Fusion Compiler™ Functional Safety Manual

November 2021, Revision 2.0

SYNOPSYS®

Copyright and Proprietary Information Notice

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.
All other product or company names may be trademarks of their respective owners.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
690 E. Middlefield Road
Mountain View, CA, 94043
www.synopsys.com

Document Control

Revision history

Version	Description	Date
1.0	First release of the document submitted for review.	8-Aug-2019
1.1	Second release of document for review. Misc updates.	12-Nov-2019
1.2	Third release for review – removed AoUs 100 and 101 and added to the Use cases the specific AoUs that were mentioned in these „Group AoUs“. Made the tool flow a bit smaller and sync‘ed it with other flow diagrams	09-Dec-2019
2.0	Updated for ISO26262 re-certitication	17-Nov-2021

Contents

1 Customer Support

This section describes the customer support that is available through the Synopsys SolvNetPlus® customer support website or by contacting the Synopsys support center.

Accessing SolvNetPlus

The SolvNetPlus support site includes an electronic knowledge base of technical articles and answers to frequently asked questions about Synopsys tools. The site also gives you access to a wide range of Synopsys online services, which include downloading software, viewing documentation, and entering a call to the Support Center.

To access the SolvNetPlus site:

1. Go to the web page at https://solvnetplus.synopsys.com/ .
2. If prompted, enter your username and password. (If you do not have a Synopsys username and password, follow the instructions to register.)

If you need help using the site, click **Help** on the menu bar.

Contacting Synopsys Support

If you have problems, questions, or suggestions, you can contact the Synopsys support center in the following ways:

- Go to the Synopsys Global Support Centers site on synopsys.com. There you can find e-mail addresses and telephone numbers for Synopsys support centers throughout the world.
- Go to either the Synopsys SolvNetPlus site or the Synopsys Global Support Centers site and open a case online (Synopsys user name and password required).

2
Scope of This Document

This section describes the scope of this document and defines terms used in this document.

Using This Document

The *Fusion Compiler Functional Safety Manual* describes the proper use of the Fusion Compiler tool in safety-related applications according to the ISO 26262 standard, and is intended to confirm the compliance of the Fusion Compiler tool to the standard when used in the context of a tool chain.

The Fusion Compiler tool enables the user to perform logic synthesis and physical implementation of very deep submicron designs. It provides the following functionality: logic synthesis, design planning, extraction and timing analysis, placement, optimization, clock tree synthesis, routing, design-for-manufacturing, and engineering change orders.

Section 3 describes an overview of the ISO 26262-8, clause 11 and the approach adopted by Synopsys to comply with the requirements of the standard. Section 4 defines the general information such as where to find the latest documentation and installation requirements regarding the use of the Fusion Compiler tool as a software tool in the development of safety-related applications. Section 5 shows the high-level overview of the tool chain that this product belongs to. Section 6 details the safety-related requirements for safety-qualified use cases of the Fusion Compiler tool. Section 7 lists the known limitations of the use cases.

Specific documentation for performing design and analysis as part of an ISO 26262 compliant flow is provided in Section 3, Section 5, Section 6 , Appendix A, and Appendix B of this document, the *Fusion Compiler Functional Safety Manual*.

Terms and Definitions

Term	Definition
AoU	Assumption of Use. An action that is assumed and required to be taken by the user of a software tool.

ASIL	Automotive Safety Integrity Level. This is a risk classification scheme defined by the standard ISO 26262. The standard identifies four levels: ASIL A, ASIL B, ASIL C, and ASIL D. ASIL D dictates the highest integrity requirements on a product and ASIL A dictates the lowest.
Component	A part of an electronic system that implements a function in a vehicle. See also Part 1 of the standard ISO 26262 for the definition. The standard also refers to elements and items, but for the *Fusion Compiler Functional Safety Manual*, there is no difference.
CoU	Condition of Use. A condition of the design, software tool, design environment, or situation that is assumed and required to be fulfilled by the user.
DCLS	Dual Core Lock Step A safety mechanism that can be implemented in functional safety that have the processors operating in parallel (e.g. in lock step). It is used for error detection of the cores of a single event upset (such as a soft error).
.db	A binary file format for storing logic library data.
DEF	Design Exchange Format. Standard file format for floorplanning data.
Defect	Product nonconformance.
DMR	Dual Modular Redundancy - This is a safety mechanism that can be implemented for functional safety. In most cases, it is used in the context of safety registers for error detection of a single event upset (such as a soft error).
Error	An error is a discrepancy between the actual and the specified or theoretically correct operation of an element. The root causes of an error can be manifold. In this document, the focus is on errors that are introduced or left undetected in a design, due to the malfunction in a software tool (e.g. generation of bad logic by a logic synthesis tool, failure of a static timing analysis tool to detect a timing violation).
Fault	An abnormal condition that can cause an element or item to fail.
Fault analysis	An analysis that determines the behavior of a system when a fault is introduced.

FC	Abbreviation for the tool name.
FFSM	Failsafe Finite State Machine State machine encoding that is used as a safety mechanism.
FMEA	Failure Mode and Effects Analysis. An analysis that looks at different parts of a system, identifies ways the parts could fail, and determines the causes and effects of these potential failures.
.frame	A binary file format for storing physical library data.
FuSa	Functional Safety
Fusion Compiler	The tool name.
HDL	Hardware Description Language Language for describing designs at RTL. Supports Verilog, SystemVerilog, and VHDL.
LVS	Layout-versus-schematic checking.
NDM	Proprietary database format used for design and library data in the Fusion Compiler tool.
RTL	Register transfer level. High-level description of the chip functionality.
SAIF	Switching Activity Interchange Format. Standard file format for switching activity data.
SCANDEF	Scan Design Exchange Format. Standard file format for scan data.
SDC	Synopsys Design Constraints. Standard file format for timing constraints.
SEU	Single Event Upset Transient faults that can occur in designs, such as impact from ion or electromagnetic radiation
Software / software tool	The Fusion Compiler tool.
Software tool criteria evaluation	Analysis according to ISO 26262 to determine the required TCL of a software tool.

Software tool qualification	Means to create evidence, that a software tool with low or medium TCL is suitable to be used in the development of safety related products according to ISO 26262.
SolvNetPlus	Synopsys customer support site.
SSF	Safety Specification Format
Standard	In this document, refers to *ISO 26262 Road Vehicles – Functional Safety*, 2011 and 2018 versions.
STAR	Synopsys Technical Action Request. A STAR documents and tracks a product Bug or Enhancement request (called a B-STAR or an E-STAR, respectively). It is stored in the Synopsys internal defect database. Only Synopsys employees can access the internal defect database. However, limited STAR information is available from SolvNetPlus for customers who are associated with the user site of a STAR. Customer contacts are notified automatically when a STAR is filed or when its status changes.
SVF	Synopsys Verification File This file is used as guidance for the Formality product.
TCL	Tool confidence level, as defined by ISO 26262-8, clause 11. Note: The TCL of a software tool does not necessarily indicate whether the tool may malfunction or not. The TCL defines the confidence level that an error in the safety-related design, which is introduced or left undetected by the software tool, can be prevented or detected in subsequent steps of the development flow, before the erroneous safety-related design is released.
Tcl	Tool Command Language. Command language used by the Fusion Compiler tool.
TD	Tool error detection, as defined in ISO 26262-8, clause 11.
TI	Tool impact, as defined in ISO 26262-8, clause 11.
TMR	Triple Modular Redundancy This is a safety mechanism that can be implemented for functional safety. In most cases, it is used in the context of safety registers for error correction of a single event upset (such as a soft error).
UPF	Unified Power Format.

