STRUCTURED ASSERTION DESIGN VERIFICATION FOR COMPLEX SAFETY-CRITICAL HARDWARE

Kristoffer Karlsson, Engineering Consultant, EADS, Germany
Håkan Forsberg, Saab Avitronics, Jönköping, Sweden
09/18/2008
MAPLD 2008
Introduction

 Previously proposed a **RTCA/DO-254 compliant methodology** for **complex Level A and B designs**

- **Using assertion simulation** and **Formal Verification (FV)**

 Tested methodology on a **ARINC 429 interface design demonstrator** [MAPLD 2006]

- **Gained experience** in the use of assertions both for simulation and FV

- **Careful planning needed** to use these verification strategies effectively
Previous Experiences of FV

- FV increases reliability of functional verification

- **FV requires experience** (formal notation, proofs and debugging)

- FV increases functional coverage but **it is not possible to measure total coverage** seamlessly with other coverage measurements used

- **FV not fully automatic** (formal specification, constraints, manual guidance of proofs)

- FV not practical for regression testing (unless design is simple)

- Not **plausible to prove the whole design** behavior

- **Deep understanding** of the design is **needed** (Requires special consideration to maintain independent verification)
In Recent Years ...

- Many papers on how to deal with certification guidelines of DO-254
- Tool and chip vendors shown interest in the area
- Assertion based verification (ABV) become more widely used (in simulation)
- Usage of Formal Verification still sparse
Managing Complexity

Harder to test PLD designs exhaustively within a foreseeable amount of time

Make better designs from the start:

• **Focus on early parts of design cycle** - requirements capture and validation
• Assure **good code quality** - rules and guidelines.
• Maintain **good design documentation**
• Spend verification efforts where it helps the design process - **supportive verification**
• Use **assertions** to give **instant feedback** to engineering (simulation/FV)
Advanced Verification Strategies

- A good quality of design **not sufficient for complex safety-critical avionics designs**

- Advanced verification methods for complex level A and B designs proposed by RTCA/DO-254:
  - Safety-Specific Analysis (SSA)
  - Elemental Analysis (EA)
  - Formal Methods (FM)

- Many other functional verification strategies available
Functional Verification Strategies

- Assertion simulation
- Formal verification (FV) (static/dynamic)
- Constrained random simulation
- High-level languages and base libraries for writing test benches
- Various coverage measures
Assertions

- PSL or SystemVerilog assertions
- Formalizes functional design requirements
  - Clear and unambiguous syntax
  - Temporal operators to describe discrete events and more expressive test cases
- Used in Assertion Based Verification to:
  - Monitor design behavior during simulation
  - Calculate formal proofs / counterproofs
Assertion Based Verification 1/2

ABV of design functionality of great use but it has some fundamental shortcomings:

- Large amount of assertions makes it hard to reason about the functional coverage obtained
- High ‘assertion density’ does not mean that the actual functionality (or what is required of it) has been verified
- Only discrete events can be considered

Writing correct and complete properties is the hardest part of using FV effectively!
Thereby, ABV does not by default provide convincing evidence that functional requirements have been fulfilled

- Assertions should be validated against the functional design requirements
- Assertions should be associated with and analyzed within the context of the overall functionality
- A Safety-Specific Analysis may be needed to constrain formal proof calculation
Formalizing Requirements

- Formal specification **suffers from limitations of the design documentation**
  - Focus on early parts of the design cycle to allow independent verification

- Impossible to write fully covering functional requirements for a complex design
  - Would mean to describe the whole design functionality deterministically

- Assertions can monitor/prove **key elements** but also be used **more extensively**
  - **Requires proper planning** - for example using UML modeling
Functional Modeling

What models are used for:

- “To capture and to precisely state requirements and domain knowledge so that all stakeholders may understand and agree on them.”

- “To think about the design of the system.”

- “To master complex systems.”

Functional Test Plan

Block-level model of design functions

• Assists in assuring that all vital functions and interfaces are covered by requirements and in verification

• Makes planning of verification strategies explicit
  • Verification planning outline
  • Assign appropriate verification strategies to individual functions of the design

• Functional blocks further broken down hierarchically into functional test trees
  • (compare with Elemental Analysis)
Functional Test Trees 1/2

- Requirements, design intent, formal properties and verification strategies can all be reviewed/validated in relation to each other.
- Supports ABV (simulation/formal) as functional properties, constraints and coverage needed can be identified.
- Makes it possible to weigh simulation/FV coverage measures against each other.
- Requirements can be imported/updated from the common requirements capture environment at set baselines.
With functional test trees **fewer assumptions** are made on the design behavior

- **Independent verification** (level A and B designs) has **disadvantage of insight** into the implemented functionality
- Verification engineers might be **unaware of what they are missing** due to lack of information
- Assures that functional behavior of the design is **correctly interpreted when formalized in assertions**
Schematic Example

- Function of particular significance
  - **FV used** to give additional confidence of events in state C

1. **Constraints used to limit input space** of proof calculation defined

2. **Requirements that shall hold under the given preconditions**

- Hierarchical level of proof/counterproof shown in tree
Conclusions

Proposed functional verification planning approach:

- Gives better possibility to **analyze the design intent and its implications on the functional verification**
- **Assign appropriate verification strategies** to functions where they are best suited
- Validate functional requirements within their context
- Validate correctness and completeness of assertions
- **Weigh simulation and formal coverage** against each other
- **Planning of FV** (constraints, hierarchical level of formal proof)