DRAFT INTERNATIONAL STANDARD
ISO/DIS 26262-11

ISO/TC **22**/SC **32** Secretariat: **JISC**

Voting begins on:
2016-09-21

Voting terminates on:
2016-12-13

Road vehicles — Functional safety —

Part 11:
Guideline on application of ISO 26262 to semiconductors

Véhicules routiers — Sécurité fonctionnelle —

Partie 11: titre manque

ICS: 43.040.10

Reference number
ISO/DIS 26262-11:2016(E)

Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2. www.iso.org/directives

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received. www.iso.org/patents

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/TC22/SC32 Electrical and electronic components and general system aspects.

A list of all parts in the ISO 26262 series can be found on the ISO website.

Introduction

ISO 26262 is the adaptation of IEC 61508 to address the sector specific needs of electrical and/or electronic (E/E) systems within road vehicles.

This adaptation applies to all activities during the safety lifecycle of safety-related systems comprised of electrical, electronic and software components.

Safety is one of the key issues in the development of road vehicles. Development and integration of automotive functionalities strengthen the need for functional safety and the need to provide evidence that functional safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, among others there are increasing risks from systematic failures and random hardware failures, these being considered within the scope of functional safety. ISO 26262 includes guidance to mitigate these risks by providing appropriate requirements and processes.

To achieve functional safety, ISO 26262:

a) provides a reference for the automotive safety lifecycle and supports the tailoring of the activities to be performed during the lifecycle phases, i.e., development, production, operation, service, and decommissioning;
b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
c) uses ASILs to specify which of the requirements of ISO 26262 are applicable to avoid unreasonable residual risk;
d) provides requirements for functional safety management, verification, validation and confirmation measures; and
e) provides requirements for relations with suppliers.

ISO 26262 is concerned with functional safety of E/E systems that is achieved through safety measures including safety mechanisms. It also provides a framework within which safety-related systems based on other technologies (e.g. mechanical, hydraulic and pneumatic) can be considered.

The achievement of functional safety is influenced by the development process (including such activities as requirements specification, design, implementation, integration, verification, validation and configuration), the production and service processes and the management processes.

Safety is intertwined with common function-oriented and quality-oriented activities and work products. ISO 26262 addresses the safety-related aspects of these activities and work products.

Figure 1 shows the overall structure of ISO 26262. ISO 26262 is based upon a V-model as a reference process model for the different phases of product development. Within the figure:

— the shaded “V”s represent the interconnection among ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7; for motorcycles ISO 26262-12 clauses 6 supplements ISO 26262-3 and clause 7, 8 and 9 supplements Part 4.

— the specific clauses are indicated in the following manner: "m-n", where "m" represents the number of the particular part and "n" indicates the number of the clause within that part.

EXAMPLE "2-6" represents Clause 6 of ISO 26262-2.

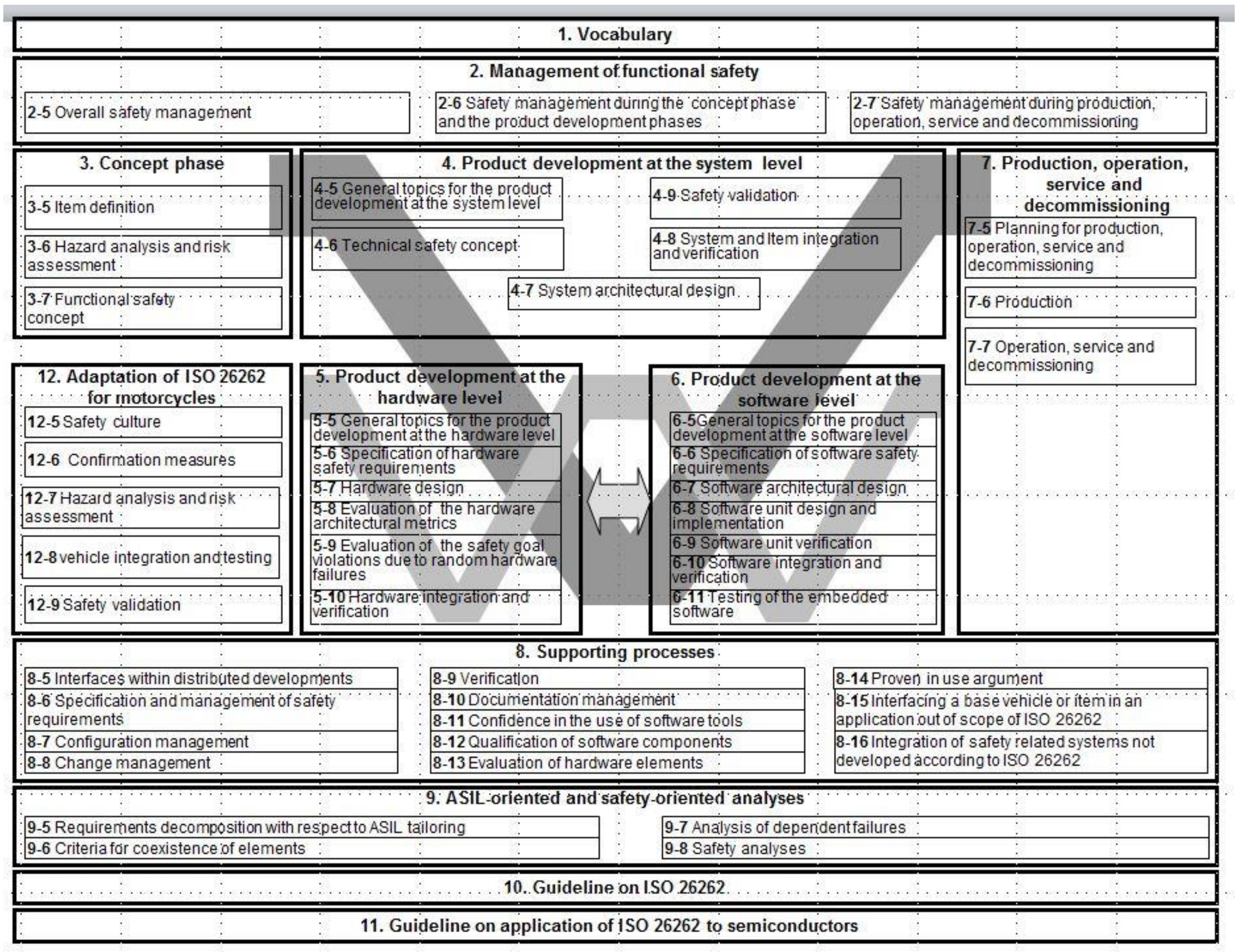

1. Vocabulary
2. Management of functional safety
2-5 Overall safety management
2-6 Safety management during the concept phase and the product development phases
2-7 Safety management during production, operation, service and decommissioning
3. Concept phase
3-5 Item definition
3-6 Hazard analysis and risk assessment
3-7 Functional safety concept
4. Product development at the system level
4-5 General topics for the product development at the system level
4-6 Technical safety concept
4-7 System architectural design
4-8 System and Item integration and verification
4-9 Safety validation
7. Production, operation, service and decommissioning
7-5 Planning for production, operation, service and decommissioning
7-6 Production
7-7 Operation, service and decommissioning
12. Adaptation of ISO 26262 for motorcycles
12-5 Safety culture
12-6 Confirmation measures
12-7 Hazard analysis and risk assessment
12-8 vehicle integration and testing
12-9 Safety validation
5. Product development at the hardware level
5-5 General topics for the product development at the hardware level
5-6 Specification of hardware safety requirements
5-7 Hardware design
5-8 Evaluation of the hardware architectural metrics
5-9 Evaluation of the safety goal violations due to random hardware failures
5-10 Hardware integration and verification
6. Product development at the software level
6-5 General topics for the product development at the software level
6-6 Specification of software safety requirements
6-7 Software architectural design
6-8 Software unit design and implementation
6-9 Software unit verification
6-10 Software integration and verification
6-11 Testing of the embedded software
8. Supporting processes
8-5 Interfaces within distributed developments
8-6 Specification and management of safety requirements
8-7 Configuration management
8-8 Change management
8-9 Verification
8-10 Documentation management
8-11 Confidence in the use of software tools
8-12 Qualification of software components
8-13 Evaluation of hardware elements
8-14 Proven in use argument
8-15 Interfacing a base vehicle or item in an application out of scope of ISO 26262
8-16 Integration of safety related systems not developed according to ISO 26262
9. ASIL-oriented and safety-oriented analyses
9-5 Requirements decomposition with respect to ASIL tailoring
9-6 Criteria for coexistence of elements
9-7 Analysis of dependent failures
9-8 Safety analyses
10. Guideline on ISO 26262
11. Guideline on application of ISO 26262 to semiconductors

Figure 1 — Overview of ISO 26262

Road vehicles—Functional Safety—Part 11: Guideline on application of ISO 26262 to semiconductors

1. Scope

ISO 26262 is intended to be applied to safety-related systems that include one or more electrical and/or electronic (E/E) systems and that are installed in series production road vehicles, excluding mopeds. ISO 26262 does not address unique E/E systems in special vehicles such as E/E systems designed for drivers with disabilities.

NOTE Other dedicated application-specific safety standards exist and may complement ISO 26262 or vice versa.

Systems and their components released for production, or systems and their components already under development prior to the publication date of this edition of ISO 26262, are exempted from the scope of this edition. For further development or alterations based on systems and their compon ents released for production prior to the publication of this edition of ISO 26262, only the modifications will be developed in accordance with this edition of ISO 26262. This edition of ISO 26262 addresses integration of existing systems not developed according to this edition of ISO 26262 and systems developed according to this edition of ISO 26262 by tailoring the safety lifecycle.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of safety-related E/E systems, including interaction of these systems. It does not address hazards related to electric shock, fire, smoke, heat, radiation, toxicity, flammability, reactivity, corrosion, release of energy and similar hazards, unless directly caused by malfunctioning behaviour of safety-related E/E systems.

ISO 26262 does not address the nominal performance of E/E systems, even if functional performance standards exist for these systems (e.g. active and passive safety systems, brake systems, adaptive cruise control).

ISO 26262 describes a framework for functional safety to assist the development of safety-related E/E systems. This framework is intended to be used to integrate functional safety activities into a company-specific development framework. Some requirements have a clear technical focus to implement functional safety into a product; others address the development process and can therefore be seen as process requirements in order to demonstrate the capability of an organization with respect to functional safety.

This Part of ISO 26262 has an informative character only. It contains possible interpretations of other parts of ISO 26262 with respect to semiconductor development. The content is not exhaustive with regard to possible interpretations, i.e., other interpretations may also be pos sible in order to fulfil the requirements defined in other parts of ISO 26262. Therefore the interpretations from this document are not considered as mandatory for audits and assessments.

2. Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 26262-1:2018, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2018, *Road Vehicles — Functional Safety— Part 2: Management of functional safety*

ISO 26262-3:2018, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-4:2018, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2018, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2018, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-7:2018, *Road vehicles — Functional safety — Part 7: Production, operation, service and decommissioning*

ISO 26262-8:2018, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2018, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3. Terms and definitions

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2018 apply.

4. A semiconductor component and its partitioning

4.1 How to consider a semiconductor component

4.1.1 Semiconductor component development

If a semiconductor component is developed as a part of an item development compliant with ISO 26262, it is developed based on hardware safety requirements derived from the top-level safety goals of the item, through the technical safety concept. Targets for hardware architectural metrics and Probabilistic Metric for random Hardware Failures (PMHF) are allocated to the item: in this case, the semiconductor component is just one of the elements. As mentioned in the EXAMPLE of ISO 26262-5:2018, 8.2, to facilitate distributed developments, target values can be assigned to the semiconductor component itself. The safety analysis of a semiconductor component is performed based on the requirements and recommendations defined in ISO 26262-5:2018, 7.4.3 and in ISO 26262-9:2018, Clause 8.

NOTE If an element has not been developed in compliance with ISO 26262, the requirements in ISO 26262-8:2018, Clause 13 can be considered.

The semiconductor component can be developed as a Safety Element Out Of Context (SEooC), as described in ISO 26262-10:2018, Clause 9. In this case, the development is done based on assumptions on the conditions of the semiconductor component usage (Assumptions of Use or AoU, see 4.4), and then the assumptions are verified at the next higher level of integration considering the semiconductor component requirements derived from the safety goals of the item in which the semiconductor component is to be used.

The descriptions and methods in this part are provided assuming the semiconductor component is an SEooC, but the described methods (e.g. the method for failure rates computation of a semiconductor component) are still valid if the semiconductor component is not considered as an SEooC. When those methods are conducted considering the stand-alone semiconductor component, appropriate assumptions are made. Subclause 4.4 describes how to adapt and verify those methods and assumptions at system or element level. At the stand-alone semiconductor component level, the requirements of ISO 26262-2:2018, ISO 26262-5:2018, ISO 26262-6:2018, ISO 26262-7:2018, ISO 26262-8:2018 and ISO 26262-9:2018 (e.g. related to safety analyses, dependent failures analysis, verification, etc.) can be applied.

4.2 Dividing a semiconductor component in parts

As shown in Figure 2 and according to the definitions in ISO 26262-1:2018, a semiconductor component can be divided into parts: the whole semiconductor hierarchy can be seen as a component, the second level of hierarchy (e.g. a CPU) as a part, the following levels of hierarchy (e.g. the CPU register bank) as sub-parts, till the elementary sub-parts (its internal registers and the related logic cone).

NOTE 1 The necessary level of detail (e.g. whether to stop at part level or go down to sub-part or elementary sub-part level) can depend on the stage of the analysis and on the safety mechanisms used (inside the semiconductor component or at the system or element level). In general, three levels of hierarchy (component, part and sub-part) are considered.

NOTE 2 The level of detail at which the elementary sub-part is defined is relative and depends on the safety concept.

EXAMPLE Flip-flop, analogue transistor, CPU, etc.

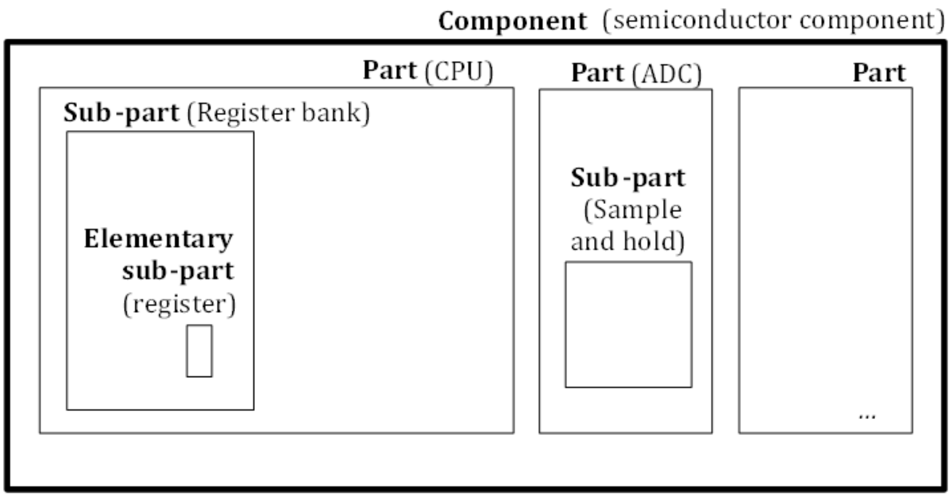

Component (semiconductor component)
Part (CPU)
Part (ADC)
Part
Sub-part (Register bank)
Elementary sub-part (register)
Sub-part (Sample and hold)
...

Figure 2 — A semiconductor, its parts and sub-parts

4.3 About hardware faults, errors and failure modes

Hardware faults, errors and failure modes of an integrated circuit are linked together as shown in Figure 3 below.

NOTE 1 The failure mode can be abstract or tailored to a specific implementation, e.g. related to a pin of a component, part or sub-part.

In general, failure modes are described in ISO 26262 as functional failure modes. Further characterisation of failure modes can be possible.

EXAMPLE An example of failure modes for digital circuits is given in Annex A.

Faults and errors described in this part of ISO 26262 are related to the physical implementation of a given semiconductor component.

NOTE 2 The use of the terms fault, error, and failure is done carefully. In ISO 26262, faults create errors which can lead to a failure. In many reliability modelling standards the terms fault and failure are used interchangeably.

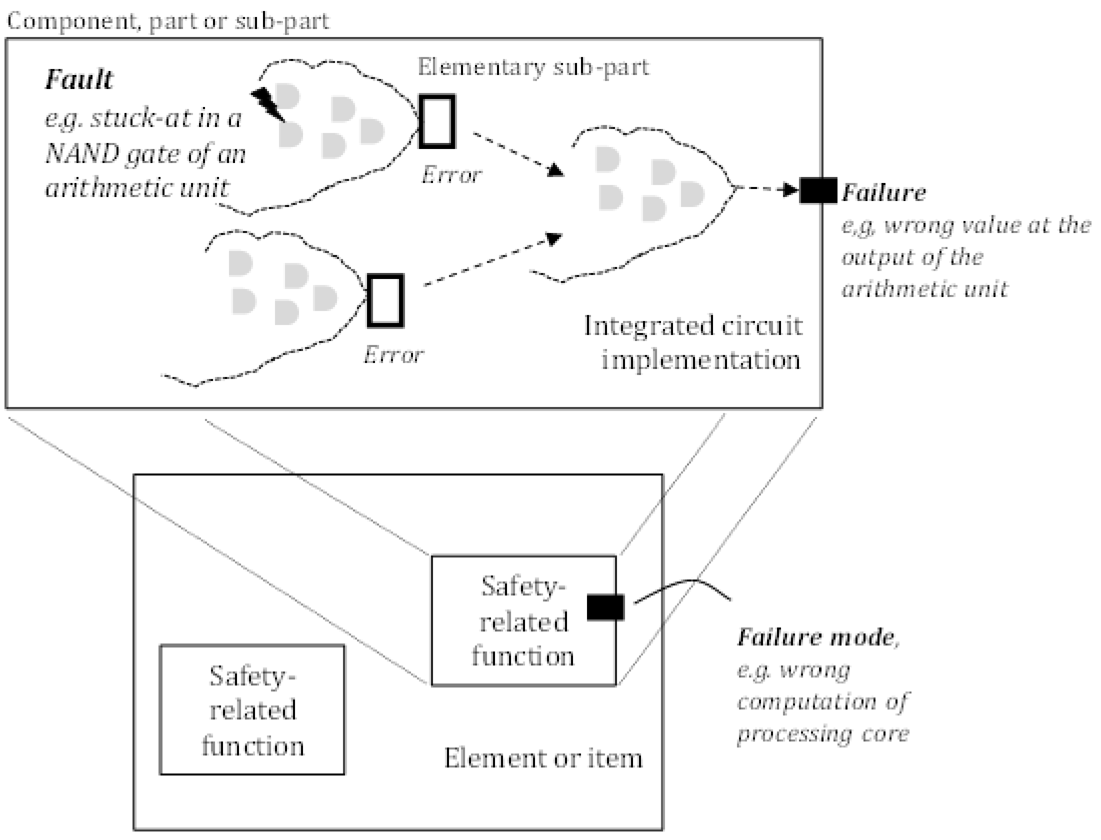

Component, part or sub-part
Fault
e.g. stuck-at in a NAND gate of an arithmetic unit
Elementary sub-part
Error
Error
Failure
e.g. wrong value at the output of the arithmetic unit
Integrated circuit implementation
Safety-related function
Safety-related function
Failure mode,
e.g. wrong computation of processing core
Element or item

Figure 3 — Relationship between hardware faults and failure modes

4.3.1 Fault models

Fault models are a necessarily abstract representation of physical faults.

ISO 26262-5:2018 Table D.1 NOTE 2 describes that diagnostic coverage depends on the failure mode distribution and the related failure mode coverage. Following the relationship between hardware faults and failure modes as represented in Figure 3, the failure mode distribution is correlated with the fault models.

EXAMPLE If a failure mode is caused at X% by stuck-at faults and Y% by shorts, and if a safety mechanism only covers stuck-at faults with a coverage of Z%, then the claimed diagnostic coverage is X% * Z%.

In the context of a semiconductor component, relevant fault models are identified based on the technology and circuit implementation.

NOTE 1 Further detail on fault models can be provided for specific technologies, see 5.1 for digital components.

NOTE 2 Typically it is not possible to evaluate every possible physical fault by itself due to the number and required level of detail.

4.3.2 Failure modes

To define failure modes, keywords are used if applicable.

NOTE 1 Examples of keywords are: wrong program flow execution, data corruption, accessing unintended locations, deadlock, livelock, incorrect Instruction execution.

NOTE 2 An example of failure modes for digital circuits is given in Annex A.

The failure mode is described at a level of detail commensurate to the safety concept and the related safety mechanism.

EXAMPLE 1 In the case of a CPU with a hardware lock-step safety mechanism, the failure mode can be defined by looking at the CPU function as a whole.

EXAMPLE 2 In the case of a CPU with a structural software-based hardware test, the failure mode is defined in more detail because the software test will cover different failure modes with different failure mode coverage.

The association between the identified failure modes and circuit implementation fault models is supported by evidence ensuring any failure mode is allocated to a part/sub-part of the component and any relevant part/sub-part has at least one failure mode.

NOTE 3 The goal is to guarantee that there are no gaps between circuit implementation and the listed failure modes.

4.3.3 The distribution of base failure rate across failure modes

Base failure rate is distributed across failure modes. The accuracy of that distribution is aligned with the level of detail of the analysis and the consideration of the relevant safety mechanisms available.

EXAMPLE 1 In the case of a CPU with a hardware lock-step safety mechanism, it is not necessary to have a detailed distribution of CPU failure modes.

EXAMPLE 2 In the case of a CPU with a structural software-based hardware test, the distribution is defined in more detail because in this way it will be possible to estimate with enough accuracy the diagnostic coverage of failure modes.

In case there is no data available to compute the distribution with the required accuracy, the failure rate is evenly distributed across the failure modes or an expert judgment is provided with related arguments.

NOTE A sensitivity analysis to the distribution can be done to evaluate the impact on the diagnostic coverage.

4.4 About adapting a semiconductor component stand-alone analysis at system level

The adaptation and verification of the semiconductor component stand-alone analysis at system or element level could be done by:

— transforming the detailed failure modes of a semiconductor component into the high-level failure modes needed during the analysis at system or element level, as shown in Figure 4;

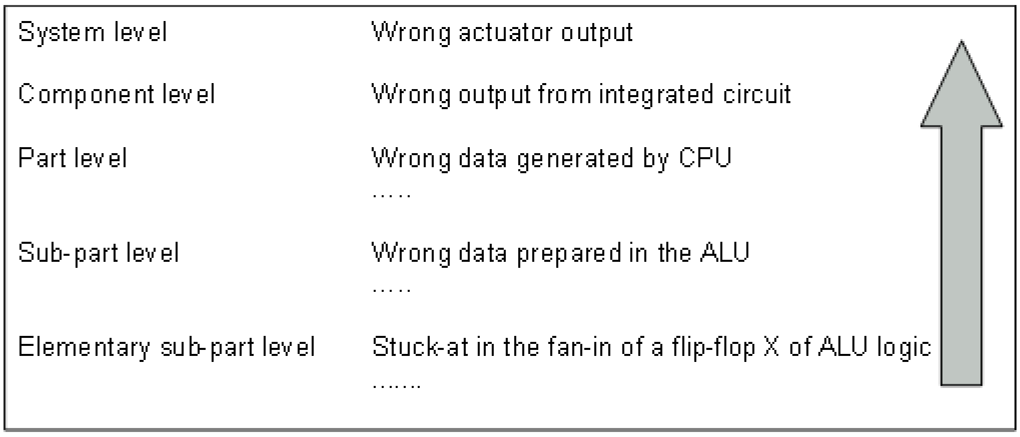

System level
Wrong actuator output
Component level
Wrong output from integrated circuit
Part level
Wrong data generated by CPU
…..
Sub-part level
Wrong data prepared in the ALU
…..
Elementary sub-part level
Stuck-at in the fan-in of a flip-flop X of ALU logic
…….

Figure 4 — Example of bottom-up approach to derive system or element-level failure modes

NOTE 1 Combining top-down (e.g. FTA) and bottom-up methods (e.g. FMEA), it can be possible to identify the detailed semiconductor component failure modes and combine them up to the component level.

NOTE 2 Starting from a detailed level of abstraction allows for a quantitative and precise failure distribution for a semiconductor component that otherwise is based on qualitative distribution assumptions.

NOTE 3 As discussed in 4.2, the necessary level of detail can depend on the stage of the analysis and on the safety mechanisms used.

— the diagnostic coverage computed at part or sub-part level could be improved by measures at the part, component level or system or item level; and

EXAMPLE 1A semiconductor component includes an ADC peripheral with no safety mechanisms implemented in hardware. At the component stand-alone level, the diagnostic coverage was considered zero. At system level, the ADC is included in a closed-loop, and its faults are detected by a software-based consistency check. In this context, the diagnostic coverage of that sub-part is increased due to the safety mechanism implemented at system-level.

— the diagnostic coverage computed at part or sub-part level could have been calculated under certain specific assumptions (“Assumptions of Use” or AoU).

NOTE 4 At system level different safety mechanisms or failure masking can be present. This can be taken into consideration in safety analysis, when a justification is possible.

EXAMPLE 2A semiconductor component includes a memory in which each single-error is corrected and signalled by the EDC to the CPU. At the component stand-alone level, it was assumed that a software driver is implemented to handle this event. At system level, for performance reasons, this software driver is not implemented, and therefore the assumption is not fulfilled. The semiconductor component is programmed to send the error correction flag directly to the outside world.

4.5 Intellectual property

4.5.1 About intellectual property

4.5.1.1 Understanding intellectual property

Intellectual property (IP) refers to a reusable unit of logical design or physical design intended to be integrated into a design as a part or a component. The term “IP integrator” is used in this clause for the organization responsible for integrating intellectual property designs from one or more sources into a design with safety requirements. The term "IP supplier" is used in reference to the organization responsible for designing or developing the IP. The IP integrator and the IP supplier can be separate parties as well as the same company or different organisations in the same company.

In a product development project involving intellectual property, the allocation of responsibilities between the IP supplier and the IP integrator can vary depending on the project. The appointment of responsibilities for each tailored activity requires effective safety management in terms of safety planning, and agreement on the development interfaces. For these activities the requirements of ISO 26262-2:2018, Clause 6 and ISO 26262-8:2018, Clause 5 are applicable.

Based on the requirements in ISO 26262 four possible approaches are identified for IP based designs. These approaches are shown in Figure 5. The IP integrator typically chooses the approach based on consideration of the information provided from the IP supplier as well as the maturity of the IP.

EXAMPLE If no supporting information is available from the IP supplier, the possible approaches can be limited to the "proven in use" argument, if applicable. Otherwise the role of the IP in the safety architecture has to be differently considered, e.g. using diverse redundancy to reduce risk of systematic and hardware random failures.

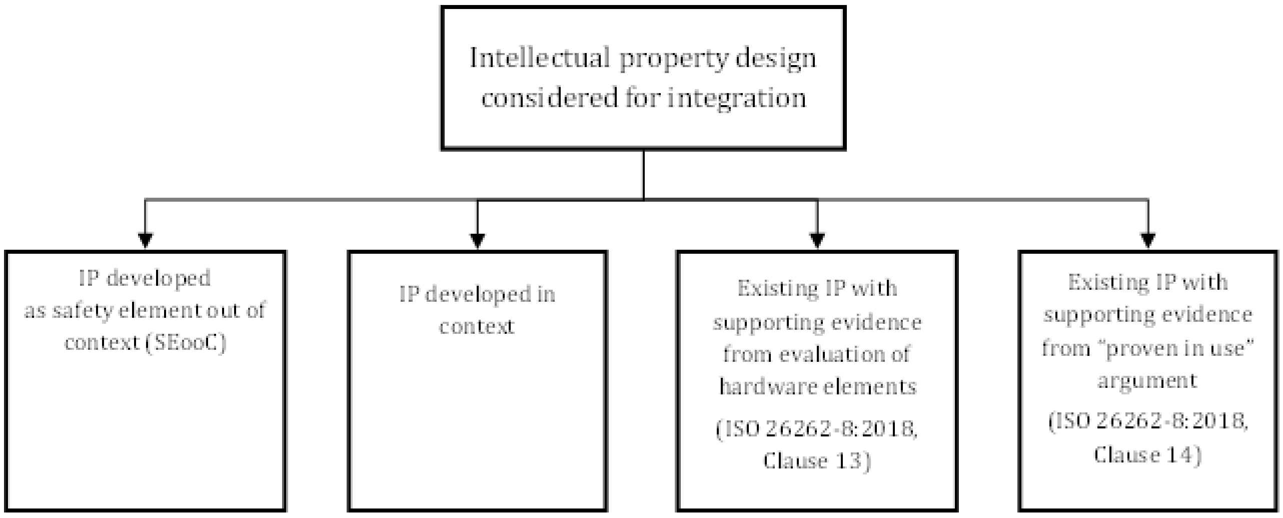

Intellectual property design considered for integration
IP developed as safety element out of context (SEooC)
IP developed in context
Existing IP with supporting evidence from evaluation of hardware elements (ISO 26262-8:2018, Clause 13)
Existing IP with supporting evidence from "proven in use" argument (ISO 26262-8:2018, Clause 14)

Figure 5 — The possible approaches for using IP in safety-related designs

The intellectual property can be an existing design with a pre-defined set of features. In this case the IP integrator has the responsibility of identifying the set of features which are required to support the safety concept of the design. Intellectual property can also be designed based on an agreed set of safety requirements. In this case the IP integrator identifies the requirements for the IP which are necessary to support the safety concept of the design. These IP use models are further described in the remainder of this clause.

The guidance in this clause can be applied to newly developed IP, modified IP, and existing unmodified IP.

NOTE A common approach is to assume the possible target usage as defined in ISO 26262-2:2018, 6.4.5.8. This option is described as safety element out of context (SEooC) in ISO 26262-10:2018, Clause 9. Development of an SEooC relies on identification of assumed uses and safety requirements which must be verified by the IP user.

4.5.1.2 Types of intellectual property

Commonly used intellectual property types are listed in Table 1. This is not an exhaustive list covering the possible intellectual property types. This document considers both types of intellectual property as applied to semiconductor designs.

Intellectual property in the form of logic design can also be configurable. The configuration options are typically specified by the IP integrator at the time of logic synthesis.

EXAMPLE 1 Configuration options to define interface bus width, memory size, and presence of fault detection mechanisms.

Intellectual property can also be generated with dedicated tools.

EXAMPLE 2 Memory compilers, C to HDL compilers, Network-on-Chip generators.

In this case:

— confidence in software tools can be demonstrated using the methods described in ISO 26262-8:2018, Clause 11;

  NOTE The confidence in software tools can be tailored based on the amount of verification performed on the generated IP;

— the IP integrator performs the necessary verification activities to guarantee the correctness of the generated IP;

— the necessary work products, as listed in following clauses, are made available; and

— the IP integrator verifies the correct integration of the IP in its context.

Table 1 — Types of intellectual property

Intellectual property type	Description
Physical representation	A complete chip layout description, containing instantiations of standard cells for a specific cell library for a target manufacturing process. EXAMPLE A/D converter macro, PLL macro.
Model representation	A description of a design in terms of a hardware description language (HDL) such as Verilog or VHDL, or analogue transistor level circuit schematic. A logic design in model representation needs to be synthesized into a list of gates consisting of basic cells, followed by placement and routing to achieve a semiconductor design. Analogue circuit schematic components, such as transistors, diodes, resistors, and capacitors, need to be mapped into target technology library components, followed by placement and routing to achieve a semiconductor design. EXAMPLE Processor or memory controller design exchanged without mapping to a particular technology, operational amplifier transistor level schematic.
NOTE 1 Physical representation IPs are also known as "hard IPs". NOTE 2 Model representation IPs are also known as "soft IPs". NOTE 3 This classification is applicable to generic IP design including digital, analogue, mixed signal, PLD, Sensors and Transducers.	

4.5.2 Category and safety requirements for intellectual property

In general, two categories of intellectual property can be determined based on the allocation of safety requirements: IP with no allocated safety requirements, and IP with one or more allocated safety requirements. When the intellectual property has no allocated safety requirements, no additional considerations are required for ISO 26262 unless identified during the safety analysis. In the case of coexistence of non-safety-related IPs with safety-related elements, dependent failure(s) analysis is used to evaluate freedom from interference. For dependent failures analysis guidance, see ISO 26262-9:2018, Clause 7 together with the additional guidance in 4.7 of this document.

If one or more safety requirements is allocated to the intellectual property, ISO 26262 requirements are applicable. In particular requirements of ISO 26262-2:2018, ISO 26262-4:2018, ISO 26262-5:2018, ISO 26262-8:2018, and ISO 26262-9:2018 are often tailored to apply to IP designs. The following text gives guidance on IP with allocated safety requirements, and how to consider these requirements for IP with and without integrated safety mechanisms.

Safety-related IPs can be further classified based on the integration of safety mechanisms. Two possible cases are illustrated in Figure 6, with subfigure (a) illustrating IP which has integrated safety mechanisms, and subfigure (b) illustrating IP which has no integrated safety mechanisms.

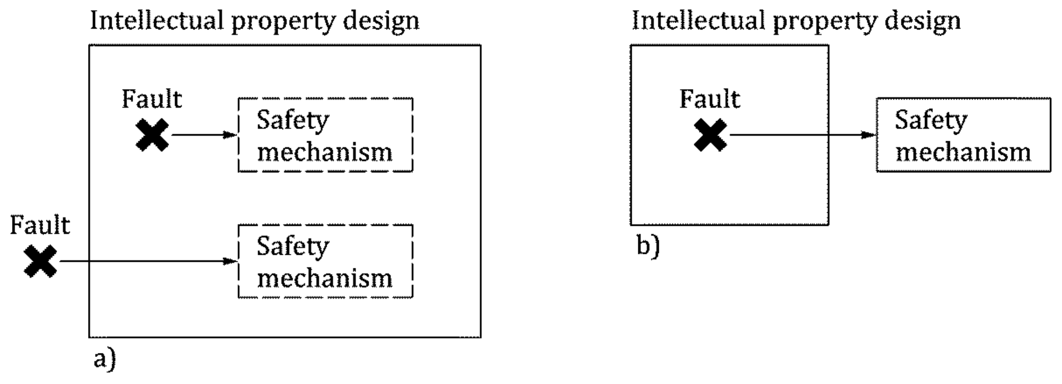


Figure 6 — Types of IP with allocated safety requirements

NOTE 1 IP safety mechanisms can be included for detection of faults internal to the IP, as well as faults external to the IP.

NOTE 2 Safety mechanisms implemented in the IP can provide full or partial diagnostic coverage of a defined set of faults. It is also possible that only fault detection is performed by the IP, with fault control being provided by components external to the IP.

The IP provider is responsible to provide the assumptions on use applied for the IP development in order to allow the IP integrator to check consistency with safety requirements.

The hardware features of the IP can be initially developed targeting its integration into a safety-related hardware environment, by providing safety mechanisms based on assumed safety requirements that aim at controlling given failure modes. In this case the requirements of ISO 26262-2:2018, ISO 26262-4:2018, ISO 26262-5:2018, ISO 26262-6:2018 (in the case of software based safety mechanisms to cover hardware failures), ISO 26262-8:2018, and ISO 26262-9:2018, whenever applicable, can be used for the design of the safety mechanisms during the development of the IP.

EXAMPLE 1 Bus "fabric" with built-in bus supervisors including error reporting logic (e.g. interrupt signals) and diagnostics (error capture information).

EXAMPLE 2 Voltage regulator with monitoring (under-voltage and over-voltage detection), protection (current limit or thermal protection) and diagnostics (monitoring and protection circuit built-in self-tests).

Alternatively the IP can be developed with no assumed safety requirements or specific safety mechanisms to detect and control faults.

EXAMPLE 3 Bus "fabric" without built-in bus supervisors or error reporting logic.

EXAMPLE 4 Voltage regulator without monitoring, protection or built-in monitoring or protection circuit diagnostics.

For IP with safety mechanisms, safety analyses defined in ISO 26262-9:2018, Clause 8 can be applied. A qualitative safety analysis and in some cases a quantitative analysis can be provided to the IP integrator to justify the capabilities of the safety mechanisms to control given failure modes. Similarly a dependent failures analysis can be provided to demonstrate required independence or freedom from interference.

NOTE 3 The IP supplier includes example information concerning failure mode distribution in the safety analysis results, based on specific implementation assumptions. Documentation related to safety mechanisms can be provided with other safety-related documentation for the IP. This information can also be combined into a single safety manual or safety application note as described in 5.1.11 (for digital components), 5.2.6 (for analogue or mixed signal components), 5.3.6 (for PLD) and 5.5.6 (for Sensors and Actuators).

NOTE 4 The base failure rate depends on the actual implementation of the IP into the integrated circuit and the use condition of the integrated circuit as described in 4.6 and so it can be provided for an IP as reference only. The IP integrator is in charge to recalculate such value according to the actual use case.

NOTE 5 This information can be included within existing documentation (e.g. integration guidelines, technical reference documents, application notes).

The IP integrator can request additional information from the IP supplier in implementing safety requirements. The IP supplier can support the request by providing information concerning measures used to avoid systematic faults, as well as safety analysis results. Safety analysis results can be used to support the determination of hardware metrics for the integrated IP, as well as to demonstrate freedom from interference and independence.

Since the IP will be integrated into a safety-related design, consideration of coexistence is important to ensure that the integrated IP can have no adverse impact on other safety-related functions. In order to claim freedom from interference, dependent failures analysis as described in ISO 26262-9:2018, Clause 6 and ISO 26262-9:2018, Clause 7 can be used, together with the additional guidance in 4.7 of this document.

If the IP integrator determines that the fulfilment of safety requirements is not possible with the supplied IP, a change request to the supplier can be done as described in ISO 26262-8:2018, 5.4.4 and ISO 26262-10:2018, 9.2.3 in cases where the IP is an SEooC. Alternatively, other measures by the IP integrator to comply with safety requirements can be applied, such as additional safety mechanisms at integration level or by fulfilling the requirements in ISO 26262-8:2018, Clause 13 and ISO 26262-8:2018, Clause 14, if applicable. Additional safety mechanisms can be implemented in hardware, software, or combination of both.

The IP integrator is responsible for each integration and associated verification and testing activities related to the allocated safety requirements and safety mechanisms, as applicable.

NOTE 6 The IP supplier provides supporting information to allow the IP integrator to conduct integration activities, including information on the verification and testing done for the IP.

4.5.3 Intellectual property lifecycle

4.5.3.1 Introduction

Avoidance and detection of systematic faults during the intellectual property lifecycle are required to ensure that the resulting design is suitable for use in applications with one or more allocated safety requirements. Requirements for avoidance and detection of systematic faults are provided in ISO 26262-5:2018, Clause 7 in the context of hardware design. In this part of ISO 26262, subclauses 5.1.9 (for digital components), 5.2.5 (for analogue or mixed-signal components), 5.3.5.3 (for PLD) and 5.5.5 (for Sensors and Transducers) provide further guidance. This guidance can be used to determine the general methods that can be used during IP development to avoid and detect systematic faults. Due to the wide range of IP designs with differing functionality and complexity, this guidance needs to be appropriately interpreted.

For IP which exhibits programmable behaviour, ISO 26262-4:2018, 7.4.5.2 can be considered.

The IP integrator is responsible for integrating the supplied IP. For the integration activities the assumptions of use and integration guidelines described for the IP are considered. The impact of assumptions of use which cannot be fulfilled or that are invalid with the design into which the IP is being integrated is analysed and considered with change management conducted as described in ISO 26262-8:2018, Clause 8. Figure 7 provides an example lifecycle based on SEooC development, as already provided in ISO 26262-10:2018, 9.2.3.

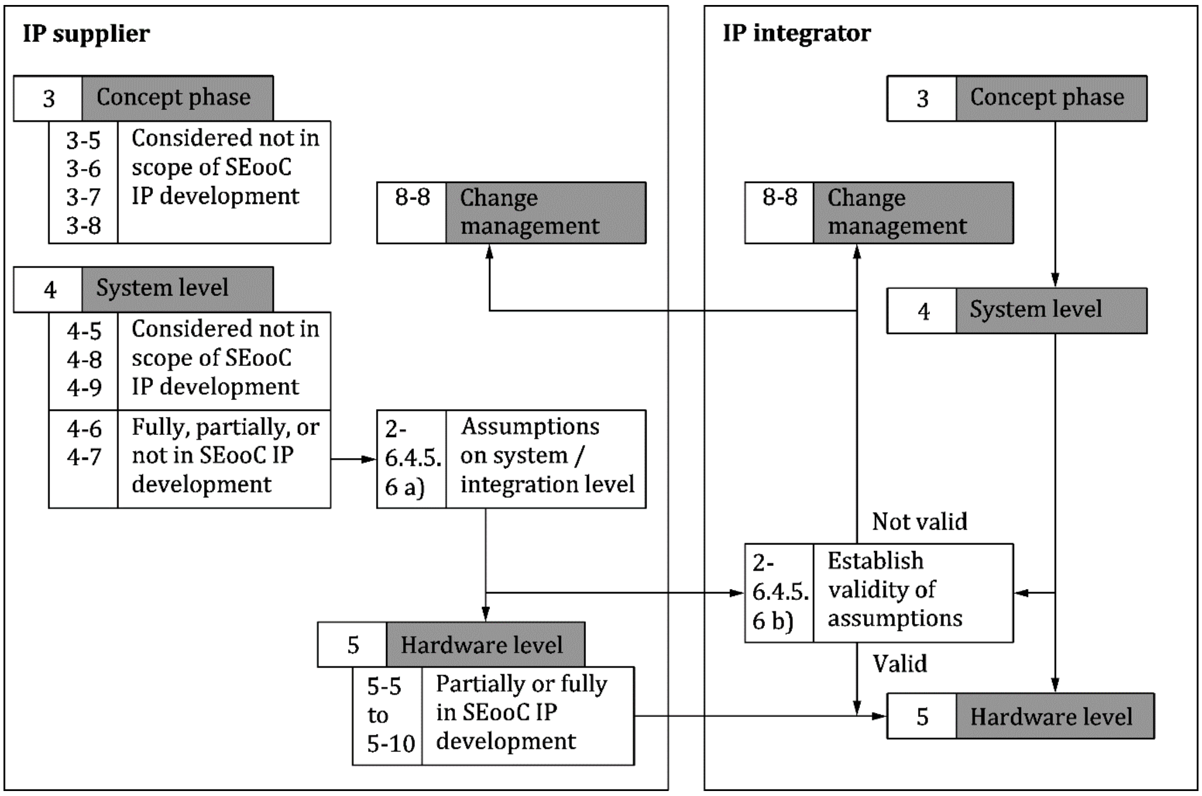

IP supplier
3 Concept phase
3-5 3-6 3-7 3-8 Considered not in scope of SEooC IP development
4 System level
4-5 4-8 4-9 Considered not in scope of SEooC IP development
4-6 4-7 Fully, partially, or not in SEooC IP development
8-8 Change management
2-6.4.5.6 a) Assumptions on system / integration level
5 Hardware level
5-5 to 5-10 Partially or fully in SEooC IP development
IP integrator
3 Concept phase
8-8 Change management
4 System level
Not valid
2-6.4.5.6 b) Establish validity of assumptions
Valid
5 Hardware level

Figure 7 — IP lifecycle when IP is treated as SEooC

NOTE ISO 26262-5:2018, Clause 10 is only partially the responsibility of the IP supplier because a number of the related requirements are not applicable to IP suppliers, such as ESD tests.

The DIA can define work products (as listed in 4.5.4) to be provided by the IP supplier to support the IP integrator in IP integration activities.

4.5.3.2 Intellectual property as safety element out of context (SEooC)

When developing an SEooC IP, applicable safety activities are tailored as described in ISO 26262-2:2018, 6.4.5.8. Such tailoring for the SEooC development does not imply that any step of the safety lifecycle can be omitted. In case where certain steps are deferred during the SEooC development, they can be completed during the item development.

In cases where a mismatch exists between the SEooC ASIL capability (see ISO 26262-1:2018, 3.2) and the ASIL requirements specified by the IP integrator, the IP integrator can implement additional safety mechanisms external to the IP. Additional safety measures for systematic failure avoidance are also considered. It is possible to use ASIL decomposition as defined in ISO 26262-9:2018 Clause 5, provided that it can be shown that there are redundant and independent requirements and the methods for systematic failure avoidance and control for the integrated IP are taken into account.

An SEooC is therefore developed based on assumptions of the intended functionality and use context which includes external interfaces. These assumptions are set up in a way that addresses a superset of components integrating the SEooC, so that the SEooC can be used later in multiple different designs. The validity of these assumptions is established in the context of the actual component integrating the SEooC.

IP developed as an SEooC can often be configured to target a number of different designs. Configuration can be done before synthesis, after synthesis, or by programming. Information provided by the IP supplier can include information on the IP configurations which have been covered by testing and verification activities.

EXAMPLE Configuration options to determine bus width for interconnects, internal cache memory sizes, number of interrupts, memory maps.

NOTE 1 IP configuration differs from configuration data for software, as described in ISO 26262-6:2018, Annex C.

NOTE 2 The IP integrator performs the necessary verification activities to guarantee the correctness of the generated IP; the necessary work products, as listed in following clauses, are made available; and the IP integrator verifies the correct integration of the IP in its context.

4.5.3.3 Intellectual property designed in context

When developing IP in context, the IP supplier tailors the safety activities as described in ISO 26262-2:2018, Clause 6.4.5.1. For in context designs, the IP supplier can develop the IP with knowledge of the safety requirements.

EXAMPLE This can be the case of an analogue component designed in context of a specific safety requirement at the system or element level.

4.5.3.4 Intellectual property use through evaluation of hardware element

In cases where no SEooC or in-context information is available for the IP, evaluation of hardware elements as described in ISO 26262-8:2018, Clause 13 can be used to increase confidence in the IP.

Activities foreseen for the evaluation of hardware elements can be applied to IP without pre-existing supporting information available (as described in 4.5.5).

4.5.3.5 Intellectual property use through the "proven in use" argument

The "proven in use" argument as described in ISO 26262-8:2018, Clause 14 can provide a means for the IP integrator to demonstrate that an IP design is appropriate for a particular application.

The conditions surrounding the validity of the "proven in use" argument can be restricting. Ensuring that an effective field monitoring program described in ISO 26262-8:2018, 14.4.5.3 is in place can be challenging due to the typically limited field feedback from designs incorporating IP or due to differences in IP configuration.

4.5.4 Work products for intellectual property

4.5.4.1 ISO 26262 and work products for intellectual property

Example work products are described in 5.1.11 (for digital components), 5.2.6 (for analogue or mixed signal components), 5.3.6 (for PLD) and 5.5.6 (for Sensors and Transducers). The following gives guidance on contents of work products which can be provided for IP designs in general.

NOTE A development interface agreement (DIA) can be used to specify which documents are made available to the IP integrator and what level of detail is included.

4.5.4.2 Safety plan

For IP with one or more allocated safety requirements, the safety plan is developed based on the requirements in ISO 26262-2:2018, 6.4.3.9. A single plan or multiple related plans can be used. Detailed plans are included for applicable supporting processes as described in ISO 26262-8:2018, covering configuration management, change management, impact analysis and change requests, verification, documentation management and software tool qualification.

4.5.4.3 Safety requirements of the IP design

The safety requirements can be allocated to the IP design as defined in ISO 26262-5:2018, Clause 6.

EXAMPLE The requirement for a safety mechanism in the IP is described, allowing the requirement to be verified at appropriate level of integration. The integration and test specifications can be linked to requirements defined in the technical safety concept.

4.5.4.4 Hardware design verification and verification review of the IP design

Defining criteria for design verification, in particular for environmental conditions (vibration, EMI, etc.), for an IP design which is provided in the form of logic design is not typically possible since the physical characteristics are highly dependent on the physical implementation of the design by the IP integrator.

For IP provided as a logical design, hardware design verification can be done using the techniques listed in 5.1.9.

A verification report includes results of the activities used to verify the IP design. Verification can be done as described in ISO 26262-8:2018, Clause 9, including planning, execution, and evaluation of verification activities.

NOTE Fault injection can be done using simulation as described in 4.8.

4.5.4.5 Safety analysis report

The requirements for safety analysis in ISO 26262-9:2018, Clause 8 are applicable for IP designs. The selection of appropriate safety analysis methods is based on ISO 26262-5:2018, Table 2.

For qualitative analysis, the supplier provides the identified failure modes of the IP in order to support its integration.

For quantitative analysis, the data included supports the evaluation of hardware architectural metrics and evaluation of safety goal violations due to random hardware faults, as specified in ISO 26262-9:2018, 8.4.10.

EXAMPLE Data includes estimated failure rate and failure mode distribution information.

NOTE 1 For IP provided as logical design, such as Register Transfer Level (RTL), quantitative analysis relies on assumptions about failure rates and failure mode distributions, and can therefore not be representative of actual physical designs. The IP integrator verifies the assumptions and quantitative safety analysis results for the specific implementation.

NOTE 2 In estimating the metrics, safety mechanisms embedded in the IP and their expected failure mode coverage (at a level that is applicable to the given IP) can be considered.

In the case of configurable IP, the safety analyses can include information about the impact of configuration options on the failure modes distribution.

NOTE 3 An analysis of the impact of configuration options on the implementation and diagnostic coverage of safety mechanisms is performed.

Additional safety mechanisms realized by a combination of features internal and external to the IP, as well as safety mechanisms implemented outside the IP can be defined. These additional safety mechanisms can rely on assumptions of use for the SEooC design, which can be validated at the appropriate level as described in ISO 26262-2:2018, 6.4.5.8.

4.5.4.6 Analysis of dependent failures

Dependent failures analysis for IP can be performed as described in ISO 26262-9:2018, Clause 7. Additional guidance on how to apply dependent failures analysis for semiconductor devices is included in 4.7 of this part.

4.5.4.7 Confirmation measure reports

Reports from conducted confirmation measures include evidence and arguments related to the IP development process and about avoidance of systematic faults. Confirmation measures are described in ISO 26262-2:2018, Table 1. For semiconductor IP typical confirmation measure reports include:

— Confirmation review of the safety plan;

— Confirmation review of the safety analyses;

— Confirmation review of the software tool criteria evaluation report;

— Confirmation review of the proven in use arguments, if applicable;

— Confirmation review of the completeness of the safety case; and

— Functional safety audit and assessment reports.

Examples of techniques applicable to IP development activities for systematic fault avoidance are included in 5.1.9 (for digital components), 5.2.5 (for analogue or mixed-signal components), 5.3.5.3 (for PLD) and 5.5.5 (for Sensors and Transducers).

4.5.4.8 Development interface agreement

The requirements for distributed development in ISO 26262-8:2018, Clause 5 are applicable to IP designs. The DIA defines the exchanged work products for IP designs, and the roles and responsibilities for safety between the IP supplier and the IP integrator.

4.5.4.9 Integration documentation set

An integration documentation set can include a safety manual or safety application note for IP developed as an SEooC. The integration documentation set can also include the following information:

— Description of the tailoring of the ISO 26262 lifecycle for the IP development;

— Assumptions of use for the IP, including for example:

  — Assumed safe states of the IP;

  — Assumptions on FTTI and Multiple Point Fault Detection Interval (MPFDI);

  — Assumptions on the integration environment for the IP, including interfaces; and

  — Recommended IP configurations.

— Description of the safety architecture, including:

  — Fault detection and control mechanisms;

  — Fault reporting capabilities;

  — Self-test capabilities and additional requirements for self-testing for potential latent faults, if applicable;

  — Fault recovery mechanisms, if applicable; and

  — Impact of configuration parameters on the above items if applicable.

— Hardware-software interfaces required to support the IP safety mechanisms, and to control failures after detection;

— Specification of software-based test routines to detect faults of the IP component, if applicable. This could also be provided as source code or binary library;

— Description of safety analysis results for the IP; and

— Description of confirmation measures used for the IP.

NOTE 1 The above information can be included in one or more separate documents, sometimes called safety manuals or safety application notes.

It is possible for the IP integrator to formally identify each hardware feature related to the safety mechanisms so that a mapping with technical safety requirements and hardware safety requirements at the level of the IP integrator can be done, and the verification and validation activities that are the responsibility of the IP integrator can be identified.

NOTE 2 The IP safety mechanism requirements are specified in a way which allows them to be traceable to IP integrator's requirements.

NOTE 3 For IP with no specific features for fault detection, providing the assumptions of use can be sufficient to comply with the IP integrator's requirements.

For IP developed in-context, similar documentation is typically provided.

NOTE 4 For in-context IP, assumptions of use are not required, as the IP is designed with full context information in place.

4.5.4.10 Applicability of work products to IP categories

The applicability of the work products described in 4.5.4.1 to 4.5.4.9 depends on the classification of the IP as described in 4.5.2. For intellectual properties without integrated safety mechanisms:

— the safety analysis report is limited to the failure modes distribution of the IP, while there is not any estimation of the hardware metrics because there is not any integrated safety mechanism. The failure mode distribution is necessary to give the possibility to the IP integrator to properly perform the safety analysis at its integration level;

— the integration documentation set (not a specific work product but it is rather the collection of information as described in 4.5.4.9) is limited to the description of the assumptions on the integration environment for the IP, including interfaces;

— it does not typically include the analysis of dependent failures.

4.5.5 Integration of black-box intellectual property

In some projects the IP integrator can encounter a situation where it is necessary to integrate an IP which contents are not fully disclosed. The IP to be integrated is a "black box" from the perspective of the IP integrator.

EXAMPLE 1 IP integrator's customer requires use of their proprietary logic, such as a specific communications interface, timer peripheral, or similar logic.

EXAMPLE 2 IP integrator is asked to integrate logic from a competitor, in order to facilitate a multi-source supply agreement.

Black box IP can be integrated in many forms, including but not limited to:

— Pre-hardened, or handed off as a gate level layout;

— As encrypted netlist, which cannot be meaningfully parsed except by trusted tools; and

— As obfuscated RTL source (where meaningful variable names are replaced with randomized character strings and any explanatory comments are removed).

NOTE 1 A black box integration approach can also be applied to cases in which no information is available from the IP supplier.

When black box IP is integrated, the responsibility between IP supplier, IP integrator and the IP integrator's customer can be defined through a development interface agreement as described in ISO 26262-8:2018, Clause 5.

EXAMPLE 3 In cases where the IP integrator is required to use black box IP, for example because of a requirement from their customer, the DIA can specify that it is the customer responsibility to evaluate and accept the suitability for the use of the black box IP in a safety-related context.

The development interface agreement can also include details about the tailoring of the safety activities as described in ISO 26262-2:2018, 6.4.5.8 and the exchange of documentation across the supply chain.

EXAMPLE 4 A development interface agreement can specify that integration details are provided by the IP supplier in the form of an integration guide, also containing a set of validation tests which can be used to confirm proper integration.

Unless the IP has been developed specifically targeting the automotive market, it is possible that ISO 26262 specific evidence is not available. In this case the responsibility for the acceptance of available evidence can be defined in the development interface agreement.

EXAMPLE 5 IP developed according to other functional safety standards such as IEC 61508:2010.

NOTE 2 In this case information on the development lifecycle and associated processes used to develop the IP can be used to perform a gap analysis to evaluate the suitability of the IP for use in an ISO 26262 context.

The IP integrator does not always have enough data to evaluate the base failure rate of a black box IP. Since this can affect the results of quantitative analysis, the development interface agreement can specify the responsibilities between the IP supplier, IP integrator and the IP integrator's customer for the estimation of the base failure rate. The responsibilities for safety analysis of the black box IP can be defined in a similar way.

NOTE 3 The integration of black box IP into a hardware development has parallels in software development, such as the case in which a developer integrates unit software from a third-party supplier as compiled object code. As such, the integrator of black box IP into a hardware development can find methods and techniques in 5.1.9.1 including the link with applicable tables of ISO 26262-6:2018.

In cases where the black box IP requires safety mechanisms, the IP integrator may not have enough information to implement the safety mechanism outside of the IP. The development interface agreement specifies requirements for such safety mechanism in these cases.

4.6 Base failure rate estimation

4.6.1 About base failure rate estimation

4.6.1.1 Impact of failure mechanisms on base failure rate estimation

The scope of this clause is to give clarifications, guidelines and examples on how to calculate and use the base (or raw) failure rate. Base failure rate is a primary input for calculation of quantitative safety metrics according to ISO 26262-5:2018.

Quantitative safety analysis in ISO 26262 focuses on random hardware failures and excludes systematic failures.

Each technique available for base failure rate estimation makes assumptions of the failure mechanisms to be considered. Differences in results obtained from different base failure rate estimation techniques are often due to a lack of consideration for the same set of failure mechanisms amongst techniques. Results from use of different techniques applied to the same component are unlikely to be comparable without harmonization on a common set of failure mechanisms.

EXAMPLE Harmonization can be done, for instance, by considering the same failure mechanisms and the same source of stresses.

Failure mechanisms for semiconductors are dependent on circuitry type, implementation technology, and environmental factors. As semiconductor technology is rapidly evolving, it is difficult for published recognized industry sources for failure rate to keep pace with the state of the art, particularly for deep submicron process technologies. Because of this, it can be helpful to consider the publications of industry groups such as JEDEC (Joint Electron Device Engineering Council), ITRS (International Technology Roadmap for Semiconductors), and the SEMATECH/ISMI Reliability Council to get a broad view of semiconductor state of the art.

JEDEC publishes several documents which can be helpful in providing references to understand and estimate specific failure mechanisms:

— Reference [16] summarises many different well understood and industry accepted failure mechanisms for silicon and packaging; it can also be used to provide a physics of failure model for estimation of failure rates for the identified failure mechanisms;

— Reference [54] provides guidance on developing a reliability evaluation methodology based on an application-specific use model (mission profile); and

— Reference [17] summarises a number of transient fault mechanisms related to exposure to naturally occurring radiation sources and provides guidance on how to experimentally derive failure rates for susceptibility to soft error.

4.6.1.2 Considerations in base failure rate estimation for functional safety

SPFM, LFM, and failure rates used for the quantitative safety analysis like calculation of PMHF are sometimes misunderstood as a reliability prediction. A careful distinction between reliability and functional safety is necessary.

4.6.1.2.1 Distinguish between systematic and random failures

ISO 26262 makes a distinction between systematic and random failures. Most available techniques for base failure rate estimation are intended to provide reliability estimates and make no such distinction. The result of such techniques can be excessively conservative compared to actual value due to inclusion of factors which estimate systematic failures. For example, estimation techniques based on observations of field failures do not, in general, have appropriate sample size or observation quality to differentiate between systematic and random failures. Similarly, models which include systematic capability as part of the base failure rate calculation can be challenging to use in an ISO 26262 context (e.g. Π_{pm} and $\Pi_{process}$ factors defined in [9]).

4.6.1.2.2 Effect of failure recovery mechanisms

Another concern with reliability standards is the handling of diagnostics which can be used to enhance availability. This leads to mix the base failure rate with diagnostics while the ISO 26262 requires separating them for the metrics computation.

EXAMPLE Consider a common SEC-DED (Single Error Correct-Dual Error Detect) EDC-ECC used in many state of the art automotive functional safety electronics. A reported MTTF (mean time to failure) for an SRAM with SEC-DED EDC-ECC-EDC cannot consider a fault which results in a correctable error – thus mixing effects of base failure rate and diagnostics, which is separated for calculation of ISO 26262 metrics.

4.6.1.2.3 Considerations about non-constant failure rates

Many standardised models make use of a “bathtub curve” simplification, which assumes that early life (infant mortality) defects have been effectively screened by the supplier and that “wear out” (end-of-life) failure mechanisms, such as electro-migration, time–dependent dielectric breakdown, hot carriers, or negative bias temperature instability will effectively occur at negligible rates during useful mission lifetime.

