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!

✓ 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
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)

ZedBoard

Motor under test (with encoder)

FMC module: control board + low-voltage board
Key controller and peripheral components

Velocity Control
- 1 kHz
- ARM Core

Current Control
- 25 kHz
- Sine & Cosine
- Clarke & Park Transformations
- FPGA Core
Key controller and peripheral components
Key controller and peripheral components

1. ADC Peripheral
   - Current Convert
2. Encoder Peripheral
   - Position Velocity Convert

- Mode Scheduler
  - Disabled
  - Open Loop
  - Encoder Calibration
  - Velocity Control
  - Current Control

- Voltage Convert
- PWM Peripheral
Key controller and peripheral components

Core Controller

- Mode Scheduler
- Disabled
- Open Loop
- Encoder Calibration
- Voltage Convert
- PWM Peripheral

ADC Peripheral
- Current Convert

Encoder Peripheral
- Position Velocity Convert
- Current Control

Model C
Model HDL
Hand HDL

MATLAB EXPO 2018
Test controller algorithm with simulation
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

Algorithm developer
Hardware designer

Embedded software engineer

MATLAB EXPO 2018
Hardware/software partitioning

Target to ARM

Target to Programmable Logic
Code Generation
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

External Signals

Internal Signals
Build FPGA Bitstream

AXI Interface Library

Library created on 03-Jan-2017 18:06:34

AXI_Enable_Inverter
AXI_Phase_Current_A
AXI_Phase_Voltage_A
AXI_Phase_Voltage_B
AXI_Phase_Voltage_C
AXI_Enable_Closed_Loop
AXI_Current_Command
AXI_Override_Offset
AXI_Interface

4.3. Build FPGA Bitstream

Analysis
Synthesis and generate bitstream for embedded system on FPGA
Input Parameters
Run build process externally
Tcl file for synthesis build: Default
Browse...

Run This Task
Run to Selected Task
Not Run
Reset This Task
What's This?
Field-Oriented Velocity Control
Zynq ARM Real-Time

Copyright 2014 The MathWorks, Inc.
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
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
• 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