Enabling Model-Based Design for DO-254 Compliance with MathWorks and Mentor Graphics Tools

The purpose of RTCA/DO-254 (referred to herein as "DO-254") is to provide guidance for the development of airborne electronic hardware. The Federal Aviation Administration (FAA), European Aviation Safety Agency (EASA), and other worldwide aviation safety authorities require this standard to ensure that complex electronic hardware used in aircraft systems works as specified under all foreseeable conditions, avoiding faulty operation and potential air disasters.

DO-254 compliance is becoming increasingly common on commercial and military aviation projects. Companies often struggle with the requirements and costs of DO-254 compliance. Engineers can use Model-Based Design for requirements analysis, design, automatic HDL code generation, and verification to produce airborne electronic hardware that adheres to DO-254. Model-Based Design for DO-254 combines automation tools from MathWorks and Mentor Graphics for design and verification to support a development process that goes from concept through implementation. This approach streamlines the development process and reduces costs.

Simulink® from MathWorks is the starting point for Model-Based Design, in which models of the complete system, including the algorithm and environment are created in the conceptual design phase. These models can be simulated and analyzed throughout the design process to ensure that algorithms meet specifications. Such an approach brings two benefits:

- Finding and fixing errors earlier in the design phase is dramatically cheaper than when they are found during implementation and testing.
- Designs, tests, and analysis can be reused throughout the development process.

Mentor Graphics offers industry-leading tools that span the design workflow. The tools described in this paper focus on chip-level solutions for hardware description language (HDL) design and verification. They also include capabilities for consistently managing and tracking requirements from design concept through implementation.

Model-Based Design promotes a requirements-oriented project view and greater integration and reuse between conceptual and detailed modeling and design work. This paper discusses the workflow for Model-Based Design at a high level, explains the types of activities performed in each environment, and highlights how the tools can be combined to maximize reuse and efficiency.

DO-254 Overview

The FAA began enforcing DO-254 in 2005, through Advisory Circular AC 20-152. DO-254 defines a set of objectives that airborne applicants and integrators must meet for their hardware to be certified for use in airborne systems. DO-254 is modeled after the equivalent standard for certifying software, DO-178B, which was originally published as DO-178 more than 25 years ago. Although DO-254 originated as a civil aviation standard, it is increasingly being considered in military airborne applications and other high-integrity applications in industries such as medical, nuclear, and transportation.

DO-254 Compliance and the Life Cycle

DO-254 defines both a design process life cycle and supporting processes that must be followed throughout the design development process. As shown in Figure 1, these two aspects of a DO-254 project are fed by an extensive planning process that identifies and details the methodology of the project.

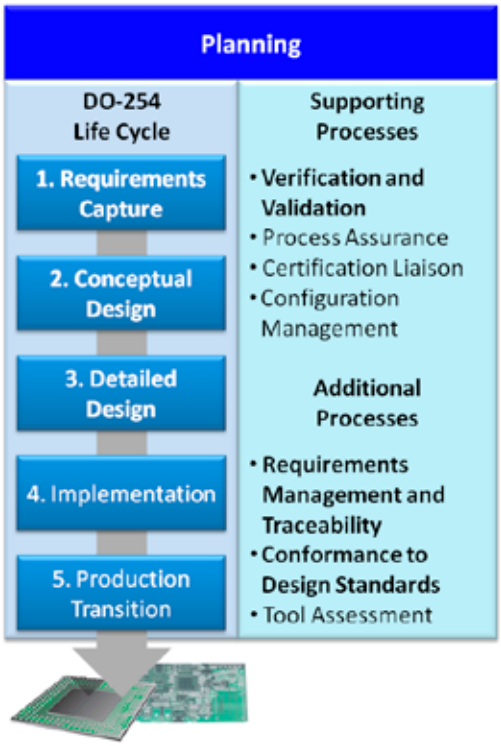

Figure 1. DO-254-compliance life cycle and associated processes. DO-254 life-cycle phases and associated processes that appear in bold are discussed in this paper.

