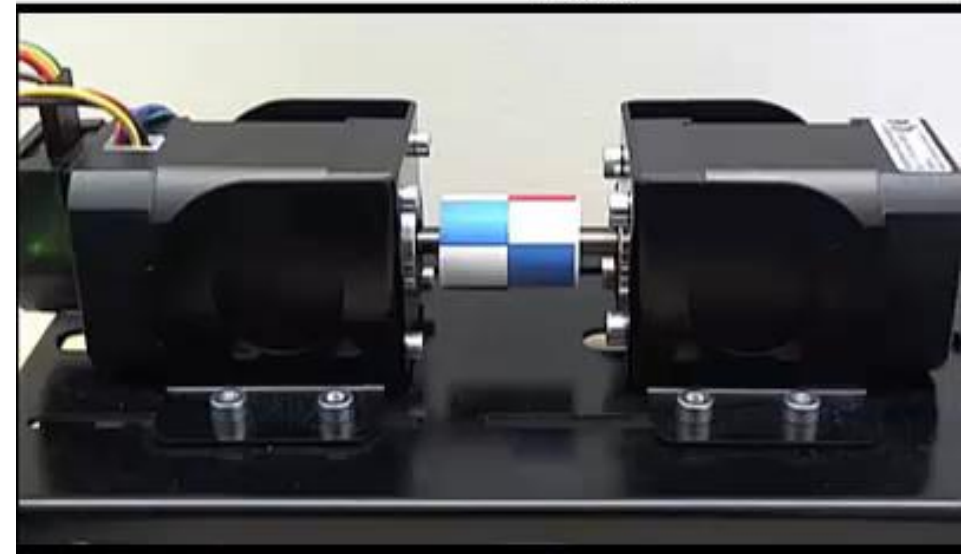
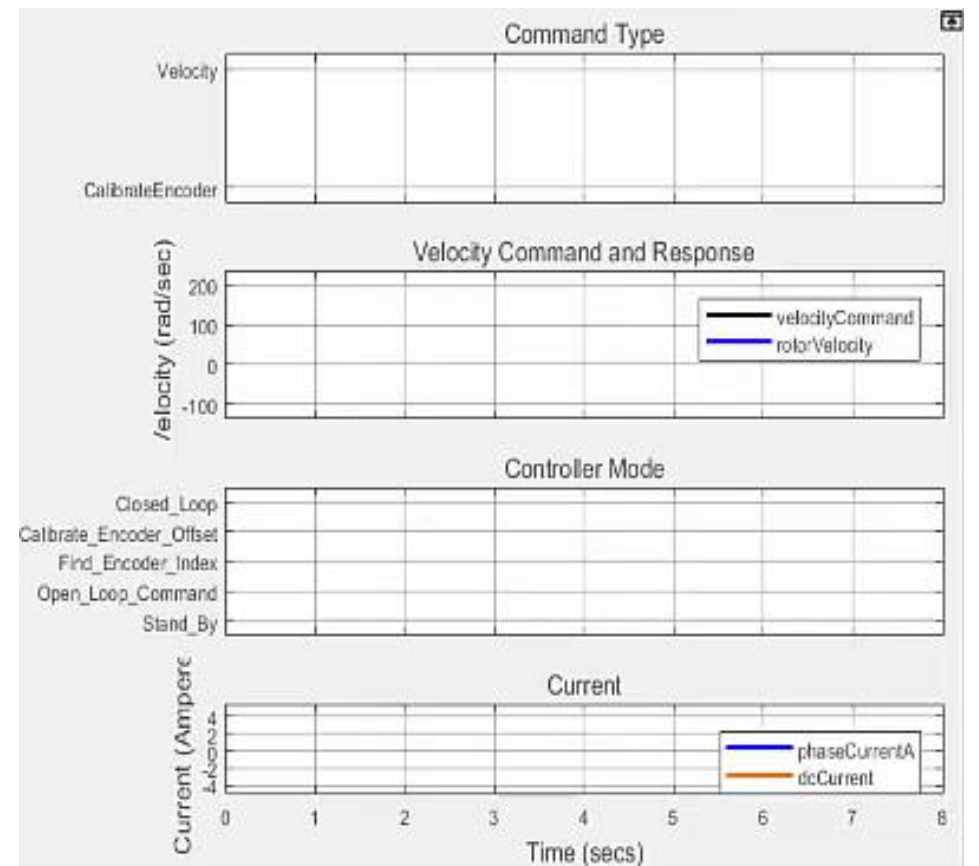


# MATLAB EXPO 2018

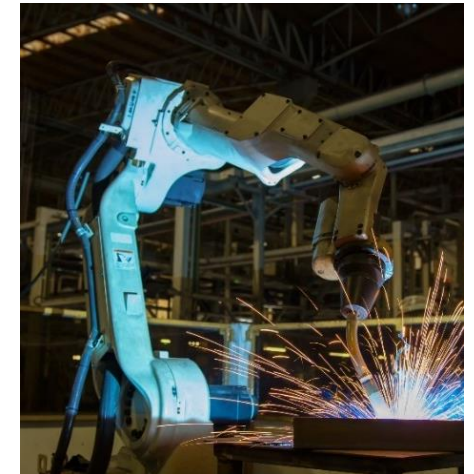
## Hardware and Software Co-Design for Motor Control Applications

