MathWorks Technology Session for GE
Model Based Design for Real-Time Testing

November 20, 2013
Abhishek Bhat
Ethan Woodruff
Tom Priestley
Marc Semma
System Modeling and Simulation
Real-Time Execution Options

- Real-Time Execution
- Hardware in the Loop
- Rapid Prototyping
- Rapid Prototyping
- Hardware in the Loop
Real-Time Execution Target HW

1. S-function
2. Desktop
3. Rapid Prototyping
4. Embedded Target
5. RTOS Task

- XML
- C

- Zynq FPGA
- iPhone
A Word About ‘Real-Time’

Normalized Simulation Time
($t_{SIM}/t_{CLOCK}$)

Number of Blocks
What’s Next

- Introduction and overview

What is real-time testing?

- How can I use it to build better products?
  - How good is your design?
  - Does it work with real hardware?
  - Does it work with real software?

- Where can I learn more?
Build better products using Real-Time Testing
Question: How do you think this is useful to your application?
Motor Controls using Real Time Testing
So what is Real-Time Testing?
Real-Time Testing

Real-Time in Engineering –

1 Second in Simulation = 1 Second in Real world

Real-Time Testing is the process of running, proving, and validating software/hardware system designs in “normal” modes of operation.
Define Requirements

Modeling & Simulation

System-Level Specification

Rapid Prototyping

Subsystem Design

Software-in-the-Loop Simulation

On Target Rapid Prototyping

Subsystem Implementation

Hardware-in-the-Loop Simulation

System-Level Integration

Subsystem Integration

Full Integration

Model-Based Design
Continuous verification and validation during system design/testing
But why bother about this? ...
Why do Model-Based Design (MBD) Real-Time Simulation & Testing?

It enables you to:

- **Test, verify, validate, and prove** your algorithmic and system designs earlier

- **Evaluate new ideas** using a flexible, scalable, production independent development platform

- **Minimize risk, reduce costs, shorten time-to-market**

- **Achieve determinism** through system modeling, automatic code generation and real-time software/hardware execution
Why Model-Based Design?

80% of development costs are spent identifying and correcting defects

Number of bugs found shifts to earlier in development phase

Software quality optimization: balancing business transformation and risk, Michael Lundblad, program manager, Rational software, IBM Software Group, Moshe Cohen, offering manager, Rational software, IBM Software Group
What’s Next

- Introduction and overview
- What is real-time testing?

How can I use it to build better products?
- Characterizing a model
- Rapid Prototyping
- Hardware-In-the-Loop

- Where can I learn more?
Real-Time Simulation & Testing Tasks:  
Rapid Prototyping

Real-Time Target Computer

Wiring and Signal Conditioning

Physical Plant Hardware
Real-Time Testing Applications

Motor Controls prototyping example

AC (electrical grid, inverter, combustion) or DC (battery) powered

- Multi-rate models
  - 1 – 20 kHz (outer position and/or velocity control loops)
  - 10 – 200 kHz (inner current control loops)
  - 1-100 MHz (PWM generation)

- Reconfigurable FPGA-based I/O modules
  - Pulse generation and capture, synchronization, quadrature decoding, digital I/O

- High-speed A/D for current measurement

- Additional I/O for supervisory control (CAN, EtherCAT, Ethernet/IP)
Real-Time Simulation & Testing Tasks:

*Hardware-in-the-loop (HIL) Simulation*

Diagram showing the process of HIL simulation with embedded hardware and real-time target computer. The diagram includes a test suite, controller model, plant model, and verification steps.
Real-Time Testing Applications

Aerospace systems example

Hardware-in-the-loop Simulation/Testing

- **Hardware Under Test**
  - Aircraft Engine Controller

- **Simulation**
  - A/C Engines

### Components

- Host-Target Network Switch
- 6 LVDT Simulation channels (IO422)
- Shared/Reflective Memory (IO902)
- FPGA 16 Encoder Emulation channels (IO312)
- 32 24V digital input channels (IO206)
- 32 24V/0.5A digital output channels (IO205)
- 16 DIFF 16-bit analog output channels (IO107)
- 32 SE/16 DIFF 16-bit analog input, 4 SE analog output, 8 TTL digital input, 8 TTL digital output channels (IO102)
- RTD simulation (IO926)
What is xPC Target?

