IEEE Std 7-4.3.2™-2003
(Revision of IEEE Std 7-4.3.2-1993)

IEEE Standards

7-4.3.2™

IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations

IEEE Power Engineering Society

Sponsored by the
Nuclear Power Engineering Committee

IEEE

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

19 December 2003

Print: SH95168
PDF: SS95168

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

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

and Electronics Engineers, Inc.
IEEE
nitted without license from IHS
Not for Resale

IEEE Std 7-4.3.2™-2003
(Revision of
IEEE Std 7-4.3.2-1993)

IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations

Sponsor

Nuclear Power Engineering Committee
of the
IEEE Power Engineering Society

Approved 11 September 2003

IEEE-SA Standards Board

Abstract: Additional computer specific requirements to supplement the criteria and requirements of IEEE Std 603™-1998 are specified. Within the context of this standard, the term computer is a system that includes computer hardware, software, firmware, and interfaces. The criteria contained herein, in conjunction with criteria in IEEE Std 603-1998, establish minimum functional and design requirements for computers used as components of a safety system.
Keywords: commercial grade item, diversity, safety systems, software, software tools, software verification and validation

---

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

Copyright © 2003 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 19 December 2003. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Incorporated.

Print: ISBN 0-7381-3810-X SH95168
PDF: ISBN 0-7381-3811-8 SS95168

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. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "**AS IS**."

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.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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 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 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. 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; +1 978 750 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 7-4.3.2-2003, IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations.)

This standard evolved from IEEE Std 7-4.3.2-1993. It represents a continued effort by an IEEE working group to support the specification, design, and implementation of computers in safety systems of nuclear power generating stations.

This standard specifies additional computer-specific requirements (incorporating hardware, software, firmware, and interfaces) to supplement the criteria and requirements of IEEE Std 603-1998. This standard should be used in conjunction with IEEE Std 603-1998 to assure the completeness of the safety system design when a computer is to be used as a component of a safety system.

This standard recognizes that development processes for computer systems continue to evolve. As such, the information presented should not be viewed as the only possible solution. This is in keeping with the desire to use advances in digital technology, provided the criteria and requirements of IEEE Std 603-1998 and this standard are met. For example, while this standard does not address specifically artificial intelligence systems or fourth generation languages, their use is not precluded.

IEEE Std 7-4.3.2-1993 referenced ASME NQA-2a-1990, Part 2.7 (referred to as Part 2.7) to address specific software development requirements. References to ASME NQA-2a-1990 have been removed from this standard, and applicable IEEE standards have been referenced.

This standard does not provide requirements associated with the operation and maintenance of the computer following installation (i.e., surveillance testing frequency). Any problems identified should be addressed through applicable standards that specifically address these requirements.

Clause 5.1 in IEEE Std 603-1998 defines the single-failure criterion. Guidance for the application of this criterion is provided in IEEE Std 379™-1988, Standard Application of the Single Failure Criterion to Nuclear Power Generating Station Class 1E Systems. The approach stated in 5.5 of IEEE Std 379-1988 is also appropriate for potential common-cause failures associated with computer hardware and software that have been developed under the requirements of IEEE Std 603-1998 and this standard. Annex B provides additional guidance for determining the need for design diversity in safety-related computer systems.

The working group revised the guidance in the 1993 standard to further address hazard analysis. These efforts resulted in a complete revision of the existing abnormal conditions and events (ACES) discussion in Annex D. Future work should consider the subject of software safety analysis in addressing system hazards. Additionally, future efforts should consider addressing these topics in the body of the standard.

The Nuclear Regulatory Commission endorsed the concept of requirements grading or classification in SECY-91-192, Digital Computer Systems for Advanced Light Water Reactors. A similar concept of safety classification is presented in ANSI/ANS 51.1983 [B1] and ANSI/ANS 52.1-1983 [B2]. If guidance is provided in a revision to IEEE Std 603-1998, efforts should then be undertaken to apply this concept in a revision to this standard.

This standard does not address justification for the selection of software tools and acceptance criteria for compilers, operating systems, and libraries. The working group felt this subject was outside of the scope of this revision.

During the NPEC preview of this revision of the standard, the topic of safety system software security was discussed. Specifically, the ability of the software system to fulfill its safety related functions in the presence of attacks. Recommendations were made that a future revision of the standard address software risks associated with attacks by insiders and from outside.

Copyright © 2003 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

The qualification of existing commercial computers, or COTS, is addressed in the body of this standard and in Annex C. It is recognized that COTS dedication is a topic that can be applied to a broader range of Safety Systems equipment than computers, but it is included in this standard because it is not addressed elsewhere at this time. It is recommended (for future IEEE standards work) that a specific standard be developed for COTS dedication. IEEE Std 7-4.3.2 should then be revised to reference the new standard for the COTS dedication process.

In summary, the following major changes were implemented in this version of IEEE Std 7-4.3.2:

— The references were updated to include current IEEE Standards

— The definitions were updated and expanded, and references were provided for definitions obtained from other standards.

— A *Software quality metrics* clause was added. Industry practice is moving towards the use of software quality metrics to assure/monitor/improve software quality in addition to the V&V that has traditionally been applied.

— The *Qualification of existing commercial computers* clause was expanded to provide additional guidance that addresses the move toward the use of more commercial hardware and software in safety systems. This clause was reviewed to ensure consistency with industry guidance (e.g., EPRI guidance).

  Discussion during the review led to the action item to move the recommendations and guidance portions of this addition (i.e., "should / may" clauses) to the annex.

— The *Software tools* clause was revised to address expanded use of software tools and methods to confirm suitability (IEC 60880-2, issued last year, specifically addresses the use of software tools.)

— The *Verification and Validation* clause was expanded to support the removal of Annex E, which addressed verification and validation activities. This standard references IEEE Std 1012™, and clarifies requirements that are applicable to safety system software. Whereas IEEE Std 1012 only mentions "independent" V&V in the annex, the authors moved the requirements for IV&V into the body of the standard. Additionally, although different "integrity levels" are defined in IEEE Std 1012, this standard identifies which "integrity level" is applicable to safety system software.

— The *Software configuration management* clause was expanded to provide additional guidance by identifying the key requirements for configuration management for safety system software using the guidance provided in IEEE Std 828™ and IEEE Std 1042™.

— A *Software project risk management* clause was added to provide additional guidance consistent with IEEE Std 1540™ on risk management and IEEE/EIA12207.0 on software life cycle processes.

— A *Fault detection and self-diagnostics* clause was added to address features that are unique to software and computer systems.

— The *Identification* clause has been expanded to include software specific requirements by extending the IEEE Std 603 identification requirements to software.

— Annex A, *Relationship to IEEE Std 603,* was updated to reflect the contents of the current standard.

— Annex B, *Diversity requirements determination,* received minor editorial updates.

— Annex C, *Electromagnetic compatibility,* was deleted because Annex B of IEEE Std 603-1998 addresses the same subject.

— Annex D, *Dedication of existing commercial computers,* was updated to more completely address COTS issues. Additionally, this annex was designated Annex C.

— Annex E, *Verification and validation,* was deleted. V&V requirements were incorporated into the body of the standard. IEEE Std 1012 is referenced in the body to provide guidance.

Copyright © 2003 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 F, *Identification and resolution of hazards,* replaced the previous version of this annex. The content was revised to better represent current practices and processes for hazards analysis. Guidance from EPRI [B3] was included in this section. Additionally, this annex was designated as Annex D.[a]

— Annex G, *Communication independence,* was designated Annex E.

— Annex H, *Computer reliability,* received minor editorial updates and was designated Annex F.

— Annex G, *Bibliography,* was added to identify informative references.

Participants

This document was prepared by the Application of Programmable Digital Computers to Safety Systems Subcommittee Working Group 6.4 of the IEEE Nuclear Power Engineering Committee. At the time this standard was completed, the Subcommittee Working Group 6.4 had the following membership:

Charles Roslund, *Chair*

Michael Waterman, *Secretary*

Larry Erin	Ronald Jarrett	James Stewart (deceased)
Ronald Greenthaler	Ifti Rana	James Sweeney
Myron Hecht	John Scott	John Waclo

At the time this standard was completed, Subcommittee 6 under the Nuclear Power Engineering Committee had the following membership:

Paul Yanosy, *Chair*

David Horvath, *Secretary*

Wesley Bowers	Randy Jamison	Mike Miller
Robert Copyak	Jim Keiper	Charles Roslund
John Disosway	Tom Klein	John Waclo
Britton Grim	Glenn Lang	David Zaprazny
	Evangelos Marinos	

The following members of the balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Stan J. Arnot	Wilmer Gangloff	William Mindick
Vincent Bacanskas	Britton Grim	Radhakrishna Rebbapragada
Farouk Baxter	Randall Groves	Charles Roslund
James Bongarra, Jr.	David Horvath	James Ruggieri
Wesley Bowers	Paul Johnson	Barry Skoras
Daniel Brosnan	James T. Keiper	Neil P. Smith
John P. Carter	Scott Malcolm	James Stoner
Guru Dutt dhingra	John R. Matras	James Thomas
Surin Dureja	Richard Meininger	T.J. Voss
Robert Fuld	Gary Michel	John Waclo

[a]The numbers in brackets correspond to those of the bibliography in Annex G.

Copyright © 2003 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

When the IEEE-SA Standards Board approved this standard on 11 September 2003, it had the following membership:

Don Wright, *Chair*
Howard M. Frazier, *Vice Chair*
Judith Gorman, *Secretary*

H. Stephen Berger
Joe Bruder
Bob Davis
Richard DeBlasio
Julian Forster*
Toshio Fukuda
Arnold M. Greenspan
Raymond Hapeman
Donald M. Heirman
Laura Hitchcock
Richard H. Hulett
Anant Jain
Lowell G. Johnson
Joseph L. Koepfinger*
Tom McGean
Steve Mills
Daleep C. Mohla
William J. Moylan
Paul Nikolich
Gary Robinson
Malcolm V. Thaden
Geoffrey O. Thompson
Doug Topping
Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Alan Cookson, *NIST Representative*
Satish K. Aggarwal, *NRC Representative*

Michelle D. Turner
IEEE Standards Project Editor

Copyright © 2003 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. Scope 1

2. References 1

3. Definitions and abbreviations 1

3.1 Definitions 1
3.2 Abbreviations and acronyms 5

4. Safety system design basis 6

5. Safety system criteria 6

5.1 Single-failure criterion 6
5.2 Completion of protective action 6
5.3 Quality 6
5.4 Equipment qualification 10
5.5 System integrity 12
5.6 Independence 13
5.7 Capability for test and calibration 14
5.8 Information displays 14
5.9 Control of access 14
5.10 Repair 14
5.11 Identification 14
5.12 Auxiliary features 14
5.13 Multi-unit stations 14
5.14 Human factor considerations 14
5.15 Reliability 15

6. Sense and command features—functional and design requirements 15

7. Execute features—functional and design requirements 15

8. Power source requirements 15

Annex A (informative) Mapping of IEEE Std 603-1998 to IEEE Std 7-4.3.2-2003 16

Annex B (informative) Diversity requirements determination 18

Annex C (informative) Dedication of existing commercial computers 20

Annex D (informative) Identification and resolution of hazards 27

Annex E (informative) Communication independence 39

Annex F (informative) Computer reliability 43

Annex G (informative) Bibliography 48

Copyright © 2003 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

and Electronics Engineers, Inc.
IEEE
nitted without license from IHS
Not for Resale

IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations

1. Scope

This standard serves to amplify criteria in IEEE Std 603™-1998 to address the use of computers as part of safety systems in nuclear power generating stations. The criteria contained herein, in conjunction with criteria in IEEE Std 603-1998, establish minimum functional and design requirements for computers used as components of a safety system.

2. References

This guide shall be used in conjunction with the following standards:

IEEE Std 603-1998, IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations.[1,2]

IEEE Std 1012™-1998, IEEE Standard for Software Verification and Validation.

IEEE Std 1042™-1987 (R1993), IEEE Guide to Software Configuration Management.[3]

IEEE/EIA Std 12207.0-1996 IEEE/EIA Standard—Industry Implementation of International Standard ISO/IEC 12207: 1995 (ISO/IEC 12207), Standard for Information Technology—Software life cycle processes.

3. Definitions and abbreviations

3.1 Definitions

For the purposes of this standard, the following terms and definitions apply. The *Authoritative Dictionary of IEEE Standards Terms* [B5][4] should be referenced for terms not defined in this clause.

---

[1]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA (http://standards.ieee.org/).

[2]The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

[3]IEEE Std 1042-1987 (Reaff 1993) has been withdrawn; however, copies can be obtained from Global Engineering, 15 Inverness Way East, Englewood, CO 80112-5704, USA, tel. (303) 792-2181 (http://global.ihs.com/).

[4]The numbers in brackets correspond to those of the bibliography in Annex G.

Copyright © 2003 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

3.1.1 acceptance testing: (1) Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. *See also:* **qualification testing, system testing. (2)** Formal testing conducted to enable a user, customer, or other authorized entity to determine whether to accept a system or component.

3.1.2 application software: Software designed to fulfill specific needs of a user; for example, software for navigation, payroll, or process control.

3.1.3 architecture: The organizational structure of a system or component.

3.1.4 commercial grade item: An item that is a) not subject to design or specification requirements unique to nuclear facilities; and b) used in applications other than nuclear facilities; and c) ordered from the manufacturer/supplier on the basis of specifications set forth in the manufacturer's published product description (for example, a catalog).

3.1.5 commercial grade item dedication: A process of evaluating and accepting commercial grade items to obtain adequate confidence of suitability for safety application.

