AN70698

PSoC® – Temperature Measurement with an RTD

Author: Todd Dust and Praveen Sekar

Associated Part Family: PSoC 3, PSoC 4, PSoC 5LP, and PSoC Analog Coprocessor

Associated Code Examples: For a complete list, click here.

Related Application Notes: For a complete list, click here.

To get the latest version of this application note, or the associated project file, please visit http://www.cypress.com/AN70698.

More code examples? We heard you.

To access an ever-growing list of hundreds of PSoC code examples, please visit our code examples web page. You can also explore the PSoC video library here.

AN70698 explains the theory of temperature measurement using an RTD, and then shows how to do so with a single PSoC® 3, PSoC 4, PSoC 5LP or PSoC Analog Coprocessor without the need for external ADCs or amplifiers. It also explains how to calculate the resolution and accuracy of a given system.

Contents

1 Introduction....1
1.1 Using this Document....2
2 RTD – Theory of Operation....2
3 RTD Resistance Measurement Method....3
3.1 Two-Wire Measurement....3
3.2 Three-Wire Measurement....4
3.3 Four-Wire Measurement....5
4 RTD – Resistance-to-Temperature Conversion....8
4.1 Positive Temperatures....8
4.2 Negative Temperatures....8
4.3 Choosing the Right Polynomial Order....8
4.4 RTD Component....9
5 RTD Temperature Measurement with PSoC....11
6 Interfacing Multiple RTDs....13
7 Broken RTD Reconfiguration....13
8 Performance Measures....13
8.1 Temperature Resolution....13
8.2 Temperature Accuracy....14
8.3 List of all Errors....19
8.4 Test Results....19
9 Summary....20
10 Related Resources....20
10.1 Related Application Notes....20
10.2 Related Code Examples....20
Appendix A. Broken RTD Reconfiguration....22
A.1 Detecting Broken RTD Wire....22
A.2 Reconfiguring Four-Wire RTD to Three-Wire....25
A.3 One-Time Wire Resistance Computation....27

1 Introduction

Temperature is one of the most frequently measured environmental variables. RTD temperature measurement is typically done using one of four sensors: resistance temperature detector (RTD), thermocouple, thermistor, or diode. Table 1 compares these four sensor types.

Table 1. Comparison of Temperature Sensors

Parameter	RTD	Thermocouple	Thermistor	Diode
Temperature range	–200 to +850	–250 to +2350	–100 to +300	–50 to +150
Sensitivity at 25 °C	0.387 Ω/°C	40 µV/°C (K-type)	416 Ω/°C	250 µV /°C
Accuracy	High	Medium to High	Medium	Low

Parameter	RTD	Thermocouple	Thermistor	Diode
Linearity	Good	Fair	Poor	Good
Typical cost (US $)	$3 – $80	$3 – $15	$0.2 – $10	<$0.2
Typical distance of sensing	Surface mount for on-board temperature three- and four-wire up to a few hundred meters	<100 meters	Surface mount for on-board temperature Leaded for <1 meter	On-board temperature
Resource requirement	Excitation current, amplifier, ADC, reference resistor	Amplifier, ADC, voltage reference, and another temperature sensor for cold junction	Excitation current, ADC, reference resistor	Excitation current, amplifier, ADC
Response time	Slow	Fast	Fast	Slow
Computational complexity (best possible accuracy)	High	Very high	Very high	Medium

Although they are more expensive than other sensor types, RTDs have the best accuracy over a wide temperature range.

RTDs are primarily made of platinum, which gives them good linearity and repeatability. Commonly used RTDs include PT100, PT500, and PT1000. PT stands for platinum, and the number (100/500/1000) indicates the resistance value at 0 °C. This application note focuses on measurements based on PT100.

PSoC 3, PSoC 4, PSoC 5LP and PSoC Analog Coprocessor can fully integrate the hardware required for RTD temperature measurement. Not only can they integrate the hardware, but PSoC 3 and PSoC 5LP measure RTDs accurately and with high resolution due to the 20-bit delta-sigma ADC available on those devices.

In addition to the measurement front end you have all of the other resources of the PSoC device are available to complete the rest of your product. PSoC devices are attractive devices for reducing BOM cost and component count in your RTD designs, while maintaining the accuracy and resolution you need.

1.1 Using this Document

This document describes the theory behind temperature measurement with an RTD. If you are looking for code examples using PSoC to measure RTD temperature there are three associated with this Application Note, links to these code examples can be found in the Related Code Examples section.

2 RTD – Theory of Operation

An RTD has a positive temperature coefficient (PTC); resistance increases as temperature increases.

The resistance-temperature relationship is not perfectly linear. Various standards approximate this non-linearity; of them, IEC 60751 is the most widely used. Equation 1 and Equation 2 define the resistance to temperature relationship in IEC 60751.

Above 0 °C, RTD temperature is specified by the RTD resistance at 0 °C (R0) and constants A and B.

Equation 1 $R_T = R_0(1 + AT + BT^2)$, T > 0

Where, R_T is the resistance at T °C

Below 0 °C, in addition to the A and B, a third constant (C) is involved, as shown in Equation 2.

Equation 2 $R_T = R_0(1 + AT + BT^2 + C(T - 100)T^3)$, T < 0

Equation 1 and Equation 2 are referred to as Callendar–Van Dusen equations, and the A, B, and C coefficients are known as Callendar–Van Dusen coefficients. The term is named after the scientists who developed the equations.

The values of A, B, and C for a PT100 RTD are specified in IEC 60751 for standard industry-grade platinum:

$A = 3.9083 * 10^{-3}\ °C^{-1}$

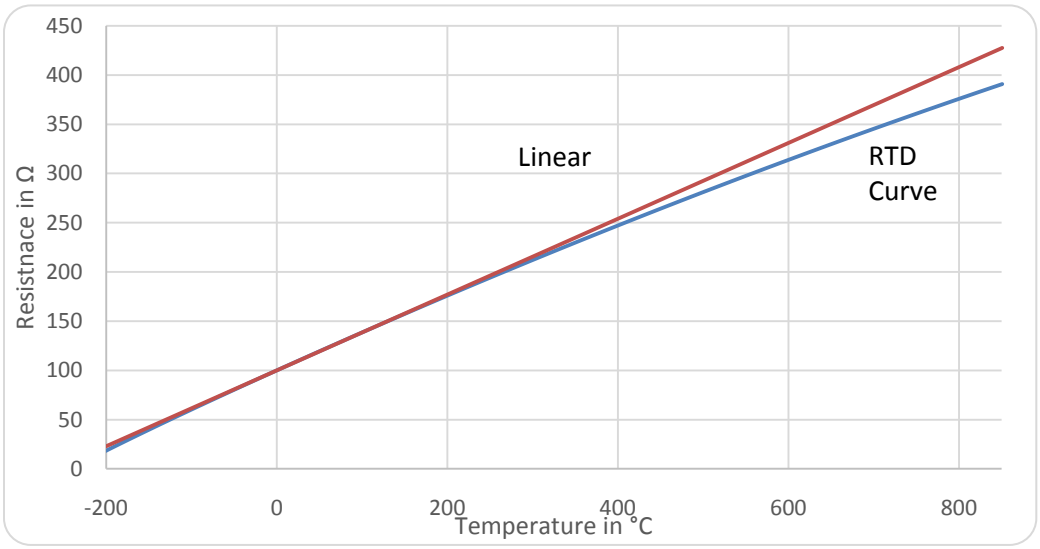
$B = -5.775 * 10^{-7}\ °C^{-2}$

$C = -4.183 * 10^{-12}\ °C^{-4}$

The resistance at 0 °C, $R_0 = 100\ \Omega$ (PT100 RTD)

Plotting resistance versus temperature yields a nearly linear curve, as Figure 1 shows. However, it is not a perfectly straight line. Figure 1 also shows the straight line approximation imposed on the RTD curve.

Figure 1. RTD Resistance versus Temperature



RTD temperature measurement involves the following two steps which are described in the consecutive sections:

1. Measure the RTD resistance accurately
2. Convert the measured resistance to temperature.

3 RTD Resistance Measurement Method

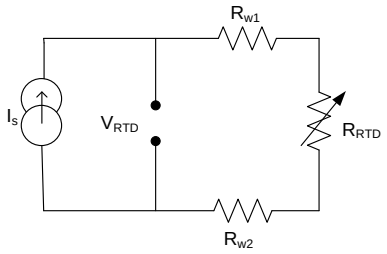
The following techniques are commonly used to measure resistance and their effectiveness for RTDs:

- Two-Wire Measurement
- Three-Wire Measurement
- Four-Wire Measurement

3.1 Two-Wire Measurement

In a two-wire measurement, a current is passed through the RTD and the voltage across the RTD is measured, as Figure 2 shows.

Figure 2. Two-Wire RTD Temperature Measurement



The linear approximation shown in Figure 1 is made by assuming the RTD changes by 0.387Ω/°C. A 1-ohm error in the resistance measurement leads to a temperature error of about 2.6 °C (1/.387). Therefore, the RTD resistance must be determined with an accuracy of <0.0385 Ω for the temperature error to be <0.1 °C.

Because of the length of the wire connected in series with the RTD, the RTD wire resistances (Rw1 and Rw2) are added to the RTD resistance. The wire resistances contribute to an error in the final RTD temperature measurement, as Equation 3 shows.

Equation 3 $V_{RTD} = I_S * (R_{RTD} + R_{W1} + R_{W2})$

R_{w1} and R_{w2} vary based on the length of the wire. For very short wires, they could be in the milliohms; for very long wires, 10s of ohms.

One way to minimize the effect of wire resistance is to use the three-wire method described in Three-Wire Measurement.

3.2 Three-Wire Measurement

With a three-wire measurement, the error due to the wire resistances is eliminated by measuring the wire resistances and subtracting it from the measured RTD resistance (see Equation 4 through Equation 6).

In Figure 3, the switch is set to position 1 and the voltage across the RTD, V_{RTD}, is measured. Then, the switch is set to position 2, and the voltage across the wire resistance, V_{wire}, is measured.