*Workflow*

- A software environment that allows for real-time execution of Simulink models on a separate x86, PCI-based target machine. Automatically:
  1. Generate code,
  2. Compile & Link,
  3. Run executable

- **Workflow Diagram:**
  1. Host Computer with MATLAB & Simulink
  2. C Compiler
  3. Real-Time Target Machine
  4. Ethernet Link

- **xPC Target Real-Time Kernel**
What is xPC Target?

Interactive control and access to the real-time application while it runs

- Live parameter tuning, signal monitoring, and control from the Simulink model,
- Real-time data logging for offline or post-test analysis in MATLAB,
- GUI/HMI support,
- 3-D visualization.
What is xPC Target?

I/O support to communicate with your hardware under test

- Includes Simulink blocks and software drivers supporting a broad suite of I/O devices and communication protocols.
- Blocks are easily configurable within the Simulink model and communicate with the I/O hardware in real-time.
# Fixed-Function I/O Modules

*Powerful “as is” functionality*

<table>
<thead>
<tr>
<th>IO Type</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog</td>
<td>High-resolution, high-speed, simultaneous sampling, BNC and XLR panels, …</td>
</tr>
<tr>
<td>Digital</td>
<td>TTL/LVCMOS, RS422/RS485/LVDS, 06-48V, low/high side, opto-coupled, …</td>
</tr>
<tr>
<td>Serial</td>
<td>RS232, RS422, RS485, SDLC, HDLC</td>
</tr>
<tr>
<td>Ethernet-based</td>
<td>EtherCAT, EtherNet/IP, Modbus TCP, POWERLINK, real-time UDP, …</td>
</tr>
<tr>
<td>Video</td>
<td>CameraLink, USB WebCam</td>
</tr>
<tr>
<td>Audio/Speech</td>
<td>Audio/Speech optimized analog IO modules</td>
</tr>
<tr>
<td>Shared Memory</td>
<td>Reflective Memory for high speed data transfer in multi-processor systems</td>
</tr>
<tr>
<td>Various</td>
<td>LVDT/RVDT, Synchro/Resolver, reed relays, programmable resistors, external signal conditioning modules (current to voltage, voltage to current, temperature, …)</td>
</tr>
</tbody>
</table>

- Delivery includes I/O cables, terminal boards, test models, and Simulink driver blocks
- 3 years of warranty, and long-term availability (7+ years for most I/O modules)
Multi-Function I/O Modules

Reconfigurable to support your application

- Execute high-speed algorithms on an FPGA connected to a model running in real time with xPC Target.
- Automatically program the FPGA without needing to know HDL code
- Quick reconfiguration of FPGA I/O promotes a flexible real-time testing environment.
- Three different use cases supported
  - Pre-configured FPGA Code Module functionality
  - Execute Simulink Applications on FPGA using automatic HDL Code Generation
  - Write and implement your own HDL Code using Speedgoat FPGA Engineering Kits
Run Simulink in Real-time

You want to run, test, and prove your Simulink design with your hardware under test at its normal operating frequency, speed, or timing.

xPC Target Turnkey

- Official fully assembled, real-time testing MathWorks solution for Simulink
- Combines xPC Target (software) with a real-time target machine and IO modules (hardware)

“We received Speedgoat’s real-time target machine in the morning, and in the evening our system under test was already up and running.

That’s how rapid prototyping should be, shouldn’t it?”

M. Feriencik, RUAG Space AG, Switzerland
Hardware Test Setup at MathWorks
Workflow
Model your system, design your controller, and test in real-time
What’s Next

- Introduction and overview
- What is real-time testing?
- How can I use it to build better products?
  - Characterizing a model
  - Rapid Prototyping
  - Hardware-In-the-Loop

Where can I learn more?
Recorded Webinars & Examples

Watch a recorded webinar:

A Simulink Real-Time Testing Solution for Power Electronics & Motor Control
http://www.mathworks.com/company/events/webinars/wbnr68656.html

Prove Your Simulink Designs with Real-Time Hardware Testing
https://www.mathworks.com/company/events/webinars/wbnr73147.html

Explore example models:

Field-Oriented Control of a Permanent Magnet Synchronous Machine
This example shows the basic workflow and key APIs for generating C code from a motor control algorithm, and for verifying its compiled behavior and execution time.

xPC Target
Examples demonstrating features of xPC Target.
(real-time parameter tuning, signal monitoring, data logging, and more)
http://www.mathworks.com/products/xpctarget/examples.html
End

THANK YOU
Backup slides
Need help getting started?