Gaurav Dubey  
Durvesh Kulkarni

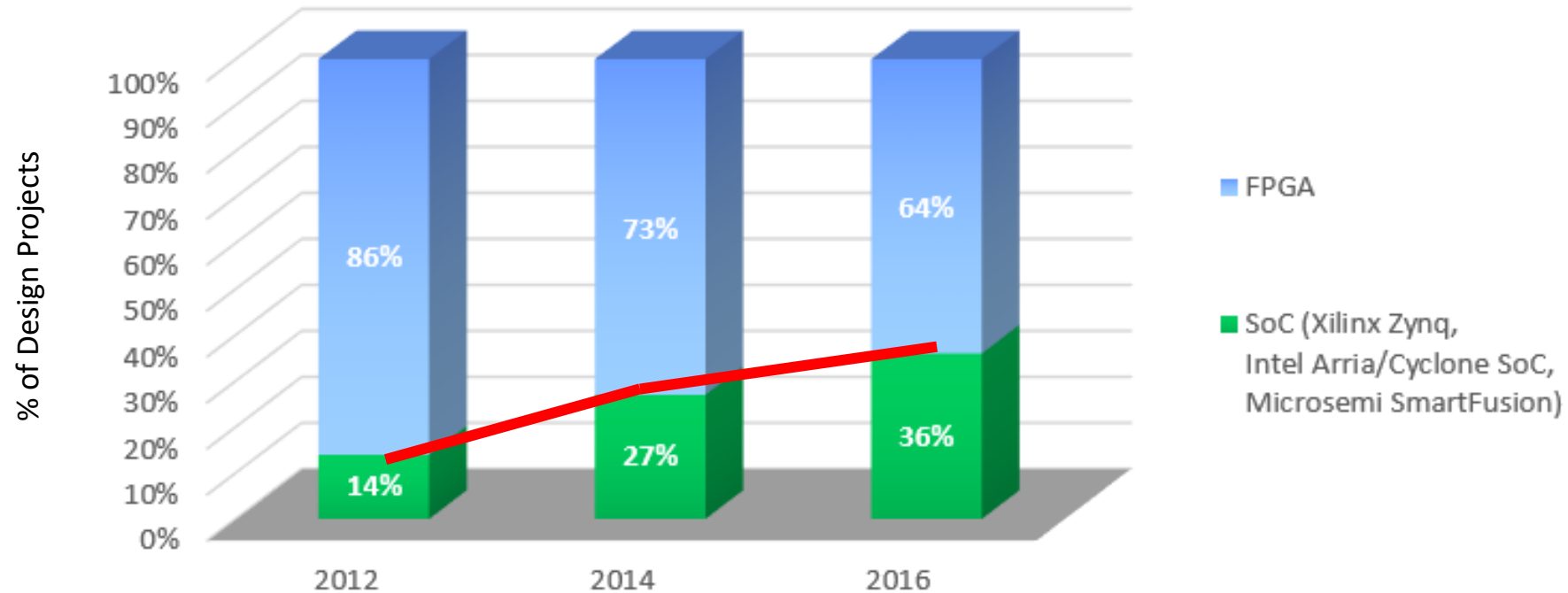


# Key trend: Increasing demands from motor drives

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



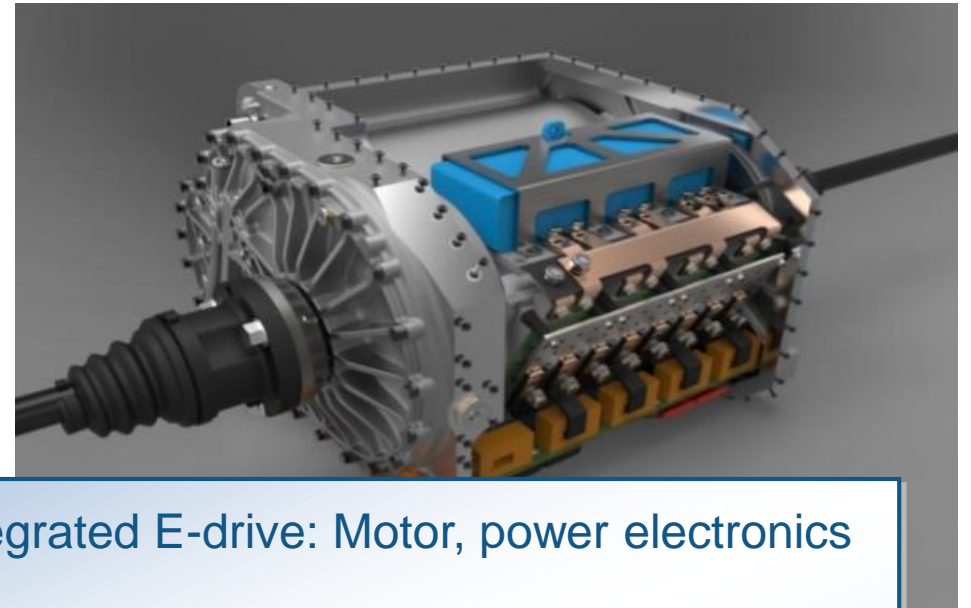
# Key Trend: SoCs are now used in 36% of new FPGA projects



# 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!

[Link to video of presentation](#)



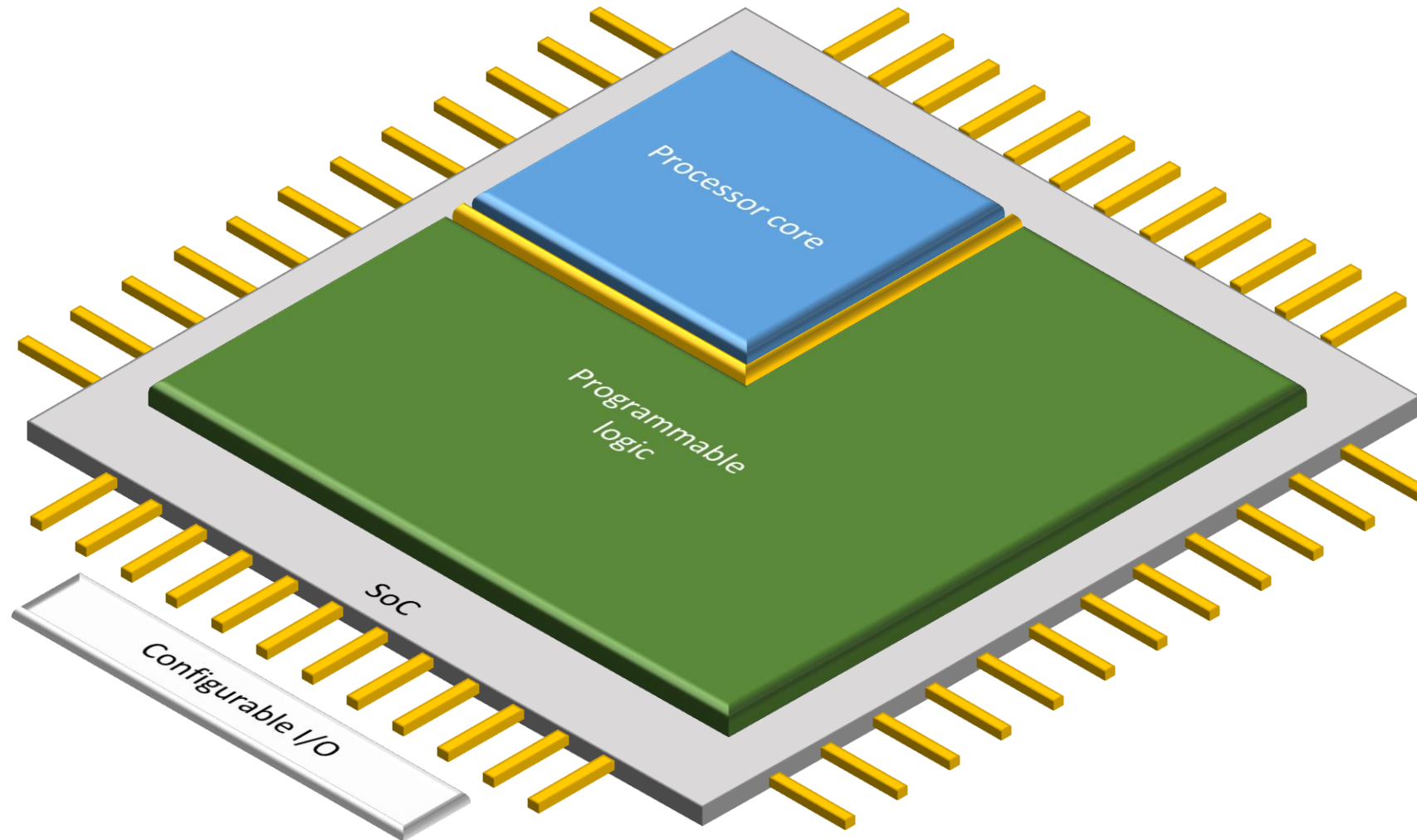
- ✓ Designed integrated E-drive: Motor, power electronics and software
- ✓ 4 different control strategies implemented
- ✓ Completed in 1.5 years with 2FTE's
- ✓ Models reusable for production
- ✓ Smooth integration and validation due to development process – thorough validation before electronics are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

