MathWorks
AUTOMOTIVE
CONFERENCE 2023
Europe

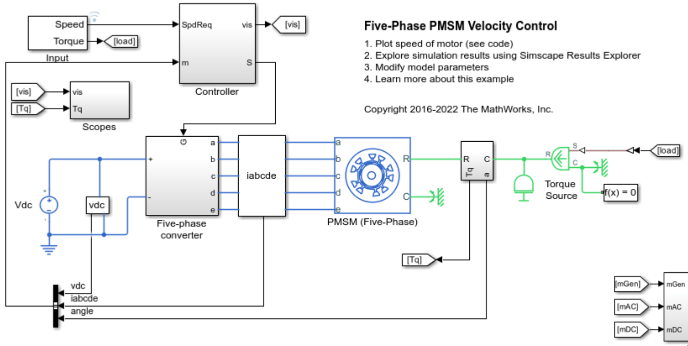
Making the Most of FPGAs for Automotive Power Electronics and Control Design

Dimitri Hamidi

MathWorks®

Power Electronics Key Role in Electrification

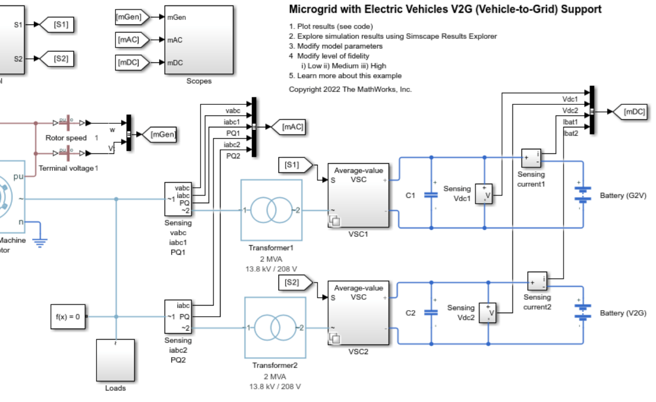
Motor Control



Five-Phase PMSM Velocity Control

1. Plot speed of motor (see code)
2. Explore simulation results using Simscape Results Explorer
3. Modify model parameters
4. Learn more about this example

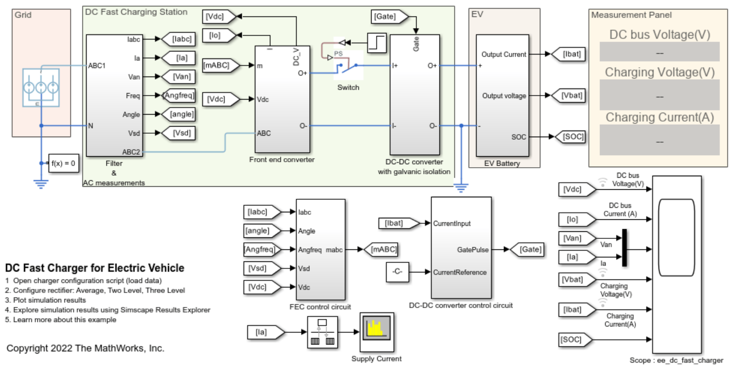
Vehicle-to-Grid



Microgrid with Electric Vehicles V2G (Vehicle-to-Grid) Support

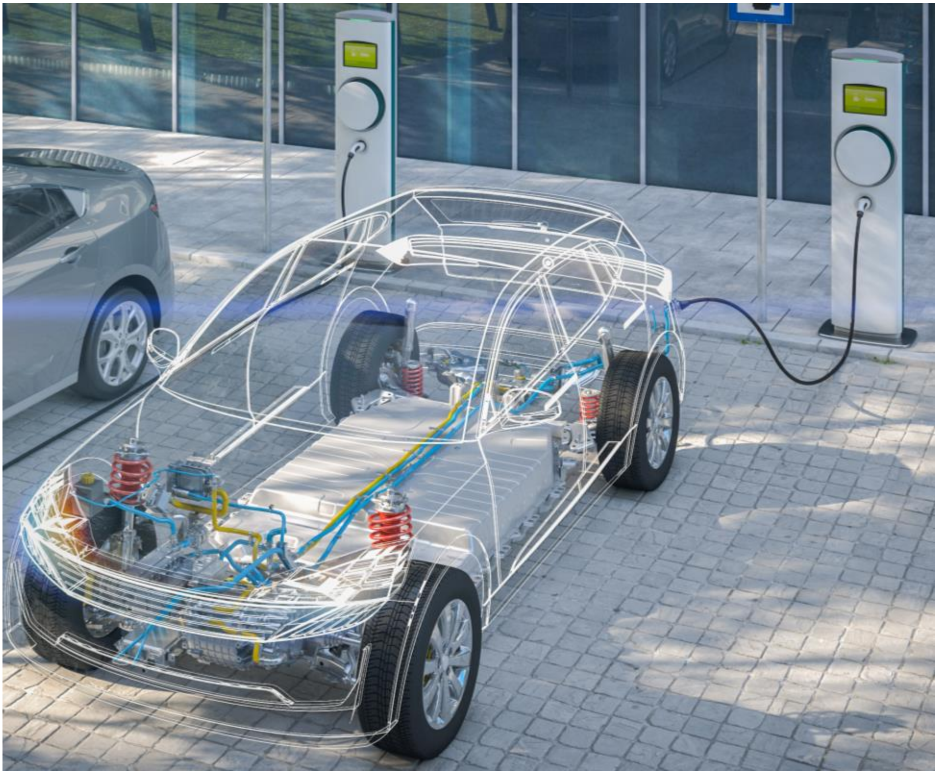
1. Plot results (see code)
2. Explore simulation results using Simscape Results Explorer
3. Modify model parameters
4 Modify level of fidelity
   i) Low ii) Medium iii) High
5. Learn more about this example

EV Charger

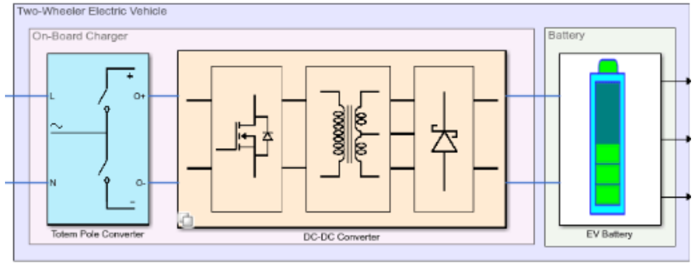


DC Fast Charger for Electric Vehicle

1 Open charger configuration script (load data)
2. Configure rectifier: Average, Two Level, Three Level
3. Plot simulation results
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example



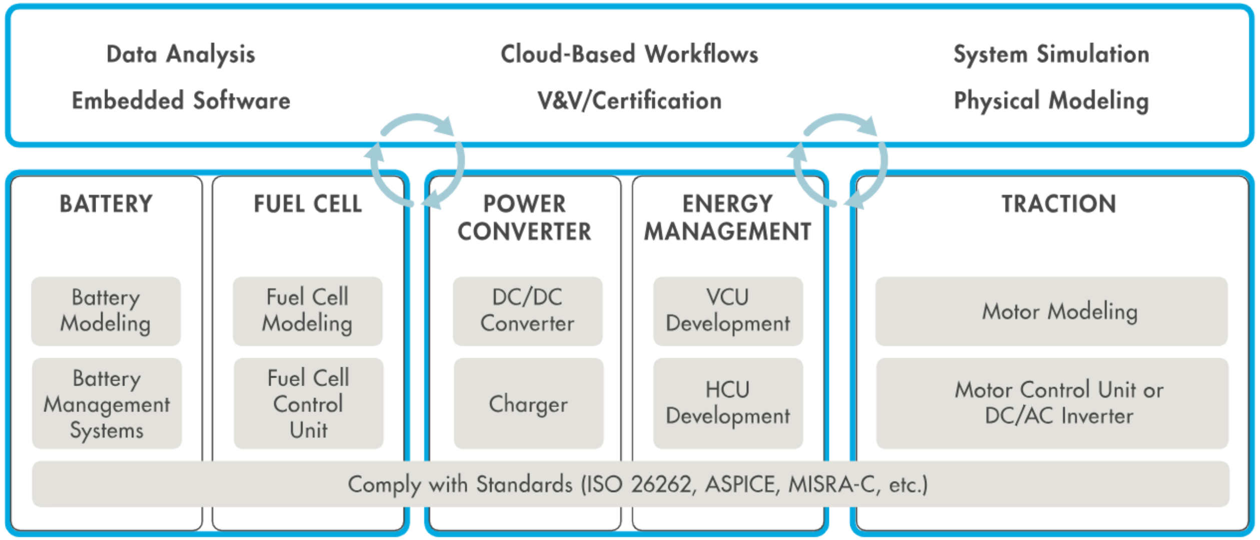
On-Board Charger



Great power electronics control design results in an elegant marriage of hardware and embedded software

But like any marriage, it comes with challenges:

- Sizing and specifying electrical components, motor, battery …
- Developing control algorithms and tuning controller gains for varying operating conditions and component degradation
- Developing software for a complete range of supervisory and fault mitigation conditions
- Hardware-software integration, mistakes first found during this stage are the hardest to troubleshoot and most expensive to fix