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 in the life cycle. The following processes, which appear in bold type in Figure 1, are discussed in this paper:

- Requirements Management and Tracing – DO-254 demands that design elements and verification artifacts link back to the requirements that they support. Traceability provides proof that a design has implemented the intended function and that it has been thoroughly verified to ensure that it performs this function under all foreseeable conditions.
- Conformance to Design Standards – In a compliant-development process, pertinent standards must be developed for each phase. As a design moves from phase to phase in the life cycle, it is necessary to show that these standards are being met.
- Verification and Validation– At each phase in the design, the designer must ensure that the current version of the design (conceptual model, HDL, netlist, hardware) achieves requirements and matches the previous version. Many different techniques and tools that range from simulation to advance analysis can be used to perform verification of the design at different phases. This paper discusses different verification methods at a high level with attention to how design activities and artifacts can be reused throughout the process.

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

Figure 2 shows a high-level DO-254 workflow using Model-Based Design. The five centered boxes represent design phases. The vertical arrows connecting the design phases represent a transformation of the design, such as the transition from spoken language requirements to a conceptual design using Simulink.

The curved arrows connecting design phases represent supporting processes, including traceability, conformance, and verification. Relevant products and tool capabilities that enable transition between design phases and supporting processes appear as bulleted items.

In this workflow, engineers collect and manage requirements with Mentor Graphics® ReqTracer™. From the requirements, an executable Simulink model is created to explore a conceptual design that links directly to requirements in ReqTracer.

Using verification and validation tools from MathWorks, engineers can then perform functional testing and formal analysis at the conceptual model level and add specific design details and implementation attributes to the model, such as fixed-point effects. These elaborated models enable engineers to verify that the design is fully tested and meets all necessary requirements. From this fully tested model, a detailed design in HDL can be automatically generated with Simulink HDL Coder™.

From this stage forward Mentor Graphics HDL Designer provides the primary environment for additional HDL development, code checking, code visualization, and review. Further verification of the detailed HDL design can be performed in the Mentor

Graphics verification environment by reusing the test vectors created at the conceptual model level with ModelSim® and Questa®. Formal analysis is supported by 0-In® Formal Verification for model checking, and FormalPro™ for logical equivalency checking. Clock domain crossing analysis is performed by 0-In® CDC. FPGA synthesis and integration with FPGA vendor place and route tools is accomplished by Precision RTL®.

