CIRCUIT CELLAR ONLINE
THE MAGAZINE FOR COMPUTER APPLICATIONS
Circuit Cellar Online offers articles illustrating creative solutions and unique applications through complete projects, practical tutorials, and useful design techniques.

RESOURCE LINKS

A Guide to online information about:

Real-Time Clocks and How to Set Them

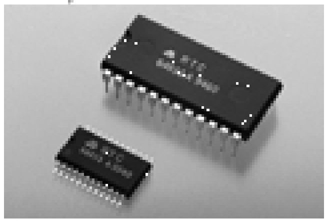
by Bob Paddock

While working on the answer of an *ASK US* question this month, the subject of real-time clock chips came up. I thought it would make a good subject for this month's Resource Page.

What is a real-time clock? A real-time clock (RTC) is, as the name suggests, a clock that keeps track of time in a "real mode," just like the clock on the wall or the watch on your wrist.

This month, I cover the real-time clock chips that are available, how to calculate the day of the week when your hardware doesn't do it for you, as well as touch on the history of Internet Time, Time Modification, and setting your clock to Atomic Standards using free source code.

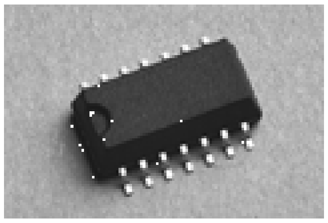
"Does any one know the real time?"



8-bit Multifunctional RTC-64611/64613

The multifunctional RTC features:

- Built-in quartz crystal, which allows adjustment-free and efficient operation 8-bit data bus, high-speed access (85 ns max), and is provided with the same interface as for SRAM and battery backup function.
- Interruption of alarm of second, day, or day of the week, and interruption signal selectable in a range of 1 Hz to 64 Hz . Leap year is automatically adjustable (Gregorian calendar).
- 1 Hz output terminal provided with START/STOP 30 s adjust function.
- RTC-64613 hasa similar mounting method to the universal type SMD IC.



RTC-8583/RTC-8593/RTC-8593SB

This features a built-in Quartz Crystal, thus it's adjustment free with the 10 pF gate capacitor Package (SOP-14 Pin) world's smallest. Three mode operations are available: internal crystal oscillation, external 50Hz, clock and event and counter I2C-Bus interface compatible. Built-in 240× 8-bit SRAM (RTC8583) alarm and timer functions.There is a wide operating voltage range from 2.5 to 6.0 V.And there is a wide data hold voltage range from 1.0 to 6.0 V, with low current consumption The RTC8593typically uses 1.0 µA, . Full application notes are available.

Real-time clocks (RTCs) selection menu.
Application Note: Time-Base Oscillator for RTC ICs.

Benchmarq Products from Texas Instruments

Dallas Semiconductor:

Timekeeping: Year 2000-compliant clocks, MUX Bus clocks, serial clocks, phantom clocks, watchdog timers, and NV Timekeeping RAMs. Includes Data Loggers and Environmental Sensing devices.

The program SWTIME10 that I wrote years ago has the code you need to use with their SmartWatch products.

Some members of the DS5000, 8051 compatible Microprocessor, have built-real-time clocks.

Dallas Semiconductor Produces the First Clock/Calendar Digital Thermometer on a Single Chip

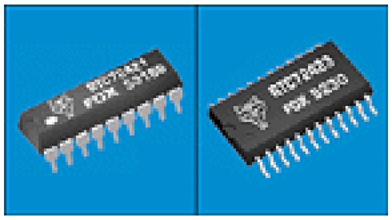
The DS1629 2-Wire digital thermometer and real- time Clock was the first digital system component to incorporate a direct-to-digital temperature sensor, a real- time clock and Y2K-correct calendar on one chip. Previously, designs for digitally monitoring thermally sensitive instruments and operating equipment required a separate chip for each function, with separate programming configurations and interfaces.

