BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Tomul LIX (LXIII), Fasc. 3, 2013
Secţia
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ

MONITORING THE DS1307 REAL TIME CLOCK USING AN ATMEL FAMILY MICROCONTROLLER

BY

PETRUŢ DUMA[*]

“Gheorghe Asachi” Technical University of Iaşi
Faculty of Electronics, Telecommunications and Information Technology

Received: June 17, 2012
Accepted for publication: June 28, 2013

Abstract. The paper describes the hardware structure required to implement a real-time clock using the DS1307 circuit, monitored by a development system equipped with an ATMEL family microcontroller. The real-time clock operates continually, powered by a battery backup supply. When the main power supply is “on”, the clock is displayed on an LCD. This setup allows the user processes to operate in real-time, preserving the time data if the main power supply fails. The developed program features a series of commands for initializing, displaying and managing the clock, but also for managing the available non-volatile memory.

Key words: real time clock; non-volatile random access memory; ATMEL microcontroller; hardware interface; command program.

1. Introduction

Managing various time parameters and also signal acquiring from sensors used in applications require a real-time clock, usually implemented through software. Therefore, interrupting the microcontroller’s power supply, or blocking the process for whatever reason, leads to the loss of time information that represents an important drawback when restarting the application. In order to eliminate this downside, it is used a real-time clock integrated circuit,

[*]*e-mail*: pduma@etti.tuiasi.ro

permanently powered from a battery for time counting and from the application's power supply unit for command and control.

The process, along with the real-time clock, is commanded and controlled by an application system equipped with an ATMEL family microcontroller. Fig. 1 shows the basic structure of a real-time process used for monitoring various sensors (temperature, humidity, pressure, etc.) and any other user parameters. The notes have the following meaning: PS – pressure sensor, PSI – pressure sensor interface, HS – humidity sensor, HIS – humidity sensor interface, TS – temperature sensor, TSI – temperature sensor interface, µC_AS – microcontroller equipped application system, RTC – real-time clock, DDC – data display console (LCD), SFM – serial FLASH memory, SI_RS232 – serial interface RS232, PC – personal computer.

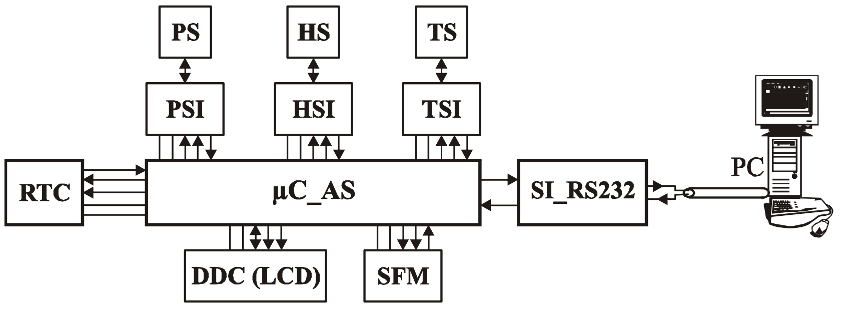

Fig. 1

The serial FLASH memory stores the values of the measured properties at certain scheduled time intervals, their extreme values and the value of the real-time clock. The data display console is optional and continually displays on an LCD the real-time clock and the measured values.

The present paper covers the real-time clock management. Further papers will cover sensor monitoring, various signals acquisition, data saving into the serial FLASH memory or data downloading onto a PC.

2. The DS1307 Real-Time Clock

The DS1307 circuit is a low power clock/calendar that, besides the time monitoring hardware also includes 56 Bytes of non-volatile static RAM (NVRAM). Time counting using this circuit provides decimal information concerning the second, minute, hour, day of week, date, month and year. The circuit command, addressing, data reading and writing are performed through an I^2C serial interface.

The internal block diagram of the real-time clock is presented in Fig. 2. The acronyms signify: Osc – oscillator; Mux – multiplexer; OB – output buffer; PC – power control block; CL – logical control block; SBI – serial bus interface (I^2C); AR – address register; AD – address decoder; UB – user buffer for reading the real-time clock; SD – seconds divisor; MD – minutes divisor; HD – hours divisor; DD – day-of-week divisor; DD* – date-of-month divisor; MD* – month divisor; YD – year divisor; CR – control register; NVRAM – non-volatile random access memory.

