INL/RPT-22-70056-Revision-0

Risk Analysis of Various Design Architectures for High Safety-significant Safety-related Digital Instrumentation and Control Systems of Nuclear Power Plants during Accident Scenarios

Han Bao, Sai Zhang, Robert Walker Youngblood III, Tate Shothill, Priyanka Pandit, Edward Chen, Jooyoung Park, Mihai Diaconeasa, Nam Dinh, Svetlana Lawrence

Idaho National Laboratory

INL is a U.S. Department of Energy National Laboratory operated by Battelle Energy Alliance, LLC

DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

INL/RPT-22-70056-Revision-0

Risk Analysis of Various Design Architectures for High Safety-significant Safety-related Digital Instrumentation and Control Systems of Nuclear Power Plants during Accident Scenarios

Han Bao, Sai Zhang, Robert Walker Youngblood III, Tate Shothill, Priyanka Pandit, Edward Chen, Jooyoung Park, Mihai Diaconeasa, Nam Dinh, Svetlana Lawrence

December 2023

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

INL/RPT-22-70056

Light Water Reactor Sustainability Program

Risk Analysis of Various Design Architectures for High Safety-Significant Safety-Related Digital Instrumentation and Control Systems of Nuclear Power Plants During Accident Scenarios

November 2022

U.S. Department of Energy

Office of Nuclear Energy

DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

INL/RPT-22-70056

Light Water Reactor Sustainability Program

Risk Analysis of Various Design Architectures for High Safety-Significant Safety-Related Digital Instrumentation and Control Systems of Nuclear Power Plants During Accident Scenarios

Han Bao[1], Sai Zhang[1], Robert Youngblood[1], Tate Shorthill[2], Priyanka Pandit[3], Edward Chen[3], Jooyoung Park[1], Heng Ban[2], Mihai Diaconeasa[3], Nam Dinh[3], Svetlana Lawrence[1]

November 2022

[1]Idaho National Laboratory
Idaho Falls, ID 83415

[2]University of Pittsburgh
Pittsburgh, PA 152601

[3]North Carolina State University
Raleigh, NC 27695

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

EXECUTIVE SUMMARY

This report documents activities performed by Idaho National Laboratory (INL) for the U.S. Department of Energy (DOE) Light Water Reactor Sustainability (LWRS) Program, Risk Informed Systems Analysis (RISA) Pathway, digital instrumentation and control (DI&C) risk assessment project. In FY 2019, the RISA Pathway initiated a project to develop a risk assessment strategy for delivering a technical basis to support effective, and secure DI&C technologies for digital upgrades/designs. A risk assessment-informed framework was proposed for this strategy, which aims to (1) provide a best-estimate, risk informed capability to quantitatively estimate the safety margin obtained from plant modernization, especially for high safety-significant safety-related (HSSSR) DI&C systems, (2) support and supplement existing risk informed DI&C design guides by providing quantitative risk information and evidence, (3) offer a capability of design architecture evaluation of various DI&C systems, (4) assure the long-term safety and reliability of HSSSR DI&C systems, and (5) reduce uncertainty in costs and support integration of DI&C systems in the plant.

To achieve these technical goals, the LWRS-developed framework provides a means to address relevant technical issues by: (1) defining a risk informed analysis process for DI&C upgrade that integrates hazard analysis, reliability analysis, and consequence analysis, (2) applying risk informed tools to address common cause failures (CCFs) and quantify corresponding failure probabilities for DI&C technologies, particularly software CCFs, (3) evaluating the impact of digital failures at the component level, system level, and plant level, and (4) providing insights and suggestions on designs to manage the risks, thus to support the development and deployment of advanced DI&C technologies in nuclear power plants (NPPs).

Adding diversity within a system or components is the primary means to eliminate and mitigate CCFs, but diversity also increases system complexity and may not address all sources of systematic failures. Optimization of diversity and redundancy applications for the safety-critical DI&C systems remains a challenge. To deal with the technical issues in addressing potential software CCFs in HSSSR DI&C systems of NPPs and supporting relevant design optimization, the proposed framework provides:

- A best-estimate, risk informed capability to address new technical digital issues quantitatively, focusing on software CCFs in HSSSR DI&C systems of NPPs;
- A common and a modularized platform for DI&C designers, software developers, cybersecurity analysts, and plant engineers to predict and prevent risk in the early design stage of DI&C systems;
- Technical bases and risk informed insights to assist users address the risk informed alternatives for evaluation of CCFs in HSSSR DI&C systems of NPPs;
- A risk informed tool that offers a capability of design architecture evaluation of various DI&C systems to support system design decisions in diversity and redundancy applications.

The research and development efforts of this project in FY 2022 are focused on methodology improvement including software CCF modeling and estimation, prevention analysis, importance analysis, and risk analysis of various design architectures of HSSSR DI&C systems. This research is intended to enhance the robustness of the methodology for the risk assessment and design optimization of safety-critical DI&C systems.

The primary audience of this report are DI&C designers, engineers, and probabilistic risk assessment (PRA) practitioners. This includes stakeholders, such as the nuclear utilities and regulators who consider the deployment and upgrade of DI&C systems, DI&C software developers and reviewers, and cybersecurity specialists.

It should be noted that all the analyses are performed for the demonstration of the methodology, not for the evaluation of an actual digital control system. Results are obtained based on limited design information and testing data.

CONTENTS

FIGURES

TABLES

ACRONYMS

AFM	alpha factor model
ATWS	anticipated transient without scram
BAHAMAS	Bayesian and HRA-Aided Method for the Reliability Analysis of Software
BBN	Bayesian Belief Network
BCA	Bistable Comparator Algorithm
BFM	beta factor model
BP	bistable processor
BTP	branch technical position
CCBE	common cause basic event
CCCG	common cause component group
CCF	common cause failure
CCS	component control system
CDF	core damage frequency
CF	coverage factor
CFG	Control Flow Path
CIM	component interface module
CPS	computer-based procedure system
DEPM	Dual Error Propagation Method
DFG	Data Flow Graph
DI&C	digital instrumentation and control
DOE	U.S. Department of Energy
DOE–NE	U.S. Department of Energy–Office of Nuclear Energy
DOM	digital output module
DPS	diverse protection system
EPRI	Electric Power Research Institute
ERP	Enhanced Resilient Plant
ESF	engineered safety feature
ESFAS	Engineered Safety Features Actuation System
ET	event tree
ETA	event tree analysis
FT	fault tree
FTA	fault tree analysis
FV	Fussell-Vesely

FY	Fiscal Year
GC	group controller
HAZCADS	Hazards and Consequence Analysis for Digital Systems
HPI	high-pressure injection
HRA	human reliability analysis
HSSSR	High Safety-significant Safety-related
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IFPD	information flat panel display
INL	Idaho National Laboratory
IPS	information processing system
LC	loop controller
LCL	local coincidence logic
LDP	large display panel
LOCA	loss-of-coolant accident
LOSC	loss-of-seal cooling
LP	logic processor
LPI	low-pressure injection
LWR	light water reactor
LWRS	Light Water Reactor Sustainability
MCR	main control room
MDAFP	motor-driven auxiliary feedwater pumps
ML	machine learning
MMI	man-machine interface
NAMAC	Near Autonomous Management and Control System
NEI	Nuclear Energy Institute
NIST	National Institute of Standards and Technology
NPP	nuclear power plant
NRC	U.S. Nuclear Regulatory Commission
ODC	orthogonal-defect classification
ORCAS	Orthogonal-defect Classification for Assessing Software reliability
PWR	pressurized water reactor
PRA	probabilistic risk assessment
PV	Process Variable
QIAS-P	qualified indication and alarm system – safety

QIAS-N	qualified indication and alarm system – non-safety
R&D	research and development
RCCA	rod cluster control assembly
RCS	reactor coolant system
RESHA	Redundancy-guided Systems-theoretic Hazard Analysis
RG	Regulatory Guide
RISA	Risk Informed Systems Analysis
RPS	reactor protection system
RSR	reserve shutdown room
RTB	reactor trip breaker
RTS	reactor trip system
RTSS	reactor trip switchgear system
SDLC	software development life cycle
SAPHIRE	Systems Analysis Programs for Hands-on Integrated Reliability Evaluations
SP	selective processor
SPADES+	safety parameter display and evaluation system+
SPAR-H	Standardized Plant Analysis Risk-Human Reliability Analysis method
SSC	system, structure, and component
ST	shunt
STPA	systems-theoretic process analysis
UCA	unsafe control action
UIF	unsafe information flow
U.S.	United States
UV	undervoltage

RISK ANALYSIS OF VARIOUS DESIGN ARCHITECTURES FOR HIGH SAFETY-SIGNIFICANT SAFETY-RELATED DIGITAL INSTRUMENTATION AND CONTROL SYSTEMS OF NUCLEAR POWER PLANTS DURING ACCIDENT SCENARIOS

1. INTRODUCTION

This report documents the activities performed by Idaho National Laboratory (INL) during fiscal year (FY) 2022 for the U.S. Department of Energy (DOE) Light Water Reactor Sustainability (LWRS) Program, Risk Informed Systems Analysis (RISA) Pathway, digital instrumentation and control (DI&C) risk assessment project [1], [2], [3]. The LWRS program, sponsored by the U.S. DOE and coordinated through a variety of mechanisms and interactions with industry, vendors, suppliers, regulatory agencies, and other industry research and development (R&D) organizations, conducts research to develop technologies and other solutions to improve economics and reliability, sustain safety, and extend the operation of nation's fleet of nuclear power plants (NPPs). The LWRS program has two objectives to maintain the long-term operations of the existing fleet: (1) to provide science- and technology-based solutions to industry to implement technology to exceed the performance of the current business model and (2) to manage the aging of systems, structures, and components (SSCs) so NPP lifetime can be extended, and the plants can continue to operate safely, efficiently, and economically.

As one of the LWRS program’s R&D pathways, the RISA Pathway aims to support decision-making related to economics, reliability, and safety providing integrated plant systems analysis solutions through collaborative demonstrations to enhance economic competitiveness of the operating fleet. The goal of the RISA Pathway is to conduct R&D to optimize safety margins and minimize uncertainties to achieve economic efficiencies while maintaining high levels of safety. This is accomplished in two ways: (1) by providing scientific basis to better represent safety margins and factors that contribute to cost and safety; and (2) by developing new technologies that reduce operating costs.

One of the research efforts under the RISA Pathway is the DI&C Risk Assessment project, which was initiated in FY 2019 to develop a risk assessment strategy for delivering a strong technical basis to support effective, licensable, and secure DI&C technologies for digital upgrades and designs [1]. An integrated risk assessment framework for the DI&C systems was proposed for this strategy which aims to:

- Provide a best-estimate, risk informed capability to quantitatively and accurately estimate the safety margin obtained from plant modernization, especially for the high safety-significant safety-related (HSSSR) DI&C systems
- Support and supplement existing advanced risk informed DI&C design guides by providing quantitative risk information and evidence
- Offer a capability of design architecture evaluation of various DI&C systems to support system design decisions and diversity and redundancy applications
- Assure the long-term safety and reliability of HSSSR DI&C systems
- Reduce uncertainty in costs and support integration of DI&C systems at NPPs.

The proposed risk assessment framework for DI&C systems is shown in Figure 1. In this framework, a redundancy-guided systems-theoretic method for hazard analysis (RESHA) was developed for HSSSR DI&C systems to support I&C designers and engineers to address both hardware and software CCFs and qualitatively analyze their effects on system availability [4] [5]. It also provides a technical basis for implementing reliability and consequence analyses of unexpected software failures, and supporting the optimization of defense-in-depth applications in a cost-efficient way. The framework integrates STPA [6], FTA, and HAZCADS [7] methodologies to effectively identify software CCFs in complex systems with multiple levels of redundancy. More specifically, STPA is reframed in a redundancy-guided way, such as (1) depicting a redundant and diverse system via a detailed representation; (2) refining different redundancy levels based on the structure of DI&C systems; (3) constructing a redundancy-guided multilayer control structure; and (4) identifying potential CCFs in different redundancy levels. This approach has been demonstrated and applied for the hazard analysis of a four-division digital RTS [4] and a four-division digital ESFAS [5]. These efforts are described in the LWRS-RISA milestone reports for FY-20 [2] and FY-21 [3].

The second part in risk analysis is the reliability analysis which includes tasks of (1) quantifying the probability of basic events of the integrated FT from the hazard analysis; (2) estimating the probability of consequences resulting from digital system failures. In the proposed framework, two methods have been developed: the Bayesian and human-reliability-analysis-aided method for the reliability analysis of software (BAHAMAS) [8] and orthogonal-defect classification for assessing software reliability (ORCAS). BAHAMAS is applicable in situations with limited data conditions (e.g., early stage of system development) and ORCAS is applicable for analyses when significant amount of data is available (e.g., fully-developed system that underwent verification and validation or a system with significant length of operating experience).

Finally, the consequence analysis is conducted to quantitatively evaluate the impact of digital failures on plant overall risks by assessing affected behaviors and responses. Some digital-based failures may initiate an event or scenario that was not analyzed before (e.g., a failure mode only applicable to a digital system), which could challenge the plant safety.

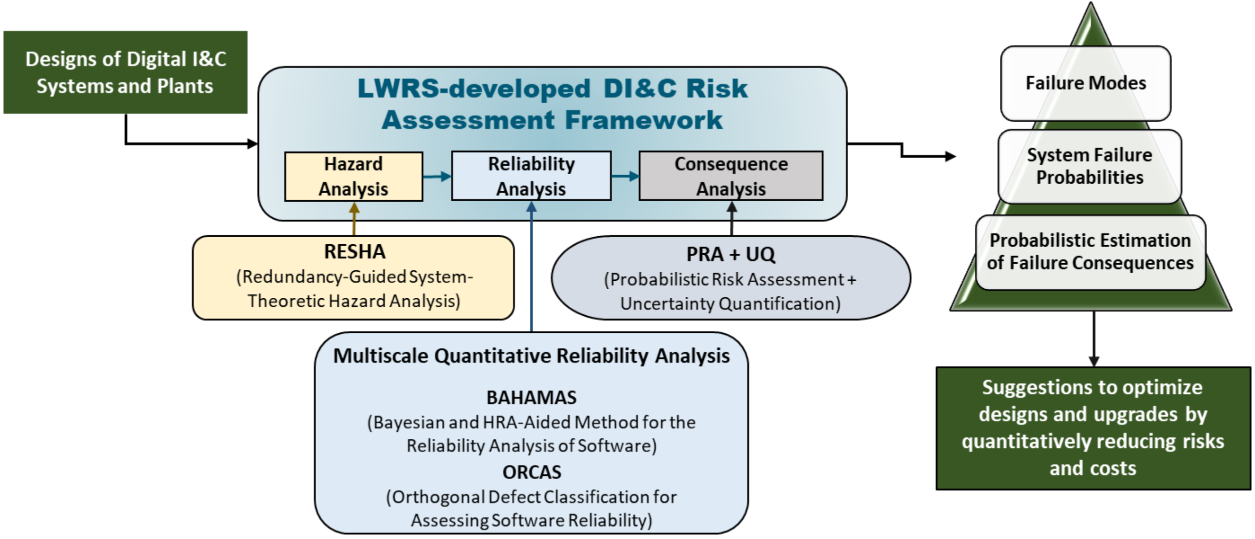

Designs of Digital I&C Systems and Plants
LWRS-developed DI&C Risk Assessment Framework
Hazard Analysis
Reliability Analysis
Consequence Analysis
RESHA
(Redundancy-Guided System-Theoretic Hazard Analysis)
PRA + UQ
(Probabilistic Risk Assessment + Uncertainty Quantification)
Multiscale Quantitative Reliability Analysis
BAHAMAS
(Bayesian and HRA-Aided Method for the Reliability Analysis of Software)
ORCAS
(Orthogonal Defect Classification for Assessing Software Reliability)
Failure Modes
System Failure Probabilities
Probabilistic Estimation of Failure Consequences
Suggestions to optimize designs and upgrades by quantitatively reducing risks and costs

Figure 1. The flexible and modularized structure of the proposed risk assessment framework for HSSSR DI&C systems.

This report outlines R&D focused on methodology improvements of software CCF modeling and estimation and introduces additional innovative approaches to risk assessment of DI&C systems such as prevention analysis and importance analysis to enable a comparative assessment of various DI&C design architectures. The remaining sections of the report are organized as follows: Section 2 describes the event tree (ET) and fault tree (FT) structures for diverse and redundant DI&C systems which are analyzed in this report. Section 3 introduces the software CCF modeling approach developed for diverse and redundant DI&C systems. Section 4 presents the results of sensitivity and importance analyses conducted for different designs of RTS and ESFAS. Section 5 discusses the application of Top Event Prevention

Analysis (TEPA) to a simplified RTS-FT model. Section 6 summarizes the work of a FY-22 summer internship completed at INL to develop a preliminary model for quantifying software CCFs using Dual Error Propagation Method (DEPM). Section 5 outlines conclusions and future work of this project.

2. ACCIDENT SCENARIOS AND SYSTEM DESIGNS FOR DIVERSE AND REDUNDANT SYSTEMS

The importance and sensitivity analyses in this report were performed based on a PRA model of a generic pressurized-water reactor (PWR) plant. A PRA model consists of ETs representing accident scenarios and FTs representing system designs. This section introduces accident scenarios selected for the analyses in this report. This section also describes the FTs representing different RTS and ESFAS designs.

2.1 Event Trees for Accident Scenarios

In this project, an accident scenario is defined as the combination of an initiating event and the success or failure states of relevant plant systems performing mitigating functions:

- An initiating event is an unplanned event that occurs while a plant is in critical operation, requiring that the plant to shut down and achieve a stable, controllable state. For the studies in this report, the same initiating events as those analyzed in previous work under this project [9] were selected including general plant transient (TRANS), small-break loss-of-coolant accident (SBLOCA), and medium-break loss-of-coolant accident (MLOCA).
- The accident scenarios led by each initiating event are modeled using ETs. An ET starts from an initiating event and ends with multiple accident scenarios in different end states. The first box in the top row of an ET represents the initiating event, the rest of boxes in the top row represents different plant systems, and the tree branches represents the success (upper branch) or failure (lower branch) of a system. If an ET has a very large structure, it may be broken down into several sub trees. The ET modeling and quantification are performed using a PRA software, Systems Analysis Program for Hands-On Integrated Reliability Analysis method (SAPHIRE), which is developed by the INL for the Nuclear Regulatory Commission. The ET structures for the three initiating events selected for this report are provided in Appendix A.

2.2 Fault Trees for System Designs

An FT can be developed based on the system design to identify what can go wrong, figure out how the go-wrongs can combine and lead to a system failure, and quantify how likely the system failure is. The following is the system descriptions investigated for this work and used as bases to construct system FTs. This work investigates the effect of diversity on the reliability values of a digital RTS and a digital ESFAS system. While full details of system descriptions can be found in [9], the digital RTS is briefly explained in this section for reader's convenience.

The digital RTS in Figure 2 consists of four redundant divisions of components to monitor and ensure safety. Division-specific sensor signals are sent to the bistable processors (BPs), which determine whether a trip is needed. When required, trip signals from the BPs are sent to each division's local coincidence logic processors (LPs). The LPs vote on the incoming trip signals and send the output via digital output modules (DOMs) to selective relays, which again vote on the trip signals. The outputs of the selective relays pass through undervoltage trip devices (e.g., RTB-D1-UV) and activate the undervoltage reactor trip breakers (e.g., RTB-A1). The correct combination of breakers results in a reactor trip. The success criteria for Trip are given by the opening of a selected set of breakers given by **[**(RTB-A1 OR RTB-B1) AND (RTB-C1 OR RTB-D1)] **OR** [(RTB-A2 OR RTB-C2) AND (RTB-B2 OR RTB-D2)**]**.

As one can imagine, the RTS system consists of numerous hardware and software components and have a large number of failure combinations. One failure path of many paths is shown as below ("->" means "caused by"):

- RTS system failure -> system actuation failure -> breaker failure -> undervoltage breaker trip failure -> logic cabinet rack failure -> digital output module failure -> local coincidence logic processor failure -> bistable processor failure.

It can be seen from above that the RTS system failure can be traced down to as many as seven levels – very complicated. A bistable processor, or BP, failure contributes to the RTS system failure. In this study, design diversity was introduced through adopting different software to control bistable processors, or BPs.

The RTS of Figure 2 serves as the baseline case (i.e., zero diversity of subcomponents). This work investigates the influence of software-based diversity on the reliability of the RTS as well as the risks (measured in core damage frequency [CDF]) from selected initiating events. Specifically, the software controlling BPs is assumed to be diverse. Figure 3 shows the diversity configuration for which the BPs of division B&D are shown in black compared to Figure 2 where all redundant BPs are orange. The functionality of the RTS remains identical to that of the baseline case.

The RTS failure is modeled using a FT consisting of multiple sub trees provided in Appendix B. The FTs of baseline RTS and diverse RTS are mostly the same, but differ in the sub trees representing single BP failures as shown in Figure 4 for the baseline design and Figure 5 for the diverse design. It can be observed that there are more BP software CCF events in the diverse design sub FT. This is because a single BP is placed in more software-related common cause component groups (CCCGs). In the baseline case, a single BP is placed in two software-related CCCGs, including a single-division-level group and a four-division-level group. In the diverse case, a single BP is placed in one more software-related CCCG including two divisions using the same software. With multiple coupling mechanisms considered, corresponding CCCGs may have different values of beta and lead to different CCF probabilities in the FTs.

By employing FT and ET analyses, this work studies the influence of software CCFs on the reliability values of the RTS and the ESFAS, as well as the impacts that the RTS and the ESFAS have on the overall plant risks. The component failure probabilities used for independent and common cause basic events of the baseline system configurations for the RTS FT and the ESFAS FT are given in Appendix B. The given hardware failure probabilities were determined in our previous work [9]. Software CCF values for the BPs were updated from our new scoring system given in the current report. ESFAS components were determined from the same processes given in our previous work [9]; given limited system details, ESFAS hardware components were assumed to have the same individual failure probabilities as the BPs. Both ESFAS and RTS rely on the same total software failure probability (i.e., 2.077E-4). The process for defining the CCFs was described in our previous work [9] and is expanded in the current report. Later sections detail the analysis for the diverse case study and the associated values for the diverse FT will be reported there.

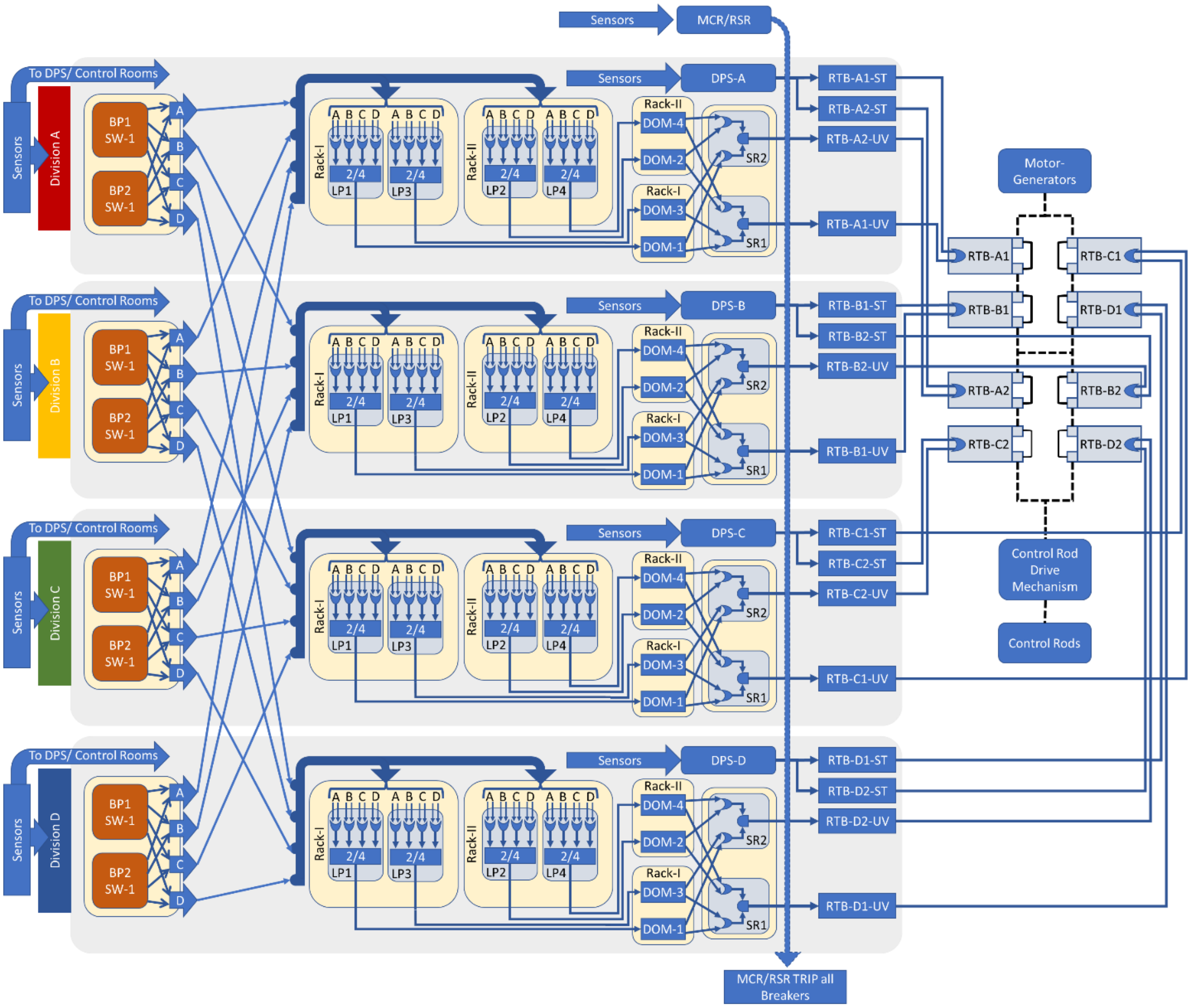


Figure 2. Baseline four-division digital reactor trip system (BPs have Software-1).

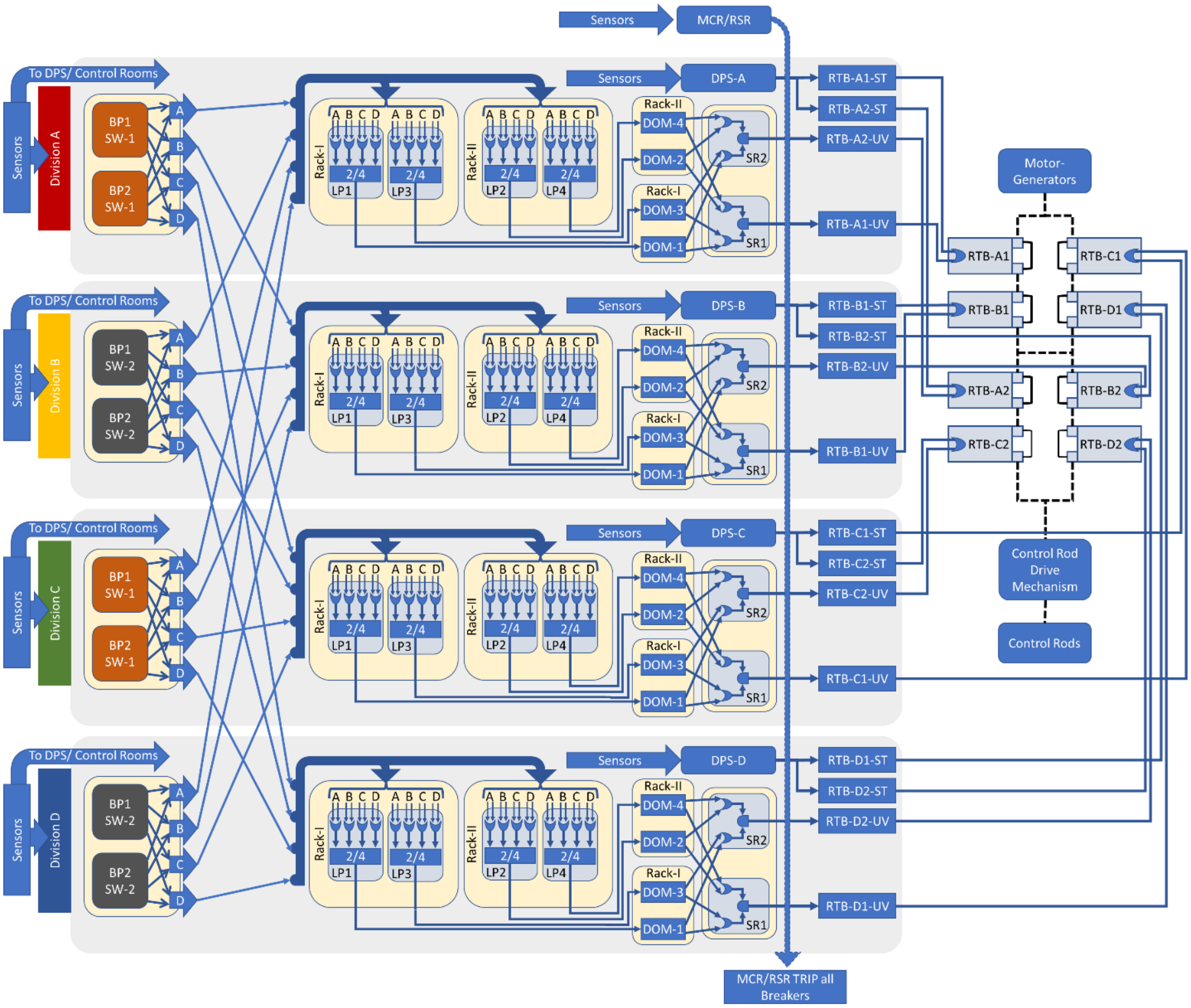


Figure 3. Diverse configuration four-division digital reactor trip system. (Orange components show Software-1 in Divisions A&C. Black components show Software-2 in Divisions B&D).

