Interfacing an MSP430 MCU and a TMP100 Temperature Sensor

MSP430 Applications

ABSTRACT

This application report describes the hardware and software interface between the Texas Instruments MSP430F413 16-bit microcontroller (MCU) and the TMP100 digital temperature sensor. This solution implements an ultra-low-power temperature measurement system that can operate from a single 3-V battery for over 10 years. This document describes how to interface the two devices in hardware and how to implement the communication protocol through a 2-wire I^{2}C bus in software. The concepts in this document can be applied to a software interface for any MSP430™ MCU connected to any I^{2}C-compatible slave peripheral device.

The source code described in this document can be downloaded from www.ti.com/lit/zip/slaa151.

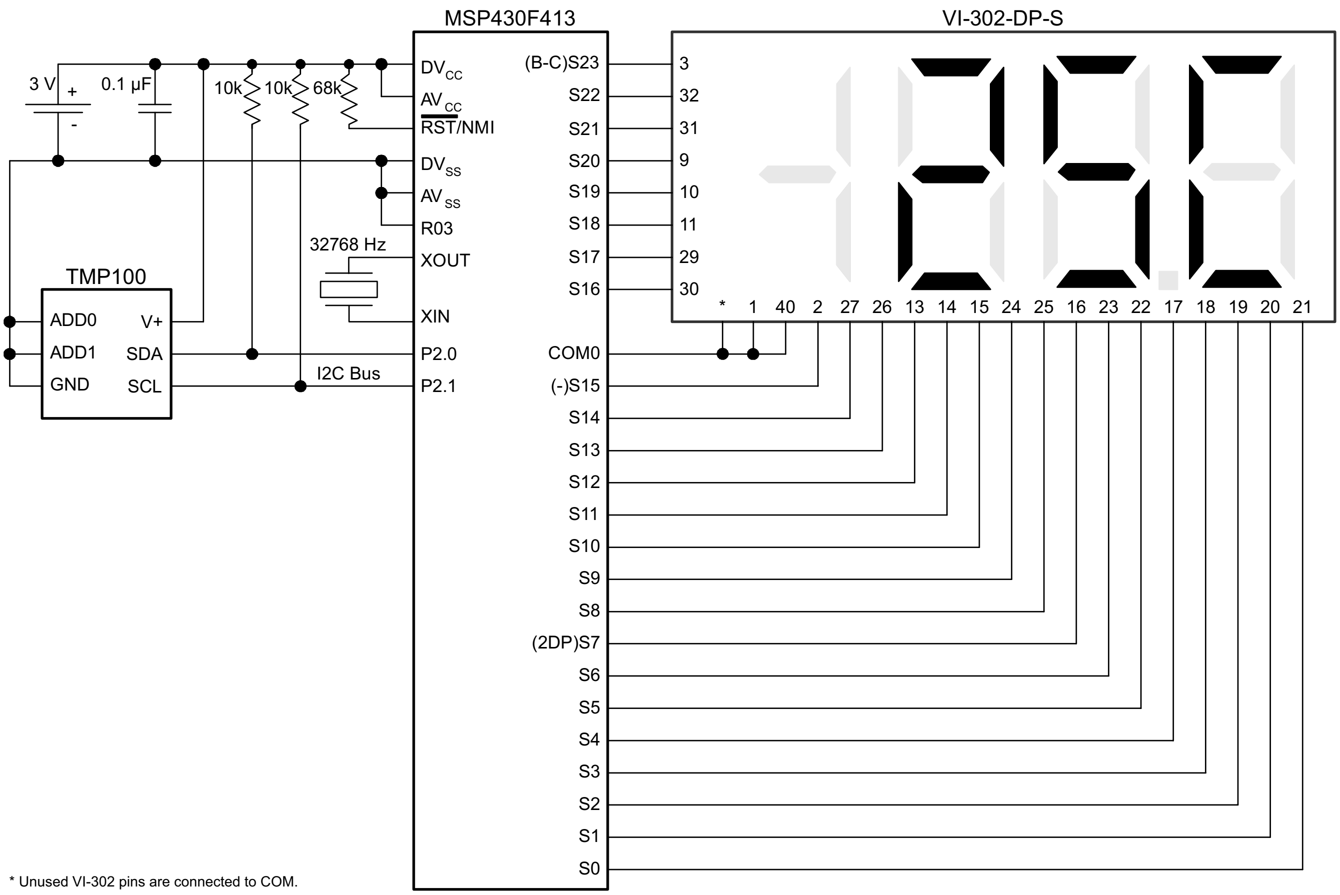

Figure 1. MSP430F413 and TMP100 Schematic

Contents

List of Figures

Trademarks

MSP430 is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

1 Introduction

Many applications require temperature sensing and measurement. Often, these measurements must be performed remotely requiring battery-powered nodes and, consequently, low power consumption to enable increased system life time. Many measurement solutions exist, including the use of discrete transistors, thermistors or thermocouples, and system-on-a-chip (SoC) integrated circuits. The TMP100 temperature sensor combined with the ultra-low-power MSP430 microcontroller is full function temperature measurement system (see Figure 1) that can operate for more than 10 years from a single coin-cell battery.

2 Theory of Operation

The system composed of an MSP430 MCU plus a TMP100 measurement is designed to operate on as little power as possible while continuously displaying the real-time temperature of the TMP100 sensor environment. When operated in one-shot mode, the TMP100 goes into shutdown mode after each conversion is completed and consumes a typical current of 0.1 µA between conversions. Coupled with the energy-efficient standby modes of the MSP430 MCU, which consumes 0.9 µA (typical) in low-power standby (LPM3), along with a 3.5-digit static LCD, these two devices create a complete temperature monitor and display system that supports 0.5°C resolution and consumes less than 2.45-µA total average current. Figure 2 shows the current profile of the system.

