IEEE Standard for Verification of Vital Functions in Processor-Based Systems Used in Rail Transit Control

Rail Transit Vehicle Interface Committee
of the
IEEE Vehicular Technology Society

Approved 30 March 2000

IEEE-SA Standards Board

Abstract: A set of standard verification tasks for processor-based equipment used in safety-critical applications on rail and transit systems is covered. This standard also covers processes that verify the level of safety achieved in the implementation of safety-critical functions that are required to be fail-safe. Quality assurance or validation processes that affect the overall level of system safety are not covered.
Keywords: rail, safety, safety critical, software, transit, verification, vital

---

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2000 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 5 April 2000. Printed in the United States of America.

Print: ISBN 0-7381-1966-0 SH94825
PDF: ISBN 0-7381-1967-9 SS94825

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

--`,,`-`-`,,`,,`,`,,`---

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS

Not for Resale

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; (978) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Introduction

(This introduction is not part of IEEE Std 1483-2000, IEEE Standard for Verification of Vital Functions in Processor-Based Systems Used in Rail Transit Control.)

The achievement of safety in railroad and transit systems is dependent upon many factors and activities. Among the required activities (for instance, those listed in MIL-STD-882C:1996) is the identification of system hazards, hazard severity, and hazard probabilities. As a result of these efforts, a portion of the system-functional requirements are identified as being critical to system safety and, as such, must be implemented fail-safely (vital functions). These vital functions must be performed as specified, and must be performed in a manner that does not increase risk in the presence of component failures (i.e., performed fail-safely). Traditional rail transit safety systems designed using only relays and/or discrete electrical or mechanical components had the performance of vital functions (with and without component failures) verified by relatively tedious, but easily understood, means. With the introduction of integrated circuits, microprocessors, and software, the complexity of safety-critical systems has grown beyond the ability of traditional analyses as the sole means of verifying their safe operation in the event of hardware component failures and/or software errors. It is the intent of this standard to provide agreed-upon safety verification procedures and analyses that, if implemented comprehensively, will verify that identified vital functions have been implemented fail-safely with a level of safety assurance adequate to meet the stated safety goals. In the context of this standard, the safety goals are defined as the quantitative limit of the probability and/or frequency that any vital function fails to be performed safely.

The scope of this standard is limited to the identification of those functions that will be performed fail-safely (i.e., vital functions) by the system, subsystem, or device under consideration, and the verification that the identified functions have been implemented fail-safely.

Identification of the set of vital functions to be verified begins with the identification of all hazards associated with the functional and operational requirements of the system [traditionally identified by a preliminary hazard analysis (PHA) within the overall system safety program], and proceeds using analytical means to identify all vital functions at the system-functional level. Vital functions to be verified are also identified through analysis of the safety-assurance concepts employed in the system design, which impose additional requirements for fail-safe implementation. The methods used to verify such functions are safety assurance concept dependent.

Verification of system safety requires the establishment of system safety goals and proceeds using analytical means to verify that vital functions have been implemented consistent with those goals.

This standard provides a structured process and defines a set of documents that comprehensively identify the set of system functions that must be implemented fail-safely (i.e., those derived from potential hazards associated with the specific functional and operational requirements), as well as those imposed by the design concepts selected. The vital functions are then associated with hardware and software components of the system implementation. Analyses and other verification methods are used to confirm that each of the identified functions are implemented fail-safely, and that the established safety goals are met.

To apply this standard, it is necessary to have established

- A clear definition of the scope of the system and its external interfaces
- System safety goals
- System functional requirements
- System hazards derived from the overall system PHA

The application of this standard

- Determines the set of functions the system must perform fail-safely (the vital functions).
- Requires verification methods appropriate to the safety assurance concept employed.
- Verifies that the design and its implementation performs vital functions fail-safely and meets the system safety goals.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Participants

At the time this standard was completed, Working Group 4 of the IEEE Vehicular Technology/Rail Transit Vehicle Interface Standards Committee had the following membership:

James Hoelscher, *Chair*
David Rutherford, *Vice Chair*

George Ackaji
George Anagnostopoulos
Richard Bamfield
Jerome Beshoner
Eric Cutright
Miodrag Denic
Vinnie Ferrando
Manuel Galdo
Bill Gallagher
Sudhir Kumar Gupta
Olivier Hocheid
Ken Jackson
Mark Jefferson
Kenneth Karg
John LaForce
Martin Lukes
Linda Martinez
Richard J. Mazur
Tom McGean
Howard Moody
Lang Nguyen
Robert D. Pascoe
William Petit
Venkat Rao Pindiprolu
Wayne Quon
Omar Rezzoug
Alan F. Rumsey
Louis Sanders
Wade Scott
Rick Shults
James Stitt
Adam Szymkowiak

The following members of the balloting committee voted on this standard:

George Anagnostopoulos
Jerome Beshoner
Linda Sue Boehmer
Eric Cutright
Vinnie Ferrando
Manuel Galdo
Bill Gallagher
James R. Hoelscher
Ronald Kangas
Abe Kanner
Kenneth Karg
James Kyle
John LaForce
John C. Lewis
Richard J. Mazur
Joe McDermott
Bob Miller
Howard Moody
Edwin A. Mortlock
Lang Nguyen
Robert D. Pascoe
William Petit
Venkat Rao Pindiprolu
Omar Rezzoug
Alan F. Rumsey
David B. Rutherford
Louis Sanders
Wade Scott
James Stitt
Linda Tajerian-Martinez
Errol F. Taylor

When the IEEE-SA Standards Board approved this standard on 30 March 2000, it had the following membership:

Donald N. Heirman, *Chair*
James T. Carlo, *Vice Chair*
Judith Gorman, *Secretary*

Satish K. Aggarwal
Mark D. Bowman
Gary R. Engmann
Harold E. Epstein
H. Landis Floyd
Jay Forster*
Howard M. Frazier
Ruben D. Garzon
James H. Gurney
Richard J. Holleman
Lowell G. Johnson
Robert J. Kennelly
Joseph L. Koepfinger*
Peter H. Lips
L. Bruce McClung
Daleep C. Mohla
James W. Moore
Robert F. Munzner
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Akio Tojo
Donald W. Zipse

*Member Emeritus

Also included is the following nonvoting IEEE-SA Standards Board liaison:

Alan Cookson, *NIST Representative*
Donald R. Volzka, *TAB Representative*

Greg Kohn
IEEE Standards Project Editor

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Contents

1. Overview 1

1.1 Scope 2
1.2 Purpose 2

2. References 3

3. Abbreviations, acronyms, and definitions 3

3.1 Definitions 3
3.2 Abbreviations and acronyms 5

4. Safety verification approach 5

4.1 Concept-level verification [Figure 2(A)] 7
4.2 Functional-level verification [Figure 2(B)] 8
4.3 Implementation-level verification [Figure 2(C)] 8

5. Safety verification tasks 8

5.1 Safety verification plan 13
5.2 External inputs 13
5.3 Concept-level tasks 15
5.4 Functional-level tasks 17
5.5 Implementation level tasks 18
5.6 Document work products 19

Annex A (informative) Verification techniques/procedures/analyses 20

Annex B (informative) Example 24

Annex C (informative) Bibliography 28

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

This page is intentionally blank.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

IEEE Standard for Verification of Vital Functions in Processor-Based Systems Used in Rail Transit Control

1. Overview

This standard defines a method for the identification and subsequent verification of vital functions implemented in processor-based equipment used in safety-critical applications on rail and transit systems. This standard requires the production of analyses and other supporting documentation necessary to demonstrate the achievement of the established safety goals.

This standard is limited to verifying fail-safe implementation of functions in processor-based equipment. This includes the identification of the functions that must be implemented fail-safely, the allocation of those functions to specific hardware and/or software components of the system, the identification of how those hardware and/or software components are expected to achieve the desired fail-safe operation (safety assurance concept), and the verification that the hardware and/or software component implements the functions in a fail-safe manner (see 3.1.2 for the definition of fail-safe used in this standard). This standard is intended to complement the execution of a total system safety program, and does not address all system safety issues. Detailed system, subsystem, and interface hazard analyses, hazard tracking, and risk assessment are considered part of the overall system safety program and, while elements of such hazard analyses may be required and/or referenced, generation of the analyses is not within the scope of this standard.

Throughout this standard the term system will be used. A system may be an entire transit or railroad control system, or it may be a subsystem or device, such as a track circuit or vehicle overspeed control unit. The application of this standard requires that the system or subsystem being verified is clearly defined, including its place within the overall system, and that all assumptions and interface requirements are clearly documented.

This standard is divided into five clauses. Clause 1 provides the scope and purpose of this standard. Clause 2 lists references to other standards that are useful in applying this standard. Clause 3 provides definitions that are either not found in other standards, or have been modified for use with this standard. Clause 4 describes the verification approach, dividing it into activities performed at three levels: functional, concept, and implementation. Clause 5 defines analyses and supporting documentation required to identify the imposed safety design requirements derived from each of the three levels, to allocate those safety requirements to the various components of the system, and, finally, to confirm that the safety goals have been attained.

The standard also contains three annexes. Annex A provides description and examples of techniques, analyses, and procedures that may be used in completing the concept-level tasks established in Clause 5. Annex B provides an example of the application of this standard. Annex C is a bibliography.

It is not intended that systems installed prior to the effective date of this standard comply with this standard.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

1.1 Scope

This standard provides a set of standard verification tasks for processor-based equipment used in safety-critical applications on rail and transit systems. The scope of this standard shall encompass, and be limited to, processes that verify the level of safety achieved in the implementation of safety-critical functions that are required to be fail-safe. This standard does not address quality assurance or validation processes, which also affect the level of overall system safety achieved.

Figure 1 illustrates the elements of the safety verification process (within the dotted box) in the context of the overall system safety and design and development processes.