3.1.6 complexity: (1) The degree to which a system or system component has a design or implementation that is difficult to understand and verify. **(2)** Pertaining to any set of structure-based metrics that measure the attribute in definition (1).

3.1.7 component: One of the parts that make up a system. A component may be hardware or software and may be subdivided into other components.

NOTE—The terms "module," "component," and "unit" are often used interchangeably or defined to be supplements of one another in different ways dependent upon the context. The relationship of these terms is not yet standardized.

3.1.8 computer: A functional programmable unit that consists of one or more associated processing units and peripheral equipment, that is controlled by internally stored programs, and that can perform substantial computation, including numerous arithmetic or logic operations, without human intervention.

3.1.9 computer instruction: (1) A statement in a programming language, specifying an operation to be performed by a computer and the addresses or values of the associated operands; for example, Move A to B. **(2)** Loosely, any executable statement in a computer program.

3.1.10 computer program: A combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions.

3.1.11 computer system: A system containing one or more computers and associated software.

3.1.12 configuration: (1) The arrangement of a computer system or component as defined by the number, nature, and interconnections of its constituent parts. **(2)** In configuration management, the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product.

3.1.13 configuration control: An element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification.

3.1.14 configuration item: An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process.

Copyright © 2003 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

3.1.15 configuration management: A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements.

3.1.16 correctness: (1) The degree to which a system or component is free from faults in its specification, design, and implementation. **(2)** The degree to which software, documentation, or other items meet the specified requirements. **(3)** The degree to which software, documentation, or other items meet user needs and expectations, whether specified or not.

3.1.17 data: (1) A representation of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automatic means. **(2)** Sometimes used as a synonym for documentation.

3.1.18 data structure: A physical or logical relationship among data elements, designed to support specific data manipulation functions.

3.1.19 design: (1) The process of defining the architecture, components, interfaces, and other characteristics of a system or component. **(2)** The result of the process in definition (1).

3.1.20 document: (1) A medium and the information recorded on it, that generally has permanence and can be read by a person or a machine. Examples in software engineering include project plans, specifications, test plans, user manuals. **(2)** To create a document as in definition (1). **(3)** To add comments to a computer program.

3.1.21 documentation: (1) A collection of documents on a given subject. **(2)** Any written or pictorial information describing, defining, specifying, reporting or certifying activities, requirements, procedures, or results. **(3)** The process of generating or revising a document. **(4)** The management of documents, including identification, acquisition, processing, storage, and dissemination.

3.1.22 error: (1) The difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. For example, a difference of 30 meters between a computed result and the correct result. **(2)** An incorrect step, process, or data definition. For example, an incorrect instruction in a computer program. **(3)** An incorrect result. For example, a computed result of 12 when the correct result is 10. **(4)** A human action that produces an incorrect result. For example, an incorrect action on the part of a programmer or operator.

NOTE—While all four definitions are commonly used, one distinction assigns definition (1) to the word "error," definition (2) to the word "fault," definition (3) to the word "failure," and definition (4) to the word "mistake."

3.1.23 execution: The process of carrying out an instruction or the instructions of a computer program by a computer.

3.1.24 failure: The inability of a system or component to perform its required functions within specified performance requirements.

NOTE—The fault tolerance discipline distinguishes between a human action (a mistake), its manifestation (a hardware or software fault), the result of the fault (a failure), and the amount by which the result is incorrect (the error).

3.1.25 fault: (1) A defect in a hardware device or component; for example, a short circuit or broken wire. **(2)** An incorrect step, process, or data definition in a computer program.

NOTE—This definition is used primarily by the fault tolerance discipline. In common usage, the terms "error" and "bug" are used to express this meaning.

--`,,`-`-`,,`,,`,`,,`---
Copyright © 2003 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

3.1.26 firmware: The combination of a hardware device and computer instructions and data that reside as read-only software on that device.

3.1.27 function: A defined objective or characteristic action of a system or component. For example, a system may have inventory control as its primary function.

3.1.28 functional unit: An entity of hardware, software, or both capable of accomplishing a specified purpose.

3.1.29 hardware: Physical equipment used to process, store, or transmit computer programs or data.

3.1.30 hazard: A condition that is a prerequisite to an accident. Hazards include external events as well as conditions internal to computer hardware or software.

3.1.31 hazard analysis: A process that explores and identifies conditions that are not identified by the normal design review and testing process. The scope of hazard analysis extends beyond plant design basis events by including abnormal events and plant operations with degraded equipment and plant systems. Hazard analysis focuses on system failure mechanisms rather than verifying correct system operation.

3.1.32 implementation: **(1)** The process of translating a design into hardware components, software components, or both. **(2)** The result of the process in definition (1).

3.1.33 interface: **(1)** A shared boundary across which information is passed. **(2)** A hardware or software component that connects two or more other components for the purpose of passing information from one to the other. **(3)** To connect two or more components for the purpose of passing information from one to the other. **(4)** To serve as a connecting or connected component as in definition (2).

3.1.34 module: **(1)** A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to, or output from an assembler, compiler, linkage editor, or executive routine. **(2)** A logically separable part of a program.

3.1.35 procedure: **(1)** The course of action taken for the solution of a problem. A course of action taken to perform a given task. **(2)** A written description of a course of action as in definition (1) for example, a documented test procedure. **(3)** A portion of a computer program that is named and that performs a specific action.

3.1.36 qualification testing: Testing performed to demonstrate to the acquirer that the software item or system meets its specified requirements.

3.1.37 requirement: **(1)** A condition or capability needed by a user to solve a problem or achieve an objective. **(2)** A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents. **(3)** A documented representation of a condition or capability as in definition (1) or definition (2).

3.1.38 requirements specification: A document that specifies the requirements for a system or component. Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.

3.1.39 safety system: A system that is relied upon to remain functional during and following design basis events to ensure: the integrity of the reactor coolant pressure boundary; the capability to shut down the reactor and maintain it in a safe shutdown condition; or the capability to prevent or mitigate the consequences of accidents that could result in potential off-site releases.

Copyright © 2003 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

3.1.40 software: Computer programs, procedures, and associated documentation and data pertaining to the operation of a computer system.

3.1.41 software maintenance: (1) Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment. **(2)** The set of activities that takes place to ensure that software installed for operational use continues to perform as intended and fulfill its intended role in system operation. Software maintenance includes improvements, aid to users, and related activities.

3.1.42 software tools: A computer program used in the development, testing, analysis, or maintenance of a program or its documentation. Examples include comparator, cross-reference generator, decompiler, driver, editor, flowcharter, monitor, test case generator, and timing analyzer.

3.1.43 specification: A document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system or system, and, often, the procedures for determining whether these provisions have been satisfied.

3.1.44 system: A collection of components organized to accomplish a specific function or a set of functions.

3.1.45 system software: Software designed to facilitate the operation and maintenance of a computer system and its associated programs; for example, operating systems, assemblers, utilities.

3.1.46 system testing: Testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

3.1.47 testing: (1) The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component. **(2)** The process of analyzing a software item to detect the difference between existing and required conditions (that is, bugs) and to evaluate the features of the software items.

3.1.48 test plan: (1) A document describing the scope, approach, resources, and schedule of intended test activities. It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. **(2)** A document that describes the technical and management approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, responsibilities, schedules, and required resources for the testing activity.

3.1.49 validation: The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.

3.1.50 verification: (1) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. **(2)** Formal proof of program correctness.

3.1.51 verification and validation (V&V): The process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements.

3.2 Abbreviations and acronyms

The following abbreviations and acronyms appear in this standard:

ATWS	anticipated transient without scram
CFR	Code of Federal Regulations
COTS	commercial off-the-shelf

Copyright © 2003 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

CRC	cyclic redundancy check
FTA	fault tree analysis
FMEA	failure modes and effects analysis
IV&V	independent verification and validation
MTBF	mean time between failures
PROM	programmable read only memory
QA	quality assurance
RAM	random access memory
SAT	site acceptance testing
SVVP	software verification and validation plan
USNRC	United States Nuclear Regulatory Commission
V&V	verification and validation

4. Safety system design basis

NOTE—See Annex A for more information about the relationship of this standard to IEEE Std 603-1998.

No requirements beyond IEEE Std 603-1998 are necessary (see also Annex B).

5. Safety system criteria

The following subclauses list the safety system criteria in the order they are listed in IEEE Std 603-1998. For some criteria, there are no additional requirements beyond what is stated in IEEE Std 603-1998. For other criteria, additional requirements are described in 5.1 through 5.15.

5.1 Single-failure criterion

No requirements beyond IEEE Std 603-1998 are necessary (see also Annex B).

5.2 Completion of protective action

No requirements beyond IEEE Std 603-1998 are necessary.

5.3 Quality

Hardware quality is addressed in IEEE Std 603-1998. Software quality is addressed in IEEE/EIA Std 12207.0-1996 and supporting standards. Computer development activities shall include the development of computer hardware and software. The integration of the computer hardware and software and the integration of the computer with the safety system shall be addressed in the development process.

A typical computer system development process consists of the following life cycle processes:

— Creating the conceptual design of the system, translation of the concepts into specific system requirements
— Using the requirements to develop a detailed system design
— Implementing the design into hardware and software functions

Copyright © 2003 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

— Testing the functions to assure the requirements have been correctly implemented
— Installing the system and performing site acceptance testing
— Operating and maintaining the system
— Retiring the system

In addition to the requirements of IEEE Std 603-1998, the following activities necessitate additional requirements that are necessary to meet the quality criterion:

— Software development
— Qualification of existing commercial computers (see 5.4.2)
— Use of software tools
— Verification and validation
— Configuration management
— Risk Management

5.3.1 Software development

Computer software shall be developed, modified, or accepted in accordance with an approved software quality assurance (QA) plan consistent with the requirements of IEEE/EIA 12207.0-1996. The software QA plan shall address all software that is resident on the computer at run time (i.e., application software, network software, interfaces, operating systems, and diagnostics). Guidance for developing software QA plans can be found in IEC 60880 (1986-09) [B4] and IEEE Std 730™-1998 [B8].

5.3.1.1 Software quality metrics

The use of software quality metrics shall be considered throughout the software life cycle to assess whether software quality requirements are being met. When software quality metrics are used, the following life cycle phase characteristics should be considered:

— Correctness/Completeness (Requirements phase)
— Compliance with requirements (Design phase)
— Compliance with design (Implementation phase)
— Functional compliance with requirements (Test and Integration phase)
— On-site functional compliance with requirements (Installation and Checkout phase)
— Performance history (Operation and Maintenance phase)

The basis for the metrics selected to evaluate software quality characteristics should be included in the software development documentation. IEEE Std 1061™-1998 [B11] provides a methodology for the application of software quality metrics.

5.3.2 Software tools

Software tools used to support software development processes and verification and validation (V&V) processes shall be controlled under configuration management.

One or both of the following methods shall be used to confirm the software tools are suitable for use:

a) A test tool validation program shall be developed to provide confidence that the necessary features of the software tool function as required.

Copyright © 2003 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

b) The software tool shall be used in a manner such that defects not detected by the software tool will be detected by V&V activities.

Tool operating experience may be used to provide additional confidence in the suitability of a tool, particularly when evaluating the potential for undetected defects.

5.3.3 Verification and validation

NOTE—See IEEE Std 1012-1998 and IEEE Std 1012a™-1998 [B10] for more information about software V&V.

V&V is an extension of the program management and systems engineering team activities. V&V is used to identify objective data and conclusions (i.e., proactive feedback) about digital system quality, performance, and development process compliance throughout the system life cycle. Feedback consists of anomaly reports, performance improvements, and quality improvements regarding the expected operating conditions across the full spectrum of the system and its interfaces.

V&V processes are used to determine whether the development products of an activity conform to the requirements of that activity, and whether the system performs according to its intended use and user needs. This determination of suitability includes assessment, analysis, evaluation, review, inspection, and testing of products and processes.

This standard adopts the IEEE Std 1012-1998 terminology of process, activity and task, in which software V&V processes are subdivided into activities, which are further subdivided into tasks. The term V&V effort is used to reference this framework of V&V processes, activities, and tasks.

V&V processes shall address the computer hardware and software, integration of the digital system components, and the interaction of the resulting computer system with the nuclear power plant.

The V&V activities and tasks shall include system testing of the final integrated hardware, software, firmware, and interfaces.

The software V&V effort shall be performed in accordance with IEEE Std 1012-1998. The IEEE Std 1012-1998 V&V requirements for the highest integrity level (level 4) apply to systems developed using this standard (i.e., IEEE Std 7-4.3.2™). See IEEE Std 1012-1998 Annex B for a definition of integrity level 4 software.

5.3.4 Independent V&V (IV&V) requirements

The previous section addresses the V&V activities to be performed. This section defines the levels of independence required for the V&V effort. IV&V activities are defined by three parameters: technical independence, managerial independence, and financial independence. These parameters are described in Annex C of IEEE Std 1012-1998.

The development activities and tests shall be verified and validated by individuals or groups with appropriate technical competence, other than those who developed the original design.

Oversight of the IV&V effort shall be vested in an organization separate from the development and program management organizations. The V&V effort shall independently select

a) The segments of the software and system to be analyzed and tested,

b) The V&V techniques, and

c) The technical issues and problems upon which to act.

Copyright © 2003 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

The V&V effort shall be allocated resources that are independent of the development resources.

See Annex C of IEEE Std 1012-1998 for additional guidance.