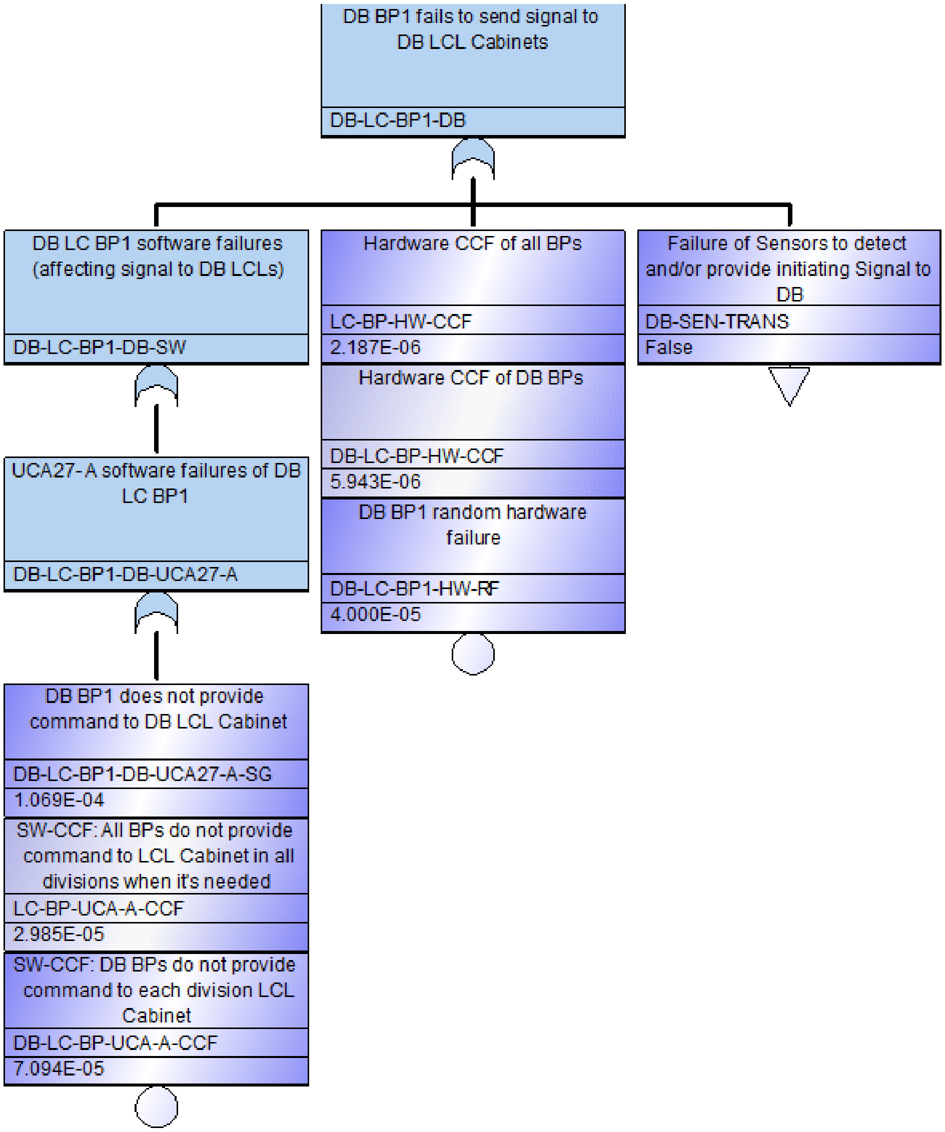

DB BP1 fails to send signal to DB LCL Cabinets
DB-LC-BP1-DB
DB LC BP1 software failures (affecting signal to DB LCLs)
DB-LC-BP1-DB-SW
Hardware CCF of all BPs
LC-BP-HW-CCF
2.187E-06
Failure of Sensors to detect and/or provide initiating Signal to DB
DB-SEN-TRANS
False
Hardware CCF of DB BPs
DB-LC-BP-HW-CCF
5.943E-06
UCA27-A software failures of DB LC BP1
DB-LC-BP1-DB-UCA27-A
DB BP1 random hardware failure
DB-LC-BP1-HW-RF
4.000E-05
DB BP1 does not provide command to DB LCL Cabinet
DB-LC-BP1-DB-UCA27-A-SG
1.069E-04
SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it's needed
LC-BP-UCA-A-CCF
2.985E-05
SW-CCF: DB BPs do not provide command to each division LCL Cabinet
DB-LC-BP-UCA-A-CCF
7.094E-05

Figure 4. Sub fault tree showing the failures of a single bistable processor in the baseline design.

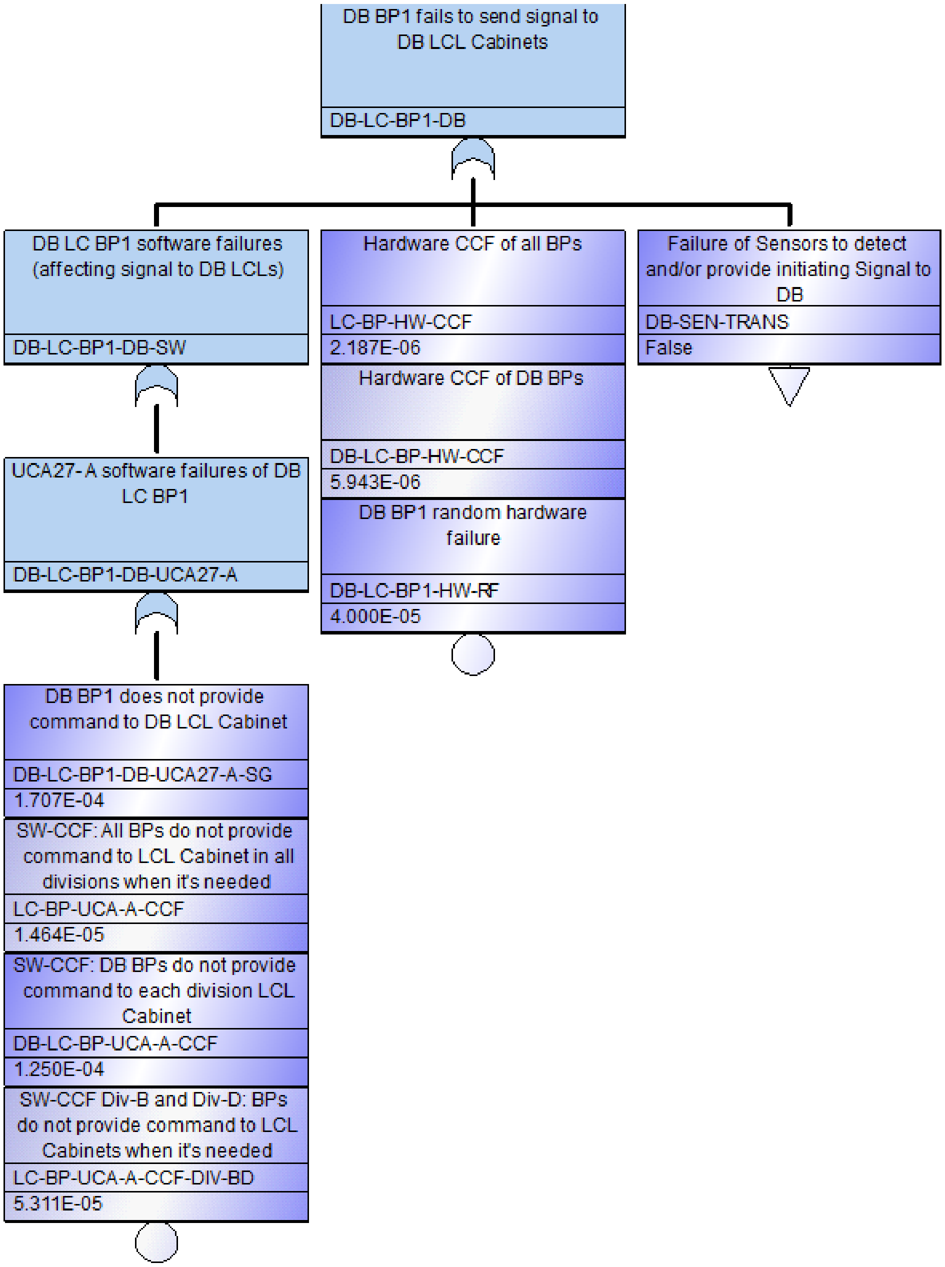

DB BP1 fails to send signal to DB LCL Cabinets
DB-LC-BP1-DB
DB LC BP1 software failures (affecting signal to DB LCLs)
DB-LC-BP1-DB-SW
Hardware CCF of all BPs
LC-BP-HW-CCF
2.187E-06
Failure of Sensors to detect and/or provide initiating Signal to DB
DB-SEN-TRANS
False
Hardware CCF of DB BPs
DB-LC-BP-HW-CCF
5.943E-06
UCA27-A software failures of DB LC BP1
DB-LC-BP1-DB-UCA27-A
DB BP1 random hardware failure
DB-LC-BP1-HW-RF
4.000E-05
DB BP1 does not provide command to DB LCL Cabinet
DB-LC-BP1-DB-UCA27-A-SG
1.707E-04
SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it's needed
LC-BP-UCA-A-CCF
1.464E-05
SW-CCF: DB BPs do not provide command to each division LCL Cabinet
DB-LC-BP-UCA-A-CCF
1.250E-04
SW-CCF Div-B and Div-D: BPs do not provide command to LCL Cabinets when it's needed
LC-BP-UCA-A-CCF-DIV-BD
5.311E-05

Figure 5. Sub fault tree showing the failures of a single bistable processor in the diverse design.

3. COMMON CAUSE FAILURE MODELING FOR DIVERSE AND REDUNDANT SYSTEMS

This section describes an approach for modeling software-based CCFs to support the LWRS-developed framework for risk assessment. The concepts presented here are investigations and improvements to our framework to allow for assessments of diversity and redundancy as they play a role in both the probability of CCF and the overall system reliability.

First, it is important to clarify what is meant by CCF. A CCF is the occurrence of two or more failure events due to the simultaneous occurrence of a shared failure cause and a coupling mechanism [10]. The failure cause is the condition to which a failure is attributed, whereas the coupling mechanism creates the condition for the failure cause to affect multiple components, thereby producing a CCF [10]. Some examples of coupling mechanisms include common hardware, function, and maintenance teams [10]. Any group of components that share similarities via coupling mechanisms may have a vulnerability to CCF and can be considered a common cause component group (CCCG) [10]. Diversity is a tool that is used to break up the commonality of components intended to reduce the probability of postulated common failures. Diversity ensures the same function as a redundant system, but by alternative technologies, methodologies, or techniques. Despite its application for analog I&C systems, diversity effectiveness for DI&C systems remains a topic of concern. The use of digital technologies may increase the potential for CCF vulnerabilities due to failures in design specifications and system interactions [11]. Knight and Leveson have indicated that independently developed software may not necessarily fail independently [12]. One argument is that coding education is so homogenous that the functional difference between software is insufficient to be considered diverse [13] [14]. This work provides a method to support the analysis or the susceptibility of diverse software to postulated CCF. The rest of this section is detailed:

Section 3.1 provides a basis for the remaining sections by reviewing our approach for CCF modeling of redundantly configured software. Section 3.2 introduces a CCF modeling approach for diverse configurations. Section 3.3 provides a set of tables for assessing defenses against CCFs. Section 3.4 introduces the application of BAHAMAS to support the concepts proposed in Section 3.2. Section 3.5 demonstrates a case study for the diverse software CCF analysis. Section 3 ends with a discussion in Section 3.6.

3.1 Common Cause Failure Modeling of Redundant Configurations

In our previous work, we introduced an approach for modeling CCFs within DI&C systems [9]. That work assumed purely redundant configurations of components. Here we provide a brief overview of the methodology. Additionally, some improvements have been made. The approach for modeling CCFs within the LWRS framework is based on a modified beta factor approach [15]. In 2012, Kančev and Čepin proposed a modification of the beta factor model (BFM) that allows components to be assigned to multiple CCCGs based on their coupling factors [15]. Our work has expanded on their original model to emphasize the software-centric coupling factors necessary for simultaneous failures of redundant software components (i.e., CCCGs). Most CCF models rely on operational data as the basis for estimating CCF parameters for modeling CCFs. A hybrid approach was developed, motivated by a lack of operational and CCF data for estimating software CCF model parameters, and a need to model single components as part of multiple CCCGs. This hybrid approach leverages existing techniques: a modified BFM to allow single components within multiple CCCGs and a second, qualitative technique to develop software-specific model parameters for each CCCG. This hybrid approach provides a means to overcome the limitations of conventional methods while offering support for design decisions under the limited data scenario. Figure 6 describes the developed approach which emphasizes the identification of CCCGs and the definition of parameters for the CCCGs for quantification of CCFs.

Identification of the CCCGs requires an emphasis on software-centric features (e.g., shared software code, shared requirements, languages). When software components have been grouped by common attributes, the next step is to define model parameters for each CCCG based on how well that group is defended against CCF. Each CCCG is assigned a beta factor (β_n) that represents the contribution of that CCCG to the total failure probability. Total failure probability (Q_T) is represented as the summation of independent (Q_I) and common failure (Q_{CCF}) probability. Wherein, Q_{CCF} is separated into contributions dependent upon the number of CCCGs. The equations associated with the modified BFM, as employed within our previous work, are shown below:

$$Q_{CCF} = P(CCCG_1) + P(CCCG_2) + \cdots P(CCCG_m) \tag{1}$$

$$P(CCCG_n) = \ (\beta_n)Q_T \tag{2}$$

$$\beta_T = \sum_1^m (\beta_n) \tag{3}$$

$$Q_{CCF} = Q_T \sum_1^m (\beta_n) \tag{4}$$

$$Q_I = (1 - \beta_T)Q_T = \left[1 - \sum_1^m (\beta_n)\right] Q_T \tag{5}$$

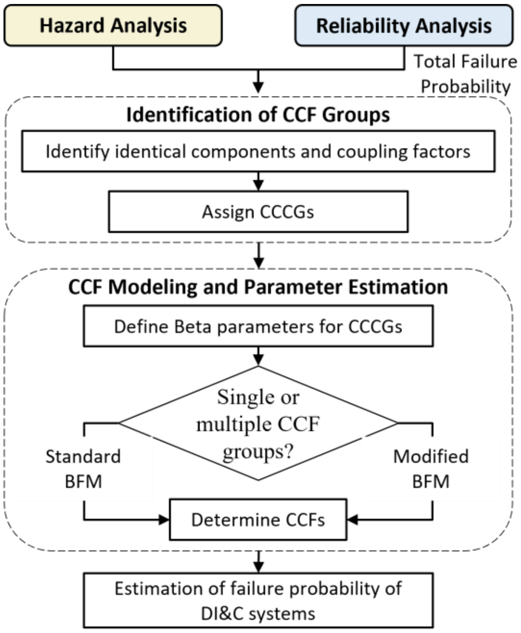

Hazard Analysis
Reliability Analysis
Total Failure Probability
Identification of CCF Groups
Identify identical components and coupling factors
Assign CCCGs
CCF Modeling and Parameter Estimation
Define Beta parameters for CCCGs
Single or multiple CCF groups?
Standard BFM
Modified BFM
Determine CCFs
Estimation of failure probability of DI&C systems

Figure 6. Flowchart for software CCF modeling and estimation.

The process of defining model parameters has its origins in the work developed by Humphreys [16] and was later modified by Brand [17] and served as a foundation for a hardware CCF model used in the International Electrotechnical Commission (IEC) 61508 report [18]. The method is founded on the question: “What attributes of a system reduce CCFs?” [16]. The attributes, called subfactors, are shown in Table 1. Each subfactor is assigned a score (e.g., A, B, and C) to indicate how well a component is defended by a particular subfactor (i.e., A= poorly defended and E= well defended). The subfactor names alone are not sufficient for assessing each subfactor; therefore, our previous work, as well as the original source material provide scoring guidance. The model produces a score for the beta factor by relying on the subfactor scores, in combination with Equation (6).

Equation (6), is a function of the assigned subfactor scores and the denominator d. The model was arranged such that the upper and lower limits for beta correspond with dependent failure values reported in literature [16]. The limits are ensured by the subfactors and d given in Table 1. The beta value determined by this method was intended to be used with the BFM; but in this work, it will be used with the modified BFM.

$$\beta = \frac{\sum(Subfactors)}{d} \tag{6}$$

Table 1. Beta factor estimation table for hardware.

Subfactors	A	A+	B	B+	C	D	E
Redundancy (& Diversity)	1800	882	433	212	104	25	6
Separation	2400		577		139	33	8
Understanding	1800		433		104	25	6
Analysis	1800		433		104	25	6
Man-machine interface	3000		721		173	42	10
Safety Culture & Training	1500		360		87	21	5
Control	1800		433		104	25	6
Tests	1200		288		69	17	4
Denominator for Equation (6), $d = 51000$.							
Note: The table was generated by an automatic calculation that provides slightly different table values than those given in [16]. The original table indicates that scoring an “A” for each subfactor will result in 0.3 for the beta factor [16]. The current table provides 0.300 while the original provides 0.302. The difference is negligible, so this work employs the automated calculation for convenience.							

Two tables are employed to estimate beta factors. Table 1 is intended for hardware CCFs. The second table, Table 2, has been adjusted for software CCFs. Software failure occurs by the activation of latent defects (e.g., deficiencies from coding errors, installation errors, maintenance errors, setpoint changes, and requirements errors). Activation of latent defects is a result of certain operational conditions (i.e., trigger events) [19]. Given a group of redundant software components, variations in their operating conditions may lead to some, but not all, components failing together. Subtle differences in coupling mechanisms may lead to unique combinations of CCFs. To account for software features, Table 1 was modified in two ways: (1) the model was adjusted to increase the upper and lower limits of beta (i.e., 0.001–0.999), allowing for greater applicability to low-diversity software systems; and (2) the subfactor weights were changed to emphasize software-centric features. It is understood that diversity affects CCFs [19]. Consequently, the subfactors that influence diversity were weighted heavily. As an example, the adjusted model emphasizes the introduction of software faults and coupling mechanisms by placing greater weight on those defenses that pertain to human interaction and the diversity of software. Subtle variations in the coupling mechanisms create quasi-diverse components, ultimately influencing the potential for CCFs. The resulting table for software beta factor estimation is shown below.

Table 2. Beta factor estimation table for software.

Subfactors	A	A+	B	B+	C	D	E
Redundancy (& Diversity)	23976	10112	4265	1799	759	135	24
Input Similarity	23976	10112	4265		759	135	24
Understanding	7992		1422		253	45	8
Analysis	7992		1422		253	45	8
Man-machine interface	11988		2132		379	67	12
Safety Culture	6993		1244		221	39	7
Control	4995		888		158	28	5
Tests	11988		2132		379	67	12
Denominator for Equation (6), $d = 100000$.							

Our current approach for CCF modeling is limited to redundant configurations only. The following is an example of how diversity leads to complications with a traditional parametric-based model. Consider a system of two components that are run in a parallel redundancy configuration. The FT for this system is shown in Figure 7 where two components, A and B, have been determined to share a CCCG and are susceptible to a CCF. In this scenario, component A has been generally determined to be more reliable than component B (perhaps due to the difference in manufacturers); the failure probability for A (Q_A) is lower than the Q_B. This difference in reliability presents a challenge for CCF modeling. The failure probabilities for A and B are given by the equations (7) and (8).

$$Q_A = Q_{A_I} + Q_{CCF_{AB}} \tag{7}$$

$$Q_B = Q_{B_I} + Q_{CCF_{AB}} \tag{8}$$

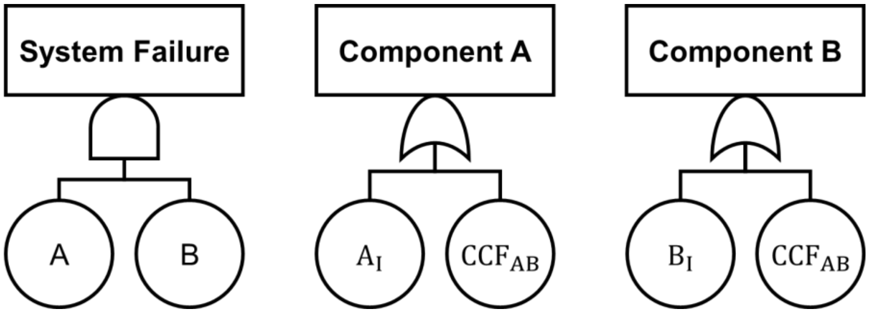


Figure 7. Simple fault tree example.

Traditional CCF modeling assumes $Q_A = Q_B = Q_T$, also known as the symmetry assumption. This assumption is made for the CCCG, and the CCF for the CCCG follows equation (2). A complication arises for the diverse case, $Q_A \neq Q_B$—it becomes necessary to select an appropriate Q_T for evaluating $Q_{CCF_{AB}}$.

Three options are often considered for this scenario (i.e., $Q_A < Q_B$ and the value of β is fixed for the CCCG of AB). Assuming actual performance data are unavailable to directly measure $Q_{CCF_{AB}}$, then the options are [17]:

1. The larger of the values – This is considered a conservative option by Brand [17]. It will predict a larger CCF value than for Option 2. However, it should be easily recognized that there is a potential logical inconsistency especially as β approaches 1.

2. The smaller of the values – Option 2, is supported by the Frechet-Hoeffding upper bound on joint probability distributions [20]. It considers the extreme case in which the entirety of failure events that lead to A fit within the events that lead to B (i.e., $Q_A < Q_B$) corresponding to $\beta = 1$, for such case it should not be possible for $Q_{CCF_{AB}}$ to exceed Q_A.
3. The geometric mean. The geometric mean is presented as an option in [17]. Hauge et al. indicate that this approach is preferred and adequate for rates or probabilities that are of the same magnitude [21].

Use of any option requires consideration of their impact. For example, Option 1 may lead to logical inconsistencies that are bounded by the Frechet-Hoeffding upper bound on joint distributions. Despite being considered as a preferred approach, Option 3 should be checked against Option 2. In all cases, there is still a complication for selecting the correct and meaningful representative parameter (e.g., beta) for the CCCG. Given these challenges, it is proposed to find an alternative approach that better represents the relationship between the components of the CCCG. The next section discusses our approach for CCF modeling of diverse configurations.

3.2 Common Cause Failure Modeling of Diverse Configurations

This work aims to provide insight regarding the influence of diversity on DI&C system reliability. This work must provide an approach for modeling CCFs of diverse software. The existence of diversity within a CCCG is also known as asymmetry. There exist several examples of asymmetric modeling, including one we follow that was developed by Kančev and Čepin [15]. Other examples exist in the literature for dealing with diversity hardware components and CCF. O’Connor and Mosleh proposed a partial alpha-factor model and a Bayesian approach (the general dependency model); an extension to the alpha-factor model, the partial alpha factor works to explicitly model coupling factors between components [22]. The general dependency model relies on a Bayesian network to account for three parameters—a cause condition probability, component fragility, and coupling factor strength [23] [24]. In 2020, Higo et al. developed a method to account for the combined influence of asymmetric and symmetric CCF probabilities by assessing the degree of shared coupling factors [25]. Higo et al. provided some interesting concepts for assessing the degree of similarity between components as measured by the coupling factor [25]. Their method scales existing, symmetric CCCG data to be used for an asymmetric CCCG as a function of the similarity that exists between the asymmetric and symmetric groups. This work was later refined when combined with a gamma-factor model to express inter-unit CCF probability [26]. The methods listed are challenged by dependence on proprietary or non-existent data. Far less data are available for software-based CCFs than for analog CCFs which challenges the application of these recent innovations. In addition, those methods accounting for qualitative differences in coupling mechanisms (e.g., [27] and [23]) rely on data that may not exist for newly designed software systems.

Our decision to employ the beta factor estimation tables has largely reduced our need for operational data for CCF modeling. The remaining challenge is selecting a failure probability value appropriate for the CCCG which can be used in CCF modeling. The primary challenge comes from the difficulty in selecting an appropriate failure value associated with the CCCG. The existing methods rely on a proportionality approach for which CCF is given as fractional (i.e., parameter) percentage of total failure probability. Each group CCF is a function of the total failure probability of the specific component. The problem with modeling CCFs of an asymmetric CCCG is that a Q_T is not the same for each component of the group. Rather than rely on Options 1–3 given in the previous section, a direct estimation is proposed to identify a group’s potential CCF. The new approach introduces a variable that directly represents the similarity of components within the CCCG regardless of any asymmetry in their total failure probabilities. Our previous work relied on Equation (4) to define failure probability of a component due to common causes which was dependent upon a Q_T. We are now proposing to rely on the alternative shown below:

$$Q_{CCF_n} = \phi_n Q_{CC_n} \tag{9}$$

Q_{CC_n}represents the theoretical CCF probability for the CCCG based on the similarity that is shared between the components of the CCCG (e.g., identical software requirements). The benefit provided by Q_{CC_n} is that we avoid approximations typically made for Q_T for the diverse CCCGs. Implementing $Q_{CC\,n}$ for the LWRS framework for CCF results necessitates slightly different equations than those given previously. Equations (1)–(5) have been modified to give:

$$Q_{CCF} = Q_{CCF_1} + Q_{CCF_2} + \cdots Q_{CCF_m} \tag{10}$$

$$Q_{CCF_n} = P(CCCG_n) = \ \phi_n Q_{CC_n} \tag{11}$$

$$Q_{CCF} = \sum_{1}^{m} \phi_n Q_{CC_n} \tag{12}$$

$$Q_I = Q_T - Q_{CCF} \tag{13}$$

$$Q_T = \sum_{1}^{m} \phi_n Q_{CC_n} + Q_I \tag{14}$$

In addition, the use of Q_{CC_n} necessitates the use of a new parameter ϕ_n. Traditionally, parameters (i.e., beta) in parametric modeling represent the fraction of total failure that is due to common cause. For this work, the new parameter, ϕ_n, represents the degree of defense that a particular CCCG has against CCF. When $\phi_n = 1$, the level of defense is poor leading to Q_{CCF_n} equivalent to the theoretical failure probability Q_{CC_n}. The variable Q_{CC_n} considers the designed commonality of the components of the CCCG, while the defense factor, ϕ_n, considers the external influences that act as defensive measures to the CCF of a CCCG. The challenge for this new approach is defining ϕ_n and Q_{CC_n}. In a later section we propose BAHAMAS as a method to define Q_{CC_n}. Regarding parameter estimation, the previous work relied on a set of tables for guidance; this work provides new or updated guidance for defining ϕ_n. The next section details these tables as they pertain to the CCF of software.

3.3 Quantification of Common Cause Failure Model Parameters

This section provides guidance for defining group-specific software-centric parameters for use with the proposed framework for CCF modeling. The adoption of Q_{CC_n} fundamentally considers the designed commonality of the components of the CCCG. This means the defense factor, ϕ_n, only needs to consider the external influences that act as defensive measures to the CCF of a CCCG. This change requires a modification to Table 2. All the subfactors given in Table 2 address an external aspect of CCF defense, while the Redundancy (&Diversity) subfactor give a measure of how similar the CCCG is based on internal aspects (i.e., design). When considering software, the Redundancy (&Diversity) subfactor provided an indication of how similar the software of the CCCG is. This measurement is inherently part of what Q_{CC_n} provides. By contrast, the remaining subfactors consider how external influences may change the commonality that is shared by the CCCG. In other words, the remaining subfactors provided an indication of defense against the internal similarity Q_{CC_n} of the CCCG. Thus, the Redundancy (&Diversity) subfactor is no longer considered for defining the ϕ_n. This change is reflected with the new Table 3. Subsequent sections detail refined descriptions and scoring guidance for each of the seven subfactors.

Table 3. Defense factor estimation table for software.

Subfactors	**A**	**A+**	**B**	**C**	**D**	**E**
Input Similarity	23976	10112	4265	759	135	24
Understanding	7992		1422	253	45	8
Analysis	7992		1422	253	45	8
Man-machine interface	11988		2132	379	67	12
Safety Culture & Training	6993		1244	221	39	7
Control	4995		888	158	28	5
Tests	11988		2132	379	67	12
Denominator for Equation (6), $d = 76000$.						

3.3.1 Input Similarity

The subfactor guidance for Separation (see Table 1) was changed to Input Similarity. Physical separation alone does not influence software failure unless there is consideration for how that physical separation changes the operational conditions of the components. Whereas the Redundancy (&Diversity) subfactor considers the degree of internal similarity, the Input Similarity subfactor considers the degree to which redundant software share external and input similarity. The degree of input similarity provides an indication of how similar the external inputs are for each member of the CCCG. An input ratio (R) provides an indication of approximately how many of the inputs to the CCCG are shared. Additional consideration has been made to account for any degree of diversity that may influence the input (e.g., analog vs. digital signals). Guidance for scoring the Input Similarity is shown in Table 4.

Table 4. Input Similarity estimation table for software.

Score	R=0	0 < R ≤ .5	.5 < R < 1	R = 1	Zero Diversity	Partial Diversity	Complete Diversity
A	X				X	X	X
A+		X			X	X	X
B			X		X		
C			X			X	X
D				X	X	X	
E				X			X
The input ratio (R) is defined: $R = (s - 1)/m$ for $s = 1$ and $R = s/m$ for $s > 1$ where, $m =$ the number of components within the CCCG, and $s = c/i$. $i =$ “inputs/sink” = (number of inputs to the CCCG)/m and $c =$ number of immediate sources.							

3.3.2 Understanding

Understanding reflects “what” the designers, operators, and analysts know in terms of the CCCG being assessed. Brand [17] assumes that that the vulnerability of a system (which we consider as CCCG) to unknown (hence parametric model) failure events is to be related to what the designers, operators, and analysis “know” for the design. Complex, novel software in complex and novel arrangements under limited experience reflect there is just not a lot “known” about the system’s probability for common failure. So, we assess this by considering:

1. **Novelty** – (At the time of design): Does CCCG software contain novel principles or configurations? Do the members of the CCCG work together in a new or novel way? Consider the novelty of the software being used in the CCCG. In diverse CCCGs, any degree of unique software concepts or any novelty at all will be considered “yes” or “novel” (in comparison to

standard practices). Most likely, the novelty score will remain constant for each CCCG that a component belongs to; however, diversity could cause CCCG scores to vary.

- Novelty = (No) low to average (no idea here is new, meaning the concepts for the CCCG configuration are well understood by industry)
- Novelty = (Yes) Above average (First-of-a-kind designs, unique or new software ideas, concepts, or methods employed in otherwise well understood systems).

2. **Complexity** – (At the time of design): From Brand's scoring, a system (or as we consider, the CCCG) may be considered complex if it performs two or more distinct functions [17]. Future work may develop alternative measures to assess the complexity of a CCCG. One idea may be to consider whether the CCCG depends on interactions among itself or with other systems. Note that complex systems will reduce the level of understanding. Most likely, the score will remain constant for each CCCG that a component belongs to; however, diversity could cause CCCG scores to vary.
    - Complexity = "Not complex" if members of the CCCG perform only single function
    - Complexity = "Complex" if members of the CCCG perform multiple functions.

3. **Misfit** – (At the time of design): Does the CCCG contain off-the-shelf rather than purpose built software? Misfit will capture the gap of understanding that designers, operators, and analysts will have for implementing software they did not specifically develop themselves. In contrast, novelty captures the gap associated with novel principles, configurations, or use of novel ideas. Most likely, the score will remain constant for each CCCG that a component belongs to; however, diversity could cause CCCGs scores to vary.
    - Misfit: Low (zero-to-minimal off-the-shelf systems, for example using pre-existing operating systems or functions)
    - Misfit: High (complete off-the-shelf systems, employing pre-build application software).

4. **Experience** – Understanding increases with time and experience; thus, experience reduces the influence that high levels of misfit, complexity, and novelty have on understanding. Experience is credited when there is actual usage data for the CCCG exceeding 10 years. (Future work may refine this number). Originally, experience could be credited when there exists data for similar systems. However, this was intended for hardware. Due to the complexity of software, such credit will only be considered for a new design if the new design employs software that has 10 years of operation experience (i.e., using an off-the-shelf product that has seen 10 years of experience in another system).
    - Experience: (High) 10+ years
    - Experience: (Low) all other conditions.

Table 5. Understanding estimation table-1 for software.

Experience (limited)	
Subfactor Score	Requirement
A	YES, YES, YES
B	NO, YES, YES
C	NO, NO, YES
D	NO, NO, NO
E	Not applicable for experience level

Table 6. Understanding estimation table-2 for software.

Experience (Yes, 10+ years)	
Subfactor Score	Requirement
A	Minimal score not applicable for experience level
B	YES, YES, YES
C	NO, YES, YES
D	NO, NO, YES
E	NO, NO, NO

3.3.3 Analysis

Brand defined the Analysis subfactor by considering the following two questions [17]: (1) "How much analysis has been done on the design?" and (2) "Are designers aware of dependent failures issues?" Question 1 considers whether an analysis has been done, and the degree for which feedback was involved. Question 2 considers the design team's knowledge and application of that knowledge. The designers' awareness can be considered a function of both their knowledge and the application of that knowledge as evident within the design [17]. One idea presented by Brand is that the design team's knowledge is the first line of defense prior to any formal assessments. In the absence of a formal design analysis and feedback, a designer, who is aware of dependent failure issues, may create a design that is equivalently defended when compared to one that has gone through analysis and feedback. Credit for designer awareness cannot entirely replace formal analysis and feedback and is only partially credited within the score table. It should be noted that application of this method is from the perspective of an independent assessment.

Analysis subfactor/guidance clarification and assumptions:

- **Analysis** refers to the assessment that was performed with regards to CCF of the CCCG. For example, analysis refers to the identification of hazards found within a system as performed by a PRA team in support of design and development. The subfactor is applied to a specific CCCG only. The score for Analysis has three levels.
    - Level 1: The CCCG has not been identified and no CCF analysis has been performed for the specific CCCG.
    - Level 2: The CCCG is identified (hazard only), and it may include partial consideration of how a CCF of this CCCG might influence the system.
    - Level 3: The CCCG is identified and analyzed for its threat to system performance (hazard and consequences are tracked perhaps as part of a Fault Tree Analysis (FTA) or another formal tool).
- **Feedback** refers to the information and recommendations provided by the analysts to the designers regarding CCF of the CCCG. For example, feedback refers to recommendations made by a PRA team to facilitate design improvements. The subfactor is applied to a specific CCCG, therefore the feedback should be given within the context of the CCCG only. The scoring for feedback consists of three levels:
    - Level 1: No feedback concerning this CCCG was provided to the design team
    - Level 2: Feedback was provided to the design team concerning the specific CCCG
    - Level 3: Detailed feedback was provided to the design team. There is evidence of the feedback, including formal tracking of recommendations (may track to specific design changes). Feedback involves leadership or management support.

