BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Volumul 66 (70), Numărul 2, 2020
Secţia
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ

MONITORING DS18B20 TEMPERATURE SENSORS USING AN 1-WIRE® NETWORK

BY

PETRUŢ DUMA*

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

Received: November 9, 2020
Accepted for publication: December 20, 2020

Abstract. The paper describes the hardware interface required for the temperature measuring using the DS18B20 digital sensor connected in an 1-wire® network, controlled by a development system equipped with an ATMEL microcontroller. The temperature sensors' network is managed using a personal computer connected to the microcontroller's UART serial interface through an USB-RS232 converter. The command program measures the temperature from the sensors in the 1-wire® network, displays it in several formats, performs various digital processings for the acquired data and executes different user commands.

Keywords: 1-wire®temperature sensors network, 1-wire® communication protocol, multi-drop capability, device search, identification serial code, parasite power mode.

1. Introduction

The sensor is a circuit that measures a certain physical property and converts it to an electric signal that can be processed, acquired, stored and displayed by a system (Baltag, 2003; Chita, 2017). Nowadays, mostly digital sensors are used, that include in the same integrated circuit the required

*Corresponding author; *e-mail*: pduma@etti.tuiasi.ro

transducer followed by the analogue amplification and processing circuit, the analog-digital converter, the voltage reference, the compensation circuit, the low-consumption circuit, the serial interface for data transmission and reception, the command and control circuit for the whole system and more (Ciobanu, 2006; Ivanov, 2018).

A monitoring system for DS18B20 temperature sensors communicating in an 1-wire® network, along with other sensors, memory chips or peripherals, has the principle structure shown in Fig.1. The notes used have the following meanings: TS_1, TS_2, … TS_n – temperature sensors, PS – pressure sensor, HS – humidity sensor, OS – other sensors, RTC – real time clock, LCD – liquid crystal display, M_1 – volatile memory (SRAM), M_2 – non-volatile memory (EEPROM), 1-W_SI – 1-wire® serial interface, µC_AS – microcontroller application system, SI – RS232 serial interface, C – USB-RS232 converter, PC – personal computer.

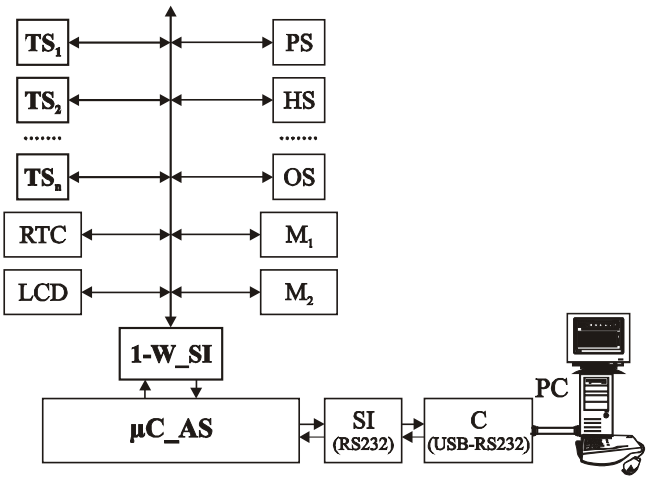

Fig. 1 – Sensors monitoring system in a 1-wire® network.

This work covers monitoring the DS18B20 temperature sensors connected in an 1-wire® network, further papers being envisioned to present other resources (different sensors, memory chips, peripherals) using 1-wire® communication or to implement different devices using this protocol.

2. The DS18B20 Digital Temperature Sensor

The DS18B20 sensor from Maxim/Dallas Semiconductor measures the ambient temperature in Celsius degrees, is completely digital and does not require external components. Other characteristics of this sensor are: measured temperature ranges between -55°C and +125°C, with an user selectable

resolution from 9 to 12 bits; ±0.5°C accuracy for the range between -10°C and +85°C and ±2°C accuracy in the rest of the measuring range; the conversion time at the maximal resolution (12 bits) is 750 ms; power range between +3V and +5.5V, but the device can also use parasite power mode using the data line; communicates through a serial 1-wire® interface (no clock signal); connects in a 1-wire® network, with a master microcontroller and several slave devices; multi-drop connection simplifies many temperature measuring applications; each device has an unique 64 bits identification serial number, stored in a non-volatile memory for searching devices in the network; contains a temperature regulator with software-selectable thresholds; a temperature alerting system searches and identifies the sensors that measured a value outside the programmed intervals (Maxim, 2019).

