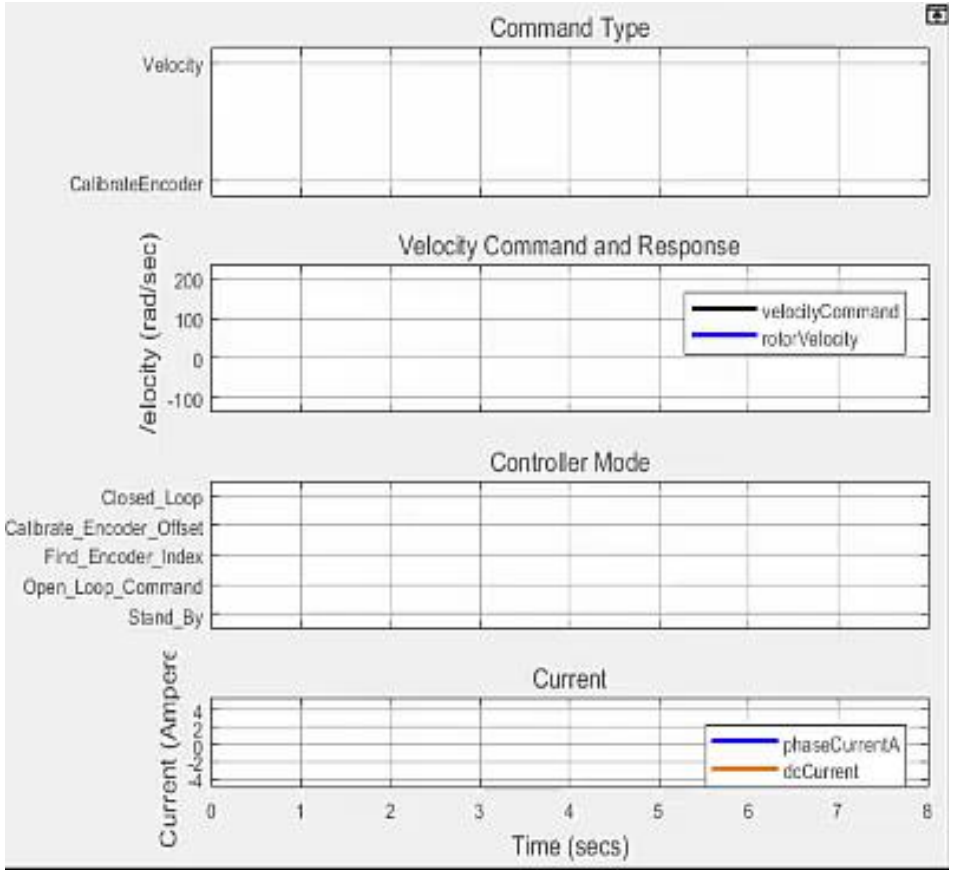
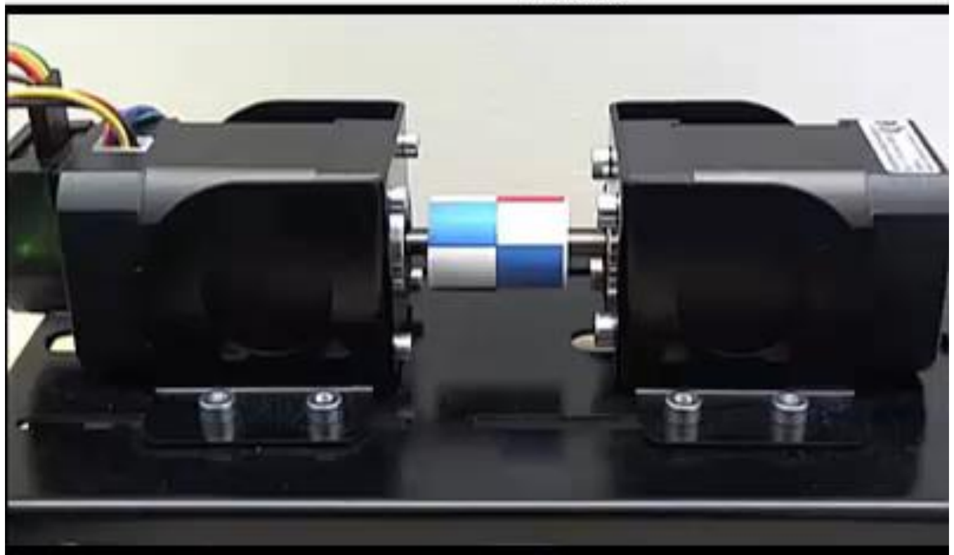
MATLAB EXPO 2018

Hardware and Software Co-Design for Motor Control Applications





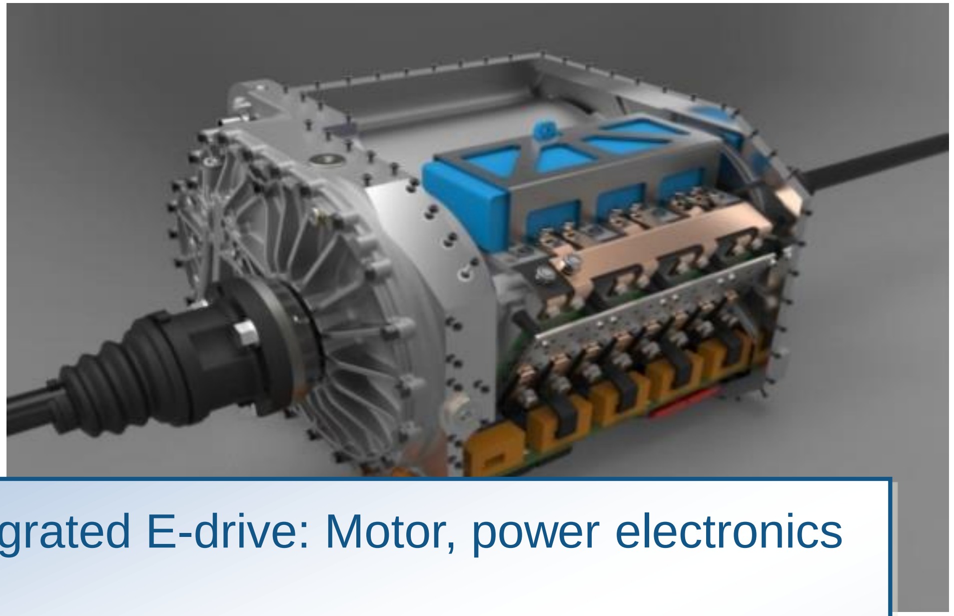
Takeaways

Model-Based Design for SoC FPGAs

- Enables early validation of specifications using simulation
- Improves design team collaboration and designer productivity.
- Reduces hardware testing time by 5x

Punch Powertrain develops complex SoC-based motor control

- Powertrains for hybrid and electric vehicles
- Need to increase power density and efficiency at a reduced cost
  - Integrate motor and power electronics in the transmission
- New switched reluctance motor
  - Fast: 2x the speed of their previous motor
    - Target to a Xilinx® Zynq® SoC 7045 device
  - Complex: 4 different control strategies
- No experience designing FPGAs!