Epson:

- Real -Time Clock Module
- Built-in quartz crystal which adjusts frequency
- Low current consumption with C-MOS IC
- Multiple functions meet various needs

RTC PART NUMBERING (PDF,72KB).
Table of Real-Time Clock module (PDF,38KB).
Application Manual of Real-Time Clock module.
Serial, 4-bit, 6-bit, and 8-bit parts are available.

Real-Time Clock Modules



The RTC72421,pin-thru, 18-pin package.and RTC72423, surface mount, 24-pin package are available.

National Semiconductor offers several real-time clocks, for example, the DP8573A.

The DP8573A is intended for use in micro-processor based systems where information is required for multitasking, data logging, or general time and date information. This device is implemented in low-voltage silicon gate microCMOS technology to provide low standby power in battery back-up environments. The circuit's architecture is such that it looks like a contiguous block of memory or I/O ports organized as one block of 32 bytes. This includes Control Registers, Clock Counters, the Alarm Compare RAM, and the Time Save RAM.

Day of week and month counters are provided. Time is controlled by an on-chip crystal oscillator, requiring only the addition of the 32.768-kHz crystal and two capacitors.

Power failure logic and control functions have been integrated on-chip. This logic is used by the RTC to issue a power fail interrupt and lock out the microprocessor interface. The time power fails may be logged into RAM automatically when V_{BB} > V_{CC}. Additionally, two supply pins are provided. When VBB > VCC, internal circuitry will automatically switch from the main supply to the battery supply.

The DP8573A's interrupt structure provides three basic types of interrupts:, Periodic, Alarm/Compare, and Power Fail. Interrupt mask and status registers enable the masking and easy determination of each interrupt.

It's features include a

- Fully functional real-time Clock/calendar
- 12/24 hour mode timekeeping,
- Day of week counter
- Parallel resonant oscillator
- Power fail features
- Internal power supply switch to external battery
- Power Supply Bus glitch protection
- Automatic log of time into RAM at power failure
- On-chip interrupt structure
- Periodic, alarm, and power fail interrupts
-

Product Folder (Datasheet)	Title	Price*	Package Type
DP8570A	Timer Control Peripheral (TCP)	$13.50	PLCC
DP8573A	Real-Time Clock (RTC)	$8.25	MDIP, PLCC
MM58167B	Microprocessor Real-Time Clock	$7.10	MDIP, wafer
MM58274C	Microprocessor Compatible Real Time-Clock	$6.50	MDIP

Philips Semiconductor:

SAA1575HL Global Positioning System (GPS) baseband processor with real-time clock, a 32.768-kHz crystal, and supply for low-power timekeeping.

The SAA1575HL is an integrated circuit, which implements a complete baseband function for Global Positioning System (GPS) receivers. It combines a 16-bit Philips 80C51XA microcontroller, 8 GPS channel correlators, and related peripherals in a single IC. Users can implement a complete GPS receiver using only the SAA1575HL, the UAA1570HL front-end Philips IC (or similar), external memory, and a few discrete components. The IC is aimed at low-cost applications.

It would be interesting to receive the time via GPS to set the internal clock. That way, the issues of when the shift ended would go away.

Overview of real-time clock ICs:

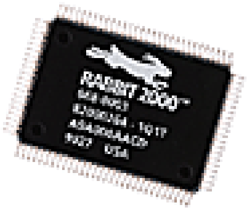
I2C Peripherals:

PCF8563: Real-Time Clock/Calendar
PCF8573: Clock/calendar with Power Fail Detector
PCF8583: Clock/calendar with 240 x 8-bit RAM
PCF8593: Low power clock/calendar.

PCD3745A is 8-bit microcontroller with 4.5-KB OTP memory and 32-kHz real-time clock.

PCD3350A is 8-bit microcontroller with a DTMF generator, 256 bytes EEPROM, and real-time clock.

The PCD3350A is designed primarily for telephony applications. It includes 8-KB ROM, 256 bytes RAM, 34 I/O lines, and an on-chip generator for dual tone multifrequency (DTMF), modem, and musical tones. In addition to dialing, the generated frequencies can be made available as square waves for melody generation, providing ringer operation.



