WHITE PAPER

Enabling Model-Based Design for DO-254 Certification Compliance

MathWorks®

Engineers can use Model-Based Design for requirements analysis, algorithm design, automatic HDL code generation, and verification, to produce airborne electronic hardware that adheres to the DO-254 standard. The proposed Model-Based Design approach for DO-254 combines tools from MathWorks® and Siemens® EDA for both design and verification. This workflow supports development phases from concept through implementation, streamlining development and reducing costs.

Simulink® from MathWorks is the starting point to enable Model-Based Design within this process. Simulink allows engineers to manage requirements, and test sets, develop architectural and behavioral models, and perform formal verification, conformance to modeling standards, and code generation and verification for VHDL® and Verilog®.

This approach delivers two main benefits:

- Development teams identify errors earlier in the design process, when they can be fixed with less impact on product schedules and costs, rather than finding them during implementation and testing.
- Development teams can reuse designs, tests, and analyses throughout the development process and share them among team members.

Model-Based Design promotes a requirements-oriented project view and greater integration and reusability among conceptual design, detailed design, and implementation. In this article, we discuss the workflow for Model-Based Design, explain the types of activities performed in each environment, and highlight how the tools can be combined to maximize reusability and efficiency.

DO-254 Overview

DO-254 Compliance and Life Cycle

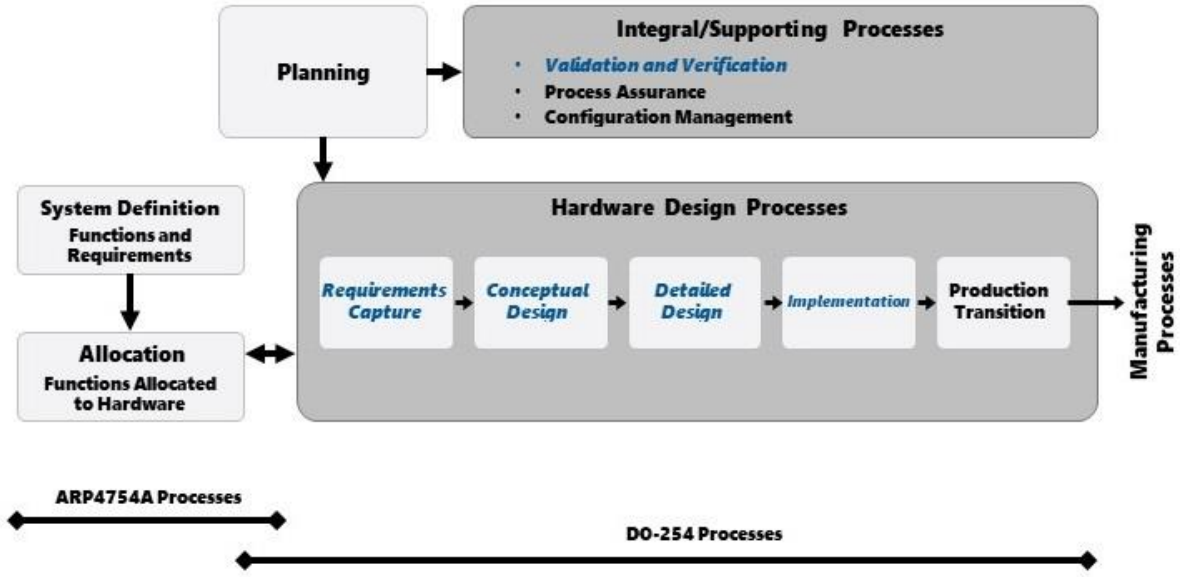

Figure 1 DO-254 compliance lifecycle and associated processes
*(Note: **Blue italics** indicate process phases covered in this paper.)*

Figure 1 shows the DO-254 life cycle and lists the processes that must be performed and documented as a design moves from phase to phase. We will focus on the following processes:

- Requirements Capture: authoring, management, and tracing
- Conceptual Design as Simulink models
- Detailed Design as HDL code
- Conformance to design standards at model level
- Validation and Verification

Overview of a DO-254 Workflow Using Model-Based Design Techniques

In this workflow, engineers use Siemens EDA Polarion to collect and manage requirements, which are exported to Requirements Toolbox™ from MathWorks. An executable Simulink model is then created from these requirements to enable conceptual design exploration. This conceptual model links to requirements at different levels in Polarion™ and Requirements Toolbox.