	Standard format for power intent.
Use case	A use case is a specific way of using a software tool, that can be characterized by: - a limited set of tool functions and features that are used; - a set of restrictions and constraints that are regarded while using the tool; and - a specific goal to be achieved or output to be generated by using the software tool Use cases may be associated with different steps or phases in the design process, or they may describe alternative ways of using the tool for a specific design step.
Verilog	Standard format for the gate-level netlist.
WVGTECH	Internal representation of the RTL-level design.

3 Confidence in the Use of Software Tools According to ISO 26262-8, Clause 11

This section provides an overview of the ISO 26262-8, clause 11. It then describes the approach adopted by Synopsys to comply with the requirements of the standard, and how this is mapped to activities performed by Synopsys and the end user of the Synopsys tools.

Overview of ISO 26262-8, Clause 11

Synopsys EDA software tools contribute significantly to the design specification, implementation, integration, verification and validation of electrical and electronic (E/E) systems and components. If these E/E systems and components are used as part of a safety-related automotive product, an error in these systems or components could have severe consequences on functional safety. Such an error may arise as a result of unforeseen operating conditions or due to a fault introduced during product development, which in turn may be caused by a software tool malfunction. ISO 26262-8, clause 11 (Confidence in the Use of Software Tools) addresses this issue and specifies requirements and methods which aim to minimize the risk of faults in the developed product due to malfunctions of a software tool affecting the product's functional safety.

According to ISO 26262, to determine the required level of confidence in a software tool that is used in the development of a safety-related automotive product, the following criteria are evaluated:

- The possibility that the malfunctioning software tool and its corresponding erroneous output can introduce or fail to detect errors in a safety-related element being developed.
- The confidence in preventing or detecting such errors in its corresponding output.

This procedure is called Software Tool Criteria Evaluation, and it must be performed for all software tools that are involved in the development a safety-related element, resulting in a required Tool Confidence Level (TCL) for each software tool.

If the software tool criteria evaluation determines that a medium or high TCL is required, then appropriate Software Qualification methods must be applied, effectively reducing the risk of a critical software tool error. The choice of software qualification methods depends on the required TCL and the maximum ASIL of all the safety requirements allocated to the element developed using the software tool. However, if the software tool criteria evaluation determines that only a low TCL is required, then there is no need to apply such software qualification methods.

The software tool criteria evaluation and software tool qualification flow are summarized in Figure 1.

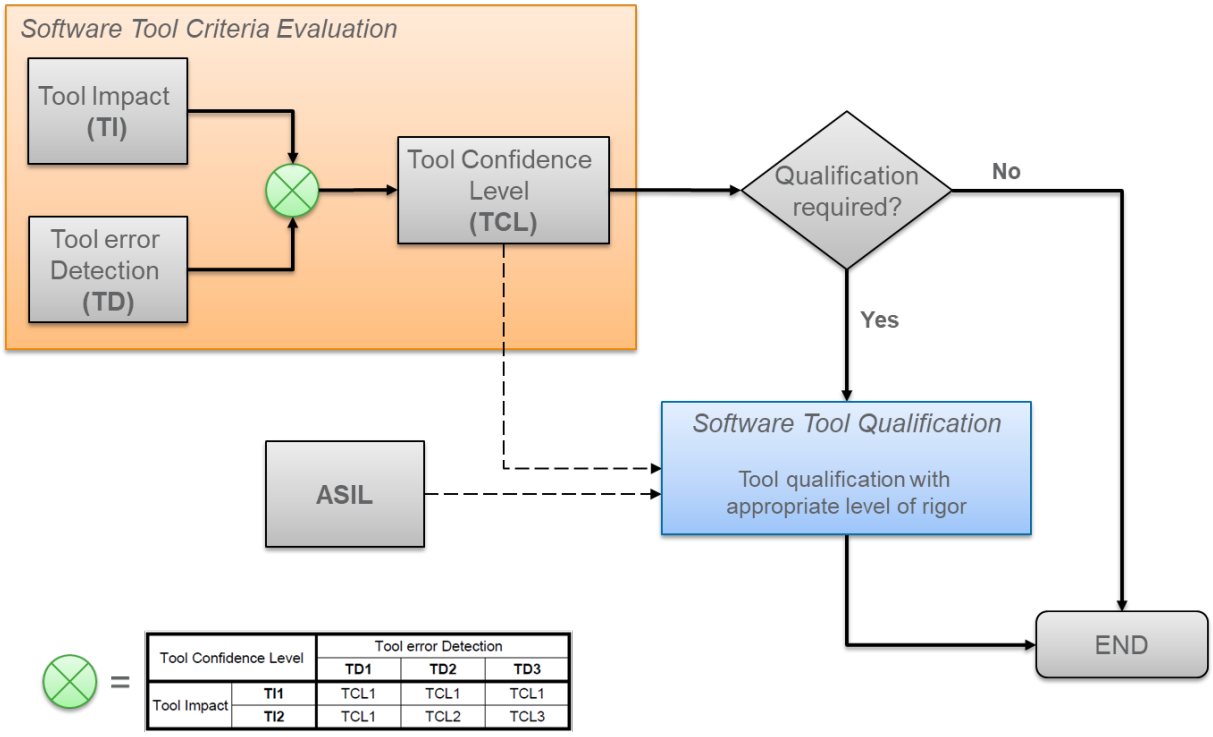

Figure 1: Software tool criteria evaluation and software tool qualification flow

Work Split Between Synopsys and Tool Users

A software tool criteria evaluation must always be performed in the development environment of the final tool user, and in the context of the actual product development. It is in this context, where potential tool malfunctions, their effect on the safety-related product, and the effectiveness of prevention and detection measures must be analyzed.

However, the tool vendor can support the tool user by performing a software tool criteria evaluation (and, if required, a software tool qualification) on their own, based on assumed tool use cases and an assumed development environment. If the assumptions made by the tool vendor match the actual situation at the tool user, then the user can take over the evaluation (and qualification) results from the tool vendor. Besides significantly reducing the effort for the tool user, this approach can also result in a better quality for the software tool criteria evaluation and qualification, since the tool vendor typically has a more detailed understanding of the inner working and possible malfunctions of the software tool.

Synopsys has adopted exactly this approach, which is summarized in Figure 2.

