Power Aware Automatic Microcontroller Based Smart, College Electric Bell System with Time Display

Rajesh Kannan Megalingam, Venkat Krishnan Balasubramanian, Mithun Muralidharan Nair, Vineeth Sarma Venugopala Sarma, Rahul Srikumar
Amrita Vishwa Vidyapeetham, Kollam, Kerala, India.
E-mail: rajeshm@amritapuri.amrita.edu, venkatkrishnan.b@gmail.com, mithun34@ieee.org, vineethisalways4u@gmail.com, rahul44@ieee.org

Abstract: This paper discusses the hardware implementation of a low power automatic bell which can be used in educational institutions and the power estimation details. The bell automatically rings at preprogrammed time intervals and also displays the time in the seven segment display continuously. The low power aspect is brought about by using 'SLEEP' mode in PIC microcontroller which keeps the system in idle state when it is not in use. The major advantage of this implementation is that it gives us the exact time and no manual operation is needed while using very less power. The system is made by establishing a serial interface between PIC microcontroller (PIC16F877A) and a Real Time Clock (RTC) IC, DS1307. The software coding part is done using MPLAB IDE and hardware implementation is done using the components. The estimated power consumption is also given.

Keywords- Real Time Clock (RTC); PIC; bell; Timer.

I. INTRODUCTION

Most of the educational institutions in India and other developing countries have their college/school bell rung manually after each hour of class. Manually controlled bell may cause some problems which include the lack of accuracy in time and the person who is in charge of ringing the bell might forget. If we consider the automated bell system (No human intervention needed), it works throughout the day. But it should be noticed that, there is wastage of power. We have analyzed this problem and have come out with a good solution to reduce the power consumption in the automated bell system with the 'SLEEP' mode. One more advantage of this system is that it also has a clock display which exactly displays the time and rings at the exact time.

The initial part of the paper describes the basic construction and working principle of the system. The second part explains the basics of the Real Time Clock (RTC DS1307). The third part of the paper discusses the timer modules of the PIC16F877A. The final part of the paper deals about coding, testing and the hardware implementation of the bell with the time display. The circuit that has been implemented is also discussed in the paper.

II. WORKING PRINCIPLE

The block diagram of the automated bell system is shown in the Fig.1. The heart of the circuit is the PIC microcontroller. The microcontroller we have used is PIC16F877A which is the master device. The slave device is the RTC IC DS1307, which automatically counts every second, once enabled. The intervals of time after which the bell should ring is already programmed and loaded into the microcontroller. Once the time that is fixed matches with the time in the RTC clock, the bell rings. The bell rings continuously for a fixed time (6 seconds in our implementation) which is also mentioned at the time of programming.

The circuit is implemented by enabling the I^2C interface of the PIC microcontroller. This is interfaced with the I^2C interface of the RTC IC DS1307. The interfaces of the two ICs are explained in the section V of the paper. It is through this serial interface that the exact time is read into the PIC microcontroller and is compared against the set of time in the code. If the present time matches with the time that is set in the program, that is when the bell should ring, logic HIGH is driven to the output port of the microcontroller. This small voltage (5V) acts as the enable to the relay circuit, which turns on the 230V to the bell and the bell rings.

Another part of the system is the time display. The time value read into the microcontroller from RTC is also given as output through its port pins every instant to be displayed, along with comparing the values internally. The output value from the microcontroller pins are displayed in the seven segment display, which gets automatically updated every minute.