Designed with you in mind, the new Rabbit, from Rabbit Semiconductor, offers a friendly instruction set, fast number crunching ability, and numerous on-chip features like the real-time clock. Of most interest to me was the evaluation system that includes the C compiler and evaluation for an affordable price, board $99.

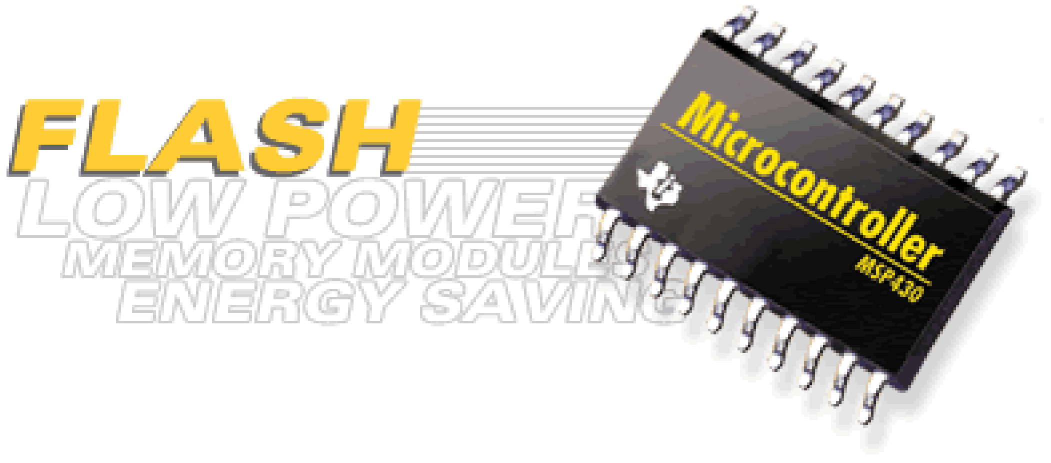
Simtek Corporation is Real-Time Clock (RTC) Design Eliminates Batteries.

Simtek has developed a real-time clock technology that combines its nonvolatile static RAM (nvSRAM) with a miniature capacitor-powered oscillator/counter. Simtek's real-time clock technology, called nvTIME(TM), eliminates the need for a back-up battery, which was previously required to operate clocks when system power is lost.

Simtek's nvTIME technology uses high value double-layer capacitors to provide power to the clock's oscillator when system power is off and uses its proven nvSRAM technology to store all-important data. Eliminating the battery makes the system easier to produce for the manufacturer and more reliable for the user.

Keeping Time with TIMEKEEPER: The most frequently encountered design challenge is timing accuracy. Many applications demand a level of timing precision that is much greater than a standard quartz crystal can deliver by itself.

USING THE MSP430 AS A REAL-TIME CLOCK This application report describes how to use an MSP430x11x device as a real-time clock. The MSP430 family of microcontrollers from Texas Instruments is a family of ultra-low-power, 16-bit RISC microcontrollers with an advanced architecture and a rich peripheral set. The report gives a general overview of the MSP430 family and discusses the real-time clock application in detail. It gives a detailed circuit diagram, a code example, and a general discussion of accuracy and implementation issues.



Now available with Flash memory.

Ultra-low-power microcontrollers applications.

XICOR When you thinks of Xicor, real-time clock is not what comes to mind first.

Xicor's Real-Time Clock Family

Part No.	Alarms	CPUsupervisor	EEPROM	BatSwitch
X1202	Polled	Yes	No	Yes
X1203(A)	IRQ	No	No	No
X1240	No	No	16 KB	Yes
X1241	No	Yes	16 KB	Yes
X1242	Polled	Yes	16 KB	Yes
X1243(A)	IRQ	No	16 KB	No

Some newer devices that are not shown here, will be announced soon.

Software-Based Real-Time Clock (RTC)

While there are a number of 8051-compatible microcontrollers that have built-in, accurate real-time clocks (especially from Dallas Semiconductor), some simple applications may benefit from a software RTC solution that uses the built-in capabilities of an 8051 microcontroller.

