Sensor Test Sequence

1.1 Purpose

Sensor Tests demonstrate the Temperature measurements captured by Thermistor, RTD and Thermocouple sensing devices using C Series Temperature Input Resources (cDAQ-9211, cDAQ-9217), C Series voltage Input Resources (cDAQ-9215) and TestScale Analog Input Resource (TS-15100, TS-15200). This example sequence can be executed in a custom Python sequence script using the measurement libraries written in Python.

NOTE: TestScale does not support RTD and Thermocouple Temperature measurements.

Example File Location

"<venv>\Lib\site-packages\nipcbatt\pcbatt_automation\sensor_tests"

1.2 Highlighted Features

- Thermistor Test - cDAQ
  - Captures temperature on Analog Input resource with voltage excitation supplied by Analog Output resource of cDAQ Chassis. Libraries used in the example are "***DcVoltageGeneration()***" and "***TemperatureMeasurementUsingThermistor()***".
- Thermistor Test - TestScale
  - Captures temperature on Analog Input resources with voltage excitation supplied by Power resource of TestScale Chassis. Libraries used in the example are "***PowerSupplySourceAndMeasure()***" and "***TemperatureMeasurementUsingThermistor()***".
- RTD Test
  - Captures temperature on Analog Input resources. Library used in the example is "***TemperatureMeasurementUsingRtd()***".
- Thermocouple Test
  - Captures temperature on Analog Input resources. Library used in the example is "***TemperatureMeasurementUsingThermocouple()***".
- Turn Off all AO Channels
  - Powers down all analog output channels by configuring the output voltage as 0 Volts. Libraries used in the example is "***DcVoltageGeneration()***".
- Turn Off Power Channel
  - Powers down Power channel by configuring the output voltage as 0.1 Volt. Libraries used in the example is "***PowerSupplySourceAndMeasure()***".

Refer this folder for more details on each Measurement library "\<venv>\Lib\site-packages\nipcbatt\pcbatt_library".

1.3 Prerequisites

- Python – 3.9 to 3.12
- DAQmx Driver – 2023 Q3 or later

1.4 Setup Diagram

Represents the hardware setup used in this example sequence. Pin Outs of each resource is added below.