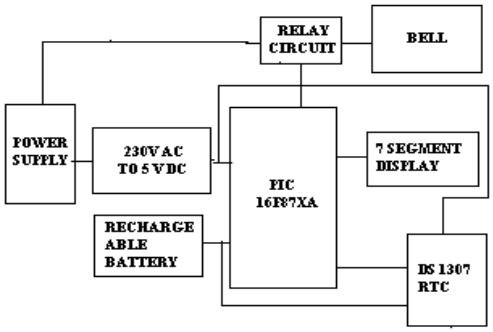

Fig.1 Block diagram of the automated bell system

978-0-7695-3938-6/09 $25.00 © 2009 IEEE
DOI 10.1109/ICMENS.2009.41

III. MASTER MODE OF PIC MICROCONTROLLER

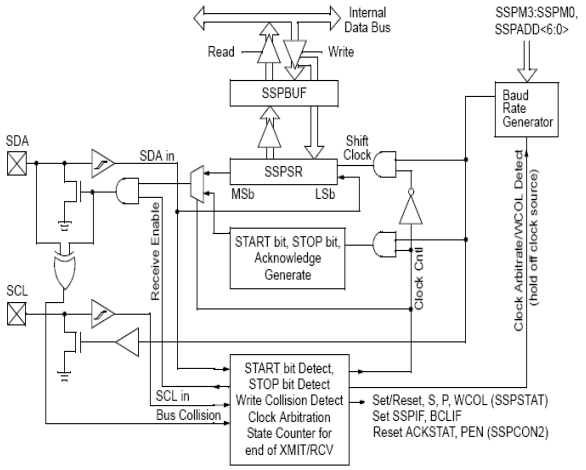

Fig 2. The master mode of the microcontroller PIC16F87XA [1].

In the application of the automatic bell that we have used, the microcontroller is configured as the master device. Microcontroller serially communicates with the RTC (DS1307), which is the slave device. The registers that have to be programmed include SSPCON, SSPCON2, SSPBUF, etc. The operation of PIC I^2C module as the master is explained below.

In the Fig 2, the SCL pin is the serial clock pin of the microcontroller and SDA pin is the serial data pin. The baud rate generator generates the clock; the clock frequency can be modified by manipulating the bits in the registers of the microcontroller. The data to be sent to the RTC first reaches the SSPBUF and then copied to the SSPSR register. The data from the SSPSR register is serially shifted out through the SDA and the SCL pin transmits the clock to regenerate the data on the receiver side. When the data is received, the data appearing at the SDA pin is sampled with the clock in the SCL pin and moved to the SSPBUF register. On receiving the data, ACK is generated and it is sent. When the data is sent the ACK is received.

IV. DS1307 SERIAL INTERFACE

The DS1307 is a serial Real Time Clock that consumes less power and has a BCD calendar. It also incorporates a 56 byte NV RAM. The IC also features an in-built power- supply sense circuit that is capable of detecting power failures and switching to backup supply. The power consumption in battery backup mode is trivial (about 500nAmps).The time keeping operations continue while in back supply mode. The DS1307 has a programmable output square wave signal.

The DS1307 is a slave by itself. It cannot initiate any activity. To initiate communication between the PIC and the DS1307 RTC, the I^2C module needs to be configured in the master mode. There are 2 registers that map the functionality of the I^2C module namely, SSPCON and SSPCON2.The binary value 1000(decimal-8) is moved to the lower order four bits of the SSPCON register. This enables the I^2C module inside the PIC to function in master mode.

Data can be written or read from the RTC. These operations are done in the Slave receiver (write) and Slave transmitter (read) mode respectively, where the DS1307 RTC acts as the slave. Our paper involves the use of the DS1307 IC in the Slave receiver mode. The DS1307 has an 8-bit control register whose bits can be manipulated to perform desired functionality.

V. DEVICE CONFIGURATION

To enable the RTC (DS1307) the configuration should be done by setting the bits in the RTC registers. This is also done using the I^2C interface. Once the microcontroller (PIC16F877A-Master) and the RTC (Slave) are connected as shown in the Fig. 4, the configuration should be done. The SDA pin or the data line remains at high when no data is transmitted. The start of the data transfer occurs when the state of the SDA pin changes from HIGH to LOW.

