Application Report
SLAA216A–February 2005–Revised May 2018

Implementing a Single-Chip Thermocouple Interface with the MSP430F42xA MCUs

MSP430 Applications

ABSTRACT

This application report describes how to implement a single-chip thermocouple interface without using the signal-conditioning circuitry normally required for thermocouples. The thermocouple interfaces directly to the MSP430F42xA microcontroller (MCU). The MSP430™ MCU uses the internal 16-bit sigma-delta analog-to-digital converter (ADC) to convert the thermocouple readings to a digital value, then converts them to a temperature, and displays the temperature on an LCD. Project collateral discussed in this application report can be downloaded from the following URL: www.ti.com/lit/zip/SLAA216.

Contents

Trademarks

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

1 Introduction

This application report describes how to implement a single-chip, direct thermocouple interface without using the signal-conditioning circuitry normally required for thermocouples. The thermocouple interfaces directly to the MSP430F42xA MCU. The MSP430F42xA MCU is an ultra-low-power microcontroller with integrated 16-bit sigma-delta ADC. The integrated ADC converts the thermocouple voltage into digital values, and the MSP430F42xA CPU converts the digital values into a temperature and displays them on an LCD. See Section 10 for a schematic. Download the code examples for more details.

For this application report, a type K thermocouple was used, and the measured temperature range was limited to 0°C to 99.9°C.

2 Thermocouples

Thermocouples are constructed of two dissimilar metals welded at one end. They produce a voltage at the nonwelded end that is proportional to the temperature difference between the two ends of the thermocouple. Measuring the thermocouple voltage is not enough to determine the temperature. That voltage is related only to the temperature difference between the two ends of the thermocouple. The temperature of the cold junction (the connection of the thermocouple to the measuring device) must also be known. As a result, some type of cold junction compensation is required. Often, circuits are employed to produce a voltage proportional to the cold junction temperature. This voltage is injected into the circuit and is part of the typical thermocouple signal conditioning circuitry.

Another technique of cold junction compensation involves measuring the temperature of the junction with a temperature sensor. This is the technique employed in this report. In this technique, the temperature of the thermocouple is calculate from the cold junction temperature and the thermocouple voltage.

Because this technique relies on a temperature sensor for the cold junction compensation, best results are obtained when the temperature of the sensor and the physical connection of the thermocouple to the circuit board are held at the same temperature. Often, commercial thermocouple measuring devices use an isothermal block to maintain or stabilize the temperature at the cold junction. The isothermal block may be as simple as pouring copper on the printed circuit board around the thermocouple connection and the cold junction temperature sensor, or the isothermal block could be more sophisticated and involve mechanical assemblies. This application report uses the simpler PCB approach.

The voltage that thermocouples produce is standardized by the National Institute of Standards and Technology. Data tables for thermocouple voltages are available from the NIST.

3 Hardware

The circuit used in this application report is shown in Section 10. The MSP430F42xA devices integrate three fully independent 16-bit sigma-delta ADCs, each with a programmable gain amplifier (PGA). This application uses two of the ADCs. One ADC measures the thermistor, and the other ADC measures the thermocouple. The resolution and PGA capability of the ADC allow the elimination of all but the simplest signal conditioning for the thermocouple. Only a simple RC filter is used for noise filtering. No other external amplifiers or compensation circuits are required.

The thermistor is used for the cold junction compensation. The thermistor used for this application report is part number KC003T-ND from Digi-Key Electronics. A resistor divider is formed with a 47-kΩ resistor and the thermistor to produce a voltage input to the ADC. The top of the divider is connect to the 1.25-V reference voltage output, and the full-scale input of the ADC is ±600 mV. The thermistor resistance decreases with increased temperature. When the thermistor is at 0°C, its resistance is 32.77 kΩ. The 47-kΩ resistor keeps the thermistor input voltage well below the ADC full-scale voltage for the temperature range of 0°C to 40°C. An RC filter is used on the thermistor input voltage for noise filtering, and the PGA is set to a gain of one.

The thermocouple is a type K. It is connected directly to the ADC, with the same RC filtering as the thermistor. No other signal conditioning is used for the thermocouple circuit, except for setting the PGA of the ADC to a gain of 16.

