Motor Controls Implementation on Systems-On-Chip

Jorik Caljouw
Key Takeaways

Meet stringent requirements and reduce costs

Reduce hardware testing time up to 5x

Manage design complexity and improve team collaboration
Punch Powertrain develops complex SoC-based motor control

- Powertrains for hybrid and electric vehicles
- Hardware choice through simulations
- Traditional microcontroller too slow
- No experience designing FPGAs!

✓ Designed integrated E-drive: Motor, power electronics and software
✓ 4 different control strategies implemented
✓ Done in 1.5 years with 2FTE’s
✓ Models reusable for production
✓ Smooth integration and validation due to development process
Key trend: Increasing demands from motor drives
Systems-on-Chip for motor control
Key Trend: SoCs are now used in 36% of new FPGA projects

Challenges in using SoCs for Motor and Power Control
Why use Model-Based Design to develop motor control applications on SoCs?
ZedBoard

Zynq SoC (XC7Z020)

Load motor

FMC module: control board + low-voltage board

Mechanical coupler

Motor under test (with encoder)
Field-Oriented Control of Velocity
Hardware/Software Test Bench

Copyright 2015-2017 The MathWorks, Inc.
Conceptual workflow targeting SoCs

System Simulation Test Bench

- Algorithm C Model
- Algorithm HDL Model
- Model of Motor & Dyno

Linux / VxWorks Reference Framework
- Algorithm C Code
- Algorithm HDL Code
- Programmable Logic Reference Framework

SoC Hard Processor
- SoC Programmable Logic
- Motor & Dyno Hardware

Embedded System

MATLAB EXPO 2017
Hardware/software partitioning

Target to ARM

Target to Programmable Logic
Code Generation

Controller Algorithm

Mode Select
- Disabled
- Open Loop
- Calibrate Encoder

Mode Changes
- Mode
- Select

Current Convert
- Encoder_Present
- Encoder_Peak
- Encoder_Count
- Encoder_Clock
- Encoder_Clock_Fault

Position Velocity
- Inverter_S enable
- Inverter_Decoder
- Inverter_Decoder_Fault

Static Code Metrics Report

Table of Contents
- File Information
- Global Variables
- Local Variables
- Executable Information

- Number of files: 5
- Number of files: 5
- Lines of code: 1,000
- Files details:
  - File 1: 417
  - File 2: 347

HDL Code Generation Report Summary for focZynqHDL

Summary
- Model: focZynqHDL
- Model name: focZynqHDL
- HDL Code generated: yes
- HDL code generated for:
  - focZynqHDL
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

“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

MATLAB EXPO 2017
Key Takeaways

Meet stringent requirements and reduce costs

Reduce hardware testing time up to 5x

Manage design complexity and improve team collaboration
Learn More

- Get an in-depth demo in the Technology Showcase
  - New: see award-winning Native Floating Point in HDL Coder!

- 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](#)
How to get started?

- Embedded Systems
- FPGA Design
- Xilinx Zynq SoCs