I'm not robot

reCAPTCHA

Continue

Rtd resistance to temperature conversion formula

An RTD resistance can be converted into temperature using standard tables that gives values of temperatures for any given resistance value of the RTD. The table below shows temperature versus resistance data in degree celsius with temperature coefficient of resistance of: 0.003916 ohm/ohm/°C. Fundamental Interval 39.16 ohms 0C 0 100 200 300 400 500 600 0C 0 100 139.16 177.14 213.95 249.59 284.04 317.33 0 10 103.97 143.01 180.88 217.57 253.09 287.43 320.59 10 20 107.93 146.85 184.60 221.17 256.57 290.79 323.84 20 30 111.87 150.68 188.31 224.77 260.05 294.15 327.08 30 40 115.81 154.49 192.01 228.35 263.51 297.50 40 50 119.73 158.30 195.70 231.92 266.96 300.83 50 60 123.64 162.09 199.37 235.47 270.40 304.15 60 70 127.54 165.87 203.03 239.02 273.83 307.47 70 80 131.42 169.64 206.69 242.55 277.25 310.76 80 90 135.30 173.40 210.33 246.08 280.65 314.05 90 100 139.16 177.14 213.95 249.59 284.04 317.33 100 Ω/0C Ave 0.390 0.380 0.368 0.356 0.345 0.333 0.325 An example of RTD temperature/resistance determination from the standard tables: Example 1: Calculate the resistance of an RTD thermometer when the temperature is: a) 0 degree Celsius b) 100 degree Celsius c) 50 degree Celsius d) 75 degree Celsius Solution: (a) From the Temperature vs Resistance data tables, At 0 degree Celsius, Resistance = 100 ohms (b) At 100 degree Celsius, Resistance = 139.16 ohms (c) At 50 degree Celsius, Resistance = 119.73 ohms (d) At 700C, resistance = 127.54ohms At 800C, resistance = 131.42ohms At 750C, let resistance = X Using interpolation method, we have: (70 - 80)/(75 - 80) = (127.54 - 131.42)/(X - 131.42) -10/-5 = -3.88/(X - 131.42) -10(X - 131.42) = 19.4 X = 131.42 - 1.94 = 129.48 Example 2: An RTD with a temperature coefficient of resistance of 0.390 has a resistance of 100Ω at 00C and a resistance of 139.16Ω at 1000C . If the RTD is used to measure the temperature of water in a water bath heater, what is the temperature of the water bath if the resistance of the RTD is 120Ω? Solution: At 0 degree C , RTD resistance = 100Ω 100 degree C, RTD resistance = 139.16Ω Let X be the temperature of the water when RTD resistance is 120Ω Using interpolation, we have: (X - 0)/(100 - 0) = (120 - 100)/(139.16 - 100) X/100 = 20/39.16 X = (20 X 100)/(39.16) X = 51.07250C Hence temperature of the water bath is 51 degree C For more information on RTD Sensors, check out: RTD Conversion tool: R to T This is used to calculate temperature value of RTD sensor from known resistance. This calculation tool used to find out the temperature value of a RTD sensor with known resistance. Calculate Temperature from Resistance Formula: Resistive Temperature Detectors (RTDs) relate resistance to temperature by the following formula: RT = Rref[1 + α(T − Tref)] Where, RT = Resistance of RTD at given temperature T (ohms) Rref = Resistance of RTD at the reference temperature Tref (ohms) α = Temperature coefficient of resistance (ohms per ohm/degree) Example: The following example shows how to use this formula to calculate the resistance of a PT100 RTD with a temperature coefficient value of 0.00392 at a temperature of 35 degrees Celsius: Assuming Temperature Reference = 0 Degrees For PT100 RTD the Rref = 100 RT = 100 Ω[1 + (0.00392)(35 − 0)] RT = 100 Ω[1 + 0.1372] RT = 100 Ω[1.1372] RT = 113.72 Ω For Temperature to Resistance conversion also the same above formula applies. The above given is a basic equation only for RTD calculation. Note: 1.The above RTD calculation tool designed for a standard PT100 sensor. 