



3 4456 0374729 3

ORNL/TM-12234

**ornl****OAK RIDGE  
NATIONAL  
LABORATORY****MARTIN MARIETTA**

# Switchover Software Reliability Estimate for Paducah Freezer/Sublimator Computer Systems

April 1993

Deborah M. Flanagan  
Statistics Group  
Mathematical Sciences Section  
Engineering Physics and Mathematics Division  
Oak Ridge National Laboratory

J. Neil Davis  
Real Time Computer Applications Department  
Instrument and Electrical Engineering Division  
K-25 Site

---

Research sponsored by Instrument and  
Electrical Engineering Division at the K-25  
Site.

---

Prepared by the  
OAK RIDGE NATIONAL LABORATORY  
Oak Ridge, Tennessee 37831  
MARTIN MARIETTA ENERGY SYSTEMS, INC.  
for the  
U.S. DEPARTMENT OF ENERGY  
under Contract No. DE-AC05-84OR21400

**OAK RIDGE NATIONAL LABORATORY****CENTRAL RESEARCH LIBRARY**

CIRCULATION SECTION

4500N ROOM 175

**LIBRARY LOAN COPY****DO NOT TRANSFER TO ANOTHER PERSON**

If you wish someone else to see this  
report, send in name with report and  
the library will arrange a loan.

RCN7983 (6-87)

MANAGED BY  
MARTIN MARIETTA ENERGY SYSTEMS, INC.  
FOR THE UNITED STATES  
DEPARTMENT OF ENERGY

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615) 576-8401, FTS 626-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express 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. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

# Switchover Software Reliability Estimate for Paducah Freezer/Sublimator Computer Systems

April 1993

Deborah M. Flanagan  
Statistics Group  
Mathematical Sciences Section  
Engineering Physics and Mathematics Division  
Oak Ridge National Laboratory

J. Neil Davis  
Real Time Computer Applications Department  
Instrument and Electrical Engineering Division  
K-25 Site

---

Research sponsored by Instrument and  
Electrical Engineering Division at the K-25  
Site.

---

Prepared by the  
OAK RIDGE NATIONAL LABORATORY  
Oak Ridge, Tennessee 37831  
MARTIN MARIETTA ENERGY SYSTEMS, INC.  
for the  
U.S. DEPARTMENT OF ENERGY  
under Contract No. DE-AC05-84OR21400





## Table of Contents

Executive Summary		vii
Abstract		ix
1.0	Introduction	1
2.0	Background	2
3.0	Method	5
3.1.	Some Reliability Estimation Issues	5
3.2.	Musa's Software Reliability Models	8
3.3.	Some Other Reliability Distributions	9
3.4.	Bayesian Methods	11
3.4.1	Bayesian Uniform Prior for Estimating Reliability with Binomial Trials	12
3.4.2	Bayesian Truncated Uniform Prior for Estimating Reliability with Binomial Trials	13
3.4.3	Bayesian Beta Prior for Estimating Reliability with Binomial Trials	14
3.5.	Reliability Growth Models for Hardware Studies	15
3.6.	Software Input Space and Testing Coverage Estimation Methods	17
4.0	Results	22
4.1.	Switchover Software Reliability Estimate	22
4.1.1	Binomial Liability Estimate	22
4.1.2	Bayesian Liability Estimate with Uniform Prior Distribution	24
4.1.3	Bayesian Reliability Estimate with a Truncated Uniform Prior Distribution	25
4.1.4	Bayesian Reliability Estimate with Beta Prior Distribution	28
4.1.5	Comparison of Software Reliability Estimates	29
4.2.	Software Input Space	31
4.3.	Switchover Software Testing Summary	38
4.4.	Testing Coverage Estimate	44
5.0	Summary and Conclusions	47
6.0	References	48
APPENDIX A Introduction to One Software Reliability Model by J. Musa		49
APPENDIX B Switchover Testing Summary		52



## List of Tables

Table 2.1	Hardware Components and Vendor-Provided Reliability Data . . . . .	4
Table 3.1	Partitioned Input Space for Mutually Exclusive Inputs . . . . .	19
Table 3.2	Partitioned Input Space with a Completely Dependent Input . . . . .	19
Table 3.3	Partitioned Input Space with a Partial Dependency . . . . .	20
Table 4.1	Comparison of Reliability Estimators . . . . .	31
Table 4.2	Trip Conditions and Frequencies for the Freezer/Sublimator . . . . .	32
Table 4.3	Functions and Frequencies for the Freezer/Sublimator . . . . .	33
Table 4.4.a	Input Space for the Freezer/Sublimator Switchover Capability . . . . .	37
Table 4.4.b	Input Space for the Freezer/Sublimator Switchover Capability . . . . .	38
Table 4.5	Code Segments for Condition Inputs . . . . .	39
Table 4.6	Code Segments for Function Inputs . . . . .	40
Table 4.7	Redundant Segments Testing Summary . . . . .	41
Table 4.8.a	Coverage Summary for the Freezer/Sublimator Switchover Capability . . . .	45
Table 4.8.b	Coverage Summary for the Freezer/Sublimator Switchover Capability . . . .	46

## List of Figures

Figure 1	Fault Tree . . . . .	4
----------	----------------------	---



## Executive Summary

Twenty four (24) Texas Instruments D3 Process control systems were purchased to monitor the Paducah production line. Each D3 system consists of two redundant processing systems connected by a utility board. For the purposes of this study, it is assumed that the computer hardware will switch properly to the redundant central processing unit (CPU) at the failure of the primary CPU and that the operating system software will perform its relevant functions correctly. The switching from the primary CPU to the redundant CPU on failure is called switchover. It is desirable to know that the software will continue to operate correctly regardless of the CPU in operation.

This document summarizes the switchover software reliability study for the Paducah Freezer/Sublimator (F/S) computer system developed by K-25 Engineering. Before testing began on the system's switchover capability, the application software had been tested on one CPU and debugged until no failures occurred. The resulting software was then tested in switchover mode by creating condition inputs on the process simulator and forcing the CPU to fail by pressing the reset button.

The software reliability estimate is not required but is provided by K-25 Engineering as supporting evidence of their confidence in the system. In the event that the operating environment is different from (1) that expected and planned for by the simulator, or (2) the specifications against which the software has been tested, more errors may occur when the new system is placed in service. Additionally, the reliability and coverage estimates provided in this report are valid only for the software in its existing form. If changes are made to the software, new reliability and coverage estimates must be determined.

The testing of the software in switchover mode consisted of 152 tests in which only one failure occurred at test 43. After that failure, the software was debugged, and testing continued with no additional failures for the remaining 109 tests. The estimated reliability  $\hat{R}$  is .9954 with a 95% probability interval of  $.9934 < R < .9998$ . Coverage estimates how well the testing represented the expected operating environment. The coverage estimate for this reliability estimate is .9948 or 99.48%, that is, almost all the inputs that are expected to occur have been tested successfully in switchover mode. This is a good reliability estimate and a good coverage estimate.

In addition to the calculated results, the report provides an overview of the computer hardware system, discussions on reliability and coverage estimation methods, and a complete summary of switchover software testing activities.

## **Abstract**

K-25 Engineering Division purchased a series of redundant computer systems and developed software for the purpose of providing continuous process monitoring and control for the Freezer/Sublimator equipment in the gaseous diffusion process at the Paducah Gaseous Diffusion Plant. The application software is loaded on two central processing units (CPU) so that in the event of a failure of the primary unit, the processing can switch to the backup unit and continue processing without error. It is the purpose of this document to demonstrate the reliability of this system with respect to its ability to switch properly between redundant CPU. The total reliability estimation problem - which considers the computer hardware, the operating system software, and the application software - has been reduced to one that considers only the application software directly involved in the switchover process. Estimates are provided for software reliability and the testing coverage. Software and hardware reliability models and reliability growth models are considered in addition to Bayesian approaches.



## **1. Introduction**

This document summarizes the software reliability study for the Paducah Freezer/Sublimator (F/S) computer system developed by K-25 Engineering. The initial reliability estimation problem - which considered the computer hardware, the operating system software, and the application software - has been reduced to one that considers only the application software directly involved in the switchover process. For the purposes of this study, it is assumed that the computer hardware will switch properly to the redundant central processing unit (CPU) at the failure of the primary CPU and that the operating system software will perform its relevant functions correctly. Before testing began on the system's switchover capability, the application software had been tested on one CPU and debugged until no failures occurred. The resulting software was then tested in switchover mode by creating alarm condition inputs on the process simulator and forcing the CPU to fail by pressing the reset button on the CPU. The processing time is much faster than the time required to force the manual switchover: therefore, it is difficult to force the CPU to fail at the appropriate times to check (1) all alarm condition software segments and (2) any software segments executing briefly. For this reason, Engineering elected to consider a software reliability estimate based on the completed tests.

The software reliability estimate is not required but is provided by K-25 Engineering as supporting evidence of their confidence in the system. In the event that the operating environment is different from that expected and planned for by the simulator and the specifications against which the software has been tested, more errors may occur when the new system is put into operation.

This report provides (1) an overview of the hardware systems, (2) discussions of some hardware and software reliability estimation methods, (3) a summary of the switchover software testing, (4) software reliability and coverage estimates, and (5) a summary and conclusions. The Background Section describes the hardware and software system. The Methods Section reviews software reliability models studied by Musa (1987), some reliability distributions, hardware reliability growth models, some Bayesian reliability methods, and the meaning of "input space" and "coverage" estimates. The Results Section provides the switchover software reliability estimate, the software input space explanation and estimate, a summary of the switchover software testing, and the testing coverage estimate. Detailed testing results are found in Appendix B.

## 2. Background

The reliability of the system switchover capability has three components: hardware reliability, system software reliability, and application reliability. That is, all three components must work properly for a switchover to be successfully completed. This study is concerned with estimating the reliability of the applications software and assumes that the hardware and operating software will operate properly. This section provides background information about the redundant CPU system.

Twenty four (24) Texas Instruments D3 Process Control Systems were purchased to monitor the Paducah production line. Similar systems are operating at other Department of Energy (DOE) installations, such as the Y-12 Weapons Plant and Savannah River, and at various other commercial sites.

The schematic in Figure 2.1 shows the organization of one redundant CPU system via a high-level fault tree. The D3 computer system consists of two redundant systems connected by a utility board. Each redundant system contains an Intel 8086 or 8386 central processing unit (CPU) and memory with a micro power supply (Micro PS), process input/output (PIO), multiplexor (MUX) input/output (MIO), R-link card, and Multibus Communications Control (MCC) network boards. Identical system and application software resides on both systems. The primary CPU accepts input from the process stream, provides appropriate checks, and outputs results to the process stream or to the operator's console. The R-link communications card keeps the backup system synchronized with the primary system; that is, each time the primary system updates a variable value, that information is sent to the secondary system. At the failure of the primary system, the secondary system should have all the current variable values including the program segment where processing should continue. Failed components in the primary system will be replaced immediately. The backup CPU system will act as the primary system until it fails. The two systems will thus alternately act as primary and backup systems.

A fault tree is developed specifically to determine the actions necessary to cause a specific failure of a system. The reliability topic of this study is that of the ability of the system to switch properly from the primary to the secondary CPU when the primary CPU fails. "Switching properly" means that the switch will be accomplished electro-mechanically and the software will resume processing in the program step. The following notes are background for the fault tree:

1. The TI system software transfers the principle CPU (PCPU) variable values and step counters to the backup CPU (BCPU) via the R-link card as each variable changes value. The R-link card is the paired communication between the CPUs.
2. Application software continually verifies correct, non-failed states for all pumps, valves, etc., for the current process mode of operation. Any discrepancy causes a correction or alarm.
3. The utility board (UB) routinely monitors a watchdog timer in the operating system. If the timer runs out, the UB assumes the PCPU has failed.
4. Software in the PCPU detects any problem reading the multiplexer. If so, the software tells the UB to switch over to the BCPU.