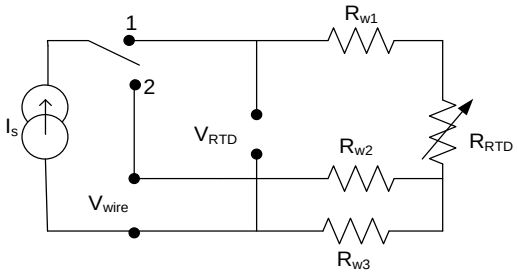
Equation 4 $V_{RTD} = I_s * (R_{RTD} + R_{w1} + R_{w3})$

Equation 5 $V_{wire} = I_s * (R_{w2} + R_{w3})$

If $R_{w1} = R_{w2}$,

Equation 6 $R_{RTD} = (V_{RTD} - V_{wire}) / I_s$

Figure 3. Three-Wire RTD Temperature Measurement

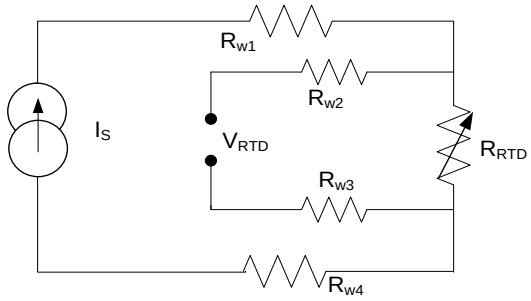


Although this method is better than the two-wire measurement method, the three-wire method provides accurate results only if $R_{w1} = R_{w2}$. To eliminate all wire resistance errors, use the four-wire measurement described in Four-Wire Measurement.

3.3 Four-Wire Measurement

The four-wire measurement shown in Figure 4 greatly reduces any error caused by wire resistances.

Figure 4. Four-Wire RTD Temperature Measurement



In this method, a known constant current is passed through the RTD. The voltage across it is measured using a separate sensing path. The separate sensing path ensures that the voltage drop across the wire resistances, R_{w1} and R_{w4}, does not affect the RTD voltage measured.

There is little voltage drop across resistances R_{w2} and R_{w3}, which are in the ADC measurement path, because there is negligible current flow into the high-input impedance terminals of the ADC.

The RTD resistance in this method is given by Equation 7.

Equation 7 $$R_{RTD} = \frac{V_{RTD}}{I_{RTD}}$$

To find the RTD resistance, the current source and the ADC measuring the voltage must be accurate. Specifically, the current source and the ADC should be free from offset, gain and non-linearity errors. Even a small error in voltage measurement can result in a large temperature error at higher temperatures.

To overcome the gain/offset error caused by the ADC and the current DAC (IDAC), add a reference resistor to your design.

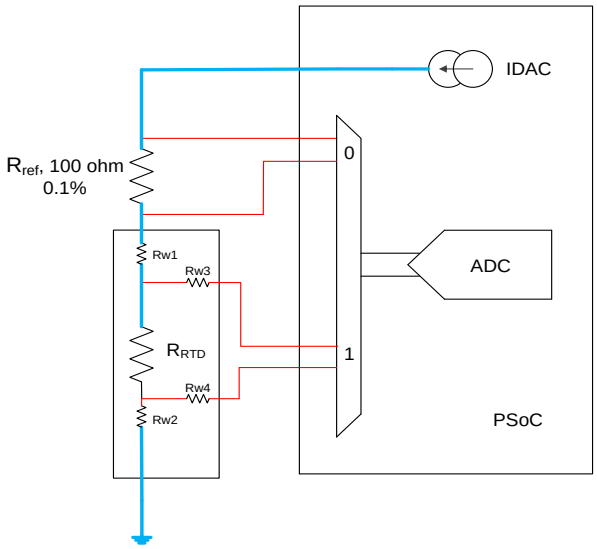
3.3.1 Reference Resistor Method

The reference resistor makes the measurement error independent of both the current source accuracy, and the ADC accuracy. The measurement error depends primarily on a reference resistance.

Figure 5 shows the schematic of this method. A constant current is passed through a known accurate reference resistance in series with the RTD. The voltage, V_{ref}, across the reference resistor and the voltage, V_{RTD}, across the RTD are measured. The RTD resistance is given by Equation 8.

Equation 8 $$R_{RTD} = \frac{V_{RTD}}{V_{ref}} * R_{ref}$$

Figure 5. RTD Temperature Measurement



The current path is shown by the blue line, and voltage measurement paths are shown in red. Current does not flow through wire resistances Rw3 and Rw4, because of the high-input impedance of the ADC. The rest of the application note discusses this method. The following sections discuss how to choose a reference resistor, and how the offset error and gain error of the ADC and IDAC become null.

3.3.1.1 Reference Resistor Selection

When choosing a reference resistor there are several rules of thumb to keep in mind.

1. Choose a small reference resistor that doesn't load the IDAC. Figure 5 uses a reference resistance of 100 ohms. PSoC IDACs have a compliance voltage of V_{DDA} – 1 V. This means that the voltage at the IDAC cannot exceed V_{DDA} – 1 V.

   If the maximum RTD resistance is 400 ohms, the reference resistor is 100 ohms, the internal PSoC routing is ~400 ohms and a 1-mA current is passed through the three in series, then a voltage of 900 mV is produced 1 mA * (400 ohms + 100 ohms + 400 ohms). This is well below the compliance voltage. For more information on PSoC internal routing resistance, consult AN58827 - PSoC® 3 and PSoC 5LP Internal Analog Routing Considerations.

2. Choose a reference resistor that uses the same ADC range as the RTD. In PSoC devices is it possible to have the ADC measure multiple input ranges, for example +/- 1.024V, or +/-V_{DDA}. Using the same range reduces the time it takes to measure the RTD, because the ADC does not need to be reconfigured.

   If the maximum RTD resistance is 400 ohms, and 1 mA is passed through it, the RTD produces a voltage of 400 mV. The delta-sigma ADC on PSoC 3 and PSoC 5LP has an input range of ±512 mV. Keep the voltage across the reference resistor in this 512-mV range. If it is outside this range, you must reconfigure the ADC every sample; reconfiguration takes time.

3. Choose a reference resistance similar to the resistance of the RTD at the temperature you are measuring. For example, if the RTD is measuring around 100 °C then pick a reference resistor with the same resistance as the RTD at 100 °C.

   If the reference resistor and RTD are in the same part of the ADC transfer function, non-linearities in the ADC are canceled out. If you are measuring a wide range of temperatures, keep the reference resistor near the middle of the range.

3.3.1.2 IDAC Current Selection

Another important factor is what current to pass through the RTD. There are two issues to consider.

1. The more current passed through the RTD and reference resistor, the more of the ADCs range can be used. For example, on PSoC4 the ADC has a range of +/-1.024 V. As stated earlier the max RTD resistance is ~400Ω thus to fill the whole range we need 1.024 V / 400 Ω = 2.56 mA.

   PSoC 3 and PSoC 5LP have an ADC input range of +/- 512 mV. Thus a smaller current of 0.512 V / 400 Ω = 1.28 mA can be used. Using more of the ADCs range leads to higher temperature resolution.

2. The more current passed through the RTD the more power it dissipates. This can lead to self-heating which leads to measurement error. Thus the IDAC current should be kept low. For more information on self-heating see RTD Self-Heating Error.

   Designers of RTD systems need to make tradeoffs between resolution and self-heating error.

   One common approach to avoid self-heating is to duty cycle the IDAC current. When not measuring, turn the IDAC off or disconnect ground. When measuring, turn it back on. In PSoC devices, the IDAC can quickly be turned off or you can connect the bottom of the RTD to a GPIO pin; set that pin to High-Z when not measuring, and set it to Strong Drive Low when measuring.

3.3.1.3 Offset Error Cancellation

In PSoC devices, the ADC offset and signal chain offset can easily be removed through correlated double sampling (CDS). In CDS the offset is measured, and then in firmware it is subtracted from the other voltage measurements. See AN66444 - PSoC® 3 and PSoC 5LP Correlated Double Sampling for details.

For RTD Temperature measurement, the best way to measure system offset is set the IDAC to source 0 mA and measure the voltage directly across the RTD or Reference Resistor.

With CDS, the equation for resistance measurement becomes:

Equation 9 $$R_{RTD} = \frac{V_{RTD} - V_0}{V_{ref} - V_0} * R_{ref}$$

Where, V_{RTD} is the voltage measured across the RTD

V_{ref} is the voltage measured across the 100-Ω resistor

V_0 is the offset voltage measured when the IDAC current is set to zero.

The offset current of the IDAC does not cause any error because it is nulled by the difference in Equation 9.

$V_{RTD} = (I+I_0) * R_{RTD}$ + ADC offset

$V_0 = I_0 * R_{RTD}$ + ADC offset

Subtracting V_0 from V_{RTD} removes the ADC offset. The difference nulls the I_0 term, which is caused by the IDAC offset. If CDS is done regularly, then offset drift is also canceled out.

Gain Error Cancellation

Assume that the ADC has a gain error of k and the DAC has a gain error of k'. These errors reflect as multiplicative factors in the voltage measurements, V_{RTD} and $V_{ref.}$ Because Equation 9 includes a ratio, the multiplicative errors k and k' cancel out.

Equation 10

$$R_{RTD} = \frac{k * k' * (V_{RTD} - V_0)}{k * k' * (V_{ref} - V_0)} * R_{ref}$$

Now the error depends primarily on the accuracy of the reference resistor, R_{ref}.

This method also removes any errors associated with gain drift, because the ratio metric measurement is being taken every time.

Using the reference resistor method, you can determine the RTD resistance accurately. The next step is to convert the RTD resistance to temperature, as described in RTD – Resistance-to-Temperature Conversion.

4 RTD – Resistance-to-Temperature Conversion

Once the resistance of the RTD is known, you must then convert it to temperature. The straightforward method to obtain temperature from resistance is to use the Callendar–Van Dusen equations. But Equations 1 and 2 show resistance in terms of temperature; you need to know temperature in terms of resistance. The solution is given below.