- **Awareness** refers to the designer's knowledge of CCFs and application of that knowledge. Credit for a designer's awareness is made by consideration of the design's defenses against CCF. Awareness is a challenging aspect to score. Designers may have training, but their design indicates minimal defenses for CCF. The reasons for the designer actions may vary; thus, it is more useful to rely on design evidence (i.e., diversity). Rather than simply assess a team member's curriculum vitae, the design itself becomes the ultimate indicator of awareness. Awareness can vary for each CCCG of the system (e.g., a designer may have only considered CCF of one group, but not another).
    - Level 1: There is no evidence of design team knowledge of CCF prevention as it pertains to the specific CCCG. The CCCG has low level of redundancy (i.e., only two redundant software components)
    - Level 2: There is evidence that the designers have general CCF knowledge as demonstrated by redundancy configurations only. No diversity is used.
    - Level 3: There is evidence that the designers have specific awareness of software-based CCFs as evident in the design by the use of diverse software configurations.
- Analysis without feedback to the design provides no defense against CCF.
- Designer awareness can be credited in the absence of analysis and feedback, as evidence is given by the design. Credit only Level C.
- Analysis, feedback, and awareness are rated by importance as follows: Feedback > Analysis > Awareness. Feedback and analysis correct the limitations of designer awareness.

Table 7. Analysis estimation table for software.

Quick guide			
Analysis and Feedback	No Awareness/Evidence (Simple redundancy of 2)	Evidence of General Awareness (2+ or complex redundancy)	Evidence of Specific Awareness (use of diversity)
A, F	B	C	D
A, F+	B	C	D
A+, F	B	C	D
A+, F+	B	D	E
No-A, No-F	A	B	C
No-An = No Analysis; An = Analysis; An+ = Detailed Analysis; No-F = No Feedback; F = Feedback: F+ = Detailed Feedback			

3.3.4 Man-Machine Interface

This subfactor remains largely unchanged from Brand [17], but the focus has been shifted to the software-based interactions. The goal of this subfactor is to assess the interactions that staff have with the CCCG. These are the human interactions that are associated with the system that can introduce errors or defects that are not directly related to development activities. Specifically, the human errors are reduced when there are checklists, written procedures, and review activities to guide the human actions. This subfactor makes adjustment of the degree to which human actions are controlled or ensured. There are two forms of interaction that are assessed for the man-machine interface (MMI): (1) Operator or user interactions and (2) maintenance interactions. Operator or user interactions are controlled by means of procedures and checklists. Maintenance work is controlled or checked by supervisors, tests, checklists, and alarms. Guidance for application of this subfactor is provided below (addressed in terms of the CCCG):

- **Operator or user interactions:**
    - Frequency: How often do the operators, users, or staff interact with the specific CCCG?
        - Minimal: Interactions are low or infrequent.
        - Normal: Interactions are regular, normal, scheduled, or routine.
    - Control: How are the interactions controlled (i.e., are there procedures?)
        - No procedures: There are no written procedures or guidance to interacting with the CCCG.
        - Procedures: There are written procedures to control the operation.
        - Checklists: Checklists are assumed to provide better control than written procedures.
- **Maintenance interactions:**
    - Maintenance interactions are controlled or ensured by supervisor, testing, checklists, and alarms.

The situation may arise when a CCCG has diverse components for which human interaction varies. For example, a CCCG of five components may be monitored by two teams (i.e., three components by Team 1, two by Team 2). The MMI subfactor for the CCCG should consider how diversity is defensive; credit is taken for the better of the MMI scores. The basis for this assumption is that diversity of the CCCG prevents a CCF of the complete CCCG. It is noted that situations may arise where interactions with the CCCG involve only operators or only maintenance teams. The table has been structured for this condition.

Table 8. Man-machine Interface estimation table for software.

Score	Operator/use interactions	Maintenance Interactions
A	(For when only maintenance work considered)	No checking
B	[Written procedures AND Normal interaction] OR [No Procedures AND minimal interaction]	Work checked by a supervisor
C	[Checklist AND Normal interaction] OR [Written Procedures AND minimal interaction]	Post maintenance proof testing
D	[Checklist with evidence that it is used AND Normal interaction] OR [Checklist AND minimal interaction]	Work is tested and checked
E	(For when only maintenance work considered)	Maintenance activities have associated alarms to guarantee correctness

3.3.5 Safety Culture and Training

This subfactor addresses both training and safety culture as influencers of human error. Safety culture may influence the willingness, likelihood, and expectations of positive human performance. While training affects the capabilities of humans to perform well. Staff training and safety culture directly affects the probability of human error. Human error has the potential to influence CCF events. However, here human error is considered with respect to general interactions with the CCCG rather than the development of system software components themselves. It is unclear how the safety culture influences MMI scoring as it was originally defined by Brand [17]. But it is understood that safety culture and training impact human interactions. The safety culture subfactor provides consideration of awareness, knowledge, and perhaps the willingness of staff to safely fulfill their duties. The MMI scoring does not account for safety culture effects. Thus, the safety culture subfactor can be considered a corrective measure for what is overlooked by the MMI subfactor. The following is a discussion of our modification to the scoring criteria used for the Safety Culture and Training subfactor. The original rules assessment for safety culture was based on three questions:

1. How much training/expertise does a person have with the system?
    a. On-the-job training, systematic training, or simulator training?
2. What experience does the person have?
3. What is the safety culture that is present?

There are several complications with these questions as they pertain to CCCGs and, generally, systems. First, simulator training is not a good differentiator for expertise, especially for systems that do not have any associated, or feasible type of simulator training. Not every CCCG will involve clear, isolated, interaction with users or maintenance teams. Second, systematic training is not clearly defined, so a different differentiator, or a clear definition is needed. Third, experience provides the same information that training can provide. Given these complications, the following new rules will be employed to maintain the general idea of the original subfactor scoring criteria, but provide additional clarity. The criteria for Safety Culture & Training subfactor will be based on the following descriptions and used for Table 9:

1. **Education:** What level of education has the staff received regarding the components and software of the CCCG?
    a. On-the-job: No formalized education concerning any aspect of the CCCG.
    b. General: Basic general education concerning of the CCCG.
    c. Specialized: Detailed training, and education related to the specific components and software of the CCCG. Staff education may include simulator training, when applicable. Also, the years of experience may be considered.
2. **Safety Culture:** What is the safety culture present for the staff who interact with the components and software of the CCCG?
    a. Casual: Staff may or may not have safety in mind. No safety training.
    b. Safety Oriented: Staff have safety in mind. Periodic safety training related to working with the components/software of the CCCG.
    c. Safety Oriented+: Staff have safety in mind, there is a clearly defined organizational safety culture, safety policies, regular safety training.

Table 9. Safety Culture and Training estimation table for software.

Score	Safety Culture	Education
A	Casual	On-the-job
B	Casual	General Education
C	Safety Oriented AND (regular OR infrequent safety training)	General Education
D	Safety Oriented AND (regular safety training)	Specialized (may include simulations)
E	Safety Oriented AND (regular safety training) AND (clear safety policy).	Specialized (may include simulations) AND multiple years of experience with the system

3.3.6 Control

Originally, this was considered environmental control and reflected the level of control maintained over the environment in which the system (i.e., CCCG) is located [17]. This subfactor was modified for software, which is largely independent of the physical environment. For software, the primary consideration is control over access to the software found within the CCCG of interest. One such example is the use of physical security measures to limit access to software. There are also non-physical control mechanisms, such as isolated or secured networks, intra-nets, and passwords. These control features have been used to create the table below based on the original concept of environmental control given by Humphreys [16] and Brand [17]

Table 10. Control estimation table for software.

Score	Criteria
A	No control, open access networks, unsecured physical locations.
B	Secured physical locations, private networks, general institutional access, multiple unrelated software systems found in single physical location.
C	Secured physical locations, private networks, limited institutional access (e.g., authorized personal only, and passwords).
D	Secured physical locations, private networks, limited access to authorized and trained personnel only. Close supervision is employed. The area where software is found is limited to software of similar purposes (i.e., multiple software programs on the same machine but all related to similar applications like nuclear power monitoring) multiple systems may be present in the same area.
E	Secured physical locations, private networks, extremely limited access, trained personnel only operating under close supervision, specialized machines (i.e., no other software present), only a single-purpose system is present in the area.

3.3.7 Testing

The original scoring table focused only on environmental testing for hardware components. Software do not need the environmental testing, but it does need software testing. This subfactor will now directly consider the degree of software testing that has occurred for the CCCG of interest. The scoring criteria follow a similar pattern to the one defined for hardware testing by Brand [17] (e.g., the parallel testing criteria for environmental testing was mirrored for software testing). The following is the subfactor table made to address software testing:

Table 11. Testing estimation table for software.

Score	Criteria
A	No testing of the system, specifically the CCCG.
B	Individual unit testing (single examples for each software type within CCCG). An example unit has been tested.
C	Detailed testing is performed on an example system (i.e., CCCG). Testing includes verification and compliance testing to ensure the CCCG meets all required criteria as a unit.
D	Commissioning tests performed on the specific CCCG to be employed. Detailed integration testing of the CCCG, in addition to stress testing.
E	In addition to C&D levels, a long-term test is conducted for the CCCG. The test is performed in parallel with existing system for approximately 1 year.

3.4 Application of Bayesian and HRA-Aided Method for the Reliability Analysis of Software for Common Cause Failure Analysis

This section details how BAHAMAS can be applied to the proposed method for CCF modeling. This section begins by reviewing the BAHAMAS methodology and how BAHAMAS can be applied for CCF modeling. Then the section describes how BAHAMAS can be implemented for CCF analysis of diverse software. Specifically, this section describes how BAHAMAS can identify Q_{CC_n}.

3.4.1 Overview of BAHAMAS Methodology

Within the LWRS framework the RESHA identifies software failure events that should be quantified; BAHAMAS quantifies those events. BAHAMAS was developed for the conditions with limited testing and/or operational data or for reliability estimations of software in early development stages [8]. BAHAMAS is intended to provide an estimation of software failure probabilities to support the design of software and target DI&C systems. This work is expanding BAHAMAS capabilities to guide the implementation of diversity and defense in depth for DI&C designs. Instead of relying on testing data, BAHAMAS employs a Bayesian Belief Network (BBN) to map the causes of software failures to specific defect types, defined by orthogonal-defect classification (ODC) [28], that can be traced to human errors in the software development life cycle (SDLC). In turn, the human errors can be modeled with HRA [8]. Figure 8 shows the general concept employed within BAHAMAS as well as the general structure of the BBN.

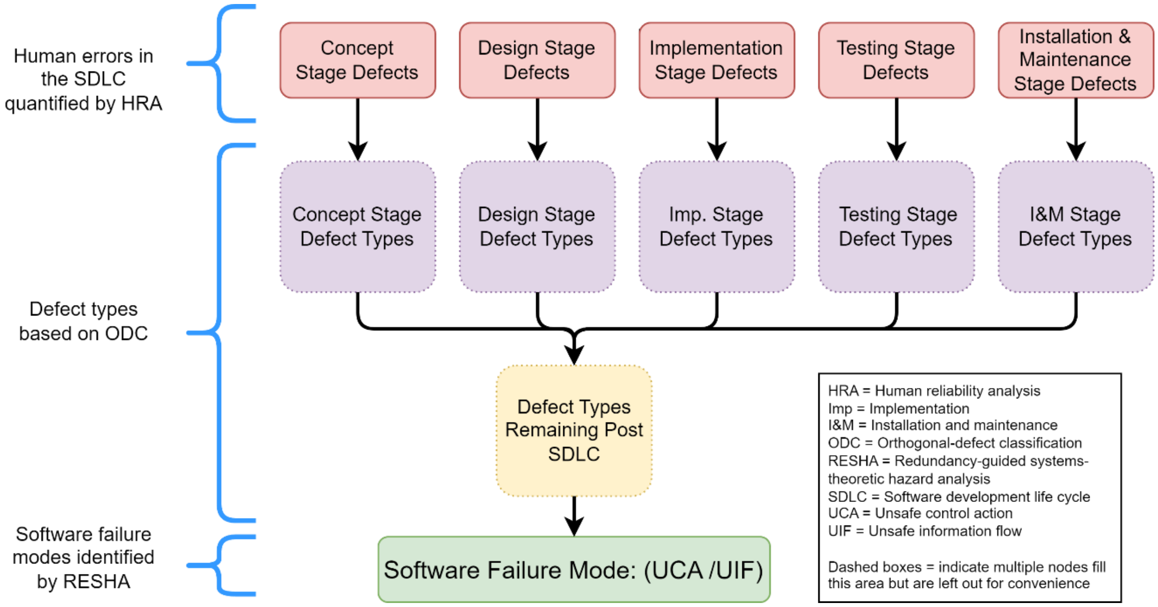

Human errors in the SDLC quantified by HRA
Concept Stage Defects
Design Stage Defects
Implementation Stage Defects
Testing Stage Defects
Installation & Maintenance Stage Defects
Defect types based on ODC
Concept Stage Defect Types
Design Stage Defect Types
Imp. Stage Defect Types
Testing Stage Defect Types
I&M Stage Defect Types
Defect Types Remaining Post SDLC
Software failure modes identified by RESHA
Software Failure Mode: (UCA /UIF)
HRA = Human reliability analysis
Imp = Implementation
I&M = Installation and maintenance
ODC = Orthogonal-defect classification
RESHA = Redundancy-guided systems-theoretic hazard analysis
SDLC = Software development life cycle
UCA = Unsafe control action
UIF = Unsafe information flow
Dashed boxes = indicate multiple nodes fill this area but are left out for convenience

Figure 8. General Structure of the BBN used within BAHAMAS.

When assessing a single piece of software, BAHAMAS considers details of that particular software's development to provide an indication of software failure probability (e.g., software failure on demand). In our previous work we investigated CCF of redundant components that share the same software [9]. For a CCF, there must be (1) a failure involving multiple components and (2) a common cause that is made "shareable" by the existence of some coupling mechanism. The shared cause for software must include a common "active" defect or fault. A CCF can occur because multiple components share copies of the same software and can be influenced by the same trigger mechanism. The only difference between the CCF and individual failure for the identical components is that a shared defect is activated in a single software vs. in the multiple redundant software components. In this instance the system configuration, and not BAHAMAS itself, determines whether a CCF or independent failure occurs. Ultimately, the susceptibility of the software to failure depends on the activation of the hidden defects.

For CCF analysis of a redundantly configured CCCG (i.e., all components have identical software) BAHAMAS effectively provides the indication of the common defects that exist for CCCG. The output of BAHAMAS provides Q_T. Consider a CCCG of two devices (i.e., A and B) each employing Software-1. BAHAMAS can provide the total failure probability of Software-1 ($Q_T = Q_{SW1}$) as indicated by the defects that are expected from the SDLC. In this example, Software-1 total failure is equivalent to the total failure for software A and B:

$$Q_{Sw1} = Q_A = Q_B \tag{15}$$

Given A and B have identical software, BAHAMAS has, in fact, tracked the common defects shared by the CCCG. By tracking the shared aspects of the SDLC, which happens to be 100% identical for A and B, BAHAMAS has provided an indication of (Q_{CC}). Barring any exterior defenses against CCF, the total failure value predicted by BAHAMAS for Software-1 is equivalent to the theoretical CCF of the CCCG:

$$Q_{Sw1} = Q_{CC} \tag{16}$$

Given defenses exist, ϕ is incorporated with equation (16). The resulting equation (17) is equivalent to equation (14) for a single CCCG:

$$Q_T = \phi Q_{CC} + Q_I \tag{17}$$

3.4.2 Application of BAHAMAS to Diverse Software Arrangements

The principal reason that BAHAMAS can be used to evaluate software CCF within redundant configurations is that the CCCG consists entirely of identical software. BAHAMAS provides an implied representation of the common defects that exist for CCCG, where these defects can be traced to human errors during the SDLC. Given a redundant configuration of two components sharing software, both components share 100% of their SDLC. It is hypothesized that diverse software may share defects due to common human errors during their respective SDLC activities, resulting in a set of defects that are common and can lead to common failure of otherwise diverse software.

As an example, consider Figure 9. Software defects found within Software A are given as a function of the SDLC of A while the software defects found within Software B are given as a function for the SDCL of B. The common defects that are shared within Software A and Software B are due to common human errors within both SDLC-A and SDLC-B. It is assumed that the common human errors can be traced to mistakes made during specific tasks that are common between SDLC-A and SDLC-B. BAHAMAS can be structured to consider only the common aspects of SDLC-A and SDLC-B. In other words, performing an analysis with BAHAMAS and only considering the shared SDLC aspects (SDLC-AB) will provide a result associated with the common cause of A and B. BAHAMAS can be employed in two ways, depending on what the needs of the analysis are:

1. For single software, BAHAMAS provides the total failure probability for that software, given as Q_{T_A} for some Software A. For Q_{T_B}, BAHAMAS depends on the SDLC for Software B.
2. For CCF analysis, the output of BAHAMAS provides the probability of failure due to common or shared defects, given as Q_{CC}. For Q_{CC}, BAHAMAS depends on the common aspects of the SDLC that is shared by multiple pieces of software (i.e., the software that is found within the CCCG).
   a. When the CCCG has only one common software, $Q_{CC} = Q_{T_{CCCG\,software}}$
   b. When the CCCG has diversity, $Q_{CC} \neq Q_T$

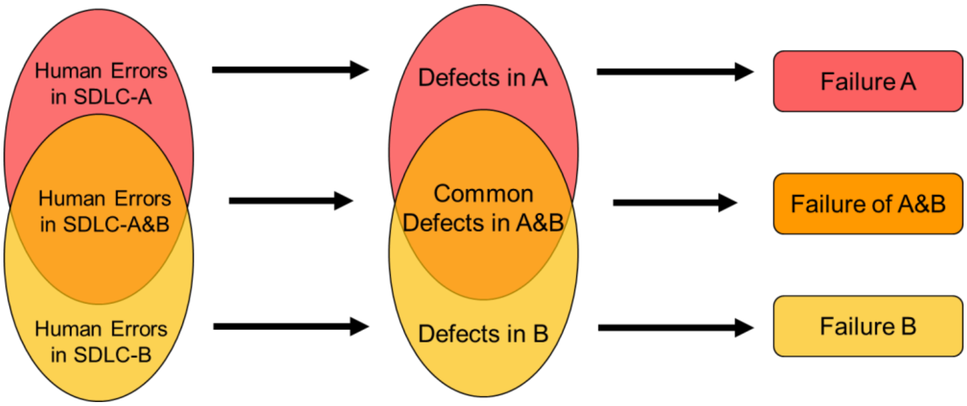


Figure 9. Tracking of common errors leading to potential CCF of diverse software.

At this stage, the output of BAHAMAS has been clarified for diverse software configurations. The next step for CCF analysis is to define the level of defense that the CCCG has against CCF for use in equation (14). The defense factor, ϕ_n, provides a means to determine the independent and CCF probabilities for a CCCG. In the traditional BFM, a beta factor is assigned which indicates the ratio of total and CCFs with the beta factors historically defined from operational data. In this work, we are defining the parameter based on the level of defense that a CCCG has against CCF. Definition of the parameter is accomplished by using the tables discussed in the previous sections.

3.5 Case Study for Diverse Software Common Cause Failure

This section details the quantification of CCFs found within the automatic trip function of a four-division, partially diverse digital RTS. The RTS exhibits partial diversity given that only one group of components has been modified with diversity by software. Figure 2 shows the original or baseline configuration of the for-division digital RTS. Figure 3 shows the partially diverse configuration. The difference between the baseline and the diverse configurations can be found in the BPs of the RTS. The baseline case employs one common software for each BP. The diverse configuration of the RTS employs two diverse software programs within the BPs. Software-1 is found within Divisions A&C while Software-2 is found in Divisions B&D. All other aspects of the diverse configuration case study are identical with that of the base case.

1. All component failure probabilities are identical to the baseline RTS, except for the software of the BPs found within Divisions B&D.
2. The hardware is identical for all BPs
3. Software-1 and Software-2 are diverse.
4. Software-1 was evaluated with BAHAMAS to provide $Q_{Sw1} = 2.077E - 4$.
5. As this is a preliminary work, we did not have design information for Software-2; therefore, its attributes had to be assumed. In this case, the $Q_{Sw2} = 1.75Q_{Sw1} = 3.635E - 4$.
6. The Redundancy (&Diversity) subfactor is a measure of the internal similarity between two components. This case study is relying on the BAHAMAS approach which accounts for the similarity of diverse software by relying on the common aspects of the shared SDLC.
7. Certain details of the system that would be common knowledge for a real system had to be assumed for this hypothetical design. These assumptions make it possible to assign subfactor scores.
    a. It is assumed that the software employed within the RTS is purpose built.
    b. The BPs do not perform any other function than initiate reactor trip.
    c. It is assumed that the software is a new design with little operational experience (less than 10 years).

d. The individuals responsible for developing this RTS performed a safety analysis that included a FT. The analysis of the BPs included written discussion for each CCCG. The FT analysis includes basic events representing only those CCCGs consisting of entirely Software-1 or Software-2). The Analysis assumes diversity prevents CCF for those CCCGs where it is employed.
e. The level of feedback provided from Analysists to designers was casual in nature. There is no evidence of formal recommendations, detailed tracking, or manager involvement in the feedback process.
f. No formal indication was provided to indicate the designer's level of knowledge of CCFs.
g. The interaction of Operators with Software-1 and Software-2 are guided by checklists and interaction is minimal.
h. Maintenance team interactions with the software are guided by procedures that include testing and checking of actions.
i. Maintenance Team 1 has regular safety training, safety oriented work culture, and specialized education. Specialized education indicates they have detailed training and education related to the specific components/software of Software-1.
j. Maintenance Team 2 is from a different organization than Team 1 and their safety training is infrequent. Also, they are a new team with only basic or, at most, a general education regarding the software found in Software-2.
k. Access to the BP software is strictly controlled. Each division is located in its own secured room where only authorized and trained personnel are allowed access. There are multiple software systems besides the RTS within each division-specific room.
l. The RTS is tested upon installation. Functional testing is conducted for the system as a whole and for individual divisions.

The proposed method for CCF analysis requires some initial work to be performed for quantification. The previous section proposed an approach for defining Q_{CC_n} by relying on the capabilities of BAHAMAS. The identification of Q_{CC_n} is necessary for performing CCF analysis. For this case study, we are relying on the previous efforts where BAHAMAS was used to define the failure probability of Software-1 as $Q_{Sw1} = 2.077E - 4$. For this work, Software-2 is assumed to be $Q_{Sw2} = 3.635E - 4$. Given the lack of information, it was necessary to assume how similar Software-1 and Software-2 are in this case study. It is assumed that they have only partial similarity in their SDLCs, such that $Q_{CC_n} = 0.5Q_{Sw1} = 1.0385E - 4$. Note that, $Q_{CC_n} = 1.0385E - 4$, for all CCCGs involving Software-1 and Software-2. For single software groups, $Q_{CC_n} = Q_{Sw1}$, or Q_{Sw2}, depending on what software fills a particular CCCG.

Figure 6 shows that the CCCGs must be assigned by identifying functionally identical components and their coupling factors. The objective of this case study is to assess the BPs in the RTS—there are eight, two per each of the four divisions. They each have identical functions and hardware, but their software is diverse. Identification of the CCCGs depends on the coupling mechanisms that can be identified and used to differentiate groups of components or software. Some examples of software-relevant coupling mechanism may include common external operational conditions (i.e., shared inputs), shared SDLC, shared human interactions (e.g., shared users, installation teams, or maintenance teams), and common requirements. The BPs within Divisions A&C employ Software-1, while Divisions B&D rely on Software-2. Location creates an operational environment that is unique for the BP software. Input from division-specific sensors create the potential for the BPs to have division-specific CCFs associated with their operational conditions; therefore, the BPs in A, B, C, and D can be separated by their input and operational conditions. In addition, all the BPs share a commonality associated with purpose or function, and it is assumed there exists some degree of commonality between Software-1 and Software-2 that is associated with common SDLC practices.

Table 12. CCCGs for the BPs.

CCCGs	Coupling Mechanisms
BPs in Division A	Software-1 SDLC, Division A Specific Input, Software-1 Maintenance Team.
BPs in Division B	Software-2 SDLC, Division B Specific Input, Software-2 Maintenance Team.
BPs in Division C	Software-1 SDLC, Division C Specific Input, Software-1 Maintenance Team.
BPs in Division D	Software-2 SDLC, Division D Specific Input, Software-2 Maintenance Team.
BPs in Division A&C	Software-1 SDLC, Software-1 Maintenance Team.
BPs in Division B&D	Software-2 SDLC, Software-2 Maintenance Team.
All BPs	Software Requirements, Function, partially shared SDLC practices.

The next step from Figure 6 is to define the model parameters. Each CCCG is assessed for how well it is defended against CCF according to the subfactors. This case study is addressing only the software CCF; therefore, only the new subfactor tables and scoring methods will be required. This section will detail the analysis for the CCF of BP1 from Division A (A1, for convenience). BP A1 belongs to three CCCGs based on similarities shared with the software of Division A, Divisions A&C, and all BPs. Table 13, Table 14, and Table 15 provide the subfactor scores which are used to define the defense factors for each CCCG. Equations (10)–(14) are used to find the independent and CCFs of the BPs. The results of the CCF analysis are shown in Table 16, Table 17, and Table 18.

Table 13. Subfactor scores for CCCG1 of the diverse RTS.

Bistable Processors: RTS Diversity Configuration 1		
CCCG 1: BPs (A1, A2) Software-1		
Subfactor	Score	Reason
Input Similarity	A	No separation of input variable type or source. (Assumed the same variable data for both in the same division from division-specific sensors.) Calculation of Subfactor: $m = 2$, the number of components within the CCCG, $i = 2/m = 1$, “inputs/sink” = (number of inputs to the CCCG)/m $c = 1$, the number of immediate sources. $s = c/i = 1/1 = 1$, For scenario when $s = 1$ $R = (s - 1)/m = 0$
Understanding	D	**1. Novelty**: (Not novel). The design of the CCCG is not novel or interesting in any way. Just a simple redundancy relationship. **2. Complexity:** (Not complex). CCCG members perform only single function. **3. Misfit:** (No misfit) Software of the CCCG is purpose built. **4. Experience:** (Low) There is not more than 10 years of experience with the performance of this CCCG and its software.
Analysis	B	**Analysis:** General level of analysis was performed for this CCCG, but no detailed FT analysis. As mentioned in the assumptions, the FT only considered a CCF for when all components that share a software fail (e.g., this corresponds to CCF of Divisions A&C). **Feedback:** The assumptions section indicates that only casual feedback was performed.

		Awareness: Designer awareness is given as minimal. No formal indication of designer awareness is provided. Therefore, design details must be considered. The design of the CCCG is of minimal redundancy. Score [(A&F) AND minimal awareness]
MMI	C	**Operator/User interaction level for CCCG?** (Low interaction) operator is assumed to have minimal interaction with the components of a trip system. **Operator/User procedures for interaction?** (Written procedures) All operator interaction is prescribed by written procedures due to the safety significance of the system. **How is maintenance interaction governed for this CCCG?** (Test and checked). Maintenance follows test and checking process. **Operator interaction** = Level 2 **Maintenance**= Level 3 The pessimistic of the two (i.e., Level 2) is selected according to the guidance. The associated score is C.
Safety Culture &Training	D	**Team 1** has regular safety training, safety oriented culture, and specialized education (Also, because this is a new system, experience cannot be credited. In the same way that experience is not credited for the Understanding subfactor. We cannot select E, so we must select D instead.)
Control	D	The CCCG is in a secure physical location. Access to the CCCG is limited to authorized and trained personnel. Access to the area is not limited to individuals working on this one piece of software. Other devices and systems are present.
Tests	D	The RTS will be tested when installed. This includes division level testing and whole system testing. Criteria "D" encompasses testing in place. Criteria "E" requires testing the CCCG in parallel with an existing system before being brought online; this has not occurred.

Table 14. Subfactor scores for CCCG2 of the diverse RTS.

Bistable Processors: RTS Diversity Configuration 1		
CCCG 2: BPs (A1, A2, C1, C2) Software-1		
Subfactor	Score	Reason
Input Similarity	A+	No separation of input variable type or source. (Assumed the same variable data for both in the same division from division-specific sensors) Calculation of Subfactor: $m = 4$, the number of components within the CCCG, $i = 4/m = 1$, "inputs/sink" = (number of inputs to the CCCG)/m $c = 2$, the number of immediate sources. $s = c/i = 2/1 = 2$, For scenario when $s > 1$ $R = s/m = 1/2$

Understanding	D	**Novelty**: (Not novel). The design of the CCCG is not novel or interesting in any way. Just a simple redundancy relationship. **Complexity:** (Not complex). CCCG members perform only single function. **Misfit:** (No misfit) Software of the CCCG is purpose built. **Experience:** (Low) There is not more than ten years of experience with the performance of this CCCG or its software.
Analysis	C	**Analysis:** A detailed analysis was performed for this CCCG, which includes a FT analysis. As mentioned in the assumptions, the FT considered a CCF for when all components that share a software fail (e.g., A&C division BPs share software 1) **Feedback:** the assumptions section indicates that only casual feedback was performed. **Awareness:** the designer has general awareness as indicated by the design. This CCCG has redundancy greater than two. This indicates designers have a general knowledge of CCF prevention. Score [(A+&F) AND General Awareness]
MMI	C	**Operator/User interaction level for CCCG?** (Low interaction) operator is assumed to have minimal interaction with the components of a trip system. **Operator/User procedures for interaction?** (Written procedures) All operator interaction is prescribed by written procedures. Due to safety significance. **How is maintenance interaction governed for this CCCG?** (Test and checked). Maintenance follows test and checking process. **Operator interaction** = level 2 **Maintenance**= level 3 The pessimistic of the two (i.e., Level 2) is selected according to the guidance. The associated score is C.
Safety Culture &Training	D	**Team 1** has regular safety training, safety oriented culture, and specialized education (Also, because this is a new system, experience cannot be credited. In the same way that experience is not credited for the Understanding subfactor. We cannot select E, so we must select D instead.)
Control	D	The components of the CCCG are in secure physical locations. Access to the CCCG is limited to authorized and trained personnel. Access to the area is not limited to individuals working on this one piece of software. Other devices and systems are present.
Tests	D	The RTS will be tested when installed. This includes division level testing and whole system testing. Criteria “D” encompasses testing in place. Criteria “E” requires testing the CCCG in parallel with an existing system before being brought online; this has not occurred. Because the analysis recognized and modeled this CCCG as a risk. Testing would certainly be performed for this group.