Data Analysis
Embedded Software
Cloud-Based Workflows
V&V/Certification
System Simulation
Physical Modeling

BATTERY
Battery Modeling
Battery Management Systems

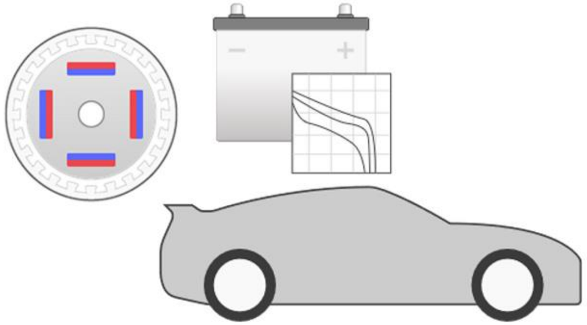
FUEL CELL
Fuel Cell Modeling
Fuel Cell Control Unit

POWER CONVERTER
DC/DC Converter
Charger

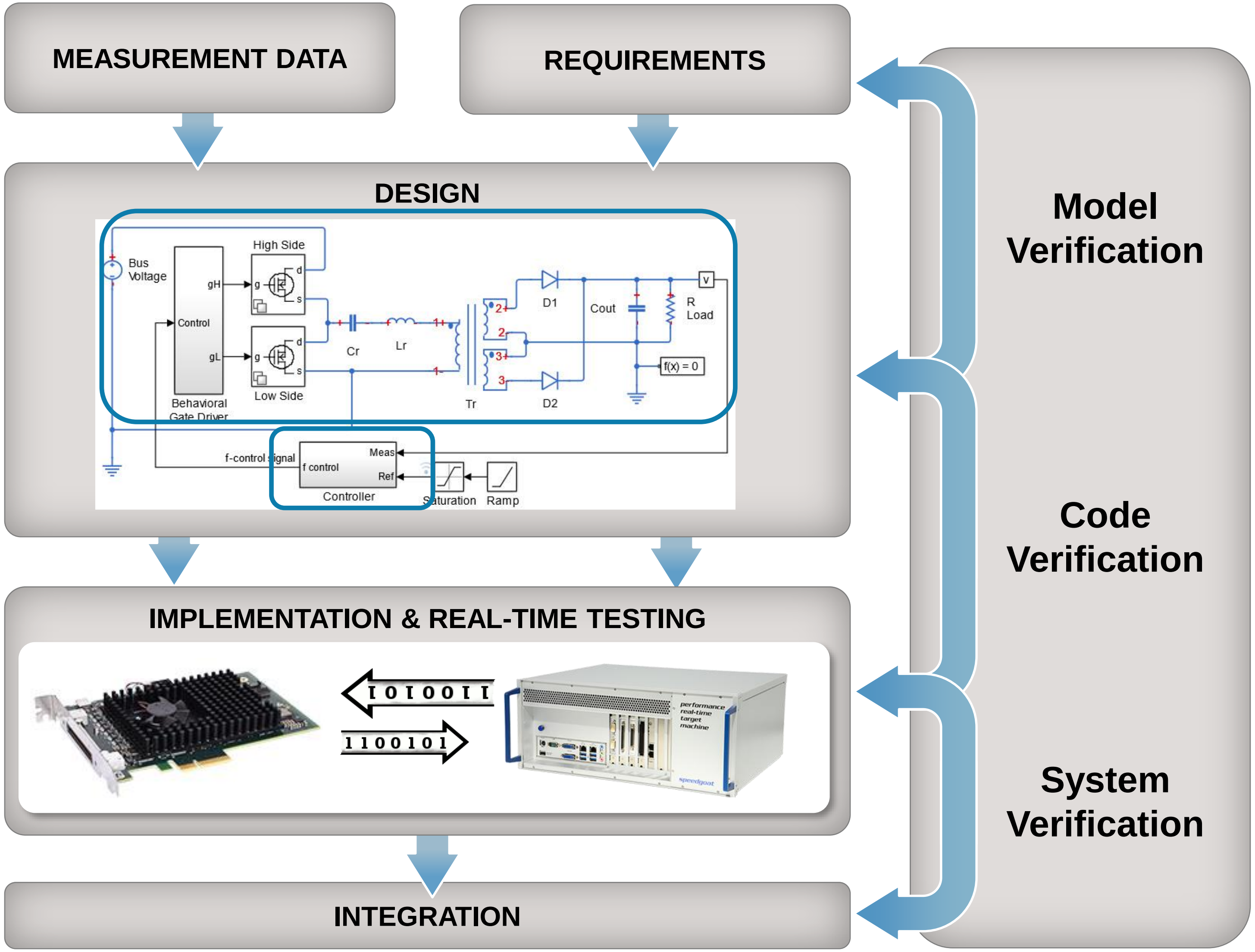
ENERGY MANAGEMENT
VCU Development
HCU Development

TRACTION
Motor Modeling
Motor Control Unit or DC/AC Inverter

Comply with Standards (ISO 26262, ASPICE, MISRA-C, etc.)