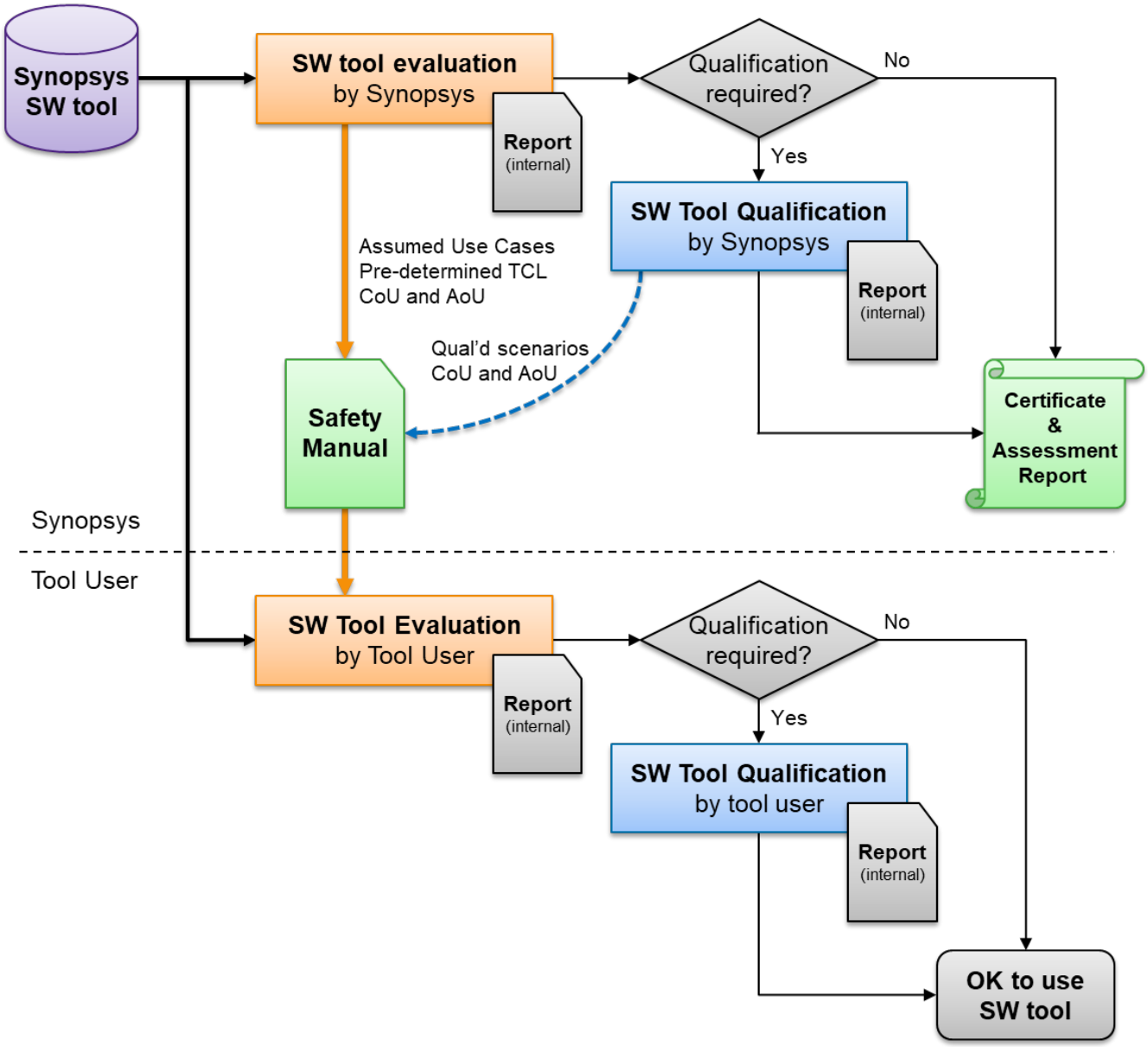

Figure 2: Work Split Between Synopsys and Tool Users

Synopsys performs the following activities:

1. Software tool criteria evaluation
   - Identification of possible **use cases** for the software tool, together with required **inputs** and expected **outputs**
   - Specification of **conditions of use (CoU)** for each use case, related to the development environment in which the tool is assumed to be deployed, including tool usage procedures and constraints
   - Analysis of potential software tool **malfunctions**, and their effect on a safety-related product that is developed with this tool
   - Analysis of **prevention** and **detection measures** internal to the software tool, to avoid tool malfunctions, or to control and mitigate their effects

- Specification of **assumptions of use (AoU)**, which are additional prevention and detection measures assumed to be performed by the end user of the tool
- Estimation of the **Tool Impact (TI)** for each malfunction, and the probability of **Tool error Detection (TD)** by the prevention and detection mechanisms (including assumptions of use)
- Determination of the required **Tool Confidence Level (TCL)** for each software tool malfunction, based on TI and TD
- Determination of the maximum TCL from all software tool malfunctions related to a use case. This is called the **pre-determined TCL** for the software tool use case
- Summary of the results in a software tool criteria evaluation report

2. Software tool qualification
   - If the pre-determined TCL indicates, that a medium (TCL2) or high (TCL3) tool confidence level is required for the software tool, then Synopsys may decide to perform a software tool qualification
   - The specific methods applied for tool qualification can vary for different tools and use cases, and they may include an evaluation of the software tool development process, the validation of the complete software tool, the validation of critical tool malfunctions with insufficient prevention and detection measures, or other methods
   - Summary of the qualification methods, procedures and results in a software tool qualification report

3. Safety manual for the software tool
   - The *Fusion Compiler Functional Safety Manual* (this document) is an important deliverable to the tool users, as it includes all end user-relevant information from the Synopsys software tool criteria evaluation and qualification
   - Software tool criteria evaluation related information, documented in Section 6, includes:
     - Description of software tool use cases
     - Description of the required inputs and expected outputs for each use case
     - Specification of conditions of use (CoU – conditions of the design, software tool, design environment, or situation that are assumed and required to be fulfilled by the user) for each use case
     - Specification of assumptions of use (AoU – actions that are assumed and required to be taken by the user of a software tool) for each use case
     - Pre-determined TCL for each use case
   - Software tool qualification related information (not required for this Fusion Compiler and therefore not included in this safety manual)
     - Description of the scope of the software tool qualification, including malfunctions and scenarios covered by the qualification
     - Specification of additional conditions of use (CoU) derived from the software tool qualification
     - Specification of additional assumptions of use (AoU) derived from the software tool qualification
   - Other information included in this safety manual
     - General information about the software tool needed by the tool user (see Appendix A)
     - Known limitations of the software tool, related to the described use cases as documented in Section 7

4. Certification and assessment report
   - Synopsys may decide to perform a functional safety assessment, to confirm the correctness, completeness and ISO 26262 conformance of the performed software tool criteria evaluation and qualification
   - Synopsys may also decide to achieve certification from an accredited third-party certification body, in addition to the functional safety assessment
   - The results of these activities are summarized in a functional safety assessment report and a certificate which can be viewed at exida Certificate for ISO 26262 Compliance

If the tool user wants to benefit from the work done by Synopsys, then according to the Figure 2 above, the user shall perform the following activities for each software tool:

