Modellbasiertes Echtzeittesten und automatische Codegenerierung
Agenda

- Introduction to Model-Based Design
- Rapid Control Prototyping
- Processor in the Loop – PIL
- Hardware in the Loop - HIL
Model-Based Design
Development Process

Requirements

System Design
- Environment
- Physical Components
- Algorithms

Verification and Validation

Component Design

Research
- Data Analysis
- Algorithm Development
- Data Modeling

Rapid Control Prototyping

System-Level Specification

Integration testing

User Acceptance Testing

Complete Integration & Test

System-Level Integration & Test HIL

Subsystem Integration & Test SIL/PIL

Code Verification and Validation

Subsystem Implementation

Implementation
- Embedded Software
- Digital Electronics
- DSP
- FPGA
- ASIC
- MCUC, C++, VHDL, Verilog

Environment
Physical Components
Algorithms

Component Design

Generate

System Design
Model-Based Design
Development Process

Requirements

System Design
- Environment
- Physical Components
- Algorithms

Component Design

Verification and Validation

Integration & Test

Code Verification and Validation

Research
- Data Analysis
- Algorithm Development
- Data Modeling

System-Level Specification

Rapid Control Prototyping

Complete Integration & Test

User Acceptance Testing

System-Level Integration & Test

HIL

Subsystem Integration & Test

SIL/PIL

Implementation

Subsystem Implementation

Implementation

Embedded Software
- C, C++

Digital Electronics
- VHDL, Verilog

Integration
- MCU, DSP, FPGA, ASIC
Model-Based Design

Rapid Prototyping

Requirements

System Design
- Environment
- Physical Components
- Algorithms

Component Design

Verification and Validation

Code Verification and Validation

User Acceptance Testing

Integration & Test

Complete Integration & Test

System-Level Integration & Test

HIL

Subsystem Integration & Test

SIL/PIL

Research
- Data Analysis
- Algorithm Development

Data Modeling

Rapid Control Prototyping

Embedded Software
- C, C++

Digital Electronics
- VHDL, Verilog

MCU
DSP
FPGA
ASIC

Implementation

Subsystem Implementation
Why do Rapid Prototyping?

- Test, verify, validate, and prove your design with hardware under test
- Evaluate new ideas using a production independent development platform
- Reduce costs, shorten time-to-market, and minimize risks
Rapid Controller Prototyping

Controller Model

Hardware (Plant/System)

Verification

HOST

TARGET

DEVICE
XPC Target workflow

Real-time execution of Simulink models
Demo – Rapid Control Prototyping
Rapid Prototyping

- Prove your design
- Evaluate new ideas
- Reduce costs
- Shorten time-to-market
- Minimize risks
Model-Based Design
Automatic Code Generation

Requirements

System Design
- Environment
- Physical Components
- Algorithms

Component Design
- Rapid Control Prototyping

System-Level Specification

Verification and Validation

Integration testing
- Code Verification and Validation
- User Acceptance Testing
- System-Level Integration & Test
- Complete Integration & Test

Research
- Data Analysis
- Algorithm Development
- Data Modeling

Implementation
- Embedded Software
  - C, C++
- Digital Electronics
  - VHDL, Verilog
- MCU
- DSP
- FPGA
- ASIC

Subsystem Implementation
- SIL/PIL Integration & Test

MATLAB EXPO 2013
Model-Based Design
Automatic Code Generation

• C/C++, VHDL/Verilog and PLC Code
• Support for Fixed Point Data Format
  • Automatic scaling
  • Supported in Simulation and Code
• Easy integration of legacy C/C++ Code
• System development independent of the target
Demo – Code Generation
Model-Based Design
Continuous Verification and Validation

Requirements

System Design
- Environment
- Physical Components
- Algorithms

Component Design

Research
- Data Analysis
- Algorithm Development
- Data Modeling

Rapid Control Prototyping

Verification and Validation

Integration testing

Code Verification and Validation

System-Level Integration & Test

User Acceptance Testing

Complete Integration & Test

Subsystem Integration & Test
- SIL/PIL

System-Level Integration & Test
- HIL

Subsystem Implementation

Code Implementation
- Embedded Software
- Digital Electronics
  - C, C++
  - VHDL, Verilog
- Integration
  - MCU
  - DSP
  - FPGA
  - ASIC

Implementation

MATLAB EXPO 2013
Why do Processor in the Loop (PIL)?

- Verify numerical output of code
- Profiling
  - Execution
  - Coverage
  - Stack
- Verify behaviour of target specific code
- Investigate compiler settings and optimizations
Use PIL Simulation to Verify

- Simulink
  - Test Signals
  - Embedded Processor
    - Serial / TCPIP
      - PIL Implementation
      - Controller Model
        - Code Generation
        - Controller C Code
      - Serial / TCPIP
        - Verifications
      - PIL Implementation
        - Controller C Code
      - Code Generation
      - Controller Model
      - Test Signals
Demo - PIL

Controller Algorithm for Permanent Magnet Syncl

Copyright 2010-2012 The MathWorks, Inc.

Model Description: Controller
Specifications:
- The controller outputs control signals
- The sensors bus/structure

Embedded Processor

Models hardware on the embedded processor, including the controller algorithm software specification and peripherals.

Motor Control

Mode Scheduler

Position

Motor

Command

On

Type

Value

Sensors

System Inputs

Signals

Difference

MATLAB EXPO 2013
Why do Processor in the Loop (PIL)?

- Verify numerical output of generated (or legacy) code
- Coverage, Execution and Stack profiling
- Verify behavior of target specific code
- Investigate effects of compiler settings and optimizations
Processor in the Loop (PIL)

- Verify numerical output of code
- Profiling
- Verify target specific code
- Investigate compiler settings and optimizations
Hardware in the Loop

Test Suite → Hardware Under Test → Plant Model → Verification
Why do Hardware in the Loop (HIL)?

- Substitute for unavailable parts of the system
- Test the system for safety and performance
- Minimize expensive downtime for the rest of the system
- Test operation and failure conditions that are difficult to replicate
Conclusion

- Model-Based Design
  - Core of the Development Process

- Rapid Control Prototyping
  - Fast Evaluation of Design

- Processor in the Loop – PIL
  - Early Test and Verification

- Hardware in the Loop – HIL
  - System level integration test
Model-Based Design

Requirements

System Design
- Environment
- Physical Components
- Algorithms

Component Design

Verification and Validation

Integration Testing

Code Verification and Validation

User Acceptance Testing

Complete Integration & Test

System-Level Integration & Test HIL

Subsystem Integration & Test SIL/PIL

Research
- Data Analysis
- Algorithm Development
- Data Modeling

System-Level Specification

Rapid Control Prototyping

Implementation

Embedded Software
- C, C++

Digital Electronics
- VHDL, Verilog

Integration
- MCU, DSP, FPGA, ASIC

DSP, FPGA, ASIC