Benefits of Model-Based Design for Power Electronics Control



MEASUREMENT DATA

REQUIREMENTS

DESIGN

IMPLEMENTATION & REAL-TIME TESTING

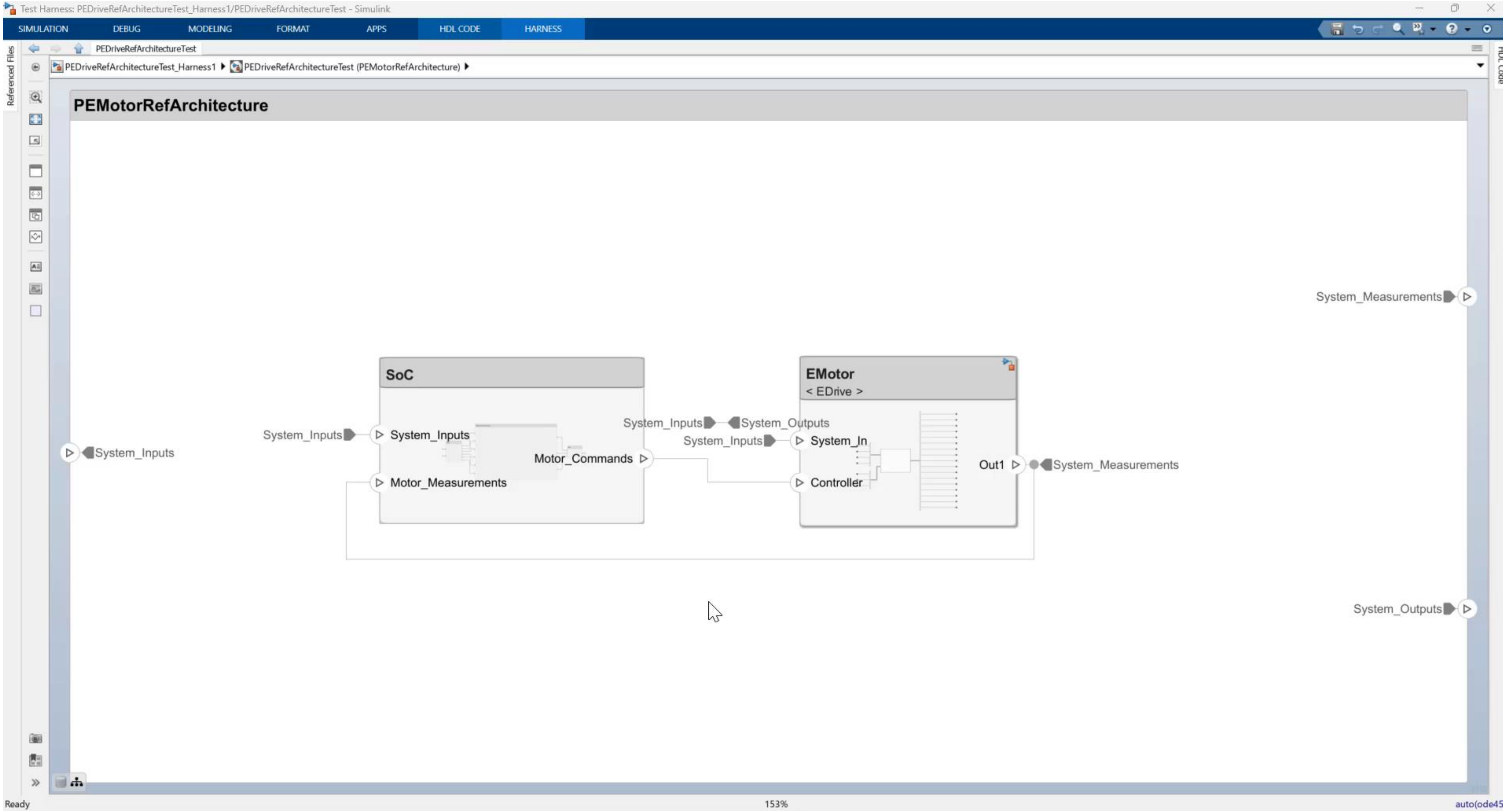
INTEGRATION

Model Verification

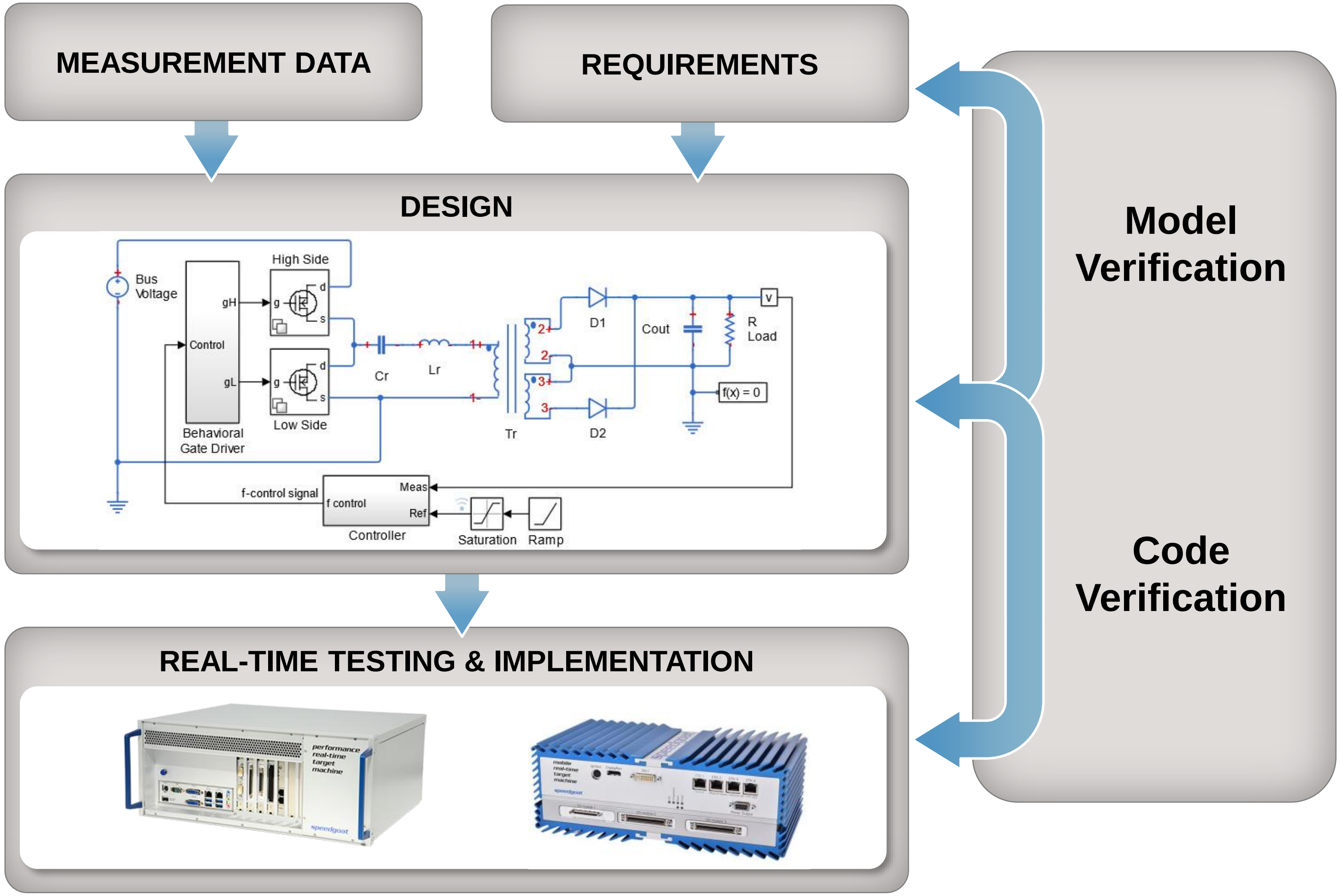
Code Verification

System Verification

- Understand HW-SW system behavior early in the project
- Handle system complexity and design changes
- Reduce the time for design cycles
- Generate code for practically any hardware platform
- Verify control algorithms using desktop and real-time simulations


Test Harness: PEDriveRefArchitectureTest_Harness1/PEDriveRefArchitectureTest - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
HDL CODE
HARNESS
PEDriveRefArchitectureTest
PEDriveRefArchitectureTest_Harness1
PEDriveRefArchitectureTest (PEMotorRefArchitecture)
Referenced Files
PEMotorRefArchitecture
System_Measurements
SoC
EMotor
< EDrive >
System_Inputs
System_Outputs
System_Inputs
System_In
System_Inputs
System_Inputs
Motor_Commands
Out1
System_Measurements
Controller
Motor_Measurements
System_Outputs
Ready
153%
auto(ode45

Model-Based Design – Verify with Real-Time Simulation UC1



MEASUREMENT DATA

REQUIREMENTS

DESIGN

REAL-TIME TESTING & IMPLEMENTATION

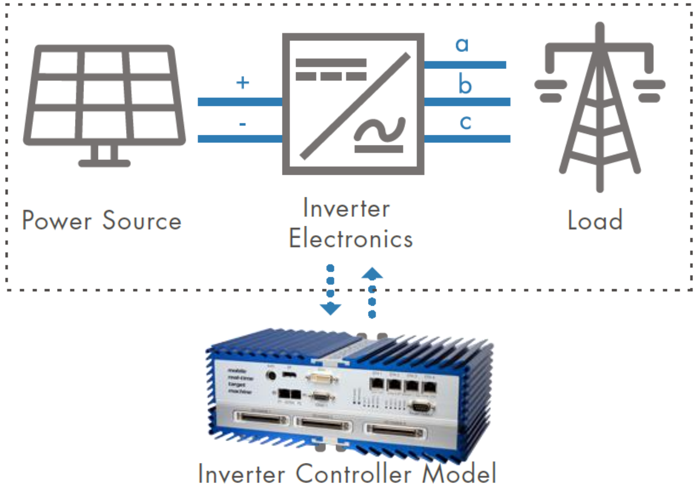
Model Verification

Code Verification

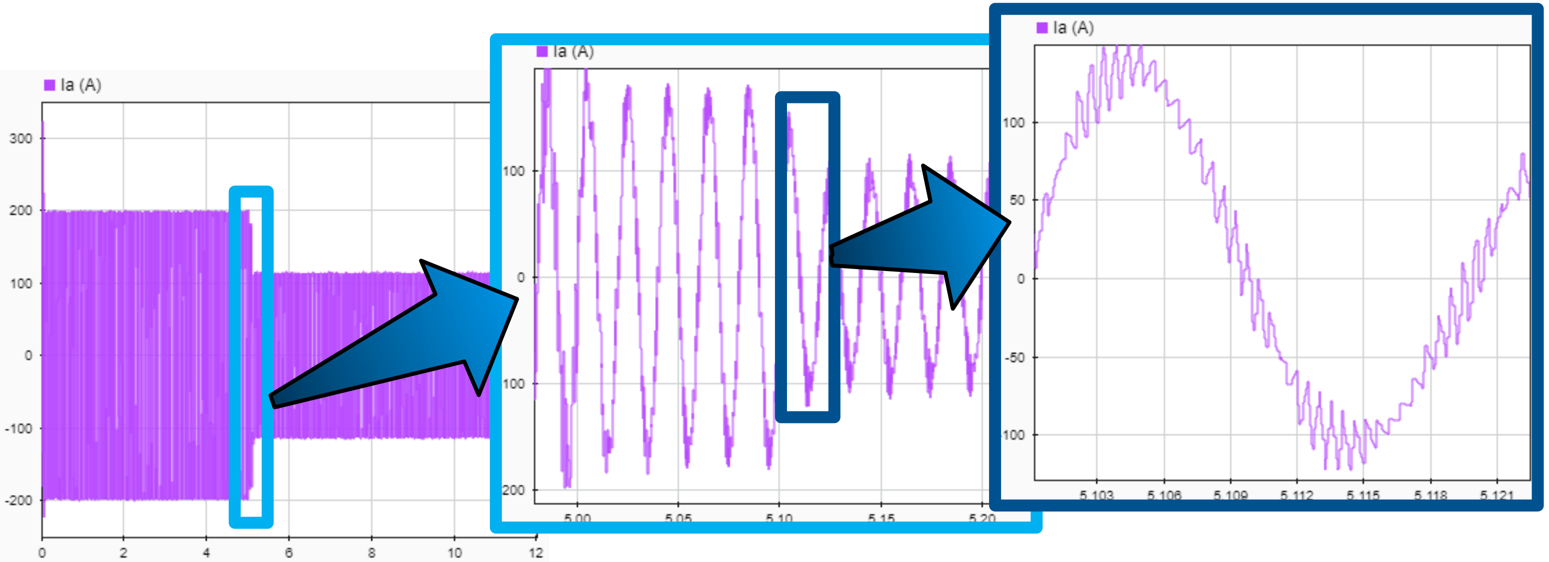
- Generate C or HDL code for Rapid Control Prototyping (RCP)
- Generate C or HDL code for Hardware-in-the-Loop (HIL) testing
- Test design changes using the *same systematic verification methods*

Why FPGAs? Real-Time Simulation for Power Electronics and Motor

- GaN and SiC device simulation requires time steps that go beyond what is achievable on CPUs.



