



connect

2024 AUSTIN



Getting Starting With Mixed Signal Instrumentation and Python

A Crash Course for Beginners

Jacob Sees

Product Manager

Technical Session Agenda

Overview	Python Overview && Integration with NI Hardware
Python Integration with NI Software	LabVIEW, TestStand, and More!
Modular Instrument APIs	Installation and Basics to Getting Started
Demo	Code Walkthrough && Simplified 'CCA' Integration Test
Tips and Tricks	Tips & Tricks for Using the NI Python APIs
Appendix	Resources & Documentation

NI and Python Overview

Defining Python

- Python is an **interpreted, object-oriented, high-level programming language**.
- Python's **high-level built-in data structures**, combined with **dynamic typing** and **dynamic binding**
- Python's **simple, easy-to-learn syntax** emphasizes **readability**.

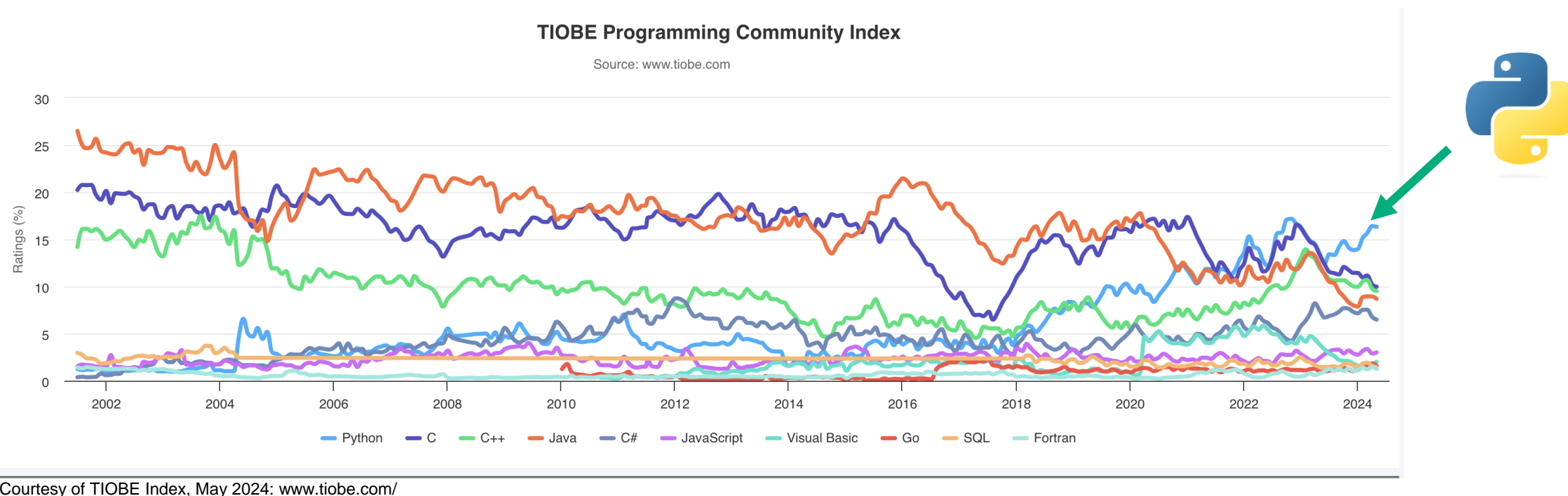
NI Software Promise: A Comprehensive, Connected Approach



JOE & LUIS
LabVIEW Developers,
EI Electronics, Ireland.

1. **Develop more quickly** in an environment tailored to the specific workflow of the test engineer
2. **Spend time where it matters most** with higher level starting points for most measurement tasks
3. **Get unbound flexibility** to meet new and evolving requirements with an ecosystem open to any HW & SW
4. **Share and reuse IP** with a connected suite of software that spans the product development lifecycle
5. **Deliver insight across the organization** with trusted data sharing and visualization.

