IEC 61508-6

Edition 2.0 2010-04

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Functional safety of electrical/electronic/programmable electronic safety-related systems –
Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3

Sécurité fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité –
Partie 6: Lignes directrices pour l'application de la CEI 61508-2 et de la CEI 61508-3

IEC 61508-6:2010

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:
Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00

A propos de la CEI

La Commission Electrotechnique Internationale (CEI) est la première organisation mondiale qui élabore et publie des normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications CEI

Le contenu technique des publications de la CEI est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

- Catalogue des publications de la CEI: www.iec.ch/searchpub/cur_fut-f.htm

Le Catalogue en-ligne de la CEI vous permet d'effectuer des recherches en utilisant différents critères (numéro de référence, texte, comité d'études,…). Il donne aussi des informations sur les projets et les publications retirées ou remplacées.

- Just Published CEI: www.iec.ch/online_news/justpub

Restez informé sur les nouvelles publications de la CEI. Just Published détaille deux fois par mois les nouvelles publications parues. Disponible en-ligne et aussi par email.

- Electropedia: www.electropedia.org

Le premier dictionnaire en ligne au monde de termes électroniques et électriques. Il contient plus de 20 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans les langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International en ligne.

- Service Clients: www.iec.ch/webstore/custserv/custserv_entry-f.htm

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions, visitez le FAQ du Service clients ou contactez-nous:
Email: csc@iec.ch
Tél.: +41 22 919 02 11
Fax: +41 22 919 03 00

IEC 61508-6

Edition 2.0 2010-04

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3

Sécurité fonctionnelle des systèmes électriques/électroniques/électroniques programmables relatifs à la sécurité – Partie 6: Lignes directrices pour l'application de la CEI 61508-2 et de la CEI 61508-3

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

PRICE CODE
CODE PRIX XE

ICS 25.040.40

ISBN 978-2-88910-529-8

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

FUNCTIONAL SAFETY OF ELECTRICAL/ELECTRONIC/ PROGRAMMABLE ELECTRONIC SAFETY-RELATED SYSTEMS –

Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61508-6 has been prepared by subcommittee 65A: System aspects, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2000. This edition constitutes a technical revision.

This edition has been subject to a thorough review and incorporates many comments received at the various revision stages.

The text of this standard is based on the following documents:

FDIS	Report on voting
65A/553/FDIS	65A/577/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts of the IEC 61508 series, published under the general title *Functional safety of electrical / electronic / programmable electronic safety-related systems*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

INTRODUCTION

Systems comprised of electrical and/or electronic elements have been used for many years to perform safety functions in most application sectors. Computer-based systems (generically referred to as programmable electronic systems) are being used in all application sectors to perform non-safety functions and, increasingly, to perform safety functions. If computer system technology is to be effectively and safely exploited, it is essential that those responsible for making decisions have sufficient guidance on the safety aspects on which to make these decisions.

This International Standard sets out a generic approach for all safety lifecycle activities for systems comprised of electrical and/or electronic and/or programmable electronic (E/E/PE) elements that are used to perform safety functions. This unified approach has been adopted in order that a rational and consistent technical policy be developed for all electrically-based safety-related systems. A major objective is to facilitate the development of product and application sector international standards based on the IEC 61508 series.

In most situations, safety is achieved by a number of systems which rely on many technologies (for example mechanical, hydraulic, pneumatic, electrical, electronic, programmable electronic). Any safety strategy must therefore consider not only all the elements within an individual system (for example sensors, controlling devices and actuators) but also all the safety-related systems making up the total combination of safety-related systems. Therefore, while this International Standard is concerned with E/E/PE safety-related systems, it may also provide a framework within which safety-related systems based on other technologies may be considered.

It is recognized that there is a great variety of applications using E/E/PE safety-related systems in a variety of application sectors and covering a wide range of complexity, hazard and risk potentials. In any particular application, the required safety measures will be dependent on many factors specific to the application. This International Standard, by being generic, will enable such measures to be formulated in future product and application sector international standards and in revisions of those that already exist.

This International Standard

– considers all relevant overall, E/E/PE system and software safety lifecycle phases (for example, from initial concept, though design, implementation, operation and maintenance to decommissioning) when E/E/PE systems are used to perform safety functions;

– has been conceived with a rapidly developing technology in mind; the framework is sufficiently robust and comprehensive to cater for future developments;

– enables product and application sector international standards, dealing with E/E/PE safety-related systems, to be developed; the development of product and application sector international standards, within the framework of this standard, should lead to a high level of consistency (for example, of underlying principles, terminology etc.) both within application sectors and across application sectors; this will have both safety and economic benefits;

– provides a method for the development of the safety requirements specification necessary to achieve the required functional safety for E/E/PE safety-related systems;

– adopts a risk-based approach by which the safety integrity requirements can be determined;

– introduces safety integrity levels for specifying the target level of safety integrity for the safety functions to be implemented by the E/E/PE safety-related systems;

NOTE 2 The standard does not specify the safety integrity level requirements for any safety function, nor does it mandate how the safety integrity level is determined. Instead it provides a risk-based conceptual framework and example techniques.

– sets target failure measures for safety functions carried out by E/E/PE safety-related systems, which are linked to the safety integrity levels;

– sets a lower limit on the target failure measures for a safety function carried out by a single E/E/PE safety-related system. For E/E/PE safety-related systems operating in
  – a low demand mode of operation, the lower limit is set at an average probability of a dangerous failure on demand of 10^{-5};
  – a high demand or a continuous mode of operation, the lower limit is set at an average frequency of a dangerous failure of 10^{-9} [h^{-1}];

NOTE 3 A single E/E/PE safety-related system does not necessarily mean a single-channel architecture.

NOTE 4 It may be possible to achieve designs of safety-related systems with lower values for the target safety integrity for non-complex systems, but these limits are considered to represent what can be achieved for relatively complex systems (for example programmable electronic safety-related systems) at the present time.

– sets requirements for the avoidance and control of systematic faults, which are based on experience and judgement from practical experience gained in industry. Even though the probability of occurrence of systematic failures cannot in general be quantified the standard does, however, allow a claim to be made, for a specified safety function, that the target failure measure associated with the safety function can be considered to be achieved if all the requirements in the standard have been met;
– introduces systematic capability which applies to an element with respect to its confidence that the systematic safety integrity meets the requirements of the specified safety integrity level;
– adopts a broad range of principles, techniques and measures to achieve functional safety for E/E/PE safety-related systems, but does not explicitly use the concept of fail safe. However, the concepts of “fail safe” and “inherently safe” principles may be applicable and adoption of such concepts is acceptable providing the requirements of the relevant clauses in the standard are met.

FUNCTIONAL SAFETY OF ELECTRICAL/ELECTRONIC/ PROGRAMMABLE ELECTRONIC SAFETY-RELATED SYSTEMS –

Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3

1 Scope

1.1 This part of IEC 61508 contains information and guidelines on IEC 61508-2 and IEC 61508-3.

- Annex A gives a brief overview of the requirements of IEC 61508-2 and IEC 61508-3 and sets out the functional steps in their application.
- Annex B gives an example technique for calculating the probabilities of hardware failure and should be read in conjunction with 7.4.3 and Annex C of IEC 61508-2 and Annex D.
- Annex C gives a worked example of calculating diagnostic coverage and should be read in conjunction with Annex C of IEC 61508-2.
- Annex D gives a methodology for quantifying the effect of hardware-related common cause failures on the probability of failure.
- Annex E gives worked examples of the application of the software safety integrity tables specified in Annex A of IEC 61508-3 for safety integrity levels 2 and 3.

1.2 IEC 61508-1, IEC 61508-2, IEC 61508-3 and IEC 61508-4 are basic safety publications, although this status does not apply in the context of low complexity E/E/PE safety-related systems (see 3.4.3 of IEC 61508-4). As basic safety publications, they are intended for use by technical committees in the preparation of standards in accordance with the principles contained in IEC Guide 104 and ISO/IEC Guide 51. IEC 61508-1, IEC 61508-2, IEC 61508-3 and IEC 61508-4 are also intended for use as stand-alone publications. The horizontal safety function of this international standard does not apply to medical equipment in compliance with the IEC 60601 series.

1.3 One of the responsibilities of a technical committee is, wherever applicable, to make use of basic safety publications in the preparation of its publications. In this context, the requirements, test methods or test conditions of this basic safety publication will not apply unless specifically referred to or included in the publications prepared by those technical committees.

1.4 Figure 1 shows the overall framework of the IEC 61508 series and indicates the role that IEC 61508-6 plays in the achievement of functional safety for E/E/PE safety-related systems.

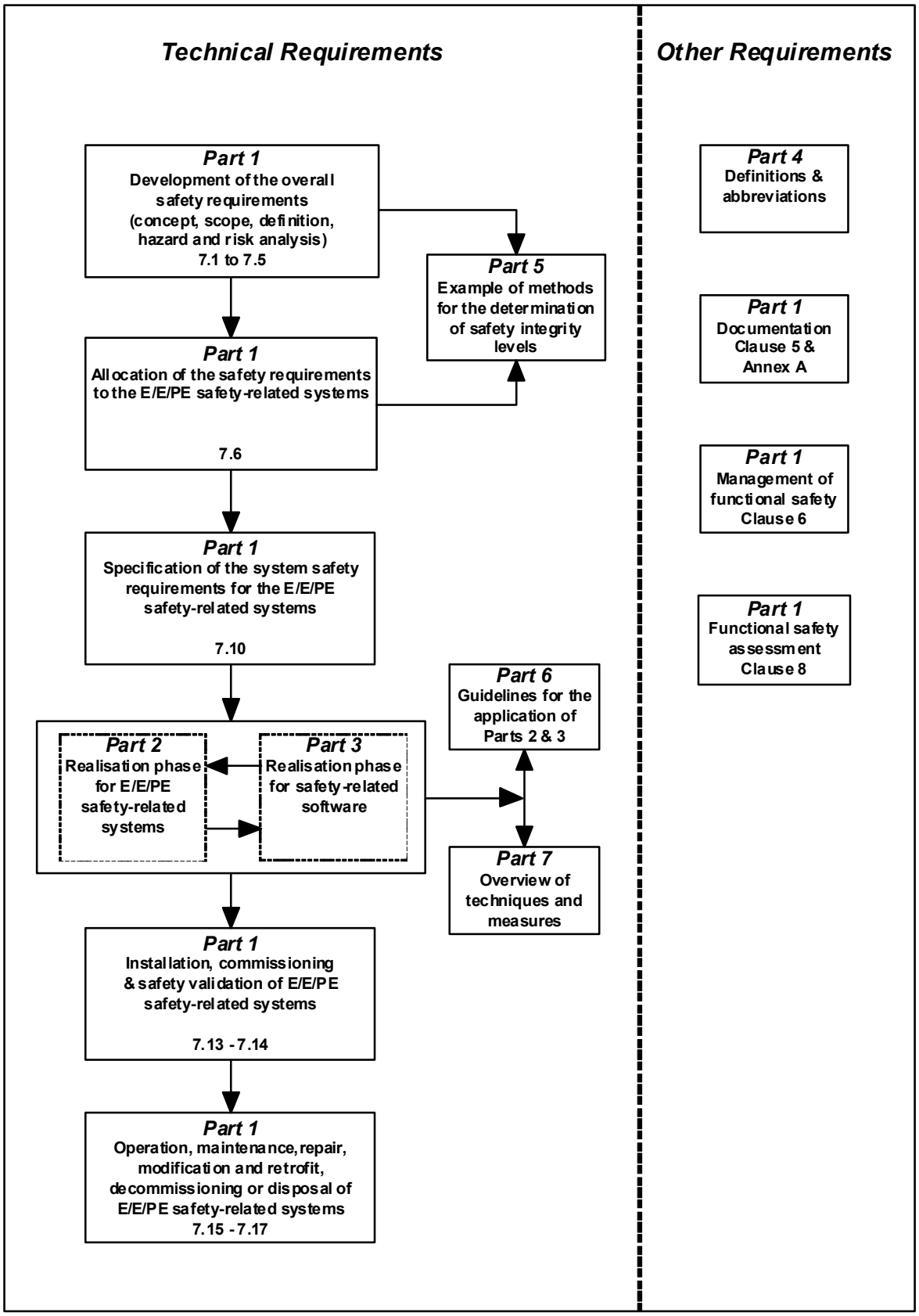


Figure 1 – Overall framework of the IEC 61508 series

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.

IEC 61508-2:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems*

IEC 61508-3:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 3: Software requirements*

IEC 61508-4:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations*

3 Definitions and abbreviations

For the purposes of this document, the definitions and abbreviations given in IEC 61508-4 apply.

Annex A
(informative)

Application of IEC 61508-2 and of IEC 61508-3

A.1 General

Machinery, process plant and other equipment may, in the case of malfunction (for example by failures of electrical, electronic and/or programmable electronic devices), present risks to people and the environment from hazardous events such as fires, explosions, radiation overdoses, machinery traps, etc. Failures can arise from either physical faults in the device (for example causing random hardware failures), or from systematic faults (for example human errors made in the specification and design of a system cause systematic failure under some particular combination of inputs), or from some environmental condition.

IEC 61508-1 provides an overall framework based on a risk approach for the prevention and/or control of failures in electro-mechanical, electronic, or programmable electronic devices.

The overall goal is to ensure that plant and equipment can be safely automated. A key objective of this standard is to prevent:

– failures of control systems triggering other events, which in turn could lead to danger (for example fire, release of toxic materials, repeat stroke of a machine, etc.); and

– undetected failures in protection systems (for example in an emergency shut-down system), making the systems unavailable when needed for a safety action.

IEC 61508-1 requires that a hazard and risk analysis at the process/machine level is carried out to determine the amount of risk reduction necessary to meet the risk criteria for the application. Risk is based on the assessment of both the consequence (or severity) and the frequency (or probability) of the hazardous event.

IEC 61508-1 further requires that the amount of risk reduction established by the risk analysis is used to determine if one or more safety-related systems[1] are required and what safety functions (each with a specified safety integrity)[2] they are needed for.

IEC 61508-2 and IEC 61508-3 take the safety functions and safety integrity requirements allocated to any system, designated as a E/E/PE safety-related system, by the application of IEC 61508-1 and establish requirements for safety lifecycle activities which:

– are to be applied during the specification, design and modification of the hardware and software; and

– focus on means for preventing and/or controlling random hardware and systematic failures (the E/E/PE system and software safety lifecycles)[3].

---

1 Systems necessary for functional safety and containing one or more electrical (electro-mechanical), electronic or programmable electronic (E/E/PE) devices are *designated* as E/E/PE safety-related systems and include all equipment necessary to carry out the required safety function (see 3.5.1 of IEC 61508-4).

2 Safety integrity is specified as one of four discrete levels. Safety integrity level 4 is the highest and safety integrity level 1 the lowest (see 3.5.4 and 3.5.8 of IEC 61508-4).

3 To enable the requirements of this standard to be clearly structured, a decision was made to order the requirements using a development process model in which each stage follows in a defined order with little iteration (sometimes referred to as a waterfall model). However, it is stressed that any lifecycle approach can be used provided a statement of equivalence is given in the safety plan for the project (see Clause 7 of IEC 61508-1).

IEC 61508-2 and IEC 61508-3 do not give guidance on which level of safety integrity is appropriate for a given required tolerable risk. This decision depends upon many factors, including the nature of the application, the extent to which other systems carry out safety functions and social and economic factors (see IEC 61508-1 and IEC 61508-5).

The requirements of IEC 61508-2 and IEC 61508-3 include:

– the application of measures and techniques[4], which are graded against the safety integrity level, for the avoidance of systematic failures[5] by preventative methods; and
– the control of systematic failures (including software failures) and random hardware failures by design features such as fault detection, redundancy and architectural features (for example diversity).

In IEC 61508-2, assurance that the safety integrity target has been satisfied for dangerous random hardware failures is based on:

– hardware fault tolerance requirements (see Tables 2 and 3 of IEC 61508-2); and
– the diagnostic coverage and frequency of proof tests of subsystems and components, by carrying out a reliability analysis using appropriate data.

In both IEC 61508-2 and IEC 61508-3, assurance that the safety integrity target has been satisfied for systematic failures is gained by:

– the correct application of safety management procedures;
– the use of competent staff;
– the application of the specified safety lifecycle activities, including the specified techniques and measures[6]; and
– an independent functional safety assessment[7].

The overall goal is to ensure that remaining systematic faults, commensurate with the safety integrity level, do not cause a failure of the E/E/PE safety-related system.

IEC 61508-2 has been developed to provide requirements for achieving safety integrity in the hardware[8] of the E/E/PE safety-related systems including sensors and final elements. Techniques and measures against both random hardware failures and systematic hardware failures are required. These involve an appropriate combination of fault avoidance and failure control measures as indicated above. Where manual action is needed for functional safety, requirements are given for the operator interface. Also diagnostic test techniques and measures, based on software and hardware (for example diversity), to detect random hardware failures are specified in IEC 61508-2.

IEC 61508-3 has been developed to provide requirements for achieving safety integrity for the software – both embedded (including diagnostic fault detection services) and application software. IEC 61508-3 requires a combination of fault avoidance (quality assurance) and fault tolerance approaches (software architecture), as there is no known way to prove the absence of faults in reasonably complex safety-related software, especially the absence of specification and design faults. IEC 61508-3 requires the adoption of such software

---

4 The required techniques and measures for each safety integrity level are shown in the tables in Annexes A and B of IEC 61508-2 and IEC 61508-3.

5 Systematic failures cannot usually be quantified. Causes include: specification and design faults in hardware and software; failure to take account of the environment (for example temperature); and operation-related faults (for example poor interface).

6 Alternative measures to those specified in the standard are acceptable provided justification is documented during safety planning (see Clause 6 of IEC 61508-1).

7 Independent assessment does not always imply third party assessment (see Clause 8 of IEC 61508-1).

8 Including fixed built-in software or software equivalents (also called firmware), such as application-specific integrated circuits.

engineering principles as: top down design; modularity; verification of each phase of the development lifecycle; verified software modules and software module libraries; and clear documentation to facilitate verification and validation. The different levels of software require different levels of assurance that these and related principles have been correctly applied.

The developer of the software may or may not be separate from the organization developing the whole E/E/PE system. In either case, close cooperation is needed, particularly in developing the architecture of the programmable electronics where trade-offs between hardware and software architectures need to be considered for their safety impact (see Figure 4 of IEC 61508-2).

A.2 Functional steps in the application of IEC 61508-2

The functional steps in the application of IEC 61508-2 are shown in Figures A.1 and A.2. The functional steps in the application of IEC 61508-3 are shown in Figure A.3.

Functional steps for IEC 61508-2 (see Figures A.1 and A.2) are as follows:

a) Obtain the allocation of safety requirements (see IEC 61508-1). Update the safety planning as appropriate during E/E/PE safety-related system development.

b) Determine the requirements for E/E/PE safety-related systems, including the safety integrity requirements, for each safety function (see 7.2 of IEC 61508-2). Allocate requirements to software and pass to software supplier and/or developer for the application of IEC 61508-3.

NOTE 1 The possibility of coincident failures in the EUC control system and E/E/PE safety-related system(s) needs to be considered at this stage (see A.5.4 of IEC 61508-5). These may result from failures of components having a common cause due to for example similar environmental influences. The existence of such failures could lead to a higher than expected residual risk unless properly addressed.

c) Start the phase of planning for E/E/PE safety-related system safety validation (see 7.3 of IEC 61508-2).

d) Specify the architecture (configuration) for the E/E/PE safety-related logic subsystem, sensors and final elements. Review with the software supplier/developer the hardware and software architecture and the safety implications of the trade-offs between the hardware and software (see Figure 4 of IEC 61508-2). Iterate if required.

e) Develop a model for the hardware architecture for the E/E/PE safety-related system. Develop this model by examining each safety function separately and determine the subsystem (component) to be used to carry out this function.

f) Establish the system parameters for each of the subsystems (components) used in the E/E/PE safety-related system. For each of the subsystems (elements), determine the following:

   – the proof test interval for failures which are not automatically revealed;
   – the mean time to restoration;
   – the diagnostic coverage (see Annex C of IEC 61508-2);
   – the probability of failure;
   – the required architectural constraints; for Route 1_H see 7.4.4.2 and Annex C of IEC 61508-2 and for Route 2_H see 7.4.4.3 of IEC 61508-2.

g) Create a reliability model for each of the safety functions that the E/E/PE safety-related system is required to carry out.

NOTE 2 A reliability model is a mathematical formula which shows the relationship between reliability and relevant parameters relating to equipment and conditions of use.

h) Calculate a reliability prediction for each safety function using an appropriate technique. Compare the result with the target failure measure determined in b) above and the requirements of Route 1_H (see 7.4.4.2 of IEC 61508-2) or Route 2_H (see 7.4.4.3 of

IEC 61508-2). If the predicted reliability does not meet the target failure measure and/or does not meet the requirements of Route 1_H or Route 2_H , then change

- where possible, one or more of the subsystem parameters (go back to f) above); and/or
- the hardware architecture (go back to d) above).

NOTE 3 A number of modelling methods are available and the analyst should choose which is the most appropriate (see Annex B for guidance on some methods that could be used).

i) Implement the design of the E/E/PE safety-related system. Select measures and techniques to control systematic hardware failures, failures caused by environmental influences and operational failures (see Annex A of IEC 61508-2).

j) Integrate the verified software (see IEC 61508-3) onto the target hardware (see 7.5 of IEC 61508-2 and Annex B of IEC 61508-2) and, in parallel, develop the procedures for users and maintenance staff to follow when operating the system (see 7.6 of IEC 61508-2 and Annex B of IEC 61508-2). Include software aspects (see A.3 f)).

k) Together with the software developer (see 7.7 of IEC 61508-3), validate the E/E/PE system (see 7.7 of IEC 61508-2 and Annex B of IEC 61508-2).

l) Hand over the hardware and results of the E/E/PE safety-related system safety validation to the system engineers for further integration into the overall system.

m) If maintenance/modification of the E/E/PE safety related system is required during operational life then re-activate IEC 61508-2 as appropriate (see 7.8 of IEC 61508-2).

A number of activities run across the E/E/PE safety related system safety lifecycle. These include verification (see 7.9 of IEC 61508-2) and functional safety assessment (see Clause 8 of IEC 61508-1).

In applying the above steps the E/E/PE safety related system safety techniques and measures appropriate to the required safety integrity level are selected. To aid in this selection, tables have been formulated, ranking the various techniques/measures against the four safety integrity levels (see Annex B of IEC 61508-2). Cross-referenced to the tables is an overview of each technique and measure with references to further sources of information (see Annexes A and B of IEC 61508-7).

Annex B provides one possible technique for calculating the probabilities of hardware failure for E/E/PE safety-related systems.

NOTE 4 In applying the above steps, alternative measures to those specified in the standard are acceptable provided justification is documented during safety planning (see Clause 6 of IEC 61508-1).

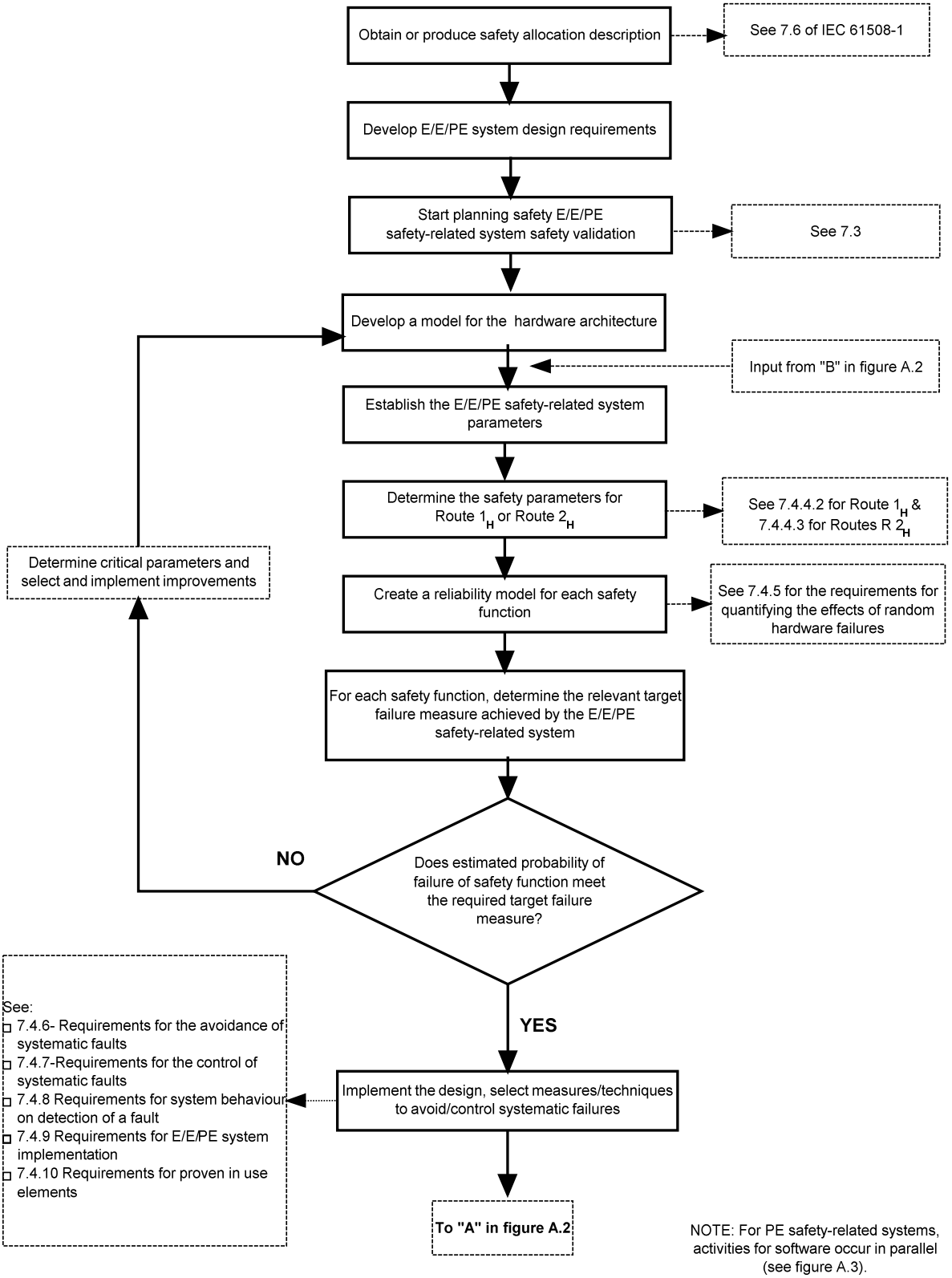
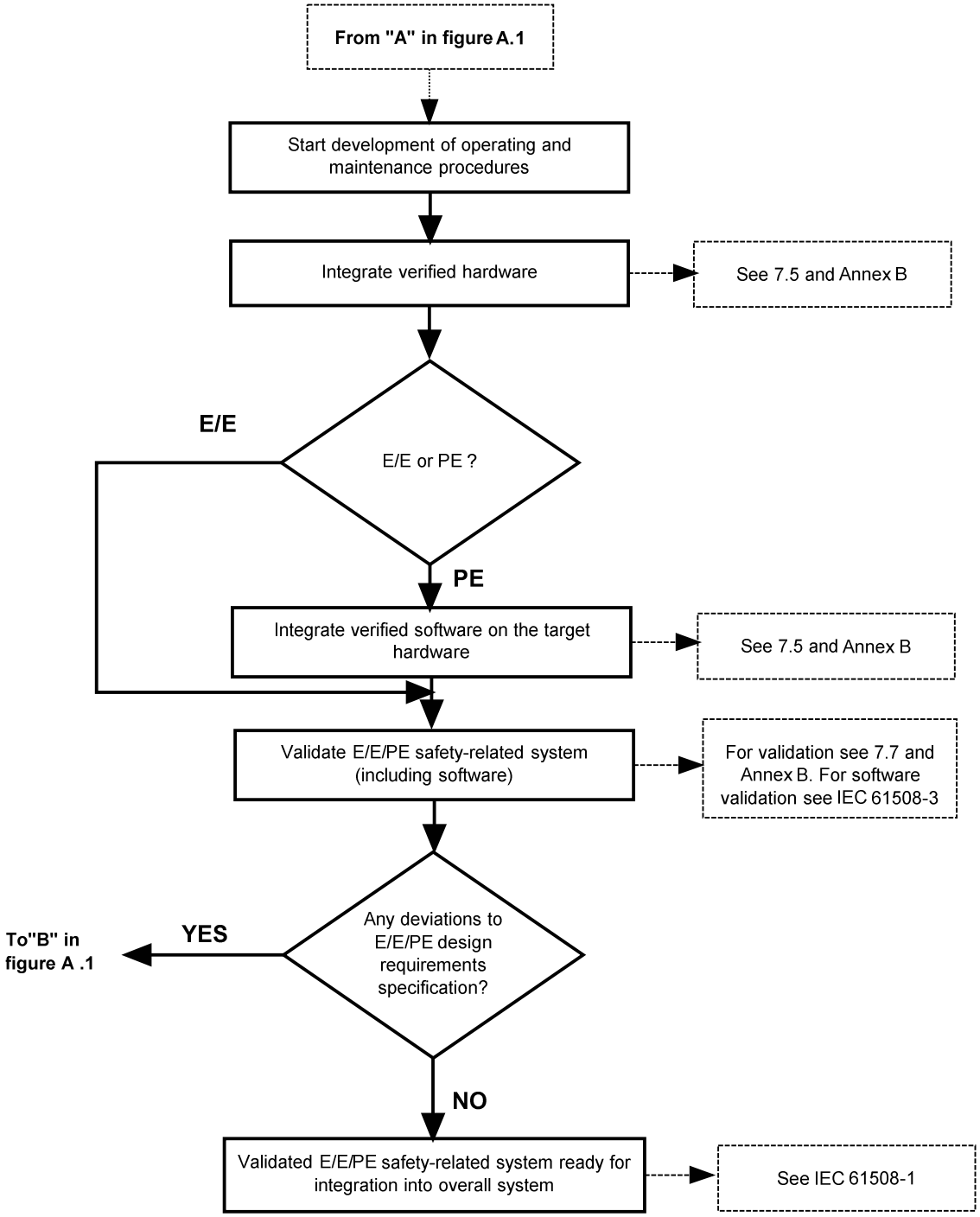


Figure A.1 – Application of IEC 61508-2



NOTE: For PE safety-related systems, activities for software occur in parallel (see figure A.3).

Figure A.2 – Application of IEC 61508-2 (Figure A.1 *continued*)

A.3 Functional steps in the application of IEC 61508-3

Functional steps for IEC 61508-3 (see Figure A.3) are as follows.

a) Obtain the requirements for the E/E/PE safety-related systems and relevant parts of the safety planning (see 7.3 of IEC 61508-2). Update the safety planning as appropriate during software development.

NOTE 1 Earlier lifecycle phases have already:

– specified the required safety functions and their associated safety integrity levels (see 7.4 and 7.5 of IEC 61508-1);

– allocated the safety functions to designated E/E/PE safety-related systems (see 7.6 of IEC 61508-1); and

– allocated functions to software within each E/E/PE safety-related system (see 7.2 of IEC 61508-2).

b) Determine the software architecture for all safety functions allocated to software (see 7.4 of IEC 61508-3 and Annex A of IEC 61508-3).

c) Review with the E/E/PE safety-related system's supplier/developer, the software and hardware architecture and the safety implications of the trade-offs between the software and hardware (see Figure 4 of IEC 61508-2). Iterate if required.

d) Start the planning for software safety verification and validation (see 7.3 and 7.9 of IEC 61508-3).

e) Design, develop and verify/test the software according to the:

– software safety planning;

– software safety integrity level; and

– software safety lifecycle.

f) Complete the final software verification activity and integrate the verified software onto the target hardware (see 7.5 of IEC 61508-3), and in parallel develop the software aspects of the procedures for users and maintenance staff to follow when operating the system (see 7.6 of IEC 61508-3, and A.2 k)).

g) Together with the hardware developer (see 7.7 of IEC 61508-2), validate the software in the integrated E/E/PE safety-related systems (see 7.7 of IEC 61508-3).

h) Hand over the results of the software safety validation to the system engineers for further integration into the overall system.

i) If modification of the E/E/PE safety-related system software is required during operational life then re-activate this IEC 61508-3 phase as appropriate (see 7.8 of IEC 61508-3).

A number of activities run across the software safety lifecycle. These include verification (see 7.9 of IEC 61508-3) and functional safety assessment (see Clause 8 of IEC 61508-3).

In applying the above steps, software safety techniques and measures appropriate to the required safety integrity are selected. To aid in this selection, tables have been formulated ranking the various techniques/measures against the four safety integrity levels (see Annex A of IEC 61508-3). Cross-referenced to the tables is an overview of each technique and measure with references to further sources of information (see Annex C of IEC 61508-7).

Worked examples in the application of the safety integrity tables are given in Annex E, and IEC 61508-7 includes a probabilistic approach to determining software safety integrity for pre-developed software (see Annex D of IEC 61508-7).

NOTE 2 In applying the above steps, alternative measures to those specified in the standard are acceptable provided justification is documented during safety planning (see Clause 6 of IEC 61508-1).

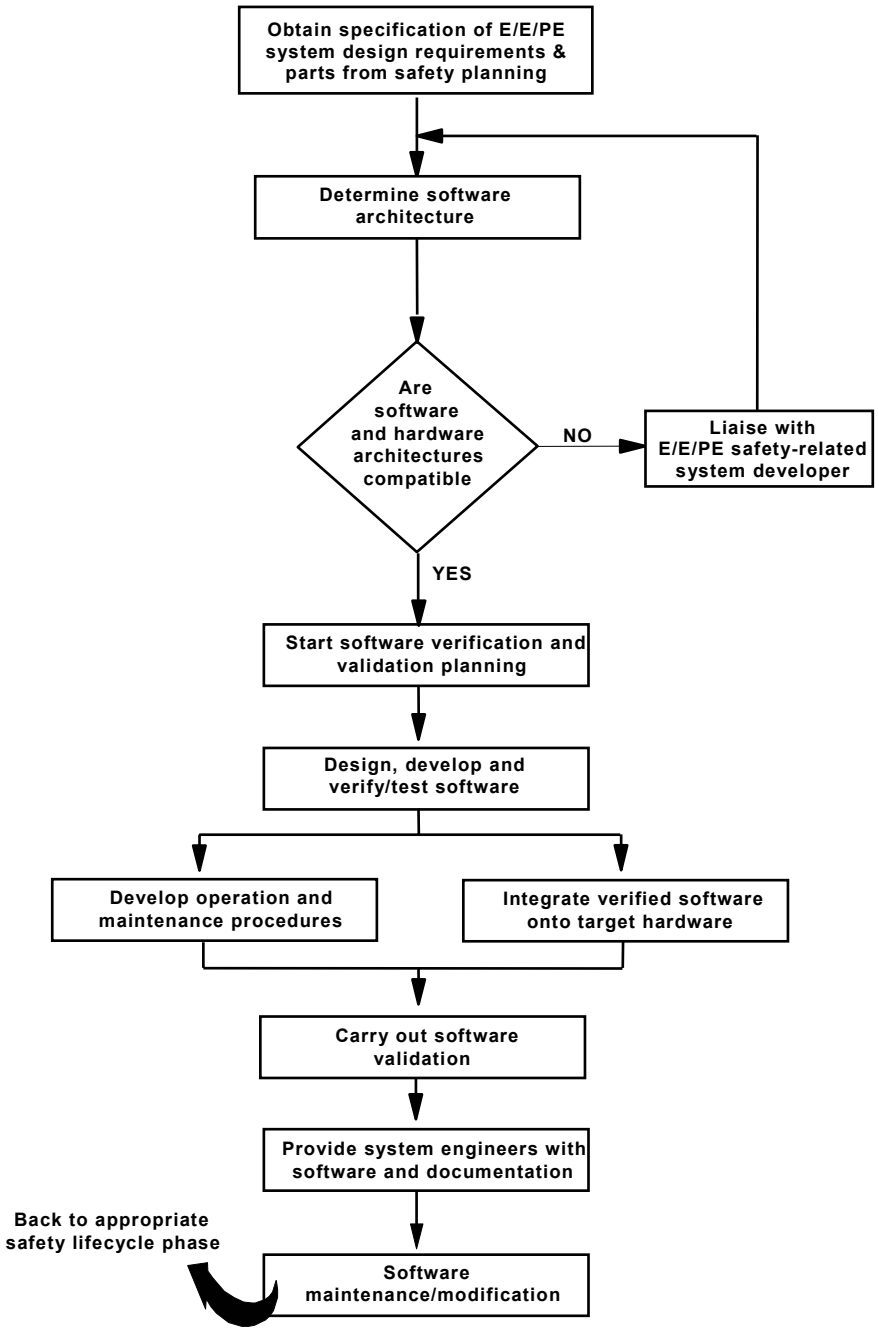


Figure A.3 – Application of IEC 61508-3

Annex B
(informative)

Example of technique for evaluating probabilities of hardware failure

B.1 General

This annex provides possible techniques for calculating the probabilities of hardware failure for E/E/PE safety-related systems installed in accordance with IEC 61508-1, IEC 61508-2 and IEC 61508-3. The information provided is informative in nature and should not be interpreted as the only evaluation techniques that might be used. It does, however, provide both a relatively simple approach for assessing the capability of E/E/PE safety-related systems and guidelines to use alternative techniques derived from the classical reliability calculations techniques.