5.3.5 Software configuration management

Software configuration management shall be performed in accordance with IEEE Std 1042-1987. IEEE Std 828™-1998 [B9] provides guidance for the development of software configuration management plans.

The minimum set of activities shall address the following:

a) Identification and control of all software designs and code

b) Identification and control of all software design functional data (e.g., data templates and data bases)

c) Identification and control of all software design interfaces

d) Control of all software design changes

e) Control of software documentation (user, operating, and maintenance documentation)

f) Control of software vendor development activities for the supplied safety system software

g) Control and retrieval of qualification information associated with software designs and code

h) Software configuration audits

i) Status accounting

Some of these functions or documents may be performed or controlled by other QA activities. In this case, the software configuration management plan shall describe the division of responsibility.

A software baseline shall be established at appropriate points in the software life cycle process to synchronize engineering and documentation activities. Approved changes that are created subsequent to a baseline shall be added to the baseline.

The labeling of the software for configuration control shall include unique identification of each configuration item, and revision and/or date time stamps for each configuration item.

Changes to the software/firmware shall be formally documented and approved consistent with the software configuration management plan. The documentation shall include the reason for the change, identification of the affected software/firmware, and the impact of the change on the system. Additionally, the documentation should include the plan for implementing the change in the system (e.g., immediately implementing the change, or scheduling the change for a future version).

5.3.6 Software project risk management

Software project risk management is a tool for problem prevention: identifying potential problems, assessing their impact, and determining which potential problems must be addressed to assure that software quality goals are achieved. Risk management shall be performed at all levels of the digital system project to provide adequate coverage for each potential problem area. Software project risks may include technical, schedule, or resource-related risks that could compromise software quality goals, and thereby affect the ability of the safety computer system to perform safety related functions. Software project risk management differs from hazard analysis, as defined in 3.1.31, in that hazard analysis is focused solely on the technical aspects of system failure mechanisms.

Copyright © 2003 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

Risk management shall include the following steps:

a) Determine the scope of risk management to be performed for the digital system.

b) Define and implement appropriate risk management strategies.

c) Identify risks to the software project in the project risk management strategy and as they develop during the conduct of the project.

d) Analyze risks to determine the priority for their mitigation.

e) Develop risk mitigation plans for risks that have the potential to significantly impact software quality goals, with appropriate metrics for tracking resolution progress. (These risks may include technical, schedule, or resource-related project risks that could compromise the ability of the safety computer system to perform safety related functions.)

f) Take corrective actions when expected quality is not achieved.

g) Establish a project environment that supports effective communications between individuals and groups for the resolution of software project risks.

Additional guidance on the topic of risk management is provided in IEEE/EIA 12207.0-1996, and IEEE Std 1540™-2001 [B13].

5.4 Equipment qualification

In addition to the equipment qualification criteria provided by IEEE Std 603-1998, the requirements listed in 5.4.1 and 5.4.2 are necessary to qualify digital computers for use in safety systems.

5.4.1 Computer system testing

Computer system qualification testing (see 3.1.36) shall be performed with the computer functioning with software and diagnostics that are representative of those used in actual operation. All portions of the computer necessary to accomplish safety functions, or those portions whose operation or failure could impair safety functions, shall be exercised during testing. This includes, as appropriate, exercising and monitoring the memory, the CPU, inputs and outputs, display functions, diagnostics, associated components, communication paths, and interfaces. Testing shall demonstrate that the performance requirements related to safety functions have been met.

5.4.2 Qualification of existing commercial computers

NOTE—See Annex C for more information about commercial grade item dedication.

The qualification process shall be accomplished by evaluating the hardware and software design using the criteria of this standard. Acceptance shall be based upon evidence that the digital system or component, including hardware, software, firmware, and interfaces, can perform its required functions. The acceptance and its basis shall be documented and maintained with the qualification documentation.

In those cases in which traditional qualification processes cannot be applied, an alternative approach to verify a component is acceptable for use in a safety-related application is commercial grade dedication. The objective of commercial grade dedication is to verify that the item being dedicated is equivalent in quality to equipment developed under a 10 CFR 50 Appendix B program [B16].

The dedication process for the computer shall entail identification of the physical, performance, and development process requirements necessary to provide adequate confidence that the proposed digital system or component can achieve the safety function. The dedication process shall apply to the computer hardware, software, and firmware that are required to accomplish the safety function. The dedication process for software and firmware shall, whenever possible, include an evaluation of the design process. There may be some

 Copyright © 2003 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

instances in which a design process cannot be evaluated as part of the dedication process. For example, the organization performing the evaluation may not have access to the design process information for a microprocessor chip to be used in the safety system. In this case, it would not be possible to perform an evaluation to support the dedication. Because the dedication process involves all aspects of life cycle processes and manufacturing quality, commercial grade item dedication should be limited to items that are relatively simple in function relative to their intended use.

Commercial grade item dedication involves preliminary phase and detailed phase activities. These phase activities are described in 5.4.2.1 through 5.4.2.2.

5.4.2.1 Preliminary phase of the COTS dedication process

In the preliminary phase, the risks and hazards are evaluated, the safety functions are identified, configuration management is established, and the safety category of the system is determined.

5.4.2.1.1 Evaluate the system safety function risks and hazards

An analysis shall be performed to identify the functional and performance requirements of the safety system. This analysis shall identify the risks and hazards that could interfere with accomplishing the safety function.

5.4.2.1.2 Identify the safety function(s) the COTS item shall perform

Once the system-level functions have been identified and the risks and hazards have been evaluated, the dedicating organization shall identify the safety functions to be performed by the COTS item. This process shall address all safety functions to be performed by the COTS, and the potential affect of the COTS function(s) on other safety-related functions or interfaces.

5.4.2.1.3 Establish configuration management controls

COTS items to be used in safety systems shall be controlled in a configuration management process that provides traceability of the COTS item development life cycle processes.

5.4.2.2 Detailed phase of the COTS dedication process

Following this preliminary phase of commercial dedication, the commercial grade item is evaluated for acceptability using detailed acceptance criteria. The critical characteristics by which a COTS item will be evaluated for use in a safety system shall be identified by a technical evaluation. Each critical characteristic shall be verifiable (e.g., by inspection, analysis, demonstration, or testing). This standard uses the following three categories of commercial grade item critical characteristics:

— Physical characteristics include attributes such as physical dimensions, power requirements, part numbers, hardware and software model and version numbers, and data communication physical requirements.
— Performance characteristics include attributes such as response time, human-machine functional requirements, memory allocation, safety function performance during abnormal conditions, reliability, error handling, required imbedded functions, and environmental qualification requirements (e.g., seismic, temperature, humidity, and electromagnetic compatibility).
— Development process characteristics include attributes such as supporting life cycle processes (e.g., verification and validation activities, configuration management processes, and hazard analyses), traceability, and maintainability.

As part of defining these critical characteristics, analyses shall identify potential hazards that could interfere with the safety functions (see Annex D).

Copyright © 2003 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 describes the processes that should be used individually or in combination to evaluate the physical, performance, and development process critical characteristics.

5.4.2.3 Maintenance of commercial dedication

If computer hardware, software, or firmware has been procured as a commercial grade item and accepted through a commercial dedication process, then changes to the commercially dedicated computer hardware, software, or firmware shall be traceable through formal documentation.

Changes to the commercially dedicated computer hardware, software, or firmware shall be evaluated in accordance with the process that formed the basis for the original acceptance. Included in this evaluation shall be consideration of the potential impact that computer hardware revisions may have on software or firmware. If any elements of the approved process have been omitted during the computer hardware, software, or firmware revision process, further evaluation shall be required.

Commercial grade dedication of computer hardware, software, or firmware is performed for a specific safety system application. Use of a commercially dedicated item in safety system applications beyond that included in the baseline dedication shall require additional evaluation for the new application.

Documentation supporting the commercial grade item dedication shall be maintained as a configuration item.

5.5 System integrity

In addition to the system integrity criteria provided by IEEE Std 603-1998, the following are necessary to achieve system integrity in digital equipment for use in safety systems:

— Design for computer integrity
— Design for test and calibration
— Fault detection and self-diagnostics

5.5.1 Design for computer integrity

The computer shall be designed to perform its safety function when subjected to conditions, external or internal, that have significant potential for defeating the safety function. For example, input and output processing failures, precision or roundoff problems, improper recovery actions, electrical input voltage and frequency fluctuations, and maximum credible number of coincident signal changes.

If the system requirements identify a safety system preferred failure mode, failures of the computer shall not preclude the safety system from being placed in that mode. Performance of computer system restart operations shall not result in the safety system being inhibited from performing its function.

5.5.2 Design for test and calibration

Test and calibration functions shall not adversely affect the ability of the computer to perform its safety function. Appropriate bypass of one redundant channel is not considered an adverse effect in this context. It shall be verified that the test and calibration functions do not affect computer functions that are not included in a calibration change (e.g., setpoint change).

V&V, configuration management, and QA shall be required for test and calibration functions on separate computers (e.g., test and calibration computer) that provide the sole verification of test and calibration data. V&V, configuration management, and QA shall be required when the test and calibration function is inherent to the computer that is part of the safety system.

 Copyright © 2003 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

V&V, configuration management, and QA are not required when the test and calibration function is resident on a separate computer and does not provide the sole verification of test and calibration data for the computer that is part of the safety system.

5.5.3 Fault detection and self-diagnostics

Computer systems can experience partial failures that can degrade the capabilities of the computer system, but may not be immediately detectable by the system. Self-diagnostics are one means that can be used to assist in detecting these failures. Fault detection and self-diagnostics requirements are addressed in this subclause.

The reliability requirements of the safety system shall be used to establish the need for self-diagnostics. Self diagnostics are not required for systems in which failures can be detected by alternate means in a timely manner. If self-diagnostics are incorporated into the system requirements, these functions shall be subject to the same V&V processes as the safety system functions.

If reliability requirements warrant self-diagnostics, then computer programs shall incorporate functions to detect and report computer system faults and failures in a timely manner. Conversely, self-diagnostic functions shall not adversely affect the ability of the computer system to perform its safety function, or cause spurious actuations of the safety function. A typical set of self-diagnostic functions includes the following:

— Memory functionality and integrity tests (e.g., PROM checksum and RAM tests)
— Computer system instruction set (e.g., calculation tests)
— Computer peripheral hardware tests (e.g., watchdog timers and keyboards)
— Computer architecture support hardware (e.g., address lines and shared memory interfaces)
— Communication link diagnostics (e.g., CRC checks)

Infrequent communication link failures that do not result in a system failure or a lack of system functionality do not require reporting.

When self-diagnostics are applied, the following self-diagnostic features shall be incorporated into the system design:

a) Self-diagnostics during computer system startup
b) Periodic self-diagnostics while the computer system is operating
c) Self-diagnostic test failure reporting

5.6 Independence

In addition to the requirements of IEEE Std 603-1998, data communication between safety channels or between safety and nonsafety systems shall not inhibit the performance of the safety function.

IEEE Std 603-1998 requires that safety functions be separated from nonsafety functions such that the nonsafety functions cannot prevent the safety system from performing its intended functions. In digital systems, safety and nonsafety software may reside on the same computer and use the same computer resources.

Either of the following approaches is acceptable to address the previous issues:

a) Barrier requirements shall be identified to provide adequate confidence that the nonsafety functions cannot interfere with performance of the safety functions of the software or firmware. The barriers

--`,,`-`-`,,`,,`,`,,`---
Copyright © 2003 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

shall be designed in accordance with the requirements of this standard. The nonsafety software is not required to meet these requirements.

b) If barriers between the safety software and nonsafety software are not implemented, the nonsafety software functions shall be developed in accordance with the requirements of this standard.

Guidance for establishing communication independence is provided in Annex E.

5.7 Capability for test and calibration

No requirements beyond IEEE Std 603-1998 are necessary.

5.8 Information displays

No requirements beyond IEEE Std 603-1998 are necessary.

5.9 Control of access

No requirements beyond IEEE Std 603-1998 are necessary.

5.10 Repair

No requirements beyond IEEE Std 603-1998 are necessary.

5.11 Identification

To provide assurance that the required computer system hardware and software are installed in the appropriate system configuration, the following identification requirements specific to software systems shall be met:

a) Firmware and software identification shall be used to assure the correct software is installed in the correct hardware component.

b) Means shall be included in the software such that the identification may be retrieved from the firmware using software maintenance tools.

c) Physical identification requirements of the digital computer system hardware shall be in accordance with the identification requirements in IEEE Std 603-1998.

5.12 Auxiliary features

No requirements beyond IEEE Std 603-1998 are necessary.

5.13 Multi-unit stations

No requirements beyond IEEE Std 603-1998 are necessary.

5.14 Human factor considerations

No requirements beyond IEEE Std 603-1998 are necessary.

 Copyright © 2003 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

5.15 Reliability

NOTE—See Annex F for more information about the reliability criterion.

In addition to the requirements of IEEE Std 603-1998, when reliability goals are identified, the proof of meeting the goals shall include the software. The method for determining reliability may include combinations of analysis, field experience, or testing. Software error recording and trending may be used in combination with analysis, field experience, or testing.

6. Sense and command features—functional and design requirements

No requirements beyond IEEE Std 603-1998 are necessary.

7. Execute features—functional and design requirements

No requirements beyond IEEE Std 603-1998 are necessary.

8. Power source requirements

No requirements beyond IEEE Std 603-1998 are necessary.

Copyright © 2003 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)

Mapping of IEEE Std 603-1998 to IEEE Std 7-4.3.2-2003

NOTE—Clause 4 through Clause 8 of this standard coincide with Clause 4 through Clause 8 in IEEE Std 603-1998.

