2010

# Hardware Certification for Real-time Safety-critical Systems: State of the Art

Andrew J. Kornecki
*Embry-Riddle Aeronautical University*, kornecka@erau.edu

Janusz Zalewski
*Florida Gulf Coast University*



## Scholarly Commons Citation

Kornecki, A. J., & Zalewski, J. (2010). Hardware Certification for Real-time Safety-critical Systems: State of the Art. *Annual Reviews in Control, 34*(1). https://doi.org/10.1016/j.arcontrol.2009.12.003

Annual Reviews in Control 34 (2010) 163–174

Contents lists available at ScienceDirect

Annual Reviews in Control

journal homepage: www.elsevier.com/locate/arcontrol

# Hardware certification for real-time safety-critical systems: State of the art

Andrew J. Kornecki [a], Janusz Zalewski [b,1,*]

[a] Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA
[b] Florida Gulf Coast University, Fort Myers, FL 33965, USA

ARTICLE INFO

*Article history:*
Received 1 April 2009
Accepted 10 December 2009
Available online 14 April 2010

*Keywords:*
Safety-critical systems
Real-time systems
Tool qualification
Hardware certification
FPGA

ABSTRACT

This paper discusses issues related to the RTCA document DO-254 *Design Assurance Guidance for Airborne Electronic Hardware* and its consequences for hardware certification. In particular, problems related to circuits' compliance with DO-254 in avionics and other industries are considered. Extensive literature review of the subject is given, including current views on and experiences of chip manufacturers and EDA industry with qualification of hardware design tools, including formal approaches to hardware verification. Some results of the authors' own study on tool qualification are presented.



## 1. Introduction

In modern societies, where computing technologies are applied in nearly every aspect of everyday life, from door locks and watches, to security systems and traffic lights, to cars, trains, airplanes and space vehicles, there is a need to protect ourselves against unexpected and undesirable behaviors of computer based systems. Such need led to the introduction of computer safety standards developed by professional organizations and enforced by government regulations. These standards can be roughly divided into two categories: those relevant to hardware and those relevant to software.

The introduction of such standards started relatively early in the domain of aviation, because of the unusual vulnerability of the society facing aircraft related accidents potentially caused by computer failures, and the corresponding urgent need of developing protective measures. In this view, the U.S. government and international agencies that regulate respective industries have issued and promoted a number of standards and guidelines, related to certification and/or other aspects of software assurance, such as qualification, licensing or validation, in their specific areas of interest. Especially important is guidance related to civil aviation and airborne systems: DO-178B in the U.S. and its European counterpart EUROCAE ED-12B (RTCA, 1992), and DO-254 in the U.S. and its equivalent EUROCAE ED-80 (RTCA, 2000).

In general, an exhaustive discussion of safety issues related to certification of digital hardware should include three types of technologies used in contemporary designs: microprocessors, programmable logic devices (PLDs), and hardware design languages (HDLs). A thorough discussion of safety aspects of using microprocessors in such systems has been published recently by Mahapatra et al. (2006–2009). Thus in this paper, we only deal with selected aspects of certification of microprocessor-based systems.

Furthermore, although HDLs are extensively used in electronic design automation, both for microprocessors and PLDs, an extensive study on safety aspects of using HDLs and their certification is still to be written. General software aspects of certification have been covered in separate papers (Kornecki & Zalewski, 2008; Kornecki & Zalewski, 2009).

Consequently, the focus of the current paper is on certification of PLDs. We discuss issues related to hardware related standard, DO-254, and respective hardware aspects of airborne systems' certification. The paper is structured as follows. The next two sections discuss issues related to circuits' compliance with DO-254 in avionics and other industries, respectively. Section 4 outlines a more formal approach to certification and Section 5 presents some views on and experiences with software tools qualification against DO-254. Some results of a study on tool qualification are presented in Section 6, and the summary in Section 7.

## 2. Circuits' compliance with DO-254

With the progress of micro-electronic technologies, the avionics hardware is typically custom generated using, as components, programmable logic devices. Field-Programmable Logic Arrays

* Corresponding author. Tel.: +1 386 226 6888.
*E-mail addresses:* kornecka@erau.edu (A.J. Kornecki), zalewski@fgcu.edu (J. Zalewski).
[1] Tel.: +1 239 590 7317.

1367-5788/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.arcontrol.2009.12.003

(FPGA) and Application Specific Integrated Circuits (ASIC) are two leading implementation technologies. More often the devices include also components containing Intellectual Property (IP) chips with dedicated algorithms or custom made solutions resembling general purpose embedded microprocessor's functionality. All this caused an emergence of RTCA document DO-254 (RTCA, 2000), which deals with safety assurance for hardware used in avionics and can be applied to other safety-critical systems.

What also contributed to the origins of DO-254 is the fact that avionics companies and designers, facing the rigors of DO-178B guidance, began moving system functionality from software to hardware (Hilderman & Baghai, 2003). As reported by Cole & Beeby (2004), "There are several schemes that have been used by some to take advantage of a current loophole that allows airborne software functionality to be embedded in firmware or programmable devices. This loophole affectively sidesteps the need to adhere to DO-178B as a software standard." Thus, a new document was introduced that forms the basis for certification of complex electronic hardware, by identifying design lifecycle process, characterizing the objectives, and offering means of complying with certification requirements. The Advisory Circular published subsequently by the FAA (2005) clarifies the applicability of DO-254 to custom micro-coded components, such as ASIC, PLD, FPGA, and similar.

Our previous papers (Kornecki and Zalewski, 2008, 2009) reported on some of these issues. In the section below, we are discussing some additional papers in more detail, review a number of most recent approaches to hardware certification according to DO-254, and cover other important points from the literature.

### 2.1. General issues of DO-254 certification

Miner et al. (2000) were probably the first to consider compliance with DO-254, before even the standard was officially released. In a joint project with the FAA, NASA Langley was developing a hardware design to gain understanding of the guidance document and generate an example suitable for training. A core subsystem of the Scalable Processor-Independent Design for Electromagnetic Resilience (SPIDER) was selected for this case study.

Hilderman and Baghai (2003) offered an advice to manufacturers to map their existing development processes to those of DO-254. At the strategic level, the recommend approach is "to focus on ensuring correctness at the conceptual design stage and then preserve the design integrity" as one proceeds through the detailed design and implementation. However, each individual vendor or designer faces multiple specific design problems that must be addressed to meet the DO-254 objectives. How they proceed depends on an individual vendor and the type of problem.

Young (2004) reported on the role commercial off-the-shelf (COTS) products could have in safety-critical avionics systems creation under DO-254 guidance. The APMC (Avionics Process Management Committee) has produced EIA-933 Standard for Preparing a COTS Assembly Management Plan. This document recommends how to select and manage suppliers of avionics COTS products.

In the white paper of the DO-254 Users Group, Baghai & Burgaud (2004) offered a package including the following five items designed to assist in the qualification process:

- The process documents, that help define, benchmark and improve the industrial design, verification, validation, and quality assurance processes.
- The quality assurance checklists, for reviews and audits, ensuring that each project is compliant with the defined industrial process.
- The tools for requirements' management and traceability, checking compliance of HDL code with coding standards, HDL code verification, and test suite optimization.
- The tools integration into the industrial process, supporting their qualification (interfaces, report generation for a certification audit, trainings, tools assessment, etc.).
- The DO-254 training by consulting partners.

Cole and Beeby (2004) studied DO-254 compliance for graphic processors, considered as COTS components, and proposed a four-phase approach to meet DO-254 objectives:

(1) Provision of a DO-254 COTS data pack to support the use of a given electronic part.
(2) Provision of a DO-254 compliance statement.
(3) Process improvement and further analysis.
(4) Ongoing support for new parts and processes.

For a part that was already fully designed and in production (as Phase 1 above), only the following could be done for compliance:

- Review of the available component management data from chip designer and chip manufacturer.
- Analysis of the available data for completeness.
- Augmentation of the available data through further analysis and testing, in case of any existing deficiencies.
- Developing recommendations to improve the process for future parts.
- Development of the COTS data pack in a format compatible with DO-254.
- Revising the data pack with DER to ensure completeness and suitability for DO-254 submission by end customers.

Phase 2 is meant to take a closer look at the design and development processes, without changing them. Rather, it provides the foundation for improving these processes and obtaining better compliance with DO-254. Phase 3 is a step to put the recommendations of Phase 2 into practice. Finally, the role of Phase 4 is to provide the continuous process of DO-254 compliance for making any new parts that might need such compliance.

Glazebrook (2007) discussed certification according to DO-254 in the British aviation industry, focusing on the 26 data items listed in the standard as the compliance suite, of which four are required for submission: (a) plan for hardware aspects of certification; (b) hardware verification plan; (c) top level drawings; and (d) hardware accomplishment summary. He made several recommendations summarized below:

- Developing robust and accurate plan early in the program before transition to the development stages of the lifecycle is essential.
- Using proofing and obsolescence robustness assessments as part of the component selection process.
- Focusing on proven techniques and approaches.
- Ensuring that robust and controlled transition criteria are implemented prior to any development.
- Ensuring that the requirements are controlled and sufficiently abstracted in the inception phase.
- Using traceability metrics from the project inception.
- Considering verification at the start of the program.
- Detecting and eliminating coding errors should be seen as part of the development activity.