4.1 Positive Temperatures

Solving Equation 1 for T,

Equation 11 $$T = \frac{-A + \sqrt{A^2 - 4B\left(1 - \frac{R_T}{R_0}\right)}}{2B}$$

The other solution of the quadratic equation is eliminated using the known point (T = 0; R =100).

Using Equation 10 involves a square root, which requires a math library and about 5800 8051 CPU cycles for computation. This is a lot of time to spend doing a single calculation.

Instead of Equation 10, use a polynomial to calculate temperature. Compute the polynomial by first constructing a resistance versus temperature table and then using curve-fitting techniques on the table. Excel is a good tool to use for curve fitting.

Polynomial computation, which does not require a math library, executes faster. The temperature accuracy increases as the order of the polynomial increases. Using a fifth-order polynomial[1] can reduce the conversion error to <0.002 °C, but higher-order polynomials require more CPU cycles for execution.

4.2 Negative Temperatures

Solving Equation 2 for T is not straightforward, because it is a fourth-order equation. Again, approximate the temperature-resistance relationship using a polynomial. Using a fourth-order polynomial reduces the conversion error to < 0.002 °C.

A single polynomial can apply to both negative and positive temperatures. However, doing so requires a polynomial order greater than 10 to reduce the conversion error to <0.002 °C. Therefore, use two polynomials, for positive and negative temperatures.

4.3 Choosing the Right Polynomial Order

The temperature range and accuracy determine the order of the polynomial. As discussed, a fifth-order polynomial can reduce the error to <0.002 °C in the -200 °C to 850 °C range, but it requires more CPU cycles for computation. Table 2 provides the number of cycles required for temperature computation and the temperature error resulting from polynomials of different orders for the -200 °C to 850 °C range.

Table 2. Accuracy and Number of Cycles Required for Computation in Using Polynomials of Different Orders

Polynomial Order	Number of Cycles Required for Computation*	Accuracy in °C (-200 °C to 850 °C Range)**	Accuracy in °C (-50 °C to 150 °C Range)**
First order	70	< 20.2	<0.55
Second order	110	< 1.7	<0.007

[1] A fifth-order polynomial for temperature in terms of resistance is in the form $T = a_5R^5 + a_4R^4 + a_3R^3 + a_2R^2 + a_1R + a_0$, where a_5, $a_{4...}a_0$ are constants determined by curve-fitting techniques.

Polynomial Order	Number of Cycles Required for Computation*	Accuracy in °C (-200 °C to 850 °C Range)**	Accuracy in °C (-50 °C to 150 °C Range)**
Third order	150	< 0.17	<0.0001
Fourth order	190	< 0.018	0
Fifth order	230	< 0.002	0

Note: (*) The number of cycles required for computation is calculated from the PSoC 5LP-based code in the associated code example. The project uses floating-point arithmetic to compute temperature. You can greatly reduce the number of cycles by using special algorithms for floating-point multiplications.
(**) This only includes the accuracy of the resistance to temperature conversion, not the whole system accuracy

If your temperature range or accuracy requirements are lower, you can use a lower-order polynomial and decrease the number of cycles required for temperature computation. If your range is just -50 °C to 150 °C, you can use a second-order polynomial to achieve a temperature error of <0.007 °C.

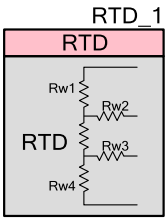
Choosing and creating the appropriate polynomial is math-intensive and time-consuming. Cypress simplifies this task by providing an RTD Component in PSoC Creator which creates a polynomial of the required order based on your temperature range and the accuracy required.

The Component automatically calculates the required order of the polynomial and polynomial coefficients. For resistance-to-temperature conversion, the Component provides an API which uses the computed polynomial coefficients to find the temperature.

4.4 RTD Component

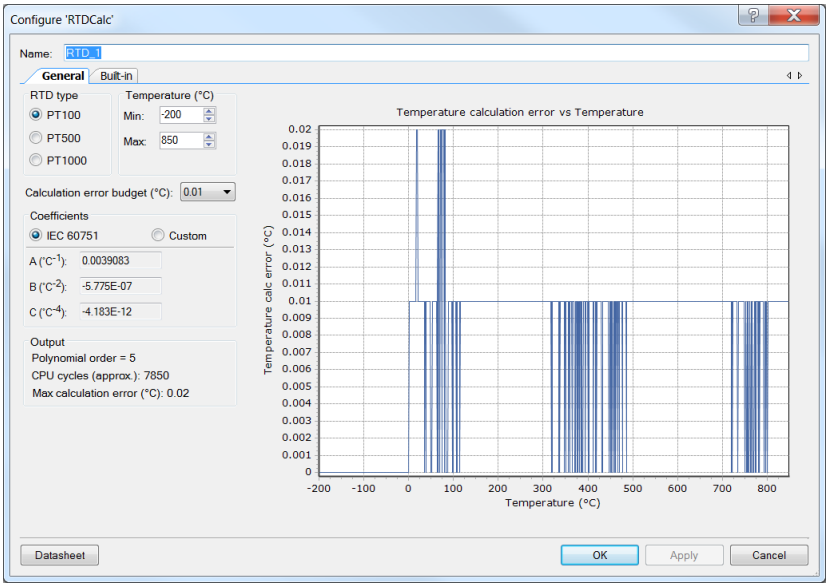
Figure 6 shows the RTD component. The RTD component supports PT100, PT500, and PT1000 RTDs.

Figure 6. RTD Component



Double-clicking the component yields the configuration dialog box shown in Figure 7.

Figure 7. RTD Component Customizer

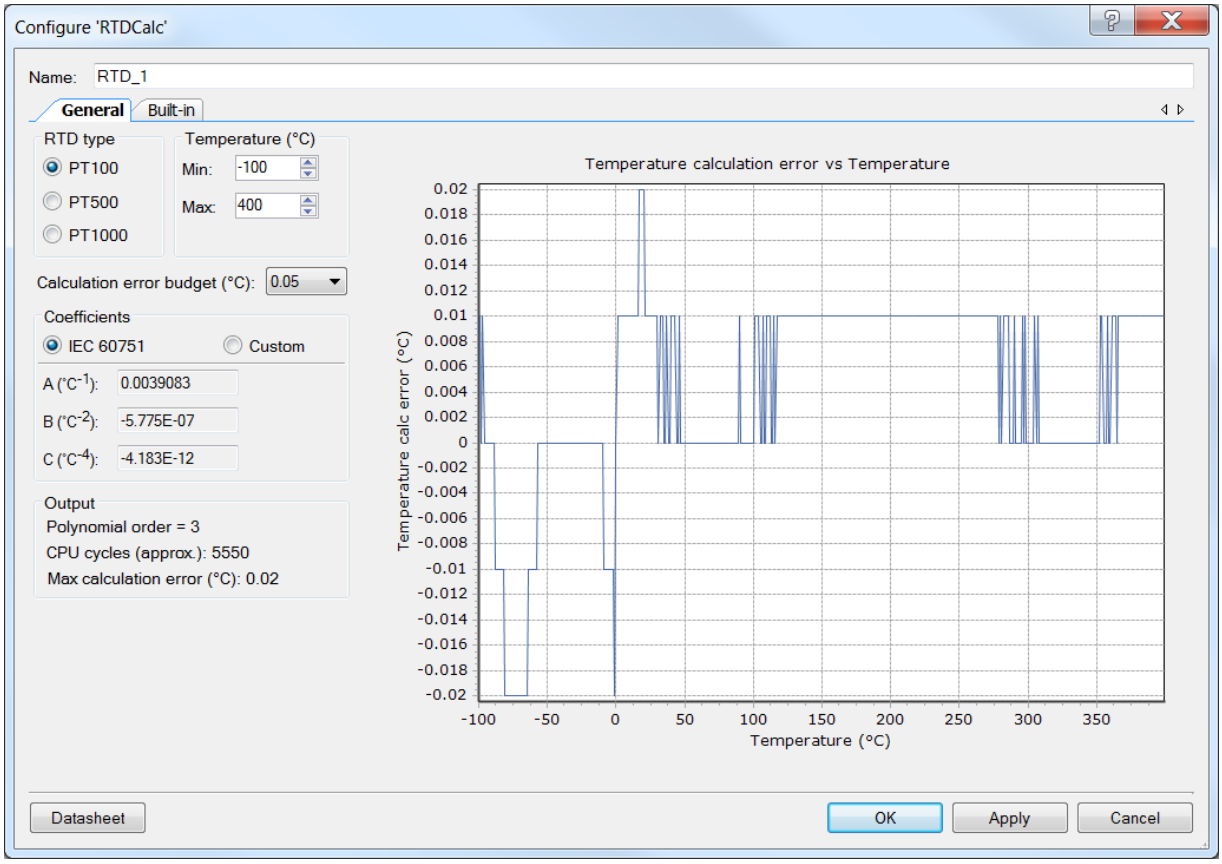


Select the RTD type, enter the maximum and minimum temperatures to which the RTD will be subjected, and select the calculation error budget. The configuration dialog returns an appropriate polynomial order for the chosen error budget, the temperature calculation error versus temperature graph, the maximum error for the chosen temperature range, and the number of cycles required for computation.

For example, if you are using a PT100 RTD, your range is -100 °C to 400 °C, and you require 0.05 °C temperature calculation accuracy. The component automatically chooses the appropriate polynomial for you.

Figure 8 shows that a third-order polynomial meets that requirement.

Figure 8. Customizer for −100 °C to 400 °C Range



In your code, call the API function RTD_GetTemperature(int32 res) to calculate temperature. The parameter, int32 res, is resistance in milliohms, and the return value is temperature in 1/100th of °C.