Table A.1—Mapping of IEEE Std 603-1998 to IEEE Std 7-4.3.2-2003

IEEE Std 603-1998 criteria	IEEE Std 7-4.3.2-2003 additional requirements	Annex for guidance
4. Safety system design basis	Safety system design basis	Annex B
5. Safety system criteria	None	Annex B
5.1 Single-failure criterion	None	—
5.2 Completion of protective action	None	—
5.3 Quality	Software development (see 5.3.1) Software tools (see 5.3.2) Verification and validation (see 5.3.3) Independent V&V (IV&V) requirements (see 5.3.4) Software configuration management (see 5.3.5) Software project risk management (see 5.3.6)	Annex D and Annex F
5.4 Equipment qualification	Testing software and diagnostics (see 5.4.1) Qualification of existing commercial computers (see 5.4.2)	Annex C
5.5 System integrity	Design for computer integrity (see 5.5.1) Design for test and calibration (see 5.5.2) Fault detection and self-diagnostics (see 5.5.3)	Annex B and Annex C
5.6 Independence	Independence (see 5.6)	Annex E
5.7 Capability for test and calibration	None	—
5.8 Information displays	None	—
5.9 Control of access	None	—
5.10 Repair	None	—
5.11 Identification	Identification (see 5.11)	—
5.12 Auxiliary features	None	—

Copyright © 2003 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

Table A.1—Mapping of IEEE Std 603-1998 to IEEE Std 7-4.3.2-2003 *(continued)*

IEEE Std 603-1998 criteria	IEEE Std 7-4.3.2-2003 additional requirements	Annex for guidance
5.13 Multi-unit stations	None	—
5.14 Human factor considerations	None	—
5.15 Reliability	Reliability (see 5.15)	Annex F
6. Sense and command feature—Functional design requirements	None	—
7. Execute feature—Functional design requirements	None	—
8. Power source requirements	None	—

Copyright © 2003 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)

Diversity requirements determination

B.1 Background

With the introduction of computers as a part of a safety system, concerns have arisen over the possibility that the use of computer software could result in a common-mode failure. Diversity is one method of addressing this concern. This annex provides a methodology for determining the need for diversity.

B.2 Discussion

Conditions may exist under which some form of diversity may be necessary to provide additional assurance beyond that provided by the design and QA programs that incorporate software QA and V&V. When a computer design is complex (e.g., all command features for a safety function are implemented on one computer) and limited operating experience exists, the guidance in this annex provides a methodology for determining the adequacy of other plant features that may be relied upon for functional diversity or defense-in-depth.

If adequate diversity exists or could be added to the nuclear power generating station design, then computer diversity is not necessary. Complex systems with limited experience to be applied in designs without diversity should employ computer diversity.

B.2.1 Methodology for determining need for diversity

Using plant design basis events analyses, determine the safety function(s) to be performed by the proposed computer and identify other safety or nonsafety design features that are capable of performing the same safety function or a different safety function that provides equivalent protection against identified “unacceptable results.” For example, an Anticipated Transient Without Scram (ATWS) system may provide functional diversity for the reactor protection system to accomplish the reactor trip function. Manual operator actions are acceptable provided that the necessary controls and indicators are available to support the accomplishment of appropriate actions within an acceptable time frame. If functional diversity exists using components not subject to software errors that are postulated for the proposed computer, then it is acceptable to use identical software in redundant channels of the proposed system.

If functional diversity does not exist, a defense-in-depth analysis should be performed to determine if diversity exists within the echelons of defense (i.e., reactor protection, engineered safety features, and controls and monitoring systems). This analysis identifies design features in each echelon that are capable of preventing or mitigating the condition being analyzed and determines whether postulated failures such as software errors in one echelon have the potential for adversely affecting other echelons. Credit may be taken for manual actions and available nonsafety controls and monitoring features that are capable of performing within required time frames.

If the analysis shows that defense-in-depth against unacceptable results exists and that the echelons of defense are not affected by the postulated software failure, then it is acceptable to use identical software in redundant channels of the proposed system.

 Copyright © 2003 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

If the analysis does not confirm either functional diversity or defense-in-depth, then a diverse design should be used. This may be achieved by a combination of computer and non-computer channels or diverse computers. Diversity of computers may be achieved through the use of separate computer functional specifications, computer hardware, computer languages, etc., to minimize the possibility of common-cause failures.

Copyright © 2003 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)

Dedication of existing commercial computers

C.1 Background

There may exist situations in which safety systems will be designed in whole or in part with computers (hardware, software, firmware, and interfaces) that have been developed outside the criteria of this standard. Subclause 5.4.2 provides general guidance for commercial grade dedication. This annex is intended to assist in addressing these situations. This will allow safety system designs that use computers that were not specifically designed for nuclear power plant applications. This annex also addresses maintenance of the dedication of existing qualified commercial computers. With respect to this standard, the commercial grade dedication process addresses the initial qualification of a COTS item and the maintenance of that qualification.

This annex outlines the key steps that should be followed to provide confidence that an existing commercial computer is of sufficiently high quality and reliability to be used in a safety system (see also 5.4.2). Aspects of the commercial grade dedication process may be appropriate when a manufacturer does not perform component development activities in accordance with a 10 CFR 50 Appendix B program [B16].

C.2 Discussion

Third-party or manufacturer qualification of safety equipment may not result in the same degree of design documentation as would be available from a design and development process consistent with the criteria of this standard. Existing commercial computers may have documented evidence of operating experience; however, credit should be taken only for operating experience that is similar to the manner and configuration in which the computer will be used in the nuclear power generating station. Operating experience is meant to supplement design process documentation and V&V activities.

The manufacturer of an existing commercial computer may have a greater degree of design information than the third party dedicator. Examples are: availability of information regarding the computer software programs, details of the design and review process (i.e., V&V), documentation of operating experience, and the maintenance of the design of the computer hardware and software (i.e., configuration management).

The objective of commercial grade item dedication is to determine, with reasonable assurance, that the item being dedicated satisfies the requirements necessary to accomplish the safety function. This involves the following:

— Identifying the safety functions the computer must perform
— Identifying the characteristics the computer must possess in order to accomplish the safety functions
— Demonstrating that the characteristics are acceptably implemented

C.2.1 Identifying the safety functions the computer must perform

Analyses should be performed to identify functional and performance requirements for the computer to implement the safety function. This analysis should also identify hazards (see Annex D) that could interfere with the computer accomplishing the safety function.

Copyright © 2003 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

C.2.2 Identifying the computer characteristics

The computer functional and performance requirements and hazards identified in C.2.1 should be allocated to hardware and software. For the software, the development process steps required in 5.4.2 should be identified.

C.2.2.1 Hardware

The functional and performance requirements and hazards allocated to the hardware component that are required in 5.4.2 should be identified. Some examples of computer hardware attributes identified in the functional and performance requirements and hazards are as follows:

— Historical data storage
— System response time
— Seismic qualifications
— Environmental qualifications including temperature, humidity, radiation, and electromagnetic interference
— Man-machine interface
— Maintenance tools

An evaluation should be performed to show that the functional and performance requirements and hazards are in compliance with acceptance criteria. This may require performance of special tests (e.g., seismic and electromagnetic susceptibility), the performance of certain V&V activities, evaluation of published vendor specifications, or reliance on documented operating experience that is similar to the manner in which the computer will be used in the nuclear power generating station.

C.2.2.2 Software

The functional and performance requirements and hazards allocated to the software component that are required in 5.4.2 should be identified. The following are examples of computer software attributes that should be identified:

— System response time
— Accomplishment of identified safety functions during abnormal conditions
— Required imbedded functions
— Correct implementation of logic and algorithms
— Operating system functions
— Communications (internal and external)
— Memory protection
— Control of setpoint changes
— Error handling
— Synchronization
— Control of interrupts
— Control of tuning constants

An evaluation should be performed to show that the functional and performance requirements are in compliance with acceptance criteria. This may require performance of special tests, performance of certain V&V activities, or evaluation of published vendor specifications, combined with reliance on documented operating experience that is similar to the manner in which the computer will be used in the nuclear power generating station.

Copyright © 2003 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

C.2.2.3 Development process steps

The software development process steps required in 5.4.2 should be identified. The following are examples of development process steps that should be identified:

— Definition and documentation of system requirements and acceptance criteria
— Definition of software requirements
— Documentation of software design
— Hardware and software V&V
— Integrated testing of hardware and software
— Hardware and software configuration management
— Manuals for programmers and users
— A process for error identification, documentation, notification, and correction

Acceptance of the development process should be based upon evidence that a documented process has been employed for the development, documentation, and V&V of the computer. All V&V activities should be completed. Compensation for a lack of documentation or performance of some development process steps may be obtained by either of the following:

— Documented operating experience that is similar to the manner in which the computer will be used in the nuclear power generating station
— Results of V&V of the developed application to support acceptance of the operating system or imbedded functions

Any required V&V activities not previously performed should be identified.

C.2.3 Demonstrating that the characteristics are acceptably implemented

Demonstrating that the characteristics are acceptably implemented should be documented to show the component and system comply with functional and performance requirements and the requirements of this standard. Commercial-grade dedication of computer components includes the following:

— Special tests and inspections
— Commercial-grade survey of suppliers
— Source verification
— Acceptable supplier/item performance record
— A combination of the above four methods

C.2.3.1 Special tests and inspections

Special tests and inspections should be used if the technical data are available, test facilities are available, and the COTS items are such that inspection and tests upon receipt are adequate to verify critical characteristics. Special tests and inspections may also be used in combination with other acceptance methods.

Special tests and inspections are most appropriate for

— Items furnished from multiple suppliers
— Items that are relatively simple in nature
— Items on which post-installation tests can be conducted to verify critical characteristics

Copyright © 2003 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

Critical characteristics data is generally available in documents such as specifications, drawings, instruction manuals, bills of material, and catalogs. Interfaces with the supplier may be necessary to obtain the required data. Where sufficient data to utilize special tests and inspections cannot be obtained from suppliers because of proprietary considerations, other methods of acceptance should be considered.

Special inspections and tests should be performed in addition to, or in conjunction with, the standard receipt inspection. Where special tests and inspections are identified, a documented plan or checklist should be developed to verify selected critical characteristics. The results of the special tests and inspections should be documented in an approved plan/checklist that includes the following:

a) The COTS items included within the scope of the special tests and inspections

b) Tests and inspections to be performed

c) Test methods and inspection techniques to be used (documented test and inspection procedures may be required as appropriate)

d) Specifications, drawings, instruction manuals, bills of material, catalogs, etc.

e) Acceptance criteria that addresses the characteristics being verified

f) Documentation requirements for inspection and test results

C.2.3.2 Commercial grade survey of supplier

A commercial grade survey of a supplier is a means by which the purchaser can take credit for the commercial controls used by the manufacturer or supplier of a COTS item or line of replacement items. A commercial grade survey of the supplier should be used when the purchaser desires to accept commercial grade items based on the merits of a supplier's commercial quality controls. These controls may constitute quality programs, procedures, or practices.

Commercial grade surveys are most appropriate for

— Equipment not qualified under a supplier's 10 CFR 50, Appendix B program [B16]
— Original equipment manufacturers/suppliers (OEM/OES)
— Remanufactured equipment suppliers
— Distributors

A commercial grade survey can be used to accept simple or complex items. Commercial grade surveys of suppliers are most appropriate in the following situations:

— There is a single supplier of the COTS item
— Required technical information cannot be obtained from a supplier
— A large group of items are repeatedly procured from a supplier for an entire line of components
— The COTS item consists of an assembly of many parts
— The purchaser cannot easily verify the critical characteristics by special tests or inspections

The survey criteria and the supplier controls may vary from item to item. The survey criteria and necessary supplier controls should be determined by the purchaser and will depend on the number and type of critical characteristics. The survey should be specific to the scope of the particular COTS items being purchased. When many items are purchased from a supplier, a survey of representative groups of commercial grade items is sufficient to demonstrate that adequate controls exist. For each item, appropriate quality controls should be confirmed as being exercised and properly documented.

Copyright © 2003 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

The following types of supplier controls should be surveyed to assure typical critical characteristics are being controlled:

— Design control
— Procurement
— Software life cycle process
— Material control
— Fabrication
— Assembly
— Calibration
— Tests
— Inspections
— Problem reporting and corrective action programs

Other controls may be necessary as they relate to the critical characteristics being verified.

The results of commercial grade surveys should be documented in an approved survey plan/checklist that includes the following:

— COTS items included within the scope of the survey
— Critical characteristics to be controlled by the supplier
— Corrective actions and notifications
— Supplier controls to be verified specific to the critical characteristics
— Survey methods or verification activities
— Conclusions attesting to the adequacy of the supplier controls

Deficiencies identified during the COTS item supplier survey may be corrected by the supplier by instituting additional controls or by using other acceptance methods in this standard to verify adequacy.

Where a supplier demonstrates adequate controls and the critical characteristics have been verified, only verification of the part number, model number, version and revision number (as applicable), and the supplier's Certificate of Conformance is required during the standard receipt inspection to complete COTS item acceptance.

The purchaser should assess the frequency for which survey information must be reconfirmed based on factors such as supplier performance, item complexity, standard receipt inspection results, and frequency of procurement.

C.2.3.3 Source verification

Source verification is the verification of critical characteristics by witnessing quality activities before the COTS item is shipped. The basic purpose of source verification is to confirm that the supplier satisfactorily controls selected COTS item critical characteristics. Source verification is most appropriate for a single COTS item or a shipment of items or items procured on an infrequent or expedited basis.

