Mini Project Report

on

Digital Real Time Clock with Alarm

submitted in partial fulfillment of the requirement
for the award of the Degree of

Bachelor of Engineering
in
Electronics Engineering

by

Pallavi Avle
Tejas Badgujar
Brendon Faleiro

under the guidance of

Dr. S. S. Rathod

Department of Electronics Engineering
Bharatiya Vidya Bhavan's
Sardar Patel Institute of Technology
Munshi Nagar, Andheri-West, Mumbai-400058
University of Mumbai
2013 - 2014

Digital Real Time Clock with Alarm

Pallavi Avle[*], Tejas Badgujar[†], Brendon Faleiro[‡]
Sardar Patel Institute of Technology,
Mumbai-400058, India
[*] avlepallavi@gmail.com [†] tejas.badgujar9022@gmail.com [‡] bren.faleiro@gmail.com

***Abstract*—A real-time clock (RTC) is a battery-powered clock that is included in a microchip in a computer motherboard. The real-time clock of the MSP 430 kit is used to implement a digital clock. The clock displays the current time in hours, minutes and seconds on the terminal and sounds a buzzer when the alarm time is reached.**

I. INTRODUCTION

The Real Time Clock of a microprocessor system has several applications. These applications span a wide array ranging from a digital clock to being used as a time-stamp in digital cameras and GSM modules.

A. What is a Real Time Clock?

This microchip is usually separate from the microprocessor and other chips and is often referred to simply as ”the CMOS” (complementary metal-oxide semiconductor). A small memory on this microchip stores system description or setup values - including current time values stored by the real-time clock. The time values are for the year, month, date, hours, minutes, and seconds. When the computer is turned on, the Basic Input-Output Operating System (BIOS) that is stored in the computer’s read-only memory (ROM) microchip reads the current time from the memory in the chip with the real-time clock.
A real-time clock (RTC) is 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.
These Real-time clock (RTC) ICs measure time, even when the power of the main device is off. During these times, RTC ICs draw power from an auxiliary battery like an internal lithium battery or supercapacitor. As expected, power consumption is a key factor in most RTC designs, but accuracy and small package size are also important. Most modern RTC ICs reduce package pin count by supporting a serial interface.
It plays a very important role in the real time systems like digital clock, attendance system, digital camera etc. In applications where time stamp is needed, RTC is a good option. Using RTC for designing such application has always been a good designers choice although the beginning might be a bit difficult.
While designing any real time system which deals with time, there are two ways of handling the time factor. One is to generate the time internally which is done by programming the timers of the controller; and the other is to use an RTC.
Although the RTC in principle is similar to other timers on the microprocessor, it does have a few important features that distinguish it from the other timers. Some of these distinguishing features have been compared in Table I.

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. This frequency is exactly 2^{15} cycles per second, which is a convenient rate to use with simple binary counter circuits.

Parameter	Internal Timers	RTC
Error	High and keeps increasing with time	Almost negligible. Of the order of 1 sec in 100 years
Memory space usage	High	Low
CPU usage	High	Low
Ease of Interfacing	Easy	Difficult
Program complexity	High	Low
Efficiency of system	Low	High
Achieving goals of the system	Difficult	Easy
Cost	Low	High

TABLE I: Comparison between an internal timer and an RTC.

B. *Purpose of RTC:*

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

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

Our project uses these features of the RTC in an MSP430 kit to implement a digital clock system with an alarm facility.

II. HARDWARE DESIGN

The system is designed using an MSP430G2553 and a buzzer. The MSP430 is connected to the PC Console where the output is constantly updated to provide us with the current time. The digital clock is implemented by using the RTC of the MSP430. The Texas Instruments MSP430 family of ultralow-power microcontrollers consists of several devices featuring different sets of peripherals targeted for various applications. The architecture, combined with five low-power modes is optimized to achieve extended battery life in portable measurement applications. The device features a powerful 16-bit RISC CPU, 16-bit registers, and constant generators that contribute to maximum code efficiency. The digitally controlled oscillator (DCO) allows wake-up from low-power modes to active mode in less than 6 s. The MSP430x11x1(A) series is an ultralow power mixed signal microcontroller with a built-in 16-bit timer, versatile analog comparator and fourteen I/O pins.

A. *RTC in MSP430:*

All MSP430 devices can implement an RTC. MSP430 devices contain both a digitally-controlled RC-type oscillator and a crystal oscillator.
The RC-type oscillator is typically used for the CPU clock and the crystal oscillator is typically used for the peripherals. In the real-time clock application, the crystal oscillator is used as the clock source for the timer/counter that serves as the time base. Therefore, the instability issues that are common to RC-type oscillators are irrelevant.

B. *Real-Time Clock Implementation:*

The general implementation of the RTC is simple. It consists of a timer/counter giving 1-second interrupts and a small CPU routine to count the interrupts. The CPU can sleep or perform other functions between interrupts.
The clock setup for the RTC implementation uses the LFXT1 oscillator in LF mode with a 32768-Hz crystal. The output of the LFXT1 oscillator is selected to source ACLK. ACLK is then selected as the clock source for the timer/counter that serves as the time base for the RTC.
The DCO generates the CPU clock, MCLK. The CPU actually runs asynchronously to the timer/counter peripheral. Accuracy of the RTC is not affected as long as the CPU is able to service each interrupt from the timer peripheral before the next interrupt arrives.