Lee (2007) analyzes the British aviation authority views on procurement and acceptance of military avionic systems based on

the continuing technical advances in electronic system design, and in particular the capabilities of Programmable Logic Devices (PLDs). An interpretation of the DO-254 from the perspective of military systems identifies several issues in DO-254 compliant development and certification, such as:

- Inadequate level of detail in requirements.
- Inadequate formal planning and following of plans.
- Lack of independence in quality assurance and verification.
- Inadequate and non-automated traceability.
- Lack of automatic testing.

Two papers from Barco-Siles S.A. (Pampagnin & Menis, 2007 and Leroy & Bezamat, 2007) report on the way the company deals with increasing demands on the hardware development processes resulting from DO-254 conformance. In the former paper, the authors state that even though implementing DO-254 has necessarily a non-negligible cost, this can be considered as an investment. It obliges the supplier to analyze in detail their processes, methodologies and tools and to apply a structured development processes, with a rigorous quality assurance. It also allows the supplier to adapt defined set of internal processes to the design assurance level targeted to optimize efforts. The resulted products have a better quality and the development cycles are optimized. Verification is focused on design errors, and effort and resources are better distributed. It obliges the subcontractor to respect a structured development processes. The initial cost has to be compared with the level of quality for the subcontractor. Applying the DO-254 gives the assurance that the applicant can obtain from its subcontractor a good level of quality, good documentation, and the ability to reuse the design, if necessary.

The latter paper deals specifically with the way how Barco-Silex has applied the DO-254 guidance for designing FPGA circuits. It addresses the development cost impact on FPGA design throughout the verification level and the amount of data delivered in this process. The golden rule to provide hardware design assurance for any design entity is that the three fundamentals design phases, Specification, conception and validation, must be performed by different people in order to avoid error propagation over the lifecycle. In many cases, validation results may need to be reviewed independently to confirm proper procedures were followed and that the results confirm that the requirements have been met.

Plastow (2007) considers DO-254 compliance from the perspective of space applications, and argues that: "Hardware development follows a process that is very similar to software development. Many of the proven software techniques used in software development/verification can be used with little or no modification on hardware." Thus, he recommends using software techniques, such as change impact analysis and fault tree analysis to provide a way to better understand and verify a complex electronics design. As he states, such interdisciplinary approach would insure better design quality.

More recently, the British Avionics Systems Standardisation Committee issued an official guidance (as an extension of (Lee, 2007)) based on DO-254, aimed at the assurance of PLDs implemented on a single integrated circuit (ASSC, 2009). It is interesting to note some of the assumptions that were mentioned as a rationale for issuing this document:

- There continues to be an exponential growth in the use of PLDs in all industrial sectors including military aerospace.
- PLDs have particular advantages over microprocessor-based systems in terms of processing power, I/O capacity, reduced footprint and a reduction in number of ICs on the circuit board.
- The DO-254 guidance has been given increased prominence in the civil aerospace sector by the FAA Advisory Circular (FAA, 2005) and is becoming a "de facto" standard.

### 2.2. Response from chip manufacturers and EDA industry

Chip and board manufacturers are particularly eager to comply with DO-254, because of their concerns about the market share. So is the entire Electronic Design Automation (EDA) industry, whose design tools and processes undergo an additional scrutiny. Since compliance with DO-254 guidance is considered a technological advantage, most of the vendors began changing their development processes towards meeting the guidance objectives. Several companies announced their readiness to comply with certification requirements.

Mentor Graphics and Aldec/Actel seem to take the lead in providing compliance of their products with DO-254. While some of respective papers have been reviewed in Kornecki and Zalewski (2009), here we provide some additional insight.

Dewey (2008) outlines the contents of the standard and comments on the meaning of its requirements to the vendors and suppliers, especially on added costs. He focuses on the properties of data (artifacts) generated throughout the design process and lists the six characteristics, as per DO-254:

- Non-ambiguity, i.e., having a single interpretation.
- Completeness, i.e., inclusion of all requirements along with the associated data.
- Verifiability, i.e., existence of means to determine that data are correct.
- Consistency, i.e., avoidance of conflicts among data.
- Modifiability, i.e., no need to change the structure of data.
- Traceability, i.e., ability to determine the origin of data.

Regarding the tool qualification, he states what the standard does, that:

- Tools generating code have more rigorous qualification process than tools that verify results.
- Each version of a tool has to be qualified in the context of particular projects.

Lange & Boer (2007) give an overview of functional hardware verification methodologies, as a part of the design process, but with very little reference to DO-254, contrary to what the paper title claims. Their important consideration, however, is that the verification techniques that served well the designs 10–15 years ago are no longer adequate due to a tremendous increase in design complexity and integration. As a consequence, design verification has become a limiting factor in safety-critical systems, especially with respect to such factors as: complexity, concurrency and metastability. Latest verification techniques are then described that take care of such issues as state explosion, design traceability and effectiveness of coverage.

One of Mentor Graphics' techniques, called Advanced Verification Methodology (AVM), consisting of constraint random test generation, a total coverage model, design intent specification, and formal model checking, is described in (Keithan et al., 2008). It has been used on a practical design of an FPGA based DMA engine at Rockwell-Collins, with the DO-254 compliant project.

The use of AVM is an important consideration for a safety-critical project, since it is based on an open source Transaction Level Modeling (TLM) class library and supports standard languages SystemVerilog and SystemC. As such, even though originated at Mentor Graphics, it is vendor neutral. Due to its open

source nature it allows code inspections that may be required for certification. Although the project has not been fully completed at the time of this writing, it was believed that AVM helps not only demonstrate that the requirements have been satisfied to the highest possible level (the ultimate verification goal off DO-254), but also assists in shortening design cycles. Interestingly, the requirements phase used a requirement capture tool DOORS, which has been typically applied in software development projects.

Lee & Dewey (2007) shed more light on meeting DO-254 objectives in a form acceptable to the auditor known as Designated Engineering Representative (DER), by explicitly addressing its subset:

- Requirements management and tracking, with the use of such tools as Reqtify or DOORS.
- Register Transfer Level (RTL) code validation, with an automated method to measure RTL to a company standard.
- Verification process assurance, with the use of AVM.
- Producing design documentation, from requirements, to the RTL code, to the bit streams or Graphic Data System (GDS) II file format.

All this is written and outlined keeping in mind that "DO-254 is not a burden but a set of guides that helps standardize hardware systems assurance, making flight systems safe."

Aldec and Actel, working in alliance, published information on their efforts towards making their products DO-254 certifiable. Sysenko and Pragasam (2007) outlined their process for airborne systems design assurance which relies on the verification methodology called Hardware Embedded Simulation (HES), and follows two traditional steps: RTL simulation and gate-level simulation. It is a hardware–software simulation platform driven by software that facilitates the implementation of the design in a reconfigurable hardware, such as an FPGA, and then verification of the design functions. Zalewski (2007) described in more technical detail (although without any references to DO-254) what seems to be the core of such approach, with using a hardware accelerator, which is essentially a hardware board and a simulator connected via a high-speed interface with intelligent clock.

Further, Zalewski (2008) elaborates on Aldec's DO-254 Compliance Tool Set (CTS), developed to address several challenges stemming from DO-254, including the following:

- Introducing inadvertent errors in functionality or timing during the FPGA design process.
- Running the design in the target FPGA device at required operational speed, and driving it with the same test cases an in the traditional HDL simulation.
- Requirements traceability ensuring complete coverage with the same set of test cases as in HDL simulation.

More information on the Aldec verification methodology is given in a White Paper (Aldec 2009). Its essential component is in-hardware testing, which – if applied properly – provides the following benefits:

- Same number of tests can be run in the target device as I the HDL simulator.
- Full design verification in the target device can be made before the system test.
- The verification process runs at full speed driven by vectors captured during HDL simulation.

Automatic documentation is generated and traceability is assured back to the design requirements.

It is interesting to note that Aldec FPGA chips, as being used in safety-critical applications, have undergone some significant scrutiny. The Japan Aerospace Exploration Agency (JAXA) performed thorough evaluation tests on Actel A54SX-A/RTSX-SU FPGAs used on satellites (Sakaide et al., 2005). In particular, the operational life tests, the temperature cycling tests, and the radiation tests were performed to collect the reliability data and assess the risk of using these products on the flight units. However, the results were meant to be applicable only to this particular project and no reference to DO-254 was made.

