Application Report

TMP6x Oversampling

Nicole Khoury

ABSTRACT

This document helps familiarize the user with the benefits of oversampling the TMP6x linear thermistor family and its applications in thermostat technology, as well as its advantages in calibration and accuracy.

The TMP6x thermistors are highly linear and have consistent sensitivity across temperature, with 6400 ppm/°C TCR at 25°C. With a typical TCR tolerance of 0.2% across the entire -40° to +125°C operating range, the TMP6x offers immunity to outside variations and robust performance. Engine management and motor monitoring systems often utilize extreme temperatures, and the linearity of the TMP6x devices offers better dynamic range at high temperatures than traditional NTC thermistors which exhibit non-linear behavior over temperature. For thermostats and other building automation systems there is a narrower operating range. For these room temperature applications oversampling can improve on accuracy and result in better performance. This document focuses primarily on room temperature applications and how the TMP6x devices can outperform NTC thermistors.

Table of Contents

List of Figures

List of Tables

Trademarks

1 Introduction

Higher accuracy temperature sensing can increase performance and efficiency by allowing systems to operate closer to thermal limits. In thermostats, technology is changing the way people are able to interact with and control their environment, and energy savings and efficiency have been key driving factors in design iterations over the years.

The high linearity of the TMP6x silicon-based thermistors enables better dynamic range at high temperatures but lower dynamic range at room temperatures. However, this linearity allows software improvements that are not possible with NTC thermistors, and when implemented the TMP6x devices can achieve higher accuracy and resolution regardless of dynamic range. This allows for maximum accuracy beyond room temperature, in addition to the better response time, lower drift, and ease of use that the TMP6x family enables.

NTC thermistors tend to have good accuracy at 25°C because this is the most common calibration point. Beyond that, the temperature acquired from the NTC voltage divider becomes less accurate. More complex, multi-point calibrations are costly and time consuming because they require a temperature soak. The linearity of the TMP6x devices allows for a single point calibration which can be accomplished without an expensive temperature soak. The resulting accuracy may then be further improved in software with oversampling methods. Generally, every eight oversamples increases the resolution of the ADC by two bits. Different oversampling methods as well as practical examples will be discussed later in this application note.

Figure 1-1. TMP61 in X1SON Package

2 Oversampling

Oversampling allows ADC resolution to be increased without resorting to more expensive, higher resolution ADCs. System resolution can be limited by an ADC, and oversampling can help to effectively increase the accuracy of the TMP61 across the entire temperature range.

This is particularly useful for room temperature applications because the environment has already been optimized. In addition to higher accuracy and measurement resolution, oversampling can improve the signal-to-noise ratio. Temperature values are stored in an array as they are calculated from the ADC bits. Following a first-in-first-out (FIFO) sequence, samples are shifted in the array as the new sample is added. The methods discussed below can be used for a variety of the values used in temperature conversion, including temperature, ADC bit value, divider voltage, or calculated resistance. However, the code examples in this document are averaging calculated temperature.

2.1 Method 1

The first method of oversampling considers the running average over an entire cycle. This can be preferable for systems such as thermostats and other environmental monitoring, because it might be better to wait for a complete cycle before averaging. This prevents passing temperature changes from drastically affecting the system response.