Figure 2.1 shows that a failure of any of four components will cause the system to switchover: the micro PS, the CPU/memory, the PIO, or the MIO. If switchover occurs, the system should continue operating unless the r-link fails or the software switch capability fails. Regardless of this, the system will fail if the utility board fails or the MUX PS fails. Notice that the system will switch properly without the MCC network boards.

The hardware components and vendor-provided reliability data are listed in Table 2.1. No information is available from the vendor on the operating system reliability. The MMES specifications required sufficient redundancy to meet 1 failure in 20 years. No specifications were made on individual components. Assuming all failed components will be replaced instead of repaired, a failure for one year in 20 is approximately equivalent to a reliability of .95 or 95%. This also assumes that the operating period without downtime for maintenance is one year, which seems too long for computer equipment. Therefore, with an operating period of less than a year, a failure of once in 20 years indicates a reliability greater than 95%. That is, the estimated probability that the system will continue to operate without total system failure for 20 years is at least .95. A study by Walraven (1986) was provided as evidence of the ability to meet this specification.

Figure 1. Fault Tree

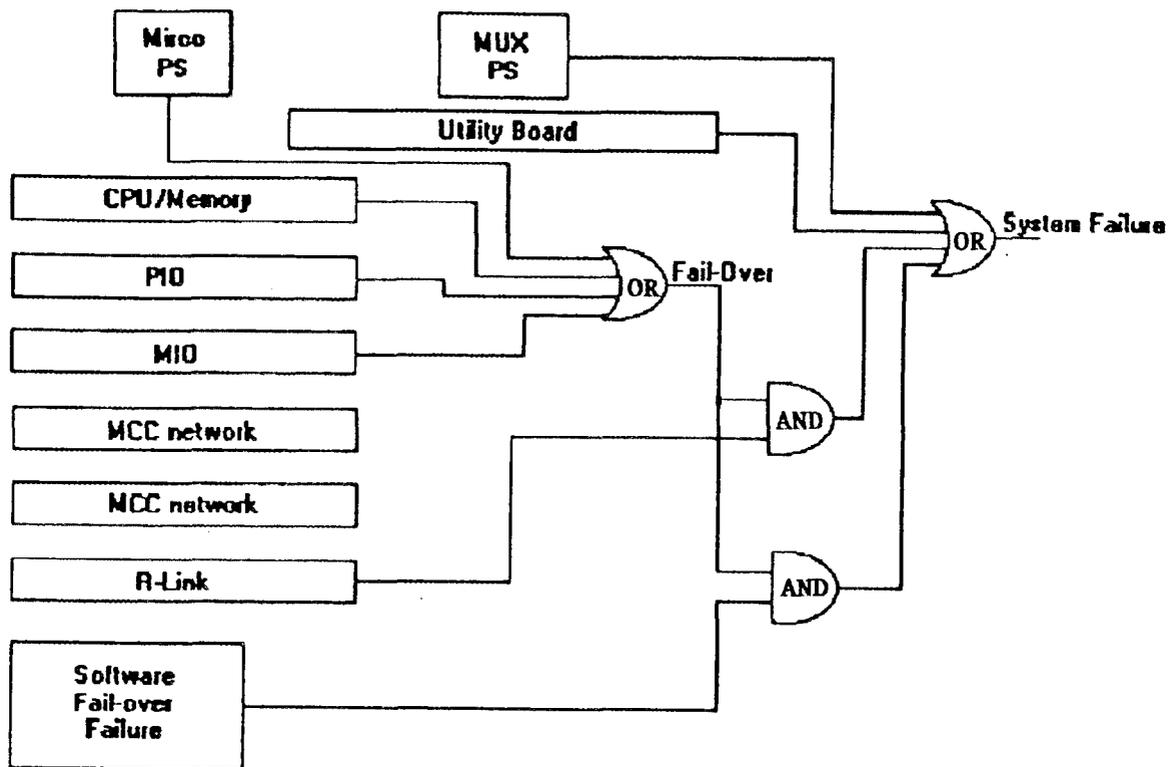


Table 2.1. Hardware Components and Vendor-Provided Reliability Data

System/Component	Vendor-Provided MTBF	Vendor-Provided MTTR
Micro Power Supply	25,876 hrs	1 hr
CPU/Memory Board	39,500	1 hr
PIO Board	30,660	1 hr
MIO Board	30,660	1 hr
Utility Board	54,021	unavailable
MUX Power Supply	17,777	unavailable
R-Link	unavailable	unavailable
MCC Network Board	16,640	unavailable

### 3. Method

Detailed switchover testing information was kept. Of the 152 tests, only one failure occurred at test 43. The software was debugged and reviewed in order to correct other similar, potential faults. No failures occurred in the remaining 109 tests. While this performance is good, the single failure makes reliability estimation difficult. This section will describe the methods and considerations for producing the reliability estimate for the switchover software. Reliability definitions and some issues in reliability estimation are discussed in Section 3.1. Some software reliability methods described by Musa (1987) are summarized in Section 3.2. Some other distributions used in reliability analyses are described in Section 3.3, and Section 3.4 discusses Bayesian reliability methods. Section 3.5 considers some reliability growth models used in hardware reliability studies. Section 3.6 explains the importance of the input space "coverage" concepts and describes how they can be estimated.

#### 3.1 Some Reliability Estimation Issues

Reliability is defined as the probability that an item (software, equipment, etc) will operate without failure under specific conditions for a stated length of time. Reliability is, therefore,  $1 - \text{Pr}(\text{Failure})$ , where  $\text{Pr}(\text{Failure})$  is the probability of failure. Reliability changes with time: the longer an item operates, the more likely it is that it will fail. Reliability and failure estimates are always reported with respect to a point in time. The reliability distribution can be modeled in different ways and needs to be chosen carefully because of the very different reliability estimates that each can produce over time. Observed failures or other prior information are the means of selecting the appropriate distribution.

Reliability estimates are statistics. Calculating accurate statistics requires obtaining data (that is, failures) to verify assumptions and to provide reasonably narrow confidence intervals. The performance of 1 failure in 152 makes selecting a reliability distribution more difficult. Routine recommended practice in reliability estimation would be to request more testing so that more failures might occur. However, for this study, the data collection was started when a higher level of reliability had already been demonstrated. This means that the data collected represent the top of a reliability growth curve, which is designed to monitor improvements in reliability based on improvements made to equipment or software. If there are no data for the initial time frame of the curve, then the initial time frame reliability cannot be estimated. Another important point is

that reliability testing that results in no failures produces a reliability estimate equal to one (1). While this is a correct estimate from a testing and mathematics stand point, users of the software may not consider it a credible estimate. Therefore, it is worthwhile to find a method that correctly produces a reliability estimate that is not equal to one (1).

There are several ways of conducting reliability tests. The following cases provide a background for the F/S software reliability issues and methods:

1. Much reliability testing, especially for equipment, tests a group of homogeneous items either until a certain number fail or until a certain amount of time elapses. The time that each item fails and the nature of the failure are carefully recorded. This failure data is analyzed to determine the reliability.
2. In some other cases, a single item may be tested repeatedly with the same or different missions. Again, the failures and times of failure are recorded carefully and analyzed.
3. Usually, there is some target reliability that is required, so that reliability tests may continue in stages with improvements made to each surviving unit at each stage. The reliability is estimated for each stage of development.

In each of the above cases, the reliability is estimated for a group of like (homogeneous) units that receive identical treatment over a specified period of time. Note that when the items are adjusted, as in case 3, a new reliability estimate is needed for the improved items. Similarly, if the testing conditions are changed in a way that might reduce or increase stress, then the reliability must also be re-estimated using test data from the new environment or some other means of adjusting the data. If it is necessary to estimate reliability for a point in time that is beyond the planned testing period, then either the test period needs to be increased or accelerated testing methods need to be used. Accelerated testing methods are discussed in detail by Nelson (1991) and consider reducing the required test time by increasing test stress factors such as temperature, pressure, weight, etc. These increases accelerate the wear and, therefore, the failures on the tested item.

So, the reliability can be estimated at any of the following times: at each stage of development, when expected operating conditions change, or only at the end of product development. Of course, estimates can be made for design change considerations, too.

Software reliability models are a little different from hardware reliability models. Usually one software program is tested repeatedly. Each time a failure occurs, the software fault that caused the failure is fixed so that the failure cannot recur. A fault in the software may cause

multiple failures. The time and nature of the failures are recorded and analyzed. It is assumed a priori that a certain number of faults exist in the code. A number of measures are made to describe the software size and complexity. Models that predict reliability as a function of CPU time have been shown to be good predictors of software reliability. Again, these models require failure data. If lengthy testing produces no failures, it seems that the reliability must be good, but it cannot be estimated very well since no failure has occurred.

An additional concern in software reliability is the testing itself. What inputs is the program expected to receive and process and with what frequency, that is, what is the input space? Which inputs will be tested? How well does the testing cover the variety of inputs expected in the processing environment? The "coverage" is the estimate of how well the testing represents the expected operating environment. Ideally, the inputs tested would represent the operating environment, that is, the percentage of testing for each input would be the same as the expected operating frequency for that input. Thorough coverage gives more confidence and credibility to the software reliability estimate.

The final statistical issue is confidence interval estimation. Confidence intervals attempt to answer the question, "how good is that estimate?" Every statistical estimate is based on observed data from testing homogeneous units. Homogeneous units themselves are not identical: there is some variation in their makeup. This variation causes the testing and operating experiences to vary from unit to unit, which causes variation in the statistical estimates of quantities like reliability. This means that if an experiment is repeated, the result is expected to be different; but, it should be close to the real value. The confidence interval is a range that the true value might be in with a certain amount of confidence. For example, if a 95% confidence interval is calculated, the true value should fall within the range calculated by this procedure 95% of the time. The more confidence that is required, the wider the confidence interval. Another important factor in confidence interval width is the number of observations used to calculate the estimate: the fewer observations, the wider the confidence interval.

With these concepts and issues in mind, the method for analyzing and estimating the reliability of the alarm switchover software reliability is as follows:

1. Consider different models for estimating reliability and provide a software reliability estimate.
2. Estimate the software input space.

3. Summarize the switchover testing experience.
4. Estimate the testing coverage.

### 3.2 Musa's Software Reliability Models

The software reliability models developed by Musa, Ianino, and Okumoto are documented in their books (1987 and 1990) and numerous journals such as *IEEE Transactions on Software Engineering*, *IEEE Transactions on Reliability*, *Journal of Systems and Software*, and *IEEE Software*. The models discussed by Musa et al (1990) develop estimates of the reliability at a point in time. This reliability represents the accumulated reliability resulting from testing, correction of failures, and continued testing and correction. In this manner, the growth or demise of reliability can be tracked throughout the testing and development process. This capability can help determine expected completion dates, additional manpower needs, and other important concerns of software development. This section will explain why the Musa models were not used for the F/S study and provide a very limited overview of the types of information needed to estimate the models.

