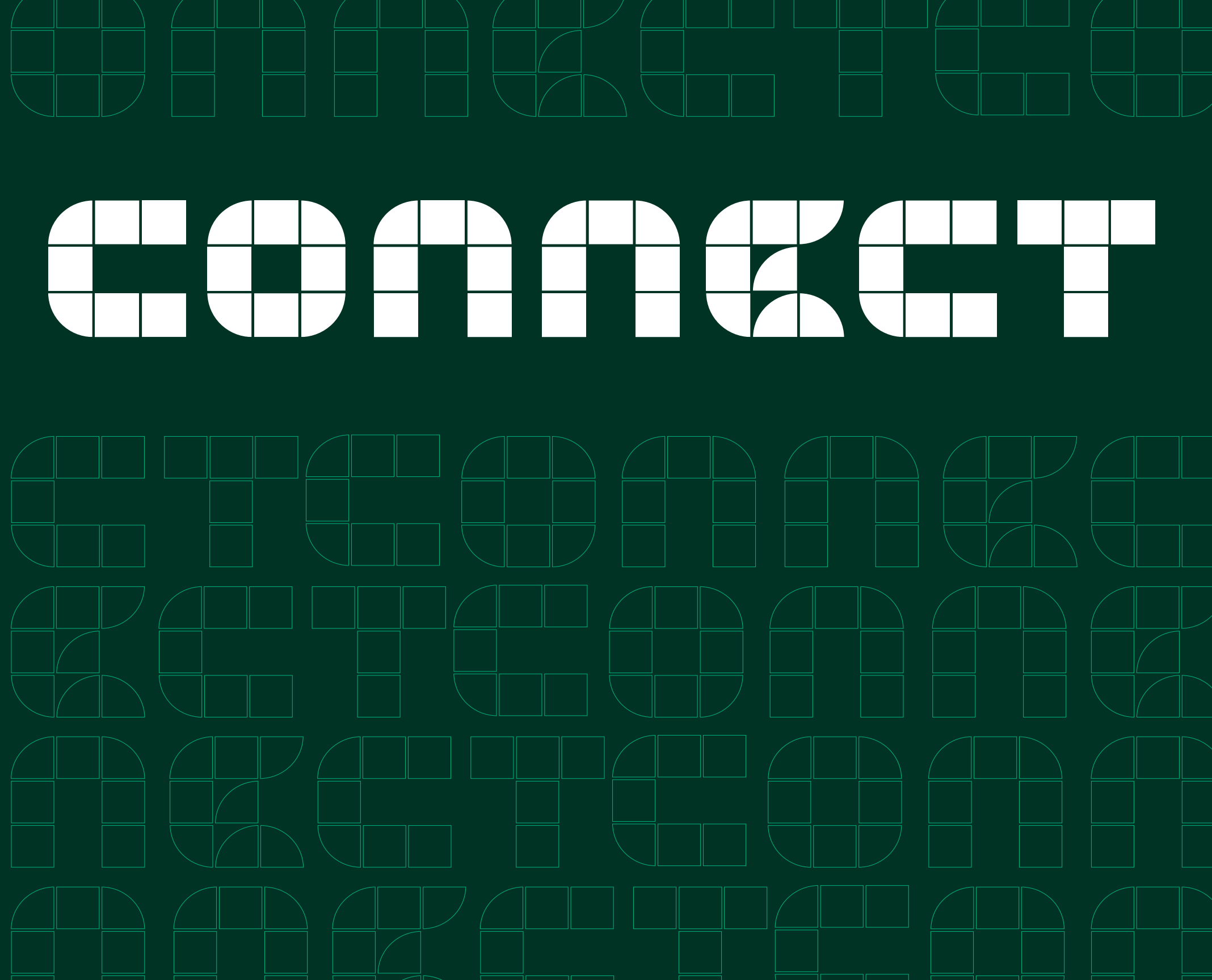
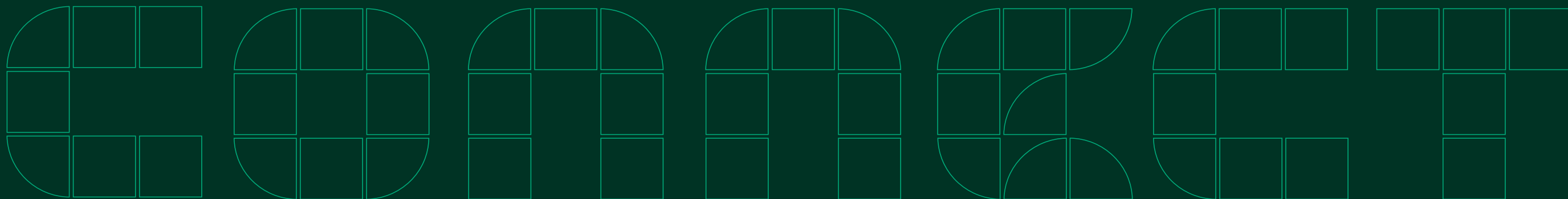


connect





Introduction to Hardware in the Loop using NI Software

Austin Hill
Chief Product Manager

Session Agenda

What is HIL?

NI Software Strategy

Phases of HIL

- IP Integration and Bring Up: [LabVIEW + VeriStand + Simulink](#)
- Manual Validation: [VeriStand](#)
- High Performance Model Integration: [LabVIEW FPGA](#)
- Automate Test Cases: [TestStand](#)
- Create a CI/CD Workflow: [SystemLink](#) or [Python](#)

What is Hardware in the Loop Testing (HIL)?

Testing embedded software deployed onto a Unit Under Test to uncover software defects in conjunction with the hardware, like timing or fault reactions.

Software is the
Device Under Test

```
1  D:\BASYS_PROJECTS\USED_CONTROL\TOGGLE_VHDL.vhd
2  -- VHDL code created by Xilinx's StateCAD 9.2i
3  -- Wed Oct 10 13:13:12 2007
4
5  -- This VHDL code (for use with Xilinx XST) was generated using:
6  -- binary encoded state assignment with structured code format.
7  -- Minimization is enabled, implied else is enabled.
8  -- and outputs are speed optimized.
9
10 LIBRARY ieee;
11 USE ieee.std_logic_1164.all;
12
13 ENTITY SHELL_TOGGLE_VHDL IS
14   PORT (CLK : IN std_logic;
15         Q0 : OUT std_logic);
16 END;
17
18 ARCHITECTURE BEHAVIOR OF SHELL_TOGGLE_VHDL IS
19   SIGNAL next : std_logic_vector (2 DOWNTO 0);
20   SIGNAL next_reg : std_logic_vector (2 DOWNTO 0);
21   CONSTANT STATES : std_logic_vector (2 DOWNTO 0) := "000";
22   CONSTANT STATES1 : std_logic_vector (2 DOWNTO 0) := "001";
23   CONSTANT STATES2 : std_logic_vector (2 DOWNTO 0) := "010";
24   CONSTANT STATES3 : std_logic_vector (2 DOWNTO 0) := "011";
25   CONSTANT STATES4 : std_logic_vector (2 DOWNTO 0) := "100";
26   CONSTANT STATES5 : std_logic_vector (2 DOWNTO 0) := "101";
27
28   SIGNAL next_shreg0, next_shreg1, next_shreg2, next_shreg3 : std_logic;
29   SIGNAL Q : std_logic_vector (2 DOWNTO 0);
30   SIGNAL shreg : std_logic_vector (3 DOWNTO 0);
31
32   SIGNAL Res_Set, Shreg0, Shreg1, Shreg2, Shreg3 : std_logic;
33 BEGIN
34   PROCESS (CLK, next_shreg0, next_shreg1, next_shreg2, next_shreg3)
35   BEGIN
36     IF CLK = '1' AND CLK'event THEN
37       Shreg0 <= next_shreg0;
```



Software is deployed
onto a prototype UUT



Real-time models run on the
test system and simulate the
environment the controller
would experience in the field



Test system tricks the control
board into thinking it's deployed
in the real world while also
creating test scenarios

Embedded Software Test

Engineer Profile

Responsible for testing deployed software to **uncover software defects** over a wide range of parameter variations. Challenged by **test case complexity, disparate IP integration** and **diversity of I/O**.

Task Workflow

Configure & Map

Configure hardware → Import & configure models → Map model to DUT signals.

Test & Bring Up

Interactively deploy → Verify test functionality → Bring up DUT and validate manually.

Automate & Execute

Create library of tests → Build Sequences → Integrate test system into larger ecosystem → Monitor and debug.

Application Profile

Verify software functionality in a range of conditions not easily created or replicated in the real world. NI recommends **Real-Time PXI & DAQ** hardware with **VeriStand** and the **LabVIEW+** Suite of software.

Existing Strengths of NI Platform

- Flexible and scalable software
- Configurable framework for I/O, alarming, fault insertion, UI creation, and bus communication
- Extensibility to 3rd party models
- Adaptability to changing requirements

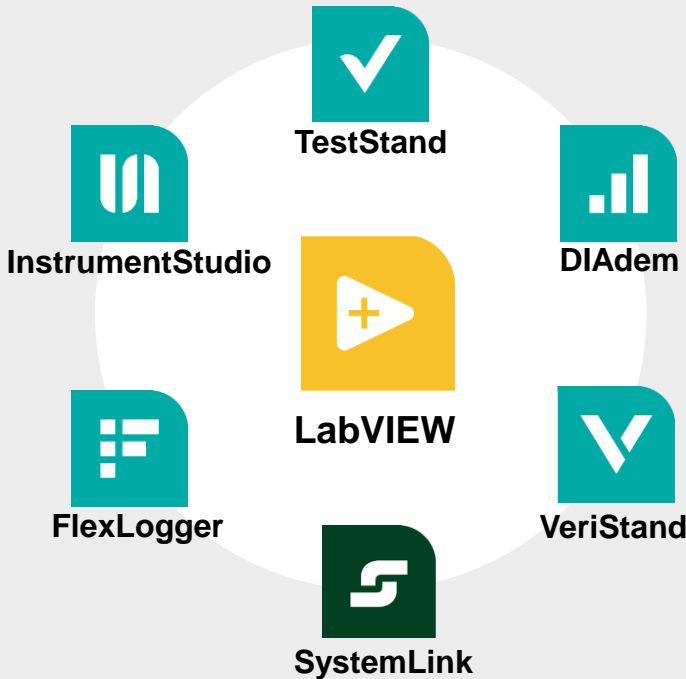
Gaps to be Addressed with Roadmap

- Model integration and reuse
- Automation and sequencing
- System debugging
- Security



Evolving NI Test Software

Enable Automated Test & Measurement Professionals



1

Strengthen LabVIEW

Deliver new capabilities in **LabVIEW & NI Software** to meet the evolving requirements of applications and users

2

Connect LabVIEW+

Bridge seamlessly between tools, tasks and teams to accelerate the productivity of test professionals

3

Build Community

Engage and collaborate with the community to empower their continued success



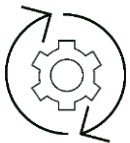
Validates hardware and performs embedded software test for HIL with model integration, real-time stimulus generation, and an extensible software environment.