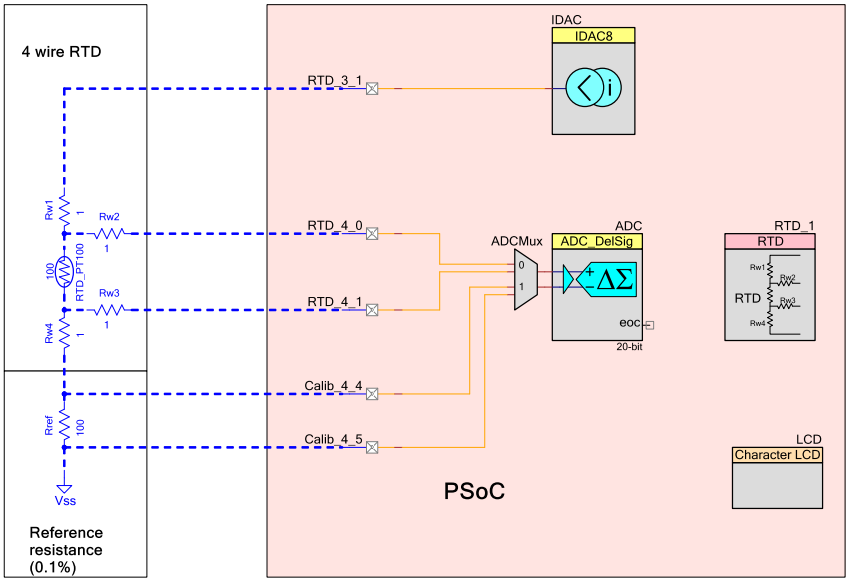
The temperature error depends on many factors in addition to resistance-to-temperature conversion error. The RTD customizer shows only the error due to resistance-to-temperature conversion. To accommodate other errors, ensure that the resistance-to-temperature conversion error is less than one-tenth of the total error budget. The other errors are discussed in the Temperature Accuracy section.

5 RTD Temperature Measurement with PSoC

CE210383 contains several projects demonstrating how to measure RTDs with PSoC 3, PSoC 4, PSoC 5LP and PSoC Analog Coprocessor. Please refer to CE210383 for details on how those projects work. This section briefly describes how to configure a PSoC device to measure an RTD Temperature.

Figure 9 shows a typical PSoC Creator schematic for a PSoC 3 or PSoC 5LP RTD temperature measurement project.

Figure 9. PSoC Creator Top Design Schematic for RTD Temperature Measurement

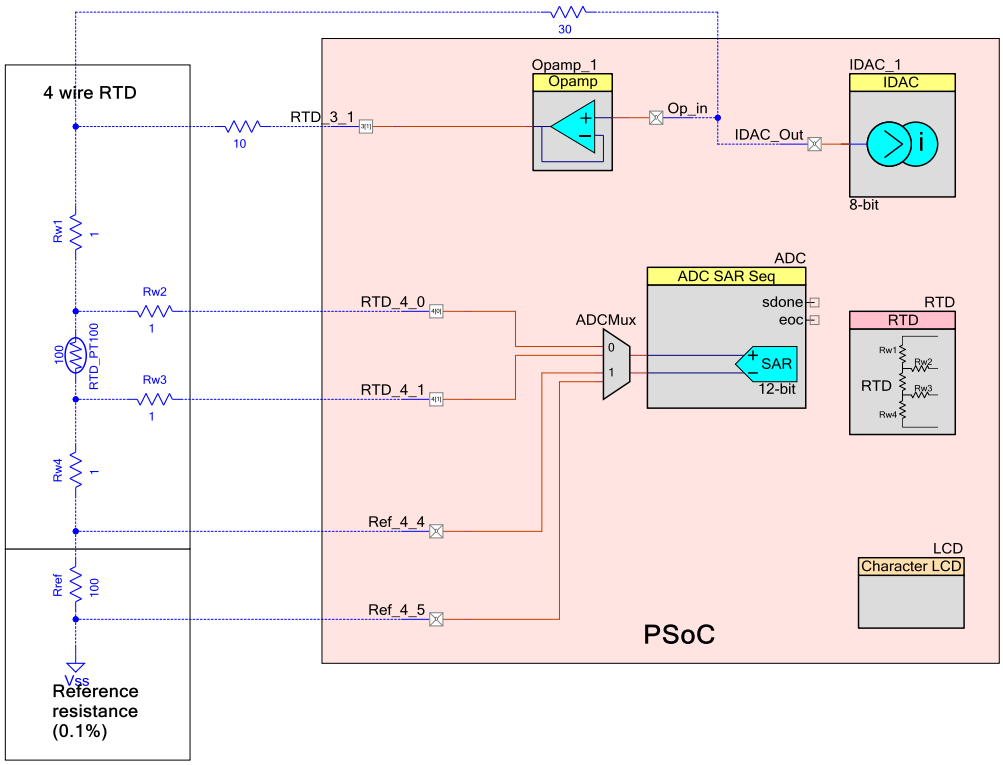


Notice how similar this schematic is to Figure 5. PSoC 3 and PSoC 5LP devices contain the required IDAC and ADC for RTD Temperature measurement.

PSoC 3 and PSoC 5LP are best suited for RTD Temperature measurement as they both have a high-precision 20-bit delta-sigma ADC with good linearity.

PSoC 4 is capable of measuring an RTD as shown in Figure 10. However, PSoC 4 and PSoC Analog Coprocessor has only a 12-bit successive approximation register (SAR) ADC. A 12-bit ADC reduces the achievable resolution of the RTD Temperature measurement to around 1 °C, and greatly reduces the accuracy when measuring RTDs. If a wide temperature range is not required it is recommended that thermistors be used for RTD temperature measurement with PSoC 4 and PSoC Analog Coprocessor. See AN66477 for more information.

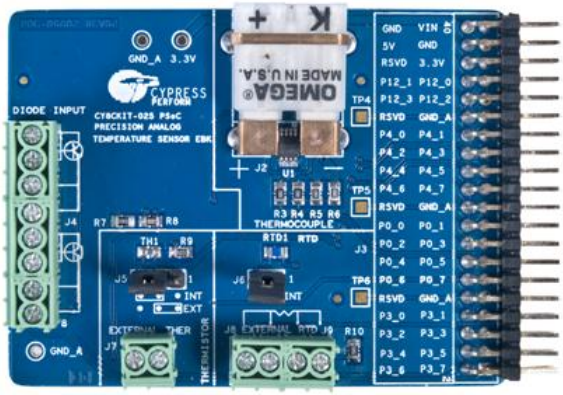
Figure 10. PSoC Creator Top Design Schematic for PSoC 4 RTD Temperature measurement



One thing to note about PSoC 4 and PSoC Analog Coprocessor is that maximum current that the IDAC can produce is 612 µA and 609.6 µA respectively. As mentioned previously, ~2.5 mA is needed to fill the entire ADC range. Thus the current needs to be increased. This can be accomplished through a simple current multiplier; two resistors are added to the opamp to increase the current by 3x. To increase the current by 4x change the 30-Ω resistor to 40 Ω.

Cypress has created a special kit for temperature sensing: the PSoC Precision Analog Temperature Sensor EBK (CY8CKIT-025). The kit provides four sensors—thermocouple, thermistor, RTD, and diode—for measuring temperature. In addition, connectors are provided to let you plug in your own thermocouple, thermistor, RTD, and diode. You can connect the EBK to the CY8CKIT-030 PSoC 3 Development Kit (DVK), or to the CY8CKIT-050 PSoC 5LP DVK. Figure 11 shows the kit. For more details on the kit, go to www.cypress.com/CY8CKIT-025.

Figure 11. PSoC Precision Analog Temperature Sensor EBK



6 Interfacing Multiple RTDs

The easiest way to interface to multiple RTDs is to wire them in series with one another. However, make sure you do not violate the compliance voltage of the IDAC, as discussed earlier.

Alternatively, you can easily interface as many as four RTDs with PSoC 3 / PSoC 5LP using the four IDACs in the device.

7 Broken RTD Reconfiguration

If one of the RTD wires is broken, you can reconfigure the four-wire RTD to a three-wire RTD and continue to measure temperature without too much loss of accuracy. Appendix A describes the broken RTD reconfiguration in detail. A project (Broken RTD reconfiguration) is also found in CE210435.

8 Performance Measures

8.1 Temperature Resolution

The temperature resolution depends on three factors:

- The temperature range to be measured
- The IDAC current
- The ADC resolution

The temperature range being measured is important because it determines the smallest change in resistance per change in temperature. As stated previously, the linear approximation for an RTD is 0.387 Ω/°C. However this approximation is not perfect at the extreme ends of the temperature range.

The first step in determining temperature resolution is to determine what the smallest change in Ω/°C is for your application. This is best done by creating a lookup table of resistance vs temperature in a spreadsheet for your RTD and temperature range. Use Equation 1 and Equation 2 to construct your table.

For example, if the temperature range is -200 °C to 850 °C, the difference between 850 °C and 849 °C is 0.2926Ω. The difference between -199 °C and -200 °C is 0.4322Ω. In general the smallest Ω/°C occurs at higher temperatures.

Next, take the minimum Ω/°C and multiply it by the current sourced by the IDAC. For this example 1 mA is used. This means that we need to resolve 0.001 A * 0.2926 Ω = 292.6 µV to resolve 1 °C. From this number we can assume that to resolve 0.1 °C we need a voltage resolution of 29.26 µV, and to resolve 0.01 °C we need a voltage resolution of 2.926 µV.

Next, we need to determine the entire voltage range for the temperature range being measured. For the temperature range is -200 °C to 850 °C the resistances are 18.52Ω to 390.77Ω respectively. Multiply this range by the IDAC current to get the voltage range. Multiplying this range by 1 mA yields 18.52mV to 390.77mV, the ADC must be able to measure this range.

For PSoC 3 and PSoC 5LP the delta-sigma ADC has a maximum resolution of 20 bits over a voltage range of +/- 0.512 V. To determine the voltage resolution, take the voltage range divided by the resolution; 1.024V/(2^20) = 0.976 µV. This shows the PSoC 3 and PSoC 5LP are able to resolve less than 0.01 °C across the entire temperature range of the RTD.