Even though the software reliability models described by Musa et al (1987, 1990) can provide valuable information when applied properly, they were not used for the F/S study for four major reasons:

1. Many assumptions were required, some of which were arbitrary and others required data to estimate or verify. The single failure that occurred during the test period provided an insufficient amount of data to verify the assumptions. If plans had been made in advance to do a software reliability study, much of the needed data could have been collected.
2. The Musa models are intended for software development projects that might market or otherwise put software into operation that is not completely debugged but whose carefully analyzed debugging and testing history provides an acceptable failure rate. The additional failures of such software products may be corrected in the next version. The F/S software was tested to zero failures on one CPU and was tested with success for almost all of the alarm conditions. It is not feasible to put a version of the F/S software into operation with potential errors in the alarm software segments.
3. The Musa models are based on failures over time and do not take advantage of the large number of successful trials.
4. Testing data for the F/S study was collected after most failures had been detected, corrected, and retested. Only the upper portion of the reliability growth curve had been recorded so that the total growth curve could not be predicted.

Appendix A contains introductory information about one of the Musa models. Many more details and applications are found in Musa et al (1987, 1990).

### 3.3 Some Other Reliability Distributions

A number of distributions are used in reliability estimation, and the use of each depends on the application, that is, the behavior of the failure data. This section considers the applicability of well-known parametric, non-parametric, and censored data techniques for reliability estimation. The binomial distribution is chosen as the classical statistics technique most appropriate for this study.

As already noted, there are too few failures in the F/S switchover software problem to use well known distributions such as the Weibull, lognormal, exponential, gamma, etc. These distributions all consider the reliability as a function of the failure time. Observed failure data are used to perform statistical tests and decide which distribution the data most likely represent. With too few failures, it is not possible to determine which of these distributions is the correct one to use for predicting reliability. The behavior of these distributions over time can be very different, so that it is important to choose the correct one. In the F/S study, only one failure has occurred, and, therefore, it is not possible to use any of these distributions for reliability estimation. Two good references in this area are Nelson (1991) and Kecegloru (1991).

Nonparametric reliability estimation techniques are appropriate when (1) it is not possible to match the data to a particular distribution (whether due to fewer observed data points or just a lack of fit) and (2) data are censored. The nonparametric methods are not the most appropriate for the F/S study because the single failure will provide a reliability estimate with a very wide confidence interval. Nelson (1991) is one of the many references in this area.

Another way to approach the F/S problem is to try to take advantage of the large number of trials by considering the binomial distribution. The binomial distribution will be used in estimating the switchover software reliability because of its ability to use the number of successful tests in the analyses. To apply the binomial distribution to the F/S problem, we must assume that each test can be considered an independent trial and that the probability that the trial is a success is the same for every trial. This is reasonable because (1) no test must be preceded by any other particular test and (2) every trial has the same expected probability of success (no failure is expected because no failures occurred at the finish of the testing of all inputs on one CPU). The

single parameter to be estimated for the binomial distribution is the probability of success  $p$ , which is estimated as  $\hat{p} = \frac{\text{the number of successes}}{\text{the number of tests}}$ . The estimate of  $p$  is also the estimate of reliability, the probability of a success.

A confidence interval can be calculated as described by Henley and Kumamoto (1981). The confidence intervals are calculated for a specified level of confidence,  $100 \times (1-\alpha)\%$ . For example, for a 95% confidence level,  $\alpha = .05$ . The upper confidence limit  $R_u$  estimates that for  $100 \times (1-\alpha)\%$  of the repetitions of this study, we do not expect the reliability to exceed  $R_u$ . The lower confidence limit  $R_L$  estimates that for  $100 \times (1-\alpha)\%$  of the repetitions of this study, we do not expect the reliability to be lower than  $R_L$ . The one-sided confidence interval considers only the upper or lower limit. A two-sided confidence interval contains both the upper and lower limits.

The upper confidence limit is

$$\sum_{y=Y}^N \frac{N!}{(N-y)!y!} R_u^{N-y} [1-R_u]^y = \alpha, \text{ for } y \neq 0, \quad (3.1)$$

$$(R_u = 1, \text{ for } y=0) \quad ,$$

where,

$N$  = number of trials,  
 $R_u, R_L$  = upper, lower reliability limits,  
 $\alpha$  = confidence level, and  
 $Y$  = number of failures observed.

The lower confidence limit is

$$\sum_{y=0}^Y \frac{N!}{(N-y)!y!} R_L^{N-y} [1-R_L]^y = \alpha, \text{ for } Y \neq N, \quad (3.2)$$

$$(R_L = 0, \text{ for } y=N) \quad .$$

Both equations can be solved for  $R_L$  or  $R_u$  by iterative methods.

If there are no failures, then a lower confidence limit can be calculated by

$$R_L = (\alpha)^{\frac{1}{N}} \quad . \quad (3.3)$$

An important note on using the binomial distribution is that the number of trials should be specified in advance. In this study, the trials were stopped due to a failure or lack of time.

### 3.4. Bayesian Methods

This section contains a discussion of some Bayesian reliability methods. First a comparison is made between classical and Bayesian statistical methods. Then, three Bayesian prior distributions are described which can all be used with the binomial trials of the F/S study to estimate the reliability, ie, the probability of success  $p$  from the binomial distribution.

Classical statistics attempt to estimate parameter values from a population by drawing a sample from the population and calculating the appropriate statistic used to estimate the parameter of interest. Then a calculated confidence interval provides the range of values that the parameter can be expected to have with some level of confidence. The parameter estimate is fixed but unknown and is estimated using data only from the current experiment or study.

Bayesian statistics use statistical relationships defined by Bayes' law of conditional probabilities. A parameter is assumed to have a known **distribution** instead of being fixed. This distribution is estimated using prior data or other information that can be justified and quantified with reasonable results. Then, using the conditional Bayes relationships, the data from the current study or experiment is combined with the prior information about the distribution of the parameter in order to provide an estimate of the parameter which includes as much useful information as possible. The new data from the current study or experiment can be thought of as updating the prior information about the parameter. The prior distribution chosen to represent the parameter must be justifiable.

A common use of Bayesian methods is when little current data is available. Classical methods usually require more data to have the desired levels of confidence in the calculations. Additional information about the philosophies and theories of the Bayesian methods can be found in Berger (1985), and Martz and Waller (1991).

Bayesian methods are applicable to the F/S study for the following additional reasons:

1. Many binomial trials are needed to support a high classical probability of success (reliability) with 95% confidence. In fact, 109 trials with no failures provide a classical lower bound on reliability of only .973 with 95% confidence.
2. Bayesian methods incorporate the prior binomial trials and new binomial trials in the calculations and produce a reliability estimate that is not equal to one (1).

### 3. Bayesian methods may produce smaller ranges of parameter estimates.

Three different prior distributions are calculated for the F/S study in order to show how the results change depending on what kind of assumptions can **validly** be made about the prior distribution. Each prior distribution and the final calculations that use both the prior and current data are discussed in sections 3.4.1. through 3.4.3. The conditional relationships used to determine the final calculations are the same throughout but are demonstrated only for the first case, the uniform prior, because it is the simplest case.

Bayesian statistics are calculated based on a loss function  $L$ , which describes the difference between the true parameter value and the calculated estimate. The squared- error loss function is used for all the methods in this section. The Bayesian estimated ranges for the reliability parameter are actual probability intervals instead of confidence intervals because the parameter has a distribution. The confidence interval of classical statistics either contains the true value of the parameter or does not, but is **expected** to contain it for  $100(1-\alpha)\%$  (the confidence level) of the repetitions of the same experiment. The Bayesian estimate, therefore, is a stronger statement about the potential range of the parameter.

#### 3.4.1 Bayesian Uniform Prior for Estimating Reliability with Binomial Trials

Mendenhall and Schaeffer (1973) provide the method to find the Bayes estimator of  $p$ , the probability of success in the binomial distribution (ie, the reliability), based on the observed number of trials  $N$  and number of failures  $y$ . If we assume that we know **nothing** about the value of  $p$ , then we can assume that any value of  $p$  is equally likely. We can, therefore, assume that the range of values that  $p$  can take on is uniformly distributed with probability density function

$$g(p) = \begin{cases} 1, & 0 \leq p \leq 1, \\ 0, & \text{elsewhere} \end{cases} \quad (3.4)$$

Now, let us focus our attention on the observed number of failures  $Y$  in  $N$  trials.  $Y$  is a binomial random variable for a fixed value of  $p$  and has conditional probability density function

$$f(Y|p) = \binom{n}{y} p^y (1-p)^{n-y}, \quad y=0, 1, \dots, N. \quad (3.5)$$

Then the joint distribution of  $y$  and  $p$  is

$$\begin{aligned}
f(y, p) &= f(y|p) g(p) \\
&= f(y|p) \cdot 1, \\
&= \binom{n}{y} p^y (1-p)^{n-y}, \quad y = 0, 1, \dots, n
\end{aligned} \tag{3.6}$$

The marginal distribution of  $y$  is

$$\begin{aligned}
f(y) &= \int_0^1 f(y, p) dp = \binom{n}{y} \int_0^1 p^y (1-p)^{n-y} dp, \\
&= \frac{1}{n+1}, \quad y = 0, 1, \dots, n.
\end{aligned} \tag{3.7}$$

Now, the conditional distribution of  $p$  given the observed number of failures is

$$f(p|y) = \frac{f(y, p)}{f(y)} = (n+1) \binom{n}{y} p^y (1-p)^{n-y}, \tag{3.8}$$

which is a beta function.

Finally, the Bayes estimator of  $p$  is the mean of the conditional distribution  $f(p|y)$

$$\hat{p} = \int_0^1 p f(p|y) dp = \frac{y+1}{n+2}. \tag{3.9}$$

In this example, it was assumed that there was no knowledge of the possible values of  $p$ . If there is some knowledge, the uniform distribution used here will underestimate larger  $p$  and overestimate smaller  $p$ .

### 3.4.2. Bayesian Truncated Uniform Prior for Estimating Reliability with Binomial Trials

Because reliability is a probability, it can only assume values from 0 to 1. Using the uniform prior distribution assumes that nothing is known about the reliability, which is not usually the case. Suppose that previous data and expert opinion show that the reliability is expected to lie only within a certain range. Suppose further that it is not suspected that the reliability should be any particular value within that range, that is, any value in the range is equally likely. Then the truncated uniform distribution  $U(p_0, p_1)$  is appropriate to use as the prior distribution for the parameter  $p$  (reliability). The probability density function for this distribution is

$$g(p; p_0, p_1) = \begin{cases} \frac{1}{p_1 - p_0}, & 0 \leq p_0 < p < p_1 \leq 1, \\ 0, & \text{elsewhere,} \end{cases} \tag{3.10}$$

where

$$\begin{aligned} p_0 &= \text{the lower bound for } p, \text{ and} \\ p_1 &= \text{the upper bound for } p. \end{aligned}$$

The estimate of  $p$  from the failed group in the F/S study (ie,  $p = .975$  and its confidence intervals) can be used to set the bounds for the truncated uniform distribution. It is important to note that the classical estimate of  $p$  and its confidence interval do **not** produce a range of values for  $p$  that are equally likely. In fact, the classical estimator for  $p$  is the most likely value of  $p$  with the likelihood dropping as one moves from  $p$  to either upper or lower bound. This means that the truncated uniform distribution, while it bounds  $p$  better than the uniform distribution, allows more extreme values of  $p$  to be just as likely as the center of the range, the expected value  $\hat{p}$ . Therefore, while the calculations are presented here for demonstration purposes, the use of this prior distribution for  $p$  is not the most applicable.