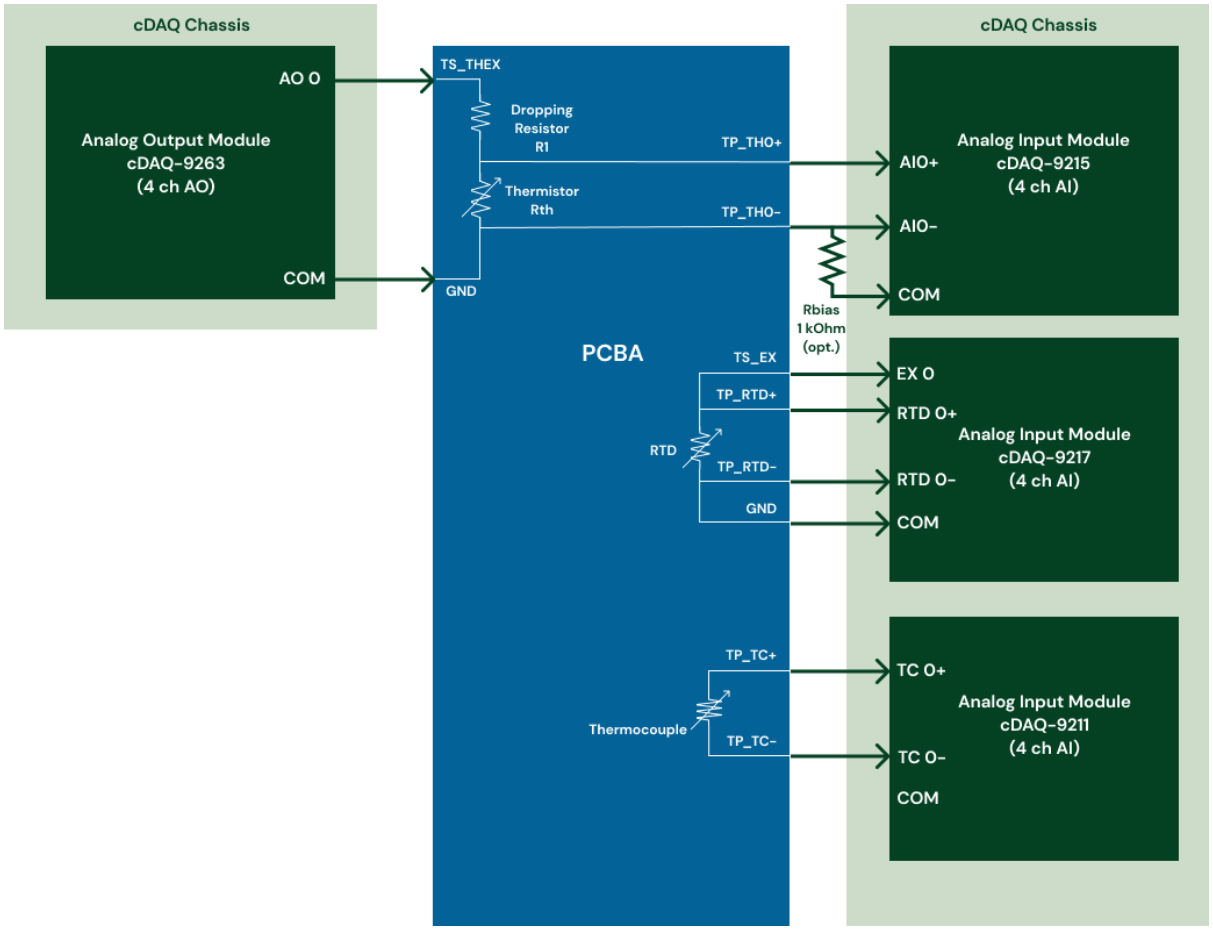


Figure 1 Setup Diagram – cDAQ

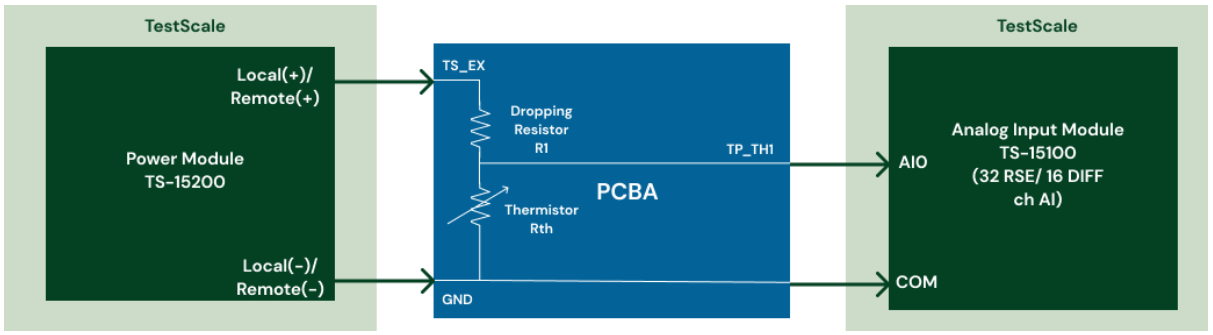


Figure 2 Setup Diagram – TestScale

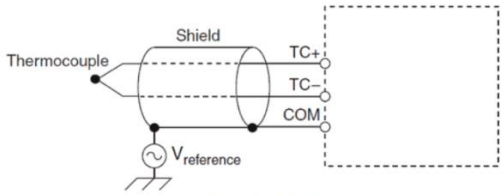
Note:

1. For cDAQ Thermistor measurements, the Analog Input terminal configuration is Differential and floating. To avoid common mode voltage errors, NI recommends connecting a Direct wire/Bias resistor between AIx- to COM based on the ground differences between the sensing and the measurement device. The value of the bias resistor could be from 1 Kohm to

10 Kohm for a good reference. For more details, refer [Measurement System Types and Signal Sources](#).

2. For TestScale Thermistor measurement, the Analog Input terminal configuration is RSE by default.

3. For shielded Thermocouple measurements, connect the COM terminal of your device to the shield and the shield to a common-mode voltage reference of the thermocouple. A common-mode voltage reference is a voltage that is within ±1.2 V of the common-mode voltage of the thermocouple. If you are using a floating thermocouple or a thermocouple within ±1.2 V of earth ground, connect COM and the shield to earth ground. The shield grounding methodology can vary depending on the application. Refer to the below figure for an illustration of a typical shielding configuration.