Joint MathWorks and Mentor Graphics Workflow

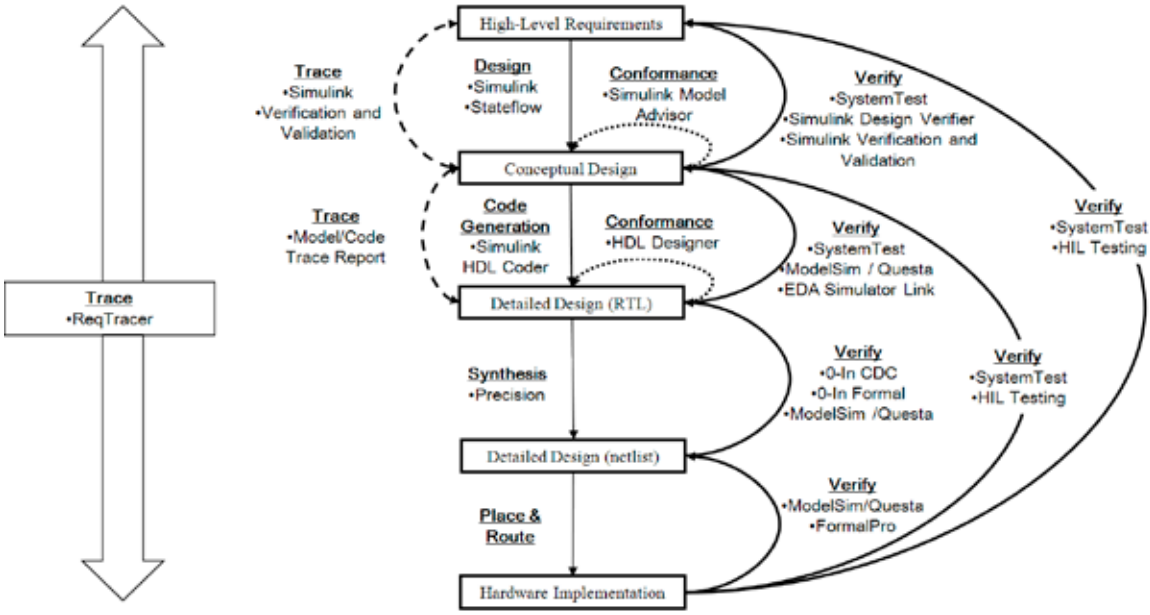

Figure 2. DO-254 workflow with Model-Based Design.

1. Requirements Capture

DO-254 projects are requirements-driven projects. Requirements define the intended function of a device, and a DO-254 compliant process ensures that a device performs its intended function. System requirements allocated to a hardware item must be reviewed, captured, managed, and traced to the pertinent design activities. Likewise, derived requirements, those derived from design decisions throughout the process, must go through these same processes.

Therefore, a DO-254 project must have mechanisms for:

- Capturing requirements, as per the first phase of the DO-254 life cycle
- Managing changes to requirements that occur throughout a program
- Tracing requirements to design and verification activities that occur throughout a program

Many companies that serve the aerospace market use enterprise-level requirements-management systems, such as the DOORS® database product from IBM. DOORS provides a database mechanism to store and manage requirements and is capable of supporting large and complex systems. Other companies, such as subcontractors developing only a component in a larger system, may use office productivity tools such as Microsoft® Word or Excel® to capture component-level requirements. In any case, it is essential that the design and verification work link back to these requirements, regardless of their source environment. In DO-254, this linking is called requirements tracing.

Capturing a static set of requirements can be achieved relatively simply. However, establishing a requirements-driven design flow and managing requirements as they evolve throughout a project is a much more daunting challenge. A requirements-driven design flow requires entering requirements, tracking changes to requirements, and linking to design and verification artifacts.

Mentor Graphics has applied its expertise in design automation tools to automate requirements management and tracing. ReqTracer pulls requirements from their source (e.g., DOORs or Word) and links them to design elements and verification artifacts. It can also help validate requirements by facilitating requirements reviews, guiding verification activities based on requirements status, and providing certification artifacts.

ReqTracer integrates with Mentor Graphics native environments of HDL development, verification, and synthesis, and is flexible enough to adapt to nearly any other tool that would be used in a DO-254 development process. To support workflows that use Model-Based Design, it has specific capabilities that integrate with Simulink. Designers are able to add requirements information to specific blocks, subsystems, and models as block properties. This requirements information is then automatically passed through to HDL code generated by Simulink HDL Coder. This process is clarified in section 3 entitled “Detailed Design.”

In addition to traceability and validation support, ReqTracer assists in project management by creating a visual depiction of project status, which shows the requirements that have and have not been designed and verified. ReqTracer can also generate the traceability matrices required to meet DO-254 traceability objectives.

In essence, ReqTracer provides a requirements-oriented project management environment from concept through implementation and supports the traceability needs of DO-254 projects. Figure 3 depicts a requirements-driven flow established with ReqTracer.