The estimated probability of success  $\hat{p}$  (ie, the reliability ) is

$$\hat{p} = \frac{x+1}{x+2} \left\{ \frac{I(p_1; x+2, n-x+1) - I(p_0; x+2, n-x+1)}{I(p_1; x+1, n-x+1) - I(p_0; x+1, n-x+1)} \right\}, \quad (3.11)$$

where

$$I(p; a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^p t^{a-1} (1-t)^{b-1} dt, \text{ the standard incomplete beta function,}$$

$$\begin{aligned} x &= \text{number of successes, and} \\ n &= \text{number of trials.} \end{aligned}$$

The 95% two-sided lower probability bound  $p_L$  is the solution to

$$I(p_L; x+1, n-x+1) = (1 - \frac{\alpha}{2}) I(p_0; x+1, n-x+1) + \frac{\alpha}{2} I(p_1; x+1, n-x+1). \quad (3.12)$$

The 95% two-sided upper probability bound  $p_u$  is the solution to

$$I(p_u; x+1, n-x+1) = (1 - \frac{\alpha}{2}) I(p_1; x+1, n-x+1) + \frac{\alpha}{2} I(p_0; x+1, n-x+1). \quad (3.13)$$

### 3.4.3. Bayesian Beta Prior for Estimating Reliability with Binomial Trials

This method assumes a specified range of possible reliability values but also assumes that some values are more likely than others. In fact, this distribution allows the center of the

reliability range to have the highest probability, and the probabilities decrease as one moves from the center to either extreme. This distribution pattern is similar to the confidence interval described previously. It is frequently used as a prior distribution for binomial trials and is a conjugate prior (that is, it has the same distribution as the posterior distribution).

The probability density function for this distribution is

$$g(p; x_0, n_0) = \frac{\Gamma(n_0)}{\Gamma(x_0) \Gamma(n_0 - x_0)} (p^{x_0-1} (1-p)^{n_0-x_0-1}), \quad (3.14)$$

where

$$\begin{aligned} p &= \text{probability of success (reliability),} \\ x_0 &= \text{number of successes, and} \\ n_0 &= \text{number of trials.} \end{aligned}$$

The estimated probability of success  $\hat{p}$  is

$$\hat{p} = \frac{x+x_0}{n+n_0}, \quad (3.15)$$

where

$$\begin{aligned} x_0, n_0 &= \text{as before,} \\ x &= \text{number of additional successes, and} \\ n &= \text{number of additional trials.} \end{aligned}$$

The lower 95% Bayesian probability bound  $p_L$  is

$$p_L = \frac{x+x_0}{x+x_0 + (n+n_0-x-x_0) F_{1-\frac{\alpha}{2}}(2n+2n_0-2x-2x_0, 2x+2x_0)} \quad (3.16)$$

The upper 95% Bayesian probability bound  $p_u$  is

$$p_u = \frac{(x+x_0) F_{1-\frac{\alpha}{2}}(2x+2x_0, 2n+2n_0-2x-2x_0)}{n+n_0-x-x_0+(x+x_0) F_{1-\frac{\alpha}{2}}(2x+2x_0, 2n+2n_0-2x-2x_0)} \quad (3.17)$$

### 3.5 Reliability Growth Models for Hardware Studies

Reliability growth methods which are used for hardware systems were investigated. Kececioglu (1991) summarizes these methods and provides additional references. These methods can use the information from the successful tests. However, their application to the problem at

hand is limited because the testing represents the top of the reliability growth curve. The following reliability growth methods were examined with these conclusions:

1. Gompertz: Reliability estimates from a previous study are required, and no such data is available for the F/S study. This method estimates and predicts the reliability of future tests. The reliability estimates are assumed to follow an increasing, concave, growth curve. The tests for this study show an observed cumulative reliability (assuming a binomial distribution) which starts at 1 and remains 1 until test 43, when it drops to .97826. As additional tests are made and no additional failures occur, the reliability increases to .99218. This observed pattern does not fit the Gompertz Type I growth curve and seems to represent the top of a growth curve.
2. Lloyd-Lipow: Reliability is calculated from observed stages of development. Each stage contains a certain number of trials. For the F/S problem, each test is a stage consisting of one trial. The results are similar to the observed cumulative reliability described above. Another possible way to calculate reliability with this method is to assume two stages with 43 and 109 trials for the pre-failure and post-failure periods, respectively. This calculation, however, provided unreasonable results.

In studies with more reliability information in earlier stages of development, the Lloyd-Lipow results may be used as inputs to the Gompertz model.

3. Gompertz Attribute Methods 1 and 2: These additional methods use attribute data (that is, failures vs successes). Method 2 has (1) a particularly attractive failure discounting feature and (2) the ability to track failure mode behavior. However, both seem to assume the Type I reliability growth curve, as discussed earlier.
4. S-Shaped Gompertz: Instead of the Type I growth curve, this method allows a positive inflection point. The value of the reliability has the same fixed value at the inflection point. This method is appropriate for increasing reliability growth curves.
5. Logistic: This is an S-shaped growth curve which uses the logistic distribution. It also fixes the reliability value at the inflection point and is appropriate for increasing reliability growth curves.
6. Modified Gompertz: This modification allows for any type of inflection point or no inflection point. A short, FORTRAN computer program calculates the revised parameters. This was not calculated because of the dearth of data.
7. U.S. Army Material Systems Analysis Activity (AMSAA): This requires several distributional assumptions, such as failures occurring in a nonhomogeneous Poisson process (NHPP) and a Weibull failure rate. The Weibull failure rate and NHPP cannot be verified with the limited data.

### 3.6 Software Input Space and Testing Coverage Estimation Methods

The software input space is defined as the list of all possible inputs and their respective probabilities of occurrence in the operating environment. This input space is required to determine the coverage of the software testing effort. The coverage is the estimate of how well the testing represented the expected operating environment (also called the operating profile or frequency). Ideally, the inputs tested would represent the operating environment, that is, the percentage of testing for each input category would be the same as the expected operating frequency for that input. Thorough coverage gives more confidence and credibility to the software reliability estimate. For example, suppose a software reliability estimate of .99 is produced from tests that cover only 50% of the input space. Then, the software is expected to work well for 50% of the input space, and it is unknown how well it will work for the other 50% of the input space.

Determining the input space can be time consuming, and most of this section is devoted to explaining some of the mathematics of determining the input space. Once the input space is determined, the coverage calculations are trivial as shown in the latter part of this section.

An input space may be made up of inputs which are partitioned, independent, or a combination of the two. The goal is to create a partitioned input space. This section describes each of these input types and how they effect the input space estimation. The discussion is limited to discrete rather than continuous input values because the F/S system inputs can all be classified as discrete.

A partitioned input space is the set of all possible inputs which has been divided into mutually exclusive inputs or input combinations called partitions. For example, if inputs A and B are mutually exclusive, they cannot occur at the same time. Since the occurrence of input A effects the occurrence of B, we also say that A and B are dependent. This is expressed mathematically as  $\Pr(A|B) = \Pr(AB)/\Pr(B) \neq \Pr(A)$ . This means that the probability that A occurs given that B occurs,  $\Pr(A|B)$ , is not equal to the probability that A occurs,  $\Pr(A)$ . For mutually exclusive events, the probability that A and B occur,  $\Pr(AB)$ , is zero. The  $\Pr(A|B)$ , therefore, is zero regardless of the probability of A or B when A and B are mutually exclusive. Another way to consider this is by breaking down the probability of A,  $\Pr(A)$ , into its components. Still using the simple example with two inputs A and B, the  $\Pr(A)$  is equal to the probability that A and B occur,  $\Pr(AB)$ , and the probability that A occurs but not B,  $\Pr(A \text{ not } B)$ , or  $\Pr(A) = \Pr(AB) +$

$\Pr(A \text{ not } B)$ . An additional feature of a partitioned input space is that the probability summed over all partitions is one.

Independent inputs are not effected by the occurrence of any other inputs. Independence is expressed mathematically as  $\Pr(A|B) = \Pr(AB)/\Pr(B) = \Pr(A)$ . The input space for a set of independent inputs, then, is not a partition but a joint distribution of the inputs. This is represented mathematically as  $\Pr(A=a, B=b) = \Pr(A=a) \times \Pr(B=b)$ . This is read as, "the probability that A has a particular value a,  $\Pr(A=a)$ , and that B has a particular value b,  $\Pr(B=b)$ , is equal to the product of each of those individual probabilities." For discrete events, in particular, the joint probability space can be transformed to a partitioned space by creating partitions of each possible combination of inputs A and B: these partitions will then be mutually exclusive.

A combination of partitioned inputs and independent inputs may occur if some inputs are mutually exclusive and some are independent. If this occurs, then the inputs should be grouped into the partitioned set and the independent sets. Since the independent sets are also independent from the partitioned set, then the whole input space becomes a joint distribution in the same manner as for a totally independent input space. The difference is that at least one of the independent inputs is itself a partitioned set. As for the discrete independent space, this combination space can also be transformed into a partitioned space by creating partitions of all possible combinations.

The idea of estimating an input space, then, becomes one of determining dependencies or independencies so as to create a partitioned input space. The discussion will continue with more ideas and examples concerning dependent inputs and partitioned input spaces.

Some inputs are related, that is, dependent. Independent inputs are not related. These relationships - or lack thereof - are important because they influence the structure of the input space and its probabilities. If no dependencies exist, then the input space probability estimates can be derived directly from a list of all inputs and their expected frequency of occurrence. The frequencies for any dependent inputs, however, need to be reduced by the amount they are influenced by other inputs. This in turn reduces the total input space and changes the probability of each input.

Some examples will help to show how this works. Suppose a software program has only three possible inputs, A, B, and C, with expected frequencies of 4, 4, and 5 times a year. If all

three inputs are mutually exclusive, that is, no input can occur if either of the other two inputs occurs, then the input space is as shown in Table 3.1. The input frequencies are summed for the total number of inputs; then, the probability of each input is estimated by the input frequency divided by the total number of inputs.

**Table 3.1. Partitioned Input Space for Mutually Exclusive Inputs**

Input	Expected Frequency	Input Space Probability
A	4	4/13
B	4	4/13
C	5	5/13
Total Inputs	13	1

The problem changes when some inputs are dependent but not mutually exclusive. Suppose that input A only occurs when input B occurs. This means that A is completely dependent on B. These two inputs, then, should be combined to form one input for the purposes of determining the input space and its probabilities as shown in Table 3.2. Input C is still mutually exclusive of A and B. Notice how the dependency of A on B reduces the total number of possible inputs and increases the probability of input C.

**Table 3.2. Partitioned Input Space with a Completely Dependent Input**

Inputs	Expected Frequency	Probability
A and B	4	4/9
C	5	5/9
Total Inputs	9	1

As a last example, perhaps inputs A and B are only partially dependent on each other. That is, A and B can occur together or alone. An example input space is shown in Table 3.3. The number of inputs and the total frequency have increased over those in Table 3.2. The probability

of input C has decreased. It is important to note that considering A and B as mutually exclusive inputs – when they are not – allows double counting of those times when A occurs with B and underestimates the true probability of C.

**Table 3.3. Partitioned Input Space with a Partial Dependency**

Inputs	Expected Frequency	Probability
A without B	1	1/10
A with B	3	3/10
B without A	1	1/10
C	5	5/10
Total Inputs	10	1

Other input combinations can and do occur in input spaces, but these examples give the reader an idea of the potential complexity of the input space estimation. Further discussions related to input space determination can be found in probability texts such as Harris (1966). It is important to emphasize that the input probability estimates for the F/S System use frequency estimates which are expert opinions of Paducha staff based on operating experience and some reported data.