Using verification and validation tools from MathWorks, engineers perform functional testing and formal analysis at the conceptual model level. In Simulink, engineers elaborate the model by adding implementation attributes such as data-streaming and fixed-point effects. This elaborated model allows engineers to verify the design meets requirements, determine the level of coverage, and check conformance to model standards, becoming the model for HDL implementation. A detailed design in HDL is also generated from the verified Simulink model using HDL Coder™. SystemVerilog testbench components are generated using HDL Verifier™.

Further verification of the detailed HDL design is performed in combination with Siemens EDA verification tools. Using HDL cosimulation with HDL Verifier, a Simulink testbench is used with a design-under-test (DUT) simulated in the Questa™ simulator to verify the DUT correctly implements the model. Test vectors created at the conceptual model level with Simulink Test™ are applied to the Simulink testbench during cosimulation. HDL code coverage is measured in Questa to determine the effectiveness of the test vectors and compared to the coverage metrics collected at model level.

The entire Simulink testbench is then exported to the Siemens verification environment by generating SystemVerilog DPI-C components representing the stimulus, reference model and checker. HDL Verifier also generates either individual SystemVerilog verification components or complete UVM verification environments.

Siemens EDA products support additional HDL development, code checking, coverage closure, code visualization, and review. Questa Formal performs static design and automated coverage analysis, while Siemens FormalPro™ performs logical equivalency for model checking. Questa CDC and RDC perform clock and reset domain crossing checks for metastability and glitch scenarios, and Questa Lint compliments domain crossing analysis with a DO-254 lint ruleset. FPGA synthesis and integration with FPGA vendor place and route tools is accomplished using Precision™ RTL. The tools and flow described above is shown in Figure 2.