NOTE 1 System architectures stated in this part are provided by way of examples and should not be considered exhaustive as there are many other architectures that may be used.

NOTE 2 See ISA-TR84.00.02-2002 [17] in the Bibliography.

A number of reliability techniques are more or less straightforwardly usable for the analysis of hardware safety integrity of E/E/PE safety-related systems. Classically, they are sorted according to the two following point of views:

– static (Boolean) versus dynamic (states/transitions) models;

– analytical versus Monte Carlo simulation calculations.

Boolean models encompass all models describing the static logical links between the elementary failures and the whole system failure. Reliability Block Diagrams (RBD) (see C.6.4 of IEC 61508-7 and IEC 61078 [4]) and Fault Trees (FT) (see B.6.6.5 and B.6.6.9 of IEC 61508-7) and IEC 61025 [18] belong to Boolean models.

States/transitions models encompass all models describing how the system behaves (jumps from states to states) according to arising events (failures, repairs, tests, etc.). Markovian (see B.6.6.6 of IEC 61508-7 and IEC 61165 [5]), Petri nets (see B.2.3.3 and B.6.6.10 of IEC 61508-7 and IEC 62551 [19]) and formal language models belong to states/transitions models. Two Markovian approaches are investigated: a simplified approach based on specific formulae (B.3) and a general approach allowing direct calculations on Markov graphs (B.5.2). For non Markovian safety systems, Monte Carlo simulations can be used instead. With present time personal computers this is achievable even for SIL 4 calculations. Subclauses B.5.3 and B.5.4 of this annex provides guidelines about handling Monte Carlo simulations (see B.6.6.8 of IEC 61508-7) on behavioural models based on Petri nets and formal languages modelling.

The simplified approach which is presented first is based on RBD graphical representations and specific Markovian formulae obtained from Taylor's developments and slightly conservative underlying hypotheses described in B.3.1.

All these methods can be used for the majority of safety related systems and, when deciding which technique to use on any particular application, it is very important that the user of a particular technique is competent in using the technique and this may be more important than the technique which is actually used. It is the responsibility of the analyst to verify that the underling hypotheses of any particular method are satisfied or any adjustments are required to obtain an adequate realist conservative result. In case of poor reliability data or dominant common cause failure, it may be sufficient to use the simplest model / techniques. Whether the loss of accuracy is significant can only be determined in the particular circumstances.

If software programmes are used to perform the calculations then the practitioner shall have an understanding of the formulae/techniques used by the software package to ensure its use is suitable for the specific application. The practitioner should also verify the software package by checking its output with some manual calculated test cases.

Where a failure of the EUC control system places a demand on the E/E/PE safety-related system, then the probability of a hazardous event occurring also depends on the probability of failure of the EUC control system. In that situation, it is necessary to consider the possibility of co-incident failure of components in the EUC control system and the E/E/PE safety-related system due to common cause failure mechanisms. The existence of such failures could lead to a higher than expected residual risk unless properly addressed.

B.2 Considerations about basic probabilistic calculations

B.2.1 Introduction

The reliability block diagram (RBD) on Figure B.1 is representing a safety loop made of three sensors (A, B, C), one logic solver (D), two final elements (E, F), and common cause failures (CCF).

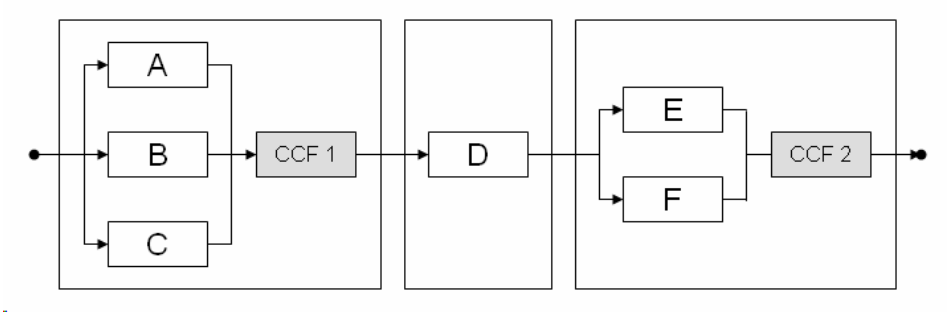


Figure B.1 – Reliability Block Diagram of a whole safety loop

This facilitates the identification of five failure combinations leading to the E/E/PE safety-related system failure. Each of them is a so-called minimal cut set:

– (A, B, C) is a triple failure;

– (E, F) is a double failure;

– (D) (CCF1) (CCF2) are single failures.

B.2.2 Low demand E/E/PE safety-related system

When a E/E/PE safety-related system is used in low demand mode, the standard requires that its PFD_{avg} (i.e. its average unavailability) be assessed. This is simply the ratio $MDT(T)/T$ where $MDT(T)$ is the mean down time over the period [0, T] of the E/E/PE safety-related system.

For safety system the probability of failure is, normally, very low and the probability to have two minimal cut sets at the same time is negligible. Therefore, the sum of the mean down times due to each cut sets gives a conservative estimate of the mean down time of the whole system. From Figure B.1 we find:

$$MDT \approx MDT^{ABC} + MDT^{D} + MDT^{EF}$$

Dividing by T gives:

$$PFD_{avg} \approx PFD_{avg}^{ABC} + PFD_{avg}^{D} + PFD_{avg}^{EF}$$

Therefore, for parts in series, PFD_{avg} calculations are very similar to those performed with ordinary probabilities when they are very small compared to 1.

However for parallel parts where multiple failure are required before the loss of the function like (E, F), it is clear that MDT^{EF} may not be calculated straightforwardly from MDT^{E} and MDT^{F}: The (E,F) system's *MDT* has to be calculated as

$$MDT^{EF} = \int_0^T PFD^E(t) PFD^F(t) dt$$

Therefore, ordinary probabilistic calculations (additions and multiplications) are no longer valid for PFD_{avg} calculations (integrals) of parts in parallel. PFD_{avg} has not the same properties as a genuine probability and its assimilation with a genuine probability is likely to lead to non conservative results. In particular, it is not possible to obtain the PFD_{avg} of an E/E/PE safety-related system just by combining in a conventional way the $PFD_{avg,i}$ of its components. As this is sometimes encouraged by commercial Boolean software packages, analysts should be vigilant to avoid such non conservative calculations which are undesirable when dealing with safety.

EXAMPLE For a redundant (1oo2) channel with a dangerous undetected failure rate λ_{DU} with a proof test interval τ, an incorrect probability model calculation could give $(\lambda_{DU}.\tau)^2/4$ when the actual result is $(\lambda_{DU}.\tau)^2/3$.

Calculations may be performed analytically or by using Monte Carlo simulation. This annex describes how to do that by using conventional reliability models based on Boolean (RBD or Fault-trees) or, states/transitions models (Markov, Petri nets, etc.).

B.2.3 Continuous or high demand mode E/E/PE safety-related system

B.2.3.1 General *PFH* formula

When an E/E/PE safety-related system is used in continuous or high demand mode, the standard requires the calculation of its *PFH* (i.e. its average frequency of dangerous failure). This is the average of the so called unconditional failure intensity (also called failure frequency) *w(t)* over the period of interest:

$$PFH(T) = \frac{1}{T}\int_0^T w(t) dt$$

Where the E/E/PE safety-related system is working in continuous mode and is the ultimate safety barrier, then the overall safety-related system failure will lead directly to a potentially hazardous situation. Hence for failures that cause the loss of the overall safety function no overall safety-related system repair can be considered in the calculations. However, if the failure of the overall safety-related system does not lead directly to the potential hazard due to some other safety barrier or equipment failure then it may be possible to consider the detection and repair of the safety-related system in its risk reduction calculation.

B.2.3.2 Un-reliability case (e.g. single barrier working in continuous mode)

This case is relevant when E/E/PE safety-related system working in continuous mode is the ultimate safety barrier. Therefore a potential hazard can occur as soon as it is failing. No overall system failures are acceptable over the period of interest.

In this case the *PFH* may be calculated by using the unreliability over the period of interest:

$$F(T):\ PFH(T) = \frac{1 - \exp\left[-\int_0^T \Lambda(t)dt\right]}{T} = \frac{F(T)}{T}$$

The overall system failure rate, $\Lambda(t)$, may be time dependent or constant.

When it is time dependant, we have: $PFH(T) = \frac{1 - \exp(-\Lambda_{avg}.T)}{T} \approx \Lambda_{avg}$

When the system is made of components completely and quickly repairable with constant failure and repair rates (e.g. dangerous detected failures), $\Lambda(t)$ reaches quickly an asymptotic constant value Λ_{as} and, when $PFH(T) << 1.$, we have:

$$PFH(T) = \frac{1 - \exp(-\Lambda_{as}.T)}{T} \approx \Lambda_{as} = \frac{1}{MTTF}$$

Λ_{as} exists only when the E/E/PE safety-related system working in continuous mode comprises only safe and DD failures (i.e. quickly detected and repaired). No repair of failures that can directly cause the overall failure of the safety function can be considered. For a redundant configuration where it is relevant to consider proof tests, then the asymptotic failure rate is not relevant and the previous equations have to be used. It is the job of the analyst to verify which case is relevant.

B.2.3.3 Unavailability case (e.g. multiple safety barriers)

When the E/E/PE safety-related system working in continuous mode is not the ultimate barrier, its failures only increase the demand frequency on other safety barriers, or when it is working in the high demand mode, such that it is possible to detect (automatically or manually) and repair a fault that could cause the direct loss of the safety function within the expected demand period. In this case its overall failures can be repaired and *PFH* may be calculated from the availability, *A(t),* and from the conditional failure intensity, $\Lambda_V(t)$, of the system.

Again, when the system is made of components completely and quickly repairable (i.e. when, in any degraded situation, there is an high probability to come back quickly to a good working state), $\Lambda_V(t)$ reaches quickly its asymptotic value, Λ_{Vas}, which, in addition, is also a good approximation of the true asymptotic overall system failure rate, $\Lambda_{as,}$ introduced in B.2.3.2.

This leads to the following approximations:

$$PFH = \frac{1}{MUT + MDT} = \frac{1}{MTBF} \approx \frac{1}{MUT} \approx \frac{1}{MTTF}$$

where

MUT is the acronym for Mean Up Time;

MDT is the acronym for Mean Down Time;

MTBF is the acronym for Mean Time Between Failure; and

MTTF is the acronym for Mean Time To Failure.

B.2.3.4 Failure rate considerations

Several formulae above use the overall system failure rate $\Lambda(t)$. Its evaluation is not so easy and some reminders are needed.

Series structures are very simple to handle as failure rates can be added. From Figure B.1, we can write $\Lambda(t) = \Lambda^{abc}(t) + \lambda^{CCF1}(t) + \lambda^{d}(t) + \Lambda^{ef}(t) + \lambda^{CCF2}(t)$ where $\Lambda(t)$ is the overall failure rate of the E/E/PE safety-related system and $\Lambda^{abc}(t)$, $\lambda^{CCF1}(t)$, $\lambda^{d}(t)$, $\Lambda^{ef}(t)$ and $\lambda^{CCF2}(t)$ the failure rates of the five minimal cut sets.

For parallel structures this is not so simple because there are no simple relationships with individual components failure rates. For example, let us consider cut set (E, F):

1) When E and F cannot be immediately be restored (e.g. DU failures), $\Lambda^{ef}(t)$ varies continuously from 0 to λ (failure rate of E or F). An asymptotic value is reached when one of the two components is likely to have failed. This is a very slow process as this arises when t becomes larger than $1/\lambda$. This asymptotic value will be never reached in actual life if E and F are periodically proof tested with a test interval $\tau << 1/\lambda$.
2) When E and F are restored in a relatively short period of time (e.g. DD failures), $\Lambda^{ef}(t)$ goes very fast to an asymptotic value $\Lambda^{ef}_{As} = 2\lambda^2/\mu$ which can be used as an equivalent constant failure rate. It is reached when t becomes greater than two or three times the larger *MTTR* of the components. It is a particular case of the completely and quickly repairable systems discussed above.

Therefore, in the general case, evaluating the overall system failure rates imply more complex calculations than the more simple series structure.

B.3 Reliability block diagram approach, assuming constant failure rate

B.3.1 Underlying hypothesis

The calculations are based on the following assumptions:

– the resulting average probability of failure on demand for the system is less than 10^{-1}, or the resultant average frequency of dangerous failure for the system is less than 10^{-5} h^{-1}

NOTE 1 This assumption means that the E/E/PE safety-related system is within the scope of IEC 61508 series and within the SIL 1 band (see Tables 2 and 3 of IEC 61508-1).

– component failure rates are constant over the life of the system;

– the sensor (input) subsystem comprises the actual sensor(s) and any other components and wiring, up to but not including the component(s) where the signals are first combined by voting or other processing (for example for two sensor channels, the configuration would be as shown in Figure B.2);

– the logic subsystem comprises the component(s) where the signals are first combined, and all other components up to and including where final signal(s) are presented to the final element subsystem;

– the final element (output) subsystem comprises all the components and wiring which process the final signal(s) from the logic subsystem including the final actuating component(s);

– the hardware failure rates used as inputs to the calculations and tables are for a single channel of the subsystem (for example, if 2oo3 sensors are used, the failure rate is for a single sensor and the effect of 2oo3 is calculated separately);

– the channels in a voted group all have the same failure rates and diagnostic coverage;

– the overall hardware failure rate of a channel of the subsystem is the sum of the dangerous failure rate and safe failure rate for that channel, which are assumed to be equal;

NOTE 2 This assumption affects the safe failure fraction (see Annex C of IEC 61508-2), but the safe failure fraction does not affect the calculated values for probability of failure given in this annex.

– for each safety function, there is perfect proof testing and repair (i.e. all failures that remain undetected are detected by the proof test), but for the effects of a non-perfect proof test see B.3.2.5;

– the proof test interval is at least an order of magnitude greater than the MRT;

– for each subsystem there is a single proof test interval and MRT;

– the expected interval between demands is at least an order of magnitude greater than the proof test interval;

– for all subsystems operating in low demand mode of operation, and for 1oo2, 1oo2D,1oo3 and 2oo3, voted groups operating in high demand or continuous mode of operation, the fraction of failures specified by the diagnostic coverage is both detected and repaired within the MTTR used to determine hardware safety integrity requirements;

EXAMPLE If a MTTR of 8 h is assumed, this includes the diagnostic test interval which is typically less than 1 h, the remainder being the MRT.

NOTE 3 For 1oo2, 1oo2D, 1oo3 and 2oo3 voted groups, it is assumed that any repair is on-line. Configuring an E/E/PE safety-related system, so that on any detected fault the EUC is put into a safe state, improves the average probability of failure on demand. The degree of improvement depends on the diagnostic coverage.

– for 1oo1 and 2oo2 voted groups operating in high demand or continuous mode of operation, the E/E/PE safety-related system always achieves a safe state after detecting a dangerous fault; to achieve this, the expected interval between demands is at least an order of magnitude greater than the diagnostic test intervals, or the sum of the diagnostic test intervals and the time to achieve a safe state is less than the process safety time;

NOTE 4 For process safety time see 3.6.20 of IEC 61508-4.

– when a power supply failure removes power from a de-energize-to-trip E/E/PE safety-related system and initiates a system trip to a safe state, the power supply does not affect the average probability of failure on demand of the E/E/PE safety-related system; if the system is energized to trip, or the power supply has failure modes that can cause unsafe operation of the E/E/PE safety-related system, the power supply should be included in the evaluation;

– where the term channel is used, it is limited to only that part of the system under discussion, which is usually either the sensor, logic or final element subsystem;

– the abbreviated terms are described in Table B.1.

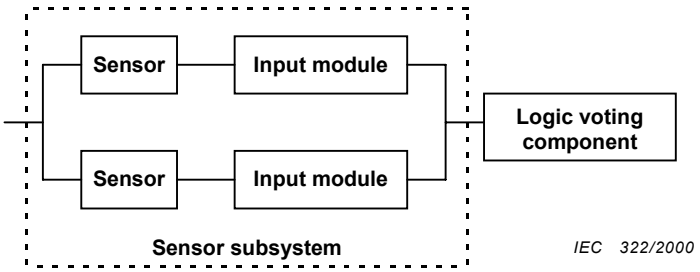


Figure B.2 – Example configuration for two sensor channels

Table B.1 – Terms and their ranges used in this annex
(applies to 1oo1, 1oo2, 2oo2, 1oo2D, 1oo3 and 2oo3)

Abbre-viation	Term (units)	Parameter ranges in Tables B.2 to B.5 and B.10 to B.13
T_1	Proof test interval (hour)	One month (730 h)[1] Three months (2 190 h)[1] Six months (4 380 h) One year (8 760 h) Two years (17 520 h)[2] Ten years (87 600 h)[2]
MTTR	Mean time to restoration (hour)	8 h NOTE MTTR = MRT = 8 hours is based on the assumption that the time to detect a dangerous failure, based on automatic detection, is << MRT.
MRT	Mean repair time (hour)	8 h NOTE MTTR = MRT = 8 hours is based on the assumption that the time to detect a dangerous failure, based on automatic detection, is << MRT
DC	Diagnostic coverage (expressed as a fraction in the equations and as a percentage elsewhere)	0 % 60 % 90 % 99 %
β	The fraction of undetected failures that have a common cause (expressed as a fraction in the equations and as a percentage elsewhere) (Tables B.2 to B.5 and B.10 to B.13 assume $\beta = 2 \times \beta_D$)	2 % 10 % 20 %
β_D	Of those failures that are detected by the diagnostic tests, the fraction that have a common cause (expressed as a fraction in the equations and as a percentage elsewhere) (Tables B.2 to B.5 and B.10 to B.13 assume $\beta = 2 \times \beta_D$)	1 % 5 % 10 %
λ_{DU}	Dangerous Undetected failure rate (per hour) of a channel in a subsystem	$0{,}05 \times 10^{-6}$ $0{,}25 \times 10^{-6}$ $0{,}5 \times 10^{-6}$ $2{,}5 \times 10^{-6}$ 5×10^{-6} 25×10^{-6}
PFD_G	Average probability of failure on demand for the group of voted channels (If the sensor, logic or final element subsystem comprises of only one voted group, then PFD_G is equivalent to PFD_S, PFD_L or PFD_{FE} respectively)	
PFD_S	Average probability of failure on demand for the sensor subsystem	
PFD_L	Average probability of failure on demand for the logic subsystem	

Abbre-viation	Term (units)	Parameter ranges in Tables B.2 to B.5 and B.10 to B.13
PFD_{FE}	Average probability of failure on demand for the final element subsystem	
PFD_{SYS}	Average probability of failure on demand of a safety function for the E/E/PE safety-related system	
PFH_G	Average frequency of dangerous failure for the group of voted channels (if the sensor, logic or final element subsystem comprises of only one voted group, then PFH_G is equivalent to PFH_S, PFH_L or PFH_{FE} respectively)	
PFH_S	Average frequency of dangerous failure for the sensor subsystem	
PFH_L	Average frequency of dangerous failure for the logic subsystem	
PFH_{FE}	Average frequency of dangerous failure for the final element subsystem	
PFH_{SYS}	Average frequency of dangerous failure of a safety function for the E/E/PE safety-related system	
λ	Total failure rate (per hour) of a channel in a subsystem	
λ_D	Dangerous failure rate (per hour) of a channel in a subsystem, equal to 0,5 λ (assumes 50 % dangerous failures and 50 % safe failures)	
λ_{DD}	Detected dangerous failure rate (per hour) of a channel in a subsystem (this is the sum of all the detected dangerous failure rates within the channel of the subsystem)	
λ_{DU}	Undetected dangerous failure rate (per hour) of a channel in a subsystem (this is the sum of all the undetected dangerous failure rates within the channel of the subsystem)	
λ_{SD}	Detected safe failure rate (per hour) of a channel in a subsystem (this is the sum of all the detected safe failure rates within the channel of the subsystem)	
t_{CE}	Channel equivalent mean down time (hour) for 1oo1, 1oo2, 2oo2 and 2oo3 architectures (this is the combined down time for all the components in the channel of the subsystem)	
t_{GE}	Voted group equivalent mean down time (hour) for 1oo2 and 2oo3 architectures (this is the combined down time for all the channels in the voted group)	
t_{CE}'	Channel equivalent mean down time (hour) for 1oo2D architecture (this is the combined down time for all the components in the channel of the subsystem)	
t_{GE}'	Voted group equivalent mean down time (hour) for 1oo2D architecture (this is the combined down time for all the channels in the voted group)	
T_2	Interval between demands (h)	
K	Fraction of the success of the autotest circuit in the 1oo2D system	
PTC	Proof Test Coverage	
[1] High demand or continuous mode only.		
[2] Low demand mode only.		

B.3.2 Average probability of failure on demand (for low demand mode of operation)

B.3.2.1 Procedure for calculations

The average probability of failure on demand of a safety function for the E/E/PE safety-related system is determined by calculating and combining the average probability of failure on demand for all the subsystems which together implement the safety function. Since in this annex the probabilities are small, this can be expressed by the following (see Figure B.3):

$$PFD_{SYS} = PFD_S + PFD_L + PFD_{FE}$$

where

PFD_{SYS} is the average probability of failure on demand of a safety function for the E/E/PE safety-related system;

PFD_S is the average probability of failure on demand for the sensor subsystem;

PFD_L is the average probability of failure on demand for the logic subsystem; and

PFD_{FE} is the average probability of failure on demand for the final element subsystem.

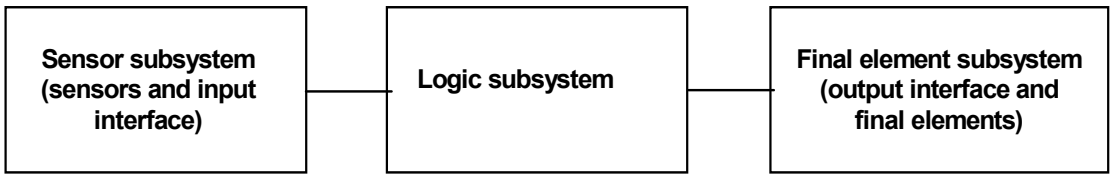


Figure B.3 – Subsystem structure

To determine the average probability of failure on demand for each of the subsystems, the following procedure should be adhered to for each subsystem in turn.

a) Draw the block diagram showing the sensor subsystem (input) components, logic subsystem components or final element subsystem (output) components. For example, sensor subsystem components may be sensors, barriers, input conditioning circuits; logic subsystem components may be processors and scanning devices; and final element subsystem components may be output conditioning circuits, barriers and actuators. Represent each subsystem as one or more 1oo1, 1oo2, 2oo2, 1oo2D, 1oo3 or 2oo3 voted groups.

b) Refer to the relevant table from Tables B.2 to B.5 which are for six-month, one-year, two-year and 10-year proof test intervals. These tables also assume an 8 h mean time to restoration for each failure once it has been revealed.

c) For each voted group in the subsystem, select from the relevant table of Tables B.2 to B.5:

   – architecture (for example, 2oo3);

   – diagnostic coverage of each channel (for example, 60 %);

   – the dangerous failure rate (per hour), λ_D, of each channel (for example, $2{,}5 \times 10^{-06}$);

   – the common cause failure β-factors, β and β_D, for the interaction between the channels in the voted group (for example, 2 % and 1 % respectively).

NOTE 1 It is assumed that every channel in the voted group has the same diagnostic coverage and failure rate (see Table B.1).

NOTE 2 It is assumed in Tables B.2 to B.5 (and in Tables B.10 to B.13) that the β-factor in the absence of diagnostic tests (also used for undetected dangerous failures in the presence of diagnostic tests), β, is 2 times the β-factor for failures detected by the diagnostic tests, β_D.

d) Obtain, from the relevant table from Tables B.2 to B.5, the average probability of failure on demand for the voted group.

e) If the safety function depends on more than one voted group of sensors or actuators, the combined average probability of failure on demand of the sensor or final element subsystem, PFD_S or PFD_{FE}, is given in the following equations, where PFD_{Gi} and PFD_{Gj} is the average probability of failure on demand for each voted group of sensors and final elements respectively:

$$PFD_S = \sum_i PFD_{Gi}$$

$$PFD_{FE} = \sum_j PFD_{Gj}$$

The formula used in all the equations for both *PFD* and system failure rate are all a function of component failure rate and mean down time (*MDT*). Where there are a number of elements in the system and it is required to calculate the combined elements overall *PFD* or system failure rate, then it is often necessary to use a single value for the *MDT* in the equations. However each element may have different failure detection mechanisms with different *MDT* and different elements may have different *MDT* values for the same failure mechanisms, in which case it is necessary to calculate a single value for the *MDT* which can represent all the elements in the path. This can be accomplished by considering the total paths overall failure rate then proportioning the individual *MDT's* equivalent to their failure rate contribution to the total failure rate under consideration.

As an example, if there are two elements in series but one with a proof test, T_1, and the other with a proof test, T_2, then the equivalent single value for the *MDT* is:

$$\lambda_T = \lambda_1 + \lambda_2$$

and

$$MDT_E = \frac{\lambda_1}{\lambda_T}\left(\frac{T_1}{2}\right) + \frac{\lambda_2}{\lambda_T}\left(\frac{T_2}{2}\right)$$

B.3.2.2 Architectures for low demand mode of operation

NOTE 1 This subclause should be read sequentially, since equations which are valid for several architectures are only stated where they are first used.

NOTE 2 The equations are based on the assumptions listed in B.3.1.

NOTE 3 The following examples are typical configurations and are not intended to be an exhaustive.

B.3.2.2.1 1oo1

This architecture consists of a single channel, where any dangerous failure leads to a failure of the safety function when a demand arises.

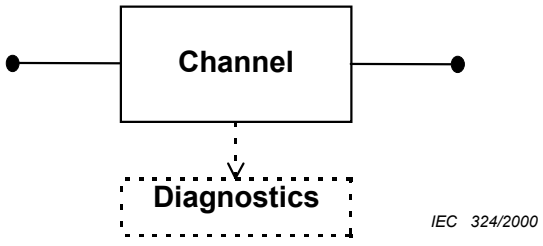


Figure B.4 – 1oo1 physical block diagram

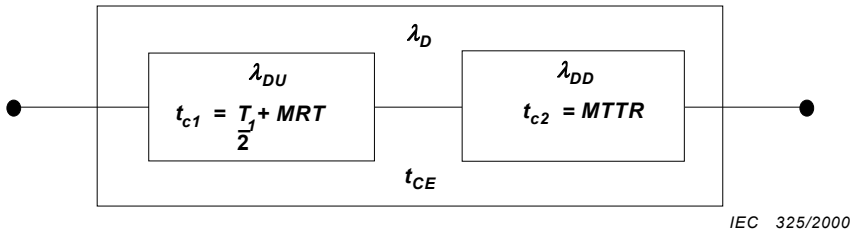


Figure B.5 – 1oo1 reliability block diagram

Figures B.4 and B.5 contain the relevant block diagrams. The dangerous failure rate for the channel is given by

$$\lambda_D = \lambda_{DU} + \lambda_{DD}$$

Figure B.5 shows that the channel can be considered to comprise of two components, one with a dangerous failure rate λ_{DU} resulting from undetected failures and the other with a dangerous failure rate λ_{DD} resulting from detected failures. It is possible to calculate the channel equivalent mean down time t_{CE}, adding the individual down times from both components, t_{c1} and t_{c2}, in direct proportion to each component's contribution to the probability of failure of the channel:

$$t_{CE} = \frac{\lambda_{DU}}{\lambda_D}\left(\frac{T_1}{2} + MRT\right) + \frac{\lambda_{DD}}{\lambda_D} MTTR$$

For every architecture, the detected dangerous failure rate and the undetected dangerous failure rate are given by

$$\lambda_{DU} = \lambda_D(1 - DC); \quad \lambda_{DD} = \lambda_D DC$$

For a channel with down time t_{CE} resulting from dangerous failures

$$PFD = 1 - e^{-\lambda_D t_{CE}}$$
$$\approx \lambda_D t_{CE} \qquad \text{since } \lambda_D t_{CE} << 1$$

Hence, for a 1oo1 architecture, the average probability of failure on demand is

$$PFD_G = (\lambda_{DU} + \lambda_{DD}) t_{CE}$$

B.3.2.2.2 1oo2

This architecture consists of two channels connected in parallel, such that either channel can process the safety function. Thus there would have to be a dangerous failure in both channels before a safety function failed on demand. It is assumed that any diagnostic testing would only report the faults found and would not change any output states or change the output voting.

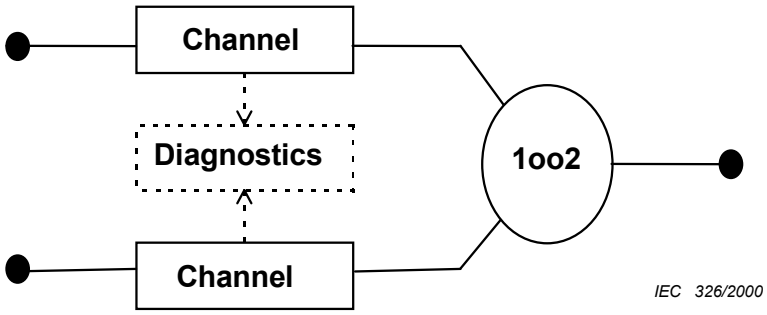


Figure B.6 – 1oo2 physical block diagram

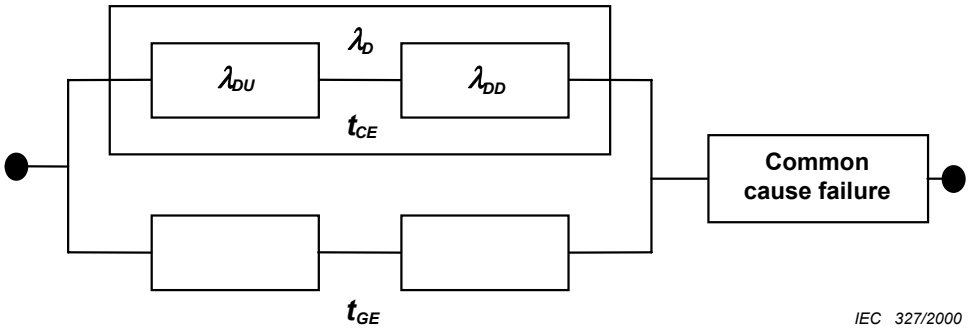


Figure B.7 – 1oo2 reliability block diagram

Figures B.6 and B.7 contain the relevant block diagrams. The value of t_{CE} is as given in B.3.2.2.1, but now it is necessary to also calculate the system equivalent down time t_{GE}, which is given by

$$t_{GE} = \frac{\lambda_{DU}}{\lambda_D}\left(\frac{T_1}{3} + MRT\right) + \frac{\lambda_{DD}}{\lambda_D} MTTR$$

The average probability of failure on demand for the architecture is

$$PFD_G = 2\left((1-\beta_D)\lambda_{DD} + (1-\beta)\lambda_{DU}\right)^2 t_{CE} t_{GE} + \beta_D \lambda_{DD} MTTR + \beta\lambda_{DU}\left(\frac{T_1}{2} + MRT\right)$$

B.3.2.2.3 2oo2

This architecture consists of two channels connected in parallel so that both channels need to demand the safety function before it can take place. It is assumed that any diagnostic testing would only report the faults found and would not change any output states or change the output voting.

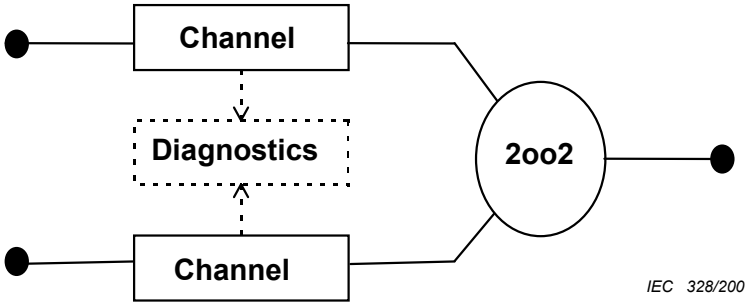


Figure B.8 – 2oo2 physical block diagram

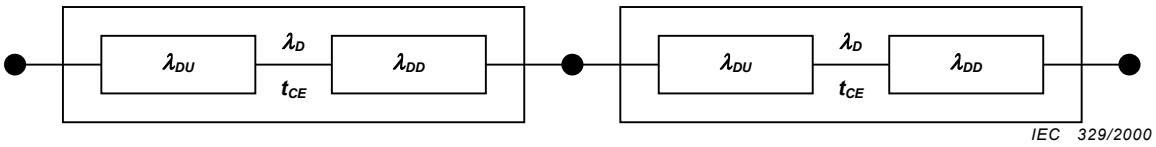


Figure B.9 – 2oo2 reliability block diagram

Figures B.8 and B.9 contain the relevant block diagrams. The value of t_{CE} is as given in B.3.2.2.1, and the average probability of failure on demand for the architecture is

$$PFD_G = 2\lambda_D t_{CE}$$

B.3.2.2.4 1oo2D

This architecture consists of two channels connected in parallel. During normal operation, both channels need to demand the safety function before it can take place. In addition, if the diagnostic tests in either channel detect a fault then the output voting is adapted so that the overall output state then follows that given by the other channel. If the diagnostic tests find faults in both channels or a discrepancy that cannot be allocated to either channel, then the output goes to the safe state. In order to detect a discrepancy between the channels, either channel can determine the state of the other channel via a means independent of the other channel. The channel comparison / switch over mechanism may not be 100 % efficient therefore K represents the efficiency of this inter-channel comparison / switch mechanism, i.e. the output may remain on the 2oo2 voting even with one channel detected as faulty.

NOTE The parameter K will need to be determined by an FMEA.

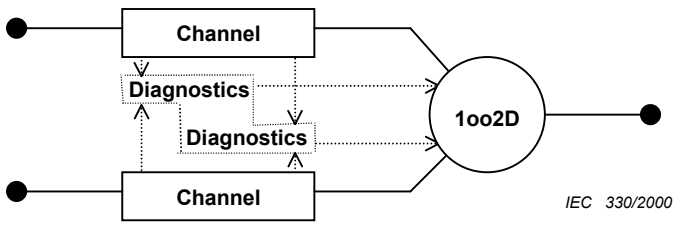


Figure B.10 – 1oo2D physical block diagram

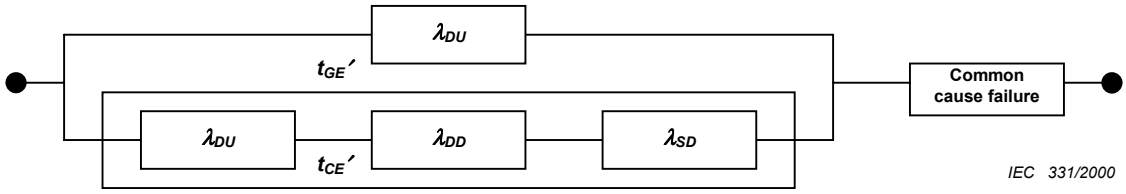


Figure B.11 – 1oo2D reliability block diagram

The detected safe failure rate for every channel is given by

$$\lambda_{SD} = \lambda_S DC$$

Figures B.10 and B.11 contain the relevant block diagrams. The values of the equivalent mean down times differ from those given for the other architectures in B.3.2.2 and hence are labelled t_{CE}´and t_{GE}´. Their values are given by

$$t_{CE}' = \frac{\lambda_{DU}\left(\frac{T_1}{2} + MRT\right) + (\lambda_{DD} + \lambda_{SD})MTTR}{\lambda_{DU} + (\lambda_{DD} + \lambda_{SD})}$$

$$t_{GE}' = \frac{T_1}{3} + MRT$$

The average probability of failure on demand for the architecture is

$$PFD_G = 2(1-\beta)\lambda_{DU}((1-\beta)\lambda_{DU} + (1-\beta_D)\lambda_{DD} + \lambda_{SD})t_{CE}'\,t_{GE}' + 2(1-K)\lambda_{DD}t_{CE}' + \beta\lambda_{DU}\left(\frac{T_1}{2} + MRT\right)$$

B.3.2.2.5 2oo3

This architecture consists of three channels connected in parallel with a majority voting arrangement for the output signals, such that the output state is not changed if only one channel gives a different result which disagrees with the other two channels.