The final discussion in this section concerns the coverage estimate. The input probabilities represented by the input space provide guidance for planning software testing. Those inputs that are likely to occur most frequently should be tested more. Of course, any input with critical impacts should be tested. If 100 tests are to be made on a system, the probabilities in the input space tables can be used to show what percentage of the tests should include each input. This helps to keep from over - or under - testing inputs. The coverage is the estimate of how well the testing "covered" or represented the input space. Coverage (C) is a simple calculation which requires the input space

$$C = \sum_{i=1}^I P_i * \delta_i ,$$

where,

$P_i$  = the probability that input  $i$  occurs in the input space,  
 $I$  = the total number of inputs, and  
 $\delta_i = 1$  (if input  $i$  is tested) or 0 (otherwise).

The coverage estimate explains that  $C \times 100\%$  of the inputs that occur have been tested. A reliability estimate of 99% with a coverage of 10% shows that the software worked very well with very limited testing. A software user should insist on better coverage. Musa (1990) describes different ways to consider coverage and testing.

## 4. Results

### 4.1. Switchover Software Reliability Estimate

The switchover software reliability estimate was calculated for each method described in Section 3.4 and the binomial method discussed in Section 3.3. Details of each of these calculations are provide in Sections 4.1.1 through 4.1.4. Section 4.1.5 compares the estimates and selects a preferred reliability estimate.

#### 4.1.1 Binomial Reliability Estimate

The total number of 152 test is divided into two groups, the failure group and the post-failure group. The failure group includes the 43 tests ( $N_1$ ) up to and including the failure. The post-failure group includes the 109 tests ( $N_2$ ) after the failure. This division is necessary because the improvements made in the software due to the failure are expected to improve the probability of success in further trials. As noted previously, the probability of success  $p$  in the binomial setting is the reliability estimate we seek. Binomial reliability estimates are calculated separately for the failure and post-failure groups.

The failure group reliability estimate  $\hat{R}$  is calculated simply using the number of trials  $N_1$  and the number of failures  $x$

$$\hat{R} = \frac{N_1 - x}{N_1} = \frac{43 - 1}{43} = \frac{42}{43} = .9767 \quad .$$

This means that the estimate of the probability that the software will operate properly is .9767.

The lower one-sided confidence limit,  $R_L$ , is found by solving the equation

$$\sum_{x=0}^x \frac{N_1!}{(N_1 - x)! (x)!} R_L^{N_1 - x} (1 - R_L)^x = \alpha \quad ,$$

where

$$\alpha = 1 - \frac{\text{confidence percentage}}{100} \quad .$$

For 95% confidence,  $\alpha = .05$ . Substituting the observed values for  $N_1$  and  $x$  and  $\alpha = .05$  yields

$$\frac{43!}{43! 0!} (R_L^{43-0}) (1-R_L)^0 + \frac{43!}{(43-1)! 1!} R_L^{43-1} (1-R_L)^1 = .05 \quad ,$$

which reduces to  $43 R_L^{42} - 42 R_L^{43} = .05$ . This equation is solved iteratively for  $R_L$ . A reasonable solution is  $R_L = .894$  and  $\alpha = .0493$ , which is close to the desired  $\alpha = .05$ .

The 95% confidence two-sided bounds, i.e., upper and lower, are provided by Martz and Waller (1991). The 95% confidence two-sided lower bound on  $R$  is

$$R_L = \frac{X}{X+(n-X+1) F_{1-\frac{\alpha}{2}}(2n-2X+2, 2X)} \quad ,$$

where

$n$  = number of tests,

$X$  = number of successes, and

$F$  = the  $1 - \frac{\alpha}{2}$  value the  $F$  distribution with  $(2n - 2X + 2, 2X)$  degrees of freedom.

Substituting observed values yields

$$\begin{aligned} R_L &= \frac{42}{42+(43-42+1) F_{.975}(2(43)-2(42)+2, 2(42))} \\ &= \frac{42}{42+(2) F_{.975}(4,84)} \\ &= \frac{42}{42+(2) (2.962)} \\ &= .8764 \quad . \end{aligned}$$

The 95% confidence two-sided upper bound on  $R$  is

$$R_u = \frac{(X+1) F_{1-\frac{\alpha}{2}}(2X+2, 2n-2X)}{(n-X)+(X+1) F_{1-\frac{\alpha}{2}}(2X+2, 2n-2X)} \quad .$$

Substituting observed values yields

$$\begin{aligned}
 R_u &= \frac{(42+1) F_{.975(2(42)+2, 2(43)-2(42))}}{(43-42)+(42+1) F_{.975(2(42)+2, 2(43)-2(42))}} \\
 &= \frac{(43) F_{.975(86,2)}}{1+(43) F_{.975(86,2)}} \\
 &= \frac{(43) (39.49)}{1+(43) (39.49)} \\
 &= .9994 \quad .
 \end{aligned}$$

Please note that the 95% confidence one-sided lower bound is to be used only if an upper bound is not calculated.

In summary, the software reliability estimate for the failure group is .9767. If there is interest only in a lower bound on the reliability, then the 95% confidence lower bound is .894. If, however, upper and lower bounds are desired, the 95% confidence interval is  $.8764 < R < .9994$ .

For the non-failure group, only an estimated lower bound on the reliability,  $R_L$ , can be calculated. This is calculated simply as

$$R_L = (\alpha)^{\frac{1}{N_2}} \quad .$$

Substituting  $\alpha = .05$  and the observed  $N_2 = 109$  yields

$$R_L = (.05)^{\frac{1}{109}} = .9729 \quad .$$

Then the 95% lower bound on the reliability for the non-failure group is .9729, a considerable increase over the  $R_L = .894$  (one-sided) or  $R_L = .876$  (two-sided) lower bounds estimated for the failure group. Because of the software improvements made, the appropriate reliability estimate using classical binomial statistics is the 95% confidence lower bound of .9729.

#### 4.1.2. Bayesian Reliability Estimate with Uniform Prior Distribution

As explained in Section 3.4, this calculation assumes that prior knowledge indicates that any value of  $p$ , the probability of success, is equally likely. That is, the prior distribution of the

binomial parameter  $p$  is itself a random variable with uniform distribution. The calculations in Section 3.4 result in the Bayesian estimator

$$\hat{p} = \frac{Y+1}{N_2+2} ,$$

where

$Y$  = number of successes.

Substituting the observed values yields

$$\hat{p} = \frac{109+1}{109+2} = \frac{110}{111} = .99099 \dots$$

From Section 3.4, the 95% confidence lower limit on the estimated reliability is

$$\begin{aligned} R_L &= \frac{x+1}{(x+1)+(n-x+1) F_{1-\frac{.05}{2}}(2n-2x+2, 2x+2)} \\ &= \frac{109+1}{(109+1)+(109-109+1) F_{.975}(2,218+2)} \\ &= \frac{110}{110+3.80} \\ &= .96661 \dots \end{aligned}$$

The Bayesian reliability estimate, then, using a uniform prior distribution for  $R$  is .991. The 95% confidence interval for  $R$  is  $.9666 \leq R \leq .99977$ . The assumption of the uniform prior distribution for  $R$  (that is, that any value of  $R$  is likely) allows very high values of  $R$  to be just as likely as the  $\hat{R} = .9729$  observed in the failure group. In fact, the "equally likely" assumption does not use the information from the failure group. For this reason, the calculation may overstate the true reliability.

#### 4.1.3. Bayesian Reliability Estimate with a Truncated Uniform Prior Distribution

If we can put some bounds on the values of the probability of success  $R$  for a test, then distributions other than the uniform can be used to represent prior information about  $R$ . One of

these is the truncated uniform prior distribution  $U(p_0, p_1)$  which has probability density function for  $R$  of

$$g(R; p_0, p_1) = \begin{cases} \frac{1}{p_1 - p_0}, & 0 \leq p_0 < R < p_1 \leq 1, \\ 0, & \text{elsewhere,} \end{cases}$$

where

$$\begin{aligned} p_0 &= \text{lower bound for } R, \text{ and} \\ p_1 &= \text{upper bound for } R. \end{aligned}$$

An additional restriction on the use of this prior distribution or "prior" is that we must assume or have reason to believe that any value of  $R$  in that range is equally likely. This "prior" will allow the use of information from the failed set of data. The lower 95% confidence bound on  $R$  from the failed group (i.e.,  $R_L = .8764$ ) can be assigned to  $p_0$ . The corresponding upper 95% confidence bound ( $R_u = .9994$ ) can be assigned to  $p_1$ . While this interval is based on an estimate which covers only 95% of a probability space, it is a reasonable choice. The software corrections made after the single failure will most likely provide a higher current reliability estimate than that for the failed group so that the previous 95% lower confidence estimate may be a valid lower bound on the current reliability estimate. The previous 95% upper confidence estimate is high enough to consider as an upper bound on the current reliability estimate.

From Section 3.4 the estimated probability of success  $\hat{R}$  is

$$\hat{R} = \frac{x+1}{n+2} \left\{ \frac{I(p_1; x+2, n-x+1) - I(p_0; x+2, n-x+1)}{I(p_1; x+1, n-x+1) - I(p_0; x+1, n-x+1)} \right\},$$

where

$$I(p; a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^p t^{a-1} (1-t)^{b-1} dt,$$

the standard incomplete beta function ratio. Substituting observed data yields

$$\begin{aligned} \hat{R} &= \frac{109+1}{109+2} \left\{ \frac{I(.9994; 109+2, 109-109+1) - I(.8764; 109+2, 109-109+1)}{I(.9994; 109+1, 109-109+1) - I(.8764; 109+1, 109-109+1)} \right\} \\ &= \frac{110}{111} \left\{ \frac{I(.9994; 111, 1) - I(.8764; 111, 1)}{I(.9994; 110, 1) - I(.8764; 110, 1)} \right\}. \end{aligned}$$

Using Splus statistical software to solve the incomplete beta functions yields

$$\hat{R} = \frac{110}{111} \left\{ \frac{.9355507 - .000000436}{.9361123 - .000000498} \right\} \\ = .9904$$

The 95% probability two-sided lower bound is the solution to

$$I(R_L; x+1, n-x+1) = \left(1 - \frac{\alpha}{2}\right) I(p_0; x+1, n-x+1) + \left(\frac{\alpha}{2}\right) I(p_1; x+1, n-x+1)$$

Substituting observed data yields

$$I(R_L; 109+1, 109-109+1) = \left(1 - \frac{.05}{2}\right) I(.8764; 109+1, 109-109+1) \\ + \left(\frac{.05}{2}\right) I(.9994; 109+1, 109-109+1) \\ = (.975) (.000000498) + (.025) (.9361123) \\ = .000000486 + .023402808 \\ I(R_L; 110, 1) = .02340329$$

The inverse beta function yields a  $R_L = .96644$ .

The 95% probability two-sided upper bound is the solution to the equation

$$I(R_u; x+1, n-x+1) = \left(1 - \frac{\alpha}{2}\right) I(p_1; x+1, n-x+1) + \left(\frac{\alpha}{2}\right) I(p_0; x+1, n-x+1)$$

Substituting observed data yields

$$I(R_u; 109+1, 109-109+1) = \left[1 - \frac{.05}{2}\right] I(.9994; 109+1, 109-109+1) \\ + \left[\frac{.05}{2}\right] I(.8764; 109+1, 109-109+1) \\ = (.975) (.9361123) + (.025) (.000000498) \\ = .9127095$$

The inverse beta function provides  $R_u = .999170$ .

In summary, the Bayesian reliability estimate  $\hat{R}$  using a truncated uniform prior distribution is .9904 with a 95% two-sided Bayesian probability interval of .9664 to .9992.

#### 4.1.4. Bayesian Reliability Estimate with Beta Prior Distribution

The last Bayesian reliability estimate presented is the binomial test using a beta prior distribution. This method assumes a specified range of possible reliability values but also assumes that some values are more likely than others.