In some cases, the failure rate distribution from reliability models does not fit well to the constant failure rate of the "bathtub curve" simplification. Using a non-constant failure rate in ISO 26262 random failure analysis is problematic as it can result in failure rates which are not constant. While such an approach is not explicitly prohibited by ISO 26262, the approach is not typically applied under current state of the art.

If one chooses to simplify complex non-constant failure rate distributions by using approximations of constant failure rate, some options include:

— Assume a constant failure rate set conservatively at the maximum failure rate of the reliability model failure rate distribution, or

— Depending on the distribution, it can be possible to limit the operating lifespan of the product such that a constant failure rate approximation is more appropriate. This case often applies when a beginning-of-life or end-of-life mechanism becomes dominant in the overall failure rate distribution.

NOTE If a bathtub model is used, reaching the end of the bathtub within the product lifetime is a systematic issue. If this is acceptable or not is not evaluated within the hardware metrics of ISO 26262-5:2018 Clause 8 and Clause 9. This is evaluated separately.

If the overall failure rate distribution is a result of integrating multiple fault models, separation of failure modes can result in the ability to simplify safety analysis by evaluating the impact of each failure mode separately using different (but constant) failure rate approximations, as recommended in 5.1.7.2 for consideration of transient faults.

4.6.1.3 Techniques and sources for base failure rate estimation

There are many different techniques which can be utilised for base failure rate estimation. In general these techniques can be summarised as follows:

— Failure rates derived from experimental testing, such as:

  — High Temp Operational Life (HTOL) testing for gate oxide breakdown,

  - Reliability test chip and/or on-chip test structures to assess intrinsic reliability of the silicon technology,
  - Soft error testing based on exposure to radiation sources, or
  - Convergence characteristic of acceleration test for screening.
- Failure rates derived from observation of field incidents, such as analysis of material returned as field failures;

  NOTE 1 For permanent faults: data provided by semiconductor industries can be based on the number of (random) failures divided by equivalent device hours. These are obtained from field data or from accelerated life testing (as defined in standards such as JEDEC and AEC) scaled to a mission profile (e.g. temperature, on/off periods) with the assumption of a constant failure rate (random failures, exponential distribution). The numbers can be used as inputs for the estimation of the failure rate, provided as a maximum failure rate based on a sampling statistics confidence level.

- Failure rates estimated by application of industry reliability data books and/or estimated by procedural models, such as:
  - IEC TR 62380 [41],
  - SN 29500 [39], or
  - FIDES Guide [9].

  NOTE 2 The actual failure rate achieved is expected to be lower than the failure rate derived from the handbooks.

  NOTE 3 If properly supported by evidence, the base failure rates derived from standards and handbooks can be de-rated by considering other factors such as density of registers and probability of occurrence of permanent faults between key-on and key-off, etc.

- Enhanced data books incorporating physics of failure elements;
- For transient faults the base failure rate can be derived from data provided by semiconductor industries derived from JEDEC standards such as JESD89A [17]; International Technology Roadmap for Semiconductor (ITRS).

4.6.1.4 Documentation on the assumptions for base failure rate calculation

When calculating the base failure rate the supplier provides a documentation describing the assumptions made and supporting rationale.

NOTE For example, assumptions can be:

- the selected method to calculate the failure rate (e.g. industry source or field data),
- how the non-operating time and solder joint were taken into account, or
- which model has been used for failure rate derived from field data (Weibull or exponential models).

4.6.2 Notes on base failure rate estimation

4.6.2.1 Transient fault quantification

As described in ISO 26262-1:2018, electromagnetic interference (EMI) and soft errors (e.g. SEU and SET) are possible causes of transient faults.

Transient faults causing soft errors initiated by internal or external α, β, neutron, or γ radiation sources are random hardware failures that can be quantified with a probabilistic method supported by measured data.

Transient faults caused by EMI or cross-talk are not quantified. Even if they can lead to the same effects as other transient faults, they are mostly related to systematic causes. These can be avoided with proper techniques and methods during design phase (e.g. cross-talk analysis during component development back-end).

ISO 26262-5:2018, 8.4.7 NOTE 2 specifies that the transient faults are considered when shown to be relevant due, for instance, to the technology used. Therefore, depending on the impact of the faults and when applicable, they can be considered in the safety analysis. The analysis for transient faults and permanent faults is done separately. This holds for qualitative or quantitative analysis.

Each elementary sub-part type (flip flops, latches, memory elements, analogue devices) is investigated if it is relevant to soft error rate, specifically with respect to direct or induced alpha particles and neutrons. The relevance to those phenomena depends on the semiconductor front end technology and the materials on top of the die's surface including the package, e.g. the mould compound and the solder material (flip chip).

EXAMPLE 1 Base failure rate for alpha particles can be influenced by the type of package, e.g. low alpha (LA) or ultra-low alpha (ULA) emitting semiconductor assembly materials.

Depending on factors such as the technology and on the operating frequency, transient fault models like single event upset (SEU), multiple-bit upset (MBU) and single event transient (SET) are considered as in references [2] and [22].

NOTE 1 Destructive single event effects like Single Event Latch-up (SEL), Single Event Burnout (SEB), and Single Event Gate Rupture (SEGR) are not considered as transient faults because these faults lead to permanent effects.

JESD89A [17] is considered as the main reference related to measurement and reporting of alpha particle and terrestrial cosmic ray-induced soft errors in semiconductors. In that context, the base failure rate for soft errors is provided together with the conditions in which it has been computed or measured.

NOTE 2 Conditions such as neutron particle flux, altitude, temperature, and supply voltage are relevant to transient failure rate estimation of soft errors. JESD89A [17] is used to understand those conditions.

ISO 26262-5:2018, 9.4.2.4 NOTE 5 states that in applying a selected industry source the following considerations are appropriate to avoid artificial reduction of the calculated PMHF: mission profile; the applicability of the failure modes with respect to the operating conditions; or failure rate.

EXAMPLE 2 In case of soft errors, reducing the base failure rate considering only the operating time of the vehicle leads to an excessive and therefore artificial reduction of the average probability per hour.

NOTE 3 If semiconductor provider delivers a de-rated soft error rate, information about the de-rating factor is made available for example in the Safety Manual as defined in 5.1.11 (for digital components), 5.2.6 (for analogue or mixed signal components), 5.3.6 (for PLD) and 5.5.6 (for Sensors and Transducers).

Moreover, the base failure rate for soft errors is provided without de-rating it with respect to "architectural vulnerability factors" nor effect of safety mechanisms such as EDC-ECC.

NOTE 4 Architectural vulnerability factor (AVF) is the probability that a fault in a processor structure will result in a visible error in the final output of a program as described in reference [25].

NOTE 5 Vulnerability factors are considered in the consideration of the amount of safe faults, as described in 5.1.7.2.

4.6.2.2 Component package failure rate

In the estimation of a hardware component failure rate, the semiconductor providers consider the failures relating to the silicon die, to the enclosure/encapsulation (e.g. case) and to the connection points (e.g. pins). The connections between the connection points to the board (e.g. solder joints) are considered as board failures and are typically considered by the system integrator during the safety analysis at the system or element level.

NOTE 1 The package failure rate $\lambda_{package}$ as calculated in IEC TR 62380, Clause 7.3.1 corresponds to the fault models inside of the package itself (including e.g. the connection between the die and the lead frame) but it also includes the failure rate related to the connection between the package connection points and the board (solder joints).

NOTE 2 The failure rate of the hardware component calculated in SN 29500-2 includes the failure models related to the die and to the package however unlike IEC TR 62380 it does not include the failure rate of the connection between the package connection points and the board which is treated separately in SN 29500-5.

NOTE 3 FIDES Guide provides separate failure rates for package (cases) and solder joints due to thermal cycling.

NOTE 4 In reality, the failure rate of the connection between the package connection points and the board is, in reality, dependent on many factors involving the specific design of the circuit board and how the board is packaged inside of a protective housing. These factors are constantly changing as both electronic components and circuit board material technologies rapidly evolve.

4.6.2.3 Consideration of power-up time and power-down time

According to ISO 26262-5:2018, 9.4.2.4 NOTE 5, in applying a selected industry source the following considerations are appropriate to avoid artificial reduction of the calculated PMHF:

— mission profile,

— the applicability of the failure modes with respect to the operating conditions, or

— failure rate.

The base failure rate is provided along with the mission profile used. If the power-up and power-down times are defined in the mission profile then they can be used to compute stress factors as foreseen by reliability handbooks like IEC TR 62380 (τ_{on} and τ_{off}) and SN 29500 (π_w).

4.6.3 Permanent base failure rate calculation methods

4.6.3.1 Permanent base failure rate calculation using industry sources

4.6.3.1.1 IEC TR 62380

IEC TR 62380 [41], Clause 7.3.1 describes a calculation method for the base failure rate of both die and package. Several parameters are used to determine the failure rate:

— A parameter per transistor per type of technology used (λ_1). A λ_1 value is provided for different type of integrated circuits families in IEC TR 62380, Table 16;

— A parameter related to the mastering of the technology and valid for the whole component regardless the number of integrated elements (λ_2);

— A parameter related to the package (λ_3);

— A parameter related to the number of transistors of the hardware component (N);

— A parameter related to the difference between the year of manufacturing or technology release/update and the reference year (1998) (α);

— A parameter related to the operating and non-operating phases seen by the hardware component (τ, τ_{on} and τ_{off});

— A parameter related to a temperature stress factor (π_t) applicable to the die part of the component;

— A parameter related to the number and the amplitude of the temperature cycling seen by the hardware component (n_i and ΔT_i); and

— A parameter related to the mismatch between the thermal coefficients of the board and the package material (α_S and α_C).

Selection of parameters can be done based on the process technology and type of circuitry utilised by the design.

NOTE 1 In IEC TR 62380, Table 16, “actual number” corresponds to the real number of transistors regardless the sizes of those transistors.

NOTE 2 To calculate the digital component die failure rate for the whole device, the number of equivalent gates is used. The number of effective equivalent transistors is computed by multiplying the equivalent gate count by the representative number of transistors per gate. When calculating the microcontroller die failure rate due to CMOS digital logic, the contribution of each digital logic of the modules (e.g. CPU, CAN, Timer, FlexRay, SPI) is included in N.

NOTE 3 The process maturity de-rating factor was introduced considering Moore’s law and the fact that device failure rates are more or less constant. If the failure rate per transistor would have stayed the same, the failure rate would have increased according to Moore’s law. This was not observed. Therefore, the transistor failure cannot stay constant when changing process nodes. Reference [41] suggests using the manufacturing date. Optionally, to reflect process technology changes, the year of first introduction of this particular technology node can be used instead of its year of manufacturing. To achieve independence from the silicon vendor, the year from the International Technology Roadmap for Semiconductor (ITRS) can be used [42].

NOTE 4 For analogue parts or for the digital component built primarily on analogue process technologies, the "Linear Circuits" entry of Table 16, "MOS : Standard circuits (3)" in [41] can be used, unless more precise data are provided by the microcontroller vendor.

Once the base FIT rate for the digital component die has been generated, a de-rating factor is applied based on thermal effects and operating time. The de-rating factor is considered based on:

— Junction temperature of the digital component die, which is calculated based on:

 — power consumption of the digital component die; and

 — package thermal resistance, based on package type, number of package pins and airflow;

— An application profile which defines 1 to Y usage phases, each of which is composed of an application “on-time” as a percentage of total device lifetime, and an ambient temperature. IEC TR 62380 provides two automotive reference profiles: “motor control” and “passenger compartment”; and

— Activation energy and frequency per technology type to complete the Arrhenius equation.

NOTE 5 Data specific to the product under consideration, such as package thermal characteristics, manufacturing process, Arrhenius equation, etc., could be used in replacement of the general factors in IEC TR 62380 to achieve a more accurate estimation of FIT rate.

4.6.3.1.1.1 Die base failure rate calculation using IEC TR 62380

Multiple interpretations exist about how to combine λ_1 and λ_2 in the case of circuit elements with different technologies (CPU, memories, etc.) implemented in the same device.

In one interpretation, each circuit element inherits the λ_1 and λ_2 of the respective technologies, so basically the λ_1 and λ_2 are summed – as shown in Table 2.

In this example, we assume a CMOS technology based MCU which consumes 0,5 W power. The digital component die is packaged in a 144 pin quad flat package and cooled by natural convection. The MCU is exposed to the “motor control” temperature profile.

The resulting increase of the junction temperature ΔT_j is 26,27 °C. An activation energy of 0,3 eV is assumed for the Arrhenius equation. Using the de-rating formula in paragraph 7.3.1 of [41], this results in a de-rating factor (i.e. the second factor of λ_{die}) of 0,17.

Table 2 — Digital component example with summed λ_2

Circuit Element	**λ_1 (FIT)**	**N (gate or transistors)**	**α**	**λ_2 (FIT)**	**Base failure rate (FIT)**	**De-rating for temp**	**Effective failure rate (FIT)**
50k gate CPU	$3,4 \times 10^{-6}$	200000 (4 transistors/gate)	10	1,7	1,73	0,17	0,29
16kB SRAM	$1,7 \times 10^{-7}$	786432 (6 transistors/bit for a low-power consumption SRAM)	10	8,8	8,802	0,17	1,50
						Die failure rate (FIT)	1,79

As alternative approach, it is possible (see Table 3) to use the following equation with a single (conservative) maximum λ_2 as representative value:

$$\lambda_{die} = \left\{ \sum_{elements} \left(\lambda_{1,element} \times N_{element}\right) \times e^{-0,35\times\alpha} + \max(\lambda_{2,element}) \right\}$$
$$\times \sum_{elements} \left\{ \left\{ \frac{\sum_{i=1}^{y} \left(\pi_{t,element}\right)_i \times \tau_i}{\tau_{on} + \tau_{off}} \right\} \times \frac{N_{element}}{N_{total}} \right\}$$

Table 3 — Mixed signal example with max of λ_2

Circuit Element	λ_1 (FIT)	N (gate or transistors)	α	Base failure rate without λ2 (FIT)	λ_2 (FIT)	De-rating for temp	Effective failure rate (FIT)
Digital Circuits	1,0*10^{-6}	28000	10	0,00085	1,7		
Linear/digital circuits low voltage (< 6V)	2,7*10^{-4}	30000	10	0,25	20		
Die failure rate (FIT)				0,25	Max(20,1,7) = 20	0,17	3,4

To simplify calculation, if the user can identify a match between its product and one of the integrated circuits families type listed in IEC TR 62380 then – as shown in Table 4 below - the user can directly apply the failure rate calculation method as described in IEC TR 62380, Clause 7.3.1.

Table 4 — Digital component example with matching device type

Circuit Element	λ_1 (FIT)	N (gate or transistors)	α	λ_2 (FIT)	Base failure rate (FIT)	De-rating for temp	Effective failure rate (FIT)
50k gate CPU	3,4 x 10^{-6}	200000 (4 transistors/gate)	10	1,7	1,80	0,17	0,31
16kB SRAM		786432 (6 transistors/bit for a low-consumption SRAM)					
						Die failure rate (FIT)	0,31

4.6.3.1.1.2 Temperature de-rating using IEC TR 62380

In Table 2, Table 3 and Table 4, the de-rating factor is calculated after considering the τ_{on} and τ_{off} time. According to ISO 26262-5:2018, 9.4.2.4 NOTE 5, τ_{off} can be set to zero. Consequently, the resultant base failure rate is usually more conservative. In the above digital component example of Table 4, setting the τ_{off} to zero gives a de-rating factor of 2,91, therefore the base failure rate value changes from 0,31 to 31 FIT.

The formula used in clause 7.3.1 of [41] ("MATHEMATICAL MODEL") to calculate the temperature de-rating factor δ_T uses the following parameters:

— $(\pi_t)_i$: ith temperature factor related to the ith junction temperature of the integrated circuit mission profile;

— τ_i : ith working time ratio of the integrated circuit for the ith junction temperature of the mission profile;

— τ_{on} : total working time ratio of the integrated circuit, with $\tau_{on} = \sum_{i=1}^{y} \tau_{\mathrm{i}}$;

— τ_{off} : time ratio for the integrated circuit being in storage (or dormant);

— $\tau_{on} + \tau_{off} = 1$.

For the calculation of a conservative temperature de-rating factor, the off time τ_{off} can be set to zero, resulting in a slightly modified version of δ_T for the temperature de-rating factor $\delta_{T,conservative}$:

$$\delta_{T,conservative} = \frac{\sum_{i=1}^{y} (\pi_t)_i \times \tau_i}{\tau_{on}}$$

4.6.3.1.1.3 Package base failure rate calculation using IEC TR 62380

The package failure rate $\lambda_{package}$ as calculated in IEC TR 62380, Clause 7.3.1 corresponds to the failure modes inside of the package itself (including e.g. the connection between the die and the lead frame) but it also includes the failure rate related to the connection between the package connection points and the board (solder joints) which represents approximately 20 % of the overall $\lambda_{package}$ FIT rate as described in [55]. The semiconductor provider may then use 80 % of $\lambda_{package}$ value for the distribution of the hardware component package FIT rate.

As shown in Table 5, the package failure rate calculation takes into account the following parameters:

— π_α : influence factor related to the thermal expansion coefficients difference between the mounting substrate and the package material;

— $(\pi_n)_i$: ith influence factor related to the annual cycles number of thermal variations seen by the package, with the amplitude ΔT_i;

— ΔT_i : ith thermal amplitude variation of the mission profile; and

— λ_3: base failure rate of the integrated circuit package.

Table 5 — Package base failure rate calculation example

Package type	**ΔTj (°C)**	**S (Number of pins)**	**D (mm)**	**Π_α**	**λ_3 (FIT)**	**De-rating for temperature cycling**	**Effective failure rate (FIT)**
PQFP 144	26,27	144	26,58	1,05	11,87	6027	207
Package failure rate including solder joints between package and board (FIT)							207
Total package failure rate without solder joints between package and board (FIT)							166

The influencing factor π_α is calculated using the formula shown in clause 7.3.1 of [41] ("Mathematical expression of the influence factor"), with α_s, α_c being the linear thermal expansion coefficients for the substrate and for the component respectively. In this example, we assume FR4 as mounting substrate and a plastic package for which the table delivers the values $\alpha_s = 16$ and $\alpha_c = 21{,}5$.

For an automotive profile, the number of cycles/year ≤ 8 760$(\pi_n)_i$ is calculated using the formula in clause 7.3.1 of [41] ("Mathematical expression of the influence factor $(\pi_n)_i$ "), with n_i : Annual number of cycles with the amplitude ΔT_i.

To calculate λ_3 in FIT, the formula for peripheral connections packages is used, using a width of 20 mm and a pitch of 0,5 mm as shown in Table 17b of [41]. Using the "motor control" temperature profile, this results in a total failure rate for the package without solder joints of:

$\lambda_{package}$ = 166 FIT.

The package failure rate is assumed to be equally distributed among the pins, leading to a pin failure rate of:

λ_{pin} = 1,15 FIT.

NOTE 1 The package in the example is a 144 pin quad flat package and cooled by natural convection. The power consumption is 0,5 W leading to an increase of the junction temperature ΔT_j of 26,27 °C. The value of D and λ_3 are computed using the Table 17b in reference [41] on the basis of the following values: pitch = 0,5 mm and width = 20 mm.

NOTE 2 In the case λ_3 value provided by IEC TR 62380 is not suitable, the supplier of the component can replace this value with supplier's internal base failure rate or with a more up to date value from other industry sources. In such case an argument is provided by the supplier to justify the value of the base package failure rate that has been used.

NOTE 3 Package failure rate estimation is based on the knowledge of the construction and thermal characteristics of the device package and the system's printed circuit board. Instead of using [8], a joint conservative estimation of package FIT rate by semiconductor supplier and system implementer can be used.

4.6.3.1.1.4 Example of failure rate resulting from electrical overstress

The failure rate for whole device due to electrical overstress can be calculated using the formula shown in clause 7.3.1 of [41] ("MATHEMATICAL MODEL"). If the device has a direct connection to the external environment, i.e. the device is an interface, π_I is equal to one. If the device is not an interface, i.e. it has no direct connection to the external environment, π_I is equal to zero.

The reference [41] shows different λ_{EOS} for various electrical environments. Unfortunately, an automotive electrical environment is not given. Instead the "civilian avionics (on board calculators)" can be chosen:

$\lambda_{EOS} = 20\,FIT$.

This results in a failure rate due to electrical overstress for the whole device of either

$\lambda_{overstress} = 20\,FIT$, if the device has a direct contact to the external environment, or

$\lambda_{overstress} = 0\ FIT$ in every other case.

To forecast the impact of electrical overstress on the device is non-trivial. If no particular impact can be argued, then $\lambda_{overstress}$ can be added to λ_{die} to increase the overall digital component die failure rate of the whole device.

NOTE Electrical over-stress can be considered a systematic failure mode and reduced to zero FIT for calculation of hardware random failure metrics.

4.6.3.1.2 SN 29500

The SN 29500 follows a table look up approach. Expected values for failure rates under specified reference conditions are given. Values are to be looked up in tables using product type, technology and transistor count as an input. If the integrated circuits are operated under conditions different from the reference conditions a calculation from reference to operating conditions is to be used. The calculation takes into consideration temperature, voltage and drift (for analogue elements). For the temperature part of the calculation to operating conditions a modified Arrhenius equation is used.

4.6.3.1.2.1 Example of computation for a semiconductor component

Parameters required for the calculation of the failure rate with SN 29500:

— N, the number of equivalent transistors;

— λ_{ref}, the basic failure rate for the hardware component, based on the process technology;

— ΔTj, the junction temperature increase; and

— The mission profile of the hardware component.

NOTE 1 In the case the number of equivalent transistors N is not listed in the failure rates families tables 1, 2 or 3 of SN 29500-2:2010 and when possible the user can use an interpolation or extrapolation method to determine the equivalent λ_{ref} and $\theta_{vj,1}$ (virtual junction temperature) values.

NOTE 2 The values regarding mission profiles are only examples. The requirements for all semiconductors within an ECU are aligned with the requirements of the respective ECU specifications.

EXAMPLE For "microprocessors and peripherals, microcontrollers and signal processors" family as defined in SN 29500-2, Table 2, the following interpolation example is done to determine λ_{ref} and $\theta_{vj,1}$ values. Assuming a microcontroller with 650 million transistors the calculation of the λ_{ref} could be done using the following steps:

— determination of $\theta_{vj,1}$ (650M) using an interpolation of virtual temperatures from table;

— calculate the λ_{ref} for 650 million transistors using the π_t or the Arrhenius factor and the λ_{ref} for 100 million transistors value from SN 29500-2, Table 2:

$$\frac{\lambda_{ref(100M)}}{\lambda_{ref(650M)}} = e^{\left(\left(\frac{E_a}{k}\right)\times\left(\frac{1}{273+\theta_{vj,1\{650M\}}}-\frac{1}{273+\theta_{vj,1\{100M\}}}\right)\right)}$$

$$\lambda_{ref(650M)} = \frac{150}{0{,}79} = 189\ FIT$$

4.6.3.1.2.2 Failure rate calculation for the semiconductor component example without non-operating phase

For the digital component example described in previous clauses, in CMOS technology with 500k to 5 million transistors we get 80 FIT at 90 degree Celsius reference temperature condition. The following parameters are listed in Table 6 and Table 7:

— A, constant;

— E_{a1}, E_{a2}, constant activation energy in eV.

Table 6 — Parameters required for failure rate calculation example with SN 29500

N (transistors)	Technology and family	λ_{ref} (FIT)	ΔT_j (°C)	Temperature dependent reference (Z_{ref}) (1/eV)	A	E_{a1} (eV)	E_{a2} (eV)
986432(Digital + SRAM)	CMOS, microprocessor	80	26,27	5,11	0,9	0,3	0,7

Assuming 500 working hours per year and using the motor control mission profile as defined in IEC TR 62380 [41], we have the result of Table 7.

Table 7 — Digital component failure rate calculation example with SN 29500

Ambient temperature θ_U (°C)	Working time (h)	Junction temperature $\theta_{j,2}$ (°C)	Dependence factor Z (1/eV)	Temperature dependence factor $\Pi_T(\theta_u)$
32	172,4	58,27	2,04	0,27
60	129,3	86,27	4,77	0,85
85	198,3	111,27	6,87	2,51
			Overall Temperature Dependent Factor Π_T	1,31
			Effective failure rate for the overall hardware component (FIT)	104,65

4.6.3.1.2.3 Failure rate calculation for the semiconductor component example with non-operating phase

There is a difference between IEC TR 62380 and SN 29500 in the way the non-operating phases are considered. In IEC TR 62380 the non-operating hours are by default included in the mission profile of the product whereas in SN 29500 only the operating hours are by default considered. As described in 4.6.3.1.1.2, an alternative approach for calculating failure rate with IEC TR 62380 is setting τ_{off} time to zero.

In a similar way, operating and non-operating phases can also be taken into account in SN 29500 for the calculation of the failure rate. This is done by applying a stress factor π_ω described in SN 29500-2, Clause 4.4. Using the motor control mission profile as defined in IEC TR 62380 and an average temperature of 10,5C° gives a stress factor value of 0,06. Applying the calculated stress factor to the digital component example failure rate gives the results of Table 8.

Table 8 — SN 29500 failure rate calculation with or without non-operating phases

N (transistors)	Technology and Family	λ_{ref} (FIT)	λ Without non-operating phase (FIT)	Stress Factor	λ With non-operating phase (FIT)
986432 (Digital + SRAM)	CMOS, microprocessor	80	104,65	0,06	6,28

NOTE The non-operating average temperature is obtained from the average worldwide night and day-light temperatures (respectively 5 °C and 15 °C) as defined in IEC TR 62380 and considering a 50 % ratio between night and day.

4.6.3.1.2.4 Method to split SN 29500 overall failure rate into die and package failure rates

As stated by the maintainer of SN 29500, the base failure rate value calculated with SN 29500 is valid for the whole hardware component only and does not provide a method to split between package failure rate and die failure rate. An estimation of the split of package and die failure rates from an SN 29500 base failure rate could be calculated by using the same ratio of other industry sources which provide such data or from field data statistics when available. The IEC TR 62380 and FIDES Guide are possible industry sources for this data.

4.6.3.1.3 FIDES Guide

The following is an example of the estimation of hardware failure rate as needed to support quantitative analysis using the methods detailed in the FIDES guide [9]. The failure rate model for a semiconductor per FIDES guide considers the failure rate of the device to be a factor of:

— Physical contributions ($\lambda_{Physical}$);

— Process contributions ($\Pi_{Process}$); and

— Part Manufacturing contributions (Π_{PM}).

The first is an additive construction term comprising physical and technological contributing factors to reliability. The second is a multiplicative term including the quality and technical control over the development, manufacturing and the usage process for the product containing the device. The third factor represents for example the quality of the manufacturing site and the experience of the supplier. $\Pi_{Process}$ and Π_{PM} are set to 1 as these factors are related to systematic issues.

The physical contribution is composed of stresses acceleration factors due to usage conditions and an induced (i.e. unexpected overstress) multiplicative term inherent to the application of the product containing the device.

The models used in the FIDES guide for integrated circuits include the following physical stress families:

— Thermal;

— temperature cycling;

— mechanical; and

— humidity.

To compute the digital component die and package base failure rates (i.e. before application of de-rating for operating conditions), it is necessary to consider the following elements:

— λ_{0TH}, the basic failure rate associated with the type of device and process technology; and

— physical stress parameters a and b associated with the type of package.

Those factors are combined using FIDES. Selection of parameters can be done based on the process technology, type of circuitry and package utilised by the design. Values are available related to Microprocessor, Microcontroller, DSP and SRAM, and PQFP package with 144 pins.

Table 9 and

Table 10 below show the computation of the failure rates used in the quantitative example of a CMOS technology based MCU which consumes 0.5W power. The digital component die is packaged in a 144 pin quad flat package and cooled by natural convection and low-conductivity board.

Table 9 — Base failure rate of the die from UTE FIDES

Circuit element	**$\lambda 0_{TH}$ (FIT)**
50 k gate CPU	0,075
16 kB SRAM	0,055
Sum	0,13

Table 10 — Base failure rate of the package from UTE FIDES

Package	**λ0TCy_Case**			**λ0TCy_Solderjoints**		
	a	**b**	**λ0TCy_Case (FIT)**	**a**	**b**	**λ0TCy_Solcerjoints (FIT)**
144 pin PQFP	12,41	1,46	0,0058	10,80	1,46	0,029

Once the base failure rate for the digital component die and package has been generated, a de-rating factor is applied based on thermal effects and operating time. The de-rating factor takes into account:

— Junction temperature of the digital component die, which is calculated based on:

  — power consumption of the digital component die; and

  — package thermal resistance, based on package type, number of package pins and airflow.

— An application profile which defines 1 to Y usage phases, each of which is composed of an application “on-time” , “cycle time”, “cycle delta temperature”, and “cycle max temperature”, and “ambient temperature”.

NOTE The profile for use in the model considers more/other parameters than those provided in the profile of [41].

At first, the simplified mission profile example shown in Table 11 is considered.

Table 11 — Simplified mission profile example

			Thermal	Thermal cycling			
PHASE	**On / Off**	**tannual-phase (hours)**	**Tambient (°C)**	**Delta T cycling (°C)**	**Teta cy (hours)**	**N cy-annual (hours)**	**T max-cycling (°C)**
non-operational day	Off	720	15	10	24,0	30	20
night start	On	168	60	55	0,25	670	60
day start	On	335	60	45	0,25	1340	60
off - operational day	Off	7,538	15	10	22,5	30	20

The die base failure rate with de-rating factors is computed as in Table 12.

Table 12 — Die base failure rate with temperature de-rating factor

Circuit element	λ0TH (FIT)	Derating for temperature	Effective failure rate (FIT)
50k gate CPU	0,075	5,79	0,43
16kB SRAM	0,055	5,79	0,32
Sum	0,13		0,75

For evaluating these de-rating factors, the junction temperature, i.e. ΔTj due to self-heating is calculated as 18K, using the parameters and formula described in FIDES (see Table 13).

Table 13— Package base failure rate with temperature cycling de-rating factor

	λTCy_case			λTCy_solderjoints		
Package	**λ0TCy_case (FIT)**	**Derating for cycling**	**Effective failure rate (FIT)**	**λ0TCy_solderjoints (FIT)**	**Derating for cycling**	**Effective failure rate (FIT)**
144 pin PQFP	0,0058	130	0,75	0,029	10	0,28

Then, the elaborated mission profile example shown in Table 14 is considered.

Table 14 — Elaborated mission profile example

			Thermal	Thermal cycling			
PHASE	**On / Off**	**tannual-phase (hours)**	**Tambient (°C)**	**Delta T cycling (°C)**	**Teta cy (hours)**	**N cy-annual (hours)**	**T max-cycling (°C)**
non-operational day	Off	720	14	10	24,0	30	19
night start	On	117	32	22	0,0	670	32
day start	On	58	32	18	0,0	1340	32
full load operation	On	201	85	53	1,0	335	85
highway operation	On	131	60	28	4,0	30	60
off - operational day	Off	7,532	14	10	23,0	30	19

The de-rating factors are listed in Table 15.

Table 15 — Effective failure rate

Circuit element	λ0TH (FIT)	Derating for temperature	Effective failure rate (FIT)
50k gate CPU	0,075	12,44	0,93
16kB SRAM	0,055	12,44	0,68
Sum (FIT)	0,13		1,61

For evaluating these de-rating factors, the junction temperature, i.e. Δ_{Tj}, due to self-heating is calculated as 18K, using the parameters and formula described in FIDES.

As shown in the Table 16, the component package failure rate is then 0,25 FIT. The solder joints failure rate value in Table 16 is given as information only and is not considered as part of the package failure rate.

Table 16 — Package and solder joints failure rate

	λTCy_case (FIT)			λTCy_solderjoints (FIT)		
Package	**λ0TCy_case (FIT)**	**Derating for cycling**	**Effective failure rate (FIT)**	**λ0TCy_solderjoints (FIT)**	**Derating for cycling**	**Effective failure rate (FIT)**
144 pin PQFP	0,0058	42	0,25	0,029	4	0,12

4.6.3.2 Permanent base failure rate calculation using field data statistics

It is important to use field data statistics with care, as it is very difficult to get an appropriate estimation. A thorough analysis of the field return process is performed and the result of the analysis is used for the quantitative evaluations. In particular the following topics are evaluated:

— How does the field return process handle known quality issues?

— What kind of information is available about the real mission profile?

— What is the effectiveness of the field monitoring process?

Because the methodology used to calculate the failure rate from field data has an influence on the confidence level of the resulting failure rate value, the following points are taken into account by the semiconductor suppliers:

— It is important to put in place a proper field data collection system as given in ISO 26262-2:2018, 7.4.2.4;

— The goal of the method used is not to approximate as close as possible the real failure rate, but to provide a failure rate value for which there is a high confidence that it is above the real failure rate value;

— Significant source of systematic faults are only removed from the field statistics if the source of the systematic faults has been mitigated;

EXAMPLE An example of a major source of systematic faults is EOS.

NOTE Evidence of mitigation of the source of the systematic fault is documented.

— Because the semiconductor suppliers may not be aware of all failures in the field, a correction factor can be applied to the total number of returns. The correction factor can depend on many parameters such as the application and the device population used to estimate the field based failure rate;

NOTE Rationales are provided by semiconductor suppliers, who claim failure rate based on field returns.

— An acceleration factor AF corresponding to the temperature stress or to the thermal cycling stress effects can be respectively calculated using Arrhenius model associated with a specified activation energy or Coffin-Manson equation;

— The total operating time of the products in the field can be estimated using the mission profiles of the products when available. The variability in car usage from the drivers can also be taken into account by estimating the quantity of hours spent in field using for example a mean of 500 hours a year with a standard deviation of 145 hours; and

— The mission profile of the field data is documented and considered appropriately in the quantitative evaluations.

4.6.3.2.1 Exponential model method

The exponential model can be used in general to determine a constant failure rate from field returns.. In this model, χ^2 (chi-square) statistical function gives a good approximation of the failure rate. It is proposed to use for example an interval estimator with a one-sided upper interval estimation at 70 % confidence level instead of using a point estimator for the failure rate. That means that with 70 % probability, the real value of the failure rate is below that value. The failure rate can be calculated using the formula below:

$$FIT = \frac{\chi^2_{CL;2n+2} \times 10^9}{2 \times \text{cumulative operational hours} \times \text{AF}}$$

Where:

— n = Number of failures multiplied by the correction factor (CF);

— CL = Confidence level value (typically 70 %); and

— AF = Acceleration factor.

NOTE The acceleration factor is used to adapt failure rate values from one mission profile to another one as described in 4.6.3.2.2.

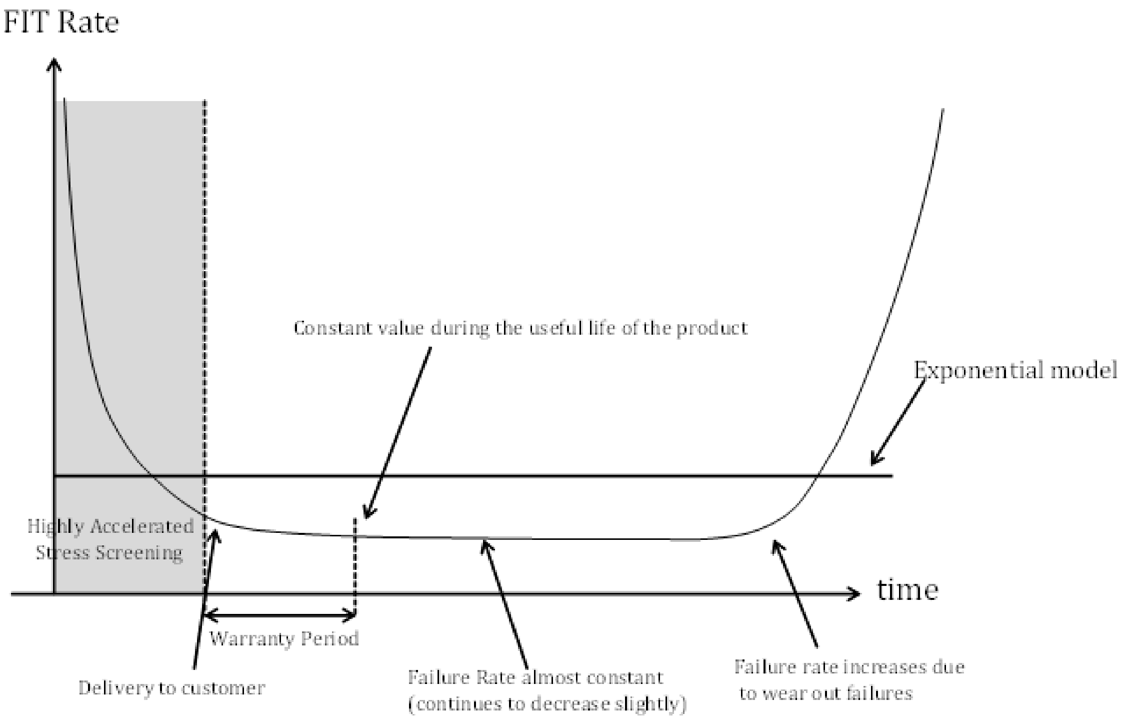

FIT Rate
Constant value during the useful life of the product
Exponential model
Highly Accelerated Stress Screening
time
Warranty Period
Delivery to customer
Failure Rate almost constant (continues to decrease slightly)
Failure rate increases due to wear out failures

Figure 8 — Bathtub curve – evolution of failure rate over time

NOTE In the Figure 8, the real bath tub curve can be approximated by the 'Constant value during the useful life of the product' or calculated by the exponential model with the confidence level of 70 %.

4.6.3.2.2 Calculation example of hardware component failure rate

In this clause an example of a die failure rate calculation using field data statistics is given using the exponential model method. The numbers used are arbitrarily chosen and have to be replaced by real data.

In this example we assume that the semiconductor supplier is collecting statistics from three products in the field as described in Table 17 below:

Table 17 — Mission profile and equivalent junction temperature $T_{j,eq}$

Tj(°C)	Chip 1 Phase Duration (hours)	Tj(°C)	Chip 2 Phase Duration (hours)	Tj(°C)	Chip 3 Phase Duration (hours)
-20	1000	-25	100	-20	500
10	2000	10	500	15	800
30	1500	35	10000	45	6000
45	6000	55	8000	80	4200
70 °C	1000	90 °C	1000	100 °C	600
100 °C	1300	100 °C	200	120 °C	300
130 °C	200	120 °C	200	150 °C	100
Mission profile Equiv. Temp $T_{j,eq}$	**55,1 °C**	**Mission profile Equiv. Temp $T_{j,eq}$**	**51,4 °C**	**Mission profile Equiv. Temp $T_{j,eq}$**	**67,4 °C**
Total duration	13000	**Total duration**	20000	**Total duration**	12500

NOTE 1 The mission profile equivalent temperature $T_{j,eq}$ corresponds to the temperature that would have the same effect as the whole mission profile from a temperature stress perspective. $T_{j,eq}$ can be calculated using the Arrhenius equation. In the above example an activation energy E_a of 0,3 eV was assumed.

NOTE 2 The device operating hours of the different devices can be summed up together if they are referred to the same reference temperature T_{ref}. In this example T_{ref} is 55 °C and the equivalent devices hours at T_{ref} are calculated using Arrhenius equation associated with an activation energy E_a of 0,3 eV.

NOTE 3 As shown in Table 18, the failure rate per mm^2 value at the reference temperature T_{ref} is calculated using the χ^2 statistical function from the total number of failures and the total number of die area hours. In this example an upper confidence level of 70 % has been used.

Table 18 — Calculation of failure rate per mm^2 at reference temperature Tref

Product Name	**Die size** mm^2	**Mission profile equivalent temp $T_{j,eq}$** (°C)	**Total Device Operating hours (in million device hours)**	**Arrhenius Acceleration Factor**	**Equivalent Operating hours at a T_{ref} of 55 °C (in million device hours)**	**Equivalent die area hours at a T_{ref} of 55 °C (in million mm^2 hours)**	**Nb of failures during warranty period**	**Nb of Failures with a Correction factor of 5**
Chip1	30	55,1	7000	1,00	7022,67	210680	1	5
Chip2	25	51,4	10200	0,89	9066,96	226674	1	5
Chip3	50	67,4	5000	1,47	7359,25	367963	2	10
Total die area hours						805317	**Total number of failures**	20
FIT/mm^2 at T_{ref} of 55 °C						0,029		

As explained in Figure 9 below, the failure rate per mm^2 at T_{ref} derived from the field data statistics can then be used to calculate the failure rate of the target product under design (see Table 19).

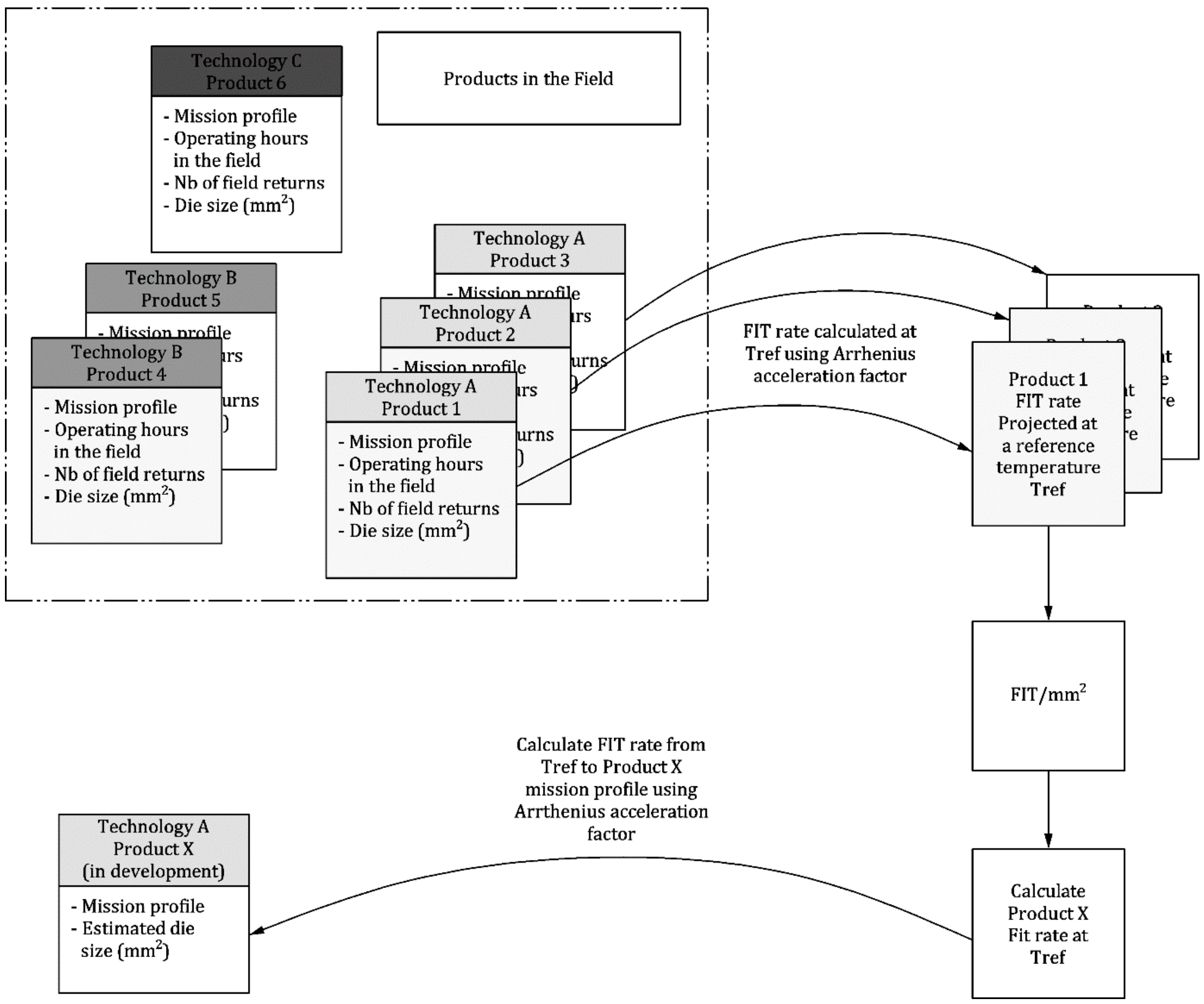

Products in the Field
Technology C Product 6
- Mission profile
- Operating hours in the field
- Nb of field returns
- Die size (mm²)
Technology B Product 5
Technology B Product 4
- Mission profile
- Operating hours in the field
- Nb of field returns
- Die size (mm²)
Technology A Product 3
Technology A Product 2
Technology A Product 1
- Mission profile
- Operating hours in the field
- Nb of field returns
- Die size (mm²)
FIT rate calculated at Tref using Arrhenius acceleration factor
Product 1 FIT rate Projected at a reference temperature Tref
FIT/mm²
Calculate FIT rate from Tref to Product X mission profile using Arrthenius acceleration factor
Calculate Product X Fit rate at Tref
Technology A Product X (in development)
- Mission profile
- Estimated die size (mm²)

Figure 9 — Die failure rate calculation method using field data statistics

Table 19 — Final chip failure rate calculation

	Mission profile Equiv. Temp Tj,eq (°C)	**Die size (mm^2)**	**FIT/mm^2 at Tref**	**Arrhenius Acceleration Factor**	**FIT/mm^2 at Equiv. Temp Tj,eq**	**Die base failure rate (FIT)**
Target Chip under design	75	23	0,029	1,84	0,053	1,22

NOTE 4 Same method is applied to calculate package failure rate but the acceleration factor is calculated using Coffin-Manson or Norris-Landzberg model (as discussed in [15] subclause 5.2.7.10 "Failure Modes", [16] clause 5.14 and [9] Clause 2.5.1 "Physics of failures and models") instead of Arrhenius model. Figure 10 gives an overview of the methods used to calculate the package failure rate using field data statistics.

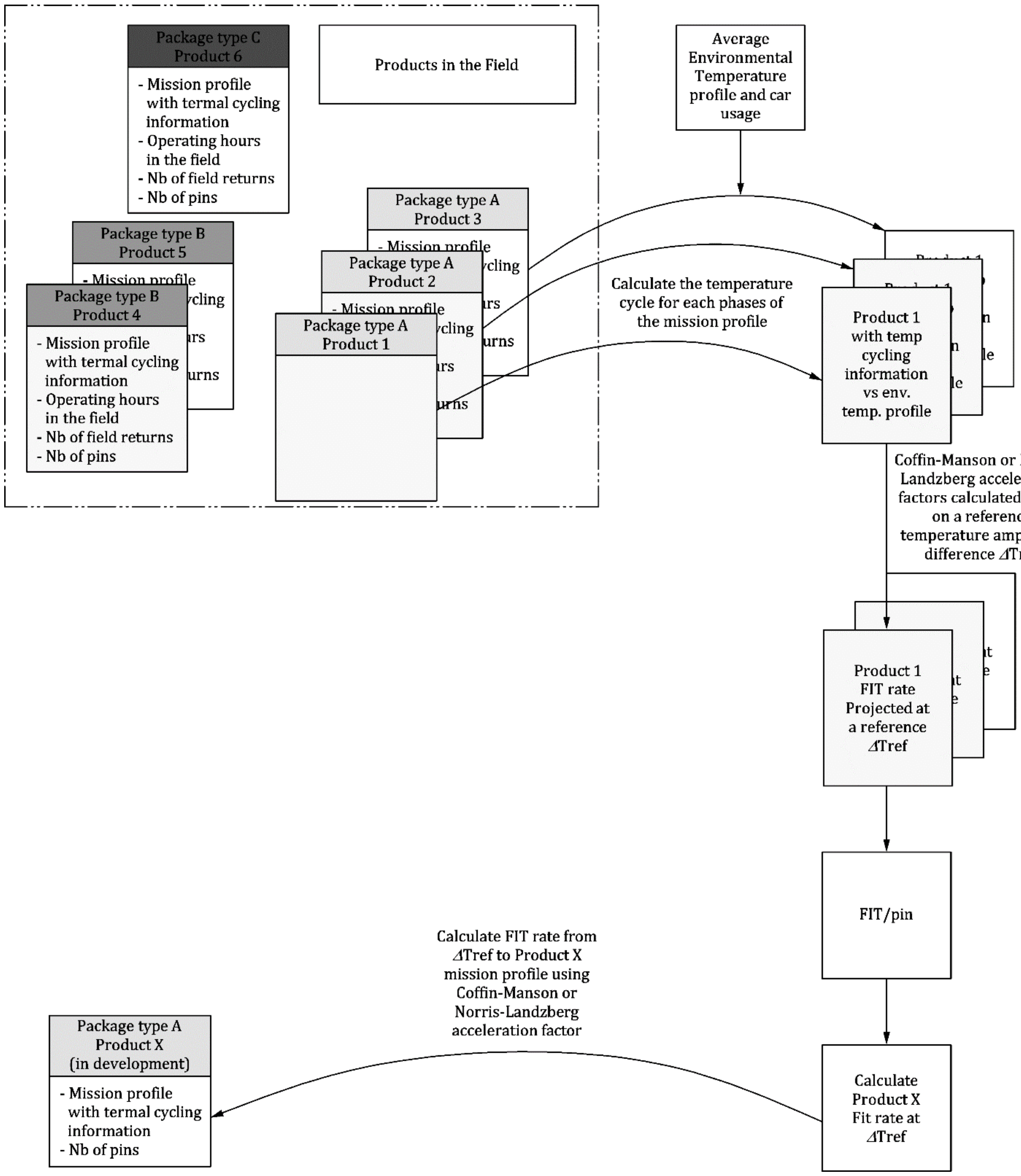

Products in the Field
Package type C Product 6
- Mission profile with termal cycling information
- Operating hours in the field
- Nb of field returns
- Nb of pins
Package type B Product 5
Package type B Product 4
- Mission profile with termal cycling information
- Operating hours in the field
- Nb of field returns
- Nb of pins
Package type A Product 3
Package type A Product 2
Package type A Product 1
Average Environmental Temperature profile and car usage
Calculate the temperature cycle for each phases of the mission profile
Product 1 with temp cycling information vs env. temp. profile
Coffin-Manson or Norris-Landzberg acceleration factors calculated based on a reference temperature amplitude difference ΔTref
Product 1 FIT rate Projected at a reference ΔTref
FIT/pin
Calculate Product X Fit rate at ΔTref
Calculate FIT rate from ΔTref to Product X mission profile using Coffin-Manson or Norris-Landzberg acceleration factor
Package type A Product X (in development)
- Mission profile with termal cycling information
- Nb of pins

Figure 10 — Package failure rate calculation method using field data statistics

NOTE 5 In case no distinction is done in the field data analysis between die and package (as it is the case for example in SN 29500 [39]) then Arrhenius law can be used to calculate the hardware component (die and package) failure rate using the mission profile temperatures and reference temperature T_{ref} as depicted in Figure 10.

4.6.3.3 Base failure rate calculation using accelerated life tests

To de-rate from the temperature at which the life test is carried out to the maximum operating temperature an acceleration factor is applied. This calculation uses Arrhenius equation with activation energy of 0,7 eV. It is recommended to estimate and verify activation energy associated with desired failure mechanism

The number of faults obtained from the sample is used in the χ^2 distribution function with a certain confidence level to obtain the number of faults that would occur over the entire population tested.

Voltage acceleration is also taken into account when determining the life of devices. This is calculated by taking the oxide thickness into consideration and de-rating from the stress test voltage to the life operating voltage.

$$AF_v = \exp{(\beta)*[V_t - V_o]}$$

Where

— AF_v = Voltage acceleration factor;

— V_o = Gate voltage under typical operating conditions (in Volts);

— V_t = Gate voltage under accelerated test conditions (in Volts);

— β = Voltage acceleration coefficient (in 1/Volts).

4.6.3.4 Failure rate distribution methods

The previous sub-clauses detail several methods to determine the base failure rate for the semiconductor component. Depending on the methods, the overall semiconductor component failure rate can be available as a single value or combination of package failure rate and die failure rate. During the safety analysis the semiconductor component failure rate is allocated to the failure modes of elements composing the semiconductor component.

Different distribution methods can be applied:

— Failure rate distribution to the die part: failure rate for internal elements of the component (like for example digital blocks, analogues blocks and memories): two methods can be considered to perform the distribution:

  — The first method consists of using a failure rate per mm^2 value obtained by dividing the die failure rate or the whole hardware component failure rate (if not separated into package and die contributions) by the die area of the hardware component. The failure rate distribution is done by multiplying the part or sub-part area related to the failure mode under analysis by the failure rate per mm^2 value; and

  — The second method is based on base failure rates and elementary sub-parts using the calculation methods described above applied to these elements. This is done by making an estimation of the number of equivalent gates (or number of transistors) for each part, sub-part or basic/elementary sub-part related to the failure mode under analysis.

— Failure rate distribution to the package: This can be derived only when the failure rate of package is available. In such a case, for pins that are safety-related, the distribution of the failure rate can be done using a failure rate per pin value which is obtained by dividing the package failure rate by the total number of pins of the package (safety-related or not).

NOTE The selection of the method used can be based on the layout (or planned layout) of the circuit under analysis or on the analysis of how failure modes are shared between the hardware elements.

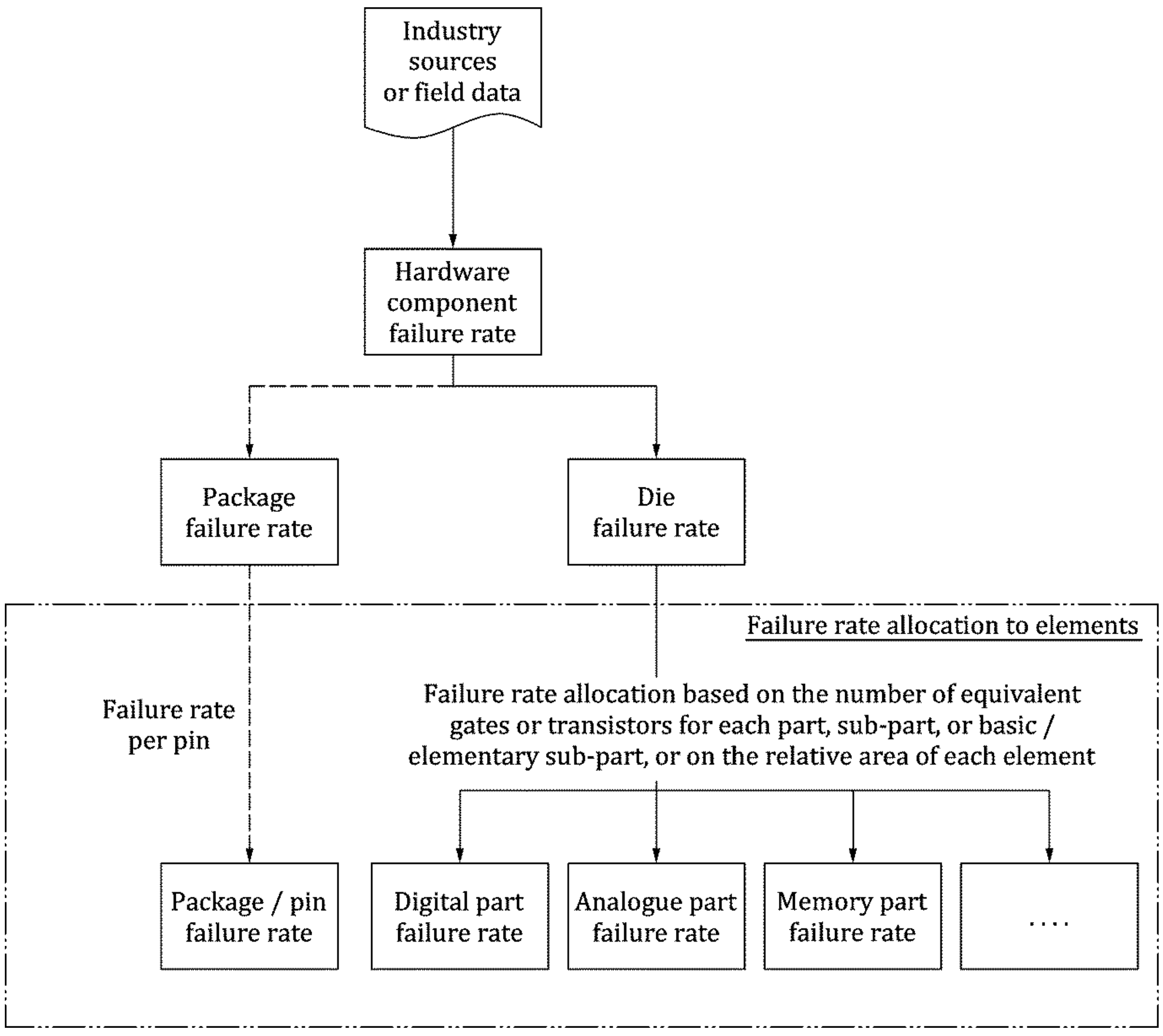

Industry sources or field data
Hardware component failure rate
Package failure rate
Die failure rate
Failure rate allocation to elements
Failure rate per pin
Failure rate allocation based on the number of equivalent gates or transistors for each part, sub-part, or basic / elementary sub-part, or on the relative area of each element
Package / pin failure rate
Digital part failure rate
Analogue part failure rate
Memory part failure rate
....

Figure 11 — Failure rate distribution

4.6.3.5 Base Failure Rate for MCM

The base failure rate for Multi Chip Modules (MCM) is carefully evaluated. In case industry sources (such as IEC 62380) are used to estimate that failure rate, arguments are provided to justify the applicability or the customisation of that industry source.

4.7 Semiconductor dependent failures analysis

4.7.1 Introduction to DFA

The goal of this clause is to provide guidelines for the identification and analysis of possible common cause and cascading failures between given elements, the assessment of their risk of violating a safety goal (or derived safety requirements) and the definition of safety measures to mitigate such risk if necessary. This is done to evaluate potential safety concept weaknesses and to provide evidence of the fulfilment of requirements concerning independence or freedom from interference identified during coexistence analysis (see ISO 26262-9:2018, Clause 6).

The scope of this clause is the Dependent Failures Analysis (DFA) between hardware elements implemented within one silicon die and between hardware and software elements. The elements under consideration are typically hardware-elements and their safety mechanisms (specified during the activities of ISO 26262-5:2018).

The scope, analysis method(s) and the necessary safety measures can depend on the nature of the given elements (e.g. just software elements, just hardware elements or a mix of hardware and software elements) and the nature of the involved safety requirements (e.g. fail safe).

The Dependent Failures Initiator (DFI) represents the root cause of dependent failures in safety scope. A list of DFI is provided as a starting point, considering different systematic, environmental and random hardware issues (Table 20 from Table 25). Some random hardware DFI, e.g. shared resources or interfering elements[1)] of the elements under consideration, can be considered within the standard safety analysis once the dependencies are identified and can be classified as either residual faults, single-point faults or multiple-point faults (ISO 26262-5:2018, 9.4.2.4 NOTE 1). The DFA addresses those DFI, which are not addressable within the standard safety analysis, in a qualitative way.

The list of DFI also contains some typical safety measures used to address these. The necessary safety measures can depend on the nature of the safety requirement, in particular if the risk of occurrence of the dependent failures in the field is to be minimized or if it is sufficient that if the dependent failures occurs the safety goal is not violated.

The requirements that aim at controlling dependent failures need to precisely identify in which manner the control measure is intended to operate:

— In the case of a fail-safe requirement of the given elements it is not necessary to avoid the occurrence of the dependent failure. It is sufficient to detect it and switch the element into a safe state, e.g. by deactivation of safety-related outputs or by reporting the error to another element that can take measures to bring the system or element into a safe state; and

— In the case of a fail-operational requirement, where deactivating the given elements can be not acceptable and no safe state can be defined that does not require at least an operation with degraded performance , safety measures can be necessary which reduce the probability of the dependent failures occurring in the field.