Lundquist (2007) in his thesis looked in more details at the problems that arise when trying to certify system-on-chip solutions with DO-254 compliance. Used as an example of an embedded FPGA, the Actel Fusion FPGA chip with integrated analog and digital functionality is tested according to the verification guidance. The thesis shows that a certification procedure for a standard non-embedded FPGA based safety-critical system is possible. As the author states, if similar solutions could be used in the aviation industry it would mean using fewer systems that could do more, thereby among other things reducing system complexity and development costs. However, the question of how these embedded chips could pass certification to be used in safety-critical systems remains unanswered in this thesis.

Vendors and manufacturers continue their efforts to, first, understand the need for DO-254 compliance, and then meet the standard's requirements. Recently, Reeve & Lange (2008) published a white paper discussing concerns of creating and executing the DO-254 compliant design project. Among the potential pitfalls that need to be addressed they list the following:

- Do not think that you can "get around" DO-254.
- Do not attempt to generate DO-254 documents after the fact.
- Engage in a DO-254 program with proper preparation.
- Be prepared that requirements traceability is a reactive process.
- Consider reusing current designs.
- Understand that tool qualification is difficult.

Altera in a White Paper on DO-254 support for FPGA design process (Altera Corporation, 2008) focuses on the cost risks and assuring proper partnerships to mitigate costs. Prospective partner roles in certification are listed as: providing verification tools, education and process support, involving independent IP suppliers, and obtaining certification services.

In another recent paper from Altera, Kenny (2008) states that "DO-254 compliance requires an ecosystem of partners that offers a range of DO-254 certifiable solutions." Further, the paper discusses the intricacies of DO-254 and emphasizes importance of dealing with several components of this ecosystem, such as:

- Planning and educational support.
- Verification methods and tools qualification.
- Certification services.
- Programmable logic and IP suppliers.

Xilinx in their White Paper (Le Mauff & Elliott, 2009) gives an interesting interpretation of DO-254, stating that "for FPGA design certification, the user has to demonstrate design assurance of both the design *and* the design process." Additionally, two specific aspects are mentioned to achieve design reliability: device-specific issues, which are the domain of the device vendor, and design specific issues, being controlled by the used. The latter can be addressed by the following fault mitigation schemes, depending on the Design Assurance Level (DAL):

- Triple-PFGA redundancy with external voting circuits.
- Dual-FGPA redundancy.

- Triple-modular redundancy (TMR) with voting circuits implemented in the FPGA.
- Circuit redundancy with arbitration inside a single FPGA.
- Bitstream scrubbing with error correction.
- Periodic FPGA reconfiguration.

## 3. Other types of circuits and industries

Complex electronic hardware, such as PLDs, FPGAs and ASICs, are the critical components to be assessed against DO-254, which has been recognized for safety-critical systems even before the emergence of this standard (Hilton & Hall, 2000; Civera et al., 2002). However, there are still other circuits used in safety-critical applications that may also require assessment and certification. They are discussed briefly in this section, along with the industries that require such assessments.

Although the FAA (2005) states in its advisory circular that there is no intention for commercial off-the-shelf processors to comply with DO-254, Section 11.2 of DO-254 addresses the use of such components in safety-critical avionics systems. In this view, Fulton (2006) discusses the use of COTS graphical processors in primary or secondary flight displays, which are safety-critical. The following requirements of DO-254 seem to be in place: manufacturer's track record, quality control procedures, service experience, and component's qualification with respect to reliability. With this in mind, a data package has been produced and the paper provides information how eleven specific criteria have been met for graphical processors.

Cole and Beeby (2004) present the entire DO-254 process for a graphical processor, while two other papers (Knaus, 2004; Quantum3D, 2007) discuss the role of certification in graphical systems using OpenGL standard. Very recently, Snyder (2008) outlines the entire idea of diverging from DO-254 by standardizing software-based graphical processor units (GPU). He states, referring to DO-254 scrutiny, that "This level of design assurance is never available for a commercial GPU chip comprising millions of gates." In contrast, software-based GPU can be developed according to a strict subset of a software standard, such as OpenGL, and "can undergo standard design assurance using the DO-178B software guidelines."

Several other electronic devices used in safety-critical systems may have to be considered for certification. For example, Forsberg & Karlsson (2006) discuss the COTS CPU selection for safety-critical applications, and Salewski and Taylor (2007) compare fault handling in FPGAs and microcontrollers for such application. General certification criteria for databuses are discussed by Rierson and Lewis (2003), and certification issues for one specific databus, AFDX, by Kornecki (2008).

Other industries made respective attempts as well, dated back as far as 1994 (Hughes & Musgrave, 1994) for automotive industries and the use of FPGAs. More recently, papers appeared discussing certification of electronic equipment for a gas burner (Gonçalves et al., 2002), micro-electro-mechanical systems, MEMS (White & Rios, 2002), and a radio altimeter (Hairion et al., 2007). Lundteigen and Rausand (2006) made one of a few attempts to look at hardware assessment for safety-critical systems from the perspective different than that of DO-254, by applying the IEC 61508 and 61511 standards.

There have always been tendencies to bring together programming languages and silicon, including implementations of a language in silicon (Schoeberl, 2004, 2008) but also using a programming language to design silicon and generate code for programmable logic devices (Hilton & Hall, 2004). As one author states it, "Software consists of bits downloaded into a prefabricated hardware device. Traditional microprocessor software bits represent sequential instructions to be executed by a programmable microprocessor. In contrast, field-programmable gate array software bits represent a circuit to be mapped onto an FPGAs configurable logic fabric" (Vahid, 2007). Thus, it seems like the "bits" loaded into the microprocessor's memory is as much software as the "bits" loaded into an FPGA.

Finally, there is a vast amount of standards across various industries that deal with software certification, but not much with hardware, with one exception. The international standard IEC 60601-1-4 (2000), on Programmable Electrical Medical Systems, states explicitly that it "goes beyond traditional testing and assessment of the finished medical electrical equipment and includes requirements for the processes by which medical electrical equipment is developed." This is an important step in the discussion on product versus process certification, since the standards further states: "Testing of the finished product is not, by itself, adequate to address the safety of complex medical electrical equipment."

The IEC standard takes as its basis the concepts of risk management and a development life cycle, on which it builds the procedures for design assurance. In particular, the guidance addresses requirements specification, architecture, detailed design and implementation, modification, and verification and validation. What seems to be a significant shortcoming is that the document redefines all basic concepts related to safety and system development, making no reference to any of the computing or engineering glossaries, where these terms have been previously defined. This fact makes the document look not very much related to other existing standardization documents, even though the definitions might be compatible with those previously formulated in other standards.

## 4. Formal approaches to H/W verification

A thorough review of literature for the last decade, or so, reveals a number of attempts to formalize reasoning about hardware. Only a handful of selected papers are analyzed below. An interested reader can find further references in two book collections on the subject (Hu & Martin, 2004; Bernardo & Cimatti, 2006). For a practical implementation, one can look at a recent paper focusing on verifying an IEEE-754 floating-point exponential function (Akbarpour et al., 2009). For the purpose of this discussion, we follow the definition of a formal method as given by the NASA Langley Formal Methods Group (NASA Lagley Formal Methods Group, 2008) :

"Formal Methods" refers to mathematically rigorous techniques and tools for the specification, design and verification of software and hardware systems. The phrase "mathematically rigorous" means that the specifications used in formal methods are well-formed statements in a mathematical logic and that the formal verifications are rigorous deductions in that logic (i.e., each step follows from a rule of inference and hence can be checked by a mechanical process.)

### 4.1. Historical perspective

In some of the earlier papers, Hoskote et al. (1997) addressed the problem of verifying the correctness of gate-level implementations of large synchronous sequential circuits with respect to their higher-level specifications in HDL. The verification strategy is to verify containment of the finite state machine (FSM) represented by the HDL description in the gate-level FSM by computing pairs of compatible states. Their formulation of the verification problem dissociates the verification process from the specification of initial states, whose encoding may be unknown or obscured during optimization and also enables verification of reset circuitry. Consequently, verification of circuits with large and diverse I/O sets, which was previously intractable due to lack of a single

effective variable order for the binary decision diagrams is now feasible.

In another older paper Kern and Greenstreet (1999) point out to two main aspects of the application of formal methods in a hardware design process: (a) the formal framework used to specify desired properties of a design, and (b) the verification techniques and tools used to reason about the relationship between a specification and a corresponding implementation. They survey a variety of frameworks and techniques proposed in the literature and applied to actual designs. The specification frameworks include temporal logics, predicate logic, abstraction and refinement, as well as regular languages. The verification techniques presented include model checking, automata-theoretic techniques, automated theorem proving, and approaches that integrate the above methods. They present a selection of case studies where formal methods were applied to industrial-scale designs, such as microprocessors, floating-point hardware, protocols, memory subsystems, and communications hardware.

O'Leary, Zhao, Gerth, & Seger (1999) gave an account of a practical exercise of formally verifying Intel processors. They have reported on there principal lessons having been learned. First, they claim that it has been feasible and practical to formally verify gate-level descriptions of floating-point hardware in the timeframe of a major processor design project. Secondly, they confirmed that verifying floating-point hardware against IEEE-level specification required significant effort and investment in terms of two full-time formal verification experts over five quarters and a third full-time expert added in the final quarter. Finally, they report that devising and executing floating-point formal verification strategies required very specialized expertise in floating-point architecture and algorithms.