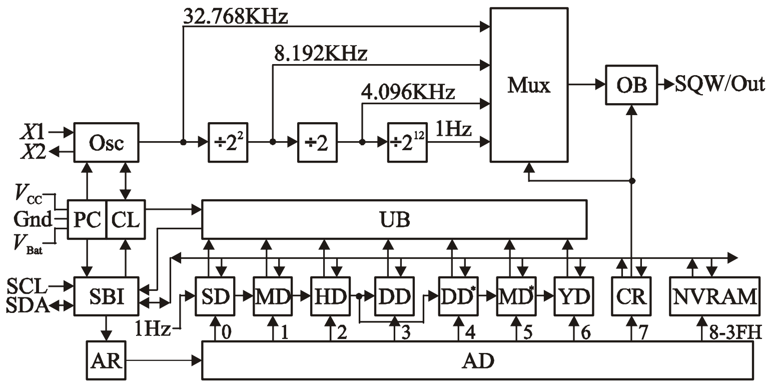

Fig. 2

The circuit includes an internal clock oscillator, driven by a quartz crystal (32.768 KHz) followed by a 2^{15} divisor that provides an output signal with a period of one second.

The internal digital multiplexer has input signals with different frequencies, and, controlled by the control register, provides one of these signals through an output buffer at the SQW/Out output. The control register also manages the output buffer in order to provide the programmable logical levels.

The one-second period signal commands the serial counter chain that provides seconds (a divisor by 60), minutes (divisor by 60) and hours (divisor by 12 or 24). The one day period signal from the hour divisor output, commands the counter for the day of week (divisor by 7), but also the serial counter chain that delivers the day of the month (divisor by 28…31), the month of the year (divisor by 12) and for the year (divisor by 100).

The real-time clock counters, the control register and the memory locations are addressed by a decoder (as shown in Table 1). The memory locations with decimal divisors and the clock command indicators are described below.

The memory location at address 00H stores in the least significant 7 bits the second divisor and in the most significant bit the clock halt indicator CH. When this indicator is set to logical “0”, the oscillator is active, and it is stopped when it is set to logical “1”.

The memory location at address 01H stores in the least significant 7 bits the minutes divisor.

The memory location at address 02H stores in the least significant 6/5 bits the hour divisor. The day can be counted either using 24 hours or 12 hours and an AM/PM indicator. The hours selection indicator HS stored in the 6th bit of the said memory location sets the day monitoring style: 24 hours if HS = 0 and 12 hours if HS = 1. In the latter case, bit 5 of the memory location stores the AM/PM indicator, logical “0” for AM and “1” for PM.

The memory location at address 03H includes in the least significant 3 bits the day of week counter. The numeric values for the days of the week are sequential and have been defined in the command software using text messages as in Table 2.

Table 1

Address	bit$_7$	bit$_6$	bit$_5$	bit$_4$	bit$_3$	bit$_2$	bit$_1$	bit$_0$	Function
00H	CH	**00…59 Seconds**							Seconds Counter
01H	0	**00…59 Minutes**							Minutes Counter
02H	0	0	**00…23 Hours**						Hours Counter
		1	0	**1…12 Hours AM**					
		1	1	**1…12 Hours PM**					
03H	0	0	0	0	0	**1-7 Day**			Day Counter
04H	0	0	**1…28/29/30/31 Data**						Data Counter
05H	0	0	0	**1…12 Month**					Month Counter
06H	**00…99 Year**								Year Counter
07H	**Out**	0	0	**SQWE**	0	0	**RS1**	**RS0**	Control Register
08H-3FH	**00…FFH User Data**								NVRAM Memory

Table 2

Numeric values	Days of the week
1	Monday
2	Tuesday
3	Wednesday
4	Thursday
5	Friday
6	Saturday
7	Sunday

The memory location at address 04H includes in the least significant 6 bits the day of month counter. This counter is automatically adjusted through the hardware structure at the end of the month if the month is less than 31 days long, including the leap years.

The memory location at address 05H includes in the least significant 5 bits the month of the year counter. The command program will display on the console the numeric values 1, 2,…,12 for the months of January, February and so forth using text messages.

The memory location at address 06H includes in the 8 bits the hundred years counter; the required compensations for the leap years until 2100 are performed internally.