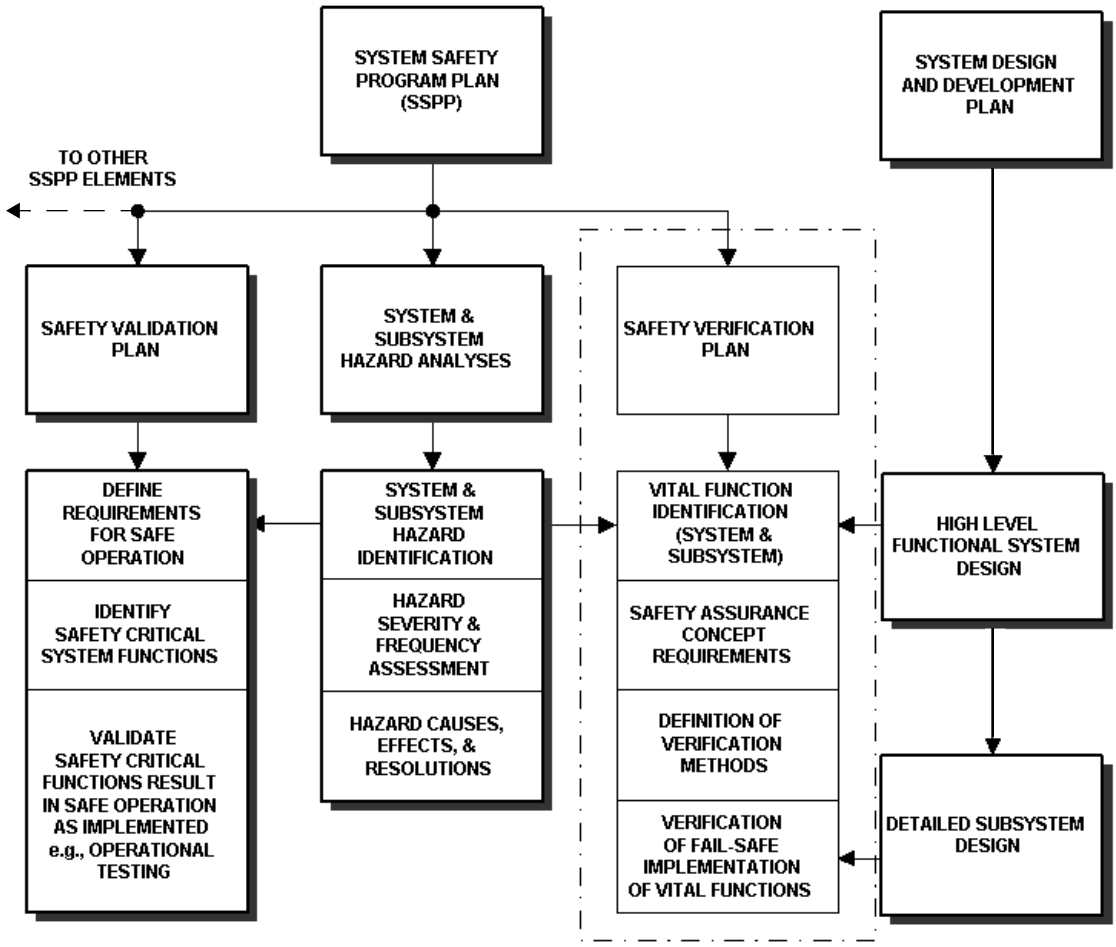

Figure 1—Safety verification standard context diagram

1.2 Purpose

The purpose of this safety verification process standard is to provide a well-defined and well-structured set of analysis methods and documentation that

— Fulfills the primary purpose of the verification process.
— Is flexible enough to accommodate all viable design methods.
— Satisfies the safety requirements of the end user.

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

This standard

— Defines a necessary and sufficient set of analyses at the concept, functional, and implementation levels that comprehensively identify and verify all functions required to be implemented as fail-safe.
— Defines responsibilities for equipment suppliers and end users.

2. References

This standard shall be used in conjunction with the following publication. If the following publication is superseded by an approved revision, the revision shall apply. In case of a conflict between this standard and the referenced document, this standard shall take precedence. Those provisions of the referenced document that are not in conflict with this standard shall apply as referenced.

MIL-Std-882C:1996, System Safety Program Plan Requirements.[1]

3. Abbreviations, acronyms, and definitions

3.1 Definitions

For the purposes of this standard, the following terms and definitions apply. IEEE 100-1996, The IEEE Standard Dictionary of Electrical and Electronic Terms [B13],[2] should be referenced for terms not defined in this clause.

3.1.1 concept level: The level of verification activities at which vital functions and vital implementation requirements, imposed on the system's design and implementation by the safety assurance concept selected, are determined and identified.

3.1.2 fail-safe: A design philosophy applied to safety-critical systems such that the result of hardware failure or the effect of software error shall either prohibit the system from assuming or maintaining an unsafe state, or shall cause the system to assume a state known to be safe.

3.1.3 fail-safely: The implementation of a function in a fail-safe manner.

3.1.4 fault tree analysis (FTA): A structured analysis method used to comprehensively identify faults and combinations of faults of software and hardware components as they relate to a hazard.

3.1.5 functional fault tree (FFT): A structured analysis method used to identify vital functions at the system functional level by comprehensively examining system functional faults that could precipitate hazards.

3.1.6 functional level: The level of verification activities at which vital system functions are identified from system functional and operational requirements.

3.1.7 hardware failure: A change in the characteristics of a system hardware element beyond its design tolerances.

3.1.8 hazard: An existing or potential condition that can result in a mishap.

[1]MIL publications are available from Customer Service, Defense Printing Service, 700 Robbins Ave., Bldg. 4D, Philadelphia, PA 19111-5094.

[2]The numbers in brackets correspond to those of the bibliography in Annex C.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

3.1.9 implementation level: The level of verification activities at which system components implementing vital functions are comprehensively identified and analyzed to verify that all functions identified as vital are implemented fail-safely.

3.1.10 mean time between hazardous events (MTBHE): The average time between occurrences of events, where hazardous events and the equipment that may precipitate them are defined at the system level. The hazardous events included in MTBHE are those whose consequences are of a given severity, as determined by the organization generating the safety goals.

3.1.11 mishap: An unplanned event or series of events resulting in death, injury, occupational illness, or damage to or loss of equipment or property, or damage to the environment; an accident.

3.1.12 safe: Having acceptable risk of the occurrence of a hazard.

3.1.13 safety assurance: A characteristic of the implementation of a system that assures a level of safe operation.

3.1.14 safety assurance concept: A design concept applied to processor-based systems that assures the fail-safe implementation of identified functions, including safe operation in the presence of hardware failures and/or software errors. Examples are: Checked Redundancy; Diversity and Self-Checking; Numerical Assurance; and N-Version Programming.

3.1.15 safety-critical: A term applied to a system or function, the correct performance of which is critical to safety of personnel and/or equipment; also a term applied to a system or function, the incorrect performance of which may result in an unacceptable risk of a hazard.

3.1.16 safety validation: A structured and managed set of activities that demonstrate that the system, as specified and implemented, performs the intended functions, and that those functions result in overall safe operation. Validation answers the question, “Did we build the right system?”

3.1.17 safety verification: A structured and managed set of activities that identify the vital functions required to be performed by the system, and demonstrate that the system, including its subsystems, interfaces and components, implements the vital functions fail-safely to a level that meets the allocated system safety goals. Verification answers the question, “Did we build the system right?”

3.1.18 self-revealing component failures: Component failures whose effects on system operation are immediately and clearly apparent to a properly trained person.

3.1.19 software error: An error in a software element which, when executed, results in unintended system operation.

3.1.20 system safety: The application of engineering and management principles, criteria, and techniques to optimize all aspects of safety within the constraints of operational effectiveness, time, and cost throughout all phases of the system life cycle.

3.1.21 system safety goals—quantitative: A quantitative limit of the probability and/or frequency with which any vital function fails to be implemented safely.

3.1.22 system safety program: The combined tasks and activities of system safety management and system safety engineering that enhance operational effectiveness by satisfying the system safety requirements in a timely, cost-effective manner throughout the system life cycle.

3.1.23 system safety program plan: A formal document that fully describes the planned safety tasks required to meet the system requirements, including organizational responsibilities, methods of

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

accomplishment, milestones, depth of effort, and integration with other program engineering and management functions.

3.1.24 unsafe: Having unacceptable risk of the occurrence of a hazard.

3.1.25 vital function: A function in a safety-critical system that is required to be implemented in a fail-safe manner.

NOTE—Vital functions are a subset of safety-critical functions.

3.2 Abbreviations and acronyms

ASCE	American Society of Civil Engineers
ATP	automatic train protection
CENELEC	European Committee for Electrotechnical Standardization
FFT	functional fault tree
FMEA	failure modes and effects analysis
FTA	fault tree analysis
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
MTBHE	mean time between hazardous events
PHA	preliminary hazard analysis
SSHA	subsystem hazard analysis
SSIHA	subsystem interface hazard analysis
SSn-FTA	subsystem fault tree analysis
SSPP	system safety program plan
SVP	safety verification plan

4. Safety verification approach

Comprehensive safety verification shall include the identification of vital functions and the verification that the identified vital functions have been implemented fail-safely to the degree required by the system safety goals.

The activities associated with safety verification shall be divided into the following three levels:

— Concept level
— Functional level
— Implementation level

Figure 2 illustrates the three levels and their relationships. These levels correspond to different areas of safety verification. Concept-level activities shall analyze the safety assurance concept employed, identify the concept-specific design requirements necessary for the fail-safe implementation of vital functions, and identify the concept-specific verification methods. Functional-level activities shall identify all user-specified system functions required to be implemented fail-safely. Implementation-level activities shall apply the verification methods determined at the concept level to the implemented system to ensure that the identified set of vital functions has been implemented fail-safely.