The 32768-Hz crystal provides a low-frequency clock for the Basic Timer. The Basic Timer supports low-power operation by allowing the rest of the device to be in LPM3 while only the LCD driver and a timer are operational. The crystal also provides the time base for the LCD driver.

The LCD used is the SBLCDA4 from Softbaugh, which is a 4-mux 7.1-digit LCD. Resistors R12, R14, and R17 provide the voltage levels for the integrated LCD driver.

Two switches are provided for a calibration function. Each switch is debounced with a 0.1-µF capacitor across its pullup resistor.

4 Software

Figure 1 shows the software flow. The following sections describe this flow.

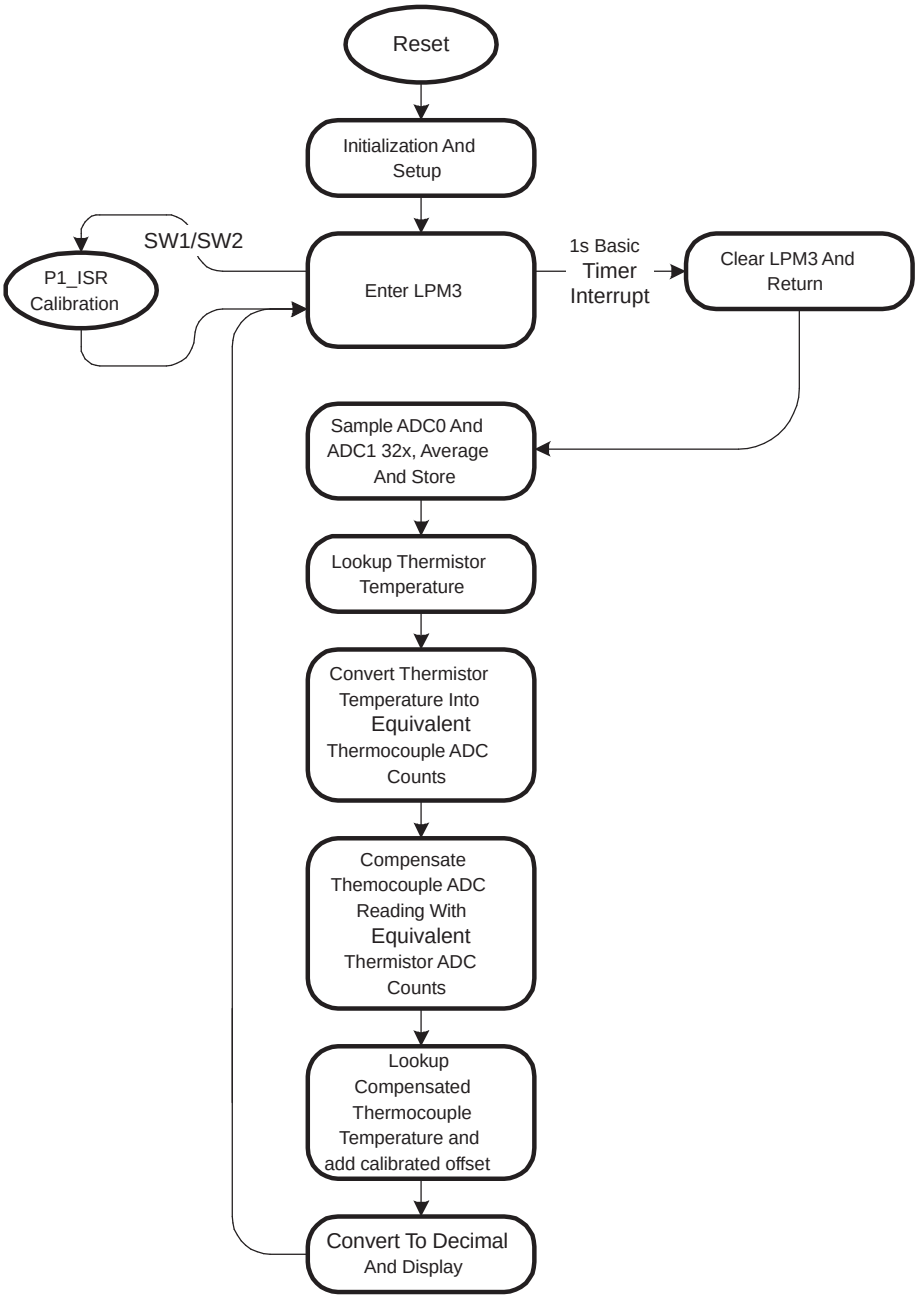

Figure 1. Software Flow

4.1 Overview

The software provides a low-power system. The MSP430 MCU is usually in low-power mode 3 (LPM3) with only a 32768-Hz clock running. The Basic Timer uses the 32768-Hz clock to provide a one-second interrupt that wakes the CPU for the temperature measurement.

This application uses two of the three integrated SD16 modules. They are grouped together to provide simultaneous sampling of the thermistor and thermocouple. During sampling, 32 conversions are performed on both the thermistor and thermocouple voltage and then averaged into a single conversion value. Next, the thermistor value is converted to temperature using a lookup table. The resulting thermistor temperature is then converted to an equivalent ADC count value for the thermocouple. This equivalent value is added to the ADC conversion value for the thermocouple, which provides the cold-junction compensation. Finally, the resulting temperature is determined using the adjusted thermocouple ADC value and a lookup table.

After the temperature is determined, it is converted to BCD format and displayed on the LCD. After display, the MSP430 CPU enters LPM3 again and waits for the next Basic Timer interrupt. The complete application consumes only 2.5 µA in LPM3, even with the LCD active.

A calibration routine is also provided. This allows for a single-point calibration of the initial temperature reading.

4.2 Initialization and Setup

During setup, the MSP430 watchdog timer is disabled, the stack pointer initialized, and the FLL is left at its default setting, which results in a CPU clock speed of 1.048 MHz. The LFXT1 crystal oscillator capacitors are selected so the 32768-Hz crystal oscillates properly. Next the Basic Timer is initialized for a one-second interrupt, and each SD16 is initialized for clock source, data format, grouping, interrupt enable, and gain setting. Finally, the MCU enters LPM3.

4.3 Mainloop