Table 15. Subfactor scores for CCCG3 of the diverse RTS

Bistable Processors: RTS Diversity Configuration 1		
CCCG 2: BPs (A1, A2, B1, B2, C1, C2, D1, D2) Software: 1 & 2		
Subfactor	Score	Reason
Input Similarity	A+	No separation of input variable type or source. (Assumed the same variable data for both in the same division from division-specific sensors) Calculation of Subfactor: $m = 8$, the number of components within the CCCG, $i = 8/m = 1$, “inputs/sink” = (number of inputs to the CCCG)/m $c = 4$, the number of immediate sources. $s = c/i = 4/1 = 4$, For scenario when $s > 1$ $R = s/m = 1/2$
Understanding	D	**Novelty**: (not novel). The design of the CCCG is not novel or interesting in any way. Just a simple redundancy relationship. **Complexity:** (Not complex). CCCG members perform only single function. **Misfit:** (No misfit) Software of the CCCG is purpose built. **Experience:** (low) There is not more than ten years of experience with the performance of this CCCG or its software.
Analysis	D	**Analysis:** A general level of analysis was performed for this CCCG, but no detailed FT analysis. As mentioned in the assumptions, the FT only considered a CCF for when all components that share a software fail (e.g., this corresponds to CCF of Divisions A&C.) Additionally, the analysists Analysis assumes diversity prevents CCF for those CCCGs where it is employed. **Feedback:** the assumptions section indicates that only casual feedback was performed. **Awareness:** the designer has specific awareness as indicated by the design. This CCCG has diversity employed. This indicates designers have a specific knowledge of CCF prevention. Score [(A&F) AND Specific Awareness]
MMI	C	This CCCG has shared operators but multiple maintenance teams. Both teams have the same rules for interactions with the CCCG. **Operator/User interaction level for CCCG?** (Low interaction) operator is assumed to have minimal interaction with the components of a trip system. **Operator/User procedures for interaction?** (Written procedures) All operator interaction is prescribed by written procedures. Due to safety significance. **How is maintenance interaction governed for this CCCG?** (Test and checked). Maintenance follows test and checking process. **Operator interaction** = level 2 **Maintenance**= level 3 The pessimistic of the two (i.e., Level 2) is selected according to the guidance. The associated score is C.

Safety Culture &Training	D	**Team 1** has regular safety training, safety oriented culture, and specialized education (Also, because this is a new system, experience cannot be credited. In the same way that experience is not credited for the Understanding subfactor. we cannot select E, so we must select D instead.) **Team 2** has infrequent safety training, safety oriented culture, and general education (C). In a CCCG with multiple components. We are modeling only the CCF for all components to fail in the CCCG. Thus, we take credit for the better of the MMI/safety culture scores. The basis for this assumption is that the CCF model employed assumed entire CCCG fails with a CCF. Diversity of the score of this subfactor prevents or reduces the CCF. For this case the better of the scores is D.
Control	D	The components of the CCCG are in secure physical locations. Access to the CCCG is limited to authorized and trained personnel. Access to the area is not limited to individuals working on this one piece of software. Other devices and systems are present.
Tests	D	The RTS will be tested when installed. This includes division level testing and whole system testing. Criteria "D" encompasses testing in place. Criteria "E" requires testing the CCCG in parallel with an existing system before being brought online; this has not occurred.

Table 16. Subfactor scores summary diversity configuration (to find CCF of BP A1).

Subfactor scores summary: Diversity Configuration 1 (**to find CCF that involves A1)**			
Subfactors	**Group 1: A1, A2**	**Group 2: A1, A2, C1, C2**	**Group 3: All BPs**
Input Similarity	A	A+	A+
Understanding	D	D	D
Analysis	B	C	D
MMI	C	C	C
Safety Culture	D	D	D
Control	D	D	D
Testing	D	D	D
Defense Factor:	0.34153611	0.14373557	0.14100031
CCFs	7.094E-05	2.985E-05	1.464E-05

Table 17. Subfactor scores summary diversity configuration (to find CCF of BP B1).

Subfactor scores summary: Diversity Configuration 1 (**to find CCF that involves B1)**			
Subfactors	**Group 1: B1, B2**	**Group 2: B1, B2, D1, D2**	**Group 3: All BPs (Unchanged)**
Input Similarity	A	A+	A+
Understanding	D	D	D
Analysis	B	C	D
MMI	C	C	C
Safety Culture	C	C	D

Control	D	D	D
Testing	D	D	D
Defense Factor:	0.34392946	0.14612892	0.14100031
CCFs	1.250E-04	5.311E-05	1.464E-05

Table 18. CCF analysis results for BPs of the diverse RTS.

Diversity Configuration 1					
Component	IND	CCCG1	CCCG2	CCCG3	Total
BPs-Divisions A&C	9.227E-05	7.094E-05	2.985E-05	1.464E-05	2.077E-04
BPs-Divisions B&D	1.707E-04	1.250E-04	5.311E-05	1.464E-05	3.635E-04

3.6 Discussion

The results of the case study show that as anticipated for the CCCGs that have diversity, the CCF probability is reduced when compared to the CCCGs without diversity. (CCCG1, CCCG2 do not have diversity and they have a higher failure probability than CCCG3). Some additional comparisons can be made with the baseline model. Table 19 shows the baseline CCF results. No change has occurred for CCCG1 for those BPs that are employing Software-1 (i.e., Divisions A&C). As for the CCCG1 that employed Software-2, the CCF is increased, because Software-2 is has a higher failure probability than Software-1. CCCG2 is not applicable for the baseline case, because there was no distinguishing coupling mechanism to form this group; whereas CCCG2 can be formed for the diversity configuration of the RTS, because its features are distinguished from CCCG3. CCCG3 is the interesting case because this is where the benefit of diversity is visualized. The CCF of CCCG3 is reduced as a result of implementing diversity. The benefits of this reduction will be investigated in the remaining sections of the report.

Table 19. CCF analysis results for BPs of the baseline RTS.

Baseline Configuration					
Component	IND	CCCG1	CCCG2	CCCG3	Total
BPs	1.069E-04	7.094E-05	Not Applicable	2.985E-5	2.077E-04

This section introduced an approach for modeling software-based CCFs. A primary motivation of the method is to provide insights under limited data conditions. Most of existing CCF methods rely on historical data (i.e., operating experience), to define model parameters; however, the absence of software failure data, such as for novel designs, demands an alternative. There are two options for assessing CCFs under limited data condition. The first option employs conservative or bounding assessments (e.g., probability of CCF is equal 1.0). The second option relies on expert judgment. The proposed framework informs the judgment process by employing subfactors (e.g., analysis, training, testing and safety culture) to determine strategically and qualitatively how well a CCCG is defended against potential CCFs. The works by Humphreys [16] and Brand [17] which have been used by the IEC [18] for hardware-based CCF modeling, serve as the technical foundation for implementing the chosen subfactors as part of this work for software-based CCF modeling. Due to the lack of software-based CCF data, the thorough verification and validation of this CCF modeling and estimation method has not yet been performed. A detailed guideline of the method is provided in this report to reduce subjectivity caused by user's interpretation. It is advised that a group of experts, rather than a single expert, work on the scoring process. Even with uncertainty and subjectivity from user effect, quantification of important software-based CCFs can be very useful for DI&C designers to make informed decisions on diversity design and optimization, and help reduce the cost of system development. For example, quantified metrics may demonstrate that System A is more robust against software CCFs compared to System B. In this case, the comparison is objective since the same approach is used to evaluate both systems. Future collaborations with industry partners may afford our team the opportunity to validate and improve this method.

4. SENSITIVITY AND IMPORTANCE ANALYSES FOR DIVERSE AND REDUNDANT SYSTEMS

This section presents the results of sensitivity and importance analyses conducted for different designs of RTS and ESFAS. The sensitivity analysis is needed to examine the variation in risk from added diversity in the system design. The sensitivity analysis results can be used to determine if an investment in diversity is worthwhile to achieve risk reduction. In this study, diversity is introduce via different software used for the BPs, which are shared by RTS and ESFAS. For the baseline design, all BPs are assumed to use the same software, i.e., all divisions use software-1 as shown in Figure 2. For the improved diverse RTS design, it is assumed that the BPs in four divisions uses two different software, i.e., divisions A&C uses software-1 and divisions B&D uses software-2 as shown in Figure 3. The impacts of diversity were evaluated at both system-risk level (measured using system failure probability) and plant-risk level (measured using core damage frequency).

The importance analysis identifies critical equipment that are risk drivers. The importance analysis results can be leveraged to understand which equipment are worth investing to improve their reliability and consequently overall system risks (i.e., informing the decision making of resource prioritization and allocation). The importance analyses were performed at system-risk level (measuring using system failure probability) for both non-diverse and diverse system designs in order to understand how diversity applications can impact the equipment importance.

4.1 System Level Sensitivity Analyses for Reactor Trip Systems and Engineered Safety Features Actuation Systems

4.1.1 Reactor Trip Systems

This section calculates system-level failure probabilities of improved non-diverse design and improved diverse design for RTS.

It can be observed from Table 20 that by introducing BP software diversity, the RTS failure probability is reduced by 5%. The reduction becomes more significant if focusing on the automatic actuation failure only, which decreases by 9% as shown in Table 21. This is because the RTS function can be either achieved by automatic actuation or by manual actuation, and the software is involved in automatic actuation only.

Table 22 presents the dominant cut sets of RTS failure for baseline design and diverse design. For both designs, hardware CCF of rod cluster control assembly (RCCAs) to drop is the most dominant cut set. Each of the other dominant cut sets require the concurrence of an operator/human system interface (HIS) failure and a hardware or software CCF. By introducing BP software diversity, the cut set with concurrent operator/HIS failure and BP software CCF is reduced by 51% from 2.985E-07 to 1.464E-07.

The above results suggest that the introduction of BP software diversity can improve RTS reliability. It can reduce BP software CCF probability, reduce RTS automatic actuation failure probability, and thus reduce the failure probability of the entire RTS. The extent of failure probability reduction is expected to be larger when expanding the scope of diversity such as by adopting different software for each division, introducing diversity to BP hardware, introducing diversity to LP hardware and/or software.

Table 20. Failure probabilities of different RTS designs.

FT Name	RTS System Failure Probability	Δ / Non-Diverse	# of Cut Sets
RTS w/ Component Diversity	2.920E-06	−5%	123
RTS w/o Component Diversity	3.086E-06	0%	114

Table 21. Failure probabilities of automatic actuation in different RTS designs.

FT Name	RTS Automatic Actuation Failure Probability	Δ / Non-Diverse	# of Cut Sets
RTS w/ Component Diversity	1.566E-04	−9%	55
RTS w/o Component Diversity	1.718E-04	0%	44

Table 22. Dominant top cut sets with greater than-1% contribution of different RTS designs.

FT Name	#	Cut Set Probability	Total %	Cut Set	
RTS w/ Component Diversity	1	1.210E-06	41.44	RPS-ROD-CF-RCCAS	Hardware CCF of 10 or more rod cluster control assembly (RCCAs) to drop
	2	1.179E-06	40.37	LC-LP-SF-CCF-TA	Software CCF of all LPs to provide trip commands to DOMs
				RPS-XHE-XE-SIGNL	Operator/human system interface (HSI) fails to respond with reactor protection system (RPS) signal present
	3	1.763E-07	6.04	RTB-UV-HD-CCF	Hardware CCF of undervoltage trip mechanisms of all reactor trip breakers (RTBs)
				RPS-XHE-XE-SIGNL	Operator/HSI fails to respond with RPS signal present
	4	1.464E-07	5.01	LC-BP-UCA-A-CCF	Software CCF of all BPs to provide a trip command to each division's LCL cabinet when needed
				RPS-XHE-XE-SIGNL	Operator/HSI fails to respond with RPS signal present
	5	3.961E-08	1.36	LP-HW-CCF	Hardware CCF of all LPs
				RPS-XHE-XE-SIGNL	Operator/HSI fails to respond with RPS signal present
(Blank row for separation)					
RTS w/o Component Diversity	1	1.210E-06	39.21	RPS-ROD-CF-RCCAS	Hardware CCF of 10 or more RCCAs to drop
	2	1.179E-06	38.20	LC-LP-SF-CCF-TA	Software CCF of all LPs to provide trip commands to DOMs
				RPS-XHE-XE-SIGNL	Operator/HSI fails to respond with RPS signal present
	3	2.985E-07	9.67	LC-BP-UCA-A-CCF	Software CCF of all BPs to provide a trip command to each division's LCL cabinet when needed
				RPS-XHE-XE-SIGNL	Operator/HSI fails to respond with RPS signal present
	4	1.763E-07	5.71	RTB-UV-HD-CCF	Hardware CCF of undervoltage trip mechanisms of all RTBs
				RPS-XHE-XE-SIGNAL	Operator/HSI fails to respond with RPS signal present
	5	3.961E-08	1.28	LP-HW-CCF	Hardware CCF of all LPs

FT Name	#	Cut Set Probability	Total %	Cut Set	
				RPS-XHE-XE-SIGNL	Operator/HSI fails to respond with RPS signal present

4.1.2 Engineered Safety Features Actuation Systems

This section calculates system-level failure probabilities of improved non-diverse design and improved diverse design for ESFAS. It needs to be re-emphasized here that the BPs used in ESFAS are the same as those used in RTS. The LPs used in ESFAS in RTS are dedicated to one system only and are modeled as distinct basic events in the SAPHIRE model, although the same basic event values are used.

It can be observed from Table 23 that by introducing BP software diversity, the ESFAS failure probability is reduced by 5%. In the current SAPHIRE model, only automatic action of ESFAS failure is modeled. So the 5% reduction represents the reduction of ESFAS automatic actuation failure probability, which is smaller than the 9% reduction of the corresponding RTS automatic actuation failure probability.

Table 24 presents the dominant cut sets of ESFAS failure for baseline design and diverse design. For both designs, each of the dominant cut sets is a hardware or software CCF. By introducing BP software diversity, the cut set BP software CCF is reduced by 51% from 2.985E-05 to 1.464E-05.

The above results suggest that the introduction of BP software diversity can improve ESFAS reliability. It can reduce BP software CCF probability and thus reduce ESFAS failure probability. Based on the results from this section and Section 4.1.1, it can be concluded that introducing BP software diversity can simultaneously reduce RTS failure probability and ESFAS failure probability, each by 5%.

Table 23. Failure probabilities of different ESFAS designs.

FT Name	ESFAS System Failure Probability	Δ / Non-Diverse	# of Cut Sets
ESFAS w/ Component Diversity	3.028E-04	−5%	22
ESFAS w/o Component Diversity	3.180E-04	0%	11

Table 24. Dominant top cut sets with greater than-1% contribution of different ESFAS system designs.

FT Name	#	Cut Set Probability	Total %	Cut Set	
ESFAS w/ Component Diversity	1	1.179E-04	38.93	LP-UCA-A-CCF	Software CCF of all LCL processors in all divisions to provide command when it is needed
	2	8.914E-05	29.44	ESF-CCS-GC-UCA-A-CCF	Software CCF of all divisions of ESF-CCS GC processors to provide actuation signal when it is needed
	3	6.842E-05	22.59	ESF-CCS-LC-UCA-A-CCF	Software CCF of all divisions of ESF-CCS LC processors to provide actuation signal
	4	1.464E-05	4.83	LC-BP-UCA-A-CCF	Software CCF of all BPs to provide a trip command to each division's LCL cabinet when needed
	5	3.961E-06	1.31	LP-HW-CCF	Hardware CCF of all LCL processors
(Blank row for separation)					

FT Name	#	Cut Set Probability	Total %	Cut Set	
ESFAS w/o Component Diversity	1	1.179E-04	37.07	LP-UCA-A-CCF	Software CCF of all LCL processors in all divisions to provide command when it is needed
	2	8.914E-05	28.03	ESF-CCS-GC-UCA-A-CCF	Software CCF of all divisions of ESF-CCS GC processors to provide actuation signal when it is needed
	3	6.842E-05	21.51	ESF-CCS-LC-UCA-A-CCF	Software CCF of all divisions of ESF-CCS LC processors to provide actuation signal
	4	2.985E-05	9.39	LC-BP-UCA-A-CCF	Software CCF of all BPs to provide a trip command to each division's LCL cabinet when needed
	5	3.961E-06	1.25	LP-HW-CCF	Hardware CCF of all LCL processors

4.2 Plant Level Sensitivity Analyses for Reactor Trip Systems and Engineered Safety Features Actuation Systems

This section compares ET quantification results of three groups of accident scenarios, including general plant transient (TRANS), small-break loss-of-coolant accident (SBLOCA), and medium-break loss-of-coolant accident (MLOCA), of non-diverse designs and diverse designs for the RTS and ESFAS.

It should be that, in this section, "diverse design" refers to adopting BP software diversity for both RTS and ESFAS and "non-diverse design" refers to not adopting any diversity for RTS or ESFAS. Therefore, the analyses in this section are based upon the impacts of BP software diversity of RTS and ESFAS as a whole, examining their downstream impacts on the risks from different accident scenarios.

4.2.1 General Plant Transient Scenarios

The ETs representing general plant transient scenarios are shown in Figure 19 through Figure 21 in Appendix A. The ETs were quantified with SAPHIRE 8 using a truncation level of 1E-12. Table 25 compares the quantified CDF of non-diverse design with the CDF of diverse design. By introducing BP software diversity, the total CDF from general plant transient is reduced by 4% from 9.0E-07 per reactor year to 8.7E-07 per reactor year. For each of non-diverse and diverse design, there are 145 INT-TRANS sequences ending in core damage and 16 of them have non-zero (i.e., non-truncated) frequencies. It can be observed from Table 26 that the introduction of BP software diversity only impacts 3 non-zero core damage sequences, including sequence INT-TRANS:21-16, INT-TRANS:21-14, and INT-TRANS:21-15, and reduces their frequencies by 5%, 5%, and 42%, respectively.

Sequence INT-TRANS:21-16 reaches core damage because of post-ATWS depressurization of reactor coolant system (RCS). Sequence INT-TRANS:21-15 reaches core damage for post-ATWS failures of both main feedwater system and auxiliary feedwater system. For sequence INT-TRANS:21-14, post-ATWS RCS depressurization fails but feedwater is available from auxiliary feedwater system; however, core damage still occurs due to loss of emergency boration. It can be concluded that all the three sequences are ATWS scenarios with RTS failure, and only sequence TRANS:21-15 involves both RTS failure and ESFAS failure. These sequence descriptions explain from a physical perspective that only the sequences involving RTS failure and/or ESFAS failure will be impacted by the introduction of BP software diversity. Also, the sequence(s) involving both RTS and ESFAS failures will be subject to a larger impact than those involving only one of these two system failures. Sequence INT-TRANS:21-15 is a scenario that involves both system failures, the frequency of which is significantly reduced by 42% from 2.82E-08 per reactor year to 1.64E-08 per reactor year; however, the contribution of this sequence to the total CDF from general plant transient is very low, i.e., 3% for non-diverse design and 2% for diverse design, therefore its downstream impact on the total CDF from general plant transient appears trivial.

Table 25. Comparison of general plant transient event tree quantification results for diverse and non-diverse designs.

Sequence	CDF			# of Cut Sets	
	Non-Diverse	Diverse	Δ CDF/ Non-Diverse CDF	Non-Diverse	Diverse
INT-TRANS:21-16	3.88E-07	3.67E-07	−5%	566	564
INT-TRANS:20	3.64E-07	3.62E-07	0%	1264	1262
INT-TRANS:02-02-09	5.83E-08	5.83E-08	0%	1248	1248
INT-TRANS:21-14	5.17E-08	4.91E-08	−5%	95	95
INT-TRANS:21-15	2.82E-08	1.64E-08	−42%	128	121
INT-TRANS:19	7.44E-09	7.41E-09	0%	288	288
INT-TRANS:02-03-09	2.73E-09	2.73E-09	0%	387	387
INT-TRANS:02-02-10	9.55E-10	9.55E-10	0%	168	168
INT-TRANS:02-04-10	5.87E-10	5.87E-10	0%	142	142
INT-TRANS:02-14-10	1.99E-10	1.99E-10	0%	81	81
INT-TRANS:02-03-10	7.65E-12	7.65E-12	0%	4	4
INT-TRANS:02-09-09	7.56E-12	7.56E-12	0%	4	4
INT-TRANS:02-06-09	7.56E-12	7.56E-12	0%	4	4
INT-TRANS:02-08-09	7.56E-12	7.56E-12	0%	4	4
INT-TRANS:02-07-09	2.29E-12	2.29E-12	0%	2	2
INT-TRANS:02-10-09	2.29E-12	2.29E-12	0%	2	2
Total	9.016E-07	8.650E-07	−4%	4387	4376

4.2.2 Small Break Loss of Coolant Accident Scenarios

The ET representing small-break loss-of-coolant accident scenarios is shown in Figure 22 in Appendix A. The ET was quantified with SAPHIRE 8 using a truncation level of 1E-12. Table 26 compares the quantified CDF of non-diverse design with the CDF of diverse design. By introducing BP software diversity, the total CDF from small-break LOCA is reduced by only 0.2% from 7.66-09 per reactor year to 7.65E-08 per reactor year. This is because the SLOCA sequences involving RTS failure (sequence INT-SLOCA:19) or ESFAS failure (sequence INT-SLOCA:17) are not dominant with regard to the total CDF from the SLOCA scenario.

Table 26. Comparison of small-break loss-of-coolant accident event tree quantification results for diverse and non-diverse designs.

Sequence	CDF			# of Cut Sets	
	Non-Diverse	Diverse	Δ CDF/ Non-Diverse CDF	Non-Diverse	Diverse
INT-SLOCA:03	6.432E-08	6.423E-08	−0.1%	564	564
INT-SLOCA:09	6.998-09	6.980E-09	−0.3%	150	150
INT-SLOCA:05	2.876E-09	2.876E-09	0.0%	97	97
INT-SLOCA:19	1.832E-09	1.736E-09	−5.2%	21	21
INT-SLOCA:18	4.502E-10	4.470E-10	−0.7%	34	34
INT-SLOCA:10	1.283E-10	1.283E-10	0.0%	7	7

INT-SLOCA:17	1.462E-11	1.246E-11	−14.8%	2	2
Total	7.661E-8	7.649E-08	−0.2%	875	875

4.2.3 Medium Break Loss-of-Coolant Accident Scenarios

The ET representing medium-break loss-of-coolant accident scenarios is shown in Figure 23 in Appendix A. The ET was quantified with SAPHIRE 8 using a truncation level of 1E-12. Table 27 compares the quantified CDF of non-diverse design with the CDF of diverse design. By introducing BP software diversity, the total CDF from medium-break LOCA is reduced by only 0.6% from 5.62E-08 per reactor year to 5.60E-08 per reactor year. This is because the MLOCA sequences involving RTS failure (sequence INT-MLOCA:14) or ESFAS failure (sequences INT-MLOCA:10,11,12) are not dominant with regard to the total CDF from MLOCA scenario.

Table 27. Comparison of medium-break loss-of-coolant accident event tree quantification results for diverse and non-diverse designs.

Sequence	CDF			# of Cut Sets	
	Non-Diverse	Diverse	Δ CDF/ Non-Diverse CDF	Non-Diverse	Diverse
INT-MLOCA:03	4.917E-07	4.917E-07	0.0%	722	722
INT-MLOCA:10	6.575E-08	6.271E-08	−4.6%	55	55
INT-MLOCA:05	1.870E-09	1.870E-09	0.0%	47	47
INT-MLOCA:09	1.866E-09	1.866E-09	0.0%	192	192
INT-MLOCA:14	6.085E-10	5.753E-10	−5.5%	18	16
INT-MLOCA:12	2.918E-10	2.797E-10	−4.1%	24	24
INT-MLOCA:11	2.614E-10	2.492E-10	−4.7%	11	11
INT-MLOCA:07	8.999E-11	8.999E-11	0.0%	26	26
Total	5.624E-07	5.593E-07	−0.6%	1095	1093

4.3 System Level Importance Analyses for Reactor Trip Systems and Engineered Safety Features Actuation Systems

This section calculates system-level importance measures of improved non-diverse designs and improved diverse designs for the RTS and the ESFAS. In this section, the importance analysis refers to an analysis that utilizes a PRA model to measure impact of model inputs on total risk and quantifies the impacts from separated factors on total risk; in other words, the importance analysis examines the impacts of individual factors one at a time. The components with relatively high importance measures are suggested as worth-watching candidates for different purposes such as prioritizing investments to make design changes and increase system reliability.

In practice, the importance measures are usually calculated at CDF level. However, as informed by the sensitivity analysis results in Section 4.2, the impact of introducing BP software diversity on CDF values from different scenarios are not significant. Hence, the importance measures in this section are calculated at a system level.

The importance measures are calculated using SAPHIRE. SAPHIRE provides seven different basic event importance measures, which can be categorized in three types: (1) ratio importance measures, including Fussell-Vesely Importance (FV), Risk Reduction Ratio, and Risk Increase Ratio; (2) interval importance measures, including Birnbaum Importance (Birnbaum), Risk Reduction Interval, and Risk Increase Interval; (3) uncertainty importance [29]. The current SAPHIRE model for the DI&C systems under this project hasn’t included uncertainty analysis yet, so uncertainty importance measure is not applicable at this moment. Two common measures, FV and Birnbaum are selected for this study to represent ratio-type measures and interval-type measures, respectively.

The mathematical equations of these two measures are not provided in this section and can be found in [29]. Conceptually, FV measures the overall percent contribution of cut sets containing a basic event of interest to the total risk, which is the RTS or ESFAS failure probability here; Birnbaum measures the rate of change in total risk (again, the RTS or ESFAS failure probability here) as a result of changes to the probability of an individual basic event. A high FV indicates a higher importance, so does a higher Birnbaum. Two commonly used cutoff values, i.e., FV > 0.0005 and Birnbaum > 0.0001, are selected for the analyses in this study.

Table 28 to Table 31 present importance measures of basic events with regard to failure probabilities of different RTS and ESFAS designs. From the results in these tables, it can be observed that the FV trend and the Birnbaum trend do not necessarily agree. In other words, a basic event with a high FV does not necessarily has a high Birnbaum. Taking the results in Table 28 as an example, among all the basic events in the non-diverse RTS FT, the basic event RPS-XHE-XE-SIGNL (operator/HIS fails to respond with RPS signal present) has the highest FV and the CCF event RPS-ROD-CF-RCCAS (CCF of 10 or more RCCAs fail to drop) has the highest Birnbaum. This can be interpreted as that RPS-XHE-XE-SIGNL has a high marginal probability and thus poses high impact on system failure probability. However, RPS-ROD-CF-RCCAS has a high risk (i.e., system failure) sensitivity; this suggests that it can be a good candidate for future investment, since just a little improvement of design and a little reduction of this CCF potential can significantly reduce system failure probability.

By comparing the results in Table 28 and Table 29, it can be observed that the importance measures and basic event rankings for non-diverse RTS and diverse RTS mostly remain the same, with slight changes in FV values and Birnbaum values, as well as the ranking swap of the events RTB-UV-HD-CCF (hardware CCF of undervoltage trip mechanism of all RTBs) and LC-BP-UCA-A-CCF (software CCF of all BPs to provide command to LCL cabinet in all divisions when it is needed). In both RTS designs, RPS-XHE-XE-SIGNL has the highest FV and RPS-ROD-CF-RCCAS has the highest Birnbaum. Four events, including LC-LP-SF-CCF-TA, RTB-UV-HD-CCF, LC-BP-UCA-A-CCF, and LP-HW-CCF, have the same second highest Birnbaum as well as high FV rankings, which can be taken as good candidates for future investment priorities of design improvements.

By comparing the results in Table 30 and Table 31, it can be observed that the importance measures and basic event rankings for non-diverse ESFAS and diverse ESFAS mostly remain the same, only with slight changes in FV values. As a result of introducing BP software diversity, the FV value of LC-BP-UCA-A-CCF (software CCF of all BPs to provide command to LCL cabinet in all divisions when needed) is reduced by 49% from 0.0939 to 0.0483. Another observation is that all the basic events with a non-truncated importance measure have the same Birnbaum value, so the suggestion for future prioritization could be based on FV values only. It should be noted that for both non-diverse ESFAS and diverse ESFAS, the FVs of software CCF events are higher than the FVs of hardware CCF events, suggesting that future investment might be prioritized to reduce software CCF potentials.

Table 28. Importance measures of basic events within fault tree of non-diverse RTS design.

Name	Prob	FV	Birnbaum	Description
RPS-XHE-XE-SIGNL	1.00E-02	5.57E-01	1.72E-04	OPERATOR/HSI FAILS TO RESPOND WITH RPS SIGNAL PRESENT
LC-LP-SF-CCF-TA	1.18E-04	4.17E-01	1.09E-02	SF-CCF: All LCL processors do not provide trip command to DOMs
RPS-ROD-CF-RCCAS	1.21E-06	3.92E-01	1.00E+00	CCF OF 10 OR MORE RCCAS FAIL TO DROP
LC-BP-UCA-A-CCF	2.99E-05	1.06E-01	1.09E-02	SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it is needed
RTB-UV-HD-CCF	1.76E-05	6.24E-02	1.09E-02	Hardware CCF of undervoltage trip mechanism of all RTBs
LP-HW-CCF	3.96E-06	1.40E-02	1.09E-02	Hardware CCF of LCL processors in all divisions
IFD-APS-UIFA	2.37E-04	1.32E-02	1.72E-04	UIF A - Alarms fail to trigger
QND-APS-UIFA	2.37E-04	1.32E-02	1.72E-04	UIF A - Alarms fail to trigger
LC-BP-HW-CCF	2.19E-06	7.74E-03	1.09E-02	Hardware CCF of all BPs
IFD-APS-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of IPS alarm processing system
IFD-CPS-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of CPS
IFD-IFPD-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of IPS IFPD
IFD-SPA-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of SPADES+
QND-APS-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of QIAS-N alarm processing function
QND-PRO-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of QIAS-N processor
QND-SV-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of QIAS-N server
QPD-H	5.00E-05	2.78E-03	1.72E-04	Hardware failure of display monitor; display breaks, cracks, or otherwise fails to display QOIs

* Full names of the acronyms can be found in the Acronyms list on Page ix.

Table 29. Importance measures of basic events within fault tree of diverse RTS design.

Name	Prob	FV	Birnbaum	Description
RPS-XHE-XE-SIGNL	1.00E-02	5.36E-01	1.57E-04	OPERATOR/HSI FAILS TO RESPOND WITH RPS SIGNAL PRESENT
LC-LP-SF-CCF-TA	1.18E-04	4.41E-01	1.09E-02	SF-CCF: All LCL processors do not provide trip command to DOMs
RPS-ROD-CF-RCCAS	1.21E-06	4.14E-01	1.00E+00	CCF OF 10 OR MORE RCCAS FAIL TO DROP
RTB-UV-HD-CCF	1.76E-05	6.59E-02	1.09E-02	Hardware CCF of undervoltage trip mechanism of all RTBs
LC-BP-UCA-A-CCF	1.46E-05	5.48E-02	1.09E-02	SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it is needed
LP-HW-CCF	3.96E-06	1.48E-02	1.09E-02	Hardware CCF of LCL processors in all divisions
IFD-APS-UIFA	2.37E-04	1.27E-02	1.57E-04	UIF A - Alarms fail to trigger
QND-APS-UIFA	2.37E-04	1.27E-02	1.57E-04	UIF A - Alarms fail to trigger
LC-BP-HW-CCF	2.19E-06	8.18E-03	1.09E-02	Hardware CCF of all BPs
IFD-APS-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of IPS alarm processing system
IFD-CPS-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of CPS
IFD-IFPD-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of IPS IFPD
IFD-SPA-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of SPADES+
QND-APS-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of QIAS-N alarm processing function
QND-PRO-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of QIAS-N processor
QND-SV-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of QIAS-N server
QPD-H	5.00E-05	2.68E-03	1.57E-04	Hardware failure of display monitor; display breaks, cracks, or otherwise fails to display QOIs

* Full names of the acronyms can be found in the Acronyms list on Page ix.

Table 30. Importance measures of basic events within fault tree of non-diverse ESFAS design.

Name	Prob	FV	Birnbaum	Description
LP-UCA-A-CCF	1.18E-04	3.71E-01	1.00E+00	SW-CCF: All LCL processors in all divisions fail to provide command when it is needed
ESF-CCS-GC-UCA-A-CCF	8.91E-05	2.80E-01	1.00E+00	SW-CCF:all divisions of ESF-CCS GC processors fail to provide actuation signal when it is needed
ESF-CCS-LC-UCA-A-CCF	6.84E-05	2.15E-01	1.00E+00	SW-CCF: all divisions of ESF-CCS LC processors fail to provide actuation signal
LC-BP-UCA-A-CCF	2.99E-05	9.39E-02	1.00E+00	SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it is needed
LP-HW-CCF	3.96E-06	1.25E-02	1.00E+00	Hardware CCF of LCL processors in all divisions
ESF-CCS-GC-HW-CCF	2.40E-06	7.55E-03	1.00E+00	Hardware CCF of ESF-CCS GC in all divisions
LC-BP-HW-CCF	2.19E-06	6.88E-03	1.00E+00	Hardware CCF of all BPs
CIM-HW-CCF	2.10E-06	6.59E-03	1.00E+00	CIM hardware CCF
ESF-CCS-LC-HW-CCF	2.10E-06	6.59E-03	1.00E+00	ESF-CCS LC hardware CCF in all divisions

* Full names of the acronyms can be found in the Acronyms list on Page ix.