4.7.2 Relationship between DFA and safety analysis

The correlated elements for which a DFA is relevant, can already be identified from the safety analyses done in accordance to ISO 26262-5:2018, 7.4.3. These can be dual-point failure scenarios like:

— Functions and their safety mechanisms (including the fault reaction path - the chain of elements and/or tasks that are required to implement the fault reaction); and

— Functional redundancies (e.g. two current drivers or two A/D converters).

And single-point (residual) failure scenarios of shared elements that belong to the semiconductor infrastructure like:

— Clock generation;

[1)] Interfering elements have the capability to corrupt resources of other hardware elements as a consequence of a random hardware fault or systematic fault: e.g. a DMA (direct memory access peripheral) writes to a wrong address and silently corrupts safety-related data.

— Embedded voltage regulators; and

— Any shared hardware resource used by the aforementioned correlated elements.

While the safety analysis primarily focuses on identifying single-point faults and dual/multiple-point faults to evaluate the targets for the ISO 26262 metrics and define safety mechanisms to improve the metrics if required, the DFA complements the analysis by ensuring that the effectiveness of the safety mechanisms is not affected by dependent failures initiators. As mentioned in ISO 26262-5:2018, 7.4.3, the safety analysis can be used in a first place to support the specification of the hardware design and subsequently can be used for the verification of the hardware design. Similarly the DFA can be applied as well during the specification of the hardware design (e.g. to specify safety mechanisms for the shared elements that have been identified) and in the second stage to verify that the assumption taken during the specification are realized and reach intended effectiveness.

4.7.3 Dependent failures scenarios

In Figure 12, Element A and Element B are correlated elements that have the potential to fail under the presence of an external root cause. The root cause can be related to a random hardware fault or to a systematic fault.

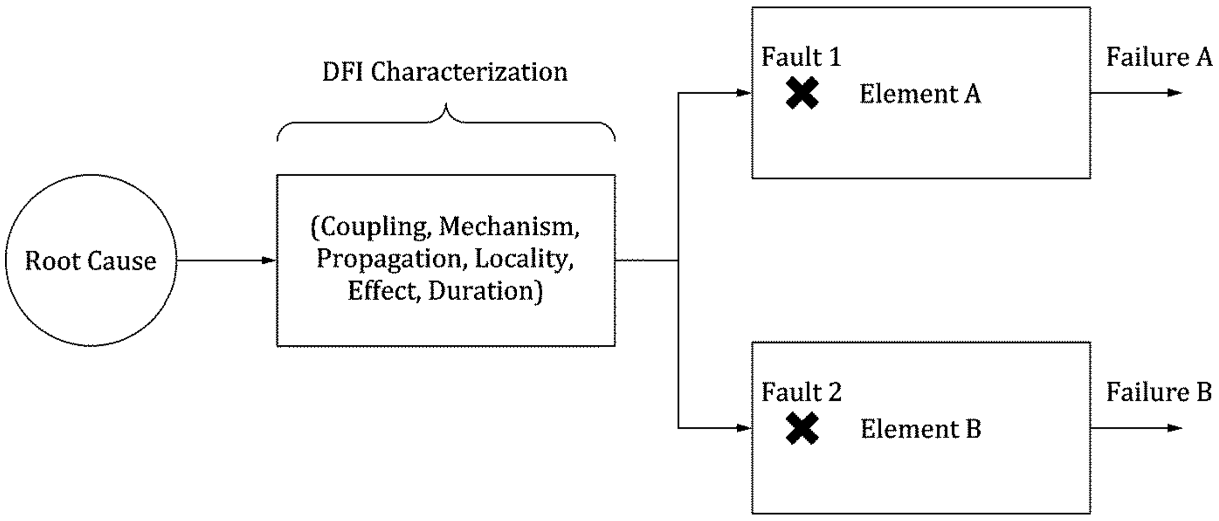


Figure 12 — Schematic representation of a dependent failures and its DFI

Typical situations related to a random hardware fault can include failure of shared resources or single physical root cause. For these situations a failure rate could be quantified and could be considered into the safety analysis according to ISO 26262-5:2018.

Typical situations related to systematic faults can include environmental faults, development faults, etc.. For these situations it is in general not possible to make a quantitative analysis. Additionally the root cause can be located inside the semiconductor element under consideration or located outside and propagates into the semiconductor element through signal or power supply interfaces for instance.

Figure 13 refers to coupling mechanism that aims at characterizing some exemplary properties of the disturbances created by a given root cause. Such properties can help to specify the mitigation measures and as well to define the adequate models that can be used to verify the effectiveness of the mitigation measures (see 4.7.5.2). They are now introduced:

- Coupling mechanism: this property characterizes the means by which a root cause induces a disturbance. Known coupling mechanisms are: Conductive coupling occurs when the coupling path between the source and the receptor is formed by direct contact with a conducting body, for example a transmission line, wire, cable, PCB trace or metal enclosure; and
  - Near field coupling occurs where the source and receiver are separated by a short distance (typically less than a wavelength). Strictly, "Near field coupling" can be of two kinds, electrical induction and magnetic induction. It is common to refer to electrical induction as capacitive coupling, and to magnetic induction as inductive coupling;
    - Capacitive coupling occurs when a varying electrical field exists between two adjacent conductors typically less than a wavelength apart, inducing a change in voltage across the gap; and
    - Inductive coupling or magnetic coupling occurs when a varying magnetic field exists between two conductors in close proximity, typically less than a wavelength apart, inducing a change in voltage along the receiving conductor.
  - Radiative coupling or electromagnetic coupling occurs when source and receiver are separated by a large distance, typically more than a wavelength. Source and receiver act as radio antennas: the source emits or radiates an electromagnetic wave which propagates across the open space in between and is picked up or received by the receiver.
- Propagation medium: this property characterizes the coupling path the disturbance uses through the semiconductor element. Typically it can be:
  - Signal lines;
  - Clock network;
  - Power supply network;
  - Substrate;
  - Package; and
  - Air.
- Locality: this property characterizes if the disturbance has the potential to affect multiple elements or is limited to a single element. In the latter case the affected element is assumed to produce a wrong output that propagates to multiple elements connected to it (cascading effect);
- End effect: this property characterizes in which manner the hardware is affected by the disturbance. Possible examples are:
  - Timing violation (e.g. caused by crosstalk, timing fault, etc.); and
  - Incorrect logical behaviour (e.g. caused by latch-up, etc.).
- Timing: this property characterizes some properties of the disturbance related to its propagation delay (e.g. for propagation of temperature gradient) or its timing behaviour like periodicity (e.g. in the case of ripple noise over power supply), etc.

In order to illustrate the aforementioned properties two examples are given in Figure 13 and Figure 14.

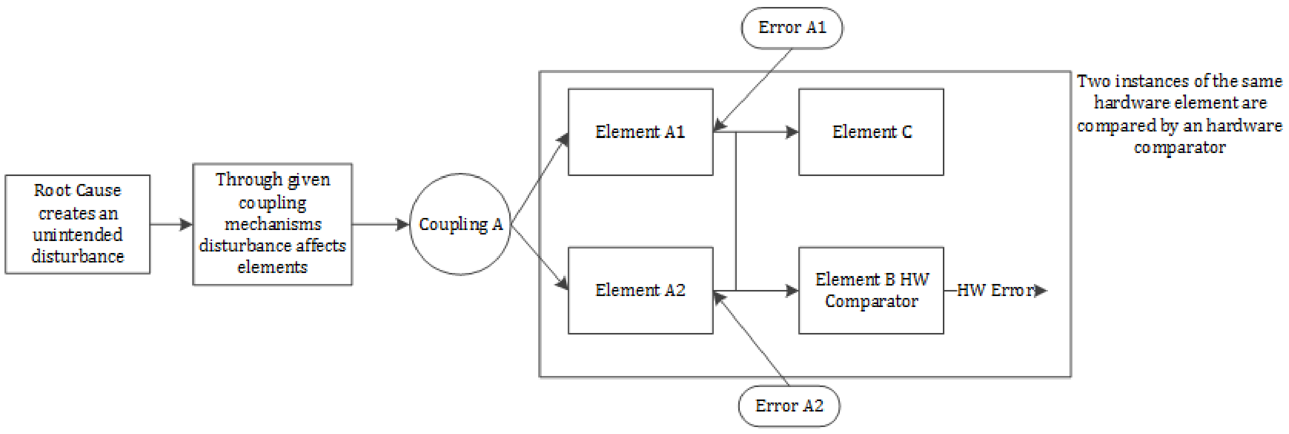

Error A1
Two instances of the same hardware element are compared by an hardware comparator
Root Cause creates an unintended disturbance
Through given coupling mechanisms disturbance affects elements
Coupling A
Element A1
Element C
Element A2
Element B HW Comparator
HW Error
Error A2

Figure 13 — Dependent failures by physical coupling

In Figure 13 Element A1 provides the outcomes used by Element C for implementing a safety function. Element A1 and Element A2 are used as redundant elements compared by Element B hardware Comparator and in the case of mismatch (Failure A1 or Failure A2), the “hardware error” signal is activated. In this example, the Element A1 and Element A2 can produce identical erroneous outputs (Error A1 and Error A2) if both elements are affected by a fault that results from a same root cause. The presence of this possible dependent failures cannot be differentiated by Element B at the time they are compared.

NOTE It is assumed for simplification that Element B itself is not affected by the disturbance. Taking into account the assumption that Element B is operational it is further assumed that as long as Error A1 and Error A2 present some temporal or spatial dissimilarity, the dependent failures situation can be controlled. Such dissimilarity can be the consequence of differences in the manner the disturbance propagates to both elements (e.g. different propagation delay of a signal glitch that takes different physical routes to reach boundaries of Element A1 and Element A2) or in differences in the effect (e.g. if the effect is a signal timing violation, it can have different effect on the respective logic of Element A and Element B).

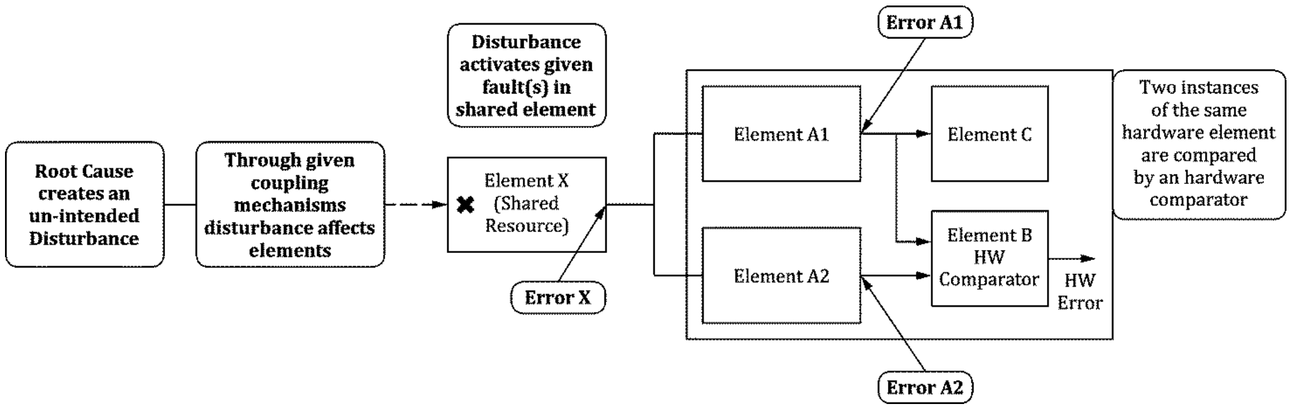

Disturbance activates given fault(s) in shared element
Error A1
Two instances of the same hardware element are compared by an hardware comparator
Root Cause creates an un-intended Disturbance
Through given coupling mechanisms disturbance affects elements
Element X (Shared Resource)
Error X
Element A1
Element C
Element B HW Comparator
HW Error
Element A2
Error A2

Figure 14 — Dependent failures due to resource sharing

Figure 14 extends Figure 13 where Element A1 and Element A2 produce erroneous outputs caused by an erroneous output of the shared Element X that is affected by a fault that results from a root cause external to the element itself. The erroneous output of Element X propagates to both Element A1 and Element A2. Element X is representative of the dependent failures initiators that fall into the category “Shared Resources”.

4.7.4 Distinction between cascading failures and common cause failures

Dependent failures analysis addresses both common cause failures and cascading failures. While in some cases this differentiation is necessary (such as for ISO 26262-9:2018, Clause 7), in other cases (such as for semiconductor devices) the exact differentiation between a cascading failure and a common cause failure in a given failure scenario is not always possible or useful. In this case, the two failure scenarios are not differentiated any further.

If the focus of the DFA is to provide evidence of freedom from interference (coexistence) between two given elements (e.g. Element A and Element B) as requested in ISO 26262-9:2018, Clause 7, the following approach can be used:

— Identify the failure modes of Element A which can have an impact on Element B ;

— Identify if these failure modes lead to possible violation of the safety goal due to Element B failure;

— If necessary define appropriate safety measures to mitigate the risk (e.g. for a DMA specify a safety mechanism that monitors the addresses generated by the DMA); and

— If necessary repeat this analysis with switched roles.

4.7.5 Dependent failures initiators and mitigation measures

4.7.5.1 List of dependent failures initiator and related mitigation measures

The following classification of DFI is proposed:

— Failure of shared resources;

— Single physical root cause;

— Environmental faults;

— Development faults;

— Manufacturing faults;

— Installation faults; and

— Repair faults.

NOTE 1 Other classifications of DFI are possible.

For each class of dependent failures, possible measures are provided.

NOTE 2 The listed measures are examples provided as a non-exhaustive list of possible solutions. Their effectiveness depends on several factors including type of circuits and technology, and they could be not effective at the same way for all possible DFI. For that reason, it is recommended to provide evidences to demonstrate the claimed effectiveness. Some measures by themselves can be not enough to achieve an appropriate risk reduction. In this case an appropriate combination of different measures can be chosen.

The measures have been split into:

— measures which prevent the dependent failures occurring during operation; and

— measures which do not prevent the occurrence of the dependent failures but prevent it from violating a safety goal.

NOTE 3 DFI that are caused by software are not included in this DFI list. Correct software development is addressed by ISO 26262-6:2018. Results of the DFA can affect the ASIL allocation of software elements.

NOTE 4 Repair in automotive typically happens by exchange of the whole ECUs or sensor modules. Semiconductor components are typically not repaired. Therefore repair faults are usually not DFI for semiconductor parts.

Table 20 — Dependent failures initiators due to random hardware faults of shared resources

DFI examples	Measures to prevent dependent failures from violating the safety goal	Measures to prevent the occurrence of dependent failures during operation
Failures in common clock elements (including PLL, clock trees, clock enable signals, etc…) Failures in common test logic including DFT (Design for Test) signals, scan chains etc…, common debug logic including debug routing network (network that provides access to analogue or digital signals or allows to read digital registers) and trace signals (mechanism to trace one or more signals synchronously, e.g. controlled by triggers or trace clocks and read the result afterwards) Failures in power supply elements including power distribution network, common voltage regulators, common references (e.g. band-gaps, bias generators and related network) Non simultaneous supply switch-on, that can cause effects like latch up or high in-rush current Failures in common reset logic including reset signals Failures in shared modules (e.g. RAM, Flash, ADC, Timers, DMA, Interrupt Controller, Busses, etc…)	Dedicated independent monitoring of shared resources (e.g. clock monitoring, voltage monitoring, EDC-ECC for memories, CRC over configuration register content, signalling of test or debug mode) Selective hardening against soft-errors or selected redundancy Self-tests at start-up or post-run or during operation of the shared resources Diversification of impact (e.g. clock delay between master & checker core, diverse master and checker core, different critical paths) Indirect detection of failure of shared resource (e.g. cyclic self-test of a function that would fail in the case of a failure of the shared resource) Indirect monitoring using special sensors (e.g. delay lines used as common-cause failure sensors)	Fault avoidance measures (e.g. conservative specification), functional redundancies within shared resources (e.g. multiple via/contacts), Fault diagnosis (e.g. ability of identifying and isolating or reconfiguring/replacing failing shared resources, corresponding design rules) Dedicated production tests (e.g. end-of-line tests for SRAM capable to find complex faults) Separate resources to reduce the amount or scope of shared resources Adaptive measures to reduce susceptibility (e.g. voltage/operating frequency decrease)

Table 21 — Dependent failures initiators due to random physical root causes

DFI examples	Measures to prevent dependent failures from violating the safety goal	Measures to prevent the occurrence of dependent failures during operation
Short circuits due to e.g.: local defects, electro migration, via migration, contact migration, oxide break down Latch up Cross talk (substrate current, capacitive coupling) Local heating caused e.g. by defective voltage regulators or output drivers	Selective hardening against soft-errors or selected redundancy Diversification of impact (e.g. clock delay between master & checker core, diverse master and checker core, different critical paths) Indirect detection (e.g. cyclic self-test of a function that would fail in the case of physical root cause) or indirect monitoring using special sensors (e.g. delay lines used as common-cause failure sensors)	Dedicated production tests Fault avoidance measures (e.g. physical separation / isolation, corresponding design rules) Physical separation on a single chip

Table 22 — Systematic dependent failures initiators due to environmental conditions

DFI examples	Measures to prevent dependent failures from violating the safety goal	Measures to prevent the occurrence of dependent failures during operation
Temperature Vibration Pressure Humidity / Condensation Corrosion EMI Overvoltage applied from external Mechanical stress Wear Aging Water and other fluids intrusion	Diversification of impact (e.g. clock delay between master & checker core, diverse master and checker core, different critical paths) Direct monitoring of environmental conditions (e.g. temperature sensor) or indirect monitoring of environmental conditions (e.g. delay lines used as dependent -failure sensors)	Fault avoidance measures (e.g. conservative specification / robust design) Physical separation (e.g. distance of the die from a local heat source external of the die) Adaptive measures to reduce susceptibility (e.g. voltage/operating frequency decrease) Limit the access frequency or limit allowed operation cycles for sub-parts (e.g. specify the number of write cycles for an EEPROM) Robust design of semiconductor packaging

Table 23 — Systematic dependent failures initiators due to development faults

DFI examples	Measures to prevent dependent failures from violating the safety goal	Measures to prevent the occurrence of dependent failures during operation
Requirement faults Specification errors Implementation faults, i.e. incorrect implementation of functionality Lack or insufficiency of design measures to avoid crosstalk Lack or insufficiency of Latch up prevention measures Wrong configuration Layout faults, such as incorrect routing e.g. over redundant blocks, insufficient insulation, insufficient separation or isolation, insufficient EMI shielding Temperature due to heating of power consuming parts of the die Temperature gradients causing mismatches within sensitive measurement circuitry	Monitors (e.g. protocol checkers)	ISO 26262 compliant design process Diversity (Depending on the DFI, diversity can be intended either as implementation / functional / architectural diversity or as development diversity)

Table 24 — Systematic dependent failures initiators due to manufacturing faults

DFI examples	Measures to prevent dependent failures from violating the safety goal	Measures to prevent the occurrence of dependent failures during operation
Related to processes procedures and training Faults in control plans and in monitoring of special characteristics Related to software flashing and end-of-line programming (e.g. wrong versions or wrong programming conditions, protocols or timings) Mask misalignment Incorrect End-of-Line trimming or fusing (e.g. Laser trimming, OTP or EEPROM programming of calibration coefficients or customization settings)		Dedicated production tests ISO 26262 compliance (see 4.9) Diversity (Depending on the DFI, diversity can be intended either as implementation / functional / architectural diversity or as development diversity)

Table 25 — Systematic dependent failures initiators due to installation faults

DFI examples	Measures to prevent dependent failures from violating the safety goal	Measures to prevent the occurrence of dependent failures during operation
Related to wiring harness routing Related to the inter-changeability of parts Failures of adjacent items or parts or elements. (e.g. wrong configuration of a connected interface delivering data to an input, or incorrect load on a driven output) Wrong microcontroller PCB connection Wrong configuration (e.g. of spare memory usage)		Dedicated installation tests ISO 26262 compliance (see 4.9) Diversity (Depending on the DFI, diversity can be intended either as implementation / functional / architectural diversity or as development diversity)

4.7.5.2 Verification of mitigation measures

This clause introduces exemplary methods to evaluate the effectiveness to control or avoid dependent failures. The methods can be based on:

— Analytical approach using known principles;

EXAMPLE 1 Reference [4] and similar provide analytical approaches that can be used as a basis to evaluate the effectiveness of the provided safety mechanisms addressing dependent failures

— Pre-silicon simulation using documented test protocols to provide evidence of robustness against the identified DFI;

EXAMPLE 2 Test protocols that allow simulation of clock or power supply disturbances, EMI simulations etc. The simulation can be based on different levels of abstraction (based on the fault model to be targeted) and use adequate fault injection techniques to produce the intended disturbance.

— Post-silicon robustness tests (e.g. EMI test, burn In studies, accelerated aging test, electrical stress tests); and

— Expert judgment supported by documented rationale.

A combination of measures can be used, e.g. references [24], [21] and similar provide a mix of analytical, fault injection and expert judgment based approaches that can be used as a basis to evaluate the effectiveness of the provided safety mechanisms addressing dependen t failures.

NOTE 1 The results and the arguments are documented and justified.

NOTE 2 The use of beta factors as in IEC 61508-2:2010 [14] for the quantification of coupling effects is not foreseen as stated in ISO 26262-9:2018, 7.4.2.

The level of detail of the evaluation is commensurate with the type of DFI, the claimed safety measures and application.

As stated in the EXAMPLE in ISO 26262-9:2018, 7.4.7, diversity is a measure that can be used to prevent, reduce or detect common cause failures. In case diversity is used as a method to control or avoid dependent failures, a rationale is provided to demonstrate that the level of implemented diversit y is commensurate to the targeted DFI.

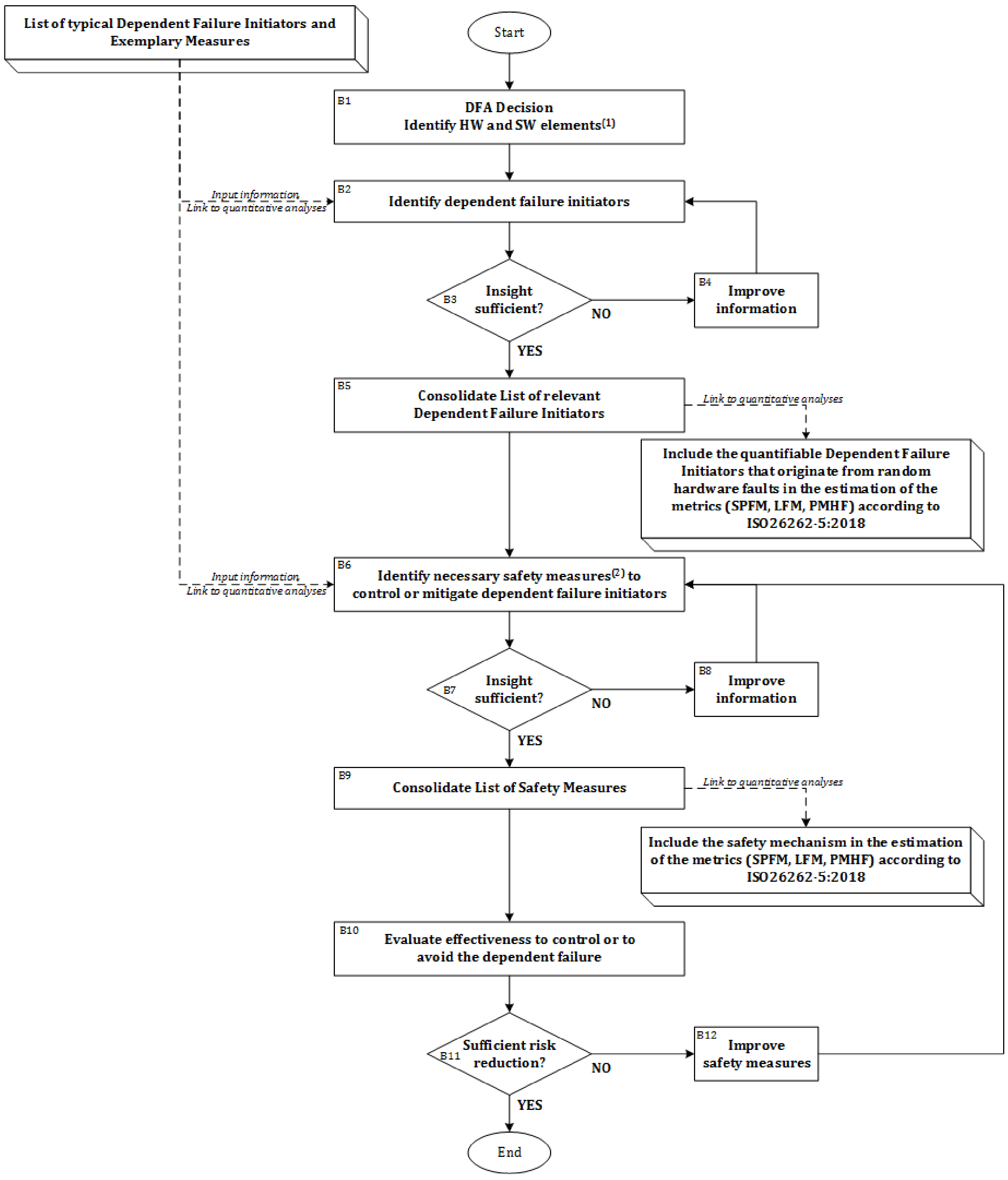
EXAMPLE 3 A rationale can be provided with a combination of analytical approach and fault injection (e.g. as described in [24]). For details on fault injection, see 4.8.

In case isolation or separation is used as a method to control or avoid dependent fail ures, a rationale is provided to demonstrate that the level of implemented isolation or separation is commensurate to the targeted DFI.

EXAMPLE 4 Simulation can be used to provide evidence of that the distance between two separated blocks is commensurate to avoid the targeted DFI.

4.7.6 DFA workflow

The purpose of the DFA workflow is to identify the main activities that are judged necessary to understand the operation of the safety mechanism s that are implemented to assure achievement of the safety requirements and verify that they comply with the requirements for independence or freedom from interference.


List of typical Dependent Failure Initiators and Exemplary Measures
Start
B1
DFA Decision
Identify HW and SW elements(1)
Input information
Link to quantitative analyses
B2
Identify dependent failure initiators
B3
Insight sufficient?
NO
B4
Improve information
YES
B5
Consolidate List of relevant Dependent Failure Initiators
Link to quantitative analyses
Include the quantifiable Dependent Failure Initiators that originate from random hardware faults in the estimation of the metrics (SPFM, LFM, PMHF) according to ISO26262-5:2018
Input information
Link to quantitative analyses
B6
Identify necessary safety measures(2) to control or mitigate dependent failure initiators
B7
Insight sufficient?
NO
B8
Improve information
YES
B9
Consolidate List of Safety Measures
Link to quantitative analyses
Include the safety mechanism in the estimation of the metrics (SPFM, LFM, PMHF) according to ISO26262-5:2018
B10
Evaluate effectiveness to control or to avoid the dependent failure
B11
Sufficient risk reduction?
NO
B12
Improve safety measures
YES
End

Notes:
(1) Firmware and any micro-code running on programmable HW elements, independently if they are classified as CPUs or not, can be considered to be SW elements.
(2) Safety measures (according to ISO 26262-1, 1.110) can be activities that show a failure is not relevant for the DFA.

Figure 15 — DFA workflow

4.7.6.1 DFA decision and identification of hardware and software elements (B1)

A DFA according to ISO 26262-9:2018, Clause 7 for a semiconductor element is conducted in any case that can require independence or freedom from interference e.g.:

— Diagnostic functions assigned to hardware or software elements;

— Similar or dissimilar redundancy of hardware or software elements;

— Shared resources on the hardware component or part (e.g. clock, reset, supply memory, ADC, I/O, test logic);

— Execution of multiple software tasks on shared hardware;

— Shared software functions (e.g. I/O-routines, interrupt handling, configuration, math library or other library functions); and

— Independence requirements derived from ASIL decomposition on system or element level that affect different elements on the IC, where the DFA needs to provide evidence of sufficient independence in the design or that the potential common causes lead to a safe state. (Refer to ISO 26262-9:2018, Clause 5).

The inputs to this step are.

— the technical safety requirements, in particular the independence and freedom from interference requirements resulting from the safety concept on system level;

— the description of the architecture, which can include block diagrams, flow charts, fault trees, state diagrams, hardware partitioning, software partitioning; and

— the safety mechanisms.

The focus of this step is to analyse the architecture and identify each pair or group of elements that can be affected by any of the above listed cases and evaluate if the architectural description is detailed enough to capture the overall design dependencies. The outcome of this step is a list of each pair or group of elements that can be affected by dependent failures and associated independence or freedom from interference requirements.

4.7.6.2 Identification of DFI (B2)

This step is based on the prior architectural analysis and it targets a check of the completeness of the derived independence or freedom from interference requirements and break them down wherever different initiators can lead to a dependent failure.

A list of typical DFI as provided in 4.7.5.1 can be used to prove whether known dependent failures other than the ones that were derived from the architecture can be applied. Further it is crosschecked if dependent failures mechanisms were identified during the quantitative analysis.

The outcome of this step is a consolidation of the list from the previous step.

4.7.6.3 Sufficiency of insight provided by the available information on the effect of identified DFI (B3 & B4)

In this step it is verified if the available documentation provides sufficient insight to each DFI that was evaluated during previous steps. In case any additional information is required to judge the validity of a DFI for the target architecture, it is added and the identification of the DFI (step 2) can be finished based on the updated descriptions.

NOTE A hierarchical approach is recommended so that the analysis can be done on an appropriate level of detail. For instance a top level view enables to understand what the shared resources are. Then a breakdown view that encapsulates a hardware sub-part and its safety mechanisms can be used to identify dependencies at the design level.

4.7.6.4 Consolidation of list of relevant DFI (B5)

Based on the provided consolidated information, the list of identified DFA relevant elements, independence requirements and the related DFI for the fulfilment of the safety requirements is consolidated (e.g. by review).

From the consolidated list, dependent failures that are caused by random hardware faults can be incorporated into the quantitative analysis of the required metrics in accordance with ISO 26262-5:2018 clauses 8 and 9.

4.7.6.5 Identification of necessary safety measures to control or mitigate DFI (B6)

In order to fulfil the safety requirements, necessary safety measures are added to mitigate the effect of the dependent failures that are relevant for the target architecture.

Examples of measures that are usually effective to mitigate DFI are given in the list of typical DFI in 4.7.5.1. Finally the required safety mechanisms have to be integrated into the documentation of the safety concept and the architecture to implement it.

NOTE 1 For dependent failures that arise from random hardware faults the result of the quantitative analysis can be used to identify the ones that are relevant to achieve the targeted metrics in accordance with ISO 26262-5:2018 clauses 8 and 9.

NOTE 2 If quantifiable random hardware failures are identified to be relevant as DFI (e.g. a shared oscillator delivering a clock that is too fast for the timing constraints of a digital core; overvoltage delivered to an internal supply due to a fault of a supply voltage regulator) they are taken into account for the quantitative analysis (see ISO 26262-5:2018, 9.4.3.2 NOTE 1). For the case that they are not quantifiable (e.g. the influence of timing effects caused by a fault in a clock tree; thermal coupling effects between an element and its safety mechanism; substrate currents due to a fault in one of the blocks that have to be independent) the evaluation and definition of mitigation measures is continued qualitatively (see ISO 26262-9:2018, 7.4.2).

4.7.6.6 Sufficiency of insight provided by the available information on the defined mitigation measures (B7 & B8)

In this step it is verified if the available documentation provides sufficient insight to analyse the effectiveness of safety measures that were introduced during the previous step. For the case that any additional information is required to judge the mitigation of a DFI for the target architecture including each safety mechanism, it is added and the definition of dependent failures mitigation measures is finished based on the updated descriptions.

4.7.6.7 Consolidate list of safety measures (B9)

The list of the defined safety measures for the mitigation of dependent failures is consolidated based on the updated documentation (e.g. by review).

NOTE 1 For safety mechanisms that were incorporated into the quantitative analysis (see B5) the effect of the safety mechanism can also be evaluated quantitatively.

NOTE 2 Additional safety mechanisms which are introduced to mitigate DFI, independently if they were introduced due to quantitative or qualitative evaluation, change the chip area and thus influence the failure rate distribution over each part of the chip. Thus the quantitative analysis usually is updated.

4.7.6.8 Evaluation of the effectiveness to control or to avoid the dependent failures (B10)

The effectiveness of the introduced safety measures to mitigate or avoid dependent failures have to be verified. The verification methods that can be applied are identical to those that are applied in the case of safety mechanisms defined to avoid or mitigate the effect of random hardware or systematic failures according to ISO 26262-5:2018, Clause 10. The following techniques can be useful:

— FTA, ETA, FMEA;

— Fault injection simulation;

— Application of specific design rules based on technology qualification tests;

— Overdesign with respect to e.g. device voltage classes or distances;

— Stress testing with respect to temperature profile or overvoltage of supply and inputs;

— EMC and ESD testing; and

— Expert judgement.

NOTE 1 The results and the arguments are documented and justified.

The verification of safety measures that were integrated into the quantitative analysis can be done in the quantitative analysis as well and sufficient improvements of the resulting metrics can be verified according to ISO 26262-5:2018, Clause 8 and 9.

NOTE 2 For the case that an introduced safety measure can be subject of dependent failures as well, their avoidance or mitigation have to be evaluated by (re)applying the DFA procedure for the newly introduced dependent failures.

NOTE 3 If there is proven experience with similar measures to mitigate dependent failures, it can be used to judge effectiveness of the measure under analysis, given that the transferability of the result can be argued.

NOTE 4 During the analysis, possible relationships between the hardware and software can be considered (see ISO 26262-6:2018, Clause 6)

4.7.6.9 Assessment of risk reduction sufficiency and if required improve defined measures (B11 & B12)

To close the DFA an evaluation of the remaining risks of dependant failures is completed. If the mitigation is not regarded to be sufficient, the safety mechanism is improved (B12) and the evaluation of the effectiveness is repeated.

For the case that residual risks can be quantified, they could be accounted in the quantitative analysis (if not already done in the quantitative analysis path via B5 & B9). For example in the case of a function and its safety mechanism which are affected by a dependent failure, the failure mode coverage of the safety mechanism has to be reduced accounting for the unmitigated dependencies.

NOTE If the targeted metrics of quantitative analyses are achieved, risk is understood as sufficiently low from the random hardware fault point of view, even if no safety mechanism is allocated to the hardware element which is affected by the fault that was identified as relevant DFI. Systematic DFI concerning the same element are handled in the DFA on a qualitative base and can lead to the definition of safety measures independent of the quantitative analysis result.

4.7.7 Examples of dependent failures analysis

Detailed examples of dependent failures analysis according to this clause are described in Annex B of this part of ISO 26262.

4.7.8 Dependent failures between software element and hardware element

Hardware and software dependent failures are in general considered separately. A joint consideration of hardware and software dependent failures is done in cases in which the safety mechanism addressing the hardware is implemented in software.

EXAMPLE 1 Software based CPU Self-Test is combined with an independent hardware watchdog so that in case the CPU fails either the CPU Self-Test will detect it or the watchdog would catch it.

EXAMPLE 2 Within the EGAS concept [56] the layer two software monitors the layer 1 software. Both software elements can run on the same hardware element. Layer one and layer two are already diverse to each other which contributes to the reduction of dependent faults violating the safety goal. To further reduce the probability of safety goal violation due to dependent faults in hardware additional safety measures are introduced, e.g. a program flow monitoring and a CPU Self-Test to address dependent failures in the CPU, inverted redundant storage of important layer two variables in the RAM module and an independent challenge and response watchdog to ensure the relevant software modules have been executed.

4.8 Fault injection

4.8.1 General

Fault injection at the semiconductor component level is a known methodology (see references [30], [31], [32], [33] and [34]) which can be used to support several activities of the lifecycle when the safety concept involves semiconductor components.

In particular, for semiconductor components, fault injection can be used for:

— Supporting the evaluation of the hardware architectural metrics:

  — Evaluating the diagnostic coverage of a safety mechanism;

  NOTE 1 If it is impractical to achieve accurate results in a reasonable time with reasonable resources, limiting the scope of the injection campaign (e.g. injection campaigns on IP block level only) or fall-back to analytical methods or to a combination with analytical methods and fault injection is possible.

  EXAMPLE 1 Fault injection used to verify the diagnostic coverage provided by software-based hardware tests or hardware-based safety mechanisms such as hardware built-in self-test.

  — Evaluating the diagnostic time interval and the fault reaction time interval; and

- — Confirming the fault effect.

  EXAMPLE 2 Fault injection used to evaluate the probability if a fault will result or not in an observable error at the output of an IP in the context of specific inputs, for example to compute the architectural vulnerability factor for transient faults as described in reference [25].

— supporting the functional verification of a safety mechanism with respect to its requirements, including its capability to detect faults and control their effect (fault reaction):

EXAMPLE 3 Fault injection used to cause an error to trigger a hardware-based safety mechanism and verify the correct reaction at related software-level.

EXAMPLE 4 Fault injection used during functional verification of safety mechanisms to verify specific corner cases.

EXAMPLE 5 Fault injection used during integration of the safety mechanisms to verify interconnectivity.

4.8.2 Characteristics or variables of fault injection

With respect to fault injection, the following information can help the verification planning:

— the description and rationale about fault models, and related level of abstraction;

— type of safety mechanism including required confidence level;

— observation points and diagnostic points;

— fault site, fault list; and

— workload used during fault injection.

In particular, the verification planning describes and justifies:

— Fault model and related level of abstraction:

- — As clarified in the following clauses for DFA, digital, analogue and PLD, fault injection can be done at the appropriate level depending on the considered fault model, the specific semiconductor technology, feasibility, observability and use case; and

  NOTE 1 Depending on the purpose, fault injection can be implemented at different abstraction levels (e.g. semiconductor component top-level, part or sub-part level, RTL, etc). A rationale for the abstraction level is provided.

  EXAMPLE 1 Selection of the abstraction level can also depend on the nature of the fault that is intended to be modelled by fault injection: the stuck-at fault can be injected at gate level netlist, whereas for bit-flips an RTL level abstraction is sufficient.

  NOTE 2 Selection can also depend on the required accuracy.

  EXAMPLE 2 The evaluation of the diagnostic coverage for stuck-at faults for a CPU software-based hardware test by fault injection at the gate level have a high confidence level.

- — Level at which to observe the effect of faults (observation points) and at which to observe the reaction of a safety mechanism (diagnostic points).

EXAMPLE 3 In the case of verification of the diagnostic coverage of a parity circuit, the observation and diagnostic points can be set at the part or sub-part level.

EXAMPLE 4 In the case of verification of the diagnostic coverage of a loopback between different IOs, they can be set at the top level.

NOTE 3 If top level fault injection is not feasible, for example, due to the complexity of the semiconductor component under test, fault injection can be done at the part or sub-part level by creating a model of the safety mechanism in the simulation environment itself. Observation and diagnostic points are set accordingly. The used model sufficiently reflects the safety properties of the safety mechanism.

EXAMPLE 5Watchdog emulation as part of the testbench.

— Fault injection method. Depending on the purpose, feasibility and observability, fault injection can be implemented via different methods;

EXAMPLE 6 Direct fault injection where fault site is known; fault injection by formal methods; fault injection by emulation; fault injection by irradiation.

— Location (fault site) and amount of faults (fault list) to be injected, considered in relationship to the failure mode under verification.

NOTE 4 Sampling factor can be used to reduce the faults list if justified with respect to the specified purpose, confidence level, type/nature of the safety mechanism, selection criteria etc.

NOTE 5 Selection criteria include (e.g. [58] and [59]): Sample size n (e.g., how many faults and time points were simulated or analysed); the result of the analysis of the sample p (for example, the ratio of stuck-at faults detected by a safety mechanism); the "desired confidence" α; the margin of error (Confidence Interval) CI, sometimes denoted by a value d such that the margin of error is p±d. A justification is provided for the choices.

EXAMPLE 7 In the context of verification of a dual-core lock-step, the relevant fault population may be limited to the compared CPU outputs and related fault locations.

EXAMPLE 8In the context of verification of the diagnostic coverage of a software-based hardware test, also CPU internal faults are relevant.

NOTE 6 Techniques like fault collapsing can also be used to reduce the faults population to prime faults.

— Fault injection controls, with respect to the related claim in the respective safety analysis; and

EXAMPLE 9 Fault injection controls can include the type of fault to be injected, the duration of a transient fault, the number of faults injected in a simulation run, time and location of fault occurrence and the window of observation of the expected action of a safety mechanism.

— Test bench (workload) used during fault injection. Depending on the specific purposes, the test bench can be an exhaustive stimuli of the circuit or similar to the expected use case.

EXAMPLE 10 In the case of verification of the completeness and correctness of a dual-core lock-step comparator, a basic workload is used, i.e. stimulating only a portion of the CPU like the outputs.

EXAMPLE 11 In the case of verification of the diagnostic coverage of an asymmetric redundancy, an exhaustive stimuli is used.

EXAMPLE 12 In the case of verification of F_{safe} (see ISO 26262-10:2018, 8.1.8) for transient faults, a workload similar to the expected use case is considered.

4.8.3 Fault injection results

Results of fault injection can be used to verify the safety concept and the underlying assumptions as listed in 4.8.1 (e.g. the effectiveness of the safety mechanism, the diagnostic coverage and amount of safe faults).

NOTE 1 Evidence of fault injection is maintained in the case of inspections during functional safety audits.

NOTE 2 An exact correspondence between the fault simulated and the fault identified in the safety analysis (e.g. for open faults) may not always exist. In such a case refinement of the safety analysis can be based on the results of other representative faults (e.g. N-detect testing as reported in 5.1.10.2).

4.9 Production and Operation

4.9.1 About Production

The first objective of ISO 26262–7:2018 Clause 5 is to develop and maintain a production process for safety-related elements or items that are intended to be installed in road vehicles.

Semiconductor products typically use standardised production processes such as wafer processing and die assembly operations. It is possible that a production process is developed for a specific product or package, but this is less common than using a standardised flow. It is not generally possible to identify distinct steps in the process flow as being safety-related or not, so everything is considered as being safety-related.

A semiconductor product is typically designed using a target process technology and associated library of device models that represent the electrical characteristics of a device fabricated with that technology. Element design is implemented in a process technology by following a sequence of standardised manufacturing processes (e.g. diffusion, oxide deposition, ion implantation, die assembly) each of which typically has risk mitigation in place through methods such as process FMEA and control plans. Libraries of device models used during product development represent the devices (e. g. transistors, resistors, capacitors) fabricated in that process technology. When the manufacturing process is in control (as verified by the implemented control plan), the element's safety-related production requirements can be achieved by following a standardised semiconductor manufacturing process. The product and process are both also verified by manufacturing test. The manufacturing test evaluates element performance against the element's electrical specification. Manufacturing process performance is evaluated against the process control specification per the process control plan. This testing process helps assure that the manufactured element complies with its requirements including the hardware safety requirements.

4.9.2 Production Work Products

The requirements of ISO 26262-7:2018, Clause 5 could be complied with to requirements of quality management systems compliant to standards such as ISO TS 16949 [52]. A semiconductor supplier or subcontractor with a quality management system compliant to such standard can find that existing work products can be partially or fully reused to satisfy the requirements of ISO 26262-7:2018, Clause 5.

EXAMPLE 1 The safety-related content of the production control plan (see ISO 26262-7:2018, 5.5.2) can partially or fully re-use the content of the quality management system's production control plan.

EXAMPLE 2 The control measures report (see ISO 26262-7:2018, 6.5.3) can partially or fully re-use the content of the quality management system's control measures report.

4.9.3 About service (maintenance and repair), and decommissioning

Typically, within the context of ISO 26262, semiconductor components have no maintenance nor decommissioning requirements, and are not repairable. As a result, work products associated with maintenance, repair and decommissioning will generally be tailored from the safety plan as they are typically out of scope for a semiconductor element.

An alignment on expectations for both the semiconductor supplier and the customer concerning service and decommissioning can be included in the Development Interface Agreement (DIA).

4.10 Interfaces within distributed developments

4.10.1 About distributed development

ISO 26262–8:2018, Clause 5 describes the procedures and allocates responsibilities within distributed developments for items and elements. This goal of this clause is to clarify the term "supplier" with respect to distributed developments involving semiconductors.

In most lifecycle tailoring the semiconductor developer is part of a distributed development as a supplier and subject to the requirements of ISO 26262-8:2018 Clause 5. The customer (i.e. Tier 1 or semiconductor integrator) is responsible for managing the semiconductor developer as a supplier with respect to safety-related development responsibility. Work products of ISO 26262-8:2018, Clause 5 which can be executed by the semiconductor developer in this context include but are not limited to:

— Development interface agreement (ISO 26262-8:2018; 5.5.2);

— Supplier's project plan (ISO 26262-8:2018; 5.5.3); and

— Supplier's safety plan (ISO 26262-8:2018; 5.5.4).

A semiconductor developer can also be a customer in a distributed development. Suppliers to semiconductor developers can be internal or external to the semiconductor developer's organization. In all such cases the semiconductor developer is responsible for managing their suppliers with respect to safety-related development responsibility. The supplier's work products for compliance to ISO 26262-8:2018, Clause 5 become part of the semiconductor developer's safety argument. Work products of ISO 26262-8:2018, Clause 5 which can be executed by the semiconductor developer in this context include but are not limited to:

— Development interface agreement (ISO 26262-8:2018; 5.5.2);

— Supplier selection report (ISO 26262-8:2018; 5.5.1); and

— Functional safety assessment report (ISO 26262-8:2018; 5.5.5).

The lowest level of a safety-related distributed development is the level at which the responsibility for safety ends. There can be suppliers at lower levels who do not have safety responsibility, such as suppliers of manufacturing materials. These lower level suppliers can be subject to requirements outside the scope of ISO 26262:2018, such as the requirements of a quality management system.

4.11 Confirmation measures and functional safety audit

Confirmation reviews, functional safety audit and functional safety assessment for semiconductors are carried out according to ISO 26262-2:2018 6.4.7, 6.4.8 and 6.4.9.

The applicability of those clauses to semiconductors is tailored according to the context in which the semiconductor device is assessed. If the semiconductor device is being developed as an SEooC, the tailoring can be done following the guidelines in ISO 26262-10:2018, 9.2.3. In the case of intellectual properties, the tailoring can be done following the guidelines in 4.5 of this part of ISO 26262.

In general, each confirmation review concerning safety at the item level (e.g. review of HARA, item integration and test plan and validation plan) will be tailored as they are typically out of scope for a semiconductor supplier.

NOTE The tailoring can be supported by checklists.

EXAMPLE The functional safety audit can be tailored by means of a Process Safety Audit (PSA). The PSA is executed according to a checklist. The PSA checklist is based on the Safety Plan and lists which are the activities and work products that are required according to the context in which the semiconductor device is assessed. If gaps are identified, measures are put in place to recover those gaps. The PSA is performed based on the required level of independency for functional safety audit as listed in ISO 26262-2:2018 Table 1.

4.12 Clarification on hardware integration and testing

The following Table 26 and Table 27 show how ISO 26262-5:2018 Table 10 and Table 11 can be applied to semiconductors.

NOTE 1 The tables are a starting point and can be modified for specific use cases with appropriate rationale.

Table 26 — Methods for deriving test cases for hardware integration testing at semiconductor level

Method	Interpretation at semiconductor level
Analysis of requirements	Relevant safety requirements are allocated to the semiconductor device NOTE This is usually done in semiconductor industry during IC functional verification (at simulation level) and functional validation (at silicon level)
Analysis of internal and external interfaces	Each verification or validation tests related to the IC integration and to the IC IOs can be claimed to be addressing this entry
Generation and analysis of equivalence classes	Test-benches are selected according to homogenous groups of features
Analysis of boundary values	Standard verification technique
Knowledge or experience based error guessing	e.g. potential design concerns identified in external analysis, e.g. design FMEA
Analysis of functional dependencies	Standard verification technique
Analysis of common limit conditions, sequences and sources of common cause failures	e.g. tests on clock, power, temperature, EMI
Analysis of environmental conditions and operational use cases	e.g. temperature cycling, SER experiments, HTOL tests
Standards if existing	e.g. standard for CAN, I2C, UART, SPI etc.
Analysis of significant variants	e.g. PVT (Process skews, Voltage, Temperature), characterization tests

Table 27 — Hardware integration tests to verify the completeness and correctness of the safety mechanisms implementation with respect to the hardware safety requirements at semiconductor level

Method	Interpretation at semiconductor level
Simulation	Standard verification technique
Post-silicon test	Limited to hardware safety requirements that can be verified at that level
Fault injection method	See 4.8.

Concerning Table 28, the use of the word "test case" is applied somewhat differently between systems and semiconductor components. Semiconductor components are tested in two ways:

— Validation testing focuses of correct integration and freedom from systematic faults and is applied to a small subset of devices;

— Production testing focuses on faults that can occur during production. State of the art production testing applies structural tests. Production testing is applied to all produced devices. This relates to clause "Production" and is not within scope of hardware integration testing.

NOTE 2 In this context, the term "test cases" refers to validation test cases that test the functional and the electrical behaviour of the design. Test structures and test equipment implemented for production testing can be helpful also for validation testing, too.

Several of the methods included in Table 28 are, in general, standard for a semiconductor test process as they relate directly to verification of data sheet technical specifications over the specified operating range (e.g. voltage, temperature, frequency) unless indicated otherwise. Methods of equivalence classes and error guessing are, in general, less relevant for the testing of semiconductor hardware and therefore less commonly used.

5 Specific semiconductor technologies and use cases

5.1 Digital components, memories

5.1.1 About digital components

Digital components include the digital part of components like microcontrollers, System on Chip (SoC) devices and Application Specific Integration Circuits (ASICs).

5.1.2 Fault models of digital components

A list of often used digital fault models include (e.g. [57]):

— Permanent;

  — stuck-at;

  — open; and

  — short.

— Transient

— SEU, MBU; and

— SET.

NOTE 1 Transition faults and similar timing related phenomena are considered when relevant for the specific technology.

NOTE 2 Some fault models can have the same effect than another fault model and therefore can be detected by the same safety mechanism. An appropriate justification is provided to show that correspondence.

EXAMPLE A safety mechanism designed to target stuck-at faults can detect bridging faults or opens that do manifest as stuck-at over time.

NOTE 3 Table 28 includes additional fault models related to memories.

5.1.3 Detailed fault models of memories

Memory fault models can vary depending on the memory architecture and memory technology. Typical fault models of semiconductor memories are shown in Table 28 which also includes guidelines for diagnostic coverage. The listing does not claim exhaustiveness and can be adjusted based on additional known faults or depending on the application.

Table 28 — fault models of memory elements

Element	Fault models
FLASH (NAND, embedded)	stuck-at, additional fault models[a,] soft error model
ROM, OTP, eFUSE	stuck-at, additional fault models[a]
EEPROM	stuck-at, additional fault models[a]
Embedded RAM	stuck-at, additional fault models[a], soft error model
DRAM	stuck-at, additional fault models[a], soft error model

[a] For example, Stuck-open Faults (SOFs), Inversion Coupling Faults (CFins), Idempotent Coupling Faults (CFids), State Coupling Faults (CFsts), Dynamic Coupling Faults (CFdyns), Bridging Coupling Faults (CFbs) as listed in references [35]. Moreover, it can include additional fault models such as addressing faults (AF), addressing delay faults (ADF), Transition Faults (TFs), Neighbourhood Pattern Sensitive Faults (NPSFs), Sense Transistor Defects (STDs), Word-line Erase Disturb (WED), Bit-line Erase Disturb (BED), Word-line Program Disturb (WPD), Bit-line Program Disturb (BPD).

NOTE 1 Fault models in b) and c) are for RAM but it can be shown (see for example references [49], [50] and [51]) that the same fault models are also valid for embedded FLASH or NAND FLASH, even if caused by different phenomena.

NOTE 2 Typically only a subset of the listed memory fault models can be activated during typical stress conditions while others can be activated at end-of-line test facilities. Therefore effectiveness of memory tests (e.g. March tests) has to be carefully evaluated.

NOTE 3 As shown by several publications (e.g.[48]), the real defect distribution can be different from memory to memory. Therefore, the previous list of fault models and the relationship with the target DC can be changed based on a specific fault model pareto.

5.1.4 Failure modes of digital components

The objective of this clause is to give a non-exhaustive list of examples about how to deal with the failure modes of digital components in the context of ISO 26262 applications.

As described in 4.2, a semiconductor component can be partitioned into parts, sub-parts and elementary sub-parts. This annex gives an example about how to characterize the failure modes at different levels of abstraction. As described in 4.3, the failure mode considers the semiconductor component at given level of hierarchy as a black box and analyses the failures according to it functional specification or function.

As example of classification, for any function of the element, the element failure can be modelled as:

— Function omission: function not delivered when needed (FM1);

— Function commission: function executed when not needed (FM2);

— Function timing: function delivered with incorrect timing (FM3); and

— Function value: function provides incorrect output (FM4).

The failure mode can be adapted to any logical function. Annex A can be used as a starting point to analyse the failure modes of typical parts and sub-parts present in a digital component. In the context of a safety analysis (ISO 26262-9:2018, Clause 8) the failure mode description is enhanced with a root cause analysis and with an effect analysis to understand how the failure mode propagates to other parts or sub-parts.
In general, the failure modes of an IP block can be described at different abstraction levels and based on different perspectives on the block's fault-free functionality and faulty behaviors. The choice of the failure mode set influences the feasibility, effort and confidence of a safety analysis. Criteria for a reasonable and objective oriented definition of the failure mode set are:

— Failure modes allow the mapping of underlying technology faults to failure modes, as described in 4.3;

— Failure modes facilitate the rationale for diagnostic coverage of applied safety measures; and

— Failure modes ideally are disjunctive, i.e. each of the originating faults ideally leads to only one particular failure mode.

NOTE With the proposed level of abstraction about FM1, FM2, FM3, FM4, it can happen that for instance in the case of FM3 also the output is incorrect (FM4) based the same physical root cause (e.g. a stuck-at fault or a soft error affecting some inner logic function). If FM3 and FM4 are controlled by different safety mechanisms with different diagnostic coverage capabilities the safety concept is more robust against failure mode distributions.

5.1.5 Example of failure mode definitions for common digital blocks

Table 29 contains exemplary, non-binding failure mode definitions for common IP blocks.

Table 29 — Example of failure modes for digital components

Part / sub-part	Function	Aspects to be considered for Failure mode[a]
	CPU	
Central Processing Unit (CPU)	Execute given instruction flow according to given Instruction Set Architecture.	CPU_FM1: given instruction flow(s) not executed (total omission) CPU_FM2: un-intended instruction(s) flow executed (commission) CPU_FM3: incorrect instruction flow timing (too early/late) CPU_FM4: incorrect instruction flow result CPU_FM1 can be further refined if necessary into: — CPU_FM1.1: given instruction flow(s) not executed (total omission) due to program counter hang up — CPU_FM1.2: given instruction flow(s) not executed (total omission) due to instruction fetch hang up

Part / sub-part	Function	Aspects to be considered for Failure mode[a]
CPU Interrupt Handler circuit (CPU_INTH)	Execute interrupt service routine (ISR) according to interrupt request	CPU_INTH_FM1: ISR not executed (omission/too few) CPU_INTH_FM2: un-intended ISR execution (commission/too many) CPU_INTH_FM3: delayed ISR execution (too early/late) CPU_INTH_FM4: incorrect ISR execution (see CPU_FM1/2/4)

1852

Table 29 *(continued)*

Part / sub-part	Function	Aspects to be considered for Failure mode[a]
CPU Memory Management Unit (CPU_MMU)	The Memory Management MMU typically performs two functions: — translates virtual addresses into physical addresses — Controls memory access permissions.	CPU_MMU_FM1: Address translation not executed CPU_MMU_FM2: Address translation when not requested CPU_MMU_FM3: delayed address translation CPU_MMU_FM4: translation with incorrect physical address CPU_MMU_FM5: un-intended blocked access CPU_MMU_FM6: un-intended allowed access CPU_MMU_FM7: delayed access
Interrupt Controller Unit (ICU)	Send interrupt requests to given CPU according to hardware-based or software-based interrupt events and according to intended quality of service (e.g. priority). The interrupt controller can service multiple CPUs.	ICU_FM1: Interrupt request to CPU missing ICU_FM2: Interrupt request to CPU without triggering event ICU_FM3: Interrupt request too early/late ICU_FM4: Interrupt request sent with incorrect data
DMA	DATA TRANSFER: Move Data when requested from source address(es) to destination address(es) and notify the data transfer completion. The set of data transferred is called a message.	DMA_FM1: No requested data transfer. The message is not sent as intended to the destination address. DMA_FM2: Data transfer without a request. DMA_FM3: Data transfer too early/late. DMA_FM4: Incorrect output
(first level of abstraction)		
Busses and Interconnects (internal communication)	Deliver bus transaction initiated from a given bus master to the target address according to the intended quality of service (TXFR). A transaction is a given set of data as defined by the bus protocol.	BUS_TXFR_FM1: Requested transaction not delivered BUS_TXFR_FM2: Transaction delivered without a request BUS_TXFR_FM3: Transaction delivered with incorrect timing BUS_TXFR_FM3: Transaction delivered with incorrect data
External SDRAM with SDRAM Controller	Volatile memory provides (read) or store (write) data to given row and column address according to input command from SDRAM controller.	SDRAM_RW1: given write/read access not executed (omission) SDRAM_RW2: un-intended write/read access executed (commission) SDRAM_RW3: incorrect write/read access result (too early/late) SDRAM_RW4: incorrect write/read access result
or (second level of abstraction)		
External SDRAM with SDRAM Controller	SDRAM controller provides row address to be prepared for read or write operation on a selected bank.	SDRAM_RA1: given row address not accessed (omission) SDRAM_RA2: un-intended row address accessed (commission) SDRAM_RA3: delayed row address result (too early/late) SDRAM_RA4: incorrect row address result

External SDRAM with SDRAM Controller	SDRAM controller provides column address to access data for read or write operation.	SDRAM_CA1: given column address not accessed (omission) SDRAM_CA2: un-intended column address accessed (commission) SDRAM_CA3: delayed column address result (too early/late) SDRAM_CA4: incorrect column address result
External SDRAM with SDRAM Controller	SDRAM controller provides commands (e. g. activate, write, read, pre-charge, refresh ...) to get data for read or write operation.	SDRAM_IN1: given instruction not executed (omission) SDRAM_IN2: un-intended instruction executed (commission) SDRAM_IN3: delayed instruction result (too early/late) SDRAM_IN4: incorrect instruction result

1853

Table 29 *(continued)*

Part / sub-part	Function	Aspects to be considered for Failure mode[a]
External SDRAM with SDRAM Controller	SDRAM data path provides write / read data to / from memory array.	SDRAM_DW1: given data word not executed (omission) SDRAM_DW2: un-intended data word executed (commission) SDRAM_DW3: delayed data word result (too early/late) SDRAM_DW4: incorrect data word result
External FLASH with FLASH Controller	Non-volatile memory provides (read) or store (write) data to given address according to input command from FLASH controller.	FLASH_RW1: given write/read access not executed (omission) FLASH_RW2: un-intended write/read access executed (commission) FLASH_RW3: delayed write/read access result (too early/late) FLASH_RW4: incorrect write/read access result
(first level of abstraction)		
SRAM with SRAM controller	Provides storage for variables and/or constants. The analysis is done after considering the access control logic called SRAM controller from the perspective of an hardware element issuing a command. Typically a command is a read, write or possibly a read-modify-write.	SRAM_RW1: given command not executed (omission) SRAM_RW2: un-intended command executed (commission) SRAM_RW3: delayed command result (too early/late) SRAM_RW4: incorrect command result
SRAM with SRAM controller	SRAM hard-macro (HM): Provides data or stores data to given address according to input command from SRAM controller.	SRAM_HM_FM1: command from SRAM controlled not executed (omission) SRAM_HM_FM2: unintended access to the SRAM caused e.g. by a transient fault SRAM_HM_FM3: SRAM command delayed (too early/late) e.g. delayed by the internal timing generation SRAM_HM_FM4: Final SRAM data corrupt or written at wrong location
Embedded FLASH (eFLASH) with eFLASH controller	Non-volatile memory (NVM) stores program code and data constants. Program and erase function. Erase suspend and resume operations to interrupt on-going erase operation.	eFLASH_E_FM1: Program or erase not performed. eFLASH_E_FM2: Program or erase performed when not requested. eFLASH_E_FM3: Incorrect Program or erase timing eFLASH_E_FM4: Program or erase performed with wrong content.
	Non-volatile memory (NVM) stores program code and data constants. Read Function	eFLASH_R_FM1: Read access not performed. eFLASH_R_FM2: Read access when not requested. eFLASH_R_FM3: Incorrect read access timing. eFLASH_R_FM4: Read access delivers wrong content.