MathWorks® | Training Services

- Product-focused training classes
- Public or customized on-site training

MathWorks® | Consulting Services

- Jump start services get you up and running quickly
- Custom, project-based, application support

Work with your Account Manager to identify topics of interest and customize services to meet your needs.
Have questions? Want to know more about xPC Target or xPC Target Turnkey? Contact your MathWorks Account Manager or Speedgoat.

**xPC Target**
www.mathworks.com/products/xpctarget

**xPC Target Turnkey**
www.mathworks.com/products/xpctarget/supported-hardware/index.html

**Speedgoat**
www.speedgoat.ch
Key Point

- Create accurate plant models by executing tests, identify parameter values and verifying against real-world data
Model your system using MATLAB and Simulink

System Model

Controller Model

Motor Model

Controller C/HDL Code

Motor Hardware

Real-Time Control System
Surface Mount PMSM Equations

**Electrical Model**

\[ v_d = R i_d - L_q p \omega_r i_q + L_d \frac{d}{dt} i_d \]

\[ v_q = R i_q + p \omega_r (L_d i_d + \lambda) + L_q \frac{d}{dt} i_q \]

\[ \omega_e = p \omega_r \]

\[ T_e = 1.5 p [\lambda i_q + (L_d - L_q) i_d i_q] \]

\[ T_e = K_t i_q \text{ (assumes round rotor, } L_d = L_q) \]

**Mechanical Model**

\[ \frac{d}{dt} \omega_r = \frac{1}{H} (T_e - \text{sgn}(\omega_r)J_0 - b \omega_r - T_{load}) \]
Required Parameters

**Electrical Model**

\[ v_d = Ri_d - L_q p \omega_r i_q + L_d \frac{d}{dt} i_d \]

\[ v_q = Ri_q + p \omega_r (L_d i_d + \lambda) + L_q \frac{d}{dt} i_q \]

\[ \omega_e = p \omega_r \]

\[ T_e = 1.5p[\lambda i_q + (L_d - L_q)i_d i_q] \]

\[ T_e = K_t i_q \text{ (assumes round rotor, } L_d = L_q) \]

**Mechanical Model**

\[ \frac{d}{dt} \omega_r = \frac{1}{H} (T_e - sgn(\omega_r)J_0 - b \omega_r - T_{load}) \]
Mapping parameters to PMSM model

- $R$
- $L_d, L_q$
- $\lambda$
- $K_t$
- $H$
- $b$
- $p$
- $J_0$
## Tests to Characterize Motor and Load

<table>
<thead>
<tr>
<th>Motor Tests</th>
<th>Parameters Identified</th>
<th>Identification method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back EMF Test</td>
<td>Number of Pole Pairs ((p))</td>
<td>Calculation</td>
</tr>
<tr>
<td></td>
<td>Flux Linkage Constant ((\lambda))</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Torque Constant ((K_t))</td>
<td></td>
</tr>
<tr>
<td>Friction Test</td>
<td>Viscous Damping Coefficient ((b))</td>
<td>Curve fitting</td>
</tr>
<tr>
<td></td>
<td>Coulomb Friction ((J_0))</td>
<td></td>
</tr>
<tr>
<td>Coast Down Test</td>
<td>Rotor Inertia ((H))</td>
<td>Curve fitting</td>
</tr>
<tr>
<td>DC Voltage Step Test</td>
<td>Resistance ((R))</td>
<td>Parameter estimation</td>
</tr>
<tr>
<td></td>
<td>Inductance ((L))</td>
<td></td>
</tr>
</tbody>
</table>
## Tests to Characterize Motor and Load