After each byte is received an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Hardware performs address recognition after reception of the slave address and direction bit (Read/Write). The slave address byte is the first byte received after the master generates the START condition. The slave address byte contains the 7-bit DS1307 address, which is 1101000, followed by the direction bit (R/W), which for a write is 0 [2]. After receiving and decoding the slave address byte, the DS1307 outputs an acknowledgement on SDA. The device address is the same for any transaction. If the configuration is correct, then a square wave is obtained at the output, through the SQWE pin. The configuration can be checked by setting the microcontroller as the receiver and the data from the configuration register is read. The Fig 5 (b) shows the read mode of the RTC.

When the configuration is complete the RTC is enabled and it starts updating the time. Every second, the time is read into the

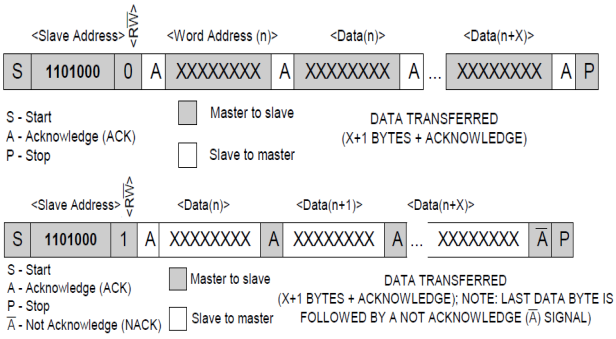

Fig 4.(a) The slave in receiver mode (b) The slave in transmitter mode [2].

microcontroller and is compared with the time stored in the microcontroller registers. If the current time read from the RTC, matches with the time stored in the microcontroller registers, a HIGH output is given in the port pins A<5:4> of the microcontroller, which in turn operates the relay circuit (mentioned in section VII). Also the updated time from the microcontroller is given to the seven segment display to display the accurate time.

The transfer of data and the clock through the SDA and SCL pin is serial with MSB transmitted first. As shown in Fig.5, first the start condition occurs and then the master sends the device address serially, along with the R/W bit. The master then waits for the acknowledgement. Once the ACK is received, the address of the location in the RTC is transmitted serially. For example the control register has the address 07H. Now the data to be written is transmitted. The serial data transmission can be viewed with the help of a DSO. Finally, at the end of the transmission, a STOP condition is generated.

VI. Analog Circuit

The analog part of the design uses fundamental electric circuits for designing the power supply. As PIC16F877A needs a DC power supply, there was a need for an AC to DC converter. The supply voltage was optimized to 5 V DC. In order to step the voltage down from 230 V AC to 6 V AC we use a center tap step down transformer having a secondary winding specification of 6-0-6,among which 6-0 are used. The 6V Peak-Peak AC voltage at the output is fed into a full wave rectifier circuit. The full wave version of 6V AC is fed into a regulator IC LM7805 because we need an optimum VDD of 5V. Relay circuit plays a significant role in the analog part. According to the TIMER 1 of PIC16F877A, a high voltage level of 3.5 V with 20mA current appears at the output pins of the microcontroller. This signal is current amplified to 60mA by using a CMOS driver IC so as to drive the relay circuit.

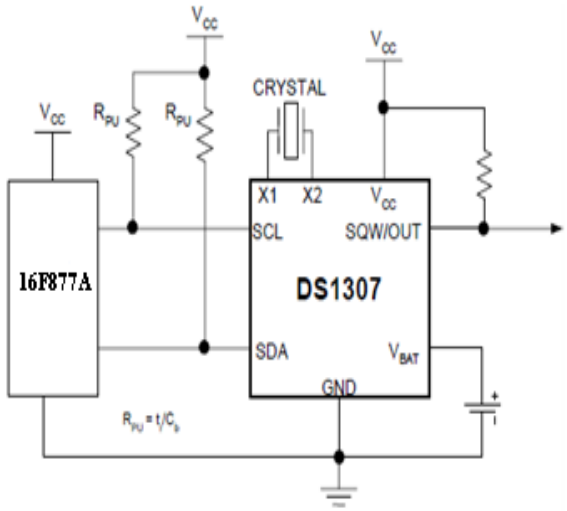

Fig 5. The I^2C Interface between the microcontroller and RTC.

