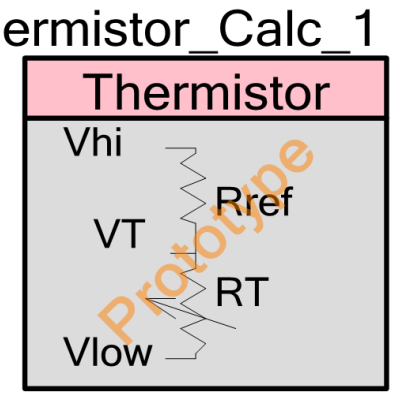
Thermistor Calculator

V 1.1



Features

- Adaptable for any NTC thermistors
- Constant voltage type of measurement
- Implementation methods
  - LUT
  - Equation
- Selectable reference resistor, based on thermistor value
- Selectable temperature range and accuracy for LUT method
- Interchangable R_{ref} and R_T position
  - R_T connected between V_T and V_{low} : Voltage V_T decreases with temperature
  - R_T connected between V_{hi} and V_T: Voltage V_T increases with temperature
- The output temperature is an integer scaled up by 100.
- The maximum accuracy is ± 0.01 ℃

General Description

The Thermistor component provides the temperature, based on the voltage of the thermistor. The component is adaptable to any Negative Temperature Coefficient(NTC) thermistors. The adaptability is due to the fact that the external reference resistor, temperature range and corresponding resistances are parameters to the component. Based on these parameters, the co-efficients required for the specific thermistor are calculated by the component. The code required for the conversion from voltage to temperature is generated in the component.

Input/Output Connections

The temperature from the thermistor component is obtained through a function call. The input and output connections are not directly to the component. The system level overview provides the input connections that are passed to the component through a function call.

System Level Overview

The overall project requires the connection of an external resistor and thermistor to PSoC. The voltages of the externally connected resistor and thermistor are measured by using an ADC and the values are passed to the thermistor component through an API call. The return value of this API call is the temperature. This method of excluding the ADC from the component makes the ADC available for other functions in a project. The block diagram for the overall system is provided in Figure 1.