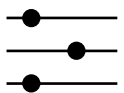
Ready to Run Software for HIL
Create, deploy, and leverage closed loop control on Linux RT for hardware-in-the-loop systems



Quickly Configure Systems
Configure I/O channels, alarming, data logging, stimulus generation, fault insertion, user interfaces and bus communications



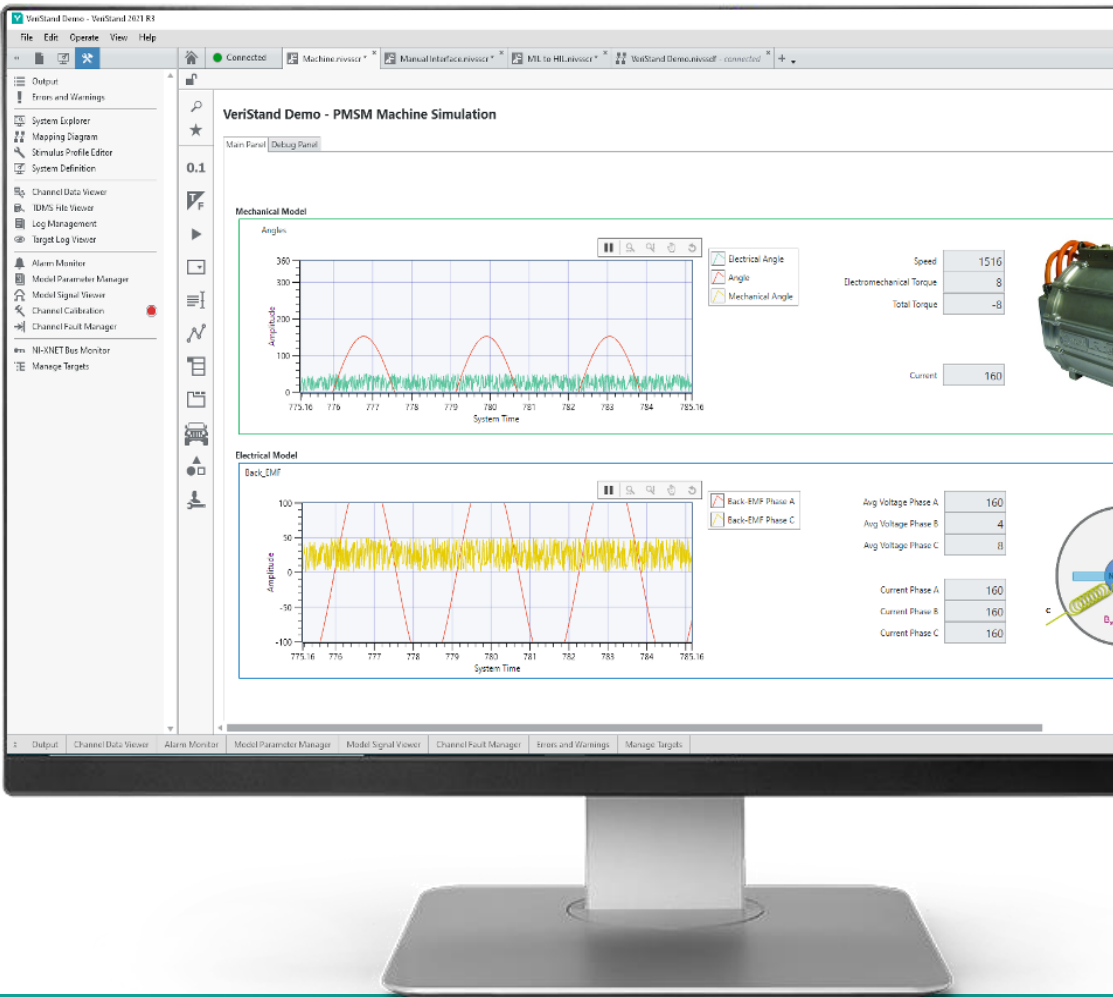
Model Integration
Integrate plant and controller models from Simulink and other 3rd party modeling environments adhering to the FMI standard



Customizability
Adapt to changing requirements with an extensible software environment.
Build plug-ins for 3rd party hardware and additional measurement types



Test Automation
Use test sequences and scripting to configure, orchestrate systems through TestStand, .NET APIs, Python, and ASAM XIL and integrate with CI/CD workflows



Connect VeriStand



With **LabVIEW** to develop plugins to extend your test system with additional functionality



With **TestStand** to build libraries of test steps for higher level orchestration



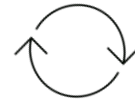
With **your test infrastructure and IP** to maximize reuse

2	0
2	4

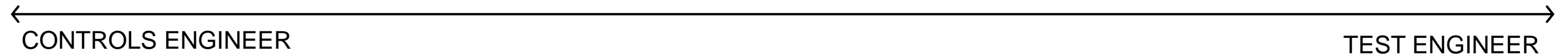
turn on

IP Integration and Bring Up

Model Based Development



MODELS

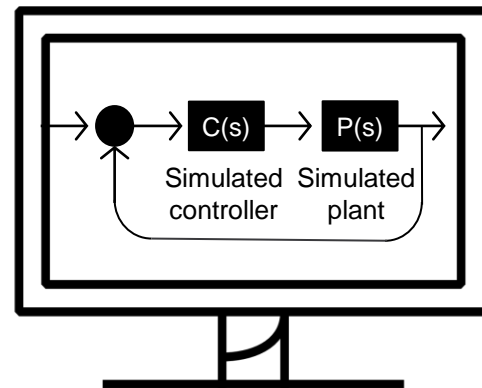


Model in
the Loop

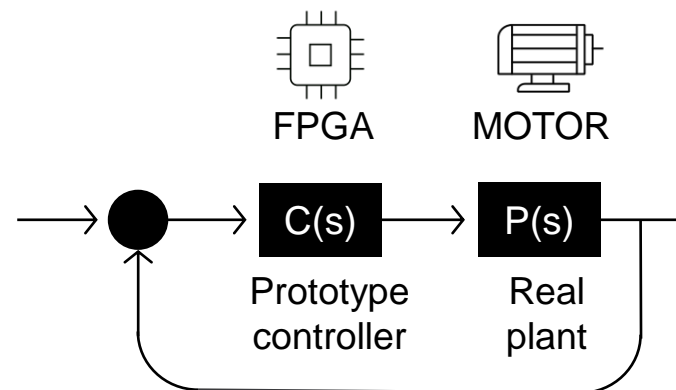
Software in
the Loop

Signal Level
Hardware in the Loop (HIL)

Design

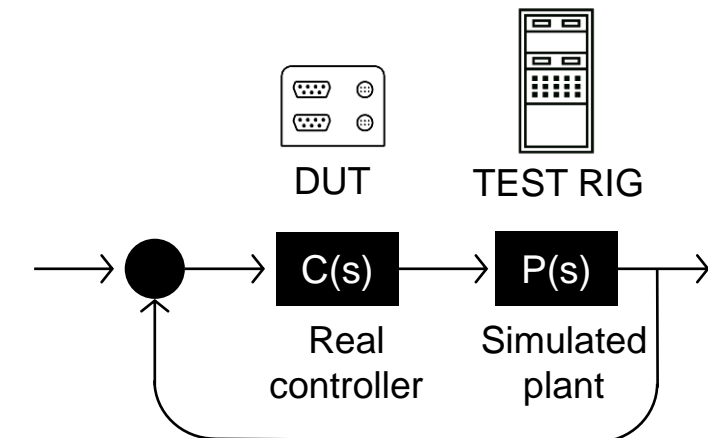


Prototype

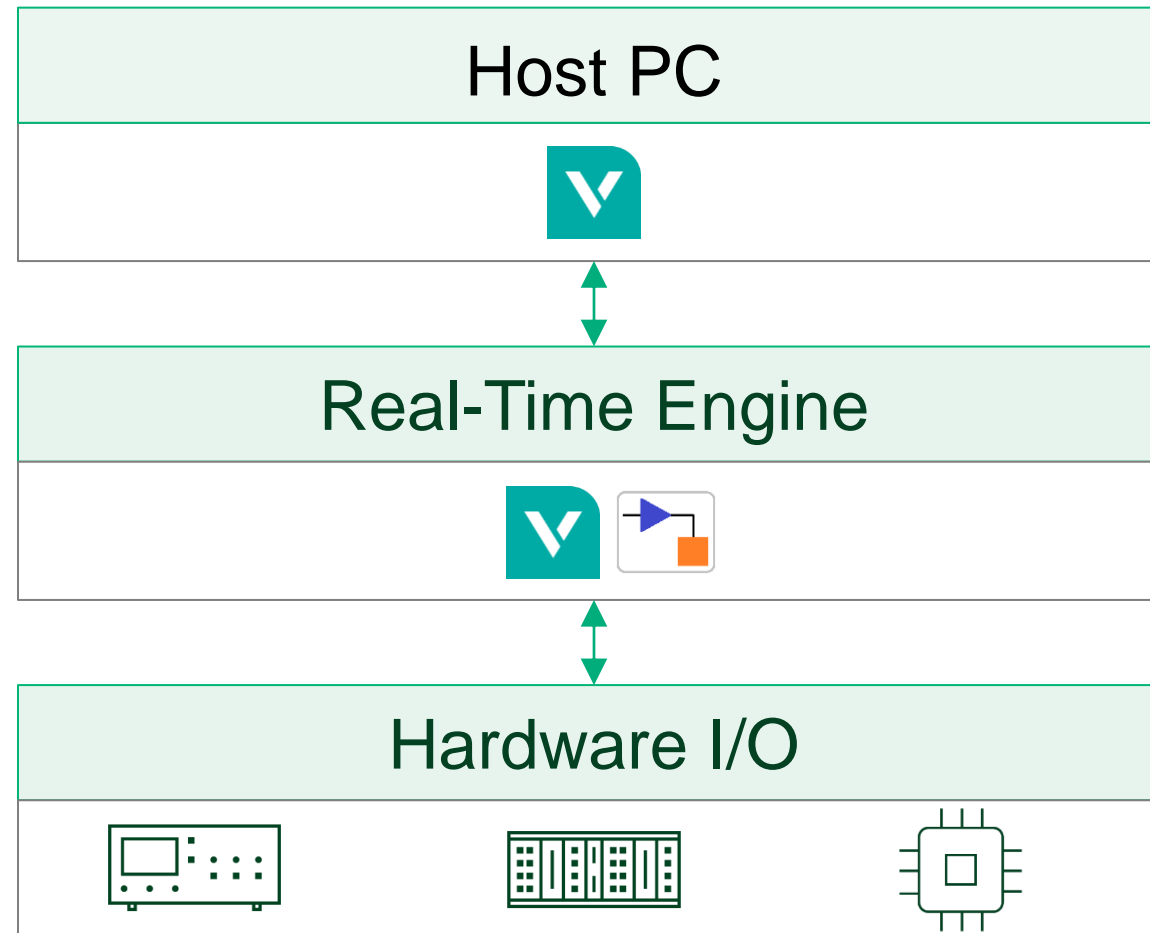


Software and Controller Test

HIL, Functional Test, etc.