VII. Specifications

Our university, Amrita Vishwa Vidyapeetham, has its working hours from 8:25am to 4:30pm. The day starts with the prayer at 8:25am and each class is 50 minutes long. There are also two coffee breaks of 10 minutes each and a one hour long lunch break. The microcontroller is programmed with the timings, when the bell should ring and also enable the RTC. Altogether the bell should ring 13 times in a day. At each instant the current time is matched with the time when the bell should ring and when a match occurs, logic HIGH is obtained at the output port pin, which is given to a current amplifier. The 20mA current from the microcontroller pins when passed through the amplifier is given to the relay circuit which switches on the relay circuit. Now the bell gets a direct connection to the 230 V power

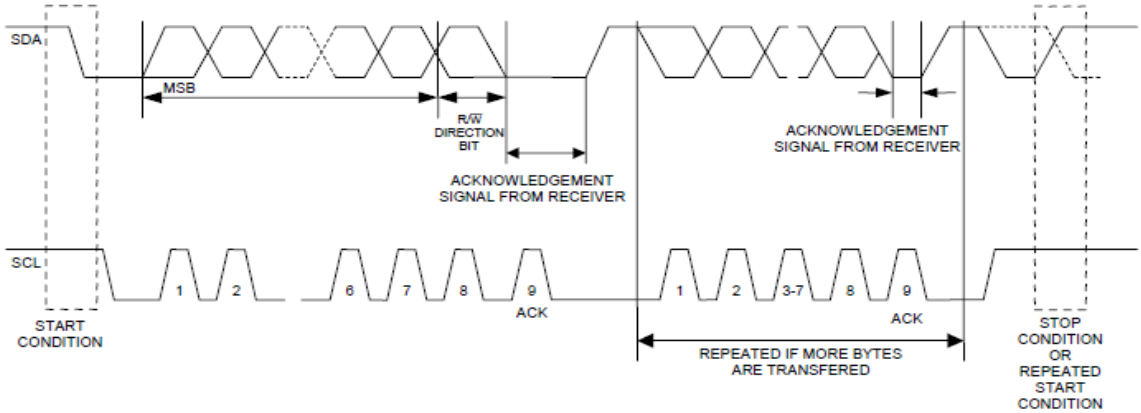

Fig 6. The waveform depicting the transaction between the master and the slave device.

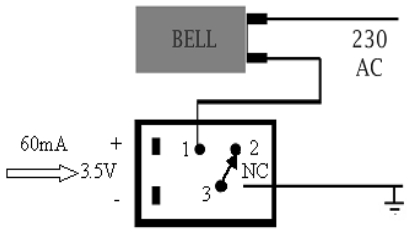

Fig 7. The relay circuit to control the electric bell.

Supply and the bell rings. The time for which the bell should ring is programmed in the microcontroller which can be defined by the user. We have given a time period of 6 seconds.

VIII. EXPERIMENTS AND RESULTS

The initial part of the testing was done using the TIMER0 and TIMER1 modules of the microcontroller. The TIMER1 module counts for 50 minutes or 10 minutes according to the necessity and the bell rings for 6s after the particular time. The assembly code was written using MPLAB. The next stage of testing was to enable the I^2C interface between the master and the slave ICs. This was to ensure better accuracy in timing. The assembly code was written to configure the interface, write the control registers of the RTC and to read and update the time that is read into the microcontroller. The output of the microcontroller was connected to the seven segment displays to check the working of the RTC.

The next stage of the implementation was to check the timing every second and to trigger an LED if the time of the RTC matches with the time that is stored in the registers. Testing was also done with the analog part of the circuit which includes the relay circuit. One of the problems we faced was that it was not possible to drive the relay circuit with the 20mA current from the microcontroller. So the output of the microcontroller was given to a current amplifier and it was tested.