For PSoC 4 the SAR ADC has a maximum resolution of 12 bits over a voltage range of +/-1.024V. The voltage resolution is 500 µV. For PSoC Analog Coprocessor, the Scan ADC has a maximum resolution of 12 bits over a voltage range of +/-1.2 V. The voltage resolution is 585.94 µV. This means that with PSoC 4 and PSoC Analog Coprocessor, the resolution is around 2.5°C over the entire range.

8.1.1 Increasing Resolution

There are several methods to increase the temperature resolution.

1. Increase the current sourced by the IDAC. Increasing the IDAC current increases the minimum voltage required to resolve 1 °C. For example if the IDAC current is increased to 3mA then the minimum Ω/°C becomes 877.8 µV. The voltage range increases to 0.055 V to 1.17 V. Now a PSoC 4 is able to resolve down to 1 °C. However, 3mA can lead to higher self-heating which decreases accuracy.
2. ADC resolution can be increased by oversampling. This is a common industry practice where multiple ADC samples are used to create a higher resolution result.
   - In order to increase the resolution by 1 bit, four ADC samples need to be summed, and then the result right shifted by 1 (divided by 2).
   - To increase the resolution by 2 extra bits, 4^2 ADC samples need to be summed and then the result right shifted by 2.
   - To increase the resolution by 3 extra bits, 4^3 ADC samples need to be summed and the result shifted right by 3.

     The resolution can be extended to any number of extra bits. The tradeoff is conversion speed – the more extra bits required, the more samples required for each conversion, and the slower the conversion.
3. Smaller Temperature Range. This allows for more IDAC current, and may increase the minimum Ω/°C if not measuring very high temperatures.

8.1.2 Noise

Another important part of temperature resolution is the noise present in the acquisition system. If the noise of the system is too large, it may reduce the resolution of the system.

The PSoC 3 and PSoC 5LP delta-sigma ADC has a specified RMS noise that is approximately 1 count in the +/- 512 mV range. That is noise of ~0.976 µV, which is well below the resolution required to resolve 0.01 °C.

To reduce the effect of noise, you can use a digital IIR filter. This helps reduce any noise flicker in the final result. For more information on digital IIR filters, refer AN2099, Single-Pole IIR filter.

8.2 Temperature Accuracy

You can calculate the temperature accuracy by summing all possible individual errors, which fit into one of two categories:

1. Error due to the measurement system
2. Error due to the RTD

8.2.1 Error Due to the Measurement System

The error due to the measurement system is due to the circuit shown in Figure 5. Consider Equation 8, which is used to obtain RTD resistance.

$$R_{RTD} = \frac{V_{RTD}}{V_{ref}} * R_{ref}$$

V_{RTD} is the voltage measured across the RTD

V_{ref} is the voltage across the 100-Ω resistance

R_{ref} is the reference resistance

As discussed, the only major source of error using this method is the accuracy of the reference resistance. The offset error is nulled by the difference, and the gain error is nulled by the ratio. The other source of measurement error is non-linearity in the ADC.

8.2.1.1 Error Due to ADC Integral Non-Linearity

The integral non-linearity (INL) of an ADC at any point is the difference between the ideal ADC count and the actual ADC count.

The delta-sigma ADC has a maximum INL of ±32 LSb in the +/-1.024 V range; 32 LSb corresponds to 64 µV for 20-bit resolution and ±1.024-V range. The ADC INL has the same analog value, 64uV, in ±0.512 V range.

Let us calculate the error due to INL at 850 °C for IDAC current of 1 mA.

$V_{RTD} = I * R_{RTD} = 1 * 390.481 = 390.481mV$ (The RTD resistance at 850 °C = 390.481, using Equation 1)

$V_{ref} = I * R_{ref} = 1 * 100 = 100mV$

Using Equation 8:

$$\text{Resistance error} = \frac{390.481}{100} * 100 - \frac{390.481 + 0.064}{100 - 0.064} * 100 = 0.3141\Omega$$

At worst case, the INL causes a measurement error of 0.3141Ω at 850 °C. This corresponds to an error of about 1 °C.

Note that we have taken the worst-case INL across PVT and substituted worst-case positive INL at the numerator and worst case negative in the denominator. In practical application, the error due to INL is much lower, and most likely in the same direction. For example, for an INL of +8 LSb at the numerator and +8 LSb at the denominator, the temperature error at 850 °C is ~0.1 °C.

The PSoC 4 SAR ADC has an INL of ~+/- 1.7 LSB in the 1.024V range at 12 bits. This corresponds to an error of ~850 µV. Plugging this into the equations above yields.

$$\text{Resistance error} = \frac{390.481}{100} * 100 - \frac{390.481 + 0.850}{100 - .850} * 100 = 4.2\Omega$$

Which corresponds to an error of ~12 °C.

Again this assumes worst case numbers. If we use more realistic numbers such as 0.8 LSB and in the same direction the error is ~1.23Ω, which is still an error of 4-5 °C.

The PSoC Analog Coprocessor Scan ADC has an INL of ~+/- 1.7 LSB in the 1.2 V range at 12 bits. This corresponds to an error of ~995 µV. Plugging this into the equations above yields.

$$\text{Resistance error} = \frac{390.481}{100} * 100 - \frac{390.481 + 0.995}{100 - .995} * 100 = 4.93\Omega$$

Which corresponds to an error of ~12.8°C.

Again, this assumes the worst-case numbers. If more realistic numbers such as 0.8 LSB are used, in the same direction the error is ~1.23 Ω, which is still an error of 4-5 °C.

So as can be seen PSoC 3 and PSoC 5LP have very good resolution and accuracy for RTD temperature measurement. PSoC 4 and PSoC Analog Coprocessor are capable of measuring an RTD however its accuracy and resolution is not good for high precision applications. If RTD temperature measurement is required for PSoC 4 and PSoC Analog Coprocessor a thermistor is recommended.

8.2.1.2 Error Due to Reference Resistance Tolerance

In Equation 8, we substituted 100 Ω for the value of the reference resistor, R_{ref}. But the actual value of R_{ref} will change because of its tolerance and temperature coefficient. Therefore, the value V_{ref}, which is measured across the reference resistance, will be erroneous. Assume that the tolerance of R_{ref} is 0.1 percent and the temperature coefficient is 10 ppm/°C.

$$V_{ref} = I * R_{ref}$$

Substituting the value of R_{ref} with the tolerance and temperature coefficient yields Equation 11.

Equation 12 $$V_{ref} = I * 100(1 + 0.001 + 0.00001 * (sysTemp - 25)$$

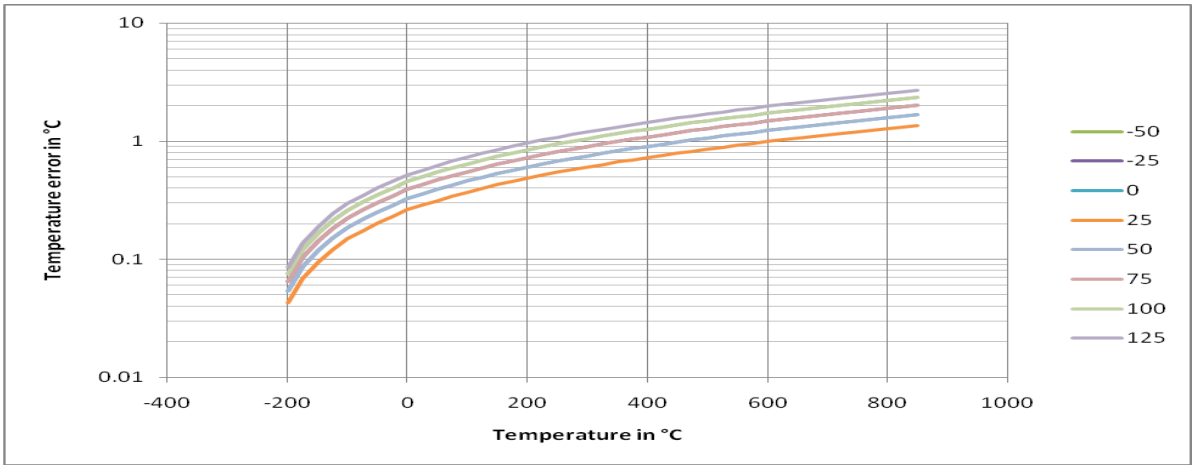
The measured value of the RTD resistance, R_{meas}, is:

Equation 13 $$R_{meas} = \frac{R_{RTD}}{(1 + 0.001 + 0.00001 * (sysTemp - 25))}$$

Where the 0.1 percent resistance tolerance contributes to the additive factor 0.001, and the temperature coefficient (10 ppm/°C) contributes to the additive factor 0.00001*(sysTemp-25). Note that the temperature coefficient is usually specified with reference to 25 °C. Therefore, the effect of the temperature coefficient is zero at 25 °C, but it increases as the temperature deviates from 25 °C.

Figure 12 shows the temperature error due to the reference resistor error at different RTD temperatures and ambient temperatures. Ambient temperature is the temperature of the reference resistance on the printed circuit board (PCB) while RTD temperature (x axis) is the actual temperature to which the RTD is exposed.

Figure 12. Temperature Error Due to Reference Resistor Accuracy and Temperature Tolerance



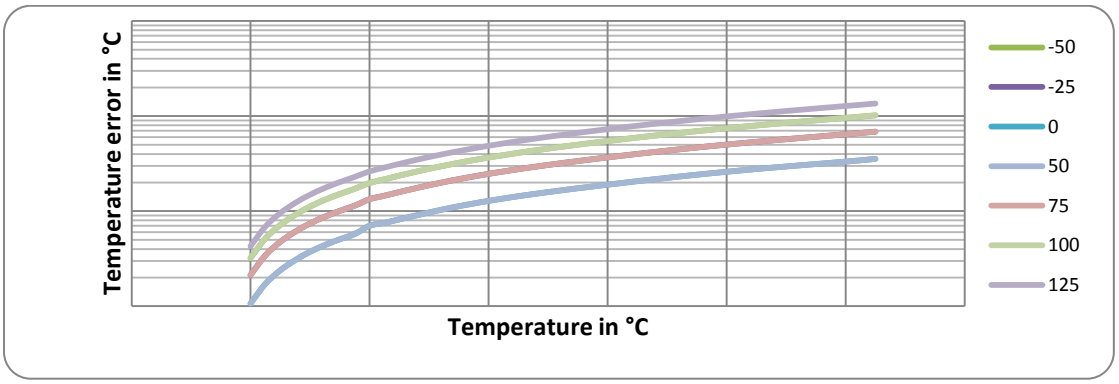
The orange line shows that for a 0.1 percent reference resistor, the error is <1.3 °C for an ambient temperature of 25 °C and RTD temperatures from -200 °C to 850 °C.