Bunker, Gopalakrishnan, & McKee (2004) reviewed one aspect of almost every hardware design, that is, protocol compliance verification. The paper presents a survey of candidate modeling languages for protocol verification, focusing on languages originally intended for hardware and software design and verification activities. The comparison is framed by first constructing taxonomy of these languages, and then by discussing the applicability of each approach to the compliance verification problem. Each discussion includes a summary of the development of the language, an evaluation of the language's utility for the problem domain, and examples of how the language might be used to specify hardware protocols.

Only relatively recently authors of papers on formal methods began considering tools supporting these approaches. A handful of related papers are discussed below.

Turner & He (2001) investigate specification, verification and test generation for synchronous and asynchronous circuits. Their approach is called Digital Logic In LOTOS (the ISO language of temporal ordering specification), or DLIL for short. They defined relations for strong conformance to verify a design specification against a high-level specification, and developed tools for automated testing and verification of conformance between an implementation and its specification.

Aljer & Devienne (2004) consider the use of a formal specification language as the foundation of real validation process. They propose architecture based upon stepwise refinement of a formal model to achieve controllable implementation. Partitioning, fault tolerance, and system management are seen as particular cases of refinement in order to conceptualize systems correct by proven construction. The basic principles of system methodologies are presented and the methodology based on the refinement paradigm is described. In order to prove this approach, the B-HDL tool based on a combination of VHDL and B method formal language has been developed.

Nehme & Lundqvist (2003) describe a framework consisting of both software tools for application verification and hardware platforms for execution and real-time monitoring. The tool translates safety-critical VHDL code into a formal representation in a form of FSM model. Different formal techniques can then be applied on this representation in order to verify properties such as liveness and deadlock and to validate that the timing constraints of the original system hold. Three aspects of the tool implementation are discussed: transformation of source code into an intermediate representation, verification of real-time properties, and some tool-related implementation issues.

Dajani-Brown et al. (2004) focus on the use of SCADE (Safety-Critical Application Development Environment) and its formal verification component, the Design Verifier, to assess the design correctness of a sensor voter algorithm used for management of three redundant sensors. The algorithm, captured as a Simulink diagram, takes input from three sensors and computes an output signal and a hardware flag indicating correctness of the output. Since synthesis of a correct environment for analysis of the voter's normal and off-normal behavior is a key factor when applying formal verification tools, this paper is focused on: (1) the different approaches used for modeling the voter's environment; (2) the strengths and shortcomings of such approaches when applied to the discussed problem.

Hilton in his thesis (2004) proposes a process for developing a system incorporating both software and PLD, suitable for safety-critical systems of the highest levels of integrity. This process incorporates the use of Synchronous Receptive Process Theory as a semantic basis for specifying and proving properties of programs executing on PLD, and extends the use of SPARK Ada to cover the interface between software and programmable logic. The author claims that the demonstrated methods are not only feasible but also scale up to realistic system sizes, allowing development of such safety-critical software–hardware systems to the levels required by current safety standards.

### 4.2. Impact of DO-254 guidelines

Karlsson and Forsberg (2005) discuss the additional design assurance strategies stated in DO-254, appendix B - "Design assurance considerations for level A and level B functions." In particular, the use of formal specification languages such as the property specification language (PSL) in combination with dynamic (simulation) and static (formal) verification methods for programmed logic devices are addressed. Using these methods, a design assurance strategy for complex programmable airborne electronics compliant with the guidelines of DO-254 is suggested. The proposed strategy is a semi-formal solution, a hybrid of static and dynamic assertion based verification. The functional specification can be used for both documentation of requirements and verification of the design's compliance. It is possible to tightly connect documents and reviews to present a complete and consistent design/verification flow.

In a more current paper (Karlsson & Forsberg, 2008), the same authors make an attempt to take a unifying approach to various functional verification strategies. Recognizing that the most difficult part of using effectively formal methods tools is writing the formal properties and constraining the verification effort, they propose a careful planning and validation process based on the use of UML, which use allows for graphical visualization of the design intent. They propose assigning suitable verification strategies to the individual functional blocks in UML (such as a test plan, for example), and then break them down into functional trees that give more detailed understanding of the individual functions. They combine this concept with their own approach to meet certification requirements, called Overlapped Layered Modular Methodology (OLMM), involving the use of assertions for formal verification and simulation-based verification.

More recently, EDA vendors started recognizing the importance of formal design verifications, and respective publications began to appear at various venues. One such example, advocating using more formal approaches for DO-254 compliance, is presented by Synopsys (Marriott & Stone, 2009). They include and discuss several interesting observations, how DO-254 is relevant to design verification from their perspective. One of their thoughts is to look at "micro-coded digital hardware" (in the language of DO-254) from the analog and mixed-signal perspective, since "ultimately, all designs are essentially analog implementations in physical transistors". Therefore, "it is important that equivalent representations of circuit blocks yield the same functional results regardless of their level of abstraction."

From the point of view of formal approaches, Marriot and Stone claim that due to the complexity of all contemporary functional designs, using a direct test approach is impractical, therefore a constraint verification approach should be used. The best way to implement such approach is via inserting assertions in the design and running many tests with different random seeds. Furthermore, since for larger designs some assertions may become difficult to prove, and DO-254 requires demonstrating that gate-level netlists are equivalent to the RTL implementation, a formal approach called equivalence checking can be applied. This can be done either with static verification or through functional comparison of the design against a reference view (such as RTL view).

Foster et al. (2009) give an interesting overview of the prospective use of formal methods in DO-254 related programs. After explaining the principal features and discussing the value of formal methods, they present a list of essential items, called formal data checklist, which should be present for an effective use of a formal method, including:

- Description of the formal methods approach.
- Formal statement of requirements (properties).
- Formal model (VHDL or any other design code).
- Proofs, results, traceability.
- Tools and assessment documentation.
- Counter-examples, new tests and properties in subsequent iterations.
- Formal methods results at the conclusion.

Discussing further the primary uses of formal methods, where to use them and where not, as well as outlining the common misconceptions and objections to formal methods, the paper presents recommendations on formal methods application:

- Allow applicants to use formal methods on DO-254 projects.
- Avoid tools that encourage a user to guide the tool towards proof.
- Before running model checking (for credit), clarify the applicability of formal methods to specific properties.
- During model checking tool usage apply precautions (rerunning it after design changes, choosing as few constraints as possible, and choosing the "starting state")
- During certification, use the formal verification checklist (presented above in this paper).

With the emergence of the FAA endorsed DO-254 document, more papers began to appear that discuss not only also compliance with the DO-254 standard but also tool support for formal approaches. This is where some discussions of product or process certification and tool qualification begin to take place. Related papers are discussed in more detail in the next section.

## 5. Tool qualification against DO-254

The increasing complexity of electronics hardware requires the use of automatic software tools. The DO-254 document includes a section on tool qualification. Tool qualification is the process necessary to obtain certification credit for use of a tool within the context of a specific airborne system. The document distinguishes between design tools, which can introduce errors into the product, and verification tools, which do not introduce errors into the product but may fail detecting errors in the product. The DO-254 tool assessment and qualification process is shown in Fig. 1.

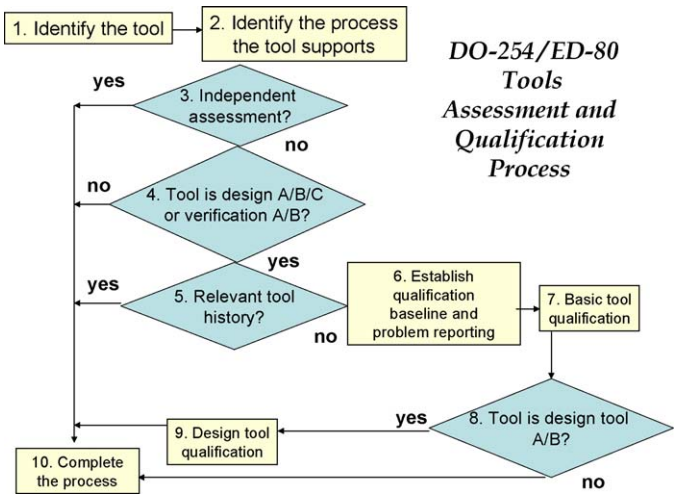


**Fig. 1.** Principle of the tool qualification process (RTCA, 2000).

Several vendors recently began dealing with tool qualification. Aldec (2007) used a sample design of a counter, with the following features: one clock domain, asynchronous reset, clock enable port, counting direction port (up/down), synchronous initial value reload ability, 64 bits output data. The system contained two boards connected through Daughter Board (DB) connectors. First of them – the main board – was Aldec HES board (HES3X3000EX) connected to the PCI bus. This board generated stimuli for Design Under Test (DUT) and collected results from DUT. The second board was a user DB with DUT.

