Motor Control: Model-Based Design from Concept to Implementation on heterogeneous SoC FPGAs

Alexander Schreiber, MathWorks
Some components of a production application

Production

ARM

Algorithm C
Linux Driver

System Code

AXI Bus

AXI Interface

Algorithm HDL

Programmable Logic

Motor

System

IP1

IP2

IP3
From simulation to production

Simulation

- Simulink
  - Algorithm Model
  - Algorithm Model
  - Motor Model

Production

- ARM
  - Algorithm C
  - Linux Driver
  - AXI Bus
  - AXI Interface
  - Algorithm HDL
- Programmable Logic
  - IP1
  - IP2
  - IP3
  - Motor
  - System

Embedded Coder
HDL Coder
From simulation to prototype to production

Simulation
- Simulink
  - Algorithm Model
  - Motor Model

Prototype
- ARM
  - Algorithm C
  - Linux Driver
  - AXI Interface
  - Prog. Logic
  - HDL Coder

Production
- ARM
  - Algorithm C
  - Linux Driver
  - AXI Interface
  - Algorithm HDL
  - Programmable Logic
  - Vivado
  - System Code
  - IP1
  - IP2
  - IP3
  - System
How can models help you design a controller for Zynq?

- **Simulate on your desktop**
  - Model controller and plant system dynamics
  - Design and debug components at control loop fidelity
  - Assemble and verify components at implementation fidelity

- **Prototype on hardware**
  - Generate HDL code and build bitstream
  - Generate C code and build ARM executable
  - Collect hardware results and verify against simulation

- **Generate C/HDL code for production**
Example: Design a six-step trapezoidal controller
Example: Design a six-step trapezoidal controller
Split system into smaller design components

System Components

- Velocity Control
- Velocity Estimate
- Six-Step Commutation
- PWM
- Hall
- Period

PWM Component

- PWM

Velocity Estimate Components

- Velocity Estimate
- Period
- Hall

Control Loop Components

- Velocity Control
- Hall
- Six-Step Commutation

C code

HDL code
Simulation let’s you choose where you want to start

System Components

Velocity Control → Six-Step Commutation → PWM
Velocity Estimate → Period → Hall

PWM Component

Velocity Estimate Components

Control Loop Components

Velocity Control → Hall → Six-Step Commutation

C code  HDL code
Let’s use the tools!

Simulate control loop dynamics
Design and simulate control loop components

- Specify tests and simulate response with continuous time solver
- Model motor and load with fidelity to capture dynamics of interest
- Model control loop C/HDL components
  - Specify velocity control to runs at 1kHz rate
  - Commutator rate will be determined by solver
Assemble components into system testbench

**Control Loop Components**

- Velocity Control
- Hall
- Six-Step Commutation

C code

HDL code
Assemble components into system testbench

System Components

- Velocity Control
- Velocity Estimate
- Six-Step Commutation
- PWM
- Hall

PWM Component

- PWM

Velocity Estimate Component

- Velocity Estimate
- Period
- Hall

Control Loop Components

- Velocity Control
- Hall
- Six-Step Commutation
Assemble components into system testbench

- Group components which will be implemented in C/HDL
- Assemble and simulate system to assess impact of peripherals on control loop
- Time to run system simulation is dictated by HDL peripheral (PWM) dynamics
Compare component and system simulations

- Control loop component simulation runs faster but ignores sensor effects
Compare component and system simulations

- Control loop component simulation runs faster but ignores sensor effects
- System simulation includes sensor effects but runs slower due to peripheral dynamics
- Velocity resolution due to hall effect sensor is visible in system level simulation
How do I get from simulation to prototype?

Processing System

- Cortex™-A9
- Algorithm C Specification Model
- Open-source LINUX

Programmable Logic

- Algorithm HDL Specification Model

Motor FMC Card

- Inverter Module
- Hall Interface

Isolation

Ethernet

MathWorks
Generate a bitstream for programmable logic

Zynq Support Package enables you to...

- Generate bitstream consisting of algorithmic HDL code from models and interfaces to FPGA pins and AXI-Bus
Generate an executable for ARM

Zynq Support Package enables you to…

- Generate ARM executable consisting of algorithmic C code from models and interfaces to AXI-Bus
Generate an executable for ARM

Zynq Support Package enables you to…

- Generate ARM executable consisting of algorithmic C code from models and interfaces to AXI-Bus
- Provides data interface between Simulink and ARM executable via Ethernet
Prototype on Zynq with interactive tests

- Switches and scopes in Simulink model act as an interface to generated executable running on ARM.
Let’s use the tools to…

Prototype on Zynq
Hardware results track simulation results
Hardware results track simulation results

- Prototyping environment enables verification of modeling assumptions against hardware
- Correlation of results provides confidence in plant and controller models
- Confidence in models enables engineers to spend more time in simulation and less time on dynos
Generate algorithmic code for production

**Algorithm C Specification Model**

Generate C code for the algorithm (i.e. function call) that you can integrate into your production project

```c
/* Model step function */
void Controller(boolean_t motor_on, real32_t *arg_velocity_command, int32_t
*arg_hall_position_delta, uint32_t *arg_hall_timer_delta,
real32_t *arg_velocity, int32_t *arg_voltage)
{
}
```

**Algorithm HDL Specification Model**

Generate HDL code for the algorithm (i.e. entity) that you can integrate into your production project

```vhdl
ENTITY Bitstream IS
PORT( CLK_IN : IN std_logic;
      reset : IN std_logic;
      clk_enable : IN std_logic;
      hall_a : IN std_logic;
      hall_b : IN std_logic;
      hall_c : IN std_logic;
      ax1_motor_on : IN std_logic;
      ax1_voltage_level : IN std_logic_vector(17 DOWNTO 0);
      oe_out : OUT std_logic;
      ...
```
How did models help us design a controller for Zynq?

Simulation
- Simulink
- Algorithm Model
- Algorithm Model
- Motor Model

Prototype
- ARM
- Algorithm C
- Linux Driver
- AXI Bus
- AXI Interface
- Algorithm HDL
- Prog. Logic
- Algorithm HDL
- Motor

Production
- Algorithm C

Diagram:
- Embedded Coder
- HDL Coder
How do models become code?

Simulation

- Simulink
- Algorithm Model
- Embedded Coder
- Algorithm C

- Algorithm Model
- HDL Coder
- Algorithm HDL
- HDL Coder
- Generates HDL code for the algorithm (i.e. an HDL entity)

Embedded Coder
- Generates C code for the algorithm (i.e. a C function call)
How can models help you prototype on Zynq?

Zynq Support Package

- Automates integrating generated C code with an ARM “parent project”
- Automates wrapping generated HDL code into an IP Core and integrating it with “parent project” for programmable logic
- Provides a data interface between Simulink and ARM
How can models help you design a controller for Zynq?

- Simulate on your desktop
  - Model controller and plant system dynamics
  - Design and debug components at control loop fidelity
  - Assemble and verify components at implementation fidelity

- Prototype on hardware
  - Generate HDL code and build bitstream
  - Generate C code and build ARM executable
  - Collect hardware results and verify against simulation

- Generate C/HDL code for production
How can you accelerate your adoption of simulation & code generation?

MathWorks Consultants can help you to:

- Achieve the desired level of accuracy of motor and control models
- Reduce dynamometer time
- Integrate simulation and C/HDL code generation into your development process
- Build in-house competency through knowledge transfer