Action Button Test Sequence

1.1 Purpose

The Action Button Test demonstrates DC-RMS Voltage Measurements by performing button actions (generating DC Voltages) on specific test points on a PCB simultaneously using Analog Output and Analog Input DAQ resources This example sequence can be executed in a custom Python sequence script using the measurement libraries written in Python.

Example File Location

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

1.2 Highlighted Features

- Action Button Test
  - Sources DC voltage with Analog output channels to perform button actions. Measures DC-RMS Voltage Measurements using Analog input channels at specific test points. Libraries used in the example are "DC Voltage Generation" and "DC-RMS Voltage Measurement"
- Turn Off all AO Channels
  - Powers down all analog output channels by configuring the output voltage as 0 Volts. Library used in the example is "DC Voltage Generation".

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

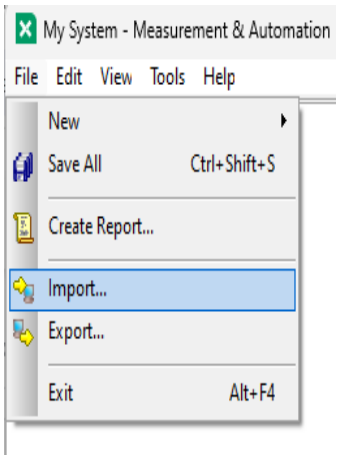
1.4 Setup Diagram

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

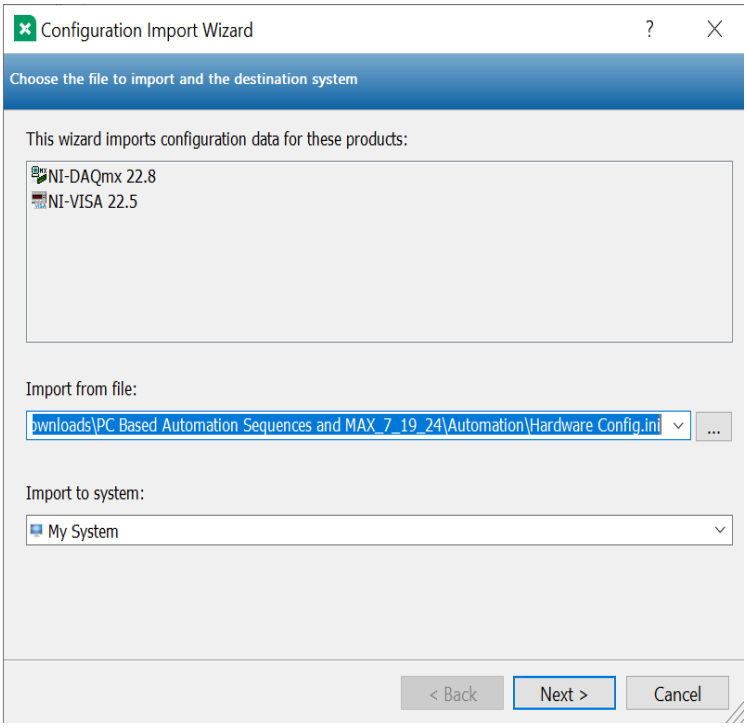
1.5 How to run this Example?

Complete the following steps to run the sequence.