The internal structure of the DS18B20 temperature sensor is detailed in Fig. 2; the notes have the following signification: S/PPC – standard or parasite power circuit; PSD – power source detection; 1-W_IS – 1-wire® serial interface; B – data reception (Rx) buffer; Q – data transmission (Tx) MOSFET transistor; I – current source (5µA); D_CCL – device command and control logic; M_CCL – memory command and control logic; SM – scratchpad memory (SRAM); ADC – 12-bit analog-digital converter; A – amplifier; TS – temperature sensor; HTAR – high threshold alarm register (T_H); LTAR – low threshold alarm register (T_L); CR – configuration register; CRC_G – CRC generator; C_{PP} – internal capacitor; R_{PU} – pull-up resistor; V_{PU} – parasite power voltage; DQ – data communication line (Maxim, 2019).

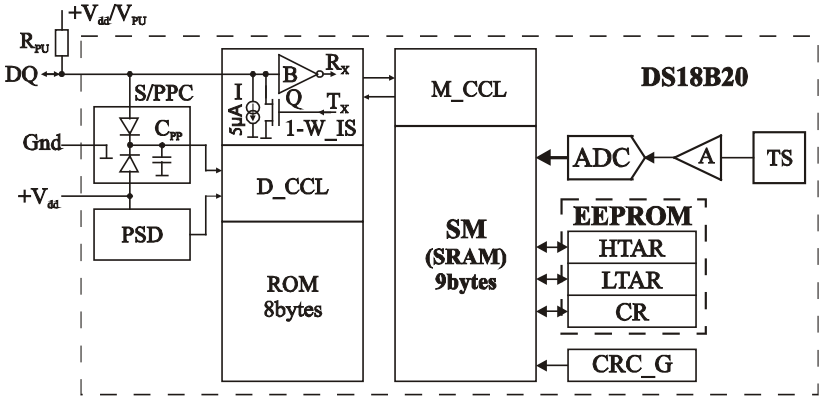

Fig. 2 – Block diagram of the DS18B20 temperature sensor.

The signal from the temperature sensor is amplified, compensated, converted to digital and loaded into the 16bit temperature register (HTR, LTR); the measured temperature is stored in Celsius degrees and is a signed 2's complement binary number.

The structure of the temperature register for a 12-bit resolution is shown in Fig.3, where: S – measured temperature sign (0 - positive, 1 - negative); $T^6T^5T^4T^3T^2T^1T^0$ – the whole number part of the temperature; $T^{-1}T^{-2}T^{-3}T^{-4}$ – the fractional part (Maxim, 2019).

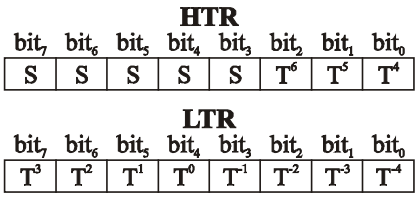

Fig. 3 – Temperature register.

Table 1 presents the relationship between the temperature and the output data from the temperature register (in binary and hexadecimal), when the sensor uses a 12-bit resolution for measuring.