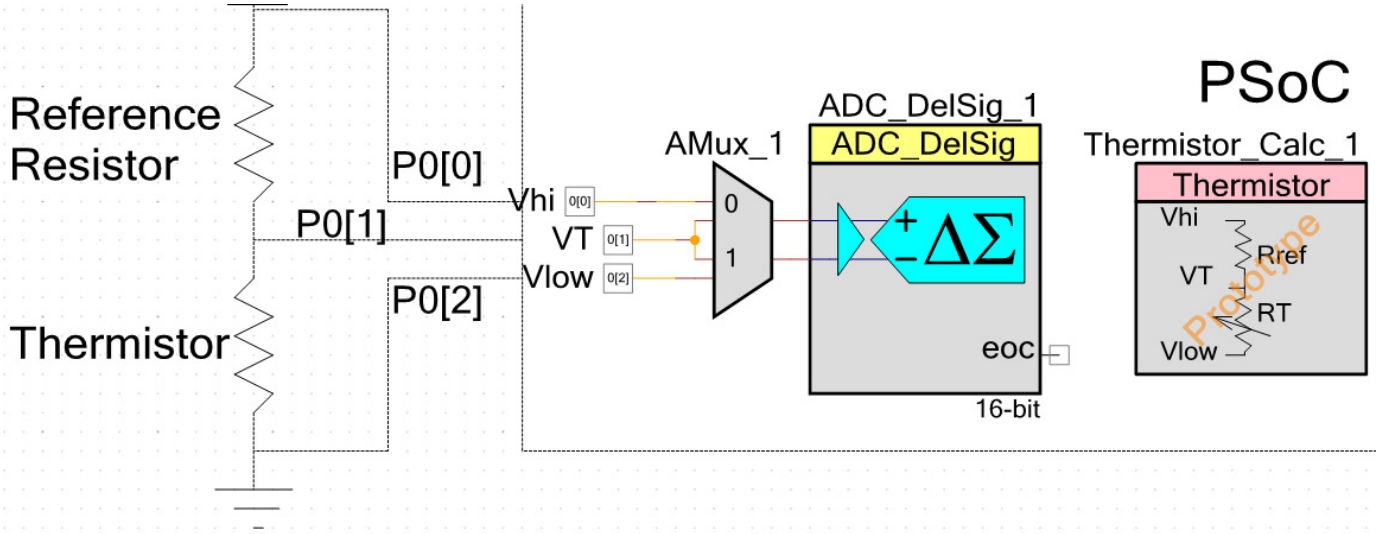


Figure 1. Block Diagram of the thermistor based temperature monitor system

A constant voltage V_{hi} is applied to the resistor and thermistor combination shown in Figure 1. The resistance of the thermistor changes with change in temperature, thus the voltage drop across the thermistor changes with temperature. The voltage drop across the thermistor and resistor are measured, and the resistance of the thermistor is found using Equation 1.

$$R_{Thermistor} = R_{reference_resistor}\left(\frac{V_T - V_{low}}{V_{hi} - V_T}\right) \quad \text{Equation 1}$$

The temperature is then determined based on the resistance, either through the Equation or LUT method. The temperature is obtained in Equation method by directly using the Steinhart-Hart Equation shown below

$$\frac{1}{T_K} = A + B * ln(R_{Thermistor}) + C * (ln(R_{Thermistor}))^3$$ Equation 2

Where A, B and C are Steinhart-Hart coefficients that are calculated by the customizer software. The coefficients are obtained by solving three simultaneous equations formed by substituting the "Thermistor Parameters", provided in Figure 2, into Equation 2.

In case of the LUT method, the table relating the temperatures and resistances is generated by the customizer software and stored in the program memory. To generate the LUT, the resistances for temperatures within the range are calculated using Equation 3. The resistances are calculated starting with "Min Temp" and increments of "Accuracy" upto "Max Temp", specified by the user. A look-up-table is formed with these resistances by the customizer software and stored in the program memory. The temperature corresponding to the specific resistance measured is then obtained from the table, during runtime.

$$R_{Thermistor} = e^{\left[(\beta-(\alpha/2))^{\frac{1}{3}} - (\beta+(\alpha/2))^{\frac{1}{3}}\right]}$$ Equation 3

Where

$$\alpha = \frac{\left(A - \left(\frac{1}{T}\right)\right)}{C}$$

$$\beta = \left[\left((B/3C)^3\right) + \left(\left(\frac{\alpha^2}{4}\right)^{\frac{1}{2}}\right)\right]$$

These equations are provided in this document user reference. The customizer makes all these calculations and provides the required temperature. Further details and examples are provided in application note – *"Temperature Measurement with Thermistor - PSoC 3 and PSoC 5".*

Parameters

The customizer for providing the parameters of the thermistor is provided in Figure 2.