Table 31. Importance measures of basic events within fault tree of diverse ESFAS design.

Name	Prob	FV	Birnbaum	Description
LP-UCA-A-CCF	1.18E-04	3.89E-01	1.00E+00	SW-CCF: all LCL processors in all divisions fail to provide command when it is needed
ESF-CCS-GC-UCA-A-CCF	8.91E-05	2.94E-01	1.00E+00	SW-CCF: all divisions of ESF-CCS GC processors fail to provide actuation signal when it is needed
ESF-CCS-LC-UCA-A-CCF	6.84E-05	2.26E-01	1.00E+00	SW-CCF: all divisions of ESF-CCS LC processors fail to provide actuation signal
LC-BP-UCA-A-CCF	1.46E-05	4.83E-02	1.00E+00	SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it is needed
LP-HW-CCF	3.96E-06	1.31E-02	1.00E+00	Hardware CCF of LCL processors in all divisions
ESF-CCS-GC-HW-CCF	2.40E-06	7.93E-03	1.00E+00	Hardware CCF of ESF-CCS GC in all divisions
LC-BP-HW-CCF	2.19E-06	7.22E-03	1.00E+00	Hardware CCF of all BPs
CIM-HW-CCF	2.10E-06	6.92E-03	1.00E+00	CIM hardware CCF
ESF-CCS-LC-HW-CCF	2.10E-06	6.92E-03	1.00E+00	ESF-CCS LC hardware CCF in all divisions

* Full names of the acronyms can be found in the Acronyms list on Page ix.

4.4 Discussion

This section performs sensitivity and importance analyses for non-diverse and diverse designs of RTS and ESFAS. The findings from the results are summarized below:

By introducing BP software diversity, notable reductions in system-level risks are observed (i.e., RTS failure probability and ESFAS failure probability are both reduced by 5%). However, the reductions in dominant cut sets are found to be substantial (i.e., the probability of cut set with concurrent operator/HIS failure and BP software CCF is reduced by 51%).

By introducing BP software diversity, trivial reductions in plant-level risks are observed (i.e., plant CDFs from general plant transient, small-break LOCA, and medium-break LOCA are reduced by 4%, 0.2%, and 0.6%, respectively).

By introducing BP software diversity, it can be observed that for both RTS and ESFAS, the importance measures and basic event rankings mostly remain the same, except for the FV value of LC-BP-UCA-A-CCF (software CCF of all BPs to provide command to LCL cabinet in all divisions when needed) is reduced by 49%.

For RTS, the events with the highest risk importance are found to be RPS-XHE-XE-SIGNL (operator/HIS fails to respond with RPS signal present) and RPS-ROD-CF-RCCAS (CCF of 10 or more RCCAs fail to drop). This can be interpreted as that RPS-XHE-XE-SIGNL has a high marginal probability and thus poses high impact on system failure probability. However, RPS-ROD-CF-RCCAS has a high risk (i.e., system failure) sensitivity; this suggests that it can be a good candidate for future investment, since just a little improvement of design and a little reduction of this CCF potential can significantly reduce system failure probability.

For ESFAS, the events with the highest risk importance are found to be LP-UCA-A-CCF (software CCF of all LCL processors), ESF-CCS-GC-UCA-A-CCF (software CCF of all group controller processors), and ESF-CCS-LC-UCA-A-CCF (software CCF of all loop controller processors). It can also be observed that for both non-diverse and diverse designs, the FVs of software CCF events are higher than the FVs of hardware CCF events, suggesting that future investment might be prioritized to reduce software CCF potentials.

5. TOP EVENT PREVENTION ANALYSIS

In contrast to the measures of “risk significance” used in Section 4, TEPA is applied in this section to a simplified RTS-FT model, which evaluates the importance of various system components based on the measure of “safety significance”. Safety significance refers to the significance of a contribution to system success probability, while risk significance refers to the significance of a contribution to system failure probability. TEPA can help identify a minimum collection of components, rather than a single component, modeled in the PRA that are effective in managing safety.

5.1. Introduction

TEPA is a method for setting priorities in establishing, maintaining, and assuring the reliability and availability of important system components, based on results of a logic model (e.g., an FT or ET model). Many methods claim to do that, but no method is perfect, and certain claims related to certain methods are invalid. TEPA addresses at least one important matter that some of the other methods address incorrectly or gloss over: the point that system elements work together, not in isolation, and a complete answer to the question, “Which system components must achieve a high level of performance assurance?” must be found. It is suboptimal to assure highly reliable performance of a particular pump if the associated downstream valves are highly unreliable. If there are several functionally redundant pumps in a given system, no one of them will seem “important” based on commonly applied “importance measures,” even if they are important when viewed as a block. TEPA does not choose components based on importance measures of that type; its results reflect system structure at the level of detail appearing in the logic model.

Based on previous Sandia work in Boolean optimization, the earliest work on TEPA itself appeared in [30], [31], [32] (the method was formulated in 1988 but received its present name only in 1995). Industry applications began in earnest after 1995, performed mainly by Blanchard and collaborators [33], [34], [35], [36]. In the course of work on [32], the central relevance of the path set concept to TEPA was appreciated, and [37] argues that safety cases could beneficially be built around the path set structure of scenario models. Ref. [38] proposes an importance measure based on path sets; given a prevention set comprising a particular list of success paths, prevention resources could, in principle, be allocated to SSCs within that set, based on the reliability achieved by path sets containing those SSCs. Finally, [39] illustrates the potential for considering varying levels of basic event prevention and developing prevention sets based on simultaneously applying multiple prevention criteria. Future work on the present problem may be based on the concepts described in [37] and [38].

This section will illustrate some of the above comments by applying TEPA to the reactor trip system-fault tree (RTS-FT) model. The mechanics of TEPA have been described in the references mentioned above and will not be recapitulated here. In this section, we will discuss:

- What questions TEPA answers
- The inputs and outputs of TEPA
- The interpretation and potential application of those outputs
- Some lessons about formulation of the FT model to make the TEPA application meaningful.

5.2. Top Event Prevention Analysis Characteristics

Many nuclear plant systems are redundant, some highly so. There may be many ways for a plant to successfully perform safety functions, which sounds like a good thing; however, if a regulatory safety case is formulated to take credit for many SSCs, then each needs “special treatment” (i.e., extra quality assurance, extra testing, extra inspection). This could turn out to be burdensome. Plant design must take credit for a just-right complement of items—enough to achieve a high level of safety, but not so many SSCs to create an unacceptable burden for the plant. The purpose of TEPA is to assist the user in thinking about what SSCs should be subject to special treatment as a function of the desired level of safety, such as to “satisfy a redundancy criterion,” “keep system failure probability below a target value,” “keep core damage frequency below a target value,” etc. The present version of TEPA works by applying prevention criteria to each minimal cut set, and deriving system- or plant level options by logically combining those results. This is not the same as globally optimizing plant- or system level reliability, but the results of this process are frequently good.

For many problems of practical interest, TEPA generates many options that satisfy the safety target, based on varying complements of systems credited. The user must choose one for implementation. Each option, called a “prevention set,” is a list of basic events corresponding to failure of various SSCs; the appearance of a basic event in a prevention set means that if that option is chosen, some effort may need to be spent to assure prevention of that failure. This entails special treatment of the corresponding SSC. The user’s choice can be based in part on various cost metrics, or proxies for cost, such as “number of basic events whose prevention would be expensive to assure.” It is straightforward to articulate multiple prevention criteria and develop prevention sets that satisfy all of them, but that work is beyond the scope of the present discussion.

Typically, in operating plants, an adequate risk value can be obtained based on only a subset of risk model basic events. For an exceptionally simple design, possibly including certain new designs, this picture could change. The cost savings could be significant if it is practical to base a safety case on a small subset of plant SSCs.

In general, a system does not succeed unless all elements of at least one of its success paths succeed. In the context of safety system operation, a success path entails operation of all of the components required to fulfill the safety function, and all of the support functions that are required by those system components (automatic actuation, cooling, electrical power, instrument air pressure), and all operator actions needed to actuate and regulate system operation. Moreover, for components to function, they (along with their supports) must not have been compromised by earthquake, fire, flood, or malicious attack. Frequently, an analysis will be scoped to exclude these compromises to component function, but in general, a system analysis will need to address support functions.

The concepts applied in the next section are:

- Cut set: A combination of basic events that implies system failure. The events in a cut set are sufficient to cause the top event.
- Minimal cut set: “Minimal” means, if an event is removed from a minimal cut set, the remaining events are no longer sufficient to cause functional failure. The events in a minimal cut set are necessary and sufficient to cause the top event.
- Path set, minimal path set: A path set is a combination of basic events where nonoccurrence implies system success; that is, a minimal path set no longer implies success if one or more events are removed from the set.
- Prevention set, minimal prevention set: A combination of “prevented” basic events (e.g., special treatments) that achieves a level of prevention to satisfy a user-provided risk target value. This “minimal” prevention set no longer satisfies the prevention level if any element of the set is removed. In the current implementation, a prevention set drives all cut sets below a user-supplied value. This is not the same thing as globally allocating performance over model elements; however, it has proven to be a very useful proxy.
- Prevention Criterion: The limit on acceptability of minimal cut set likelihood. The prevention criterion can be qualitative (prevent a number *N* of components in every cut set) or quantitative (drive cut set probability below a given cutoff). When *N* is specified, the resulting prevention sets are said to be “Level N” prevention sets.

Despite the name ("prevention set"), prevention of failure is not absolute; in this context, the term means only that engineering effort is applied to justify a certain amount of credit for element performance. For example, "Level 2 prevention" requires preventing at least two elements of each minimal cut set. This corresponds to the single-failure criterion: there should not be a single event whose failure leads to system failure. This does not mean that components have to be perfect. It means there should be some assurance of component performance, and there should be no single-element cut sets.

Inputs required by TEPA include:

- Minimal cut sets
- At least one prevention criterion to be applied at the cut set level (e.g., prevent at least two events in every minimal cut set; prevent enough basic events to drive every cut set probability below a target value)
- Basic event data needed to quantify the prevention criteria (e.g., if a probabilistic prevention criterion is used, basic event probabilities must be input).

Output of Prevention Analysis:

- Minimal prevention sets satisfying the prevention criteria.

5.3. Application of Top Event Prevention Analysis to the Reactor Trip Systems-Fault Tree Model

This preliminary TEPA study is performed based on a simplified RTS model as shown in Figure 10. To better focus on the automatic control path, MSR/RSR and DPS controls have been removed. The minimal cut sets can be found in Table 32. The basic events that are highlighted in yellow are in Path Set #1.

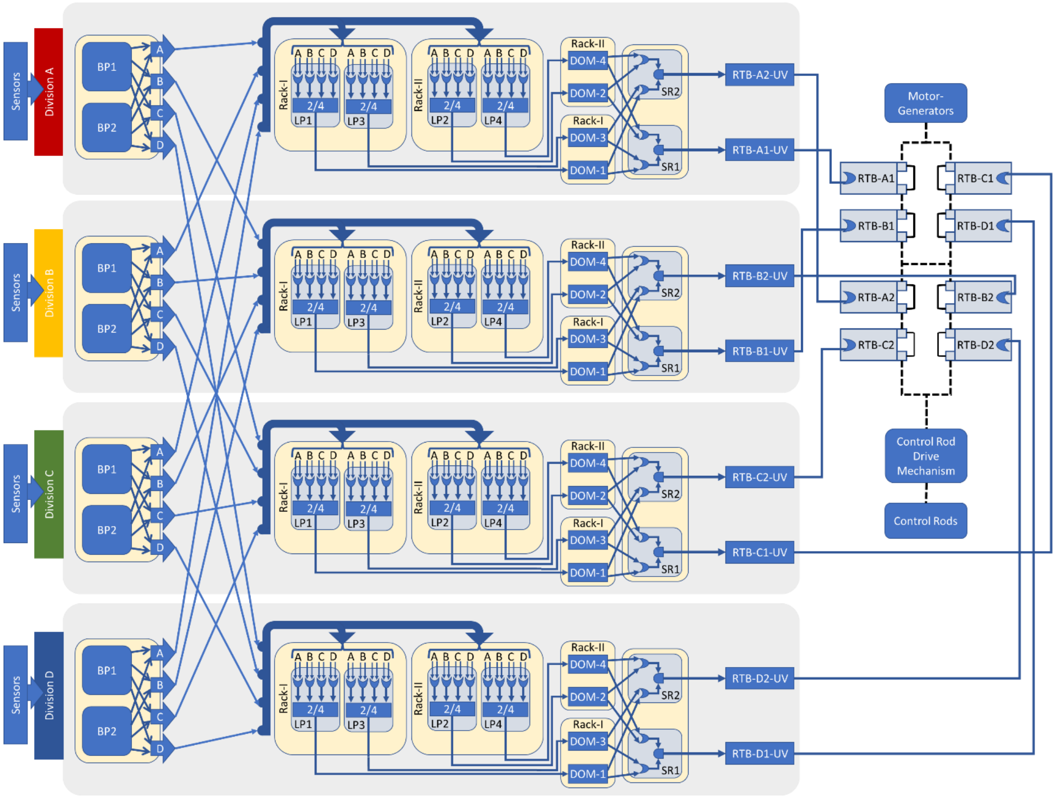


Figure 10. Simplified four-division digital reactor trip system.

Table 32. Minimal cut sets for the RTS model.

TOP =

1. RTB-UV-HD-CCF +
2. SP-HD-CCF +
3. RTB-SYS-1-HD-CCF * RTB-SYS-2-HD-CCF +
4. RTB-A1-UV-HD * RTB-B1-UV-HD * RTB-SYS-2-HD-CCF +
5. DD-SR-HD-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
6. RTB-A1-UV-HD * RTB-A2-UV-HD * RTB-B1-UV-HD * RTB-C2-UV-HD +
7. DA-SR-HD-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
8. RTB-A2-UV-HD * RTB-C2-UV-HD * RTB-SYS-1-HD-CCF +
9. RTB-A2-UV-HD * RTB-C1-UV-HD * RTB-C2-UV-HD * RTB-D1-UV-HD +
10. RTB-C1-UV-HD * RTB-D1-UV-HD * RTB-SYS-2-HD-CCF +
11. DC-SR-HD-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
12. RTB-A1-UV-HD * RTB-B1-UV-HD * RTB-B2-UV-HD * RTB-D2-UV-HD +
13. DB-SR-HD-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
14. RTB-B2-UV-HD * RTB-D2-UV-HD * RTB-SYS-1-HD-CCF +
15. RTB-B2-UV-HD * RTB-C1-UV-HD * RTB-D1-UV-HD * RTB-D2-UV-HD +
16. LP-A-R1-HW-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
17. RTB-B1-UV-HD * RTB-C2-UV-HD * DA-LC-R1-DOM-HD-CCF +
18. LP-A-R2-HW-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
19. RTB-B1-UV-HD * RTB-C2-UV-HD * DA-LC-R2-DOM-HD-CCF +
20. LP-B-R1-HW-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
21. RTB-A1-UV-HD * RTB-D2-UV-HD * DB-LC-R1-DOM-HD-CCF +
22. LP-B-R2-HW-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
23. RTB-A1-UV-HD * RTB-D2-UV-HD * DB-LC-R2-DOM-HD-CCF +
24. LP-C-R1-HW-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
25. RTB-A2-UV-HD * RTB-D1-UV-HD * DC-LC-R1-DOM-HD-CCF +
26. LP-C-R2-HW-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
27. RTB-A2-UV-HD * RTB-D1-UV-HD * DC-LC-R2-DOM-HD-CCF +
28. LP-D-R1-HW-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
29. RTB-B2-UV-HD * RTB-C1-UV-HD * DD-LC-R1-DOM-HD-CCF +
30. LP-D-R2-HW-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
31. RTB-B2-UV-HD * RTB-C1-UV-HD * DD-LC-R2-DOM-HD-CCF +
32. LP-HW-CCF +
33. LC-DOM-HD-CCF +
34. LC-LP-SF-CCF-TA +
35. LP-A-HW-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
36. DA-LC-DOM-HD-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
37. LP-B-HW-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
38. DB-LC-DOM-HD-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
39. LP-C-HW-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
40. DC-LC-DOM-HD-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
41. LP-D-HW-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +

42. DD-LC-DOM-HD-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
43. DA-LC-BP-UCA-A-CCF * DB-LC-BP-UCA-A-CCF * DC-LC-BP-UCA-A-CCF +
44. DA-LC-BP-UCA-A-CCF * DB-LC-BP-UCA-A-CCF * DD-LC-BP-UCA-A-CCF +
45. DA-LC-BP-UCA-A-CCF * DC-LC-BP-UCA-A-CCF * DD-LC-BP-UCA-A-CCF +
46. DB-LC-BP-UCA-A-CCF * DC-LC-BP-UCA-A-CCF * DD-LC-BP-UCA-A-CCF +
47. LC-BP-HW-CCF +
48. LC-BP-UCA-A-CCF

The highlighted basic events are elements of a minimal path set for this model: a combination of basic events whose nonoccurrence (component not failed) implies system success. Since every cut set contains at least one "prevented" basic event, failure of this path set is deemed to be "prevented." All events in single-element cut sets need to be in every path set, and in every prevention set as well.

Table 33 below shows a dictionary of basic events referred to in the present discussion, along with a tabulation of how they map into the present designation of path sets and prevention sets. In some applications, one can get by without renaming events in this fashion, but they have been renamed here for multiple reasons.

1. In the context of the system model, an event name like LP-C-R1-HW-CCF means that a particular CCF has occurred, while its Q-name (Q-13-1H) means that resources have been applied to reduce the probability of that particular event (#13 in Table 33) to a specific level (the significance of "1H" in the Q-name). In practice, context prevents confusion on whether the failure or its prevention is being referred to by a given literal. In principle, a given literal ought to have at most one meaning.

2. The Q-name carries additional information not reflected in the model's basic event name, and is beyond the intended scope of the present discussion: the "1H" corresponds to a particular user-designated level of prevention. (The method is capable of working with degrees of prevention, rather than simply "prevented" or "not prevented.")

3. The multiple computer routines used to generate these results have different limitations on event name lengths. The main Prevention Analysis engine has a 16-character limit on event names, and the Q-nomenclature satisfies that limit.

The second column in Table 33 indicates which events are included in Path Set #1. Referring back to Table 32, we see that in every cut set, at least one basic event is "prevented." We see also that in some cut sets, more than one event is "prevented," which initially sounds like something unnecessary is being prevented. For example, consider cut set #4, which includes RTB-A1-UV-HD * RTB-B1-UV-HD. Since RTB-B1-UV-HD is prevented, can prevention of RTB-A1-UV-HD be dropped? No, because RTB-A1-UV-HD is the only event prevented in cut set #13. If it were dropped, failure of the entire path set would no longer be prevented. Something else would have to be added to address cut set #13, and perhaps other cut sets. Even in this simple model, choosing a minimal path set is not trivial.

Table 33. Mapping from basic event names in the RTS model ("Name") to names in the path sets and prevention sets ("Q-Name"), and designation of events in Path Set 1 and Prevention Set 1.
Notes:
1. An asterisk (*) in a cell under "Prevention Set 1" (Path Set 1) means that the element in that row is contained in the prevention set (Path Set).
2. Event numbering is not consecutive because the model's event data base includes many events that are not in the RTS model; those have been skipped here in order to save space.

Prev Set 1	Path Set 1	Event #	Name	Q-Name
		1	DA-LC-DOM-HD-CCF	Q-1-1H

		2	DB-LC-DOM-HD-CCF	Q-2-1H
		3	DC-LC-DOM-HD-CCF	Q-3-1H
		4	DD-LC-DOM-HD-CCF	Q-4-1H
		9	LP-A-R1-HW-CCF	Q-9-1H
		10	LP-A-R2-HW-CCF	Q-10-1H
		11	LP-B-R1-HW-CCF	Q-11-1H
		12	LP-B-R2-HW-CCF	Q-12-1H
		13	LP-C-R1-HW-CCF	Q-13-1H
		14	LP-C-R2-HW-CCF	Q-14-1H
		15	LP-D-R1-HW-CCF	Q-15-1H
		16	LP-D-R2-HW-CCF	Q-16-1H
		17	DA-LC-BP-UCA-A-CCF	Q-17-1H
*	*	18	DB-LC-BP-UCA-A-CCF	Q-18-1H
*	*	19	DC-LC-BP-UCA-A-CCF	Q-19-1H
*		20	DD-LC-BP-UCA-A-CCF	Q-20-1H
*	*	21	LC-LP-SF-CCF-TA	Q-21-1H
*	*	69	RTB-A1-UV-HD	Q-69-1H
*		70	RTB-B1-UV-HD	Q-70-1H
*		71	RTB-C1-UV-HD	Q-71-1H
*	*	72	RTB-D1-UV-HD	Q-72-1H
*		73	RTB-A2-UV-HD	Q-73-1H
*	*	74	RTB-C2-UV-HD	Q-74-1H
*	*	75	RTB-B2-UV-HD	Q-75-1H
*		76	RTB-D2-UV-HD	Q-76-1H
		77	DA-LC-R1-DOM-HD-CCF	Q-77-1H
		78	DA-LC-R2-DOM-HD-CCF	Q-78-1H
		79	DB-LC-R1-DOM-HD-CCF	Q-79-1H
		80	DB-LC-R2-DOM-HD-CCF	Q-80-1H
		81	DC-LC-R1-DOM-HD-CCF	Q-81-1H
		82	DC-LC-R2-DOM-HD-CCF	Q-82-1H
		83	DD-LC-R1-DOM-HD-CCF	Q-83-1H
		84	DD-LC-R2-DOM-HD-CCF	Q-84-1H
*	*	93	RTB-UV-HD-CCF	Q-93-1H
*	*	94	RTB-SYS-1-HD-CCF	Q-94-1H
*		95	RTB-SYS-2-HD-CCF	Q-95-1H
*	*	96	LC-DOM-HD-CCF	Q-96-1H
*	*	97	LC-BP-HW-CCF	Q-97-1H

*	*	98	LP-HW-CCF	Q-98-1H
		155	DA-SR-HD-CCF	Q-155-1H
		156	DB-SR-HD-CCF	Q-156-1H
		157	DC-SR-HD-CCF	Q-157-1H
		158	DD-SR-HD-CCF	Q-158-1H
*	*	219	SP-HD-CCF	Q-219-1H
		220	LP-A-HW-CCF	Q-220-1H
		221	LP-B-HW-CCF	Q-221-1H
		222	LP-C-HW-CCF	Q-222-1H
		223	LP-D-HW-CCF	Q-223-1H
*	*	224	LC-BP-UCA-A-CCF	Q-224-1H
*				SLACK-1-SH1
*				SLACK-2-SH1
*				SLACK-32-SH1
*				SLACK-33-SH1
*				SLACK-34-SH1
*				SLACK-47-SH1
*				SLACK-48-SH1

* Full names of the acronyms can be found in the Acronyms list on Page ix.

Prevention Set 1, designated in the leftmost column, is visually shown in Table 34 below, with prevention set elements highlighted in green. This run was done with Prevention Level = 2, meaning that at least two events in every cut set should be prevented. One sees in Table 34 that this condition is satisfied for multiple event cut sets. To run single-event cut sets to completion, the algorithm assigns extra elements; these are the last seven entries in Table 33 above. The first “slack variable” entry, SLACK-1-SH1, means that Cut Set 1 was SHort 1 event of satisfying the prevention criterion. If we were running with a prevention level of 3, that slack variable would have been SLACK-1-SH2. The matter of these variables will be taken up in the Section 5.4.

Although the prevention level was 2, we see that some cut sets have more than two events highlighted. The reason for this is analogous to the reason that some path sets have more than one event highlighted.

Table 34. Minimal cut sets with Prevention Set 1 elements highlighted in green.

TOP =
RTB-UV-HD-CCF +
SP-HD-CCF +
RTB-SYS-1-HD-CCF * RTB-SYS-2-HD-CCF +
RTB-A1-UV-HD * RTB-B1-UV-HD * RTB-SYS-2-HD-CCF +
DD-SR-HD-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
RTB-A1-UV-HD * RTB-A2-UV-HD * RTB-B1-UV-HD * RTB-C2-UV-HD +
DA-SR-HD-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
RTB-A2-UV-HD * RTB-C2-UV-HD * RTB-SYS-1-HD-CCF +

RTB-A2-UV-HD * RTB-C1-UV-HD * RTB-C2-UV-HD * RTB-D1-UV-HD +
RTB-C1-UV-HD * RTB-D1-UV-HD * RTB-SYS-2-HD-CCF +
DC-SR-HD-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
RTB-A1-UV-HD * RTB-B1-UV-HD * RTB-B2-UV-HD * RTB-D2-UV-HD +
DB-SR-HD-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
RTB-B2-UV-HD * RTB-D2-UV-HD * RTB-SYS-1-HD-CCF +
RTB-B2-UV-HD * RTB-C1-UV-HD * RTB-D1-UV-HD * RTB-D2-UV-HD +
LP-A-R1-HW-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
RTB-B1-UV-HD * RTB-C2-UV-HD * DA-LC-R1-DOM-HD-CCF +
LP-A-R2-HW-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
RTB-B1-UV-HD * RTB-C2-UV-HD * DA-LC-R2-DOM-HD-CCF +
LP-B-R1-HW-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
RTB-A1-UV-HD * RTB-D2-UV-HD * DB-LC-R1-DOM-HD-CCF +
LP-B-R2-HW-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
RTB-A1-UV-HD * RTB-D2-UV-HD * DB-LC-R2-DOM-HD-CCF +
LP-C-R1-HW-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
RTB-A2-UV-HD * RTB-D1-UV-HD * DC-LC-R1-DOM-HD-CCF +
LP-C-R2-HW-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
RTB-A2-UV-HD * RTB-D1-UV-HD * DC-LC-R2-DOM-HD-CCF +
LP-D-R1-HW-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
RTB-B2-UV-HD * RTB-C1-UV-HD * DD-LC-R1-DOM-HD-CCF +
LP-D-R2-HW-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
RTB-B2-UV-HD * RTB-C1-UV-HD * DD-LC-R2-DOM-HD-CCF +
LP-HW-CCF +
LC-DOM-HD-CCF +
LC-LP-SF-CCF-TA +
LP-A-HW-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
DA-LC-DOM-HD-CCF * RTB-B1-UV-HD * RTB-C2-UV-HD +
LP-B-HW-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
DB-LC-DOM-HD-CCF * RTB-A1-UV-HD * RTB-D2-UV-HD +
LP-C-HW-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
DC-LC-DOM-HD-CCF * RTB-A2-UV-HD * RTB-D1-UV-HD +
LP-D-HW-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
DD-LC-DOM-HD-CCF * RTB-B2-UV-HD * RTB-C1-UV-HD +
DA-LC-BP-UCA-A-CCF * DB-LC-BP-UCA-A-CCF * DC-LC-BP-UCA-A-CCF +
DA-LC-BP-UCA-A-CCF * DB-LC-BP-UCA-A-CCF * DD-LC-BP-UCA-A-CCF +
DA-LC-BP-UCA-A-CCF * DC-LC-BP-UCA-A-CCF * DD-LC-BP-UCA-A-CCF +
DB-LC-BP-UCA-A-CCF * DC-LC-BP-UCA-A-CCF * DD-LC-BP-UCA-A-CCF +
LC-BP-HW-CCF +
LC-BP-UCA-A-CCF

5.4. Prevention Worth

5.4.1. Basic Concept

As a prelude to future work, the “safety significance” of basic events are examined in this analysis. Some workers consider “risk significance” and “safety significance” the same thing. However, it is more useful to note the usual measures of risk significance, Fussell-Vesely and risk achievement worth (RAW), which are grounded in failure space have counterparts in success space. This can be regarded as measuring “safety significance.” The failure space Fussell-Vesely measure for basic event *i* is the probability of the union of minimal cut sets (MCS) containing event *i*, divided by the probability of the union of all of the minimal cut sets. That is,

$$F - V^i = \frac{P\left(\bigcup_j MCS_j^i\right)}{P\left(\bigcup_j MCS_J\right)} \tag{18}$$

For example, an F − V of 0.5 for event *i* means that half of the top event probability is due to cut sets containing event *i*. RAW of event *i* is often defined as risk with that event set to TRUE divided by the baseline risk (top event probability with all events at nominal probabilities). If, for example, risk doubles when event *i* is set to TRUE, then the RAW of event *i* is 2.0:

$$RAW^i = \frac{R_i^+}{R_0} \tag{19}$$

A success space analog of F − V is the prevention worth (PW). Instead of measuring the worth of the minimal cut sets containing failure of component *i*, as the F − V does, PW measures the worth of the minimal path sets containing success of component *i*:

$$PW^i = P\left(\bigcup_j MPS_j^i\right) \tag{20}$$

As defined above, PW is essentially the reliability provided by the selected path sets. Because reliability is ordinarily a number very close to 1.0, the PW as defined above will tend to be close to 1.0, and it is more useful to consider the quantity:

$$-log_{10}\left(1 - P\left(\bigcup_j MPS^j\right)\right) \tag{21}$$

The above mapping of the success probability is called the NINES Index. To see why, suppose that the total success probability (corresponding to the PW value inside the inner parentheses) is 0.999. Then the argument of the log function is 0.001; the negative of that logarithm is 3.0. In other words, the NINES index of *j* indicates the number of nines of success probability provided by the path sets containing event *j*. This index has the added virtue of applying to non-integral numbers of “nines.”

5.4.2. Example

An example derived from the FT analyzed above is provided below. However, the present discussion is intended simply to motivate work currently planned for next FY. The significance of PW is discussed more fully in [38].

Compared to quantification of F − V, quantification of a union of success paths is computationally arduous. In order to quantify F − V, it is usually adequate to quantify each cut set and sum the values.

This works when the basic events are all “rare” (probabilities significantly less than 0.1) so that cut set probabilities are small, and the “rare-event approximation” is a good one. Trying to do PW in that way fails, because each term value in the path set expression is of order unity, and summing the term values will give probabilities greater than unity, a result that is wrong. In theory, one could quantify the path set expression directly by bracketing (i.e., taking the sum of the term probabilities, subtracting off the pair intersection terms, adding in the triple intersection terms, and so on, through many, many steps. For practical problems, that calculation would be difficult.

This finds a preferable, although arduous, approximation: we calculate the *un*reliability of the union of the success paths, and subtract that from unity. To do that, we must reduce the logical path set expression “NOT,” a significant calculation. The difficulty of the calculation has so far prevented its implementation in any software drop-down menu (to our knowledge). But for present purposes, a few manually completed examples are presented below in Table 35.

Table 35. NINES Index of Selected Basic Events.

Fault Tree (FT) Basic Event		Unreliability (UR) of Paths Containing Event	Reliability of Paths Containing Event	NINES Index of Event
Name	Number			
RTB-UV-HD-CCF	93	2.310004E-04	9.997690E-01	3.636387
RTB-B2-UV-HD	75	1.931009E-03	9.980690E-01	2.714216
RTB-SYS-1-HD-CCF	94	2.387894E-04	9.997612E-01	3.621985

RTB-UV-HD-CCF:

It will be recalled from the discussion in earlier subsection that RTB-UV-HD-CCF is a single-element cut set, meaning that its complement is in every path set. Therefore, the failure probability of that union of path sets is identical to the top event probability of the FT being analyzed in this section. The PW of RTB-UV-HD-CCF is an upper bound on the PW of any element in the problem.

RTB-B2-UV-HD:

By definition, since we are examining the path sets containing a given element, that element’s failure will be a single-event cut set when we compute the NOT of the union of those paths. This basic event has one of the highest probabilities in the problem (1.7E-3), which limits the reliability of path sets containing this element. In this particular problem, that leads to this element having a low PW and a low NINES.

RTB-SYS-1-HD-CCF:

This basic event has probability 2E-6, so its appearance as a singleton in the NOT of the paths does not in itself lead to a high unreliability and a low PW.

