RTD (Resistance Temperature Detector) Simulator

Introduction:

This document details how a Resistance Temperature Detector (RTD) simulation can be achieved using digital potentiometers. This design is now used in many of our rigs.

Resistance temperature detectors (RTDs) are typically made of nickel, copper, or platinum. One of the most linear, stable, and reproducible RTDs is the 100 or 1000 platinum type, sometimes referred to as a platinum resistance thermometer, or PRT. These devices have a near-linear positive temperature coefficient from –200°C to 800°C, which, along with its other attributes, have established the RTD as a de facto industry standard. Temperature is determined by measuring resistance and then using the RTD's resistance vs. temperature characteristics to extrapolate temperature.

Requirement:

There was the need for a cheap but stable RTD simulator to help perform HSIT work.
We had a very interesting requirement in one of our project. The Test Stand had to simulate the RTD value. In this case we need to send a voltage/current equivalent of a known temperature value.

The target provided only two pins (HI and LO, refer the Figure 1). The HI was providing the excitation voltage to the sensor and LO is the output from the sensor.

Typically this is how the sensor is connected to UUT (Unit Under Test).

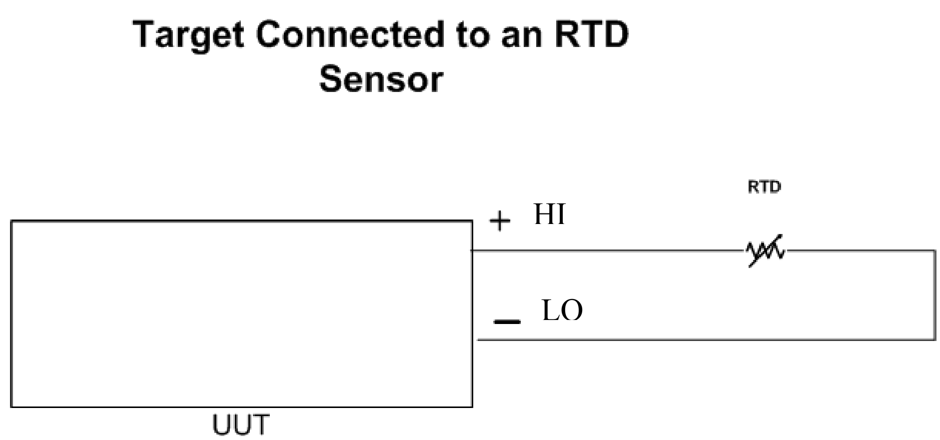


Figure 1: Normal connection of Temperature Sensor to the Target

In a test environment, it was required to emulate the behavior of the RTD sensor that is used in the real application.

Solutions:

A Solution we could work out was, allow the user to enter the Temperature value that he/she wants and simulate it using hardware. The Solution seems to be simple, but how do we implement it?

First there was a requirement to understand the working of the Temperature Sensor for this. The RTD Sensor requires an excitation voltage. Based on the temperature, it sends back an equivalent current/voltage value (Figure 2).

Now to simulate this, the Test Stand needs to read the excitation voltage sent by the UUT. For this we used an Analog Input Reader (16-bit ADC).

The Test Stand Software accepts temperature input to be simulated from the user, reads the excitation voltage and based on the conversion formula, would send an appropriate voltage back.

To send the output of sensor, we used an Analog Output Card (13 bit DAC).

The connection would be as follows (Figure 2).