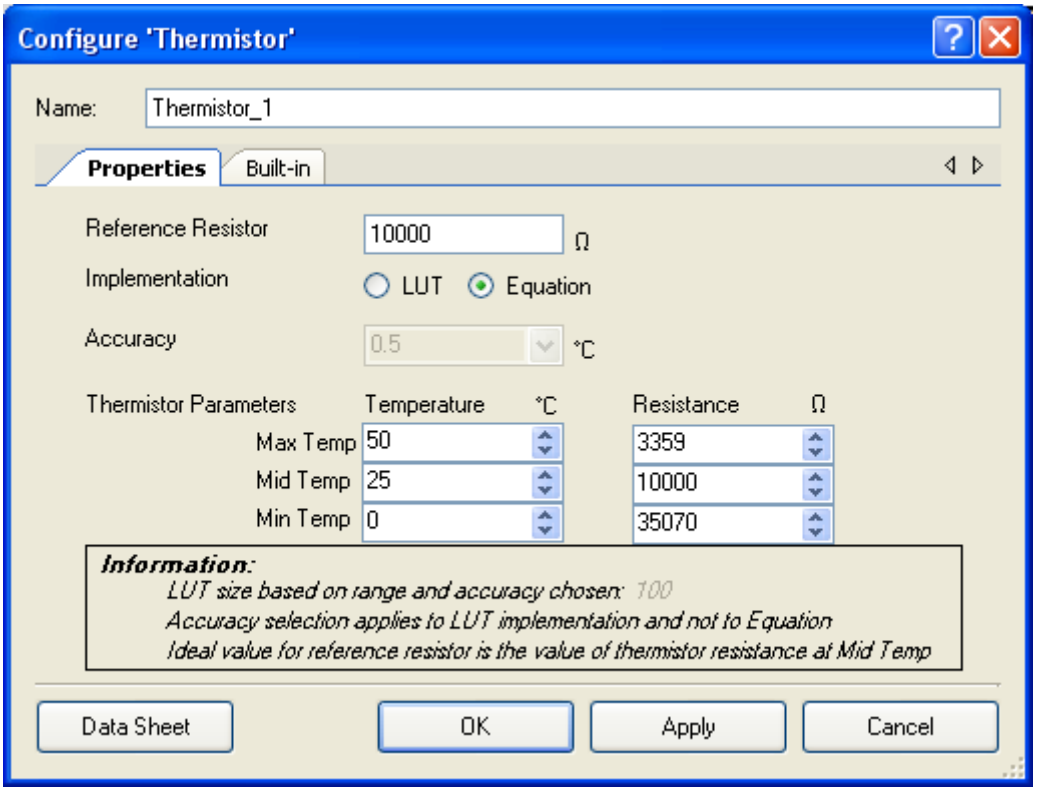


Figure 2. Parameter editing customizer for thermistor component

Reference Resistor

The reference resistor is connected with the thermistor, as shown in the symbol, for the constant voltage type of temperature measurement. The R_{ref} and R_T can be interchanged to get either increasing or decreasing voltage values with increasing temperatures. Ideally the value of the reference resistor should be equal to the value of the thermistor at the middle of the temperature range required. This leads to a comparatively linear V vs. T curve for the temperature, as shown in Figure 3. Non-linear curve implies that the required resolution of the ADC is different to at the two ends of temperature range, to obtain the same accuracy. Thus choosing the reference resistor value to be equal to the value of the middle of the range provides uniformity in accuracy of measurement.