HIL Automation Framework



Real-Time Modeling Environment Support

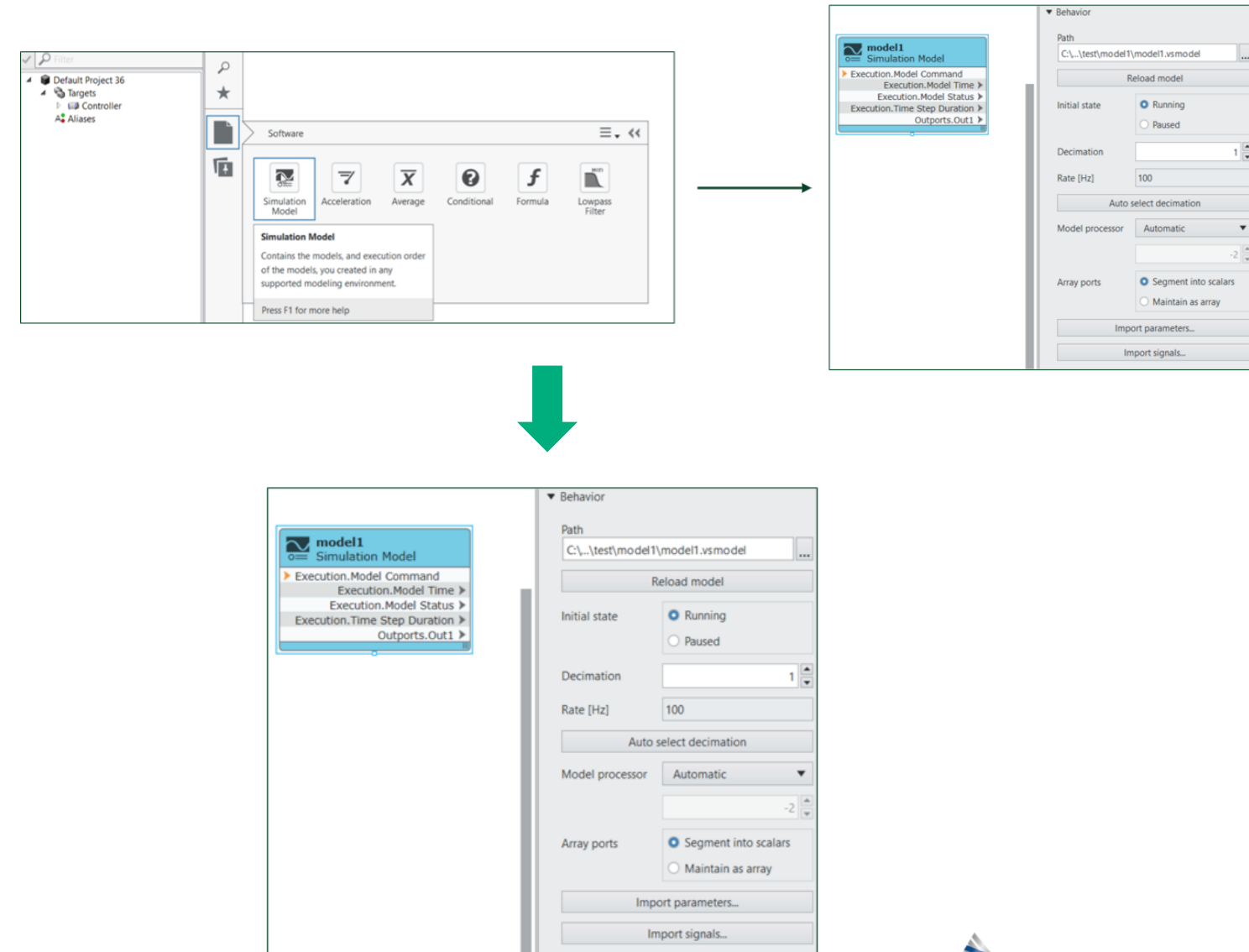
MathWorks Simulink® Software

LabVIEW

C/C++

FMI 2.0/3.0 CoSimulation Support:

- AVL Boost
- FMU SDK
- Wolfram SystemModeler
- MapleSim
- Altair Activate
- And Can Support More [Here](#)

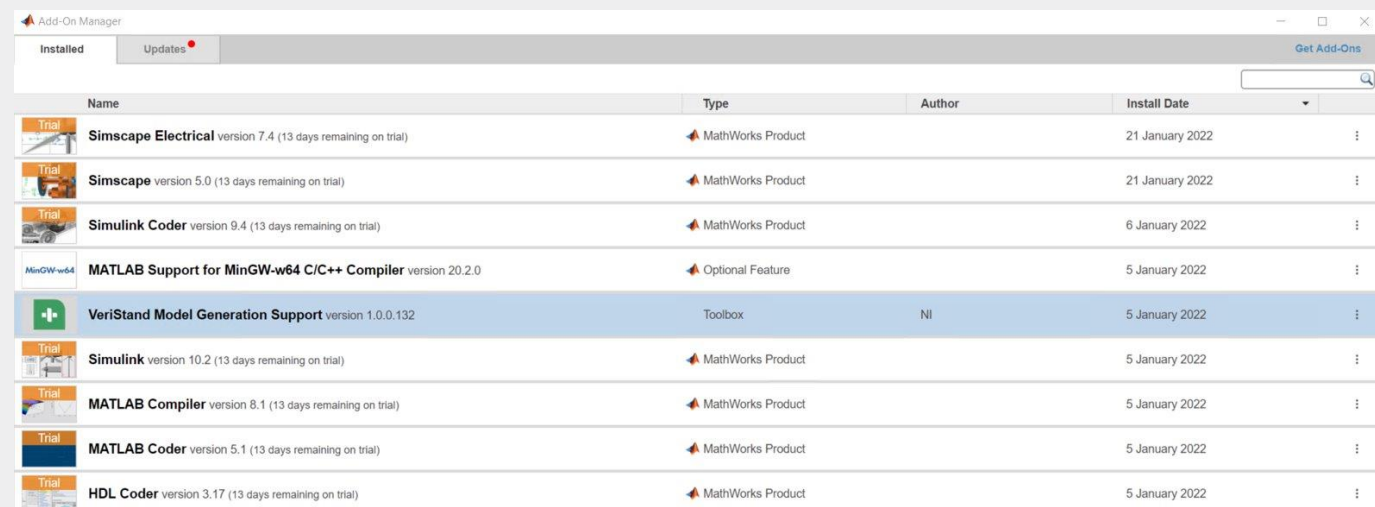


VeriStand Model Generation Support










New Simulink toolbox for CPU model integration

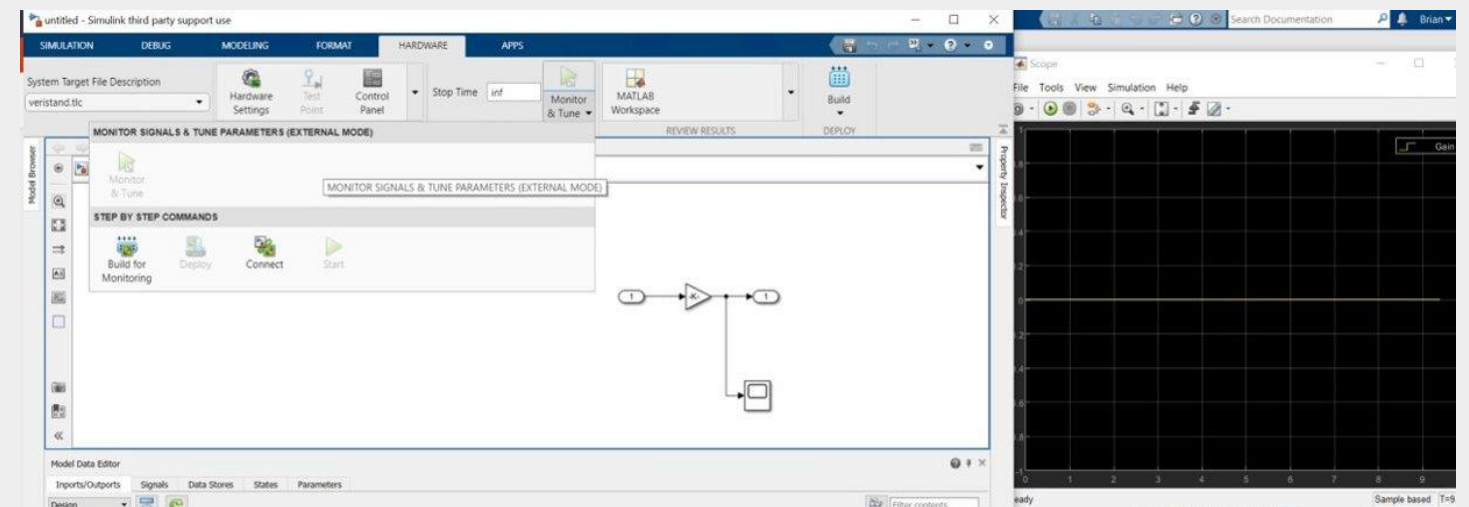
VeriStand 2021 supports new model type: .vsmodels

- Addresses difficulty of getting started, compilation, supporting toolkits, and supporting new versions
- Subsequent releases through 2022 and 2023 expand support, including:
 - External Mode: from Simulink, visualize and debug execution of model executing in VeriStand
 - Import Simulink signals as channels into VeriStand