1. First, we must configure NI-MAX to reflect the virtual channels which will be used by the Python script names action_button_main_sequence.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 action_button_test.py file will create a log file in the form of a simple text (.txt) file. The default file path it will use is

```
"C:\\Windows\\Temp\\action_button_test_results.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 *action_button_main_sequence.py* and *action_button_test.py* in your IDE or text editor of choice. The following steps are performed within *action_button_test.py*
    a. **Action Button Test** - demonstrates DC-RMS Voltage Measurements by performing button actions (generating DC Voltages) on specific test points. Below are the steps included in the test.
        i. Initialize the DC Voltage Generation and DC-RMS Voltage Measurement Libraries by creating instances of the ***DCVoltageGeneration*** and ***DcRmsVoltageMeasurement*** classes and then using the ***initialize()*** method on each object. This is completed within the ***setup()*** function defined starting on line 27.
        ii. Configure DC Voltage Generation to generate DC voltage to perform Action Button OFF/ON Conditions. In this example, setting **TS_ButtonEnable0** to 3.3 Volts and 0 Volts are considered as Button ON and OFF conditions respectively.
        iii. Measure DC-RMS Voltage Measurement for the expected Button action states. These steps are completed within the ***main()*** function starting on like 56.
        Refer the comments in the sequence for more details.
    b. **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. These steps are accomplished within ***turn_off_all_ao_channels.py***
        i. Initialize DC Voltage Generation Library by creating a ***DcVoltageGeneration()*** object and calling the ***initialize()*** method on it.
        ii. Configure the DC Voltage Generation to source 0 Volts in specified Analog Output channels by creating an output configuration and using it with the ***configure_and_generate()*** method.