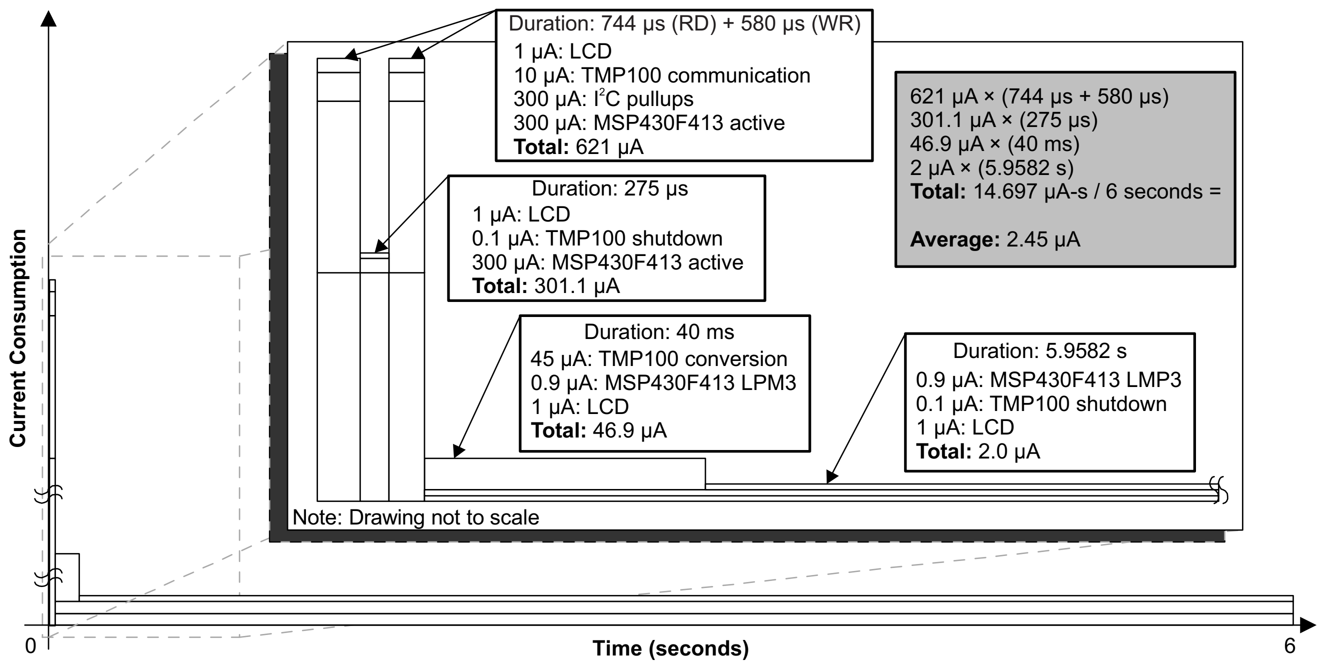

Figure 2. Current Consumption Profile

A 3-V CR-2032 lithium coin-cell battery provides power for the system. This battery is rated to provide a minimum of 220 mA-hours of operation. With the total system average current consumption of 2.45 µA (see Figure 2), a complete measurement and display platform can operate for 10 years from a single battery (see Equation 1).

$$220 \text{ mAh} / 2.45 \text{ µA} = 89796 \text{ hours} = 10.25 \text{ years} \quad (1)$$

To achieve such a low average current, the MSP430 software must use a minimum number of active clock cycles. The fast 6-µs wake-up time of the MSP430 DCO and CPU from low-power sleep mode along with the one-shot operation of the TMP100 provides the flexibility to create such a software flow. Figure 3 shows the software flow of the MSP430 MCU through a complete conversion and display cycle.

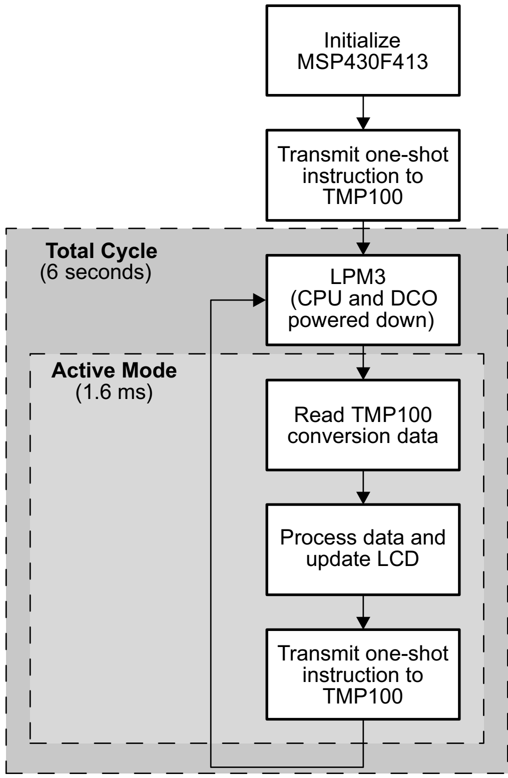

Figure 3. MSP430F413 and TMP100 Software Flow

By minimizing the active time of the MSP430 MCU to 1.6 ms per conversion cycle, the total system power consumption can be controlled. The fast wake-up of the CPU and DCO from LPM3 allow the MSP430 MCU to perform the required tasks and enter low-power mode as quickly as possible. The CPU then waits for the next timer-generated interrupt and repeats the cycle. The following sections give more description of the system operation and the MSP430 assembly code. More code optimizations exist, such as straight-line techniques to reduce subroutine calls. For clarity of the software flow, the subroutines are not included in the main loop. In addition to the software flow in Figure 3, additional routines (such as multiple peripheral support or Celsius-to-Fahrenheit conversion) can be added.

3 Hardware Interface

The system interface between the MSP430F413 and the TMP100 in Figure 1 requires few discrete components or connections. System power is provided from the 3-V lithium battery and recommended 0.1-µF decoupling capacitor. A 68-kΩ pullup resistor is recommended on the reset ($\overline{RST}$) input and a 32.768-kHz watch crystal provides the sleep-mode clock source for the MSP430 MCU during CPU power down. Two additional 10-kΩ pullup resistors are required on the SCL (clock input) and SDA (data I/O) of the TMP100 as required by the Phillips I^{2}C specification.

Temperature readout is provided using a 3.5-digit static liquid crystal display (LCD). The built-in LCD driver on the MSP430F413 supports up to 24-segment drive outputs and 4 common outputs, which gives a total display support of 96 possible segments. The implementation shown uses a static LCD, available from Varitonix, using a single common line and a total of 24 segment drive outputs. Although this example uses the MSP430F413, the same segment implementation can be applied to any of the MSP430F4xx MCUs.