NI and Python: Python's Rise



Popularity:

- Python is **simple** and had easily **readable syntax**
- Python can support **wide variety of applications**
- **Large and Supportive Community** of Developers

NI and Python: Connect with Ease



Connect with Ease:

- Integration with NI's diverse variety of both hardware and software in NI's portfolio
- Access hardware directly from Python
- Start in Python and integrate into preferred NI software

NI Python Libraries Support Variety of NI Hardware



DAQ Modules



RIO Devices



CAN/LIN/FlexRay Modules



Modular Instruments



Key Advantages of NI Python Drivers

```
#####
Hardware Required: PXIe-5433
#Function: Generate Square Wave to Tr
#Inputs: N/A
#Outputs: SQA to DUT
#####
def _sqaoutput(resource_name, channel

    with nifgen.Session(pcie5433) as
        #Configuration of Fgen Session
        session.output_mode = nifgen.
        session.configure_standard_w

        #Start waveform output
        with session.initiate():
            #print("A4G Test String")

            time.sleep(10) #Run Wavefo

#####
Hardware Required: Scope, passive pr
#Function: Measure Output Voltage from
#Inputs: DUT to DMC
#Outputs: N/A
#####
def _scopeinput(channels):
    with niscopes.Session(pcie5162) as

        session.channels[0].configure

        session.configure_horizonta

        channels = [ch.strip() for ch
        num_channels = len(channels)
        num_records = 5
        total_num_wfs = num_channels

        with session.initiate():
            waveforms = session.chann
            for wfm in waveforms:
                print('Channel (0), recor
```