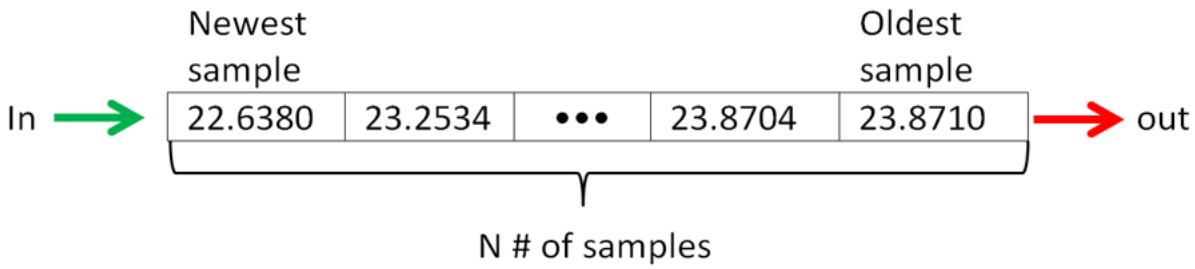

Figure 2-1. FIFO Averaging Method 1

2.1.1 Method 1 Example Code

```
// (1) Method one will read the ADC and average the last N values as set in "#define Tmp_1_length"
// FIFO setup of the temporary arrays and define the filter depth (samples to average)
#define Tmp_1_length 16     // Sample length for the averaging filter for oversampling
Float Tmp_1_array[Tmp_1_length];     // The FIFO arrays for averaging the ADC value

float ADC_AVG = 0;     // This is the averaged ADC value over (x) samples
float ADC_Value = 0;     // This is the most recent ADC value captured
int i = 0;     // set to 0
float sum_array_1 = 0;     // set to 0

void FIFO_AVG(void)
{
     // FIFO to average thermistor temperature
     i = 0;     // reset to 0
     sum_array_1 = 0;     // reset to 0
     for (i = 0; i < Tmp_1_Length - 1; i++)     // shift the array as a FIFO and drop the last data
value
     {
          Tmp_1_array[i] = Tmp_1_array[i+1];     // makes all the arra indexes equal to the number
after them
     }
     Tmp_1_array[Tmp_1_length - 1] = ADC_Value;     // add the new value to the beginning of the
array
     for (i = 0; i < Tmp_1_length; i++)     // sum the array
     {
          sum_array_1 += Tmp_1_array[i];      // add all of the array elements
     }
     ADC_AVG = sum_array_1 / Tmp_1_length;     // divide the sum of the array to get an average
}

// Read the ADC and place the bit value into ACD_Value
// Call the ADC_AVG function to get the last ADC value added and averaged into the array
FIFO_AVG();
// The ADC average value will be placed into ADC_AVG register
```

2.2 Method 2

An alternative method of oversampling involves continuously averaging the array elements each cycle. This is beneficial for systems that require continuous monitoring, such as a motor. If the motor starts to overheat, it is more beneficial to take very fast samples and then average them together as one, then all temperature changes can be closely monitored.

