

INTERNATIONAL
STANDARD

ISO
26262-1

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 1:
Vocabulary**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 1: Vocabulaire*



Reference number
ISO 26262-1:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

The reproduction of the terms and definitions contained in this International Standard is permitted in teaching manuals, instruction booklets, technical publications and journals for strictly educational or implementation purposes. The conditions for such reproduction are: that no modifications are made to the terms and definitions; that such reproduction is not permitted for dictionaries or similar publications offered for sale; and that this International Standard is referenced as the source document.

With the sole exceptions noted above, no other part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction.....	v
Scope	1
1 Terms and definitions	1
2 Abbreviated terms.....	18
Bibliography.....	21
Alphabetical index.....	22

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-1 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

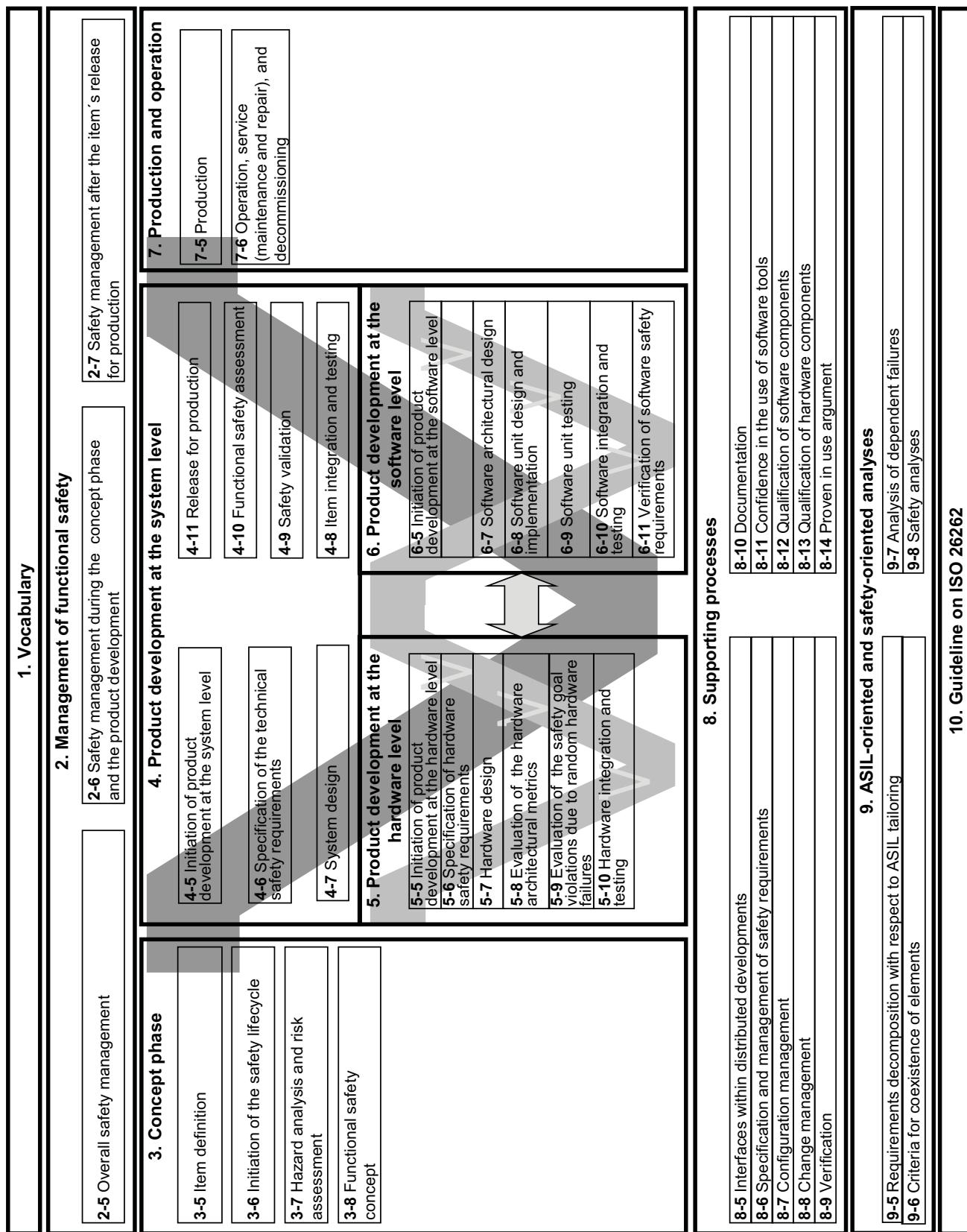


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 1: Vocabulary

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the terms, definitions and abbreviated terms for application in all parts of ISO 26262.

1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

1.1 **allocation**

assignment of a requirement to an architectural **element** (1.32)

NOTE Intent is not to divide an atomic requirement into multiple requirements. Tracing of an atomic **system** (1.129) level requirement to multiple lower level atomic requirements is allowed.

1.2 **anomaly**

condition that deviates from expectations, based, for example, on requirements, specifications, design documents, user documents, standards, or on experience

NOTE Anomalies can be discovered, among other times, during the **review** (1.98), **testing** (1.134), analysis, compilation, or use of **components** (1.15) or applicable documentation.

1.3

architecture

representation of the structure of the **item** (1.69) or functions or **systems** (1.129) or **elements** (1.32) that allows identification of building blocks, their boundaries and interfaces, and includes the **allocation** (1.1) of functions to hardware and software elements

1.4

assessment

examination of a characteristic of an **item** (1.69) or **element** (1.32)

NOTE A level of **independence** (1.61) of the party or parties performing the assessment is associated with each assessment.

1.5

audit

examination of an implemented process

1.6

Automotive Safety Integrity Level

ASIL

one of four levels to specify the **item's** (1.69) or **element's** (1.32) necessary requirements of ISO 26262 and **safety measures** (1.110) to apply for avoiding an unreasonable **residual risk** (1.97), with D representing the most stringent and A the least stringent level

1.7

ASIL decomposition

apportioning of safety requirements redundantly to sufficiently independent **elements** (1.32), with the objective of reducing the **ASIL** (1.6) of the redundant safety requirements that are allocated to the corresponding elements

1.8

availability

capability of a product to be in a state to execute the function required under given conditions, at a certain time or in a given period, supposing the required external resources are available

1.9

baseline

version of a set of one or more work products, **items** (1.69) or **elements** (1.32) that is under configuration management and used as a basis for further development through the change management process

NOTE See ISO 26262-8:2011, Clause 8.

1.10

branch coverage

percentage of branches of the control flow that have been executed

NOTE 1 100 % branch coverage implies 100 % **statement coverage** (1.127).

NOTE 2 An if-statement always has two branches - condition true and condition false - independent of the existence of an else-clause.

1.11

calibration data

data that will be applied after the software build in the development process

EXAMPLE Parameters (e.g. value for low idle speed, engine characteristic diagrams); vehicle specific parameters (adaptation values) (e.g. limit stop for throttle valve); variant coding (e.g. country code, left-hand/right-hand steering).

NOTE Calibration data cannot contain executable or interpretable code.

1.12 candidate

item (1.69) or **element** (1.32) whose definition and conditions of use are identical to, or have a very high degree of commonality with, an item or element that is already released and in operation

NOTE This definition applies where candidate is used in the context of a **proven in use argument** (1.90).

1.13 cascading failure

failure (1.39) of an **element** (1.32) of an **item** (1.69) causing another element or elements of the same item to fail

NOTE Cascading failures are **dependent failures** (1.22) that are not **common cause failures** (1.14). See Figure 2, Failure A.



Figure 2 — Cascading failure

1.14 common cause failure CCF

failure (1.39) of two or more **elements** (1.32) of an **item** (1.69) resulting from a single specific event or root cause

NOTE Common cause failures are **dependent failures** (1.22) that are not **cascading failures** (1.13). See Figure 3.

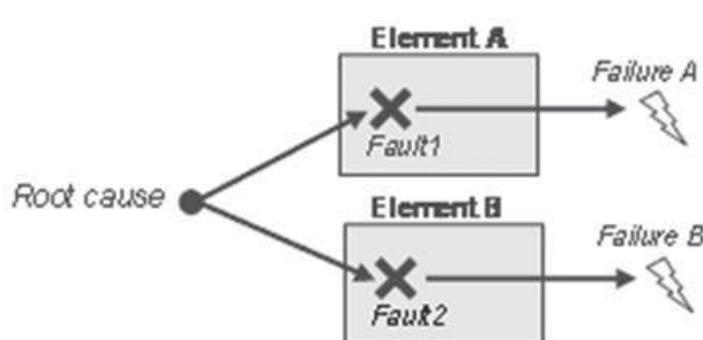


Figure 3 — Common cause failure

1.15 component

non-system (1.129) level **element** (1.32) that is logically and technically separable and is comprised of more than one **hardware part** (1.55) or of one or more **software units** (1.125)

NOTE A component is a part of a system.

1.16 configuration data

data that is assigned during software build and that controls the software build process

EXAMPLE Pre-processor instructions; software build scripts (e.g. XML configuration files).

NOTE 1 Configuration data cannot contain executable or interpretable code.

NOTE 2 Configuration data controls the software build. Only code, or data selected by configuration data can be included in the executable code.

1.17

confirmation measure

confirmation review (1.18), **audit** (1.5) or **assessment** (1.4) concerning **functional safety** (1.51)

1.18

confirmation review

confirmation that a work product meets the requirements of ISO 26262 with the required level of **independence** (1.61) of the reviewer

NOTE 1 A complete list of confirmation reviews is given in ISO 26262-2.

NOTE 2 The goal of confirmation reviews is to ensure compliance with ISO 26262.

1.19

controllability

ability to avoid a specified **harm** (1.56) or damage through the timely reactions of the persons involved, possibly with support from **external measures** (1.38)

NOTE 1 Persons involved can include the driver, passengers or persons in the vicinity of the vehicle's exterior.

NOTE 2 The parameter C in **hazard analysis and risk assessment** (1.58) represents the potential for controllability.

1.20

dedicated measure

measure to ensure the **failure rate** (1.41) claimed in the evaluation of the probability of violation of **safety goals** (1.108)

EXAMPLE Design feature [such as **hardware part** (1.55) over-design (e.g. electrical or thermal stress rating) or physical separation (e.g. spacing of contacts on a printed circuit board)]; special sample test of incoming material to reduce the **risk** (1.99) of occurrence of **failure modes** (1.40) which contribute to the violation of safety goals; burn-in test; dedicated control plan.

1.21

degradation

strategy for providing **safety** (1.103) by design after the occurrence of **failures** (1.39)

NOTE Degradation can include reduced functionality, reduced performance, or both reduced functionality and performance.

1.22

dependent failures

failures (1.39) whose probability of simultaneous or successive occurrence cannot be expressed as the simple product of the unconditional probabilities of each of them

NOTE 1 Dependent failures A and B can be characterized when

$$P_{AB} \neq P_A \times P_B$$

where

P_{AB} is the probability of the simultaneous occurrence of failure A and failure B;

P_A is the probability of the occurrence of failure A;

P_B is the probability of the occurrence of failure B.

NOTE 2 Dependent failures include **common cause failures** (1.14) and **cascading failures** (1.13).

1.23**detected fault**

fault (1.42) whose presence is detected within a prescribed time by a **safety mechanism** (1.111) that prevents the fault from being latent

EXAMPLE The fault can be detected by a dedicated **safety mechanism** (1.111) (e.g. detection of the **error** (1.36) and notifying the driver via an alerting device on the instrument panel) as defined in the **functional safety concept** (1.52).

1.24**development interface agreement****DIA**

agreement between customer and supplier in which the responsibilities for activities, evidence or work products to be exchanged by each party are specified

1.25**diagnostic coverage**

proportion of the hardware **element** (1.32) **failure rate** (1.41) that is detected or controlled by the implemented **safety mechanisms** (1.111)

NOTE 1 Diagnostic coverage can be assessed with regard to **residual faults** (1.96) or with regard to latent **multiple-point faults** (1.77) that might occur in a hardware element.

NOTE 2 The definition can be represented in terms of the equations given in ISO 26262-5.

NOTE 3 Safety mechanisms implemented at different levels in the **architecture** (1.3) can be considered.

1.26**diagnostic test interval**

amount of time between the executions of online diagnostic tests by a **safety mechanism** (1.111)

1.27**distributed development**

development of an **item** (1.69) or **element** (1.32) with development responsibility divided between the customer and supplier(s) for the entire item or element, or for subsystems

NOTE Customer and supplier are roles of the cooperating parties.

1.28**diversity**

different solutions satisfying the same requirement with the aim of **independence** (1.61)

EXAMPLE Diverse programming; diverse hardware.

NOTE Diversity does not guarantee independence, but addresses certain types of **common cause failures** (1.14).

1.29**dual-point failure**

failure (1.39) resulting from the combination of two independent **faults** (1.42) that leads directly to the violation of a **safety goal** (1.108)

NOTE 1 Dual-point failures are **multiple-point failures** (1.76) of order 2.

NOTE 2 Dual-point failures that are addressed in ISO 26262 include those where one fault affects a **safety-related element** (1.113) and another fault affects the corresponding **safety mechanism** (1.111) intended to achieve or maintain a **safe state** (1.102).

NOTE 3 For a dual-point failure to directly violate a safety goal, the presence of both independent faults is necessary, i.e. the violation of a safety goal due to a combination of a **residual fault** (1.96) with a **safe fault** (1.101) is not considered a dual-point failure since the residual fault leads to a violation of a safety goal with or without the presence of a second independent fault.

1.30

dual-point fault

individual **fault** (1.42) that, in combination with another independent fault, leads to a **dual-point failure** (1.29)

NOTE 1 A dual-point fault can only be recognized after the identification of dual-point failure, e.g. from cut set analysis of a fault tree.

NOTE 2 See also **multiple-point fault** (1.77).

1.31

electrical and/or electronic system

E/E system

system (1.129) that consists of electrical and/or electronic **elements** (1.32), including programmable electronic elements

EXAMPLE Power supply; sensor or other input device; communication path; actuator or other output device.

1.32

element

system (1.129) or part of a system including **components** (1.15), hardware, software, **hardware parts** (1.55), and **software units** (1.125)

1.33

embedded software

fully-integrated software to be executed on a processing **element** (1.32)

NOTE The processing element is normally a micro-controller, a field programmable gate array (FPGA) or an application-specific integrated circuit (ASIC), but it can also be a more complex **component** (1.15) or subsystem.

1.34

emergency operation

degraded functionality from the state in which a **fault** (1.42) occurred until the transition to a **safe state** (1.102) is achieved as defined in the **warning and degradation concept** (1.140)

1.35

emergency operation interval

specified time-span that **emergency operation** (1.34) is needed to support the **warning and degradation concept** (1.140)

NOTE Emergency operation is part of the **warning and degradation concept** (1.140).

1.36

error

discrepancy between a computed, observed or measured value or condition, and the true, specified or theoretically correct value or condition

NOTE 1 An error can arise as a result of unforeseen operating conditions or due to a **fault** (1.42) within the **system** (1.129), subsystem or **component** (1.15) being considered.

NOTE 2 A fault can manifest itself as an error within the considered **element** (1.32) and the error can ultimately cause a **failure** (1.39).

1.37

exposure

state of being in an **operational situation** (1.83) that can be **hazardous** (1.57) if coincident with the **failure mode** (1.40) under analysis

1.38

external measure

measure that is separate and distinct from the **item** (1.69) which reduces or mitigates the **risks** (1.99) resulting from the item

1.39**failure**

termination of the ability of an **element** (1.32), to perform a function as required

NOTE Incorrect specification is a source of failure.

1.40**failure mode**

manner in which an **element** (1.32) or an **item** (1.69) fails

1.41**failure rate**

probability density of **failure** (1.39) divided by probability of survival for a hardware **element** (1.32)

NOTE The failure rate is assumed to be constant and is generally denoted as " λ ".

1.42**fault**

abnormal condition that can cause an **element** (1.32) or an **item** (1.69) to fail

NOTE 1 Permanent, intermittent and **transient faults** (1.134) (especially soft-errors) are considered.

NOTE 2 An intermittent fault occurs time and time again, then disappears. This type of fault can occur when a **component** (1.15) is on the verge of breaking down or, for example, due to a glitch in a switch. Some **systematic faults** (1.131) (e.g. timing marginalities) could lead to intermittent faults.

1.43**fault model**

representation of **failure modes** (1.40) resulting from **faults** (1.42)

NOTE Fault models are generally based on field experience or reliability handbooks.

1.44**fault reaction time**

time-span from the detection of a **fault** (1.42) to reaching the **safe state** (1.102)

See Figure 4.

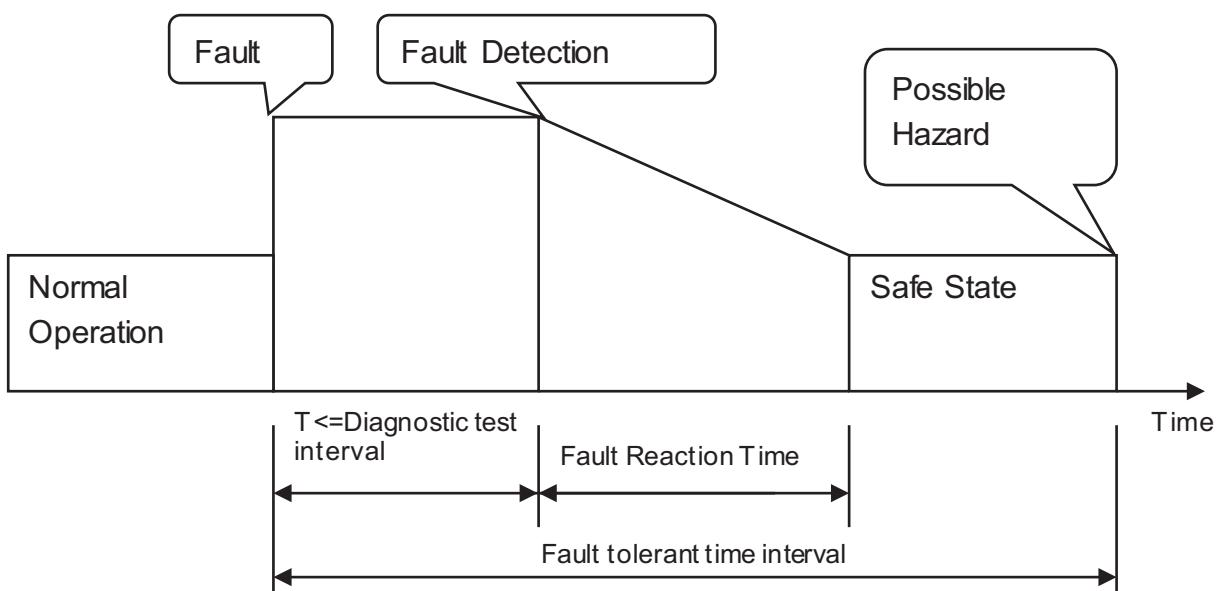


Figure 4 — Fault reaction time and fault tolerant time interval

1.45

fault tolerant time interval

time-span in which a **fault** (1.42) or faults can be present in a **system** (1.129) before a **hazardous** (1.57) event occurs

1.46

field data

data obtained from the use of an **item** (1.69) or **element** (1.32) including cumulative operating hours, all **failures** (1.39) and in-service anomalies

NOTE Field data normally comes from customer use.

1.47

formal notation

description technique that has both its syntax and semantics completely defined

EXAMPLE Z notation (Zed); NuSMV (symbolic model checker); Prototype Verification System (PVS); Vienna Development Method (VDM).

1.48

formal verification

method used to prove the correctness of a **system** (1.129) against the specification in **formal notation** (1.47) of its required behaviour

1.49

freedom from interference

absence of **cascading failures** (1.13) between two or more **elements** (1.32) that could lead to the violation of a safety requirement

EXAMPLE 1 Element 1 is free of interference from element 2 if no **failure** (1.39) of element 2 can cause element 1 to fail.

EXAMPLE 2 Element 3 interferes with element 4 if there exists a failure of element 3 that causes element 4 to fail.

1.50

functional concept

specification of the intended functions and their interactions necessary to achieve the desired behaviour

NOTE The functional concept is developed during the concept **phase** (1.89).

1.51

functional safety

absence of **unreasonable risk** (1.136) due to **hazards** (1.57) caused by **malfunctioning behaviour** (1.73) of **E/E systems** (1.31)

1.52

functional safety concept

specification of the **functional safety requirements** (1.53), with associated information, their **allocation** (1.1) to architectural **elements** (1.32), and their interaction necessary to achieve the **safety goals** (1.108)

1.53

functional safety requirement

specification of implementation-independent **safety** (1.103) behaviour, or implementation-independent **safety measure** (1.110), including its safety-related attributes

NOTE 1 A functional safety requirement can be a safety requirement implemented by a safety-related **E/E system** (1.31), or by a safety-related **system** (1.129) of **other technologies** (1.84), in order to achieve or maintain a **safe state** (1.102) for the **item** (1.69) taking into account a determined **hazardous event** (1.59).

NOTE 2 The functional safety requirements might be specified independently of the technology used in the concept **phase** (1.89), of product development.

NOTE 3 Safety-related attributes include information about **ASIL** (1.6).

1.54**hardware architectural metrics**

metrics for the **assessment** (1.4) of the effectiveness of the hardware **architecture** (1.3) with respect to **safety** (1.103)

NOTE The **single-point fault** (1.122) metric and the **latent fault** (1.71) metric are the hardware architectural metrics.

1.55**hardware part**

hardware which cannot be subdivided

1.56**harm**

physical injury or damage to the health of persons

1.57**hazard**

potential source of **harm** (1.56) caused by **malfunctioning behaviour** (1.73) of the **item** (1.69)

NOTE This definition is restricted to the scope of ISO 26262; a more general definition is potential source of harm.

1.58**hazard analysis and risk assessment**

method to identify and categorize **hazardous events** (1.59) of **items** (1.69) and to specify **safety goals** (1.108) and **ASILs** (1.6) related to the prevention or mitigation of the associated hazards in order to avoid **unreasonable risk** (1.136)

1.59**hazardous event**

combination of a **hazard** (1.57) and an **operational situation** (1.83)

1.60**homogeneous redundancy**

multiple but identical implementations of a requirement

1.61**independence**

absence of **dependent failures** (1.22) between two or more **elements** (1.32) that could lead to the violation of a safety requirement, or organizational separation of the parties performing an action

NOTE By definition, **ASIL decomposition** (1.7) or **confirmation measures** (1.17) include requirements on independence.

1.62**independent failures**

failures (1.39) whose probability of simultaneous or successive occurrence can be expressed as the simple product of their unconditional probabilities

1.63**informal notation**

description technique that does not have its syntax completely defined

EXAMPLE Description in figure or diagram.

NOTE An incomplete syntax definition implies that the semantics are also not completely defined.

1.64**informal verification**

verification (1.137) methods not considered as semi-formal or **formal verification** (1.48) techniques

EXAMPLE Design **review** (1.98); model review.

1.65

inheritance

passing attributes of requirements in an unchanged manner to the next level of detail during the development process

1.66

initial ASIL

ASIL (1.6) resulting from the hazard analysis or the ASIL resulting from a preceding **ASIL decomposition** (1.7)

NOTE The initial ASIL is the starting point for **ASIL decomposition** (1.7) or further ASIL decomposition.

1.67

inspection

examination of work products, following a formal procedure, in order to detect anomalies

NOTE 1 Inspection is a means of **verification** (1.137).

NOTE 2 Inspection differs from **testing** (1.134) in that it does not normally involve the operation of the associated **item** (1.69) or **element** (1.32).

NOTE 3 Any anomalies that are detected are usually addressed by rework, followed by re-inspection of the reworked products.

NOTE 4 A formal procedure normally includes a previously defined procedure, checklist, moderator and **review** (1.98) of the results.

1.68

intended functionality

behaviour specified for an **item** (1.69), **system** (1.129), or **element** (1.32) excluding **safety mechanisms** (1.111)

1.69

item

system (1.129) or array of systems to implement a function at the vehicle level, to which ISO 26262 is applied

1.70

item development

complete process of implementing an **item** (1.69)

1.71

latent fault

multiple-point fault (1.77) whose presence is not detected by a **safety mechanism** (1.111) nor perceived by the driver within the **multiple-point fault detection interval** (1.78)

1.72

lifecycle

entirety of **phases** (1.89) from concept through decommissioning of the **item** (1.69)

1.73

malfunctioning behaviour

failure (1.39) or unintended behaviour of an **item** (1.69) with respect to its design intent

1.74

model-based development

development that uses models to describe the functional behaviour of the **elements** (1.32) to be developed

NOTE Depending on the level of abstraction used for such a model, the model can be used for simulation or code generation or both.

1.75**modification**

authorized alteration of an **item** (1.69)

NOTE 1 Modification is used in ISO 26262 with respect to re-use for **lifecycle** (1.72) tailoring.

NOTE 2 A change is applied during the lifecycle of an item, while a modification is applied to create a new item from an existing item.

1.76**multiple-point failure**

failure (1.39), resulting from the combination of several independent **faults** (1.42), which leads directly to the violation of a **safety goal** (1.108)

NOTE For a multiple-point failure to directly violate a safety goal, the presence of all independent faults is necessary, i.e. the violation of a safety goal due to a combination of a **residual fault** (1.96) with other independent faults is not considered a multiple-point failure.

1.77**multiple-point fault**

individual **fault** (1.42) that, in combination with other independent faults, leads to a **multiple-point failure** (1.76)

NOTE A multiple-point fault can only be recognized after the identification of multiple-point failure, e.g. from cut set analysis of a fault tree.

1.78**multiple-point fault detection interval**

time span to **detect multiple-point fault** (1.77) before it can contribute to a **multiple-point failure** (1.76)

See Figure 4.

1.79**new development**

process of creating an **item** (1.69) having previously unspecified functionality, a novel implementation of an existing functionality, or both

1.80**non-functional hazard**

hazard (1.57) that arises due to factors other than incorrect functioning of the **E/E system** (1.31), safety-related **systems** (1.129) of **other technologies** (1.84), or **external measures** (1.38)

1.81**operating mode**

perceivable functional state of an **item** (1.69) or **element** (1.32)

EXAMPLE **System** (1.129) off; system active; system passive; degraded operation; **emergency operation** (1.34).

1.82**operating time**

cumulative time that an **item** (1.69) or **element** (1.32) is functioning

1.83**operational situation**

scenario that can occur during a vehicle's life

EXAMPLE Driving; parking; maintenance.

1.84

other technology

technology different from E/E technologies within the scope of ISO 26262

EXAMPLE Mechanical technology; hydraulic technology.

NOTE Other technologies can either be considered in the specification of the **functional safety concept** (1.52) (see ISO 26262-3:2011, Clause 8 and Figure 2), during the **allocation** (1.1) of safety requirements (see ISO 26262-3 and ISO 26262-4), or as an **external measure** (1.38).

1.85

partitioning

separation of functions or **elements** (1.32) to achieve a design

NOTE Partitioning can be used for **fault** (1.42) containment to avoid **cascading failures** (1.13). To achieve **freedom from interference** (1.49) between partitioned design elements, additional non-functional requirements can be introduced.

1.86

passenger car

vehicle designed and constructed primarily for the carriage of persons and their luggage, their goods, or both, having not more than a seating capacity of eight, in addition to the driver, and without space for standing passengers

1.87

perceived fault

fault (1.42) whose presence is deduced by the driver within a prescribed time interval

EXAMPLE The fault can be directly perceived through obvious limitation of **system** (1.129) behaviour or performance.

1.88

permanent fault

fault (1.42) that occurs and stays until removed or repaired

NOTE Direct current (d.c.) faults, e.g. stuck-at and bridging faults, are permanent faults. **Systematic faults** (1.131) manifest themselves mainly as permanent faults.

1.89

phase

stage in the safety **lifecycle** (1.72) that is specified in a distinct part of ISO 26262

NOTE The phases in ISO 26262 are specified in distinct parts, i.e. ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7 specify, respectively, the phases of:

- concept,
- product development at the system level,
- product development at the hardware level,
- product development at the software level, and
- production and operation.

1.90

proven in use argument

evidence, based on analysis of **field data** (1.46) resulting from use of a **candidate** (1.12), that the probability of any **failure** (1.39) of this candidate that could impair a **safety goal** (1.108) of an **item** (1.69) that uses it meets the requirements for the applicable **ASIL** (1.6)

1.91

proven in use credit

substitution of a given set of **lifecycle** (1.72) **subphases** (1.128) with corresponding work products by a **proven in use argument** (1.90)

1.92**random hardware failure**

failure (1.39) that can occur unpredictably during the lifetime of a hardware **element** (1.32) and that follows a probability distribution

NOTE Random hardware **failure rates** (1.41) can be predicted with reasonable accuracy.

1.93**reasonably foreseeable event**

event that is technically possible and has a credible or measurable rate of occurrence

1.94**redundancy**

existence of means in addition to the means that would be sufficient for an **element** (1.32) to perform a required function or to represent information

NOTE Redundancy is used in ISO 26262 with respect to achieving a **safety goal** (1.108) or a specified safety requirement, or to representing safety-related information.

EXAMPLE 1 Duplicated functional **components** (1.15) can be an instance of redundancy for the purpose of increasing **availability** (1.8) or allowing **fault** (1.42) detection.

EXAMPLE 2 The addition of parity bits to data representing safety-related information provides redundancy for the purpose of allowing fault detection.

1.95**regression strategy**

strategy to verify that an implemented change did not affect the unchanged, existing and previously verified parts or properties of an **item** (1.69) or an **element** (1.32)

1.96**residual fault**

portion of a **fault** (1.42) that by itself leads to the violation of a **safety goal** (1.108), occurring in a hardware **element** (1.32), where that portion of the fault is not covered by **safety mechanisms** (1.111)

NOTE This presumes that the hardware element has safety mechanism coverage for only a portion of its faults.

EXAMPLE If low (60 %) coverage is claimed for a **failure mode** (1.40), the other 40 % of that same failure mode is the residual fault.

1.97**residual risk**

risk (1.99) remaining after the deployment of **safety measures** (1.110)

1.98**review**

examination of a work product, for achievement of the intended work product goal, according to the purpose of the review

NOTE Reviews can be supported by checklists.

1.99**risk**

combination of the probability of occurrence of **harm** (1.56) and the **severity** (1.120) of that harm

1.100**robust design**

design that has the ability to function correctly in the presence of invalid inputs or stressful environmental conditions

NOTE Robustness can be understood as follows:

- for software, robustness is the ability to respond to abnormal inputs and conditions;
- for hardware, robustness is the ability to be immune to environmental stress and stable over the service life within design limits;
- in the context of ISO 26262, robustness is the ability to provide safe behaviour at boundaries.

1.101

safe fault

fault (1.42) whose occurrence will not significantly increase the probability of violation of a **safety goal** (1.108)

NOTE 1 As shown in ISO 26262-5:2011, Annex B, both non-safety and **safety-related elements** (1.113) can have safe faults.

NOTE 2 **Single-point faults** (1.122), **residual faults** (1.96) and dual-point faults do not constitute safe faults.

NOTE 3 Unless shown relevant in the safety concept, **multiple-point faults** (1.77) with higher order than 2 can be considered as safe faults.

1.102

safe state

operating mode (1.81) of an **item** (1.69) without an unreasonable level of **risk** (1.99)

EXAMPLE Intended operating mode; degraded operating mode; switched-off mode.

1.103

safety

absence of **unreasonable risk** (1.136)

1.104

safety activity

activity performed in one or more **subphases** (1.128) of the **safety lifecycle** (1.72)

1.105

safety architecture

set of **elements** (1.32) and their interaction to fulfil the safety requirements

1.106

safety case

argument that the safety requirements for an **item** (1.69) are complete and satisfied by evidence compiled from work products of the safety activities during development

NOTE Safety case can be extended to cover **safety** (1.103) issues beyond the scope of ISO 26262.

1.107

safety culture

policy and strategy used within an organization to support the development, production and operation of safety-related systems (1.129)

NOTE See ISO 26262-2:2011, Annex B.

1.108

safety goal

top-level safety requirement as a result of the **hazard analysis and risk assessment** (1.58)

NOTE One safety goal can be related to several **hazards** (1.57), and several safety goals can be related to a single hazard.

1.109**safety manager**

role filled by the person responsible for the **functional safety** (1.51) management during the **item** (1.69) development

1.110**safety measure**

activity or technical solution to avoid or control **systematic failures** (1.130) and to detect **random hardware failures** (1.92) or control random hardware failures, or mitigate their harmful effects

NOTE 1 Examples of safety measures are FMEA and software without the use of global variables.

NOTE 2 Safety measures include **safety mechanisms** (1.111).

1.111**safety mechanism**

technical solution implemented by E/E functions or **elements** (1.32), or by **other technologies** (1.84), to detect **faults** (1.42) or control **failures** (1.39) in order to achieve or maintain a **safe state** (1.102)

NOTE 1 Safety mechanisms are implemented within the **item** (1.69) to prevent faults from leading to **single-point failures** (1.121) or to reduce residual failures and to prevent faults from being latent.

NOTE 2 The safety mechanism is either

- a) able to transition to, or maintain, the item in a safe state, or
- b) able to alert the driver such that the driver is expected to control the effect of the **failure** (1.39),

as defined in the **functional safety concept** (1.52).

1.112**safety plan**

plan to manage and guide the execution of the **safety activities** (1.104) of a project including dates, milestones, tasks, deliverables, responsibilities and resources

1.113**safety-related element**

element (1.32) that has the potential to contribute to the violation of or achievement of a **safety goal** (1.108)

NOTE Fail-safe elements are considered safety-related if they can contribute to at least one safety goal.

1.114**safety-related function**

function that has the potential to contribute to the violation of a **safety goal** (1.108)

1.115**safety-related special characteristic**

characteristic of an **item** (1.69) or an **element** (1.32), or else their production process, for which reasonably foreseeable deviation could impact, contribute to, or cause any potential reduction of **functional safety** (1.51)

NOTE 1 Term special characteristics are defined in ISO/TS 16949.

NOTE 2 Safety-related special characteristics are derived during the development **phase** (1.89) of the item or the elements.

EXAMPLE Temperature range; expiration date; fastening torque; production tolerance; configuration.

1.116**safety validation**

assurance, based on examination and tests, that the **safety goals** (1.108) are sufficient and have been achieved

NOTE ISO 26262-4 provides suitable methods for validation.

1.117

semi-formal notation

description technique whose syntax is completely defined but whose semantics definition can be incomplete

EXAMPLE System Analysis and Design Techniques (SADT); Unified Modeling Language (UML).

1.118

semi-formal verification

verification (1.137) that is based on a description given in **semi-formal notation** (1.117)

EXAMPLE Use of test vectors generated from a semi-formal model to test that the **system** (1.129) behaviour matches the model.

1.119

service note

documentation of **safety** (1.103) information to be considered when performing maintenance procedures for the **item** (1.69)

EXAMPLE **Safety-related special characteristic** (1.115); safety operation that can be required.

1.120

severity

estimate of the extent of **harm** (1.56) to one or more individuals that can occur in a potentially **hazardous** (1.57) situation

NOTE The parameter "S" in **hazard analysis and risk assessment** (1.58) represents the potential severity of harm.

1.121

single-point failure

failure (1.39) that results from a **single-point fault** (1.122) and that leads directly to the violation of a **safety goal** (1.108)

NOTE 1 A single-point failure is equivalent to a residual failure for an **element** (1.32) with 0 % **diagnostic coverage** (1.25).

NOTE 2 If at least one **safety mechanism** (1.111) is defined for an HW element (e.g. a watchdog for a microcontroller), then no **faults** (1.42) of the considered hardware element are single-point faults.

1.122

single-point fault

fault (1.42) in an **element** (1.32) that is not covered by a **safety mechanism** (1.111) and that leads directly to the violation of a **safety goal** (1.108)

NOTE See also **single-point failure** (1.121).

1.123

software component

one or more **software units** (1.125)

1.124

software tool

computer program used in the development of an **item** (1.69) or **element** (1.32)

1.125

software unit

atomic level **software component** (1.123) of the software **architecture** (1.3) that can be subjected to stand-alone **testing** (1.134)

1.126**special-purpose vehicle**

vehicle intended to perform a function that requires special body arrangements, equipment or both

EXAMPLE Motor caravan; armoured vehicle; ambulance; hearse; trailer caravan; mobile crane.

NOTE ECE TRANS/WP.29/78/Rev.1/Amend.2 provides definitions for special-purpose vehicles.

1.127**statement coverage**

percentage of statements within the software that have been executed

1.128**subphase**

subdivision of a stage in the safety **lifecycle** (1.72) that is specified in a distinct clause of ISO 26262

EXAMPLE **Hazard analysis and risk assessment** (1.58) is a subphase of the safety lifecycle specified in ISO 26262-3:2011, Clause 7.

1.129**system**

set of **elements** (1.32) that relates at least a sensor, a controller and an actuator with one another

NOTE 1 The related sensor or actuator can be included in the system, or can be external to the system.

NOTE 2 An element of a system can also be another system.

1.130**systematic failure**

failure (1.39), related in a deterministic way to a certain cause, that can only be eliminated by a change of the design or of the manufacturing process, operational procedures, documentation or other relevant factors

1.131**systematic fault**

fault (1.42) whose **failure** (1.39) is manifested in a deterministic way that can only be prevented by applying process or design measures

1.132**technical safety concept**

specification of the **technical safety requirements** (1.133) and their **allocation** (1.1) to **system** (1.129) **elements** (1.32) for implementation by the system design

1.133**technical safety requirement**

requirement derived for implementation of associated **functional safety requirements** (1.53)

NOTE The derived requirement includes requirements for mitigation.

1.134**testing**

process of planning, preparing, and operating or exercising an **item** (1.69) or an **element** (1.32) to verify that it satisfies specified requirements, to detect **anomalies** (1.2), and to create confidence in its behaviour

1.135**transient fault**

fault (1.42) that occurs once and subsequently disappears

NOTE Transient faults can appear due to electromagnetic interference, which can lead to bit-flips. Soft errors such as Single Event Upset (SEU) and Single Event Transient (SET) are transient faults.

1.136

unreasonable risk

risk (1.99) judged to be unacceptable in a certain context according to valid societal moral concepts

1.137

verification

determination of completeness and correct specification or implementation of requirements from a **phase** (1.89) or **subphase** (1.128)

1.138

verification review

verification (1.137) activity to ensure that the result of a development activity fulfils the project requirements, or technical requirements, or both

NOTE 1 Individual requirements on verification reviews are given in specific clauses of individual parts of ISO 26262.

NOTE 2 The goal of verification reviews is technical correctness and completeness of the **item** (1.69) or **element** (1.32) with respect to use cases and **failure modes** (1.40).

EXAMPLE Technical **review** (1.98); **walk-through** (1.139); **inspection** (1.67).

1.139

walk-through

systematic examination of **work products** (1.142) in order to detect anomalies

NOTE 1 Walk-through is a means of **verification** (1.137).

NOTE 2 Walk-through differs from **testing** (1.134) in that it does not normally involve the operation of the associated **item** (1.69) or **element** (1.32).

NOTE 3 Any anomalies that are detected are usually addressed by rework, followed by a walk-through of the reworked work products.

EXAMPLE During a walk-through, the developer explains the work product step-by-step to one or more assessors. The objective is to create a common understanding of the work product and to identify any anomalies within the work product. Both **inspections** (1.67) and walk-throughs are types of peer **review** (1.98), where a walk-through is a less stringent form of peer review than an inspection.

1.140

warning and degradation concept

specification of how to alert the driver of potentially reduced functionality and of how to provide this reduced functionality to reach a **safe state** (1.102)

1.141

well-trusted

previously used without known **safety** (1.103) **anomalies** (1.2)

EXAMPLE Well-trusted design principle; well-trusted tool; well-trusted hardware **component** (1.15).

1.142

work product

result of one or more associated requirements of ISO 26262

NOTE A reference can be an independent document containing the complete information of a work product or a list of references to the complete information of a work product.

2 Abbreviated terms

ACC Adaptive Cruise Control

AEC Automotive Electronics Council

AIS	Abbreviated Injury Scale
ASIC	Application-Specific Integrated Circuit
ASIL	Automotive Safety Integrity Level (see definition 1.6)
BIST	Built-In Self-Test
CAN	Controller Area Network
CCF	Common Cause Failure (see definition 1.14)
COTS	Commercial Off The Shelf
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DC	Diagnostic Coverage (see definition 1.25)
d.c.	Direct Current
DIA	Development Interface Agreement (see definition 1.24)
DSC	Dynamic Stability Control
ECU	Electronic Control Unit
EDC	Error Detection and Correction
E/E system	Electrical and/or Electronic system (see definition 1.31)
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
ESD	Electrostatic Discharge
ESC	Electronic Stability Control
ETA	Event Tree Analysis
FPGA	Field Programmable Gate Array
FIT	Failures In Time
FMEA	Failure Mode and Effects Analysis
FTA	Fault Tree Analysis
HAZOP	HAZard and Operability analysis
HSI	Hardware-Software Interface
HW	Hardware
H&R	Hazard analysis and Risk assessment (see definition 1.58)
IC	Integrated Circuit
I/O	Input – Output
MC/DC	Modified Condition/Decision Coverage
MMU	Memory Management Unit
MPU	Memory Protection Unit

MUX	MULTipleXer
OS	Operating System
PLD	Programmable Logic Device
PMHF	Probabilistic Metric for random Hardware Failures
QM	Quality Management
RAM	Random Access Memory
ROM	Read Only Memory
RFQ	Request For Quotation
SIL	Safety Integrity Level
SOP	Start Of Production
SRS	System Requirements Specification
SW	Software
UML	Unified Modeling Language
V&V	Verification and Validation
XML	eXtensible Markup Language

Bibliography

- [1] ISO 3779, *Road vehicles — Vehicle identification number (VIN) — Content and structure*
- [2] ISO/TS 16949, *Quality management systems — Particular requirements for the application of ISO 9001:2008 for automotive production and relevant service part organizations*
- [3] ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*
- [4] ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*
- [5] ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*
- [6] ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*
- [7] ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*
- [8] ISO 26262-7:2011, *Road vehicles — Functional safety — Part 7: Production and operation*
- [9] ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*
- [10] ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- [11] ISO 26262-10, *Road vehicles — Functional safety — Part 10: Guideline on ISO 26262*
- [12] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*
- [13] ECE TRANS/WP.29/78/Rev.1/Amend.2, *Consolidated Resolution on the construction of vehicles (R.E.3)*

Alphabetical index

A

allocation 1.1
anomaly 1.2
architecture 1.3
ASIL 1.6
ASIL decomposition 1.7
assessment 1.4
audit 1.5
Automotive Safety Integrity Level 1.6
availability 1.8

B

baseline 1.9
branch coverage 1.10

C

calibration data 1.11
candidate 1.12
cascading failure 1.13
CCF 1.14
common cause failure 1.14
component 1.15
configuration data 1.16
confirmation measure 1.17
confirmation review 1.18
controllability 1.19

D

dedicated measure 1.20
degradation 1.21
dependent failures 1.22
detected fault 1.23
development interface agreement 1.24
DIA 1.24
diagnostic coverage 1.25
diagnostic test interval 1.26
distributed development 1.27
diversity 1.28
dual-point failure 1.29
dual-point fault 1.30

E

E/E system 1.31
electrical and/or electronic system 1.31
element 1.32
embedded software 1.33
emergency operation 1.34

emergency operation interval 1.35
error 1.36
exposure 1.37
external measure 1.38

F

failure 1.39
failure mode 1.40
failure rate 1.41
fault 1.42
fault model 1.43
fault reaction time 1.44
fault tolerant time interval 1.45
field data 1.46
formal notation 1.47
formal verification 1.48
freedom from interference 1.49
functional concept 1.50
functional safety 1.51
functional safety concept requirement 1.52
functional safety requirement 1.53

H

hardware architectural metrics 1.54
hardware part 1.55
harm 1.56
hazard 1.57
hazard analysis and risk assessment 1.58
hazardous event 1.59
homogeneous redundancy 1.60

I

independence 1.61
independent failures 1.62
informal notation 1.63
informal verification 1.64
inheritance 1.65
initial ASIL 1.66
inspection 1.67
intended functionality 1.68
item 1.69
item development 1.70

L

latent fault 1.71
lifecycle 1.72

M

malfunctioning behaviour 1.73
model-based development 1.74
modification 1.75
multiple-point failure 1.76
multiple-point fault 1.77
multiple-point fault detection interval 1.78

N

new development 1.79
non-functional hazard 1.80

O

operating mode 1.81
operating time 1.82
operational situation 1.83
other technology 1.84

P

partitioning 1.85
passenger car 1.86
perceived fault 1.87
permanent fault 1.88
phase 1.89
proven in use argument 1.90
proven in use credit 1.91

R

random hardware failure 1.92
reasonably foreseeable event 1.93
redundancy 1.94
regression strategy 1.95
residual fault 1.96
residual risk 1.97
review 1.98
risk 1.99
robust design 1.100

S

safe fault 1.101
safe state 1.102
safety 1.103
safety activity 1.104
safety architecture 1.105
safety case 1.106
safety culture 1.107
safety goal 1.108

safety manager 1.109
safety measure 1.110
safety mechanism 1.111
safety plan 1.112
safety validation 1.116
safety-related element 1.113
safety-related function 1.114
safety-related special
 characteristic 1.115
semi-formal notation 1.117
semi-formal verification 1.118
service note 1.119
severity 1.120
single-point failure 1.121
single-point fault 1.122
software component 1.123
software tool 1.124
software unit 1.125
special-purpose vehicle 1.126
statement coverage 1.127
subphase 1.128
system 1.129
systematic failure 1.130
systematic fault 1.131

T

technical safety concept 1.132
technical safety
 requirement 1.133
testing 1.134
transient fault 1.135

U

unreasonable risk 1.136

V

verification 1.137
verification review 1.138

W

walk-through 1.139
warning and degradation
 concept 1.140
well-trusted 1.141
work product 1.142

This page is intentionally blank.

This page is intentionally blank.

ICS 01.040.43; 43.040.10

Price based on 23 pages

INTERNATIONAL
STANDARD

ISO
26262-2

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 2:
Management of functional safety**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 2: Gestion de la sécurité fonctionnelle*



Reference number
ISO 26262-2:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction.....	v
1 Scope.....	1
2 Normative references.....	1
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	2
4.3 ASIL-dependent requirements and recommendations	3
5 Overall safety management.....	3
5.1 Objective	3
5.2 General	3
5.3 Inputs to this clause.....	7
5.4 Requirements and recommendations	7
5.5 Work products	9
6 Safety management during the concept phase and the product development.....	9
6.1 Objectives	9
6.2 General	9
6.3 Inputs to this clause.....	10
6.4 Requirements and recommendations	10
6.5 Work products	17
7 Safety management after the item's release for production.....	17
7.1 Objective	17
7.2 General	17
7.3 Inputs to this clause.....	17
7.4 Requirements and recommendations	18
7.5 Work products	18
Annex A (informative) Overview of and workflow of functional safety management.....	19
Annex B (informative) Examples for evaluating a safety culture.....	20
Annex C (informative) Aim of the confirmation measures	21
Annex D (informative) Overview of the verification reviews	23
Annex E (informative) Example of a functional safety assessment agenda (for items that have an ASIL D safety goal).....	24
Bibliography.....	26

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-2 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

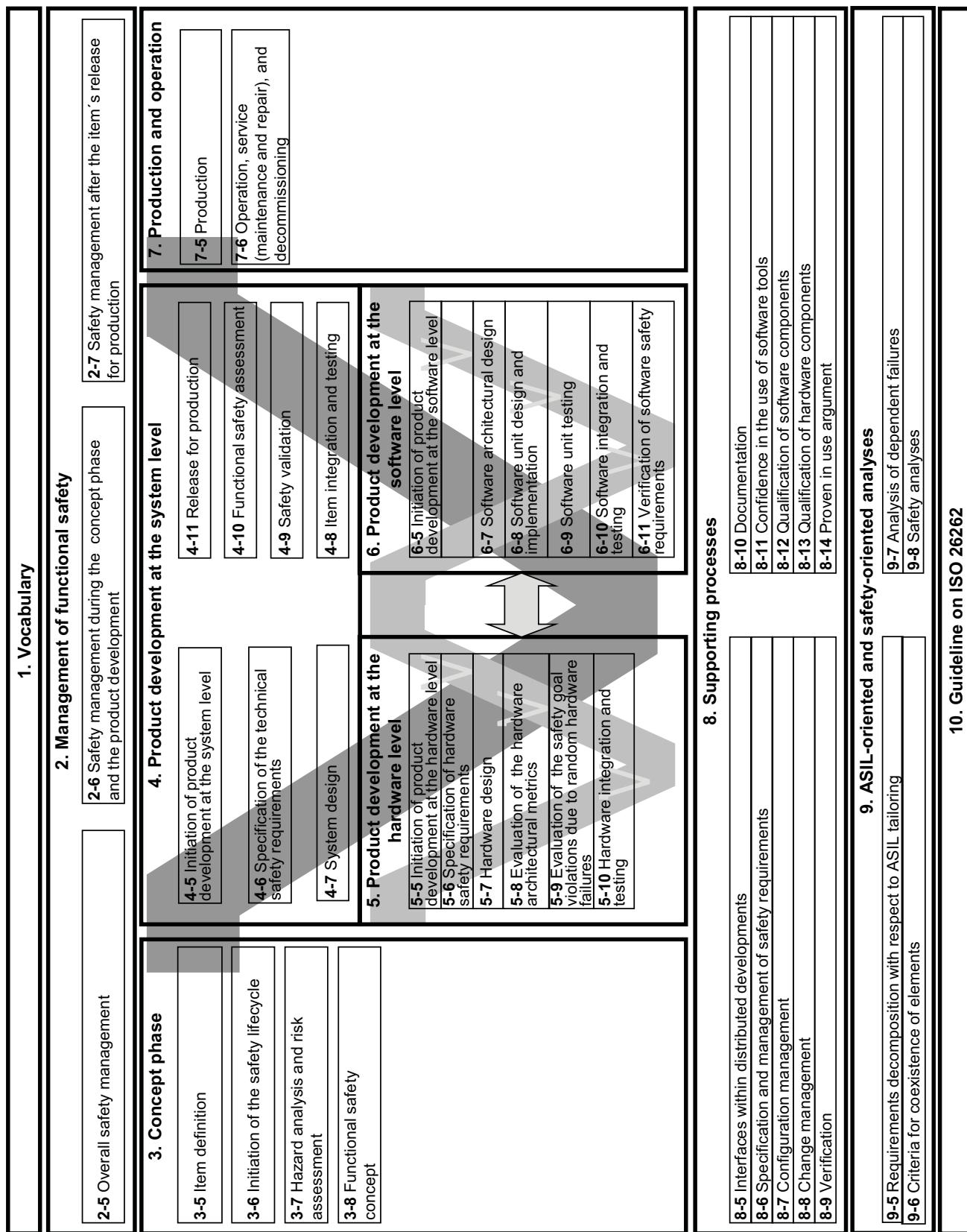


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 2: Management of functional safety

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for functional safety management for automotive applications, including the following:

- project-independent requirements with regard to the organizations involved (overall safety management), and
- project-specific requirements with regard to the management activities in the safety lifecycle (i.e. management during the concept phase and product development, and after the release for production).

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-7:2011, *Road vehicles — Functional safety — Part 7: Production and operation*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- a) tailoring of the safety activities in accordance with this part of ISO 26262 has been planned and shows that the requirement does not apply, or
- b) a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with this part of ISO 26262.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Overall safety management

5.1 Objective

The objective of this clause is to define the requirements for the organizations that are responsible for the safety lifecycle, or that perform safety activities in the safety lifecycle.

This clause serves as a prerequisite to the activities in the ISO 26262 safety lifecycle.

5.2 General

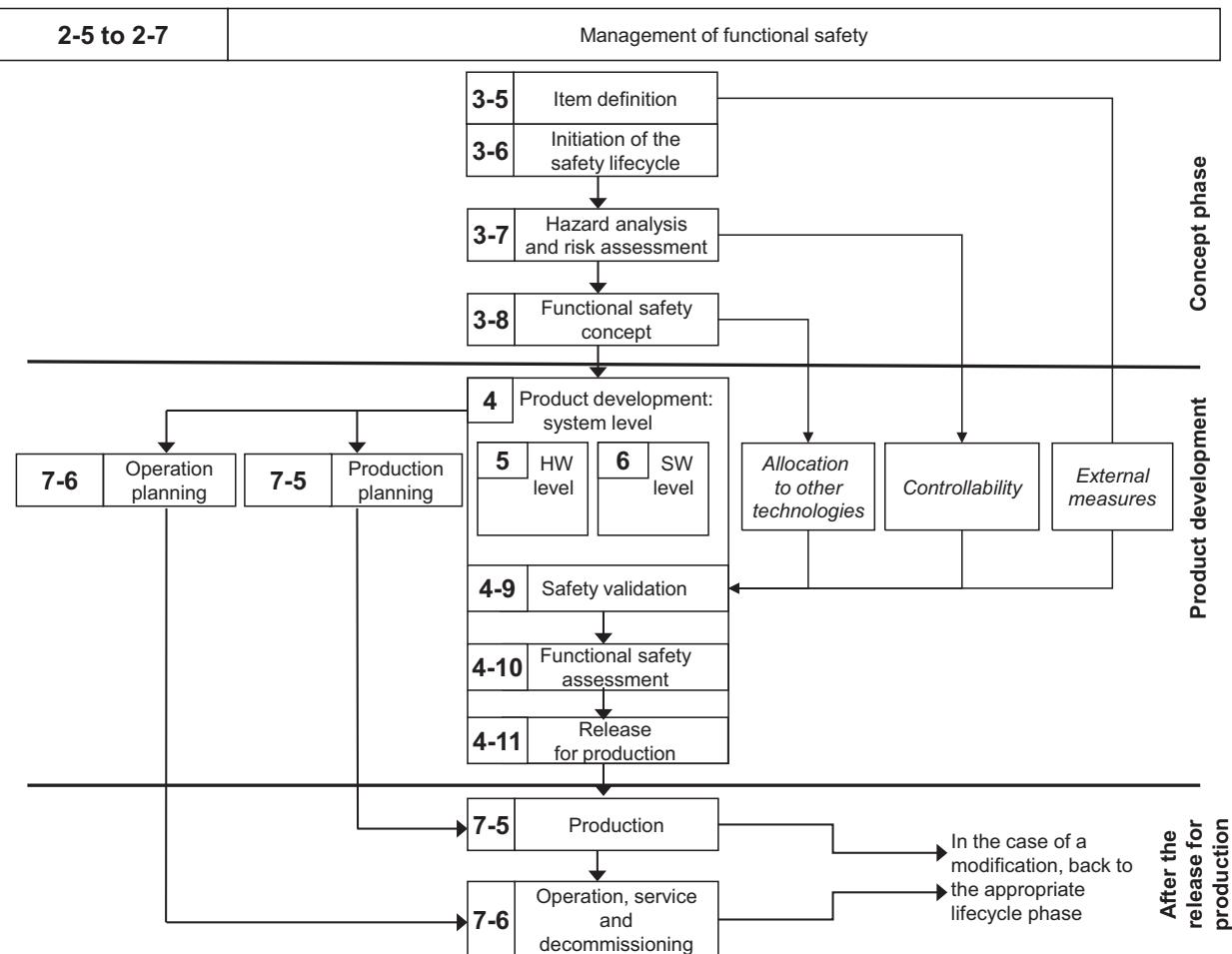
5.2.1 Overview of the safety lifecycle

The ISO 26262 safety lifecycle (see Figure 2) encompasses the principal safety activities during the concept phase, product development, production, operation, service and decommissioning. Planning, coordinating and documenting the safety activities of all phases of the safety lifecycle are key management tasks.

Figure 2 represents the reference safety lifecycle model. Tailoring of the safety lifecycle, including iterations of subphases, is allowed.

NOTE 1 The activities during the concept phase and the product development, and after the release for production are described in detail in ISO 26262-3 (concept phase), ISO 26262-4 (product development at the system level), ISO 26262-5 (product development at the hardware level), ISO 26262-6 (product development at the software level) and ISO 26262-7 (production and operation).

NOTE 2 Table A.1 provides an overview of the objectives, prerequisites and work products of the particular phases of the management of functional safety.



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: "m-n", where "m" represents the number of the part and "n" indicates the number of the clause, e.g. "3-6" represents Clause 6 of ISO 26262-3.

Figure 2 — Safety lifecycle

5.2.2 Explanatory remarks on the safety lifecycle

ISO 26262 specifies requirements with regard to specific phases and subphases of the safety lifecycle, but also includes requirements that apply to several, or all, phases of the safety lifecycle, such as the requirements for the management of functional safety.

The key management tasks are to plan, coordinate and track the activities related to functional safety. These management tasks apply to all phases of the safety lifecycle. The requirements for the management of functional safety are given in this part, which distinguishes:

- overall safety management (see this clause);
- safety management during the concept phase and the product development (see Clause 6);
- safety management after the item's release for production (see Clause 7).

The following descriptions explain the definitions of the different phases and subphases of the safety lifecycle, as well as other key concepts:

a) The subphase: item definition

The initiating task of the safety lifecycle is to develop a description of the item with regard to its functionality, interfaces, environmental conditions, legal requirements, known hazards, etc. The boundary of the item and its interfaces, as well as assumptions concerning other items, elements, systems and components are determined (see ISO 26262-3:2011, Clause 5).

b) The subphase: initiation of the safety lifecycle

Based on the item definition, the safety lifecycle is initiated by distinguishing between either a new development, or a modification of an existing item.

If an existing item is modified, the results of an impact analysis are used to tailor the safety lifecycle (see ISO 26262-3:2011, Clause 6).

c) The subphase: hazard analysis and risk assessment

After the initiation of the safety lifecycle, the hazard analysis and risk assessment is performed as given in ISO 26262-3:2011, Clause 7. First, the hazard analysis and risk assessment estimates the probability of exposure, the controllability and the severity of the hazardous events with regard to the item. Together, these parameters determine the ASILs of the hazardous events. Subsequently, the hazard analysis and risk assessment determines the safety goals for the item, with the safety goals being the top level safety requirements for the item. The ASILs determined for the hazardous events are assigned to the corresponding safety goals.

During the subsequent phases and subphases, detailed safety requirements are derived from the safety goals. These safety requirements inherit the ASIL of the corresponding safety goals.

d) The subphase: functional safety concept

Based on the safety goals, a functional safety concept (see ISO 26262-3:2011, Clause 8) is specified considering preliminary architectural assumptions. The functional safety concept is specified by functional safety requirements that are allocated to the elements of the item. The functional safety concept can also include other technologies or interfaces with external measures, provided that the expected behaviours thereof can be validated (see ISO 26262-4:2011, Clause 9). The implementation of other technologies is outside the scope of ISO 26262 and the implementation of the external measures is outside the scope of the item development.

e) The phase: product development at the system level

After having specified the functional safety concept, the item is developed from the system level perspective, as given in ISO 26262-4. The system development process is based on the concept of a V-model with the specification of the technical safety requirements, the system architecture, the system design and implementation on the left hand branch and the integration, verification, validation and the functional safety assessment on the right hand branch.

The hardware-software interface is specified in this phase.

Figure 1 provides an overview of the subphases of the product development at the system level.

The product development at the system level incorporates validation tasks for activities occurring within other safety lifecycle phases, including

- the validation of the aspects of the functional safety concept that are implemented by other technologies;

- the validation of the assumptions concerning the effectiveness and the performance of external measures; and
- the validation of the assumptions concerning human response, including controllability and operational tasks.

The release for production is the final subphase of the product development and provides the item's release for series production (see ISO 26262-4:2011, Clause 11).

f) The phase: product development at the hardware level

Based on the system design specification, the item is developed from the hardware level perspective (see ISO 26262-5). The hardware development process is based on the concept of a V-model with the specification of the hardware requirements and the hardware design and implementation on the left hand branch and the hardware integration and testing on the right hand branch.

Figure 1 provides an overview of the subphases of the product development at the hardware level.

g) The phase: product development at the software level

Based on the system design specification, the item is developed from the software level perspective (see ISO 26262-6). The software development process is based on the concept of a V-model with the specification of the software requirements and the software architectural design and implementation on the left hand branch, and the software integration and testing, and the verification of the software requirements on the right hand branch.

Figure 1 provides an overview of the subphases of the product development at the software level.

h) Production planning and operation planning

The planning for production and operation, and the specification of the associated requirements, starts during the product development at the system level (see ISO 26262-4). The requirements for production and operation are given in ISO 26262-7:2011, Clauses 5 and 6.

i) The phase: production and operation, service and decommissioning

This phase addresses the production processes relevant for the functional safety goals of the item, i.e. the safety-related special characteristics, and the development and management of instructions for the maintenance, repair and decommissioning of the item to ensure functional safety after the item's release for production (see ISO 26262-7:2011, Clauses 5 and 6).

j) Controllability

In the hazard analysis and risk assessment (see ISO 26262-3:2011, Clause 7), credit can be taken for the ability of the driver, or the other persons at risk, to control hazardous situations. The assumptions regarding the controllability in the hazard analysis and risk assessment and the functional and technical safety concept are validated during the safety validation (see Figure 2 and ISO 26262-4:2011, Clause 9).

NOTE The exposure and the severity are factors that depend on the scenario. The eventual controllability through human intervention is influenced by the design of the item and is therefore evaluated during the validation (see ISO 26262-4:2011, 9.4.3.2).

k) External measures

The external measures refer to the measures outside the item, as specified in the item definition (see Figure 2 and ISO 26262-3:2011, Clause 5), that reduce or mitigate the risks resulting from the item. External measures can include not only additional in-vehicle devices such as dynamic stability controllers or run-flat tyres, but also devices external to the vehicle, like crash barriers or tunnel fire-fighting systems.

The assumptions regarding the external measures in the item definition, the hazard analysis and risk assessment and the functional and technical safety concept are validated during the safety validation (see Figure 2 and ISO 26262-4:2011, Clause 9).

External measures can be considered in the hazard analysis and risk assessment. However, if credit is taken from an external measure in the hazard analysis and risk assessment, that external measure cannot be considered as a risk reduction in the functional safety concept.

ISO 26262 also applies to those external measures that are in the scope of ISO 26262.

I) Other technologies

Other technologies, e.g. mechanical and hydraulic technologies, are those different from electrical and/or electronic technologies that are in the scope of ISO 26262. These can be considered in the specification of the functional safety concept (see Figure 2 and ISO 26262-3:2011, Clause 8), during the allocation of safety requirements (see ISO 26262-3 and ISO 26262-4), or as an external measure.

NOTE If an implementation in another technology is specified as an external measure, then it can be useful to repeat the hazard analysis and risk assessment to consider the associated risk reduction, which could potentially result in a reduced ASIL of a corresponding safety goal.

5.3 Inputs to this clause

5.3.1 Prerequisites

None.

5.3.2 Further supporting information

The following information can be considered:

- existing evidence of a quality management system complying with a quality management standard, such as ISO/TS 16949, ISO 9001, or equivalent.

5.4 Requirements and recommendations

5.4.1 General

The organizations involved in the execution of the safety lifecycle shall comply with 5.4.2 to 5.4.5.

5.4.2 Safety culture

5.4.2.1 The organization shall create, foster, and sustain a safety culture that supports and encourages the effective achievement of functional safety.

EXAMPLE Examples for evaluating a safety culture are given in Annex B.

5.4.2.2 The organization shall institute, execute and maintain organization-specific rules and processes to comply with the requirements of ISO 26262.

NOTE Such organization-specific rules and processes can include the creation and maintenance of a generic safety plan and process description.

5.4.2.3 The organization shall institute, execute and maintain processes to ensure that identified functional safety anomalies are explicitly communicated to the applicable safety manager(s) and the other responsible persons.

EXAMPLE The safety manager of the customer and the safety manager of a supplier, the safety manager of the development of a related item.

5.4.2.4 The organization shall institute, execute and maintain a safety anomaly resolution process to ensure that the analysis, evaluation, resolution and disposition of functional safety anomalies are performed in a timely and effective manner.

NOTE The anomaly resolution process can include a root cause analysis that results in a corrective action for the future.

5.4.2.5 During the execution of the safety lifecycle, the organization shall perform the required functional safety activities, including the production and management of the associated documentation in accordance with ISO 26262-8:2011, Clause 10.

5.4.2.6 The organization shall provide the resources required for the achievement of functional safety.

NOTE Resources include human resources, tools, databases, and templates.

5.4.2.7 The organization shall institute, execute and maintain a continuous improvement process, based on:

- learning from the experiences gained during the execution of the safety lifecycle of other items, including field experience; and
- derived improvements for application on subsequent items.

5.4.2.8 The organization shall ensure that the persons performing or supporting the safety activities are given sufficient authority to fulfil their responsibilities.

5.4.3 Competence management

5.4.3.1 The organization shall ensure that the persons involved in the execution of the safety lifecycle have a sufficient level of skills, competences and qualifications corresponding to their responsibilities.

NOTE 1 One of the possible means to achieve a sufficient level of skills and competences in development is a training and qualification programme that considers the following knowledge areas:

- usual safety practices, concepts and designs;
- ISO 26262 and, if applicable, further safety standards;
- organization-specific rules for functional safety;
- functional safety processes instituted in the organization.

NOTE 2 To evaluate the skills, competences and qualifications to carry out activities to comply with ISO 26262, the experience from previous professional activities can be considered, e.g.

- domain knowledge of the item;
- expertise on the environment of the item;
- management experience.

5.4.4 Quality management during the safety lifecycle

5.4.4.1 The organizations involved in the execution of the safety lifecycle shall have an operational quality management system complying with a quality management standard, such as ISO/TS 16949, ISO 9001, or equivalent.

5.4.5 Project-independent tailoring of the safety lifecycle

5.4.5.1 The organization may tailor the safety lifecycle for application across item developments, i.e. apply a project-independent tailoring, but only if such a tailoring is limited to applying one or more of the following permissions:

- a) subphases, activities or tasks may be combined or split, or

NOTE Individual subphases can be combined if the method used makes it difficult to clearly distinguish between the individual subphases, e.g. computer-aided development tools can support activities of several subphases within one step.

- b) an activity or task may be performed in a different phase or subphase, or
- c) an activity or task may be performed in an added phase or subphase, or
- d) phases or subphases may be iterated.

5.5 Work products

5.5.1 Organization-specific rules and processes for functional safety, resulting from 5.4.2 and 5.4.5.

5.5.2 Evidence of competence, resulting from 5.4.3.

5.5.3 Evidence of quality management, resulting from 5.4.4.

6 Safety management during the concept phase and the product development

6.1 Objectives

The first objective of this clause is to define the safety management roles and responsibilities, regarding the concept phase and the development phases in the safety lifecycle (see Figure 1 and Figure 2).

The second objective of this clause is to define the requirements for the safety management during the concept phase and the development phases, including the planning and coordination of the safety activities, the progression of the safety lifecycle, the creation of the safety case, and the execution of the confirmation measures.

6.2 General

Safety management includes the responsibility to ensure that the confirmation measures are performed. Depending on the applicable ASIL, some confirmation measures require independence regarding resources, management and release authority (see 6.4.7).

Confirmation measures include confirmation reviews, functional safety audits and functional safety assessments:

- the confirmation reviews are intended to check the compliance of selected work products to the corresponding requirements of ISO 26262;
- a functional safety audit evaluates the implementation of the processes required for the functional safety activities;
- a functional safety assessment evaluates the functional safety achieved by the item.

In addition to the confirmation measures, verification reviews are performed. These reviews, which are required in other parts of ISO 26262, are intended to verify that the associated work products fulfil the project requirements, and the technical requirements with respect to use cases and failure modes.

Table 1 lists the required confirmation measures. Annex D lists the reviews concerning verification and refers to the applicable parts of ISO 26262.

Safety management includes the responsibility for the description and justification of any tailored safety activity (see 6.4.5).

6.3 Inputs to this clause

6.3.1 Prerequisites

The following information shall be available:

- organization-specific rules and processes for functional safety in accordance with 5.5.1;
- evidence of competence in accordance with 5.5.2;
- evidence of quality management in accordance with 5.5.3.

6.3.2 Further supporting information

If available, the following information can be considered:

- project plan (from external source);
- dependencies on other activities, including other safety activities.

6.4 Requirements and recommendations

6.4.1 General

The organizations involved in the execution of the safety lifecycle shall comply with 6.4.2 to 6.4.9 for items that have at least one safety goal with an ASIL A, B, C, or D, unless stated otherwise.

6.4.2 Roles and responsibilities in safety management

6.4.2.1 A project manager shall be appointed at the initiation of the item development.

6.4.2.2 The project manager shall be given the responsibility and the authority, in accordance with 5.4.2.8, to ensure that:

- a) the safety activities required to achieve functional safety are performed; and
- b) compliance with ISO 26262 is achieved.

6.4.2.3 The project manager shall verify that the organization has provided the required resources for the functional safety activities, in accordance with 5.4.2.6.

NOTE The estimation, determination and allocation of sufficient resources are generally performed in the planning phase.

6.4.2.4 The project manager shall ensure that the safety manager is appointed, in accordance with 5.4.3.

NOTE 1 The role of the safety manager can be fulfilled by the project manager.

NOTE 2 As the term “safety manager” is defined as a role (see ISO 26262-1), its assignment can be split between different persons in a matrix organization.

6.4.3 Planning and coordination of the safety activities

6.4.3.1 The safety manager shall be responsible for the planning and coordination of the functional safety activities in the development phases of the safety lifecycle, in accordance with 5.4.2.8.

NOTE 1 The safety manager can delegate tasks to persons that possess the required skills, competences and qualifications (see 5.4.3).

NOTE 2 The planning of the safety activities is included in the safety plan (see 6.4.3.5). Certain planned safety activities are further detailed in other work products of ISO 26262 (see 6.4.3.3).

NOTE 3 Depending on whether the item is a new development or a modification of an existing item (see ISO 26262-3:2011, Clause 6), the extent of the safety activities can vary, and the activities are planned accordingly.

6.4.3.2 The safety manager shall be responsible for maintaining the safety plan, and for monitoring the progress of the safety activities against the safety plan.

6.4.3.3 The responsibilities with regard to detailing and coordinating the safety activities in the plans listed below shall be assigned and communicated within the organization in accordance with 5.4.2.8 and 5.4.3:

- the item integration and testing plan in accordance with ISO 26262-4;
- the validation plan in accordance with ISO 26262-4;
- the software verification plan in accordance with ISO 26262-6; and
- the functional safety assessment plan in accordance with 6.4.9.

A person assigned with such a responsibility shall also be responsible for maintaining the respective plan and for monitoring the progress of these activities against the respective plan.

6.4.3.4 The safety plan shall either be

- a) referenced in the project plan, or
- b) included in the project plan, such that the safety activities are distinguishable.

NOTE The safety plan can incorporate cross-references to other information under configuration management (see ISO 26262-8:2011, Clause 7). Cross-references are generally preferable to the parallel description of activities in different work products, or in other documents that are under configuration management.

6.4.3.5 The safety plan shall include:

- a) the planning of the activities and procedures for achieving functional safety;
- b) the implementation of project-independent safety activities in accordance with Clause 5 into project-specific safety management;
- c) the definition of the tailored safety activities, in accordance with 6.4.5, if applicable;
- d) the planning of the hazard analysis and risk assessment in accordance with ISO 26262-3:2011, Clause 7;
- e) the planning of the development activities, including the development and implementation of the functional safety concept in accordance with ISO 26262-3:2011, Clause 8, the product development at the system level in accordance with ISO 26262-4, the product development at the hardware level in accordance with ISO 26262-5 and the product development at the software level in accordance with ISO 26262-6;
- f) the planning of the development interface agreement (DIA) in accordance with ISO 26262-8:2011, Clause 5, if applicable;

- g) the planning of the supporting processes, in accordance with ISO 26262-8;
- h) the planning of the verification activities in accordance with ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-8:2011, Clause 9; and the planning of the safety validation activities in accordance with ISO 26262-4:2011, Clause 9;

NOTE The verification activities are detailed in the item integration and testing plan (see ISO 26262-4) and the software verification plan (see ISO 26262-6). The validation activities are detailed in the validation plan (see ISO 26262-4). See also 6.4.3.3.

- i) the planning of the confirmation reviews, the initiation of the functional safety audit(s) and the initiation of the functional safety assessment in accordance with 6.4.7 to 6.4.9;

NOTE The level of independence given in 6.4.7, and the qualifications given in 5.4.3, of the persons that carry out the confirmation measures are specified in the safety plan.

- j) the planning of the analysis of dependent failures in accordance with ISO 26262-9:2011, Clause 7, if applicable, and the safety analyses in accordance with ISO 26262-9:2011, Clause 8;
- k) the provision of the proven in use arguments of the candidates in accordance with ISO 26262-8:2011, Clause 14, if applicable; and
- l) the provision of the confidence in the usage of software tools in accordance with ISO 26262-8:2011, Clause 11, if applicable.

6.4.3.6 The planning of a safety activity shall include describing:

- a) the objective;
- b) the dependencies on other activities or information;
- c) the resource responsible for performing the activity;
- d) the required resources for performing the activity;
- e) the starting point in time and duration; and
- f) the identification of the corresponding work product.

6.4.3.7 This requirement shall be complied with for items that have at least one safety goal with an ASIL B, C, or D: the safety plan in accordance with 6.4.3.1 to 6.4.3.6 shall be approved by an authorized person, who shall consider the confirmation review of the safety plan in accordance with 6.4.7.

6.4.3.8 Identified safety anomalies shall be reported to the safety manager, and the other responsible persons, in accordance with 5.4.2.3.

NOTE A change that results from the resolution of a safety anomaly is entered into the change management process (see ISO 26262-8:2011, Clause 8).

6.4.4 Progression of the safety lifecycle

6.4.4.1 The safety activities in a subsequent subphase of the safety lifecycle shall be started only when there is sufficient information from the pertinent subphases.

NOTE Information can be considered as sufficient if the lack of information does not cause an unacceptable risk to the achievement of the item's safety goals.

6.4.4.2 The work products required per the safety plan shall be subject to configuration management, change management and documentation, in accordance with ISO 26262-8:2011, Clauses 7, 8 and 10, respectively, no later than the time of entering the phase "product development at system level" (see Figure 1).

6.4.5 Tailoring of the safety activities

6.4.5.1 A safety activity with regard to a specific item development may be tailored i.e. omitted or performed in a different manner. If such a safety activity is tailored, then

- a) the tailoring shall be defined in the safety plan (see 6.4.3.5, c); and
- b) a rationale as to why the tailoring is adequate and sufficient to achieve functional safety shall be available.

NOTE 1 The rationale considers the ASILs of the corresponding requirements.

NOTE 2 The rationale for the tailoring is included in the safety plan and reviewed during the confirmation review of the safety plan (see 6.4.7) or during the functional safety assessment (see 6.4.9).

NOTE 3 This requirement applies to tailoring for application on a specific item. With regard to tailoring of the safety lifecycle for application across item developments within an organization, only 5.4.5 applies.

6.4.5.2 If a safety activity is tailored in accordance with 6.4.5.1 because of a modification of an existing item, then ISO 26262-3:2011, Clause 6 shall be complied with.

6.4.5.3 If a safety activity is tailored in accordance with 6.4.5.1 as a result of a proven in use argument, then ISO 26262-8:2011, Clause 14 shall be complied with.

6.4.5.4 If a safety activity is not applied in accordance with 6.4.5.1 as a result of an ASIL decomposition, then ISO 26262-9:2011, Clause 5 shall be complied with.

6.4.5.5 If a safety activity is tailored in accordance with 6.4.5.1 based on a rationale that considers the confidence in the usage of software tools, then ISO 26262-8:2011, Clause 11 shall be complied with.

6.4.5.6 If the safety activities are tailored in accordance with 6.4.5.1 because an element is developed separately from an item, then

- a) the development of the element developed separately from an item shall be based on a requirement specification that is derived from assumptions on an intended use and context, including its external interfaces; and
- b) the validity of the assumptions on the intended use and context of the element developed separately from an item shall be established.

NOTE ISO 26262 as a whole cannot be applied to an element developed separately from an item, because functional safety is not an element property (however, an element of an item can be identified as safety related). Functional safety is an item property which can be evaluated by means of a functional safety assessment.

EXAMPLE A microcontroller developed separately from an item.

6.4.6 Safety case

6.4.6.1 This requirement shall be complied with for items that have at least one safety goal with an ASIL (A), B, C or D: a safety case shall be developed in accordance with the safety plan.

6.4.6.2 The safety case should progressively compile the work products that are generated during the safety lifecycle.

6.4.7 Confirmation measures: types, independency and authority

6.4.7.1 The confirmation measures specified in Table 1 shall be performed, in accordance with the required level of independency, Table 2, 6.4.3.5 i), 6.4.8 and 6.4.9.

NOTE 1 The confirmation reviews are performed for those work products that are specified in Table 1 and required by the safety plan.

NOTE 2 A confirmation review includes the checking of correctness with respect to formality, contents, adequacy and completeness regarding the requirements of ISO 26262.

NOTE 3 Table 1 includes the confirmation measures. An overview of the verification reviews is given in Annex D.

NOTE 4 A report that is a result of a confirmation measure includes the name and revision number of the work products or process documents analysed (see ISO 26262-8:2011, 10.4.5).

NOTE 5 If the item changes subsequent to the completion of confirmation reviews or functional safety assessments, then these will be repeated or supplemented (see ISO 26262-8:2011, 8.4.5.2).

NOTE 6 The aim of each confirmation measure is given in Annex C.

NOTE 7 Confirmation measures such as confirmation reviews and functional safety audits can be merged and combined with the functional safety assessment to support the handling of comparable variants of an item.

Table 1 — Required confirmation measures, including the required level of independency

Confirmation measures	Degree of independency ^a applies to ASIL				Scope
	A	B	C	D	
Confirmation review of the hazard analysis and risk assessment of the item (see ISO 26262-3:2011, Clauses 5 and 7, and, if applicable, ISO 26262-8:2011, Clause 5) Independence with regard to the developers of the item, project management and the authors of the work product	I3	I3	I3	I3	The scope of this review shall include the correctness of the determined ASILs and quality management (QM) ratings of the identified hazardous events for the item, and a review of the safety goals
Confirmation review of the safety plan (see 6.5.1) Independence with regard to the developers of the item, project management and the authors of the work product	—	I1	I2	I3	Applies to the highest ASIL among the safety goals of the item
Confirmation review of the item integration and testing plan (see ISO 26262-4) Independence with regard to the developers of the item, project management and the authors of the work product	I0	I1	I2	I2	Applies to the highest ASIL among the safety goals of the item
Confirmation review of the validation plan (see ISO 26262-4) Independence with regard to the developers of the item, project management and the authors of the work product	I0	I1	I2	I2	Applies to the highest ASIL among the safety goals of the item
Confirmation review of the safety analyses (see ISO 26262-9:2011, Clause 8) Independence with regard to the developers of the item, project management and the authors of the work products	I1	I1	I2	I3	Applies to the highest ASIL among the safety goals of the item
Confirmation review of the software tool criteria evaluation report and the software tool qualification report ^b (see ISO 26262-8:2011, Clause 11) Independence with regard to the persons performing the qualification of the software tool	—	I0	I1	I1	Applies to the highest ASIL of the requirements that can be violated by the use of the tool

Table 1 (continued)

Confirmation measures	Degree of independency^a applies to ASIL				Scope
	A	B	C	D	
Confirmation review of the proven in use arguments (analysis, data and credit), of the candidates (see ISO 26262-8:2011, Clause 14) Independence with regard to the author of the argument	I0	I1	I2	I3	Applies to the ASIL of the safety goal or requirement related to the considered behaviour, or function, of the candidate
Confirmation review of the completeness of the safety case (see 6.5.3) Independence with regard to the authors of the safety case	I0	I1	I2	I3	Applies to the highest ASIL among the safety goals of the item
Functional safety audit in accordance with 6.4.8 Independence with regard to the developers of the item and project management	—	I0	I2	I3	Applies to the highest ASIL among the safety goals of the item
Functional safety assessment in accordance with 6.4.9 Independence with regard to the developers of the item and project management	—	I0	I2	I3	Applies to the highest ASIL among the safety goals of the item

^a The notations are defined as follows:

- : no requirement and no recommendation for or against regarding this confirmation measure;
- I0: the confirmation measure should be performed; however, if the confirmation measure is performed, it shall be performed by a different person;
- I1: the confirmation measure shall be performed, by a different person;
- I2: the confirmation measure shall be performed, by a person from a different team, i.e. not reporting to the same direct superior;
- I3: the confirmation measure shall be performed, by a person from a different department or organization, i.e. independent from the department responsible for the considered work product(s) regarding management, resources and release authority.

b A software tool development is outside the item's safety lifecycle whereas the qualification of such a tool is an activity of the safety lifecycle.

Table 2 — Procedural requirements for confirmation measures

Topic	Confirmation review	Functional safety audit	Functional safety assessment
Subject for evaluation	Work product	Implementation of the processes required for functional safety	Item as described in the item definition in accordance with ISO 26262-3:2011, Clause 5
Result	Confirmation review report ^a	Functional safety audit report ^a in accordance with 6.4.8	Functional safety assessment report in accordance with 6.4.9
Responsibility of the persons that perform the confirmation measure	Evaluation of the compliance of the work product with the corresponding requirements of ISO 26262	Evaluation of the implementation of the required processes	Evaluation of the achieved functional safety Provision of a recommendation for acceptance, a conditional acceptance or a rejection, in accordance with 6.4.9.6
Timing during the safety lifecycle	After completion of the corresponding safety activity Completion before the release for production	During the implementation of the required processes	Progressively during development, or in a single block Completion before the release for production
Scope and depth	In accordance with the safety plan	Implementation of the processes against the definitions of the activities referenced or specified in the safety plan	The work products required by the safety plan, the implementation of the required processes and a review of the implemented safety measures that can be assessed during the item development

^a This report can be included in a functional safety assessment report.

6.4.7.2 The persons who carry out a confirmation measure shall have access to, and shall be supported by, the persons and organizational entities that carry out safety activities during the item development.

6.4.7.3 The persons who carry out a confirmation measure shall have access to the relevant information and tools.

6.4.8 Functional safety audit

6.4.8.1 A functional safety audit shall be carried out for items, where the highest ASIL of the item's safety goals is ASIL (B), C, or D, in accordance with 6.4.7, 6.4.3.5 i) and 6.4.8.2.

6.4.8.2 One or more persons shall be appointed to carry out one or more functional safety audits, in accordance with 5.4.3. The appointed persons shall provide a report that contains an evaluation of the implementation of the processes required for functional safety.

NOTE 1 If a functional safety audit is performed by a Software Process Improvement and Capability Determination (SPICE) assessor, then this functional safety audit and a SPICE assessment (see ISO/IEC 15504) can be performed simultaneously. There can be sufficient commonality in content between ISO 26262 and SPICE to allow synchronization of the planning. If synchronized, the SPICE assessor can provide feedback to the functional safety auditor. However, a SPICE assessment can only be synchronized with regard to the examination of some of the supporting process specified in ISO 26262-8 and is not sufficient to perform the functional safety assessment (see 6.4.9).

NOTE 2 An organization's process definitions can address multiple standards at the same time, e.g. ISO 26262 and SPICE configuration management process requirements. This coordination of processes can help to avoid duplication of work or process inconsistencies. For these coordinated processes, organization-specific process cross-references to the requirements of ISO 26262 and to SPICE can be provided.

6.4.9 Functional safety assessment

6.4.9.1 A functional safety assessment shall be carried out for items, where the highest ASIL of the item's safety goals is ASIL (B), C, or D, in accordance with 6.4.7 and 6.4.9.2 to 6.4.9.8.

6.4.9.2 A functional safety assessment shall be planned in accordance with 6.4.3.3 and 6.4.3.5 i).

EXAMPLE Agenda for a functional safety assessment given in Annex E.

6.4.9.3 One or more persons shall be appointed to carry out a functional safety assessment, in accordance with 5.4.3. The appointed persons shall provide a report that contains a judgement of the achieved functional safety.

6.4.9.4 The scope of a functional safety assessment shall include

- the work products required by the safety plan;
- the processes required for functional safety; and

NOTE 1 The evaluation of the implemented processes can be based on the results of the functional safety audit(s).

— reviewing the appropriateness and effectiveness of the implemented safety measures that can be assessed during the item development.

NOTE 2 The safety measures that are to be implemented during the production subphase, but that cannot be assessed during the item development, are evaluated in conjunction with the production process capability (see ISO 26262-7:2011, 5.4.2.2).

6.4.9.5 A functional safety assessment shall consider:

- a) the planning of the other confirmation measures [see 6.4.3.5 i)];
- b) the results from the confirmation reviews and functional safety audit(s); and

- c) the recommendation(s) resulting from the previous functional safety assessment(s), if applicable (see 6.4.9.7, 6.4.9.8 and ISO 26262-8:2011, 8.4.5.2).

6.4.9.6 A functional safety assessment report, in accordance with 6.4.9.3, shall include a recommendation for acceptance, conditional acceptance, or rejection of the functional safety of the item. In the case of conditional acceptance:

- a) conditional acceptance shall only be given, if the functional safety of the item is considered evident, despite the identified open issues; and
- b) the recommendation for conditional acceptance shall include the deviations from the functional safety assessment criteria and the rationales as to why the specific deviations are considered acceptable.

6.4.9.7 If the recommendation in a functional safety assessment report in accordance with 6.4.9.6 is a conditional acceptance of the achieved functional safety, the corrective actions provided in the functional safety assessment report should be carried out.

6.4.9.8 If the recommendation in a functional safety assessment report in accordance with 6.4.9.6 is a rejection of the achieved functional safety, then:

- a) adequate corrective actions shall be initiated; and
- b) the functional safety assessment shall be repeated.

6.5 Work products

6.5.1 Safety plan, resulting from 6.4.3 to 6.4.5.

6.5.2 Project plan (refined), resulting from 6.4.3.4.

6.5.3 Safety case, resulting from 6.4.6.

6.5.4 Functional safety assessment plan, resulting from 6.4.9.

6.5.5 Confirmation measure reports, resulting from 6.4.7 to 6.4.9.

7 Safety management after the item's release for production

7.1 Objective

The objective of this clause is to define the responsibilities of the organizations and persons responsible for functional safety after the item's release for production. This relates to the general activities for ensuring the required functional safety of the item during the lifecycle subphases after the release for production.

7.2 General

See 5.2.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- evidence of quality management in accordance with 5.5.3.

7.3.2 Further supporting information

None.

7.4 Requirements and recommendations

7.4.1 General

The organizations involved in the execution of the safety lifecycle shall comply with 7.4.2 for items that have at least one safety goal with an ASIL A, B, C, or D.

7.4.2 Responsibilities, planning and required processes

7.4.2.1 The organization shall appoint persons with the responsibility and the corresponding authority, in accordance with 5.4.2.8, to maintain the functional safety of the item after its release for production.

7.4.2.2 The activities for ensuring the functional safety of the item after its release for production shall be planned, in accordance with ISO 26262-7, and shall be initiated during the product development at the system level in accordance with ISO 26262-4.

7.4.2.3 The organization shall institute, execute and maintain processes in order to maintain the functional safety of the item in the lifecycle phases after the release for production.

7.4.2.4 The organization shall institute, execute and maintain a field monitoring process with respect to the item's functional safety.

NOTE 1 The field monitoring process for safety incidents includes the reporting of incidents, the measures for correction, e.g. a recall, as well as the corresponding decision-making processes.

NOTE 2 The data collected from field monitoring can be used for proven in use arguments (see ISO 26262-8:2011, Clause 14).

7.4.2.5 If the item changes after its release for production, the release for production in accordance with ISO 26262-4:2011, Clause 11, shall be reissued.

NOTE These changes are subject to change management (see ISO 26262-8:2011, Clause 8).

7.5 Work products

7.5.1 Evidence of field monitoring, resulting from 7.4.2.4.

Annex A (informative)

Overview of and workflow of functional safety management

Table A.1 provides an overview of the objectives, prerequisites and work products of the particular phases of the management of functional safety.

Table A.1 — Functional safety management: overview

Clause	Objectives	Prerequisites	Work products
5 Overall safety management	<p>The objective of Clause 5 is to define the requirements for the organizations that are responsible for the safety lifecycle, or that perform safety activities in the safety lifecycle.</p> <p>Clause 5 serves as a prerequisite to the activities in the ISO 26262 safety lifecycle.</p>	None	<p>5.5.1 Organization-specific rules and processes for functional safety.</p> <p>5.5.2 Evidence of competence.</p> <p>5.5.3 Evidence of quality management.</p>
6 Safety management during the concept phase and the product development	<p>The first objective of Clause 6 is to define the safety management roles and responsibilities, regarding the concept phase and the development phases in the safety lifecycle.</p> <p>The second objective of Clause 6 is to define the requirements for the safety management during the concept phase and the development phases, including the planning and coordination of the safety activities, the progression of the safety lifecycle, the creation of the safety case, and the execution of the confirmation measures.</p>	<p>Organization-specific rules and processes for functional safety (see 5.5.1)</p> <p>Evidence of competence (see 5.5.2)</p> <p>Evidence of quality management (see 5.5.3)</p>	<p>6.5.1 Safety plan.</p> <p>6.5.2 Project plan (refined).</p> <p>6.5.3 Safety case.</p> <p>6.5.4 Functional safety assessment plan.</p> <p>6.5.5 Confirmation measure reports.</p>
7 Safety management after the item's release for production	<p>The objective of Clause 7 is to define the responsibilities of the organizations and persons responsible for functional safety after the item's release for production. This relates to the general activities for ensuring the required functional safety of the item during the lifecycle subphases after the release for production.</p>	Evidence of quality management (see 5.5.3).	7.5 Evidence of field monitoring.

Annex B (informative)

Examples for evaluating a safety culture

Table B.1 — Examples for evaluating a safety culture

Examples indicative of a poor safety culture	Examples indicative of a good safety culture
Accountability is not traceable	The process assures that accountability for decisions related to functional safety is traceable
Cost and schedule always take precedence over safety and quality	Safety is the highest priority
The reward system favours cost and schedule over safety and quality	The reward system supports and motivates the effective achievement of functional safety The reward system penalizes those who take shortcuts that jeopardize safety or quality
Personnel assessing safety, quality and their governing processes are influenced unduly by those responsible for executing the processes	The process provides adequate checks and balances, e.g. the appropriate degree of independence in the integral processes (safety, quality, verification, validation and configuration management)
Passive attitude towards safety, e.g. — heavy dependence on testing at the end of the product development cycle, — management reacts only when there is a problem in the field	Proactive attitude towards safety, e.g. — safety and quality issues are discovered and resolved from the earliest stage in the product lifecycle
The required resources are not planned or allocated in a timely manner	The required resources are allocated Skilled resources have the competence commensurate with the activity assigned
“Groupthink” “Stacking the deck” when forming review groups Dissenter is ostracised or labelled as “not a team player” Dissent reflects negatively on performance reviews “Minority dissenter” is labelled or treated as a “troublemaker”, “not a team player” or a “whistleblower” Concerned employees fear repercussion	The process uses diversity to advantage: — intellectual diversity is sought, valued and integrated in all processes — behaviour which counters the use of diversity is discouraged and penalised Supporting communication and decision-making channels exist and the management encourages their usage: — self-disclosure is encouraged — disclosure of discovery by anyone else is encouraged — the discovery and resolution process continues in the field
No systematised continuous improvement processes, learning cycles or other forms of “lessons learned”	Continuous improvement is integral to all processes
Processes are “ad hoc” or implicit	A defined, traceable and controlled process is followed at all levels, including: — management — engineering — development interfaces — verification — validation — functional safety audit — functional safety assessment

Annex C (informative)

Aim of the confirmation measures

C.1 Review of the safety plan (see 6.5.1)

C.1.1 Evaluation of the compliance of the safety plan with the ISO 26262 safety lifecycle. If applicable, evaluation of the tailoring of the safety activities, i.e. concerning the safety activities that are omitted or performed in a different manner compared to the reference safety lifecycle, including the corresponding rationales (see 6.4.5).

C.1.2 Evaluation of the compliance of the safety plan with the ISO 26262 requirements on the planning of the safety activities (see 6.4.3 to 6.4.5).

C.2 Review of the completeness of the safety case (see 6.5.3)

C.2.1 Confirmation that the work products referenced in the safety case are available and sufficiently complete, so that the item's achievement of functional safety can be adequately evaluated.

NOTE The referenced work products can be the work products that are identified as relevant to support the safety case.

C.2.2 Confirmation that the work products referenced in the safety case:

- are traceable from one to another,
- have no contradictions within or between work products, and
- either have no open issues that can lead to the violation of a safety goal, or have only open issues that are controlled and have a plan for closure.

C.3 Functional safety audit (see 6.4.8 and 6.5.5)

Evaluation of the implementation of the functional safety processes, during the execution of these processes, against the definitions of the activities referenced or specified in the safety plan.

C.4 Functional safety assessment (see 6.4.9 and 6.5.5)

C.4.1 Evaluation of the compliance of the work products required by the safety plan with the corresponding requirements of ISO 26262, including but not limited to the work products that require a confirmation review. For the latter work products, the results of the confirmation reviews are also considered.

C.4.2 Evaluation of the implementation of the functional safety processes, considering the results of the performed functional safety audit(s) (see 6.4.8).

C.4.3 A review of the appropriateness and effectiveness of the implemented safety measures that can be assessed during the item development.

C.4.4 Follow-up of the recommendations resulting from the previous functional safety assessments, including any performed corrective actions, if applicable (see 6.4.9.7 and 6.4.9.8).

C.5 Review of the hazard analysis and risk assessment (see ISO 26262-3:2011, Clause 7, and, if applicable, ISO 26262-8:2011, Clause 5)

Evaluation of the completeness of the hazard analysis and risk assessment, and of the correctness of the determined ASILs (including QM) and the safety goals, considering the ISO 26262 hazard analysis and risk assessment procedure.

C.6 Review of the item integration and testing plan (see ISO 26262-4)

Evaluation of the compliance of the item integration and testing plan with the ISO 26262 requirements on the integration and testing activities.

C.7 Review of the validation plan (see ISO 26262-4)

Evaluation of the compliance of the validation plan with the ISO 26262 requirements on the safety validation activities.

C.8 Review of the proven in use arguments of the candidates (see ISO 26262-8:2011, Clause 14), if applicable

C.8.1 Evaluation to determine whether the results of the proven in use analyses justify the claimed proven in use credits of the candidates (with regard to any associated tailoring of safety activities).

C.8.2 Evaluation of the efficiency of the field monitoring process.

C.8.3 Evaluation of the candidate changes that are considered by the proven in use argument.

C.9 Review of the software tool criteria evaluation report and the software tool qualification report (see ISO 26262-8:2011, Clause 11)

C.8.1 Confirmation of the correct evaluation of the required level of confidence in the software tool(s) used in the development of an item or a safety-related element.

C.8.2 Evaluation of the qualification of the software tool(s), considering the required level of confidence of the respective software tools.

C.10 Review of the safety analyses (see ISO 26262-9:2011, Clause 8)

Evaluation of the correct execution of the safety analyses that can identify faults or inadequate safety mechanisms that can lead to the violation of a safety goal.

Annex D (informative)

Overview of the verification reviews

Table D.1 provides an overview of verification reviews which are required in other parts of ISO 26262 (see also ISO 26262-8:2011, Clause 9).

Table D.1 — Overview of verification reviews

Verification review subject	Highest ASIL among the safety goals of the item				Clause in which required or recommended
	A	B	C	D	
Hazard analysis and risk assessment of the item (see ISO 26262-3:2011, Clauses 5 and 7, and, if applicable, ISO 26262-8:2011, Clause 5)	required ^a				ISO 26262-3:2011, Clause 7
Safety goals	required				ISO 26262-3:2011, Clause 7
Functional safety concept	required				ISO 26262-3:2011, Clause 8
Technical safety requirements specification	required				ISO 26262-4:2011, Clause 6
System design	required				ISO 26262-4:2011, Clause 7
Hardware safety requirements	required				ISO 26262-5:2011, Clause 6
Hardware design	required				ISO 26262-5:2011, Clause 7
Results of the applied methods with regard to the evaluation of the hardware architectural metrics	^b	recommended	required	required	ISO 26262-5:2011, Clause 8
Analysis of the potential safety goal violations due to random hardware failures, considering the applied evaluation method	^b	recommended	required	required	ISO 26262-5:2011, Clause 9
Software safety requirements and the refined hardware-software interface requirements	required				ISO 26262-6:2011, Clauses 6 and 11
Software architectural design	required				ISO 26262-6:2011, Clause 7
Software unit design and implementation	required				ISO 26262-6:2011, Clause 8
Software component qualification report	required for the qualified software components				ISO 26262-8:2011, Clause 12
Hardware component qualification report	required for the qualified hardware components				ISO 26262-8:2011, Clause 13
Safety analyses	required				ISO 26262-9:2011, Clause 8

^a The scope of this review also includes hazardous events rated as QM.

^b No requirement and no recommendation for or against.

Annex E (informative)

Example of a functional safety assessment agenda (for items that have an ASIL D safety goal)

E.1 Safety management

- E.1.1** Application of the organization's safety culture and supporting processes in the assessed project.
- E.1.2** Application of the competence management and the continuous improvement practice in the assessed project.
- E.1.3** Roles and responsibilities in the assessed project.
- E.1.4** Safety plan of the assessed project and planning of the distributed development.
- E.1.5** Tailoring of the safety lifecycle, including the proven in use arguments of the candidates, of the assessed project.
- E.1.6** Functional safety audits, the safety case and available documents.

E.2 Safety activities during the concept phase

- E.2.1** Item definition.
- E.2.2** Hazard analysis and risk assessment.
- E.2.3** Functional safety concept.
- E.2.4** Dependencies of the item and its safety concept with other systems/functions.
- E.2.5** Allocation of functional safety requirements to:
 - E/E elements;
 - elements implemented by other technologies;
 - interfaces with external measures.
- E.2.6** Verification of the functional safety concept.

E.3 Safety activities during the system development

- E.3.1** Planning of the system development, integration and validation.
- E.3.2** Technical safety concept and its verification.
- E.3.3** System design and avoidance of systematic failures.
- E.3.4** Allocation of the technical safety requirements to hardware and software elements and a review of the hardware-software interface.

E.3.5 Verification of the system design.

E.4 Hardware development

- E.4.1** Planning of the hardware development, qualification and integration.
- E.4.2** Hardware safety requirements, hardware design and verification.
- E.4.3** Hardware architectural constraints.
- E.4.4** Evaluation of the probability of violation of the safety goals by random hardware failures.
- E.4.5** Hardware integration and testing.

E.5 Software development

- E.5.1** Planning of the software development, qualification and integration.
- E.5.2** Software safety requirements, software architectural design, software unit design and implementation.
- E.5.3** Software unit testing.
- E.5.4** Software integration and testing.
- E.5.5** Verification of the software safety requirements.

E.6 Item integration

- E.6.1** Planning of the integration tests.
- E.6.2** Hardware-software integration and testing.
- E.6.3** System/item integration.
- E.6.4** Vehicle integration.

E.7 Safety validation and the release for production

- E.7.1** Validation activities.
- E.7.2** Validation documentation and the release for production.

E.8 Planning of production and maintenance

- E.8.1** Safety-related special characteristics for production.
- E.8.2** Safety-related special characteristics for operation, service and decommissioning.

E.9 Summary

Functional safety assessment documentation, the recommendations and actions to be taken after the functional safety assessment.

Bibliography

- [1] ISO 9001, *Quality management systems — Requirements*
- [2] ISO/TS 16949, *Quality management systems — Particular requirements for the application of ISO 9001:2008 for automotive production and relevant service part organizations*
- [3] ISO/IEC 15504 (all parts), *Information technology — Process assessment*
- [4] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*

This page is intentionally blank.

This page is intentionally blank.

ICS 43.040.10

Price based on 26 pages

INTERNATIONAL
STANDARD

ISO
26262-3

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 3:
Concept phase**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 3: Phase de projet*



Reference number
ISO 26262-3:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction.....	v
1 Scope.....	1
2 Normative references.....	1
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	2
4.3 ASIL-dependent requirements and recommendations	3
5 Item definition	3
5.1 Objectives	3
5.2 General	3
5.3 Inputs to this clause.....	3
5.4 Requirements and recommendations	4
5.5 Work products	4
6 Initiation of the safety lifecycle	5
6.1 Objectives	5
6.2 General	5
6.3 Inputs to this clause.....	5
6.4 Requirements and recommendations	5
6.5 Work products	6
7 Hazard analysis and risk assessment.....	6
7.1 Objectives	6
7.2 General	7
7.3 Inputs to this clause.....	7
7.4 Requirements and recommendations	7
7.5 Work products	12
8 Functional safety concept.....	12
8.1 Objectives	12
8.2 General	12
8.3 Inputs to this clause.....	13
8.4 Requirements and recommendations	14
8.5 Work products	16
Annex A (informative) Overview and document flow of concept phase	17
Annex B (informative) Hazard analysis and risk assessment.....	18
Bibliography.....	25

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-3 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

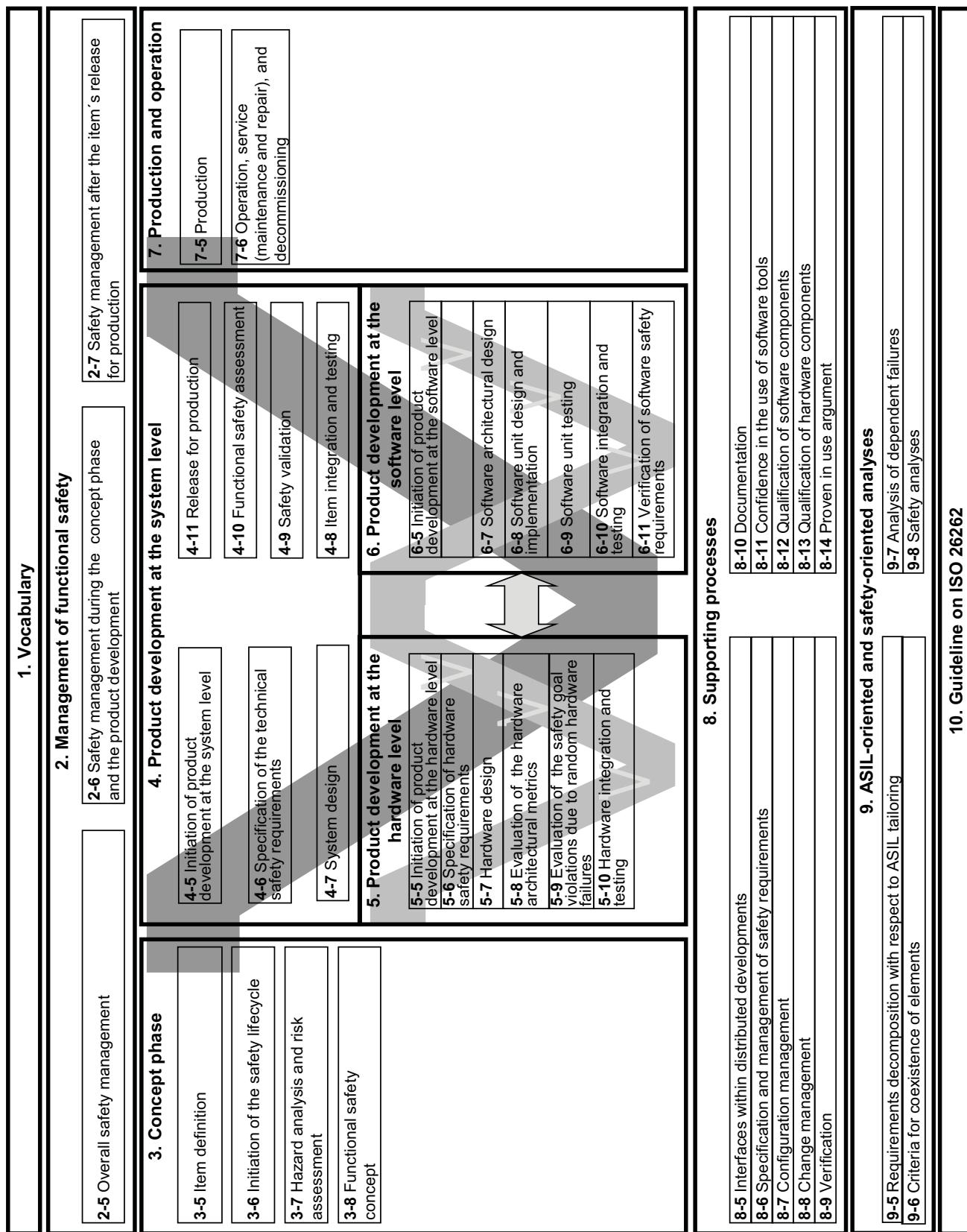


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 3: Concept phase

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for the concept phase for automotive applications, including the following:

- item definition,
- initiation of the safety lifecycle,
- hazard analysis and risk assessment, and
- functional safety concept.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- a) tailoring of the safety activities in accordance with ISO 26262-2 has been planned and shows that the requirement does not apply, or
- b) a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Item definition

5.1 Objectives

The first objective is to define and describe the item, its dependencies on, and interaction with, the environment and other items.

The second objective is to support an adequate understanding of the item so that the activities in subsequent phases can be performed.

5.2 General

This clause lists the requirements and recommendations for establishing the definition of the item with regard to its functionality, interfaces, environmental conditions, legal requirements, hazards, etc. This definition serves to provide sufficient information about the item to the persons who conduct the subsequent subphases: “Initiation of safety lifecycle” (see Clause 6), “Hazard analysis and risk assessment” (see Clause 7) and “Functional safety concept” (see Clause 8).

NOTE Table A.1 provides an overview of objectives, prerequisites and work products of the concept phase.

5.3 Inputs to this clause

5.3.1 Prerequisites

None.

5.3.2 Further supporting information

The following information can be considered:

- any information that already exists concerning the item, e.g. a product idea, a project sketch, relevant patents, the results of pre-trials, the documentation from predecessor items, relevant information on other independent items.

5.4 Requirements and recommendations

5.4.1 The functional and non-functional requirements of the item as well as the dependencies between the item and its environment shall be made available.

NOTE 1 Requirements can be classified as safety-related after safety goals and their respective ASIL have been defined.

NOTE 2 The required information is a necessary input for the item definition although it is not safety-related. If not already available, its generation can be triggered by the requirements of this clause.

This information includes:

- a) the functional concept, describing the purpose and functionality, including the operating modes and states of the item;
- b) the operational and environmental constraints;
- c) legal requirements (especially laws and regulations), national and international standards;
- d) behaviour achieved by similar functions, items or elements, if any;
- e) assumptions on behaviour expected from the item; and
- f) potential consequences of behaviour shortfalls including known failure modes and hazards.

NOTE This can include known safety-related incidents on similar items.

5.4.2 The boundary of the item, its interfaces, and the assumptions concerning its interaction with other items and elements, shall be defined considering:

- a) the elements of the item;

NOTE The elements could also be based on other technology

- b) the assumptions concerning the effects of the item's behaviour on other items or elements, that is the environment of the item;
- c) interactions of the item with other items or elements;
- d) functionality required by other items, elements and the environment;
- e) functionality required from other items, elements and the environment;
- f) the allocation and distribution of functions among the involved systems and elements; and
- g) the operating scenarios which impact the functionality of the item.

5.5 Work products

Item definition resulting from the requirements of 5.4.

6 Initiation of the safety lifecycle

6.1 Objectives

The first objective of the initiation of the safety lifecycle is to make the distinction between a new item development and a modification to an existing item (see ISO 26262-2:2011, Figure 2).

The second objective is to define the safety lifecycle activities (see ISO 26262-2:2011, Figure 2) that will be carried out in the case of a modification.

6.2 General

Based on the item definition, the safety lifecycle is initiated by distinguishing between either a new development, or a modification of an existing item. In the case of a modification, the tailoring of the safety-related activities takes place.

6.3 Inputs to this clause

6.3.1 Prerequisites

The following information shall be available:

- item definition in accordance with 5.5.

6.3.2 Further supporting information

The following information can be considered:

- any existing information, not already covered by the item definition, being useful for conducting the impact analysis.

EXAMPLE Product concept, requests for change, implementation planning, proven in use argument.

6.4 Requirements and recommendations

6.4.1 Determination of the development category

6.4.1.1 It shall be determined whether the item is either a new development, or if it is a modification of an existing item or its environment:

- a) in the case of a new development, the development shall be continued with the hazard analysis and risk assessment in accordance with Clause 7;
- b) in the case of a modification of the item or its environment the applicable lifecycle subphases and activities shall be determined in accordance with 6.4.2.

NOTE A proven in use argument can be applied to modification (see ISO 26262-8:2011, Clause 14).

6.4.2 Impact analysis and possible tailored safety lifecycle, in the case of modification

6.4.2.1 An impact analysis shall be carried out in order to identify and describe the intended modification applied to the item or its environment and to assess the impact of these modifications.

NOTE 1 Modifications to the item include design modifications and implementation modifications. Design modification can result from requirements modifications (e.g. functional or performance enhancement or cost optimisation). Implementation modifications do not affect the specification or performance of the item, but only the implementation features.

EXAMPLE Implementation modifications can result from corrections of software, or the use of new development or production tools.

NOTE 2 Modifications to configuration data or calibration data are considered as modifications to the item if they impact the functional behaviour of the item.

NOTE 3 Modifications to the environment of the item can result from the installation of the item in a new target environment (e.g. another vehicle variant) or by the upgrading of other items or elements interacting with (or in the vicinity of) the item.

