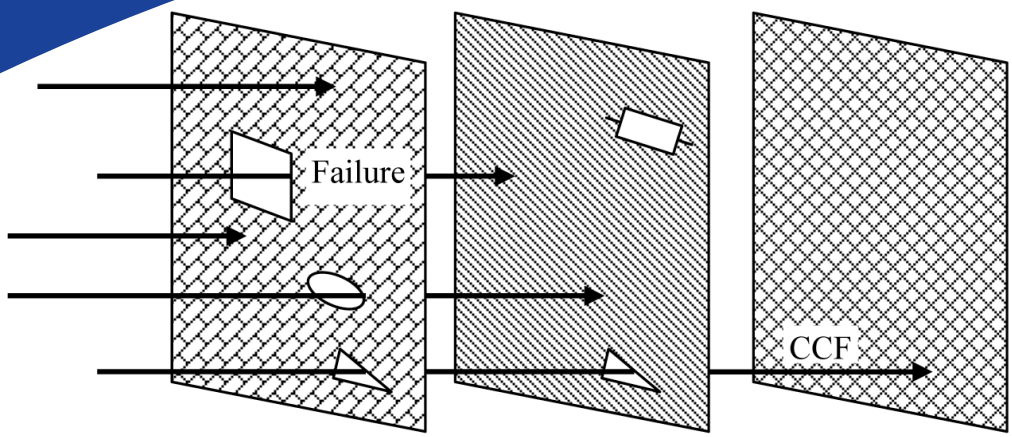
EPRI | ELECTRIC POWER RESEARCH INSTITUTE

2010 TECHNICAL REPORT

Protecting Against Digital Common-Cause Failure: Combining Defensive Measures and Diversity Attributes



Protecting Against Digital Common-Cause Failure: Combining Defensive Measures and Diversity Attributes

1019182

Final Report, November 2010

EPRI Project Manager
R. Torok

ELECTRIC POWER RESEARCH INSTITUTE
3420 Hillview Avenue, Palo Alto, California 94304-1338 ▪ PO Box 10412, Palo Alto, California 94303-0813 ▪ USA
800.313.3774 ▪ 650.855.2121 ▪ askepri@epri.com ▪ www.epri.com

DISCLAIMER OF WARRANTIES AND LIMITATION OF LIABILITIES

THE FOLLOWING ORGANIZATION(S), UNDER CONTRACT TO EPRI, PREPARED THIS REPORT:

Électricité de France (EDF)

NOTE

For further information about EPRI, call the EPRI Customer Assistance Center at 800.313.3774 or e-mail askepri@epri.com.

ACKNOWLEDGMENTS

The following organizations, under contract to the Electric Power Research Institute (EPRI), prepared this report:

Électricité de France (EDF)
6, Quai Watier
78400 Chatou, France

Principal Investigator
T. Nguyen

Electric Power Research Institute (EPRI)
3420 Hillview Avenue,
Palo Alto, California 94304-1338

Principal Investigator
R. Torok

This report describes research sponsored by EPRI.

This publication is a corporate document that should be cited in the literature in the following manner:

Protecting Against Digital Common-Cause Failure: Combining Defensive Measures and Diversity Attributes. EPRI, Palo Alto, CA: 2010. 1019182.

PRODUCT DESCRIPTION

Software common-cause failure (CCF) remains an unsettled technical and regulatory issue, both for new plants and digital upgrades at operating plants. The concern is the potential for CCFs to disable multiple equipment trains or systems that use identical software-based components. This report will help utilities ensure that plants have adequate protection against digital CCF.

Results and Findings

There is much that can be done to reduce the likelihood and/or effects of digital CCF, including the use of 1) development practices effective in avoiding, detecting, and eliminating errors, 2) hardware architecture and software design features that effectively preclude or limit the extent of certain types of failures, and 3) various types of diversity to prevent or mitigate CCF. This report provides technical insights on the mechanisms of CCFs induced by digital faults, such as software defects, in order to identify which protective measures could be taken or to help determine when digital systems might be considered susceptible to digital CCF.

Challenges and Objectives

Conventional approaches to protecting against digital CCF focus on using various forms of diversity—diverse digital platforms, software languages, programming teams, and functionality—while overlooking design and development attributes that arguably provide more effective protection and avoid unnecessary system complexity. The objective of this project was to present a more balanced approach that credits all contributors to CCF protection and allows the use of engineering judgment to determine the most appropriate combination of protective measures for specific applications.

This report is intended for engineers responsible for the specification, design, implementation, testing, operation, and maintenance of digital I&C systems in nuclear power plants. It is important that utility engineers have sufficient understanding of potential digital CCF sources and countermeasures to make appropriate design tradeoffs and informed decisions on plant modifications that involve digital systems.

Applications, Values, and Use

This report will help utility engineers specify and evaluate CCF protection for digital systems provided by vendors and integrators, reducing the likelihood of future CCF problems with new equipment. The report will also better prepare engineers to address regulatory questions regarding potential new failure modes and effects of digital equipment. Finally, the report will provide technical basis information in support of a failure analysis methodology for digital systems to be developed in 2011. The information in this report is applicable to both safety and critical non-safety systems that use redundant hardware architectures to help achieve high reliability.

EPRI Perspective

The potential for software CCF has been recognized for many years, but this study is one of the first to call attention to the importance of defensive design measures in protecting against software failures and CCF. The nuclear industry stresses the importance of applying industry standards to ensure high quality in digital systems. However, good processes cannot guarantee perfect results, and it is not generally possible to prove that software is defect free. Such a dilemma often points to diversity—the use of different software-based components or systems—as the panacea for CCF protection.

Diversity, however, is not the only means of protection against digital CCF, and in some situations is not the most appropriate. It necessarily adds complexity and is of limited value in protecting against requirements errors that can lead to CCF or, in typical safety-system architectures that use replicated hardware, divisions with identical functionality. It is important to consider that defensive design measures—now commonly used by equipment suppliers to eliminate many classes of defects or to render them harmless by preventing their triggering conditions—typically do not increase system complexity. The real objective is not defect-free software, but software that is highly unlikely to cause undesired or unsafe system behaviors. Defensive design measures may be the most important contributors to achieving this goal, augmented where needed by diversity. Further, operating experience points to other aspects of implementing and managing digital systems that have been more problematic than software as potential CCF sources (EPRI report 1016731, December 2008). Sound engineering judgment backed by supporting data should be applied to understand digital CCF vulnerabilities and select the appropriate combination of defensive design measures and diversity to provide adequate protection.

Approach

The project team systematically investigated digital faults, dividing them into two main categories: specification faults and design faults. The fault types were then related to their corresponding failure mechanisms, modes, and system-level effects, with emphasis on the types of defensive measures that are effective in managing them.

Keywords

Common-Cause Failure (CCF)
Diversity
Defense-in-Depth
Digital Systems
Instrumentation and Control Systems

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1
BACKGROUND

The Issue

The introduction of software-based instrumentation and control (I&C) equipment in critical safety and control system applications in nuclear power plants raises the concern that latent software and other 'digital' faults could create new failure modes that could defeat traditional protective features.

Safety systems achieve very high reliability in part by using independent and redundant hardware trains to ensure that they can perform their critical functions even in the presence of hardware failures. However, digital I&C introduces the potential for common-cause failures (CCFs) due to errors made during development that can render the redundant hardware architecture ineffective, and disable multiple equipment trains or systems that use identical software elements or shared resources. I&C designs for generation-critical non-safety systems are also migrating toward redundant, fault-tolerant digital architectures to improve reliability, and here again, errors made during development could lead to common-cause failure that could defeat such architectures.

The likelihood of hardware common-cause failure in nuclear safety systems is generally considered negligibly low due to factors such as the high quality standards applied in its development and manufacture, physical separation of redundant equipment, and the recognition that degradation mechanisms that could result in common failures (e.g., corrosion or premature wear-out) are slow to develop and would be detected in maintenance and surveillance activities before they could disable a critical function. For the purposes of this discussion, analog I&C is treated the same as other hardware components.

Digital systems are also developed using high quality standards and extensive testing, but software does not wear out in the sense that hardware does. Software failures result from faults (bugs) in the code that cause unintended or undesired behaviors under specific conditions, usually conditions unanticipated by the designer. Effectively, the software does exactly what it was programmed to do, but it may not do what the designer had intended. Such 'failures' are caused by designed-in errors that can originate in several ways. Examples are incorrect requirements or misunderstandings of requirements, or coding mistakes that have been designed into the code and overlooked during verification and validation (V&V). There is no significant wear-out mechanism. Hence, the notion that failures are likely to be detected during maintenance and testing before multiple redundancies have been affected does not generally apply. Even though with the right development methods, languages and tools it is possible to rigorously justify the absence of certain classes of faults, in most cases software cannot be

proven to be completely error free, so software faults are considered credible, and some software faults can become sources of CCF.

Protecting Against the Effects of Digital Commmon-Cause Failure

Utilities need to ensure that plants have adequate protection against digital CCF in the form of an appropriate combination of preventive and mitigative measures. Preventive measures reduce the likelihood of CCF; mitigative measures reduce the effects. Both contribute to overall protection.

There is much that can be done to reduce the likelihood and/or effects of digital CCF, including the use of development practices effective in avoiding, detecting and eliminating errors, the use of hardware architecture and software design features that effectively preclude or limit the extent of certain types of failures, and the use of diversity to prevent or mitigate CCF.

The most appropriate combination of preventive and mitigative measures to provide adequate CCF protection will likely vary with several factors – system complexity, functional complexity, likely CCF sources, safety and operational significance of CCF consequences, etc. Both types have advantages and disadvantages. Preventive measures are less likely to add complexity, but may not address some significant CCF sources. Some types of diversity provide significant CCF protection, but a diversity-only approach may unnecessarily increase system and operational complexities, while still missing significant CCF sources. This report is intended to help the reader better understand the potential sources of CCF and mechanisms for protecting against their effects.

Purpose & Scope

This report provides technical insights on the mechanisms of common-cause failures induced by digital faults, such as software faults, in order to determine which protective measures could be taken, or to help determine when digital systems might be considered susceptible to digital CCF. For the purposes of this discussion, a digital CCF is a systematic common-cause failure resulting from digital faults in I&C systems or components, e.g., a defect in the functional specification requirements, a design error in the software of a programmable logic controller (PLC), or an error in the logic coded in a field programmable gate array (FPGA). It is anticipated that this input will be helpful in helping system designers and system assessors on this topic.

This report does not cover other, non-digital types of CCF, such as those induced by random hardware faults, extreme environmental conditions, manufacturing or installation faults, or operation and maintenance errors (e.g., incorrect parameter values or miscalibrations) that are essentially the same for analog and digital systems. This is an important qualifier, because non-digital CCFs may actually be more common than digital CCFs, even for digital systems. While such non-digital failures remain possible (and do sometimes occur), nuclear quality assurance (QA) requirements, including proper procedures, documentation, environmental qualification testing, etc., are credited with providing reasonable assurance of adequate protection against them. It is anticipated that non-digital CCFs will continue to be addressed by existing QA processes and administrative procedures, and are not part of the digital CCF discussion.

Overview

Section 2 presents a brief overview of the notions of *digital faults*, *digital failure* and *digital common-cause failure* (CCF). It also discusses the notions of *failure mode* and *failure mechanism*.

Section 3 introduces the concept of *digital CCF context*, and considers three main cases: CCF of digital I&C systems in *different lines of defense*, CCF of *subsystems of the same I&C system* that are separated and implement diverse functions, and CCF of essentially *identical redundancies* of a safety I&C system.

Section 4 distinguishes four digital CCF categories: CCF due to *single point digital vulnerabilities* common to multiple functional units (i.e., I&C systems, subsystems or redundancies), CCF due to *identical or similar digital faults* in multiple functional units, CCF due to *failure propagation* from one functional unit to the other, and CCF due to *vulnerability to global stressing conditions* shared by multiple functional units. Section 4 also lists the necessary 'ingredients' for each of these mechanisms: protective measures against each mechanism may target one or more of these ingredients.

Section 5 proposes a number of recommendations to protect against digital CCF and their effects. The first set of recommendations aims at minimizing the potential for digital failure, targeting in particular digital faults and activating conditions. The second set of recommendations target each of the four digital CCF categories, considering, when appropriate, the different CCF contexts.

Conclusions, definitions and references are given respectively in sections 6, 7 and 8.

Appendix A provides tables summarizing the defensive measures suggested beyond the mere application of diversity.

2
UNDERSTANDING DIGITAL FAULTS, DIGITAL FAILURES AND DIGITAL CCF

The two notions of *fault* and *failure* must not be confused. A system may harbor a number of faults, but as long as none of them is challenged and activated by a particular operating condition, the system will behave as expected. It is only when an *activating condition* occurs and triggers an up-to-now dormant fault that the system will start deviating from the expected behavior and possibly fail. Protection against failure (and common-cause failure) can act on either or both of these two necessary 'ingredients'.

Digital Faults

A *fault* in a system (and in particular in a digital system) can be defined as a defect that may cause a reduction in, or loss of the capability of, the system to perform a required function when subjected to a particular set of normal or abnormal conditions. A fault may also cause the system to perform a function at an unintended time (i.e. spurious actuation).

Examples of faults include:

- Defects in hardware caused by aging.
- Errors or inadequacies in the system requirements specification.
- Errors in the design, manufacturing, installation, operation or maintenance of the system hardware.
- Errors in the design and / or the implementation of the logic of the system, e.g., in the software of a Programmable Logic Controller (PLC) or in the programming of a Field Programmable Gates Array (FPGA).

Some faults, like hardware defects or faults resulting from incorrect maintenance, appear at unpredictable times during the operation phase of the system lifecycle. Thus, this category of faults is often called *random faults*.