The error contributed by the reference resistor can be calibrated out using two methods. For the first method, measure the value of the reference resistor at 25 °C in the factory using an accurate ohmmeter, and then store it in the PSoC EEPROM. Use this value (not 100 Ω) for R_{ref} in Equation 8.

For the second method, replace the RTD with a known precision resistor. Measure the value of the resistor. Compute the ratio of this measured resistance against the actual resistance. Store this ratio in flash. Multiply all subsequent resistances by this ratio. This method is described in more detail in RTD Calibration.

By calibrating out the initial tolerance of the reference resistor, the remaining error is only due to the temperature coefficient (see Figure 13).

Figure 13. Temperature Error Due to Reference Resistor Temperature Tolerance



The temperature coefficient of the reference resistor still causes an error if the ambient temperature is much higher or lower than 25 °C. The ambient temperature can be measured using the on-chip temperature sensor in PSoC devices. and the error due to the temperature coefficient of the reference resistor can be corrected.

For this method, the reference resistor needs to be measured at manufacturing at several temperatures. This approach will not be explained any further.

8.2.2 RTD Error

This is divided into two factors:

1. RTD interchangeability error
2. RTD self-heating error

RTD Interchangeability Error

This error is caused by replacing one RTD with another one of the same part number (process variation). This is defined in RTD datasheet. IEC 60751 defines two primary classes of RTD tolerances, as shown in Table 3.

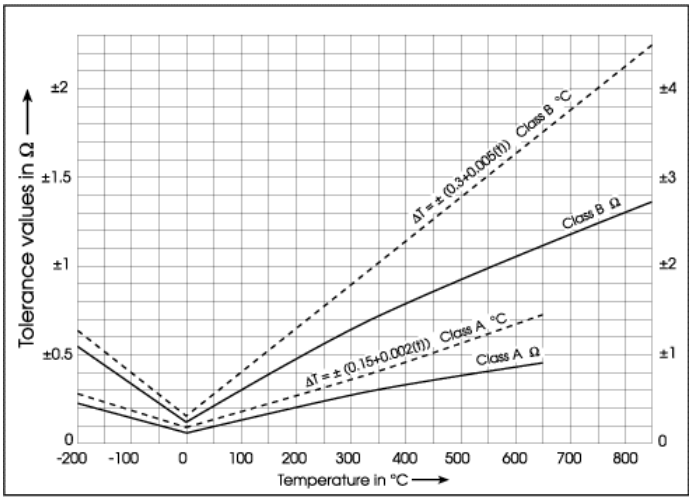
Table 3. RTD Tolerance Classes

Tolerance Class	Tolerance °C)
A	0.15 + 0.002 |t|
B	0.30 + 0.005 |t|

At no point does the temperature error of a class B RTD exceed the value 0.3 + 0.005 |t|, Where |t| = Absolute value of temperature in °C.

Figure 14 shows the graph of RTD tolerance. The errors at 25 °C and 800 °C due to class B RTD are calculated below.

Figure 14. Temperature Error due to RTD Interchangeability Error



At 25 °C, the worst-case temperature error = 0.3 + 25 * 0.005 = 0.43 °C.

At 800 °C, the worst-case temperature error = 0.3 + 800 * 0.005 = 4.3 °C.

If this error is not acceptable, the RTD must be calibrated. For a high-performance RTD (0.1% or 1 °C), calibration is required.

RTD Calibration

For RTD calibration, perform the following steps:

1. Make sure that all other sources of error are nulled. Offset should be nulled by correlated double sampling, and the gain error is automatically nulled by the ratio.
2. Adjust the RTD to a known temperature, T_1, for this method. T_1 should be close to 0 °C. Validate and measure T_1 with an accurate thermometer.
3. Measure the RTD resistance, R_{meas}, at T_1.
4. Calculate the actual RTD resistance at T_1, R_{actual}, using Equations 1 and 2. The scale error is given by Equation 14.

   Equation 14 $$scale = \frac{R_{actual}}{R_{meas}}$$

5. Store the scale error in EEPROM.
6. Multiply all measured RTD resistances by this scale.

If an accurate and stable temperature cannot be achieved, replace the RTD with a known resistor. Using a known resistor calibrates out only the error due to the reference resistor. It does not calibrate out the RTD interchangeability error.

Note that the reference resistance tolerance also causes a fixed scale error in the measured resistance. Hence, when the above six steps are completed, the reference resistance is also calibrated. The scale error value can be stored in PSoC EEPROM and retrieved each time the device goes through a power cycle.

CE210434 demonstrates how this type of calibration is done. Figure 14 and Table 3 show that the RTD error has an offset and a gain. Thus, it may become necessary to perform a two-point temperature calibration. For example, place the RTD at 0 °C and measure the temperature (T_{offset}). Subtract this measured temperature from subsequent temperature readings.

Second, place the RTD at 100 °C and measure the temperature (T_{Gain}). Next, compute a scale factor using the equation.

Scale = T_{Gain} – T_{offset} / 100.

The 100 comes because you measured at 0 °C and 100 °C, for a difference of 100. Now multiply all temperatures by this scale factor. This method is not demonstrated.

RTD Self-Heating Error

An RTD self-heating error is the increase in temperature of the RTD due to the current flowing through the RTD.

As a result of self-heating, the RTD can show a temperature slightly higher than ambient. This error can be found in the RTD datasheet. For RTD on the CY8CKIT-025, the value is specified at <= 0.8 °C/mW.

At 150 °C, the RTD resistance is 157.325 Ω. If 1-mA current is passed, the power dissipation is 390 µW or 0.39 mW. This corresponds to RTD self-heating of <0.13 °C. Note that the power dissipation constant of SMD RTDs are generally higher (about 1 °C/mW).

8.3 List of all Errors

Table 4 shows the temperature error due to various components at 150 °C. As seen from the table, RTD interchangeability and reference resistance tolerance are the biggest sources of error. By comparison, the other errors are negligible.

Table 4. Possible Errors in RTD Temperature Measurement at 150 °C

Error Source	Error Value at 150 °C (0.1% Reference Resistor, class B RTD)	Error Value at 150 °C (Both Reference Resistor and RTD Calibrated)
Signal Chain Error		
Offset Error/drift	0 °C	0 °C
Gain Error/drift	0 °C	0 °C
ADC INL*	0.2 °C	0.2 °C
RTD self-heating error (PTS080501B100RP100 SMD RTD)	< 0.13 °C	<0.13 °C
Error due to reference resistor (Ambient Temperature = 25 °C)	0.43 °C	Limited only by calibration accuracy and reference resistor temperature coefficient(very accurate)
Error due to reference resistor (Ambient Temperature = 80 °C)	0.6 °C	~0.2 °C unless temperature calibration is completed
Sensor Error		
Error due to RTD interchangeability (Class B RTD)	1.05 °C	Limited only by calibration accuracy (very accurate)
Polynomial fit error (fifth-order polynomial)	0.0003 °C	0.0003 °C

Note(*): worst-case INL is used at the numerator. Typically, the error is < 0.1 °C, Also this is only for PSoC 3 and PSoC 5LP.

8.4 Test Results

An RTD was simulated by a potentiometer and the signal chain accuracy was tested in the whole temperature range of the RTD. A potentiometer was connected in the external RTD slot of CY8CKIT-025 and the temperature shown in the LCD was noted. The potentiometer resistance was measured by a precision multimeter and the resistance value was noted. The resistance was converted into temperature manually using a fifth-order polynomial. The test results shown in Table 5 indicate that the RTD signal chain is highly accurate.

Table 5. PSoC 5LP RTD Temperature Measurement Test Results

Resistance Value (Ω)	Expected Temperature (°C)	Measured Temperature (°C)	Error (°C)
27.285	−179.6	−179.6	0.0
32.47	−167.3	−167.4	0.1
66.58	−84.3	−84.4	0.1
80.21	−50.2	−50.3	0.1
118.296	47.1	47	0.1
149.464	129.0	128.9	0.1
218.374	317.9	317.8	0.1
325.314	636.3	636.3	0.0

9 Summary

When high accuracy is critical in measuring temperatures, an RTD is the sensor of choice. PSoC 3 and PSoC 5LP have the necessary hardware integrated in the device to achieve high accuracy. The PSoC Creator RTD Component makes designing with RTDs easier.

10 Related Resources

10.1 Related Application Notes

- AN2099 – PSoC® 1, PSoC 3, PSoC 4, and PSoC 5LP - Single-Pole Infinite Impulse Response (IIR) Filters
- AN58827 – PSoC® 3 and PSoC 5LP Internal Analog Routing Considerations
- AN60590 – PSoC® 3, PSoC 4, PSoC 5LP and PSoC Analog Coprocessor – Temperature Measurement with a Diode
- AN65977 – PSoC 3 and PSoC 5LP - Creating an Interface to a TMP05/TMP06 Digital Temperature Sensor
- AN66477 – PSoC® 3, PSoC 4, PSoC 5LP and PSoC Analog Coprocessor - Temperature Measurement with a Thermistor
- AN66444 – PSoC 3 and PSoC 5LP Correlated Double Sampling to Reduce Offset, Drift, and Low Frequency Noise
- AN75511 – PSoC® 3 / PSoC 5LP - Temperature Measurement with a Thermocouple
- AN211294 – AFE Implementation Using PSoC® Analog Coprocessor

10.2 Related Code Examples

- CE210383 – PSoC 3, PSoC 4, PSoC 5LP and PSoC Analog Coprocessor - Temperature Sensing with an RTD
- CE210434 – PSoC 3 and PSoC 5LP RTD Calibration
- CE210435 – PSoC 3 and PSoC 5LP Broken RTD Reconfiguration

About the Author

Name: Praveen Sekar
Title: Applications Engineer
Background: Praveen holds a Bachelor's degree in Electronics and Communication from the College of Engineering, Guindy, Chennai. He focuses on analog modules in PSoC

Name: Todd Dust

Title:	Applications Engineer Sr. Staff
Background:	Todd holds a Bachelor's degree in Electrical Engineering from Seattle Pacific University.