The estimated reliability  $\hat{R}$  is

$$\hat{R} = \frac{x+x_0}{n+n_0} ,$$

where

- $x_0$  = beta parameter from Martz and Waller,
- $n_0$  = beta parameter from Martz and Waller,
- $x$  = number of additional successes, and
- $n$  = number of additional tests.

Appendix C of Martz and Waller (1991) gives beta distribution parameters  $X_0$  and  $n_0$  that correspond to estimates of the 50 percentile and either the 5 percentile or 95 percentile of the parameter distribution. The percentile estimates may be based on data, judgment, or a combination of data and judgment. For these calculations, the  $n_0 = 58.52051$  and  $x_0 = 57.35010$  estimates from Martz and Waller correspond to the expected reliability  $\hat{R} = .9767$  and 95% one-sided upper confidence bound of .9988 calculated from the binomial trials of the failed group.

Substituting observed and tabled values yields

$$\hat{R} = \frac{109+57.35010}{109+58.52051} = .99301 .$$

The lower 95% Bayesian probability estimate  $R_L$  is

$$R_L = \frac{x+x_0}{x+x_0+(n+n_0-x-x_0) F_{1-\frac{\alpha}{2}}(2n+2n_0-2x-2x_0, 2x+2x_0)} .$$

Substituting observed values yields

$$\begin{aligned}
 R_L &= \frac{109+57.35010}{109+57.35010+(109+58.52051-109-57.35010) F_{.975} (2, 302)} \\
 &= \frac{166.3501}{166.3501+(1.17041) (3.80)} \\
 &= .97396
 \end{aligned}$$

The upper 95% Bayesian probability estimate  $R_U$  is

$$R_U = \frac{(x+x_0) F_{1-\frac{\alpha}{2}} (2x+2x_0, 2n+2n_0-2x-2x_0)}{n+n_0-x-x_0+(x+x_0) F_{1-\frac{\alpha}{2}} (2x+2x_0, 2n+2n_0-2x-2x_0)}$$

Substituting the observed values yields

$$\begin{aligned}
 R_U &= \frac{(109+57.35010) F_{.975} (302, 2)}{109+58.52051-109-57.35010+(109+57.35010) F_{.975} (302, 2)} \\
 &= \frac{(166.3501) (39.49)}{1.17041+(166.3501) (39.49)} \\
 &= .9998
 \end{aligned}$$

Using the previous values of  $n_0 = 43$  and  $x_0 = 42$  produces results very close to these.

In summary, the estimated Bayesian reliability using a beta prior distribution is .9934, and the 95% Bayesian probability range on that estimate is .9754 to .9998.

#### 4.1.5. Comparison of Software Reliability Estimates

The software reliability estimates are summarized in Table 4.1, which includes the assumptions required for each calculation method. It is interesting that the reliability estimate increases and the probability interval ranges decrease as more of the available information is used in selecting the prior distribution of the reliability parameter  $R$  (ie, the binomial parameter  $p$ ). When selecting the appropriate calculation, it is important to consider the assumptions required for the calculation, how well they can be applied to the problem at hand, and the reasonableness of the results. All assumptions can be met for the classical binomial calculation for the non-failed group as explained in Section 3.3.

Both of the Bayesian methods using the uniform and truncated uniform distributions as priors can be eliminated. The uniform distribution prior for  $p$  is inappropriate because it assumes that we know nothing about  $p$ , and, therefore, it is equally likely that  $p$  can take on any values between 0 and 1. Because of the failed-group experience, we know that  $p$  should have values above .9. The truncated uniform distribution allows us to restrict the values of  $p$  to the confidence interval of the failed group. Allowing  $p$  to be equally likely within that range, however, allows equal weight to be given to the lower end of the range of possible  $p$  values. This produces an expected reliability estimate of .966, which is even lower than the classical estimate of .975 from the failed group and does not make sense. It seems that using a prior distribution which allows both bounding  $p$  and applying heavier weights for the more likely values of  $p$  is more appropriate. The beta distribution makes this possible. Martz and Waller (1991) note that the beta distribution is frequently selected as a prior for the parameter  $p$  in binomial trial situations, such as the F/S study. Because the software was improved after the single failure, it is believed that the reliability of the improved software should be at least as good - and maybe better - than that represented by the failed group. For this reason, using the failed group estimates as boundaries of the success parameter  $p$  for the beta prior is appropriate.

The Bayesian reliability estimate using the beta prior distribution is  $R = .9934$  with a 95% probability interval of  $.9754 < R < .9998$ . This Bayesian result and the non-failed group classical method provide essentially the same lower bounds. The Bayesian method additionally provides an estimated reliability that is not equal to one and an upper bound. The result from either the Bayesian method with the beta prior or the classical binomial method calculated from the non-failed group can be quoted as the appropriate estimate. The author prefers the Bayesian estimate because it provides an estimate of reliability that is not equal to 1.

**Table 4.1 Comparison of Reliability Estimators**

Group	Method	$\hat{R}$	$\hat{R}_L$	$\hat{R}_U$	Assumptions
Failed	Classical: Binomial	.975	.8764	.9994	$p$ is fixed and unknown
Non-Failed	Classical: Binomial	--	.9729	--	$p$ is fixed and unknown
Non-Failed	Bayesian: Uniform prior with Binomial trials	.9909	.9666		$p$ has a uniform distribution
Non-Failed	Bayesian: Truncated Uniform prior with Binomial trials	.9904	.9664	.9992	$p$ has a truncated uniform distribution which uses the classical binomial confidence interval from the failed group as its bounds.
Non-Failed	Bayesian: Beta prior with Binomial trials	.9934	.9754	.9998	$p$ has a beta distribution which uses the classical binomial parameters from the failed group as its parameters.

#### 4.2. Software Input Space

This section will describe the following: (1) the software inputs and their potential annual frequency, (2) the independence or dependence of the inputs, and (3) the resulting partitioned input space and probabilities. The results will proceed using these steps:

1. List and describe the inputs to the software and their estimated annual frequency of occurrence.
2. Determine dependencies among the inputs, that is, determine whether the occurrence of one input influences the occurrence of another input.
3. Based on steps 1 and 2, reduce the input space to its final partitioned form.

The F/S software has two categories of inputs, those initiated by the hardware or software and those initiated by the operator. Actions (called trips) initiated by the hardware and software are listed in Table 4.2. These trips are the result of checks by the software or hardware that result in an incorrect state or situation. Each trip signals an alarm condition or warning and changes to a safer state. Actions initiated by the operator are listed in Table 4.3. These operator actions cause the hardware or software to reset or otherwise change the state of the

**Table 4.2. Trip Conditions and Frequencies for the Freezer/Sublimer**

	<b>Trip Condition</b>	<b>Frequency (per F/S per Yr)</b>	<b>Number of Inputs to Initiate the Alarm</b>
1.	High Weight (trip to cold standby) <sup>a</sup>	25	Any one of 4 analog
2.	High-High Weight (trip to moderate hot standby) <sup>a</sup>	17	Any one of 4 analog
3.	Safety System High Weight <sup>b</sup>	16	Any one of 4 digital
4.	Safety System Low Weight (trip to hot standby) <sup>b</sup>	18	Any one of 4 digital
5.	RCW/Freon Differential Pressure (DP) (trip to hot standby) <sup>a</sup>	41	1 analog
6.	Safety System RCW/Freon DP (trip to criticality mode) <sup>b</sup>	6	Any one of 2 digital
7.	RCW/Freon DP (trip to criticality mode) <sup>a</sup>	6	1 analog and 2 digital
8.	Safety System Power Failure (trip to power fail mode) <sup>b</sup>	5	1 digital
9.	UF6 High Pressure (trip to cold standby with the vent valve open) <sup>a</sup>	1	1 analog
10.	Low Freon Temperature (trip to hot standby) <sup>a</sup>	1	1 analog
11.	Low RCW Temperature (trip to hot standby) <sup>a</sup>	1	1 analog
12.	State error or mode failure (trip to hot standby) <sup>b</sup>	1	Any one of 16 digital

<sup>a</sup> Software trip

<sup>b</sup> Hardware trip.

**Table 4.3. Functions and Frequencies for the Freezer/Sublimer**

Function	Frequency (per F/S per yr)	Number of Inputs to Initiate the Function
1. Operator changes the mode from the DPCS console	730	4 Operator-Initiated inputs by device or skid panel
2. Operator changes the mode from the F/S cabinet	24	5 Operator-Initiated buttons
3. Software Trip Reset	93	Operator Initiated
4. Hardware Trip Reset	45	Operator Initiated
5. Set the Freezer/Sublimer (F/S) to Available	200	Operator Initiated
6. Set the F/S to Unavailable	24	Operator Initiated
7. Set the F/S to Maintenance Mode	24	Operator Initiated
8. Weight Test	8	Operator Initiated
9. Cancel Weight Test	4	Operator Initiated

Freezer/Sublimer (F/S). Each digital input in Table 4.2 has two values (for example, open or closed), and each analog input has a numeric range of values. The operator inputs in Table 4.3 represent a single digital input for each function.

The estimated frequencies per year per F/S for each software or hardware trip are listed in Table 4.2. Table 4.3 lists the estimated frequencies for the operator-initiated actions. These estimated frequencies are based on judgments made by Paducah staff who are responsible for the system. These estimates are best guesses at the frequencies. Monitoring the operating process will provide improved estimates which will be useful for updating this study and for other engineering and operations studies.

Some trip conditions and functions are related, that is, dependent. Independent inputs are not related. These relationships - or lack thereof - are important because they influence the structure of the input space and its probabilities. Each trip and function is discussed with respect to its independence or dependence. If no dependencies exist, then the input space probability estimates can be derived directly from Tables 4.2 and 4.3. The frequencies for any dependent inputs,

however, need to be reduced by the amount they are influenced by other inputs as described in Section 3.6. Each input is discussed - with respect to its purpose and dependencies - in the remainder of this section.

Trip Conditions 1 through 4 all initiate weight alarms. The same analog inputs are used for Conditions 1 and 2; but, Condition 2 is triggered only for higher analog weight values. For Condition 3, (Safety System High Weight), the four analog inputs create four corresponding digital inputs for even higher analog weight values. If a higher condition is triggered, the lower condition may or may not have been triggered. Condition 4, Safety System Low Weight, can be triggered only if Conditions 1 through 3 are not triggered, so that it is mutually exclusive of the high weight alarms. In a hardware common cause failure analysis, the use of the same analog inputs for the different alarms would represent a common cause failure for these inputs. We can ignore this because we are assuming that the hardware will operate correctly, and so we will study only the software dependencies. While inputs 1 through 3 all initiate high weight alarms and can be related, they will be considered mutually exclusive inputs because of the definition used in the analysis. The frequencies and high weight conditions in Table 4.2 should be interpreted as follows:

1. Condition 1 (High Weight) will occur without escalating to higher weight conditions about 25 times a year.
2. Condition 2 (High-High Weight) will occur without escalating to Condition 3 about 17 times a year. This condition definition includes those times when (1) only Condition 2 occurs or (2) Condition 1 occurs and escalates no higher than Condition 2.
3. Condition 3 (Safety System High Weight) will occur about 16 times a year. This condition definition includes those times when (1) only condition 3 occurs or (2) Conditions 1 or 2 occur and escalate to Condition 3.

