MathWorks® | Training Services
Accelerating the pace of engineering and science

Applying Model-Based Design for ISO 26262

Training Objectives

This five-day course describes guiding principles for applying Model-Based Design to meet ISO 26262 and IEC 61508 compliance for safety-related software development. It enables users to take advantage of the Simulink® environment to synthesize, implement, and validate their software components in a manner consistent with the principles of functionals safety guidelines. Topics include:

- Design and implement modular software using Simulink subsystems, libraries, and models.
- Manage traceability between requirements, architecture, subsystems, tests, and code.
- Practice early verification and validation during software development using model-based and code-based testing.
- Establish and enforcing software standards across all stages in the development process.
- Streamline tool qualification using the IEC Certification Kit (for ISO 26262 and IEC 61508).

Prerequisites

MATLAB Onramp and Simulink Onramp. This course is intended for intermediate or advanced Simulink users. Knowledge of C programming language is recommended. Knowledge of the ISO 26262 standard or IEC 61508 standard is recommended.

Products

- Simulink
- Requirements Toolbox™
- Simulink Test™
- Simulink Coverage™
- Simulink Check™
- Simulink Design Verifier™
- System Composer®
- Embedded Coder®
- Simulink Report Generator™
- Polyspace Bug Finder®
- Polyspace Code Prover®
- IEC Certification Kit (for ISO 26262 and IEC 61508)

Course Outline

SLIS1223V2

Day 1 of 5

Overview of ISO 26262 and Model-Based Design (2.0 hrs)

Objective: Get an overview of ISO 26262 and its role in the automotive industry. Discuss MathWorks' involvement and level of support within this standard.

- ISO 26262 standard
- Model-Based Design overview
- Reference workflow

Project Management (1.0 hrs)

Objective: Organize project files (models, data, documentation). Familiarize with the project environment.

- Project setup
- File shortcuts and labels
- File dependency analysis

Model Creation (3.0 hrs)

Objective: Create and simulate a Simulink model for algorithm development. Manage model data using data dictionaries.

- Simulink environment
- Discrete-time models
- Sample time
- Simulation and analysis
- Data dictionary
- Solver selection

Model Compliance (1.0 hrs)

Objective: Explore how to set up and enforce modeling standards and check for common modeling errors.

- Modeling standards
- Edit-time checks
- Model Advisor
- Results reporting

Day 2 of 5

Requirements Management (1.5 hrs)

Objective: Link a Simulink model to software requirements.

- Requirement sets
- Requirements import

- Requirements linking

Software Unit Verification (2.5 hrs)

Objective: Create time-based and logic-based test cases for a Simulink model.

- Types of verification
- Design error detection
- Test harness creation
- Test inputs
- Logic in tests
- Requirement-based assessments

Code Generation for Software Unit (3.0 hrs)

Objective: Generate code for a software unit. Customize the generate code to optimize data storage and execution.

- Code generation for step function
- Function prototypes
- Data storage optimization
- Data types and storage classes
- Data objects
- Function templates

Day 3 of 5

Subsystems (2.5 hrs)

Objective: Create functional partitioning within a software unit using subsystems. Package subsystems into library blocks for reuse. Create partitions in the generated code.

- Subsystems
- Variant subsystems
- Subsystem references
- Masks
- Libraries
- Subsystem code generation

Multirate Modeling (2.0 hrs)

Objective: Introduce rate-based and export function modeling approach. Handle rate transition between rates.

- Block execution
- Single-rate systems
- Multirate systems
- Rate transitions
- Export function models

Architecture Modeling (2.5 hrs)

Objective: Create a software architecture model using System Composer. Analyze the software architecture and link to behavioral model.

- Architecture model
- Profiles and stereotypes
- Interface Editor
- Views
- Behavioral model linking

Day 4 of 5

System Integration (3.0 hrs)

Objective: Organize software units into an integration model using model referencing. Configure model settings and data dictionaries so they can be shared across different models in the integration stage.

- System component considerations
- Model references
- Referenced data dictionaries
- Referenced configuration sets
- Code generation for integration model
- Model workspace

In-the-Loop Testing (1.5 hrs)

Objective: Testing and verification of the generated code using in-the-loop testing techniques.

- Software-in-the-loop testing
- Code profiling
- Model reference software testing
- Processor-in-the-loop testing

Verification Automation (2.5 hrs)

Objective: Create repeatable groups of tests and automatically generate reports from the test results.

- Test files
- Model coverage
- Code coverage
- Automatic test generation
- Test results reporting

Day 5 of 5

Code Verification (1.0 hrs)

Objective: Perform static analysis on the generated code to ensure the code is compliant with MISRA C:2012.

- Code verification using Polyspace Bug Finder
- Software MISRA C:2012 compliance
- Code metrics

Reporting (1.5 hrs)

Objective: Discuss the methods of automatically creating reports and documentation from Simulink models. Discuss configuration management methods in the project environment.

- Model Testing Dashboard
- Web views
- Standard reports
- Source control integration
- File differences

Tool Qualification (1.5 hrs)

Objective: Use the IEC Certification Kit (for ISO 26262 and IEC 61508) to qualify MathWorks tools to meet compliance with ISO 26262

- Tool qualification
- IEC Certification Kit (for ISO 26262 and IEC 61508)

Case Study (3.0 hrs)

Objective: Apply Model-Based Design to implement a control algorithm to showcase the reference workflow.

Appendices

Overview of IEC 61508 and Model-Based Design

Summary: Review how IEC 61508 is applied to software development. Discuss how Model-Based Design workflow aligns with the functional safety development framework.

- IEC 61508 standard
- Model-Based Design overview
- Reference workflow