5.4.3. Discussion of Interpretation of Prevention Worth

It is not possible to combine PW values or NINES index values in any simple way, any more than one can do that for F − V measures of components chosen at random, unless we know that there are no events in common between the values being combined. There can be near-total overlap in the contents of the logic terms giving rise to the values obtained. However, it is well *worth preventing* failure of elements whose success states have high NINES indices, because failure of those elements takes down a safety-significant complement of path sets. RTB-UV-HD-CCF is a prime example of that: its complement appears in every path set. In the model provided, that failure event was assigned a low failure probability, but still dominated top event probability. Making that failure probability higher translates directly into a significantly higher top event probability, hence the name “Prevention Worth.”

Readers familiar with RAW will expect a tendency for elements with high RAW to have high PW. In exceptionally simple models, this correlation can be significant, but the measures are not at all the same. The RAW of an element depends on what that element is in parallel with—PW of an element depends on what that element is in series with. Moreover, as acknowledged in [40], RAW is a crude measure.

The RAW of a component is not a discriminating measure, and it must be interpreted very carefully. While it can be an appropriate measure for assessing a temporary change in which an SSC is to be made unavailable, if it is used in the context of assessing permanent changes, it is an extreme, bounding measure. However, it is commonly used as an intuitive measure of the margin provided by the component.

PW is not affected by the above concern any more than F − V. Both are properties of collections of terms in a logic model, and do not postulate large changes in basic event probability.

As discussed in [38], all single-event importance measures (F − V, RAW, PW, others) depend on what other events are being credited within the analysis. It presently seems likely that PW will be valuable when we are trying to set priorities for event prevention within specific prevention sets; and in that context, the computations needed to quantify PW will already have been partially accomplished during the development of the prevention sets themselves.

5.5. Discussion

The RTS-FT model is a very simplified model of a four-division RTS with only automatic control functionality considered. This simplified system is highly redundant, and its interconnections are complex. The present model is, however, dominated by CCFs, including a handful of single-event CCF cut sets. Taking the cut sets at face value, single-event cut sets clearly must be prevented to some level due to the dominance of model results by single-event CCF cut sets. The reliability performance of various prevention sets varies only minutely as a result of the dominance of the single-event CCFs. Some highly important basic events with high NINES (e.g., RTB-UV-HD-CCF, RTB-SYS-1-HD-CCF) have been identified and suggested to be prevented because it takes down a safety-significant complement of path sets.

For a model as simple as the one used in this illustration, and given the probability assignments, there are no surprises in the prevention analysis. Even for this simple problem, there are hundreds of path sets and hundreds of Level 2 prevention sets.

Interesting exercises for future work include:

1. Computing the reliability performance of a given Prevention Set: The calculations are arduous, and, for the present model, they would be unrewarding because all Prevention Sets contain the same dominant CCF events, so variations in Prevention Set reliability will appear only in distant decimal places.

2. Considering varying levels of basic event prevention: Instead of either being not prevented at all or prevented to a single default variable, one can consider allocating other failure probabilities. It might be possible to save money on assurance of certain basic events, by settling for higher assessed failure probabilities.

3. Examining the path set content of the prevention sets: Looking back at Table 33, we see that every event in Path Set 1 also appears in Prevention Set 1. In fact, Prevention Sets of Prevention Level > 1 are typically combinations of complete path sets. There are variants of Prevention Analysis in which this may not be true, but to the extent that it is true, it offers intuition into how to think about Prevention Sets. A Path Set is a Prevention Set of Level N = 1; Prevention Sets of Level N > 1 are, in some sense, generalizations of path sets. In light of how the TEPA works, a Prevention Set containing an incomplete fragment of a path set would be a significant curiosity. And of course, it is not useful to assure performance of only part of a path set. It may be objected that other path set elements may be inherently

reliable, and do not need to be “assured,” but for purposes of the present discussion, we would deem performance of inherently reliable elements to be “assured,” albeit perhaps with minimal effort. A higher-level insight emerges from this discussion; that is, the meaningful unit of reliability performance is arguably the Success Path, not the individual components.

4. Computing the Prevention Worth of basic events other than the single-event CCF events: The Prevention Worth of an element is derived from the reliability of the complement of success paths containing that element. The Prevention Worth of elements appearing in single-event cut sets is the same as the Prevention Worth of the entire system, but events appearing in other cut sets might support an interesting discussion.

6. QUANTIFYING SOFTWARE COMMON CAUSE FAILURE USING MODEL-BASED METHOD

To explore the feasibility of applying model-based approaches in software CCF modeling and propagation, this section develops a preliminary model for quantifying software CCFs using DEPM. Section 6.1 describes the concepts used in the proposed quantification approach, including alpha factor model (AFM) and DEPM. Section 6.2 demonstrates the proposed CCF quantification approach based on a simplified case study. Section 6.3 concludes the work and outlines the future work.

6.1. Methodology

6.1.1. Common Cause Failures

CCFs are defined as component failures that meet four criteria: (1) two or more individual components fail or are degraded, including failures during demand, in-service testing, or deficiencies that would have resulted in a failure if a demand signal had been received; (2) components fail within a selected period such that success of the PRA mission would be uncertain; (3) component failures result from a single shared cause and coupling mechanism; and (4) a component failure occurs within the established component boundary.

For modeling and quantification of CCFs in PRA, various methods have been proposed and applied [4]. Among those existing CCF methods, the parametric methods are commonly applied in PRA for NPPs and are categorized into two major groups: (1) non-shock ratio models (e.g., Basic Parameter Model, BFM, AFM, and Multiple Greek Letter model) and (2) shock models (e.g., Binomial Failure Rate model and Multinomial Failure Rate model). NUREG/CR-5485 provides structured guidance on assessing the CCFs in classical PRA using parametric methods [10]. For this work, the AFM is applied in this proposed CCF quantification approach.

The AFM defines CCF probabilities from a set of failure frequency ratios and the total component frequency failure, Q_T. In terms of the basic event probabilities, the probabilities of a Common Cause Basic Event (CCBE) involving k-specific components in a CCCG of size m are given as [10],

For a staggered testing scheme,

$$Q_{m,k} = \frac{1}{\binom{m-1}{k-1}} \alpha_k Q_T \tag{22}$$

For a non-staggered testing scheme,

$$Q_{m,k} = \frac{k}{\binom{m-1}{k-1}} \frac{\alpha_k}{\alpha_T} Q_T \tag{23}$$

In the specific case of safety systems, CCF parameters estimators for standby components depend on the periodic test schemes. Classically, the testing schemes are either staggered (alternation of tests on redundant components) or non-staggered (all components are tested at the same time). In the case of staggered testing, there are two extremes: the redundant component is tested immediately upon failure of the component being tested, or it is tested on the next scheduled test. In each test episode, only one component is tested unless that component is found to have failed in which case the remaining components are tested. With a non-staggered testing regime, components are usually tested sequentially but within a short time. If the first component works, there may be no CCF. However, if the first fails, the subsequent test performed on the second will reveal if there is a CCF.

The estimator for the AFM involving k specific components in a CCCG of size m are given as,

$$\alpha_k = \frac{n_k}{\sum_{k=1}^{m} n_k} \tag{24}$$

6.1.2. Dual Error Propagation Method

The purpose of DEPM is to provide the possibility of a simultaneous probabilistic analysis of both control and data flow in the system under consideration [41]. In DEPM, two directed graph models are defined using the set of elements of a system: a data flow graph (DFG) and a control flow graph (CFG). Nodes of both graphs represent the elements of the system. Arcs of the DFG define paths of data transfers between the elements, which are also seen as paths of error propagation as shown in Figure 11.

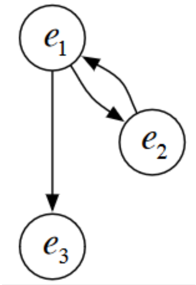


Figure 11: Example of a Data Flow Graph.

Arcs of the CFG represent control flow transitions between the elements, determining the order of their execution, as shown in Figure 12. The arcs of the CFG are weighted and show the probabilities of control transitions, like in a state graph of a discrete time Markov chain.

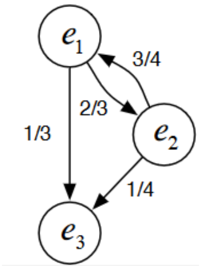


Figure 12: Example of a Control Flow Graph.

Faults can be activated in the elements during their execution and result in the occurrence of errors. The occurred errors propagate to other elements through the data transfer and control flow paths. The error propagation between the elements is determined by the DFG and the CFG structure. For example, the system shown in Figure 13 has a fault activated at element e_1, with a probability of $EP_{e1} = 0.1$. Then the failure probability of element e_7 is given by,

$$P_f(e_7) = P_{e1\rightarrow e2} \cdot P_{e2\rightarrow e5} \cdot P_{e5\rightarrow e7} \cdot EP_{e1} = \frac{1}{2} \cdot 1 \cdot \frac{1}{2} \cdot 0.1 = 0.025 \tag{25}$$

Similarly, for state-based quantification, all the possible states of the system are enumerated and the failure probability is calculated by multiplying the probabilities of the arcs and, in case of multiple failure states, adding the products of the arcs. This will be demonstrated in next section.

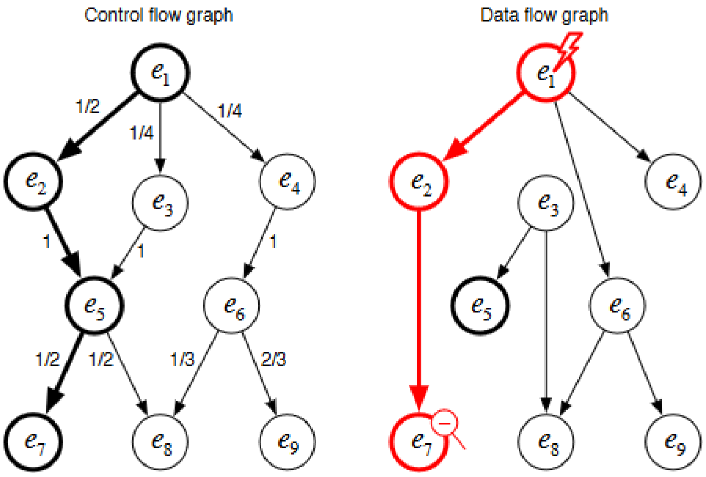


Figure 13: Quantifying Failure Probability.

6.2. A Quantification Approach for Software Common Cause Failures

In this section, the proposed quantification approach is demonstrated using the example of two BPs. For every division of the reactor trip system, there are two redundant BPs. The function of a BP is to compare an incoming process variable from a sensor, to a predefined setpoint, and to send a trip signal to the local coincidence logic processor if the process variable does not fit what the setpoint specifies. Figure 14 shows the internal functions of the BP.

The process variable is first converted from analog signal to a digital signal. The bistable comparator algorithm (BCA) then receives the digital process variable. The setpoint algorithm is where the logic for each process variable setpoint is present. For some process variables, the incoming signal should be less than the setpoint. For other process variables, the incoming variables should be greater than the setpoint.

Depending on the setpoint logic, the BCA performs a comparison of the process variables with their respective setpoints and outputs a binary digital output, termed as low and high, which respectively represents conditions are satisfied or not. The trip algorithm's function is to send the trip signal to the local coincidence logic processors if the binary digital output from the BCA is high.

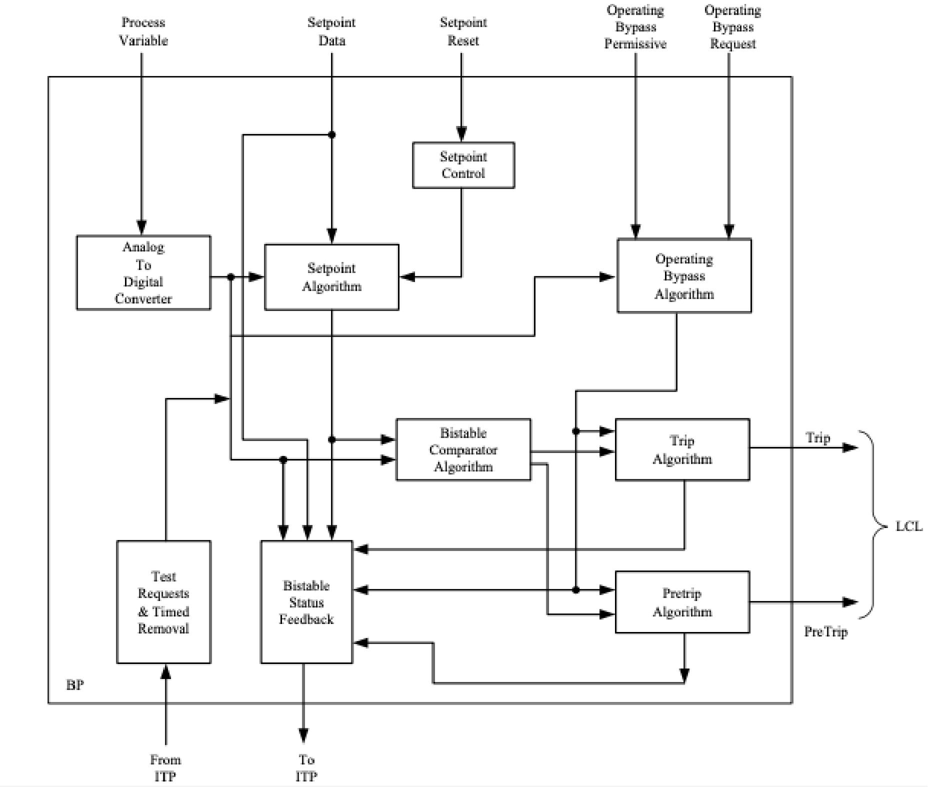


Figure 14: Bistable trip logic functional block diagram.

Figure 15 shows the DEPM model for two BPs in one division of a representative reactor trip system. To reduce the complexity in modeling, we have not considered the analog to digital converter; instead, we assume that the process variable is in digital format when it comes to the BCA. Table 36 gives the representations for the symbols used in Figure 15. As shown in Figure 15, the purple blocks represent elements, and the black arrows represent control flow. The gray blocks represent data, and the blue arrows represent data flow. The orange block represents a failure state. The control flow and data flow information for Figure 15 is given in Table 37.

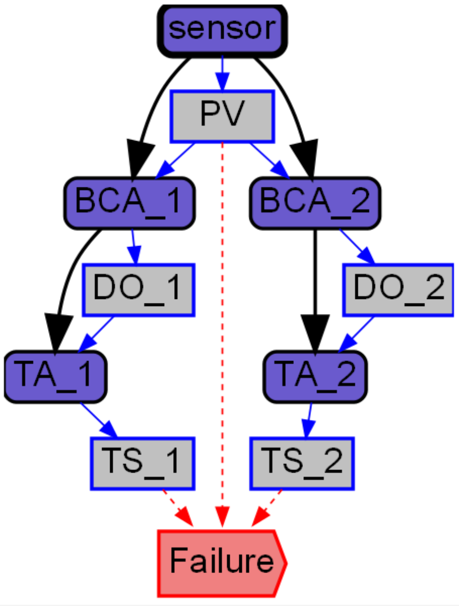


Figure 15: Dual Error Propagation Model for Two Bistable Processors

Table 36: Dual Error Propagation Model Information.

Symbol	Representation
PV	Process Variable
BCA_1	Bistable Comparator Algorithm of Bistable Processor 1
DO_1	Digital Output from BCA_1
TA_1	Trip Algorithm of Bistable Processor 1
TS_1	Trip Signal from TA_1
BCA_2	Bistable Comparator Algorithm of Bistable Processor 2
DO_2	Digital Output from BCA_2
TA_2	Trip Algorithm of Bistable Processor 2
TS_2	Trip Signal from TA_2

Table 37: CFG and DFG Information for DEPM Model.

Element/Data/Failure	Probabilities and Conditions.
Sensor	Control flow is initiated at the sensor and goes to elements BCA_1 and BCA_2.
PV	In this example we consider the range of 10,11 and 12 units to be the process variables with probabilities of 0.333 each.
BCA_1	The setpoint is set at 12 units. If the PV is 10 or 11units, then DO_1 is low, if it is 12 units, then the PV is high.
DO_1	DO_1 has two states, high and low.
TA_1	If DO_1 is low, then TS_1 will be off. If DO_1 is high, then TS_1 is on.
TS_1	TS_1 has two states, on and off.
BCA_2	The setpoint is set at 11 units. If the PV is 10units, then DO_1 is low, if it is 11 or 12 units, then the PV is high. It has an error probability of 0.1, which means that there is a 0.1 chance that trip signal may be a false off or a false on.
DO_2	DO_2 has two states, high and low.
TA_2	If DO_2 is low, then TS_2 will be off. If DO_2 is high, then TS_2 is on.

TS_2	TS_2 has two states, on and off.
Failure	In this example, we set up the failure for the conditions of an incorrect setpoint being given to BCA_1. The correct setpoint is supposed to be 11 but is set to 12 in BCA_1. Hence, we can define failure as the condition that process variable is 11 units, TS_1 is off and TS_2 is off. That is, we need one out of the two BCAs to function properly.

* Full names of the acronyms can be found in the Acronyms list on Page ix.

The state space for the DEPM model with two independent Bistable comparator algorithms is shown in Figure 16. The failure probability can be calculated as,

$$P_f(11units, TS_1 = off, TS_2 + 2) = P_{11\,units} \cdot EP_{BCA_2} = 0.333 \cdot 0.1 = 0.0333 \tag{26}$$

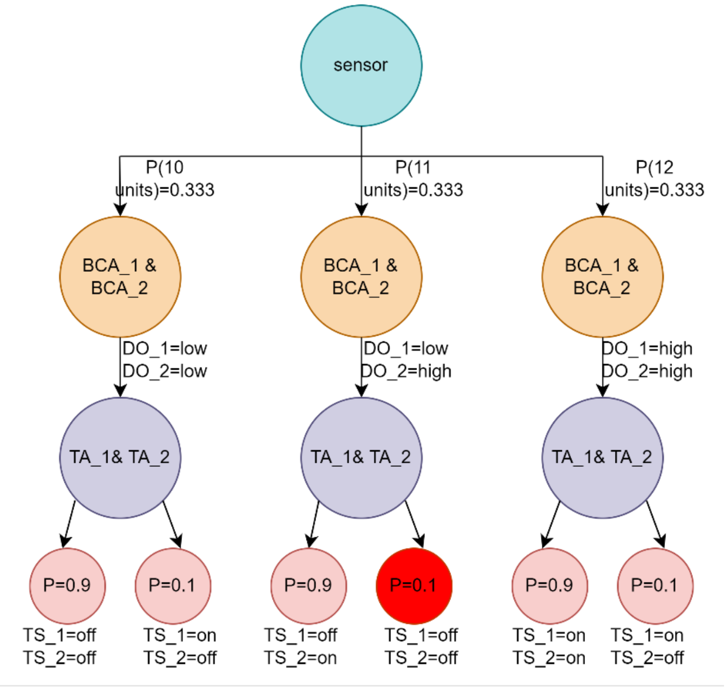


Figure 16. State Space for DEPM model with two BCAs.

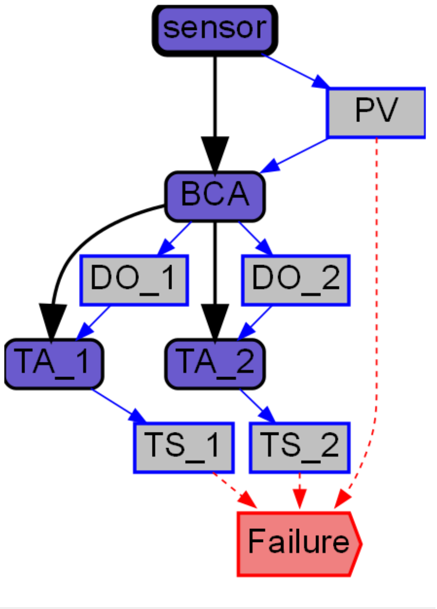


Figure 17: DEPM Model for common BCA.

Table 38. CFG and DFG for DEPM model with common code for BCAs.

Element/Data/Failure	Probabilities and Conditions.
Sensor	Control flow is initiated at the sensor and goes to elements BCA_1 and BCA_2.
PV	In this example we consider the range of 10,11 and 12 units to be the process variables with probabilities of 0.333 each.
BCA	The setpoint is set at 12 units. If the PV is 10 or 11units, then DO_1 is low, if it is 12 units, then the PV is high.
DO_1	DO_1 has two states, high and low.
TA_1	If DO_1 is low, then TS_1 will be off. If DO_1 is high, then TS_1 is on.
TS_1	TS_1 has two states, on and off.
DO_2	DO_2 has two states, high and low.
TA_2	If DO_2 is low, then TS_2 will be off. If DO_2 is high, then TS_2 is on.
TS_2	TS_2 has two states, on and off.
Failure	In this example, we set up the failure for the conditions of an incorrect setpoint being given to the BCA. The correct setpoint is supposed to be 11 but is set to 12. Hence, we can define failure as the condition that process variable is 11 units, TS_1 is off and TS_2 is off.

* Full names of the acronyms can be found in the Acronyms list on Page ix.

We can see from Figure 18—having a common software component between the BPs results in one branch of the state space satisfying the conditions for failure, with a probability of 0.333.

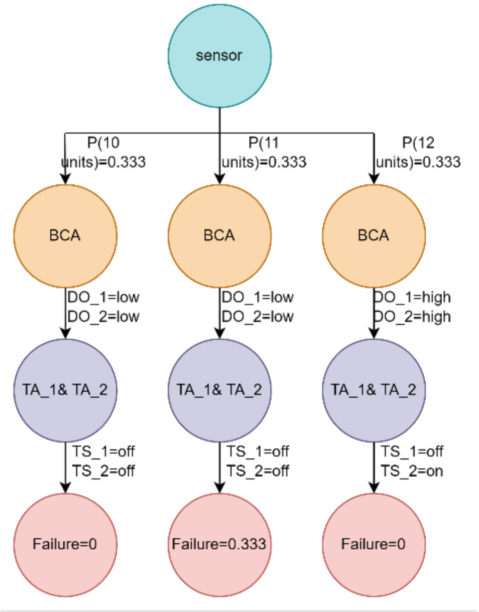


Figure 18. State space for DEPM Model with common code for BCAs.

Thus, for the coupling mechanism of an incorrect setpoint, we can use the alpha factor estimator to get its specific alpha factors. We can denote them as $\alpha_{1,1}$ and $\alpha_{1,2}$, which respectively refer to the independent failure probability of the bistable processor due to the incorrect setpoint and the specific alpha factor for the software CCF of the two bistable processors due to the incorrect setpoint. Then we get,

$$\alpha_{1,1} = \frac{n_1}{\sum_{k=1}^{2} n_k} = \frac{0.1 * 0.0333}{0.333 + 0.1 \cdot 0.33} = 0.0909;\ \alpha_{1,2} = \frac{n_2}{\sum_{k=1}^{2} n_k} = \frac{0.0333}{0.333 + 0.1 \cdot 0.33} = 0.909 \qquad (27)$$

Similarly, we exhaustively analyze the probability of failures due to software coupling mechanisms in a system and get the specific alpha factors. Then for N coupling mechanisms, we can obtain the alpha factors from,

$$\begin{matrix} \alpha_1 \\ \alpha_2 \end{matrix} = \left(\begin{matrix} \alpha_{1,1} \\ \alpha_{2,1} \end{matrix} + \begin{matrix} \alpha_{1,2} \\ \alpha_{2,2} \end{matrix} + \cdots + \begin{matrix} \alpha_{1,N} \\ \alpha_{2,N} \end{matrix} \right) / N \qquad (28)$$

For N coupling mechanisms and a CCBE of size n,

$$\begin{matrix} \alpha_1 \\ \vdots \\ \alpha_N \end{matrix} = \begin{pmatrix} \alpha_{1,1} & \cdots & \alpha_{1,n} \\ \vdots & \ddots & \vdots \\ \alpha_{N,1} & \cdots & \alpha_{N,n} \end{pmatrix} / N \qquad (29)$$

In our case study, we considered one of the coupling mechanisms (incorrect setpoint set in the Bistable Comparator Algorithm) that can cause a common cause failure of the bistable processors. However, there can be other mechanisms that can cause a CCF of the bistable processer. For example, software error in the trip algorithm, sensor malfunction, etc. When we calculate the alpha factors for software CCFs of the bistable processors, we need to consider all the 'N' number of mechanisms in which it will fail, which will each have its own specific alpha factors.

6.3. Discussion

This section presented an approach to obtaining the parameters of the AFM from system design, input parameter, and error propagation information. A preliminary model for quantifying software CCFs is developed using a model-based approach called DEPM, which quantitatively provides software CCF modeling and error propagation between digital components of a simplified RTS model. This preliminary study builds up a technical basis for the application of model-based system engineering approaches on the risk assessment and design optimization of DI&C systems. The model-based simulation methods ure review focus either on representing the software as elements of a non-probabilistic model, or they use functions to quantify physical failure. In contrast, DEPM offers simultaneous probabilistic modeling of software and hardware components of the system. As compared to classical PRA methods that model only the failures of the I&C elements, DEPM can model the execution of the element's functions, where we can introduce and propagate errors, to quantify the number of times that we get a failure state.

Relevant future work will be focused on the model uncertainty quantification and validation, and, further demonstration on the digital RTS and ESFAS. This approach needs to be validated on the systems where CCF parameters already exist. Further, the availability of operational and error propagation data required for DEPM modeling of software CCFs needs to be investigated and consolidated.

7. CONCLUSIONS AND FUTURE WORKS

7.1. Conclusions

This report outlines R&D focused on methodology improvements of software CCF modeling and estimation and introduces additional innovative approaches to risk assessment of DI&C systems such as prevention analysis and importance analysis to enable a comparative assessment of various DI&C design architectures. This research is intended to enhance the robustness of the methodology for the risk assessment and design optimization of safety-critical DI&C systems.

An approach for modeling software-based CCFs is proposed to support the LWRS-developed framework for risk assessment of both redundant and diverse configurations of safety-critical DI&C designs. A set of tables for assessing defenses against CCFs was provided, and relevant concepts were supported by the proposed BAHAMAS approach in the case study for diverse software CCF analysis. The results of the case study show that, as anticipated for the CCCGs that have diversity, the CCF probability is reduced when compared to the CCCGs without diversity. By employing FT and ET analyses, the effect of diversity on the reliability of a digital RTS, the influence of software CCFs on the reliability of the RTS and the impact that the RTS has on the NPP have been analyzed.

Sensitivity and importance analyses have been performed for different designs of RTS and ESFAS. In addition to tractional importance measures of risk significance, a new method called TEPA is also applied to evaluate the importance of various system components based on the measure of safety significance. Safety significance refers to the significance of a contribution to system success probability, while risk significance refers to the significance of a contribution to system failure probability. TEPA helps identify a minimum collection of components, rather than a single component, modeled in the PRA that are effective in managing safety. The investigation conducted in this work suggests an approach that integrates the importance measures for both risk and safety significance and can provide more informative and insightful recommendations on DI&C system design optimization.

A preliminary model for quantifying software CCFs is developed using a model-based approach called DEPM, which quantitatively provides software CCF modeling and error propagation between digital components of a simplified RTS model. This preliminary study builds up a technical basis on the application of model-based system engineering approaches for the risk assessment and design optimization of DI&C systems of NPPs.

7.2. Future Works

The project develops methods for quantitative assessment of risks associated with DI&C systems to fill the technical gaps that exist in the very urgent industry need of the digital modernization of existing NPPs. The work scope of this project in FY-23 is to demonstrate the framework using industry data and make improvements based on the findings from the demonstration as well as based on the identified industry needs. Key activities in FY-23 include:

- Improve current framework and complete a demonstration case of evaluation of various safety-critical DI&C design architectures in terms of safety assurance and economic efficiencies.
- Develop capability to perform risk informed and performance-based analysis of various DI&C design architectures. Explore the applicability of performance-based approaches for the safety, reliability, and resiliency assurance of safety-critical DI&C systems.
- Collaborate with university partners to evaluate, adjust, and integrate state-of-the-art methods in the current framework for estimating probabilities of potential software CCFs in highly redundant and diverse safety-critical DI&C systems.
- Collaborate with the industry to couple the LWRS-developed Framework with existing risk informed approaches to better support the optimization of safety-critical DI&C designs and upgrades.

8. REFERENCES

[1] H. Bao, H. Zhang and K. Thomas, "An Integrated Risk Assessment Process for Digital Instrumentation and Control Upgrades of Nuclear Power Plants," Idaho National Laboratory, Idaho Falls, ID, August 2019.

[2] H. Bao, H. Zhang and T. Shorthhill, "Redundancy-guided System-theoretic Hazard and Reliability Analysis of Safety-related Digital Instrumentation and Control Systems in Nuclear Power Plants," Idaho National Laboratory, Idaho Falls, ID, August 2020.

[3] H. Bao, T. Shorthill, E. Chen and H. Zhang, "Quantitative Risk Analysis of High Safety-significant Safety-related Digital Instrumentation and Control Systems in Nuclear Power Plants using IRADIC Technology," Idaho National Laboratory, Idaho Falls, ID, August 2021.

[4] T. Shorthill, H. Bao, H. Zhang and H. Ban, "A Redundancy-Guided Approach for the Hazard Analysis of Digital Instrumentation and Control Systems in Advanced Nuclear Power Plants," *Nuclear Technology,* vol. 208, no. 5, pp. 892-911, 2022.

[5] H. Bao, T. Shorthill and H. Zhang, "Hazard Analysis for Identifying Common Cause Failures of Digital Safety Systems using a Redundancy-Guided Systems-Theoretic Approach," *Annals of Nuclear Energy,* vol. 148, p. 107686, 2020.

[6] N. G. Leveson and J. P. Thomas, STPA Handbook, March 2018.

[7] A. J. Clark, A. D. Williams, A. Muna and M. Gibson, "Hazard and Consequence Analysis for Digital Systems – A New Approach to Risk Analysis in the Digital Era for Nuclear Power Plants," in *Transactions of the American Nuclear Society*, Orlando, Florida, USA, 2018.

[8] T. Shorthill, H. Bao, Z. Hongbin and H. Ban, "A novel approach for software reliability analysis of digital instrumentation and control systems in nuclear power plants," *Annals of Nuclear Energy,* vol. 158, 2021.

[9] H. Bao, T. Shorthill, E. Chen, J. Park, S. Zhang, A. V. Jayakumar, C. Elks, N. Dinh, H. Ban, H. Zhang, E. Quinn and S. Lawrence, "An Integrated Framework for Risk Assessment of High Safety-significant Safety-related Digital Instrumentation and Control Systems in Nuclear Power Plants: Methodology and Demonstration," Idaho National Laboratory, Idaho Falls, ID, August 2022.

[10] A. Mosleh, D. Rasmuson and F. Marshall, "Guidelines on Modeling Common-Cause Failures in Probabilistic Risk Assessment," NUREG/CR-5485, U.S. Nuclear Regulatory Commission, Washington, D.C., USA, 1998.

[11] M. Muhlheim and R. Wood, "Technical Basis for Evaluating Software-Related Common-Cause Failures," Oak Ridge National Laboratory, Oak Ridge, 2016.

[12] J. C. Knight and N. G. Leveson, "An Experimental Evaluation of the Assumption of Independence in Multiversion Programming," *IEEE Transactions on Software Engineering,* vol. 12, no. 1, pp. 96-109, 1986.

[13] National Research Council, "Digital Instrumentation and Controls Systems in Nuclear Power Plants: Safety and Reliability Issues," The National Academies Press, Washington, 1997.

[14] K. Salako and L. Strigini, "When Does "Diversity" in Development Reduce Common Failures? Insights from Probabilistic Modeling," *IEEE Transactions on Dependable and Secure Computing,* vol. 11 No. 2, pp. 193-206, 2014.

[15] D. Kančev and M. Čepin, "A new method for explicity modelling of single failure evetn within different common cause failure groups," *Reliability Engineering and System Safety,* vol. 103, pp. 84-93, 2012.

[16] R. A. Humphreys, "Assigning a Numerical Value to the Beta Factor Common Cause Evaluation," in *Reliability '87*, 1987.