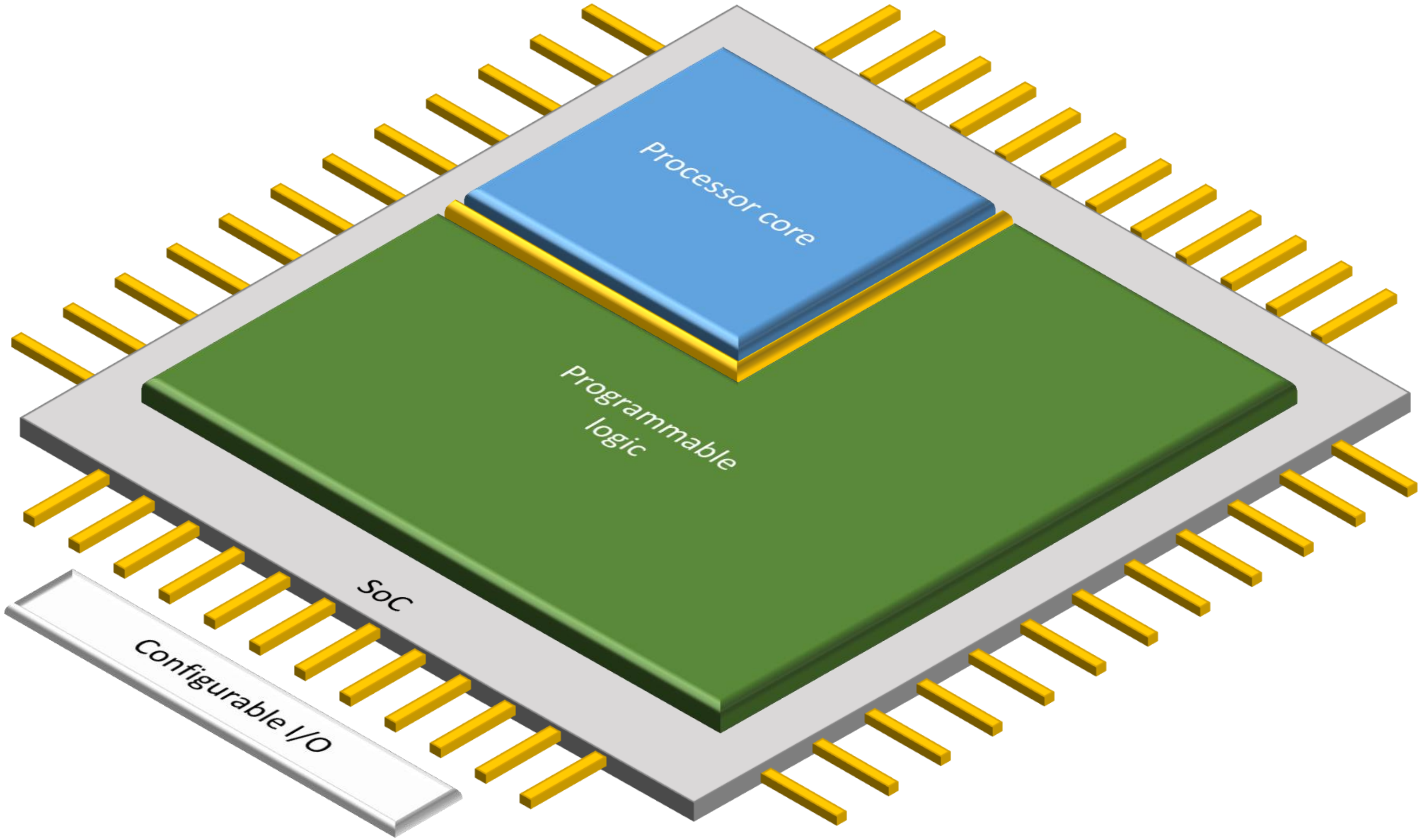
- ✓ Designed integrated E-drive: Motor, power electronics and software
- ✓ 4 different control strategies implemented
- ✓ Completed in 1.5 years with 2FTE's
- ✓ Models reusable for production
- ✓ Smooth integration and validation due to development process – thorough validation before electronics are produced and put in the testbench

Link to video of presentation

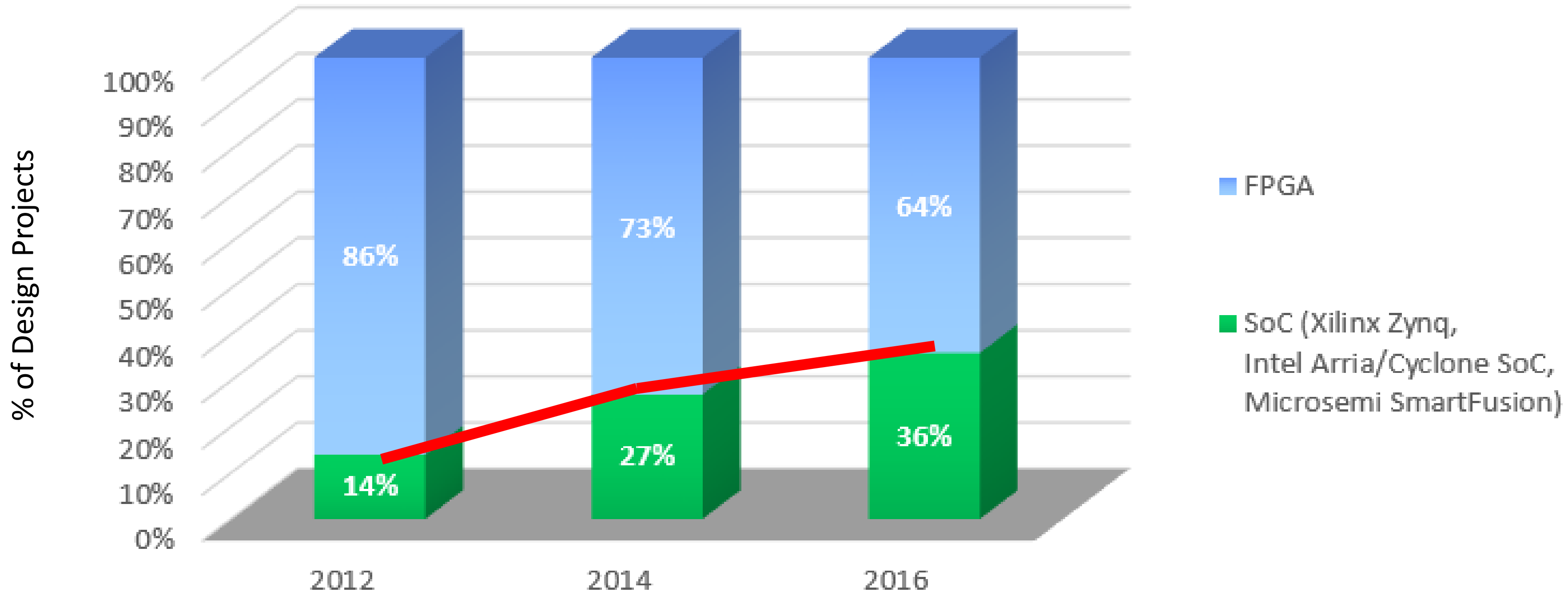
Key trend: Increasing demands from motor drives

- Advanced algorithms require faster computing performance.
  - Field-Oriented Control
  - Sensorless motor control
  - Vibration detection and suppression
  - Multi-axis control

What's an SoC?



Key Trend: SoCs are now used in 36% of new FPGA projects



Source: Wilson Research Group and Mentor Graphics,
2016 Functional Verification Study

Challenges in using SoCs for Motor and Power Control

- Integration of software and hardware partitions of algorithm on SoC drives need for collaboration

- Validation of design specifications with limits on access to motors in labs.

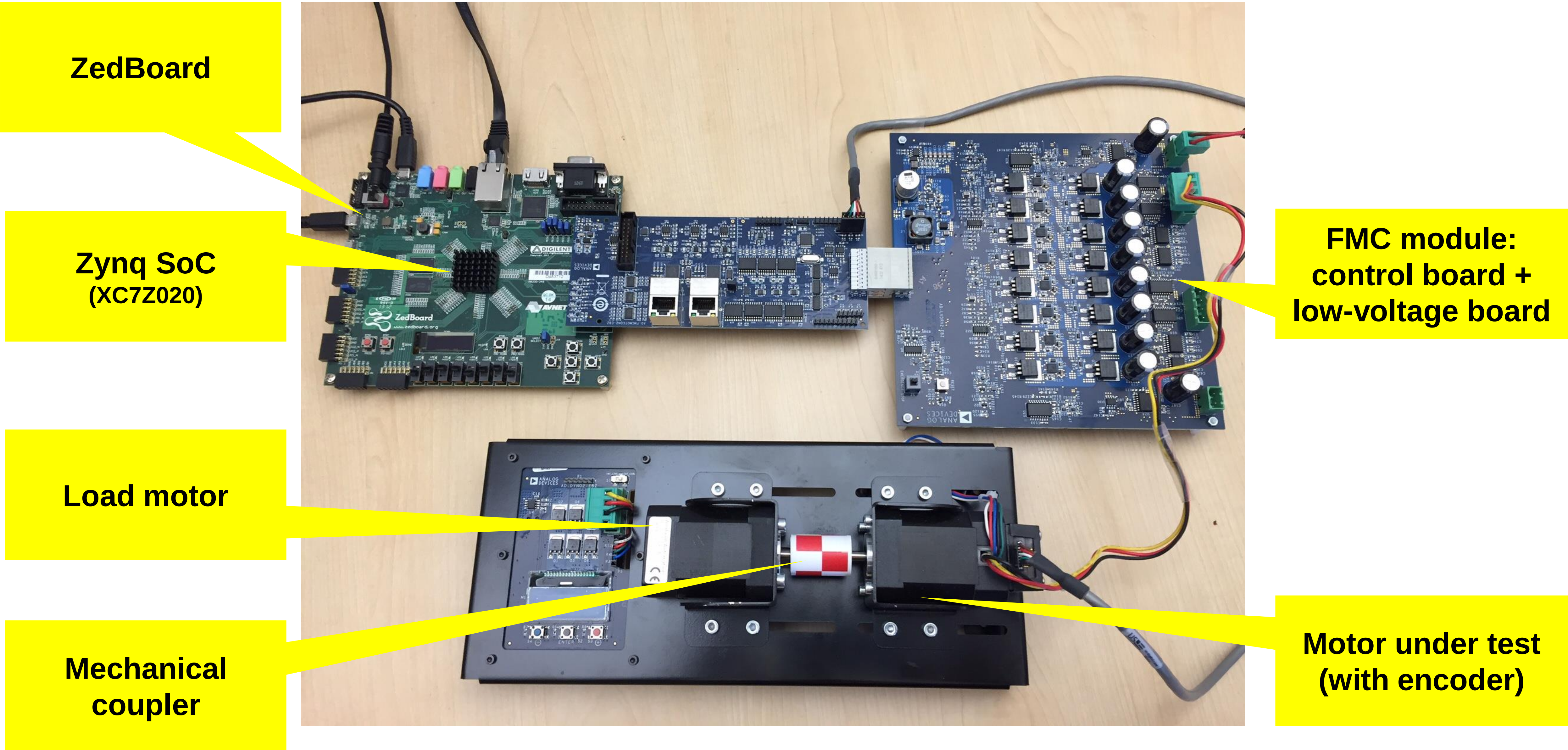
- How to make design decisions that cut across system components?