When it is confirmed during a source verification that the supplier adequately controls the critical characteristics, the part number, model number, serial number, version and revision number (as applicable) should be verified upon receipt. The item is accepted upon completion of the standard receipt inspection and documentation of the source verification results.

Copyright © 2003 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

The controls to be witnessed by the purchaser will vary from item to item and are dependent upon the number and type of critical characteristics. The scope of the surveillance should include witnessing development processes, participating in software life cycle process audits and activities, or witnessing performance of factory acceptance tests. It should include confirmation of the supplier's design, procurement, and control methods employed for the particular commercial grade item being purchased.

The results of the source verification should be documented in an approved surveillance plan/checklist, which includes the following:

— COTS items included within the scope of the surveillance
— Critical characteristics to be controlled by the supplier
— Supplier controls to be verified specific to the critical characteristics
— Surveillance methods and verification activities
— Evaluation of the adequacy of the supplier controls

Deficiencies identified during the source verification may be corrected by the supplier instituting additional controls, or by using other acceptance methods in this standard to verify adequacy.

The COTS item commercial dedication documentation should provide objective evidence that control of specific critical characteristics was observed.

C.2.3.4 Acceptable supplier/item performance record

Use of acceptable supplier/item performance records allows the purchaser to accept commercial grade items based upon a confidence in the COTS item achieved through documented performance of the item, in combination with other commercial dedication processes.

The results of supplier tests may be used to verify certain critical characteristics. Information such as reliability data from operating nuclear power generating stations, suppliers, or industrial users that support performance history of the item may also be considered. Acceptable supplier/item performance records are most appropriate for COTS items where results of historical performance can be compiled using

— Monitored performance
— Vendor product performance data
— Industry product tests and performance data (e.g., Part 21 notices)
— Summaries of component failures in nuclear power plants
— Other industry data bases (e.g., military or aerospace)
— Documents issued by the USNRC (e.g., Information Notices)

Documented historical performance data should be used in combination with special tests and inspections, supplier surveys, and source verifications. The performance data should be directly applicable to the critical characteristics.

Performance history should be determined primarily by monitoring the performance of an item that was purchased from a particular supplier, and by monitoring the performance of the parent component in which the item was installed. This performance data is normally available from maintenance records. The dedicator should use this approach with caution, as all failures of a component may not be reported to the supplier, or failures of a component may occur after the analysis was completed. The supporting documentation should be periodically updated and reviewed to assure the supplier/item maintains an acceptable performance record.

Copyright © 2003 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

Evaluations of historical performance data should be documented. This documentation should include the following information:

a) Supplier/item being evaluated

b) Previously established critical characteristics specific to the item or supplier

c) Identification of the utility/industry data used in evaluating the supplier/item

d) The basis for determining that industry data substantiates acceptability of the supplier/item

e) A summary justifying the acceptability of the supplier/item

Acceptance is based upon engineering judgment that sufficient evidence exists to provide adequate confidence for the use of the existing commercial computer. All acceptance activities should be planned and documented. In addition, justification should be documented for exceptions to the acceptance criteria.

Copyright © 2003 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 D

(informative)

Identification and resolution of hazards

D.1 Background

Computer development requires identification of hazards (i.e., abnormal conditions and events, or ACEs) that have the potential for defeating a safety function. A hazard is a condition that is a prerequisite to an accident. Hazards include external events as well as conditions internal to the computer hardware or software. This annex provides guidance for identifying, evaluating, and resolving hazards. It presents a brief discussion on the use of fault tree analysis (FTA), failure modes and effects analysis (FMEA), and an adaptation of the concepts presented in IEEE Std 1228™-1994 [B12] and MIL-Std-882B [B15] in the form of considerations that might be used during the design process. The concepts from these standards include various design analyses and checklist issues. Additionally, this annex presents guidelines for the resolution of hazards once they are identified. Identified hazards should serve as input to appropriate V&V activities (see 5.3.3) and reliability calculations (see Annex F).

One method of determining hazards is through the use of analysis techniques such as FTA and FMEA. IEEE Std 603-1998 (5.15 through reference to IEEE Std 352™-1987 [B6]) suggests using an FMEA for performing reliability analyses. These techniques can be useful for identifying potential hazards. IEEE Std 1228-1994 [B12], and MIL-Std-882B [B15], identify a different technique for identifying hazards. This technique is one that attempts to identify the introduction of hazards during the design process.

D.2 Discussion

Hazards can result from system considerations (e.g., design bases conditions, failure modes of system components, human error, etc.), or from the specific design and implementation interactions (e.g., subsystem interface incompatibility, buffer overflows, input/output timing, initiation states, out-of-sequence events, etc.). Design and V&V activities should provide adequate confidence that the identified hazards have been appropriately addressed.

Subclause 5.5.1 requires consideration of hazards that have significant potential for adversely affecting computer hardware or software elements that are essential for performing safety functions. The significance of a hazard is based upon the probability of occurrence and the consequences of the occurrence. Only those events that produce consequences that could defeat required safety functions should be considered. Either quantitative or qualitative judgment of the probability of occurrence should be sufficient to determine if further action is required. Only those conditions with significant consequences and significant probability of occurrence need to be resolved in the design process. The conditions considered and the basis for significance determination should be documented. This annex does not supersede the requirements of IEEE Std 603-1998.

D.3 Purpose of hazard analysis

The purpose of a hazard analysis is to explore and identify conditions that are not identified by the normal design review and testing process. The normal design verification and validation process ensures that the design requirements are met by the safety system. Normally this process will evaluate different combinations of failures as required by the plant design basis and the effect of failures on the system. The scope of

Copyright © 2003 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

hazard analysis extends beyond plant design basis events by including abnormal events and plant operations with degraded equipment and plant systems. Hazard analysis focuses on system failure mechanisms rather than verifying correct system operation. The critical failure modes that could result in hazards are identified and then evaluated to determine the associated risk level (e.g., high/unacceptable or low/acceptable risk) based upon both the probability of occurrence and the consequence of occurrence. The failure modes that are classified as having unacceptable risks can be further evaluated to determine the causes of the specific failure modes. The probability of occurrence can be determined quantitatively or qualitatively. Most hazard analyses should be qualitative, while a quantitative analysis can be used to determine prioritization.

D.4 Hazard analysis implementation guidelines

The following guidelines should be considered when performing a hazard analysis:

— Avoidance of hazards
— Identification and evaluation of hazards
— Identification of hazards throughout the system life cycle
— Resolution of hazards
— Evaluation of hazards in previously developed systems
— Documentation of hazard analysis plans, responsibilities, and results

D.4.1 Avoidance of hazards

The following good engineering practices are normally used in the design process to reduce the number of hazards during the system design phase:

— Use of industry standards as guidelines to avoid hazards. Industry experts have developed standards that define methods on avoiding some of the hazards that have been observed.
— Use of checklists. Using technical checklists is another method of hazard avoidance. Technical checklists are listings of known hazards or methods on how to avoid hazards. Checklists are normally developed from past hazard experiences. Because checklists may not capture all hazards, checklists should be used to supplement a structured hazard analysis.
— Use of experts in the different development areas such as software development, systems maintenance, design engineering, and operations.
— Use of requirements analysis can assist in the early identification of system hazards.

D.4.2 Identification and evaluation of hazards during the detailed design phase

During the detailed design phase, the system hazard analysis process should be structured to maintain a minimal impact on system development activities (e.g., no organizational changes, minimal process changes). Additionally, the process should be adaptable for either a life cycle or a post-design analysis. This process should begin early in the plant upgrade process (life cycle approach) with development of a plan describing the anticipated project-specific hazard analysis activities. Life cycle hazard analysis processes are described in D.4.3.

D.4.2.1 Structure

To facilitate the hazard identification process, hazard identification should be an integral part of the normal design process. Hazards identification becomes part of the life cycle process by making it part of the normal design process in the early stages of system development as opposed to making it a back-fit process after the

--`,,`,,-`-`,,`,,`,`,,`---
Copyright © 2003 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

system has been developed. Hazards that are identified early in a system life cycle (e.g., in the design phase of the life cycle) can be corrected easier and for less cost than those hazards identified in a back-fit analysis process. The hazard identification process should use the same system development and maintenance elements that are used during the normal design process, such as the following:

— Project structure and organizations
— Design verification
— Review meetings
— Documentation (10 CFR 50 Appendix B process) [B16]
— Configuration management
— Testing (factory acceptance, simulation, and post-modification testing)
— Quality assurance

Life cycle hazards analyses should be designed to address changes during the development, testing, and implementation of the system.

D.4.2.2 Planning

A major resistance to hazard analysis that may exist at the beginning of system development is a requirement to maintain development costs as low a possible. Real and identifiable hazards do not exist at the start of the design process, so justifying or assigning resources to the hazard analysis process can be difficult to quantify or justify. A philosophy that the new digital system will be equal to or better than the system being replaced may not be always true.

At the start of the design project, a hazards identification and evaluation plan should be developed that includes the following steps:

— Identify critical functions such as reactor trip, emergency coolant injection, etc.
— Identify top-level undesired events (events that could lead to a loss of a critical function)
— Identify organizational responsibilities
— Select the techniques to be used
— Identify analysis assumptions
— Perform a hazards identification analysis
— Evaluate identified hazards for consequences and probability of occurrence
— Perform needed corrective actions and re-evaluate the impact of any changes with respect to the critical functions

D.4.2.3 Hazards identification

The first step in the hazards identification process should be to identify the critical functions of the subject system. A multi-discipline team approach should be used for the identification of the critical functions in all areas of the system development process (e.g., hardware and software development, operations, design, maintenance, and testing). Once the critical functions are identified, further analyses can be performed to identify events that would prevent these critical functions from occurring upon demand. Hazard identification should not rely on a single technique, but should use several different techniques during the course of the analysis. The following techniques can be used in the identification of potential hazards:

— Preliminary hazard analysis (PHA)
— Fault tree analysis and failure modes and effects analysis
— System modeling

Copyright © 2003 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

— Software requirements hazard analysis
— Walkthroughs (e.g., design reviews and code reviews)
— Simulator/plant model testing

D.4.2.3.1 Preliminary hazard analysis

Preliminary hazard analysis (PHA) is an initial identification technique that is similar to a brainstorming session among experts on various portions of the system. The list of critical functions and undesired functions of the system identified during the hazards identification planning process provides a starting point and scope of the PHA. A list of questions or a checklist can also be used to guide and focus the PHA discussions. A key to a successful PHA is choosing the participants, who should have a variety of backgrounds and perspectives of the system. The PHA team members could include the following:

— System engineers
— Design engineers
— Operators
— Maintenance personnel from the appropriate disciplines
— Software and system developers
— Probability risk assessment analysts

The listing of critical functions developed during the PHA process is used to focus on the safety significant areas of concern. The possible failure modes of these critical functions are then determined, prioritized by safety significance, and evaluated for probability of occurrence.

D.4.2.3.2 Fault tree analysis and failure modes and effects analysis

Fault tree analysis (FTA) and failure modes and effects analysis (FMEA) are techniques that can be used to determine hazards. These techniques address the introduction of hazards during the design process. FTA and FMEA are structured development methods that use the highest priority failures identified during a PHA. FTA is a top-down approach that focuses the analysis in the specific area of the cause of a hazard. A FMEA is a bottom-up approach that addresses a much broader area and can be evaluated against identified hazards to identify potential causes. Suppliers may be more comfortable performing FMEAs, although FMEAs do not address multiple failures such as common-cause failures, which should be evaluated. Both of these techniques can be useful for the identification of potential hazards.

A technique for identifying hazards is to enumerate failures and undesired consequences and then identify the specific system design or implementation that creates each failure or consequence. For example, an undesired consequence might be the failure to open a valve under specified conditions. This consequence may in turn be used as the top event in a FTA, which would then be decomposed to lower-level intermediate events and terminated in the lowest level of design for which qualitative or quantitative probabilities could be assessed. In this example, lower-level events might be hardware failures, operating system failures, sensor failures, device driver failures, or faults in the application software decision logic. If this high-level analysis identifies the software as a possible contributor to the undesired consequence, the FTA would then be expanded into the code modules and, if necessary, lower in the software design. This method serves as a tool to identify the most vulnerable sections of the design, which, if a design error or random fault were to occur, would be a significant contributor to the undesirable consequence.

D.4.2.3.3 System modeling

The system modeling technique is used to identify hazards in the system design by creating and subsequently executing software models of the system design. With this modeling approach, abnormal conditions can be introduced to determine the effect of the conditions on system performance.

 Copyright © 2003 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

D.4.2.3.4 Software requirements hazard analysis

This activity is similar to PHA, except the focus is on the software requirements and the interfaces between the software and other components. Additionally, a software requirements hazard analysis checklist can be used to identify omissions and inconsistencies in the documented software requirements. After the software requirements hazard analysis is performed, all potential hazards should be prioritized according to their potential affect on critical functions. If a high priority hazard directly affects the software, the hazard should be further defined and evaluated.

D.4.2.3.5 Walkthroughs

Requirements, design, and code walkthroughs should be a part of the software development process. Walkthroughs focus on the thorough examination of a specific portion of a system, and should involve personnel representing diverse areas of engineering expertise. The typical focus of a walkthrough is on correctness (e.g., assuring the design accurately addresses the requirements allocated to it). This focus should be extended to address hazard concerns (e.g., assuring the design does not allow the system to operate in an undesired or unexpected way).

D.4.2.3.6 Simulator/plant model testing

Test cases can be performed by connecting the digital system to a plant simulator or a plant computer model. During this testing, not only can the design requirements be verified but also potential hazards can be further explored and tested for validity and system response.