It is assumed that any diagnostic testing would only report the faults found and would not change any output states or change the output voting.

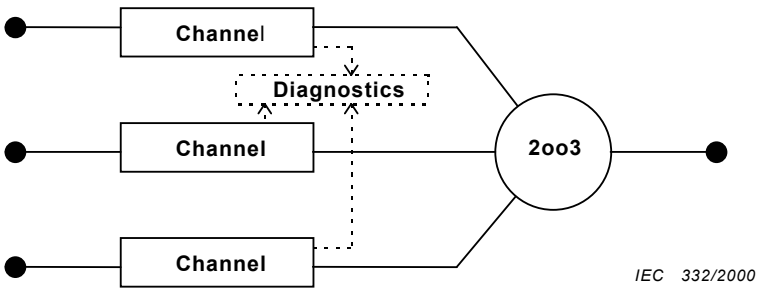


Figure B.12 – 2oo3 physical block diagram

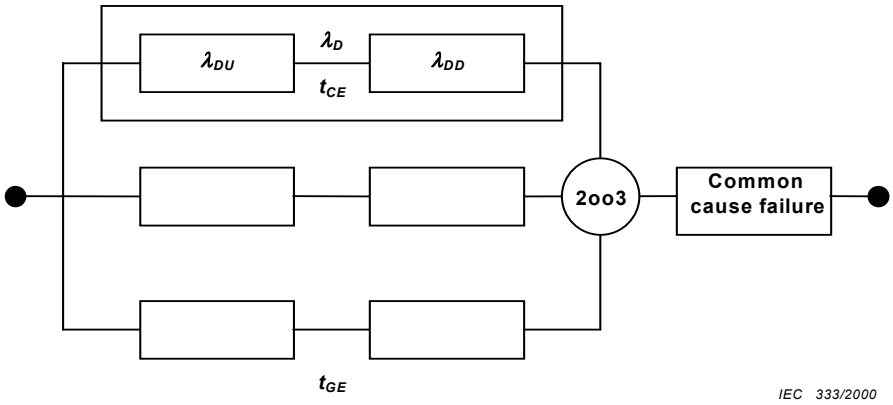


Figure B.13 – 2oo3 reliability block diagram

Figures B.12 and B.13 contain the relevant block diagrams. The value of t_{CE} is as given in B.3.2.2.1 and the value of t_{GE} is as given in B.3.2.2.2. The average probability of failure on demand for the architecture is

$$PFD_G = 6((1-\beta_D)\lambda_{DD} + (1-\beta)\lambda_{DU})^2 t_{CE} t_{GE} + \beta_D \lambda_{DD} MTTR + \beta\lambda_{DU}\left(\frac{T_1}{2} + MRT\right)$$

B.3.2.2.6 1oo3

This architecture consists of three channels connected in parallel with a voting arrangement for the output signals, such that the output state follows 1oo3 voting.

It is assumed that any diagnostic testing would only report the faults found and would not change any output states or change the output voting.

The reliability diagram will be the same as for the 2oo3 case but with voting 1oo3. The value of t_{CE} is as given in B.3.2.2.1 and the value of t_{GE} is as given in B.3.2.2.2. The average probability of failure on demand for the architecture is

$$PFD_G = 6((1-\beta_D)\lambda_{DD} + (1-\beta)\lambda_{DU})^3 t_{CE} t_{GE} t_{G2E} + \beta_D \lambda_{DD} MTTR + \beta\lambda_{DU}\left(\frac{T_1}{2} + MRT\right)$$

Where

IEC 332/2000

$$t_{G2E} = \frac{\lambda_{DU}}{\lambda_D}\left(\frac{T_1}{4} + MRT\right) + \frac{\lambda_{DD}}{\lambda_D} MTTR$$

B.3.2.3 Detailed tables for low demand mode of operation

Table B.2 – Average probability of failure on demand for a proof test interval of six months and a mean time to restoration of 8 h

Architecture	*DC*	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 %, β_D = 1 %	β = 10 %, β_D = 5 %	β = 20 %, β_D = 10 %	β = 2 %, β_D = 1 %	β = 10 %, β_D = 5 %	β = 20 %, β_D = 10 %	β = 2 %, β_D = 1 %	β = 10 %, β_D = 5 %	β = 20 %, β_D = 10 %
1oo1 (see Note 2)	**0 %**	1,1E-04			5,5E-04			1,1E-03		
	60 %	4,4E-05			2,2E-04			4,4E-04		
	90 %	1,1E-05			5,7E-05			1,1E-04		
	99 %	1,5E-06			7,5E-06			1,5E-05		
1oo2	**0 %**	2,2E-06	1,1E-05	2,2E-05	1,1E-05	5,5E-05	1,1E-04	2,4E-05	1,1E-04	2,2E-04
	60 %	8,8E-07	4,4E-06	8,8E-06	4,5E-06	2,2E-05	4,4E-05	9,1E-06	4,4E-05	8,8E-05
	90 %	2,2E-07	1,1E-06	2,2E-06	1,1E-06	5,6E-06	1,1E-05	2,3E-06	1,1E-05	2,2E-05
	99 %	2,6E-08	1,3E-07	2,6E-07	1,3E-07	6,5E-07	1,3E-06	2,6E-07	1,3E-06	2,6E-06
2oo2 (see Note 2)	**0 %**	2,2E-04			1,1E-03			2,2E-03		
	60 %	8,8E-05			4,4E-04			8,8E-04		
	90 %	2,3E-05			1,1E-04			2,3E-04		
	99 %	3,0E-06			1,5E-05			3,0E-05		
1oo2D (see Note 3)	**0 %**	2,2E-06	1,1E-05	2,2E-05	1,1E-05	5,5E-05	1,1E-04	2,4E-05	1,1E-04	2,2E-04
	60 %	1,4E-06	4,9E-06	9,3E-06	7,1E-06	2,5E-05	4,7E-05	1,4E-05	5,0E-05	9,3E-05
	90 %	4,3E-07	1,3E-06	2,4E-06	2,2E-06	6,6E-06	1,2E-05	4,3E-06	1,3E-05	2,4E-05
	99 %	6,0E-08	1,5E-07	2,6E-07	3,0E-07	7,4E-07	1,3E-06	6,0E-07	1,5E-06	2,6E-06
2oo3	**0 %**	2,2E-06	1,1E-05	2,2E-05	1,2E-05	5,6E-05	1,1E-04	2,7E-05	1,1E-04	2,2E-04
	60 %	8,9E-07	4,4E-06	8,8E-06	4,6E-06	2,2E-05	4,4E-05	9,6E-06	4,5E-05	8,9E-05
	90 %	2,2E-07	1,1E-06	2,2E-06	1,1E-06	5,6E-06	1,1E-05	2,3E-06	1,1E-05	2,2E-05
	99 %	2,6E-08	1,3E-07	2,6E-07	1,3E-07	6,5E-07	1,3E-06	2,6E-07	1,3E-06	2,6E-06
1oo3	**0 %**	2,2E-06	1,1E-05	2,2E-05	1,1E-05	5,5E-05	1,1E-04	2,2E-05	1,1E-04	2,2E-04
	60 %	8,8E-07	4,4E-06	8,8E-06	4,4E-06	2,2E-05	4,4E-05	8,8E-06	4,4E-05	8,8E-05
	90 %	2,2E-07	1,1E-06	2,2E-06	1,1E-06	5,6E-06	1,1E-05	2,2E-06	1,1E-05	2,2E-05
	99 %	2,6E-08	1,3E-07	2,6E-07	1,3E-07	6,5E-07	1,3E-06	2,6E-07	1,3E-06	2,6E-06

Architecture	*DC*	λ_D = 2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 %, β_D = 1 %	β = 10 %, β_D = 5 %	β = 20 %, β_D = 10 %	β = 2 %, β_D = 1 %	β = 10 %, β_D = 5 %	β = 20 %, β_D = 10 %	β = 2 %, β_D = 1 %	β = 10 %, β_D = 5 %	β = 20 %, β_D = 10 %
1oo1 (see Note 2)	**0 %**	5,5E-03			1,1E-02			5,5E-02		
	60 %	2,2E-03			4,4E-03			2,2E-02		
	90 %	5,7E-04			1,1E-03			5,7E-03		
	99 %	7,5E-05			1,5E-04			7,5E-04		
1oo2	**0 %**	1,5E-04	5,8E-04	1,1E-03	3,7E-04	1,2E-03	2,3E-03	5,0E-03	8,8E-03	1,4E-02
	60 %	5,0E-05	2,3E-04	4,5E-04	1,1E-04	4,6E-04	9,0E-04	1,1E-03	2,8E-03	4,9E-03
	90 %	1,2E-05	5,6E-05	1,1E-04	2,4E-05	1,1E-04	2,2E-04	1,5E-04	6,0E-04	1,2E-03
	99 %	1,3E-06	6,5E-06	1,3E-05	2,6E-06	1,3E-05	2,6E-05	1,4E-05	6,6E-05	1,3E-04
2oo2 (see Note 2)	**0 %**	1,1E-02			2,2E-02			>1E-01		
	60 %	4,4E-03			8,8E-03			4,4E-02		
	90 %	1,1E-03			2,3E-03			1,1E-02		
	99 %	1,5E-04			3,0E-04			1,5E-03		
1oo2D (see Note 3)	**0 %**	1,5E-04	5,8E-04	1,1E-03	3,8E-04	1,2E-03	2,3E-03	5,0E-03	9,0E-03	1,4E-02
	60 %	7,7E-05	2,5E-04	4,7E-04	1,7E-04	5,2E-04	9,5E-04	1,3E-03	3,0E-03	5,1E-03
	90 %	2,2E-05	6,6E-05	1,2E-04	4,5E-05	1,3E-04	2,4E-04	2,6E-04	6,9E-04	1,2E-03
	99 %	3,0E-06	7,4E-06	1,3E-05	6,0E-06	1,5E-05	2,6E-05	3,0E-05	7,4E-05	1,3E-04
2oo3	**0 %**	2,3E-04	6,5E-04	1,2E-03	6,8E-04	1,5E-03	2,5E-03	1,3E-02	1,5E-02	1,9E-02
	60 %	6,3E-05	2,4E-04	4,6E-04	1,6E-04	5,1E-04	9,4E-04	2,3E-03	3,9E-03	5,9E-03
	90 %	1,2E-05	5,7E-05	1,1E-04	2,7E-05	1,2E-04	2,3E-04	2,4E-04	6,8E-04	1,2E-03
	99 %	1,3E-06	6,5E-06	1,3E-05	2,7E-06	1,3E-05	2,6E-05	1,5E-05	6,7E-05	1,3E-04
1oo3	**0 %**	1,1E-04	5,5E-04	1,1E-03	2,2E-04	1,1E-03	2,2E-03	1,4E-03	5,7E-03	1,1E-02
	60 %	4,4E-05	2,2E-04	4,4E-04	8,8E-05	4,4E-04	8,8E-04	4,6E-04	2,2E-03	4,4E-03
	90 %	1,1E-05	5,6E-05	1,1E-04	2,2E-05	1,1E-04	2,2E-04	1,1E-04	5,6E-04	1,1E-03
	99 %	1,3E-06	6,5E-06	1,3E-05	2,6E-06	1,3E-05	2,6E-05	1,3E-05	6,5E-05	1,3E-04

NOTE 1 This table gives example values of PFD_G, calculated using the equations in B.3.2 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFD_G is equivalent to PFD_S, PFD_L or PFD_{FE} respectively (see B.3.2.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average probability of failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

Table B.3 – Average probability of failure on demand for a proof test interval of one year and mean time to restoration of 8 h

Architecture	*DC*	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see Note 2)	**0 %**	2,2E-04			1,1E-03			2,2E-03		
	60 %	8,8E-05			4,4E-04			8,8E-04		
	90 %	2,2E-05			1,1E-04			2,2E-04		
	99 %	2,6E-06			1,3E-05			2,6E-05		
1oo2	**0 %**	4,4E-06	2,2E-05	4,4E-05	2,3E-05	1,1E-04	2,2E-04	5,0E-05	2,2E-04	4,4E-04
	60 %	1,8E-06	8,8E-06	1,8E-05	9,0E-06	4,4E-05	8,8E-05	1,9E-05	8,9E-05	1,8E-04
	90 %	4,4E-07	2,2E-06	4,4E-06	2,2E-06	1,1E-05	2,2E-05	4,5E-06	2,2E-05	4,4E-05
	99 %	4,8E-08	2,4E-07	4,8E-07	2,4E-07	1,2E-06	2,4E-06	4,8E-07	2,4E-06	4,8E-06
2oo2 (see Note 2)	**0 %**	4,4E-04			2,2E-03			4,4E-03		
	60 %	1,8E-04			8,8E-04			1,8E-03		
	90 %	4,5E-05			2,2E-04			4,5E-04		
	99 %	5,2E-06			2,6E-05			5,2E-05		
1oo2D (see Note 3)	**0 %**	4,5E-06	2,2E-05	4,4E-05	2,4E-05	1,1E-04	2,2E-04	5,0E-05	2,2E-04	4,4E-04
	60 %	2,8E-06	9,8E-06	1,9E-05	1,4E-05	4,9E-05	9,3E-05	2,9E-05	9,9E-05	1,9E-04
	90 %	8,5E-07	2,6E-06	4,8E-06	4,3E-06	1,3E-05	2,4E-05	8,5E-06	2,6E-05	4,8E-05
	99 %	1,0E-07	2,8E-07	5,0E-07	5,2E-07	1,4E-06	2,5E-06	1,0E-06	2,8E-06	5,0E-06
2oo3	**0 %**	4,6E-06	2,2E-05	4,4E-05	2,7E-05	1,1E-04	2,2E-04	6,2E-05	2,4E-04	4,5E-04
	60 %	1,8E-06	8,8E-06	1,8E-05	9,5E-06	4,5E-05	8,8E-05	2,1E-05	9,1E-05	1,8E-04
	90 %	4,4E-07	2,2E-06	4,4E-06	2,3E-06	1,1E-05	2,2E-05	4,6E-06	2,2E-05	4,4E-05
	99 %	4,8E-08	2,4E-07	4,8E-07	2,4E-07	1,2E-06	2,4E-06	4,8E-07	2,4E-06	4,8E-06
1oo3	**0 %**	4,4E-06	2,2E-05	4,4E-05	2,2E-05	1,1E-04	2,2E-04	4,4E-05	2,2E-04	4,4E-04
	60 %	1,8E-06	8,8E-06	1,8E-05	8,8E-06	4,4E-05	8,8E-05	1,8E-05	8,8E-05	1,8E-04
	90 %	4,4E-07	2,2E-06	4,4E-06	2,2E-06	1,1E-05	2,2E-05	4,4E-06	2,2E-05	4,4E-05
	99 %	4,8E-08	2,4E-07	4,8E-07	2,4E-07	1,2E-06	2,4E-06	4,8E-07	2,4E-06	4,8E-06

Architecture	*DC*	λ_D = 2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see Note 2)	**0 %**	1,1E-02			2,2E-02			>1E-01		
	60 %	4,4E-03			8,8E-03			4,4E-02		
	90 %	1,1E-03			2,2E-03			1,1E-02		
	99 %	1,3E-04			2,6E-04			1,3E-03		
1oo2	**0 %**	3,7E-04	1,2E-03	2,3E-03	1,1E-03	2,7E-03	4,8E-03	1,8E-02	2,4E-02	3,2E-02
	60 %	1,1E-04	4,6E-04	9,0E-04	2,8E-04	9,7E-04	1,8E-03	3,4E-03	6,6E-03	1,1E-02
	90 %	2,4E-05	1,1E-04	2,2E-04	5,1E-05	2,3E-04	4,5E-04	3,8E-04	1,3E-03	2,3E-03
	99 %	2,4E-06	1,2E-05	2,4E-05	4,9E-06	2,4E-05	4,8E-05	2,6E-05	1,2E-04	2,4E-04
2oo2 (see Note 2)	**0 %**	2,2E-02			4,4E-02			>1E-01		
	60 %	8,8E-03			1,8E-02			8,8E-02		
	90 %	2,2E-03			4,5E-03			2,2E-02		
	99 %	2,6E-04			5,2E-04			2,6E-03		
1oo2D (see Note 3)	**0 %**	3,8E-04	1,2E-03	2,3E-03	1,1E-03	2,7E-03	4,9E-03	1,8E-02	2,5E-02	3,4E-02
	60 %	1,7E-04	5,1E-04	9,5E-04	3,8E-04	1,1E-03	1,9E-03	3,9E-03	7,1E-03	1,1E-02
	90 %	4,4E-05	1,3E-04	2,4E-04	9,1E-05	2,7E-04	4,8E-04	5,8E-04	1,4E-03	2,5E-03
	99 %	5,2E-06	1,4E-05	2,5E-05	1,0E-05	2,8E-05	5,0E-05	5,4E-05	1,4E-04	2,5E-04
2oo3	**0 %**	6,8E-04	1,5E-03	2,5E-03	2,3E-03	3,8E-03	5,6E-03	4,8E-02	5,0E-02	5,3E-02
	60 %	1,6E-04	5,1E-04	9,4E-04	4,8E-04	1,1E-03	2,0E-03	8,4E-03	1,1E-02	1,5E-02
	90 %	2,7E-05	1,2E-04	2,3E-04	6,4E-05	2,4E-04	4,6E-04	7,1E-04	1,6E-03	2,6E-03
	99 %	2,5E-06	1,2E-05	2,4E-05	5,1E-06	2,4E-05	4,8E-05	3,1E-05	1,3E-04	2,5E-04
1oo3	**0 %**	2,2E-04	1,1E-03	2,2E-03	4,6E-04	2,2E-03	4,4E-03	4,7E-03	1,3E-02	2,3E-02
	60 %	8,8E-05	4,4E-04	8,8E-04	1,8E-04	8,8E-04	1,8E-03	1,0E-03	4,5E-03	8,9E-03
	90 %	2,2E-05	1,1E-04	2,2E-04	4,4E-05	2,2E-04	4,4E-04	2,2E-04	1,1E-03	2,2E-03
	99 %	2,4E-06	1,2E-05	2,4E-05	4,8E-06	2,4E-05	4,8E-05	2,4E-05	1,2E-04	2,4E-04

NOTE 1 This table gives example values of PFD_G, calculated using the equations in B.3.2 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFD_G is equivalent to PFD_S, PFD_L or PFD_{FE} respectively (see B.3.2.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average probability of failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

Table B.4 – Average probability of failure on demand for a proof test interval of two years and a mean time to restoration of 8 h

Architecture	*DC*	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see Note 2)	**0 %**	4,4E-04			2,2E-03			4,4E-03		
	60 %	1,8E-04			8,8E-04			1,8E-03		
	90 %	4,4E-05			2,2E-04			4,4E-04		
	99 %	4,8E-06			2,4E-05			4,8E-05		
1oo2	**0 %**	9,0E-06	4,4E-05	8,8E-05	5,0E-05	2,2E-04	4,4E-04	1,1E-04	4,6E-04	8,9E-04
	60 %	3,5E-06	1,8E-05	3,5E-05	1,9E-05	8,9E-05	1,8E-04	3,9E-05	1,8E-04	3,5E-04
	90 %	8,8E-07	4,4E-06	8,8E-06	4,5E-06	2,2E-05	4,4E-05	9,1E-06	4,4E-05	8,8E-05
	99 %	9,2E-08	4,6E-07	9,2E-07	4,6E-07	2,3E-06	4,6E-06	9,2E-07	4,6E-06	9,2E-06
2oo2 (see Note 2)	**0 %**	8,8E-04			4,4E-03			8,8E-03		
	60 %	3,5E-04			1,8E-03			3,5E-03		
	90 %	8,8E-05			4,4E-04			8,8E-04		
	99 %	9,6E-06			4,8E-05			9,6E-05		
1oo2D (see Note 3)	**0 %**	9,0E-06	4,4E-05	8,8E-05	5,0E-05	2,2E-04	4,4E-04	1,1E-04	4,6E-04	9,0E-04
	60 %	5,7E-06	2,0E-05	3,7E-05	2,9E-05	9,9E-05	1,9E-04	6,0E-05	2,0E-04	3,7E-04
	90 %	1,7E-06	5,2E-06	9,6E-06	8,5E-06	2,6E-05	4,8E-05	1,7E-05	5,2E-05	9,6E-05
	99 %	1,9E-07	5,4E-07	9,8E-07	9,5E-07	2,7E-06	4,9E-06	1,9E-06	5,4E-06	9,8E-06
2oo3	**0 %**	9,5E-06	4,4E-05	8,8E-05	6,2E-05	2,3E-04	4,5E-04	1,6E-04	5,0E-04	9,3E-04
	60 %	3,6E-06	1,8E-05	3,5E-05	2,1E-05	9,0E-05	1,8E-04	4,7E-05	1,9E-04	3,6E-04
	90 %	8,9E-07	4,4E-06	8,8E-06	4,6E-06	2,2E-05	4,4E-05	9,6E-06	4,5E-05	8,9E-05
	99 %	9,2E-08	4,6E-07	9,2E-07	4,6E-07	2,3E-06	4,6E-06	9,3E-07	4,6E-06	9,2E-06
1oo3	**0 %**	8,8E-06	4,4E-05	8,8E-05	4,4E-05	2,2E-04	4,4E-04	8,8E-05	4,4E-04	8,8E-04
	60 %	3,5E-06	1,8E-05	3,5E-05	1,8E-05	8,8E-05	1,8E-04	3,5E-05	1,8E-04	3,5E-04
	90 %	8,8E-07	4,4E-06	8,8E-06	4,4E-06	2,2E-05	4,4E-05	8,8E-06	4,4E-05	8,8E-05
	99 %	9,2E-08	4,6E-07	9,2E-07	4,6E-07	2,3E-06	4,6E-06	9,2E-07	4,6E-06	9,2E-06

Architecture	*DC*	λ_D =2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see Note 2)	**0 %**	2,2E-02			4,4E-02			>1E-01		
	60 %	8,8E-03			1,8E-02			8,8E-02		
	90 %	2,2E-03			4,4E-03			2,2E-02		
	99 %	2,4E-04			4,8E-04			2,4E-03		
1oo2	**0 %**	1,1E-03	2,7E-03	4,8E-03	3,3E-03	6,5E-03	1,0E-02	6,6E-02	7,4E-02	8,5E-02
	60 %	2,8E-04	9,7E-04	1,8E-03	7,5E-04	2,1E-03	3,8E-03	1,2E-02	1,8E-02	2,5E-02
	90 %	5,0E-05	2,3E-04	4,5E-04	1,1E-04	4,6E-04	9,0E-04	1,1E-03	2,8E-03	4,9E-03
	99 %	4,7E-06	2,3E-05	4,6E-05	9,5E-06	4,6E-05	9,2E-05	5,4E-05	2,4E-04	4,6E-04
2oo2 (see Note 2)	**0 %**	4,4E-02			8,8E-02			>1E-01		
	60 %	1,8E-02			3,5E-02			>1E-01		
	90 %	4,4E-03			8,8E-03			4,4E-02		
	99 %	4,8E-04			9,6E-04			4,8E-03		
1oo2D (see Note 3)	**0 %**	1,1E-03	2,7E-03	4,8E-03	3,4E-03	6,6E-03	1,1E-02	6,7E-02	7,7E-02	9,0E-02
	60 %	3,8E-04	1,1E-03	1,9E-03	9,6E-04	2,3E-03	4,0E-03	1,3E-02	1,9E-02	2,6E-02
	90 %	9,0E-05	2,6E-04	4,8E-04	1,9E-04	5,4E-04	9,8E-04	1,5E-03	3,2E-03	5,3E-03
	99 %	9,6E-06	2,7E-05	4,9E-05	1,9E-05	5,4E-05	9,8E-05	1,0E-04	2,8E-04	5,0E-04
2oo3	**0 %**	2,3E-03	3,7E-03	5,6E-03	8,3E-03	1,1E-02	1,4E-02	1,9E-01	1,8E-01	1,7E-01
	60 %	4,8E-04	1,1E-03	2,0E-03	1,6E-03	2,8E-03	4,4E-03	3,2E-02	3,5E-02	4,0E-02
	90 %	6,3E-05	2,4E-04	4,6E-04	1,6E-04	5,1E-04	9,4E-04	2,4E-03	4,0E-03	6,0E-03
	99 %	4,8E-06	2,3E-05	4,6E-05	1,0E-05	4,7E-05	9,2E-05	6,9E-05	2,5E-04	4,8E-04
1oo3	**0 %**	4,6E-04	2,2E-03	4,4E-03	1,0E-03	4,5E-03	8,9E-03	2,4E-02	3,7E-02	5,5E-02
	60 %	1,8E-04	8,8E-04	1,8E-03	3,6E-04	1,8E-03	3,5E-03	3,1E-03	9,9E-03	1,8E-02
	90 %	4,4E-05	2,2E-04	4,4E-04	8,8E-05	4,4E-04	8,8E-04	4,6E-04	2,2E-03	4,4E-03
	99 %	4,6E-06	2,3E-05	4,6E-05	9,2E-06	4,6E-05	9,2E-05	4,6E-05	2,3E-04	4,6E-04

NOTE 1 This table gives example values of PFD_G, calculated using the equations in B.3.2 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFD_G is equivalent to PFD_S, PFD_L or PFD_{FE} respectively (see B.3.2.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average probability of failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

Table B.5 – Average probability of failure on demand for a proof test interval of ten years and a mean time to restoration of 8 h

Architecture	*DC*	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see Note 2)	**0 %**	2,2E-03			1,1E-02			2,2E-02		
	60 %	8,8E-04			4,4E-03			8,8E-03		
	90 %	2,2E-04			1,1E-03			2,2E-03		
	99 %	2,2E-05			1,1E-04			2,2E-04		
1oo2	**0 %**	5,0E-05	2,2E-04	4,4E-04	3,7E-04	1,2E-03	2,3E-03	1,1E-03	2,7E-03	4,8E-03
	60 %	1,9E-05	8,9E-05	1,8E-04	1,1E-04	4,6E-04	9,0E-04	2,7E-04	9,6E-04	1,8E-03
	90 %	4,4E-06	2,2E-05	4,4E-05	2,3E-05	1,1E-04	2,2E-04	5,0E-05	2,2E-04	4,4E-04
	99 %	4,4E-07	2,2E-06	4,4E-06	2,2E-06	1,1E-05	2,2E-05	4,5E-06	2,2E-05	4,4E-05
2oo2 (see Note 2)	**0 %**	4,4E-03			2,2E-02			4,4E-02		
	60 %	1,8E-03			8,8E-03			1,8E-02		
	90 %	4,4E-04			2,2E-03			4,4E-03		
	99 %	4,5E-05			2,2E-04			4,5E-04		
1oo2D (see Note 3)	**0 %**	5,0E-05	2,2E-04	4,4E-04	3,7E-04	1,2E-03	2,3E-03	1,1E-03	2,7E-03	4,8E-03
	60 %	2,9E-05	9,9E-05	1,9E-04	1,7E-04	5,1E-04	9,5E-04	3,8E-04	1,1E-03	1,9E-03
	90 %	8,4E-06	2,6E-05	4,8E-05	4,3E-05	1,3E-04	2,4E-04	9,0E-05	2,6E-04	4,8E-04
	99 %	8,9E-07	2,6E-06	4,8E-06	4,5E-06	1,3E-05	2,4E-05	8,9E-06	2,6E-05	4,8E-05
2oo3	**0 %**	6,2E-05	2,3E-04	4,5E-04	6,8E-04	1,5E-03	2,5E-03	2,3E-03	3,7E-03	5,6E-03
	60 %	2,1E-05	9,0E-05	1,8E-04	1,6E-04	5,0E-04	9,3E-04	4,7E-04	1,1E-03	2,0E-03
	90 %	4,6E-06	2,2E-05	4,4E-05	2,7E-05	1,1E-04	2,2E-04	6,3E-05	2,4E-04	4,5E-04
	99 %	4,4E-07	2,2E-06	4,4E-06	2,3E-06	1,1E-05	2,2E-05	4,6E-06	2,2E-05	4,4E-05
1oo3	**0 %**	4,4E-05	2,2E-04	4,4E-04	2,2E-04	1,1E-03	2,2E-03	4,6E-04	2,2E-03	4,4E-03
	60 %	1,8E-05	8,8E-05	1,8E-04	8,8E-05	4,4E-04	8,8E-04	1,8E-04	8,8E-04	1,8E-03
	90 %	4,4E-06	2,2E-05	4,4E-05	2,2E-05	1,1E-04	2,2E-04	4,4E-05	2,2E-04	4,4E-04
	99 %	4,4E-07	2,2E-06	4,4E-06	2,2E-06	1,1E-05	2,2E-05	4,4E-06	2,2E-05	4,4E-05

Architecture	*DC*	λ_D =2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see Note 2)	**0 %**	>1E-01			>1E-01			>1E-01		
	60 %	4,4E-02			8,8E-02			>1E-01		
	90 %	1,1E-02			2,2E-02			>1E-01		
	99 %	1,1E-03			2,2E-03			1,1E-02		
1oo2	**0 %**	1,8E-02	2,4E-02	3,2E-02	6,6E-02	7,4E-02	8,5E-02	>1E-01	>1E-01	>1E-01
	60 %	3,4E-03	6,6E-03	1,1E-02	1,2E-02	1,8E-02	2,5E-02	>1E-01	>1E-01	>1E-01
	90 %	3,8E-04	1,2E-03	2,3E-03	1,1E-03	2,8E-03	4,9E-03	1,8E-02	2,5E-02	3,5E-02
	99 %	2,4E-05	1,1E-04	2,2E-04	5,1E-05	2,3E-04	4,5E-04	3,8E-04	1,3E-03	2,3E-03
2oo2 (see Note 2)	**0 %**	>1E-01			>1E-01			>1E-01		
	60 %	8,8E-02			>1E-01			>1E-01		
	90 %	2,2E-02			4,4E-02			>1E-01		
	99 %	2,2E-03			4,5E-03			2,2E-02		
1oo2D (see Note 3)	**0 %**	1,8E-02	2,5E-02	3,3E-02	6,6E-02	7,7E-02	9,0E-02	1,6E+00	1,5E+00	1,4E+00
	60 %	3,9E-03	7,1E-03	1,1E-02	1,3E-02	1,9E-02	2,6E-02	2,6E-01	2,7E-01	2,8E-01
	90 %	5,7E-04	1,4E-03	2,5E-03	1,5E-03	3,1E-03	5,2E-03	2,0E-02	2,7E-02	3,5E-02
	99 %	4,6E-05	1,3E-04	2,4E-04	9,5E-05	2,7E-04	4,9E-04	6,0E-04	1,5E-03	2,5E-03
2oo3	**0 %**	4,8E-02	5,0E-02	5,3E-02	1,9E-01	1,8E-01	1,7E-01	4,6E+00	4,0E+00	3,3E+00
	60 %	8,3E-03	1,1E-02	1,4E-02	3,2E-02	3,5E-02	4,0E-02	7,6E-01	7,1E-01	6,6E-01
	90 %	6,9E-04	1,5E-03	2,6E-03	2,3E-03	3,9E-03	5,9E-03	4,9E-02	5,4E-02	6,0E-02
	99 %	2,7E-05	1,2E-04	2,3E-04	6,4E-05	2,4E-04	4,6E-04	7,1E-04	1,6E-03	2,6E-03
1oo3	**0 %**	4,7E-03	1,3E-02	2,3E-02	2,4E-02	3,7E-02	5,5E-02	2,5E+00	2,0E+00	1,6E+00
	60 %	1,0E-03	4,5E-03	8,9E-03	3,0E-03	9,8E-03	1,8E-02	1,7E-01	1,8E-01	1,9E-01
	90 %	2,2E-04	1,1E-03	2,2E-03	4,6E-04	2,2E-03	4,4E-03	4,8E-03	1,3E-02	2,4E-02
	99 %	2,2E-05	1,1E-04	2,2E-04	4,4E-05	2,2E-04	4,4E-04	2,2E-04	1,1E-03	2,2E-03

NOTE 1 This table gives example values of PFD_G, calculated using the equations in B.3.2 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFD_G is equivalent to PFD_S, PFD_L or PFD_{FE} respectively (see B.3.2.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average probability of failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

B.3.2.4 Example for low demand mode of operation

Consider a safety function requiring a SIL 2 system. Suppose that the initial assessment for the system architecture, based on previous practice, is for one group of three analogue pressure sensors, voting 2oo3. The logic subsystem is a redundant 1oo2D configured PE system driving a single shut-down valve plus a single vent valve. Both the shut-down and vent valves need to operate in order to achieve the safety function. The architecture is shown in Figure B.14. For the initial assessment, a proof test period of one year is assumed.

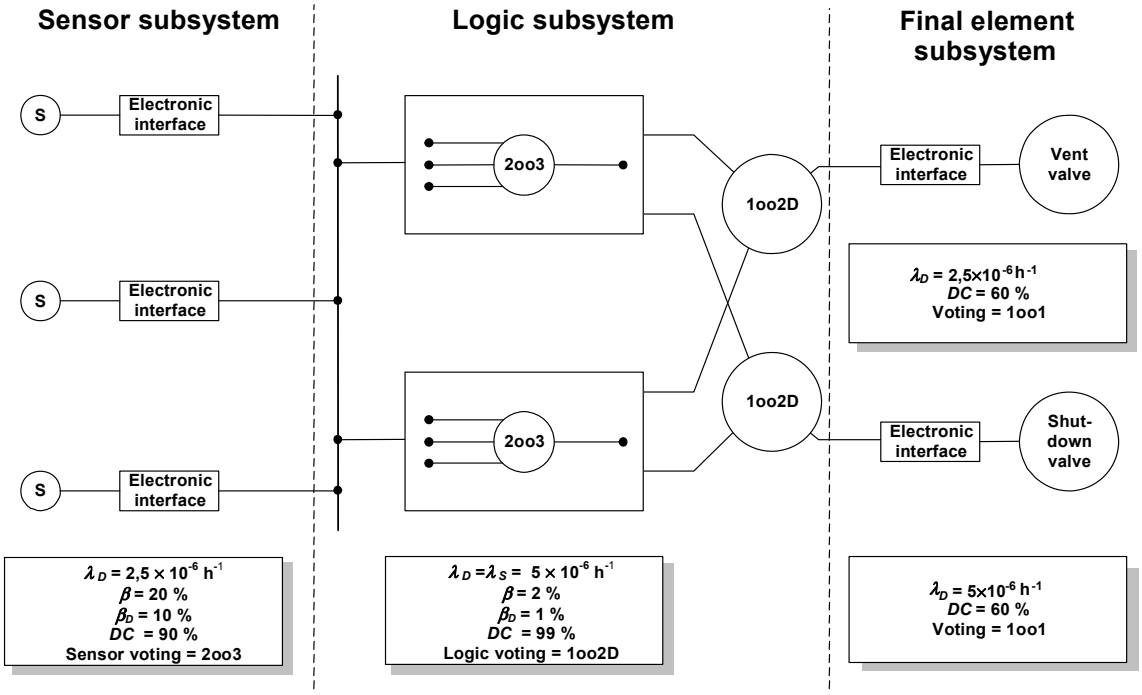


Figure B.14 – Architecture of an example for low demand mode of operation

Table B.6 – Average probability of failure on demand for the sensor subsystem in the example for low demand mode of operation (one year proof test interval and 8 h *MTTR*)

Architecture	*DC*	λ_D = 2,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
2oo3	0 %	6,8E-04	1,5E-03	2,5E-03
	60 %	1,6E-04	5,1E-04	9,4E-04
	90 %	2,7E-05	1,2E-04	2,3E-04
	99 %	2,5E-06	1,2E-05	2,4E-05
NOTE This table is abstracted from Table B.3.				