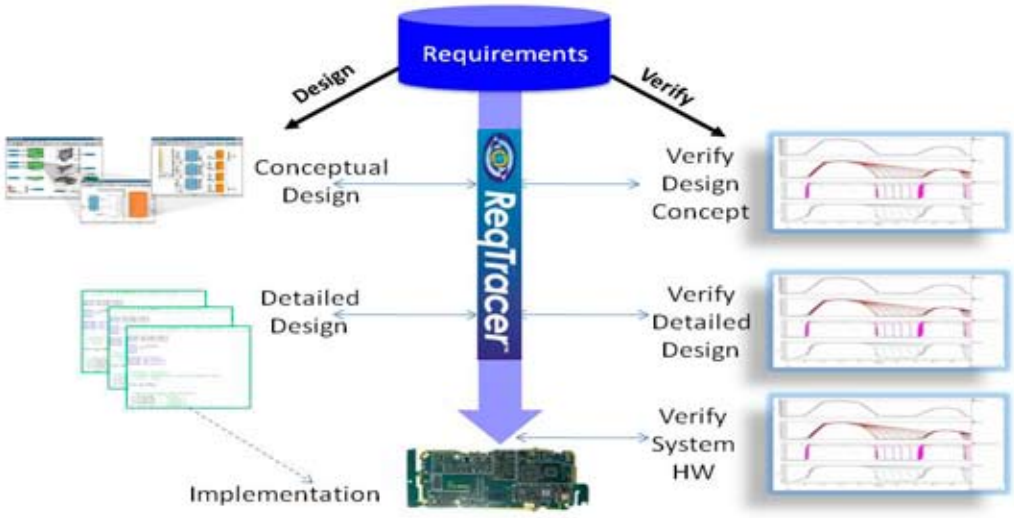

Figure 3. A requirements-driven workflow with ReqTracer.

2. Conceptual Design

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

Conceptual Model Design

Simulink is an industry-standard tool for designing, implementing, and testing aerospace designs. It serves as the platform for Model-Based Design. From the requirements captured previously, a design engineer constructs an executable version of the design. Simulink enables engineers to build up these algorithmic models in an intuitive graphical manner. Figure 4 shoes an example video compression algorithm that was developed in Simulink using Model-Based Design. Stateflow®, an add-on to Simulink, is used to develop finite-state machines and logic. Additional blocksets provide higher level functionality for application-specific tasks.

Simulating the larger system and environment in which the hardware will operate enables engineers to fully test a system before implementation. For example, consider the design of a takeoff-abort algorithm. This algorithm can be designed independently within Simulink or developed as part of a larger system-level aircraft model. The system-level aircraft model can include the logic algorithm, a six-degree-of-freedom airframe model with environmental effects, sensor models, and actuator models.

Having a system model enables engineers to test their designs earlier in the process and quickly evaluate what-if scenarios. In this example, the system-level model lets an engineer test the takeoff-abort algorithm under varying takeoff speeds, sensor failures, and pilot inputs.

As confidence increases in the design, models are elaborated to specify architecture and include implementation effects. Simulink Fixed Point™ provides capabilities to model and analyze a design to help engineers choose optimal fixed point word lengths. Simulink enables these effects to be simulated and compared back to a reference design to ensure that requirements are still being met.