Three independent stages of the verification process include: simulation, verification, and comparison. During simulation, using Active-HDL simulator, stimuli and results are captured in waveform on specified edge of user clock. The clock line of DUT is not stored in waveform file. It is generated on HES main board during verification in order to assure constant frequency. The *PrototypeVerificationTool* (PVT) program is used for hardware verification, which sends test vectors to DUT and retrieves response data from DUT. The two tasks: writing stimuli to *SinFIFO and* reading results from *RoutFIFO* are the basis for the verification process. Results from *RoutFIFO* are written to a raw binary file, and then transformed to a waveform. At the comparison stage, using Aldec waveform viewer, the waveform captured during simulation is compared with waveform from hardware verification. No differences indicate that verification has finished successfully.

For the tool qualification, Aldec advocates in their process named Compliance Tool Set (CTS, Aldec 2008) applying only the first three steps from Fig. 1. In step 1, "Identification of the Tool", caution is suggested, in case the tool version changes, all relevant activities should be repeated (for example, HDL simulations re-run). For step 2, "Identification of the Process the Tool Support", emphasis is placed on the need to include information about the outputs produced by each tool. In step 3, answering the question "Is the Tool Output Independently Assessed", the following the following methods of independently assessing the simulation tool outputs are listed:

- Manual review (for smaller projects), comparing the expected results with those actually produced by the simulator.

- Use of another simulator, which is the case in CTS, which comes with two HDL simulators, Active-HDL and Riviera-Pro, that can be used for assessment of other vendors' simulators outputs, with a variety of hardware design languages.
- Comparing outputs of the assessed tool with another tool's outputs (as stated in DO-254), for example, verifying the synthesis and place and route tools by comparing the outputs of in-hardware testing to the outputs of post-synthesis and timing level simulations.

Metastability occurs in digital circuits when the clock and data inputs change values at approximately the same time, which leads to flip-flop output oscillating, that is, going "metastable". Lange (2008) addresses circuit metastability in the context of DO-254 tool certification. Such situation may occur in designs with multiple asynchronous clocks, when two or more discrete systems communicate. Metastability is a severe problem in safety-critical designs as may be the cause of intermittent failures. A comprehensive verification solution is offered by Mentor Graphics *0-In Clock Domain Crossing (CDC)* tool that provides metastability verification solution:

- A structural analysis of the RTL code identifies and analyzes all signals crossing clock domains, and determines if their synchronization schemes are present and correct.
- Simulation transfer protocols are monitored and verified to assure that the synchronization schemes are used correctly.
- Global checks for re-convergence are performed, by injecting the effects of potential metastability into the simulation environment and determining how the design will react.

The *0-In CDC* tool provides added assurance that the design will function correctly within the intended system. The independent output assessment (see Fig. 1) is the suggested method of tool assessment to verify the system clock domain crossings and identify and eliminate instances of metastability.

Lange (2007) discusses another tool from Mentor Graphics, *ModelSim*, which is considered a verification tool, since it does not generate the code to be used in the production circuit. *ModelSim* is used for digital simulation of directed test cases and provides coverage data. The paper outlines ten steps to follow the DO-254 assessment and qualification process, as presented in Fig. 1. Tool qualification is bypassed by using an independent output assessment (Step 3 in Fig. 1). For *ModelSim* it can be done by:

- Reviewing RTL simulation outputs for their match against synthesized (gate-level) simulations; in case of match, the likelihood of an error in *ModelSim* is extremely low.
- Applying selection of identical tests as done in simulation on the physical device; if the results match, then one can logically conclude that the tool, *ModelSim*, is correctly simulating the model.

According to the author, the above two steps constitute argument that the DO-254 tool assessment criteria are met.

The verification steps/techniques must be performed in concert with the RTL design, ultimately leading to automatic circuit synthesis, which is the subject of another paper by Lange & Dewey (2008). Since automatic synthesis and conversion to gate-level designs is often done with optimizations by the hardware design tools, it may be counterproductive in safety-critical designs, which require strict adherence to the requirements. For this and other reasons DO-254 has rather rigorous requirements on tool qualification, "to ensure that tool used to design and verify hardware perform to an acceptable level on confidence on the target project."

The paper comments on three methods of DO-254 allowed tool assessment: relevant history, independent output assessment, and tool qualification. Since proving relevant history and qualifying the tool are both tedious processes, requiring the submittal of data, which may not be easily available, the paper suggests that the way to go is to demonstrate that "the hardware item must be thoroughly verified against the functional requirements", thus, the independent tool assessment is not necessary. In the opinion of the authors of this review, this may not be the right thing to do, since the tool output is still an abstract entity, not the hardware item yet, and may contain errors that cannot be detected during verification.

In yet another paper from Mentor Graphics, Thatte and Lange (2009) are focusing on FPGA synthesis tools. They look at the DO-254 compliance related aspects beyond the traditional considerations, such as design optimization, and make suggestions regarding the following:

- Finite state machine (FSM) synthesis for single event upset protection.
- Redundancy control for single-point failure avoidance.
- Managing late-state design changes, which may inadvertently impact the product quality.

TNI (Baghai & Burgaud, 2006) presents a technical note how the *Reqtify* tool complies with the DO-254 objectives. The tools supports requirement traceability, impact analysis and automated documentation generation. The functionality of *Reqtify* includes: requirement coverage analysis, upstream and downstream impact analysis, requirement change, update and deletion tracking throughout the project life cycle, requirement attribute handling, filtering and display depending on these attributes, user configurable documentation generation, and regression analysis.

According to DO254 classification, Reqtify is a verification tool. Prior to the use of the tool, a tool assessment should be performed. The purpose of tool assessment and qualification is to ensure that the tool is capable of performing the particular activity to an acceptable level of confidence for which the tool will be used. It is only necessary to assess those functions of the tool used for a specific hardware life cycle activity, not the entire tool. The assessment activity focuses as much or more on the application of the tool as the tool itself. Verification tool only needs to be qualified if the function that it performs is not verified by another activity. The flow chart from DO-254 is applied and indicates the tool assessment considerations and activities and provides guidance for when tool qualification may be necessary.

Dellacherie et al. (2003) describe a static formal approach that could be used, in combination with requirements traceability features, to apply formal methods in the design and verification of hardware controllers to support such protocols as ARINC 429, ARINC 629, MIL-STD-1553B, etc. Their paper describes the application of a formal tool, *imPROVE-HDL*, in the design and verification of airborne electronic hardware developed in a DO-254 context. *imPROVE-HDL* is a formal property checker that complements simulation in performing exhaustive debugging of VHDL/Verilog RTL hardware models of complex avionics protocol controllers without the need to create testbenches. *Reqtify* tool is used to track the requirements throughout the verification process and to produce coverage reports. According to the authors, using *imPROVE-HDL* coupled with *Reqtify*, avionics hardware designers can assure that their bus controllers meet the most stringent safety guidelines outlined in DO-254.

One additional issue is worth mentioning here, because it poses certain challenges to DO-254 based tool qualification. It is a combined use of conventional software development tools, such as MATLAB, with typical EDA tools, as described by Anderson

(2009a,b). MATLAB, having abstractions such as complex numbers, matrix operations, built-in libraries for DSP functions and waveform analysis, is a very desirable tool for DSP algorithm development suitable for the FPGA implementation. This is a quintessential problem, because FPGA designers normally have limited knowledge of the application domain and those who develop algorithms need not have significant insight into FPGA technology.

In addition, as Anderson points out, there exists a significant gap between these algorithmic abstractions and hardware, and the question arises: can MATLAB be used to create executable specifications for hardware? The author's answer is positive, provided appropriate verification is done. The author then presents a discussion of the use of Xilinx AccelDSP synthesis tool capable of automatically generating a C/C++ model from a floating-point MATLAB model, and provides a working example.

## 6. Tool questionnaire

A survey has been conducted to collect user feedback regarding the use of programmable logic tools as applied to design and verification of complex electronic hardware according to the DO-254 guidelines. The objective was to capture the experience and collect opinions from industry and certification authorities, related to the assessment and qualification of the tools.

The questionnaire has been developed and distributed at several national and international conferences, including those organized by the FAA over the last two years. In addition we followed-up with a mailing to over 150 individuals engaged in development of the aviation software and hardware. The questionnaire was also distributed internally in a few companies engaged in the design of programmable logic devices, and has been made available from the DO-254 Users Group website (http://www.do-254.org/?p=tools). As a result of these activities a sample of almost forty completely filed responses was received. Even though this may not be a sample fully statistically valid, the collected results make for several interesting observations.

The majority of respondents represented avionics or engine control developers (65%). Over 95% of respondents have technical background (55% bachelor and 45% master degrees), with over 72% having educational background in electronics. While 97% of respondents have more than three years of experience, 59% have more than 12 years. The most frequent respondents' roles relevant to the complex electronics tools include:

- Use of the tools for development or verification of systems (60%).
- Managing and acting as company's designated engineering representative (26%).
- Development of components (12%).
- Development of the tools (2%).