D.4.2.4 Hazards evaluation

Hazards evaluation activities focus on assessing the credibility of potential hazards. All potential hazards should be evaluated as early in the life cycle as possible to maximize the benefits of early hazard identification. The following steps are recommended for hazards evaluation:

— Evaluate hazard cost tradeoffs
— Determine the potential effects of a hazard
— Determine the category and type of hazards
— Identify and evaluate the system-level impact of hazards
— Determine the disposition of hazards

D.4.2.4.1 Evaluate hazard cost tradeoffs

The analyst should determine whether it is more efficient to eliminate a potential hazard than to evaluate whether the hazard poses a significant risk to the system.

D.4.2.4.2 Determine the potential effects of a hazard

There are two parts to establishing the potential effects of a hazard. Once a potential hazard has been identified, the analyst should determine whether the system is capable of producing the hazard. The analyst should then assess the possibility of an error or fault occurring as a result of the hazard. The results of a FTA could then be used to determine the relative importance of each hazard. This can be done by evaluating the probabilities of the minimal probability risk analysis cut sets or the sizes of the cut sets (see IEEE Std 352-1987 [B6]). If the probability of a fault or error is acceptably low for the occurrence of the undesirable consequence, no further action should be required. If not, a design change might be required or a safety evaluation of the total system may be needed.

Copyright © 2003 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

Secondly, the analyst should determine (qualitatively or quantitatively) the probability that the potential hazard will occur in an operational situation. This may require reliability data for hardware components. For software, this may involve an estimate of the probability that the software will be subjected to conditions or inputs that could cause the hazard to occur.

D.4.2.4.3 Determine the category and type of hazards

The hazards analyst should determine in which of the following categories the hazard originated:

— Hardware only
— Software only
— Both hardware and software (i.e., a system level problem)
— System development environment (e.g., faulty diagnostic equipment that results in an inaccurate hazard identification)

For root-cause determination, the analyst should also determine the development phase during which the hazard was incorporated into the system. For example, a hazard identified during the hardware implementation phase may actually trace to a problem in the hardware design, which may trace to a problem in the hardware requirements specification. In this example, the type of hazard is a requirements hazard. Other types of hazards include design, implementation, test, installation, maintenance and operation hazards. This information can be used to identify appropriate personnel for resolution of the hazard. Records of the types of hazards can be compared to identify weaknesses in the development process.

D.4.2.4.4 Identify and evaluate the system-level impact of hazards

The system-level impact of a hazard may be subtle, such as the display of an erroneous value that subsequently causes an operator to take an inappropriate action. Conversely, the hazard may be more obvious, such as causing the system to fail at an inappropriate time. Alternatively, higher-level logic or interlocks may prevent a potential hazard from having an undesired effect. The system level impact can be determined using an informal general analysis, or as part of an FMEA. Functional FTA may also be useful. Potential hazards should be prioritized based on a comparison and evaluation of system-level impacts. For example, a single random failure of a hardware module that causes an undesirable consequence for any single computer may be acceptable in a system design that has sufficient redundancy. Another example is the use of diversity and defense in depth to compensate for common-mode software vulnerability. The affect of the hazard on the overall system can be used to prioritize hazards.

D.4.2.4.5 Determine the disposition of hazards

Determining the disposition of a hazard involves deciding whether to confirm and subsequently resolve the hazard. The system developer should evaluate the system-level impact and credibility of the hazard and determine whether the hazard should be withdrawn or confirmed. Several methods are used to address hazards such as elimination, reduction of exposure, and controlling or minimizing the effects of a hazard. If eliminating a hazard is not cost beneficial, then the hazard should be addressed by redesigning the system. Controlling the hazard should not be the responsibility of the nuclear plant operator. The hazard analysis should be revised to address design changes to identify newly created hazards.

D.4.3 Identification of hazards throughout the system life cycle

The following subclauses provide specific guidance for evaluating hazards throughout the system life cycle.

Copyright © 2003 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

D.4.3.1 Safety system hazards identification

The safety system hazards identification process begins with an understanding of the required safety functions, design basis conditions, selected system design elements (e.g., subsystems, diverse systems, etc.), and regulatory requirements.

The process should consider the following:

- Occurrence of design basis conditions identified in the plant safety analysis report
- Possible independent, dependent, and simultaneous hazards events considering failures of safety equipment, including power supplies and common-cause conditions that could create a hazard
- Interface considerations among various elements of the system, (e.g., electromagnetic interference, inadvertent actuations of hardware and software controls). This should include consideration of the potential contribution by software (including software developed by others) to subsystem/system mishaps. Safety design criteria to control safety software commands and responses (e.g., inadvertent command, failure to command, untimely command or responses, or undesired events) should be identified and appropriate action taken to incorporate the safety design criteria in the software (and related hardware) specifications.
- Environmental constraints including the operating environments (e.g., seismic, temperature, rapid temperature changes, noise, exposure to foreign substances, fire, electrostatic discharge, lightning, electromagnetic environmental effects, and radiation)
- Operating, test, maintenance, and emergency procedures (e.g., human factors engineering; human error analysis of operator functions, tasks, and requirements; effects of factors such as equipment layout, lighting requirements; effects of noise or elevated temperature)
- Design and use of test and maintenance equipment that has the potential for introducing faults and software errors
- Safety equipment design and possible alternate approaches (e.g., interlocks, system redundancy, hardware or software fail safe design considerations, and subsystem protection)
- Degradation in a system caused by the operation of another subsystem (including non-safety systems)
- Modes of failure, including reasonable human errors as well as single point failures, and hazards created when failures occur in subsystem components
- Potential contribution of software (including software that is developed by others), events, faults, and occurrences (such as improper timing) on safety of the system
- Potential common-mode failures
- The method of implementing the software design requirements and corrective actions that could impair or degrade the safety system or introduce new hazards
- The method of controlling design changes during and after system acceptance to assure the safety system is not degraded and new hazards are not created

D.4.3.2 Computer hazards identification

As a result of the safety system analysis, certain safety functions, design conditions, limitations, and unresolved hazards may be identified for the computer system. The computer system design should specify those functions that will be required of the hardware or software to a) prevent the system from entering a hazards state, or b) move the system from a hazards state to a non-hazards state. Additionally, the interfaces between the software and the computer system should be identified and specified. Identification of computer-level hazards should consider items similar to those described in D.4.3.1, except the focus should be on the conceptual design of the hardware and software. Additionally, the following should be considered as potential hazards:

Copyright © 2003 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) Interdependencies between hardware and software such as interrupts and the operating system

b) Sequences of actions that can cause the system to enter a hazard state

c) System-specific credible hazards, such as the following:

— Early or late outputs
— Sensor input processing failures
— Precision or round-off problems
— Improper handling of exceptions
— Recovery actions
— System interrupts
— Electrical input voltage and frequency fluctuations
— Maximum credible number of coincident signal changes
— Electromagnetic interference
— Out-of-range values (e.g., dividing by zero or non-initialized pointers)

D.4.3.3 Software requirements hazards identification

The software requirements hazards identification process evaluates software and interface requirements and identifies errors and deficiencies that could contribute to a hazard. The compatibility of software requirements with the hardware design should be an underlying issue in the performance of software requirements hazards identification, which include the following activities.

a) Software requirements should be evaluated to identify those that are essential to accomplishing the safety function (i.e., critical requirements). These critical requirements should be evaluated to assess the significance of hazard conditions.

b) Timing and size requirements should ensure adequate resources for execution time, clock time, and memory allocation are provided to support the critical requirements, including maximum loading under worst-case conditions.

c) In designs involving the integration of multiple software systems, consideration should be given for interdependencies and interactions between the components of the systems.

d) Existing software should be evaluated to ensure adequate confidence that no "unintended functions" detrimental to the operation of the safety system are introduced. Possible interpretations of unintended functions include:

— Unused resident functions. The design process should address any unused resident functions (see 5.6). In some cases, such as for operating systems and compilers, the V&V process is unsuitable for addressing unused resident functions as the total number may be unknown.
— Unpredictable responses to external or internal conditions. A documented effort to identify unpredictable responses to external or internal conditions should be made during the design process, and appropriate actions should be taken to resolve these hazards. The V&V process should then confirm proper response to these hazards.
— Defects due to design or implementation errors. The V&V process should address defects due to design or implementation errors.
— Development aids not removed from the software. A documented decision should be made to state whether development aids will remain in the software. If the decision is made to leave the development aids in the software, they could be left active or made inactive. In either case, if development aids are to remain in the software, V&V activities should be performed.

Copyright © 2003 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

D.4.3.4 Software design hazards identification

Software design hazards identification consists of activities that provide adequate confidence that no new hazards are introduced. Potential computational problems, including incorrect equations, insufficient precision, scan rate, and sign convention faults should be evaluated. Equations, algorithms, and control logic should be evaluated for potential problems including

- Logic errors
- Cases or steps that have not been addressed
- Duplicate logic
- Extreme conditions neglected
- Unnecessary functions
- Misinterpretation of requirements
- Missing condition tests
- Checking wrong variable
- Incorrect loop iterations

Evaluation of data structures and their intended use should be performed for data dependencies that circumvent isolation partitioning, data aliasing, and fault containment issues affecting safety and the control or mitigation of hazards. Potential data handling problems, including incorrect initialized data, accessed or stored data, scaling or units of data, dimensioned data, and scope of data should be evaluated.

Interface design considerations, including internal and external interfaces with other modules of the system, should be reviewed. The major areas of concern with interfaces are properly defined protocols, and control and data linkages. External interfaces should be evaluated to verify that communication protocols in the design are compatible with interfacing requirements. The interfaces evaluation should support claims of redundancy management partitioning, and hazards containment. Potential interface and timing problems include interfaces addressed incorrectly, incorrect input and output timing, and subroutine/module mismatches.

Assurance that the design is enveloped within the identified system constraints should be addressed. The impacts of the physical environment on this hazard analysis can include such items as the location and relation of high-frequency clocks to circuit cards and the timing of bus latches when using the longest safety timing constraints to fetch data from the most remote circuit card.

Software modules that implement critical functions should be identified. Potential problems may occur in interfacing and communicating data to other modules, incompatibility of word format or data structure, synchronization with other modules for purposes of data sharing, etc. Additionally, non-safety modules should be evaluated to provide assurance that they do adversely affect safety software.

D.4.3.5 Software implementation hazards identification

Hazards can be created during the software implementation life cycle phase. The following activities should be performed during software implementation:

- Evaluate equations, algorithms, and control logic for potential problems including logic errors, omitted cases or steps, duplicate logic, omission of extreme conditions, unnecessary functions, misinterpretation of requirements, missing condition tests, variables not checked, and incorrect iteration of loops.
- Confirm the correctness of algorithms including accuracy, precision, and equation discontinuities, out of range conditions, breakpoints, erroneous inputs, scan rates, etc.

Copyright © 2003 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

— Evaluate the data structure and usage in the code to provide assurance the data items are defined and used properly.
— Confirm interface compatibility of software modules with external hardware and software.
— Confirm the software operates within the constraints imposed upon it by the requirements, the design, and the target computer system to ensure that the program operates within these constraints.
— Examine non-critical code to assure the code does not adversely affect the function of critical software. As a general rule, safety software should be isolated from nonsafety software. The intent of this examination is to prove this isolation is complete.
— Verify the software code is within timing and sizing constants.
— Verify the use of good software practices such as limits on code size, avoidance of multi-use registers, control of reusable code, initialization of code, etc.

D.4.3.6 Computer system integration testing for hazards conditions

Computer integration testing verifies that hazards requirements (e.g., inhibits, traps, and interlocks) have been correctly implemented. This testing verifies the software functions properly within its specified environment. This testing should occur as an inherent part of testing activities performed during computer development. The following activities should be performed during computer system integration testing:

— Computer software unit level testing to verify correct execution of safety software elements
— Interface testing to verify safety software units operate as expected
— Computer software configuration item testing to verify the execution of the software as a unit
— System-level testing to verify software performance within the overall system
— Stress testing to verify the software will not cause a hazard under abnormal circumstances, such as unexpected input values.

D.4.3.7 Computer system validation testing

Tests similar to those described in D.4.3.6 should be performed on the software operating in the final hardware configuration as part of the overall V&V process to provide assurance that identified hazards have been addressed.

Software FTA (see D.4.2.3.2) can be useful for identifying software faults that could cause the loss of a safety function, or to validate that such faults do not exist.

D.4.3.8 Maintenance and modification hazard analysis

Consideration should be given to identification of hazards that could be introduced as a result of maintenance and modifications made after system acceptance. The extent of the analysis is determined by the scope of the maintenance and modification. Guidelines from D.4.4 should be followed to address these hazards.

D.4.4 General guidelines for hazards resolution

Computer-based safety systems consist of hardware and software components whose functions are essential to accomplishing safety functions. Additionally, the computer system may contain components that are not essential to accomplishing safety functions (e.g., self-tests). The focus of hazards identification and resolution is to assure the safety functions are protected from identified hazards, and nonsafety functions do not create hazards for the safety functions when the nonsafety components are subjected to identified hazards. The following general guidelines should be addressed:

 Copyright © 2003 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

— Identify, evaluate, and eliminate hazards associated with each system throughout the entire life cycle of the system. At each phase of the development life cycle, resolve hazards that were unresolved in earlier phases and analyze the design existing in the current development phase to identify new hazards.
— Minimize risks resulting from excessive environmental conditions (e.g., temperature, pressure, seismic, vibration humidity, radiation, and electromagnetic interference).
— Address in the system design risks created by human errors in the operation and support of the system.
— Create unambiguous requirements definitions to minimize the probability of misinterpretation by developers. Potential problems include ambiguous statements, unspecified conditions, precision requirements not defined, response to hazards not defined, and requirements that are: incomplete, incorrect, conflicting, difficult to implement, illogical, unreasonable, not verifiable, or not achievable.
— Consider and use historical hazards data, including lessons learned from other systems.
— Minimize risk by using existing designs and test techniques wherever possible.
— Analyze for hazards and document changes in design configuration or system requirements (see D.4.6).
— Document identified hazards and their resolution (i.e., design changes or determination of no further action) (see D.4.6).