Others, like inadequate functional requirement specification or design errors, exist in the system right from the beginning of operation. Though their existence is postulated, the details of most or all of them are unknown (otherwise, they would have been corrected and removed from the system). This is generally due to the fact that, except for systems with very simple functionality, it is extremely difficult to ensure that a digital system does not contain specification or design errors, even when an extremely rigorous development approach has been applied, as is the case for digital I&C systems important to safety or plant performance. Such faults usually remain

dormant and unrevealed until one is activated by a specific set of conditions in the operation of the system, resulting in an inappropriate and undesired behavior.

This type of fault can also affect non-digital systems, as has been shown by the detailed analysis of some existing analog systems. However, being extremely powerful and versatile, digital technology provides the means for more functionally ambitious I&C systems. Indeed, this is the main reason for the ubiquity of the technology in all aspects of industry and everyday life. Greater functional ambitions also mean more complex functional requirements and designs, a higher potential for mistakes, significantly increased difficulty for verification and validation, and, ultimately, a much higher concern for residual faults that could lead to common-cause failure. Thus, in this document, such faults are called *digital faults*.

Digital faults may affect all aspects of a digital system, including the design of its hardware (in particular of complex electronic components such as microprocessors or FPGAs), of its software (if it is a microprocessor- or microcontroller-based system), of its architecture, and the specification of the system's functional requirements.

Categories of Digital Faults

In this document, digital faults will be divided into two main categories: *specification faults*, and *design faults*.

Experience in the operation of highly reliable digital systems in various industry sectors has shown that specification faults are an important, and sometimes a dominant source of digital failure.

For example, see "Safeware" from Nancy G. Leveson [35], Page 22:

"*Myth 4. Increasing software reliability will increase safety*

... most safety-critical software errors can be traced back to errors in the requirements - that is, to misunderstandings about what the software should do."

See also EPRI TR-1016731 "Operational Experience Insights on Common-Cause Failure in Digital Instrumentation and Control Systems" [37].

Specification faults typically result from specifications that are incomplete, incorrect or ambiguous. In particular, a significant percentage of digital failures involve systems that encounter off-normal conditions that were not anticipated or well understood at the requirements specification stage.

Design faults result from errors made in the subsequent phases of system development, after the specification of functional requirements.

They can come in a wide variety of types. For example, design faults include:

- Defects in system architecture (e.g., insufficient processing power, communication bandwidth or memory, use of components that are not fully compatible with one another).
- Defects in pre-developed components (e.g., in the software of the I&C platform or of a 'smart component').
- Defects in application-specific software or logic.

A specific category of digital faults is worthwhile mentioning: *intrinsic digital faults*. These faults (which could be specification faults or design faults) can usually be recognized without a detailed knowledge and understanding of the real functional requirements of the system. Their particular interest lies in the fact that software tools are now available that can be used to systematically detect them, either in a formal or semi-formal requirements specification (e.g., in a functional diagram), or in software or hardware design code (in the case of FPGAs).

They can also come in a wide variety of types. For example, intrinsic software faults include constructs that could cause:

- Use of non-initialized variables.
- Division by zero.
- Out-of-bounds indices.
- Digital overflows or underflows.
- Deadlocks (in multi-tasking software).
- Memory leaks.
- Unprotected access to shared resources.
- etc.

Minimizing Digital Faults

As already mentioned, with current techniques, guaranteeing that a real life digital system (not an oversimplified example) is completely free from digital faults is practically impossible. Two main strategies can be used, often complementarily, to minimize their potential as far as reasonably feasible: 1) **avoidance** (whereby some particular types of digital faults are minimized or cannot occur altogether); and 2) **detection** and subsequent removal (using various forms of verification and validation such as inspections and reviews, testing, and formal verification).

Digital Failures

A *failure* is a deviation of a function, system, subsystem or component (hardware or software) behavior from an expected behavior, such that a required service cannot be provided. For systems important to safety or to plant performance, one essential objective of system designers is to ensure that component or subsystem failures do not result in system or essential functions failures. Such component or subsystem failures are often called *partial failures*.

A *digital failure* is a failure that occurs when a digital fault is challenged and brought to life by an *activating condition*. This type of failure raises very specific concerns due to the deterministic nature of digital systems, which tend to exhibit the exact same behavior whenever they are put in the same operating conditions. In redundant systems in which the redundancies are identical or nearly identical and thus are likely to contain the same specification or design faults, this could lead to the concurrent, *common-cause failure* of multiple redundancies. Thus, contrary to other types of failures, such as those originating in random faults, redundancy does not necessarily provide an adequate defense against digital failures. Concurrent, common-cause failure could also affect multiple systems in different lines of defense if these systems contain identical or similar faulty components.

System developers should do (and indeed, they often do) their best to avoid, detect and remove digital faults, but they can also design their systems also minimize the potential for unexpected operating conditions that could act as activating conditions for residual faults. And fortunately, for digital systems specifically designed for very high reliability, it is generally possible to have a rigorous identification and analysis of all the factors that can influence the behavior of the system and its software.

Digital Common-Cause Failure

Common-cause failure, or CCF, can be defined as the concurrent failure of two or more functions, systems, subsystems or components as a consequence of the same cause. Concurrent means that the time interval separating the failures is too short to allow repairing the first failed item and restoring it to full operational order before one or several other items also fail. The term is usually used with reference to redundant equipment or systems or to uses of identical equipment in redundant systems. CCFs can occur due to design, operational, environmental, or human factor initiators.

A digital common-cause failure, or digital CCF, is a common-cause failure that results from the activation of a digital fault.

System Failure vs. Function Failure

Another important difference between digital failures and other types of failures is that a *function failure*, i.e., the inability to correctly perform a required function, does not necessarily imply a *system failure* in a conventional sense, where the system would be out of use and unable to perform **any** function. In particular when the function failure originates in an inadequate functional requirements specification, the system itself may be in full operational order and still be able to perform correctly other required functions, provided that these other functions do not depend on the failed one.

Failure Modes and Failure Mechanisms

The use of these two terms in many reference documents is sometimes confusing. Therefore, their use in this report (which is consistent to the definitions given in [36]) deserves clarification.

Failure Modes

A *failure mode* of a system, component or function is defined by its **external** behavior, with the system, component or function effectively viewed as a black-box. The set of failure modes that are theoretically possible is completely determined by the functional requirements applicable to, and the outputs of, the system, component or function. A priori, they are independent from design and implementation technology.

For a reactor protection function that has only one periodical Boolean output (decision to perform or not to perform the protective action, made at regular time intervals) locked when the protective action is triggered (output is latched in the triggered state), there are only three postulated failure modes:

1. No protective action, when one would be warranted.
2. Spurious protective action, when none is warranted.
3. No answer in the required response time.

This latter mode may be subdivided into sub-modes, depending on how late the answer is given.

Failure Mechanisms

A *failure mechanism* is defined as an event or chain of events occurring during operation and / or maintenance, that leads to the failure of the system, component or function.

For example, in a PLC-based system, a division by zero may raise an exception, resulting in the processor being stopped. In the previous example, barring other design measures, this will lead to failure mode #3 (no answer in the required time period).

Postulated failure mechanisms are dependent on the design decisions made and on the implementation technology used. It is to be noted that different failure mechanisms may lead to the same failure mode.

> Failure mode #3 may also be caused by a random hardware error in the microprocessor or in the memory), by a memory protection violation, …
>
> Similarly, a spurious protection action (failure mode #2) may result from very different mechanisms: random high energy radiation, specification errors, software implementation errors, wiring errors, …

3
CCF CONTEXTS

A key word in the expression *common-cause failure* is obviously the word *common*. The applicability and efficiency of the various protective measures against digital common-cause failure will depend heavily on what the failure is common to. Measures that are effective and legitimate in one context may have limited efficiency and / or create more problems in another. The three CCF contexts listed hereafter are typical of existing I&C architectures and systems.

I&C Systems in Different Lines of Defense

One of the most essential safety concepts in nuclear power plants (and in many other safety-critical installations) is the notion of *defense-in-depth*, i.e., the application of more than one line of defense for a given safety objective, such that the objective is achieved even if one of the lines of defense fails (see Figure 3-1). I&C architectures that are part of defense-in-depth schemes in safety-critical installations should be designed so that there is adequate protection against potential digital CCF of I&C systems in the different lines of defense that could jeopardize the defense-in-depth concepts of the plant.

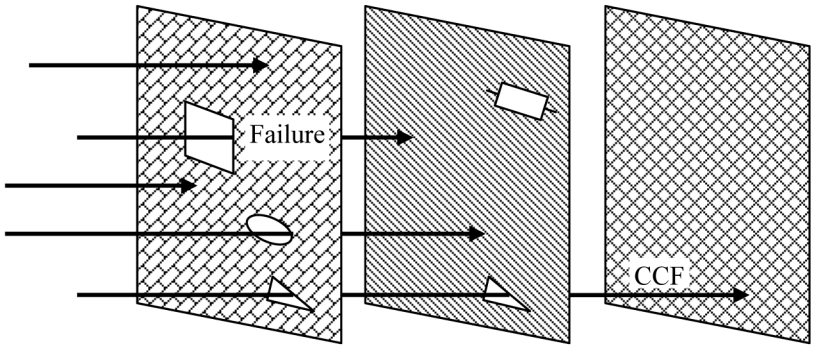


Figure 3-1
The Defense-in-Depth Concept:
Measures Are Taken So That Possible / Postulated 'Holes' in One Line of Defense Are Likely to Be Covered By One or Several Other Lines of Defense.

Subsystems Implementing Diverse I&C Functions

Digital I&C systems are often distributed systems, i.e., they consist of multiple, interconnected and interacting *computing units*. However, in some cases, the system architecture has provisions for two or more subsystems which are more or less well-separated (i.e., these subsystems do not communicate and do not interact with one another, or have limited and controlled communication and interaction) and which implement essentially diverse functions (see examples in Figures 3-2a and 3-2b). Thus, even when these subsystems are based on the same I&C platform and components, with appropriate protective measures against digital CCF, the failure of one subsystem will not necessarily imply the failure of the other subsystem(s).

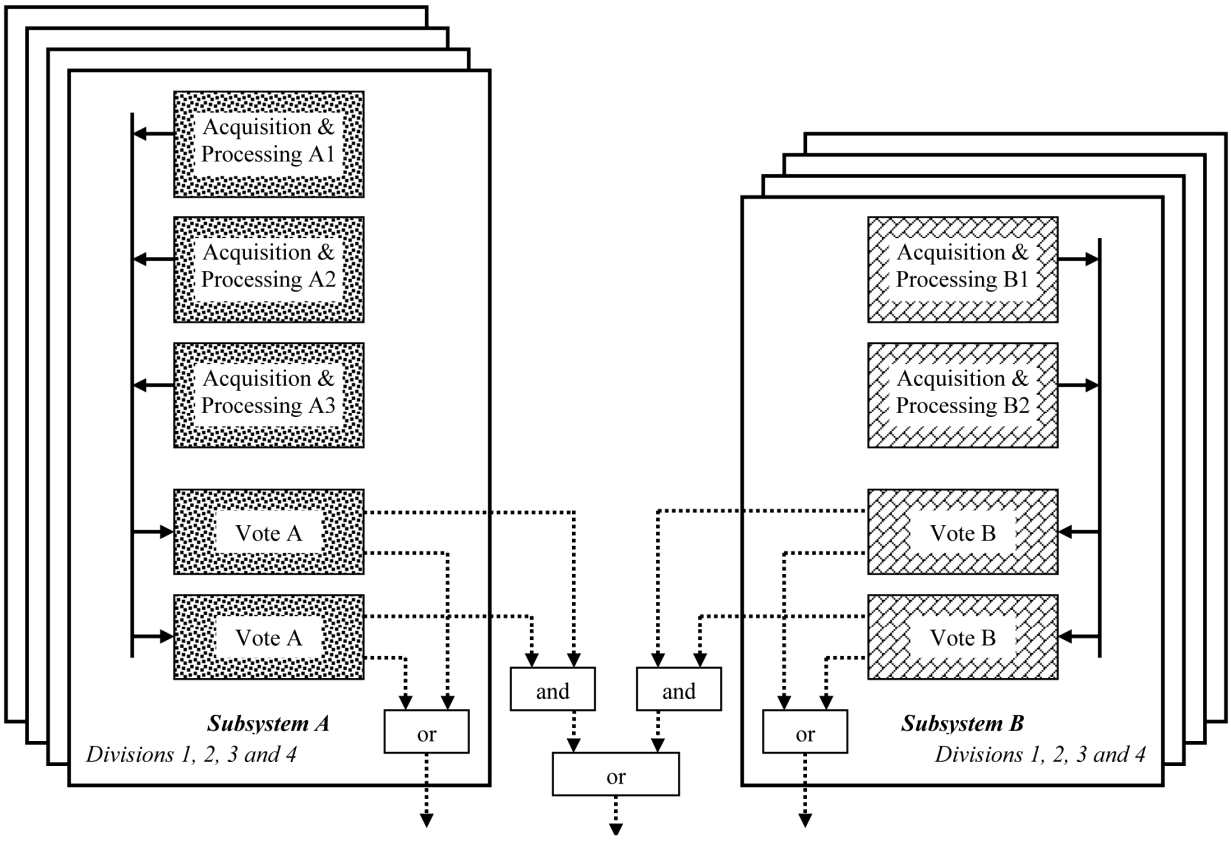


Figure 3-2a
Example of well-separated subsystems implementing diverse I&C functions. All processing units (Acquisition & Processing, and Vote) and all communication links are based on the same I&C platform.