Setup and install is quick
and easy

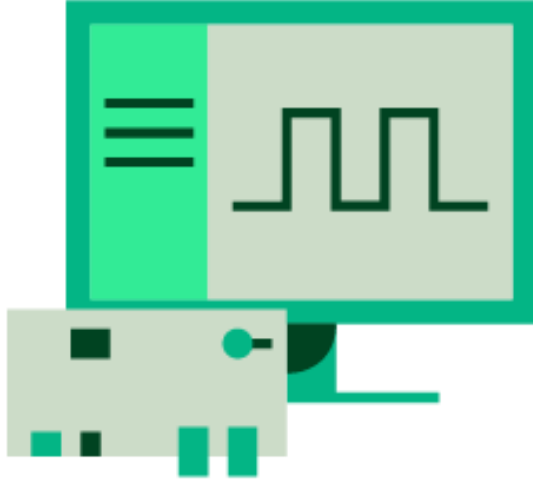
Task creation and usage is
simple

Open Source

Python familiarity makes
picking up the driver quick

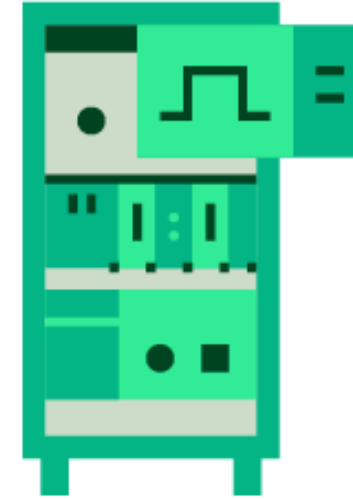
Integration with NI Software

Software Integration



Develop Automated Test Systems

Utilize the Python node in LabVIEW to develop UIs and call Python code



Create Sequences with TestStand

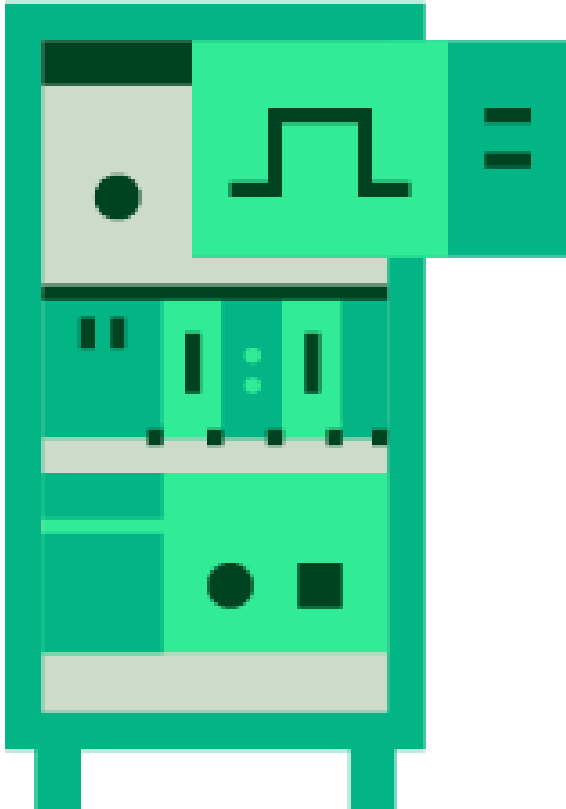
Sequence tests in TestStand with the Python Adapter

A stylized illustration of a computer monitor and a circuit board. The monitor has a green frame and a light gray screen. On the left side of the screen is a green vertical bar with three horizontal black lines. On the right side of the screen is a black square wave. In front of the bottom left of the monitor is a gray circuit board with several colored components: a black square, a green circle, a green square, a black square, and two green rectangles.

The screenshot shows a LabVIEW block diagram with the following elements:

- File Path:** A text box contains the path `C:\Users\NI\Desktop\pyClass.py`.
- Function Name:** A pink box labeled "function name" contains the text `ObjInitialize`.
- Input:** A numeric constant `3.9` is connected to the `ObjInitialize` function.
- Python Object Refnum:** A Python icon block is connected to the `ObjInitialize` function.
- Function Name:** A pink box labeled "function name" contains the text `NumberForIncrement`.
- Input:** A numeric constant `1` is connected to the `NumberForIncrement` function.
- Result:** The output of the `NumberForIncrement` function is connected to a numeric display block labeled "result" showing `DBL`.
- Error Handling:** An "error in (no error)" block is connected to the `ObjInitialize` function. The output of the `ObjInitialize` function is connected to an "error out" block.

Integration with TestStand



TestStand allows you to directly call and integrate with code written in various languages, including Python using the “Call Executable Step”.

```
23 class ClassDemo(object):
24     # Operation Scope: N/A
25     # Operation Type: Create Class Instance
26     # Class Name: ClassDemo
27     # Function Name: N/A
28     def __init__(self, first, last, population):
29         self.first = first
30         self.last = last
31         self.population = 3
32         self.population += 1
33         self.population2 += 1
```

Step Settings for Action

Module: C:\Users\lvadmin\Documents\Test.py

Operation Scope: Class Operation Type: Create Class Instance

Class Name: ClassDemo Function Name: Demo1_method

Class Instance:

NAME	TYPE	LOG	VALUE
Return Value	Object	<input type="checkbox"/>	Locals.ClassInstance fx ✓
first	Dynamic	<input type="checkbox"/>	Locals.String fx ✓
last	Dynamic	<input type="checkbox"/>	Locals.Boolean fx ✓
population	Dynamic	<input type="checkbox"/>	Locals.Number fx ✓

Enhance Even More NI Software with Python



VeriStand

Utilize for custom
real-time testing
and HIL
applications



DIAdem

Automate data
analysis and
reporting with the
Scripting Tool



SystemLink

Extend system
monitoring and
management
capabilities

NI Hardware and Python APIs

NI Modular Instruments



Instrument	PXI Oscilloscopes	PXI Arbitrary Waveform Generators	PXI Programmable Power Supplies and Source Measure Units
Driver	NI-SCOPE	NI-FGEN	NI-DCPower
API	niscope	nifgen	nidcpower