C. *Crystal Accuracy and Selection*

Crystal accuracy is affected primarily by two things, namely, the crystal frequency tolerance, and the specified load capacitance.
The tolerance of the crystal is self-explanatory: the tighter the tolerance on the crystal frequency, the more accurate the RTC will be. For example, an RTC using a 30-ppm crystal is more accurate then

one using a 200-ppm crystal. The specified load capacitance of the crystal also affects the accuracy of the RTC. The load capacitance of a crystal is the amount of capacitance required by the crystal, not the amount provided by the crystal. The crystal requires the proper load capacitance to oscillate at its specified frequency. The 32768-Hz oscillator on all MSP430 devices has integrated load capacitors with a specified nominal capacitance of 12 pF. This provides an overall 6-pF load for the crystal because the capacitors add serially.

In addition, the stray capacitance of the board and pins add in a parallel fashion, thus increasing the capacitive load seen by the crystal. For example, 2 pF of stray capacitance is not uncommon on a circuit board. The addition of the 2 pF to the 6 pF yields a total capacitive load for the crystal of 8 pF. In some cases, depending on crystal specification and stray capacitances, parallel capacitors must be added to the circuit to provide the proper capacitive load for the crystal.

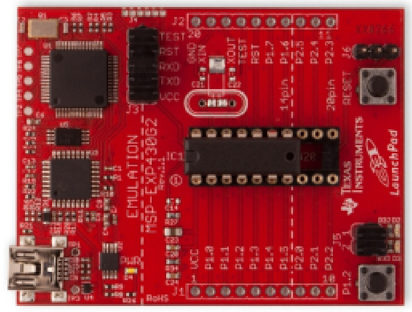

Fig. 1: MSP430 Launchpad

D. MSP430 (Mixed Signal Microprocessor) Features:

1) Low Supply Voltage Range 1.8 V to 3.6 V
2) Ultralow Power Consumption
    a) Active Mode: 160 A at 1 MHz, 2.2 V
    b) Standby Mode: 0.7 A
    c) Off Mode (RAM Retention): 0.1 A
3) Wake-Up From Standby Mode in Less Than 6 μs
4) 16-Bit RISC Architecture, 125 ns Instruction Cycle Time
5) Basic Clock Module Configurations:
    a) Various Internal Resistors
    b) Single External Resistor
    c) 32-kHz Crystal
    d) High-Frequency Crystal
    e) Resonator
    f) External Clock Source 16-Bit Timer_A With Three Capture/Compare Registers
6) On-Chip Comparator for Analog Signal Compare Function or Slope Analog-to-Digital (A/D) Conversion
7) Serial Onboard Programming, No External Programming Voltage Needed, Programmable Code Protection by Security Fuse

III. SOFTWARE IMPLEMENTATION:

The following code was written and debugged using Energia(0101E0012).

Code for the Digital Clock with alarm:

```
#include "D:\energia-0101E0011\hardware
\msp430\libraries\RTCplus.cpp"

RealTimeClock myClock; // Initializing
//internal RTC
void setup()
{
  // setup code to run once:
  myClock.begin();
  myClock.Set_Time(11,50,00);
// Initializing internal RTC and setting
  myClock.Set_Date(2014,4,19);
  myClock.Set_Year(2014);
  Serial.begin(9600);
  pinMode(2,OUTPUT);
}
float currentTime()
// Returns the current time in Hrs.
{
 return (myClock.RTC_hr +
(myClock.RTC_min/60.0) +
(myClock.RTC_sec/3600.0));
```

```
}

void displayTime()
//Display Time for 30 seconds
{
  for(int i=0; i<=40; i++)
  Serial.println();
  Serial.print(myClock.RTC_hr,DEC;
  Serial.print(" : ");
  Serial.print(myClock.RTC_min,DEC;
  Serial.print(" : ");
  Serial.println(myClock.RTC_sec,
  DEC);
}

void loop()
{
  // Main code runs repeatedly:
  displayTime();
  float x= currentTime();
  int hr = floor(x);
  int minute = floor((x-hr)*60);
  if(hr==11&& minute==51)
  //Alarm time
  digitalWrite(2,HIGH);
  else
  digitalWrite(2,LOW);
  delay(1000);
}

interrupt(TIMER1_A0_VECTOR)Tic_Tac(void)
// Generates Interrupts every 1 Sec
{
      myClock.Inc_sec();
      // Update seconds
}
```

IV. CONCLUSION:

The Digital Clock system was implemented on the MSP430 launchpad. The second-by-second output was read on the console. When the alarm time was reached a buzzer was sounded and kept ringing for an entire minute.

REFERENCES

[1] Ala-Paavola, Jaakko (2000-01-16). ”Software interrupt based real time clock source code project for PIC microcontroller”. Retrieved 2007-08-23.

[2] Enabling Timekeeping Function and Prolonging Battery Life in Low Power Systems, NXP Semiconductors, 2011

[3] US 5893044 Real time clock apparatus for fast acquisition or GPS signals

[4] Application Note 10337, ST Microelectronics, 2004, p. 2

[5] Application Note U-502, Texas Instruments, 2004, p. 13

[6] Application Note 1994, Maxim/Dallas Semiconductor, 2003

[7] http://www.ti.com/lit/ds/symlink/msp430g2553.pdf

[8] MSP430x1xx Family User Guide, literature number SLAU049

[9] MSP430x11x1 data sheet, literature number SLAS241C

[10] MSP430 Family Application Report Book, literature number SLAA024