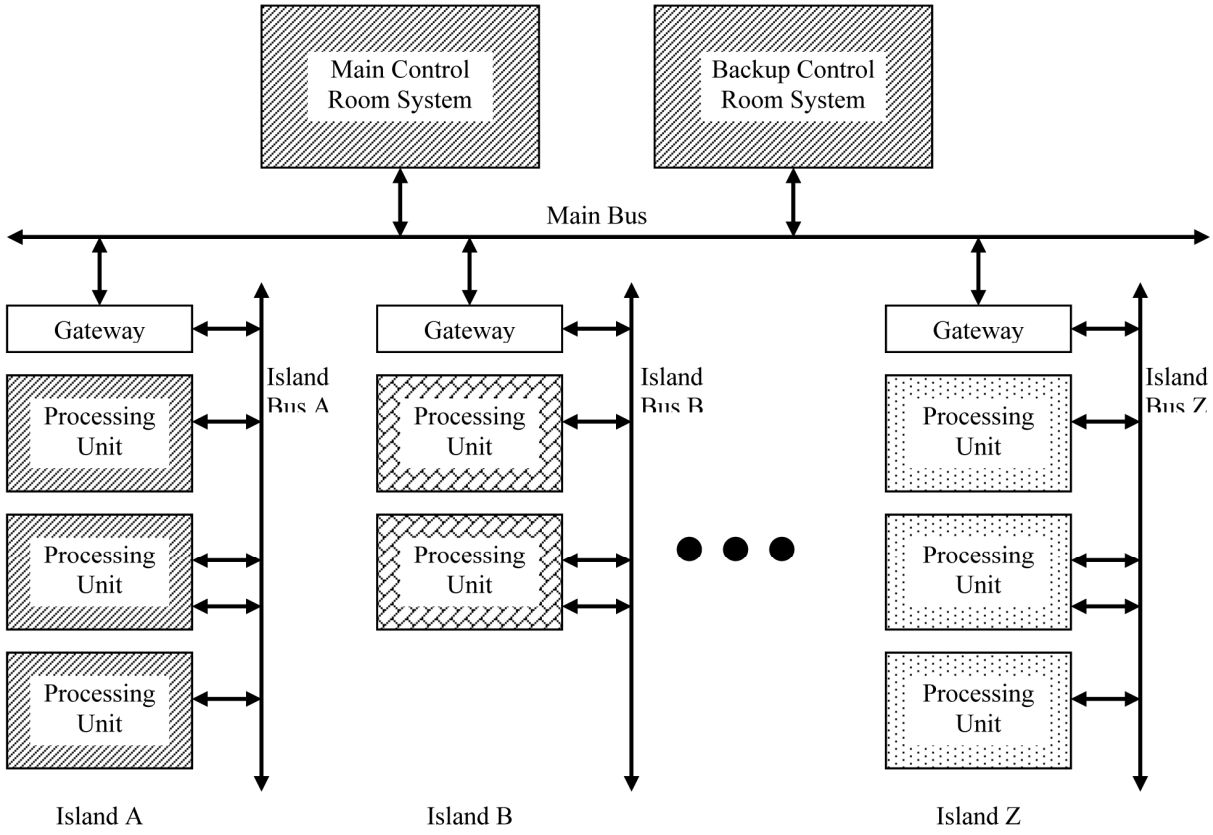


Figure 3-2b
Example of less well-separated subsystems (islands) implementing diverse I&C functions. As can be seen in some digital control system architectures. Islands do not communicate with one another, but communicate both ways to control room systems. High quality gateways provide some level of protection against erroneous data communication on the Main Bus.

Redundancies of the Same I&C System

Safety I&C systems are based on redundant architectures, with three or more independent but mostly identical redundant divisions implementing the same functions, possibly assorted with a voting subsystem (see Figure 3-3a). A number of measures, like geographical and electrical separation, are taken to guarantee that not all divisions will be lost due to a single physical event. However, digital safety I&C systems need to take additional measures to minimize the potential for digital CCF of the divisions.

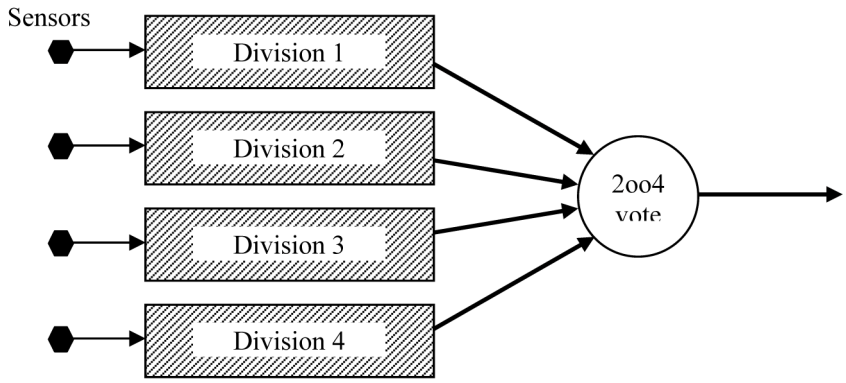


Figure 3-3a
Example of Digital System Architecture with Independent, Redundant Divisions.

I&C systems of lower safety significance, or non-safety I&C systems critical to plant performance, may also be based on a redundant architecture. For such systems, the Independent Redundant Divisions plus Vote architecture is not always the solution of choice, and master-slave architectures are often used (see Figure 3-3b). For such architectures, different protective measures may be used.

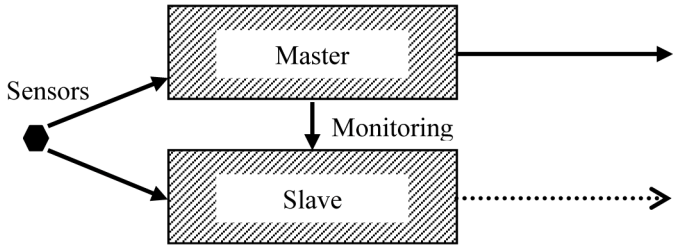


Figure 3-3b
A Master-Slave Architecture. The Slave Monitors the Health of the Master, and Takes Over the Master Fails.

4
CCF CATEGORIES

The Four CCF Categories

In order to provide adequate protection against digital CCF, or to properly assess the residual susceptibility to digital CCF, it is essential to have a clear and full understanding of the possible digital CCF mechanisms. One may classify these mechanisms into four categories.

In devising a defense strategy against digital CCF for a given I&C system or I&C architecture, each CCF category should be considered. As each category requires a specific list of 'ingredients', a possible approach is to target all or some of these ingredients.

Single Point Digital Vulnerability

When a single component that is susceptible to digital faults is shared by several functional units, the failure of, or incorrect interactions with, this component could cause a concurrent loss of multiple units (see Figure 4-1).

> The shared component does not really need to be faulty from its own standpoint: it could just, in certain conditions, have behaviors incompatible with the functional units. The component would then be considered faulty from the system standpoint. In this report, a faulty component is faulty from the system standpoint.

Smart power supplies, data communication links and digital real-time clocks providing common time references are examples of shared components that may be at the origin of this type of digital CCF.

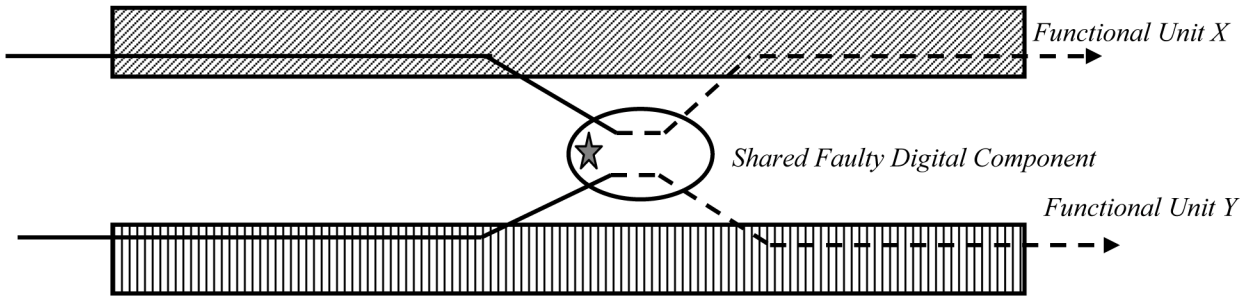


Figure 4-1
Digital CCF due to a shared faulty digital component.
The star represents a fault which, when challenged by an activating condition, causes the shared component to fail and send incorrect (faulty) information to the two functional units. The arrows represent the course of execution of the functional units.

The necessary ingredients for this CCF category are:

- A shared, single component.
- A digital fault in this component.
- A condition that activates the fault and leads to a faulty behavior and failure of the component.
- Inappropriate protection of the functional units against the credible failure modes of the shared single component.

Identical or Similar Digital Faults

Systems in different lines of defense, independent subsystems or redundancies may contain identical components, which in turn may harbor identical or similar digital faults which, if activated concurrently, could lead to the CCF of these units (see Figure 4-2).

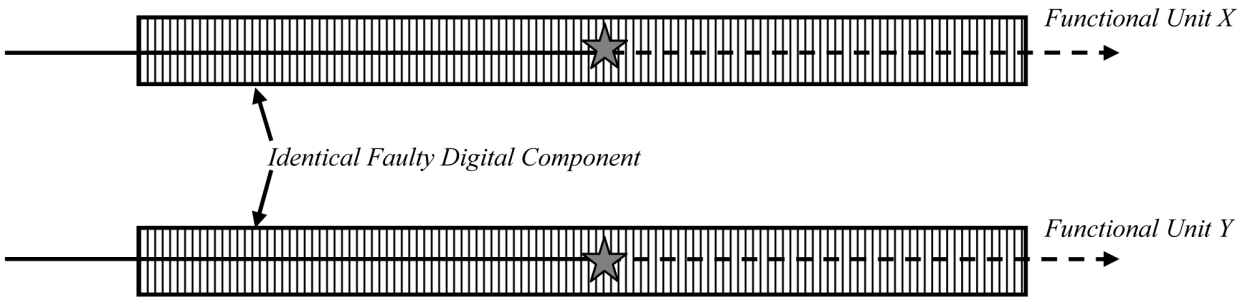


Figure 4-2
Digital CCF Due to Identical or Similar Faulty Components.

The necessary ingredients for this CCF category are:

- Identical or similar components in the various functional units under consideration.
- A digital fault that is replicated in these components.
- A condition that activates the fault and leads to the failure of one of the functional units.
- A concurrent occurrence of the activating condition in one or more of the other functional units, leading to the concurrent failure of multiple functional units.

Failure Propagation

A characteristic of most recent digital I&C architectures is the presence of various communication links. Both multiplexed networks and point-to-point connections have been used. From a functional standpoint, the use of such links may be justified, as they allow concentration and / or validation of information (e.g., to allow sound automatic or human decisions). Unfortunately, they also raise concerns of digital CCF: incorrect values or incorrect interactions from a failed functional unit may be conveyed through communication links and cause other units to fail concurrently (see Figure 4-3). The use of communication links may also increase the complexity of an architecture, with possible adverse effects upon the ability to analyze the performance of a system. It is to be noted that failure propagation does not necessarily require data communication networks: it could also occur with point-to-point communication, and even with non-digital, analog communication.

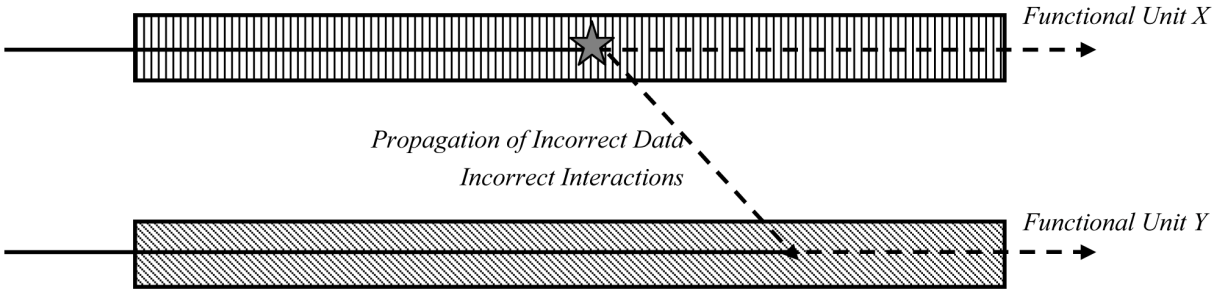


Figure 4-3
Digital CCF Due to Failure Propagation from Unit X to Unit Y.

The necessary ingredients for this CCF category are:

- A digital fault in one functional unit or communication link.
- An activating condition that triggers the fault and leads to the failure of the functional unit or communication link.
- The transmission, through a communication link, of the incorrect behavior to one or several other functional units.

- Inappropriate protection of these other functional units against possible incorrect communicated data values and interactions (including total lack of interaction on the part of failed functional units or communication links).

Shared Susceptibility to Global Stressing Conditions

Shared susceptibility to global stressing conditions such as special dates (e.g., January 1st 2000) or data communication storms could trigger dormant faults, even separate faults made independently by different design teams in diverse digital systems, and thus also result in digital CCF (see Figure 4-4).