Software and hardware components implementing vital functions are thus identified from two sources; the user specified system functional requirements identified at the functional level, and concept-specific design

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

requirements imposed on the hardware and software design by the safety assurance concept identified at the concept level.

The results of the verification activities shall demonstrate achievement of the system safety goals.

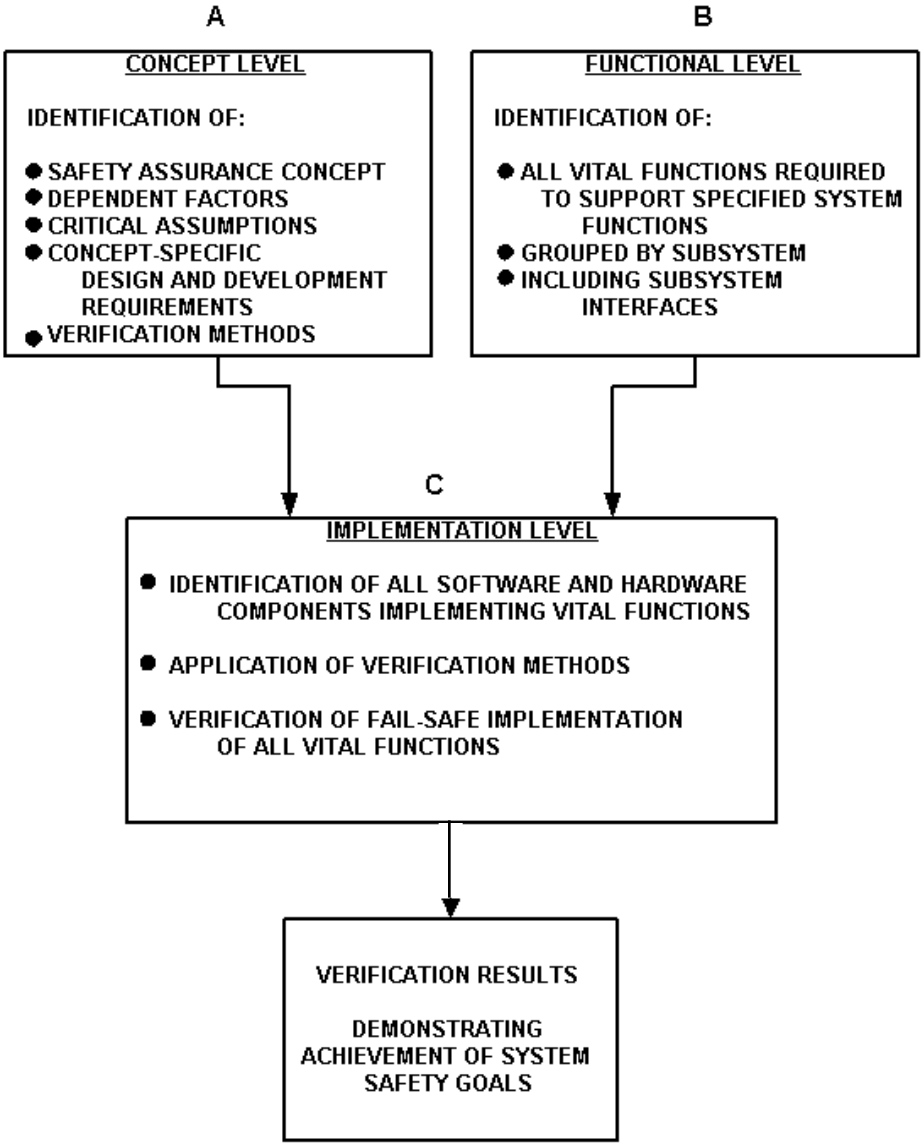

Figure 2—Approach to safety verification—Division of safety verification activities into levels

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

4.1 Concept-level verification [Figure 2(A)]

Activities at the concept level shall include

- Describing the "safety assurance concept(s)" employed to ensure fail-safe implementation of vital functions.
- Comprehensive identification of factors specific to the concept upon which the assurance of fail-safe implementation is dependent.
- Defining the concept-specific design requirements necessary for fail-safe implementation.
- Identifying verification methods to be applied at the implementation level that verify that the concept-specific design requirements have been met and that vital functions have been implemented fail-safely.

Safety assurance concepts used to design and implement vital functions in processor-based systems include, but are not limited to

- Checked Redundancy
- Numerical Assurance
- N-Version Programming
- Diversity
- Self-Checking

In addition, fail-safe discrete circuit design used to implement vital input/output functions in processor-based systems is also identified as a "safety assurance concept" (refer to Annex A).

NOTE—An example of a concept-specific design requirement imposed by the safety assurance concept Checked Redundancy would be that vital parameters and results of vital functions performed by the redundant systems must be periodically compared by hardware and/or software elements to detect discrepancies in the results and take action to assure safe operation. This concept-specific fail-safe requirement is not imposed by the user specified system functional requirements; however, it must be identified as a requirement and incorporated into the implementation for the assurance of safe operation.

Each safety assurance concept has an associated set of factors upon which the fail-safe implementation of vital functions is dependent, that is, "dependent factors," and a set of critical assumptions. The dependent factors shall be identified and used to subsequently identify concept-specific design and development requirements. The critical assumptions shall be identified and included in the concept level documentation.

NOTE—Again using Checked Redundancy as an example, dependent factors include: the ability of the comparison mechanism to detect all discrepancies that affect safe operation; frequency of comparison; the extent to which failures in one redundant element are detectable at the points of comparison; executable software correctness; and others.

A Checked Redundancy concept-specific design requirement related to the dependent factor, frequency of comparison, would be that the interval between comparison of the outputs of each redundant unit be implemented fail-safely such that it is no greater than a specified time.

One example of a concept-specific development requirement associated with the dependent factor "software correctness" would be that the software is developed in accordance with a process that ensures a specified level of correctness.

Verification methods shall be identified at the concept level. Verification methods shall comprise analysis, demonstration, test, and/or inspection applied to products and processes, as required, to verify that the processor-based system fail-safely implements all vital functions, both those identified at the concept level and those identified at the functional level, to the degree required by the system safety goals.

--`,,`-`-`,,`,,`,`,,`---
Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

4.2 Functional-level verification [Figure 2(B)]

Verification activities performed at the functional level shall include analyses that comprehensively identify the safety-critical functions associated with the performance of user-specified system functions implemented by the processor-based equipment. Safety-critical functions shall be identified using a functional fault tree (FFT). A subset of the fault events identified in the FFT shall be associated with functions that shall be implemented fail-safely to avoid the occurrence of those fault events or protect the system against the effects of those fault events when they occur. This set of associated functions shall comprise the vital functions.

The FFT shall be developed in three levels. The top level of the functional fault tree (FFT1) shall identify system fault events based on PHA identified hazards, prior to the allocation of system functions to individual subsystems. The second level of the functional fault tree (FFT2) shall continue the identification of faults grouped according to subsystem function allocation. The third level of the functional fault tree (FFT3) shall include the identification of faults associated with the interfaces between subsystems.

The terminal fault events of the FFT shall include, per subsystem, all faults for which functions shall be implemented in a fail-safe manner (vital functions) to prevent and/or protect against the occurrence of those faults.

4.3 Implementation-level verification [Figure 2(C)]

Verification activities performed at the implementation level shall include the identification of the hardware and software components involved in the implementation of the identified vital functions within each subsystem. In addition, these activities shall include the application of previously defined concept-specific verification methods (4.1) to verify that the identified vital functions have been implemented fail-safely to the degree required by the system safety goals.

Verification methods shall also be applied to those concept-specific development activities previously identified.

5. Safety verification tasks

This clause describes in detail the tasks that shall be performed at each verification level (concept, functional, implementation) and the associated documentation required for safety verification of the processor-based portion of a system. Figure 3 illustrates the overall safety verification approach used by this standard and its relationship to the overall system safety program plan (SSPP).

Graphical flow diagrams describing the safety verification tasks are shown in Figure 4 through Figure 6. Plain blocks indicate documents, analyses, or activities performed for this standard. Blocks with shadow indicate documents, analyses, or activities referenced external to this standard. Dotted lines indicate an external relationship to items in the standard. Solid lines indicate a firm relationship between items in the standard. The identification numbers associated with each block on Figure 4 through Figure 6 reference specific process activities that are described in this clause. The order in which process activities are numbered and described does not mandate the order in which they are performed. Process activities are performed in an order consistent with the overall system development and when required inputs are available. Figure 4 illustrates the concept-level verification process linked to Figure 6, which shows a standard implementation-level verification process. Figure 5 illustrates the functional-level verification process that is also linked to Figure 6.

These verification process activities shall culminate in a safety verification final report confirming that identified vital functions have been implemented fail-safely to the degree required by the system safety goals. This report shall provide references to all supporting documentation.

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Safety verification documentation, described below, shall be "living documents" and, as such, shall be amended, updated, and configuration managed during the lifecycle of the system.