The Recirculating Cooling Water (RCW) / Freon Differential Pressure (DP) inputs can trigger Conditions 5 through 7. The single analog pressure input for Condition 5 is also used for Condition 7. Condition 5 is triggered for low analog pressure values. Condition 6, Safety System RCW/Freon DP, occurs when an even lower pressure value is detected by two digital inputs. Condition 7, RCW/Freon DP, occurs if the safety system (condition 6) does not trip and if an even lower pressure is input from Condition 6. Condition 6 and 7 pressure inputs are considered mutually exclusive; Condition 5 is not mutually exclusive. All three differential pressure inputs, however, will be considered mutually exclusive because of the definitions for the

condition events used in the analyses. The frequencies and differential pressure conditions in Table 4.2 should be interpreted as follows:

1. Condition 5 (RCW/Freon DP trip to hot standby) will occur without escalating to a more severe low temperature about 41 times a year.
2. Condition 6 (Safety System RCW/Freon DP) will occur without escalating to Condition 7 about 6 times a year. This condition definition includes those times when (1) only Condition 6 occurs or (2) Condition 5 occurs and escalates no higher than Condition 6.
3. Condition 7 (RCW/Freon DP trip to criticality mode) will occur about 6 times a year. This condition definition includes those times when (1) only Condition 7 occurs or (2) Conditions 5 or 6 occur and escalate to Condition 7.

Conditions 10 and 11 have two different analog temperature inputs but are related by their function in the process. The RCW heats and cools the freon. There is a possibility that one condition causes the other; but, other conditions may cause both to occur. It is not known for sure whether the occurrence of one guarantees the occurrence of the other, but they are closely related. Each of these inputs is expected to happen only once a year. Considering the large total number of annual inputs, the dependency of these two inputs should be negligible and will be considered mutually exclusive at this time. These two inputs, however, should be included in any process monitoring to verify the assumption.

The following trip conditions are not known to be related to each other or to any other trip conditions: Safety System Power Failure (Condition 8), UF6 High Pressure (Condition 9), and State error or mode failure (Condition 12).

Functions 1 and 2 both involve mode changes made by the operator. The same changes can be made from either the DPCS console (Function 1) or the F/S cabinet (Function 2). These two need separate entries because there are two separate software paths. Function 1 has the following digital operator inputs which are asserted via device or skid panel:

1. "CSB" changes the system to cold standby.
2. "HSB" changes the system to hot standby.
3. "FREEZE" initiates the freeze mode of the F/S.
4. "SUBLIME" initiates the sublime mode of the F/S.

Function 2 has all the inputs of Function 1 plus an additional input: "DPC" turns control over to the software. All Function 2 inputs are digital operator-initiated buttons.

Functions 3 and 4 are the Software and Hardware Trip Resets, respectively. These two functions are related in the same manner as the corresponding trip conditions discussed previously. These resets are completed only if the alarms have been triggered and must be resolved and reset before the system is allowed to operate normally. For this reason, the alarm trigger and the trip reset will be considered one event, and the inputs will be the condition trip inputs paired with the operator-initiated reset. It is assumed that the operator will always reset any alarm because the system is not allowed to operate normally until the alarm has been reset.

Functions 5 (Set F/S to Available), 6 (Set F/S to Unavailable), and 7 (Set F/S to Maintenance Mode) form a mutually exclusive group defining the availability status of the F/S.

Functions 8 and 9, the Weight Test and its cancellation, are not related to the weight conditions in Trip Conditions 1 through 4. Function 9 occurs only if there is a weight test. Each year there must be 4 weight tests per F/S system. The weight test cannot occur with any other input; therefore, if a weight test is underway and an alarm occurs, the weight test must be cancelled. In addition, if a weight test is in progress during a switchover, the weight test is cancelled after the switchover occurs. It is assumed that as many as 4 weight tests may be interrupted in this way so that the total number of weight tests is the required 4 plus an estimated 4 interruptions for a total of 8 tests. Because the tests and cancellations are directly dependent, the partial weight test and cancellation (Function 9) will be considered one input with an annual frequency of 4, and the required, complete weight test (Function 8) will be a mutually exclusive test with an annual frequency of 4. Table 4.4.b exhibits these results.

These additional assumptions are made for the inputs:

1. In those cases where multiple analog inputs may initiate the same alarm, it is assumed that each of the inputs is equally likely.
2. In those cases where the input is analog and a single threshold value of the analog input will trigger the trip condition, such analog inputs will be treated as digital inputs. This excludes Trip Conditions 1, 2, 5, and 7.
3. The availability functions (5, 6, and 7), the alarm conditions/reset functions (Table 4.2 and Functions 3 and 4 from Table 4.3), and the cancellation of the weight test (Function 9) are all mutually exclusive.

The final estimated input space appears in Tables 4.4.a and 4.4.b.

**Table 4.4.a. Input Space for the Freezer/Sublimer Switchover Capability**

<b>Trip Condition and Reset</b>	<b>Frequency (per F/S per Yr)</b>	<b>Input Space Probability</b>
1. High Weight (trip to cold standby) and trip reset <sup>a</sup>	25	.0218
2. High-High Weight (trip to moderate hot standby) and trip reset <sup>a</sup>	17	.0148
3. Safety System High Weight and trip reset <sup>b</sup>	16	.0139
4. Safety System Low Weight (trip to hot standby) and trip reset <sup>b</sup>	18	.0157
5. RCW/Freon Differential Pressure (DP) (trip to hot standby) and trip reset <sup>a</sup>	41	.0357
6. Safety System RCW/Freon DP (trip to criticality mode) and trip reset <sup>b</sup>	6	.0052
7. RCW/Freon DP (trip to criticality mode) and trip reset <sup>a</sup>	6	.0052
8. Safety System Power Failure (trip to power fail mode) and trip reset <sup>b</sup>	5	.0044
9. UF6 High Pressure (trip to cold standby with the vent valve open) and trip reset <sup>a</sup>	1	.0008
10. Low Freon Temperature (trip to hot standby) and trip reset <sup>a</sup>	1	.0009
11. Low Freon RCW Temperature (trip to hot standby) and trip reset <sup>a</sup>	1	.0009
12. State error or mode failure (trip to hot standby) and trip reset <sup>b</sup>	1	.0009

- <sup>a</sup> Software trip
- <sup>b</sup> Hardware trip

**Table 4.4.b. Input Space for the Freezer/Sublimer Switchover Capability**

<b>Function</b>	<b>Frequency (per F/S per yr)</b>	<b>Input Space Probability</b>
1. Operator changes the mode from the DPCS console	730	.6359
2. Operator changes the mode from the F/S cabinet	24	.0209
5. Set the Freezer/Sublimer (F/S) to Available	200	.1742
6. Set the F/S to Unavailable	24	.0209
7. Set the F/S to Maintenance Mode	24	.0209
8. Weight Test	4	.0034
9. Cancel Weight Test	4	.0034
<b>TOTAL</b>	1148	1.00

#### 4.3. Switchover Software Testing Summary

The requirements for the software specifications and testing are contained in document SD-KIE-17204-FSD-A. All testing and debugging were completed on a single CPU until no failures occurred. Then, the software testing in switchover mode was started. This section summarizes the testing for the switchover capability only. Tables 4.5 and 4.6 show the F/S software code segments that are executed for each of the inputs. In the tables, "DALMXX 1" means "segment 1 of the DALMXX program." Table 4.7 lists the following information for each program and segment: the number of lines of code in the segment ("code count"), estimated execution time in milliseconds (based on the number of lines multiplied by the CPU speed), an asterisk if the segment is critical with respect to safety, the number of tests performed on that segment, and any additional comments.

Table 4.5. Code Segments for Condition Inputs

Trip Condition and Reset	Code Segments
1. High Weight (trip to cold standby) <sup>a</sup>	DALMXX 1; DALMXX 10; MDTXXX 1, 2 (no reset)
2. High-High Weight (trip to moderate hot standby) <sup>a</sup>	DALMXX 1; DALMXX 5; MDTXXX 1, 2 (no reset)
3. Safety System High Weight and trip reset <sup>b</sup>	HALMXX 1; HALMXX 4; MDTXXX 1, 2 (reset: HALMXX 1, HALMXX 8; MDTXX 1, 2)
4. Safety System Low Weight (trip to hot standby) and trip reset <sup>b</sup>	HALMXX 1; HALMXX 5; MDTXXX 1, 2 (reset: HALMXX 1; HALMXX 9; MDTXX 1, 2)
5. RCW/Freon Differential Pressure (DP) (trip to hot standby) and trip reset <sup>a</sup>	DALMXX 1; DALMXX 7; MDTXXX 1, 2 (reset: STSXXX 1, 2, 21, 27)
6. Safety System RCW/Freon DP (trip to criticality mode) and trip reset <sup>b</sup>	HALMXX 1; HALMXX 2; MDTXXX 1, 2 (reset: HALMXX 1; HALMXX 7; MDTXX 1, 2)
7. RCW/Freon DP (trip to criticality mode) and trip reset <sup>a</sup>	DALMXX 1; DALMXX 3; MDTXXX 1, 2 (reset: STSXXX 1, 2, 21, 23)
8. Safety System Power Failure (trip to power fail mode) and trip reset <sup>b</sup>	HALMXX 1; HALMXX 2; MDTXXX 1, 2 (reset: HALMXX 1; HALMXX 6; MDTXX 1, 2)
9. UF6 High Pressure (trip to cold standby with the vent valve open) and trip reset <sup>a</sup>	DALMXX 1; DALMXX 6; MDTXXX 1, 2 (reset: STSXXX 1, 2, 21, 25)
10. Low Freon Temperature (trip to hot standby) and trip reset <sup>a</sup>	DALMXX 1; DALMXX 8; MDTXXX 1, 2 (reset: STSXXX 1, 2, 21, 28)
11. Low RCW Temperature (trip to hot standby) and trip reset <sup>a</sup>	DALMXX 1; DALMXX 9; MDTXXX 1, 2 (reset: STSXXX 1, 2, 21, 29)
12. State error or mode failure (trip to hot standby) and trip reset <sup>b</sup>	DALMXX 1; DALMXX 4; MDTXXX 1, 2 (reset: STSXXX 1, 2, 21, 24)

- Software trip
- Hardware trip

**Table 4.6. Code Segments for Function Inputs**

Function	Code Segments
1. Operator changes the mode from the DPCS console	OCM30 1, 2; IMTXXX 1, 2, 3; MDTXXX 1, 2
2. Operator changes the mode from the F/S cabinet	IMTXXX 1, 2, 3; LMODXX 1; LMODXX 2 or 21 (if DPC button was pushed then a mode button pushed); 3, 31, or 32 (if mode button was pushed then DPC button pushed); 4 or 41 (if mode button was pushed then mode button pushed); 5 or 51 (if mode button bypass existed then DPC button pushed); (to assert all pre-existing alarm conditions: MDTXXX 1, 2; DALMXX 3, 4, 5, 6, 10)
5. Set the Freezer/Sublimator (F/S) to Available	STSXXX 1, 3; (if in maintenance mode, also IMTXXX 1, 2, 3); MDTXXX 1, 2
6. Set the F/S to Unavailable	STSXXX 1, 4; (if in maintenance mode, also IMTXXX 1, 2, 3); MDTXXX 1, 2
7. Set the F/S to Maintenance Mode	STSXXX 5, 51;
8. Weight Test	WTTXX; DALMXX 1, 5, 10; HALMXX 1, 4, 5;
9. Weight Test Cancellation	STSXXX 1;