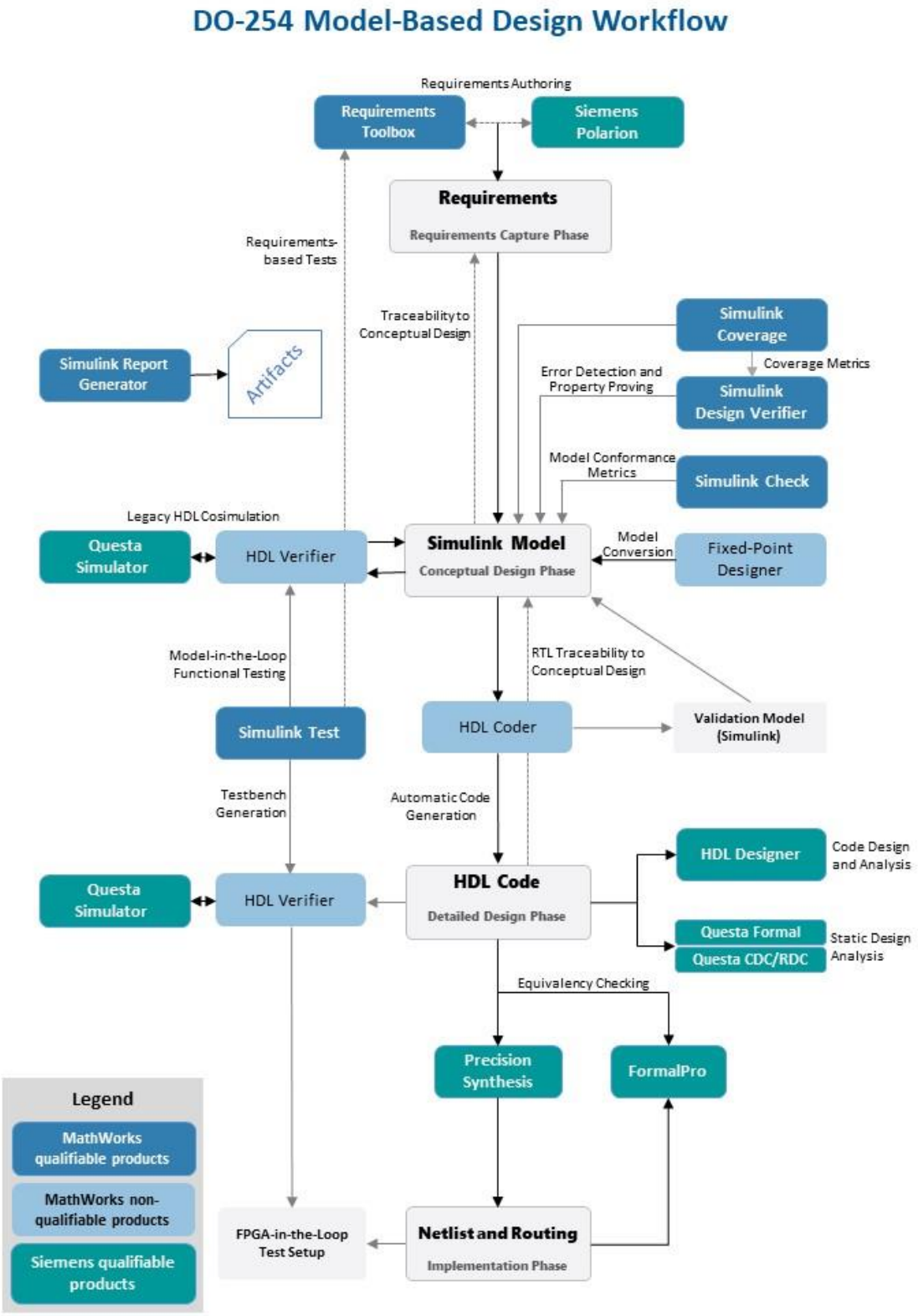

Figure 2. DO-254 workflow with Model-Based Design.

Requirements Capture

Companies serving the aerospace market use enterprise-level requirements management tools such as Siemens Polarion ALM™. Design and verification tasks link back to these requirements through requirements tracing. A requirements-driven design flow requires entering requirements, tracking changes to requirements, and linking to design and verification artifacts.

The integration of Polarion and Requirements Toolbox establishes traceability among design elements and verification artifacts at model and code levels. It also validates requirements by facilitating requirements reviews, offering coverage information, guiding verification activities based on requirements status, and providing certification artifacts. Polarion and Requirements Toolbox integrate with HDL Coder from MathWorks as well as Siemens tools for HDL development, verification, and synthesis, and are flexible enough to adapt to other tools used in a DO-254 development processes. Designers link requirements information to specific blocks, subsystems, and entire models. This information is then automatically passed through to HDL code generated by HDL Coder (see "Detailed Design" section below).

In addition to traceability and validation support, these requirements tools assist project management by creating a visual depiction of project status and generate traceability matrices required to meet DO-254 objectives.

Conceptual Design

Once requirements are firmly established, the next step is for a design engineer to develop a conceptual design that achieves the requirements captured in the previous phase. This section discusses how Simulink and other MathWorks model-level tools are used to develop and verify the conceptual design.

Conceptual Model Design

Simulink is widely used for designing, implementing, and verifying aerospace systems. Using Simulink, design engineers build up algorithmic models in a graphical manner, constructing executable designs linked to captured requirements.

Figure 3 shows an aircraft control system developed in Simulink using Model-Based Design. Stateflow®, companion product to Simulink, was used to model switching between the control system's operating modes. The control system algorithm was then integrated into a larger Simulink system-level model. Another companion product to Simulink, System Composer™, was used to specify software and hardware architectures, perform and analyses, and allocate requirements. The complete system-level aircraft simulation model incorporates the mode logic algorithm, a six-degree-of-freedom airframe model with environmental effects, as well as models for sensors and actuators.

Having a system-level model enables engineers to test their designs earlier in the process and quickly evaluate what-if scenarios. In this example, the system-level model allows an engineer to test the control laws under varying conditions, subject it to sensor failures, or inject a range of pilot inputs.

We can then elaborate the model to specify architecture and incorporate implementation effects. Simulink enables these effects to be simulated and compared to a reference design using ***back-to-back testing***.