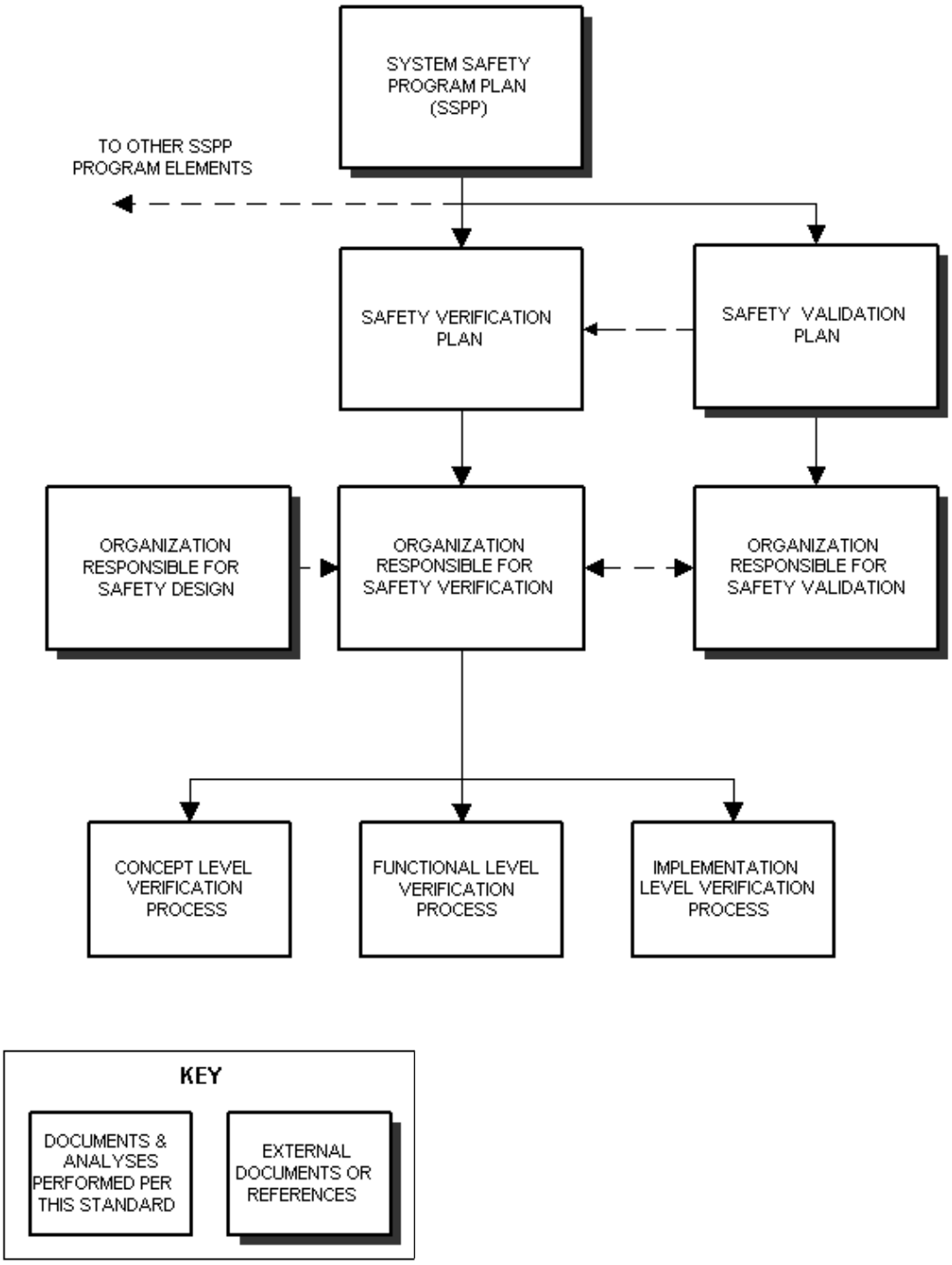

Figure 3—Safety verification—Standard safety verification plan

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

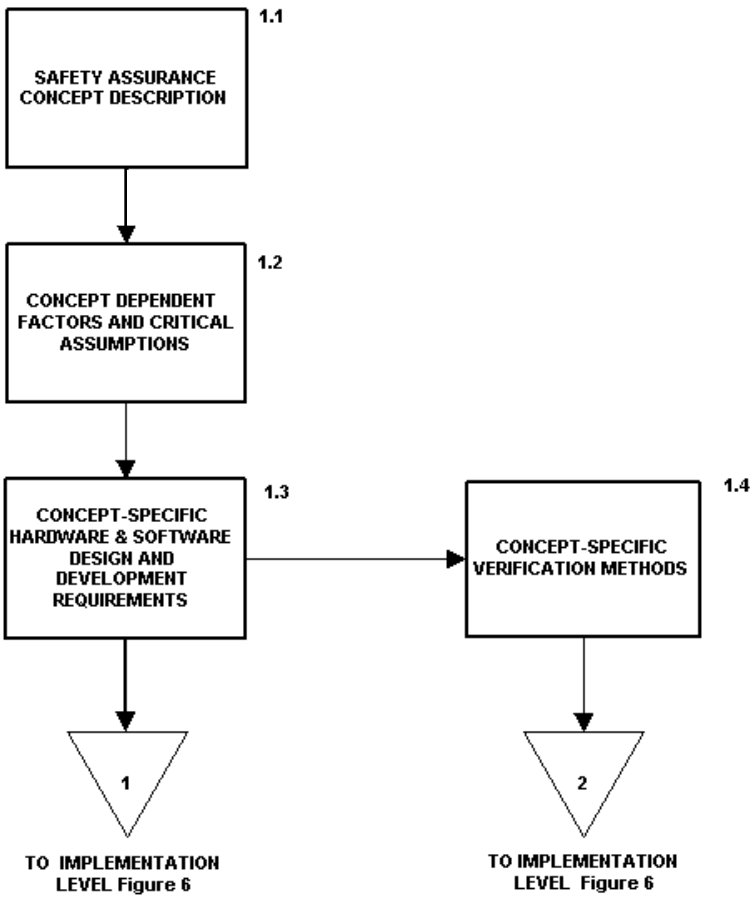

Figure 4—Safety verification—Standard concept-level verification tasks

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

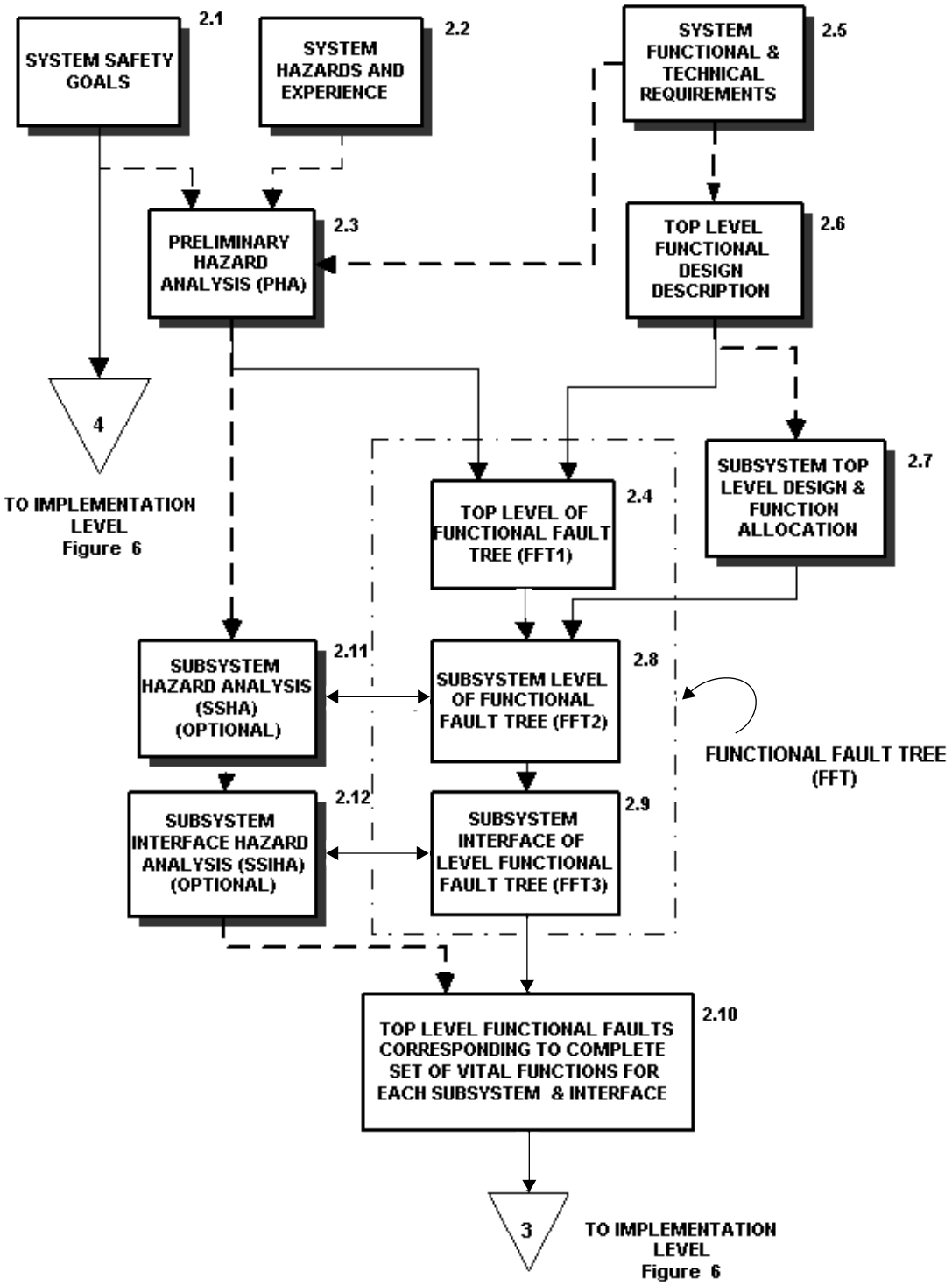

Figure 5—Safety verification—Standard functional-level verification tasks

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

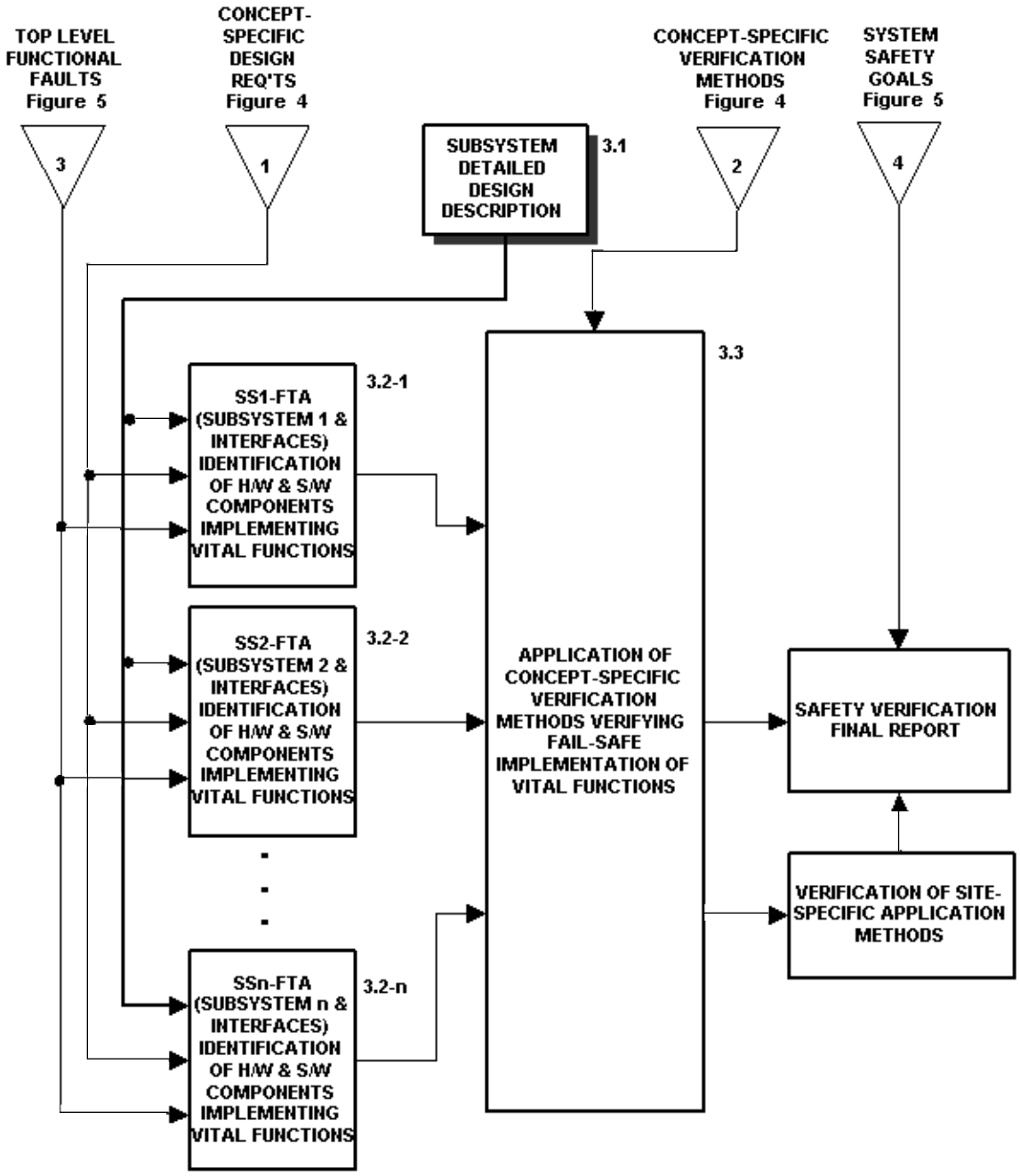

Figure 6—Safety verification—Standard implementation-level verification tasks

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

5.1 Safety verification plan

A safety verification plan shall be developed that describes in detail the tasks, resources, and processes to be performed in accordance with the safety verification tasks described in 5.3–5.5 and in this subclause, including their interrelationships and objectives. The safety verification plan shall also include justification for any of the tasks in this subclause not being completed and identify the organizations responsible for the verification activities and their relationship to design and validation organizations. The safety verification plan may be a distinct stand-alone document or may be incorporated as a distinct section of the overall SSPP.

Figure 3 shows the safety verification plan as part of the overall system safety program plan. The safety verification plan shall contain a description of the verification process to be performed at the concept, functional, and implementation levels, and a description of the responsible safety organizations involved.

5.2 External inputs

The following external inputs required to perform the safety verification tasks are generated outside the scope of this standard. The input required for each safety verification task is described in 5.2.1–5.2.9. The source of the external inputs shall be identified in the safety verification plan (SVP).

5.2.1 System safety goals (Figure 5, block 2.1)

The system safety goals document defines the safety goals of the system being verified using this standard. The system safety goals shall be expressed quantitatively. It is the responsibility of the organization generating the safety goals to state them in terms to which the results of the verification process can be compared.

NOTE—For example, a quantitative requirement that a processor based signal control system "provide a Mean Time Between Hazardous Events (MTBHE) of at least 10X operating hours" should give sufficient additional information defining the "scope" of the system under consideration, so that an MTBHE target for each subsystem under verification can be determined.

It is understood that the quantification of the probability or frequency of occurrence of elements within the safety verification process may be required to be subjectively (i.e., qualitatively) determined. All subjectively determined elements shall be identified, and the rationale and justification for the estimation shall be described.

5.2.2 System hazards and experience (Figure 5, block 2.2)

The development of a hazard analysis requires knowledge of the operating environment and functionality of the system, including the influences of system operation, maintenance, and procedures. Such knowledge may be gained from experience with other similar systems, results of similar analyses, and/or understanding of the system's operational requirements. This background information shall be used during hazard analysis, to support identification of the top level hazards associated with the system being analyzed.

5.2.3 System requirements (Figure 5, block 2.5)

The system requirements provide a complete and comprehensive description of all functions that the system is required to perform, and all associated technical requirements, including a clear definition of the scope of the system and its external interfaces. This information is used in the development of the preliminary hazard analysis.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

5.2.4 Top-level functional design description (Figure 5, block 2.6)

The top-level functional design description defines the top-level system design in terms of its functional architecture. This information, in conjunction with the results of the preliminary hazard analysis (PHA), shall be used to develop the top level of the functional fault tree (FFT1).

5.2.5 Subsystem top-level design and function allocation (Figure 5, block 2.7)

The subsystem top-level design and function allocation describes the design architecture to the subsystem level and allocates system functional requirements to the appropriate subsystems. This information shall be used, in conjunction with the results of the top level of the functional fault tree (FFT1), to complete the functional fault tree at the subsystem level (FFT2).