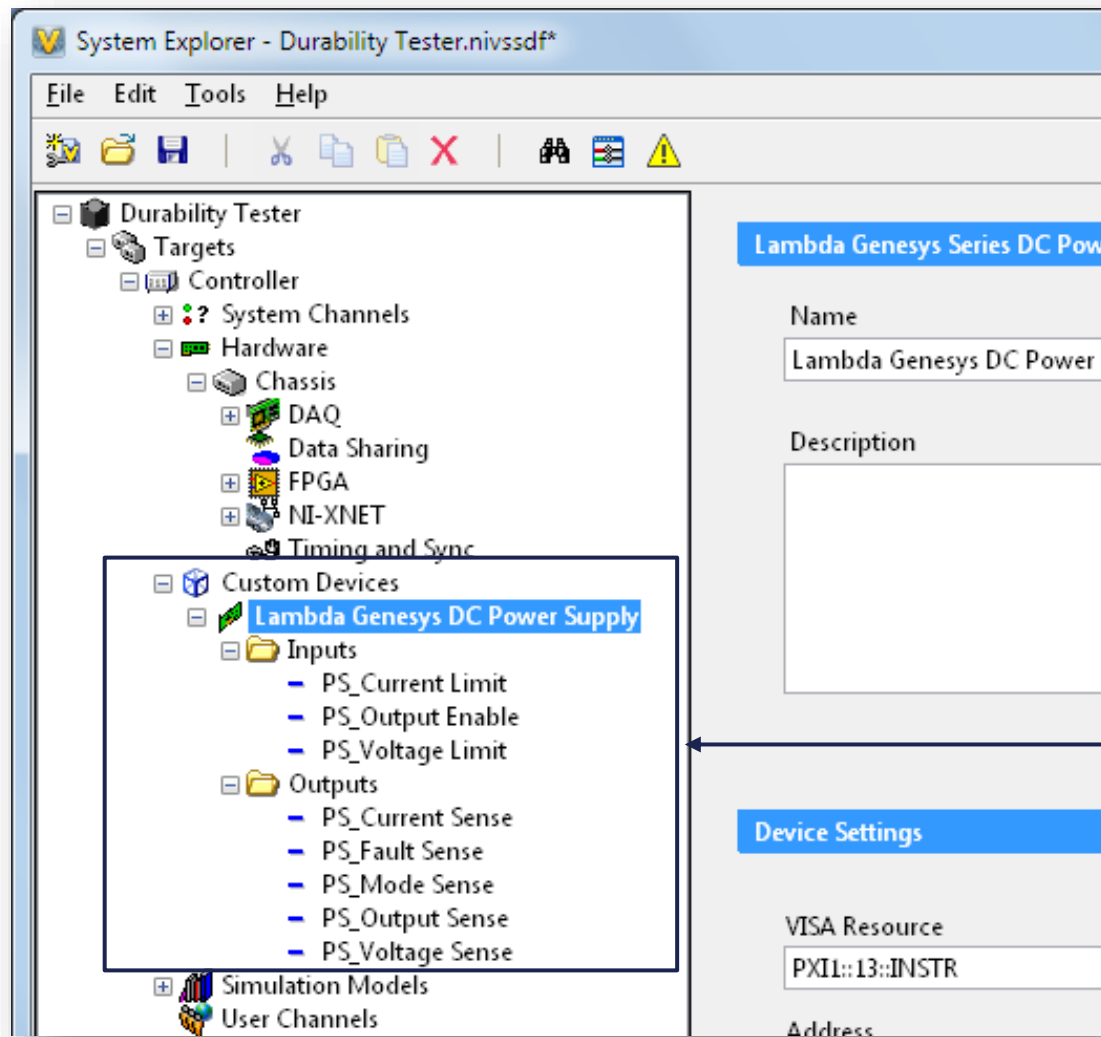


The screenshot shows the 'Add-On Manager' window with the 'Installed' tab selected. It displays a list of installed toolboxes and their details.

Name	Type	Author	Install Date
 Simscape Electrical version 7.4 (13 days remaining on trial)	MathWorks Product		21 January 2022
 Simscape version 5.0 (13 days remaining on trial)	MathWorks Product		21 January 2022
 Simulink Coder version 9.4 (13 days remaining on trial)	MathWorks Product		6 January 2022
 MATLAB Support for MinGW-w64 C/C++ Compiler version 20.2.0	Optional Feature		5 January 2022
 VeriStand Model Generation Support version 1.0.0.132	Toolbox	NI	5 January 2022
 Simulink version 10.2 (13 days remaining on trial)	MathWorks Product		5 January 2022
 MATLAB Compiler version 8.1 (13 days remaining on trial)	MathWorks Product		5 January 2022
 MATLAB Coder version 5.1 (13 days remaining on trial)	MathWorks Product		5 January 2022
 HDL Coder version 3.17 (13 days remaining on trial)	MathWorks Product		5 January 2022



Extending VeriStand with Custom Devices

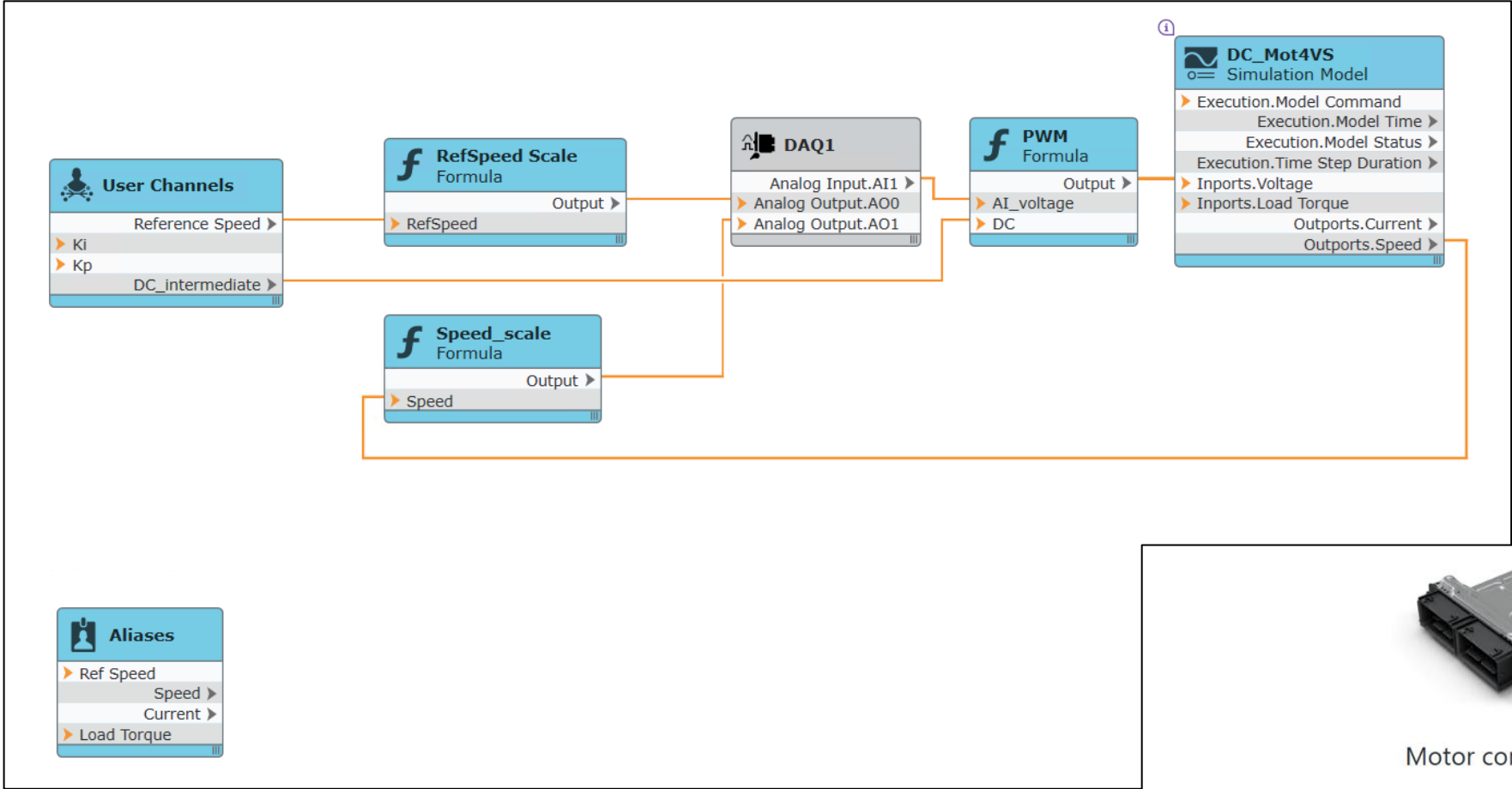


As part of the integration and bring up process, Custom Devices can be built to extend VeriStand to interface with units under test.

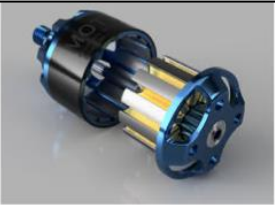
Use VeriStand real-time plugins to add custom functionality to VeriStand applications such as:

- Instrument interfaces
- Third-party hardware support
- User-defined tasks

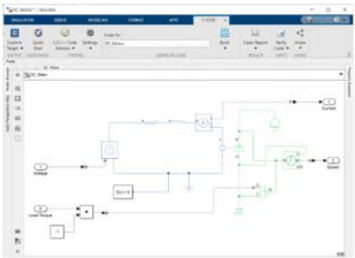
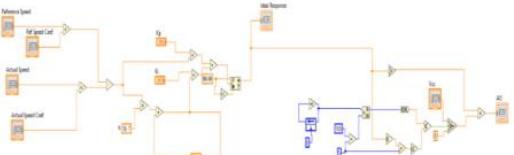
Simple HIL System Definition



Motor controller (DUT)



Motor (Plant)