Once a hazard is identified, the resolution of the hazard should be addressed. The following guidelines for resolving identified hazards should be addressed:

— Eliminate identified hazards or reduce the associated risk through design, if possible.
— If an identified hazard cannot be eliminated by changing the system design, reduce the associated risk to an acceptable level by adding safety devices.
— When neither design nor safety devices can effectively eliminate an identified hazard or adequately reduce the associated risk, devices should be used to detect the condition and to produce an adequate warning signal to alert personnel of the hazard. Warning signals and their application should be designed to minimize the probability of incorrect personnel reaction to the signals and should be standardized within like types of systems.
— Where it is impractical to eliminate hazards through design selection or adequately reduce the associated risk with safety and warning devices, develop procedures and training to address reactions to the occurrence of the hazard.

D.4.5 Evaluation of hazards in previously developed systems

Previously developed computers, hardware, or software components of developed systems may be used in the design of safety systems. Subclause 5.4.2 requires that previously developed systems be qualified using a commercial grade item dedication process. Annex C provides guidance for commercial grade item dedication, including consideration of hardware, software, or firmware failures (i.e., hazards that could interfere with accomplishing the safety function). The guidelines of D.4.3 and D.4.4 should be applied to the extent possible, realizing that extensive software design hazards identification and software code hazards identification may not be achievable or necessary.

D.4.6 Documentation of hazard analysis plans, responsibilities, and results

Documentation of hazard analysis plans, responsibilities, and results is important to ensure that these activities are conducted in an orderly manner and the results are auditable. The activities described in this annex should be integrated into the computer development process. The activities may be documented in those

Copyright © 2003 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

documents that govern and record the development process possible in the form of a checklist. A separate set of documents is not recommended.

D.4.7 Preliminary hazard analysis questions

The following example questions provide guidance for conducting a hazards analysis:

1) How could the system malfunction in a way that will cost the company more than a million dollars (not counting subsequent design changes)?
2) How could the system malfunction in a way that would defeat the safety function?
3) Does the way the user will interact with the new system differ significantly or subtly from the way the user interacts with the existing system?
4) What would happen if the operator followed the old procedures while using the new system?
5) What would happen if a maintenance technician followed the old procedures while using the new system?
6) What would happen if a maintenance technician made online changes to the system?
7) Are the system inputs and outputs incompatible (electrically and mechanically) with the corresponding plant interfaces (i.e., is there an interface problem)?
8) Is there any potential failure of the system (especially a failure that causes a system lock-up) that is not obviously indicated to the operator?
9) Are bus contentions and timing problems possible under any operating conditions (within the operations environment specified in the requirements specification)?
10) Are the new operations, maintenance, or training procedures incompatible with the current procedures? (Will the plant personnel have a problem knowing what to do with the new system?)
11) Is there a potential for new system test procedures to introduce new hazards into the system (e.g., a test that leaves a safety system function inhibited after the test is completed)?
12) Are the system self-diagnostics active or passive? How do the self-diagnostics affect the system?
13) Does the system have hardware or software interrupts? If it does, how do they affect the system? Are unused hardware interrupts tied to a reference potential such as ground or are they left floating, which could result in a system failure?

NOTE—No one person will have the background to complete this checklist so it should be a team-prepared checklist.

Copyright © 2003 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 E

(informative)

Communication independence

E.1 Background

The use of computers in safety systems has provided an opportunity for a high level of data communication between computers within a single safety channel, between safety channels, and between safety and non-safety computer systems (see also 5.6). Improper use of this communication ability could result in the loss of a computer's ability to perform its function or multiple functions and thereby inhibit the safety system from performing its function. This annex provides detailed methods that could be employed to allow the greatest use of communication without negatively affecting the safety system. Isolation should be considered in order to prevent fault propagation between safety channels and from a nonsafety computer to a safety computer.

E.2 Discussion

Whenever communication techniques are employed, the major concern relates to the need to eliminate the potential loss of safety functions as a result of communication activities. This includes transmission of data and any vehicle for acknowledging receipt of the data or indicating a failure in data transmission. The detection and correction of any communication failures should not be allowed to impede or interfere with the performance of safety functions.

For proper independence of the safety computer from nonsafety equipment, both electrical and communication isolation should be ensured. It should be noted that electrical and communication physical points of isolation may be different. Electrical isolation requirements are provided in IEEE Std 384™-1992 [B7]. Recommendations for methods of communication isolation follow.

E.2.1 Communication between computers in different safety channels

Communication between computers in different safety channels may be desired for such purposes as voter logic or time stamp synchronization. Upon a failure of the communication, the preferred failure state should be set if one has been identified. Figure E.1 and Figure E.2 depict ways in which this can be accomplished.

Figure E.1 depicts broadcast communication between a safety computer in channel A and a safety computer in channel B. The one-way communication path provides a point of software isolation. The physical link(s) between the computers provide electrical isolation. This isolation may be accomplished optically (i.e., fiber optic cable or optical isolators). Communications isolation is provided through the broadcast communication.

The buffering circuit provides an interface allowing acknowledgment or no acknowledgment of data transfer between channels, collision avoidance, etc. It serves as a buffering feature between the communications link and the safety function to ensure integrity of the safety function. The buffering circuit should be separate (e.g., at a minimum on a different board) from the processor performing the safety function. The buffering circuit may be another processor, memory card(s), etc. V&V activities should include the buffering circuit. The physical link between the buffering circuits should serve as the point of electrical isolation.

--`,,,-`-`,,`,,`,`,,`---
Copyright © 2003 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 E.2 depicts a method with two separate points of isolation, one electrical and one for communications. This method allows two-way communication between safety computers, as long as a buffering circuit is employed.

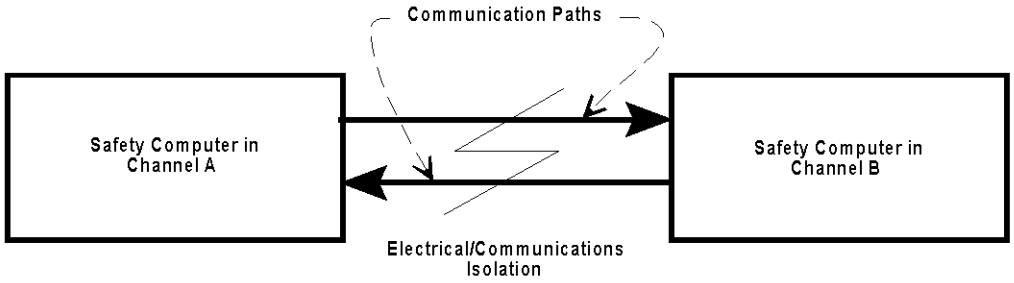

Figure E.1—Communication between safety channels (one-way communication)

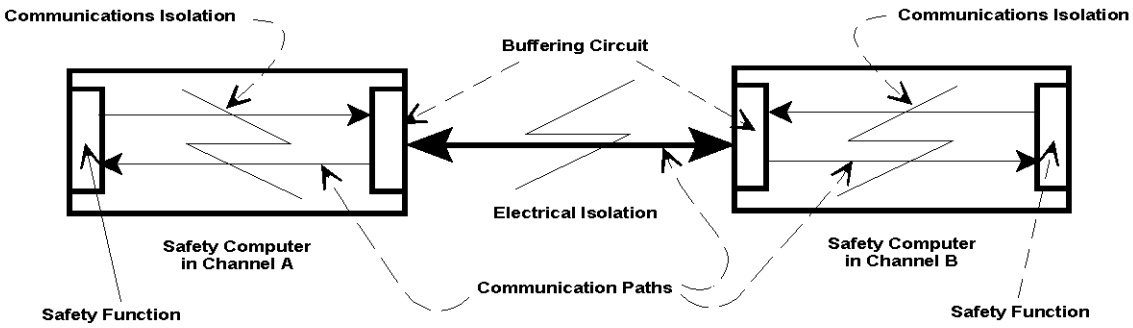

Figure E.2—Communication between safety channels (two-way communication)

The broadcast communication link between the safety function and the buffering circuit serves as a route for data to be sent out by the safety computer. The separate communication from the buffering circuit allows the safety function processor to receive data from another channel. The process of requesting and receiving data from another channel should not result in loss of either of the safety functions.

E.2.2 Communication between safety and nonsafety computers

Communication between safety and nonsafety computers may be desired for purposes of time stamp synchronization and installation of approved setpoint changes. However, at no time should the safety computer require input from the nonsafety computer in order to perform its safety function. The following figures depict ways communication between safety and nonsafety computers can be accomplished.

Figure E.3 graphically shows a broadcast communication between the safety computer and the nonsafety computer. The one-way communication path provides for communication isolation. The physical link(s) between computers might provide both electrical and communications isolation as required. Electrical isolation may be accomplished optically (i.e., fiber optic cable or optical isolators). Communications isolation is provided through the broadcast communication path.

Copyright © 2003 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

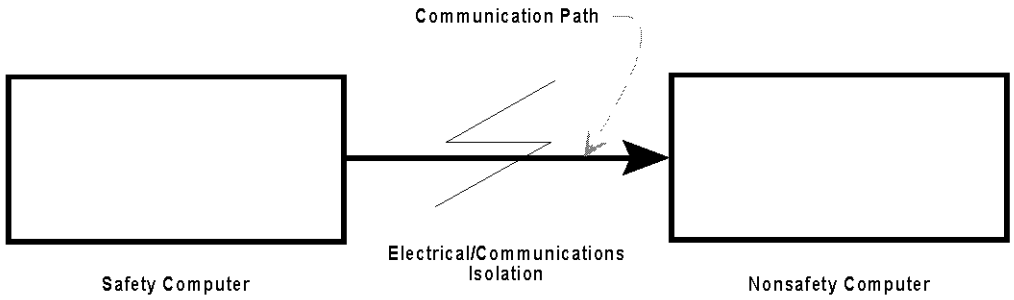

Figure E.3—Communication between safety and nonsafety computers (one-way communication)

Figure E.4 depicts a method with two separate points of isolation, one electrical and one communications. This method allows two-way communication between the safety computer and the nonsafety computer, as long as a buffering circuit is employed in the safety computer. Use of this method may be necessary when a separate computer is used for test and calibration purposes.

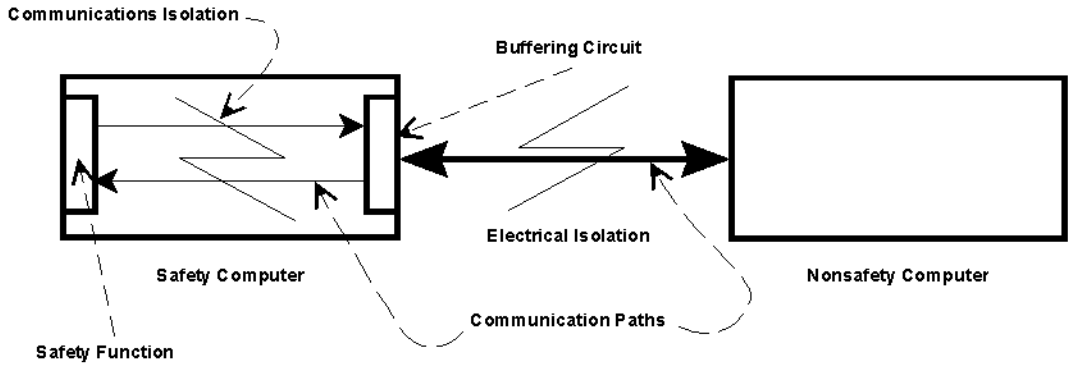

Figure E.4—Communication between safety and nonsafety computers (two-way communication)

The buffering circuit provides an interface allowing acknowledgment or no acknowledgment of data transfer between channels, collision avoidance, etc. It serves as a buffering feature between the communications link and safety function to assure the integrity of the safety function. The buffering circuit should be separate (i.e., at a minimum on a different board) from the processor performing the safety function. It may be another processor, memory card(s), etc. V&V activities should include the buffering circuit. As required, the link between the buffering circuit and the nonsafety computer provides electrical isolation.

The broadcast communication link between the safety function and the buffering circuit serves as a route for data to be sent by the safety computer. The process of sending data should not result in loss of the safety function. The broadcast communications link from the buffering circuit to the safety function is necessary when a separate test and calibration computer is employed.

Copyright © 2003 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 E.5 is similar to Figure E.4 except that an optional buffering circuit and communication path is employed in the nonsafety computer.