5.2.6 Subsystem detailed design description (Figure 6, block 3.1)

The subsystem detailed design description provides the detailed subsystem hardware and software design description for the implementation of the system functional requirements. This information shall be used, in conjunction with the faults identified at the subsystem and subsystem interface level of the functional fault tree (FFT2 and FFT3), to allocate safety requirements to the lowest levels of the system.

5.2.7 Preliminary hazard analysis (PHA) [Figure 5, block 2.3]

The PHA identifies the top-level faults that could lead to a hazardous situation. The PHA analyzes the functional requirements of the system to determine the top-level functional hazards.

The highest-level hazards developed in the PHA are those events that could result in a mishap, such as “trains collide,” “train derails,” and “passenger/staff injury,” should a fault in the system occur. The functional hazards and faults that precipitate the hazards identified in the PHA shall be used as the initial faults in the top level of the functional fault tree (FFT1) analysis.

5.2.8 Subsystem hazard analysis (SSHA) (Figure 5, block 2.11)

If the SSHA is performed, the safety-critical hazards and their causes and effects shall be cross checked with the FFT at the FFT2 level to ensure comprehensive identification of faults.

NOTE—The SSHA identifies and describes all safety-critical hazards associated with each subsystem, the hazard causes and effects, their potential resolution, and an assessment of their risk.

The SSHA is a traditional element in the overall SSPP based on MIL-STD-882C:1996. It may or may not be required for a given system. The SSHA is noted in this standard as an analysis document that may act as an adjunct to the FFT at the FFT2 level.

The SSHA contains descriptions of safety-critical system hazards, not all of which are resolved by the fail-safe implementation of system functions. However, the faults identified at the FFT2 level, the vast majority of which are by definition to be resolved by fail-safe implementation, should represent a subset of those identified and described in the SSHA.

The SSHA should be used as a vehicle for describing FFT2 faults in detail, associating them with proposed resolutions, and assessing their risk. Here, the risk assessment of a fault to be mitigated by design may be posed as a target limit of the frequency of occurrence. (It is assumed that the severity element of the risk assessment for the hazards precipitated by such faults is the highest severity.)

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

5.2.9 Subsystem interface hazard analysis (SSIHA) (Figure 5, block 2.12)

If the SSIHA is performed, the safety-critical hazards and their causes and effects shall be cross checked with the FFT at the FFT3 level to assure comprehensive identification of faults.

NOTE—The SSIHA identifies and describes all safety-critical hazards associated with the interfaces between and among all subsystems, the hazard causes and effects, their potential resolution, and an assessment of their risk.

The SSIHA contains descriptions of safety-critical system hazards, not all of which are resolved by the fail-safe implementation of system functions. However, the faults identified in the FFT at level FFT3, the vast majority of which are by definition to be resolved by fail-safe implementation, should represent a subset of those identified and described in the SSIHA.

The SSIHA should be used as a vehicle for describing FFT3 faults in detail, associating them with proposed resolutions, and assessing their risk. Here, the risk assessment of a fault to be mitigated by design may be posed as a limit of the frequency of occurrence. (It is assumed that the severity element of the risk assessment for the hazards precipitated by such faults is the highest severity.)

5.3 Concept-level tasks

The SVP shall require the completion of all the tasks in this subclause and the generation of the associated documentation.

Annex A provides examples of procedures, analyses, and tests, along with additional information to aid in the completion of the following tasks.

5.3.1 Safety assurance concept description (Figure 4, block 1.1)

5.3.1.1 Objectives

The objective of this task is to identify and describe in detail the safety assurance concept(s) to be used in the design and implementation of each subsystem's hardware and software elements. One or more safety assurance concepts may be applied to the design of different subsystems, all of which must be described.