Appendix A. Broken RTD Reconfiguration

If one of the four wires of the RTD breaks, PSoC 3 and PSoC 5LP can automatically detect a broken wire, reconfigure a four-wire RTD connection to a three-wire connection, and measure temperature with minimal degradation to accuracy.

Broken RTD reconfiguration involves three steps.

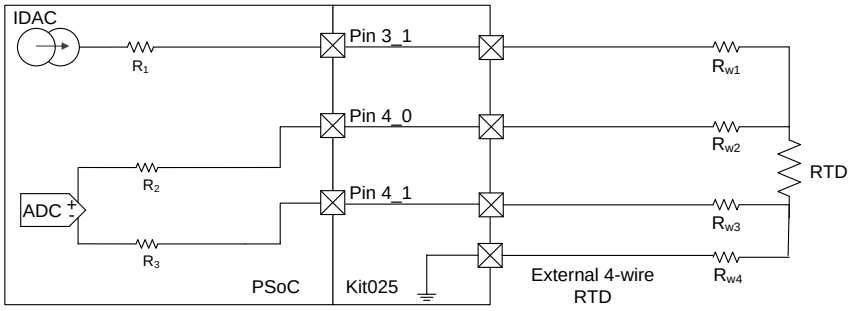
1. Detect broken wire connection
2. Reconfigure the analog routing to change four-wire connection to three-wire connection
3. Compensate for the additional wire resistance due to the three wire mode and measure temperature.

The PSoC Creator project (Broken RTD Reconfiguration) found in CE210435 demonstrates this behavior. It is only available with PSoC 3 and PSoC 5LP.

A.1 Detecting Broken RTD Wire

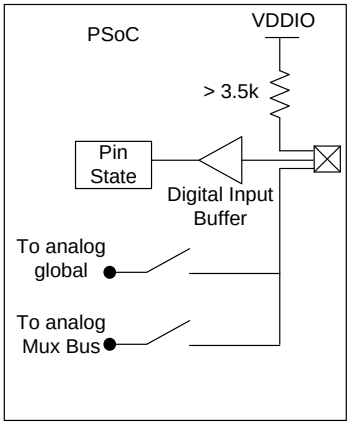
Figure 15 shows a four-wire RTD connection to PSoC. The current is passed through pin 3_1 into RTD wire 1 and it is grounded through RTD wire 4 (RTD wire 4 is not connected to a PSoC pin). The ADC differential inputs are connected to pins 4_0 and 4_1, which are connected to wires 2 and 3 of the RTD. The pin choices are made according to the connections in CY8CKIT-025 PSoC precision analog temperature sensor EBK.

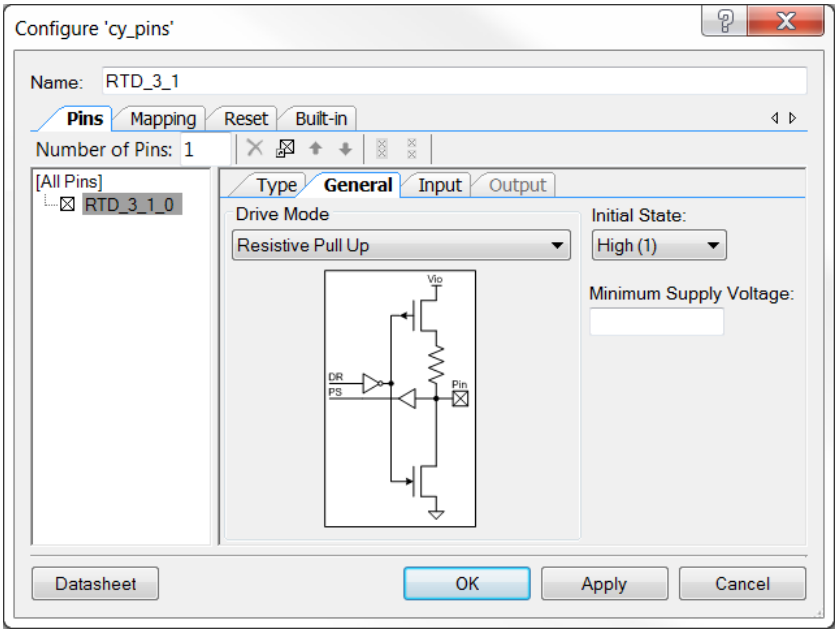
Figure 15. Four-Wire RTD Connected to PSoC



A broken wire can be detected by using PSoC's GPIO structure. PSoC GPIO can be configured to source V_{DDIO} through a pull up resistor while simultaneously sensing the pin state through its digital input buffer, as Figure 16 shows.

Figure 16. Pin Configured in Resistive Pull-Up Mode and Digital Input Mode



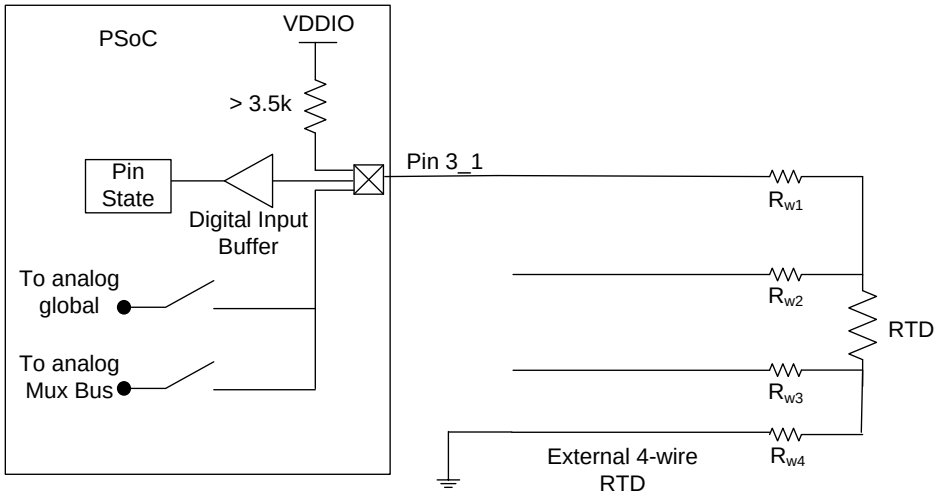


To detect the broken RTD wire connected to pin 3_1, configure pin 3_1 to the resistive pull-up mode, and the pin state is sensed back see Figure 17.

When wires 1 and 4 are not broken, the RTD resistance forms a resistor divider with the internal pull-up resistor. The voltage across the RTD is sensed back as the pin state.

The pull-up resistance has a minimum value of 3.5 kΩ, and the RTD can have a maximum resistance of 390 Ω (at 850 °C). Assuming wire resistances (R_{w1} and R_{w4}) = 5 Ω each, we get a maximum value of 400 Ω.

Figure 17. Detecting Wires 1 and 4 for Breakage



Voltage sensed by pin = Voltage across RTD+R_{w1}+R_{w4}

$$= VDDIO * \frac{R_{RTD} + R_{w1} + R_{w2}}{R_{RTD} + R_{w1} + R_{w2} + R_p}$$

$$= VDDIO * \frac{400}{3900}$$

$$< 0.1 * VDDIO$$

VIL of the pin < 0.3 * VDDIO

Therefore, when pin 3_1 is configured as resistive pull up (with a high voltage forced through the pin) and when no RTD wire is broken, the pin state is low. When either RTD wire 1 or 4 is broken, the pin state is high.

Similarly, we can detect if RTD wires 2 and 3 are broken by configuring the respective pins to resistive pull-up modes and reading the pin state back.

To find which RTD wire is broken, follow these steps:

1. Disconnect pins 3_1, 4_0 and 4_1 from ADC and DAC.
2. Configure pin 3_1 in resistive pull up mode.
3. Drive high through pin 3_1.
4. Read the pin state of pin 3_1.
5. Repeat steps 2, 3, and 4 for pin 4_0 and pin 4_1.

Let the pin states of pin 3_1, pin 4_0 and pin 4_1 be stored in variables A, B, and C respectively. Based on different values of A, B, and C, we can have eight states, as shown in Table 6.

Table 6. Wire State

A	B	C	Result
0	0	0	No Wire Broken
0	0	1	Wire 3 broken
0	1	0	Wire 2 broken
0	1	1	Wires 2 and 3 broken
1	0	0	Wire 1 broken
1	0	1	Wires 1 and 3 broken
1	1	0	Wires 1 and 2 broken
1	1	1	Wire 4 broken

The table also shows the result of each combination of A, B, and C.

If any of A, B and C is equal to 0, then wire 4 is not broken.

If A, B and C are all equal to 1, wire 4 is definitely broken. Apart from wire 4 any other wire can also be broken. But in such a case reconfiguration is not possible. Reconfiguration is possible only if one of the wires is broken.