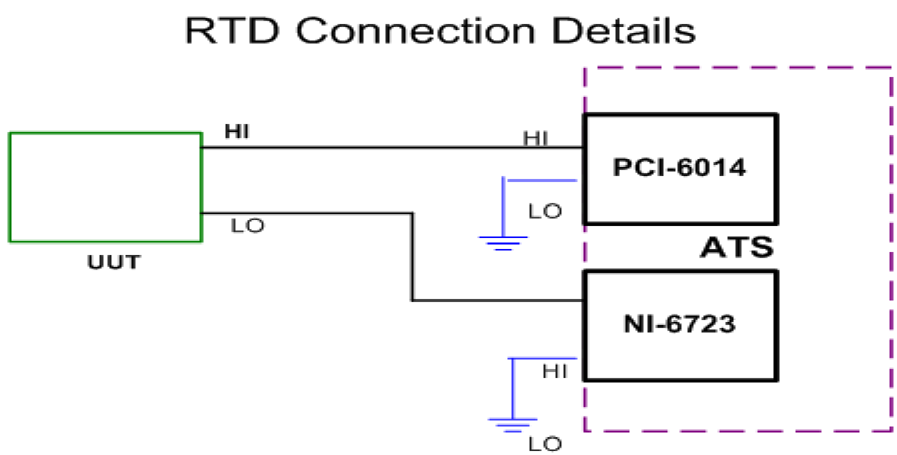


Figure 2: Test stand setup to simulate the Temperature sensor

Notes: CI-6014 card reads excitation from UUT and NI-6723 card outputs the simulated temperature as voltage output.

However there was a major flaw in this. Since the GND connection for each of the Analog device was different, each day, the output of RTD simulator used to be different. We tried fixing this by connecting all groundings to a common GND.

This did reduce the error to a great extent, but still we were unhappy, as during certain times there were offsets while measuring.

Alternative Solutions:

Now we thought, enough was enough; we will place a manual system as an RTD simulator. We went in for very high precision variable resistors. Bought a few rotary switches and created a manual system, which was exactly like Figure 1. This one is working very well in Test stand 1. However the disadvantage of this is that not all the values can be driven. As this is a manual system, the tester has to have adequate patience to change the resistance value. Above all, if some one is interested to measure the actual resistance, then we have to stop the power supply to the circuit and measure it.

Final Automated Solution:

During implementation and testing of the automation phase, we came across digital potentiometers from Analog Devices and Maxim Company.

We once again studied our requirements and tried finding matching Digipot. (Digipot is nothing but a Digital Potentiometer) Interesting thing about Digipot is, it can keep incrementing/decrementing the resistance using a step value and a higher resolution can be achieved in step values.

It is a compact device having 256 viper positions, programmable using SPI bus. It is very much same as a mechanical potentiometer. However the total resistance would be deciding factor in choosing the right Digipot.

We wanted to simulate the RTD with the following characteristics: -

RTD type:	RTD sensor in accordance with MS28034. (Two types)
Ranges required:	Type A: 0 – 130 °C., Type B: -40 to +70 °C etc
Sensitivity:	Type A: 90.38 ohms resistance at 0 °C Type B: 500 ohms resistance at 0 °C

We selected Analog Devices' AD8400 family digipot. We bought many dozens of them and they cost very little. Each pot can be communicated with an SPI (Serial Peripheral Interface) Bus. (This project gave us a chance to work on SPI.). Our earlier design favored a micro-controller controlled Digipots. However later we discovered that we could straightaway use PC's parallel port to communicate with Digipots.

Implementation of Final Solution

a) Stage one: Evaluation

We formed a small team, procured a breadboard and the required Digipots. The team went on experimenting with Digipots using a PC. Initially results with the breadboard were not very encouraging and often there were no hopes. (But somewhere in my mind I was always dreaming that it would work.) We had many discussions with the Hardware Team. Re-wrote our requirements, explained about the actual sensor. Then a small working design was evolved.

b) Stage two: Practical Circuit

The initial circuit had many problems and the result was not stable. However the Hardware team worked overnight and improved its design. Finally there was a circuit that we could use to simulate the required outputs. We had a small software team that accumulated knowledge on SPI interfacing in stage one. We took them back again and set some aggressive and ambitious goal.