NI Modular Instruments

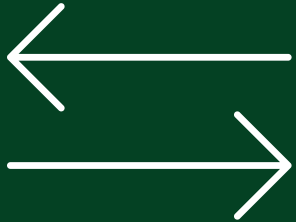


Instrument	PXI Switches	PXI Digital Multimeters	PXI Digital Instruments
Driver	NI-SWITCH	NI-DMM	NI-Digital Pattern Driver
API	niswitch	nidmm	nidigital

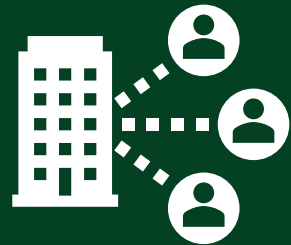
Even More APIs for Python



NI-TCIk

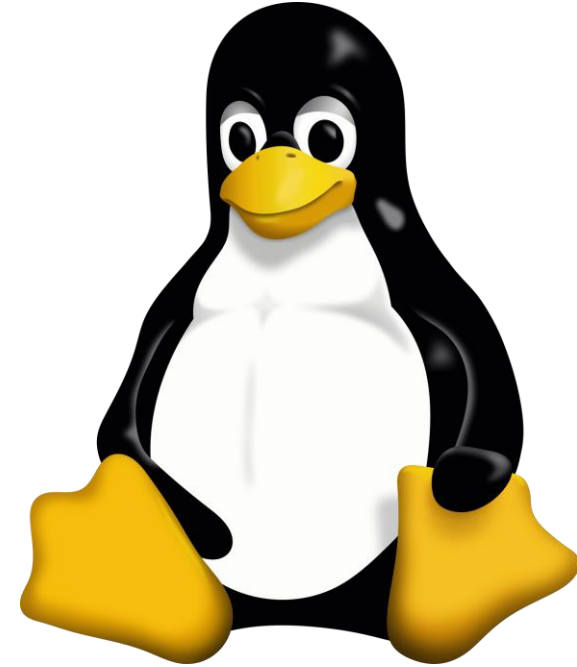


Switch Executive



NI gRPC

Installation of NI Python Modules



Python APIs can run on Windows or Linux!

Installation of NI Python Modules

```
(base) jsees@jacobs-mbp ~ % python -m pip install nidcpower
Collecting nidcpower
  Obtaining dependency information for nidcpower from https://files.pythonhoste
4.8-py3-none-any.whl.metadata
```

TO INSTALL:

“python -m pip install (modular instrument python library)”

Installation of NI Python Modules

```
(base) jsees@jacobs-mbp ~ % python -m pip install nidcpower
Collecting nidcpower
  Obtaining dependency information for nidcpower from https://files.pythonhoste
4.8-py3-none-any.whl.metadata
```

NOTE: Must have corresponding drivers installed on your system as a prerequisite

- Visit ni.com/downloads to download the driver runtime for your device

Getting Started: Read the Docs Site

- ReadTheDocs.io provides in-depth information on:
 - Installation
 - Usage
 - Examples
 - gRPC Support

Documentation

- [niscopes module](#)
 - [Installation](#)
 - [Usage](#)
 - [API Reference](#)
 - [Session](#)
 - [Methods](#)
 - [Properties](#)
 - [NI-TCI Support](#)
 - [Repeated Capabilities](#)
 - [Enums](#)
 - [Exceptions and Warnings](#)
 - [Examples](#)
 - [gRPC Support](#)

<https://nimi-python.readthedocs.io/en/master/>

Getting Started: Sessions

- **Creates** a new IVI instrument driver, with optional initial settings
- **Opens** a session to the designated device using the specified interface and address (resourceName).