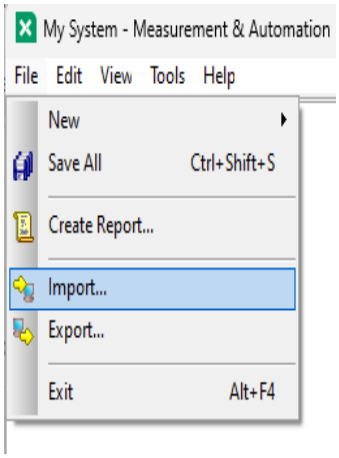


Connecting a Shielded Thermocouple

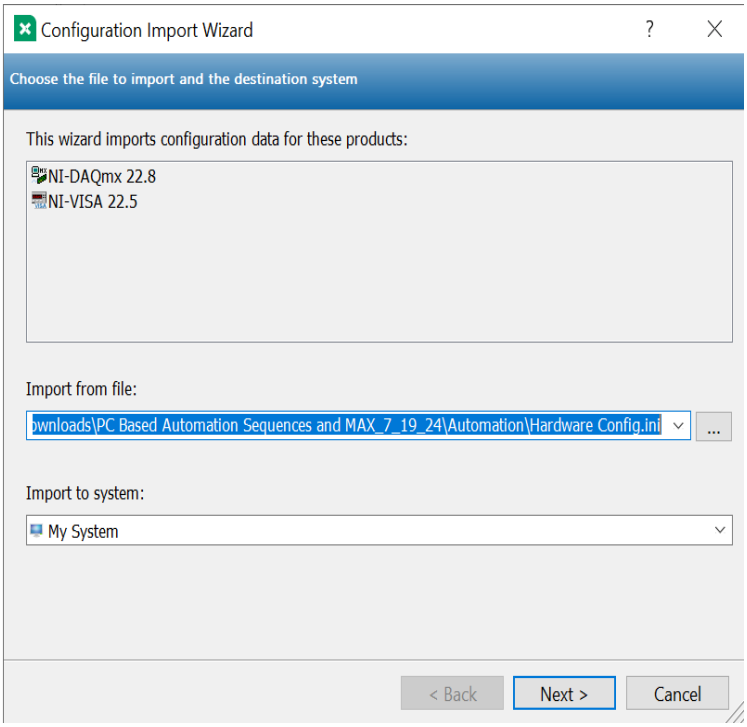
1.5 How to run this Example?

Complete the following steps to run the sequence.

1. First, we must configure the virtual channels which will be used by the Python script names mentioned in ***thermistor_test_cdaq.py or thermistor_test_testscale.py or rtd_test.py or thermocouple_test.py:***
    a. A hardware configuration file for NI-MAX is required to run this example. The configuration file contains a set of predefined global channel names which are used by the nidaqmx driver to communicate with the Python scripts.
    b. To import the "Hardware Config" open NI-MAX.
    c. Click on File -> Import to open the Configuration Import Wizard



d. In the Configuration Import Wizard window, click on the Browse (...) button and locate the *Hardware Config.ini* file in *"\<venv>\Lib\site-packages\nipcbatt\pcbatt_automation".* Then click on *Next -> Import -> Finish*



e. NI-MAX now holds the same virtual channel name references contained in the examples provided

The ***thermistor_test_cdaq.py, thermistor_test_testscale.py, rtd_test.py and thermocouple_test.py*** files will create log files in the form of a simple text (.txt) file. The default file path it will use is

"C:\\Windows\\Temp\\Thermistor_test_cDAQ.txt"
*"C:\\Windows\\Temp*Thermistor_test_TestScale*.txt"*
*"C:\\Windows\\Temp*RTD_test*.txt"*
*"C:\\Windows\\Temp*Thermocouple_test*.txt"*

If you wish to create this file in a different location on your PC, change the value of the string variable *DEFAULT_FILEPATH*.