The respondents' primary interest was divided between verification (32%), development (27%), hardware (22%) and concept/architecture (18%).

Considering criteria for the selection of tools for use in DO-254 projects (Table 1), as the most important have been reported the following: the available documentation, ease of qualification, previous tool use, and host platform, followed by the quality of support, tool functionality, tool vendor reputation, and the previous use on airborne project. Selection of a tool for the project is based either on a limited familiarization with the demo version (50%) or an extensive review and test (40%). The prevailing approach is to review and test the tool by training of the personnel and by using a trial period on a smaller project.

From those who have actually experienced effort to qualify programmable logic tools (only 14% of respondents, Fig. 2), the quality of the guidelines is sufficient or appropriate (62%, Fig. 3), so is the ease of finding required information (67%), while the increase of workload was deemed negligible or moderate (80%, Fig. 4). An interesting observation concerns the scale of safety improvement due to qualification: marginal (43%), moderate

**Table 1**
Importance of tool evaluation criteria (from highest to lowest).

| Importance | Criterion |
|---|---|
| 1 | Documentation quality |
| 2 | Ease of qualification |
| 3–4 | Previous use on airborne products |
| | Host platform |
| 5 | Quality of support |
| 6 | Reliability |
| 7 | Functionality |
| 8–12 | Tool performance |
| | Internal evaluation |
| | Previous familiarity with the tool |
| | Availability of training |
| | Compatibility with development platform |
| 13 | Vendor reputation |
| 14 | Compatibility with existing tools |
| 15–17 | Acquisition cost |
| | Amount of training needed |
| | Compatibility with selected PLDs |
| 18 | Other criteria |

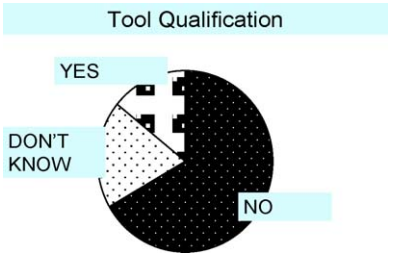


**Fig. 2.** Experience of respondents with tool qualification.

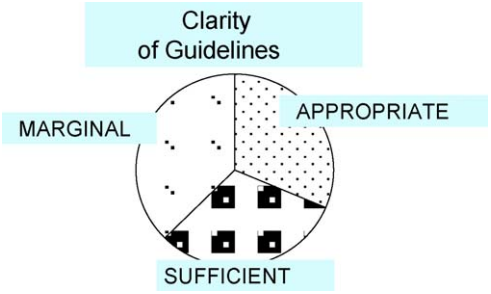


**Fig. 3.** Distribution of responses on quality of tool guidelines.

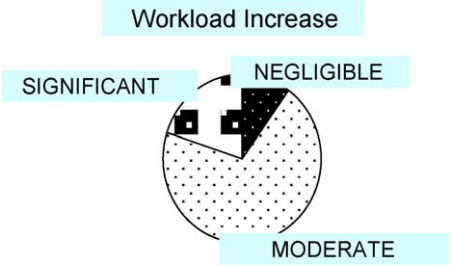


**Fig. 4.** Responses on workload increase due to tool qualification.

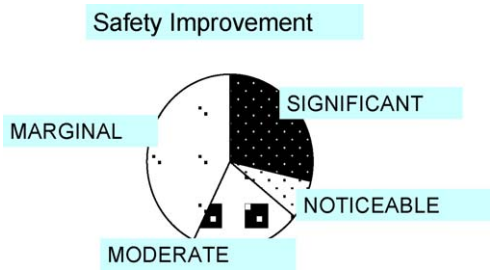


**Fig. 5.** Responses on safety improvement due to tool qualification.

(21%), noticeable (7%) and significant (29%)—see Fig. 5. Similarly, the question about errors found in the tools may be a source for concern: no errors (11%), few and minor errors (50%), significant and numerous (17%).

The respondents observed that the nature of tools is better suited for rapid freeform development than for requirement-based methodical design. The major issues with tool use and qualification identified by the respondents in a comments section of the survey included:

- flawed simulation not accounting for hardware effects
- incorrect timing analysis
- lack of tool conformance to published IEEE standards
- marginal visibility and traceability of tool output
- hidden advanced tool features
- no access to proprietary tool design data
- no access to tool simulation and synthesis algorithms
- lack of detailed guidance on what is acceptable for qualification
- lack of vendor independent standard test suite for tools
- frequent tool updates with marginal version control.

Despite all raised issues, the satisfaction level towards programmable logic tools was high: more than 96% of respondents marked their satisfaction level as 4 out of 5.

In summary, it is evident that software tools used in design and verification of complex electronics in safety-critical applications should be scrutinized because of concerns that they may introduce design errors leading to potential safety violations and accidents. However, the conducted survey indicated that the most important criteria for tool selection are considered to be: available documentation, ease of qualification, and previous tool use, none of which is technical. In this view, work should be done on developing more objective criteria for tool qualification and conducting experiments with tools to identify their most vulnerable functions that may be a source of subsequent design faults and operational errors. *Some of the authors specifically point out that the* lack of research investment in certification technologies will have a significant impact on levels of autonomous control approaches that can be properly flight certified, and could lead to limiting capability for future autonomous systems.

The tool assessment process must follow the DO-254 guidelines, but the relative vagueness of these guidelines causes significant differences in interpretation by industry and should be eliminated. Possibly, a common ground should be found between DO-254 and DO-178B guidelines.

## 7. Summary and conclusion

The subject of DO-254 is electronic hardware. The software verification, software/hardware integration verification, and system integration verification processes are not addressed in DO-254. However, verification of hardware requirements during these processes is a valid method of hardware verification (RTCA, 2000). Any complete airborne system must be certified. The system definition and subsequent safety analysis determine which part of the system functionality is allocated to hardware and which part to software. Subsequently, the guidance of either DO-178B or DO-254 needs to be applied. The software falls under guidelines of DO-178B. However, addressing the hardware part is not that simple. We have HDL code (that is, software) representing the circuit with its interconnections, as well as software running on a workstation used to simulate the circuit behavior and analyze its performance. The guidance of DO-254 does not address these issues. The paper's main point can be summarized that the use of modern software-based tools, while designing electronic circuits, effectively moves the point of emphasis from hardware to software.

There is a significant evidence of a widespread use of the FPGA devices in safety-critical systems, developed with the aid of tools we discuss in this paper. However, despite the fact that often design and verification can be done by the same individual, the level of verification may be insufficient due to limited selection of external stimuli, simulations may be skipped, and the design approved by non-engineering managers. Additionally, the research shows that the quality of vendor-supplied soft core or macro libraries is not guaranteed, while synthesis tools can generate faults.

Considering the above, there is no surprise that vendors want to stay on the safe side. The excerpts from manufacturers' product descriptions are symptomatic. Despite great progress and improved trustworthiness of new products, there is no certainty that the product is perfect. This leads to the limited warranties and legal disclaimers, such as the one below (NASA Office of Logic Design 2004):

IN NO EVENT SHALL <vendor> OR ITS LICENSORS OR THEIR AGENTS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL OR INCIDENTAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTIONS, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF <vendor> AND/OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES

When designing circuits for implementation in programmable logic, it is essential to use dedicated programs. A typical tool's functions would include software for initial design entry, logic optimization, device fitting, simulation, and configuration. There are tools specific to just one type of hardware such as analog design, there are tools that serve multiple purposes, and there are dedicated ASIC/FPGA tools. Tools common to analog and fixed digital systems are rather uncomplicated, with a thin layer of abstraction between the users input and the tool's output. A designer can input the design, similar to CAD-like drawing, then test and simulate it for minor post processing and simplifications. ASIC/FPGA tools, by contrast, have a thick layer of abstraction between the user's input and the tool's output. The designer operating the tool commonly enters the design in a language such as Verilog or VHDL. The tool then interprets the language, synthesizes the logic, creates net lists, and interprets the net list into a hardware specific layout. Synthesis involves typically the optimization of logic, timing, and various other aspects. It can be thought of as a black box, with design as an input and "synthesized" design as an output.

FPGA/CPLD vendors are primarily interested in developing software required to take a design captured either in schematics or HDL into a form that can be used to program a circuit. However, because the Electronic Design Automation (EDA) tool industry is fairly dynamic and hardware keeps evolving, the software developers for back-end tools have to attend to two primary activities, developing libraries for new EDA tools and simulators,

while creating better fitters and routers for new hardware with more resources and more complex architectures.

Evolving interchange standards such as EDIF and Verilog/VHDL help standardize interfaces to CAD tools and simulators. For example, a CAD vendor can now provide an EDIF-compatible library of design elements using Verilog or VHDL to implement the models necessary for simulation environments. Recognizing that there is a potentially large learning curve, FPGA/CPLD vendors are also offering cost-effective entry-level design environments. The CEH tool landscape is very diverse and not standardized. The proprietary and closed nature of the tools makes them rather difficult to evaluate. Therefore, new evaluation criteria are needed that would address technical aspects of tool use.