Power electronics system for HIL simulation



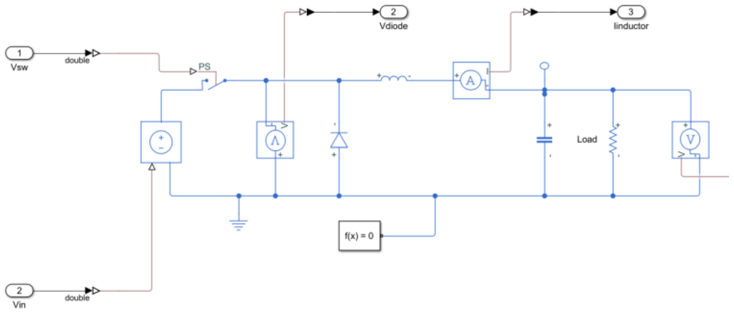
CPU HIL works well for simulation time steps up to 25us

FPGA HIL can run at simulation time steps of 1us

Workflow from Simscape Electrical to FPGA

R2020a

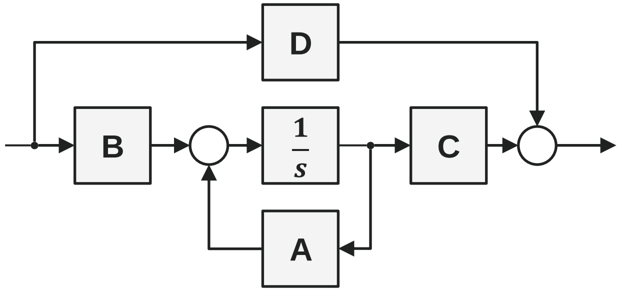
1



Simscape Electrical model

Simscape™ HDL Workflow Advisor

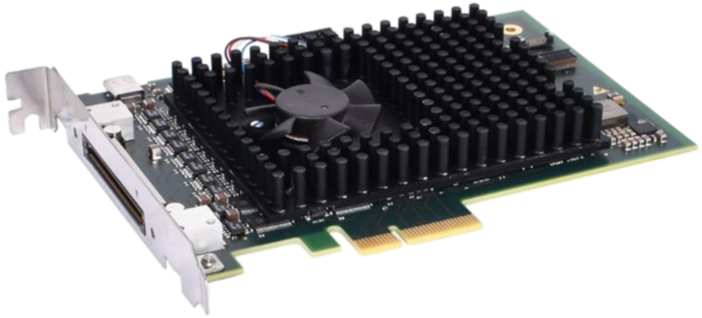
2



State-space description and solver in Simulink

VHDL/Verilog

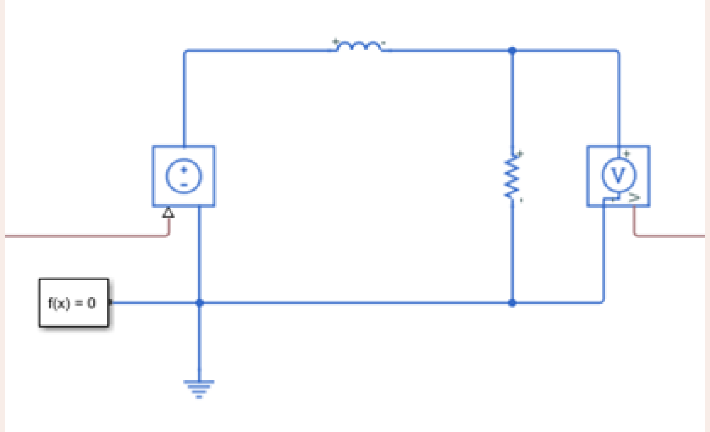
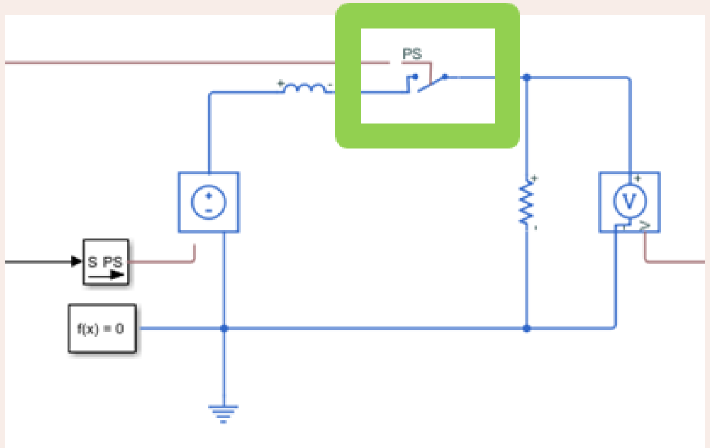
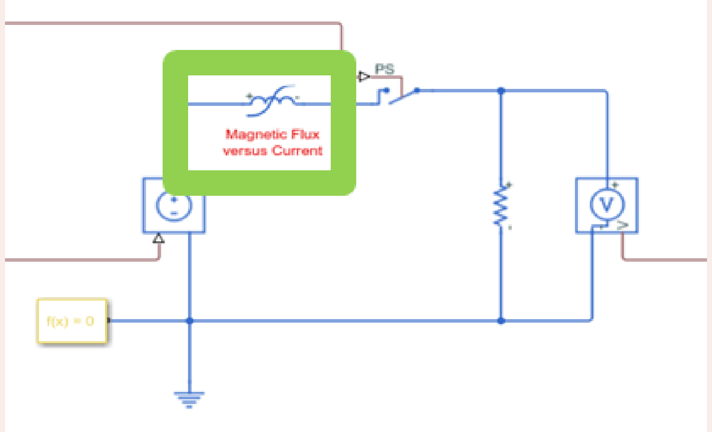
3