The Mainloop is a simple loop of subroutine calls. This loop is executed once per second, triggered by the Basic Timer interrupt. First the sample routine is called. Then the thermistor temperature is determined. If the thermistor temperature is in range, it is translated to thermocouple ADC counts. Then the thermocouple ADC value is adjusted using the translated thermistor counts and the thermocouple temperature is determined. Finally, the temperature is converted to decimal format and displayed on the LCD.

4.4 Sample

The Sample routine first measures the offset of each ADC by selecting the shorted channel of each ADC and taking and buffering 32 ADC conversions. The data is averaged into one offset value for each ADC and stored.

After offset measurement, the thermistor and thermocouple are measured 32 times each. Each ADC conversion is buffered and then averaged into a single value and stored. Finally, the measured offset of each ADC is subtracted from the measurements.

Because the software is written for low power, the reference is enabled only during the Sample routine. After the measurements, the reference is turned off. The SD16 automatically enters a low-power mode if it is not actively converting, so no other reconfiguration is required for low-power operation.

5 Get_TR_Temp – Determining the Thermistor Temperature

To determine the thermistor temperature from the ADC value, first a table lookup is performed to determine the temperature to the nearest degree. Then, an interpolation is done to determine the temperature to the nearest tenth of a degree. The whole and tenths portion of the thermistor temperature are stored in separate variables and used separately to adjust the thermocouple ADC value.

Equation 1 interpolates the tenths portion.

((higher – ADCvalue) × 10) / (higher – lower)

where

- ADCvalue = ADC conversion value of thermistor voltage
- higher = the next higher value in the table
- lower = the next lower value in the table (1)

The table of thermistor values in the code is specific to this application. This table uses measured values for the 47-kΩ resistor and for the ADC reference. The general formula for computing the table is:

$$\text{ADCvalue} = \text{hex}\left(2^{N} \times \frac{\text{voltage}}{2 \times \text{Vref}}\right)$$

where

- N = ADC resolution in bits
- voltage = resulting voltage from the voltage divider
- Vref = reference voltage
- For this application the equation for the voltage divider is: (2)

$$\text{voltage} = \frac{V \times Rt}{Rt + 47\ k\Omega}$$

where

- Rt = thermistor resistance and
- V = the voltage source for the divider (Vref in this application) (3)