The memory location at address 07H includes the control register consisting of four indicators used to command the output SQW/Out. The square wave enable SQWE indicator, when set to logical “1”, produces a square wave at output SQW/Out. In this case, the frequency of the signal is set by a multiplexer that is addressed by the rate selection registers RS1, RS0 according to Table 3. When SQWE is set to logical “0”, the output SQW/Out transmits the logical level from the output control indicator Out that is commanded by software.

Table 3

Control Register				SQW/Out Output
Out	SQWE	RS1	RS0	
–	1	0	0	1 Hz signal
–	1	0	1	4.096 KHz signal
–	1	1	0	8.192 KHz signal
–	1	1	1	32.768 KHz signal
0	0	–	–	“0” logical level
1	0	–	–	“1” logical level

The memory locations from address 08H up to 3FH are assigned to store various byte size user data.

3. Interfacing the Real-Time Clock

The clock oscillator included in DS1307 requires a quartz crystal connected to input *X*1, while the output *X*2 with nominal frequency of 32.768 KHz, must be connected to a typical load capacitor of 12.5 pF and a maximal serial resistor of 45 KΩ (Fig. 3). The Oscillator requires no other external component for its operation and has a startup time of less than a second.

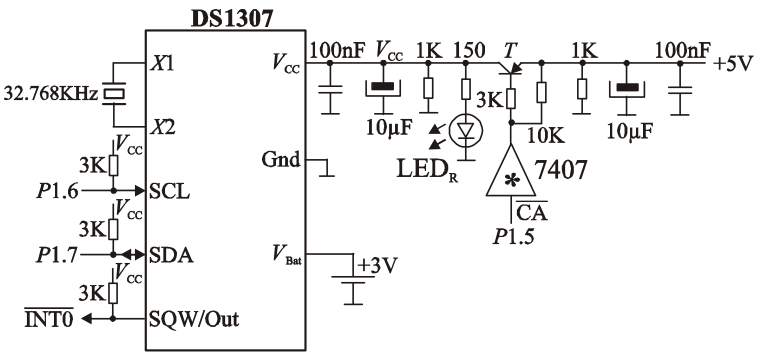

Fig. 3

The DS1307 circuit is powered from a primary DC power source of +5 V connected to V_{cc} and Gnd which is also used to power the user process. If this voltage is within normal limits then the time counting, the internal memory and the serial interface, I^{2}C, circuits are powered. In this case, the maximal current input is no more than 1.5 mA. The circuit is also powered by a backup supply consisting of a standard Lithium battery with 3 V nominal voltage connected to V_{Bat} and Gnd. The battery voltage must be within normal operating limits, starting from 2 V up to 3.5 V. If no backup power supply is used, than terminal V_{Bat} is connected to the ground. The power supply control block includes detection circuits for various threshold voltages and switching circuits of the power supply from the main supply to the backup one and back.

When the main power supply voltage, V_{cc} , drops below $1.25V_{\mathrm{Bat}}$, then the on-going access through I^2C interface is completed, the internal address counter is reset and no more data is read or written in order to prevent faulty operations. When V_{cc} drops even below V_{Bat}, then the power supply is switched to the backup supply that only sustains time counting and memory circuits. The current absorbed from the backup supply is of 300 nA if output SQW/Out is unused, or 480 nA if output SQW/Out delivers 32.768 KHz frequency signal. As V_{cc} surges above V_{Bat} + 0.2 V, the power supply is switched back to the main supply and, finally, as V_{cc} rises above $1.25V_{\mathrm{Bat}}$, the communication interface I^2C is re-activated. In Fig. 3, the +5 V main supply is switched through software by transistor *T*, under the control of *P*1.5, through an open collector buffer in order to perform real-time clock power supply from V_{cc}. The backup power supply is a Lithium battery that insures an operation of the DS1307 circuit for more than 10 years in the absence of the main power supply.

The SQW/Out output is open-drain, thus requiring an external pull-up resistor, and generates a square wave signal of a certain frequency or it can be commanded through software. It is used to apply 1 KHz frequency interrupt requests to the microcontroller at input $\overline{\mathrm{INT0}}$.

The real-time clock only communicates to a slave circuit through the I^2C serial interface. This one has a clock input line (SCL) and a bidirectional data line (SDA). The clock signal is always delivered by the master microcontroller at pin *P*1.6 and works at a standard maximal frequency of 100 KHz. The bi-directional open-drain data line is connected to pin *P*1.7 of the microcontroller and requires an external resistor connected to V_{cc} for data transmission.