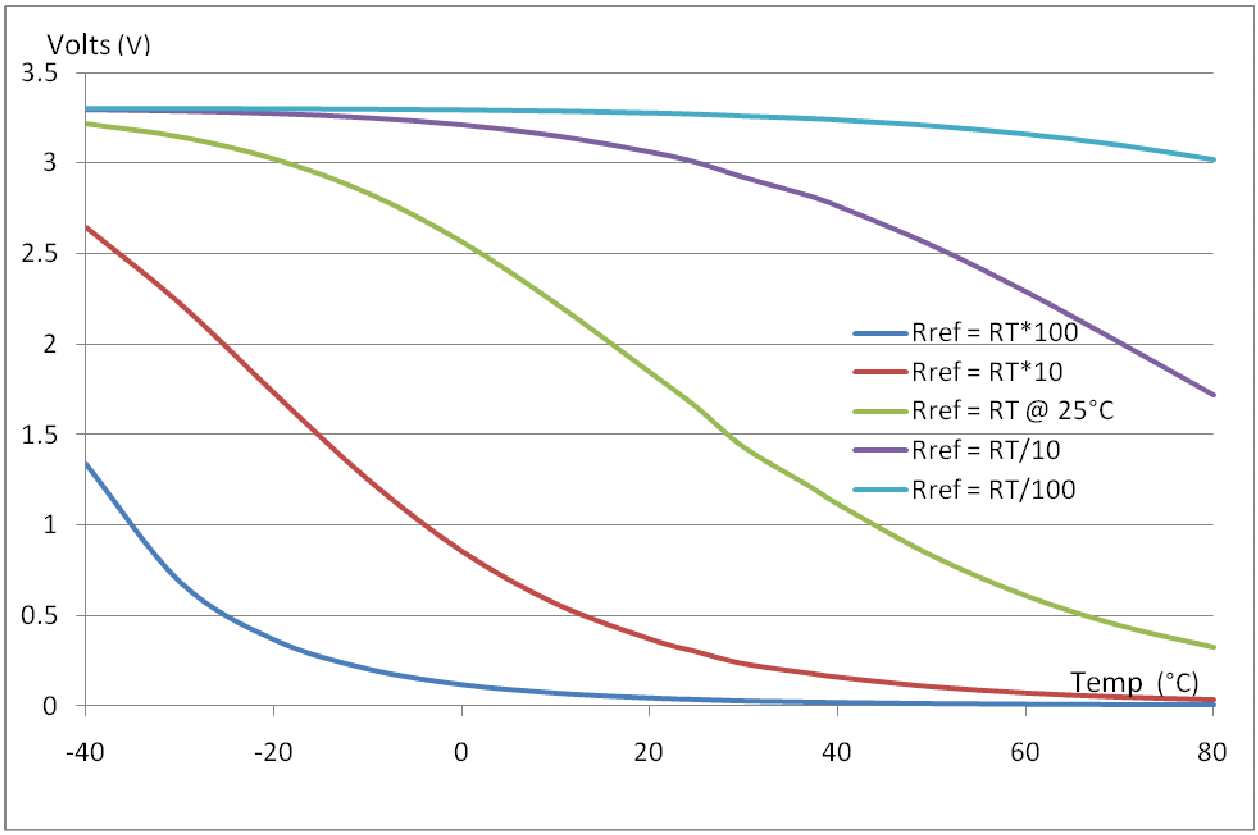


Figure 3. Output voltage variation with temperature across different reference resistor values

Implementation

The method of obtaining the temperature can be through a Look-Up-Table (LUT) or Equation. The tradeoffs between the two methods are memory, speed, range and accuracy. The LUT will use less memory and has faster response time, as compared to Equation method. The Equation method is more accurate and has fixed range and accuracy. The equation method uses more memory because it requires the floating point math library. If the project already uses the floating point math library, then it is beneficial to use the equation method since there is no added memory over-headed. The comparison between the two methods is provided in the table 1

Table 1. Tradeoffs in the two methods of implementations

	Equation	LUT	Comments
Accuracy (+/-)	> 0.01	≤ 0.01	Accuracy shown is the accuracy of calculation alone. This excludes the accuracy of thermistor, reference resistor, voltage reference or ADC. Though the accuracy of equation is better than ± 0.01 °C, the output is limited to resolution of ±

			0.01 ℃, since it is scaled by 100.
Memory usage	Higher if FP* not included	Lower if FP not included	The memory usage of Equation is fixed and is due to the floating point library. If the floating point is already used by other components or functions, then the Equation method is efficient. The memory usage of LUT depends on the range and accuracy chosen
	Lower if FP included	Higher if FP included	
Range	Wider than specified	Limited to specified	In Equation the temperature can be measured outside of the range specified (lower accuracy). In LUT the temperature values outside the range specified will not be measured.

Accuracy

The accuracy of the temperature measured in the LUT method is determined by this parameter. The step size between the temperatures in the LUT is equal to accuracy specified. This is parameter selection is relevant only to the LUT method of implementation and not Equation method.

The accuracy provided by this parameter is the accuracy of the temperature measurement. This implies that the accuracy corresponds to conversion of the voltage to temperature. It does not account for the other inaccuracies in the system, like the tolerance of the reference resistor, variation of the reference voltage and the accuracy of the ADC. Assuming an accurate voltage measurement, this parameter provides the accuracy of the temperature output of the component.

Thermistor Parameters

The temperature and corresponding resistance parameters are entered into the “Thermistor Parameters”, shown in Figure 2. The co-efficients (Steinhart co-efficients) required for obtaining the temperature are calculated from these parameters. These parameters also determine the temperature range for the LUT method. The highest and the lowest temperature values entered in these parameters form the starting and the ending values of the LUT.

Note: If the thermistor parameters are entered incorrectly, the LUT can be filled with zeros in LUT method of implementation. The co-efficients in the Equation method, corresponding to incorrect parameters, will be NaNs. These parameters should be entered correctly based on the chosen thermistor datasheet.