Combining the two equations results in:

$$\text{ADCvalue} = \text{hex}\left[\left(\frac{2^{N}}{2 \times \text{Vref}}\right)\left(\frac{\text{Vref} \times Rt}{Rt + 47\ k\Omega}\right)\right] \quad (4)$$

$$\text{ADCvalue} = \text{hex}\left(\frac{2^{N-1} \times Rt}{Rt + 47\ k\Omega}\right) \quad (5)$$

The Get_TR_Temp also includes range checking to determine if the thermistor temperature is beyond the allowable range. If so, the appropriate flag is set to alert other sections of the program to the out-of-range condition.

6 Adjust_TC_Val – Compensating The Thermocouple Measurement

The Adjust_TC_Val routine is the cold junction compensation for the thermocouple. To compensate the thermocouple, the temperature of the cold junction must be known and translated into an equivalent thermocouple voltage, and then added to the thermocouple voltage. Because this application performs the cold junction compensation in software, after measuring the cold junction temperature with the thermistor, the cold junction temperature is translated into equivalent ADC counts of thermocouple voltage.

The Adjust_TC_Val routine takes the whole portion of the thermistor temperature, TR_Whole, and pulls the corresponding thermocouple ADC value from the thermocouple data table. It then subtracts it from the next higher value in the table and multiplies this difference by the tenths portion of the thermistor temperature, TR_Tenths. Then this value is added to the table value that corresponds to the whole portion of the temperature. This yields an equivalent thermocouple ADC count value for the cold junction temperature, to the nearest tenth of a degree, even though the thermocouple data table is in one degree steps. Finally, the equivalent thermocouple ADC count value for the cold junction is added to the actual thermocouple ADC reading.

The Adjust_TC_Val routine is only executed if the thermistor temperature is in range.

7 Get_TC_Temp – Determining the Thermocouple Temperature

Converting the adjusted thermocouple ADC value into temperature traditionally involves complicated math routines. However, this application report simplified the task by using a table lookup rather than a calculation. A table was constructed in Excel with the desired temperatures to be measured and the corresponding type K thermocouple voltage. Then the voltage was converted into equivalent ADC counts. Determining the thermocouple temperature then becomes a simple table lookup with tenths interpolation just as was done for the thermistor (see Section 5). After the thermocouple temperature is determined, the temperature adjustment, set in the calibration routine, is subtracted to give the final corrected temperature.

As with the thermistor, the table of values used for the thermocouple temperature lookup is specific to this application and uses a measured value for the reference. Equation 6 shows the general formula for computing the table.

$$\text{ADCvalue} = \text{hex}\left(\frac{2^{N} \times \text{PGA} \times \text{Vtc}}{2 \times \text{Vref}}\right) \quad (6)$$

where

- N = 16
- PGA = 28.35 (see device datasheet)
- Vref = Reference for the ADC
- Vtc = Thermocouple voltage

The Get_TC_Temp also includes range checking to determine if the thermocouple temperature is beyond the allowable range. If so, the appropriate flag is set to alert other sections of the program to the out-of-range condition.

8 P1_ISR – Calibration Routine

The P1_ISR routine (see Figure 2) is the interrupt service routine for the two switches and implements a calibration routine. The calibration is a single-point calibration intended to calibrate initial temperature error. To enter the calibration routine, simultaneously press SW1 and SW2. When calibration mode is entered, the last measured temperature is displayed on the LCD. Press SW1 or SW2 to increment or decrement the displayed temperature. Set the displayed value to the correct current temperature and then press SW1 and SW2 simultaneously again to exit calibration mode. When calibration mode is exited, the difference between the measured temperature and the user-set temperature is calculated and stored as a temperature adjustment for the measured thermocouple temperature value.