1. Software tool criteria evaluation
   - Review and verify that the software tool criteria evaluation (and qualification) performed by Synopsys, as documented in the tool's Functional Safety Manual, matches the actual situation of the user's product development process
     - Verify whether the actual use case(s) of the software tool match those evaluated by Synopsys
     - Verify whether the actual inputs and outputs are identical to or a sub-set of those as evaluated by Synopsys
     - Verify that all conditions of use (CoU) specified by Synopsys are met, or whether the development process can be adjusted to meet these CoU(s)
     - Verify that all assumptions of use (AoU) specified by Synopsys are met, or whether the development process can be adjusted to meet these AoU(s)
     - Verify that the pre-determined Tool Confidence Level (TCL) for the relevant use case(s) are TCL1, *or*
     - Verify that Synopsys has successfully performed an additional software tool qualification for all TCL2 and TCL3 scenarios to conclude that the tool is suitable to be used for the development of a safety-related element of the same or higher ASIL than required by the user
   - If all the verification steps described above are successful, then the results of the Synopsys software tool criterial evaluation (and qualification) are applicable to the tool user, which means:
     - The required TCL pre-determined by Synopsys can be taken over by the tool user for actual product development
     - If the pre-determined TCL is TCL1, then the tool can be used without the need to perform any additional software tool qualification
     - If the pre-determined TCL is TCL2 or TCL3, then the software tool qualification performed by Synopsys is sufficient, and the tool can be used without the need for further software tool qualification by the end user
   - All of the steps above must be documented in a software tool criteria evaluation report, including evidence for the successful conclusion of all verification steps, which may include reference to the Synopsys Functional Safety Manual, and optionally, to the Synopsys certification and assessment report

2. Software tool qualification
   - If any of the verification steps described above as part of the tool user's software tool criteria evaluation fails (e.g. different use case, CoU or AoU cannot be met, pre-determined TCL is not TCL1 and Synopsys has not performed a software tool qualification), then the user must perform his/her own software tool qualification
   - The specific methods applied for tool qualification are decided and planned by the tool user -- Synopsys does not recommend any specific methods or procedures
   - The summary of the qualification methods, procedures and results shall be documented in a software tool qualification report

4
Fusion Compiler Description

This section provides a general description regarding the use of the Fusion Compiler tool as a software tool in the development of safety-related applications and describes where to get the latest product documentation and the runtime environment required to use the Fusion Compiler tool.

Coverage

The *Fusion Compiler Functional Safety Manual* is intended to be used starting with the version P-2019.03 and later versions of the Fusion Compiler tool per the use cases presented in this document. In general, unless otherwise noted, the failure modes and detection mechanisms noted in the use cases presented in Section 6 are tool version independent.

Compliance with ISO 26262

The Fusion Compiler tool can be used in the development of safety-related elements according to ISO 26262, with allocated safety requirements up to a maximum Automotive Safety Integrity Level D (ASIL D), if the tool is used in the context of a tool chain and in compliance with this document, the *Fusion Compiler Functional Safety Manual*.

See the exida Certificate for ISO 26262 Compliance of Synopsys Fusion Compiler when used in a tool chain flow.

Product Documentation and Support

Comprehensive documentation for using the Fusion Compiler tool is provided on SolvNetPlus. The latest documentation for the Fusion Compiler tool can be accessed at Fusion Compiler Online Help on SolvNetPlus.

.

Specific documentation for performing design and analysis as part of an ISO 26262 compliant flow is provided in Section 3, Section 5, Section 6 and Appendix A of this document, the *Fusion Compiler Functional Safety Manual*.

Synopsys provides online customer support for the Fusion Compiler tool. See Section 1 for more information.

Installation and Supported Platforms

The installation of the Fusion Compiler tool must follow the guidelines in the *Synopsys® Installation Guide* as well as the specific *Fusion Compiler Installation Notes* document.

Users are required to download the tool executable and INSTALL_README from the SolvNetPlus site at https://solvnet.synopsys.com/DownloadCenter/dc/product.jsp.

Supported platforms and operating systems requirements:

- For installation instructions, see the *Synopsys® Installation Guide* at https://www.synopsys.com/install.
- For the latest supported binary-compatible hardware platform or operating system, including required operating system patches, see https://www.synopsys.com/qsc.
- If updates (including security patches) to computing environments (including operating systems) are backward compatible with previous versions of the computing environment used to test the Fusion Compiler tool, the results of the testing performed by Synopsys using such previous versions are applicable.

Additional information:

- For information about the compute platforms roadmap, go to https://www.synopsys.com/support/licensing-installation-computeplatforms/compute-platforms/compute-platforms-roadmap.html.
- For platform notices, go to https://www.synopsys.com/support/licensing-installation-computeplatforms/compute-platforms/platform-notice.html.
- For information regarding the license key retrieval process, go to https://solvnet.synopsys.com/smartkeys/smartkeys.cgi.

User Competence

To properly use the Fusion Compiler tool, a user must have a good understanding and working knowledge of the following:

- Electrical engineering and circuit design
- The ISO 26262 standard
- Documentation of the Fusion Compiler tool, such as the User Guide, at Fusion Compiler Online Help on SolvNetPlus.
- This Functional Safety Manual
- The published list of safety-related defects for the Fusion Compiler tool available at Fusion Compiler Safety-Related Issues Master List. (link this text to your product's SolvNetPlus article that lists Safety-Related Issues using the following path: https://solvnetplus.synopsys.com/s/article/Fusion-Compiler-Master-List-of-Safety-Related-Issues

- Applicability of the Fusion Compiler tool in the overall tool chain

Managing Known Safety-Related Defects

Synopsys maintains current information for every reported defect through STARs. The Fusion Compiler team evaluates each reported issue for potential impact on functional safety.

A list of all known safety-related defects for each release of Fusion Compiler is available on a SolvNetPlus knowledge base article and is referenced from the *Fusion Compiler Release Notes*.

Fusion Compiler users must assess, as part of their own software tool criteria evaluation, the potential impact of the known safety-related defects in their design and must ensure mitigation of any relevant safety-related defects.

Managing New Releases

Synopsys can release new versions of the Fusion Compiler tool at any time to extend its functionality or to fix defects. When a new version is available, notification is posted on the SolvNetPlus site. A subscription service is available for users to be notified of any new product releases.

When installing a new version of the Fusion Compiler tool, users must evaluate the impact of any known safety-related defects in their design by checking the accompanying *Fusion Compiler Release Notes* for the following:

- Any changes that apply to safety-related use cases
- List of known safety-related defects in the new version of the Fusion Compiler tool

In addition, users must refer to the latest version of this document, the *Fusion Compiler Functional Safety Manual*, available with the product release contents.

5 Synopsys Digital Tool Chain

This section provides an overview of where the Fusion Compiler is used in the tool chain.

The ISO 26262 standard provides a methodology and requirements for software tool criteria evaluation and qualification (see ISO 26262-8, clause 11). It applies to software tools used for the development of safety-related designs where it is essential that the tool operates correctly without introducing or failing to detect errors in the safety-related design.