## Acknowledgements

The presented work was supported in part by the Aviation Airworthiness Center of Excellence under contract DTFACT-07-C-00010 sponsored by the FAA. Findings contained herein are not necessarily those of the FAA. The authors are grateful to the anonymous reviewers for constructive comments.

## References

Akbarpour, B., Abdel-Hamid, A.T., Tahar, S., and Harrison, J. (2009). Verifying a synthesized implementation of IEEE-754 floating-point exponential function using HOL. The Computer Journal (10 April), 1–24.

Aldec Inc. (2007). DO-254 Hardware verification: prototyping with vectors mode. White Paper, Rev. 1.2. Henderson, Nevada, June 26, 2007.

Aldec Inc. (2008). Tool Assessment and Qualification with the Aldec DO-254 Compliance Tool Set, Rev. 2.0. Henderson, Nevada, August 29, 2008.

Aldec Inc. (2009). Design Verification Methodology with the Aldec DO-254 Compliance Tool Set, Rev. 2.0. Henderson, Nevada, February 24, 2009.

Aljer, A., & Devienne, P. (2004). Co-design and refinement for safety critical systems. Proceedings of DFT 04' 19th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems. *IEEE, 2004*, 78–86.

Altera Corporation (2008). DO-254 support for FPGA design flows. White Paper. San Jose, CA, July 2008.

Anderson, R. (2009a). *Using MATLAB to Create an Executable Specification for Hardware*. San Jose, CA: Xilinx Inc.

Anderson, R. (2009b). *Algorithm/Hardware C-Design Using MATLAB*. San Jose, CA: Xilinx Inc.

ASSC. (2009). *IPT Guidance for Acquisition of Systems with Complex Programmable Hardware Using DO-254, January*. Leatherhead, UK: ERA Technology Ltd.

Baghai, T., Burgaud, L. (2004). DO254 Package Process and Checklists: Overview & Compliance with RTCA/DO-254 Document. White Paper, DO-254 Users Group, March 2004.

Baghai, T., & Burgaud, L. (2006). *Reqtify: Product Compliance with RTCA/DO-254 Document*. Caen, France: TNI-Valiosys.

Bernardo, M., Cimatti, A. (Eds.) (2006). Formal methods for hardware verification. Proc. SFM 2006. 6th International School on Formal Methods for the Design of Computer, Communication, and Software Systems. Bertinoro, Italy, May 22–27, 2006. Lecture Notes in Computer Science, vol. 3965. Springer-Verlag, 2006.

Bunker, A., Gopalakrishnan, G., & McKee, S. A. (2004). Formal hardware specification languages for protocol compliance verification. *ACM Transactions on Design Automation of Electronic Systems 9(January (1)).*

Civera, P., Macchiarul, L., Rebaudengo, M., Sonza Reorda, M., & Violante, M. (2002). An FPGA-based approach for speeding-up fault injection campaigns on safety-critical circuits. *Journal of Electronic Testing: Theory and Applications, 18*, 261–271.

Cole, P., & Beeby, M. (2004). Safe COTS graphics solutions: impact of DO-254 on the use of COTS graphics devices for avionics. In *Proceedings of DASC 23rd Digital Avionics Systems Conference, October 24–28, 2004* 8A2-8.1/7.

Dajani-Brown, S., Cofer, D. Bouali, A. (2004). Formal verification of an avionics sensor voter using SCADE. Proceedings of FORMATS 2004 Joint International Conference on Formal Modelling and Analysis of Timed Systems, and FTRTFT 2004 Formal Techniques in Real-Time and Fault-Tolerant Systems. Lecture Notes in Computer Science, vol. 3253, pp. 5–20.

Dellacherie, S., Burgaud, L., & di Crescenzo, P. (2003). Improve – HDL: a DO-254 formal property checker used for design and verification of avionics protocol controllers. In *Proceedings of DACS'03, 22nd Digital Avionics Systems Conference, October 12–16, 2003, vol. 1* pp. 1.A.1-1.1-8.

Dewey, T. (2008). Demistifying DO-25. White Paper. Wilsonville, Ore.: Mentor Graphics, March 2008.

FAA (2005). Advisory Circular AC 20-152, RTCA Document RTCA/DO-254 Design Assurance Guidance for Airborne Electronic Hardware, Federal Aviation Administration, June 30, 2005.

Forsberg, H., & Karlsson, K. (2006). COTS CPU selection guidelines for safety-critical applications. *Procedings of 25th DASC, Digital Avionics Systems Conference, October 15–19, 2006* pp. 4A3-1/12.

Foster, H., Landoll, D., Lange, M. (2009). Understanding formal methods for use in DO-254 programs. White Paper, Rev. 1.0. Mentor Graphics Corp., Wilsonville, Ore., July 2009. (presented at the 2009 National FAA Software and Airborne Electronic Hardware Conference). San Jose, CA, August 18–20, 2009).

Fulton, R. (2006). RTCA/DO-254 data package for commercial-off-the-shelf graphical processors. In *Proceedings of 25th DASC, Digital Avionics Systems Conference, October 15–19, 2006* pp. 6E6-1/6.

Glazebrook, I. (2007). The Certification of Complex Hardware Programmable Logic Devices (PLDs) for Military Applications. White Paper, DNV UK, London.

Gonçalves, F. M., Santos, M. B., Teixeira, I. C., & Teixeira, J. P. (2002). Design and test of a certifiable asic for a safety-critical gas burner control system. *Journal of Electronic Testing: Theory and Applications, 18*, 285–294.

Hairion, D., Emeriau, S., Combot, E., & Sarlotte, M. (2007). New safety critical radio altimeter for airbus and related design flow. In *Proceedings of DATE'07, 2007 Design, Automation & Test in Europe Conference & Exhibition, April 16–20, 2007* (pp. 688–694).

Hilderman, V., & Baghai, T. (2003). Avionics hardware must now meet same FAA requirements as airborne software. *COTS Journal, 5*(September (9)), 32–36.

Hilton, A. J. (2004). High-Integrity Hardware-Software Codesign. Ph.D. Thesis. The Open University, April 2004.

Hilton, A., & Hall, J. G. (2000). *On Applying Software Development Best Practices to FPGAs in Safety-Critical Systems*. The Open University.

Hilton, A. J., & Hall, J. G. (2004). High-integrity interfacing to programmable logic with Ada. In *Proceedings of Ada-Europe 2004 9th International Conference on Reliable Software Technologies, June 14–18, 2004* (pp. 249–260).

Hoskote, Y. V., Abraham, J. A., Fussell, D. S., & Moondanos, J. (1997). Automatic verification of implementations of large circuits against HDL specifications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 17*(3), 217–227.

Hu, A. J., Martin, A. K. (Eds.) (2004). Formal methods in computer-aided design. Proceedings of 5th International Conference, FMCAD 2004, November 15–17, 2004. Austin, Texas. Lecture Notes in Computer Science, vol. 3312, Springer-Verlag, 2004.

Hughes, R. B., & Musgrave, G. (1994). Formal CAD techniques for safety-critical FPGA design and development in embedded systems. In *Proceedings of FPL'94, International Workshop on Field Programmable Logic and Applications, September 7–9, 1994* (pp. 135–137).

IEC (2000). IEC 60601-1-4, Edition 1.1b:2000 Medical Electrical Equipment - Part 1-4: General Requirements for Safety. Collateral Standard: Programmable Electrical Medical Systems. Geneva: International Electrotechnical Commission, April 2000.

Karlsson, K., & Forsberg, H. (2005). Emerging verification methods for complex hardware in avionics. In *Proceedings of DASC 2005, 24th Digital Avionics Systems Conference, 30 October–3 November 2005, vol. 1* pp. 6B.1–61-12.

Karlsson, K., & Forsberg, H. (2008). Structured assertion design verification for complex safety-critical hardware. In *Proceedings of MAPLD'08, Military and Aerospace Programmable Logic Devices Conference, September 15–18, 2008*.

Keithan, J., Landoll, D., Marriott, P., & Logan, B. (2008). The use of advanced verification methods to address DO-254 design assurance. In *Proceedings of 2008 IEEE Aerospace Conference, March 1–8, 2008*.

Kenny, J. R. (2008). Team effort is central focus in DO-254 compliance. *COTS Journal, 10*(August (8)), 40–44.

Kern, C., & Greenstreet, M. R. (1999). Formal verification in hardware design: a survey. *ACM Transactions on Design Automation of Electronic Systems, 4*(2), 123–193.

Knaus, C. (2004). OpenGL ES: Safety-Critical Profile Philosophy, July 2004.

Kornecki, A. (2008). Airborne software: communication and certification. *Scalable Computing: Practice and Experience, 9*(1), 77–82.