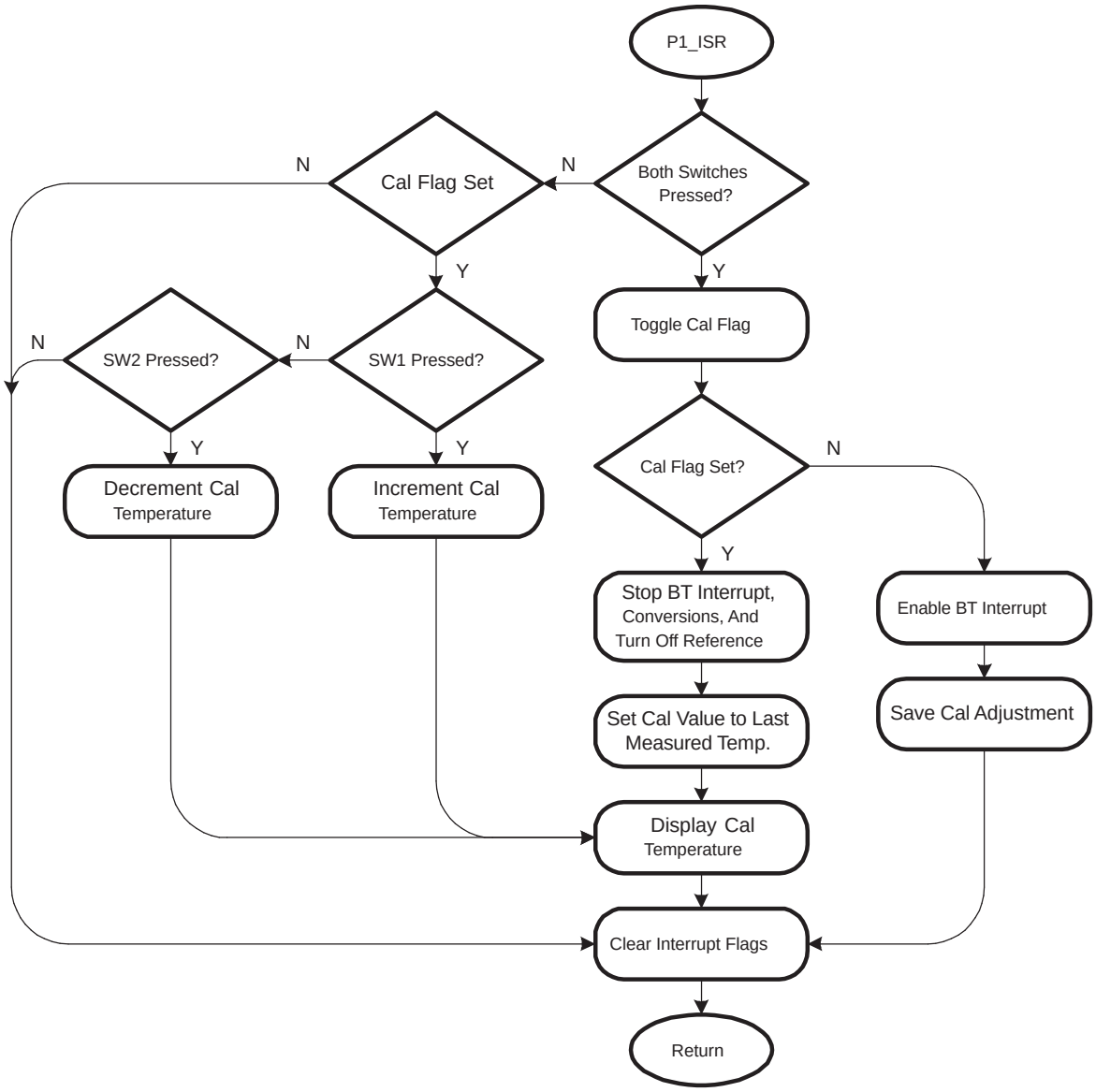

Figure 2. P1_ISR – Calibration Routine

9 Binary-to-BCD and Display

After the temperature is determined, it is converted from binary to BCD format and displayed on the LCD. The LCD display routine is generic for the chosen LCD. Unused portions of the LCD are blanked for this application. HI or LO is displayed if the thermistor or thermocouple temperature is out-of-range.

10 Schematic

Figure 3 shows the schematic.