The suitability of a software tool to be used in the development of a safety-related design is determined in the software tool criteria evaluation, which results in a Tool Confidence Level (TCL): a level of confidence that the software tool does not introduce or fail to detect an error in the design without being noticed, and mitigated before the design is released as a safety-related product. This evaluation is best performed in the context of the overall software tool chain and development flow, in which the individual software tool is used. The following high-level diagram reflects the tool chain for which the Fusion Compiler tool is applicable.

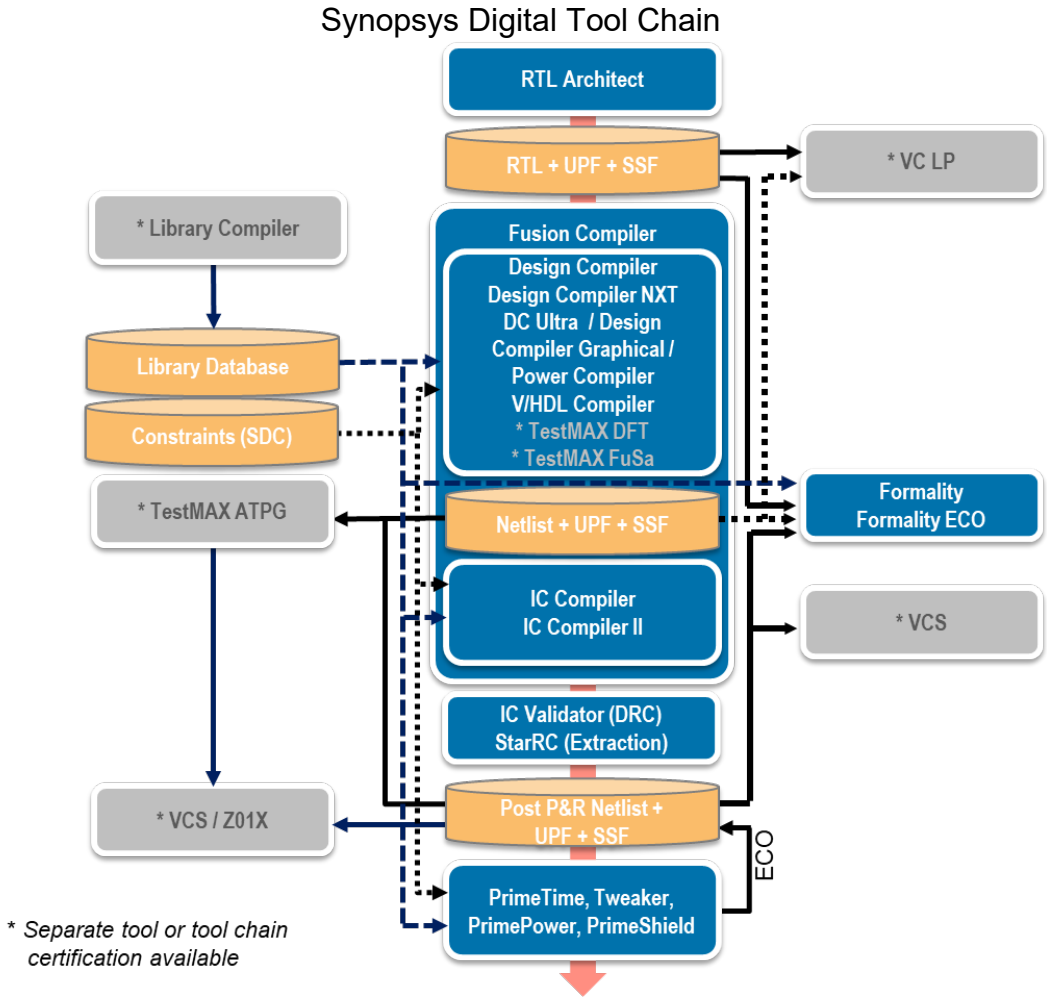

Figure 3: Synopsys Digital Tool Chain

6
Use Cases

This section describes the safety-qualified use cases of the Fusion Compiler tool. Users should also perform TCL determination based on their specific Use Cases.

The Fusion Compiler tool enables the user to perform physical implementation of very deep submicron designs. It provides the following functionality:

- Design Initialization
  To initialize the design, you must perform the following tasks:
  - Read and elaborate the HDL netlist
  - Read the design constraints
- Mapping, Placement, and Optimization
  During mapping, placement, and optimization, the tool implements the gate-level netlist and assigns a physical location for each cell in the block. The tool selects the cells and their locations according to the timing, congestion, and multi-voltage constraints.
- Functional Safety Register Synthesis
  This is an additional design and optimization step that can be run on top of technology mapping and logic optimization, covering designs with safety mechanisms in the form of safety registers such as TMR, DMR, and fault-tolerant registers.
- Functional Safety Core Synthesis
  This is an additional design and optimization step that can be run on top of wireload-based technology mapping and logic optimization or physical-based mapping and optimization, covering designs with safety mechanisms in the form of safety cores, such as DCLS.
- Failsafe Finite State Machine (FSM) Synthesis
  This is an additional design and optimization step that can be run on top of wireload-based technology mapping and logic optimization or physical-based mapping and optimization, covering designs with safety mechanisms in the form of failsafe finite state machines.
- Clock Tree Synthesis and Optimization
  During clock tree synthesis and optimization, the tool buffers the clock trees to balance the skew between the clocks in the placed design according to the timing, congestion, and multi-voltage constraints.
- Routing
  During routing, the tool connects the pins of the cells in the design according to the routing design rules and the timing and multi-voltage constraints.
- Postroute Optimization
  During post-route optimization, the tool optimizes the routed design according to the design

rules and timing, congestion, and multi-voltage constraints.

- Chip Finishing
  During chip finishing, the user inserts decoupling capacitors, filler cells, and metal fill. The user also performs electromigration analysis.

- Write Data
  During write data, the user saves the final design after completing place and route. The user must save the following data:
  - Layout data (GDSII or OASIS)
  - Floorplanning data (DEF)
  - Placed and routed gate-level netlist (Verilog)
  - Updated power intent (UPF)
  - Updated Synopsys Verification Format (SVF)
  - Design database (NDM)
  - Library data (NDM)
  - Log files
  - Report files
  - Switching activity data (SAIF)

Use Case 1: Design Initialization

In the Design Initialization use case, the goal is to prepare the design for the Fusion Compiler flow by performing the following tasks:

- Read and elaborate the HDL netlist
- Read the design constraints

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - RTL design files such Verilog (.v), SystemVerilog (.sv), or VHDL (.vhd) files
  - Flow scripts (such as the Fusion Compiler Reference Methodology)
  - Floorplanning data (DEF)
  - Design implementation constraints (Tcl)
  - Timing constraints (SDC)
  - Power intent (UPF)

- Switching activity data (SAIF)
- NDM Cell Libraries (.clib)
- Cell library source data (.db + .frame) (only in library assembly flow
- Safety Intent (.ssf or .tcl)

- Expected outputs:
  - NDM design library
  - NDM cell libraries (.clib) (only in library assembly flow)
  - Log or command files
  - Report files

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting

time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.

- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of the Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).
- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).

- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-017: User shall review all relevant QoR reports, such as timing and latency reports, to verify that all intended clocks have been propagated and have the expected latencies and that the expected timing QoR has been achieved.
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-021: User shall verify correct reset behavior on the final implementation netlist by a gate-level simulation using a logic simulator (such as VCS).
- AoU-FC-022: Where the infer_mux_override pragma has been used in the RTL, user shall check that the corresponding SEL_OP cells in the WVGTECH netlist have the infer_mux_override attribute set.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

All analyzed failure modes and prevention, detection and mitigation measures (including conditions and assumptions of use listed above) are independent of the exact Fusion Compiler tool version.

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 1: Design Initialization

In this case, no further activities for software tool qualification are required.

Use Case 2: Mapping, Placement, and Optimization

In the Mapping, Placement, and Optimization use case, the goal is to implement the gate-level netlist and find a suitable physical location for each cell in the block according to the timing, congestion, and multi-voltage constraints.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an

additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.

- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of the Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required

specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).

- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-021: User shall verify correct reset behavior on the final implementation netlist by a gate-level simulation using a logic simulator (such as VCS).
- AoU-FC-024: User shall review paths with the infer_mux_override attribute in the mapped netlist to ensure multiplexer cells are used.
- AoU-FC-025: User shall verify clock domain crossings using an independent CDC checking tool (such as SpyGlass CDC).
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-027: When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.
- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that

supports safety intent checking (such as Formality) including SVF information and applicable safety intent.

- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 2: Mapping, Placement, and Optimization

In this case, no further activities for software tool qualification are required.

Use Case 3: Functional Safety Register Synthesis

Safety mechanisms in the form of safety registers, such as triple-modular redundancy (TMR), dual-modular redundancy (DMR), and fault tolerant registers, can be inserted to mitigate the impact of single event upsets (SEU) due to transient faults such as those coming from ion or electromagnetic radiation. Functional safety register synthesis is a design technique applied on top of physical synthesis that is performed during mapping, placement, and optimization.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs:

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
  - Safety intent (SSF or TCL)
  - SPFM loss calculations on registers (TCL)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files
  - Modified or incremental safety intent (SSF')

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of the Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.

- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-023: User shall verify that the GDSII, OASIS, DEF, gate‐level netlist, UPF, SAIF, SVF, and SSF files on disk have an up‐to-date timestamp.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-028: User shall verify gate-level SPFM meets design ASIL requirement on post-synthesis netlist with a fault modeling tool (such as TestMAX FuSa or Z01X). The netlist verified should be the same netlist and format that is to be used in later implementation and verification steps.
- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 3: Functional Safety Register Synthesis

In this case, no further activities for software tool qualification are required.

Use Case 4: Functional Safety Core Synthesis

Safety mechanisms in the form of safety cores, such as Dual Core LockStep (DCLS), can be used to mitigate the impact of SEUs due to transient faults such as those coming from ion or electromagnetic

radiation. Functional safety core synthesis is a design technique applied on top of physical synthesis that is performed during mapping, placement, and optimization.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs:

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
  - Safety intent (SSF or TCL)
  - SPFM loss calculations on registers (TCL)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files
  - Modified or incremental safety intent (SSF')

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.

- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of the Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-023: User shall verify that the GDSII, OASIS, DEF, gate‐level netlist, UPF, SAIF, SVF, and SSF files on disk have an up‐to-date timestamp.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 4: Functional Safety Core Synthesis

In this case, no further activities for software tool qualification are required.

Use Case 5: Failsafe Finite State Machine (FFSM) Synthesis

Safety mechanisms in the form of failsafe finite state machines (FSM) can mitigate the impact of SEUs due to transient faults such as those coming from ion or electromagnetic radiation. Failsafe finite state machine (FFSM) synthesis is a design technique applied on top of physical synthesis that is performed during mapping, placement, and optimization.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs:

- Inputs:
  - RTL design files such Verilog (.v), SystemVerilog (.sv), or VHDL (.vhd) files
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
  - Safety intent (SSF or TCL)
  - SPFM loss calculations on registers (TCL)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files
  - Modified or incremental safety intent (SSF')

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.

- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of the Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.

- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-023: User shall verify that the GDSII, OASIS, DEF, gate‐level netlist, UPF, SAIF, SVF, and SSF files on disk have an up‐to-date timestamp.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 5: Failsafe Finite State Machine (FFSM) Synthesis

In this case, no further activities for software tool qualification are required.

Use Case 6: Clock Tree Synthesis and Optimization

In the Clock Tree Synthesis and Optimization use case, the goal is to buffer the clock trees to balance the skew between the clocks in the placed design according to the timing, congestion, and multi-voltage constraints.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.

- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-006: User shall verify that all clocks in the clock latency reports have expected latencies.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).
- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).

- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-017: User shall review all relevant QoR reports, such as timing and latency reports, to verify that all intended clocks have been propagated and have the expected latencies and that the expected timing QoR has been achieved.
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-027: When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.
- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

All analyzed failure modes and prevention, detection and mitigation measures (including conditions and assumptions of use listed above) are independent of the exact Fusion Compiler tool version.

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 6: Clock Tree Synthesis and Optimization

In this case, no further activities for software tool qualification are required.

Use Case 7: Routing

In the Routing use case, the goal is to connect the pins of the cells in the design according to the routing design rules and the timing and multi-voltage constraints.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an

additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.

- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required

specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).

- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-027: When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.
- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

All analyzed failure modes and prevention, detection and mitigation measures (including conditions and assumptions of use listed above) are independent of the exact Fusion Compiler tool version.

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 7: Routing

In this case, no further activities for software tool qualification are required.

Use Case 8: Postroute Optimization

In the Post-route Optimization use case, the goal is to optimize the routed design according to the design rules and timing, congestion, and multi-voltage constraints.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.

- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).

- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).
- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-027: When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.
- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.

- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

All analyzed failure modes and prevention, detection and mitigation measures (including conditions and assumptions of use listed above) are independent of the exact Fusion Compiler tool version.

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 8: Postroute Optimization

In this case, no further activities for software tool qualification are required.

Use Case 9: Chip Finishing

In the Chip Finishing use case, the goal is to improve reliability by inserting decoupling capacitors, inserting filler cells, inserting metal fill, and performing electromigration analysis.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
- Expected outputs:
  - NDM design library
  - Log files
  - Report files

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.

- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).

- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).
- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
- AoU-FC-027: When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.

- AoU-FC-029: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

All analyzed failure modes and prevention, detection and mitigation measures (including conditions and assumptions of use listed above) are independent of the exact Fusion Compiler tool version.

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 9: Chip Finishing

In this case, no further activities for software tool qualification are required.

Use Case 10: Write Data

In the Write Data use case, the goal is to output the placement and routing results for the design.

In this use case, the Fusion Compiler tool uses and generates the following main inputs and outputs.