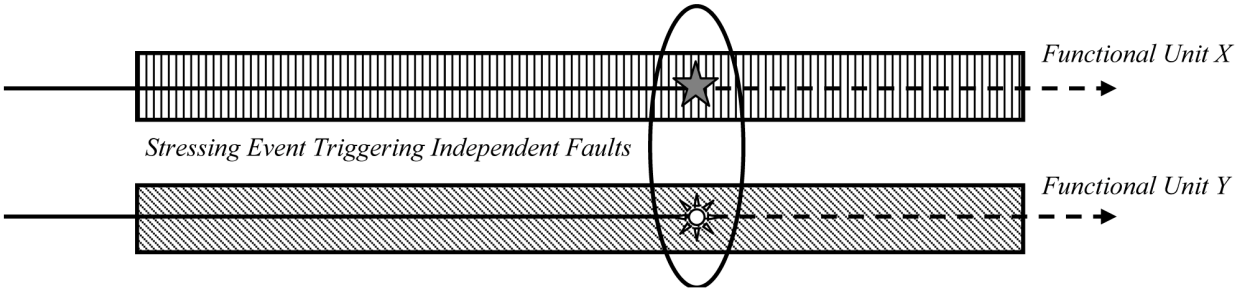


Figure 4-4
Digital CCF due to share susceptibility to global stressing conditions.
Here, the two functional units are diverse and have different faults. However, the two faults are activated concurrently by the same global condition.

The necessary ingredients for this CCF category are:

- A susceptibility (shared by two or more functional units) to a global stressing condition.
- The occurrence of this global stressing condition, affecting concurrently these functional units.
- One or more digital faults in each of these functional units that could be triggered by the stressing condition.

Note on Concurrent Revelation of Silent Failures

In digital I&C systems operating on demand (such as typical RPS and ESFAS), concurrent failures on demand could occur when multiple 'silent' failures occur during the stand-by period and remain unrevealed in multiple functional units. These silent failures could happen at different times due to unrelated and / or non-digital causes, are typically addressed by self-monitoring and periodic testing, and are usually not considered CCF.

5
PROTECTION AGAINST DIGITAL CCF

In this section, each of the four main digital CCF categories is considered, and possible protective approaches are presented. However, all four types of CCFs depend on the existence of digital faults and the occurrence of activating conditions that could give life to these faults and lead to failure. The first part of the section addresses the twin issues of minimizing, as far as reasonably achievable, the potential for digital faults and the occurrence of activating conditions. Such measures are in practice very efficient and much needed defenses against digital CCF.

The second part of the section addresses each of the digital CCF categories, and provides additional specific recommendations. As it concerns mainly digital CCF caused by identical or similar digital faults, diversity as discussed by US NRC NUREG/CR-7007 ORNL/TM-2009/302 *Diversity Strategies for Nuclear Power Plants and Instrumentation and Control Systems* [7] is presented in the corresponding sub-section. The CCF contexts discussed in section 3 are taken into consideration mainly in the cases of identical or similar digital faults, and failure propagation: the protective measures for the two other digital CCF categories (single point digital vulnerabilities and shared susceptibility to global stressing conditions) are independent from the CCF context.

Minimizing the Potential for Digital Faults

This very general topic has been the object of decades of effort by countless computer scientists, digital systems developers, software engineers, and electronic circuit designers. Therefore, this report only gives a very brief overview of the subject, and only for what may concern typical digital I&C systems of nuclear power plants. The measures mentioned here are described in their main principles. It will up to system designers to decide which measures to apply, and how, based on the specifics of their projects and systems.

Digital faults are a necessary ingredient of all four digital CCF categories. Thus, when digital CCF is of concern, it is of particular importance to do our best to minimize the potential for residual digital faults. This can be achieved using two complementary approaches: fault avoidance and fault removal. The general principles of the two approaches are presented first. Then, as most avoidance and detection measures target specific types of digital faults, their application to the different types of digital faults is discussed in more detail.

Fault Avoidance

Avoidance of digital faults is performed during development by the application of development rules and the use of development languages and methods that can preclude or seriously limit particular types of faults.

For example, dynamic memory allocation (whereby chunks of memory are requested in the course of software execution, used for information storage, and then relinquished when no longer needed so memory is available for future requests) is often a significant source of faults and failures. One can eliminate the issue altogether by using static memory allocation (whereby all memory segments necessary to the software program are allocated once and for all at the initialization of the digital system, at fixed, pre-defined locations).

The use of high quality code generators from high-level application-friendly languages is another example of widely used means to avoid software coding faults.

Appendix A proposes a non-exhaustive list of fault avoidance measures that could be considered by system designers. However, caution should be applied when using the list, as many of the measures are not applicable in all situations.

Some particular types of applications have no inherent bounds for memory usage, and for them, dynamic memory allocation is a useful and sometimes necessary technique. Attempts to force static memory allocation would simply displace the resolution of the issue, often to new, application-specific code that would probably be more error-prone than well-proven standard memory management software.

Fault Detection and Removal

As not all types of digital faults can be avoided altogether, fault detection and removal remains an essential part of the arsenal of digital systems developers. In general, various and complementary methods are used, such as rigorous development and quality assurance processes, testing to rigorous coverage criteria, critical design reviews (preferably by independent, competent experts), tool-supported static analysis and formal verification. In the case of commercial products, the existence of extensive, credible and positive experience in operation in conditions comparable to, or more severe than, the intended use may also provide additional confidence in a low level of residual faults.

Specification Faults

As implied by the name, specification faults are made at the very beginning of the digital system lifecycle and lie in the functional requirements specification. Experience in the operation of highly reliable digital systems in various industry sectors has shown that specification faults are an important, and sometimes a dominant, source of digital failure (see [35] and [37]).

Specification faults may be classified according to two orthogonal criteria. The first criterion is related to their human causes, the second criterion is related to their type. Table 5-1 presents an overview of this classification.

Table 5-1
Classification of Specification Faults.

Type \ Human Causes	Inadequate Expression	Insufficient Understanding
Incompleteness	X	X
Incorrectness	X	X
Ambiguity	X	--

Human Causes

Two main human causes may be identified at the root of most specification errors: **inadequate expression**, and **insufficient understanding**.

In the case of insufficient understanding, the mistakes are already in the specifiers' minds, as they do not have a complete or correct view of all the situations the I&C system will be facing, and / or of the expected behavior in each of these situations. The resulting specification faults are the most difficult to deal with, and their detection often necessitates the intervention of independent, competent reviewers or testers, with a fresh mind state.

The situations the digital I&C system may face arise from the states of the items that constitute its environment. These are in particular the plant physical and operational processes, the systems and equipment directly connected to the digital I&C system, and the digital I&C system itself. The states include functional states (e.g., commissioning, starting up states, nominal states, shutting down and shut down states, modification of parameters, on-line testing, etc.) and dysfunctional states (i.e., postulated failure modes, or accidental / incidental conditions). Insufficient understanding often results from a failure to systematically identify, characterize and address these items, their states and the resulting combinations, possibly due to an incomplete set of design basis documents.

Insufficient understanding may also occur when using pre-existing programmable or highly configurable I&C platforms, the behavior of which in the different situations that the digital system may face is not fully understood.

In the particular case of I&C upgrades, insufficient understanding may also result from a lack of appropriate documentation regarding the existing system, from inaccurate reverse-engineering, and / or from an inadequate assessment of the impacts of the changes imposed by the new system on installation, commissioning, operation and maintenance.

In the case of inadequate expression, the specifiers fully understand what the I&C system needs to do, but do not translate this understanding appropriately in the functional requirements specification. (This is a recurrent issue in digital systems engineering and does not affect only functional requirements specification: in all likelihood, the majority of software faults throughout all industrial sectors also result from inadequate expression.) Fortunately, these faults are usually less difficult to avoid and eliminate.

Types of Specification Faults

The most severe specification faults (i.e., those that could directly lead to a system failure) may be classified into three main types: **incompleteness**, **incorrectness** and **ambiguity**.

In the case of incompleteness, some aspects of the behavior required of the digital system are left unspecified, and the specification fails to cover all the different situations that the digital system may face. Incompleteness may result either from insufficient understanding or from inadequate expression, of from both.

> Incompleteness due to inadequate expression can sometimes be detected without a detailed understanding of the functional objectives of the I&C system. For example, there could be a suspicion of incompleteness when the system requirements specification does not state the expected system outputs for all possible combinations of input values (not just the expected values).

In the case of incorrectness, one or several specified requirements are not appropriate for some or all of the situations that the digital system may face. Incorrectness may result from inadequate consideration of existing process or plant systems studies, or from inadequate consideration of all the situations that the digital system may face. Incorrectness may result from insufficient understanding, from inadequate expression, or from both.

In the case of *ambiguity*, different readers could understand the same specified requirement and / or its conformance criteria differently. Ambiguity could arise due to the use of undefined or unclearly defined terms and expressions, to requirements for which there are no well-defined pass/fail criteria, or to the use of semi-formal languages that have no well-defined syntax and / or semantics. Ambiguity is usually considered a type of inadequate expression.

> The functional requirements specification may also be:
>
> - Overly ambitious, leading to higher complexity than strictly required.
> - Difficult to understand by those who need to use it, which could lead to misinterpretation.
> - Inconsistent, e.g., when wholly different behaviors are required of the I&C system in different but similar conditions, without adequate justification.
>
> These deficiencies usually have a less direct consequence on system digital failures. Also, avoidance and detection measures against them are often the same as those suggested for incompleteness, incorrectness and ambiguity. Thus, they are not addressed further in this report.

Avoidance / Detection Measures Against Specification Faults

This subsection briefly describes and comments each of the measures suggested either to avoid or detect the various specification faults. Table 5-2 provides an overview of the association between measures and specification fault categories.

Choice of specification languages. Even though the requirements specification of most I&C systems includes parts that are written in natural languages such as English, such languages are usually considered too imprecise and too ambiguous to be used as the only specification language. The current state of the practice is to also use one, or a combination of, *specification languages*. There is a wide variety of specification languages: semi-formal or formal, textual or graphical, general-purpose or application-specific. Most individual languages have a too narrow scope to be used to specify all necessary requirements applicable to the I&C system. Instead, most system developers use a combination of languages, each of them appropriate for specific aspects of the system: functional aspects, time performance aspects, human interface aspects, etc. For example, in the nuclear industry, graphical function block languages are often used to specify control, ESFAS or reactor protection I&C functions. When choosing such languages, notwithstanding other desirable properties beyond the scope of this report, one should look for features that might help avoid specification faults due to inadequate expression.

Typical desirable features would include:

- Clearly (possibly formally) defined and well-documented syntax and semantics, in order to minimize ambiguities.
- Mastery (by appropriate training and / or previous experience) by those who will write the specification, but also by those who will review it and / or use it as a reference for their own work.
- 'Application-friendliness', i.e., the language should facilitate the expression of requirements typical of the application domain of the I&C systems under consideration.
- Support for traceability of requirements.
- Provide features that help cope with complexity (i.e., traceable, hierarchical decomposition).
- Interoperability, when multiple languages are used.
- Features that help avoid forgetting to address situations that the I&C system could encounter.
- Suitable tool support, both for developing the system requirements specification and for verifying them.

Use of specification rules. Specification rules could provide significant help in avoiding specification faults caused by inadequate expression. They may serve many purposes, such as:

- Addressing complexity, either by setting complexity limits or by suggesting ways to help cope with it.
- Avoidance of undesirable features of the specification languages used.
- Standardization of terms (glossary), layouts, naming conventions, tables of contents, files organization, etc.
- Listing issues to be systematically considered when developing the system requirements specification.

Such rules are often project or organization-specific, and should take into consideration the types of applications considered, the chosen specification languages and tools, and the particular customs and habits of those concerned.

Systematic collection of relevant input documents. This measure is an essential one for the avoidance of specification faults due to inadequate understanding. The input documents to I&C system requirements specification would typically provide all relevant input information regarding:

- The plant physical and operational processes of relevance to the I&C system.
- The systems and equipment connected to the I&C system, including those that are connected intermittently (e.g., engineering workstations, system monitoring equipment, system test equipment).
- The I&C system itself.

It is important that the input documentation provide a clear description of the different possible states of these three items, as particular states might lead to particular interface conditions and require particular I&C system behaviors. This includes normal states (e.g., commissioning, starting up, nominal states, testing, maintenance in operation, operator requests, shutting down) and possible failed states.

It is also important that the right version of each essential document is used.

Reverse engineering (upgrade projects). When upgrading or replacing an existing legacy system, it is often the case that the existing documentation does not provide all necessary information. It is then necessary to proceed to some form of reverse engineering in order to reconstitute the missing elements.

Back-to-back testing (upgrade projects). When upgrading / replacing an existing legacy system by a functionally equivalent system, back-to-back testing (where the new and existing systems are fed with the same inputs, and the results of the new system are compared with those of the existing one) could be a very useful way to verify that the new system indeed behaves exactly as the existing one.

Traceability to / from input documents. Traceability is a well-known and widely used approach to ensure and to verify that all relevant inputs are fully and correctly taken into account in the I&C system.

Critical review. Critical review is also a well-known and widely used approach to verify that an item (here the system requirements specification) has all the desired properties (in particular completeness, correctness and clarity). The review should involve experts of various competencies so as to benefit from a variety of standpoints. In particular, when the functional requirements specification is developed by plant system experts, it is worthwhile considering including digital system and / or software experts on the review team, as such experts are often well aware of, well-trained on, and used to addressing completeness and ambiguity issues.

Independent critical review. Independent critical review is a more thorough form of critical review, where the reviewers form their judgment not through discussion with the authors of the system requirements specification, but by analyzing themselves the input documents, and going independently through a complete or nearly complete analysis process.

Simulation of system requirements specification. It is a common practice nowadays to use simulation to verify the part of system requirements specification written in formal or semi-formal specification language. (Simulators are often integral parts of I&C platforms.) Simulation coverage criteria can be used to enhance the likelihood of revealing specification faults.

Co-simulation with plant process simulator. Co-simulation of I&C specification and plant processes can be used in order to enhance the verification capability of simulation and to improve the likelihood of revealing specification faults due to insufficient understanding.

Systematic identification of intrinsic specification faults. Intrinsic faults are faults that can be identified without a full understanding of the functions of the I&C system.