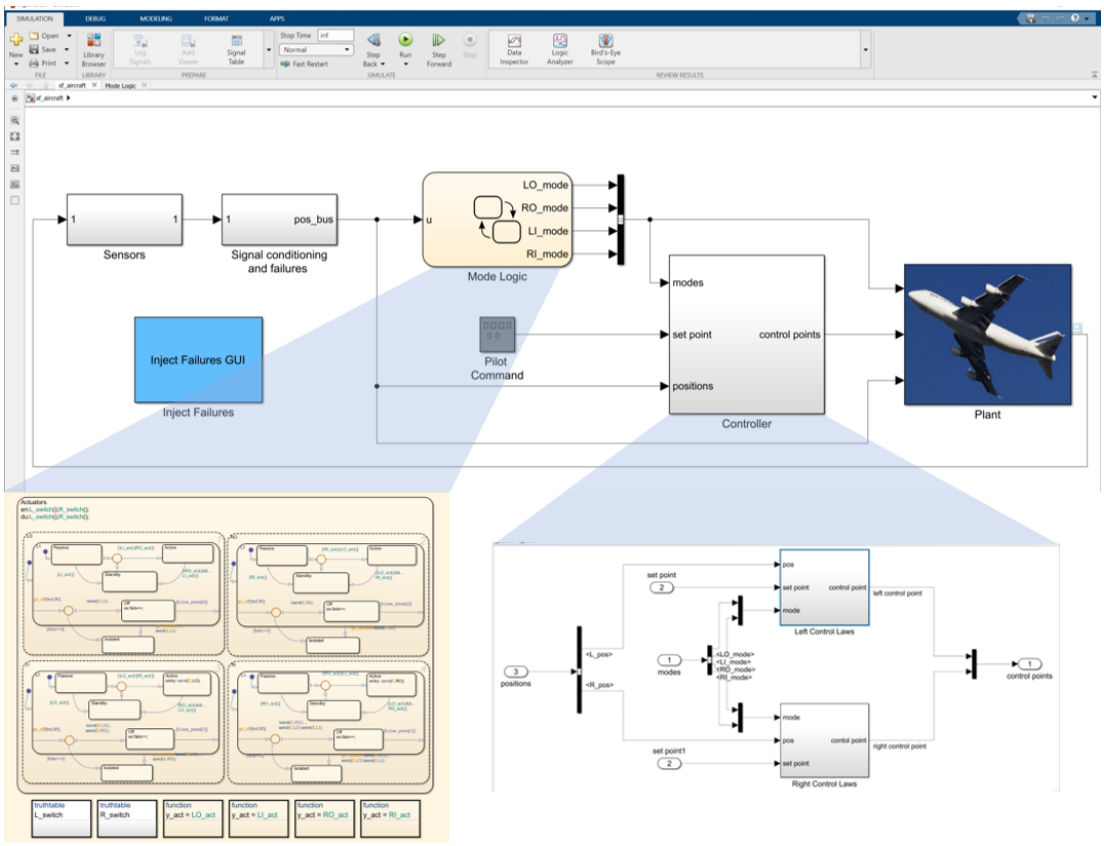

Figure 3. Aircraft control model including Simulink and Stateflow.

Traceability in the Conceptual Model

In Model-Based Design, all elements of the conceptual design should be traceable to the requirements. MathWorks and Siemens provide traceability support via Requirements Toolbox and Polarion. This traceability is preserved in the generated HDL in the form of comments that link to the Simulink source and extends throughout HDL analysis and testing.

Verifying the Conceptual Model

The conceptual design can then be analyzed to verify whether it meets requirements with these techniques:

- MATLAB® can be used to execute scripts, conduct parameter sweeps, and perform analysis on simulation outputs.
- Simulink Test is used to create and execute tests of the Simulink model. Tests are authored to demonstrate that specific functional requirements are being satisfied.
- Simulink Report Generator™ is used with Simulink Test to generate artefacts and Simulink Coverage™ to generate tests and requirements coverage reports. These artefacts can be used for certification evidence.