Session

```
class niscopes.Session(self, resource_name, id_query=False, reset_device=False, options={}, *,  
grpc_options=None)
```

Performs the following initialization actions:

- Creates a new IVI instrument driver and optionally sets the initial state of the following session properties: Range Check, Cache, Simulate, Record Value Coercions
- Opens a session to the specified device using the interface and address you specify for the **resourceName**
- Resets the digitizer to a known state if **resetDevice** is set to True
- Queries the instrument ID and verifies that it is valid for this instrument driver if the **IDQuery** is set to True
- Returns an instrument handle that you use to identify the instrument in all subsequent instrument driver method calls

Getting Started: Resource Names

The screenshot displays the NI MAX interface. On the left, a tree view under 'My System' shows the hierarchy: 'Data Neighborhood' > 'Devices and Interfaces' > 'NI PXIe-1082 "PXIChassis1"'. A green box highlights the list of slots under 'NI PXIe-1082 "PXIChassis1"', which includes:

- 1: NI PXIe-8880 "DESKTOP-OAIHREL"
- 1: Built-in GPIB "GPIB0"
- 2: NI PXIe-5162 (4CH) "PXI1Slot2"
- 3: NI PXIe-2544 "PXI1Slot3"
- 4: NI PXIe-5172 (8CH - 410T) "PXI1Slot4"
- 5: NI PXI-5142 "PXI1Slot5"
- 6: NI PXIe-4081 "PXI1Slot6"
- 7: NI PXI-4130 "PXI1Slot7"
- 8: NI PXIe-5433 (2CH) "PXI1Slot8"

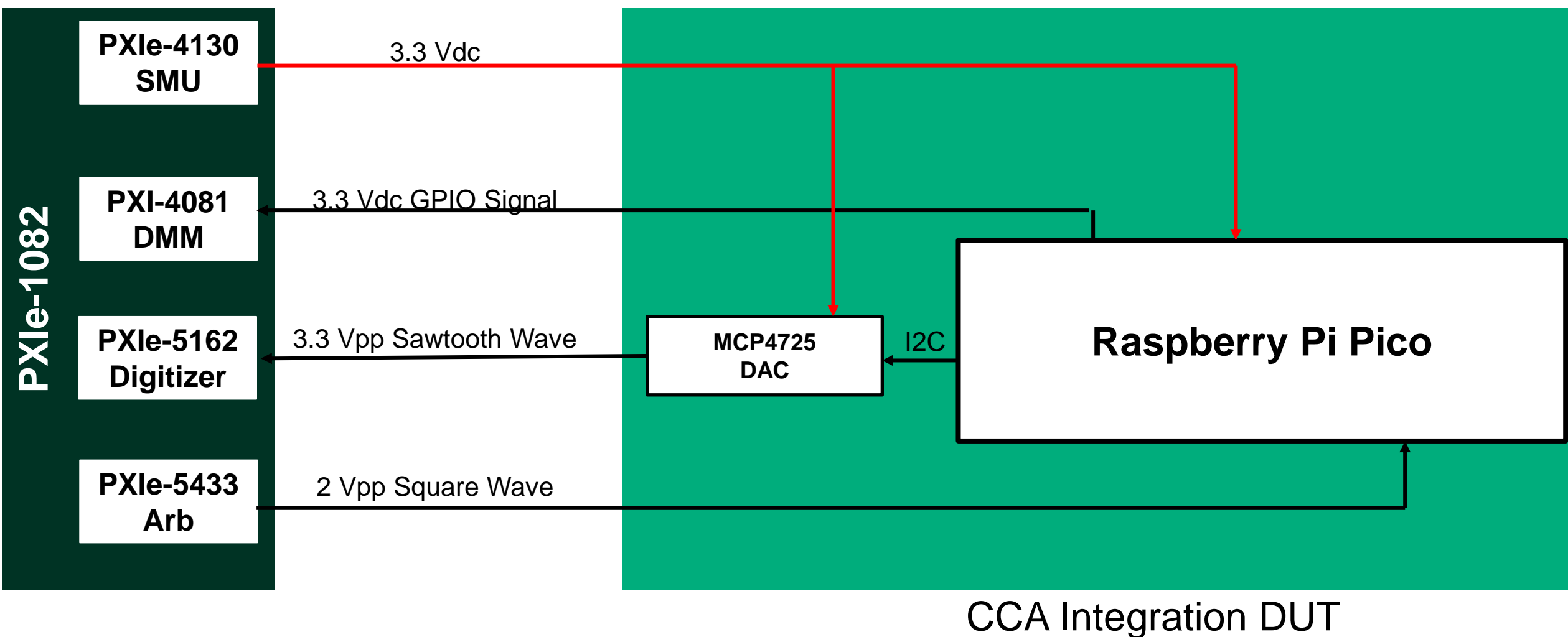
Below this list are other devices like 'NI PXIe-5442 "Arb"', 'NI PXIe-5442 "Dev1"', and 'NI PXIe-5442 "Dev2"', followed by 'Network Devices' and 'NI Switch Executive Virtual Devices'. The right pane shows 'System Settings' for the selected device, with fields for Hostname, DNS Name, Vendor, Model, Serial Number, Firmware Version, Hardware Revision, Operating System, Slot Number, System Start Time, Description, and System Configuration Web Access. A green arrow points from the highlighted list of resource names to a dark green button at the bottom right.