Table B.7 – Average probability of failure on demand for the logic subsystem in the example for low demand mode of operation (one year proof test interval and 8 h *MTTR*)

Architecture	*DC*	λ_D = 0,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo2D	0 %	1,1E-03	2,7E-03	4,8E-03
	60 %	2,0E-04	9,0E-04	1,8E-03
	90 %	4,5E-05	2,2E-04	4,4E-04
	99 %	4,8E-06	2,4E-05	4,8E-05
NOTE This table is abstracted from Table B.3.				

Table B.8 – Average probability of failure on demand for the final element subsystem in the example for low demand mode of operation (one year proof test interval and 8 h *MTTR*)

Architecture	*DC*	λ_D = 2,5E-06	λ_D = 0,5E-05
1oo1	0 %	1,1E-02	2,2E-02
	60 %	4,4E-03	8,8E-03
	90 %	1,1E-03	2,2E-03
	99 %	1,3E-04	2,6E-04
NOTE This table is abstracted from Table B.3.			

From Tables B.6 to B.8 the following values are derived.

For the sensor subsystem,

$PFD_S = 2{,}3 \times 10^{-4}$

For the logic subsystem,

$PFD_L = 4{,}8 \times 10^{-6}$

For the final element subsystem,

$PFD_{FE} = 4{,}4 \times 10^{-3} + 8{,}8 \times 10^{-3}$

$= 1{,}3 \times 10^{-2}$

Therefore, for the safety function,

$PFD_{SYS} = 2{,}3 \times 10^{-4} + 4{,}8 \times 10^{-6} + 1{,}3 \times 10^{-2}$

$= 1{,}3 \times 10^{-2}$

$\equiv$ **safety integrity level 1**

To improve the system to meet safety integrity level 2, one of the following could be done:

a) change the proof test interval to six months

$PFD_S = 1{,}1 \times 10^{-4}$

$PFD_L = 2{,}6 \times 10^{-6}$

$PFD_{FE} = 2{,}2 \times 10^{-3} + 4{,}4 \times 10^{-3}$

$= 6{,}6 \times 10^{-3}$

$PFD_{SYS} = 6{,}7 \times 10^{-3}$

≡ **safety integrity level 2**

b) change the 1oo1 shutdown valve (which is the output device with the lower reliability) to 1oo2 (assuming β = 10 % and β_D = 5 %)

PFD_S = $2{,}3 \times 10^{-4}$

PFD_L = $4{,}8 \times 10^{-6}$

PFD_{FE} = $4{,}4 \times 10^{-3} + 9{,}7 \times 10^{-4}$

= $5{,}4 \times 10^{-3}$

PFD_{SYS} = $5{,}6 \times 10^{-3}$

≡ **safety integrity level 2**

B.3.2.5 Effects of a non-perfect proof test

Faults in the safety system that are not detected by either diagnostic tests or proof tests may be found by other methods arising from events such as a hazardous event requiring operation of the safety function or during an overhaul of the equipment. If the faults are not detected by such methods it should be assumed that the faults will remain for the life of the equipment. Consider a normal proof test period of T_1 where the fraction of faults detected when a proof test is performed is designated as PTC (proof test coverage) and the fraction of the faults not detected when a proof test is performed is designated as (1-PCT). These latter faults which are not detected at the proof test will only be revealed when a demand is made on the safety-related system at demand period T_2. Therefore, the proof test period (T_1) and the demand period (T_2) govern the effective down time..

An example of this is given below for a 1oo2 architecture. T_2 is the time between demands on the system:

$$t_{CE} = \frac{\lambda_{DU}(PTC)}{\lambda_D}\left(\frac{T_1}{2} + MRT\right) + \frac{\lambda_{DU}(1-PTC)}{\lambda_D}\left(\frac{T_2}{2} + MRT\right) + \frac{\lambda_{DD}}{\lambda_D}MTTR$$

$$t_{GE} = \frac{\lambda_{DU}(PTC)}{\lambda_D}\left(\frac{T_1}{3} + MRT\right) + \frac{\lambda_{DU}(1-PTC)}{\lambda_D}\left(\frac{T_2}{3} + MRT\right) + \frac{\lambda_{DD}}{\lambda_D}MTTR$$

$$PFD_G = 2\left((1-\beta_D)\lambda_{DD} + (1-\beta)\lambda_{DU}\right)^2 t_{CE} t_{GE} + \beta_D \lambda_{DD} MTTR + \beta\lambda_{DU}(PTC)\left(\frac{T_1}{2} + MRT\right) +$$

$$\beta\lambda_{DU}(1-PTC)\left(\frac{T_2}{2} + MRT\right)$$

Table B.9 below gives the numeric results of a 1oo2 system with a 100 % one-year proof test (T_1 = 1 year) compared against a 90 % proof test where the demand period T_2 is assumed to be 10 years. This example has been calculated assuming a failure rate of $0{,}5 \times 10^{-5}$ per hour, a β value of 10 % and a β_D value of 5 %.

Table B.9 – Example for a non-perfect proof test

Architecture	*DC*	λ_D = 0,5E-05	
		100 % proof test	90 % proof test
		β = 10 % β_D = 5 %	β = 10 % β_D = 5 %
1oo2	0 %	2,7E-03	6,0E-03
	60 %	9,7E-04	2,0E-03
	90 %	2,3E-04	4,4E-04
	99 %	2,4E-05	4,4E-05

B.3.3 Average frequency of dangerous failure (for high demand or continuous mode of operation)

B.3.3.1 Procedure for calculations

The method for calculating the probability of failure of a safety function for an E/E/PE safety-related system operating in high demand or continuous mode of operation is identical with that for calculating for a low demand mode of operation (see B.2.1), except that average probability of failure on demand (PFD_{SYS}) is replaced with average frequency of dangerous failure (PFH_{SYS}).

The overall probability of a dangerous failure of a safety function for the E/E/PE safety-related system, PFH_{SYS}, is determined by calculating the dangerous failure rates for all the sub-systems which together provide the safety function and adding together these individual values. Since in this annex the probabilities are small, this can be expressed by the following:

$$PFH_{SYS} = PFH_S + PFH_L + PFH_{FE}$$

where

PFH_{SYS} is the average frequency of dangerous failure of a safety function for the E/E/PE safety-related system;

PFH_S is the average frequency of dangerous failure for the sensor subsystem;

PFH_L is the average frequency of dangerous failure for the logic subsystem; and

PFH_{FE} is the average frequency of dangerous failure for the final element subsystem.

B.3.3.2 Architectures for high demand or continuous mode of operation

NOTE 1 This subclause should be read sequentially, since equations which are valid for several architectures are only stated where they are first used. See also B.3.2.2.

NOTE 2 The calculations are based on the assumptions listed in B.3.1.

B.3.3.2.1 1oo1

Figures B.4 and B.5 show the relevant block diagrams.

$$\lambda_D = \lambda_{DU} + \lambda_{DD}$$

$$t_{CE} = \frac{\lambda_{DU}}{\lambda_D}\left(\frac{T_1}{2} + MRT\right) + \frac{\lambda_{DD}}{\lambda_D} MTTR$$

$$\lambda_{DU} = \lambda_D(1 - DC); \qquad \lambda_{DD} = \lambda_D DC$$

If it is assumed that the safety system puts the EUC into a safe state on detection of any failure, for a 1oo1 architecture the following is obtained

$$PFH_G = \lambda_{DU}$$

B.3.3.2.2 1oo2

Figures B.6 and B.7 show the relevant block diagrams. The value of t_{CE} is as given in B.3.3.2.1. If it is assumed that the safety system puts the EUC into a safe state once there is detection of a failure in both channels and taking a conservative approach, the following is obtained

$$PFH_G = 2((1 - \beta_D)\lambda_{DD} + (1 - \beta)\lambda_{DU})(1 - \beta)\lambda_{DU} t_{CE} + \beta\lambda_{DU}$$

B.3.3.2.3 2oo2

Figures B.8 and B.9 show the relevant block diagrams. If it is assumed that each channel is put into a safe state on detection of any fault, for a 2oo2 architecture, the following is obtained

$$PFH_G = 2\lambda_{DU}$$

B.3.3.2.4 1oo2D

Figures B.10 and B.11 show the relevant block diagrams.

$$\lambda_{SD} = \frac{\lambda}{2} DC$$

$$t_{CE}' = \frac{\lambda_{DU}\left(\frac{T_1}{2} + MRT\right) + (\lambda_{DD} + \lambda_{SD})MTTR}{\lambda_{DU} + \lambda_{DD} + \lambda_{SD}}$$

$$PFH_G = 2(1-\beta)\lambda_{DU}((1-\beta)\lambda_{DU} + (1-\beta_D)\lambda_{DD} + \lambda_{SD})t_{CE}' + 2(1-K)\lambda_{DD} + \beta\lambda_{DU}$$

B.3.3.2.5 2oo3

Figures B.12 and B.13 show the relevant block diagrams. The value of t_{CE} is as given in B.3.3.2.1. If it is assumed that the safety system puts the EUC into a safe state once there is detection of a failure in any two channels and taking a conservative approach, the following is obtained

$$PFH_G = 6((1-\beta_D)\lambda_{DD} + (1-\beta)\lambda_{DU})(1-\beta)\lambda_{DU}t_{CE} + \beta\lambda_{DU}$$

B.3.3.2.6 1oo3

Figures B.12 and B.13 show the relevant block diagrams. The value of t_{CE} and t_{GE} is as given in B.3.3.2.1. and B.3.2.2.2. If it is assumed that the safety system puts the EUC into a safe state once there is detection of a failure in the tree channels and taking a conservative approach, the following is obtained

$$PFH_G = 6((1-\beta_D)\lambda_{DD} + (1-\beta)\lambda_{DU})^2(1-\beta)\lambda_{DU}t_{CE}t_{GE} + \beta\lambda_{DU}$$

B.3.3.3 Detailed tables for high demand or continuous mode of operation

Table B.10 – Average frequency of a dangerous failure (in high demand or continuous mode of operation) for a proof test interval of one month and a mean time to restoration of 8 h

Architecture	DC	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	5.0E-08			2.5E-07			5.0E-07		
	60 %	2.0E-08			1.0E-07			2.0E-07		
	90 %	5.0E-09			2.5E-08			5.0E-08		
	99 %	5.0E-10			2.5E-09			5.0E-09		
1oo2	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.0E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
2oo2 (see note 2)	**0 %**	1.0E-07			5.0E-07			1.0E-06		
	60 %	4.0E-08			2.0E-07			4.0E-07		
	90 %	1.0E-08			5.0E-08			1.0E-07		
	99 %	1.0E-09			5.0E-09			1.0E-08		
1oo2D (see note 3)	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07
	60 %	1.6E-09	3.2E-09	5.2E-09	8.0E-09	1.6E-08	2.6E-08	1.6E-08	3.2E-08	5.2E-08
	90 %	1.9E-09	2.3E-09	2.8E-09	9.5E-09	1.2E-08	1.4E-08	1.9E-08	2.3E-08	2.8E-08
	99 %	2.0E-09	2.0E-09	2.1E-09	1.0E-08	1.0E-08	1.0E-08	2.0E-08	2.0E-08	2.1E-08
2oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.1E-09	2.5E-08	5.0E-08	1.1E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.1E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
1oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.0E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09

Architecture	DC	λ_D =2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	2.5E-06			5.0E-06			2.5E-05		
	60 %	1.0E-06			2.0E-06			1.0E-05		
	90 %	2.5E-07			5.0E-07			2.5E-06		
	99 %	2.5E-08			5.0E-08			2.5E-07		
1oo2	**0 %**	5.4E-08	2.5E-07	5.0E-07	1.2E-07	5.2E-07	1.0E-06	9.5E-07	2.9E-06	5.3E-06
	60 %	2.1E-08	1.0E-07	2.0E-07	4.3E-08	2.0E-07	4.0E-07	2.7E-07	1.1E-06	2.1E-06
	90 %	5.1E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07	5.5E-08	2.5E-07	5.0E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.1E-09	2.5E-08	5.0E-08
2oo2 (see note 2)	**0 %**	5.0E-06			1.0E-05			5.0E-05		
	60 %	2.0E-06			4.0E-06			2.0E-05		
	90 %	5.0E-07			1.0E-06			5.0E-06		
	99 %	5.0E-08			1.0E-07			5.0E-07		
1oo2D (see note 3)	**0 %**	5.4E-08	2.5E-07	5.0E-07	1.2E-07	5.2E-07	1.0E-06	9.5E-07	2.9E-06	5.3E-06
	60 %	8.1E-08	1.6E-07	2.6E-07	1.6E-07	3.2E-07	5.2E-07	8.7E-07	1.7E-06	2.7E-06
	90 %	9.5E-08	1.2E-07	1.4E-07	1.9E-07	2.3E-07	2.8E-07	9.6E-07	1.2E-06	1.4E-06
	99 %	1.0E-07	1.0E-07	1.0E-07	2.0E-07	2.0E-07	2.1E-07	1.0E-06	1.0E-06	1.0E-06
2oo3	**0 %**	6.3E-08	2.6E-07	5.1E-07	1.5E-07	5.5E-07	1.0E-06	1.8E-06	3.6E-06	5.9E-06
	60 %	2.2E-08	1.0E-07	2.0E-07	4.9E-08	2.1E-07	4.1E-07	4.2E-07	1.2E-06	2.2E-06
	90 %	5.2E-09	2.5E-08	5.0E-08	1.1E-08	5.1E-08	1.0E-07	6.6E-08	2.6E-07	5.1E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.4E-09	2.5E-08	5.0E-08
1oo3	**0 %**	5.0E-08	2.5E-07	5.0E-07	1.0E-07	5.0E-07	1.0E-06	5.1E-07	2.5E-06	5.0E-06
	60 %	2.0E-08	1.0E-07	2.0E-07	4.0E-08	2.0E-07	4.0E-07	2.0E-07	1.0E-06	2.0E-06
	90 %	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07	5.0E-08	2.5E-07	5.0E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08

NOTE 1 This table gives example values of PFH_G, calculated using the equations in B.3.3 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFH_G is equivalent to PFH_S, PFH_L or PFH_{FE} respectively (see B.3.3.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average frequency of a dangerous failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

Table B.11 – Average frequency of a dangerous failure (in high demand or continuous mode of operation) for a proof test interval of three month and a mean time to restoration of 8 h

Architecture	DC	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	5.0E-08			2.5E-07			5.0E-07		
	60 %	2.0E-08			1.0E-07			2.0E-07		
	90 %	5.0E-09			2.5E-08			5.0E-08		
	99 %	5.0E-10			2.5E-09			5.0E-09		
1oo2	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.1E-09	2.5E-08	5.0E-08	1.1E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.1E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
2oo2 (see note 2)	**0 %**	1.0E-07			5.0E-07			1.0E-06		
	60 %	4.0E-08			2.0E-07			4.0E-07		
	90 %	1.0E-08			5.0E-08			1.0E-07		
	99 %	1.0E-09			5.0E-09			1.0E-08		
1oo2D (see note 3)	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.1E-09	2.5E-08	5.0E-08	1.1E-08	5.0E-08	1.0E-07
	60 %	1.6E-09	3.2E-09	5.2E-09	8.0E-09	1.6E-08	2.6E-08	1.6E-08	3.2E-08	5.2E-08
	90 %	1.9E-09	2.3E-09	2.8E-09	9.5E-09	1.2E-08	1.4E-08	1.9E-08	2.3E-08	2.8E-08
	99 %	2.0E-09	2.0E-09	2.1E-09	1.0E-08	1.0E-08	1.0E-08	2.0E-08	2.0E-08	2.1E-08
2oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.4E-09	2.5E-08	5.0E-08	1.2E-08	5.1E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.1E-09	1.0E-08	2.0E-08	4.3E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
1oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.0E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09

Architecture	DC	λ_D = 2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	2.5E-06			5.0E-06			2.5E-05		
	60 %	1.0E-06			2.0E-06			1.0E-05		
	90 %	2.5E-07			5.0E-07			2.5E-06		
	99 %	2.5E-08			5.0E-08			2.5E-07		
1oo2	**0 %**	6.3E-08	2.6E-07	5.1E-07	1.5E-07	5.4E-07	1.0E-06	1.8E-06	3.6E-06	5.9E-06
	60 %	2.2E-08	1.0E-07	2.0E-07	4.9E-08	2.1E-07	4.1E-07	4.2E-07	1.2E-06	2.2E-06
	90 %	5.1E-09	2.5E-08	5.0E-08	1.1E-08	5.0E-08	1.0E-07	6.4E-08	2.6E-07	5.1E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.2E-09	2.5E-08	5.0E-08
2oo2 (see note 2)	**0 %**	5.0E-06			1.0E-05			5.0E-05		
	60 %	2.0E-06			4.0E-06			2.0E-05		
	90 %	5.0E-07			1.0E-06			5.0E-06		
	99 %	5.0E-08			1.0E-07			5.0E-07		
1oo2D (see note 3)	**0 %**	6.3E-08	2.6E-07	5.1E-07	1.5E-07	5.4E-07	1.0E-06	1.8E-06	3.6E-06	5.9E-06
	60 %	8.2E-08	1.6E-07	2.6E-07	1.7E-07	3.3E-07	5.3E-07	1.0E-06	1.8E-06	2.8E-06
	90 %	9.5E-08	1.2E-07	1.4E-07	1.9E-07	2.3E-07	2.8E-07	9.6E-07	1.2E-06	1.4E-06
	99 %	1.0E-07	1.0E-07	1.0E-07	2.0E-07	2.0E-07	2.1E-07	1.0E-06	1.0E-06	1.0E-06
2oo3	**0 %**	9.0E-08	2.8E-07	5.3E-07	2.6E-07	6.3E-07	1.1E-06	4.5E-06	5.9E-06	7.6E-06
	60 %	2.6E-08	1.1E-07	2.0E-07	6.6E-08	2.2E-07	4.2E-07	8.5E-07	1.6E-06	2.5E-06
	90 %	5.4E-09	2.5E-08	5.0E-08	1.2E-08	5.1E-08	1.0E-07	9.3E-08	2.9E-07	5.3E-07
	99 %	5.1E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.7E-09	2.6E-08	5.1E-08
1oo3	**0 %**	5.0E-08	2.5E-07	5.0E-07	1.0E-07	5.0E-07	1.0E-06	5.5E-07	2.5E-06	5.0E-06
	60 %	2.0E-08	1.0E-07	2.0E-07	4.0E-08	2.0E-07	4.0E-07	2.0E-07	1.0E-06	2.0E-06
	90 %	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07	5.0E-08	2.5E-07	5.0E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08

NOTE 1 This table gives example values of PFH_G, calculated using the equations in B.3.3 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFH_G is equivalent to PFH_S, PFH_L or PFH_{FE} respectively (see B.3.3.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average frequency of a dangerous failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

Table B.12 – Average frequency of a dangerous failure (in high demand or continuous mode of operation) for a proof test interval of six month and a mean time to restoration of 8 h

Architecture	DC	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	5.0E-08			2.5E-07			5.0E-07		
	60 %	2.0E-08			1.0E-07			2.0E-07		
	90 %	5.0E-09			2.5E-08			5.0E-08		
	99 %	5.0E-10			2.5E-09			5.0E-09		
1oo2	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.3E-09	2.5E-08	5.0E-08	1.1E-08	5.1E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.2E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
2oo2 (see note 2)	**0 %**	1.0E-07			5.0E-07			1.0E-06		
	60 %	4.0E-08			2.0E-07			4.0E-07		
	90 %	1.0E-08			5.0E-08			1.0E-07		
	99 %	1.0E-09			5.0E-09			1.0E-08		
1oo2D (see note 3)	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.3E-09	2.5E-08	5.0E-08	1.1E-08	5.1E-08	1.0E-07
	60 %	1.6E-09	3.2E-09	5.2E-09	8.0E-09	1.6E-08	2.6E-08	1.6E-08	3.2E-08	5.2E-08
	90 %	1.9E-09	2.3E-09	2.8E-09	9.5E-09	1.2E-08	1.4E-08	1.9E-08	2.3E-08	2.8E-08
	99 %	2.0E-09	2.0E-09	2.1E-09	1.0E-08	1.0E-08	1.0E-08	2.0E-08	2.0E-08	2.1E-08
2oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.8E-09	2.6E-08	5.1E-08	1.3E-08	5.3E-08	1.0E-07
	60 %	4.1E-10	2.0E-09	4.0E-09	2.1E-09	1.0E-08	2.0E-08	4.5E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.1E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
1oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.0E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09

Architecture	DC	λ_D = 2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	2.5E-06			5.0E-06			2.5E-05		
	60 %	1.0E-06			2.0E-06			1.0E-05		
	90 %	2.5E-07			5.0E-07			2.5E-06		
	99 %	2.5E-08			5.0E-08			2.5E-07		
1oo2	**0 %**	7.6E-08	2.7E-07	5.2E-07	2.1E-07	5.9E-07	1.1E-06	3.1E-06	4.7E-06	6.8E-06
	60 %	2.4E-08	1.0E-07	2.0E-07	5.7E-08	2.1E-07	4.1E-07	6.3E-07	1.4E-06	2.3E-06
	90 %	5.3E-09	2.5E-08	5.0E-08	1.1E-08	5.1E-08	1.0E-07	7.8E-08	2.7E-07	5.2E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.4E-09	2.5E-08	5.0E-08
2oo2 (see note 2)	**0 %**	5.0E-06			1.0E-05			5.0E-05		
	60 %	2.0E-06			4.0E-06			2.0E-05		
	90 %	5.0E-07			1.0E-06			5.0E-06		
	99 %	5.0E-08			1.0E-07			5.0E-07		
1oo2D (see note 3)	**0 %**	7.6E-08	2.7E-07	5.2E-07	2.1E-07	5.9E-07	1.1E-06	3.1E-06	4.7E-06	6.8E-06
	60 %	8.4E-08	1.6E-07	2.6E-07	1.8E-07	3.3E-07	5.3E-07	1.2E-06	2.0E-06	2.9E-06
	90 %	9.5E-08	1.2E-07	1.4E-07	1.9E-07	2.3E-07	2.8E-07	9.8E-07	1.2E-06	1.4E-06
	99 %	1.0E-07	1.0E-07	1.0E-07	2.0E-07	2.0E-07	2.1E-07	1.0E-06	1.0E-06	1.0E-06
2oo3	**0 %**	1.3E-07	3.2E-07	5.5E-07	4.2E-07	7.7E-07	1.2E-06	8.4E-06	9.2E-06	1.0E-05
	60 %	3.3E-08	1.1E-07	2.1E-07	9.1E-08	2.4E-07	4.4E-07	1.5E-06	2.1E-06	2.9E-06
	90 %	5.8E-09	2.6E-08	5.1E-08	1.3E-08	5.3E-08	1.0E-07	1.3E-07	3.2E-07	5.6E-07
	99 %	5.1E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	6.1E-09	2.6E-08	5.1E-08
1oo3	**0 %**	5.0E-08	2.5E-07	5.0E-07	1.0E-07	5.0E-07	1.0E-06	7.1E-07	2.7E-06	5.1E-06
	60 %	2.0E-08	1.0E-07	2.0E-07	4.0E-08	2.0E-07	4.0E-07	2.1E-07	1.0E-06	2.0E-06
	90 %	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07	5.0E-08	2.5E-07	5.0E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08

NOTE 1 This table gives example values of PFH_G, calculated using the equations in B.3.3 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFH_G is equivalent to PFH_S, PFH_L or PFH_{FE} respectively (see B.3.3.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average frequency of a dangerous failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

Table B.13 – Average frequency of a dangerous failure (in high demand or continuous mode of operation) for a proof test interval of one year and a mean time to restoration of 8 h

Architecture	DC	λ_D = 0,5E-07			λ_D = 2,5E-07			λ_D = 0,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	5.0E-08			2.5E-07			5.0E-07		
	60 %	2.0E-08			1.0E-07			2.0E-07		
	90 %	5.0E-09			2.5E-08			5.0E-08		
	99 %	5.0E-10			2.5E-09			5.0E-09		
1oo2	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.5E-09	2.5E-08	5.0E-08	1.2E-08	5.2E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.1E-09	1.0E-08	2.0E-08	4.3E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.1E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
2oo2 (see note 2)	**0 %**	1.0E-07			5.0E-07			1.0E-06		
	60 %	4.0E-08			2.0E-07			4.0E-07		
	90 %	1.0E-08			5.0E-08			1.0E-07		
	99 %	1.0E-09			5.0E-09			1.0E-08		
1oo2D (see note 3)	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.5E-09	2.5E-08	5.0E-08	1.2E-08	5.2E-08	1.0E-07
	60 %	1.6E-09	3.2E-09	5.2E-09	8.1E-09	1.6E-08	2.6E-08	1.6E-08	3.2E-08	5.2E-08
	90 %	1.9E-09	2.3E-09	2.8E-09	9.5E-09	1.2E-08	1.4E-08	1.9E-08	2.3E-08	2.8E-08
	99 %	2.0E-09	2.0E-09	2.1E-09	1.0E-08	1.0E-08	1.0E-08	2.0E-08	2.0E-08	2.1E-08
2oo3	**0 %**	1.1E-09	5.1E-09	1.0E-08	6.6E-09	2.6E-08	5.1E-08	1.6E-08	5.5E-08	1.0E-07
	60 %	4.1E-10	2.0E-09	4.0E-09	2.3E-09	1.0E-08	2.0E-08	5.0E-09	2.1E-08	4.1E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.2E-10	2.5E-09	5.0E-09	1.1E-09	5.1E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09
1oo3	**0 %**	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07
	60 %	4.0E-10	2.0E-09	4.0E-09	2.0E-09	1.0E-08	2.0E-08	4.0E-09	2.0E-08	4.0E-08
	90 %	1.0E-10	5.0E-10	1.0E-09	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08
	99 %	1.0E-11	5.0E-11	1.0E-10	5.0E-11	2.5E-10	5.0E-10	1.0E-10	5.0E-10	1.0E-09

Architecture	DC	λ_D = 2,5E-06			λ_D = 0,5E-05			λ_D = 2,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %	β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo1 (see note 2)	**0 %**	2.5E-06			5.0E-06			2.5E-05		
	60 %	1.0E-06			2.0E-06			1.0E-05		
	90 %	2.5E-07			5.0E-07			2.5E-06		
	99 %	2.5E-08			5.0E-08			2.5E-07		
1oo2	**0 %**	1.0E-07	2.9E-07	5.4E-07	3.1E-07	6.8E-07	1.1E-06	5.8E-06	6.9E-06	8.5E-06
	60 %	2.9E-08	1.1E-07	2.1E-07	7.4E-08	2.3E-07	4.2E-07	1.1E-06	1.7E-06	2.6E-06
	90 %	5.5E-09	2.5E-08	5.0E-08	1.2E-08	5.2E-08	1.0E-07	1.0E-07	3.0E-07	5.4E-07
	99 %	5.1E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.6E-09	2.6E-08	5.0E-08
2oo2 (see note 2)	**0 %**	5.0E-06			1.0E-05			5.0E-05		
	60 %	2.0E-06			4.0E-06			2.0E-05		
	90 %	5.0E-07			1.0E-06			5.0E-06		
	99 %	5.0E-08			1.0E-07			5.0E-07		
1oo2D (see note 3)	**0 %**	1.0E-07	2.9E-07	5.4E-07	3.1E-07	6.8E-07	1.1E-06	5.8E-06	6.9E-06	8.5E-06
	60 %	8.9E-08	1.7E-07	2.7E-07	1.9E-07	3.5E-07	5.4E-07	1.7E-06	2.3E-06	3.2E-06
	90 %	9.6E-08	1.2E-07	1.4E-07	1.9E-07	2.3E-07	2.8E-07	1.0E-06	1.2E-06	1.4E-06
	99 %	1.0E-07	1.0E-07	1.0E-07	2.0E-07	2.0E-07	2.1E-07	1.0E-06	1.0E-06	1.0E-06
2oo3	**0 %**	2.1E-07	3.8E-07	6.1E-07	7.3E-07	1.0E-06	1.4E-06	1.6E-05	1.6E-05	1.6E-05
	60 %	4.6E-08	1.2E-07	2.2E-07	1.4E-07	2.9E-07	4.7E-07	2.8E-06	3.2E-06	3.8E-06
	90 %	6.6E-09	2.6E-08	5.1E-08	1.6E-08	5.6E-08	1.0E-07	2.1E-07	3.9E-07	6.2E-07
	99 %	5.2E-10	2.5E-09	5.0E-09	1.1E-09	5.1E-09	1.0E-08	6.9E-09	2.7E-08	5.1E-08
1oo3	**0 %**	5.1E-08	2.5E-07	5.0E-07	1.1E-07	5.1E-07	1.0E-06	1.4E-06	3.2E-06	5.5E-06
	60 %	2.0E-08	1.0E-07	2.0E-07	4.0E-08	2.0E-07	4.0E-07	2.6E-07	1.0E-06	2.0E-06
	90 %	5.0E-09	2.5E-08	5.0E-08	1.0E-08	5.0E-08	1.0E-07	5.1E-08	2.5E-07	5.0E-07
	99 %	5.0E-10	2.5E-09	5.0E-09	1.0E-09	5.0E-09	1.0E-08	5.0E-09	2.5E-08	5.0E-08

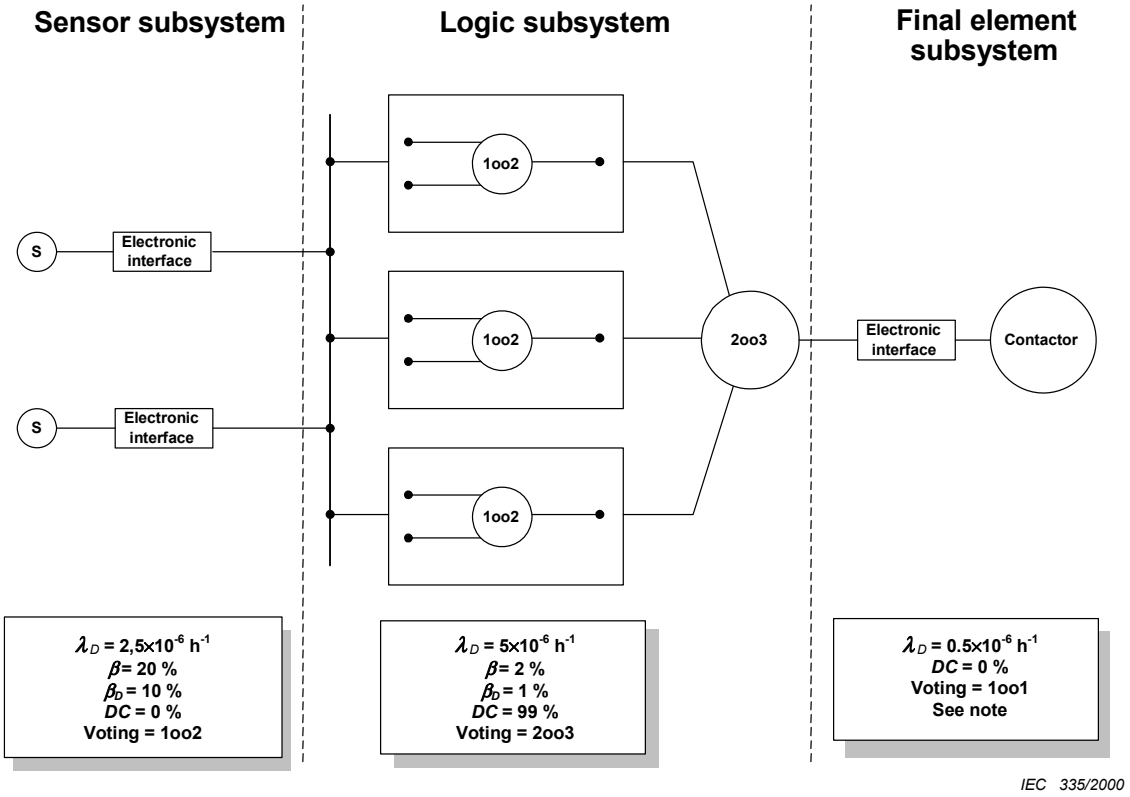
NOTE 1 This table gives example values of PFH_G, calculated using the equations in B.3.3 and depending on the assumptions listed in B.3.1. If the sensor, logic or final element subsystem comprises of only one group of voted channels, then PFH_G is equivalent to PFH_S, PFH_L or PFH_{FE} respectively (see B.3.3.1).

NOTE 2 The table assumes $\beta = 2 \times \beta_D$. For 1oo1 and 2oo2 architectures, the values of β and β_D do not affect the average frequency of a dangerous failure.

NOTE 3 The safe failure rate is assumed to be equal to the dangerous failure rate and K = 0,98.

B.3.3.4 Example for high demand or continuous mode of operation

Consider a safety function requiring a SIL 2 system. Suppose that the initial assessment for the system architecture, based on previous practice, is for one group of two sensors, voting 1oo2. The logic subsystem is a redundant 2oo3 configured PE system driving a single shutdown contactor. This is shown in Figure B.15 For the initial assessment, a proof test period of six months is assumed.



NOTE The final element subsystem has an overall safe failure fraction greater than 60 %.

Figure B.15 – Architecture of an example for high demand or continuous mode of operation

Table B.14 – Average frequency of a dangerous failure for the sensor subsystem in the example for high demand or continuous mode of operation (six month proof test interval and 8 h *MTTR*)

Architecture	*DC*	λ_D =2,5E-06		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
1oo2	0 %	7,6E-08	2,7E-07	5,2E-07
	60 %	2,4E-08	1,0E-07	2,0E-07
	90 %	5,3E-09	2,5E-08	5,0E-08
	99 %	5,0E-10	2,5E-09	5,0E-09
NOTE This table is abstracted from Table B.12.				

Table B.15 – Average frequency of a dangerous failure for the logic subsystem in the example for high demand or continuous mode of operation (six month proof test interval and 8 h *MTTR*)

Architecture	*DC*	λ_D = 0,5E-05		
		β = 2 % β_D = 1 %	β = 10 % β_D = 5 %	β = 20 % β_D = 10 %
2oo3	0 %	4,2E-07	7,7E-07	1,2E-06
	60 %	9,1E-08	2,4E-07	4,4E-07
	90 %	1,3E-08	5,3E-08	1,0E-07
	99 %	1,0E-09	5,0E-09	1,0E-08
NOTE This table is abstracted from Table B.12.				

Table B.16 – Average frequency of a dangerous failure for the final element subsystem in the example for high demand or continuous mode of operation (six month proof test interval and 8 h *MTTR*)

Architecture	*DC*	λ_D = 0,5E-06
1oo1	0 %	5,0E-07
	60 %	2,0E-07
	90 %	5,0E-08
	99 %	5,0E-09
NOTE This table is abstracted from Table B.12.		

From Tables B.14 to B.16 the following values are derived.

For the sensor subsystem,

$PFH_S = 5{,}2 \times 10^{-7}$ / h

For the logic subsystem,

$PFH_L = 1{,}0 \times 10^{-9}$/ h

For the final element subsystem,

$PFH_{FE} = 5{,}0 \times 10^{-7}$ / h

Therefore, for the safety function,

$PFH_{SYS} = 5{,}2 \times 10^{-7} + 1{,}0 \times 10^{-9} + 5{,}0 \times 10^{-7}$

$= 1{,}02 \times 10^{-6}$ / h

≡ **safety integrity level 1**

To improve the system to meet safety integrity level 2, one of the following could be done:

a) change the input sensor type and mounting to improve the defences against common cause failure, thus improving β from 20 % to 10 % and β_D from 10 % to 5 %;

$PFH_S = 2{,}7 \times 10^{-7}$ / h

$PFH_L = 1{,}0 \times 10^{-9}$ / h

$PFH_{FE} = 5{,}0 \times 10^{-7}$ / h

$PFH_{SYS} = 7{,}7 \times 10^{-7}$ / h

≡ **safety integrity level 2**

b) change the single output device to two devices in 1oo2 (β = 10 % and β_D = 5 %).