6.4.2.2 The impact analysis shall identify and address areas affected by the modifications to the item and modifications between previous and future conditions of use of the item, including:

- a) operational situations and operating modes;
- b) interfaces with the environment;
- c) installation characteristics such as location within the vehicle, vehicle configurations and variants; and
- d) a range of environmental conditions e.g. temperature, altitude, humidity, vibrations, Electromagnetic Interference (EMI) and fuel types.

6.4.2.3 The implication of the modification with regard to functional safety shall be identified and described.

6.4.2.4 The affected work products that need to be updated shall be identified and described.

6.4.2.5 The safety activities shall be tailored in accordance with the applicable lifecycle phases.

6.4.2.6 Tailoring shall be based on the results of the impact analysis.

6.4.2.7 The results of tailoring shall be included in the safety plan in accordance with ISO 26262-2:2011, 6.4.3.

6.4.2.8 The affected work products shall be reworked.

NOTE The affected work products include the validation plan (see ISO 26262-4).

6.4.2.9 In the case of missing work products or work products that do not comply with ISO 26262, the necessary activities to reach ISO 26262 compliance shall be determined.

6.5 Work products

6.5.1 Impact analysis resulting from the requirements of 6.4.2.1 to 6.4.2.4.

6.5.2 Safety plan (refined) resulting from the requirements 6.4.2.5 to 6.4.2.9.

7 Hazard analysis and risk assessment

7.1 Objectives

The objective of the hazard analysis and risk assessment is to identify and to categorise the hazards that malfunctions in the item can trigger and to formulate the safety goals related to the prevention or mitigation of the hazardous events, in order to avoid unreasonable risk.

7.2 General

Hazard analysis, risk assessment and ASIL determination are used to determine the safety goals for the item such that an unreasonable risk is avoided. For this, the item is evaluated with regard to its potential hazardous events. Safety goals and their assigned ASIL are determined by a systematic evaluation of hazardous events. The ASIL is determined by considering the estimate of the impact factors, i.e. severity, probability of exposure and controllability. It is based on the item's functional behaviour; therefore, the detailed design of the item does not necessarily need to be known.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- item definition in accordance with 5.5.

7.3.2 Further supporting information

The following information can be considered:

- impact analysis, if applicable (see 6.5.1); and
- relevant information on other independent items (from external source).

7.4 Requirements and recommendations

7.4.1 Initiation of the hazard analysis and risk assessment

7.4.1.1 The hazard analysis and risk assessment shall be based on the item definition.

7.4.1.2 The item without internal safety mechanisms shall be evaluated during the hazard analysis and risk assessment, i.e. safety mechanisms intended to be implemented or that have already been implemented in predecessor items shall not be considered in the hazard analysis and risk assessment.

NOTE 1 In the evaluation of an item, available and sufficiently independent external measures can be beneficial.

EXAMPLE Electronic stability control can mitigate the effect of failures in chassis systems by providing increased control if it is shown to be available and sufficiently independent.

NOTE 2 Safety mechanisms of the item that are intended to be implemented or that have already been implemented are incorporated as part of the functional safety concept.

7.4.2 Situation analysis and hazard identification

7.4.2.1 Situation analysis

7.4.2.1.1 The operational situations and operating modes in which an item's malfunctioning behaviour will result in a hazardous event shall be described, both for cases when the vehicle is correctly used and when it is incorrectly used in a foreseeable way.

NOTE The operational situation addresses the limits within which the item is expected to behave in a safe manner. For example, a normal passenger road vehicle is not expected to travel cross-country at high speed.

7.4.2.2 Hazard identification

7.4.2.2.1 The hazards shall be determined systematically by using adequate techniques.

NOTE Techniques such as brainstorming, checklists, quality history, FMEA and field studies can be used for the extraction of hazards at the item level.

7.4.2.2.2 Hazards shall be defined in terms of the conditions or behaviour that can be observed at the vehicle level.

NOTE 1 In general, each hazard will have a variety of potential causes related to the item's implementation but they do not need to be considered in the hazard analysis and risk assessment for the definition of the conditions or behaviour, which result from a functional behaviour of the item.

NOTE 2 Only hazards associated with the item itself can be considered, every other system (external measure) is presumed to be functioning correctly provided it is sufficiently independent.

7.4.2.2.3 The hazardous events shall be determined for relevant combinations of operational situations and hazards.

7.4.2.2.4 The consequences of hazardous events shall be identified.

NOTE If failures at an item level induce the loss of several functions of the item, then the situation analysis and hazard identification considers the resulting hazardous events from the combined dysfunctional behaviour of the item or vehicle.

EXAMPLE Failure of the vehicle electrical power supply system can cause the simultaneous loss of a number of functions including "engine torque", "power assisted steering" and "forward illumination".

7.4.2.2.5 If there are hazards identified in 7.4.2.2 that are outside of the scope of ISO 26262 (see Clause 1), then the need for appropriate measures to mitigate or control these hazards shall be highlighted and reported to the responsible persons.

NOTE As these hazards are outside the scope of ISO 26262, hazard classification is not necessary.

7.4.3 Classification of hazardous events

7.4.3.1 All hazardous events identified in 7.4.2.3 shall be classified, except those that are outside the scope of ISO 26262.

NOTE If classification of a given hazard with respect to severity, probability of exposure or controllability is difficult to make, it is classified conservatively, i.e. whenever there is any doubt, a higher ASIL classification is given rather than a lower.

7.4.3.2 The severity of potential harm shall be estimated based on a defined rationale for each hazardous event. The severity shall be assigned to one of the severity classes S0, S1, S2 or S3 in accordance with Table 1.

NOTE 1 The risk assessment of hazardous events focuses on the harm to each person potentially at risk – including the driver or the passengers of the vehicle causing the hazardous event, and other persons potentially at risk such as cyclists, pedestrians or occupants of other vehicles. The description of the Abbreviated Injury Scale (AIS) can be used for characterising the severity and can be found in Annex B. For informative examples of different types of severity and accidents see Annex B.

NOTE 2 The severity class can be based on a combination of injuries, and this can lead to a higher evaluation of the severity than would result from just looking at single injuries.

NOTE 3 The estimate considers reasonable sequences of events for the situation being evaluated.

NOTE 4 The severity determination is based on a representative sample of individuals for the target markets.

Table 1 — Classes of severity

	Class			
	S0	S1	S2	S3
Description	No injuries	Light and moderate injuries	Severe and life-threatening injuries (survival probable)	Life-threatening injuries (survival uncertain), fatal injuries

7.4.3.3 The severity class S0 may be assigned if the hazard analysis determines that the consequences of a malfunctioning behaviour of the item are clearly limited to material damage and do not involve harm to persons. If a hazard is assigned to severity class S0, no ASIL assignment is required.

7.4.3.4 The probability of exposure of each operational situation shall be estimated based on a defined rationale for each hazardous event. The probability of exposure shall be assigned to one of the probability classes, E0, E1, E2, E3 and E4, in accordance with Table 2.

NOTE 1 For classes E1 to E4, the difference in probability from one E class to the next is an order of magnitude.

NOTE 2 The exposure determination is based on a representative sample of operational situations for the target markets.

NOTE 3 For details and examples related to the probability of exposure see Annex B.

Table 2 — Classes of probability of exposure regarding operational situations

	Class				
	E0	E1	E2	E3	E4
Description	Incredible	Very low probability	Low probability	Medium probability	High probability

7.4.3.5 The number of vehicles equipped with the item shall not be considered when estimating the probability of exposure.

NOTE The evaluation of the probability of exposure is performed assuming each vehicle is equipped with the item. This means that the argument “the probability of exposure can be reduced, because the item is not present in every vehicle (as only some vehicles are equipped with the item)” is not valid.

7.4.3.6 Class E0 may be used for those situations that are suggested during hazard analysis and risk assessment, but which are considered to be extremely unusual, or incredible, and therefore not followed up. A rationale shall be recorded for the exclusion of these situations. If a hazard is assigned to exposure class E0, no ASIL assignment is required.

EXAMPLE E0 can be used in the case of “force majeure” risk (see Clause B.3).

7.4.3.7 The controllability of each hazardous event, by the driver or other persons potentially at risk, shall be estimated based on a defined rationale for each hazardous event. The controllability shall be assigned to one of the controllability classes C0, C1, C2 and C3 in accordance with Table 3.

NOTE 1 For classes C1 to C3, the difference in probability from one C class to the next is an order of magnitude.

NOTE 2 The evaluation of the controllability is an estimate of the probability that the driver or other persons potentially at risk are able to gain sufficient control of the hazardous event, such that they are able to avoid the specific harm. For this purpose, the parameter C is used, with the classes C1, C2 and C3, to classify the potential of avoiding harm. It is assumed that the driver is in an appropriate condition to drive (e.g. he/she is not tired), has the appropriate driver training (he/she has a driver's licence) and is complying with all applicable legal regulations, including due care requirements to avoid risks to other traffic participants. Some examples, which serve as an interpretation of these classes, are listed in Table B.4. Reasonably foreseeable misuse is considered.

NOTE 3 Where the hazardous event is not related to the control of the vehicle direction and speed, e.g. potential limb entrapment in moving parts, the controllability can be an estimate of the probability that the person at risk is able to remove themselves, or to be removed by others from the hazardous situation. When considering controllability, note that the person at risk might not be familiar with the operation of the item.

NOTE 4 When controllability involves the actions of multiple traffic participants, the controllability assessment can be based on the controllability of the vehicle with the malfunctioning item, and the likely action of other participants.

Table 3 — Classes of controllability

	Class			
	C0	C1	C2	C3
Description	Controllable in general	Simply controllable	Normally controllable	Difficult to control or uncontrollable

7.4.3.8 Class C0 may be used for hazards addressing the unavailability of the item if they do not affect the safe operation of the vehicle (e.g. some driver assistance systems). Class C0 may also be assigned if dedicated regulations exist that specify the functional performance with respect to a defined hazard, and C0 is argued using the corresponding existing experience concerning sufficient controllability. If a hazard is assigned to the controllability class C0, no ASIL assignment is required.

EXAMPLE A dedicated regulation is the certification of a vehicle system with a precise definition of forces or acceleration values in the case of a failure.

7.4.4 Determination of ASIL and safety goals

7.4.4.1 An ASIL shall be determined for each hazardous event using the parameters "severity", "probability of exposure" and "controllability" in accordance with Table 4.

NOTE 1 Four ASILs are defined: ASIL A, ASIL B, ASIL C and ASIL D, where ASIL A is the lowest safety integrity level and ASIL D the highest one.

NOTE 2 In addition to these four ASILs, the class QM (quality management) denotes no requirement to comply with ISO 26262.

Table 4 — ASIL determination

Severity class	Probability class	Controllability class		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

7.4.4.2 It shall be ensured that the chosen level of detail of the list of operational situations does not lead to an inappropriate lowering of the ASIL of the corresponding safety goals.

NOTE A very detailed list of operational situations (see 7.4.2.1.1) for one hazard, with regard to the vehicle state, road conditions and environmental conditions, can lead to a very granular classification of hazardous events. This can make it easier to rate controllability and severity. However, a larger number of different operational situations can lead to a consequential reduction of the respective classes of exposure, and thus to an inappropriate lowering of the ASIL of the corresponding safety goals.

7.4.4.3 A safety goal shall be determined for each hazardous event with an ASIL evaluated in the hazard analysis. If similar safety goals are determined, these may be combined into one safety goal.

NOTE Safety goals are top-level safety requirements for the item. They lead to the functional safety requirements needed to avoid an unreasonable risk for each hazardous event. Safety goals are not expressed in terms of technological solutions, but in terms of functional objectives.

7.4.4.4 The ASIL determined for the hazardous event shall be assigned to the corresponding safety goal. If similar safety goals are combined into a single one, in accordance with 7.4.4.3, the highest ASIL shall be assigned to the combined safety goal.

NOTE If combined safety goals refer to the same hazard in different situations, then the resulting ASIL of the safety goal is the highest one of the considered safety goals of every situation.

7.4.4.5 If a safety goal can be achieved by transitioning to, or by maintaining, one or more safe states, then the corresponding safe state(s) shall be specified.

NOTE Safe states are further elaborated in Clause 8.

EXAMPLE A safe state could be switched off, locked, vehicle stationary, and maintained functionality in the case of a failure over a defined time.

7.4.4.6 The safety goals together with their attributes (ASIL) shall be specified in accordance with ISO 26262-8:2011, Clause 6.

NOTE The safety goal can include features such as the fault tolerant time interval, or physical characteristics (e.g. a maximum level of unwanted steering-wheel torque, maximum level of unwanted acceleration) if they were relevant to the ASIL determination.

7.4.5 Verification

7.4.5.1 The hazard analysis, risk assessment and the safety goals shall be verified in accordance with ISO 26262-8:2011, Clause 9, to show their:

- completeness with regard to situations (7.4.2.1) and hazards (7.4.2.2);
- compliance with the item definition;
- consistency with related hazard analyses and risk assessments;
- completeness of the coverage of the hazardous events; and
- consistency of the assigned ASILs with the corresponding hazardous events.

NOTE This verification review checks the hazard analysis and risk assessment of the item for correctness and completeness, i.e. considered situations, hazards and parameter estimations (severity, probability of exposure and controllability). In contrast, the confirmation review of the hazard analysis and risk assessment in accordance with ISO 26262-2, checks formally that the hazard analysis and risk assessment procedure complies with the requirements of Clause 7. The confirmation review is performed by a person or persons from a different department or organisation, than the developers of the item.

7.5 Work products

7.5.1 **Hazard analysis and risk assessment** resulting from the requirements of 7.4.1.1 to 7.4.4.2

7.5.2 **Safety goals** resulting from the requirements of 7.4.4.3 to 7.4.4.6

7.5.3 **Verification review report of the hazard analysis and risk assessment and the safety goals** resulting from the requirement of 7.4.5.

8 Functional safety concept

8.1 Objectives

The objective of the functional safety concept is to derive the functional safety requirements, from the safety goals, and to allocate them to the preliminary architectural elements of the item, or to external measures.

8.2 General

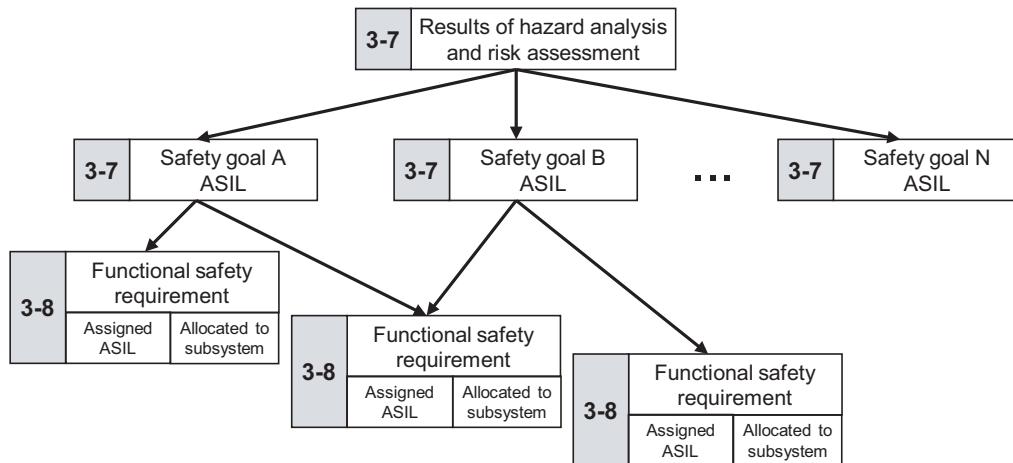
To comply with the safety goals, the functional safety concept contains safety measures, including the safety mechanisms, to be implemented in the item's architectural elements and specified in the functional safety requirements.

The functional safety concept addresses:

- fault detection and failure mitigation;
- transitioning to a safe state;
- fault tolerance mechanisms, where a fault does not lead directly to the violation of the safety goal(s) and which maintains the item in a safe state (with or without degradation);
- fault detection and driver warning in order to reduce the risk exposure time to an acceptable interval (e.g. engine malfunction indicator lamp, ABS fault warning lamp); and
- arbitration logic to select the most appropriate control request from multiple requests generated simultaneously by different functions.

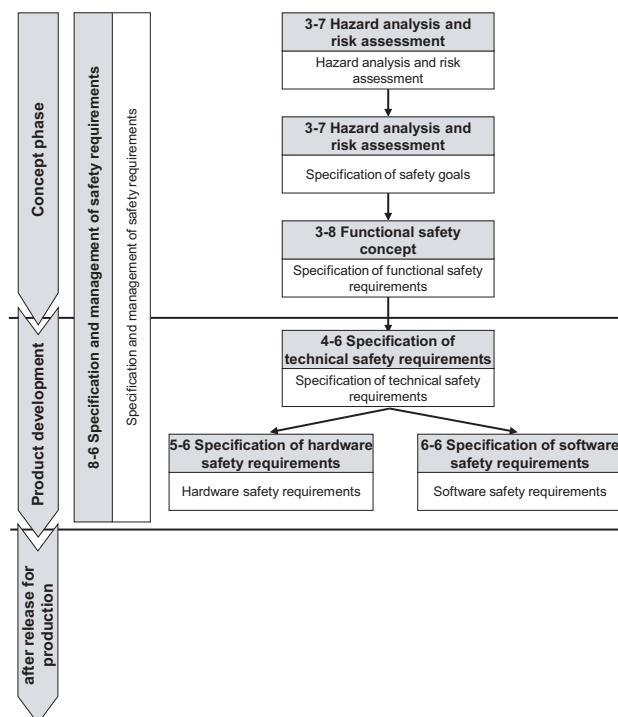
Figure 2 illustrates the hierarchical approach by which the safety goals are determined as a result of the hazard analysis and risk assessment. The functional safety requirements are then derived from the safety goals.

The structure and distribution of the safety requirements within the corresponding Parts of ISO 26262 are illustrated in Figure 3. The functional safety requirements are allocated to the elements of the preliminary architecture.



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: “m-n”, where “m” represents the number of the part and “n” indicates the number of the clause, e.g. “3-6” represents Clause 6 of ISO 26262-3.

Figure 2 — Hierarchy of safety goals and functional safety requirements



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: “m-n”, where “m” represents the number of the part and “n” indicates the number of the clause, e.g. “3-6” represents Clause 6 of ISO 26262-3.

Figure 3 — Structure of the safety requirements

8.3 Inputs to this clause

8.3.1 Prerequisites

The following information shall be available:

- item definition in accordance with 5.5;
- hazard analysis and risk assessment in accordance with 7.5.1; and
- safety goals in accordance with 7.5.2.

8.3.2 Further supporting information

The following information can be considered:

- preliminary architectural assumptions (from external source).

8.4 Requirements and recommendations

8.4.1 General

The functional safety requirements shall be specified in accordance with ISO 26262-8:2011, Clause 6.

8.4.2 Derivation of functional safety requirements

8.4.2.1 The functional safety requirements shall be derived from the safety goals and safe states, taking into account the preliminary architectural assumptions.

8.4.2.2 At least one functional safety requirement shall be specified for each safety goal.

NOTE One functional safety requirement can be valid for several safety goals.

8.4.2.3 Each functional safety requirement shall be specified by considering the following, if applicable:

- a) operating modes;
- b) fault tolerant time interval;
- c) safe states;
- d) emergency operation interval, and
- e) functional redundancies (e.g. fault tolerance).

NOTE This activity can be supported by safety analyses (e.g. FMEA, FTA, HAZOP) in order to develop a complete set of effective functional safety requirements.

8.4.2.4 If a safe state cannot be reached by a transition within an acceptable time interval, an emergency operation shall be specified.

EXAMPLE When a safe state cannot be reached by immediately switching off a system, a suitable emergency operation needs to be specified.

8.4.2.5 The warning and degradation concept shall be specified as functional safety requirements.

NOTE The transitions to and from a safe state and the conditions for transitioning (switching to the safe state and recovering from the safe state) are described in the warning and degradation concept.

EXAMPLE 1 Fault detection and failure mitigation by switching to a safe state.

EXAMPLE 2 Fault detection and driver warning in order to reduce the risk exposure time to an acceptable interval (e.g. engine malfunction indicator lamp, ABS fault warning lamp).

8.4.2.6 If assumptions are made about the necessary actions of the driver, or other persons potentially at risk, in order to comply with the safety goals, then the following shall apply:

NOTE 1 The actions include those for which credit was taken during controllability estimation, and any further necessary actions taken to comply with the safety goals after the implementation of the safety requirements.

EXAMPLE ACC: the override of brake activation by the driver pushing the accelerator pedal.

- a) these actions shall be specified in the functional safety concept; and
- b) the adequate means and controls available to the driver or other persons potentially at risk shall be specified in the functional safety concept.

NOTE 1 Driver task analysis can be helpful to consider prevention of driver overload, prevention of driver surprise/panic/shock (loss of capability to control vehicle), and mode confusion (an incorrect assumption about the operating mode).

NOTE 2 The specification of the warning and degradation concept and the necessary actions of the driver and other persons potentially at risk is an input for the user manual (see ISO 26262-7:2011, 6.4.1).

8.4.3 Allocation of functional safety requirements

- 8.4.3.1** The functional safety requirements shall be allocated to the elements of the preliminary architectural assumptions:

NOTE Redundancy and independence issues can be checked by an analysis of dependent failures (see ISO 26262-9:2011, Clause 7).

- a) During the course of allocation, the ASIL and information given in 8.4.2.3 shall be inherited from the associated safety goal or, if ASIL decomposition is applied, from the level above.
- b) If several functional safety requirements are allocated to the same architectural element, then the architectural element shall be developed in accordance with the highest ASIL for those safety requirements if independence or freedom from interference cannot be argued in the preliminary architecture.
- c) If the item comprises more than one system, then the functional safety requirements for the individual systems and their interfaces shall be specified, considering the preliminary architectural assumptions. These functional safety requirements shall be allocated to the systems.
- d) If ASIL decomposition is applied during the allocation of the functional safety requirements, then it shall be applied in accordance with ISO 26262-9:2011, Clause 5.

- 8.4.3.2** If the functional safety concept is to rely on elements of other technologies, then the following shall apply:

- a) The functional safety requirements implemented by elements of other technologies shall be derived and allocated to the corresponding elements of the architecture.
- b) The functional safety requirements relating to the interfaces with elements of other technologies shall be specified.
- c) The implementation of functional safety requirements by elements of other technologies shall be ensured through specific measures that are outside the scope of ISO 26262.
- d) No ASIL should be assigned to these elements.

NOTE The adequacy of elements of other technologies is shown during validation activities (see ISO 26262-4).

- 8.4.3.3** If the functional safety concept is to rely on external measures, then the following shall apply:

- a) The functional safety requirements implemented by external measures shall be derived and communicated.
- b) The functional safety requirements of interfaces with external measures shall be specified.
- c) If the external measures are implemented by one or more E/E systems, the functional safety requirements shall be addressed using ISO 26262.

d) The implementation of functional safety requirements by external measures shall be ensured.

NOTE The adequacy of external measures is shown during validation activities (see ISO 26262-4).

8.4.4 Validation criteria

8.4.4.1 The acceptance criteria for safety validation of the item shall be specified based on the functional safety requirements.

NOTE For further requirements on detailing the criteria and a list of characteristics to be validated, see ISO 26262-4:2011, 6.4.6.2 and 9.4.3.2.

8.4.5 Verification of the functional safety concept

8.4.5.1 The functional safety concept shall be verified in accordance with ISO 26262-8:2011, Clause 9, to show

- a) its consistency and compliance with the safety goals; and
- b) its ability to mitigate or avoid the hazardous events.

NOTE 1 The verification of the ability to mitigate or avoid a hazardous event during concept phase can be based on the same methods that are used for validation. The results of the evaluation can give an indication for concept improvements. However, it has to be kept in mind that the basis for safety validation in ISO 26262-4:2011, Clause 9, is an item developed according to ISO 26262 and safety validation cannot be based on concept studies (e.g. prototypes).

EXAMPLE The ability to mitigate or to avoid a hazardous event can be evaluated by tests, trials or expert judgement; with prototypes, studies, subject tests, or simulations.

NOTE 2 The verification of the ability to mitigate or to avoid a hazardous event addresses the characteristics of the fault (e.g. being transient or permanent).

NOTE 3 For verification, a traceability based argument can be used, i.e. if the item complies with the functional safety requirements, then the item complies with the safety goals as a result of this requirement.

8.5 Work products

8.5.1 Functional safety concept resulting from the requirements of 8.4.1 to 8.4.4.

8.5.2 Verification report of the functional safety concept resulting from the requirements of 8.4.5.

Annex A (informative)

Overview and document flow of concept phase

Table A.1 provides an overview of objectives, prerequisites and work products of the concept phase.

Table A.1 — Overview of concept phase

Clause	Objectives	Prerequisites	Work products
5 Item definition	The first objective is to define and describe the item, its dependencies on and interaction with the environment and other items. The second objective is to support an adequate understanding of the item so that the activities in subsequent phases can be performed.	None	5.5 Item definition
6 Initiation of the safety lifecycle	The first objective of the initiation of the safety lifecycle is to make the distinction between a new item development and a modification to a existing item (see ISO 26262-2:2011, Figure 2) The second objective is to define the safety lifecycle activities (see ISO 26262-2:2011, Figure 2) that will be carried out in the case of a modification.	Item definition	6.5.1 Impact analysis 6.5.2 Safety plan (refined)
7 Hazard analysis and risk assessment	The objective of the hazard analysis and risk assessment is to identify and to categorise the hazards that malfunctions in the item can trigger and to formulate the safety goals related to the prevention or mitigation of the hazardous events, in order to avoid unreasonable risk.	Item definition	7.5.1 Hazard analysis and risk assessment 7.5.2 Safety goals 7.5.3 Verification review report of the hazard analysis and risk assessment and the safety goals
8 Functional safety concept	The objective of the functional safety concept is to derive the functional safety requirements, from the safety goals, and to allocate them to the preliminary architectural elements of the item, or to external measures.	Item definition Hazard analysis and risk assessment Safety goals	8.5.1 Functional safety concept 8.5.2 Verification report of the functional safety concept

Annex B (informative)

Hazard analysis and risk assessment

B.1 General

This annex gives a general explanation of the hazard analysis and risk assessment. The examples in Clauses B.2 (severity), B.3 (probability of exposure) and B.4 (controllability) are for information only and are not exhaustive.

For this analytical approach, a risk (R) can be described as a function (F), with the frequency of occurrence (f) of a hazardous event, the ability to avoid specific harm or damage through timely reactions of the persons involved, that is the controllability (C) and the potential severity (S) of the resulting harm or damage:

$$R = F(f, C, S)$$

The frequency of occurrence f is, in turn, influenced by several factors. One factor to consider is how frequently and for how long individuals find themselves in a situation where the aforementioned hazardous event can occur. In ISO 26262 this is simplified to be a measure of the probability of the driving scenario taking place in which the hazardous event can occur (exposure, E). Another factor is the failure rate of the item that could lead to the hazardous event (failure rate, λ). The failure rate is characterised by hazardous hardware random failures and systematic faults that remained in the system:

$$f = E \times \lambda$$

Hazard analysis and risk assessment is concerned with setting requirements for the item such that unreasonable risk is avoided.

The ASILs that result from the hazard analysis and risk assessment determine the minimum set of requirements on the item, in order to control or reduce the probability of random hardware failures, and to avoid systematic faults. The failure rate of the item is not considered *a priori* (in the risk assessment) because an unreasonable residual risk is avoided through the implementation of the resulting safety requirements.

The hazard analysis and risk assessment subphase comprises three steps, as described below.

- a) Situation analysis and hazard identification (see 7.4.2): the goal of the situation analysis and hazard identification is to identify the potential unintended behaviours of the item that could lead to a hazardous event. The situation analysis and hazard identification activity requires a clear definition of the item, its functionality and its boundaries. It is based on the item's behaviour; therefore, the detailed design of the item does not necessarily need to be known.

EXAMPLE Factors to be considered for situation analysis and hazard identification can include:

- vehicle usage scenarios, for example high speed driving, urban driving, parking, off-road;
- environmental conditions, for example road surface friction, side winds;
- reasonably foreseeable driver use and misuse; and
- interaction between operational systems.

- b) Classification of hazardous events (see 7.4.3): the hazard classification scheme comprises the determination of the severity, the probability of exposure, and the controllability associated with the hazardous events of the item. The severity represents an estimate of the potential harm in a particular driving situation, while the probability of exposure is determined by the corresponding situation. The controllability rates how easy or difficult it is for the driver or other road traffic participant to avoid the considered accident type in the considered operational situation. For each hazard, depending on the number of related hazardous events, the classification will result in one or more combinations of severity, probability of exposure, and controllability.
- c) ASIL determination (see 7.4.4): determining the required automotive safety integrity level.

B.2 Examples of severity

B.2.1 General

The potential injuries as a result of a hazard are evaluated for the driver, passengers and people around the vehicle, or to individuals in surrounding vehicles to determine the severity class for a given hazard. From this evaluation, the corresponding severity class is then determined, for example, as shown in Table B.1.

Table B.1 presents examples of consequences which can occur for a given hazard, and the corresponding severity class for each consequence.

Because of the complexity of accidents and the many possible variations of accident situations, the examples provided in Table B.1 represent only an approximate estimate of accident effects. They represent expected values based on previous accident analyses. Therefore, no generally valid conclusions can be derived from these individual descriptions.

Accident statistics can be used to determine the distribution of injuries that can be expected to occur in different types of accidents.

In Table B.1, AIS represents a categorisation of injury classes, but only for single injuries. Instead of AIS, other categorisations such as Maximum AIS (MAIS) and Injury Severity Score (ISS) can be used.

The use of a specific injury scale depends on the state of medical research at the time the analysis is performed. Therefore, the appropriateness of the different injury scales, such as AIS, ISS, and NISS, can vary over time (see References [2], [4], [5]).

B.2.2 Description of the AIS stages

To describe the severity, the AIS classification is used. The AIS represents a classification of the severity of injuries and is issued by the Association for the Advancement of Automotive Medicine (AAAM). (See Reference [2]). The guidelines were created to enable an international comparison of severity. The scale is divided into seven classes:

- AIS 0: no injuries;
- AIS 1: light injuries such as skin-deep wounds, muscle pains, whiplash, etc.;
- AIS 2: moderate injuries such as deep flesh wounds, concussion with up to 15 minutes of unconsciousness, uncomplicated long bone fractures, uncomplicated rib fractures, etc.;
- AIS 3: severe but not life-threatening injuries such as skull fractures without brain injury, spinal dislocations below the fourth cervical vertebra without damage to the spinal cord, more than one fractured rib without paradoxical breathing, etc.;
- AIS 4: severe injuries (life-threatening, survival probable) such as concussion with or without skull fractures with up to 12 hours of unconsciousness, paradoxical breathing;

- AIS 5: critical injuries (life-threatening, survival uncertain) such as spinal fractures below the fourth cervical vertebra with damage to the spinal cord, intestinal tears, cardiac tears, more than 12 hours of unconsciousness including intracranial bleeding;
- AIS 6: extremely critical or fatal injuries such as fractures of the cervical vertebrae above the third cervical vertebra with damage to the spinal cord, extremely critical open wounds of body cavities (thoracic and abdominal cavities), etc.

Table B.1 — Examples of severity classification

	Class of severity (see Table 1)			
	S0	S1	S2	S3
Reference for single injuries (from AIS scale)	<ul style="list-style-type: none"> — AIS 0 and less than 10 % probability of AIS 1-6 — Damage that cannot be classified safety-related 	<p>More than 10 % probability of AIS 1-6 (and not S2 or S3)</p>	<p>More than 10 % probability of AIS 3-6 (and not S3)</p>	<p>More than 10 % probability of AIS 5-6</p>
Examples	<ul style="list-style-type: none"> — Bumps with roadside infrastructure — Pushing over roadside post, fence, etc. — Light collision — Light grazing damage — Damage entering/ exiting parking space — Leaving the road without collision or rollover 	<ul style="list-style-type: none"> — Side impact with a narrow stationary object, e.g. crashing into a tree (impact to passenger cell) with very low speed — Side collision with a passenger car (e.g. intrudes upon passenger compartment) with very low speed — Rear/front collision with another passenger car with very low speed — Collision with minimal vehicle overlap (10 % to 20 %) — Front collision (e.g. rear-ending another vehicle, semi-truck, etc.) without passenger compartment deformation 	<ul style="list-style-type: none"> — Side impact with a narrow stationary object, e.g. crashing into a tree (impact to passenger cell) with low speed — Side collision with a passenger car (e.g. intrudes upon passenger compartment) with low speed — Rear/front collision with another passenger car with low speed — Pedestrian/bicycle accident while turning (city intersection and streets) 	<ul style="list-style-type: none"> — Side impact with a narrow stationary object, e.g. crashing into a tree (impact to passenger cell) with medium speed — Side collision with a passenger car (e.g. intrudes upon passenger compartment) with medium speed — Rear/front collision with another passenger car with medium speed — Pedestrian/bicycle accident (e.g. 2-lane road) — Front collision (e.g. rear-ending another vehicle, semi-truck, etc.) with passenger compartment deformation

B.3 Examples and explanations of the probability of exposure

An estimation of the probability of exposure requires the evaluation of the scenarios in which the relevant environmental factors that contribute to the occurrence of the hazard are present. The scenarios to be evaluated include a wide range of driving or operating situations.

These evaluations result in the designation of the hazard scenarios into one of five probability of exposure classifications, given the nomenclature E0 (lowest exposure level), E1, E2, E3 and E4 (highest exposure level).

The first of these, E0, is assigned to situations which, although identified during a hazard and risk analysis, are considered to be unusual or incredible. Subsequent evaluation of the hazards associated exclusively with these E0 scenarios may be excluded from further analysis.

EXAMPLE Typical examples of E0 include the following:

- a) a very unusual, or infeasible, co-occurrence of circumstances, e.g.
 - a vehicle involved in an accident with another vehicle that is carrying a hazardous material (note this does not apply to a vehicle which is designed to carry that material);
 - a vehicle involved in an incident which includes an aeroplane landing on a highway;
- b) natural disasters, e.g. earthquake, hurricane, forest fire.

The remaining E1, E2, E3 and E4 levels are assigned for situations that can become hazardous depending on either the duration of a situation (temporal overlap) or the frequency of occurrence of a situation.

NOTE 1 The classification can depend on, for example, geographical location or type of use (see 7.4.3.4).

If the exposure is ranked based on the duration of a situation, the probability of exposure can be estimated by the proportion of time (of being in the considered situation) compared to the total operating time (ignition on). In special cases the total operating time can be the vehicle life-time (including ignition off). Table B.2 gives examples of these durational situation classifications and the typical exposure rankings.

NOTE 2 A hazard can be related to the duration of a given scenario (e.g. the average time spent negotiating traffic intersections), while another hazard can be related to the frequency of this same scenario (e.g. the rate of repetition with which a vehicle negotiates traffic intersections).

Alternatively, some exposure estimations can be determined more appropriately by using the frequency of occurrence of a related driving situation. In these circumstances, pre-existing system faults lead to the hazardous event within a short interval after the situation occurs. Examples of these driving situations and typical exposure rankings are given in Table B.3.

A driving situation may have both duration and a frequency, such as driving in a parking lot. In this case, the examples in Tables B.2 and B.3 might not lead to the same exposure category, so the most appropriate exposure rank is selected for the analysis of the considered driving scenario.

If the time period in which a failure remains latent is comparable to the time period before the hazardous event can be expected to take place, then the estimation of the probability of exposure considers that time period. Typically this will concern devices that are expected to act on demand, e.g. airbags.

In this case, the probability of exposure is estimated by $\sigma \times T$ where; σ is the rate of occurrence of the hazardous event and T is the duration that the failure is not perceived (possibly up to the lifetime of the vehicle). This approximation $\sigma \times T$ is valid when this resulting product is small.

NOTE 3 With regard to the duration of the considered failure, the hazard analysis and risk assessment does not consider safety mechanisms that are part of the item (see 7.4.1.2).

Table B.2 — Classes of probability of exposure regarding duration in operational situations

		Class of probability of exposure in operational situations (see Table 2)			
		E1	E2	E3	E4
Duration (% of average operating time)		Not specified	<1 % of average operating time	1 % to 10 % of average operating time	>10 % of average operating time
Examples	Road layout	—	<ul style="list-style-type: none"> — Mountain pass with unsecured steep slope — Country road intersection — Highway entrance ramp — Highway exit ramp 	<ul style="list-style-type: none"> — One-way street (city street) 	<ul style="list-style-type: none"> — Highway — Secondary road — Country road
	Road surface	—	<ul style="list-style-type: none"> — Snow and ice on road — Slippery leaves on road 	<ul style="list-style-type: none"> — Wet road 	—
	Nearby elements	<ul style="list-style-type: none"> — Lost cargo or obstacle in lane of travel (highway) 	<ul style="list-style-type: none"> — In car wash — Nearing end of congestion (highway) 	<ul style="list-style-type: none"> — In tunnel — Traffic congestion 	—
	Vehicle stationary state	<ul style="list-style-type: none"> — Vehicle during jump start — In repair garage (on roller rig) 	<ul style="list-style-type: none"> — Trailer attached — Roof rack attached — Vehicle being refuelled — In repair garage (during diagnosis or repair) — On hoist 	<ul style="list-style-type: none"> — Vehicle on a hill (hill hold) 	—
	Manoeuvre	—	<ul style="list-style-type: none"> — Driving in reverse (from parking spot) — Driving in reverse (city street) — Overtaking — Parking (with sleeping person in vehicle) — Parking (with trailer attached) 	<ul style="list-style-type: none"> — Heavy traffic (stop and go) 	<ul style="list-style-type: none"> — Accelerating — Decelerating — Executing a turn (steering) — Parking (parking lot) — Lane change (city street) — Stopping at traffic light (city street) — Lane change (highway)
	Visibility	—	—	<ul style="list-style-type: none"> — Unlighted roads at night 	—

Table B.3 — Classes of probability of exposure regarding frequency in operational situations

		Class of probability of exposure in operational situations (see Table 2)			
		E1	E2	E3	E4
Frequency of situation	Occurs less often than once a year for the great majority of drivers	Occurs a few times a year for the great majority of drivers	Occurs once a month or more often for an average driver	Occurs during almost every drive on average	
Examples	Road layout	—	— Mountain pass with unsecured steep slope	—	—
	Road surface	—	— Snow and ice on road	— Wet road	—
	Nearby elements	—	—	— In tunnel — In car wash — Traffic congestion	—
	Vehicle stationary state	— Stopped, requiring engine restart (at railway crossing) — Vehicle being towed — Vehicle during jump start	— Trailer attached — Roof rack attached	— Vehicle being refuelled — Vehicle on a hill (hill hold)	—
	Manoeuvre	—	— Evasive manoeuvre, deviating from desired path	— Overtaking	— Starting from standstill — Shifting transmission gears — Accelerating — Braking — Executing a turn (steering) — Using indicators — Manoeuvring vehicle into parking position — Driving in reverse

B.4 Examples of controllability (chances to avoid harm)

The determination of the controllability class, for a given hazard, requires an estimation of the probability that the representative driver will be able to retain or regain control of a vehicle if a given hazard were to occur.

This probability estimation involves the consideration of the likelihood that representative drivers will be able to retain or regain control of the vehicle if the hazard were to occur, or that individuals in the vicinity or the situation will contribute to the avoidance of the hazard by their actions. This consideration is based on assumptions about the control actions necessary by the individuals involved in the hazard scenario to retain or regain control of the situation, as well as the representative driving behaviours of the drivers involved (which can be related to the target market, individuals' age, eye-hand coordination, driving experience, cultural background, etc.).

NOTE Controllability estimations can be influenced by a number of factors including the cultural background of the analyst, the target market for the vehicle, or driver profiles for the target market.

To aid in these evaluations, Table B.4 provides examples of driving situations in which a malfunction is introduced, and the assumptions about the corresponding control behaviours that would avoid harm. These situations are mapped to the controllability rankings, clarifying the 90 % and 99 % breakpoint levels for judging participant controllability.

Table B.4 — Examples of possibly controllable hazardous events by the driver or by the persons potentially at risk

Driving factors and scenarios	Class of controllability (see Table 3)			
	C0	C1	C2	C3
Controllable in general	99 % or more of all drivers or other traffic participants are usually able to avoid harm	90 % or more of all drivers or other traffic participants are usually able to avoid harm	Less than 90 % of all drivers or other traffic participants are usually able, or barely able, to avoid harm	
Examples	Situations that are considered distracting	Maintain intended driving path	—	—
	Unexpected radio volume increase	Maintain intended driving path	—	—
	Warning message - gas low	Maintain intended driving path	—	—
	Unavailability of a driver assisting system	Maintain intended driving path	—	—
	Faulty adjustment of seat position while driving	—	Brake to slow/stop vehicle	—
	Blocked steering column when starting the vehicle	—	Brake to slow/stop vehicle	—
	Failure of ABS during emergency braking	—	—	Maintain intended driving path
	Headlights fail while night driving at medium/high speed on unlighted road	—	—	Steer to side of road or brake to stop.
	Motor failure at high lateral acceleration (motorway exit)	—	—	Maintain intended driving path
	Failure of ABS when braking on low friction road surface while executing a turn	—	—	—
	Failure of brakes	—	—	—
	Incorrect steering angle with high angular speed at medium or high vehicle speed (steering angle change not aligned to driver intent)	—	—	—
	Faulty driver airbag release when travelling at high speed	—	—	—

NOTE 1 For C2, a feasible test scenario in accordance with RESPONSE 3 (see Reference [3]) is accepted as adequate: "Practical testing experience revealed that a number of 20 valid data sets per scenario can supply a basic indication of validity". If each of the 20 data sets complies with the pass-criteria for the test, a level of controllability of 85 % (with a level of confidence of 95 % which is generally accepted for human factors tests) can be proven. This is appropriate evidence of the rationale for a C2-estimate.

NOTE 2 For C1 a test to provide a rationale that 99 % of the drivers "pass" the test in a certain traffic scenario might not be feasible because a huge number of test subjects would be necessary as the appropriate evidence for such a rationale.

NOTE 3 As no controllability is assumed for category C3, it is not relevant to have appropriate evidence of the rationale for such a classification.

Bibliography

- [1] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*
- [2] Abbreviated injury scale; Association of the advancement of Automotive medicine; Barrington, IL, USA
Information is also available at www.carcrash.org or <http://www.unfallforensik.de/>
- [3] Code of Practice for the design and evaluation of ADAS, EU Project RESPONSE 3: Oct. 2006
- [4] BAKER, S.P., O'NEILL, B., HADDON, W., LONG, W.B., The injury severity score: a method for describing patients with multiple injuries and evaluating emergency care, *The Journal of Trauma*, Vol. 14, No. 3, 1974
- [5] BALOGH, Z., OFFNER, P.J., MOORE, E.E., BIFFL, W.L., NISS predicts post injury multiple organ failure better than ISS, *The Journal of Trauma*, Vol. 48, No. 4, 2000

ICS 43.040.10

Price based on 25 pages

INTERNATIONAL
STANDARD

ISO
26262-4

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 4:
Product development at the system level**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 4: Développement du produit au niveau du système*



Reference number
ISO 26262-4:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	v
Introduction.....	vi
1 Scope.....	1
2 Normative references.....	2
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	3
4.3 ASIL-dependent requirements and recommendations	3
5 Initiation of product development at the system level	3
5.1 Objectives	3
5.2 General	4
5.3 Inputs to this clause.....	6
5.4 Requirements and recommendations	6
5.5 Work products	6
6 Specification of the technical safety requirements	7
6.1 Objectives	7
6.2 General	7
6.3 Inputs to this clause.....	7
6.4 Requirements and recommendations	7
6.5 Work products	10
7 System design	10
7.1 Objectives	10
7.2 General	11
7.3 Inputs to this clause.....	11
7.4 Requirements and recommendation	11
7.5 Work products	16
8 Item integration and testing	16
8.1 Objectives	16
8.2 General	16
8.3 Inputs to this clause.....	16
8.4 Requirements and recommendation	17
8.5 Work products	25
9 Safety validation	25
9.1 Objectives	25
9.2 General	25
9.3 Inputs to this clause.....	26
9.4 Requirements and recommendation	26
9.5 Work products	27
10 Functional safety assessment	28
10.1 Objectives	28
10.2 General	28
10.3 Inputs to this clause.....	28
10.4 Requirements and recommendation	28
10.5 Work products	28
11 Release for production	28

11.1	Objectives	28
11.2	General.....	29
11.3	Inputs to this clause	29
11.4	Requirements and recommendations	29
11.5	Work products.....	30
Annex A (informative) Overview and document flow of product development at the system level.....		31
Annex B (informative) Example contents of hardware-software interface.....		33
Bibliography		36

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-4 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

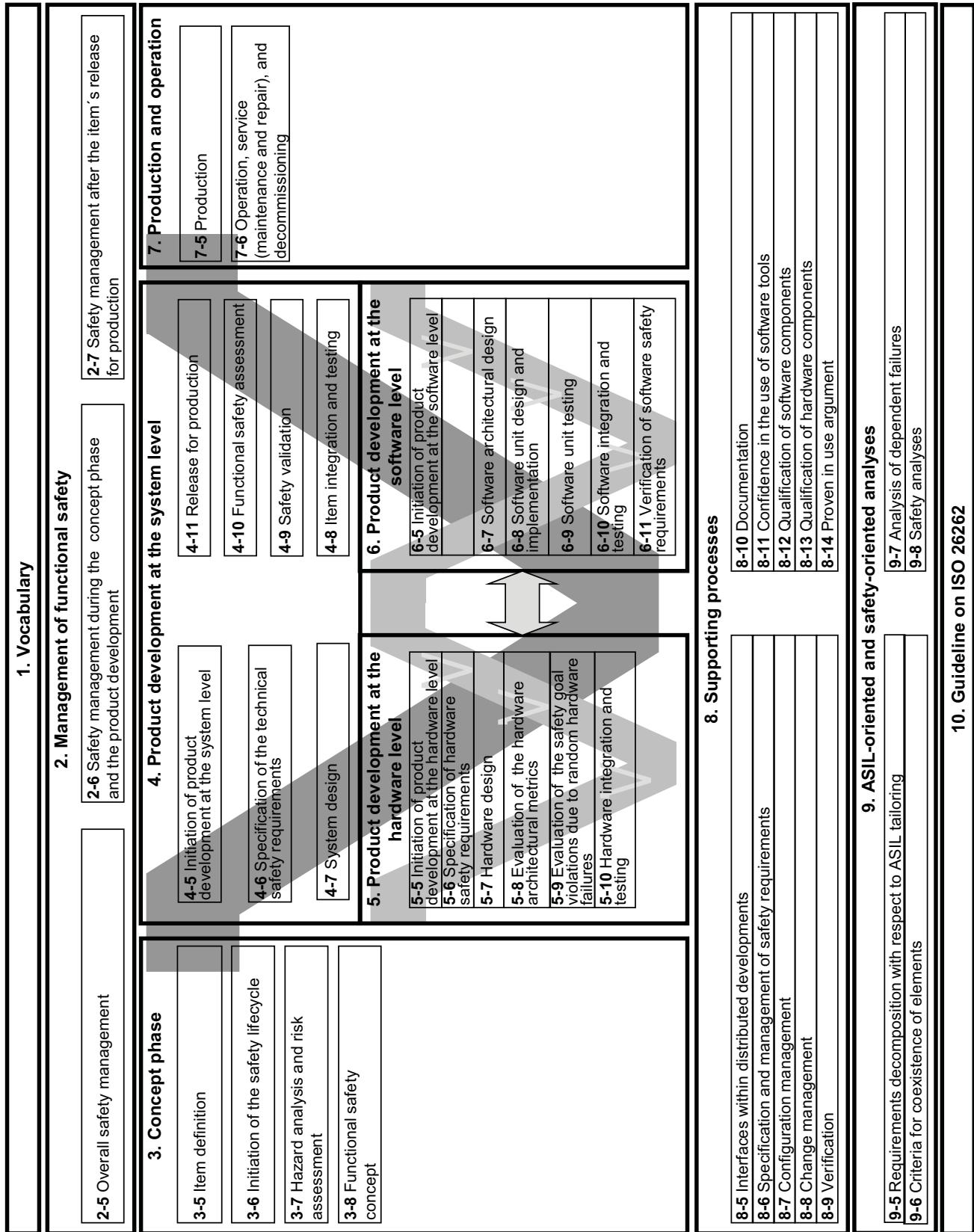


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 4: Product development at the system level

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for product development at the system level for automotive applications, including the following:

- requirements for the initiation of product development at the system level,
- specification of the technical safety requirements,
- the technical safety concept,
- system design,
- item integration and testing,
- safety validation,
- functional safety assessment, and
- product release.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-7:2011, *Road vehicles — Functional safety — Part 7: Production and operation*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- tailoring of the safety activities in accordance with ISO 26262-2 has been planned and shows that the requirement does not apply, or
- a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Initiation of product development at the system level

5.1 Objectives

The objective of the initiation of the product development at the system level is to determine and plan the functional safety activities during the individual subphases of system development. This also includes the necessary supporting processes described in ISO 26262-8.

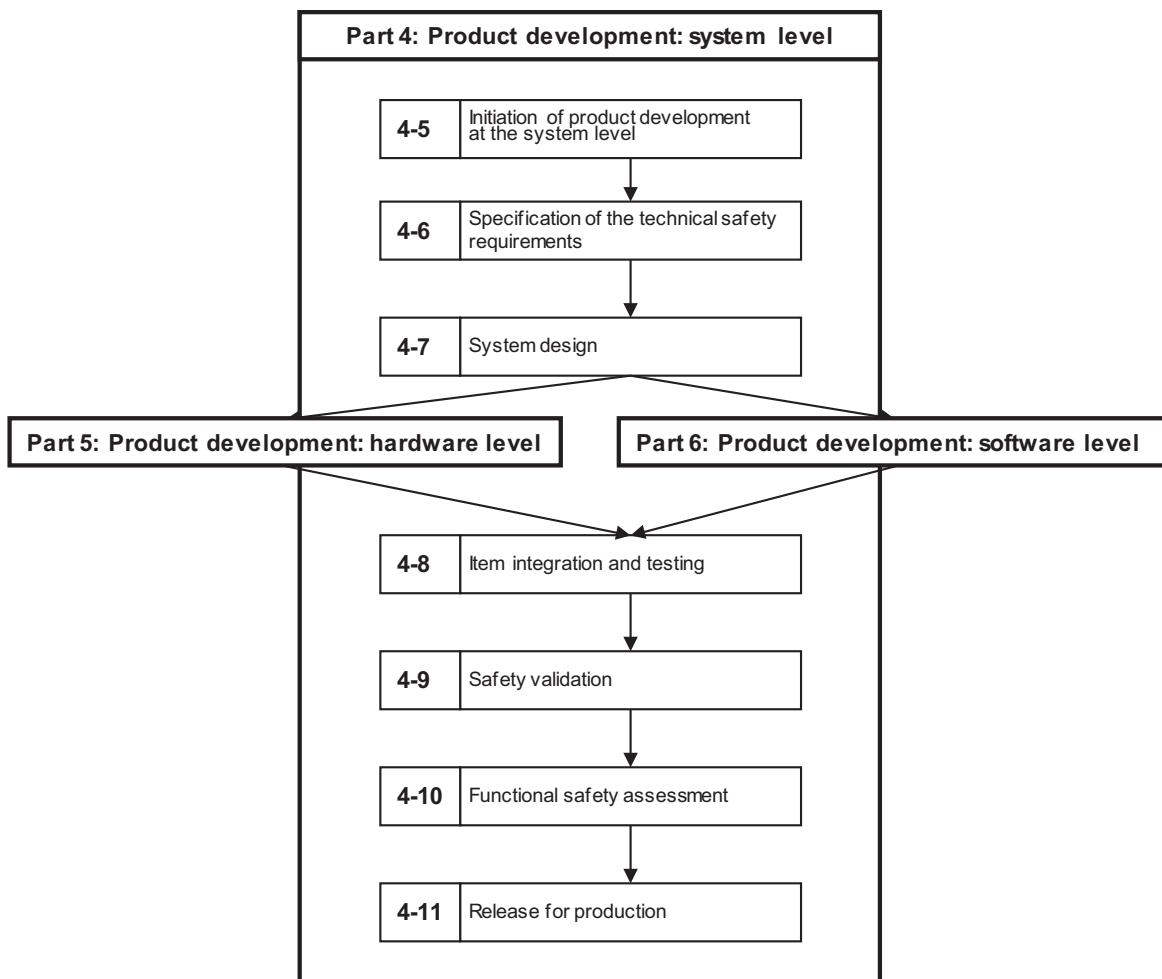
This planning of system-level safety activities will be included in the safety plan.

5.2 General

The necessary activities during the development of a system are given in Figure 2. After the initiation of product development and the specification of the technical safety requirements, the system design is performed. During system design the system architecture is established, the technical safety requirements are allocated to hardware and software, and, if applicable, on other technologies. In addition, the technical safety requirements are refined and requirements arising from the system architecture are added, including the hardware-software interface (HSI). Depending on the complexity of the architecture, the requirements for subsystems can be derived iteratively. After their development, the hardware and software elements are integrated and tested to form an item that is then integrated into a vehicle. Once integrated at the vehicle level, safety validation is performed to provide evidence of functional safety with respect to the safety goals.

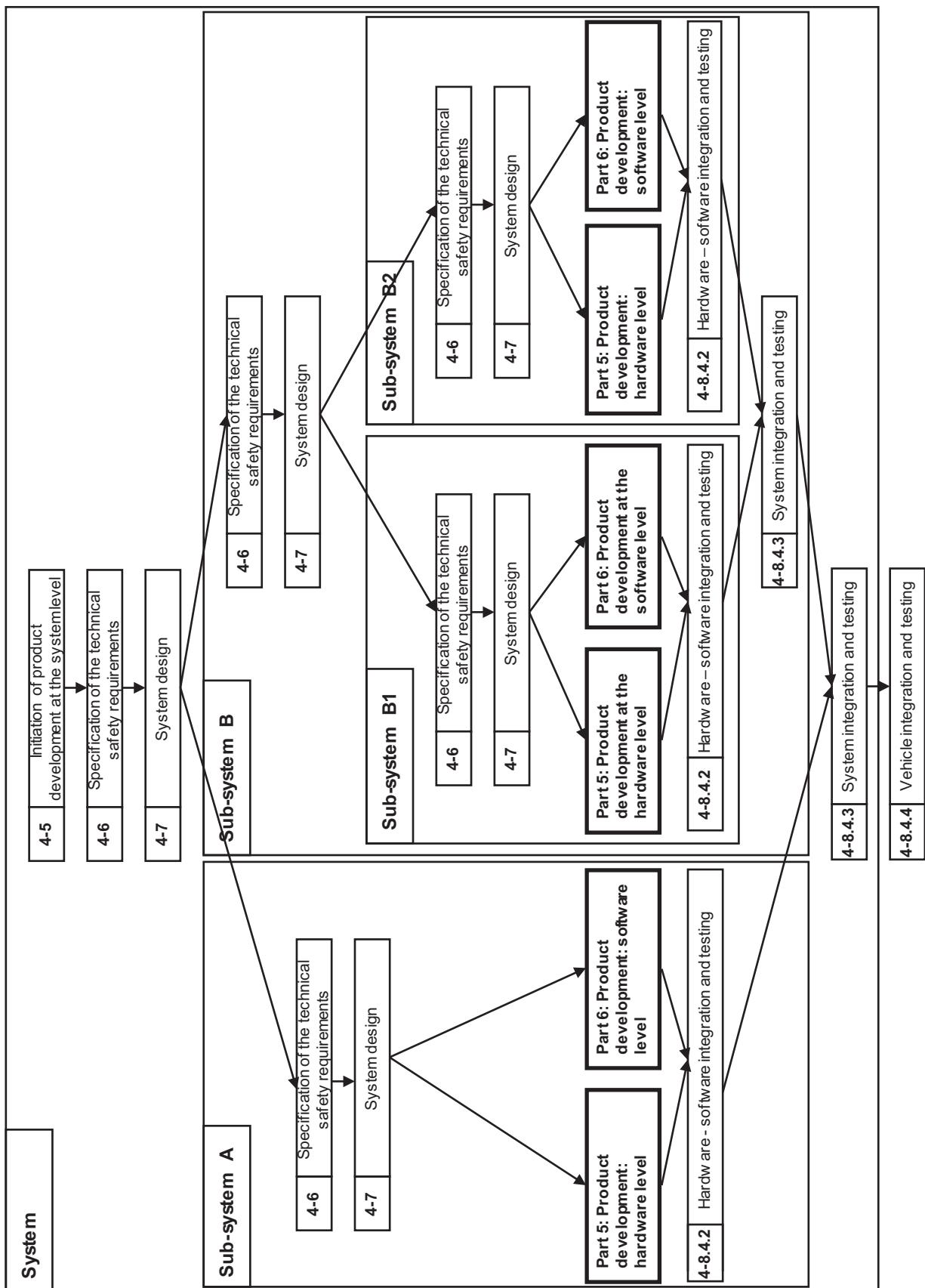
ISO 26262-5 and ISO 26262-6 describe the development requirements for hardware and software. This part of ISO 26262 applies to both the development of systems and subsystems. Figure 3 is an example of a system with multiple levels of integration, illustrating the application of this part of ISO 26262, ISO 26262-5 and ISO 26262-6.

NOTE 1 Table A.1 provides an overview of objectives, prerequisites and work products of the particular subphases of product development at the system level.



NOTE 2 Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: “m-n”, where “m” represents the number of the part and “n” indicates the number of the clause, e.g. “4-5” represents Clause 5 of ISO 26262-4.

Figure 2 — Reference phase model for the development of a safety-related item



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: "m-n", where "m" represents the number of the part and "n" indicates the number of the clause, e.g. "4-5" represents Clause 5 of ISO 26262-4.

Figure 3 — Example of a product development at the system level

5.3 Inputs to this clause

5.3.1 Prerequisites

The following information shall be available:

- project plan (refined) in accordance with ISO 26262-2:2011, 6.5.2;
- safety plan in accordance with ISO 26262-3:2011, 6.5.2;
- functional safety assessment plan in accordance with ISO 26262-2:2011, 6.5.4; and
- functional safety concept in accordance with ISO 26262-3:2011, 8.5.1.

5.3.2 Further supporting information

The following information can be considered:

- preliminary architectural assumptions (from external source); and
- item definition (see ISO 26262-3:2011, 5.5).

5.4 Requirements and recommendations

5.4.1 The safety activities for the product development at the system level shall be planned including determination of appropriate methods and measures during design and integration.

NOTE The results of planning of the verification activities during design in accordance with 6.4.6 (Verification and validation) and 7.4.8 (Verification of system design) are part of the safety plan while the planning of item integration and testing in accordance with 8.4.2 (hardware/software), 8.4.3 (element integration) and 8.4.4 (item integration) is represented in a separate item integration and testing plan in accordance with requirement 8.4.1.3.

5.4.2 The validation activities shall be planned.

5.4.3 The functional safety assessment activities for the product development at the system level shall be planned (see also ISO 26262-2).

NOTE An example of a functional safety assessment agenda is provided in ISO 26262-2:2011, Annex E.

5.4.4 The tailoring of the lifecycle for product development at system level shall be performed in accordance with ISO 26262-2, and based on the reference phase model given in Figure 2.

NOTE The project plan can be used to provide the relationship between the individual subphases of product development at the system level and the hardware and software development phases. This can include the integration steps at each level.

5.5 Work products

5.5.1 **Project plan (refined)** resulting from requirement 5.4.4.

5.5.2 **Safety plan (refined)** resulting from requirement 5.4.1 to 5.4.4.

5.5.3 **Item integration and testing plan** resulting from requirement 5.4.1.

5.5.4 **Validation plan** resulting from requirement 5.4.2.

5.5.5 **Functional safety assessment plan (refined)** resulting from requirement 5.4.3.

6 Specification of the technical safety requirements

6.1 Objectives

The first objective of this subphase is to specify the technical safety requirements. The technical safety requirements specification refines the functional safety concept, considering both the functional concept and the preliminary architectural assumptions (see ISO 26262-3).

The second objective is to verify through analysis that the technical safety requirements comply with the functional safety requirements.

6.2 General

Within the overall development lifecycle, the technical safety requirements are the technical requirements necessary to implement the functional safety concept, with the intention being to detail the item-level functional safety requirements into the system-level technical safety requirements.

NOTE Regarding the avoidance of latent faults, requirements elicitation can be performed after a first iteration of the system design subphase.

6.3 Inputs to this clause

6.3.1 Prerequisites

The following information shall be available:

- functional safety concept in accordance with ISO 26262-3:2011, 8.5.1; and
- validation plan in accordance with 5.5.4.

6.3.2 Further supporting information

The following information can be considered:

- safety goals (see ISO 26262-3:2011, 7.5.2);
- functional concept (from external source, see ISO 26262-3:2011, 5.4.1); and
- preliminary architectural assumptions (from external source, see ISO 26262-3:2011, 8.3.2).

6.4 Requirements and recommendations

6.4.1 Specification of the technical safety requirements

6.4.1.1 The technical safety requirements shall be specified in accordance with the functional safety concept, the preliminary architectural assumptions of the item and the following system properties:

- a) the external interfaces, such as communication and user interfaces, if applicable;
- b) the constraints, e.g. environmental conditions or functional constraints; and
- c) the system configuration requirements.

NOTE The ability to reconfigure a system for alternative applications is a strategy to reuse existing systems.

EXAMPLE Calibration data (see ISO 26262-6:2011, Annex C) is frequently used to customise electronic engine control units for alternate vehicles.

6.4.1.2 The consistency of the preliminary architectural assumptions in ISO 26262-3:2011, 8.3.2 and the preliminary architecture assumptions in this subphase shall be ensured.

6.4.1.3 If other functions or requirements are implemented by the system or its elements, in addition to those functions for which technical safety requirements are specified in accordance with 6.4.1 (Specification of the technical safety requirements), then these functions or requirements shall be specified or references made to their specification.

EXAMPLE Other requirements are coming from Economic Commission for Europe (ECE) rules, Federal Motor Vehicle Safety Standard (FMVSS) or company platform strategies.

6.4.1.4 The technical safety requirements shall specify safety-related dependencies between systems or item elements and between the item and other systems.

6.4.2 Safety mechanisms

6.4.2.1 The technical safety requirements shall specify the response of the system or elements to stimuli that affect the achievement of safety goals. This includes failures and relevant combinations of stimuli in combination with each relevant operating mode and defined system state.

EXAMPLE The Adaptive Cruise Control (ACC) ECU disables the ACC functionality if informed by the brake system ECU that the Vehicle Stability Control functionality is unavailable.

6.4.2.2 The technical safety requirements shall specify the necessary safety mechanisms (see ISO 26262-8:2011, Clause 6) including:

- a) the measures relating to the detection, indication and control of faults in the system itself;

NOTE 1 This includes the self-monitoring of the system or elements to detect random hardware faults and, if appropriate, to detect systematic failures.

NOTE 2 This includes measures for the detection and control of failure modes of the communication channels (e.g. data interfaces, communication buses, wireless radio link).

- b) the measures relating to the detection, indication and control of faults in external devices that interact with the system;

EXAMPLE External devices include other electronic control units, power supply or communication devices.

- c) the measures that enable the system to achieve or maintain a safe state;

NOTE 3 This includes prioritization and arbitration logic in the case of conflicting safety mechanisms.

- d) the measures to detail and implement the warning and degradation concept; and

- e) the measures which prevent faults from being latent [see 6.4.4 (Avoidance of latent faults)].

NOTE 4 These measures are usually related to tests that take place during power up (pre-drive checks), as in the case of measures a) to d), during operation, during power-down (post-drive checks), and as part of maintenance.

6.4.2.3 For each safety mechanism that enables an item to achieve or maintain a safe state the following shall be specified:

- a) the transition to the safe state;

NOTE 1 This includes the requirements to control the actuators.

- b) the fault tolerant time interval;

NOTE 2 In-vehicle testing and experimentation can be used to determine the fault tolerant time interval.

- c) the emergency operation interval, if the safe state cannot be reached immediately; and

NOTE 3 In-vehicle testing and experimentation can be used to determine the emergency operation interval.

EXAMPLE 1 Switching off can be an emergency operation.

- d) the measures to maintain the safe state.

EXAMPLE 2 A safety mechanism for a brake-by-wire application, which depends on the power supply, can include the specification of a secondary power supply or storage device (capacity, time to activate and operate, etc.).

6.4.3 ASIL Decomposition

6.4.3.1 If ASIL decomposition is applied during the specification of the technical safety requirements it shall be applied in accordance with ISO 26262-9:2011, Clause 5 (Requirements decomposition with respect to ASIL tailoring).

6.4.4 Avoidance of latent faults

6.4.4.1 This requirement applies to ASILs (A), (B), C, and D, in accordance with 4.3: if applicable, safety mechanisms shall be specified to prevent faults from being latent.

NOTE 1 Concerning random faults, only multiple-point faults have the potential to include latent faults.

EXAMPLE On-board tests are safety mechanisms which verify the status of components during the different operation modes such as power-up, power-down, at runtime or in an additional test mode to detect latent faults. Valve, relay or lamp function tests that take place during power up routines are examples of such on-board tests.

NOTE 2 Evaluation criteria that identify the need for safety measures preventing faults from being latent are derived in accordance with good engineering practice. The latent fault metric, given in ISO 26262-5:2011, Clause 8, provides evaluation criteria.

6.4.4.2 This requirement applies to ASILs (A), (B), C, and D, in accordance with 4.3: to avoid multiple-point failures, the multiple-point fault detection interval shall be specified for each safety mechanism implemented in accordance with 6.4.4 (Avoidance of latent faults).

6.4.4.3 This requirement applies to ASILs (A), (B), C, and D, in accordance with 4.3: to determine the multiple-point fault detection interval, the following parameters should be considered:

- a) the reliability of the hardware component with consideration given to its role in the architecture;
- b) the probability of exposure of the corresponding hazardous event(s);
- c) the specified quantitative target values for the maximum probability of violation of each safety goal due to hardware random failures (see requirement 7.4.4.3); and
- d) the assigned ASIL of the related safety goal.

NOTE The use of the following measures depends on the time constraints:

- periodic testing of the system or elements during operation;
- on board tests of elements during power-up or power-down; and
- testing the system or elements during maintenance.

6.4.4.4 This requirement applies to ASILs (A), (B), C, and D, in accordance with 4.3: the development of safety mechanisms that prevent dual point faults from being latent shall comply with:

- a) ASIL B for technical safety requirements assigned ASIL D;
- b) ASIL A for technical safety requirements assigned ASIL B and ASIL C; and
- c) engineering judgement for technical safety requirements assigned ASIL A.

6.4.5 Production, operation, maintenance and decommissioning

6.4.5.1 The technical safety requirements concerning functional safety of the item or its elements during production, operation, maintenance, repair and decommissioning, addressed in ISO 26262-7, shall be specified.

NOTE There are two aspects that assure safety during production, operation, maintenance, repair and decommissioning. The first aspect relates to those activities performed during the development phase which are given in requirement 6.4.5.1 and 7.4.7 (Requirements for production, operation, service and decommissioning), while the second aspect relates to those activities performed during the production and operation phase, which are addressed in ISO 26262-7.

6.4.6 Verification and validation

6.4.6.1 The technical safety requirements shall be verified in accordance with ISO 26262-8:2011, Clause 9, to provide evidence for their:

- a) compliance and consistency with the functional safety concept; and
- b) compliance with the preliminary architectural design assumptions.

6.4.6.2 The criteria for safety validation of the item shall be refined based on the technical safety requirements.

NOTE The system validation planning and the system validation specifications are developed in parallel with the technical safety requirements (see Clause 9).

6.5 Work products

6.5.1 Technical safety requirements specification resulting from requirements 6.4.1 to 6.4.5.

6.5.2 System verification report resulting from requirement 6.4.6.

6.5.3 Validation plan (refined) resulting from requirement 6.4.6.2.

7 System design

7.1 Objectives

The first objective of this subphase is to develop the system design and the technical safety concept that comply with the functional requirements and the technical safety requirements specification of the item.

The second objective of this subphase is to verify that the system design and the technical safety concept comply with the technical safety requirements specification.

7.2 General

The development of the system design and the technical safety concept is based on the technical safety requirements specification derived from the functional safety concept. This subphase can be applied iteratively, if the system is comprised of subsystems.

In order to develop a system architectural design, functional safety requirements, technical safety requirements and non-safety-related requirements are implemented. Hence in this subphase safety-related and non-safety-related requirements are handled within one development process.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- item integration and testing plan in accordance with 5.5.3; and
- technical safety requirements specification in accordance with 6.5.1.

7.3.2 Further supporting information

The following information can be considered:

- preliminary architectural assumptions (from external source, see ISO 26262-3:2011, 8.3.2);
- functional concept (from external source); and
- functional safety concept (see ISO 26262-3:2011, 8.5.1).

7.4 Requirements and recommendation

7.4.1 System design specification and technical safety concept

7.4.1.1 The system design shall be based on the functional concept, the preliminary architectural assumptions and the technical safety requirements. The consistency of the preliminary architectural assumptions in ISO 26262-3:2011, 8.3.2 and the preliminary architectural assumptions in this subphase shall be ensured.

7.4.1.2 The technical safety requirements shall be allocated to the system design elements.

7.4.1.3 The system design shall implement the technical safety requirements.

7.4.1.4 With regard to the implementation of the technical safety requirements the following shall be considered in the system design:

- a) the ability to verify the system design;
- b) the technical capability of the intended hardware and software design with regard to the achievement of functional safety; and
- c) the ability to execute tests during system integration.

7.4.2 System architectural design constraints

7.4.2.1 The system and subsystem architecture shall comply with the technical safety requirements at their respective ASILs.

7.4.2.2 Each element shall inherit the highest ASIL from the technical safety requirements that it implements.

7.4.2.3 If an element is comprised of sub-elements with different ASILs assigned, or of non-safety-related sub-elements and safety-related sub-elements, then each of these shall be treated in accordance with the highest ASIL, unless the criteria for coexistence, in accordance with ISO 26262-9:2011, Clause 6, are met.

7.4.2.4 Internal and external interfaces of safety-related elements shall be defined, in order to avoid other elements having adverse safety-related effects on the safety-related elements.

7.4.2.5 If ASIL decomposition is applied to the safety requirements during system design, it shall be applied in accordance with ISO 26262-9:2011, Clause 5.

7.4.3 Measures for the avoidance of systematic failures

7.4.3.1 Safety analyses on the system design to identify the causes of systematic failures and the effects of systematic faults shall be applied in accordance with Table 1 and ISO 26262-9:2011, Clause 8.

Table 1 — System design analysis

Methods	ASIL			
	A	B	C	D
1 Deductive analysis ^a	o	+	++	++
2 Inductive analysis ^b	++	++	++	++

^a Deductive analysis methods include FTA, reliability block diagrams, Ishikawa diagram.
^b Inductive analysis methods include FMEA, ETA, Markov modelling.

NOTE 1 The purpose of these analyses is to assist in the design. Therefore at this stage, qualitative analysis is likely to be sufficient. Quantitative analysis can be performed if necessary.

NOTE 2 The analysis is conducted at the level of detail necessary to identify or exclude causes and effects of systematic failures.

7.4.3.2 Identified internal causes of systematic failures shall be eliminated or their effects mitigated.

7.4.3.3 Identified external causes of systematic failures shall be eliminated or their effects mitigated.

7.4.3.4 To reduce systematic failures, well-trusted automotive systems design principles should be applied. These may include the following:

- a) re-use of well-trusted technical safety concepts;
- b) re-use of well-trusted designs for elements, including hardware and software components;
- c) re-use of well-trusted mechanisms for the detection and control of failures; and
- d) re-use of well-trusted or standardised interfaces.

7.4.3.5 To ensure the suitability of well-trusted design principles or elements in the new item, the results of their application shall be analysed and the underlying assumptions checked before reuse.

NOTE The impact analysis includes the capability and feasibility of the determined diagnostics, environmental constraints, timing constraints, compatibility of the determined resources, and the robustness of the system design.

7.4.3.6 This requirement applies to ASIL D: a decision not to re-use well-trusted design principles should be justified.

7.4.3.7 This requirement applies to ASILs (A), (B), C, and D, in accordance with 4.3: in order to avoid failures resulting from high complexity, the architectural design shall exhibit all of the following properties by use of the principles in Table 2:

- a) modularity;
- b) adequate level of granularity; and
- c) simplicity.

Table 2 — Properties of modular system design

	Properties	ASIL			
		A	B	C	D
1	Hierarchical design	+	+	++	++
2	Precisely defined interfaces	+	+	+	+
3	Avoidance of unnecessary complexity of hardware components and software components	+	+	+	+
4	Avoidance of unnecessary complexity of interfaces	+	+	+	+
5	Maintainability during service	+	+	+	+
6	Testability during development and operation	+	+	++	++

7.4.4 Measures for control of random hardware failures during operation

7.4.4.1 Measures for detection and control, or mitigation of random hardware failures shall be specified with respect to the system design given in 7.4.1 (System design specification and technical safety concept).

EXAMPLE 1 Such measures can be hardware diagnostic features and their usage by the software to detect random hardware failures.

EXAMPLE 2 A hardware design which directly leads to the safe state in the case of a random hardware failure controls a failure even without detection.

7.4.4.2 This requirement applies to ASILs (B), C, and D, in accordance with 4.3: the target values for single-point fault metric and latent-point fault metric (see ISO 26262-5:2011, Clause 8), shall be specified for final evaluation at the item level (see requirement 9.4.3.3).

7.4.4.3 This requirement applies to ASILs (B), C, and D, in accordance with 4.3: one of the alternative procedures of evaluation of violation of the safety goal due to random hardware failures (see ISO 26262-5:2011, Clause 9) shall be chosen and the target values shall be specified for final evaluation at item level (see requirement 9.4.3.3).

7.4.4.4 This requirement applies to ASILs (B), C, and D, in accordance with 4.3: appropriate target values for failure rates and diagnostic coverage should be specified at element level in order to comply with:

- a) the target values of the metrics in ISO 26262-5:2011, Clause 8; and
- b) the procedures in ISO 26262-5:2011, Clause 9.

7.4.4.5 This requirement applies to ASILs (B), C, and D, in accordance with 4.3: for distributed developments (see ISO 26262-8:2011, Clause 5), the derived target values shall be communicated to each relevant party.

NOTE Architectural constraints described in ISO 26262-5:2011, Clauses 8 and 9, are not directly applicable to COTS parts and components. This is because suppliers usually cannot foresee the usage of their products in the end-item and the potential safety implications. In such a case, basic data such as failure rate, failure modes, failure rate distribution per failure modes, built-in diagnosis, etc. are made available by the part supplier in order to allow the estimation of architectural constraints at overall hardware architecture level.

7.4.5 Allocation to hardware and software

7.4.5.1 The technical safety requirements shall be allocated directly or by further refinement to hardware, software or both.

7.4.5.2 If technical safety requirements are allocated to custom hardware elements that incorporate programmable behaviour (such as ASICs, FPGA or other forms of digital hardware) an adequate development process, combining requirements from ISO 26262-5 and ISO 26262-6, should be defined and implemented.

NOTE The evidence of compliance with an allocated safety requirement for some of those hardware elements can be provided through qualification measures in accordance with ISO 26262-8:2011, Clause 13, if the criteria for applying this clause are met.

7.4.5.3 The system design shall comply with the allocation and partitioning decisions.

NOTE To achieve independence and to avoid propagation of failures, the system design can implement the partitioning of functions and components.

7.4.6 Hardware-software interface specification (HSI)

7.4.6.1 The HSI specification shall specify the hardware and software interaction and be consistent with the technical safety concept. The HSI specification shall include the component's hardware devices that are controlled by software and hardware resources that support the execution of software.

EXAMPLE The aspects and characteristics detailed in the HSI are given in Annex B.

7.4.6.2 The HSI specification shall include the following characteristics:

- a) the relevant operating modes of hardware devices and the relevant configuration parameters;

EXAMPLE 1 Operating modes of hardware devices such as: default, init, test or advanced modes.

EXAMPLE 2 Configuration parameters such as: gain control, band pass frequency or clock prescaler.

- b) the hardware features that ensure the independence between elements and that support software partitioning;

- c) shared and exclusive use of hardware resources;

EXAMPLE 3 Memory mapping, allocation of registers, timers, interrupts, I/O ports.

- d) the access mechanism to hardware devices; and

EXAMPLE 4 Serial, parallel, slave, master/slave.

- e) the timing constraints defined for each service involved in the technical safety concept.

7.4.6.3 The relevant diagnostic capabilities of the hardware and their use by the software shall be specified in the HSI specification:

- a) the hardware diagnostic features shall be defined; and

EXAMPLE Detection of over-current, short-circuit or over-temperature.

- b) the diagnostic features concerning the hardware, to be implemented in software, shall be defined.

7.4.6.4 The HSI shall be specified during the system design and will be refined during hardware development (see ISO 26262-5:2011, Clause 7) and during software development (see ISO 26262-6:2011, Clause 7).

7.4.7 Requirements for production, operation, service and decommissioning

7.4.7.1 Diagnostic features shall be specified to provide the required data that enables field monitoring for the item or its elements during operation, with consideration being given to the results of safety analyses and the implemented safety mechanisms.

7.4.7.2 To maintain functional safety, diagnostic features shall be specified that allow fault identification by workshop staff when servicing is needed.

7.4.7.3 The requirements for production, operation, service and decommissioning, identified during the system design, shall be specified (see ISO 26262-7). These include:

- a) the assembly instructions requirements;
- b) the safety-related special characteristics;
- c) the requirements dedicated to ensure proper identification of systems or elements;

EXAMPLE 1 Labelling of elements.

- d) the verification methods and measures for production;
- e) the service requirements including diagnostic data and service notes; and
- f) the decommissioning requirements.

EXAMPLE 2 Decommissioning instructions.

7.4.8 Verification of system design

7.4.8.1 The system design shall be verified for compliance and completeness with regard to the technical safety concept using the verification methods listed in Table 3.

Table 3 — System design verification

Methods		ASIL			
		A	B	C	D
1a	System design inspection ^a	+	++	++	++
1b	System design walkthrough ^a	++	+	o	o
2a	Simulation ^b	+	+	++	++
2b	System prototyping and vehicle tests ^b	+	+	++	++
3	System design analyses ^c	see Table 1			

^a Methods 1a and 1b serve as a check of complete and correct implementation of the technical safety requirements.

^b Methods 2a and 2b can be used advantageously as a fault injection technique.

^c For conducting safety analyses, see ISO 26262-9:2011, Clause 8.

NOTE Anomalies and incompleteness identified between the system design, regarding the technical safety concept, will be reported in accordance with ISO 26262-2:2011, 5.4.2.

7.4.8.2 Newly identified hazards by the system design not covered in a safety goal shall be introduced and evaluated in the hazard analysis and risk assessment in accordance with ISO 26262-3 and the change management process in ISO 26262-8:2011, Clause 8.

NOTE Newly identified hazards, not already reflected in a safety goal, are usually non-functional hazards. Non-functional hazards are outside the scope of ISO 26262, but they can be annotated in the hazard analysis and risk assessment with the following statement “No ASIL is assigned to this hazard as it is not within the scope of ISO 26262”. However, an ASIL might be assigned for reference purpose.

7.5 Work products

- 7.5.1 Technical safety concept** resulting from requirements 7.4.1 and 7.4.5.
- 7.5.2 System design specification** resulting from requirements 7.4.1 to 7.4.5.
- 7.5.3 Hardware-software interface specification (HSI)** resulting from requirements 7.4.6.
- 7.5.4 Specification of requirements for production, operation, service and decommissioning** resulting from requirements 7.4.7.
- 7.5.5 System verification report (refined)** resulting from requirement 7.4.8.
- 7.5.6 Safety analysis reports** resulting from requirement 7.4.3.

8 Item integration and testing

8.1 Objectives

The integration and testing phase comprises three phases and two primary goals as described below: the first phase is the integration of the hardware and software of each element that the item comprises. The second phase is the integration of the elements that comprise an item to form a complete system. The third phase is the integration of the item with other systems within a vehicle and with the vehicle itself.

The first objective of the integration process is to test compliance with each safety requirement in accordance with its specification and ASIL classification.

The second objective is to verify that the “System design” covering the safety requirements [see Clause 7 (System design)] are correctly implemented by the entire item.

8.2 General

The integration of the item's elements is carried out in a systematic way starting from software-hardware integration through system integration to vehicle integration. Specified integration tests are performed at each integration stage to provide evidence that the integrated elements interact correctly.

After sufficient completion of hardware and software development in accordance with ISO 26262-5 and ISO 26262-6, the system integration in accordance with Clause 8 (Item integration and testing) can start.

8.3 Inputs to this clause

8.3.1 Prerequisites

The following information shall be available:

- safety goals in accordance with ISO 26262-3:2011, 7.5.2;
- functional safety concept in accordance with ISO 26262-3:2011, 8.5.1;
- item integration and testing plan in accordance with 5.5.3;

- technical safety concept in accordance with 7.5.1;
- system design specification in accordance with 7.5.2; and
- hardware-software interface specification (HSI) in accordance with 7.5.6.

8.3.2 Further supporting information

The following information can be considered:

- vehicle architecture (from external source);
- technical safety concepts of other vehicle systems (from external source); and
- safety analysis reports (see 7.5.6).

8.4 Requirements and recommendation

8.4.1 Planning and specification of integration and testing

8.4.1.1 To demonstrate that the system design is compliant with the functional and technical safety requirements, integration testing activities shall be performed in accordance with ISO 26262-8:2011, Clause 9.

NOTE The following test goals are addressed in Tables 4 to 18:

- a) the correct implementation of functional safety and technical safety requirements;
- b) the correct functional performance, accuracy and timing of safety mechanisms;
- c) the consistent and correct implementation of interfaces;
- d) the effectiveness of a safety mechanism's diagnostic or failure coverage; and
- e) the level of robustness.

8.4.1.2 An integration and test strategy shall be defined, which is based on the system design specification, the functional safety concept, the technical safety concept and the item integration and testing plan and provides evidence that the test goals are covered sufficiently. The integration and test strategy shall cover both E/E elements and elements of other technologies considered in the safety concepts.

NOTE Usually the integration levels are HW/SW, System, and Vehicle level.

8.4.1.3 To enable the system integration subphase the following shall be performed:

- a) the integration and testing plan shall be refined for the hardware-software integration and testing;
- b) the item integration and testing plan shall be refined to include the specification of integration tests for the system and vehicle levels. It shall ensure that open issues from hardware-software verifications are addressed; and
- c) the system and vehicle level item integration and testing plan shall consider interfaces between vehicle sub-systems (internal and external concerning the item) and the environment.

NOTE 1 When planning the vehicle level integration and testing, the correct vehicle behaviour under typical and extreme vehicle conditions and environments can be considered, but with a subset being sufficient (see Table 4).

NOTE 2 Integration and testing planning, carried out at the hardware-software integration and item levels, considers the interface and interaction between hardware and software.

8.4.1.4 If the system uses configurations or calibration data the verification at the system or vehicle level shall provide evidence of compliance with safety requirements for each configuration at implementation level or for every configuration that is intended for serial production.

NOTE If a complete verification of each configuration at the system or vehicle level is not feasible, then a reasonable subset might be selected.

8.4.1.5 The test equipment shall be subject to the control of a monitoring quality system.

8.4.1.6 Each functional and technical safety requirement shall be verified (if applicable by testing) at least once in the complete integration subphase.

NOTE 1 A common practice is to verify a safety requirement at the next higher level of integration to which it has been specified.

NOTE 2 Safety anomalies identified during integration testing are reported in accordance with ISO 26262-2:2011, 5.4.2.

8.4.1.7 To enable the appropriate specification of test cases for the integration tests, test cases shall be derived using an appropriate combination of methods, as listed in Table 4, and by considering the integration level.

Table 4 — Methods for deriving test cases for integration testing

	Methods	ASIL			
		A	B	C	D
1a	Analysis of requirements	++	++	++	++
1b	Analysis of external and internal interfaces	+	++	++	++
1c	Generation and analysis of equivalence classes for hardware-software integration	+	+	++	++
1d	Analysis of boundary values	+	+	++	++
1e	Error guessing based on knowledge or experience	+	+	++	++
1f	Analysis of functional dependencies	+	+	++	++
1g	Analysis of common limit conditions, sequences, and sources of dependent failures	+	+	++	++
1h	Analysis of environmental conditions and operational use cases	+	++	++	++
1i	Analysis of field experience	+	++	++	++

8.4.2 Hardware-software integration and testing

8.4.2.1 Hardware-software integration

8.4.2.1.1 The hardware developed in accordance with ISO 26262-5 and the software developed in accordance with ISO 26262-6 shall be integrated to be used as the subject of the test activities in Tables 4 to 8.

8.4.2.1.2 This requirement applies to ASILs C, and D, in accordance with 4.3: the hardware-software interface (HSI) requirements shall be tested with appropriate coverage, with consideration to the ASIL or a rationale shall be given that no issues with respect to the HSI remain.

NOTE The use of production-intent hardware and software is preferred. Modified hardware or software might be used where necessary for particular test techniques.

8.4.2.2 Test goals and test methods during hardware-software testing

8.4.2.2.1 To detect systematic faults, present in the system design, during hardware-software integration, the test goals resulting from the requirements 8.4.2.2.2 to 8.4.2.2.6 shall be addressed by the application of adequate test methods, as given in the corresponding tables.

NOTE Depending on the function implemented, its complexity or the distributed nature of the system, it can be reasonable to move test methods to other integration subphases with adequate rationale.

8.4.2.2.2 The correct implementation of the technical safety requirements at the hardware-software level shall be demonstrated using feasible test methods given in Table 5.

Table 5 — Correct implementation of technical safety requirements at the hardware-software level

	Methods	ASIL			
		A	B	C	D
1a	Requirements-based test ^a	++	++	++	++
1b	Fault injection test ^b	+	++	++	++
1c	Back-to-back test ^c	+	+	++	++

^a A requirements-based test denotes a test against functional and non-functional requirements.

^b A fault injection test uses special means to introduce faults into the test object during runtime. This can be done within the software via a special test interface or specially prepared hardware. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.

^c A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation.

NOTE The differences in the level of effort applied for Clause 1b in Table 5 and Table 10 result from the amount of efforts to conduct fault injection tests at system level.

8.4.2.2.3 This requirement applies to ASIL (A), B, C, and D, in accordance with 4.3: the correct functional performance, accuracy and timing of the safety mechanisms at the hardware-software level shall be demonstrated using feasible test methods given in Table 6.

Table 6 — Correct functional performance, accuracy and timing of safety mechanisms at the hardware-software level

	Methods	ASIL			
		A	B	C	D
1a	Back-to-back test ^a	+	+	++	++
1b	Performance test ^b	+	++	++	++

^a A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation.

^b A performance test can verify the performance (e.g. task scheduling, timing, power output) in the context of the whole test object, and can verify the ability of the intended control software to run with the hardware.

8.4.2.2.4 This requirement applies to ASIL (A), B, C, and D, in accordance with 4.3: the consistent and correct implementation of the external and internal interfaces at the hardware-software level shall be demonstrated using feasible test methods given in Table 7.

Table 7 — Consistent and correct implementation of external and internal interfaces at the hardware-software level

Methods		ASIL			
		A	B	C	D
1a	Test of external interfaces ^a	+	++	++	++
1b	Test of internal interfaces ^a	+	++	++	++
1c	Interface consistency check ^a	+	++	++	++

^a Interface tests of the test object include tests of analogue and digital inputs and outputs, boundary tests and equivalence-class tests to completely test the specified interfaces, compatibility, timings and other specified ratings for the test object. Internal interfaces of an ECU can be tested by static tests for the compatibility of software and hardware as well as dynamic tests of Serial Peripheral Interface- (SPI) or Integrated Circuit- (IC) communications or any other interface between elements of an ECU.

8.4.2.2.5 This requirement applies to ASIL (A), (B), C, and D, in accordance with 4.3: the effectiveness of the hardware fault detection mechanisms' diagnostic coverage at the hardware-software level, with respect to the fault models, shall be demonstrated using feasible test methods given in Table 8.

NOTE For references to fault models, see ISO 26262-5:2011, Annex D.

Table 8 — Effectiveness of a safety mechanism's diagnostic coverage at the hardware-software level

Methods		ASIL			
		A	B	C	D
1a	Fault injection test ^a	+	+	++	++
1b	Error guessing test ^b	+	+	++	++

^a A fault injection test uses special means to introduce faults into the test object during runtime. This can be done within the software via a special test interface or specially prepared hardware. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.

^b An error guessing test uses expert knowledge and data collected through lessons learned to anticipate errors in the test object. Then a set of tests along with adequate test facilities is designed to check for these errors. Error guessing is an effective method given a tester who has previous experience with similar test objects.

8.4.2.2.6 This requirement applies to ASIL (A), (B), (C), and D, in accordance with 4.3: the level of robustness of the elements at the hardware-software level shall be demonstrated using feasible test methods given in Table 9.

Table 9 — Level of robustness at the hardware-software level

Methods		ASIL			
		A	B	C	D
1a	Resource usage test ^a	+	+	+	++
1b	Stress test ^b	+	+	+	++

^a A resources usage test can be done statically (e.g. by checking for code sizes or analyzing the code regarding interrupt usage, in order to verify that worst-case scenarios do not run out of resources), or dynamically by runtime monitoring.

^b A stress test verifies the test object for correct operation under high operational loads or high demands from the environment. Therefore, tests under high loads on the test object, or with exceptional interface loads, or values (bus loads, electrical shocks, etc.), as well as tests with extreme temperatures, humidity or mechanical shocks, can be applied.

8.4.3 System integration and testing

8.4.3.1 System integration

8.4.3.1.1 The individual elements incorporated in the system shall be integrated in accordance with the system design, tested in accordance with the system integration tests and tested in accordance with the specified system integration tests of ISO 26262-5 and ISO 26262-6.

NOTE The tests are intended to provide evidence that each system element interacts correctly, complies with the technical and functional safety requirements, and gives an adequate level of confidence that unintended behaviours, that could violate a safety goal, are absent.

8.4.3.2 Test goals and test methods during system testing

8.4.3.2.1 To detect systematic faults during system integration, the test goals resulting from the requirements 8.4.3.2.2 to 8.4.3.2.6 shall be addressed by the application of adequate test methods, as given in the corresponding tables.

NOTE Depending on the function implemented, its complexity or the distributed nature of the system, it can be reasonable to move test methods to other integration subphases with adequate rationale.

8.4.3.2.2 The correct implementation of the functional and technical requirements at the system level shall be demonstrated using feasible test methods given in Table 10.

Table 10 — Correct implementation of functional safety and technical safety requirements at the system level

	Methods	ASIL			
		A	B	C	D
1a	Requirement-based test ^a	++	++	++	++
1b	Fault injection test ^b	+	+	++	++
1c	Back-to-back test ^c	o	+	+	++

^a A requirements-based test denotes a test against functional and non-functional requirements.

^b A fault injection test uses special means to introduce faults into the system. This can be done within the system via a special test interface or specially prepared elements or communication devices. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.

^c A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation.

8.4.3.2.3 This requirement applies to ASIL (A), (B), (C), and D, in accordance with 4.3: the correct functional performance, accuracy and timing of the safety mechanisms at the system level shall be demonstrated using feasible test methods given in Table 11.

Table 11 — Correct functional performance, accuracy and timing of safety mechanisms at the system level

	Methods	ASIL			
		A	B	C	D
1a	Back-to-back test ^a	o	+	+	++
1b	Performance test ^b	o	+	+	++

^a A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation.

^b A performance test can verify the performance (e.g. actuator speed or strength, whole system response times) of the safety mechanisms concerning the system.

8.4.3.2.4 The consistent and correct implementation of the external and internal interfaces at the system level shall be demonstrated using feasible test methods given in Table 12.

Table 12 — Consistent and correct implementation of external and internal interfaces at the system level

Methods	ASIL			
	A	B	C	D
1a Test of external interfaces ^a	+	++	++	++
1b Test of internal interfaces ^a	+	++	++	++
1c Interface consistency check ^a	o	+	++	++
1d Test of interaction/communication ^b	++	++	++	++

^a An interface test of the system includes tests of analogue and digital inputs and outputs, boundary tests, and equivalence-class tests, to completely test the specified interfaces, compatibility, timings, and other specified characteristics of the system. Internal interfaces of the system can be tested by static tests (e.g. match of plug connectors) as well as by dynamic tests concerning bus communications or any other interface between system elements.

^b A communication and interaction test includes tests of the communication between the system elements, as well as between the system under test and other vehicle systems during runtime, against the functional and non-functional requirements.

8.4.3.2.5 This requirement applies to ASIL (A), (B), C, and D, in accordance with 4.3: the effectiveness of the safety mechanisms' failure coverage at the system level shall be demonstrated using feasible test methods given in Table 13.

Table 13 — Effectiveness of a safety mechanism's failure coverage at the system level

Methods	ASIL			
	A	B	C	D
1a Fault injection test ^a	+	+	++	++
1b Error guessing test ^b	+	+	++	++
1c Test derived from field experience	o	+	++	++

^a A fault injection test uses special means to introduce faults into the system. This can be done within the system via a special test interface, specially prepared elements, or communication devices. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety measures are not invoked.

^b An error guessing test uses expert knowledge and data collected through lessons learned and field experience to anticipate errors in the system. Then a set of tests along with adequate test facilities is designed to check for these errors. Error guessing is an effective method given a tester who has previous experience with similar systems.

8.4.3.2.6 The level of robustness at the system level shall be demonstrated using feasible test methods given in Table 14.

Table 14 — Level of robustness at the system level

Methods	ASIL			
	A	B	C	D
1a Resource usage test ^a	o	+	++	++
1b Stress test ^b	o	+	++	++
1c Test for interference resistance and robustness under certain environmental conditions ^c	++	++	++	++

^a At the system level resource usage testing is usually performed in dynamic environments (e.g. lab cars or prototypes). Issues to test include power consumption and bus load.

^b A stress test verifies the correct operation of the system under high operational loads or high demands from the environment. Therefore, tests under high loads on the system, or with extreme user inputs or requests from other systems, as well as tests with extreme temperatures, humidity or mechanical shocks, can be applied.

^c A test for interference resistance and robustness, under certain environmental conditions, is a special case of stress testing. This includes EMC and ESD tests (e.g. see [2], [3]).

8.4.4 Vehicle integration and testing

8.4.4.1 Vehicle integration

8.4.4.1.1 The item shall be integrated into the vehicle and the vehicle integration tests shall be completed.

8.4.4.1.2 The verification of the interface specification of the item with the in-vehicle communication network and the in-vehicle power supply network shall be performed.

8.4.4.2 Test goals and test methods during vehicle testing

8.4.4.2.1 To detect systematic faults during vehicle integration, the test goals, resulting from the requirements 8.4.4.2.2 to 8.4.4.2.6, shall be addressed by the application of adequate test methods as given in the corresponding tables.

NOTE Depending on the function implemented, its complexity or the distributed nature of the item, it can be reasonable to move test methods to other integration subphases with adequate rationale.

8.4.4.2.2 The correct implementation of the functional safety requirements at the vehicle level shall be demonstrated using feasible test methods given in Table 15.

Table 15 — Correct implementation of the functional safety requirements at the vehicle level

	Methods	ASIL			
		A	B	C	D
1a	Requirement-based test ^a	++	++	++	++
1b	Fault injection test ^b	++	++	++	++
1c	Long-term test ^c	++	++	++	++
1d	User test under real-life conditions ^c	++	++	++	++

^a A requirements-based test denotes a test against functional and non-functional requirements.

^b A fault injection test uses special means to introduce faults into the item. This can be done within the item via a special test interface or specially prepared elements or communication devices. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.

^c A long-term test and a user test under real-life conditions are similar to tests derived from field experience but use a larger sample size, normal users as testers, and are not bound to prior specified test scenarios, but performed under real-life conditions during everyday life. These tests can have limitations if necessary to ensure the safety of the testers, e.g. with additional safety measures or disabled actuators.

8.4.4.2.3 This requirement applies to ASIL (A), (B), C, and D, in accordance with 4.3: the correct functional performance, accuracy and timing of the safety mechanisms at the vehicle level shall be demonstrated using feasible test methods given in Table 16.

Table 16 — Correct functional performance, accuracy and timing of safety mechanisms at the vehicle level

	Methods	ASIL			
		A	B	C	D
1a	Performance test ^a	+	+	++	++
1b	Long-term test ^b	+	+	++	++
1c	User test under real-life conditions ^b	+	+	++	++

^a A performance test can verify the performance (e.g. fault tolerant time intervals and vehicle controllability in the presence of faults) of the safety mechanisms concerning the item.

^b A long-term test and a user test under real-life conditions are similar to tests derived from field experience but use a larger sample size, normal users as testers, and are not bound to prior specified test scenarios, but performed under real-life conditions during everyday life. These tests can have limitations if necessary to ensure the safety of the testers, e.g. with additional safety measures or disabled actuators.

8.4.4.2.4 This requirement applies to ASIL (A), (B), C, and D, in accordance with 4.3: the consistency and correctness of the implementation of the external interfaces at the vehicle level shall be demonstrated using feasible test methods given in Table 17.

Table 17 — Consistent and correct implementation of internal and external interfaces at the vehicle level

Methods	ASIL			
	A	B	C	D
1a Test of external interfaces ^a	O	+	++	++
1b Test of interaction/communication ^b	O	+	++	++

^a An interface test at the vehicle level tests the interfaces of the vehicle systems for compatibility. This can be done statically by validating value ranges, ratings or geometries as well as dynamically during operation of the whole vehicle.
^b A communication and interaction test includes tests of the communication between the systems of the vehicle during runtime against functional and non-functional requirements.

8.4.4.2.5 This requirement applies to ASIL (A), (B), C, and D, in accordance with 4.3: the effectiveness of the safety mechanisms' failure coverage at the vehicle level shall be demonstrated using feasible test methods given in Table 18.

Table 18 — Effectiveness of a safety mechanism's failure coverage at the vehicle level

Methods	ASIL			
	A	B	C	D
1a Fault injection test ^a	O	+	++	++
1b Error guessing test ^b	O	+	++	++
1c Test derived from field experience ^c	O	+	++	++

^a A fault injection test uses special means to introduce faults into the vehicle. This can be done within the vehicle via a special test interface, specially prepared hardware or communication devices. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety measures are not invoked.
^b An error guessing test uses expert knowledge and data collected through lessons learned to anticipate errors in the vehicle. Then a set of tests along with adequate test facilities is designed to check for these errors. Error guessing is an effective method given a tester who has previous experience with similar vehicle applications.
^c A test derived from field experience uses the experience and data gathered from the field. Erroneous vehicle behaviour or newly discovered operational situations are analysed and a set of tests is designed to check the vehicle with respect to the new findings.

8.4.4.2.6 This requirement applies to ASIL (A), (B), C, and D, in accordance with 4.3: the level of robustness at the vehicle level shall be demonstrated using feasible test methods given in Table 19.

Table 19 — Level of robustness at the vehicle level

Methods	ASIL			
	A	B	C	D
1a Resource usage test ^a	o	+	++	++
1b Stress test ^b	o	+	++	++
1c Test for interference resistance and robustness under certain environmental conditions ^c	o	+	++	++
1d Long-term test ^d	o	+	++	++

^a At the item level, resource usage testing is usually performed in dynamic environments (e.g. lab cars or prototypes). Issues to test include item internal resources, power consumption or limited resources of other vehicle systems.

^b A stress test verifies the correct operation of the vehicle under high operational loads or high demands from the environment. Therefore tests under high loads on the vehicle or with extreme user inputs or requests from other systems as well as tests with extreme temperatures, humidity or mechanical shocks can be applied.

^c A test for interference resistance and robustness, under certain environmental conditions, is a special case of stress testing. This includes EMC and ESD tests (e.g. see [2], [3]).

^d A long-term test and a user test under real-life conditions are similar to tests derived from field experience but use a larger sample size, normal users as testers, and are not bound to prior specified test scenarios, but performed under real-life conditions during everyday life.

8.5 Work products

8.5.1 Item integration and testing plan (refined) resulting from requirement 8.4.1.

8.5.2 Integration testing specification(s) resulting from requirements 8.4.1.

8.5.3 Integration testing report(s) resulting from requirements 8.4.2, 8.4.3 and 8.4.4.

9 Safety validation

9.1 Objectives

The first objective is to provide evidence of compliance with the safety goals and that the functional safety concepts are appropriate for the functional safety of the item.

The second objective is to provide evidence that the safety goals are correct, complete and fully achieved at the vehicle level.

9.2 General

The purpose of the preceding verification activities (e.g. design verification, safety analyses, hardware, software, and item integration and testing) is to provide evidence that the results of each particular activity comply with the specified requirements.

The validation of the integrated item in representative vehicle(s) aims to provide evidence of appropriateness for the intended use and aims to confirm the adequacy of the safety measures for a class or set of vehicles. Safety validation does cover assurance, that the safety goals are sufficient and have been achieved, based on examination and tests.

9.3 Inputs to this clause

9.3.1 Prerequisites

The following information shall be available:

- hazard analysis and risk assessment in accordance with ISO 26262-3:2011, 7.5.1;
- safety goals in accordance with ISO 26262-3:2011, 7.5.2; and
- functional safety concept in accordance with ISO 26262-3:2011, 8.5.1.

9.3.2 Further supporting information

The following information can be considered:

- project plan (refined) (see 5.5.1);
- technical safety concept (see 7.5.1);
- functional concept (from external source); and
- item integration and testing plan (refined) (see 8.5.1); and
- safety analysis reports (see 7.5.6).

9.4 Requirements and recommendation

9.4.1 Validation environment

9.4.1.1 The safety goals shall be validated for the item integrated in a representative vehicle.

NOTE This integrated item includes, where applicable: system; software; hardware; elements of other technologies, external measures.

9.4.2 Planning of validation

9.4.2.1 The validation plan shall be refined, including:

- a) the configuration of the item subjected to validation including its calibration data in accordance with ISO 26262-6:2011, Annex C;
NOTE If a complete validation of each item configuration is not feasible, then a reasonable subset can be selected.
- b) the specification of validation procedures, test cases, driving manoeuvres, and acceptance criteria; and
- c) the equipment and the required environmental conditions.

9.4.3 Execution of validation

9.4.3.1 If testing is used for validation, then the same requirements as provided for verification testing (see ISO 26262-8:2011, 9.4.2 and 9.4.3) may be applied.

9.4.3.2 The safety goals of the item shall be validated at the vehicle level by evaluating the following:

- a) the controllability;

NOTE Controllability can be validated using operating scenarios, including intended use and foreseeable misuse.

- b) the effectiveness of safety measures for controlling random and systematic failures;
- c) the effectiveness of the external measures; and
- d) the effectiveness of the elements of other technologies.

9.4.3.3 This requirement applies to ASILs (B), C, and D of the safety goal: the validation of the metrics for random hardware failures shall be carried out at the item level for:

- a) the evaluation of safety goal violations due to random hardware failures as determined in ISO 26262-5:2011, Clause 9, against the target values as defined by requirement 7.4.4.3; and
- b) the evaluation of the hardware architectural metrics in accordance with the assessment criteria of ISO 26262-5:2011, Clause 8, against the target values as defined by requirement 7.4.4.2.

NOTE Quantitative evaluation for elements of the item is defined in ISO 26262-5:2011, 9.4.2 and 9.4.3. The whole item is evaluated qualitatively in case other technologies are involved in the item.

9.4.3.4 The validation at the vehicle level, based on the safety goals, the functional safety requirements and the intended use, shall be executed as planned using:

- a) the validation procedures and test cases for each safety goal including detailed pass/fail criteria; and
- b) the scope of application. This may include issues such as configuration, environmental conditions, driving situations, operational use cases, etc.

NOTE Operational use cases can be created to help focus the safety validation at the vehicle level.

9.4.3.5 An appropriate set of the following methods shall be applied:

- a) repeatable tests with specified test procedures, test cases, and pass/fail criteria;

EXAMPLE 1 positive tests of functions and safety requirements, black box testing, simulation, tests under boundary conditions, fault injection, durability tests, stress tests, highly accelerated life testing (HALT), simulation of external influences.

- b) analyses;

EXAMPLE 2 FMEA, FTA, ETA, simulation.

- c) long-term tests, such as vehicle driving schedules and captured test fleets;
- d) user tests under real-life conditions, panel or blind tests, expert panels; and
- e) reviews.

9.4.4 Evaluation

9.4.4.1 The results of the validation shall be evaluated.

9.5 Work products

9.5.1 Validation plan (refined) resulting from requirement 9.4.2.

9.5.2 Validation report resulting from requirements 9.4.3 and 9.4.4.

10 Functional safety assessment

10.1 Objectives

The objective of the requirements in this clause is to assess the functional safety that is achieved by the item.

10.2 General

The organizational entity with responsibility for functional safety (e.g. the vehicle manufacturer or the supplier, if the latter is responsible for functional safety) initiates an assessment of functional safety.

10.3 Inputs to this clause

10.3.1 Prerequisites

The following information shall be available:

- safety case in accordance with ISO 26262-2:2011, 6.5.3;
- safety plan (refined) in accordance with 5.5.2, ISO 26262-5:2011, 5.5.2 and ISO 26262-6:2011, 5.5.2;
- confirmation measure reports in accordance with ISO 26262-2:2011, 6.5.5;
- audit report if available in accordance with ISO 26262-2:2011, 6.5.4; and
- functional safety assessment plan (refined) in accordance with 5.5.5.

10.3.2 Further supporting information

None.

10.4 Requirements and recommendation

10.4.1 This requirement applies to ASILs (B), C, and D of the safety goal: for each step of the safety lifecycle in ISO 26262-2:2011, Figure 2, the specific topics to be addressed by the functional safety assessment shall be identified.

10.4.2 This requirement applies to ASILs (B), C, and D of the safety goal: the functional safety assessment shall be conducted in accordance with ISO 26262-2:2011, 6.4.9 (Functional safety assessment).

10.5 Work products

10.5.1 Functional safety assessment report resulting from requirements 10.4.1 and 10.4.2.

11 Release for production

11.1 Objectives

11.1.1 The objective of this clause is to specify the release for production criteria at the completion of the item development. The release for production confirms that the item complies with the requirements for functional safety at the vehicle level.

11.2 General

11.2.1 The release for production confirms that the item is ready for series-production and operation.

11.2.2 The evidence of compliance with the prerequisites for serial production is provided by:

- The completion of the verification and validation during the development at the hardware, software, system, item and vehicle level; and
- the successful overall assessment of functional safety.

11.2.3 This release documentation forms a basis for the production of the components, systems or vehicles, and is signed by the person responsible for the release.

11.3 Inputs to this clause

11.3.1 Prerequisites

The following information shall be available:

- functional safety assessment report in accordance with 10.5.1; and
- safety case in accordance with ISO 26262-2:2011, 6.5.3.

11.3.2 Further supporting information

None.

11.4 Requirements and recommendations

11.4.1 Release of production

11.4.1.1 The release for production of the item shall only be approved if the work products listed under 11.3.1 are available, if applicable (depending on the ASIL), and provide confidence of functional safety.

11.4.2 Documentation for release for production

11.4.2.1 The documentation of functional safety for release for production shall include the following information:

- a) the name and signature of the person responsible for release;
- b) the version(s) of the released item;
- c) the configuration of the released item;
- d) references to associated documents; and
- e) the release date.

NOTE The documentation of functional safety can be part of the release for production documentation of the item, or it can be a separate document.

11.4.2.2 At release for production, a baseline for software and a baseline for hardware shall be available, and that shall be documented in accordance with ISO 26262-8:2011, Clause 10.

11.4.2.3 Identified safety anomalies shall be addressed in accordance with ISO 26262-2:2011, 5.4.2, and ISO 26262-8:2011, Clause 8.

11.5 Work products

11.5.1 Release for production report resulting from requirements 11.4.1 and 11.4.2.

Annex A (informative)

Overview and document flow of product development at the system level

Table A.1 provides an overview of objectives, prerequisites and work products of the particular subphases of product development at the system level.