Examples are:

- Contradiction, whereby different, incompatible system behaviors are required for the same situations.
- Intrinsic incompleteness, for example when the system behavior under some possible combination of inputs or other conditions is not specified.
- Non-determinism. An example is provided in Figure 5-1.

Tools are increasingly available to systematically detect such specification faults.

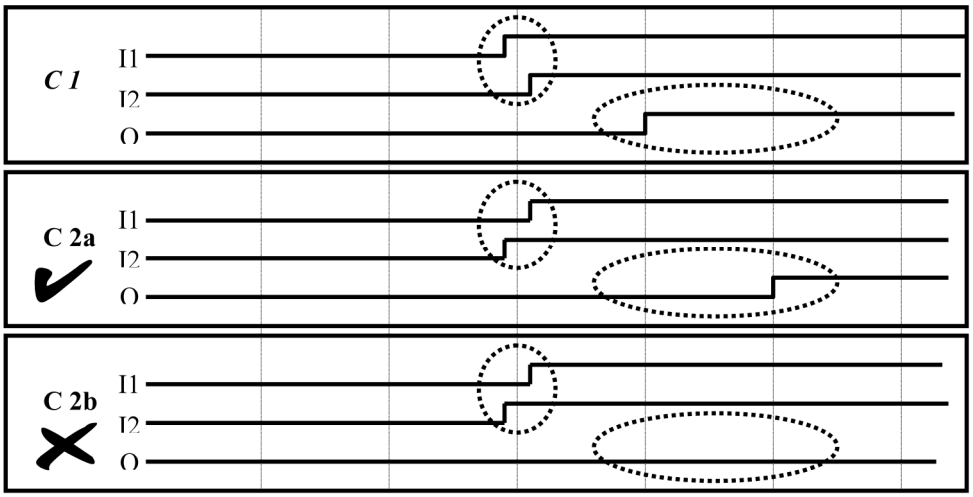


Figure 5-1
Non-deterministic functional specification.
The function has two Boolean inputs, and one Boolean output that is updated periodically. A specification requiring behaviors shown in the two top cases (C1 and C2a) would be acceptably deterministic: the output timings are different, but within an acceptable margin. A specification requiring behaviors shown in the top and bottom cases (C1 and C2b) would be unacceptably non-deterministic, as a small, uncontrollable difference in inputs timings leads to profoundly different outputs.

Table 5-2
Avoidance and Detection Measures Against Specification Faults.

	Inadequate Expression		Insufficient Understanding	
	Avoidance	**Detection**	**Avoidance**	**Detection**
Incompleteness	Choice of specification languages Use of specification rules	Critical reviews	Upgrade projects: reverse engineering Systematic collection of relevant input documents Use of specification rules	Independent critical reviews
Incorrectness	Choice of specification languages Use of specification rules	Critical reviews Simulation of functional requirements specification	Traceability to / from input documents	Independent critical reviews Independent Simulation / Testing of functional requirements specification Co-simulation together with plant process simulator
Ambiguity	Choice of specification languages Use of specification rules (glossary)	Critical reviews	--	--

Design Faults

Design faults in digital systems can take an extremely wide variety of forms, and it is difficult to provide a single set of recommendations that would be applicable to all cases. Considering the current practices of the nuclear industry regarding the development of digital I&C systems, this report organizes its recommendations according to the following classification:

- I&C platforms for safety applications.
- I&C platforms for safety-related applications or for applications critical to plant performance.
- Smart devices.
- Application-specific software.
- Application-specific FPGA designs.
- I&C System architectures.

Also, additional guidance is provided regarding the use of operational experience

I&C Platforms for Safety Applications

Here the term *safety application* designates those applications that have the highest safety classification level. In the US, this would designate 1E classified applications. According to IEC 61226, this would designate Category A applications.

As most digital I&C systems development projects are likely to rely on a pre-existing, commercially available I&C platform, the main focus is to select a platform that offers the highest level of assurance of dependability. One possible and commonly used approach is to assess the platform (or select a platform that has been assessed) against the recommendations of EPRI TR 107330 *Generic Requirements Specification for Qualifying a Commercially Available PLC for Safety-Related Applications in Nuclear Power Plants*. A possible alternative and also commonly used approach is to assess the platform (or select a platform that has been assessed) against the recommendations of the IEC family of nuclear standards applicable to digital systems implementing Category A functions (i.e., IEC 61513 for system aspects, IEC 60880 for software aspects, IEC 61500 for data communication aspects, IEC 62340 for CCF aspects).

Also, as formal verification tools that can detect intrinsic software faults (e.g., software code that could lead to divisions by zero, out-of-bounds array indexes, use of non-initialized variables, integer overflows or underflows, ...) are increasingly available and efficient, I&C system designers (or I&C platform vendors) might in addition consider their use for increased assurance, as such tools might help detect faults that are difficult or unlikely to be revealed by more classical means such as testing.

I&C Platforms for Safety-Related Applications or for Applications Critical to Plant Performance

Here the term *safety-related application* designates those applications that have a lower safety classification level than *safety applications*. According to IEC 61226, this would designate Category B or C applications.

Here again, the main focus is to select a platform that offers a high level of assurance of dependability. As for I&C platforms for safety applications, different routes may be considered:

- Assessment against the recommendations of the IEC family of nuclear standards applicable to digital systems implementing Category B or C functions (i.e., IEC 61513 for system aspects, IEC 62138 for software aspects).
- Assessment against the recommendations of the IEC family of functional safety standard 61508. This family of standards is not specific to the nuclear industry, and is widely used in the process and oil industries. As most of the I&C platforms considered for safety-related or non-safety-classified applications are also used in these industries, a number of them have already been assessed and certified against IEC 61508.

IEC 61508 recommends the application of a safety lifecycle for the development of digital systems and devices important to safety. It also identifies four safety integrity levels (SILs), each with associated recommendations, and associated reliability levels, as indicated in Table 5-3. The reliability levels are expressed in terms of probability of failure on demand (PFD) for systems and devices that operate only on demand and for which the demand rate is low (typically one demand per year or less), and in terms of probability of dangerous failure per hour for systems and devices that operate continuously or that operate only on demand and for which the demand rate is high (typically more than one demand per year).

Table 5-3
IEC 61508 – Failure Rates According to SIL

SIL	Low Demand Mode of Operation (PFD)	High Demand or Continuous Mode of Operation (Probability of Dangerous Failure per Hour)
1	$10^{-2} \leq PFD < 10^{-1}$	$10^{-6} \leq DF/h < 10^{-5}$
2	$10^{-3} \leq PFD < 10^{-2}$	$10^{-7} \leq DF/h < 10^{-6}$
3	$10^{-4} \leq PFD < 10^{-3}$	$10^{-8} \leq DF/h < 10^{-7}$
4	$10^{-5} \leq PFD < 10^{-4}$	$10^{-9} \leq DF/h < 10^{-8}$

Smart Devices

Smart devices are dedicated devices of limited, specific functionality, such as sensors, valve positioners, power meters, soft-starters, inverters, dedicated display units, uninterruptible power supplies (UPS) and dedicated communications interfaces that contain or may contain microprocessor-based or ASIC/FPGA-based controllers. As analog (i.e., non-digital) counterpart devices are progressively disappearing from the market, and as most smart devices are developed for the needs of the general industry and not specifically to the needs and requirements of the nuclear industry, specific approaches are often needed to ensure that those devices selected to perform functions important to plant safety and / or efficiency provide suitable assurance of a well conceived and executed design .

One possible approach is to consider devices that have already been assessed and certified against IEC 61508, as this would give a certain assurance of well-defined and reasonably rigorous development process.

Also, EPRI Technical Report TR 1011710 *Handbook for Evaluating Critical Digital Equipment and Systems* provides practical yet efficient guidelines for the conduct of critical design reviews for the digital aspects of smart devices.

Application-Specific Software

Application-specific software is the part of the I&C system software that is developed specifically for the purposes of a given I&C project, most often using the engineering software tools provided with the selected I&C platform. In addition to rigorous development processes and quality assurance (including rigorous configuration management) inherent to any engineering project, various measures may be taken to limit the potential for application software design faults. In particular:

- **Automated code generation** (a fault avoidance measure), whereby application software code is automatically generated by software tools from the functional requirements specification or from a high level representation close to the functional requirements specification. This can significantly limit the potential for application developers to introduce coding errors in the application-specific software.
- **Cautious use of trusted tools** (fault avoidance measure), in particular of code generation tools (including those used to generate the executable binary code). This type of measure is particularly important, considering the fact that such tools are usually complex and imperfect, and could inadvertently introduce errors in application software. One can for example select tools for which sufficient successful operating history is available, or the development of which was performed under adequate software quality assurance. Also, rules may be applied for their use so that atypical and / or less proven usages are avoided.
- **Tool diversity** (fault detection measure), whereby the outputs of a given generation tool are verified using other independent tools (e.g., test and simulation tools, formal verification tools) based on different principles.

- **Coding rules** (fault avoidance measure) may also be used to set limits on complexity of the application-specific software (thus also facilitating its verification), preclude the use of un-trusted features of the languages and tools used, facilitate understanding by application software developers, application code reviewers and application engineers who might need to perform modifications in the future.
- **Simulation & testing** (fault detection measure) is the most commonly used approach to detect faults in application software. Simulation may be performed on high-level representations of the application software to ensure that the logic specified in these high level representations is correct. Once the application code is generated, it can then be gradually integrated and tested together with the rest of the I&C systems. Rigorous planning and application of test coverage criteria should be used to increase the effectiveness of simulation and testing.
- **Critical design reviews and code inspections** (fault detection measure) are also often used in complement to simulation and testing, to increase further the likelihood of detecting errors. For safety applications and for applications that are the most critical to plant performance, reviews and inspections by **independent** reviewers should be considered, and may required by regulation.
- **Formal verification** (fault detection measure), whereby the generated application code is shown to be consistent with formally specified requirements, or is shown to be free from particular types of faults, using rigorous, mathematics-based approaches. Such techniques are rarely used currently, but are becoming increasingly available and feasible.

Application-Specific FPGA Design

Field programmable gate arrays (FPGA) and similar technologies are now an integral part of the solutions portfolio at the disposal of I&C engineers. Though FPGA programming has many similarities with software programming, there are a number of important differences. EPRI 1019181, *Guidelines on the Use of Field Programmable Gate Arrays (FPGAs) in Nuclear Power Plants I&C Systems* provides an introduction to the technology, and also some guidelines in the design of FPGA-based solutions.

More detailed design guidelines are provided by the upcoming IEC standard 62566 *Nuclear Power Plants - Instrumentation and Control Important to Safety - Development of HDL-Programmed Integrated Circuits for Systems Performing Category A functions*.

Digital System Architecture

Modern digital I&C systems are typically composed of multiple, interconnected, functional units, with implementation of the various functions required of the system distributed among the functional units. Various design errors may affect the system architecture, such as:

- **Inappropriate aggregation of functions**, where requirements regarding the independence of particular functions of the I&C system are not fully satisfied.
- **Inadequate allocation of resources** such as computing power, memory, data communication bandwidth, where some worst case situations (such as data communication storms) could not be handled satisfactorily.
- **Inadequate distribution of a given function**, where the processing performed by the different functional units over which the function is distributed do not form a consistent whole to fully and correctly implement the function.

System engineering tools provided by the I&C platform often may be used to address all or part of these issues and avoid related errors. For issues not well protected by these tools, specific measures may be taken, such as design rules (as a fault avoidance measure) or verification, inspection and testing (as fault detection measures).

Simplicity - Determinism

A key digital systems characteristic for avoiding and / or eliminating digital faults and unanticipated conditions is **simplicity**. Indeed, it is generally accepted that overly complex items (systems or components) cannot be made reliable to the level required of most safety-related systems and some systems critical to plant performance.

Simplicity can be viewed from two main standpoints:

- **Functional simplicity** views an item as a black box, and implies a low number of functions, functional internal states, parameters, inputs and outputs, interfaces and interactions.
- **Design simplicity** views the item as a white box, and implies either or both of:
  - **Structural simplicity**, i.e. simplicity of code and architecture (e.g., modular design, minimal branching, etc.). Various metrics for structural simplicity have been proposed and are supported by commercially available complexity measurement tools.
  - **Behavioral simplicity**, i.e., simplicity of the internal functioning of the item, and low number of factors that can influence its behavior (e.g., insensitivity to plant operating conditions and transients). Behavioral simplicity of is often associated with **determinism**.

Simplicity of a digital system, subsystem or calculational module can be intrinsic. It may also be imposed by constraining its conditions of use based on the identification and characterization of its influence factors. For example, most of the adjustable parameters may be fixed once and for all; some functions, components or capabilities may be discarded and not used. Also, the variation of inputs may be constrained before they are used in calculations.

Experience in Operation

Experience in operation may be used to increase the level of assurance that an item is unlikely to suffer from digital failure. However, there are three necessary conditions:

- Sufficiently large **volume of positive experience**. The volume necessary for justifying acceptably low levels of residual digital faults depends on several factors, in particular on the level of assurance that is sought, on the functional complexity of the item, and on the applicability and credibility of experience information.
- **Applicability of experience** to the contemplated operating conditions. In particular:
  - Does all the experience apply to the version of the item under consideration? If not, what justifies the inclusion of the experience of other versions?
  - Are the conditions of use during the experience known? If so, are they sufficiently close to the intended use? If not, how likely are they to be sufficiently close to the intended use?
  - How likely are failures in the field to be reported? Have the failure reports been analyzed appropriately so as to identify with reasonable confidence the causes of failures?
- **Credibility of the experience**: Are we sure of the claims made regarding the experience collected (volume of the experience, reporting and analysis of failures, conditions of use)? Do we have appropriate evidence?