PFH_S = $5{,}2 \times 10^{-7}$ / h

PFH_L = $1{,}0 \times 10^{-9}$ / h

PFH_{FE} = $5{,}1 \times 10^{-8}$ / h

PFH_{SYS} = $5{,}7 \times 10^{-7}$ / h

≡ **safety integrity level 2**

B.4 Boolean approach

B.4.1 General

The Boolean approach encompasses the techniques representing the logical function linking the individual component failures to the overall system failure. The main Boolean models used in the reliability field are Reliability Block Diagrams (RBD), Fault Trees (FT), Event Trees and Cause Consequence Diagrams. Only the first two methods are considered here. The aim of all these methods is to represent the logical structure of the system. However, its behaviour over time is not included in these model techniques. Therefore, care has to be taken when considering behavioural features (e.g. time dependent features such as periodic proof tests) when undertaking the calculations. The first approach for using Boolean models is to split the graphical representation from the calculations. This has been described in the previous section where RBD are used to model the structure and Markovian calculations used to assess *PFD* or *PFH*. Further considerations are now discussed for probabilistic calculations on RBD and FT.

This approach is limited to components behaving reasonably independently from each other.

B.4.2 Reliability block diagram model

A lot of examples of RBD have been previously given and Figure B.1 represents, for example, a whole safety loop made of three sensors (A, B, C) working in 1oo3, one logic solver (D) and two terminal elements (E, F) working in 1oo2.

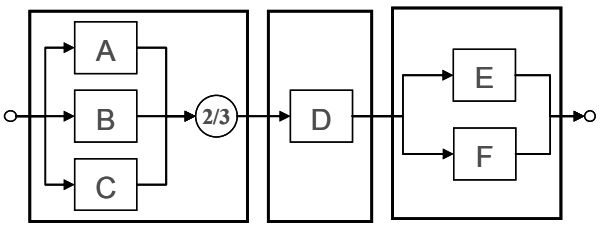


Figure B.16 – Reliability block diagram of a simple whole loop with sensors organised into 2oo3 logic

Figure B.16 shows a similar loop with sensors working in 2oo3. The main interest of such graphical representation is threefold: it remains very close to the physical structure of the system under study, it is widely used by engineers and it is a good support for discussion.

The main shortcoming is that RBD is more a method of representation than a method of analysis in itself.

For more details on RBD see C.6.4 of IEC 61508-7 and IEC 61078.

B.4.3 Fault tree model

Fault trees have exactly the same properties as RBD but in addition they constitute an effective deductive (top-down) method of analysis helping reliability engineers to develop

models step by step from the top event (unwanted or undesirable event) to the individual components failures.

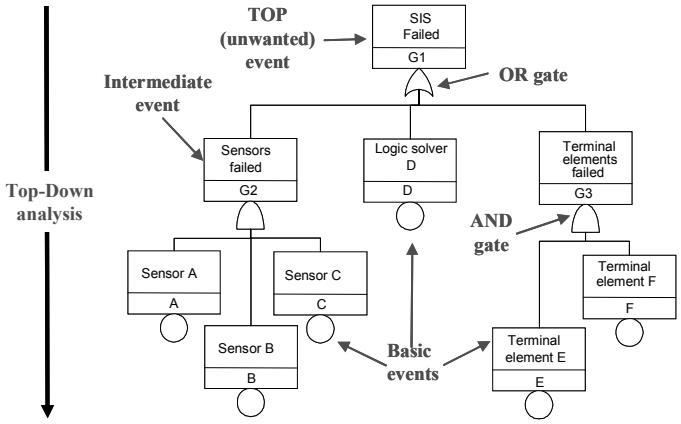


Figure B.17 – Simple fault tree equivalent to the reliability block diagram presented on Figure B.1

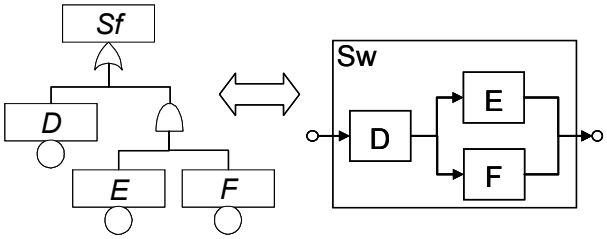
Figure B.17 shows a fault tree which is perfectly equivalent to the RBD presented on Figure B.1 but where the steps of the top-down analysis are identified (for example: E/E/PE safety-related system failed => Sensor failed => sensor A failed). In FT, the elements in series are linked by "OR gates" and element in parallel (i.e. redundant) are linked by "AND gates".

For more details on FT see B.6.6.5 and B.6.6.9 of IEC 61508-7 and IEC 61025.

B.4.4 *PFD* calculations

B.4.4.1 General overview

RBD and FT representing exactly the same things, the calculations may be handled exactly in the same way. Figure B.18 shows small equivalent FT and RBD which will be used to show the main principles of the calculations.



NOTE In this figure, italic letters are used for failed items and non-italic letters for working items.

Figure B.18 – Equivalence fault tree / reliability block diagram

The small FT represents the logical function $Sf = D \cup (E \cap F)$ where *Sf* is the failure of the system and *D, E, F* the failures of the individual components. The small RBD represents the logical function $Sw = D \cap (E \cup F)$ where Sw is the good functioning of the system and D, E, F the good functioning of the individual components. Then *Sf* = NOT Sw, and *Sf* and Sw represent exactly the same information (i.e. the logical function and its dual).

The primary use of FT and RBD is identifying the combinations of the various component failures leading to the overall system failure. They are the minimal cut sets so-called because they indicate where to cut the RBD in order that a signal sent at the input does not reach the output. In this case, there are two cut sets: the single failure (*D*) and the double failure (*E, F*).

Applying basic probabilistic mathematics on logical functions lead straightforwardly to the probability of failure P_{Sf} of the system and we obtain

$$P_{Sf} = P(D) + P(E \cap F) - P(D \cap E \cap F)$$

If the components are independent this formula becomes:

$$P_{Sf} = P_D + P_E P_F - P_D P_E P_F$$

where

P_i is the probability that component i is failed.

This formula is time independent and reflects only the logical structure of the system.

Therefore both RDB and FT are basically static, i.e. time independent models.

Nevertheless, if the probability of failure of each individual component at time *t* is independent of what happens on the other component over *[0, t]* the above formula remains valid at any time and we can write:

$$P_{Sf}(t) = P_D(t) + P_E(t)P_F(t) - P_D(t)P_E(t)P_F(t)$$

The analyst should verify if the required approximations are acceptable or not and finally, the instantaneous unavailability $U_{Sf}(t)$ of the system is obtained:

$$U_{Sf}(t) = U_D(t) + U_E(t)U_F(t) - U_D(t)U_E(t)U_F(t)$$

The conclusion is that fault trees or reliability block diagrams allow calculating directly the instantaneous unavailability $U_{Sf}(t)$ of E/E/PE safety-related systems and that additional calculations are needed and according to B.2.2:

$$PFD_{avg}(T) = \frac{1}{T} MDT(T) = \frac{1}{T}\int_0^T U_{Sf}(t)dt$$

This principle may be applied to minimal cut sets:

– single failure (*D*): $PFD^D(\tau) = \frac{1}{\tau}\int_0^\tau \lambda_D t.dt = \lambda_D \tau / 2$

– double failure (*E, F*): $PFD^{EF}(\tau) = \frac{1}{\tau}\int_0^\tau \lambda_E.\lambda_F.t^2 dt = \lambda_E \lambda_F \tau^2 / 3$

B.4.4.2 Calculations based on fault trees or reliability block diagram tools

The formula $U_{Sf}(t) = U_D(t) + U_E(t)U_F(t) - U_D(t)U_E(t)U_F(t)$ described above is only a particular case of the so-called Poincaré formula. More generally if $Sf = \bigcup_i C_i$ where (C_i) represents the minimal cut sets of the system:

$$P(\bigcup_{i-1}^{n} C_i) = \sum_{j=1}^{n} P(C_j) - \sum_{j=1}^{n}\sum_{i=1}^{j-1} P(C_j \bigcap C_i) + \sum_{j=3}^{n}\sum_{i=2}^{j-1}\sum_{k=1}^{j-1} P(C_j \bigcap C_i \bigcap C_k) - \ldots .$$

The number of minimal cut sets increases exponentially when the number of individual components increases. Then the Poincaré formula leads to a combinatory explosion of terms very quickly intractable by hand. Fortunately this problem has been analysed over the last forty years and numerous algorithms have been developed to manage such calculations. At the present time the last developments and the most powerful ones are based on the so-called Binary Decision Diagrams (BDD) which are derived from a sophisticated Shannon decomposition of the logical function.

A lot of commercial software packages mainly based on fault tree models are used daily by reliability engineers in various industry fields (nuclear, petroleum, aeronautics, automotive, etc.). They can be used for PFD_{avg} calculation but analysts have to be very cautious because some of them implement wrong PFD_{avg} calculations. The main mistake encountered is the conventional combination of the $PFD_{avg,i}$ of individual components (generally obtained simply by $\lambda_i \tau/2$) to produce a result which is supposed to be the whole system PFD_{avg}. As shown above this is wrong and non conservative.

Anyway, fault tree software packages may be used to calculate the instantaneous system unavailability $U_{Sf}(t)$ from the instantaneous unavailabilities of its components $U_i(t)$. Then the average of $U_{Sf}(t)$ may be done over the period of interest to evaluate the PFD_{avg}. Depending on the software in use this can be done by the software itself or by side calculations.

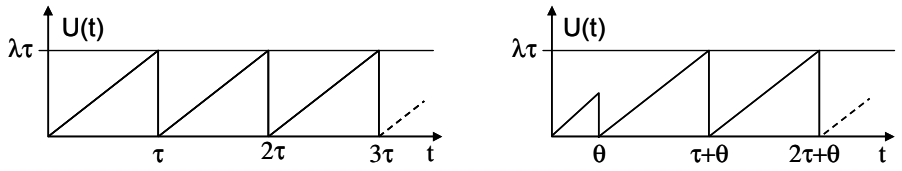


Figure B.19 – Instantaneous unavailability *U(t)* of single periodically tested components

The ideal case previously described is presented on the left of Figure B.19:

$$U_i(t) = \lambda \zeta \text{ and } \zeta = t \text{ modulo } \tau.$$

This is a so called saw-tooth curve increasing linearly from 0 to $\lambda\tau$ and restarting from 0 after a test or a repair (which are considered to be instantaneous as the EUC is stopped during them).

When several components are used in redundant structures, the tests may be staggered as shown on the right of Figure B.19 where the first test interval is different from the others. That has no impact on the PFD_{avg} or on maximum value which are equal to $\lambda\tau/2$ and $\lambda\tau$ in both cases.

Of course in less ideal cases these curves may be more complicated than that. Guidelines will be given in B.5.2 to design more accurate saw tooth curves but for the purpose of this chapter the curves presented on Figure B.19 are sufficient.

This can be applied to the small fault tree presented on Figure B.18 as illustrated in Figure B.20 (where DU means Dangerous Undetected and CCF means Common Cause Failure) . We have considered that the system was made of two redundant components (E and F) and that (D) is a common cause failure on these components. The calculation has been achieved with the following figures:

$$\lambda_{DU} = 3{,}5 \times 10^{-6}/\text{h},\ \tau = 4\ 380 \text{ h and } \beta = 1\ \%$$

A small β factor has been chosen to ensure that CCF does not dominate the result at the top and to gain a better understanding on how that works.

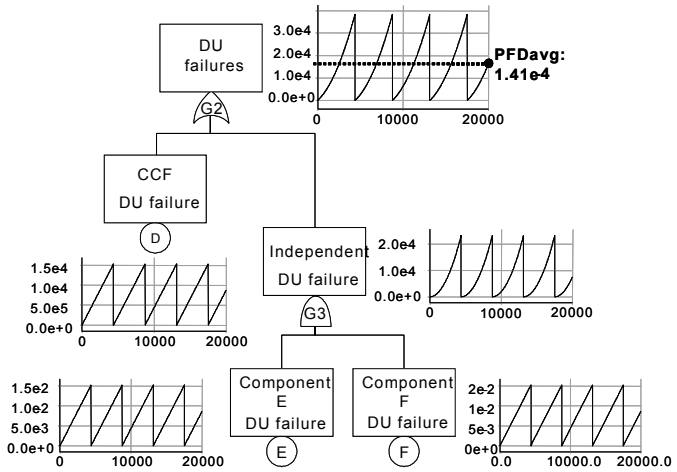


Figure B.20 – Principle of PFD_{avg} calculations when using fault trees

It is easy to recognize the kind of saw-tooth curves presented on the left hand side of Figure B.19 as inputs for *D, E* and *F*. The CCF (*D*) is tested each time *E* or *F* are tested. Then, as *E* and *F* are tested at the same time every 6 months, the CCF (*D*) is also tested every 6 months.

By using one of the algorithms developed for fault tree calculations, it is easy to draw the saw-tooth curves at outputs of all logical gates. The PFD_{avg} is calculated by averaging the result obtained for the top event. This can be performed by Fault Tree software itself or by manual calculations. $PFD_{avg} = 1{,}4 \times 10^{-4}$ is obtained and according to this standard, this meets the target failure measure for SIL 3 for a low demand mode of operation.

As shown on Figure B.20, the graphs are smooth between tests. Therefore there are no difficulties to evaluate the average, provided the instant of tests are identified and taken under consideration.

It is interesting to notice that as soon as redundancy is implemented, the saw-tooth curves at top event level are no longer linear between tests (i.e. the overall system failure rate is no longer constant).

It is also interesting to measure the impact on the PFD_{avg} of staggering the tests of the redundant components instead of performing them at the same time. This is illustrated in Figure B.21 where the tests of the component F have been staggered from those of component E by 3 months.

This has several important effects:

– CCF are now tested every 3 months (i.e. each time E is tested and each time F is tested). This proof test frequency is the double as the previous case.

– the top event saw-tooth curve has also a proof test frequency double than the one applied before.

– the saw tooth curve is less spread around its average than in the previous case.

– the PFD_{avg} has decreased to $8{,}3 \times 10^{-5}$: with this new test policy the system is SIL 4.

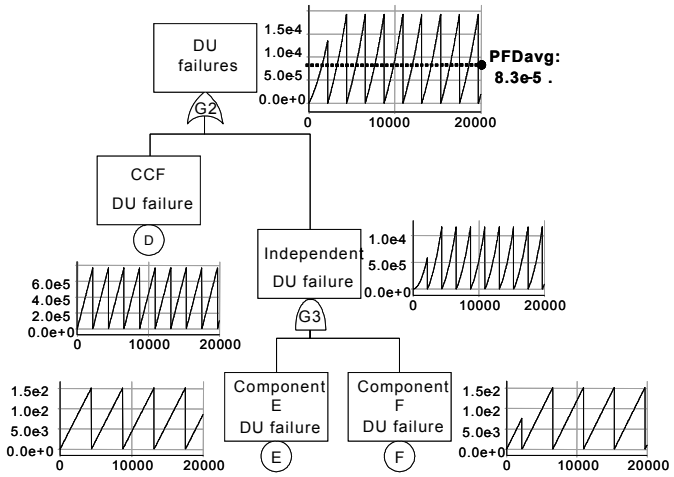


Figure B.21 – Effect of staggering the tests

If the tests are staggered and adequate procedures implemented this will increase the likelihood of detecting CCFs and is an effective method of reducing the CCF for systems operating in a low demand mode of operation. Here it has been improved from SIL 3 to SIL 4 (from hardware failures point of view and if other requirements of the IEC 61508 series are met).

Figure B.22 represents the saw-tooth curve obtained when adding a component G (λ_{DU} = 7 × 10^{-9}/h and never tested) and a component H (λ_{DU} = 4 × 10^{-8}/h tested every 2 years) in series with the system modelled on Figure B.20.

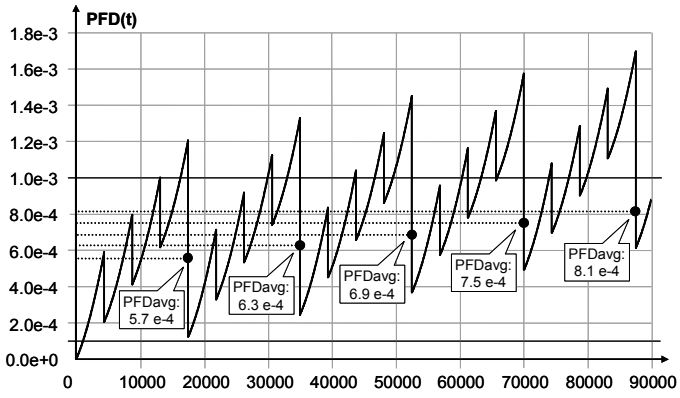


Figure B.22 – Example of complex testing pattern

The impact of the component G never tested is twofold: *PFD(t)* never goes back to zero after the test performed every two years and PFD_{avg} increases continuously (black circles corresponding to the PFD_{avg} over the period covered by the related dotted line).

Even if the elementary saw-tooth curves (like those presented on Figure B.19) are very simple, the results at the top event level may be rather complicated but this does not raise particular difficulties.

This clause aims only to illustrate the principle of the calculation by using Boolean models. Subclause B.5.2 related to the Markovian approaches, will give some guidelines to design more sophisticated input saw-tooth curves for elementary components.

It can be concluded that, when the individual components are reasonably independent there is no problem to handle PFD_{avg} calculations for E/E/PE safety-related systems by using

classical Boolean techniques. This is not so simple from theoretical point of view and the analyst doing the study should have acquired a sound knowledge of the probabilistic calculations to identify and discard the wrong PFD_{avg} implementations sometimes encountered. Provided these precautions are taken, any fault tree software package may be used for above calculations.

Boolean techniques may be also used for *PFH* calculations but the theoretical developments are beyond the purpose of this informative annex.

B.5 States/transition approaches

B.5.1 General

Boolean models are basically time-independent and the introduction of time is possible only in specific particular cases. This is rather artificial and a good knowledge of probabilistic calculations is needed to avoid mistakes. Therefore other probabilistic models, dynamic in nature may be used instead. In the reliability field they are fundamentally based on the next approach in two steps:

– identification of all the states of the system under study;

– analysis of the jumps (transitions) of the system from states to states, according to events arising and along its life.

This is why they are gathered in the category of states/transitions models.

The general approach actually consists in building a kind of automaton behaving like the system under study when events (failures, repairs, tests, etc.) are arising. As per this standard, E/E/PE safety-related systems have only discrete states, this is equivalent to building a so-called finite state automaton. Those models are dynamic in nature and may be implemented in various manners: graphic representations, specific formal languages or common programming languages. This annex presents two of them which are very different but complementary:

– Markov model which has been developed at the very beginning of the last century. It is rather well known and handled analytically;

– Petri net model which has been developed in the sixties. It is less well known (but more and more used because of its flexibility) and handled by Monte Carlo simulation.

Both are based on graphical drawings very helpful for users. Other techniques based on formal languages modelling will be very quickly analysed at the end of this clause.

B.5.2 Markovian approach

B.5.2.1 Principle of modelling

The Markovian approach is the elder of all the dynamic approaches used in the reliability field. Markov processes are split between those which are "amnesic" (homogeneous Markov processes where all transition rates are constant) and the others (semi Markov processes). As the future of a homogeneous Markov process does not depend on its past, analytical calculations are relatively straightforward. This is more difficult for semi Markov processes for which Monte Carlo simulation can be used. In this part of the IEC 61508 series, only homogeneous Markov processes are considered and the term "Markov processes" is used for the sake of simplicity (see C.6.4 of IEC 61508-7 and IEC 61165).

The fundamental basic formula of Markov processes is the following:

$$P_i(t+dt) = \sum_{k \neq i} P_k(t)\lambda_{ki}dt + P_i(t)(1 - \sum_{k \neq i} \lambda_{ik}dt)$$

In this formula, λ_{ki} is the transition rate (e.g. failure or repair rate) from state *i* to state *k*. It is self explaining: the probability to be in state *i* at *t+dt* is the probability to jump toward *i* (when in another state *k*) or to remain in state *i* (if already in this state) between *t* and *t* + *dt*.

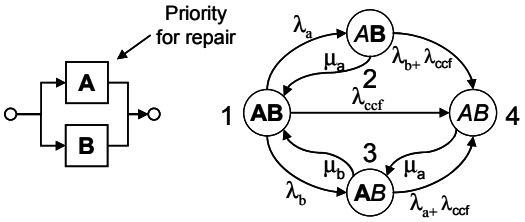


Figure B.23 – Markov graph modelling the behaviour of a two component system

There is a straightforward relationship between the above equation and a graphical representation like Figure B.23 which models a system made of two components with a single repair team (component A having priority to be repaired) and a common cause failure. In this figure **A** indicates that A is working and *A* that it has failed. As the detection times must be considered μ_a and μ_b in Figure B.23 are the restoration rates of the components (i.e. μ_a=*1/MTTR*$_a$ and μ_b=*1/MTTR*$_b$).

For example the probability to be in state 4 is simply calculated as follows:

$$P_4(t+dt) = [P_1(t)\lambda_{ccf} + P_2(t)(\lambda_b + \lambda_{ccf}) + P_3(t)(\lambda_a + \lambda_{ccf})]dt + P_4(t)(1-\mu_a dt)$$

This leads to a vectorial differential equation,

$d\vec{P}(t)/dt = [M]\vec{P}(t)$, which is conventionally solved by:

$$\vec{P}(t) = e^{t[M]}\vec{P}(0)$$

where

[*M*] is the Markovian matrix containing the transition rates and $\vec{P}(0)$ the vector of the initial conditions (generally a column vector with 1 for the perfect state an 0 for the others).

Even if an exponential of a matrix has not exactly the same properties of an ordinary exponential, it is possible to write:

$$\vec{P}(t) = e^{(t-t1)[M]}e^{t1[M]}\vec{P}(0) = e^{(t-t1)[M]}\vec{P}(t1)$$

This demonstrates the basic property of Markov processes: the knowledge of the probabilities of the states at a given instant *t1* summarizes all the past and is enough to calculate how the system evolves in the future from *t1*. This is very useful for *PFD* calculations.

Efficient algorithms have been developed and implemented in software packages a long time ago in order to solve above equations. Then, when using this approach, the analyst can focus only on the building of the models and not on the underlying mathematics even if, anyway, he has to understand at least what is described in this appendix.

Figure B.24 shows the principle of *PFD* calculations:

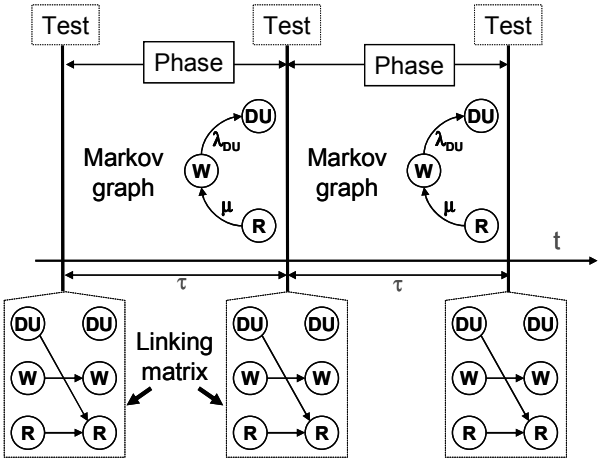


Figure B.24 – Principle of the multiphase Markovian modelling

PFD calculations are related to E/E/PE safety-related systems working in low demand mode and periodically (proof) tested. For such systems repairs are initiated only when tests are performed. The tests are singular points along the time but this is not a problem as a multi phase Markovian approach may be used to deal with.

For example, a simple system made of one periodically tested single component has three states as shown on Figure B.24: working (W), dangerous failure undetected (DU) and under repair (R).

Between tests its behaviour is modelled by the Markovian process on the upper part of Figure B.24: it can fail (W→ DU) or under repair (R→ W). As no repair may be started within a test interval, there is no transition from DU to R. Because the diagnostic of the failure has been performed before entering state R, μ is the repair rate of the component (i.e. μ=1/*MRT*) in Figure B.24.

When a test is performed (see linking matrix on Figure B.14), a repair is started if a failure has occurred (DU→ R), the component remains working if it was in a good functioning state (W→ W) and in the very hypothetical case that a repair started at the previous test is not finished, remains under repair (R→ R). A linking matrix [*L*] may by used to calculate the initial conditions at the beginning of state *i+1* from the probabilities of the states at the end of test *i*. This gives the following equation:

$$\begin{bmatrix} P_{DU}(0) \\ P_w(0) \\ P_R(0) \end{bmatrix}_{i+1} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_{DU}(\tau) \\ P_w(\tau) \\ P_R(\tau) \end{bmatrix}_i \equiv \vec{P}_{i+1}(0) = [L]\vec{P}_i(\tau)$$

Replacing $\vec{P}_i(\tau)$ by its value, leads to an equation of recurrence which allows to calculate the initial conditions at the beginning of each test intervals:

$$\vec{P}_{i+1}(0) = [L]e^{t[M]}\vec{P}_i(0)$$

This can be used to calculate the probabilities at any time $t = i\tau + \zeta$. For example, within test interval *i,* the following is obtained:

$$\vec{P}(t) = \vec{P}_i(\zeta) = e^{\zeta[M]}\vec{P}_i(0), \quad (i-1)\tau \le t < i\tau, \quad \zeta = t \bmod \tau$$

Obtaining the instantaneous unavailability is straightforward by summing the probabilities of the states where the system is unavailable. A line vector (q_k) is helpful to express that:

$$U(t) = \sum_{k=1}^{n} q_k P_k(t)$$

where $q_k = 1$ if the system is unavailable in state k, and $q_k = 0$ otherwise.

For the simple model, $PFD(t) = U(t) = P_{DU}(t) + P_R(t)$ is obtained and the look of the saw-tooth curve obtained with this model is illustrated on Figure B.25.

The PFD_{avg} is calculated in the way previously described through the *MDT* which in turn is easy to calculate from the Mean Cumulated Times spent in the states:

$$\vec{MCT}(T) = \int_0^T \vec{P}(t)dt$$

Like for $\vec{P}(t)$, efficient algorithms are well known to perform this calculation over *[0, T]* to finally obtain: $PFD_{avg}(T) = \frac{1}{T}\sum_{k=1}^{n} q_k MCT_k(T)$

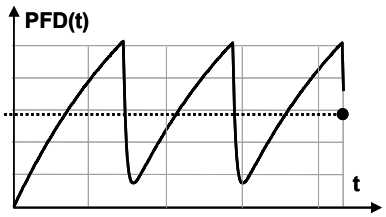


Figure B.25 – Saw-tooth curve obtained by multiphase Markovian approach

Applying this formula on the model presented on Figure B.24 leads to:

$$PFD_{avg}(T) = \frac{1}{T}\left[MCT_{DU}(T) + MCT_R(T)\right]$$

This may be reduced to the first term if the EUC is shut down during the repair.

The black circle on Figure B.25 is the PFD_{avg} of the saw-tooth curve over the whole period of calculation.

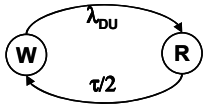


Figure B.26 – Approximated Markovian model

Note that the above calculations are often performed by using the approximated model presented on Figure B.26 where the state DU and R have been merged and where $\tau/2$ (i.e. the mean time to detect the failure) has been used as equivalent restoration time. This is valid only if the Markovian equations have been previously solved by another way in order to find this equivalence. This approximation is only applicable if the repair time is negligible. Also, the method may be very difficult for large complex systems.

The simple model of Figure B.24 may be easily improved for more realistic components. On Figure B.27, the linking matrix models a component which has both a probability, γ,to be failed due to the test (i.e. a genuine on demand failure) and a probability, σ, that the failure was not discovered by the test (by human error).

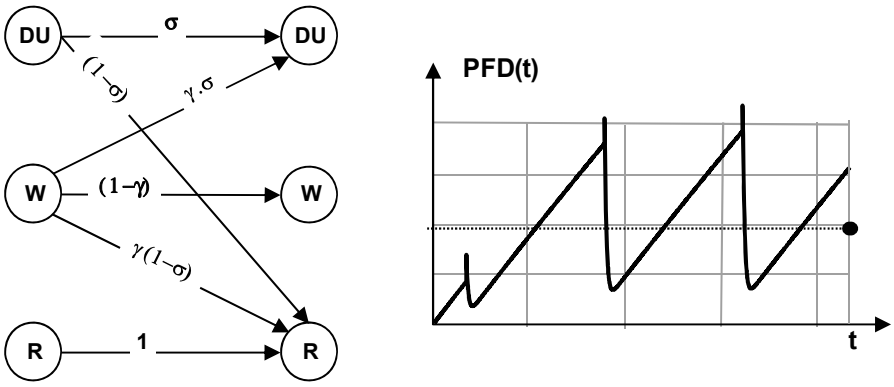


Figure B.27 – Impact of failures due to the demand itself

The look of the saw-tooth curve has changed and the jumps observed for each test correspond to the probability of on demand failure γ. Again the black circle presents the PFD_{avg}.

When a (redundant) component is disconnected to be tested, it becomes unavailable during the whole performance of the test and this contributes to its PFD_{avg}. Therefore, the test duration, π,shall be considered and an additional phase introduced between test intervals. This is shown on Figure B.28 where states R and W are modelled in this phase only for the sake of the completeness.

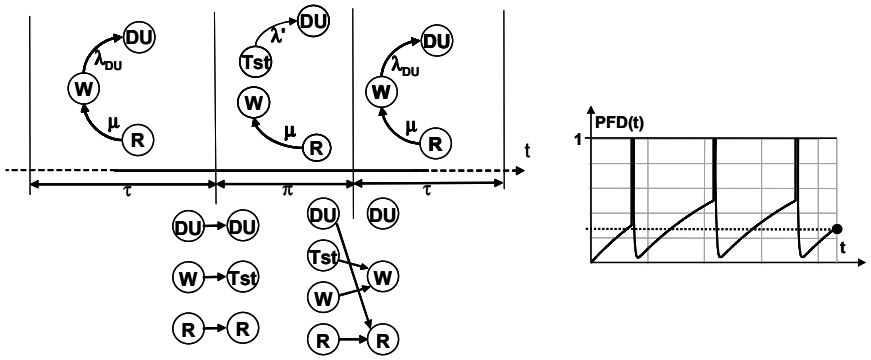


Figure B.28 – Modelling of the impact of test duration

In this Markov model, the system is unavailable in states R, DU and Tst. This is more complicated than before but the principle of calculation remains exactly the same. The behaviour of the saw-tooth curve is shown on the right. The system is unavailable during the test durations and this may be the top contribution to PFD_{avg}.

On the previous Markov graphs, only the dangerous undetected failures have been considered but the dangerous detected failure may be represented as well. The difference is that repair starts at once as it is represented on Figure B.29. Therefore μ_{DD} is the restoration rate of the component (μ_{DD} =*1/MTTR*)when μ_{DU} is its repair rate (μ_{DU}=1*/MRT)*.

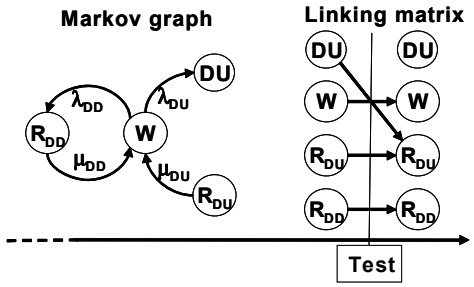


Figure B.29 – Multiphase Markovian model with both DD and DU failures

Safe failures should be represented if needed but the Markov graphs presented here are chosen to be as simple as possible.

The main problem with Markov graphs is that the number of states increases exponentially when the number of components of the system under study increases. Therefore building Markov graphs and performing above calculations without drastic approximations becomes very quickly intractable by hand.

Using an efficient Markovian software package helps to cope with calculation difficulties. There are a lot of such packages available even if they are not necessarily usable directly for PFD_{avg} calculation purpose: most of them perform instantaneous unavailability calculations but only some of them calculate the mean cumulated times spent in the states and only a few allow multiphase modelling. Anyway, there are no real difficulties to adapt them to PFD_{avg} calculations.

Concerning the modelling itself and when dependencies between components are light, Markovian and Boolean approaches can be mixed:

– Markov models are used to establish the instantaneous unavailabilities of each of the components;

– fault trees or reliability block diagrams are used to combine the individual unavailabilities to calculate the instantaneous unavailability *PFD(t)* of the whole system;

– PFD_{avg} is obtained by averaging *PFD(t)*.

This mixed approach has been described in Clause B.4 and saw-tooth curves like those on Figures B.25, B.27 and B.28 may be used as input to fault trees.

When dependencies between components cannot be neglected, some tools are available to build automatically the Markov graphs. They are based on models of a higher level than Markov models (e.g. Petri nets, formal language). Due to the combinatory explosion of the number of states, can still lead to difficulties.

This combined approach is very efficient for modelling complex systems.

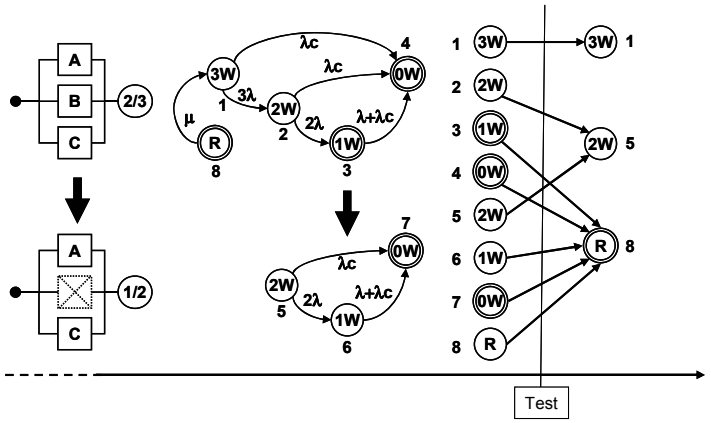


Figure B.30 – Changing logic (2oo3 to 1oo2) instead of repairing first failure

The system modelled in Figure B.30 is made of three components tested at the same time and working in 2oo3. When a failure is detected, the logic is changed from 2oo3 to 1oo2 as a 1oo2 logic is better than a 2oo3 logic from safety point of view (but worse from spurious failure point of view). It is only when a second failure is detected that the repair occurs and this consists in replacing all the three elements by three new ones. This introduces systemic constraints and it is not possible to build the behaviour of the whole system just by combining the independent behaviours of its components.

B.5.2.2 Principle of *PFH* calculations

For *PFH* calculations, the same type of multiphase Markovian modelling can be used for DU failures detected by proof tests. In order to simplify, only the principle of *PFH* calculations for DD failures which only need conventional (monophase) Markov models is shown. Of course, for E/E/PE safety-related systems working in continuous mode and having DU failures detected by periodical proof tests, the multiphase Markovian approach should be used. This does not change the principle considered hereafter.

Figure B.31 presents two Markov graphs modelling the same system made of two redundant components with a common cause failure. On the left hand side, the components (A and B) are repairable. On the right hand side, they are not repairable.

On both graphs state 4 (*AB*) is absorbing. The system remains failed after an overall failure and *P(t) = P1(t) + P2(t) + P3(t)* is the probability that no failure has occurred over *[0, t]*. Then, *R(t) = P(t)* is the reliability of the system and *F(t) = 1 - R(t) = P4(t)* is its unreliability.

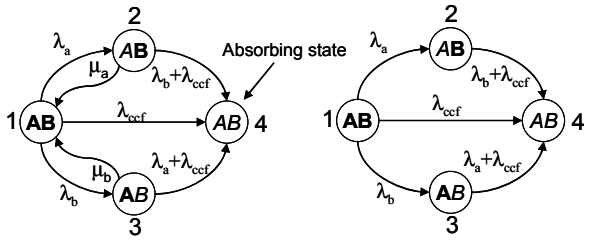


Figure B.31 – "Reliability" Markov graphs with an absorbing state

As discussed in B.2.3., this reliability model is adequate for dealing with the situation when the failure of the E/E/PE safety-related system leads immediately to a dangerous situation. Again μ_a and μ_b are the restoration rates of the components (i.e. μ_a=1/*MTTR*$_a$ and μ_b=1/*MTTR*$_b$)

Such a reliability Markov graph provides the *PFH* directly by *PFH* = *F(T)/T*. For example from Figure B.31 $PFH(T) = P4(T)/T$ (provided *P4(T)* << 1), is directly obtained.