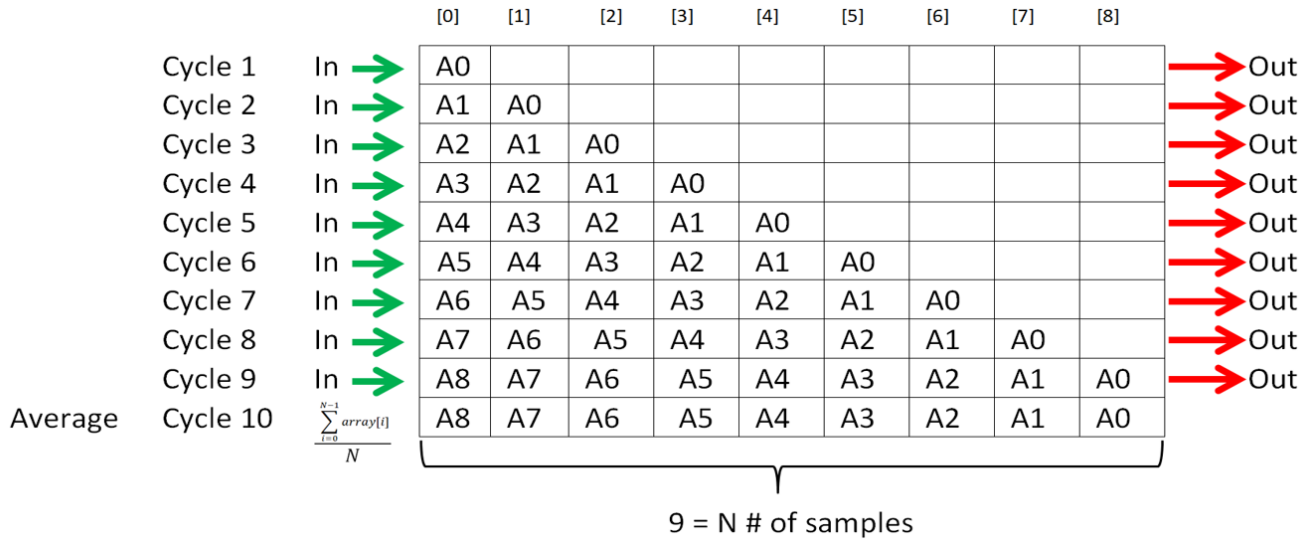

Figure 2-2. FIFO Averaging Method 2

2.2.1 Method 2 Example Code

```
// (2) Method two will read the ADC and insert that value into the FIFO N times as set in "#define
Tmp_1_length" then average the values
// FIFO setup of the temporary arrays and define the filter depth (samples to average)
#define Tmp_1_length 16     // Sample length for the averaging filter for oversampling
Float Tmp_1_array[Tmp_1_length];     // The FIFO arrays for averaging the ADC value

float ADC_AVG = 0;     // This is the averaged ADC value over (x) samples
float ADC_Value = 0;     // This is the most recent ADC value captured
int i = 0;     // set to 0 for the array position counter
float sum_array_1 = 0;     // set to 0
int j = 0;     // set to 0 for the sample counter

void FIFO_AVG(void)
{
     // FIFO to average thermistor temperature
     i = 0;     // reset to 0
     j = 0;     // reset to 0
     sum_array_1 = 0;     // reset to 0
     for (j = 0; j < Tmp_1_Length - 1; i++)     // shift the array as a FIFO and drop the last data
value, add the new value to the array for (x) samples
     {
          // read the ADC and put the new value into the ADC_Value register
          for (i = 0; i < Tmp_1_length - 1; i++)     // shift the array as a FIFO and drop the last
data value
          {
               Tmp_1_array[i] = Tmp_1_array[i + 1];     // makes all the array indexes equal to the
number after them
          }
          Tmp_1_array[Tmp_1_length - 1] = ADC_Value;     // add the new value to the beginning of
the array
          for (i = 0; i < Tmp_1_length; i++)     // sum the array
          {
               sum_array_1 += Tmp_1_array[i];     // add all of the array elements
          }
          ADC_AVG = sum_array_1 / Tmp_1_length;     // divide the sum of the array to get an average

// Call the FIFO_AVG routine to get N ADC values averaged
FIFO_AVG();
// The ADC average value will be placed into ADC_AVG register
```

2.3 Nyquist Rate

The Nyquist rate needs to be considered when oversampling for more resolution. It can be found according to the Nyquist Theorem, which states that it is two times the highest frequency component of the input signal. Sampling with a frequency above the highest frequency input component is known as oversampling.

2.4 Dithering

Dithering is the practice of intentionally adding noise to a system. In the case of averaging, this can help improve resolution error by allowing oversampling to improve resolution. If this purposeful noise is placed well outside of the system's frequency range, it can be easily filtered out. This results in a final measurement that is higher resolution and has less noise.

Neglecting to place a capacitor on the resistor divider is a good way to add noise, and this will usually be sufficient for averaging.

2.5 Resolution

Table 2-1 shows a comparison of the TMP61 and an NTC thermistor before any averaging, with both devices uncalibrated. While the NTC seems to have better voltage resolution, this should not be misinterpreted as accuracy. The temperature error of both devices is about the same.

Table 2-1. Comparison of TMP61 and NTC Thermistor Before Averaging

	TMP61	Reference	NTC
	22.6368 °C	23.5468 °C	24.2259 °C
	23.2534 °C	23.5468 °C	24.1310 °C
	23.8704 °C	23.5468 °C	24.0361 °C
Step size per °C	**0.6170 °C**		**0.0949 °C**
	Error		**Error**
	0.9100 °C		0.6791 °C
	0.2934 °C		0.5842 °C
	-0.3236 °C		0.4893 °C

Table 2-2 shows the resolution and error of the TMP61 and an NTC after method 1 was used to average the 16 element array. Again, while the resolution of the NTC is smaller, the temperature stability of the TMP61 allows it to greatly benefit from the averaging. The temperature error has decreased significantly for the TMP61 and the better voltage resolution of the NTC still does not improve the accuracy.

Table 2-2. Resolution and Error of TMP61 and NTC After Method

	TMP61	Reference	NTC
	23.2920 °C	23.5468 °C	24.1369 °C
	23.2535 °C	23.5468 °C	24.1429 °C
	23.2149 °C	23.5468 °C	24.1488 °C
Step size per °C	**0.0386 °C**		**0.0059 °C**
	Error		**Error**
	0.2548 °C		0.5901 °C
	0.2933 °C		0.5961 °C
	0.3319 °C		0.6020 °C

3 References

For related documentation, see the following:

- Texas Instruments, Thermistor Design Tool
- Texas Instruments, Methods to Reduce Thermistor Linearization Error, Memory, and Power Requirements Over Wide Operating Temperature Ranges Application Report
- Texas Instruments, TMP61 ±1% 10-kΩ Linear Thermistor With 0402 and 0603 Package Options Data Sheet
- Texas Instruments, Ultra-small, low-cost analog temperature sensor measurement circuit with ADC Analog Engineer's Circuit: Data Converters

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