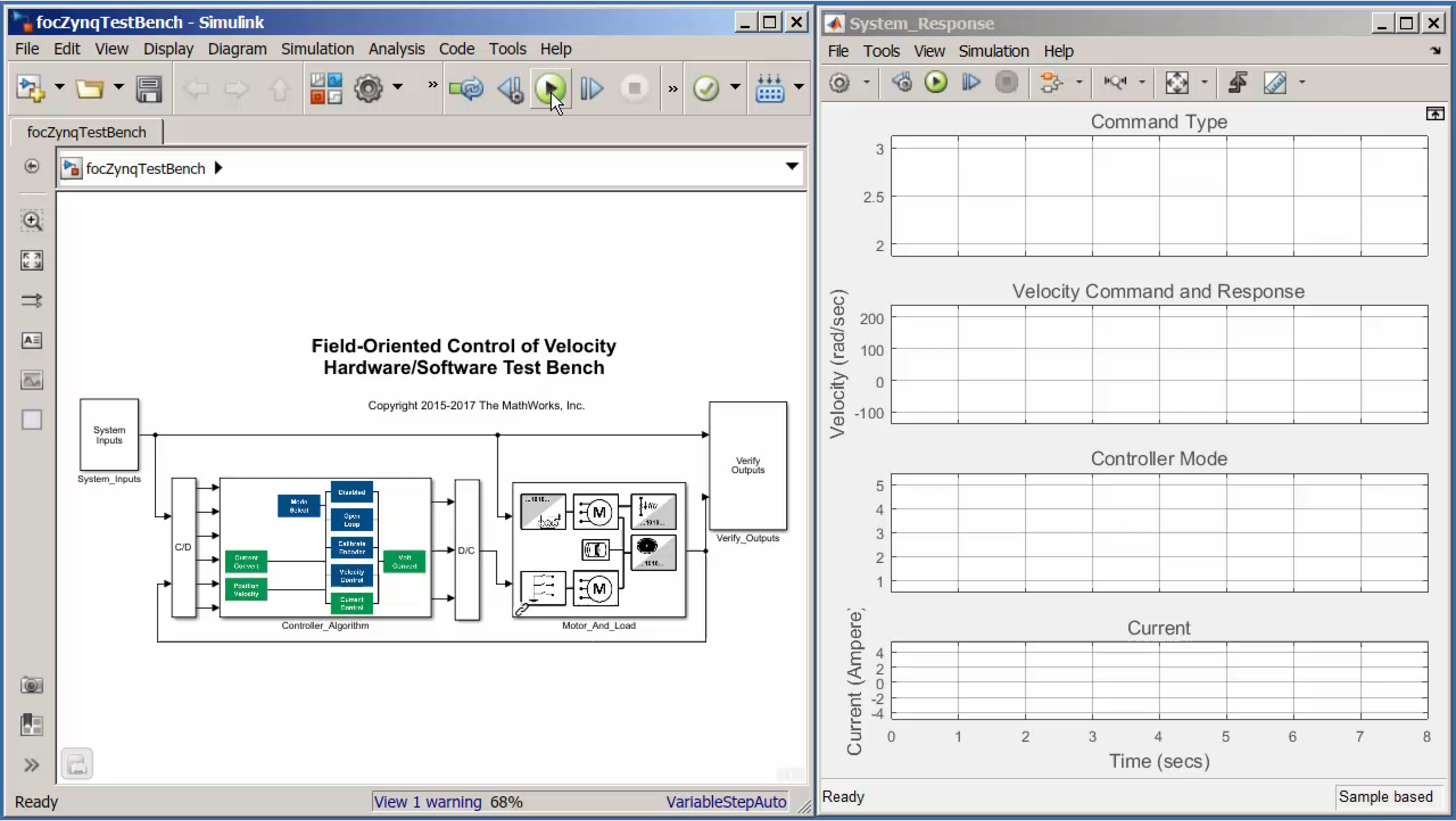
Why use Model-Based Design to develop motor control applications on SoCs?

- Enables early validation of specifications using simulation months before hardware is available.

- Improves design team collaboration and designer productivity by using a shared design environment.

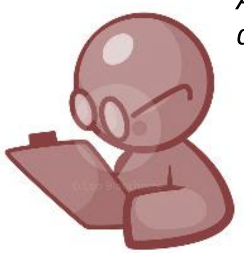
- Reduces hardware testing time by 5x by shifting design from lab to the desktop


ZedBoard
Zynq SoC
(XC7Z020)
Load motor
Mechanical coupler
FMC module: control board + low-voltage board
Motor under test (with encoder)
ZedBoard