5.3.1.2 Requirements

This documentation shall describe in detail how the selected safety assurance concept(s) are used to obtain safe operation, provide sufficient detail as to how the system designed using the concept is intended to perform, and describe the characteristics or attributes of the concept that are critical to achieving safe operation. The safety assurance concept description shall address, as a minimum, characteristics of the concept that mitigate the following list of potential faults related to processor-based systems:

- Incorrect or ambiguous system inputs
- Inputs not present at the prescribed time
- Incorrect system outputs
- Outputs not present at the prescribed time
- System output when necessary inputs are not present
- Changed volatile and nonvolatile memory contents
- Changed clock rate
- Data not current
- Improper program flow (for example: failure to exit from loop, entry to or exit from a routine at the wrong time, illegal or invalid execution of a program segment, skipping program segments)
- Improper or incorrect instruction execution (for example: arithmetic error, sign error)

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

— Latent faults
— Common cause faults
— Transient faults
— Other faults particular to the specific technology

The documentation shall provide or reference verification methods (5.3.4) for evidence that the concept will achieve the described safety assurance when implemented properly.

5.3.2 Concept-dependent factors and critical assumptions (Figure 4, block 1.2)

5.3.2.1 Objectives

The objective of this task is to comprehensively identify and describe, for each concept used, critical assumptions made and factors upon which the fail-safe implementation of vital functions is dependent. The list of dependent factors shall form the basis for determining the verification methods (5.3.4) that shall be used to verify the complete and correct implementation of each vital function.

5.3.2.2 Requirements

For each safety assurance concept employed, provide a detailed written description of each dependent factor. Identify and explain critical assumptions made upon which safe operation is dependent.

5.3.3 Concept-specific hardware and software design and development requirements [Figure 4, block 1.3]

5.3.3.1 Objectives

The objective of this task is the identification of safety-related design and development requirements imposed by the safety assurance concept(s) being used.

5.3.3.2 Requirements

This document shall describe in detail all safety-related design and development requirements affecting hardware and software components necessary to achieve fail-safe implementation of vital functions using the safety assurance concept(s), including those imposed by the dependent factors. The requirements shall be presented in a matrix or similar form to be used for verification of safety. Each requirement shall clearly identify the hardware and/or software subsystem for which it will be applied.

5.3.4 Concept-specific verification methods (Figure 4, block 1.4)

5.3.4.1 Objectives

The objective of this task is to describe the methods that shall be used to verify that all dependent factors have been satisfied, all safety-related design and development requirements have been met, and that each hardware and software component involved in the fail-safe implementation of vital functions meets or exceeds the minimum safety goals for the system.

5.3.4.2 Requirements

This document shall describe in detail the verification methods, comprising analysis, demonstration, test, and/or inspection applied to the product or process. This shall include the necessary inputs, the form of the output, and the relationship each plays in fulfilling the verification of fail-safety. The descriptions shall provide sufficient detail to allow understanding of the method used, its scope, the format of the output, and the

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

significance of the results. The verification methods shall be adequate to verify at the implementation level that

— The safety assurance concept performs as described (5.2.1).
— The dependent factors (5.2.2) are satisfied.
— The identified hardware and software design and development safety-related requirements (5.2.3) are met.
— The selected safety assurance technique mitigates the effects of all hardware failures and/or software errors.
— All vital functions identified at the functional level have been implemented fail-safely.

5.4 Functional-level tasks

The tasks performed at this level identify the system vital functions, that is, those that must be implemented fail-safely, based on the user-defined functional requirements for the system. Tasks shall be performed as necessary to ensure that the derived set of functional faults is complete and comprehensive. The terminal events of the FFT identify functional faults that could affect safe system operation. Those functions implemented in the processor-based system that protect the system from the effects of these faults form the set of vital functions.

5.4.1 Functional fault tree—Top level (FFT1) (Figure 5, block 2.4)

5.4.1.1 Objectives

The objective of this task is to identify the functional faults at the system level (prior to subsystem allocation) that could precipitate the PHA identified hazards.

5.4.1.2 Requirements

The FFT1 shall identify, in a structured fault tree format, the functional faults which may result in PHA identified hazards, considering the top-level functional system design. Each hazard or system fault that is identified from the PHA is decomposed, to a level not dependent upon subsystem functional allocation.

5.4.2 Functional fault tree—subsystem level (FFT2) (Figure 5, block 2.8)

5.4.2.1 Objectives

The objective of this task is to identify the functional faults in each of the subsystems that could precipitate the terminal faults identified in FFT1.

5.4.2.2 Requirements

The FFT2 shall describe, in a structured fault tree format, the functional faults in each of the subsystems that could precipitate the terminal faults identified in the FFT1. This analysis makes reference to the system's breakdown of functions into subsystems for design implementation, which is illustrated in Figure 5 as element 2.7.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS Not for Resale

5.4.3 Functional fault tree—Subsystem interface level (FFT3) (Figure 5, block 2.9)

5.4.3.1 Objectives

The objective of this task is to identify the functional faults in each of the subsystem interfaces that could precipitate the terminal faults identified in FFT1 and FFT2.

5.4.3.2 Requirements

The FFT3 shall describe, in a structured fault tree format, the functional faults in each of the subsystem interfaces that could precipitate the terminal faults identified in the FFT1 and FFT2.

NOTE—Figure B.1 in Annex B illustrates an example FFT.

5.5 Implementation-level tasks

The tasks performed at this level

a) Identify the hardware and software components in each subsystem involved in the implementation of vital functions.
b) Identify the application of the concept specific verification methods defined at the concept level to those hardware and software components.
c) Verify fail-safe implementation of vital functions based on items a) and b).

All of the following tasks, with the exception of 5.5.4, are required. Task 5.5.4 is required for systems that have site-specific application data added after the completion of these safety verification tasks.

5.5.1 Subsystem fault tree analyses (SSn-FTA) (Figure 6, block 3.2-n)

5.5.1.1 Objectives

The objective of this task is to identify the hardware and software components implementing the vital functions within each subsystem.

5.5.1.2 Requirements

A detailed FTA shall be performed for each subsystem, including the subsystem interfaces. The FTA shall identify all subsystem components required to implement the set of vital functions defined at the functional level. The top-level faults of each subsystem FTA shall be identified from the lowest level FFT faults related to that subsystem and its interfaces.

5.5.2 Fail-safe software and hardware component verification (Figure 6, block 3.3)

5.5.2.1 Objectives

The objective of this task is to verify that the subsystem hardware and software components identified as performing vital functions implement those functions fail-safely.

5.5.2.2 Requirements

This document shall describe in detail the application of the verification methods developed in 5.3.4 to all hardware and software components identified as implementing vital functions, and the results of the verification analyses, tests, and/or processes. These methods may include standard analysis techniques or

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

concept-specific techniques that serve to verify that the hardware/software implementation is sufficient to satisfy the requirements imposed by the concept dependent factors.

There are many types of analyses and/or tests that may be used to verify that the implemented design provides fail-safe operation. The types of tests and analyses required will be dictated by the safety concept(s) being employed and the complexity of the system/subsystem being verified.

5.5.3 Safety verification final report (Figure 6, block 3.4)

5.5.3.1 Objectives

The objective of this task is to determine the degree of safety assurance achieved in the implementation of the system based on the results of the verification performed in 5.5.2, and confirm that it meets or exceeds the system safety goals.

5.5.3.2 Requirements

A system safety verification report shall be completed that includes or references the results of all safety verification activities. The results of these activities shall be expressed quantitatively. These verification results shall be incorporated into the calculation of the system MTBHE. All subjectively determined elements shall be identified and the rationale for their quantitative estimation described.

5.5.4 Verification of site-specific application methods analysis (Figure 6, block 3.5)

5.5.4.1 Objectives

The objective of this task is to identify and describe verification methods required to verify correct implementation of site specific application data or configuration information.

5.5.4.2 Requirements

Methods shall be described and implemented to verify that the means used for incorporating site-specific application data and configuration information into the system and/or subsystem does not adversely affect the safe operation of the system.

NOTE—Validation activities outside the scope of this standard are required to ensure that the site-specific application data and configuration information result in safe operation. The verification methods described here are to ensure that the process used to incorporate this data into the system does not adversely affect safety.

5.6 Document work products

The following is a minimum list of document work products that shall be produced:

- Safety verification plan (5.1)
- Safety assurance concept description (5.3.1)
- Concept-dependent factors and critical assumptions (5.3.2)
- Concept-specific hardware and software design and development requirements (5.3.3)
- Verification methods (5.3.4)
- Functional fault tree (5.4.1, 5.4.2, and 5.4.3)
- Subsystem fault tree analyses (5.5.1)
- Fail-safe software and hardware component verification (5.5.2)
- Safety verification final report (5.5.3)

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Annex A

(informative)

Verification techniques/procedures/analyses

A.1 Introduction

This annex provides descriptions and examples of techniques, analyses, and/or procedures used in the concept-level safety verification tasks defined in 5.3. The information is not intended to be a complete list of all possible methods. It is provided for guidance and to clarify what is required by these tasks.

A.1.1 Safety assurance concept description

The description of the safety assurance concept (5.3.1) contains information describing the manner in which a system, designed using the concept, mitigates the effects of hardware failures and/or software errors. This document should include all the necessary supporting information to allow the reviewer to determine that the safety assurance concept, if implemented, as described, effectively mitigates the effects of hardware failures and/or software errors. Failures and/or errors, which have been verified to be mitigated by the safety assurance concept being used, need not be further analyzed in the FTAs. It is necessary at the implementation level that the safety assurance concept's correct implementation be verified. For example, if the safety assurance concept document includes a description of how the concept protects against addressing errors or failures and supporting evidence is provided to back up the claim, it is not necessary to consider this failure mode as part of the FTAs. However, it is necessary to verify that the concept as implemented includes the feature described.