1854

1855

Table 29 *(continued)*

Part / sub-part	Function	Aspects to be considered for Failure mode[a]
Data coherency	Coherency is defined by coherence invariants independent of the underlying architecture. The invariants chosen for this example are based on [53]	Based on the complexity of the topic the failure modes are just few examples on situations that can lead to a non-coherent state of given addresses. COHERENCY_FM1: Write to memory A not executed (omission). Memory is seen as updated by the participants in the coherency. This failure mode leads to a non-coherent state for memory A. COHERENCY_FM2: Un-intended write to memory A (comission). This situation can be related to the situation where many cores attempt to write to the same location. COHERENCY_FM3: delayed update (write) of memory A (too early/late). A possible situation is when a legal write is delayed but the other agents participating in the coherency protocol think the address content is coherent. COHERENCY_FM4: Content of memory A is corrupt. This can be caused by an incorrect write command (see e.g. SRAM) or by a defect in the storage element.
Communication Peripheral (COM) Can be applied to CAN, Flexray, Ethernet, SPI	Transfer Data provided by software to external interface according to the interface protocol. Receive and process data provided by an external interface according to interface protocol. Notify software about the availability of data. The set of data transferred is called a message.	COM_TX_FM1: No message transferred as requested COM_TX_FM2: Message transferred when not requested COM_TX_FM3: Message transferred too early/late COM_TX_FM4: Message transferred with incorrect value COM_RX_FM1: No incoming message processed COM_RX_FM2: Message transferred when not requested COM_RX_FM3: Message transferred too early/late COM_RX_FM4: Message transferred with incorrect value
Signal processing accelerator	Takes high bandwidth signals from a source (e.g. sensor data) and processes them (e.g. arithmetically) according to a given code and/or configuration (e.g. GPU, DSP). Typically this is done to offload a general purpose CPU which could do that task only less efficiently. Typically this processing needs to comply with real time requirements.	SP_FM1: Processing stalled, no or constant output (service omission) SP_FM2: Unrequested output or interrupts (service commission) SP_FM3: Output structurally broken, e.g. corrupt frames (service timing) SP_FM4: Output structurally OK, but erroneous data (service value)
[a] Failure Modes can be caused by permanent and transient random hardware faults		

5.1.6 Qualitative and quantitative analysis of digital component

As seen in ISO 26262-9:2018, Clause 8, qualitative and quantitative safety analyses are performed at the appropriate level of abstraction during the concept and product development phases. In the case of a digital component:

— Qualitative analysis is useful to identify failures. One of the possible ways in which it can be performed uses information derived from digital component block diagrams and information derived from this part of ISO 26262;

NOTE 1 Annex A gives an example about how to define failure modes for digital components.

NOTE 2 Qualitative analysis includes dependent failures analysis of this part as seen in 4.7.

— Quantitative analysis is performed using a combination of:

  — Logical block level structuring;

  — Information derived from the digital component Register Transfer Level (RTL) description (to obtain functional information) and gate-level net list (to obtain functional and structural information);

  — Information to evaluate potential unspecified interaction of sub-functions (dependent failures, see clause 4.7);

  — Layout information - only available in the final stage;

  — Information for the verification of diagnostic coverage with respect to some specific fault models such as bridging faults. This can be applicable to only some cases like the points of comparison between a part and its corresponding safety mechanism; and

  — Expert judgement supported by rationale and careful consideration of the effectiveness of the system-level measures.

NOTE 3 ISO 26262-5:2018, Annex D can be used as a starting point for diagnostic coverage (DC) with the claimed DC supported by a proper rationale.

NOTE 4 The information for quantitative analysis can be progressively available during the digital component development phase. Therefore, the analysis could be repeated based on the latest information.

EXAMPLE 1 During a first step of the quantitative analysis, a pre-Design For Test (DFT) pre-layout gate-level net list could be available, while later the analysis is repeated using post-DFT and post-layout gate-level net list.

— Since the parts and sub-parts of a digital component can be implemented in a single physical component, both dependent failures analysis and analysis of independence or freedom from interference are important analyses for digital components. See 4.7 for further details.

NOTE 5 The analysis of dependent failures is performed on a qualitative basis because no general and sufficiently reliable method exists for quantifying such failures.

EXAMPLE 2 The evaluation of dependent failures starts early in design. Design measures are specified to avoid and reveal potential sources of dependent failures or to detect their effect on the “System on Chip” safety performance. Layout confirmation is used in the final design stage.

5.1.7 Determination of failure rate associated with residual faults of a digital component

5.1.7.1 General

Requirements and recommendations for the failure rates computation in general are defined in ISO 26262-5:2018 and tailored for semiconductor components in 4.6 of this part of ISO 26262.

Following the example given in ISO 26262-5:2018, Annex E, the failure rates and the metrics can be computed in the following way for digital components:

— First, the digital component is divided into hierarchical levels (parts, sub-parts or elementary sub-parts) as required;

NOTE 1 Assumptions on the independence of identified parts are verified during the dependent failures analysis.

NOTE 2 The necessary level of detail (e.g. whether to stop at part level or if to go down to sub-part or elementary sub-part level) can depend on the stage of the analysis and on the safety mechanisms used (inside the digital component or at the system or element level).

EXAMPLE 1 In the case of a CPU with a hardware lock-step safety mechanism, the analysis considers the CPU function as a whole while more detail can be needed for the lock-step comparator.

EXAMPLE 2 In the case of a CPU with a structural software-based hardware test, the failure mode is defined in more detail because the software test will cover different failure modes with different failure mode coverage.

EXAMPLE 3 The confidence of the computation of failure rate of parts or sub-parts can be proportional to the level of detail: a low level of detail could be appropriate for analysis at concept stage while a higher level detail could be appropriate for analysis at the development stage.

NOTE 3 Due to the complexity of modern digital components (hundreds or thousands of parts and sub-parts), to guarantee completeness of the analysis, it is helpful to support the division process with automatic tools. Care is taken to ensure digital component level analysis across module boundaries. Partitions are done along levels of RTL hierarchy if RTL is available.

— Second, the failure rates of each part or sub-part can be computed using one of the following two methods, as already described in 4.6.3.4:

— If the total failure rate of the whole digital component die (i.e. excluding package and bonding) is given (in FIT), then the failure rate of the part or sub-part could be assumed to be equal to the occupying area of the part or sub-part (i.e. area related to gates, flip-flops and related interconnections) divided by the total area of the digital component die multiplied by the total failure rate, or

NOTE 4 For mixed signal chips with power stages, this approach is applied within each domain, as the total failure rate for the digital domain can be different from the analogue and power domain. See 5.2 for further details.

EXAMPLE 4 If a CPU area occupies 3 % of the whole digital component die area, then its failure rate could be assumed to be equal to 3 % of the total digital component die failure rate.

— If the base failure rates, i.e. the failure rate of basic sub-parts like gates of the digital component, are given, then the failure rate of the part or sub-part could be assumed to be equal to the sum of the number of those basic sub-parts multiplied by its failure rate.

NOTE 5 See 4.6 for examples for how to derive the base failure rate values.

— The evaluation is completed by classifying the faults into safe faults, residual faults, detected dual-point faults and latent dual-point faults; and

EXAMPLE 5 Certain portions of a debug unit implemented inside a CPU are safety-related (because the CPU itself is safety-related), but they themselves cannot lead to a direct violation of the safety goal or their occurrence cannot significantly increase the probability of violation of the safety goal.

— Finally, the failure mode coverage with respect to residual and latent faults of that part or sub-part is determined.

EXAMPLE 6 The failure mode coverage associated with a certain failure rate can be computed by dividing the sub-part into smaller sub-parts, and for each of them compute the expected capability of the safety mechanisms to cover each sub-part. For example, the failure mode coverage of a failure in the CPU register bank can be computed by dividing the register bank into smaller sub-parts, each one related to the specific register (e.g. R0, R1,…), and computing the failure mode coverage of the safety mechanism for each of them, e.g. combining the failure mode coverage for each of the corresponding low-level failure modes.

NOTE 6 The effectiveness of safety mechanisms could be affected by dependent failures. Adequate measures are considered as listed in 4.7.

NOTE 7 Due to the complexity of modern digital components (millions of gates), fault injection methods can assist the computation and be used for verification of the amount of safe faults and especially of the failure mode coverage. See 4.8 and 5.1.10 for details. Fault injection is not the only method, and other approaches are possible as described in 5.1.10.

5.1.7.2 How to consider transient faults of digital components

5.1.7.2.1 Failure rate of transient fault

As seen in NOTE 2 of ISO 26262-5:2018, 8.4.7, the transient faults are considered when shown to be relevant due, for instance, to the technology used. They can be addressed either by specifying and verifying a dedicated target “single-point fault metric” value to them or by a qualitative rationale.

NOTE The selection is justified.

When the quantitative approach is used, failure rates for transient faults of each part or sub-part is computed using the base failure rate for transient faults.

Due to the amount and density of memory elements in RAM memories, the resulting failure rates for transient faults can be significantly higher than the ones related to processing logic or other parts of a digital component. Therefore, as recommended in NOTE 1 of ISO 26262-5:2018, 8.4.7, it can be helpful to compute a separate metric for RAM memories and for the other parts of the digital component.

5.1.7.2.2 Classification of transient fault

For transient faults, the amount of safe faults can be particularly relevant. To justify the estimated amount of safe transient faults, a rationale is made available

NOTE 1 The rationale can be derived from fault injection as described in clause 4.8 or arguments based on the circuit architecture or application

EXAMPLE 1 A fault in a register storing a safety-related constant (i.e. a value written only once but read at each clock cycle and, if wrong, violating the safety goal) is never safe. If instead, for example, the register is written every 10 ms but used for a safety-related calculation only once, 1 ms after it is written, a random transient fault in the register would result in 90 % safe faults because in the remaining 90 % of the clock cycles, a fault in that register will not cause a violation of the safety goal.

NOTE 2 As seen in Note 2 of ISO 26262-5:2018, 8.4.7, transient faults can be addressed via a single-point fault metric. Transient faults are not considered as far as latent faults are concerned. No failure mode coverage for latent faults is computed for transients because the root cause rapidly disappears (per definition of transient). Furthermore, it is assumed that in the greatest majority of the cases, the effect will rapidly be removed, e.g. by a following power-down cycle removing the erroneous state of the flip-flop or memory cell that was changed by the transient fault, before a second fault can cause the occurrence of a multiple-point failure. In special cases, this assumption could not be valid and additional measures can be necessary and addressed on a case by case basis.

NOTE 3 Transient faults are contained within the affected sub-part and do not spread inadvertently to other sub-parts if they are not logically connected.

NOTE 4 Some of the coverage values of safety mechanisms defined in tables from D.3 to D.10 of ISO 26262-5:2018, Annex D are valid for permanent faults only. This important distinction can be found in the related safety mechanism description, in which it is written how the coverage value can be considered for transient faults.

EXAMPLE 2 The typical value of the coverage of RAM March test (see Table 32) is rated HIGH. However in the related description (clause 5.1.13.7), it is written that these types of tests are not effective for soft error detection. Therefore, for example, the coverage of RAM March test with respect to transient faults is zero.

5.1.8 Example of quantitative analysis

An example of quantitative analysis is given in Annex C.

5.1.9 Example of techniques or measures to detect or avoid systematic failures during design of a digital component

The general requirements and recommendations related to hardware architecture and detailed design are respectively defined in ISO 26262-5:2018, 7.4.1 and ISO 26262-5:2018, 7.4.2. Moreover, requirements related to hardware verification are given in ISO 26262-5:2018, 7.4.4.

A digital component is developed based on a standardised development process. The two following approaches are examples of how to provide evidence that sufficient measures for avoidance of systematic failures are taken during the development of a digital component:

— using a checklist such as the one reported in Table 30; and

— giving the rationale by field data of similar products which are developed based on the same process as the target device.

Moreover, the following general guidelines can be considered:

— the documentation of each design activity, test arrangements and tools used for the functional simulation and the results of the simulation;

— the verification of each activity and its results, for example by simulation, equivalence checks, timing analysis or checking the technology constraints;

— the usage of measures for the reproducibility and automation of the design implementation process (script based, automated work and design implementation flow); and

NOTE This implies ability to freeze tool versions to enable reproducibility in the future in compliance with the legal requirements.

— the usage – for 3rd party soft-cores and hard-cores – of validated macro blocks and to comply with each constraint and procedure defined by the macro core provider if practicable.

Table 30 — Example of techniques or measures to achieve compliance with ISO 26262-5:2018 requirements during the development of a digital component

ISO 26262-5:2018 requirement	Design phase	Technique/Measure	Aim
7.4.1.6 Modular design properties	**Design entry**	Structured description and modularization	The description of the circuit's functionality is structured in such a fashion that it is easily readable, i.e. circuit function can be intuitively understood on the basis of description without simulation efforts.
7.4.1.6 Modular design properties		Design description in HDL	Functional description at high level, e.g. at Register Transfer Level (RTL), in hardware description language, for example, VHDL or Verilog.
7.4.4 Verification of hardware design		HDL simulation	Functional verification of circuit described in VHDL or Verilog by means of simulation.
7.4.4 Verification of hardware design		Formal assertion based verification (model checking)	Functional verification of circuit described in VHDL or Verilog by means of formal functional verification.
7.4.4 Verification of hardware design		Requirement Driven Verification	All functional and safety-related requirements are verified. To be shown via traceability between specification and verification plan.
7.4.4 Verification of hardware design		Functional verification on module level	Functional verification "bottom-up" for example by simulation or formal verification.
7.4.4 Verification of hardware design		Functional verification on top level	Verification of the entire circuit.
7.4.2.4 Robust design principles		Restricted use of asynchronous constructs	Avoidance of typical timing anomalies during synthesis, avoidance of ambiguity during simulation and synthesis caused by insufficient modelling, design for testability. This does not exclude that for certain types of circuitry, such as reset logic or for very low-power microcontrollers, asynchronous logic could be useful: in this case, the aim is to suggest additional care to handle and verify those circuits.
7.4.2.4 Robust design principles		Synchronisation of primary inputs and control of metastability	Avoidance of ambiguous circuit behaviour as a result of set-up and hold timing violation.
7.4.4 Verification of hardware design		Functional and structural coverage-driven verification (with coverage of verification goals in percentage)	Quantitative assessment of the applied verification scenarios during the functional test. The target level of coverage is defined and shown.
7.4.2.4 Robust design principles		Observation of coding guidelines	Strict observation of the coding style results in a syntactic and semantic correct circuit code.
7.4.4 Verification of hardware design		Application of code checker	Automatic verification of coding rules ("Coding style") by code checker tool.
7.4.4 Verification of hardware design		Documentation of simulation results	Documentation of each data needed for a successful simulation in order to verify the specified circuit function.
7.4.4 Verification of hardware design	**Synthesis**	To check timing constraints, or static analysis of the propagation delay (STA - Static Timing Analysis)	Verification of the achieved timing constraint during synthesis.

7.4.4 Verification of hardware design		Comparison of the gate netlist with the reference model (formal equivalence check)	Functional equivalence check of the synthesised gate netlist.
7.4.1.6 Modular design properties		Documentation of synthesis constraints, results and tools	Documentation of each defined constraint that is necessary for an optimal synthesis to generate the final gate netlist.
7.4.1.6 Modular design properties		Script based procedures	Reproducibility of results and automation of the synthesis cycles.
7.4.2.4 Robust design principles		Adequate time margin for process technologies in use for less than 3 years	Assurance of the robustness of the implemented circuit functionality even under strong process and parameter fluctuation.

2027

2028 **Table 30 *(continued)***

ISO 26262-5:2018 requirement	Design phase	Technique/Measure	Aim
7.4.1.6 Modular design properties (testability)	**Test insertion and test pattern generation**	Design for testability (depending on the test coverage in percent)	Avoidance of not testable or poorly testable structures in order to achieve high test coverage for production test or on-line test.
7.4.1.6 Modular design properties (testability)		Proof of the test coverage by ATPG (Automatic Test Pattern Generation) based on achieved test coverage in percent	Determination of the test coverage that can be expected by synthesised test pattern (Scan-path, BIST) during the production test. The target level of coverage and fault model are defined and shown.
7.4.4 Verification of hardware design		Simulation of the gate netlist after test insertion, to check timing constraints, or static analysis of the propagation delay (STA)	Verification of the achieved timing constraint during test insertion.
7.4.4 Verification of hardware design		Comparison of the gate netlist after test insertion with the reference model (formal equivalence check)	Functional equivalence check of the gate netlist after test insertion.
7.4.4 Verification of hardware design	**Placement, routing, layout generation**	Simulation of the gate netlist after layout, to check timing constraints, or static analysis of the propagation delay (STA)	Verification of the achieved timing constraint during back-end.
7.4.4 Verification of hardware design		Analysis of power network	Show robustness of power network and effectiveness of related safety mechanisms. Example: IR drop test.
7.4.4 Verification of hardware design		Comparison of the gate netlist after layout with the reference model (formal equivalence check)	Functional equivalence check of the gate netlist after back-end.
7.4.4 Verification of hardware design		Design rule check (DRC)	Verification of process design rules.
7.4.4 Verification of hardware design		Layout versus schematic check (LVS)	Verification of the layout.
7.4.5 Production, operation, service and decommissioning 9.4.2.5 Dedicated measures	**Safety-related special characteristic s during chip production**	Determination of the achievable test coverage of the production test	Evaluation of the test coverage during production tests with respect to the safety-related aspects of the digital component.
7.4.5 Production, operation, service and decommissioning 9.4.2.5 Dedicated measures		Determination of measures to detect and weed out early failures	Assurance of the robustness of the manufactured chip for the selected technology process. For example, for gate oxide integrity (GOI): high temp/high voltage operation (Burn-In), high current operation, voltage stress, etc. Other example of tests are EM, Stress migration and NBTI tests.
7.4.5 Production, operation, service and decommissioning	**Qualification of hardware component**	Definition and execution of qualification tests like Brown-out test, High	For a digital component with integrated brown-out detection, the digital component functionality is tested to verify that the outputs of the digital

10 Hardware integration and testing		Temperature Operating Lifetime (HTOL) test and functional testcases	component are set to a defined state (for example by stopping the operation of the microcontroller in the reset state) or that the brown-out condition is signalled in another way (for example by raising a safe-state signal) when any of the supply voltages monitored by the brown-out detection reach a low boundary as defined for correct operation. For a digital component without integrated brown-out detection, the digital component functionality is tested to verify if the digital component sets its outputs to a defined state (for example by stopping the operation of the digital component in the reset state) when the supply voltages drop from nominal value to zero. Otherwise an assumption of use is defined, and an external measure is considered.

5.1.9.1 Principles, techniques or measures to detect or avoid systematic failures during RTL design

Some of the principles, techniques or measures used for software development (see ISO 26262-6:2018) can be considered in order to mitigate systematic failures during RTL design.

Due to the differences between using RTL for hardware design and software development, none of the contents of ISO 26262-6:2018 can be applied directly without adequate tailoring and adoption of the specific needs of RTL hardware design.

EXAMPLE 1 Similar effects of static code analysis (see ISO 26262-6:2018, Table 9, entry 1h) can be achieved by application of automatic verification of coding rules ("Coding style") by code checker tool.

EXAMPLE 2 Similar effects of methods listed in ISO 26262-6:2018, Table 9, ISO 26262-6:2018, Table 10 and ISO 26262-6:2018, Table 11 can be achieved by application of functional and structural coverage-driven verification (with coverage of verification goals in percentage) and formal methods based on properties.

NOTE 1 About quantitative assessment of the applied verification scenarios during the functional test, the target level of coverage can be based on: statement coverage, block coverage, conditional/expression coverage, branch/decision coverage, toggle coverage and Finite State Machine (FSM) coverage.

NOTE 2 In the case of a high-level synthesis flow, like developing in OpenCL, C-to-HDL flows, or a model based approach, interactions with the requirements of ISO 26262-6:2018 can be more applicable.

5.1.10 Verification using fault injection simulation

5.1.10.1 General

As mentioned in 4.8, fault injection is a useful method for semiconductor components. This is especially true for digital circuits for which fault insertion testing of single-event upsets at the hardware level is impractical or even impossible for certain fault models. Therefore, fault injection using design models (e.g. fault injection done at the gate-level netlist) is helpful to complete the verification step.

NOTE 1 Fault injection can be used both for permanent (e.g. stuck-at faults) and transient (e.g. single-event upset) faults.

NOTE 2 Fault injection is just one of the possible methods for verification, and other approaches are possible.

Fault injection utilizing design models can be successfully used to assist in verification of safe faults and computation of their amount and failure mode coverage.

EXAMPLE 1 Injecting faults and utilizing well-specified observation points to determine if the fault caused a measurable effect. Moreover, it can be used to assist the computation and to verify the values of failure mode coverage, i.e. injecting faults that were able to cause a measurable effect and determining if those faults were detected within the FTTI by the safety mechanisms.

The confidence of the computation and verification with fault injection is proportional to:

— the quality and completeness of the test-bench used to stimulate the circuit under test;

NOTE 2 The quality and completeness of a test-bench is measured in terms of its capability to activate the circuit under test. It can be measured in terms of functional coverage of the test-bench.

— the completeness of the fault injection campaign measured as a ratio of fault scenarios with respect to each possible scenarios;

NOTE 3 A scenario includes the fault site, fault occurrence, fault duration, etc.

— the level of detail of the circuit representation; and

EXAMPLE 2 Gate-level netlist is appropriate for fault injection of permanent faults such as stuck-at faults. Hardware accelerator-based methods could be helpful in order to maximize test execution speed. "Register Transfer Level" is also an acceptable approach for stuck-at faults, provided that the correlation with gate level is shown.

EXAMPLE 3 Modelling at a Register Transfer Level (RTL) is appropriate for fault injection of SEU transient faults. Simulation models are also an acceptable approach for SEU transient faults, provided that suitable correlation is demonstrated with RTL or gate-level models.

— the details available for the safety mechanisms to be simulated.

5.1.10.2 About verification of fault models different than stuck-at

Clause 5.1.2 shows that fault models others than stuck-at need to be considered.

EXAMPLE 1 A suitable way to simplify the verification of non-stuck-at faults can be to provide evidence that the fault distribution of stuck-open/bridging faults is a very limited portion of the whole fault models population, i.e. much lower than the stuck-at 0/1 fault population.

EXAMPLE 2 In general, hardware safety mechanisms can be more effective to detect each kind of fault and easier to be verified using e.g. the N-detect approach. On the other hand, in the case of a software-based safety mechanism addressing random hardware failures, it can be difficult with the N-detect technique to gain a high level of confidence in the pattern richness due to the possible change of the context between subsequent executions of the test at run time. In this case, alternative solutions can be applied (e.g. [40]).

If properly exercised, methods derived from stuck-at simulations (like N-detect testing, see for example [36]-[38]) are known to be effective for verification of non-stuck-at fault models as well.

EXAMPLE 3 Since exhaustiveness is not required, the non-stuck-at fault models analysis can be applied to a sub-set of the digital component sub-parts selected depending on their possible impact (for example comparators) or on a statistical basis.

EXAMPLE 4 For N-detect testing, "properly exercised" means that N different detections of the same fault are guaranteed by the pattern set (i.e. pattern richness). N can range from 5 to 10.

NOTE Fault injection can also be used to inject bridging faults in specific locations based on layout analysis or to verify the impact of dependent failures such as injection of clock and reset faults.

5.1.11 Example of safety documentation for a digital component

The necessary information from the work products is provided to the system integrator, including documentation of assumed requirements, assumptions related to the design external to the SEooC and applicable work products.

On that basis, the safety documentation for an SEooC digital component can include the following documents or a sub-set of them as specified in the DIA:

— the safety case related to the digital component, see ISO 26262-2:2018, 6.5.3;

— the safety plan for the digital component, see ISO 26262-2:2018, 6.5.2;

— other plans as seen in ISO 26262-8:2018, when applicable, such as configuration management plan, change management plan, impact analysis and change request plan, verification plan, documentation management plan and software tool qualification plan;

— the evidence related to the execution of the applicable steps of a safety plan as seen in ISO 26262-2;

— the hardware specifications as seen in ISO 26262-5:2018, such as hardware safety requirements specification, hardware-software Interface (HSI) specification and hardware design specification;

— the reports related to the execution of the applicable steps of the verification plan and other plans as seen in ISO 26262-5:2018 and ISO 26262-8:2018, such as hardware safety requirements verification report, hardware design verification report, and hardware integration and verification report; and

— the reports related to safety analyses as seen in ISO 26262-5:2018, ISO 26262-8:2018 and ISO 26262-9:2018, such as hardware safety analysis report, review report of the effectiveness of the architecture of the digital component to cope with random hardware failures, review report of evaluation of safety goal violations due to random hardware failures and results of analyses of dependent failures.

NOTE 1 The DIA specifies which documents are made available and what level of detail is provided to the digital component's customer.

The following information can be considered:

— the description of ISO 26262 lifecycle tailored for the digital component; list of applicable work products (description of which work products of the ISO 26262 lifecycle are applicable for the digital component);

— the description of the digital component safety architecture with an abstract description of digital component functionalities and description of safety mechanisms;

— the description of Assumptions of Use (AoU) of the digital component with respect to its intended use, including: assumption on the digital component safe state; assumptions on FTTI and MPFDI; assumptions on the digital component context, including its external interfaces;

— the description of the digital component configuration and related hardware and/or software procedures to control a failure after its detection;

— the DIA defines which of the following reports are needed at system/item level:

— Hardware safety analysis report;

— Report of the effectiveness of the architecture of digital component to cope with random hardware faults;

— Report of evaluation of safety goal violation due to random hardware failures; and

— Results of analyses of dependent failures.

— the description of the functional safety assessment process; list of confirmation measures and description of the independency level; summary of process for avoidance of systematic failures in the digital component.

NOTE 2 This documentation can be combined in one document named a “Safety Manual” or “Safety Application Note” of the digital component.

5.1.12 Examples of safety mechanisms for digital components and memories

For memories, the following Table 31 and Table 32 can be applied.

Table 31 — Non-volatile memory

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Parity bit	ISO 26262-5:2018, D.2.5.2	Low	—
Memory monitoring using error-detection-correction codes (EDC-ECC)	5.1.13.1	High	The effectiveness depends on the number of redundant bits. Can be used to correct errors
Modified checksum	5.1.13.2	Low	Depends on the number and location of bit errors within test area
Memory Signature	5.1.13.3	High	—
Block replication	ISO 26262-5:2018, D.2.4.4	High	—

Table 32 — Volatile memory

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
RAM pattern test	5.1.13.5	Medium	High coverage for stuck-at failures. No coverage for linked failures. Can be appropriate to run under interrupt protection
RAM March test	ISO 26262-5:2018, D.2.5.3	High	Depends on the write read order for linked cell coverage. Test generally not appropriate for run time
Parity bit	5.1.13.6	Low	—
Memory monitoring using error-detection-correction codes (EDC-ECC)	5.1.13.1	High	The effectiveness depends on the number of redundant bits. Can be used to correct errors
Block replication	ISO 26262-5:2018, D.2.4.4	High	Common failure modes can reduce diagnostic coverage

Running checksum/CRC	ISO 26262-5:2018, D.2.5.4	High	The effectiveness of the signature depends on the polynomial in relation to the block length of the information to be protected. Care needs to be taken so that values used to determine checksum are not changed during checksum calculation Probability is 1/maximum value of checksum if random pattern is returned

For general digital logic, Table 33 can be applied.

Table 33 — Combinatorial and sequential logic

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Self-test by software	ISO 26262-5:2018, D.2.3.1	Medium	—
Self-test supported by hardware (one-channel)	ISO 26262-5:2018, D.2.3.2	Medium	Higher coverage is possible, depending on effectiveness of test. Gate level is an appropriate level for this test

For on-chip interconnect, Table 34 can be applied.

Table 34 — On-chip communication

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
One-bit hardware redundancy	ISO 26262-5:2018, D.2.7.1	Low	—
Multi-bit hardware redundancy (including EDC-ECC)	ISO 26262-5:2018, D.2.7.2	Medium	Multi-bit redundancy can achieve high coverage by proper interleaving of data, address and control lines, and if combined with some complete redundancy, e.g. for the arbiter.
Complete hardware redundancy	ISO 26262-5:2018, D.2.7.3	High	Common failure modes can reduce diagnostic coverage
Test pattern	ISO 26262-5:2018, D.2.6.1	High	Depends on type of pattern

5.1.13 Overview of techniques for digital components and memories

5.1.13.1 Memory monitoring using error-detection-correction codes (EDC-ECC)

NOTE 1 This technique/measure is referenced in Table 31 and Table 32 of this document.

Aim: To detect each single-bit failure, each two-bit failure, some three-bit failures, and some all-bit failures in a word (typically 32, 64 or 128 bits).

Description: Every word of memory is extended by several redundant bits to produce a modified Hamming code with a Hamming distance of at least 4. Every time a word is read, checking of the redundant bits can determine whether or not a corruption has taken place. If a difference is found, a failure message is produced.

The procedure can also be used to detect addressing failures, by calculating the redundant bits for the concatenation of the data word and its address. Otherwise for addressing failures, the probability of detection is dependent on the number of EDC-ECC bits for random data returned (for example, address line open or address line shorted to another address line such that an average of the two cells is returned). The coverage is 0 % if the addressing error leads to a completely different cell selected.

For RAM cell write-enable failure, EDC-ECC can provide high coverage if the cell cannot be initialized. The coverage is 0 % if the write-enable failure affects the entire cell after it has been initialized.

5.1.13.2 Modified checksum

NOTE This technique/measure is referenced in Table 31 of this document.

Aim: To detect each single bit failure.

Description: A checksum is created by a suitable algorithm which uses each of the words in a block of memory. The checksum can be stored as an additional word in ROM, or an additional word can be added to the memory block to ensure that the checksum algorithm produces a predetermined value. In a later memory test, a checksum is created again using the same algorithm, and the result is compared with the stored or defined value. If a difference is found, a failure message is produced (see Reference [35]). The probability of a missed detection is 1/(2^size of checksum) if a random result is returned. If certain data disturbances are more probable, some checksums can provide a better detection ratio than the one for random results.

5.1.13.3 Memory signature

NOTE 1 This technique/measure is referenced in Table 31 of this document.

Aim: To detect each one-bit failure and most multi-bit failures.

Description: The contents of a memory block are compressed (using either hardware or software) into one or more bytes using, for example, a cyclic redundancy check (CRC) algorithm. A typical CRC algorithm treats the whole contents of the block as byte-serial or bit-serial data flow, on which a continuous polynomial division is carried out using a polynomial generator. The remainder of the division represents the compressed memory contents – it is the "signature" of the memory – and is stored. The signature is computed once again in later tests and compared with one already stored. A failure message is produced if there is a difference.

CRCs are particularly effective in detecting burst errors. The effectiveness of the signature depends on the polynomial in relation to the block length of the information to be protected. The probability of a missed detection is 1/(2^size of checksum) if a random result is returned (see Reference [35]).

NOTE 2 Use of an 8 bit CRC is not generally considered the state of the art for memory sizes above 4k.

5.1.13.4 Block replication (for example double memory with hardware or software comparison)

NOTE This technique/measure is referenced in Table 31 and Table 32 of this document.

Aim: To detect each bit failure.

Description: The address space is duplicated in two memories. The first memory is operated in the normal manner. The second memory contains the same information and is accessed in parallel to the first. The outputs are compared and a failure message is produced if a difference is detected. Dependent on memory subsystem design, storage of inverse data in one of the two memories can enhance diagnostic coverage. Coverage can be reduced if failure modes (such as common address lines, write-enables) exist that are common to both blocks or if physical placement of memory cells makes logically distant cells physical neighbours.

5.1.13.5 RAM Pattern test

NOTE 1 This technique/measure is referenced in Table 32 of this document.

Aim: To detect predominantly static bit failures.

Description: A bit pattern followed by the complement of that bit pattern is written into the cells of memory.

RAM locations are generally tested individually. The cell content is stored and then all 0s are written to the cell. The cell contents are then verified by a read back of the 0 values. The procedure is repeated by writing all 1s to the cell and reading the contents back. If a transition failure from 1 to 0 is a failure mode of concern, an additional write and read of 0s can be performed. Finally, original contents of the cell are restored (see Reference [35], Section 4.2.1). The test is effective at detecting stuck-at and transition failures but cannot detect most soft errors, addressing faults and linked cell faults.

NOTE 2 The test is often implemented in the background with interrupt suppression during the test of each individual location.

NOTE 3 Because the implementation includes a read of a just written value, optimizing compilers have a tendency to optimize out the test. If an optimizing compiler is used, good design practice is to verify the test code by an assembler-level code inspection.

NOTE 4 Some RAMs can fail such that the last memory access operation is echoed back as a read. If this is a plausible failure mode, the diagnostic can test two locations together, first writing a 0 to 1 and then a 1 to the next and then verifying a 0 is read from the first location.

5.1.13.6 Parity bit

NOTE This technique/measure is referenced in Table 31 and Table 32 of this document.

Aim: To detect a single corrupted bit or an odd number of corrupted bits failures in a word (typically 8 bits, 16 bits, 32 bits, 64 bits or 128 bits).

Description: Every word of the memory is extended by one bit (the parity bit) which completes each word to an even or odd number of logical 1s. The parity of the data word is checked each time it is read. If the wrong number of 1s is found, a failure message is produced. The choice of even or odd parity ought to be made such that, whichever of the zero word (nothing but 0s) or the one word (nothing but 1s) is the more unfavourable in the event of a failure, then that word is not a valid code.

Parity can also be used to detect addressing failure, when the parity is calculated for the concatenation of the data word and its address. Otherwise, for addressing failures, there is a 50 % probability of detection for random data returned (for example, address line open or address line shortened to another address line such that an average of the two cells is returned). The coverage is 0 % if the addressing error leads to a completely different cell selected.

For RAM cell write-enable failure, parity can detect 50 % of failures if the cell is unable to be initialized. The coverage is 0 % if the write-enable failure affects entire cell after it has been initialized.

5.1.13.7 RAM March test

NOTE 1 This technique/measure is referenced in Table 32 of this document.

Aim: To detect predominantly persistent bit failures, bit transition failures, addressing failures and linked cell failures.

Description: A pattern of 0s and 1s is written into the cells of memory in a specific pattern and verified in a specific order.

A March test consists of a finite sequence of March elements; while a March element is a finite sequence of operations applied to every cell in the memory array before proceeding to the next cell. For example, an operation can consist of writing a 0 into a cell, writing a 1 into a cell, reading an expected 0 from a cell, and reading an expected 1 from a cell. A failure is detected if the expected “1” is not read. The coverage level for linked cells depends on the write/read order.

Reference [35], Chapter 4, lists a number of different March tests designed to detect various RAM failure modes: stuck-at faults, transition faults (inability to transition from a one to a zero or a zero to a one but not both), address faults and linked cell faults. These types of tests are not effective for soft error detection.

NOTE 2 These tests can usually only be run at initialization or shutdown.

5.1.13.8 Running checksum/CRC

NOTE This technique/measure is referenced in Table 32 of this document.

Aim: To detect single bit, and some multiple bit, failures in RAM.

Description: A checksum/CRC is created by a suitable algorithm which uses each of the words in a block of memory. The checksum is stored as an additional word in RAM. As the memory block is updated, the RAM checksum/CRC is also updated by removing the old data value and adding in the new data value to be stored to the memory location. Periodically, a checksum/CRC is calculated for the data block and compared to the stored checksum/CRC. If a difference is found, a failure message is produced. The probability of a missed detection is 1/size of checksum/CRC if a random result is returned. DC can be reduced as memory size increases.

5.2 Analogue/mixed signal components

5.2.1 About analogue and mixed signal components

As described in 4.2, a semiconductor component is structured in parts and sub-parts. If the signals that are handled in an element (component, part or sub-part) are not limited to digital states, this element is seen as analogue element. This is the case for each measurement interfaces to the physical world, including sensors, actuator outputs, and power supplies.

For analogue components, each element is analogue and no digital element is included. Mixed signal components consist of at least one analogue element and one digital element. Since analogue and digital elements require different methodologies and tooling for design, layout, verification and testing, it is recommended to clearly divide the analogue and digital blocks. This can result in a variety of configurations ranging from analogue dominated components with digital support blocks (e.g. digitally configurable voltage regulators or auto zeroing amplifiers) to microcontrollers with a few mixed signal peripherals (e.g. analogue to digital converters and phase locked loops). A hierarchy of a typical mixed signal component including exemplary parts and sub-parts is shown in Figure 16.

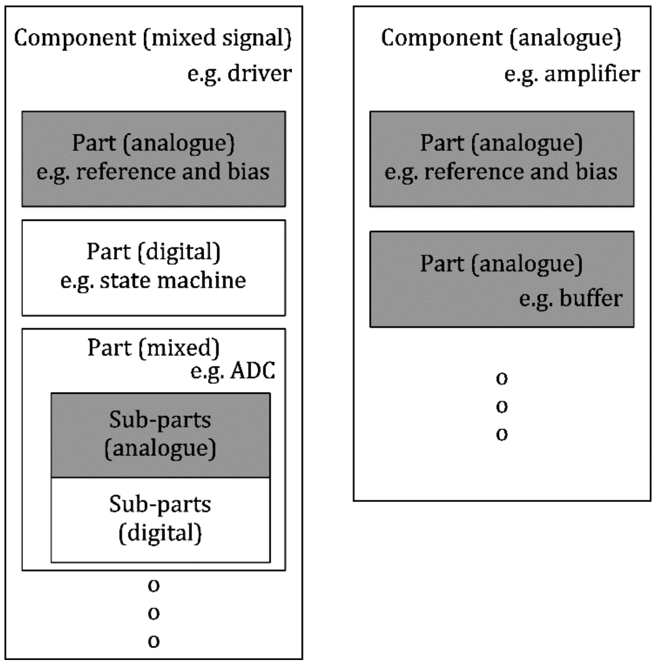


Figure 16 — Generic hierarchy of analogue and mixed signal components

It can be helpful to divide a mixed signal component in a way that simplifies the safety analysis. For an easy definition of fault models and failure modes, the analogue part boundaries can be defined by their function. Additionally, each element that has freedom from interference or independence requirements (e.g. redundant paths or functions and corresponding diagnostic functions) is separated by part or sub-part boundaries. There are several additional criteria to further divide a mixed signal element (component or part) into sub elements (part or sub-part):

— Signal flow;

EXAMPLE 1 Mixed signal control loops can consist of feedback ADC, digital regulator and output driver.

— Connectivity;

EXAMPLE 2 Reference and bias circuits can serve multiple analogue blocks and oscillators can serve multiple digital or mixed signal blocks.

— Different technologies;

EXAMPLE 3 HV switch is a DMOS transistor while the gate driver can use conventional MOS devices.

NOTE One benefit for a separation of these parts is that they can have failure rates with different orders of magnitude or different fault models.

— Different supply domains; and

EXAMPLE 4 Feedback DAC can be supplied with different supplies than the other mixed signal block output driver.

— Other criteria for partitioning.

EXAMPLE 5 High versus low frequency sub-parts.

The level of detail of the analysis and granularity of parts and sub-parts are determined by the relevant safety requirements, safety mechanisms and the need to provide evidence of independence of safety mechanisms. A higher granularity does not necessarily result in a significant benefit for the safety analysis.

5.2.2 Analogue and mixed signal components and failure modes

5.2.2.1 About failure modes

The failure modes affecting a hardware element depend on its function. The failure mode distribution depends on the hardware element implementation.

NOTE 1 The implementation includes both the actual circuit and the technology process used.

The classification of a failure mode depends on the functional and safety requirements of the system integrating the element. Based on the integration, a specific failure mode can or cannot lead to a violation of a safety requirement. Table 35 identifies possible failure modes that can be of concern for an analogue and mixed signal part or sub-part. The table can be used to extend the list of failure modes reported in ISO 26262-5:2018, Annex D.

The failure modes identified in Table 35 as well as the mentioned parts and sub-parts, are a general reference and can be adjusted on a case by case basis. Failure modes for analogue circuits can be derived by applying key words as mentioned in 4.3.2.

The actual failure mode list used in a specific project can be adjusted (adding or removing failure modes) based on the specific implementation details or on the level of granularity deemed necessary for the analysis.

It is noted that the relevance of the failure modes, including but not limited to the ones listed in Table 37 are dependent on the context of the function to be analysed.

EXAMPLE 1 The obvious failure modes of a voltage regulator are over-voltage and under-voltage. These failure modes can be detected by an over voltage and under voltage (OV/UV) monitor as described in 5.2.4.2.

Besides the obvious failure modes reported in the above example, it is important to identify each relevant failure mode in order to perform a complete and thorough analysis.

EXAMPLE 2 If a voltage regulator is used as a sensor supply or as an ADC reference supply, then the failure modes affecting the stability and the accuracy of the output voltage, even within the OV/UV thresholds, can be critical. Output voltage with insufficient accuracy and output voltage oscillation within the OV/UV thresholds can be mitigated by using appropriate measures. An independent ADC (internal or external) can be used to periodically measure the regulator output voltage with the required accuracy to detect those failure modes.

EXAMPLE 3 If a voltage regulator is used as a supply for a radio frequency (RF) module which has tight supply voltage ripple requirements, the prevention of fluctuation on the regulated output voltage caused by input voltage variations (i.e. the PSRR, power supply rejection ratio) is an important feature. Failure modes like output voltage oscillation within the OV/UV (i.e. ripple) limits and spikes affecting the regulated voltage can be relevant. A low pass filter as described in 5.2.4.8 can be used to mitigate these failures.

EXAMPLE 4 If a voltage regulator is used as an MCU core supply is sensitive to output voltage drops during start-up (power-up) due to in-rush current exceeding regulator load current and/or current limit, a too fast start-up time can be critical. A proper regulator soft-start function can be used to mitigate such failure.

If failure modes are classified as not safety-related, an argument is provided in the safety analysis to support the classification.

Given the variety of implementations and the lack of data available from the field and from theory, Table 35 does not give any indication about the quantitative impact of the listed failur e modes, i.e. the failure mode distribution. It is the responsibility of the safety analyst to identify such quantitative data. An example is given in 5.2.3.3.

NOTE 2 Even though it is known that a single physical root cause can lead to more than one failure mode, it is a common simplification that the sum of the distribution of each failure mode is 100 % which is a prerequisite for the quantitative analysis.

NOTE 3 Transient failure modes are considered if they are relevant, for example if for the technology in use the risk of single-event effects (SEE) is not negligible, see 5.2.2.3.

Table 35 — Possible failure modes of analogue and mixed signal parts and sub-parts

Part / sub-part	Short description	Failure modes
	Regulators and Power stages	
Voltage regulators (linear, SMPS, etc.)	Hardware part/sub-part that maintains the voltage of a power source within a prescribed range that can be tolerated by elements using that voltage.	Output voltage higher than a high threshold of the prescribed range (i.e. over voltage – OV) Output voltage lower than a low threshold of the prescribed range (i.e. under voltage – UV) Output voltage affected by spikes[b] Incorrect start-up time (i.e. outside the expected range) Output voltage accuracy too low, including drift[c] Output voltage oscillation[a] within the prescribed range Output voltage affected by a fast oscillation[a] outside the prescribed range but with average value within the prescribed range Quiescent current (i.e. current drawn by the regulator in order to control its internal circuitry for proper operation) exceeding the maximum value
Charge pump, regulator boost	Hardware part/sub-part that converts, and optionally regulates, voltages using switching technology and capacitive-energy storage elements, and maintains a constant output voltage with a varying voltage input.	Output voltage higher than a high threshold of the prescribed range (i.e. over voltage – OV) Output voltage lower than a low threshold of the prescribed range (i.e. under voltage – UV) Output voltage affected by spikes[b] Incorrect start-up time (i.e. outside the expected range) Quiescent current (i.e. current drawn by the regulator in order to control its internal circuitry for proper operation) exceeding the maximum value

Part / sub-part	Short description	Failure modes
High-side/Low-side (HS/LS) driver	Hardware part/sub-part that applies voltage to a load in a single direction: high side driver to connect the load to high rail, low side driver to connect the load to low rail.	HS/LS driver is stuck in ON or OFF state HS/LS driver is floating (i.e. open circuit, tri-stated) HS/LS driver resistance too high when turned on HS/LS driver resistance too low when turned off HS/LS driver turn-on time too fast or too slow HS/LS driver turn-off time too fast or too slow
Half-bridge driver or full-bridge (H-bridge) driver	Hardware part/sub-part that can apply voltage across a load in either direction. A half-bridge driver is built with two drivers (one HS and one LS driver). An H-bridge (or full-bridge) driver is built with four drivers (two HS and two LS drivers)	HS/LS driver is stuck in ON or OFF state HS/LS driver is floating (i.e. open circuit, tri-stated) HS/LS driver ON resistance too high when turned on HS/LS driver OFF resistance too low when turned off HS/LS driver turn-on time too fast or too slow HS/LS driver turn-off time too fast or too slow 'Dead time' is too short (i.e. when turning off high-side driver and turning on low-side driver, or when turning off low-side driver and turning on high-side driver) 'Dead time' is too long
High-side/Low-side pre-driver	Hardware part/sub-part driving a gate of an external FET that is used as a HS or LS driver.	HS/LS pre-driver is stuck in ON or OFF states HS/LS pre-driver output voltage/current too high or too low HS/LS pre-driver is floating (i.e. open circuit, tri-stated) HS/LS pre-driver slew rate too slow or too fast

2361

Table 35 *(continued)*

Part / sub-part	Short description	Failure modes
Analogue to digital and digital to analogue converters [d]		
N bits digital to analogue converters (DAC) [d]	Hardware part/sub-part converting digital data coded on "N bits" into an analogue signal (voltage or current).	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Offset error (not including stuck or floating conditions on the outputs, low resolution) Linearity error with monotonic conversion curve not including stuck or floating conditions on the outputs, low resolution Full-scale gain-error not including stuck or floating conditions on the outputs, low resolution No monotonic conversion curve Incorrect settling time (i.e. outside the expected range) Oscillation[a] of the output signal including drift[c]
N bits analogue to digital converters (N-bit ADC) [d]	Hardware part/sub-part converting a continuous-time and continuous-amplitude analogue signal (i.e. a voltage value) to a discrete-time and discrete-amplitude digital signal coded on "N bits."	One or more outputs are stuck (i.e. high or low) One or more outputs are floating (i.e. open circuit) Accuracy error (i.e. Error exceeds the LSBs) Offset error not including stuck or floating conditions on the outputs, low resolution No monotonic conversion characteristic (i.e. given two input analogue voltage V1>V2, the correspondent digital values are D1<D2) Full-scale error not including stuck or floating conditions on the outputs, low resolution Linearity error with monotonic conversion curve not including stuck or floating conditions on the outputs, low resolution Incorrect settling time (i.e. outside the expected range)
Oscillators and clock generators		
Oscillator	Hardware part/sub-part generating a periodic, oscillating signal. It can be used as clock in a digital circuit.	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect output signal swing (i.e. outside the expected range) Incorrect frequency of the output signal (i.e. outside the expected range, including harmonics when applicable, for instance EMC emissions) Incorrect duty cycle of the output signal (i.e. outside the expected range) Drift[c] of the output frequency Jitter too high in the output signal

Part / sub-part	Short description	Failure modes
Phase locked loop (PLL)	Hardware part/sub-part controlling an oscillator in order to generate a square wave signal that maintains a constant phase angle (i.e. lock) on the frequency of an input, or reference signal. It can be used as clock in a digital circuit.	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect frequency of the output signal (i.e. outside the expected range, including harmonics when applicable, e.g. EMC emissions) Incorrect duty cycle of the output signal (i.e. outside the expected range) Drift[c] of the output frequency Jitter too high in the output signal Loss of lock condition (i.e. phase error, output clock not in sync with input clock not leading to incorrect frequency and incorrect duty cycle Missing pulse in the output signal Extra pulse in the output signal
Generic		
Operational amplifier and buffer	Hardware part/sub-part integrating a DC-coupled high-gain voltage amplifier with a differential input and, usually, a single-ended output.	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect gain on the output voltage (i.e. outside the expected range) Incorrect offset on the output voltage (i.e. outside the expected range) Incorrect output dynamic range (i.e. outside the expected range) Incorrect input dynamic range (i.e. outside the expected range) Output voltage accuracy too low, including drift[c] Output voltage affected by spikes[b] Output voltage oscillation[a] Settling time of the output voltage too low
Analogue switch	Hardware part/sub-part capable of switching or routing analogue signals based on the level of a digital control signal. Commonly implemented using a "transmission gate".	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit or tri-stated) Offset too high affecting the output signal Resistive or capacitive coupling between control signal and output signal including crosstalk Attenuation of the output signal Drift[c] affecting the output signal Spikes[b] affecting the output signal , e.g. during switching

2362

2363

2364 **Table 35 *(continued)***

Part / sub-part	Short description	Failure modes
Voltage/Current comparator	Hardware part/sub-part comparing an input analogue signal with a predefined threshold (i.e. voltage or current constant value) and producing a binary signal at the output; the output depends on which is higher between the input signal and the threshold and it remains constant as the difference between them stays with the same polarity.	Voltage/Current comparator not triggering when expected Voltage/Current comparator falsely triggering Output is stuck (i.e. high or low) Output is floating (i.e. open) Oscillation[a] of the output
Sample & hold	Hardware part/sub-part sampling the voltage of a continuously varying analogue input signal and holding its value at a constant level for a specified minimum period of time.	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect sampling leading to gain/offset error on output voltage dependent on input signal Incorrect gain on the output voltage (i.e. outside the expected range) Incorrect offset on the output voltage (i.e. outside the expected range) Incorrect output dynamic range (i.e. outside the expected range) Incorrect input dynamic range (i.e. outside the expected range) Output voltage accuracy too low during hold phase, including drift[c] Output voltage during hold phase affected by spikes[b] Output voltage oscillation[a] during hold phase Output does not settle sufficiently accurate during hold time

Part / sub-part	Short description	Failure modes
Analogue multiplexer	Hardware part/sub-part consisting of multiple analogue input signals, multiple control inputs and one output signal. .	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect channel selection Offset affecting the output signal too high Resistive or capacitive coupling among input channels and output signal including crosstalk Resistive or capacitive coupling among selectors and output signal including crosstalk Incorrect output dynamic range (i.e. outside the expected range) Attenuation of the output signal Drift[c] affecting the output signal Spikes[b] affecting the output signal (i.e. during switching)
Voltage references	Hardware part/sub-part producing a constant DC (direct-current) output voltage regardless of variations in external conditions such as temperature, barometric pressure, humidity, current demand, or the passage of time.	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect output voltage value (i.e. outside the expected range) Output voltage accuracy too low, including drift c Output voltage affected by spikes[b] Output voltage oscillation[a] within the expected range Incorrect start-up time (i.e. outside the expected range)
Passive network	Hardware part/sub-part consisting of a network of passive devices (resistor and capacitor) providing a specific low pass transfer function	Output is stuck (i.e. high or low) Output is floating (i.e. open circuit) Incorrect output dynamic range (i.e. outside the expected range) Incorrect attenuation of the output signal (i.e. outside the expected range) Incorrect settling time (i.e. outside the expected range) Drift[c] affecting the output signal Oscillation[a] affecting the output signal (i.e. due to crosstalk, coupling or parasitic effects) Spikes[b] affecting the output (i.e. due to crosstalk, coupling or parasitic effects)

Table 35 *(continued)*

Part / sub-part	Short description	Failure modes
Current source (including bias current generator)	hardware part/sub-part delivering or absorbing a current (i.e. reference current) which is independent of the voltage across it. It typically includes multiple branches which are routed to other circuits requiring a reference or bias current.	One or more outputs are stuck (i.e. high or low) One or more outputs are floating (i.e. open circuit) Incorrect reference current (i.e. outside the expected range) Reference current accuracy too low , including drift[c] Reference current affected by spikes[b] Reference current oscillation[a] within the expected range One or more branch currents outside the expected range while reference current is correct One or more branch currents accuracy too low , including drift[c] One or more branch currents affected by spikes[b] One or more branch currents oscillation[a] within the expected range

Part / sub-part	Short description	Failure modes

[a] An oscillation is an instability of the part/sub-part caused by internal failure, e.g. regulation loop failures, lower or negative hysteresis for a comparator, etc.. Oscillation includes any repetitive voltage and current variation (i.e. periodic pulse).

[b] A spike is a non-repetitive variation on the output voltage or current, i.e. pulse due to load jumps, etc.

[c] Drift is a slow and continuous variation of a parameter (i.e. current, voltage, threshold, etc.) outside the expected range reported into the circuit specification. Slow variation means slower as compared to the FTTI. For example drift covers floating or stuck at open failure modes.

[d] Several of the failure modes reported for the ADC or DAC can be grouped into two main sets: static error and absolute accuracy (total) error. Static errors are errors that affect the accuracy of a converter when it is converting static (DC) signals and can be completely described by four terms: offset error, gain error, integral nonlinearity, differential nonlinearity.

NOTE 1 Each term can be expressed in LSB units or sometimes as a percentage of the full scale range (FSR). For example, an error of ½ LSB for an 8-bit converter corresponds to 0,2 % FSR.

NOTE 2 The absolute accuracy (total) error is the maximum value of the difference between an analogue value and the ideal mid-step value. It includes offset, gain, and integral linearity errors, and also the quantization error in the case of an ADC.

5.2.2.2 About safe faults

ISO 26262-10:2018, 8.1.7 states that safe faults can be faults of one of two categories:

— all n point faults with n > 2, unless the safety concept shows them to be a relevant contributor to a safety requirement, or

— faults that will not contribute to the violation of a safety requirement.

Analogue components are characterized by continuous (output) signal (function) regions and as such, tolerances must be taken into consideration when used in systems. The tolerances on analogue functions as specified as part of the safety requirements allocated to that analogue component can be less constrained than the actual tolerance of the analogue component itself. For this reason, the fraction of the failure mode that leads to parametric failure or drift, but which remains within these tolerance ranges is safe. An analogue component has therefore an inherent capability to tolerate a fault. These faults are safe faults.

EXAMPLE 1 A resistor is used to limit the current flowing through a specific branch. A failure in the accuracy of the resistor increasing its value (e.g. of 50 %) but not preventing the current limiting function would be a safe fault.

A specific fault in an element can have a different classification depending on the specific safety requirement considered. For more details see ISO 26262-5:2018.

Depending on the system configuration and the safety requirements some failure modes are not relevant, i.e. they cannot violate the requirements. In this case, these failure modes can be classified as safe: They contribute to the safety metrics increasing the failure rate of safe faults.

EXAMPLE 2 An output driver can have an output slope control to limit the rise and fall times of the output value for EMI purposes. If the slew rate is irrelevant for the violation of the safety goal, failures in this slope control would be safe faults.

EXAMPLE 3 If a voltage regulator is used to supply digital circuits only, failure modes affecting the stability and the accuracy of the output voltage within the OV/UV thresholds can be classified as safe.

5.2.2.3 About transient faults

As defined in ISO 26262-1:2018, a transient fault is a fault that occurs once and subsequently disappears. Soft errors such as Single Event Upset (SEU) and Single Event Transient (SET) are defined as transient faults. ISO 26262-5:2018, 8.4.7 states that transient faults are considered when shown to be relevant due, for instance, to the technology used and can be addressed either by a quantitative approach, specifying and verifying a dedicated target “single-point fault metric” value to them or by a qualitative rationale based on the verification of the effectiveness of the internal safety mechanisms implemented to cover these transient faults.

In terrestrial analogue circuits, transient faults are caused by alpha-particle or neutron hits or by electromagnetic interference such as power transients and crosstalk. They can cause SEU or even SET also called Analogue Single Event Transients (ASETs), such as transient pulses in operational amplifiers, comparators or reference voltage circuits.

Due to the intrinsic nature of analogue technology (in which transient or noise effects are considered by design), the susceptibility to transient faults is lower than in digital circuits by orders of magnitude. Therefore, the analysis of those effects can be limited in a first approximation to their digital part (e.g. the digital decimation filter of a sigma-delta ADC).

However in some cases, like in the early part of the conversion cycle of an ADC (see reference [28]) or in PLL (see reference [20]) or differential switched-capacitor circuits (see reference [10]), the vulnerability to soft-error can be high. In those cases, more detailed analyses are done and appropriate countermeasures are identified (see reference [1]).

For mixed signal components, the impact of soft errors in the digital part is considered as described in 5.1.7.2.

NOTE If more detailed analyses are needed in the analogue part, since SER evaluation by irradiation tests in analogue circuits is not a simple task, in those cases measurement is done mainly by analytical.

5.2.3 Notes about safety analysis

5.2.3.1 General

The examples and guidelines given in 5.1 are also valid for an analogue or mixed signal component. The following clauses describe some of the topics that can require additional clarification for an analogue or mixed signal component.

5.2.3.2 Level of granularity of analysis

One of the key aspects for the safety analysis of analogue elements is the proper identification of the level of hierarchy on which to base the analysis. On one hand, a lower level of granularity is beneficial as it allows for a better understanding of the failure modes and failure mode distributions. On the other, a higher level of granularity allows for a clear allocation of safety mechanisms. Analogue elements are often used to interface with physical objects making it useful to also consider mechanical characteristics and differentiate the failure modes accordingly.

As seen in ISO 26262-9:2018, Clause 8, qualitative and quantitative safety analyses are performed at the appropriate level of abstraction during the concept and product development phases. The level of abstraction can be consequently adjusted depending on the target of the analysis. Qualitative analysis is more suited to identify failure modes while quantitative analysis to quantify their failure rate and distribution.

Consider an example in which, a linear voltage regulator is monitored using a windowed voltage monitor. The voltage monitor is at the output of the regulator and is able to detect over-voltage conditions. If the output value, allowed in the working condition to fluctuate in a range around a nominal value, e.g. 1,2V +/- 0,12V, moves outside that range it is to be considered faulty. If the analysis focuses on the output of the regulator it can be relatively easy to discriminate between types of failures (e.g. safe because within allowed range, over or under voltage) and quantify the protection offered by the voltage monitor. However it is difficult to quantify the likelihood of each type of failure as required for metric computation. If the analysis goes inside the regulator and focuses, for instance, on faults of the bandgap it is easier to analyse propagation and likelihood of each failure of the regulator but not simple to quantify the protection that the external voltage monitor offers on the bandgap itself.

For the safety analysis, the type of safety mechanisms can drive the selection of the level of abstraction. If the safety mechanisms addressing analogue features are located at system or element level, going down in the block structure can lead to an overly complex analysis. The quantification of the failure mode distribution can require an investigation on lower levels of abstraction. For instance, applying an equal distribution to the failure modes of the linear voltage regulator can give less accurate results than applying an equal distribution to the blocks composing the linear voltage regulator as, for instance, the bandgap, the buffer, the driver, etc. With respect to terminology, in line with the classification described in 4.2, the linear voltage regulator is to be considered a part and the bandgap, the buffer, the driver, etc. sub-parts.