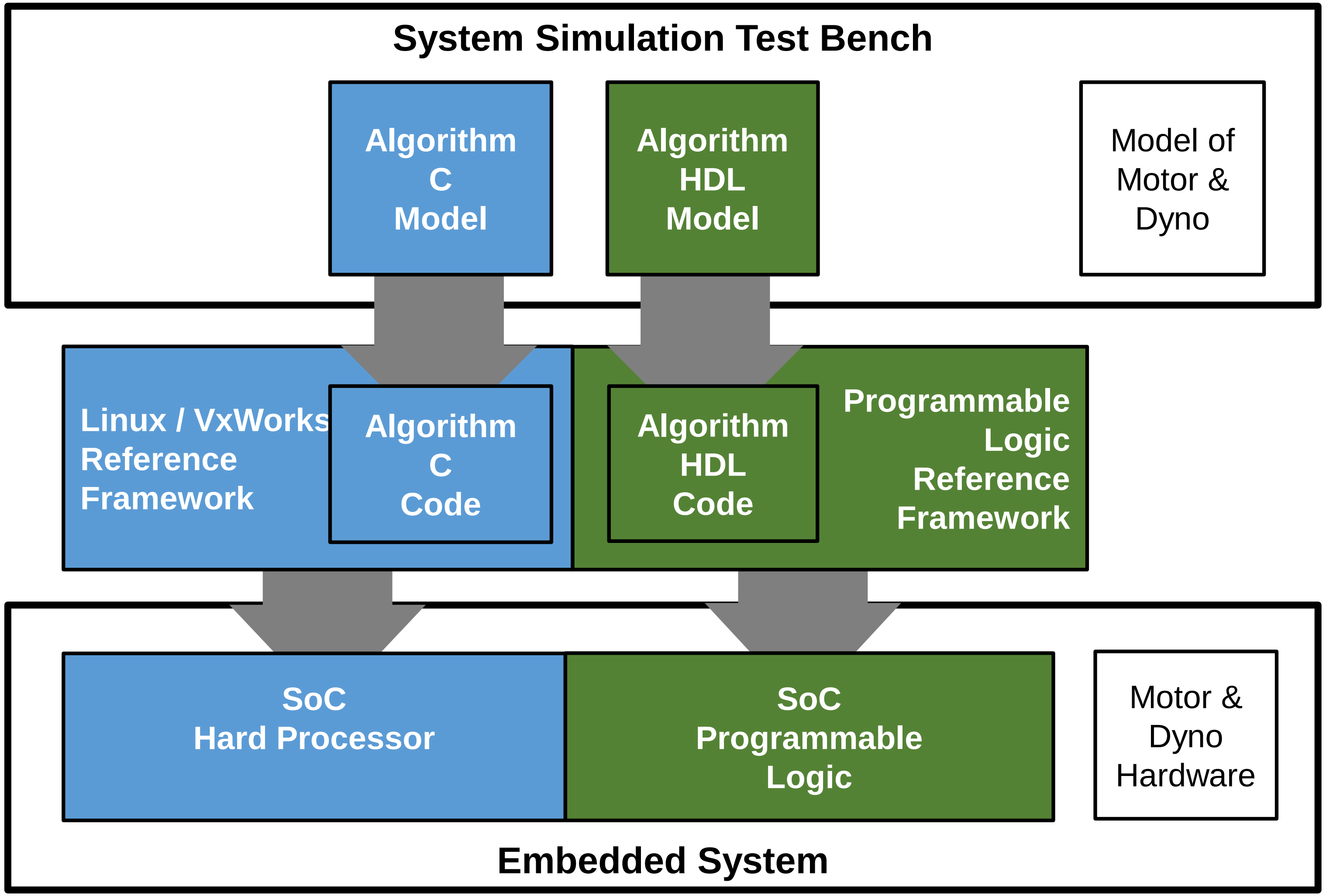

focZynqTestBench - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
focZynqTestBench
focZynqTestBench
Field-Oriented Control of Velocity
Hardware/Software Test Bench
Copyright 2015-2017 The MathWorks, Inc.
System Inputs
System_Inputs
C/D
Controller_Algorithm
D/C
Motor_And_Load
Verify Outputs
Verify_Outputs
Ready
View 1 warning 68%
VariableStepAuto
System_Response
File Tools View Simulation Help
Command Type
Velocity Command and Response
Velocity (rad/sec)
Controller Mode
Current
Current (Ampere)
Time (secs)
Ready
Sample based