- Inputs:
  - NDM design library
  - NDM cell libraries (.clib)
  - Flow scripts (.tcl)
- Expected outputs:
  - NDM design library
  - Layout data (GDSII or OASIS)
  - Floorplanning data (DEF)
  - Floorplanning data (DEF)

- Final gate-level netlist
- Updated power intent (.UPF)
- Updated Synopsys Verification Format (.SVF)
- Switching activity data (SAIF)
- Log files
- Report files
- Modified or incremental safety intent (.ssf)

- Expected outputs:

For this use case of the Fusion Compiler tool, the following conditions of use (constraints for the design and design environment, recommended procedures for the tool usage, etc.) shall be met:

- CoU-FC-001: User shall review all error and warning messages and take appropriate action.
- CoU-FC-002: User shall follow the Fusion Compiler Reference Methodology or use equivalent scripts.
- CoU-FC-003: For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. TCL scripts and log files shall be retained as design signoff records.
- CoU-FC-004: User shall not use "Dirty data handling" or "Incomplete UPF" mode.
- CoU-FC-005: User shall not use the extracted parasitics from the Fusion Compiler tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
- CoU-FC-006: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-007: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.
- CoU-FC-008: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by

Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

For this use case of Fusion Compiler tool, the following assumptions of use (required actions to be taken by the tool user to prevent or detect design errors due to possible tool malfunctions) shall be met:

- AoU-FC-001: User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
- AoU-FC-002: User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results.
- AoU-FC-003: User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
- AoU-FC-004: User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
- AoU-FC-005: User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
- AoU-FC-007: User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
- AoU-FC-008: User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-009: User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).
- AoU-FC-010: User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-011: User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-012: User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
- AoU-FC-013: User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).

- AoU-FC-014: User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
- AoU-FC-015: User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
- AoU-FC-016: User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
- AoU-FC-018: User shall review independent tool log files for error messages, warning messages, random characters, and correct file loading during the independent verification steps for DRC checking, LVS checking, formal verification (with SVF & SSF), and power intent verification.
- AoU-FC-019: User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
- AoU-FC-020: User shall review the floorplanning data in the Fusion Compiler GUI.
- AoU-FC-023: User shall verify that the GDSII, OASIS, DEF, gate‐level netlist, UPF, SAIF, SVF, and SSF files on disk have an up-to-date timestamp.
- AoU-FC-030: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

All analyzed failure modes and prevention, detection and mitigation measures (including conditions and assumptions of use listed above) are independent of the exact Fusion Compiler tool version.

A software tool criteria evaluation performed by Synopsys according to ISO 26262-8, clause 11, which assumes the fulfillment of all conditions of use (CoU) and assumptions of use (AoU) as described above, results in a required tool confidence level:

TCL1 for Fusion Compiler Use Case 10: Write Data

In this case, no further activities for software tool qualification are required.

7
Limitations of Use Cases

This section describes all known limitations of the use cases mentioned in the previous section.

All known safety-related issues for the Fusion Compiler tool are listed in the Fusion Compiler Safety-Related Issues Master List available on SolvNetPlus.

LIM-1: LCA Features

Each release of the Fusion Compiler tool may contain hidden, undocumented features for testing or evaluation purposes, known as “Limited Customer Availability” (LCA) features. Use LCA features only for testing and evaluating the proposed new features, not for production work.

Appendix A
Software Tool Information

This section provides general information about the Fusion Compiler software tool, which is needed by the tool user for performing his/her software tool criteria evaluation.

The following information about Fusion Compiler is required according to ISO 26262-8, for the planning of the usage of a software tool (clause 11.4.4) and the preparation of the own software tool criteria evaluation (clause 11.4.5).

Please note that some of the information below provided by Synopsys simply needs to be confirmed by the tool user and can be used without modification. Other information must be completed or updated by the tool user to reflect his/her actual situation.

Required Info	Tool Information	Reference / Comment
Tool vendor	Synopsys, Inc.	ISO 26262-8, 11.4.4.1.a
Tool name and version	Fusion Compiler version P-2019.03 and later versions	ISO 26262-8, 11.4.4.1.a To determine tool version, use: `report_version -options`
Tool use cases		ISO 26262-8, 11.4.4.1.c ISO 26262-8, 11.4.5.1.a To be completed by the tool user. Align with / verify against use cases described in Section 6 of this document.
Tool inputs and expected outputs		ISO 26262-8, 11.4.5.1.b To be completed by the tool user. Align with / verify against inputs and outputs described in Section 6 of this document.
Tool configuration and constraints		ISO 26262-8, 11.4.4.1.b ISO 26262-8, 11.4.5.1.c To be completed by the tool user. Align with / verify against CoU for the use cases described in Section 6 of this document.
Tool environment (OS)	Refer to the Fusion Compiler Installation Notes at https://solvnet.synopsys.com/DownloadCenter. Click the Fusion Compiler tool name, release	ISO 26262-8, 11.4.4.1.d To be completed by the tool user. Align with / verify against the OS version evaluated by Synopsys.

	number, and then "View installation guide" for tool version-specific OS support.	To determine Linux version, use: `uname -osr`
Tool environment (CAD tool chain)		ISO 26262-8, 11.4.4.1.d To be completed by the tool user. To determine name and version of your tool chain, please consult your CAD department.
Maximum ASIL	ASIL D	ISO 26262-8, 11.4.4.1.e
Tool qualification methods	Not applicable	ISO 26262-8, 11.4.4.1.f Software tool qualification is not required for Fusion Compiler
User manual and other usage guide documents	Fusion Compiler Online Help Fusion Compiler Reference Methodologies	ISO 26262-8, 11.4.4.2.a – d Tool user to include a link to these documents (Synopsys SolvNetPlus or local copy), and to add any additional company-internal tool usage guidelines.
Known software tool malfunctions, and appropriate work arounds ...	For limitations, refer to Section 7 of this document. Fusion Compiler Safety-Related Issues Master List	ISO 26262-8, 11.4.4.2.e Tool user to include a link to these documents (Synopsys SolvNetPlus or local copy), and to add any additional company-internal work around descriptions.
Measures for the detection of tool malfunctions ...		ISO 26262-8, 11.4.4.2.f To be completed by the tool user. Align with / verify against AoU for the use cases described in Section 6 of this document.

Appendix B
Complete List of CoU and AoU IDs

The complete list of Conditions of Use (CoU) for Fusion Compiler is shown in the table below. CoU define a condition of the design, software tool, design environment, or situation that is assumed and required to be fulfilled by the user.

The complete list of Conditions of Use (CoU) for Fusion Compiler is in the table below. CoU defines a condition of the design, software tool, design environment, or situation that is assumed and required to be fulfilled by the user.

ID	Description
CoU-FC-001	User shall review all error and warning messages and take appropriate action.
CoU-FC-002	User shall follow the *Fusion Compiler* Reference Methodology or use equivalent scripts.
CoU-FC-003	For the final run, Tcl script-based batch mode execution shall be used, without interactive command-line entry or GUI manual command entry. Tcl scripts and log files shall be retained as design signoff records.
CoU-FC-004	User shall not use "Dirty data handling" or "Incomplete UPF" mode.
CoU-FC-005	User shall not use the extracted parasitics from the *Fusion Compiler* tool for signoff verification steps, extracted parasitics shall come from a signoff extraction tool (such as StarRC).
CoU-FC-006	When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.
CoU-FC-007	When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.