5.2.3.3 Deriving failure mode distributions for analogue

The failure distribution model is dependent on the circuit implementation and targeted process. Each supplier provides details on the failure mode distribution model used in the analysis.

EXAMPLE 1 A simple and pessimistic model can be used for the initial analysis, like considering only failure modes capable of violating a safety requirement (i.e. not a safe failure mode) and using a linear distribution for the defined failure modes; for instance if five failure modes are defined, each failure mode is allocated 20 % distribution.

NOTE 1 In the EXAMPLE 1 above, this analysis considers each applicable failure mode except those not capable of violating the safety requirement. Safe failure modes are not included in the computation.

If the analysis using such failure mode distribution model does not comply with the required Single-Point Fault and/or Latent-Fault metrics for the targeted ASIL level, the definition of the failure modes and related distribution is further refined.

EXAMPLE 2 Failure modes not capable of violating the safety requirement, i.e. safe failure modes, that are applicable to the circuit under analysis, are added in the computation with Fsafe=100 %

NOTE 2 The uniform failure mode distribution and the list of safe and not safe failure modes are considered in the FMEA example in 5.2.3.5.

EXAMPLE 3 A more detailed distribution for each failure mode can be considered based on area; if the area of the circuit or circuits identified as the root cause for the defined failure mode is 5 %, then the allocated failure mode distribution is 5 %.

Applicable failure modes and detailed failure mode distributions are justified according to the circuit implementation and its physical area and documented in the product safety case.

5.2.3.4 Example of failure rates estimation for an analogue component

A detailed example of failure rates estimation for an analogue component is described in Annex D of this part of ISO 26262.

5.2.3.5 Example of safety metrics computation

A detailed example of quantitative analysis and safety mechanisms determination for analogue components is described in Annex D of this part of ISO 26262.

5.2.3.6 Dependent failures analysis

As stated in ISO 26262-9:2018, 7.4.2, the analysis of dependent failures is performed on a qualitative basis because no general and sufficiently reliable method exists for quantifying such failures.

The steps reported in 4.7 are applicable also for analogue and mixed signal components. In the dependent failures analysis, there are aspects that can be clearly considered when addressing analogue components, parts or sub-parts.

Analogue circuits are by nature sensitive to noise and interference among different blocks or functions. For this reason, structures to guarantee sufficient independence by means of isolation and separation (e.g. by implementing barriers and/or guard-rings or placing circuits at certain distances or separating the power supply distribution and even the ground layer) are implemented for functional reasons. In fact, substrate, power supply and global signals like bias, clock or reset are often considered as a source of interference and special care is taken to reduce such effect. This good design practice, usually followed for functional reasons, provides benefits in terms of dependent failures avoidance.

Analogue circuits can be very sensitive to process variation resulting in mismatches in the device behaviour. To ensure the “same” transfer function of two blocks, as in the case of redundant parts, the symmetry of the design and physical layout is a key factor. In such cases, special attention is taken to ensure exactly the same layout of the two blocks including orientation, symmetrical placing, routing etc.; therefore diversity is not always a viable solution to improve the common cause failure avoidance for analogue circuits.

As a consequence of these aspects, the dependent failures initiators are often addressed by techniques ensuring isolation or separation instead of with techniques aiming to differentiate their effects.

In other cases, diversity can still be a valid technique to achieve the detection or avoidance of dependent failures. For instance, in a dual channel approach, using two diverse ADC architectures (e.g. successive approximation ADC and sigma delta ADC) can reduce significantly the probability of common cause failures.

5.2.3.7 Verification of architectural metrics computation

This clause is addressing a specific part of the safety analysis verification: the verification of the architectural safety metrics and in particular the fraction of safe faults and the failure mode coverage.

Possible approaches include:

— Expert judgment founded on an engineering approach given that any data, either qualitative or quantitative, is supported by rationale and relevant arguments included in the safety case ;

NOTE 1 In some cases, such arguments can be derived from the functional characterization of the hardware elements responsible for the claimed parameters. The aim of the functional characterization is the systematic failure avoidance and not the hardware random failure but, in some cases, it can be used as evidence to prove the level of coverage with respect to a specific failure mode: This is the case in which the aim of a safety mechanism is to detect 100 % of one of more failure modes and this capability is guaranteed by design.

EXAMPLE 1 A voltage monitor as described in 5.2.4.2 is a typical safety mechanism used to detect overvoltage and under-voltage failure modes affecting the voltage regulator. If, during the hardware design verification, the functional characterization of the voltage monitor shows that:

— any event leading to a regulated voltage outside the expected range defined in the specification for enough time to make the supplied hardware circuit malfunction is detected by the voltage monitor; and

— any event leading to a variation of the regulated voltage inside the range defined in the specification for any time does not prevent the correct behaviour of the hardware circuit supplied by the regulator;

then, such characterizations can be used as arguments to claim a detection equal to 100 % of the mentioned failure modes.

— As mentioned in 4.8, fault injection simulation during the development phase is a valid method to verify completeness and correctness of safety mechanism implementation with respect to hardware safety requirements and fault injection using design models can be successfully used to assist the verification. This method can be applied to analogue and mixed signal components; and

NOTE 2 The fault injection campaign can be limited to a subset of faults or failures that are judged to be critical in a specific case. The most critical failure modes are identified after considering their distribution, their claimed amount of safe faults, their claimed level of detection and the safety mechanisms or safety requirements responsible for those levels.

EXAMPLE 2A failure mode is deemed too complex for expert judgement. This specific failure mode is a candidate to be characterized using fault injection.

— A combination of both methods, i.e., fault injection which supports expert judgment by providing arguments and evidence for the cases judged more critical and /or addressable by fault injection method alone.

5.2.4 Examples of safety mechanisms

The following tables give a non-exhaustive list of examples of commonly used analogue safety mechanisms and complement the information contained in ISO 26262-5:2018, Annex D.

Some analogue safety mechanisms have a digital output signal which is used to control the reaction to a failure and bring the component to a safe state. In many cases, this information is stored so that it can be communicated through a digital interface. Other analogue safety mechanisms control or suppress a fault from resulting in the violation of a safety requirement and do not interface with the digital domain.

To comply with ISO 26262-5:2018, 8.4.8, the safety mechanisms described in the following tables can require additional measures to detect faults affecting them that, as dual-point faults, can lead to the violation of the safety goal.

The examples given from Table 36 to Table 39 are not exhaustive and other techniques can be used.

NOTE 1 It is not possible to give a general guidance on the DC because it strongly depends on the specific technology, type of circuit, use case etc.

NOTE 2 Evidence is provided to support the claimed diagnostic coverage.

Table 36 – Power supply

Safety mechanism/measure	See overview of techniques	Notes
Over and under voltage monitoring	5.2.4.2	Typically an analogue circuit with an output latched in a digital core.
Voltage clamp (limiter)	5.2.4.3	Typically used to suppress voltage transients or spikes.
Over-current monitoring	5.2.4.4	Typically an analogue circuit with an output latched in digital core.
Current limiting	5.2.4.5	Typically an analogue circuit with feedback to an analogue control loop (e.g. to disable regulator main pass element).
Power on reset	5.2.4.6	Functional block which keeps the circuit in a known initialized state until power supply rails and/or the clock signal are stable.

Table 37 — Analogue I/O

Safety mechanism/measure	See overview of techniques	Notes
Resistive pull up/down	5.2.4.1	Typically used on input signals to avoid floating conditions due to pin failure or external pin interconnect failure.
Filter	5.2.4.8	Analogue or digital circuit, typically used to suppress high frequency signal variation, like an output from analogue over & under voltage monitoring circuit.

Table 38 — Component

Safety mechanism/measure	See overview of techniques	Notes
Analogue watchdog	5.2.4.7	Typically a monostable circuit used to monitor proper operation of an oscillator.
Thermal monitor	5.2.4.9	Typically an analogue circuit with an output latched in digital core, or feedback to an analogue circuit control loop (e.g. to disable affected circuit).
ADC monitoring	5.2.4.11	An analogue circuit typically controlled and evaluated by a digital circuit.
Analogue BIST	5.2.4.10	Typically an analogue circuit controlled by a digital circuit that verifies correct functionality of analogue safety mechanisms like under/over voltage monitoring, current limit protection and thermal protection circuits.

Table 39 — Analogue to digital converter

Safety mechanism/measure	See overview of techniques	Notes
ADC attenuation detection	5.2.4.12	Typically an analogue circuit controlled by a digital circuit that validates the ADC conversion path by measuring a known and stable signal value.
Stuck on ADC channel detection	5.2.4.13	Typically an analogue circuit controlled by a digital circuit that validates the ADC conversion path by measuring a known and stable signal value.

5.2.4.1 Resistive pull up/down

Aim: To define a default voltage for a circuit node.

Description: A resistor is connected from a circuit node to either a supply voltage or ground to define a default voltage in the event that the driving signal becomes disconnected/high impedance. Commonly used on I/O pins.

EXAMPLE An un-driven or disconnected device/module input pin would be at an unknown voltage level. A pull-up resistor to the I/O supply voltage (or module supply voltage) or pull-down resistor to ground is used to keep the input at a known voltage level. The circuit itself could be a passive resistor or an active circuit like a current mirror.

5.2.4.2 Over & under voltage monitoring

Aim: To detect, as early as possible, when a regulated voltage is outside the specified range.

Description: The regulated voltage is compared via a differential input pair to a low and/or a high analogue reference voltage representing the limits of the specified operating range. The monitor output will change state when the regulated voltage is outside of the defined voltage window indicating a fault.

EXAMPLE A window comparator is used to monitor the output of a LDO regulator with reference voltages set to the minimum and maximum specified voltage levels in regulation.

5.2.4.3 Voltage clamp (limiter)

Aim: To prevent the voltage of a circuit node from exceeding the maximum voltage that can be safely supported.

Description: A voltage clamp limits the positive and/or negative voltage of a circuit node to an acceptable level determined by system and/or device process capability. Voltage clamps can be biased or unbiased. Unbiased clamps typically use Zener diodes to define the reference voltage while biased clamps use a voltage source in combination with specialized diodes (Zener, Schottky) to define the acceptable voltage level. Voltage clamps are typically used to protect against transient events.

EXAMPLE An ESD protection circuit is a specialized voltage clamp typically implemented on I/O pins. It is designed to shunt the energy of a high voltage electrostatic discharge on the I/O pins away from the internal circuitry to ensure that internal circuitry is not exposed to excessive voltage levels during the ESD event.

5.2.4.4 Over-current monitoring

Aim: To detect, as early as possible, when the output current exceeds a certain value.

Description: The implementation of over-current monitoring can vary. A typical approach for a voltage regulator circuit with an MOS output device is to add a sense FET in parallel with a regulator main FET. The sense FET current, which is proportional to the main FET current, flows across a sense resistor. The voltage drop across the sense resistor is amplified and monitored by a voltage monitor.

NOTE The output of an over-current monitor is a digital output which is subsequently used as feedback to an analogue circuit control loop, and/or latched in a digital core which interfaces to the control and/or status monitoring circuits.

5.2.4.5 Current limiter

Aim: To limit output current to a maximum level in order to maintain a safe operating area of the output device and prevent electrical overstress.

Description: A closed loop system using negative feedback from a current monitor to reduce the drive to the output device thereby limiting the output current.

5.2.4.6 Power on reset

Aim: To hold the outputs of a system in a known state (typically off) until internal nodes have stabilized upon power up or power reset conditions.

Description: Typically, a bandgap-derived voltage reference is compared to an attenuated supply voltage in order to detect the minimum specified supply voltage which will ensure correct operation. Hysteresis is typically required to prevent oscillation as the attenuated supply voltage exceeds the reference voltage.

EXAMPLE An under-voltage monitor is a mechanism used to detect and drive power-on reset.

5.2.4.7 Analogue watchdog

Aim: To monitor proper operation of an oscillator.

Description: Typically implemented with a monostable circuit (one shot) which is reset on each cycle of the oscillator. If an oscillator transition does not occur within a specified time peri od defined by the monostable circuit, a fault signal is produced.

5.2.4.8 Filter

Aim: A filter can be used in multiple ways as a safety mechanism and depends upon the safety requirement under consideration including:

EXAMPLE 1 A bypass capacitor can be used to suppress voltage transients. An RC time constant is used to qualify whether the duration of a fault which violates a safety requirement is within the FTTI.

EXAMPLE 2 A digital de-glitch circuit can be used to filter level shifted analogue voltage comparator outputs. The de-glitch time duration is defined by the minimum signal transient duration that has to be detected as a valid voltage fault condition.

5.2.4.9 Thermal monitor

Aim: To detect when circuit temperature exceeds a specified limit.

Description: Typically, a PTAT (proportional to absolute temperature) voltage is compared to a temperature independent reference voltage usually derived from a bandgap. The comparator will generate a fault signal when the PTAT voltage exceeds the reference voltage.

5.2.4.10 Analogue Built-in Self-Test (Analogue BIST)

Aim: Typically, to verify correct operation of diagnostic circuits and increase the detection of latent faults.

Description: The implementation of analogue BIST varies according to the diagnostic function to be verified. Analogue BIST typically involves exercising diagnostic circuits into and out of fault scenarios by injecting currents or voltages into the diagnostic circuit to ensure the diagnostic circuit can switch to both faulted and non-faulted states.

5.2.4.11 ADC monitoring

Aim: To measure an analogue signal by means of digital conversion with an output processed/evaluated in the digital core as an independent/ redundant analogue signal monitor.

Description: A critical analogue signal for which accuracy is relevant is converted in a digital code by means of an independent ADC (e.g. located outside the component or, at least biased by an independent source). The digital code is then processed by the CPU or an equivalent digital machine in order to determine if the original analogue signal has the required performance in terms of accuracy and static and dynamic behaviour. The frequency of the sampling and the resolution of the ADC and digital processing define which failure modes can be detected and to what accuracy.

5.2.4.12 ADC attenuation detection

Aim: To detect incorrect conversion of an analogue signal into its digital interpretation.

Description: Upon each background conversion loop, the element performs the conversion of the internal V_{mid} voltage both with and without the selectable attenuation switched in. The conversion results are stored respectively in separate SPI fields. A mathematical operation of dividing the attenuated result by the non-attenuated result verifies that the attenuation factor is within specified limits.

5.2.4.13 Stuck on ADC channel detection

Aim: To detect stuck on faults affecting the input signal to be converted by the ADC

Description: The element provides a multiplexer channel with series resistor RPOST, which is selected only when converting the test voltage channels (V_{high}, V_{low}, V_{mid}), and RPOST is otherwise bypassed. The value of RPOST is chosen such that a stuck-on channel within the post-buffer mux pulls one or more of the test voltage channels out of the expected voltage range.

EXAMPLE Each software loop, the MCU reads the ADC conversion results for the V_{hig}h, V_{low} and V_{mid} component ADC channels over SPI, and compares them against fixed detection thresholds.

5.2.5 About avoidance of systematic faults during the development phase

Analogue and mixed signal components are developed based on a standardised development process.

The general requirements and recommendations related to hardware architecture and detailed design are defined in ISO 26262-5:2018, Clause 7.

The guideline in 5.1.9 applies to the analogue and mixed signal components well if:

- Table 30 is replaced by Table 40; and
- the usage of 3rd party validated macro blocks and to comply with each constraint and procedure defined by the macro core provider, if practicable, is restricted to hard cores only.

NOTE Wear and aging are considered during development with proper verification and validation procedures.

Table 40 — Examples of measures to avoid systematic failures in analogue and mixed signal components

ISO 26262-5:2018 Clause	Design phase	Technique/Measure	Aim
6.5.1 hardware safety requirements specification	Specification	Using an appropriate requirement management tool	To streamline the identification and tracking of the safety requirements for the hardware element.
6.5.2 hardware/software interface specification		Using a model to describe hardware/software interface for critical elements	To reduce the risk of misinterpretation and to ensure consistency between hardware and software design.
7.5.1 hardware design specification		Using an appropriate tool to allocate requirements to hardware design	To streamline the identification and tracking of the design specification for the hardware element.
7.4.1.6 Properties of modular hardware design	Design	Use of modular, hierarchical, and simple design	The description of the circuit's functionality is structured in such a fashion that it is easily to understand. i.e. circuit function can be intuitively understood by its description without simulation efforts
7.4.1.6 Properties of modular hardware design		hardware design using schematics	Schematic entry is the method typically used for analogue circuitry.
7.4.4 Verification of hardware design		Behavioural model simulation for critical elements	Behavioural models are simplified models of the design. Behavioural modelling for analogue circuits allows for the evaluation of functionality in an early design stage (e.g. to prove the design concept) and a reduction in simulation time.
7.4.4 Verification of hardware design		Transistor level simulation	Simulation on transistor level is the method used to verify and validate dedicated critical functionalities of analogue circuits where simulation time is feasible.
7.4.4 Verification of hardware design		Safe operating area (SOA) checks done by design review and/or tools	An analogue circuit is composed of devices with different current/voltage capabilities. SOA checking ensures that each device will work safely within its specific operational area according to its technology.
7.4.4 Verification of hardware design		Corner simulations (i.e. technology process and environmental conditions spread)	In order to ensure block-level functionality, simulations are performed which take the spread of process parameters and environmental conditions into account.
7.4.4 Verification of hardware design		Monte Carlo simulations of most sensitive blocks	In order to ensure block-level functionality of critical circuits, the effect of on-chip process spread is simulated using a statistical approach (i.e. Monte Carlo simulations)
7.4.4 Verification of hardware design		Mixed mode simulations for critical elements	To ensure the correctness of critical elements, e.g. analogue to digital interfaces, analogue/digital closed loop control, digital circuits are simulated in the analogue domain.
7.4.4 Verification of hardware design		Requirement Driven Verification	All functional and safety-related requirements are verified. To be shown via traceability between specification and verification plan
7.4.4 Verification of hardware design		Design for testability	Specific hardware structures (e.g. test modes, multiplexers) are included into the design and layout in order to test otherwise inaccessible circuit nodes and improve the test coverage
7.4.2.4 Robust design principles		Application of schematic design guidelines	Manual checks
7.4.4 Verification of hardware design		Application of schematic checkers	To perform automatic checks for example on interconnections or on the selection of the proper devices as a function of polarities. For example SOA (Safe Operating Area) checker
7.4.4 Verification of hardware design		Documentation of simulation results	Documentation of each data needed for a successful simulation in order to verify the specified circuit function
7.4.4 Verification of hardware design		Schematic design inspection or walk-through	Design review usually includes inspection or walk-through.
7.4.4 Verification of hardware design		Application and validation of hard-core (reused schematic design and/or layout)	Usage of an already proven schematic or layout.
7.4.4 Verification of hardware design		Verification for behavioural models (if used) against the transistor level description	Cross check between behavioural model and the transistor level schematic design by simulation
7.4.4 Verification of hardware design		Simulation of netlist with parasitics extracted from layout for critical elements	Back-annotated netlist simulated by analogue simulator

2672 **Table 40 *(continued)***

ISO 26262-5:2018 Clause	Design phase	Technique/Measure	Aim
7.4.4 Verification of hardware design		Verification of netlist with parasitics extracted from layout against the schematic netlist for critical elements	Back-annotated netlist is checked against the schematic description in terms of simulation results in order to consider parasitic layout effects.
7.4.4 Verification of hardware design	**Design**	Layout inspection or walk-through (avoid cross talk between noisy and sensitive nets; avoid signal path with minimum width; use of multiple contacts/vias to connect layers)	The layout of analogue circuits is mainly done manually (automation is very limited with respect to the analogue blocks) and so layout inspection is crucial. The design review usually includes layout inspection or walk-through.
7.4.4 Verification of hardware design		Design rule check (DRC)	The layout of analogue circuits is mainly done manually (automation is very limited with respect to the analogue blocks) and so design rule checking is more crucial than in the digital domain.
7.4.4 Verification of hardware design		Layout versus schematic check (LVS)	The layout of analogue circuits is typically done manually (automation is very limited compared to the analogue blocks) and so checking layout versus schematic is more crucial than in the digital domain.
7.4.4 Verification of hardware design	**hardware design verification**	Development by hardware prototyping	Verification of implemented functions by prototype (e.g. test chips, boards), can check particular points of the hardware design where design review is not sufficient.
6.5.3 hardware safety requirement verification report	**Verification**	hardware safety requirement verification report	Provide evidence of consistency with hardware specification, completeness and correctness
10.5.1 hardware integration and testing activities	**hardware integration testing**	Verification of the completeness and correctness of the design implementation on the component level	Perform component tests and reports
7.4.5 Production, operation, service and decommissioning 9.4.2.5 Dedicated measures	**Safety-related special Characteristics during Chip production**	Determination of the achievable test coverage of production test	Evaluation of the test coverage during production test with respect to the safety-related aspects of the component.
7.4.5 Production, operation, service and decommissioning 9.4.2.5 Dedicated measures		Determination of measures to detect and cull early failures	Assurance of the robustness of the manufactured component. In most, but not every process, gate oxide integrity (GOI) is the key early life failure mechanism. There are multiple methods of screening early life GOI failures including high temp/high voltage operation (Burn-In), high current operation and voltage stress however these methods could have no benefit if GOI is not the primary contributor to early life failures in a process.
7.4.5 Production, operation, service and decommissioning 10 Hardware integration and testing	**Qualification of hardware component**	Definition and execution of qualification tests like Brown-out test , High Temperature Operating Lifetime (HTOL) test and functional test-cases, Specification of requirements related to production, operation, service and decommission hardware integration and testing report	For an analogue component with integrated brown-out detection, the component functionality is tested to verify that the outputs of the analogue circuit are set to a defined state (for example by stopping the operation of the analogue circuits in the reset state) or that the brown-out condition is signalled in another way (for example by raising a safe-state signal) when any of the supply voltages monitored by the brown-out detection reach a low boundary as defined for correct operation. For an analogue component without integrated brown-out detection, the analogue functionality is tested to verify if the analogue circuit sets its outputs to a defined state (for example by stopping the operation of the analogue circuit in the reset state) when the supply voltages drop from nominal value to zero. Otherwise an assumption of use is defined and an external measure is considered.

2673

2674

5.2.6 Example of safety documentation for an analogue/mixed-signal component

Analogue and mixed-signal components are predominantly developed within a distributed development due to the specific nature of their functionality.

Guidelines reported in 5.1.11 for digital components can be used as a reference for the safety work products to be exchanged, however, an adaptation to the different development approach can be necessary.

— The DIA between the component manufacturer and the end user specifies which documents are to be made available from each party as well as the level of work-share between the parties; and

— The safety requirement specification defines the expected functionality of the compon ent. It is critical that such specifications are carefully compiled by the end user, according to ISO 26262-8:2018, Clause 6, to ensure that correct functionality is understood by each supplier in the distributed development. A description about the usage of the elements of the component as well as identification of pre-defined on-chip/off-chip safety mechanisms is important to allow a proper safety analysis at a system or element level (e.g. to allow fault classification into safe, potential to violate a safety goal, etc., for each safety goal considered).

NOTE 1 If the component is developed out of context, the requirements derived from the technical safety concept are replaced by assumptions of use.

Documentation describing the capabilities of analogue an d mixed signal components are listed below:

— The results of the checks against the applicable requirements of ISO 26262 including confirmation measures reports (if applicable);

— Safety analysis results as per agreement; (these can be simply raw failures of the component, their distribution and diagnostic coverage offered from the specified safety mechanisms or a full FMEA for different safety requirements);

— Information regarding the calculation of the failure rate (e.g. number of transistors); and

— A description of any assumptions of use of the component with respect to its intended usage.

NOTE 2 Such documentation can be combined into one document constituting a “Safety Manual” or “Safety Application Note” of the analogue or mixed signal component.

5.3 Programmable logic devices

5.3.1 About programmable logic devices

5.3.1.1 General

As shown in Figure 17, PLDs can be seen as a combination of configurable I/O, non-fixed functions composed by logic blocks and user memory with a related configuration technology to configure them, signal routing capabilities connecting those logic blocks and fixed logic functions.

The non-fixed logic functions can include, but are not limited to, simple logic gates, multiplexers, inverters, flip-flops and memory to more complex functions such as digital signal processing functionality. Signal routing capabilities can range from simple point-to-point solutions, to complex bus interconnects with flexible routing possibilities and clocking options. PLDs can differ in their implementation of user memory. Some devices provide limited memory capabilities, whilst others provide local or global memory structures that can be used for a wide variety of applications. The more complex devices can also implement fixed functions such as CPUs, memory controllers, security modules, and others, thus freeing up design resources for user configurability. Clock, power and reset circuitries are fixed functions. It is up to the PLD design if single or multiple instances are implemented.

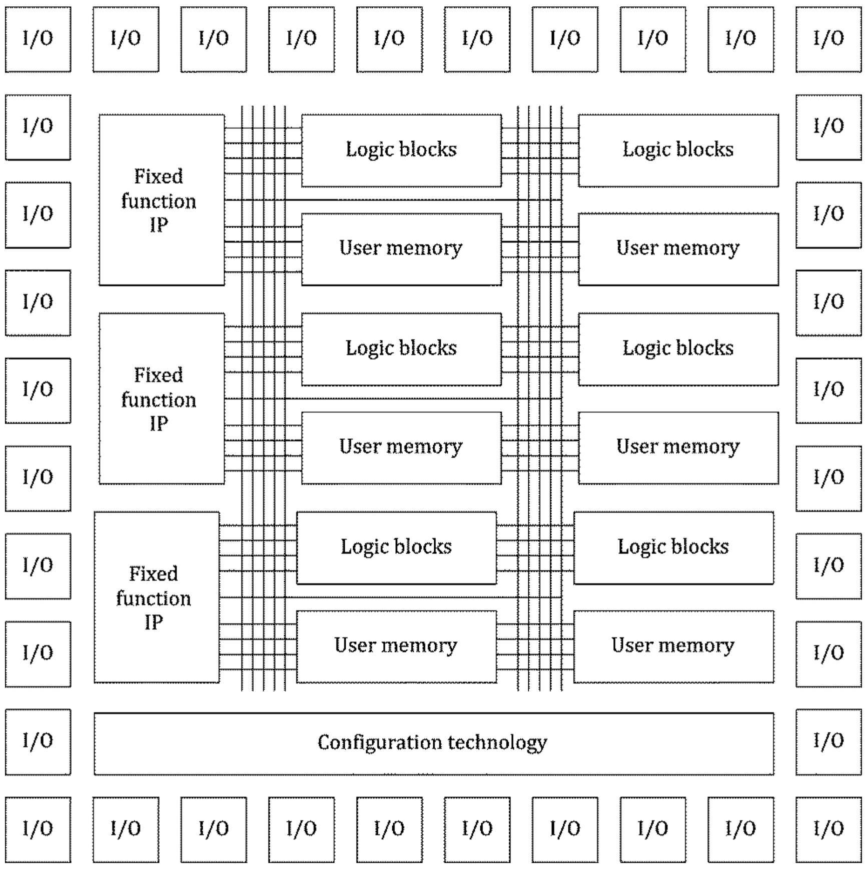


Figure 17 — A generic block diagram of a PLD

A common feature of PLDs is that users can configure them with the functionality adapted to the specific application needs. The design or configuration of the devices can be done with a variety of tools, ranging from very simple to entire development suites supporting complex features such as timing analysis and optimization of the design. Once the user design is completed it can be programmed into the device. Different technologies are used to allow either one time programmability or the reprogramming of the device multiple times. These methods can be further distinguished by providing volatile or non-volatile capabilities. This is represented in the block diagram by the block labelled "configuration technology".

NOTE The safety-related capabilities of non-volatile technologies such as Flash (reprogrammable) or Antifuse (programmable) can differ from those of volatile technologies such as SRAM.

5.3.1.2 About PLD types

Table 41 provides a non-exhaustive list of commonly used PLD types.

Table 41 — Commonly used PLD types

Type	Description
Programmable **A**rray **L**ogic (PAL)	One-time programmable devices that allow implementing sum-of-products logic for each of its outputs.

Gate **A**rray **L**ogic (GAL)	Similar functionality as PALs with the feature of being programmable many times.

Table 41 *(continued)*

Complex **P**rogrammable **L**ogic **D**evice (CPLD)	Non-volatile devices with similar functionality as PALs with a much higher integration rate and additional complex feedback paths.
Field **P**rogrammable **G**ate **A**rray (FPGA)	Mostly volatile implementation of very sophisticated logic, routing and memory functions.

5.3.1.3 ISO 26262 Lifecycle mapping to PLD

5.3.1.3.1 General

Figure 18 describes, using the same structure of ISO 26262-10:2018 Figure 22, how the ISO 26262 lifecycle is tailored to PLDs.

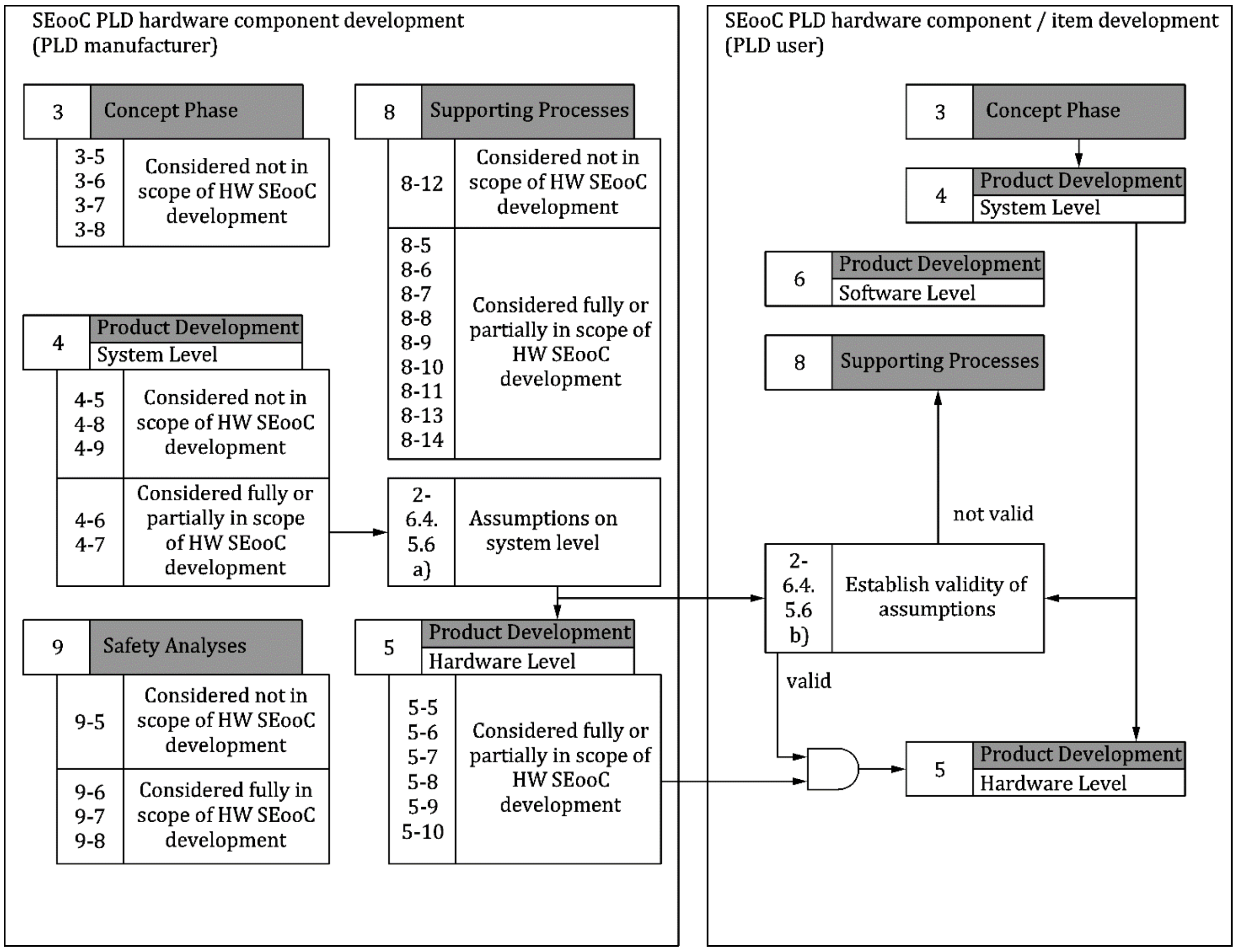

SEooC PLD hardware component development (PLD manufacturer)
3 Concept Phase
3-5 3-6 3-7 3-8 Considered not in scope of HW SEooC development
8 Supporting Processes
8-12 Considered not in scope of HW SEooC development
8-5 8-6 8-7 8-8 8-9 8-10 8-11 8-13 8-14 Considered fully or partially in scope of HW SEooC development
4 Product Development System Level
4-5 4-8 4-9 Considered not in scope of HW SEooC development
4-6 4-7 Considered fully or partially in scope of HW SEooC development
2-6.4.5.6 a) Assumptions on system level
9 Safety Analyses
9-5 Considered not in scope of HW SEooC development
9-6 9-7 9-8 Considered fully in scope of HW SEooC development
5 Product Development Hardware Level
5-5 5-6 5-7 5-8 5-9 5-10 Considered fully or partially in scope of HW SEooC development
SEooC PLD hardware component / item development (PLD user)
3 Concept Phase
4 Product Development System Level
6 Product Development Software Level
8 Supporting Processes
not valid
2-6.4.5.6 b) Establish validity of assumptions
valid
5 Product Development Hardware Level

Figure 18 — ISO 26262 lifecycle mapping to PLD

NOTE 1 In the context of this document, PLD manufacturer means an organization that develops the PLD and has the responsibility for the manufacturing of the PLD as semiconductor product. PLD user means an organization that develops a program for PLD or uses it in the application.

NOTE 2 Providers of IP blocks for PLD are considered in 4.5.

NOTE 3 Although each clause of ISO 26262 is not shown in Figure 18, this does not imply that they are not applicable.

The following clauses give examples with respect to some specific part of ISO 26262 for either PLD manufacturers or PLD users.

5.3.1.3.2 ISO 26262-2:2018 (management of functional safety)

In general, ISO 26262-2:2018 adapted to the appropriate level is applicable for the PLD manufacturer and the PLD user.

EXAMPLE 1 ISO 26262-2:2018, 6.4.2.1 requires that a project manager is appointed at the initiation of the item development. For a PLD manufacturer it means that a project manager is appointed at the initiation of the PLD development.

EXAMPLE 2 According to ISO 26262-2:2018, 6.4.3.9 the safety plan includes the planning of the hazard analysis and risk assessment as given in ISO 26262-3:2018, Clause 6. Since the hazard analysis and risk assessment is done on item level only this requirement is not applicable for a safety plan on PLD level.

EXAMPLE 3 ISO 26262-2:2018, 6.4.8 requires a functional safety audit to be carried out for the item. Since it is not possible for the PLD manufacturer to carry out a safety audit on item level he carries it out on the PLD level instead.

EXAMPLE 4 ISO 26262-2:2018, 7.4.2.1 requires the organization to appoint persons with the responsibility and the corresponding authority, as given in ISO 26262-2:2018, 5.4.2.7, to maintain the functional safety of the item after its release for production. For a PLD manufacturer this means that a person is appointed to maintain the functional safety of the PLD after its release for production since he cannot be responsible for maintaining the functional safety of the whole item.

5.3.1.3.3 ISO 26262-3:2018 (concept phase)

With respect to ISO 26262-3:2018, the PLD manufacturer usually does not have any responsibility during the concept phase, unless the PLD manufacturer also assumes the role of item integrator. For the PLD user, this part is applicable if the PLD user also has responsibility at the item level.

5.3.1.3.4 ISO 26262-4:2018 (product development at the system level)

For an SEooC development, ISO 26262-4:2018, Clause 6 and ISO 26262-4:2018, Clause 7 are partially or fully in scope. The same principle as discussed in ISO 26262-10:2018, 9.2.3 can be applied, where assumptions on the technical safety requirements and on the system-level design are made.

EXAMPLE Dedicated hardware safety measures can be implemented on the PLD by the PLD manufacturer to support the technical safety concept. Other measures can depend on the implemented user circuitry and can require specific measures (e.g. redundancy in logic, external watchdog) and are the responsibility of the user. The assumptions made by the PLD manufacturer on the system level measures is documented and verified by the PLD user.

If the PLD user is also the item integrator, ISO 26262-4:2018 is fully in scope.

5.3.1.3.5 ISO 26262-5:2018 (product development at the hardware level)

All the ISO 26262-5:2018 clauses, including ISO 26262-5:2018, Clause 8 and ISO 26262-5:2018, Clause 9, are applicable to PLD manufacturers and PLD users according to their level of contribution to the overall safety concept.

EXAMPLE If the PLD does not include any hardware safety mechanisms, the main role of PLD manufacturer is to provide base failure rate, failure modes, and failure modes distribution using, for example, the methods described in 4.6 of this document. A reference or exemplary computation of hardware architectural metrics can be provided but the PLD user computes the metrics for the specific design the user implements in the PLD.

With respect to ISO 26262-5:2018, Clause 8 and ISO 26262-5:2018, Clause 9, the responsibility of PLD manufacturers is generally limited to providing the distribution of failure modes or the information/methods/tools needed to enable PLD users to compute/verify the metrics and to provide diagnostic coverage values for the safety mechanisms that are embedded in the PLD (see 5.3).

With respect to ISO 26262-5:2018, Clause 10, for semiconductor components it is assumed that it is not related only to integration tests but it is applicable as well to PLD manufacturers and PLD users testing activities according to their level of contribution to the overall safety concept. Regarding evaluation of the diagnostic coverage (ISO 26262-5:2018, Annex D), see 5.3.4.

5.3.1.3.6 ISO 26262-6:2018 (product development at the software level)

Based on ISO 26262-4:2018, 7.4.5.3 and ISO 26262-5:2018, Clause 1, requirements of ISO 26262-5:2018 and ISO 26262-6:2018 can be combined in the case of programmable logic like PLDs.

In the case of a high-level synthesis flow, like developing in OpenCL, C-to-HDL flows, or a model based approach, interactions with the requirements of ISO 26262-6:2018 are considered for the development of the high level language code. ISO 26262-5:2018 is considered for follow on steps used for traditional PLD development.

In the case when the development flow for PLD users and PLD manufacturers is based on HDL languages, this is similar to the one used to develop microcontrollers, so ISO 26262-5:2018 applies. ISO 26262-6:2018 is not considered in this case.

NOTE Specific techniques and measures for user PLD circuit development are discussed in 5.3.5.3. For many methods there are similarities with respect to what is specified in ISO 26262-6:2018, e.g. observation of coding guidelines.

The level of application of ISO 26262-6:2018 also depends on the type of PLD technology. For example, in the case of a PAL, the part is in general simple enough that ISO 26262-6:2018 is not applied.

5.3.1.3.7 ISO 26262-7:2018 (production and operation)

In general ISO 26262-7:2018 adapted to the appropriate level is applicable for the PLD manufacturer. It is also applicable to the PLD user if he is involved in the production of a hardware element of the item or of the item itself.

EXAMPLE 1 In ISO 26262-7:2018, 5.4.1.1 the requirement is to plan the production process by evaluating the item. In the context of the PLD manufacturer the planning is done by evaluating the PLD instead of the item.

EXAMPLE 2 ISO 26262-7:2018, 5.4.1.4 requires to identify reasonably foreseeable process failures and their effect on functional safety and to implement appropriate measure to address these issues. It is applicable to a PLD production without modification.

EXAMPLE 3 ISO 26262-7:2018, 5.4.3.5 requirements for decommissioning instructions are typically not applicable for PLDs

EXAMPLE 4 To comply with ISO 26262-7:2018, 7.4.1.1 the PLD manufacturer implements a field monitoring process for the PLD.

5.3.1.3.8 ISO 26262-8:2018 (supporting processes)

In general ISO 26262-8:2018 adapted to the appropriate level is applicable for the PLD manufacturer and the PLD user.

ISO 26262-8:2018, Clause 13 could be applied to PLD.

Regarding ISO 26262-8:2018, Clause 11, please refer to contents of 5.3.5.2.

5.3.1.3.9 ISO 26262-9:2018 (Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses)

All ISO 26262-9:2018 clauses are applicable to PLD manufacturers and PLD users according to their level of contribution to the overall safety concept.

Regarding ISO 26262-9:2018, Clause 7, please refer to contents of 5.3.3.2.

5.3.2 Failure modes of PLD

In line with the lifecycle shown in 5.3.1.3, Table 42 summarise the failure modes that can be of concern for PLD users. Failure modes for PLD can be derived by applying key words as mentioned in 4.3.2.

The listings do not claim exhaustiveness and can be adjusted based on additional known failure modes. They can be used as a starting point to evaluate the diagnostic coverage of the provided safety mechanisms with the claimed DC. Any such claims are supported by a proper rationale.

Table 42 — Example of failure mode for PLD

Element (see Figure 17)	Description	Analysed failure modes
Fixed Function IP	See 5.3.1.1	See Table 29.
PLD Digital I/O		See ISO 26262-5:2018, Table D.1, element “Digital I/O” and Table 29.
Logic Block		Permanent corruption of the function implemented by the logic block. Transient corruption of the function implemented by the logic block.
Configuration Technology		Unintentional permanent change of the configuration of the logic block. Unintentional transient change of the configuration of one logic block. [c]
PLD Analogue I/O		See ISO 26262-5:2018, Table D.1, element “Analogue I/O” and Table 35.
User Memory		See 5.1.3.
Signal Routing capability[e]		Permanent corruption of the function implemented by a group of logic blocks, including time delay of the function. Transient corruption of the function implemented by a group of logic blocks.

[a] As described in 5.3.1, the fixed function IPs are a combination of elements similar to the ones that can be found in microcontrollers. They are typically implemented in a separated area with respect to the non-fixed functions and therefore they can be considered in each aspect similar to the elements discussed in ISO 26262-5:2018, Table D.1 and 5.1.2 and 5.1.3 for digital components.

[b] The relevance of this failure mode depends on the type of PLD technology and type of Logic Block, see 5.3.1.2.

[c] The relevance of this failure mode depends on the type of PLD technology, see 5.3.1.2.

[d] The I/O configuration logic can be inside the fixed function IP or in the I/O itself.

[e] Wires and routing of configuration technology are considered in "Signal Routing Capability"

5.3.3 Notes about safety analyses for PLDs

5.3.3.1 Quantitative analysis for a PLD

A similar approach as discussed in 5.1 can be also used for PLDs. A quantitative analysis of the PLD including the user design can be performed on different abstraction levels depending on the information available to the PLD user.

Information about the PLD usage and user design is refined during the development phase of the design and the analysis is repeated based on the latest information. The quantitative analysis of the PLD design can be augmented by a dependent failures analysis as described in 5.3.3.2.

The following two subclauses describe examples of PLD die failure rate calculations and examples of the distribution of the failure rate to the identified failure modes.

The hardware architectural metrics can be determined similar to the example given in Annex C of this part of ISO 26262. The level of detail required for the analysis depends on the targeted ASIL and the application.

5.3.3.1.1 Example of PLD die failure rate calculation per IEC TR 62380

The failure rates can be estimated as described in 4.6.

NOTE If failure rates provided by the PLD manufacturer are used, any de-rating factor applied to the provided data is made available.

This example follows the example given in 4.6.3.1.1.1. It makes similar assumptions and not each note is repeated in this clause. A PLD with the characteristics outlined in Table 43 is used for the example.

Table 43 — PLD resource overview

Element	Resources	Assumed IEC 62380 category
Logic blocks	1000	CPLD (EPLD, MAX, FLEX, FPGA, etc.)
User memory	16 kb	Low-consumption SRAM
Fixed function IP	20 k gates	Digital circuits, microcontroller, DSP
Configuration technology	10 kb	Low-consumption SRAM
NOTE For the Logic blocks, the CPLD entry of IEC TR 62380 has been used as example. For modern volatile FPGA devices, the LCA (RAM based) entry can be preferable.		

The complete PLD failure rate can be computed as shown in Table 44.

Table 44 — Example of the computation of the failure rates for the PLD

Element	λ_1	N	α	λ_2	Base FIT	De-rating for temp	Effective FIT
Logic blocks	$2{,}0 \times 10^{-5}$	100000 (100 transistors per macrocell)	10	34	34,0604	0,17	5,7903
User memory	$1{,}7 \times 10^{-7}$	98304 (6 transistors/bit for a low-consumption SRAM)	10	8,8	8,8005	0,17	1,4961
Fixed function IP	$3{,}4 \times 10^{-6}$	80000 (4 transistors / gate)	10	1,7	1,7082	0,17	0,2904
Configuration technology (based	$1{,}7 \times 10^{-7}$	61440 (6 transistors/bit for a	10	8,8	8,8003	0,17	1,4961

Element	λ1	N	α	λ2	Base FIT	De-rating for temp	Effective FIT
on SRAM)		low-consumption SRAM)					
Sum					53,3694		9,0729

NOTE 1 It is assumed that the number of transistors per macrocell (100, as derived from IEC TR 62380) does not include the transistors related to the configuration technology. For this reason the configuration technology is considered as a separate entry of the computation. An alternative approach could be to adapt the number of transistors and include the configuration technology in the logic blocks, user memory entries and other relevant elements.

NOTE 2 1 FIT corresponds to 1 failure per 10^9 hours of device operation.

NOTE 3 This table can be used also to derive a unitary FIT by dividing the resulting effective FIT with the number of elements.

EXAMPLE The FIT/logic block can be computed as 5,7903/1000 = 0,0057.

NOTE 4 As shown in 4.6, other alternatives are possible for the temperature de-rating factor. Those alternatives are applicable as well for PLDs.

The failure rates in Table 44 can be used to calculate the failure rates for this specific user design. The assumptions made for the user design are given in Table 45.

Table 45 — Example of user design resource usage and failure rate calculation

Element	Resource usage	Effective FIT
Logic blocks	23 %	1,3318
User memory	10 %	0,1496
Fixed function IP	100 %	0,2904
Configuration technology (based on SRAM)	15 %	0,2244
Sum		1,9962

NOTE 1 The unused resources are considered as not safety-related. Depending on the PLD structure, a dependent failures analysis can analyse the influence of the unused logic on the user design.

NOTE 2 An alternative approach is to consider the unused logic as safety-related and to estimate the respective fraction of faults that will lead to a safe failure (F_{safe} according ISO 26262-10:2018, Figure 10). This estimation can be done by means of a quantitative analysis supported by information provided by the PLD manufacturer.

The data can be further refined if more detail about the user design is available. For example a logic block has different configuration options and the user design can only use a certain configuration. This allows to further de-rate the calculated failure rate.

NOTE 1 A dependent failures analysis can be used to analyse the influence of the different configuration options on the user design.

NOTE 2 The derivation of the de-rating factor can be facilitated by appropriate design tools.

5.3.3.1.2 Example of transient failure rate calculation for PLD

The computation of the transient failure rate for PLD can follow 4.6, i.e. considering data provided by the PLD manufacturer derived from JEDEC standards such as JESD89A or, if this data are not available, soft error rate derived from public sources such as International Technology Roadmap for Semiconductors (ITRS).

NOTE In case the transient failure rate provided by the PLD manufacturer includes a de-rating factor (for example based on average PLD utilization factor or based on operational profile), this factor is explained to the PLD user.

Table 55 can be used to calculate the failure rates for this specific user design in the same way for transient faults, as shown in the previous clause.

5.3.3.1.3 Example of distribution of PLD failure rate to failure modes

Once the PLD failure rate has been estimated, it is distributed to the identified failure modes, i.e. the failure modes distribution is computed.

For PLD manufacturers, the failure modes distribution can be computed as described in 5.1.

The following are examples of approaches for identification of failure modes and respective determination of the failure modes distribution for PLD users:

a) Identification of the failure modes at the functional block level of the user PLD design; assumption of an equal distribution of the PLD failure rate to the identified failure modes;

b) Identification of the failure modes at the functional block level of the user PLD design; estimation of the distribution of the PLD failure rate to the identified failure modes based on expert judgment taking resource estimation (e.g. fixed function IP, number of logic blocks, user memory, etc.) into account, supported by documented evidences; and

c) Identification of the failure modes by means of a partitioning of the implemented user PLD design in elementary sub-parts; estimation of the distribution of the PLD failure rate to the identified failure modes based on the implemented user PLD design facilitated by information provided by the PLD manufacturer taking detailed resource utilization into account. This could be supported by appropriate design tools.

NOTE 1 In the context of PLD manufacturer, the elementary sub-part can be intended as a set of flip-flop and gates (e.g. logic cone). At the same way, in the context of PLD users, the elementary sub-part can be intended as the cone constructed of flip-flop in a logic block and the combinatorial logic represented by logic blocks. The level of detail, i.e. the number of elementary sub-parts considered depends on the targeted ASIL, the type of safety mechanism used and the application.

NOTE 2 The level of accuracy of the resulting quantitative data varies based on the approach used.

EXAMPLE 1 If information on the implemented user PLD design is available, then approach c) can provide the highest level of accuracy. If this information is not available and no argument can be given why one of the failure modes is more likely than the other, the approach a) can be used.

NOTE 3 The required level of accuracy of the failure mode distribution depends also on the targeted ASIL, the type of safety mechanism used and the application.

EXAMPLE 2 In the case of a user PLD design in lock-step, approach a) can be sufficient because a non-uniform distributed value for the failure mode distribution will not affect the claimed diagnostic coverage. Instead, for a user PLD design relying on a software test library to periodically test the PLD hardware, if arguments exist that one of the failure modes is more likely than the other approaches b) or c) are used depending on the required level of accuracy.

NOTE 4 A detailed failure mode definition like the one provided by approach c) can help to provide rationale for diagnostic coverage.

NOTE 5 For transient faults, the resource utilization can consider number of flip flops included in the logic blocks and the number of user memory bits of the user PLD design and number of configuration bits utilised by the user PLD design

Table 46 shows an example of the three approaches described above. It considers a SPI module implemented in a PLD.

Table 46 — Example of approaches for PLD failure modes distribution computation at PLD user level

Failure mode	Sub-parts involved	a)	b) See NOTE 1	c) See NOTE 2
Wrong or no clock	Clock generation	25 %	10/110 = 9,09 %	10/90 = 11,11 %
Wrong or no data reception	Peripheral bus interface Input shift register Data received register I/O pads	25 %	40/110 = 36,36 %	30/90 = 33,33 %
Wrong or no data send	Peripheral bus interface Output shift register Data send register I/O pads	25 %	40/110 = 36,36 %	30/90 = 33,33 %
Wrong configuration of SPI	Configuration registers Peripheral bus interface	25 %	20/110 = 18,18 %	20/90 = 22,22 %
NOTE 1 For this example, it is estimated that each sub-part consumes 10 logic blocks and therefore it is estimated that each failure mode has a failure mode distribution proportional to the sum of logic blocks consumed by each sub-part involved in the failure mode. NOTE 2 The difference between b) and c) is that the resource usage for the specific failure mode is not estimated anymore but the actual number of resources which contribute to the failure mode is computed. This is done not necessarily only on the sub-part level but also down to the elementary sub-parts level, if the logic blocks contributing to the failure mode span different sub-parts. In the example, it is measured that: Input shift register, output shift register, data received register and data send register are contributing 100 % to the respective failure mode and 0 % to the others; peripheral bus interface is measured to contribute 50 % to each data related failure mode and 100 % to configuration failure mode; I/O pads are measured to contribute 50 % to each data related failure mode.				

5.3.3.1.4 Verification of completeness and correctness of safety mechanism implementation with respect to hardware

As described in 4.8, fault injection simulation during development phase is a valid method to verify completeness and correctness of safety mechanism implementation with respect to hardware safety requirements as also to assist verification of safe faults and computation of their amount and failure mode coverage, as described in 5.1.10. This applies for PLD manufacturers as well.

With respect to PLD users, in case fault injection is necessary and no detailed information is available about how the user PLD design is mapped to PLD logic blocks, fault injection can be performed on the logic design before mapping.

EXAMPLE If fault injection is necessary to provide rationale of the diagnostic coverage claimed by a software test library periodically testing the user PLD design, then fault injection can be executed at a different level. For example, starting from the RTL design describing the user PLD design and then synthesizing it to obtain a reference netlist on which fault injection is performed. If the reference netlist does not correspond to the PLD design, then an argument is provided to explain why the injected faults are meaningful with respect to the assumed implementation of PLD design.

5.3.3.2 Dependent failures analysis for a PLD

As for any integrated circuit, dependent failures are important to be considered especially if hardware safety mechanisms or requirements for redundancy are implemented in the same component.

The flow for Dependent Failures Analysis (DFA) considered in this clause is the same as the one described in 4.7. Table 47 describes specificities – if any – to be considered in addition with respect to the steps defined in 4.7, for both PLD manufacturer and PLD users.

Table 47 — Specificities of DFA for PLD manufacturers and PLD users with respect to 4.7

Step (see Figure 15)	PLD manufacturer	PLD user
B1 – Identify hardware and software elements.	As defined in 4.7.	As defined in 4.7.
B2 – Identify dependent failures initiators.	Analysis considers also the interactions between configurable and fixed logic, including interactions related to reset or the configuration technology [a] .	Analysis considers also the impact of failures affecting the configuration technology and therefore potentially affecting multiple logic blocks at the same time.
B6 – Identify necessary safety measures to control or mitigate dependent failures initiators.	Analysis considers also the possibilities for providing separation between configurable and fixed logic	Analysis considers also the possibilities for providing separation between logic blocks
B10 – Evaluate the effectiveness to control or to avoid the dependent failure.	As defined in 4.7.	As defined in 4.7.
[a] For example, a fault in the fixed logic causing the configurable logic to lose the configuration		

The list for dependent failures initiators (DFI) considered in this clause is the same than the one described in 4.7. Table 48 and Table 49 describe specificities – if any – to be considered in addition with respect to DFI defined in 4.7, for both PLD manufacturer and PLD users, and the related countermeasures.

Table 48 — Specificities of DFI for PLD manufacturer and PLD user with respect to 4.7

Dependent Failures Initiators (DFI)	PLD manufacturer DFI	PLD user DFI
Failure of shared resources [a]	As defined in 4.7	Potential dependency of the available clock networks Failures of configuration technology (e.g. shared short or long distance common interconnects) Failures of shared programmable I/Os Wrong PLD configuration due to failures of external configuration memory or related interconnection
Single physical root cause	As defined in 4.7	Faults (e.g. in reset logic) causing the complete or partial loss of the PLD configuration
Development faults	Insufficient distance or isolation between fixed and configurable logic	Wrong usage of tools provided by PLD manufacturer [b] See also 4.7
Manufacturing faults	As defined in 4.7	Wrong usage of tools for configuration programming [b]
Installation faults	As defined in 4.7	As defined in 4.7
Repair faults	As defined in 4.7	Wrong usage of on-line reconfiguration functions
[a] In the context of PLD, “common” has to be interpreted not only as shared resources within either configurable or fixed logic but also as shared resources between configurable and fixed logic. [b] For example, user wrongly applies isolation/separation constraints		

Table 49 — Countermeasures related to DFI for PLD manufacturer and PLD user

Dependent Failures Initiators (DFI)	PLD manufacturer countermeasures	PLD user countermeasures
Failure of shared resources [a]	As defined in 4.7	Analysis of dependency of clock networks and dedicated clock monitors Analysis of failures of configuration technology and consequent adoption of separation/isolation techniques Analysis of failures of shared programmable I/Os and consequent adaptation of I/Os safety protocols CRC check of PLD configuration during runtime
Single physical root cause	As defined in 4.7	Analysis of dependency of the reset networks and dedicated watchdogs

Table 49 *(continued)*

Dependent Failures Initiators (DFI)	PLD manufacturer countermeasures	PLD user countermeasures
Development faults	Proper isolation or separation between fixed and configurable logic	As defined in 4.7
Manufacturing faults	As defined in 4.7	Proper instructions in PLD tool manual to prevent DFI
Installation faults	As defined in 4.7	As defined in 4.7
Repair faults	As defined in 4.7	Restricted use of on-line reconfiguration functions

[a] In the context of PLD, “common” has to be interpreted not only as shared resources within either configurable or fixed logic but also as shared resources between configurable and fixed logic.

5.3.4 Examples of safety mechanisms for PLD

Table 50 lists examples of safety mechanisms that can be used to address PLD failure modes described in Table 42.

NOTE This table is not exhaustive and other techniques can be used, provided evidence is available to support the claimed diagnostic coverage.

Table 50 — Mapping of PLD safety mechanisms with ISO 26262-5:2018; Annex D

Element	Examples of safety mechanisms
Fixed function IP	Table 33.
Clock	ISO 26262-5:2018, Table D.8 On-chip clock status indication [a]
Power supply	ISO 26262-5:2018, Table D.7 Separate voltage planes [b]
Digital I/O	ISO 26262-5:2018, Table D.5
Analogue I/O	ISO 26262-5:2018, Table D.5
Logic block	ISO 26262-5:2018, Tables D.4 Table 33 Mix of spatial and temporal redundancy by means of reconfiguration
Off-chip communication	ISO 26262-5:2018, Tables D.6
Configuration technology	Table 31, Table 32 Read-back on download by downloading device [c]
User memory	Table 31, Table 32
Signal routing capability	Table 34

[a] Many PLDs offer clock generation and management resources and also provide monitoring of clock functionality and associated status pins/register to indicate when a specific clock is functioning properly (e.g. whether or not a clock output is in proper phase with a master clock input).

[b] Voltage plane means electrically isolated voltage supply plane regions with each plane region being connectable to an external supply voltage.

[c] Refers to the capability of many programmable devices to check the contents of its configuration registers and compare those to the intended (design specific) contents. If a mismatch is detected, this feature can change the status of an output pin or generate an interrupt so that the system can respond appropriately. To improve the usability as an online monitoring safety mechanism an efficient read back test can prioritize between safety-related and non-safety-related parts within a device. Safety related parts can be checked more frequently to considerably shorten failure detection time.

5.3.5 Avoidance of systematic faults for PLD

5.3.5.1 Avoiding systematic faults in the implementation of PLD

Since there are no significant differences in the specification, design and verification flow used by PLD manufacturers with respect to the flow used by digital component manufacturers, the same recommendations given in 5.1.9 (and related Table 35) can be applied.

5.3.5.2 About PLD supporting tools

PLD related tools can be distinguished in two categories:

- tools used prior to the production (i.e. used by PLD manufacturers); and
- tools used by PLD users.

The confidence in use of tools belonging to both categories are analysed according to the requirements of ISO 26262-8:2018, Clause 11.

EXAMPLE 1 According ISO 26262-8:2018, Clause 11, a tool used for place and route by the PLD manufacturer can be considered TI2, since its malfunction can introduce an errors in a safety-related element being developed; If it can be shown that design rule check (DRC) and layout versus schematic (LVS) with appropriate rule sets, as foreseen in state-of-the-art IC design flows, can detect possible errors introduced by the tool with a high degree of confidence, then a TD1 can be claimed. In this case it can be considered TCL1 based on ISO 26262-8:2018, Table 3.

EXAMPLE 2 According ISO 26262-8:2018, Clause 11, a tool used for place and route by the PLD users can be considered TI2, since its malfunction can introduce an error in a safety-related element being developed. If the error can be detected with a medium degree of confidence by the consequent hardware and integration tests, due to the complexity of the circuitry, then it can be considered TD2. Therefore it can be considered as TCL2 based on the ISO 26262-8:2018, Table 3. If the ASIL of the respective item is for example ASIL B, the tool provider can qualify the software tool by using an appropriate combination of "increased confidence from use" and "evaluation of the tool development process".

5.3.5.3 Avoiding systematic faults for PLD users

For PLD manufacturers, as for a microcontroller, a PLD is developed based on a standardised development process for which the example in 5.1.9 applies.