[17] V. P. Brand, Ed., UPM 3.1: A pragmatic approach to dependent failures assessment for standard systems, SRDA-R13, Warrington, UK: AEA Technology, Safety and Reliability Directorate, 1996.

[18] International Electrotechnical Commission, "Part 6: Guidelines on the application of parts 2 and 3," in *IEC 61508 Functional safety of electrical/electronic/programmable electronic safety-related systems*, 2 ed., International Electrotechnical Commission, 2010.

[19] M. Muhlheim and R. Wood, "Technical Basis for Evaluating Software-Related Common-Cause Failures," Oak Ridge National Laboratory, Oak Ridge, 2016.

[20] D. M. Rasmuson and D. L. Kelly, "Common-cause failure analysis in event assessment," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 222, pp. 521-532, 2008.

[21] S. Hauge, S. Håbrekke and M. A. Lundteigen, "Reliability Prediction Method for Safety Instrumented Systems – PDS Example collection, 2010 Edition," SINTEF Technology and Society , Trondheim, Norway, 2010.

[22] A. O'Connor and A. Mosleh, "Extending the Alpha Factor Model for Cause Based Treatment of Common Cause Failure Events in PRA and Event Assessment," in *PSAM 2014 - Probabilistic Safety Assessment and Management*, Honolulu, Hawaii, 2014.

[23] A. O'Connor and A. Mosleh, "A general cause based methodology for analysis of common cause and dependent failures in system risk and reliability assessments," *Reliability Engineering and System Safety*, vol. 145, pp. 341-350, 2016.

[24] Z. Ma, J. K. Knudsen, J. A. Schroeder and C. L. Smith, "Feasibility Study of Developing Alternative Common-Cause Failure Model for Event Assessment," INL/EXT-21-33376, Idaho National Laboratory, August 2021.

[25] E. Higo, S. Soga and H. Miura, "Inter-unit common cause failure analysis based on data from intra-unit cases," in *ICONE2020 - Proceedings of the 2020 Conference on Nuclear Engineering*, Virtual, 2020.

[26] S. Soga, E. Higo and H. Miura, "A systematic approach to estimate an inter-unit common-cause failure probability," *Reliability Engineering and System Safety,* vol. 2015, 2021.

[27] A. O'Connor and A. Mosleh, "A General Cause Based Methodology for Analysis of Dependent Failures in System Risk and Reliability Assessments," University of Maryland, 2013.

[28] R. Chillarege, I. S. Bhandari, J. K. Chaar, M. J. Halliday, D. S. Moebus, B. K. Ray and M.-Y. Wong, "Orthogonal Defect Classification - A Concept for In-Process Measurements," *International Institute of Electrical Engineers Transactions on Software Engineering*, vol. 18, no. 11, pp. 943-956, 1992.

[29] C. Smith, J. Knudsen, K. Vedros, M. Calley, K. Kvarfordt and T. Wood, "SAPHIRE 8 Basics An Introduction to Probabilistic Risk Assessment via the Systems Analysis Program for Hands-On Integrated Reliability Evaluations (SAPHIRE) Software," Idaho National Laboratory, Idaho Falls, ID, 2016.

[30] R. Youngblood and L. Oliveira, "Application of An Allocation Methodology," in *International Topical Meeting on Probability, Reliability and Safety Assessment*, Pittsburgh, PA, 1989.

[31] R. Youngblood and L. Oliveira, "A Boolean Approach to Redundancy Optimization," in *Tranctions of the American Nuclear Society*, Nashville, TN, 1990.

[32] R. Worrell and D. Blanchard, "Top Event Prevention Analysis to Estimate Requirements Marginal to Safety," in *Proceedings of the 1995 Joint ASME/JSME Pressure Vessels and Piping Conference*, Honolulu, HI, 1995.

[33] R. Worrell and D. Blanchard, "Top event prevention analysis: A deterministic use of PRA," in *ICONE 4: ASME/JSME international conference on nuclear engineering,*, New Orleans, LA, 1996.

[34] R. White and D. Blanchard, "Development of a Risk-Informed IST Program at Palisades Using Top Event Prevention," in *ICONE 10: 10. international conference on nuclear engineering,* Arlington, VA, 2002.

[35] D. Blanchard, B. Varnado and R. Worrell, "Risk-Informed Physical Security; Dynamic Allocation of Resources," in *ANS International Topical Meeting on Probabilistic Safety Analysis,* San Francisco, CA, 2005.

[36] D. Blanchard and R. Youngblood, "Risk Informed Safety Margin Characterization Case Study: Selection of Electrical Equipment To Be Subjected to Environmental Qualification," Idaho National Laboratory, Idaho Falls, ID, 2012.

[37] R. Youngblood, "Applying Risk Models To Formulation Of Safety Cases," *RIsk Analysis,* vol. 18, no. 4, p. 433, 1998.

[38] R. Youngblood, "Risk Significance and Safety Significance," *Reliability Engineering and System Safety,* vol. 73, pp. 121-136, 2001.

[39] R. Youngblood, "Multi-State Top Event Prevention Analysis," in *The 30th Europeam Safety and Reliability Conference, the 15th Probabilistic Safety Assessment and Management Conference,* Venice, Italy, 2020.

[40] M. Cheok, G. Parry and R. Sherry, "Use of Importance Measures in Risk-informed Regulatory Applications," *Relaibility Engineering and System Safety,* vol. 60, no. 3, pp. 213-226, 1998.

[41] A. Morozov, "Dual-graph model for error propagation analysis of mechatronic systems," Technische Universit¨at Dresden, Germany, 2012.

[42] U.S. NRC, "A Defense-In-Depth and Diversity Assessment of the RESAR-414 Integrated Protection System," U.S. Nuclear Regulatory Commission, Washington, DC, 1979.

APPENDIX A – EVENT TREES FOR ACCIDENT SCENARIOS

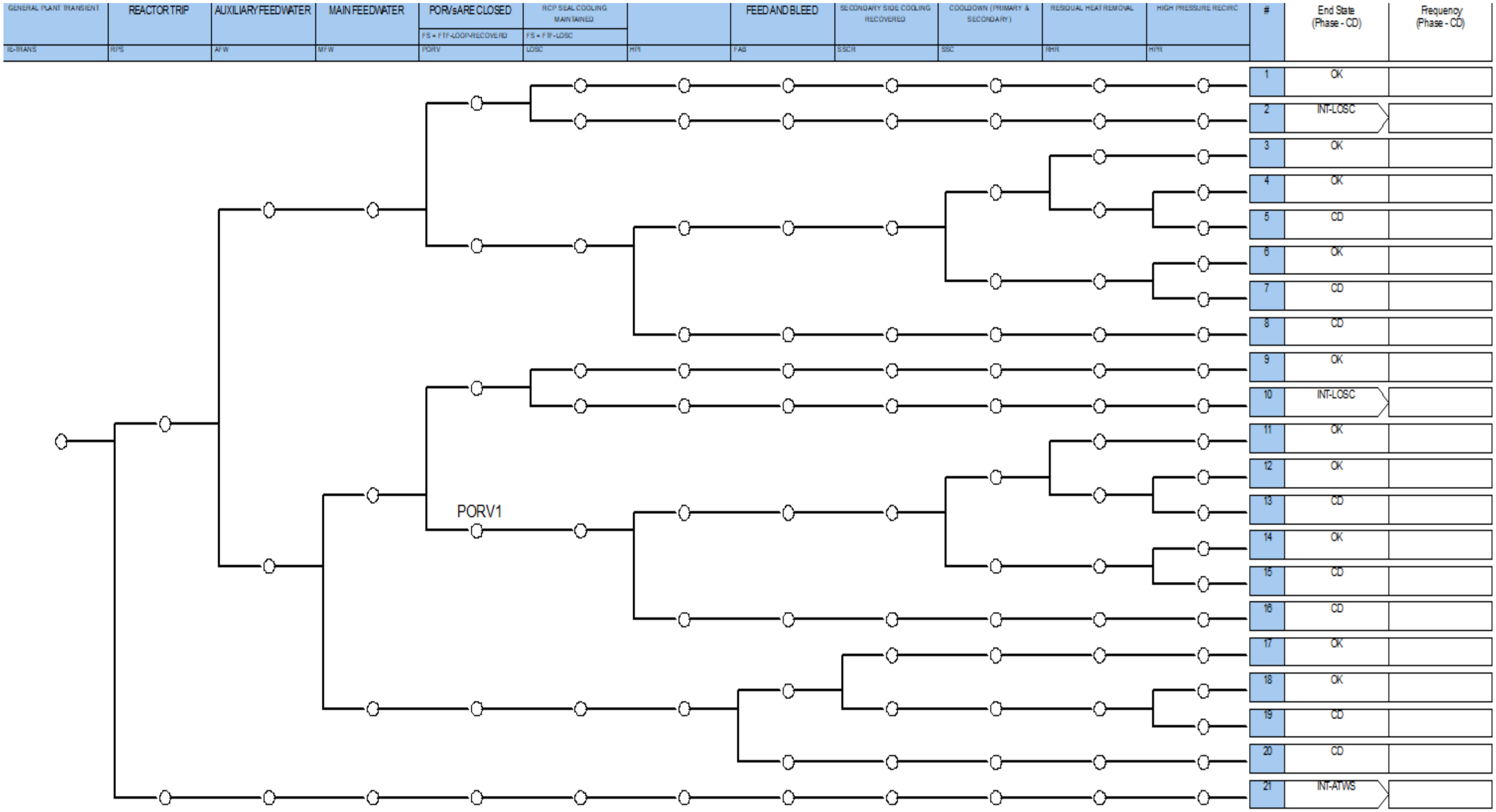


Figure 19. Generic pressurized water reactor event tree for general plant transient (TRANS).

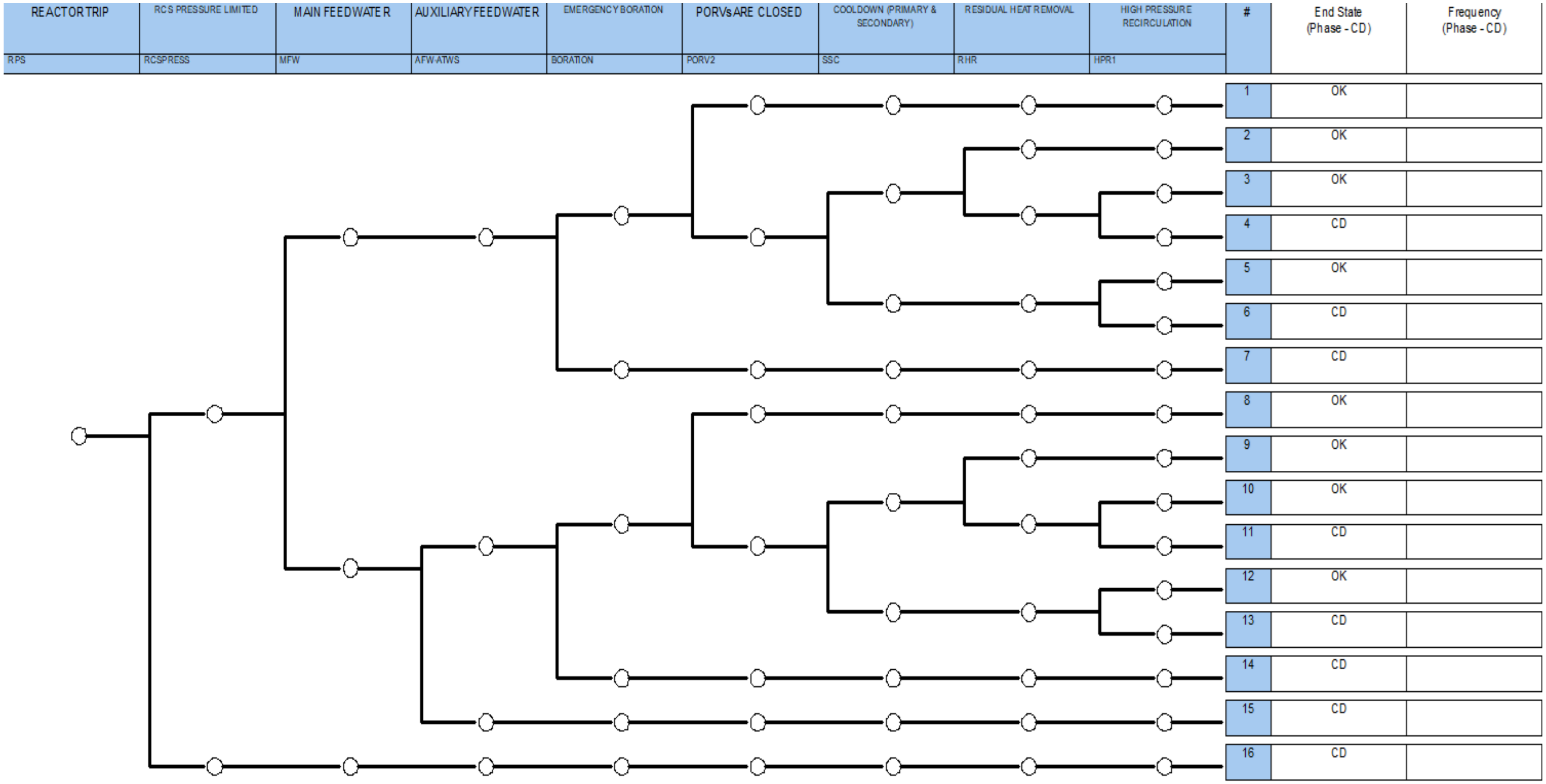
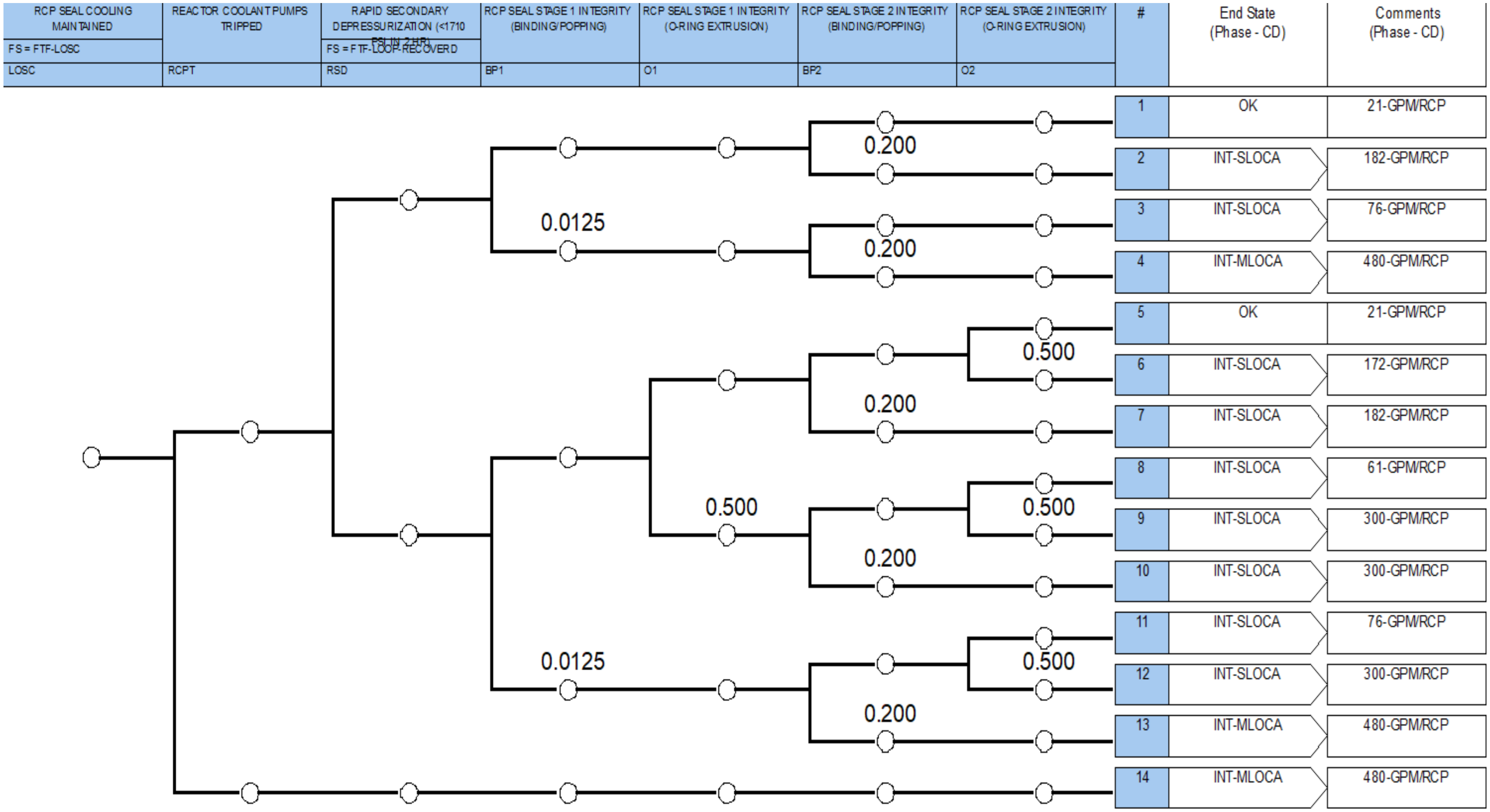


Figure 20. Generic pressurized water reactor sub event tree for anticipated transient without scram (ATWS, transferred from TRANS)


RCP SEAL COOLING MAINTAINED
FS = FTF-LOSC
LOSC
REACTOR COOLANT PUMPS TRIPPED
RCPT
RAPID SECONDARY DEPRESSURIZATION (<1710 PSI IN 2 HR)
FS = FTF-LOOP-RECOVERD
RSD
RCP SEAL STAGE 1 INTEGRITY (BINDING/POPPING)
BP1
RCP SEAL STAGE 1 INTEGRITY (O-RING EXTRUSION)
O1
RCP SEAL STAGE 2 INTEGRITY (BINDING/POPPING)
BP2
RCP SEAL STAGE 2 INTEGRITY (O-RING EXTRUSION)
O2
0.200
0.0125
0.200
0.500
0.200
0.500
0.500
0.200
0.0125
0.500
0.200

#	End State (Phase - CD)	Comments (Phase - CD)
1	OK	21-GPM/RCP
2	INT-SLOCA	182-GPM/RCP
3	INT-SLOCA	76-GPM/RCP
4	INT-MLOCA	480-GPM/RCP
5	OK	21-GPM/RCP
6	INT-SLOCA	172-GPM/RCP
7	INT-SLOCA	182-GPM/RCP
8	INT-SLOCA	61-GPM/RCP
9	INT-SLOCA	300-GPM/RCP
10	INT-SLOCA	300-GPM/RCP
11	INT-SLOCA	76-GPM/RCP
12	INT-SLOCA	300-GPM/RCP
13	INT-MLOCA	480-GPM/RCP
14	INT-MLOCA	480-GPM/RCP

Figure 21. Generic pressurized water reactor sub event tree for loss-of-seal cooling (LOSC, transferred from TRANS).

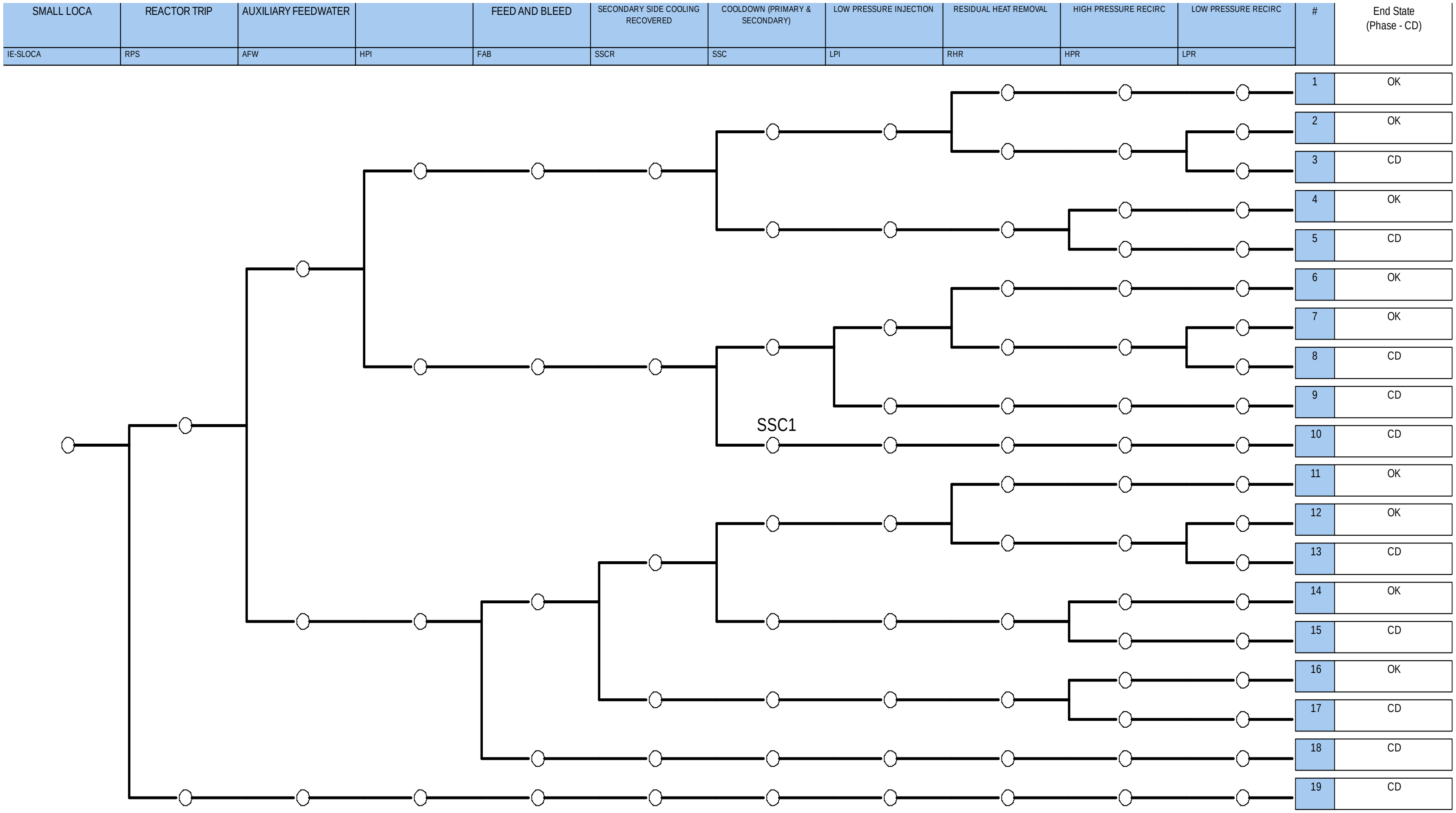
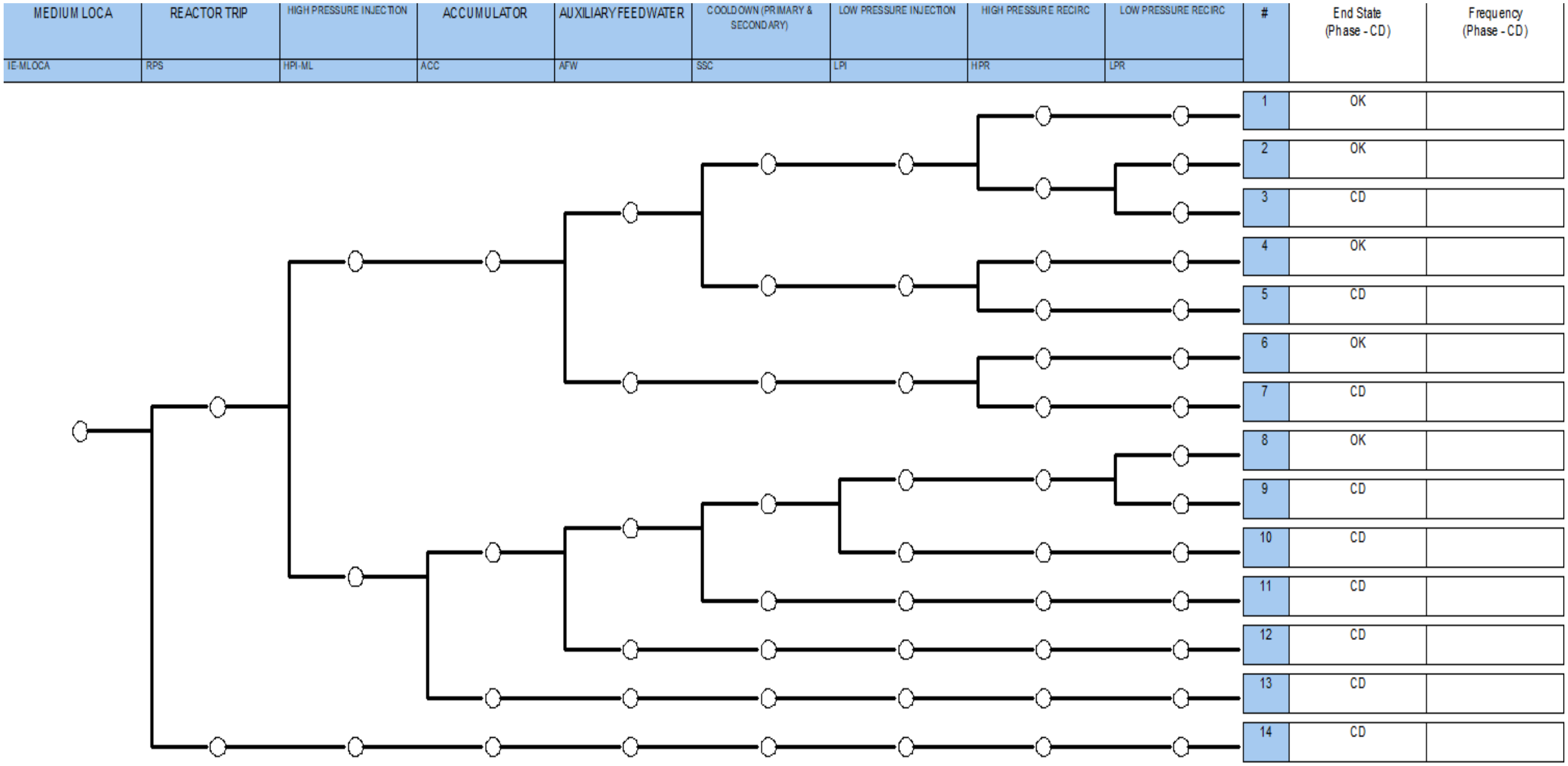


Figure 22. Generic pressurized water reactor event tree for small-break loss-of-coolant accident (SLOCA).



#	End State (Phase - CD)	Frequency (Phase - CD)
1	OK	
2	OK	
3	CD	
4	OK	
5	CD	
6	OK	
7	CD	
8	OK	
9	CD	
10	CD	
11	CD	
12	CD	
13	CD	
14	CD	

Figure 23. Generic pressurized water reactor event tree for medium-break loss-of-coolant accident (MLOCA).

APPENDIX B – FAULT TREES AND BASIC EVENTS FOR SYSTEM DESIGNS

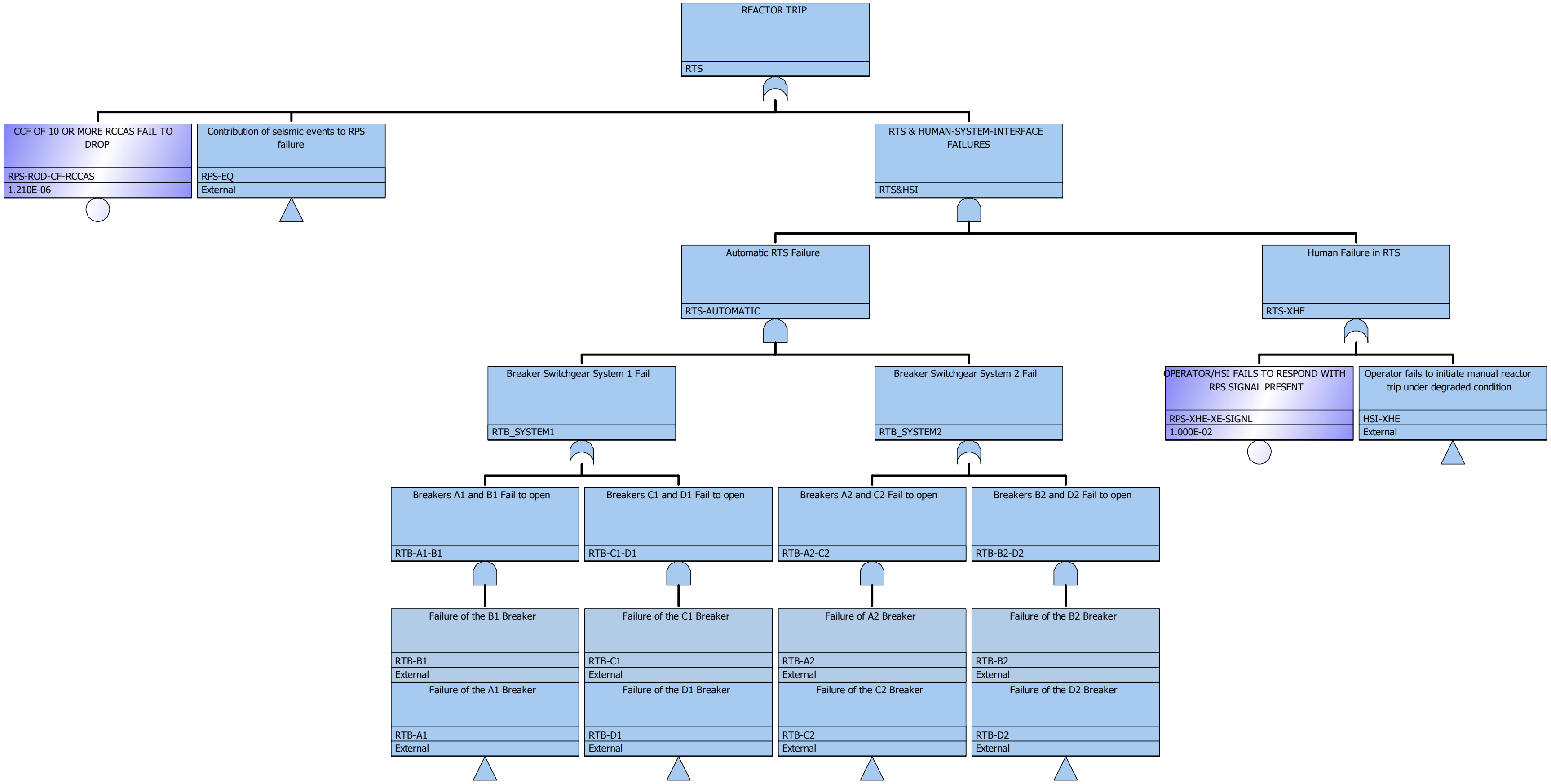

REACTOR TRIP
RTS
CCF OF 10 OR MORE RCCAS FAIL TO DROP
RPS-ROD-CF-RCCAS
1.210E-06
Contribution of seismic events to RPS failure
RPS-EQ
External
RTS & HUMAN-SYSTEM-INTERFACE FAILURES
RTS&HSI
Automatic RTS Failure
RTS-AUTOMATIC
Human Failure in RTS
RTS-XHE
Breaker Switchgear System 1 Fail
RTB_SYSTEM1
Breaker Switchgear System 2 Fail
RTB_SYSTEM2
OPERATOR/HSI FAILS TO RESPOND WITH RPS SIGNAL PRESENT
RPS-XHE-XE-SIGNL
1.000E-02
Operator fails to initiate manual reactor trip under degraded condition
HSI-XHE
External
Breakers A1 and B1 Fail to open
RTB-A1-B1
Breakers C1 and D1 Fail to open
RTB-C1-D1
Breakers A2 and C2 Fail to open
RTB-A2-C2
Breakers B2 and D2 Fail to open
RTB-B2-D2
Failure of the B1 Breaker
RTB-B1
External
Failure of the A1 Breaker
RTB-A1
External
Failure of the C1 Breaker
RTB-C1
External
Failure of the D1 Breaker
RTB-D1
External
Failure of A2 Breaker
RTB-A2
External
Failure of the C2 Breaker
RTB-C2
External
Failure of the B2 Breaker
RTB-B2
External
Failure of the D2 Breaker
RTB-D2
External

Figure 24. Fault tree showing the RTS failure. The RTS failure can be a result of hardware failures of rod cluster control assemblies, or automatic actuation failures, or manual actuation failures. The automatic actuation failure can be traced down to selective combinations of eight breaker failures, which are modeled in sub fault trees.