Conceptual workflow targeting SoCs

Algorithm developer

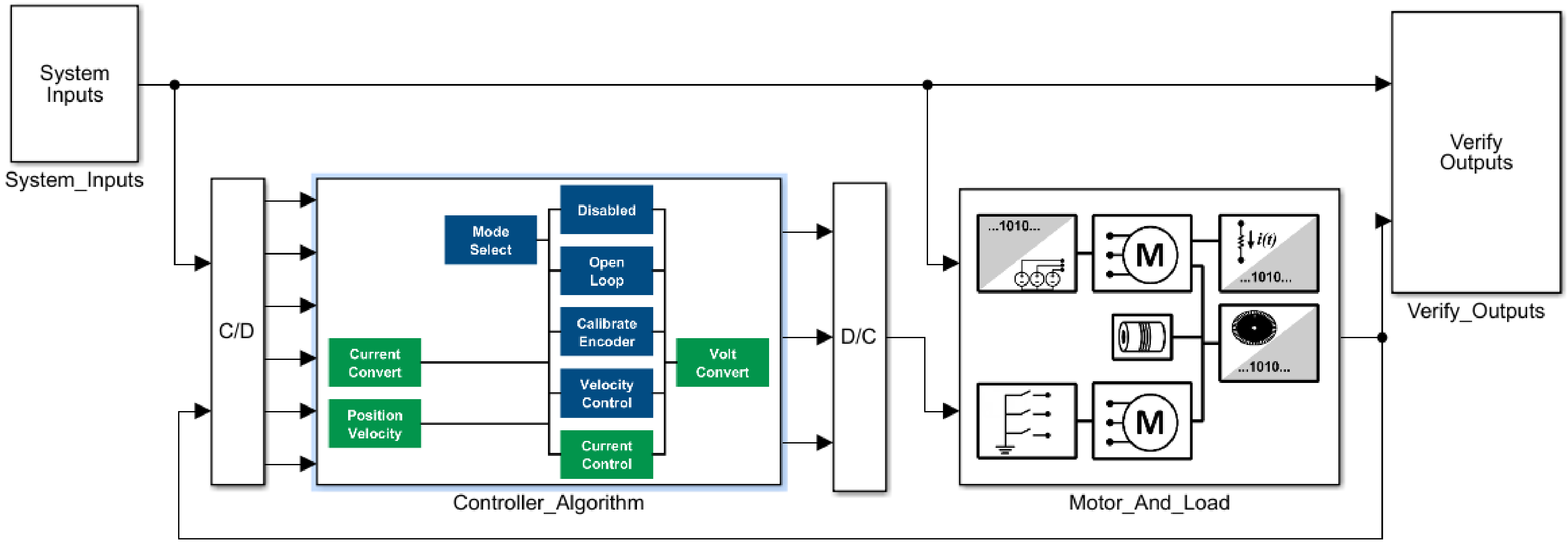


Embedded software engineer

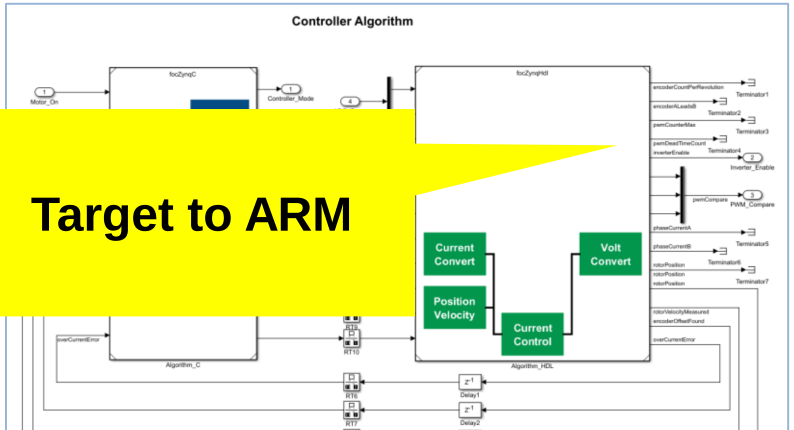


Hardware designer

Hardware/software partitioning

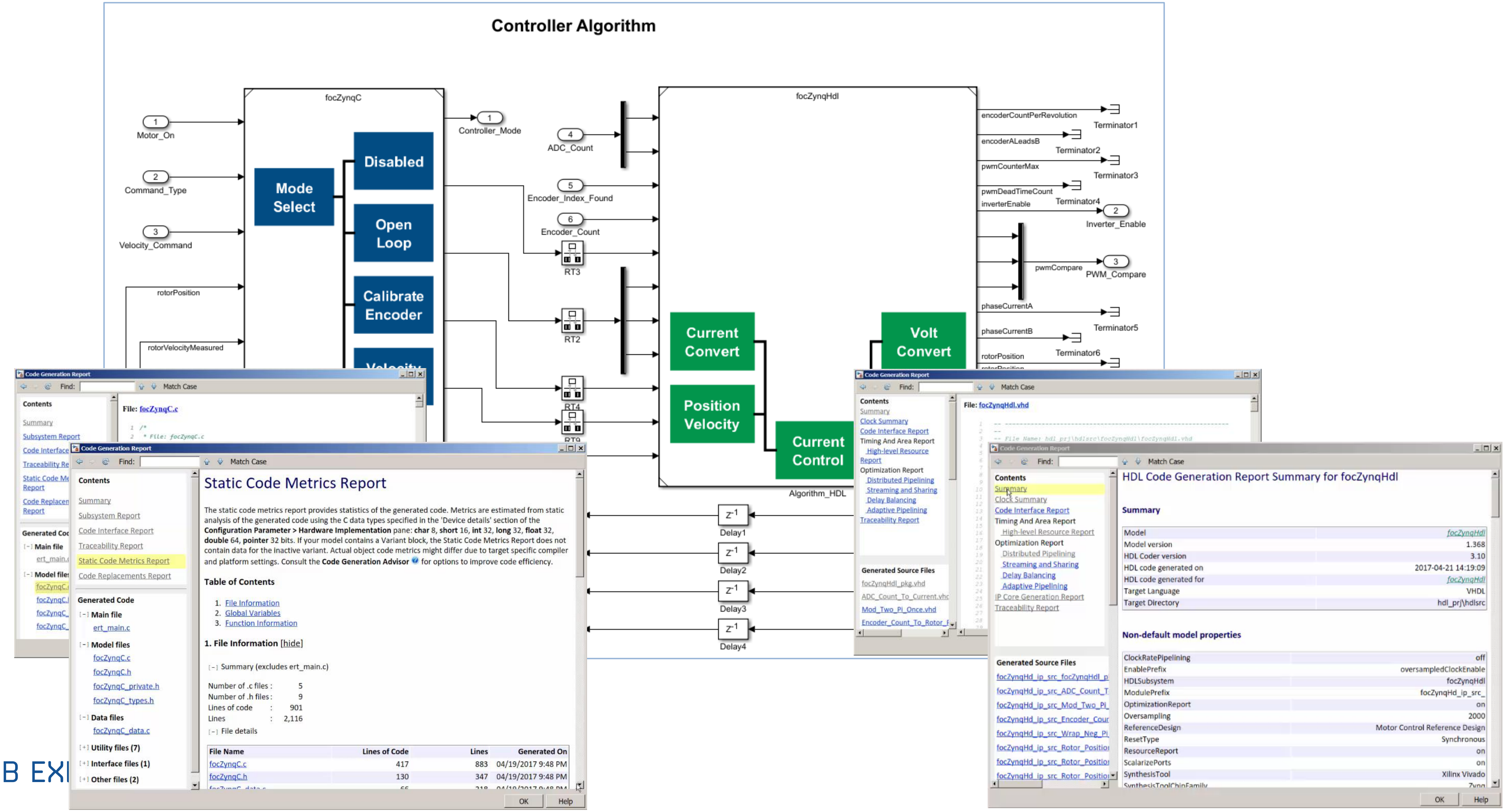


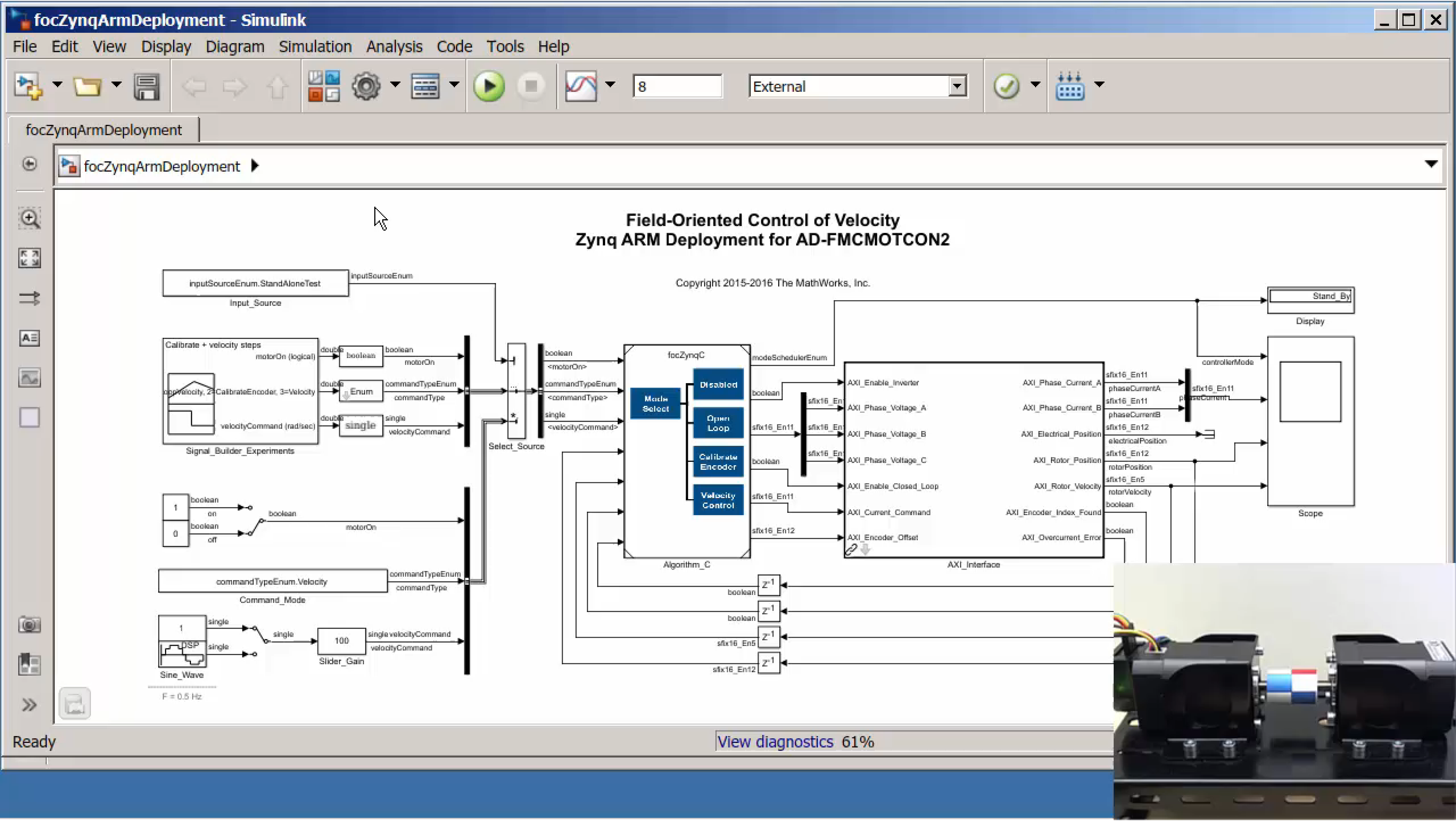
Target to Programmable Logic

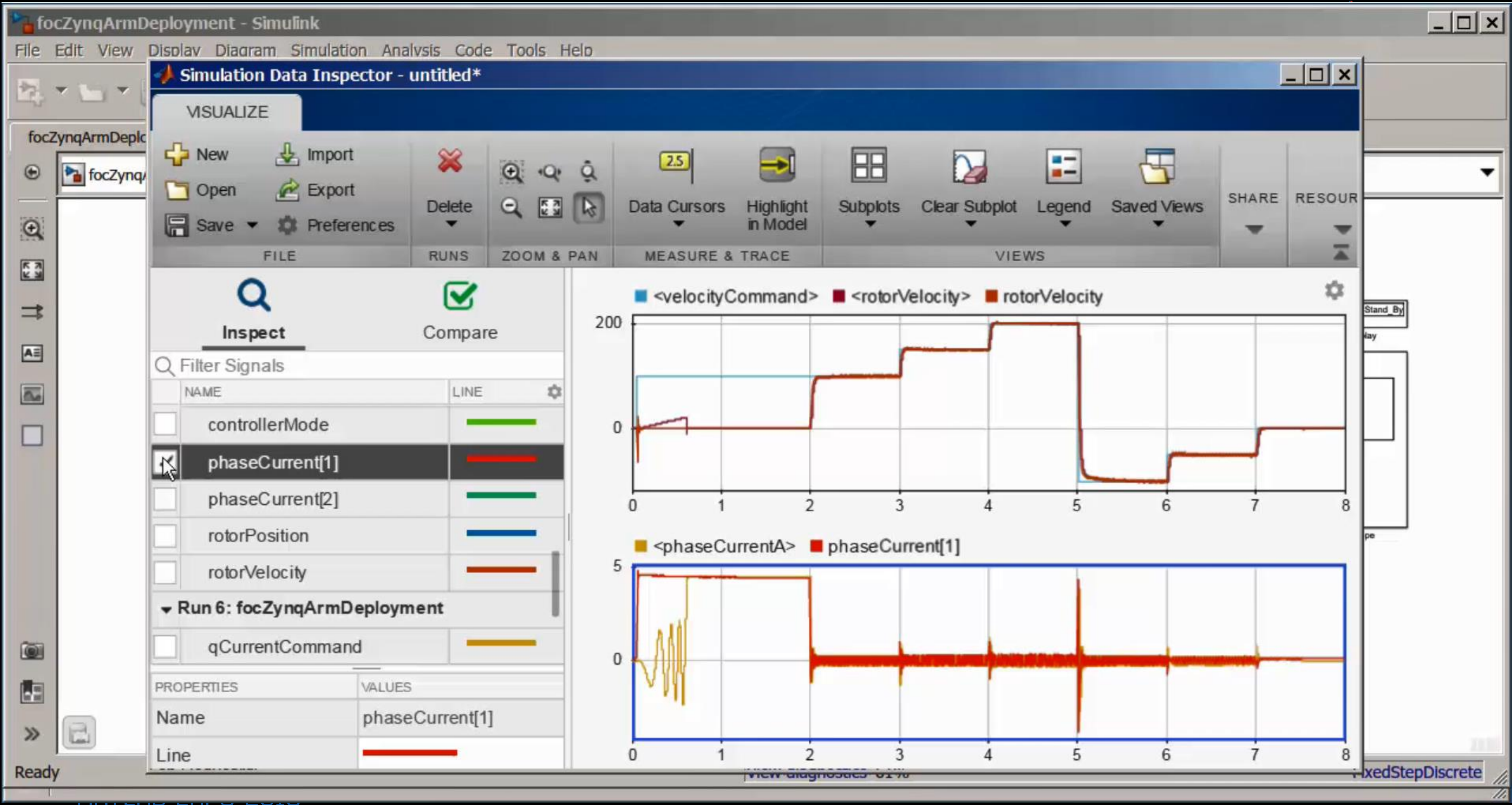


Target to ARM

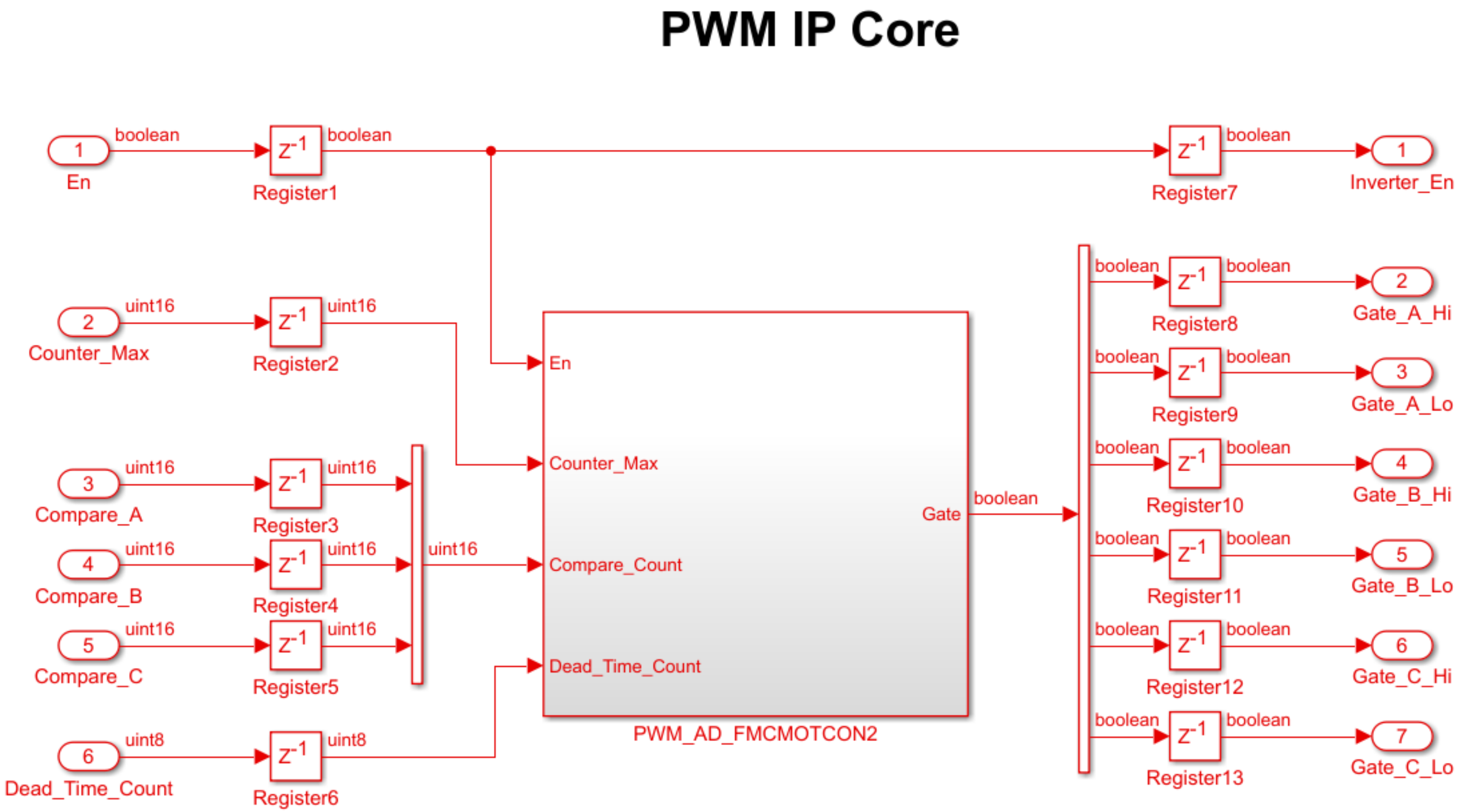
Code Generation

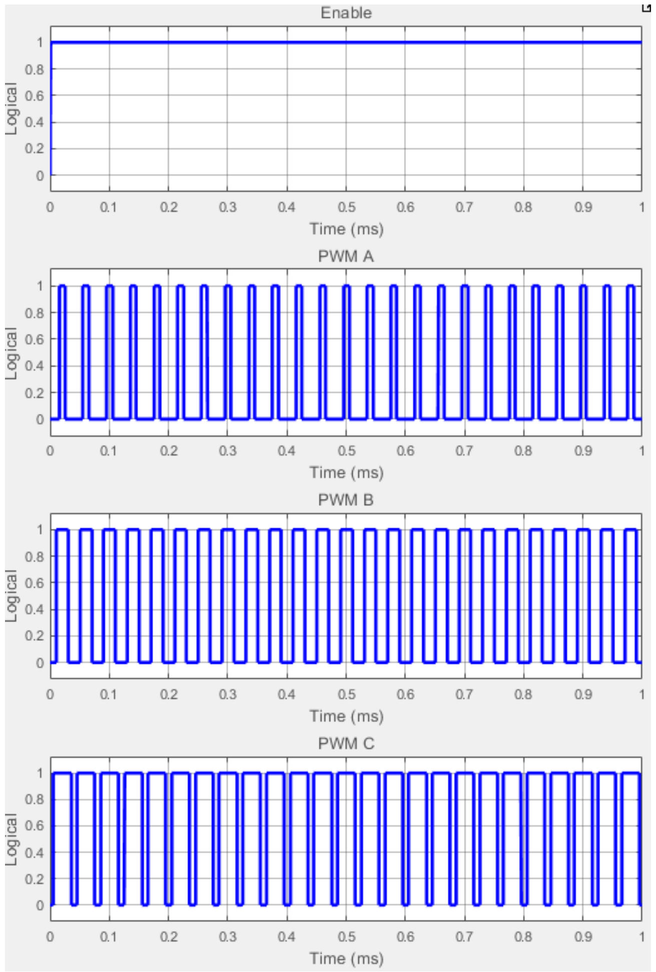



focZynqArmDeployment - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
8
External
focZynqArmDeployment
Field-Oriented Control of Velocity
Zynq ARM Deployment for AD-FMCMOTCON2
Copyright 2015-2016 The MathWorks, Inc.
inputSourceEnum.StandAloneTest
Input_Source
Calibrate + velocity steps
Signal_Builder_Experiments
Select_Source
focZynqC
Mode Select
Disabled
Open Loop
Calibrate Encoder
Velocity Control
Algorithm_C
AXI_Interface
Display
Stand_By
Scope
commandTypeEnum.Velocity
Command_Mode
Sine_Wave
F = 0.5 Hz
Slider_Gain
Ready
View diagnostics 61%


focZynqArmDeployment - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
Simulation Data Inspector - untitled*
VISUALIZE
New
Import
Open
Export
Save
Preferences
FILE
Delete
RUNS
ZOOM & PAN
Data Cursors
Highlight in Model
MEASURE & TRACE
Subplots
Clear Subplot
Legend
Saved Views
VIEWS
SHARE
Inspect
Compare
Filter Signals
NAME
LINE
controllerMode
phaseCurrent[1]
phaseCurrent[2]
rotorPosition
rotorVelocity
Run 6: focZynqArmDeployment
qCurrentCommand
PROPERTIES
VALUES
Name
phaseCurrent[1]
Line
<velocityCommand>
<rotorVelocity>
rotorVelocity
<phaseCurrentA>
phaseCurrent[1]
Ready

How could I check PWM signals in case of an issue?