The two following approaches are examples of how to provide evidence that sufficient measures for avoidance of systematic failures are taken care of by the PLD user during the development, by using appropriate processes:

- using a checklist such as the one reported in Table 51; and
- giving the rationale by field data of similar products which are developed based on the same process as the target device (for example using ISO 26262-8:2018, Clause 14).

3005 **Table 51 — Examples of measures to avoid systematic failures for PLD users**

ISO 26262-5:2018 requirement	Design phase	Technique / Measure	Aim
7.4.1.6 Modular design properties	**Design entry**	Structured description and modularization	The description of the PLDs functionality is structured in such a fashion that it is easily readable, i.e. circuit function can be intuitively understood on basis of description without simulation efforts
7.4.1.6 Modular design properties		Design description in HDL	Functional description at high level in hardware description language, for example such like VHDL or Verilog.
7.4.2.4 Robust design principles		Observation of coding guidelines	Strict observation of the coding style results in a syntactic and semantic correct circuit code
7.4.2.4 Robust design principles	**Design entry**	Restricted use of asynchronous constructs	Avoidance of typical timing anomalies during synthesis, avoidance of ambiguity during simulation and synthesis caused by insufficient modelling, design for testability. This does not exclude that for certain types of PLD implementations, asynchronous logic could be useful; in this case, the aim is to suggest additional care to handle and verify those circuits. The timing of asynchronous resets bears risks due to different propagation times to a potentially large number of attached elements. Since the asynchronous reset signal is not correlated to the clock of attached synchronous elements, metastability can be a problem upon reset deassertion. Arising problems are expected to depend on design and environment factors, such as temperature and fanout of the reset net.
7.4.2.4 Robust design principles		Synchronisation of primary inputs and control of metastability	Avoidance of ambiguous circuit behaviour as a result of set-up and hold timing violation
7.4.4 Verification of hardware design		HDL simulation	Functional verification of circuit described in VHDL or Verilog by means of simulation
7.4.4 Verification of hardware design		Functional test on module level (using for example HDL test benches)	Functional verification "Bottom-up"
7.4.4 Verification of hardware design		Functional test on top level	Verification of the PLD (entire function)
7.4.4 Verification of hardware design		Functional and structural coverage-driven verification (with coverage of verification goals in percentage)	Quantitative assessment of the applied verification scenarios during the functional test. The target level of coverage is defined and shown
7.4.4 Verification of hardware design		Application of code checker	Automatic verification of coding rules ("coding style") by code checker tool.
7.4.4 Verification of hardware design		Documentation of simulation results	Documentation of each data needed for a successful simulation in order to verify the specified circuit function.
7.4.4 Verification of hardware design		Integration and verification of soft IPs	See Clause 4.5
7.4.4 Verification of hardware design	**Synthesis, mapping, floor planning, placement, routing**	Check of PLD vendor requirements and constraints	Requirements and constraints defined by PLD vendor are considered during PLD design
7.4.4 Verification of hardware design		Analysis of PLD supporting tool outputs	Outputs of PLD supporting tools are analysed. Arguments are provided to waive warnings and Errors.
7.4.1.6 Modular design properties		Documentation of constraints, results and tools	Documentation of each defined constraint that is necessary for an optimal synthesis, mapping, placement and routing of the PLD design
7.4.1.6 Modular design properties		Script based procedures	Reproducibility of results and automation of the synthesis, mapping, placement and routing

Table 51 *(continued)*

ISO 26262-5:2018 requirement	Design phase	Technique / Measure	Aim
7.4.4 Verification of hardware design		Simulation and timing verification of the final netlist	Independent verification of the netlist after synthesis, mapping, placement and routing – including timing verification
7.4.4 Verification of hardware design		Comparison of the final netlist with the reference model (formal equivalence check)	Functional equivalence check of the final netlist with RTL.
7.4.2.4 Robust design principles		Adequate time margin for process technologies in use for less than three years	Assurance of the robustness of the implemented circuit functionality even under strong process and parameter fluctuation. A time margin in the timing analysis is considered either in the libraries or by PLD user.
7.4.4 Verification of hardware design		Design rule check (DRC)	Execution of design rule checks on floor planned logic
9.4.2.5 Dedicated measures 10 Hardware integration and testing	**PLD integration and testing**	PLD verification	Verification of the PLD prototype, including verification of PLD correct configuration (e.g. using checksums).
7.4.5 Production, operation, service and decommissioning 9.4.2.5 Dedicated measures 10 Hardware integration and testing		PLD integration	Verification and integration of the PLD in the system

5.3.6 Example of safety documentation for a PLD

Recommendations in terms of the safety documentation for an SEooC digital component are given in 5.1.11, as also in terms of the contents of the documentation which can be consolidated in a so called “Safety Manual” or “Safety Application Note”. Those recommendations can be used also by PLD manufacturers and PLD users, with the following remarks:

— The DIA between PLD manufacturer and PLD user specifies which documents are made available and what level of detail is provided to the PLD user;

— The main focus of the safety documentation provided by PLD manufacturer is:

    — the description of the results of the analyses of the development processes of the PLD manufacturer with respect to the applicable requirements of ISO 26262:2018;

    — the description of the results of the analyses of the PLD supporting tools with respect to the applicable requirements of ISO 26262:2018;

    — the provision of information (for example the PLD failure rate, the PLD failure modes with the related failure modes distribution, the claimed diagnostic coverage for safety mechanisms that are already implemented in the PLD etc.) to be used by PLD users during their safety analyses;

    — proposals or examples of safety mechanisms, for example with respect to dependent failures etc.; and

    — the list of assumptions of use to guide PLD users in the correct utilisation of the safety-related information provided with the PLD.

— The work products of the safety lifecycle are provided by the PLD user. The completeness of the work products depends on whether the PLD user also assumes the role of the item integrator.

5.3.7 Example of safety analysis for PLD

A detailed example of a quantitative safety analysis for PLD is described in Annex E of this part of ISO 26262.

5.4 Multi-core components

5.4.1 Types of multi-core components

Table 52 summarises the different types of multi-core components considered in this clause.

Table 52 — Types of multi-core components

Multi-core component type	Description
Homogeneous multi-core component	Homogeneous multi-core components include only identical PE
Heterogeneous multi-core component	Heterogeneous multi-core components have non-identical PEs, typically with different Instruction Set Architecture (ISA)

EXAMPLE Figure 19 shows a diagram of a generic homogeneous dual-core system, with CPU-local level 1 caches, and a shared, on-die level 2 cache.

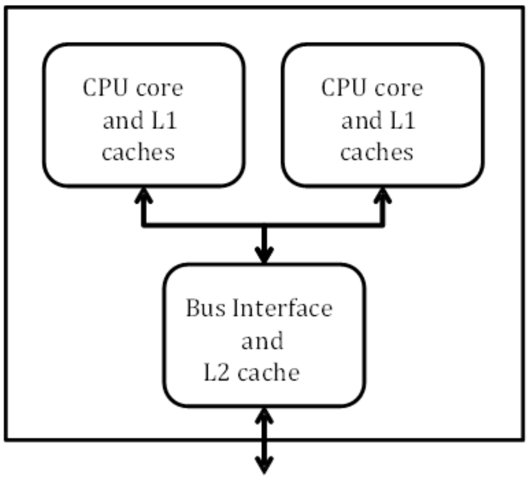


Figure 19 — Generic diagram of a dual-core system

5.4.2 Implications of ISO 26262 on multi-core components

5.4.2.1 Introduction

This clause provides some clarifications and examples for semiconductor vendors as also for system developers for which safety requirements – previously allocated to multiple components – are now allocated to a multi-core.

5.4.2.2 Clarifications on ASIL decomposition in multi-core components

As shown in Figure 20, the initial safety requirement can be decomposed to two (or more) safety requirements which are allocated to sufficiently independent hardware and/or software elements.

EXAMPLE 1 An ASIL B safety requirement is decomposed in two redundant requirements, ASIL B(B) – satisfied by the software running in PE1 - and QM(B), satisfied by the software running in PE2.

NOTE 1 If a software or hardware based comparator is used to compare the results of software 1 and software 2, the comparator has to be developed according the initial ASIL.

NOTE 2 software redundancy can be implemented also outside the ASIL decomposition, as a safety mechanism to provide diagnostic coverage.

EXAMPLE 2 Examples of software redundancies are: software heterogeneous redundancy; software architecture in which a redundant copy of the software is executed by two identical PE in parallel and then compared by another software unit. This technique is usually referred to as software lock-step or loosely coupled lock-step. A description of those types of software redundancies is not in the scope of this clause.

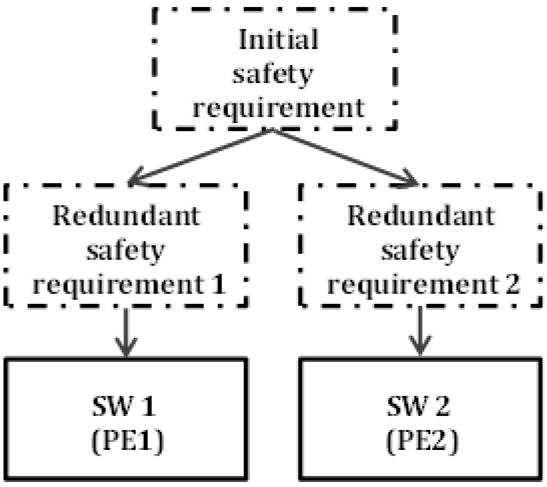

Initial safety requirement
Redundant safety requirement 1
Redundant safety requirement 2
SW 1 (PE1)
SW 2 (PE2)

Figure 20 — ASIL decomposition in the context of multi-core

This decomposition follows the requirements described in ISO 26262-9:2018, Clause 5. Guidelines of application to multi-core components are given in the following sub-clauses.

NOTE 3 ASIL decomposition has effect on both hardware and software systematic failures. This clause provides clarifications only with respect to the software level, for example how shared resources are considered in that context. Moreover, this clause provides clarification on how requirements on the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures of the multi-core component remains unchanged by ASIL decomposition. It also clarifies how the software redundancy inherently related to ASIL decomposition can be considered in the metrics evaluation.

Application of ASIL decomposition between two or more diverse software elements is possible if sufficient independence regarding software caused dependent failures can be shown between the corresponding software elements.

Shared resources are a known DFI. For a software element a shared resource can be a hardware element (e.g. PE, RAM, cache) as well as a software element (e.g. drivers). Within a multi-core the issue caused by shared resources (e.g. memory, time, execution or exchange of information interferences) can be resolved by assigning the corresponding software elements to independent PEs without the same shared resources. Other issues (e.g. shared memory, commonly used software elements) are addressed analogously to a single core system (e.g. memory encapsulation via MPU by the OS, developing the commonly used software elements compliant with the initial ASIL).

NOTE 4 Safety mechanisms ensuring the independence of the corresponding software elements are implemented compliant with the initial ASIL and not with the decomposed ASIL.

EXAMPLE 3 The task to read and monitor an external sensor is allocated to the software. The initial requirement is rated with an ASIL X. In the further development steps this requirement is allocated to software element software_Mon.1 with an ASIL Y(X) and to software element software_Mon.2 with an ASIL Z(X). A DFA has shown that next to other issues the shared resources (cores, RAM and a software driver "software peripheral" forwarding the sensor values to software_Mon.1 and software_Mon.2) can threaten the independence requirement, i.e. causing memory, time, execution or exchange of information interferences between software_mon.1 and software_mon.2. In this example the shared core issue is addressed by mapping software_Mon.1 and software_Mon.2 to two different PEs, therefore un-sharing the cores. The memory interference aspect is addressed by memory encapsulation via a MPU which is configured by the OS. Since in this case the OS is a safety mechanism ensuring the independence between software_Mon.1 and software_Mon.2 it is developed compliant with ASIL X. The issue with the shared software resource "software peripheral" is addressed by developing it compliant with the initial ASIL, ASIL X.

When applied, ASIL decomposition requires a sufficient level of independence between the redundant elements. As stated in ISO 26262-9:2018 5.4.10, "sufficient" does not mean completely independent. Sufficient independence can be achieved not only by prevention of dependent failures but also by detection and mitigation of dependent failures at appropriate levels depending on allocated safety requirements.

The requirements on the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures of the multi-core component remain unchanged by ASIL decomposition as given in ISO 26262-5:2018.

An ASIL decomposition by itself has no impact on the metric evaluation, i.e. no metric requirements are altered as a result of ASIL decomposition.

EXAMPLE 4 As described in the EXAMPLE of ISO 26262-9:2018, 5.4.8, an ASIL D requirement is first decomposed into one ASIL C(D) requirement and one ASIL A(D) requirement. Then the ASIL C(D) requirement can then subsequently be decomposed into one ASIL B(D) requirement and one ASIL A(D) requirement, each mapped to a different PE. The decomposition has no impact on the necessity to evaluate the hardware metrics compliant with ASIL D requirements of the item, i.e. the ASIL decomposition procedure does not automatically infers a lower ASIL requirement as far as the metrics evaluation is concerned: a safety analysis is needed to verify the overall metric compliance to the initial ASIL requirement.

Because the requirements on the evaluation of safety goal violations due to random hardware failures of the multi-core component remain unchanged by ASIL decomposition, the normative requirements for ASIL C and ASIL D as given in ISO 26262-5:2018, Clause 9 are applicable, including:

— ISO 26262-5:2018, 9.4.1.2 and 9.4.1.3 (for ASIL C and ASIL D);

— ISO 26262-5:2018, 9.4.3.8 (for ASIL D); and

— ISO 26262-5:2018, 9.4.3.9 (for ASIL C)

EXAMPLE 5 In the case of an ASIL D decomposition into ASIL B(D) for PE1 and ASIL B(D) for PE2, both PE1 and PE2 have to be considered as driven by ASIL D requirement. For example, according to ISO 26262-5:2018, 9.4.3.8, a dual-point failure is considered plausible if one or both hardware parts involved have a diagnostic coverage (with respect to the latent faults) of less than 90 %; or one of the dual-point faults causing the dual-point failure remains latent for a time longer than the multiple-point fault detection interval as specified in requirement ISO 26262-5:2018, 6.4.8.

5.4.2.3 Clarifications on Freedom from interference (FFI) in multi-core components

If in a multi-core context multiple software elements with different ASIL ratings coexist, a freedom from interference analysis according to ISO 26262-9:2018, Clause 6 is carried out.

The exemplary faults listed in ISO 26262-6:2018, Annex D can be a starting point for the analysis.

NOTE 1 This clause focuses only on cascading faults between software elements implemented in PEs. Interferences can also be caused by hardware dependent failures, in this case ISO 26262-9:2018 clause 7 applies.

With respect to interference against "Memory" entries of ISO 26262-6:2018, Annex D.2.3, the case of interference with private resources is considered. This type of interference can affect data or program regions belonging to one of the PEs.

EXAMPLE 1 Private data can be variables that belong to a safety-related software element in one of the PEs: A corruption of such variables from the other PEs leads to a malfunction of the software. In this case, a safety mechanism supervising the access and ensuring exclusive access helps to avoid interference. This example is related to software interferences (i.e. the variable corruption is caused by a software error). Interferences can also be caused by hardware dependent failures, in this case ISO 26262-9:2018, Clause 7 applies.

EXAMPLE 2 Private program regions can be related to the corruption of a program in a non-volatile memory. In this case a mechanism restricting programming only from the higher ASIL elements helps to avoid interferences. This example can be applied to software related interference (in a case where the program corruption is caused by a software error; for example wrong permissions causing software to overwrite the program memory). In this case ISO 26262-9:2018, Clause 7 applies.

This type of interference can also affect resources shared between different PEs.

EXAMPLE 3 A CAN peripheral is used by more than one core to exchange information with other ECUs. Interference can lead to an incorrect message transmission. In this case usage of robust end-to-end protection mechanisms (for example the ones listed in ISO 26262-5:2018, Table D.6) can help to detect interferences.

With respect to interference against "Time and execution" entries of ISO 26262-6:2018, Annex D.2.2, the primary case to consider is interference that affects the execution latency or correct programming sequence of one core.

EXAMPLE 4 A CAN peripheral is used by more than one core to exchange information with other ECUs. If the PEs, processing tasks with a lower ASIL continuously request transmissions from the CAN peripheral then the higher ASIL tasks running in another core are not able to receive and/or transmit required information. A time monitoring mechanism (for example using the principles described for the safety mechanisms listed in ISO 26262-5:2018, Table D.8) can help to identify such conditions.

NOTE 2 Additional requirements related to timing are described in 5.4.2.4.

With respect to the interference against "Exchange of information" entries of ISO 26262-6:2018, Annex D.2.4, interferences manifesting as failures in "Memory" or "Time and execution" can be caused by failures in exchange of information between different PEs.

EXAMPLE 5 A message from a non-safety-related core is interpreted as safety-related (masquerading fault).

NOTE 3 Usage of robust end-to-end protection mechanisms (for example the ones listed in ISO 26262-5:2018, Table D.6) can help to detect interference.

When software partitioning, e.g. separation of functions or elements to avoid cascading failures, is used to implement freedom from interference between software components, ISO 26262-6:2018, 7.4.11 is applied.

Techniques such as hypervisors can help to achieve software partitioning (e.g. references [26] and [5]).

NOTE 4 Other techniques are also possible, such as microkernels (e.g. reference [12]).

It is worth considering the following points during safety analyses of multi-core involving hypervisors technologies:

— Virtualization technologies can support the argument to guarantee freedom from interference between software elements running in multi-core. A dependent failures analysis on software level is required and can be supported by consideration of the failure modes listed in ISO 26262-6:2018, Annex D; and

  NOTE 5 Positive effects of virtualization technologies with respect to freedom from interference can be compromised by systematic faults in hypervisor software. Similarly, virtualization technologies can be affected by hardware faults in the supporting hardware resources (like memory management unit) or in the related shared resources. Those faults are analysed according to the methods described in ISO 26262-9:2018, Clause 8 and dedicated guidance for digital components is described in 5.1. Virtualization technologies can also be affected by hardware dependent failures; in this case ISO 26262-9:2018, Clause 7 applies.

  NOTE 6 If some of the hypervisor functions are delegated to tasks in the software partitions, then the analysis mentioned in NOTE 1 extends also to the partitions.

— Virtualization technologies are typically not able to provide sufficient prevention or detection of permanent or transient faults affecting the multi-core.

  NOTE 7 It is possible for virtualization technologies to detect random failures if they manifest as violations of software partitioning enforced through virtualization. Detection of specific hardware failure modes can be demonstrated by means of case by case detailed analyses based on the methods described in ISO 26262-9:2018, Clause 8. Dedicated guidance for digital components is described in 5.1.

5.4.2.4 Timing requirements in multi-core component

There are some clauses in ISO 26262-6:2018 related to execution timing requirements, for example:

— ISO 26262-6:2018, 6.4.2 e) requires that the specification of the software safety requirements considers timing constraints;

— ISO 26262-6:2018, 7.4.14 requires that an upper estimation of required resources for the embedded software is made, including execution time;

— ISO 26262-6:2018, Table 12 Note c) indicates that there are relations between hardware and software that can influence e.g. the average and maximum processor performance, minimum or maximum execution times; and

— ISO 26262-6:2018, Annex D describes timing and execution failure modes (including incorrect allocation of execution time) as potential initiators of interferences between software elements.

Multi-cores are potentially subject to timing faults (see reference [26]); therefore the previous listed clauses are carefully considered with dedicated analyses and adequate countermeasures identified.

EXAMPLE 1 Typical dedicated analyses for the identification of timing faults potentially violating the safety goal are based on the upper estimation of execution time (e.g. reference [6]).

EXAMPLE 2 Typical hardware-based countermeasures for detection of violation of timing requirements are watchdogs, timing supervision units and specific hardware circuits (e.g. reference [26]). Software-based countermeasures are also possible (e.g. reference [3]).

5.5 Sensors and transducers

5.5.1 About Sensors and transducers

5.5.1.1 Terminology of sensors and transducers

As defined in ISO 26262-1:2018, a transducer is a hardware part that converts energy from one form to another and, as such, it is a critical element to be considered with respect to automotive functional safety. The quantification of the output energy form as compared to the input energy form is dependent upon the sensitivity of the transducer. Input energy includes energy which is stored within chemical bonds.

A sensor is an element that includes at least a transducer and a hardware element that supports, conditions or further processes the transducer output for utilization in an E/E system.

EXAMPLE 1 DC bias, amplification, filtering.

The relationship between a transducer and a sensor is shown in Figure 21.

EXAMPLE 2 The transducer in Figure 21 can be a separate component and the supporting circuitry can be a separate component or multiple components. The functionality of the transducer and supporting circuitry together would make up the sensor function.

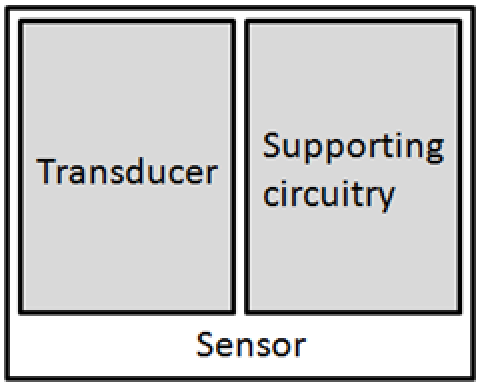


Figure 21 — General relationship between sensor and transducer

Like other electrical elements, a sensor can be made up of parts and sub-parts, and be of varying complexity.

EXAMPLE 3 A semiconductor component with analogue output consisting of a transducer and amplifier.

EXAMPLE 4 An element consisting of housing, a sensor IC with digital signal processing and digital output, required external components (resistor(s), capacitor(s), etc.) and a connector which interfaces to a wiring harness (see Figure 22) In this example, both the sensor IC and other elements can be classified as sensors but exist at different levels of hierarchy.

The use of the term ‘transducer’ in this clause will refer specifically to those transducers that are fabricated using semiconductor process technology, including Micro Electro Mechanical Systems

(MEMS). The use of the term ’sensor’ in this clause will refer specifically to those sensors containing transducers, as previously described, and having an electrical output.

Sensors can be classified in various ways, as indicated in [44].

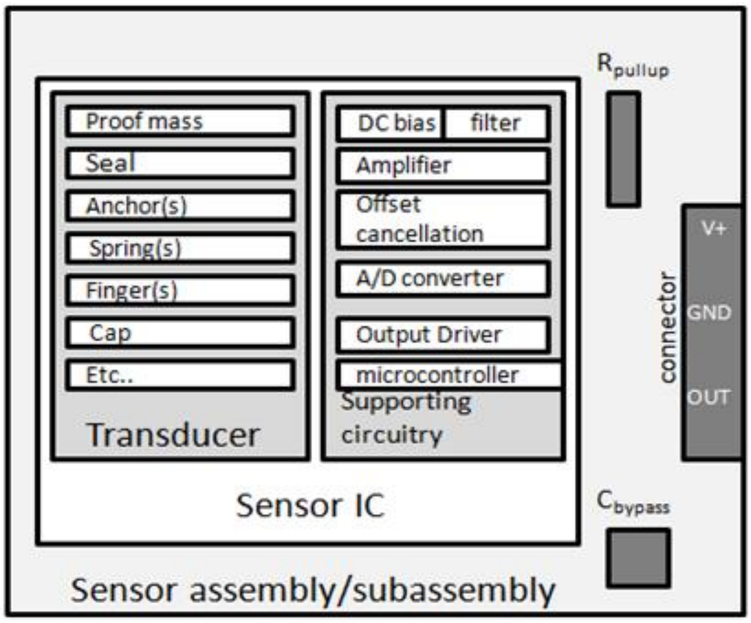


Figure 22 — Example of a complex hierarchical sensor

5.5.2 Sensor failure modes

5.5.2.1 About failure modes

In the scope of this clause, the output of each transducer is in the electrical domain. It then follows that the failure modes of the transducer will be electrical failure modes regardless of cause. Any failure of an element in the signal path starting at the transducer can have an effect on the sensor output.

Failure modes for transducers can be derived by applying key words as mentioned in 4.3.2.

Table 53 includes failure modes that are common to a variety of different types and complexities of transducers (independent of measurement, detection means, conversion means, etc.) [43]. This table is not exhaustive as electrical failure modes of a transducer depend upon the type and function of the specific transducer and is used for example only. Failure modes of digital or analogue supporting circuitry that are contained in the sensor signal path is covered in 5.1 and 5.2.

The failure modes of the transducer appear as deviations to the nominal sensor output. Failure modes of the sensor also originate from faults in the supporting circuitry in the signal path between the transducer output and sensor output. The correlation between the failure modes of the transducer and failure modes of the sensor output will depend on the specific implementation of the transducer in the sensor. According to ISO 26262–5:2018, Table D.1, a detailed analysis of the actual sensor type is necessary to identify each failure mode.

Possible effects of transducer failure modes on the system output are included in Table 53. Whether these effects are considered relevant failure modes of the sensor depends on the safety requirements of the sensor. In general, a deviation in nominal performance of a sensor within a specified range can be accounted for by a system or element as long as the deviation remains predictable. Any performance excursions outside of a predicted range or behavioural model can lead to violations of sensor safety requirements.

Table 53 — Example of transducer failure modes (electrical)

Technical Specification	**Failure mode**	**Description**
Offset	Offset outside of specified range	Transducer output is offset from the ideal value in the absence of stimulus (input energy)
	Offset error over temperature	Offset error over temperature is beyond specified limits
	Offset drift	Offset value changes over time
Dynamic Range	Out of range	Transducer output is outside of prescribed operational range
Sensitivity (Gain)	Sensitivity too high/low	Sensitivity deviates beyond specified limits
	Stuck at	Sensitivity is zero due to mechanical and electrical failure (e. g. particle short, stiction)
	Nonparametric sensitivity	Sensitivity deviates from a mathematical relationship within its specified range including discontinuities or clipping of output response
	Noise, poor repeatability	Variable threshold required to overcome dynamic noise floor
	Sensitivity error over temperature	Sensitivity deviates beyond specified limits over temperature
NOTE 1 Possible effects at system level includes: Inaccurate switching threshold, Changes in switching threshold over temperature, Changes in switching threshold over time, Loss of function, Inaccurate switching threshold, Phase shift (leading, lagging), C hanges in duty cycle, Variation of output switching threshold, Changes in switching threshold over temperature, Phase shift over temperature, Changes in duty cycle over temperature		

5.5.2.1.1 Example failure modes of an image sensor based camera

A typical camera based image sensor is composed of the following parts and sub-parts:

— Pixel array;

— Analog chain, clock and power supply;

— Configuration and calibration circuitries;

— Memories including RAM, OTP;

— Special circuitries;

— Digital control; and

— Interface.

Failure modes of digital control, memories and related interface are analysed according to what is described in 5.1, while the failure modes of analogue chain, clock and supply are analysed according to what is described in 5.2. The following are examples of failure modes that can affect the pixel array and the remaining parts and sub-parts, based on the categories listed in Table 53:

— Specific failure modes:

— Camera fault (intended as a major fault of the array leading to full image fault);

— Loss of single image rows or Horizontal line failure;

— Loss of single image columns or Vertical line failure;

— Loss of image frames;

— Related to sensitivity (gain):

— Loss of pixel data or corrupted bits in the image;

— Noise in the image;

— Related to offset:

— Horizontally or vertically shifted images;

— Related to dynamic range:

— Under or over exposed image / pixel, including issues related to dynamic range.

5.5.2.1.2 Production processes and failure modes

The manufacturing of semiconductor based sensors and transducers is a multi-step process including many mechanical procedures such as wafer grind/thinning, saw, pick and place, die attach, wire bond, die stacking, and encapsulation. The mechanical stresses induced by these processes can impact material properties such as mobility which then result in fluctuations of device parameters. The technical specifications of a transducer/sensor, such as offset, are impacted directly by the stresses of the assembly process. A sensor or transducer that does not exhibit a specific failure mode before a mechanical production process is not guaranteed to be free of that failure mode after the process.

Sensors are typically calibrated by various methods, such that their technical specifications (e.g. offset, sensitivity) are centred within their respective ranges, before being shipped by the supplier. The supplier's production processes, however, are not the only source of assembly-induced mechanical stress. The production processes of the direct customer, and possibly those further down the supply chain, can introduce mechanical stresses or other environmental factors that can result in a failure mode of the sensor. Such processes can include, but are not limited to, surface mounting, clamping, pick and place, reflow and conformal coating processes. If possible, it is a valuable practice to verify that the sensor/transducer is functioning within specification after the final stage of each successive supplier's production flow.

Table 54 lists some commonly occurring failure modes of sensors/transducers that can result from assembly processes. This table is not exhaustive. The capability to detect any deviations in sensor performance introduced by these processes, as well as their mitigation, are considered during the design phase to ensure adequate robustness (e.g. offset cancellation, sensitivity adjustment, and test modes). Refer to 6.5.6 for more information concerning the avoidance of systematic faults during the development phase.

Table 54 — Sensor Anomalies which can be introduced during Production Processes

Production-Related Failure mode	**Possible Effect**	**Possible Causes**
Sensitivity shift	Inaccurate switching threshold, Phase shift	Mechanical stress (Piezo-resistance), Temperature induced mechanical stress, Mechanical short or open

	Duty cycle shift	(e.g. broken metal, foreign material, ILD void), Trapped Charge, Drop, shock, compression/decompression, vibration, Moisture intrusion, Plastic deformation caused by temperature cycling, material curing
Loss of sensitivity	Loss of system	
Offset	Inaccurate switching threshold	

5.5.2.1.3 Microelectromechanical Causes of Failure

MEMS sensors are used in a variety of applications and employ a mechanical detection method to sense the environment by a typically elastoelectric (movement based) means of conversion. Because the conversion method is mechanical, the performance of the transducer is directly affected by its physical structure and any deviations in the structure from the nominal specifications.

A representation of a generic MEMS transducer is shown in Figure 23 and Figure 24.

Figure 23 shows individual parts of a generic MEMS transducer including electrodes, proof mass, anchors, springs and capacitive plates.

Figure 24 shows additional detail from a side view including the cavity, sealing cap and anti-stiction coating. Any non-ideal physical/mechanical characteristic of these parts will have an (electrical) effect on the transducer output.

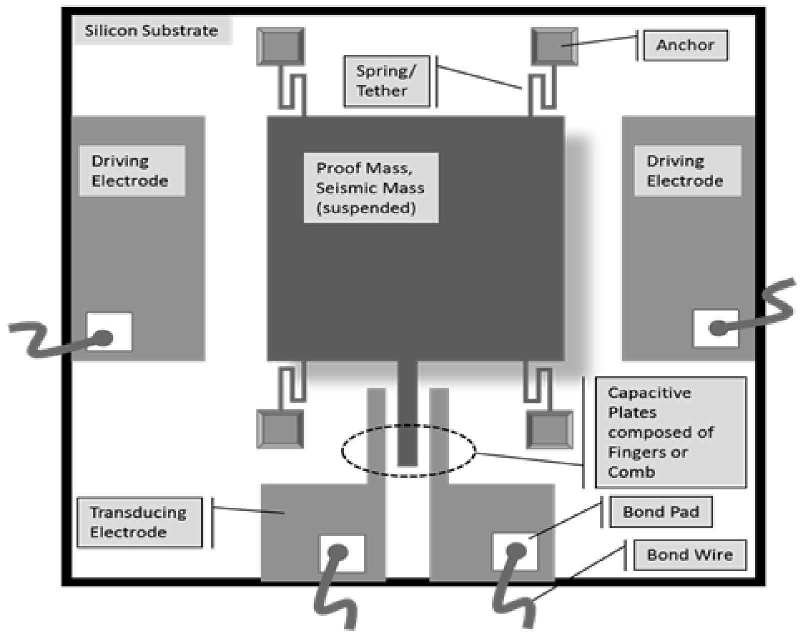


Figure 23 — Example of a MEMS transducer (top-view)

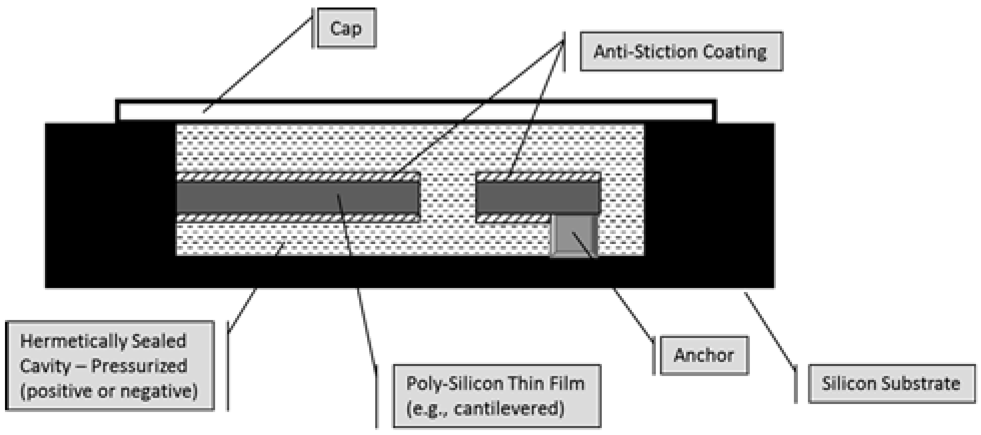

Cap
Anti-Stiction Coating
Hermetically Sealed Cavity – Pressurized (positive or negative)
Poly-Silicon Thin Film (e.g., cantilevered)
Anchor
Silicon Substrate

Figure 24 — Example of a MEMS transducer (side-view)

Some common mechanical root causes of failures and the associated failure modes of MEMS transducers are listed in Table 55, which is not exhaustive.

Table 55 — Examples of Root Causes & Associated Modes of MEMS Transducer Failures [46]

Mechanical Root Cause	Transducer Failure Mode	Description
Fractured spring	Non-parametric Sensitivity	MEMS motion transducers are typically designed with a collection of springs to provide mechanical positioning, establish linear sensitivity, and limit the travel. If a spring in the collection fractures, the proof mass becomes unbalanced such that portions of the travel appear normal, but the portion nearest the fractured spring would be relaxed or potentially unlimited, resulting in non-linear sensitivity.
Fractured finger	Sensitivity shift	MEMS motion transducers are typically designed with multiple sets of capacitive interdigitated fingers for sensing the proof mass movement. The sensitivity is proportional to the total device capacitance, which is the summation of each of the individual finger capacitances. If a finger fractures, the total capacitance is reduced, resulting in a decrease of sensitivity.
Cavity seal breech		The gap between the fingers provides an aerodynamic dampening due to the sealed gas molecules inside the MEMS cavity structure. The sensitivity is proportional to the pressure of the sealed gas. If the seal is breached, the pressure reduces, resulting in an increase of sensitivity.
Fractured diaphragm	Offset shift, Stuck-at	MEMS pressure transducers are typically designed as diaphragms, either to exert a strain on piezo-resistive elements or to change the capacitive gap. If the diaphragm fractures, an offset or a complete loss of sensitivity can occur, resulting in a stuck-at ground fault.
Fractured Anchor		MEMS motion transducers are typically designed with anchors for the springs, or with similar structures used to limit travel distance. If the anchor, or travel-limiter fractures, the proof mass becomes misaligned or travels outside of the allowable boundary coming in contact with the inner surfaces of the cavity, resulting in a stuck-at fault.
Particle(s)	Sensitivity shift Non-parametric Sensitivity Offset shift Stuck at	A particle is capable of introducing multiple failure modes depending on the conductivity of the particle and the individual parts of the transducer that it is contacting. If a particle is conductive, it can short parts together and if it is resistive, it can impede the movement of the parts. Particles can also account for transient faults and general unpredictability if the particle is free to move within the cavity. Particles can be generated during production processes or due to breakage/wear during operation.
Anti-Stiction coating anomaly	Sensitivity shift Non–parametric Sensitivity Stuck-at	Capillary or electrostatic forces cause suspended/cantilevered surfaces to become stuck to other moving surfaces or to fixed surfaces due to anomalies of coatings used to prevent such effects.
General Mechanical Overstress	Sensitivity shift Non-parametric Sensitivity Offset shift Stuck at	Sources of mechanical overstress can include shock, fatigue, vibration, corrosion or the effects of electrical overstress (EOS) or electrostatic discharge (ESD) that result in structural damage to MEMS transducer parts or sub-parts.

5.5.3 Safety analysis

5.5.3.1 Considerations in the determination and allocation of base failure rate

There can be specific challenges in determining the failure rate of integrated transducers and allocating base failure rates to transducers and supporting circuitry. The following points are considered when conducting a quantitative analysis:

— Passive transducers that take up a substantial percentage of die area which also includes active circuitry;

EXAMPLE 1 Hall cell based sensor.

NOTE 1 There can be a disparity in the failure rates between active and passive elements as well as those devices with larger versus smaller geometries.

— Transducers that are manufactured on top of active circuitry taking no area of the active die;

EXAMPLE 2 GMR (giant magnetoresistance).

— Handbooks do not typically cover MEMS elements since the technology has been subject to rapid advances;

— Transducer failure rate distribution is structure dependent;

EXAMPLE 3 MEMS for pressure sensor with a cavity, relatively large diaphragm, and small piezo-electrical conversion element.

— Transducers can be assembled with no supporting circuitry and it is therefore not possible to apply commonly used reliability standards to determine base failure rate;

— For new technologies, field data is not available and reliability data is limited; and

— Failure rates for the transducers versus supporting circuitry can be derived from different sources.

NOTE 2 Appropriate scaling is applied if the failure rates are not from same source and conditions.

In each case, the method of determining the base failure rate of a sensor and how the failure rate is allocated to the transducer element is based on a sound and documented rationale.

EXAMPLE 4 The following is an example of a method for determination of failure rate for new MEMS transducer (no field/reliability data):

1) Begin with a failure model of an established MEMS device that includes overall failure rate, failure mechanisms (e.g. particles, stiction, cavity breach) and distribution based on established data (e.g. field return or other similar reliability source);

2) Establish the baseline failure rate for each failure mechanism;

3) For each failure mechanism, assign a susceptibility factor that compares the transducer under design/evaluation to the transducer used to derive the data in steps 1 and 2 above. This susceptibility factor assesses the relative risk between the reference transducer(s) and the transducer under evaluation, e.g. higher, lower or the same;

4) Combine the data from steps 2 and 3 to produce a weighted failure rate for each failure mechanism for the transducer under evaluation; and

5) Apply the failure mode distribution from step 1 to generate a single predicted failure rate for the new MEMS transducer.

NOTE 3 This is an example method only. The procedures defined are neither exhaustive nor restrictive nor restricted to MEMS and are assumed to be based on a rationale that has been documented and substantiated with appropriate evidence.

5.5.3.2 Dependent failures analysis

Dependent failures analysis is performed according to the flow described in 4.7, if independence or freedom from interference is required. Table 56 gives examples for dependent failures initiators for various types of sensors.

Table 56 — Dependent failures initiators for sensors/transducers

DFI classes defined in 4.7.5	Examples
Dependent failures initiators due to random hardware faults of shared resources	Common calibration and/or configuration resources (e.g. eFUSE to control the CMOS based image sensor)
Dependent failures initiators due to random physical root causes	Temporal Noise or Fixed Pattern Noise
Systematic dependent failures initiators due to environmental conditions	Extended exposure to excessive heat, humidity, or strong sunlight Electrostatic discharge
Systematic dependent failures initiators due to development faults	Wrong design of image sensor
Systematic dependent failures initiators due to manufacturing faults	Sensor manufacturing defects
Systematic dependent failures initiators due to installation faults	Magnetic sensor target wheel mounted off axis (runout) Incorrect positioning of mirror in image sensor

Methods to evaluate the effectiveness of controlling or avoiding dependent failures for sensors and transducers can be derived from the exemplary methods described in 4.7.5.2.

5.5.3.3 Quantitative analysis

There are no procedural differences in the quantitative analysis concerning the evaluation of hardware architectural metrics and the evaluation of safety goal (requirement) violations due to random hardware failures for a sensor compared to any other hardware element.

The significant difference is related to the inclusion of the transducer element within the analysis since violations of sensor safety requirements are significantly related to failure modes of the transducer elements. The following points are considered for the inclusion of the transducer within a quantitative analysis:

— Level of granularity (how it is categorized into part and/or sub-parts);

— Quantified failure rate and derived source;

— Failure mode distribution; and

— Inclusion of sensor specific safety mechanisms (see 5.5.4).

Quantitative analysis is conducted for the electrical failure modes of the semiconductor part and the mechanical part according to ISO 26262-5:2018, Clause 8 and ISO 26262-5:2018, Clause 9.

Quantitative analysis of supporting circuitry is conducted according to guidance contained within 5.1 for digital circuitry and 5.2 for analogue.

5.5.4 Examples of Safety Measures

Table 57 provides examples of safety mechanisms that are commonly used with sensors/transducers that support the unique role of the transducer element in evaluating the environment.

Because a sensor can include a wide range of supporting circuitry both in quantity and type, these safety mechanisms are in addition to any analogue or digital safety mechanisms contained in 5.1 for digital, 5.2 for analogue, 5.3 for PLD, 5.5 for Sensors and Transducers safety mechanisms respectively.

The examples included in Table 57 are not exhaustive and other techniques can be used. Evidence is provided to support the claimed diagnostic coverage.

NOTE It is not possible to give a general guidance on the DC for sensors/transducers because it strongly depends on the specific technology, type of circuit, use case.

Table 57 —Example of Safety Mechanisms for Sensors/Transducers

Safety mechanism/measure	See overview of techniques	Notes
Sealed Proof mass Filter	5.5.4.1	MEMS specific implementation.
Redundant Diaphragms	5.5.4.2	MEMS specific on-chip calibrated reference.
Offset cancellation	5.5.4.3	Allows for offset optimization.
Transducer specific self-test	5.5.4.4	Various methods to test signal path integrity.
Automatic Gain Control	5.5.4.5	Accounts for low levels of environmental stimulus and increases dynamic range.
Sensitivity adjustment	5.5.4.6	Allows for sensitivity centering.
MEMS specific non E/E safety measures	5.5.4.7	Measures that assess physical properties of MEMS transducers.

5.5.4.1 Sealed Proof Mass Filter

Aim: To provide a low-pass filter mechanism which rejects noise that could otherwise alias into the band of interest. Commonly used on MEMS accelerometer transducers.

Description: A proof mass chamber sealed with greater than atmospheric pressure dampens the environmentally induced movement of MEMS transducer parts.

EXAMPLE A MEMS transducer can consist of groups of ‘comb’ fingers with a gap defined at a close tolerance. As the proof mass chamber is sealed under pressure, the ambient gas provides a squeeze-film dampening effect, similar to a shock absorber, filtering the high frequency vibrations. Higher pressures trap more gas molecules and, in effect, lower the cut-off frequencies. Lower pressures trap fewer gas molecules allowing higher cut-off frequencies.

5.5.4.2 Redundant Diaphragms

Aim: To provide a permanent reference with which to compare to the primary transducing element of the system.

Description: Inclusion of a reference transducer to allow comparison of the primary sensing diaphragm which is allowed to displace due to environmental factors to an equal but non-moving diaphragm. Commonly used on MEMS pressure transducers.

EXAMPLE A MEMS transducer could be fabricated with a non-moving 'twin' that is formed at the same time under the same process steps and critical dimensions, and would be subject to the same process tolerances. As such, common variables such as sensitivity due to temperature or applied voltage would be shared and mathematically cancel each other, leaving the moving vs. non-moving reaction as the only remaining difference when sampled.

5.5.4.3 Offset Cancellation

Aim: To minimize offset in the transducer output.

Description: There are various hardware and software methods that can be employed to cancel the built in offset caused by non-ideal characteristics of a transducer. The chosen method will depend on the type of transducer used.

EXAMPLE A linear magnetic sensor provides a specified quiescent voltage of VCC/2 in the absence of a magnetic field. A calibration routine is run on each power-up cycle to quantify the offset voltage with no magnetic stimulus. This value is stored and used to adjust readings taken during operating mode.

5.5.4.4 Transducer specific self-test

Aim: To provide a means of evaluating a specific type of transducer.

Description: Because transducers respond to the environment, it can be challenging to evaluate the integrity of a sensor/transducer in the absence of the environmental condition. There are variou s ways to stimulate a transducer by self-test and the accuracy and availability of these tests depend upon the specific type of transducer used and technical specification being evaluated. In general, the test is set up to evaluate the integrity of the entire signal path or to isolate a clause of the signal path such as the analogue front end close to the transducer or the digitally processed back end.

EXAMPLE A MEMS transducer could contain two sets of sense electrodes, electrically connected in opposite polarity. Summing of the two absolute values is set to zero (within specified tolerances) independent of the MEMS mechanical movement. A value outside of the allowable zero range would indicate an imbalance or fracture of the proof mass or sensing electrode integrity.

5.5.4.5 Automatic gain control

Aim: To support sensor functionality over low levels of environmental stimulus.

Description: Typically, the electrical output of transducers must be amplified in order to be further utilised in a sensing system. Automatic gain control, or AGC, allows for the gain of transducer amplification to be adjusted based on the amplitude of the transducer output signal. At low transducer output levels, the gain is increased and at higher transducer output levels, the gain is decreased to allow for greater dynamic range.

5.5.4.6 Sensitivity adjustment

Aim: To maintain sensitivity within its specified range

Description: The sensitivity of a sensor/transducer must be within its specified range over the operating temperature range of the sensor in order to ensure an accurate output. There are various methods to adjust the sensitivity of a transducer in order to account for environmental fluctuations.

EXAMPLE 1 The use of a micro-heater activated by current to maintain sensitivity of MEMS parts over temperature [47].

EXAMPLE 2 The modification of bias current through a hall cell to maintain sensitivity over temperature.

EXAMPLE 3 The application of an electrostatic potential to MEMS fingers which electrically dampens movement and decreases sensitivity when applied.

5.5.4.7 MEMS specific non E/E safety mechanisms

Aim: To provide mechanical safety mechanisms specific to MEMS transducer parts

Description: In most cases, detection of a non-electrical failure in the transducer by electronic means (after the transducer interface of the signal chain) is done based upon estimations of the effect of failures upon the signal itself. In these cases, direct observation of the failure is typically not possible, therefore only inferences can be used to determine if the transducer has experienced a failure (see Figure 25b). The nature of this inferential method can be susceptible to incorrect or missed detections.

EXAMPLE In-range faults of the transducer.

It is plausible that methods and technologies other than post-transduction electrical or electronic technologies can be permanently employed within a MEMS transducer to directly detect or control failure modes within the transducer itself (see Figure 25a). For example, additional mechanical or optical mechanisms (e.g. [45], [46]) can be used within the transducer as safety mechanisms such as a simple stop or floating cantilevered finger.

These simple mechanical mechanisms can optionally include a separate signal output to allow the transducer to enter a safe state upon detection of a failure mode thereby eliminating the transducer as the DFI of a specific safety goal or hardware requirement in a system. This would be in addition to any dedicated measures or traditional E/E safety mechanisms and could potentially provide coverage against both random and systematic faults within the transducer.

Such non-E/E safety mechanisms could be defined in the application of diagnostic coverage. The level of diagnostic coverage afforded by a non-E/E safety mechanism for a specific use case would require sound engineering evaluation by domain experts to derive the proper value with each rationale and verification activity fully documented and included in the safety case. Once verified and validated, such non-E/E safety mechanisms in a component can contribute to the system or element achieving the ASIL of a given safety requirement or safety goal.

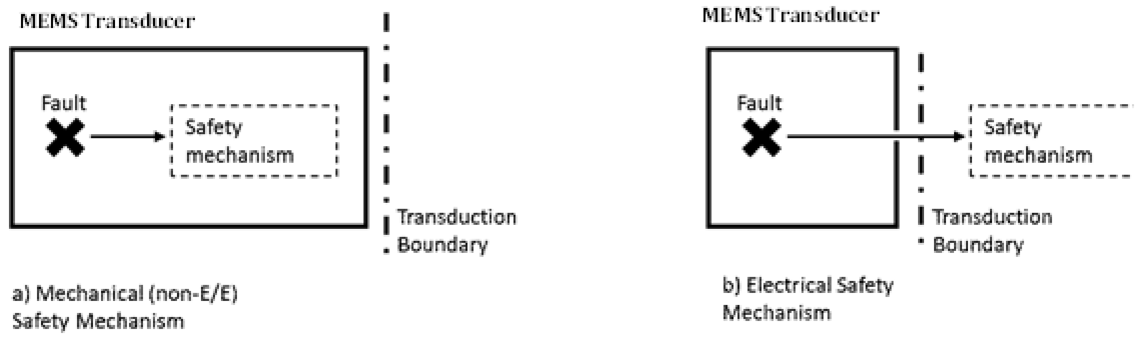


Figure 25 — Distinction between mechanically detected and electrically inferred transducer faults

5.5.4.8 Dedicated measures for sensors

As described in ISO 26262-5:2018, 9.4.1.2 and 9.4.1.3, dedicated measures can be considered to ensure the failure rate claimed in the evaluation of the probability of violation of safety goals or requirements.

Example of dedicated measures for sensors and transducers include:

— Overdesign of parts or sub-parts of sensor or transducer for robustness (e.g. electrical or thermal stress rating);

— A special sample test or 100 % production test of a critical sensor or transducer specification to reduce the risk of occurrence of the failure mode;

— Layout related measures;

EXAMPLE 1 Quad hall cell configuration to minimize stress related offsets.

— Bond pad order that minimizes opportunity for interaction;

EXAMPLE 2 Common-mode stray capacitance or current leakage affecting switch capacitance proof mass movement.

— Technology measures.

EXAMPLE 3 Use of wet etch instead of dry etch technique for the removal of buried oxide layer resulting in smoother surfaces and increased strength of MEMS parts [47].

— Other measures relating to sensors/transducers.

5.5.5 About avoidance of systematic faults during the development phase

In addition to what is described in 5.1.9 and 5.2.5 for digital and analogue components, the measures described in Table 58 can be adopted for sensors and transducers.

Table 58 — Example of techniques or measures to achieve compliance with ISO 26262-5:2018 requirements during the development of a sensors or transducers

ISO 26262-5:2018 requirement	Design phase	Technique/Measure	Aim
7.4.4 Verification of hardware design	**Verification**	Verification of internal interfaces	To verify by means of dedicated tests the correct integration between mechanical, electro-mechanical, opto-electrical, magnetic part of the sensor or transducer and the related analogue and/or digital part.
10.5.1 hardware integration and testing activities	**hardware integration and testing**	Testing of influences of package	To test the influences of package (including for example supports like mirrors) to the sensor / transducer characteristics.
7.4.4 Verification of hardware design	**Design**	Finite Element Analysis (FEA)	To mitigate influences of induced stress
7.4.3 Safety Analyses	**Design**	FMEA	To consider the completeness and correctness of the transducer failure model including failure modes, distributions and their effects on sensor output
7.4.2.4 Robust design principles	**Design**	Design for manufacturing	To consider manufacturing process variations on sensor/transducer electrical characteristics in order to increase robustness.
7.4.4 Verification of hardware design	**Design**	Design for testability	To design in necessary hardware to allow for full evaluation of transducer performance and sensor/transducer safety mechanisms.
7.4.5 Production, operation, service and decommissioning 9.4.2.5 Dedicated measures	**Safety-related special Characteristic s during Chip production**	Optical pattern inspection to detect and cull early failures	Specific layers of the semiconductor process are optically compared to reference geometries in order to detect patterning anomalies.
10.5.1 hardware integration and testing activities	**Qualification of hardware component**	Environmental testing to simulate actual operating conditions	Extended reliability testing is performed that simulates environmental conditions of use e. g. vibration test.
10.5.1 hardware integration and testing activities	**hardware integration testing**	Unique test for sensors with environmental stimulus	Having the ability to expose sensor/transducer to the environmental stimulus that it is sensing (measurand) e.g. acceleration, magnetic field, pressure

5.5.6 Safety documentation for sensors/transducers

Safety documentation for sensors and transducers is produced in line with the documentation described for digital (see 5.1.11) and analogue components (see 5.2.6). It includes:

— Base failure rates, including the assumptions and the rationale with which they have been estimated;

NOTE It is useful if the base failure rate shows how the failure rate is distributed over the different fault models that can affect the sensor and transducer.

EXAMPLE In the case of an image sensor based camera, the percentage with which a fault in the pixel array can affect a single pixel, a whole column, a whole row, many pixels or the full array is provided.

— The list of transducer failure modes, with end effect (at sensor output), and failure mode distribution; and

— User information such as safety manual or safety application note, with specific emphasis to:

— Safety mechanisms integrated in the device and their availability;

— Configuration or calibration parameters (and related procedures) that can influence the safety characteristics of the device; and

— Production related instructions affecting functional safety.

Annex A (informative) Example on how to use digital failure modes for diagnostic coverage evaluation

A.1 Example of evaluation of a DMA safety mechanism

A.1.1 Description of the use case

The following is the DMA use case considered in this example:

- A message is received by a communication peripheral every X ms;
- As soon as the message is received by the communication peripheral it triggers a DMA request;
- The DMA transfers the message from the peripheral receive buffer to a RAM region;
  - The transfer is always to the same RAM region, independent from the message content
- After the DMA is finished with the transfer it triggers a CPU interrupt; and
- The CPU copies the message into a different buffer within the RAM depending on the message ID.

A.1.2 Description of the safety mechanisms

In this example, the following safety mechanisms are available to monitor the correct DMA activity:

- SafMech_01_DMA_MPU: Dedicated memory protection unit defining the memory regions which are accessible via DMA:
  - Write access is restricted to the destination addresses; and
  - Read access is restricted to the source addresses.
- SafMech_02_E2E_Protection:
  - The DMA transfers messages which are end-to-end protected by:
    - A 8 bit CRC over the data content, the message ID and the message counter;
    - Message ID (4 bit); and
    - Message counter (4 bit).
  - Out of the $2^4 = 16$ message IDs only 12 are valid;
  - The counter is reset to zero after reaching its maximum value of 0xF; and
  - The message is copied to a different RAM region by the CPU after receiving the data transfer complete signal. This memory region is not accessible by the DMA. The E2E protection mechanisms are checked after the copy operation by the CPU. The application only uses this copy; it does not use the data in the destination address of the DMA.
- SafMech_03_Timeout_Mon: The data transfer is supposed to occur periodically. The frequency is known by the system. It monitors if a data transfer occurs within the specified time frame; and
- SafMech_04_IR_Source_Mon: In the case of an interrupt request this safety mechanism checks if the trigger came from a legal source.

A.1.3 Definition of the failure modes and estimation of diagnostic coverage

Based on the described use case and safety mechanisms, the following failure modes are defined and the following values for diagnostic coverage can be estimated.

A.1.3.1 DMA_FM1: no requested data transfer

This failure mode is detected by SafMech_03_Timeout_Mon since there is no data transfer completed signal within the specified time frame. The FMCDMA_FM1 is estimated as 100 %.

A.1.3.2 DMA_FM2: data transfer without a request

The DMA transfers data from the source to the destination address. It signals the data transfer completion. Depending on the content of the source address this could be a previous message (DMA_FM2.1) or a random value (DMA_FM2.2; modelled as "white noise" i.e. each possible error state is equally probable).

In more detail:

— DMA_FM2.1: The previous message will be detected via the message counter or the message ID of the E2E protection (SafMech_02_E2E_Protection). The FMCDMA_FM2.1 is estimated as 100 %;

— DMA_FM2.2: In the case of a random value:

  — The probability pCRC,legal of randomly matching a legal CRC value is $1/2^8$;

  — The probability pID,legal of randomly matching a legal ID is 12/16;

  — The probability pCounter,legal of randomly matching the correct counter value is $1/2^4$ (since only one of the 2^4 values is the correct one);

  — The overall probability pRF that no error is triggered is pRF = pCRC,legal * pID,legal * pCounter,legal = 0,000183; and

  — The FMCDMA_FM2.2 is estimated as 1 - pRF so equal to 99,98 %.

To derive an accurate estimation of the overall failure mode coverage FMCDMA_FM2 the failure mode distribution between the two failure modes DMA_FM2.1 and DMA_FM2.2 needs to be estimated.

Since here both values are very high and very close to each other the effort of estimating the failure mode distribution of these two failure modes is omitted and just the lower value is used: FMCDMA_FM2 and FMCDMA_FM2.2 are estimated as 99,98 %.

A.1.3.3 DMA_FM3: data transfer too early/too late

For the evaluation the failure modes needs to be further elaborated:

— DMA_FM3.1: The data transfer is triggered before the correct request. This failure mode is equivalent to DMA_FM2 and is not further evaluated here. FMCDMA_FM3.1 is estimated as 99,98 %;

— DMA_FM3.2: The data transfer is triggered too late after the correct request. Depending on the delay the effect could be one of the following:

  - DMA_FM3.2a: Depending on the communication peripheral either the message gets overwritten by the following message before it is fetched by the DMA or the following message cannot be received. Both cases result in a loss of a message. This will be detected by either by SafMech_03_Timeout_Mon or by SafMech_02_E2E_Protection (via the message counter) with a FMC = 100 %. Depending on the communication peripheral additional error signals can be generated by the communication peripheral itself;

  - DMA_FM3.2b: During the fetch operation by the DMA the next message is received partially overwriting the previous one. This results in a corrupted message consisting partly of the two messages:

    - The ID is legal (pID,legal = 1);

    - The counter of the successive message could have a high probability of being the same as the counter of the predecessor message (if both messages have the same transmission frequency). Here the worst case probability of pCounter,legal = 1 is assumed;

    - The data corruption is modelled as "white noise" rendering a probability pCRC,legal of randomly matching a legal CRC value of $1/2^8$; and

    - FMC = 1 - pCRC,legal * pID,legal * pCounter,legal = 99,6 %.

    - Depending on the communication peripheral;

      - additional error signals can be generated, increasing the effective FMC, or

      - this failure mode is not possbile, leaving only DMA_FM3.2a.

  For an accurate estimation of FMCDMA_FM3.2 the failure mode distribution between DMA_FM3.2a and DMA_FM3.2b would need to be derived. For a conservative first estimation the lower FMC of the two can be used: FMCDMA_FM3.2 = 99,6 %.

- DMA_FM3.3: The data transfer completed signal is provided before the transfer is complete. This would result in a partially corrupted message where the message in the destination buffer consists out of a mix of two messages. As far as detection by SafMech_02_E2E_Protection is concerned the argument is analogue to DMA_FM3.2b FMCDMA_FM3.3 is estimated as 99,6 %;

- DMA_FM3.4: The data transfer completed signal is provided too late after the transfer is complete. This failure mode can lead to:

  - DMA_FM3.4a: The message being overwritten by the successive message before the CPU can fetch it. This results in a loss of message and is detected by either by SafMech_03_Timeout_Mon or by SafMech_02_E2E_Protection with a FMC = 100 %. This is analogue to DMA_FM3.2a; and

  - DMA_FM3.4b: The message is overwritten by the DMA during the fetch by the CPU. This results in a partially corrupted message. FMC = 99,6 % (analogue to DMA_FM3.2b).

  With the same argument as before the overall FMCDMA_FM3.4 can be estimated as 99,6 %.

A.1.3.4 DMA_FM4: incorrect output

In contrast to the previous failure modes which were timing related this failure mode addresses incorrect outputs but with the right timing. In this example the DMA has following outputs:

- Control signal: read or write;

— Control signal: access width (8 bit, 16 bit, 32 bit);

— Control signal: address to be accessed;

— Data (in the case of writes); and

— Four different interrupt request signals.

Following sub failure modes can be distinguished:

— DMA_F4.1a: read instead of write;

  — Instead of writing to the RAM destination the DMA will execute a read access from this address. There will be no more updates of the messages. After the "transfer" the DMA still triggers the CPU interrupt request. The old message will be detected by SafMech_02_E2E_Protection either by checking the ID or by checking the counter. In addition SafMech_01_DMA_MPU will detect an illegal access (read instead of a write). $FMC_{DMA\_FM4.1a}$ is estimated as 100 %.