A rechargeable battery is also integrated with the circuit, because if there is no battery connected to the circuit the PIC and the RTC would get reset every time the power failure occurs. The battery that is connected gets charged when there is power supply and during power failure the microcontroller is still kept working by the 5V V_{CC} from the battery.

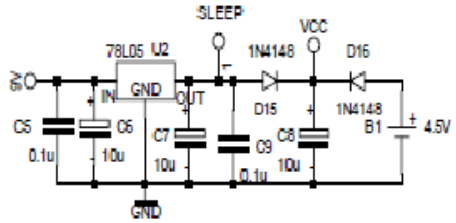

Fig 8. The circuit used for implementing SLEEP mode.

The 'SLEEP' node in the Fig.8 is connected to port pin RB7 of the microcontroller. The SLEEP mode circuit has capacitors, diodes, etc connected to the microcontroller through a resistor. The Fig.9 shows the entire circuit of the automated bell system where the interface between the microcontroller and the RTC IC is shown. It also depicts the output taken from the microcontroller, given to the seven segment displays to display the time.

The comparison of the power consumption can be made by taking two bell systems into consideration, the automated bell with no sleep mode and the automated bell with the sleep mode. The difference is shown in Table I in terms of energy for the microcontroller alone. The power consumption due to the RTC, the bell, the seven segment display is given in Table II. To find the power consumption of the display, it should be noted that all the seven segments does not work all the time. So the average number of segments working is found out and used in the calculations. The quantity in the Table III depicts the difference in the energy in an automated system with and without SLEEP mode.

TABLE I. POWER CONSUMPTION COMPARISON FOR MICROCONTROLLER IN THE TWO SYSTEMS

MODE	Total time in normal mode	Total time in SLEEP mode	Total Energy (J)
Automation without SLEEP mode	29100 seconds	Nil	29100
Automation with SLEEP mode	78 seconds	29022	107

TABLE II. POWER CONSUMPTION FOR OTHER COMPONENTS IN THE SYSTEM

DEVICE	Time of operation per day (s)	Power consumption (W)	Energy consumed per day (KJ)
DS1307	29100	0.0005	0.02
BELL	78	25	1.95
7 SEGMENT DISPLAY	29100	5.414	157.55
TOTAL			159.52

TABLE III. TOTAL POWER CONSUMPTION FOR THE TWO AUTOMATED SYSTEMS

MODE	Energy (KJ)		
	PIC16F877A	Other modules	Total
WITHOUT SLEEP	29.100	159.52	188.62
SLEEP	0.107	159.52	159.62