Such a reliability Markov graph also allows the *MTTF* of the system to be calculated by the following formula:

$$MTTF = \lim_{t \to \infty} \sum_{k=1}^{n} a_k MCT_k(t)$$

In this formula $MCT_k(t)$ is the mean cumulated time spent in state *k,* and a_k = 1 if *k* is a good working state and a_k = 0 otherwise.

An upper bound is obtained by:

$$PFH \approx 1/MTTF$$

Efficient algorithms have been developed and almost all the Markovian software packages may be used for *F(T)* and *MTTF* calculations.

The above *PFH* estimations are valid in any case even if there is no overall system constant failure rate (as for the graph on the right hand side of Figure B.31). The only constraint is to use a reliability Markov graph with one (or several) absorbing state(s). Of course this holds when using a multiphase model.

When all the states are completely and quickly repairable, the overall system failure rate $\Lambda(t)$ converge quickly toward an asymptotic value Λ_{as} = 1/*MTTF*. In such graphs, except the perfect and the absorbing states, all states are quasi instantaneous (because the *MTTRs* of the components are short compared to their *MTTF*). This allows evaluating directly the overall system constant failure rates of each scenario starting from the perfect state and leading to the absorbing state. The Markov graph on the left hand side of Figure B.31 models such a completely and quickly repairable system. That is:

– $1 \rightarrow 4$: $\Lambda_{14} = \lambda_{ccf}$

– $1 \rightarrow 2 \rightarrow 4$: $\Lambda_{124} = \lambda_a . (\lambda_b + \lambda_{ccf})/[(\lambda_b + \lambda_{ccf}) + \mu_a] \approx \lambda_a . (\lambda_b + \lambda_{ccf})/\mu_a$

– $1 \rightarrow 3 \rightarrow 4$: $\Lambda_{134} = \lambda_b . (\lambda_a + \lambda_{ccf})/[(\lambda_a + \lambda_{ccf}) + \mu_b] \approx \lambda_b . (\lambda_a + \lambda_{ccf})/\mu_b$

For the scenario $1 \rightarrow 3 \rightarrow 4$ in above formulae, λ_b is the transition rate governing the jump out of the perfect state, and $(\lambda_a + \lambda_{ccf})/\mu_b$ is the probability to jump to 4 rather to come back to 1 when in state 3.

Finally: $\Lambda_{as} = \Lambda_{12} + \Lambda_{124} + \Lambda_{134} = 1/\text{MTTF}$

This can be easily generalized to complex Markov graphs but this is valid only for completely and quickly repairable systems, i.e. DD failures.

The Markov graph on the right hand side of Figure B.31 is not completely and quickly repairable. Therefore applying above calculation would lead to wrong results.

When the E/E/PE safety-related system working in continuous mode is used in conjunction with other safety barriers, its availability shall be considered. This is what is represented on both graphs of Figure B.32 below: there is no absorbing state and the system is repaired after an overall failure. *P(t) = P1(t) + P2(t) + P3(t)* is the probability that the system is working at *t*. Then, *A(t) = P(t)* is its availability and *U(t)* = 1 - *A(t) = P4(t)* its unavailability.

This case is very different from the example represented in Figure B.31 and *R(t)* and *A(t)* should be used correctly, as should *U(T)* and *F(T)* if correct results are to be obtained.

In case of DD failures, the simplest way to handle this problem is to calculate the upper bound of *PFH* through *MDT* and *MUT* as explained in B.2.3.

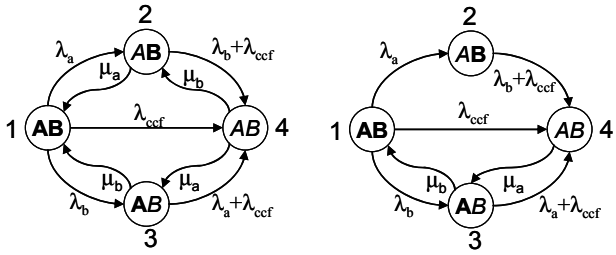


Figure B.32 – "Availability" Markov graphs without absorbing states

An interesting property of availability Markov graphs is that they reach an asymptotic equilibrium when the probability to enter a given state is equal to the probability to get out of it. Noted by:

– $P_{i,as} = \lim_{t\to\infty} P_i(t)$ the asymptotic value of $P_i(t)$

– $\lambda_i = \sum_{j\neq i} \lambda_{ij}$ the transition rate from *i* to any other state

Each time the system visits state *i*, the mean time in this state is $Mst_i = 1/\lambda_i$.

This allows calculating $MUT = \sum_i (1-q_i) P_{i,as} Mst_i$ and $MDT = \sum_i q_i P_{i,as} Mst_i$

where

q_i = 0, if *i* is a working state, and 1 otherwise.

Finally the following is obtained: $PFH = 1/(MUT + MDT) = 1/\sum_i P_{i,as} Mst_i = 1/\sum_i \frac{P_{i,as}}{\lambda_i}$

It should be noted that the number of failures observed over *[0, T]* is given by: $n = T/\sum_i \frac{P_{i,as}}{\lambda_i}$.

As most of the Markovian software packages are able to find the asymptotic probabilities there are no particular difficulties to achieve above calculations.

When the period under interest is too short for allowing the convergence of the Markov process, *PFH* may be calculated with: $w(t) = \sum_{i\neq f} \lambda_{if} P_i(t)$.

This gives: $PFH(T) = \sum_{i\neq f} \lambda_{if} \frac{\int_0^T P_i(t)dt}{T} = \frac{\sum_{i\neq f} \lambda_{if} MCT_i(T)}{T}$

There is again no difficulty to perform these calculations with a Markov software package providing the cumulated time in each of the states.

In the case of completely and quickly repairable systems (DD failures) the Vesely rate $\Lambda_v(t)$ converges very quickly toward an asymptotic value, Λ_{as}, which is a good approximation of the overall system constant failure rate of the system. Therefore in this case, the *PFH* may be calculated in the same way as in the reliability case.

In case of DU failures, this is more complicated due to the multiphase modelling. The above formula may be generalized to: $PFH(T) = \dfrac{\sum_{\varphi=1}^{n} \sum_{i \neq f} \lambda_{if} MCT_i(T_\varphi)}{\sum_{\varphi=1}^{n} T_\varphi}$

In this formula, T_φ is the duration of phase φ.

Multiphase Markovian processes generally reach equilibrium when the probability to jump out off a given state is equal to the probability to enter in it. The asymptotic values have nothing to do with those which are described above but they can be used in the above formula.

In conclusion, it can be said that the Markovian approach provides a lot of possibilities to calculate the *PFH* of an E/E/PE safety-related system working in continuous mode. Nevertheless, a good understanding of the underlying mathematics is needed to use them properly.

B.5.3 Petri nets and Monte Carlo simulation approach

B.5.3.1 Principle of modelling

An efficient way of modelling dynamic systems is to build a finite state automaton behaving as close as possible as the E/E/PE safety-related systems under study. Petri nets (see B.2.3.3 and B.6.6.10 of IEC 61508-7) have been proven to be very efficient for this purpose for the following reasons:

– they are easy to handle graphically;

– the size of the models increases linearly according to the number of components to be modelled;

– they are very flexible and allow modelling almost all type of constraints;

– they are a perfect support for Monte Carlo simulation (see B.6.6.8 of IEC 61508-7).

Originally developed in the 1960's for the formal proof on automata, they have been quickly hijacked in two steps by reliability engineers, in the seventies, for the automatic building of big Markov graphs and in the 1980's, for Monte Carlo simulation purpose.

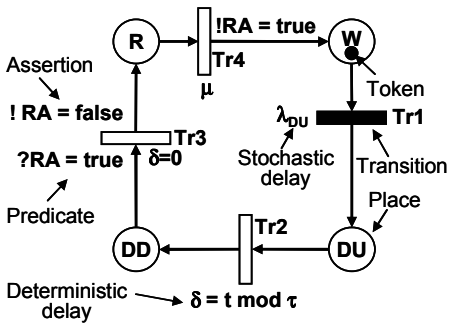


Figure B.33 – Petri net for modelling a single periodically tested component

The typical sub-Petri net for a simple periodically tested component is made of three parts:

1) the static part (i.e. a drawing):
   a) places (circles) corresponding to potential states;
   b) transitions (rectangles) corresponding to potential events;
   c) upstream arrows (from places to transitions) validating transitions;
   d) downstream arrows (from transitions to places) indicating what happens when transitions are fired.
2) the scheduling part:
   a) stochastic delays representing random delays elapsing before events occur;
   b) deterministic delays representing known delays elapsing before events occur.
3) the dynamic part:
   a) tokens (small black circles) moving when events occur to indicate which of the potential states are actually achieved;
   b) predicates (any formula which may be true or false) validating transitions;
   c) assertions (any equation) updating some variables when a transition is fired.

In addition some rules allow validating and firing a transition:

4) validation of a transition (i.e. conditions for the corresponding event may arise):
   a) all upstream places have at least one token;
   b) all predicates must be "true".
5) firing of a transition(i.e. what happens when the corresponding event is arising):
   a) one token is removed from upstream places;
   b) one token is added in downstream places;
   c) assertions are updated.

Most of the notions in relationship with Petri nets are introduced above and the remaining ones will be introduced when needed.

B.5.3.2 Monte Carlo simulation principle

Monte Carlo simulation consists of the animation of behavioural models by using random numbers to evaluate how many times the system remains in states governed either by random or deterministic delays (see also B.6.6.8 of IEC 61508-7).

This can be explained by using the Petri net presented on Figure B.33:

– At the beginning the token is in place *W* and the component is in good working order.
– Only one event may arise from this state - a dangerous undetected failure- (transition *Tr1* is valid and drawn in black).
– The time spent in this state is stochastic and governed by an exponential distribution of parameter λ_{DU}. The Monte Carlo simulation consists of firing a random number (see below) to calculate the delay *d1* before the failure is going to occur (i.e. *Tr1* is going to be fired).
– When *d1* is elapsed, *Tr1* is fired and the token moves to place *DU* (more precisely one token is removed from place *W* and one token is added in place *DU*).
– The component reaches the dangerous undetected state and the transition *Tr2* becomes valid.
– The detection of the dangerous failure occurs after a deterministic delay *d2* (*d*2 = *t modulo* τ when *t* is the current time and τ the test interval). This simulates the test interval.

– When *t*2 is elapsed, i.e. the dangerous failure is detected, the token enters in place *DD.* The component is now waiting to be repaired and *Tr*3 becomes valid.

– The delay *d*3 for firing *Tr*3 (start of repair) does not depend on the component itself but on the availability of the repair resources represented by the message *RA*. This is governed by events arising in another part of the whole Petri net not represented on Figure B.33.

– The repair starts as soon as the repair team becomes available (i.e. the predicate *?RA = true* becomes true) and the token enters in place *R*. Repair resources become immediately unavailable for another intervention and the assertion *!RA = false* is used to update the value of *RA*. This prevents any other repair at the same time.

– The stochastic transition *Tr*4 (i.e. the end of the repair) becomes valid and the delay *d4* may be calculated by firing a random number according to the repair rate μ.

– When *d*4 is elapsed, *Tr*4 is fired and the component is coming back in its good working state (the token enters in place *W*). The repair resources are again available and *RA* is updated through the assertion *!RA = true*.

– And so on... as long as the firing of next valid transition belongs to the period under interest *[0, T]*.

When the next firing is no longer inside *[0, T]*, the simulation is stopped and one history of the component is achieved. All along the progress of the history, relevant parameters may be recorded as the mean marking of the places (i.e. the ratio of the time with one token in the place over the duration T), the transition firing frequencies, the time to the first occurrence of a given event, etc.

The principle of Monte Carlo simulation is to realize a great number of such histories and to perform classical statistics on the results in order to assess the relevant parameters.

Contrary to analytical calculations, Monte Carlo simulation allows to mix easily deterministic and random delays which may be simulated from their cumulated probability distribution *F(d)* and random numbers z_i uniformly distributed over *[0, 1]*. Those random numbers are available in almost any programming languages and powerful algorithms are available to do that.

Then, a sample (d_i), distributed according to *F(d),* is obtained from a sample (z_i) by the operation: $d_i = F^{-1}(z_i)$. This is very easy when the analytical form of $F^{-1}(z)$ is available as, for example, exponentially distributed delays: $d_i = -\frac{1}{\lambda_{DU}} Log(z_i)$.

In respect of the accuracy of the calculations, concerning a given simulated parameter *X*, the basic statistics allow the calculation of the average, variance, standard deviation and confidence of the sample (X_i) which has been simulated:

– average : $\overline{X} = \frac{\sum_i x_i}{N}$

– variance: $\sigma^2 = \frac{\sum_i (x_i - \overline{X})^2}{N}$ and standard deviation : σ

– 90 % confidence interval around $\overline{X}$: $Conf = 1{,}64 \frac{\sigma}{\sqrt{N}}$

Therefore, when using Monte Carlo simulation, the accuracy of the results can always be estimated. For example, considering 90 % of chance that the true result $\hat{X}$ belongs to the interval $\left[\overline{X} - 1{,}64\sigma / \sqrt{N},\quad \overline{X} + 1{,}64\sigma / \sqrt{N}\right]$.

This interval is reducing when the number of histories increases and when the frequency of occurrence of *X* increases.

With present time personal computers, there are no real difficulties to achieve calculations even for SIL 4 E/E/PE safety-related systems.

B.5.3.3 Principle of *PFD* calculations

The sub-Petri-net on Figure B.33 may be used directly to evaluate the PFD_{avg} of the component because the mean marking of place *W* which is equal to the ratio of the time spent in *W* (i.e. with *W* marked by the token) to the duration *T*, is in fact the average availability *A* of the component. We have $PFD_{avg} = 1 - A$.

The accuracy of the calculation may be estimated as explained above.

More complex behaviours may be represented by using specific sub-Petri Nets. Figure B.34 gives an idea of what can be done for modelling periodically tested components, common cause failures (CCF) and repair resources.

On the left hand side is modelled a periodically tested component which jumps across the states working (*W*), dangerously failed undetected (*DU*), under testing (DUT), dangerously failed detected (*DD*), ready for repair (*RR*) and under repair (*R*).

When it fails (*DU*), the message *!-Ci* (equivalent to *!Ci = false*) is emitted to inform that the component is failed. Then it waits until a periodical test is started (DUT). The periodic test interval is τ and the staggering θ. After what the test is performed for a duration equal to π and the state *DD* is reached. If a spare part is available (at least one token in *SP*), the component becomes ready for repair (*RR*) and the variable *NbR* is increased by 1 to inform the repair resources of the number of components needing to be repaired. When the repair resources are on location (one token in *OL*) the repair starts (*R*) and the token is removed from *OL*. When it is achieved the component comes back in good functioning state, the message *!Ci* is emitted (i.e. !*Ci* = *true*), *NbR* is decreased by 1 and the token is given back in *OL* to allow further repairs. And so on.

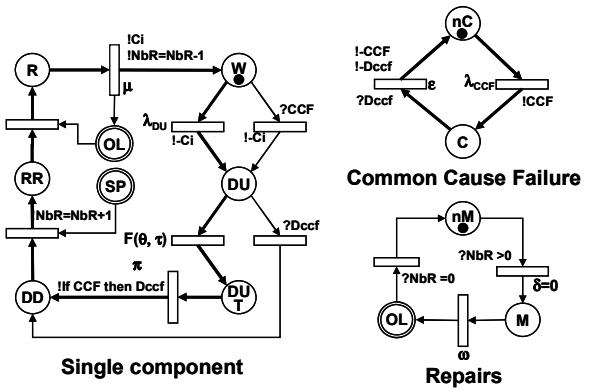


Figure B.34 – Petri net to model common cause failure and repair resources

The variable *NbR* is used by the sub-Petri net devoted to repairs. When it becomes positive, the mobilisation of the resources is started (*M*) and after a certain delay they are ready to work on location (*OL*). The token in *OL* is used to validate the starting of the repair of one failed component. Therefore, only one repair may be done at the same time. When all repairs have been performed (i.e. *NbR* = 0) the repair resources are demobilized.

On Figure B.34 a common cause failure (CCF) has also been modelled. When it occurs (λ_{DCC}), the message *!CCF* becomes true and is used to put all the affected components in their *DU* states. The relevant messages C_i become false and the components are repaired

independently from each other. When the test of a component is finished, the assertion *!IF CCF then Dccf* allows to inform all the other components that a CCF has been detected. This message is used to put them immediately in their *DD* states. This message is also used to reset the CCF sub-Petri net but this is done after a while (ε) to insure that all components have been put in their *DD* state before resetting.

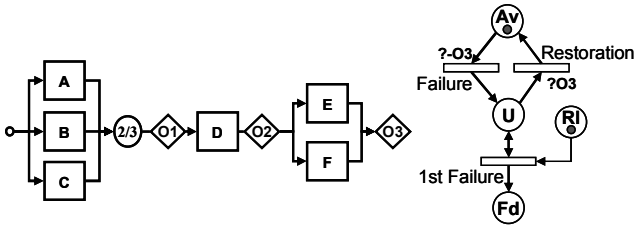


Figure B.35 – Using reliability block diagrams to build Petri net and auxiliary Petri net for *PFD* and *PFH* calculations

The sub-Petri Nets on Figure B.34 are intended to be used as parts of more complex models. One way to use them is illustrated on Figure B.35 where the reliability block diagram of Figure B.16 which has been slightly adapted by introducing the intermediary outputs *Oi*.

The components A, B, C, D, E, F may be modelled by a set of sub- Petri nets like those on Figure B.34 with for example a CCF for (A, B, C) and another one for (E, F) and the same repair resources for all components. The remaining problem is only to link the component together according to the logic of the reliability block diagram and to calculate the PFD_{avg} under interest.

Linking the components is very easy by using the messages C_i and building the following assertions:

– $O_1 = C_a.C_b + C_a.C_c + C_b.C_c$
– $O_2 = O_1.C_d$
– $O_3 = O_2.(C_e + C_f)$

Therefore when O_3 is true, the whole E/E/PE safety-related system is working well and it is unavailable otherwise. This message is used in the sub-Petri net on the right hand side in order to model the various states of the E/E/PE safety-related systems: available (*Av*), unavailable (*U*), reliable(*Rl*) and unreliable (*Fd*).

For *PFD* calculation the focus is only on *Av* and *U*: when O_3 becomes false, the system fails and becomes unavailable and when O_3 becomes true, the system is restored and becomes available again. This is very simple and the mean marking of *Av* is the average availability of the system and the mean marking of *U* its average unavailability, i.e. its PFD_{avg}.

Therefore, the Monte Carlo simulation performs automatically the integral of the instantaneous unavailability and it is not necessary to calculate it except if the saw tooth curve is wanted. This would be easily done by evaluating the mean marking of *U* at given instant rather over the whole *[0, T]* period.

What is exposed above only intends to illustrate the broad lines of using Petri nets for SIL calculations purposes but the potential modelling possibilities are virtually endless.

B.5.3.4 Principle of *PFH* calculations

For the *PFH* calculation, the principles remain exactly the same as above and the same sub models can be used for DU failures. Figure B.36 illustrates a sub-Petri net modelling a DD failure which is revealed and repaired as soon as it has been detected.

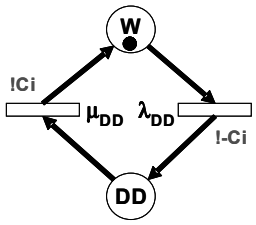


Figure B.36 – Simple Petri net for a single component with revealed failures and repairs

Such components' models may be used as explained above in relationship with a reliability block diagram representing the whole system like on Figure B.35.

When the E/E/PE safety-related system working in continuous mode is the ultimate safety barrier, an accident occurs as soon as it is failing and the *PFH* evaluation must be achieved through the reliability of the system. This is done by the lower part of the sub PN presented on the right hand side of Figure B.35: the average frequency of the first failure of the system over *[0, T]* is its unreliability *F(T)*. Then, when *F(T)* is small compared to 1, and according to the *PFH* definition, we have *PFH = F(T)/T*.

Due to the token in *RI*, the first failure is a one shot transition. Provided that all histories lead to a failure (i.e. T is long enough) the mean time spent with a token in *RI* is the *MTTF* of the system. Then *PFH* ≈ 1/*MTTF* provides an upper bound of *PFH*.

When the E/E/PE safety-related system working in continuous mode is not the ultimate safety barrier it does not provoke directly an accident when it fails. Then it is repaired after an overall failure and its *PFH* shall be calculated through the unavailability of the system. It is obtained directly from the frequency *Nbf* of the transition failure. This provides the number of times the system has failed over a given period and therefore we have *PFH(T) = Nbf/T*.

It is interesting to note that when *T* is long enough, the *MUT* may be calculated through the mean cumulated time MCT_{Av} in state *Av* and the *MDT* through the mean cumulated time MCT_U in state *U*. The mean cumulated times MCT_A and MCT_U are very easy to calculate within the Monte Carlo simulation just by cumulating the times when a token is present in *Av* or *U*. We have $MUT = MCT_A / Nbf$ and $MUT = MCT_u / Nbf$. This may be used for evaluating *PFH* = 1/*(MUT + MDT)* = 1/*MTBF = Nbf/T.*

All these results are obtained directly because the Monte Carlo simulation provides naturally the average values. What is exposed above only intends to illustrate the broad lines of using Petri nets for SIL calculations purposes but the potential modelling possibilities are virtually endless.

B.5.4 Other approaches

The relationship between the size of the models and the number of components of the system under study varies dramatically according to the type of approach in use. It is linear for fault trees and Petri nets but exponential for Markov processes. Therefore fault tree and Petri net approaches make it potentially easier to handle much larger systems than Markovian approach. This is why Petri nets are sometimes used to produce large Markov graphs.

The underlying formal languages behind the graphical representations described here above produce flat models: each component is described separately, at the same level. This makes large models sometimes hard to master and to maintain. A way to overcome this problem is to use structured languages providing compact hierarchical models. Several such formal languages have been developed recently and some software packages are available. As an example, we can consider, the AltaRica Data Flow language published in 2 000 in order to be freely used by the reliability community and designed to accurately model the functional and dysfunctional properties of industrial systems (see references in B.7).

Figure B.37 is equivalent to the reliability block diagram presented Figure B.1. This model is hierarchical because single modules are modelled only once and then reused as many times as needed at the system level (i.e. in the node main). This allows achieving very compact models.

In order to simplify the presentation, only two transitions have been represented for the components: failure and repair (i.e. DD failures revealed and repaired as soon as they arise).

Logical operators (*or*, *and*) are used to describe the logic of the system. This is done in direct relationship with the reliability block diagram and the flow Out models the state of the system: it is working when *Out* is *true* and failed when *Out* is *false*.

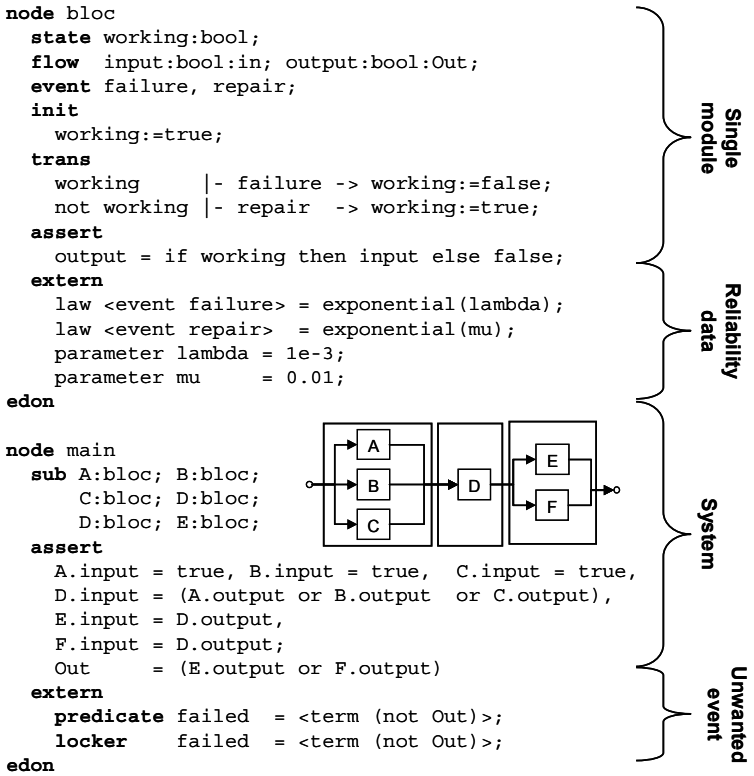


Figure B.37 – Example of functional and dysfunctional modelling with a formal language

This provides good behavioural models for efficient Monte Carlo simulation and all what has been described above for Petri net remains valid here for PFD_{avg} or *PFH* calculations. Therefore we are not developing this part further.

This formal language possesses similar mathematical properties as Petri nets and therefore it is possible to compile one model towards the other without difficulty. But it also generalizes the properties of the underlying languages behind Fault trees or Markov processes. Therefore, providing that the description has been restricted to the properties of Markov processes or Fault trees, it is possible to compile the model into equivalent Markov graph or fault trees. It is the purpose of the key words “predicate” and “locker” at the end of the model to provide directives for fault trees or Markov model generation or for direct Monte Carlo simulation.

Using a formal language designed to model properly the system behaviour both from functional and dysfunctional points of view allows:

– Monte Carlo simulations to be performed directly on the models;

– Markov graphs to be generated and analytical calculations to be performed as previously shown (when the language has been restricted to Markov properties);

– Equivalent Fault trees to be generated and analytical calculations to be performed as previously shown (when the language has been restricted to Boolean properties).

Such functional and dysfunctional formal languages are general purpose languages. There are no difficulties to use them in the particular case of E/E/PE safety-related systems. They provide an efficient way to master PFD_{avg} and *PFH* calculations of E/E/PE safety-related systems when several protection layers, various types of failure modes, complex proof test patterns, components dependencies, maintenance resources, etc., i.e. when the other methods have shown that they have reached their limits.

B.6 Handling uncertainties

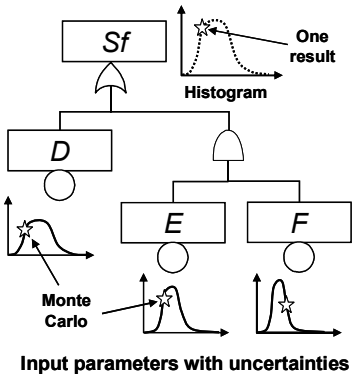


Figure B.38 – Uncertainty propagation principle

One of the main problems encountered when performing probabilistic calculations is linked to the uncertainties on the reliability parameters. Therefore, it is useful when performing *PFD* or *PFH* calculations to evaluate what the corresponding impact is on the uncertainty of the results.

Care needs to be taken when dealing with this issue and using Monte Carlo simulation provides an efficient way for this purpose as it is illustrated on Figure B.38.

On this figure the input reliability parameters (e.g. the dangerous undetected failure rates) are no longer certain and they are replaced by random variables. The density of probability of such random variable is more or less sharp or flat according to the degree of uncertainty: the probability density of F is sharper than the one of E or D. This means that there is less uncertainty on F than on E or D for example.

The principle of calculation is the following:

1) generating one set of input parameters by using random numbers according to the probabilistic distributions of those parameters (similar to what has been explained in B.3.2) ;
2) performing one calculation by using the above generated set of input parameters;
3) recording the output result (this constitutes one value used in step 4);
4) performing steps 1 to 3 again and again until a sufficient number of values (for example 100 or 1 000) is obtained in order to constitute an histogram (dotted line on Figure B.38);
5) analysing statistically the histogram to obtain the average and the standard deviation of the output result.

The average of this histogram is the PFD_{avg} or the *PFH* according to the type of calculation performed and the standard deviation measures the uncertainty on this results. The smaller the standard deviation, the more accurate is the PFD_{avg} or the *PFH* calculation.

The principle illustrated here on fault tree is very general and may be applied on any of the calculation methods depicted in this annex: simplified formulae, Markov processes and even Petri nets or formal languages approaches. When the calculation is already performed by Monte Carlo simulation, a two step Monte Carlo simulation shall be used.

The probabilistic distribution for a given input reliability parameter shall be chosen according to the knowledge gathered about it. This may be:

– a uniform distribution between lower and upper bounds;

– a triangular distribution with a most probable value;

– a log-normal law with a given error factor;

– a Chi square law, etc.

The first ones may be assessed by engineering judgement when there is not very much field feedback available. When there is more field feedback available, the last ones may be used because the field feedback provides average parameter values as well as confidence intervals on these average values.

For example if n failures have been observed over a cumulated observation time T, we have:

– $\hat{\lambda} = n/T$ is the maximum likelihood estimator of the failure rate

– $\lambda_{Inf,\alpha} = \frac{1}{2T}\chi^2_{(1-\alpha),2n}$ lower bound with a probability α % to be lower than $\lambda_{\text{Inf},\alpha}$

– $\lambda_{Sup,\alpha} = \frac{1}{2T}\chi^2_{\alpha,2(n+1)}$ upper bound with a probability α % to be higher than $\lambda_{\text{Sup},\alpha}$

When α = 5 %, then the true value of λ has 90 chances over 100 to belong to the interval $[\lambda_{\text{Inf},\alpha}, \lambda_{\text{Sup},\alpha}]$. The smaller this interval, the more accurate the value of λ. Normally, good reliability data bases provide this information. Analysts should consider reliability data provided without confidence intervals (or information allowing to calculate them) very cautiously.

$\hat{\lambda}$, $\lambda_{\text{Inf},\alpha}$ and $\lambda_{\text{Sup},\alpha}$ can be used to build a relevant distribution to model the failure rate λ of a given failure mode and its uncertainties. This is clear for χ^2 distribution but the log-normal law has also shown this to be very efficient for that purpose. Such a log-normal law is defined by its average $\hat{\lambda}$ or its median $\lambda_{50\ \%}$ and its so-called error factor.

This law has a very interesting property: $\lambda_{\text{Inf},\alpha} = \lambda_{50\ \%}/ef_\alpha$ and $\lambda_{\text{Sup},\alpha} = \lambda_{50\ \%}.ef_\alpha$.

Then it is defined with only two parameters: $\hat{\lambda}$ *and* $ef_\alpha \approx \sqrt{\frac{\lambda_{Sup,\alpha}}{\lambda_{Inf,\alpha}}}$

When ef_α = 1 there is no uncertainties, when ef_α = 3,3 there is a factor of about 10 between the lower and upper confidence bounds, etc.

These laws may be used in turn with Monte Carlo simulations in order to take under consideration both the impacts of average values and uncertainties on PFD_{avg} or *PFH*. Therefore it is always possible to master the uncertainties within probabilistic calculations. Some software packages implement directly such calculations.

When analysing redundant systems the analysis should not only consider the uncertainty of the basic element failure rate but also the accuracy of the CCF failure rate. Even if there is good field feed back data for each of the elements there is rarely good CCF field data and hence this will be the most uncertain.

B.7 References

References [4] to [9] and [22] to [24] in the Bibliography give further details on evaluating probabilities of failure.

Annex C
(informative)

Calculation of diagnostic coverage and safe failure fraction – worked example

A method for calculating diagnostic coverage and safe failure fraction is given in Annex C of IEC 61508-2. This annex briefly describes the use of this method to calculate the diagnostic coverage. It is assumed that all of the information specified in IEC 61508-2 is available and has been used where required in obtaining the values shown in Table C.1. Table C.2 gives limitations on diagnostic coverage that can be claimed for certain E/E/PE safety-related elements. The values in Table C.2 are based on engineering judgement.

To understand all the values in Table C.1, a detailed hardware schematic would be required, from which the effect of all failure modes could be determined. These values are only examples, for instance some components in Table C.1 assume no diagnostic coverage because it is practically impossible to detect all failure modes of all components.

Table C.1 has been derived as follows:

a) A failure mode and effect analysis has been carried out to determine the effect of each failure mode for every component on the behaviour of the system without diagnostic tests. The fractions of the overall failure rate associated with each failure mode are shown for each component, divided between safe (S) and dangerous (D) failures. The division between safe and dangerous failures may be deterministic for simple components but is otherwise based on engineering judgement. For complex components, where a detailed analysis of each failure mode is not possible, a division of failures into 50 % safe, 50 % dangerous is generally accepted. For this table, the failure modes given in reference a) have been used, although other divisions between failure modes are possible and may be preferable.

b) The diagnostic coverage for each specific diagnostic test on each component is given (in the column labelled "DC_{comp}"). Specific diagnostic coverages are given for the detection of both safe and dangerous failures. Although open-circuit or short-circuit failures for simple components (for example resistors, capacitors and transistors) are shown to be detected with a specific diagnostic coverage of 100 %, the use of table C.2 has limited the diagnostic coverage with respect to item U16, a complex type B component, to 90 %.

c) Columns (1) and (2) give the safe and dangerous failure rates, in the absence of diagnostic tests, for each component (λ_S and $\lambda_{DD} + \lambda_{Du}$ respectively).

d) We can consider a detected dangerous failure to be effectively a safe failure, so we now find the division between effectively safe failures (i.e. either detected safe, undetected safe or detected dangerous failures) and undetected dangerous failures. The effective safe failure rate is found by multiplying the dangerous failure rate by the specific diagnostic coverage for dangerous failures and adding the result to the safe failure rate (see column (3)). Likewise, the undetected dangerous failure rate is found by subtracting the specific diagnostic coverage for dangerous failures from one and multiplying the result by the dangerous failure rate (see column (4)).

e) Column (5) gives the detected safe failure rate and column (6) gives the detected dangerous failure rate, found by multiplying the specific diagnostic coverage by the safe and dangerous failure rates respectively.

f) The table yields the following results:

   – total safe failure rate $\sum \lambda_S + \sum \lambda_{Dd} = 9{,}9 \times 10^{-7}$
     (including detected dangerous failures)

   – total undetected dangerous failure rate $\sum \lambda_{Du} = 5{,}1 \times 10^{-8}$

- total failure rate $\sum \lambda_S + \sum \lambda_{Dd} + \sum \lambda_{Du}$ = 1,0 × 10^{-6}
- total undetected safe failure rate $\sum \lambda_{Su}$ = 2,7 × 10^{-8}
- diagnostic coverage for safe failures $\frac{\sum \lambda_{Sd}}{\sum \lambda_S} = \frac{3{,}38}{3{,}65}$ = 93 %
- diagnostic coverage for dangerous failures
  $\frac{\sum \lambda_{Dd}}{\sum \lambda_{Dd} + \sum \lambda_{Du}} = \frac{6{,}21}{6{,}72}$ = 92 % (normally termed simply “diagnostic coverage”)
- safe failure fraction $\frac{\sum \lambda_S + \sum \lambda_{Dd}}{\sum \lambda_S + \sum \lambda_{Dd} + \sum \lambda_{Du}} = \frac{986}{365 + 672}$ = 95 %

g) The division of the failure rate without diagnostic tests is 35 % safe failures and 65 % dangerous failures.