Table A.1 — Overview of product development at the system level

Clause	Objectives	Prerequisites	Work products
5 Initiation of product development at the system level	<p>The objective of the initiation of the product development at the system level is to determine and plan the functional safety activities during the individual subphases of system development. This also includes the necessary supporting processes described in ISO 26262-8.</p> <p>This planning of system-level safety activities will be included in the safety plan.</p>	<p>Project plan (refined) (see ISO 26262-2:2011, 6.5.2)</p> <p>Safety plan (see ISO 26262-2:2011, 6.5.1)</p> <p>Functional safety concept (see ISO 26262-3:2011, 8.5.1)</p>	<p>5.5.1 Project plan (refined)</p> <p>5.5.2 Safety plan (refined)</p> <p>5.5.3 Item integration and testing plan(s)</p> <p>5.5.4 Validation plan</p> <p>5.5.5 Functional safety assessment plan (refined)</p>
6 Specification of the technical safety requirements	<p>The first objective of this subphase is to specify the technical safety requirements. The technical safety requirements specification refines the functional safety concept, considering both the functional concept and the preliminary architectural assumptions (see ISO 26262-3).</p> <p>The second objective is to verify through analysis that the technical safety requirements comply with the functional safety requirements.</p>	<p>Functional safety concept (see ISO 26262-3:2011, 8.5.1)</p> <p>Validation plan (see 5.5.4)</p>	<p>6.5.1 Technical safety requirements specification</p> <p>6.5.2 System verification report</p> <p>6.5.3 Validation plan (refined)</p>
7 System design	<p>The first objective of this subphase is to develop the system design and the technical safety concept that comply with the functional requirements and the technical safety requirements specification of the item.</p> <p>The second objective of this subphase is to verify that the system design and the technical safety concept comply with the technical safety requirements specification.</p>	<p>Item integration and testing plan (see 5.5.3)</p> <p>Technical safety requirements specification (see 6.5.1)</p>	<p>7.5.1 Technical safety concept</p> <p>7.5.2 System design specification</p> <p>7.5.3 Hardware-software interface specification (HSI)</p> <p>7.5.4 Specification of requirements for production, operation, service and decommissioning</p> <p>7.5.5 System verification report (refined)</p> <p>7.5.6 Safety analysis reports resulting from requirement 7.4.3</p>

Table A.1 (continued)

Clause	Objectives	Prerequisites	Work products
8 Item integration and testing	<p>The integration and testing phase comprises three phases and two primary goals as described below: the first phase is the integration of the hardware and software of each element that the item comprises. The second phase is the integration of the elements that comprise an item to form a complete system. The third phase is the integration of the item with other systems within a vehicle and with the vehicle itself.</p> <p>The first objective of the integration process is to test compliance with each safety requirement in accordance with its specification and ASIL classification.</p> <p>The second objective is to verify that the "System design" covering the safety requirements [see Clause 7 (System design)] are correctly implemented by the entire item.</p>	<p>Safety goals (see ISO 26262-3:2011, 7.5.2)</p> <p>Functional safety concept (see ISO 26262-3:2011, 8.5.1)</p> <p>Item integration and testing plan (see 5.5.3)</p> <p>Technical safety concept (see 7.5.1)</p> <p>System design specification (see 7.5.2)</p> <p>Hardware-software interface specification (HSI) (see 7.5.3)</p>	<p>8.5.1 Item integration and testing plan (refined)</p> <p>8.5.2 Integration testing specification(s)</p> <p>8.5.3 Integration testing report(s)</p>
9 Safety validation	<p>The first objective is to provide evidence of compliance with the safety goals and that the functional safety concepts are appropriate for the functional safety of the item.</p> <p>The second objective is to provide evidence that the safety goals are correct, complete and fully achieved at the vehicle level.</p>	<p>Hazard analysis and risk assessment (see ISO 26262-3:2011, 7.5.1)</p> <p>Safety goals (see ISO 26262-3:2011, 7.5.2)</p> <p>Functional safety concept (see ISO 26262-3:2011, 8.5.1)</p> <p>Validation plan (refined) (see 6.5.3)</p>	<p>9.5.1 Validation plan (refined) resulting from requirement 9.4.2</p> <p>9.5.2 Validation report resulting from requirements 9.4.3 to 9.4.4</p>
10 Functional safety assessment	The objective of the requirements in this clause is to assess the functional safety that is achieved by the item.	<p>Safety case (see ISO 26262-2:2011, 6.5.3)</p> <p>Safety plan (refined) (see 5.5.2, ISO 26262-5:2011, 5.5.2 and ISO 26262-6:2011, 5.5.2)</p> <p>Confirmation review reports (see ISO 26262-2:2011, 6.5.5)</p> <p>Audit report if available (see ISO 26262-2:2011, 6.5.5)</p> <p>Functional safety assessment plan (refined) (see 5.5.5)</p>	<p>10.5.1 Functional safety assessment report resulting from requirements 10.4.1 and 10.4.2</p>
11 Product release	The objective of this clause is to specify the release for production criteria at the completion of the item development. The release for production confirms that the item complies with the requirements for functional safety at the vehicle level.	<p>Functional safety assessment report (see 10.5.1)</p> <p>Safety case (see ISO 26262-2:2011, 6.5.3)</p>	<p>11.5.1 Release for production report resulting from requirements 11.4.1 and 11.4.2</p>

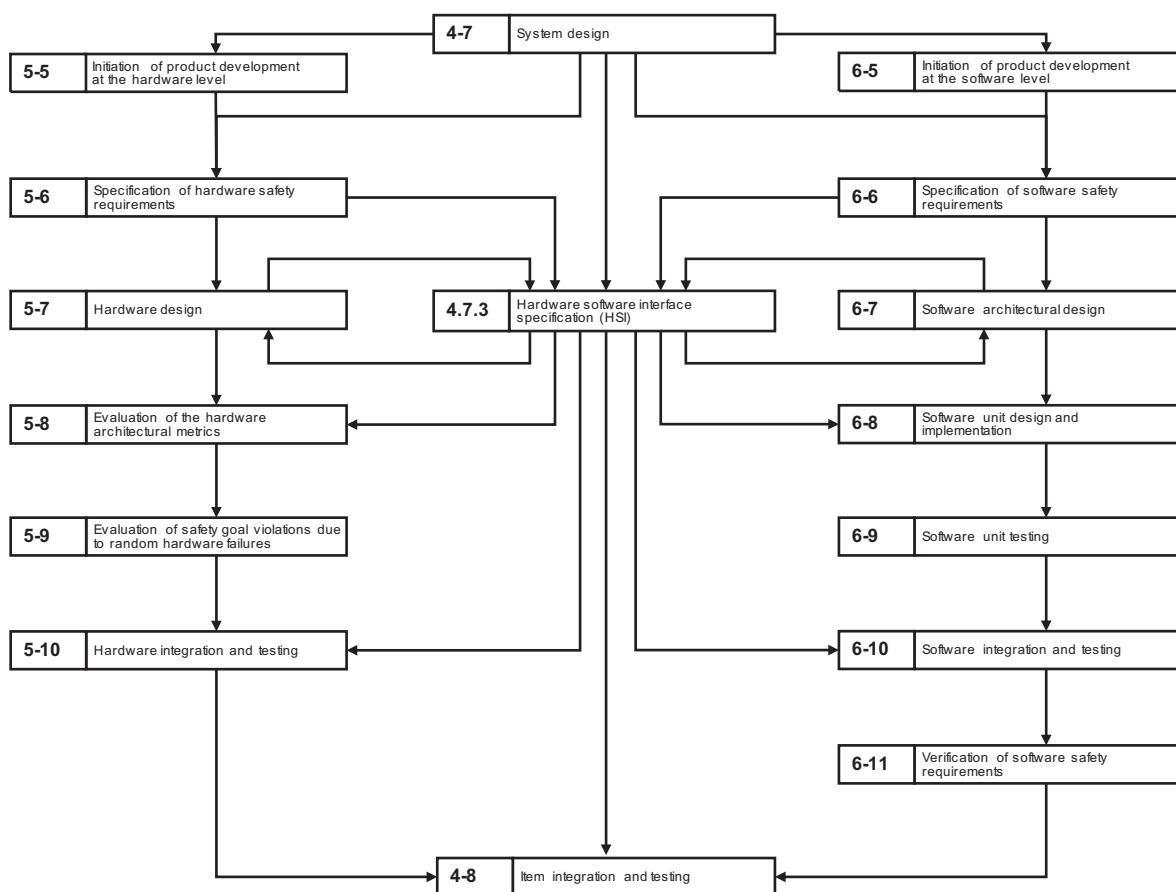
Annex B (informative)

Example contents of hardware-software interface

B.1 This annex provides further explanation on the hardware-software interface.

The hardware-software interface is specified in this part of ISO 26262 during the subphase “System design”. As development continues during the hardware development (ISO 26262-5) and software development (ISO 26262-6) subphases, this specification is refined.

First, an overview is given in Figure B.1 which provides the relationship between product development at the system, hardware and software level, and the role of the hardware-software interface. The hardware-software interface acts as the linkage between the different levels of development. The hardware-software interface is used to agree topics relevant to both hardware and software development.



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: “m-n”, where “m” represents the number of the part and “n” indicates the number of the clause, e.g. “3-6” represents Clause 6 of ISO 26262-3.

Figure B.1 — Overview on interaction with the hardware-software interface

Second, to make specifying the hardware-software interface easier, a list of typical hardware-software interface elements is given as well as a non-exhaustive list of implementation characteristics implemented by the elements of the hardware-software interface.

B.2 The following hardware-software interface elements can be considered when specifying the hardware-software interface:

- a) memory:
 - 1) volatile memory (e.g. RAM);
 - 2) non-volatile memory (e.g. NvRAM);
- b) bus interfaces [e.g. controller area network (CAN), local interconnect network (LIN), internal high-speed serial link (HSSL)];
- c) converter:
 - 1) A/D converter;
 - 2) D/A converter;
 - 3) pulse-width modulation (PWM);
- d) multiplexer;
- e) electrical I/O;
- f) watchdog:
 - 1) internal;
 - 2) external.

B.3 The following characteristics of the hardware-software interface can be considered when specifying the hardware-software interface:

- a) interrupts;
- b) timing consistency;
- c) data integrity;
- d) initialization:
 - 1) memory and registers;
 - 2) boot management;
- e) message transfer:
 - 1) send message;
 - 2) receive message;
- f) network modes:
 - 1) sleeping;

- 2) awakening;
- g) memory management:
 - 1) reading;
 - 2) writing;
 - 3) diagnostic;
 - 4) address space;
 - 5) data types;
- h) real-time counter:
 - 1) start counter;
 - 2) stop counter;
 - 3) freeze counter;
 - 4) load counter.

Table B.1 provides an example to help with the allocation of hardware-software interface characteristics to hardware-software interface elements.

Table B.1 — Example for inputs of internal signals

Description	HW-Identifier	SW-Identifier	Channel 1	Channel 2	MUX No. - Channel 1	MUX No. - Channel 2	Data type HW Interface	Address Channel 1	Address Channel 2	Unit	Interface Type	Comments	Range of values	Accuracy (% of range of values)
Inputs														
Input 1	IN_1	IN_1	x		4		U16	0x8000		V	Analogue - Internal	Analogue Input 1	0 to 5	0,50 %

Bibliography

- [1] ISO 11451 (all parts), *Road vehicles — Vehicle test methods for electrical disturbances from narrowband radiated electromagnetic energy*
- [2] IEC 61000-6-1, *Electromagnetic compatibility (EMC) — Part 6-1: Generic standards — Immunity for residential, commercial and light-industrial environments*
- [3] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*

This page is intentionally blank.

This page is intentionally blank.

This page is intentionally blank.

ICS 43.040.10

Price based on 36 pages

INTERNATIONAL
STANDARD

ISO
26262-5

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 5:
Product development at the hardware
level**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 5: Développement du produit au niveau du matériel*



Reference number
ISO 26262-5:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	v
Introduction.....	vi
1 Scope.....	1
2 Normative references.....	2
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	3
4.3 ASIL-dependent requirements and recommendations	3
5 Initiation of product development at the hardware level.....	3
5.1 Objectives	3
5.2 General	4
5.3 Inputs to this clause.....	5
5.4 Requirements and recommendations	5
5.5 Work products	5
6 Specification of hardware safety requirements	5
6.1 Objectives	5
6.2 General	6
6.3 Inputs to this clause.....	6
6.4 Requirements and recommendations	6
6.5 Work products	8
7 Hardware design.....	8
7.1 Objectives	8
7.2 General	8
7.3 Inputs to this clause.....	9
7.4 Requirements and recommendations	9
7.5 Work products	13
8 Evaluation of the hardware architectural metrics.....	13
8.1 Objectives	13
8.2 General	13
8.3 Inputs of this clause.....	14
8.4 Requirements and recommendations	15
8.5 Work products	17
9 Evaluation of safety goal violations due to random hardware failures	18
9.1 Objectives	18
9.2 General	18
9.3 Inputs to this clause.....	18
9.4 Requirements and recommendations	19
9.5 Work products	26
10 Hardware integration and testing	26
10.1 Objectives	26
10.2 General	26
10.3 Inputs of this clause.....	26
10.4 Requirements and recommendations	27
10.5 Work products	29
Annex A (informative) Overview of and workflow of product development at the hardware level	30

Annex B (informative) Failure mode classification of a hardware element.....	32
Annex C (normative) Hardware architectural metrics	34
Annex D (informative) Evaluation of the diagnostic coverage	39
Annex E (informative) Example calculation of hardware architectural metrics: “single-point fault metric” and “latent-fault metric”	66
Annex F (informative) Application of scaling factors	72
Bibliography	75

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-5 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

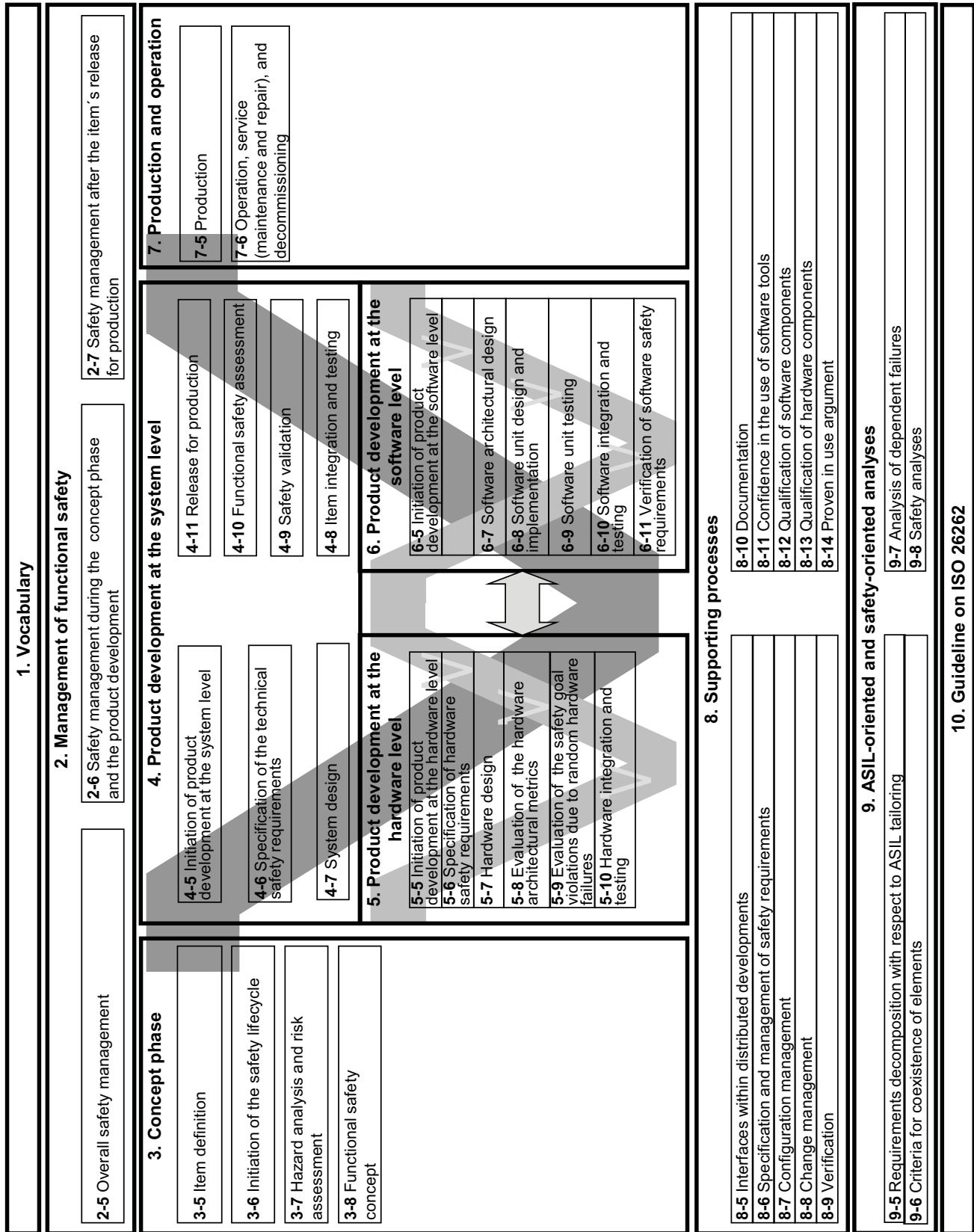


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 5: Product development at the hardware level

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for product development at the hardware level for automotive applications, including the following:

- requirements for the initiation of product development at the hardware level,
- specification of the hardware safety requirements,
- hardware design,
- hardware architectural metrics, and
- evaluation of violation of the safety goal due to random hardware failures and hardware integration and testing.

The requirements of this part of ISO 26262 for hardware elements are applicable both to non-programmable and programmable elements, such as ASIC, FPGA and PLD. Furthermore, for programmable electronic elements, requirements in ISO 26262-6, ISO 26262-8:2011, Clause 11, and ISO 26262-8:2011, Clause 12, are applicable.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-7:2011, *Road vehicles — Functional safety — Part 7: Production and operation*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- a) tailoring of the safety activities in accordance with ISO 26262 has been planned and shows that the requirement does not apply, or
- b) a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Initiation of product development at the hardware level

5.1 Objectives

The objective of the initiation of the product development for the hardware is to determine and plan the functional safety activities during the individual subphases of hardware development. This also includes the necessary supporting processes described in ISO 26262-8.

This planning of hardware-specific safety activities is included in the safety plan (see ISO 26262-2:2011, 6.4.3, and ISO 26262-4:2011, 5.4).

5.2 General

The necessary activities and processes needed to develop hardware that meets the safety requirements are planned. Figure 2 illustrates the hardware level product development process steps in order to comply with the requirements of this part of ISO 26262, and the integration of these steps within the ISO 26262 framework.

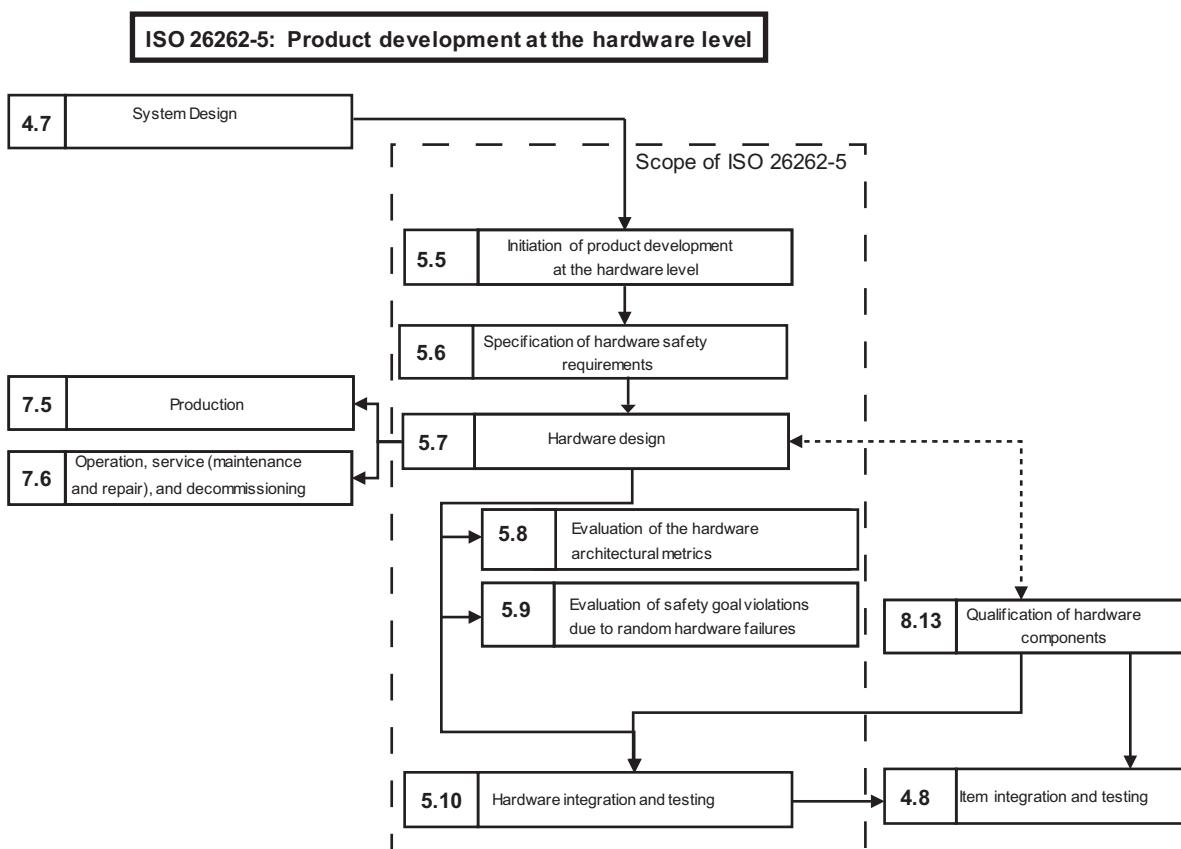
The necessary activities and processes for the product development at the hardware level include:

- the hardware implementation of the technical safety concept;
- the analysis of potential hardware faults and their effects; and
- the coordination with software development.

By contrast to the software development subphases, this part of ISO 26262 contains two clauses describing quantitative evaluations of the overall hardware architecture of the item.

Clause 8 describes two metrics to evaluate the effectiveness of the hardware architecture of the item and the implemented safety mechanisms to cope with random hardware failures.

As a complement to Clause 8, Clause 9 describes two alternatives to evaluate whether the residual risk of safety goal violations is sufficiently low, either by using a global probabilistic approach or by using a cut-set analysis to study the impact of each identified fault of a hardware element upon the violation of the safety goals.



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: "m-n", where "m" represents the number of the part and "n" indicates the number of the clause, e.g. "4.7" represents Clause 7 of ISO 26262-4.

Figure 2 — Reference phase model for the product development at the hardware level

5.3 Inputs to this clause

5.3.1 Prerequisites

The following information shall be available:

- project plan (refined) in accordance with ISO 26262-4:2011, 5.5.1;
- safety plan (refined) in accordance with ISO 26262-4:2011, 5.5.2; and
- item integration and testing plan (refined) in accordance with ISO 26262-4:2011, 5.5.3.

5.3.2 Further supporting information

The following information can be considered:

- qualification report (of hardware components or parts), if applicable (see ISO 26262-8:2011, 13.5.3).

5.4 Requirements and recommendations

5.4.1 The safety plan in accordance with ISO 26262-2 shall be detailed, including determination of appropriate methods and measures, with respect to the activities for the product development at the hardware level, consistent with the planning of activities in ISO 26262-6.

5.4.2 The hardware development process for the hardware of the item, including methods and tools, shall be consistent across all subphases of the hardware development, and consistent with system and software subphases, so that the requirement flow retains its accuracy and consistency during the hardware development.

5.4.3 The tailoring of the safety lifecycle activities for product development at the hardware level shall be performed in accordance with ISO 26262-2:2011, 6.4.5, and based on the reference phase model given in Figure 2.

5.4.4 The reuse of hardware components, or the use of qualified hardware components or parts, shall be identified and the resulting tailoring of the safety activities shall be described.

5.5 Work products

5.5.1 Safety plan (refined) resulting from requirements 5.4.1 to 5.4.4.

6 Specification of hardware safety requirements

6.1 Objectives

The first objective of this clause is to specify the hardware safety requirements. They are derived from the technical safety concept and system design specification.

The second objective is to verify that the hardware safety requirements are consistent with the technical safety concept and the system design specification.

A further objective of this phase is to detail the hardware-software interface (HSI) specification initiated in ISO 26262-4:2011, Clause 7.

6.2 General

The technical safety requirements are allocated to hardware and software. The requirements that are allocated to both are further partitioned to yield hardware only safety requirements. The hardware safety requirements are further detailed, considering design constraints and the impact of these design constraints on the hardware.

6.3 Inputs to this clause

6.3.1 Prerequisites

The following information shall be available:

- safety plan (refined) in accordance with 5.5;
- technical safety concept in accordance with ISO 26262-4:2011, 7.5.1;
- system design specification in accordance with ISO 26262-4:2011, 7.5.2; and
- hardware-software interface specification in accordance with ISO 26262-4:2011, 7.5.3.

6.3.2 Further supporting information

The following information can be considered:

- software safety requirements specification (see ISO 26262-6:2011, 6.5.1).

6.4 Requirements and recommendations

6.4.1 A hardware safety requirements specification for the hardware elements of the item shall be derived from the technical safety requirements allocated to hardware.

6.4.2 The hardware safety requirements specification shall include each hardware requirement that relates to safety, including the following:

NOTE 1 The hardware safety requirements described in bullets a), b), c), or d) include the attributes needed to ensure the effectiveness of the above safety mechanisms.

- a) the hardware safety requirements and relevant attributes of safety mechanisms to control internal failures of the hardware of the element, this includes internal safety mechanisms to cover transient faults when shown to be relevant due, for instance, to the technology used;

EXAMPLE 1 Attributes can include the timing and detection abilities of a watchdog.

- b) the hardware safety requirements and relevant attributes of safety mechanisms to ensure the element is tolerant to failures external to the element;

EXAMPLE 2 The functional behaviour required for an ECU in the event of an external failure, such as an open-circuit on an input of the ECU.

- c) the hardware safety requirements and relevant attributes of safety mechanisms to comply with the safety requirements of other elements;

EXAMPLE 3 Diagnosis of sensors or actuators.

- d) the hardware safety requirements and relevant attributes of safety mechanisms to detect and signal internal or external failures; and

NOTE 2 The hardware safety requirements described in bullet d) include safety mechanisms to prevent faults from being latent.

EXAMPLE 4 The specified fault reaction time for the hardware part of a safety mechanism, so as to be consistent with the fault tolerant time interval.

- e) the hardware safety requirements not specifying safety mechanisms.

EXAMPLE 5 Examples are:

- requirements on the hardware elements to meet the target values for random hardware failures as described in 6.4.3 and 6.4.4;
- requirements for the avoidance of a specific behaviour (for instance, “a particular sensor shall not produce an unstable output”);
- requirements allocated to hardware elements implementing the intended functionality; and
- requirements specifying design measures on harnesses or connectors.

6.4.3 This requirement applies to ASIL (B), C, and D of the safety goal. The target values specified to comply with ISO 26262-4:2011, Clause 7, for the metrics of Clause 8 of this part of ISO 26262 shall be considered when deriving values for the hardware elements of the item.

NOTE This activity can include a split of target values in the case of a distributed development as given in ISO 26262-8:2011, Clause 5.

6.4.4 This requirement applies to ASIL (B), C, and D of the safety goal. The target values specified to comply with ISO 26262-4:2011, Clause 7, for the procedures of Clause 9 of this part of ISO 26262 shall be considered when deriving values for the hardware elements of the item.

NOTE This activity can include a split of target values in the case of a distributed development as given in ISO 26262-8:2011, Clause 5.

6.4.5 The hardware safety requirements shall be specified in accordance with ISO 26262-8:2011, Clause 6.

6.4.6 The criteria for design verification of the hardware of the item or element shall be specified, including environmental conditions (temperature, vibration, EMI, etc.), specific operational environment (supply voltage, mission profile, etc.) and component specific requirements:

- a) for verification by qualification for hardware components or part of intermediate complexity, the criteria shall meet the needs of ISO 26262-8:2011, Clause 13, and
- b) for verification by testing, the criteria shall meet the needs of Clause 10.

6.4.7 The hardware safety requirements shall comply with the fault tolerant time interval for safety mechanisms as specified in ISO 26262-4:2011, 6.4.2.3.

6.4.8 The hardware safety requirements shall comply with the multiple-point fault detection interval as specified in ISO 26262-4:2011, 6.4.4.2.

NOTE 1 In the case of ASIL C and D safety goals, and if the corresponding safety concept does not prescribe specific values, the multiple-point fault detection intervals can be specified to be equal or lower than the item's “power-up to power-down” cycle.

NOTE 2 Appropriate multiple-point fault detection intervals can also be justified by the quantitative analysis of the occurrence of random hardware failures (see Clause 9).

6.4.9 The hardware safety requirements shall be verified in accordance with ISO 26262-8:2011, Clauses 6 and 9, in order to provide evidence of their:

- a) consistency with the technical safety concept, the system design specification and the hardware specifications;
- b) completeness with respect to the technical safety requirements allocated to the hardware element;
- c) consistency with the relevant software safety requirements; and
- d) correctness and accuracy.

6.4.10 The HSI specification initiated in ISO 26262-4:2011, Clause 7, shall be refined sufficiently to allow for the correct control and usage of the hardware by the software, and shall describe each safety-related dependency between hardware and software.

6.4.11 The persons responsible for hardware and software development shall be jointly responsible for the verification of the adequacy of the refined HSI specification.

6.5 Work products

6.5.1 **Hardware safety requirements specification (including test and qualification criteria)** resulting from requirements 6.4.1 to 6.4.8.

6.5.2 **Hardware-software interface specification (refined)** resulting from requirements 6.4.10 and 6.4.11.

NOTE This work product refers to the same work product as given in ISO 26262-6:2011 6.5.2.

6.5.3 **Hardware safety requirements verification report** resulting from requirement 6.4.9.

7 Hardware design

7.1 Objectives

The first objective of this clause is to design the hardware in accordance with the system design specification and the hardware safety requirements.

The second objective of this clause is to verify the hardware design against the system design specification and the hardware safety requirements.

7.2 General

Hardware design includes hardware architectural design and hardware detailed design. Hardware architectural design represents all hardware components and their interactions with one another. Hardware detailed design is at the level of electrical schematics representing the interconnections between hardware parts composing the hardware components.

In order to develop a single hardware design both hardware safety requirements as well as all non-safety requirements have to be complied with. Hence, in this subphase, safety and non-safety requirements are handled within one development process.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- hardware safety requirements specification in accordance with 6.5.1;
- hardware-software interface specification (refined) in accordance with 6.5.2;
- system design specification in accordance with ISO 26262-4:2011, 7.5.2; and
- safety plan (refined) in accordance with 5.5.

7.3.2 Further supporting information

The following information can be considered:

- software safety requirements specification (see ISO 26262-6:2011, 6.5.1).

7.4 Requirements and recommendations

7.4.1 Hardware architectural design

7.4.1.1 The hardware architecture shall implement the hardware safety requirements defined in Clause 6.

7.4.1.2 Each hardware component shall inherit the highest ASIL from the hardware safety requirements it implements.

NOTE Each characteristic of the hardware component will inherit the highest ASIL from the hardware safety requirements that it implements.

7.4.1.3 If ASIL decomposition is applied to the hardware safety requirements during hardware architectural design, it shall be applied in accordance with ISO 26262-9:2011, Clause 5.

7.4.1.4 If a hardware element is made of sub-elements that have different ASILs assigned, or sub-elements that have no ASIL assigned and safety-related sub-elements, then each of these shall be treated in accordance with the highest ASIL, unless the criteria for coexistence in accordance with ISO 26262-9 are met.

7.4.1.5 The traceability between the hardware safety requirements and their implementation shall be maintained down to the lowest level of hardware components.

NOTE The traceability is not required down to hardware detailed design and no ASILs are assigned to hardware parts.

7.4.1.6 In order to avoid failures resulting from high complexity the hardware architectural design shall exhibit the following properties by use of the principles listed in Table 1:

- a) modularity;
- b) adequate level of granularity; and
- c) simplicity.

Table 1 — Properties of modular hardware design

	Properties	ASIL			
		A	B	C	D
1	Hierarchical design	+	+	+	+
2	Precisely defined interfaces of safety-related hardware components	++	++	++	++
3	Avoidance of unnecessary complexity of interfaces	+	+	+	+
4	Avoidance of unnecessary complexity of hardware components	+	+	+	+
5	Maintainability (service)	+	+	++	++
6	Testability ^a	+	+	++	++

^a Testability includes testability during development and operation.

7.4.1.7 Non-functional causes for failure of a safety-related hardware component shall be considered during hardware architectural design, including the following influences, if applicable: temperature, vibrations, water, dust, EMI, cross-talk originating either from other hardware components of the hardware architecture or from its environment.

7.4.2 Hardware detailed design

7.4.2.1 In order to avoid common design faults, relevant lessons learned shall be applied in accordance with ISO 26262-2:2011, 5.4.2.7.

7.4.2.2 Non-functional causes for failure of a safety-related hardware part shall be considered during hardware detailed design, including the following influences, if applicable: temperature, vibrations, water, dust, EMI, noise factor, cross-talk originating either from other hardware parts of the hardware component or from its environment.

7.4.2.3 The operating conditions of the hardware parts used in the hardware detailed design shall comply with the specification of their environmental and operational limits.

7.4.2.4 Robust design principles should be considered.

NOTE Robust design principles can be shown by use of checklists based on QM methods.

EXAMPLE Conservative specification of components.

7.4.3 Safety analyses

7.4.3.1 Safety analyses on hardware design to identify the causes of failures and the effects of faults shall be applied in accordance with Table 2 and ISO 26262-9:2011, Clause 8.

NOTE 1 The initial purpose of the safety analyses is to support the specification of the hardware design. Subsequently, the safety analyses can be used for verification of the hardware design (see 7.4.4).

NOTE 2 In its aims of supporting the specification of the hardware design, qualitative analysis can be appropriate and sufficient.

Table 2 — Hardware design safety analysis

	Methods	ASIL			
		A	B	C	D
1	Deductive analysis ^a	o	+	++	++
2	Inductive analysis ^b	++	++	++	++
NOTE The level of detail of the analysis is commensurate with the level of detail of the design. Both methods can, in certain cases, be carried out at different levels of detail.					
^a A typical deductive analysis method is FTA.					
^b A typical inductive analysis method is FMEA.					

7.4.3.2 This requirement applies to ASIL (B), C, and D of the safety goal. For each safety-related hardware component or part, the safety analyses shall identify the following for the safety goal under consideration:

- a) safe faults;
- b) single-point faults or residual faults; and
- c) multiple-point faults (either perceived, detected or latent).

NOTE 1 In most of the cases, the analysis can be limited to dual-point faults. But sometimes multiple-point faults of a higher order than two can be shown relevant in the technical safety concept (e.g. when implementing redundant safety mechanisms).

NOTE 2 The intention of the identification of dual-point faults is not to require a systematic analysis of every possible combination of two hardware faults but, at a minimum, to consider combinations that derive from the technical safety concept (for instance the combination of two faults where one fault affects a safety-related element and another fault affects the corresponding safety mechanism intended to achieve or maintain a safe state).

7.4.3.3 This requirement applies to ASIL (B), C, and D of the safety goal. Evidence of the effectiveness of safety mechanisms to avoid single-point faults shall be made available.

For that purpose:

- a) evidence of the ability of the safety mechanisms to maintain a safe state, or to switch safely into a safe state, shall be made available (in particular, appropriate failure mitigation ability within the fault tolerant time interval); and
- b) diagnostic coverage with respect to residual faults shall be evaluated.

NOTE 1 A fault that can occur at anytime (e.g. not only at power-up) cannot be considered as being effectively covered if its diagnostic test interval, plus the fault reaction time of the associated safety mechanism, is longer than the relevant fault tolerant time interval.

NOTE 2 If a fault is such that it is possible to demonstrate that it occurs at power-up only and its probability of occurrence is negligible during the duration of the vehicle trip, then for those faults to execute a test at start-up after power-on is acceptable.

NOTE 3 An analysis such as FMEA or FTA can be used to structure the rationale.

NOTE 4 Depending on the knowledge of the failure modes of the hardware elements and their consequences at higher levels, the evaluation can be either a global diagnostic coverage of the hardware element, or a more detailed failure mode coverage evaluation.

NOTE 5 Annex D can be used as a starting point for diagnostic coverage (DC) with the claimed DC supported by a proper rationale.

7.4.3.4 This requirement applies to ASIL (B), C, and D of the safety goal. Evidence of the effectiveness of safety mechanisms to avoid latent faults shall be made available.

For that purpose:

- evidence of the failure detection, and the ability to notify to the driver, within the acceptable multiple-point fault detection interval for latent faults, shall be made available in order to determine which faults remain latent and which faults are not latent; and
- diagnostic coverage with respect to latent faults shall be evaluated.

NOTE 1 A fault cannot be considered covered if its diagnostic test interval, plus the fault reaction time of the associated safety mechanism, is longer than the relevant multiple-point fault detection interval for latent faults.

NOTE 2 An analysis such as FMEA or FTA can be used to structure the rationale.

NOTE 3 Annex D can be used as a starting point for DC with the claimed DC supported by a proper rationale.

NOTE 4 Depending on the knowledge of the failure modes of the hardware elements and their consequences at higher levels, the evaluation can be either a global diagnostic coverage of the hardware element, or a more detailed failure mode coverage evaluation.

7.4.3.5 If applicable, evidence that the hardware design is compliant with its requirements on independence shall be provided based on an analysis of dependent failures in accordance with ISO 26262-9:2011, Clause 7.

7.4.3.6 If new hazards introduced by the hardware design are not already covered by an existing safety goal, they shall be introduced and evaluated in the hazard analysis and risk assessment in accordance with the change management process in ISO 26262-8.

NOTE Newly identified hazards, not already covered by an existing safety goal, are usually non-functional hazards. Non-functional hazards are outside the scope of ISO 26262, but they can be annotated in the hazard analysis and risk assessment with the following statement “No ASIL is assigned to this hazard as it is not within the scope of ISO 26262”. However, an ASIL can be assigned for reference purposes.

7.4.4 Verification of hardware design

7.4.4.1 The hardware design shall be verified in accordance with ISO 26262-8, Clause 9, for compliance and completeness with respect to the hardware safety requirements. To achieve this, the methods listed in Table 3 shall be considered.

Table 3 — Hardware design verification

Methods	ASIL			
	A	B	C	D
1a Hardware design walk-through ^a	++	++	o	o
1b Hardware design inspection ^a	+	+	++	++
2 Safety analyses	In accordance with 7.4.3			
3a Simulation ^b	o	+	+	+
3b Development by hardware prototyping ^b	o	+	+	+
NOTE The scope of this verification review is technical correctness of the hardware design.				
^a Methods 1a and 1b serve as a check of the complete and correct implementation of the hardware safety requirements in the hardware design.				
^b Methods 3a and 3b serve as a check of particular points of the hardware design (e.g. as a fault injection technique) for which analytical methods 1 and 2 are not considered to be sufficient.				

7.4.4.2 If it is discovered, during hardware design, that the implementation of any hardware safety requirement is not feasible, a request for change shall be issued in accordance with the change management process in ISO 26262-8.

7.4.5 Production, operation, service and decommissioning

7.4.5.1 Safety-related special characteristics shall be specified if safety analyses have shown them to be relevant. Attributes of safety-related special characteristics shall include:

- a) the verification measures for production and operation; and
- b) the acceptance criteria for these measures.

EXAMPLE A safety analysis of hardware design that relies on new sensor technologies (e.g., camera or radar sensors) can reveal the relevance of special installation procedures for these sensors. In such a case, additional verification measures for these components can be necessary during the production phase.

7.4.5.2 Instructions for assembly, disassembly and decommissioning of safety-related hardware elements shall be specified, if these operations can impact the technical safety concept.

7.4.5.3 The traceability of safety-related hardware elements shall be ensured, in accordance with ISO 26262-7:2011, 5.4.1.2.

NOTE This can include adequate labelling or other identification of hardware elements to indicate that they are safety-related.

7.4.5.4 Instructions for the maintenance of safety-related hardware elements shall be specified, if the maintenance can impact the technical safety concept.

7.5 Work products

7.5.1 **Hardware design specification** resulting from requirements in 7.4.1 and 7.4.2.

7.5.2 **Hardware safety analysis report** resulting from requirements in 7.4.3.

7.5.3 **Hardware design verification report** resulting from requirements in 7.4.4.

7.5.4 **Specification of requirements related to production, operation, service and decommissioning** resulting from requirements in 7.4.5.

8 Evaluation of the hardware architectural metrics

8.1 Objectives

The objective of this clause is to evaluate the hardware architecture of the item against the requirements for fault handling as represented by the hardware architectural metrics.

8.2 General

This clause describes two hardware architectural metrics for the evaluation of the effectiveness of the architecture of the item to cope with random hardware failures.

These metrics and associated target values apply to the whole hardware of the item and are complementary to the evaluation of safety goal violations due to random hardware failures described in Clause 9.

The random hardware failures addressed by these metrics are limited to some of the item's safety-related electrical and electronic hardware parts, namely those that can significantly contribute to the violation or the achievement of the safety goal, and to the single-point, residual and latent faults of those parts. For electromechanical hardware parts, only the electrical failure modes and failure rates are considered.

NOTE Hardware elements whose faults are multiple-point faults with a higher order than two can be omitted from the calculations unless they can be shown to be relevant in the technical safety concept.

The hardware architectural metrics can be applied iteratively during the hardware architectural design and the hardware detailed design.

The hardware architectural metrics are dependent upon the whole hardware of the item. Compliance with the target figures prescribed for the hardware architectural metrics is achieved for each safety goal in which the item is involved.

These hardware architectural metrics are defined to achieve the following objectives:

- be objectively assessable: metrics are verifiable and precise enough to differentiate between different architectures;
- support evaluation of the final design (the precise calculations are done with the detailed hardware design);
- make available ASIL dependent pass/fail criteria for the hardware architecture;
- reveal whether or not the coverage by the safety mechanisms, to prevent risk from single-point or residual faults in the hardware architecture, is sufficient (single-point fault metric);
- reveal whether or not the coverage by the safety mechanisms, to prevent risk from latent faults in the hardware architecture, is sufficient (latent-fault metric);
- address single-point faults, residual faults and latent faults;
- ensure robustness concerning uncertainty of hardware failure rates;
- be limited to safety-related elements; and
- support usage on different element levels, e.g. target values can be assigned to suppliers' hardware elements.

EXAMPLE To facilitate distributed developments, target values can be assigned to microcontrollers or ECUs.

8.3 Inputs of this clause

8.3.1 Prerequisites

The following information shall be available:

- hardware safety requirements specification in accordance with 6.5.1;
- hardware design specification in accordance with 7.5.1; and
- hardware safety analysis report in accordance with 7.5.2.

8.3.2 Further supporting information

The following information can be considered:

- technical safety concept (see ISO 26262-4:2011, 7.5.1); and
- system design specification (see ISO 26262-4:2011, 7.5.2).

8.4 Requirements and recommendations

8.4.1 This requirement applies to ASIL (B), C, and D of the safety goal. The concepts of diagnostic coverage, single-point fault metric and latent-fault metric, in accordance with Annex C, shall apply to requirements 8.4.2 to 8.4.9.

8.4.2 This requirement applies to ASIL (B), C, and D of the safety goal. The diagnostic coverage of safety-related hardware elements by safety mechanisms shall be estimated with respect to residual faults and with respect to relevant latent faults.

NOTE 1 For this purpose, Tables D.1 to D.14 can be used as a starting point with the claimed DC supported by a proper rationale.

NOTE 2 Depending on the knowledge of the failure modes of the hardware elements and their consequences at a higher level, the evaluation can be either a global diagnostic coverage of the hardware element, or a more detailed failure mode coverage evaluation.

8.4.3 This requirement applies to ASIL (B), C, and D of the safety goal. The estimated failure rates for hardware parts used in the analyses shall be determined:

- a) using hardware part failure rates data from a recognised industry source, or

EXAMPLE Commonly recognised industry sources to determine the hardware part failure rates and the failure mode distributions include IEC/TR 62380, IEC 61709, MIL HDBK 217 F notice 2, RIAC HDBK 217 Plus, UTE C80-811, NPPD 95, EN 50129:2003, Annex C, IEC 62061:2005, Annex D, RIAC FMD97 and MIL HDBK 338.

NOTE 1 The failure rate values given in these databases are generally considered to be pessimistic.

- b) using statistics based on field returns or tests. In this case, the estimated failure rate should have an adequate confidence level, or
- c) using expert judgement founded on an engineering approach based on quantitative and qualitative arguments. Expert judgement shall be exercised in accordance with structured criteria as a basis for this judgement. These criteria shall be set before the estimation of failure rates is made.

NOTE 2 The criteria for expert judgment can include field experience, testing, reliability analysis, and novelty of design.

8.4.4 This requirement applies to ASIL (B), C, and D of the safety goal. If sufficient evidence of the calculated failure rate of a single-point fault or latent fault cannot be made available, alternative means shall be proposed (e.g. add safety mechanisms to detect and control this fault).

NOTE Sufficient evidence means, for instance, that evidence is given that the failure rate has been determined using one of the methods listed in 8.4.3.

8.4.5 This requirement applies to ASIL (B), C, and D of the safety goal. For each safety goal, a quantitative target value for the “single-point fault metric” as required in ISO 26262-4:2011, 7.4.4.2, shall be based on one of the following sources of reference target values:

- a) derived from the hardware architectural metrics calculation applied on similar well-trusted design principles, or

NOTE 1 Two similar designs have similar functionalities and similar safety goals with the same assigned ASIL.

- b) derived from Table 4.

Table 4 — Possible source for the derivation of the target “single-point fault metric” value

	ASIL B	ASIL C	ASIL D
Single-point fault metric	≥90 %	≥97 %	≥99 %

NOTE 2 This quantitative target is intended to provide:

- design guidance; and
- evidence that the design complies with the safety goals.

8.4.6 This requirement applies to ASIL (B), (C), and D of the safety goal. For each safety goal, a quantitative target value for “latent-fault metric” as required in ISO 26262-4:2011, 7.4.4.2 shall be based on one of the following sources of reference target values:

- a) derived from the hardware architectural metrics calculation applied on similar well-trusted design principles; or

NOTE 1 Two similar designs have similar functionalities and similar safety goals with the same assigned ASIL.

- b) derived from Table 5.

Table 5 — Possible source for the derivation of the target “latent-fault metric” value

	ASIL B	ASIL C	ASIL D
Latent-fault metric	≥60 %	≥80 %	≥90 %

NOTE 2 This quantitative target is intended to provide:

- design guidance; and
- evidence that the design complies with the safety goals.

8.4.7 This requirement applies to ASIL (B), C, and D of the safety goal. For each safety goal, the whole hardware of the item shall comply with one of the following alternatives:

- a) to meet the target “single-point fault metric” value, as described in 8.4.5, or
- b) to meet the appropriate targets prescribed at the hardware element level which are sufficient to comply with the single-point fault metric’s target value assigned to the whole hardware of the item, given in requirement 8.4.5, with the rationale for compliance with these targets at the hardware element level.

NOTE 1 If an item contains different kinds of hardware elements with significantly different failure rate levels, the risk exists that compliance with the hardware architectural metrics only focus on the kind of hardware elements with the highest magnitude of failure rates. (One example where this can occur is for the single-point fault metric for which compliance can be achieved by considering the failure rates for failures of wires / fuses / connectors, while disregarding the failure rates of hardware parts with significantly lower failure rates.) The prescription of appropriate metric target values for each kind of hardware helps to avoid this side effect.

NOTE 2 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 (as explained in NOTE 1) or by a qualitative rationale based on the verification of the effectiveness of the internal safety mechanisms implemented to cover these transient faults.

NOTE 3 If the target is not met, the rationale for how the safety goal is achieved will be assessed as given in 4.1.

NOTE 4 Some or all of the applicable safety goals can be considered together for the determination of the single-point fault metric; but in this case the metric's target to be considered is that of the safety goal with the highest ASIL.

8.4.8 This requirement applies to ASIL (B), (C), and D of the safety goal. For each safety goal, the whole hardware of the item shall comply with one of the following alternatives:

- a) to meet the target “latent-fault metric” value, as described in 8.4.6, or
- b) to meet the appropriate targets prescribed at the hardware element level which are sufficient to comply with the latent-fault metric's target value assigned to the whole hardware of the item as described in requirement 8.4.6 and to provide the rationale for compliance with these targets at the hardware element level, or
- c) to meet the target values for the diagnostic coverage, with respect to latent faults, identical to the target value given in reference 8.4.6 for the latent-fault metric (treated as a diagnostic coverage), for each hardware element with faults that can lead to the unavailability of a safety mechanism (to prevent a fault from violating the safety goal). This alternative applies when each safety mechanism, whose unavailability can contribute to the violation of the safety goal, is based on fault detection,

NOTE 1 Alternative c) is limited to the cases where each relevant safety mechanism is based on fault detection. It is supposed that in this case the potentially latent faults of the intended functionality are alerted through the detection of these safety mechanisms. In other cases this alternative cannot be applied and alternatives a) and b) are the only possibilities.

NOTE 2 In the case of c), a metric is not calculated, only the coverage of the hardware elements by safety mechanisms with respect to latent faults is evaluated.

NOTE 3 If an item contains different kinds of hardware elements with significantly different failure rate levels, the risk exists that compliance with the hardware architectural metrics only focuses on the kind of hardware elements with the highest magnitude of failure rates. (One example where this can occur is for the single-point fault metric for which compliance could be achieved by considering the failure rates for failures of wires / fuses / connectors, while disregarding the failure rates of hardware parts with significantly lower failure rates.) The prescription of appropriate metric target values for each kind of hardware helps to avoid this side effect.

NOTE 4 If the target is not met, the rationale for how the safety goal is achieved will be assessed as given in 4.1.

NOTE 5 Some or all of the applicable safety goals can be considered together for the determination of the latent-fault metric; but in this case the metric's target to be considered is that of the safety goal with the highest ASIL.

8.4.9 This requirement applies to ASIL (B), C, and D of the safety goal. A verification review of the result of the applied methods in 8.4.7 and 8.4.8 shall be performed in order to provide evidence of its technical correctness and completeness in accordance with ISO 26262-8:2011, Clause 9.

NOTE Careful verification of the single-point fault metric ensures that only failure rates of safety-related hardware elements are taken into consideration, so that the metric is not inappropriately skewed by unnecessary safety-related hardware elements without the potential for having single-point faults or residual faults (e.g. by adding unnecessary hardware elements to a safety mechanism).

8.5 Work products

8.5.1 Analysis of the effectiveness of the architecture of the item to cope with the random hardware failures resulting from requirements 8.4.1 to 8.4.8.

8.5.2 Review report of evaluation of the effectiveness of the architecture of the item to cope with the random hardware failures resulting from requirement 8.4.9.

9 Evaluation of safety goal violations due to random hardware failures

9.1 Objectives

The objective of the requirements in this clause is to make available criteria that can be used in a rationale that the residual risk of a safety goal violation, due to random hardware failures of the item, is sufficiently low.

NOTE “Sufficiently low” means “comparable to residual risks on items already in use”.

9.2 General

Two alternative methods (see 9.4) are proposed to evaluate whether the residual risk of safety goal violations is sufficiently low.

Both methods evaluate the residual risk of violating a safety goal due to single-point faults, residual faults, and plausible dual-point faults. Multiple-point faults can also be considered if shown to be relevant to the safety concept. In this analysis, coverage of safety mechanisms will be considered for residual and dual-point faults, and exposure duration will be considered as well for dual-point faults.

The first method consists of using a probabilistic metric called “Probabilistic Metric for random Hardware Failures” (PMHF), to evaluate the violation of the considered safety goal using, for example, quantified FTA and to compare the result of this quantification with a target value.

The second method consists of the individual evaluation of each residual and single-point fault, and of each dual-point failure leading to the violation of the considered safety goal. This analysis method can also be considered to be a cut-set analysis.

NOTE In the context of reliability analysis, a cut-set in a fault tree is a set of basic events whose occurrence leads to the occurrence of the top event.

The chosen method can be applied iteratively during the hardware architectural design and the hardware detailed design.

The scope of this clause is limited to the random hardware failures of the item. The parts considered in the analyses are the electrical and electronic hardware parts. For electromechanical hardware parts, only the electrical failure modes and failures rate are considered.

9.3 Inputs to this clause

9.3.1 Prerequisites

The following information shall be available:

- hardware safety requirements specification in accordance with 6.5.1;
- hardware design specification in accordance with 7.5.1; and
- hardware safety analysis report in accordance with 7.5.2.

9.3.2 Further supporting information

The following information can be considered:

- technical safety concept (see ISO 26262-4:2011, 7.5.1); and
- system design specification (see ISO 26262-4:2011, 7.5.2).

9.4 Requirements and recommendations

9.4.1 General

This requirement applies to ASIL (B), C and D of the safety goal. The item shall comply with either 9.4.2 or 9.4.3.

9.4.2 Evaluation of Probabilistic Metric for random Hardware Failures (PMHF)

9.4.2.1 This requirement applies to ASIL (B), C, and D of the safety goal. Quantitative target values for the maximum probability of the violation of each safety goal due to random hardware failures as required in ISO 26262-4:2011, 7.4.4.3, shall be defined using one of the sources a), b) or c) of reference target values, as outlined below:

- a) derived from Table 6, or
- b) derived from field data from similar well-trusted design principles, or
- c) derived from quantitative analysis techniques applied to similar well-trusted design principles using failure rates in accordance with 8.4.3.

NOTE 1 These quantitative target values derived from sources a), b), or c) do not have any absolute significance and are only useful to compare a new design with existing ones. They are intended to make available design guidance as described in 9.1 and to make available evidence that the design complies with the safety goals.

NOTE 2 Two similar designs have similar functionalities and similar safety goals with the same assigned ASIL.

Table 6 — Possible source for the derivation of the random hardware failure target values

ASIL	Random hardware failure target values
D	$<10^{-8} \text{ h}^{-1}$
C	$<10^{-7} \text{ h}^{-1}$
B	$<10^{-7} \text{ h}^{-1}$

NOTE The quantitative target values described in this table can be tailored as specified in 4.1 to fit specific uses of the item (e.g. if the item is able to violate the safety goal for durations longer than the typical use of a passenger car).

9.4.2.2 This requirement applies to ASIL (B), C, and D of the safety goal. Quantitative target values of requirement 9.4.2.1 shall be expressed in terms of average probability per hour over the operational lifetime of the item.

9.4.2.3 This requirement applies to ASIL (B), C, and D of the safety goal. A quantitative analysis of the hardware architecture with respect to the single-point, residual and dual-point faults shall provide evidence that target values of requirement 9.4.2.1 have been achieved. This quantitative analysis shall consider:

- a) the architecture of the item;
- b) the estimated failure rate for the failure modes of each hardware part that would cause a single-point fault or a residual fault;

- c) the estimated failure rate for the failure modes of each hardware part that would cause a dual-point fault;
- d) the diagnostic coverage of safety-related hardware elements by safety mechanisms; and
- e) the exposure duration in the case of dual-point faults.

NOTE 1 Failure modes of hardware elements that can cause a failure of a safety-related hardware element and its safety mechanism simultaneously are considered in the quantitative analysis. They can be single-point faults, residual faults or multiple-point faults.

NOTE 2 Exposure duration starts as soon as the fault can occur and includes:

- a) the multiple-point fault detection interval associated with each safety mechanism, or the lifetime of the vehicle if the fault is not indicated to the driver (latent fault);
- b) the maximum duration of a trip (in the case that the driver is requested to stop in a safe manner); and
- c) the average time interval until the vehicle is at the workshop for repair (in the case that the driver is alerted to have the vehicle repaired).

Therefore, exposure duration depends on the type of monitoring involved (e.g. continuous monitoring, periodic self-tests, driver monitoring, no monitoring) and the kind of reaction when the fault has been detected. It can be as short as a few milliseconds in the case of a continuous monitoring triggering a transition to a safe state. It can be as long as the car lifetime when there is no monitoring.

Example of assumptions on the average time to vehicle repair, depending on the fault type:

- 200 vehicle trips for reduction of comfort features;
- 50 vehicle trips for reduction of driving support features;
- 20 vehicle trips for amber warning lights or impacts on driving behaviour;
- one vehicle trip for red warning lights.

The time taken for repair is usually not considered (except to evaluate hazards that can expose maintenance personnel).

The mean duration of a vehicle trip can be considered as being equal to 1 h.

NOTE 3 In most cases, multiple-point failures of a higher order than two have a negligible contribution with respect to the quantitative target values. However, in some particular cases (very high failure rate or poor diagnostic coverage), it can be necessary to provide two redundant safety mechanisms to reach the target. When the technical safety concept is based on redundant safety mechanisms, multiple-point failures of a higher order than two are considered in the analysis.

NOTE 4 For safety mechanisms using integrated diagnostics, Tables D.1 to D.14 can be used as a starting point to evaluate the diagnostic coverage of these safety mechanisms, with the claimed DC supported by a proper rationale.

NOTE 5 Situations when the item is in power-down mode are not included in the calculation of the average probability per hour, thereby preventing the artificial reduction of the average probability per hour. Thus, for an item that is only operational 1 h each day, the remaining 23 h are not considered in the calculation of this operational target value.

NOTE 6 If the target is not met, the rationale for how the safety goal is achieved will be assessed as given in 4.1.

NOTE 7 Depending on the knowledge of the failure modes of the hardware elements and their consequences at a higher level, the evaluation can be either a global diagnostic coverage of the hardware element, or a more detailed failure mode coverage evaluation.

9.4.2.4 This requirement applies to ASIL C and D of the safety goal. A single-point fault occurring in a hardware part shall only be considered acceptable if dedicated measures are taken.

NOTE Dedicated measures can include:

- a) design features such as hardware part over design (e.g. electrical or thermal stress rating) or physical separation (e.g. spacing of contacts on a printed circuit board);
- b) a special sample test of incoming material to reduce the risk of occurrence of this failure mode;
- c) a burn-in test;
- d) a dedicated control set as part of the control plan; and
- e) assignment of safety-related special characteristics.

9.4.2.5 This requirement applies to ASIL C and D of the safety goal. A hardware part shall be dealt with by dedicated measures (the note in 9.4.2.4 lists examples of dedicated measures) if its diagnostic coverage (with respect to residual faults) is lower than 90 %.

NOTE The proportion of safe faults of the hardware part can be considered when determining the coverage of the safety mechanisms. In this case the calculation of the coverage is done analogously to the calculation of the single-point fault metric, but at the hardware part level instead of at the item level.

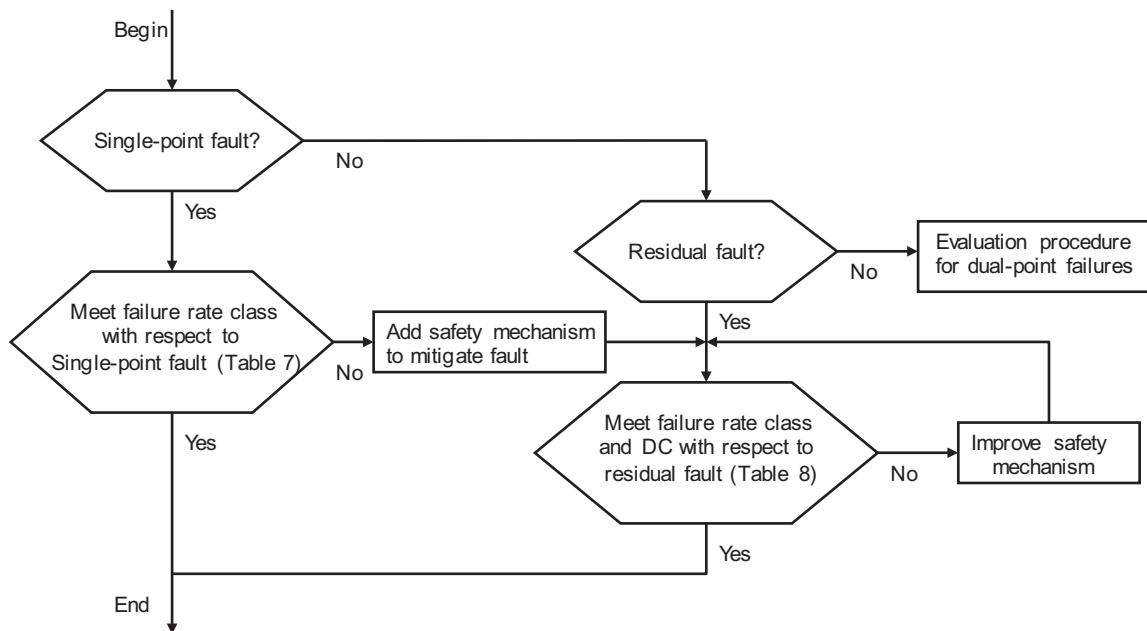
9.4.2.6 This requirement applies to ASIL (B), C, and D of the safety goal. The failure rates for hardware parts used in the analyses shall be estimated in accordance with 8.4.3.

9.4.2.7 This requirement applies to ASIL (B), C, and D of the safety goal. In order to avoid bias in the quantification, if failure rates from multiple sources are combined, they shall be scaled using a scaling factor to be consistent. Scaling is possible if a rationale for the scaling factor between two failure rate sources is available.

NOTE Guidance is given in Annex F on the application of scaling factors.

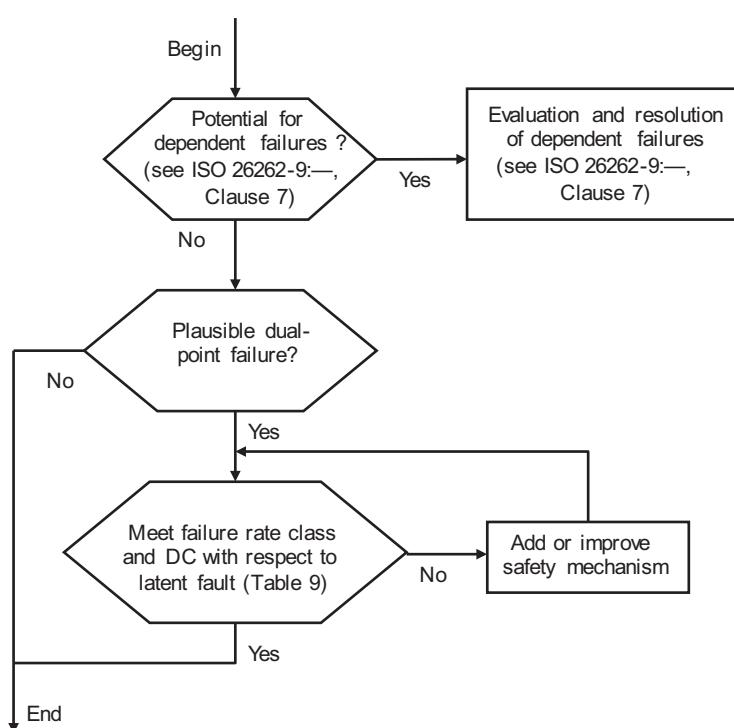
9.4.3 Evaluation of each cause of safety goal violation

9.4.3.1 A method for evaluation of each cause of a safety goal violation due to random hardware failures is illustrated by flowcharts in Figures 3 and 4. Each single-point fault is evaluated using criteria on the occurrence of the fault. Each residual fault is evaluated using criteria combining the occurrence of the fault and the efficiency of the safety mechanism.

**Figure 3 — Evaluation procedure for single-point and residual faults**

The procedure to be applied for dual-point failures is illustrated by the flowchart in Figure 4. Each dual-point failure is first evaluated regarding its plausibility. A dual-point failure is considered not plausible if both faults leading to the failure are detected or perceived in a sufficiently short time with sufficient coverage. If the dual-point failure is plausible, the faults causing it are then evaluated using criteria combining occurrence of the fault and coverage of the safety mechanisms. The evaluation procedures described in Figures 3 and 4 apply to the hardware parts (transistors, etc.) level.

NOTE For complex hardware parts like microcontrollers, it can be appropriate to apply this procedure on a more detailed level like CPU, RAM, ROM, etc.

**Figure 4 — Evaluation procedure for dual-point failures**

9.4.3.2 This requirement applies to ASIL (B), C, and D of the safety goal. An individual evaluation of each single-point fault, residual fault and dual-point failure violating the considered safety goal shall be performed at the hardware part level. This evaluation shall provide evidence that each single-point fault, residual fault and dual-point failure violating the considered safety goal is acceptable in accordance with requirements 9.4.3.3 to 9.4.3.12.

NOTE 1 This analysis can be viewed as a review of cut sets where absence or incompleteness of coverage is treated as a fault.

NOTE 2 In most cases, multiple-point failures of a higher order than two are negligible. However, in some particular cases (very high failure rate or poor diagnostic coverage), it can be necessary to provide two redundant safety mechanisms. Therefore it is necessary to consider multiple-point failures of a higher order than two in the analysis when the technical safety concept is based on redundant safety mechanisms.

NOTE 3 For complex hardware parts like microcontrollers it can be appropriate to apply this procedure at a more detailed level like CPU, RAM, ROM, etc.

9.4.3.3 This requirement applies to ASIL (B), C, and D of the safety goal. The failure rate class ranking for a hardware part failure rate shall be determined as follows:

NOTE 1 The failure rate classes 1, 2 and 3 are introduced to address the failure occurrence rates. These classes are analogous to the occurrence levels 1, 2 and 3, respectively, used in an FMEA, where a 1 is assigned to failure modes which have the lowest occurrence rate.

- a) the failure rate corresponding to failure rate class 1 shall be less than the target for ASIL D divided by 100; unless 9.4.3.4 is applied;

NOTE 2 The target values given in Table 6 can be used.

- b) the failure rate corresponding to failure rate class 2 shall be less than or equal to 10 times the failure rate corresponding to failure rate class 1;
- c) the failure rate corresponding to failure rate class 3 shall be less than or equal to 100 times the failure rate corresponding to failure rate class 1; and
- d) the failure rate corresponding to failure rate class i , $i > 3$ shall be less than or equal to $10^{(i-1)}$ times the failure rate corresponding to failure rate class 1.

NOTE 3 The failure rate class assignment is based upon the hardware part failure rate.

NOTE 4 For the case where a small number of parts (such as a microcontroller) have failure rates higher than the failure rate class i upper limit, then these parts can be assigned a class i occurrence if the resulting average failure rate of parts assigned class i is lower than failure rate class i 's upper limit.

9.4.3.4 If a rationale is provided, the failure rate class ranking may be divided by a number lower than 100. In this case, it shall be ensured that a correct ranking is maintained while considering the single-point faults, residual faults and higher degree cut-sets together.

EXAMPLE The rationale can be based on the number of minimal cut-sets.

9.4.3.5 This requirement applies to ASIL (B), C and D of the safety goal. A single-point fault occurring in a hardware part shall only be considered as acceptable if the corresponding hardware part failure rate ranking complies with the targets given in Table 7.

Table 7 — Targets of failure rate classes of hardware parts regarding single-point faults

ASIL of the safety goal	Failure rate class
D	Failure rate class 1 + dedicated measures ^a
C	Failure rate class 2 + dedicated measures ^a or Failure rate class 1
B	Failure rate class 2 or Failure rate class 1

^a The note in requirement 9.4.2.4 gives examples of dedicated measures.

NOTE When assessing the failure rate class, the proportion of safe faults of the hardware part can be considered.

9.4.3.6 This requirement applies to ASIL (B), C, and D of the safety goal. A residual fault occurring in a hardware part shall be considered acceptable if the failure rate class ranking complies with the targets given in Table 8 for the diagnostic coverage (with respect to residual faults) of the corresponding hardware part.

NOTE 1 The considered failure rate is the hardware part failure rate and does not take into account the effectiveness of the safety mechanisms.

Table 8 — Maximum failure rate classes for a given diagnostic coverage of the hardware part – residual faults

ASIL of the safety goal	Diagnostic coverage with respect to residual faults			
	≥99,9 %	≥99 %	≥90 %	<90 %
D	Failure rate class 4	Failure rate class 3	Failure rate class 2	Failure rate class 1 + dedicated measures ^a
C	Failure rate class 5	Failure rate class 4	Failure rate class 3	Failure rate class 2 + dedicated measures ^a
B	Failure rate class 5	Failure rate class 4	Failure rate class 3	Failure rate class 2

^a The note in requirement 9.4.2.4 gives examples of dedicated measures.

NOTE 2 Table 8 specifies the connection between maximum failure rate class allowed given the target ASIL and the diagnostic coverage. Lower failure rate classes are acceptable but not required.

NOTE 3 “Lower failure rate classes” means failure rate classes with a lower number. For example, “lower failure rate classes” with respect to failure rate class 3 means failure rate classes 2 and 1.

NOTE 4 The proportion of safe faults of the hardware part can be taken into account when determining the coverage of the safety mechanisms. In this case, the calculation of the coverage is done analogously to the calculation of the single-point fault metric, but at the hardware part level instead of at the item level.

9.4.3.7 This requirement applies for ASIL C and D of the safety goal. For failure rate classes i , $i > 3$, a residual fault shall be considered as acceptable if the diagnostic coverage is greater than or equal to $[100 - 10^{(3-i)}] \%$ for ASIL D or greater than or equal to $[100 - 10^{(4-i)}] \%$ for ASIL C.

NOTE 1 The considered failure rate is the hardware part failure rate, and does not take into account the effectiveness of the safety mechanisms.

NOTE 2 The proportion of safe faults of the hardware part can be taken into account when determining the coverage of the safety mechanisms. In this case the calculation of the coverage is done analogously to the calculation of the single-point fault metric, but at the hardware part level instead of at the item level.

9.4.3.8 This requirement applies to ASIL D of the safety goal. A dual-point failure shall be considered plausible if

- a) one or both hardware parts involved has a diagnostic coverage (with respect to the latent faults) of less than 90 %, or
- b) 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 6.4.8.

NOTE The proportion of safe faults of the hardware part can be taken into account when determining the coverage of the safety mechanisms. In this case the calculation of the coverage is done analogously to the calculation of the latent fault metric, but at the hardware part level instead of at the item level.

9.4.3.9 This requirement applies to ASIL C of the safety goal. A dual-point failure shall be considered plausible if

- a) one or both hardware parts involved has a diagnostic coverage (with respect to the latent faults) of less than 80 %; or
- b) 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 6.4.8.

NOTE The proportion of safe faults of the hardware part can be taken into account when determining the coverage of the safety mechanisms. In this case the calculation of the coverage is done analogously to the calculation of the latent fault metric, but at the hardware part level instead of at the item level.

9.4.3.10 This requirement applies to ASIL C and D of the safety goal. A dual-point failure that is not plausible shall be considered compatible with the safety goal target and thus acceptable.

9.4.3.11 This requirement applies to ASIL C and D of the safety goal. A dual-point fault occurring in a hardware part and contributing to a plausible dual-point failure shall be considered acceptable if the corresponding hardware part complies with the targets for the failure rate class ranking and diagnostic coverage (with respect to latent faults) given in Table 9.

NOTE 1 The considered failure rate is the hardware part failure rate. Therefore, it does not consider the effectiveness of safety mechanisms.

Table 9 — Targets of failure rate class and coverage of hardware part regarding dual-point faults

ASIL of safety goal	Diagnostic coverage with respect to latent faults		
	>= 99 %	>= 90 %	<90 %
D	Failure rate class 4	Failure rate class 3	Failure rate class 2
C	Failure rate class 5	Failure rate class 4	Failure rate class 3

NOTE 2 Table 9 specifies the maximum failure rate class allowed given the target ASIL level and the level of diagnostic coverage achieved. Lower failure rate classes are acceptable but not required.

NOTE 3 “Lower failure rate classes” means failure rate classes with a lower number. For example, “lower failure rate classes” with respect to failure rate class 3 means failure rate classes 2 and 1.

NOTE 4 The proportion of safe faults of the hardware part can be taken into account when determining the coverage of the safety mechanisms. In this case the calculation of the coverage is done analogously to the calculation of the latent fault metric, but at the hardware part level instead of at the item level.

9.4.3.12 This requirement applies to ASIL (B), C, and D of the safety goal. The failure rate class ranking of the hardware part failure rate used in the analyses shall be justified by using sources of failure rates described in 8.4.3. If failure rates from multiple data sources are used in the analyses, then the rates shall be scaled as described in 9.4.2.7.

9.4.4 Verification review

This requirement applies to ASIL (B), C, and D of the safety goal. A verification review of the analysis resulting from the set of requirements 9.4.2 or 9.4.3 shall be performed in order to provide evidence of its technical correctness and completeness in accordance with ISO 26262-8:2011, Clause 9.

9.5 Work products

9.5.1 Analysis of safety goal violations due to random hardware failures resulting from requirements in 9.4.2 or in 9.4.3.

9.5.2 Specification of dedicated measures for hardware, if needed, including the rationale regarding the effectiveness of the dedicated measures, resulting from requirements 9.4.2.4, 9.4.2.5, 9.4.3.5 and 9.4.3.6.

9.5.3 Review report of evaluation of safety goal violations due to random hardware failures resulting from requirement 9.4.4.

10 Hardware integration and testing

10.1 Objectives

The objective of this clause is to ensure, by testing, the compliance of the developed hardware with the hardware safety requirements.

The requirements in 10.4.1 to 10.4.6 apply to the hardware of an element.

10.2 General

The activities described in this clause aim at integrating hardware elements and testing the hardware design to verify its compliance with the hardware safety requirements in accordance with the appropriate ASIL.

Hardware integration and testing differs from the qualification of hardware components activity of ISO 26262-8:2011, Clause 13, which gives evidence of the suitability of intermediate level hardware components and parts for their use as parts of items, systems or elements developed in compliance with ISO 26262.

10.3 Inputs of this clause

10.3.1 Prerequisites

The following information shall be available:

- safety plan (refined) in accordance with 5.5;
- item integration and testing plan (refined) in accordance with ISO 26262-4:2011, 5.5.3;
- hardware safety requirements specification in accordance with 6.5.1; and
- hardware design specification in accordance with 7.5.1.

10.3.2 Further supporting information

The following information can be considered:

- project plan (refined) (see ISO 26262-4:2011, 5.5.1); and
- hardware safety analysis report (see 7.5.2).

10.4 Requirements and recommendations

10.4.1 Hardware integration and testing activities shall be performed in accordance with ISO 26262-8:2011, Clause 9.

10.4.2 Hardware integration and testing activities shall be coordinated with the item integration and testing plan given in ISO 26262-4:2011, 5.5.5.

NOTE If ASIL decomposition is applied, as defined in ISO 26262-9:2011, Clause 5, the corresponding integration activities of the decomposed elements, and the subsequent activities, are applied at the ASIL before decomposition.

10.4.3 The test equipment shall be subject to the control of a monitoring quality system.

10.4.4 To enable the appropriate specification of test cases for the selected hardware integration tests, test cases shall be derived using an appropriate combination of methods listed in Table 10.

Table 10 — Methods for deriving test cases for hardware integration testing

	Methods	ASIL			
		A	B	C	D
1a	Analysis of requirements	++	++	++	++
1b	Analysis of internal and external interfaces	+	++	++	++
1c	Generation and analysis of equivalence classes ^a	+	+	++	++
1d	Analysis of boundary values ^b	+	+	++	++
1e	Knowledge or experience based error guessing ^c	++	++	++	++
1f	Analysis of functional dependencies	+	+	++	++
1g	Analysis of common limit conditions, sequences and sources of dependent failures	+	+	++	++
1h	Analysis of environmental conditions and operational use cases	+	++	++	++
1i	Standards if existing ^d	+	+	+	+
1j	Analysis of significant variants ^e	++	++	++	++

^a In order to derive the necessary test cases efficiently, analysis of similarities can be conducted.
^b For example, values approaching and crossing the boundaries between specified values, and out of range values.
^c “Error guessing tests” can be based on data collected through a lessons learned process, or expert judgment, or both. It can be supported by FMEA.
^d Existing standards include ISO 16750 and ISO 11452.
^e The analysis of significant variants includes worst-case analysis.

10.4.5 The hardware integration and testing activities shall verify the completeness and correctness of the implementation of the safety mechanisms with respect to the hardware safety requirements.

To achieve these objectives, the methods listed in Table 11 shall be considered.

Table 11 — Hardware integration tests to verify the completeness and correctness of the safety mechanisms implementation with respect to the hardware safety requirements

	Methods	ASIL			
		A	B	C	D
1	Functional testing ^a	++	++	++	++
2	Fault injection testing ^b	+	+	++	++
3	Electrical testing ^c	++	++	++	++

^a Functional testing aims at verifying that the specified characteristics of the item have been achieved. The item is given input data, which adequately characterises the expected normal operation. The outputs are observed and their response is compared with that given by the specification. Anomalies with respect to the specification and indications of an incomplete specification are analysed.

^b Fault injection testing aims at introducing faults in the hardware product and analysing the response. This testing is appropriate whenever a safety mechanism is defined. Model-based fault injection (e.g. fault injection done at the gate-level netlist level) is also applicable, especially when fault injection testing is very difficult to do at the hardware product level. For example, showing the response of safety mechanisms to transient faults inside hardware parts, such as a microcontroller, is very difficult to do with fault insertion at the hardware product level since it would require irradiation tests.

^c Electrical testing aims at verifying compliance with hardware safety requirements within the specified (static and dynamic) voltage range.

10.4.6 The hardware integration and testing activities shall verify robustness of hardware against external stresses.

To achieve these objectives, the methods listed in Table 12 shall be considered.

Table 12 — Hardware integration tests to verify robustness and operation under external stresses

Methods	ASIL			
	A	B	C	D
1a Environmental testing with basic functional verification ^a	++	++	++	++
1b Expanded functional test ^b	o	+	+	++
1c Statistical test ^c	o	o	+	++
1d Worst case test ^d	o	o	o	+
1e Over limit test ^e	+	+	+	+
1f Mechanical test ^f	++	++	++	++
1g Accelerated life test ^g	+	+	++	++
1h Mechanical Endurance test ^h	++	++	++	++
1i EMC and ESD test ⁱ	++	++	++	++
1j Chemical test ^j	++	++	++	++

^a During environmental testing with basic functional verification the hardware is put under various environmental conditions during which the hardware requirements are assessed. ISO 16750-4 can be applied.

^b Expanded functional testing checks the functional behaviour of the item in response to input conditions that are expected to occur only rarely (for instance extreme mission profile values), or that are outside the specification of the hardware (for instance, an incorrect command). In these situations, the observed behaviour of the hardware element is compared with the specified requirements.

^c Statistical tests aim at testing the hardware element with input data selected in accordance with the expected statistical distribution of the real mission profile. The acceptance criteria are defined so that the statistical distribution of the results confirms the required failure rate.

^d Worst-case testing aims at testing cases found during worst-case analysis. In such a test, environmental conditions are changed to their highest permissible marginal values defined by the specification. The related responses of the hardware are inspected and compared with the specified requirements.

^e In over limit testing, the hardware elements are submitted to environmental or functional constraints increasing progressively to values more severe than specified until they stop working or they are destroyed. The purpose of this test is to determine the margin of robustness of the elements under test with respect to the required performance.

^f Mechanical test applies to mechanical properties such as tensile strength.

^g Accelerated life test aims at predicting the behaviour evolution of a product in its normal operational conditions by submitting it to stresses higher than those expected during its operational lifetime. Accelerated testing is based on an analytical model of failure mode acceleration.

^h The aim of these tests is to study the mean time to failure or the maximum number of cycles that the element can withstand. Test can be performed up to failure or by damage evaluation.

ⁱ ISO 7637-2, ISO 7637-3, ISO 10605, ISO 11452-2 and ISO 11452-4 can be applied for EMC tests; ISO 16750-2 can be applied for ESD tests.

^j For chemical tests, ISO 16750-5 can be applied.

10.5 Work products

10.5.1 Hardware integration and testing report resulting from requirements 10.4.1 to 10.4.6.

Annex A (informative)

Overview of and workflow of product development at the hardware level

Table A.1 provides an overview of objectives, prerequisites and work products of the particular phases of product development at the hardware level.

Table A.1 — Overview of product development at the hardware level

Clause	Objectives	Prerequisites	Work products
5 Initiation of product development at the hardware level	<p>The objective of the initiation of the product development for the hardware is to determine and plan the functional safety activities during the individual subphases of hardware development. This also includes the necessary supporting processes described in ISO 26262-8.</p> <p>This planning of hardware-specific safety activities is included in the safety plan (see ISO 26262-2:2011, 6.4.3 and ISO 26262-4:2011, 5.4).</p>	<p>Project plan (refined) (in accordance with ISO 26262-4:2011, 5.5.1)</p> <p>Safety plan (refined) (in accordance with ISO 26262-4:2011, 5.5.2)</p> <p>Item integration and testing plan (refined) (in accordance with ISO 26262-4:2011, 5.5.5)</p>	5.5 Safety plan (refined)
6 Specification of hardware safety requirements	<p>The first objective of this clause is to specify the hardware safety requirements. They are derived from the technical safety concept and system design specification.</p> <p>The second objective is to verify that the hardware safety requirements are consistent with the technical safety concept and the system design specification.</p> <p>A further objective of this phase is to detail the hardware-software interface (HSI) specification initiated in ISO 26262-4:2011, Clause 7.</p>	<p>Safety plan (refined) (in accordance with 5.5)</p> <p>Technical safety concept (in accordance with ISO 26262-4:2011, 7.5.1)</p> <p>System design specification (in accordance with ISO 26262-4:2011, 7.5.2)</p> <p>Hardware software interface specification (in accordance with ISO 26262-4:2011, 7.5.3)</p>	<p style="text-align: center;">6.5.1 Hardware safety requirements specification (including test and qualification criteria)</p> <p style="text-align: center;">6.5.2 Hardware-software interface specification (refined)</p> <p style="text-align: center;">6.5.3 Hardware safety requirements verification report</p>
7 Hardware design	<p>The first objective of this clause is to design the hardware with respect to the system design specification and the hardware safety requirements.</p> <p>The second objective of this clause is to verify the hardware design against the system design specification and the hardware safety requirements.</p>	<p>Hardware safety requirements specification (in accordance with 6.5.1)</p> <p>Hardware-software interface specification (refined) (in accordance with 6.5.2)</p> <p>System design specification (in accordance with ISO 26262-4:2011, 7.5.2)</p> <p>Safety plan (refined) (in accordance with 5.5)</p>	<p style="text-align: center;">7.5.1 Hardware design specification</p> <p style="text-align: center;">7.5.2 Hardware safety analysis report</p> <p style="text-align: center;">7.5.3 Hardware design verification report</p> <p style="text-align: center;">7.5.4 Specification of requirements related to production, operation, service and decommissioning</p>

Table A.1 (continued)

Clause	Objectives	Prerequisites	Work products
8 Evaluation of the hardware architectural metrics	The objective of this clause is to evaluate the hardware architecture of the item against the requirements for fault handling as represented by the hardware architectural metrics.	Hardware safety requirements specification (in accordance with 6.5.1) Hardware design specification (in accordance with 7.5.1) Hardware safety analysis report (in accordance with 7.5.2)	8.5.1 Analysis of the effectiveness of the architecture of the item to cope with the random hardware failures 8.5.2 Review report of evaluation of the effectiveness of the architecture of the item to cope with the random hardware failures
9 Evaluation of safety goal violations due to random HW failures	The objective of the requirements in this clause is to make available criteria that can be used in a rationale that the residual risk of a safety goal violation, due to random hardware failures of the item, is sufficiently low.	Hardware safety requirements specification (in accordance with 6.5.1) Hardware design specification (in accordance with 7.5.1) Hardware safety analysis report (in accordance with 7.5.2)	9.5.1 Analysis of safety goal violations due to random hardware failures 9.5.2 Specification of dedicated measures for hardware, if needed, including the rationale regarding the effectiveness of the dedicated measures 9.5.3 Review report of evaluation of safety goal violations due to random hardware failures
10 Hardware integration and testing	The objective of this clause is to ensure, by testing, the compliance of the developed hardware with the hardware safety requirements. The requirements in 10.4.1 to 10.4.6 apply to the hardware of an element.	Safety plan (refined) (in accordance with 5.5) Item integration and testing plan (refined) (in accordance with ISO 26262-4:2011, 5.5.5) Hardware safety requirements specification (in accordance with 6.5.1) Hardware design specification (in accordance with 7.5.1)	10.5 Hardware integration and testing report

Annex B
(informative)**Failure mode classification of a hardware element**

Failure modes of a hardware element can be classified as shown in Figure B.1. The flow diagram shown in Figure B.2 describes how a failure mode of a hardware element can be placed into one of these classifications.

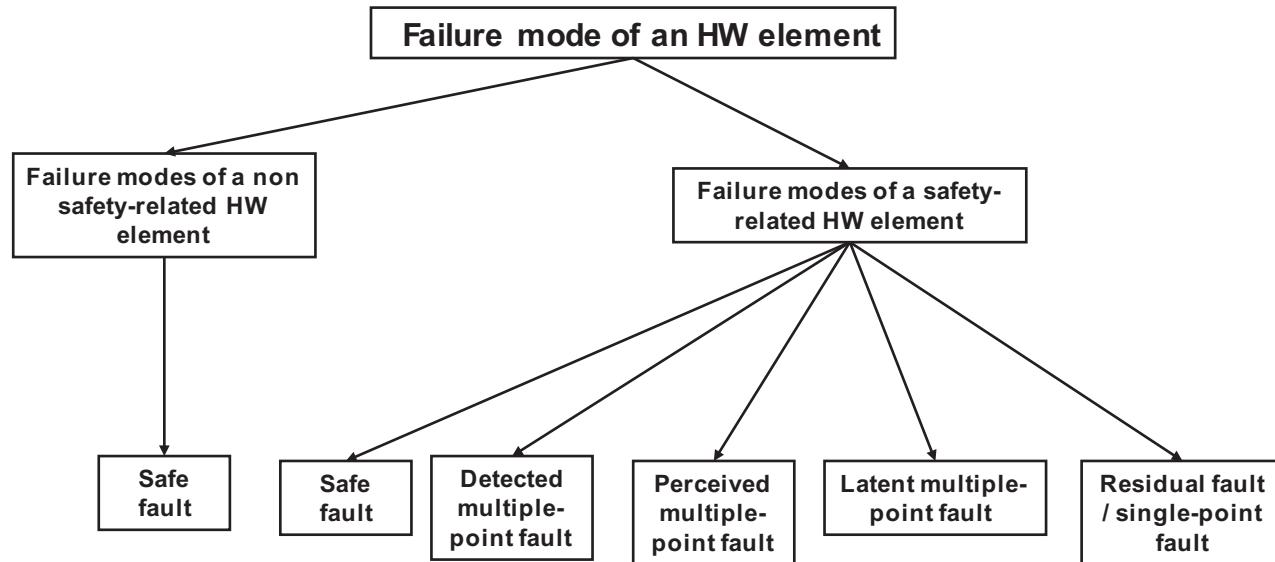
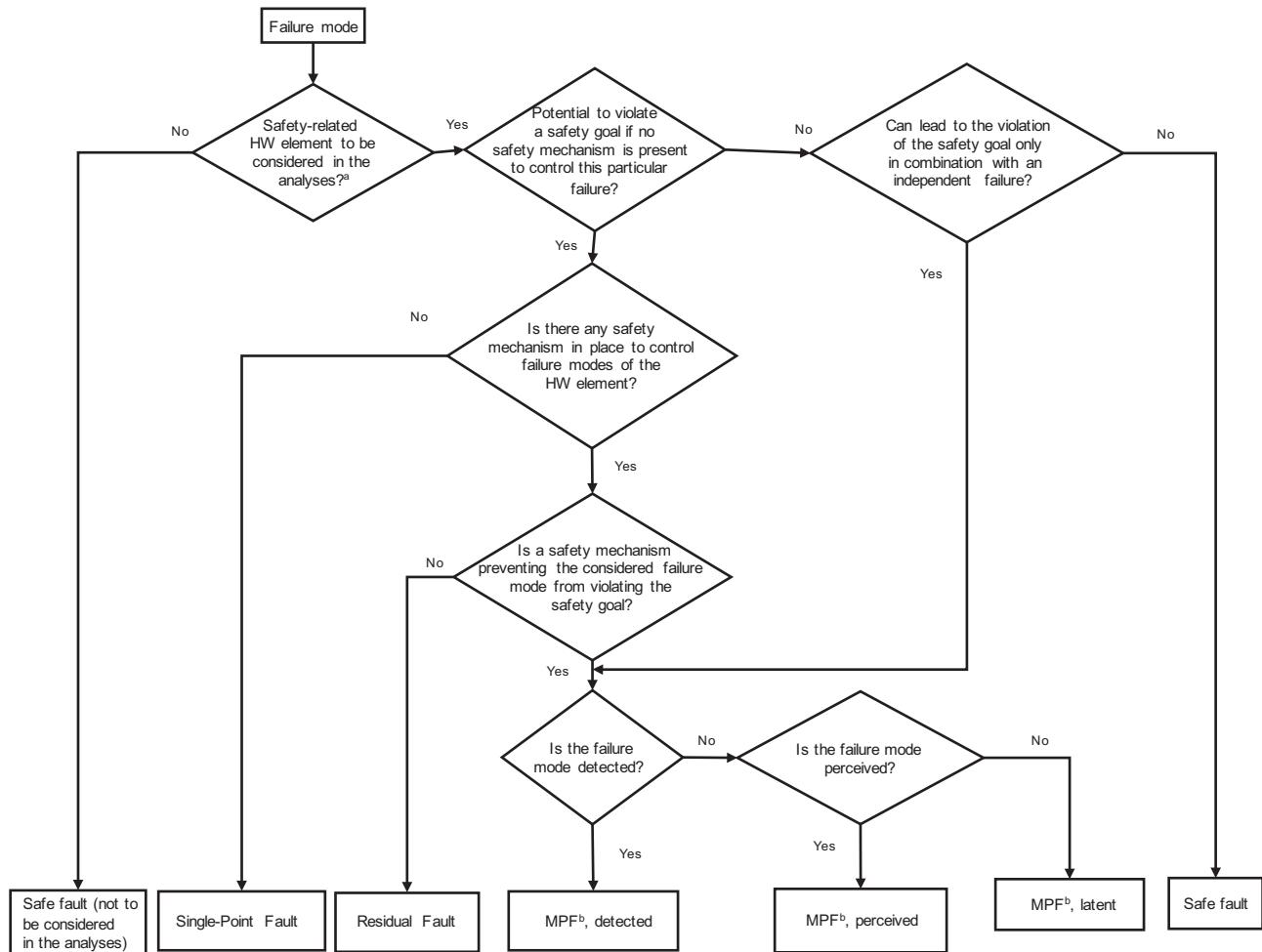


Figure B.1 — Failure mode classifications of a hardware element



a The elements with failures that do not significantly increase the probability of the violation of a safety goal can be omitted from the analyses and their failure modes can be classified as safe faults, e.g. hardware elements whose faults only contribute to multiple-point failures with $n>2$ unless shown to be relevant in the technical safety concept.

b MPF stands for multiple-point fault.

NOTE 1 Multiple-point faults with $n>2$ are considered as safe faults unless shown to be relevant in the technical safety concept.

NOTE 2 The same fault can be placed in different classes when being considered for different safety goals.

Figure B.2 — Example of flow diagram for failure mode classification

Annex C (normative)

Hardware architectural metrics

C.1 Fault classification and diagnostic coverage

C.1.1 This requirement applies to ASIL (B), C, and D of the safety goal. Hardware architectural metrics shall be defined for the hardware of an item and shall address only safety-related hardware elements that have the potential to contribute significantly to the violation of the safety goal.

EXAMPLE Hardware elements whose faults are multiple-point faults with $n > 2$ can be omitted from the calculations unless shown to be relevant in the technical safety concept.

C.1.2 This requirement applies to ASIL (B), C, and D of the safety goal. Each fault occurring in a safety-related hardware element shall be classified, as illustrated in Figure B.1, as:

- a) single-point fault;
- b) residual fault;

EXAMPLE A hardware element that can have “open”, “short to ground”, and “short to high” faults, but only the “open” and “short to ground” faults are covered by safety mechanisms. The “short to high” fault is a residual fault, since it is not covered by a safety mechanism, if it leads to the violation of the specified safety goal.

- c) multiple-point fault;
- d) safe fault.

Figure C.1 gives a graphical representation of fault classification of safety-related hardware elements of an item:

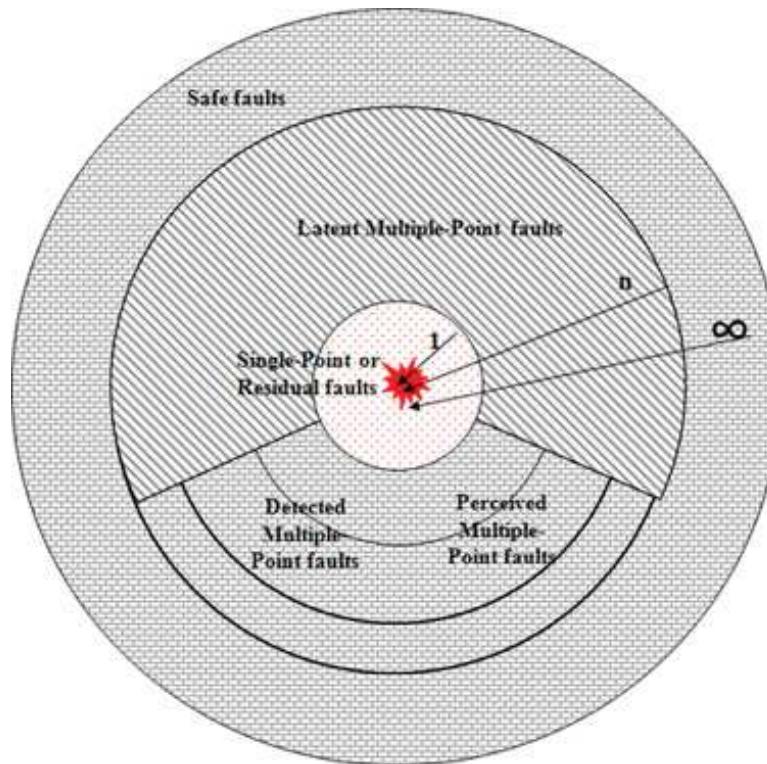


Figure C.1 — Fault classification of safety-related hardware elements of an item

In this graphical representation:

- the distance n represents the number of independent faults present at the same time that cause a violation of the safety goal ($n = 1$ for single-point or residual faults, $n = 2$ for dual-point faults, etc.);
- faults with distance equal to n are located in the area between the circles n and $n-1$; and
- multiple-point faults of distance strictly higher than $n=2$ are to be considered as safe faults unless shown to be relevant in the technical safety concept.

NOTE 1 In the case of a transient fault, for which a safety mechanism restores the item to a fault free state, such a fault can be considered as a detected multiple-point fault even if the driver is never informed of its existence.

EXAMPLE In the case of an error correction code used to protect a memory against transient faults, the item is restored to a fault free state if the safety mechanism – in addition to delivering a correct value to the CPU – repairs the content of the flipped bit inside the memory array (e.g. by writing back the corrected value).

The failure rate, λ , of each safety-related hardware element can therefore be expressed according to Equation (C.1) (assuming all failures are independent and follow the exponential distribution), as follows:

$$\lambda = \lambda_{\text{SPF}} + \lambda_{\text{RF}} + \lambda_{\text{MPF}} + \lambda_S \quad (\text{C.1})$$

where

- λ_{SPF} is the failure rate associated with hardware element single-point faults;
- λ_{RF} is the failure rate associated with hardware element residual faults;
- λ_{MPF} is the failure rate associated with hardware element multiple-point faults;
- λ_S is the failure rate associated with hardware element safe faults.

The failure rate associated with hardware element multiple-point faults, λ_{MPF} , can be expressed according to Equation (C.2), as follows:

$$\lambda_{MPF} = \lambda_{MPF,DP} + \lambda_{MPF,L} \quad (C.2)$$

where

$\lambda_{MPF,DP}$ is the failure rate associated with hardware element perceived or detected multiple-point faults;

$\lambda_{MPF,L}$ is the failure rate associated with hardware element latent faults.

The failure rate assigned to residual faults can be determined using the diagnostic coverage of safety mechanisms that avoid single-point faults of the hardware element. Equation (C.3) gives a conservative estimation of the failure rate associated with the residual faults:

$$\begin{aligned} K_{DC,RF} &= \left(1 - \frac{\lambda_{RF,est}}{\lambda}\right) \times 100 \\ \lambda_{RF} \leq \lambda_{RF,est} &= \lambda \times \left(1 - \frac{K_{DC,RF}}{100}\right) \end{aligned} \quad (C.3)$$

where

$\lambda_{RF,est}$ is the estimated failure rate with respect to residual faults;

$K_{DC,RF}$ is the diagnostic coverage with respect to residual faults, expressed as a percentage.

The failure rate assigned to latent faults can be determined using the diagnostic coverage of safety mechanisms that avoid latent faults of the hardware element. Equation (C.4) gives a conservative estimation of the failure rate associated with latent faults:

$$\begin{aligned} K_{DC,MPF,L} &= \left(1 - \frac{\lambda_{MPF,L,est}}{\lambda}\right) \times 100 \\ \lambda_{MPF,L} \leq \lambda_{MPF,L,est} &= \lambda \times \left(1 - \frac{K_{DC,MPF,L}}{100}\right) \end{aligned} \quad (C.4)$$

where

$\lambda_{MPF,L,est}$ is the estimated failure rate with respect to latent faults;

$K_{DC,MPF,L}$ is the diagnostic coverage with respect to latent faults, expressed as a percentage.

NOTE 2 For this purpose, Annex D can be used as a basis for diagnostic coverage (DC) with the claimed DC supported by a proper rationale.

NOTE 3 If the above estimations are considered too conservative, then a detailed analysis of the failure modes of the hardware element can classify each failure mode into one of the fault classes (single-point faults, residual faults, latent, detected or perceived multiple-point faults or safe faults) with respect to the specified safety goal and determine the failure rates apportioned to the failure modes. Annex B describes a flow diagram that can be used to make the fault classification.

C.2 Single-point fault metric

C.2.1 This metric reflects the robustness of the item to single-point and residual faults either by coverage from safety mechanisms or by design (primarily safe faults). A high single-point fault metric implies that the proportion of single-point faults and residual faults in the hardware of the item is low.

C.2.2 This requirement applies to ASIL (B), C, and D of the safety goal. The calculation in Equation (C.5) shall be used to determine the single-point fault metric:

$$1 - \frac{\sum_{SR,HW} (\lambda_{SPF} + \lambda_{RF})}{\sum_{SR,HW} \lambda} = \frac{\sum_{SR,HW} (\lambda_{MPF} + \lambda_S)}{\sum_{SR,HW} \lambda} \quad (C.5)$$

where $\sum_{SR,HW} \lambda_x$ is the sum of λ_x of the safety-related hardware elements of the item to be considered for the metrics.

NOTE 1 Only the safety-related hardware elements of the item whose failures have the potential to contribute significantly to the violation of the safety goal are considered for this metric.

EXAMPLE Hardware elements whose faults are multiple-point faults with $n > 2$ can be omitted from the calculations unless shown to be relevant in the technical safety concept.

NOTE 2 Figure C.2 gives a graphical representation of the single-point fault metric.

NOTE 3 An example of calculation of “latent-fault metric” is given in Annex E.

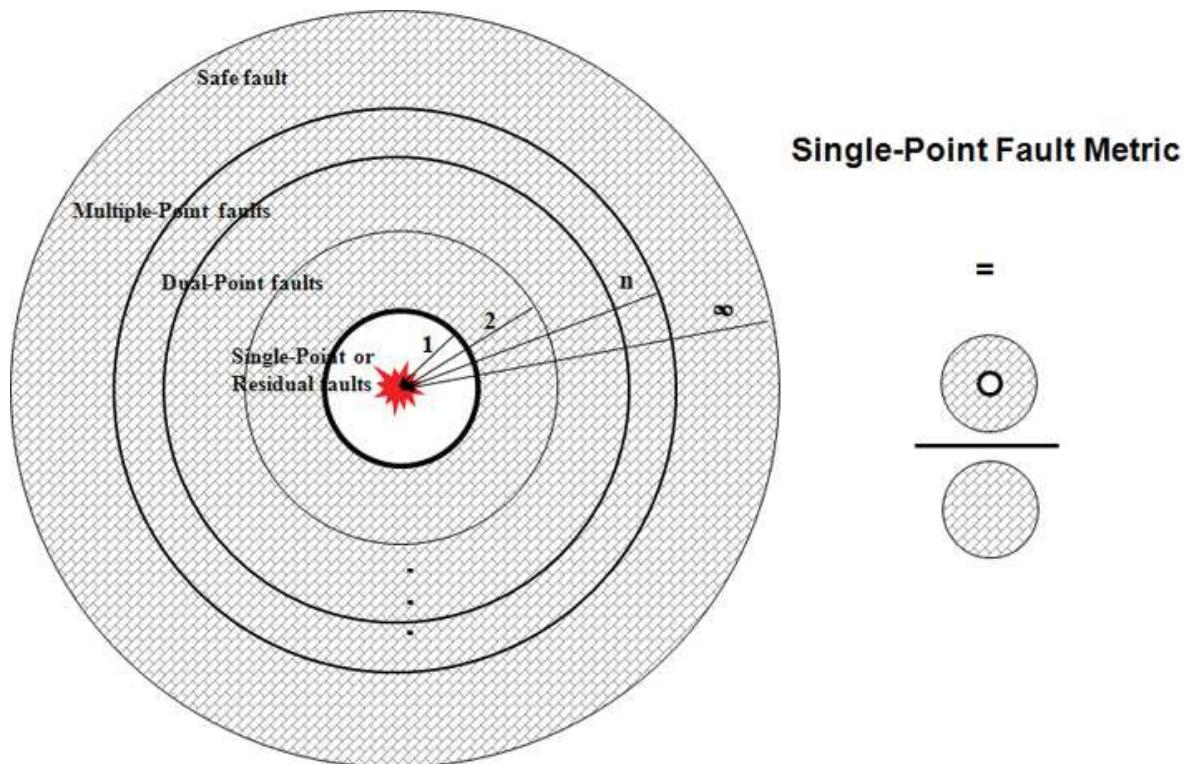


Figure C.2 — Graphical representation of the single-point fault metric

C.3 Latent-fault metric

C.3.1 This metric reflects the robustness of the item to latent faults either by coverage of faults in safety mechanisms or by the driver recognizing that the fault exists before the violation of the safety goal, or by design (primarily safe faults). A high latent-fault metric implies that the proportion of latent faults in the hardware is low.

C.3.2 This requirement applies to ASIL (B), (C), and D of the safety goal. The calculation in Equation (C.6) shall be used to determine the latent-fault metric:

$$1 - \frac{\sum_{SR,HW} (\lambda_{MPF,latent})}{\sum_{SR,HW} (\lambda - \lambda_{SPF} - \lambda_{RF})} = \frac{\sum_{SR,HW} (\lambda_{MPF,perceived\ or\ detected} + \lambda_S)}{\sum_{SR,HW} (\lambda - \lambda_{SPF} - \lambda_{RF})} \quad (C.6)$$

where $\sum_{SR,HW} \lambda_x$ is the sum of λ_x of the safety-related hardware elements of the item to be considered for the metrics.

NOTE 1 Only the safety-related hardware elements of the item whose failures have the potential to contribute significantly to the violation of the safety goal are considered for this metric.

EXAMPLE Hardware elements whose faults are multiple-point faults with $n > 2$ can be omitted from the calculations unless shown to be relevant in the technical safety concept.

NOTE 2 Figure C.3 gives a graphical representation of the latent-fault metric.

NOTE 3 An example of calculation of “latent-fault metric” is given in Annex E.

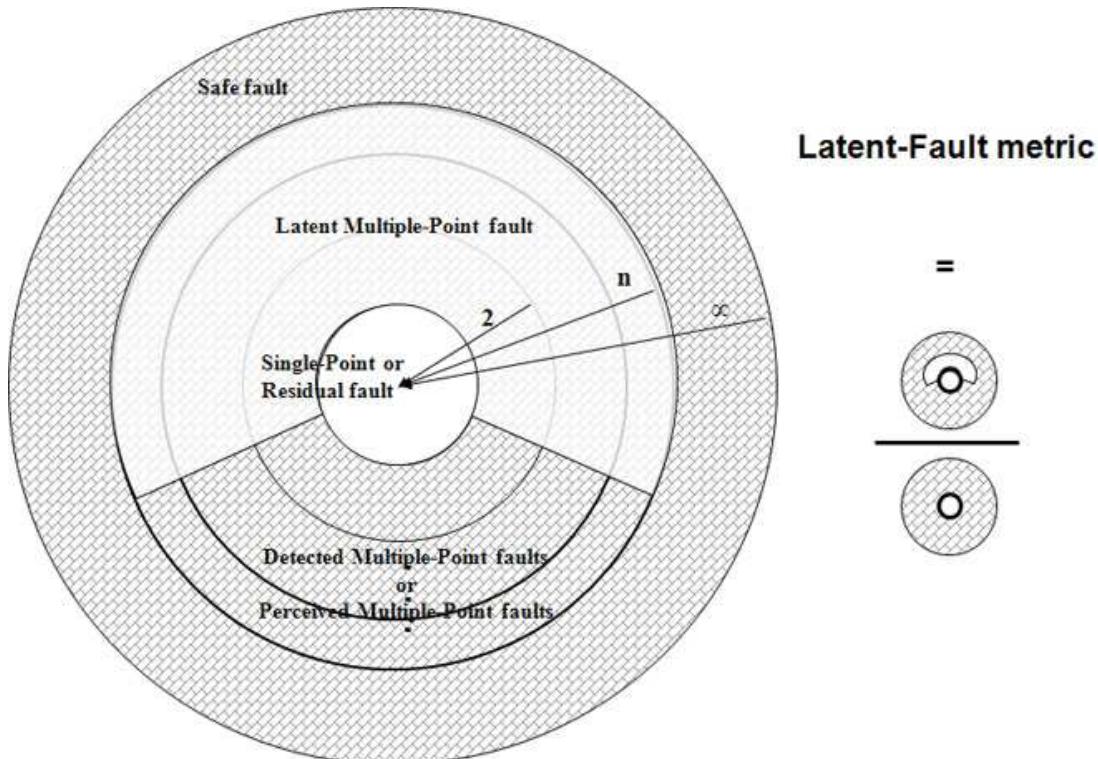


Figure C.3 — Graphical representation of the latent-fault metric

Annex D (informative)

Evaluation of the diagnostic coverage

D.1 General

This annex is intended to be used as:

- a) an evaluation of the diagnostic coverage to produce a rationale for:
 - 1) the compliance with the single-point fault and latent-fault metrics defined in Clause 8;
 - 2) the compliance with the evaluation of the safety goal violations due to random hardware failures as defined in Clause 9;
- b) a guideline in order to choose appropriate safety mechanisms to be implemented in the E/E architecture to detect failures of elements.

Figure D.1 shows the generic hardware of an embedded system. Typical faults or failures of the hardware elements of this system are shown in Table D.1 which also includes guidelines for diagnostic coverage. Each element listed in the leftmost column is associated with one or more faults which are captured in the columns to the right of the element. The listing does not claim exhaustiveness and can be adjusted based on additional known faults or depending on the application.

Additional detail on the safety mechanisms associated with these element faults are referenced in each row (Tables D.2 to D.14). The effectiveness of these typical safety mechanisms for the given elements is categorized according to their ability to cover the listed faults to achieve low, medium or high diagnostic coverage of the element. These low, medium and high diagnostic coverage rankings correspond to typical coverage levels at 60 %, 90 % or 99 %, respectively.

The assignment of the faults and their corresponding safety mechanisms to diagnostic coverage levels can vary from that listed in Table D.1 depending on:

- c) variations in the source of the fault type detected by the diagnostic
- d) the effectiveness of the safety mechanism
- e) the specific implementation of the safety mechanism
- f) the execution timing of the safety mechanism (periodicity)
- g) the hardware technologies implemented in the system
- h) the probability of the failure modes, based on hardware in the system
- i) a more detailed analysis of the faults and their classification into several subclasses with different diagnostic coverage levels.

In summary, Table D.1 provides guidelines which are adapted based on analysis of the system elements.

These guidelines do not address specific constraints that can be specified in the safety concepts in order to avoid the violation of the safety goals. These constraints, such as timing aspects (periodicity of diagnostic) for example, are not considered when evaluating the generic typical diagnostic coverage by the safety

mechanism. They will be considered when evaluating the specific diagnostic coverage by a safety mechanism used in the item to avoid the violation of the safety goals.

EXAMPLE A safety mechanism can have a high generic typical diagnostic coverage in this annex but if the diagnostic test interval used is longer than the diagnostic test interval needed to comply with the relevant fault tolerant time interval, the specific diagnostic coverage with respect to the avoidance of violation of the safety goal, will be much lower.

Therefore Tables D.1 to D.14 can be used as a starting point to evaluate the diagnostic coverage of these safety mechanisms with the claimed DC supported by a proper rationale. In addition, the given information is intended to help define the faults or failure modes of the element; however the relevant failure modes are ultimately dependent on the application in which the elements are used.

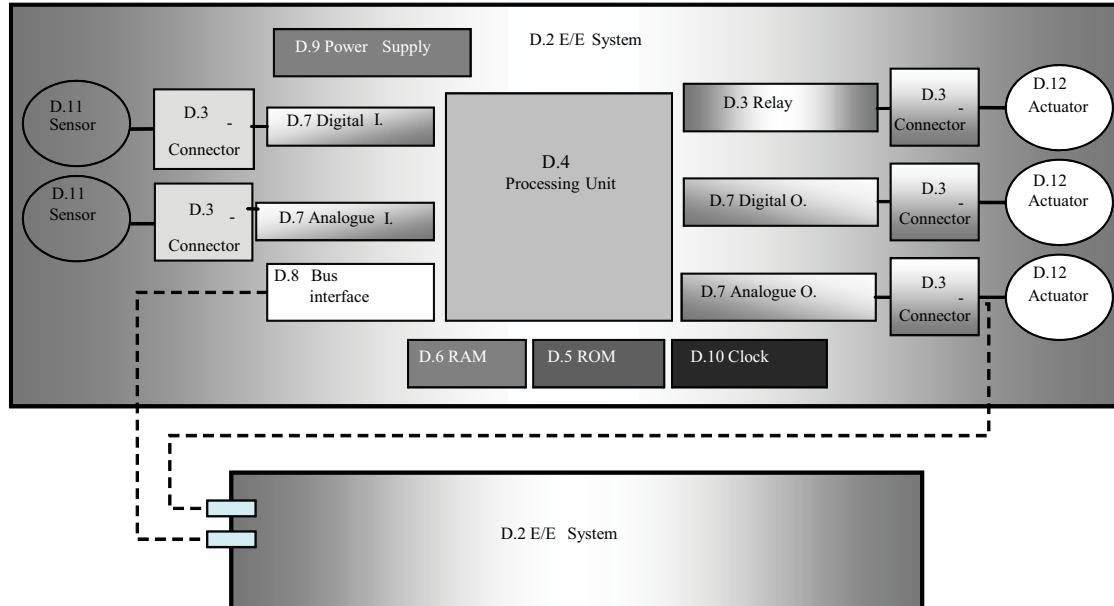


Figure D.1 — Generic hardware of a system

Tables D.2 to D.14 support the information of Table D.1 by giving guidelines on techniques for diagnostic tests. Tables D.1 to D.14 are not exhaustive and other techniques can be used, provided evidence is available to support the claimed diagnostic coverage. If justified, higher diagnostic coverage can be estimated, up to 100 % for simple or complex elements.