Some useful hints may be taken into consideration. For example, one can examine the history and the frequency of modifications, and the criticality of past or current applications (for highly critical applications, failures are more likely to be reported and analyzed). One can also assess the vendor's program for encouraging users to report failures; for collecting, managing and analyzing failure reports; and for communicating the resulting information to other users.

Experience in operation is often complemented with other measures such as testing in intended conditions of use, restricted conditions of use, and "encapsulation" (allowing verification of inputs and outputs, and monitoring of correct functioning).

Activating Conditions

Digital systems have a basically deterministic behavior in that they will react the same way every time they encounter the same conditions and input sequences. However, this can be obscured by complexity (when one can no longer know in sufficient detail the current state of the system and predict what it will do in the next steps) and by the influence of too-numerous or hard-to-predict external influence factors (e.g., interrupts). However, for the highly reliable digital systems typically used to implement safety-related or production-critical functions, designers usually take measures to severely restrict complexity. In particular, the measures taken to reduce behavioral complexity can often have very beneficial effects regarding the minimization of the potential for activating conditions.

Identified and Limited Influence Factors

An influence factor is anything that can affect the functioning of a digital system or component. Influence factors may be divided into two groups: factors acting **permanently**, and factors acting only **occasionally**. Permanent influence factors typically include inputs from the plant process, external and internal interrupts, configuration parameters, system internal states, time elapsed since initialization, date and time of the day, availability of necessary resources, synchronization and interactions with external equipment. Occasional influence factors typically include operator requests and actions (for example for testing and maintenance, or for changing parameters such as setpoints), hardware failures, exceptions (e.g., loss of power supply), special dates (e.g., January 1st, 2000), data communication storms, and seldom encountered plant conditions.

Designers of highly reliable digital systems usually aim at eliminating or restricting these factors as much as reasonably feasible in order to reduce the likelihood of encountering "surprising" (untested and / or unanticipated) combinations of factors that could activate and reveal a residual digital fault. For systems or components with well-documented, simple, stable and deterministic behavior, it is often feasible to systematically identify and characterize the residual influence factors. For example:

- Input values are checked before any further processing, in order to limit the potential for "surprising" values, combinations and / or sequences.
- Interactions with external devices are designed so that the digital system or component always has the initiative (polling its inputs at its own will, instead of receiving interrupts at uncontrollable rates and times) and so that no synchronization is required from its standpoint (whatever the state of the opposing party, it just "reads" the information).
- Date and time-of-the-day are neither computed nor used, and all counters are verified to be protected against overflow.
- Elapsed time is maintained only over short periods of time.
- Few (if any) configuration parameters are changed during operation.
- Operator requests and actions are performed off-line, whenever possible, with appropriate verification of correct operation before reinsertion into normal operation.
- Use of internal states and memory (i.e., what the digital system or component remembers from the past) is limited as much as possible.

Simple, Stable, Deterministic Behavior

One often used designed-in defensive measure to reduce the likelihood of encountering an activating condition is to have the digital system or component follow a deterministic, repetitive routine in very stable and restricted conditions, so that tests and verification can be performed to reach a high level of behavioral coverage, and that untested and / or never-encountered conditions during normal operation (under permanent influence factors) are extremely unlikely (see Figures 5-2 and 5-3).

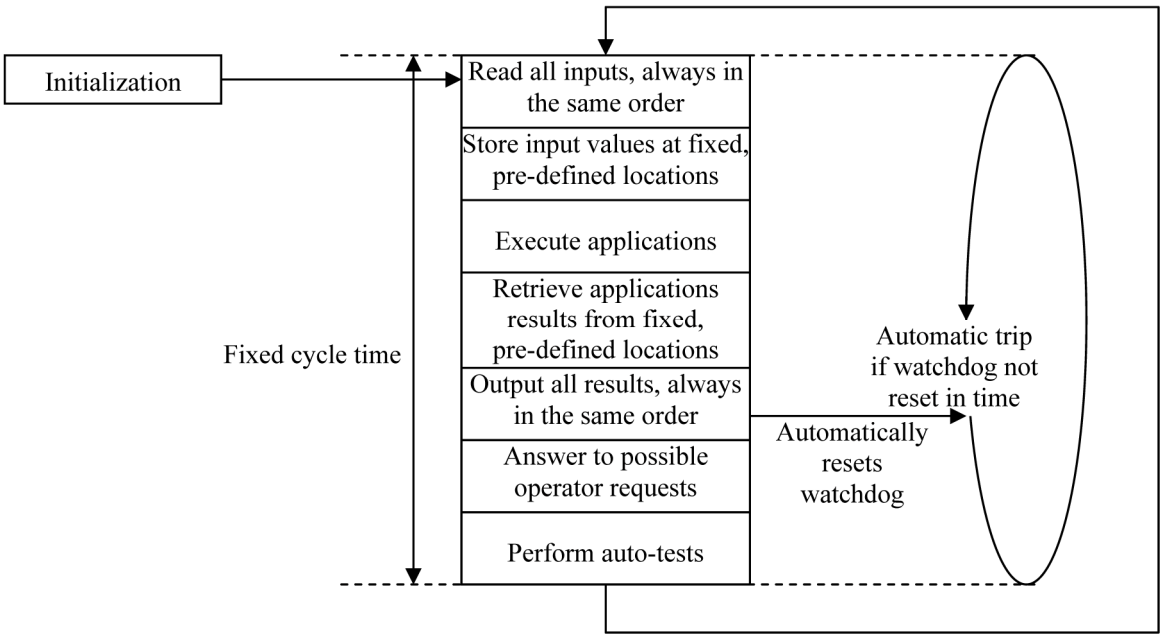


Figure 5-2
Example of Fixed, Repetitive Behavior.

Only under the action of occasional influence factors would the system depart from its routine. Systematic identification, characterization and constraining of these factors may be used to enhance the effectiveness of tests and other forms of verification.

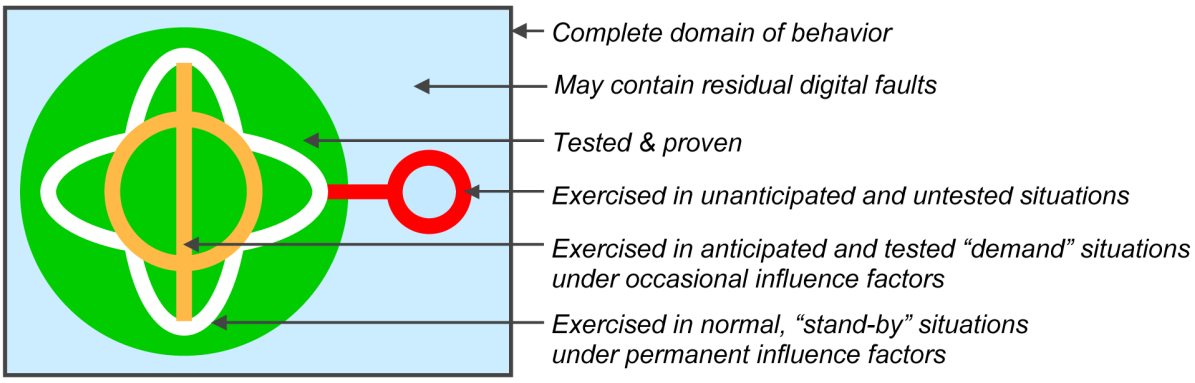


Figure 5-3
Stable, repetitive behavior in normal conditions helps focus attention on occasional situations.
Where digital failure could occur if the system is challenged by new, unanticipated conditions.

Components Not Affected by Particular Influence Factors

Even though the digital system as a whole may be influenced by particular influence factors, some of its components may be designed and used in such a way that they are not affected by these factors. For example, some operating systems are designed to be "transparent" to process conditions and not influenced by applications: whatever happens in the process, the operating system is unaffected and unaware of it; it is the application specific software and some of the elementary functions that will handle the situation (see Figure 5-4).

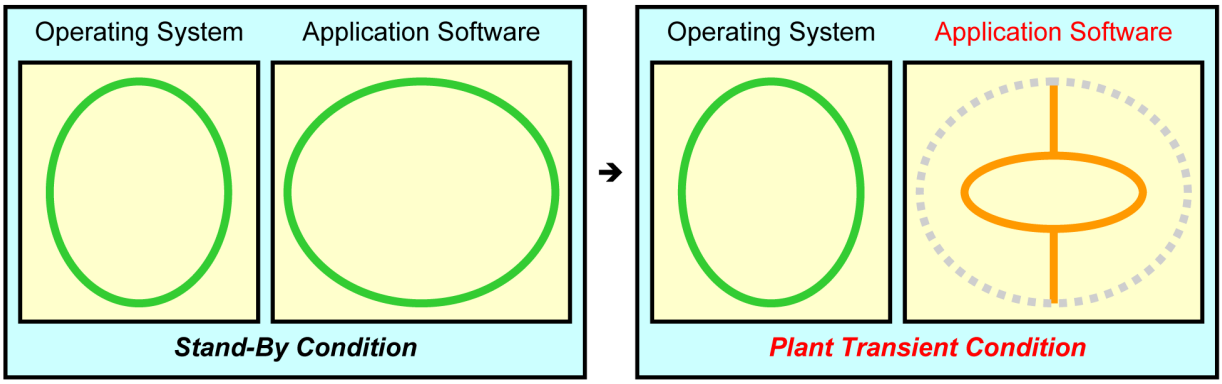


Figure 5-4
Components Operating in Stable Conditions and Not Affected by Particular Influence Factors Are Unlikely to Fail When These Factors Change.

Margins

Even with a very simple, stable and deterministic design, there are usually small variations in the actual functioning of a digital system. Designers of highly reliable systems usually have provision for sufficient margins so that these small variations remain well within acceptable ranges. For example, such margins may concern timing, use of resources and accuracy of computations.

Defense Against CCF Due to Identical or Similar Digital Faults

To prevent digital CCF that could be caused by the concurrent activation of identical or similar digital faults in multiple functional units, system designers can take various measures, which can be classified in the following categories:

- Reduction of the potential for digital failure, as discussed in the previous sub-sections.
- Diversity measures aimed at reducing the likelihood of such identical / similar faults in the functional units under consideration.
- Diversity measures aimed at reducing the likelihood that the functional units under consideration are concurrently affected by an activating condition.

Diversity of I&C Systems

US NRC NUREG/CR-7007 ORNL/TM-2009/302, *Diversity Strategies for Nuclear Power Plants and Instrumentation and Control Systems* [7] presents various diversity attributes that could be used to limit the likelihood of identical / similar faults in multiple I&C systems. These are:

- **Functional diversity**, with a combination of different purposes, functions, control logic, actuation means, underlying mechanisms, response-time scales.
- **Signal diversity** (use of different sensors for the same parameters, use of different parameters sensed by same or different physical effects).
- **Design diversity** (use of different technologies, different design approaches and / or different architectures).
- **Equipment diversity** (from different vendors and / or with different designs).
- **Logic** (software) diversity with a combination of different algorithms, logic, program architectures, timing or order of execution, runtime environments and functional representations.
- **Life-cycle diversity** (reliance on different design organizations or teams, independent verification and validation teams).

The document also discusses three main diversity strategies: use of fundamentally diverse technologies, use of distinctly different technologies, and use of architectural variations within a technology.

The diversity approach suggested by NUREG/CR-7007 [7] provides a solution that is mostly applicable to the first of the CCF contexts mentioned in Section 3, i.e., for I&C systems in different lines of defense, where it can be applied with minimal adverse impact, because the affected systems are already separate, independent and functionally diverse. Indeed, for this context, I&C systems diversity is the most commonly used approach.

It is to be noted that a number of studies of high integrity software-based systems in nuclear plants and in other industries with high safety concerns (EPRI 1016731, Leveson and others [35, 37]) point to defects in functional specifications as a prominent, if not dominant source of digital failures. This suggests that functional and signal diversity (intentional use of different functional requirements) will be the most important types of diversity to employ in protecting against CCF. Further, in light of the adverse impacts of implementing diversity – increased complexity of the overall I&C design, operation, maintenance, etc., it may well be that the other diversity types suggested by NUREG/CR-7007 - design, equipment, logic and life-cycle diversity are best used when complementing functional and signal diversity.

Diversity of Operating Conditions

The solutions suggested by NUREG/CR-7007 apply less well to the other two CCF contexts presented in section 3:

- In the case of subsystems implementing diverse I&C functions, functional and signal diversity may be implemented to provide a significant level of defense against digital CCF. However, as these subsystems are based on the same I&C platform, the CCF protection offered by the other forms of diversity described in NUREG/CR-7007 will probably be limited.
- In the case of redundancies of the same I&C system, even though some minor diversity features could be implemented, that would likely remain limited: if not, there would not be a single redundant system, but multiple non-redundant systems.

In addition, complete diversity comes with a cost, in terms of increased functional, design and operation and maintenance complexity. It can also introduce difficulties in deciding which course of action should be taken when two diverse systems disagree. As noted in NUREG/CR-6463 (*Review Guidelines on Software Languages for Use in Nuclear Power Plant Safety Systems,* June 1996):

> "Uncontrolled or unspecified external diversity can lead to a proliferation of interfaces which impact safety due to difficult maintenance, testing, verification, and validation."

Therefore, careful assessment of CCF susceptibility is critical in keeping the defense strategy against digital CCF simple, practical and effective. Fortunately, other types of protective measures against digital CCF may be used, that rely on the avoidance of concurrent fault activating conditions. To this end, an analysis of each residual occasional influence factor can be made. Some are inherently random (e.g., hardware failures) and are unlikely to affect multiple subsystems or redundant divisions concurrently and lead to a digital CCF. Others are associated with operation and maintenance procedures, which could be designed so that only one subsystem or division is affected at a time (what could be called "operational diversity" measures). For example:

- Operator requests could be performed on only one subsystem or division at a time, off-line whenever possible, with a sufficient time interval before another subsystem or division is affected.
- The different subsystems or divisions could be started at different times, so that postulated failures related to elapsed time would not occur concurrently.