# 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

# What's an SoC?



# 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



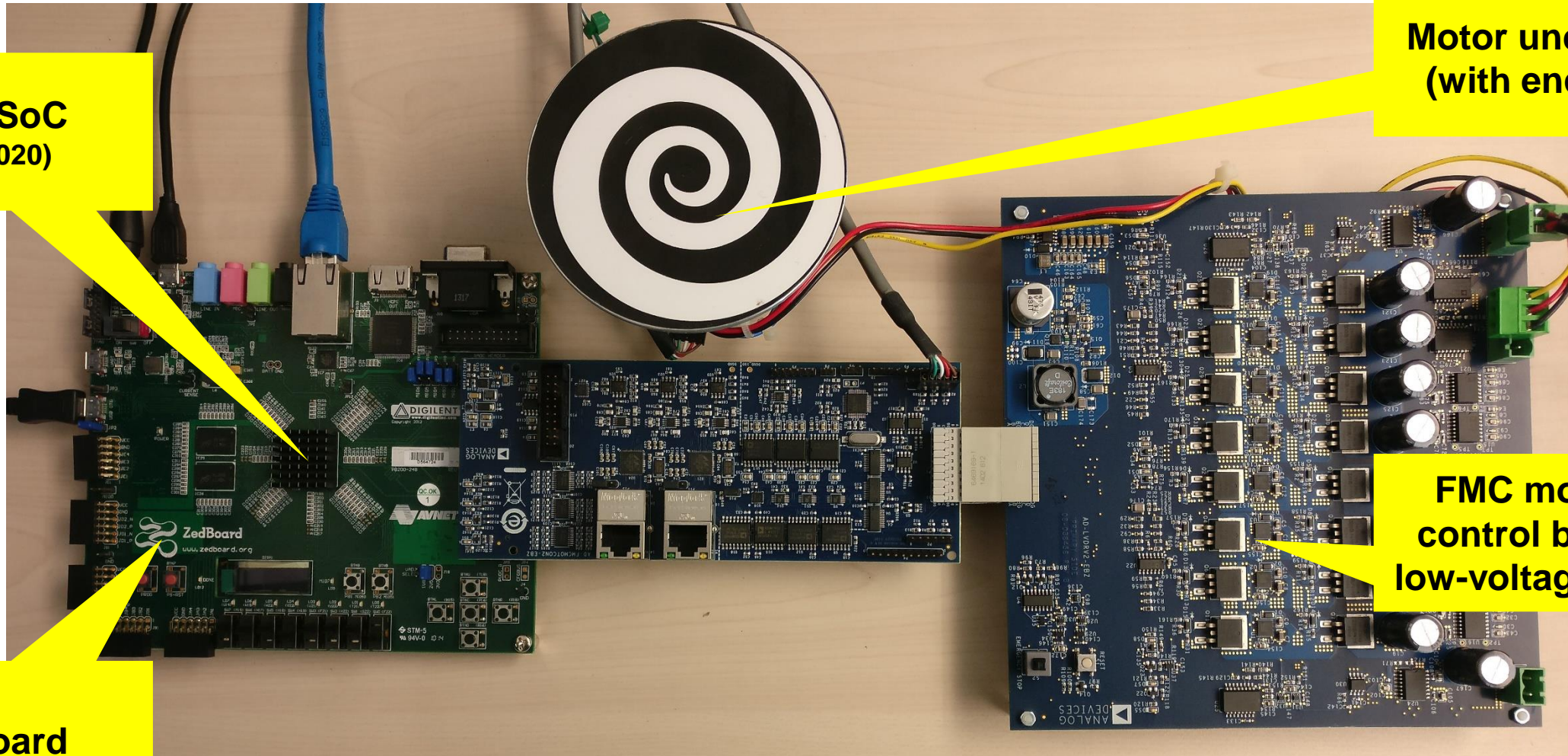
# SoC Hardware Setup

Zynq SoC  
(XC7Z020)

Motor under test  
(with encoder)

FMC module:  
control board +  
low-voltage board

ZedBoard



**focZynqTestBench - Simulink**

File Edit View Display Diagram Simulation Analysis Code Tools Help

focZynqTestBench

### Field-Oriented Control of Velocity Hardware/Software Test Bench

Copyright 2015-2017 The MathWorks, Inc.

System\_Inputs

C/D

Controller\_Algorithm

D/C

Motor\_And\_Load

Verify\_Outputs

Ready

View 1 warning 68%

VariableStepAuto

**System\_Response**

File Tools View Simulation Help

#### Command Type

Velocity (rad/sec)

#### Velocity Command and Response

#### Controller Mode

#### Current

Current (Ampere)

Time (secs)