2. Open the Python scripts ***thermistor_test_cdaq.py, thermistor_test_testscale.py, rtd_test.py, thermocouple_test.py, turn_off_all_ao_channels.py, turn_off_all_power_channesl.py and sensor_test_main_sequence.py*** in your IDE or text editor of choice. The following steps are performed within ***sensor_test_main_sequence.py:***
   a. ***Thermistor Test cDAQ*** - Demonstrates Thermistor temperature measurements of cDAQ Analog Input resource with cDAQ9263 (Analog Output) Voltage Excitation Below are the steps included in the test.
      i. Initialize the DC Voltage Generation and Temperature Measurement libraries using Thermistor Libraries by creating the instances of ***DcVoltageGeneration()*** and ***TemperatureMeasurementUsingThermistor()*** classes and then using ***initialize()*** methods on each object.
      ii. Configure the DC Voltage Generation to source 5 Volts in specified Analog Output channels by calling the ***configure_and_generate()*** method using the

default parameters. In this example, a DC voltage of 5 Volt is generated for a duration of 0.1 seconds.

iii. Fetches the captured temperature waveforms and provides temperature measurements with ***TemperatureMeasurementUsingThermistor()*** class instance by calling the ***configure_and_measure()*** method.

iv. Use the ***close()*** methods on both instances to stop generation and measurement.

Refer the help/comments in the sequence for more details to know more about trigger configuration.

b. ***Thermistor Test Testscale*** - Demonstrates Thermistor temperature measurements of TS-15100 Analog Input resource with TS-15200 Power resource Voltage Excitation.Below are the steps included in the test.

i. Initialize Power Supply and Time Domain Measurement libraries by creating instances of ***PowerSupplySourceAndMeasure()*** and ***TemperatureMeasurementUsingThermistor()*** classes and then using ***initialize()*** methods on each object.

ii. Configure the Temperature Measurement using Thermistor class.

iii. Configure the Power Supply Source and start sourcing. Here the example **supplies 5V** for 0.1 seconds. (in the backend, Power Supply resource sends the Trigger in the backplane once the Source started which in turn starts the measurement in Analog Input resource).

iv. Fetches the captured temperature waveforms and provides temperature measurements.

v. Use the ***close()*** methods on both instances to stop generation and measurement.

Refer the help/comments in the sequence for more details to know more about trigger configuration.

c. ***RTD Test*** - Demonstrates RTD temperature measurements cDAQ-9217 Analog Input Resource. Below are the steps included in the test.

i. Initialize Temperature RTD Measurement Library by creating instance of ***TemperatureMeasurementUsingRtd***() class and then using ***initialize()*** method on each object.

ii. Configure the ***Rtd_test*** to fetch the captured temperature waveforms and provides temperature measurements by calling the ***configure_and_generate()*** method using the default parameters.
Refer the help/comments in the sequence for more details to know more about trigger configuration.

iii. Use the ***close()*** methods on both instances to stop generation and measurement.

d. ***Thermocouple Test*** - Demonstrates Thermocouple temperature measurements cDAQ-9211 Analog Input Resource. Below are the steps included in the test.

i. Initialize Temperature Thermocouple Measurement Library by creating instance of ***TemperatureMeasurementUsingThermocouple()*** class and then using ***initialize()*** method on each object***.*** .

ii. Configure the Temperature Thermocouple Measurement to fetch the captured temperature waveforms and provides temperature measurements by calling the ***configure_and_generate()*** method using the default parameters.
Refer the help/comments in the sequence for more details to know more about trigger configuration.

iii. Use the ***close()*** methods on both instances to stop generation and measurement.

e. ***Turn Off all AO Channels*** – Power downs all Analog output channels by configuring them to 0 Volts. Below are the steps included in the test.

i. Initialize the DC Voltage Generation library by creating an instance of ***DcVoltageGeneration()*** class and then using ***Initialize()*** method.

ii. Configure the DC Voltage Generation to source 0 Volts in specified Analog Output channels by calling the ***configure_and_generate()*** method using the default parameters.

iii. Use ***close()*** instance after setting AO channels to 0 Volts.

f. ***Turn Off Power Channels*** – Power down Power channel by configuring it to 0 Volts. Below are the steps included in the test.

i. Initialize an instance of ***PowerSupplySourceAndMeasure()*** class to call the ***initialize()*** method.

ii. Configure the power supply to source 0 Volts and Output status to "Disable Output when Task stopped" in specified Power channels by calling the ***configure_and_generate()*** method using the default parameters
DEFAULT_POWER_SUPPLY_SOURCE_AND_MEASURE_CONFIGURATION.

iii. Finally use the ***close()*** method.

3. When the execution completes, **review the results** on the ***.txt*** files generated by the logger at the specified location.
   a. The report has the configurations and Measurement values captured (runs with simulated instrument by default)
   b. Verify the Measurement and data formats returned by the Measurement library.