<table>
<thead>
<tr>
<th>Motor Tests</th>
<th>Parameters Identified</th>
<th>Identification method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back EMF Test</td>
<td>Number of Pole Pairs ($p$)</td>
<td>Calculation</td>
</tr>
<tr>
<td></td>
<td>Flux Linkage Constant ($\lambda$)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Torque Constant ($K_t$)</td>
<td></td>
</tr>
<tr>
<td>Friction Test</td>
<td>Viscous Damping Coefficient ($b$)</td>
<td>Curve fitting</td>
</tr>
<tr>
<td></td>
<td>Coulomb Friction ($J_0$)</td>
<td></td>
</tr>
<tr>
<td>Coast Down Test</td>
<td><strong>Rotor Inertia ($H$)</strong></td>
<td>Curve fitting</td>
</tr>
<tr>
<td>DC Voltage Step Test</td>
<td><strong>Resistance ($R$)</strong></td>
<td>Parameter estimation</td>
</tr>
<tr>
<td></td>
<td><strong>Inductance ($L$)</strong></td>
<td></td>
</tr>
</tbody>
</table>
# Tests to Characterize Motor and Load

<table>
<thead>
<tr>
<th>Motor Tests</th>
<th>Parameters Identified</th>
<th>Identification method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back EMF Test</td>
<td>Number of Pole Pairs ( (p) )</td>
<td>Calculation</td>
</tr>
<tr>
<td></td>
<td>Flux Linkage Constant ( (\lambda) )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Torque Constant ( (K_t) )</td>
<td></td>
</tr>
<tr>
<td>Friction Test</td>
<td>Viscous Damping Coefficient ( (b) )</td>
<td>Curve fitting</td>
</tr>
<tr>
<td></td>
<td>Coulomb Friction ( (J_0) )</td>
<td></td>
</tr>
<tr>
<td>Coast Down Test</td>
<td>Rotor Inertia ( (H) )</td>
<td>Curve fitting</td>
</tr>
<tr>
<td>DC Voltage Step Test</td>
<td>Resistance ( (R) )</td>
<td>Parameter estimation</td>
</tr>
<tr>
<td></td>
<td>Inductance ( (L) )</td>
<td></td>
</tr>
</tbody>
</table>
Coast Down Test

Coast down test used to identify:
- Rotor inertia (H)

Hardware Test Data

Control \( \omega_{r0} \) → PMSM → Rotor Velocity \( \omega_r \) (rads/sec)
Identify Rotor Inertia (H)

\[ \frac{d\omega_r}{dt} = \frac{1}{H} (T_{em} - b\omega_r - J_0 - T_{load}) \]

If
\[ T_{load} = 0 \]
\[ T_{em} = 0 \]

Then
\[ \omega_r = \left( \omega_{r0} + \frac{J_0}{b} \right) e^{\frac{b}{H}t} - \frac{J_0}{b} \]

**Note:** J₀ and b are known from friction test. Curve fit equation for \( \omega_r \) to find H

NRMSD = Normalized Root Mean Square Deviation
Demo: Run Coast Down Test

Using xPC Target as a real time testing platform:
- Review model
- Build code
- Run model
- That’s it!
Validate Coast Down Test

PMSM Coast Down Testbench

The initial velocity is specified in the motor dialog. The motor coastes down to zero speed.

NRMSD = Normalized Root Mean Square Deviation
# Tests to Characterize Motor and Load

<table>
<thead>
<tr>
<th>Motor Tests</th>
<th>Parameters Identified</th>
<th>Identification method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back EMF Test</td>
<td>Number of Pole Pairs (p)</td>
<td>Calculation</td>
</tr>
<tr>
<td></td>
<td>Flux Linkage Constant (λ)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Torque Constant (Kₜ)</td>
<td></td>
</tr>
<tr>
<td>Friction Test</td>
<td>Viscous Damping Coefficient (b)</td>
<td>Curve fitting</td>
</tr>
<tr>
<td></td>
<td>Coulomb Friction (J₀)</td>
<td></td>
</tr>
<tr>
<td>Coast Down Test</td>
<td>Rotor Inertia (H)</td>
<td>Curve fitting</td>
</tr>
<tr>
<td>DC Voltage Step Test</td>
<td>Resistance (R)</td>
<td>Parameter estimation</td>
</tr>
<tr>
<td></td>
<td>Inductance (L)</td>
<td></td>
</tr>
</tbody>
</table>
Identify Phase Resistance (R)

\[ v_{In}(t) = R_{Limit}i(t) + \hat{R}i(t) + \hat{L}\frac{di}{dt} \]

\[ v_{In}(t) = R_Ti(t) + \hat{L}\frac{di}{dt} \]

\[ i(t) = \frac{V_{In}}{R_T} \left(1 - e^{-\frac{R_T}{L}t}\right) \]

where: \( R_T = (R_{Limit} + \hat{R}) \)

\[ \hat{R} = \lim_{t \to \infty} \left[\frac{v_{In}(t)}{i(t)}\right] \]

\[ R = \frac{\hat{R}}{2} \]

\[ L = \frac{\hat{L}}{2} \]
Identify Inductance (L) using Parameter Estimation
Validate DC Voltage Step Test