Setting	Value
Hostname	DESKTOP-OAIHREL
DNS Name	DESKTOP-OAIHREL
Vendor	National Instruments
Model	NI PXIe-8880
Serial Number	031062D0
Firmware Version	2.0.1f0
Hardware Revision	D
Operating System	Microsoft Windows 10 Enterprise
Slot Number	1
System Start Time	5/17/2024 1:35 PM
Description	
System Configuration Web Access	Local Only

Check Resource Names in MAX

DUT Block Diagram

DUT Block Diagram

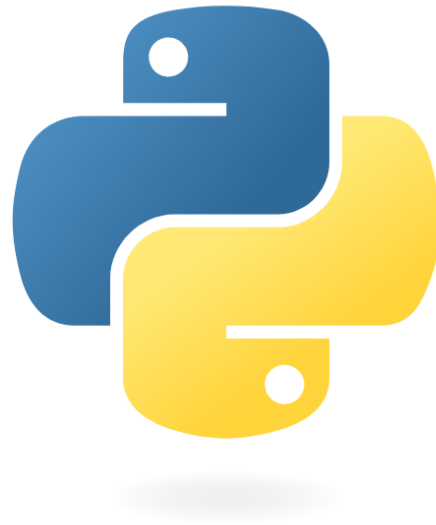


Code & Demo

Tips and Tricks for using the NI Python APIs

Tips and Tricks: NI Python Style Guide

Improve the Readability of Your Code



- Useful if company does not already have internal style guide
- Considers PEP 8 Style Conventions & PEP 257 Docstring Conventions
- Demonstrates best practices

Tips and Tricks: Modularize Your Code

Modularize Your Code:

- **Avoids Unwanted Side Effects:** When importing a module, only the necessary code executes, preventing unintended side effects.
- **Promotes Reusability:** Functions can be reused across different parts of your application

```
def acquire_data():  
    # Code to acquire data from NI hardware  
    pass  
  
def analyze_data(data):  
    # Code to process and analyze acquired data  
    pass  
  
if __name__ == "__main__":  
    acquired_data = acquire_data()  
    analyze_data(acquired_data)
```

Tips and Tricks: Execution Control

```
if __name__ == "__main__":  
    acquired_data = acquire_data()  
    analyze_data(acquired_data)
```

Use `__name__` to Control Execution:

- The `__name__` attribute allows you to determine whether a Python script is being run as the main program or imported as a module.
- Create a `main()` function to contain the code you want to execute when the script is run directly.