        iii. Close the DAQmx task after setting AO channels to 0 Volts with the ***close()*** method.
3. Run the script *action_button_main_sequence.py* from the command line or your IDE of choice
4. When execution is complete, **review the *action_button_test_results.txt* file located in the C:\\Windows\\Temp*.***
    a. The report contains the measurement values captured by the simulation
    b. Verify the measurement and data returned by the measurement library are the expected values

1.6 How to enable the Hardware?

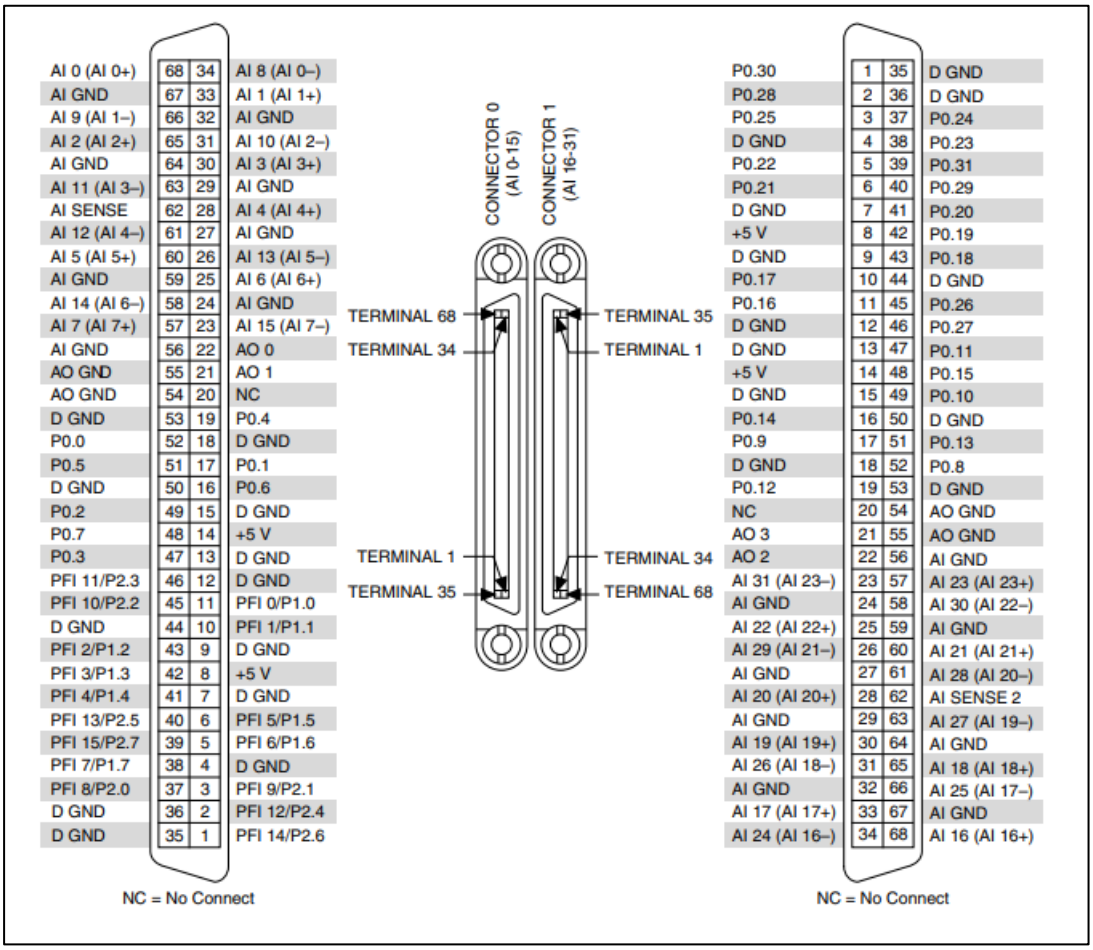
Power Supply test sequence runs with simulated hardware by default. Once the hardware setup is available, you can do the below changes to enable running the test with the hardware.
Note: In this example, physical and global virtual channels are used to configure the terminal or pin to perform the instrument 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 which imports simulated virtual channels in MAX as in section 1.5. If already done, you can simply update the channel names (physical or virtual) in the ***initialize()*** step of each automation sequence to match the hardware connected/detected in NI-MAX.

Note: Please ensure correct trigger sources as mentioned in the steps below.

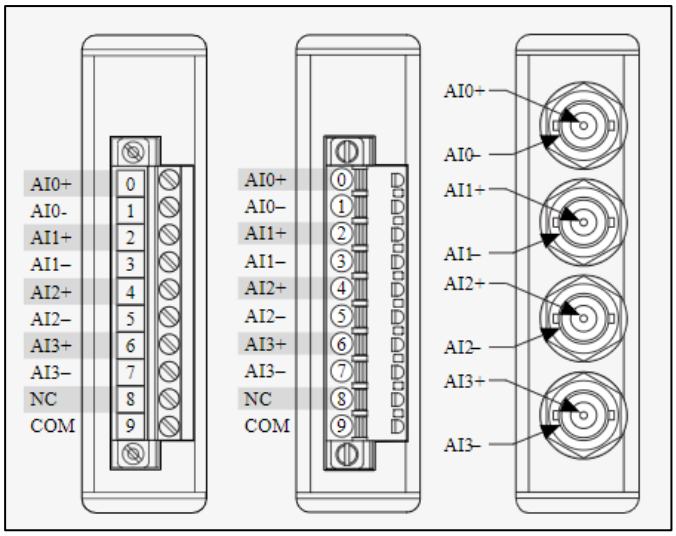
2. Follow the below steps for each script. Refer "**Note to run with Hardware**" labels in the sequence.
    - **i. Action Button Test**
        1. Open *action_button_test.py*
        2. Open NI-MAX and update the physical Channel linked to the Global Channels – **TS_ButtonEnable0, TP_LineOut0, TP_LineOut1** (used in the initialize step of DC Voltage Generation and DC-RMS Voltage Measurement)
           Note: The script should be executed at least once with simulated hardware to create the required Global Virtual Channels in NI-MAX to modify.
        3. Update the generated DC Output Voltages based on Action Button ON/OFF conditions in the "DC Voltage Generation - Action Button ON/OFF State" steps. Note the following considerations,