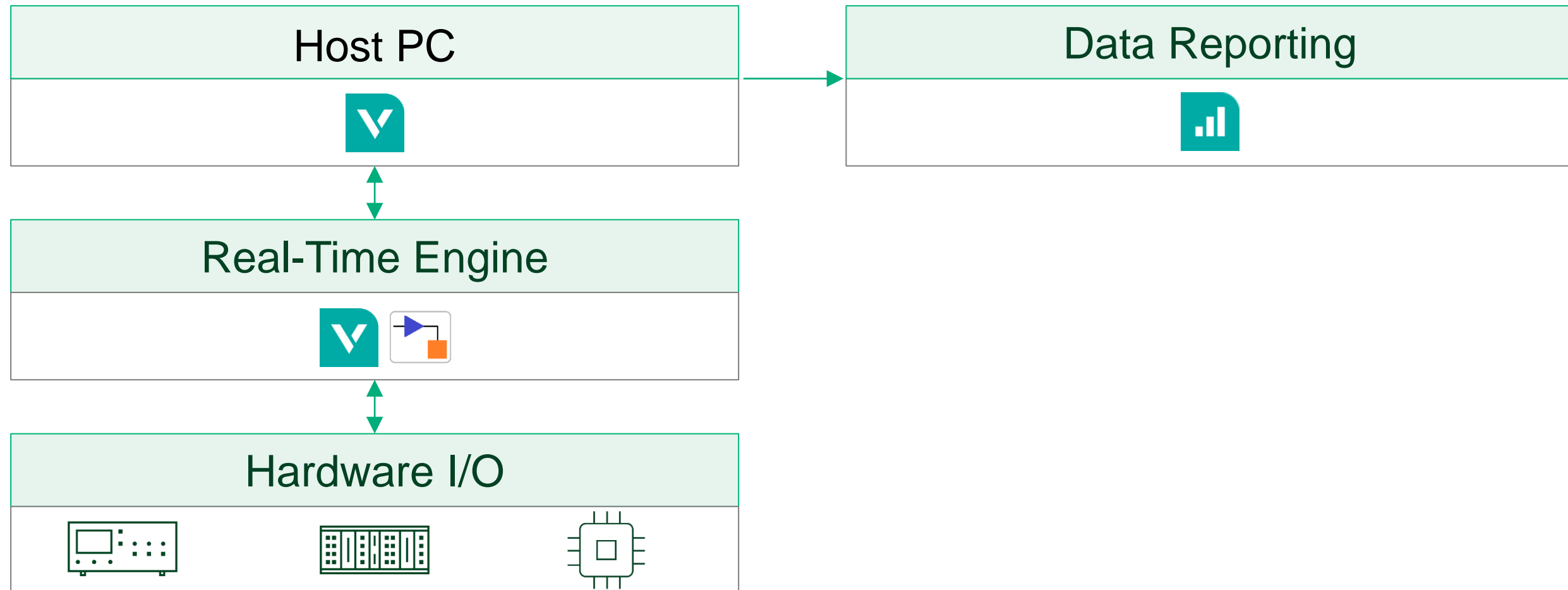


2	0
2	4

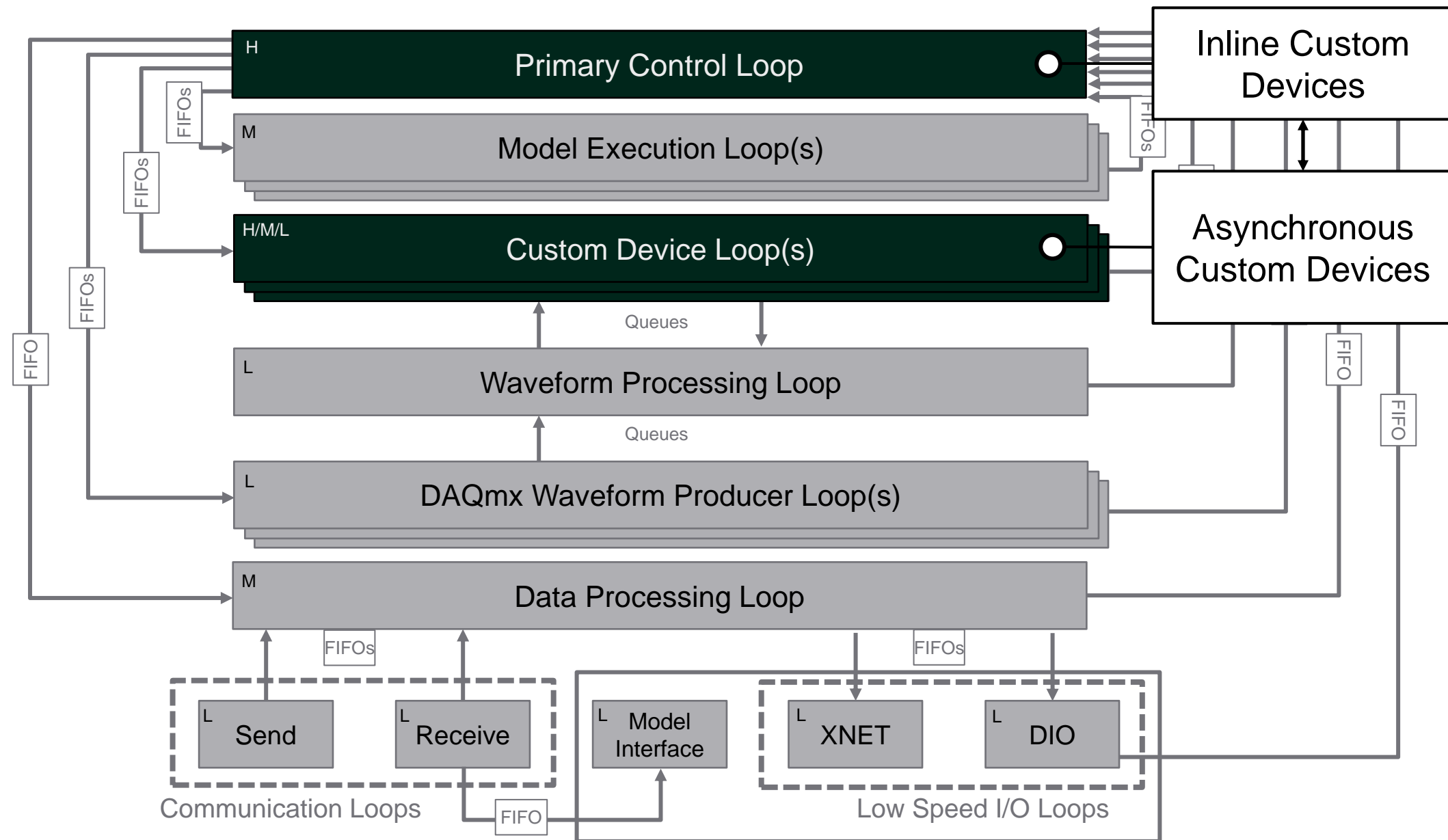
turnon

Manual Validation

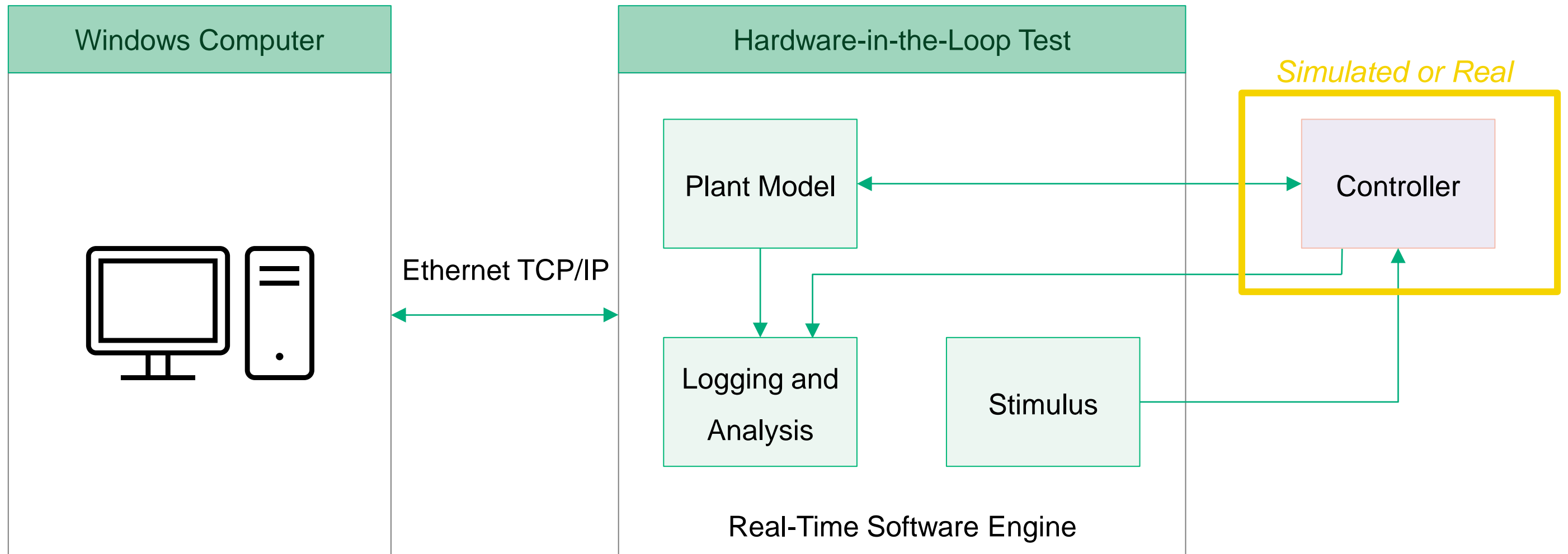
HIL Automation Framework



VeriStand Engine Architecture



Common VeriStand Setup



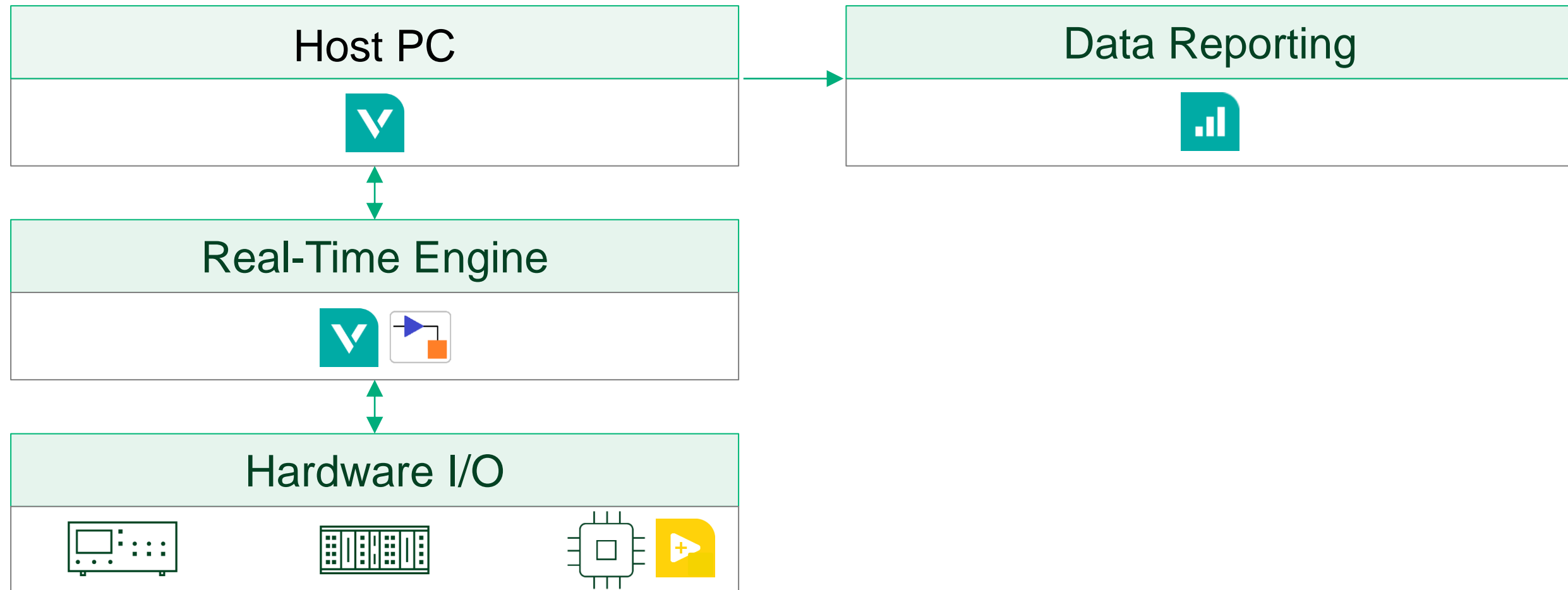
SIL/HIL Demo

2	0
2	4

turnon

High Performance Model Integration

HIL Automation Framework