This page will go through the development of a simple software-based RTC solution using 8051 Timer 1 (T1). Thus, your software application will have the benefit of an RTC without requiring any additional hardware.

I prefer time keeping Epoch style, where you simply increment a 32-bit unsigned long, once per second. This makes for and fast interrupt routines. The time and date only need to be converted to real-time when it is going to be used by humans.

You can find some 68K and C code that I wrote years ago to implement thisTYME.ZIP on the Circuit Cellar FTP site as the file.

As you can see, there are many real-time clocks out there, but not all of them give you the day of the week. In the late 19th century, a fellow by the name of Zeller developed a simple formula for calculating the day of the week when given the standard calendar date. [1]

Shown below is a snippet of C code that does Zeller's Congruence. It can be easily converted to any other language or machine code.

```
/* Zeller's Congruence: From "Acta Mathematica #7", Stockholm, 1887.

Determine the day of the week give the year, month, and the day of the month;
which are passed in the structure 'utc'

return(  0 = Sunday...6=Saturday ) and also set utc->wday

J = Century (ie 19), K = Year (ie 91), q = Day of the month, m = Month

March = month #3....December = month #12,
January = month #13, February = month #14 OF THE PREVIOUS YEAR.

[q + [((m + 1) ×26 ) / 10] + K + (K / 4) + (J / 4) - (2× J)] % 7

*/

UINT weekday( gmt )
struct utc *gmt;
{
 UINT  mth, year, cent;
 register UINT temp;

 year = gmt->year;
 mth  = gmt->month;

 if( mth < 3 )
 {
  mth += 12;
  --year;
 }
 cent = year / 100;  /* 19th, 20th, or 21th etc century */
 year %= 100;   /* Tens of years (00->99) */

 temp  = gmt->day;   /* Start with the day of the month */

 temp += (((mth + 1) * 26) / 10); /* Advance to the start of the month */

 temp += year;  /* [K]   Add in the year */
 temp += (year / 4); /* [K/4] Correct for leap years */

/* Because of the "% 7" term, -(2*J), and +(5*J) give the same answer: */

 temp += (cent * 5); /* [J*5]  Correct for centuries  */
 temp += (cent / 4); /* [J/4] Give extra day ever 400 years */

 temp %= 7;  /* 7 days in a week */

 if( !temp )  /* Wrap Saturday to be the last day of week */
  temp = 7;

 temp -= 1;  /* 0 = Sunday...6=Saturday */
 gmt->wday = temp;

 return( temp );
}
```

I've had some experience using real-time clocks in an industrial environment.I used the clock as an hour meter, which is easy to do by recording the power on and power off times (keep saving the time each second and read what you saved when power comes back on). You'll find some code in the TYME.ZIP that will help with this, as well.

Because I knew what the time was, I would display it on the system display if there where no important messages everyone complained the clock but,was always wrong— it gained several minutes per month. It is hard to design a– accurate RTC, but Dallas Semiconductor made it easier with- the new *TCXO: DS32KHz Temperature Compensated Crystal Oscillator.* Read the Tech Brief number 16 "*Accuracy: What do you really need?*."

Anyway, I told the operators how to set the clock. The operators advanced the time minute by minute,and clock was more reliable, as they where using it to tell when their shift ended.

I gave up and removed the clock from the display.

The moral of the story is make sure your clock is accurate. Your people will be most unhappy about getting shorted a single second of overtime. Make sure the setting of the clock is somehow protected, with a password maybe, so that it is not perpetually the time for overtime.

For the ultimate in accuracy, if you have an Internet, or MODEM connection in your application, check out the *Time and Frequency Division*, which is an operating unit of the Physics Laboratory of the National Institute of Standards and Technology (NIST).

