Using Custom Thermistors with the Temp-to-Bits Family

The LTC2983, LTC2984, LTC2986 and LTC2986-1 Temp-to-Bits converters allow the use of custom sensors in addition to built-in standard types. This post focuses on custom thermistors and two ways of configuring the converter.

Thermistors are resistive temperature sensors based on semiconductor materials. They are available in two types - those that have a negative temperature coefficient [NTC], and those that have a positive temperature coefficient [PTC]. While many metals, including Nickel and Platinum used in RTDs have a positive temperature coefficient, they differ from thermistors in both the linearity of their response as well as the range of resistances they exhibit for a given temperature range (due to the logarithmic response of the thermistor).

Users of thermistors must determine temperature from a measured resistance with help from the thermistor vendor. The Temp-to-Bits family includes built-in coefficients for many common thermistors including the popular 44004/44033 2.252kΩ, 44006/44031 10kΩ, and 44008/44032 30kΩ devices. Using one of these standard types is the simplest way to make thermistor temperature measurements. However, as this is not always possible or desired, support for custom sensors is also provided.

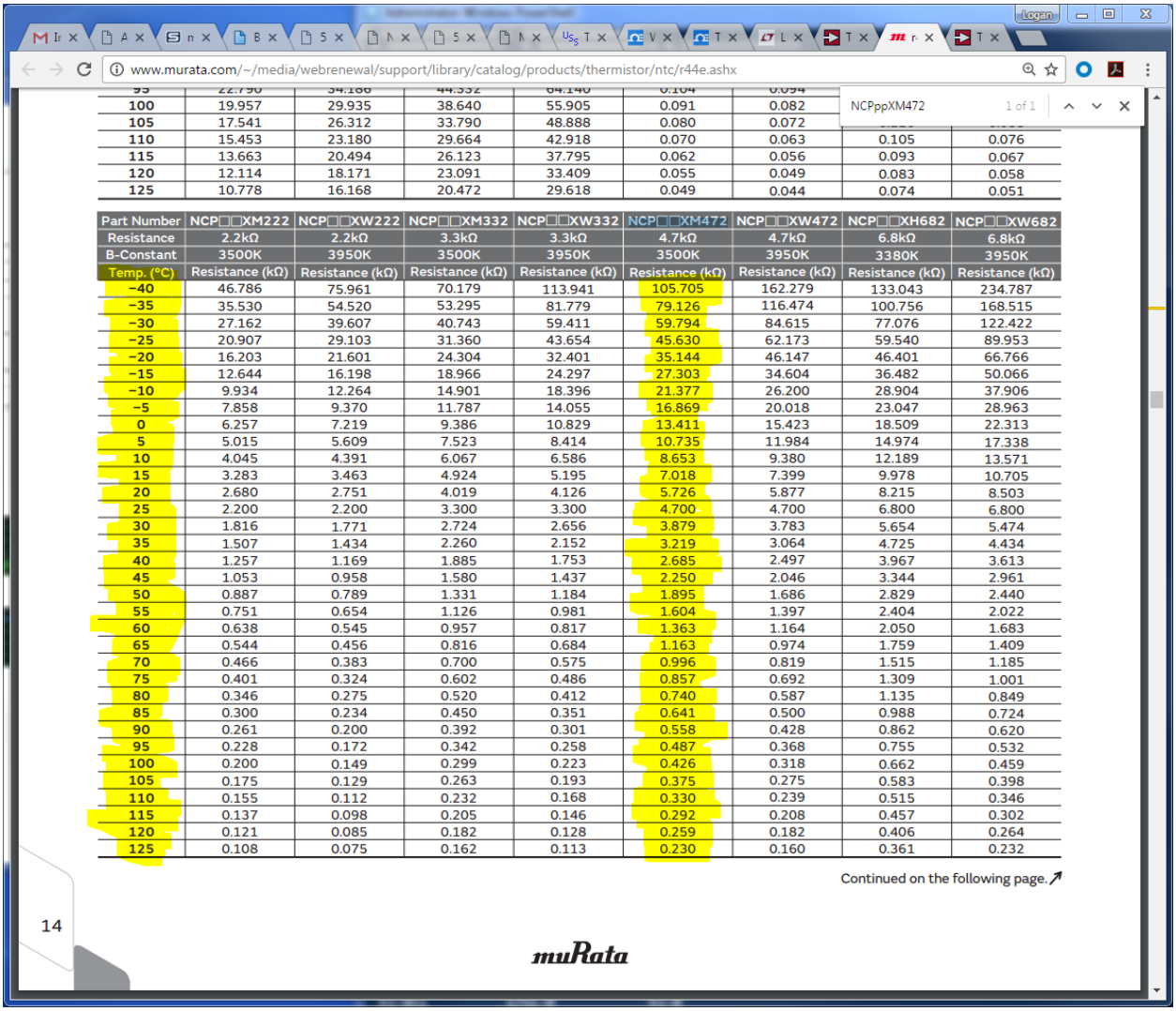
Vendors will supply either a Resistance-Temperature table, coefficients for the Steinhart-Hart equation, or a Beta value. Beta used here is a single number that approximates the Steinhart-Hart equation over a limited range. Since most vendors who supply a Beta value also provide Resistance-Temperature tables that cover the sensor's full range, we will focus on the Steinhart-Hart and table entry methods for our custom thermistors.

Entering a custom Thermistor into Memory:

We will use the TestBench GUI to demonstrate this functionality. Note that by starting with the GUI we are not shooting ourselves in the foot if we need C code later – the GUI can help by generating C code including tables and coefficients, while showing effective ways to load the custom sensor parameters into RAM. Testing a new sensor configuration with the GUI and demo circuits is a great way to ensure the trickier parts of using a custom sensor are working.

The most straightforward way to use a custom resistive sensor is by entering an R-T table into memory. Let's choose a random NTC thermistor and get started. The Murata NCP15XM472J03RC 4.7kΩ nominal @ 25°C happens to be tied for least expensive thermistor on Digi-Key, and 4.7kΩ nominal is not found in our standard curves.

A click on the datasheet shows us a Resistance-Temperature table nearly instantly:



100	19.957	29.935	38.640	55.905	0.091	0.082		
105	17.541	26.312	33.790	48.888	0.080	0.072		
110	15.453	23.180	29.664	42.918	0.070	0.063	0.105	0.076
115	13.663	20.494	26.123	37.795	0.062	0.056	0.093	0.067
120	12.114	18.171	23.091	33.409	0.055	0.049	0.083	0.058
125	10.778	16.168	20.472	29.618	0.049	0.044	0.074	0.051

Part Number	NCP□□XM222	NCP□□XW222	NCP□□XM332	NCP□□XW332	NCP□□XM472	NCP□□XW472	NCP□□XH682	NCP□□XW682
Resistance	2.2kΩ	2.2kΩ	3.3kΩ	3.3kΩ	4.7kΩ	4.7kΩ	6.8kΩ	6.8kΩ
B-Constant	3500K	3950K	3500K	3950K	3500K	3950K	3380K	3950K
Temp. (°C)	Resistance (kΩ)	Resistance (kΩ)	Resistance (kΩ)	Resistance (kΩ)	Resistance (kΩ)	Resistance (kΩ)	Resistance (kΩ)	Resistance (kΩ)
−40	46.786	75.961	70.179	113.941	105.705	162.279	133.043	234.787
−35	35.530	54.520	53.295	81.779	79.126	116.474	100.756	168.515
−30	27.162	39.607	40.743	59.411	59.794	84.615	77.076	122.422
−25	20.907	29.103	31.360	43.654	45.630	62.173	59.540	89.953
−20	16.203	21.601	24.304	32.401	35.144	46.147	46.401	66.766
−15	12.644	16.198	18.966	24.297	27.303	34.604	36.482	50.066
−10	9.934	12.264	14.901	18.396	21.377	26.200	28.904	37.906
−5	7.858	9.370	11.787	14.055	16.869	20.018	23.047	28.963
0	6.257	7.219	9.386	10.829	13.411	15.423	18.509	22.313
5	5.015	5.609	7.523	8.414	10.735	11.984	14.974	17.338
10	4.045	4.391	6.067	6.586	8.653	9.380	12.189	13.571
15	3.283	3.463	4.924	5.195	7.018	7.399	9.978	10.705
20	2.680	2.751	4.019	4.126	5.726	5.877	8.215	8.503
25	2.200	2.200	3.300	3.300	4.700	4.700	6.800	6.800
30	1.816	1.771	2.724	2.656	3.879	3.783	5.654	5.474
35	1.507	1.434	2.260	2.152	3.219	3.064	4.725	4.434
40	1.257	1.169	1.885	1.753	2.685	2.497	3.967	3.613
45	1.053	0.958	1.580	1.437	2.250	2.046	3.344	2.961
50	0.887	0.789	1.331	1.184	1.895	1.686	2.829	2.440
55	0.751	0.654	1.126	0.981	1.604	1.397	2.404	2.022
60	0.638	0.545	0.957	0.817	1.363	1.164	2.050	1.683
65	0.544	0.456	0.816	0.684	1.163	0.974	1.759	1.409
70	0.466	0.383	0.700	0.575	0.996	0.819	1.515	1.185
75	0.401	0.324	0.602	0.486	0.857	0.692	1.309	1.001
80	0.346	0.275	0.520	0.412	0.740	0.587	1.135	0.849
85	0.300	0.234	0.450	0.351	0.641	0.500	0.988	0.724
90	0.261	0.200	0.392	0.301	0.558	0.428	0.862	0.620
95	0.228	0.172	0.342	0.258	0.487	0.368	0.755	0.532
100	0.200	0.149	0.299	0.223	0.426	0.318	0.662	0.459
105	0.175	0.129	0.263	0.193	0.375	0.275	0.583	0.398
110	0.155	0.112	0.232	0.168	0.330	0.239	0.515	0.346
115	0.137	0.098	0.205	0.146	0.292	0.208	0.457	0.302
120	0.121	0.085	0.182	0.128	0.259	0.182	0.406	0.264
125	0.108	0.075	0.162	0.113	0.230	0.160	0.361	0.232

Figure 1: Murata Datasheet showing R-T Table

I've highlighted the sensor we're looking at – we have a nice Temperature column and a matching Resistance column. That's all the data we need to configure our custom thermistor so let's go ahead and get set up for a bench test.