Ready

Sample based

# Key controller and peripheral components

Model  
C

Model  
HDL

Hand  
HDL

## Velocity Control

- 1 kHz
- ARM Core

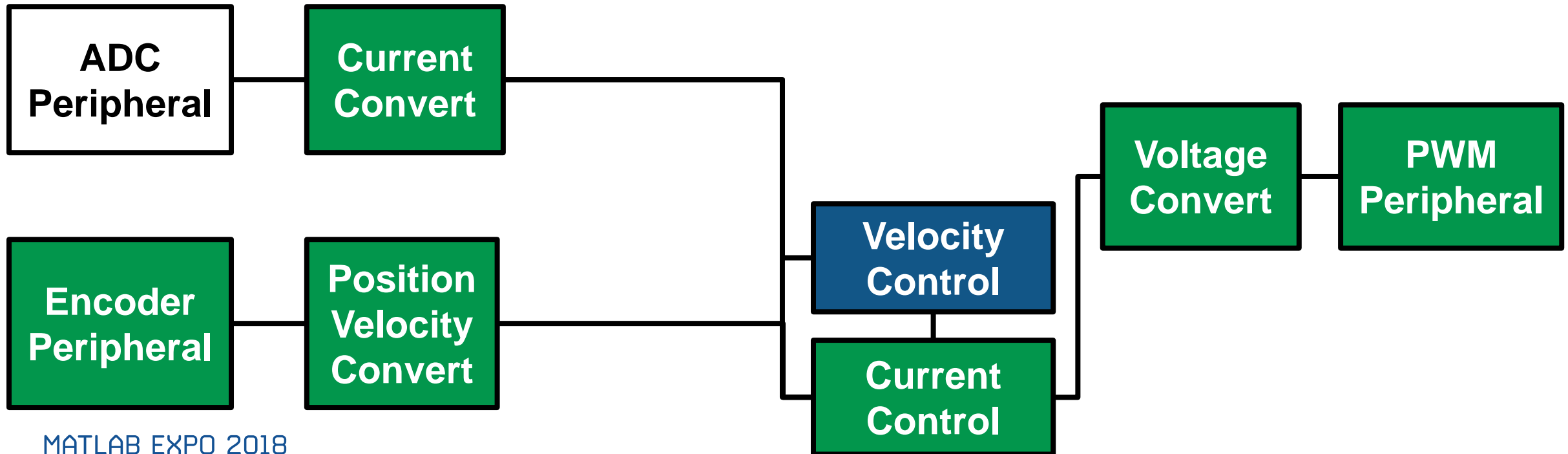
## Current Control

- 25 kHz
- Sine & Cosine
- Clarke & Park Transformations
- FPGA Core

Velocity  
Control

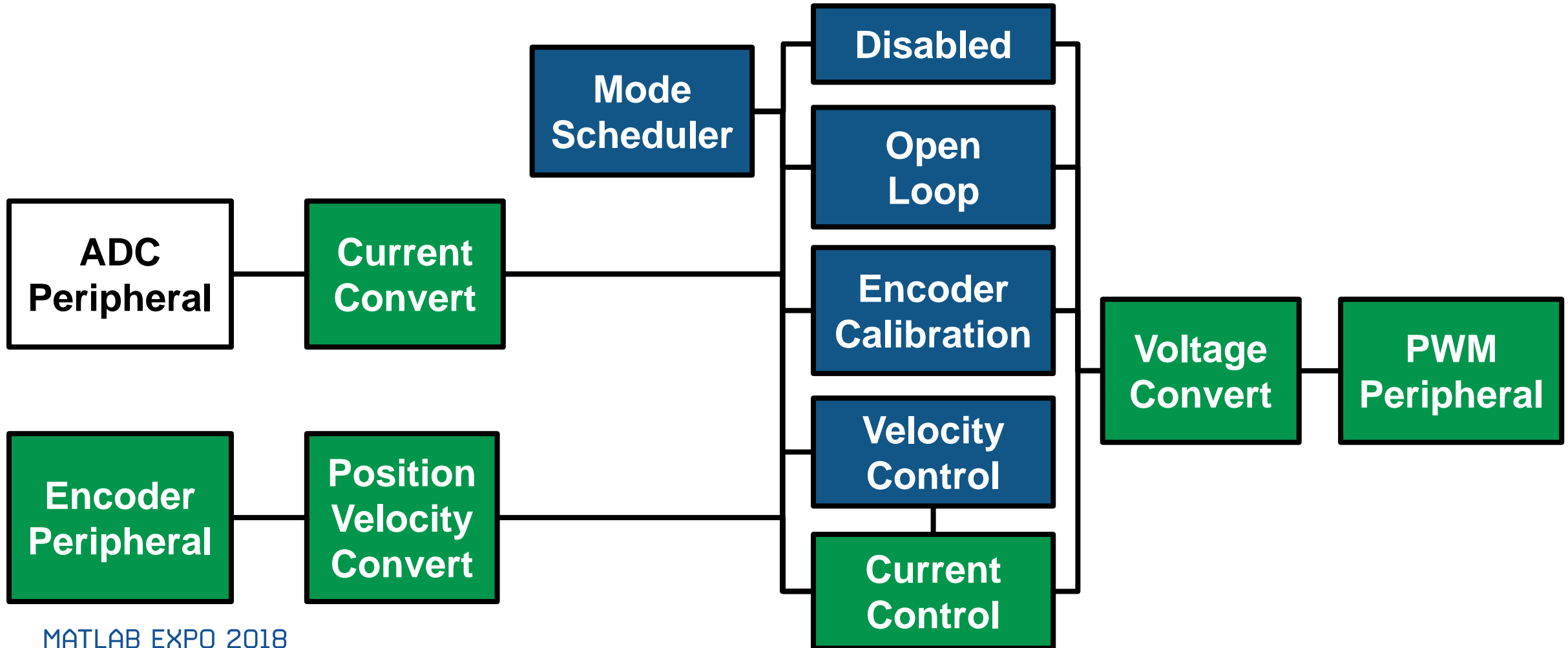
Current  
Control

# Key controller and peripheral components

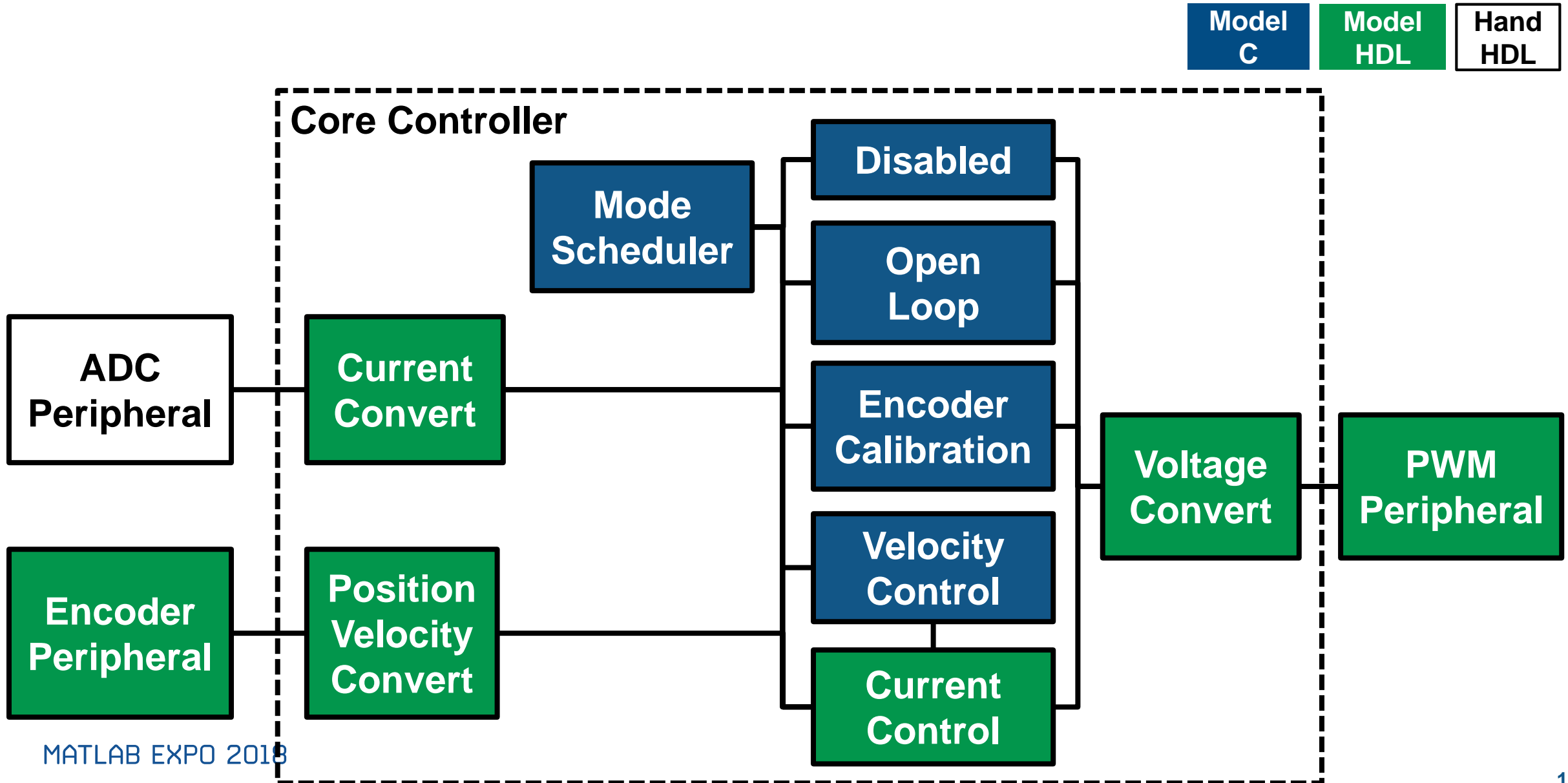


# Key controller and peripheral components

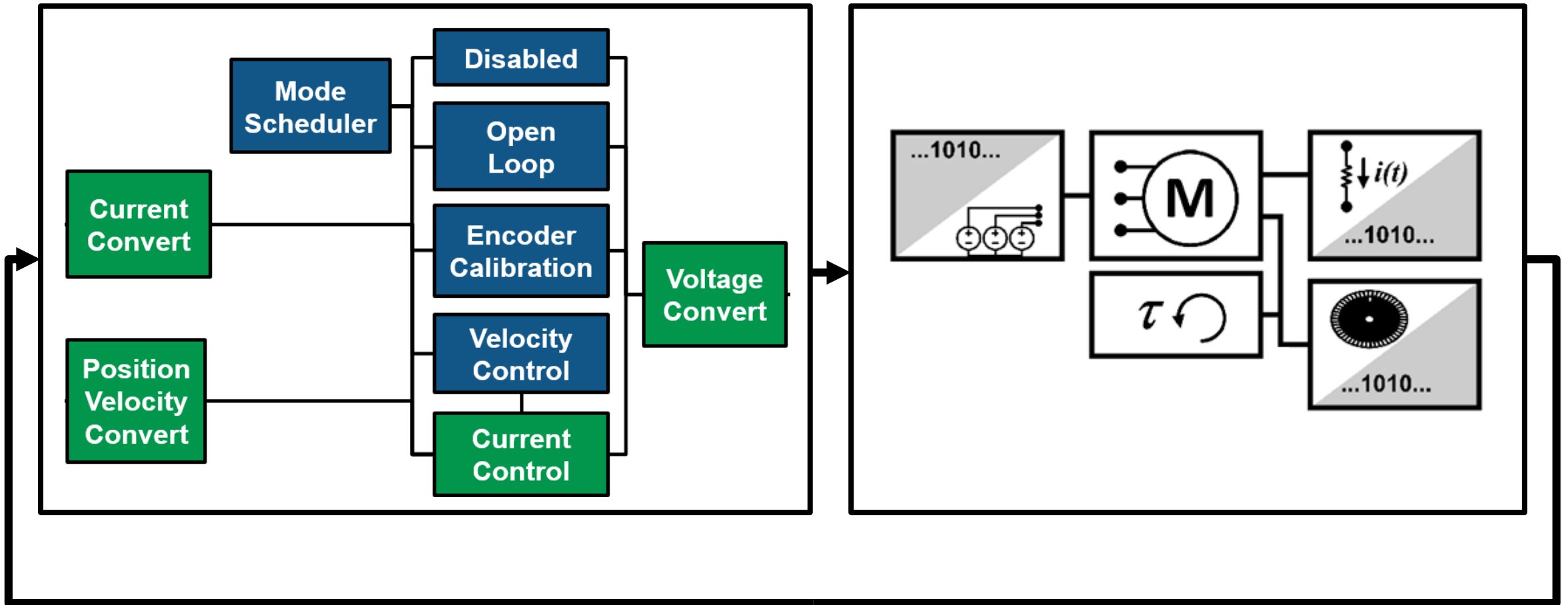
Model C	Model HDL	Hand HDL
------------	--------------	-------------