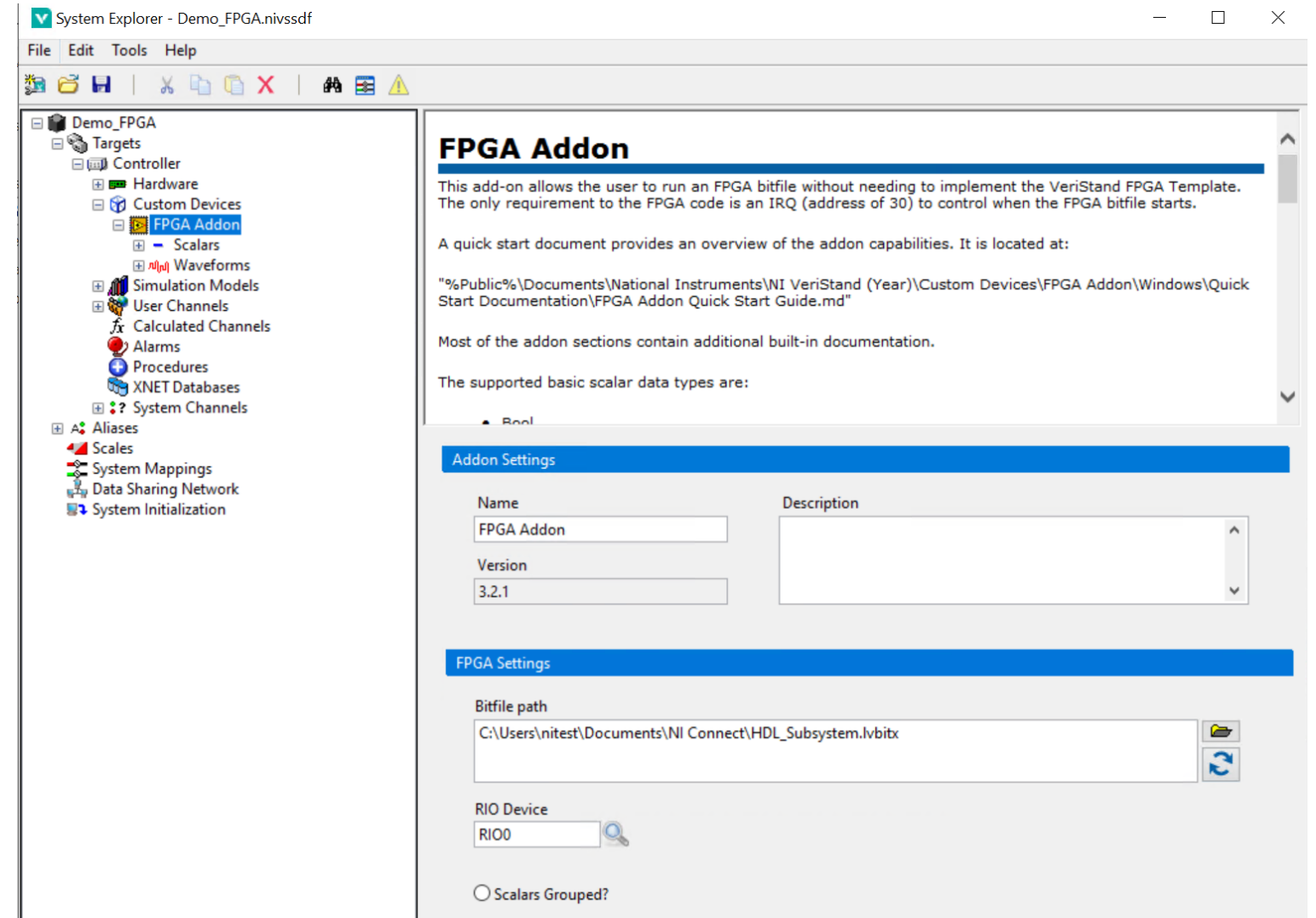


FPGA Integration

FPGA Addon Custom Device ships natively with VeriStand

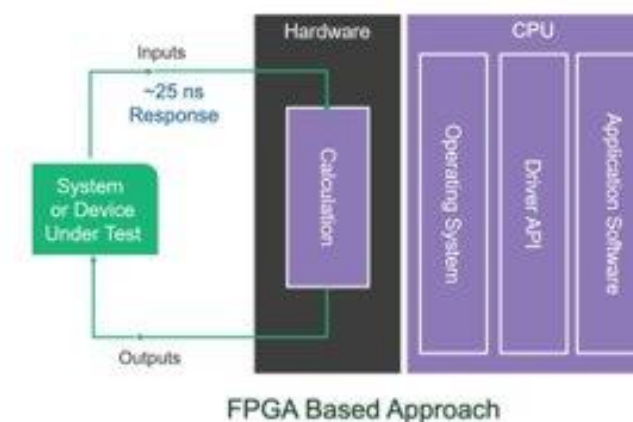
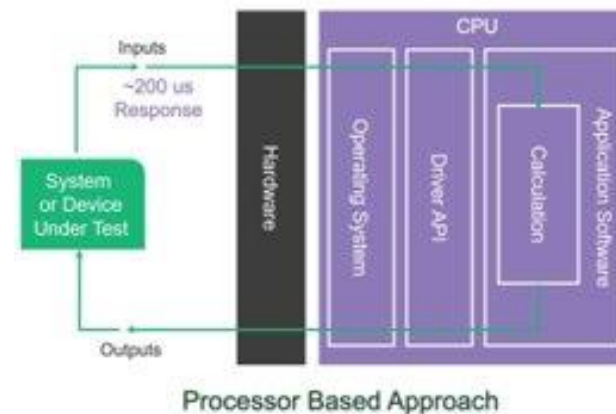
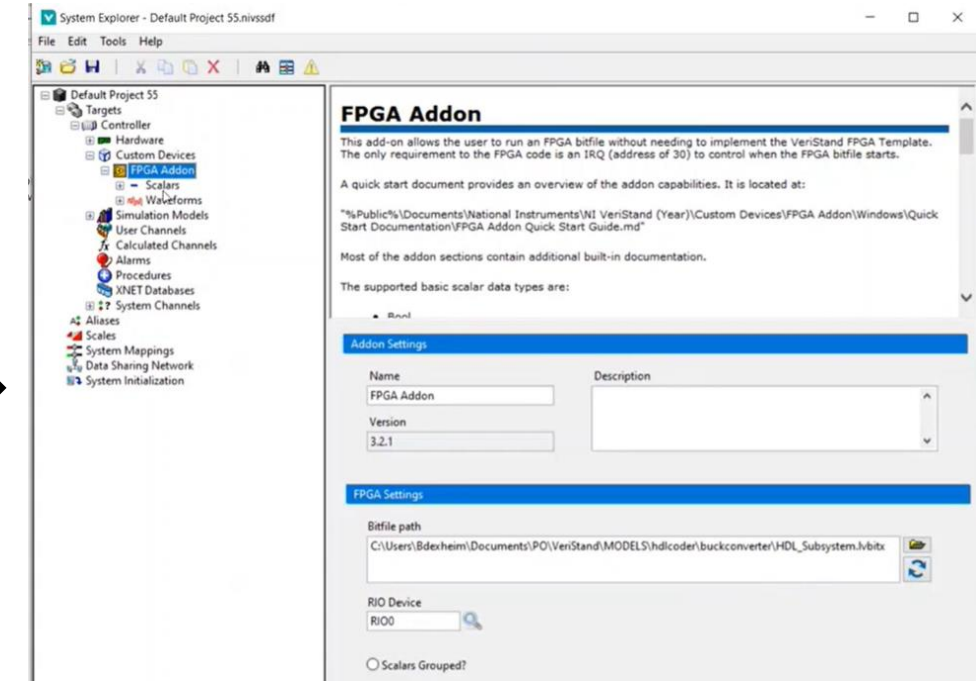
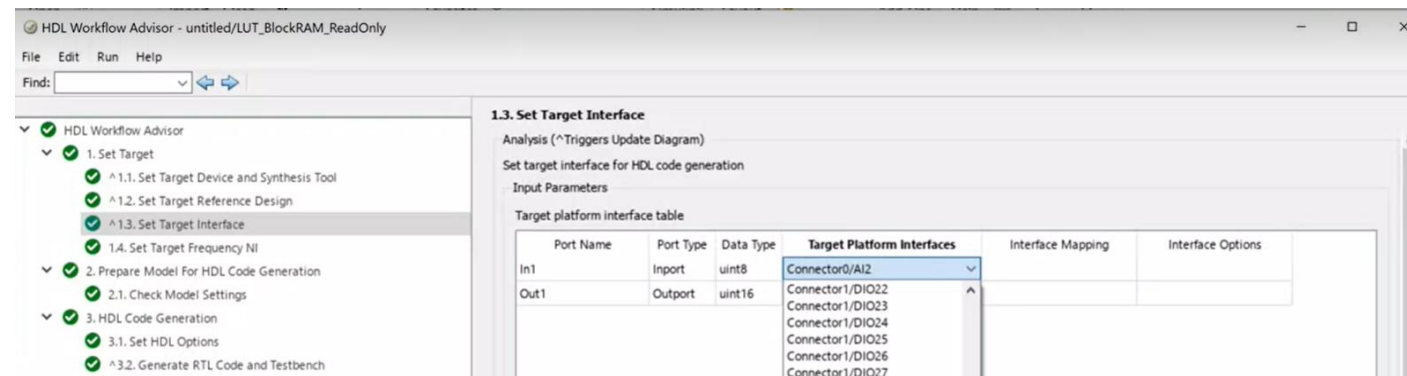
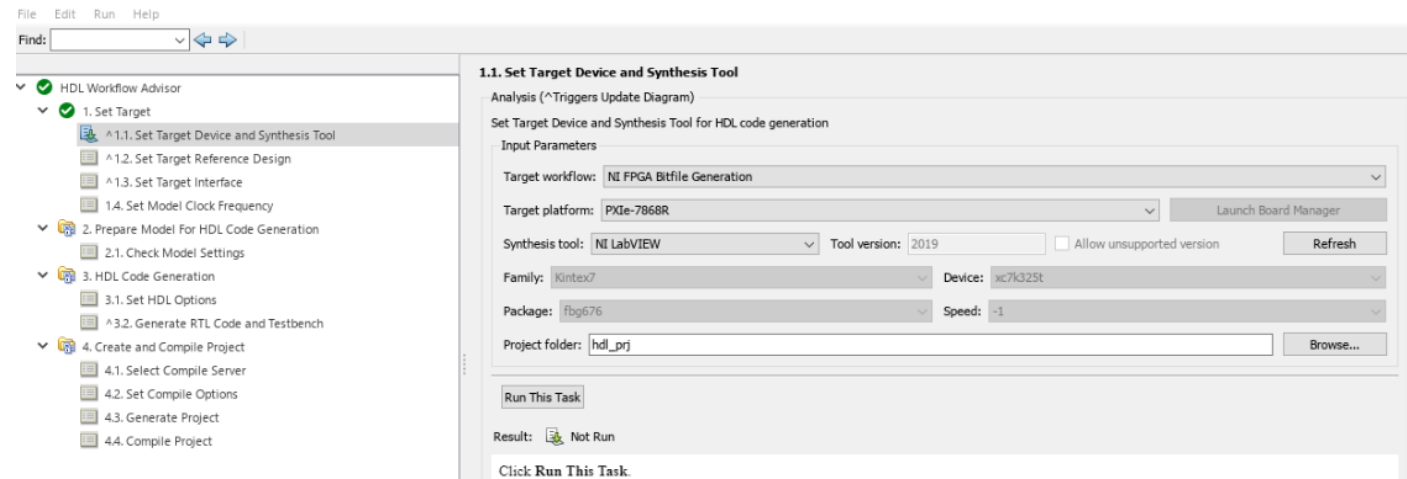
Handles low level communication between the Primary Control Loop and the FPGA