The team came out with a solution on how we could communicate simultaneously with all the Digipots using very few communication lines, so that one parallel port could be used to simulate at least 4 different RTDs.
The software has a small footprint and smart. It generates clock pulses and transmits them over a designated line on the parallel port. On each clock pulse the software transmits a bit of data to the Digipots connected. There is one more line to select the chip (CS).

The connection is explained as below (Figure 3) for one digipot.

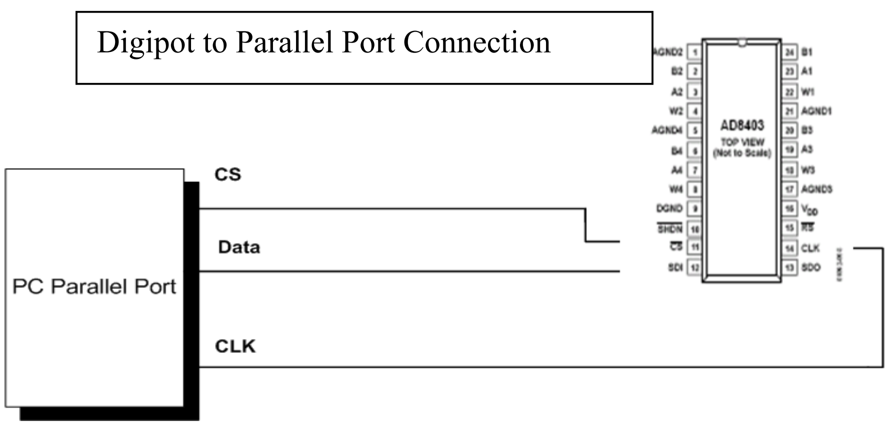


Figure 3: A simple connection to Digipot over an SPI interface

c) Stage three: Integration into the Project

Finally we managed to get the complete circuit done for a small price from a local vendor and we got our first working Hardware. We have shown in Figure 5 one such circuit among the 4 different RTDs that we simulated.

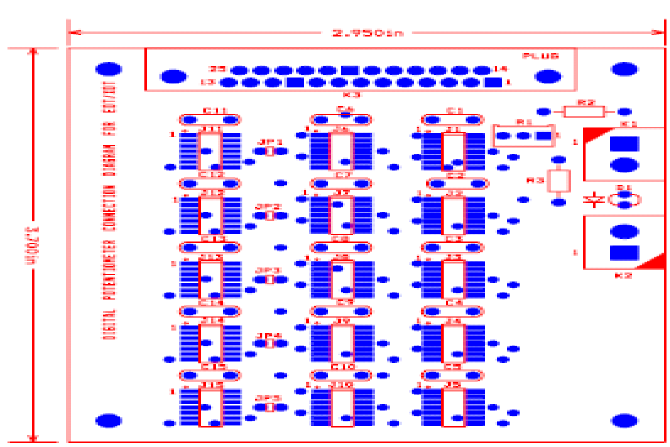
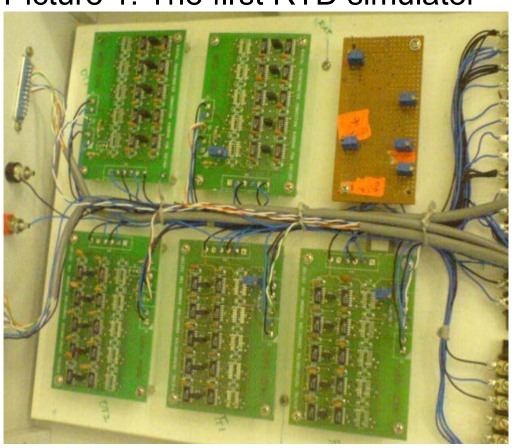


Figure 5: A complete arrangement of Digipots for simulating one type of RTD

We wrote a set of drivers for the PC side and integrated that into Test stand Software.

Picture 1: The first RTD simulator



Picture 2: An array of Digpots arranged for a high-resolution output.