In many cases, the safety assurance concept descriptions and supporting documentation can be reused from system to system. With comprehensive descriptions and verification of concept effectiveness, it should be possible to apply the concepts to other similar systems.

Any possible hardware failures or software errors not addressed by the concept description must be included in the FTAs and thus will result as software and/or hardware safety-imposed design requirements.

A.1.2 Concept-dependent factors and critical assumptions

The safety assurance concepts listed in 4.1 are described in A.1.2.1–A.1.2.5. For each concept, a minimum list of dependent factors is provided. The factors listed represent basic concept dependencies that must be adequately addressed and resolved in the design and implementation of the system.

A.1.2.1 Safety concept: Intrinsic fail-safe design

This concept addresses the design of fail-safe hardware circuits or systems employing discrete mechanical and/or electrical components. Verification of fail-safe operation for systems designed using this concept requires that the effect of every relevant failure mode of each component, and relevant combinations of component failure modes, be considered, analyzed, and documented in a comprehensive failure modes and effects analysis (FMEA).

The verification of fail-safe operation of an intrinsic fail-safe design via FMEA relies upon the comprehensive identification of all possible failure modes of each discrete electronic and/or mechanical component. Several components have been accepted by the signaling industry as exhibiting verifiable failure

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

characteristics, that is, specific failure modes are assumed not to occur. (For example: wire-wound resistors, MIL-type RWR-79 manufactured to standard MIL-R-26E have been accepted as not having an increasing resistance failure mode except for a possible abrupt change to 500 kΩ or greater, provided they are applied in a circuit that limits their power dissipation to less than 10% of their rated power dissipation. Four-terminal capacitors that are properly constructed and installed are accepted as not having an open circuit failure mode—the capacitor will not fail open circuit without opening the circuit path obtained by the four-terminal configuration.)

A.1.2.1.1 Dependent factors

The degree of safety achieved is dependent upon

a) The correctness of selected component failure characteristics.
b) The comprehensive and accurate identification of all component failure modes.
c) The extent to which all combinations of failure modes can be analyzed.

A.1.2.1.2 Critical assumption

A critical assumption associated with this concept is that multiple, independent self-revealing component failures will not occur simultaneously (within the time it takes for the first failure to occur and safe action to be taken).

A.1.2.2 Safety concept: Checked Redundancy

The concept of Checked Redundancy is typified by the use of two or more identical, independent hardware units, executing identical software and performing identical functions. A means is provided to periodically compare vital parameters and results of the independent redundant units, requiring agreement of compared parameters to assert or maintain a permissive output. If the units do not agree, safety-critical functions and outputs default to a known safe state.

A.1.2.2.1 Dependent factors

The degree of safety achieved is dependent upon the following factors:

a) The ability of the checking process to provide permissive outputs only if the redundant systems agree, that is, the vitality of the comparison mechanism.
b) The degree to which the checking process includes comparison of all vital parameters and of all results of vital functions.
c) The ability to place and maintain the system in a known safe state under any failure which could affect safety.
d) The degree to which the effects of common mode failures have been eliminated, that is, the degree of independence achieved between redundant hardware units.
e) The ability to detect all hardware failures affecting safe operation in either redundant unit at the point of comparison.
f) The frequency with which the checking process between redundant units is performed.
g) The degree to which the software employed by each redundant unit is either error-free or does not contain identical errors.

--`,,`-`-`,,`,,`,`,,`---
Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

A.1.2.2.2 Critical assumption

A critical assumption associated with this concept is that independent hardware failures in each of the redundant processing units that produce the same erroneous unsafe effect will not occur simultaneously, that is, within the interval between correspondence checks.

A.1.2.3 Safety concept: N-Version Programming

This concept requires at least two software programs, executing together, performing identical functions. The software programs are written by independent teams, using different tools. The multiple independently-written software programs comprise multiple redundant systems, executing either on separate redundant hardware units (which may or may not be identical) or residing in one hardware unit. A means is provided to compare the results and output states of the multiple redundant software systems. If the systems do not agree, then the safety critical function being performed defaults to a known safe state.

A.1.2.3.1 Dependent factors

The degree of safety achieved is dependent upon the following factors:

a) The ability of the checking process to provide permissive outputs only if the parallel programmed systems agree, that is, the vitality of the comparison mechanism.
b) The degree to which the checking process includes comparison of all vital parameters and of all results of vital functions.
c) The ability to place and maintain the system in a known safe state under any failure which could affect safety.
d) The degree of safety achieved is dependent upon the degree to which the effects of common mode failures have been eliminated.
e) The ability to detect all software errors and/or hardware failures affecting safe operation in any redundant unit at the point of comparison.
f) The frequency with which the checking process between redundant units is performed.
g) The degree of independence achieved for the software employed by the parallel programmed systems.

A.1.2.3.2 Critical assumption

A critical assumption associated with this concept is that two or more software systems, which have been independently designed and coded to perform functions defined from a common high level specification, will not contain identical errors.

A.1.2.4 Safety concept: Diversity and Self-Checking

This concept requires that all critical functions be performed in diverse ways, using diverse software operations and/or diverse hardware channels, and that critical hardware be tested with Self-Checking routines. Permissive decisions are allowed only if the results of the diverse operations correspond and the Self-Checking reveals no failures.

A.1.2.4.1 Dependent Factors

The degree of safety achieved is dependent upon the following factors:

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

a) The ability of the checking process to provide permissive outputs only if the diverse results agree.
b) The degree to which the diverse operation and Self-Checking processes include verification of all vital results of all vital functions.
c) The unit's ability to place and maintain the system into a known safe state under any failure which could affect safety.
d) The comprehensiveness of the Diversity and Self-Checking process.
e) The degree to which the software employed is free of common mode errors.

A.1.2.5 Safety concept: Numerical Assurance

This concept requires that the state of each system vital parameter be uniquely represented by a large encoded numerical value, such that permissive results are calculated by pseudo-randomly combining the representative numerical values of each of the critical constituents of a permissive decision. Vital algorithms are entirely represented by data structures containing numerical values with verified characteristics, and no vital decisions are made in the executing software.

A.1.2.5.1 Dependent Factors

The degree of safety achieved is dependent upon the following factors:

a) The size and uniqueness of values representing system parameters.
b) The correctness of the numerical characteristics embodied in the data structures.
c) The degree to which the system can be verified to *not* contain prestored values representing permissive results.
d) The ability of the unit to place and maintain the system into a known safe state if there exists any lack of correctness of any of the numerical results.
e) The comprehensiveness of the coverage of the numerical values and the frequency at which the numerical values are calculated and verified.
f) The probability that the correct numerical result will not be obtained if the corresponding permissive conditions do not exist.
g) The degree to which the vital functions are correctly represented in the data structures.

A.1.2.5.2 Critical assumptions

Critical assumptions associated with this concept are that

a) Any failure or failures within the nonvital digital hardware will have a random effect on the generation of a permissive numerical result.
b) Any coding or design error within the executable software will have a random effect on the generation of a permissive numerical result.

As part of the Verification Methods (5.3.4) it is necessary to verify that the software and hardware components implementing fail-safe functions mitigate the set of dependent factors.

A.1.3 Concept-specific hardware and software design and development requirements (Figure 4, block 1.3)

The dependent factors identified for a given concept will, in many cases, impose requirements on the design and/or development processes such that the dependent factors may be satisfied.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

For example, satisfying the dependent factor of error-free software when using the Checked Redundancy or Diversity and Self-Checking concepts may be accomplished by instituting a strict software development process, rather than relying on analytical techniques, to prove correctness.

As another example, satisfying the dependence upon a vital mechanism to verify correspondence between two checked redundant units, may imply hardware and/or software design requirements to assure fail-safe correspondence checks.

The document must completely identify all concept-related design requirements necessary to satisfy the dependent factors.

A.1.4 Verification Methods [Figure 4, block 1.4]

The analysis methods required to verify that all vital functions have been implemented fail-safely are, to a large degree, concept dependent. The analysis methods defined in this standard must be sufficient to verify that the identified dependent factors have been satisfied, that the safety assurance concept is comprehensive, and that each hardware and/or software component performing fail-safe functions meets or exceeds the safety goals. For this discussion, it is assumed that safety requirements have been identified through a separate process (e.g., a comprehensive hazard analysis or FMEA).

The process and techniques used to demonstrate correctness of the software design and implementation falls into general categories of verification that are well described in existing standards work. It is not the intent of this subclause to duplicate the software verification techniques, processes, and methods that are well-described elsewhere.

The following partial list of applicable standards work for the industry domain of rail transit provides several formalized software development lifecycle phases along with the verification activities used to ensure that the software performing fail safe functions meets or exceeds the safety goals. The verification document may contain the specific verification methods described in the following bodies of work (see the bibliography of Annex C for more information on these documents):

— IEC 61508-1:1998 [B6]
— IEC CDIS 61508-2 [B7]
— IEC 61508-3:1998 [B8]
— IEC 61508-4:1998 [B9]
— IEC 61508-5:1998 [B10]
— IEC CDIS 61508-6 [B11]
— IEC 61508-7:1998 [B12]
— ATCS Specification 140, May 1995 [B2]
— prENV 50126-1998 (CENELEC) [B20]
— prENV 50128-1998 (CENELEC) [B21]
— prENV 50129-1998 (CENELEC) [B22]
— DOT-VNTSC-FRA-95-8.I [B3]
— DOT-VNTSC-FRA-95-8.I [B4]
— *Fault Tree Handbook* [B5]
— IEEE Std 610.12-1990 [B14]
— IEEE Std 730-1998 [B15]
— IEEE Std 828-1998 [B16]
— IEEE Std 1012-1998 [B17]
— IEEE Std 1059-1993 [B18]

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Annex B

(informative)

Example

This example is designed to assist in understanding the application of this standard to the verification of safety process. It is not intended that this example provide a complete application of the safety verification procedures; instead, it represents a sample of a specific application.

