Making your ISO 26262 Flow Flawless

Establishing Confidence in Verification Tools

Bryan Ramirez
DVT Automotive Product Manager

August 2015
What is Tool Confidence?

- **Principle:** If a tool supports any process governed by ISO 26262 (i.e., if the activities rely on a correct output from the tool), then the user must be able to rely on the correct functioning of the tool.

- A malfunction could mean — Introducing a bug into the product — Failing to find a bug in the product

- This is a two part process
  1. Provide information to determine level of confidence needed
  2. Demonstrate qualification of a tool when applicable

See ISO 26262-8, Section 11
Why Should You Care About Tool Confidence?

- ISO 26262 requires that you establish tool confidence and qualify as necessary

- IC development tool chains are complex with many different tools
  - Many stages in design process
  - Many ways to verify each step

- Tool qualification can be a large effort
  - Safety critical flows already add significant overhead
  - Needs to be planned for accordingly – strategy, time and money

- Tool qualification doesn’t ultimately add incremental value to your product
  - Leverage tool vendors as much as possible to ease this effort

- Varying levels of expertise within the industry
  - Mature organizations will be more efficient with this process
A Typical Semiconductor Flow

Chip Development Flow

Capture Requirements -> Create Design Concept -> Create RTL Design -> Synthesize Design -> Insert Test Structures

Perform Place & Route -> Manufacture ASIC

Program FPGA

Evaluate HW
A Typical Semiconductor Flow

**Chip Development Flow**

- **Capture Requirements**
- **Create Design Concept**
- **Create RTL Design**
- **Synthesize Design**
- **Insert Test Structures**
- **Perform Place & Route**
- **Evaluate HW**
- **Manufacture ASIC**

**Verification and Validation**

- **Requirements Tracing**
- **Modeling**
- **Simulation**
- **Simulation**
- **Simulation**
- **Manufacture Testing**
- **Test Harness**
- **Lab Equipment**

- **Requirements Validation**
- **Analysis & Reviews**
- **Formal Methods**
- **Formal Methods**
- **Static Timing Analysis**
- **Analysis & Reviews**
- **Emulation & Prototyping**
A Modern Verification Suite
So Many Challenges, So Many Tool Choices

- Stimulus based (dynamic)
- Formal methods (static)
- Emulation
- Prototyping — Virtual — FPGA
- Different approaches all linked with a common infrastructure
- The methods, combinations, and feature sets are endless
- Each flow is unique
Determining Tool Confidence Level

See ISO 26262-8, Section 11

Tool Error Detection

Tool Impact

Tool Confidence Level
Determining Tool Confidence Level:

**Tool Impact**

- Tool Impact (TI): "the possibility that a malfunction of a particular software tool can introduce or fail to detect errors in a safety-related item or element being developed"

**Tool Impact:**
Can the software tool malfunction such that it introduces or fails to detect errors of safety requirements?

---

<table>
<thead>
<tr>
<th>TD1</th>
<th>TD2</th>
<th>TD3</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TI1</th>
<th>TCL1</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TI2</th>
<th>TCL2</th>
<th>TCL3</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Determining Tool Confidence Level:

Tool Error Detection

- Tool error Detection (TD): “confidence in measures that prevent the software tool from malfunctioning and producing corresponding erroneous output, or in measures that detect that the software tool has malfunctioned and has produced corresponding erroneous output”

Tool Error Detection:
Degree of confidence that the software tool’s malfunction and its corresponding erroneous output will be prevented or detected?

<table>
<thead>
<tr>
<th>TD1</th>
<th>TD2</th>
<th>TD3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TCL1</td>
<td></td>
</tr>
<tr>
<td>TCL1</td>
<td></td>
<td>TCL2</td>
</tr>
<tr>
<td>TCL1</td>
<td>TCL2</td>
<td>TCL3</td>
</tr>
</tbody>
</table>

Tool Impact:
Can the software tool malfunction such that it introduces or fails to detect errors of safety requirements?

- No
- Yes

Tool Confidence Level

- High
- Medium
- Low / None
Determining Tool Confidence Level:

**Tool Confidence Level**

- TCL1 – No further tool qualification activities needed
- TCL2 / 3 – Formalized tool qualification required
- TCL can be improved by enhancing development process

**Tool Error Detection:**
Degree of confidence that the software tool’s malfunction and its corresponding erroneous output will be prevented or detected?

- High
- Medium
- Low / None

**Tool Impact:**
Can the software tool malfunction such that it introduces or fails to detect errors of safety requirements?

- No
- Yes

**Tool Confidence Level:**
Level of confidence that the software tool malfunctions will not lead to the violation of safety requirements.

- No Qual Needed
- Qual Required
Determining Tool Confidence Level

**Tool Impact:**
Can the software tool malfunction such that it introduces or fails to detect errors of safety requirements?