Limited to NI FPGA hardware





HDL Coder Support Package for NI FPGA Hardware



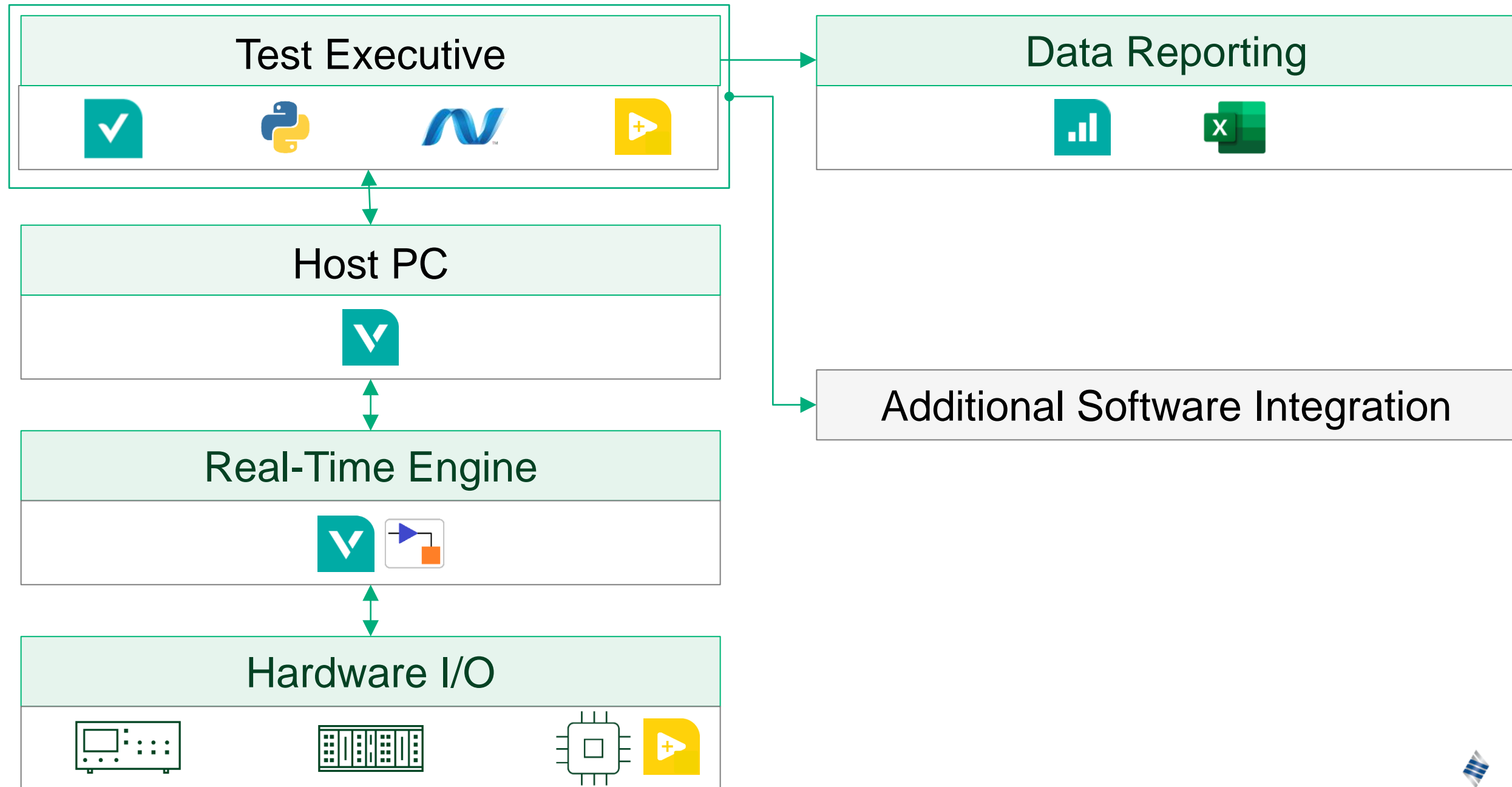
HIL with FPGA Demo

2	0
2	4

testcon

Automate Test Cases

HIL Automation Framework



Control Tuning

Fault Handling

I/O Validation

Parameter Variation

Control Performance Analysis

Control Stability Analysis

State Machines

Thermal Management

Performance Mapping

Sensor Failure


DUT Bring-Up

Safe Operating Regions


Testing Enabled Through HIL

- Validate DUT performance over a wide range of parameter variations to achieve full test coverage
- Verify functionality in range of conditions, including extreme environments not easily created or replicated in the real world
- Map test cases to requirements to ensure complete test coverage
- Perform regression tests with ease to quickly validate design iterations


Reasons to Automate




Reduce engineering effort by creating a library of tests to enable re-use across projects and teams.




Increase the test throughput by automating the HIL system test with a purpose-built test executive.




Synthesize test execution, test data, and requirement tracking to meet regulatory standards.



Extend HIL systems to test many different types of devices, even when new DUTs require changes to the system definition.



Integrate HIL systems as part of a larger ecosystem with technology from several vendors.



Present a custom, tailored application to the end user that leverages VeriStand under the hood.

Early Access: VeriStand Steps for TestStand

File Edit View Execute Debug Configure Source Control Tools Window Help

< > * [Icons] ?

Insertion Palette

Step Types ^

.N .NET

- Tests
 - .N Pass/Fail Test
 - .N Numeric Limit Test
 - .N Multiple Numeric Limit Test
 - .N String Value Test
- .N Action
 - FTP Files
 - Additional Results
 - Sequence Call
 - fx Statement
 - Property Loader
 - Label
 - Message Popup
 - Call Executable
- Flow Control
- Synchronization
- Database
- Data Streams
- LabVIEW Utility
- NI VeriStand
 - Launch VeriStand Project
 - Deploy System Definition
 - Undeploy System Definition
 - Get/Set Channels

Set Engine RPM.seq+

Steps: Set Desired RPM and Wait For Actual RPM

STEP	DESCRIPTION	SETTINGS
- Setup (1)	Specifies the VeriStand system definition to deploy with a relative path string expression.	
Deploy System Definition	System Definition Path: @"VeriStand Project\VeriStand TestStand Steps...."	
<End Group>		
- Main (6)	Assigns the desired RPM value to a local variable	
fx Statement	Locals.DesiredRPM = 5000	
Turns the engine on and updates channel values to set the desired RPM.		
Get/Set Channels	2 channels configured	
Polls for the actual RPM value until reaching the desired value.		
While	Abs(Locals.ActualRPM - Locals.DesiredRPM) > 1	
Reads the actual RPM.		
Get/Set Channels	Read "Aliases/ActualRPM"	
Waits a set time interval before polling again.		
Wait	TimeInterval(0.25)	
End		
<End Group>		
- Cleanup (1)	Undeploys the system definition.	
Undeploy System Definition		

Step Settings for Get/Set Channels

Get/Set Channels Properties

CHANNELS	UNITS	ACTION	CHANNEL VALUES
"Targets/Controller/Simulation Models/Models/Engine Demo/Inport..." fx ✓		Set	True
"Aliases/DesiredRPM" fx ✓	rpm	Set	Locals.DesiredRPM