Q&A

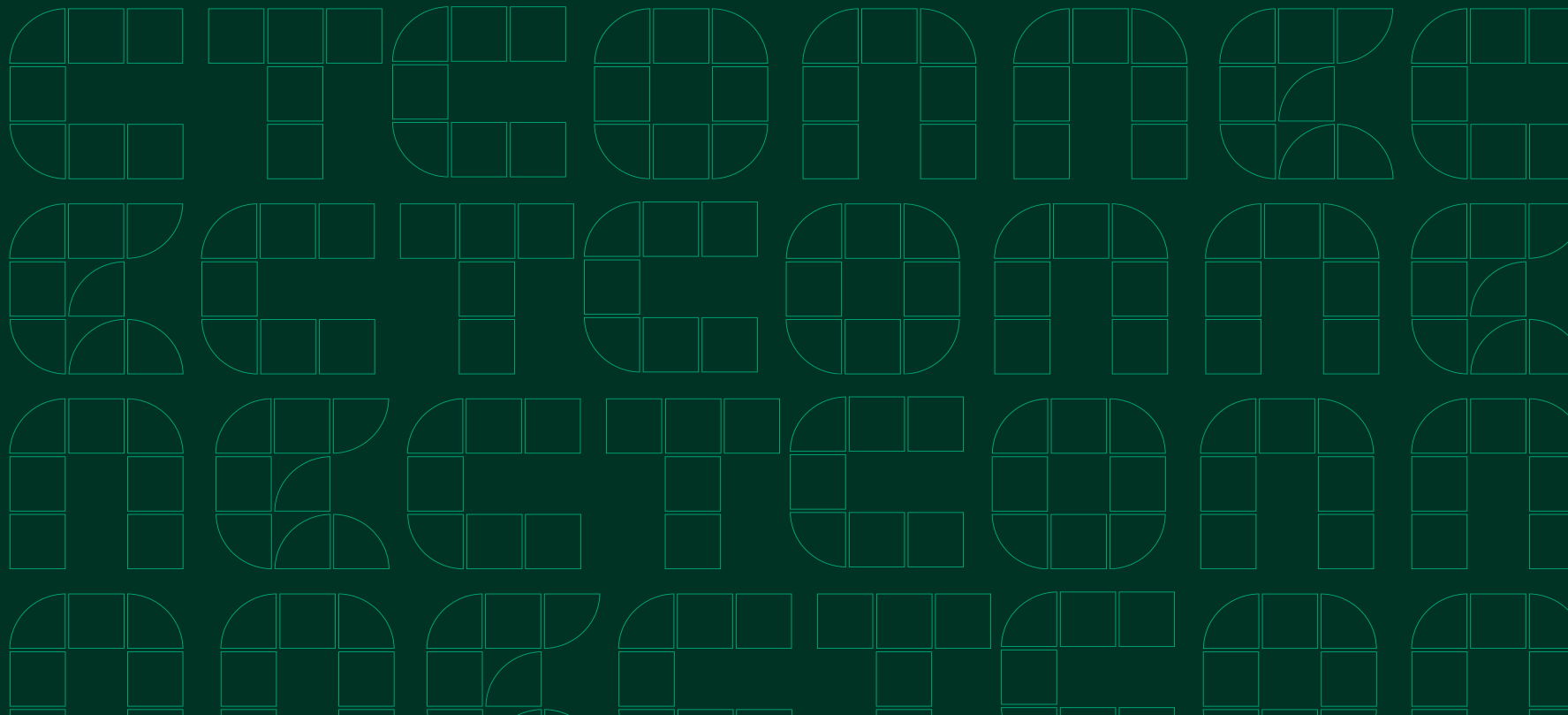
Marcos Kirsch

Chief Software Engineer

Jacob Sees

Product Manager

connect



Appendix

Collected Python Resources for NI
Hardware and Software

More Resources on NI Modular Instruments & Python

NI Website

- <https://www.ni.com/en/support/documentation/supplemental/16/python-resources-for-ni-hardware-and-software.html>

Read the Docs Webpage for Modular Instruments

- <https://nimi-python.readthedocs.io/en/master/>

Expo Floor to LabView / TestStand Demo

- Hands-on experiences!

NI-Python-StyleGuide

- <https://github.com/ni/python-styleguide?tab=readme-ov-file>