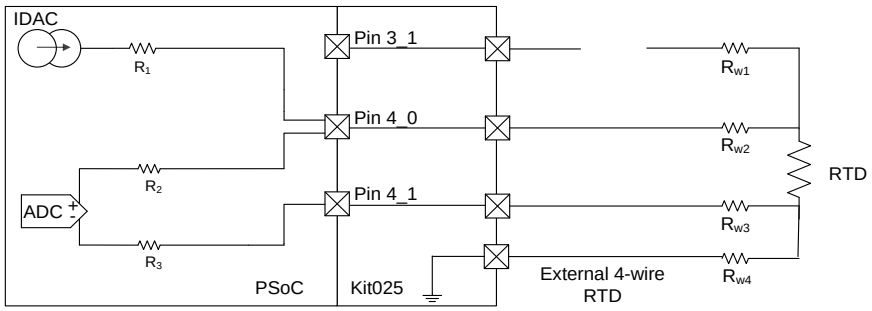
A.2 Reconfiguring Four-Wire RTD to Three-Wire

After detecting the broken RTD wire, we must reconfigure the four-wire RTD to three-wire RTD, eliminating the broken wire. The flexible analog routing structure of PSoC 3 and PSoC 5LP makes the reconfiguration easy to do. The reconfiguration routes are: RTD Wire 1 Broken, RTD Wire 2 Broken, RTD Wire 3 Broken, and RTD Wire 4 Broken.

A.2.1 RTD Wire 1 Broken

If RTD wire 1 is broken, the current path is opened. To close the current path, the routing is reconfigured such that the current is forced through the ADC pin 4_0 as shown in Figure 18.

Figure 18. RTD Wire 1 Broken



R1 = Routing resistance from IDAC to pin

R2 = Routing resistance from ADC (positive) to pin

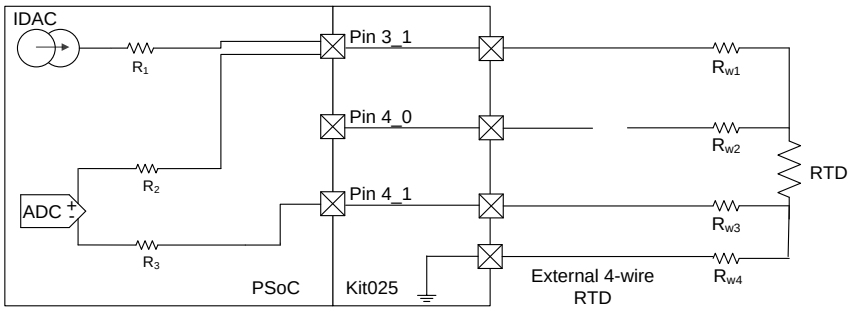
R3 = Routing resistance from ADC (negative) to pin

In this case, the IDAC current flows through R_1, R_{w2}, through RTD to ground. Since R_{w2} is in the measurement path of the ADC, the RTD resistance and wire 2 resistance is also measured. This wire resistance can be eliminated through calibration explained in the one-time wire resistance computation section below.

A.2.2 RTD Wire 2 Broken

If RTD wire 2 is broken, the path from RTD to ADC positive terminal is opened. To close the path, we connect ADC positive terminal to pin 3_1 as shown in Figure 19.

Figure 19. RTD Wire 2 Broken

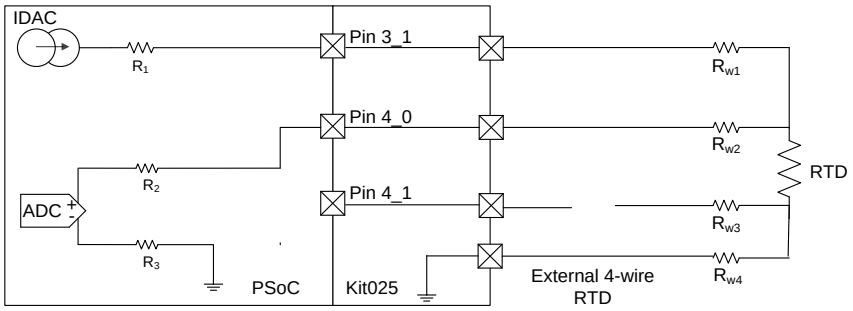


In this case, the ADC measures RTD wire resistance 1 in addition to the RTD resistance.

A.2.3 RTD Wire 3 Broken

If RTD wire 3 is broken, the path from RTD to ADC negative terminal is opened. To close the path, we connect ADC negative terminal to ground as shown in Figure 20.

Figure 20. RTD Wire 3 Broken

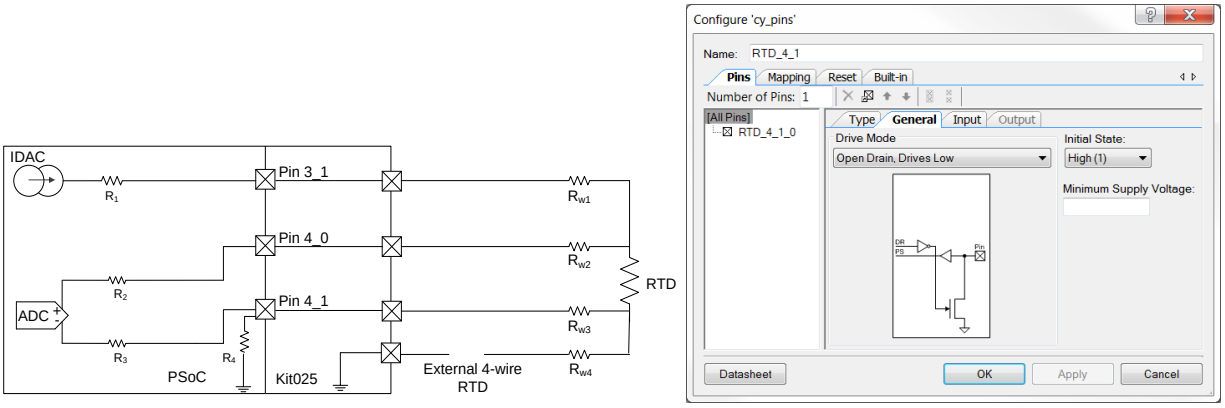


In this case, the ADC measures RTD wire resistance 4 in addition to the RTD resistance. Also, any difference in potential between the two grounds (kit-025 ground and the chip internal ground) adds to measurement error. One-time offset correction eliminates both the wire resistance error and the ground difference error.

A.2.4 RTD Wire 4 Broken

If RTD wire 4 is broken, the current path from RTD to ground is opened. The ADC input terminal is Hi-Z and no current flows through the ADC input. To close the path, we provide the ground path by configuring the pin in open drain low mode as shown in Figure 21.

Figure 21. RTD Wire 4 Broken



In this case, the ADC measures RTD wire resistance 3 in addition to the RTD resistance.

A.3 One-Time Wire Resistance Computation

When reconfiguring a four-wire RTD into a three-wire RTD, wire resistances affect the RTD temperature measurement accuracy. 1-ohm wire resistance can cause 3 °C error in measured temperature. To eliminate the error due to the wire resistances, perform one-time wire resistance computation before making the RTD measurements.

The following steps are used to compute the wire resistances:

1. Configure the RTD in four-wire mode as shown in Figure 15 and find the RTD resistance (R_0)
2. Configure the RTD in three-wire mode (wire 1 broken) as shown in Figure 18 and calculate the resistance (R_1)
3. Compute additional wire resistance, CompRes1 = ($R_1 – R_0$). When wire 1 breaks and the RTD is reconfigured as shown in Figure 18, CompRes1 should be subtracted from the measured resistance.
4. Configure the RTD in three-wire mode (wire 2 broken) as shown in Figure 19 and calculate the resistance (R_2).
5. Compute additional wire resistance, CompRes2 = ($R_2 – R_0$). When wire 2 breaks and the RTD is reconfigured as shown in Figure 19, CompRes2 should be subtracted from the measured resistance.
6. Configure the RTD in three-wire mode (wire 3 broken) as shown in Figure 20 and calculate the resistance (R_3)
7. Compute additional wire resistance, CompRes3 = ($R_3 – R_0$). When wire 3 breaks and the RTD is reconfigured shown in Figure 20, CompRes3 should be subtracted from the measured resistance.
8. Configure the RTD in three-wire mode (wire 4 broken) as shown in Figure 21 and calculate the resistance (R_4)
9. Compute additional wire resistance, CompRes4 = ($R_4 – R_0$). When wire 4 breaks and the RTD is reconfigured as shown in Figure 21, CompRes4 should be subtracted from the measured resistance.

CE210435 provides an example of how all of this is done.

Document History

Document Title: AN70698 - PSoC® – Temperature Measurement with an RTD

Document Number: 001-70698

Rev	ECN	Orig. of Change	Submission Date	Description of Change
**	3458038	PFZ	12/12/2011	New Application note
*A	3490797	PFZ	01/12/2012	MEH Review Feedback in CDT#116240
*B	3520653	PFZ	02/08/2012	Updated project. No change to document
*C	3689958	PFZ	08/09/2012	RTD component includes support for PT100, PT500 and PT1000 RTDs A new section on broken RTD reconfiguration has been added Other minor changes
*D	3740378	PFZ	09/11/2012	Updated associated project files.
*E	3818484	PFZ	11/21/2012	Updated title to read as “PSoC® 3 and PSoC 5LP – Temperature Measurement with an RTD”. Updated Associated Part Family as “All PSoC 3 and PSoC 5LP Parts”. Updated Related Application Notes as “AN75511, AN66477, AN60590”. Updated Introduction. Updated RTD – Resistance-to-Temperature Conversion (Updated Choosing the Right Polynomial Order (Updated description), updated RTD Component (Updated Figure 6, Figure 7, Figure 8) Updated Project Description (Updated Figure 11 and Figure 9) Updated Appendix B (Updated Broken RTD reconfiguration (Updated Project Description (Updated Figure 25))). Updated Replaced PSoC 5 with PSoC 5LP in all instances across the document.
*F	4057734	TDU	07/11/2013	Added two projects and discussed different performance ranges.
*G	4152296	TDU	10/09/2013	Updated attached Associated Project. Completing Sunset Review.
*H	4202789	TDU	11/26/2013	Fixed formatting errors.
*I	5075960	TDU	01/07/2016	Moved Example Projects to CEs. Updated to latest template. Added PSoC 4.
*J	5698367	AESATP12	04/17/2017	Updated logo and copyright
*K	5741154	JSLN	05/18/2017	Changed title Added support to PSoC Analog Coprocessor Updated template

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2011–2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.