**Tool Error Detection:**
Degree of confidence that the software tool’s malfunction and its corresponding erroneous output will be prevented or detected?

- **No**
- **Yes**
- **High**
- **Medium**
- **Low / None**

**Tool Confidence Level:**
Level of confidence that the software tool malfunctions will not lead to the violation of safety requirements.

- **No Qual Needed**
- **Qual Required**
The Challenges of Defining the TCL

- How do you define High, Medium and Low tool detection confidence?

- No quantitative means are provided (by the document)

- The TCL determination process can be quite subjective!

- The details as to “how” this should be done are left as an exercise for the tool user

- The approach defined by the tool user must then be reviewed and accepted by the end product customer’s Safety Manager

- Most tool users:
  1. Really don’t want to have to figure this out
  2. May not have confidence in their analysis/approach
A Practical Approach to Establishing Confidence Levels

- Since no specific methods are provided by ISO 26262 to quantitatively determine confidence levels, why not use an approach well known to the safety community?

**Failure Modes and Effects Analysis**

- Failure Modes and Effects Analysis = FMEA
- FMEA is an inductive reasoning approach involving single point of failure analysis that has been a core task of Safety Engineering since the 1950’s
- It involves identifying failure modes (of components and systems), along with their causes and effects
- The failure modes and their resulting effects on the rest of the system are recorded in a specific FMEA worksheet
- This same analytical approach can be applied to Tool Confidence!
Tool FMEA to Establish TD Level

- A tool “Failure Mode” is any situation where a faulty tool output could result in a created/missed design bug in the end product.

- A team of tool experts can evaluate the tool to determine what these possible situations might be.

- For each “Failure Mode” identified, internal or external means may be available to either detect or prevent a resulting design bug.
  - Internal means may involve tool development processes, tool use limitations, or tool self-check situations.
  - External means may involve message reviews or downstream design flow activities.

- Individual failure modes are captured and analyzed for:
  - How severe the potential failure is (severity)
  - How often does it occur (occurrence)
  - How it can be detected (detection)

- This quantitative approach applies a numerical rating of 1-10 and these numbers are multiplied together to create the Risk Priority Number (RPN).

- Industry guidance provides High, Medium and Low ranges for RPN.
## ISO 26262 Tool Qualification Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>TCL3</th>
<th>TCL2</th>
<th>TCL1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIL A</td>
<td>ASIL B</td>
<td>ASIL C</td>
<td>ASIL D</td>
</tr>
<tr>
<td>1a</td>
<td>Increased confidence from use in accordance with 11.4.7</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>1b</td>
<td>Evaluation of the tool in accordance with 11.4.8</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>1c</td>
<td>Validation of the software tool in accordance with 11.4.9</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>1d</td>
<td>Development in accordance with a safety standard(^a)</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

++ = highly recommend for the identified ASIL  
+ = recommend for the identified ASIL  

\(^a\) – No safety standard is fully applicable to the development of software tools. Instead, a relevant subset of requirements of the safety standard can be selected.

EXAMPLE – Development of the software tool in accordance with ISO 26262, IEC 61508, or RTCA DO-178.

ASIL = Automotive Safety Integrity Level  
A = Lowest level  
D = Highest level
Tool Qualification Strategies

- **Increased Confidence from Use (1a)**
  - This requires extensive use with the same (or very similar) version, constraints, uses cases, environment, etc.
  - Any tool issues should be documented, monitored, and appropriately worked around if needed

- **Evaluation of Tool Development Process (1b)**
  - The tools must be developed to comply with an appropriate standard (e.g., Automotive SPICE, CMMI, ISO 15504)
  - Process should be evaluated and proper application of the assessed development process shall be demonstrated

- **Validation of Software Tool (1c)**
  - Verifying the tool performs as expected against its requirements
  - Typically done by testing functional and non-functional aspects

- **Development in Accordance with a Standard (1d)**
  - The tool itself is developed in such a way as to be compliant to the relevant aspects of a safety standard (e.g. ISO 26262, IEC 61508 or RTCA DO-178)
  - NOTE: This is very rare

- **Note that a combination of 1b and 1c is the most common approach by far for ASIL C & D**
ISO 26262 requires a *Software Tool Criteria Evaluation Report* for each tool or tool chain used.

This report must include:

1. Description of the tool
2. Planned usage of the tool
   1. Version, configuration, and environment of the tool
   2. How it's used in the flow (with specific use cases)
   3. The maximum ASIL of the project it will be used on
3. An evaluation of TI, TD, and ultimately the TCL classification
Documenting the Tool Confidence Analysis

- Regardless of whether qualification is needed, the Tool Confidence analysis process needs to be captured and presented for review within the Safety program.