The generated tests are reused later in the design process. Simulation helps validate that requirements are satisfied by enabling a design to be exercised over a range of conditions. To ensure a complete functional test coverage, formal analysis is used in conjunction with

simulation to generate test cases and to perform property proving. These techniques use mathematically rigorous procedures to search through all model's possible execution paths to find test cases and counter examples.

This systematic analysis provides deeper understanding of the behavior of designs. For example, consider the algorithm discussed above. Typically, this type of logic in software or hardware involves sensor inputs such as airspeed, acceleration, and pilot input. Using formal property proving, an engineer can use formal methods to verify a certain system behavior: e.g., "Prove to me that this logic will never engage if the airspeed and acceleration are within certain ranges". Simulink Design Verifier™ allows a developer to define these mission-critical properties and prove that certain scenarios cannot happen under any conditions at model level.

Throughout testing, model coverage is a metric to assess how fully tests are exercising a model. Simulink Coverage can track and report on model coverage. Although functional tests are used to ensure that design requirements are met, they often do not exercise 100% of the design. Simulink Design Verifier™ uses formal methods to generate test cases that complement functional tests and ensure 100% coverage (using modified condition/decision coverage methods) at the model level.

If test cases do not achieve 100% coverage, that is an indication that additional requirements are needed, that some design elements are redundant, or that a design is inherently difficult to test. Significant savings may be realized by fixing these errors in the conceptual design phase. Testing must also be exercised on HDL and later stages of the design, but the test cases generated on the conceptual model can be reused in HDL testing.

Conforming to Conceptual Model Design Standards

Development of and adherence to design and coding standards is required by DO-254. Conceptual design standards can be developed and applied to the Simulink model. Modeling standards are equivalent to coding standards and can dictate functional aspects of the model. The Model Advisor within Simulink executes pre-packaged sets of model checks, and Simulink Check™ enables customization and deployment of these checks within an organization. Simulink Check includes a DO-254 set of rules for enforcement of style adherence.

These static checks cover areas such as settings, data types, code generator settings, and HDL settings. This process can detect simple mistakes, such as a missing connection for a block input or output, but can also detect more complex issues, such as block settings that may result in an overflow in a fixed-point operation.

Detailed Design

The detailed design process begins at the HDL (or RTL) stage of development. This development can be handwritten or automatically generated with HDL Coder. Automatically generating RTL can increase efficiency by reducing hand coding and enabling faster design iterations; however, verification activities discuss here may be used whether hand coding or RTL generation is used.

Generated HDL from the Conceptual Model

Generated HDL code is read into HDL Designer™ from Siemens for independent assessment and integration with existing HDL or HDL authored by other means. Within HDL Designer, code

is examined via code reviews, automatically checked against HDL coding standards, and visualized for ease of understanding. HDL Coder also generates scripts for HDL Designer to perform linting on generated HDL code.

Traceability in Detailed Design

Conformance of the detailed design to HDL coding standards must also be demonstrated. HDL Coder offers a variety of ways to customize the generated code to comply with process requirements. HDL Designer has an integrated design rules checking engine that includes a DO-254 rule set (as specified in Order 8110-105) and automates code checks.

Reviewing the Code

HDL code must be examined to ensure that it conforms to HDL coding standards and correctly implements the required functionality. The HDL Designer rules-checking engine ensures conformance to HDL standards. HDL Coder and HDL Designer assist with reviewing HDL code to ensure that it implements the required functionality:

- The Code Generation report of HDL Coder facilitates bidirectional navigation between HDL code, the Simulink conceptual model, and requirements. The graphical conceptual model in conjunction with the generated HDL code helps reviewers understand and analyze designs. A traceability report is also generated to aid in this review process.
- HDL Designer helps facilitate code reviews by providing features to visualize the HDL code. These visualizations – along with the rules checking results, examination of the requirements links, and functional verification – provide independent output assessment of the generated code.

Verifying the HDL Model

Verification at the detailed design level is required for handwritten code and generated code. As with the conceptual model, both simulation-based and formal verification techniques are employed. Verification of the detailed design compared to the conceptual design is performed using HDL Verifier™, either through cosimulation of Simulink with Questa or generation of SystemVerilog DPI-C components or UVM environments for use with Questa.