# Key controller and peripheral components

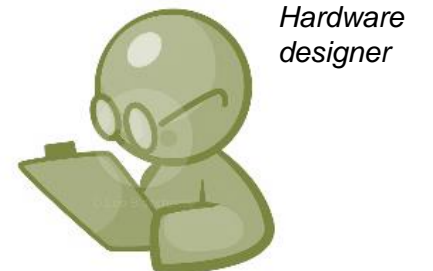
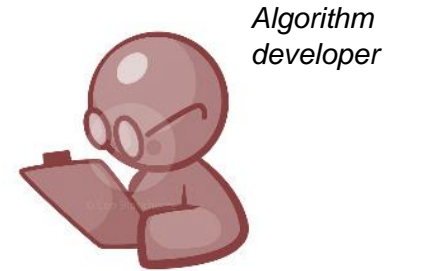
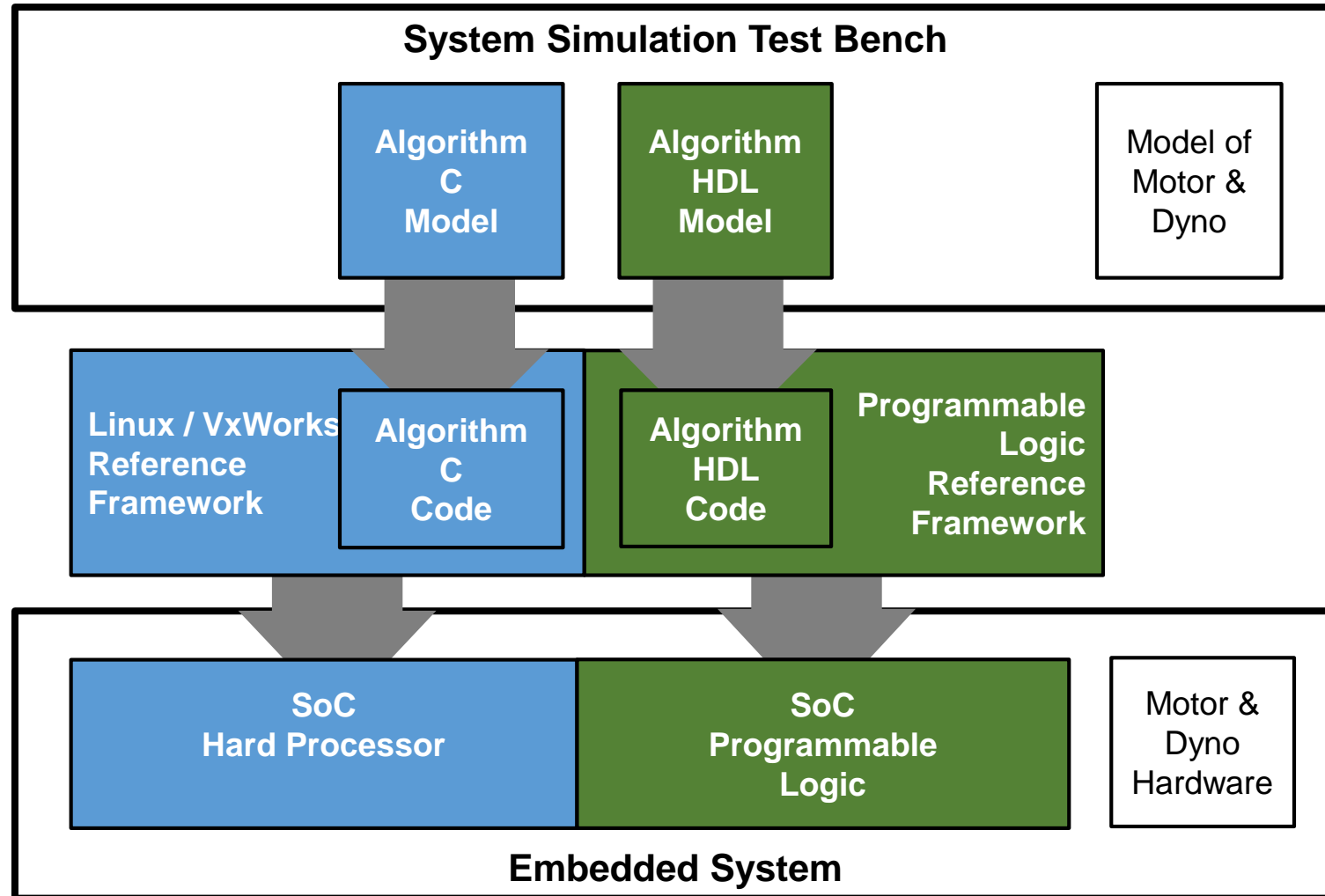
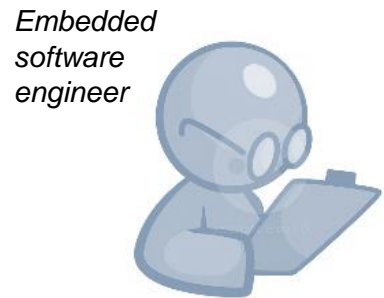