Kornecki, A., & Zalewski, J. (2008). Software certification for safety-critical systems: a status report. In *Proceedings of IMCSIT2008/RTS'08 Real-Time Software Workshop, October 20, 2008* (pp. 665–672).

Kornecki, A., & Zalewski, J. (2009). Certificataion of software for real-time safety-critical systems: state of the art. *Innovations in Systems and Software Engineering: A NASA Journal, 5*(June (2)), 149–161.

Lange, M. (2007). Assessing the ModelSim Tool for Use in DO-254 and ED-80 Projects. White Paper. Wilsonville, Ore.: Mentor Graphics Corp., May 2007.

Lange, M. (2008). Automated CDC verification protects complex electronic hardware from metastability issues. *VME Critical Systems, 26*(August (3)), 24–26.

Lange, M., Boer, T. J. (2007). Effective functional verification methodologies for DO-254 level A/B and other safety-critical devices. White Paper, Rev. 1.1. Wilsonville, Ore., 2007.

Lange, M., & Dewey, T. (2008). Achieving quality and traceability in FPGA/ASIC flow for DO-254 aviation projects. In *Proceedings of 2008 IEEE Aerospace Conference, March 1–8, 2008*.

Lee, C. (2007). *IPT Guidance for Acquisition of Systems with Complex Programmable Hardware Using DO-254 Ref. 7D0134813, June*. Leatherhead, Surrey, UK: Avionics Systems Standardisation Committee and ERA Technology Ltd.

Lee, M., & Dewey, T. (2007). Accelerating DO-254 for ASIC/FPGA designs. *VME and Critical Systems, 25*(June (3)), 28–30.

Le Mauff, J., Elliott, J. (2009). Meeting DO-254 and ED-80 guidelines when using Xilinx FPGAs. White Paper. San Jose, CA: Xilinx Inc., January 26, 2009.

Leroy, J. E., Bezamat, J. (2007). Experience at Barco-Silex in FPGA Design with DAL C (DO254). Barco-Siles S.A., Peynier, France, Internal Paper.

Lundquist P. (2007), Certification of Actel Fusion according to RTCA DO-254. Master Thesis, Report LiTH-ISY-EX-ET-07/0332-SE. Sweden: Linköping University, May 4, 2007.

Lundteigen, M. A., & Rausand, M. (2006). Assessment of hardware safety integrity requirements. *Proceeings of 30th ESReDA European Safety, Reliability and Data Association Seminar on Reliability of Safety Critical Systems, June 7–8, 2006*.

Mahapatra, R. N., et al. (2006–2009). Microprocessor Evaluations for Safety-Critical Real-Time Applications. Phase 1, 2 and 3. Technical Reports DOT/FAA/AR-06/34, DOT/FAA/AR-08/14, DOT/FAA/AR-08/55. Washington, DC: Federal Aviation Administration.

Marriott P., Stone, A.D. (2009). Understanding DO-254 Compliance for the Verification of Airborne Digital Hardware. White Paper. CA: Synopsys Inc., Mountain View, October 2009.

Miner, P. S., Carreño, V. A., Malekpour, M., & Torres, W. (2000). A case-study application of RTCA DO-254: design assurance guidance for airborne electronic hardware. In *Proceedings of DASC 2000, 19th Digital Avionics Systems Conference, October 7–13, 2000, vol. 1* pp. 1A1/1-1A1/8.

NASA Office of Logic Design (2004). Design Guidelines and Criteria for Space Flight Digital Electronics. Section XII. Review of Digital Electronic Circuits. http://klabs.org/DEI/References/design_guidelines/nasa_guidelines/review/review.htm.

NASA Lagley Formal Methods Group (2008). What Is Formal Methods? http://shemesh.larc.nasa.gov/fm/fm-what.html.

Nehme, C., & Lundqvist, K. (2003). A tool for translating VHDL to finite state machines. In *Proceedings of 22nd Digital Avionics Systems Conference, October 12–16, 2003, vol. 1* pp. 3.B.6-3.1-7.

O'Leary, J., Zhao, X., Gerth, R., & Seger, C-J.H. (1999). Formally verifying IEEE compliance of floating-point hardware. *Intel Technology Journal, 3*(1), 1–14.

Pampagnin, P., Menis, J. F. (2007). DO254-ED80 for High Performance and High Reliable Electronic Component. Barco-Siles S.A.. Peynier, France, Internal Paper.

Plastow R. A. (2007). Filling the Assurance Gap on Complex Electronics. Proceedings of 2nd IASS Conf. on Space Safety in a Global World, Chicago, May 14–16, 2007. Report SP-645. European Space Agency, July 2007.

Quantum3D (2007). IGL – A Certifiable Software Based OpenGL SC GPU. White Paper, Version 1.0. San Jose, CA, April 2007.

Reeve, T., & Lange, M. (2008). *DO-254 Compliance: Reducing Project Cost by Avoiding Common Pitfalls, September*. Wilsonville, Ore: Mentor Graphics Technical Library.

Rierson, D., & Lewis, J. (2003). Criteria for certifying databuses on civil aircraft. In *Proceedings of DASC'03, 22nd Digital Avionics Systems Conference, October 12–16, 2003* pp. 1.A.2-1/9.

RTCA. (1992). *DO-178B (EUROCAE ED-12B)*. Software Considerations in Airborne Systems and Equipment Certification, December, Washington, DC: RTCA Inc.

RTCA. (2000). *DO-254 (EUROCAE ED-80)*. Design Assurance Guidance for Airborne Electronic Hardware, April, Washington, DC: RTCA Inc.

Sakaide, Y., Nemoto, N., Kariu, K., Midorikawa, M., Iide, Y., Ichikawa, M., et al. (2005). Evaluation of Actel FPGA Products by JAXA. In *Proceedings of 2005 Annual MAPLD International Conference, September 7–9, 2005*.

Salewski, F., & Taylor, A. (2007). Fault handling in FPGAs and microcontrollers in safety-critical embedded applications: a comparative survey. In *Proceedings of DSD 2007 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools, August 29–31, 2007*.

Schoeberl, M. (2004). Java technology in an FPGA. In *Proceedings of the FPL 2004 International Conference on Field-Programmable Logic and its Applications, August 30–September 1, 2004*.

Schoeberl, M. (2008). A Java processor architecture for embedded real-time systems. *Journal of Systems Architecture, 54*(January/February (1/2)), 265–286.

Snyder, M. (2008). Designing a safety-certifiable OpenGL software CPU. *VME and Critical Systems, 26*(December (5)), 20–22.

Sysenko, I., & Pragasam, R. (2007). Hardware-based solution aides: design assurance for airborne systems. *Military Embedded Systems*, July: 26–28.

Thatte, S., & Lange, M. (2009). FPGA synthesis tools meet the DO-254 challenge. *VME and Critical Systems, 27*(April (2)), 24–26.

Turner, K. J., He, J. Formally-based Design Evaluation (2001), Proceedings of CHARME 2001, 11th IFIP WG 10.5 Advanced Research Working Conference on Correct Hardware Design and Verification Methods. Lecture Notes in Computer Science, vol. 2144, pp. 104–109.

Vahid, F. (2007). It's time to stop calling circuits "Hardware". *IEEE Computer, 40*(September (9)), 106–108.

White, E., & Rios, J. A. (2002). FAA certification of a MEMS attitude and heading reference system. In *Proceedings of 2002 Institute of Navigation National Technical Meeting, January 28–30, 2002*.

Young, D. (2004). RTCA/DO-254: no hiding place for avionics suppliers? *VMEbus Systems 22(February (1))*.

Zalewski, Z. (2007). Embedded systems verification—where FPGA meets processor. *Embedded Control Europe ECE Magazine,* April: 37–38.

Zalewski, Z. (2008). *Traditional Approach vs In-Hardware Simulation: Aldec DO-254 Compliance Tools Set (CTS), September*. Henderson, Nevada: Aldec Inc.

**Andrew J. Kornecki** is a Professor at the Dept. of Computer and Software Engineering, Embry Riddle Aeronautical University, in Daytona Beach, Florida, USA. He has over twenty years of research and teaching experience in areas of real-time computer systems. He contributed to research on intelligent simulation training systems, safety-critical software systems, and served as a visiting researcher with the FAA. He has been conducting industrial training on real-time safety-critical software in medical and aviation industries and for the FAA Certification Services. Recently he has been engaged in work on certification issues and assessment of development tools for real-time safety-critical systems.

**Janusz Zalewski** is a Professor of Computer Science and Engineering at Florida Gulf Coast University, in Fort Myers, Florida, USA. Before taking a university position, he worked for various nuclear research institutions, including the Data Acquisition Group of Superconducting Super Collider and Computer Safety and Reliability Center of Lawrence Livermore National Laboratory. He also worked on projects and consulted for a number of private companies, including Lockheed Martin, Harris, and Boeing. He served as a Chairman of IFIP Working Group 5.4 on Industrial Software Quality and of an IFAC TC on Safety of Computer Control Systems. His major research interests include safety-related real-time computer systems