Table 1

Relationship between the Temperature and the Output Data

Temperature	Digital Output Binary				Digital Output Hexadecimal
$+125^0$C	0000	0111	1101	0000	07D0H
$+85^0$C	0000	0101	0101	0000	0550H
$+43.625^0$C	0000	0010	1011	1010	02BAH
$+22.25^0$C	0000	0001	0110	0100	0164H
$+0.5^0$C	0000	0000	0000	1000	0008H
$+0.125^0$C	0000	0000	0000	0010	0002H
$+0.0625^0$C	0000	0000	0000	0001	0001H
0^0C	0000	0000	0000	0000	0000H
-0.0625^0C	1111	1111	1111	1111	FFFFH
-0.75^0C	1111	1111	1111	0100	FFF4H
-25.3125^0C	1111	1110	0110	1011	FE6BH
-55^0C	1111	1100	1001	0000	FC90H

The DS18B20 sensor memory contains a ROM memory (8 bytes), a scratchpad memory (9 bytes) and an EEPROM memory (3 bytes).The device's ROM memory contains an unique identification code (Fig. 4), where: FC – family code (28H for Maxim/Dallas Semiconductor circuit); SN – serial number for individual circuit identification; CRC8 - cyclic redundancy check code. The less significant 8 bits from the ROM include the family code, the next 48 bits include the unique serial number and the most significant 8 bits, the CRC code computed for FC and SN (Maxim, 2019).

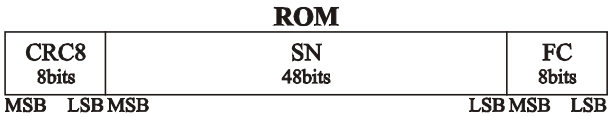

Fig. 4 – ROM structure.

The scratchpad memory is a fast volatile memory (SRAM) that has the possibility of storing certain registers in a non-volatile memory (EEPROM). The structure of this memory is shown in Fig. 5, where: HTR - high temperature register; LTR - low temperature register; HTAR& LTAR - register storing the high and the low temperature alarm thresholds, respectively; CR - configuration register; R - inaccessible memory locations reserved for internal operations; CRC_G - CRC generator. HTAR, LTAR& CR registers are to be found both in the scratchpad memory (noted with S exponent) and the EEPROM memory (E exponent); on device power-on, the contents of the EEPROM registers is automatically loaded in their scratchpad counterparts (Maxim, 2019).

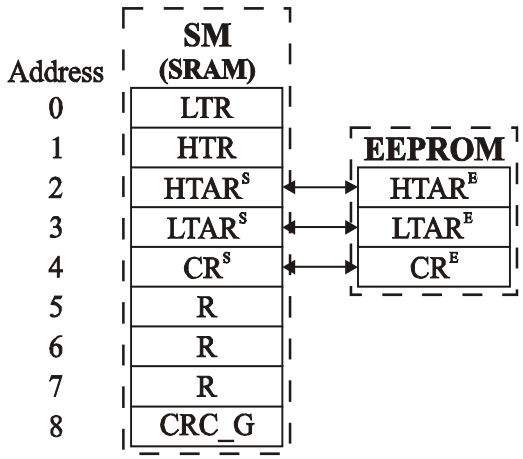

Fig. 5 – Scratchpad & EEPROM structure.

After every temperature conversion, the measured value (only the signed whole number part) is compared to the alarm thresholds set by the user. These thresholds from the HTAR & LTAR registers have each only a byte containing the whole number part and are represented in signed 2's complement. The structure of these registers is shown in Fig. 6; the most significant bit indicates the sign (S), while the least significant seven bits indicate the whole number part of the respective threshold (T_H - high temperature threshold; T_L - low temperature threshold).

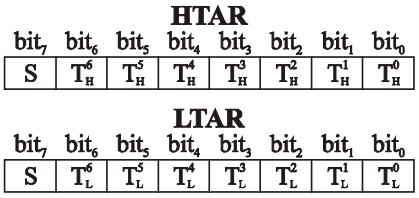

Fig. 6 – Thresholds alarm registers.

If the measured temperature is lower or equal than the value stored in LTAR or if it is higher or equal than the value in HTAR, then an internal alarm flag in the device is set. The alarm flag is automatically updated after each temperature measuring operation and the master can identify the devices in the network that are in this alert state. If the temperature alarm function is not used, then the thresholds alarm registers can be used as general use memory locations.

The structure of the configuration register is shown in Fig. 7 (where R_1 and R_0 are configuration indicators).The temperature sensor has a resolution from 9 to 12 bits and a conversion time from 93.75 to 750 ms (Table 2). The default values of these indicators on power-on are R_1=1, R_0=1 and correspond to a 12-bit resolution (t_{CONV}= 750 ms). The rest of the bits in the configuration register are reserved for internal operations and are not usable (Maxim, 2019).

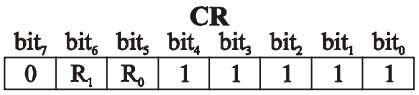

Fig. 7 – Configuration register.

Table 2

Resolution & Conversion Time of the DS18B20 Temperature Sensor

R_1	R_0	Resolution	t_{CONV}	
0	0	9 bits	93.75ms	$t_{CONV}/8$
0	1	10 bits	187.5ms	$t_{CONV}/4$
1	0	11 bits	375ms	$t_{CONV}/2$
1	1	12 bits	750ms	t_{CONV}

The CRC bytes are provided by the device when the serial number from the ROM or the data from the scratchpad are read. The microcontroller calculates the CRC for the data received from the sensor and then it compares the calculated CRC to the received CRC. If the two values are equal, then the data received is errorless; otherwise, the data received is erroneous and the read command is repeated. The CRC generator of the sensor is implemented with the polynomial function $g(x)=X^8 \oplus X^5 \oplus X^4 \oplus 1$ (Maxim, 2019).

The temperature sensor features a parasite power mode, when the power source V_{dd} is missing. In this case, the device is powered through resistor R_{PU} from V_{PU} when DQ is 1 (capacitor C_{PP} is loading) or from C_{PP} when DQ is 0 (Fig. 2). This operation mode provides the sensor with the energy required to perform most of the commands, but some of the commands that have higher energy consumption (temperature conversion or EEPROM writing) require switching the data line DQ to V_{PU} with a transistor during the execution of respective command. In the present application, the DS18B20 sensors are powered from an external DC source, in which case the 1-wire bus can be used during performing the interrogation command.

3. Interfacing DS18B20 Temperature Sensors

The command and control of the DS18B20 temperature sensors used for checking, testing and developing the application, are made with a development system equipped with an ATMEL microcontroller (ATMEL, 2005). The interface for sensor monitoring is simple, consisting of a controlled DC voltage source and open drain buffers for sense separation and level translation (Fig. 8) (Aghion and Ursaru, 2015; Duma, 2007; Petreus, 2005).

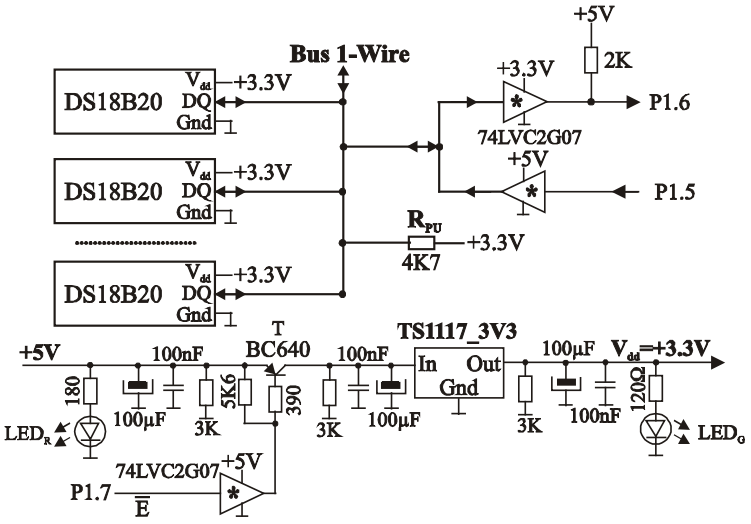

Fig. 8 – Interface to the DS18B20 sensors network.

The temperature sensor is powered within a DC voltage range from +3V to 5.5V. The application is designed to accommodate the connection of further 1-wire® devices, therefore the implemented interface powering all the circuits at +3.3V voltage. For the power source, a TS1117_3V3 integrated three-points voltage stabilizer is used and requires only the connection of some

filtering capacitors (Taiwan Semiconductor, 2003). This source is software controlled through the open drain buffer (74LVC2G07) and the transistor T. The enabling system of the power source $\overline{E}$ is active on logical 0 and is provided by line P1.7 of the microcontroller; when the sensors network is not powered, $\overline{E}$ is inactive (logical 1). After powering on the sensors, a delta of a few hundreds of milliseconds is introduced, in order to allow internal circuits initialization, the data line of the serial interface passes into the high impedance state, internal registers are initialized, EEPROM registers contents is copied into the scratch pad and, in the end, the device enters the standby state. In this state, the sensor receives any command transmitted by the microcontroller, checks its destination, starts the command execution and receives the parameters (if any), transmits the data if the command requires an answer and resumes the standby status. The DS18B20 temperature sensor consumes an operating current of up to 1.5 mA (maximum current at conversion and EEPROM writing), while in standby, the consumed current is up to 1 μA.

The bi-directional serial data line DQ is connected through open drain buffers (74LVC2G07) at line P1.6 for input data and line P1.5 for output data through which ATMEL microcontroller receives data and respectively, sends data (Texas Semiconductor, 2007). These buffers are necessary for data sense separation and for level translation between the development system powered at +5V and the temperature sensors powered at +3.3V.

4. Monitoring DS18B20 Temperature Sensors

The master microcontroller manages the slave temperature sensor through the 1-wire® serial interface. Any device with this serial interface has a buffer (B) on the DQ data line for receiving data and a MOSFET (Q) transistor for transmitting data (Fig. 2). A 0 bit is sent by saturating the transistor, while an 1 bit is transmitted by blocking it (in this latter case, R_{PU} ensures the logical level 1 required for bus inactivity). The commands and data circulating on the bus start with the least significant bit of the byte and end with the most significant one (Maxim, 2002a).

The microcontroller initiates any command for accessing the DS18B20 temperature sensor. This is done in the following sequence of operations: 1. *Initialization*; 2. *ROM memory command* followed by a data transfer, if necessary; 3. *Functional sensor command* followed by a data transfer, if necessary. The operations sequence is very important and must be observed. Otherwise, the device does not respond and does not execute the command. Exceptions to this rule are the commands *Search ROM* and *Alarm Search*, for which the operating sequence consists only of the first two steps.

Any command starts with an initialization consisting of a pulse sent by the microcontroller, followed by the presence pulses sent by the slave devices;

these pulses inform the master that the slave devices are available for use and are awaiting commands. After receiving the presence pulse, the microcontroller can transmit a ROM memory operation command, using the unique identification code (UIC) of the device. These commands allow the use of a single sensor or of all of them, determining the number and the type of the devices in the network or checking if the alarm was triggered (Maxim, 2019).

The *Search ROM command* (code 0F0h) identifies the codes stored in the ROM memory for all the slave devices connected to the 1-wire®, thus determining the number and the types of these circuits. These codes are read from ROM and, through an elimination process consisting of several search cycles, the UIC is determined for every device connected in the network.

The *Read ROM command* (code 033h) reads the unique identification code from the ROM memory only when a single device is connected on the bus.

The *Match ROM command* (code 055h) followed by the UIC of a device has the role to address the following operation command only to the defined slave circuit.

The *Skip ROM command* (code 0CCh) addresses consequently all the devices in the network for the following operation command, which does not require a data response. In this manner, a temperature measurement can be initiated for all the sensors in the network.

The *Alarm Search command* (code 0ECh) determines all the devices in the network (their UIC stored in the ROM) that have an active alarm flag.

After the sensor receives a ROM memory operation command, the microcontroller sends one of the operation commands that perform temperature measurement, reading and writing in the scratchpad memory, data transfer between memory areas or determining the source type powering the sensors.

The *Convert Temperature command* (code 044h) induces the sensor to measure the temperature. The result of converting the temperature into Celsius degrees is a signed 2's complement 16-bit value, stored into the temperature register. After the operation, the sensor passes in standby.

The *Write Scratchpad command* (code 04Eh) writes three data bytes into the high alarm, low alarm and configuration registers in the scratchpad memory.

The *Read Scratchpad command* (code 0BEh) reads the contents of the registers in the scratchpad memory.

The *Copy Scratchpad command* (code 048h) reads the contents of the high alarm, low alarm and configuration registers in the scratchpad memory and writes it in the EEPROM.

The *Recall EEPROM command* (code 0B8h) reads the contents of the registers in the EEPROM and writes them into the corresponding memory locations in the scratchpad.

The *Read Power Supply command* (code 0B4h) reads a time slot from the 1-wire bus to determine whether there are parasite-powered slave devices.

The parasite-powered sensors pull the bus into logical 0, while those powered from an external source keep the bus in logical 1.

The DS18B20 devices use a 1-wire® communication protocol to ensure the integrity of the transmitted/received data. This protocol defines the following operations: initialization pulse, presence pulse, write 0 bit, write 1 bit, read 0 bit and read 1 bit.

The device initialization operation is shown in Fig. 9, where: the thick black line indicates that the master microcontroller sets DQ in 0; the thin black line indicates that resistor R_{PU} pulls DQ in 1; the dotted black line indicates that the slave device sets DQ in 0 (Maxim, 2019).

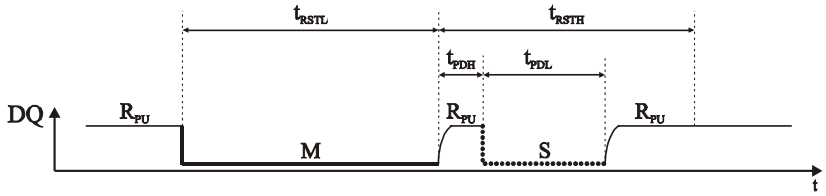

Fig. 9 – Initialization and slave devices presence.

The microcontroller sends the initialization pulse by keeping the 1-wire bus in logical 0 for at least 480µs (t_{RSTL}), then the master releases the bus (R_{PU} pulls DQ in 1) and passes into the data reception mode. The master awaits for the presence pulse for the following time duration of at least 480µs (t_{RSTH}). The slave device detects the rising edge from the initialization pulse sent by the microcontroller and waits for a time interval (t_{PDH}) from 15µs to 60µs. Afterwards, it confirms its presence by sending a 0 level on the bus for a time duration (t_{PDL}) from 60µs to 240µs. In the end, the bus is released and pulled by R_{PU} in 1. Data writing and reading are made in a time slot (t_{SLOT}) with the duration from 60µs to 120µs. Between two consecutive time slots, a recovery time t_{REC} is required of at least 1µs (Fig.10). These time slots are initiated by the microcontroller by transmitting a logical 0 on the 1-wire bus for writing data into the slave, but also for reading from it (Maxim, 2009).

The master writes a 0 bit in a time slot WR0, the operation consisting in transmitting a 0 level on DQ for a time duration from 60µs to 120µs (t_{LOW0}), then the bus is released and pulled by R_{PU} in 1. In a similar manner, the master writes a 1 bit during a time slot WR1 consisting in transmitting a 0 level on DQ for a time duration from 1µs to 15µs (t_{LOW1}), then the released bus is pulled in 1. The DS18B20 temperature sensor samples the bus in a time frame from 15µs to 60µs from the start of the writing time slot initialization. The logical level on the bus during the sampling frame determines the reception by the slave of the corresponding data byte.

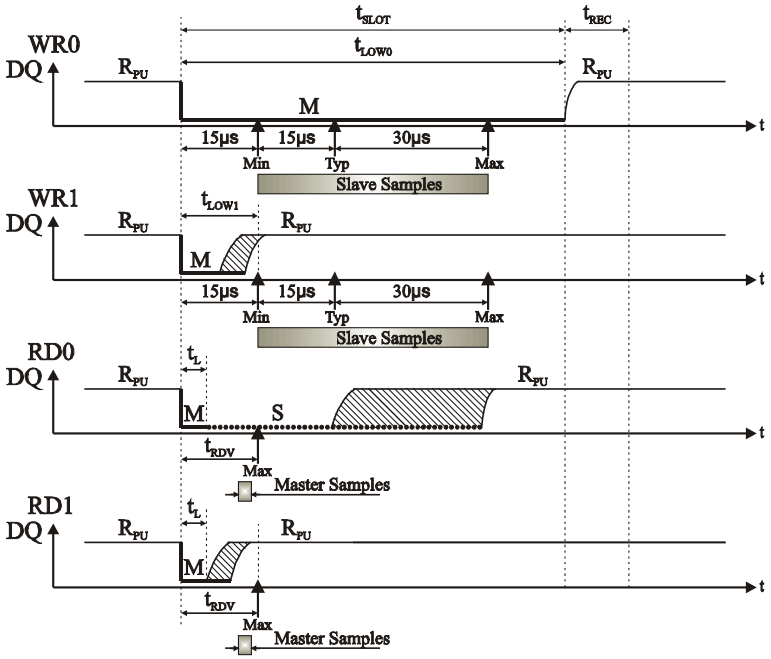

Fig. 10 – Time slots for writing and reading data.

The slave device can send data to the microcontroller only when the master sends a reading time slot and only following a command requiring a response from the slave. The microcontroller initiates a reading time slot by sending a 0 level on DQ for at least 1µs (t_L), then it releases the bus. The slave device sends a 0 bit by keeping the bus in 0 during a time slot RD0. In this case, the sensor releases the bus after the reading time slot, then R_{PU} pulls DQ in 1. The slave device sends a 1 bit by releasing the bus (R_{PU} pulls DQ in 1) during a RD1 time slot. Data at the slave device's output is valid no more than 15µs (t_{RDV}) from the falling edge that initiated the reading time slot. The microcontroller must release the bus after at least 1µs from the activation and must read the bus status in no more than 15µs from the start of the reading time slot (Maxim, 2019).

The command program is based on a few essential subroutines that perform various elementary and repetitive activities. An important and frequently used subroutine (*IP*) sends the initialization pulse and awaits for receiving the presence pulse. It performs the following activities: awaits for the 1-wire® to become inactive; sends the initialization pulse for 600µs; awaits for the bus to be inactive from 10µs to 75µs after sending the initialization pulse; awaits the reception of the presence pulse and checks that its duration is between 45µs and 300µs; lastly, it awaits for the bus to become inactive for at

least 100µs. If the activities listed above are performed, then it is considered that all the devices are initialized and are available to receive commands; otherwise, there is an error and a notification message is displayed on the console accordingly (Dragomir and Dragomir, 2013).

Other two basic subroutines perform one byte (command/data) writing in a single device or in all the devices connected to a bus (*WRByte*) and, respectively, one byte (data) reading from the addressed device (*RDByte*). These subroutines are implemented for a 90µs time slot and a 10µs recovery time. The one byte writing subroutine sends level 0 on the bus for 90µs for a 0 bit, while for a 1 bit, the level 0 is sent on the bus for 12µs. In the byte reading subroutine, the master initiates the reading of any bit by sending level 0 on the bus for 2µs; the microcontroller reads a data bit by sampling the bus after 12µs from the initialization of the reading time slot. These subroutines ensure a maximal data transfer baud rate of 10 kbit/s (Maxim, 2002b).

These basic subroutines are used to implement the following commands for managing the DS18B20 temperature sensor:
I_ - initialization of the slave devices in the 1-wire® network;
S_ - searching the unique identification codes of the devices connected in the network;
RR_ - reading the EEPROM memory of a single device connected in the network;
M_ - addressing only the device with the specified unique identification code;
SR_ - addressing all the devices in the 1-wire® network;
A_ - searching all the devices in the network with the alarm triggered;
T_ - temperature measuring;
W_ - writing into the scratchpad memory three data bytes;
RS_ - reading the scratchpad memory (9 bytes);
CS_ - copying the scratchpad memory into the EEPROM memory;
RE_ - restoring the EEPROM memory into the scratchpad;
P_ - reading from the 1-wire bus the slave devices' powering mode;
V_ - powering the devices in the network from an external DC power source.

The command program for managing the DS18B20 temperature sensors consists of the system variables initialization sequence, followed by programming the peripheral circuits, clearing the console display, then messages for execution launch and other operations. After that, the program enters a loop for keyboard interrogation that determines the input of various user commands, with the required parameters (if any), followed by displaying in various formats the data acquired from the sensors, using the subroutines of the monitor program.