1.5.1 How to enable the Hardware?

Sensor Test sequence runs with simulated hardware by default. Once the hardware setup is available, you can do the below changes to enable running the test with the hardware.

Note : In this example, physical and global virtual channels are used to configure the terminal or pin to perform the instrument actions. Global Virtual Channels are software entities that encapsulate the physical channel along with other channel specific information—range, terminal configuration, and custom scaling. Global Channels can be created in NI-MAX and called in Measurement Libraries.

1. Skip the first step "Import Hardware Config" in section 1.5. This step imports the Simulated Hardware and creates Global Virtual Channels with Simulated instrument in NI-MAX.

2. Global channel specified in the Sensor Test sequences can be calibrated in NI MAX to provide more accurate measurements. For more details refer, Calibration section
3. Follow the below steps for each sequence. Refer "**Note to run with Hardware**" labels in the sequence.
    i. ***Thermistor Test cDAQ***
        1. Step into the "***thermistor_test_cdaq***" sequence
        2. Open NI-MAX and update the physical Channel linked to the Global Channels –**TS_THEX, TP_TH0** (called in the initialize step of DC Voltage Generation, Temperature Thermistor Measurement respectively)
        3. Review and update the "Thermistor Parameter" and "Terminal Configuration" based on the thermistor settings for accurate temperature measurements. For more details on Thermistor Parameter, refer the Temperature Thermistor Library documentation.
        4. Review the Configurations of Analog Output and Analog Input Pins for the intended use case
    ii. ***Thermistor Test Testscale***
        1. Step into the "***thermistor_test_testscale***" sequence
        2. Open NI-MAX and update the physical Channel linked to the Global Channels –**TP_TH1** (called in the initialize step of Temperature Thermistor Measurement)
        3. Open 'PS – Initialize' step and update the 'Physical Channels' input to the TestScale Power Resource pin.
        4. Review and update the "Thermistor Parameter" and "Terminal Configuration" based on the thermistor settings for accurate temperature measurements. For more details on Thermistor Parameter, refer the Temperature Thermistor Library documentation.
        5. Review the Configurations of Power and Analog Input Pins for the intended use case
    iii. ***RTD Test***
        1. Step into the "***rtd_test***" sequence
        2. Open NI-MAX and update the physical Channel linked to the Global Channels –**TP_RTD** (called in the initialize step of Temperature RTD Measurement respectively)
        3. Review and update the RTD specific parameters namely "RTD Type", "Sensor Resistance at 0 deg C (ohm)", "Resistance Configuration", "Current Excitation Source", Current Excitation (A)", "ADC Timing Mode" based on the RTD settings for accurate temperature measurements. For more details on RTD parameters, refer the Temperature RTD Library documentation.
    iv. ***Thermocouple Test***
        1. Step into the "***Thermocouple_Test***" sequence
        2. Open NI-MAX and update the physical Channel linked to the Global Channels – **TP_TC** (called in the initialize step of Temperature Thermocouple Measurement respectively)
        3. Review and update the Thermocouple specific parameters namely "Thermocouple Type", "CJC Temp (deg C)", "Enable Autozero" and "Auto Zero Mode" based on the thermocouple settings for accurate

temperature measurements. For more details on Thermocouple Parameter, refer the Temperature Thermocouple Library documentation.

v. ***Turn Off all AO Channels***
    1. Step into the "***turn_off_all_ao_channels***" sequence
    2. Update the "Physical Channels" input with Analog Output Channel in the initialize step of Turn Off All AO Channels sequence
    3. Review the Configurations of Analog Output for the intended use case

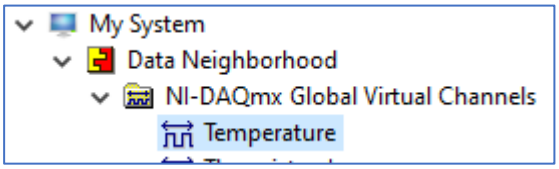
vi. ***Turn Off Power Channels***
    1. Step into the "***turn_off_power_channels***" sequence
    2. Update the "Physical Channels" input with Power Channel in the initialize step of Turn Off Power Channel sequence
    3. Review the Configurations of Analog Output for the intended use case