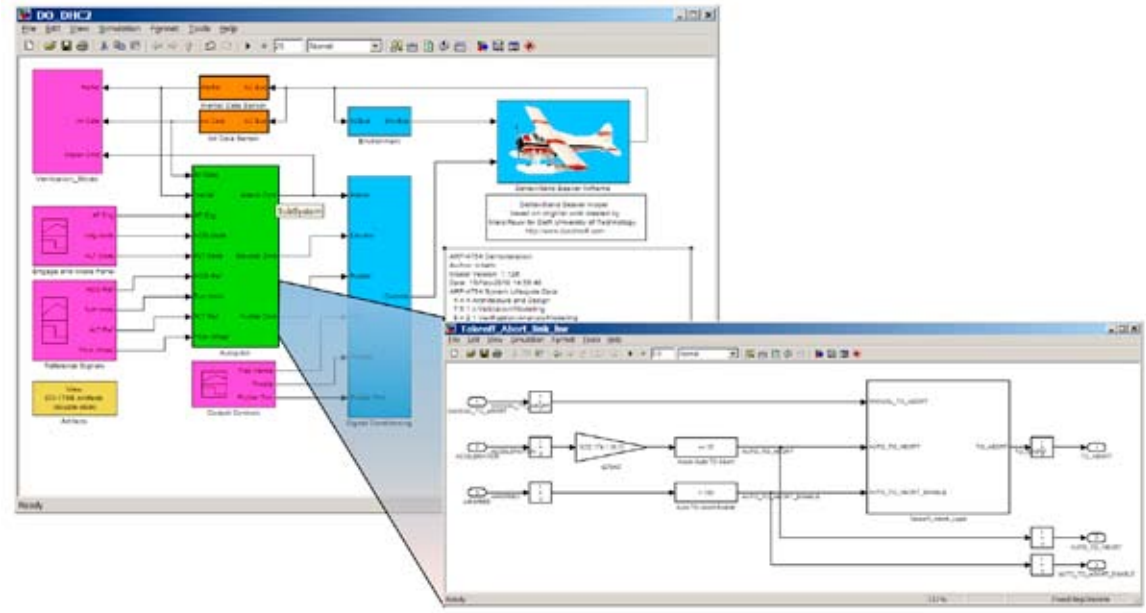

Figure 4. System-level aircraft model including takeoff-abort algorithm.

Traceability in the Conceptual Model

In a workflow with Model-Based Design, all elements of the conceptual design should be traceable to the requirements that they satisfy. MathWorks provides basic traceability support via Simulink Verification and Validation™.

ReqTracer also provides traceability support. In this workflow ReqTracer is used to establish traceability at the Simulink model level. This traceability is preserved in the generated HDL and extends throughout HDL analysis and testing.

Verifying the Conceptual Model

The conceptual design must be analyzed to verify whether it meets requirements. Several MathWorks products can assist in this task. For example, MATLAB® can be used to script execution, conduct parameter sweeps, and perform analysis on simulation

outputs. These tasks can be run in parallel on multi-core machines or clusters using parallel and distributed computing capabilities.

MathWorks has also developed tools specifically to aid in system verification. SystemTest™ is a testing platform that can be used to create and execute tests of the Simulink model. Tests can be authored to demonstrate that specific functional requirements are being satisfied. SystemTest automatically generates reports, which can be verification artifacts. As discussed below, these tests can be 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. While simulation is essential, it can be a challenge to ensure that a set of simulations fully exercises a design over all conditions. To ensure complete functional test coverage, formal analysis can be used in conjunction with simulation to generate test cases. These techniques use mathematically rigorous procedures to simplify and search through a 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 takeoff-abort algorithm discussed above. Typically, this type of logic in software or hardware involves a number of sensor inputs, such as airspeed, acceleration, and pilot input. With property proving technology, an engineer can use a commercial formal verification tool to verify a certain system behavior: "Prove to me that this logic will never engage if the airspeed and acceleration are within certain ranges." Simulink Design Verifier™ lets a developer define these mission-critical properties and prove that certain scenarios cannot happen under any conditions on the model level.

Throughout testing, model coverage can be a useful metric to assess how fully tests are exercising a model. Simulink Verification and Validation can track and report on model coverage. These coverage metrics should first be gathered using functional based tests executed against the model. Although functional tests are used to ensure that design requirements are met, they often do not exercise 100% of the design. Simulink Design Verifier leverages formal methods to automatically generate test cases to complement functional tests and ensure that 100% modified condition/decision coverage (MC/DC) coverage of the design at the model level.

Even if not a requirement for certification, this model-level coverage testing can be useful in validating and verifying a design. If test cases are not achieving 100% coverage, it may be an indication that additional requirements are needed, design elements are unnecessary, or that a design is inherently difficult to test. These insights are valuable in refining requirements, developing a conceptual design, and creating tests. Significant savings are realized by fixing these errors early in the conceptual design phase.