4. Monitoring the Real-Time Clock

The master microcontroller only initiates a data transfer if the bus is available and controls the access to the interface I^2C bus. In order to perform a data transfer, the START condition is generated and the slave identification address and the operation mode identifier are sent. Afterwards, the memory location address is sent, either one or several data bytes are sent in order to be written into the slave or data bytes are received from the slave and finally the STOP condition is generated.

In order to initialize data transfer on the bus, the clock and data lines must be set to logical “1”. Any data transfer begins with the transmission of the START condition; this consists of sending on the free bus lines a transition from 1 to 0, while the clock line is maintained in logical “1”. Ending the data transfer is made by sending the STOP condition; in this case, a transition from logical “0” to logical “1” is sent on the data line, while maintaining the clock line in logical “1” (Fig. 4).

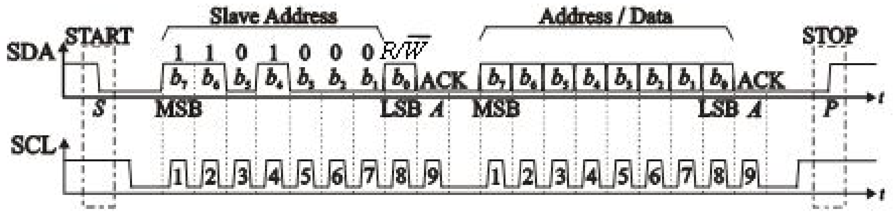

Fig. 4

On the data line, data bits are sent starting with the most significant bit and ending with the least significant one. The transmitted data bits are changed only during the logical “0” level of the clock signal, while during the logical “1”, they must remain stable. The changes appearing on the data line while the clock line is in logical “1” are interpreted as control signals. The data is transferred bit by bit, and every other eighth sent data bit, the receiver confirms the received byte with an acknowledgement bit ACK, defined by a logical “0” during the ninth clock period from the master.

After the START condition, the master sends an identification byte consisting of the 7-bit address of the slave circuit and a bit with the $R/\overline{W}$ indicator that selects the operation mode of the slave circuit. The DS1307 real-time clock has the address of 1101000, while the indicator $R/\overline{W}$ specifies the direction of the data transfer. If $R/\overline{W} = 0$, the slave receiver mode is selected, while for $R/\overline{W} \neq 0$ the slave transmitter mode is set.

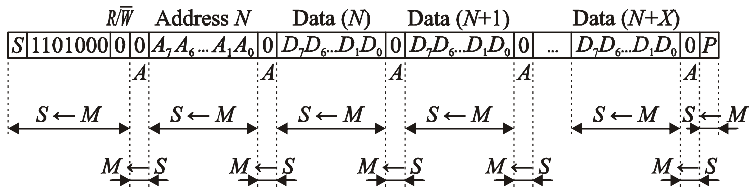
R/W̄
Address N
Data (N)
Data (N+1)
Data (N+X)
S 1101000 0 0 A7A6...A1A0 0 D7D6...D1D0 0 D7D6...D1D0 0 ... D7D6...D1D0 0 P
A
S←M
M←S

Fig. 5

In the slave receiver mode, a data transfer occurs for writing a data block coming from the master transmitter towards the slave receiver. Fig. 5 presents graphically a data transfer performed in this manner, which consists of the following steps: the master begins by transmitting the START condition (noted *S*); the master sends the identification byte 1101000 consisting of the slave address and $R/\overline{W} = 0$; the slave circuit is identified and it transits the acknowledgement bit ACK (noted *A*); the master sends the byte with the memory address location ($A_7A_6 \ldots A_1A_0$); the slave acknowledges the received address (ACK) and loads it in the address register of the real-time clock; the master sends the data byte ($D_7D_6 \ldots D_1D_0$) that is to be written in the memory location; the slave acknowledges the received data (ACK), writes it in the memory location with the specified address and increments by a unit the address register; the last two steps are repeated for each data that is written in the slave in memory location at consecutive addresses; in the end, the master sends the STOP condition (noted *P*).