Table C.1 – Example calculations for diagnostic coverage and safe failure fraction

Item	No	Type	Division of safe and dangerous failures for each failure mode								Division of safe and dangerous failures for diagnostic coverage and calculated failure rates ($\times 10^{-9}$)						
			OC		SC		Drift		Function		DC_{comp}		(1)	(2)	(3)	(4)	(5)	(6)
			S	D	S	D	S	D	S	D	S	D	λ_S	$\lambda_{DD}+\lambda_{Du}$	$\lambda_S+\lambda_{DD}$	λ_{DU}	λ_{SD}	λ_{DD}
Print	1	Print	0,5	0,5	0,5	0,5	0	0	0	0	0,99	0,99	11,0	11,0	21,9	0,1	10,9	10,9
CN1	1	Con96pin	0,5	0,5	0,5	0,5					0,99	0,99	11,5	11,5	22,9	0,1	11,4	11,4
C1	1	100nF	1	0	1	0	0	0	0	0	1	0	3,2	0,0	3,2	0,0	3,2	0,0
C2	1	10µF	0	0	1	0	0	0	0	0	1	0	0,8	0,0	0,8	0,0	0,8	0,0
R4	1	1M	0,5	0,5	0,5	0,5					1	1	1,7	1,7	3,3	0,0	1,7	1,7
R6	1	100k									0	0	0,0	0,0	0,0	0,0	0,0	0,0
OSC1	1	OSC24 MHz	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	1	1	16,0	16,0	32,0	0,0	16,0	16,0
U8	1	74HCT85	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,99	0,99	22,8	22,8	45,4	0,2	22,6	22,6
U16	1	MC68000-12	0	1	0	1	0,5	0,5	0,5	0,5	0,90	0,90	260,4	483,6	695,6	48,4	234,4	435,2
U26	1	74HCT74	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,99	0,99	22,8	22,8	45,4	0,2	22,6	22,6
U27	1	74F74	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,99	0,99	14,4	14,4	28,7	0,1	14,3	14,3
U28	1	PAL16L8A	0	1	0	1	0	1	0	1	0,98	0,98	0,0	88,0	86,2	1,8	0,0	86,2
T1	1	BC817	0	0	0	0,67	0	0,5	0	0	1	1	0,0	0,2	0,4	0,0	0,0	0,2
Total													365	672	986	50,9	338	621

NOTE None of the failure modes of item R6 are detected, but a failure does not affect either safety or availability.

Key

S	Safe failure
D	Dangerous failure
OC	Open circuit
SC	Short circuit
Drift	Change of value
Function	Functional failures
DC_{comp}	Specific diagnostic coverage for the component

See also Table B.1, although in this table failure rates are for the individual components in question rather than every component in a channel.

Table C.2 – Diagnostic coverage and effectiveness for different elements

Component	Low diagnostic coverage	Medium diagnostic coverage	High diagnostic coverage
CPU (see Note 3)	total less than 70 %	total less than 90 %	
register, internal RAM	50 % - 70 %	85 % - 90 %	99 % - 99,99 %
coding and execution including flag register	50 % - 60 %	75 % - 95 %	-
(see Note 3)	50 % - 70 %	85 % - 98 %	-
address calculation (see Note 3)	50 % - 60 %	60 % - 90 %	85 % - 98 %
program counter, stack pointer	50 % - 70 %		
	40 % - 60 %		
Bus			
memory management unit	50 %	70 %	90 % - 99 %
bus-arbitration	50 %	70 %	90 % - 99 %
Interrupt handling	40 % - 60 %	60 % - 90 %	85 % - 98 %
Clock (quartz) (see Note 4)	50 %	-	95 % - 99 %
Program flow monitoring			
temporal (see Note 3)	40 % - 60 %	60 % - 80 %	-
logical (see Note 3)	40 % - 60 %	60 % - 90 %	-
temporal and logical (see Note 5)	-	65 % - 90 %	90 % - 98 %
Invariable memory	50 % - 70 %	99 %	99,99 %
Variable memory	50 % - 70 %	85 % - 90 %	99 % - 99,99 %
Discrete hardware			
digital I/O	70 %	90 %	99 %
analogue I/O	50 % - 60 %	70 % - 85 %	99 %
power supply	50 % - 60 %	70 % - 85 %	99 %
Communication and mass storage	90 %	99,9 %	99,99 %
Electromechanical devices	90 %	99 %	99,9 %
Sensors	50 % - 70 %	70 % - 85 %	99 %
Final elements	50 % - 70 %	70 % - 85 %	99 %

NOTE 1 This table should be read in conjunction with Table A.1 of IEC 61508-2 which provides the failure modes to be considered.

NOTE 2 When a range is given for diagnostic coverage, the upper interval boundaries may be set only for narrowly tolerated monitoring means, or for test measures that stress the function to be tested in a highly dynamic manner.

NOTE 3 For techniques where there is no high diagnostic coverage figure, at present no measures and techniques of high effectiveness are known.

NOTE 4 At present no measures and techniques of medium effectiveness are known for quartz clocks.

NOTE 5 The minimum diagnostic coverage for a combination of temporal and logical program flow monitoring is medium.

See references [10] to [12] in the Bibliography.

Annex D
(informative)

A methodology for quantifying the effect of hardware-related common cause failures in E/E/PE systems

D.1 General

D.1.1 Introduction

This standard incorporates a number of measures which deal with systematic failures. However, no matter how well these measures are applied, there is a residual probability of systematic failures occurring. Although this does not significantly affect the reliability calculations for single-channel systems, the potential for failures which may affect more than one channel in a multi-channel system (or several components in a redundant safety system), i.e. common cause failures, results in substantial errors when reliability calculations are applied to multi-channel or redundant systems.

This informative annex describes two methodologies which allow common cause failures to be taken into account in the safety assessment of multi-channel or redundant E/E/PE systems. Using these methodologies gives a more accurate estimation of the integrity of such a system than ignoring the potential for common cause failures.

The first methodology is used to calculate a value for β, factor frequently used in the modelling of common cause failures. This can be used to estimate the rate of common cause failures applicable to two or more systems operating in parallel from the random hardware failure rate of one of those systems (see D.5). It is generally accepted that the random hardware failure figures that are collected will include a number of failures that were caused by systematic failures.

Alternative methodologies may be preferred in some cases, for example, where a more accurate β-factor can be proven as a result of the availability of data on common cause failures or when the number of impacted elements is higher than four. The second methodology, i.e. the binomial failure rate (also called shock model) method, can be used

D.1.2 Brief overview

The failures of a system are considered to arise from two dissimilar sources:

– random hardware failures; and

– systematic failures.

The former are assumed to occur randomly in time for any component and to result in a failure of a channel within a system of which the component forms part when the latter appears immediately and in a deterministic way when the system reach the situation for which the underlying systematic error is existing.

There is a finite probability that independent random hardware failures could occur in all channels of a multi-channel system so that all of the channels were simultaneously in a failed state. Because random hardware failures are assumed to occur randomly with time, the probability of such failures concurrently affecting parallel channels is low compared to the probability of a single channel failing. This probability can be calculated using well-established techniques but the result may be very optimistic when the failures are not fully independent from each other.

Dependent failures are traditionally split between (see reference [18] in the Bibliography):

– Common Cause Failure (CCF) causing multiple failures from a single shared cause. The multiple failures may occur simultaneous or over a period of time;

– Common Mode Failures (CMF) which are a particular case of CCF in which multiple equipment items fail in the same mode;

– Cascade failures which are propagating failures.

The term CCF is often used to cover all kind of dependant failures as it is done in this annex. They are also split between

– Dependent failures due to clear deterministic causes;

– Residual potential multiple failure events not explicitly considered in the analysis because of not enough accuracy, no clear deterministic causes or impossibility to gather reliability data.

The first one should be analysed, modelled and quantified in a conventional way and only the second ones should be handled as shown in this informative Annex D. Nevertheless, the systematic failures -which are perfect dependent failures not identified during the safety analysis (otherwise they would have been removed)- are handled in particular manner in this standard and this annex applies mainly for hardware random dependent failures.

Therefore, common cause failures which result from a single cause, may affect more than one channel or more than one component. These may result from a systematic fault (for example, a design or specification mistake) or an external stress leading to an early random hardware failure (for example, an excessive temperature resulting from the random hardware failure of a common cooling fan, which accelerates the life of the components or takes them outside their specified operating environment) or, possibly, a combination of both. Because common cause failures are likely to affect more than one channel in a multi-channel system, the probability of common cause failure is likely to be the dominant factor in determining the overall probability of failure of a multi-channel system and if this is not taken into account a realistic estimate of the safety integrity level of the combined system is unlikely to be obtained.

D.1.3 Defence against common cause failures

Although common cause failures result from a single cause, they do not all manifest themselves necessarily simultaneously in all channels. For example, if a cooling fan fails, all of the channels of a multi-channel E/E/PE system could fail, leading to a common cause failure. However, all of the channels are unlikely to warm at the same rate or to have the same critical temperature. Therefore, failures occur at different times in the different channels.

The architecture of programmable systems allows them to carry out internal diagnostic testing functions during their on-line operation. These can be employed in a number of ways, for example

– a single channel PE system can continuously be checking its internal operation together with the functionality of the input and output devices. If designed from the outset, a test coverage in the region of 99 % is achievable (see [13] in the Bibliography). If 99 % of internal faults are revealed before they can result in a failure, the probability of single-channel faults which can ultimately contribute to common cause failures is significantly reduced.

– in addition to internal testing, each channel in a PE system can monitor the outputs of other channels in a multi-channel PE system (or each PE device can monitor another PE device in a multi-PE system). Therefore, if a failure occurs in one channel, this can be detected and a safe shut-down initiated by the one or more remaining channels that have not failed and are executing the cross-monitoring test. (It should be noted that cross-monitoring is effective only when the state of the control system is continuously changing, for example the interlock of a frequently used guard in a cyclic machine, or when brief changes can be introduced without affecting the controlled function.) This cross-

monitoring can be carried out at a high rate, so that, just before a non-simultaneous common cause failure, a cross-monitoring test is likely to detect the failure of the first channel to fail and is able to put the system into a safe state before a second channel is affected.

In the case of the cooling fan example, the rate of temperature rise and the susceptibility of each channel are slightly different, resulting in the second channel failing possibly several tens of minutes after the first. This allows the diagnostic testing to initiate a safe shutdown before the second channel succumbs to the common cause fault.

As a result of the above

– PE-based systems have the potential to incorporate defences against common cause failures and, therefore, be less susceptible to them when compared to other technologies;

– a different β-factor may be applicable to PE-based systems when compared to other technologies. Therefore, β-factor estimates based on historic data are likely to be invalid. (None of the existing investigated models used for estimating the probability of common cause failure allow for the effect of automatic cross-monitoring.)

– because common cause failures that are distributed in time may be revealed by the diagnostic tests before they affect all channels, such failures may not be recognized or reported as being common cause failures.

There are three avenues that can be taken to reduce the probability of potentially dangerous common cause failures.

a) Reduce the number of random hardware and systematic failures overall. (This reduces the areas of the ellipses in Figure D.1 leading to a reduction in the area of overlap.)

b) Maximize the independence of the channels (separation and diversity). (This reduces the amount of overlap between the ellipses in Figure D.1 whilst maintaining their area.)

c) Reveal non-simultaneous common cause failures while only one, and before a second, channel has been affected, i.e. use diagnostic tests or proof test staggering.

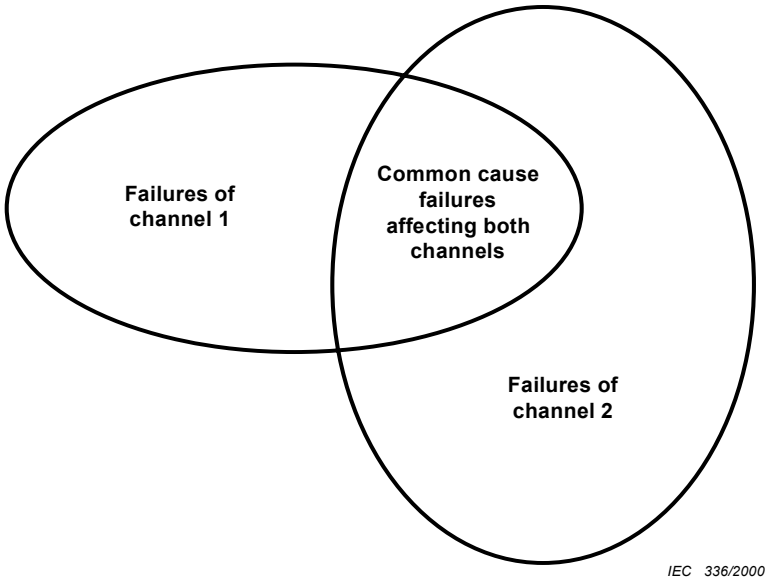


Figure D.1 – Relationship of common cause failures to the failures of individual channels

With systems with more than two channels, the common cause failure may affect all channels or only multiple channels but not all depending on the source of the common mode. Thus the approach taken in this annex, according to the first method, is to calculate the β value based on a duplex system voting 1oo2 and then use a multiplying factor to the derived β depending on the total number of channels and the voting requirements. (see Table D.5).

D.1.4 Approach adopted in the IEC 61508 series

The IEC 61508 series is based on these three avenues and requires a threefold approach:

a) Apply the techniques specified in IEC 61508-2/3 to reduce the overall probability of systematic failure to a level commensurate with the probability of random hardware failure.

b) Quantify those factors that can be quantified, i.e. take into account the probability of random hardware failure, as specified in IEC 61508-2.

c) Derive, by what is considered at the present time to be the best practicable means, a factor relating the probability of common cause failure of the hardware to the probability of random hardware failure. The methodology described in this annex relates to the derivation of this factor.

Most methodologies for estimating the probability of common cause failures attempt to make their predictions from the probability of random hardware failure. Clearly, the justification for any direct relationship between these probabilities is tenuous, nevertheless, such a correlation has been found in practice and probably results from second-order effects. For example, the higher the probability of random hardware failure of a system

– the higher the amount of maintenance required by the system. The probability of a systematic fault being introduced during maintenance depends on the number of times maintenance is carried out, and this also affects the rate of human errors leading to common cause failures. This leads to a relationship between the probability of random hardware failure and the probability of common cause failure. For example:
  - a repair, followed by testing and, possibly, recalibration is required each time a random hardware failure occurs;
  - for a given safety integrity level, a system with a higher probability of random hardware failure requires proof tests to be carried out more frequently and with greater depth/complexity, leading to additional human interference.
– the more complex the system. The probability of random hardware failure depends on the number of components, and, hence, the complexity of a system. A complex system is less easily understood, so is more prone to the introduction of systematic faults. In addition, the complexity makes it difficult to detect the faults, by either analysis or test, and can lead to parts of the logic of a system not being exercised except in infrequent circumstances. Again, this leads to a relationship between the probability of random hardware failure and the probability of common cause failure.

Several approaches are currently in use to handle CCF (β factor, multiple Greek letters, α factor, binomial failure rate ...) [20]. Two of the current models are proposed in this informative annex for the third part of the threefold approach already described. Despite the limitations, it is believed that they represent the best way forward at the present time to handle the probability of common cause failure:

– the well-established β-factor model which is widely used and realistic to deal with multi-channel system typically up to four dependent elements.
– the binomial failure rate [21] (also known as the shock model) which can be used when the number of dependent elements is greater than four.

The following two difficulties are faced when using the β-factor or the shock models on a E/E/PE system.

– What value should be chosen for the parameters? Many sources (for example reference [13]) suggest ranges within which the value of the β-factor is likely to occur but no actual value is given, leaving the user to make a subjective choice. To overcome this problem, the β-factor methodology in this annex is based on the system originally described in reference [14] and recently redefined in reference [15].

– Neither the β-factor nor the shock models take into account the sophisticated diagnostic testing capabilities of modern PE systems, which can be used to detect a non-simultaneous common cause failure before it has had sufficient time to manifest itself fully. To overcome this deficiency, the approach described in references [14] and [15] has been modified to reflect the effect of diagnostic tests in the estimation of the likely value of β.

The diagnostic testing functions running within a PE system are continuously comparing the operation of the PE system with predefined states. These states can be predefined in software or in hardware (for example, by a watchdog timer). Looked on in this way, the diagnostic testing functions may be thought of as an additional, and partially diverse, channel running in parallel with the PE system.

Cross-monitoring between channels also can be carried out. For many years, this technique has been used in dual-channel interlocking systems based solely on relays. However, with relay technology, it is usually possible to carry out the cross-checks only when the channels change state, making such tests inappropriate for revealing non-simultaneous common cause failures where systems remain in the same (for example, ON) state for long periods. With PE system technology, cross-monitoring may be carried out with a high repetition frequency.

D.2 Scope of the methodology

The scope of the methodology is limited to common cause failures within hardware. The reasons for this include the following:

– the β-factor and shock models relate the probability of common cause failure to the probability of random hardware failure. The probability of common cause failures which involve the system as a whole depends on the complexity of the system (possibly dominated by the user software) and not on the hardware alone. Clearly, any calculations based on the probability of random hardware failure cannot take into account the complexity of the software;

– reporting of common cause failures is generally limited to hardware failures, the area of most concern to the manufacturers of the hardware;

– it is not considered practicable to model systematic failures (for example software failures);

– the measures specified in IEC 61508-3 are intended to reduce the probability of software-related common cause failure to an acceptable level for the target safety integrity level.

Therefore, the estimate of the probability of common cause failure derived by this methodology relates to only those failures associated with the hardware. It should NOT be assumed that the methodology can be used to obtain an overall failure rate which takes the probability of software-related failure into account.

D.3 Points taken into account in the methodology

Because sensors, logic subsystem and final elements are subject to, for example, different environmental conditions and diagnostic tests with varying levels of capability, the methodology should be applied to each of these subsystems separately. For example, the logic subsystem is more likely to be in a controlled environment, whereas the sensors may be mounted outside on pipework that is exposed to the elements.

Programmable electronic channels have the potential for carrying out sophisticated diagnostic testing functions. These can

– have a high diagnostic coverage within the channels;
– monitor additional redundancy channels;
– have a high repetition rate; and
– in an increasing number of cases, also monitor sensors and/or final elements.

A large fraction of common cause failures do not occur concurrently in all of the affected channels. Therefore, if the repetition frequency of the diagnostic tests is sufficiently high, a large fraction of common cause failures can be revealed and, hence, avoided before they affect all available channels.

Not all features of a multi-channel system, that has a bearing on its immunity to common cause failures, can be evaluated by diagnostic tests. However, those features relating to diversity or independence are made more effective. Any feature which is likely to increase the time between channel failures in a non-simultaneous common cause failure (or reduce the fraction of simultaneous common cause failures) increases the probability of the diagnostic tests detecting the failure and putting the plant into a safe state. Therefore, the features relating to immunity to common cause failures are divided into those whose effect is thought to be increased by the use of diagnostic tests and those whose effect is not. This leads to the two columns, X and Y respectively, in Table D.1.

Although, for a three-channel system, the probability of common cause failures which affect all three channels is likely to be slightly lower than the probability of failures which affect two channels, it is assumed, in order to simplify the β-factor methodology, that the probability is independent of the number of affected channels, i.e. it is assumed that if a common cause failure occurs it affects all channels. An alternative method is the shock model.

There is no known data on hardware-related common cause failures available for the calibration of the methodology. Therefore, the tables in this annex are based on engineering judgement.

Diagnostic test routines are sometimes not regarded as having a direct safety role so may not receive the same level of quality assurance as the routines providing the main control functions. The methodology was developed on the presumption that the diagnostic tests have an integrity commensurate with the target safety integrity level. Therefore, any software-based diagnostic test routines should be developed using techniques appropriate to the target safety integrity level.

D.4 Using the β-factor to calculate the probability of failure in an E/E/PE safety-related system due to common cause failures

Consider the effect of common cause failures on a multi-channel system with diagnostic tests running within each of its channels.

Using the β-factor model, the common cause dangerous failure rate is:

$$\lambda_D \beta$$

where

λ_D is the random hardware dangerous failure rate for each individual channel and β is the β-factor in the absence of diagnostic tests, i.e. the fraction of single-channel failures that affect all channels.

We now assume that common cause failures affect all channels, and that the span of time between the first channel and all channels being affected is small compared to the time interval between successive common cause failures.

Suppose that there are diagnostic tests running in each channel which detect and reveal a fraction of the failures. We can divide all failures into two categories: those that lie outside the coverage of the diagnostic tests (and so can never be detected) and those that lie within the coverage (so would eventually be detected by the diagnostic tests).

The overall failure rate due to dangerous common cause failures is then given by:

$$\lambda_{DU}\beta + \lambda_{DD}\beta_D$$

where

- λ_{DU} is the undetected dangerous failure rate of a single channel, i.e. the failure rate of the failures which lie outside the coverage of the diagnostic tests; clearly, any reduction in the β-factor resulting from the repetition rate of the diagnostic tests cannot affect this fraction of the failures;
- β is the common cause failure factor for undetectable dangerous faults, which is equal to the overall β-factor that would be applicable in the absence of diagnostic testing;
- λ_{DD} is the detected dangerous failure rate of a single channel, i.e. the failure rate of the failures of a single channel that lie within the coverage of the diagnostic tests. If the repetition rate of the diagnostic tests is high, a fraction of the failures are revealed leading to a reduction in the value of β, i.e. β_D;
- β_D is the common cause failure factor for detectable dangerous faults. As the repetition rate of the diagnostic testing is increased, the value of β_D falls increasingly below β;
- β is obtained from Table D.5 which uses the results of D.4, using a score, $S = X + Y$ (see D.5);
- β_D is obtained from Table D.5 which uses the results of D.4, using a score, $S_D = X(Z+1)+Y$.

D.5 Using the tables to estimate β

The β-factor should be calculated for the sensors, the logic subsystem and the final elements separately.

In order to minimize the probability of occurrence of common cause failures, one should first establish which measures lead to an efficient defence against their occurrence. The implementation of the appropriate measures in the system lead to a reduction in the value of the β-factor used in estimating the probability of failure due to common cause failures.

Table D.1 lists the measures and contains associated values, based on engineering judgement, which represent the contribution each measure makes in the reduction of common cause failures. Because sensors and final elements are treated differently to the programmable electronics, separate columns are used in the table for scoring the programmable electronics and the sensors or final elements.

Extensive diagnostic tests may be incorporated into programmable electronic systems which allow the detection of non-simultaneous common cause failures. To allow diagnostic tests to be taken into account in the estimation of the β-factor, the overall contribution of each measure in Table D.1 is divided, using engineering judgement, into two sets of values, X and Y. For each measure, the X:Y ratio represents the extent to which the measure's contribution against common clause failures can be improved by diagnostic testing.

The user of Table D.1 should ascertain which measures apply to the system in question, and sum the corresponding values shown in each of columns X_{LS} and Y_{LS} for the logic subsystem,

or X_{SF} and Y_{SF} for the sensors or final elements, the sums being referred to as X and Y, respectively.

Tables D.2 and D.3 may be used to determine a factor Z from the frequency and coverage of the diagnostic tests, taking into account the important Note 4 which limits when a non-zero value of Z should be used. The score S is then calculated using the following equations, as appropriate (see previous clause):

– $S = X + Y$ to obtain the value of β_{int} (the β-factor for undetected failures); and

– $S_D = X(Z + 1) + Y$ to obtain the value of $\beta_{D\ int}$ (the β-factor for detected failures).

Here S or S_D is a score which is used in Table D.4 to determine the appropriate β_{int} -factor.

β_{int} and $\beta_{D\ int}$ are the values of the common cause failure prior to considering the effect of different degrees of redundancy.

Table D.1 – Scoring programmable electronics or sensors/final elements

Item	Logic subsystem		Sensors and final elements	
	X_{LS}	Y_{LS}	X_{SF}	Y_{SF}
Separation/segregation				
Are all signal cables for the channels routed separately at all positions?	1,5	1,5	1,0	2,0
Are the logic subsystem channels on separate printed-circuit boards?	3,0	1,0		
Are the logic subsystems physically separated in an effective manner? For example, in separate cabinets.	2,5	0,5		
If the sensors/final elements have dedicated control electronics, is the electronics for each channel on separate printed-circuit boards?			2,5	1,5
If the sensors/final elements have dedicated control electronics, is the electronics for each channel indoors and in separate cabinets?			2,5	0,5
Diversity/redundancy				
Do the channels employ different electrical technologies for example, one electronic or programmable electronic and the other relay?	8,0			
Do the channels employ different electronic technologies for example, one electronic, the other programmable electronic?	6,0			
Do the devices employ different physical principles for the sensing elements for example, pressure and temperature, vane anemometer and Doppler transducer, etc?			9,0	
Do the devices employ different electrical principles/designs for example, digital and analogue, different manufacturer (not re-badged) or different technology?			6,5	
Is low diversity used, for example hardware diagnostic tests using the same technology?	2,0	1,0		
Is medium diversity used, for example hardware diagnostic tests using different technology?	3,0	2,0		
Were the channels designed by different designers with no communication between them during the design activities?	1,5	1,5		
Are separate test methods and people used for each channel during commissioning?	1,0	0,5	1,0	2,0
Is maintenance on each channel carried out by different people at different times?	3,0		3,0	
Complexity/design/application/maturity/experience				
Does cross-connection between channels preclude the exchange of any information other than that used for diagnostic testing or voting purposes?	0,5	0,5	0,5	0,5
Is the design based on techniques used in equipment that has been used successfully in the field for > 5 years?	0,5	1,0	1,0	1,0
Is there more than 5 years experience with the same hardware used in similar environments?	1,0	1,5	1,5	1,5
Is the system simple, for example no more than 10 inputs or outputs per channel?		1,0		
Are inputs and outputs protected from potential levels of over-voltage and over-current?	1,5	0,5	1,5	0,5
Are all devices/components conservatively rated (for example, by a factor of 2 or more)?	2,0		2,0	
Assessment/analysis and feedback of data				
Have the results of the failure modes and effects analysis or fault-tree analysis been examined to establish sources of common cause failure and have predetermined sources of common cause failure been eliminated by design?		3,0		3,0
Were common cause failures considered in design reviews with the results fed back into the design? (Documentary evidence of the design review activity is required.)		3,0		3,0
Are all field failures fully analyzed with feedback into the design? (Documentary evidence of the procedure is required.)	0,5	3,5	0,5	3,5
Procedures/human interface				
Is there a written system of work to ensure that all component failures (or degradations) are detected, the root causes established and other similar items inspected for similar potential causes of failure?		1,5	0,5	1,5
Are procedures in place to ensure that: maintenance (including adjustment or calibration) of any part of the independent channels is staggered, and, in addition to the manual checks carried out following maintenance, the diagnostic tests are allowed to run satisfactorily between the completion of maintenance on one channel and the start of maintenance on another?	1,5	0,5	2,0	1,0
Do the documented maintenance procedures specify that all parts of redundant systems (for example, cables, etc.) intended to be independent of each other, are not to be relocated?	0,5	0,5	0,5	0,5
Is all maintenance of printed-circuit boards, etc. carried out off-site at a qualified repair centre and have all the repaired items gone through a full pre-installation testing?	0,5	1,0	0,5	1,5
Does the system have low diagnostic coverage (60 % to 90 %) and report failures to the level of a field-replaceable module?	0,5			
Does the system have medium diagnostics coverage (90 % to 99 %) and report failures to the level of a field-replaceable module?	1,5	1,0		
Does the system have high diagnostics coverage (>99 %) and report failures to the level of a field-replaceable module?	2,5	1,5		
Do the system diagnostic tests report failures to the level of a field-replaceable module?			1,0	1,0

Table D.1 *(continued)*

Item	Logic subsystem		Sensors and final elements	
	X_{LS}	Y_{LS}	X_{SF}	Y_{SF}
Competence/training/safety culture				
Have designers been trained (with training documentation) to understand the causes and consequences of common cause failures?	2,0	3,0	2,0	3,0
Have maintainers been trained (with training documentation) to understand the causes and consequences of common cause failures?	0,5	4,5	0,5	4,5
Environmental control				
Is personnel access limited (for example locked cabinets, inaccessible position)?	0,5	2,5	0,5	2,5
Is the system likely to operate always within the range of temperature, humidity, corrosion, dust, vibration, etc., over which it has been tested, without the use of external environmental control?	3,0	1,0	3,0	1,0
Are all signal and power cables separate at all positions?	2,0	1,0	2,0	1,0
Environmental testing				
Has the system been tested for immunity to all relevant environmental influences (for example EMC, temperature, vibration, shock, humidity) to an appropriate level as specified in recognized standards?	10,0	10,0	10,0	10,0

NOTE 1 A number of the items relate to the operation of the system, which may be difficult to predict at design time. In these cases, the designers should make reasonable assumptions and subsequently ensure that the eventual user of the system is made aware of, for example, the procedures to be put in place in order to achieve the designed level of safety integrity. This could be by including the necessary information in the accompanying documentation.

NOTE 2 The values in the *X* and *Y* columns are based on engineering judgement and take into account the indirect as well as the direct effects of the items in column 1. For example, the use of field-replaceable modules leads to

– repairs being carried out by the manufacturer under controlled conditions instead of (possibly incorrect) repairs being made under less appropriate conditions in the field. This leads to a contribution in the *Y* column because the potential for systematic (and, hence, common cause) failures is reduced;

– a reduction in the need for on-site manual interaction and the ability quickly to replace faulty modules, possibly on-line, so increasing the efficacy of the diagnostics for identifying failures before they become common-cause failures. This leads to a strong entry in the *X* column.

Table D.2 – Value of *Z* – programmable electronics

Diagnostic coverage	Diagnostic test interval		
	Less than 1 min	Between 1 min and 5 min	Greater than 5 min
≥ 99 %	2,0	1,0	0
≥ 90 %	1,5	0,5	0
≥ 60 %	1,0	0	0

Table D.3 – Value of *Z* – sensors or final elements

Diagnostic coverage	Diagnostic test interval			
	Less than 2 h	Between 2 h and two days	Between two days and one week	Greater than one week
≥ 99 %	2,0	1,5	1,0	0
≥ 90 %	1,5	1,0	0,5	0
≥ 60 %	1,0	0,5	0	0

NOTE 1 The methodology is most effective if account is taken uniformly across the list of the categories in Table D.1. Therefore, it is strongly recommended that the total score in the *X* and *Y* columns for each category should be not less than the total score in the *X* and *Y* columns divided by 20. For example, if the total score ($X + Y$) is 80, none of the categories (for example, procedures/human interface) should have a total score ($X + Y$) of less than four.

NOTE 2 When using Table D.1, take account of the scores for all items that apply. The scoring has been designed to allow for items which are not mutually exclusive. For example, a system with logic subsystem channels in separate racks is entitled to both the score for "Are the logic subsystem channels in separate cabinets?" and that for "Are the logic subsystem channels on separate printed-circuit boards?".

NOTE 3 If sensors or final elements are PE-based, they should be treated as part of the logic subsystem if they are enclosed within the same building (or vehicle) as the device that constitutes the major part of the logic subsystem, and as sensors or final elements if they are not so enclosed.

NOTE 4 For a non-zero value of *Z* to be used, it should be ensured that the equipment under control is put into a safe state before a non-simultaneous common cause failure can affect all the channels. The time taken to assure this safe state should be less than the claimed diagnostic test interval. A non-zero value for *Z* can be used only if:

– the system initiates an automatic shut-down on detection of a fault; or

– a safe shut-down is not initiated after a first fault [9], but the diagnostic tests:

  – determine the locality of the fault and are capable of localizing the fault; and

  – continue to be capable of placing the EUC in a safe state after the detection of any subsequent faults; or

– a formal system of work is in place to ensure that the cause of any revealed fault is fully investigated within the claimed diagnostic test interval; and

  – if the fault has the potential for leading to a common cause failure, the plant is immediately shut-down; or

  – the faulty channel is repaired within the claimed diagnostic test interval.

NOTE 5 In the process industries, it is unlikely to be feasible to shut down the EUC when a fault is detected within the diagnostic test interval as described in Table D.2. This methodology should not be interpreted as a requirement for process plants to be shut down when such faults are detected. However, if a shut-down is not implemented, no reduction in the β-factor can be gained by the use of diagnostic tests for the programmable electronics. In some industries, a shut-down may be feasible within the described time. In these cases, a non-zero value of *Z* may be used.

NOTE 6 Where diagnostic tests are carried out in a modular way, the repetition time used in Tables D.2 or D.3 is the time between the successive completions of the full set of diagnostic testing modules. The diagnostic coverage is the total coverage provided by all of the modules.

Table D.4 – Calculation of β_{int} or $\beta_{D\ int}$

Score (*S* or S_D)	Corresponding value of β_{int} or $\beta_{D\ int}$ for the:	
	Logic subsystem	**Sensors or final elements**
120 or above	0,5 %	1 %
70 to 120	1 %	2 %
45 to 70	2 %	5 %
Less than 45	5 %	10 %
NOTE 1 The maximum levels of $\beta_{D\ int}$ shown in this table are lower than would normally be used, reflecting the use of the techniques specified elsewhere in this standard for the reduction in the probability of systematic failures as a whole, and of common cause failures as a result of this.		
NOTE 2 Values of $\beta_{D\ int}$ lower than 0,5 % for the logic subsystem and 1 % for the sensors would be difficult to justify.		

The β_{int} derived from Table D.4 is the common cause failure associated with a 1oo2 system. For other levels of redundancy (MooN) this β_{int} value will change as given in Table D.5 to yield the final value of β.

Table D.5 can also be used to determine the final value of β_D but where there is β_{int} this can be substituted for $\beta_{D\ int}$.

NOTE 7 For related relevant information (on PDS method) see [25] in Bibliography

9) The operation of the system on the identification of a fault should be taken into account. For example, a simple 2oo3 system should be shut down (or repaired) within the times quoted in Tables D.2 or D.3, following the identification of a single failure. If this is not done, a failure of a second channel could result in the two failed channels outvoting the remaining (good) channel. A system which automatically reconfigures itself to 1oo2 voting when one channel fails, and which automatically shuts down on the occurrence of a second failure, has an increased probability of revealing the fault in the second channel and so a non-zero value for *Z* may be claimed.

Table D.5 – Calculation of β for systems with levels of redundancy greater than 1oo2

MooN		N			
		2	3	4	5
M	1	β_{int}	0,5 β_{int}	0,3 β_{int}	0,2 β_{int}
	2	-	1,5 β_{int}	0,6 β_{int}	0,4 β_{int}
	3	-	-	1,75 β_{int}	0,8 β_{int}
	4	-	-	-	2 β_{int}

D.6 Examples of the use of the β-factor methodology

In order to demonstrate the effect of using the β-factor methodology, some simple examples have been worked through in Table D.6 for the programmable electronics.

For categories not relating to diversity nor redundancy, typical values for X and Y were used. These were obtained by halving the maximum score for the category.

In the diverse system examples, the values for the diversity/redundancy category are derived from the following properties considered in Table D.1:

– one system is electronic, the other uses relay technology;
– the hardware diagnostic tests use different technologies;
– the different designers did not communicate during the design process;
– different test methods and test personnel were used to commission the systems; and
– maintenance is carried out by different people at different times.

In the redundancy system examples, the values for the diversity/redundancy category are derived from the property that the hardware diagnostics are carried out by an independent system, which uses the same technology as the redundancy systems.

For both the diverse and redundancy systems, a maximum and minimum value was used for *Z*, leading to four example systems in total.

Table D.6 – Example values for programmable electronics

Category		Diverse system with good diagnostic testing	Diverse system with poor diagnostic testing	Non Diverse system with good diagnostic testing	Non Diverse system with poor diagnostic testing
Separation/segregation	*X*	3,50	3,50	3,50	3,50
	Y	1,50	1,50	1,50	1,50
Diversity/redundancy	*X*	14,50	14,50	2,00	2,00
	Y	3,00	3,00	1,00	1,00
Complexity/design/.....	*X*	2,75	2,75	2,75	2,75
	Y	2,25	2,25	2,25	2,25
Assessment/analysis/....	*X*	0,25	0,25	0,25	0,25
	Y	4,75	4,75	4,75	4,75
Procedures/human interface	*X*	3,50	3,50	3,50	3,50
	Y	3,00	3,00	3,00	3,00
Competence/training/...	*X*	1,25	1,25	1,25	1,25
	Y	3,75	3,75	3,75	3,75
Environmental control	*X*	2,75	2,75	2,75	2,75
	Y	2,25	2,25	2,25	2,25
Environmental test	*X*	5,00	5,00	5,00	5,00
	Y	5,00	5,00	5,00	5,00
Diagnostic coverage	*Z*	2,00	0,00	2,00	0,00
Total *X*		33,5	33,5	21	21
Total *Y*		25,5	25,5	23,5	23,5
Score *S*		59	59	44,5	44,5
β		2 %	2 %	5 %	5 %
Score S_D		126	59	86,5	44,5
β_D		0,5 %	2 %	1 %	5 %
Diverse system 1002 (Table D.5) Non Diverse system is triplex with 2003 voting (Table D.5)		0,5 %	2 %	1,5 %	7,5 %

D.7 Binomial failure rate (Shock model) – CCF approach

The common cause failure (CCF) field feedback shows that if numerous double failures, a few triple failures and perhaps one quadruple failure have been observed, no multiple failures beyond order four have ever been observed from an explicit single cause which could not have been identified during the safety analysis. Consequently, the probability of multiple dependent failures decreases when the order of the CCF increases. Therefore, if the β-factor model is realistic for double failures and slightly pessimistic for triple failures it becomes much too conservative for quadruple failures and beyond. Let us consider the typical example of a safety instrumented system which closes the n production wells (e.g. n=150) of an oil field when a blocked outlet is occurring. Of course 2, 3 or even 4 wells may fail to close due to a non explicit CCF but not the n wells as modelled by the β-factor (otherwise the CCF would be explicit and should be analysed as an individual failure). Another typical example occurs when dealing with several safety layers at the same time. Considering, for example, the potential CCF between sensors of two protection layers may imply to consider the CCFs between six sensors (i.e. 3 sensors for each layer).