Note that testing must also be exercised on HDL and later stages of design. However, as discussed below, the test cases generated on the conceptual model can be reused in HDL-level testing.

Conforming to Conceptual Model Design Standards

As discussed previously, the development of and adherence to design and coding styles 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 aesthetic and functional aspects of the model. Model Advisor is a standard feature of Simulink that can execute prepackaged sets of model checks. Simulink Verification and Validation enables the customization and deployment of these checks within an organization.

These checks are static, which means that design engineers are not executing the model, but rather looking at it statically and analyzing its characteristics. Typical characteristics include settings, data types, code generator settings, and HDL settings. This static process can detect simple mistakes, such as a missing connection for a block input or output. It can also detect more complex and serious issues, such as block settings that may result in an overflow in a fixed-point operation. The HDL detailed design must also conform to standards. Checking that HDL code conforms to acceptable standards is discussed in the next section.

3. Detailed Design

The detailed design process is generally agreed to begin at the HDL stage of development. This development can be handwritten or automatically generated with Simulink HDL Coder. Automatically generating HDL can increase efficiency by reducing the amount of hand coding required and enabling faster design iterations. The workflow discussed below includes automatic HDL generation.

It is worth noting that the verification activities discussed throughout the DO-254 workflow can be used whether hand coding or automatically generating HDL.

Generating the HDL from the Conceptual Model

In a workflow using Model-Based Design, the generated HDL model can be read into HDL Designer from Mentor Graphics for independent assessment and integration with either existing HDL or portions of the design less suited for conceptual modeling in the MathWorks environment.

Within HDL Designer, the HDL code is examined via code reviews, automatically checked against HDL coding standards, and visualized for ease of understanding.

Traceability in Detailed Design

When the HDL detailed design is generated from the conceptual Simulink models, all of the information contained in the conceptual design is preserved in the detailed design. For example, traceability information is captured within the Simulink model.

Simulink HDL Coder inserts all requirements information into the generated HDL. This traceability can then be viewed and managed throughout the rest of the process

with ReqTracer, which establishes traceability from detailed design back to conceptual design and requirements.

Checking HDL Coding Standards

Just as conformance of the conceptual design to design standards was demonstrated at the Simulink model level, conformance of the detailed design to HDL coding standards must also be demonstrated. HDL Designer has an integrated design rules checking engine that is sometimes referred to as a linting tool. This feature provides a number of different rule sets that can be used as is or customized. One of these is called the "DO-254 rule set." This rule set contains a number of coding rules that can be used to meet the DO-254 objective of defining a set of HDL coding standards (as specified in Order 8110-105). HDL Designer can automatically check HDL code to ensure that it conforms to this rule set.

Reviewing the Code

HDL code must be examined by independent review to ensure that it conforms to HDL coding standards and correctly implements the required functionality. Using HDL Designer rules checking engine to ensure that conformance to HDL standards was discussed previously.

Both Simulink HDL Coder and HDL Designer can assist with reviewing HDL code to ensure that it implements the required functionality. The HTML Code Generation report generated by Simulink HDL facilitates navigating from HDL code back to the blocks in the Simulink conceptual model and requirement. This navigation is bidirectional. Using the graphical conceptual model in conjunction with the generated HDL code can help reviewers more quickly understand and analyze a design. A traceability report is also generated to aid in this review process.

HDL Designer can also help 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 (described in next section) should serve provide independent output assessment of any generated code.

Verifying the HDL Model

Verification must also be performed at the detailed design level. This task is required for both handwritten code and automatically generated code. As with the conceptual model, various techniques can be employed for performing verification. These techniques range from basic simulation to advanced formal methods.