In the slave transmitter mode, a data transfer is enabled for the reading data block from the slave transmitter to the master receiver. The graphical representation of this data transfer is illustrated in Fig. 6, the steps having the following sequence: the master begins with the transmission of the START condition; the master sends the identification byte 1101000 & $R/\overline{W}=1$; the slave circuit is identified, which sends the acknowledgement bit ACK; the identified slave circuit sends the data byte ($D_7D_6 \ldots D_1D_0$) from the memory location with the address specified in the address register, that is to be read by the master, then the address register is incremented with one unit; the master recognizes the received data and sends the acknowledgement bit ACK except for the last byte of data; the last two steps are repeated for every data read from the slave, from memory locations with consecutive addresses; for the last received data byte, the master sends a non-acknowledgement data bit ACK with logical level "1" (noted $\overline{A}$); in the end, the master sends the STOP condition.

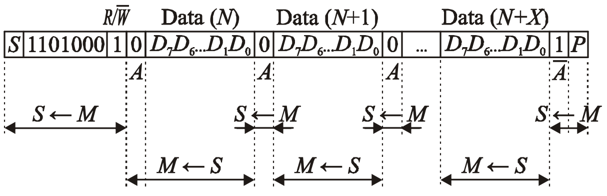

Fig. 6

The described slave transmitter mode has the downside that reading the data block starts from the address contained in the address register. This address cannot be specified by the operation mode command in order to allow it to be set to any numeric value.

The receiver and transmitter slave operating mode discards this drawback and merges parts from the two modes previously presented. The first five stages of the slave receiver mode load the address register of the real-time clock with a certain numeric value, then all the stages of the slave transmitter mode follow, beginning with the retransmission of the START command (noted *R*), followed by the others in order to read the data block (Fig. 7). Thus, the reading of the data block from slave by the master is possible from any address.

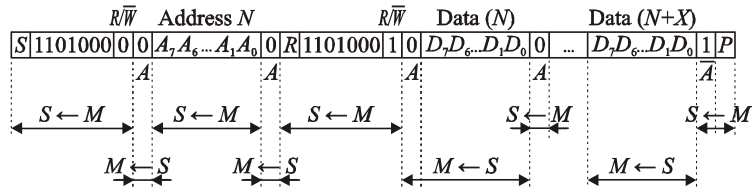

Fig. 7.

The command program consists of a segment of initializations which load the application variables with the necessary numeric values, starts and deletes the liquid crystal display (LCD) and activates the interrupt system for the interrupt requests applied every second *via* the $\overline{\text{INT0}}$ line. This program also verifies if a serial console is connected to the microcontroller, along with other necessary initializations.

If a serial console is connected to the microcontroller, the console display is erased. Then, after the display of a launch message, the base program is run, that consists of a keyboard interrogation loop from which various user commands are sent. The significance of the main commands, which are implemented through software in order to manage the real-time clock and non-volatile memory, is briefly described below.

A. – The command supplies the real-time clock with DC voltage (+5 V). This has the effect of connecting/disconnecting the primary power supply. The RETURN (.) key is used for closing all commands.

D. – The command displays on the console the contents of the non-volatile memory (from address 00H to address 3FH). The address is displayed on every row, then the contents of 16 memory locations in hexadecimal.

S p1_ – The command shows and/or substitutes the contents of the non-volatile memory location at address p1. Using the BLANK (_) separator displays and/or substitutes the contents of the next memory location, whilst the use of the COMMA (,) separator displays and/or substitutes the contents of the previous memory location.

SS. – The command displays and/or substitutes seconds.

SM. – The command displays and/or substitutes minutes.

SH. – The command displays and/or substitutes the hours.

SZ. – The command displays and/or substitutes the day of the week.

SD. – The command displays and/or substitutes the date of the month.

SL. – The command displays and/or substitutes the month of the year.

SY. – The command displays and/or substitutes the year.

SI. – The command displays and/or substitutes the control register indicators: SQW/Out – square-wave enable indicator, RS1 and RS0 – the rate selection indicators, Out – the exit control indicator, followed by the clock command indicators: CH – the clock halt indicator, HS – the hour selection indicator, AM/PM – the indicator for antemeridian and postmeridian.

This command displays on every row a text message with the indicator name followed by the binary numeric value of the corresponding indicator, then the input of the new binary value for its substitution is awaited (if desired).

C. – The command displays the clock on the serial console. Every second, on one row of the system console, are displayed the current hour, minute and second, then the day of the week as text, the date, the month of the year as text and the year. The command is exited using the CTRL+P key sequence.

R. – The command resets the non-volatile memory area from address 08H to address 3FH (the immediate data 00H is loaded in every location).