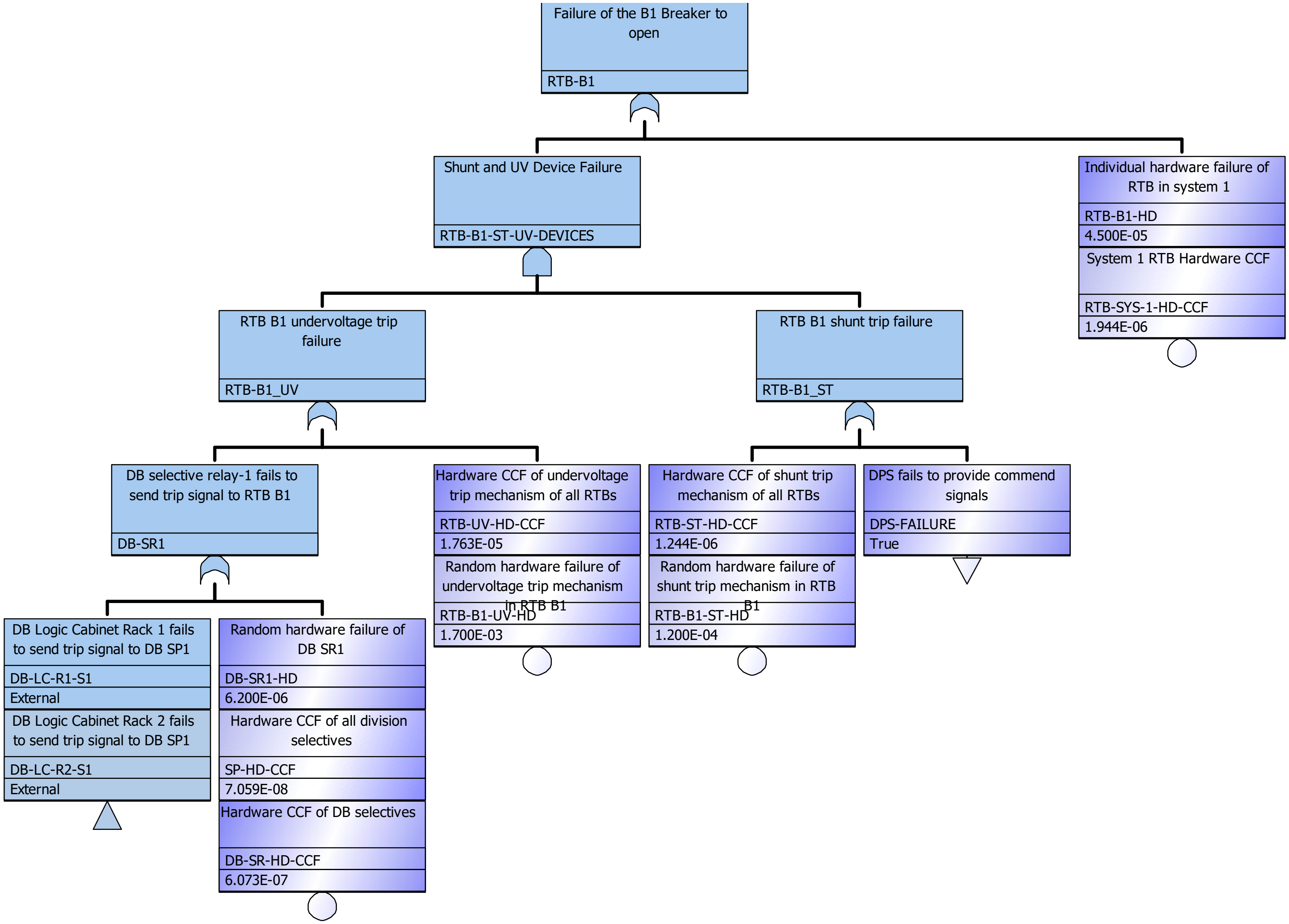

Failure of the B1 Breaker to open
RTB-B1
Shunt and UV Device Failure
RTB-B1-ST-UV-DEVICES
Individual hardware failure of RTB in system 1
RTB-B1-HD
4.500E-05
System 1 RTB Hardware CCF
RTB-SYS-1-HD-CCF
1.944E-06
RTB B1 undervoltage trip failure
RTB-B1_UV
RTB B1 shunt trip failure
RTB-B1_ST
DB selective relay-1 fails to send trip signal to RTB B1
DB-SR1
Hardware CCF of undervoltage trip mechanism of all RTBs
RTB-UV-HD-CCF
1.763E-05
Random hardware failure of undervoltage trip mechanism in RTB B1
RTB-B1-UV-HD
1.700E-03
Hardware CCF of shunt trip mechanism of all RTBs
RTB-ST-HD-CCF
1.244E-06
Random hardware failure of shunt trip mechanism in RTB B1
RTB-B1-ST-HD
1.200E-04
DPS fails to provide commend signals
DPS-FAILURE
True
DB Logic Cabinet Rack 1 fails to send trip signal to DB SP1
DB-LC-R1-S1
External
DB Logic Cabinet Rack 2 fails to send trip signal to DB SP1
DB-LC-R2-S1
External
Random hardware failure of DB SR1
DB-SR1-HD
6.200E-06
Hardware CCF of all division selectives
SP-HD-CCF
7.059E-08
Hardware CCF of DB selectives
DB-SR-HD-CCF
6.073E-07

Figure 25. Sub fault tree showing a breaker failure. A breaker failure can be a result of hardware failure or control failure. The control function can be achieved by either undervoltage trip or shunt trip. Taking undervoltage trip as an example, its failure can be traced down to either hardware failure or logic cabinet rack failures to send out trip signals.

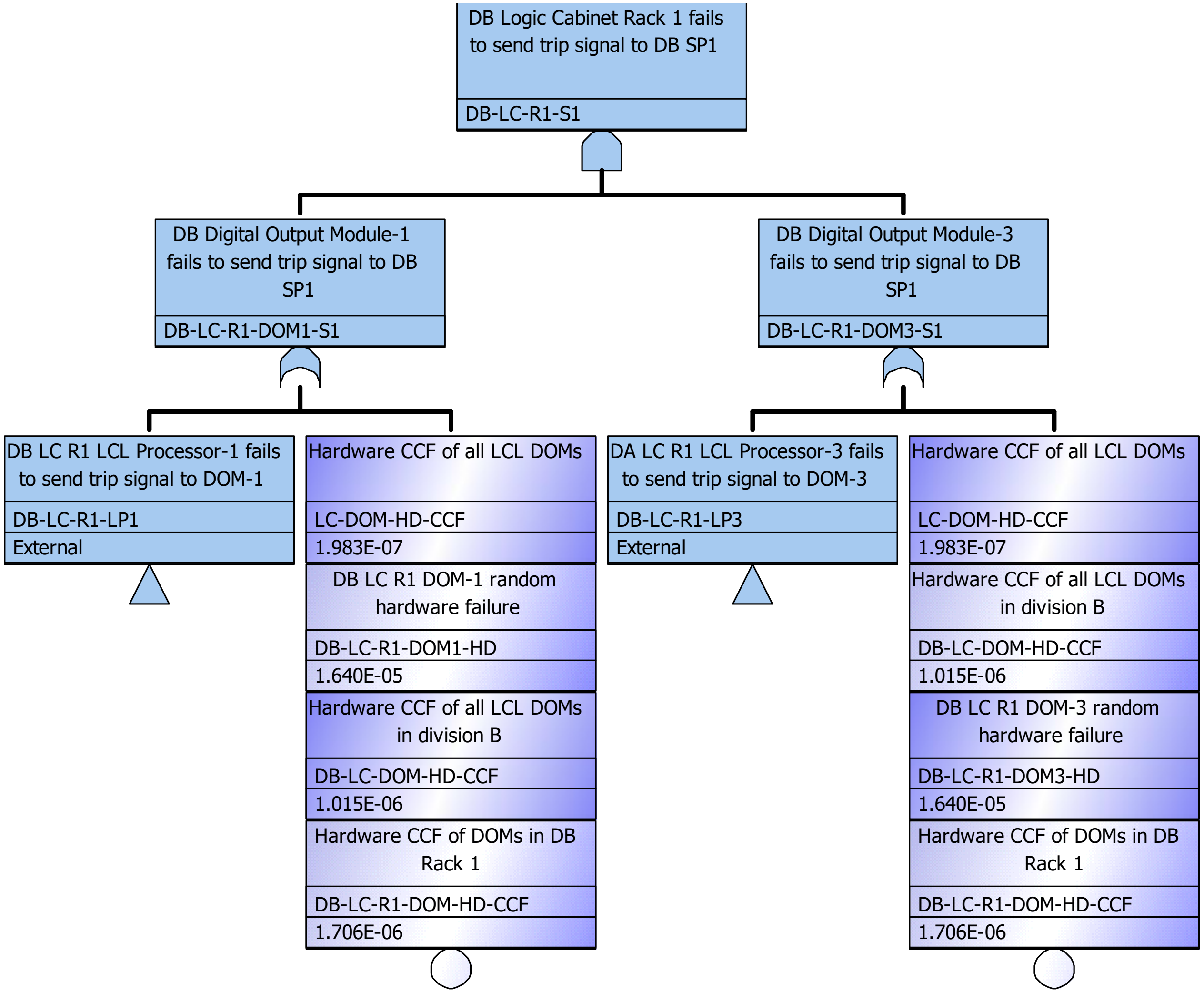

DB Logic Cabinet Rack 1 fails to send trip signal to DB SP1
DB-LC-R1-S1
DB Digital Output Module-1 fails to send trip signal to DB SP1
DB-LC-R1-DOM1-S1
DB Digital Output Module-3 fails to send trip signal to DB SP1
DB-LC-R1-DOM3-S1
DB LC R1 LCL Processor-1 fails to send trip signal to DOM-1
DB-LC-R1-LP1
External
Hardware CCF of all LCL DOMs
LC-DOM-HD-CCF
1.983E-07
DB LC R1 DOM-1 random hardware failure
DB-LC-R1-DOM1-HD
1.640E-05
Hardware CCF of all LCL DOMs in division B
DB-LC-DOM-HD-CCF
1.015E-06
Hardware CCF of DOMs in DB Rack 1
DB-LC-R1-DOM-HD-CCF
1.706E-06
DA LC R1 LCL Processor-3 fails to send trip signal to DOM-3
DB-LC-R1-LP3
External
Hardware CCF of all LCL DOMs
LC-DOM-HD-CCF
1.983E-07
Hardware CCF of all LCL DOMs in division B
DB-LC-DOM-HD-CCF
1.015E-06
DB LC R1 DOM-3 random hardware failure
DB-LC-R1-DOM3-HD
1.640E-05
Hardware CCF of DOMs in DB Rack 1
DB-LC-R1-DOM-HD-CCF
1.706E-06

Figure 26. Sub fault tree showing a logic cabinet rack failure, which can be caused by loss of digital output modules. A digital output module can fail either due to random hardware failure or local coincidence logic processor failure.

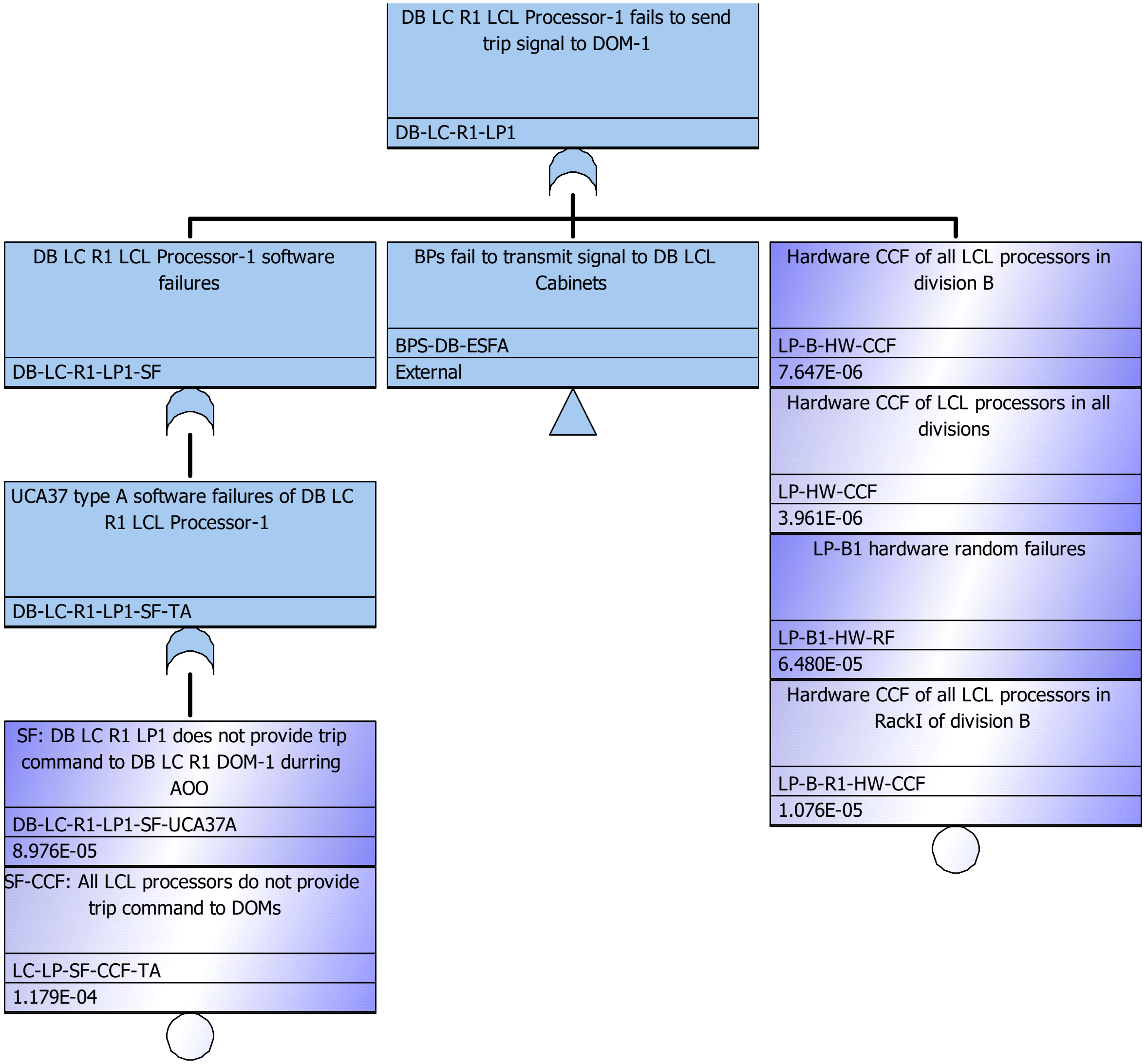

DB LC R1 LCL Processor-1 fails to send trip signal to DOM-1
DB-LC-R1-LP1
DB LC R1 LCL Processor-1 software failures
DB-LC-R1-LP1-SF
BPs fail to transmit signal to DB LCL Cabinets
BPS-DB-ESFA
External
Hardware CCF of all LCL processors in division B
LP-B-HW-CCF
7.647E-06
Hardware CCF of LCL processors in all divisions
LP-HW-CCF
3.961E-06
LP-B1 hardware random failures
LP-B1-HW-RF
6.480E-05
Hardware CCF of all LCL processors in RackI of division B
LP-B-R1-HW-CCF
1.076E-05
UCA37 type A software failures of DB LC R1 LCL Processor-1
DB-LC-R1-LP1-SF-TA
SF: DB LC R1 LP1 does not provide trip command to DB LC R1 DOM-1 durring AOO
DB-LC-R1-LP1-SF-UCA37A
8.976E-05
SF-CCF: All LCL processors do not provide trip command to DOMs
LC-LP-SF-CCF-TA
1.179E-04

Figure 27. Sub fault tree showing a local coincidence logic processor failure, which can be caused by hardware failure, software failure, or loss of inputs from bistable processors.

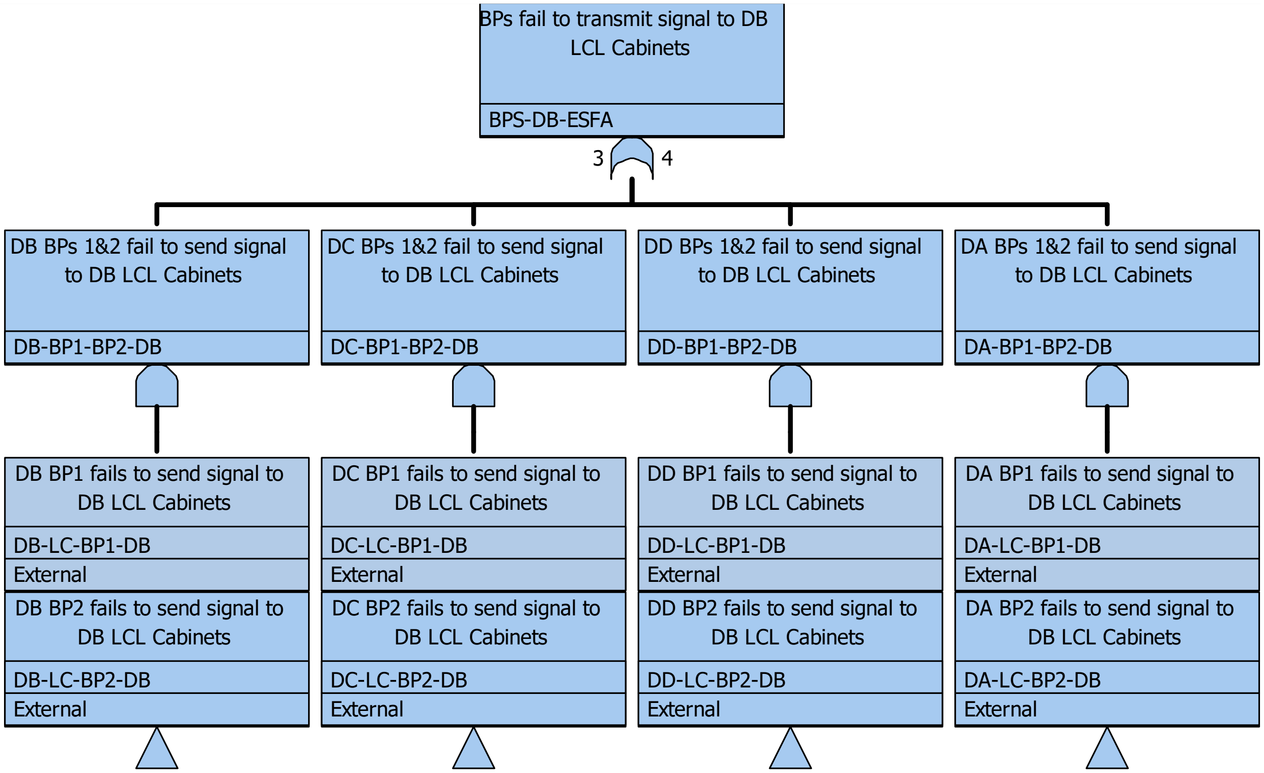

BPs fail to transmit signal to DB LCL Cabinets
BPS-DB-ESFA
3
4
DB BPs 1&2 fail to send signal to DB LCL Cabinets
DB-BP1-BP2-DB
DC BPs 1&2 fail to send signal to DB LCL Cabinets
DC-BP1-BP2-DB
DD BPs 1&2 fail to send signal to DB LCL Cabinets
DD-BP1-BP2-DB
DA BPs 1&2 fail to send signal to DB LCL Cabinets
DA-BP1-BP2-DB
DB BP1 fails to send signal to DB LCL Cabinets
DB-LC-BP1-DB
External
DB BP2 fails to send signal to DB LCL Cabinets
DB-LC-BP2-DB
External
DC BP1 fails to send signal to DB LCL Cabinets
DC-LC-BP1-DB
External
DC BP2 fails to send signal to DB LCL Cabinets
DC-LC-BP2-DB
External
DD BP1 fails to send signal to DB LCL Cabinets
DD-LC-BP1-DB
External
DD BP2 fails to send signal to DB LCL Cabinets
DD-LC-BP2-DB
External
DA BP1 fails to send signal to DB LCL Cabinets
DA-LC-BP1-DB
External
DA BP2 fails to send signal to DB LCL Cabinets
DA-LC-BP2-DB
External

Figure 28. Sub fault tree showing the failures of all bistable processors.

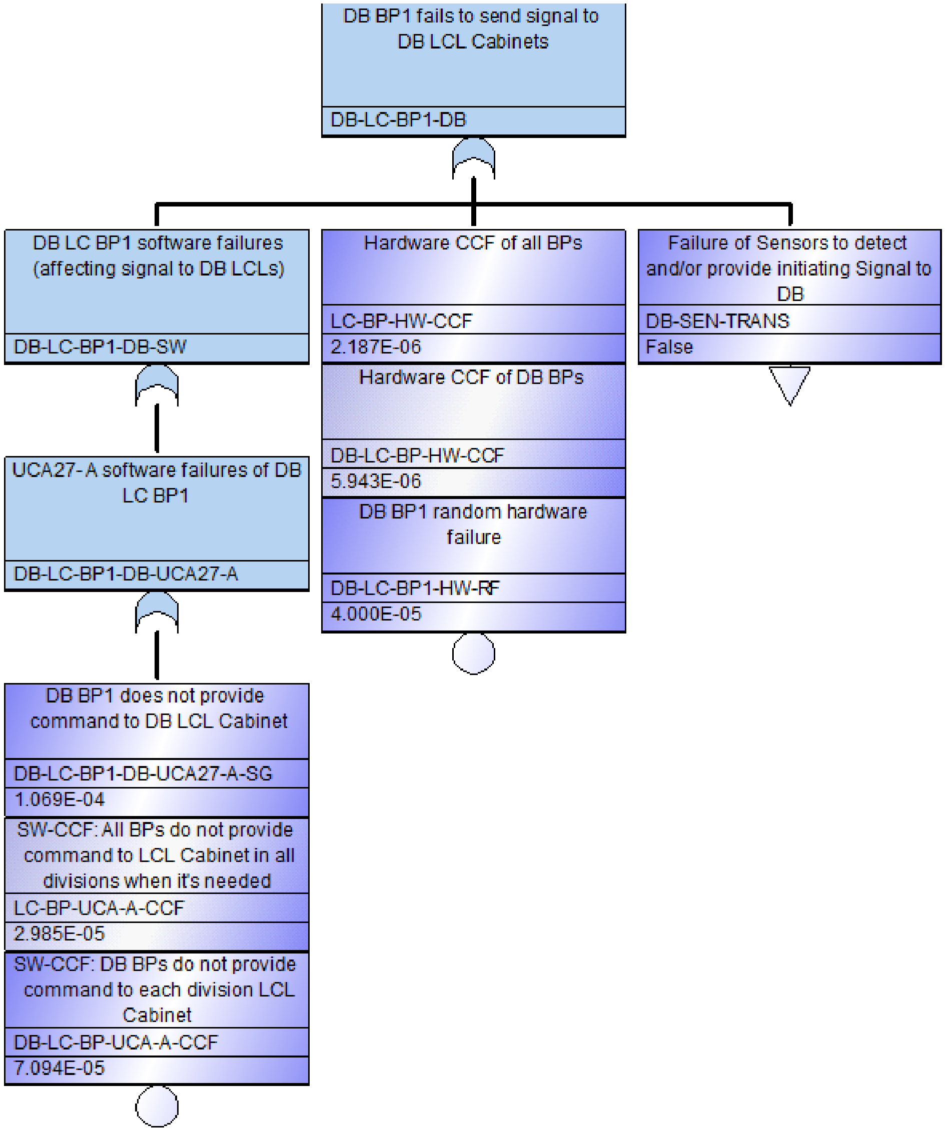

DB BP1 fails to send signal to DB LCL Cabinets
DB-LC-BP1-DB
DB LC BP1 software failures (affecting signal to DB LCLs)
DB-LC-BP1-DB-SW
Hardware CCF of all BPs
LC-BP-HW-CCF
2.187E-06
Failure of Sensors to detect and/or provide initiating Signal to DB
DB-SEN-TRANS
False
Hardware CCF of DB BPs
DB-LC-BP-HW-CCF
5.943E-06
UCA27- A software failures of DB LC BP1
DB-LC-BP1-DB-UCA27-A
DB BP1 random hardware failure
DB-LC-BP1-HW-RF
4.000E-05
DB BP1 does not provide command to DB LCL Cabinet
DB-LC-BP1-DB-UCA27-A-SG
1.069E-04
SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it's needed
LC-BP-UCA-A-CCF
2.985E-05
SW-CCF: DB BPs do not provide command to each division LCL Cabinet
DB-LC-BP-UCA-A-CCF
7.094E-05

Figure 29. Sub fault tree showing the failures of a single bistable processor in the baseline design.

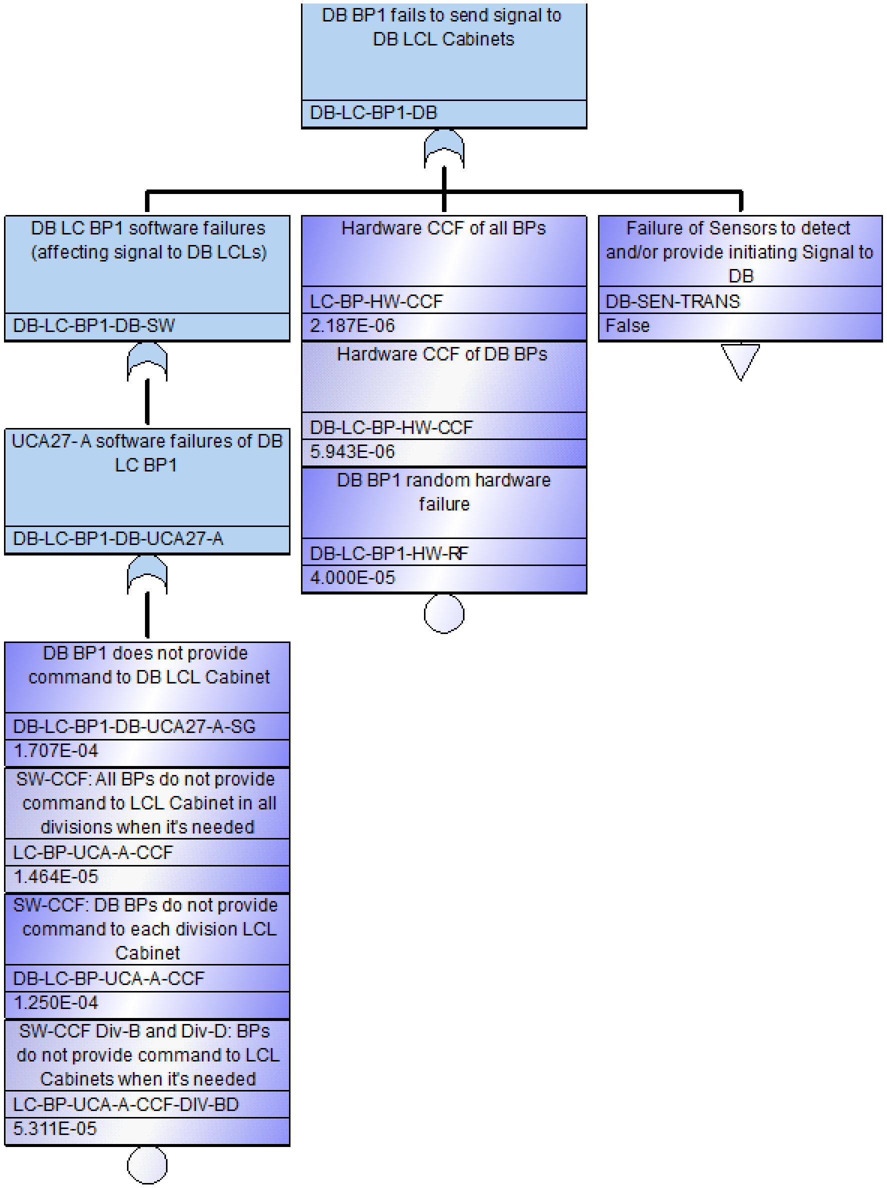

DB BP1 fails to send signal to DB LCL Cabinets
DB-LC-BP1-DB
DB LC BP1 software failures (affecting signal to DB LCLs)
DB-LC-BP1-DB-SW
Hardware CCF of all BPs
LC-BP-HW-CCF
2.187E-06
Hardware CCF of DB BPs
DB-LC-BP-HW-CCF
5.943E-06
DB BP1 random hardware failure
DB-LC-BP1-HW-RF
4.000E-05
Failure of Sensors to detect and/or provide initiating Signal to DB
DB-SEN-TRANS
False
UCA27-A software failures of DB LC BP1
DB-LC-BP1-DB-UCA27-A
DB BP1 does not provide command to DB LCL Cabinet
DB-LC-BP1-DB-UCA27-A-SG
1.707E-04
SW-CCF: All BPs do not provide command to LCL Cabinet in all divisions when it's needed
LC-BP-UCA-A-CCF
1.464E-05
SW-CCF: DB BPs do not provide command to each division LCL Cabinet
DB-LC-BP-UCA-A-CCF
1.250E-04
SW-CCF Div-B and Div-D: BPs do not provide command to LCL Cabinets when it's needed
LC-BP-UCA-A-CCF-DIV-BD
5.311E-05

Figure 30. Sub fault tree showing the failures of a single bistable processor in the diverse design.

Table 39. Basic event values employed by the baseline study for the reactor trip system.

Component	Individual Failure	Rack-CCF	Division-CCF	All Component CCF	Total Failure
Digital output module	1.640E-05	1.706E-06	1.015E-06	1.983E-07	1.932E-05
Selective relay	6.200E-06	N/A	6.073E-07	7.059E-08	6.878E-06
Reactor trip breaker – Undervoltage device	1.700E-03	N/A	N/A	1.763E-05	1.718E-03
Reactor trip breaker – Shunt device	1.200E-04	N/A	N/A	1.244E-06	1.212E-04
Reactor trip breaker	4.500E-05	N/A	N/A	1.944E-06	4.694E-05
Bistable processor – Hardware	4.000E-05	N/A	5.943E-06	2.187E-06	4.813E-05
Local coincidence logic processor – Hardware	6.480E-05	1.076E-05	7.647E-06	3.961E-06	8.717E-05
Bistable processor – Software	1.069E-04	N/A	7.094E-05	2.985E-05	2.077E-04
Local coincidence logic processor – Software	8.976E-05	N/A	N/A	1.179E-04	2.077E-04

Table 40. Basic event values employed by the baseline study for the engineered safety features actuation system.

Component	Individual Failure	Rack-CCF	Division-CCF	All Component CCF	Total Failure
Component interface module – Hardware	4.000E-05	N/A	N/A	2.095E-06	4.209E-5
Loop controller – Hardware	4.000E-05	N/A	N/A	2.095E-06	4.209E-5
Group controller – Hardware	4.000E-05	N/A	5.973E-06	2.402E-06	4.838E-05
Loop controller – Software	1.393E-04	N/A	N/A	6.842E-05	2.077E-04
Group controller – Software	6.207E-07	N/A	1.179E-4	8.914E-05	2.077E-04
Bistable processor – Hardware	4.000E-05	N/A	5.943E-06	2.187E-06	4.813E-05
Local coincidence logic processor – Hardware	6.480E-05	1.076E-05	7.647E-06	3.961E-06	8.717E-05
Bistable processor – Software	1.069E-04	N/A	7.094E-05	2.985E-05	2.077E-04
Local coincidence logic processor – Software	8.976E-05	N/A	N/A	1.179E-04	2.077E-04