Several models have been proposed [18] to deal with this difficulty but most of them require so many reliability parameters (e.g. multiple Greek letters or α-models) that they become unrealistic. Among them, the binomial failure rate (Shock model) introduced by Vesely in 1977 and improved by Atwood in 1986 provides a pragmatic solution [18, 19]. The principle is that when a CCF occurs, it is similar to a shock on the related components. This shock may be lethal (i.e. same impact as in the β-factor model) or non-lethal and in this case there is only a certain probability that a given component fails due to the shock. Then the probability of having *k* failures due to the non-lethal shock is binomially distributed.

This model needs only 3 parameters to be implemented:

– ω lethal shock rate;
– ρ non-lethal shock rate;
– γ conditional probability of failure of a component given a non-lethal shock.

Figure D.2 gives an example on implementing this method when using a fault tree.

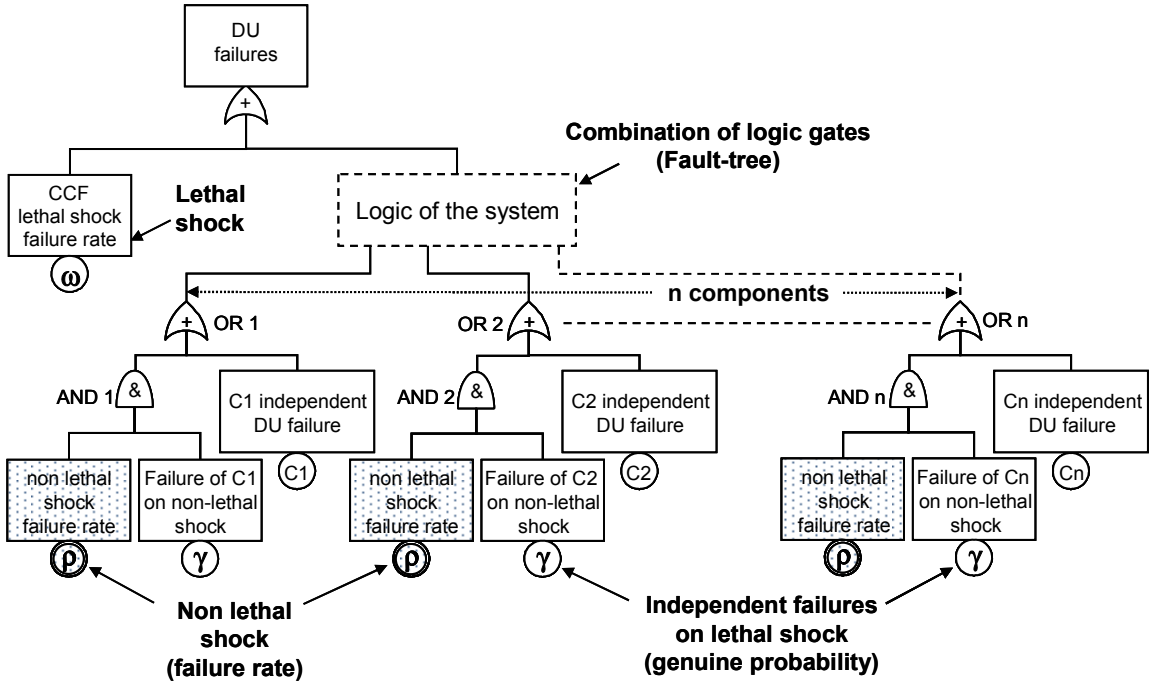


Figure D.2 – Implementing shock model with fault trees

Identical components can be linked to the *β*-factor model by splitting β in two parts β_L and β_{NL}:

– $\beta = \beta_L + \beta_{NL}$
– failure rate due to lethal shock: $\lambda_{DU} \times \beta_L$
– failure rate due to non lethal shock: $\lambda_{DU} \times \beta_{NL}$
– independent failure rate: $\lambda_{DU}[1 - (\beta_L + \beta_{NL})]$

In the fault tree represented in Figure D.2, this becomes:

– lethal shock rate: $\omega = \lambda_{DU} \times \beta_L$
– non-lethal shock rate : $\rho = \lambda_{DU} \times \beta_{NL} / \gamma$

As usual, the main problem is to evaluate the values of the three parameters (ω, ρ, γ) or (β_L, β_{NL}, $\tilde{\gamma}$) Reference [19] gives some indications and provides other references about the statistical treatments allowing to evaluate(ω, ρ, γ) from field feedback.

If no data are available the engineering judgement can be used through pragmatic approaches. For example the following procedure may be used with fault tree modelling when there are more than 3 similar items:

1) estimate β as in the β-factor method;
2) consider that β_L is negligible ($\beta_{NL} = \beta$);

3) estimate γ in order to be sure to obtain conservative results. Considered that double failures have at least an impact 10 times higher than the quadruple failure (hypothesis which is certainly conservative), the following formula may be used:

$$\gamma = \sqrt{\frac{C_N^2}{10.C_N^4}}$$

where

N is the number of similar items;

C_N^2 is the number of potential double failures; and

C_N^4 is the number of potential quadruple failures

4) calculate ρ in function of the number N of similar items:

$$\rho = \frac{\beta\lambda_{DU}}{C_N^2\gamma^2 + C_N^3\gamma^3}$$

In this method of working, the top contributors are double and triple failures and the results are conservative compared to the results obtained with the β factor method with only 3 components. The CCF double and triple failures are taken under consideration properly and unrealistic multiple failures are not completely neglected.

This model is very easily implemented into fault tree calculation models like those presented in Annex B, e.g. fault trees in B.4.3. This allows handling safety systems comprising a lot of similar components in a very simple and easy way.

D.8 References

References [13] to [15] and [20] and [21] in the Bibliography provide useful information relating to common cause failures.

Annex E
(informative)

Example applications of software safety integrity tables of IEC 61508-3

E.1 General

This annex gives two worked examples in the application of the software safety integrity tables specified in Annex A of IEC 61508-3:

a) safety integrity level 2: a programmable electronic safety-related system required for a process within a chemical plant;

b) safety integrity level 3: a shut-down application based on a high-level language.

These examples illustrate how software development techniques might be selected in particular circumstances from the tables of Annexes A and B of IEC 61508-3.

It should be emphasized that these illustrations are not definitive applications of the standard to these examples. IEC 61508-3 states clearly at several points that given the large number of factors that affect software systematic capability, it is not possible to give an algorithm for combining the techniques and measures that will be correct for any given application.

For a real system, all the entries in the tables should be supported by documented justification that the comments made are correct and that they represent an appropriate response for the particular system and application. This justification is likely to be assisted by referring to the guidance of IEC 61508-3 Annex C which discusses the desirable properties which, if achieved in the appropriate lifecycle phase, may convincingly justify confidence that the eventual software has sufficient systematic safety integrity.

E.2 Example for safety integrity level 2

This example is a safety integrity level 2 programmable electronic safety-related system required for a process within a chemical plant. The programmable electronic safety-related system utilizes ladder logic for the application program, and is an illustration of limited variability language application programming.

The application consists of several reactor vessels linked by intermediate storage vessels which are filled with inert gas at certain points in the reaction cycle to suppress ignition and explosions. The programmable electronic safety-related system functions include: receiving inputs from the sensors; energizing and interlocking the valves, pumps and actuators; detecting dangerous situations and activating the alarm; interfacing to a distributed control system, as required by the safety requirements specification.

Assumptions:

– the programmable electronic safety-related system controller is a PLC;

– the hazard and risk analysis has established that a programmable electronic safety-related system is required, and that safety integrity level 2 is required in this application (by the application of IEC 61508-1 and IEC 61508-2);

– although the controller operates in real time, only a relatively slow response is needed;

– there are interfaces to a human operator and to a distributed control system;

– the source code of the system software and the design of the programmable electronics of the PLC is not available for examination, but has been qualified against IEC 61508-3 to safety integrity level 2;

– the language used for application programming is ladder logic, produced using the PLC supplier's development system;

– the application code is required to run on only a single type of PLC;

– the whole of the software development was reviewed by a person independent of the software team;

– a person independent of the software team witnessed and approved the validation testing;

– modifications (if needed) require authorization by a person independent of the software team.

NOTE 1 For the definition of an independent person, see IEC 61508-4.

NOTE 2 See the notes to 7.4.2, 7.4.3, 7.4.4 and 7.4.5 of IEC 61508-3 for information on the division of responsibility between the PLC supplier and user when limited variability programming is used.

The following tables show how Annex A of IEC 61508-3 may be interpreted for this application.

Table E.1 – Software safety requirements specification

(See 7.2 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1a	Semi-formal methods	Table B.7	R	Cause-effect diagrams, sequence diagrams, function blocks. Typically used for PLC application software requirements specification
1b	Formal methods	B.2.2, C.2.4	R	Not used for limited variability programming
2	Forward traceability between the system safety requirements and the software safety requirements	C.2.11	R	Check completeness: review to ensure that all system safety requirements are addressed by software safety requirements
3	Backward traceability between the safety requirements and the perceived safety needs	C.2.11	R	Minimise complexity and functionality: review to ensure that all software safety requirements are actually needed to address system safety requirements
4	Computer-aided specification tools to support appropriate techniques/measures above	B.2.4	R	Development tools supplied by the PLC manufacturer

NOTE 1 In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.

NOTE 2 The software safety requirements were specified in natural language.

Table E.2 – Software design and development – software architecture design

(see 7.4.3 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Fault detection	C.3.1	R	Checking of data range, watch-dog timer, I/O, communication. Raise an alarm if errors (see 3a)
2	Error detecting codes	C.3.2	R	Embedded with user options - careful selection required
3a	Failure assertion programming	C.3.3	R	Dedicate some PLC program ladder logic to test certain essential safety conditions (see 1)
3b	Diverse monitor techniques (with independence between the monitor and the monitored function in the same computer)	C.3.4	R	Not preferred: increased software complexity to guarantee independence.
3c	Diverse monitor techniques (with separation between the monitor computer and the monitored computer)	C.3.4	R	Check legal I/O combinations in an independent hardware safety monitor
3d	Diverse redundancy, implementing the same software safety requirements specification	C.3.5	---	Not preferred: insufficient increased safety benefit over 3c.
3e	Functionally diverse redundancy, implementing different software safety requirements specification	C.3.5	---	Not preferred: substantially achieved by 3c.
3f	Backward recovery	C.3.6	R	Embedded with user options – careful selection required
3g	Stateless software design (or limited state design)	C.2.12	---	Not used. Process control needs states to memorise plant condition.
4a	Re-try fault recovery mechanisms	C.3.7	R	Used as required by the application (see 2 and 3c)
4b	Graceful degradation	C.3.8	R	Not used for limited variability programming
5	Artificial intelligence - fault correction	C.3.9	NR	Not used for limited variability programming
6	Dynamic reconfiguration	C.3.10	NR	Not used for limited variability programming
7	Modular approach	Table B.9	HR	
8	Use of trusted/verified software elements (if available)	C.2.10	HR	Pre-existing code from earlier projects
9	Forward traceability between the software safety requirements specification and software architecture	C.2.11	R	Check completeness: review to ensure that all software safety requirements are addressed by the software architecture
10	Backward traceability between the software safety requirements specification and software architecture	C.2.11	R	Minimise complexity and functionality: review to ensure that all architecture safety requirements are actually needed to address software safety requirements
11a	Structured diagrammatic methods	C.2.1	HR	Data flow methods and data logic tables may be used for representing at least the design architecture
11b	Semi-formal methods	Table B.7	R	May be used for DCS interface
11c	Formal design and refinement methods	B.2.2, C.2.4	R	Rarely used for limited variability programming
11d	Automatic software generation	C.4.6	R	Not used for limited variability programming
12	Computer-aided specification and design tools	B.2.4	R	Development tools supplied by the PLC manufacturer

Technique/Measure		Ref.	SIL 2	Interpretation in this application
13a	Cyclic behaviour, with guaranteed maximum cycle time	C.3.11	HR	Not used. Hardware monitoring of PLC cycle time.
13b	Time-triggered architecture	C.3.11	HR	Not used. Hardware monitoring of PLC cycle time.
13c	Event-driven, with guaranteed maximum response time	C.3.11	HR	Not used. Hardware monitoring of PLC cycle time.
14	Static resource allocation	C.2.6.3	R	Not used. Dynamic resources issues do not arise in limited variability programming
15	Static synchronisation of access to shared resources	C.2.6.3	---	Not used. Dynamic resources issues do not arise in limited variability programming
NOTE 1 In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B. NOTE 2 It is impractical to implement some of the above techniques in limited variability programming.				

Table E.3 – Software design and development – support tools and programming language

(See 7.4.4 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Suitable programming language	C.4.5	HR	Usually ladder, and often the proprietary variety of the PLC supplier
2	Strongly typed programming language	C.4.1	HR	Not used. Use PLC-oriented structured text (see [16] in the Bibliography)
3	Language subset	C.4.2	---	Beware of complex "macro" instructions, interrupts which alter PLC scan cycle, etc.
4a	Certified tools and certified translators	C.4.3	HR	Available from some PLC manufacturers
4b	Tools and translators: increased confidence from use	C.4.4	HR	PLC supplier's development kit; in-house tools developed over several projects
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.4 – Software design and development – detailed design

(See 7.4.5 and 7.4.6 of IEC 61508-3)
(Includes software system design, software module design and coding)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1a	Structured methods	C.2.1	HR	Not used for limited variability programming
1b	Semi-formal methods	Table B.7	HR	Cause-effect diagrams, sequence diagrams, function blocks. Typical for limited variability programming
1c	Formal design and refinement methods	B.2.2, C.2.4	R	Not used for limited variability programming
2	Computer-aided design tools	B.3.5	R	Development tools supplied by the PLC manufacturer
3	Defensive programming	C.2.5	R	Included in the system software
4	Modular approach	Table B.9	HR	Order and group the PLC program ladder logic to maximize its modularity with respect to the functions required
5	Design and coding standards	C.2.6 Table B.1	HR	In-house conventions for documentation and maintainability
6	Structured programming	C.2.7	HR	Similar to modularity in this context
7	Use of trusted/verified software elements (if available)	C.2.10	HR	Function blocks, part programs
8	Forward traceability between the software safety requirements specification and software design	C.2.11	R	Check completeness: review to ensure that all software safety requirements are addressed by the software design

NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.

Table E.5 – Software design and development – software module testing and integration

(See 7.4.7 and 7.4.8 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Probabilistic testing	C.5.1	R	Not used for limited variability programming
2	Dynamic analysis and testing	B.6.5 Table B.2	HR	Used
3	Data recording and analysis	C.5.2	HR	Records of test cases and results
4	Functional and black box testing	B.5.1 B.5.2 Table B.3	HR	Input data is selected to exercise all specified functional cases, including error handling. Test cases from cause consequence diagrams, boundary value analysis, and input partitioning
5	Performance testing	Table B.6	R	Not used for limited variability programming
6	Model based testing	C.5.27	R	Not used for limited variability programming
7	Interface testing	C.5.3	R	Included in functional and black-box testing
8	Test management and automation tools	C.4.7	HR	Development tools supplied by the PLC manufacturer
9	Forward traceability between the software design specification and the module and integration test specifications	C.2.11	R	Check completeness: review to ensure that an adequate test is planned to examine the functionality of all modules and their integration with appropriately related modules.
10	Formal verification	C.5.12	---	Not used for limited variability programming
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.6 – Programmable electronics integration (hardware and software)

(See 7.5 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Functional and black box testing	B.5.1 B.5.2 Table B.3	HR	Input data is selected to exercise all specified functional cases, including error handling. Test cases from cause consequence diagrams, boundary value analysis, and input partitioning
2	Performance testing	Table B.6	R	When the PLC system is assembled for factory acceptance test
3	Forward traceability between the system and software design requirements for hardware/software integration and the hardware/software integration test specifications	C.2.11	R	Review to ensure that the hardware/software integration tests are adequate
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.7 – Software aspects of system safety validation

(See 7.7 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Probabilistic testing	C.5.1	R	Not used for limited variability programming
2	Process simulation	C.5.18	R	Not used for limited variability programming, but becoming more commonly used in PLC systems development
3	Modelling	Table B.5	R	Not used for limited variability programming, but becoming more commonly used in PLC systems development
4	Functional and black-box testing	B.5.1 B.5.2 Table B.3	HR	Input data is selected to exercise all specified functional cases, including error handling. Test cases from cause consequence diagrams, boundary value analysis, and input partitioning
5	Forward traceability between the software safety requirements specification and the software safety validation plan	C.2.11	R	Check completeness: review to ensure that adequate software validation tests are planned to address the software safety requirements
6	Backward traceability between the software safety validation plan and the software safety requirements specification	C.2.11	R	Minimise complexity: review to ensure that all validation tests are relevant.
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.8 – Software modification

(See 7.8 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Impact analysis	C.5.23	HR	An impact analysis is carried out to consider how the effect of the proposed changes is limited by the modularity of the overall system
2	Reverify changed software module	C.5.23	HR	Repeat earlier tests
3	Reverify affected software modules	C.5.23	HR	Repeat earlier tests
4a	Revalidate complete system	Table A.7	R	Impact analysis showed that the modification is necessary, so revalidation is done as required
4b	Regression validation	C.5.25	HR	
5	Software configuration management	C.5.24	HR	Baselines, records of changes, impact on other system requirements
6	Data recording and analysis	C.5.2	HR	Records of test cases and results
7	Forward traceability between the Software safety requirements specification and the software modification plan (including reverification and revalidation)	C.2.11	R	Adequate modification procedures to achieve the software safety requirements
8	Backward traceability between the software modification plan (including reverification and revalidation)and the Software safety requirements specification	C.2.11	R	Adequate modification procedures to achieve the software safety requirements
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.9 – Software verification

(See 7.9 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Formal proof	C.5.12	R	Not used for limited variability programming
2	Animation of specification and design	C.5.26	R	
3	Static analysis	B.6.4 Table B.8	HR	Clerical cross-referencing of usage of variables, conditions, etc.
4	Dynamic analysis and testing	B.6.5 Table B.2	HR	Automatic test harness to facilitate regression testing
5	Forward traceability between the software design specification and the software verification (including data verification) plan	C.2.11	R	Check completeness: review to ensure adequate test of functionality.
6	Backward traceability between the software verification (including data verification) plan and the software design specification	C.2.11	R	Minimise complexity: review to ensure that all verification tests are relevant.
7	Offline numerical analysis	C.2.13	R	Not used. The numerical stability of calculations is not a major concern here
Software module testing and integration		See Table E.5 of this standard		
Programmable electronics integration testing		See Table E.6 of this standard		
Software system testing (validation)		See Table E.7 of this standard		
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.10 – Functional safety assessment

(see Clause 8 of IEC 61508-3)

Technique/Measure		Ref.	SIL 2	Interpretation in this application
1	Checklists	B.2.5	R	Used
2	Decision/truth tables	C.6.1	R	Used to a limited degree
3	Failure analysis	Table B.4	R	Cause-consequence diagrams at system level, but otherwise, failure analysis is not used for limited variability programming
4	Common cause failure analysis of diverse software (if diverse software is actually used)	C.6.3	R	Not used for limited variability programming
5	Reliability block diagram	C.6.4	R	Not used for limited variability programming
6	Forward traceability between the requirements of Clause 8 and the plan for software functional safety assessment	C.2.11	R	Check completeness of coverage of the functional safety assessment
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

E.3 Example for safety integrity level 3

This second example is a shut-down application based on a high-level language, of safety integrity level 3.

The software system is relatively large in terms of safety systems; more than 30 000 lines of source code are developed specifically for the system. Also, the usual intrinsic functions are used – at least two diverse operating systems and pre-existing code from earlier projects (proven in use). In total, the system constitutes more than 100 000 lines of source code, if it were available as such.

The whole hardware (including sensors and actuators) is a dual-channel system with its outputs to the final elements connected as a logical AND.

Assumptions:

– although fast response is not required a maximum response time is guaranteed;
– there are interfaces to sensors, actuators and annunciators to human operators;
– the source code of the operating systems, graphic routines and commercial mathematical routines is not available;
– the system is very likely to be subject to later changes;
– the specifically developed software uses one of the common procedural languages;
– it is partially object oriented;
– all parts for which source code is not available are implemented diversely, with the software components being taken from different suppliers and their object code generated by diverse translators;
– the software runs on several commercially available processors that fulfil the requirements of IEC 61508-2;
– all requirements of IEC 61508-2 for control and avoidance of hardware faults are fulfilled by the embedded system; and
– the software development was assessed by an independent organization.

NOTE For the definition of an independent organization, see IEC 61508-4.

The following tables show how the annex tables of IEC 61508-3 may be interpreted for this application.

Table E.11 – Software safety requirements specification

(See 7.2 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1a	Semi-formal methods	Table B.7	HR	Block diagrams, sequence diagrams, state transition diagrams
1b	Formal methods	B.2.2, C.2.4	R	Only exceptionally
2	Forward traceability between the system safety requirements and the software safety requirements	C.2.11	HR	Check completeness: review to ensure that all system safety requirements are addressed by software safety requirements
3	Backward traceability between the safety requirements and the perceived safety needs	C.2.11	HR	Minimise complexity and functionality: review to ensure that all software safety requirements are actually needed to address system safety requirements
4	Computer-aided specification tools to support appropriate techniques/measures above	B.2.4	HR	Tools supporting the chosen methods
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.12 – Software design and development – software architecture design

(see 7.4.3 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Fault detection	C.3.1	HR	Used as far as dealing with sensor, actuator and data transmission failures and which are not covered by the measures within the embedded system according to the requirements of IEC 61508-2
2	Error detecting codes	C.3.2	R	Only for external data transmissions
3a	Failure assertion programming	C.3.3	R	Results of the application functions are checked for validity
3b	Diverse monitor techniques (with independence between the monitor and the monitored function in the same computer)	C.3.4	R	Not preferred: increased software complexity to guarantee independence.
3c	Diverse monitor techniques (with separation between the monitor computer and the monitored computer)	C.3.4	R	Used for some safety related functions where 3a is not used
3d	Diverse redundancy, implementing the same software safety requirements specification	C.3.5	---	Used for some functions where source code is not available
3e	Functionally diverse redundancy, implementing different software safety requirements specification	C.3.5	R	Not preferred: substantially achieved by 3c.
3f	Backward recovery	C.3.6	---	Not used
3g	Stateless software design (or limited state design)	C.2.12	R	Not used. A controlled shutdown needs states to memorise plant condition.
4a	Re-try fault recovery mechanisms	C.3.7	---	Not used
4b	Graceful degradation	C.3.8	HR	Yes, because of the nature of the technical process

Technique/Measure		Ref.	SIL 3	Interpretation in this application
5	Artificial intelligence - fault correction	C.3.9	NR	Not used
6	Dynamic reconfiguration	C.3.10	NR	Not used
7	Modular approach	Table B.9	HR	Needed because of the size of the system
8	Use of trusted/verified software elements (if available)	C.2.10	HR	pre-existing code from earlier projects
9	Forward traceability between the software safety requirements specification and software architecture	C.2.11	HR	Review to ensure that all software safety requirements are addressed by the software architecture
10	Backward traceability between the software safety requirements specification and software architecture	C.2.11	HR	Minimise complexity and functionality: review to ensure that all architecture safety requirements are actually needed to address software safety requirements
11a	Structured diagrammatic methods	C.2.1	HR	Needed because of the size of the system
11b	Semi-formal methods	Table B.7	HR	Block diagrams, sequence diagrams, state transition diagrams
11c	Formal design and refinement methods	B.2.2, C.2.4	R	Not used
11d	Automatic software generation	C.4.6	R	Not used. Avoid translator/generator uncertainty.
12	Computer-aided specification and design tools	B.2.4	HR	Tools supporting the chosen method
13a	Cyclic behaviour, with guaranteed maximum cycle time	C.3.11	HR	Not used
13b	Time-triggered architecture	C.3.11	HR	Not used
13c	Event-driven, with guaranteed maximum response time	C.3.11	HR	Not used
14	Static resource allocation	C.2.6.3	HR	Not used. Choose programming language to avoid dynamic resources issues
15	Static synchronisation of access to shared resources	C.2.6.3	R	Not used. Choose programming language to avoid dynamic resources issues
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.13 – Software design and development – support tools and programming language

(See 7.4.4 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Suitable programming language	C.4.5	HR	Full variability high-level language selected
2	Strongly typed programming language	C.4.1	HR	Used
3	Language subset	C.4.2	HR	Defined subset for the selected language
4a	Certified tools and certified translators	C.4.3	HR	Not available
4b	Tools and translators: increased confidence from use	C.4.4	HR	Available, and used
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.14 – Software design and development – detailed design

(See 7.4.5 and 7.4.6 of IEC 61508-3)
(Includes software system design, software module design and coding)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1a	Structured methods	C.2.1	HR	Widely used. In particular, SADT and JSD
1b	Semi-formal methods	Table B.7	HR	Finite state machines/state transition diagrams, block diagrams, sequence diagrams
1c	Formal design and refinement methods	B.2.2, C.2.4	R	Only exceptionally, for some very basic components only
2	Computer-aided design tools	B.3.5	HR	Used for the selected methods
3	Defensive programming	C.2.5	HR	All measures except those which are automatically inserted by the compiler are explicitly used in application software where they are effective
4	Modular approach	Table B.9	HR	Software module size limit, information hiding/encapsulation, one entry/one exit point in subroutines and functions, fully defined interface, …
5	Design and coding standards	C.2.6 Table B.1	HR	Use of coding standard, no dynamic objects, no dynamic variables, limited use of interrupts, limited use of pointers, limited use of recursion, no unconditional jumps, …
6	Structured programming	C.2.7	HR	Used
7	Use of trusted/verified software elements (if available)	C.2.10	HR	Available, and used
8	Forward traceability between the software safety requirements specification and software design	C.2.11	HR	Review to ensure that all software safety requirements are addressed by the software design

NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.

Table E.15 – Software design and development – software module testing and integration

(See 7.4.7 and 7.4.8 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Probabilistic testing	C.5.1	R	Used for software modules where no source code available and the definition of boundary values and equivalence classes for test data is difficult
2	Dynamic analysis and testing	B.6.5 Table B.2	HR	Used for software modules where source code is available. Test cases from boundary value analysis, performance modelling, equivalence classes and input partitioning, structure-based testing
3	Data recording and analysis	C.5.2	HR	Records of test cases and results
4	Functional and black box testing	B.5.1 B.5.2 Table B.3	HR	Used for software module testing where no source code is available and for integration testing. Input data is selected to exercise all specified functional cases, including error handling. Test cases from cause consequence diagrams, prototyping, boundary value analysis, equivalence classes and input partitioning

Technique/Measure		Ref.	SIL 3	Interpretation in this application
5	Performance testing	Table B.6	HR	Used during integration testing on the target hardware
6	Model based testing	C.5.27	HR	Not used
7	Interface testing	C.5.3	HR	Included in functional and black-box testing
8	Test management and automation tools	C.4.7	HR	Used where available
9	Forward traceability between the software design specification and the module and integration test specifications	C.2.11	HR	Review to ensure that the integration tests are sufficient
10	Formal verification	C.5.12	R	Not used
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.16 – Programmable electronics integration (hardware and software)

(See 7.5 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Functional and black box testing	B.5.1 B.5.2 Table B.3	HR	Used as additional tests to software integration testing (see Table E.15 above) Input data is selected to exercise all specified functional cases, including error handling. Test cases from cause consequence diagrams, prototyping, boundary value analysis, equivalence classes and input partitioning
2	Performance testing	Table B.6	HR	Extensively used
3	Forward traceability between the system and software design requirements for hardware/software integration and the hardware/software integration test specifications	C.2.11	HR	Review to ensure that the integration tests are sufficient
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.17 – Software aspects of system safety validation

(See 7.7 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Probabilistic testing	C.5.1	R	Not used for validation
2	Process simulation	C.5.18	HR	Finite state machines, performance modelling, prototyping and animation
3	Modelling	Table B.5	HR	Not used for validation
4	Functional and black-box testing	B.5.1 B.5.2 Table B.3	HR	Input data is selected to exercise all specified functional cases, including error handling. Test cases from cause consequence diagrams, boundary value analysis, and input partitioning
5	Forward traceability between the software safety requirements specification and the software safety validation plan	C.2.11	HR	Check completeness: review to ensure that all software safety requirements are addressed by the validation plan
6	Backward traceability between the software safety validation plan and the software safety requirements specification	C.2.11	HR	Minimise complexity: review to ensure that all validation tests are relevant
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.18 – Modification

(See 7.8 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Impact analysis	C.5.23	HR	Used
2	Reverify changed software module	C.5.23	HR	Used
3	Reverify affected software modules	C.5.23	HR	Used
4a	Revalidate complete system	Table A.7	HR	Depends on the result of the impact analysis
4b	Regression validation	C.5.25	HR	Used
5	Software configuration management	C.5.24	HR	Used
6	Data recording and analysis	C.5.2	HR	Used
7	Forward traceability between the Software safety requirements specification and the software modification plan (including reverification and revalidation)	C.2.11	HR	Check completeness: review to ensure that the modification procedures are adequate to achieve the software safety requirements
8	Backward traceability between the software modification plan (including reverification and revalidation)and the software safety requirements specification	C.2.11	HR	Minimise complexity: review to ensure that all modification procedures are necessary
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.19 – Software verification

(See 7.9 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Formal proof	C.5.12	R	Only exceptionally, for some very basic classes only
2	Animation of specification and design	C.5.26	R	Not used
3	Static analysis	B.6.4 Table B.8 C.5.14	HR	For all newly developed code. Boundary value analysis, checklists, control flow analysis, data flow analysis, Fagan inspections, design reviews
4	Dynamic analysis and testing	B.6.5 Table B.2	HR	For all newly developed code
5	Forward traceability between the software design specification and the software verification (including data verification) plan	C.2.11	HR	Check completeness: review to ensure that the modification procedures are adequate for the software safety requirements
6	Backward traceability between the software verification (including data verification) plan and the software design specification	C.2.11	HR	Minimise complexity: review to ensure that all modification procedures are necessary
7	Offline numerical analysis	C.2.13	HR	Not used. The numerical stability of calculations is not a major concern here
Software module testing and integration		See Table E.15 of this standard		
Programmable electronics integration testing		See Table E.16 of this standard		
Software system testing (validation)		See Table E.17 of this standard		
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Table E.20 – Functional safety assessment

(see Clause 8 of IEC 61508-3)

Technique/Measure		Ref.	SIL 3	Interpretation in this application
1	Checklists	B.2.5	R	Used
2	Decision/truth tables	C.6.1	R	Used, to a limited degree
3	Failure analysis	Table B.4	HR	Fault-tree analysis is extensively used, and cause consequence diagrams are used to a limited degree
4	Common cause failure analysis of diverse software (if diverse software is actually used)	C.6.3	HR	Used
5	Reliability block diagram	C.6.4	R	Used
6	Forward traceability between the requirements of Clause 8 and the plan for software functional safety assessment	C.2.11	HR	Check completeness of coverage of the functional safety assessment
NOTE In the reference columns (entitled Ref), the informative references “B.x.x.x”, “C.x.x.x” refer to descriptions of techniques in IEC 61508-7 Annexes B and C, while “Table A.x”, “Table B.x” refer to tables of techniques in IEC 61508-3 Annexes A and B.				

Bibliography

[1] IEC 61511 (all parts), *Functional safety – Safety instrumented systems for the process industry sector*

[2] IEC 62061, *Safety of machinery – Functional safety of safety-related electrical, electronic and programmable electronic control systems*

[3] IEC 61800-5-2, *Adjustable speed electrical power drive systems – Part 5-2: Safety requirements – Functional*

The following references give further details on evaluating probabilities of failure (see Annex B).

[4] IEC 61078:2006, *Analysis techniques for dependability – Reliability block diagram and boolean methods*

[5] IEC 61165:2006, *Application of Markov techniques*

[6] BS 5760, *Reliability of system equipment and components – Part 2: Guide to assessment of reliability*

[7] D. J. SMITH, *Reliability, maintainability and risk – Practical methods for engineers*, Butterworth-Heinemann, 5th edition, 1997, ISBN 0-7506-3752-8

[8] R. BILLINGTON and R. N. ALLAN, *Reliability evaluation of engineering systems*, Plenum, 1992, ISBN 0-306-44063-6

[9] W. M. GOBLE, *Evaluating control system reliability – Techniques and applications*, Instrument Society of America, 1992, ISBN 1-55617-128-5

Useful references for the calculation of diagnostic coverage (see Annex C) include the following.

[10] Reliability Analysis Center (RAC), *Failure Mode/Mechanism Distributions*, 1991, Department of Defense, United States of America, PO Box 4700, 201 Mill Street, Rome, NY 13440-8200, Organization report number: FMD-91, NSN 7540-01-280-5500

[11] ALLESSANDRO BIROLINI*, Qualität und Zuverlassigkeit technischer Systeme, Theorie, Praxis, Management*, Dritte Auflage,1991, Springer-Verlag, Berlin Heidelberg New York, ISBN 3-540-54067-9, 3 Aufl., ISBN 0-387-54067-9 3 ed. (available in German only)

[12] MIL-HDBK-217F, *Military Handbook Reliability prediction of electronic equipment*, 2 December 1991, Department of Defense, United States of America

The following references provide useful information relating to common cause failures (see Annex D).

[13] *Health and Safety Executive Books*, email hsebooks@prolog.uk.com

[14] R. HUMPHREYS, A., PROC., *Assigning a numerical value to the beta factor common-cause evaluation,* Reliability 1987

[15] UPM3.1, *A pragmatic approach to dependent failures assessment for standard systems*, AEA Technology, Report SRDA-R-13, ISBN 085 356 4337, 1996

The following standard is referred to in Table E.3.

[16] IEC 61131-3:2003, *Programmable controllers – Part 3: Programming languages*

[17] ISA-TR84.00.02-2002 – Parts 1-5, Safety Instrumented Functions (SIF) Safety Integrity Level (SIL) Evaluation Techniques Package.

[18] IEC 61025:2006, *Fault tree analysis (FTA)*

[19] IEC 62551, *Analysis techniques for dependability – Petri Net technique*[10]

[20] ANIELLO AMENDOLA, kluwer academic publisher, ISPRA 16-19 November 1987, *Advanced seminar on Common Cause Failure Analysis in Probabilistic Safety Assessment,* ISBN 0-7923-0268-0

[21] CORWIN L. ATWOOD, *The Binomial Failure Rate Common Cause Model,* Technometrics May 1986 Vol 28 n°2

[22] A. ARNOLD, A. GRIFFAULT, G. POINT, AND A. RAUZY. *The altarica language and its semantics*. *Fundamenta Informaticae*, 34,pp.109–124, 2000

[23] M. BOITEAU, Y. DUTUIT, A. RAUZY AND J.-P. SIGNORET, The AltaRica *Data-Flow Language in Use: Assessment of Production Availability of a MultiStates System*, *Reliability Engineering and System Safety*, Elsevier, Vol. 91, pp 747-755

[24] A. RAUZY. *Mode automata and their compilation into fault trees. Reliability Engineering and System Safety*, Elsevier 2002, Volume 78, Issue 1, pp 1-12

[25] For PDS method; see <www.sintef.no/pds>; and further background material in:Hokstad, Per; Corneliussen, Kjell Source: Reliability Engineering and System Safety, v 83, n 1, p 111-120, January 2004

[26] IEC 60601 (all parts), *Medical electrical equipment*

[27] IEC 61508-1:2010 *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 1: General requirements*

[28] IEC 61508-5:2010 *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 5: Examples of methods for the determination of safety integrity levels*

[29] IEC 61508-7:2010 *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 7: Overview of techniques and measures*

———————

10 Under consideration.

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch