Temperature sensor network using the 1-Wire interface

Marin Marinov*, Ivan Topalov*, Sergei Lukin**

*Technical University Sofia, Department of Electronics, 1756-Sofia, P.O. Box 43, Bulgaria, mbm@vmei.acad.bg
**IFMO, Per. Grizkova 14, St.Petersburg, Russia Lukin@ifmo.ru

It is hard to measure temperature in huge production areas and large buildings using one control center. The proposed solution offers the flexibility of connecting as many temperature sensors as they are needed. And what is more, you need only two wires for all the sensors. Main advantages of this system are its price, simple realization, mass production of its components, and easy implementation. All sensors need to be attached to the main system using only two wires. Compared to the wireless temperature sensors, the price is lower because of the absence of the RF transistor, the small microcontroller and the battery. Instead of these three components 2 wires are used for power and communication.

1. INTRODUCTION

The features of the system include: a possibility of attaching a great number of sensors – theoretically the number is unlimited, practically depends on the microcontroller and its memory resources. Range is 750 meters without the use of any repeaters; topology is distributed, which means that sensors are attached in parallel to each other. As it was mentioned before, there is a need only for two wires - one is ground and the other one is data line and also used for a power source. In case of malfunctioning of a sensor it is easy replaced, without even powering off the system, and the new sensor starts operating straight away. For clear and proper reading of the measured temperature a big 2 lines by 16 characters LCD is used. In case of too many sensors attached to the system it is better to use a personal computer as a visualization tool. The system offers a serial connection to a personal computer.

Before we start with the description of the system and its architecture we will discuss the temperature sensor and the 1-Wire protocol.

2. 1-WIRE TEMPERATURE SENSOR

The particular temperature sensor, which is used in the system, is DS18B20 from Dallas Semiconductors. The DS18B20 Digital Thermometer provides 9 to 12-bit centigrade temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. It has an operating temperature range of –55 °C to +125 °C and is accurate to – 0,5 °C over the range of –10 °C to +85 °C. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1–wire bus; thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment or machinery, and process monitoring and control systems.

Figure 1 shows a block diagram of the DS18B20. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (TH and TL), and the 1-byte configuration register. The configuration register allows the user to set the resolution of the temperature-to-digital conversion to 9, 10, 11, or 12 bits. The TH, TL and configuration registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

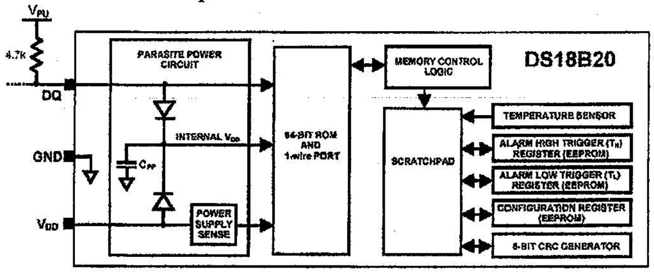

Figure 1. DS18B20 Block Diagram

The DS18B20 uses Dallas' 1-Wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18B20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one bus is virtually unlimited. Another feature of the DS18B20 is the ability to operate without an external power supply. Power is instead supplied through the 1-Wire pull-up resistor via the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (C_{PP}), which then supplies power to the device when the bus is low. This method of deriving power from the 1-Wire bus is referred to as "parasite power." As an alternative, the DS18B20 may also be powered by an external supply on V_{DD}.

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0,5 °C, 0,25 °C, 0,125 °C, and 0,0625 °C, respectively. The default resolution at power-up is 12-bit. The DS18B20 powers-up in a low-power idle state. The DS18B20 output temperature data is calibrated in degrees centigrade; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register. The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1.

The DS18B20 can be powered by an external supply on the VDD pin, or it can operate in "parasite power" mode, which allows the DS18B20 to function without a local external supply. Parasite power is very useful for applications that require remote temperature sensing or that are very space constrained. Figure 1 shows the DS18B20's parasite-power control circuitry, which "steals" power from the 1-Wire bus via the DQ pin when the bus is high. The stolen charge powers the DS18B20 while the bus is high, and some of the charge is stored on the parasite power capacitor (C_{PP}) to provide power when the bus is low. When the DS18B20 is used in parasite power mode, the VDD pin must be connected to ground.