Information

The LUT size is displayed in the first line and it is limited to 2000 steps. The LUT size is determined by Equation 4. As shown in the equation, the LUT size depends on the range and accuracy. Thus, when the LUT size exceeds the limit, an error is provided next to the accuracy, that either the accuracy or the range has to be lowered.

$$LUT_SIZE = Accuracy * (MAX_TEMP - MIN_TEMP) \quad \text{Equation 4}$$

Invalid parameters

An error message, as shown in Figure 4, is displayed when LUT size exceeds the limit (2000 entries). The actual error message “LUT cannot be greater than 2000. Lower the range or accuracy” can be noticed when the cursor is scrolled over the error symbol.

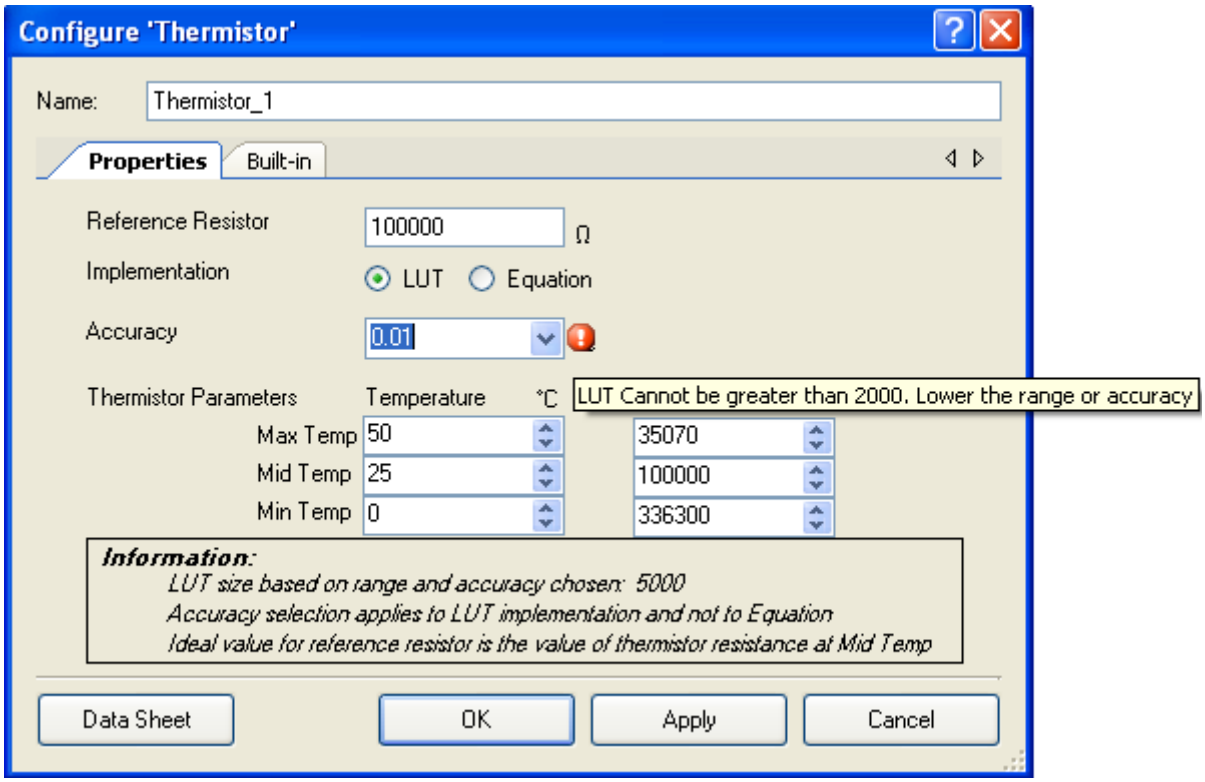


Figure 4. Invalid settings for range and accuracy in LUT mode

When invalid combinations of temperature vs. resistance are entered into the Thermistor parameters, the project will provide errors during execution/build of the project. In case of LUT method, the table is filled with zeros for all temperature values. In case of the Equation method, the Steinhart co-efficients are stored as NaNs. Care should be taken to enter the correct combination of temperature and resistance parameters from the datasheet of the thermistor.