Located in Boulder, Colorado at the NIST Boulder Laboratories, the Time and Frequency Division:

- Maintains the primary frequency standard for the United States
- Develops and operates standards of time and frequency
- Coordinates U. S. T&F standards with other world standards
- Provides time and frequency services for United States clientele
- Performs research in support of improved standards and services
    - gives free source code and programs

One of the compact and easy ways to manipulate Time/Date formats is the Epoch style.An arbitrary base date and time is chosen to represent the beginning of time in Unix (Jan.1, 1970 at 00:00:00 UTC). When properly used as a 32-bit unsigned long, they can represent the calander fromJan.1,1970 to the year 2106. Unfortunately, many implementations treat the 32-bit number as signed long, cutting its usable span of time in half, or around the year 2038. That's the next Y2K crisis to look forward to.

Shown below are scripts for converting to and from Unix Epoch time to the standard human date time.

These scripts are written in the new up-and-coming dialecting language based on Denotational Semantics, Rebol.

Rebol shines in the area of Internet via having all of the standard Internet protocols as a native part of the language. For example, you can read the time from a Small Network Time Protocol (SNTP) sever with one line:

print read daytime://www.host.com

Daytime retrieves the current day and time of the day using the (SNTP).

Maybe one day, Rebol/View will settle all of the controversies concerning Java by making it obsolete. At least that is my hope.

```
REBOL
        Title: "Convert Epoch Time to Date"
        Author: "Ralph Roberts"
        File: %epoch-to-date.r
        Date: 21-Feb-2000
        Purpose: {converts UNIX Epoch time (seconds after 1-1-1970)
                        to current date and time }
        Example: {outputs "Epoch date 951142987 is 21-Feb-2000
                14:38:52 GMT or 9:38:52 Local" }


        epoch: 951505087

        days: divide epoch 86400

        days2: make integer! days
        time: (days - days2) * 24
        hours: make integer! time
        minutes: (time - hours) * 60
        minutes2: make integer! minutes
        seconds: make integer! (minutes - minutes2) * 60
        time2: make time! ((((hours * 60) + minutes2) * 60) + seconds)

print ["Epoch date" epoch "is" 1-Jan-1970 + days2 time2]
print [" GMT or" time2 + now/zone "Local"]

; And to do it the other way around ( i.e, generate an Epoch date:)

REBOL[
 ]

date: now

seconds: ((date - 1-1-1970)* 86400) + (date/time/hour * 3600) +
(date/time/minute * 60) + date/time/second

zone: now/zone

zone: zone/hour

zone: zone * 3600

seconds: seconds - zone ; minus a minus gives plus

print seconds      ; Epoch date
```

Ref #1:

From Webster's New Universal Unabridged Dictionary 1983 edition:

Gregorian Calendar: A corrected form of the Julian calendar, introduced by Pope Gregory XIII in 1582 used now in most countries of the world. It provides for an ordinary year of 365 days and a leap year of 366 days every fourth even year, exclusive of century years, which are leap years only if exactly divisible by 400.

The century years of 1600, 2000, 2400, 2800, and so on are leap years. Century years of 1700, 1800, 1900, 2100, 2200, and so on are not leap years.

For Internet history buffs:

Network Working Group   Jerry Cole, UCLA
Request for Comments: 32   February 5, 1970
SOME THOUGHTS ON SRI'S PROPOSED REAL-TIME CLOCK
Re: NWG/RFC's #28 and 29.

If you've been a regular reader of my Resource Pages over the last year can't you know I pass up an opportunity to put in something about time rate control theory (time machine) on a page that deals with clocks.

POSSIBILITY OF EXPERIMENTAL STUDY OF PROPERTIES OF TIME

[Unpublished article by N. A. Kozyrev: English title as above; Pulkovo, "O VOZMOZHNOSTI EKSPERIMENTAL'NGO ISSLEDOVANIYA SVOYSTV VREMENI", Russian, September, 1967, pp. 1–49.]

I've been told some of the English translations of his classic "POSSIBILITY OF EXPERIMENTAL STUDY OF PROPERTIES OF TIME" paper, union was originally translated by our Commerce Department in 1969, are not exactly correct.

If you would like to add any information on this topic or request a specific topic to be covered, contact Bob Paddock.