Configuring the DC2210, TestBench GUI, custom sensor, and performing a bench verification test:

First I'll get some hardware set up so that when I configure the custom sensor I can test right away.

I'm using the DC2026C Linduino One to interface with the DC2209A LTC2983 Demo Circuit. Finally, to connect test resistors I'm using the DC2210A experimenter board. On the DC2210A here in the lab I've installed a jumper between COM and GND and some headers for convenience.

For this application, I will choose an Rsense of the same order of magnitude as largest resistance I expect to encounter – this will give me the most excitation current range and accuracy. 10kΩ is a great starting point for most thermistor designs. In this case I chose a precision 5kΩ resistor for Rsense, and have installed it between CH1 and CH2. I've installed a 2.2kΩ test resistor in place of the thermistor,

which if the custom sensor is configured correctly, should read between 45 and 50C according to the datasheet. Since the test resistor is between CH7 and CH8, I'll need a jumper between the bottom of the sense resistor on CH2 and the top of my sensor on CH7. The test setup is shown in Figure 2.

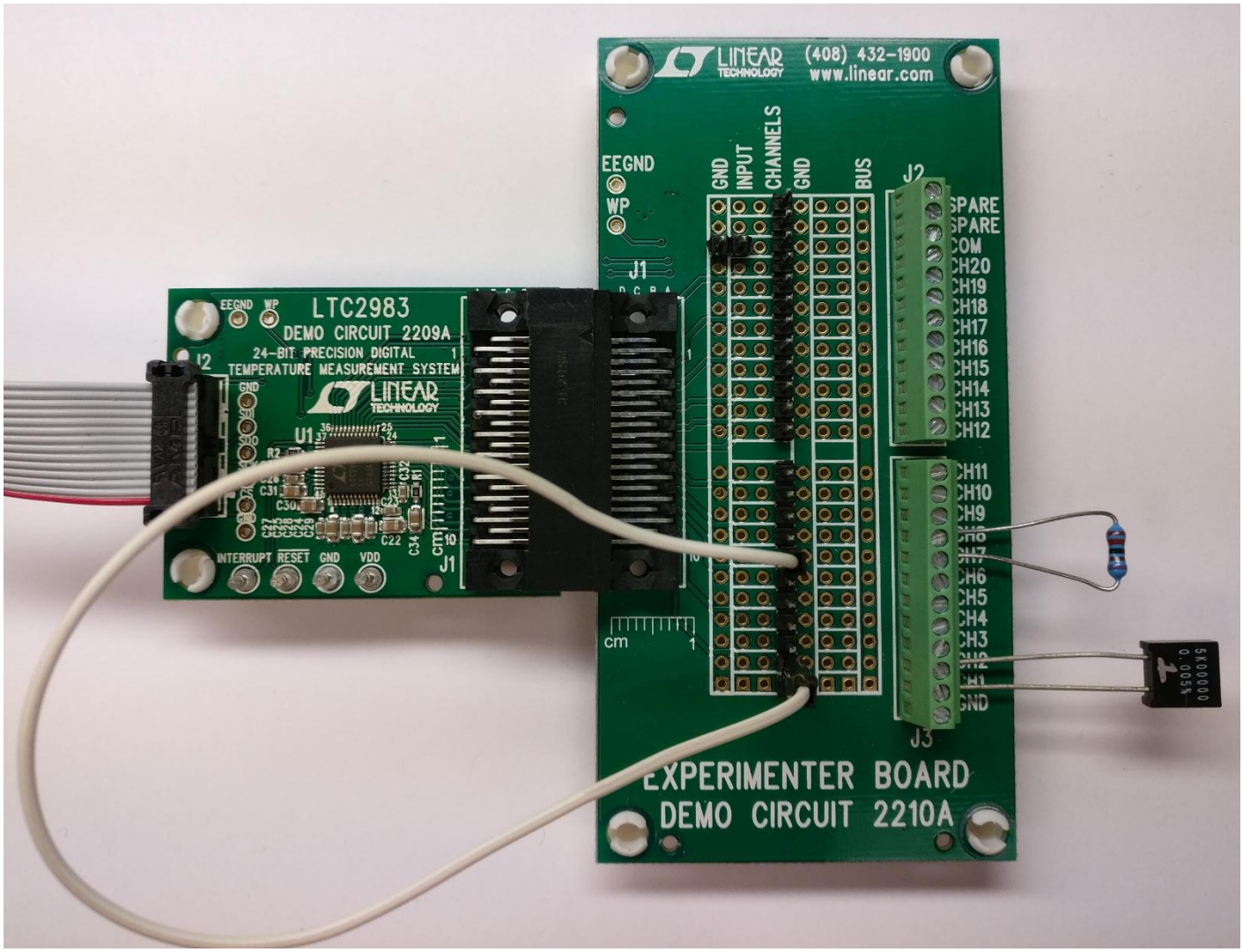


Figure 2: DC2210A setup for Thermistor Test (Linduino One not shown)

Now that we have our hardware set-up, let's take a look at the GUI configuration.
First we'll need to configure a 5kΩ sense resistor on channel 2:

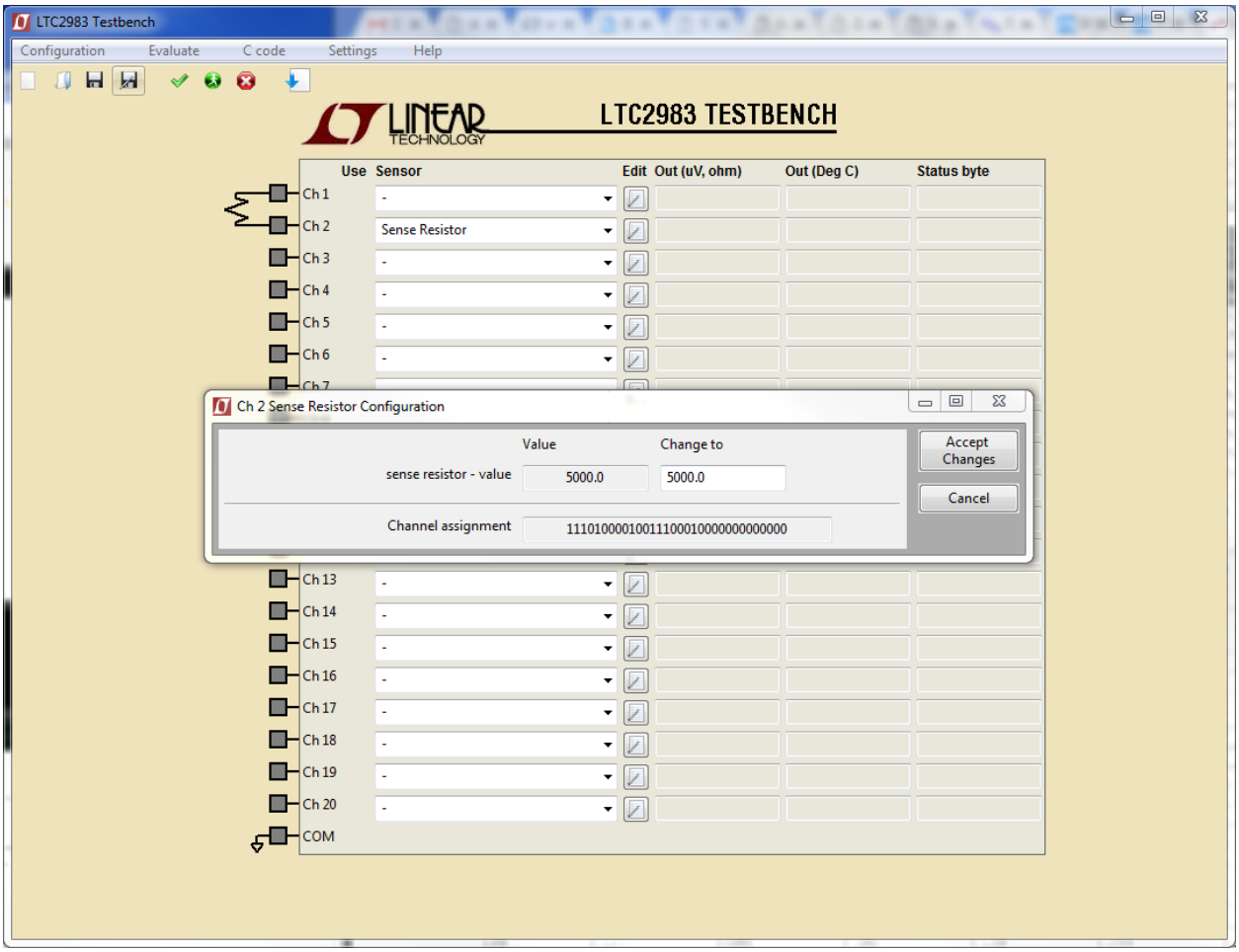


Figure 3: Sense Resistor Configuration

Next, we'll configure our Thermistor Custom Table on channel 8. The datasheet lists a maximum excitation current of 0.31mA. The LTC2983 has a maximum differential voltage of 1.25V, so with our 5kΩ maximum resistance we can use up to 250µA of excitation current – fitting nicely under the datasheet maximum. Since the actual sensor chosen is a small device, we'll lower this to 100µA to reduce self-heating. Note that my favorite excitation current selection, Autorange, is not allowed for custom sensors, so I am using one current that will continue to work well at higher temperatures where the NTC resistance is lower.

Now we're nearly ready to enter our custom data. Custom sensor data can be placed anywhere within user RAM so we'll need to be cautious of overlap. Since this is my first custom sensor the RAM is a blank slate and I do not need to offset the data in memory, so we'll leave the custom address at 0. I also haven't counted our data points yet, so we'll leave the table length to 0 for now – we'll have to set this before closing the dialog box.

Let's look at the custom value entry window:

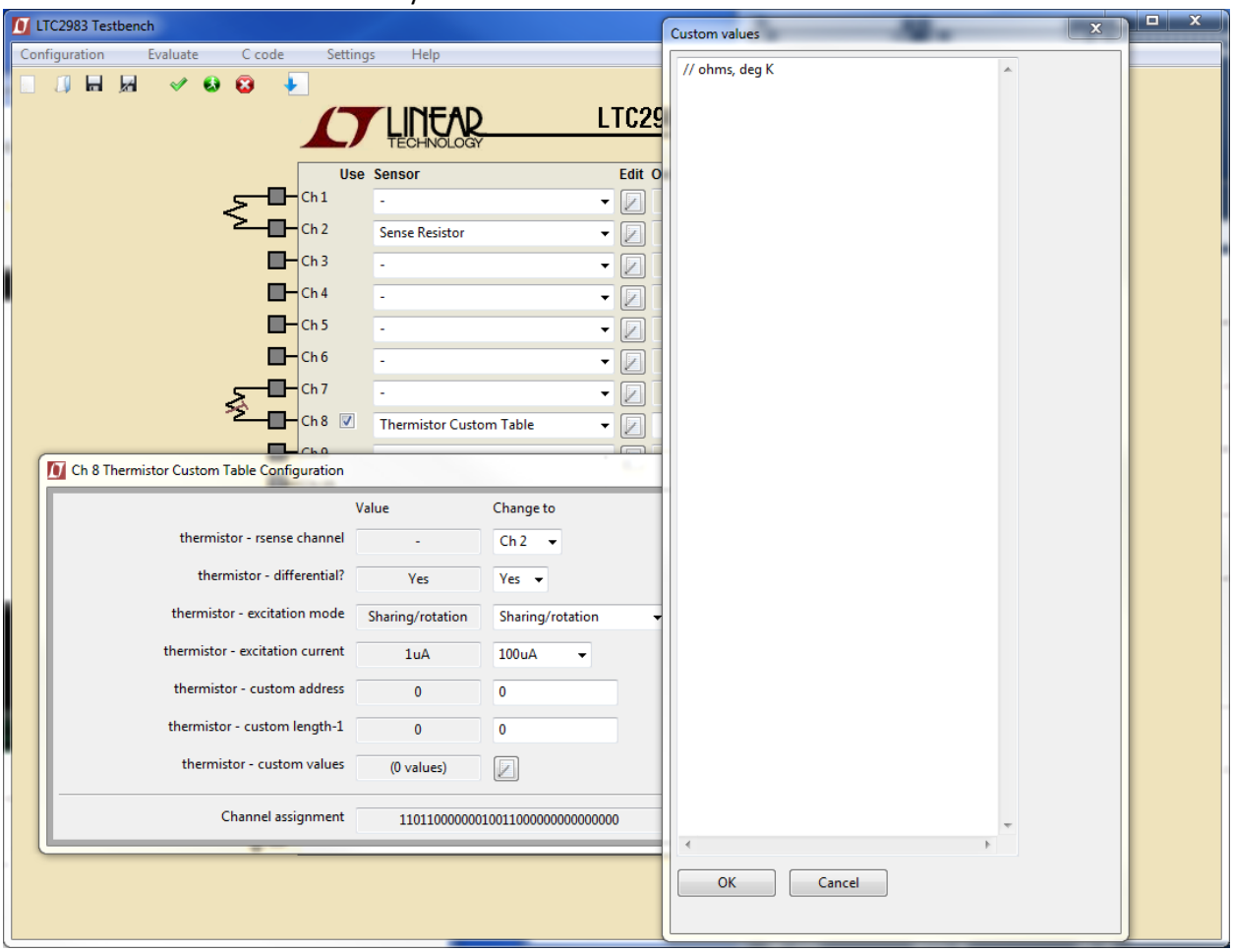


Figure 4: The custom value table must be in Ohms, deg K

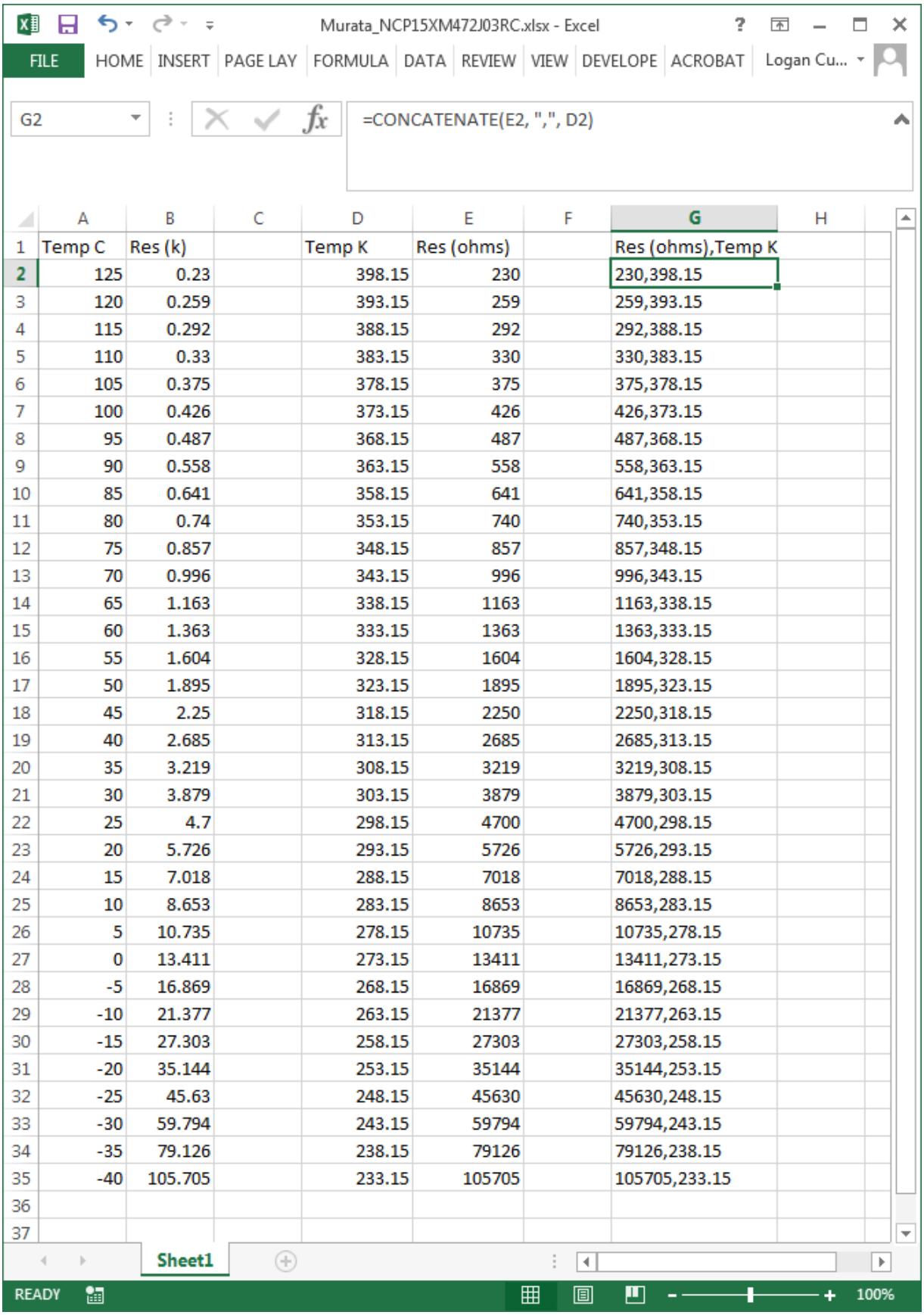
Right off the bat I notice it's going to be a little work to type in each value, and that the datasheet table isn't quite in the right format – the datasheet has temperature in Celsius and kilo-ohms instead of ohms, which the LTC2983 needs. Instead of trying to modify these one by one as I type them in, I'll get some help from a spreadsheet.

Thankfully, Murata's PDF is text-copyable, so a couple of copy-pastes later we have some data. Other vendors are not so nice, but with the spreadsheet method you only do data-entry once.

Next we'll create new columns for Temp in Kelvin, and Resistance in Ohms, these are simple formula columns (e.g. "=A2+273.15" "=B2*1000"). Noting that the custom data must be in Ohms, Temp (K) and must be monotonically increasing (ohms), we'll sort the sheet to be increasing by Resistance. See page 65 of the LTC2983 datasheet for more.

Finally, we'll create a column to give us a comma separated value ready for import into the TestBench GUI – again, this is a built-in Excel function (e.g. "=CONCATENATE(E2, ",", D2)"). Our spreadsheet shown

in Figure 5 is now complete:



	A	B	C	D	E	F	G	H
1	Temp C	Res (k)		Temp K	Res (ohms)		Res (ohms),Temp K	
2	125	0.23		398.15	230		230,398.15	
3	120	0.259		393.15	259		259,393.15	
4	115	0.292		388.15	292		292,388.15	
5	110	0.33		383.15	330		330,383.15	
6	105	0.375		378.15	375		375,378.15	
7	100	0.426		373.15	426		426,373.15	
8	95	0.487		368.15	487		487,368.15	
9	90	0.558		363.15	558		558,363.15	
10	85	0.641		358.15	641		641,358.15	
11	80	0.74		353.15	740		740,353.15	
12	75	0.857		348.15	857		857,348.15	
13	70	0.996		343.15	996		996,343.15	
14	65	1.163		338.15	1163		1163,338.15	
15	60	1.363		333.15	1363		1363,333.15	
16	55	1.604		328.15	1604		1604,328.15	
17	50	1.895		323.15	1895		1895,323.15	
18	45	2.25		318.15	2250		2250,318.15	
19	40	2.685		313.15	2685		2685,313.15	
20	35	3.219		308.15	3219		3219,308.15	
21	30	3.879		303.15	3879		3879,303.15	
22	25	4.7		298.15	4700		4700,298.15	
23	20	5.726		293.15	5726		5726,293.15	
24	15	7.018		288.15	7018		7018,288.15	
25	10	8.653		283.15	8653		8653,283.15	
26	5	10.735		278.15	10735		10735,278.15	
27	0	13.411		273.15	13411		13411,273.15	
28	-5	16.869		268.15	16869		16869,268.15	
29	-10	21.377		263.15	21377		21377,263.15	
30	-15	27.303		258.15	27303		27303,258.15	
31	-20	35.144		253.15	35144		35144,253.15	
32	-25	45.63		248.15	45630		45630,248.15	
33	-30	59.794		243.15	59794		59794,243.15	
34	-35	79.126		238.15	79126		79126,238.15	
35	-40	105.705		233.15	105705		105705,233.15	
36								
37								

Figure 5: Using a spreadsheet program gets us CSV formatted data ready for import into the TestBench GUI

Now we can simply copy-paste the resulting table into the custom values window in the TestBench GUI:

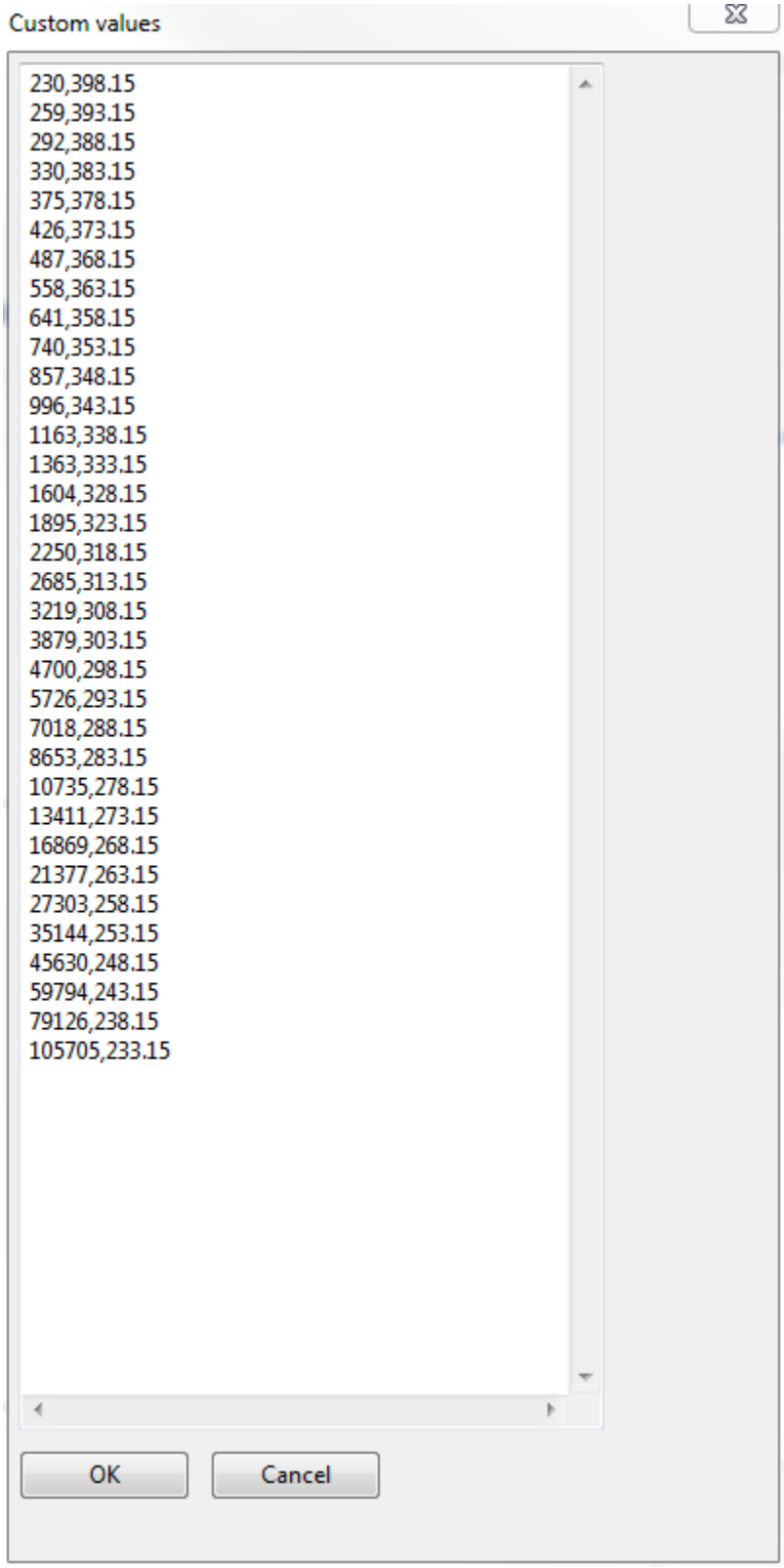


Figure 6: The correctly formatted CSV column directly copies into the TestBench GUI

Hit OK. We know from our spreadsheet that we have 34 rows of data, or 34 table entries. Since the GUI needs length-1, we'll enter 33 for our table length, and hit accept changes:

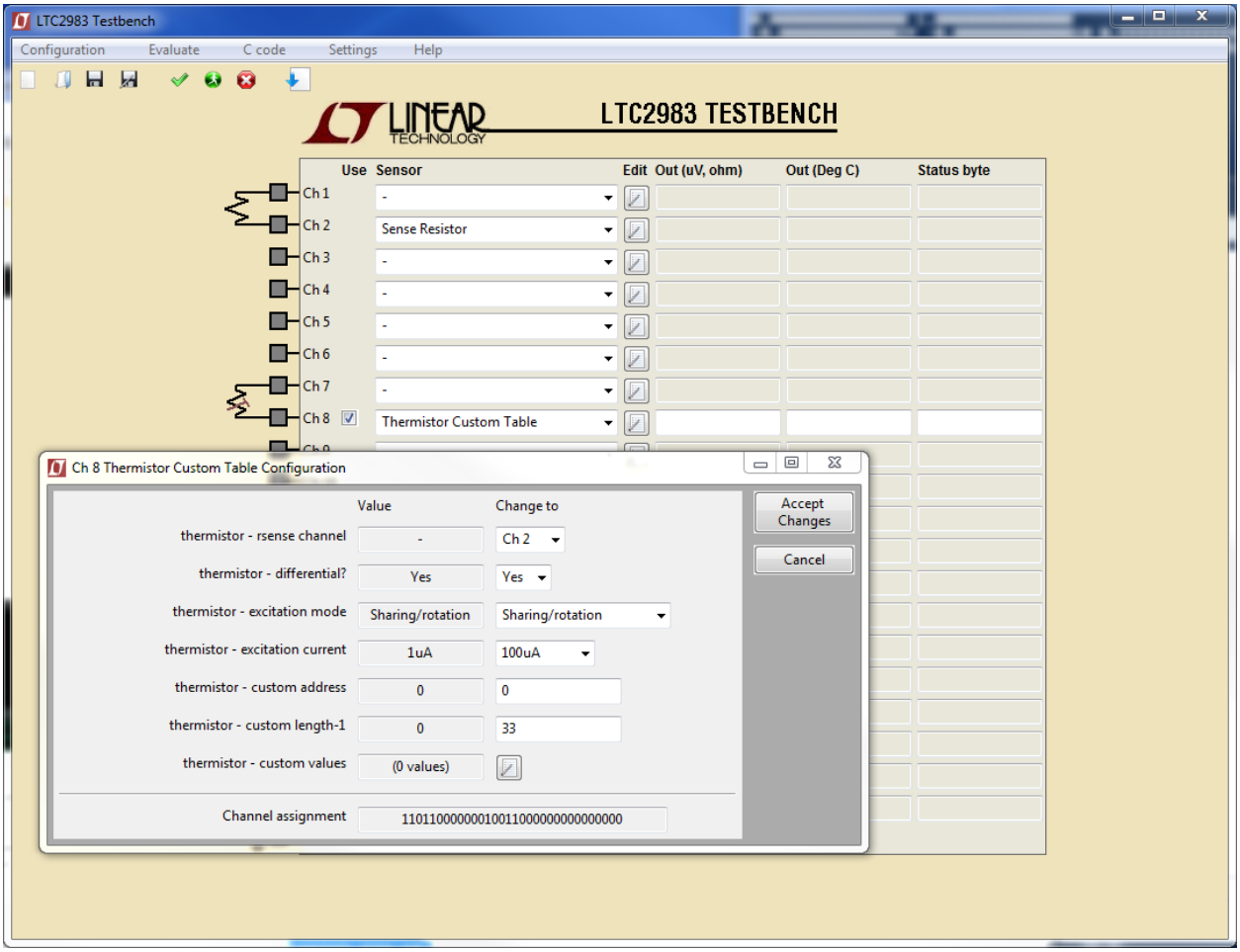


Figure 7: Final Custom Table Configuration window showing length of table data in User RAM

At this point the GUI should display a wiring diagram which matches our test setup:

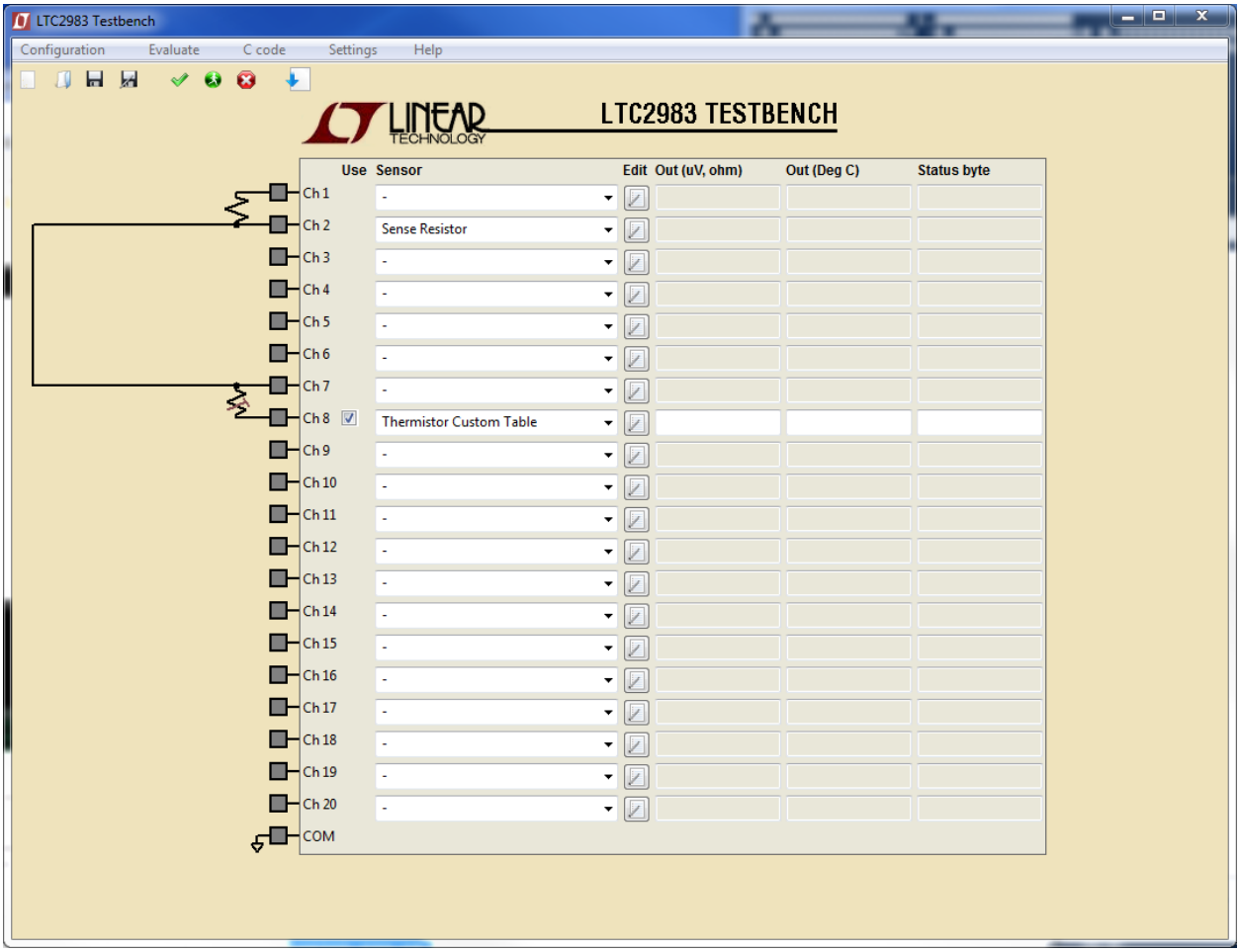


Figure 8: Verifying the wiring diagram in the GUI is a great way to catch configuration issues early

Let's test! First let's validate our config using the green checkmark up top:

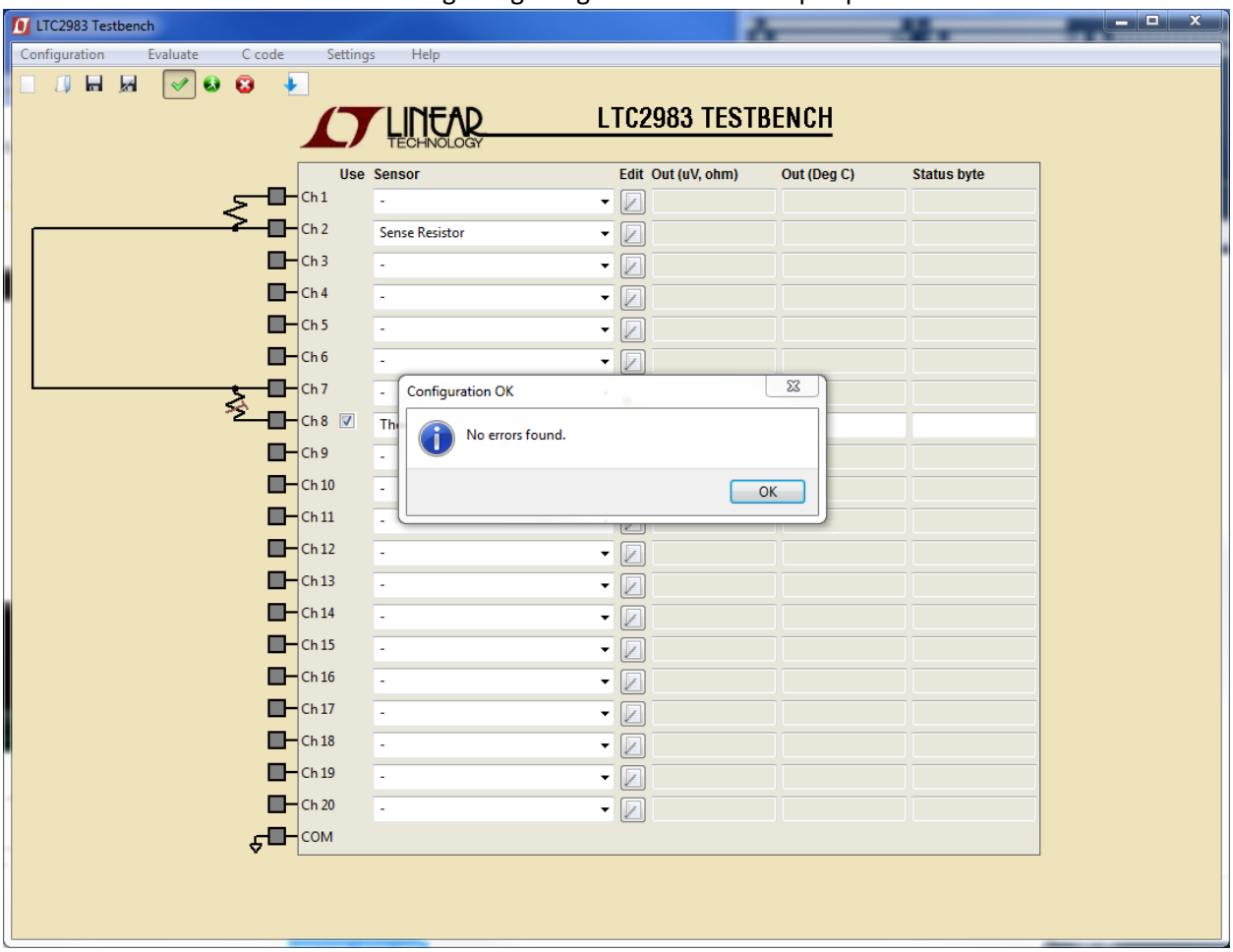


Figure 9: The TestBench GUI can validate a config including checks for overlapping leads and missing Rsense connections

If all went well, when we run to evaluate our config (green button) we should see a temperature reported in Degrees C right around 46 C (guesstimate from the datasheet table):

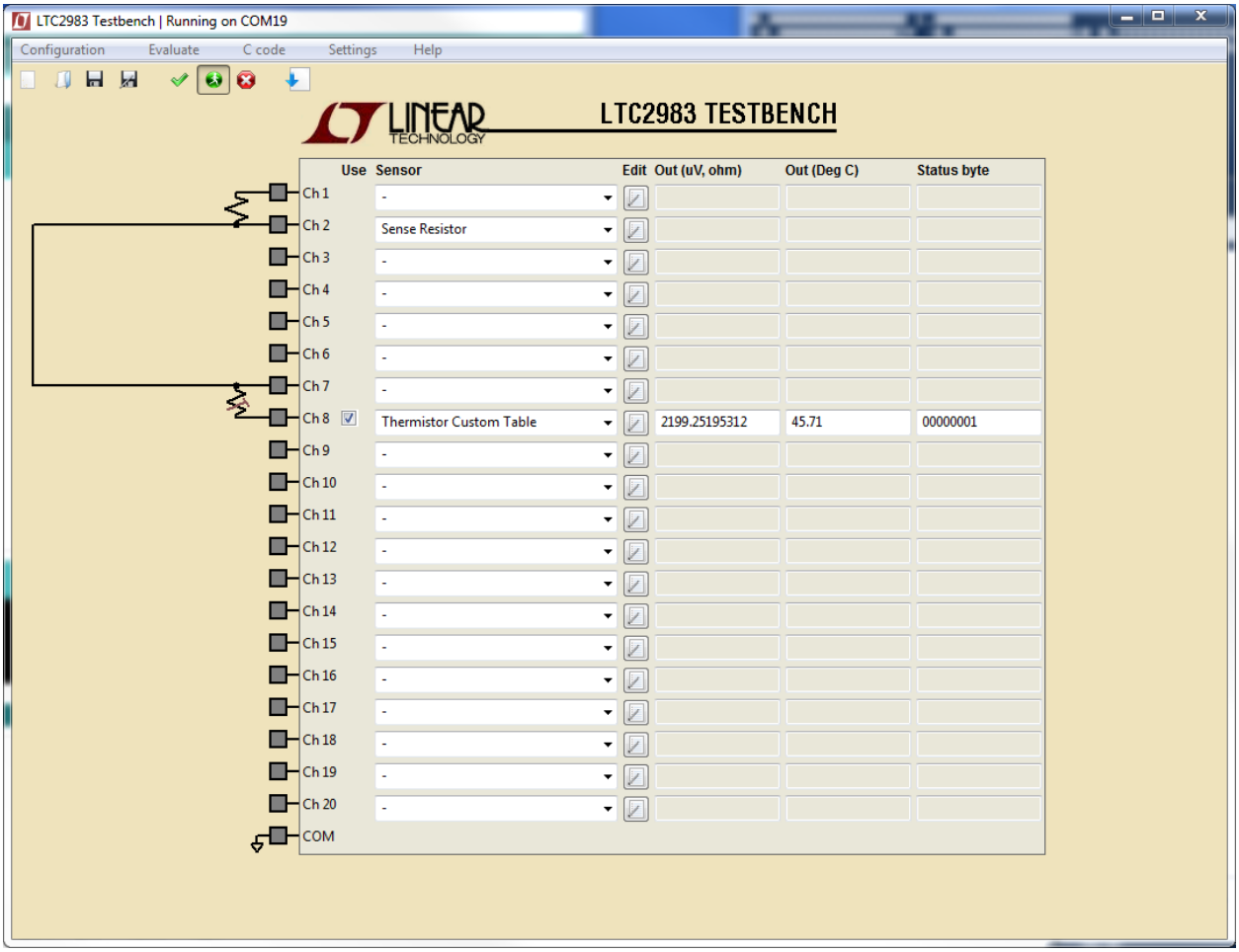


Figure 10: The GUI displays the calculated temperature as well as measured resistance, enabling quick checks of Rsense value

Success! The resistance shown in Figure 10 is correct, meaning our sense resistor configuration was spot on, and the temperature is correctly interpolated between our 45C and 50C data points from the table.

Let's check one more value to make sure our table works, a 330-ohm resistor as this should display exactly 110C:

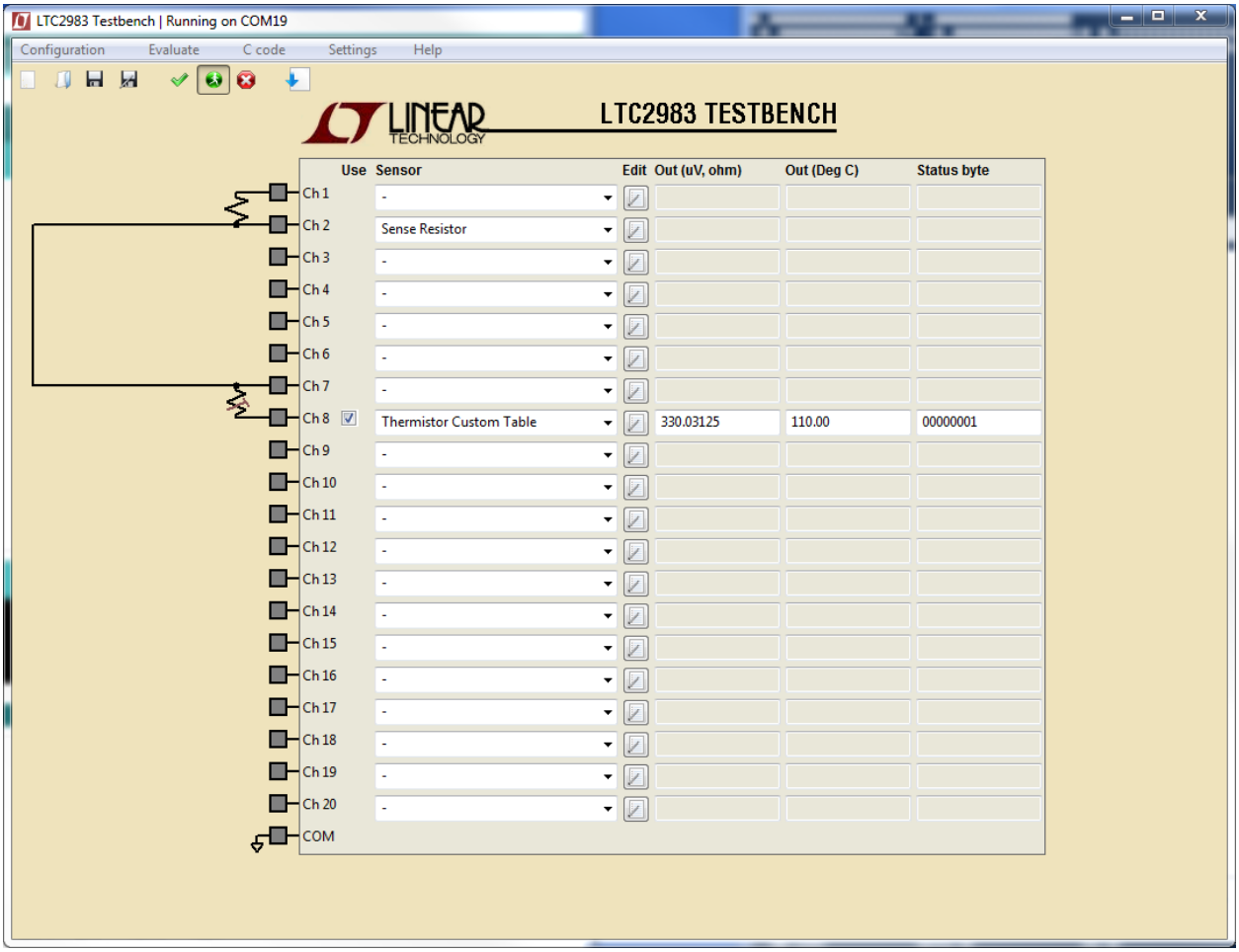


Figure 11: A second resistance value shows our table is being interpolated correctly

Not bad.

Thermistor Custom Steinhart-Hart

Now what about the other custom method – Steinhart-Hart coefficients? Some vendors provide these directly usable for our method (e.g. Omega) while others provide them in a slightly different, but still valid way (e.g. Vishay) – we'll take a look at one that needs some minor manipulation to give an idea of what's required.

The Vishay NTCLE100E3 family of leaded NTC thermistors are popular low cost sensors. Let's take a look at the 10kΩ nominal at 25°C which has a Beta(25/85) of 3997K. Vishay provides a table of coefficients,

shown in Figure 11:

R_T VALUE AND TOLERANCE

These thermistors have a narrow tolerance on the B-value, the result of which provides a very small tolerance on the nominal resistance value over a wide temperature range. For this reason the usual graphs of $R = f(T)$ are replaced by Resistance Values at Intermediate Temperatures Tables, together with a formula to calculate the characteristics with a high precision.

FORMULAE TO DETERMINE NOMINAL RESISTANCE VALUES

The resistance values at intermediate temperatures, or the operating temperature values, can be calculated using the following interpolation laws (extended *"Steinhart and Hart"*):

$$R_{(T)} = R_{ref} \times e^{(A + B/T + C/T^2 + D/T^3)} \quad (1)$$

$$T_{(R)} = \left(A_1 + B_1 \ln\frac{R}{R_{ref}} + C_1 \ln^2\frac{R}{R_{ref}} + D_1 \ln^3\frac{R}{R_{ref}}\right)^{-1} \quad (2)$$

where:

A, B, C, D, A_1, B_1, C_1 and D_1 are constant values depending on the material concerned; see table below.

$R_{ref.}$ is the resistance value at a reference temperature (in this event 25 °C, $R_{ref.} = R_{25}$).

T is the temperature in K.

Formulae numbered and are interchangeable with an error of max. 0.005 °C in the range 25 °C to 125 °C and max. 0.015 °C in the range - 40 °C to + 25 °C.

DETERMINATION OF THE RESISTANCE/TEMPERATURE DEVIATION FROM NOMINAL VALUE

The total resistance deviation is obtained by combining the "R_{25}-tolerance" and the "resistance deviation due to B-tolerance".

When:

X = R_{25}-tolerance
Y = resistance deviation due to B-tolerance
Z = complete resistance deviation,

then: $Z = \left[\left(1 + \frac{X}{100}\right) \times \left(1 + \frac{Y}{100}\right) - 1\right] \times 100\ \%$ or $Z \approx X + Y$

When:

TCR = temperature coefficient
ΔT = temperature deviation,

then: $\Delta T = \frac{Z}{TCR}$

The temperature tolerances are plotted in the graphs on the previous page.

Example: at 0 °C, assume X = 5 %, Y = 0.89 % and TCR = 5.08 %/K (see table), then:

$$Z = \left\{\left[1 + \frac{5}{100}\right] \times \left[1 + \frac{0.89}{100}\right] - 1\right\} \times 100\,\%$$

$$= \{1.05 \times 1.0089 - 1\} \times 100\ \% = 5.9345\ \%\ (\approx 5.93\ \%)$$

$$\Delta T = \frac{Z}{TCR} = \frac{5.93}{5.08} = 1.167\ °C\ (\approx 1.17\ °C)$$

A NTC with a R_{25}-value of 10 kΩ has a value of 32.56 kΩ between - 1.17 °C and + 1.17 °C.

PARAMETER FOR DETERMINING NOMINAL RESISTANCE VALUES

NUMBER	$B_{25/85}$ (K)	NAME	TOL. B (%)	A	B (K)	C (K^2)	D (K^3)	A_1	B_1 (K^{-1})	C_1 (K^{-2})	D_1 (K^{-3})
1	2880	Mat O. with Bn = 2880K	3	- 9.094	2251.74	229098	- 2.744820E+07	3.354016E-03	3.495020E-04	2.095959E-06	4.260615E-07
2	2990	Mat P. with Bn = 3990K	3	- 10.2296	2887.62	132336	- 2.502510E+07	3.354016E-03	3.415560E-04	4.955455E-06	4.364236E-07
3	3041	Mat Q. with Bn = 3041K	3	- 11.1334	3658.73	- 102895	5.166520E+05	3.354016E-03	3.349290E-04	3.683843E-06	7.050455E-07
4	3136	Mat R. with Bn = 3136K	3	- 12.4493	4702.74	- 402687	3.196830E+07	3.354016E-03	3.243880E-04	2.658012E-06	- 2.701560E-07
5	3390	Mat S. with Bn = 3390K	3	- 12.6814	4391.97	- 232807	1.509643E+07	3.354016E-03	2.993410E-04	2.135133E-06	- 5.672000E-09
6	3528 (1)	Mat I. with Bn = 3528K	0.5	- 12.0596	3687.667	- 7617.13	- 5.914730E+06	3.354016E-03	2.909670E-04	1.632136E-06	7.192200E-08
	3528 (2)			- 21.0704	11903.95	- 2504699	2.470338E+08	3.354016E-03	2.933908E-04	3.494314E-06	- 7.712690E-07
7	3560	Mat H. with Bn = 3560K	1.5	- 13.0723	4190.574	- 47158.4	- 1.199256E+07	3.354016E-03	2.884193E-04	4.118032E-06	1.786790E-07
8	3740	Mat B. with Bn = 3740K	2	- 13.8973	4557.725	- 98275	- 7.522357E+06	3.354016E-03	2.744032E-04	3.666944E-06	1.375492E-07
9	3977	Mat A. with Bn =3977K	0.75	- 14.6337	4791.842	- 115334	- 3.730535E+06	3.354016E-03	2.569850E-04	2.620131E-06	6.383091E-08

Figure 12: Vishay uses a slightly different form of the Steinhart-Hart equation in their thermistor datasheets

I've highlighted the coefficients of interest, as well as the Steinhart-Hart equation Vishay uses. Note that Vishay's equation is a little different than the one the LTC2983 expects:

Table address pointers.

Table 50. Thermistor Channel Assignment Word

	(1) THERMISTOR TYPE					(2) SENSE RESISTOR CHANNEL POINTER					(3) SENSOR CONFIGURATION			(4) EXCITATION CURRENT						(5) CUSTOM THERMISTOR DATA POINTER									
	TABLE 51					TABLE 27					TABLE 52			TABLE 53						TABLES 76, 77, 78, 80, 81									
Measurement Class	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Thermistor	Type = 19 to 27					R_{SENSE} Channel Pointer [4:0]					SGL = 1 DIFF = 0	Excitation Mode		Excitation Current [3:0]				Not Used 0 0 0			Custom Address [5:0]						Custom Length – 1 [5:0]					

Table 51. Thermistor Type: $1/T = A + B \bullet \ln(R) + C \bullet \ln(R)^2 + D \bullet \ln(R)^3 + E \bullet \ln(R)^4 + F \bullet \ln(R)^5$

B31	B30	B29	B28	B27	THERMISTOR TYPE	A	B	C	D	E	F
1	0	0	1	1	Thermistor 44004/44033 2.252kΩ at 25°C	1.46800E-03	2.38300E-04	0	1.00700E-07	0	0
1	0	1	0	0	Thermistor 44005/44030 3kΩ at 25°C	1.40300E-03	2.37300E-04	0	9.82700E-08	0	0
1	0	1	0	1	Thermistor 44007/44034 5kΩ at 25°C	1.28500E-03	2.36200E-04	0	9.28500E-08	0	0

Figure 13: The full Steinhart-Hart equation as built into the LTC2983, LTC2984, LTC2986, and LTC2986-1.

We can set E and F to zero, so there is no issue not having them in the Vishay datasheet. However, the LTC2983 equation uses coefficients based on R values in Ohms, while the Vishay coefficients take R relative to Rref, or R25 – the nominal value at 25°C. There's a little arithmetic involved in the conversion:

First we'll want to get rid of the fraction inside the logarithm. For that we can use the identity

$$\ln\left(\frac{x}{y}\right) = \ln(x) - \ln(y)$$

So then the Vishay equation becomes:

$$\frac{1}{T} = A_1 + B_1(\ln(R) - \ln(R_{25})) + C_1(\ln(R) - \ln(R_{25}))^2 + D_1(\ln(R) - \ln(R_{25}))^3$$

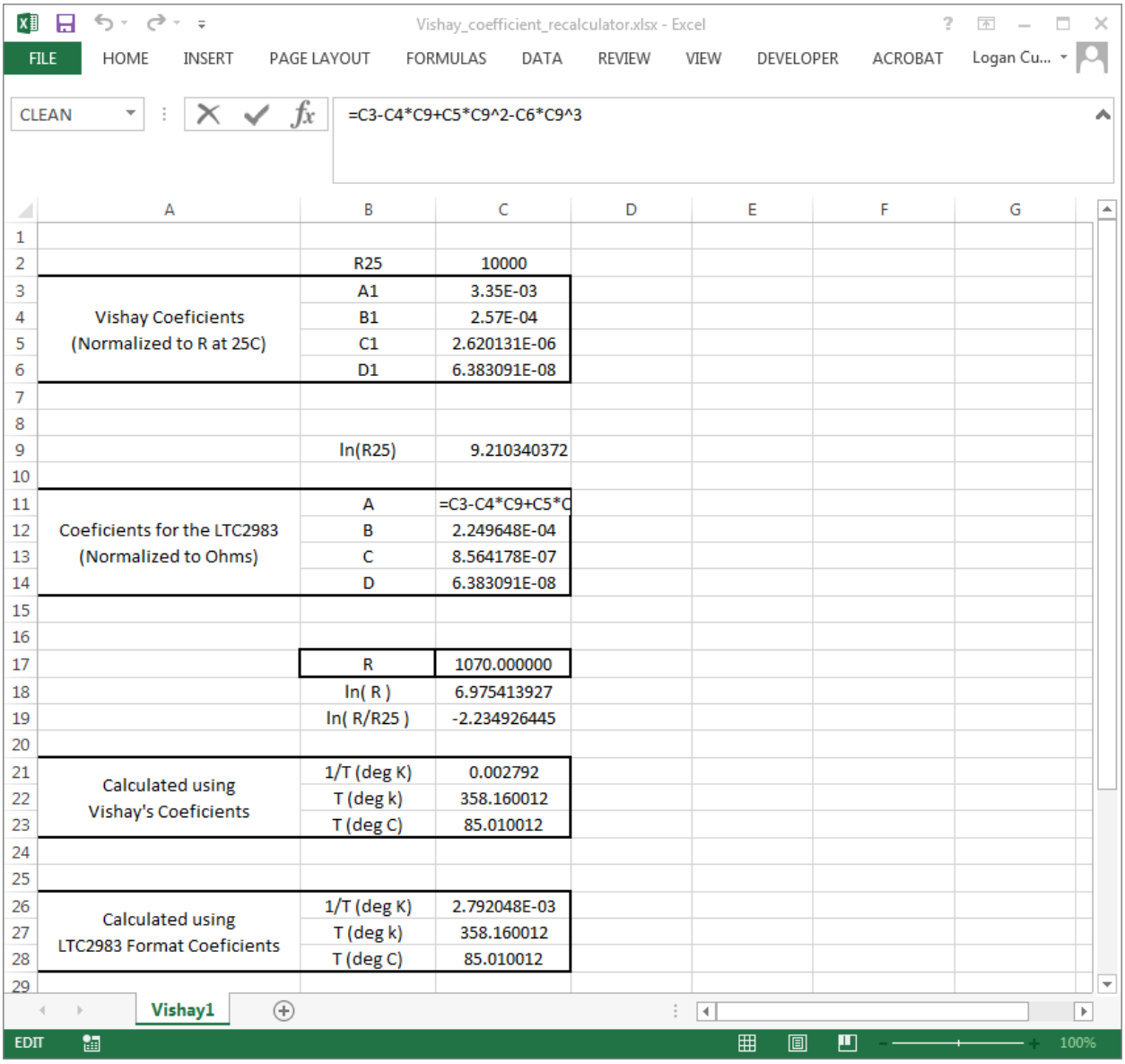
Expanding the polynomial terms and multiplying the coefficients through gives us:

$$\begin{aligned}\frac{1}{T} = {} & A_1 \\ & + B_1 \ln(R) - B_1 \ln(R_{25}) \\ & + C_1(\ln(R))^2 - 2C_1 \ln(R) \ln(R_{25}) + C_1(\ln(R_{25}))^2 \\ & + D_1(\ln(R))^3 - 3D_1(\ln(R))^2(\ln(R_{25})) + 3D_1(\ln(R))(\ln(R_{25}))^2 - D_1(\ln(R_{25}))^3\end{aligned}$$

Vishay gives us A_1, B_1, C_1, D_1 and R_{25} which are constants and allow us to rearrange the above so that we have an equation in terms of $\ln(R)$, $(\ln(R))^2$, *etc.* to match our datasheet format. Our new coefficients then become:

$$\begin{aligned}A &= A_1 - B_1 \ln(R_{25}) + C_1(\ln(R_{25}))^2 - D_1(\ln(R_{25}))^3 \\ B &= B_1 - 2C_1 \ln(R_{25}) + 3D_1(\ln(R_{25}))^2 \\ C &= C_1 - 3D_1(\ln(R_{25})) \\ D &= D_1\end{aligned}$$

Now we can calculate our new coefficients and enter them into the GUI. Again, a spreadsheet helps here to maintain the equations and perform some rudimentary testing:



	A	B	C
2		R25	10000
3	Vishay Coeficients (Normalized to R at 25C)	A1	3.35E-03
4		B1	2.57E-04
5		C1	2.620131E-06
6		D1	6.383091E-08
9		ln(R25)	9.210340372
11	Coeficients for the LTC2983 (Normalized to Ohms)	A	=C3-C4*C9+C5*C
12		B	2.249648E-04
13		C	8.564178E-07
14		D	6.383091E-08
17		R	1070.000000
18		ln(R)	6.975413927
19		ln(R/R25)	-2.234926445
21	Calculated using Vishay's Coeficients	1/T (deg K)	0.002792
22		T (deg k)	358.160012
23		T (deg C)	85.010012
26	Calculated using LTC2983 Format Coeficients	1/T (deg K)	2.792048E-03
27		T (deg k)	358.160012
28		T (deg C)	85.010012

Figure 14: Using a spreadsheet to convert coefficients between Vishay's form of the Steinhart-Hart coefficients and ours

We'll set up a new custom thermistor on channel 12 to test the coefficients:

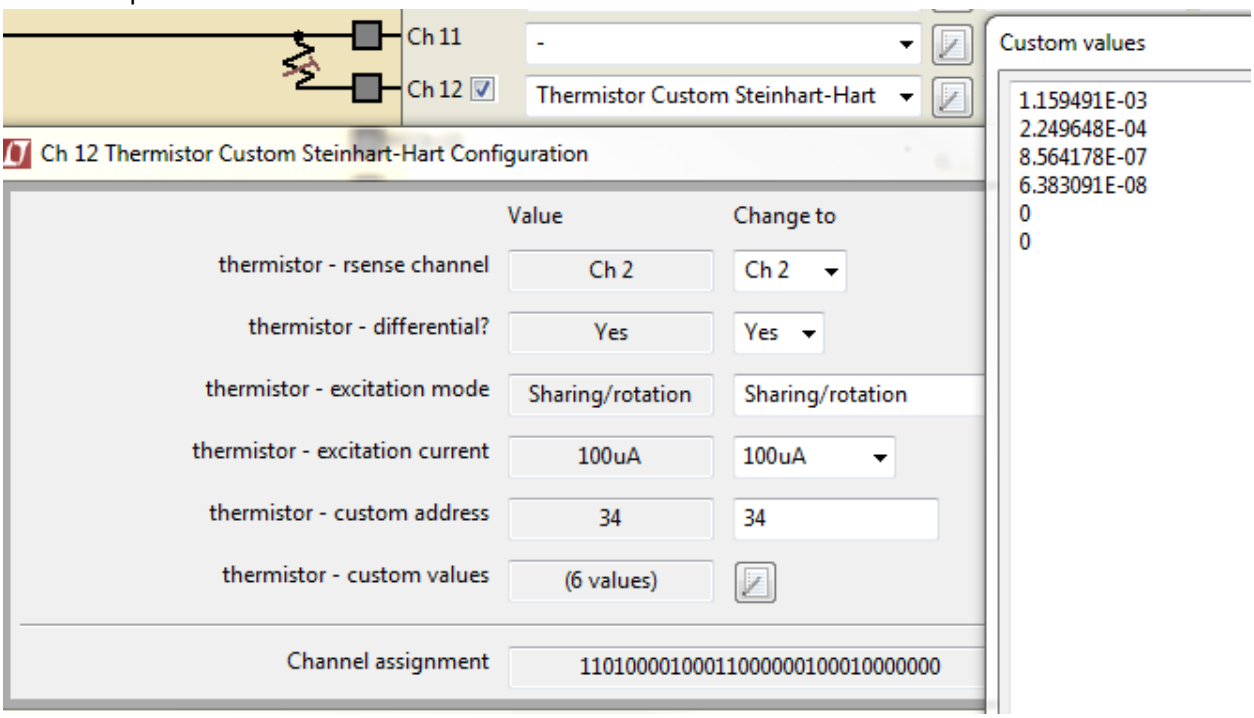


Figure 15: The new custom sensor configuration window shows the required offset to not overwrite our existing table data

We'll share the same sense resistor on channel 2 as before, but in this case since we are already storing table data in user RAM starting at position 0, we'll need to offset the location of the new coefficients to prevent any overlap. Since we had 34 entries in the table starting at address 0, we'll put the new coefficients right after them at position 34 as shown in Figure 15. Using the verify configuration tool of the GUI we can confirm we have no conflicts.

Finally, looking back at the datasheet we find an R-T table that we can use to verify our coefficients work – for example, 1070Ω should give 85°C – setting up a bench test with a grab bag 5% resistor we can

check our results:

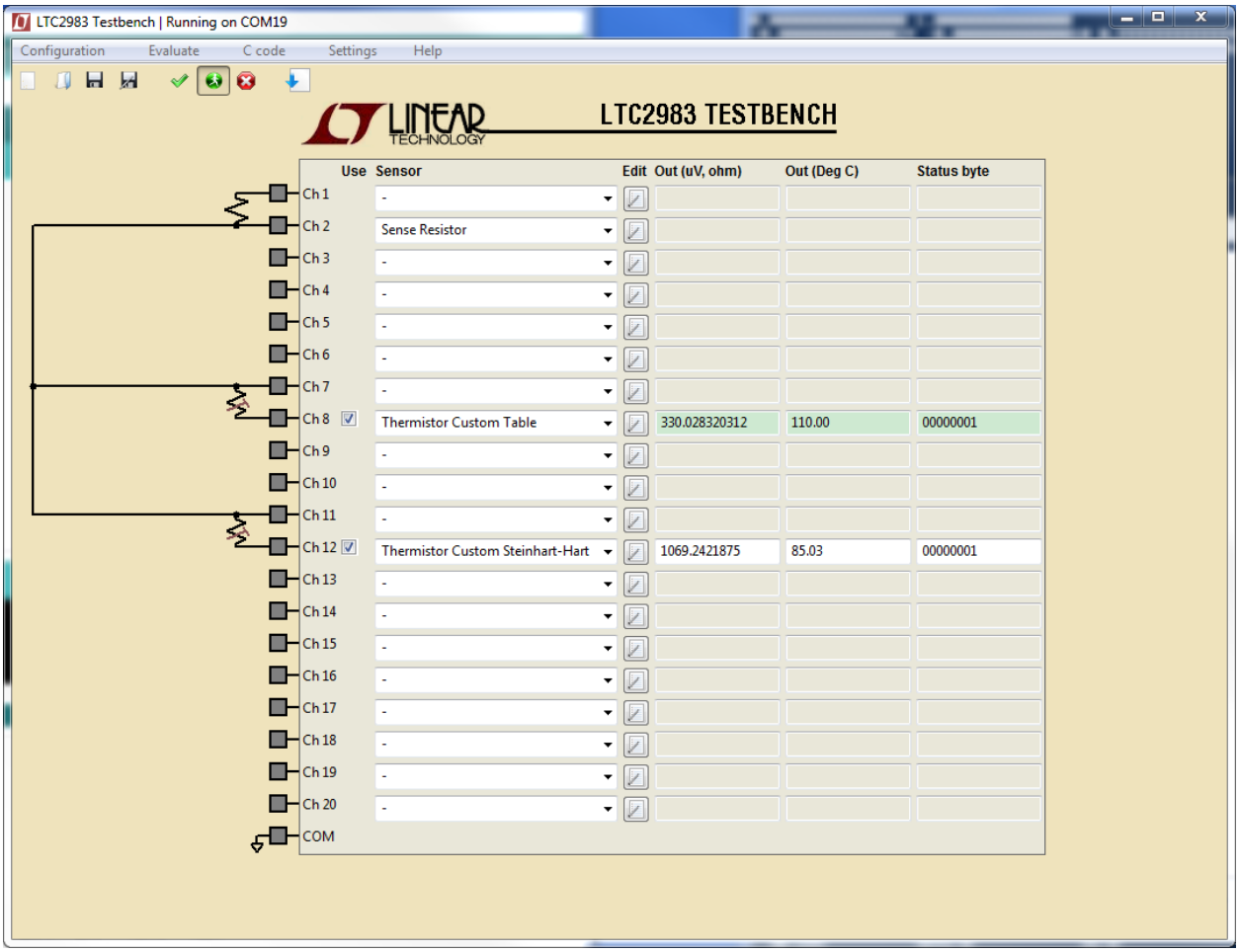


Figure 16: Two custom thermistors - two valid measured resistance to temperature conversions

Looks like it's working!

Conclusion

In this post we've taken a look at two ways of entering custom thermistor data into the Temp-to-Bits family. One noticeable result is that resistance-temperature tables are by far more straightforward to adapt and enter, while Steinhart-Hart coefficients use far less memory and may give better results due to the interpolation methods used.

It should also be mentioned that fitting methods can be used to calculate Steinhart-Hart coefficients from resistance-temperature table data. In some ways this can be very useful if multiple types of custom sensors are used in a single system as the memory saved can be large.

For questions about custom thermistor setup and the Temp-to-Bits family, please do not hesitate to get in touch.

Reference links:

http://www.vishay.com/docs/29049/ntcle100.pdf

http://www.murata.com/~/media/webrenewal/support/library/catalog/products/thermistor/ntc/r44e.ashx

http://cds.linear.com/docs/en/datasheet/2983fc.pdf

http://www.linear.com/solutions/5482