# Test controller algorithm with simulation

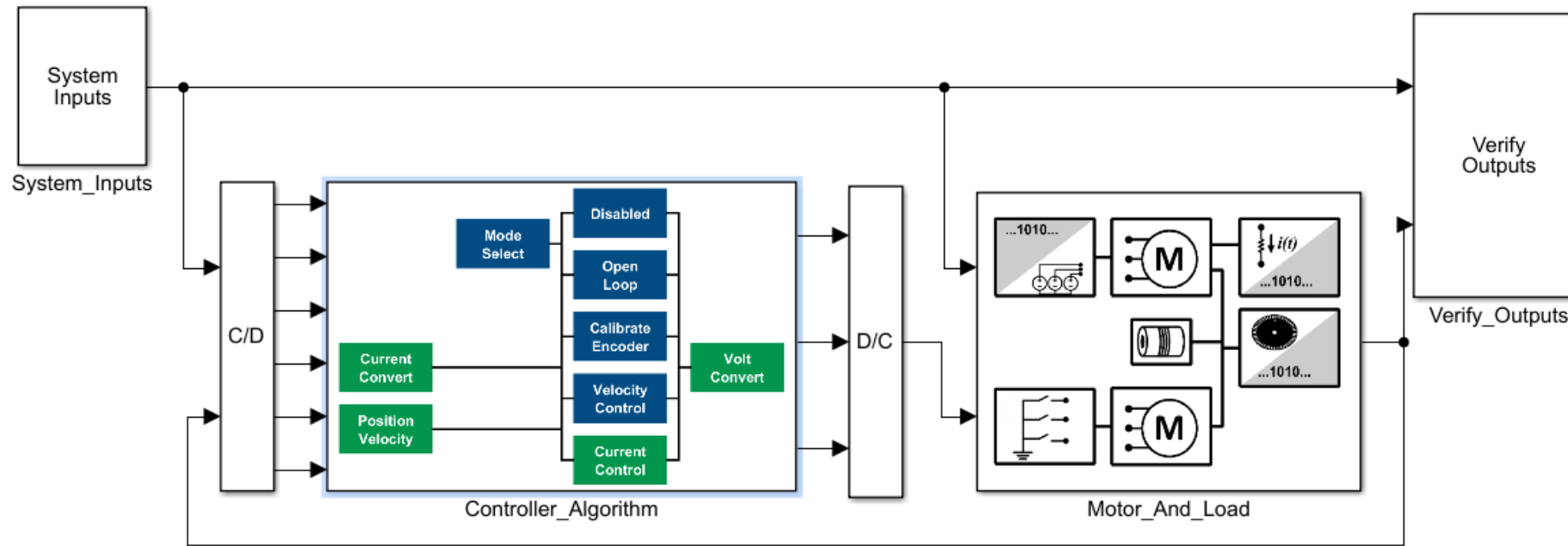




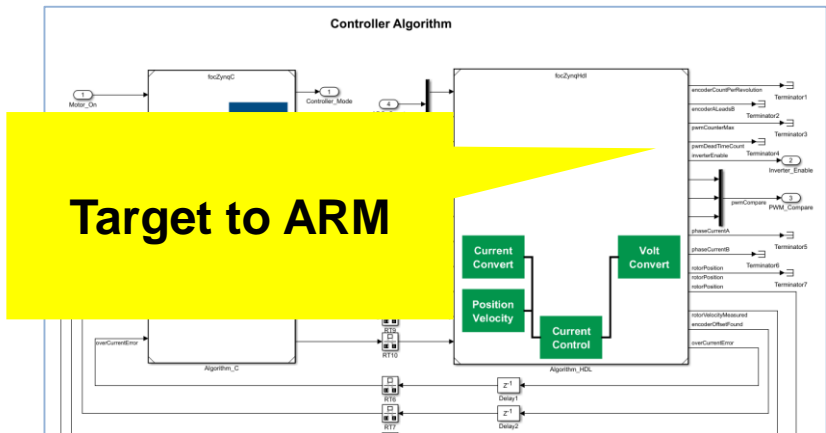
# Conceptual workflow targeting SoCs



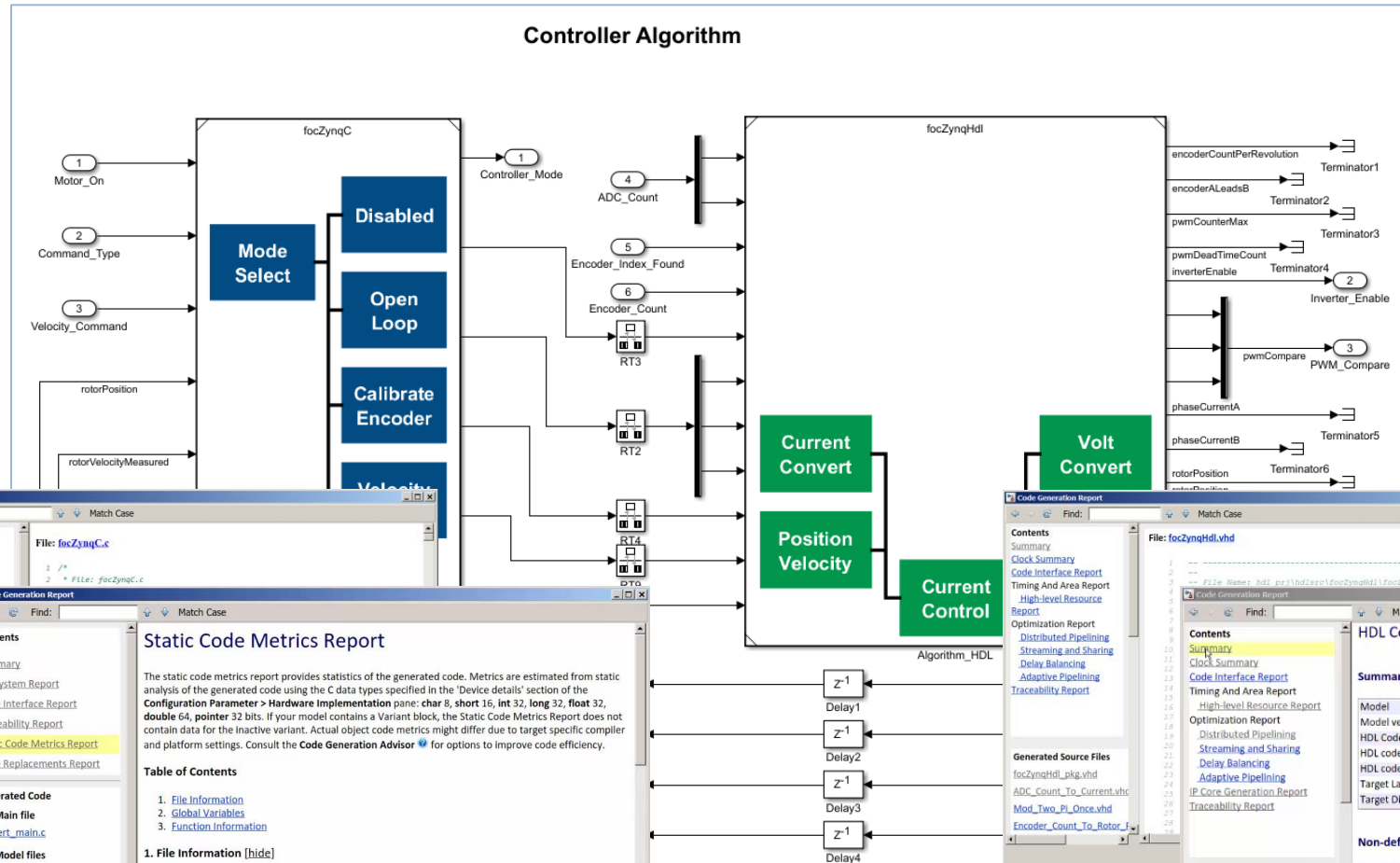
# Hardware/software partitioning



**Target to Programmable Logic**



# Code Generation



**Static Code Metrics Report**

The static code metrics report provides statistics of the generated code. Metrics are estimated from static analysis of the generated code using the C data types specified in the 'Device details' section of the **Configuration Parameter > Hardware Implementation** pane: char 8, short 16, int 32, long 32, float 32, double 64, pointer 32 bits. If your model contains a Variant block, the Static Code Metrics Report does not contain data for the inactive variant. Actual object code metrics might differ due to target specific compiler and platform settings. Consult the **Code Generation Advisor** for options to improve code efficiency.

**Table of Contents**

- File Information
- Global Variables
- Function Information

**1. File Information [hide]**

(-) Summary (excludes ert\_main.c)

Number of .c files : 5  
 Number of .h files : 9  
 Lines of code : 901  
 Lines : 2,116

(-) File details

File Name	Lines of Code	Lines	Generated On
focZynqC.c	417	883	04/19/2017 9:48 PM
focZynqC.h	130	347	04/19/2017 9:48 PM
focZynqC_data.c	66	318	04/19/2017 9:48 PM

**HDL Code Generation Report Summary for focZynqHdl**

**Summary**

Model	focZynqHdl
Model version	1.368
HDL Code version	3.10
HDL code generated on	2017-04-21 14:19:09
HDL code generated for	focZynqHdl
Target Language	VHDL
Target Directory	hdl_prj\hdlsrc

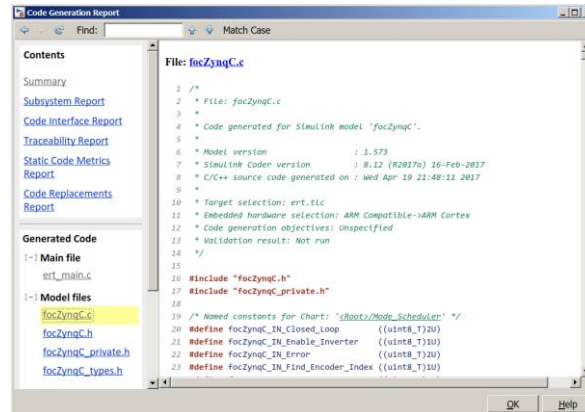
**Non-default model properties**

ClockRatePipelining	off
EnablePrefix	oversampledClockEnable
HDLSubsystem	focZynqHdl
ModulePrefix	focZynqHdl_ip_src
OptimizationReport	on
Oversampling	2000
ReferenceDesign	Motor Control Reference Design
ResetType	Synchronous
ResourceReport	on
ScalarizePorts	on
SynthesisTool	Xilinx Vivado
SynthesisToolChinFamily	7ynn

# Code Generation

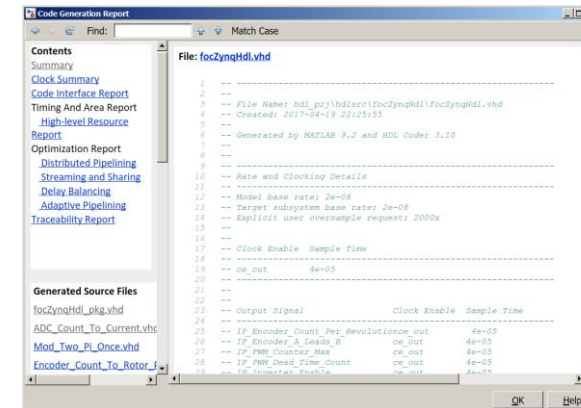
## C Code

Generating Industry Standards Production C Code Using Embedded Coder  
 16:30–17:15