ID	Description
CoU-FC-008	When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.

The complete list of Assumptions of Use (AoU) for Fusion Compiler is shown in the table below. AoU define an action that is assumed and required to be taken by the user of a software tool.

ID	Description
AoU-FC-001	User shall review the log files for error messages, warning messages, correct, complete, or unexpected flow execution, and random characters.
AoU-FC-002	User shall review all relevant QoR reports, such as timing and power reports, for error messages, warning messages, random characters, and expected results..
AoU-FC-003	User shall review all relevant flow status and checking reports, such as check_design, report_references, report_dft, and dft_drc, for error messages, warning messages, random characters, and expected results.
AoU-FC-004	User shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool (such as Formality).
AoU-FC-005	User shall verify that all output reports, log files and NDM database disk files have an up-to-date timestamp.
AoU-FC-006	User shall verify that all clocks in the clock latency reports have expected latencies.
AoU-FC-007	User shall perform independent power intent verification of the RTL to final gate netlist and UPF file using a low-power static verification tool (such as VC LP).
AoU-FC-008	User shall perform independent DRC verification of the GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
AoU-FC-009	User shall perform independent validation of the DFT structures in the gate-level netlist after final implementation to ensure they are correct and meet the required specifications. This testing should be performed through test DRC validation using an ATPG or other test verification tool (such as TetraMAX or TestMAX ATPG).

AoU-FC-010	User shall perform independent dynamic power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
AoU-FC-011	User shall perform independent cell electromigration verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
AoU-FC-012	User shall perform independent power and signal electromigration verification of the gate-level netlist and DEF file after final implementation using a power integrity and reliability analysis tool.
AoU-FC-013	User shall perform independent static timing verification of the final gate-level netlist and DEF file after place and route using a static timing analysis tool (such as PrimeTime) and parasitic extraction tool (such as StarRC).
AoU-FC-014	User shall perform independent LVS connectivity verification of the gate-level netlist and GDSII or OASIS file after final implementation using a physical verification tool (such as IC Validator).
AoU-FC-015	User shall perform independent static and dynamic voltage drop analysis using a power integrity and reliability analysis tool.
AoU-FC-016	User shall perform independent static power verification of the gate-level netlist and DEF file after final implementation using a signoff power analysis tool (such as PrimePower) and a parasitic extraction tool (such as StarRC).
AoU-FC-017	User shall review all relevant QoR reports, such as timing and latency reports, to verify that all intended clocks have been propagated and have the expected latencies and that the expected timing QoR has been achieved.
AoU-FC-018	User shall review independent tool log files for error messages, warning messages, random characters, and correct file loading during the independent verification steps for DRC checking, LVS checking, formal verification (with SVF & SSF), and power intent verification.
AoU-FC-019	User shall review the log files for error messages, warning messages, and correct operation during the saving and subsequent reloading of the NDM libraries.
AoU-FC-020	User shall review the floorplanning data in the Fusion Compiler GUI.
AoU-FC-021	User shall verify correct reset behavior on the final implementation netlist by a gate-level simulation using a logic simulator (such as VCS).
AoU-FC-022	Where the infer_mux_override pragma has been used in the RTL, user shall check that the corresponding SEL_OP cells in the WVGTECH netlist have the infer_mux_override attribute set.
AoU-FC-023	User shall verify that the GDSII, OASIS, DEF, gate-level netlist, UPF, SAIF, SVF, and SSF files on disk have an up-to-date timestamp.

AoU-FC-024	User shall review paths with the infer_mux_override attribute in the mapped netlist to ensure multiplexer cells are used.
AoU-FC-025	User shall verify clock domain crossings using an independent CDC checking tool (such as SpyGlass CDC).
AoU-FC-026	AoU-FC-026: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.
AoU-FC-027	When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.
AoU-FC-028	User shall verify gate-level SPFM meets design ASIL requirement on post-synthesis netlist with a fault modeling tool (such as TestMAX FuSa or Z01X). The netlist verified should be the same netlist and format that is to be used in later implementation and verification steps.
AoU-FC-029	When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.
AoU-FC-030	When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).

Appendix C
List of CoU and AoU Changes

Conditions of Use (CoU) for Fusion Compiler which has changed along with a description of that change when compared to a previous document version indicated are shown in the table below.

ID	Description of Change	Version
CoU-FC-006	Newly added CoU: When StarRC is used as the signoff extraction tool and StarRC extraction mode is used during post-route optimization in Fusion Compiler, then the user shall perform an additional signoff comparison of the final reports in Fusion Compiler using the native extraction engine against the final signoff reported by StarRC. The native implementation extraction engine shall be turned on by setting extract.starrc_mode to "none". Any significant discrepancy shall be reviewed and acted upon appropriately.	v2.0
CoU-FC-007	Newly added CoU: When PrimeTime is used as the STA signoff tool and the PrimeTime delay kernel is used during post-route optimization in Fusion Compiler, then the user shall perform an additional timing comparison of the final timing reported in Fusion Compiler using the native implementation timer against the final signoff timing reported by PrimeTime. The native implementation timer shall be turned on by setting time.enable_low_accuracy_delay_calculation to "true". Any significant timing discrepancy shall be reviewed and acted upon appropriately.	v2.0
CoU-FC-008	Newly added CoU: When RedHawk AF is used during optimization in Fusion Compiler, then the user shall perform an independent comparison of the final power rail analysis reported by Fusion Compiler with an independent power integrity and reliability signoff tool. Any significant power integrity and reliability discrepancy shall be reviewed and acted upon appropriately.	v2.0

Assumptions of Use (AoU) for Fusion Compiler which has changed along with a description of that change when compared to a previous document version indicated are shown in the table below.

ID	Description of Change	Version
AoU-FC-022	Changed AoU to provide examples for QoR reports such as timing and power reports	v2.0
AoU-FC-023	Added SSF files to the AoU	v2.0
AoU-FC-026	Newly added AoU: When SSF has been utilized, user shall review the relevant safety reports (report_safety_status, report_safety_register_groups, report_fsm) and validate expected safety measures are implemented.	v2.0
AoU-FC-027	Newly added AoU: When SSF has been utilized, user shall perform independent checks to confirm the validity of the functional safety report.	v2.0
AoU-FC-028	Newly added AoU: User shall verify gate-level SPFM meets design ASIL requirement on post-synthesis netlist with a fault modeling tool (such as TestMAX FuSa or Z01X). The netlist verified should be the same netlist and format that is to be used in later implementation and verification steps.	v2.0
AoU-FC-029	Newly added AoU: When SSF has been utilized, user shall perform independent formal equivalence checking of RTL to final gate netlist using a formal verification tool that supports safety intent checking (such as Formality) including SVF information and applicable safety intent.	v2.0
AoU-FC-030	Newly added AoU: When SSF has been utilized, user shall validate correctness of any SSF output from the tool. (Ex: generate safety reports; write SSF' and read it back into the tool; generate safety reports and compare).	v2.0