Questa simulates HDL designs with an emphasis on debugging. It also provides built-in code coverage analysis in support of the DO-254 elemental analysis method for level A/B designs. Questa supports a range of advanced verification capabilities:

- SystemVerilog, System C and PSL support
- Transaction-level modeling
- Constrained-random testing
- Object-oriented programming (OOP) techniques for test-bench creation
- Automated test stimulus
- Dynamic assertion-based verification, including an assertion debugger
- A unified coverage database (UCDB) that has been accepted as an Accellera standard
- A verification management environment for ease of managing and reporting on project verification activities

Questa also has a deep integration with Polarion for round-trip traceability from verification artifacts stored in UCDB to requirements. These advanced verification methods aid in verifying devices of substantial complexity that contain concurrent behaviors, making Questa suitable for ASICs and large FPGAs.

Reuse of Test Cases from Conceptual Design

Simulink is the simulation engine during the conceptual design phase, while Questa is the simulation engine for the detailed design phase. The two simulators are linked through HDL cosimulation and HDL testbench generation.

HDL Verifier from MathWorks enables tests authored in the MATLAB, Simulink, and Simulink Test environments to be executed against HDL code simulated in Questa. HDL cosimulation lets engineers reuse Simulink test cases and analysis routines developed during conceptual design to ensure that the specification model and HDL are functionally equivalent.

The aircraft control model discussed earlier was designed and simulated in Simulink as part of a larger system-level aircraft model. From this Simulink conceptual model, an HDL detailed design of the algorithm was generated using HDL Coder. HDL Verifier allows the designer to run the system-level model and tests from the conceptual design against the generated HDL running in Questa, where HDL code coverage analysis can be performed.

HDL Verifier also includes the ability to export Simulink testbenches as SystemVerilog DPI-C components; UVM environments can also be generated from Simulink models. Exporting verification components in this manner enables reuse of Simulink testbench components when hardware engineers do not have access to Simulink.

Both cosimulation and testbench export promote test case reuse and enable engineers to quickly test the HDL design, reducing design iteration time. They also allow engineers to use the analysis capabilities of both Simulink and Questa.

Advanced Analysis in Detailed Design

Reuse achieved through HDL cosimulation and testbench generation is well suited for functional testing. Siemens EDA provides these additional analyses.

Clock-Domain Crossing Analysis: Today's ASICs typically consist of multiple asynchronous clock and reset domains. Designers must implement dedicated hardware to correctly move data from one domain to another to avoid metastability and glitches. Clock domain crossing problems are extremely difficult and expensive to debug and fix because they may not be detected until a failure occurs in hardware. Questa CDC and Questa RDC deliver advanced structural and formal analysis capabilities at multiple points through the design lifecycle to ensure correct clock and reset domain crossing. Using Questa CDC/RDC on designs with two or more asynchronous clock or reset domains helps reduce the likelihood of metastability.

Formal Verification (HDL Model Checking): Proving safety-critical properties may be done in HDL just as in models. Model checking – with HDL as the "model" – exhaustively proves that a design performs its intended function [this is mentioned in DO-254 Appendix B as an acceptable method of advanced verification for level A/B devices]. Questa Formal Autocheck identifies scenarios such as combinatorial loops, FSM deadlocks, arithmetic overflows, and more. Questa Formal Covercheck complements Autocheck and root causes code coverage gaps.

HDL Coding Rules: DO-254, complimented by FAA Order 8110.105A stipulates that HDL coding standards must be enforced for DAL A/B designs and is considered best practice for all designs. Siemens EDA, in collaboration with the United States and European DO-254 user groups, defines a set of coding standards as the foundation to assess HDL design quality. This ruleset is available within Questa Lint, Siemens EDA next generation Lint solution.

Synthesizing the HDL

Logic synthesis – the transformation of RTL code into a technology-based netlist – is central to PLD, FPGA, and ASIC design. While synthesis tools generally focus on timing, design area, power consumption, and tool run time, military and aerospace applications require synthesis tools to consider additional aspects.