FPGA

Nonlinear Simulations on FPGA Supported from R2021a

R2021a

Linear	**Switched-linear**	**Nonlinear**
C or HDL	**C or HDL**	**C, HDL**
		

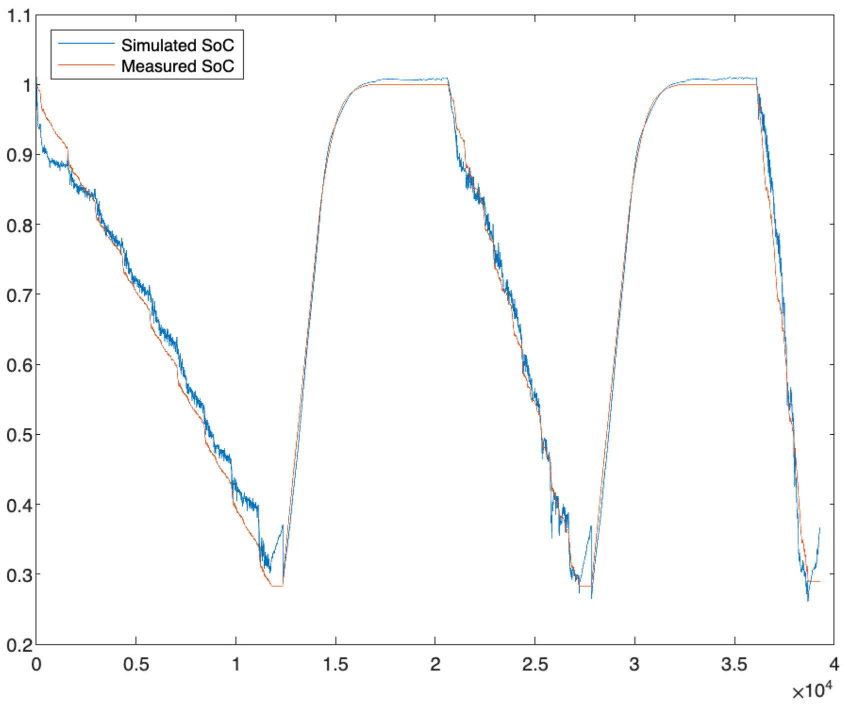
Nonlinear PMSM FPGA for HiL Example

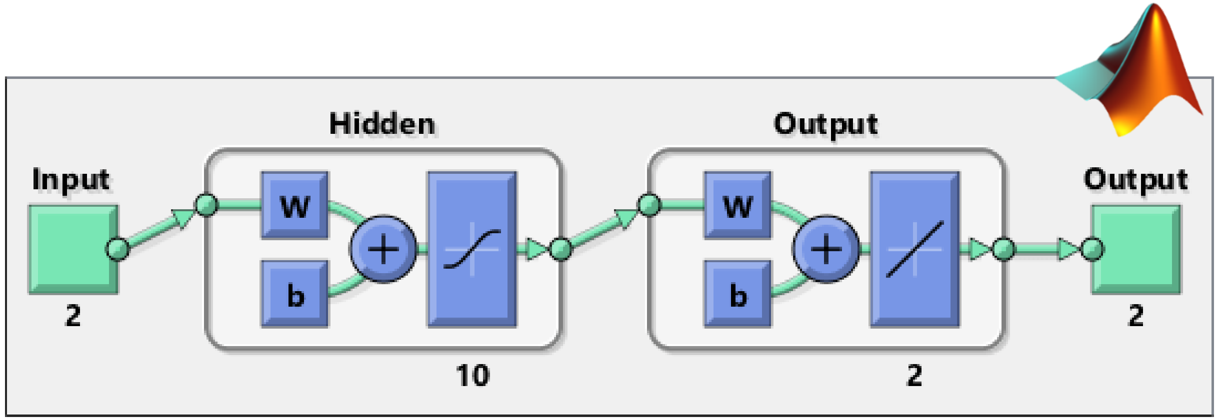
FireFly™ cable with 4x MGT

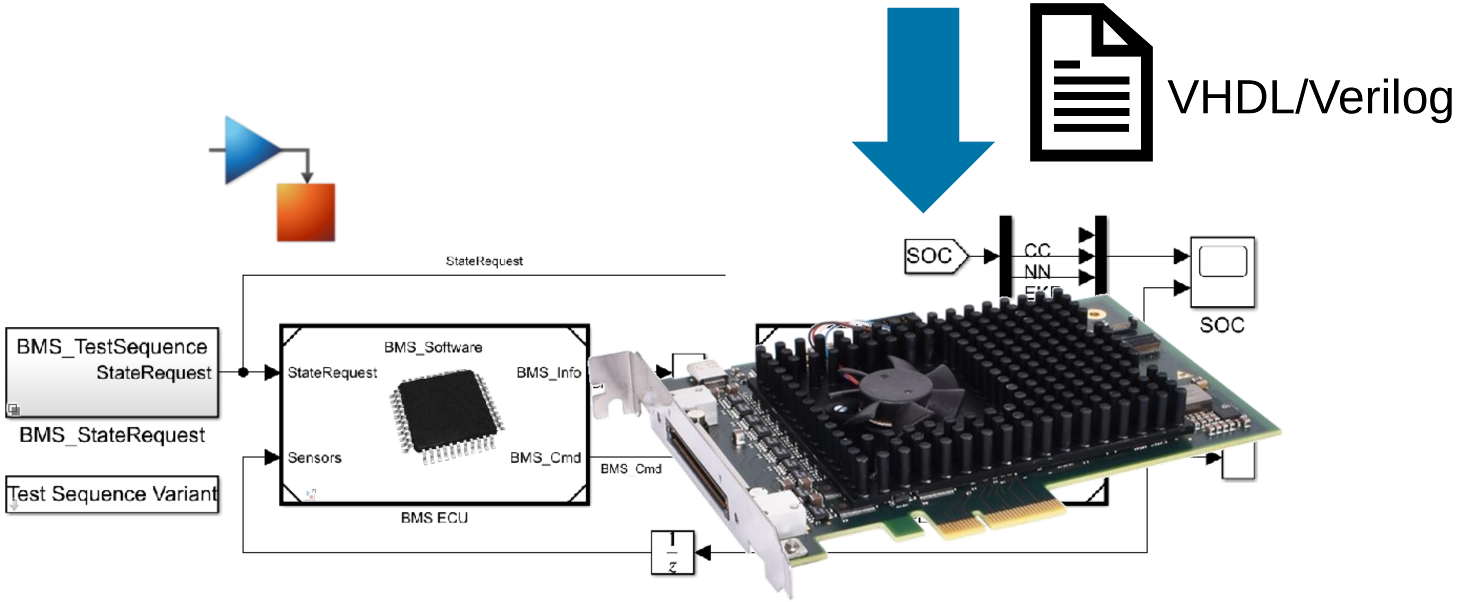
lug-in

nth.rpt.

Deploy Neural Network Regression Model to FPGA Platforms

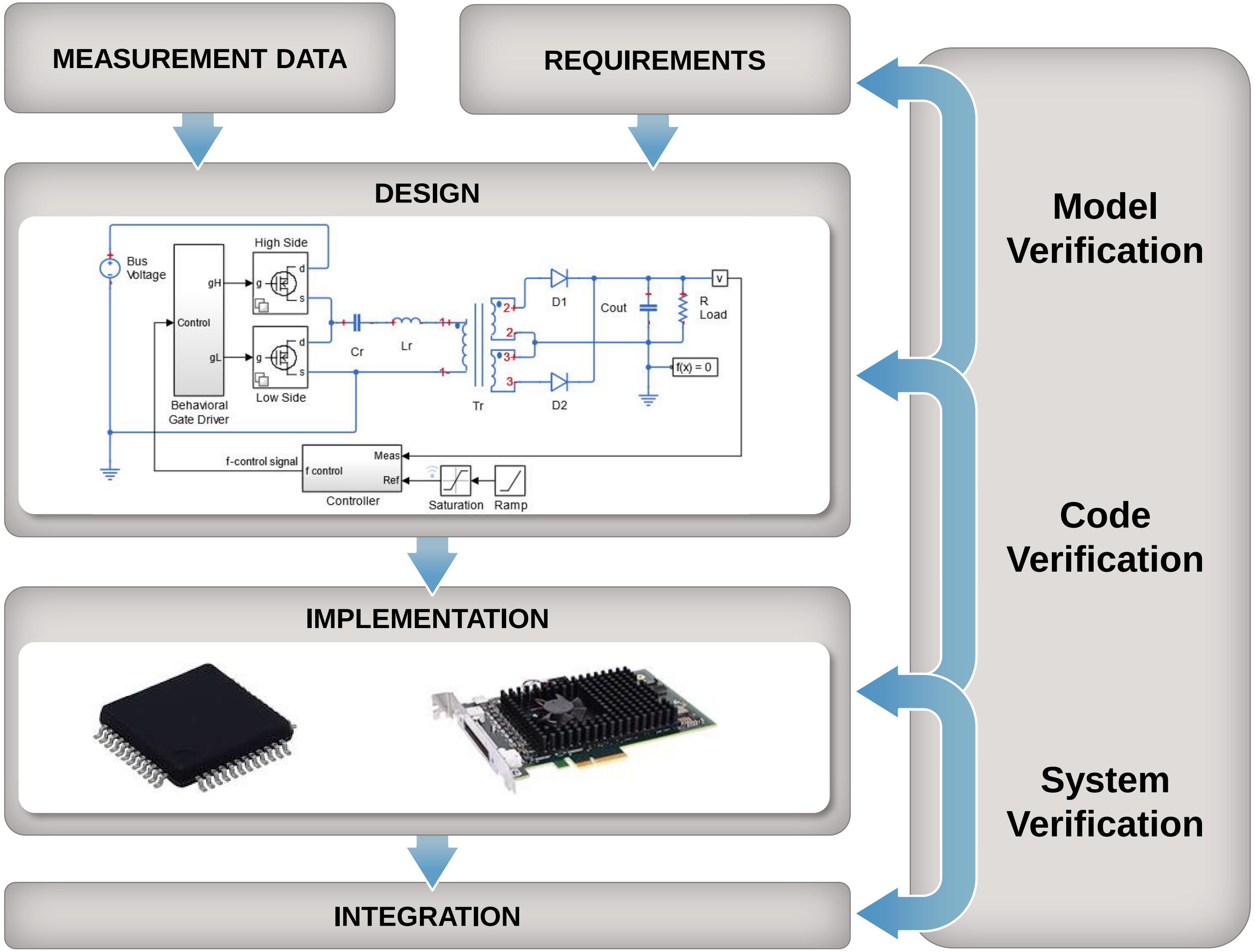






https://www.mathworks.com/help/releases/R2023a/stats/deploy-neural-network-regression-model-to-fpga-platform.html

Model-Based Design – Generate Production-Ready Code – UC2



MEASUREMENT DATA

REQUIREMENTS

DESIGN

IMPLEMENTATION

INTEGRATION

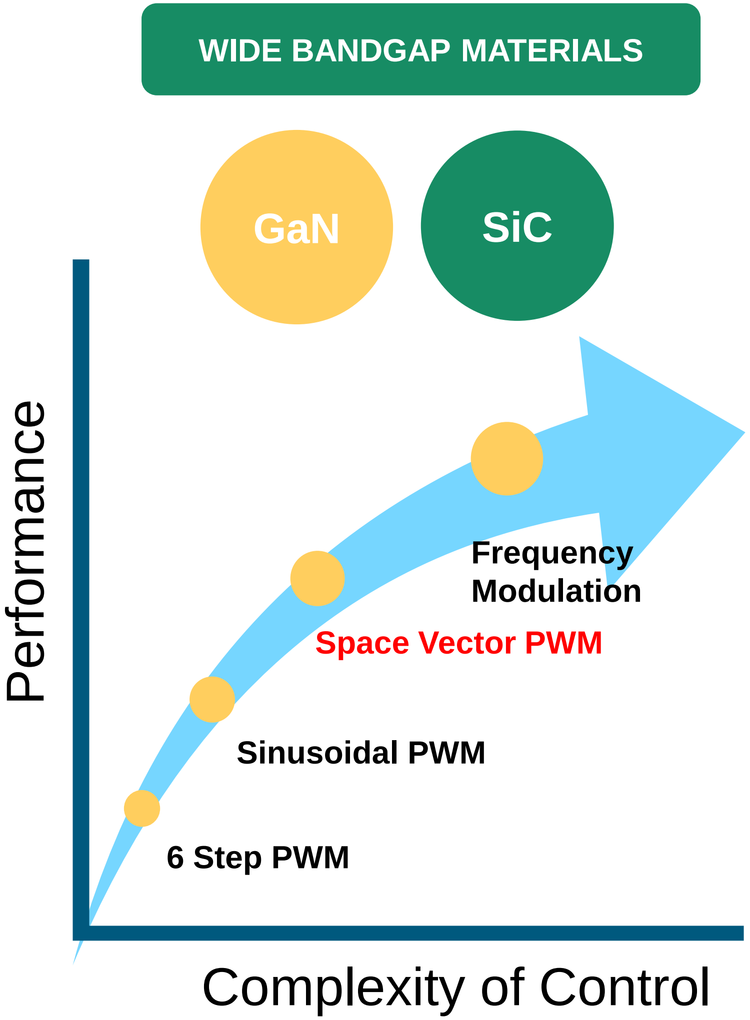
Model Verification

Code Verification

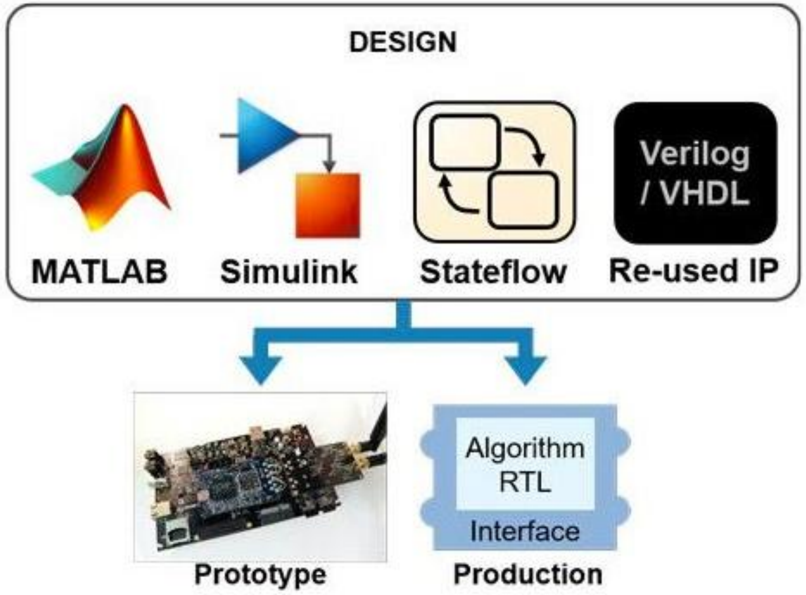
System Verification

- Generate *C or HDL* code for microcontrollers, FPGAs, and ASICs
- Verify HDL code with co-simulation and FPGA-in-the-Loop
- Verify C Code with Software-in-the-Loop (SIL) and Processor-in-the-Loop (PIL) testing

Why FPGAs? Real-Time Performance for Control Algorithms!



Deployment on FPGAs & ASIC Design

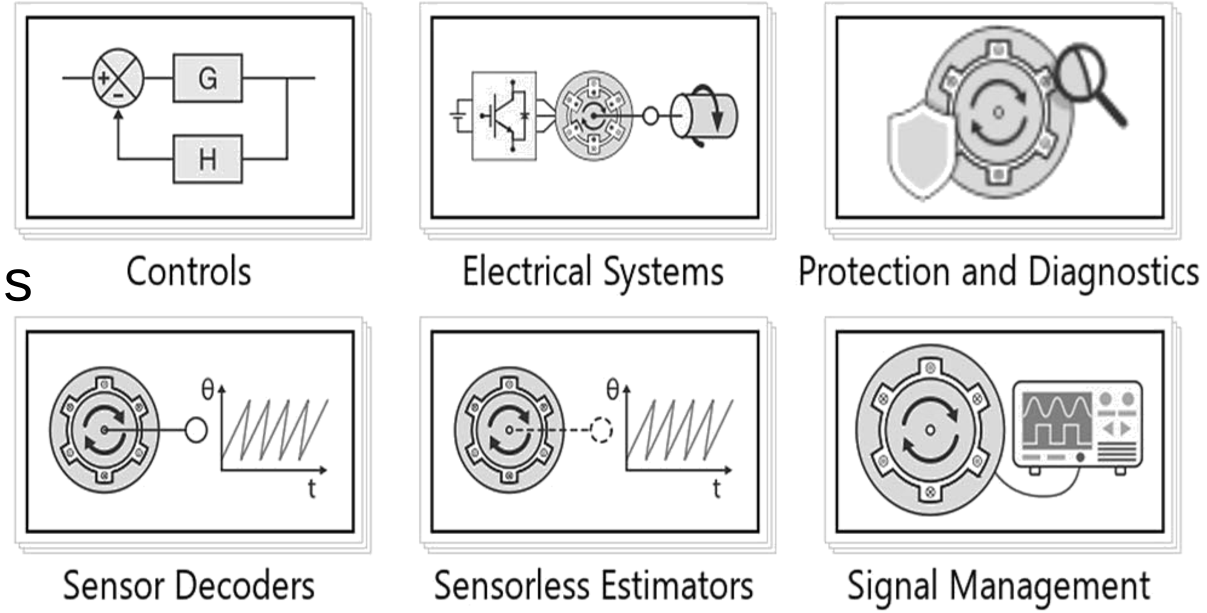


Certification
Deterministic
Flexible Interfaces
Performance
Modularity
Low Latency
Reconfigurability
Parallel Processing

Accelerate Your Motor Control Development for Electric Vehicles

- Simulink blocks for creating and tuning field-oriented control, field-weakening control, and other algorithms for BLDC and PMSM
- Verify control algorithms in closed-loop simulation using motor and inverter models
- Parameter estimation tool for accurate estimation of stator resistance back EMF, inertia, and friction ….
- generate C and HDL code for production

Motor Control Blockset R2020a

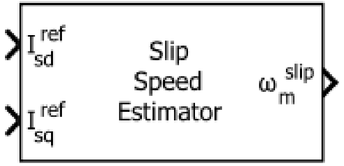


Motor Control Blockset HDL Support

Control Reference

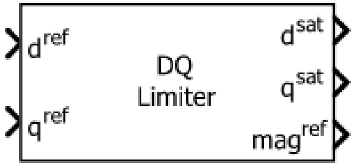
ACIM Control Reference

ACIM Feed Forward Control

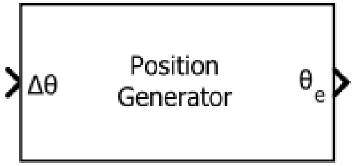


ACIM Slip Speed Estimator

ACIM Torque Estimator

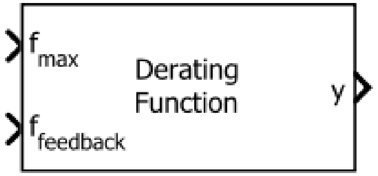


DQ Limiter

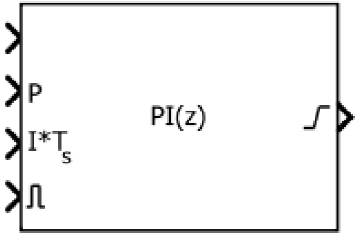


Position Generator

Controllers

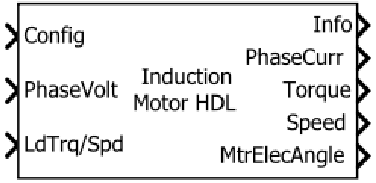


Derating Function

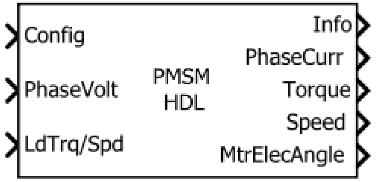


PI Controller

Motor Models

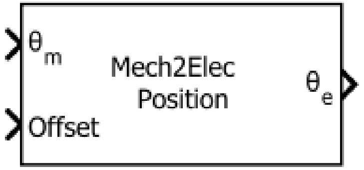


Induction Motor HDL

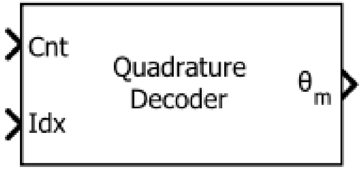


PMSM HDL

Sensor Decoders



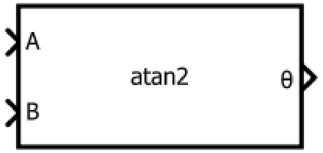
Mechanical to Electrical Position



Quadrature Decoder

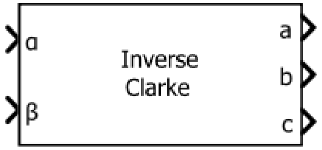
Speed Measurement

Math Transforms

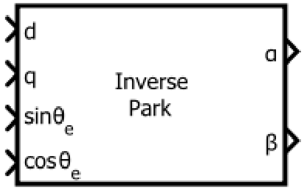


atan2

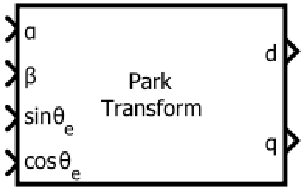
Clarke Transform



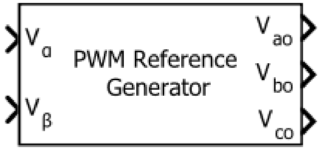
Inverse Clarke Transform



Inverse Park Transform

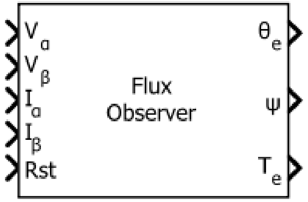


Park Transform



PWM Reference Generator

Sensorless Estimators



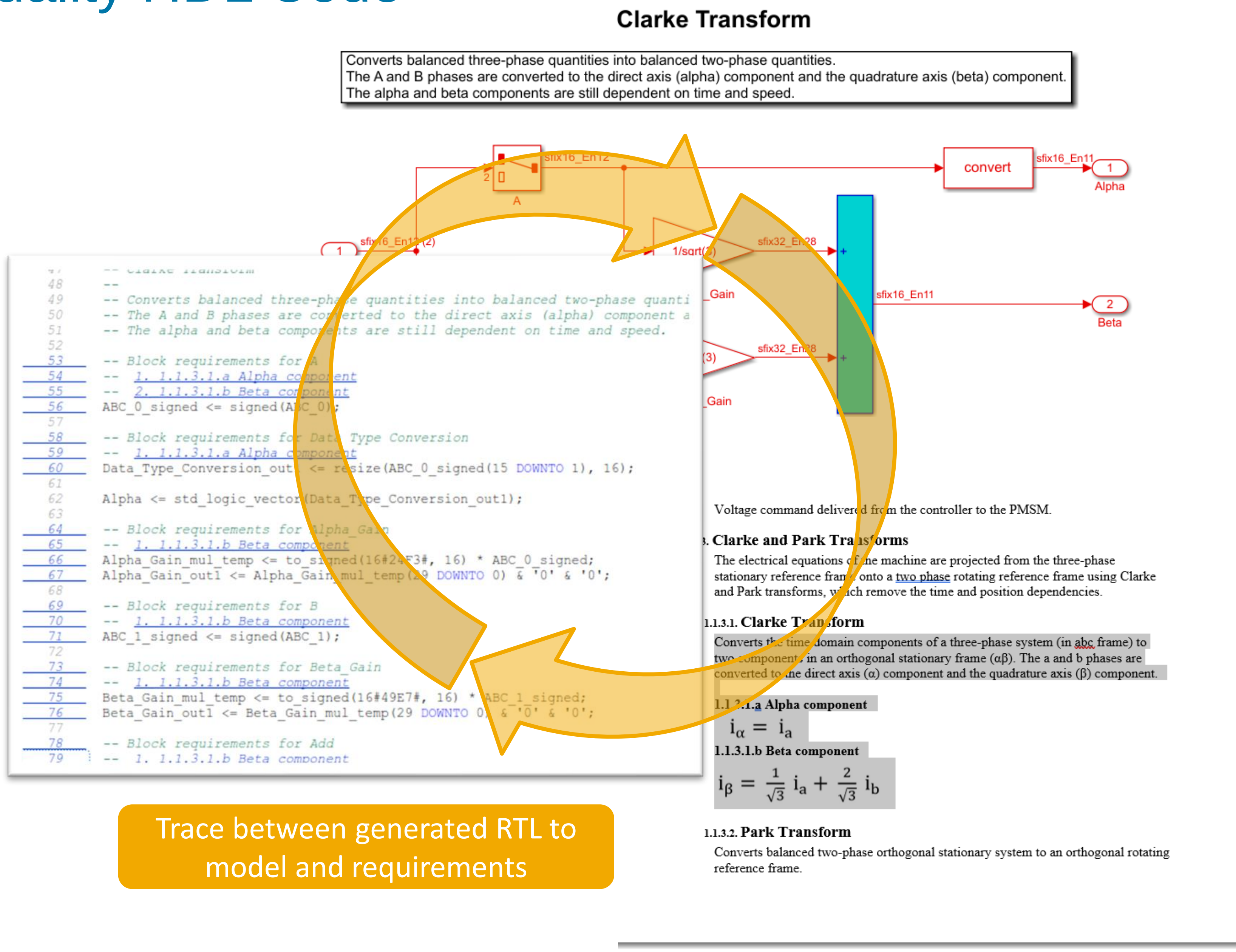
Flux Observer

Reference Application Example:

Sensorless FOC of PMSM Using FPGA-Based Motor Control Development Kit

Generate Production Quality HDL Code

- Portable VHDL or Verilog
- Bit and cycle accurate
- Readable, customizable,
- structured, commented
- Retains model hierarchy
- ports and signal naming
- Bidirectional traceability



Clarke Transform

Converts balanced three-phase quantities into balanced two-phase quantities.
The A and B phases are converted to the direct axis (alpha) component and the quadrature axis (beta) component.
The alpha and beta components are still dependent on time and speed.

```
-- Clarke Transform
--
-- Converts balanced three-phase quantities into balanced two-phase quanti
-- The A and B phases are converted to the direct axis (alpha) component a
-- The alpha and beta components are still dependent on time and speed.

-- Block requirements for A
--  1. 1.1.3.1.a Alpha component
--  2. 1.1.3.1.b Beta component
ABC_0_signed <= signed(ABC_0);

-- Block requirements for Data Type Conversion
--  1. 1.1.3.1.a Alpha component
Data_Type_Conversion_out1 <= resize(ABC_0_signed(15 DOWNTO 1), 16);

Alpha <= std_logic_vector(Data_Type_Conversion_out1);

-- Block requirements for Alpha_Gain
--  1. 1.1.3.1.b Beta component
Alpha_Gain_mul_temp <= to_signed(16#24F3#, 16) * ABC_0_signed;
Alpha_Gain_out1 <= Alpha_Gain_mul_temp(29 DOWNTO 0) & '0' & '0';

-- Block requirements for B
--  1. 1.1.3.1.b Beta component
ABC_1_signed <= signed(ABC_1);

-- Block requirements for Beta_Gain
--  1. 1.1.3.1.b Beta component
Beta_Gain_mul_temp <= to_signed(16#49E7#, 16) * ABC_1_signed;
Beta_Gain_out1 <= Beta_Gain_mul_temp(29 DOWNTO 0) & '0' & '0';

-- Block requirements for Add
--  1. 1.1.3.1.b Beta component
```

Voltage command delivered from the controller to the PMSM.

3. Clarke and Park Transforms

The electrical equations of the machine are projected from the three-phase stationary reference frame onto a two phase rotating reference frame using Clarke and Park transforms, which remove the time and position dependencies.

1.1.3.1. Clarke Transform

Converts the time domain components of a three-phase system (in abc frame) to two components in an orthogonal stationary frame (αβ). The a and b phases are converted to the direct axis (α) component and the quadrature axis (β) component.

1.1.3.1.a Alpha component

$$i_\alpha = i_a$$

1.1.3.1.b Beta component

$$i_\beta = \frac{1}{\sqrt{3}} i_a + \frac{2}{\sqrt{3}} i_b$$

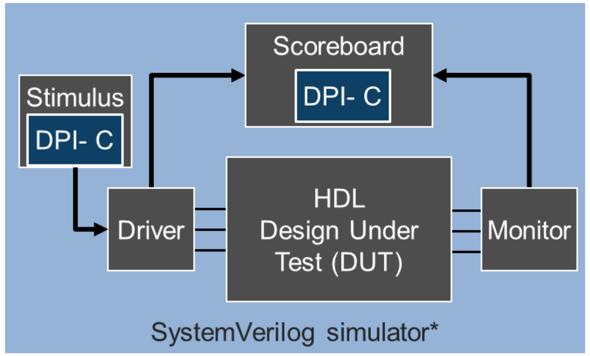
1.1.3.2. Park Transform

Converts balanced two-phase orthogonal stationary system to an orthogonal rotating reference frame.

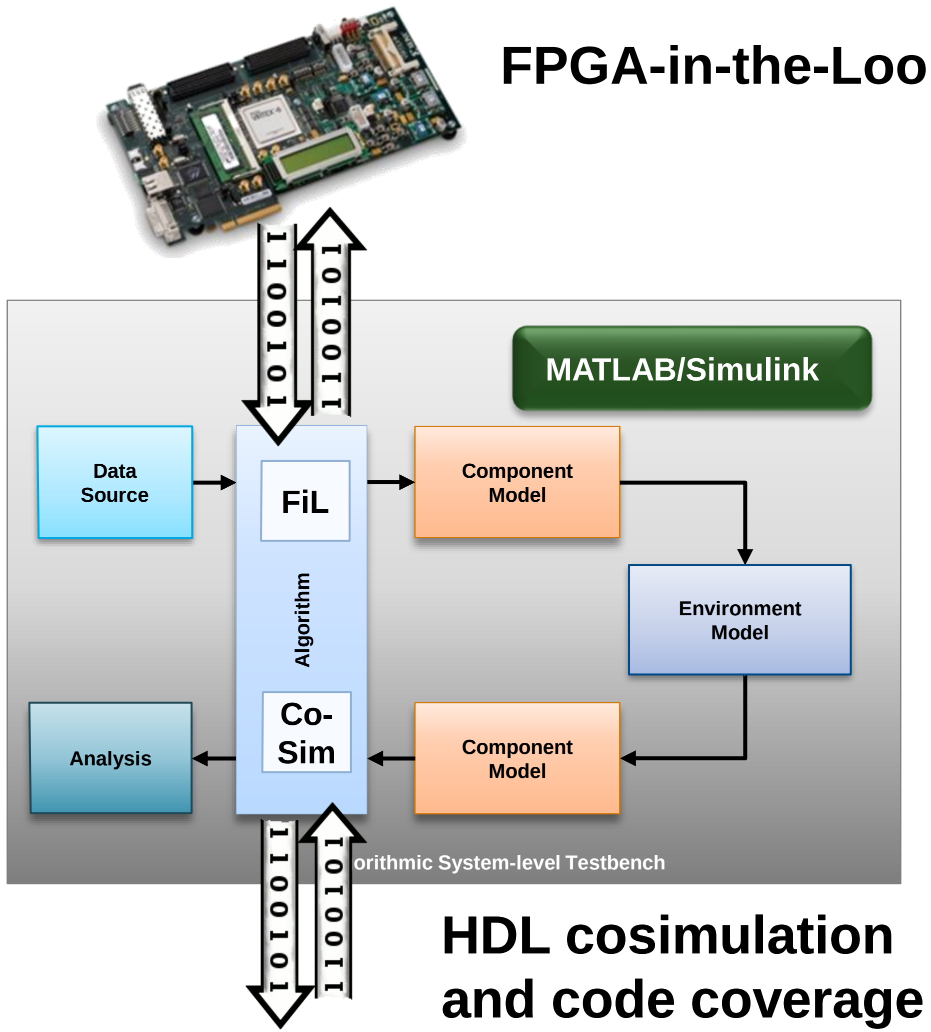
Trace between generated RTL to model and requirements

Verify RTL Automatically Using the Methodology of Choice

SystemVerilog UVM



Export components and testbenches

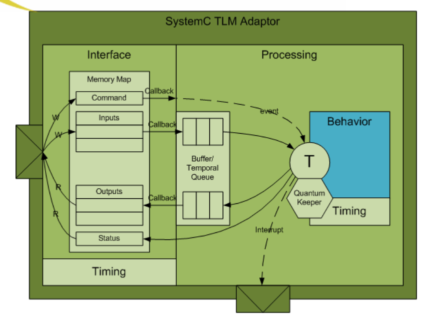


FPGA-in-the-Loop

HDL cosimulation and code coverage

Questa **Xcelium** **VIVADO**

SYSTEM C™ TLM



Export components and testbenches

HDL Coder for High-Integrity Applications

HDL Coder is qualified by TÜV SÜD for:

- **General**: IEC 61508
- **Automotive:** ISO 2626 any ASIL
- **Railway:** EN 50128 and EN 50657 any SIL
- **Agric. Machinery:** ISO25119 any SRL
- **Medical:** IEC 62304 any class

認證證書 ◆ СЕРТИФИКАТ ◆ CERTIFICADO ◆ CERTIFICAT

TÜV SÜD Product Service

CERTIFICATE

No. Z10 067052 0025 Rev. 00

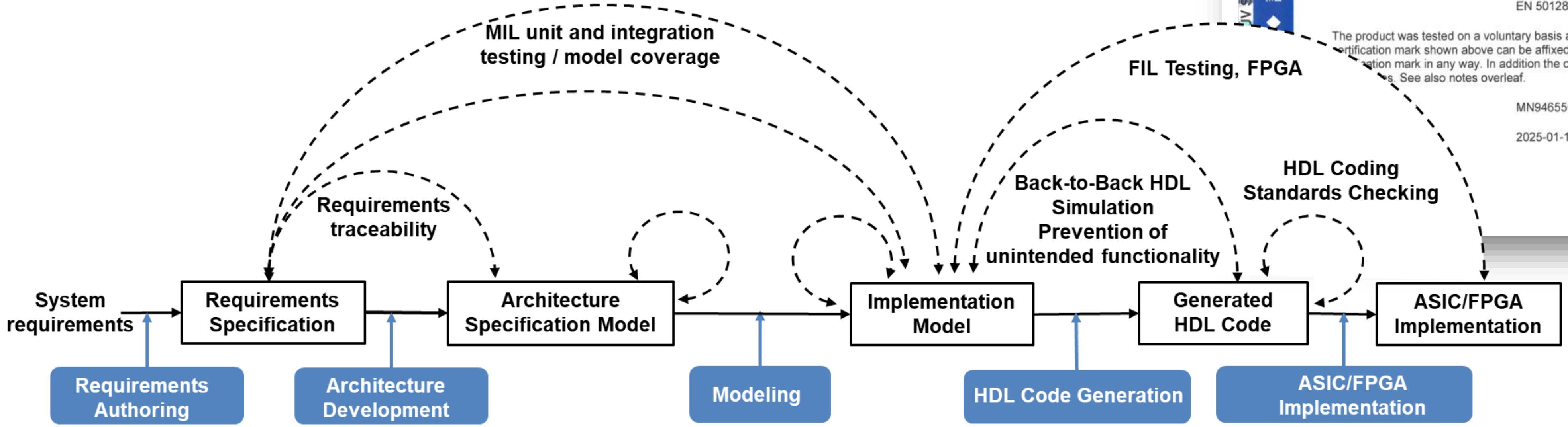
Holder of Certificate:	**The MathWorks, Inc.** 3 Apple Hill Drive Natick MA 01760-2098 USA
Factory(ies):	067052
Certification Mark:	
Product:	**Software Tool for Safety Related Development**
Model(s):	**HDL Coder™**
Parameters:	The code generator is suitable for use to develop safety-related software according to IEC 61508 and EN 50128 for any SIL and ISO 25119 for any SRL. The code generator is a qualified tool according to ISO 26262 for any ASIL. It is suitably validated for use in safety-related development according to IEC 62304. The report MN94655C is a mandatory part of this certificate.
Tested according to:	IEC 61508-3:2010 IEC 62304:2015 ISO 25119-3:2018 ISO 26262-8:2018 EN 50128:2011/AC:2014

The product was tested on a voluntary basis and complies with the essential requirements. The ertification mark shown above can be affixed on the product. It is not permitted to alter the ation mark in any way. In addition the certification holder must not transfer the certificate to s. See also notes overleaf.

MN94655C

2025-01-15

(Peter Weiß)



Where are you on the Model-Based Design Adoption Grid?

	Modeling and Simulation	***Real-Time Simulation and Testing***	***Production Code Development***
Systematic Testing of Algorithms	System Verification	Automated Real-Time Testing	Certification
Simulating Algorithms with System Models	System Simulation	Hardware-in-the-Loop (HIL) Simulation	SIL, PIL, CoSim, FIL
Developing Algorithms	Algorithm Modeling	Rapid Prototyping	Embedded Code Generation

Model Usage

Code Generation

HDL Coder Integrated Workflows with 3rd Party Hardware

National Instruments

HDL Coder Support Package for NI FPGA Hardware

Version 23.3.0.0 (1.75 MB) by NI Team HIL

Generates bitfiles for NI FPGA hardware

https://github.com/ni/hdlcoder-support-package-for-nifpga-hardware

★★★★★ (8)

140 Downloads

Updated 18 Apr 2023

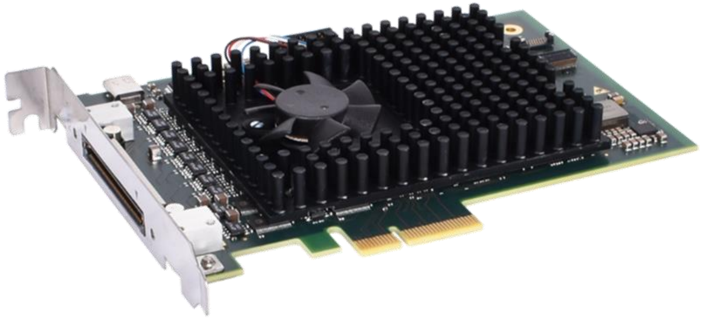
From GitHub

View License on GitHub

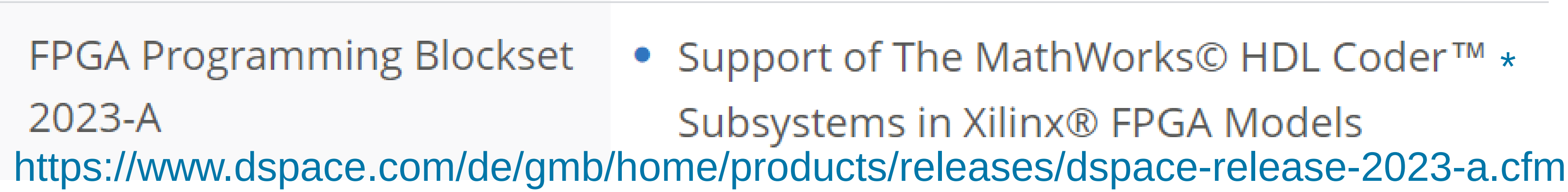
speedgoat

real-time simulation and testing

Any FPGA



dSPACE



FPGA Programming Blockset 2023-A	- Support of The MathWorks© HDL Coder™ * Subsystems in Xilinx® FPGA Models

https://www.dspace.com/de/gmb/home/products/releases/dspace-release-2023-a.cfm

Summary

- FPGAs are playing a key role in power electronics control design
- Model-Based Design is the leading methodology in this domain
- It enables real-time simulation of power electronics on FPGAs
- Deployment of complex control algorithms on FPGAs
- Facilitates the transition to other targets such as ASIC or MCUs
- Integrated workflows for hardware platforms and HiLs

Questions?

Dimitri Hamidi
dhamidi@mathworks.com

MathWorks
AUTOMOTIVE CONFERENCE 2023
Europe

Thank you