In parasite power mode, the 1-Wire bus and C_{PP} can provide sufficient current to the DS18B20 for most operations as long as the specified timing and voltage requirements are met. However, when the DS18B20 is performing temperature conversions or copying data from the scratchpad memory to EEPROM, the operating current can be as high as 1,5mA. This current can cause an unacceptable voltage drop across the weak 1-Wire pull-up resistor and is more current than can be supplied by C_{PP}. To assure that the DS18B20 has sufficient supply current, it is necessary to provide a strong pull-up on the 1-Wire bus whenever temperature conversions are taking place or data is being copied from the scratchpad to EEPROM. This can be accomplished by using a MOSFET to pull the bus directly to the rail as shown in Figure 2.

The DS18B20 can also be powered by the conventional method of connecting an external power supply to the V_{DD} pin. The advantage of this method is that the MOSFET pull-up is not required, and the 1-Wire bus is free to carry other traffic during the temperature conversion time.

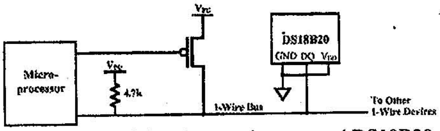

Figure 2. Supplying the parasite-powered DS18B20

Each DS18B20 contains a unique 64-bit code stored in ROM. The least significant 8 bits of the ROM code contain the DS18B20's 1-Wire family code: 28h. The next 48 bits contain a unique serial number. The most significant 8 bits contain a cyclic redundancy check (CRC) byte that is calculated from the first 56 bits of the ROM code.

3. 1-WIRE BUS SYSTEM AND PROTOCOL

The 1-Wire bus system uses a single bus master to control one or more slave devices. The DS18B20 is always a slave. The 1-Wire bus has by definition only a single data line. Each device (master or slave) interfaces to the data line via an open-drain or 3-state port. This allows each device to "release" the data line when the device is not transmitting data so the bus is available for use by another device. The 1-Wire port of the DS18B20 (the DQ pin) is open drain with an internal circuit equivalent to that shown in Figure 3. The 1-Wire bus requires an external pullup resistor (MOSFET) of approximately 5kΩ; thus, the idle state for the 1- Wire bus is high. If for any reason a transaction needs to be suspended, the bus must be left in the idle state if the transaction is to resume. Infinite recovery time can occur between bits so long as the 1-Wire bus is in the inactive (high) state during the recovery period. If the bus is held low for more than 480 μs, all components on the bus will be reset.

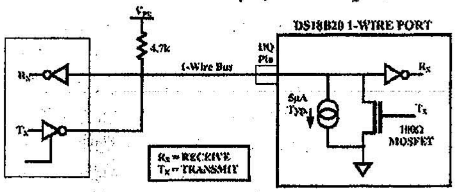

Figure 3. Hardware Configuration

All transactions on the 1-Wire bus begin with an initialization sequence. The initialization sequence consists of a reset pulse transmitted by the bus master followed by presence pulse(s) transmitted by the slave(s). The presence pulse lets the bus master know that slave devices (such as the DS18B20) are on the bus and are ready to operate. Timing for the reset and presence pulses is detailed in the datasheet of the DS18B20.

After the bus master has detected a presence pulse, it can issue a ROM command. These commands operate on the unique 64-bit ROM codes of each slave device and allow the master to single out a specific device if many are present on the 1-Wire bus. These commands also allow the master to determine how many and what types of devices are present on the bus or if any device has experienced an alarm condition.

The DS18B20 uses a strict 1-Wire communication protocol to insure data integrity. Several signal types are defined by this protocol: reset pulse, presence pulse, write 0, write 1, read 0, and read 1. The bus master initiates all of these signals, with the exception of the presence pulse.

4. Sensor Network

System architecture is shown on Figure 4. The main controller, is Microchip's PIC16F877, working on 20 MHz with a single instruction cycle for a 200 ns allowing enough speed for manipulating more than the needed sensors. Main advantages of the proposed controller are its program memory, which is of type FLASH. This reduces the time for debugging the firmware and allows easy update of it when it is connected to a personal computer through the serial port. The update is done on-the-fly, the whole system is working and by command from the personal computer the system goes into programming mode, programs it, and after it is done it restarts itself and goes back to normal operation. The other features of the microcontroller, which are used, are the timers (including the watchdog timer), the hardware serial port and some of the IO ports.