— DMA_F4.1b: write instead of read;

  — Write instead of read: The DMA will perform a write access to the communication peripheral instead of a read access. Depending on the communication peripheral this can already lead to an error reaction by the communication peripheral. In addition the illegal write access will be detected by SafMech_01_DMA_MPU. $FMC_{DMA\_FM4.1b}$ is estimated as 100 %.

— DMA_F4.2: incorrect access width;

  — Incorrect access width: This failure mode will result in a corrupted message, which is detectable via the CRC of SafMech_02_E2E_Protection. ID check and illegal message counter can also lead to an error detection (see also SafMech_01_DMA_MPU). $FMC_{DMA\_FM4.2}$ is estimated as 99,6 %.

— DMA_F4.3: incorrect access address;

  — Incorrect access address: This failure mode will lead to the access of an illegal address by the DMA and will be detected by SafMech_01_DMA_MPU. $FMC_{DMA\_FM4.1b}$ is estimated as 100 %.

— DMA_F4.4: incorrect data output; and

  — Incorrect data output: This failure mode will lead to randomly corrupted message, similar to DMA_FM2.2. FMCDMA_FM4.4 is estimated as 99,98 %.

— DMA_F4.5: incorrect interrupt request

  — Incorrect interrupt request: In this example the DMA triggers just one CPU IR request. Therefore SafMech_04_IR_Source_Mon will detect this fault. $FMC_{DMA\_FM4.5}$ is estimated as 100 %.

Annex B (informative) Examples of dependent failures analysis

B.1 Microcontroller example

B.1.1 Description

The microcontroller component described in Figure B.1 is used to illustrate the dependent failures analysis methodology for a digital component.

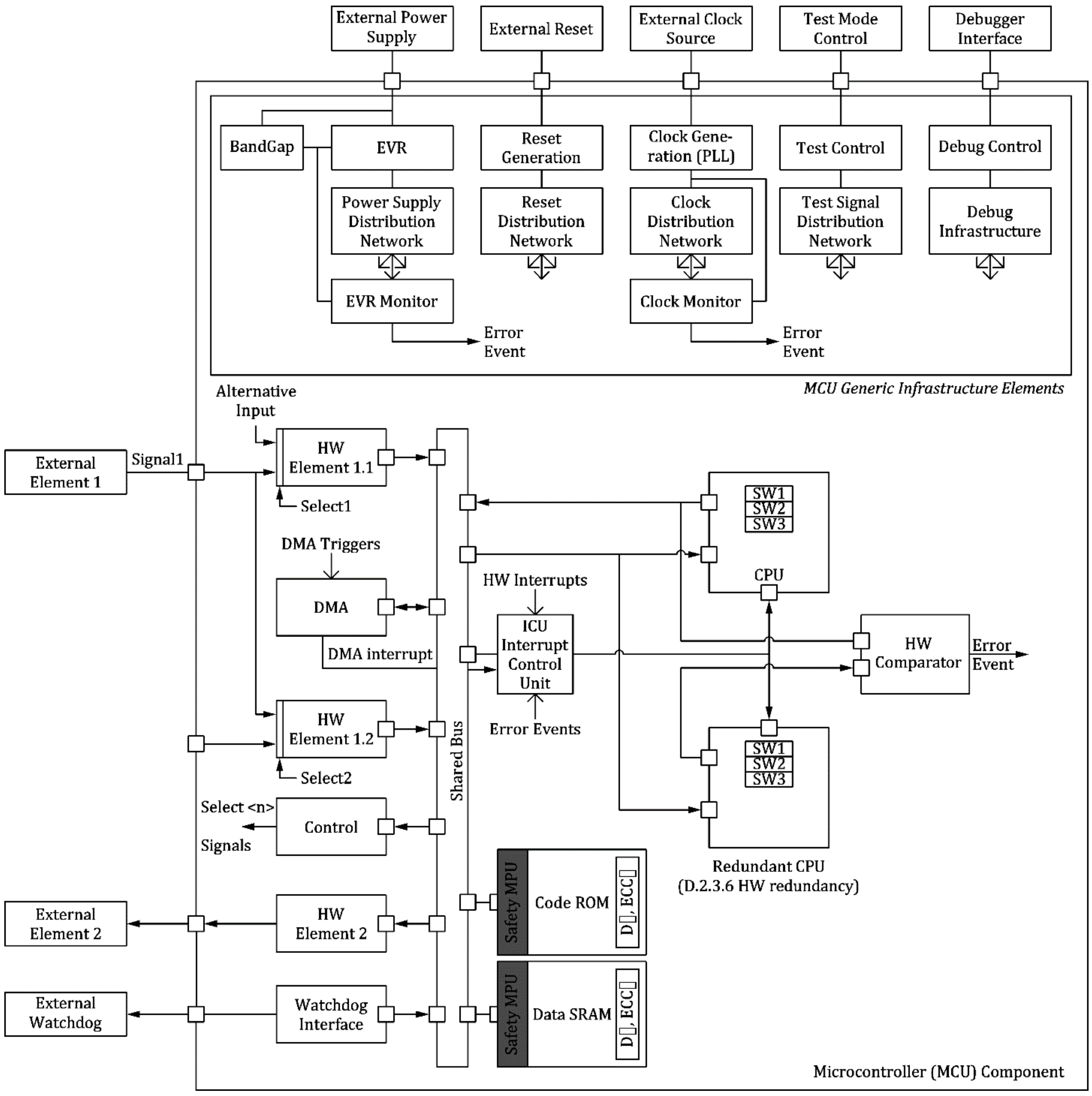


Figure B.1 — Microcontroller component example

First an introduction to the hardware and software elements is done to highlight the hardware safety mechanisms that are going to be used for the DFA. It is not in the scope of this example to provide a comprehensive specification of the hardware safety requirements and the safety mechanisms.

— hardware Element 1.1: Interface processing element that enables to receive information from hardware elements connecting to the Microcontroller (e.g. Signal 1 from External Element 1 in)

- hardware Element 1.2: Interface processing element identical to hardware Element 1.1 from a functional point of view
- hardware Element 2: This element is used to control the External Element 2
- Control: This element provides the select signals that enable to control the connectivity of hardware Element 1.1 and 1.2 with different input interfaces of the microcontroller.
- CPU: Central Processing Unit where software elements are executed
- Data SRAM: Memory where software elements store their own private variables. It also contains communication buffers between software and DMA and between software elements themselves.
- Code ROM: Read-only Memory containing the code that is executed by the software elements and possibly constant data used by the software elements.
- Software Elements: In this example three software elements are listed: software1, software2 and software3.
- Watchdog Interface: It enables to communicate with an external watchdog hardware element.
- Shared Resources: The following shared resources are identified:
  - DMA (Direct Memory Access) hardware Element: The DMA can be used by each software elements and has read and write access to any addressable resource (Memory, Configuration Register)
  - EVR (Embedded Voltage Regulator): The EVR provides the power supply to each hardware element inside the microcontroller with the exception of the input/output pads that are powered by the “External Power Supply”.
  - Reset Generation & Distribution: Controls the reset state of the microcontroller based on reset commands originating from the external reset source or internal reset actions controlled by hardware or software elements
  - Clock Generation & Distribution: Delivers the intended clocks for each hardware element based on a PLL using an External Clock Source”.
  - Test Logic: Test structures required for the production tests of the microcontroller.

Functional safety concept and requirement concept: Signal S1 is an analogue signal that indicates the state of an actuator. An unintended state shall be recognized and shall lead to the de-activation of the actuator: this is considered to be the safe state. For that purpose the Signal S1 is converted into digital information and then processed by a software element software1 to identify a possible hazardous state of the actuator. The software element software2 is responsible to redundantly acquire information from Hardware Element 1.1 and 1.2. The main task of software2 is to control the DMA to fetch the conversion results from Hardware Element 1.1 and 1.2 and store them in separated data sets in a shared buffer located in Data SRAM. DMA informs software2 about the completion of transfers by sending an interrupt to the ICU. Upon reception of this event software2 compares the plausibility of the data sets and in the case of mismatch it provides pre-defined error information to software1. The software element software3 is responsible for a periodic refresh of the external watchdog. The refresh requires sending a dynamic code with a given sequence. The code to be sent is only provided by software element software1. If software3 fails to refresh the watchdog or sends an incorrect code, the external watchdog enters timeout state that leads to the de-activation of the actuator.

This annex provides exemplary safety requirements. The specification of the set of safety requirements is reduced to a minimum set that is suitable for the DFA.

- MCU-REQ-1: Faults during the processing of Signal 1 by hardware element 1.1 shall be detected within 20 milliseconds [ASIL X]
  - MCU-REQ-1.1: Signal 1 shall be redundantly processed by hardware Element 1.2
  - MCU-REQ-1.2: Results of hardware Element 1.1 and 1.2 shall be monitored by software. In the presence of a mismatch software shall send an error message to the external watchdog through the watchdog interface.
- MCU-REQ-2: Random hardware fault leading to a wrong output of CPU shall be detected within 20 milliseconds [ASIL X]
  - MCU-REQ-2.1: CPU shall be monitored by a redundant CPU. Outputs of CPU and Redundant CPU shall be compared every clock cycle by an hardware comparator
  - MCU-REQ-2.2: In the presence of a mismatch between CPU and Redundant CPU an error event shall be generated

B.1.2 Dependent failures analysis

The DFA will only focus on the DFI that have the potential to lead to a violation of the safety requirement MCU-REQ-2. The analysis will follow the proposed workflow. To simplify the analysis not each step will be considered. With respect to the requirements MCU-REQ-2, this step focuses on analysing the architecture focusing on steps B1 and B2 of the DFA workflow. The analysis is supported by a qualitative fault tree (see Figure B.2) that identifies the shared resources and the redundant elements.

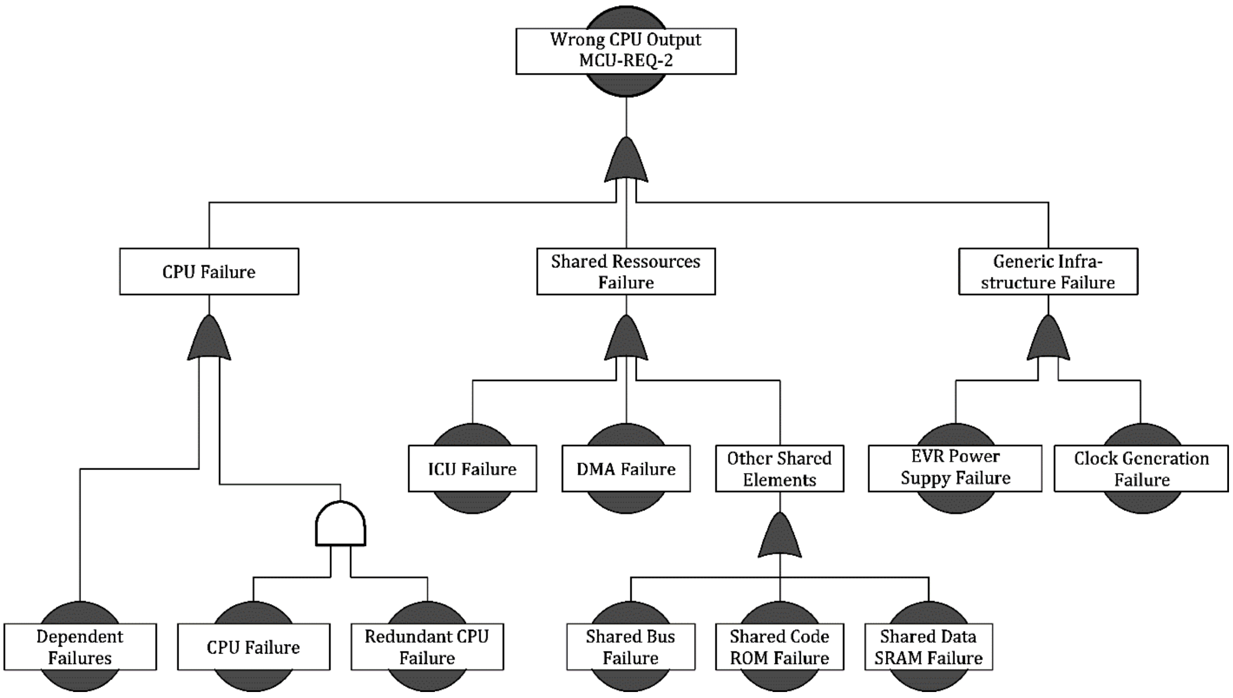


Figure B.2 — Shared elements overview

For the shared resources, each failure base event or AND gate needs to be analysed on its own. For the CPU and redundant CPU a base event Dependent Failures has already been introduced because the safety mechanism is already visible on the proposed architectural level. It is recommended to analyse the Generic Infrastructure Elements that have a global effect separately, in order to avoid considering them for each shared element independently. This is possible for the power supply and clock generation because they have own safety mechanisms. However for the Reset Generation, Test Signals and Debug Infrastructure it is necessary to analyse them at a lower level where their influence to the shared elements safety mechanisms can be analysed. For the Generic Infrastructure Elements the analysis will concentrate on the power supply and clock generation.

Table B.1 shows an example for a microcontroller DFA.

Table B.1 — DFA for microcontroller example

ID	Element	Redundant element	Dependent failures initiators			DFA	
	Short name and description	Short name and description	Systematic faults	Shared resources	Single physical root cause	Measure for fault (A)voidance or (C)ontrol	Verification method
				Generic Infrastructure Elements			
PS1	Power Supply	Power Supply Monitor: Measurement of voltage levels within operating conditions		Shared bandgap has the potential to lead to undetected over voltage.		(C) Add a bandgap monitor	Silicon-level robustness test

PLL1	Clock	Clock Monitor Frequency Measurement		Shared Input Frequency has the potential to prevent accurate Frequency measurement.		(C) Add an independent clock source (Oscillator) to measure the PLL frequency (A) Design dissimilarity: dissimilarity between drift behaviour of PLL and drift behaviour of reference oscillator used by clock monitor thanks to different implementation.	Design Inspection Silicon-level robustness test
PLL2	Clock	Clock Monitor Frequency Measurement		Loss of clock that prevents monitor to report failure condition		(C) Semiconductor monitoring by External Watchdog Function.	
PLL3	Clock	Clock Monitor Frequency Measurement			Needs to be analysed based on a detailed block diagram of the clock generation and clock monitoring where the relevant interfaces, sideband signals and configuration registers are visible.	<other measures>	<other verification methods>

3755

3756 **Table B.1 *(continued)***

ID	Element	Redundant element	Dependent failures initiators			DFA	
	Short name and description	Short name and description	Systematic faults	Shared resources	Single physical root cause	Measure for fault (A)voidance or (C)ontrol	Verification method
Processing Elements							
CPU1	CPU, Computation	Redundant CPU + Hardware Comparator		Power Supply		Covered by Power Supply Analysis	
CPU2	CPU, Computation	Redundant CPU + Hardware Comparator		Clock: incorrect frequency		Covered by PLL Analysis	
CPU3	CPU, Computation	Redundant CPU + Hardware Comparator		Clock: clock glitch		Other measures	
CPU4	CPU, Computation	Redundant CPU + Hardware Comparator		Shared Bus		Other measures	

CPU5	CPU, Computation	Redundant CPU + Hardware Comparator		Data SRAM		Safety Mechanisms for Data SRAM (e.g. EDC-ECC) are covered by Safety Analysis. EDC-ECC is evaluated by redundant CPU enabling to control this dependent failures related to interface to Data SRAM.	
CPU6	CPU, Computation	Redundant CPU + Hardware Comparator		Code SRAM		Other measures	
CPU7	CPU, Computation	Redundant CPU + Hardware Comparator		ICU		Other measures	
CPU8	CPU, Computation	Redundant CPU + Hardware Comparator			Short-circuit between signals belonging to CPU and signals belonging to Redundant CPU	(A) Physical separation according to technology design rules	Analysis of design rules Physical Layout inspection
CPU9	CPU, Computation	Redundant CPU + Hardware Comparator			Latch-up affecting logic belonging to CPU and logic belonging to Redundant CPU	(A) Physical separation according to technology design rules for isolation of standard cells against latch-up (A) Physical separation related to Soft-Error Induced Latch-up	Analysis of design rules Physical Layout inspection

After the architectural enhancements resulting from the DFA the microcontroller component block diagram is updated to show:

— the new Bandgap Monitor element to mitigate the dependent failures related to the Bandgap drift failure mode; and

— the new Oscillator element to mitigate the dependent failures related to the clock drift failure mode.

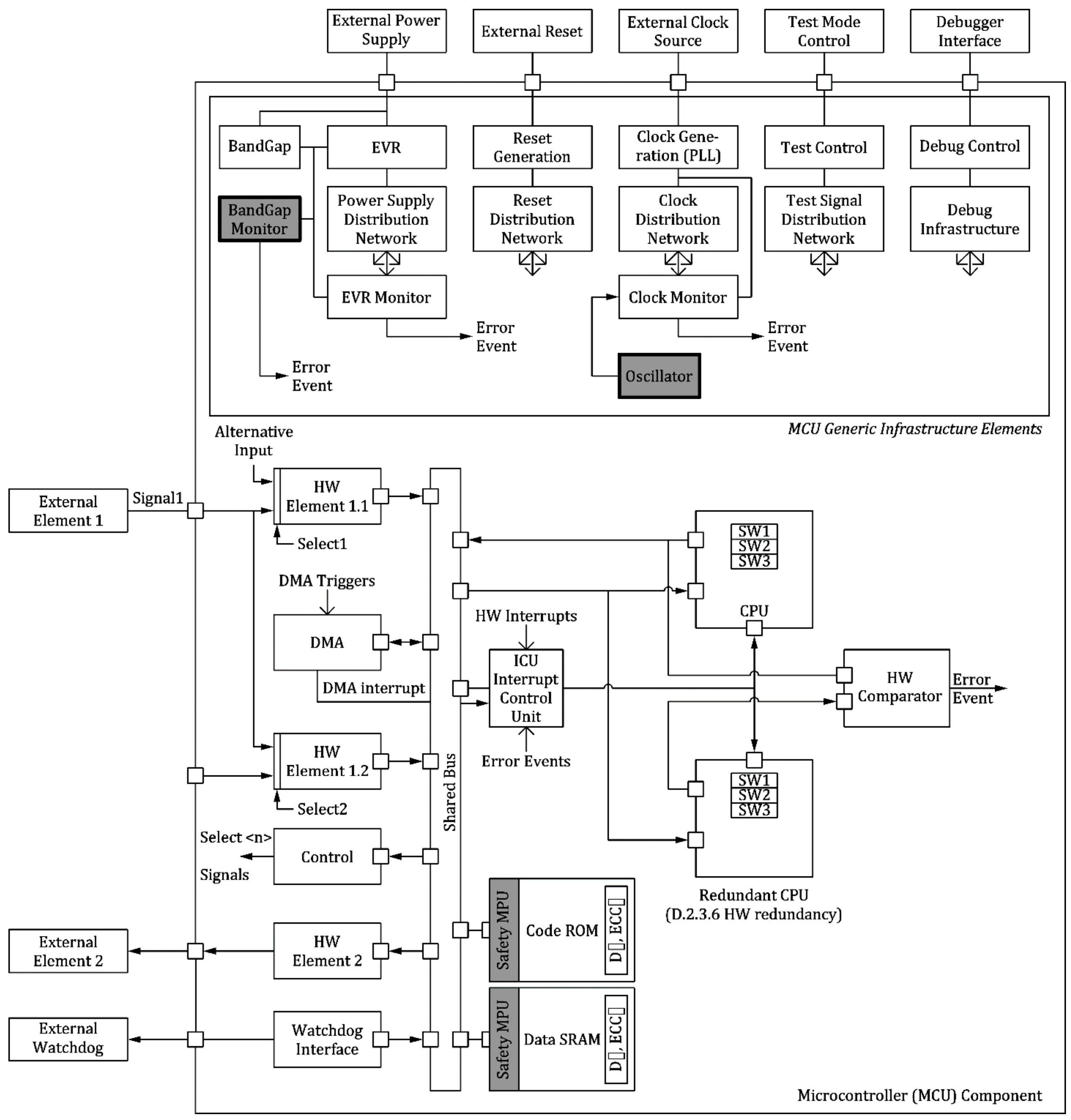


Figure B.3 — Enhanced microcontroller component

B.2 Analog example

B.2.1 Description

The analogue example is intended to provide guidance on the application of a DFA to analogue components, part or sub-parts. The detailed failure modes, relevant DFI, safety requirements and choice of considered safety and mitigation measures are typical examples, but they are not to be considered as exhaustive and can change depending on the details of the application, system architecture, circuit design and IC-technology.

The DFA of an analogue part is explained in the following clauses based on an assumed architecture of a switched output stage. The architecture of this output stage is sketched in Figure B.4 It uses high voltage N-DMOS switch transistors to activate the current path through a load which can for example be part of an actuator in a safety application. In order to avoid that faults of a switch transistor or its gate driver can activate the actuator inadvertently, the switches are redundantly placed in the high side and low side current paths to the load. The high side and low side drivers are supplied by a regulated voltage Vdd which is significantly lower than the external supply Vbat coming from the board net connected to the 12V battery of the vehicle. The output of the supply voltage regulator is already monitored by a voltage monitor which is used for non-safety purposed like the provision of a power on reset. The gate voltage that is needed to turn on the high side N-DMOS switch transistor is delivered by a charge pump in order to make the driver insensitive to EMC on the board net.

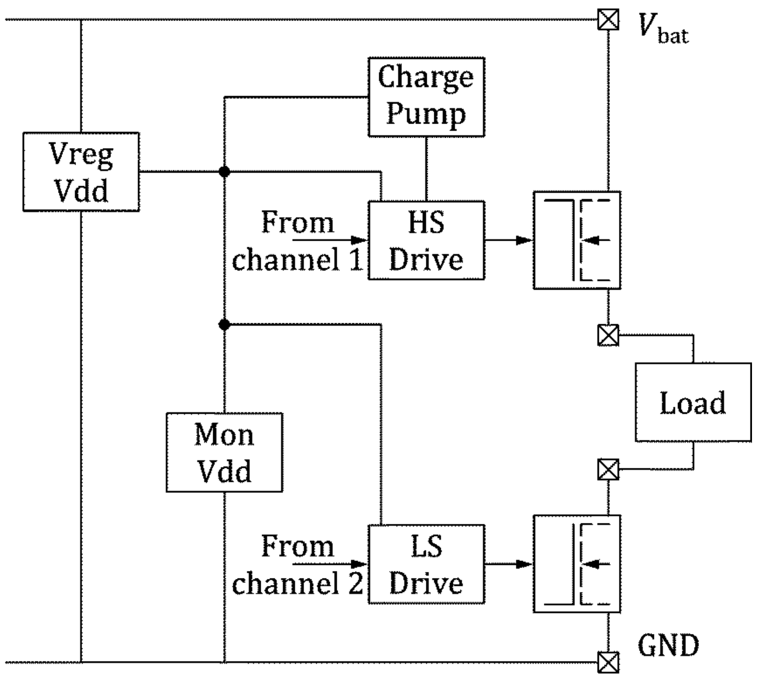


Figure B.4 — Analogue output driver example

In order to be able to identify dependent failures mechanisms, the following safety requirement is assumed:

— In the inactive state, the load connected between the high side switch transistor output and low side switch transistor output shall not be supplied with a current of more than 1mA for longer than 1ms.

NOTE The current of 1mA is assumed to be much lower than the current that is drawn by the load in the case that the switches are turned on (e.g. 1A).

B.2.2 Dependent failures by shared supply voltage regulator

The primary fault that leads to the exemplary dependent failures is illustrated in Figure B.5. The supply voltage regulator that supplies the internal driver circuitry for the control of the switch transistor gate voltages fails in a way that the pass device (pass device is the transistor that is in the supply current path) is permanently turned on. The fault mechanism could be a defect of the pass transistor itself or a fault of the control loop that causes instability like e.g. loss of a compensation capacitor. The consequence is a rise of the internal supply level Vdd to the external supply level V_{bat}.

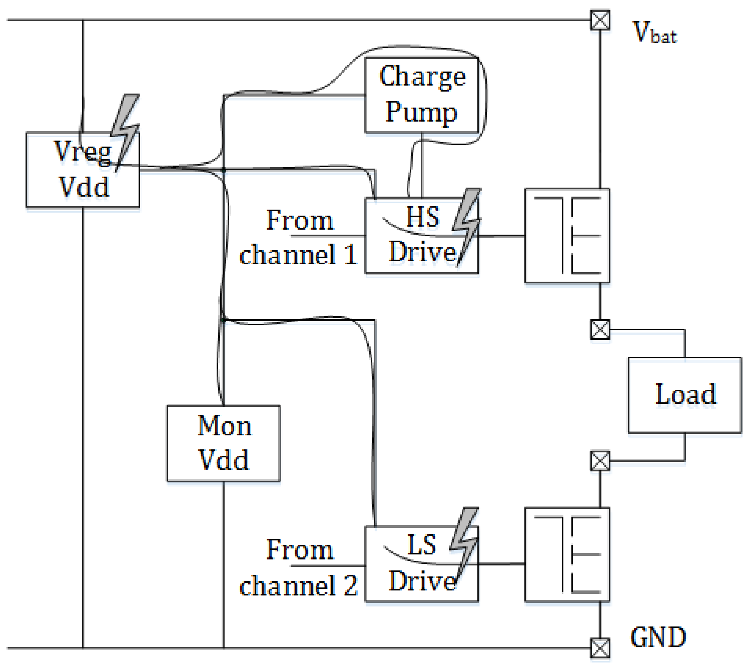


Figure B.5 — Dependent failures by shared supply voltage regulator

The fault is assumed to violate the safety requirement in the case of its appearance, since the complex driver circuit that we assume for this example cannot be realized in a way that allows operating it shorted to the external supply.

Thus severe damage of the driver must be assumed and the driver output cannot be assum ed to keep the gate voltages of the switch transistors on a level that keeps the switch transistors in a high impedance state. Thus the dependent failures that is caused by the “overvoltage” that is applied to the supply of the driver stages is assumed to have worst case consequences for the driver stages. Consequently it propagates to the top level failure in the fault tree shown in Figure B.6.

In quantitative safety analysis the SPFM of the “overvoltage” failure mode of the supply voltage regulator (not necessarily each failure mode of the supply voltage regulator e.g. under voltage) would be added directly to the SPFM for violating the defined safety goal, as shown by the grey under laid base event for overvoltage from the V_{dd} supply voltage regulator connected to the top level “OR” gate in the FTA.

NOTE There are other dependent failures that could appear as a consequence of overvoltage delivered by the supply voltage regulator. The first one is a fault induced in the charge pump, which is shown as a dotted line in the block diagram. In the worst case this fault can have the same effect than a damage of the high side driver due to overvoltage at its Vdd supply input and is therefore already included in the way the Vdd supply overvoltage fault was introduced in the FTA. Another dependent failures that could be induced by the overvoltage is the damage of the voltage monitor which can cause that the overvoltage stays undetected; this will be handled later on in the discussion of the measures to mitigate the dependent failures of the gate drivers.

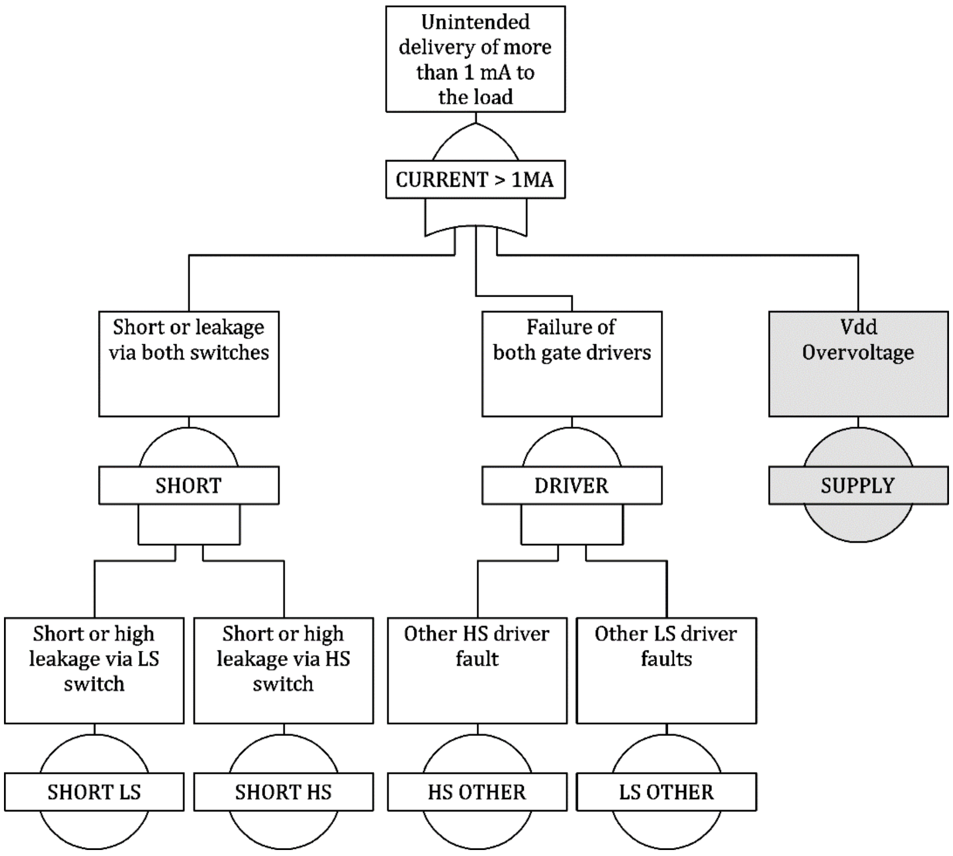

Unintended delivery of more than 1 mA to the load
CURRENT > 1MA
Short or leakage via both switches
Failure of both gate drivers
Vdd Overvoltage
SHORT
DRIVER
SUPPLY
Short or high leakage via LS switch
Short or high leakage via HS switch
Other HS driver fault
Other LS driver faults
SHORT LS
SHORT HS
HS OTHER
LS OTHER

Figure B.6 — FTA including shared supply

The following freedom from interference requirement could be derived in order to assure the achievement of the safety requirement for the case that the described fault in the supply voltage regulator appears.

— "A failure in the supply voltage regulator block shall not cause an activation of either the high side or the low side switch transistor in a way that the corresponding output could deliver a current of more than 1mA to the load for longer than 1ms."

In order to achieve the demanded freedom from interference, safety measures have to be defined in order to avoid a violation of the safety requirement in the case of a connection between the internal supply of the driver stages and the external supply voltage V_{bat}. Example of taken measures as shown in Figure B.7:

— Introduce sub-parts to pull down the switch transistor gate source voltages below their threshold voltages. The pull down blocks are activated by the supply monitoring block; and

— Limit of the current that can pass the connection between the driver output and the switch transistor gate to assure that the pull down is able to keep the gate source voltage sufficiently low for the case of a short to the supply at the gate driver output.

As a consequence of the introduction of the above mentioned safety mechanisms, the architecture of the system is changed and a rise of the internal supply to the level of the board net is no longer causing a violation of the safety requirement by the initial dependent failures as long as the pull down sub-parts are activated. If there is no other cascading effect which could impact the function of this safety mechanism the mitigation of the dependent failures would be sufficient. The adaptation of the fault tree according to the defined mitigation measures that result from the DFA is shown in Figure B.8.

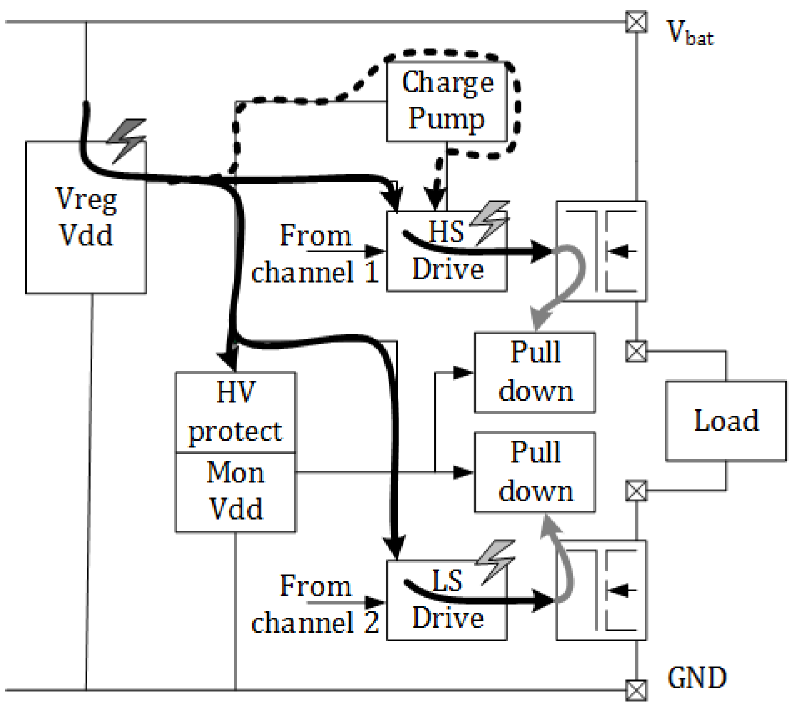


Figure B.7 — Shared supply fault mitigation

For the case that the change of the architecture introduces other additional dependent failures mechanisms that could impact the effectiveness of the new safety mechanism (a) and (b) that were introduced to mitigate the initial dependent fault, an additional freedom from interference requirement has to be derived. For this case the new freedom from interference requirement could be formulated as follows:

"A failure of the supply voltage regulator that shorts the internal supply V_{dd} to the external supply voltage V_{bat} shall not cause a failure in the voltage monitor or a failure of the pull down blocks, which disables the pull down current paths in a way that the threshold of the switch transistors can be exceeded longer than 1ms."

For the achievement of this new freedom from interference requirement additional safety measures are installed for the switch transistors. These pull down blocks are not affected by the initial fault (short of the internal supply V_{dd} to the external board net supply V_{bat}) in a way that prevents them from keeping the gates of the output switch transistor pulled down.

Example of taken measures:

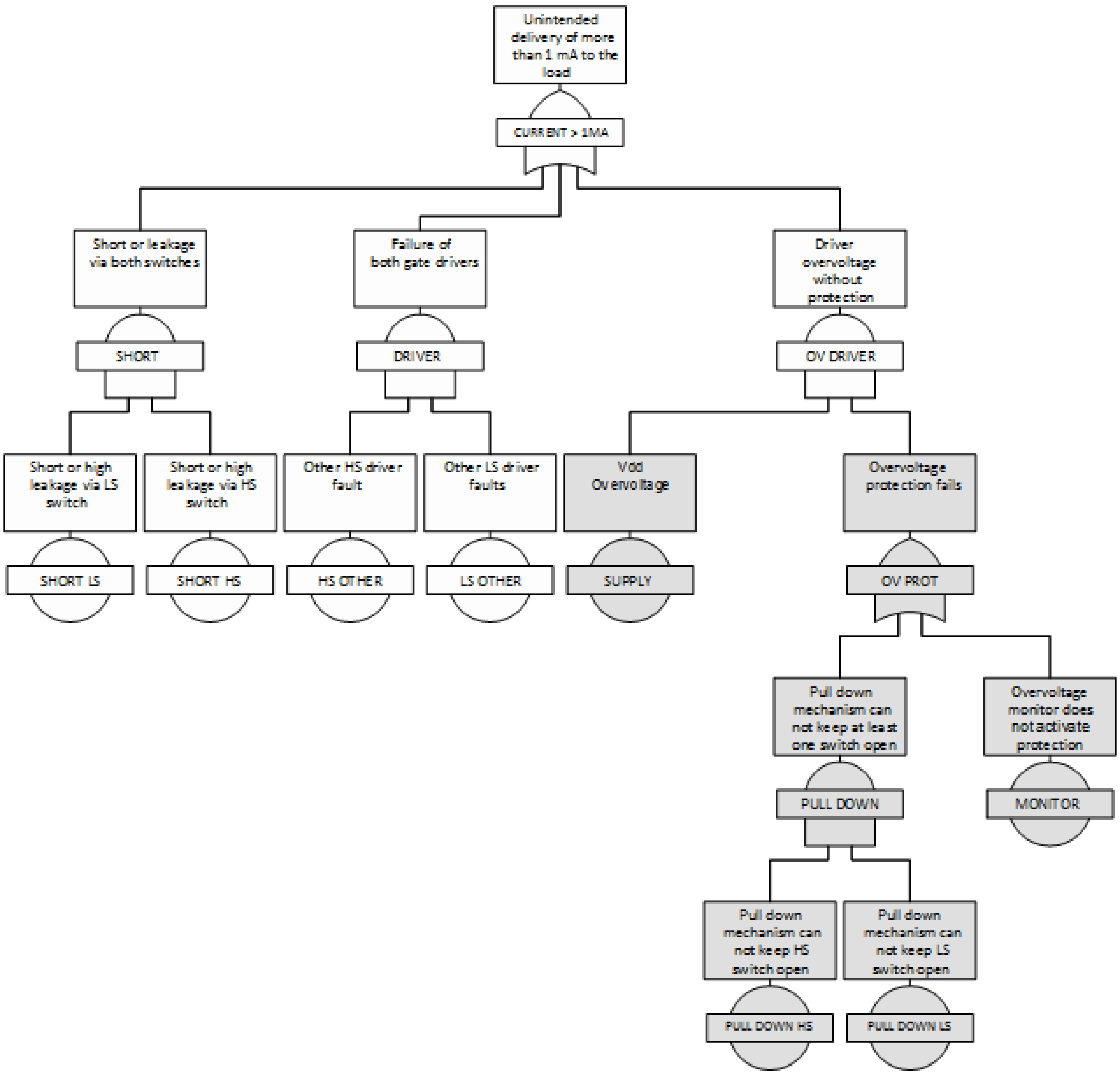
— Introduction of a high voltage protection block for the supply monitor; and

— Design of the gate pull down has to be dimensioned for operation with the external supply voltage.

For this example it is assumed that the IC technology allows to implement these measures in a way that provides sufficient safety margin. This assumption is justifiable in a qualitative evaluation, since the supply monitor and the pull down blocks are small and can be realized in a way (e.g. increased channel length, cascaded HV transistors, serial resistors) that allows increased safety margin compared to the supply voltage regulator (higher absolute maximum rating for supply voltage). Of course the safety requirements, fault mechanisms and suggested mitigation method are just exemplary and based on assumptions of the following boundary conditions:

- a circuit architecture;
- application requirements; and
- capabilities of an IC technology which will be used to fabricate the circuit.

The example is used here in order to explain how to perform a DFA of an analogue part and not as reference for the mitigation of dependent failures caused by overvoltage faults of the supply voltage regulator in real switched output stages. Other methods or variants to mitigate the same fault can be exploited instead and have to be selected based on the final knowledge of the real boundary conditions (e.g. technology options, external safety mechanisms). Finally the new sub-parts that have been introduced to mitigate the dependent failures that was caused by the supply overvoltage have to be included in the latent fault analysis. If required they have to be tested in repeated time intervals (e. g. at each system start-up) to avoid that they are not functional when the overvoltage fault case appears.



Unintended delivery of more than 1 mA to the load
CURRENT > 1MA
Short or leakage via both switches
SHORT
Failure of both gate drivers
DRIVER
Driver overvoltage without protection
OV DRIVER
Short or high leakage via LS switch
SHORT LS
Short or high leakage via HS switch
SHORT HS
Other HS driver fault
HS OTHER
Other LS driver faults
LS OTHER
Vdd Overvoltage
SUPPLY
Overvoltage protection fails
OV PROT
Pull down mechanism can not keep at least one switch open
PULL DOWN
Overvoltage monitor does not activate protection
MONITOR
Pull down mechanism can not keep HS switch open
PULL DOWN HS
Pull down mechanism can not keep LS switch open
PULL DOWN LS

Figure B.8 — FTA including shared supply fault mitigation

B.2.3 Dependent failures by coupling mechanism

The primary fault that leads to the second exemplary dependent failure is illustrated in Figure B.9. It is a random hardware fault that appears in the high side driver. It leads to a failur e of the high side path, which results in a conductance of the high side switch transistor. It further activates a coupling effect that can initiate a dependent failures in the low side path.

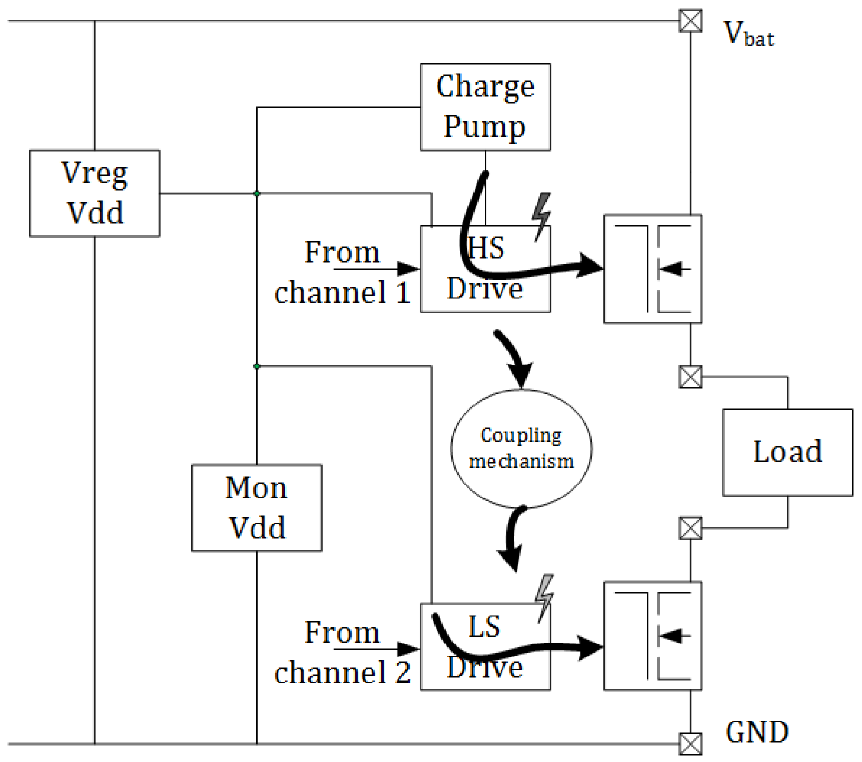


Figure B.9 — Dependent failures by coupling mechanism

An independence requirement could be stated as: “A failure of the high side path shall not induce a failure in the low side path that leads to an activation of the low side switch transistor in a way that it can deliver more than 1mA.” As a result of the evaluation of the DFI list we identified the following relevant initiators (see Table B.2) and their corresponding coupling mechanisms that require a definition of special mitigation measures.

NOTE This is an example and does of course not imply that these 3 DFI are the only relevant for gate drivers.

For each dependent failure listed in Table B.2, the fault tree in Figure B.10 is used. It shows that besides independent random faults in every channel, a coupling between the channels can lead to a fault in the second channel that is not directly affected by the initial fault.

In the case of temperature increases (reference number 1 in Table B.2) or break down of the supply (reference number 2 in Table B.2) the dependent failures can be avoided by implementation of a safety mechanism that detects the coupling effect and brings the system or element into a safe state. In the case of the substrate current injection (reference number 3 in Table B.2) mitigation could be achieved by technology and/or layout measures that break the coupling mechanism.

Table B.2 — Example of identified relevant coupling mechanisms

Reference number	DFI	Coupling mechanism
1	Local hot spot in one of the gate driver circuits (e.g. caused by a defect of a device inside the gate driver block that heats up due to	Heat propagation via the substrate causes an exceed of the maximum rating of the temperature range of the other gate driver.

Reference number	DFI	Coupling mechanism
	increased power consumption of the defective device).	
2	Short circuit in one of the gate drivers leading to a current consumption above the specification of the supply voltage regulator.	Break down of the supply of the other gate driver causes an undefined state (neither within the operating range nor in the range that leads to power on reset).
3	Injection of current into the substrate within one of the gate drivers e.g. caused by defects of substrate pn junctions or by activation of parasitic bipolar transistor of power devices.	Latch up induced including circuit elements of the other gate driver due to increasing voltage drop along the path of the substrate current to GND.

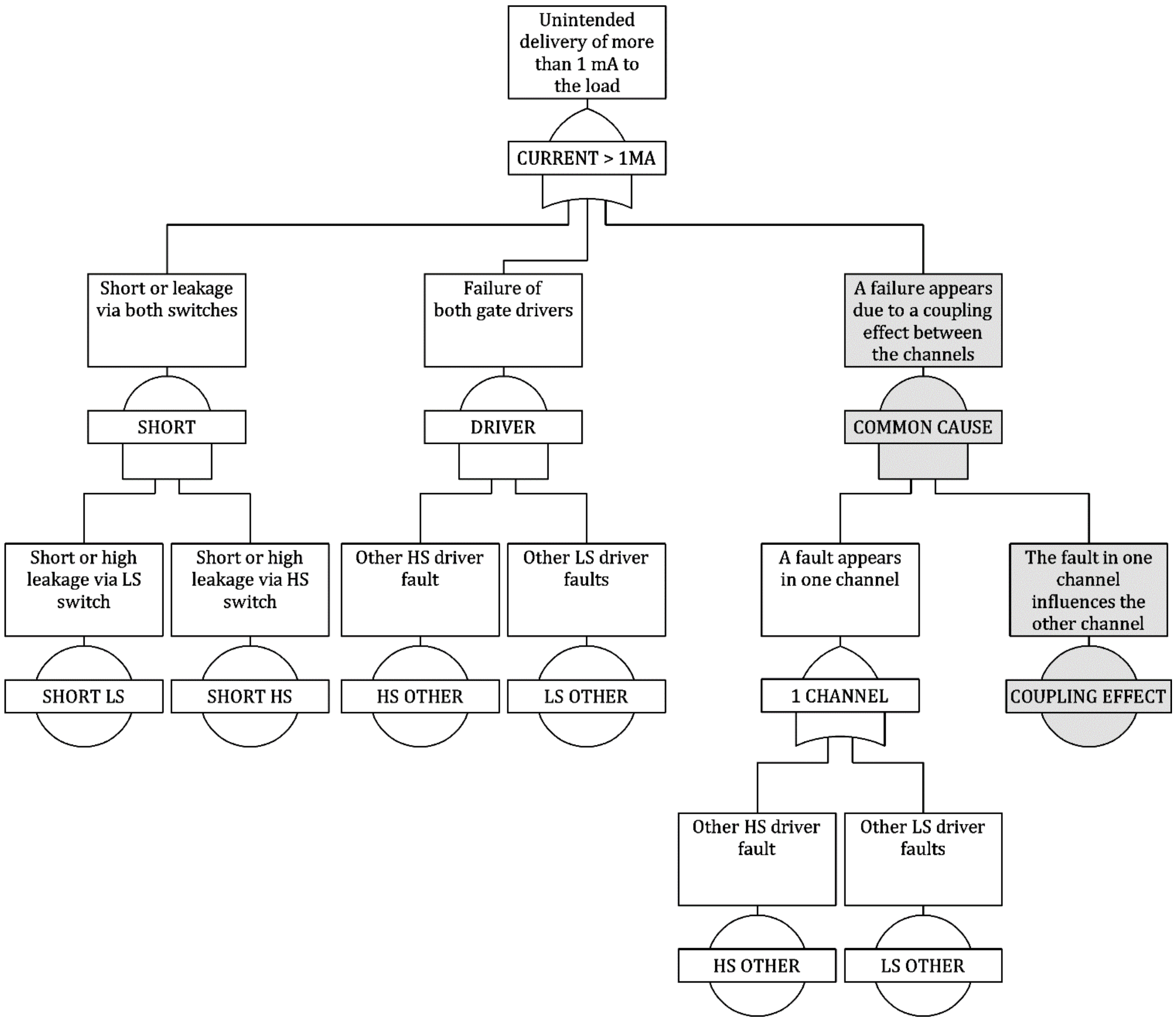

Unintended delivery of more than 1 mA to the load
CURRENT > 1MA
Short or leakage via both switches
SHORT
Failure of both gate drivers
DRIVER
A failure appears due to a coupling effect between the channels
COMMON CAUSE
Short or high leakage via LS switch
SHORT LS
Short or high leakage via HS switch
SHORT HS
Other HS driver fault
HS OTHER
Other LS driver faults
LS OTHER
A fault appears in one channel
1 CHANNEL
The fault in one channel influences the other channel
COUPLING EFFECT
Other HS driver fault
HS OTHER
Other LS driver faults
LS OTHER

Figure B.10 — Fault tree including coupling effect

In order to achieve a mitigation of the identified dependent failures we define additional safety mechanisms in Table B.3.

NOTE The mitigation of the dependent failures can require one or a combination of the mitigation measures, a final prove of the evidence of the chosen measures is made available with respect to the real design, layout, technology, package and application.

Table B.3 — Examples for the mitigation of coupling effects

Reference number	Dependent failures mitigation
1	Temperature measurement in the proximity of the gate drivers (the acceptable distance depends on the thermal resistance of the heat sink path and can be found by thermal simulation, sensor elements can be resistors or bipolar transistors) and shut down of the gate driver supply in the case of over temperature. Current limitation in the supply voltage regulator to limit the power that is available to heat up the chip and brings it into a defined under voltage reset state. A thermal segregation (e.g. sufficient distance in combination with a backside heat sink via an exposed die pad) of the independent paths (high side & low side path, each consisting of a switch transistor and its associated gate driver) that is sufficient to prevent the overheating of the fault free path (the one that is not affected by the initial fault). Dimension of the required segregations can be evaluated e.g. based on thermal simulations).
2	Current measurement of the block supplies and shut down of the gate driver supply in the case of overcurrent. Voltage monitor with under voltage reset that avoids undefined states by setting the reset threshold inside the safe operation range of the circuit. Passive pull down of the gates e.g. with resistors to keep switch transistors in off state if the supply is low.
3	Physical separation (e.g. spacing, guard rings, separate wells, trenches, buried layer, sinkers – depends on the IC technology) with the target to interrupt the latch up mechanism between the parts that have to be independent.

Annex C (informative) Examples of quantitative analysis for digital component

C.1 Description

The following is an example of a quantitative analysis using the method described in 5.1.7.

NOTE 1 Numbers used in this example (e.g. failure rates, amount of safe faults and failure mode coverage) are examples. They can vary from architecture to architecture.

NOTE 2 The following examples divide a portion of the digital component into the sub-parts level. As discussed in 5.1.7, the necessary level of detail can depend on the stage of the analysis and on the safety mechanisms used.

NOTE 3 The following examples use the quantitative approach to compute a dedicated target "single-point fault metric" value for transient faults. As discussed in 5.1.7.2, transient faults can be also addressed by qualitative rationale. The rationale includes the reason why the qualitative approach is adequate.

The example considers a small portion of a digital component, i.e. only two parts:

— a small CPU, divided in five sub-parts: register bank, ALU, load-store unit, control logic and debug. Each sub-part is further divided in several sub-parts; and

— a 16KB of RAM divided in three sub-parts: cell array, address decoder and logic for end-of-line test, and management of spare rows (redundancies) of RAM.

NOTE 4 The FIT numbers shown in the example do not include peripherals or other features such as package, handling or overstress. They are given just as an example of a possible method for FIT rate computation. For this reason, those values are not comparable with FIT rates of a complete packaged digital component as shown for example in SN 29500.

NOTE 5 The aim of the following example is to avoid a requirement that each smallest digital component sub-part be addressed in the system-level analysis. At system-level analysis, component or part level detail can be sufficient. The aim of this example is to provide evidence that for a digital component at stand-alone level, a deeper analysis (e.g. at sub-part level) can be needed in order to compute with the required accuracy the failure rates and failure mode coverage of parts and sub-parts, to be used afterwards by system engineers. In other words, without an accurate and detailed digital component stand-alone level analysis, it can be very difficult to have good data for system-level analysis.

The following four safety mechanisms are considered:

— a hardware safety mechanism (SM1) performing a logical monitoring of program sequence of CPU. This safety mechanism is able to detect with certain coverage the faults in the control logic that could cause the software to run out of sequence. However, this safety mechanism is poor at detecting faults (such as wrong arithmetic operations) leading to wrong data;

  NOTE 6 In this example, it is assumed that each detected permanent single bit faults affecting the CPU is signalled to the system (e.g. by activating an output signal of the digital component). A requirement is set at system or element level to make proper use of this signal (e.g. to enter a safe state and inform the driver). For suspect transient faults, the CPU can try to remove these faults by a reset. If the fault persists, it means it is permanent, and therefore it can be signalled to the system as previously described. If the fault disappears (i.e. it was really transient), the CPU can continue.

— a software-based safety mechanism addressing random hardware failures (SM2) executed at key-on to verify the absence of latent faults in the logical monitoring of program sequence of CPU (SM1);

— an error detection-correction logic EDC-ECC with the capability to correct all single bit faults (single error correction, SEC) and detect all double bit faults (double error detection, DED) for the RAM (SM3); and

NOTE 7 In this example, it is assumed that each detected permanent single bit fault – even if corrected by the EDC-ECC — is signalled to the software (e.g. by an interrupt), and the software reacts accordingly. A requirement is set at system or element level to make proper use of this event (e.g. to go in a safe state and inform the driver). For suspected transient faults corrected by EDC-ECC, the CPU can try to remove these faults by writing back in the memory the correct value. If the fault persists, it means it is permanent and therefore is signalled to the system as previously described. If the fault disappears (i.e. it was transient), the CPU can continue. To distinguish intermittent and transient faults, counting numbers of corrections could be a possible method.

— a software-based safety mechanism addressing random hardware failures (SM4) executed at key-on to verify the absence of latent faults in the EDC-ECC (SM3).

Table C.1 is divided in three separated calculations for better visibility.

Table C.1 gives the view of failure modes at sub-parts level. Table C.2 shows how the low-level failure modes can be identified and therefore how the overall failure distribution can be computed, following the approach described in 4.4.

EXAMPLE 1 The table shows that the failure rate of a permanent fault in the flip-flop X1 and its related fan-in is 0,01 FIT. Summing each of those low-level failure modes, it is possible to compute the failure rate of a permanent fault of the ALU logic as a whole (0,0348 FIT). With the same procedure, by summing each of the failure rates related to the sub-part, it is possible to compute the FIT rate for a permanent fault in the ALU.

NOTE 8 Going up in the failure modes abstraction tree (i.e. from the low-level failure modes to the higher ones), failure rates of different sub-parts failure modes could be combined to compute the failure rate for the higher-level failure mode, especially if those higher-level failure modes are defined in a more generic way.

EXAMPLE 2 If a higher-level failure mode (e.g. at part-level) is defined as "wrong instruction processed by CPU", the failure rate of this failure mode can be a combination of the failure rates of many failure modes at sub-parts level, such as a permanent fault in the pipeline, a permanent fault in the register bank, etc. Therefore, if the low-level failure rates are available, the higher-level failure rate can be computed with a bottom-up approach (assumes independent faults).

NOTE 9 Columns of tables can be correlated to the flow diagram for fault classification and fault class contribution calculation described in 7.1.7:

— failure rate (FIT) is equal to λ;

— amount of safe faults is equal to F_{safe};

— failure mode coverage with respect to violation of safety goal is equal to $K_{\mathrm{FMC,RF}}$;

— residual or single-point fault failure rate is equal to λ_{SPF} or λ_{RF} depending on whether the failure is single-point or residual. In the example, no single-point faults are considered, so this failure rate is always equal to λ_{RF};

— failure mode coverage with respect to latent failures is equal to $K_{\mathrm{FMC,MPF}}$; and

— latent multiple-point fault failure rate is equal to λ_{MPF}.

NOTE 10 The amount of safe faults is the fraction of the failure mode that has neither the potential to violate the safety goal in absence of safety mechanisms nor in combination with independent failures of another sub-part.

NOTE 11 The failure mode coverage is computed with a detailed analysis of the capability of SM1 to cover each sub-part. In this example, R0 and R1 are registers chosen by the compiler to pass function parameters, so they have a slightly higher probability to cause a program sequence error detectable by SM1. The aim of this example is to provide evidence that by means of a detailed analysis, it is possible to identify differences in the coverage of the sub-parts.

NOTE 12 The failure mode coverage of the EDC-ECC (SM3) is computed, for example, with a detailed analysis combining the high probability of EDC-ECC of detecting single and double bit errors with the lower probability of detection (it could be less than 90 %) of multiple-bit errors. This is shown in Table C.2.

NOTE 13 Certain sub-parts can be covered by several safety mechanisms: in such cases, the resulting failure mode coverage combines the coverage for each failure mode determined by means of a detailed analysis.

NOTE 14 The example shows that without proper coverage of the EDC-ECC with respect to multiple bit errors and without the coverage of the RAM address decoder, it can be difficult to achieve a high single-point fault metric.

NOTE 15 The example shows that some safety mechanisms can cause a direct violation of the safety goal, and therefore they are considered in the computation of residual faults. In this example, a fault in the EDC-ECC (SM3) can corrupt the mission data without a corresponding fault in the memory.

NOTE 16 The example shows that, in a digital component, sub-parts could coexist which potentially are not safety-related but for which it is impossible to establish a clear separation or distinction from the safety-related sub-parts (the debug inner logic). Instead, other parts (the debug interface) could be easily isolated and disabled in a way that they can be considered not safety-related without risks.

NOTE 17 At this level of detail, it can be possible to find out that certain low-level failure modes (e.g. a single-event upset and single-event transient fault in flip-flop X2 and its fan-in) are safe (e.g. because that bit is seldom used by the ALU architecture).

NOTE 18 The failure rate of the memory for a single permanent fault causing n>2 bit errors is computed, for example, considering memory layout information, structure of the address decoder, etc.

NOTE 19 The EDC-ECC (SM3) coverage for >2 bit errors is computed with a detailed analysis considering the number of bits in each coded word (in this case 32) and the number of code bits (in this case 7). Depending on those parameters, coverage can be much higher.

