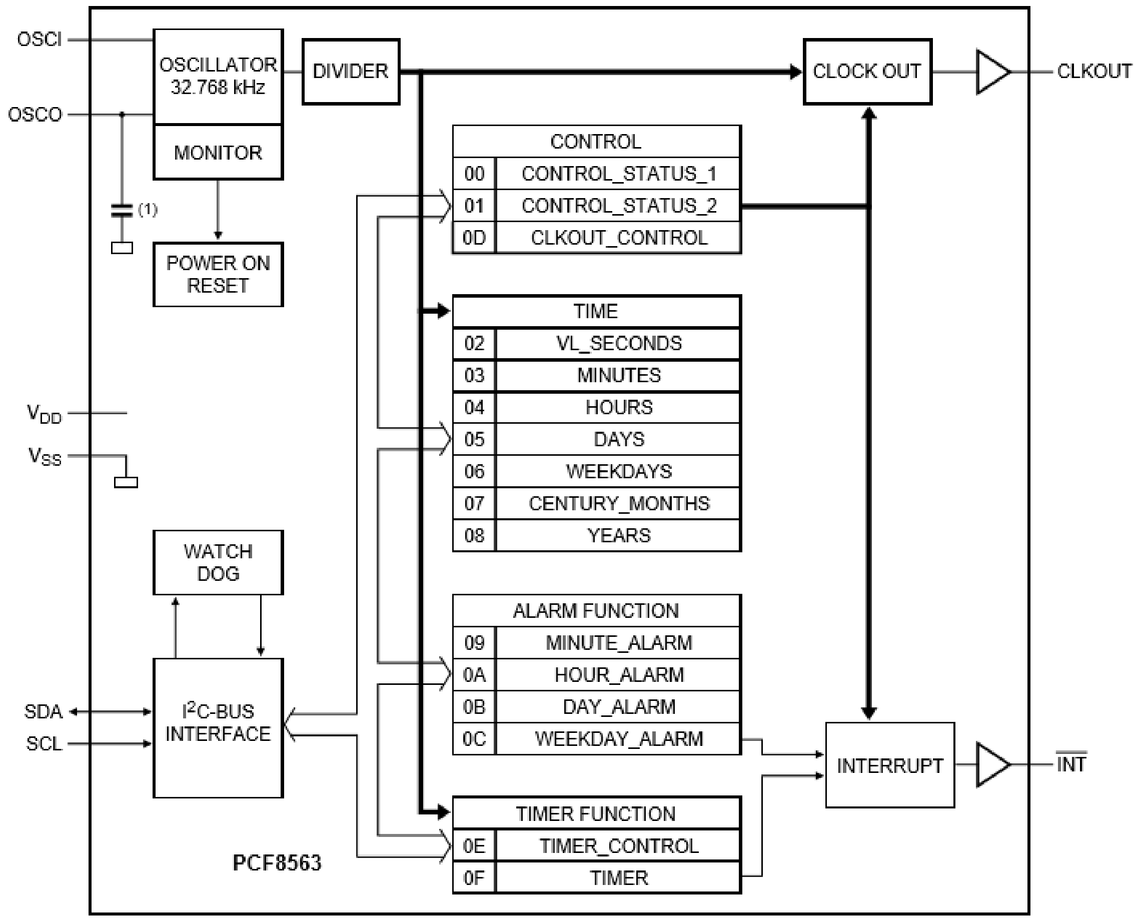


Fig 3.2 Architecture of RTC

## 3.3 OSCILATION CIRCUIT

An oscillation circuit is an electronic circuit that produces a periodic, oscillating electronic signal, often a square wave. They are widely used in many electronic devices. Common examples of signals generated by oscillators is a clock signal that regulates computers and quartz clocks. Some of the factors that affect the frequency stability of an oscillator generally include the variations in temperature, variations in the load as well as changes to its DC power supply voltage and many other factors. The frequency stability of the output signal can be improved by the proper selection of the components used for the resonant feedback circuit including the amplifier. The frequency stability is needed in RTC because it plays an important role. The clock generated will base on the frequency and if the frequency is not stable, it will affect the counting thus affect the accuracy of the RTC.To obtain a very high level of oscillator stability, a Quartz Crystal is generally used as the frequency determining the device to produce another type of oscillator circuit known generally as a Quartz Crystal Oscillator (XO). There are many different types of crystal substances which can be used as oscillators with the most important of these for electronic circuits being the quartz minerals because of their greater mechanical strength. The physical size and thickness of a piece of quartz crystal are tightly controlled since it affects the final or fundamental frequency of oscillations. Then once cut and shaped, the crystal cannot be used at any other frequency. In other words, its size and shape determine its fundamental oscillation frequency. Also, in the crystal oscillator, there are many types of the crystal oscillator with a different configuration for different specification such as Colpitts Crystal Oscillator, Pierce Oscillator, and CMOS Crystal Oscillator. As for their oscillation frequency, all RTC use the 32.768kHz crystal oscillator due to the stability of the crystal.