Table D.1 — Analyzed faults or failures modes in the derivation of diagnostic coverage

Element	See Tables	Analyzed failure modes for 60 %/90 %/99 % DC		
		Low (60 %)	Medium (90 %)	High (99 %)
General elements				
E.E Systems	D.2	No generic fault model available. Detailed analysis necessary.	No generic fault model available. Detailed analysis necessary.	No generic fault model available. Detailed analysis necessary.
Electrical elements				
Relays	D.3	Does not energize or de-energize. Welded contacts	Does not energize or de-energize. Individual contacts welded	Does not energize or de-energize. Individual contacts welded
Harnesses including splice and connectors		Open Circuit Short Circuit to Ground Short Circuit to Vbat Short Circuit between neighbouring pins	Open Circuit Short Circuit to Ground (d.c Coupled) Short Circuit to Vbat Short Circuit between neighbouring pins	Open Circuit Contact Resistance Short Circuit to Ground (d.c Coupled) Short Circuit to Vbat Short Circuit between neighbouring pins Resistive drift between pins
Sensors including signal switches	D.11	No generic fault model available. Detailed analysis necessary. Typical failure modes to be covered include Out-of-range Stuck in range	No generic fault model available. Detailed analysis necessary. Typical failure modes to be covered include Out-of-range Offsets Stuck in range	No generic fault model available. Detailed analysis necessary. Typical failure modes to be covered include Out-of-range Offsets Stuck in range Oscillations
Final elements (actuators, lamps, buzzer, screen...)	D.12	No generic fault model available. Detailed analysis necessary.	No generic fault model available. Detailed analysis necessary.	No generic fault model available. Detailed analysis necessary.
General semiconductor elements				
Power supply	D.9	Under and over Voltage	Drift Under and over Voltage	Drift and oscillation Under and over Voltage Power spikes
Clock	D.10	Stuck-at ^a	d.c. fault model ^b	d.c. fault model ^b Incorrect frequency Period jitter
Non-volatile memory	D.5	Stuck-at ^a for data and addresses and control interface, lines and logic	d.c. fault model ^b for data and addresses (includes address lines within same block) and control interface, lines and logic	d.c. fault model ^b for data, addresses (includes address lines within same block) and control interface, lines and logic

Table D.1 (continued)

Element	See Tables	Analyzed failure modes for 60 %/90 %/99 % DC		
		Low (60 %)	Medium (90 %)	High (99 %)
Volatile memory	D.6	Stuck-at ^a for data, addresses and control interface, lines and logic	d.c. fault model ^b for data, addresses (includes address lines within same block and inability to write to cell) and control interface, lines and logic Soft error model ^c for bit cells	d.c. fault model ^b for data, addresses (includes address lines within same block and inability to write to cell) and control interface, lines and logic Soft error model ^c for bit cells
Digital I/O	D.7	Stuck-at ^a (including signal lines outside of the microcontroller)	d.c. fault model ^b (including signal lines outside of the microcontroller)	d.c. fault model ^b (including signal lines outside of the microcontroller) Drift and oscillation
Analogue I/O		Stuck-at ^a (including signal lines outside of the microcontroller)	d.c. fault model ^b (including signal lines outside of the microcontroller) Drift and oscillation	d.c. fault model ^b (including signal lines outside of the microcontroller) Drift and oscillation
Specific semiconductor elements				
Processing units	ALU - Data Path	D.4/D.13	Stuck-at ^a	Stuck-at ^a at gate level d.c. fault model ^b Soft error model ^c (for sequential parts)
	Registers (general purpose registers bank, DMA transfer registers...), internal RAM	D.4	Stuck-at ^a	Stuck-at ^a at gate level Soft error model ^c d.c. fault model ^b including no, wrong or multiple addressing of registers Soft error model ^c
	Address calculation (Load/Store Unit, DMA addressing logic, memory and bus interfaces)	D.4/D.5/D.6	Stuck-at ^a	Stuck-at ^a at gate level Soft error model ^c (for sequential parts) d.c. fault model ^b including no, wrong or multiple addressing Soft error model ^c (for sequential parts)
	Interrupt handling	D.4/D.10	Omission of or continuous interrupts	Omission of or continuous interrupts Incorrect interrupt executed Wrong priority Slow or interfered interrupt handling causing missed or delayed interrupts service
	Control logic (Sequencer, coding and execution logic including flag registers and stack control)	D.4/D.10	No code execution Execution too slow Stack overflow/underflow	Wrong coding or no execution Execution out of order Execution too fast or too slow Stack overflow/underflow
	Configuration Registers	D.4	—	Corruption of registers (soft errors) Stuck-at ^a fault model
	Other sub-elements not belonging to previous classes	D.4/D.13	Stuck-at ^a	Stuck-at ^a at gate level d.c. fault model ^b Soft error model ^c (for sequential part)

Table D.1 (continued)

Element		See Tables	Analyzed failure modes for 60 %/90 %/99 % DC		
			Low (60 %)	Medium (90 %)	High (99 %)
Communication	On-chip communication including bus-arbitration	D.14	Stuck-at ^a (data, control, address and arbitration signals) Time out No or continuous arbitration	d.c. fault model ^b (data, control, address and arbitration signals) Time out No or continuous arbitration	d.c. fault model ^b (data, control, address and arbitration signals) Time out No or continuous or wrong arbitration Soft errors (for sequential part)
	Data transmission (to be analysed with ISO 26262-6:2011, Annex D)	D.8	Failure of communication peer Message corruption Message delay Message loss Unintended message repetition	Previous + Resequencing Insertion of message	Previous + Masquerading

NOTE 1 Higher DC can be claimed based on analysis. Likewise, lower coverage would result if the dominant failure mode is not listed.

NOTE 2 Transient faults are considered when shown to be relevant due, for instance, to the technology used.

NOTE 3 Failure modes for Processing Units can be adjusted to recognize a.c. fault models, such as transition faults (slow to rise and slow to fall nodes at application frequency) and path delays. Faults of this type are expected to be more prevalent with smaller process geometries. Usually tests for these types of faults are done at start-up, or power-down, or both, due to their intrusive nature and their ability to detect failures early with margin tests. Since they are hard to quantify, these failure modes are generally not included in failure rate calculations.

NOTE 4 If properly exercised, methods derived from stuck-at simulations (e.g. N-detect testing), but executed at application conditions, are known to be effective for d.c. fault and transition models as well.

^a “Stuck-at”: is a fault category that can be described with continuous “0” or “1” or “on” at the pins of an element. It is valid only for elements which have element level pin interfaces.

^b “d.c. fault model” (“direct current fault model”) includes the following failure modes: stuck-at faults, stuck-open, open or high impedance outputs, as well as short circuits between signal lines. It is not intended here to require an exhaustive analysis, for example to require the exhaustive analysis of bridging faults that can affect any theoretical combination of any signal inside a microcontroller or in a complex PCB. The analysis focuses on main signals or on very highly coupled interconnections identified with a layout level analysis.

^c “soft error model”: soft errors (e.g. bit flips) are the results of transient faults caused by alpha particles from package decay, neutrons, etc. These transient faults are also referred as Single Event Upset (SEU) and Single Event Transient (SET).

Table D.2 — Systems

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Failure detection by on-line monitoring	D.2.1.1	Low	Depends on diagnostic coverage of failure detection
Comparator	D.2.1.2	High	Depends on the quality of the comparison
Majority voter	D.2.1.3	High	Depends on the quality of the voting
Dynamic principles	D.2.2.1	Medium	Depends on diagnostic coverage of failure detection
Analogue signal monitoring in preference to digital on/off states	D.2.2.2	Low	—
Self-test by software cross exchange between two independent units)	D.2.3.3	Medium	Depends on the quality of the self test

Table D.3 — Electrical elements

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Failure detection by on-line monitoring	D.2.1.1	High	Depends on diagnostic coverage of failure detection

NOTE This table deals only with safety mechanisms dedicated to electrical elements. General techniques like a technique based on a data comparison (see D.2.1.2) are also able to detect failures of electrical elements but are not integrated in this table (already included in Table D.2).

Table D.4 — Processing units

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Self-test by software: limited number of patterns (one channel)	D.2.3.1	Medium	Depends on the quality of the self test
Self-test by software cross exchange between two independent units	D.2.3.3	Medium	Depends of the quality of the self test
Self-test supported by hardware (one-channel)	D.2.3.2	Medium	Depends on the quality of the self test
Software diversified redundancy (one hardware channel)	D.2.3.4	High	Depends on the quality of the diversification. Common mode failures can reduce diagnostic coverage
Reciprocal comparison by software	D.2.3.5	High	Depends on the quality of the comparison
HW redundancy (e.g. Dual Core Lockstep, asymmetric redundancy, coded processing)	D.2.3.6	High	It depends on the quality of redundancy. Common mode failures can reduce diagnostic coverage
Configuration Register Test	D.2.3.7	High	Configuration registers only
Stack over/under flow Detection	D.2.3.8	Low	Stack boundary test only
Integrated Hardware consistency monitoring	D.2.3.9	High	Coverage for illegal hardware exceptions only

NOTE This table deals only with safety mechanisms dedicated to processing units. General techniques like one based on data comparison (see D.2.1.2) are also able to detect failures of electrical elements but are not integrated in this table (already included in Table D.2).

Table D.5 — Non-volatile memory

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Parity bit	D.2.5.2	Low	—
Memory monitoring using error-detection-correction codes (EDC)	D.2.4.1	High	The effectiveness depends on the number of redundant bits. Can be used to correct errors
Modified checksum	D.2.4.2	Low	Depends on the number and location of bit errors within test area
Memory Signature	D.2.4.3	High	—
Block replication	D.2.4.4	High	—

Table D.6 — Volatile memory

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
RAM pattern test	D.2.5.1	Medium	High coverage for stuck-at failures. No coverage for linked failures. Can be appropriate to run under interrupt protection
RAM March test	D.2.5.3	High	Depends on the write read order for linked cell coverage. Test generally not appropriate for run time
Parity bit	D.2.5.2	Low	—
Memory monitoring using error-detection-correction codes (EDC)	D.2.4.1	High	The effectiveness depends on the number of redundant bits. Can be used to correct errors
Block replication	D.2.4.4	High	Common failure modes can reduce diagnostic coverage
Running checksum/CRC	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

Table D.7 — Analogue and digital I/O

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Failure detection by on-line monitoring (Digital I/O) ^a	D.2.1.1	Low	Depends on diagnostic coverage of failure detection
Test pattern	D.2.6.1	High	Depends on type of pattern
Code protection for digital I/O	D.2.6.2	Medium	Depends on type of coding
Multi-channel parallel output	D.2.6.3	High	—
Monitored outputs	D.2.6.4	High	Only if dataflow changes within diagnostic test interval
Input comparison/voting (1oo2, 2oo3 or better redundancy)	D.2.6.5	High	Only if dataflow changes within diagnostic test interval

^a Digital I/O can be periodic.

Table D.8 — Communication bus (serial, parallel)

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
One-bit hardware redundancy	D.2.7.1	Low	—
Multi-bit hardware redundancy	D.2.7.2	Medium	—
Read back of sent message	D.2.7.9	Medium	—
Complete hardware redundancy	D.2.7.3	High	Common mode failures can reduce diagnostic coverage
Inspection using test patterns	D.2.7.4	High	—
Transmission redundancy	D.2.7.5	Medium	Depends on type of redundancy. Effective only against transient faults
Information redundancy	D.2.7.6	Medium	Depends on type of redundancy
Frame counter	D.2.7.7	Medium	—
Timeout monitoring	D.2.7.8	Medium	—
Combination of information redundancy, frame counter and timeout monitoring	D.2.7.6, D.2.7.7 and D.2.7.8	High	For systems without hardware redundancy or test patterns, high coverage can be claimed for the combination of these safety mechanisms

Table D.9 — Power supply

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Voltage or current control (input)	D.2.8.1	Low	—
Voltage or current control (output)	D.2.8.2	High	—

Table D.10 — Program sequence monitoring/Clock

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Watchdog with separate time base without time-window	D.2.9.1	Low	—
Watchdog with separate time base and time-window	D.2.9.2	Medium	Depends on time restriction for the time-window
Logical monitoring of program sequence	D.2.9.3	Medium	Only effective against clock failures if external temporal events influence the logical program flow. Provides coverage for internal hardware failures (such as interrupt frequency errors) that can cause the software to run out of sequence
Combination of temporal and logical monitoring of program sequence	D.2.9.4	High	—
Combination of temporal and logical monitoring of program sequences with time dependency	D.2.9.5	High	<p>Provides coverage for internal hardware failures that can cause the software to run out of sequence.</p> <p>When implemented with asymmetrical designs, provides coverage regarding communication sequence between main and monitoring device</p> <p>NOTE Method to be designed to account for execution jitter from interrupts, CPU loading, etc.</p>

Table D.11 — Sensors

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Failure detection by on-line monitoring	D.2.1.1	Low	Depends on diagnostic coverage of failure detection
Test pattern	D.2.6.1	High	—
Input comparison/voting (1oo2, 2oo3 or better redundancy)	D.2.6.5	High	Only if dataflow changes within diagnostic test interval
Sensor valid range	D.2.10.1	Low	Detects shorts to ground or power and some open circuits
Sensor correlation	D.2.10.2	High	Detects in range failures
Sensor rationality check	D.2.10.3	Medium	—

Table D.12 — Actuators

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
Failure detection by on-line monitoring	D.2.1.1	Low	Depends on diagnostic coverage of failure detection
Test pattern	D.2.6.1	High	—
Monitoring (i.e. coherence control)	D.2.11.1	High	Depends on diagnostic coverage of failure detection

Table D.13 — Combinatorial and sequential logic

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable (d.c. fault model)	Notes
Self-test by software	D.2.3.1	Medium	—
Self-test supported by hardware (one-channel)	D.2.3.2	High	Effectiveness depends on the type of self-test. Gate level is an appropriate level for this test

Table D.14 — On-chip communication

Safety mechanism/measure	See overview of techniques	Typical diagnostic coverage considered achievable	Notes
One-bit hardware redundancy	D.2.7.1	Low	—
Multi-bit hardware redundancy	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	D.2.7.3	High	Common failure modes can reduce diagnostic coverage
Test pattern	D.2.6.1	High	Depends on type of pattern

NOTE This table provides coverage for communication buses within microprocessors.

D.2 Overview of techniques for embedded diagnostic self-tests

D.2.1 Electrical

Global objective: To control failures in electromechanical elements.

D.2.1.1 Failure detection by on-line monitoring

NOTE 1 This technique/measure is referenced in Tables D.2, D.3, D.7, D.11 and D.12.

Aim: To detect failures by monitoring the behaviour of the system in response to the normal (on-line) operation.

Description: Under certain conditions, failures can be detected using information about (for example) the time behaviour of the system. For example, if a switch is normally actuated and does not change state at the expected time, a failure will have been detected. It is not usually possible to localize the failure.

NOTE 2 In general, there is no specific hardware element for the realisation of the on-line monitoring diagram. On-line monitoring detects abnormal behaviour of the system with respect to certain conditions of activation. For example, if such parameter is inverted when the vehicle speed is different from zero, then detection of incoherence between this parameter and vehicle speed leads to failure detection.

D.2.1.2 Comparator

NOTE This technique/measure is referenced in Table D.2.

Aim: To detect, as early as possible, (non-simultaneous) failures in independent hardware or software.

Description: The output signals of independent hardware or output information of independent software, are compared cyclically or continuously by a comparator. Detected differences lead to a failure message. For instance: two processing units exchange data (including results, intermediate results and test data) reciprocally. A comparison of the data is carried out using software in each unit and detected differences lead to a failure message.

D.2.1.3 Majority voter

NOTE 1 This technique/measure is referenced in Table D.2.

Aim: To detect and mask failures in one of at least three channels.

Description: A voting unit using the majority principle (2 out of 3, 3 out of 3, or m out of n) is used to detect and mask failures.

NOTE 2 Unlike the comparator, the majority voter technique increases the availability by ensuring the functionality of the redundant channel even after the loss of one channel.

D.2.2 Electronic

Global objective: To control failure in solid-state elements.

D.2.2.1 Dynamic principles

NOTE This technique/measure is referenced in Table D.2.

Aim: To detect static failures by dynamic signal processing.

Description: A forced change of otherwise static signals (internally or externally generated) helps to detect static failures in elements. This technique is often associated with electromechanical elements.

D.2.2.2 Analogue signal monitoring in preference to digital on/off states

NOTE This technique/measure is referenced in Table D.2.

Aim: To improve confidence in measured signals.

Description: Wherever there is a choice, analogue signals are used in preference to digital on/off states. For example, trip or safe states are represented by analogue signal levels, usually with signal level tolerance monitoring. In the case of a digital signal, it is possible to monitor it with an analogue input. The technique gives continuity monitoring and a higher level of confidence in the transmitter, reducing the required frequency of the periodic test performed to detect failures of the transmitter sensing function.

D.2.3 Processing units

Global objective: To detect failures in processing units which lead to incorrect results.

D.2.3.1 Self-test by software

NOTE This technique/measure is referenced in Tables D.4 and D.13.

Aim: To detect, as early as possible, failures in the processing unit and other sub-elements consisting of physical storage (for example, registers) or functional units (for example, instruction decoder or an EDC coder/decoder), or both, by means of software.

Description: The failure detection is realised entirely by software which perform self-tests using a data pattern, or set of data patterns, to test the physical storage (for example, data and address registers) or the functional units (for example, the instruction decoder) or both.

EXAMPLE 1 The processing unit is tested for functional correctness by applying at least one pattern per instruction. Instructions not executed in the safety-related path can be omitted from the test but coverage can be limited as not all gates of the processing unit will be tested. In general it is possible that not all dedicated and special purpose registers, core timers, and exceptions can be covered. Coverage for order dependencies, such as pipelines, or timing related fault modes can be limited. Determining the actual coverage of the tested gates (in contrast to covered instructions) typically requires extensive fault simulation. This test provides very limited or no coverage for soft errors.

EXAMPLE 2 In the case of sub-elements like an EDC coder/decoder, the software can read pre-written intentionally corrupted words to test the behaviour of the EDC logic. Corrupted words can also be written by the software test itself if the EDC and memory interface have an HW switch to access both data and code bits. Coverage depends on the amount and richness of patterns. This test provides no coverage for soft errors.

D.2.3.2 Self-test supported by hardware (one-channel)

NOTE This technique/measure is referenced in Tables D.4 and D.13.

Aim: To detect, as early as possible, failures in the processing unit and other sub-elements, using special hardware that increases the speed and extends the scope of failure detection.

Description: Additional special hardware facilities support self-test functions to detect failures in the processing unit and other sub-elements (for example an EDC coder/decoder) at a gate level. The test can achieve high coverage. Typically only run at the initialization or power-down of the processing unit due to its intrusive nature. Typical usage is for multipoint fault detection.

EXAMPLE In the case of sub-elements like an EDC coder/decoder, a special HW mechanism, like a logic BIST, can be added to generate inputs to the coder-decoder and check for expected results. Typically inputs are generated by random pattern generators (e.g. MISR). Its coverage depends on the amount and richness of patterns – but usually the coverage is quite high due to the automatic pattern generation. This test provides no coverage for soft errors.

D.2.3.3 Self-test by software cross exchanged between two independent units

NOTE This technique/measure is referenced in Tables D.2 and D.4.

Aim: To detect, as early as possible, failures in the processing unit consisting of physical storage (for example registers) and functional units (for example, instruction decoder).

Description: The failure detection is realised entirely by means of two or more processing units each executing additional software functions which perform self-tests (for example walking-bit pattern) to test the physical storage (data and address registers) and the functional units (for example instruction decoder). The processing units exchange the results. This test provides very limited or no coverage for soft errors.

D.2.3.4 Software diversified redundancy (one hardware channel)

NOTE 1 This technique/measure is referenced in Table D.4.

Aim: To detect, as early as possible, failures in the processing unit, by dynamic software comparison.

Description: The design consists of two redundant diverse software implementations in one hardware channel. In some cases, using different hardware resources (e.g. different RAM, ROM memory ranges) can increase the diagnostic coverage.

One implementation, referred to as the primary path, is responsible for the calculations that if calculated erroneously can cause a hazard. The second implementation, referred to as the redundant path, is responsible for verifying the primary path's calculations and taking action if a failure is detected. Often the redundant path is implemented using separate algorithm designs and code to provide for software diversity. Once both paths are complete, a comparison of the output data of the two redundant software implementations is carried out. Detected differences lead to a failure message (see Figure D.2). The design includes methods to coordinate the two paths and to resynchronise the paths for transient errors.

Generally the comparison involves some type of hysteresis and filtering to allow for minor differences due to the diverse software paths. Examples of algorithm diversity are: A+B=C versus C-B=A and one path using normal calculations and the other path using two's complement mathematics. A redundant path can be as simple as a magnitude or rate-limit check on the calculation of the primary path.

NOTE 2 Due to the potential common cause failures between the primary and redundant paths, an additional watchdog processor can be used to verify the operation of the primary controller via a question and response diagnostic (see Reference [21]).

Another version of this safety mechanism is to implement the redundant path as an exact copy of the primary path (or to execute the primary path twice). This version, without software redundancy, only provides coverage for soft errors. Medium coverage can be achieved if the code is executed a third time with known inputs generating outputs to be verified versus a set of expected outputs. This technique results in a very easy pass-fail criterion (compared results are expected to agree exactly) and easy implementation (the redundant path does not need to be designed) but the concept includes the preservation of history terms (e.g. dynamic states, integrators, rate-limits, etc.).

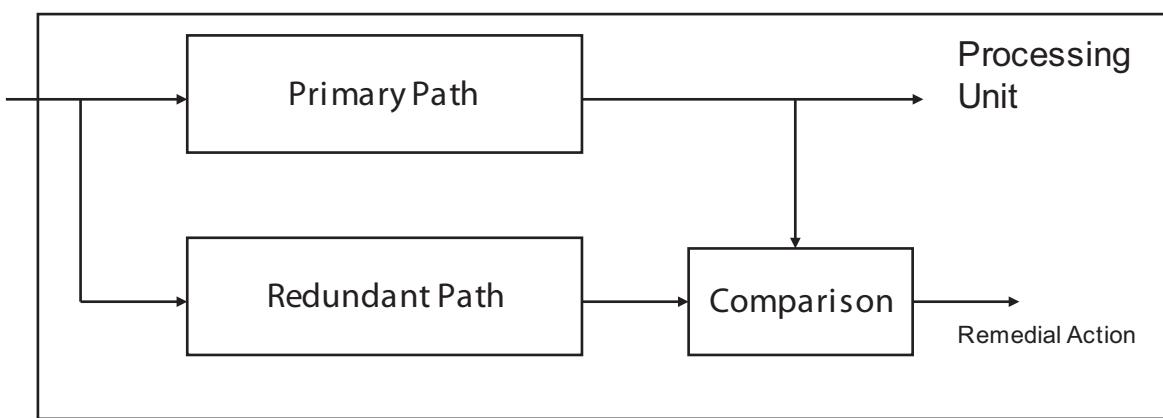


Figure D.2 — Redundant software comparison same processing unit

D.2.3.5 Reciprocal comparison by software in separate processing units

NOTE This technique/measure is referenced in Table D.4.

Aim: To detect, as early as possible, failures in the processing unit, by dynamic software comparison.

Description: Two processing units exchange data (including results, intermediate results and test data) reciprocally. A comparison of the data is carried out using software in each unit and detected differences lead to a failure message (see Figure D.3). This approach allows for hardware and software diversity if different processor types are used as well as separate algorithm designs, code and compilers. The design includes methods to avoid false error detections due to differences between the processors (e.g. loop jitter, communication delays, processor initialization).

Paths can be implemented using separate cores of a dual core processor. In this case, the method includes analysis to understand common cause failures modes, due to the shared die and package, of the two cores.

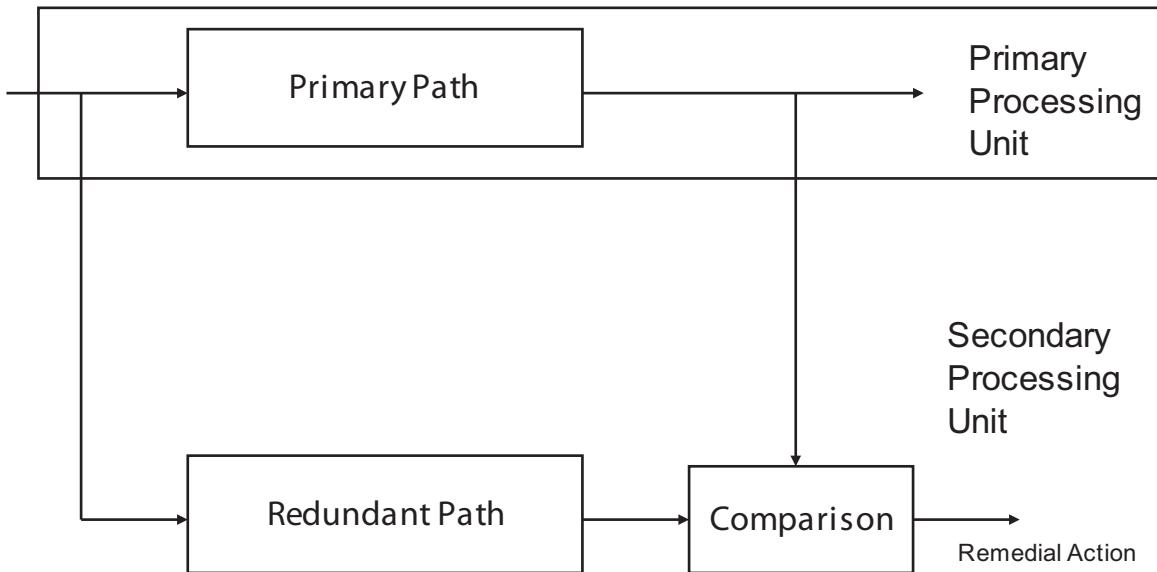


Figure D.3 — Redundant software comparison different processing units

D.2.3.6 HW redundancy (e.g. Dual Core Lockstep, asymmetric redundancy, coded processing)

NOTE This technique/measure is referenced in Table D.4.

Aim: To detect, as early as possible, failures in the processing unit, by step-by-step comparison of internal or external results or both produced by two processing units operating in lockstep.

Description: In one version of this type of diagnostic technique, the Dual Core Lockstep, two symmetrical processing units are contained on one die (see Reference [22]). The processing units run duplicate operations in lockstep (or delayed by a fixed period) and the results are compared. Any mismatch results in an error condition and usually a reset condition. This is very effective for transient errors and for ALU type failures. Depending on the level of redundancy, coverage can be extended to the memory addressing lines and configuration registers. The technique has the advantage that no separate code for the parallel path is required but has the disadvantage of having two processing units providing only the performance of a single processing unit. In good designs, common cause failures are understood and addressed (for example, common clock failure). This approach by itself does not provide diagnostic coverage for systematic errors.

Other types of HW redundancies are possible, such as asymmetric redundancy. In those architectures (e.g. Reference [25]), a diverse and dedicated processing unit is tightly coupled with the main processing units by means of an interface enabling a step-by-step comparison of internal and external results. This is very effective for both a d.c. fault model and for soft errors: moreover, the interface reduces complexity and shortens error detection latency, for example, for faults affecting the processing unit registers bank. No separate code is needed for the parallel path and the dedicated processing unit can be smaller than the main one. The hardware diversity provides effective coverage for common cause failures and systematic failures. The disadvantage of this approach is that it can require a detailed analysis to prove the diagnostic coverage.

Coded processing is also possible: processing units can be designed with special failure-recognising or failure correcting circuit techniques. These approaches can guarantee high coverage for very small processors with limited functionalities or they can be suitable for processor sub-units like ALU (e.g. Reference [26]). Hardware and software coding can be combined using approaches like the Vital Coded Processor (see Reference [27]). A detailed analysis can be needed to prove the diagnostic coverage.

D.2.3.7 Configuration register test

NOTE This technique/measure is referenced in Table D.4.

Aim: To detect, as early as possible, failures in the configuration registers of a processing unit. Failures can be hardware related (stuck values or soft errors induced bit flips) or software related (incorrect value stored or register corrupted by software error).

Description: Configuration register settings are read and then compared to an encoding of the expected settings (e.g. a mask). If the settings do not match, the registers are reloaded with their intended value. If the error persists for a pre-determined number of checks the fault condition is reported.

D.2.3.8 Stack over/under flow detection

NOTE This technique/measure is referenced in Table D.4.

Aim: To detect, as early as possible, stack over or under flows.

Description: The boundaries of the stack in volatile memory are loaded with predefined values. Periodically, the values are checked and if they have changed, an over or under flow is detected. A test is not needed if writes outside stack boundaries are controlled by a memory management unit.

D.2.3.9 Integrated hardware consistency monitoring

NOTE This technique/measure is referenced in Table D.4.

Aim: To detect, as early as possible, illegal conditions in the processing unit.

Description: Most processors are equipped with mechanisms that trigger hardware exceptions when errors are detected (division by zero and invalid op-codes, for example). Interrupt processing of these errors can then be used to trap these conditions to isolate the system from their effects. Typically, hardware monitoring is used to detect systematic failures but can also be used to detect certain kinds of random hardware faults. The technique provides low coverage for some coding errors and is good design practice.

D.2.4 Non-volatile memory

Global objective: The detection of information corruption in the non-volatile memory.

NOTE Depending on the type of memory implemented, a single fault can affect multiple memory locations. For example, an open row select line would prevent reading an entire row of memory. This type of failure can be easier to detect if multiple locations are tested.

D.2.4.1 Memory monitoring using error-detection-correction codes (EDC)

NOTE 1 This technique/measure is referenced in Tables D.5 and D.6.

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 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 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.

NOTE 2 This technology is often referred to as ECC (Error Correcting Code).

D.2.4.2 Modified checksum

NOTE This technique/measure is referenced in Table D.5.

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 [20]). The probability of a missed detection is $1/(2^{\text{size of checksum}})$ 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.

D.2.4.3 Memory signature

NOTE 1 This technique/measure is referenced in Table D.5.

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 $1/(2^{\text{size of checksum}})$ if a random result is returned (see Reference [20]).

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

D.2.4.4 Block replication (for example double memory with hardware or software comparison)

NOTE This technique/measure is referenced in Tables D.5 and D.6.

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.

D.2.5 Volatile memory

Global objective: Detecting failures during addressing, writing, storing and reading.

NOTE Depending on the type of memory implemented, a single fault can affect multiple memory locations. For example an open row select line would prevent reading an entire row of memory. This type of failure can be easier to detect if multiple locations are tested.

D.2.5.1 RAM Pattern test

NOTE 1 This technique/measure is referenced in Table D.6.

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 [20], 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.

D.2.5.2 Parity bit

NOTE This technique/measure is referenced in Tables D.5 and D.6.

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.

D.2.5.3 RAM March test

NOTE 1 This technique/measure is referenced in Table D.6.

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 [20], 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.

D.2.5.4 Running checksum/CRC

NOTE This technique/measure is referenced in Table D.6.

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.

D.2.6 I/O-units and interfaces

Global objective: To detect failures in input and output units (digital, analogue) and to prevent the sending of inadmissible outputs to the process.

D.2.6.1 Test pattern

NOTE This technique/measure is referenced in Tables D.7, D.11, D.12 and D.14.

Aim: To detect static failures (stuck-at failures) and cross-talk.

Description: This is a dataflow-independent cyclical test of input and output units. It uses a defined test pattern to compare observations with the corresponding expected values. Test coverage is dependent on the degree of independence between the test pattern information, the test pattern reception, and the test pattern evaluation. In a good design, the functional behaviour of the system is not unacceptably influenced by the test pattern.

D.2.6.2 Code protection

NOTE This technique/measure is referenced in Table D.7.

Aim: To detect random hardware and systematic failures in the input/output dataflow.

Description: This procedure protects the input and output information from both systematic and random hardware failures. Code protection gives dataflow-dependent failure detection of the input and output units, based on information redundancy, or time redundancy, or both. Typically, redundant information is superimposed on input data, or output data, or both. This gives a means to monitor the correct operation of the input or output circuits. Many techniques are possible, for example a carrier frequency signal can be superimposed on the output signal of a sensor and the logic unit can then check for the presence of the carrier frequency, or redundant code bits can be added to an output channel to allow the monitoring of the validity of a signal passing between the logic unit and final actuator.

D.2.6.3 Multi-channel parallel output

NOTE This technique/measure is referenced in Table D.7.

Aim: To detect random hardware failures (stuck-at failures), failures caused by external influences, timing failures, addressing failures, drift failures and transient failures.

Description: This is a dataflow-dependent multi-channel parallel output with independent outputs for the detection of random hardware failures. Failure detection is carried out via external comparators. If a failure occurs, the system can possibly be switched off directly. This measure is only effective if the dataflow changes during the diagnostic test interval.

D.2.6.4 Monitored outputs

NOTE This technique/measure is referenced in Table D.7.

Aim: To detect individual failures, failures caused by external influences, timing failures, addressing failures, drift failures (for analogue signals) and transient failures.

Description: This is a dataflow-dependent comparison of outputs with independent inputs to ensure compliance with a defined tolerance range (time, value). A detected failure cannot always be related to the defective output. This measure is only effective if the dataflow changes during the diagnostic test interval.

D.2.6.5 Input comparison/voting

NOTE This technique/measure is referenced in Tables D.7 and D.11.

Aim: To detect individual failures, failures caused by external influences, timing failures, addressing failures, drift failures (for analogue signals) and transient failures.

Description: This is a dataflow-dependent comparison of independent inputs to ensure compliance with a defined tolerance range (time, value). There will be 1 out of 2, 2 out of 3 or better redundancy. This measure is only effective if the dataflow changes during the diagnostic test interval.

D.2.7 Communication bus

Global objective: To detect failures in the information transfer.

D.2.7.1 One-bit hardware redundancy

NOTE This technique/measure is referenced in Tables D.8 and D.14.

Aim: To detect each odd-bit failure, i.e. 50 % of all the possible bit failures in the data stream.

Description: The communication bus is extended by one line (bit) and this additional line (bit) is used to detect failures by parity checking.

EXAMPLE Parity bit as implemented in a standard UART.

D.2.7.2 Multi-bit hardware redundancy

NOTE This technique/measure is referenced in Tables D.8 and D.14.

Aim: To detect failures during the communication on a bus and in serial transmission links.

Description: The communication bus is extended by two or more lines and these additional lines are used in order to detect failures by using Hamming code techniques.

D.2.7.3 Complete hardware redundancy

NOTE This technique/measure is referenced in Tables D.8 and D.14.

Aim: To detect failures during the communication by comparing the signals on two buses.

Description: The bus is duplicated and the additional lines are used to detect failures.

EXAMPLE Dual channel FlexRay implementation: the bus is duplicated and the additional lines (bits) are used in order to detect failures.

D.2.7.4 Inspection using test patterns

NOTE This technique/measure is referenced in Table D.8.

Aim: To detect static failures (stuck-at failure) and cross-talk.

Description: This is a dataflow-independent cyclical test of data paths. It uses a defined test pattern to compare observations with the corresponding expected values.

Test coverage is dependent on the degree of independence between the test pattern information, the test pattern reception, and the test pattern evaluation. In a good design, the functional behaviour of the system is not unacceptably influenced by the test pattern.

D.2.7.5 Transmission redundancy

NOTE This technique/measure is referenced in Table D.8.

Aim: To detect transient failures in bus communication.

Description: The information is transferred several times in sequence. The technique is only effective in detecting transient failures.

D.2.7.6 Information redundancy

NOTE 1 This technique/measure is referenced in Table D.8.

Aim: To detect failures in bus communication.

Description: Data is transmitted in blocks, together with a calculated checksum or CRC (cyclic redundancy check) (see References [28] and [29]) for each block. The receiver then re-calculates the checksum of the received data and compares the result with the received checksum. For CRC coverage depends on the length of the data to be covered, the size of the CRC (number of bits) and the polynomial. The CRC can be designed to address the more probable communication failure modes of the underlying hardware (for example burst errors).

The message ID can be included in the checksum/CRC calculation to provide coverage for corruptions in this part of the message (masquerading).

a) Low diagnostic coverage: Hamming distance of 2 or less

EXAMPLE 1 CRC value for message information embedded in message; with a CRC size of 5 bits and a polynomial 0x12 results in a Hamming distance of 2 for a data length of less than 2 048 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value.

b) Medium diagnostic coverage: Hamming distance of 3 or more

EXAMPLE 2 CRC value for message information embedded in message; with a CRC size of 8 bits and a polynomial 0x97 results in a Hamming distance of 4 for a data length of less than 119 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (typically used in a LIN bus).

EXAMPLE 3 CRC value for message information and message ID embedded in the message; with a CRC size of 10 bits and a polynomial 0x319 results in a Hamming distance of 4 for a data length of less than 501 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value.

EXAMPLE 4 CRC value for message information and message ID embedded in the message; with a CRC size of 15 bits and a polynomial 0x4599 results in a Hamming distance of 5 for a data length of less than 127 bits. As well, burst errors of length up to 15 can be detected. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in CAN).

EXAMPLE 5 CRC value for message information embedded in the message, with a CRC size of 24 bits and a polynomial 0x5D6DCB results in a Hamming distance of the CRC of 6 for a data length of less than or equal to 248 bytes and a Hamming distance of the CRC of 4 for a data length of greater than 248 bytes. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in FlexRay for the frame CRC).

EXAMPLE 6 CRC value for message header including the message ID embedded in the message; with a CRC size of 11 bits and a polynomial 0x385 results in a Hamming distance of 6 for a data length of less than or equal to 20 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in FlexRay for the header CRC).

NOTE 2 High coverage can be reached concerning data and ID corruption, however, overall high coverage cannot be reached by checking only the coherence of the data and the ID with a signature, whatever the efficiency of the signature. Specifically, a signature does not cover the message loss or the unintended message repetition.

NOTE 3 If a checksum algorithm has a Hamming distance of less than 3, a high coverage concerning data and ID corruption can still be claimed if supported by a proper rationale.

D.2.7.7 Frame counter

NOTE This technique/measure is referenced in Table D.8.

Aim: To detect frame losses. A frame is a coherent set of data sent from one controller to other controller(s). The unique frame is identified by a message ID.

Description: Each unique safety-related frame includes a counter as part of the message which is transmitted on the bus. The counter is incremented (with roll-over) during the creation of each successive frame transmitted. The receiver is then able to detect any frame loss or non-refreshment by verifying that the counter is incrementing by one.

A special version of the frame counter would be to include separate signal counters tied to the refreshment of safety-related data. In this situation, if a frame contained more than one piece of safety-related data, an individual counter for each piece of safety-related data is provided.

D.2.7.8 Timeout monitoring

NOTE This technique/measure is referenced in Table D.8.

Aim: To detect loss of data between the sending node and the receiving node.

Description: The receiver monitors each expected safety-related message ID for time between the receipt of valid frames with this message ID. A failure would be indicated by too long a period elapsing between messages. This is intended to detect continuous loss of a communications channel or the continuous loss of one a specific message (no frames received for a specific message ID).

D.2.7.9 Read back of sent message

NOTE 1 This technique/measure is referenced in Table D.8.

Aim: To detect failures in bus communication.

Description: The transmitter reads back its sent message from the bus and compares it with the original message.

NOTE 2 This safety mechanism is used by CAN.

NOTE 3 High coverage can be reached concerning data and ID corruption, however, overall high coverage cannot be reached by checking only the coherence of the data and the ID. Other failure modes like the unintended message repetition are not necessarily covered by this safety mechanism.

D.2.8 Power supply

Global objective: To detect failures caused by a defect in the power supply.

D.2.8.1 Voltage or current control (input)

NOTE This technique/measure is referenced in Table D.9.

Aim: To detect as soon as possible wrong behaviour of input current or voltage values.

Description: Monitoring of input voltage or current.

D.2.8.2 Voltage or current control (output)

NOTE This technique/measure is referenced in Table D.9.

Aim: To detect as soon as possible wrong behaviour of output current or voltage values.

Description: Monitoring of output voltage or current.

D.2.9 Temporal and logical program sequence monitoring

NOTE This group of techniques and measures is referenced in Table D.10.

Global objective: To detect a defective program sequence. A defective program sequence exists if the individual elements of a program (for example, software modules, subprograms or commands) are processed in the wrong sequence or period of time, or if the clock of the processor is faulty.

D.2.9.1 Watchdog with separate time base without time-window

NOTE This technique/measure is referenced in Table D.10.

Aim: To monitor the behaviour and the plausibility of the program sequence.

Description: External timing elements with a separate time base (for example, watchdog timers) are periodically triggered to monitor the processor's behaviour and the plausibility of the program sequence. It is important that the triggering points are correctly placed in the program. The watchdog is not triggered at a fixed period, but a maximum interval is specified.

D.2.9.2 Watchdog with separate time base and time-window

NOTE This technique/measure is referenced in Table D.10.

Aim: To monitor the behaviour and the plausibility of the program sequence.

Description: External timing elements with a separate time base (for example watchdog timers) are periodically triggered to monitor the processor's behaviour and the plausibility of the program sequence. It is important that the triggering points are correctly placed in the program (e.g. not in an interrupt service routine).

A lower and upper limit is given for the watchdog timer. If the program sequence takes a longer or shorter time than expected, action is taken.

D.2.9.3 Logical monitoring of program sequence

NOTE This technique/measure is referenced in Table D.10.

Aim: To monitor the correct sequence of the individual program sections.

Description: The correct sequence of the individual program sections is monitored using software (counting procedure, key procedure) or using external monitoring facilities (References [23, 24]). It is important that the checking points are placed in the program so that paths which can result in a hazard if they fail to complete or execute out of sequence, due to a single or multiple-point fault, are monitored. The sequences can be updated between each function call or more tightly integrated into the program execution.

D.2.9.4 Combination of temporal and logical monitoring of program sequences

NOTE This technique/measure is referenced in Table D.10.

Aim: To monitor the behaviour and the correct sequence of the individual program sections.

Description: A temporal facility (for example a watchdog timer) monitoring the program sequence is retriggered only if the sequence of the program sections is also executed correctly. This is a combination of the technique in D.2.9.3 and either D.2.9.1 or D.2.9.2.

D.2.9.5 Combination of temporal and logical monitoring of program sequences with time dependency

NOTE This technique/measure is referenced in Table D.10.

Aim: To monitor the behaviour, correct sequencing and the execution time interval of the individual program sections.

Description: A Program Flow Monitoring strategy is implemented where software update points are expected to occur within a relative time window. The PFM sequence result and time calculation are monitored by external monitoring facilities.

D.2.10 Sensors

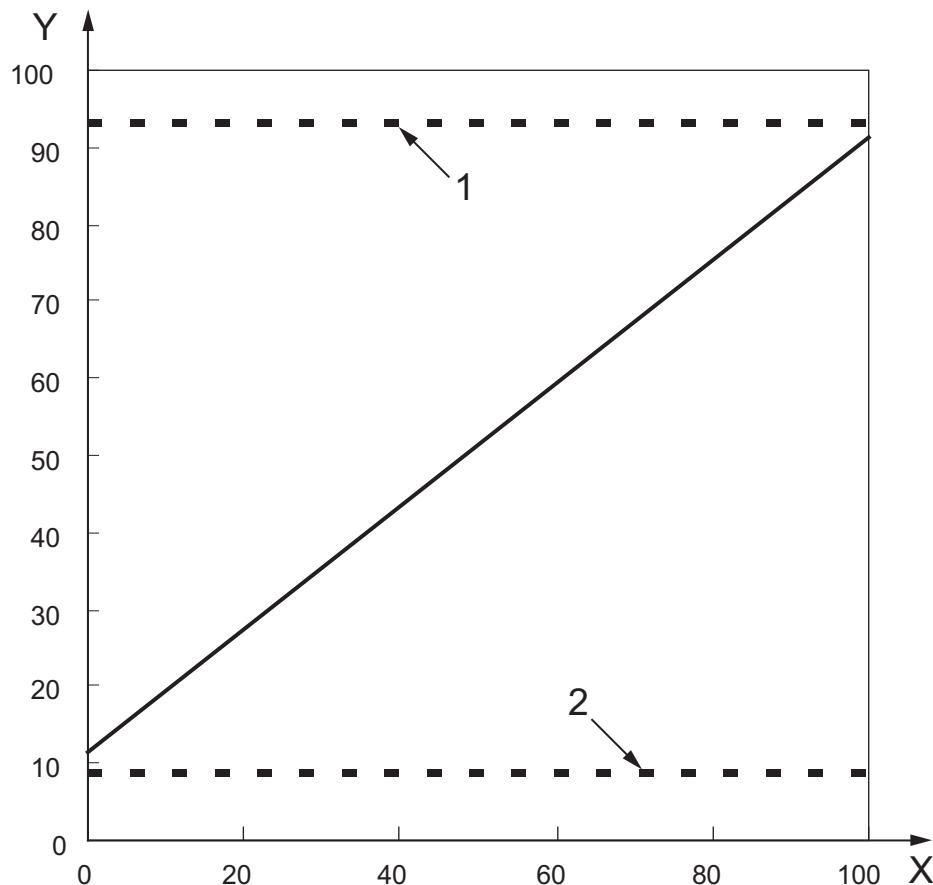
Global objective: To control failures in the sensors of the system.

D.2.10.1 Sensor valid range

NOTE This technique/measure is referenced in Table D.11.

Aim: To detect sensor shorts to ground or power and some open circuits.

Description: Limit valid reading to the middle part of the sensor electrical range (see Figure D.4 for example). If a sensor reading is in an invalid region, this indicates an electrical problem with the sensor such as a short to power or to ground. Typically used with sensors read by the ECU using ADCs.

**Key**

- X physical sensor reading, in %
- Y measured sensor reading, in % of reference voltage
- 1 out-of-range high
- 2 out-of-range low

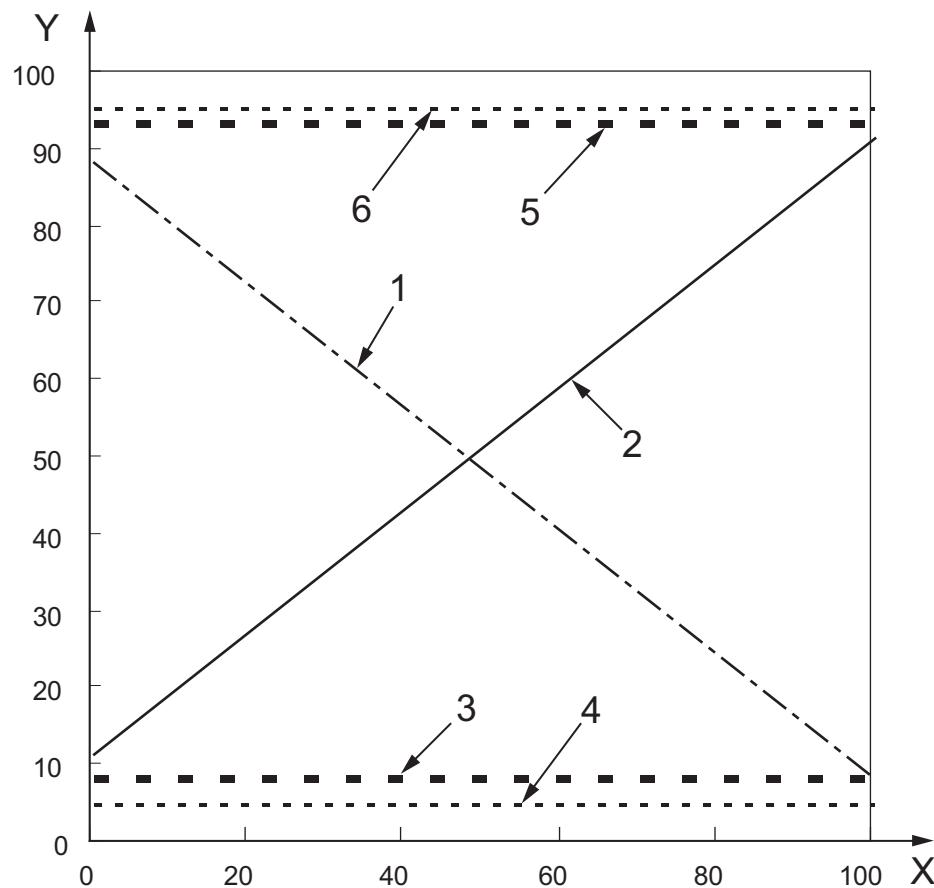
Figure D.4 — Sensor with out-of-range region**D.2.10.2 Sensor correlation**

NOTE This technique/measure is referenced in Table D.11.

Aim: To detect sensor-in-range drifts, offsets or other errors using a redundant sensor.

Description: Comparison of two identical or similar sensors to detect in-range failures such as drifts, offsets or stuck-at failures. See Figure D.5 for an example with two equal but opposite slope sensors. Note, that the out-of-range region is different for each sensor. Typically used with sensors read by the ECU using ADCs.

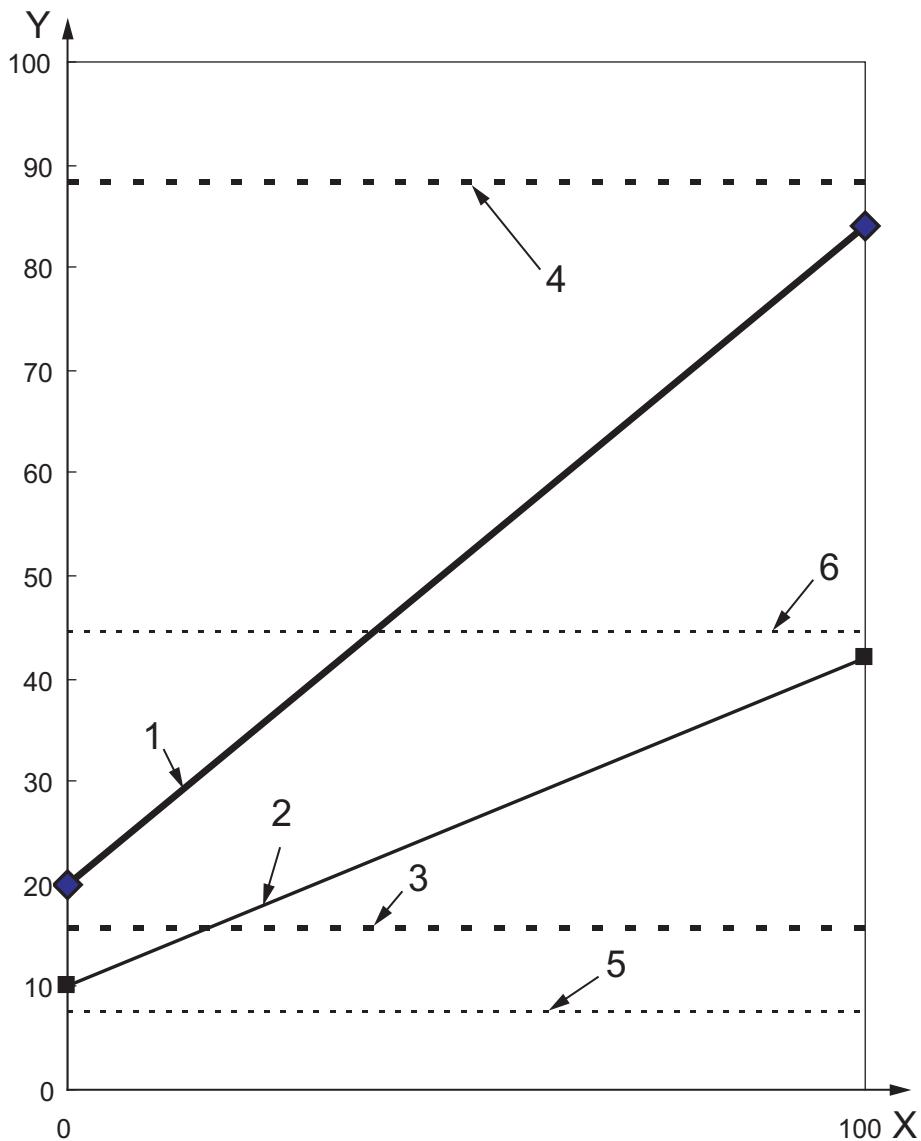
For the example of Figure D.5, sensors would be converted to equal slope and compared to agree within a threshold. The threshold is selected taking into account the ADC tolerance and the variation in the electrical elements. Both sensors are sampled by the ECU at as close to the same time as possible to avoid false failures due to the sensor readings dynamically changing.

**Key**

- X physical sensor reading, in %
- Y measured sensor reading, in % of reference voltage
- 1 sensor 1
- 2 sensor 2
- 3 out-of-range sensor 1 low
- 4 out-of-range sensor 2 low
- 5 out-of-range sensor 1 high
- 6 out-of-range sensor 2 high

Figure D.5 — Equal and opposite slope sensors with out-of-range regions

Equal slope sensor based diagnostics do not detect situations where the two sensors are shorted together yielding correlated readings at the crossing point or common cause failures where a single component, e.g. the ADC, corrupts both sensor results in a similar way. An alternative design based on one full and one half slope sensor is given in Figure D.6.

**Key**

- X physical sensor reading, in %
- Y measured sensor reading, in % of reference voltage
- 1 sensor 1
- 2 sensor 2
- 3 out-of-range sensor 1 low
- 4 out-of-range sensor 1 high
- 5 out-of-range sensor 2 low
- 6 out-of-range sensor 2 high

Figure D.6 — One full and one half slope sensor with out-of-range regions

D.2.10.3 Sensor rationality check

NOTE This technique/measure is referenced in Table D.11.

Aim: To detect sensor-in-range drifts, offsets or other errors using multiple diverse sensors.

Description: Comparison of two (or more) sensors measuring different properties to detect in-range failures such as drifts, offsets or stuck-at failures. The sensor measurements are converted to equivalent values using a model to provide values that can be compared.

EXAMPLE The comparison of gasoline engine throttle position, manifold pressure and mass air flow sensors after each is converted to an air flow reading. The usage of diverse sensors reduces the problem of systematic faults.

D.2.11 Actuators

Global objective: To control failures in the final elements of the system.

D.2.11.1 Monitoring

NOTE 1 This technique/measure is referenced in Table D.12.

Aim: To detect the incorrect operation of an actuator.

Description: The operation of the actuator is monitored.

NOTE 2 Monitoring can be done at the actuator level by physical parameter measurements (which can have high coverage) but also at the system level regarding the actuator failure effect.

EXAMPLE 1 For a cooling radiator fan, monitoring at system level uses a temperature sensor to detect failure of the cooling radiator fan. Monitoring of physical parameters measures the voltage, or the current, or both, on the inputs of the cooling radiator fan.

EXAMPLE 2 Feedback control is used to move a throttle blade to a desired position. The actual position is measured and compared to the expected throttle position determined from the commanded throttle position and a model of the desired performance. If the two values differ from each other, after taking into account hysteresis, an error can be declared.

Annex E

(informative)

Example calculation of hardware architectural metrics: “single-point fault metric” and “latent-fault metric”

This annex gives an example of calculating the single-point fault metric and the latent-fault metric for each safety goal of the item as required in alternative a) of 8.4.7 and 8.4.8.

The system for this example realises two functions implemented on a single ECU.

Function 1 has one input (temperature measured via sensor R3) and one output (valve 2 controlled by I71) and its behaviour is to open valve 2 when the temperature is higher than 90 °C.

If no current flows through I71, valve 2 is open.

The associated safety goal 1 is “valve 2 shall not be closed for longer than x ms when the temperature is higher than 100 °C”. The safety goal is assigned ASIL B. The safe state is: valve2 open.

The value of sensor R3 is read by the microcontroller ADC. R3 resistance decreases as the temperature rises. There is no monitoring on this input. The output stage controlling T71 is monitored by the analogue input InADC1 (Safety mechanism SM1 in the tables). In this example, we will assume that the safety mechanism SM1 is able to detect certain failure modes of T71 with a 90 % diagnostic coverage with respect to the violation of the safety goal. If a failure is detected by SM1, the safe state is activated but no lamp is switched on. Therefore, the diagnostic coverage with respect to latent faults is claimed to be only 80 % (the driver will notice the failure through the functionality degradation).

Function 2 has two inputs (wheel speed measured via sensors I1 and I2 generating pulses) and one output (valve 1 controlled by I61) and its behaviour is to open valve 1 when the vehicle speed is higher than 90 km/h.

If no current flows through I61, valve 1 is open.

The associated safety goal 2 is “valve 1 shall not be closed for longer than y ms when the speed is higher than 100 km/h”. The safety goal is assigned ASIL C. The safe state is: valve 1 open.

The values of sensors I1 and I2 pulses are read by the microcontroller. The wheel speed is computed using the mean value given by the sensors. The safety mechanism 2 (Safety Mechanism SM2 in the tables) compares both inputs. It detects the failures of each input with a 99 % diagnostic coverage. In the case of an inconsistency, Out.1 is set to 0. This opens valve1 (A “0” voltage on transistor opens the gate. A “0” voltage on I61 opens valve 1). Therefore, 99 % of the faults that have the potential to violate the safety goal are detected and lead to the safe state. When the safe state is activated, the lamp L1 is on. Therefore, these faults are 100 % perceived. The remaining 1 % of faults are residual faults and not latent faults.

The output stage controlling T61 is monitored by the analogue input InADC2 (Safety mechanism SM3 in the tables). The wheel speed is computed using the mean value given by the sensors.

The microcontroller has no internal redundancy. If no detailed information about the ratio of safe faults of a complex part is available, a conservative ratio of 50 % safe faults can be assumed. A global coverage of 90 % with respect to the violation of the safety goal, through internal self tests and the external watchdog (Safety Mechanism SM4 in the tables) is also assumed. The watchdog gets a live signal via the output 0 of the microcontroller. When the watchdog is no longer refreshed, its output goes low. A fault detection by SM4 (watchdog and microcontroller self tests) switches both functions to their safe state and switches L1 on. Therefore the diagnostic coverage with respect to the latent faults is claimed to be 100 %.

L1 is an LED on the dashboard, it is lit upon detection of a multiple-point failure, of which only a proportion can be detected, and indicates to the driver that the safe state of function 1 (valve 1 open) has been activated.

NOTE 1 The harness failures are not considered in this example.

NOTE 2 The fault model used for a given electronic part can differ depending on the application.

EXAMPLE 1 The fault model of a resistor depends if the hardware part is used in a digital input (such as R11, R12, R13...) or an analogue input (such as R3). In the first case the fault model can be “open/closed” whereas in the second case it can be “open/closed/drift”.

NOTE 3 The first metric only uses the failure mode coverage of the safety mechanisms that aim at preventing the violation of the safety goal. The second metric only uses the failure mode coverage of the safety mechanisms that aim at preventing the failure mode from being latent.

EXAMPLE 2 The failure mode “open” of R21 has the potential to violate safety goal 2 in the absence of a safety mechanism. Safety mechanism 3 detects this failure mode with a failure mode coverage of 99 % and switches the system into a safe state. When detecting this failure mode, an alert is displayed; the failure mode coverage with respect to latent failures is 100 %.

NOTE 4 In this example, assumptions on the failure mode distribution of the hardware elements have been considered. If no particular failure mode distribution can be argued or referenced, an equal distribution of the failure modes can be assumed.

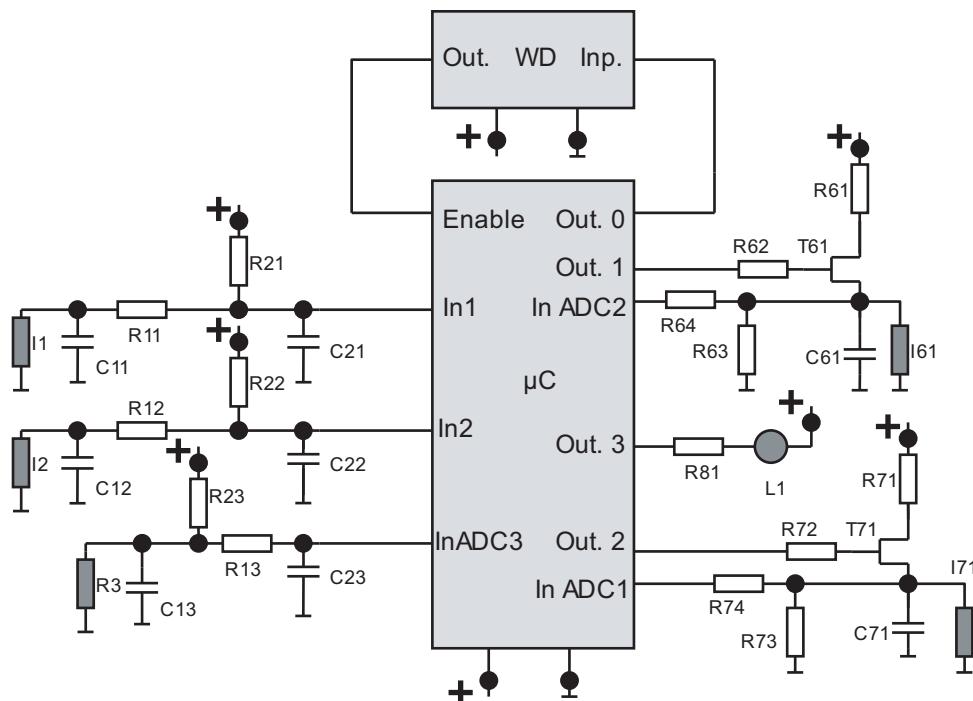


Figure E.1 — Example diagram

Please note that in the following tables, the coverage by a safety mechanism of a specific failure mode of a hardware element is called “failure mode coverage”.

Component Name	Failure rate/FIT	Safety-related component to be considered in the calculations?	Failure Mode	Failure rate distribution	Failure mode that has the potential to violate the safety goal in absence of safety mechanisms?	Safety mechanism(s) allowing to prevent the failure mode from violating the safety goal?	Failure mode coverage wrt. violation of safety goal	Residual or Single-Point Fault failure rate/FIT	Failure mode that may lead to the violation of safety goal in combination with an independent failure of another component?	Detection means? Safety mechanism(s) allowing to prevent the failure mode from being latent?	Failure mode coverage with respect to latent failures	Latent Multiple-Point Fault failure rate/FIT
R3 note 1	3	YES	open	30 %	X	none	0 %	0,9				
			closed	10 %								
			drift 0,5	30 %								
			drift 2	30 %								
R13 note 1, note 2 and note 7	2	YES	open	90 %	X	none	0 %	1,8				
			closed	10 %	X	none	0 %	0,2				
R23 note 1	2	YES	open	90 %	X	none	0 %	0,2				
			closed	10 %	X	none	0 %	0,4				
C13 note 3 and note 7	2	YES	open	20 %								
			closed	80 %								
C23	2	NO	open	20 %								
			closed	80 %								
WD	20	YES	Out. Stuck at 1	50 %								
			Out. Stuck at 0	50 %								
T71	5	YES	open circuit	50 %	X	SM1	90 %	0,25	X	none	0 %	10
			short circuit	50 %					X	SM1	80 %	0,45
R71 note 2 and note 7	2	YES	open	90 %								
			closed	10 %								
R72 note 2 and note 7	2	YES	open	90 %								
			closed	10 %								
R73	2	NO	open	90 %								
			closed	10 %								
R74 note 2 and note 7	2	YES	open	90 %								
			closed	10 %								
171	5	NO	open	70 %								
			closed	20 %								
C71 note 3	2	YES	open	20 %								
			closed	80 %								
R81	2	NO	open	90 %								
			closed	10 %								
L1	10	NO	open	90 %								
			closed	10 %								
μ C	100	YES	All	50 %	X	SM4	90 %	5	X	SM4	100 %	0
			All	50 %								
								Σ 9,65			Σ 13,25	

Total failure rate

163

Single-Point Fault Metric = $1 - (9,65/142) = 93,2 \%$

Total Safety Related

142

Total Not Safety Related

21

Latent Fault Metric = $1 - (13,25/(142-9,65)) = 90,0 \%$

Figure E.2 — Safety goal 1 (continued)

Safety goal 1 is assigned ASIL B, which has, if Table 4 is used, a single-point fault metric recommendation of $\geq 90\%$, and, if Table 5 is used, a latent-fault metric recommendation of $\geq 60\%$. The single-point fault metric recommendation is satisfied by the calculated metric of 93,2 % and the latent-fault metric recommendation is satisfied by the value of 90 %.

NOTE 1 The failure modes “open” on R3 and R13 and “closed” on R23 are single-point faults. They lead directly to the violation of the safety goal and no safety mechanism covers faults of these hardware parts.

NOTE 2 The purpose of this hardware part is electrical protection. The closed failure mode means loss of protection.

NOTE 3 The purpose of this hardware part is ESD protection. The open failure mode means loss of protection.

NOTE 4 The purpose of this hardware part is electrical protection. One failure mode is the loss of electrical protection. The other mode has the potential to violate the safety goal in the absence of safety mechanisms.

NOTE 5 The elements with failures that do not have the potential to significantly contribute to the violation of the safety goal are not considered in the calculations. Here, L1 and R81 are elements which implement a safety mechanism to prevent dual-point faults from being latent. The multiple-point faults with $n > 2$ are considered to be safe faults.

NOTE 6 The faults that lead directly to the violation of the safety goal (single-point faults or residual faults) cannot contribute anymore to the latent faults population. Therefore, for instance, the failure rate of the latent failure mode “closed gate” of T71 is computed as follows:

$$\lambda_{MPF,L} = \left[(\lambda_{T71} \times \text{FailureModeDistrib}_{\text{closed gate}}) - \lambda_{T71_{RF}} \right] \times (1 - \text{FMC}_{\text{LatentFaults}}) = [(5 \times 0,1) - 0,05] \times (1 - 0,8) = 0,09$$

NOTE 7 The classification of the failure modes leading to the loss of ESD or electrical protection is based on a case-by-case analysis and takes into consideration the likelihood of the ESD or electrical stress and the characterized effects of the ESD or electrical stress with respect to the safety goal. If for example the ESD event is likely to occur during the vehicle lifetime and its effects can lead to the violation of the safety goal in the absence of the given protection, then the failure mode leading the loss of the protection is classified as a single-point fault. This annex is an example on how to handle those cases within the metrics. In practice ESD or EMI stresses do not have this impact on typical designs similar to that of the example.

Figure E.2 — Safety goal 1

Component Name	Failure rate/FIT	Safety-related component to be considered in the calculation?	Failure Mode	Failure rate distribution	Failure mode that has the potential to violate the safety goal in absence of safety mechanisms?	Safety mechanism(s) allowing to prevent the failure mode from violating the safety goal?	Failure mode coverage wrt. violation of safety goal	Residual or Single-Point Fault failure rate/FIT	Failure mode that may lead to the violation of safety goal in combination with an independent failure of another component?	Detection means? Safety mechanism(s) allowing to prevent the failure mode from being latent?	Failure mode coverage wrt. Latent failures	Latent Multiple-Point Fault failure rate/FIT
R11 note 1, note 6 and note 7	2	YES	open	90 %	X	SM2	99 %	0,018	X	SM2	100 %	0
R12 note 1, note 6 and note 7	2	YES	closed	10 %	X	SM2	99 %	0,002	X	SM2	100 %	0
R21 note 2	2	YES	open	90 %	X	SM2	99 %	0,018	X	SM2	100 %	0
R22 note 2	2	YES	closed	10 %	X	SM2	99 %	0,002	X	SM2	100 %	0
C11 note 1, note 6 and note 7	2	YES	open	20 %	X	SM2	99 %	0,004	X	SM2	100 %	0
C12 note 1, note 6 and note 7	2	YES	closed	80 %	X	SM2	99 %	0,016	X	SM2	100 %	0
C21	2	YES	open	20 %	X	SM2	99 %	0,004	X	SM2	100 %	0
C22	2	YES	closed	80 %	X	SM2	99 %	0,016	X	SM2	100 %	0
I1	4	YES	open	20 %	X	SM2	99 %	0,016	X	SM2	100 %	0
I2	4	YES	closed	80 %	X	SM2	99 %	0,016	X	SM2	100 %	0
WD	20	YES	open	70 %	X	SM2	99 %	0,028	X	SM2	100 %	0
T61	5	YES	closed	20 %	X	SM2	99 %	0,008	X	SM2	100 %	0
R61 note 3 and note 6	2	YES	drift 0,5	5 %	X	SM2	99 %	0,002	X	SM2	100 %	0
R62 note 3 and note 6	2	YES	drift 2	5 %	X	SM2	99 %	0,028	X	none	0 %	10
R63	2	NO	Out. Stuck at 1	50 %	X	SM2	99 %	0,008	X	SM3	100 %	0
R64 note 1 and note 6	2	YES	Out. Stuck at 0	50 %	X	SM2	99 %	0,002	X	none	0 %	0,2
I61	5	NO	open circuit	50 %	X	SM3	90 %	0,25	X	none	0 %	0,2
C61 note 4 and note 6	2	YES	short circuit	50 %	X	SM3	90 %	0,25	X	none	0 %	0,2
R81	2	NO	open	90 %	X	SM4	90 %	5	X	none	0 %	1,8
L1	10	NO	closed	10 %	X	SM4	90 %	5	X	none	0 %	0,2
μC	100	YES	All	50 %							Σ 5,48	Σ 12,80

Total failure rate

176

Single-Point Fault Metric = $1 - (5,48/157) = 96,5 \%$

Total Safety Related

157

Total Not Safety Related

19

Latent Fault Metric = $1 - (13,99/(157-5,48)) = 91,6 \%$

Figure E.3 — Safety goal 2 (continued)

Safety goal 2 is assigned ASIL C, which has, if Table 4 is used, a single-point fault metric requirement of $\geq 97\%$, and, if Table 5 is used, a latent-fault metric recommendation of $\geq 80\%$. The single-point fault metric requirement is not satisfied by the calculated metric of 96,5 % and the latent-fault metric recommendation is satisfied by the value of 91,6 %.

NOTE 1 The purpose of this hardware part is electrical protection. One failure mode is the loss of electrical protection. The other mode has the potential to violate the safety goal in absence of safety mechanisms.

NOTE 2 Both failure modes have the potential to violate the safety goal in absence of safety mechanisms as in both cases, no speed pulses are transmitted. This leads to a wrong speed acquisition. The sensor is an open-collector sensor.

NOTE 3 The purpose of this hardware part is electrical protection. The close failure mode means loss of protection.

NOTE 4 The purpose of this hardware part is ESD protection. The open failure mode means loss of protection.

NOTE 5 The elements with failures that do not have the potential to significantly contribute to the violation of the safety goal are not considered in the calculations. Here, L1 and R81 are elements which implement a safety mechanism to prevent dual-point faults from being latent. The multiple-point faults with $n > 2$ are considered to be safe faults.

NOTE 6 The classification of the failure modes leading to the loss of ESD or electrical protection is based on a case-by-case analysis and takes into consideration the likelihood of the ESD or electrical stress and the characterized effects of the ESD or electrical stress with respect to the safety goal. If for example the ESD event is likely to occur during the vehicle lifetime and its effects can lead to the violation of the safety goal in the absence of the given protection, then the failure mode leading the loss of the protection is classified as a single-point fault. This annex is an example on how to handle those cases within the metrics. In practice ESD or EMI stresses do not have this impact on typical designs similar to that of the example. Moreover it is considered here that SM4 does not cover these failure modes even if they can lead to some damage to the microcontroller.

NOTE 7 The loss of electrical protection will cause a wrong input value and will be detected by SM2 and therefore will not be latent.

Figure E.3 — Safety goal 2

Annex F (informative)

Application of scaling factors

The scaling factor is a factor that is used to combine failure rates from multiple sources in the calculations of the Probabilistic Metric for random Hardware Failures (PMHF).

A target value, defined in 9.4.2.1, for the PMHF for each safety goal is derived from one of the three sources of reference:

- Table 6, or
- field data of similar well-trusted design principles, or
- quantitative analysis techniques applied on similar well-trusted design principles using failure rates as given in 8.4.3.

To verify that the hardware design meets the defined target values, failure rates are calculated based on the hardware parts involved. The failure rate of hardware parts can be estimated based on one of the three sources described in 8.4.3:

- a) hardware part failure rates data from a recognized industry source, or
- b) statistics based on field returns or tests (with an adequate confidence level), or
- c) expert judgment founded on an engineering approach based on quantitative and qualitative arguments.

Therefore, in the calculations, different failure rate sources can be used for different hardware parts of the item.

Let T_a , T_b and T_c be the three possible sources for the definition of the target values for the PMHF and F_a , F_b and F_c be the three possible sources for the estimation of a hardware part failure rate. Let $\pi_{F_i \rightarrow F_j}$ be the scaling factor between sources F_i and F_j . This factor can be used to scale a hardware part failure rate based on F_i to a failure rate based on F_j , as defined in Equation (F.1):

$$\pi_{F_i \rightarrow F_j} = \frac{\lambda_{k,F_j}}{\lambda_{k,F_i}} \quad (\text{F.1})$$

where

λ_{k,F_j} is the failure rate for a hardware part using F_j as the source for the failure rate;

λ_{k,F_i} is the failure rate for the same hardware part using F_i as the source for the failure rate.

In this case, knowing the corresponding scaling factor enables the scaling of a similar hardware part failure rate based on F_i to a failure rate based on F_j , as defined in Equation (F.2):

$$\lambda_{l,F_j} = \pi_{F_i \rightarrow F_j} \times \lambda_{l,F_i} \quad (\text{F.2})$$

Table F.1 shows the possible combinations of target values and failure rates.

NOTE 1 The targets of Table 6 are based on calculations using handbook data and under the assumption that handbook data are very pessimistic.

NOTE 2 If the source of data for the target and for the hardware part failure rate are similar, then no scaling is necessary.

Table F.1 — Possible combinations of sources of target values and failure rates to produce consistent failure rates for use in calculations

Data source for failure rates of hardware parts	Data source for target value		
	Table 6 9.4.2.1 a)	Field data 9.4.2.1 b)	Quantitative analysis 9.4.2.1 c)
Standard database 8.4.3 a)	$\lambda_{k,Fa}$ ^a	$\lambda_{k,Fb} = \pi_{Fa \rightarrow Fb} \times \lambda_{k,Fa}$	b
Statistics 8.4.3 b)	$\lambda_{k,Fa} = \pi_{Fb \rightarrow Fa} \times \lambda_{k,Fb}$	$\lambda_{k,Fb}$	b
Expert judgment 8.4.3 c)	$\lambda_{k,Fa} = \pi_{Fc \rightarrow Fa} \times \lambda_{k,Fc}$	$\lambda_{k,Fb} = \pi_{Fc \rightarrow Fb} \times \lambda_{k,Fc}$	b

^a For some types of hardware parts, different handbooks can give different estimates of the failure rate of the same type of hardware part. Therefore the scaling factor can be used to scale the failure rates of a hardware part using different handbooks.
^b To have a consistent approach, failure rates have the same origin as the failure rates used in the calculation of the target value.

The scaling is possible if sufficient evidence is available that a factor exists between two possible sources of target values.

For example, if sufficient data exist about a “predecessor” system whose failure rate can be considered representative of that expected for the item under consideration.

EXAMPLE 1 Evidence can be made available that $\frac{10^{-8}}{h}$ with a 99 % level of confidence is similar to $\frac{10^{-9}}{h}$ with a 70 % level of confidence. Therefore, failure rates based on a recognized industry source considered with a 99 % level of confidence can be scaled to failure rates based on statistics with a 70 % level of confidence using the scaling factor $\pi_{Fa \rightarrow Fb} = \left(\frac{10^{-9}}{h}\right) \div \left(\frac{10^{-8}}{h}\right) = \frac{1}{10}$ or the other way round.

NOTE 3 Based on experience, a 99 % level of confidence can be considered for failure rates based on recognized industry sources as referred to in 8.4.3.

EXAMPLE 2 From a previous design, calculated failure rates from a data handbook and warranty data have been obtained. We know that

$$\frac{\lambda_{\text{handbook}}}{\lambda_{\text{warranty}}} = \pi_{Fb \rightarrow Fa} = 10 \quad (\text{F.3})$$

where

$\lambda_{\text{handbook}}$ is the calculated failure rates from a data handbook,

$\lambda_{\text{warranty}}$ is the calculated failure rates from warranty data;

$\pi_{Fb \rightarrow Fa}$ is the resulting scaling factor.

If in a new design, we use the handbook data to determine the failure rates except for one hardware part (hardware part 1) for which we have only warranty data, then we can determine the handbook scaled data for this hardware part:

$$\lambda_{1,\text{handbook}} = \pi_{Fb \rightarrow Fa} \times \lambda_{1,\text{warranty}} \quad (\text{F.4})$$

where

$\lambda_{1,\text{handbook}}$ is the failure rate of the hardware part 1 using handbook data;

$\lambda_{1,\text{warranty}}$ is the failure rate of the hardware part 1 using warranty data.

For instance, if $\lambda_{1,\text{warranty}} = \frac{9 \times 10^{-9}}{h}$, then $\lambda_{1,\text{handbook}}$ can be calculated as $(9 \times 10^{-9}) \times 10 = \frac{9 \times 10^{-8}}{h}$.

Using this $\lambda_{1,\text{handbook}}$, a consistent evaluation of the violation of the safety goal due to random hardware failures can be done.

Bibliography

- [1] ISO 7637-2, *Road vehicles — Electrical disturbances from conduction and coupling — Part 2: Electrical transient conduction along supply lines only*
- [2] ISO 7637-3, *Road vehicles — Electrical disturbances from conduction and coupling — Part 3: Electrical transient transmission by capacitive and inductive coupling via lines other than supply lines*
- [3] ISO 10605, *Road vehicles — Test methods for electrical disturbances from electrostatic discharge*
- [4] ISO 11452-2, *Road vehicles — Component test methods for electrical disturbances from narrowband radiated electromagnetic energy — Part 2: Absorber-lined shielded enclosure*
- [5] ISO 11452-4, *Road vehicles — Component test methods for electrical disturbances from narrowband radiated electromagnetic energy — Part 4: Harness excitation methods*
- [6] ISO 16750-2, *Road vehicles — Environmental conditions and testing for electrical and electronic equipment — Part 2: Electrical loads*
- [7] ISO 16750-4, *Road vehicles — Environmental conditions and testing for electrical and electronic equipment — Part 4: Climatic loads*
- [8] ISO 16750-5, *Road vehicles — Environmental conditions and testing for electrical and electronic equipment — Part 5: Chemical loads*
- [9] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*
- [10] IEC 61709, *Electronic components — Reliability — Reference conditions for failure rates and stress models for conversion*
- [11] IEC 62061:2005, *Safety of machinery — Functional safety of safety-related electrical, electronic and programmable electronic control systems*
- [12] IEC/TR 62380, *Reliability data handbook — Universal model for reliability prediction of electronics components, PCBs and equipment*
- [13] EN 50129:2003, *Railway applications — Communication, signalling and processing systems — Safety related electronic systems for signalling*
- [14] MIL HDBK 217 F notice 2, *Military handbook: Reliability prediction of electronic equipment*
- [15] MIL HDBK 338, *Military handbook: Electronic reliability design handbook*
- [16] Nprd 95, Non-electronic Parts Reliability Data
- [17] RIAC FMD 97, Failure Mode / Mechanism Distributions
- [18] RIAC HDBK 217 Plus, Reliability Prediction Models
- [19] UTE C80-811, Reliability methodology for electronic systems
- [20] VAN DE GOOR, A.J.: *Testing Semiconductor Devices, Theory and Practice*, A.J. van de Goor/ComTex Publishing, 1999

- [21] SUNDARAM, P. and D'AMBROSIO, J.G., *Controller Integrity in Automotive Failsafe System Architectures*, SAE 2006 World Congress, 2006-01-0840
- [22] FRUELING, T., *Delphi Secured Microcontroller Architecture*, SAE 2000 World Congress, SAE# 2000-01-1052
- [23] MAHMOOD, A. and MCCLUSKEY, E.J., "Concurrent Error Detection Using Watchdog Processors – A Survey", *IEEE Trans. Computers*, 37(2), 160-174 (1988)
- [24] LEAPHART, E., CZERNY, B., D'AMBROSIO, J., et al, *Survey of Software Failsafe Techniques for Safety-Critical Automotive Applications*, SAE 2005 World Congress, 2005-01-0779
- [25] MARIANI, R., FUHRMANN, P., VITTORELLI, B., *Cost-effective Approach to Error Detection for an Embedded Automotive Platform*, 2006-01-0837, SAE 2006 World Congress & Exhibition, April 2006, Detroit, MI, USA"
- [26] PATEL, J., FUNG, L. "Concurrent Error Detection in ALU's by Recomputing with Shifted Operands", *IEEE Transactions on Computers*, Vol. C-31, pp.417-422, July 1982
- [27] FORIN, P., *Vital Coded Microprocessor: Principles and Application for various Transit Systems*, Proc. IFAC-GCCT, Paris, France, 1989
- [28] RAMABADRAN, T.V.; Gaitonde, S.S. (1988). "A tutorial on CRC computations". *IEEE Micro* 8 (4): 62–75, 1988
- [29] Koopman, Philip; Chakravarty, Tridib (2004). *Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks* The International Conference on Dependable Systems and Networks, DSN-2004, http://www.ece.cmu.edu/~koopman/rooses/dsn04/koopman04_crc_poly_embedded.pdf

ICS 43.040.10

Price based on 76 pages

INTERNATIONAL
STANDARD

ISO
26262-6

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 6:
Product development at the software
level**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 6: Développement du produit au niveau du logiciel*



Reference number
ISO 26262-6:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	v
Introduction.....	vi
1 Scope.....	1
2 Normative references.....	2
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	2
4.3 ASIL-dependent requirements and recommendations	3
5 Initiation of product development at the software level.....	3
5.1 Objectives	3
5.2 General	3
5.3 Inputs to this clause.....	4
5.4 Requirements and recommendations	4
5.5 Work products	6
6 Specification of software safety requirements.....	6
6.1 Objectives	6
6.2 General	7
6.3 Inputs to this clause.....	7
6.4 Requirements and recommendations	7
6.5 Work products	9
7 Software architectural design	9
7.1 Objectives	9
7.2 General	9
7.3 Inputs to this clause.....	9
7.4 Requirements and recommendations	10
7.5 Work products	15
8 Software unit design and implementation	15
8.1 Objectives	15
8.2 General	15
8.3 Inputs to this clause.....	16
8.4 Requirements and recommendations	16
8.5 Work products	18
9 Software unit testing	19
9.1 Objectives	19
9.2 General	19
9.3 Inputs to this clause.....	19
9.4 Requirements and recommendations	19
9.5 Work products	21
10 Software integration and testing	22
10.1 Objectives	22
10.2 General	22
10.3 Inputs to this clause.....	22
10.4 Requirements and recommendations	23
10.5 Work products	25
11 Verification of software safety requirements	25

11.1	Objectives	25
11.2	General.....	25
11.3	Inputs to this clause	25
11.4	Requirements and recommendations	26
11.5	Work products.....	27
Annex A (informative) Overview of and workflow of management of product development at the software level		28
Annex B (informative) Model-based development.....		31
Annex C (normative) Software configuration.....		33
Annex D (informative) Freedom from interference between software elements		38
Bibliography		40

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-6 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

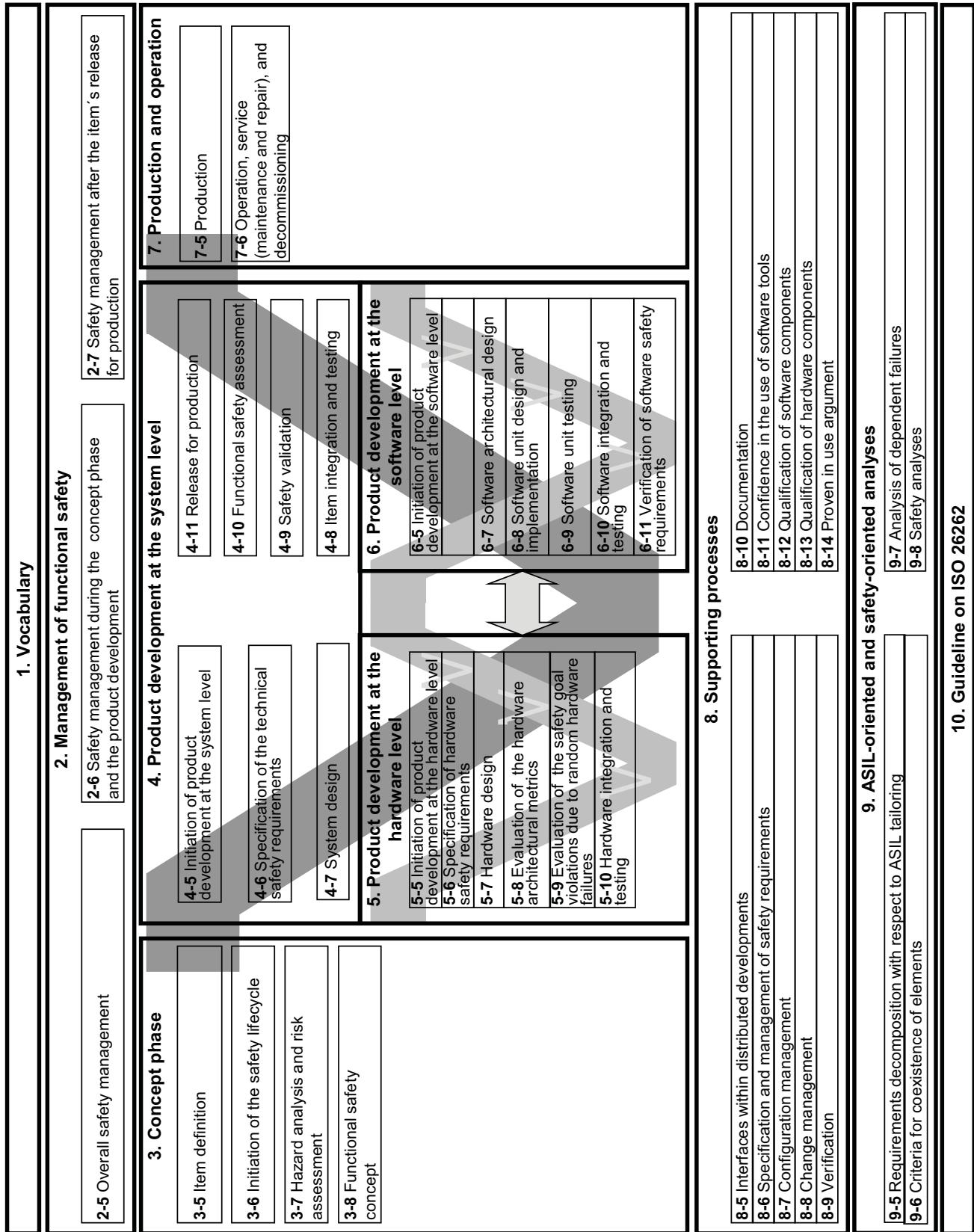


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 6: Product development at the software level

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for product development at the software level for automotive applications, including the following:

- requirements for initiation of product development at the software level,
- specification of the software safety requirements,
- software architectural design,
- software unit design and implementation,
- software unit testing,
- software integration and testing, and
- verification of software safety requirements.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- tailoring of the safety activities in accordance with ISO 26262-2 has been planned and shows that the requirement does not apply, or
- a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or

- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Initiation of product development at the software level

5.1 Objectives

The objective of this sub-phase is to plan and initiate the functional safety activities for the sub-phases of the software development.

5.2 General

The initiation of the software development is a planning activity, where software development sub-phases and their supporting processes (see ISO 26262-8 and ISO 26262-9) are determined and planned according to the extent and complexity of the item development. The software development sub-phases and supporting processes are initiated by determining the appropriate methods in order to comply with the requirements and their respective ASIL. The methods are supported by guidelines and tools, which are determined and planned for each sub-phase and supporting process.

NOTE Tools used for software development can include tools other than software tools.

EXAMPLE Tools used for testing phases.

The planning of the software development includes the coordination with the product development at the system level (see ISO 26262-4) and the hardware level (see ISO 26262-5).

5.3 Inputs to this clause

5.3.1 Prerequisites

The following information shall be available:

- project plan (refined) in accordance with ISO 26262-4:2011, 5.5.1;
- safety plan (refined) in accordance with ISO 26262-4:2011, 5.5.2;
- technical safety concept in accordance with ISO 26262-4:2011, 7.5.1;
- system design specification in accordance with ISO 26262-4:2011, 7.5.2; and
- item integration and testing plan (refined) in accordance with ISO 26262-4:2011, 8.5.1.

5.3.2 Further supporting information

The following information can be considered:

- qualified software tools available (see ISO 26262-8:2011, Clause 11);
- qualified software components available (see ISO 26262-8:2011, Clause 12);
- design and coding guidelines for modelling and programming languages (from external source);
- guidelines for the application of methods (from external source); and
- guidelines for the application of tools (from external source).

5.4 Requirements and recommendations

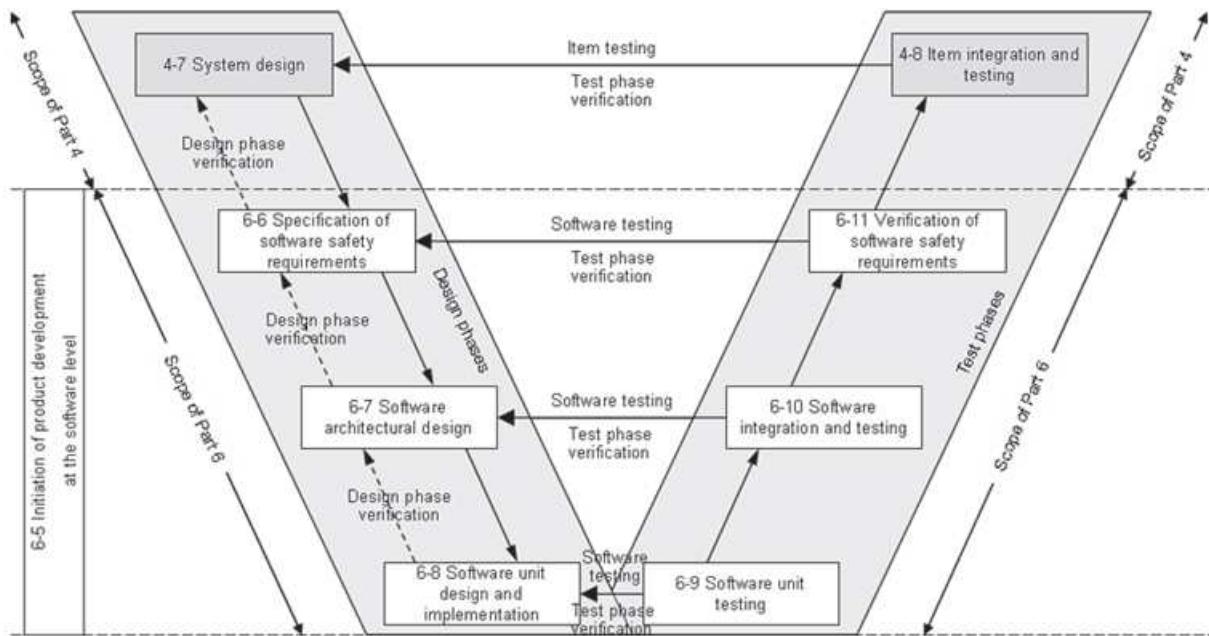
5.4.1 The activities and the determination of appropriate methods for the product development at the software level shall be planned.

5.4.2 The tailoring of the lifecycle for product development at the software level shall be performed in accordance with ISO 26262-2:2011, 6.4.5, and based on the reference phase model given in Figure 2.

5.4.3 If developing configurable software, Annex C shall be applied.

5.4.4 The software development process for the software of an item, including lifecycle phases, methods, languages and tools, shall be consistent across all the sub-phases of the software lifecycle and be compatible with the system and hardware development phases, such that the required data can be transformed correctly.

NOTE The sequencing of phases, tasks and activities, including iteration steps, for the software of an item is to ensure the consistency of the corresponding work products with the product development at the hardware level (see ISO 26262-5) and the product development at the system level (see ISO 26262-4).



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: “m-n”, where “m” represents the number of the part and “n” indicates the number of the clause, e.g. “4.7” represents Clause 7 of ISO 26262-4.

Figure 2 — Reference phase model for the software development

5.4.5 For each sub-phase of software development, the selection of the following, including guidelines for their application, shall be carried out:

- methods; and
- corresponding tools.

5.4.6 The criteria that shall be considered when selecting a suitable modelling or programming language are:

- an unambiguous definition;

EXAMPLE Syntax and semantics of the language.

- the support for embedded real time software and runtime error handling; and
- the support for modularity, abstraction and structured constructs.

Criteria that are not sufficiently addressed by the language itself shall be covered by the corresponding guidelines, or by the development environment.

NOTE 1 The selected programming language (such as ADA, C, C++, Java, Assembler or a graphical modelling language) supports the topics given in 5.4.7. Programming or modelling guidelines can be used to comply with these topics.

NOTE 2 Assembly languages can be used for those parts of the software where the use of high-level programming languages is not appropriate, such as low-level software with interfaces to the hardware, interrupt handlers, or time-critical algorithms.

5.4.7 To support the correctness of the design and implementation, the design and coding guidelines for the modelling, or programming languages, shall address the topics listed in Table 1.

NOTE 1 Coding guidelines are usually different for different programming languages.

NOTE 2 Coding guidelines can be different for model-based development.

NOTE 3 Existing coding guidelines can be modified for a specific item development.

EXAMPLE MISRA C^[3] and MISRA AC AGC^[4] are coding guidelines for the programming language C.

Table 1 — Topics to be covered by modelling and coding guidelines

	Topics	ASIL			
		A	B	C	D
1a	Enforcement of low complexity ^a	++	++	++	++
1b	Use of language subsets ^b	++	++	++	++
1c	Enforcement of strong typing ^c	++	++	++	++
1d	Use of defensive implementation techniques	o	+	++	++
1e	Use of established design principles	+	+	+	++
1f	Use of unambiguous graphical representation	+	++	++	++
1g	Use of style guides	+	++	++	++
1h	Use of naming conventions	++	++	++	++

^a An appropriate compromise of this topic with other methods in this part of ISO 26262 may be required.
^b The objectives of method 1b are

- Exclusion of ambiguously defined language constructs which may be interpreted differently by different modellers, programmers, code generators or compilers.
- Exclusion of language constructs which from experience easily lead to mistakes, for example assignments in conditions or identical naming of local and global variables.
- Exclusion of language constructs which could result in unhandled run-time errors.

^c The objective of method 1c is to impose principles of strong typing where these are not inherent in the language.

5.5 Work products

5.5.1 Safety plan (refined) resulting from requirements 5.4.1 to 5.4.7.

5.5.2 Software verification plan resulting from requirements 5.4.1 to 5.4.5 and 5.4.7.

5.5.3 Design and coding guidelines for modelling and programming languages resulting from requirements 5.4.6 and 5.4.7.

5.5.4 Tool application guidelines resulting from requirements 5.4.5 and 5.4.6.

6 Specification of software safety requirements

6.1 Objectives

The first objective of this sub-phase is to specify the software safety requirements. They are derived from the technical safety concept and the system design specification.

The second objective is to detail the hardware-software interface requirements initiated in ISO 26262-4:2011, Clause 7.

The third objective is to verify that the software safety requirements and the hardware-software interface requirements are consistent with the technical safety concept and the system design specification.

6.2 General

The technical safety requirements are refined and allocated to hardware and software during the system design phase given in ISO 26262-4:2011, Clause 7. The specification of the software safety requirements considers constraints of the hardware and the impact of these constraints on the software. This sub-phase includes the specification of software safety requirements to support the subsequent design phases.

6.3 Inputs to this clause

6.3.1 Prerequisites

The following information shall be available:

- technical safety concept in accordance with ISO 26262-4:2011, 7.5.1;
- system design specification in accordance with ISO 26262-4:2011, 7.5.2;
- hardware-software interface specification in accordance with ISO 26262-4:2011, 7.5.3;
- safety plan (refined) in accordance with 5.5.1;
- software verification plan in accordance with 5.5.2.

6.3.2 Further supporting information

The following information can be considered:

- hardware design specification (see ISO 26262-5:2011, 7.5.1);
- guidelines for the application of methods (from external source).

6.4 Requirements and recommendations

6.4.1 The software safety requirements shall address each software-based function whose failure could lead to a violation of a technical safety requirement allocated to software.

EXAMPLE Functions whose failure could lead to a violation of a safety requirement can be:

- functions that enable the system to achieve or maintain a safe state;
- functions related to the detection, indication and handling of faults of safety-related hardware elements;
- functions related to the detection, notification and mitigation of faults in the software itself;

NOTE 1 These include both the self-monitoring of the software in the operating system and application-specific self-monitoring of the software to detect, indicate and handle systematic faults in the application software.

- functions related to on-board and off-board tests;

NOTE 2 On-board tests can be carried out by the system itself or through other systems within the vehicle network during operation and during the pre-run and post-run phase of the vehicle.

NOTE 3 Off-board tests refer to the testing of the safety-related functions or properties during production or in service.

- functions that allow modifications of the software during production and service; and
- functions related to performance or time-critical operations.

6.4.2 The specification of the software safety requirements shall be derived from the technical safety concept and the system design in accordance with ISO 26262-4:2011, 7.4.1 and 7.4.5, and shall consider:

- a) the specification and management of safety requirements in accordance with ISO 26262-8:2011, Clause 6;
- b) the specified system and hardware configurations;

EXAMPLE 1 Configuration parameters can include gain control, band pass frequency and clock prescaler.

- c) the hardware-software interface specification;
- d) the relevant requirements of the hardware design specification;
- e) the timing constraints;

EXAMPLE 2 Execution or reaction time derived from the required response time at the system level.

- f) the external interfaces; and

EXAMPLE 3 Communication and user interfaces.

- g) each operating mode of the vehicle, the system, or the hardware, having an impact on the software.

EXAMPLE 4 Operating modes of hardware devices can include default, initialization, test, and advanced modes.

6.4.3 If ASIL decomposition is applied to the software safety requirements, ISO 26262-9:2011, Clause 5, shall be complied with.

6.4.4 The hardware-software interface specification initiated in ISO 26262-4:2011, Clause 7, shall be detailed down to a level allowing the correct control and usage of hardware, and shall describe each safety-related dependency between hardware and software.

6.4.5 If other functions in addition to those functions for which safety requirements are specified in 6.4.1 are carried out by the embedded software, these functions shall be specified, or else a reference made to their specification.

6.4.6 The verification of the software safety requirements and of the refined specification of the hardware software interface shall be planned in accordance with ISO 26262-8:2011, Clause 9.

6.4.7 The refined hardware-software interface specification shall be verified jointly by the persons responsible for the system, hardware and software development.

6.4.8 The software safety requirements and the refined hardware-software interface requirements shall be verified in accordance with ISO 26262-8:2011, Clauses 6 and 9, to show their:

- a) compliance and consistency with the technical safety requirements;
- b) compliance with the system design; and
- c) consistency with the hardware-software interface.

6.5 Work products

6.5.1 Software safety requirements specification resulting from requirements 6.4.1 to 6.4.3 and 6.4.5.

6.5.2 Hardware-software interface specification (refined) resulting from requirement 6.4.4.

NOTE This work product refers to the same work product as given in ISO 26262-5:2011 6.5.2

6.5.3 Software verification plan (refined) resulting from requirement 6.4.6.

6.5.4 Software verification report resulting from requirements 6.4.7 and 6.4.8.

7 Software architectural design

7.1 Objectives

The first objective of this sub-phase is to develop a software architectural design that realizes the software safety requirements.

The second objective of this sub-phase is to verify the software architectural design.

7.2 General

The software architectural design represents all software components and their interactions in a hierarchical structure. Static aspects, such as interfaces and data paths between all software components, as well as dynamic aspects, such as process sequences and timing behaviour are described.

NOTE The software architectural design is not necessarily limited to one microcontroller or ECU, and is related to the technical safety concept and system design. The software architecture for each microcontroller is also addressed by this chapter.

In order to develop a software architectural design both software safety requirements as well as all non-safety-related requirements are implemented. Hence in this sub-phase safety-related and non-safety-related requirements are handled within one development process.

The software architectural design provides the means to implement the software safety requirements and to manage the complexity of the software development.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- safety plan (refined) in accordance with 5.5.1;
- design and coding guidelines for modelling and programming languages in accordance with 5.5.3;
- hardware-software interface specification in accordance with ISO 26262-4:2011, 7.5.3;
- software safety requirements specification in accordance with 6.5.1;
- software verification plan (refined) in accordance with 6.5.3; and
- software verification report in accordance with 6.5.4.

7.3.2 Further supporting information

The following information can be considered:

- technical safety concept (see ISO 26262-4:2011, 7.5.1);
- system design specification (see ISO 26262-4:2011, 7.5.2);
- qualified software components available (see ISO 26262-8:2011, Clause 12);
- tool application guidelines in accordance with 5.5.4; and
- guidelines for the application of methods (from external source).

7.4 Requirements and recommendations

7.4.1 To ensure that the software architectural design captures the information necessary to allow the subsequent development activities to be performed correctly and effectively, the software architectural design shall be described with appropriate levels of abstraction by using the notations for software architectural design listed in Table 2.

Table 2 — Notations for software architectural design

	Methods	ASIL			
		A	B	C	D
1a	Informal notations	++	++	+	+
1b	Semi-formal notations	+	++	++	++
1c	Formal notations	+	+	+	+

7.4.2 During the development of the software architectural design the following shall be considered:

- a) the verifiability of the software architectural design;

NOTE This implies bi-directional traceability between the software architectural design and the software safety requirements.

- b) the suitability for configurable software;
- c) the feasibility for the design and implementation of the software units;
- d) the testability of the software architecture during software integration testing; and
- e) the maintainability of the software architectural design.

7.4.3 In order to avoid failures resulting from high complexity, the software architectural design shall exhibit the following properties by use of the principles listed in Table 3:

- a) modularity;
- b) encapsulation; and
- c) simplicity.

Table 3 — Principles for software architectural design

	Methods	ASIL			
		A	B	C	D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components ^a	++	++	++	++
1c	Restricted size of interfaces ^a	+	+	+	+
1d	High cohesion within each software component ^b	+	++	++	++
1e	Restricted coupling between software components ^{a, b, c}	+	++	++	++
1f	Appropriate scheduling properties	++	++	++	++
1g	Restricted use of interrupts ^{a, d}	+	+	+	++

^a In methods 1b, 1c, 1e and 1g "restricted" means to minimize in balance with other design considerations.
^b Methods 1d and 1e can, for example, be achieved by separation of concerns which refers to the ability to identify, encapsulate, and manipulate those parts of software that are relevant to a particular concept, goal, task, or purpose.
^c Method 1e addresses the limitation of the external coupling of software components.
^d Any interrupts used have to be priority-based.

NOTE An appropriate compromise between the methods listed in Table 3 can be necessary since the methods are not mutually exclusive.

7.4.4 The software architectural design shall be developed down to the level where all software units are identified.

7.4.5 The software architectural design shall describe:

- a) the static design aspects of the software components; and

NOTE 1 Static design aspects address:

- the software structure including its hierarchical levels;
- the logical sequence of data processing;
- the data types and their characteristics;
- the external interfaces of the software components;
- the external interfaces of the software; and
- the constraints including the scope of the architecture and external dependencies.

NOTE 2 In the case of model-based development, modelling the structure is an inherent part of the overall modelling activities.

- b) the dynamic design aspects of the software components.

NOTE 1 Dynamic design aspects address:

- the functionality and behaviour;
- the control flow and concurrency of processes;
- the data flow between the software components;
- the data flow at external interfaces; and

— the temporal constraints.

NOTE 2 To determine the dynamic behaviour (e.g. of tasks, time slices and interrupts) the different operating states (e.g. power-up, shut-down, normal operation, calibration and diagnosis) are considered.

NOTE 3 To describe the dynamic behaviour (e.g. of tasks, time slices and interrupts) the communication relationships and their allocation to the system hardware (e.g. CPU and communication channels) are specified.

7.4.6 Every safety-related software component shall be categorized as one of the following:

- a) newly developed;
- b) reused with modifications; or
- c) reused without modifications.

7.4.7 Safety-related software components that are newly developed or reused with modifications shall be developed in accordance with ISO 26262.

NOTE In these cases ISO 26262-8:2011, Clause 12, does not apply.

7.4.8 Safety-related software components that are reused without modifications shall be qualified in accordance with ISO 26262-8:2011, Clause 12.

NOTE The use of qualified software components does not affect the applicability of Clauses 10 and 11. However, some activities described in Clauses 8 and 9 can be omitted.

7.4.9 The software safety requirements shall be allocated to the software components. As a result, each software component shall be developed in compliance with the highest ASIL of any of the requirements allocated to it.

NOTE Following this allocation, further refinement of the software safety requirements can be necessary.

7.4.10 If the embedded software has to implement software components of different ASILs, or safety-related and non-safety-related software components, then all of the embedded software shall be treated in accordance with the highest ASIL, unless the software components meet the criteria for coexistence in accordance with ISO 26262-9:2011, Clause 6.

7.4.11 If software partitioning (see Annex D) is used to implement freedom from interference between software components it shall be ensured that:

- a) the shared resources are used in such a way that freedom from interference of software partitions is ensured;

NOTE 1 Tasks within a software partition are not free from interference among each other.

NOTE 2 One software partition cannot change the code or data of another software partition nor command non-shared resources of other software partitions.

NOTE 3 The service received from shared resources by one software partition cannot be affected by another software partition. This includes the performance of the resources concerned, as well as the rate, latency, jitter and duration of scheduled access to the resource.

- b) the software partitioning is supported by dedicated hardware features or equivalent means (this requirement applies to ASIL D, in accordance with 4.3);
- c) the part of the software that implements the software partitioning is developed in compliance with the same or an ASIL higher than the highest ASIL assigned to the requirements of the software partitions; and

NOTE In general the operating system provides or supports software partitioning.

- d) the verification of the software partitioning during software integration and testing (in accordance with Clause 10) is performed.

7.4.12 An analysis of dependent failures in accordance with ISO 26262-9:2011, Clause 7, shall be carried out if the implementation of software safety requirements relies on freedom from interference or sufficient independence between software components.

7.4.13 Safety analysis shall be carried out at the software architectural level in accordance with ISO 26262-9:2011, Clause 8, in order to:

- identify or confirm the safety-related parts of the software; and
- support the specification and verify the efficiency of the safety mechanisms.

NOTE Safety mechanisms can be specified to cover both issues associated with random hardware failures as well as software faults.

7.4.14 To specify the necessary software safety mechanisms at the software architectural level, based on the results of the safety analysis in accordance with 7.4.13, mechanisms for error detection as listed in Table 4 shall be applied.

NOTE When not directly required by technical safety requirements allocated to software, the use of software safety mechanisms is reviewed at the system level to analyse the potential impact on the system behaviour.

Table 4 — Mechanisms for error detection at the software architectural level

	Methods	ASIL			
		A	B	C	D
1a	Range checks of input and output data	++	++	++	++
1b	Plausibility check ^a	+	+	+	++
1c	Detection of data errors ^b	+	+	+	+
1d	External monitoring facility ^c	o	+	+	++
1e	Control flow monitoring	o	+	++	++
1f	Diverse software design	o	o	+	++

^a Plausibility checks can include using a reference model of the desired behaviour, assertion checks, or comparing signals from different sources.

^b Types of methods that may be used to detect data errors include error detecting codes and multiple data storage.

^c An external monitoring facility can be, for example, an ASIC or another software element performing a watchdog function.

7.4.15 This subclause applies to ASIL (A), (B), C and D, in accordance with 4.3: to specify the necessary software safety mechanisms at the software architectural level, based on the results of the safety analysis in accordance with 7.4.13, mechanisms for error handling as listed in Table 5 shall be applied.

NOTE 1 When not directly required by technical safety requirements allocated to software, the use of software safety mechanisms is reviewed at the system level to analyse the potential impact on the system behaviour.

NOTE 2 The analysis at software architectural level of possible hazards due to hardware is described in ISO 26262-5.

Table 5 — Mechanisms for error handling at the software architectural level

	Methods	ASIL			
		A	B	C	D
1a	Static recovery mechanism ^a	+	+	+	+
1b	Graceful degradation ^b	+	+	++	++
1c	Independent parallel redundancy ^c	o	o	+	++
1d	Correcting codes for data	+	+	+	+

^a Static recovery mechanisms can include the use of recovery blocks, backward recovery, forward recovery and recovery through repetition.
^b Graceful degradation at the software level refers to prioritizing functions to minimize the adverse effects of potential failures on functional safety.
^c Independent parallel redundancy can be realized as dissimilar software in each parallel path.

7.4.16 If new hazards introduced by the software architectural design are not already covered by an existing safety goal, they shall be introduced and evaluated in the hazard analysis and risk assessment in accordance with the change management process in ISO 26262-8:2011, Clause 8.

NOTE Newly identified hazards, not already reflected in a safety goal, are usually non-functional hazards. If those non-functional hazards are outside the scope of this standard then it is recommended that they be annotated in the hazard analysis and risk assessment with the following statement “No ASIL is assigned to this hazard as it is not within the scope of ISO 26262.” However, an ASIL is allowed for reference purposes.