4025 **Table C.1 — Example of quantitative analysis (at sub-parts level)**

					Permanent failures								Transient failures				
Part	Sub-part		Safety Related Component ? No Safety Related Component ?	Failure modes	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) preventing latent faults	Failure mode coverage wrt. Latent failures	Latent Multiple Point Fault failure rate / FIT	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT
CPU	Register bank	Register R0	SR	permanent fault	0.0029	0%	SM1	40%	0.00174	SM1	100%	0.00000					
				transient fault									0.032005	0%	SM1	40%	0.01920
		Register R1	SR	permanent fault	0.0029	0%	SM1	40%	0.00174	SM1	100%	0.00000					
				transient fault									0.032005	0%	SM1	40%	0.01920
		Register R2	SR	permanent fault	0.0029	0%	SM1	20%	0.00232	SM1	100%	0.00000					
				transient fault									0.032005	0%	SM1	10%	0.02880
		Register R3	SR	permanent fault	0.0029	0%	SM1	20%	0.00232	SM1	100%	0.00000					
				transient fault									0.032005	0%	SM1	10%	0.02880
	ALU	ALU	SR	permanent fault	0.0348	0%	SM1	20%	0.02784	SM1	100%	0.00000					
				transient fault									0.00038	20%	SM1	10%	0.00027
		MUL	SR	permanent fault	0.0290	0%	SM1	20%	0.02320	SM1	100%	0.00000					
				transient fault									0.00037	70%	SM1	10%	0.00010
		DIV	SR	permanent fault	0.0232	0%	SM1	20%	0.01856	SM1	100%	0.00000					
				transient fault									0.00036	70%	SM1	10%	0.00010
	Control logic	Pipeline	SR	permanent fault	0.0174	0%	SM1	90%	0.00174	SM1	100%	0.00000					
				transient fault									0.00103	20%	SM1	90%	0.00008
		Sequencer	SR	permanent fault	0.0406	0%	SM1	90%	0.00406	SM1	100%	0.00000					
				transient fault									0.00307	50%	SM1	90%	0.00015
		Stack control	SR	permanent fault	0.0029	0%	SM1	70%	0.00087	SM1	100%	0.00000					
				transient fault									0.000325	50%	SM1	40%	0.00010
	Load Store Unit	Address generation	SR	permanent fault	0.0174	0%	SM1	60%	0.00696	SM1	100%	0.00000					
				transient fault									0.00103	10%	SM1	60%	0.00037
		Load Unit	SR	permanent fault	0.0145	0%	SM1	50%	0.00725	SM1	100%	0.00000					
				transient fault									0.000345	10%	SM1	50%	0.00016
		Store Unit	SR	permanent fault	0.0145	0%	SM1	50%	0.00725	SM1	100%	0.00000					
				transient fault									0.000345	10%	SM1	50%	0.00016
	Debug	Debug Inner Logic	SR	permanent fault	0.0058	20%	none	0%	0.00464	none							
				transient fault									0.00017	20%	none	0%	0.00014
		Debug Interface	NSR	permanent fault	0.0783												
				transient fault									0.001635				
				Σ					0.11049			0.00000					0.09764

Total failure rate	0.29000	Total failure rate	0.13708
Total Safety Related	0.21170	Total Safety Related	0.13545
Total Not Safety Related	0.07830	Total Not Safety Related	0.00164

Single Point Faults Metric 47.8% | Single Point Faults Metric 27.91%

Latent Faults Metric 100.0%

Part	Sub-part			Failure modes	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) preventing latent faults	Failure mode coverage wrt. Latent failures	Latent Multiple Point Fault failure rate / FIT	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT
Volatile Memory	RAM (16KB)	RAM data bits	SR	permanent fault	1.5000	0%	SM3	96.9%	0.04688	SM3	100%	0.00000					
				transient fault									131.072	0%	SM3	99.69%	0.40894
		Address Decoder	SR	permanent fault	0.0087	0%	none	0%	0.00870								
				transient fault									0.000335	0%	none	0%	0.00034
		Test/redundancy	SR	permanent fault	0.0058	50%	none	0%	0.00290								
				transient fault									0.00033	90%	none	0%	0.00003
				Σ					0.05848			0.00000					0.40931

Total failure rate	1.51450	Total failure rate	131.07
Total Safety Related	1.51450	Total Safety Related	131.07
Total Not Safety Related	0.00000	Total Not Safety Related	0.00

Single Point Faults Metric 96.1% | Single Point Faults Metric 99.69%

Latent Faults Metric 100.0%

Part	Sub-part			Failure modes	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) preventing latent faults	Failure mode coverage wrt. Latent failures	Latent Multiple Point Fault failure rate / FIT	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT
Safety Mech.	SM1	Detection Logic	SR	permanent fault	0.0029	0%				SM2	90%	0.00029					
				transient fault									0.000105				
		Alarm Generation	SR	permanent fault	0.0029	50%				SM2	90%	0.00015					
				transient fault									0.000055				
	SM3	EDC Coder	SR	permanent fault	0.0029	0%	SM3	90%	0.00029	SM4	90%	0.00026					
				transient fault									0.000325	0%	none	0%	0.00033
		EDC Decoder	SR	permanent fault	0.0029	0%	SM3	90%	0.00029	SM4	90%	0.00026					
				transient fault									0.000325	0%	none	0%	0.00033
		Alarm Generation	SR	permanent fault	0.0029	50%				SM4	90%	0.00015					
				transient fault													
		RAM EDC bits	SR	permanent fault	0.328125	0%	SM3	96.9%	0.01025	SM4	90%	0.03179					
				transient fault									28.6720	0%	SM3	99.69%	0.08946
				Σ					0.00058			0.03289					0.09011

Total failure rate	0.34263	Total failure rate	28.67281
Total Safety Related	0.34263	Total Safety Related	28.67281
Total Not Safety Related	0.00000	Total Not Safety Related	0.00000

Single Point Faults Metric 99.8% | Single Point Faults Metric 99.69%

4026 Latent Faults Metric 90.4%

4027 **Table C.2 — Example of quantitative analysis (at low-level failures level)**

						Permanent failures										Transient failures				
Part	Sub-part	Elementary sub-parts	Safety Related Component ? No Safety Related Component ?	Failure modes		Failure rate (FIT)	Amount of safe faults	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT		Safety mechanism(s) preventing latent faults	Failure mode coverage wrt. Latent failures	Latent Multiple Point Fault failure rate / FIT		Failure rate (FIT)	Amount of safe faults	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT
CPU	ALU	ALU	SR	permanent fault in the flip-flop X1 and its fan-in		0,0100	0%	SM1	20%	0,00800		SM1	100%	0,00000						
				SEU and SET in the flip-flop X1 and its fan-in												0,0001	0%	SM1	10%	0,00009
				permanent fault in the flip-flop X2 and its fan-in		0,0150	0%	SM1	20%	0,01200		SM1	100%	0,00000						
				SEU and SET in the flip-flop X2 and its fan-in												0,0001	70%	none	0%	0,00003
				etc....		...	..	..	..	..		..	..	..		..	..	..	..	..
					Σ	0,0348				0,02784				0,00000		0,00038				0,00027
Volatile Memory	RAM (16KB)	RAM data bits	SR	permanent fault causing ≤2 bit errors in same coded word		1,3500	0%	SM3	100,0%	0,00000		SM3	100%	0,00000						
				≤2 SEUs in same coded word												129,76128	0%	SM3	100,00%	0,00000
				permanent fault causing >2 bit errors in same coded word		0,1500	0%	SM3	68,8%	0,04688		SM3	100%	0,00000						
				>2 SEUs in same coded word												1,31072	0%	SM3	68,8%	0,40894
					Σ	1,5000				0,04688				0,00000		131,0720				0,40894

4028

Annex D(informative) Examples of quantitative analysis for analogue component

D.1 Description

The following is an example of a quantitative analysis using the method described in 5.1.7 in order to calculate the single-point fault metric and the latent-fault metric for a given safety requirement allocated to the mixed signal hardware element depicted in Figure D.1.

The example consists of a mixed signal hardware element composed of:

— a low drop voltage regulator (low drop voltage regulator in) providing an output voltage within a prescribed range;

— a voltage monitor capable of detecting overvoltage (VA > OV_th) and under-voltage (VA < UV_th) on the LDO output by monitoring the regulated voltage VA and comparing it with two predefined thresholds; the predefined thresholds are generated from a reference voltage provided by an independent bandgap in order to ensure independence with respect to the voltage regulator;

— an analogue BIST controlled through the digital system (the digital controller is not depicted in the block diagram in Figure D.1); and

— an ADC channel.

The ASIL B safety requirement is: "The regulated voltage output does not go out of regulation, i.e. the regulated voltage VA is not outside the UV_th-OV_th range for more than 1ms."

The component can be considered in a safe state when an out of regulation condition is detected and signalled to an external element of the system/item. The external system is responsible for fault reaction including transitioning the system or element to a safe state.

As shown in Figure D.3 the voltage monitor is composed of two voltage comparators, a passive network and a bandgap; the low drop regulator includes a bandgap, a current limiter, the bias generator and the regulator core as shown in Figure D.2.

The ADC is included in the mixed signal hardware element but it is not used for any function related to the safety requirement and so its potential failure cannot contribute to the violation of such requirement; therefore the ADC is assumed not safety-related.

The following safety mechanisms are considered:

— The voltage monitor detecting overvoltage (safety mechanism SM2) and under-voltage (safety mechanism SM1) failures with a diagnostic coverage of 99,9 %. The safety mechanism is described in 5.2.4.2.

— the analogue BIST detecting failures affecting the voltage monitor with a diagnostic coverage of 60 % (safety mechanism SM6). The safety mechanism is described in 5.2.4.10.

The coverage levels claimed by the safety mechanisms are reported in Table D.1. They are assumed to be proven with simulations, testing to characterize and confirm the behaviour of the silicon and the related evidences are documented in the product safety case. It is out of the scope of this example to provide those evidences.

Each safety mechanism signals the detection of a fault to an element of the system/item which is then responsible to transition the system or element to a safe state.

Under this assumption, the failure mode coverage with respect to latent failures related to the low drop regulator is claimed to be 100 % based on the example in ISO 26262-5:2018, Annex E.

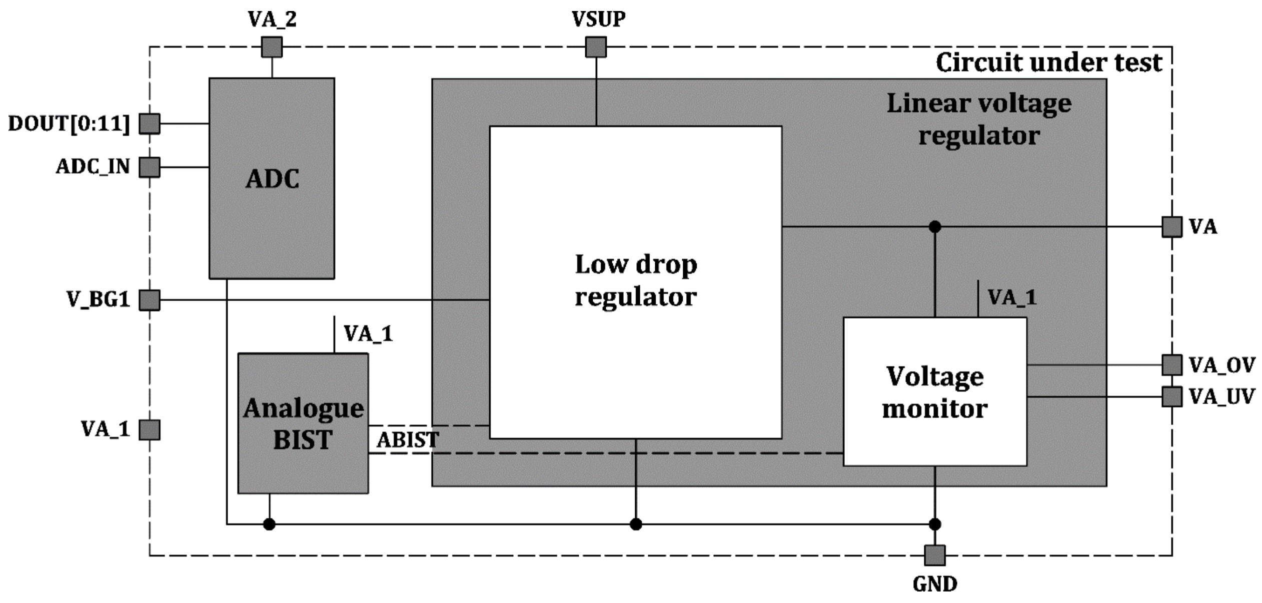


Figure D.1 — Example of analogue and mixed signal hardware element (circuit under analysis)

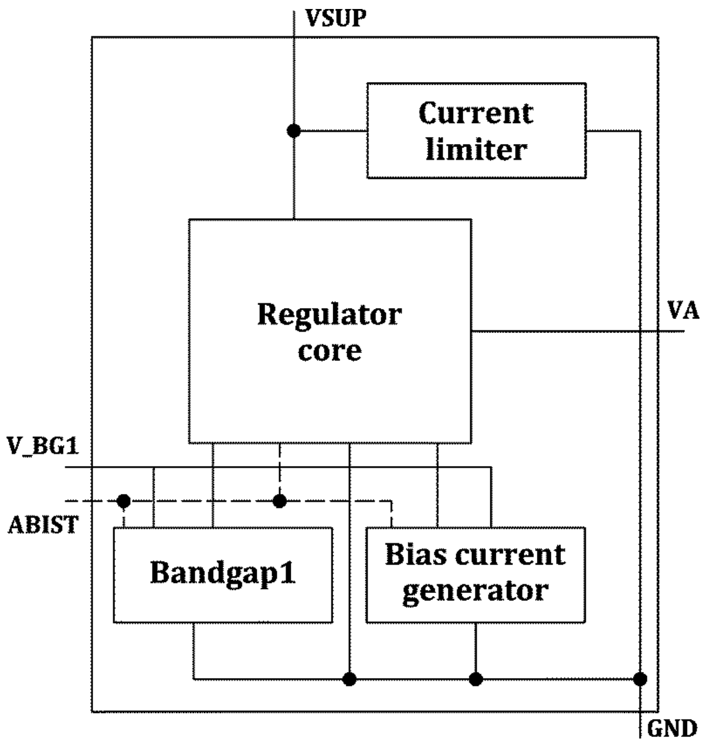


Figure D.2 — Detailed block diagram of the low drop regulator part

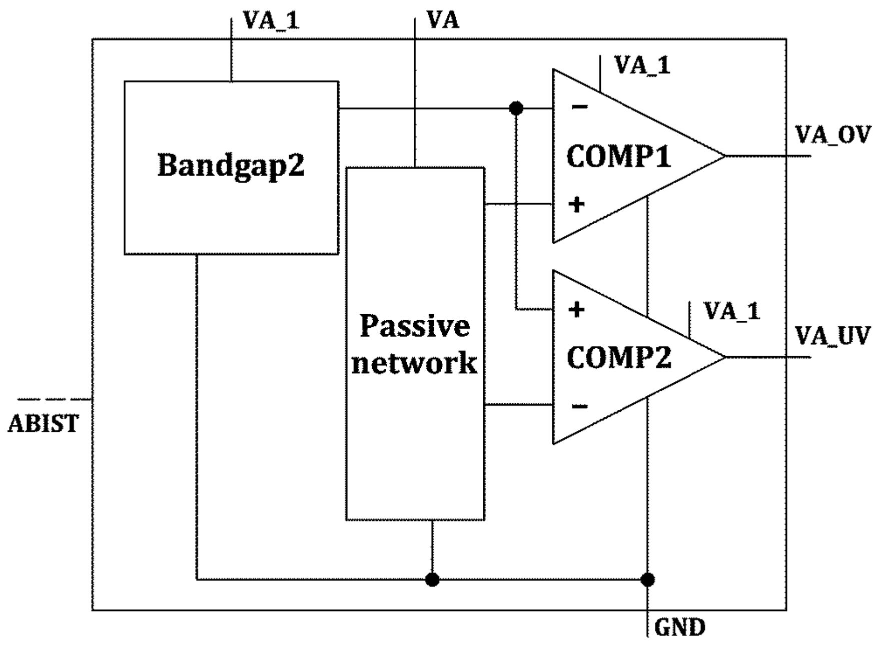


Figure D.3 — Detailed block diagram of the voltage monitor part

Table D.1 — Safety mechanisms considered in the example and related coverage for hardware element

ID	Safety mechanism	Claimed failure mode coverage
SM1	Under-voltage (UV) Monitor	99,9 %
SM2	Over-voltage (OV) Monitor	99,9 %
SM6	Analogue BIST diagnostics	60 %

NOTE 1 The example shows that parts which could be easily isolated and disabled in a way that they can be considered not safety-related without risk, can coexist with parts that are safety-related.

NOTE 2 The effectiveness of safety mechanisms could be affected by dependent failures. Adequate measures are considered as described in 5.2.3.6.

Based on the guidelines provided in 5.1.7.1, the failure rates and the metrics can be computed in the following way for analogue and mixed signal hardware elements:

— First, the hardware element is divided into parts or sub-parts;

NOTE 3 The validity of assumptions on the independence of identified parts is established during the dependent failures analysis.

NOTE 4 The necessary level of detail (e.g. if analysis at part level or sub-part level) can depend on the stage of the analysis and on the safety mechanisms.

— Second, the failure rates of each part or sub-part can be computed using one of the methods described in 5.2.3.4;

NOTE 5 In this example the failure rate distribution is assumed to be proportional to the area both for permanent and transient faults using the values reported in D.1.1.

— For each part/sub-part the relevant failure modes are listed and a failure mode distribution is assigned to each of them;

The failure mode distribution in the examples of Table D.2 and Table D.3 is considered equally distributed over the failure modes belonging to each part/sub-part. This assumption is to be understood as reference only, valid for the specific examples.

— The evaluation is completed by classifying the faults into safe faults, residual faults, detected dual-point faults and latent dual-point faults; and

— Finally, the failure mode coverage with respect to residual and latent faults of that part or sub-part is determined.

NOTE 6 Numbers used in this example (e.g. failure rates, amount of safe faults and failure mode coverage) can vary from architecture to architecture.

The example of quantitative analysis, limited to permanent faults, is reported in Table D.2 and Table D.3 using the same format of Table C.1. The quantitative analysis gives the view of failure modes at sub-part level.

NOTE 7 In this example a separate analysis with respect to transient faults is not reported but it can be added when relevant.

Depending on the system functions and safety requirements, different operating phases can be relevant and so additional failure modes can be considered.

EXAMPLE For systems that need to comply with start-stop requirements, the regulator start phase can be safety-related and the failure mode "Incorrect start-up time (i.e. outside the expected range) - Voltage ramp too fast" can be added.

Table D.2 — Example of quantitative analysis – mission parts

Part	Sub-part	Safety-related Component or No Safety-related Component	Failure Mode	Potential Effect of Failure Mode in Absence of Safety Mechanism (SM) on IC level[a]	Fault Model[b]	Failure distribution	Failure rate (FIT)	Amount of Safe Faults	Safety mechanism(s) preventing the violation of the safety requirement	Failure mode coverage with respect to violation of safety requirement	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) to prevent latent faults	Failure mode coverage with respect to Latent failures	Latent Multiple Point Fault failure rate / FIT
Linear Voltage Regulator	Low Drop Regulator	SR	Output voltage higher than a predefined high threshold of the prescribed range (i.e. Over voltage – OV)	Regulated voltage higher than VA_OV	P	14 %	2,16E-03	0 %	SM2	99,9 %	2,16E-06	SM2	100 %	0,0E+00
		SR	Output voltage lower than a predefined low threshold of the prescribed range (i.e. Under voltage – UV)	Regulated voltage lower than VA_UV	P	14 %	2,16E-03	0 %	SM1	99,9 %	2,16E-06	SM1	100 %	0,0E+00
		SR	Output voltage affected by spikes	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	14 %	2,16E-03	0 %	SM1 SM2	99,9 %	2,16E-06	SM1 SM2	100 %	0,0E+00

Table D.2 *(continued)*

Part	Sub-part	Safety-related Component or No Safety-related Component	Failure Mode	Potential Effect of Failure Mode in Absence of Safety Mechanism (SM) on IC level[a]	Fault Model[b]	Failure distribution	Failure rate (FIT)	Amount of Safe Faults	Safety mechanism(s) preventing the violation of the safety requirement	Failure mode coverage with respect to violation of safety requirement	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) to prevent latent faults	Failure mode coverage with respect to Latent failures	Latent Multiple Point Fault failure rate / FIT
		SR	Output voltage oscillation within the prescribed range	No effect- Regulated voltage within the expected range but with low accuracy	P	14 %	2,16E-03	100 %			0,0E+00			0,0E+00
		SR	Output voltage fast oscillation outside the prescribed range but with average value within the prescribed range	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	14 %	2,16E-03	0 %	SM1 SM2	99,9 %	2,16E-06	SM1 SM2	100 %	0,0E+00
		SR	Output voltage drift within the prescribed range	No effect- Regulated voltage within the expected range but with low accuracy	P	14 %	2,16E-03	100 %			0,0E+00			0,0E+00
		SR	Incorrect start-up time (i.e. outside the expected range) - Voltage ramp too fast	no effect - assuming during the voltage regulator start up the item is in safe state	P	0 %	0,00E+00	100 %			0,00E+00			0,0E+00
		SR	Incorrect start-up time (i.e. outside the expected range) - Voltage ramp too slow	no effect - assuming during the voltage regulator start up the item is in safe state	P	0 %	0,00E+00	100 %			0,00E+00			0,0E+00
		SR	Quiescent current (i.e. current drawn by the regulator in order to control its internal circuitry for proper operation) exceeding the maximum value	Regulated voltage potentially with low accuracy or out of regulation depending on the actual quiescent current	P	14 %	2,16E-03	50 %			1,08E-03	SM1 SM2	100 %	0,0E+00
ADC	ADC	NSR			P	100 %	7,00E-03				0,0E+00			0,0E+00
						Σ					0,00109			0,0E+00
				Total failure rate			0,0221							
				Total Safety-related			0,0151							
				Total Not Safety-related			0,0070							
									Single-Point Fault Metric		92,8 %			
													Latent-Fault Metric	100 %

[a] Depending on complexity it can be beneficial to have a dedicated entry in the FMEA giving more details about the potential root causes and the end effect of each failure mode.

[b] Fault model can be permanent fault (P) or transient fault (T); the example is limited to permanent faults.

Table D.3 — Example of quantitative analysis – safety mechanisms

Part	Sub-part	Safety-related Component or No Safety-related Component	Failure Mode	Potential Effect of Failure Mode in Absence of Safety Mechanism (SM) on IC level[a]	Fault Model[b]	Failure distribution	Failure rate (FIT)	Amount of Safe Faults	Safety mechanism(s) preventing the violation of the safety requirement	Failure mode coverage with respect to violation of safety requirement	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) to prevent latent faults	Failure mode coverage with respect to Latent failures	Latent Multiple Point Fault failure rate / FIT
Linear Voltage Regulator	Voltage Monitor (SM1, SM2)	SR	UV Monitor (SM1) falsely triggering UV event	Nuisance shutdown at nominal regulator loads.	P	25 %	1,45E-03	100 %						0,0E+00
		SR	UV Monitor (SM1) not triggering valid UV event	Regulated voltage lower than VA_UV	P	25 %	1,45E-03	0 %				SM6	60 %	5,80E-04
		SR	OV Monitor (SM2) falsely triggering OV event	Nuisance shutdown at nominal regulator loads.	P	25 %	1,45E-03	100 %						0,0E+00
		SR	OV Monitor (SM2) not triggering valid OV event	Regulated voltage higher than VA_OV	P	25 %	1,45E-03	0 %				SM6	60 %	5,80E-04
Analog BIST	Analog BIST (SM6)	SR	Analogue BIST (SM6) falsely detects misbehaviour of the linear voltage regulator	No effect [c]	P	50 %	3,50E-03	100 %						0,0E+00
		SR	Analogue BIST (SM6) does not detect misbehaviour of the linear voltage regulator	No effect[c]	P	50 %	3,50E-03	100 %						0,0E+00
						Σ								1.16E-03
				Total failure rate			0,01280							
				Total Safety-related			0,01280							
				Total Not Safety-related			0,00000							
									Single-Point Fault Metric		100 %			
													Latent-Fault Metric	90.0 %

[a] Depending on complexity it can be beneficial to have a dedicated entry in the FMEA giving more details about the potential root causes and the end effect of each failure mode.

[b] Fault model can be permanent fault (P) or transient fault (T); the example is limited to permanent faults.

[c] It requires more than two faults before it becomes safety-related: #1 fault: Failure on main safety mechanism (SM1, SM2 or SM3), #2 fault: LDO out of regulation, #3 fault: BIST Diagnostic failure.

Combining together the results of Table D.2 and Table D.3, the overall values are:

— Single-Point Fault Metric = 96,1 %; and

— Latent-Fault Metric = 95,7 %.

The following example considers the benefit of finer sub-part granularity for the same hardware element with a more stringent safety requirement: "The accuracy and the stability of the regulated voltage is such that VA < VA0+Δ and VA > VA0-Δ where VA0 is within Vmin – Vmax and Δ = 5mV."

The component can be considered in a safe state when the low accuracy/stability condition is detected and signalled to an external element of the system/item. The external system is responsible for fault reaction including transitioning the system or element to a safe state.

The example of quantitative analysis limited to permanent faults is reported in Table D.5 using the same format of Table C.1. The safety mechanisms considered in the analysis are:

— The voltage monitor detecting overvoltage (safety mechanism SM2) and under-voltage (safety mechanism SM1) failures;

— The independent ADC channel detecting variation of the regulated voltage higher than Δ = 5mV (safety mechanism SM3). The safety mechanism is described in 5.2.4.11;

— A current limiter detecting failures affecting circuits supplied by the low drop voltage (safety mechanism SM5). The safety mechanism is described in 5.2.4.5; and

— An analogue BIST detecting failures affecting the voltage monitor.

NOTE 8 The independence of the current limiter with respect to the regulator core has to be evaluated in the safety analysis.

NOTE 9 The ADC used as safety mechanism SM3 is assumed to be external to the hardware element under analysis and so it is not considered in the FMEA. There is an ADC included in the hardware element which is not SM3: It is therefore reported in the FMEA as not safety-related.

The coverage levels claimed by the safety mechanisms are reported in Table D.4.

Table D.4 — Safety mechanisms considered in the example with the new safety requirement

ID	**Safety mechanism**	**Claimed failure mode coverage**
SM1	Under-voltage (UV) Monitor	99,9 %
SM2	Over-voltage (OV) Monitor	99,9 %
SM3	Independent ADC monitoring	97 %
SM5	Current limiter	98 %
SM6	Analogue BIST diagnostics	90 %

NOTE 10 The effectiveness of safety mechanisms could be affected by dependent failures. Adequate measures are considered as described in 5.2.3.6.

Moreover, each safety mechanism signals the detection of a fault to an external element of the system/item which is then responsible to transition the system or element to a safe state

Under this assumption, the failure mode coverage with respect to latent failures related to the mission circuit is claimed to be 100 % according to ISO 26262-5:2018, Annex E.

Table D.5 shows the quantitative analysis for the mission part conducted at a finer level of granularity than the one in Table D.2 and Table D.3. The examples show that a different safety requirement impacts the level of partitioning and the diagnostic coverage requirement for one or more safety mechanisms.

NOTE 11 In this example the analysis with respect to transient faults is not reported but it can be added when relevant.

Table D.5 — Example of quantitative analysis in the case of fine granularity – mission parts

Part	Sub-part	Safety-related Component or No Safety-related Component	Failure Mode	Potential Effect of Failure Mode in Absence of Safety Mechanism (SM) on IC level[a]	Fault Model[b]	Failure distribution	Failure rate (FIT)	Amount of Safe Faults	Safety mechanism(s) preventing the violation of the safety requirement	Failure mode coverage with respect to violation of safety requirement	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) to prevent latent faults	Failure mode coverage with respect to Latent failures	Latent Multiple Point Fault failure rate / FIT
Low drop regulator	Regulator core	SR	Output voltage higher than a predefined high threshold of the prescribed range (i.e. Over voltage – OV)	Regulated voltage higher than VA_OV	P	14 %	1,49E-03	0 %	SM2	99,9 %	1,49E-06	SM2	100 %	0,00E+00
		SR	Output voltage lower than a predefined low threshold of the prescribed range (i.e. Under voltage – UV)	Regulated voltage lower than VA_UV	P	14 %	1,49E-03	0 %	SM1	99,9 %	1,49E-06	SM1	100 %	0,00E+00
		SR	Output voltage affected by spikes	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	14 %	1,49E-03	0 %	SM1 SM2	99,9 %	1,49E-06	SM1 SM2	100 %	0,00E+00
		SR	Output voltage oscillation within the prescribed range	Regulated voltage within the expected range but with low accuracy	P	14 %	1,49E-03	0 %	SM3	97,0 %	4,46E-05	SM3	100 %	0,00E+00
		SR	Output voltage fast oscillation outside the prescribed range but with average value within the prescribed range	Regulated voltage within the expected range but with low accuracy	P	14 %	1,49E-03	0 %	SM1 SM2	99,9 %	1,49E-06	SM1 SM2	100 %	0,00E+00
		SR	Output voltage drift within the prescribed range	Regulated voltage within the expected range but with low accuracy	P	14 %	1,49E-03	0 %	SM3	97,0 %	4,46E-05	SM3	100 %	0,00E+00
		SR	Quiescent current (i.e. current drawn by the regulator in order to control its internal circuitry for proper operation) exceeding the maximum value	Regulated voltage potentially with low accuracy depending on the actual quiescent current	P	14 %	1,49E-03	50 %	SM3	97,0 %	2,23E-05	SM3	100 %	0,00E+00
	Bandgap 1	SR	Output is stuck (high or low)	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	20 %	6,00E-04	0 %	SM1 SM2	99,9 %	6,00E-07	SM1 SM2	100 %	0,00E+00
		SR	Output is floating (e.g. open circuit)	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	20 %	6,00E-04	0 %	SM1 SM2	99,9 %	6,00E-07	SM1 SM2	100 %	0,00E+00
		SR	Output voltage oscillation within the expected range	Regulated voltage within the expected range but with low accuracy	P	20 %	6,00E-04	0 %	SM3	97,0 %	1,80E-05	SM3	100 %	0,00E+00
		SR	Incorrect output voltage value (i.e. outside the expected range)	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	20 %	6,00E-04	0 %	SM1 SM2	99,9 %	6,00E-07	SM1 SM2	100 %	0,00E+00
		SR	Output voltage accuracy too low, including drift	Regulated voltage within the expected range but with low accuracy	P	20 %	6,00E-04	50 %	SM3	97,0 %	9,00E-06	SM3	100 %	0,00E+00
		SR	Output voltage affected by spikes	Not applicable due to circuit implementation	P	0 %	0,00E+00	0 %			0,00E+00	SM1 SM2	100 %	0,00E+00
	Bias current generator	SR	One or more outputs are stuck (high or low)	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	10 %	2,00E-05	0 %	SM1 SM2	99,9 %	2,00E-08	SM1 SM2	100 %	0,00E+00

Table D.5 *(continued)*

Part	Sub-part	Safety-related Component or No Safety-related Component	Failure Mode	Potential Effect of Failure Mode in Absence of Safety Mechanism (SM) on IC level[a]	Fault Model[b]	Failure distribution	Failure rate (FIT)	Amount of Safe Faults	Safety mechanism(s) preventing the violation of the safety requirement	Failure mode coverage with respect to violation of safety requirement	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) to prevent latent faults	Failure mode coverage with respect to Latent failures	Latent Multiple Point Fault failure rate / FIT
		SR	One or more outputs are floating (e.g. open circuit)	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	10 %	2,00E-05	0 %	SM1 SM2	99,9 %	2,00E-08	SM1 SM2	100 %	0,00E+00
		SR	Incorrect reference current (i.e. outside the expected range)	Regulated voltage out of the expected range (VA_UV-VA_OV)	P	10 %	2,00E-05	0 %	SM1 SM2	99,9 %	2,00E-08	SM1 SM2	100 %	0,00E+00
		SR	Reference current accuracy too low , including drift	Regulated voltage within the expected range but with low accuracy	P	10 %	2,00E-05	0 %	SM3	97,0 %	6,00E-07	SM3	100 %	0,00E+00
		SR	Reference current affected by spikes	Regulated voltage with low accuracy for a limited time period if the spike is not filtered out; No effect otherwise	P	10 %	2,00E-05	50 %			1,00E-05	SM1 SM2	100 %	0,00E+00
		SR	Reference current oscillation within the expected range	Regulated voltage within the expected range but with low accuracy	P	10 %	2,00E-05	0 %	SM3	97,0 %	6,00E-07	SM3	100 %	0,00E+00
		SR	One or more bias currents outside the expected range while reference current is correct	Regulated voltage with low accuracy or out of regulation	P	10 %	2,00E-05	0 %	SM3	97,0 %	6,00E-07	SM3	100 %	0,00E+00
		SR	One or more bias currents accuracy too low , including drift	Regulated voltage within the expected range but with low accuracy	P	10 %	2,00E-05	0 %	SM3	97,0 %	6,00E-07	SM3	100 %	0,00E+00
		SR	One or more bias currents affected by spikes	Regulated voltage with low accuracy for a limited time period if the spike is not filtered out; No effect otherwise	P	10 %	2,00E-05	50 %			1,00E-05	SM1 SM2	100 %	0,00E+00
		SR	One or more bias currents oscillation within the expected range	Regulated voltage within the expected range but with low accuracy	P	10 %	2,00E-05	0 %	SM3	97,0 %	6,00E-07	SM3	100 %	0,00E+00
ADC	ADC	NSR			P	100 %	7,00E-03				0,0E+00			0,0E+00
							Σ				0,00017			0,0E+00
						Total failure rate	0,0206							
						Total Safety-related	0,0136							
						Total Not Safety-related	0,0070							
										Single-Point Fault Metric	98,8 %			
													Latent-Fault Metric	100 %

[a] Depending on complexity it can be beneficial to have a dedicated entry in the FMEA giving more details about the potential root causes and the end effect of each failure mode.

[b] Fault model can be permanent fault (P) or transient fault (T); the example is limited to permanent faults.

D.1.1 Example of computation of failure rate for analogue component

Calculation methods to derive the base failure rate for analogue and mixed signal components are described in 4.6.

The base failure rate is allocated to the different elements composing the hardware component. Different allocation methods can be applied depending on the type of elements considered.

The base failure rate can be considered proportional to the area of the circuit.

EXAMPLE 1 The base failure rate is divided by the overall area of the component in order to obtain FIT/mm^2 for each relevant fault model.

Table D.6 — Base failure rate allocation based on area

Fault model	Failure rate value	Unit
Permanent faults	2,00E-02	FIT/mm^2
Transient faults	2,00E-05	FIT/mm^2

The failure rate of each sub-part of the analogue and mixed signal component shown in the previous example is computed by using the FIT/mm^2 reported in Table D.6.

The results of the computation, considering the block diagrams of the previous example, are reported in Table D.7.

Table D.7 — Failure rate for each part/sub-part

Part	Sub-part	Block Area (mm^2)	Failure rate Permanent faults (FIT)	Failure rate Transient faults (FIT)
Low Drop Regulator	Regulator Core	0,52	0,0104	0,0000104
	BANDGAP 1	0,15	0,0030	0,0000030
	Bias Current Generator	0,01	0,0002	0,0000002
	Current Limiter	0,075	0,0015	0,0000015
	TOTAL	**0,755**	**0,0151**	**0,0000151**
Voltage Monitor	CMP1	0,03	0,0006	0,0000006
	CMP2	0,03	0,0006	0,0000006
	Passive Network	0,08	0,0016	0,0000016
	BANDGAP 2	0,15	0,0030	0,0000030
	TOTAL	**0,29**	**0,0058**	**0,0000058**
ADC	ADC	0,85	0,0170	0,0000170
Analogue BIST	Analogue BIST	0,35	0,0070	0,0000070
TOTAL		**2,535**	**0,0507**	**0,0000507**

NOTE 1 The numbers reported here are only examples.

NOTE 2 Block area reported here includes internal routing. Routing at top level, if relevant, is included in a separate block.

As an alternative to the area-based approach, as seen in 5.1.7.1, the failure rate and failure mode distribution can be estimated based on the number of equivalent transistors for each sub-part or elementary sub-part. In the case of mixed signal or analogue components, distinction between active devices, passive devices and routing can be taken into account in the estimation of the number of equivalent transistors. The selection of the method used can be based on the layout (or planned layout) of the circuit under analysis or on the analysis of how failure modes are shared between the hardware elements.

NOTE 3 For a transient fault model, the base failure rate proportional to area is a simplified example because, in reality, not each element in a mixed signal circuit has the same probability of failure.

EXAMPLE 2 In switched-capacitor architectures, the capacitors holding the signal are more sensitive with respect to transient faults than other portions of the circuit because they are used as memory elements.

Annex E (informative) Examples of quantitative analysis for PLD component

E.1 Architecture of the example

Figure E.1 is an example system used to demonstrate the concepts outlined in this clause. The system is intended for a safety-related application where two microcontrollers are used for redundancy and the final control output is implemented using a PLD. The two microcontrollers send their values to the PLD via SPI (Serial Peripheral Interface) and the PLD communicates its output via a CAN (Controller Area Network) bus. For this example, it is assumed that a calculated output too high (i.e. greater than the value that would have been determined by a non-faulted system plus a threshold) is a potential hazard but an output too low is acceptable from a functional safety point-of-view. It is also assumed that the components receiving the CAN message can detect the loss of CAN messages and take appropriate remedial action such as defaulting the receive signal to its minimum value and that the receiving module can tolerate corrupted CAN messages (i.e. values higher than intended) for x number of messages.

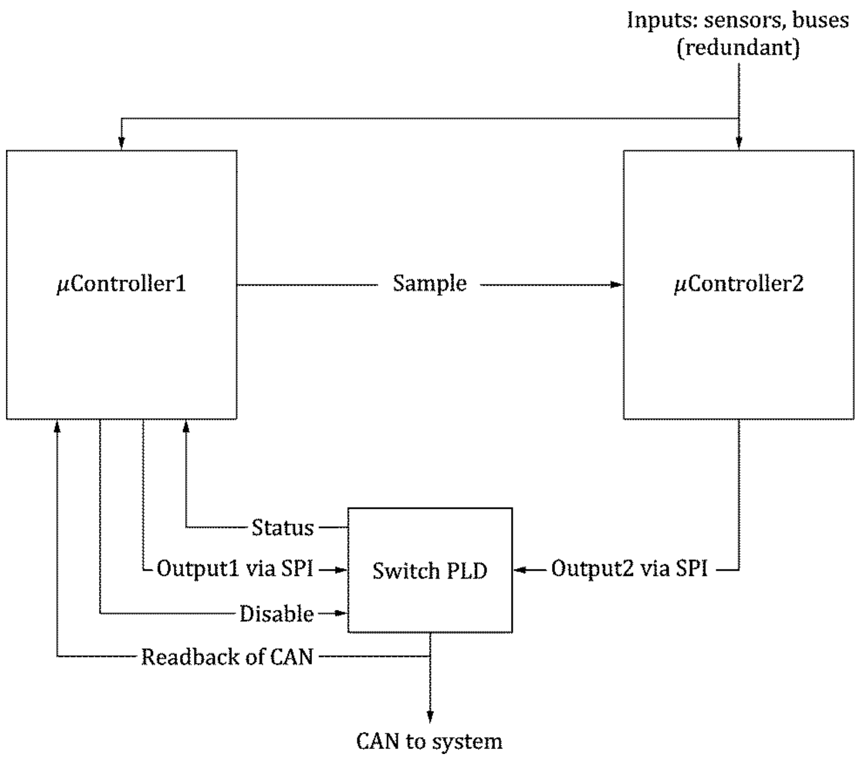


Figure E.1 — Example of PLD usage – output switch

NOTE The hardware component “Controller” is implemented using two microcontrollers and one PLD.

The derived safety requirement for the hardware component “Controller” could be:

— SafReq_hardware_Comp_Controller_001: The output of a wrong value which is larger than the correct value plus a threshold for x number of messages in-a-row shall be avoided; and

— SafReq_hardware_Comp_Controller_002: Undetected lack of CAN outputs for longer than y ms shall be avoided.

The hardware component “Controller” is implemented using two microcontrollers (μController1 and μController2) and one PLD. Both μController1 and μController2 have the same input/output history and send their calculated outputs to the PLD. Both outputs agree within the threshold when no fault has occurred. The PLD is responsible for taking the minimum of the two signals and communicating this output to the rest of the system via CAN. SafReq_hardware_Comp_Controller_002 can be fulfilled by entities outside of the controller (e.g. timeout supervision).

The derived safety requirement for the PLD could be:

— SafReq_PLD_001: Output of a value larger than the minimum of the two input values from µController1 and µController2 shall be avoided (derived from SafReq_hardware_Comp_Controller_001); and

— SafReq_PLD_002: Undetected corruption of the CAN output value from PLD which leads to an output too high shall be avoided (derived from SafReq_hardware_Comp_Controller_001).

The following clause addresses, as an example, two different approaches for the PLD's safety and dependent failures analysis. The safety analysis and the dependent failures analysis concerning µController1 and µController2 are out of scope of this document.

Failures of the PLD can be addressed by two approaches:

— Utilizing safety measures which are external to the PLD, or

— Utilizing safety measures which are internal to the PLD. The PLD includes diagnostic measures to detect faults of the PLD. Faults are communicated via the status signal to µController1, which can disable the PLD based on the severity of the fault.

E.2 PLD external measures

The following safety mechanisms are implemented by elements other than the PLD:

— SafMech_PLD_001: CAN Read back and comparison. The CAN output of the PLD is read back by µController1. µController1 checks if the PLD has output a value equal or less than its output. If this check fails the µController1 disables the PLD via the Disable signal; and

— SafMech_Network_001: The receivers implement a time-out monitoring.

As a first step of the safety analysis the relevant failure modes can be identified. Since none of the safety mechanisms are implemented within the PLD it is sufficient to describe the observable failure modes on its output level:

— FM_PLD_OP_01: no output;

— FM_PLD_OP_02: output of old message;

— FM_PLD_OP_03: corrupt output;

— FM_PLD_OP_04: do not output minimum value;

— FM_PLD_OP_05: always output µController1 value;

— FM_PLD_OP_06: always output µController2 value; and

— FM_PLD_OP_07: active "Disable" discrete signal does not prevent CAN transmission.

As described in 5.3.3.1.3, to derive a probability distribution over the above mentioned failure modes typically detailed knowledge of the PLD internal structure is necessary. If this information is not available and no argument can be given why one of the failure modes is more likely than the other, the approach described in 5.3.3.1.3 a) can be adopted.

Table E.1 — Example of a PLD safety analysis in the case of PLD external measures

Failure mode	Permanent distribution	Transient distribution	PVSG?	MPF?	Safety mechanisms
FM_PLD_OP_01: No output	14,2 %	14,2 %	1	0	SafMech_Network_001
FM_PLD_OP_02: Output of old message	14,2 %	14,2 %	1	0	SafMech_PLD_001
FM_PLD_OP_03: Corrupt output	14,2 %	14,2 %	1	0	SafMech_PLD_001
FM_PLD_OP_04: Do not output minimum value	14,2 %	14,2 %	0	1	SafMech_PLD_001
FM_PLD_OP_05: Always output µController1 value	14,2 %	14,2 %	0	1	
FM_PLD_OP_06: Always output µController2 value	14,2 %	14,2 %	0	1	SafMech_PLD_001
FM_PLD_OP_07: Active “Disable” discrete signal does not prevent CAN transmission	14,2 %	14,2 %	0	1	
NOTE PVSG = potential to directly violate the safety goal; MPF = multiple-point failure					

As far as the dependent failures analysis (out of scope of this document) is concerned the correlation of following elements could be of interest:

— PLD & µController1

— PLD & µController2

— µController1 & µController2

E.3 PLD internal measures

The rest of the example considers utilizing safety measures which are internal to the PLD. The internal architecture of the PLD is presented in Figure E.2. The data sent from the µController has to be buffered before it can be transferred via the CAN bus. The buffers are implemented as user memory, whereas the state machine controlling the buffer operation, the multiplexer are implemented by logic blocks and the CAN module is a fixed function IP. The functionality of the logic blocks and the routing between the blocks and memory are controlled by the configuration technology. For simplicity the switch control logic which determines whether data from Buffer 1 or Buffer 2 is sent is not covered in this example.

The design is also susceptible to intermittent and permanent hardware failures. Any chip infrastructure such as clock or power could be a source of a common mode failure. These failures can be addressed by redundancy with detection and reporting for single mode failures. Other examples include incorrect load of code at initialization and bit flip in memory. These could be detected using checksums and parity; however, some of these failures could result in a possible violation of the safety goal and would be an unacceptable risk. Error-detection-correction codes (ECC) are a superior technique as they correct errors and could report after correction that a potential problem exists in the chip. Single failures in the I/O of the chip only impact one output and would represent less risk.

NOTE 1 Depending on the functionality of the implemented circuitry it is necessary to perform further activities besides correcting the fault to restore the functionality of the design (e.g. a fault in the configuration technology leads to a non-recoverable state of a state-machine, even though the fault in the configuration technology was corrected).

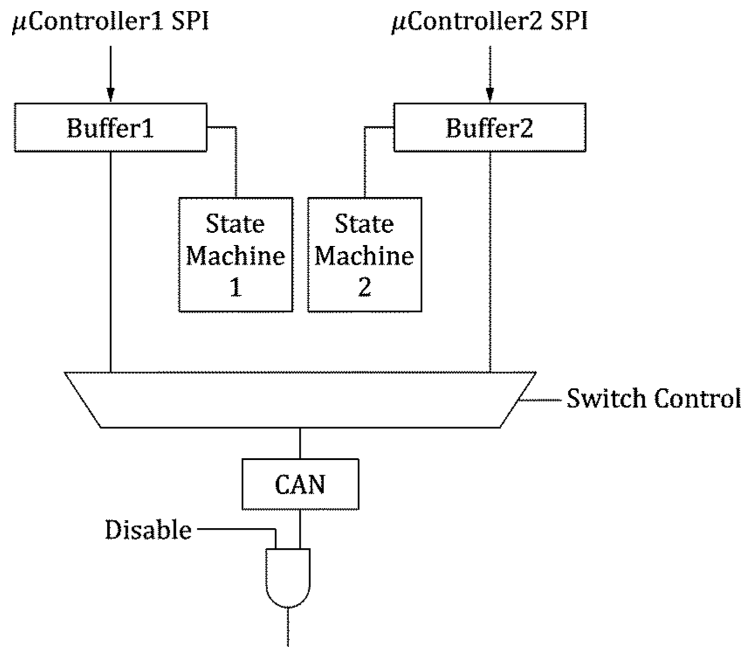


Figure E.2 — PLD architecture

If the fault has the potential to violate the safety goal without being detected by the internal safety mechanisms it would be detected by μController1 through loss of the CAN signal or a mismatch between commanded outputs and the CAN read. This is acceptable if μController1 can disable the PLD via the “Disable” signal. A dependent failures analysis is done to ensure that the risk of the PLD violating a safety goal in combination with the failure of the deactivation via the disable signal is sufficiently low.

EXAMPLE A potential hazard could occur if the switch is unable to respond to the disable command from μController1. This would be a multiple-point fault situation as if both μController1 and 2 are good; the PLD output would still represent safe values. There would not be a potential risk until one of the μControllers fails and the PLD responds incorrectly. To detect this multiple-point fault, a periodic test of the disable logic can be implemented. Since this would be performed at system or element level, the specific details are out of scope of this document and are not described further.

NOTE 2 In this simple example, the external measures can replace the internal safety mechanisms. In general, cases exist in which the internal measures are necessary to reach target diagnostic coverage and therefore the detailed analysis of internal safety mechanisms described in this clause is applied.

Random hardware faults can be analysed by applying an inductive fault analysis (e.g. FMEA) on the design. Faults of the user design, but also faults of the PLD technology are taken into account and consider permanent and transient faults. The qualitative analysis of the design is followed up with a quantitative analysis, similar to the one described in Annex C of this part of ISO 26262.

As described in 5.3.3.1, inputs to the quantitative analysis can be made available by the PLD manufacturer with regard to the failure rates of the elementary sub-parts of the PLD and the failure mode distribution.

NOTE 3 In this case of PLD internal measures, for failure mode distribution determination the approaches like described in 5.3.3.1.3 b) or c) are preferable.

Table E.2 provides a framework for a quantitative analysis of the above design, which can be augmented with information similar to Table C.1.

NOTE 4 As discussed in 5.1.7, the necessary level of detail can depend on the stage of the analysis and on the safety mechanisms used.

Table E.2 — Example framework for quantitative analysis of scenario 2

Part	Sub-part	Safety-related (SR) or not safety-related (NSR) element?	Failure modes
I/O interface	I/O buffer	SR	Permanent
	Configuration technology	SR	Permanent
			Transient
	Routing resources	SR	Permanent
			Transient
Buffer 1	RAM data bits	SR	Permanent
			Transient
	Address decoder	SR	Permanent
			Transient
	Test/redundancy	SR	Permanent
			Transient
	Configuration technology	SR	Permanent
			Transient
	Routing resources	SR	Permanent
			Transient
Buffer 2	RAM data bits	SR	Permanent
			Transient
	Address decoder	SR	Permanent
			Transient
	Test/redundancy	SR	Permanent
			Transient
	Configuration technology	SR	Permanent
			Transient
	Routing resources	SR	Permanent
			Transient
State Machine 1	Logic blocks	SR	Permanent
			Transient
	Configuration technology	SR	Permanent
			Transient
	Routing resources	SR	Permanent
			Transient
State Machine 2	Logic blocks	SR	Permanent
			Transient
	Configuration technology	SR	Permanent
			Transient
	Routing resources	SR	Permanent
			Transient
Multiplexer	Logic blocks	SR	Permanent
			Transient
	Configuration technology	SR	Permanent
			Transient
	Routing resources	SR	Permanent
			Transient
CAN	Logic	SR	Permanent
			Transient

Part	Sub-part	Safety-related (SR) or not safety-related (NSR) element?	Failure modes
	RAM data bits	SR	Permanent
			Transient
	Address decoder	SR	Permanent
			Transient

NOTE 1 Depending on the role of each PLD part in the system, a more detailed analysis can be necessary.
NOTE 2 The example does not list the quantitative numbers for simplicity.

The analysis also includes PLD related external components such as power supplies, clocks and reset circuitry. Further, if the configuration of the PLD is loaded from an external device, it is analysed if the loading of the configuration into the PLD is considered safety-related or if the process of loading the configuration can lead to a failure of the item.

In particular, if the PLD is loaded from µController1, common cause failures in µController1 that affect the loading mechanism and µController1 functionality is considered. A dependent failures analysis is performed if separate channels or diagnostic measures are implemented in the PLD. An example of such an analysis can be found in Annex B of this part of ISO 26262. In this example independence of the individual sub-parts is not considered as the detection of a fault of the PLD is performed by reading back the output of the CAN module with a µController.

Bibliography

[1] Askari, S., Nourani, M., "Design methodology for mitigating transient errors in analogue and mixed-signal circuits", Circuits, Devices & Systems, IET, Vol.6, No.6, pp.447-456, Nov. 2012.

[2] Baumann, R.C., “Radiation-Induced Soft Errors in Advanced Semiconductor Technologies”, IEEE Transactions on device and materials reliability, Vol. 5, No. 3, Sep. 2005.

[3] Baruah, S.K., Goossens, J., "Rate-monotonic scheduling on uniform multiprocessors", Proceedings of the 23rd International Conference on Distributed Computing Systems 2003, pp.360-366.

[4] Börcsök, J., Schaefer, S., Ugljesa, E., "Estimation and Evaluation of Common Cause Failures", Second International Conference on Systems, 2007, ICONS '07, pg.41.

[5] Bressoud, T.C., Schneider, F.B., “Hypervisor-based fault tolerance”, Proceedings of the fifteenth ACM symposium on Operating systems principles, 1995, pp.1–11.

[6] Chattopadhyay, S., Kee, C.L., Roychoudhury, A., Kelter, T., Marwedel, P., Falk, H., "A Unified WCET Analysis Framework for Multi-core Platforms", 2012 IEEE 18th Real-Time and Embedded Technology and Applications Symposium, 2012, pp.99-108.

[7] Clegg, J.R., "Arguing the safety of FPGAs within safety critical systems," Incorporating the SaRS Annual Conference, 4th IET International Conference on Systems Safety, 2009, pp.1-6.

[8] Conmy, P.M., Pygott, C., Bate, I, "VHDL guidance for safe and certifiable FPGA design," 5th IET International Conference on System Safety, 2010, pp.1-6.

[9] FIDES guide 2009 Edition A (September 2010), “Reliability Methodology for Electronic Systems”.

[10] Fleming, P.R., Olson, B.D., Holman, W.T., Bhuva, B.L., Massengill, L.W., "Design Technique for Mitigation of Soft Errors in Differential Switched-Capacitor Circuits", IEEE Transactions on Circuits and Systems II: Express Briefs, Vol.55, No.9, pp.838-842, Sep. 2008.

[11] Franklin, M., “Incorporating Fault Tolerance in Superscalar Processors”, Proceedings of International Conference on High Performance Computing, Dec. 1996.

[12] Hayek, A, Borcsok, J., "SRAM-based FPGA design techniques for safety-related systems conforming to IEC 61508 a survey and analysis", 2012 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA), 2012, pp.319-324.

[13] Heiser, G., "The role of virtualization in embedded systems", Proceedings of the 1st workshop on Isolation and integration in embedded systems, IIES '08, 2008, pp.11-16.

[14] IEC 61508-2:2010, Functional safety of electrical/electronic/programmable electronic safety-related systems.

[15] IEC 61709 Ed. 3.0 (2016) / IEC 56/1648/CDV:2015-12, Electrical components – Reliability - Reference conditions for failure rates and stress models for conversion.

[16] JEDEC - JEP122G (October 2011), Failure Mechanisms and Models for Semiconductor Devices.

[17] JEDEC - JESD89A (October 2006), Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices.

[18] Keckler, S.W., Olukotun, K., Hofstee, H.P., Multicore Processors and Systems, Springer, 2009.

[19] Kervarreca, G., et al., A universal field failure based reliability prediction model for SMD Integrated Circuits, Elsevier, 2000.

[20] Lazzari, C., et al., “Phase-Locked Loop Automatic Layout Generation and Transient Fault Injection Analysis: A Case Study” 12th IEEE International On-Line Testing workshop, Como, Italy, July 10-12, 2006, pp. 117-127.

[21] Mariani, R., et al., “Practical experiences of fault insertion in microcontrollers for automotive applications”, 15th IEEE European Test Symposium, ETS2010, May 2010.

[22] Mariani, R., “Soft errors on digital components”, in Fault Injection Techniques and Tools for Embedded Systems Reliability Evaluation, Frontiers in Electronic Testing, Vol. 23, Kluwer Academic Publisher, 2003, pp. 49-60.

[23] MIL-HDBK-217, Military Handbook – Reliability Prediction of Electronic Equipment.

[24] Mitra, S., Saxena, N.R., McCluskey, E.J., "Common-mode failures in redundant VLSI systems: a survey", IEEE Transactions on Reliability, Vol.49, No.3, Sep. 2000, pp.285-295.

[25] Mukherjee, S. S. et al., “A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor in microarchitecture”, MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium, Dec. 2003, pp. 29-40.

[26] Niimi, Y., et al., "Virtualization Technology and Using Virtual CPU in the Context of ISO 26262: The E-Gas Case Study", SAE Technical Paper, April 2013.

[27] Paolieri, M., Mariani, R., "Towards functional-safe timing-dependable real-time architectures", IEEE 17th International On-Line Testing Symposium (IOLTS), 2011, pp.31-36.

[28] Singh, M., et al., "Transient Fault Sensitivity Analysis of Analog-to-Digital Converters (ADCs)", Proceedings of the IEEE Workshop on VLSI (WVLSI '01), 2001.

[29] White, M., Bernstein, J.B., “Microelectronics Reliability: Physics-of-Failure Based Modeling and Lifetime Evaluation”, JPL Publication, 2008.

[30] Arlat J. et al., “Fault Injection and Dependability Evaluation of Fault-Tolerant Systems”, IEEE Transactions on Computers, Vol. 42, No. 8, August 1993, pp. 913—9.

[31] Benso A. and Prinetto P “Fault Injection Techniques and Tools for Embedded Systems Reliability Evaluation”, Kluwer Academic Publishers, 2003, ISBN 1-4020-7589-8.

[32] Wei, Jiesheng, et al. "Quantifying the accuracy of high-level fault injection techniques for hardware faults." Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on. IEEE, 2014.

[33] Kejun Wu; Pahlevanzadeh, H.; Peng Liu; Qiaoyan Yu, "A new fault injection method for evaluation of combining SEU and SET effects on circuit reliability," Circuits and Systems (ISCAS), 2014 IEEE International Symposium on , vol., no., pp.602,605, 1-5 June 2014.

[34] Benso, A.; Bosio, A.; Di Carlo, S.; Mariani, R., "A Functional Verification based Fault Injection Environment," Defect and Fault-Tolerance in VLSI Systems, 2007. DFT '07. 22nd IEEE International Symposium on , vol., no., pp.114,122, 26-28 Sept. 2007.

[35] Van de Goor, A.J.: Testing Semiconductor Memories, Theory and Practice, 2nd edn. ComTex Publishing, Gouda.

[36] Enamul Amyeen, M., et al., "Evaluation of the Quality of N-Detect Scan ATPG Patterns on a Processor", Proceedings of the International Test Conference 2004, ITC'04, p.669-678.

[37] Benware, B., et al., "Impact of Multiple-Detect Test Patterns on Product Quality", Proc. of the International Test Conference 2003, ITC'03, pp. 1031-1040.

[38] Patel, J. H., "Stuck-At Fault: A Fault Model for the Next Millennium?", Proceedings of the International Test Conference 1998, ITC'98, pp.1166.

[39] Siemens AG, "Failure Rates of Components – Expected Values, General", SN 29500 (2004).

[40] Paschalis, A., and Gizopoulos, D., Effective Software-Based Self-Test Strategies for On-Line Periodic Testing of Embedded Processors. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 24, 1 (Jan.2005), 88-99.

[41] IEC TR 62380:2004, Reliability data handbook — Universal model for reliability prediction of electronics components, PCBs and equipment.

[42] The International Technology Roadmap For Semiconductors (ITRS), 2009 Edition.

[43] IEEE Std 2700™-2014, IEEE Standard for Sensor Performance Parameter Definitions.

[44] White, Richard M., "A Sensor Classification Scheme", in IEEE TRANSACTIONS ON ULTRASONICS, FERROELECTRICS, AND FREQUENCY CONTROL, VOL. UFFC-34, NO. 2,1987,pp. 124-126.

[45] Gupta, Vijay; R. Snow, M.C. Wu, A. Jain, J. Tsai., "Recovery of Stiction-Failed MEMS Structures Using Laser-Induced Stress Waves" in Journal of Microelectromechanical Systems, Vol. 13, No. 4, August 2004, pp. 696-700.

[46] Walraven, Jeremy A. "Failure Mechanisms in MEMS." in IEEE ITC International Test Conference, Paper 33.1, 2003, pp.828-833.

[47] J. Iannacci, Reliability of MEMS: A perspective on failure mechanisms, improvement solutions and best practices at development level, Displays VOL. 37, April 2015, pp. 62-71.

[48] Vonkyoung Kim; Chen, T., "Assessing SRAM test coverage for sub-micron CMOS technologies," VLSI Test Symposium, 1997., 15th IEEE , vol., no., pp.24,30, 27 Apr-1 May 1997.

[49] Ginez, O.; Daga, J.-M.; Combe, M.; Girard, P.; Landrault, C.; Pravossoudovitch, S.; Virazel, A., "An overview of failure mechanisms in embedded flash memories," in VLSI Test Symposium, 2006. Proceedings. 24th IEEE , vol., no., pp.6 pp.-113, April 30 2006-May 4 2006.

[50] Di Carlo, S.; Fabiano, M.; Piazza, Roberto; Prinetto, P., "Exploring modeling and testing of NAND flash memories," in Design & Test Symposium (EWDTS), 2010 East-West, vol., no., pp.47-50, 17-20 Sept. 2010.

[51] Al-Ars, Z.; Hamdioui, S.; van de Goor, A.J., "Space of DRAM Fault Models and Corresponding Testing," in Design, Automation and Test in Europe, 2006. DATE '06. Proceedings , vol.1, no., pp.1-6, 6-10 March 2006.

[52] ISO TS 16949, Quality management systems -- Particular requirements for the application of ISO 9001:2008 for automotive production and relevant service part organizations.

[53] Daniel J. Sorin, Mark D. Hill, and David A. Wood. 2011. A Primer on Memory Consistency and Cache Coherence (1st ed.). Morgan & Claypool Publishers.

[54] JEDEC - JESD94 Application Specific Qualification Using Knowledge Based Test Methodology

[55] G. Kervarrec et al., “A universal reliability prediction model for SMD integrated circuits based on field failures” - Microelectronics Reliability Volume 39, Issues 6–7, June–July 1999, Pages 765-771 - European Symposium on Reliability of Electron Devices, Failure Physics and Analysis

[56] Standardized E-GAS Monitoring Concept for Gasoline and Diesel Engine Control Units,https://www.iav.com/publikationen/technische-veroeffentlichungen/e-gas-monitoring-concepts"

[57] P1804 - IEEE Draft Standard for Fault Accounting and Coverage Reporting to Digital Modules (FACR)

[58] R. Leveugle, A. Calvez, P. Maistri and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," 2009 Design, Automation & Test in Europe Conference & Exhibition, Nice, 2009, pp. 502-506

[59] “Understanding Binomial Confidence Intervals” by Philip Mayfield, http://www.sigmazone.com/binomial_confidence_interval.htm