There is a high degree of flexibility in how verification activities are performed. In some cases, MathWorks provides access to Mentor Graphics simulation capabilities from within the MathWorks tool chain. Other activities will be conducted solely within the Mentor Graphics tool chain. An organization should consider which tools and techniques will be used and discuss them with their certification liaison. The sections below

focus on how to reuse testing data and artifacts developed earlier in the conceptual design process.

Simulating the HDL Model

The section entitled "Conceptual Design" discussed how simulation was an important element of verifying the conceptual design. Simulation is also an essential tool for verifying the detailed HDL design. ModelSim is an industry-leading simulator in the military and aerospace industry. ModelSim 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. ModelSim supports designs of all complexities, but it is most frequently used on small to mid-sized FPGAs.

Questa combines the ModelSim simulation engine with advanced verification capabilities from languages such as SystemVerilog, PSL, and SystemC. These capabilities include:

- 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), which has been donated and accepted as an Accellera standard
- A verification management environment for ease of managing and reporting on project verification activities

It also has a deep integration with ReqTracer to provide an added level of automation from the UCDB to the source of requirements (e.g., DOORS). These advanced verification methods aid in verifying devices of substantial complexity that contain concurrent behaviors. Thus, Questa is typically used on complex devices, including ASICs and large FPGAs.

Verifying the HDL Model – Conceptual Design Test Case Reuse

During the conceptual design phase, the simulation engine is Simulink. During the detailed design phase, the simulation engine is ModelSim or Questa. While the type of simulation being employed has changed, there are several ways to leverage the verification activities performed in the conceptual design phase in the detailed design phase. The two primary tools are through cosimulation and test-bench generation.

EDA Simulator Link ™ from MathWorks enable tests authored in the MATLAB, Simulink, and SystemTest environments to be executed against HDL code simulated in ModelSim and Questa. This cosimulation lets engineers easily reuse the test cases and

analysis routines developed during conceptual design to ensure that they are functionally equivalent.

The takeoff-abort algorithm discussed earlier was designed and simulated as part of a larger system-level aircraft model. From this Simulink conceptual model, an HDL detailed design of the algorithm was automatically generated using Simulink HDL Coder. EDA Simulator Link enables the designer to run the system-level model and tests from the conceptual design against the generated HDL running in ModelSim or Questa.

Simulink HDL Coder also includes the ability to generate HDL test-bench files based on tests performed on the conceptual design. In the HDL generation options, the designer can specify that test-bench files be generated along with the algorithmic HDL. In this manner tests performed on the Simulink conceptual design can be reused when hardware engineers do not have access to Simulink.

Both cosimulation and test-bench generation promote test case reuse and enable engineers to quickly test the detailed design (HDL). This ability can dramatically increase iteration times and reduce associated costs. It also allows the engineers to leverage the analysis capabilities of both environments. High-level functional testing can be quickly performed and analyzed from the Simulink design environment. Detailed analysis can be performed in ModelSim and Questa.

Verifying the HDL Model – Advanced Analysis

A DO-254 workflow that uses Model-Based Design promotes the concepts of reuse in design and verification. Reuse achieved through cosimulation and test-bench generation is well suited for functional testing. However there are additional analyses available to the designer in Mentor Graphics design environments. These advanced analysis techniques are discussed below.

Clock-Domain Crossing Analysis

Integrating multiple functions into a chip is commonplace today. Integration usually involves a single device with multiple, asynchronous clocks. Clock signals that cross domains can lead to a condition called metastability, which is a leading cause of device failure. The problems associated with signals that cross clock domains are extremely difficult and expensive to debug and fix because they typically are not detected until a failure occurs in the lab or field. 0-In CDC is a clock-domain crossing analysis tool based on formal methods. A design with two or more asynchronous clock domains should use 0-In CDC during the design process to help reduce the likelihood of metastability.