Q. – The command quits the base program.

The interrupt request, sent by the real-time clock every second, displays on the LCD the day of week as text for five seconds, then the date, month and year for another five seconds, then the current time – hour, minute and second, updated on every subsequent interrupt request for ten seconds, and finally all this data is redisplayed in the same sequence.

The command software used to manage the real-time clock is based on a series of elementary subroutines that perform certain functions with this circuit, briefly described as follows: GSTART and GSTOP – generate the START and STOP conditions, respectively; TxARxS – transmits the address of the slave circuit and sets the slave receiver operation mode; TxATxS – transmits the address of the slave circuit and sets the slave transmitter operation mode; RxA – receives the acknowledgement bit from the slave receiver; TxA – sends the acknowledgement bit to the slave transmitter; TxNA – sends the non-acknowledgement bit to the slave transmitter after reading the last byte of data from it; TxAdr – transmits the address of the memory location that is to be accessed by the slave circuit; TxDat – transmits to the slave receiver the data byte that is to be written; RxDat – receives from the slave transmitter the read data byte.

Using these subroutines, two base subroutines are developed that perform: WRBD – writing a data block into the slave circuit memory starting from a specified memory location address and RDBD – reading a data block from the slave circuit memory starting from a specified memory location address.

5. Conclusions

The described hardware structure was built in practice. It is simple and consists of the DS1307 real-time clock, requiring a minimal amount of external components, and a development system equipped with microcontroller AT89S8253. This structure is connected to several digital sensors for temperature, pressure, humidity, etc., and a serial FLASH memory that allows to perform and to store real-time measurements.

The hardware real-time clock added to the structure of various user processes guarantees a continuous operation for at least 10 years, based on its battery power supply, regardless if the main power supply is “on” or “off”.

The command program displays permanently the real-time clock on the LCD, while the base program implements a series of commands for initializing, displaying and monitoring it from a serial console. Other user commands allow to display and substitute the non-volatile memory in various ways. A program sequence performs the automatic adjustment of the clock settings for daylight saving. This command program uses a 2.7 Kbytes of program memory, a remarkably low amount of memory compared to its features and possibilities.

The program can be improved with a routine that compensates for the time drift produced by the quartz crystal tolerance and by the temperature variations.

REFERENCES

Duma P., *Microcontrolerul INTEL 8051. Aplicaţii*. Edit. „TEHNOPRESS”, Iaşi, 2004.

Hintz J.K., Tabak D., *Microcontrollers. Arhitecture, Implementation and Programming.* McGraw Hill, New York, 1993.

Lance A. *Leventhal, Programmation en langage assembleur.* Edit. Radio, Paris, 1998.

Peatmann B.J., *Design with Microcontrollers*. McGraw Hill, New York, 1998.

* * * *Family Microcontroller*. ATMEL, Data Book, 1998.

* * * *Semiconductor, DS1307 Real-Time Clock*. Dallas, Data Sheet, 2006.

* * * *Semiconductor, Using a DS1307 with a PIC Microcontroller*. Dallas, Application Note, 2006.

* * * *Semiconductor, Selecting a Backup Source for Real-Time Clocks*. Dallas, Application Note, 2006.

* * * *Semiconductor, Oscillator Design Considerations for Low-Current Appli-cations.* Dallas, Application Note, 2011.

* * * *Semiconductor, State Machine Logic in Binary-Coded Decimal (BCD)-Formatted Real-Time Clocks*. Dallas, Application Note, 2012.

* * * *Multichannel RS232 Drivers/Receivers*. MAXIM, MAX232A Data Sheet, 2006.

MONITORIZAREA CEASULUI DE TIMP REAL DS1307 CU MICROCONTROLER DIN FAMILIA ATMEL

(Rezumat)

Se descrie structura hardware necesară pentru implementarea unui ceas de timp real cu DS1307, iar pentru monitorizarea acestuia se foloseşte un sistem de dezvoltare echipat cu microcontroler din familia ATMEL. Ceasul de timp real funcţionează continuu fiind alimentat de la o sursă de rezervă cu baterie, iar când este conectată şi sursa de tensiune principală se afişează ceasul pe un LCD. Acest mod permite proceselor utilizator să lucreze în timp real, iar întreruperea sursei principale de alimentare să nu ducă la pierderea informaţiilor de timp. Programul de comandă dezvoltat asigură o serie de comenzi pentru iniţializare, afişare şi gestionarea ceasului, dar şi pentru gestionarea memoriei nevolatile de care dispune.