Precision RTL from Siemens EDA is an FPGA-vendor independent synthesis product that balances aspects of safe synthesis with conventional goals, ensuring that circuitry intended for proper operation, such as specialized reset circuitry and special state machine encoding, are preserved during synthesis. It also supports the DO-254 principle of repeatability, providing a means to generate a deterministic and repeatable netlist given a consistent environment and conditions. Finally, Precision RTL is integrated with the FormalPro™ logical equivalency checking tool for analysis of the generated netlist.

Precision RTL is integrated FPGA vendor software that performs placement and routing of the netlist into the physical device and can directly launch these tools from the Precision RTL environment.

Verifying the Netlist

Verification is needed at each phase in the DO-254 life cycle to ensure that the design meets requirements and matches the previous version. This design assurance is paramount, especially for DO-254 Level A/B designs. Appendix B of DO-254 states: “As the design assurance level increases, the approach needed to verify that a given design meets its safety requirements may need overlapping, layered combinations of design assurance methods”. There are several ways to perform this verification on the post-synthesis gate-level design.

Static Timing Analysis*:* Precision RTL performs internal static timing analysis as a part of the synthesis process. The analysis is an estimate since it precedes actual physical placement of the netlist. During the place and route process, FPGA vendor tools such as AMD-Xilinx® Vivado®, Intel® Quartus® or Microchip Libero® run final timing analysis once physical placement is complete. ASIC implementation tools also have static timing analysis built in, but signoff analysis is typically performed using standalone tools.

Gate-Level Simulation (GLS) with Timing*:* Questa supports verification of the gate-level netlist. Verification can be done at the output of synthesis with timing estimates or by including the final timing information back-annotated from the place and route procedure. In either case, the HDL testbench environment used during HDL simulation is used to perform gate level simulations. Comparing and contrasting HDL and GLS design models provide a means of independent assessment. HDL Verifier also supports cosimulation at this level of design.

Logical Equivalency Checking: Repeating functional verification at the gate level is generally accepted as the means to validate synthesis results and to verify the results of HDL simulation. This repetition can be incredibly time consuming for large and complex designs, so a faster

approach for verifying synthesis results is formal logical equivalency checking. Siemens FormalPro compares HDL implementations to determine whether they are functionally equivalent. This comparison is typically done on the input and output of a process.

For example, FormalPro compares the RTL used for synthesis with the generated netlist to determine whether they are functionally equivalent. This same process is used to compare the input to place and route (i.e., the synthesized netlist) with the output of place and route. Use of these formal methods enables faster verification than gate-level simulation.

***FPGA-in-the-Loop Testing**:* FPGA-in-the-loop testing is a technique for verifying that a netlist programmed into an actual FPGA device conforms to the conceptual design model in Simulink. HDL Verifier automates the process of connecting a testbench executing in a Simulink session on a host computer to a DUT that has been programmed into an FPGA development board, with communication between the host and board via PCI-Express, Ethernet or JTAG. When used with HDL Coder, this entire process of synthesis, place and route, programming and setting up host/board communication is automated. FPGA-in-the-loop is available for boards with FPGAs from AMD-Xilinx, Intel, and Microchip Technology.

Implementation and Production Transition

The DO-254 workflow using Model-Based Design that was discussed in this paper has centered on the requirements capture, conceptual design, and detailed design phases of development. DO-254 compliance entails a broader scope of activities including implementation, such as programming the FPGA device, and production transition (handing-off of the data and artifacts required to produce a repeatable, identical final hardware item). The design and verification artifacts outlined above are reused in these phases. A detailed discussion of these phases is beyond the scope of this paper.

Summary and Conclusion

Widening use of DO-254 is compelling companies to evaluate how they can be more efficient in their development processes while achieving DO-254 compliance. The use of products from MathWorks and Siemens EDA in design, test, implementation, and verification can address these needs for companies developing complex airborne electronics. A DO-254 workflow using Model-Based Design promotes a consistent requirements-oriented project view and increases reuse of design and verification efforts throughout all phases of the DO-254 life cycle.

Learn More

- Model-Based Design for DO-254 (Overview)
- FPGA, ASIC, and SoC Development (Overview)
- How to Use Model-Based Design to Demonstrate DO-254 Compliance (Video)
- Verifying Algorithms on FPGAs and ASICs (Ebook)