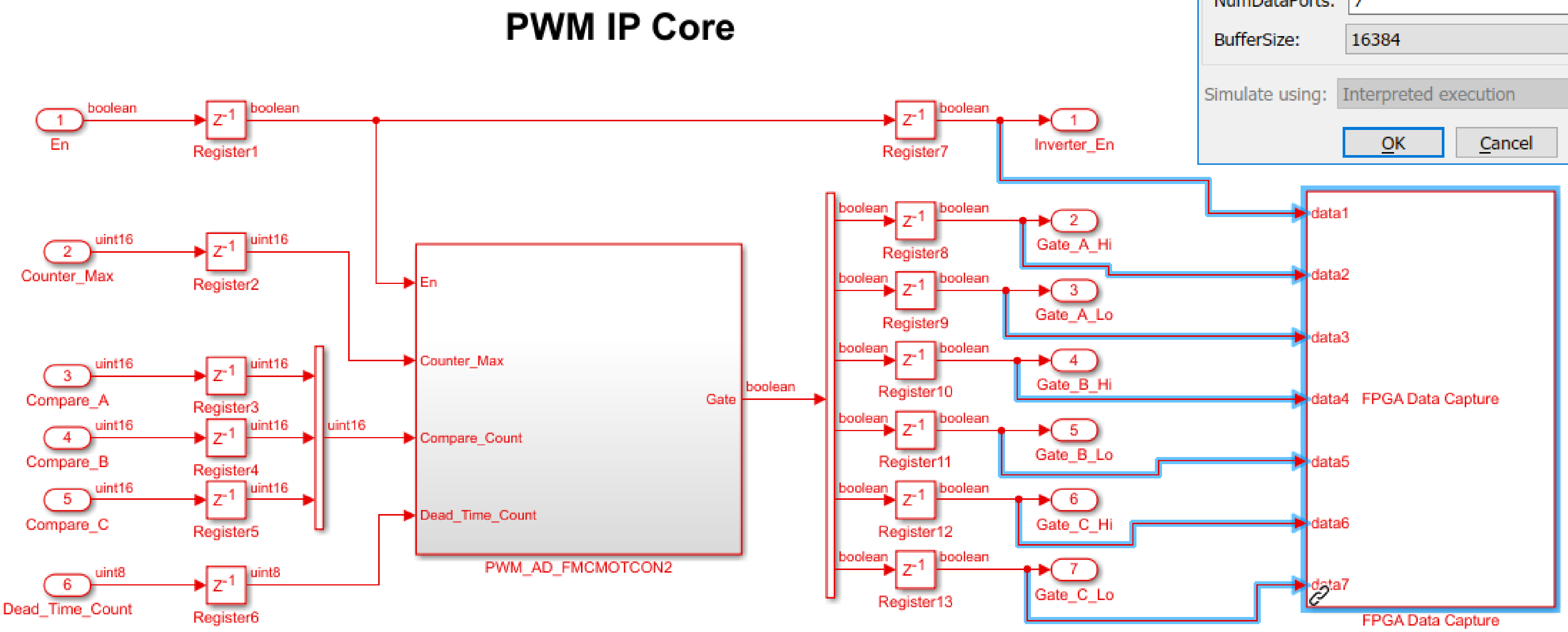


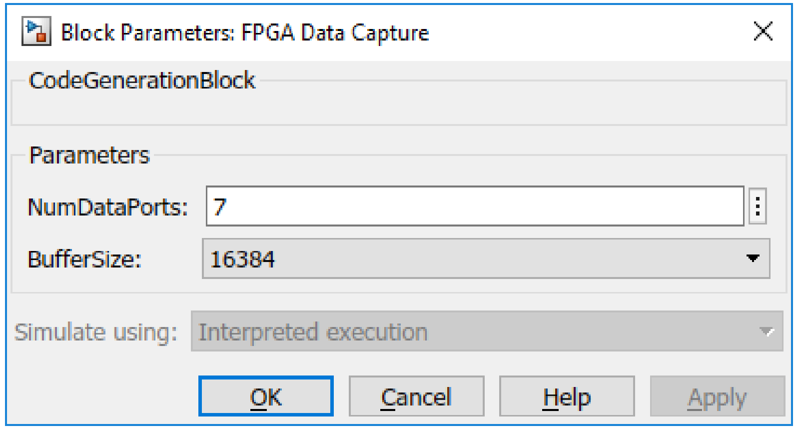


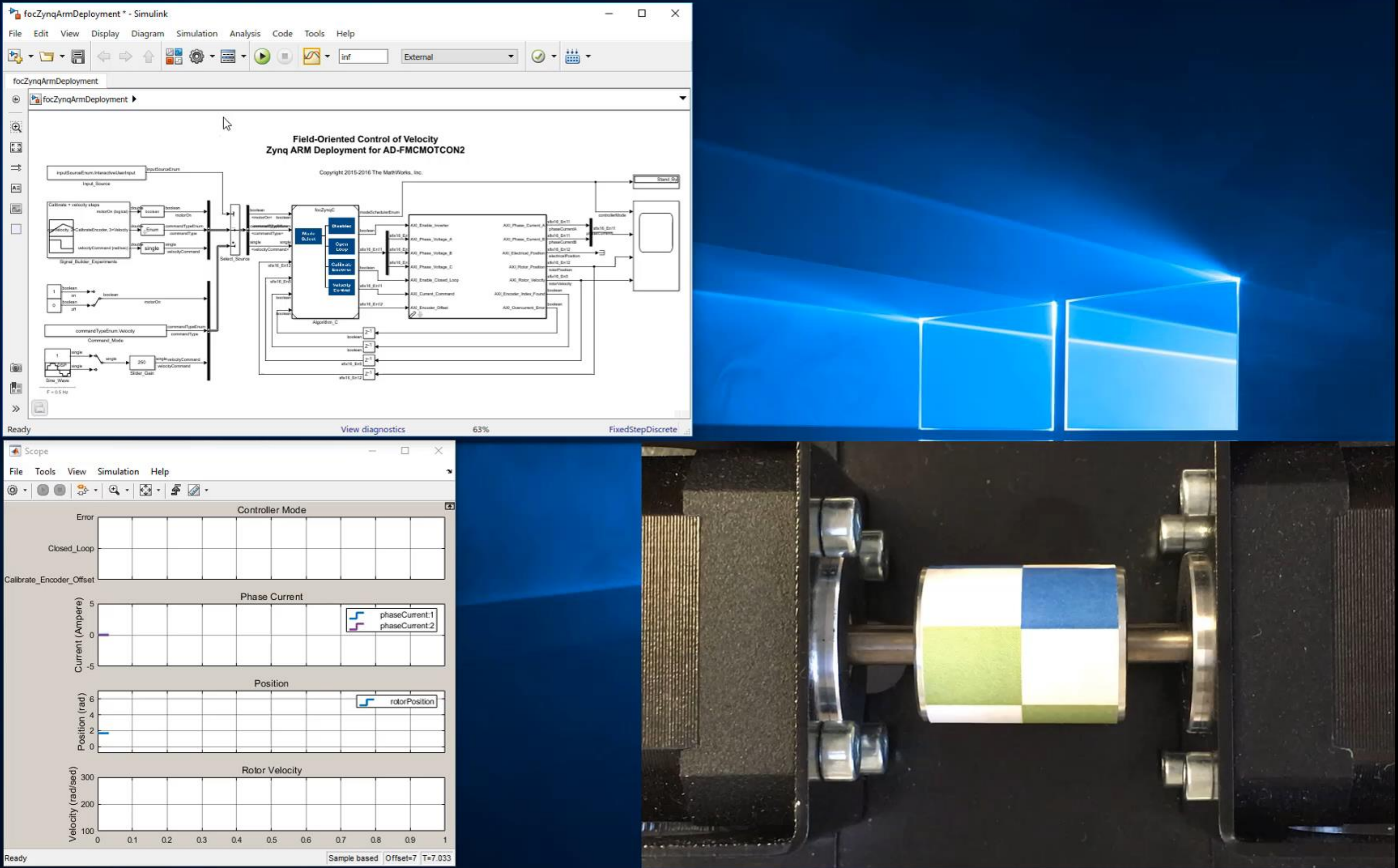
PWM rate: 25 kHz

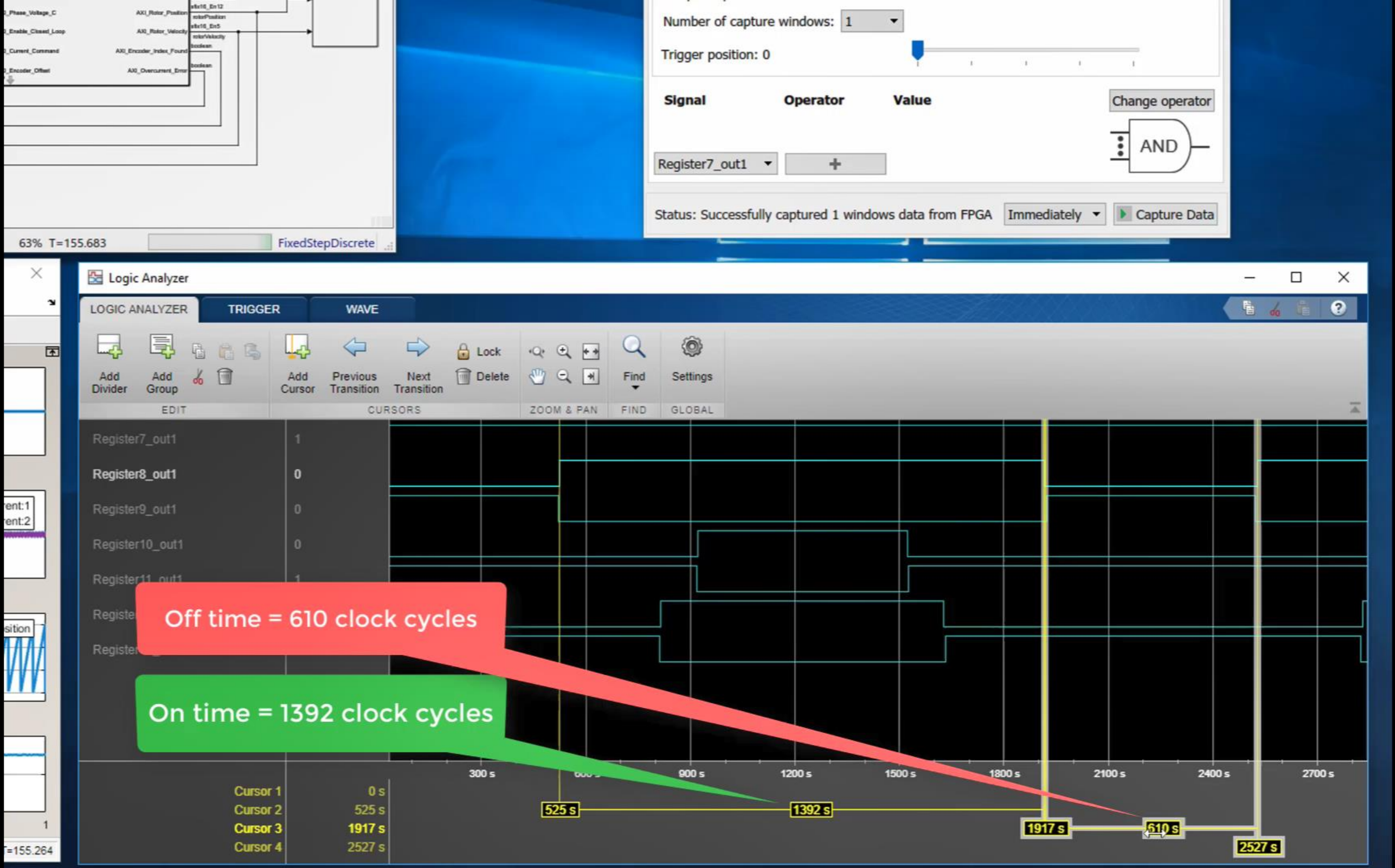
Clock rate: 50 MHz

Add FPGA Data Capture to record 50 MHz outputs from PWMs






focZynqArmDeployment * - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
inf
External
focZynqArmDeployment
Field-Oriented Control of Velocity
Zynq ARM Deployment for AD-FMCMOTCON2
Copyright 2015-2016 The MathWorks, Inc.
Ready
View diagnostics
63%
FixedStepDiscrete
Scope
File Tools View Simulation Help
Controller Mode
Error
Closed_Loop
Calibrate_Encoder_Offset
Phase Current
Current (Ampere)
phaseCurrent:1
phaseCurrent:2
Position
Position (rad)
rotorPosition
Rotor Velocity
Velocity (rad/sec)
Ready
Sample based
Offset=7
T=7.033


Number of capture windows: 1
Trigger position: 0
Signal
Operator
Value
Change operator
AND
Register7_out1
Status: Successfully captured 1 windows data from FPGA
Immediately
Capture Data
63% T=155.683
FixedStepDiscrete
Logic Analyzer
LOGIC ANALYZER
TRIGGER
WAVE
Add Divider
Add Group
Add Cursor
Previous Transition
Next Transition
Lock
Delete
Find
Settings
EDIT
CURSORS
ZOOM & PAN
FIND
GLOBAL
Register7_out1 1
Register8_out1 0
Register9_out1 0
Register10_out1 0
Off time = 610 clock cycles
On time = 1392 clock cycles
300 s
900 s
1200 s
1500 s
1800 s
2100 s
2400 s
2700 s
Cursor 1 0 s
Cursor 2 525 s
Cursor 3 1917 s
Cursor 4 2527 s
525 s
1392 s
1917 s
610 s
2527 s
T=155.264

3T Develops Robot Emergency Braking System with Model-Based Design

A SCARA robot.

Challenge

Design and implement a robot emergency braking system with minimal hardware testing

Solution

Model-Based Design with Simulink and HDL Coder to model, verify, and implement the controller