1.5.2 Calibration

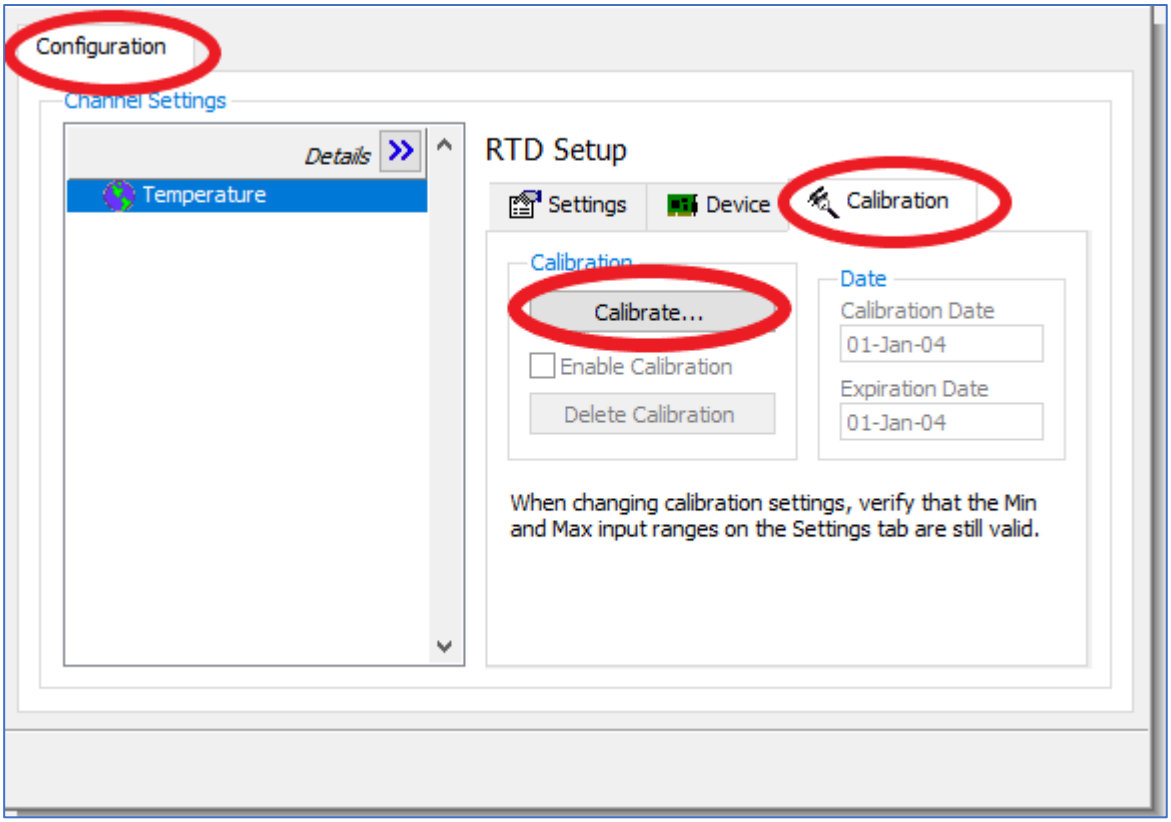
Calibrate a channel to increase the accuracy of a measurement by compensating for errors due to cabling, wiring, or sensors. NI MAX provides calibration option for DAQ devices virtual global channels.

With the use of Virtual global channels, we can calibrate input measurements individually, save the calibration, edit them later or recalibrate it. It can be useful for sensors calibrations to compensate sensors errors. To calibrate a virtual global channel, follow the below steps,

1. Open NI MAX
2. Create/Select the Virtual global channel to calibrate under **My System»Data Neighborhood»NI-DAQmx Global Virtual Channels»*YourChannelName*** from the **Configuration** view in NI MAX.