Particular attention should be given to those residual influence factors that cannot be protected in such manner. For example, plant conditions are likely to concurrently affect multiple functional units, and particular effort should be focused on those components that are not transparent to plant conditions (like application-specific code).

In the case of master-slave architectures, a possible additional defense against CCF due to identical or similar faults in the master and the slave is to have designs such that the slave does not do the same things as the master. This is a form of functional diversity.

> For example, the slave could be content to monitor the master activity, and acquire the minimum set of information (without processing it) that would allow it to take over when needed.

Defense Against CCF Due to Failure Propagation

This topic is the subject of the NRC interim staff guidance on data communication (Reference 17) and is not addressed in detail here. The following is a brief reminder of defensive measures that could help prevent digital CCF by failure propagation.

Communication Architecture

Measures could include:

- Stable or constrained communication loads and patterns, with sufficient margins, and enforced by communication equipment. *Stable communication load* entails that for a given communication link, the volume of data sent through the link per unit of time is constant, i.e., that there are no 'peaks' and no possibility of overloading the communication link. *Stable communication pattern* entails that the communication is cyclic, i.e., that there is a fixed, pre-defined cycle time, and that at each cycle, the same senders send the same fixed-length messages (possibly with varying contents) to the same receivers, following the same order (see Figure 5-5). *Constrained communication* entails that worst-case communication situations (e.g., data communication storms) are determined (possibly with enforcement by *communication equipment*; communication equipment are those devices that constitute the communication link, such as physical media, communication interface boards, switches, routers and gateways) and shown to be within the limits acceptable by the communication links and architecture and by the *communicating stations* (i.e., the various equipment connected to the communication link that send or receive information through the communication link). This type of measure serves different purposes. In particular, stable and cyclic operating conditions help minimize the occurrence of unexpected situations that could activate a postulated dormant design fault in the communication link. Also, stable and cyclic communication help provide stable conditions to the communicating stations, so that they also are protected against unexpected situations.

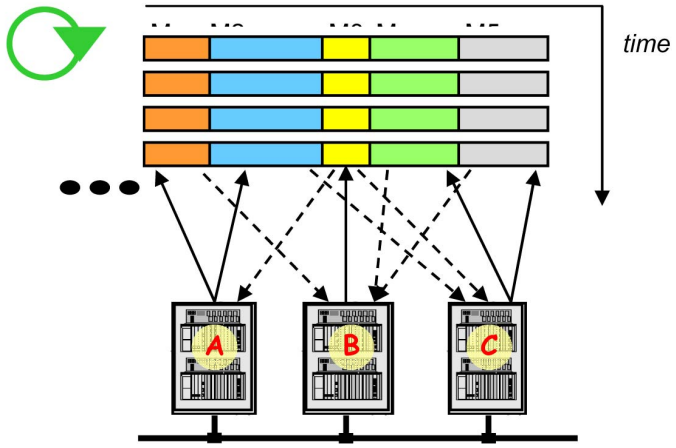


Figure 5-5
Stable Communication Load and Pattern.
At Each Cycle, in the Same Order, the Same Senders Send Messages of Pre-Determined Lengths to the Same Receivers.

- Protection of *communication equipment* from abnormal behavior of *communicating stations*. The objective of this measure is to protect the communication link, and to a lesser extent, the other communicating stations, from being 'forcefully invaded' by the abnormal behavior of the failed communicating station. A typical measure is to allow data exchanges between a communicating station and the communication link interface only through non-synchronizing double entry registers. Each partner in the interface sees the other partner only through what it reads (at its own rhythm) from the registers, so that misbehaviors of the partner can only affect the readings. From the communication link side, this means that if the link does not detect the failed state of the station, it will transmit incorrect data to the receiving stations, but it will not be itself affected by the station failure.

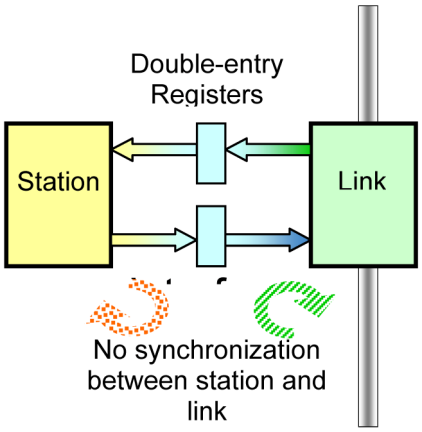


Figure 5-6
Possible Interface Between Communication Stations and Communication Links.

- One-way communication when appropriate, enforced by communication equipment.
- Predefined communication network configurations, enforced by communication equipment. In typical office communication networks, network configurations are highly 'dynamic', with new stations and links added, and stations and links removed, 'on-line'. In communication networks important to safety or plant performance, it is better to avoid this and opt for a much stricter approach that helps minimize unexpected and untested situations. The only variability that would remain would be the loss (and subsequent reappearance) of stations and links.
- Communication networks separated into multiple, independent 'islands' to limit possible failure propagation within an island.
- Redundant communication links.

Communication Equipment

Measures could include:

- Reliable communication equipment (see discussion on design faults).
- Transparency to plant conditions (see discussion on activating conditions).
- Careful planning of human interventions to limit risks of digital CCF (see discussion on diversity of operating conditions).

Communicating Stations

Measures could include:

- Limited interface with communication equipment ensuring that communication equipment failures cannot cause the station to fail or be overloaded (see discussion on communication architecture).
- Tolerance to communication failures (which can originate either from communication equipment or other communicating stations). In particular, each communicating station should have clearly defined behavior in the case of loss of, or discrepancies in, communication.
- Plausibility checks of received data giving reasonable assurance that incorrect data is detected and discarded. Also, analysis could be performed regarding the acceptability of the behavior of stations that receive plausible but incorrect data.

Defense Against CCF Due to Single Point Digital Vulnerability

Several approaches may be used to prevent digital CCF that could be caused by the digital failure of single components shared by multiple functional units.

The first approach is to avoid altogether such shared components. However, this is not always possible. For example, many redundant systems rely on voting, and a failure of the voting subsystem could defeat redundancy.

When a shared component is present, two main approaches may be applied:

- Ensuring that the component is not susceptible to digital failure (see discussion on design faults and activating conditions).
- Ensuring that no postulated digital failure mode of the shared component will prevent the correct performance of a safety function.

Defense Against CCF Due to Shared Susceptibility to Global Stressing Conditions

The first step of the general approach to prevent digital CCF caused by a shared susceptibility to global stressing events consists in a systematic identification of those global stressing conditions that could adversely and concurrently affect multiple digital equipment and systems. These typically include special dates and times; data communication storms; cyber aggressions; and extreme ambient, electromagnetic and seismic conditions that could fail aged electronic components.

For a given stressing condition, different complementary approaches may then be used:

- Avoidance: Designed-in measures may be taken to ensure that particular functional units are not susceptible to the condition. Examples include:
  - Special dates and times: some functional units may be designed so that they do not compute and manage dates and times.
  - Data communication storms: data communication may be designed so that the data communication load remains constant.
- Fault removal by focused verification: For units that could be affected by the condition, focused verification may be performed to ensure that no residual digital fault could be activated by the condition.
- Tolerance: Designed-in measures may be taken to ensure that should one or several functional units fail due to the condition, they fail in specified safe failure modes.

6
CONCLUSION

Diversity is not the only means of protection against digital CCF. Other measures (collectively called “defensive measures”) may be used to limit or preclude digital CCF, or can reduce the scope of CCF protection needed from diversity measures. However, to decide whether a given digital system or digital architecture is, or is not, susceptible to digital CCF, or decide whether defensive measures or diversity offers the most appropriate CCF protection, sound engineering judgment and supporting data are needed.

Also, in order to be credible, claims regarding defensive measures should be supported by sound argument and appropriate evidence. This will often require detailed understanding of the design of the digital system and of its environment.

7
GLOSSARY

Definitions

This section provides definitions for key terms as they are used in this guideline. When the definition is taken directly from another document, the source is noted in brackets [ ].

Activating condition. A specific condition affecting a digital system that activates and gives life to a dormant digital fault and causes a digital failure.

Application software. Part of the software that performs the tasks related to the process being controlled rather than to the functioning of the system [adapted from IEC 61513].

Common-cause failures. Concurrent failures (that is, failures which occur over a time interval during which it is not plausible that the failures would be corrected) of equipment or systems that occur as a consequence of the same cause. The term is usually used with reference to redundant equipment or systems or to uses of identical equipment in redundant systems. CCFs can occur due to design, operational, environmental, or human factor initiators.

Communicating station. Functional unit or device connected to a communication link, and that sends messages to, and / or receives messages from, the other stations connected to the link.

Communication link. Set of equipment and media that allow two or more communicating stations to exchange messages.

Defense-in-depth. A concentric arrangement of protective barriers or means, all of which must be breached before a hazardous material or dangerous energy can adversely affect human beings or the environment. For instrumentation and control systems, the application of the defense in depth concept includes the control system; the reactor trip or scram system; the Engineered Safety Features Actuation System (ESFAS); and the monitoring and indicator system and operator actions based on existing plant procedures. The echelons may be considered to be concentrically arranged in that when the control system fails, the reactor trip system shuts down reactivity; when both the control system and the reactor trip system fail, the ESFAS continues to support the physical barriers to radiological release by cooling the fuel, thus allowing time for other measures to be taken by reactor operators to reduce reactivity. [5]

Design fault. Digital fault affecting the overall design of a digital system, its software or the programming of its FPGAs.

Digital fault. Functional requirement specification or design error resulting from the development of a digital system, and that exists in the system right from the beginning of operation.

Digital upgrade. A modification to a plant system or component which involves installation of equipment containing one or more computers (see above definition of computer). These upgrades are often made to plant instrumentation and control (I&C) systems, but the term as used in this document also applies to the replacement of mechanical or electrical equipment when the new equipment contains a computer (e.g., installation of a new heating and ventilation system which includes controls that use one or more embedded microprocessors). [3]

Diversity. Existence of two or more different ways or means of achieving a specified objective.

Failure. Termination of the ability of a functional unit to perform a required function

Failure mechanism. An event or chain of events occurring during operation and / or maintenance, and leading to the failure of the system, component or function.

Failure mode. External behavior of a system, component or function in case of a failure, the system, component or function being viewed as a black-box.

Fault. A defect that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function when subjected to a particular set of normal or abnormal operating conditions.

Functional specification. A document that specifies the functions that a system or component must perform. [IEEE 610.12.1990]

I&C architecture. Organizational structure of the I&C systems of the plant which are important to safety. [IEC 61513]

I&C platform. Set of hardware and software components that may work co-operatively in one or more defined architectures (configurations) [IEC 61513]

I&C system. System, based on electrical and/or electronic and/or programmable electronic technology, performing I&C functions as well as service and monitoring functions related to the operation of the system itself.

Intrinsic fault. Digital fault that can be recognized independently of the functional objective of the system, and without full knowledge or understanding of its functional requirements specification.

Operating system. The machine resident software that enables a computer to function. Without it, application programs could not be loaded or run

Partial failure. Failure of a component or a subsystem that does not prevent the whole system from performing an expected or required function.

Random fault. Fault appearing at a random time, which results from one or more of the possible degradation mechanisms in the hardware

Random fault or failure. Failure, occurring at a random time, which results from the activation of one or more random faults

Redundancy. The provision of alternative (identical or diverse) equipment or systems so that any one can perform the required function, regardless of the state of operation or failure of any other. [3]

Reliability. The characteristic of an item expressed by the probability that it will perform a required mission under stated conditions for a stated mission time. [IEEE-577-1991 and IEEE-352-1987]

Software. Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. This includes software that is implemented as firmware. [3]

Specification fault. Fault in the functional requirements specification of a system.

Susceptibility. Quality of being sensitive to an extraneous agent or effect. For the purposes of this document, susceptible means that digital CCFs are possible at the component, system or multiple system level. However, such failures do not necessarily affect risk significantly at the plant level. (this is in contrast to use of "vulnerability").

System. A collection of equipment that is configured and operated to serve some specific plant function(s) (e.g., provides water to the steam generators, sprays water into the containment, injects water into the primary system).

Digital Common-Cause Failure (digital CCF). A systematic common-cause failure resulting from a design fault in a digital system or component (e.g., a design error in the software or software-hardware interaction).

Vulnerability. Quality of being open to attack or damage. In this document, vulnerability is used to refer to increased risk at the plant level due to digital CCF (this is in contrast to the use of "susceptibility").