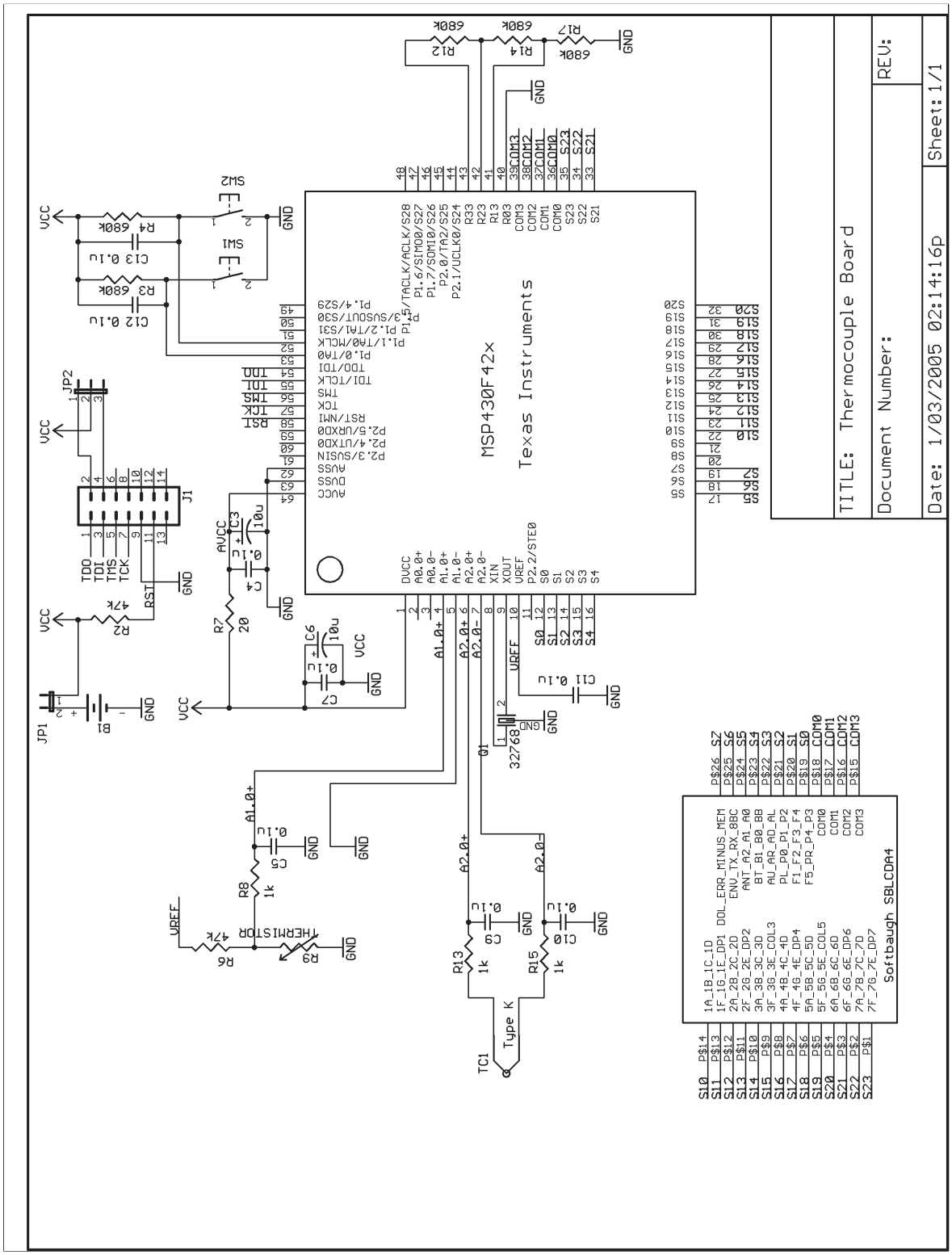

Figure 3. Schematic

11 References

1. MSP430x4xx Family User's Guide
2. MSP430F42xA Mixed-Signal Microcontrollers data sheet

Revision History

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

Changes from October 24, 2009 to May 29, 2018	Page
• Editorial changes throughout document	1
• Changed device from MSP430F42x to MSP430F42xA	1

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

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