3. Select **Configuration»Calibration»Calibrate...** in the NI_DAQmx Global Channel view.

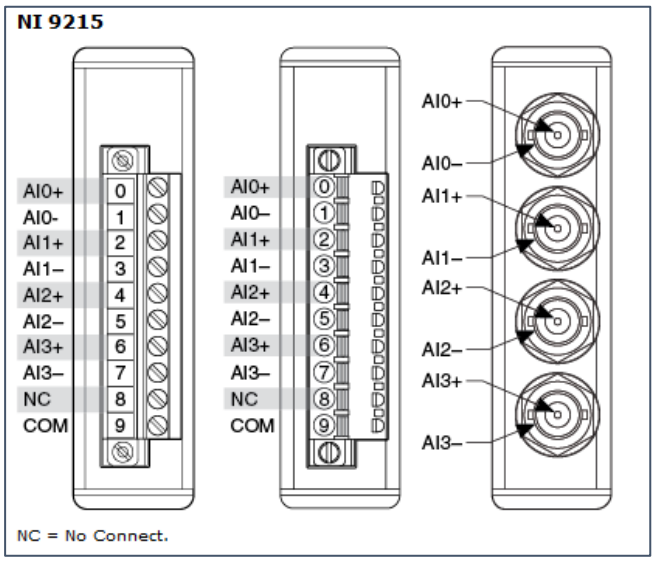


4. In Channel Calibration Wizard, enter channel calibration details.

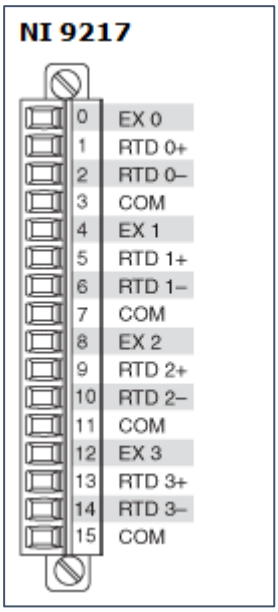
For more details on performing calibration of virtual global channels, refer [Performing DAQmx Channel Calibration in MAX Using Wizard].

1.6 Pinouts of cDAQ Modules

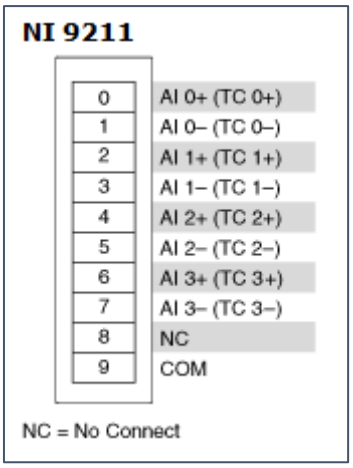
1. Thermistor Analog Input Module (NI-9215)