Results

- Cleanroom time reduced from weeks to days
- Late requirement changes rapidly implemented
- Complex bug resolved in one day

"With Simulink and HDL Coder we eliminated programming errors and automated delay balancing, pipelining, and other tedious and error-prone tasks. As a result, we were able to easily and quickly implement change requests from our customer and reduce time-to-market."

Ronald van der Meer
3T

Link to user story

Why use Model-Based Design to develop motor control applications on SoCs?

Challenges:

- Integration of software and hardware partitions of algorithm on SoC drives need for collaboration
- Validation of design specifications with limits on access to motors in labs.
- How to make design decisions that cut across system?

Model-Based Design

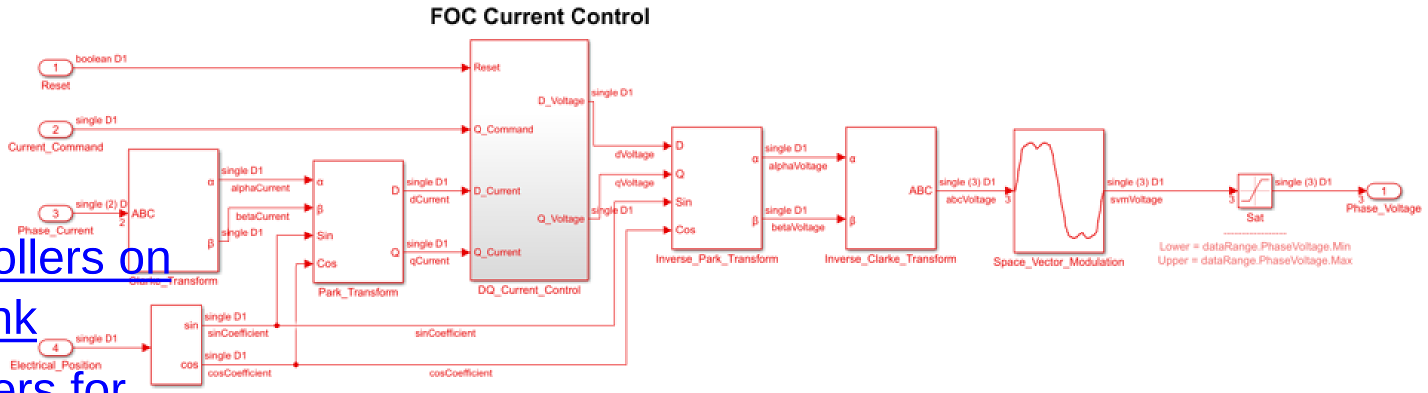
✓ Enables early validation of specifications using simulation months before hardware is available.

✓ Improves design team collaboration and designer productivity by using a shared design environment.

✓ Reduces hardware testing time by 5x by shifting design from lab to the desktop

Learn More

- Visit us in the Technology Showcase - see Native Floating Point
- Videos
  - HDL Coder: Native Floating Point
- Webinars
  - Prototyping SoC-based Motor Controllers on Intel SoCs with MATLAB and Simulink
  - How to Build Custom Motor Controllers for Zynq SoCs with MATLAB and Simulink



- Articles
  - How Modeling Helps Embedded Engineers Develop Applications for SoCs (MATLAB Digest)
  - MATLAB and Simulink Aid HW-SW Codesign of Zynq SoCs (Xcell Software Journal)
- Tutorials:
  - Custom Reference Design Workflow for HDL Coder
  - Field-Oriented Control of a Permanent Magnet Synchronous Machine on SoCs