"White box". System or component whose internal contents or implementation are known. [IEEE 610.12.1990]

Abbreviations

ASIC	Application Specific Integrated Circuit
BTP	Branch Technical Position
CCF	Common-Cause Failure
CFR	Code of Federal Regulations
D3	Diversity & Defense-in-Depth
EPRI	Electric Power Research Institute
ESFAS	Engineered Safety Features Actuation System
FPGA	Field Programmable Gate Array
I&C	Instrumentation & Control
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineer
IAEA	International Atomic Energy Agency
NEI	Nuclear Energy Institute
NRC	Nuclear Regulatory Commission
NUREG	Nuclear Regulation
P_{DF}	Probability of Digital Failure
PLC	Programmable Logic Controller
QA	Quality Assurance
RTS	Reactor Trip System
SIL	Safety Integrity Level
UPS	Uninterruptible Power Supply

8
REFERENCES

1. Branch Technical Position HICB-19, “Guidance for Evaluation of Defense-in-Depth and Diversity in Digital Computer-Based Instrumentation and Control Systems”
2. Regulatory Guide 1.174, “An Approach for Using Probabilistic Risk Assessment in Risk-informed Decisions on Plant-Specific Changes to the Licensing Basis”, July 1998
3. EPRI TR-102348 Revision 1 (NEI 01-01), “Guideline on Licensing Digital Upgrades – A Revision to EPRI TR-102348 to Reflect Changes to the 10 CFR 50.59 Rule”, (March 2002)
4. NRC Regulatory Issue Summary 2002-22, “Guideline on Licensing Digital Upgrades: EPRI TR-102348 Revision 1, NEI 01-01: A Revision of EPRI TR-102348 to Reflect Changes to the 10 CFR 50.59 Rule”, (November 2002)
5. NUREG/CR-6303, “Method for Performing Diversity and Defense-in-Depth Analyses of Reactor Protection Systems”, December 1994
6. NUREG 0711 Revision 1, “Human Factors Engineering Program Review Model”
7. NUREG/CR-7007 ORNL/TM-2009/302, "Diversity Strategies for Nuclear Power Plants and Instrumentation and Control Systems"
8. Safety Requirements Memorandum, “SECY-93-087: Policy, Technical, and Licensing issues Pertaining to Evolutionary and Advanced Light-Water Reactor (ALWR) Designs”, July 1993
9. Code of Federal Regulations, Title 10, Part 100: “Reactor Site Criteria”
10. Code of Federal Regulations, Title 10, Part 50.59: “Changes, Test and Experiments”
11. Chris Garrett & George Apostolakis, “Context in the Risk Assessment of Digital Systems”, in Risk Analysis, Vol. 19, N° 1, 1999
12. ASME RA-S-2002, Standard for Probabilistic Risk Assessment for Nuclear Power Plant Applications
13. ANS N18.2-1973, “Nuclear Safety Criteria for the Design of Stationary Pressurized Water Reactor Plants”
14. Code of Federal Regulations, Title 10, Part 50.62, “Requirements for Reduction of Risk From Anticipated Transients Without Scram (ATWS) Events for Light-Water-Cooled Nuclear Power Plants”
15. NUREG-0800, “Standard Review Plan for the Review of Safety Analysis Reports for Nuclear Power Plants, Chapter 7, Instrumentation and Controls”
16. NEI 96-07 Revision 1, “Guidelines for 10 CFR 50.59 Implementation” November 2000

17. Interim Staff Guidance, DIGITAL INSTRUMENTATION AND CONTROLS, DI&C-ISG-04, Task Working Group #4: Highly-Integrated Control Rooms—Communications Issues (HICRc), , Revision 1, ML083310185, March 06, 2009 DI&C-ISG-04
18. IAEA NS-G-1.3, “Instrumentation and Control Systems Important to Safety in Nuclear Power Plants – Safety Guide”
19. IEC 61226, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - Classification of instrumentation and control functions”, 2009
20. IEC 60880, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - Software aspects for computer-based systems performing category A functions”, 2006
21. IEC 60987, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - Hardware design requirements for computer-based systems”, 2007
22. IEC 61500, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - Functional requirements for data communication", 2009
23. IEC 61508, parts 1 to 7, 'Functional safety of electrical / electronic / programmable electronic safety-related systems", 2010
24. IEC 61513, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - General requirements for systems", 2001
25. IEC 62138, “Nuclear Power Plants - Instrumentation and Control Important for Safety - Software aspects for computer-based systems performing category B or C functions”, 2004
26. IEC 62340, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - Requirements for coping with common-cause failure (CCF)", 2007
27. IEC 62566, “Nuclear Power Plants - Instrumentation and Control Systems Important for Safety - Development of HDL-programmed integrated circuits for systems implementing category A functions"
28. D.P. Blanchard & R.C. Torok, “A Risk-Informed Approach to Evaluating Digital Upgrades”, 13th Annual Joint ISA POWID / EPRI Control & Instrumentation Conference, June 2003
29. IEEE Std 603-1998, “IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations”
30. EPRI TR-106439, “Guideline on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications”, (October 1996)
31. EPRI TR-1002835, “Guideline for Performing Defense-in-Depth and Diversity Assessments for Digital I&C Upgrades”, (December 2004)
32. Generic Letter 88-20 "Independent Plant Examination for Severe Accident Vulnerabilities", USNRC, November 23, 1988
33. "Use of Probabilistic Risk Assessment Methods in Nuclear Regulatory Activities", Federal Register Vol. 60 pg. 42622, August 16, 1995
34. SECY-99-007A, “Recommendations for Reactor Oversight Process Improvements (Follow-up to SECY-99-007)”, March 1999
35. Nancy G. Leveson, Safeware, Addison Wesley, September 95

36. NUREG/CR 0492, "Fault Tree Handbook", January 1981

37. EPRI TR-1016731 "Operational Experience Insights on Common-Cause Failure in Digital Instrumentation and Control Systems", December 2008

A
APPENDIX A

Fault Avoidance Measures Against Functional Specification Faults

Table A-1 lists a set of fault avoidance measures against functional specification faults and their potential benefits. Such measures are primarily process related, particularly those that protect against functional mistakes. Although they are process focused, they often generate documentation that can be used to confirm their use after the fact.

Measures for Programmable Equipment

Table A-2 provides an example of a set of measures that would be appropriate for programmable equipment. As fault detection measures are usually well known and the object of safety standards (for example, see [18], [20], [23], [24], [25]), the table focuses mostly on fault avoidance measures, and on measures designed to minimize to potential for activating conditions.

Measures for Smart Devices with Simple, Fixed Functionality

Table A-3 lists a set of measures that are particularly appropriate for simple devices. This set is based on a list of desirable attributes for the assessment of built-in quality of commercial grade smart devices introduced in EPRI TR 106439 [4]. It includes measures to ensure that the device functionality is appropriate for its intended purpose, measures to ensure that the device has a low potential for residual digital faults, and measures to ensure that device failures can be tolerated. While the measures listed in Table 2 are generally for more complex devices, they may also be useful in simple devices.

Table A-1
Examples of Avoidance Measures Against Functional Specifications Faults

Avoidance Measures	Benefits
Functional specification focused on what is strictly necessary for safety, and for the operation of the digital system.	Avoid functional mistakes, including: • Oversight of some of the operational conditions that may face the digital system. • Incorrect characterization of anticipated operational conditions. • Incorrect characterization of interfaces and interactions. • Specification of inappropriate behavior for some operational conditions. • Failure to specify actions and operational concerns for faults and failures • Failure to extend an existing system's logic into all operating conditions
Static and rigorous determination of all the entities interacting with the digital system, and of their different states.	
Functional specification addressing all resulting operational conditions.	
Simplicity of interfaces and interactions.	
Identification and examination of the differences with the I&C system to be replaced or with similar I&C systems that have proven to be adequate.	
Functional specification languages, elementary functions and tools with clearly defined and simple syntax and semantics.	Avoid technical mistakes, e.g.,: • Incompleteness. • Ambiguousness. • Insufficient accuracy. • Oversight of possible effects of digitization. • Oversight of possible adverse side-effects. • Intrinsically unsound expressions. • Incorrect translation of results of functional studies into functional specification.
Specification methods and tools well-adapted to application domain, allowing simple functional specification.	
Specification methods and tools that can help avoid or detect incompleteness and intrinsically unsound expressions (e.g., expressions that could lead to divisions by zero).	
Functional specification process guaranteeing that relevant functional studies are taken into account correctly.	
Functional specification process providing clear guidance regarding effects of digitization.	
Systematic verification of correctness and completeness of functional specification versus plant functional and safety requirements.	Reveals and removes existing functional specification faults.
Existence of an unequivocal and easy to reach safe failure position.	Reduce the likelihood of potentially unsafe failures. Ensure rigorous treatment of potentially unsafe conditions (as opposed to abnormal behaviors that cannot result in unsafe conditions, e.g., unspecified behaviors during commissioning tests before fuel is loaded)
Boolean safety outputs with clearly identified failure modes and unsafe failure modes.	
Particular focus on plant conditions for which incorrect or incomplete system specifications could result in unsafe failures (e.g., plant transients).	
Verification of functional specification particularly focused on potentially unsafe outputs.	
Specification of the conditions that should be satisfied by inputs (pre-conditions), and of conditions that must be satisfied by outputs (post-conditions).	

Table A-2
Examples of Defensive Design Features for Programmable Equipment

Defensive Measures	Benefits
Rigorous development and modification processes.	Lower likelihood of introducing faults and higher likelihood of fault detection, resulting in fewer residual digital design faults.
Use of trustworthy tools for development and verification.	
Focus on safety, avoidance of non required components and capabilities.	
No generic susceptibilities (e.g., no management of time and date).	
Static allocation of resources.	
Deterministic behavior.	Avoidance of triggering conditions through rigorous identification, characterization and minimization of factors that can influence the functioning of software. Software is confined to well-tested trajectories.
Invariability of software during operation.	
Validation of inputs prior to further processing.	
Clearly identified short term memory.	
Interrupts only for exceptions and clock, no process driven interrupts.	
Cyclic functioning.	Avoidance of various potential fault triggering conditions. Increased assurance that a safety system CCF is very unlikely to occur with the system in normal “run” mode (checking real-time plant parameters against trip setpoints)
Single-tasking.	
Limited amount of short term memory.	
Asynchronous operation (no internal clock).	
Non- software watchdogs (failure of the digital system or channel to periodically reset a watchdog results in a specified safe action within a specified time frame).	Fault tolerance: software deviations and failures are detected, and,the system is rapidly driven to a safe state.
Surveillance of short and long term memory. Defensive programming.	
Rigorous operational procedures for operator requests (one channel at a time, only when absolutely necessary).	Fault tolerance through measures ensuring that digital failures triggered by operator request or elapsed time, e.g., counter or buffer overflows, do not concurrently affect multiple channels or systems.
Staggered startups of redundant channels and diverse systems	
“Dissociation” of Operating System from Application Software.	Avoidance of triggering conditions for Operating System faults, Operating system confined to well-tested trajectories. Plant transients and unanticipated plant behaviors cannot trigger residual faults in Operating System. Improved fault tolerance of Operating System functionality
Transparency of Operating System to plant transients.	
.Constant bus loading (processors and communications)	
Further decomposition of Operating System into dissociated modules	
Application Function Library composed of dissociated, simple, stateless, well-proven modules.	Additional assurance that the Application Function Library is very unlikely to contain design faults that could lead to digital failures.

Table A-3
Examples of Defensive Measures for Smart Devices with Simple Fixed Functionality

Defensive Measures	Benefits
Application of documented and rigorous configuration management program.	Precise identification of the item, assuring that items with the same identification are identical.
Track record for control of changes and versions, and notification of changes (especially software fixes).	
Complete and unambiguous documentation.	Characterization of the item, stating in particular what it does, how well it does it, what is guaranteed it will not do, how it can fail, how it should be used, what it needs for correct operation.
Accurate documentation consistent with actual design.	
Adequacy to support needed functionality.	Fitness to purpose.
Unneeded / unused capabilities shown to have no adverse impact on required functionality.	
Rigorous development, manufacturing, and modification processes.	Low level of residual digital design faults.
Functional and technical simplicity.	
Sufficient amount of credible, relevant, and successful operating history.	
Testing in expected operational conditions.	
Error handling capabilities, built-in protective features, ability to handle expected and unforeseen errors and abnormal conditions and events.	Robustness, fault-tolerance.
Technical assurance that the device is used in narrow operational conditions, consistent with the bounds of its qualification.	Safe use of the device.
External surveillance by other portions of the I&C system, which increases the likelihood that failures or drifts are rapidly detected.	

Export Control Restrictions

Access to and use of EPRI Intellectual Property is granted with the specific understanding and requirement that responsibility for ensuring full compliance with all applicable U.S. and foreign export laws and regulations is being undertaken by you and your company. This includes an obligation to ensure that any individual receiving access hereunder who is not a U.S. citizen or permanent U.S. resident is permitted access under applicable U.S. and foreign export laws and regulations. In the event you are uncertain whether you or your company may lawfully obtain access to this EPRI Intellectual Property, you acknowledge that it is your obligation to consult with your company's legal counsel to determine whether this access is lawful. Although EPRI may make available on a case-by-case basis an informal assessment of the applicable U.S. export classification for specific EPRI Intellectual Property, you and your company acknowledge that this assessment is solely for informational purposes and not for reliance purposes. You and your company acknowledge that it is still the obligation of you and your company to make your own assessment of the applicable U.S. export classification and ensure compliance accordingly. You and your company understand and acknowledge your obligations to make a prompt report to EPRI and the appropriate authorities regarding any access to or use of EPRI Intellectual Property hereunder that may be in violation of applicable U.S. or foreign export laws or regulations.

The Electric Power Research Institute Inc., (EPRI, www.epri.com) conducts research and development relating to the generation, delivery and use of electricity for the benefit of the public. An independent, nonprofit organization, EPRI brings together its scientists and engineers as well as experts from academia and industry to help address challenges in electricity, including reliability, efficiency, health, safety and the environment. EPRI also provides technology, policy and economic analyses to drive long-range research and development planning, and supports research in emerging technologies. EPRI's members represent more than 90 percent of the electricity generated and delivered in the United States, and international participation extends to 40 countries. EPRI's principal offices and laboratories are located in Palo Alto, Calif.; Charlotte, N.C.; Knoxville, Tenn.; and Lenox, Mass.

Together...Shaping the Future of Electricity

Program:

Nuclear Power

1019182

Electric Power Research Institute
3420 Hillview Avenue, Palo Alto, California 94304-1338 • PO Box 10412, Palo Alto, California 94303-0813 USA
800.313.3774 • 650.855.2121 • askepri@epri.com • www.epri.com