B.1 Functional requirement traced from PHA through FFT to FTA

Figure B.1 illustrates a sample of a FFT developed for an on-board automatic train protection (ATP) control system that is required to measure train speed, compare the speed to the speed limit, and activate train braking if overspeed. This example assumes the following:

a) That the ATP control system requirements specified that a manually set wheel wear switch be used as an input device to determine wheel diameter.
b) That the high-level, architectural design and detailed design of the ATP system has been completed.
c) That the following subsystems have been designed: “Speed Measurement,” “Speed Limit Processing,” and “Brake and Propulsion Control.”

The purpose of the FFT is to identify those functions of the example control system that must be implemented fail-safely, that is, the vital system functions. The FFT identified vital functions are grouped by implemented subsystems.

In this example, Element 1 of the figure (train derails) represents one of the high-level mishaps identified in the PHA. The elements identified as 2.x represent a set of hazard conditions that may result in the top-level mishap. For each level of fault identification in the example FFT, the subset of faults depicted represent a single subset of faults, which may be numerous.

The FFT proceeds to analyze and develop the complete set of faults that could precipitate each hazardous condition. In this example, the hazard element 2.2 (train is overspeed) is developed further, expanding into level 3.x faults.

Fault 3.2 (train speed is determined lower than actual speed) is further decomposed into level 4.x faults. Finally, the decomposition of fault 4.2 (wheel wear compensation incorrectly performed) results in the lowest-level faults for the hazardous condition.

Fault 5.1 is considered a terminal fault of the FFT in this example, since further development of the fault is dependent upon the implementation of the wheel wear processing. Fault 5.1 will be used as one of the high-level faults in the FTA, and corresponds to the vital function of processing the wheel wear switch information.

Fault 5.2 is also a terminal fault event in the FFT and will be further expanded in the FTA. Fault 5.2 corresponds to the vital function of reading the wheel wear switch. This fault implies that the functional design of the example control system requires fail-safe implementation of the hardware and software involved in the ATP system interface, as well as fail-safe design of the wheel wear switch.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

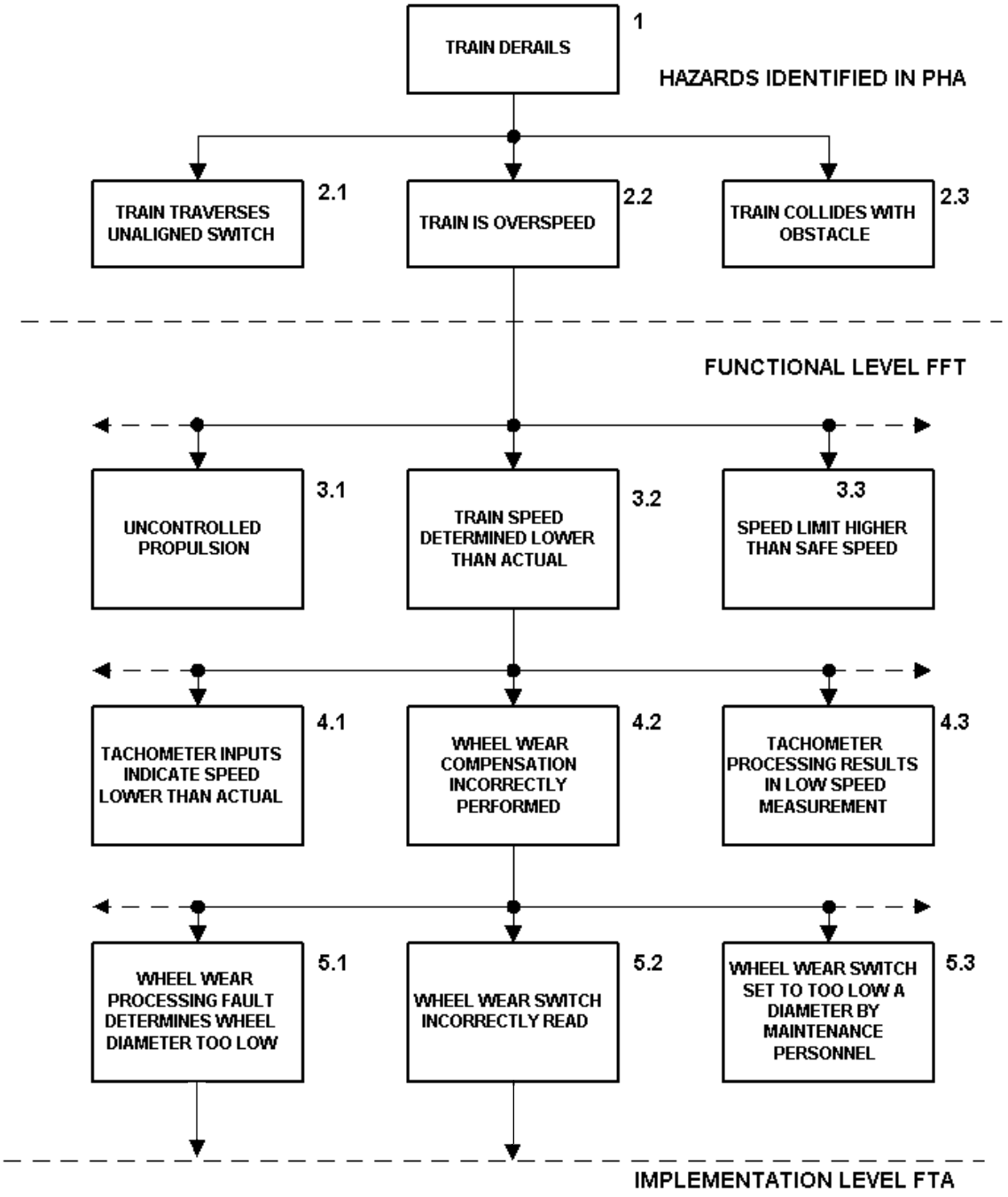

Figure B.1—Example FFT

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

Fault 5.3 is an example of a fault outside the functional domain of the example ATP control system, external measures must be identified for mitigation. In this example, this safety-critical function is identified as being mitigated by a maintenance procedure.

It is notable that each terminal fault in the FFT that can be mitigated by the design or implementation (faults 5.1 and 5.2) of the analyzed system corresponds to a vital system function. Traversing the FFT in reverse from the terminal faults identifies the system functions that at the highest functional level must be performed vitally.

For the example ATP control system, faults 5.1 and 5.2 identify functional faults of the “Speed Measurement” subsystem. Terminal faults derived from fault 3.1 (uncontrolled propulsion) would be associated with the “Braking and Propulsion Control” subsystem.

This example demonstrates the advantage of utilizing an FFT to identify vital system functions through analysis of faults precipitated by a hazard rather than through inspection of the design. FFT analysis provides a structured mechanism to identify and trace hazards to system functions, resulting in a comprehensive list of vital functions.

Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS Not for Resale

Annex C

(informative)

Bibliography

[B1] ASCE 21-96, Automated People Mover Standards, Chapter 3.

[B2] ATCS Specification 140, May 1995, Recommended Practices for Safety and Systems Assurance, Revision 4.0, Railway Association of Canada/American Association of Railroads.

[B3] DOT-VNTSC-FRA-95-8.I, Safety of High Speed Ground Transportation Systems: Analytical Methodology for Safety Validation of Computer Controlled Subsystems, Volume I: State-of-the-Art and Assessment of Safety Verification/ Validation Methodologies.

[B4] DOT-VNTSC-FRA-95-8.I, Safety of High Speed Ground Transportation Systems: Analytical Methodology for Safety Validation of Computer Controlled Subsystems, Volume II: Development of a Safety Validation Methodology.

[B5] *Fault Tree Handbook,* U.S. Nuclear Regulatory Commission, Report No. NUREG-0492, Jan. 1981.

[B6] IEC 61508-1:1998-12, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part 1: General requirements.

[B7] IEC CDIS 61508-2, Edition 1, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems.

[B8] IEC 61508-3:1998, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part3: Software requirements.

[B9] IEC 61508-4:1998, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part 4: Definitions and abbreviations.

[B10] IEC 61508-5:1998, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part 5: Examples of methods for the determination of safety integrity levels.

[B11] IEC CDIS 61508-6, Edition 1, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part 6: Guidelines on the application of Parts 2 and 3.

[B12] IEC 61508-7:1998, Functional safety of electrical/electronic/programmable electronic safety-related systems—Part 7: Overview of techniques and measures.

[B13] IEEE 100-1996, IEEE Standard Dictionary of Electrical and Electronics Terms. Sixth Edition.

[B14] IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology.

[B15] IEEE Std 730-1998, IEEE Standard for Software Quality Assurance Plans.

[B16] IEEE Std 828-1998, IEEE Standard for Software Configuration Management Plans.

[B17] IEEE Std 1012-1998, IEEE Standard for Software Verification and Validation Plans.

Copyright © 2000 IEEE. All rights reserved.
Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS
Not for Resale

[B18] IEEE Std 1059-1993. IEEE Guide for Software Verification and Validation Plans.

[B19] Leveson, N. G., *SAFEWARE: System Safety and Computers,* Addison-Wesley, 1995.

[B20] prENV 50126-1998 (CENELEC), Railway applications—The specification and demonstration of reliability, availability, maintainability and safety (RAMS).

[B21] prENV 50128-1998 (CENELEC) Railway applications—Software for railway control and protection systems.

[B22] prENV 50129-1998 (CENELEC) Railway applications—Safety related electronic systems for signalling.

[B23] Storey, N., *Safety-Critical Computer Systems,* New York: Addison-Wesley, 1996.

--`,,`-`-`,,`,,`,`,,`---
Copyright © 2000 IEEE. All rights reserved.

Copyright The Institute of Electrical and Electronics Engineers, Inc.
Provided by IHS under license with IEEE
No reproduction or networking permitted without license from IHS Not for Resale