2.If you are interested to calculate for a different RTD then change the fixed constant values as per the sensor type. Convert a known Resistance Thermometer resistance in ohms to a temperature reading in °C. Convert a temperature in °C to Resistance Thermometer resistances. Results: A Pt100 reading {{ temperature }}°C will be {{ pt100_ohms }} A Pt500 reading {{ temperature }}°C will be {{ pt500_ohms }} A Pt1000 reading {{ temperature }}°C will be {{ pt1000_ohms }} NoteThe calculation uses the coefficients as defined in IEC 60751. Although the convertor has been tested and checked please only use it for reference. Resistance thermometers, also known as resistance temperature detectors or RTDs use resistance to measure temperature. As temperature changes so does the resistance of the resistance thermometer. There are several different standard types of resistance thermometer, depending on the type will determine the ohms at 0°C. Resistance Thermometer Types RTD TypeResistance in ohms (Ω) at 0°CPt100100Pt500500Pt10001000 The above calculator enables you to work out what the temperature should be at a certain resistance and what the resistance should be at a specified temperature. If you would like further assistance with your Resistance Thermometer or are unsure of what resistance thermometer is the best option for your process, please contact us to discuss your requirements and our technical team will work with you and advise the appropriate solution. View Resistance Thermometers View Other Resources Contact Us © 2020-21 - Peak Sensors Ltd - All Rights Reserved - Company Reg. 3386191. - Vat No. GB695025618. Resistance is the electrical property of a material that opposes the flow of electricity through it. This degree of resistance to electricity is defined by another property of the material called Resistivity. The resistivity of a material is defined as the resistance to current flow between the opposite faces of a unit cube of the material (ohm per unit length). Hence the resistance R of a component is expressed by: Where: R = Resistance of the component in Ohms L = Length of the component A = Cross sectional Area of the component ρ = Resistivity of the material To use the above formula, L and A must be in compatible units The resistivity, ρ, and resistance, R, are temperature dependant, usually having a positive temperature coefficient (resistance increases as temperature increases), except for some metal oxides and semiconductors which have a negative temperature coefficient. The metal oxides are used for making thermistors. The variation of resistance with temperature is given by: R(T2) = R(T1)[1 + αΔT] Where: R(T2) = Resistance at temperature T2 R(T1) = Resistance at temperature T1 α = Temperature coefficient of resistance ΔT = (T2 – T1), Temperature difference between T1 and T2 The principle of temperature change with resistance is what is utilized in Resistance Temperature Detectors, RTDs to sense and measure temperature in many industrial applications. RTD resistance versus temperature tables are used to determine the resistance of an RTD at any given temperature. See How to Convert RTD Resistance to Temperature. Next we take a look at how to convert resistance to temperature: Problem 1: What is the resistance of a platinum resistor at 250°C, if its resistance at 20°C is 1000 Ω? Take α = 0.00385 per degree C Solution: R(T1) = 1000Ω , T1 = 20 degree C, T2 = 250 degree C and α for platinum = 0.00385 Hence R(T2) = 1000[1+ (250 – 20)*0.00385] = 1,885.5Ω Problem 2: A tungsten filament has a resistance of 1998 Ω at 20°C. What will its resistance be at 263°C? Take α for tungsten = 0.0045. Solution: R(T1) = 1998Ω , T1 = 20 degree C, T2 = 263 degree C and α for tungsten = 0.0045 Hence R(T2) = 1998[1+ (263 – 20)*0.0045] = 4,182.813ΩProblem 3: What is the coefficient of resistance per degree Celsius of a material, if the resistance is 2246Ω at 63°F and 3074Ω at 405°F? Solution: Recall that: R(T2) = R(T1)[1 + αT] This we can re-arrange to give: α = [R (T2)/R (T1) – 1]/ΔT Now, R(T2) = 3074Ω, R(T1) = 2246Ω, T2 = 405 degree F, T1 = 63 degree F ΔT = 405 – 63 = 342 degree F But degree C = (F - 32)*5/9 Therefore 342 degree F = [(342 – 32)*5]/9 = 172.222 degree C α = [(3074/2246) – 1]/172.222 = 0.00214 per degree C This Calculator with convert Pt100, Pt500 & Pt1000 Ohms to degrees centigrade. The conversion is done using the common IEC rtd values. Only use this calculator for reference as we cannot guarantee the results to be error free. Joined Apr 28, 2004 Messages 32 Helped 0 Reputation 0 Reaction score 0 Trophy points 1,286 Activity points 360 pt100 formula Hi ! Does someone knows where I can found the conversion formula of RTD from resistance to temperature and vice versa ? Thanks Joined Feb 5, 2002 Messages 839 Helped 58 Reputation 116 Reaction score 9 Trophy points 1,298 Location Pakistan Activity points 7,800 rtd formula I think it is OMEGA. a company that makes temperature stuff. you can find its website on google easily. it has all the charts in pdf. I am on my palmpilot at this time, otherwise would have given you the full url. Joined Nov 26, 2004 Messages 1,582 Helped 384 Reputation 768 Reaction score 87 Trophy points 1,328 Activity points 19,971 rtd conversion table The ASTM standard E1137 deals with RTD's. You can buy the standard from ASTM Omega offers their "The Temperature Handbook", which contains conversion charts. It is free, but shipping is not cheap. Try using this .pdf. I don't know how accurate these coefficients are. I can check them on Monday, when I get back to work. Note that for temperatures above 0C, the formulas are simpler. www.atpsensor.com/pdfs/rtd.pdf Joined Feb 5, 2002 Messages 839 Helped 58 Reputation 116 Reaction score 9 Trophy points 1,298 Location Pakistan Activity points 7,800 rtd conversion here is the complete table. Joined Apr 26, 2005 Messages 1 Helped 0 Reputation 0 Reaction score 0 Trophy points 1,281 Activity points 1,288 pt100 resistance formula Hi ! Does someone knows where I can found the conversion formula of RTD from resistance to temperature and vice versa ? Thanks [{(0.00389)(PT100)}deg. C]+PT100 = Rt, you can work it out in this formula by substituting the RTD type like PT200, PT500 Joined Apr 8, 2012 Messages 201 Helped 2 Reputation 4 Reaction score 2 Trophy points 1,298 Activity points 0 Re: pt100 resistance formula [{(0.00389)(PT100)}deg. C]+PT100 = Rt, you can work it out in this formula by substituting the RTD type like PT200, PT500 In the equation {(0.00389)(PT100)}deg. C]+PT100 = Rt what is the value of PT100? PT100/PT1000 temperatures calculation suffers from accuracy issues for large sub-zero temperatures. UliEngineering implements a polynomial-fit based algorithm to provide 58.6 \mu{\degree}C peak-error over the full defined temperature range from -200 {\degree}C to +850 °C.Use this code snippet (replace pt1000_ by pt100- to use PT100 coefficients) to compute an accurate temperature (in degrees celsius) e.g. for a resistane of 829.91 Ω of a PT1000 sensor.from UliEngineering.Physics.RTD import pt1000_temperature # The following calls are equivalent and print -43.2316359463 print(pt1000_temperature("829.91 Ω")) print(pt1000_temperature(829.91)) You install the library (compatible to Python 3.2+) using$ pip3 install -U UliEngineeringThe problemThe formula to compute PT100/PT1000 resistance from temperature is well-known (see e.g. Thermometrics): R_t\,=\,R_0\cdot(1\,+\,A\cdot t\,+\,B\cdot t^2\,+\,C\cdot (t-100)\cdot t^3)where t is the temperature, R_0 is the zero-{\degree}C resistance (i.e. 100 Ω for PT100 and 1000 Ω for PT1000).The remaining parameters A, B and C depend on the temperature standard in use and might be measured by the sensor manufacturer for additional accuracy. For the ITU-90 standard they equal (see code10.info) \begin{array}{lll}A&=&3.9083\cdot10^{-3}\\B\,&=&\,-5.7750\cdot10^{-7}\\C\,&=&\,\begin{cases}-4.1830\cdot 10^{-12}&\text{for}\ t&\lt& 0{\degree}C\\0.0&\text{for}\ t &\geq&\;0\;{\degree}C\end{cases}\end{array}Furthermore, C is set to 0 for temperatures > 0°C, simplifying the formula to: R_t\,=\,R_0\cdot(1\,+\,A\cdot t\,+\,B\cdot t^2)It it obvious that, given this information, the resistance at a given temperature can be calculated without any error term.The problems arise when attempting to solve the equation in the general case.The formula for t\,\geq\,0 is easily solvable (source for the shown form): t\,=\,\frac{-R_0\cdot\,a\,+\,\sqrt{R_0^2\cdot a^2\,-\,4\cdot R_0\cdot B\cdot (R_0-R_t)}}{2\cdot R_0\cdot b}Although the formula for t\lt0 {\degree}C has exact algebraic solutions, it is so large that it probably won't fit on your screen. It therefore can be considered infeasible to implement this formula as one would sacrifice simplicity and speed for an exact solution.However, it is inherently true that a given sensor can only work up to a certain precision. Therefore, an approximate solution with sufficient precision is sufficent for all practical applications.The solution: Fit a polynomial to the error functionAt first I thought of implementing an iterative function that refines an initial temperature guess. While this approach would certainly work as an exact error function is available, it does not scale well and does not have deterministic runtime.Let's visualize the error function being present without any correction term. As the C term reduces to 0 for t\geq0{\degree}C, we are interested only in the range from -200 °C to 0°C (PT100/PT1000 sensors are not defined below -200°C in the relevant standards, but in principle this method extends down to 0°K).The method I'm about to present — as well as all tools neccessary to validate it — is implemented in my UliEngineering library in the UliEngineering.Physics.RTD module.Using matplotlib and UliEngineering, generating this plot is possible in only 12 lines of code:import matplotlib.pyplot as plt from UliEngineering.Physics.RTD import * import numpy as np plt.style.use("ggplot") plt.gcf().set_size_inches(12, 4) plt.title("Uncorrected PT1000 error") plt.ylabel("Relative error [°C]") plt.xlabel("Absolute actual temperature [°C]") # Create 1M datapoints of reference temperatures temp = np.linspace(-200.0, 0.0, 1000000) # Plot deviation: reference temperature - calculated temperature x, y, _ = checkCorrectionPolynomialQuality(1000.0, temp, poly=noCorrection) plt.plot(temp, y) plt.savefig("PT1000-uncorrected.svg")We call checkCorrectionPolynomialQuality() with the canned noCorrection polynomial which always evaluates to zero: In this configuration, the function computes the resistances from our reference temperatures and the re-computes actual reference temperature values from said resistances. It returns three values: – A numpy array of resistances, corresponding to our reference temperatures – A numpy array of difference from the reference temperature at any given resistance / temperature, in °C – A peak-absolute value scalar quality indicator (i.e. what's the worst error to expect)It is easily observable that, while the error reaches almost 2.5°C at -200°C, it is montonous and uniformly continous.Our approach therefore comprises of fitting a polynomial on this function, minimizing the difference from the reference temperature using np.polyfit.This algorithm is available in the computeCorrectionPolynomial() function from UliEngineering. It has been determined experimentally that a 5th-degree polynomial exhibits results that are significantly better that those of higher- or lower-degree polynomials. Nevertheless, the function lets you specify a custom degree if you intend to experiment with the parameters.plt.gcf().clf() # Clear current figure plt.gcf().set_size_inches(12, 4) plt.title("Polynomially corrected PT1000 error") plt.ylabel("Relative error [°C]") plt.xlabel("Absolute actual temperature [°C]") # Compute correction potential mypoly = computeCorrectionPolynomial(1000.0) # Plot deviation: reference temperature - calculated temperature _, y, pp = checkCorrectionPolynomialQuality(1000.0, temp, poly=mypoly) plt.plot(temp, y) plt.savefig("PT1000-corrected.svg") print("Peak-to-peak error: {0}".format(pp))It can be clearly seen that the remaining error is extremely small now: Its peak-absolute value over the entire defined temperature range is only 58.6\,\mu{\degree}C. This is considered sufficient for all practical applications besides reference temperature measurement in metrology (and even there a few tens of micro-degrees.For a simple (and relatively fast-to-compute) 5th-degree polynomial, these results are astonishingly good. The current git version of the UliEngineering library implements this algorithm by using precomputed polynomials that are automatically selected if you pass R_0=100.0 or R_0=1000.0 to ptx_temperature(), which is internally called from pt100_temperature() and pt1000_temperature(). For other R_0 values you'll need to manually compute the polynomial and pass it to ptx_temperature().Of course, NumPy arrays and similar objects can also be passed to the functions in UliEngineering.