4 System Operation

The I^2C communication to the TMP100 is provided entirely in software and uses two general-purpose I/O pins of the MSP430F413. The TMP100 addressing configuration supports up to 8 individual devices on a single I^2C bus through two address pins. In this application, these two inputs are tied to ground to set the 7-bit address of the TMP100 to 1001000.

All access to the TMP100 first requires the proper address (7-bit address and R/W bit) to be sent followed by the desired 8-bit pointer address. The pointer address specifies one of three registers to be updated by the host controller. In this application, the TMP100 configuration register is updated each time through the main software loop to initiate a new conversion using one-shot mode. The same methods can also be used to write to the remaining registers of the sensor.

The communication required to read the temperature conversion data register of the TMP100 begins the same as for the write cycle instructions. First, the device address and pointer data must be sent to the TMP100. Again, the 8-bit pointer specifies the register to be read by the MSP430F413. This is followed by a repeated start condition and sending of the TMP100 device address again. During this second device address transmission, the read/write bit is set high to indicate a read command of the register that was specified by the previously sent pointer address.

Reading the temperature conversion data from the TMP100 requires 16 bits of data to by read by the host. Two 8-bit transfers from the TMP100, sent MSB first, accomplish this. The conversion result depends on the desired resolution of the conversion, which can be configured from 9 to 12 bits in 1-bit increments. The power-up default is a 9-bit conversion result providing a 0.5°C/LSB resolution. This is the configuration used for the implementation described in this application report. The trailing LSBs transferred by the TMP100 are read as zero and can be ignored.

For general-purpose temperature display such as in thermostat applications, a temperature resolution of 1°C is often adequate. In such an application, the second byte (8 LSBs) of the result can be ignored as the required 1°C resolution is contained in the first 8 data bits. The second byte can be read out and handled by the MSP430F413 in software or simply not read at all. This is accomplished by simply not acknowledging transfer of the first byte of data. While this method returns an 8-bit result, it does not allow for continuous temperature conversion into the Fahrenheit scale. Reading only the first byte slightly reduces the active time of both the MSP430F413 and the TMP100, reducing the total system current, and is adequate for displaying Centigrade temperature with 1°C resolution.

After acquiring the conversion result from the TMP100, the MSP430F413 displays the calculated temperature on the LCD. This step involves handling of the MSB for negative temperature values, binary-to-BCD conversion, and updating the LCD registers that drive the display.

After the data is processed and the display is updated, the MSP430F413 returns to low-power mode 3 (LPM3), consuming 2.0 µA of total supply current (typical). Using a defined interrupt duration of 6 seconds, the CPU wakes up and begins the process again. The Basic Timer is used to generate the interrupt that wakes the MSP430F413. The software example uses the Basic Timer module set for a 2-second interval, which results in a typical average current of 3.55 µA and an estimated battery life of 7.5 years. Multiple methods can be used to manage the interrupt frequency to increase battery life, including the addition of multiple LMP3 calls or use of Timer_A. The interrupt frequency is fully programmable and can be adjusted to meet the requirements of a given application.

5 Summary

Temperature measurement tasks can be accomplished in a number of different ways. The manner in which any task is completed is almost always a function of the physical constraints imposed on the system. With portability and low power consumption, the MSP430F413 and TMP100 offer an effective and efficient solution to this common task.

Using the hardware design and software methods described here, the ultra-low-power combination of the MSP430F413 and TMP100 provide a complete temperature measurement and display system solution. Simple, small, and low-cost, this system not only completes the temperature measurement task, but does so in a manner that can provide continuous operation for 10 years from a single 3-V battery.

6 References

1. MSP430x41x Mixed-Signal Microcontrollers data sheet
2. MSP430x4xx Family User's Guide
3. TMP100 Temperature Sensor With I2C and SMBus Interface With Alert Function in SOT-23 Package data sheet
4. The I2C-Bus Specification Version 6.0, NXP Semiconductors
5. VI-302 LCD, Varitronix International Ltd

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from August 13, 2002 to June 22, 2018	Page
• Formatting and editorial changes throughout document	1

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265