7.4.17 An upper estimation of required resources for the embedded software shall be made, including:

- a) the execution time;
- b) the storage space; and

EXAMPLE RAM for stacks and heaps, ROM for program and non-volatile data.

- c) the communication resources.

7.4.18 The software architectural design shall be verified in accordance with ISO 26262-8:2011, Clause 9, and by using the software architectural design verification methods listed in Table 6 to demonstrate the following properties:

- a) compliance with the software safety requirements;
- b) compatibility with the target hardware; and

NOTE This includes the resources as specified in 7.4.17.

- c) adherence to design guidelines.

Table 6 — Methods for the verification of the software architectural design

	Methods	ASIL			
		A	B	C	D
1a	Walk-through of the design ^a	++	+	o	o
1b	Inspection of the design ^a	+	++	++	++
1c	Simulation of dynamic parts of the design ^b	+	+	+	++
1d	Prototype generation	o	o	+	++
1e	Formal verification	o	o	+	+
1f	Control flow analysis ^c	+	+	++	++
1g	Data flow analysis ^c	+	+	++	++

^a In the case of model-based development these methods can be applied to the model.

^b Method 1c requires the usage of executable models for the dynamic parts of the software architecture.

^c Control and data flow analysis may be limited to safety-related components and their interfaces.

7.5 Work products

7.5.1 Software architectural design specification resulting from requirements 7.4.1 to 7.4.6, 7.4.9, 7.4.10, 7.4.14, 7.4.15 and 7.4.17.

7.5.2 Safety plan (refined) resulting from requirement 7.4.7.

7.5.3 Software safety requirements specification (refined) resulting from requirement 7.4.9.

7.5.4 Safety analysis report resulting from requirement 7.4.13.

7.5.5 Dependent failures analysis report resulting from requirement 7.4.12.

7.5.6 Software verification report (refined) resulting from requirement 7.4.18.

8 Software unit design and implementation

8.1 Objectives

The first objective of this sub-phase is to specify the software units in accordance with the software architectural design and the associated software safety requirements.

The second objective of this sub-phase is to implement the software units as specified.

The third objective of this sub-phase is the static verification of the design of the software units and their implementation.

8.2 General

Based on the software architectural design, the detailed design of the software units is developed. The detailed design will be implemented as a model or directly as source code, in accordance with the modelling or coding guidelines respectively. The detailed design and the implementation are statically verified before proceeding to the software unit testing phase. The implementation-related properties are achievable at the source code level if manual code development is used. If model-based development with automatic code generation is used, these properties apply to the model and need not apply to the source code.

In order to develop a single software unit design both software safety requirements as well as all non-safety-related requirements are implemented. Hence in this sub-phase safety-related and non-safety-related requirements are handled within one development process.

The implementation of the software units includes the generation of source code and the translation into object code.

8.3 Inputs to this clause

8.3.1 Prerequisites

The following information shall be available:

- design and coding guidelines for modelling and programming languages in accordance with 5.5.3;
- software verification plan (refined) in accordance with 6.5.3;
- software architectural design specification in accordance with 7.5.1;
- safety plan (refined) in accordance with 7.5.2;
- software safety requirements specification (refined) in accordance with 7.5.3; and
- software verification report (refined) in accordance with 7.5.6.

8.3.2 Further supporting information

The following information can be considered:

- technical safety concept (see ISO 26262-4:2011, 7.5.1);
- system design specification (see ISO 26262-4:2011, 7.5.2);
- tool application guidelines in accordance with 5.5.4;
- hardware-software interface specification (refined) (see 6.5.2);
- safety analysis report in accordance with 7.5.4; and
- guidelines for the application of methods (from external source).

8.4 Requirements and recommendations

8.4.1 The requirements of this subclause shall be complied with if the software unit is safety-related.

NOTE “Safety-related” means that the unit implements safety requirements, or that the criteria for coexistence (see ISO 26262-9:2011, Clause 6) of the unit with other units are not satisfied.

8.4.2 To ensure that the software unit design captures the information necessary to allow the subsequent development activities to be performed correctly and effectively, the software unit design shall be described using the notations listed in Table 7.

Table 7 — Notations for software unit design

	Methods	ASIL			
		A	B	C	D
1a	Natural language	++	++	++	++
1b	Informal notations	++	++	+	+
1c	Semi-formal notations	+	++	++	++
1d	Formal notations	+	+	+	+

NOTE In the case of model-based development with automatic code generation, the methods for representing the software unit design are applied to the model which serves as the basis for the code generation.

8.4.3 The specification of the software units shall describe the functional behaviour and the internal design to the level of detail necessary for their implementation.

EXAMPLE Internal design can include constraints on the use of registers and storage of data.

8.4.4 Design principles for software unit design and implementation at the source code level as listed in Table 8 shall be applied to achieve the following properties:

- a) correct order of execution of subprograms and functions within the software units, based on the software architectural design;
- b) consistency of the interfaces between the software units;
- c) correctness of data flow and control flow between and within the software units;
- d) simplicity;
- e) readability and comprehensibility;
- f) robustness;

EXAMPLE Methods to prevent implausible values, execution errors, division by zero, and errors in the data flow and control flow.

- g) suitability for software modification; and
- h) testability.

Table 8 — Design principles for software unit design and implementation

	Methods	ASIL			
		A	B	C	D
1a	One entry and one exit point in subprograms and functions ^a	++	++	++	++
1b	No dynamic objects or variables, or else online test during their creation ^{a,b}	+	++	++	++
1c	Initialization of variables	++	++	++	++
1d	No multiple use of variable names ^a	+	++	++	++
1e	Avoid global variables or else justify their usage ^a	+	+	++	++
1f	Limited use of pointers ^a	o	+	+	++
1g	No implicit type conversions ^{a,b}	+	++	++	++
1h	No hidden data flow or control flow ^c	+	++	++	++
1i	No unconditional jumps ^{a,b,c}	++	++	++	++
1j	No recursions	+	+	++	++

^a Methods 1a, 1b, 1d, 1e, 1f, 1g and 1i may not be applicable for graphical modelling notations used in model-based development.
^b Methods 1g and 1i are not applicable in assembler programming.
^c Methods 1h and 1i reduce the potential for modelling data flow and control flow through jumps or global variables.

NOTE For the C language, MISRA C^[3] covers many of the methods listed in Table 8.

8.4.5 The software unit design and implementation shall be verified in accordance with ISO 26262-8:2011 Clause 9, and by applying the verification methods listed in Table 9, to demonstrate:

- a) the compliance with the hardware-software interface specification (in accordance with ISO 26262-5:2011, 6.4.10);
 - b) the fulfilment of the software safety requirements as allocated to the software units (in accordance with 7.4.9) through traceability;
 - c) the compliance of the source code with its design specification;
- NOTE In the case of model-based development, requirement c) still applies.
- d) the compliance of the source code with the coding guidelines (see 5.5.3); and
 - e) the compatibility of the software unit implementations with the target hardware.

Table 9 — Methods for the verification of software unit design and implementation

	Methods	ASIL			
		A	B	C	D
1a	Walk-through ^a	++	+	o	o
1b	Inspection ^a	+	++	++	++
1c	Semi-formal verification	+	+	++	++
1d	Formal verification	o	o	+	+
1e	Control flow analysis ^{b,c}	+	+	++	++
1f	Data flow analysis ^{b,c}	+	+	++	++
1g	Static code analysis	+	++	++	++
1h	Semantic code analysis ^d	+	+	+	+

^a In the case of model-based software development the software unit specification design and implementation can be verified at the model level.

^b Methods 1e and 1f can be applied at the source code level. These methods are applicable both to manual code development and to model-based development.

^c Methods 1e and 1f can be part of methods 1d, 1g or 1h.

^d Method 1h is used for mathematical analysis of source code by use of an abstract representation of possible values for the variables. For this it is not necessary to translate and execute the source code.

NOTE Table 9 lists only static verification techniques. Dynamic verification techniques (e.g. testing techniques) are covered in Tables 10, 11 and 12.

8.5 Work products

8.5.1 Software unit design specification resulting from requirements 8.4.2 to 8.4.4.

NOTE In the case of model-based development, the implementation model and supporting descriptive documentation, using techniques listed in Table 8, specifies the software units.

8.5.2 Software unit implementation resulting from requirement 8.4.4.

8.5.3 Software verification report (refined) resulting from requirement 8.4.5.

9 Software unit testing

9.1 Objectives

The objective of this sub-phase is to demonstrate that the software units fulfil the software unit design specifications and do not contain undesired functionality.

9.2 General

A procedure for testing the software unit against the software unit design specifications is established, and the tests are carried out in accordance with this procedure.

9.3 Inputs to this clause

9.3.1 Prerequisites

The following information shall be available:

- hardware-software interface specification (refined) in accordance with 6.5.2;
- software verification plan (refined) in accordance with 6.5.3;
- safety plan (refined) in accordance with 7.5.2;
- software unit design specification in accordance with 8.5.1;
- software unit implementation in accordance with 8.5.2; and
- software verification report (refined) in accordance with 8.5.3.

9.3.2 Further supporting information

The following information can be considered:

- tool application guidelines in accordance with 5.5.4; and
- guidelines for the application of methods (from external source).

9.4 Requirements and recommendations

9.4.1 The requirements of this subclause shall be complied with if the software unit is safety-related.

NOTE “Safety-related” means that the unit implements safety requirements, or that the criteria for coexistence of the unit with other units are not satisfied.

9.4.2 Software unit testing shall be planned, specified and executed in accordance with ISO 26262-8:2011, Clause 9.

NOTE 1 Based on the definitions in ISO 26262-8:2011, Clause 9, the test objects in the software unit testing are the software units.

NOTE 2 For model-based software development, the corresponding parts of the implementation model also represent objects for the test planning. Depending on the selected software development process the test objects can be the code derived from this model or the model itself.

9.4.3 The software unit testing methods listed in Table 10 shall be applied to demonstrate that the software units achieve:

- compliance with the software unit design specification (in accordance with Clause 8);
- compliance with the specification of the hardware-software interface (in accordance with ISO 26262-5:2011, 6.4.10);
- the specified functionality;
- confidence in the absence of unintended functionality;
- robustness; and

EXAMPLE The absence of inaccessible software, the effectiveness of error detection and error handling mechanisms.

- sufficient resources to support their functionality.

Table 10 — Methods for software unit testing

	Methods	ASIL			
		A	B	C	D
1a	Requirements-based test ^a	++	++	++	++
1b	Interface test	++	++	++	++
1c	Fault injection test ^b	+	+	+	++
1d	Resource usage test ^c	+	+	+	++
1e	Back-to-back comparison test between model and code, if applicable ^d	+	+	++	++

^a The software requirements at the unit level are the basis for this requirements-based test.
^b This includes injection of arbitrary faults (e.g. by corrupting values of variables, by introducing code mutations, or by corrupting values of CPU registers).
^c Some aspects of the resource usage test can only be evaluated properly when the software unit tests are executed on the target hardware or if the emulator for the target processor supports resource usage tests.
^d This method requires a model that can simulate the functionality of the software units. Here, the model and code are stimulated in the same way and results compared with each other.

9.4.4 To enable the specification of appropriate test cases for the software unit testing in accordance with 9.4.3, test cases shall be derived using the methods listed in Table 11.

Table 11 — Methods for deriving test cases for software unit testing

	Methods	ASIL			
		A	B	C	D
1a	Analysis of requirements	++	++	++	++
1b	Generation and analysis of equivalence classes ^a	+	++	++	++
1c	Analysis of boundary values ^b	+	++	++	++
1d	Error guessing ^c	+	+	+	+

^a Equivalence classes can be identified based on the division of inputs and outputs, such that a representative test value can be selected for each class.
^b This method applies to interfaces, values approaching and crossing the boundaries and out of range values.
^c Error guessing tests can be based on data collected through a "lessons learned" process and expert judgment.

9.4.5 To evaluate the completeness of test cases and to demonstrate that there is no unintended functionality, the coverage of requirements at the software unit level shall be determined and the structural coverage shall be measured in accordance with the metrics listed in Table 12. If the achieved structural coverage is considered insufficient, either additional test cases shall be specified or a rationale shall be provided.

EXAMPLE 1 Analysis of structural coverage can reveal shortcomings in requirement-based test cases, inadequacies in requirements, dead code, deactivated code or unintended functionality.

EXAMPLE 2 A rationale can be given for the level of coverage achieved based on accepted dead code (e.g. code for debugging) or code segments depending on different software configurations; or code not covered can be verified using complementary methods (e.g. inspections).

Table 12 — Structural coverage metrics at the software unit level

Methods		ASIL			
		A	B	C	D
1a	Statement coverage	++	++	+	+
1b	Branch coverage	+	++	++	++
1c	MC/DC (Modified Condition/Decision Coverage)	+	+	+	++

NOTE 1 The structural coverage can be determined by the use of appropriate software tools.

NOTE 2 In the case of model-based development, the analysis of structural coverage can be performed at the model level using analogous structural coverage metrics for models.

NOTE 3 If instrumented code is used to determine the degree of coverage, it can be necessary to show that the instrumentation has no effect on the test results. This can be done by repeating the tests with non-instrumented code.

9.4.6 The test environment for software unit testing shall correspond as closely as possible to the target environment. If the software unit testing is not carried out in the target environment, the differences in the source and object code, and the differences between the test environment and the target environment, shall be analysed in order to specify additional tests in the target environment during the subsequent test phases.

NOTE 1 Differences between the test environment and the target environment can arise in the source code or object code, for example, due to different bit widths of data words and address words of the processors.

NOTE 2 Depending on the scope of the tests, the appropriate test environment for the execution of the software unit is used (e.g. the target processor, a processor emulator or a development system).

NOTE 3 Software unit testing can be executed in different environments, for example:

- model-in-the-loop tests;
- software-in-the-loop tests;
- processor-in-the-loop tests; and
- hardware-in-the-loop tests.

NOTE 4 For model-based development, software unit testing can be carried out at the model level followed by back-to-back comparison tests between the model and the object code. The back-to-back comparison tests are used to ensure that the behaviour of the models with regard to the test objectives is equivalent to the automatically-generated code.

9.5 Work products

9.5.1 Software verification plan (refined) resulting from requirements 9.4.2 to 9.4.6.

9.5.2 **Software verification specification** resulting from requirements 9.4.2 and 9.4.4 to 9.4.6.

9.5.3 **Software verification report (refined)** resulting from requirement 9.4.2.

10 Software integration and testing

10.1 Objectives

The first objective of this sub-phase is to integrate the software elements.

The second objective of this sub-phase is to demonstrate that the software architectural design is realized by the embedded software.

10.2 General

In this sub-phase, the particular integration levels and the interfaces between the software elements are tested against the software architectural design. The steps of the integration and testing of the software elements correspond directly to the hierarchical architecture of the software.

The embedded software can consist of safety-related and non-safety-related software elements.

10.3 Inputs to this clause

10.3.1 Prerequisites

The following information shall be available:

- hardware-software interface specification (refined) in accordance with 6.5.2;
- software architectural design specification in accordance with 7.5.1;
- safety plan (refined) in accordance with 7.5.2;
- software unit implementation in accordance with 8.5.2;
- software verification plan (refined) in accordance with 9.5.1;
- software verification specification in accordance with 9.5.2; and
- software verification report (refined) in accordance with 9.5.3.

10.3.2 Further supporting information

The following information can be considered:

- qualified software components available (see ISO 26262-8:2011, Clause 12);
- software tool qualification report in accordance with ISO 26262-8:2011, 11.5.2;
- tool application guidelines in accordance with 5.5.4; and
- guidelines for the application of methods (from external source).

10.4 Requirements and recommendations

10.4.1 The planning of the software integration shall describe the steps for integrating the individual software units hierarchically into software components until the embedded software is fully integrated, and shall consider:

- a) the functional dependencies that are relevant for software integration; and
- b) the dependencies between the software integration and the hardware-software integration.

NOTE For model-based development, the software integration can be replaced with integration at the model level and subsequent automatic code generation from the integrated model.

10.4.2 Software integration testing shall be planned, specified and executed in accordance with ISO 26262-8:2011, Clause 9.

NOTE 1 Based on the definitions in ISO 26262-8:2011, Clause 9, the software integration test objects are the software components.

NOTE 2 For model-based development, the test objects can be the models associated with the software components.

10.4.3 The software integration test methods listed in Table 13 shall be applied to demonstrate that both the software components and the embedded software achieve:

- a) compliance with the software architectural design in accordance with Clause 7;
- b) compliance with the specification of the hardware-software interface in accordance with ISO 26262-4:2011, Clause 7;
- c) the specified functionality;
- d) robustness; and

EXAMPLE Absence of inaccessible software; effective error detection and handling.

- e) sufficient resources to support the functionality.

Table 13 — Methods for software integration testing

	Methods	ASIL			
		A	B	C	D
1a	Requirements-based test ^a	++	++	++	++
1b	Interface test	++	++	++	++
1c	Fault injection test ^b	+	+	++	++
1d	Resource usage test ^{cd}	+	+	+	++
1e	Back-to-back comparison test between model and code, if applicable ^e	+	+	++	++

^a The software requirements at the architectural level are the basis for this requirements-based test.
^b This includes injection of arbitrary faults in order to test safety mechanisms (e.g. by corrupting software or hardware components).
^c To ensure the fulfilment of requirements influenced by the hardware architectural design with sufficient tolerance, properties such as average and maximum processor performance, minimum or maximum execution times, storage usage (e.g. RAM for stack and heap, ROM for program and data) and the bandwidth of communication links (e.g. data buses) have to be determined.
^d Some aspects of the resource usage test can only be evaluated properly when the software integration tests are executed on the target hardware or if the emulator for the target processor supports resource usage tests.
^e This method requires a model that can simulate the functionality of the software components. Here, the model and code are stimulated in the same way and results compared with each other.

10.4.4 To enable the specification of appropriate test cases for the software integration test methods selected in accordance with 10.4.3, test cases shall be derived using the methods listed in Table 14.

Table 14 — Methods for deriving test cases for software integration testing

Methods		ASIL			
		A	B	C	D
1a	Analysis of requirements	++	++	++	++
1b	Generation and analysis of equivalence classes ^a	+	++	++	++
1c	Analysis of boundary values ^b	+	++	++	++
1d	Error guessing ^c	+	+	+	+

^a Equivalence classes can be identified based on the division of inputs and outputs, such that a representative test value can be selected for each class.

^b This method applies to parameters or variables, values approaching and crossing the boundaries and out of range values.

^c Error guessing tests can be based on data collected through a “lessons learned” process and expert judgment.

10.4.5 To evaluate the completeness of tests and to obtain confidence that there is no unintended functionality, the coverage of requirements at the software architectural level by test cases shall be determined. If necessary, additional test cases shall be specified or a rationale shall be provided.

10.4.6 This subclause applies to ASIL (A), (B), C and D, in accordance with 4.3: To evaluate the completeness of test cases and to obtain confidence that there is no unintended functionality, the structural coverage shall be measured in accordance with the metrics listed in Table 15. If the achieved structural coverage is considered insufficient, either additional test cases shall be specified or a rationale shall be provided.

EXAMPLE Analysis of structural coverage can reveal shortcomings in the requirement-based test cases, inadequacies in the requirements, dead code, deactivated code or unintended functionality.

Table 15 — Structural coverage metrics at the software architectural level

Methods		ASIL			
		A	B	C	D
1a	Function coverage ^a	+	+	++	++
1b	Call coverage ^b	+	+	++	++

^a Method 1a refers to the percentage of executed software functions. This evidence can be achieved by an appropriate software integration strategy.

^b Method 1b refers to the percentage of executed software function calls.

NOTE 1 The structural coverage can be determined using appropriate software tools.

NOTE 2 In the case of model-based development, software architecture testing can be performed at the model level using analogous structural coverage metrics for models.

10.4.7 It shall be verified that the embedded software that is to be included as part of a production release in accordance with ISO 26262-4:2011, Clause 11, contains all the specified functions, and only contains other unspecified functions if these functions do not impair the compliance with the software safety requirements.

EXAMPLE In this context unspecified functions include code used for debugging or instrumentation.

NOTE If deactivation of these unspecified functions can be ensured, this is an acceptable means of compliance with this requirement. Otherwise the removal of such code is a change (see ISO 26262-8:2011, Clause 8).

10.4.8 The test environment for software integration testing shall correspond as closely as possible to the target environment. If the software integration testing is not carried out in the target environment, the differences in the source and object code and the differences between the test environment and the target environment shall be analysed in order to specify additional tests in the target environment during the subsequent test phases.

NOTE 1 Differences between the test environment and the target environment can arise in the source or object code, for example, due to different bit widths of data words and address words of the processors.

NOTE 2 Depending on the scope of the tests and the hierarchical level of integration, the appropriate test environments for the execution of the software elements are used. Such test environments can be the target processor for final integration, or a processor emulator or a development system for the previous integration steps.

NOTE 3 Software integration testing can be executed in different environments, for example:

- model-in-the-loop tests;
- software-in-the-loop tests;
- processor-in-the-loop tests; and
- hardware-in-the-loop tests.

10.5 Work products

10.5.1 Software verification plan (refined) resulting from requirements 10.4.1 to 10.4.6 and 10.4.8.

10.5.2 Software verification specification (refined) resulting from requirements 10.4.1, 10.4.2, 10.4.4, 10.4.5, 10.4.7 and 10.4.8.

10.5.3 Embedded software resulting from requirement 10.4.1.

10.5.4 Software verification report (refined) resulting from requirement 10.4.2.

11 Verification of software safety requirements

11.1 Objectives

The objective of this sub-phase is to demonstrate that the embedded software fulfils the software safety requirements.

11.2 General

The purpose of the verification of the software safety requirements is to demonstrate that the embedded software satisfies its requirements in the target environment.

11.3 Inputs to this clause

11.3.1 Prerequisites

The following information shall be available:

- software architectural design specification in accordance with 7.5.1;
- safety plan (refined) in accordance with 7.5.2;
- software safety requirements specification (refined) in accordance with 7.5.3;

- software verification plan (refined) in accordance with 10.5.1;
- software verification specification (refined) in accordance with 10.5.2;
- software verification report (refined) in accordance with 10.5.4; and
- integration testing report in accordance with ISO 26262-4:2011, 8.5.3.

11.3.2 Further supporting information

The following information can be considered:

- validation plan (refined) (see ISO 26262-4:2011, 6.5.3);
- technical safety concept (see ISO 26262-4:2011, 7.5.1);
- system design specification (see ISO 26262-4:2011, 7.5.2);
- tool application guidelines in accordance with 5.5.4; and
- guidelines for the application of methods (from external source).

11.4 Requirements and recommendations

11.4.1 The verification of the software safety requirements shall be planned, specified and executed in accordance with ISO 26262-8:2011, Clause 9.

11.4.2 To verify that the embedded software fulfils the software safety requirements, tests shall be conducted in the test environments listed in Table 16.

NOTE Test cases that already exist, for example from software integration testing, can be re-used.

Table 16 — Test environments for conducting the software safety requirements verification

Methods		ASIL			
		A	B	C	D
1a	Hardware-in-the-loop	+	+	++	++
1b	Electronic control unit network environments ^a	++	++	++	++
1c	Vehicles	++	++	++	++

^a Examples include test benches partially or fully integrating the electrical systems of a vehicle, “lab-cars” or “mule” vehicles, and “rest of the bus” simulations.

11.4.3 The testing of the implementation of the software safety requirements shall be executed on the target hardware.

11.4.4 The results of the verification of the software safety requirements shall be evaluated with regard to:

- a) compliance with the expected results;
- b) coverage of the software safety requirements; and
- c) pass or fail criteria.

11.5 Work products

- 11.5.1 **Software verification plan (refined)** resulting from requirements 11.4.1 to 11.4.3.
- 11.5.2 **Software verification specification (refined)** resulting from requirements 11.4.1 to 11.4.3.
- 11.5.3 **Software verification report (refined)** resulting from requirements 11.4.1 and 11.4.4.

Annex A (informative)

Overview of and workflow of management of product development at the software level

Table A.1 provides an overview of objectives, prerequisites and work products of the particular phases of the product development at the software level.

Table A.1 — Overview of product development at the software level

Clause	Objectives	Prerequisites	Work products
5 Initiation of product development at the software level	Plan and initiate the functional safety activities for the sub-phases of the software development activity.	Project plan (refined) (see ISO 26262-4:2011, 5.5.1) Safety plan (refined) (see ISO 26262-4:2011, 5.5.2) Technical safety concept (see ISO 26262-4:2011, 7.5.1) System design specification (see ISO 26262-4:2011, 7.5.2) Item integration and testing plan (refined) (see ISO 26262-4:2011, 7.5.4)	5.5.1 Safety plan (refined) 5.5.2 Software verification plan 5.5.3 Design and coding guidelines for modelling and programming languages 5.5.4 Tool application guidelines
6 Specification of software safety requirements	Specify software safety requirements. The software safety requirements are derived from the technical safety concept and the system design specification. Detail the hardware-software interface requirements. Verify that the software safety requirements and the hardware-software interface requirements are consistent with the technical safety concept and the system design specification.	Technical safety concept (see ISO 26262-4:2011, 7.5.1) System design specification (see ISO 26262-4:2011, 7.5.2) Hardware-software interface specification (see ISO 26262-4:2011, 7.5.6) Safety plan (refined) (see 5.5.1) Software verification plan (see 5.5.2)	6.5.1 Software safety requirements specification 6.5.2 Hardware-software interface specification (refined) 6.5.3 Software verification plan (refined) 6.5.4 Software verification report
7 Software architectural design	Develop a software architectural design that realizes the software safety requirements. Verify the software architectural design.	Safety plan (refined) (see 5.5.1) Design and coding guidelines for modelling and programming languages (see 5.5.3) Hardware-software interface specification (see ISO 26262-4:2011, 7.5.6) Software safety requirements specification (see 6.5.1) Software verification plan (refined) (see 6.5.3) Software verification report (refined) (see 6.5.4)	7.5.1 Software architectural design specification 7.5.2 Safety plan (refined) 7.5.3 Software safety requirements specification (refined) 7.5.4 Safety analysis report 7.5.5 Dependent failures analysis report 7.5.6 Software verification report (refined)

Table A.1 (continued)

Clause	Objectives	Prerequisites	Work products
8 Software unit design and implementation	<p>Specify the software units in accordance with the software architectural design and the associated software safety requirements.</p> <p>Implement the software units as specified.</p> <p>Static verification of the software unit design and their implementation.</p>	Design and coding guidelines for modelling and programming languages (see 5.5.3) Software verification plan (refined) (see 6.5.3) Software architectural design specification (see 7.5.1) Safety plan (refined) (see 7.5.2) Software safety requirements specification (refined) (see 7.5.3) Software verification report (refined) (see 7.5.6)	8.5.1 Software unit design specification 8.5.2 Software unit implementation 8.5.3 Software verification report (refined)
9 Software unit testing	Demonstrate that the software units fulfil the software unit design specifications and do not contain undesired functionality.	Hardware-software interface specification (refined) (see 6.5.2) Software verification plan (refined) (see 6.5.3) Safety plan (refined) (see 7.5.2) Software unit design specification (see 8.5.1) Software unit implementation (see 8.5.2) Software verification report (refined) (see 8.5.3)	9.5.1 Software verification plan (refined) 9.5.2 Software verification specification 9.5.3 Software verification report (refined)
10 Software integration and testing	Integrate the software elements. Demonstrate that the software architectural design is realized by the embedded software.	Hardware-software interface specification (refined) (6.5.2) Software architectural design specification (see 7.5.1) Safety plan (refined) (see 7.5.2) Software unit implementation (see 8.5.2) Software verification plan (refined) (see 9.5.1) Software verification specification (refined) (see 9.5.2) Software verification report (refined) (see 9.5.3)	10.5.1 Software verification plan (refined) 10.5.2 Software verification specification (refined) 10.5.3 Embedded software 10.5.4 Software verification report (refined)
11 Verification of software safety requirements	Demonstrate that the embedded software fulfils the software safety requirements.	Software architectural design specification (see 7.5.1) Safety plan (refined) (see 7.5.2) Software safety requirements specification (refined) (see 7.5.3) Software verification plan (refined) (see 10.5.1) Software verification specification (refined) (see 10.5.2) Software verification report (refined) (see 10.5.4) Integration testing report (see ISO 26262-4:2011, 8.5.2)	11.5.1 Software verification plan (refined) 11.5.2 Software verification specification (refined) 11.5.3 Software verification report (refined)

Table A.1 (*continued*)

Clause	Objectives	Prerequisites	Work products
Annex C Software configuration	Enable controlled changes in the behaviour of the software for different applications	See applicable prerequisites of the relevant phases of the safety lifecycle in which software configuration is applied.	C.5.1 Configuration data specification C.5.2 Calibration data specification C.5.3 Safety plan (refined) C.5.4 Configuration data C.5.5 Calibration data C.5.6 Software verification plan (refined) C.5.7 Verification specification C.5.8 Verification report

Annex B (informative)

Model-based development

B.1 Objectives

This annex describes the concept of model-based development of in-vehicle software and outlines its implications on the product development at the software level.

B.2 General

Mathematical modelling, which has been extensively used in many engineering domains, is also gaining widespread use in the development of embedded software. In the automotive sector, modelling is used for the conceptual capture of the functionality to be realized (open/closed loop control, monitoring) as well as for the simulation of real physical system behaviours (vehicle environment).

Modelling is usually carried out with commercial off-the-shelf modelling and simulation software tools. They support the development and definition of system/software elements, and their connections and interfaces by semi-formal graphical models. These models employ editable, hierarchical block diagrams (e.g. control diagrams) and extended state transition diagrams (e.g. state charts). The software tools provide the necessary means of description, computation techniques and interpreters/compilers. Graphical editors permit an intuitive development and description of complex models. Hierarchically structured modularity is used in order to control complexity. A model consists of function blocks with well-defined inputs and outputs. Function blocks are connected within the block diagram by directed edges between their interfaces, which describe signal flows. With this, they represent equations in the mathematical model, which relate the interface variables of different elements. The connection lines represent causally motivated directions of action, which define the outputs of one block as the inputs of another. Other tool-specific modelling semantics also can be used to impose order of execution and timing. The hierarchy of elements can contain several levels of refinement.

Such models can be simulated, i.e. executed. During simulation the calculation causality follows the defined directions of action until the entire model has been processed. There is a range of different solvers available for solving the equations described by the model. Variable-step solvers are used primarily for modelling the vehicle and the environment. For the development of embedded software, fixed-step solvers are used, which represent a necessary prerequisite for efficient code generation.

The modelling style described is used extensively within the scope of model-based development of embedded in-vehicle software. Typically, both an executable model of the control software (e.g. a functional model) and a model of the surrounding system (e.g. a vehicle model) and its environment (e.g. an environment model) are created early in the development cycle and are simulated together. This way, it is possible to model even highly complex automotive systems with a high degree of detail at an acceptable calculation speed and to simulate their behaviour close to reality. While the vehicle/environment model is gradually replaced during the course of development by the real system and its real environment, the functional model can serve as a blueprint for the implementation of embedded software on the control unit through code generation.

One characteristic of the model-based development paradigm is the fact that the functional model not only specifies the desired function but also provides design information and finally even serves as the basis of the implementation by means of code generation. In other words, such a functional model represents specification aspects as well as design and implementation aspects. In practice, these different aspects are reflected in an evolution of the functional model from an early specification model via a design model to an implementation model and finally its automatic transformation into code (model evolution). In comparison to code-based software development with a clear separation of phases, in model-based development a stronger coalescence

of the phases “Software safety requirements”, “Software architectural design” and “Software unit design and implementation” can be noted. Moreover, one and the same graphical modelling notation is used during the consecutive development stages. Verification activities can also be treated differently since models can be used as a useful source of information for the testing process (e.g. model-based testing), or can serve as the object to be verified. The seamless utilization of models facilitates highly consistent and efficient development.

NOTE The paradigm of model-based development does not depend on the existence of the model type mentioned above. Alternative models such as UML can be used.

Annex C (normative)

Software configuration

C.1 Objectives

The objective of software configuration is to enable controlled changes in the behaviour of the software for different applications.

C.2 General description

Configurable software enables the development of application specific software using configuration and calibration data (see Figure C.1).

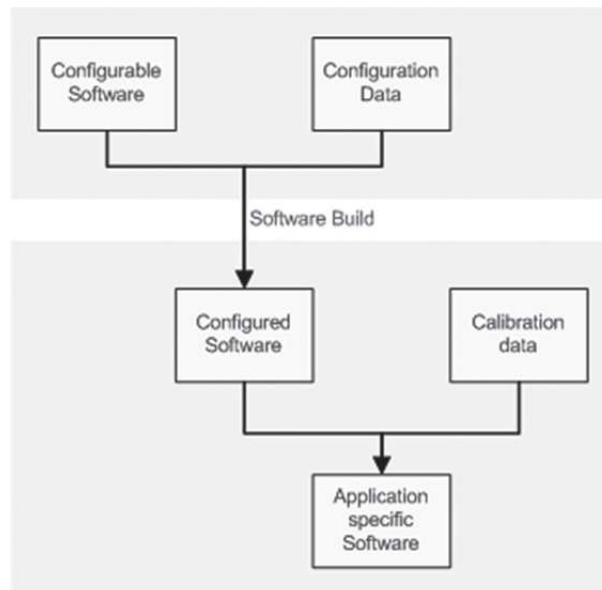


Figure C.1 — Creating application specific software

C.3 Inputs to this clause

C.3.1 Prerequisites

The prerequisites are in accordance with the relevant phases in which software configuration is applied.

C.3.2 Further supporting information

See applicable further supporting information of the relevant phases in which software configuration is applied.

C.4 Requirements and recommendations

C.4.1 The configuration data shall be specified to ensure the correct usage of the configurable software during the safety lifecycle. This shall include:

- a) the valid values of the configuration data;
- b) the intent and usage of the configuration data;
- c) the range, scaling, units; and
- d) the interdependencies between different elements of the configuration data.

C.4.2 Verification of the configuration data shall be performed to ensure:

- a) the use of values within their specified range; and
- b) the compatibility with the other configuration data.

NOTE The testing of the configured software is performed within the test phases of the software lifecycle [see Clauses 9 (Software unit testing), 10 (Software integration and testing), 11 (Verification of software safety requirements) and ISO 26262-4:2011, Clause 8 (Item integration and testing)].

C.4.3 The ASIL of the configuration data shall equal the highest ASIL of the configurable software by which it is used.

C.4.4 The verification of configurable software shall be planned, specified and executed in accordance with ISO 26262-8:2011, Clause 9. Configurable software shall be verified for the configuration data set that is to be used for the item development under consideration.

NOTE Only that part of the embedded software whose behaviour depends on the configuration data is verified against the configuration data set.

C.4.5 For configurable software a simplified software safety lifecycle in accordance with Figures C.2 or C.3 may be applied.

NOTE A combination of the following verification activities can achieve the complete verification of the configured software:

- a) “verification of the configurable software”,
- b) “verification of the configuration data”, and
- c) “verification of the configured software”.

This is achieved by either

- verifying a range of admissible configuration data in a) and showing compliance to this range in b), or
- by showing compliance to the range of admissible configuration data in b) and performing c).

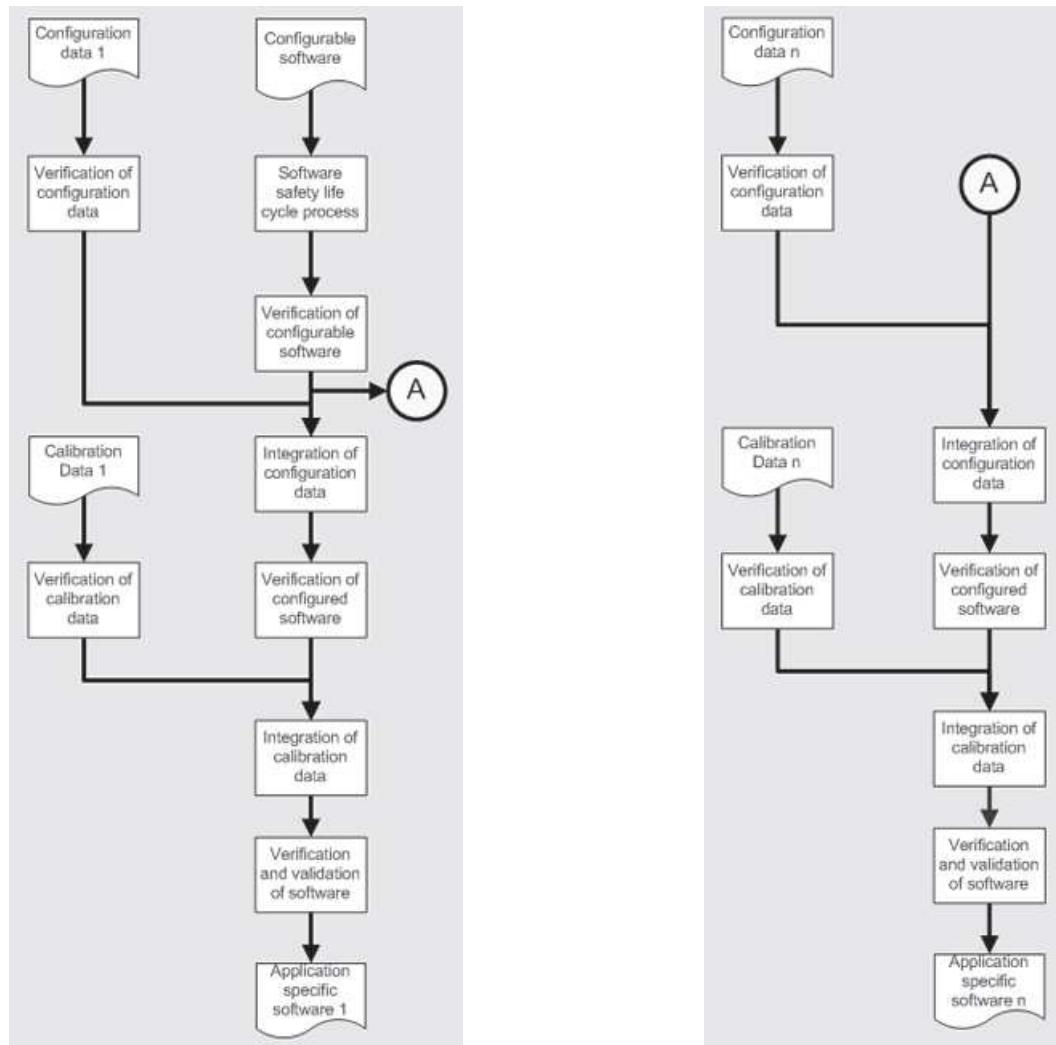


Figure C.2 — Variants of the reference phase model for software development with configurable software and different configuration data

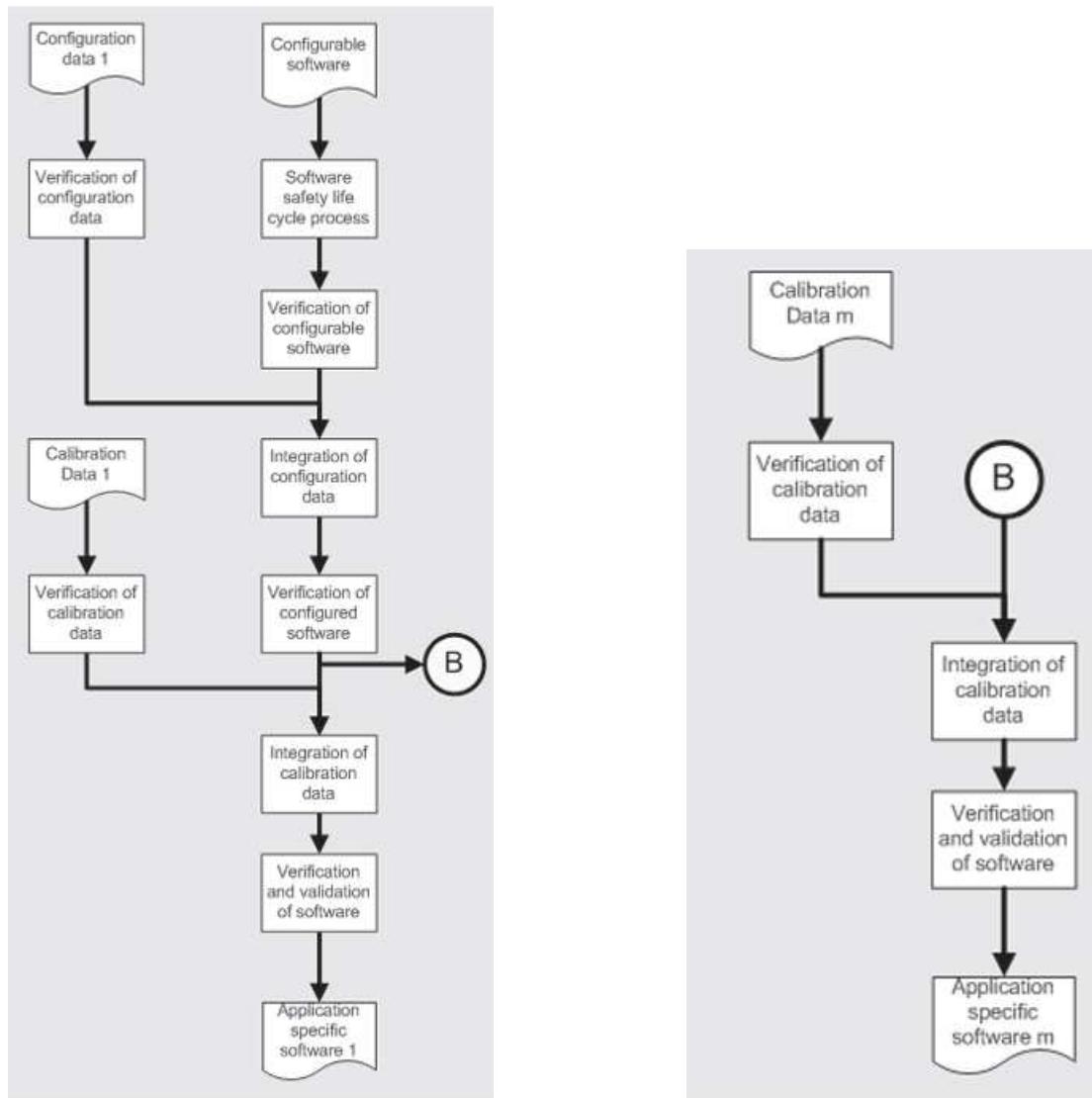


Figure C.3 — Variants of the reference phase model for software development with configurable software and different calibration data

C.4.6 The calibration data associated with software components shall be specified to ensure the correct operation and expected performance of the configured software. This shall include:

- the valid values of the calibration data;
- the intent and usage of the calibration data;
- the range, scaling and units, if applicable, with their dependence on the operating state;
- the known interdependencies between different calibration data; and

NOTE Interdependencies can exist between calibration data within one calibration data set or between calibration data in different calibration data sets such as those applied to related functions implemented in the software of separate ECUs.

- the known interdependencies between configuration data and calibration data.

NOTE Configuration data can have an impact on the configured software that uses the calibration data.

C.4.7 The verification of the calibration data shall be planned, specified and executed in accordance with ISO 26262-8:2011, Clause 9. The verification of calibration data shall examine whether the calibration data is within its specified boundaries.

NOTE Verification of calibration data can also be performed within application-specific software verification, or at runtime by the configurable software.

C.4.8 The ASIL of the calibration data shall equal the highest ASIL of the software safety requirements it can violate.

C.4.9 To detect unintended changes of safety-related calibration data, mechanisms for the detection of unintended changes of data as listed in Table C.1 shall be applied.

Table C.1 — Mechanisms for the detection of unintended changes of data

Methods	ASIL			
	A	B	C	D
1a Plausibility checks on calibration data	++	++	++	++
1b Redundant storage of calibration data	+	+	+	++
1c Error detecting codes ^a	+	+	+	++

^a Error detecting codes may also be implemented in the hardware in accordance with ISO 26262-5.

C.4.10 The planning of the generation and application of calibration data shall specify:

- a) the procedures that shall be followed;
- b) the tools for generating calibration data; and
- c) the procedures for verifying calibration data.

NOTE Verification of calibration data can include checking the value ranges of calibration data or the interdependencies between different calibration data.

C.5 Work products

C.5.1 Configuration data specification resulting from requirements C.4.1 and C.4.3.

C.5.2 Calibration data specification resulting from requirement C.4.6.

C.5.3 Safety plan (refined) resulting from requirements C.4.1, C.4.4, C.4.5, C.4.9 and C.4.10.

C.5.4 Configuration data resulting from requirement C.4.3.

C.5.5 Calibration data resulting from requirement C.4.8.

C.5.6 Software verification plan (refined) resulting from requirements C.4.2, C.4.4, C.4.7 and C.4.10.

C.5.7 Verification specification resulting from requirements C.4.4 and C.4.7.

C.5.8 Verification report resulting from requirements C.4.1, C.4.4, C.4.7 and C.4.8.

Annex D (informative)

Freedom from interference between software elements

D.1 Objectives

The objective is to provide examples of faults that can cause interference between software elements (e.g. software elements of different software partitions). Additionally, this Annex provides examples of possible mechanisms that can be considered for the prevention, or detection and mitigation of the listed faults.

NOTE The capability and effectiveness of the mechanisms used to prevent, or to detect and mitigate relevant faults is assessed during development.

D.2 General

D.2.1 Achievement of freedom from interference

To develop or evaluate the achievement of freedom from interference between software elements, the effects of the exemplary faults, and the propagation of the possible resulting failures can be considered.

D.2.2 Timing and execution

With respect to timing constraints, the effects of faults such as those listed below can be considered for the software elements executed in each software partition:

- blocking of execution;
- deadlocks;
- livelocks;
- incorrect allocation of execution time;
- incorrect synchronization between software elements.

EXAMPLE Mechanisms such as cyclic execution scheduling, fixed priority based scheduling, time triggered scheduling, monitoring of processor execution time, program sequence monitoring and arrival rate monitoring can be considered.

D.2.3 Memory

With respect to memory, the effects of faults such as those listed below can be considered for software elements executed in each software partition:

- corruption of content;
- read or write access to memory allocated to another software element.

EXAMPLE Mechanisms such as memory protection, parity bits, error-correcting code (ECC), cyclic redundancy check (CRC), redundant storage, restricted access to memory, static analysis of memory accessing software and static allocation can be used.

D.2.4 Exchange of information

With respect to the exchange of information, the causes for faults or effects of faults such as those listed below can be considered for each sender or each receiver:

- repetition of information;
- loss of information;
- delay of information;
- insertion of information;
- masquerade or incorrect addressing of information;
- incorrect sequence of information;
- corruption of information;
- asymmetric information sent from a sender to multiple receivers;
- information from a sender received by only a subset of the receivers;
- blocking access to a communication channel.

NOTE The exchange of information between elements executed in different software partitions or different ECUs includes signals, data, messages, etc.

EXAMPLE 1 Information can be exchanged using I/O-devices, data busses, etc.

EXAMPLE 2 Mechanisms such as communication protocols, information repetition, loop back of information, acknowledgement of information, appropriate configuration of I/O pins, separated point-to-point unidirectional communication objects, unambiguous bidirectional communication objects, asynchronous data communication, synchronous data communication, event-triggered data buses, event-triggered data buses with time-triggered access, time-triggered data busses, mini-slotted, bus arbitration by priority and bus arbitration by priority can be used.

EXAMPLE 3 Communication protocols can contain information such as identifiers for communication objects, keep alive messages, alive counters, sequence numbers, error detection codes and error-correcting codes.

Bibliography

- [1] ISO/IEC 12207, *Systems and software engineering — Software life cycle processes*
- [2] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*
- [3] MISRA-C:2004, *Guidelines for the use of the C language in critical systems*, ISBN 978-0-9524156-2-6, MIRA, October 2004
- [4] MISRA AC AGC, *Guidelines for the application of MISRA-C:2004 in the context of automatic code generation*, ISBN 978-1-906400-02-6, MIRA, November 2007

This page is intentionally blank.

This page is intentionally blank.

This page is intentionally blank.

ICS 43.040.10

Price based on 40 pages

INTERNATIONAL
STANDARD

ISO
26262-7

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 7:
Production and operation**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 7: Production et utilisation*



Reference number
ISO 26262-7:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction.....	v
1 Scope.....	1
2 Normative references.....	1
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	2
4.3 ASIL-dependent requirements and recommendations	3
5 Production.....	3
5.1 Objectives	3
5.2 General	3
5.3 Inputs to this clause.....	3
5.4 Requirements and recommendations	4
5.5 Work products	6
6 Operation, service (maintenance and repair), and decommissioning	6
6.1 Objectives	6
6.2 General	7
6.3 Input to this clause.....	7
6.4 Requirements and recommendations	7
6.5 Work products	9
Annex A (informative) Overview on and document flow of production and operation	10
Bibliography.....	11

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-7 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

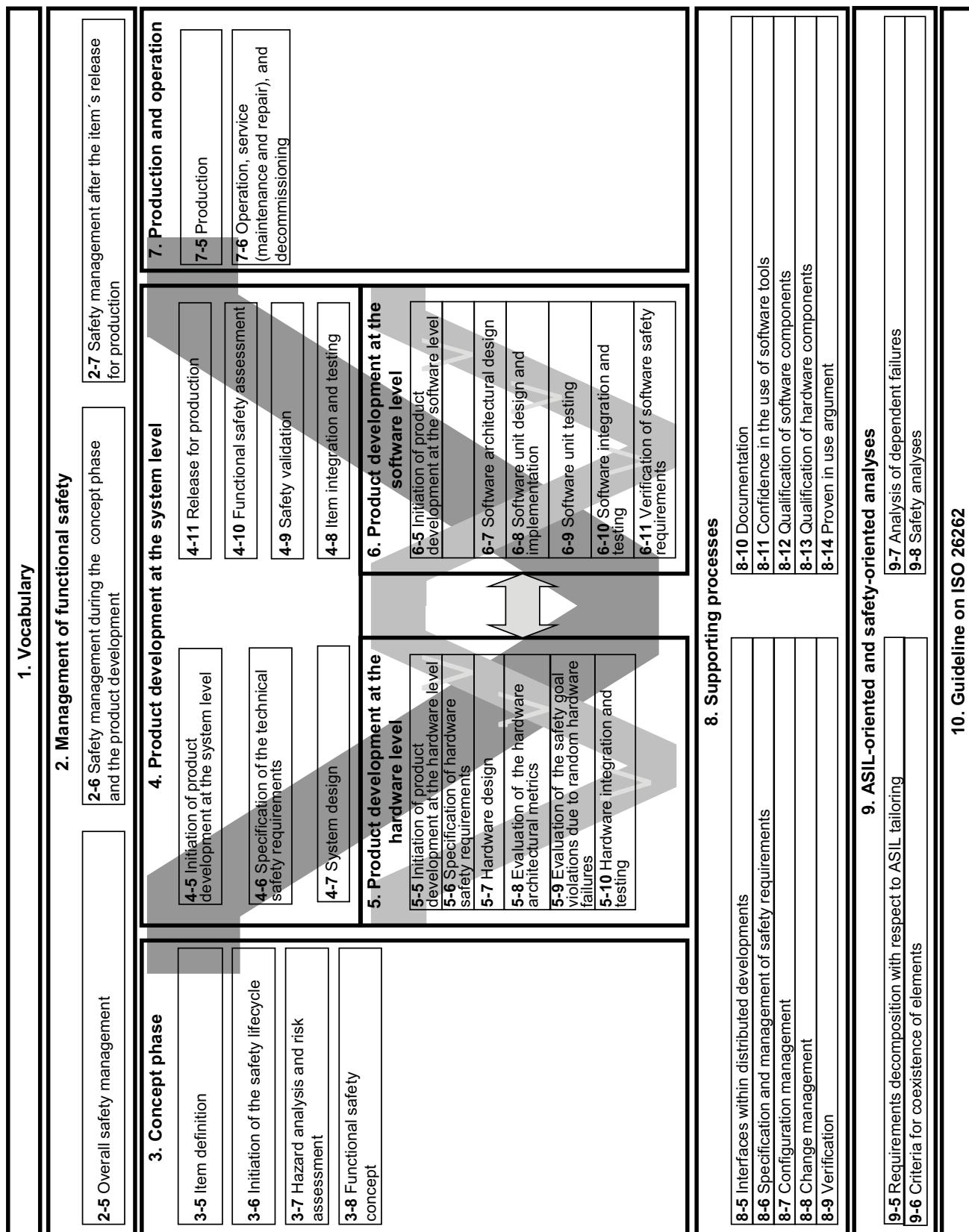


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 7: Production and operation

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for production, operation, service and decommissioning.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- a) tailoring of the safety activities in accordance with ISO 26262-2 has been planned and shows that the requirement does not apply, or
- b) a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Production

5.1 Objectives

The first objective of this clause is to develop and maintain a production process for safety-related elements or items that are intended to be installed in road vehicles.

The second objective is to achieve functional safety during the production process by the relevant manufacturer or the person or organisation responsible for the process (vehicle manufacturer, supplier, sub-supplier, etc.).

5.2 General

The compliance with safety-related special characteristics of items or elements during their production, determined during the development phases, is necessary to achieve functional safety. Examples of such safety-related special characteristics are specific process parameters (e.g. temperature range or fastening torque), material characteristics, production tolerance, or configuration.

This phase defines requirements ensuring that functional safety is achieved during the production process by including these safety-related special characteristics in production planning and control.

The requirements and recommendations of this clause apply to the production and installation in the vehicle of items, systems or elements

5.3 Inputs to this clause

5.3.1 Prerequisites

The following information shall be available:

- specification of requirements related to production, operation, service and decommissioning in accordance with ISO 26262-4:2011, 7.5.4, and ISO 26262-5:2011, 7.5.4;
- specification of dedicated measures for hardware in accordance with ISO 26262-5:2011, 9.5.2; and
- release for production report in accordance with ISO 26262-4:2011, 11.5.1.

5.3.2 Further supporting information

The following information can be considered:

- production plan (from external source); and
- production control plan (from external source).

5.4 Requirements and recommendations

5.4.1 Production planning

5.4.1.1 The production process shall be planned by evaluating the item and by considering the following:

- a) the requirements for production;

EXAMPLE Assembly instructions (e.g. the calibration and setup of a sensor); safety-related special characteristics (e.g. the tolerance for the selection of elements).

- b) the conditions for storage, transport and handling of hardware elements;

EXAMPLE Allowed storage time for the element.

- c) the approved configurations defined in the release for production documentation;

- d) the lessons learned on the capability from previously released production plans;

- e) the suitability of the production process, means of production, tools and test equipment concerning the safety-related special characteristics; and

- f) the competences of the personnel.

5.4.1.2 The production plan shall describe the production steps, sequence and methods required to achieve the functional safety of the item, system or element. It shall include:

- a) the production process flow and instructions;

- b) the production tools and means;

- c) the implementation of the traceability measures; and

EXAMPLE Labelling for the element.

- d) if applicable, the implementation of dedicated measures applying to hardware parts and specified during hardware development in accordance with ISO 26262-5:2011, 9.4.2.4.

NOTE The production process also includes processes or operations required to rework the item.

5.4.1.3 A procedure shall be defined to ensure that the correct embedded software and the associated calibration data are loaded into the ECUs as part of the production process.

EXAMPLE 1 The use of a checksum, so that the checksum of the loaded executable and configuration data is compared to the correct checksum for this particular model and vehicle configuration.

EXAMPLE 2 Read back of the part number from the software loaded into the ECUs and comparison with the target part number for that specific vehicle from the bill of materials; as well as read back and comparison of the loaded calibration data with the calibration data for that specific vehicle from the bill of materials.

5.4.1.4 When developing the production control plan, the controls' description and criteria for the item, system or element as well as the safety-related special characteristics shall be considered.

5.4.1.5 The sequence and methods of the control steps shall be described in the production control plan, together with the necessary test equipment, tools and test criteria.

5.4.1.6 Reasonably foreseeable process failures and their effects on functional safety shall be identified and the appropriate measures implemented to address the relevant process failures.

5.4.1.7 The system, hardware or software development level safety requirements on the producibility of the item, system or element arising during production planning shall be specified and directed to the persons responsible for the development (see ISO 26262-4, ISO 26262-5 and ISO 26262-6).

EXAMPLE Adding a mistake-proofing feature (poka-yoke) in a connector to ensure it is plugged into the ECU correctly during assembly.

5.4.1.8 If changes to the item, system or element are required during the production process, the change management process described in ISO 26262-8:2011, Clause 8, shall be complied with.

5.4.2 Pre-production series production

5.4.2.1 The pre-production process and its control measures should correspond to the target production process.

NOTE Pre-production series are items, systems or elements, produced before release for production.

5.4.2.2 Differences between pre-production process and target production process shall be analysed in order to identify which part of the production process can be assessed at the pre-production stage and for which part of the target production process an assessment will be required.

NOTE If the pre-production process equals the target production process, the result of assessments (e.g. proof of capability of the production process) can be used when performing the functional safety assessment in accordance with ISO 26262-2:2011, 6.4.9.4.

EXAMPLE Deviations can concern the production rate, the sequence and methods of the production or control steps, as well as necessary means of production, test equipment, and tools.

5.4.3 Production

5.4.3.1 The production process and its control measures shall be implemented and maintained as planned.

NOTE The appropriate training of the personnel involved in production is part of this implementation.

5.4.3.2 Process failures occurring during production (including deviation of safety-related special characteristics from their authorised range) and their potential effects on functional safety shall be analysed, the appropriate measures shall be taken and their ability to maintain functional safety shall be verified.

EXAMPLE Such measures can include performing further control measures, sorting, processing, and exchange of elements.

5.4.3.3 The capability of the following shall be assessed and maintained with regard to functional safety:

- a) production process;
- b) means of production; and
- c) tools and test equipment.

NOTE 1 The capability of the process can be proven by periodic process audits or by periodic qualification measures for each person performing the process steps.

NOTE 2 The capability of the process covers the ability to maintain the safety-related special characteristics.

5.4.3.4 The test equipment shall be subject to control of monitoring and measuring devices.

5.4.3.5 The controls shall be performed in accordance with the production control plan. The related control report shall include the following information: the control date, the identification of controlled object, and the control results.

NOTE 1 In the case of manual controls, the identification of the controlled object and control results are sufficient.

NOTE 2 The identification of the controlled object can be a vehicle identification number or a production number for a vehicle-level control measure or a part number or a serial number for a controlled component.

NOTE 3 The control results can consist of either a single status (e.g. pass or fail) or the evaluation of a collection of data against boundary limits.

5.4.3.6 Only approved configurations shall be produced, as defined in the release for production documentation, unless a deviation from the release for production documentation is authorized by the responsible person(s). The release for production documentation may be updated later in accordance with this authorized deviation.

5.4.3.7 Changes to the production process initiated during the production phase shall comply with Clause 5.

5.5 Work products

5.5.1 Safety-related content of the production plan resulting from requirements 5.4.1.1, 5.4.1.2, 5.4.1.3, 5.4.1.6 and 5.4.3.2.

5.5.2 Safety-related content of the production control plan including the test plan, resulting from requirements 5.4.1.4, 5.4.1.5, 5.4.3.4, and 5.4.3.6.

5.5.3 Control measures report resulting from requirement 5.4.3.5.

5.5.4 If applicable, specification of requirements on the producibility at system, hardware or software development level resulting from requirement 5.4.1.7.

NOTE This specification can be appended to the relevant documentation of the corresponding phases.

5.5.5 Assessment report for capability of the production process, resulting from requirement 5.4.2.2 and 5.4.3.3.

6 Operation, service (maintenance and repair), and decommissioning

6.1 Objectives

The objective of this clause is to specify the customer information, maintenance and repair instructions, as well as disassembly instructions regarding the item, system or element, in order to maintain the functional safety over the lifecycle of the vehicle.

6.2 General

This clause provides requirements for developing repair instructions and user information, including the user manual and the planning, execution and monitoring of the maintenance work, taking into account the safety-related special characteristics of the item.

During decommissioning, the phases “before disassembling”, “disassembling” and “after disassembling” can be distinguished. This clause addresses only those activities “before disassembling”.

6.3 Input to this clause

6.3.1 Prerequisites

The following information shall be available:

- requirements specification for production, operation, service and decommissioning in accordance with ISO 26262-5:2011, 7.5.4;
- release for production report in accordance with ISO 26262-4:2011, 11.5.1; and
- warning and degradation concept, included in the functional safety concept in accordance with ISO 26262-3:2011, 8.5.1.

6.3.2 Further supporting information

The following information can be considered:

- maintenance plan (from external source).

6.4 Requirements and recommendations

6.4.1 Planning of operation, service (maintenance and repair), and decommissioning

6.4.1.1 The operation, repair and maintenance processes shall be planned by evaluating the item and by considering the following:

- a) the requirements for maintenance and repair;
- b) the requirements for the information that shall be made available to the user to ensure the safe operation of the vehicle;
- c) the warning and degradation concept;
- d) the measures for field data collection and analysis;
- e) the conditions for storage, transport and handling of the hardware elements;
EXAMPLE Allowed storage time for the element.
- f) the approved configurations defined in the release for production documentation; and
EXAMPLE Allowed configurations of hardware, software and software calibration data during repair.
- g) the competence of the personnel involved.

6.4.1.2 The maintenance plan shall describe the sequence and methods of the maintenance steps or activities, the maintenance intervals, and the necessary means of maintenance and tools.

6.4.1.3 The maintenance plan and repair instructions shall describe the following:

- a) the work steps, procedures, diagnostic routines and methods;
- b) the maintenance tools and means;

EXAMPLE Programming, sensor calibration/setup and diagnostic equipment.

- c) the sequence and methods of the control steps and control criteria used to verify the safety-related special characteristics;
- d) the relevant item, systems or elements configurations, including the traceability measures;

NOTE This includes maintenance tool features used to ensure that the correct version of software is loaded into the vehicle, if such an operation is performed during maintenance.

EXAMPLE Labelling for the element is a way of ensuring traceability.

- e) the allowed deactivation of the item, systems or elements and necessary changes in the vehicle;
- f) the driver information for the allowed deactivations and changes; and

EXAMPLE Notifying the driver that an assistance function has been deactivated.

- g) the provision of replacement parts.

6.4.1.4 User information, including the user's manual, shall provide relevant usage instructions and warnings concerning the proper usage of the item, as well as the following information if applicable:

- a) a description of the relevant functions, (i.e. the intended usage, the status information or user interaction) and their operating modes;
- b) a description of the customer actions required to ensure controllability in the case of a failure indicated by the warning and degradation concept;
- c) a description of the maintenance activities expected from the customer in the case of a failure indicated by the warning and degradation concept;
- d) the warnings regarding known hazards resulting from interactions with third party products; and

EXAMPLE Park assist when using an additional third party tow hitch with a trailer. The user needs to be aware that the park assist can no longer scan behind the vehicle.

- e) the warnings regarding safety-related innovative functions of the item that could lead to driver's misunderstanding or misuse.

EXAMPLE A misuse of the automatic park brake when compared to manual park brake can lead to a driver leaving the vehicle without engaging the parking brake.

6.4.1.5 The decommissioning instructions shall describe the activities and measures to be applied before disassembly, and required to prevent the violation of a safety goal during disassembling, handling or decommissioning of the vehicle, the item or its elements.

EXAMPLE Instructions for the deactivation of airbags before the disassembly of the vehicle to avoid harm to the decommissioning personnel.

6.4.1.6 System, hardware or software level safety requirements arising during the planning of operation, service (maintenance and repair), and decommissioning, shall be specified and directed to the persons responsible for the development (see ISO 26262-4, ISO 26262-5 and ISO 26262-6).

EXAMPLE Specification of an error logging function in the ECU to ease diagnosis during service.

6.4.2 Operation, service (maintenance and repair), and decommissioning

6.4.2.1 The field monitoring process for functional safety incidents that relate to the item shall be implemented as planned in accordance with ISO 26262-2:2011, 7.4.2.4, in order to:

- a) provide the field data that shall be analysed to detect the presence of any functional safety issues and, if found, trigger actions that address those issues, and
- b) provide the evidence required by the proven in use argument if it is intended to use this argument in accordance with ISO 26262-8:2011, Clause 14.

6.4.2.2 The maintenance, repair and decommissioning of the item, its systems or its elements should be conducted and documented in accordance with the maintenance plan and the maintenance and repair instructions.

NOTE This includes the application of repair and maintenance procedures and the provision of either paper or electronic documentation of this application.

6.4.2.3 The supply of parts and their storage and transport shall be implemented as planned in accordance with 6.4.1.3.

6.4.2.4 If changes to the item for subsequent production are initiated by operation, field monitoring, maintenance, repair or decommissioning, a change management process in accordance with ISO 26262-8:2011, Clause 8, shall be complied with.

6.5 Work products

6.5.1 Safety-related content of the maintenance plan resulting from requirement 6.4.1.1, 6.4.1.2, 6.4.1.3.

6.5.2 Repair instructions resulting from requirement 6.4.1.3.

6.5.3 Safety-related content of the information made available to the user resulting from requirement 6.4.1.4.

6.5.4 Instructions regarding field observations resulting from requirement 6.4.2.1.

6.5.5 Safety-related content of the instructions for decommissioning resulting from requirement 6.4.1.5.

6.5.6 If applicable, specification of **requirements relating to operation, service and decommissioning at system, hardware or software development level** resulting from requirement 6.4.1.6.

NOTE This specification can be appended to the relevant documentation of the corresponding phases.

Annex A (informative)

Overview on and document flow of production and operation

Table A.1 provides an overview on objectives, prerequisites and work products of the particular phases of production and operation.

Table A.1 — Overview of production and operation

Clause	Objectives	Prerequisites	Work products
5 Production	<p>The first objective of this clause is to develop and maintain a production process for safety-related elements or items that are intended to be installed in road vehicles.</p> <p>The second objective is to achieve functional safety during the production process by the relevant manufacturer or the person or organisation responsible for the process (vehicle manufacturer, supplier, sub-supplier, etc.).</p>	<p>Requirements specification for production, operation, service and decommissioning in accordance with ISO 26262-5:2011, 7.5.4</p> <p>Specification of dedicated measures for hardware in accordance with ISO 26262-5:2011, 9.5.2</p> <p>Release for production report in accordance with ISO 26262-4:2011, 11.5.1</p>	<p>5.5.1 Safety-related content of the production plan resulting from requirements 5.4.1.1, 5.4.1.2, 5.4.1.3, 5.4.1.6 and 5.4.3.2.</p> <p>5.5.2 Safety-related content of the production control plan including the test plan, resulting from requirements 5.4.1.4, 5.4.1.5, 5.4.3.4, and 5.4.3.6.</p> <p>5.5.3 Control measures report resulting from requirement 5.4.3.5.</p> <p>5.5.4 If applicable, requirements specification on the producibility at system, hardware or software development level resulting from requirement 5.4.1.7 and appended to the relevant documentation of the corresponding phases.</p> <p>5.5.5 Assessment report for capability of the production process, resulting from requirement 5.4.2.2 and 5.4.3.3.</p>
6 Operation, service (maintenance and repair) and decommissioning	<p>The objective of this clause is to specify the customer information, maintenance and repair instructions, and disassembly instructions regarding the item, system or element in order to maintain the functional safety over the lifecycle of the vehicle.</p>	<p>Requirements specification for production, operation, service and decommissioning in accordance with ISO 26262-5:2011, 7.5.4</p> <p>Release for production report in accordance with ISO 26262-4:2011, 11.5.1</p> <p>Warning and degradation concept, included in the functional safety concept in accordance with ISO 26262-3:2011, 8.5.1</p>	<p>6.5.1 Safety-related content of the maintenance plan resulting from requirement 6.4.1.1, 6.4.1.2, 6.4.1.3.</p> <p>6.5.2 Repair instructions resulting from requirement 6.4.1.3.</p> <p>6.5.3 Safety-related content of the information made available to the user resulting from requirement 6.4.1.4.</p> <p>6.5.4 Instructions regarding field observations resulting from requirement 6.4.2.1.</p> <p>6.5.5 Safety-related content of the instructions for decommissioning resulting from requirement 6.4.1.5.</p> <p>6.5.6 If applicable, requirements specification relating to operation, service and decommissioning at system, hardware or software development level resulting from requirement 6.4.1.6.</p>

Bibliography

- [1] ISO/TS 16949, *Quality management systems — Particular requirements for the application of ISO 9001:2008 for automotive production and relevant service part organizations*
- [2] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*

This page is intentionally blank.

This page is intentionally blank.

ICS 43.040.10

Price based on 11 pages

INTERNATIONAL
STANDARD

ISO
26262-8

First edition
2011-11-15

**Road vehicles — Functional safety —
Part 8:
Supporting processes**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 8: Processus d'appui*



Reference number
ISO 26262-8:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	v
Introduction.....	vi
1 Scope.....	1
2 Normative references.....	2
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	3
4.3 ASIL-dependent requirements and recommendations	3
5 Interfaces within distributed developments	3
5.1 Objectives	3
5.2 General	3
5.3 Inputs to this clause.....	4
5.4 Requirements and recommendations	4
5.5 Work products	7
6 Specification and management of safety requirements.....	7
6.1 Objectives	7
6.2 General	7
6.3 Inputs to this clause.....	9
6.4 Requirements and recommendations	9
6.5 Work products	12
7 Configuration management.....	12
7.1 Objectives	12
7.2 General	12
7.3 Inputs to this clause.....	12
7.4 Requirements and recommendations	13
7.5 Work products	13
8 Change management.....	13
8.1 Objectives	13
8.2 General	13
8.3 Inputs to this clause.....	13
8.4 Requirements and recommendations	14
8.5 Work products	15
9 Verification	16
9.1 Objectives	16
9.2 General	16
9.3 Inputs to this clause.....	16
9.4 Requirements and recommendations	17
9.5 Work products	18
10 Documentation	19
10.1 Objectives	19
10.2 General	19
10.3 Inputs to this clause.....	19
10.4 Requirements and recommendations	19
10.5 Work products	20
11 Confidence in the use of software tools	20

11.1	Objectives	20
11.2	General.....	21
11.3	Inputs to this clause	21
11.4	Requirements and recommendations	22
11.5	Work products.....	27
12	Qualification of software components	27
12.1	Objectives.....	27
12.2	General.....	27
12.3	Inputs to this clause	28
12.4	Requirements and recommendations	28
12.5	Work products.....	30
13	Qualification of hardware components	30
13.1	Objectives.....	30
13.2	General.....	31
13.3	Inputs to this clause	32
13.4	Requirements and recommendations	33
13.5	Work products.....	35
14	Proven in use argument.....	35
14.1	Objectives.....	35
14.2	General.....	35
14.3	Inputs to this clause	36
14.4	Requirements and recommendations	37
14.5	Work products.....	40
Annex A (informative)	Overview on and document flow of supporting processes	41
Annex B (informative)	DIA example.....	43
Bibliography		48

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-8 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

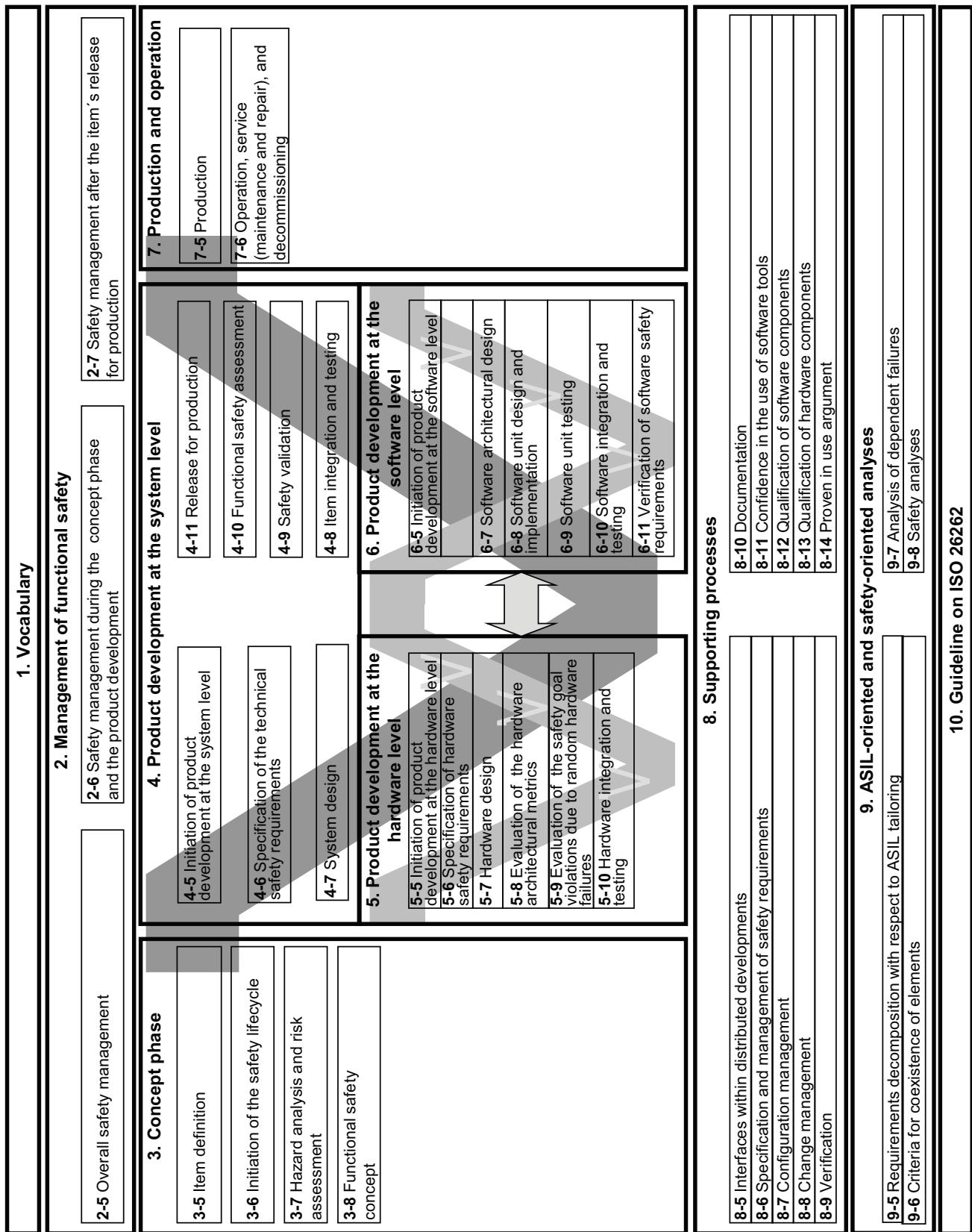


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 8: Supporting processes

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for supporting processes, including the following:

- interfaces within distributed developments,
- overall management of safety requirements,
- configuration management,
- change management,
- verification,
- documentation,
- confidence in the use of software tools,
- qualification of software components,
- qualification of hardware components, and
- proven in use argument.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-7:2011, *Road vehicles — Functional safety — Part 7: Production and operation*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

ISO/IEC 12207, *Systems and software engineering — Software life cycle processes*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- tailoring of the safety activities in accordance with ISO 26262-2 has been planned and shows that the requirement does not apply, or
- a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with ISO 26262-9:2011, Clause 5, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Interfaces within distributed developments

5.1 Objectives

The objective of this clause is to describe the procedures and to allocate associated responsibilities within distributed developments for items and elements.

5.2 General

The customer (e.g. vehicle manufacturer) and the suppliers for item developments jointly comply with the requirements specified in ISO 26262. Responsibilities are agreed between the customer and the suppliers. Subcontractor relationships are permitted. Just as with the customer's safety-related specifications concerning planning, execution and documentation for in-house item developments, comparable procedures are to be

agreed for co-operation with the supplier on distributed item developments, or item developments where the supplier has the full responsibility for safety.

NOTE This clause is not relevant for the procurement of standard components and parts or development commissions which do not place any responsibility for safety on the supplier.

5.3 Inputs to this clause

5.3.1 Prerequisites

See applicable prerequisites of the relevant phases of the safety lifecycle for which a distributed development is planned and carried out.

5.3.2 Further supporting information

The following information can be considered:

- the draft version of development interface agreement (DIA) (from external source);
- the supplier's tender based on a request for quotation (RFQ) (from external source).

5.4 Requirements and recommendations

5.4.1 Application of requirements

5.4.1.1 The requirements of Clause 5 shall apply to each item and element developed according to ISO 26262, except for off-the-shelf hardware parts, if either of the following applies:

- a) there are no specific hardware safety requirements allocated to the hardware parts, or
- b) the off-the-shelf hardware parts are qualified according to well-established procedures based on worldwide quality standards (e.g. AEC standards for electronic components), and the qualification of the off-the-shelf hardware parts covers ranges of parameters with regard to the intended application.

5.4.1.2 Requirements on the customer-supplier relationship (interfaces and interactions) shall apply to each level of the customer-supplier relationship.

NOTE 1 This includes subcontracts taken out by the top level supplier, subcontracts taken out by those subcontractors, etc.

NOTE 2 Internal suppliers can be managed in the same way as external suppliers.

5.4.2 Supplier selection criteria

5.4.2.1 The supplier selection criteria shall include an evaluation of the supplier's capability to develop and produce items and elements of comparable complexity and ASIL according to ISO 26262.

NOTE Supplier selection criteria includes:

- evidence of the supplier's quality management system;
- the supplier's past performance and quality;
- the confirmation of the supplier's capability concerning functional safety as part of the supplier's tender;
- results of previous safety assessments according to ISO 26262-2:2011, 6.4.9;
- recommendations from the development, production, quality and logistics departments of the vehicle manufacturer as far as they impact functional safety.

5.4.2.2 The RFQ from the customer to the supplier candidates shall include:

- a) a formal request to comply with ISO 26262,
- b) the item definition or functional specification of the element, and
- c) the safety goals, the functional safety requirements or the technical safety requirements, including their respective ASIL if already available, depending on what the supplier is quoting for.

NOTE If the ASIL is not known at the time of supplier selection, a conservative assumption is made.

5.4.3 Initiation and planning of distributed development

5.4.3.1 The customer and the supplier shall specify a DIA including the following:

NOTE An example of a DIA is given in Annex B.

- a) the appointment of the customer's and the supplier's safety managers,
- b) the joint tailoring of the safety lifecycle in accordance with ISO 26262-2:2011, 6.4.5,
- c) the activities and processes to be performed by the customer and the activities and processes to be performed by the supplier,
- d) the information and the work products to be exchanged,

NOTE 1 This includes an agreement on the documentation to be provided for the completion of the customer's and supplier's safety cases.

NOTE 2 The information exchanged includes the safety-related special characteristics.

NOTE 3 In the case of a distributed development, the relevant parts of the work products necessary for the activities of the development parties involved can be identified and exchanged.

- e) the parties or persons responsible for the activities,
- f) the communication of the target values, derived from the system level targets, to each relevant party in order for them to meet the target values for single-point faults metric and latent faults metric in accordance with the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures (see ISO 26262-5), and
- g) the supporting processes and tools, including interfaces, to assure compatibility between customer and supplier.

5.4.3.2 If the supplier conducts the hazard analysis and risk assessment, then the hazard analysis and risk assessment shall be provided to the customer for verification.

5.4.3.3 The party responsible for the item development shall create the functional safety concept in accordance with ISO 26262-3. The functional safety requirements shall be agreed between the customer and the supplier.

5.4.4 Execution of distributed development

5.4.4.1 The supplier shall report to the customer each issue which increases the risk of not conforming to the project plan, the safety plan, integration and testing plan in accordance with ISO 26262-4 or the software verification plan in accordance with ISO 26262-6, or other provisions of the DIA.

5.4.4.2 The supplier shall report to the customer each anomaly which occurs during the development activities in their area of responsibility or in that of their subcontractors.

5.4.4.3 The supplier shall determine whether each safety requirement can be complied with. If not, the safety concept shall be re-examined and, if necessary, modified to yield safety requirements that will be met.

5.4.4.4 Each change potentially affecting the safety of the item or the planned activities to demonstrate compliance with ISO 26262 shall be communicated to the other party to support the impact analysis in accordance with Clause 8.

5.4.4.5 Both parties should consider previous experience gained in similar developments in accordance with ISO 26262-2:2011, 5.4.2.7, when deriving safety requirements for the current development.

5.4.4.6 The supplier shall report to the customer's safety manager the progress achieved against the tasks and milestones defined in the safety plan. The format of the report and the delivery dates shall be agreed between the supplier and the customer.

EXAMPLE At regular intervals, or when the milestones specified in the framework of the schedule have been reached, the customer inspects the released quality management reports compiled by the supplier.

5.4.4.7 An agreement shall be reached on which party (supplier or customer) shall perform the safety validation in accordance with ISO 26262-4.

NOTE If the supplier performs the integration and validation, an agreement on the capabilities and resources needed by the supplier is important since safety validation requires the integrated vehicle (see ISO 26262-4).

5.4.4.8 This requirement applies to ASIL D in accordance with 4.3. The customer shall be allowed to perform additional functional safety audits at the supplier's premises at any appropriate time.

5.4.5 Functional safety assessment at supplier's premises

5.4.5.1 This requirement applies to ASILs (B), C, D in accordance with 4.3. One or more functional safety assessments shall be carried out upon reaching defined milestones, these assessments shall include each phase of the item development. The functional safety assessments shall be at the level of detail appropriate for the complexity of the item and the ASILs of its safety goals. The functional safety assessment shall be performed in accordance with ISO 26262-2:2011, 6.4.9.

5.4.5.2 This requirement applies to ASIL B in accordance with 4.3. A functional safety assessment should be carried out.

NOTE This can be done by the customer, another organization or by the supplier itself.

5.4.5.3 This requirement applies to ASILs C and D in accordance with 4.3. A functional safety assessment in accordance with ISO 26262-2:2011, 6.4.9, shall be carried out at the supplier's premises by the customer, or by an organization or person designated by the customer.

NOTE This can be done by the supplier itself.

5.4.5.4 This requirement applies to ASILs (B), C and D in accordance with 4.3. The functional safety assessment report shall be available at the customer's and at the supplier's premises.

5.4.5.5 This requirement applies to ASILs (B), C and D in accordance with 4.3. Each anomaly identified, that potentially impacts the deliverables from the supplier, shall be analyzed and actions shall be derived to resolve them. An agreement between both parties shall be reached on who performs the actions required.

5.4.6 After release for production

5.4.6.1 The supplier shall provide evidence to the customer that the process capability is being met and maintained in accordance with ISO 26262-2:2011, Clause 7, and ISO 26262-7:2011, Clause 5.

5.4.6.2 A supply agreement between the customer and the supplier shall address the responsibilities for functional safety in accordance with ISO 26262-2:2011, 7.4.2.1, and define the safety activities for each party.

5.4.6.3 The supply agreement shall state the access to, and exchange of, production monitoring records between the parties for the safety-related special characteristics.

5.4.6.4 Each party that becomes aware of a safety-related event shall report this in a timely manner and according to the supply agreement. If a safety-related event occurs, an analysis of that event shall be performed. This analysis should include similar items and related parties which are potentially affected by a similar event.

5.5 Work products

5.5.1 Supplier selection report resulting from requirements 5.4.2.1 and 5.4.2.2.

5.5.2 Development interface agreement (DIA) resulting from requirement 5.4.3.

5.5.3 Supplier's project plan resulting from requirement 5.4.3.

5.5.4 Supplier's safety plan resulting from requirement 5.4.3.

5.5.5 Functional safety assessment report resulting from requirements 5.4.5.1 to 5.4.5.5.

5.5.6 Supply agreement resulting from requirements 5.4.6.2 to 5.4.6.3.

6 Specification and management of safety requirements

6.1 Objectives

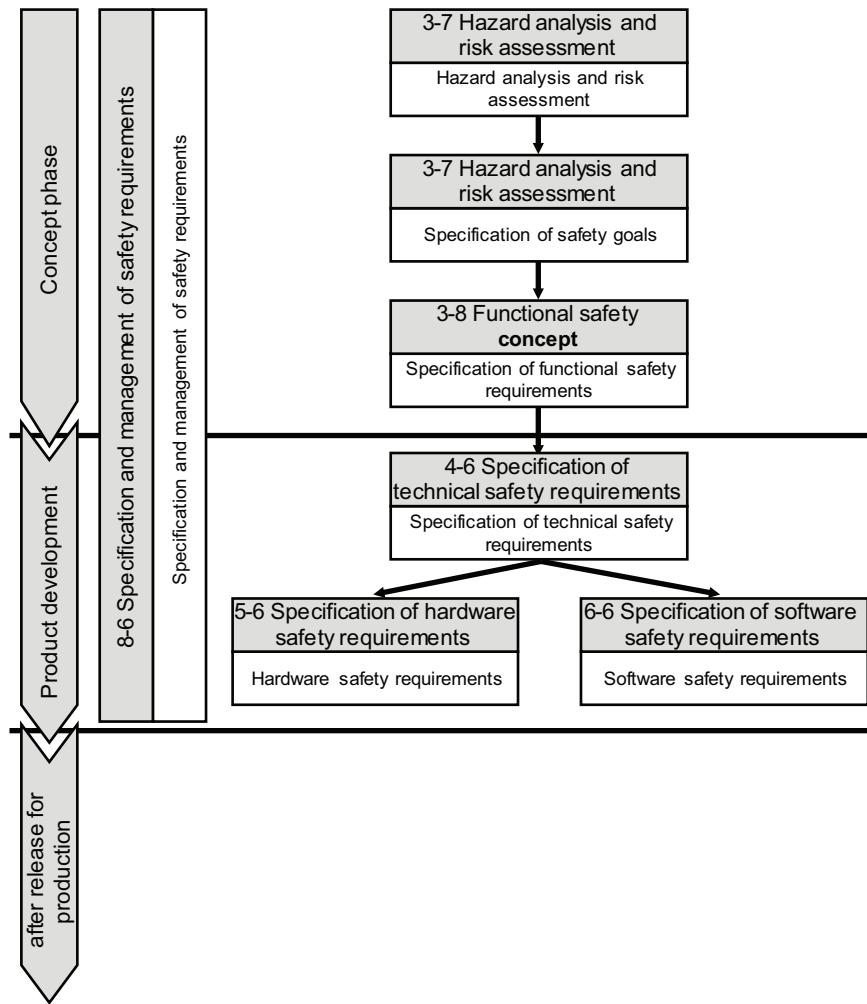
The first objective is to ensure the correct specification of safety requirements with respect to their attributes and characteristics.

The second objective is to ensure consistent management of safety requirements throughout the entire safety lifecycle.

6.2 General

Safety requirements constitute all requirements aimed at achieving and ensuring the required ASILs.

During the safety lifecycle, safety requirements are specified and detailed in a hierarchical structure. The structure and dependencies of safety requirements used in ISO 26262 are illustrated in Figure 2. The safety requirements are allocated or distributed among the elements.



NOTE Within the figure, the specific clauses of each part of ISO 26262 are indicated in the following manner: "m-n", where "m" represents the number of the part and "n" indicates the number of the clause, e.g. "3-7" represents Clause 7 of ISO 26262-3.

Figure 2 — Structure of safety requirements

The management of safety requirements includes managing requirements, obtaining agreement on the requirements, obtaining commitments from those implementing the requirements, and maintaining traceability.

In order to support the management of safety requirements, the use of suitable requirements management tools is recommended.

This clause includes requirements on the specification and management of safety requirements (see Figure 3).

The specific requirements concerning the content of the safety requirements at different hierarchical levels are listed in ISO 26262-3, ISO 26262-4, ISO 26262-5 and ISO 26262-6.

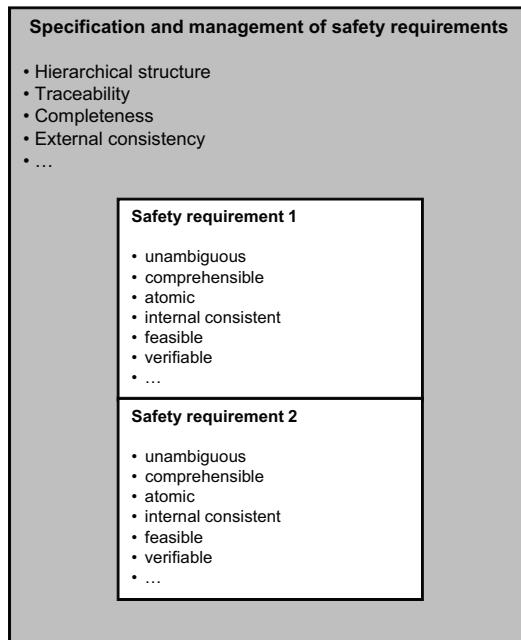


Figure 3 — Relationship between management of safety requirements and particular safety requirements

6.3 Inputs to this clause

6.3.1 Prerequisites

See applicable prerequisites of the relevant phases of the safety lifecycle in which safety requirements are specified or managed.

6.3.2 Further supporting information

See applicable further supporting information of the relevant phases of the safety lifecycle in which safety requirements are specified or managed.

6.4 Requirements and recommendations

6.4.1 Specification of safety requirements

6.4.1.1 To achieve the characteristics of safety requirements listed in 6.4.2.4, safety requirements shall be specified by an appropriate combination of:

- a) natural language, and
- b) methods listed in Table 1.

NOTE For higher level safety requirements (e.g. functional and technical safety requirements) natural language is more appropriate while for lower level safety requirements (e.g. software and hardware safety requirements) notations listed in Table 1 are more appropriate.

Table 1 — Specifying safety requirements

	Methods	ASIL			
		A	B	C	D
1a	Informal notations for requirements specification	++	++	+	+
1b	Semi-formal notations for requirements specification	+	+	++	++
1c	Formal notations for requirements specification	+	+	+	+

6.4.2 Attributes and characteristics of safety requirements

6.4.2.1 Safety requirements shall be unambiguously identifiable as safety requirements.

NOTE In order to comply with this requirement, safety requirements can be listed in a separate document. If safety requirements and other requirements are administered in the same document, safety requirements can be identified explicitly by using a special attribute as described in 6.4.2.5.

6.4.2.2 Safety requirements shall inherit the ASIL from the safety requirements from which they are derived, except if ASIL decomposition is applied in accordance with ISO 26262-9.

NOTE As safety goals are the top level safety requirements, the inheritance of ASILs starts at the safety goal level (see ISO 26262-1:2011, definition 1.108).

6.4.2.3 Safety requirements shall be allocated to an item or an element.

6.4.2.4 Safety requirements shall have the following characteristics:

- a) unambiguous and comprehensible,

NOTE 1 A requirement is unambiguous if there is common understanding of the meaning of the requirement.

NOTE 2 A requirement is comprehensible if the reader at an adjacent abstraction level (i.e. either the stakeholder or the consumer of that requirement) understands its meaning.

- b) atomic,

NOTE Safety requirements at one hierarchical level are atomic when they are formulated in such a way that they can not be divided into more than one safety requirement at the considered level.

- c) internally consistent,

NOTE Unlike external consistency, in which multiple safety requirements do not contradict each other, internal consistency means that each individual safety requirement contains no contradictions within itself.

- d) feasible, and

NOTE A requirement is feasible if it can be implemented within the constraints of the item development (resources, state-of-the-art, etc.).

- e) verifiable.

6.4.2.5 Safety requirements shall have the following attributes:

- a) a unique identification remaining unchanged throughout the safety lifecycle,

EXAMPLE A unique identification of a requirement can be achieved in a variety of ways, such as subscripting each instance of the word "shall", e.g. "The system shall₉₇₈₂ check ...", or numbering consecutively each sentence containing the word "shall", e.g. "In the case of ... the system shall check ...".

- b) a status, and

EXAMPLE A status of a safety requirement can be “proposed”, “assumed”, “accepted”, or “reviewed”.

- c) an ASIL.

6.4.3 Management of safety requirements

6.4.3.1 The set of safety requirements shall have the following properties:

- a) hierarchical structure,

NOTE Hierarchical structure means that safety requirements are structured in several successive levels as presented in Figure 2. These levels are always aligned to comply with the corresponding design phases.

- b) organizational structure according to an appropriate grouping scheme,

NOTE Organization of safety requirements means that safety requirements within each level are grouped together, usually corresponding to the architecture.

- c) completeness,

NOTE Completeness means that the safety requirements at one level fully implement all safety requirements of the previous level.

- d) external consistency,

NOTE Unlike internal consistency, in which an individual safety requirement does not contradict itself, external consistency means that multiple safety requirements do not contradict each other.

- e) no duplication of information within any level of the hierarchical structure, and

NOTE No duplication of information means that the content of safety requirements is not repeated in any other safety requirement at one single level of the hierarchical structure and this is true at each hierarchical level.

- f) maintainability.

NOTE Maintainability means that the set of requirements can be modified or extended, e.g. by the introduction of new versions of requirements or by adding/removing requirements to the set of requirements.

6.4.3.2 Safety requirements shall be traceable with a reference being made to:

- a) each source of a safety requirement at the upper hierarchical level,
- b) each derived safety requirement at a lower hierarchical level, or to its realisation in the design, and
- c) the specification of verification in accordance with 9.4.2.

NOTE Additionally, traceability supports:

- an impact analysis if changes are made to particular safety requirements, and
- the functional safety assessment.

6.4.3.3 An appropriate combination of the verification methods listed in Table 2 shall be applied to verify that the safety requirements comply with the requirements in this clause and that they comply with the specific requirements on the verification of safety requirements within the respective parts of ISO 26262 where safety requirements are derived.

Table 2 — Methods for the verification of safety requirements

	Methods	ASIL			
		A	B	C	D
1a	Verification by walk-through	++	+	o	o
1b	Verification by inspection	+	++	++	++
1c	Semi-formal verification ^a	+	+	++	++
1d	Formal verification	o	+	+	+

^a Method 1c can be supported by executable models.

6.4.3.4 Safety requirements shall be placed under configuration management in accordance with Clause 7.

EXAMPLE When the safety requirements at a lower level are consistent with the higher level safety requirements, the configuration management can define a baseline as the basis for subsequent phases of the safety lifecycle.

6.5 Work products

None.

7 Configuration management

7.1 Objectives

The first objective is to ensure that the work products, and the principles and general conditions of their creation, can be uniquely identified and reproduced in a controlled manner at any time.

The second objective is to ensure that the relations and differences between earlier and current versions can be traced.

7.2 General

Configuration management is a well established practice within the automotive industry and can be applied according to ISO/TS 16949, ISO 10007 and ISO/IEC 12207.

Each work product of ISO 26262 is managed by configuration management.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- Safety plan in accordance with ISO 26262-2:2011, 6.5.1.
- Applicable prerequisites of the relevant phases of the safety lifecycle where configuration management is planned or managed.

7.3.2 Further supporting information

None.

7.4 Requirements and recommendations

7.4.1 Configuration management shall be planned.

7.4.2 The configuration management process shall comply with:

- a) the respective requirements of a quality management system (e.g. ISO/TS 16949, or ISO 9001), and
- b) the specific requirements for software development according to the clause on configuration management in ISO/IEC 12207.

7.4.3 The work products required by the safety plan in accordance with ISO 26262-2 shall be placed under configuration management and baselined according to the configuration management strategy.

7.4.4 Work products placed under configuration management shall be documented in the configuration management plan.

7.4.5 Configuration management shall be maintained throughout the entire safety lifecycle.

7.5 Work products

7.5.1 Configuration management plan resulting from requirements in 7.4.1, 7.4.2 and 7.4.5.

8 Change management

8.1 Objectives

The objective of change management is to analyse and control changes to safety-related work products throughout the safety lifecycle.

8.2 General

Change management ensures the systematic planning, control, monitoring, implementation and documentation of changes, while maintaining the consistency of each work product. Potential impacts on functional safety are assessed before changes are made. For this purpose, decision-making processes for change are introduced and established, and responsibilities are assigned to the parties involved.

NOTE Here change is understood as modification due to: anomalies, removals, additions, enhancements, obsolescence of components or parts, etc.

8.3 Inputs to this clause

8.3.1 Prerequisites

The following information shall be available:

- configuration management plan in accordance with 7.5.1
- safety plan in accordance with ISO 26262-2:2011, 6.5.2.

8.3.2 Further supporting information

None.

8.4 Requirements and recommendations

8.4.1 Planning and initiating change management

8.4.1.1 The change management process shall be planned and initiated, before changes are made to work products.

NOTE Configuration management and change management are initiated at the same time. Interfaces between the two processes are defined and maintained to enable the traceability of changes.

8.4.1.2 The work products to be subject to change management shall be identified and shall include those work products required by ISO 26262 to be placed under configuration management.

8.4.1.3 The schedule for applying the change management process shall be defined for each work product.

8.4.1.4 The change management process shall include:

- a) the change requests in accordance with 8.4.2,
- b) the analysis of change requests in accordance with 8.4.3,
- c) the decision and rationale regarding change requests in accordance with 8.4.4,
- d) the implementation of the accepted changes in accordance with 8.4.5, and
- e) the documentation in accordance with 8.4.5.

8.4.2 Change requests

8.4.2.1 A unique identifier shall be assigned to each change request.

8.4.2.2 As a minimum, every change request shall include the following information:

- a) the date,
- b) the reason for the requested change,
- c) the exact description of the requested change, and
- d) the configuration on which the requested change is based.

8.4.3 Change request analysis

8.4.3.1 An impact analysis on the item involved, its interfaces and connected items, shall be carried out for each change request. The following shall be addressed:

- a) the type of change request,

NOTE Possible types of changes include: error resolution, adaptation, enhancement, prevention.

- b) the identification of the work products to be changed and the work products affected,
- c) the identification and involvement of the parties affected, in the case of a distributed development,
- d) the potential impact of the change on functional safety, and
- e) the schedule for the realisation and verification of the change.

8.4.3.2 Each change to work products shall initiate the return to the applicable phase of the safety lifecycle. Subsequent phases shall be carried out in compliance with ISO 26262.

8.4.4 Change request evaluation

8.4.4.1 The change request shall be evaluated using the results of the impact analysis in compliance with 8.4.3.1 and a decision regarding acceptance, rejection or delay shall be made by the authorized persons.

EXAMPLE Typically, the authorised persons include:

- project manager,
- safety manager,
- person in charge of quality assurance, and
- developers involved.

NOTE The accepted change requests can be prioritised and combined with related accepted change requests.

8.4.4.2 For each accepted change request it shall be decided who shall carry out the change and when the change is due. This decision shall consider the interfaces involved in carrying out the change request.

8.4.5 Carrying out and documenting the change

8.4.5.1 The changes shall be carried out and verified as planned.

8.4.5.2 If the change has an impact on safety-related functions, the assessment of functional safety and the applicable confirmation reviews, in accordance with ISO 26262-2:2011, 6.4.7 and 6.4.9, shall be updated before releasing the item.

8.4.5.3 The documentation of the change shall contain the following information:

- a) the list of changed work products at an appropriate level including configurations and versions, in accordance with Clause 7 (Configuration management),
- b) the details of the change carried out, and
- c) the planned date for the deployment of the change.

NOTE In the case of a rejected change request, the change request and the rationale for the rejection are also documented.

8.5 Work products

8.5.1 **Change management plan** resulting from requirements 8.4.1.1 to 8.4.1.3.

8.5.2 **Change request** resulting from requirements 8.4.2.

8.5.3 **Impact analysis** and **change request plan** resulting from requirements 8.4.3.1, 8.4.4.1 and 8.4.4.2.

8.5.4 **Change report** resulting from requirement 8.4.5.3.

9 Verification

9.1 Objectives

The objective of verification is to ensure that the work products comply with their requirements.

9.2 General

Verification is applicable to the following phases of the safety lifecycle.

- a) In the concept phase, verification ensures that the concept is correct, complete and consistent with respect to the boundary conditions of the item, and that the defined boundary conditions themselves are correct, complete and consistent, so that the concept can be realised.
- b) In the product development phase, verification is conducted in different forms, as described below.
 - 1) In the design phases, verification is the evaluation of the work products, such as requirement specification, architectural design, models, or software code, thus ensuring that they comply with previously established requirements for correctness, completeness and consistency. Evaluation can be performed by review, simulation or analysis techniques. The evaluation is planned, specified, executed and documented in a systematic manner.

NOTE Design phases are ISO 26262-4:2011, Clause 7 (System design), ISO 26262-5:2011, Clause 7 (Hardware design), ISO 26262-6:2011, Clause 7 (Software architectural design) and ISO 26262-6:2011, Clause 8 (Software unit design and implementation).

- 2) In the test phases, verification is the evaluation of the work products within a test environment to ensure that they comply with their requirements. The tests are planned, specified, executed, evaluated and documented in a systematic manner.
- c) In the production and operation phase, verification ensures that:
 - 1) the safety requirements are appropriately realised in the production process, user manuals and repair and maintenance instructions; and
 - 2) the safety-related properties of the item are met by the application of control measures within the production process.

NOTE This is a generic verification process that is instantiated by phases of the safety lifecycle in ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7. Safety validation is not addressed by this process. See ISO 26262-4:2011, Clause 9 (Safety validation), for further details.

9.3 Inputs to this clause

9.3.1 Prerequisites

See applicable prerequisites of the relevant phases of the safety lifecycle in which verification is planned or carried out.

9.3.2 Further supporting information

See applicable further supporting information of the relevant phases of the safety lifecycle in which verification is planned or carried out.

9.4 Requirements and recommendations

9.4.1 Verification planning

9.4.1.1 The verification planning shall be carried out for each phase and subphase of the safety lifecycle and shall address the following:

- a) the content of the work products to be verified,
- b) the methods used for verification,

NOTE Methods for verification include review, walk-through, inspection, model-checking, simulation, engineering analyses, demonstration, and testing. Typically verification applies a combination of these and other methods.

- c) the pass and fail criteria for the verification,
- d) the verification environment, if applicable,

NOTE A verification environment can be a test or simulation environment.

- e) the tools used for verification, if applicable,
- f) the actions to be taken if anomalies are detected, and
- g) the regression strategy.

NOTE A regression strategy specifies how verification is repeated after changes have been made to the item or element. Verification can be repeated fully or partially and can include other items or elements that might affect the results of the verification.

9.4.1.2 The planning of verification should consider the following:

- a) the adequacy of the verification methods to be applied,
- b) the complexity of the work product to be verified,
- c) prior experiences related to the verification of the subject material, and

NOTE This includes service history as well as the degree to which a proven in use argument has been achieved.

- d) the degree of maturity of the technologies used, or the risks associated with the use of these technologies.

9.4.2 Verification specification

9.4.2.1 The verification specification shall select and specify the methods to be used for the verification, and shall include:

- a) review or analysis checklists; or
- b) simulation scenarios; or
- c) test cases, test data and test objects.

9.4.2.2 For testing, the specification of each test case shall include the following:

- a) a unique identification,
- b) the reference to the version of the associated work product to be verified,

- c) the preconditions and configurations,

NOTE If a complete verification of the possible configurations of a work product (e.g. variants of a system) is not feasible, a reasonable subset is selected (e.g. minimum or maximum functionality configurations of a system).

- d) the environmental conditions, if appropriate,

NOTE Environmental conditions relate to the physical properties (e.g. temperature) of the surroundings in which the test is conducted or is simulated as part of the test.

- e) the input data, their time sequence and their values, and

- f) the expected behaviour which includes output data, acceptable ranges of output values, time behaviour and tolerance behaviour.

NOTE 1 When specifying the expected behaviour, it might be necessary to specify the initial output data in order to detect changes.

NOTE 2 To avoid the redundant specification and storage of preconditions, configurations and environmental conditions used for various test cases, the use of an unambiguous reference to such data is recommended.

9.4.2.3 For testing, test cases shall be grouped according to the test methods to be applied. For each test method, in addition to the test cases, the following shall be specified:

- a) the test environment,
- b) the logical and temporal dependencies, and
- c) the resources.

9.4.3 Verification execution and evaluation

9.4.3.1 The verification shall be executed as planned in accordance with 9.4.1 and specified in accordance with 9.4.2.

9.4.3.2 The evaluation of the verification results shall contain the following information:

- a) the unique identification of the verified work product,
- b) the reference to the verification plan and verification specification,
- c) the configuration of the verification environment and verification tools used, and the calibration data used during the evaluation, if applicable,
- d) the level of compliance of the verification results with the expected results,
- e) an unambiguous statement of whether the verification passed or failed; if the verification failed the statement shall include the rationale for failure and suggestions for changes in the verified work product, and

NOTE The verification is evaluated according to the criteria for completion and termination of the verification [see 9.4.1.1 c)] and to the expected verification results.

- f) the reasons for any verification steps not executed.

9.5 Work products

9.5.1 Verification plan resulting from requirements 9.4.1.1 and 9.4.1.2.

9.5.2 Verification specification resulting from requirements 9.4.2.1 to 9.4.2.3.

9.5.3 Verification report resulting from requirements 9.4.3.1 and 9.4.3.2.

10 Documentation

10.1 Objectives

The primary objective is to develop a documentation management strategy for the entire safety lifecycle in order to facilitate an effective and repeatable documentation management process.

10.2 General

The documentation requirements in ISO 26262 focus mainly on information, and not on layout and appearance.

The information need not be made available in physical documents, unless explicitly specified by ISO 26262. The documentation can take various forms and structures and tools can be used to generate documents automatically.

EXAMPLE Possible forms are: paper, electronic media, databases.

What is deemed adequate information depends on a variety of factors, including the complexity, the extent of the safety-related systems/subsystems, and the requirements relating to the special application.

Duplication of information within a document, and between documents, should be avoided to aid maintainability.

NOTE An alternative to duplicating information is the use of a cross-reference within one document, directing the reader to the information source document.

10.3 Inputs to this clause

10.3.1 Prerequisites

The following information shall be available:

- safety plan in accordance with ISO 26262-2:2011, 6.5.1

10.3.2 Further supporting information

None.

10.4 Requirements and recommendations

10.4.1 The documentation process shall be planned in order to make documentation available:

- a) during each phase of the entire safety lifecycle for the effective completion of the phases and verification activities,
- b) for the management of functional safety, and
- c) as an input to the functional safety assessment.

10.4.2 The identification of a work product in ISO 26262 shall be interpreted as a requirement for documentation containing the information concerning the results of the associated requirements.

NOTE The documentation can be in the form of a single document containing the complete information for the work product or a set of documents that together contain the complete information for the work product.

10.4.3 The documents should be:

- a) precise and concise,
- b) structured in a clear manner,
- c) easy to understand by the intended users, and
- d) maintainable.

10.4.4 The structure of the entire documentation should consider in-house procedures and working practices. It shall be organized to facilitate the search for relevant information.

EXAMPLE Documentation tree.

10.4.5 Each work product or document shall be associated with the following formal elements:

- a) the title, referring to the scope of the content,
- b) the author and approver,
- c) unique identification of each different revision (version) of a document,
- d) the change history, and

NOTE The change history contains, per change, the name of the author, the date and a brief description.

- e) the status.

EXAMPLE "Draft", "released".

10.4.6 It shall be possible to identify the current applicable revision (version) of a document or item of information, in accordance with Clause 7.

10.5 Work products

10.5.1 Documentation management plan resulting from requirement 10.4.1.

10.5.2 Documentation guideline requirements resulting from requirements 10.4.3 to 10.4.6.

11 Confidence in the use of software tools

11.1 Objectives

The first objective of this clause is to provide criteria to determine the required level of confidence in a software tool when applicable.

The second objective of this clause is to provide means for the qualification of the software tool when applicable, in order to create evidence that the software tool is suitable to be used to tailor the activities or tasks required by ISO 26262 (i.e. the user can rely on the correct functioning of a software tool for those activities or tasks required by ISO 26262).

11.2 General

A software tool used in the development of a system or its software or hardware elements, can support or enable a tailoring of the safety-lifecycle, through the tailoring of activities and tasks required by ISO 26262. In such cases confidence is needed that the software tool effectively achieves the following goals:

- a) the risk of systematic faults in the developed product due to malfunctions of the software tool leading to erroneous outputs is minimized, and
- b) the development process is adequate with respect to compliance with ISO 26262, if activities or tasks required by ISO 26262 rely on the correct functioning of the software tool used.

NOTE The understanding of “software tool” can vary from a separately used stand-alone software tool to a set of software tools integrated into a tool-chain.

EXAMPLE Such software tools can be commercial tools, open source tools, freeware tools, shareware tools or tools developed in-house by the user.

To determine the required level of confidence in a software tool used within development under the conditions mentioned above, the following criteria are evaluated:

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

To evaluate the confidence in prevention or detection measures, measures internal to the software tool (e.g. monitoring) as well as measures external to the software tool (e.g. guidelines, tests, reviews) implemented in the development process of the safety-related item or element are considered and can be assessed.

If indicated by the determined tool confidence level, then appropriate qualification methods are applied to comply with both this tool confidence level and the maximum ASIL of all the safety requirements allocated to the item or element that is to be developed using the software tool. Otherwise there is no need to apply such qualification methods.

11.3 Inputs to this clause

11.3.1 Prerequisites

The following information shall be available:

- safety plan in accordance with ISO 26262-4:2011, 5.5.2;
- applicable prerequisites of the phases of the safety lifecycle where a software tool is used.

11.3.2 Further supporting information

The following information can be considered:

- pre-determined maximum ASIL;
- user manual for the software tool (from external source);
- environment and constraints of the software tool (from external source).

11.4 Requirements and recommendations

11.4.1 General requirement

11.4.1.1 If the safety lifecycle incorporates the use of a software tool for the development of a system, or its hardware or software elements, such that activities or tasks required by ISO 26262 rely on the correct functioning of a software tool, and where the relevant outputs of that tool are not examined or verified for the applicable process step(s), such software tools shall comply with the requirements of this clause.