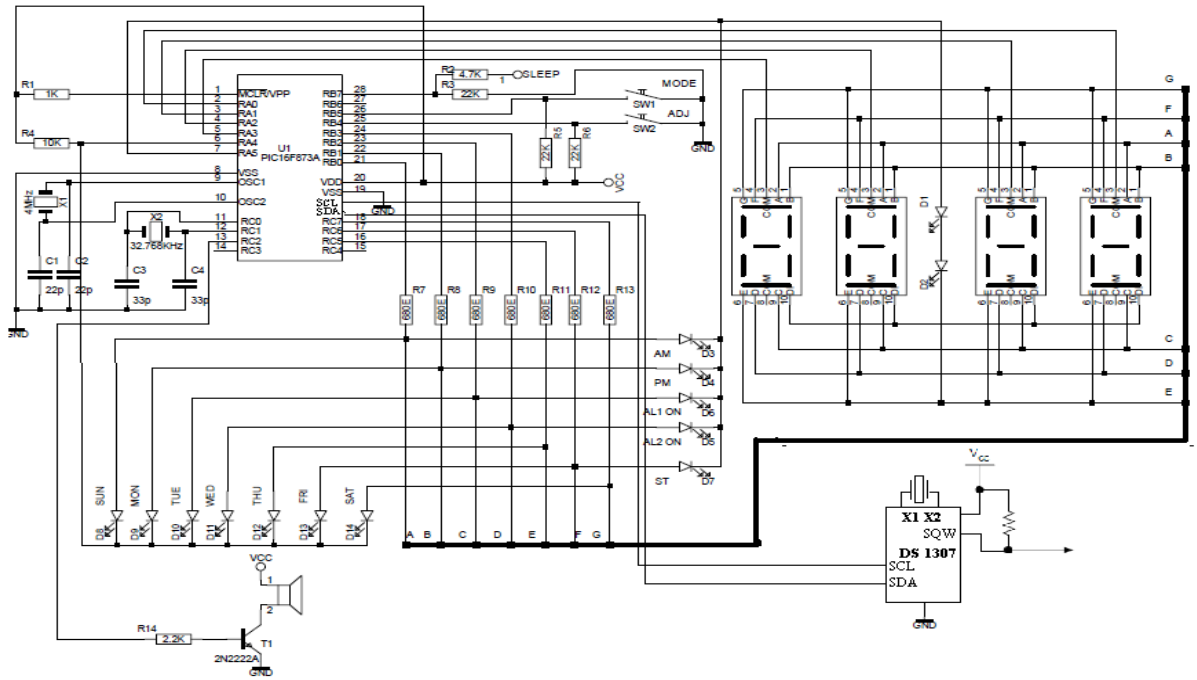

Fig 9. The circuit showing the I^2C interface between the microcontroller, the RTC and the time display.

From Table III, it can be seen that when SLEEP mode is not implemented the net energy consumption is 188.62 KJ and when the SLEEP mode is introduced, the energy consumed decreases to 159.62 KJ. So, this method can be considered to be more energy efficient.

IX. Conclusion

The assembly code for the automation of the college bell was written and tested in the hardware. The transmission of the data from the master to the slave and slave to the master was verified using a DSO. The hardware was first tested using an LED, instead of a bell and then was integrated with the relay circuit to ensure the working of the bell. The SLEEP mode that is introduced helps in conserving the power as the bell enters the SLEEP mode when it is not ringing. The bell timing can be set according to the institution and it requires no manual operation. The power estimation of automated bell system with and without sleep mode shows that, 10.07% of total power is saved in automated bell system with SLEEP mode implementation.

Acknowledgment

We would like to acknowledge the Almighty GOD who gave us strength and health to successfully complete this venture. We also wish to express our heart-felt gratitude to Amrita Vishwa Vidyapeetham, in particular the Digital library, for access to their research facilities and for providing us the laboratory facilities for conducting the research. The authors wish to express their deep gratitude to all persons who helped them directly or indirectly towards the successful completion of this paper.

References

[1] PIC 16F87XA Microcontroller datasheet, Microchip Technologies Inc.
[2] DS1307 Real Time Clock IC Datasheet, Maxim, Dallas Semiconductors.
[3] Adel S Sedra , Kenneth C Smith 'Microelectronic circuits', 5th edition, Oxford University Press.
[4] KIA 7805 Datasheet, Korea Electronics Ltd.
[5] T.R Padmanabhan 'Introduction to microcontrollers and their applications', Narosa Publication.
[6] B.L Theraja 'A Textbook of Electrical Technology', paperback.
[7] Arun Kumar Singh, 'Microcontroller and Embedded System' , 1st Edition.
[8] Ms. Seint Seint Htwe " Remote Token Display and Sound System", World academy of science, engineering and technology.
[9] Bylander, E.G., "Electronic Displays," McGraw-Hill Edition, Inc., 1979.
[10] Floyd, T.L., "Electronic Devices I and II," 4th ed., Prentice Hall International, Inc., (1996).
[11] Wilkinson, B. and Makki, R., "Digital System Design," Prentice Hall, International (UK) Ltd., 1992.