2. RTD Analog Input Module (NI-9217)

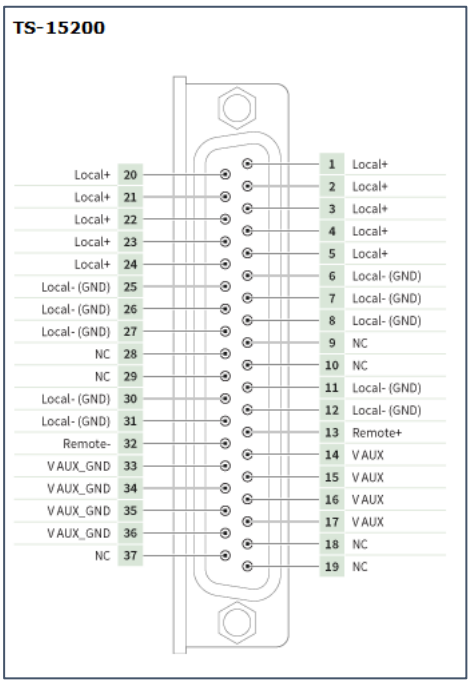


3. Thermocouple Analog Input Module (NI-9211)

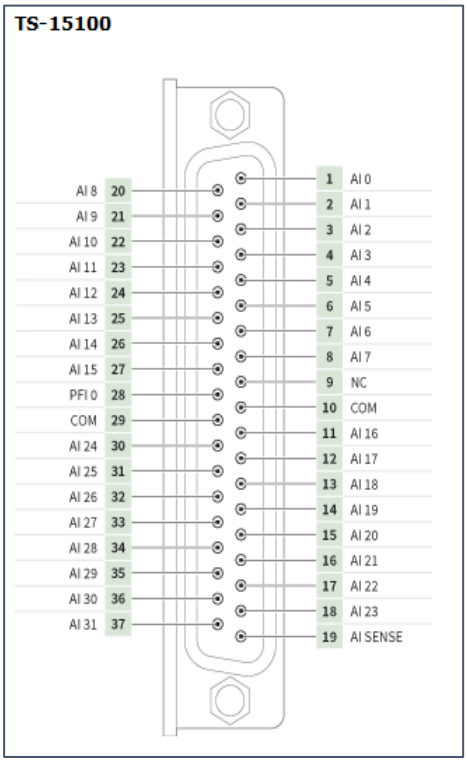


1.7 Pinouts of TestScale Modules

1. **Power Module (TS-15200)**



2. **Analog Input Module (TS-15100)**



1.8 How to create/Modify Global Virtual Channels?

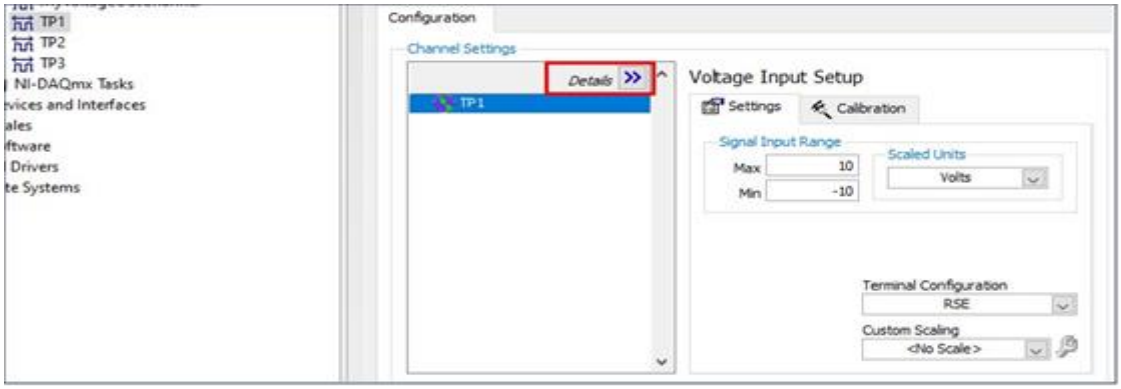
A virtual channel is a collection of settings such as a name, a physical channel, input terminal connections, the type of measurement or generation, and can include scaling information. A virtual channel created outside a task is a Global Virtual Channel.

Follow the below steps to **create Global Virtual Channel** in NI-MAX.

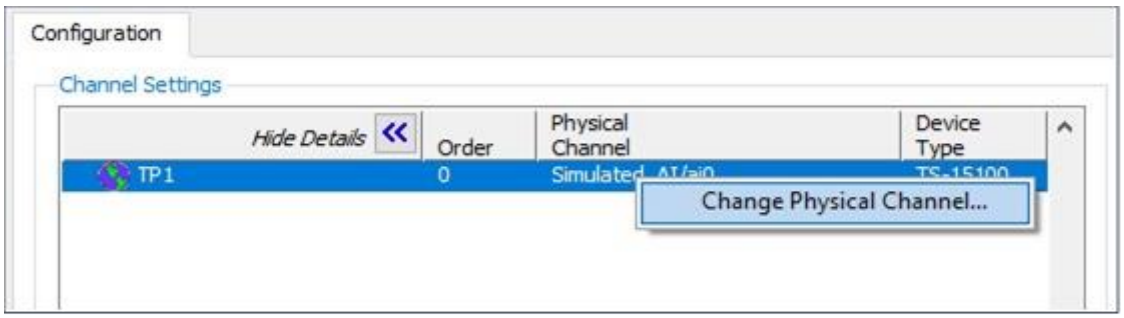
1. Launch NI-MAX
2. In NI-MAX, right-click **Data Neighbourhood** and select **Create New**
3. In the Create New window, select **NI-DAQmx Global Virtual Channel** and click **Next**. The DAQ Assistant opens.
4. Select an I/O type, such as analog input
5. Select the physical channel of Hardware 6
6. Type the global virtual channel name. Click **Finish**
7. Save your configuration.

Follow the below steps to **modify the existing Global Virtual Channel** in NI-MAX.

1. Launch NI-MAX
2. In NI-MAX, expand **Data Neighbourhood** > **NI-DAQmx Global Virtual Channel**
3. Select the Global Channel to modify. Configuration window opens.



4. Click on “Details >>” as highlighted above to view the Physical Channel
5. Right click and **Change Physical Channel** to update the Physical Channel. Select the Physical Channel from Hardware as per the connection and Click “Ok”



6. **Save** your configuration