## 3.4 BUS INTERFACE

For conventional RTC, there are two types of interface technique used which is I²C (Inter-Integrated Circuit protocol) and SPI (Serial Peripheral Interface). These two interfaces have their own advantage depending on their application. Both protocols are well-suited for communications between integrated circuits, or with onboard peripherals. I²C is developed by Philips (now known as NXP Semiconductor). It is a simple bidirectional 2-wire bus for efficient inter-IC control. With this bus interface, only two bus lines are required that is a serial data line (SDA) & a serial clock line (SCL). Serial, 8-bit oriented, bidirectional data transfers can be made at up to 100 kbit/s in the Standard-mode, up to 400 kbit/s in the Fast-mode, up to 1 Mbit/s in the Fast-mode Plus, or up to 3.4 Mbit/s in the High-speed mode. The Ultra-Fast-mode is a uni-directional mode with data transfers of up to 5 Mbit/s. SPI is developed by Motorola. This interface provides full-duplex synchronous serial communication between master & slave devices. It is commonly used for communication with the flash memory, sensors, real-time clock (RTCs), analog to digital converter and many more.

# 3.5 ADVANCE PERIPHERAL BUS

The propose RTC [4] is a dedicated RTC for a microcontroller and will be as an onboard RTC. The microcontroller will use an Advanced Microcontroller Bus Architecture (AMBA) bus protocol as their bus interface. The proposed RTC needs to follow the AMBA bus protocol to be interfaced with the microcontroller

bu sily
in vith
A BA
hi for
m face
to low
ba sing
A
Sp

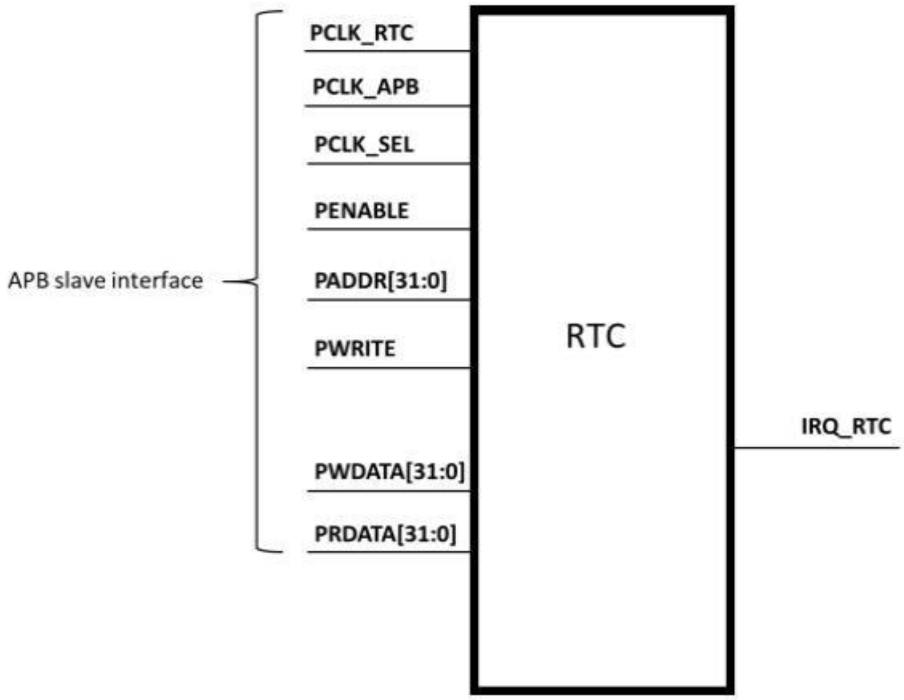


Fig 3.3 Peripheral Bus

## 3.6 CONCLUSION

In a conclusion, the proposed design of the RTC in the previous work has been put through the ASIC design flow to generate the layout. The generated layout then is optimized in term of size and power consumption. This work will be continued to the sign off stage before being sent out for fabrication. This IP will be integrated into the microcontroller alongside with the other IP to be fabricated. The main contribution of this RTC development is to provide the customized microcontroller with its own time and date system to manage various time-based application without external RTC being used with the microcontroller.