Placement

The is no placement specific information for the thermistor component

Resources

This is a completely firmware based component. Thus the analog and digital resources are not used. The only resource used by the component is the Flash and RAM and usage is provided in Table 2

Table 2. Comparison of memory used in the two methods of implementation

	Flash (bytes)	RAM (bytes)
Equation	2246	30
LUT (Size = 100)	796	19

The Flash used for the Equation will remain the same for any range of temperatures, but the amount used for the LUT increases linearly with increase in LUT size. The Flash usage in Equation method is due to overhead of floating point library. If the floating point library is already used in the target project, the Equation method will should be used.

The resources required external to the component, to complete the temperature measurement are provided below:

- Analog to Digital Convertor
- Analog MUX

Application Programming Interface

Application Programming Interface (API) routines in the Thermistor component provide the interface to pass the voltage values to the component. The API returns the temperature, as either a floating point number or a long integer, based on the method of implementation. There are no hardware resources in the firmware and thus no Start or Stop functions.

int32 Thermistor_1_GetResistance(int32 Vreference, int32 VThermistor)

Description: The digital values of the voltages across the reference resistor and the thermistor are passed to this function as parameters. These can be considered as the inputs to the component. The function returns (outputs) the resistance, based on the voltage values.

Parameters: Vreference is the voltage across the reference resistor.
Vthermistor is the voltage across the thermistor.

Return Value: The return value is the resistance. The return type is 32-bit integer as shown in the function prototype provided above. The value returned is the scaled version of temperature. The value is scaled up by 100 to avoid use of floating point math in the calculations. For example, the return value is 2345, when the actual temperature is 23.45

int32 Thermistor_1_GetTemperature (int32 ResT)

Description:	The digital values of the voltages across the reference resistor and the thermistor are passed to this function as parameters. These can be considered as the inputs to the component. The function returns (outputs) the resistance, based on the voltage values.
Parameters:	ResT is the resistance across the thermistor.
Return Value:	The return value is the temperature. The return type is 32-bit integer as shown in the function prototype provided above. The value returned is the scaled version of temperature. The value is scaled up by 100 to avoid use of floating point math in the calculations. For example, the return value is 2345, when the actual temperature is 23.45

Sample Firmware Source Code

The following is a C language example demonstrating the basic functionality of the Thermistor component. This example assumes the component has been placed in a design with the default name "<InstanceName>_1."

Note If you rename your component you must also edit the example code as appropriate to match the component name you specify.

```
void main()
{
      int32 iVref, iVtherm;
      int32 iTemp;
/*Insert code to obtain the voltage across reference resistor iVref*/
/*Insert code to obtain the voltage across thermistor iVtherm*/
      iTemp = Thermistor_1_GetTemperature(iVref, iVtherm);
}
```

DC Characteristics

The accuracy of the temperature measured with the thermistor is dependent on various steps:

1. Accuracy of the thermistor chosen
2. Variation of the reference resistor
3. Accuracy of the voltage reference
4. Measurement accuracy of the buffer and ADC
5. Calculation accuracy of the thermistor component

The first four dependencies are outside the thermistor component. The accuracy of the calculation is provided below:

Equation

In the equation method the accuracy is better than ± 0.01 °C, for the temperature range specified in the customizer. The coefficients are calculated based the parameters provided and the

corresponding code is generated. The output temperature, of the Equation method, is multiplied by 100 and returned as an integer. This is to maintain consistency in return value of the function call from the two methods. Thus the resolution in the output is only ± 0.01 °C

LUT

The accuracy of the measurement can be chosen in the LUT method. The different options available are provided in Table 3. The constraint is the size of the LUT ≤ 2000. Thus the temperature range is limited, as shown in Table 3, for different accuracies that can be chosen with the LUT method.

Table 3. Accuracy and the temperature ranges in the LUT method

Accuracy (°C)	Temperature Range (Max T - Min T)
0.01	20
0.05	100
0.1	200
0.5	1000
1	2000
2	4000

Reference Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes
1.1	New component and datasheet

© Cypress Semiconductor Corporation, 2012. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark , and PSoC Creator™, and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

Document Revision History

The Document History Page is only for approval of this template and must be deleted.

Document Title: Thermistor Component Datasheet **Document Number:**			
Revision	**ECN#**	**Origin of Change**	**Description of Change**
**		YARA	New Thermistor Component Data Sheet
Distribution: Internal			
Posting: None			