11.4.2 Validity of predetermined tool confidence level or qualification

11.4.2.1 If the confidence level evaluation or qualification of a software tool is performed independently from the development of a particular safety-related item or element, the validity of this predetermined tool confidence level or qualification shall be confirmed, in accordance with ISO 26262-2:2011, Table 1, prior to the software tool being used for the development of a particular safety-related item or element.

NOTE The collection of information about the software tools can be a cross-organizational activity, thus facilitating the classification or qualification effort.

11.4.3 Software tool compliance with its evaluation criteria or its qualification

11.4.3.1 When using a software tool, it shall be ensured that its usage, its determined environmental and functional constraints and its general operating conditions comply with its evaluation criteria or its qualification.

EXAMPLE Use of identical version and configuration settings for the same use cases together with the same implemented measures for the prevention or detection of malfunctions and their corresponding erroneous output, as documented in the qualification report for this software tool.

11.4.4 Planning of usage of a software tool

11.4.4.1 The usage of a software tool shall be planned, including the determination of:

- a) the identification and version number of the software tool,
- b) the configuration of the software tool,

EXAMPLE The configuration of a compiler is defined by setting compiler switches and “#pragma” statements in a C source file.

- c) the use cases of the software tool,

NOTE 1 Use cases can describe the user's interactions with a software tool or an applied subset of the software tool's functionality.

NOTE 2 Use cases can include requirements for the tool's configuration and the environment in which the software tool is executed.

- d) the environment in which the software tool is executed,
- e) the maximum ASIL of all the safety requirements, allocated to the item or the element that can be violated, if the software tool is malfunctioning and producing corresponding erroneous output, and

NOTE The maximum ASIL can be determined with regard to a specific development, or an assumption can be made with regard to the generic usage of the software tool. In the case of an assumed pre-determined ASIL, such an assumption is verified.

- f) the methods to qualify the software tool, if required based on the determined level of confidence.

11.4.4.2 To ensure the proper evaluation or usage of the software tool, the following information shall be available:

- a) a description of the features, functions and technical properties of the software tool,
- b) the user manual or other usage guides, if applicable,
- c) a description of the environment required for its operation,
- d) a description of the expected behaviour of the software tool under anomalous operating conditions, if applicable,

EXAMPLE 1 Anomalous operating conditions can be prohibited combinations of compiler switches, an environment not complying with the user manual or an incorrect installation.

EXAMPLE 2 Expected behaviour under an anomalous operating condition can be a suppression of output generation, a user indication or a user report.

- e) a description of known software tool malfunctions and the appropriate safeguards, avoidance or work-around measures, if applicable, and

EXAMPLE 1 Usage guidelines or workarounds addressing known malfunctions, limitation of code optimisation by compilers or the use of a limited set of building blocks for modelling.

EXAMPLE 2 Safeguards include prevention through usage constraints, detection, reporting of all known malfunctions and issues, and provision of safe alternate techniques to perform the corresponding activity.

- f) the measures for the detection of malfunctions and the corresponding erroneous output of the software tool identified during the determination of the required level of confidence for this software tool.

NOTE Measures for the detection of erroneous corresponding outputs can address both known and potential errors in the output of software tools.

EXAMPLE Comparisons of outputs of redundant software tools, tests performed, static analyses or reviews, analyses of log files of the software tool.

11.4.5 Evaluation of a software tool by analysis

11.4.5.1 The description of the usage of a software tool shall contain the following information:

- a) the intended purpose,

EXAMPLE Simulation of a function, the generation of source code, or the test of embedded software, the tailoring of the safety-lifecycle or the simplification or automation of activities and tasks required by ISO 26262.

- b) the inputs and expected outputs, and

EXAMPLE Data required as input for a subsequent development activity, source code, results of a simulation, results of a test, or other work products of ISO 26262.

- c) the environmental and functional constraints, if applicable.

EXAMPLE Embedding the software tool into the development processes, the usage of shared data by different software tools and other usage conditions, measures to prevent or detect malfunctions placed around the software tool.

11.4.5.2 The intended usage of the software tool shall be analysed and evaluated to determine:

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

- 1) TI1 shall be selected when there is an argument that there is no such possibility;
- 2) TI2 shall be selected in all other cases;
- b) the confidence in measures that prevent the software tool from malfunctioning and producing corresponding erroneous output, or in measures that detect that the software tool has malfunctioned and has produced corresponding erroneous output. This is expressed by the classes of Tool error Detection (TD):
- 1) TD1 shall be selected if there is a high degree of confidence that a malfunction and its corresponding erroneous output will be prevented or detected;
 - 2) TD2 shall be selected if there is a medium degree of confidence that a malfunction and its corresponding erroneous output will be prevented or detected;
 - 3) TD3 shall be selected in all other cases.

NOTE 1 Prevention or detection can be accomplished through process steps, redundancy in tasks or software tools or by rationality checks within the software tool itself.

NOTE 2 TD3 typically applies if there are no systematic measures in the development process available, and therefore malfunctions of the software tool and their corresponding erroneous outputs can only be detected randomly.

NOTE 3 If a software tool is used to verify the output from another software tool, the interdependency between those software tools is considered when evaluating the subsequent software tool and an adequate TD is selected for this subsequently-used software tool.

NOTE 4 The level of detail for such a usage analysis only needs to permit the proper determination of both of the classes of TI and TD.

EXAMPLE 1 TD1 can be chosen for a code generator in case the produced source code is verified in accordance with ISO 26262.

EXAMPLE 2 Usage guidelines can prevent malfunctions such as the incorrect or ambiguous interpretation of code constructs by a compiler.

11.4.5.3 If the correct selection of TI or TD is unclear or doubtful, TI and TD should be estimated conservatively.

11.4.5.4 If a software tool is used for the tailoring of the development process in such a way that activities or tasks required by ISO 26262 are omitted, TD2 shall not be selected.

11.4.5.5 Based on the values determined for the classes of TI and TD (in accordance with 11.4.5.2, 11.4.5.3 or 11.4.5.4), the required software tool confidence level shall be determined according to Table 3.

Table 3 — Determination of the tool confidence level (TCL)

		Tool error detection		
		TD1	TD2	TD3
Tool impact	TI1	TCL1	TCL1	TCL1
	TI2	TCL1	TCL2	TCL3

11.4.6 Qualification of a software tool

11.4.6.1 For the qualification of software tools classified at TCL3, the methods listed in Table 4 shall be applied. For the qualification of software tools classified at TCL2, the methods listed in Table 5 shall be applied. A software tool classified at TCL1 needs no qualification methods.

Table 4 — Qualification of software tools classified TCL3

	Methods	ASIL			
		A	B	C	D
1a	Increased confidence from use in accordance with 11.4.7	++	++	+	+
1b	Evaluation of the tool development process in accordance with 11.4.8	++	++	+	+
1c	Validation of the software tool in accordance with 11.4.9	+	+	++	++
1d	Development in accordance with a safety standard ^a	+	+	++	++

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

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

Table 5 — Qualification of software tools classified TCL2

	Methods	ASIL			
		A	B	C	D
1a	Increased confidence from use in accordance with 11.4.7	++	++	++	+
1b	Evaluation of the tool development process in accordance with 11.4.8	++	++	++	+
1c	Validation of the software tool in accordance with 11.4.9	+	+	+	++
1d	Development in accordance with a safety standard ^a	+	+	+	++

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

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

11.4.6.2 The qualification of the software tool shall be documented including the following:

- a) the unique identification and version number of the software tool,
- b) the maximum Tool Confidence Level for which the software tool is classified together with a reference to its evaluation analysis,
- c) the pre-determined maximum ASIL, or specific ASIL, of any safety requirement which might be violated if the software tool is malfunctioning and produces corresponding erroneous output,
- d) the configuration and environment for which the software tool is qualified,
- e) the person or organization who carried out the qualification,
- f) the methods applied for its qualification in accordance with 11.4.6.1,
- g) the results of the measures applied to qualify the software tool, and
- h) the usage constraints and malfunctions identified during the qualification, if applicable.

11.4.7 Increased confidence from use

11.4.7.1 If the method “Increased confidence from use” in accordance with Table 4 or Table 5 is applied for the qualification of a software tool the requirements of this subclause shall be complied with.

11.4.7.2 A software tool shall only be argued as having increased confidence from use, if evidence is provided for the following:

NOTE The requirements of the proven in use argument from Clause 14 are not applicable to this subclause.

- a) the software tool has been used previously for the same purpose with comparable use cases and with a comparable determined operating environment and with similar functional constraints,
- b) the justification for increased confidence from use is based on sufficient and adequate data,

NOTE Data can be obtained through accumulated amount of usage (e.g. duration or frequency).

- c) the specification of the software tool is unchanged, and
- d) the occurrence of malfunctions and corresponding erroneous outputs of the software tool acquired during previous developments are accumulated in a systematic way.

11.4.7.3 The experience from the previous usage of the software tool during given development activities shall be analysed and evaluated by considering the following information:

- a) the unique identification and version number of the software tool,
- b) the configuration of the software tool,
- c) the details of the period of use and relevant data on its use,

EXAMPLE Used features of the software tool and frequency of their use for relevant use cases of the software tool.

- d) the documentation of malfunctions and corresponding erroneous outputs of the software tool with details of the conditions leading to them,
- e) the list of the previous versions monitored, listing the malfunctions fixed in each relevant version, and
- f) the safeguards, avoidance measures or work-arounds for the known malfunctions, or detection measures for a corresponding erroneous output, if applicable.

EXAMPLE Sources for the usage report can be a log-book; the version history provided by the supplier of the software tool, published errata sheets.

11.4.7.4 The increased confidence from use argument shall only be valid for the considered version of the software tool.

11.4.8 Evaluation of the tool development process

11.4.8.1 If the method “Evaluation of the tool development process” in accordance with Table 4 or Table 5 is applied for the qualification of a software tool, the requirements of this subclause shall be complied with.

11.4.8.2 The development process applied for the development of the software tool shall comply with an appropriate standard.

NOTE For open source developments, some of the standards used by those communities can also be appropriate.

11.4.8.3 The evaluation of the development process applied for the development of the software tool shall be provided by an assessment based on an appropriate national or international standard and the proper application of the assessed development process shall be demonstrated.

NOTE This assessment covers the development of an adequate and relevant subset of the features of the software tool.

EXAMPLE Using an assessment method based on Automotive SPICE, CMMI, ISO 15504.

11.4.9 Validation of the software tool

11.4.9.1 If the method “Validation of the software tool” according to Table 4 or Table 5 is applied for the qualification of a software tool, the requirements of this subclause shall be complied with.

11.4.9.2 The validation of the software tool shall meet the following criteria:

- a) the validation measures shall demonstrate that the software tool complies with its specified requirements,

NOTE Tests designed to evaluate functional and non-functional quality aspects of the software tool can be used for validation.

EXAMPLE The standard for a programming language helps to define the requirements for validating the associated compiler.

- b) the malfunctions and their corresponding erroneous outputs of the software tool occurring during validation shall be analysed together with information on their possible consequences and with measures to avoid or detect them, and

- c) the reaction of the software tool to anomalous operating conditions shall be examined;

EXAMPLE Foreseeable misuse, incomplete input data, incomplete update of the software tool, use of prohibited combinations of configuration settings.

11.4.10 Confirmation review of qualification of a software tool

This subclause applies to ASILs (B), C, D, in accordance with 4.3.

The confidence in the use of the software tool shall be evaluated in accordance with ISO 26262-2:2011 Table 1 to ensure:

- a) the correct evaluation of the required level of confidence in the software tool, and
- b) the appropriate qualification of the software tool in accordance with its required level of confidence.

11.5 Work products

11.5.1 Software tool criteria evaluation report resulting from requirements 11.4.1, 11.4.2, 11.4.3, 11.4.4, 11.4.5 and 11.4.10.

11.5.2 Software tool qualification report resulting from requirements 11.4.1 to 11.4.10.

12 Qualification of software components

12.1 Objectives

The objective of the qualification of software components is to provide evidence for their suitability for re-use in items developed in compliance with ISO 26262.

12.2 General

The re-use of qualified software components avoids re-development for software components with similar or identical functionality.

NOTE Software components are understood to include source code, models, pre-compiled code, or compiled and linked software.

EXAMPLE Software components addressed by this clause include:

- software libraries from third-party suppliers [commercial off-the-shelf (COTS) software];
- in-house components already in use in electronic control units.

12.3 Inputs to this clause

12.3.1 Prerequisites

The following information shall be available:

- requirements of the software component (from external source).

12.3.2 Further supporting information

The following information can be considered:

- design specification of the software component (from an external source);
- results of previous verification measures of the software component (from an external source).

12.4 Requirements and recommendations

12.4.1 General

To be able to consider a software component as qualified, the following shall be available:

- a) the specification of the software component in accordance with 12.4.3.1,
- b) evidence that the software component complies with its requirements in accordance with 12.4.3.2, 12.4.3.3, and 12.4.3.4,
- c) evidence that the software component is suitable for its intended use in accordance with 12.4.4, and
- d) evidence that the software development process for the component is based on an appropriate national or international standard.

NOTE Some re-engineering activities can be performed in order to comply with this subclause in the case of previously developed software components.

12.4.2 Software component qualification planning

12.4.2.1 The planning of qualification of a software component shall determine:

- a) the unique identification of the software component,
- b) the maximum target ASIL of any safety requirement which might be violated if the software component performs incorrectly, and
- c) the activities that shall be carried out to qualify the software component.

12.4.3 Qualification of a software component

12.4.3.1 The specification of the software component shall include:

- a) the requirements of the software component,

EXAMPLE The following requirements:

- functional requirements,
- accuracy of algorithm or numerical accuracy, where accuracy of algorithm considers procedural errors, which only provide approximate solutions and numerical accuracy considers rounding errors, resulting from computational inaccuracy, and truncation errors caused by the approximate representation of many functions in the electronic control unit,
- behaviour in the case of failure,
- response time,
- resource usage,
- requirements on the runtime environment; and
- behaviour in an overload situation (robustness).

- b) the description of the configuration,

NOTE For software components that contain more than one software unit, the description of the configuration includes the unique identification and configuration of each software unit.

- c) the description of interfaces,
- d) the application manual, where appropriate,
- e) the description of the software component integration,

NOTE Description might include the development tools required to integrate and use the software component.

- f) the reactions of the functions under anomalous operating conditions,

EXAMPLE Re-entrant calling of non-re-entrant software component functions.

- g) the dependencies with other software components, and
- h) a description of known anomalies with corresponding work-around measures.

12.4.3.2 To provide evidence that a software component complies with its requirements the verification of this software component shall:

- a) show a requirement coverage in accordance with ISO 26262-6:2011, Clause 9,

NOTE This verification is primarily based on requirements-based testing. The results of requirements-based tests of the software component executed during its development or during previous integration tests can be used.

EXAMPLE Application of a dedicated qualification test suite, analysis of all the tests already executed during the implementation and any integration of the software component.

- b) cover both normal operating conditions and behaviour in the case of failure, and
- c) result in no known errors that lead to violation of safety requirements.

12.4.3.3 This subclause applies to ASIL D in accordance with 4.3.

The structural coverage shall be measured in accordance with ISO 26262-6:2011, Clause 9, to evaluate the completeness of the test cases. If necessary, additional test cases shall be specified or a rationale shall be provided.

12.4.3.4 The verification in accordance with 12.4.3.2, shall only be valid for an unchanged implementation of the software component.

12.4.3.5 The qualification of a software component shall be documented including the following information:

- a) the unique identification of the software component,
- b) the unique configuration of the software component,
- c) the person or organization who carried out the qualification,
- d) the environment used for qualification,
- e) the results of the verification measures applied to qualify the software component, and
- f) the maximum target ASIL of any safety requirement that might be violated if the software component performs incorrectly.

12.4.4 Verification of qualification of a software component

12.4.4.1 The results of qualification of a software component together with the validity of these results regarding the intended use of the software component shall be verified. If necessary, additional measures shall be applied.

NOTE The validity of the qualification can be influenced when the qualification has been performed in the context of another industrial or automotive domain.

EXAMPLE Engine control, body control and chassis control are different automotive domains. Railways and civil avionics are different industrial domains.

12.4.4.2 The specification of the software component shall comply with the requirements of the intended use of this software component.

12.5 Work products

12.5.1 Software component documentation resulting from requirement 12.4.3.1.

12.5.2 Software component qualification report resulting from requirement 12.4.3.5.

12.5.3 Safety plan (refined) resulting from requirement 12.4.2.

13 Qualification of hardware components

13.1 Objectives

The first objective of the qualification of hardware components is to provide evidence of the suitability of intermediate level hardware components and parts for their use as part of items, systems or elements, developed in compliance with ISO 26262, concerning their functional behaviour and their operational limitations for the purposes of the safety concept.

The second objective of the qualification of hardware components is to provide relevant information regarding:

- their failure modes,
- their failure mode distribution, and
- their diagnostic capability with respect to the safety concept for the item.

13.2 General

Every safety-related hardware component and part used within the scope of ISO 26262 is subject to standard qualification to address general functional performance, conformity of production, environmental endurance and robustness.

EXAMPLE 1 Qualification in accordance with ISO 16750, or with AEC-Q100 or AEC-Q200 standards, for electronic parts or equivalent company standards.

For basic parts (passive component, discrete semiconductor), standard qualification is sufficient. These basic parts can then be used in a hardware design in accordance with ISO 26262-5.

The requirements of this clause apply to intermediate level hardware components or parts, which provide dedicated functionality to the system.

EXAMPLE 2 Sensors, actuators, ASICs with dedicated functionality (e.g. protocol adapter).

If the intermediate level hardware component or part is safety-related, depending on its level, it is integrated and tested in accordance with ISO 26262-4 or ISO 26262-5 or both in addition to its qualification in accordance with this clause.

Usually the qualification described in this clause can be applied to components or parts whose failure modes or malfunctions are known and that are adequately testable regarding their possible failures.

EXAMPLE 3 During the development of a fuel pressure sensor the correct function of the sensor was approved within its boundary of operation up to 200 bar fuel pressure and 140 °C temperature. The qualification of this fuel pressure sensor enables the use of this sensor for the realisation of a particular safety-related item with regard to functional performance of the sensor and its malfunctions, provided that the same or lower boundaries of operation apply. In such a situation, the design analysis and the integration and testing of the basic hardware of the sensor according to ISO 26262-5 can be omitted and the integration activities can be carried out directly following ISO 26262-4 with regard to the technical safety requirements allocated to the sensor.

A summary for qualification and integration of basic parts, hardware parts and components is shown in Table 6.

Table 6 — Qualification, integration and test activities to be conducted depending on the level of hardware part or component

Activity	Hardware part or component			
	Safety-related basic hardware part	Safety-related intermediate hardware part	Safety-related intermediate hardware component	Safety-related complex hardware component
	(e.g. resistors, transistors)	(e.g. gray code decoder)	(e.g. fuel pressure sensor)	(e.g. ECU)
Standard qualification	Applicable	Applicable	—	—
Qualification in accordance with Clause 13	—	Applicable	Applicable	—
Integration/test in accordance with ISO 26262-5	—	Applicable ^a	Applicable ^a	Applicable
Integration/test in accordance with ISO 26262-4	—			Applicable

^a The hardware part or component will be integrated in accordance with ISO 26262-4, or ISO 26262-5, or both ISO 26262-4 and ISO 26262-5, depending on its level.

Qualification of hardware components or parts can be done using two different methods: testing or analysis. These methods can be used individually or in combinations depending on the hardware components or parts.

- When testing, the hardware component or part is exposed to the intended environmental and operational conditions and compliance with its functional requirements is assessed. Reproducing exact environmental conditions is difficult and also any extrapolations are subject to error, therefore the limitations of such test conditions are considered when interpreting the results of tests.
- A qualification through analysis relies on a rationale for the analytical methods and assumptions used. In general, a hardware component is too complex to be qualified by analysis alone. However, the analysis can be used effectively for the extrapolation of testing data and to determine the effects of smaller changes in the already tested hardware component.

Even if different qualification methods are used, the final results are available in a qualification report (which can consist of a set of documents that include reports on findings, notes on interpretation, etc.) that gives evidence of the assumptions, conditions and test cases and results used to qualify the hardware components or parts with associated results. If possible, it is better to formulate the synthesis in such a way that independent checking is possible; it usually includes the performance data, the qualification process, the results and the rationales.

Directions present in ISO 16750 are useful for defining the type and sequence of qualification tests.

13.3 Inputs to this clause

13.3.1 Prerequisites

The following information shall be available:

- related safety requirements,
- qualification criteria (analysis and tests) in accordance with ISO 26262-5:2011, Clause 6, and

- the manufacturer's hardware component or part specification, or, if unavailable, the assumptions on hardware component or part specification (from an external source).

13.3.2 Further supporting information

The following information can be considered:

- test criteria in accordance with ISO 26262-5:2011, Clause 6;
- see further supporting information for the phases of the safety lifecycle where the qualification of hardware components is applied.

13.4 Requirements and recommendations

13.4.1 General

13.4.1.1 The criteria to apply this clause are:

- a) the component or part to be qualified shall have an intermediate complexity excluding complex hardware components and basic hardware parts, and
- b) the relevant failure modes of the component or part to be qualified shall be assumed to be verifiable by testing, analysis or both.

13.4.2 Goals of qualification of hardware components or part

13.4.2.1 The following goals shall be achieved by the qualification of hardware component or part:

- a) adequate functional performance of components or parts for the purposes of the safety concept,
- b) identification of failure modes and models (quantification of their distribution) by using appropriate tests (such as over limit test, accelerated test...) or analyses,
- c) sufficient robustness, and
- d) identification of limits of use for components or parts.

13.4.3 Methods for qualification of the hardware component or part

13.4.3.1 The qualification of the hardware component or part shall be carried out using an appropriate selection of the following methods:

- a) analyses, and
- b) testing.

13.4.4 Qualification plan

13.4.4.1 A qualification plan shall be developed and shall describe:

- a) precise identification and version of the hardware component or part,
- b) specification of the environment in which the hardware component or part is intended to be used,
- c) the qualification strategy and the rationale,

NOTE The strategy includes: analysis, tests necessary and step by step description.

- d) the necessary tools and equipment enabling this strategy,
- e) the party responsible for carrying out this strategy, and
- f) the criteria used to assess the qualification of a hardware component or part as passed or failed.

13.4.5 Qualification argument

13.4.5.1 A comprehensive argument that the performance of the hardware component or part complies with its specification shall be made available.

NOTE The required performances encompass behaviour when it is subjected to the established normal environmental conditions and to the environmental conditions in combination with an assumed failure initiating event.

13.4.5.2 The comprehensive argument of 13.4.5.1 shall be based on a combination of the following types of information:

- a) analytical methods and assumptions used; or
- b) data from operational experience; or
- c) existing testing results.

13.4.5.3 A rationale for each assumption, including extrapolations, shall be given.

13.4.6 Qualification by analyses

13.4.6.1 The analysis shall be expressed in a form that can be easily understood and checked by persons who are qualified in the relevant engineering or scientific disciplines.

NOTE Analytical methods that can be used include extrapolations, mathematical models, damage analysis or similar methods.

13.4.6.2 The analyses shall consider all the environmental conditions to which the hardware component or part is exposed, the limits of these conditions and, other additional strains related to operation (e.g. expected switch cycles, charging and discharging, long turn-off times).

13.4.7 Qualification by Testing

13.4.7.1 A test plan shall be developed and shall contain the following information:

- a) description of the functions of the hardware component or part,
- b) number and sequence of tests to be conducted,
- c) requirements for assembly and connections,
- d) procedure for accelerated ageing, considering the operating conditions of the hardware component or part,
- e) operating and environmental conditions to be simulated,
- f) pass/fail criteria to be established,
- g) environmental parameters to be measured,
- h) requirements for the testing equipment, including accuracy, and
- i) maintenance and replacement processes permitted during the testing.

13.4.7.2 A standardized test specification shall be used.

NOTE This specification can be based on the ISO 16750 series of parts or equivalent company standards.

13.4.7.3 The test shall be conducted as planned and the resulting test data shall be made available.

13.4.8 Qualification report

13.4.8.1 The qualification report shall state whether the hardware component or part has passed or failed the qualification with respect to the operating envelope.

NOTE The qualification report can consist of a set of documents that includes reports on findings and notes on interpretation.

13.4.8.2 The qualification report shall be verified in accordance with Clause 9.

13.5 Work products

13.5.1 Qualification plan resulting from requirement 13.4.4.

13.5.2 Hardware component test plan if applicable, resulting from requirement 13.4.7.1.

13.5.3 Qualification report resulting from requirement 13.4.8.1.

14 Proven in use argument

14.1 Objectives

This clause provides guidance for a proven in use argument. A proven in use argument is an alternate means of compliance with ISO 26262 that may be used in the case of reuse of existing items or elements when field data is available.

14.2 General

A proven in use argument can be applied to any type of product whose definition and conditions of use are identical to or have a very high degree of commonality with a product that is already released and in operation. It can also be applied to any work product related to such products.

NOTE 1 Proven in use argument is not inter-changeability: one product, with alternate design or implementation, that is intended to replace a proven in use product cannot be considered to be proven in use because it fulfils the original functional requirements, unless this product meets the criteria specified in this clause.

An item or an element, such as system, function, hardware or software product, can be a candidate for a proven in use argument.

A candidate can also refer to system, hardware or software work products such as a technical safety concept, algorithms, models, source code, object code, software components, a set of configurations or calibration data.

The motivation for using the argument for proven in use includes:

- a) an automotive application in commercial use intended to be partly or completely carried over to another target; or
- b) an ECU in operation intended to implement an additional function; or
- c) a candidate being in the field prior to the release of ISO 26262; or

- d) a candidate being used in other safety-related industries; or
- e) a candidate being a widely used COTS product not necessarily intended for automotive applications.

The proven in use argument is substantiated by appropriate documentation on the candidate, configuration management and change management records, and field data regarding safety-related incidents.

Once a candidate has been defined (see 14.4.3) with the expected proven in use credit (see 14.4.2), two important criteria need to be considered when preparing a proven in use argument:

- the relevance of field data during the service period of the candidate (see 14.4.5), and
- the changes, if any, that could have impacted the candidate since its service period (see 14.4.4).

NOTE 2 With regard to the relevance of field data, the proven in use argument is intended to address systematic and random failures of the candidate; it does not address failures related to ageing of the candidate.

Using proven in use items or elements does not exempt those items or elements from the following project-dependent safety management activities:

- the proven in use credit is described in the safety plan, and
- the data and work products resulting from the proven in use argument are part of the safety case and subject to confirmation measures.

14.3 Inputs to this clause

14.3.1 Prerequisites

The following information shall be available:

- regarding the intended use of a candidate:
 - candidate specification,
 - applicable safety goal(s) or safety requirement(s) with corresponding ASIL(s), and
 - foreseeable operational situation and intended operating modes and interfaces;
- regarding the previous use of a candidate:
 - field data from the service period (from an external source).

14.3.2 Further supporting information

The following information can be considered:

- regarding the previous use of a candidate:
 - safety case in accordance with ISO 26262-2:2011, 6.5.3.

NOTE For a candidate not developed in accordance with ISO 26262 (e.g. COTS products, candidates developed under a safety standard other than ISO 26262, such as IEC 61508 or RTCA DO-178), some work products of the safety case might not be available, then they are substituted by available data resulting from the development of the candidate.

14.4 Requirements and recommendations

14.4.1 General

14.4.1.1 The following subclauses refer to the ASILs applicable to the future use of the candidate.

14.4.2 Proven in use credit

14.4.2.1 A proven in use credit shall be used only when the candidate complies with 14.4.2 to 14.4.5.

14.4.2.2 The proven in use credit resulting from a proven in use argument shall be planned in accordance with ISO 26262-2:2011, 6.4.3.5.

14.4.2.3 The proven in use credit shall be limited to the safety lifecycle subphases and activities covered by the proven in use argument of the candidate.

14.4.2.4 Integration measures of proven in use elements in an item or element shall be carried out at the appropriate level in accordance with ISO 26262-4:2011, Clause 8.

EXAMPLE The hardware of an ECU has a satisfactory service history and is intended to be 100 % carried over to a new application. The proven in use credit can be applied to the subphases and activities of development of this hardware element. Similarly, if the software is a 100 % carryover with a satisfactory service history then the proven in use credit can also be applied to the software subphases and activities.

14.4.2.5 Safety validation of an item which embeds proven in use elements shall be carried out in accordance with ISO 26262-4:2011, Clause 9.

14.4.2.6 The confirmation measures of an item that embeds proven in use elements shall consider the proven in use arguments and related data in accordance with ISO 26262-2:2011, 6.4.7.

14.4.2.7 Any change to a proven in use item or element shall comply with 14.4.4 for the corresponding proven in use credit to be maintained.

NOTE This clause applies to any type of modification including those initiated as a result of a safety-related incident.

14.4.3 Minimum information on candidate

14.4.3.1 A description of the candidate and its previous use shall be available, that includes:

- a) the identification and traceability of the candidate with a catalogue of internal elements or components if any,
- b) the corresponding fit, form and function requirements that describe, if applicable, interface and environmental, physical and dimensional, functional and performance characteristics of the candidate, and
- c) the safety requirements of the candidate in the previous use and the corresponding ASILs, if available.

14.4.4 Analysis of changes to the candidate

14.4.4.1 Proven in use candidates

Changes to candidates and their environment shall be identified in accordance with 14.4.4.2 to 14.4.4.3.

NOTE 1 Changes to candidates address design changes and implementation changes. Design changes can result from modification of requirements, functional enhancements or performance enhancement. Implementation changes do not affect specification or performances of the candidate but only its implementation features. Implementation changes can result from software corrections or use of new development or production tools.

NOTE 2 Changes to configuration data or calibration data are considered as changes to the candidate when they impact its behaviour with regard to the violation of the safety goals.

NOTE 3 Changes to the environment of a candidate can result from use of this candidate in a new type of application with different safety goals or requirements, its installation in a new target environment (e.g. variant of vehicle, range of environmental conditions) or the upgrading of the components interacting with it or located in its vicinity.

14.4.4.2 Changes to items introduced for a future application

Changes to items and their environment introduced for the purpose of a future application shall comply with ISO 26262-3:2011, 6.4.2.

14.4.4.3 Changes to elements introduced for a future application

Changes to elements and their environment introduced for the purpose of a future application within a different item shall comply with Clause 8.

14.4.4.4 Changes to candidate independent of future application

Changes to a candidate introduced after its service period, independent of future applications, shall provide evidence that the proven in use status remains valid.

14.4.5 Analysis of field data

14.4.5.1 Configuration management and change management

Evidence shall be provided that the candidate has been kept under configuration management and change management during and after its service period so that the current status of the candidate can be established.

14.4.5.2 Target values for proven in use

NOTE When any ASIL is not yet assigned to the candidate, ASIL D target is selected conservatively.

14.4.5.2.1 The rationale for the calculation of the service period of the candidate shall be available.

14.4.5.2.2 The service period of the candidate shall result from the addition of the observation period of all the specimens taken in reference in accordance with 14.4.5.2.3.

14.4.5.2.3 The observation period of each specimen with the same specification and realization as the candidate and running in a vehicle shall exceed the average yearly vehicle operating time before being considered in the analysis of the service period of the candidate.

14.4.5.2.4 For a proven in use status to be obtained by the candidate, its service period shall demonstrate compliance with each safety goal that can be violated by the candidate in accordance with Table 7 with a single-sided lower confidence level of 70 % (using a chi-square distribution).

NOTE 1 For the purpose of the proven in use argument, an observable incident means a failure that is reported to the manufacturer and caused by the candidate with the potential to lead to the violation of a safety goal.

Table 7 — Limits for observable incident rate

ASIL	Observable incident rate
D	$<10^{-9}/h$
C	$<10^{-8}/h$
B	$<10^{-8}/h$
A	$<10^{-7}/h$

NOTE 2 The character and rate of observable incidents are interpreted when analysing the potential violation of the safety goals in the field

NOTE 3 Table 8 gives an example of the required minimum service period without observable incident which is necessary to achieve 70 % confidence:

Table 8 — Targets for minimum service period of candidate

ASIL	Minimum service period without observable incident
D	$1,2 \times 10^9$ h
C	$1,2 \times 10^8$ h
B	$1,2 \times 10^8$ h
A	$1,2 \times 10^7$ h

NOTE 4 If observable incidents are found in the collected data of the specimens, the necessary minimum service period, t_{service} , can be adjusted as follows:

$$t_{\text{service}} = t_{\text{MTTF}} \times \frac{(\chi_{\text{CL};2f+2})^2}{2}$$

where

CL is the confidence level as an absolute value (e.g. 0,7 for 70 %);

t_{MTTF} is the mean time to failure (1/failure rate);

f is the number of safety-related incidents;

$(\chi_{\alpha,\nu})^2$ is the chi-squared distribution with error probability α and ν degrees of freedom.

14.4.5.2.5 The application of the proven in use credit may be anticipated provisionally, before a proven in use status is obtained (in accordance with 14.4.5.2.4). In this case, the service period of the candidate shall demonstrate compliance with each safety goal that can be violated by the candidate in accordance with Table 9 with a single sided lower confidence level of 70 % (using a chi-square distribution).

Table 9 — Limits for observable incident rate (interim period)

ASIL	Observable incident rate
D	$<3 \times 10^{-9}/\text{h}$
C	$<3 \times 10^{-8}/\text{h}$
B	$<3 \times 10^{-8}/\text{h}$
A	$<3 \times 10^{-7}/\text{h}$

14.4.5.2.6 In the case of any observed incident in the field during the interim period described in 14.4.5.2.5, the following shall be complied with:

- to stop using Table 9 to the observable incident rate and to use Table 7 for the candidate; or alternatively
- to provide evidence that the root cause of the observed incident is fully identified and eliminated in accordance with ISO 26262, and to keep on counting the cumulated hours for the candidate, to reset the counter of cumulated hours for this specific root cause and to record this evidence in the safety case.

14.4.5.2.7 In the case of a candidate with a non-constant failure rate, additional measures shall be applied for the proven in use argument, for instance in the case of damage resulting from fatigue.

NOTE Those measures apply to candidates with failure rates significantly dependent on factors such as wear, ageing or operating hours regarding the lifetime of the item. They can include dedicated endurance tests, or a longer observation period.

14.4.5.3 Field problems

The problem reporting system shall ensure that any observed incident with potential safety impact caused by the candidate in the field, is recorded and retrievable during the period of operation of the candidate (see ISO 26262-7:2011, 6.4.2.1).

14.5 Work products

14.5.1 Safety plan (refined) resulting from requirements 14.4.2.1 to 14.4.2.7.

14.5.2 Description of candidate for proven in use argument resulting from requirement 14.4.3.

14.5.3 Proven in use analysis reports resulting from requirements 14.4.4 to 14.4.5.

Annex A (informative)

Overview on and document flow of supporting processes

Table A.1 provides an overview on objectives, prerequisites and work products of the supporting processes.

Table A.1 — Overview of supporting processes

Clause	Objectives	Prerequisites	Work products
5 Interfaces within distributed developments	The objective of this clause is to describe the procedures and to allocate associated responsibilities within distributed developments for items and elements.	See prerequisites of the relevant phases of the safety lifecycle for which the distributed development is carried out.	5.5.1 Supplier selection report 5.5.2 Development interface agreement (DIA) 5.5.3 Supplier's project plan 5.5.4 Supplier's safety plan 5.5.5 Safety assessment report 5.5.6 Supply agreement
6 Specification and management of safety requirements	The first objective is to ensure the correct specification of safety requirements with respect to their attributes and characteristics. The second objective is to ensure consistent management of safety requirements throughout the entire safety lifecycle	See applicable prerequisites of the relevant phases of the safety lifecycle in which safety requirements are specified or managed.	None
7 Configuration management	The first objective is to ensure that the work products, and the principles and general conditions of their creation, can be uniquely identified and reproduced at any time. The second objective is to ensure that the relations and differences between earlier and current versions can be traced.	Safety plan in accordance with ISO 26262-2:2011, 6.5.1. Applicable prerequisites of the relevant phases of the safety lifecycle where configuration management is planned or managed.	7.5.1 Configuration management plan
8 Change management	The objective of change management is to analyse and control changes to safety-related work products occurring throughout the safety lifecycle.	Configuration management plan (see 7.5.1). Project plan in accordance with ISO 26262-2:2011, 6.5.2.	8.5.1 Change management plan 8.5.2 Change request 8.5.3 Impact analysis and change request plan 8.5.4 Change report
9 Verification	The objective of verification is to ensure that the work products comply with their requirements.	See applicable prerequisites of the relevant phases of the safety lifecycle in which verification is planned or carried out.	9.5.1 Verification plan 9.5.2 Verification specification 9.5.3 Verification report
10 Documentation	The primary objective is to develop a documentation management strategy for the entire safety lifecycle in order to facilitate an effective and repeatable documentation management process.	Safety plan in accordance with ISO 26262-2:2011, 6.5.1.	10.5.1 Document management plan 10.5.2 Documentation guideline requirements

Table A.1 (continued)

Clause	Objectives	Prerequisites	Work products
11 Confidence in the use of software tools	<p>The first objective of this clause is to provide criteria to determine the required level of confidence in a software tool when applicable.</p> <p>The second objective of this clause is to provide means for the qualification of the software tool when applicable, in order to create evidence that the software tool is suitable to be used to tailor the activities or tasks required by ISO 26262 (i.e. the user can rely on the correct functioning of a software tool for those activities or tasks required by ISO 26262).</p>	<p>Safety plan (see ISO 26262-4, 5.5.2).</p> <p>Applicable prerequisites of the phases of the safety lifecycle where a software tool is used.</p>	11.5.1 Software tool criteria evaluation report 11.5.2 Software tool qualification report
12 Qualification of software components	The objective of the qualification of software components is to provide evidence for their suitability for re-use in items developed in compliance with ISO 26262.	Requirements of the software component.	12.5.1 Software component documentation 12.5.2 Software component qualification report 12.5.3 Safety plan (refined)
13 Qualification of hardware components	<p>The first objective of the qualification of hardware components is to provide evidence of the suitability of intermediate level hardware components and parts for their use as part of items, systems or elements, developed in compliance with ISO 26262, concerning their functional behaviour and their operational limitations for the purposes of the safety concept.</p> <p>The second objective of the qualification of hardware components is to provide relevant information regarding their failure modes, their failure mode distribution, and their diagnostic capability with respect to the safety concept for the item.</p>	Related safety requirements. Qualification criteria (analysis and tests) in accordance with ISO 26262-5:2011, Clause 6; and manufacturer's hardware component specification, or, if unavailable, the assumptions on hardware component specification.	13.5.1 Qualification plan 13.5.2 Hardware component test plan 13.5.3 Qualification report
14 Proven in use argument	This clause provides guidance for a proven in use argument. A proven in use argument is an alternate means of compliance with ISO 26262 that may be used in the case of reuse of existing items or elements when field data is available.	Regarding the intended use of a candidate: candidate specification; applicable safety goal(s) or safety requirement(s) with corresponding ASIL(s); foreseeable operational situation and intended operating modes and interfaces. Regarding the previous use of a candidate: field data from service period (from external source).	14.5.1 Safety Plan (refined) 14.5.2 Definition of a candidate for proven in use argument 14.5.3 Proven in use analysis reports

Annex B (informative)

DIA example

B.1 Purpose

This annex provides an illustrative example of a development interface agreement (DIA), in accordance with the requirements of Clause 5 [especially 5.4.3.1 c) to g)], with organization-specific adaptation under the requirements and recommendations of ISO 26262-2, 5.4.5 and 5.5.1, if any. Project specific tailoring, in accordance with ISO 26262-2, 6.4.5, can also be applied.

B.2 General

Many factors will affect the type and amount of customer-supplier interactions; the example is simplified, based on an application scenario described in B.3 and a set of premises listed in B.4.

Tables B.1 to B.3 constitute an example of a DIA as follows:

- Table B.1 approximately corresponds to the requirements of 5.4.2, with some organization-specific additions, intended to avoid or eliminate risk from a supplier with inadequate capability.
- Table B.2 approximately corresponds to the requirements of 5.4.3, with some organization-specific additions, intended to avoid or eliminate risk from improper understanding or definition of the boundary of Component C and its interactions with its environment.
- Table B.3 approximately corresponds to the requirements of 5.4.4, as applied to hardware Component C.

NOTE In each table, the corresponding ISO 26262 clause is indicated in parentheses.

B.3 Application scenario

The DIA example shown in Tables B.1 to B.3 is based on the following application scenario:

- a) The customer is responsible for engineering and manufacturing the vehicle.
- b) The customer is responsible for engineering the system comprised of many hardware and software components of which one hardware component, C, is to be sourced from some supplier.
- c) Component C will be allocated requirements with assigned ASIL D.
- d) Component C has not been developed previously, i.e., it is not a commercial off-the-shelf (COTS) product. It involves new technology for which there is an inadequate pool of proven suppliers.
- e) Multiple suppliers are interested in the supply of Component C, but adequate capability to support the project is not evident.
- f) A model-based development process is used.

B.4 Premises

This example is developed on the following premises:

- a) Resources required for project management and engineering are available when needed.
- b) Assessment teams that qualify as “independent” are available to each participating organization, and are used where needed.
- c) The same process and architectural framework is in use in all the participating organizations, independently assessed to qualify for the highest integrity level.
 - 1) Reusable assets conform to the process and architectural framework, and are independently assessed to qualify for the required integrity level.
 - 2) Other resources, e.g., tools, conform to the process and architectural framework, and are independently assessed to qualify for the required integrity level.
 - 3) The participating organizations choose specific processes and tools that are compatible, and commit to the same architecture.
 - 4) Explicit meta-models or specifications define unambiguously the semantics of the tools, modelling languages, programming languages, and the produced models.
 - 5) Models of externally-visible behaviour, performance (including worst-case), and failure modes and effects are available for hardware components, including I/O devices. The models are in a form that can be correctly integrated to create (sub-)system models.
- d) There is high quality execution of other customer-supplier interactions, not unique to high integrity engineering, not included in this example, e.g., interactions for business processes, project management, and quality management.

In case the premises above do not hold, additional customer-supplier interactions and effort will be required – not identified in this example.

Table B.1 — Customer-supplier data exchanges to qualify and select supplier

ID	Activity	Data from customer to supplier	Data from supplier to customer
A.1	Pre-qualify ^a suppliers; project independent criteria; feeds into 5.4.2	Capability assessment questionnaire ^a : — safety culture (ISO 26262-2:2011, 5.4.2); — evidence of competence (ISO 26262-2:2011, 5.4.3); — evidence of quality management (ISO 26262-2:2011, 5.4.4); — ISO 26262 Consent, e.g.: — independent assessment (5.4.5); — DIA template	—
A.2		—	Acceptance of conditions ^a
A.3		—	Capability assessment ^a (ISO 26262-2:2011, Clause 5) Disclosure ^a Corrective action proposed ^a
A.4		Evaluation: ASILs for which not qualified ^a	—
A.5	Qualify suppliers (short-list) ^a 5.4.2	Customer-organization-specific process adaptation of ISO 26262-2:2011, 5.4.5 incl. methods, languages, tools & usage constraints/guidelines.	—
		—	1 st party assessment of compliance. Disclosure ^a Track record (5.4.2.1). Corrective action proposed ^a Alternative approach or proposal to meet objectives ^a
		Iterative evaluation & enquiries about gaps and alternatives ^a	Iterative revisions to plans and alternatives ^a
		Evaluation: ASILs for which not qualified ^a	—
A.6	Invite proposal 5.4.2.2	RFP/RFQ, including project-specific tailored process [5.4.3.1 b)], product concept i.e. item definition (ISO 26262-3:2011, 5.5) and safety goals (ISO 26262-3:2011, 7.5.2).	—
A.7	—	—	Offer; Statement of compliance; Updates to previously submitted information ^a
A.8	Select supplier 5.4.2	Proposed DIA (project-specific) 5.4.3	—
A.9		—	Selected project resources and their capability assessment, e.g. safety team members' skills, competencies and qualification (ISO 26262-2:2011, 5.5.2); Organization-specific rules and processes (ISO 26262-2:2011, 5.5.1), incl. tools, libraries; Preliminary plans, e.g. safety plan (ISO 26262-2:2011, 6.5.1)
A.10		Iterative evaluation and enquiries, e.g. regarding skill gaps ^a	Iterative revisions addressing customer concerns ^a
A.11		Acceptance of DIA. (5.5.2) Selection report (5.5.1)	Acceptance of DIA (5.5.2)
A.12		Contract for concept (ISO 26262-3; ISO 26262-4) and planning phase (ISO 26262-4:2011, Clause 5) incl. statement of development work.	Acceptance.

^a Activity or data which is organization-specific and is not required in ISO 26262.

Table B.2 — Customer-supplier data exchanges in project initiation and system concept

ID	Activity	Data from customer to supplier	Data from supplier to customer
B.1	Initiate project (5.4.3) Create functional safety concept (ISO 26262-3:2011, Clauses 5 to 8)	System level plans Item definition (ISO 26262-3:2011, 5.5) and its lifecycle (Figure 1, ISO 26262-2:2011, 5.2.2; ISO 26262-2:2011, Figure 2 and ISO 26262-2:2011, 6.4.5) Functional safety concept (ISO 26262-3:2011, Clause 8)	—
B.2	—	—	Project plan (5.5.3) Safety plan (5.5.4) H&R analysis (5.4.3.2), hardware component behaviour models, incl. fault metrics [5.4.3.1 f], ISO 26262-5:2011, Annex B, and ISO 26262-5:2011, 9.4.3.1]. Independent assessment of plans, incl. assurance that processes and resources are configured and allocated to match the required work products, incl. skill- sets. [5.4.3 c) e), g), 5.4.5]
B.3	—	Acceptance	—
B.4	Consideration of experience gained from proven in use components, tools, libraries used in similar projects (5.4.4.5), as well as proven in use data and analyses of possible candidates (ISO 26262-8:2011, Clause 14)	Initial safety plan (ISO 26262-2:2011, Clause 5), incl. system safety case structure	—
B.5	—	—	Proven in use elements offered (Clause 14), with independent assessment of fitness for the project (5.4.5 and ISO 26262-2:2011, Table 1)
B.6	—	Acceptance	—
B.7	System development lifecycle [5.4.3 b)]	Technical safety concept (ISO 26262-4:2011, 7.5.1), relevant parts of system design specs, hardware specs, design & implementation (D&I) constraints, hardware-software Interface (HSI) specifications (ISO 26262-4:2011, 7.5.3).	Iterative evaluation, clarification-queries, and feedback about conflicts, completeness, consistency, etc.; technological limitations, if any; change requests, if any (5.4.4). Updated behaviour models, incl. fault models.
B.8		Iterative clarifications, responses, and revisions, including updates to system architecture design & verification specifications (ISO 26262-4:2011, 7.5.2, ISO 26262-4:2011, 7.5.5), hardware specifications (ISO 26262-5:2011, 7.5.1) relevant to Component C, HSI, allocation, etc.	Feedback about boundary between Component C & its environment.
B.9	—	—	Acceptance

Table B.3 — Customer-supplier data exchanges in hardware development lifecycle

ID	Activity	Data from customer to supplier	Data from supplier to customer
C.1	Plan (5.4.3)	Authorisation for hardware development	-
C.2		—	Plans: Safety plan (5.5.4 and ISO 26262-5:2011, 5.5.1), Project plan (5.5.3 and ISO 26262-5:2011, 5.5.2), item integration and testing plan (see ISO 26262-4:2011, 5.5.3), planning of DIA (5.4.3) etc. Independent reviews of conformance to planning (5.4.4.8 and 5.4.5).
C.3		Acceptance. Authorisation to commence requirements specification.	—
C.4	Requirements (5.4.5 and ISO 26262-5)	—	hardware specifications - derived; refined; D&I constraints (ISO 26262-5:2011, 7.5.1). Extension to Verification Plan ^a HSI change requests, if any (ISO 26262-5:2011, 10.5). Independent safety audit (5.4.4.8) Independent confirmation (5.4.5 and 5.5.5).
C.5	—	Acceptance. Authorisation to commence design.	—
C.6	Design (5.4.5, and ISO 26262-5)	—	Design specs (ISO 26262-5:2011, 7.5.1); implementation constraints, incl. architectural (ISO 26262-5:2011, Clause 8). Extension or modification to H&R analysis (ISO 26262-3:2011, Clause 7), if any. Extension to item integration and testing plan (ISO 26262-5:2011, 10.5). HSI change requests, if any (ISO 26262-5:2011, 10.5). Independent safety audit (5.4.4.8, 5.4.5)
C.7	5.4.4 and 5.4.5	Iterative evaluation and feedback concerning conflicts discovered at system level.	Iterative clarifications, revisions, and other responses addressing customer feedback and enquiries. Independent assessment (5.4.5 and 5.5.5).
C.8	5.4.4 and 5.4.5	Acceptance of component design. Authorisation to implement.	Implementation. Requirements from the environment. Independent assessment (5.4.5 and 5.5.5).
C.9	—	Acceptance	—
C.10	—	—	Prototype part Integrated verification (ISO 26262-5:2011, 10.5) Independent assessment (5.4.5).
C.11	—	Integrated evaluation (ISO 26262-4:2011, Clause 8). Change requests, if any.	—
C.12	—	—	Reviews & audits of processed changes Independent assessment (5.4.5, 5.5.5).
C.13	—	Acceptance	—
C.14	—	—	Sample for series production Independent assessment (5.4.5, 5.5.5).
C.15	—	Integrated evaluation (ISO 26262-4:2011, Clause 8) Change requests, if any.	—
C.16	—	—	Reviews & audits of processed changes Independent assessment (5.4.4, 5.4.5 and 5.5.5).
C.17	—	Authorisation for commencing production phase	—
C.18	—	—	Post-SOP reports (5.4.6 and 5.5.6 and ISO 26262-2:2011, 7.5).

^a Activity or data which is organization-specific and is not required in ISO 26262.

Bibliography

- [1] ISO 10007, *Quality management systems — Guidelines for configuration management*
- [2] ISO 16750 (all parts), *Road vehicles — Environmental conditions and testing for electrical and electronic equipment*
- [3] ISO/TS 16949, *Quality management systems — Particular requirements for the application of ISO 9001:2008 for automotive production and relevant service part organizations*
- [4] ISO/IEC 15504 (all parts), *Information technology — Process assessment*
- [5] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*
- [6] RTCA DO-178 B, *Software Considerations in Airborne Systems and Equipment Certification*
- [7] CMMI, <http://www.sei.cmu.edu/cmmi/>
- [8] German V-Model, <http://www.v-modell-xt.de/>
- [9] AEC-Q100, *Stress Qualification For Integrated Circuits*
- [10] AEC-Q200, *Stress Test Qualification For Passive Components*

This page is intentionally blank.

This page is intentionally blank.

This page is intentionally blank.

ICS 43.040.10

Price based on 48 pages

INTERNATIONAL
STANDARD

ISO
26262-9

First edition
2011-11-15

Road vehicles — Functional safety —

Part 9:

**Automotive Safety Integrity Level (ASIL)-
oriented and safety-oriented analyses**

Véhicules routiers — Sécurité fonctionnelle —

Partie 9: Analyses liées aux niveaux d'intégrité de sécurité automobile
(ASIL) et à la sécurité



Reference number
ISO 26262-9:2011(E)

© ISO 2011



COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction.....	v
1 Scope.....	1
2 Normative references.....	1
3 Terms, definitions and abbreviated terms	2
4 Requirements for compliance.....	2
4.1 General requirements	2
4.2 Interpretations of tables.....	2
4.3 ASIL-dependent requirements and recommendations	3
5 Requirements decomposition with respect to ASIL tailoring.....	3
5.1 Objectives	3
5.2 General	3
5.3 Inputs to this clause.....	4
5.4 Requirements and recommendations	4
5.5 Work products	7
6 Criteria for coexistence of elements	7
6.1 Objectives	7
6.2 General	7
6.3 Inputs to this clause.....	8
6.4 Requirements and recommendations	8
6.5 Work products	9
7 Analysis of dependent failures	9
7.1 Objectives	9
7.2 General	9
7.3 Inputs to this clause.....	9
7.4 Requirements and recommendations	10
7.5 Work products	11
8 Safety analyses.....	11
8.1 Objectives	11
8.2 General	11
8.3 Inputs to this clause.....	13
8.4 Requirements and recommendations	13
8.5 Work products	14
Annex A (informative) Overview of and document flow of Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses	15
Bibliography.....	16

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-9 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

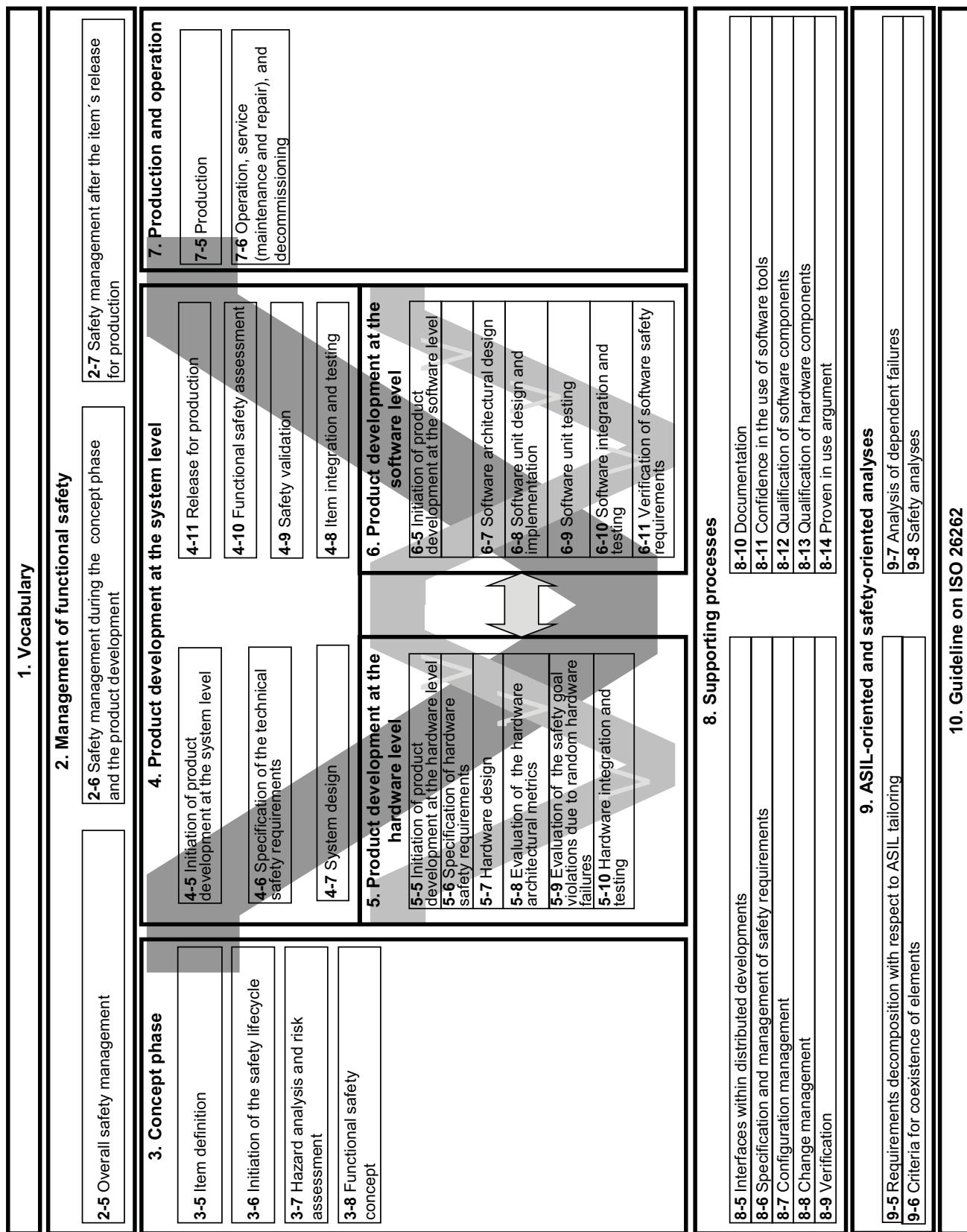


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 9:

Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 specifies the requirements for Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses, including the following:

- requirements decomposition with respect to ASIL tailoring,
- criteria for coexistence of elements,
- analysis of dependent failures, and
- safety analyses.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Requirements for compliance

4.1 General requirements

When claiming compliance with ISO 26262, each requirement shall be complied with, unless one of the following applies:

- a) tailoring of the safety activities in accordance with ISO 26262-2 has been planned and shows that the requirement does not apply, or
- b) a rationale is available that the non-compliance is acceptable and the rationale has been assessed in accordance with ISO 26262-2.

Information marked as a “NOTE” or “EXAMPLE” is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase. Given that certain requirements of a clause are ASIL-dependent or may be tailored, certain work products may not be needed as prerequisites.

“Further supporting information” is information that can be considered, but which in some cases is not required by ISO 26262 as a work product of a previous phase and which may be made available by external sources that are different from the persons or organizations responsible for the functional safety activities.

4.2 Interpretations of tables

Tables are normative or informative depending on their context. The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

- a) a consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
- b) an alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A

rationale shall be given that the selected combination of methods complies with the corresponding requirement.

NOTE A rationale based on the methods listed in the table is sufficient. However, this does not imply a bias for or against methods not listed in the table.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

- “++” indicates that the method is highly recommended for the identified ASIL;
- “+” indicates that the method is recommended for the identified ASIL;
- “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

4.3 ASIL-dependent requirements and recommendations

The requirements or recommendations of each subclause shall be complied with for ASIL A, B, C and D, if not stated otherwise. These requirements and recommendations refer to the ASIL of the safety goal. If ASIL decomposition has been performed at an earlier stage of development, in accordance with Clause 5 of this part of ISO 26262, the ASIL resulting from the decomposition shall be complied with.

If an ASIL is given in parentheses in ISO 26262, the corresponding subclause shall be considered as a recommendation rather than a requirement for this ASIL. This has no link with the parenthesis notation related to ASIL decomposition.

5 Requirements decomposition with respect to ASIL tailoring

5.1 Objectives

This clause provides rules and guidance for decomposing safety requirements into redundant safety requirements to allow ASIL tailoring at the next level of detail.

5.2 General

The ASIL of the safety goals of an item under development is propagated throughout the item's development process. Starting from safety goals, the safety requirements are derived and refined during the development phases. The ASIL, as an attribute of the safety goal, is inherited by each subsequent safety requirement. The functional and technical safety requirements are allocated to architectural elements, starting with preliminary architectural assumptions and ending with the hardware and software elements.

The method of ASIL tailoring during the design process is called "ASIL decomposition". During the allocation process, benefit can be obtained from architectural decisions including the existence of sufficiently independent architectural elements. This offers the opportunity:

- to implement safety requirements redundantly by these independent architectural elements, and
- to assign a potentially lower ASIL to these decomposed safety requirements.

If the architectural elements are not sufficiently independent, then the redundant requirements and the architectural elements inherit the initial ASIL.

NOTE 1 ASIL decomposition is an ASIL tailoring measure that can be applied to the functional, technical, hardware or software safety requirements of the item or element.

NOTE 2 As a basic rule, the application of ASIL decomposition requires redundancy of safety requirements allocated to architectural elements that are sufficiently independent.

NOTE 3 In the case of use of homogenous redundancy (e.g. by duplicated device or duplicated software) and with respect to systematic failures of hardware and software, the ASIL cannot be reduced unless an analysis of dependent failures provides evidence that sufficient independence exists or that the potential common causes lead to a safe state. Therefore, homogenous redundancy is in general not sufficient for reducing the ASIL due to the lack of independence between the elements.

NOTE 4 In general, ASIL decomposition does not apply to elements ensuring the channel selection or switching in multi-channel architectural designs.

In general, ASIL decomposition allows the apportioning of the ASIL of a safety requirement between several elements that ensure compliance with the same safety requirement addressing the same safety goal. ASIL decomposition between an intended functionality and its corresponding safety mechanism is allowed under certain conditions (see 5.4.7).

The requirements specific to the random hardware failures, including the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures (see ISO 26262-5) remain unchanged by ASIL decomposition.

5.3 Inputs to this clause

5.3.1 Prerequisites

The following information shall be available:

- the safety requirements at the level at which the ASIL decomposition is to be applied: system, or hardware, or software in accordance with ISO 26262-3:2011, 8.5.1, or ISO 26262-4:2011, 6.5.1, or ISO 26262-5:2011, 6.5.1 or ISO 26262-6:2011, 6.5.1; and
- the architectural information at the level at which the ASIL decomposition is to be applied: system, or hardware, or software in accordance with ISO 26262-4:2011, 7.5.2, or ISO 26262-5:2011, 7.5.1, or ISO 26262-6:2011, 7.5.1.

5.3.2 Further supporting information

The following information can be considered:

- item definition (see ISO 26262-3:2011, 5.5); and
- safety goals (see ISO 26262-3:2011, 7.5.2).

5.4 Requirements and recommendations

5.4.1 If ASIL decomposition is applied, all the requirements within this clause shall be complied with.

5.4.2 ASIL decomposition shall be performed by considering each initial safety requirement individually.

NOTE Several safety requirements can be allocated to the same independent elements as the result of ASIL decompositions of different initial safety requirements.

5.4.3 The initial safety requirement shall be decomposed to redundant safety requirements implemented by sufficiently independent elements.

5.4.4 Each decomposed safety requirement shall comply with the initial safety requirement by itself.

NOTE This requirement provides redundancy by definition.

5.4.5 The requirements on the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures shall remain unchanged by ASIL decomposition in accordance with ISO 26262-5.

5.4.6 If ASIL decomposition is applied at the software level, sufficient independence between the elements implementing the decomposed requirements shall be checked at the system level and appropriate measures shall be taken at the software level, or hardware level, or system level to achieve sufficient independence.

5.4.7 If ASIL decomposition of an initial safety requirement results in the allocation of decomposed requirements to the intended functionality and an associated safety mechanism, then:

- a) the associated safety mechanism should be assigned the highest decomposed ASIL;

NOTE In general, the safety mechanisms have a lower complexity and lower size than the intended functionality.

- b) a safety requirement shall be allocated to the intended functionality and implemented applying the corresponding decomposed ASIL.

NOTE If the decomposition scheme ASIL $x(x) + QM(x)$ is chosen, then $QM(x)$ means that the quality management system can be sufficient to develop element(s) that implement the safety requirement allocated to the intended functionality. $QM(x)$ also means that the quality management system can support the rationale for the independence between the intended functionality and the safety mechanism.

5.4.8 If the violation of an initial safety requirement cannot be prevented by switching off the element, then adequate availability of the sufficiently independent elements implementing the decomposed safety requirements shall be shown.

5.4.9 When applying ASIL decomposition to a safety requirement, then:

- a) ASIL decomposition shall be applied in accordance with 5.4.10;
- b) ASIL decomposition may be applied more than once;
- c) each decomposed ASIL shall be marked by giving the ASIL of the safety goal in parenthesis.

EXAMPLE If an ASIL D requirement is decomposed into one ASIL C requirement and one ASIL A requirement, then these are marked as "ASIL C(D)" and "ASIL A(D)". If the ASIL C(D) requirement is further decomposed into one ASIL B requirement and one ASIL A requirement, then these are also marked with the ASIL of the safety goal as "ASIL B(D)" and "ASIL A(D)".

5.4.10 One of the following decomposition schemes outlined below shall be chosen in accordance with the ASIL before decomposition (as shown in Figure 2), or a scheme resulting in higher ASILs may be used.

NOTE The step from one level of the selected decomposition scheme to the lower next level defines one decomposition of the ASIL.

- a) An ASIL D requirement shall be decomposed as one of the following:

- 1) one ASIL C(D) requirement and one ASIL A(D) requirement; or
- 2) one ASIL B(D) requirement and one ASIL B(D) requirement; or
- 3) one ASIL D(D) requirement and one QM(D) requirement.

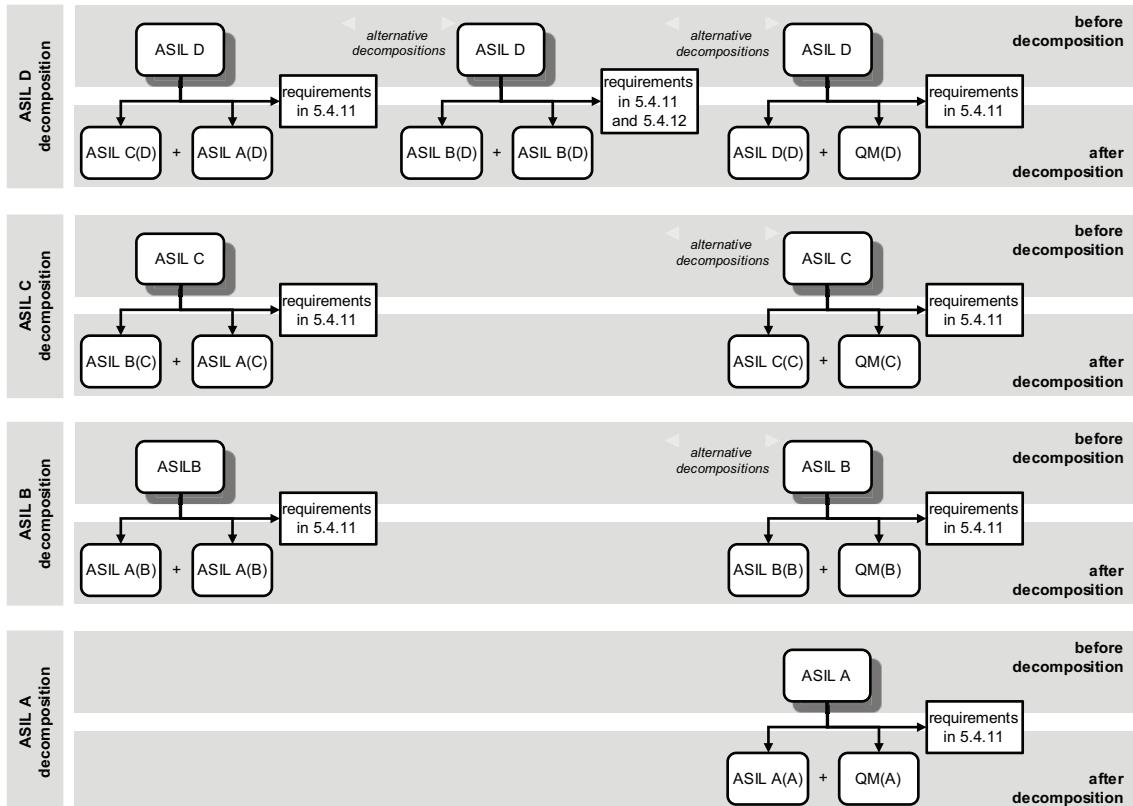
- b) An ASIL C requirement shall be decomposed as one of the following:

- 1) one ASIL B(C) requirement and one ASIL A(C) requirement; or
- 2) one ASIL C(C) requirement and one QM(C) requirement.

- c) An ASIL B requirement shall be decomposed as one of the following:

- 1) one ASIL A(B) requirement and one ASIL A(B) requirement; or

- 2) one ASIL B(B) requirement and one QM(B) requirement.
- d) An ASIL A shall not be further decomposed, except, if needed, as one ASIL A(A) requirement and one QM(A) requirement.



EXAMPLE The cases described in 5.4.7, where QM is assigned to the intended functionality and an ASIL equal to the initial ASIL is assigned to its associated safety mechanism, are shown in the rightmost column.

NOTE The uppermost shadowed box of each decomposition step represents the ASIL before decomposition.

Figure 2 — ASIL decomposition schemes

5.4.11 When using any of the decomposition schemes given in 5.4.10, then:

- confirmation measures in accordance with ISO 26262-2:2011, 6.4.7, shall be applied in compliance with the ASIL of the safety goal;
- evidence for sufficient independence of the elements after decomposition shall be made available.

NOTE The elements are sufficiently independent if the analysis of dependent failures (see Clause 7 of this part of ISO 26262) does not find a cause of dependent failures that can lead to the violation of a safety requirement before decomposition, or if each identified cause of dependent failures is controlled by an adequate safety measure according to the ASIL of the safety goal.

5.4.12 When using the decomposition scheme for ASIL D given in 5.4.10 a) 2), then:

- decomposed safety requirements shall be specified in accordance with ASIL C requirements of ISO 26262-8:2011, Clause 6;

NOTE The more formalized notation required for ASIL C compared to ASIL B increases the avoidance of systematic failures and decreases dependencies between the two ASIL B(D) implementations.

- b) if the same software tools are used for the development of the decomposed elements, then these software tools shall be considered as software tools for developing ASIL D items or elements, in accordance with the confidence in the use of software tools in ISO 26262-8.

5.4.13 Development of the decomposed elements at the system level and at the software level shall be performed, as a minimum, in accordance with the ASIL requirements (after decomposition) of ISO 26262-4 and ISO 26262-6. Development of the decomposed elements at the hardware level shall be performed, as a minimum, in accordance with the ASIL requirements (after decomposition) of ISO 26262-5, except for the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures (see 5.4.5).

5.4.14 At each level of the design process at which decomposition is applied, the corresponding integration activities of the decomposed elements and subsequent activities shall be applied in accordance with the requirements of the ASIL before decomposition.

5.5 Work products

5.5.1 Update of architectural information, resulting from 5.4.

5.5.2 Update of ASIL as attribute of safety requirements and elements, resulting from 5.4.

6 Criteria for coexistence of elements

6.1 Objectives

This clause provides criteria for the coexistence within the same element of:

- safety-related sub-elements with sub-elements that have no ASIL assigned; and
- safety-related sub-elements that have different ASILs assigned.

6.2 General

By default, when an element is composed of several sub-elements, each of those sub-elements is developed in accordance with the measures corresponding to the highest ASIL applicable to the element, i.e. the highest ASIL of the safety requirements allocated to the element (see ISO 26262-4:2011, 7.4.2.3).

In the case of the coexistence of sub-elements that have different ASILs assigned or the coexistence of sub-elements that have no ASIL assigned with safety-related ones, it can be beneficial to avoid raising the ASIL for some of them to the ASIL of the element. For this purpose, this clause provides guidance for determining the ASIL of sub-elements of an element. This clause is based on the analysis of interference of a sub-element with the other sub-elements of an element.

Interference is the presence of cascading failures from a sub-element with no ASIL assigned, or a lower ASIL assigned, to a sub-element with a higher ASIL assigned leading to the violation of a safety requirement of the element (see ISO 26262-1:2011, definitions 1.13 and 1.49).

When determining the ASIL of sub-elements of an element, the rationale for freedom from interference is supported by analyses of dependent failures focused on cascading failures (see Clause 7 of this part of ISO 26262).

6.3 Inputs to this clause

6.3.1 Prerequisites

The following information shall be available:

- the safety requirements at the level at which the analysis is to be performed: system, or hardware, or software in accordance with ISO 26262-3:2011, 8.5.1, or ISO 26262-4:2011, 6.5.1, or ISO 26262-5:2011, 6.5.1, or ISO 26262-6:2011, 6.5.1; and
- the architectural information of the element at the level at which the analysis is to be performed: system, or hardware, or software in accordance with ISO 26262-4:2011, 7.5.2, or ISO 26262-5:2011, 7.5.1, or ISO 26262-6:2011, 7.5.1.

6.3.2 Further supporting information

None.

6.4 Requirements and recommendations

6.4.1 This clause may be applied at any refinement step during the design process, in parallel with the allocation of the safety requirements to the elements and sub-elements of an architecture, typically during the subphases of system design, or hardware design, or software architectural design, in accordance with ISO 26262-4, or ISO 26262-5, or ISO 26262-6.

6.4.2 The safety requirements shall be allocated to the sub-elements of the element before applying this clause.

NOTE The allocation of safety requirements to the sub-elements results in safety-related sub-elements and sub-elements that have no ASIL assigned.

6.4.3 The following shall be considered during the analysis of an element:

- a) each safety requirement allocated to the element; and
- b) each sub-element that is part of the element.

6.4.4 If a sub-element with no ASIL assigned and safety-related sub-elements coexist in the same element, then the sub-element with no ASIL assigned shall only be treated as a QM sub-element if evidence is made available that it cannot violate, directly or indirectly, any safety requirement allocated to the element, i.e. it cannot interfere with any safety-related sub-element of the element.

NOTE 1 This means that cascading failures from this sub-element to the safety-related elements are absent.

NOTE 2 This can be achieved by design precautions such as those concerning the data flow and control flow for software, or the I/O signals and control lines for hardware.

Otherwise, this sub-element shall be assigned the highest ASIL of the coexisting safety-related sub-elements for which evidence of freedom from interference is not made available.

6.4.5 If safety-related sub-elements with different ASILs, including QM(x) (see 5.4.10), coexist in the same element, then a sub-element shall only be treated as a sub-element with a lower ASIL assigned if evidence is made available that, for each safety requirement allocated to the element, it cannot interfere with any sub-element with a higher ASIL assigned. Otherwise, this sub-element shall be assigned the highest ASIL of the coexisting safety-related sub-elements for which evidence of freedom from interference is not made available.

6.5 Work products

6.5.1 Update of ASIL as attribute of sub-elements of elements, resulting from 6.4.

7 Analysis of dependent failures

7.1 Objectives

The analysis of dependent failures aims to identify the single events or single causes that could bypass or invalidate a required independence or freedom from interference between given elements and violate a safety requirement or a safety goal.

7.2 General

The analysis of dependent failures considers architectural features such as:

- similar and dissimilar redundant elements;
- different functions implemented with identical software or hardware elements;
- functions and their respective safety mechanisms;
- partitions of functions or software elements;
- physical distance between hardware elements, with or without barrier;
- common external resources.

According to the definitions given in ISO 26262-1, independence is threatened by common cause failures and cascading failures, while freedom from interference is only threatened by cascading failures.

EXAMPLE 1 A high intensity electromagnetic field that causes different electronic devices to fail in a way that depends on design and use is an example of a common cause failure. Biased vehicle speed information that affects the behaviour of one or more vehicle functions is an example of cascading failures.

Dependent failures can manifest themselves simultaneously, or within a sufficiently short time interval, to have the effect of simultaneous failures.

EXAMPLE 2 A monitor designed to detect anomalous behaviour of a function can be rendered inoperative some time before the monitored function fails if both the monitor and the monitored function are subjected to the same event or cause.

7.3 Inputs to this clause

7.3.1 Prerequisites

The following information shall be available:

- the independence requirements at the level at which they are applied: system, or hardware, or software in accordance with ISO 26262-3:2011, 8.5.1, or ISO 26262-4:2011, 6.5.1, or ISO 26262-5:2011, 6.5.1, or ISO 26262-6:2011, 6.5.1;
- the freedom from interference requirements at the level at which they are applied: system, or hardware, or software in accordance with ISO 26262-3:2011, 8.5.1, or ISO 26262-4:2011, 6.5.1, or ISO 26262-5:2011, 6.5.1, or ISO 26262-6:2011, 6.5.1; and

- the architectural information at the level at which the independence or freedom from interference requirements are to be applied: system, or hardware, or software in accordance with ISO 26262-4:2011, 7.5.2, or ISO 26262-5:2011, 7.5.1, or ISO 26262-6:2011, 7.5.1.

NOTE The architectural information is used to determine the boundaries of the analyses of dependent failures.

7.3.2 Further supporting information

None.

7.4 Requirements and recommendations

7.4.1 The potential for dependent failures shall be identified from the results of safety analyses in accordance with Clause 8.

NOTE 1 Both systematic failures and random hardware failures have the potential to be dependent failures.

NOTE 2 The identification of potential for dependent failures can be based on deductive analyses: examination of cut sets or repeated identical events of an FTA can indicate potential for dependent failures.

NOTE 3 The identification can also be supported by inductive analyses: similar parts or components with similar failure modes that appear several times in an FMEA can give additional information about the potential for dependent failures.

7.4.2 Each identified potential for dependent failures shall be evaluated to determine its plausibility, i.e. if a reasonably foreseeable cause exists which leads to the dependent failure and consequently violates a required independence or freedom from interference between given elements.

NOTE When quantification of random hardware failures is required, as for the evaluation of the safety goal violations due to random hardware failures (see ISO 26262-5), the contribution of common cause failures is estimated on a qualitative basis because no general and sufficiently reliable method exists for quantifying such failures.

7.4.3 This evaluation shall consider the operational situations as well as the different operating modes of the item or element being analyzed.

7.4.4 This evaluation shall consider the following topics as applicable:

NOTE 1 The evaluation of the potential dependent failures plausibility can be supported by appropriate checklists, e.g. checklists based on field experience. The checklists provide the analysts with representative examples of root causes and coupling factors such as: same design, same process, same component, same interface, proximity. IEC 61508 provides information that can be used as a basis to establish such checklists.

NOTE 2 This evaluation can also be supported by the adherence to process guidelines which are intended to prevent the introduction of root causes and coupling factors that could lead to dependent failures.

a) random hardware failures;

EXAMPLE Failures of common blocks such as clock, test logic and internal voltage regulators in large scale integrated circuits (microcontrollers, ASICs, etc.).

b) development faults;

EXAMPLE Requirement faults, design faults, implementation faults, faults resulting from the use of new technologies and faults introduced when making modifications.

c) manufacturing faults;

EXAMPLE Faults related to processes, procedures and training; faults in control plans and in monitoring special characteristics; faults related to software flashing and end-of-line programming.

d) installation faults;

EXAMPLE Faults related to wiring harness routing; faults related to the inter-changeability of parts; failures of adjacent items or elements.

e) repair faults;

EXAMPLE Faults related to processes, procedures and training; faults related to trouble shooting; faults related to the inter-changeability of parts and faults due to backward incompatibility.

f) environmental factors;

EXAMPLE Temperature, vibration, pressure, humidity / condensation, pollution, corrosion, contamination, EMC.

g) failures of common external resources; and

EXAMPLE Power supply, input data, inter-system data bus and communication.

h) stress due to specific situations.

EXAMPLE Wear, ageing.

7.4.5 Rationale for the plausibility of dependent failures and their impact shall be made available.

NOTE Plausible dependent failures are those for which the evaluation as given in 7.4.2 has revealed a reasonably foreseeable cause.

7.4.6 Measures for the resolution of plausible dependent failures shall be specified during the development phase, in accordance with the change management in ISO 26262-8.

7.4.7 Measures for the resolution of plausible dependent failures shall include the measures for preventing their root causes, or for controlling their effects, or for reducing the coupling factors.

EXAMPLE Diversity is a measure that can be used to prevent, reduce or detect common cause failures.

7.5 Work products

7.5.1 Analysis of dependent failures, resulting from 7.4.

8 Safety analyses

8.1 Objectives

The objective of safety analyses is to examine the consequences of faults and failures on the functions, behaviour and design of items and elements. Safety analyses also provide information on conditions and causes that could lead to the violation of a safety goal or safety requirement.

Additionally, the safety analyses also contribute to the identification of new functional or non-functional hazards not previously identified during the hazard analysis and risk assessment.

8.2 General

The scope of the safety analyses includes:

- the validation of safety goals and safety concepts;
- the verification of safety concepts and safety requirements;
- the identification of conditions and causes, including faults and failures, that could lead to the violation of a safety goal or safety requirement;

- the identification of additional requirements for detection of faults or failures;
- the determination of the required responses (actions/measures) to detected faults or failures; and
- the identification of additional requirements for verifying that the safety goals or safety requirements are complied with, including safety-related vehicle testing.

Safety analyses are performed at the appropriate level of abstraction during the concept and product development phases. Quantitative analysis methods predict the frequency of failures while qualitative analysis methods identify failures but do not predict the frequency of failures. Both types of analysis methods depend upon a knowledge of the relevant fault types and fault models.

Qualitative analysis methods include:

- qualitative FMEA at system, design or process level;
- qualitative FTA;
- HAZOP;
- qualitative ETA.

NOTE 1 The qualitative analysis methods listed above can be applied to software where no more appropriate software-specific analysis methods exist.

Quantitative safety analyses complement qualitative safety analyses. They are used to verify a hardware design against defined targets for the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures (see ISO 26262-5). Quantitative safety analyses require additional knowledge of the quantitative failure rates of the hardware elements.

Quantitative analysis methods include:

- quantitative FMEA;
- quantitative FTA;
- quantitative ETA;
- Markov models;
- reliability block diagrams.

NOTE 2 The quantitative analysis methods only address random hardware failures. These analysis methods are not applied to systematic failures in ISO 26262.

Another criteria for the classification of safety analyses is given by the way they are conducted:

- inductive analysis methods are bottom-up methods that start from known causes and forecast unknown effects;
- deductive analysis methods are top-down methods that start from known effects and seek unknown causes.

EXAMPLE System, design and process FMEAs, ETA and Markov modelling are inductive analysis methods. FTA and reliability block diagrams are deductive analysis methods.

8.3 Inputs to this clause

8.3.1 Prerequisites

The following information shall be available:

- the safety requirements at the level at which the safety analysis is to be performed: system, or hardware, or software in accordance with ISO 26262-3:2011, 8.5.1, or ISO 26262-4:2011, 6.5.1, or ISO 26262-5:2011, 6.5.1, or ISO 26262-6:2011, 6.5.1;
- the architectural information of the element at the level at which the safety analysis is to be performed: system, or hardware, or software in accordance with ISO 26262-4:2011, 7.5.2, or ISO 26262-5:2011, 7.5.1, or ISO 26262-6:2011, 7.5.1; and

NOTE 1 The architectural information is used to determine the boundaries of the safety analyses.

- the safety plan in accordance with ISO 26262-2:2011, 6.5.1

NOTE 2 The safety plan contains the objective of the safety analyses.

8.3.2 Further supporting information

The following information can be considered:

- fault models (from external sources).

8.4 Requirements and recommendations

8.4.1 The safety analyses shall be performed in accordance with appropriate standards or guidelines.

8.4.2 The results of the safety analyses shall indicate if the respective safety goals or safety requirements are complied with or not.

8.4.3 If a safety goal or a safety requirement is not complied with, the results of the safety analyses shall be used for deriving prevention, or detection, or effect mitigation measures regarding the faults or failures causing the violation.

8.4.4 The measures derived from the safety analyses shall be implemented as part of the product development at the system level, or at the hardware level, or at the software level, respectively in accordance with ISO 26262-4, or ISO 26262-5, or ISO 26262-6.

8.4.5 Newly identified hazards by safety analyses during product development not covered in a safety goal shall be introduced and evaluated in the hazard analysis and risk assessment in accordance with the change management in ISO 26262-8.

8.4.6 The fault models used for the safety analyses shall be consistent with the appropriate development subphases, e.g. hardware design, evaluation of the hardware architectural metrics and evaluation of safety goal violations due to random hardware failures in ISO 26262-5.

8.4.7 The need for additional safety-related test cases shall be determined by using the fault models and the results of the safety analyses.

8.4.8 The results of the safety analyses shall be verified in accordance with ISO 26262-8.

8.4.9 The qualitative safety analyses shall include:

- a) a systematic identification of faults or failures that could lead to the violation of safety goals or safety requirements, originating in:

- the item or element itself; or
 - the interaction of the item or element with other items or elements; or
 - the usage of the item or element;
- b) the evaluation of the consequences of each identified fault to determine the potential to violate safety goals or safety requirements;
- c) the identification of the causes of each identified fault; and
- d) the identification, or the support for the identification, of potential safety concept weaknesses, including the ineffectiveness of safety mechanisms in handling anomalies such as latent faults, multiple-point faults, common cause failures and cascading failures.

NOTE The examination of interactions with other items or elements, within and outside the item, is done in order to assess the degree of independence or interference.

8.4.10 If quantitative safety analyses are applicable, then they shall include:

- a) the quantitative data to support the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures (see ISO 26262-5);
- b) a systematic identification of faults or failures that could lead to violation of safety goals or safety requirements;
- c) the evaluation and ranking of the potential safety concept weaknesses, including the ineffectiveness of safety mechanisms; and
- d) the diagnostic test interval, the emergency operation interval, and the time between fault detection and repair.

8.4.11 If qualitative safety analyses are applied to support the compliance with quantitative requirements, the level of detail within these safety analyses shall be chosen appropriately.

8.5 Work products

8.5.1 Safety analyses, resulting from 8.4.

Annex A (informative)

Overview of and document flow of Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses

Table A.1 provides an overview of objectives, prerequisites and work products of ASIL-oriented and safety-oriented analyses.

Table A.1 — Overview of ASIL-oriented and safety-oriented analyses

Clause	Objectives	Prerequisites	Work products
5 Requirements decomposition with respect to ASIL tailoring	This clause provides rules and guidance for decomposing safety requirements into redundant safety requirements to allow ASIL tailoring at the next level of detail.	The safety requirements at the level at which the ASIL decomposition is to be applied: system, or hardware, or software level. The architectural information at the level at which the ASIL decomposition is to be applied: system, or hardware, or software level.	5.5.1 Update of architectural information 5.5.2 Update of ASIL as attribute of safety requirements and elements
6 Criteria for coexistence of elements	This clause provides criteria for the coexistence within the same element of: — safety-related sub-elements with sub-elements that have no ASIL assigned; and — safety-related sub-elements that have different ASILs assigned.	The safety requirements at the level at which the analysis is to be performed: system, or hardware, or software. The architectural information of the element at the level at which the analysis is to be performed: system, or hardware, or software.	6.5.1 Update of ASIL as attribute of sub-elements of elements
7 Analysis of dependent failures	The analysis of dependent failures aims to identify the single events or single causes that could bypass or invalidate a required independence or freedom from interference between given elements and violate a safety requirement or a safety goal.	The independence requirements at the level at which they are applied: system, or hardware, or software. The freedom from interference requirements at the level at which they are applied: system, or hardware, or software. The architectural information at the level at which the independence or freedom from interference requirements are to be applied: system, or hardware, or software.	7.5.1 Analysis of dependent failures
8 Safety analyses	The objective of safety analyses is to examine the consequences of faults and failures on the functions, behaviour and design of items and elements. Safety analyses also provide information on conditions and causes that could lead to the violation of a safety goal or safety requirement. Additionally, the safety analyses also contribute to the identification of new functional or non-functional hazards not previously considered during hazard analysis and risk assessment.	The safety requirements at the level at which the safety analysis is to be performed: system, or hardware, or software. The architectural information of the element at the level at which the safety analysis is to be performed: system, or hardware, or software. The safety plan.	8.5.1 Safety analyses

Bibliography

- [1] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*

ICS 43.040.10

Price based on 16 pages

INTERNATIONAL
STANDARD

ISO
26262-10

First edition
2012-08-01

**Road vehicles — Functional safety —
Part 10:
Guideline on ISO 26262**

*Véhicules routiers — Sécurité fonctionnelle —
Partie 10: Lignes directrices relatives à l'ISO 26262*



Reference number
ISO 26262-10:2012(E)

© ISO 2012



COPYRIGHT PROTECTED DOCUMENT

© ISO 2012

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction.....	v
1 Scope	1
2 Normative references.....	1
3 Terms, definitions and abbreviated terms	2
4 Key concepts of ISO 26262.....	2
4.1 Functional safety for automotive systems (relationship with IEC 61508)	2
4.2 Item, system, element, component, hardware part and software unit.....	4
4.3 Relationship between faults, errors and failures	5
5 Selected topics regarding safety management.....	6
5.1 Work product	6
5.2 Confirmation measures	6
5.3 Understanding of safety cases	9
6 Concept phase and system development.....	10
6.1 General	10
6.2 Example of hazard analysis and risk assessment.....	10
6.3 An observation regarding controllability classification	11
6.4 External measures.....	12
6.5 Example of combining safety goals	13
7 Safety process requirement structure - Flow and sequence of safety requirements.....	14
8 Concerning hardware development	17
8.1 The classification of random hardware faults	17
8.2 Example of residual failure rate and local single-point fault metric evaluation	22
8.3 Further explanation concerning hardware	34
9 Safety element out of context	36
9.1 Safety element out of context development	36
9.2 Use cases	37
10 An example of proven in use argument	45
10.1 General	45
10.2 Item definition and definition of the proven in use candidate	46
10.3 Change analysis	46
10.4 Target values for proven in use	46
11 Concerning ASIL decomposition.....	47
11.1 Objective of ASIL decomposition	47
11.2 Description of ASIL decomposition	47
11.3 An example of ASIL decomposition	47
Annex A (informative) ISO 26262 and microcontrollers	51
Annex B (informative) Fault tree construction and applications	73
Bibliography.....	89

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 26262-10 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 26262 consists of the following parts, under the general title *Road vehicles — Functional safety*:

- *Part 1: Vocabulary*
- *Part 2: Management of functional safety*
- *Part 3: Concept phase*
- *Part 4: Product development at the system level*
- *Part 5: Product development at the hardware level*
- *Part 6: Product development at the software level*
- *Part 7: Production and operation*
- *Part 8: Supporting processes*
- *Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*
- *Part 10: Guideline on ISO 26262*

Introduction

ISO 26262 is the adaptation of IEC 61508 to comply with needs specific to the application sector 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 of future automobile development. New functionalities not only in areas such as driver assistance, propulsion, in vehicle dynamics control and active and passive safety systems increasingly touch the domain of system safety engineering. Development and integration of these functionalities will strengthen the need for safe system development processes and the need to provide evidence that all reasonable system safety objectives are satisfied.

With the trend of increasing technological complexity, software content and mechatronic implementation, there are increasing risks from systematic failures and random hardware failures. ISO 26262 includes guidance to avoid these risks by providing appropriate requirements and processes.

System safety is achieved through a number of safety measures, which are implemented in a variety of technologies (e.g. mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic) and applied at the various levels of the development process. Although ISO 26262 is concerned with functional safety of E/E systems, it provides a framework within which safety-related systems based on other technologies can be considered. ISO 26262:

- a) provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases;
- b) provides an automotive-specific risk-based approach to determine integrity levels [Automotive Safety Integrity Levels (ASIL)];
- c) uses ASILs to specify applicable requirements of ISO 26262 so as to avoid unreasonable residual risk;
- d) provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved;
- e) provides requirements for relations with suppliers.

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 by the management processes.

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

Figure 1 shows the overall structure of this edition 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 between ISO 26262-3, ISO 26262-4, ISO 26262-5, ISO 26262-6 and ISO 26262-7;
- 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.

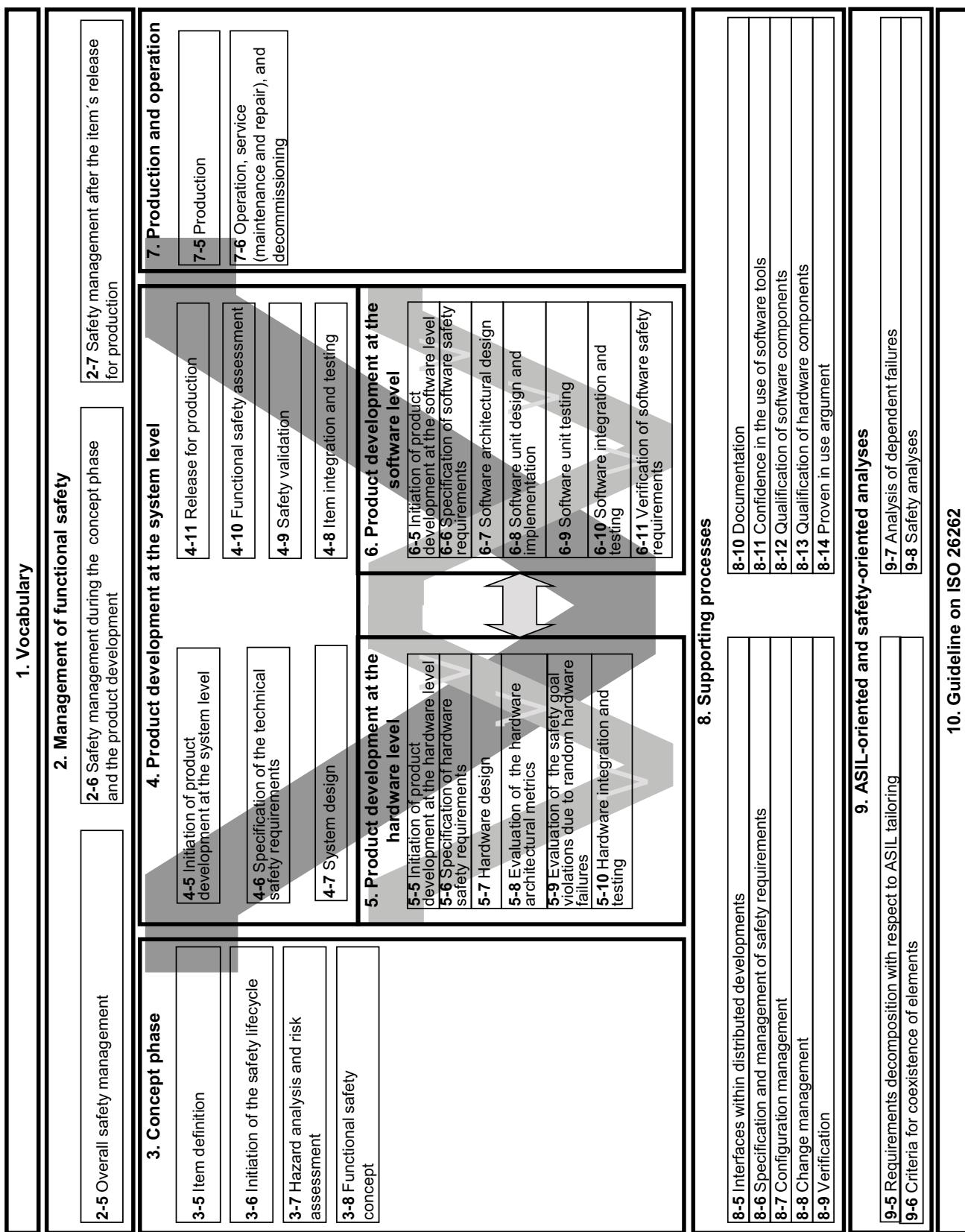


Figure 1 — Overview of ISO 26262

Road vehicles — Functional safety —

Part 10: Guideline on ISO 26262

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 passenger cars with a maximum gross vehicle mass up to 3 500 kg. ISO 26262 does not address unique E/E systems in special purpose vehicles such as vehicles designed for drivers with disabilities.

Systems and their components released for production, or systems and their components already under development prior to the publication date of ISO 26262, are exempted from the scope. For further development or alterations based on systems and their components released for production prior to the publication of ISO 26262, only the modifications will be developed in accordance with ISO 26262.

ISO 26262 addresses possible hazards caused by malfunctioning behaviour of E/E safety-related 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 E/E safety-related systems.

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

This part of ISO 26262 provides an overview of ISO 26262, as well as giving additional explanations, and is intended to enhance the understanding of the other parts of ISO 26262. It has an informative character only and describes the general concepts of ISO 26262 in order to facilitate comprehension. The explanation expands from general concepts to specific contents.

In the case of inconsistencies between this part of ISO 26262 and another part of ISO 26262, the requirements, recommendations and information specified in the other part of ISO 26262 apply.

2 Normative references

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

ISO 26262-1:2011, *Road vehicles — Functional safety — Part 1: Vocabulary*

ISO 26262-2:2011, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-3:2011, *Road vehicles — Functional safety — Part 3: Concept phase*

ISO 26262-4:2011, *Road vehicles — Functional safety — Part 4: Product development at the system level*

ISO 26262-5:2011, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

ISO 26262-6:2011, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-7:2011, *Road vehicles — Functional safety — Part 7: Production and operation*

ISO 26262-8:2011, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 26262-9:2011, *Road vehicles — Functional safety — Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses*

3 Terms, definitions and abbreviated terms

For the purposes of this document, the terms, definitions and abbreviated terms given in ISO 26262-1:2011 apply.

4 Key concepts of ISO 26262

4.1 Functional safety for automotive systems (relationship with IEC 61508)

IEC 61508, *Functional safety of electrical/electronic/programmable electronic safety-related systems*, is designated by IEC as a generic standard and a basic safety publication. This means that industry sectors will base their own standards for functional safety on the requirements of IEC 61508.

In the automotive industry, there are a number of issues with applying IEC 61508 directly. Some of these issues and corresponding differences in ISO 26262 are described below.

IEC 61508 is based upon the model of “equipment under control”, for example an industrial plant that has an associated control system as follows:

- a) A hazard analysis identifies the hazards associated with the equipment under control (including the equipment control system), to which risk reduction measures will be applied. This can be achieved through E/E/PE systems, or other technology safety-related systems (e.g. a safety valve), or external measures (e.g. a physical containment of the plant). ISO 26262 contains a normative automotive scheme for hazard classification based on severity, probability of exposure and controllability.
- b) Risk reduction allocated to E/E/PE systems is achieved through safety functions, which are designated as such. These safety functions are either part of a separate protection system or can be incorporated into the plant control. It is not always possible to make this distinction in automotive systems. The safety of a vehicle depends on the behaviour of the control systems themselves.

ISO 26262 uses the concept of safety goals and a safety concept as follows:

- a hazard analysis and risk assessment identifies hazards and hazardous events that need to be prevented, mitigated or controlled;
- a safety goal is formulated for each hazardous event;
- an Automotive Safety Integrity Level (ASIL) is associated with each safety goal;
- the functional safety concept is a statement of the functionality to achieve the safety goal(s);
- the technical safety concept is a statement of how this functionality is implemented on the system level by hardware and software; and
- software safety requirements and hardware safety requirements state the specific safety requirements which will be implemented as part of the software and hardware design.

EXAMPLE

- The airbag system: one of the hazards is unintended deployment.
- An associated safety goal is that the airbag does not deploy unless a crash occurs that requires the deployment.

- The functional safety concept can specify a redundant function to detect whether the vehicle is in a collision.
- The technical safety concept can specify the implementation of two independent accelerometers with different axial orientations and two independent firing circuits. The squib deploys if both are closed.

IEC 61508 is aimed at singular or low volume systems. The system is built and tested, then installed on the plant, and then safety validation is performed. For mass-market systems such as road vehicles, safety validation is performed before the release for volume (series) production. Therefore, the order of lifecycle activities in ISO 26262 is different. Related to this, ISO 26262-7 addresses requirements for production. These are not covered in IEC 61508.

IEC 61508 does not address specific requirements for managing development across multiple organizations and supply chains, whereas ISO 26262 addresses explicitly the issue, including the Development Interface Agreement (DIA) [see ISO 26262-8:2011, Clause 5 (Interfaces within distributed developments)], because automotive systems are produced by one or more suppliers of the customer, e.g. the vehicle manufacturer, the supplier of the customer, or the customer.

IEC 61508 does not contain normative requirements for hazard classification. ISO 26262 contains an automotive scheme for hazard classification. This scheme recognizes that a hazard in an automotive system does not necessarily lead to an accident. The outcome will depend on whether the persons at risk are actually exposed to the hazard in the situation in which it occurs, and whether they are able to take steps to control the outcome of the hazard. An example of this concept applied to a failure which affects the controllability of a moving vehicle is given in Figure 2.

NOTE This concept is intended only to demonstrate that there is not necessarily a direct correlation between a failure occurring and the accident. It is not a representation of the hazard analysis and risk assessment process, although the parameters evaluated in this process are related to the probabilities of the state transitions shown in the figure.

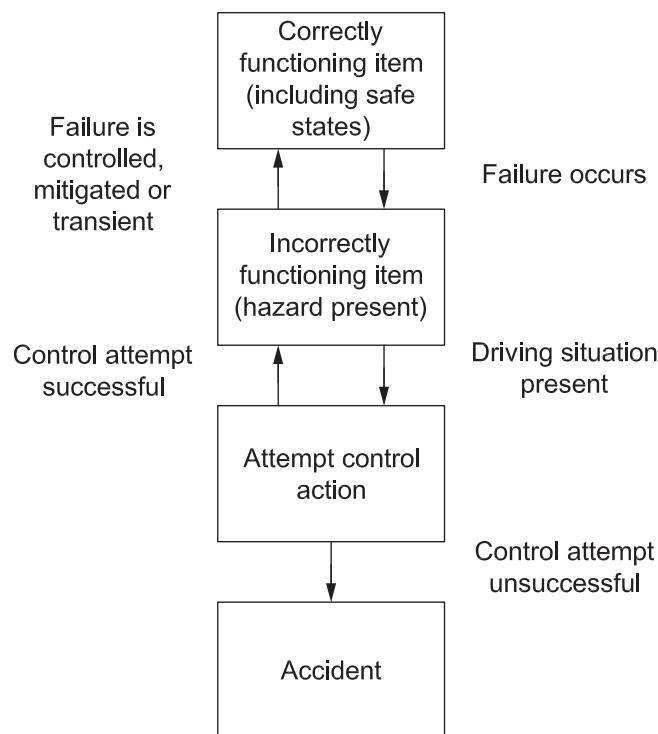


Figure 2 — State machine model of automotive risk

The requirements for hardware development (ISO 26262-5) and software development (ISO 26262-6) are adapted for the state-of-the-art in the automotive industry. Specifically, ISO 26262-6 contains requirements concerned with model-based development; IEC 61508 prescribes the application of specific methods. A detailed rationale for the use of any alternative method has to be provided. For the methods listed in

ISO 26262, specific goals are provided. To achieve these goals, the provided methods can be applied, or a rationale that alternative methods can also achieve the goal is provided.

Safety requirements in ISO 26262 are assigned an ASIL (Automotive Safety Integrity Level) rather than a SIL (Safety Integrity Level). The main motivation for this is that the SIL in IEC 61508 is stated in probabilistic terms (see IEC 61508-1:2010, Table 3). IEC 61508 states: "It is accepted that only with respect to the hardware safety integrity will it be possible to quantify and apply reliability prediction techniques in assessing whether the target failure measures have been met. Qualitative techniques and judgements have to be made with respect to the precautions necessary to meet the target failure measures with respect to the systematic safety integrity." An ASIL is not based on this probabilistic requirement concerning the occurrence of the hazard; however, there are probabilistic targets associated with compliance to the requirements of an ASIL.

4.2 Item, system, element, component, hardware part and software unit

The terms item, system, element, component, hardware part, and software unit are defined in ISO 26262-1. Figure 3 shows the relationship of item, system, component, hardware part and software unit. Figure 4 shows an example of item dissolution. A divisible element can be labelled as a system, a subsystem or a component. A divisible element that meets the criteria of a system can be labelled as a system or subsystem. The term subsystem is used when it is important to emphasize that the element is part of a larger system. A component is a non-system-level, logically and technically separable element. Often the term component is applied to an element that is only comprised of parts and units, but can also be applied to an element comprised of lower-level elements from a specific technology area, e.g. electrical/electronic technology (see Figure 4).

EXAMPLE In the case of a microcontroller or ASIC, the following partitioning can be used: the whole microcontroller is a component, the processing unit (e.g. a CPU) is a part, the registers inside the processing unit (e.g. the CPU register bank) is a sub-part. In the case of microcontroller (MCU) analyses, a higher level of detail in the partitioning could be needed; to aid in this purpose, it is possible to partition a part into sub-parts which can be further divided into basic/elementary sub-parts.

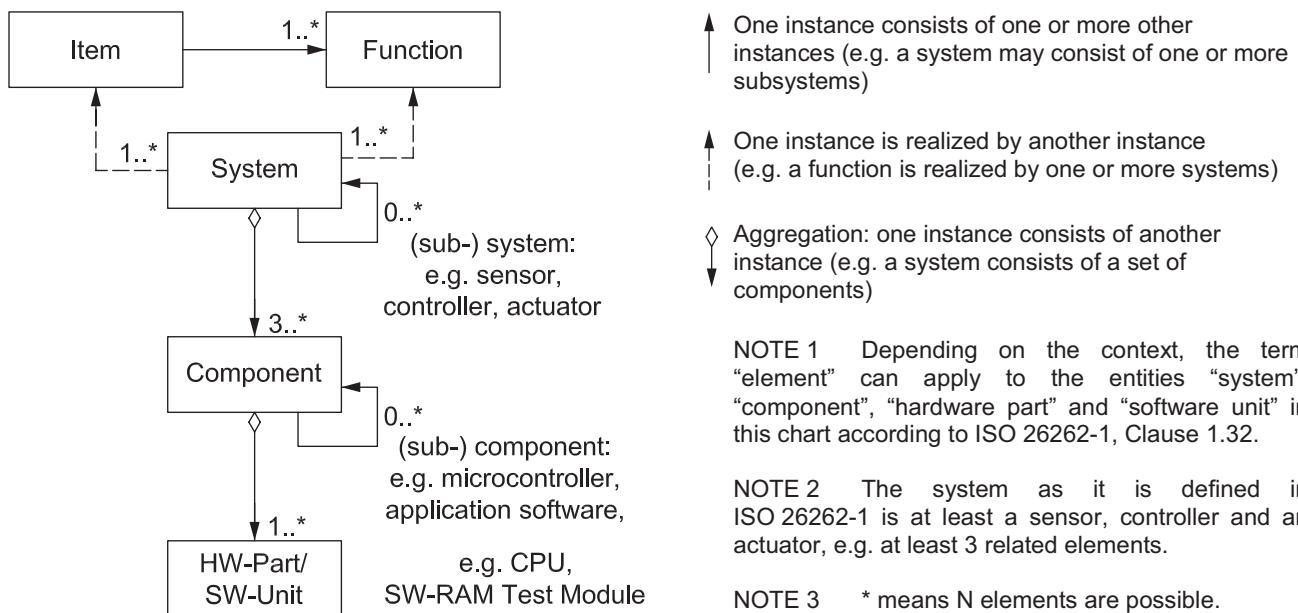


Figure 3 — Relationship of item, system, component, hardware part and software unit

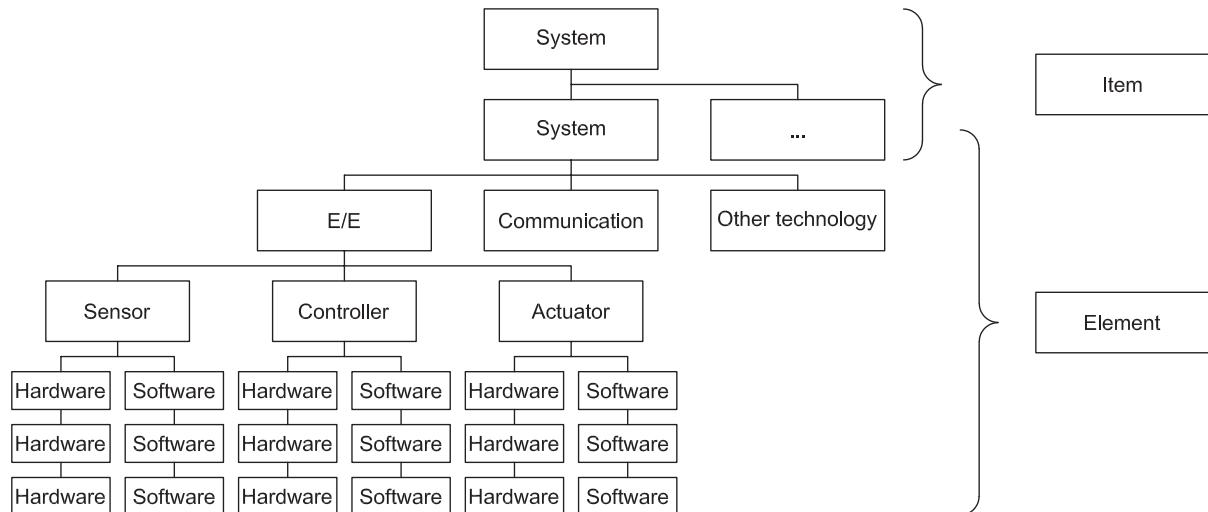


Figure 4 — Example item dissolution

4.3 Relationship between faults, errors and failures

The terms fault, error and failure are defined in ISO 26262-1. Figure 5 depicts the progression of faults to errors to failures from three different types of causes: systematic software issues, random hardware issues and systematic hardware issues. Systematic faults (see ISO 26262-1) are due to design or specifications issues; software faults and a subset of hardware faults are systematic. Random hardware faults (see ISO 26262-1) are due to physical processes such as wear-out, physical degradation or environmental stress. At the component level, each different type of fault can lead to different failures. However, failures at the component level are faults at the item level. Note that in this example, at the vehicle level, faults from different causes can lead to the same failure. A subset of failures at the item level will be hazards (see ISO 26262-1) if additional environmental factors permit the failure to contribute to an accident scenario.

EXAMPLE If unexpected behaviour of the vehicle occurs while the vehicle is starting to cross an intersection, a crash can occur, e.g. the risk of the hazardous event “vehicle bucking when starting to cross intersection” is assessed for severity, exposure and controllability (“bucking” refers to making sudden jerky movements).