            a. In this example, supplying 3.3V to the **TS_ButtonEnable0** channel is considered the button action ON State and 0V supply is considered the button action OFF state. Review and modify the voltage based on the use case.
            b. The maximum current drive per channel is usually several mA on Analog Output typical channels. This limit should be considered for button action control.
        4. Review the configurations of Analog Output and Analog Input Pins for the intended use case
    - ii. **Turn Off AO Channel Sequence**
        1. Open the ***turn_off_all_ao_channels.py***
        2. Update the "***ANALOG_OUT_CHANNELS***"string with Analog Output Channel so the correct configuration will be used in the ***initialize( )*** method.
        3. Review the configuration of the Analog Output pins for the intended use case.

1.7 Pinouts of PCIe-6323

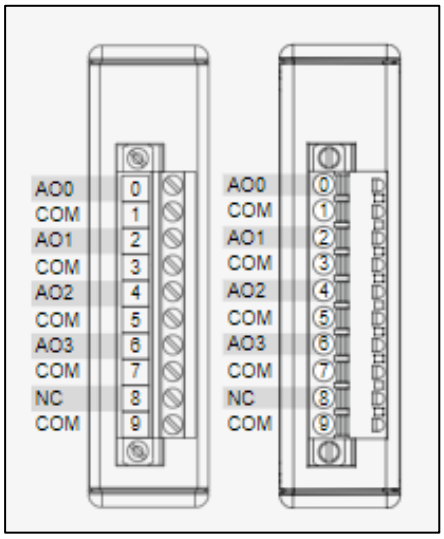


1.8 Pinouts for cDAQ Modules

1. **Analog Input Module (NI-9215)**

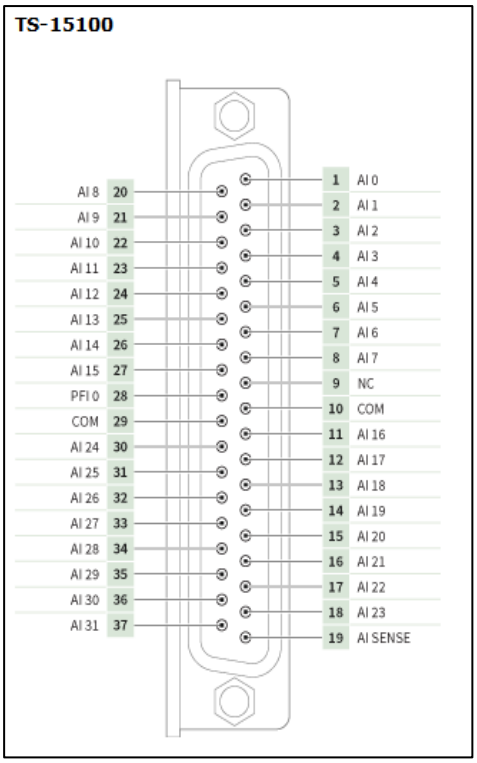


2. **Analog Output Module (NI-9263)**

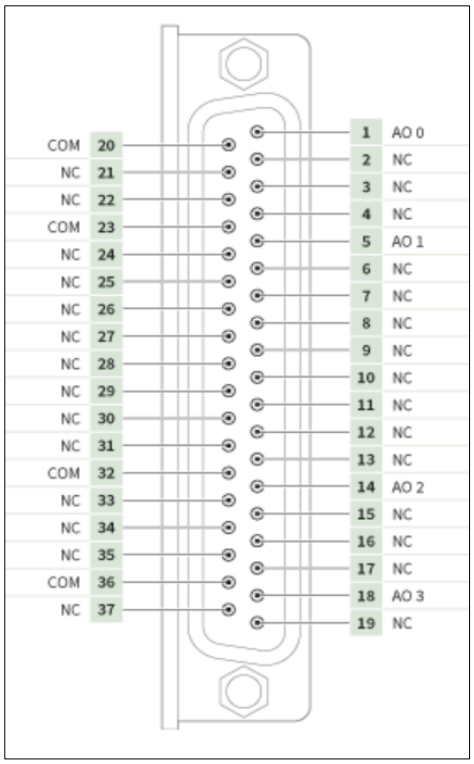


1.9 Pinouts of TestScale Modules

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



2. **Analog Output Module (TS-15110)**



1.10 How to create/Modify Global Virtual Channels?

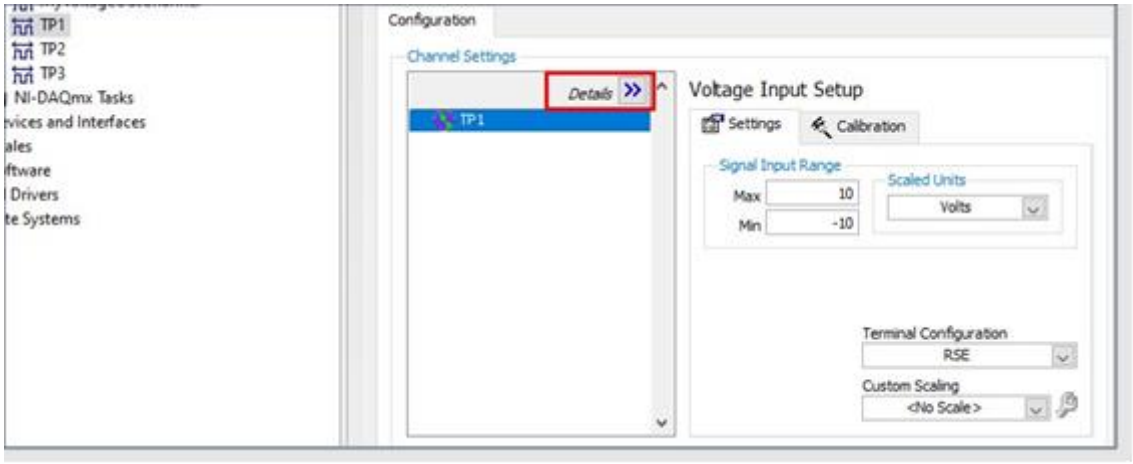
A virtual channel is a collection of settings such as a name, a physical channel, input terminal connections, the type of measurement or generation, and can include scaling information. A virtual channel created outside a task is a Global Virtual Channel. Follow the below steps to **create Global Virtual Channel** in NI-MAX.

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

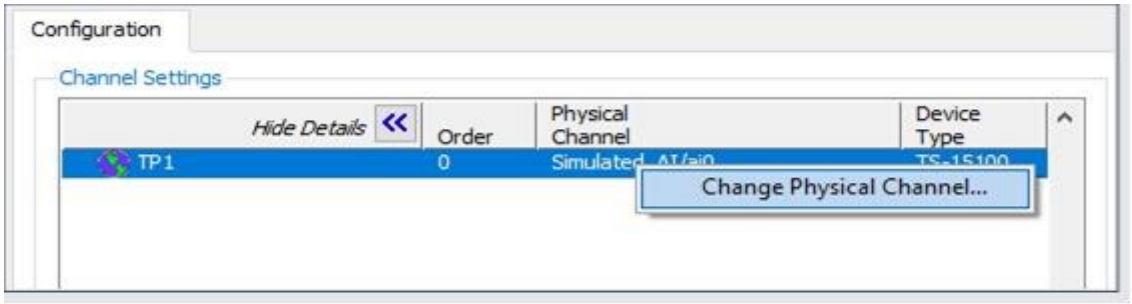
1. Launch NI-MAX
2. In NI-MAX, right-click **Data Neighbourhood** and select **Create New**
3. In the Create New window, select **NI-DAQmx Global Virtual Channel** and click **Next**. The DAQ Assistant opens.
4. Select an I/O type, such as analog input
5. Select the physical channel of Hardware
6. Type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