5. Conclusions

The hardware structure described has been built in practice, it is simple and consists of an 1-wire® network with dozens of DS18B20 temperature

sensors requiring no external components; a development system equipped with an ATMEL microcontroller and a serial 1-wire® interface manages the sensors network. The application is used for measuring single-drop or multiple-drop temperature in various domains, in industrial and thermic systems, for controlling programmable thresholds temperature regulators, in health monitoring systems, in consumer products etc.

The temperature is measured sequentially from each sensor or simultaneously from all the sensors, is read from each individual sensor using the 1-wire® communication protocol, displayed in decimal on the serial console with a 0.5°C accuracy and a resolution up to 0.0625°C. A specific program sequence searches and identifies each sensor connected in the network and from those, only the ones selected by the user are usable for temperature measuring.

The command program written in machine language uses 4.3 Kbytes of flash memory, measures the temperature and features various commands for managing the sensors in the network, for data input and for displaying the acquired data in different formats. The program uses little memory space compared to its features and the possibilities it offers.

The application can be developed further, by adding other types of sensors, memories or peripherals to the 1-wire® network. Connecting a real-time clock and a non-volatile memory (EEPROM) allows to set up a database in order to store for each record the time and the data acquired from the sensors.

REFERENCES

Aghion C., Ursaru O., *Informatică aplicată. Introducere în microcontrolere*, Edit. PIM, Iaşi, 2015.

Baltag O., *Senzori şi traductoare*, Edit. BIT, Iaşi, 2003.

Chita M.A., *Senzori şi actuatoare*, Edit. Matrix Rom, Bucureşti, 2017.

Ciobanu L., *Senzori şi traductoare*, Edit. Matrix Rom, Bucureşti, 2006.

Dragomir F., Dragomir O.E., *Programarea în limbaj de asamblare a microcontrolerelor*, Edit. MatrixRom, Bucureşti, 2013.

Duma P., *Microcontrolere în telecomunicaţii*, Edit. TEHNOPRESS, Iaşi, 2007.

Ivanov V., *Senzori şi traductoare*, Edit. Universitaria, Craiova, 2018.

Petreuş D., Muntean G., Juhos Z., Palaghiţa N., *Aplicaţii cu Microcontrolere din Familia 8051*, Edit. Mediamira, Cluj-Napoca, 2005.

** ATMEL, *AT89S8253 Microcontroller*, Data Sheet, 2005.

** Maxim, *Interfacing the DS18X20/DS1822 1-Wire® Temperature Sensor in a Microcontroller Environment*, Application Note 162, 2002.

** Maxim, *1-Wire® Communication through Software*, Application Note 126, 2002.

** Maxim, *Reading and Writing 1-Wire® Devices Through Serial Interfaces*, Application Note 74, 2009.

** Maxim, *Programmable Resolution 1-Wire® Digital Thermometer*, DS18B20 Data Sheet, 2019 (rev.6).

** Taiwan Semiconductor, *Low Dropout Positive Voltage Regulator*, TS1117 Data Sheet, 2003.
** Texas Instruments, *Digital Logic*, Data Book, 2007.

MONITORIZAREA SENZORILOR DE TEMPERATURĂ DS18B20 ÎNTR-O REŢEA 1-WIRE®

(Rezumat)

Lucrarea descrie interfaţa hardware necesară pentru măsurarea temperaturii cu senzorul digital DS18B20, într-o reţea 1-wire® comandată şi controlată de un sistem de dezvoltare echipat cu un microcontroler din familia ATMEL. Gestionarea reţelei de senzori de temperatură foloseşte un calculator personal care este conectat la interfaţa serială UART a microcontrolerului, printr-un convertor USB-RS232. Programul de comandă măsoară temperatura de la senzorii din reţeaua 1-wire® , afişează temperatura în diverse formate, efectuează diverse prelucrări numerice pentru datele achiziţionate sau execută diverse comenzi utilizator.