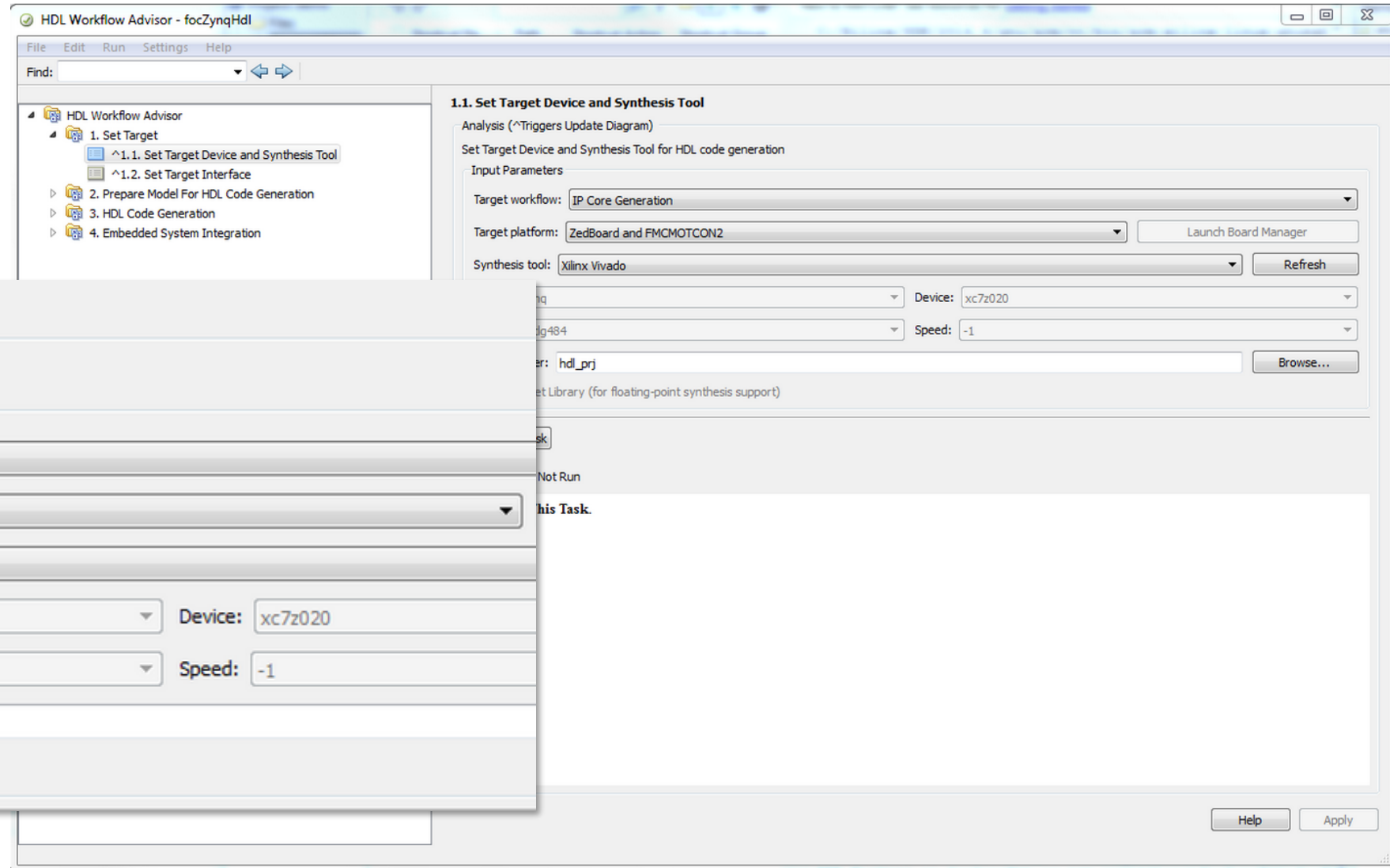


## HDL Code

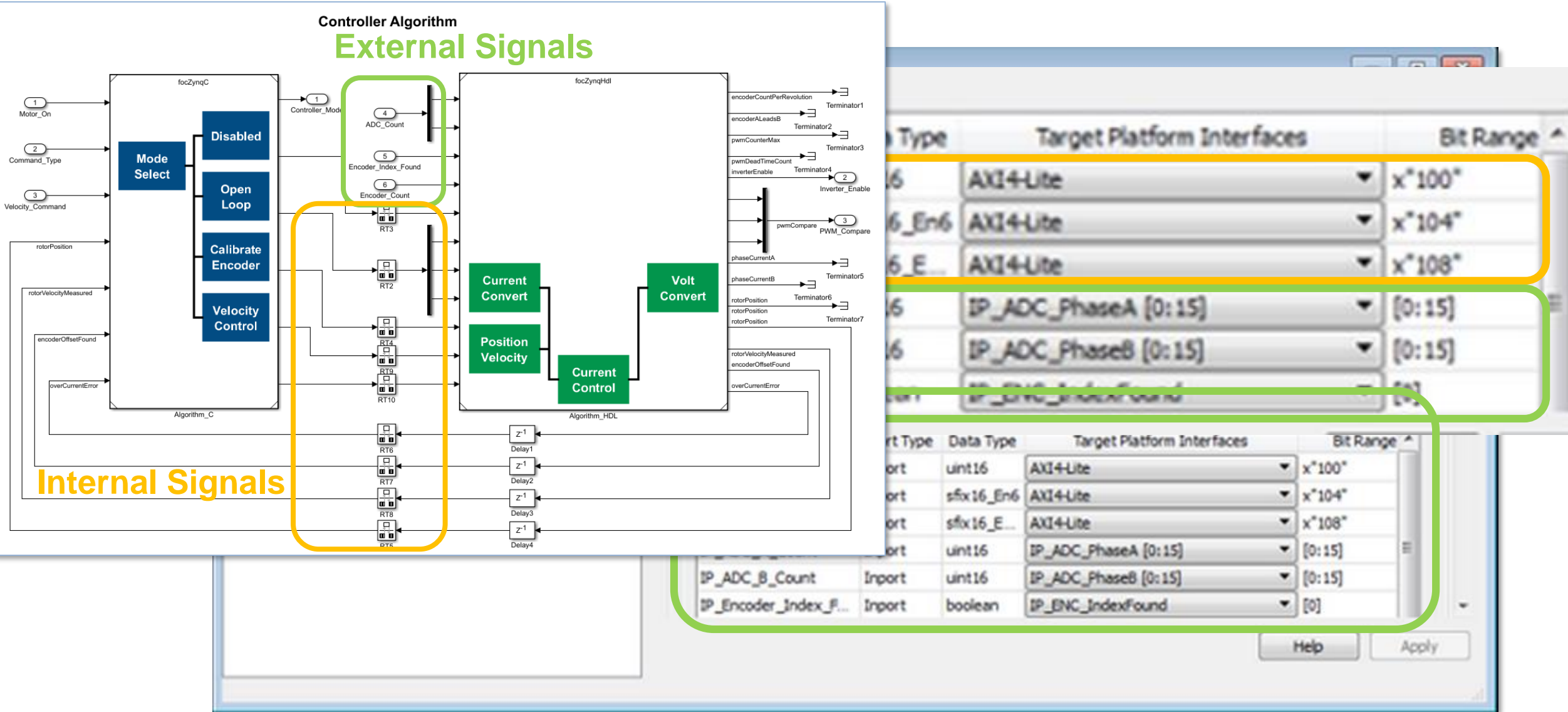
Designing and Prototyping Digital Systems on SoC FPGAs  
 16:30–17:15



# Deploy Bitstream to Programmable Logic



# Set Target Interface for Bitstream





# Build FPGA Bitstream

The screenshot shows the HDL Workflow Advisor interface for a project named 'focZynqHdl'. The main window displays the '4.3. Build FPGA Bitstream' task configuration. The task description is 'Synthesis and generate bitstream for embedded system on FPGA'. Under 'Input Parameters', the 'Run build process externally' checkbox is checked, and the 'Tcl file for synthesis build' is set to 'Default'. A 'Run This Task' button is visible below the configuration.

An 'AXI Interface Library' window is overlaid on the left side of the main window. It shows a list of interface signals:

- AXI\_Enable\_Inverter
- AXI\_Phase\_Voltage\_A
- AXI\_Phase\_Voltage\_B
- AXI\_Phase\_Voltage\_C
- AXI\_Enable\_Closed\_Loop
- AXI\_Current\_Command
- AXI\_Encoder\_Offset
- AXI\_Phase\_Current\_A
- AXI\_Phase\_Current\_B
- AXI\_Electrical\_Position
- AXI\_Rotor\_Position
- AXI\_Rotor\_Velocity
- AXI\_Encoder\_Index\_Found
- AXI\_Overcurrent\_Error

The library is titled 'AXI Interface Library' and was created on '03-Jan-2017 18:06:34'. The name 'AXI\_Interface' is visible at the bottom of the library window.