Formal Verification (HDL Model Checking)

Model checking is a formal methods technique that analyzes a design against its requirements, which are written as assertions. Model checking was discussed earlier in the section entitled “Verifying the Conceptual Model.” The same concept of exhaustively proving safety-critical properties is true at this level of design as well; in this case, the model is an HDL version of the detailed design. Model checking can exhaustively prove

that a design performs its intended function, and it is thus mentioned in DO-254 Appendix B as an acceptable method of advanced verification for level A/B devices. 0-In Formal Verification is the Mentor Graphics model checking tool

Synthesizing the HDL Code

Synthesis, which is still considered part of the detailed design process, is a transformation of HDL code into a technology-based netlist. Design synthesis is at the heart of all modern PLD, FPGA, and ASIC design flows. Designers, and in turn their synthesis tools, have historically tended to focus on achieving three main design goals: timing performance, design area, and tool run time. However, in military and aerospace applications where design assurance is critical, a synthesis tool must take into account additional considerations.

Precision Synthesis, an FPGA-vendor independent synthesis solution from Mentor Graphics, balances aspects of safe synthesis with performance, optimization, and timing goals. It ensures 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. In addition, it provides integration with the Mentor Graphics FormalPro logical equivalency checking tool to provide an added measure of assurance for the generated netlist. More information on FormalPro appears in the next section).

Placement and routing of the netlist into a physical device depends on specific knowledge about the target FPGA device. This process requires tools provided by the FPGA vendor. Precision Synthesis has integration with the FPGA vendor software and can directly launch these tools from the Precision environment.

Verifying the Netlist

As described in the introduction, 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 and ensure that it is equivalent to the HDL detailed design.

Static Timing Analysis

Precision Synthesis has an internal static timing analysis that runs as part of the synthesis process. At this point, the analysis is done with estimations only since the actual physical location of the

circuitry is not yet known. During the place and route process, the FPGA vendor tool will run the final timing analysis when physical placement in the target device is known.

Gate-Level Simulation with Timing

ModelSim and Questa support 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 same test bench from HDL verification should be used. EDA Simulator Link also supports cosimulation at this level of design.

Logical Equivalency Checking

In a DO-254 compliant workflow, 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). However, for large and complex designs, this repetition can be incredibly time-consuming. A faster approach for verifying synthesis results is a type of formal verification known as logical equivalency checking, or LEC. The Mentor Graphics LEC tool is FormalPro.

FormalPro compares one model to another to determine whether they are functionally equivalent. This comparison is typically done on the input and output of a process. For example, FormalPro can compare the HDL fed into synthesis with the netlist generated to determine if they are functionally equivalent. This same process can be done to compare the input to place and route (i.e., the synthesized netlist) with the output of place and route. This formal methods approach enables faster verification than gate-level simulation.

4/5 - 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 can be reused in these phases. A detailed discussion of these phases is beyond the scope of this paper.

Summary and Conclusion

The increasing prevalence and cost of projects needing to comply with the DO-254 standard is forcing companies to evaluate how they can be more efficient in their development processes and support DO-254 compliance throughout their processes.

Customers developing complex airborne electronics currently utilize a range of tools for design, test, and implementation. MathWorks tools are well established in algorithm design, simulation, implementation, and analysis. Mentor Graphics offers industry-standard capabilities for hardware design, simulation, analysis, and implementation.

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. This paper described the ways in which MathWorks and Mentor Graphics tools can be combined in such a workflow.

Resources

VISIT
www.mathworks.com

TECHNICAL SUPPORT
www.mathworks.com/support

ONLINE USER COMMUNITY
www.mathworks.com/matlabcentral

DEMOS
www.mathworks.com/demos

TRAINING SERVICES
www.mathworks.com/training

THIRD-PARTY PRODUCTS AND SERVICES
www.mathworks.com/connections

WORLDWIDE CONTACTS
www.mathworks.com/contact

E-MAIL
info@mathworks.com

MathWorks®
Accelerating the pace of engineering and science