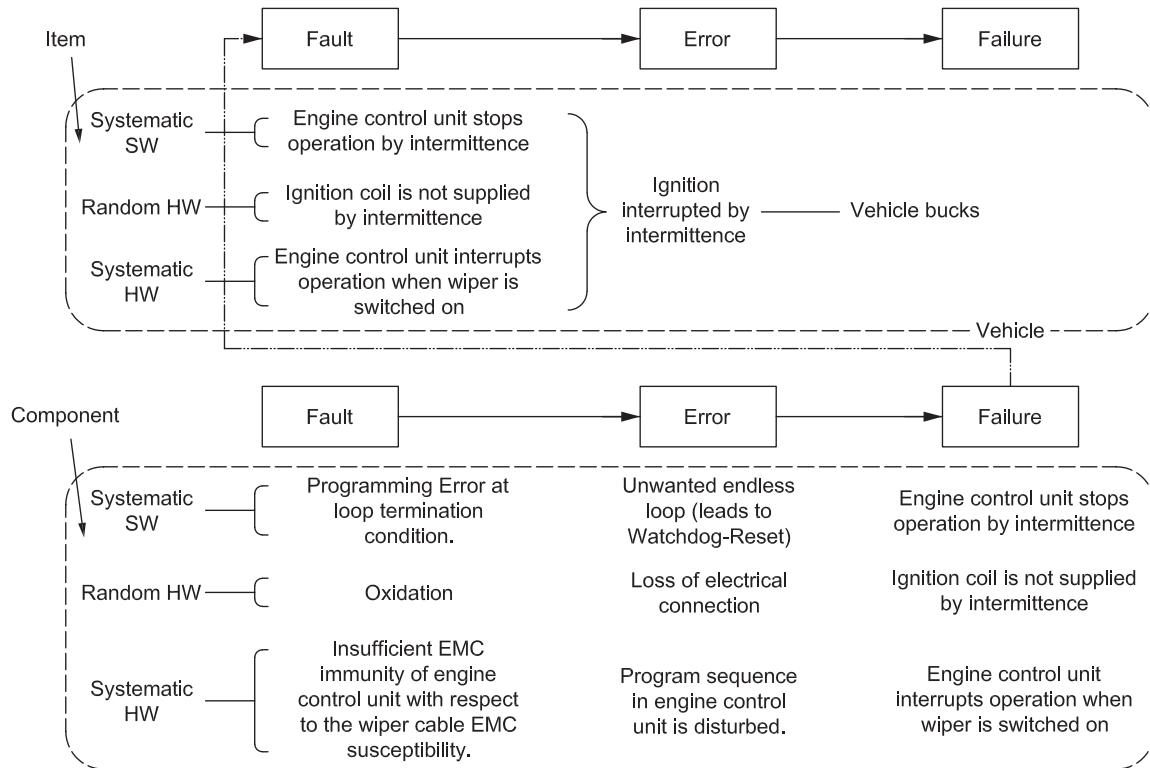


Figure 5 — Example of faults leading to failures

5 Selected topics regarding safety management

5.1 Work product

This subclause describes the term "work product".

A work product is the result of meeting the corresponding requirements of ISO 26262 (see ISO 26262-1). Therefore, a documented work product can provide evidence of compliance with these safety requirements.

EXAMPLE A requirements specification is a work product that can be documented by means of a requirements database or a text file. An executable model is a work product that can be represented by modelling language files that can be executed, e.g. for simulation purposes by using a software tool.

The documentation of a work product [see ISO 26262-8:2011, Clause 10 (Documentation)] serves as a record of the executed safety activities, safety requirements or of related information. Such documentation is not restricted to any form or medium.

EXAMPLE The documentation of a work product can be represented by electronic or paper files, by a single document or a set of documents. It can be combined with the documentation of other work products or with documentation not directly dedicated to functional safety.

To avoid the duplication of information, cross-references within or between documentation can be used.

5.2 Confirmation measures

5.2.1 General

In ISO 26262, specified work products are evaluated during subsequent activities, either as part of the confirmation measures or as part of the verification activities. This subclause describes the difference between verification and confirmation measures.

On the one hand, the verification activities are performed to determine the completeness and correct specification, or implementation, of safety requirements. The verification of work products can include:

- verification reviews to verify the specification, or implementation, of derived safety requirements against the safety requirements at a higher level, regarding completeness and correctness; or
- the execution of test cases or the examination of test results to provide evidence of the fulfilment of specified safety requirements, by exercising the item or its element(s).

The verification activities are specified in ISO 26262-3, ISO 26262-4, ISO 26262-5 and ISO 26262-6. Furthermore, generic requirements regarding the verification activities in ISO 26262 are specified in ISO 26262-8:2011, Clause 9 (Verification), and further details specific to the verification of safety requirements are specified in ISO 26262-8:2011, Clause 6 (Specification and management of safety requirements).

On the other hand, the confirmation measures are performed to evaluate the item's achievement of functional safety, including a confirmation of:

- the proper definition, tailoring and execution of the safety activities performed during the item development and of the implemented safety processes, with regard to the ISO 26262 requirements; and
- the proper content of the work products with regard to the corresponding ISO 26262 requirements.

The confirmation measures are specified in ISO 26262-2:2011, Clause 6 (Safety management during the concept phase and the product development).

EXAMPLE If an ASIL decomposition is applied during the system design phase:

- the verification of the resulting system design is performed against the technical safety concept (see ISO 26262-4:2011, 7.4.8); and
- the confirmation of the correct application of the ASIL decomposition can be performed as part of a functional safety assessment, with regard to ISO 26262-9:2011, Clause 5 (Requirements decomposition with respect to ASIL tailoring), including the confirmation that a dependent failure analysis has been performed and justifies the claim of sufficient independence between the elements that implement the corresponding redundant safety requirements.

5.2.2 Functional safety assessment

If the highest ASIL of the item's safety goals is ASIL C or D, a functional safety assessment is performed to evaluate an item's achievement of functional safety. In ISO 26262-2, certain aspects of a functional safety assessment are described separately, i.e. the functional safety audit and the confirmation reviews.

A functional safety assessment includes:

- a) a review of the appropriateness and effectiveness of the implemented safety measures that can be assessed during the item development;
- b) an evaluation of the work products that are required by the safety plan. The review of selected work products is emphasised. These are coined as confirmation reviews and aim to confirm the compliance of such work products with the corresponding requirements of ISO 26262; and
- c) one or more functional safety audits to evaluate the implementation of the processes required for functional safety.

A functional safety assessment can be repeated or updated.

EXAMPLE 1 A functional safety assessment update because of a change of the item, or element(s) of the item, that is identified by the change management as having an impact on the functional safety of the item [see ISO 26262-8:2011, Clause 8 (Change management)].

EXAMPLE 2 An iteration of a functional safety assessment triggered by the follow-up of a functional safety assessment report that included a recommendation for a conditional acceptance or rejection of the item's functional safety. In this case, the iteration includes a follow-up of the recommendations resulting from the previous functional safety assessment(s), including an evaluation of the performed corrective actions, if applicable.

If the highest ASIL of the item's safety goals is ASIL B, a functional safety assessment can be omitted or performed less rigorously. However, even if the functional safety assessment is not performed, other confirmation measures are still performed, i.e. the confirmation reviews of the hazard analysis and risk assessment, the safety plan, the item integration and testing plan, the validation plan, the applicable safety analyses, the proven in use arguments (if applicable), and the completeness of the safety case (see ISO 26262-2:2011, Table 1).

If the highest ASIL of the item's safety goals is ASIL A, there is no requirement or recommendation in ISO 26262 for or against performing a functional safety assessment. However, confirmation reviews of the hazard analysis and risk assessment and of the applicable safety analyses are still performed.

In the case of a distributed development, the scope of a functional assessment includes the work products generated, and the processes and safety measures implemented, by a vehicle manufacturer and the suppliers in the item's supply chain [see ISO 26262-2 and ISO 26262-8:2011, Clause 5 (Interfaces within distributed developments)].

The purpose of a functional safety assessment is to evaluate an item's achievement of functional safety, which is only possible at the item level. Therefore, a functional safety assessment at the premises of a supplier (that develops elements of the item) refers only to an assessment with a limited scope, which essentially serves as an input for the subsequent functional safety assessment activities (at the customer level). As the final customer in the item development, the vehicle manufacturer appoints person(s) to perform a functional safety assessment in its full scope, so as to judge an item's achievement of functional safety. This judgement includes providing a recommendation for acceptance, conditional acceptance, or rejection of the item's functional safety.

NOTE For the case where a Tier 1 supplier is responsible for the item development including vehicle integration, this supplier takes over the aforementioned role of the vehicle manufacturer.

In a practical manner, a functional safety assessment in the case of a distributed development can thus be broken down into:

- functional safety assessments with a limited scope at the supplier's premises, concerning the suppliers in the supply chain. The applicable ASIL is the highest inherited ASIL (of the item's safety goals) across the elements, of the item, that are developed by the supplier (see also ISO 26262-8:2011, 5.4.5); and
- a final functional safety assessment that includes a judgement of the functional safety achieved by the integrated item, e.g. performed by the vehicle manufacturer. The applicable ASIL is the highest ASIL of the item's safety goals (see also ISO 26262-2).

EXAMPLE A vehicle manufacturer develops an item with an ASIL D Safety Goal (SG1) and an ASIL A Safety Goal (SG2), and will perform a functional safety assessment regarding this item. It is possible that, for example, a Tier 2 or Tier 3 supplier only develops ASIL A elements of the item, i.e. only elements that inherit the ASIL of SG2 [however, refer to ISO 26262-9:2011, Clause 6 (Criteria for coexistence of elements), if applicable]. There is no requirement or recommendation (for or against) in ISO 26262 to perform a functional safety assessment at this supplier's premises regarding this item development.

The scope, procedure (e.g. work products to be made available by the supplier, work products to be reviewed by the customer) and execution of a functional safety assessment concerning the interface between a customer and a supplier are specified in the corresponding Development Interface Agreement [see ISO 26262-8:2011, Clause 5 (Interfaces within distributed developments)].

EXAMPLE DIA between a vehicle manufacturer (customer) and a Tier 1 supplier. DIA between a Tier 1 supplier (customer) and a Tier 2 supplier.

A possible manner to perform a functional safety assessment in the case of a distributed development is that the vehicle manufacturer and the suppliers in the supply chain each address those aspects of the assessment activities [see bullets a), b) and c) above] for which the respective party is responsible for, as follows:

- a supplier reviews the safety measures implemented in the developed elements including their appropriateness and effectiveness to comply with the corresponding safety goals or safety requirements (provided by the customer or developed by the supplier), and evaluates its implemented processes and the applicable work products. A supplier also evaluates the potential impacts of the developed elements on the item's functional safety, e.g. identifies whether implemented safety measures can lead to new hazards; and
- the vehicle manufacturer evaluates the functional safety of the integrated item. A part of the evaluation can be based on the work products or information provided by one or more suppliers, including reports of the functional safety assessments performed at supplier's premises.

NOTE A customer can evaluate the safety measures implemented by a supplier and the work products made available by a supplier. A customer can also evaluate the processes implemented by a supplier at the supplier's premises (see ISO 26262-8:2011, 5.4.4.8)

5.3 Understanding of safety cases

5.3.1 Interpretation of safety cases

The purpose of a safety case is to provide a clear, comprehensive and defensible argument, supported by evidence, that an item is free from unreasonable risk when operated in an intended context.

The guidance given here focuses on the scope of ISO 26262.

There are three principal elements of a safety case, namely:

- the requirements;
- the argument; and
- the evidence, i.e. ISO 26262 work products.

The relationship between these three elements, in the context of ISO 26262, is depicted in Figure 6.

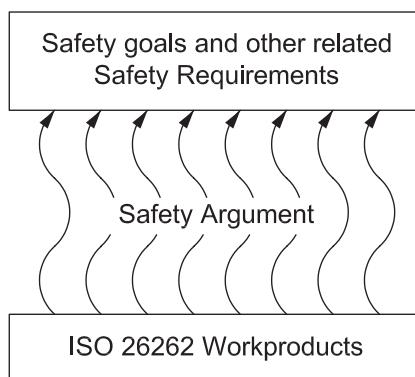


Figure 6 — Key elements of a safety case (see [2])

The safety argument communicates the relationship between the evidence and the objectives. The role of the safety argument is often neglected. It is possible to present many pages of supporting evidence without clearly explaining how this evidence relates to the safety objectives. Both the argument and the evidence are crucial elements of the safety case and go hand-in-hand. An argument without supporting evidence is unfounded, and therefore unconvincing. Evidence without an argument is unexplained, resulting in a lack of clarity as to

how the safety objectives have been satisfied. Safety cases are communicated through the development and presentation of safety case reports. The role of a safety case report is to summarize the safety argument and then reference the reports capturing the supporting safety evidence (e.g. test reports).

Safety arguments used to date in other industries have often been communicated in safety case reports through narrative text. Narrative text can describe how a safety objective has been interpreted, allocated and decomposed, ultimately leading to references to evidence that demonstrate fulfilment of lower-level safety claims. Alternatively, it is becoming increasingly popular to use graphical argument notations (such as Claims–Argument–Evidence and the Goal Structuring Notation [2]) to visually and explicitly represent the individual elements of a safety argument (requirements, claims, evidence and context) and the relationships that exist between these elements (i.e. how individual requirements are supported by specific claims, how claims are supported by evidence and the assumed context that is defined for the argument).

A safety argument that argues safety through direct appeal to features of the implemented item (e.g. the behaviour of a timing watchdog) is often termed a product argument. A safety argument that argues safety through appeal to features of the development and assessment process (e.g. the design notation adopted) is often termed a process argument.

Both types of argument can be used to achieve a sound argument for the safety of the item where a process argument can be seen as providing the confidence in the evidences used in the product argument.

5.3.2 Safety case development lifecycle

The development of a safety case can be treated as an incremental activity that is integrated with the rest of the development phases of the safety lifecycle.

NOTE The safety plan can include the planning for incremental steps and the preliminary versions of the safety case.

Such an approach allows intermediate versions of the safety case at given milestones of the product development. For example, a preliminary version of the safety case can be created after the verification of the technical safety requirements; an interim version of the safety case can be created after the verification of the system design; and the final version can be created just prior to the functional safety assessment.

The safety case is subject to a confirmation review as given in ISO 26262-2:2011, 6.4.7 (Confirmation measures: types, independency and authority).

If the item is modified, the impact on the safety case is evaluated and, if necessary, the safety case is updated considering modifications.

6 Concept phase and system development

6.1 General

This section provides an overview of the principles behind the hazard analysis and risk classification using simplified examples to the concepts.

6.2 Example of hazard analysis and risk assessment

6.2.1 General

Consider the example of an item controlling an energy storage device embedded in the vehicle. For the purpose of this example, the stored energy is intended to be released only if the vehicle is running greater than or equal to 15 km/h. The release of the stored energy at less than 15 km/h can lead to the overheating and consequent explosion of the device.

6.2.2 Analysis 1

- a) Hazard identification
 - failure leading to an unwanted release of energy of the device that can result in an explosion
- b) Hazardous event

For the purpose of this example, the driving situation considered for the hazard analysis and risk assessment is:

- driving less than 15 km/h in a traffic congestion

An unwanted release of energy due to a failure in the item occurs. The energy storage device explodes, causing severe harm to the occupants of the vehicle.

- c) Classification of the identified hazardous event

The explosion leads to life-threatening injuries for the passengers of the vehicle, with survival uncertain, so the severity is estimated as S3.

The vehicle is travelling in traffic congestion, below the speed of 15 km/h. Based on traffic statistic for the target market of the vehicle, the exposure of this situation could be estimated as E3 (occurring between 1 % and 10 % of the driving time).

The ability of the driver or the passengers of the vehicles to control the item failure and the explosion of the device is considered as implausible: this controllability could be estimated as C3 (difficult to control or uncontrollable).

The application of ISO 26262-3:2011, Table 4: ASIL determination leads to an ASIL C.

6.2.3 Analysis 2

- a) Hazard identification
 - failure does not lead to the release of energy
- b) Hazardous event
 - any driving situation

A failure in the item occurs but does not lead to the release of any energy from the storage device, so it leads to no harm.

- c) Classification of the identified hazardous event

Since the item failure does not lead to harm, the severity is classified as S0 and controllability does not need to be determined. Therefore a safety goal does not need to be defined.

6.3 An observation regarding controllability classification

As explained in ISO 26262-3:2011, Clause 7 (Hazard analysis and risk assessment), the controllability represents an estimation of the probability that the driver or other traffic participant is able to avoid the specific harm.

In the simplest case, only one outcome is considered for a given hazardous event, and the controllability represents an estimation of the probability that this outcome is avoided. However, there can be other cases. For example, a severe outcome (e.g. severity class S2) can be possible but relatively easy to avoid (e.g. controllability C1), while a less severe outcome (e.g. S1) is more difficult to avoid (e.g. C3). Assuming that the

exposure class is E4, the following set of values can be the result, which illustrates that it is not necessarily the highest severity that leads to the highest ASIL:

- E4, S2, C1 => ASIL A
- E4, S1, C3 => ASIL B

In this example, ASIL B is an appropriate classification of the hazardous event.

6.4 External measures

6.4.1 General

An external measure is a measure separate and distinct from the item that reduces or mitigates the risks resulting from a failure of the item.

6.4.2 Example of vehicle-dependent external measures 1

Vehicle A is equipped with a manually operated transmission gear box which can be left in any gear, including neutral, upon key off. Vehicle B is equipped with an automated gear box which, at key off, maintains one gear engaged and a normally closed clutch. Both vehicles have the added item, Electrical Parking Brake (EPB).

A scenario is analyzed for both vehicles which includes:

- The vehicle is parked (key off, driver not present).
- Parked surface is kerbside and sloped, located in a populated urban area.
- A failure involving a sudden loss of EPB occurs.

In this scenario, Vehicle A, when left in neutral at key off (a situation that corresponds to a reasonably foreseeable misuse), will potentially move if left unattended. This can result in an assessed controllability rating of C3, a severity rating of S2 or higher depending on the presence of nearby vulnerable persons, and an exposure ranking greater than E0. Depending on the exposure rating actually assigned, the ratings proposed result in an assigned ASIL classification between QM and C.

Vehicle B however always engages a gear so does not move, thus there is no resulting hazard. The vehicle-dependent external measures included in this design contribute to the elimination of risk for this scenario, but only if the robotized gear box and the EPB can be shown to be sufficiently independent.

6.4.3 Example of vehicle-dependent external measures 2

Vehicle A is equipped with dynamic stability control in addition to a Stop & Start feature. Vehicle B is only equipped with the Stop & Start feature.

A scenario is analyzed for both vehicles which includes:

- The vehicle is being driven at medium-high speed [50 km/h < v < 90 km/h].
- The road surface is paved and dry, and in a suburban area.
- The vehicle is approaching a medium curvature bend in the road.
- The vehicle speed and road curvature contribute to a medium-high lateral acceleration.
- A failure in the Stop & Start feature triggering an undesired engine shutdown results in a sudden loss of traction power during the scenario.

As a result of the sudden loss of traction power, a yaw moment is induced on the vehicle, requiring the driver to adjust steering input to re-establish the control of the vehicle. Performing this manoeuvre in Vehicle B can be shown to have a lower controllability, which can contribute to an ASIL rating of C or D. By contrast, the

dynamic stability control feature in Vehicle A limits the effects of the lateral instability. As a result, the controllability rating will be lower for Vehicle A. Therefore, the vehicle-dependent external measures provided by the dynamic stability control contribute to the reduction of risk for this scenario. However, this is the case only if it can be shown that the failure in the Stop & Start function being considered cannot propagate to the dynamic stability control function and is not a dependent failure with regard to both functions.

6.5 Example of combining safety goals

6.5.1 Introduction

Safety goals are top-level safety requirements for the item. They lead to the functional safety requirements needed to avoid an unreasonable risk for an hazardous event. They are determined in the concept phase in accordance with ISO 26262-3:2011, 7.4.8. When safety goals are similar or refer to the same hazard in different situations, they can be combined into a single safety goal with the highest ASIL of the original safety goals. This can simplify the further development, as fewer safety goals have to be managed, while still covering all the identified hazards.

6.5.2 General

In the following example, the item, the safety goals and the ASIL classifications shown are only intended to illustrate the safety goal combination process. This example does not reflect the application of ISO 26262 on a similar real-life project. In particular, it is not complete in terms of failure modes identification, situation analysis and the assessment of vehicle level effects.

For simplicity, the example is limited to the composition of two safety goals, but the same approach can be extended to a higher number of initial safety goals.

6.5.3 Function definition

Consider a vehicle equipped with an Electrical Parking Brake (EPB) system. The EPB system, when activated by a specific driver's request, applies brake torque to the vehicle's rear wheels to prevent unintended vehicle movement while parked (parking function).

6.5.4 Safety goals applying to the same hazard in different situations

6.5.4.1 Hazard analysis and risk assessment

To simplify the example, consider the following failure mode of the parking function:

- unintended parking brake activation.

NOTE In this context, the term “unintended activation” is intended as a function actuation without the driver's request.

This failure mode can lead to different vehicle effects depending on the specific situation present when the fault occurs, as shown in Table 1.

Table 1 — Safety goals resulting from the same hazard in different situations

Failure mode	Hazard	Specific situation	Hazardous event	Possible consequences	ASIL	Safety goal	Safe state
Unintended parking brake activation	Unexpected deceleration	High speed OR taking a bend OR low friction surface	Unexpected deceleration at high speed OR taking a bend OR low friction surface	Loss of vehicle stability	Higher ASIL	Avoid activating the parking function without the driver's request when the vehicle is moving	EPB disabled
Unintended parking brake activation	Unexpected deceleration	Medium-low speed AND high friction surface	Unexpected deceleration at medium-low speed AND high friction surface	Rear end collision with the following vehicle	Lower ASIL	Avoid activating the parking function without the driver's request when the vehicle is moving	EPB disabled

6.5.4.2 Safety goals elaboration

As shown above, the same safety goals and safe states are applicable to both situations. Therefore, the following safety goal can be defined:

- Safety goal: Avoid activating the parking function when the vehicle is moving, without the driver's request.
- Safe state: EPB disabled.
- ASIL: *Higher ASIL* determined in Table 1 is assigned to this safety goal.

7 Safety process requirement structure - Flow and sequence of safety requirements

The flow and sequence of the safety requirement development in accordance with ISO 26262 is illustrated in Figure 7 and Figure 8, and outlined below. The specific clauses are indicated in the following manner: "m-n", where "m" represents the number of the part and "n" indicates the number of the clause or subclause within that part.

A hazard analysis and risk assessment is performed to identify the risks and to define the safety goals for these risks. [See ISO 26262-3:2011, Clause 7 (Hazard analysis and risk assessment).]

A functional safety concept is derived which specifies functional safety requirements to satisfy the safety goals. These requirements define the safety mechanisms and the other safety measures that will be used for the item. In addition, the system architectural elements that support these requirements are identified. [See ISO 26262-3:2011, Clause 8 (Functional safety concept).]

A technical safety concept is derived which specifies the technical safety requirements and their allocation to system elements for implementation by the system design. These technical safety requirements will indicate the partitioning of the elements between the hardware and the software. [See ISO 26262-4:2011, Clause 6 (Specification of the technical safety requirements).]

The system design will be developed in accordance with the technical safety requirements. Their implementation can be specified in the system design specification. [See ISO 26262-4:2011, Clause 7 (System design).]

Finally, the hardware and software safety requirements will be provided to comply with the technical safety requirements and the system design. [See ISO 26262-5:2011, Clause 6 (Specification of hardware safety requirements) and ISO 26262-6:2011, Clause 6 (Specification of software safety requirements).]

Figure 7 illustrates the relationship between the hardware requirements and the design phases of ISO 26262.

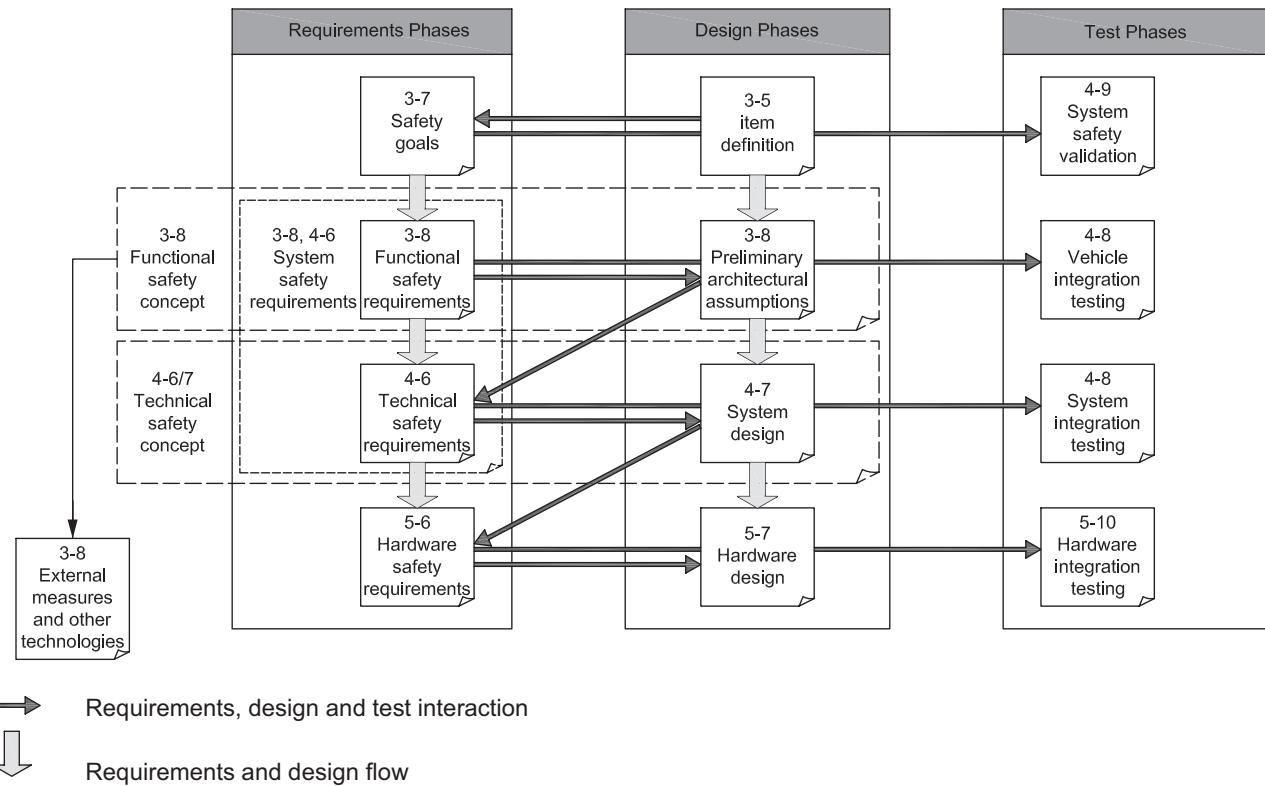


Figure 7 — Safety requirements, design and test flow from concept to hardware

Figure 8 illustrates the relationship between the software requirements, the design and the test subphases of ISO 26262.

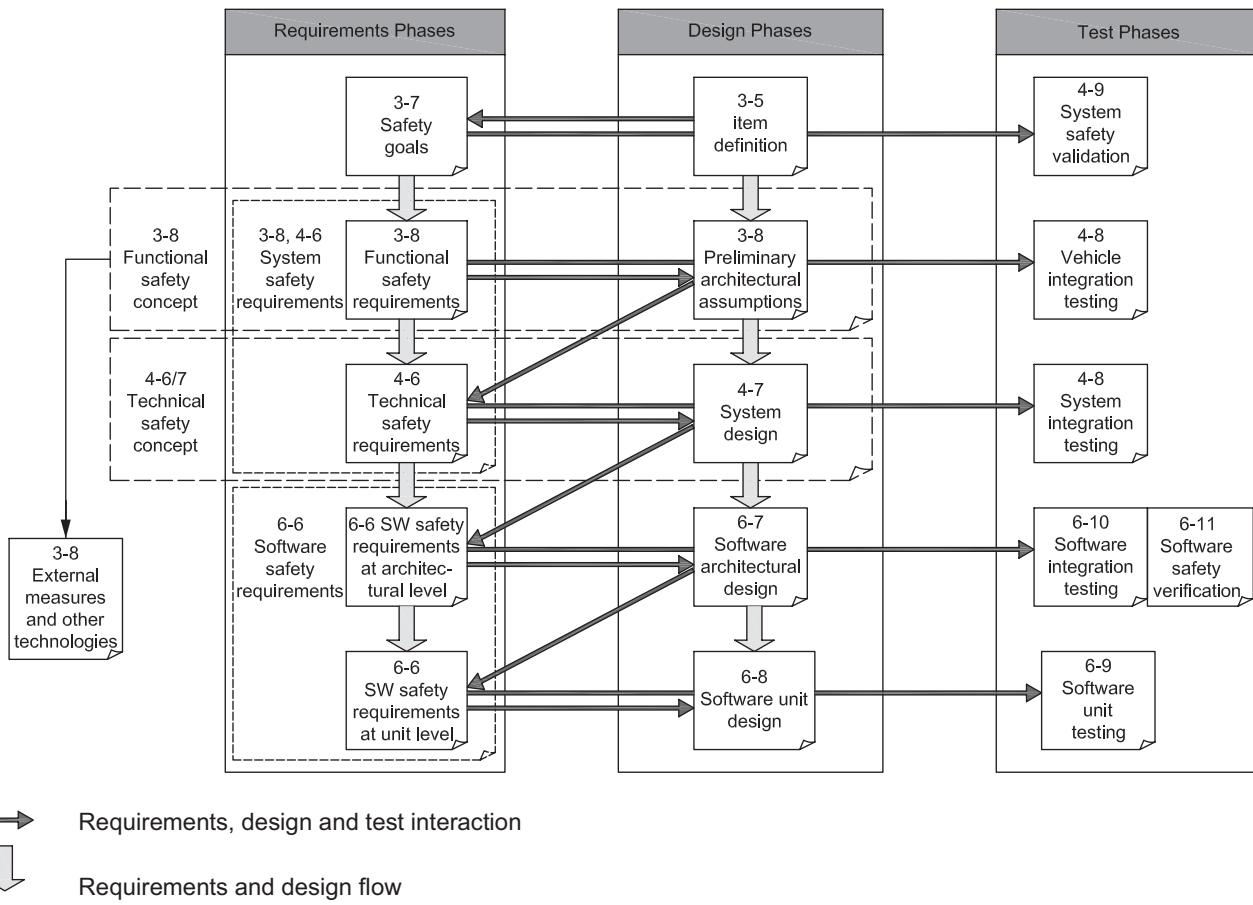


Figure 8 — Safety requirements, design and test flow from concept to software

— System design:

The system design is continuously refined from the item definition (3-6) to the preliminary architectural assumptions and the system design (4-7).

— Dependence among test levels:

The test specifications and test cases on each level mainly depend on the corresponding requirements and design. They do not depend on the test specifications, test cases and tests results of other test levels. The test specifications typically depend on the test environment.

— Dependence of test levels on requirements and design levels:

Test specifications and test cases are derived from the requirements on the same level, supported by information on the design at the same level.

EXAMPLE For performance testing, information on the design is necessary.

— Software safety requirements verification:

The phase of software safety requirements verification (6-11) requires the integration of software and hardware.

— External measures and other technologies:

External measures and other technologies are validated at the vehicle level.

8 Concerning hardware development

8.1 The classification of random hardware faults

8.1.1 General

In general, the combinations of faults that are considered are limited to combinations of two independent hardware faults, unless analysis based on the functional or technical safety concept has shown that n point faults with $n > 2$ are relevant. Therefore, for a given safety goal and a given HW element a fault can be classified in most cases as either:

- a) single-point fault;
- b) residual fault;
- c) detected dual-point fault;
- d) perceived dual-point fault;
- e) latent dual-point fault; or
- f) safe fault.

Explanations on the various fault classes, as well as examples, are given below.

8.1.2 Single-point fault

This fault:

- can lead directly to the violation of a safety goal; and
- is a fault of a hardware element for which not one safety mechanism prevents some of the faults of the hardware element from violating the safety goal.

EXAMPLE An unsupervised resistor for which at least one failure mode (e.g. open circuit) has the potential to violate the safety goal.

NOTE If a hardware part has at least one safety mechanism (e.g. a watchdog for a microcontroller), then none of the faults of that part are classified as a single-point faults. The faults for which the safety mechanisms do not prevent the violation of the safety goal are classified as residual faults.

8.1.3 Residual fault

This fault:

- can lead directly to the violation of the safety goal; and
- is a fault of a hardware element for which at least one safety mechanism prevents some of the faults of the hardware element from violating the safety goal.

EXAMPLE If a Random Access Memory (RAM) module is only checked by a checkerboard RAM test safety mechanism, certain kinds of bridging faults are not detected. The violation of the safety goals due to these faults are not prevented by the safety mechanism. These faults are examples of residual faults.

NOTE The safety mechanism has less than 100 % diagnostic coverage in this case.

8.1.4 Detected dual-point fault

This fault:

- contributes to the violation of the safety goal;
- can only lead to a safety goal violation in combination with one other independent hardware fault that is related to the dual-point fault; and
- is detected by a safety mechanism which prevents it from being latent.

EXAMPLE 1 Flash memory that is protected by parity: a single bit fault which is detected and triggers a reaction according to the technical safety concept, like switching off the system and informing the driver via a warning lamp.

EXAMPLE 2 Flash memory that is protected by Error Detection and Correction Code (EDC): faults in the EDC logic that are detected by a test and a reaction is triggered according to the technical safety concept, like informing the driver via a warning lamp.

In the case of a transient fault, where a safety mechanism restores the item to a fault-free state, such a fault can be considered as a detected dual-point fault even if the driver is never informed about its existence.

EXAMPLE A transient bit flip which is corrected by an Error Detection and Correction Code (EDC) before the data is provided to the CPU and is corrected later on by writing back the correct value. Logging can be used to distinguish between intermittent faults and true transient faults.

8.1.5 Perceived dual-point fault

This fault:

- contributes to the violation of the safety goal but will only lead to a safety goal violation in combination with one other independent hardware fault that is related to the dual-point fault; and
- is perceived by the driver with or without detection by a safety mechanism within a prescribed time;

EXAMPLE A dual-point fault can be perceived by the driver if the functionality is significantly and unambiguously affected by the consequence of the fault.

NOTE If a dual-point fault is perceived by the driver as well as detected by a safety mechanism it can be classified as either a detected or a perceived dual-point fault. It cannot be classified as both simultaneously since the latent-fault metric would be incorrectly calculated due to the fact that one fault would then contribute to the detected dual-point faults as well as to the perceived dual-point faults, counting this fault twice.

8.1.6 Latent dual-point fault

This fault:

- contributes to the violation of the safety goal but will only lead to the violation of the safety goal in combination with one other independent fault; and
- is neither detected by a safety mechanism nor perceived by the driver. Until the occurrence of the second independent fault, the system is still operable and the driver is not informed about the fault.

EXAMPLE 1 In the case of a flash memory that is protected by EDC: a permanent single bit fault for which the value is corrected by the EDC when read but that is neither corrected in the flash memory nor signalled. In this case, the fault cannot lead to a safety goal violation (since the faulty bit is corrected), but it is neither detected (since the single bit fault is not signalled) nor perceived (since there is no impact on the functionality of the application). If an additional fault occurs in the EDC logic, it can lead to a loss of control of this single bit fault, leading to a potential violation of the safety goal.

EXAMPLE 2 In the case of a flash memory which is protected by EDC: a fault in the EDC logic leading to an unavailability of the EDC which is not detected by a test.

8.1.7 Safe fault

Safe faults can be faults of one of two categories:

- a) all n point faults with $n > 2$, unless the safety concept shows them to be a relevant contributor to a safety goal violation, or
- d) faults that will not contribute to the violation of a safety goal.

EXAMPLE 1 In the case of a flash memory that is protected by EDC and a Cyclic Redundancy Check (CRC): a single bit fault which is corrected by EDC but is not signalled. The fault is prevented from violating the safety goal but is not signalled by the EDC. If the EDC logic fails, the fault is detected by the CRC and the system switched off. Only if a single bit fault in the flash is present, the EDC logic fails and the CRC checksum supervision fails, can a violation of a safety goal occur ($n=3$).

EXAMPLE 2 In case three resistors are connected in series to overcome the problem of a single-point fault in the case of a short circuit, the short circuit of each individual resistor can be considered to be a safe fault as three independent short circuits are needed ($n=3$).

8.1.8 Flow diagram for fault classification and fault class contribution calculation

Failure modes of a hardware element can be classified as shown in ISO 26262-5:2011, Figure B.1 and using the flow diagram described in ISO 26262-5:2011, Figure B.2. Figure 9 shows the calculation of the various failure rates considering the basic failure rate and coverage of the different failure modes (residual vs. latent).

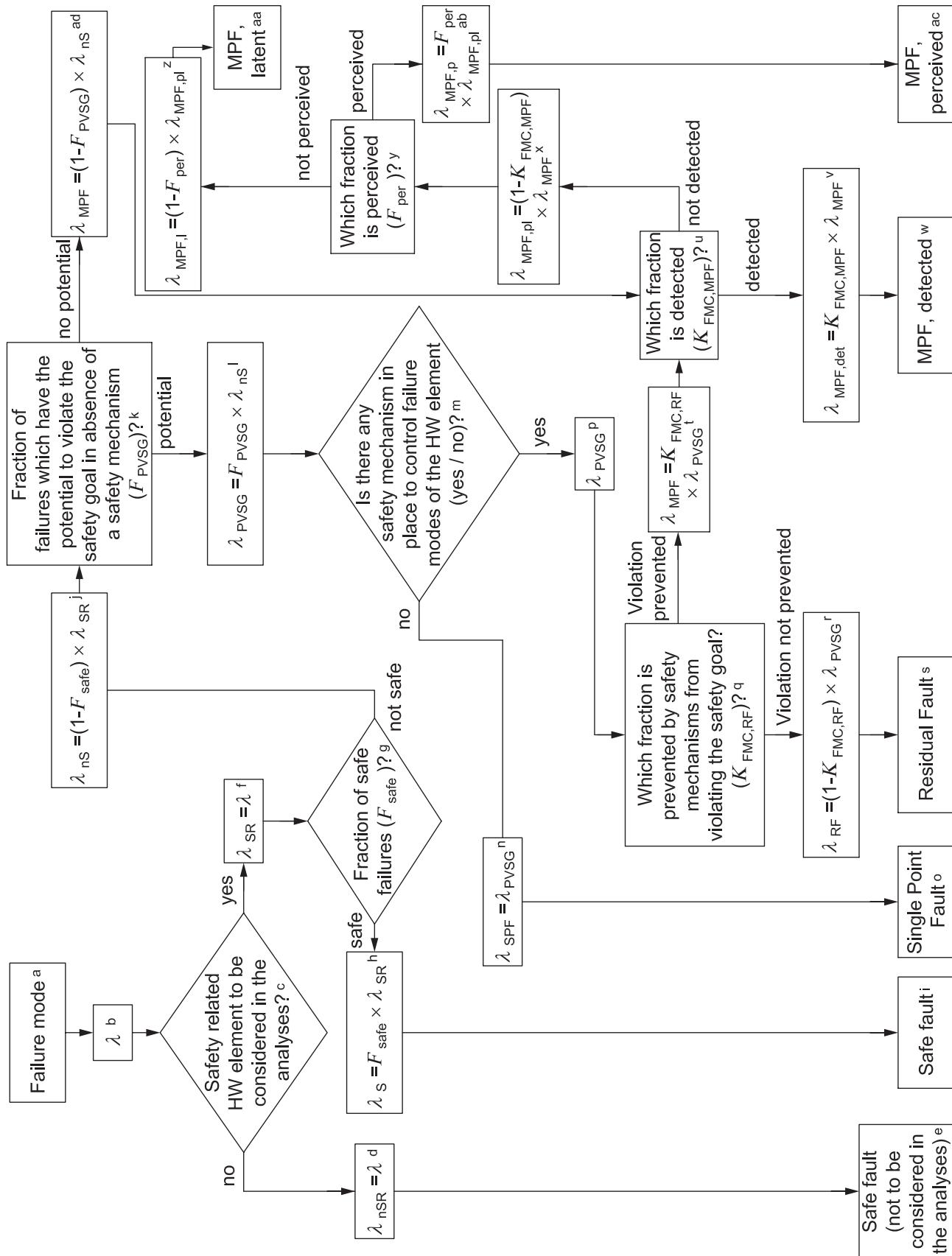


Figure 9 — Classification of failure categories and calculation of corresponding failure rates

- ^a Failure mode to be analyzed.
- ^b λ is the failure rate associated with the failure mode under consideration.
- ^c If any failure mode of the HW element which is analyzed is safety-related, then the hardware element is safety-related.
- ^d λ_{nSR} is the “**not Safety-Related**” failure rate. $\lambda_{nSR} = \lambda$ if all failure modes of the HW element under consideration are not safety-related.
- ^e Faults that are not safety-related are safe faults and are not included in the single-point fault metric or the latent-fault metric.
- ^f λ_{SR} is the “**Safety-Related**” failure rate. They are considered within the single-point fault metric and the latent-fault metric.
- ^g F_{safe} is the fraction of safe faults of this failure mode. Safe faults do not significantly contribute to the violation of the safety goal. For complex HW elements (e.g. microcontrollers), it is difficult to give the exact proportion. In this case, a conservative F_{safe} of 0,5 (i.e. 50 %) can be assumed.
- ^h λ_S is the failure rate for the “**Safe**” faults. It is equal to $\lambda_{SR} \times F_{safe}$.
- ⁱ The λ_S will contribute to the total rate of safe faults.
- ^j λ_{nS} is the “**not Safe**” failure rate. These include the single-point faults, residual faults and multiple-point faults (with $n = 2$). It is equal to $(1 - F_{safe}) \times \lambda_{SR}$.
- ^k F_{PVSG} is the fraction of λ_{nS} that have the potential to directly violate the **safety goal** without considering any of the safety mechanisms that can exist to prevent this.
- ^l λ_{PVSG} is the failure rate of the faults which have the potential to directly violate the safety goal without considering any safety mechanisms that can exist to prevent this. It is equal to $F_{PVSG} * \lambda_{nS}$.
- ^m Determine if the faults leading to the failure mode under consideration are single-point faults. They are if no safety mechanism is implemented to prevent any fault of the hardware element under consideration from violating a safety goal.
- ⁿ λ_{SPF} is the “**Single-Point Faults**” failure rate. If there is not at least one safety mechanism present to control failures of the considered hardware element, all λ_{PVSG} are single-point faults.
- ^o λ_{SPF} will contribute to the total rate of single-point faults.
- ^p If the HW element under consideration has at least one safety mechanism which prevents at least one of its failures from violating a safety goal, the faults leading to the failure under consideration are not single-point faults. In the following procedure, the λ_{PVSG} is split up into residual fault and detected, perceived and latent multiple-point faults.
- ^q What fraction of λ_{PVSG} is prevented by safety mechanisms from violating the safety goal? This fraction is equivalent to the failure mode coverage with respect to residual faults [see also ISO 26262-5:2011, Annex E (Example calculation of hardware architectural metrics: “single-point fault metric” and “latent-fault metric”)]. $K_{FMC,RF}$ is the acronym of the failure mode coverage with respect to residual faults.
- ^r λ_{RF} is the “**Residual Fault**” failure rate. $\lambda_{RF} = (1 - K_{FMC,RF}) \times \lambda_{PVSG}$
- ^s λ_{RF} contributes to the total rate of residual faults.
- ^t λ_{MPF} is the “**Multiple-Point Faults**” failure rate. $\lambda_{MPF} = K_{FMC,RF} \times \lambda_{PVSG}$.

- ^u Identify detected and not detected faults. $K_{FMC,MPF}$ is the failure mode coverage with respect to multiple-point faults.

NOTE There are two sources for multiple-point faults:

- faults which could lead to a safety goal violation, but for which a safety mechanism exists to prevent this;
- faults which by themselves cannot lead to a violation of the safety goal, but which contribute to a multiple-point failure scenario.

Depending on the source of the multiple-point fault, the failure mode coverage with respect to multiple-point faults can vary.

^v $\lambda_{MPF,det}$ is the “**Multiple-Point Faults, detected**” failure rate. $\lambda_{MPF,det} = \lambda_{MPF} * K_{FMC,MPF}$.

^w $\lambda_{MPF,det}$ contributes to the total rate of detected multiple-point faults.

^x $\lambda_{MPF,pl}$ is the “**Multiple-Point Faults, perceived or latent**” failure rate.

^y F_{per} is the fraction of the $\lambda_{MPF,pl}$ which is perceived by the driver.

^z $\lambda_{MPF,l}$ is the “**Multiple-Point Faults, latent**” failure rate.

^{aa} $\lambda_{MPF,l}$ contributes to the total rate of latent multiple-point faults.

^{ab} $\lambda_{MPF,p}$ is the “**Multiple-Point Faults, perceived**” failure rate.

^{ac} $\lambda_{MPF,p}$ contributes to the total rate of perceived multiple-point faults.

^{ad} λ_{MPF} is the “**Multiple-Point Faults**” failure rate.

8.1.9 How to consider the failure rate of multiple-point faults related to software-based safety mechanisms addressing random hardware failures

While systematic faults of software and hardware are not quantified in ISO 26262, a failure rate can be calculated for random hardware failures of hardware resources that support the execution of the software-based safety mechanisms addressing random hardware failures.

If those hardware resources are shared with functions which have the potential to directly violate a safety goal, then the fault models are chosen to reflect this and potential dependent failures are considered.

8.2 Example of residual failure rate and local single-point fault metric evaluation

8.2.1 General

This example demonstrates a way to evaluate the residual failure rate $\lambda_{RF,Sensor}$, the single-point failure rate λ_{SPF} and the localized version of the single-point fault metric $M_{SPFM,Sensor}$ of a sensor. In this example, the sensor is compared to the value of another sensor where both sensors measure the same physical quantity and have known tolerances. The values of a sensor, A_Master, are used by a feature of the application. The values of the other sensor, A_Checker, are solely used to validate values of sensor A_Master.

This monitoring is referenced in ISO 26262-5:2011, Annex D either as “Sensor Rationality Check” or as “Input comparison/voting”.

Only faults of the sensor A_Master are classified and evaluated in this example. Faults of sensor A_Checker are not addressed here.

Since sensor A_Master has a safety mechanism defined, all remaining faults which have the potential to violate the safety goal and which are not controlled (i.e. the violation of the safety goal is not prevented) are classified as residual faults. The single-point failure rate λ_{SPF} is (per definition) equal to zero.

8.2.2 Technical safety requirement for sensor A_Master

The boundary for safe operation of sensor A_Master is shown in Figure 10, and is regarded as a given in this example (i.e. the derivation from the safety goal is not discussed here). It can be expressed using the following terms:

With

$$\mu_{\text{SafRel,A,min}} = \text{Maximum}(C_{\text{PVSG}} ; v \times (1 + a))$$

where

C_{PVSG} is a constant value;

$\mu_{\text{SafRel,A,min}}$ is the safety-related lower boundary of sensor A_Master;

v is the physical value that is to be measured;

a is a constant value.

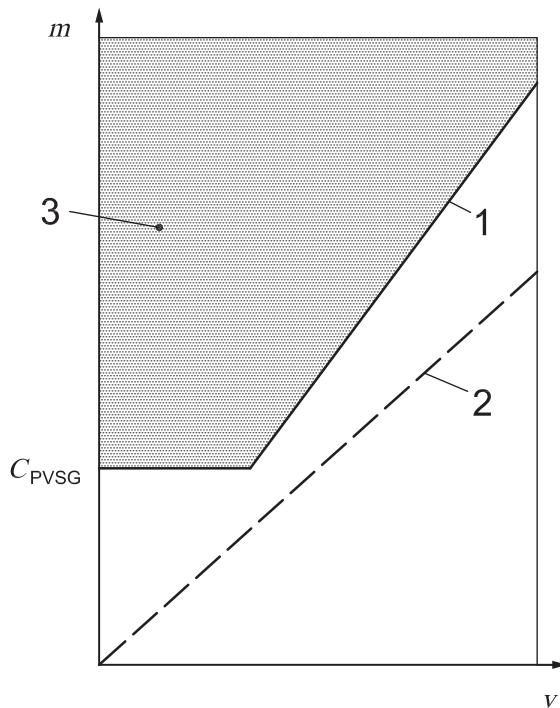
A safety-related failure of the sensor occurs when

$$m_{\text{A,Master}} \geq \mu_{\text{SafRel,A,min}}$$

where

$m_{\text{A,Master}}$ is the value reported by the sensor A_Master.

The safety requirement is to detect and control a safety-related failure of sensor A_Master within the fault tolerant time interval of T_{SenA} .



- Key:**
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 faults with the potential to violate the safety goal

Figure 10 — The boundary for safe operation of sensor A_Master

In Figure 10, the x axis is the real physical value v to be measured, the y axis is the value $m_{A,\text{Master}}$ reported by sensor A_Master. The dashed line shows the return value of an ideal sensor (i.e. a sensor with zero tolerance) as a reference. The solid line shows $\mu_{\text{SafRel},A,\min}$. If the sensor A_Master reports a value $m_{A,\text{Master}}$ that is on or above the solid line, a violation of a safety goal can occur.

8.2.3 Description of the safety mechanism

The elements of the safety mechanisms are the sensor A_Checker and the monitor hardware, which consists of a microcontroller with embedded software. The software periodically compares the values of the two sensors with each other, with the periodicity being smaller than the fault tolerant time T_{SenA} . The evaluation is done by the following pseudo code:

```

 $\Delta_A = m_{A,\text{Master}} - m_{A,\text{Checker}}$ 
if  $\Delta_A \geq \Delta_{\text{Max}}$  then failure is TRUE
if failure is TRUE then switch into safe state
where
 $m_{A,\text{Master}}$  is the value reported by the sensor A_Master;
 $m_{A,\text{Checker}}$  is the value reported by the sensor A_Checker;
 $\Delta_{\text{Max}}$  is a predefined constant maximum threshold used as pass/fail criteria.

```

It is assumed that the sensors have the following known tolerances:

$$m_{A,\text{Master}} = v +/ - C_{A,\text{Master}}$$

$$m_{A,Checker} = v +/- c_{A,Checker}$$

where

- $m_{A,Master}$ is the value reported by the sensor A_Master;
- $m_{A,Checker}$ is the value reported by the sensor A_Checker;
- $c_{A,Master}$ is a constant value representing the tolerance of sensor A_Master;
- $c_{A,Checker}$ is a constant value representing the tolerance of sensor A_Checker;
- v is the physical value to be measured.

The value Δ_{Max} is chosen so that a failure of sensor A_Master that can violate the safety goal is detected. To prevent false failure detections, Δ_{Max} is selected considering the tolerances of each sensor and other tolerances summarized in $c_{A,other}$ e.g. effects of sampling at different times:

$$\Delta_{Max} \geq c_{A,Master} + c_{A,Checker} + c_{A,other}$$

With this approach, the worst case of an undetected failure is:

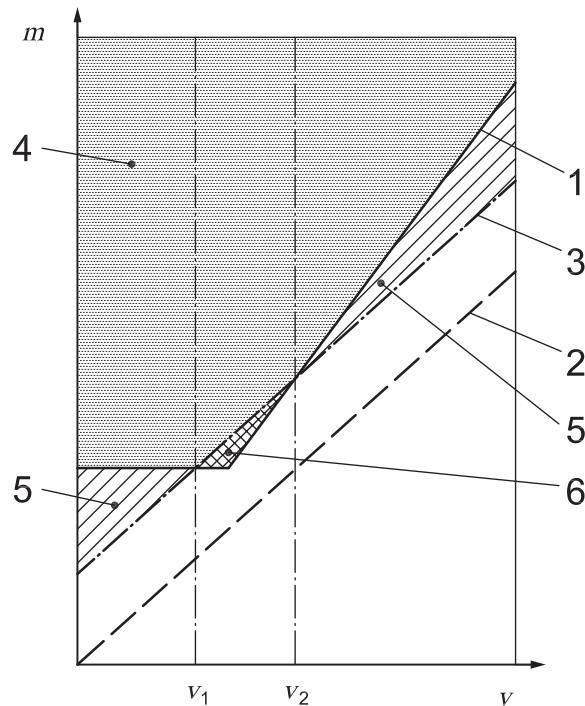
$$\begin{aligned}\mu_{A,Master,wc} &= m_{A,Checker} + \Delta_{Max} \\ &= v + c_{A,Checker} + \Delta_{Max}\end{aligned}$$

where

- $\mu_{A,Master,wc}$ is the worst case detection threshold, i.e. the maximum value $m_{A,Master}$ of sensor A_Master that is not detected as a failure;
- $m_{A,Checker}$ is the value reported by the sensor A_Checker;
- Δ_{Max} is a predefined constant maximum threshold used as pass/fail criteria;
- v is the physical value to be measured.

Every value $m_{A,Master}$ above $\mu_{A,Master,wc}$ is classified as a sensor failure.

Depending on the tolerance values, different detection scenarios are possible. Two examples are visualized in Figure 11 and Figure 12.



- Key:**
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,\text{Master},wc}$
 - 4 dual-point faults, detected
 - 5 detected faults with no potential to violate the safety goal
 - 6 residual faults

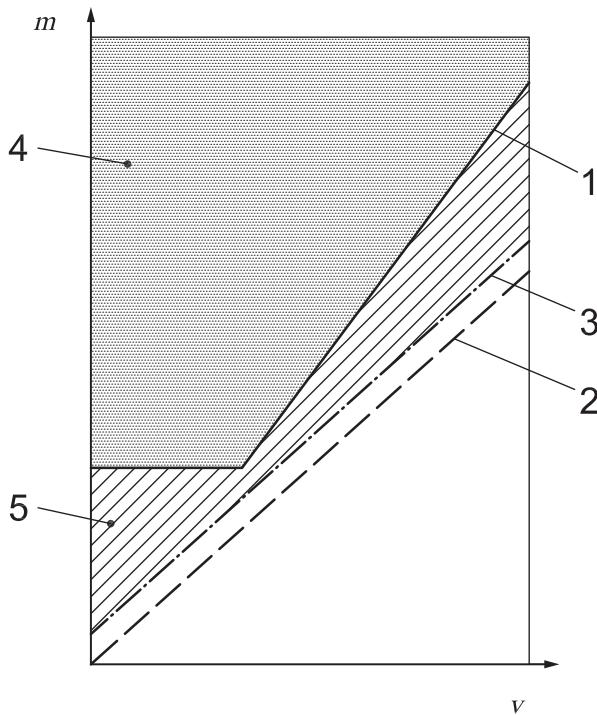
Figure 11 — Example 1 of worst case detection threshold (too high)

Three regions are indicated by arrows in Figure 11.

Region 5 - “detected faults with no potential to violate the safety goal” are faults that are detected by the safety mechanism because they are above the worst case detection threshold $\mu_{A,\text{Master},wc}$ but alone would not cause a violation of the safety goal because they are below the safety-related lower boundary $\mu_{\text{SafRel},A,\min}$.

Region 4 - “dual-point faults, detected” are faults that could cause a violation of the safety goal but are detected and mitigated by the safety mechanism. They are above both the worst case detection threshold $\mu_{A,\text{Master},wc}$ and the safety-related lower boundary $\mu_{\text{SafRel},A,\min}$. The dual-point nature of these faults means that it would require a failure of the safety mechanism and the sensor to cause a potential violation of the safety goal.

Region 6 - “residual faults” are not detected by the safety mechanism and can directly lead to a violation of the safety goal. The region $\mu_{\text{SafRel},A,\min} < \mu_{A,\text{Master},wc}$ for $v \in [v_1, v_2]$ lies below the worst case detection threshold $\mu_{A,\text{Master},wc}$ but above the safety-related lower boundary $\mu_{\text{SafRel},A,\min}$.



- Key:**
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,\text{Master},wc}$
 - 4 dual-point faults, detected
 - 5 detected faults with no potential to violate the safety goal

Figure 12 — Example 2 of worst case detection threshold ($M_{\text{SPFM},\text{Sensor}} = 100 \%$)

In the case of Figure 12, the worst case detection threshold $\mu_{A,\text{Master},wc}$ is always smaller than the safety-related lower boundary $\mu_{\text{SafRel},A,\min}$. In this case, the residual failure rate is zero, and the local single-point fault metric $M_{\text{SPFM},\text{Sensor}}$ of the sensor is equal to 100 %.

8.2.4 Evaluation of example 1 described in Figure 11

8.2.4.1 General

In the case of Figure 11, there are conditions when the worst case detection threshold $\mu_{A,\text{Master},wc}$ is higher than the safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ for sensor A_Master:

for $V \in [V_1, V_2]$: $\mu_{\text{SafRel},A,\min} \leq \mu_{A,\text{Master},wc}$

To determine the residual failure rate $\lambda_{RF,\text{Sensor}}$ and the $M_{\text{SPFM},\text{Sensor}}$ under these conditions, further analysis is necessary. The following is an example of this analysis. In ISO 26262-5:2011, Annex D the following failure modes are stated:

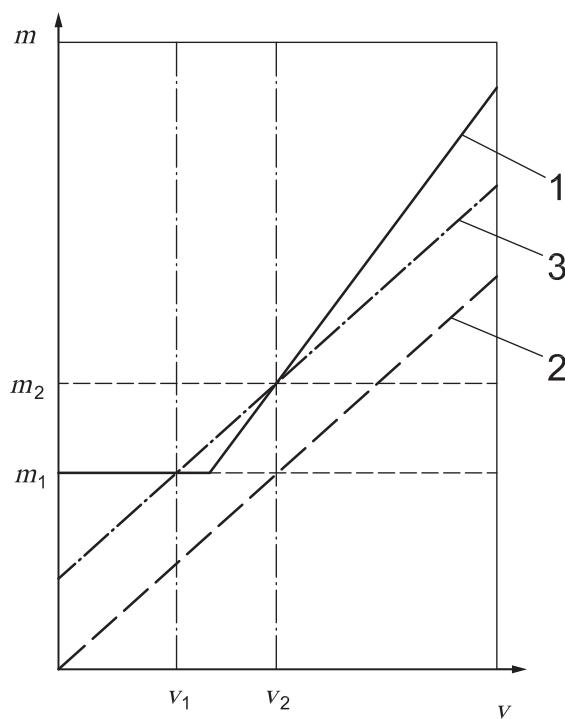
Table 2 — Example of failure modes of a sensor

Element	See Tables	Analyzed failure modes for 60/90/99 % DC		
		Low (60 %)	Medium (90 %)	High (99 %)
General Elements				
Sensors including signal switches	D.11	No generic fault model available. Detailed analysis necessary. Typical failure modes to be covered include <ul style="list-style-type: none">• Out-of-range• Stuck in range	No generic fault model available. Detailed analysis necessary. Typical failure modes to be covered include <ul style="list-style-type: none">• Out-of-range• Offsets• Stuck in range• Oscillations	No generic fault model available. Detailed analysis necessary. Typical failure modes to be covered include <ul style="list-style-type: none">• Out-of-range• Offsets• Stuck in range• Oscillations

Within this example, only the stuck-at a constant value m (in range) is evaluated. For a complete assessment of the residual failure rate of the sensor and the $M_{SPFM,Sensr}$, all other failure modes need evaluating.

For the analysis, we distinguish three different stuck-at fault scenarios for the sensor (see Figure 13):

- 1) sensor stuck-at value $m > m_2$;
- 2) sensor stuck-at value $m < m_1$; and
- 3) sensor stuck-at value m between m_1 and m_2 ;



- Key:**
- 1 safety-related lower boundary $\mu_{SafRel,A,min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,Master,wc}$

Figure 13 — Stuck-at fault scenarios

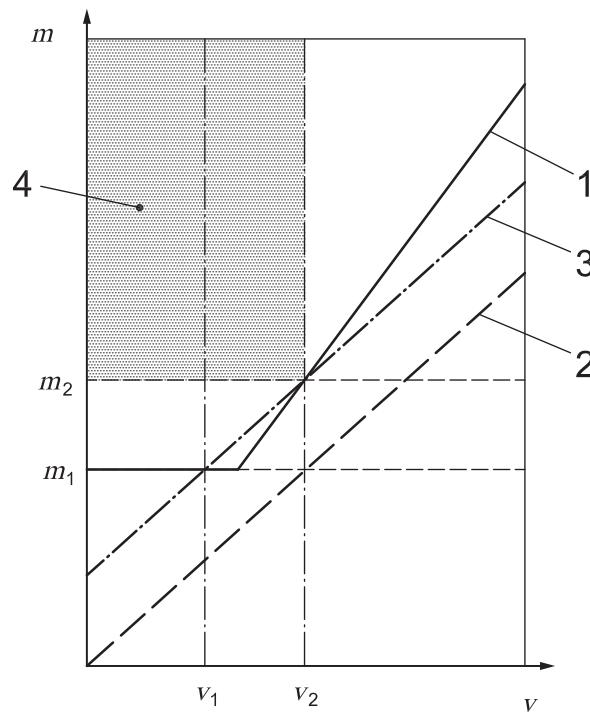
The impact of the stuck-at fault of the sensor at the system level depends on the current physical value v , e.g. a stuck-at m_2 fault has the potential to violate a safety goal for the physical values $v \leq v_2$. For values $v > v_2$ this fault does not have the potential to violate a safety goal. In the following analysis, the probability p_{RF} of a fault being a residual fault is evaluated considering the detection thresholds as well as the physical values v and their probability distribution.

8.2.4.2 Case 1: Sensor stuck-at value $m > m_2$ fault

If $v \leq v_2$, the fault has the potential to violate a safety goal (see Figure 14). The sensor deviation, however, is always above the worst case detection threshold $\mu_{A,\text{Master},wc}$, so the safety-related sensor failure is detected and controlled in time. Every fault is a detected dual-point fault. The probability p_{RF} of a residual fault in the case of $v \leq v_2$ is zero.

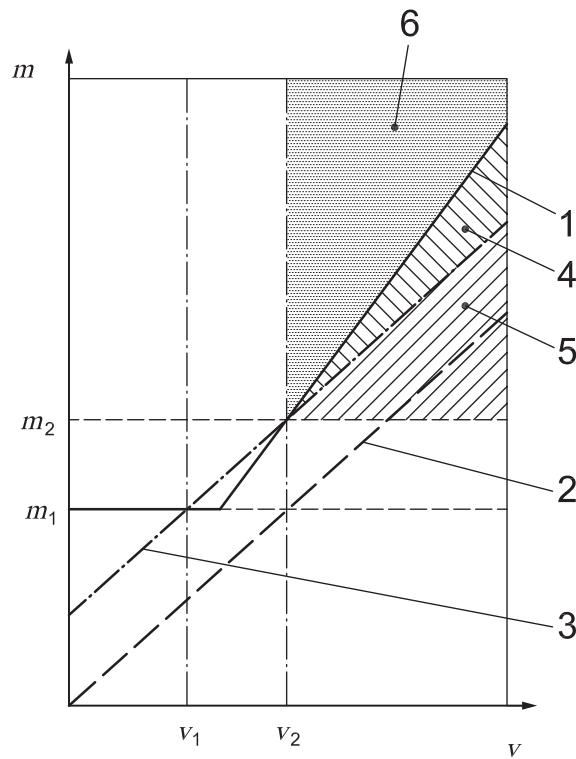
If $v > v_2$ the fault does not always have the potential to violate a safety goal (see Figure 15). If the fault has the potential to violate a safety goal (Figure 15, region 6), it will be above the worst case detection threshold and is detected in time. Note that some of the faults (Figure 15, regions 4 and 5) cannot be considered as safe, even though $v > v_2$, so they cannot lead to a violation of a safety goal, since they have the potential to violate a safety goal if $v \leq v_2$. Some of these faults lie above the worst case detection threshold and are detected (Figure 15, region 4). The probability p_{RF} of a residual fault in the case of $v > v_2$ is zero.

Stuck-at faults with $m > m_2$ have the potential to violate a safety goal if $v \leq v_2$, therefore they cannot be considered safe faults. Since all faults are detected and controlled before they can lead to the violation of a safety goal, they are detected dual-point faults; therefore, the probability $p_{RF_stuck@m>m_2}$ of a residual fault for a stuck-at $m > m_2$ fault is zero.



- Key:**
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\text{min}}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,\text{Master},wc}$
 - 4 dual-point faults, detected

Figure 14 — Fault classification for stuck-at $m > m_2$ fault, with $v \leq v_2$

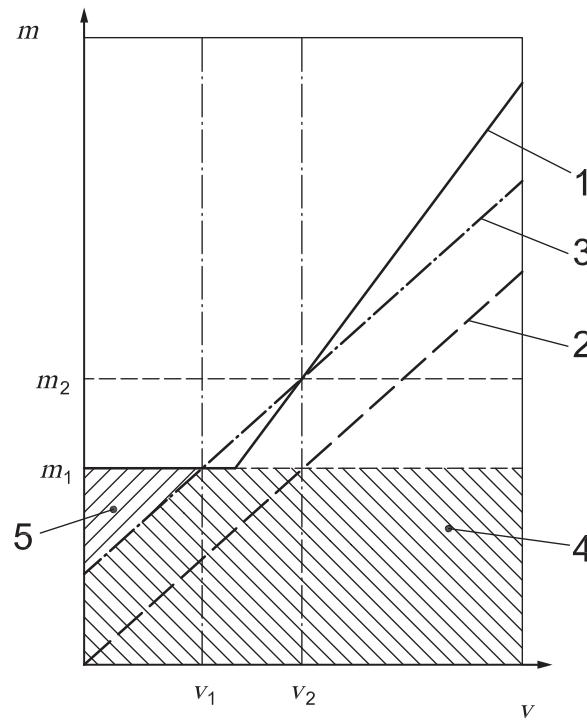


- Key:**
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,\text{Master},wc}$
 - 4 detected faults with no potential to violate the safety goal
 - 5 not detected faults with no potential to violate the safety goal
 - 6 dual-point faults, detected

Figure 15 — Fault classification for stuck-at $m > m_2$ fault, with $v > v_2$

8.2.4.3 Case 2: Sensor stuck-at value $m < m_1$ fault

Stuck-at faults with $m < m_1$ are visualized in Figure 16. These faults are safe faults, as they cannot lead to a safety-related failure, as they are always below the worst case detection threshold for the whole range of physical value v . Therefore, the resulting probability $p_{RF_stuck@m < m1}$ of a residual fault for the whole range of physical value v is zero.



- Key:
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,\text{Master},wc}$
 - 4 safe faults, undetected
 - 5 safe faults, detected

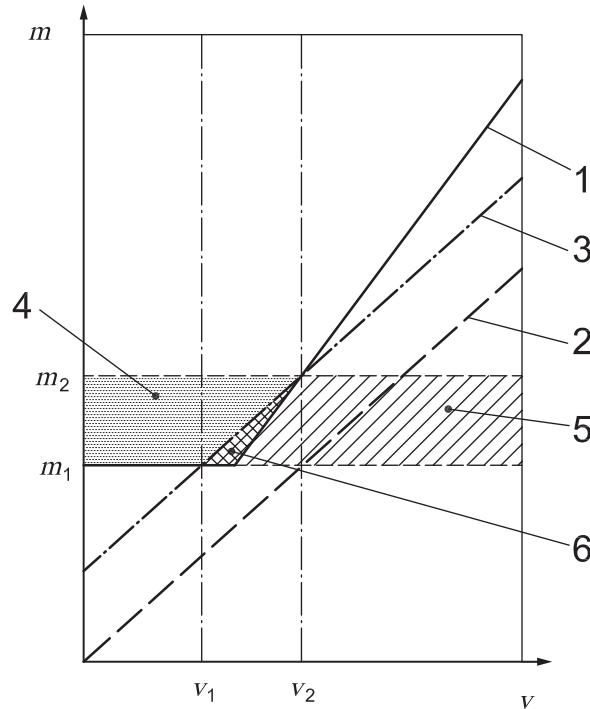
Figure 16 — Fault classification for stuck-at $m < m_1$ fault

8.2.4.4 Case 3: Sensor stuck-at value $m \in [m_1, m_2]$ fault

The potential to violate a safety goal and the detection of a stuck-at fault with $m \in [m_1, m_2]$ depend on the current physical value v (see Figure 17), i.e. the probability of a violation of a safety goal depends on the current value of v at the time when the fault occurs. The probability of a stuck-at residual fault, $p_{RF_stuck@m \in [m_1, m_2]}$, is evaluated for three different intervals of v at the time of fault occurrence:

- $v < v_1$;
- $v_1 \leq v \leq v_2$; and
- $v > v_2$.

For each of these conditions, the probability of a residual fault is evaluated separately. The final probability of a residual fault is calculated using the value of these three probabilities:



- Key:
- 1 safety-related lower boundary $\mu_{\text{SafRel},A,\min}$ of sensor A_Master
 - 2 return value of an ideal sensor with zero tolerance (as reference)
 - 3 worst case detection threshold $\mu_{A,\text{Master},wc}$
 - 4 dual-point faults, detected
 - 5 faults that do not violate the safety goal but remain undetected
 - 6 residual faults

Figure 17 — Fault classification for stuck-at $m \in [m_1, m_2]$ fault

Depending on the current value of v , the faults can be detected dual-point faults (region 4), residual faults (region 6) or do not have the potential to violate the safety goal (region 5).

$$\begin{aligned} P_{\text{RF,stuck}@m \in [m_1, m_2]} = & P_{\text{RF,stuck}@m \in [m_1, m_2], v < v_1} \times p_{v < v_1} \\ & + P_{\text{RF,stuck}@m \in [m_1, m_2], v_1 \leq v \leq v_2} \times p_{v_1 \leq v \leq v_2} \\ & + P_{\text{RF,stuck}@m \in [m_1, m_2], v > v_2} \times p_{v > v_2} \end{aligned}$$

where

$P_{\text{RF,stuck}@m \in [m_1, m_2]}$ is the probability that a stuck-at value m sensor fault, with $m \in [m_1, m_2]$, manifests itself as a residual fault;

$p_{\text{RF,stuck}@m \in [m_1, m_2], v < v_1}$ is the probability that a stuck-at value m sensor fault, with $m \in [m_1, m_2]$, manifests itself as a residual fault if the $v < v_1$ at the point of time when the fault occurs;

$p_{v < v_1}$ is the probability that $v < v_1$ at the point of time when the fault occurs;

$p_{\text{RF,stuck}@m \in [m_1, m_2], v_1 \leq v \leq v_2}$ is the probability that a stuck-at value m sensor fault, with $m \in [m_1, m_2]$, manifests itself as a residual fault if $v_1 \leq v \leq v_2$ at the point of time when the fault occurs;

$p_{v_1 \leq v \leq v_2}$ is the probability that $v_1 \leq v \leq v_2$ at the point of time when the fault occurs;

$p_{\text{RF}, \text{stuck}@m \in [m_1, m_2], v > v_2}$ is the probability that a stuck-at value m sensor fault, with $m \in [m_1, m_2]$, manifests itself as a residual fault if $v > v_2$ at the point of time when the fault occurs;

$p_{v > v_2}$ is the probability that $v > v_2$ at the point of time when the fault occurs;

$$p_{v < v_1} + p_{v_1 \leq v \leq v_2} + p_{v > v_2} = 1$$

If $v < v_1$, the stuck-at faults have the potential to violate a safety goal, but are detected in time. The probability $p_{\text{RF_stuck}@m \in [m_1, m_2], v < v_1}$ of a residual fault is zero.

If $v > v_2$, the stuck-at fault does not have the potential to violate a safety goal, but it is not detected. Since sooner or later the value v is in between v_1 and v_2 , $p_{\text{RF_stuck}@m \in [m_1, m_2], v > v_2} = p_{\text{RF_stuck}@m \in [m_1, m_2], v_1 \leq v \leq v_2}$.

If $v_1 \leq v \leq v_2$, the probability $p_{\text{RF_stuck}@m \in [m_1, m_2], v_1 \leq v \leq v_2}$ of a residual fault is not zero.

The exact determination of the probability of remaining in the residual fault area long enough to lead to a potential violation of a safety goal is not trivial. It can depend on parameters like:

- dynamic behaviour of the physical value v and its corresponding probability distributions, e.g. a temperature value is more of a static signal while the angle position of an electric motor in use is more of a dynamic signal;
- probability distribution of value v within $v \in [v_1, v_2]$;
- reaction time of the monitoring software, e.g. due to filtering times. In the example, a single event with $\Delta_A \geq \Delta_{\text{Max}}$ is enough to detect a sensor failure and switch into the safe state. It is common practice, however, to implement an error counter, so that more than one event is necessary in order to assess a sensor failure and switch into the safe state. Error counter recovery, e.g. resetting the error counter once one non-safety-related event (in this example, this would correspond to $\Delta_A < \Delta_{\text{Max}}$) is detected, can have a significant impact on the detection capability of the monitoring software, drastically reducing it;
- the number of measured safety-related sensor deviations necessary to lead to a potential violation of a safety goal. Also, the number of valid measurements that must lie between two measured safety-related sensor deviations, so that the safety goal is no longer violated, can be of interest.

If the exact detail of each influencing parameter is not available, it is legitimate to use expert judgement and engineering practises (e.g. using an equal distribution for unknown probability distributions) to derive a conservative estimate.

Having assessed the different probabilities $p_{\text{RF_stuck}@m > m_2}$, $p_{\text{RF_stuck}@m < m_1}$ and $p_{\text{RF_stuck}@m \in [m_1, m_2]}$, the probability $p_{\text{RF_stuck}@m}$ of a sensor stuck-at residual fault can be calculated:

$$p_{\text{RF}, \text{stuck}@m} = p_{\text{RF}, \text{stuck}@m < m_1} \times p_{m < m_1} + p_{\text{RF}, \text{stuck}@m \in [m_1, m_2]} \times p_{m_1 \leq m \leq m_2} + p_{\text{RF}, \text{stuck}@m > m_2} \times p_{m > m_2}$$

Where

$p_{m < m_1}$ is the probability of a stuck-at $m < m_1$ fault

$p_{m_1 \leq m \leq m_2}$ is the probability of a stuck-at $m_1 \leq m \leq m_2$ fault

$p_{m > m_2}$ is the probability of a stuck-at $m > m_2$ fault

$$p_{m < m_1} + p_{m_1 \leq m \leq m_2} + p_{m > m_2} = 1$$

8.2.4.5 Final residual failure rate assessment

If each relevant failure mode FM_i is assessed the same way as above, the overall probability $p_{RF, Sensor}$ of a sensor fault manifesting itself as a residual fault can be calculated:

$$p_{RF, Sensor} = \sum_i p_{FM,i} \times p_{RF, FM,i}$$

Where

$p_{FM,i}$ is the probability of failure mode FM_i

$p_{RF, FM,i}$ is the probability that failure mode FM_i manifests itself as a residual fault

$$\sum_i p_{FM,i} = 1$$

With this probability, the residual failure rate $\lambda_{RF, Sensor}$ can be assessed as

$$\lambda_{RF, Sensor} = p_{RF, Sensor} \times \lambda_{Sensor}$$

leading to a $M_{SPFM, Sensor}$ of

$$M_{SPFM, Sensor} = 1 - \frac{\lambda_{RF, Sensor}}{\lambda_{Sensor}} = 1 - p_{RF, Sensor}$$

8.2.4.6 Improvement of $SPFM_{Sensor}$

An efficient way to reduce the residual failure rate of the sensor is to reduce the value of Δ_{Max} . The reduction of Δ_{Max} could be done without a significant increase of false detection under the following conditions:

- The probability distribution of the tolerances could show that the estimated worst case scenario is extremely unlikely. Therefore, the probability of a false alarm is sufficiently low and therefore acceptable.
- A redesign of the system can lead to improved tolerance values.

Note that in this example only sensor faults are evaluated, not faults occurring in the remaining sensor path. The malfunction of shared HW resources which could lead to a malfunction of both sensors or which could falsify both sensor values, e.g. the ADC of the microcontroller, are evaluated separately. In addition, a dependent failure analysis as given in ISO 26262-9:2011, Clause 7 (Analysis of dependent failures) is done.

8.3 Further explanation concerning hardware

8.3.1 How to deal with microcontrollers in the context of ISO 26262 application

Microcontrollers are an integral component of modern E/E automotive systems. They can be developed as a safety element out of context (SEooC, see clause 9).

Their complexity is handled by combining qualitative and quantitative safety analyses of the microcontroller's parts and sub-parts, performed at the appropriate level of abstraction, i.e. from block diagram to the netlist and layout level, during the concept and product development phases.

Annex A is a guideline and a non-exhaustive list of examples of how to deal with microcontrollers in the context of ISO 26262.

It describes a method for the calculation of failure rates of a microcontroller, including how to consider permanent and transient faults.

It includes examples of:

- dependent failures analysis;
- avoidance of systematic failures during microcontroller design;
- verification of the safety mechanisms of the microcontroller; and
- consideration of the microcontroller stand-alone analysis at the system level.

8.3.2 Safety analysis methods

Annex B discusses techniques for analysing system fault modes, including inductive and deductive analysis, and includes an example of fault tree analysis.

8.3.3 Consideration of exposure duration in the calculation of Probabilistic Metric for random Hardware Failures (PMHF)

As described in ISO 26262-5:2011, 9.4.2.3, quantitative analysis provides evidence that target values of requirement ISO 26262-5:2011, 9.4.2.1 have been achieved. As given in ISO 26262-5:2011, 9.4.2.3, this quantitative analysis considers the exposure duration in the case of dual-point faults.

Based on the Note 2 in ISO 26262-5:2011, 9.4.2.3, the exposure duration starts as soon as the fault occurs.

It includes:

- the multiple-point fault detection interval associated with each safety mechanism, or the lifetime of the vehicle if the fault is not indicated to the driver (latent fault);
- the maximum duration of a trip (in the case where the driver is requested to stop in a safe manner); and
- the average time interval between a warning and the vehicle repair in a workshop (in the case where the driver is alerted to have the vehicle repaired).

The following example is provided to show a possible way to consider the exposure duration. In this example, it is assumed that an intended functionality (the mission block “m”) is supervised by a safety mechanism “sm”.

The value of the probabilistic metric for random hardware failures M_{PMHF} , considering the conditional probability that a failure of the safety mechanism is followed by a failure of the mission block, can be calculated using the formula:

$$M_{\text{PMHF}} = \frac{\lambda_{m,\text{RF}} \times T_{\text{Lifetime}} + \lambda_{m,\text{DPF}} \times T_{\text{Lifetime}} \times 0,5 \times (\lambda_{\text{sm,DPF,latent}} \times T_{\text{Lifetime}} + \lambda_{\text{sm,DPF,detected}} \times \tau_{\text{SM}})}{T_{\text{Lifetime}}}$$

Where

- | | |
|--------------------------|--|
| M_{PMHF} | is the value for the probabilistic metric for random hardware failures (PMHF) |
| $\lambda_{m,\text{RF}}$ | is the residual failure rate of the intended functionality (the mission block “m”) |
| $\lambda_{m,\text{DPF}}$ | is the dual-point failure rate of the mission block “m” |
| T_{Lifetime} | is the vehicle lifetime |

$\lambda_{sm,DPF,latent}$ is the latent dual-point failure rate of the safety mechanism "sm"

$\lambda_{sm,DPF,detected}$ is the detected dual-point failure rate of the safety mechanism "sm"

τ_{SM} is the multiple-point fault detection interval of the safety mechanism "sm"

If the term $\lambda_{m,DPF} * \lambda_{sm,DPF,detected} * \tau_{SM}$, representing the probability of a mission failure in combination

with a failure of the corresponding safety mechanism within one τ_{SM} , is very small [e.g. if τ_{SM} is in the order of one driving cycle (even with $\lambda_{m,DPF} = \lambda_{sm,latent} = 1000$ FIT the contribution $\leq 10^{-12}$ 1/h within this example)], it can be neglected, simplifying the formula:

$$M_{PMHF} = \lambda_{RF} + 0,5 \times \lambda_{m,DPF} \times \lambda_{sm,DPF,latent} \times T_{Lifetime}$$

If conditional probability does not apply, e.g. the order of failure is irrelevant, the formula changes to

$$M_{PMHF} = \lambda_{RF} + \lambda_{m,DPF} \times \lambda_{sm,DPF,latent} \times T_{Lifetime}$$

9 Safety element out of context

9.1 Safety element out of context development

The automotive industry develops generic elements for different applications and for different customers. These generic elements can be developed independently by different organizations. In such cases, assumptions are made about the requirements and the design, including the safety requirements that are allocated to the element by higher design levels and on the design external to the element.

Such an element can be developed by treating it as a safety element out of context (SEooC). An SEooC is a safety-related element which is not developed for a specific item. This means it is not developed in the context of a particular vehicle.

An SEooC can be a system, an array of systems, a subsystem, a software component, a hardware component or a part. Examples of SEooCs include system controllers, ECUs, microcontrollers, software implementing a communication protocol or an AUTOSAR software component.

An SEooC cannot be an item as the development of an item always requires the context of a vehicle intended for series production. In the case where the SEooC is a system, this system is not developed in this context, and therefore it is not an item.

SEooCs differ from qualified components described in ISO 26262-8:2011, Clause 12 (Qualification of software components) and ISO 26262-8:2011, Clause 13 (Qualification of hardware components):

- An SEooC is developed, based on assumptions, in accordance with ISO 26262. It is intended to be used in multiple different items when the validity of its assumptions can be established during integration of the SEooC.
- Qualification of software and hardware components addresses the use of pre-existing elements for an item developed under ISO 26262. The components are not necessarily designed for reusability nor developed under ISO 26262.

Table 3 describes the intended use of qualification, safety element out of context and the proven in use argument for different software elements.

The classification of the software components in the Table 3 is as described in ISO 26262-6:2011, 7.4.6.

Table 3 — Classification of the software components

Classification of software component	Part 6 in context of an item	Part 8 – 12 Qualification of SW component	Part 6 as safety element out of context	Part 8 – 14 Proven in use argument
Newly developed	Suitable	Not suitable	Suitable	Not suitable
Re-use with change	Suitable	Not suitable	Suitable	Suitable ^a
Re-use without change	Not suitable	Suitable	Suitable (if developed as SEooC)	Suitable

^a See ISO 26262-8:2011, 14.4.4.

When developing an SEooC, applicable safety activities are tailored as described in ISO 26262-2:2011, 6.4.5.6. Such tailoring for the SEooC development does not imply that any step of the safety lifecycle can be omitted. In case certain steps are deferred during the SEooC development, they are completed during the item development.

The ASIL capability of an SEooC designates the capability of the SEooC to comply with assumed safety requirements assigned with a given ASIL. Consequently, it defines the requirements of ISO 26262 that are applied for the development of this SEooC.

An SEooC is thus developed based on assumptions; on an intended functionality and use context which includes external interfaces. These assumptions are set up in a way that addresses a superset of items, so that the SEooC can be used later in multiple different, but similar, items. In the case where certain steps are deferred during the SEooC development, they are to be completed during the item development.

The validity of these assumptions is established in the context of the actual item while integrating the SEooC.

It is possible to build an item out of multiple SEooCs, with SEooCs interfacing directly with each other. In this case, the validity of the assumptions of one SEooC is established considering the interfacing SEooC.

In the case where the validity of the assumptions made during the SEooC development cannot be established during its integration into the item, either a change to the SEooC or to the item is to be made as described in ISO 26262-8:2011, Clause 8 (Change management).

9.2 Use cases

9.2.1 General

The development of an SEooC involves making assumptions on the prerequisites of the corresponding phase in the product development, e.g. for a software component, which is a part of the software architectural design, the corresponding phase is the subphase corresponding to ISO 26262-6:2011, Clause 7. It is not necessary to make assumptions on all prerequisites, e.g. safety plan.

Figure 18 shows the relationship between assumptions and SEooC development. The development of an SEooC can start at a certain hierarchical-level of requirements and design. Each piece of information on requirements or design prerequisites is pre-determined with the status "assumed".

The correct implementation of the requirements for the SEooC (derived from the assumed high-level requirements and assumptions on the design external to the SEooC) will be verified during the SEooC development.

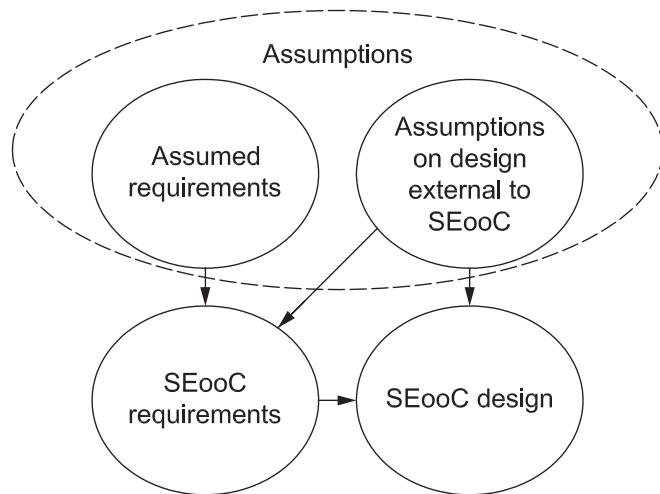


Figure 18 — Relationship between assumptions and SEooC development

The correct implementation of the requirements for the SEooC (derived from the assumed high-level requirements and assumptions on the design external to the SEooC) will be verified during the SEooC development. The validation of these requirements and assumptions are then established during the development of the item.

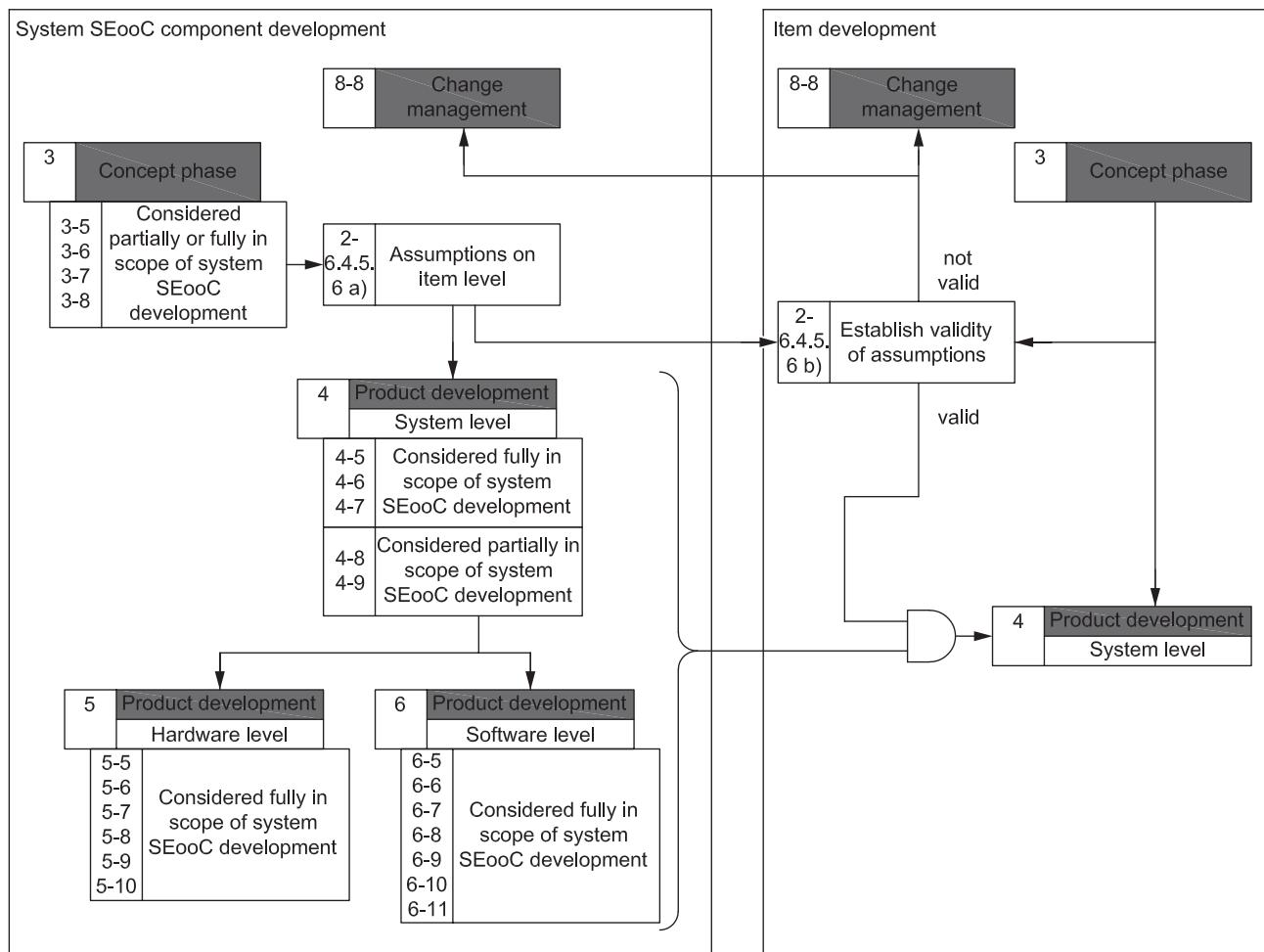
Similarly, verification activities demonstrate that a developed SEooC, at any level, is consistent with the requirements in the context where it is used. For example, when a software component, developed out of context, is used, the verification of the software specification can demonstrate that the requirements in the software architectural design specification are met. This verification report can be produced when development of the SEooC is finished and the item development reaches the phase where requirements on the safety element are formulated.

Some typical examples of SEooC are given below; namely a system, a hardware component and a software component.

9.2.2 Development of a system as a safety element out of context

This section is intended to show how the tailoring of the SEooC concept is applied to a new E/E system which can be integrated by different vehicle manufacturers.

For the purpose of this example, the system includes functionality to both activate a function under certain vehicle conditions and to allow the deactivation of the function on proper driver requests. The process flow is given in Figure 19.



NOTE 1 Some additional tailoring of the requirements can be necessary depending on the exact nature of the SEooC.

NOTE 2 Depending on the exact nature of the SEooC, some requirements of parts 3 and 4 cannot be applicable, and therefore only partial consideration is made.

NOTE 3 Although all the clauses of ISO 26262 are not shown, this does not imply that they are not applicable.

Figure 19 — SEooC system development

Step 1a – Definition of the scope of SEooC

Based on assumptions, the SEooC developer defines the purpose, functionalities and external interfaces of the SEooC.

Examples of such assumptions on the scope of the SEooC can be:

- The system is designed for vehicles with a gross mass up to 1800 kg.
- The system is designed for front-wheel driven vehicles.
- The system is designed for maximum road slope of 32 %
- The system has interfaces with other external systems to get the required vehicle information.
- Functional requirements:
 - The system activates the function when requested by the driver in certain vehicle conditions;

- The system deactivates the function when requested by the driver.

Step 1b – Assumptions on safety requirements for the SEooC

The development of an SEooC needs to make assumptions about the item definition, the safety goals of the item and the corresponding functional safety requirements related to the SEooC functionality in order to identify the technical safety requirements of the SEooC.

Examples of assumptions on the functional safety requirements allocated to the SEooC can be:

- The system does not activate the function at high vehicle speeds (ASIL x).
- The system does not deactivate the functionality when the driver request is not detected (ASIL y).

In order to achieve the assumed safety goals, specific assumptions on the context are defined.

Examples of assumptions on the context of the SEooC can be:

- An external source will provide information at the requested ASIL enabling the system to detect the proper vehicle condition (ASIL x).
- An external source will provide information about the driver request at the requested ASIL (ASIL y).

Step 2 – Development of the SEooC

When the technical safety requirements have been derived from the assumed functional safety requirements of the item, the SEooC is developed following the requirements of ISO 26262.

Step 3 – Work products

At the end of the SEooC development, the work products that show that the derived technical safety requirements are fulfilled are made available. All necessary information from the work products is then provided to the item integrator, including SEooC safety requirements and the assumptions made on the context.

Step 4 – Integration of the SEooC into the item

During item development, the safety goals and the functional safety requirements are specified. The functional safety requirements of the item are matched with the functional safety requirements assumed for the SEooC to establish their validity.

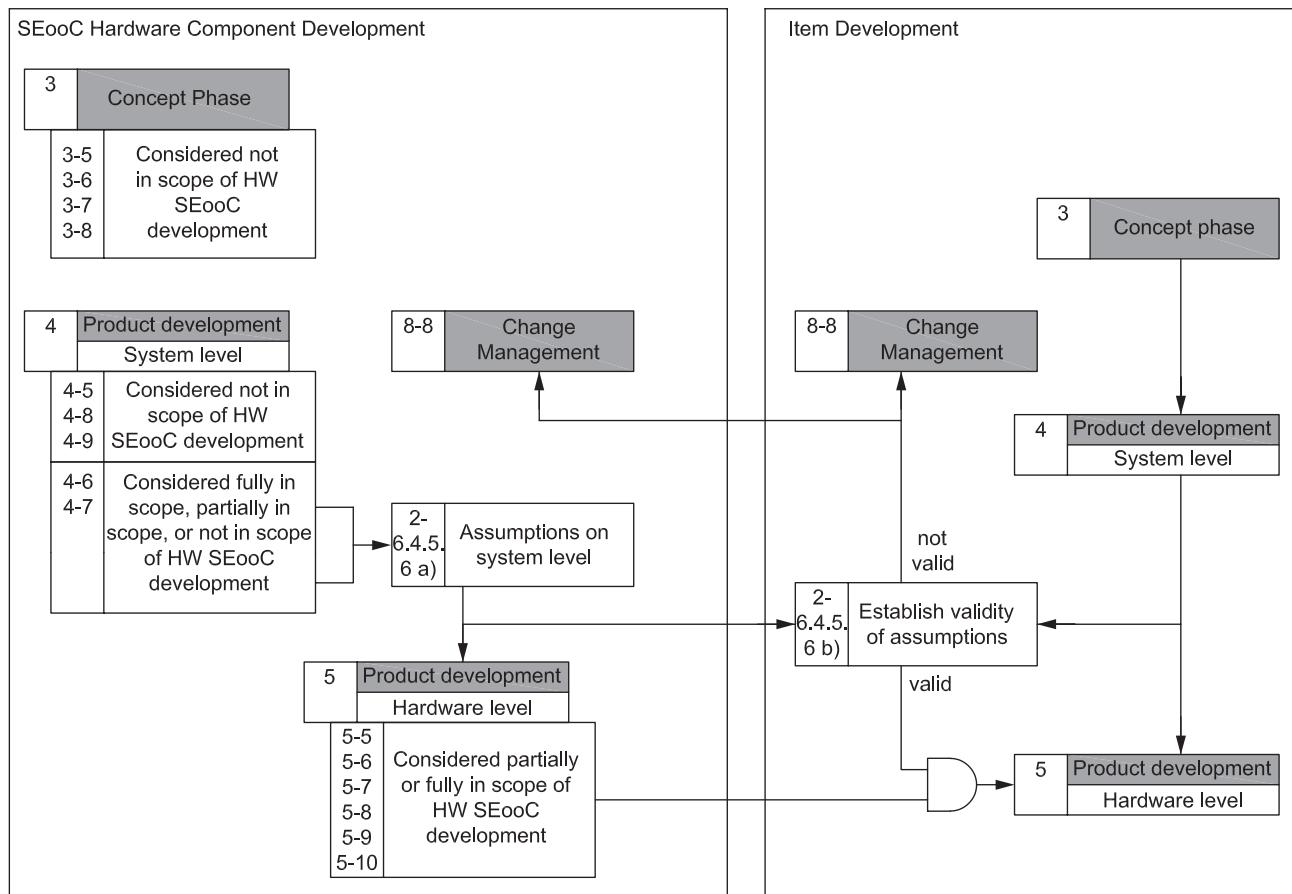
In the case of an SEooC assumption mismatch, a change management activity, beginning with an impact analysis, is conducted as described by ISO 26262-8:2011, Clause 8 (Change management). Potential outcomes include:

- the difference can be deemed to be acceptable with regard to the achievement of the safety goal, and no action is taken;
- the difference can be deemed to impact the achievement of the safety goal, and a change can be necessary to either the item definition or the functional safety concept;
- the difference can be deemed to impact the safety goal, and a change is required to the SEooC component (including possibly a change of component).

9.2.3 Development of a hardware component as a safety element out of context

9.2.3.1 General

This section uses the microcontroller (MCU) example of Annex A as an example hardware component SEooC. The process flow is given in Figure 20.



NOTE 1 Some additional tailoring of the requirements can be necessary depending on the exact nature of the SEooC, e.g. to adapt target values for the probability to violate a safety goal due to random hardware failure.

NOTE 2 Depending on the exact nature of the SEooC, some requirements of part 5 are not applicable, and therefore only partial consideration is made.

NOTE 3 Although all the clauses of ISO 26262 are not shown, this does not imply that they are not applicable.

Figure 20 — SEooC hardware component development

9.2.3.2 Step 1 – Assumptions on system level

The development of a microcontroller (MCU) (see Figure 20) as an SEooC starts (step 1) with an assumption of the system-level attributes and requirements as per ISO 26262-2:2011, 6.4.5.6.

This stage can be broken down into two sub-steps (1a and 1b) based on the analysis of some reference applications. The requirements are assumed with respect to the prerequisites for HW product development (ISO 26262-5:2011, Table A.1); examples follow.

9.2.3.3 Step 1a – Assumptions on technical safety requirements

Below are some example assumed technical safety requirements created for the MCU example:

Assumptions on technical safety requirements (step 1a)

- a) Failures of the CPU instruction memory are mitigated by safety mechanism(s) in hardware with at least the target value (e.g. 90 %) assigned for the single-point fault metric at the HW part level (might also be expressed in terms of required DC).

- b) The contribution of the MCU to the total probability of violation of a safety goal is no more than 10 % of the allowed probability for the relevant ASIL.
- c) The MCU implements a safe state defined as all I/O driving outputs to a low state when reset is asserted.
- d) Any safety mechanisms implemented related to the processing function completes in less than 10 milliseconds (assigned portion of the fault tolerant time interval).
- e) Debug interfaces of the MCU are not used during safety-related operation. Therefore, any faults in the debug logic will be considered safe faults.
- f) A memory protection unit is present to provide the possibility of separating software tasks with different ASILs.

ASIL capability is established at this step.

9.2.3.4 Step 1b – Assumptions on system-level design

Some examples of system-level design assumptions, external to the SEooC:

- a) The system will implement a safety mechanism on the power supply to the MCU to detect over voltage and under voltage failure modes.
- b) The system will implement a windowed watchdog safety mechanism external to the MCU to detect either clocking or program sequence failures of the MCU.
- c) A software test will be implemented to detect latent faults in the EDC safety mechanism of the MCU (SM4).
- d) A SW-based test (SM2) is executed at key-on to verify the absence of latent faults in the logical monitoring of program sequence of CPU (SM1).

9.2.3.5 Step 2 – Execution of hardware development

On the basis of these decisions (assumed technical safety requirements and assumptions related to the design external to the SEooC), the SEooC is developed (step 2) as written in ISO 26262-5, and each applicable work product is prepared. For example, the evaluation of safety goal violations due to random HW failures (see work product written in ISO 26262-5:2011, 9.5.1) is done considering the SEooC assumptions including any budget for FIT rate found in the assumed technical safety requirements. On the basis of the SEooC assumptions, the safety analyses and the analysis of dependent failures internal to the MCU are performed according to ISO 26262-9.

For the MCU example in A.3.5, the safety requirement a) is fulfilled because the single-point fault metric of memory is greater than the 90 % target assigned at that HW part level (99,8 %, permanent faults and 99,69 % transient faults). The assumption c) on system design is implemented by safety mechanism SM4.

9.2.3.6 Step 3 – Work products

At the end of the MCU product development (step 3), the necessary information from the work products is provided to the system integrator; this includes the following documentation: assumed requirements, assumptions related to the design external to the SEooC and applicable work products of ISO26262 (for example, the report on the probability of a violation of a safety goal due to random HW failure).

9.2.3.7 Step 4 – Integration of the SEooC into the item

When the MCU developed as an SEooC is considered in the context of the item HW product development phase, the validity of all SEooC assumptions including SEooC assumed technical safety requirements and the assumptions related to the design external to the SEooC are established (step 4). It is plausible that mismatches between SEooC assumptions and system requirements will occur. For example, the item

developer could decide not to implement an assumed external component. As a consequence, the evaluation of safety goal violations due to random HW failures done by the SEooC developer might no longer be consistent with the item.

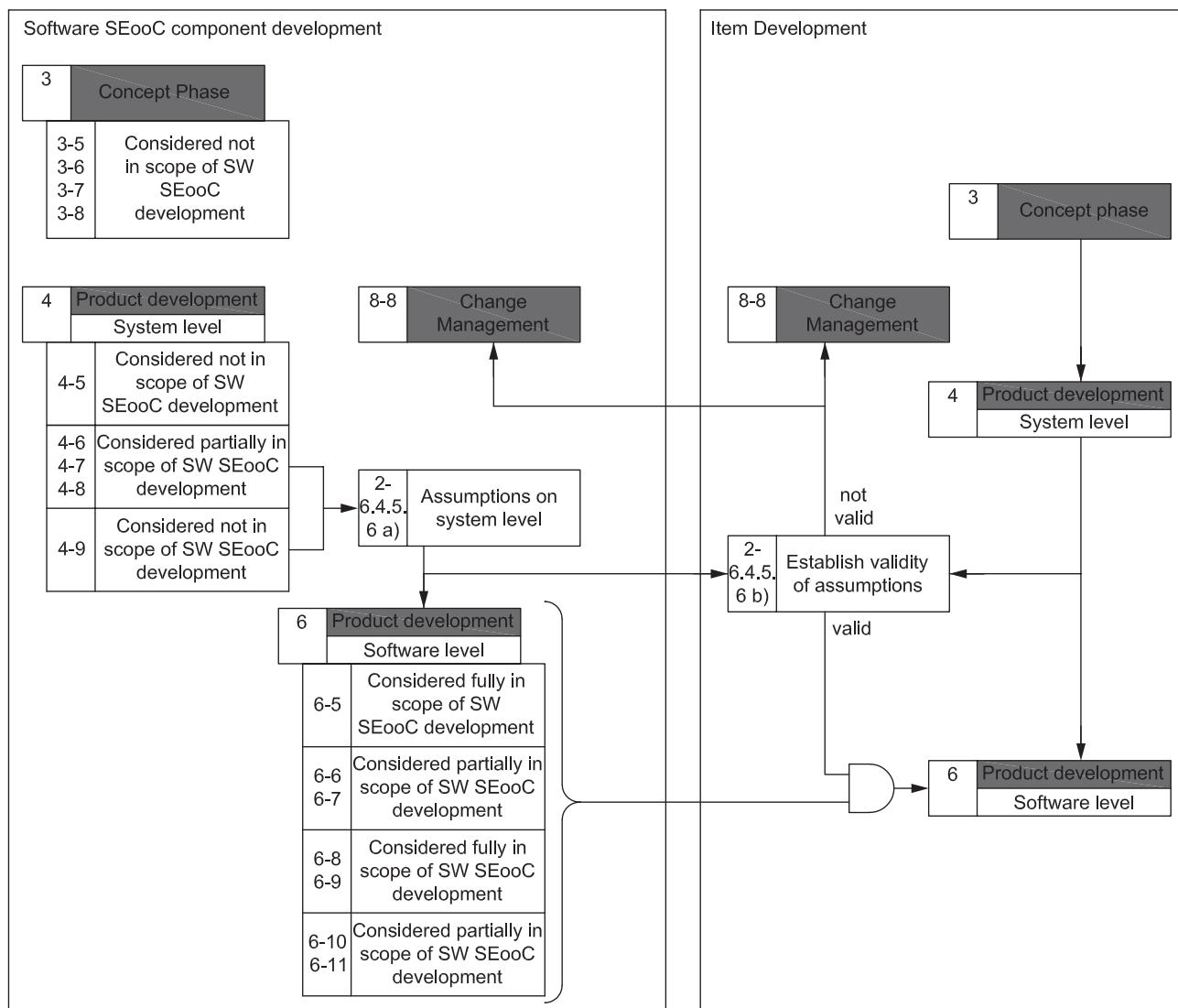
In the case of an SEooC assumption mismatch, a change management activity beginning with impact analysis is conducted as written in ISO 26262-8:2011, Clause 8 (Change management). Potential outcomes include:

- The difference can be deemed to be acceptable with regard to the achievement of the safety goal, and no action is taken.
- The difference can be deemed to impact the achievement of the safety goal, and a change can be necessary to either the functional safety concept or the technical safety requirements.
- The difference can be deemed to impact the achievement of the safety goal, and a change is required to the SEooC component (including possibly a change of component).
- The difference can be deemed to impact the achievement of the safety goal, and therefore safety metrics are recalculated, but the recalculated metrics show that the design meets the system targets, so no change is necessary.

9.2.4 Development of a software component as a safety element out of context

9.2.4.1 General

This section illustrates the different steps of the application of the SEooC concept to a new medium/low level software component. The process flow is given in Figure 21.



NOTE 1 Some additional tailoring of the requirements can be necessary depending on the exact nature of the SEooC.

NOTE 2 Depending on the exact nature of the SEooC, some requirements of part 6 are not applicable, and therefore only partial consideration is made.

NOTE 3 Although all the clauses of ISO 26262 are not shown, this does not imply that they are not applicable.

Figure 21 — SEooC software component development

9.2.4.2 Step 1a – Assumptions on the scope of the software component as an SEooC

This step is intended to state the relevant assumptions regarding the purpose of the software component, its boundaries, its environment and its functionalities.

Examples of such assumptions include:

- The software component is integrated into a given software layered architecture.

- Any potential interference caused by the software component is detected and handled by its environment.
- The software component provides the following functions: *list of the functional software requirements*.

9.2.4.3 Step 1b – Assumptions on the safety requirements of the software component

Step 1b is intended to make assumptions on higher level safety requirements that potentially impact the software component in order to derive its software safety requirements. For example, if a given set of data calculated by the software component is assumed to be of high integrity (ASIL x), then the resulting software safety requirements allocated to the SEooC can be:

- The software component detects any corruption on the following input data: *list of input data (ASIL x)*;
- The software component signals the following error conditions: *list of error conditions (ASIL x)*;
- A default value is returned with a fault status for any error condition detected (ASIL x);
- The software component returns the following results coded with CRC and a status (ASIL x).

9.2.4.4 Step 2 – Development of the software component

Once the necessary assumptions on the software component are explicitly stated, the SEooC is developed in accordance with the requirements of ISO 26262-6 corresponding to its ASIL capability (ASIL x in this example). All applicable work products are made available for further integration in different contexts, including the work products related to the verification of the assumed software safety requirements.

9.2.4.5 Step 3 – Integration of the software component in a new particular context

Before the software component is integrated with other software components in a new particular context, the validity of all the assumptions made on this SEooC are checked with regard to this context. This includes the assumed software safety requirements with their ASIL capability and all the assumptions made on the purpose, boundaries, environment and functionalities of the software component (see 9.2.4.2 and 9.2.4.3).

In the case where some assumptions regarding the software component do not fit with this new context, an impact analysis is initiated in accordance with ISO 26262-8:2011, Clause 8 (Change management). Potential outcomes of the impact analysis include:

- The discrepancies are acceptable with regard to the achievement of the safety requirements applicable at the software architectural design level, and no further action is taken.
- The discrepancies impact the achievement of the safety requirements applicable at the software architectural design level. Depending on the particular case, a change is applied in accordance with ISO 26262-8:2011, Clause 8 (Change management), either to the software component, or to the safety requirements applicable at the software architectural design level.

NOTE: In the case where the integration of a software component in a particular software architectural design results in the coexistence of software safety-related elements that have different ASILs assigned, the criteria for coexistence of elements are fulfilled as described in ISO 26262-9:2011, Clause 6 (Criteria for coexistence of elements), or alternatively the elements with lower ASILs are upgraded to the higher ASIL.

10 An example of proven in use argument

10.1 General

The item and its requirements described in this clause are an example. The safety goal, its ASIL and the following requirements are given to illustrate the proven in use argument defined in ISO 26262-8:2011, Clause 14 (Proven in use argument). This example does not reflect what the application of ISO 26262 on a similar real-life example would be.

10.2 Item definition and definition of the proven in use candidate

A vehicle manufacturer wants to integrate a new functionality into a new vehicle. For the purpose of this example, the item implementing this functionality is composed of sensors, one ECU that includes the complete hardware and software necessary to the functionality, and one actuator.

The incorrect activation of the functionality is ranked ASIL C by the vehicle manufacturer. The corresponding safety goal is derived into an ASIL C functional safety requirement allocated to the ECU.

The supplier of the ECU proposes to carry over an existing ECU already in the field.

The differences between the previous use of the ECU and its intended use in the new application are analysed. The analysis shows that the software has to be modified to implement the new functionality by changing calibration data, but the ECU hardware can be carried over without modification. The supplier intends to substitute the demonstration of compliance to requirements of ISO 26262-5 by a proven in use argument for the hardware of the ECU. The hardware of the ECU is therefore the proven in use candidate.

10.3 Change analysis

To establish a proven in use credit, the supplier performs a change analysis of the proven in use candidate.

This analysis shows that no change that could have an impact on the safety behaviour of the proven in use candidate has been introduced since the beginning of its production.

Moreover, the analysis shows that the differences between the previous use of the proven in use candidate and its intended use have no safety impact:

- the candidate's boundary is within the specification limits;
- the previous integration environment requires the same technical behaviour; and
- the cause and effects at the boundary of the candidate are the same in the previous and future integration environments.

10.4 Target values for proven in use

To establish the validity of the proven in use argument, the supplier estimates the number of cumulated hours the proven in use candidate has been in the field. The supplier also analyses the field data from the service period for any safety-related event, i.e. any reported event that would potentially cause, or contribute to, the violation of a safety goal or a safety requirement regarding the intended usage of the candidate in the new item.

The estimation of the duration of the service history is performed, based on the number of produced vehicles embedding the proven in use candidate, together with their production date and data on the typical usage of a vehicle in this segment of the market (number of driving hours per year).

The service history is based on the field return of the different vehicles embedding the proven in use candidate:

- Warranty claims;
- In-the-field defects analyses; or
- Return of defective parts from the vehicle manufacturers.

At the date of the initiation of the hardware development of the item, these analyses show that no safety-related event has occurred in the field. The total cumulated driving hours are estimated to be less than the target for the definite proven in use status for an ASIL C, but meet the interim service period as defined in ISO 26262-8:2011, 14.4.5.2.5.

The conclusion is then as follows:

- The development of the item can carry on taking credit that the hardware of the ECU is provisionally anticipated to be proven in use.
- The field observation continues to obtain a definite proven in use status (see ISO 26262-8:2011, 14.4.5.2.5 and ISO 26262-8:2011, 14.4.5.2.6).