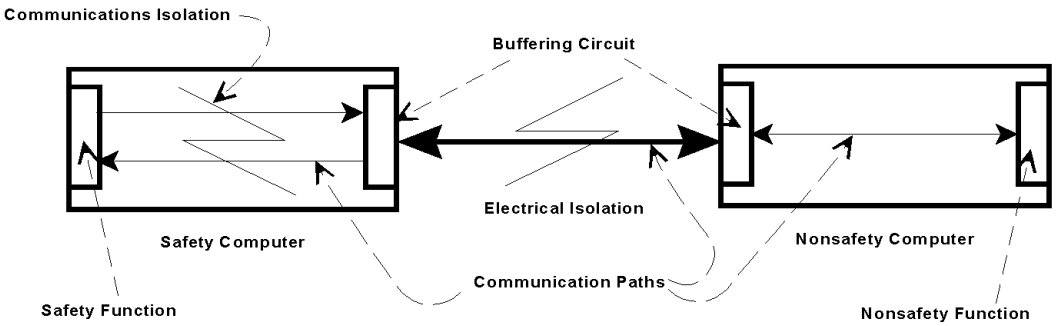

Figure E.5—Communication between safety and nonsafety computers

Copyright © 2003 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 F

(informative)

Computer reliability

F.1 Background

Subclause 5.15 requires the consideration of both hardware and software whenever a quantitative reliability goal is required. This annex provides guidance on how a measurement may be made. The approach taken here considers the computer as a whole (i.e., hardware, system software, firmware, and applications). Other methods that predict software reliability using structural attributes such as volume, cyclometric complexity, data flow, extrapolation of mean time between failure (MTBF) growth from development, failure and correction experience, and others have not yet reached a sufficient state of maturity to provide adequate confidence in the reliability predictions.

The method presented in this annex provides a quantitative measurement of reliability with the following limitations:

— Computer reliability is measured with respect to the functional requirements in the specifications. This method does not account for specification defects. Significant proportions of failures in tested systems are attributable to defects in specifications.
— This method results in a quantitative reliability measurement of nonredundant computers (i.e., single string processing in a single channel). This annex does not prescribe a measurement or other methodology to account for the effects of common-mode or common-cause failures on the reliability of redundant computers.

When applying the results of reliability measurements with these limitations in a reliability prediction, the results should be adjusted to account for possible errors in specifications and the unavailability contribution from common-mode and common-cause failures. Because of the variety of installed safety system designs and software architectures, it is not possible within the scope of this annex to prescribe widely applicable methods to perform these adjustments.

Characteristics and consequences of real time computer failures are as follows:

— Failures occur at unpredictable times. Consequently, any failure (originating in either hardware or software) can be treated as a random event in a manner similar to analog hardware failures.
— Failures exist as result of software being executed on a processor interacting with the environment through sensors, actuators, communication interfaces, and displays. Consequently, it is most meaningful to consider software reliability in conjunction with the integrated computer system upon which it is executed.

Failures are the result of the interactions of events with defects in the hardware, software, or firmware. Consequently, the potential exists for the development process to not account for some of these types of failures.

Failures generally involve a number of subtle problems, including interfacing with system software (operating system, device drivers, etc.), unanticipated behavior, failure/error handling, or timing and processor loading.

Copyright © 2003 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

Because either a hardware or software problem can cause a computer failure, it is necessary to address how the probability of both hardware and software originated failures can be best characterized, measured, and analyzed.

The most problematic area of computer reliability assessment is the treatment of software. In the method presented here, software failures are treated no differently than hardware failures. However, it is necessary to distinguish between two types of computer reliability calculations: those oriented to prediction versus those oriented to measurement.

Computer reliability measurement is both possible and desirable as part of the reliability determination of a safety system. One measure of computer reliability is in terms of a failure rate, specifically, the number of failures per unit of time. Measurement of computer reliability is then similar to measurement of safety system or hardware reliability and, in fact, should be based on the same set of data (i.e., determine the number of failures and the amount of time in the test period).

A key element in the measurement process is the recording and reporting of failures. In accordance with IEEE Std 1012-1998 the term anomaly report is used to refer to a trouble or failure report. Anomaly reporting and analysis is important to assess the reliability of the computer.

Specific information that anomaly reports provide includes requirements or design inadequacies, qualitative data on failure modes, quantitative data on computer (hardware and software) reliability, and data on performance capability (e.g., capacity, response time, throughput). The specific items for inclusion in anomaly reporting are described in IEEE Std 1012-1998. As data is collected, it can be analyzed for MTBF as measured in the integrated operational system by identifying problematic hardware or software modules; failure detection and recovery effectiveness (i.e., coverage); and the rate of discovery of unanticipated hazards (see Annex D).

F.2 Discussion

A necessary condition for computer reliability assessment is to complete anomaly data recording. Given a complete anomaly database, it is possible to analyze the effectiveness of the development process and determine computer reliability.

F.2.1 Development process evaluation

Evaluation of the development process can minimize the existence of computer failures resulting from the result of interaction of events with defects in the hardware, software, or firmware (see F.1). Anomaly reports written as a result of missing or incorrect requirements, failure modes identification, failure detection, recovery effectiveness, and reliability assessments should be considered. Analysis of the root cause and development phase data items should be evaluated.

F.2.2 Use of anomaly report data

Anomaly reports identifying previously unidentified failure modes, ineffective detection mechanisms, and ineffective recovery mechanisms should be considered as input for determining computer reliability.

The integration of computers into safety systems can potentially increase the number of hazards. Failure modes specific to the architecture, application, hardware, and other characteristics of the design may not be anticipated. Hence, a FMEA (see Annex D) should be maintained and updated as new failure modes are identified through anomaly reports.

Copyright © 2003 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

Establishment of failure classes and discrimination criteria is necessary to distinguish between different types of failures. For example, a computer "crash" might be distinguished from a computer "hang" on the basis of whether the operating system failed or only the application software failed. A necessary condition to using computer failure data for reliability measurement is a conceptual basis for the manner in which failures occur and propagate.

Failure-mode data also should be collected and analyzed specifically to characterize the behavior of the software at and beyond the maximum stress limits for throughput, capacity, and data to determine the behavior of the software under anomalous external conditions. Over time, a decreasing trend in the rate of discovery of new failure modes should be observed. The absence of such a trend may be an indication that the failure behavior of the system has not yet been adequately characterized. Determination of computer reliability may need to be delayed until this downward trend is observed.

Fault detection and recovery effectiveness (i.e., coverage) is an important factor affecting reliability in redundant computers. This effectiveness or coverage can be estimated as the number of successful recoveries after a fault occurs divided by the number of relevant (depending on the classifications used in the FMEA) failures occurring, and then applying the appropriate confidence intervals. During the development process, thorough fault injection and failure simulation testing should occur. However, the most effective measurement of coverage is through spontaneous failures occurring during operation. Failure reports providing data on coverage and on individual coverage mechanisms are the primary basis for making such estimations. Determination of reliability may need to be delayed until confidence exists in the detection and recovery effectiveness.

The above evaluations of anomaly reports are useful in developing new systems and in the integration of commercial grade computers. Operating experience data in similar applications, specifically failure mode and reliability analyses, is important for determining reliability of previously developed or commercial grade software. Establishing the degree of similarity, completeness, and comprehensiveness of the data collected is essential to characterizing the quantitative and qualitative reliability of such software.

F.2.3 Computer reliability

The reliability of a nonredundant computer in a safety channel can be divided into two measurable components: MTBF measurement, and the correct response probability. MTBF is an indication of the sustained operation over a long period. The correct response probability is an indication of the computer response to an initiating event given that the underlying hardware, system software, and other runtime environment components are functioning correctly.

F.2.3.1 MTBF measurement

MTBF can be estimated by dividing the cumulative operating time by the number of failures. Policies for determination of both operating time and computer failures should be specifically defined. Examples of issues that should be addressed in defining operating time include the following:

- Specification of relevant hardware and software configurations over which operating time can be collected
- The means by which unchanged modules running under previous releases can be credited for operating
- The fidelity of the external inputs and outputs, and how processor operating time data are collected

The following are examples of issues that should be addressed in defining failures:

- What constitutes a relevant failure

Copyright © 2003 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

— Discrimination criteria used to attribute the failure to the computer as opposed to a transient failure or other phenomenon
— Accounting for failures that have been corrected
— Accounting for multiple failures caused by the same defect

Once both cumulative operating time and failures have been defined, upper and lower bounds for the MTBF can be calculated using the procedures described in section 4.6 of MIL-HDBK 781 [B14], under the assumption that the MTBF is constant during the measurement period. Once a failure rate has been measured in this fashion, it is possible to determine the quantitative reliability over a defined time interval assuming an exponential or alternative reliability distribution.

F.2.3.2 Correct response probability

Correct response probability is the measurement of the probability of success upon demand. This process can be performed using data from system testing. Success can be determined by running relevant test cases and determining the proportion of successful versus unsuccessful test cases. The following issues should be considered as a part of this determination:

— Establishment of a representative and unbiased test case set. In cases where only a one-time output is needed (e.g., the initiating signal for a reactor scram), the test case sets need consider only the combinations of inputs. However, where continuous closed loop control is involved, the test case sets should also account for the ranges of anticipated operating times as well as the dynamics of the controlled system.
— The means by which success proportions can be determined when only a sample of the input space is tested rather than the entire input space (testing of the entire input space may be feasible for many simpler safety systems).
— Fidelity of the test environment and the handling of uncertainties in the operating environment with respect to the test environment.
— The means by which results of previous testing can be combined with regression testing.
— Handling of partially versus totally successful results, particularly in the cases of continuous control (e.g., valve oscillation over a portion of a run).
— Methods for determining actual success or failure, particularly for continuous control functions.

Methods for combining the results of test cases to determine an overall single probability of correct response depend on the nature of the tests and the anticipated results. For example, where a discrete success/failure result can be ascertained (as is typically the case with a sense and command function), the weighted average of successes and failures and a confidence interval derived using a binomial distribution may be appropriate. In other cases involving continuous closed loop control over an extended time period, a continuous scoring method may have been adapted. In such cases, a different approach to defining the confidence interval may be necessary.

F.2.3.3 Combining MTBF and correct response probability

Combining MTBF and the correct response probability into a single reliability quantity can be performed in the manner shown below with the following assumptions:

— The MTBF is transformed to a reliability estimate using an exponential distribution unless other indications are present.
— The MTBF is representative of the entire computer, including system hardware, software, device drivers, and other components of the runtime environment. The reliability estimate represents the probability that the system will function without a failure over a given time interval so that if there is a challenge to the safety system, the relevant data will be successfully input to the application.

Copyright © 2003 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

— The correct response probability represents the probability that the application will output a correct result given that the data associated with the safety system challenge has been successfully placed in its input buffer.
— The failure probability of the underlying system software and runtime environment and the failure probability of the logic used in the safety system application are independent.

Given these assumptions, it is possible to determine the channel reliability as follows:

$$Rch = Rc \times S$$

where

Rch is the channel reliability,
Rc is the channel reliability derived from the measured MTBF,
S is the correct response probability

F.2.3.4 Combining channel reliabilities to define a safety system reliability

Techniques similar to those for hardware as presented in IEEE Std 352-1987 [B6] can be applied to define the reliability of a safety system. However, the following issues should be considered:

a) To what extent the channel failures are independent, i.e., the relative importance of common-mode and common-cause failures. The latter could be due to use of the same hardware, operating system, operating conditions, or surveillance and maintenance procedures.

b) The means by which output from individual channels are combined and the extent to which failures occur at the point of combination.

c) System response time requirements versus recovery and repair time requirements.

In cases where MTBF may be sufficient to meet reliability allocations or goals, but the correct response probability adjusted for common-mode and common-cause failures cannot be shown to meet such allocations or goals, it may be necessary to introduce diverse implementations of the function-specific software into the safety system (see Annex B). When such measures are necessary, an evaluation (see Annex D) should be conducted on the diverse implementations to establish that failure modes are diverse and uncorrelated.

Copyright © 2003 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 G

(informative)

Bibliography

[B1] ANSI/ANS 51.1-1983 (R1988), Nuclear Safety Criteria for the Design of Stationary Pressurized Water Reactor Plants.

[B2] ANSI/ANS 52.1-1983 (R1988), Nuclear Safety Criteria for the Design of Stationary Boiling Water Reactor Plants.

[B3] EPRI TR-104595-V1, Abnormal Conditions and Events for Instrumentation and Control Systems: Volume 1: Methodology for Nuclear Power Plant Digital Upgrades; Volume 2: Survey and Evaluation of Industry Practices Report.

[B4] IEC 60880 (1986-09), Software for Computers in the Safety Systems of Nuclear Power Stations.

[B5] IEEE 100, *The Authoritative Dictionary of IEEE Standards Terms*.

[B6] IEEE Std 352-1987 (R1999), IEEE Guide for General Principles of Reliability Analysis of Nuclear Power Generating Station Safety Systems.

[B7] IEEE Std 384-1992, IEEE Standard Criteria for Independence of Class 1E Equipment and Circuits.

[B8] IEEE Std 730-1998, IEEE Standard for Software Quality Assurance Plans.

[B9] IEEE Std 828-1998, IEEE Standard for Software Configuration Management Plans.

[B10] IEEE Std 1012a-1998, IEEE Standard for Software Verification and Validation—Content Map to IEEE/EIA 12207.1.

[B11] IEEE Std 1061-1998, IEEE Standard for a Software Quality Metrics Methodology.

[B12] IEEE Std 1228-1994, IEEE Standard for Software Safety Plans.

[B13] IEEE Std 1540-2001, IEEE Standard for Life Cycle Processes – Risk Management.

[B14] MIL-HDBK 781, Reliability Test Methods, Plans, and Environments for Engineering Development, Qualification and Production.

[B15] MIL-STD-882B, System Safety Program Requirements.

[B16] Title 10 of the Code of Federal Regulations (10CFR50)[5].

[5]Information on the Code of Federal Regulations is vailable at http://www.gpoaccess.gov/cfr/index.html.

Copyright © 2003 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