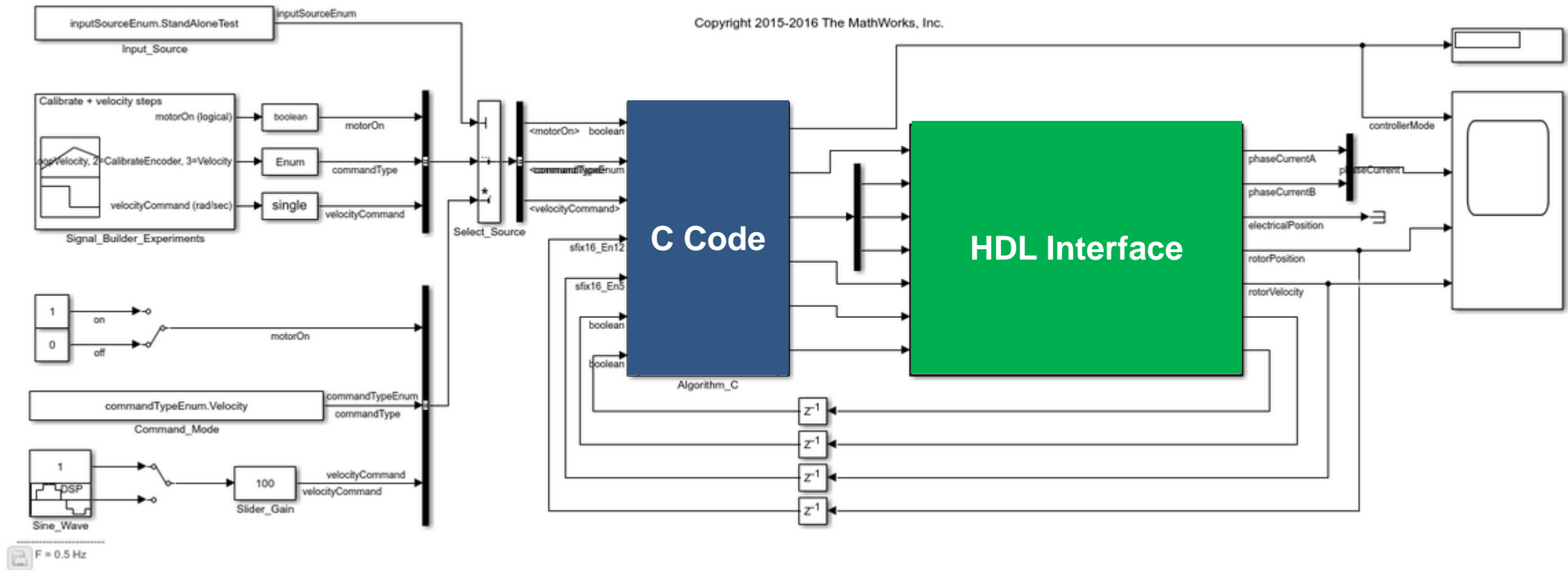
A context menu is open over the 'Run This Task' button, showing options: 'Run This Task', 'Run to Selected Task', 'Reset This Task', and 'What's This?'. The 'Run to Selected Task' option is highlighted.



# Zynq ARM Deployment

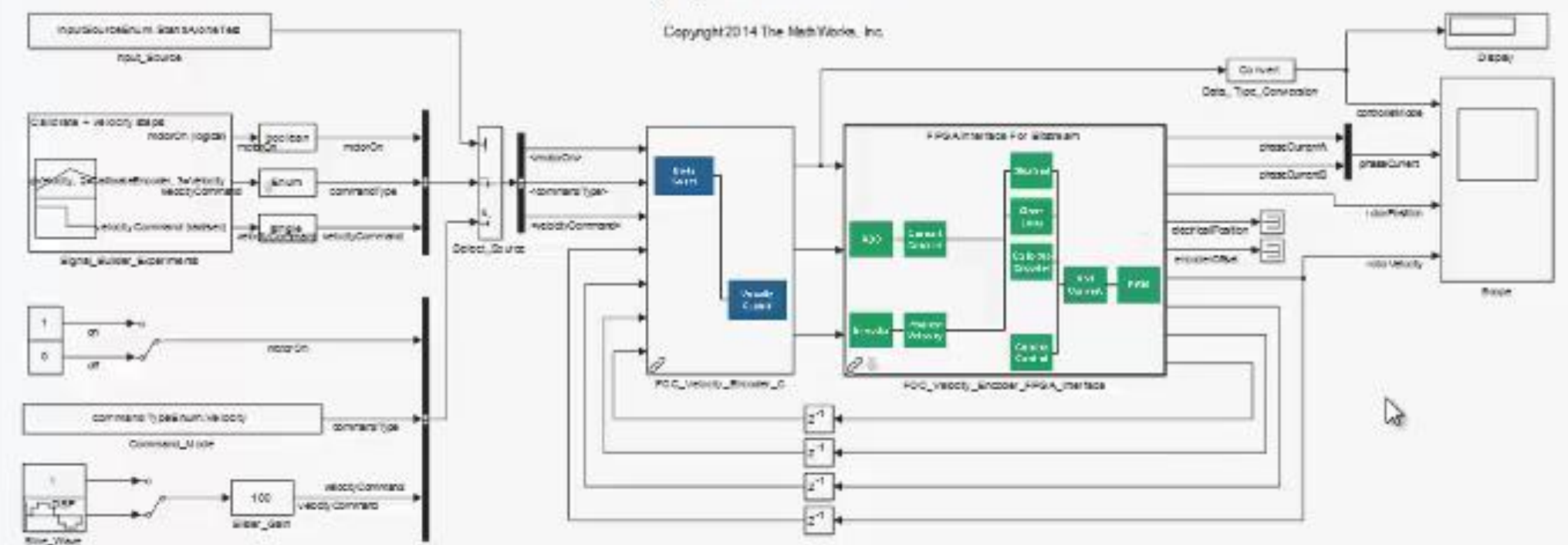
## Field-Oriented Control of Velocity Zynq ARM Deployment for AD-FMCMOTCON2

Copyright 2015-2016 The MathWorks, Inc.



### Field-Oriented Velocity Control Zynq ARM Real-Time

Copyright 2014 The MathWorks, Inc.



Documentation

0.9 1

Offset=0

100%

5:07 AM

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

## 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



A SCARA robot.

**“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

# 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
  - Field-Oriented-Control based Motor Control Application using System-on-Chip (SoC) Architecture
  - Achieve Industry & Safety Standards Compliance using Efficient Model Verification & Validation and Production Code Generation
  - Model-Based Design for Software-Defined Radio
- 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:
  - [Define and Register Custom Board and Reference Design for SoC Workflow](#)
  - [Field-Oriented Control of a Permanent Magnet Synchronous Machine on SoCs](#)



MathWorks is honored to receive the Embedded World Award 2017 in the Tools Category for HDL Coder. <http://owl.li/nBzd309XYxW>



288 interessant • 6 commentaren

### Speaker Details

Email: [Durvesh.Kulkarni@mathworks.in](mailto:Durvesh.Kulkarni@mathworks.in)

LinkedIn: <https://www.linkedin.com/in/durvesh-kulkarni-17402527/>

### Speaker Details

Email: [Gaurav.Dubey@mathworks.in](mailto:Gaurav.Dubey@mathworks.in)

LinkedIn: <https://www.linkedin.com/in/gauravdubey4/>

- **Share your experience with MATLAB & Simulink on Social Media**

- Use #MATLABEXPO
- I use #MATLAB because..... Attending #MATLABEXPO
- Examples
  - I use #MATLAB because it helps me be a data scientist! Attending #MATLABEXPO
  - Learning new capabilities in #MATLAB and #Simulink at #MATLABEXPO.

- **Share your session feedback:**

Please fill in your feedback for this session in the feedback form

### Contact MathWorks India

Products/Training Enquiry Booth

Call: 080-6632-6000

Email: [info@mathworks.in](mailto:info@mathworks.in)