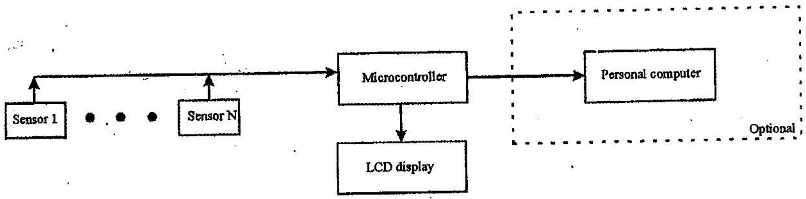

Figure 4. Sensing System Architecture

The sensing subsystem is using two IO ports of the micro. One is used for the communication with the 1-wire interface and the other one is the control pin for the strong pull-up. The IO sensor connector on the printed circuit board is of the type RJ45 and has also positive power supply extra to the IO pin and ground. In case one doesn't want to use the "parasite power" mode he can use the power supply provided.

The 1-wire communication is based solely on timing. When interrupt features are used then the microcontroller can be free for other jobs to do.

The LCD is using 7 IO lines for communication. Four data lines and three control ones. The serial connection to the personal computer is using the hardware serial port of the microcontroller. The maximum transfer speed might be 57600 bps and is adjustable. For easy navigation in the menu four buttons are used – one is "SELECT", one "CANCEL" and the other two are the directions UP and DOWN.

The firmware is written C. There are two separate libraries, one for the temperature sensors and the other one is for the LCD. The system works as follows: after the power is turned on the device goes into low power mode called sleep. It wakes up by the watchdog or by the press of a button. The system starts operating only when the power button is hold for more than 1,5 seconds. When the system is off but with VDD connected, the microcontroller is in sleep mode and wakes up by the watchdog timer, and goes back to sleep mode. It wakes up also by a key pressed. Once the system is turned on, the system checks for availability of sensors. The unique IDs of all the sensors are stored in the internal EEPROM at a previous stage, which are described later. The system checks each ID for the sensor it corresponds to. If system finds it on the sensor line then the sensor is online. After all the sensors are checked the system tells all the sensors to measure the temperature and then reads the values and displays them on the LCD. Then the microcontroller goes into sleep mode and after half a second it makes the next measurement and update of the display. That's all concerning normal operation. Anytime a button is pressed, it is decoded and the corresponding function is executed.

There is a special routine in the menu responsible for finding out the IDs of all the sensors on the 1-wire protocol. After they are found they are stored in the EEPROM so next time the system is powered up it knows all the IDs. During normal operation if a sensor is not responding its ID is deleted and it is not accessed anymore. If a new sensor is attached, the searching routine can be invoked again and it will find the new ID and added to the existing database of sensors.

The accuracy of the system depends only the accuracy of the sensors. In our case the accuracy is fixed ±0,5 °C because all the sensors are factory calibrated.

Every time a new temperature measurement is made it is transmitted over the serial port to the personal computer, where LabVIEW software collects the data, makes the calculations and displays the information.

5. Conclusions

Architectures incorporating 1-Wire technologies have produced low-cost, low-power, and accurate sensor systems, ideal for portable and distributed measurement applications. Integrated systems allow the designer the opportunity to place the signal conditioning necessary for direct sensor interface on a chip, which reduces analog circuit design and layout complexity. These systems also offer better control of specifications and error budgets than those made up of discrete components.

The system conceptually developed here, allows investigating some problems related to the distributed measurement applications and the possible solutions. More and more, measurement systems' performance will be defined by software. And for that reason users will demand to be able to interchange hardware and software from multiple vendors. The industry can make this possible only through software standardization based on widely accepted technologies such as standard interfaces.

6. References

[1] Goes, F.: Low-Cost Smart Sensor Interfacing. Delft University Press. Delft, 1996.

[2] Hauptmann, P.: Sensoren-Prinzipien und Anwendungen. Hanser Verlag, München, Wien, 1990.

[3] Ristic, L.: Sensor Technology and Devices, Motorola edition, 1991.

[4] Schmidt, W-D.,: Sensorschaltungstechnik, Vogel Verlag, Würzburg, 1997.

[5] Sze, S.: Semiconductor Sensors, John Wiley, New York, 1994.