- The type of information includes:
  - Description of the tool
  - Intended purpose
  - Tool identification and version
  - Tool operational environment
  - Inputs and outputs
  - Tool configuration
  - Tool restrictions
  - Tool use cases
  - Environmental or functional constraints
  - Level of Safety of design (that tool is being used on)
  - Tool errata
  - Tool impact
  - Tool detection
  - Tool confidence level (which may involve a FMEA analysis as described)
  - Tool qualification approach (if needed)

Mentor Graphics can provide tool users with a documentation kit to facilitate this process.
What Does a Validation Approach Entail?

- Documentation of the Tool Confidence Analysis (as described previously) PLUS
- A documented Tool Qualification Plan
- Tool requirements
  - Functional and non-functional requirements that need to be verified
- A tool test suite
  - Mapped to the requirements being tested
  - These requirements should be tied to those specific failures left undetected through internal/external means
- A way to run the tests and capture results
  - So the tool user can run the validation suite in the project environment
- A tool qualification summary
  - Documenting the process and results
Tool Confidence Example:  
A Requirements Management Tool

- Requirements management software is used to trace functional and safety requirements from definition through implementation with correlation to results.

- The tool use is as follows:
  - Requirements are entered into the tool via the user or imported through other requirement tools.
  - The tool traces requirements through implementation and correlates results back to requirements.
  - The tool outputs metrics regarding the completeness of each requirements.

- It’s Tool Impact is TI1
  - The tool does not create design code and cannot create an error in the design.
  - The tool does not verify the design and thus cannot fail to detect a design error.

- Its Tool Error Detect is TD1
  - Based on the FMEA analysis.
  - And the documented use in the Safety Manual.
  - The RPN number falls into the LOW range.

- The Tool Confidence Level is determined to be TCL1.

<table>
<thead>
<tr>
<th>Tool Impact</th>
<th>Tool Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI1</td>
<td>TCL1</td>
</tr>
</tbody>
</table>
| TI2         | TCL1 | TCI2 | TCL3 |}

TCL1: No qualification necessary.
Tool Confidence Example: 
A Simulator

- Simulation is to verify RTL and gate-level models of the design during development.

- The tool use is as follows:
  - Stimulus is created (i.e., test cases that verify the requirements)
  - The tool executes the test cases on the design model
  - The tool outputs the behavior of the model to the test cases (and compares to expected responses)

- It’s Tool Impact is TI2
  - The tool does not create design code and cannot create an error in the design, BUT...
  - The tool does verify the design, and can potentially fail to detect a design error

- Its Tool Error Detect is TD1
  - Based on the FMEA analysis
  - And the documented use in the Safety Manual
  - The RPN number falls into the LOW range

- The Tool Confidence Level is determined to be TCL1

---

TCL1: No qualification necessary.
Tool Confidence Example:
A Formal Methods Application

- A clock domain crossing tool that uses formal methods to identify clock-domain crossing issues that could cause design metastability

- The tool use is as follows:
  - Cross clock domains are identified and formally checked for correctness against asynchronous domain crossing models

- It’s Tool Impact is TI2
  - The tool does not create design code and cannot create an error in the design, BUT...
  - The tool does verify the design, and can potentially fail to detect a design error

- Its Tool Error Detect is TD1
  - Based on the FMEA analysis
  - And the documented use in the Safety Manual
  - The RPN number falls into the LOW range

- The Tool Confidence Level is determined to be TCL1

<table>
<thead>
<tr>
<th>Tool Impact</th>
<th>Tool Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI1</td>
<td>TD1</td>
</tr>
<tr>
<td>TI2</td>
<td>TD1</td>
</tr>
</tbody>
</table>

TCL1: No qualification necessary.
Summary

- Tools are a vital part of a safety program and confidence in them must be established.
- Different tools have different functions and therefore different potential impacts on the design.
- If a tool malfunction may result in a design bug (being created or missed in verification) this must be examined further.
- Internal and external means may be available to prevent or detect design bugs.
- Hazard analysis (FMEA/RPN) can add a familiar and structured quantitative aspect to this not-well-defined process.
- TCL1 is your friend...
- If sufficient means for detection/prevention aren’t available, then Tool Qualification is necessary.
- Several options exist for Tool Qualification.
- Tool vendors can assist tool users with this process.
Mentor is a leading supplier of both automotive software and EDA tools.

Mentor’s industry leading tools are being used by top automotive semiconductor suppliers.

Mentor is investing in ISO 26262 with several ISO 26262 experts on staff.

Mentor is assisting companies with ISO 26262 tool confidence needs.

Mentor is formalizing Tool Qualification kits and enhancing tools/flows to further facilitate our customers’ ISO 26262 processes.