TestStand Automation Demo

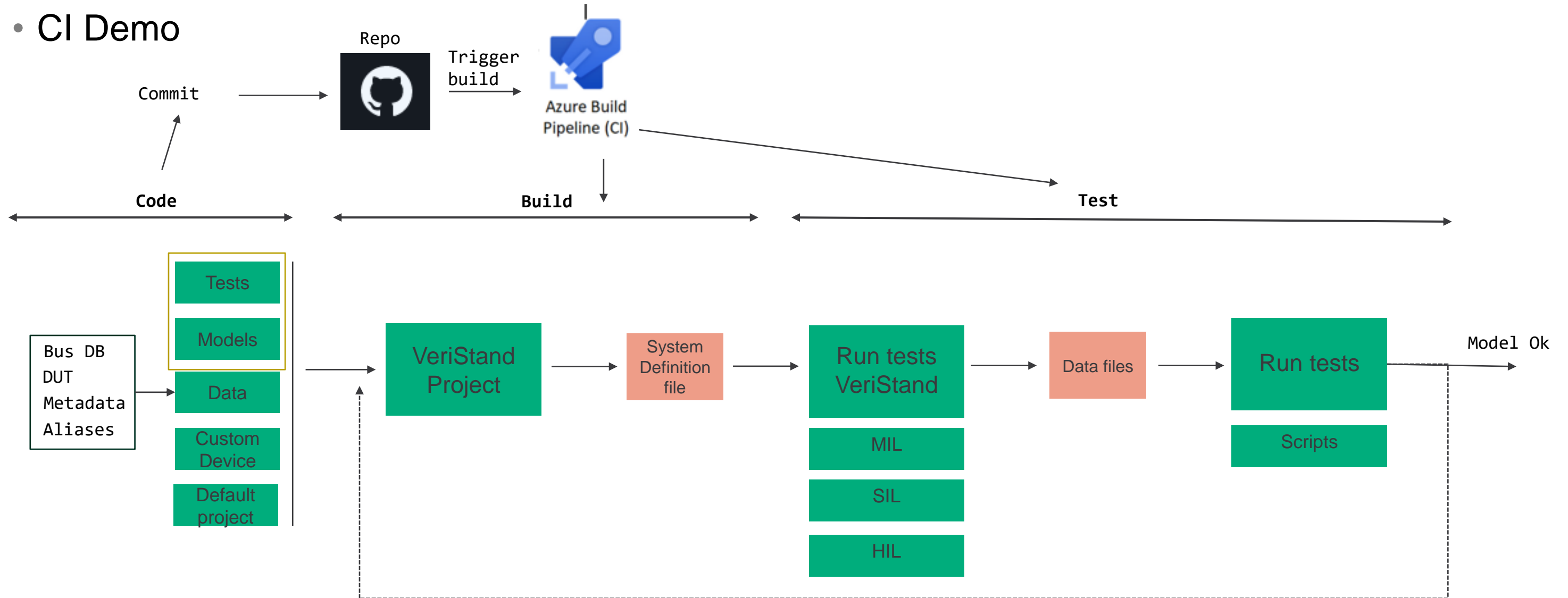
2	0
2	4

tutorialspoint

Create a CI/CD Workflow

Automation - Technical Workflow

- CI Demo



SystemLink Enterprise Overview

SystemLink™ streamlines lab operations and amplifies engineering insights as an integrated, scalable, enterprise solution.

Specification Compliance

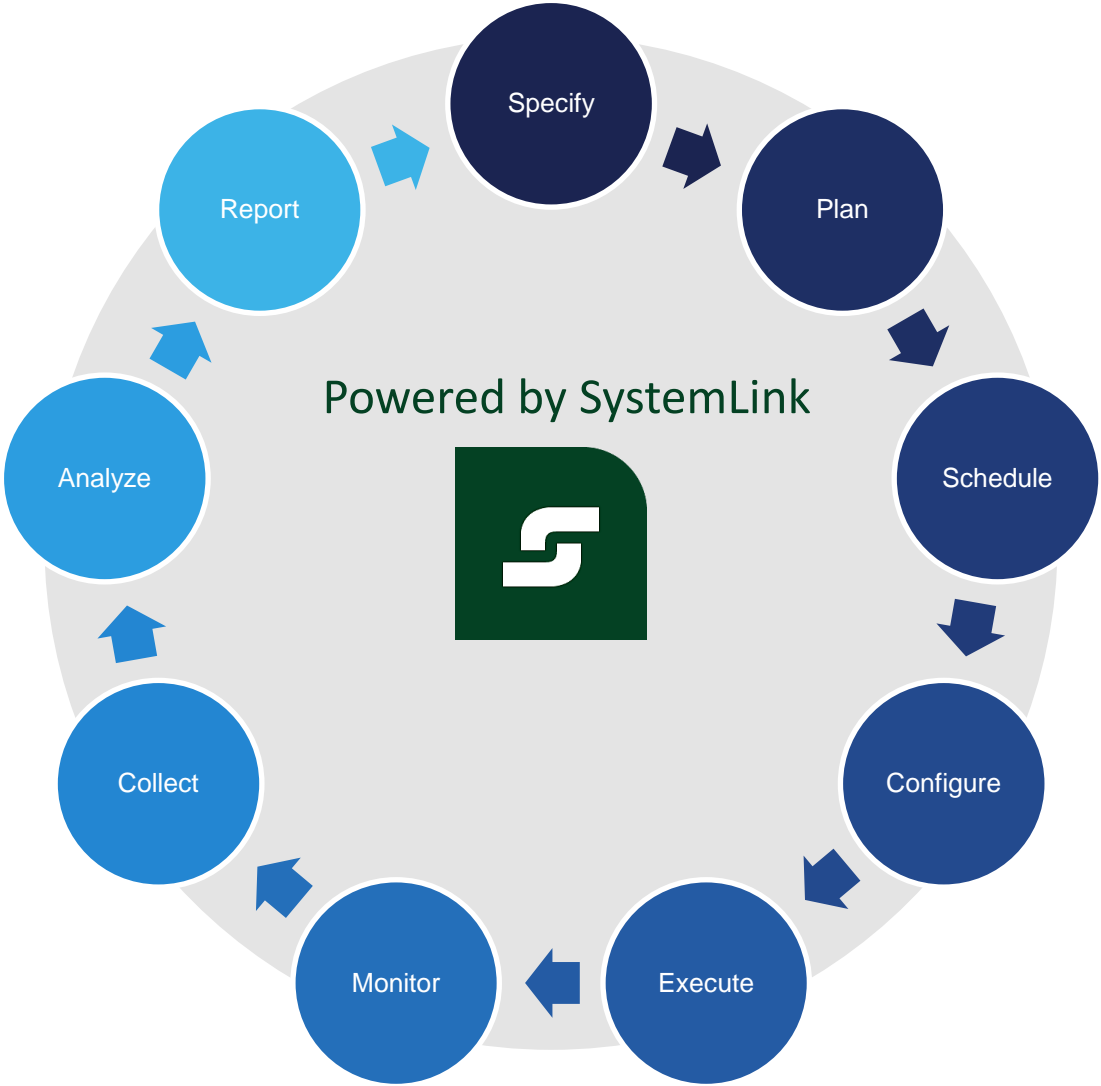
Elaborate requirements into engineering specifications to drive test consistency, quickly discover product issues, and track test progress.

Analysis Automation & Reporting

Fully integrated Jupyter Notebook development environment to create Python scripts to extract, transform, and analyze data.

Test Monitoring & Insights

Collect and view test results, files and parametric data; filter data for additional insights and track KPIs with dashboards and customizable analytics.



Work Orders & Test Plans

Track incoming test requests, define test requirements, schedule, deploy, and remotely execute tests.

Systems & Asset Management

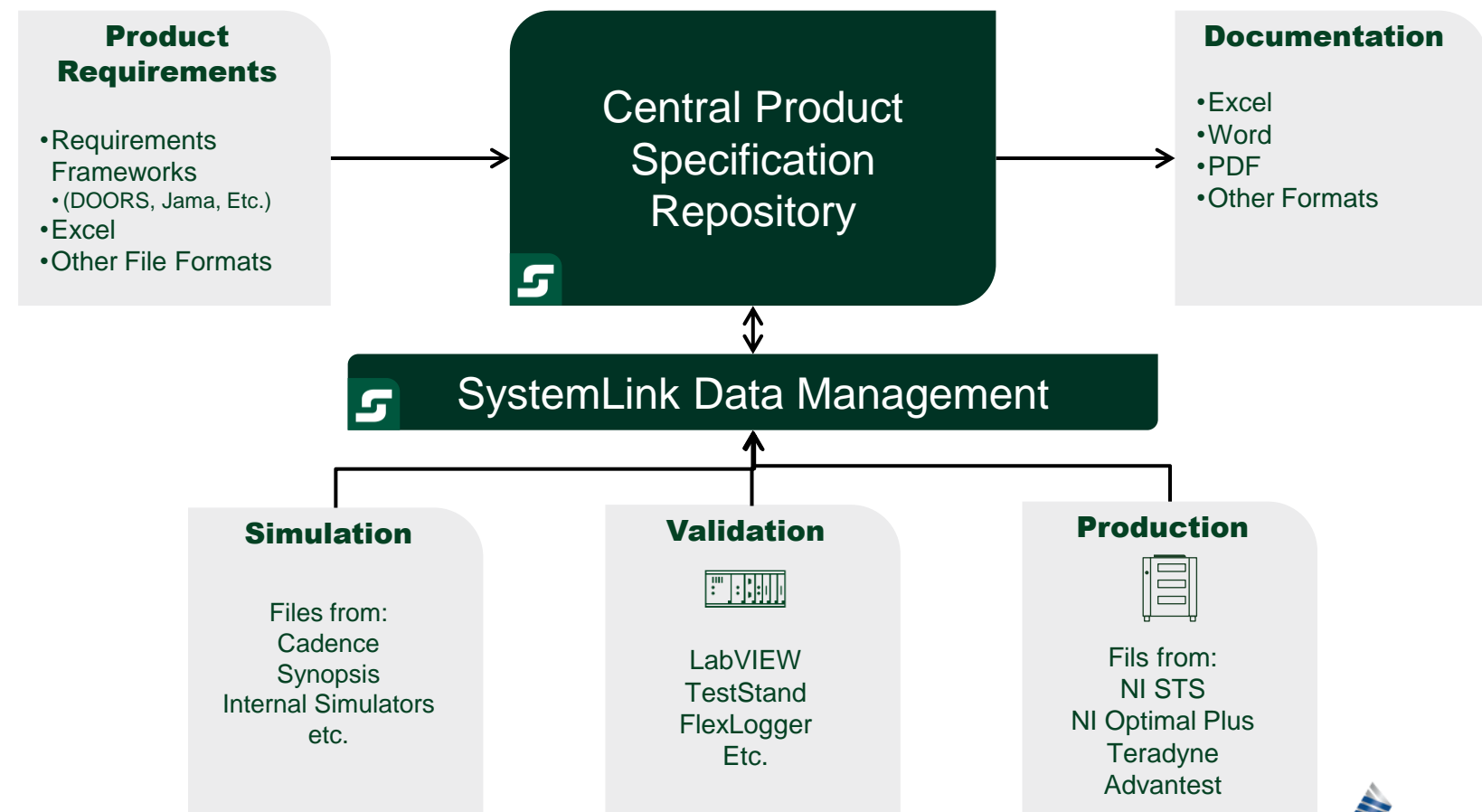
Manage and install software for your entire test fleet, monitor test system health, and manage and track the assets connected to your test systems.

Asset Traceability & Utilization

Automatic asset tracking for NI and 3rd party LXI, USB-TMC, GPIB instruments to maximize utilization and optimize spend.



- Central repository for specifications
- Connect to and analyze measurements from across the product lifecycle
- Specification compliance and coverage reporting
- Custom measurement & specification analysis.



Q & A