11 Concerning ASIL decomposition

11.1 Objective of ASIL decomposition

The objective of ASIL decomposition is to apply redundancy in order to comply with the safety goal with respect to systematic failures. ASIL decomposition can result in redundant requirements and their corresponding decomposed ASILs implemented by sufficiently independent elements.

11.2 Description of ASIL decomposition

ASIL decomposition refers to the allocation of redundant safety requirements to sufficiently independent elements of the item. Redundancy in this context does not necessarily imply classical modular redundancy (see ISO 26262-1:2011, 1.94).

EXAMPLE The main processor of an ECU can be monitored by a redundant monitoring processor, both of which are independently capable of initiating a defined safe state, even if the monitoring processor is not able to fulfil the functional requirements allocated to the ECU.

ASIL decomposition can only be understood in the context of systematic failures, that is, the methods and measures applied to reduce the likelihood of these failures. The requirements on the evaluation of the hardware architectural metrics and the evaluation of safety goal violations due to random hardware failures will remain unchanged by ASIL decomposition (see ISO 26262-9:2011, 5.4.5).

EXAMPLE In the case of an ASIL B(D) decomposition, it is not allowed to decompose the ASIL D target for the evaluation of the hardware architectural metrics into separate ASIL B targets for each HW element. As written in ISO 26262-5:2011, 8.2, target values can be assigned to hardware elements, but those targets are assigned case-by-case based on an analysis started at the level of the whole hardware of the item. The target metric according to the safety goal applies at the item level.

In such a decomposed architecture, the relevant safety goal is only violated if both elements violate their decomposed safety requirements simultaneously.

The permitted decompositions in ISO 26262 are described in ISO 26262-9:2011, Clause 5 (Requirements decomposition with respect to ASIL tailoring).

11.3 An example of ASIL decomposition

11.3.1 General

The item and its requirements described in this clause are examples. The safety goal, its ASIL and the following requirements are only designed to illustrate the ASIL decomposition process. This example does not reflect what the application of ISO 26262 on a similar real-life example would be.

11.3.2 Item definition

Consider the example of a system with an actuator that is triggered on demand by the driver using a dashboard switch. For the purpose of this example, the actuator provides a comfort function if the vehicle is at zero speed, but can cause hazards if activated above 15 km/h.

For the purpose of this example, the initial architecture of the item is as follows:

- The dashboard switch input is read by a dedicated ECU (referred to as "AC ECU" in this example), which powers the actuator through a dedicated power line.
- The vehicle equipped with the item is also fitted with an ECU which is able to provide the vehicle speed. For the purpose of this example, the ability of this ECU to provide the information that the vehicle speed is greater than 15 km/h is assumed to be compliant with ASIL C requirements. This ECU is referred to as "VS ECU" in this section.

11.3.3 Hazard analysis and risk assessment

The hazardous event considered in the analysis is the activation of the actuator while driving at a speed above 15 km/h, with or without a driver request.

For the purpose of the example, the ASIL associated to this hazardous event is classified as ASIL C.

11.3.4 Associated safety goal

Safety Goal 1: Avoid activating the actuator while the vehicle speed is greater than 15 km/h : ASIL C

11.3.5 Preliminary architecture and safety concept

11.3.5.1 General

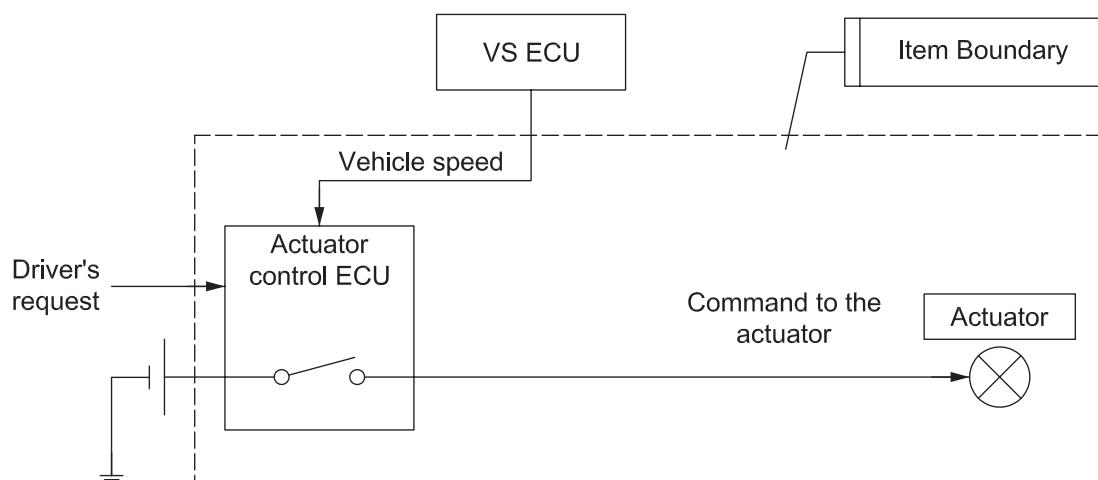


Figure 22 — Item boundary

11.3.5.2 Purpose of the elements (initial architecture)

- The dynamic VS ECU provides the Actuator Control ECU (AC ECU) with the vehicle speed.
- The AC ECU monitors the driver's requests, tests if the vehicle speed is less than or equal to 15 km/h, and if so commands the actuator.
- The actuator is activated when it is powered.

11.3.6 Functional safety concept

11.3.6.1 General

This example of a functional safety concept is only being used as an illustration of the ASIL decomposition; it is not intended to be exhaustive and does not include all the functional safety requirements.

- Requirement A 1: The VS ECU sends the accurate vehicle speed information to the AC ECU. =>ASIL C

- Requirement A 2: The AC ECU does not power the actuator if the vehicle speed is greater than 15 km/h.
=> ASIL C
- Requirement A 3: The actuator is activated only when powered by the AC ECU. => ASIL C

11.3.6.2 Evolved safety concept of the item

The developers can choose to introduce a redundant element, here a Safety Switch, as illustrated in Figure 23. By introducing this redundant element, the AC ECU is developed with an ASIL that is equal to or lower than ASIL C, in accordance with the results of an ASIL decomposition.

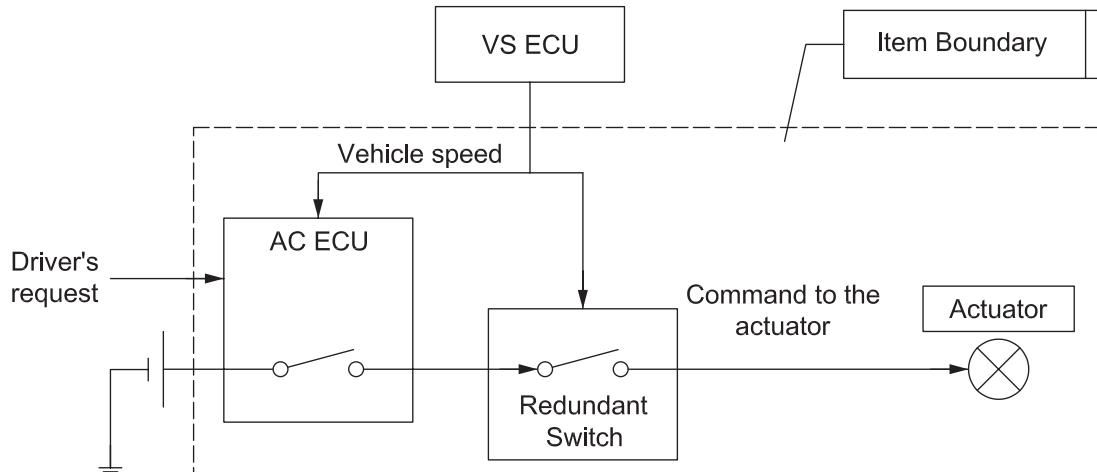


Figure 23 —Second iteration on the item design

Purpose of these elements (evolved architecture):

- The VS ECU control unit provides the AC ECU with the vehicle speed.
- The AC ECU monitors the driver's requests, tests if the vehicle speed is less than or equal to 15 km/h, and if so commands the actuator.
- The Redundant Switch is located on the power line between the AC ECU and the actuator. It switches on if the speed is less than or equal to 15 km/h, and off whenever the speed is greater than 15 km/h. It does this regardless of the state of the power line (its power supply is independent).
- The actuator operates only when it is powered.

Functional safety requirements:

- Requirement B 1: the VS ECU sends accurate vehicle speed information to the AC ECU. => ASIL C
- Alternatively: the incorrect transmission that vehicle speed is less than or equal to 15 km/h is prevented.
=> ASIL C
- Requirement B 2: the AC ECU does not power the actuator if the vehicle speed is greater than 15 km/h
=> ASIL X (C) (see Table 4)
- Requirement B 3: the VS ECU sends accurate vehicle speed information to the Redundant Switch. => ASIL C
- Requirement B 4: The Redundant Switch is in an open state if the vehicle speed is greater than 15 km/h.
=> ASIL Y (C) (see Table 4)

- Requirement B 5: The actuator operates only when powered by the AC ECU and the Redundant Switch is closed. => ASIL C

To permit an ASIL decomposition, the developers add an independency requirement if deemed necessary:

- Requirement B 6: Sufficient independence of the AC ECU and the Redundant Switch is shown. => ASIL C

The original requirement A2 has been replaced by the redundant requirements B2 and B4, both of which comply with the safety goal, and therefore ASIL decomposition can be applied.

Table 4 — Possible decompositions

	Requirement B2 : ASIL X(C)	Requirement B4 : ASIL Y(C)
Possibility 1	ASIL C(C) requirements	QM(C) requirements
Possibility 2	ASIL B(C) requirements	ASIL A(C) requirements
Possibility 3	ASIL A(C) requirements	ASIL B(C) requirements
Possibility 4	QM(C) requirements	ASIL C(C) requirements

Annex A (informative)

ISO 26262 and microcontrollers

A.1 General

The objective of this chapter is to give a non-exhaustive list of examples about how to deal with microcontrollers in the context of ISO 26262 application.

A.2 A microcontroller, its parts and sub-parts

A microcontroller (also MCU or μ C) is a small computer on a single integrated circuit consisting internally of a CPU, clock generator, timers, peripherals, I/O ports and memory. Program memory in the form of Non Volatile Memories (e.g. FLASH or OTP ROM) is also often included on the chip, as well as a certain amount of RAM.

As shown in the figure below, the whole microcontroller hierarchy can be seen as a component and the processing unit (e.g. a CPU) as a part. As explained in the example of 4.2 and further detailed in paragraph A.3.3, in certain cases (e.g. depending on the type of used safety mechanisms at the microcontroller or system level), each part could be further divided in sub-parts (e.g. the CPU register bank and its internal registers).

This represents a logical view of the microcontroller. It does not necessarily translate into its physical implementation and does not necessarily represent the dependencies between the parts and sub-parts.

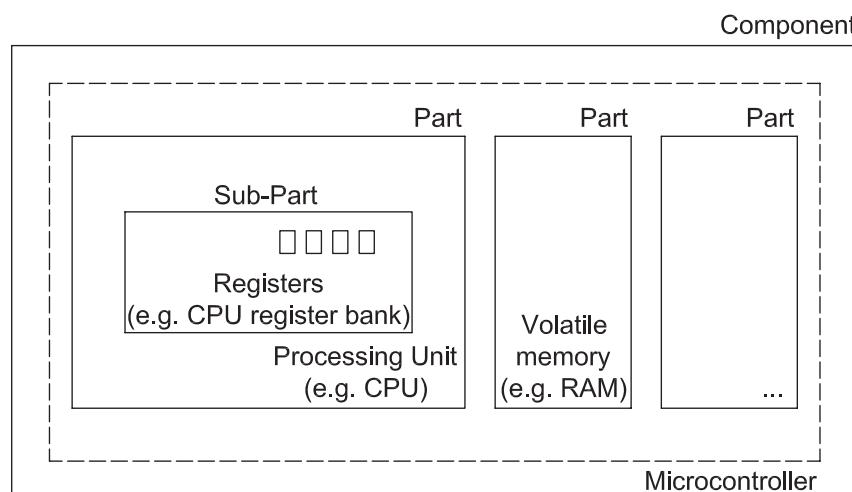


Figure A.1 — A microcontroller, its parts and sub-parts

ISO 26262-5:2011, Annex D, and specifically ISO 26262-5:2011, Table D.1, gives a list of parts and sub-parts of a microcontroller. Parts or sub-parts not included in ISO 26262-5:2011, Table D.1 can be classified considering analogies with parts or sub-parts therein defined: Table A.1 gives some examples.

Table A.1 — Example of classification of parts or sub-parts of a microcontroller as seen in ISO 26262-5

Elements in ISO 26262-5:2011, Table D.1	Examples for a microcontroller	
	Part	Sub-part
Power Supply	Embedded Voltage Regulator (EVR), Power Management Unit (PMU)	
Clock	Phase Locked Loop (PLL), Ring oscillator, Clock Generation Unit (CGU), Clock tree	
Non-volatile memory	FLASH, EEPROM, ROM, One-Time-Programmable (OTP) Memory	Memory cells array, Address Decoder, interface circuitry, Test/Redundancy logic, Memory controllers
Volatile Memory	RAM, Caches	Memory cells array, Address Decoder, interface circuitry, Test/Redundancy logic, Memory controllers
Analogue I/O and Digital I/O	General Purpose IOs (GPIO), Pulse-Width Modulator (PWM)	
	Analogue-Digital Converter (ADC), Digital-Analogue Converter (DAC)	
Processing Unit	CPU	Arithmetic Logic Unit (ALU), Data Path of CPUs
		Register Bank, internal RAM of CPU such as small data caches
		Load/Store Unit and bus interfaces
		Sequencer, coding and execution logic including flag registers and stack control
		CPU local memory controllers including cache controllers
	Interrupt Controller	Configuration registers of Interrupt Controller
	General purpose timers	
Communication	On-chip communication including bus-arbitration	Bus matrices/switch fabric; Protocol, data width, and clock domain conversion (e.g. bus bridges)
	On-chip communication using Direct Memory Access (DMA)	DMA addressing logic, DMA addressing registers, DMA buffering registers
	Serial-Peripheral Interface (SPI), Serial-Memory Interface (SMI), Inter-Integrated Circuit (I2C) interface, Controller Area Network (CAN) interface, Time Triggered CAN (TTCAN), FlexRay, Local Interconnect Network (LIN), Single Edge Nibble Transmission (SENT), Ethernet, Distributed Systems Interface (DSI), Peripheral Sensor Interface (PSI5)	

NOTE This table is an example: the part / sub-part list and partitioning of the microcontroller can be different.

A.3 Overview of microcontroller development and safety analysis as seen in ISO 26262

A.3.1 General

If a microcontroller is developed as a part of an item development compliant with ISO 26262, it is developed based on the safety requirements, which are derived from the top-level safety goals of the item. Targets for HW architectural metrics and Probabilistic Metric for random Hardware Failures are allocated to the item: in this case, the microcontroller is just one of the elements. As seen in the example of ISO 26262-5:2011, 8.2, to facilitate distributed developments, target values can be assigned to the microcontroller itself. The safety analysis of a microcontroller is performed based on the requirements and recommendations defined in ISO 26262-5:2011, 7.4.3 and in ISO 26262-9:2011, Clause 8 (Safety analysis).

On the other hand, in the case that the target item does not yet exist, the microcontroller can be developed as a safety element out of context (SEooC, refer to Clause 8 of this part). In this case, the development is done based on assumptions on the conditions of the microcontroller usage (Assumptions of Use), and then the validity of the assumptions is established based on the microcontroller requirements derived from the safety goals of the item in which the microcontroller is to be used.

The analyses and related examples described in the following part of this section are done assuming the microcontroller is an SEooC, but the described methods (e.g. the method for failure rates computation of a microcontroller) are still valid if the microcontroller is not considered a SEooC. When those analyses are conducted considering the stand-alone microcontroller, appropriate assumptions are made. Section A.3.9 describes how to adapt and verify those analyses and assumptions at system level. At the stand-alone microcontroller level, each requirement of ISO 26262-5, ISO 26262-8 and ISO 26262-9 (e.g. related to safety analyses, dependent failures analysis, verification, etc.) remains valid.

A.3.2 Qualitative and quantitative analysis of a microcontroller

As seen in ISO 26262-9:2011, 8.2, 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 microcontroller:

- a) Qualitative analysis is useful to identify failures. One of the possible ways in which it can be performed uses information derived from microcontroller block diagrams and information derived from ISO 26262-5:2011, Annex D.

NOTE 1 ISO 26262-5:2011, Annex D can be used as a starting point for diagnostic coverage (DC) with the claimed DC supported by a proper rationale.

NOTE 2 Qualitative analysis includes dependent failure analysis of this part as seen in A.3.6 (Example of dependent failures analysis).

- b) Quantitative analysis is performed using a combination of:

- i) Logical block level structuring;
- ii) Information derived from the microcontroller Register Transfer Level (RTL) description (to obtain functional information) and gate-level net list (to obtain functional and structural information);
- iii) Information to evaluate potential unspecified interaction of sub-functions (dependent failures, see section A.3.6);
- iv) Layout information - only available in the final stage;
- v) 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

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

NOTE 1 The analysis of dependent failures is performed on a qualitative basis because no general and sufficiently reliable method exists for quantifying such failures.

NOTE 2 This information can be progressively available during the microcontroller development phase. Therefore, the analysis could be repeated based on the latest information.

EXAMPLE 1 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.

EXAMPLE 2 During a first step of the quantitative analysis, a pre-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 (DFT = Design for Test).

- c) Since the parts and sub-parts of a microcontroller can be implemented in a single physical component, both dependent failures analysis and analysis of independence or freedom from interference are important analyses for microcontrollers. See paragraph A.3.6 for further details.

A.3.3 A method for failure rates computation of a microcontroller

A.3.3.1 General

Requirements and recommendations for the failure rates computation in general are defined in ISO 26262-5, and requirements about the computation of metrics are given in its Annex C.

Following the example given in ISO 26262-5:2011, Annex E, the failure rates and the metrics can be computed in the following way for microcontrollers:

- First, the microcontroller is divided into parts or sub-parts.

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

NOTE 2 The necessary level of detail (e.g. if 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 microcontroller or at the system level).

EXAMPLE 1 If the functionality of the CPU is monitored by a different CPU running in lockstep, the analysis does not need to consider each and every CPU internal register while more detail can be needed for the lock-step comparator. If on the other hand the functionality of the CPU is monitored by a SW self-test, a detailed analysis of the various CPU sub-parts can be appropriate.

EXAMPLE 2 The confidence of the computation is 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 microcontrollers (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 microcontroller 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:

- 1) If the total failure rate of the whole microcontroller 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 microcontroller die multiplied by the total failure rate.

NOTE 1 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.

NOTE 2 A detailed knowledge of the microcontroller can be useful.

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

- 2) If the base failure rates, i.e. the failure rate of basic sub-parts like gates of the microcontroller, 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 1 A detailed knowledge of the microcontroller can be useful.

NOTE 2 See paragraph A.3.4 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.

EXAMPLE 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 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 1 The effectiveness of safety mechanisms could be affected by dependent failures. Adequate measures are considered as listed in section A.3.6.

NOTE 2 Since the fault detection ability of the vehicle driver cannot be considered at this level of analysis, the concept of perceived fault is not applicable at microcontroller level. See paragraph A.3.9 for further details about how to combine microcontroller level information with the application.

NOTE 3 Due to the complexity of modern microcontrollers (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 paragraph A.3.8.2 for further details. Fault injection is not the only method, and other approaches are possible as described in paragraph A.3.8.2.

A.3.3.2 How to consider transient faults

As seen in Note 2 of ISO 26262-5:2011, 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.

When the quantitative approach is used, failure rates and metrics for transient faults can be computed the following example given in ISO 26262-5:2011, Annex E, supported by the following method:

- First, the microcontroller is divided into parts or sub-parts as for paragraph A.3.3

NOTE 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 microcontroller. Therefore, as recommended in Note 1 of ISO 26262-5:2011, 8.4.7, it can be helpful to compute a separate failure rate (and metric) for RAM memories and for the other parts of the microcontroller.

- Second, the failure rates of each part or sub-part are computed using the base failure rate for transient faults.

EXAMPLE Following the method defined in A.3.3, the base failure rate can be computed as a function of the base failure rate with respect to single-event upset and single-event transient and the related interested portion of circuit (for example expressed in number of flip-flops and gates). See paragraph A.3.4 for examples of how to derive the base failure rate values.

- Finally, the evaluation is completed by classifying the faults into safe faults and residual faults, i.e. the amount of safe faults related to the failure rate of that part or sub-part.

NOTE For estimations of the amount of safe transient faults, when there is a clear dependency from the application software and if that software is not available during the microcontroller development, a 50 %-50 % estimation could be acceptable. When the application software is available or if there is a direct dependency on microcontroller architecture, a specific analysis to determine this value could be preferable.

EXAMPLE 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 1 As seen in Note 2 of ISO 26262-5:2011, 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 repaired, 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 could not be valid and additional measures can be necessary, though these can be addressed on a case by case basis.

NOTE 2 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 3 Some of the coverage values of safety mechanisms defined in tables from D.2 to D.14 of ISO 26262-5:2011, 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 The typical value of the coverage of RAM March test (ISO 26262-5:2011, Table D.6) is rated HIGH. However in the related description (ISO 26262-5:2011, D.2.5.3), 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.

When the qualitative approach is used, a rationale is given based on the verification of the effectiveness of the safety mechanisms implemented (either internal to the microcontroller or at system level) to cover the transient faults.

EXAMPLE For data path elements, time-redundancy in processing of data (i.e. process the same information more than once) would already guarantee a high level of protection against transient faults.

A.3.4 How to derive base failure rates that can be used for microcontrollers

A.3.4.1 General

As seen in ISO 26262-5:2011, 8.4.3, failure rates data can be derived from a recognized industry source. The following list gives an example of standards and handbooks from which it is possible to derive the base failure rates for the method defined in paragraphs A.3.3 and A.3.3.2:

- for permanent faults: data provided by semiconductor industries or use of standards such as IEC/TR 62380 [8]; SN 29500 [6];

NOTE 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 provided as a maximum FIT based on a sampling statistics confidence level.

- for transient faults: data provided by semiconductor industries derived from JEDEC standards such as JESD89; International Technology Roadmap for Semiconductor (ITRS).

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

A.3.4.2 Example of microcontroller die FIT rate calculation per IEC/TR 62380

ISO 26262-5:2011, 8.4.3 states that failure rates data can be derived from a recognized industry source, for example IEC/TR 62380, IEC 61709, MIL HDBK 217 F notice 2, RIAC HDBK 217 Plus. The following is an example of estimation of hardware FIT rate as needed to support quantitative analysis using the methods detailed in IEC/TR 62380 [8]. The FIT rate model for a semiconductor per IEC/TR 62380 considers the failure rate of the device to be the sum of three subcomponents: microcontroller die, package and interface electrical over-stress effects.

NOTE 1 FIT corresponds to 1 failure per 10^9 hours of device operation.

A.3.4.2.1 Example of microcontroller die failure rate calculation

To compute the base microcontroller die FIT rate component (i.e. before application of de-rating for operating conditions), it is necessary to consider four key elements:

- λ_1 , the basic FIT rate per transistor, based by the process technology;
- N, the number of implemented transistors;
- α , a de-rating factor for process maturity; as process technology matures, per-transistor failure rate tends to reduce exponentially to an asymptotic level; and
- λ_2 , the process technology driven FIT rate that does not scale with number of transistors or age.

Those factors are combined using the formula in clause 7.3.1 of IEC/TR 62380:2004 [8] ("MATHEMATICAL MODEL").

Selection of parameters can be done based on the process technology and type of circuitry utilized by the design. Values are available in Table 16 of [8] for CMOS logic, analogue and multiple memory types (SRAM, DRAM, EEPROM, flash EEPROM, etc.).

Table A.2 shows the computation of the failure rates used in the quantitative example of paragraph A.3.5. For the process maturity de-rating factor, 2008 is considered as manufacturing year.

Table A.2 — Example of the computation of the failure rates

Circuit Element	λ_1	N	α	λ_2	Base FIT
50k gate CPU	$3,4 \times 10^{-6}$	200000 (4 transistors/gate)	10	1,7	1,72
16kB SRAM	$1,7 \times 10^{-7}$	786432 (6 transistors/bit for a low-consumption SRAM)	10	8,8	8,802
Sum					10,52

NOTE 1 Multiple values of λ_1 and λ_2 can be valid for a given circuit type. In such case, the party performing the estimation ensures the value selected best matches the metrics for the specific manufacturing technology utilized and provides appropriate rationale.

NOTE 2 To simplify calculation, estimation can be done using a single selection of λ_1 and λ_2 for the entire device.

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. [8] 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 [9].

NOTE 4 To calculate the microcontroller 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 5 For analogue parts or for the microcontroller built primarily on analogue process technologies, the "Linear Circuits" entry of Table 16, "MOS : Standard circuits (3)" in [8] can be used, unless more precise data are provided by the microcontroller vendor.

NOTE 6 For computation of I/O contribution to the FIT rate, they can be considered in terms of number of equivalent transistors of CMOS digital logic as seen in Note 4, unless more precise data are provided by the microcontroller vendor.

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

- Junction temperature of the microcontroller die, which is calculated based on:
 - power consumption of the microcontroller die;
 - 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. [8] provides two automotive reference profiles: "motor control" and "passenger compartment".
- Activation energy and frequency per technology type to complete the Arrhenius Equation.

For this example, we assume a CMOS technology based MCU which consumes 0,5 W power. The microcontroller 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 of [8], this results in a de-rating factor of 0,17.

When the de-rating factor is applied, we have an effective FIT rate per component as shown in Table A.3.

Table A.3 — Example of effective FIT rate per component

Circuit Element	Base FIT	De-rating for temp	Effective FIT
50k gate CPU	1,72	0,17	0,29
16kB SRAM	8,80	0,17	1,50
Sum			1,79

NOTE 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 [8] to achieve a more accurate estimation of FIT rate.

A.3.4.2.2 An alternative calculation for the de-rating factor of IEC/TR 62380

Even though the failure rate provided by [8] is more in line with current reliability data, it can be useful to provide more conservative data, e.g. to be more in line with older failure rate handbooks like the SN 29500. This task can be achieved by a slight change of the used temperature de-rating factor.

The formula used in clause 7.3.1 of [8] ("MATHEMATICAL MODEL") to calculate the temperature de-rating factor δ_T uses the following parameters:

$(\pi_t)_i$: i^{th} temperature factor related to the i^{th} junction temperature of the integrated circuit mission profile

τ_i : i^{th} working time ratio of the integrated circuit for the i^{th} junction temperature of the mission profile

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

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 $\delta_{T,\text{conservative}}$:

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

Applying these formulas leads to a temperature "de-rating" factor of 2,91, leading to results shown in Table A.4

Table A.4 — Example of effective FIT rate per component

Circuit Element	Base FIT	De-rating for temp	Effective FIT
50k gate CPU	1,72	2,91	5,01
16kB SRAM	8,80	2,91	25,61
Sum			30,62

A.3.4.2.3 Example of package failure rate

The package failure rate is calculated using the formula shown in clause 7.3.1 of [8] ("Mathematical expression of the influence factor π_α ") and using the following parameters:

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

$(\pi_n)_i$: i^{th} influence factor related to the annual cycles number of thermal variations seen by the package, with the amplitude ΔT_i

ΔT_i : i^{th} thermal amplitude variation of the mission profile

λ_3 : base failure rate of the integrated circuit package

For this example, we assume a CMOS technology based MCU which consumes 0,5 W power. The microcontroller die is packaged in a 144 pin quad flat package and cooled by natural convection, leading to a junction temperature increase of $\Delta T_j = 26,27$ °C. The MCU is exposed to the "motor control" temperature profile.

The influencing factor π_α is calculated using the formula shown in clause 7.3.1 of [8] ("Mathematical expression of the influence factor"), with α_s , α_c being the linear thermal expansion coefficients for the substrate and 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 $\alpha_c = 21,5$.

Since for a automotive profile the number of cycles/year $\leq 8\ 760(\pi_n)_i$ is calculated using the formula in clause 7.3.1 of [8] ("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 [8].

Using the "motor control" temperature profile, this results in a total failure rate for the package of

$$\lambda_{\text{package}} = 207 \text{ FIT}$$

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

$$\lambda_{\text{pin}} = 1,44 \text{ FIT}$$

NOTE 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 MCU supplier and system implementer can be used.

A.3.4.2.4 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 [8] ("MATHEMATICAL MODEL"). If the device has a direct connection to the external environment, i.e. the device is an interface, π_l is equal to one. If the device is not an interface, i.e. it has no direct connection to the external environment, π_l is equal to zero.

[8] offers 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:

$$\lambda_{EOS} = 20 \text{ FIT}$$

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

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

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

To forecast the impact of electrical overstress on the device is not trivial. If no particular impact can be argued, then $\lambda_{\text{overstress}}$ can be added to λ_{die} to increase the overall microcontroller die failure rate of the whole device.

NOTE For some analysis standards, electrical over-stress can be considered a systematic failure mode and reduced to zero FIT for calculation of random failure metrics.

A.3.5 Example of quantitative analysis

The following is an example of a quantitative analysis using the method described in paragraph A.3.3.

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 microcontroller into the sub-parts level. As discussed in paragraph A.3.3, 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 paragraph A.3.3.2, transient faults can be also addressed by qualitative rationale.

The example considers a small portion of a microcontroller, i.e. only two parts:

- a) 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.
- b) 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 1 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 microcontroller as shown for example in SN 29500.

NOTE 2 The aim of the following example is to avoid a requirement that each smallest microcontroller 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 microcontroller 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 microcontroller stand-alone level analysis, it can be very difficult to have good data for system-level analysis.

The following four safety mechanisms are considered:

- 1) An HW 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 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 microcontroller). A requirement is set at system 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.

- 2) 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).
- 3) A Single-Error Correction and Double-Error Detection EDC for the RAM (SM3).

NOTE In this example, it is assumed that each detected permanent single bit fault – even if corrected by the EDC – is signalled to the SW (e.g. by an interrupt), and the SW reacts accordingly. A requirement is set at system 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, 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.

- 4) A software-based safety mechanism addressing random hardware failures (SM4) executed at key-on to verify the absence of latent faults in the EDC (SM3).

Table A.5 is divided in three separated calculations for better visibility.

Table A.5 gives the view of failure modes at sub-parts level. Table A.6 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 paragraph A.3.9.

EXAMPLE Table A.6 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 (0,07 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 1 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 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 2 Columns of Table A.5 and Table A.6 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 $K_{\text{FMC,RF}}$;
- 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 $K_{\text{FMC,MPF}}$; and
- latent multiple-point fault failure rate is equal to λ_{MPF} .

Table A.5 — Example of quantitative analysis (at sub-parts level)

Part	Sub-part	Safety Related Component?		Failure modes		Permanent failures		Transient failures				
		No Safety Related Component?				Failure rate (FT)	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 / FT	Safety mechanism(s) preventing latent faults	Failure mode coverage wrt. Latent failures	Latent Multiple Point Fault failure rate / FT
CPU	Register bank	Register R0	SR	permanent fault transient fault		0.0029 0%	SM1 40%	0.00174	SM1 100% 0.00000			
		Register R1	SR	permanent fault transient fault		0.0029 0%	SM1 40%	0.00174	SM1 100% 0.00000			
		Register R2	SR	permanent fault transient fault		0.0029 0%	SM1 20%	0.00232	SM1 100% 0.00000			
		Register R3	SR	permanent fault transient fault		0.0029 0%	SM1 20%	0.00232	SM1 100% 0.00000			
	ALU	ALU	SR	permanent fault transient fault		0.0348 0%	SM1 20%	0.02784	SM1 100% 0.00000			
		MUL	SR	permanent fault transient fault		0.0290 0%	SM1 20%	0.02320	SM1 100% 0.00000			
		DIV	SR	permanent fault transient fault		0.0232 0%	SM1 20%	0.01856	SM1 100% 0.00000			
	Control logic	Pipeline	SR	permanent fault transient fault		0.0174 0%	SM1 90%	0.00174	SM1 100% 0.00000			
		Sequencer	SR	permanent fault transient fault		0.0406 0%	SM1 90%	0.00406	SM1 100% 0.00000			
		Stack control	SR	permanent fault transient fault		0.0029 0%	SM1 70%	0.00087	SM1 100% 0.00000			
	Load Store Unit	Address generation	SR	permanent fault transient fault		0.0174 0%	SM1 60%	0.00696	SM1 100% 0.00000			
		Load Unit	SR	permanent fault transient fault		0.0145 0%	SM1 50%	0.00725	SM1 100% 0.00000			
		Store Unit	SR	permanent fault transient fault		0.0145 0%	SM1 50%	0.00725	SM1 100% 0.00000			
	Debug	Debug Inner Logic	SR	permanent fault transient fault		0.0058 20%	none 0%	0.00464	none			
		Debug Interface	NSR	permanent fault transient fault		0.0783						
		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%		Latent Faults Metric 100.0%				Single Point Faults Metric 27.91%		
Volatile Memory	RAM (16KB)	RAM data bits	SR	permanent fault transient fault		1.5000 0%	SM3 96.9%	0.04688	SM3 100% 0.00000			
		Address Decoder	SR	permanent fault transient fault		0.0087 0%	none 0%	0.00870				
		Test/redundancy	SR	permanent fault transient fault		0.0058 50%	none 0%	0.00290				
		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%		Latent Faults Metric 100.0%				Single Point Faults Metric 99.69%		
Safety Mech.	SM1	Detection Logic	SR	permanent fault transient fault		0.0029 0%			SM2 90% 0.00029			
		Alarm Generation	SR	permanent fault transient fault		0.0029 50%			SM2 90% 0.00015			
		EDC Coder	SR	permanent fault transient fault		0.0029 0%	SM3 90%	0.00029	SM4 90% 0.00026			
		EDC Decoder	SR	permanent fault transient fault		0.0029 0%	SM3 90%	0.00029	SM4 90% 0.00026			
		Alarm Generation	SR	permanent fault transient fault		0.0029 50%			SM4 90% 0.00015			
	SM3	RAM EDC bits	SR	permanent fault transient fault		0.328125 0%	SM3 96.9%	0.01025	SM4 90% 0.03179			
		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%		Latent Faults Metric 90.4%				Single Point Faults Metric 99.69%		

NOTE 1 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 an independent failure of another sub-part.

NOTE 2 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 3 The failure mode coverage of the EDC (SM3) is computed, for example, with a detailed analysis combining the high probability of EDC 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 A.6.

NOTE 4 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 5 The example shows that without proper coverage of the EDC 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 6 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 (SM3) can corrupt the mission data without a corresponding fault in the memory.

NOTE 7 The example shows that, in a microcontroller, 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.

Table A.6 — Example of quantitative analysis (at low-level failures level)

Part	Sub-part	Elementary sub-parts	Safety Related Component? No Safety Related Component?	Permanent failures										Transient failures					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	
				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																																								
				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																																								
				etc....																		
					$\Sigma 0,0348$		$0,02784$		$0,00000$		$0,00038$		$0,00027$																															
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																																								
				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																																								
					$\Sigma 1,5000$		$0,04688$		$0,00000$		$131,0720$		$0,40894$																															

NOTE 1 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 2 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 3 The EDC (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.

A.3.6 Example of dependent failures analysis

The general requirements and recommendations related to identification, evaluation and resolution of dependent failures are respectively defined in ISO 26262-9.

The dependent failures analysis is structured into the following steps:

- 1) Identify parts which could be subject to dependent failures.

NOTE 1 Structures of parts which are claimed to be independent to each other in the safety concept of the microcontroller can be susceptible to dependent failure.

NOTE 2 The identification can be supported by deductive safety analyses: Events assumed to be independent in a dual and multiple-point failures analysis provide useful information about parts vulnerable to dependent failures.

- 2) Identify sources for potential dependent failures.

The topics listed in this section and other foreseeable physical and logical dependent failure sources (shared logical parts and signals) are considered, including effects due to the coexistence of functions with different ASILs.

- 3) Identify the coupling mechanism between the parts enabling dependent failures.

- 4) Qualitatively list and evaluate the measures to prevent the dependent failures.

- 5) Qualitatively list and evaluate the design measures taken to control the effect caused by the remaining dependent failures on each structure of parts identified in step 1.

NOTE As stated in the Note of ISO 26262-9:2011, 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.

As written in Note 1 of ISO 26262-9:2011, 7.4.4, the evaluation of dependent failures can be supported by appropriate checklists, i.e. checklists based on field experience. Those checklists aim to provide the analysts with representative examples of root causes and coupling factors such as: same design, same process, same component, same interface and proximity.

Table A.7 lists the topics considered for dependent failures evaluation as seen in ISO 26262-9:2011, 7.4.4. The table also gives a non-exhaustive example of initiators and coupling mechanisms that could lead to dependent failures with examples for avoidance or detection measures.

NOTE The listed measures are just some of the possible options. Other measures for avoidance or detection of dependent failures are possible, e.g. based on system-level safety mechanisms.

Table A.7 — Topics for dependent failures evaluation, potential initiators and related measures

Topics of ISO 26262-9:2011, 7.4.2.3	Examples for potential initiators and coupling mechanisms	Examples for measures
Hardware failures	Physical defects able to influence both a part and its safety mechanism in such a way that a violation of the safety goal can occur	Can be addressed by measures like physical separation, diversity, production tests, etc.
Development faults	Faults introduced within development which have the capability to cause a dependent failure, for example, crosstalk, incorrect implementation of functionality, specification errors, wrong microcontroller configuration, etc. (see also A.3.7)	Can be addressed by measures like development process definition, diversity, design rules, configuration protection mechanisms, etc.
Manufacturing faults	Faults introduced within manufacturing which have the capability to cause a dependent failure, for example, masks misalignment faults	Can be addressed by a thorough production test of the microcontroller
Installation faults	Faults introduced during installation which have the capability to cause a dependent failure, for example, microcontroller PCB connection, interference of adjacent parts, etc.	Can be addressed by production test of ECU, installation manuals, etc.
Repair faults	Faults introduced during repair which have the capability to cause a dependent failure, for example, faults in memory spare columns/rows	Can be addressed by production tests, repair manuals, etc.
Environmental factors	Typical environmental factors are temperature, EMI, humidity, mechanical stress, etc.	Can be addressed by measures like qualification tests, stress tests, dedicated sensors, diversity, etc.
Failures of common internal and external resources	For a microcontroller, typical shared resources are clocks, reset and power supply, including power distribution	Can be addressed by measures like clock supervision, internal or external supply supervision, diverse distribution, etc.
Stress due to specific situations, e.g. wear, ageing.	Ageing and wear mechanisms are, for example, electro migration, etc.	Can be addressed by design rules, qualification tests, diversity, start-up tests, etc.

Logical failures of shared resources with the potential capabilities of influencing the behaviour of several parts or safety mechanisms within an MCU are not included in this section. They are considered as part of the standard qualitative and quantitative analysis.

EXAMPLE Typical examples falling into this category are DMA controller, interrupt controllers and test/debug logic.

A.3.7 Example of techniques or measures to detect or avoid systematic failures during design of a microcontroller

The general requirements and recommendations related to HW architecture and detailed design are respectively defined in ISO 26262-5:2011, 7.4.1 and ISO 26262-5:2011, 7.4.2. Moreover, requirements related to HW verification are given in ISO 26262-5:2011, 7.4.4.

A microcontroller is developed based on a standardized 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 microcontroller:

- a) using a checklist such as the one reported in Table A.8; and
- b) giving the rationale by field data of similar products which are developed based on the same process as the target device.

Table A.8 — Example of techniques or measures to achieve compliance with ISO 26262-5 requirements during the development of a microcontroller

ISO 26262-5 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 in hardware description language, for example, VHDL or Verilog.
7.4.4 Verification of HW design		HDL simulation	Functional verification of circuit described in VHDL or Verilog by means of simulation.
7.4.4 Verification of HW design		Functional test on module level (using for example HDL test benches)	Functional verification "bottom-up".
7.4.4 Verification of HW design		Functional test on top level	Verification of the microcontroller (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 HW 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 HW design		Application of code checker	Automatic verification of coding rules ("Coding style") by code checker tool.
7.4.4 Verification of HW design		Documentation of simulation results	Documentation of all data needed for a successful simulation in order to verify the specified circuit function.
7.4.4 Verification of HW design	Synthesis	Simulation of the gate netlist, to check timing constraints, or static analysis of the propagation delay (STA - Static Timing Analysis)	Independent verification of the achieved timing constraint during synthesis.
7.4.4 Verification of HW 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.
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 HW design		Simulation of the gate netlist after test insertion, to check timing constraints, or static analysis of the propagation delay (STA)	Independent verification of the achieved timing constraint during test insertion.
7.4.4 Verification of HW 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 HW design	Placement, routing, layout generation	Simulation of the gate netlist after layout, to check timing constraints, or static analysis of the propagation delay (STA)	Independent verification of the achieved timing constraint during back-end.
7.4.4 Verification of HW 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 HW 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 HW design		Design rule check (DRC)	Verification of process design rules.
7.4.4 Verification of HW design		Layout versus schematic check (LVS)	Independent verification of the layout.
7.4.5 Production, operation, service and decommissioning 9.4.2.4 Dedicated measures	Safety-related special characteristics 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 microcontroller.
7.4.5 Production, operation, service and decommissioning 9.4.2.4 Dedicated measures		Determination of measures to detect and weed out early failures	Assurance of the robustness of the manufactured chip. In most, but not each process, gate oxide integrity (GOI) is the key early childhood failure mechanism. GOI childhood failures have many valid methods for screening: high temp/high voltage operation (Burn-In), high current operation, voltage stress, etc. However, the same methods could have no benefit if GOI is not the primary contributor to childhood failures in a process.
7.4.5 Production, operation, service and decommissioning 10 Hardware integration and testing	Qualification of HW component	Definition and execution of qualification tests like Brown-out test, High Temperature Operating Lifetime (HTOL) test and functional testcases	For a microcontroller with integrated brown-out detection, the microcontroller functionality is tested to verify that the outputs of the microcontroller 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 microcontroller without integrated brown-out detection, the microcontroller functionality is tested to verify if the microcontroller sets its outputs to a defined state (for example by stopping the operation of the microcontroller 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.

Moreover, the following general guidelines can be considered:

- c) the documentation of each design activity, test arrangements and tools used for the functional simulation and the results of the simulation;
- d) the verification of each activity and its results, for example by simulation, equivalence checks, timing analysis or checking the technology constraints;
- e) 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.

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

A.3.8 Microcontroller HW design verification

A.3.8.1 General

As seen in ISO 26262-5:2011, 7.4.4.1, the hardware design is verified as given in ISO 26262-8:2011, Clause 9 (Evaluation of safety goal violations due to random hardware failures), for compliance and completeness with respect to the hardware safety requirements.

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

EXAMPLE 1 Either usage of expert judgement or previous proven results when state of the art solutions exist as higher level protocols on communication elements (e.g. SW layer on CAN communications as defined in IEC 61784).

The choice and depth of verification can depend on the stage of the analysis and on the safety mechanisms used (inside the microcontroller or at the system level).

EXAMPLE 2 Following the reasoning of Example 1 of paragraph A.3.3, in the case of a full HW redundancy (e.g. usage of dual core lock step solutions in which each of the outputs of two identical CPUs are compared by HW at each clock cycle), the verification of the failure mode coverage does not need to consider each and every CPU internal register. Instead, a more detailed verification can be needed for the CPU interfaces and for the lock-step comparator.

A.3.8.2 Verification using fault injection simulation

As mentioned in ISO 26262-5:2011, Table 3, 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.

This is especially true for microcontrollers for which fault insertion testing of single-event upset at HW level is impractical or even impossible for certain fault models. Therefore, fault injection using design models (e.g. fault injection done on 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 ISO 26262-5:2011, Table D.1 shows that d.c. fault models (others than stuck-at 0 and 1) need to be considered to be able to claim a high level of diagnostic coverage for certain elements when no other suitable evidence is available. It also describes that it is not intended to require an exhaustive analysis of those fault models and that, if properly exercised, methods derived from stuck-at simulations (like N-detect testing, see references [3]-[5]) are known to be effective for verification of d.c. fault models as well.

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

EXAMPLE 2 Since exhaustiveness is not required, the d.c. fault models analysis can be applied to a sub-set of the microcontroller sub-parts selected depending on their possible impact from d.c. fault models (for example comparators) or on a statistical basis.

EXAMPLE 3 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.

EXAMPLE 4 In general, HW safety mechanisms can be more effective to detect each kind of d.c. 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 applicable (e.g. [7]).

NOTE 3 Fault injection can also be used to inject bridging faults in specific locations based on layout analysis or to verify impact of dependent failures such as injection of clock and reset faults.

Fault injection using design models can be successfully used to assist verification of safe faults and computation of their amount and failure mode coverage, i.e. as seen in paragraphs A.3.3 and A.3.3.2.

EXAMPLE Injecting faults and determining in well-specified observation points 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 fault tolerant time interval by the safety mechanisms.

NOTE 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, the amount of faults injected and the level of detail of the circuit representation.

EXAMPLE Gate-level netlist is appropriate for fault injection of permanent faults such as stuck-at faults. FPGA type 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.

A.3.9 How to adapt and verify microcontroller stand-alone analysis at system level

The adaptation and verification of the microcontroller stand-alone analysis at system level could be done by:

- transforming the detailed failure modes of a microcontroller into the high-level failure modes needed during the analysis at system level;

NOTE 1 This could be done with a bottom-up process (as shown in the following figure): using the method described in paragraphs A.3.2, A.3.3 and A.3.5, it can be possible to identify the detailed microcontroller failure modes and combine them up to the component level.

NOTE 2 Starting from a detailed level of abstraction makes possible a quantitative and precise failure distribution for a microcontroller that otherwise is based on qualitative distribution assumptions.

NOTE 3 As discussed in paragraph A.3.3, the necessary level of detail can depend on the stage of the analysis and on the safety mechanisms used.

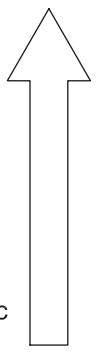
System Level	Wrong actuator output	
Component Level	Wrong output from microcontroller	
Part Level	Wrong data generated by CPU	
Sub-part Level	Wrong data prepared in the ALU	
Elementary sub-part Level	Stuck-at in the fan-in of a flip-flop X of ALU logic	
	

Figure A.2 — Example of bottom-up approach to derived system-level failure modes

- the failure mode coverage computed at part or sub-part level could be improved by measures at the application level; and

EXAMPLE At microcontroller stand-alone level, the failure mode coverage of an ADC peripheral has been considered zero because no safety mechanisms are implemented inside the microcontroller to cover those faults. However, at application level, the ADC is included in a closed-loop, and its faults are detected by a SW-based consistency check. In this case, the failure mode coverage of that sub-part can be increased thanks to the application-level safety mechanism.

- c) the failure mode coverage computed at part or sub-part level could have been calculated under certain specific assumptions (“assumptions of use”).

NOTE In this case the assumptions are verified at application-level, and if not valid, other assumptions could be made and failure mode coverage recalculated based on the new assumptions.

EXAMPLE At microcontroller stand-alone level, a permanent latent fault of the memory has been considered detected because each single-error correction is signalled by the EDC to the CPU. The assumption was that a software driver had been implemented to handle this event. However, for performance reasons, this software driver was not implemented, and therefore the assumption is not valid anymore. An alternative measure is to program the microcontroller to send the error correction flag directly to the outside world. The latent fault coverage of the memory can be recalculated.

A.3.10 Example of safety documentation for an SEooC microcontroller

As seen in paragraph 8.2.3.6, for a microcontroller developed as SEooC, 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 microcontroller can include the following documents or a sub-set of them as specified in the DIA:

- the safety case related to the microcontroller, see ISO 26262-2:2011, 6.5.3;
- the safety plan for the microcontroller, see ISO 26262-2:2011, 6.5.1 and ISO 26262-5:2011, 5.5;
- other plans as seen in ISO 26262-8, when applicable, such as configuration management plan, change management plan, impact analysis and change request plan, verification plan, documentation management plan and SW tool qualification plan;
- the evidence related to the execution of the applicable steps of a safety plan as seen in ISO 26262-2;
- the HW specifications as seen in ISO 26262-5, such as HW safety requirements specification, hardware-software Interface (HSI) specification and HW 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 and ISO 26262-8, such as HW safety requirements verification report, hardware design verification report, and hardware integration and verification report;
- the reports related to safety analyses as seen in ISO 26262-5, ISO 26262-8 and ISO 26262-9, such as hardware safety analysis report, review report of the effectiveness of the architecture of the microcontroller 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 The DIA specifies which documents are made available and what level of detail is provided to the microcontroller's customer.

Moreover, it is worthwhile to collect the following information:

- the description of ISO 26262 lifecycle tailored for the microcontroller; list of applicable work products (description of which work products of the ISO 26262 lifecycle are applicable for the microcontroller);
- the description of the microcontroller safety architecture with an abstract description of microcontroller functionalities and description of safety mechanisms;
- the description of Assumptions of Use (AoU) of the microcontroller with respect to its intended use, including: assumption on the microcontroller safe state; assumptions on fault tolerant time interval and multiple-point faults detection interval; assumptions on the microcontroller context, including its external interfaces;

- the description of the microcontroller configuration and related HW and/or SW procedures to control a failure after its detection;
- the description of safety analysis results at microcontroller level useful for the system integrator, such as description of fault models, failure modes and failure rates considered in the analysis; HW architectural metrics (single-point fault and latent-fault metrics); Probabilistic Metric for random Hardware Failures (PMHF); evaluation of each cause of safety goal violation (ISO 26262-5:2011, 9.4.3); description of dependent failures initiators; description of assumed or implemented measures for avoidance or detection of dependent failures; description of AoUs on which safety analysis are based (e.g. software-based safety mechanisms addressing random hardware failures, etc.);
- 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 microcontroller.

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

Annex B (informative)

Fault tree construction and applications

B.1 General

The two most common techniques for analyzing faults and failures of item and elements are FTA and FMEA. The FMEA is an inductive (bottom-up, see Figure B.1) approach focusing on the individual parts of the system, how they can fail and the impact of these failures on the system. The FTA is a deductive (top down, see Figure B.2) approach starting with the undesired system behaviour and determining the possible causes of this behaviour.

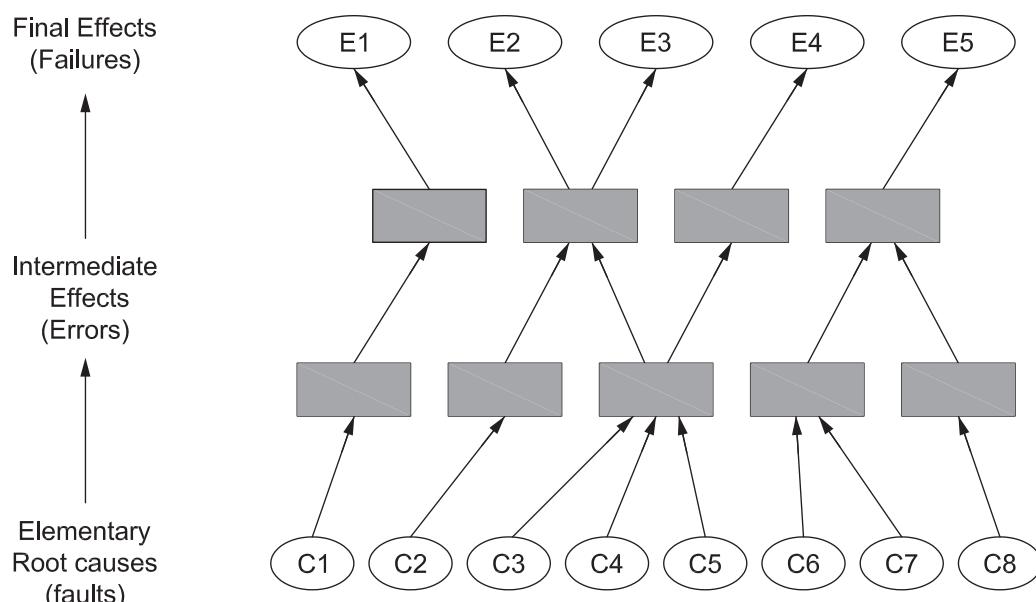


Figure B.1 — Illustration of FMEA, bottom-up approach

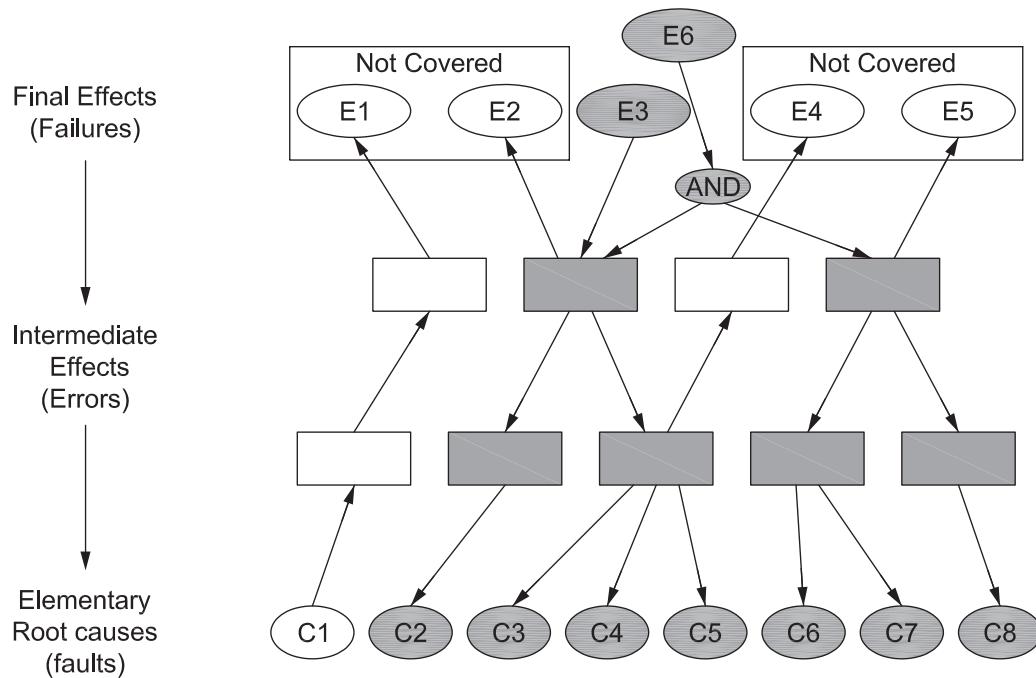


Figure B.2 — Illustration of FTA, top-down approach

The approaches are complementary as stated in ISO 26262-5:2011, 7.4.3.1 Table 2 Note: “The level of detail of the analysis is commensurate with the level of detail of the design. Both methods can, in certain cases, be carried out at different levels of detail.” The “Cx” ovals of Figures B.1 and B.2 represent either hardware or software components. A typical approach is to use the FTA to analyse the hazards down to the component level. The failure modes of the components are then analysed from the bottom up using an FMEA to determine their failure modes and safety mechanisms to close out the bottom level of the fault tree. It is desirable to avoid duplicate work which would be caused by overlap between FTA modelling and FMEA. Preferably the results of the FMEA of serial system parts are fed as failure rates of the base events into the fault tree model.

NOTE As stated in ISO 26262-9:2011, 7.4.2.1, the contribution of dependent failures is estimated on a qualitative basis because no general and sufficiently reliable method exists for quantifying such failures. So the quantification method shown in this chapter is related only to quantifiable dependent failures, such as in the Figure B.9 the common-mode contribution of a permanent fault of SM1 to both the cut-trees of R0 transient and permanent faults.

B.2 Combining FTA and FMEA

Systems are composed of many parts and sub-parts. FTA and FMEA can be combined to provide the safety analysis with the right balance of top-down and bottom-up approach. Figure B.3 shows a possible approach to combine an FTA with an FMEA. In this figure, the basic events are derived from different FMEA (labelled FMEA A-E within this example) which are done on a lower level of abstraction (e.g. sub-part, part or component level). Within this example, FMEA B does not impact basic events 1 and 2, while FMEA D impacts both.

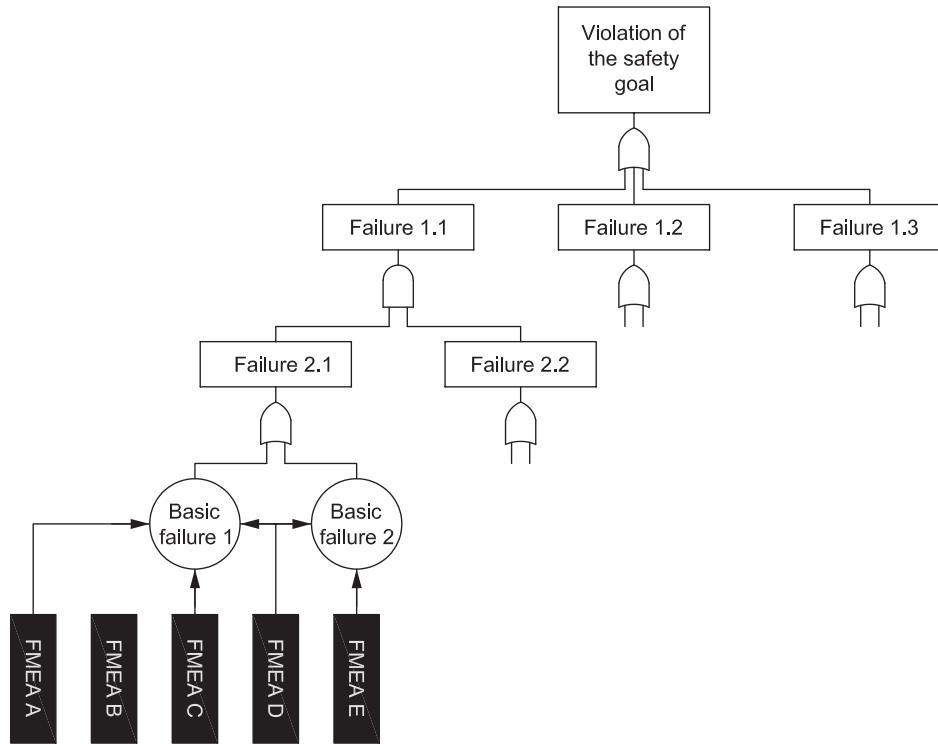


Figure B.3 — Illustration of a combination of FTA and FMEA

B.3 Example Fault Tree

B.3.1 General

A fault tree can be constructed for the microcontroller example of Annex A. This example is not an example of integration of FMEA and FTA, but an example on how to construct a fault tree.

The fault tree is constructed by taking each line of Table A.5 and converting it into a branch of the tree. The complete fault tree is contained in this Annex, Figures B.6 – B.21. The fault tree example is used to illustrate the two methods to evaluate whether the residual risk of safety goal violations is sufficiently low: each branch of the fault tree is evaluated based on its probability of violating the safety goal. Therefore, the top level probability of violating the safety goal does not need to be calculated.

Fault trees are not used to determine diagnostic coverage, or the single-point and latent-fault metrics. Once the diagnostic coverage is determined from, for example, an FMEA, it can be entered into the fault tree so that the probability of failure over the system lifetime can be calculated.

Figure B.4 shows generic examples of FTA in relationship with single-point, residual and dual-point faults.

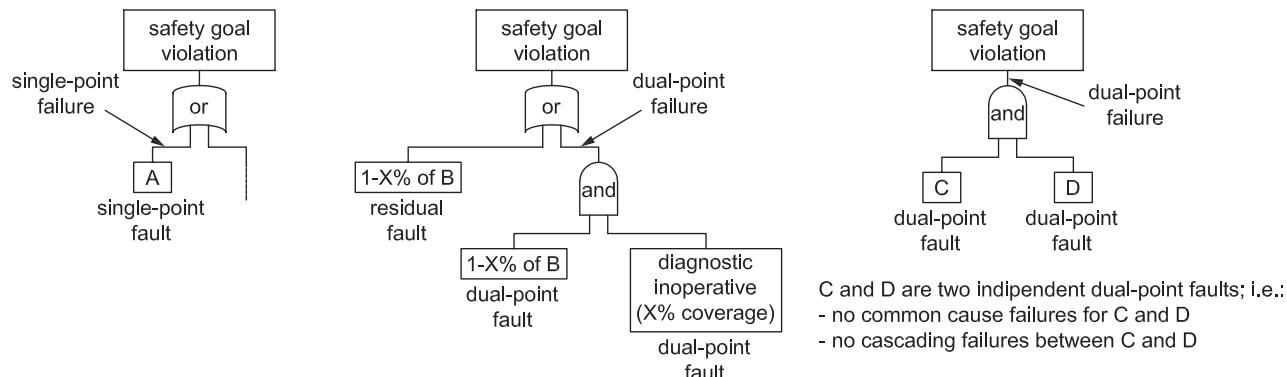


Figure B.4 — Illustration of FTA for single-point, residual and dual-point faults

B.3.2 Example of constructing a fault tree branch

As an example of the construction of one branch, the fault tree branch for Register R0 is described in detail. Figure B.9 shows that the first two rows of Table A.5 are connected by OR gate together. This assumes that the permanent and transient failures of Register R0 are independent failure modes. Since the transient failure rates and diagnostic coverage are known, transient faults are included in the fault tree in the same manner as permanent faults. If failure rates and diagnostic coverage are not known, transient faults can be handled separately as described in ISO 26262-5:2011, 8.4.7 Note 2.

NOTE The following example shows a method to combine both permanent and transient faults. Per ISO 26262-5:2011, 8.4.7 NOTE 2, when transient faults are shown to be relevant, they are included in the analysis. Since for this example, they are both relevant and quantifiable, they are included in the fault tree in a similar manner as the permanent faults.

Constructing the transient fault case first, this branch consists of a transient fault with a failure rate of 0,032 005 FIT ($3,200 \times 10^{-11}/\text{h}$ of operation) which is connected to an AND gate; the other input of the AND gate is connected to an OR gate. The OR gate has two components, a fixed probability of 60 % representing the failure mode coverage (1 – the failure mode coverage with respect to violation of safety goal) and the probability of a latent fault failure of the safety mechanism SM1. The “r=” indication under an event block represents a failure rate per hour, and the “Q=” indicates the probability of failure for that block or branch over the expected life time of the system.

NOTE The fixed probability event dominates the diagnostic coverage plus latent fault branch. For practical systems, the diagnostic coverage plus latent fault branch (the output of GATE 19 in Figure B.9) can be ignored, simplifying the tree. However, the latent-fault metric still is evaluated and satisfied.

The latent fault would cause the system to not detect a transient failure that is included in the diagnostic coverage and would normally be detected. Note that the latent/primary fault combination is order dependant. If the latent fault of the safety mechanism occurs prior to the primary fault, the primary faults cannot be detected, and the mitigation of the hazard cannot occur. This is represented by a small boxed “L” indicating that it must occur last in sequence with the other member of the AND block.

The SM1 block is constructed from the Safety Mechanism portion of Table A.5, the two rows for permanent failures for Detection Logic and Alarm Generation. Transient failures are not included as these will not cause a latent fault because the probability that they occur simultaneously with the primary fault is very low and therefore negligible. This is consistent with ISO 26262-5:2011, 8.4.7 Note 2, that consider transient faults for single-point fault metric only. The Detection Logic has a failure rate of 0,002 9 FIT.

The failure of the Detection Logic is further diagnosed by safety mechanism SM2 at a DC of 90 %, this block is connected by AND with gate SM1 which calculates the probability associated with the latent fault of SM1. Safety mechanism SM2 detects latent faults in SM1 and is run at every key start. From ISO 26262-5:2011, 9.4.2.3, the mean duration of a vehicle trip can be considered as being equal to one hour. One hour is represented by the tau=1 below the DL LATENT1 event, which is multiplied by 0,9 (90 %), the latent fault

coverage of SM2. The portion of the Alarm Generation faults not covered by SM2 is represented by the standard failure rate event ALARM LATENT multiplied by 0,1 (10 % = 1 – latent fault DC).

The permanent fault branch is constructed in a similar manner. The underlining of the triangular-shaped SM1 block indicates that it is not simply a copy of the existing SM1 block in the transient fault portion of the tree but exactly the same failure mode. This can be useful in common mode failure analysis; for example, in Figure B.9, the AND blocks TRANS R0 and PERM R0 contain a common branch.

The example of Table A.5 contains safe faults. Considering the safe faults, the failure rate for basic events is adjusted. For example, the transient ALU fault (row 10 in Table A.5, 20 % safe faults) 0,000 38 FIT is multiplied by 0,8 (1 – 20 %), $3,8 \times 10^{-13}/h * 0,8 = 3,04 \times 10^{-13}/h$.

B.4 Probability analysis using the fault tree

Typical quantitative data for component failures are documented as failure rates. For a complex fault tree with many ANDs and ORs, the individual failure rates cannot be combined into an overall system failure rate. For example, a system consisting of two blocks which are connected by an AND-Gate and which have an exponential distribution where both failure rates of λ_1 and λ_2 are very small (both $\lambda*t$ values are very small where t represents the system lifetime), will have an approximate probability of failure of $\lambda_1*t*\lambda_2*t$ or $\lambda_1\lambda_2t^2$.

If the system is an ASIL D system and the failure rate targets of ISO 26262-5:2011, Table 7 are used, the target failure probability per hour is $< 10^{-8}/h$ which refers to a different time frame than $\lambda\lambda_2t^2$. One potential way to handle the situation is to use the target failure probability per hour of $10^{-8}/h$ at system lifetime $10^{-8}/h*t$ and ensure that $\lambda_1\lambda_2t^2 \leq 10^{-8}/h*t$ or $\lambda_1\lambda_2t \leq 10^{-8}/h$. This requires knowledge of the system lifetime, which can be obtained from past usage profiles or system requirements. The fault tree of this Annex was created assuming an arbitrary 5000 h system lifetime.

For the R0 TRANSIENT branch, the probability of a missed detection is primarily due to a lack of diagnostic coverage ($Q=0,6$) and not due to a latent fault ($Q=2,175 \times 10^{-9}$). This is typical of most practical systems unless the DC is equal to or very close to 100 %. For the cases where the latent fault probabilities are negligible, removing them from the FTA analysis greatly simplifies the analysis. The latent-fault metric is still documented as a separate part of the safety case.

Within the approach of ISO 26262-5:2011, 9.4.3, the probability of a safety-related fault is evaluated at a hardware part level. If this analysis is carried out by means of an FTA, the FTA is structured in such a way that the evaluation at HW part level is possible.

Within this example, this is done in Figure B.6. In ISO 26262-5:2011, 9.4.3.5 and 9.4.3.6, the requirements are given in form of a correlation between the failure rate class of the hardware part and its diagnostic coverage. If the evaluation is done within the FTA, it is useful to translate the requirements of ISO 26262-5:2011, 9.4.3.5 and 9.4.3.6 into an effective requirement concerning the residual failure rate or the single-point failure rate of the hardware part. If, for example, the failure rate class 1 is chosen to be the ASIL D target divided by 100, and the ASIL D target is chosen to be $< 10^{-8} h^{-1}$, the requirement of ISO 26262-5:2011, 9.4.3.6 can be reformulated to $\lambda_{RF,HW\ part} \leq 10^{-10} h^{-1}$, with $\lambda_{RF,HW\ part}$ being the residual failure rate of the hardware part. In Figure B.6, we see that the probability of a CPU failure leading to a potential violation of a safety goal over the lifetime of 5000 h is $1,040\ 631 \times 10^{-6}$. This results in a corresponding probability of $2,08 \times 10^{-10}$ per hour. Since this value is greater than the required residual failure rate, the requirements of 9.4.3.6 would not be fulfilled.

Using the FTA as described to evaluate the fulfilment of ISO 26262-5:2011, 9.4.3.5 or 9.4.3.6, is conservative since the effective requirements refer to the residual or single-point failure rate of the HW part; the FTA, however, provides the probability also including the dual-point failure scenarios.

Since the FTA does not address the requirements of ISO 26262-5:2011, 9.4.3.11, an additional analysis is necessary to provide evidence that the corresponding requirements concerning the dual-point failure scenarios are fulfilled.

If a rationale is provided, the failure rate class ranking can be divided by a number lower than 100. In this case, it is ensured that a correct ranking is maintained while considering the single-point fault, residual faults and higher degree cut-sets together.

EXAMPLE The rationale can be based on the number of minimal cut-sets.

B.5 Example of Fault Tree

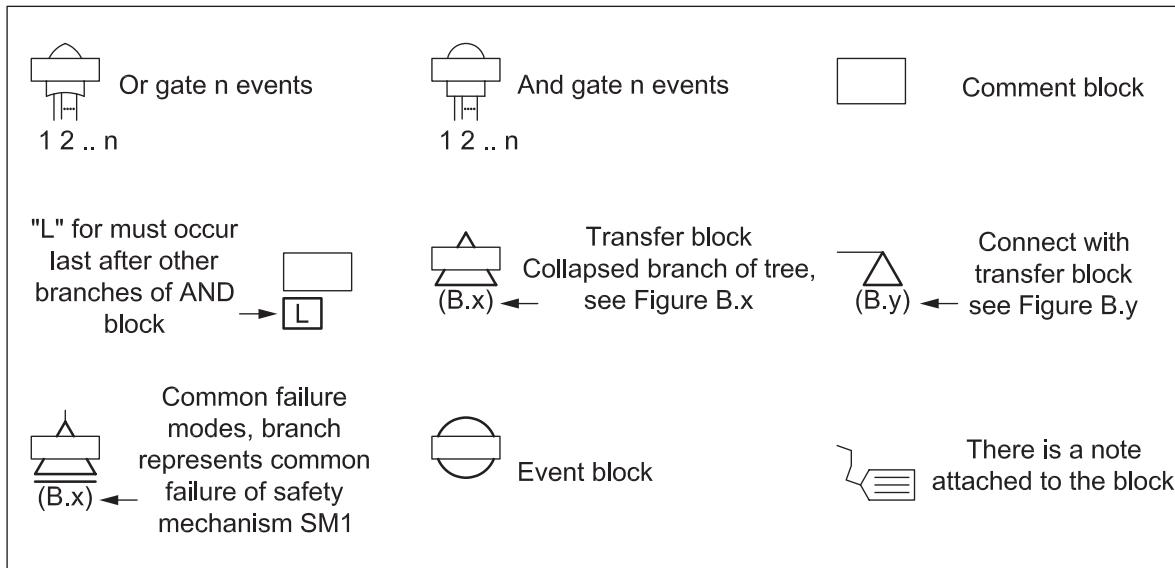


Figure B.5 — Introductory notes of FTA symbols

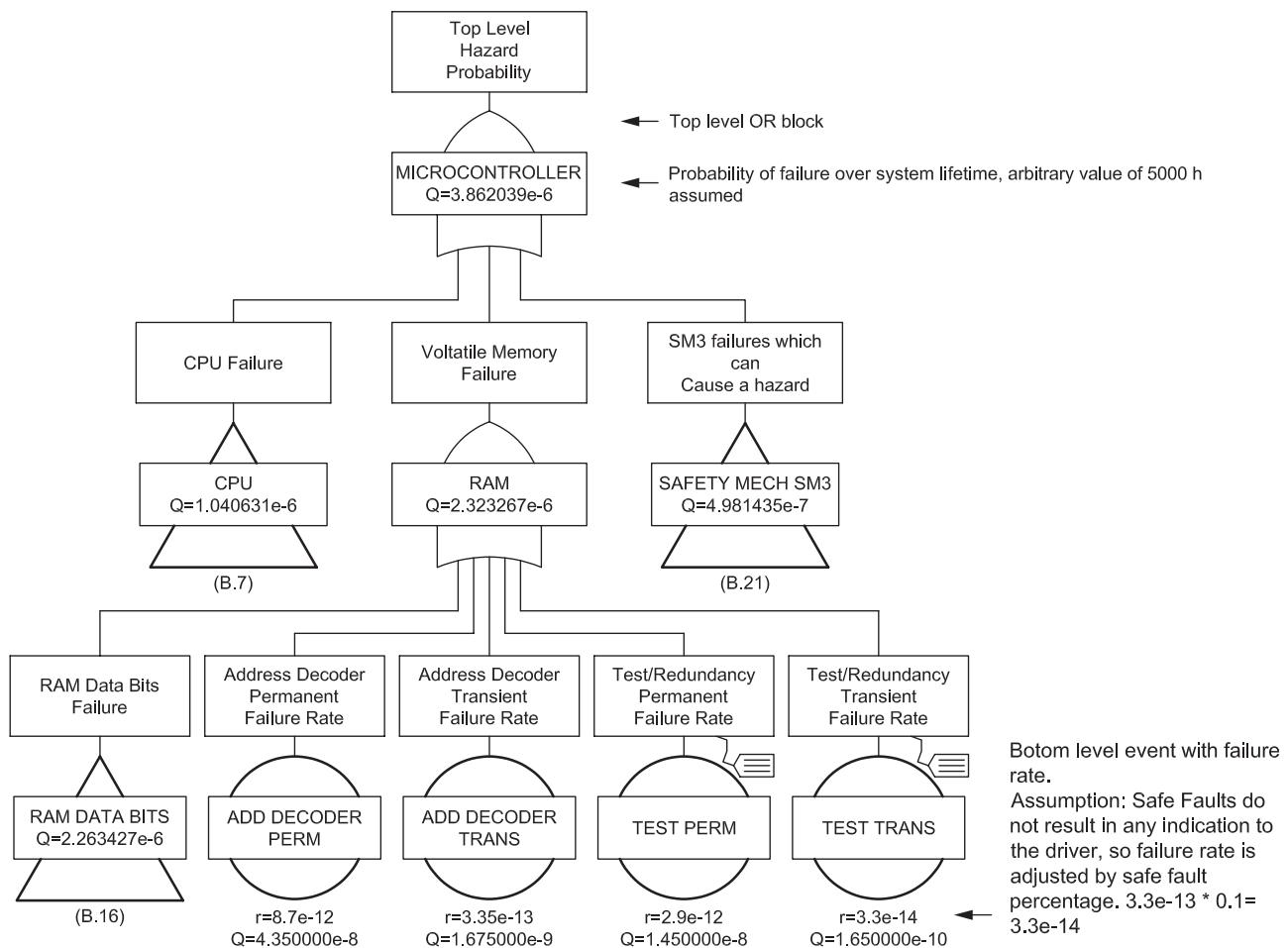


Figure B.6 — Top level of fault tree

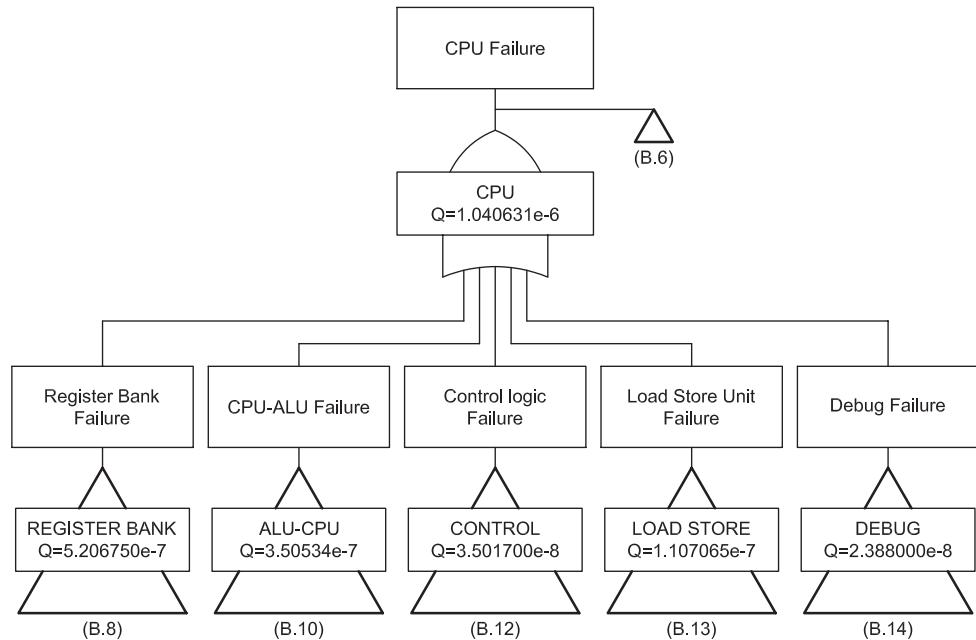
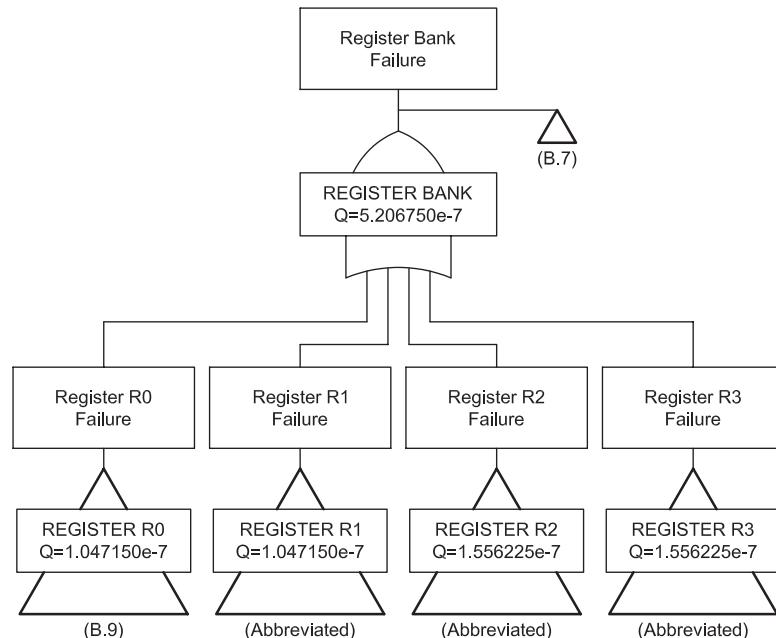
**Figure B.7 — Top level of fault tree for CPU branch****Figure B.8 — Top level of fault tree for register bank for CPU branch**



Figure B.9 — Register R0 fault tree branch

Except for the diagnostic coverage values, Figure B.9 is a representative fault tree for each of the Register fault trees. Detailed fault trees for Register 1, 2 and 3 are not provided.

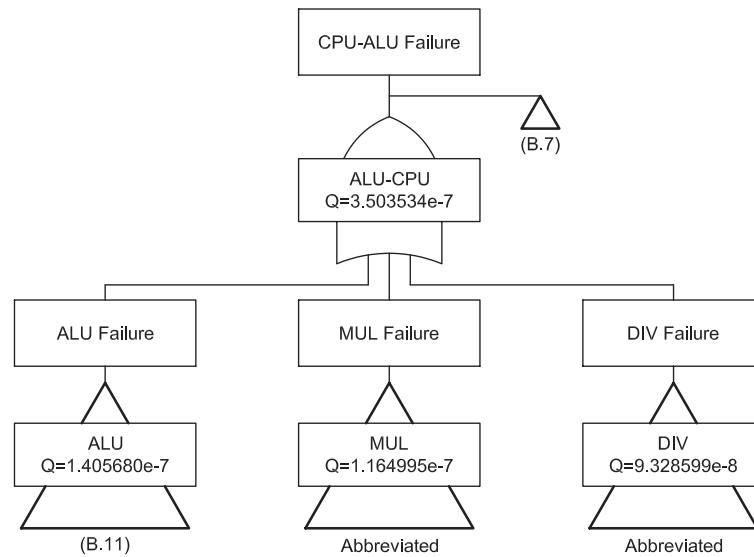


Figure B.10 — Top level of fault tree for ALU-CPU branch

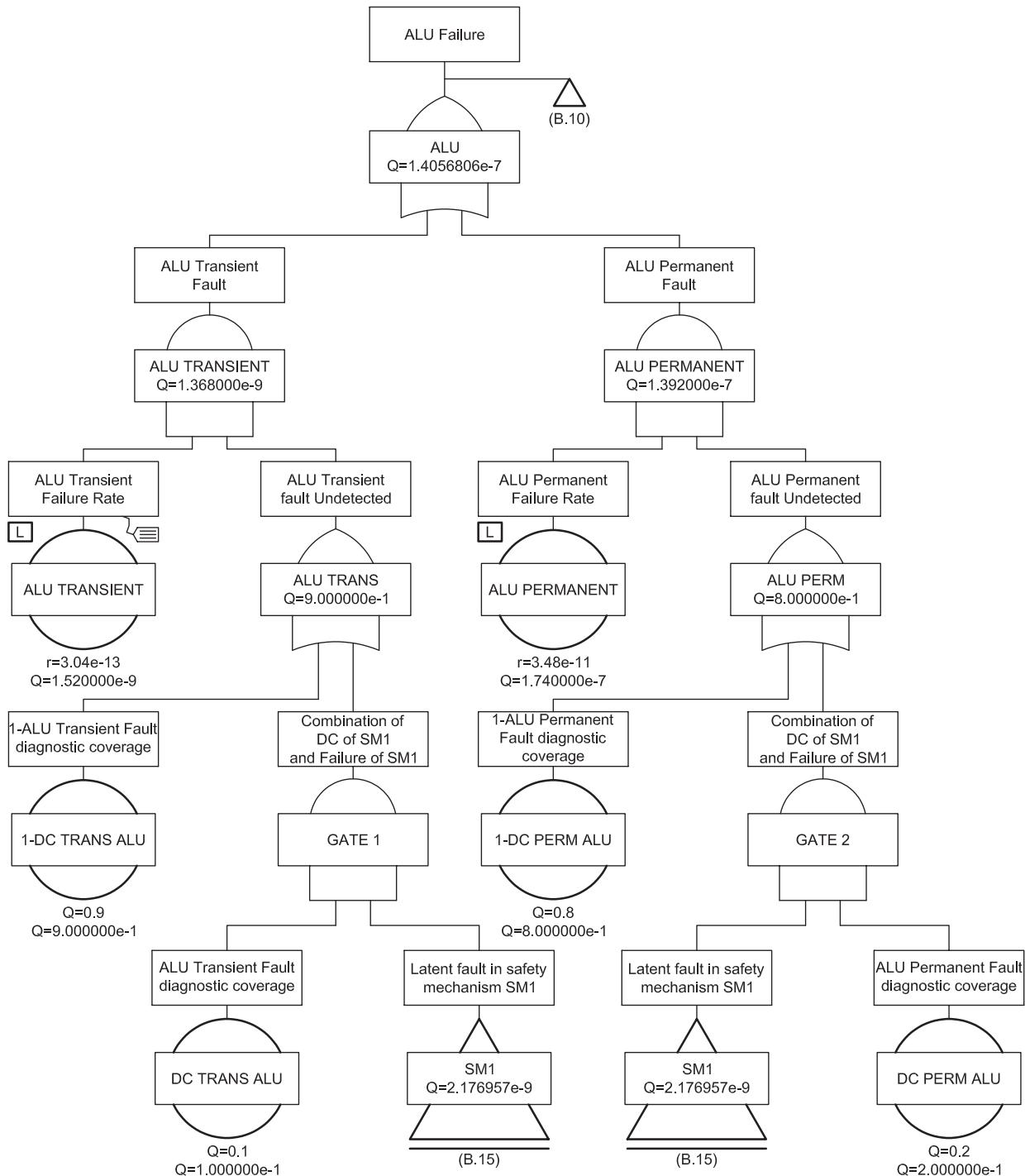
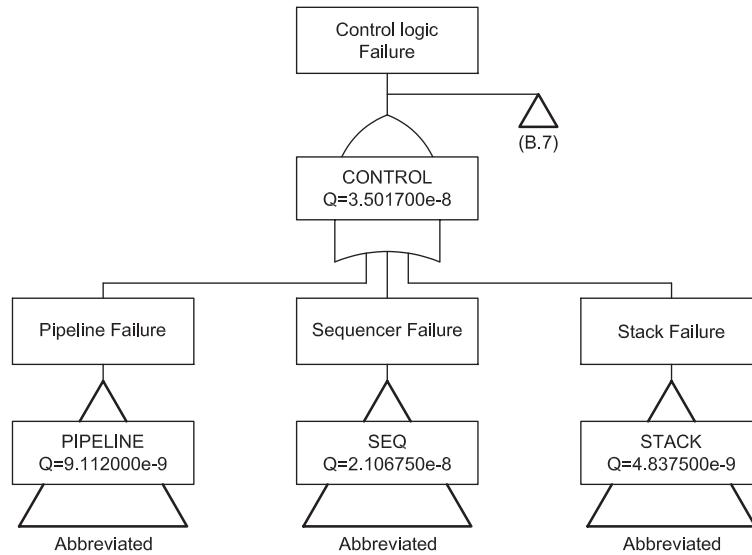
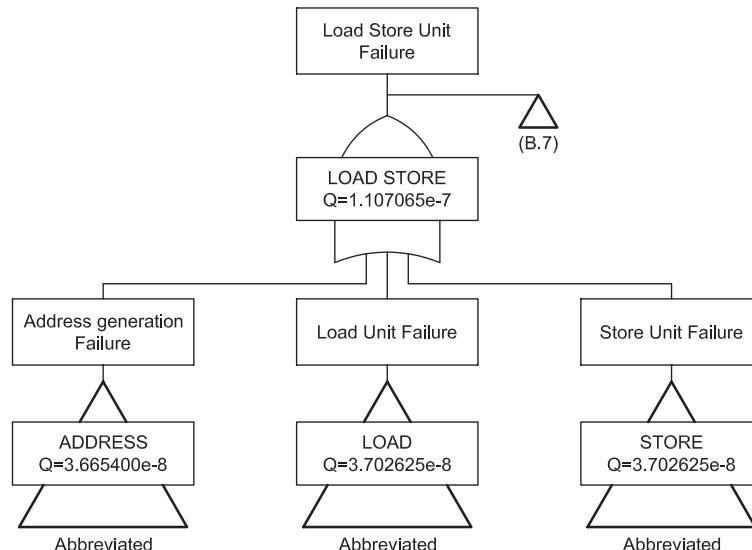


Figure B.11 — ALU fault tree branch

Except for the diagnostic coverage values and FIT rates, Figure B.11 is a representative fault tree for the MUL, DIV, pipeline, sequencer, stack control, address generation, load and store fault trees. Detailed fault trees for these branches are not provided.

**Figure B.12 — Top level of fault tree for control branch****Figure B.13 — Top level of fault tree for load store unit branch**

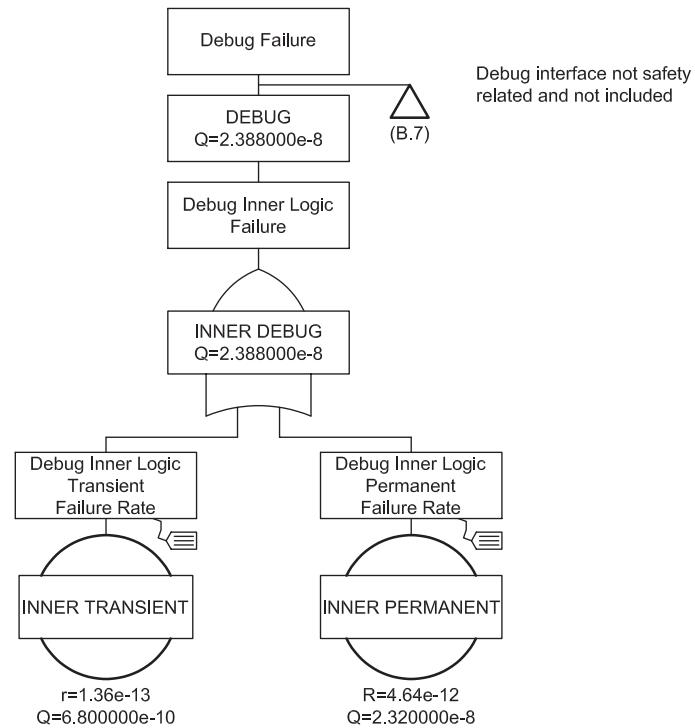


Figure B.14 — Debug fault tree branch

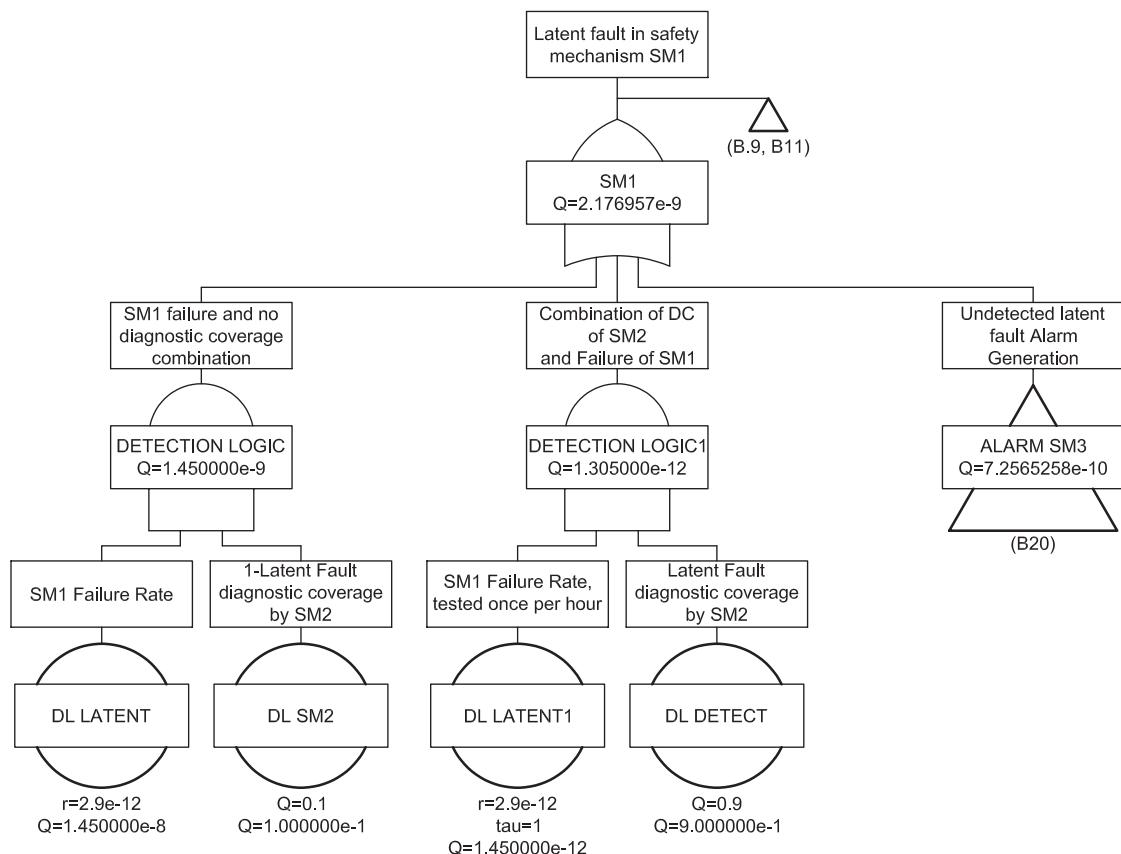


Figure B.15 — Latent fault coverage of safety mechanism SM2

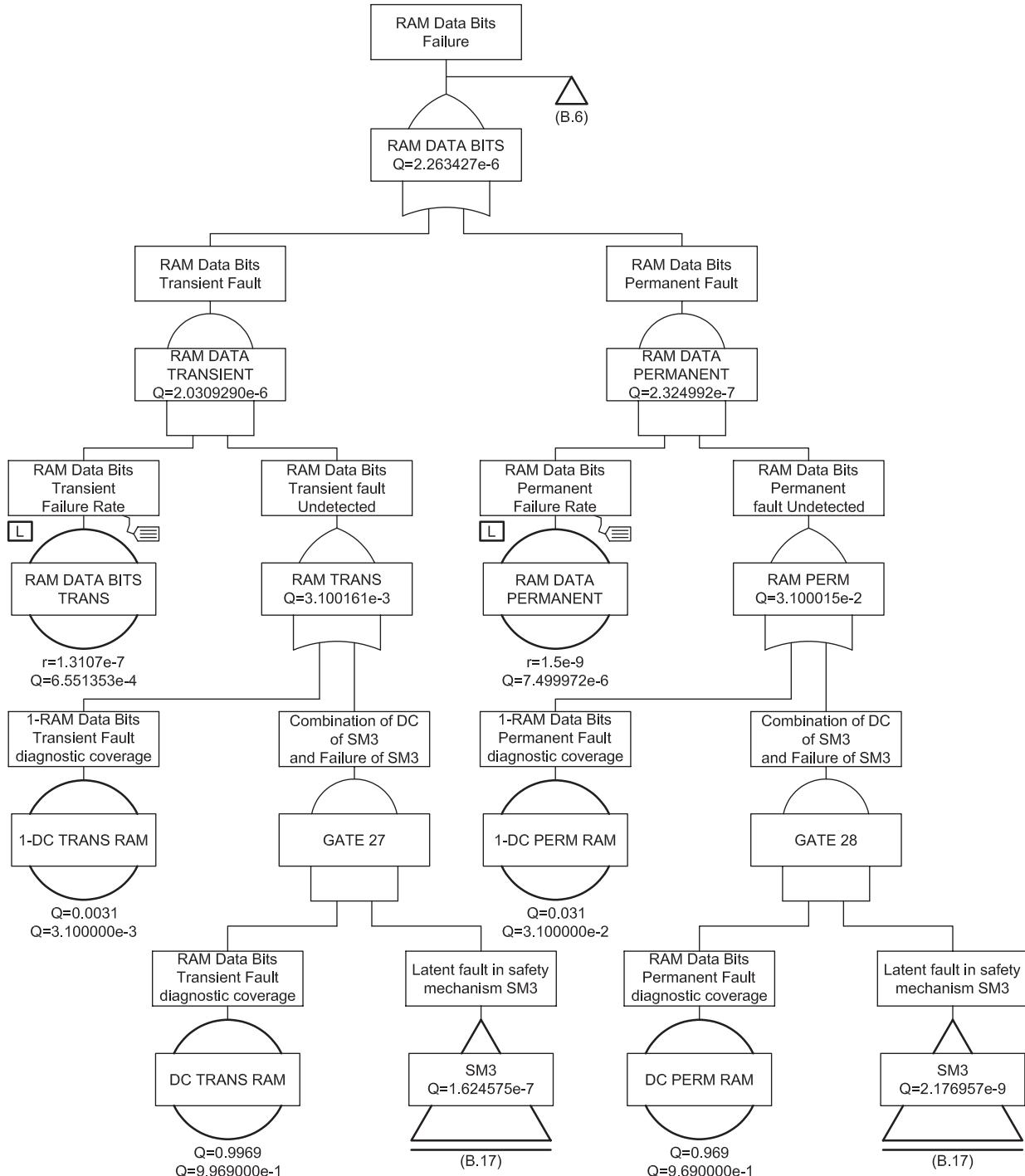


Figure B.16 — Top level of fault tree for volatile memory branch

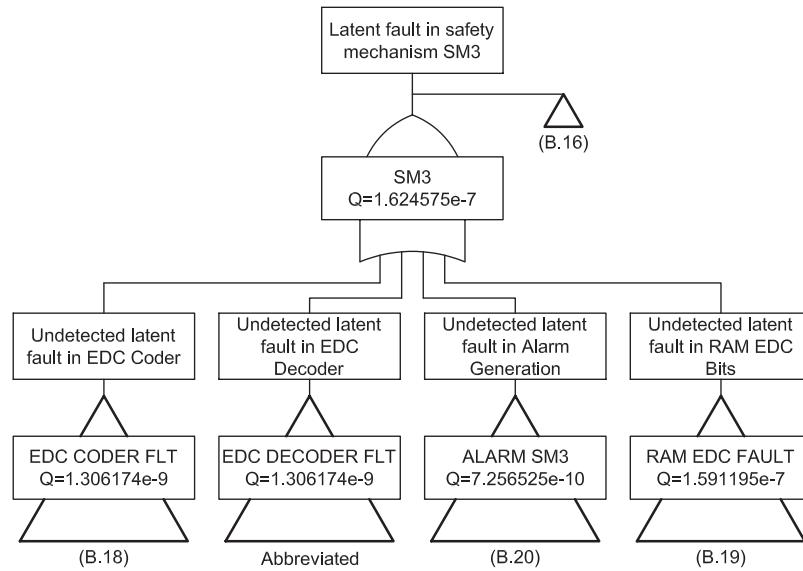


Figure B.17 — Safety mechanism SM3 fault tree branch – Top level

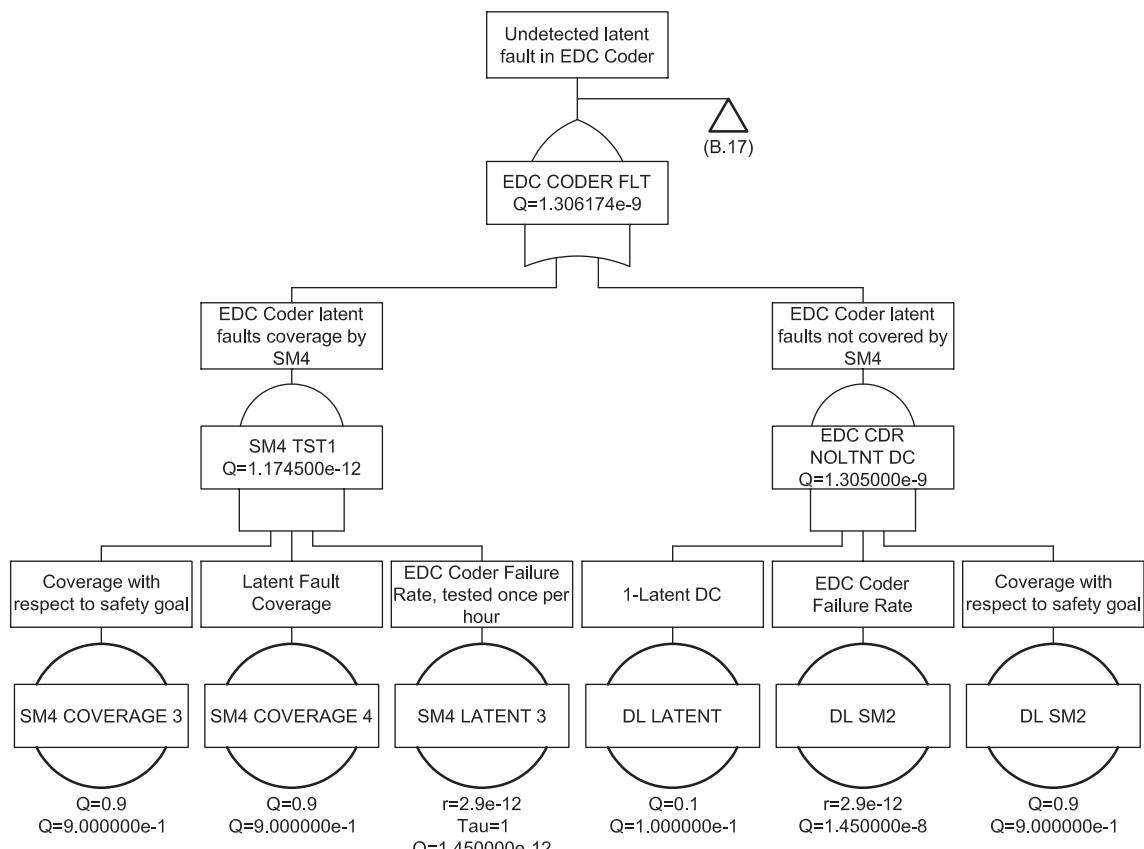


Figure B.18 — EDC coder latent fault tree branch

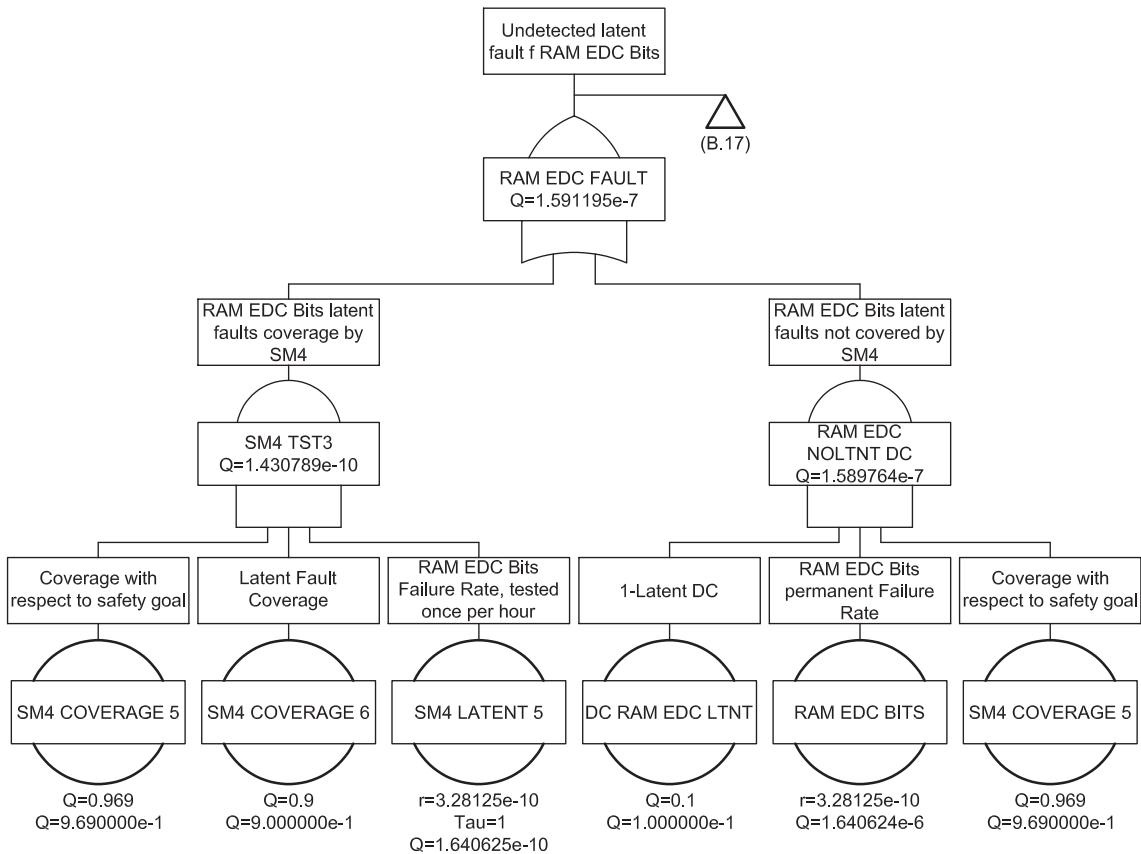


Figure B.19 — RAM EDC bits latent fault tree branch

Except for the diagnostic coverage values, Figure B.19 is a representative fault tree for the EDC decoder fault tree. A detailed fault trees for this branch is not provided.

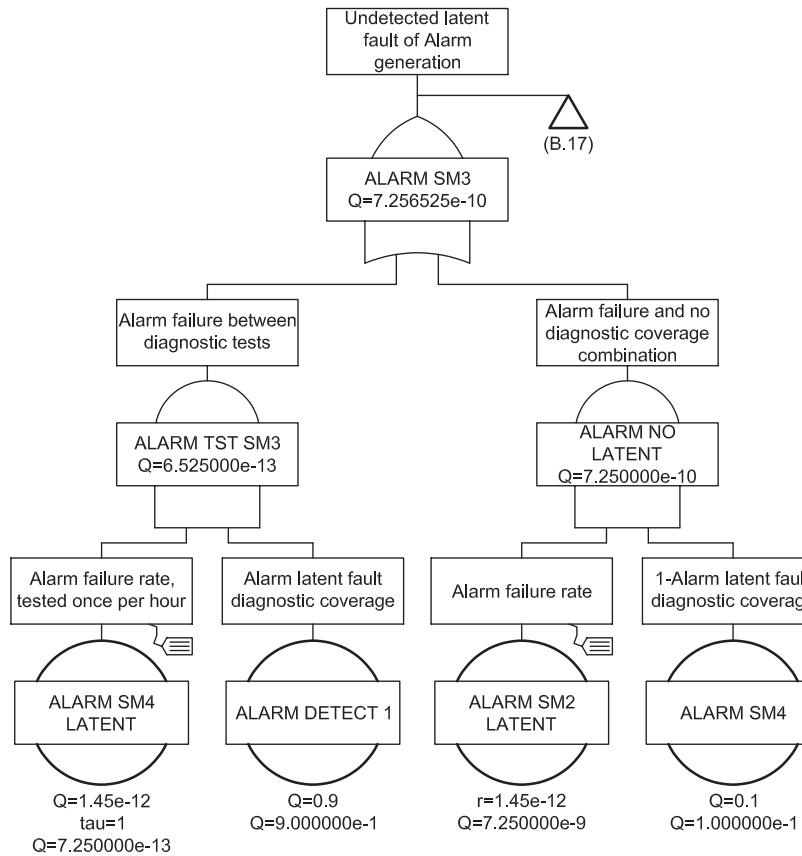


Figure B.20 — Alarm latent generation latent fault branch

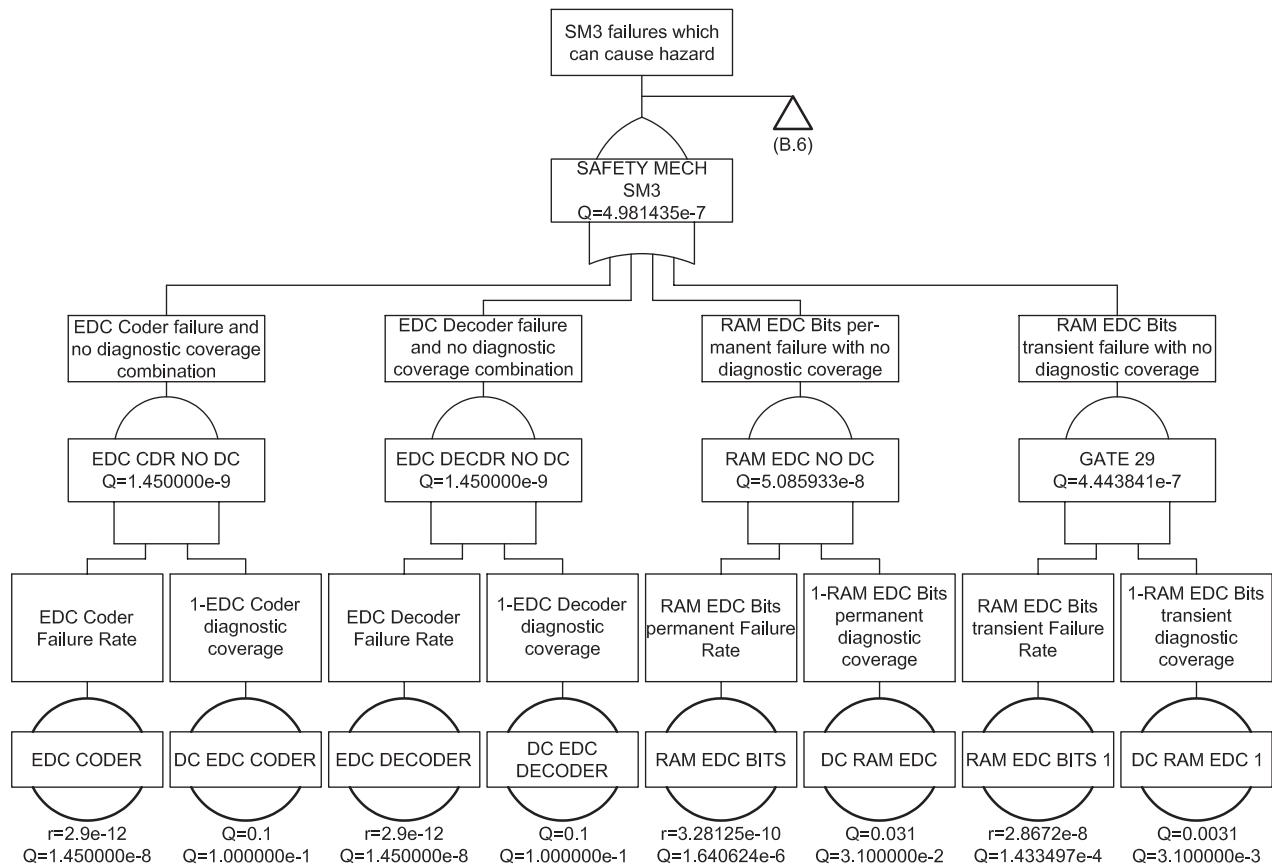


Figure B.21 — SM3 fault tree branch, failures that directly contribute to the top level hazard

Bibliography

- [1] IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*
- [2] Kelly, T. P., Arguing Safety – A Systematic Approach to Safety Case Management, DPhil Thesis, Department of Computer Science, University of York, UK, 1998
- [3] 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
- [4] 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
- [5] Patel, J. H., "Stuck-At Fault: A Fault Model for the Next Millennium?", Proceedings of the International Test Conference 1998, ITC'98, pp.1166
- [6] Siemens AG, "Failure Rates of Components – Expected Values, General", SN 29500 (2004)
- [7] Paschalidis, 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
- [8] IEC/TR 62380:2004, *Reliability data handbook — Universal model for reliability prediction of electronics components, PCBs and equipment*
- [9] The International Technology Roadmap For Semiconductors (ITRS), 2009 Edition
- [10] IEC 61709, *Electric components — Reliability — Reference conditions for failure rates and stress models for conversion*
- [11] IEC 61784 (all parts), *Industrial communication networks — Profiles*

ICS 43.040.10

Price based on 89 pages