PMSM DC Voltage Step Testbench

With the rotor locked, apply a step response across two phase voltages and measure the corresponding current.

NRMSD = Normalized Root Mean Square Deviation

Hardware vs. Simulation

NRMSD = 3.05%
Key Point

- Create accurate plant models by executing tests, identify parameter values and verifying against real-world data
Agenda

What we’ll cover today

- MathWorks Overview

- What is real-time testing?

How can I use it to build a better model?

- Characterize a plant
- **Optimize a controller**
- Verify a system design

- Where can I learn more?
Design your controller through simulation
Design Controller through Simulation

- Select model architecture
- Optimize closed-loop performance
- Test failure modes
Demo: check control response against requirements
Demo: Tune Controller using Optimization

Control Velocity of Permanent Magnet Synchronous Machine With Disc Load

Model Description: Control Velocity of Permanent Magnet Synchronous Machine With Disc Load

Demonstrates a Field-Oriented Control algorithm with Space Vector Modulation for a Permanent Magnet Synchronous Machine (PMSM).

The test bench can be used to evaluate the system performance. Examples include turning the motor on, searching for a valid rotor position, transitioning to closed loop operation, and changing speed and torque during closed loop control.

The Embedded Processor subsystem contains the controller algorithm (which supports C code generation) as well as simulation models of peripherals.
Demo: check optimized controller response
Design Controller through Simulation

- Select model architecture
- Optimize closed-loop performance
- Test failure modes
Agenda
What we’ll cover today

- MathWorks Overview

- What is real-time testing?

How can I use it to build a better model?
- Characterize a plant
- Optimize a controller
  - Verify a system design

- Where can I learn more?
Test your controller model with actual motor

Real-Time Control System

System Model

Controller Model

Motor Model

Controller C/HDL Code

Motor Hardware
Test and Verify your design

System Model

Controller Model

Motor Model

Controller C/HDL Code

Motor Hardware

Real-Time Control System
Demo: Comparison of Simulation and Hardware Results

Hardware vs. Simulation
NRMSD = 1.67%

Hardware vs. Simulation
NRMSD = 0.818%
Agenda
What we’ll cover today

- MathWorks Overview
- What is real-time testing?
- How can I use it to build a better model?
  - Characterize a plant
  - Optimize a controller
  - Verify a system design

Where can I learn more?
Product Details & Contact Information

Have questions? Want to know more about xPC Target or xPC Target Turnkey? Contact your MathWorks Account Manager or Speedgoat.

xPC Target
www.mathworks.com/products/xpctarget

xPC Target Turnkey
www.mathworks.com/products/xpctarget/supported-hardware/index.html

Speedgoat
www.speedgoat.ch
Recorded Webinars & Examples

Watch a recorded webinar:

A Simulink Real-Time Testing Solution for Power Electronics & Motor Control
http://www.mathworks.com/company/events/webinars/wbnr68656.html

Prove Your Simulink Designs with Real-Time Hardware Testing
https://www.mathworks.com/company/events/webinars/wbnr73147.html

Explore example models:

Field-Oriented Control of a Permanent Magnet Synchronous Machine
This example shows the basic workflow and key APIs for generating C code from a motor control algorithm, and for verifying its compiled behavior and execution time.

xPC Target
Examples demonstrating features of xPC Target.
(real-time parameter tuning, signal monitoring, data logging, and more)
http://www.mathworks.com/products/xpctarget/examples.html
Need help getting started?

**MathWorks® | Training Services**

- Product-focused training classes
- Public or customized on-site training

**MathWorks® | Consulting Services**

- Jump start services get you up and running quickly
- Custom, project-based, application support

Work with your Account Manager to identify topics of interest and customize services to meet your needs.