Table 4.7 Redundant Segments Testing Summary

Program Name	Segment Number	Code Count	Exec. time in millisecc	Critical	Test Performed	Comments	
OCM30 (1.0)	1	3	1.7		Many	Operator queries	
	2	31	17.1		9		
IMTXXX (2.0)	1	188					
	2	27	14.9				
	3	26	14.3				
MDTXXX (3.0)	1	2			110	Wait until	
	2	65		*	42	Lots of "Wait 1 second" and Indeterminate waits	
ASMXXX (4.0)	none			*	many		
STSXXX (5.0)	1	3			137	Wait until	
	2	51	28.1		8		
	21	25	13.8	*			
	22	43	23.7	*		Segment numbers 22 through 29 are very similar	
	23	43	23.7	*			
	24	43	23.7	*			
	25	43	23.7	*			
	26	43	23.7	*			
	27	43	23.7	*	2		
	28	43	23.7	*			
	29	43	23.7	*			
		3	54	5029.7		1	Wait 5 seconds
		4	39	5021.5			Wait 5 seconds
		5	72			4	Operator Queries
		51	25	13.8			

WTTXXX (6.0)	none				29	Weight Test is canceled on failover (29 weight tests)
HALMXX (7.0)	1	14			143	Repeated continuously
	2	39	21.5	*	1	
	3	40	22.0	*		
	4	46	25.3	*	1	
	5	41	22.6	*		
	6	93	51.2	*	2	
	7	93	51.2	*	2	
	8	93	51.2	*	1	
	9	93	51.2	*	2	
LMDXX (8.0)	1	6			142	Repeated continuously
	2	5	2.8			
	21	43	23.7		1	
	3	5	2.8			
	31	12	6.6			
	32	6	6.6			
	4	6	3.3		4	
	41	8	4.4		1	
	5	5	2.8			
	51	13	7.2			
DALMXX (9.0)	1	32			111	Repeated Continuously
	2				32	This segment was deleted after the first 46 tests.
	3	52	28.6	*		
	4	53	29.2	*	1	
	5	57	31.4	*		
	6	55	30.3	*		
	7	53	29.2	*	3	

	8	54	29.7	*	3	
	9	54	29.7	*	3	
	10	54	29.7	*		
STRXXX (10.0)	none					This program starts the software and will be restarted
SCHKXX (11.0)	none			*	many	Restarts scanning from the top on failover.

#### 4.4 Testing Coverage Estimate

Tables 4.8.a and 4.8.b provide the number of tests for each input and include the percentage testing for each input. The test log and some summary tables appear in Appendix B. It is important to note that while some tests involved activating a new input in the presence of other potential inputs, each test is counted only for the new input. For example, if Conditions 11 and 10 were present when Condition 2 was activated in switchover mode, the test counted only for Condition 2. It is physically possible to handle multiple inputs because the F/S system is a timeshared system that processes concurrent programs.

As discussed in Section 3.6, the coverage estimate is the sum of the input probabilities for each input tested. In Table 4.8, each tested input is marked with an asterisk in the "Input Space Probability" column. The coverage estimate is the sum of all the probabilities which are marked with an asterisk. This shows that the coverage estimate for this set of tests is .9948 or 99.48 %, that is, almost all the inputs that are expected to occur have been tested successfully. Only two inputs, Condition 7 (RCW/Freon DP trip to criticality mode) and Function 9 (Cancel Weight Test) were not tested.

Table 4.8 shows that using the input space probabilities to design the testing plan would have reduced a number of tests. The "Input Space Probability" column is multiplied by 100 to provide the ideal percentage total testing for each input. Less testing could have been done for several inputs, especially the weight test and its cancellation. The weight test, however, is different in that it is a long section of code that could have been interrupted in several places during switchover.

**Table 4.8.a. Coverage Summary for the Freezer/Sublimer Switchover Capability**

<b>Trip Condition and Reset</b>	<b>Number of Tests</b>	<b>Percent of Total Testing</b>	<b>Input Space Probability</b>
1. High Weight (trip to cold standby) and trip reset <sup>a</sup>	1	.66%	.0218*
2. High-High Weight (trip to moderate hot standby) and trip reset <sup>a</sup>	7	4.61%	.0148*
3. Safety System High Weight and trip reset <sup>b</sup>	9	5.92%	.0139*
4. Safety System Low Weight (trip to hot standby) and trip reset <sup>b</sup>	14	9.21%	.0157*
5. RCW/Freon Differential Pressure (DP) (trip to hot standby) and trip reset <sup>a</sup>	16	10.53%	.0357*
6. Safety System RCW/Freon DP (trip to criticality mode) and trip reset <sup>b</sup>	14	9.21%	.0052*
7. RCW/Freon DP (trip to criticality mode) and trip reset <sup>a</sup>	0	0%	.0052
8. Safety System Power Failure (trip to power fail mode) and trip reset <sup>b</sup>	15	9.87%	.0044*
9. UF6 High Pressure (trip to cold standby with the vent valve open) and trip reset <sup>a</sup>	1	.66%	.0009*
10. Low Freon Temperature (trip to hot standby) and trip reset <sup>a</sup>	3	1.97%	.0009*
11. Low Freon RCW Temperature (trip to hot standby) and trip reset <sup>a</sup>	6	3.95%	.0009*
12. State error or mode failure (trip to hot standby) and trip reset <sup>b</sup>	4	2.63%	.0009*

- \* Software trip.
- <sup>b</sup> Hardware trip.
- Include probability in input space coverage estimate.

**Table 4.8.b. Coverage Summary for the Freezer/Sublimer Switchover Capability**

<b>Function</b>	<b>Number of Tests</b>	<b>Percentage of Total Testing</b>	<b>Input Space Probability</b>
1. Operator changes the mode from the DPCS console	8	5.26%	.6359*
2. Operator changes the mode from the F/S cabinet	8	5.26%	.0209*
5. Set the Freezer/Sublimer (F/S) to Available	7	4.61%	.1742*
6. Set the F/S to Unavailable	4	2.63%	.0209*
7. Set the F/S to Maintenance Mode	6	3.95%	.0209*
8. Weight Test	29	19.08%	.0034*
9. Weight Test Cancellation	0	0%	.0034
<b>TOTAL</b>	152	100	.994

\*Include probability in input space coverage estimate.

## 5. Summary and Conclusions

K-25 Engineering Division purchased a series of redundant computer systems and developed software for the purpose of providing continuous process monitoring and control for the Freezer/Sublimator equipment in the gaseous diffusion process at the Paducah Gaseous Diffusion Plant. The application software is loaded on two central processing units (CPU) so that in the event of a failure of the primary unit, the processing can switch to the backup unit and continue processing without error. It is the purpose of this document to demonstrate the reliability of this system with respect to its ability to switch between redundant CPU. The total reliability estimation problem - which considers the hardware, operating system software, and application software - has been reduced to one that considers only the application software directly involved in the switchover process.

The switchover software reliability estimate  $R$  is .9934 with a 95% Bayesian probability interval of  $.9754 < R < .9998$ . This is a high reliability estimate, as it needs to be because the software reacts to alarm conditions. The estimate was calculated from 152 tests with a single failure occurring at test 43. After the failure, the software was corrected, and the remaining 109 tests were completed without failures. The coverage of the reliability estimate is 99.48%. Therefore, the estimate of the probability that the software will continue to work is .9934 as determined by testing 99.48% of the inputs.

It is important to understand that the reliability estimate can be quoted only for the existing software. If changes are made in the software, then new reliability and coverage estimates must be made. It will also be important to monitor the F/S process to validate whether or not the process has the expected number of alarms. This information will be helpful for any new software or changes in the existing software which may be made in the future.

## 6. References

- Berger, James O., (1985), *Statistical Decision Theory and Bayesian Analysis*, Second Edition, Springer-Verlag, New York, New York.
- Davis, J. N., and Devan, W. R., (1990), "F/S Functional System Design for Process Inventory Control System Distributed Process Control," K-25 Site, Report SD-KIE-17204-FSD-A.
- Harris, Bernard, *Theory of Probability*, Addison-Wesley, Reading, Massachusetts, pp. 6-20.
- Henley, Ernest J., and Kumamoto, Hiromitsu, (1981), *Reliability Engineering and Risk Assessment*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, pp. 254-256.
- Keceglioulu, Dimitri, (1991), *Reliability Engineering Handbook*, Volumes 1 and 2, Prentice Hall, Englewood Cliffs, New Jersey, Volume 2 pp. 401-465.
- Martz, Harry, F., and Waller, Ray A., (1991), *Bayesian Reliability Analysis*, Krieger Publishing Company, Malabar, Florida.
- Mendenhall and Schaeffer (1973), *Mathematical Statistics with Applications*, Wadsworth Publishing Company, Inc., Belmont, California, pp. 315-317.
- Musa, John D., (1975), "A Theory of Software Reliability and its Application," *IEEE Transactions on Software Engineering*, SE-1(3), pp. 312-327.
- Musa, John D., (1985), "Software Engineering: The Future of a Profession," *IEEE Software*, 2(1), pp. 55-62.
- Musa, John D., and Okumoto, K., (1984a), "A Comparison of Time Domains for Software Reliability Models," *Journal of Systems and Software*, 4(4), pp. 277-287.
- Musa, John D., Iannino, Anthony, and Okumoto, Kazuhira, (1990), *Software Reliability Measurement, Prediction, Application Professional Edition*, McGraw-Hill Publishing Company, New York, New York.
- Musa, John D., Iannino, Anthony, and Okumoto, Kazuhira, (1987), *Software Reliability Measurement, Prediction, Application*, McGraw-Hill Book Company, New York, New York.
- Musa, John D., (1979d), "Validity of Execution Time Theory of Software Reliability," *IEEE Transactions on Reliability*, R-28(3), pp. 181-191.
- Nelson, Wayne, (1991), *Accelerated Testing*, John Wiley and Sons, New York, New York.
- Walraven, Ing. W. M., (1986), "Emcon D/3 Integrated Process Control System," Organization for Applied Scientific Research (TNO-IWECO), Evaluation Report E2500 T86.

## **APPENDIX A**

### APPENDIX A Introduction to one Musa Model

The software reliability models developed by Musa, Ianino, and Okumoto are documented in their books (1987 and 1990) and numerous journals, such as *IEEE Transactions on Software Engineering*, *IEEE Transactions on Reliability*, *Journal of Systems and Software*, and *IEEE Software*. The models discussed by Musa et al (1990) develop estimates of the reliability at a point in time. This reliability represents the accumulated reliability resulting from testing, correction of failures, and continued testing and correction. In this manner, the growth or demise of reliability can be tracked throughout the testing and development process. This capability can help determine expected completion dates, additional manpower needs, and other important concerns of software development.

Accurate software reliability estimation requires up front planning in order to be able to record all the data required and monitor progress. The models require failure times or failures per time interval (grouped data). Additionally, the Musa models require an estimated number of faults in the software ( $\omega_o$ ) and an estimated initial failure intensity ( $\lambda_o$ ). Musa et al (1987 and 1990) discuss both of these as follows.

A fault is defined as an error in the software that may cause multiple failures, ie, multiple inputs may fail in the same or different ways when encountering the software fault. Before testing starts, an initial estimate is needed for  $\omega_o$ , the total number of faults (inherent faults) in the software.

The estimated initial failure intensity  $\lambda_o$  is defined as

$$\lambda_o = f \times K \times \omega_o,$$

where,

$f$  = linear execution frequency,  
 $K$  = fault exposure ratio, and  
 $\omega_o$  = inherent faults.

Further,  $f$  is the number of times the program would be executed per unit time without branches or loops and is estimated by

$$f = r/I \quad ,$$

where,

$r$  = instruction execution rate, and  
 $I$  = executable object instructions.

The number of inherent faults is

$$\omega_o = \omega_i \times I,$$

where,

$\omega_i$  = inherent faults per developed source instructions (or fault density), and  
 $I$  = developed executable source instructions.

For details on values of  $K$ , see Musa (1979).

Musa et al (1990) discuss and provide applications for two types of models, the basic execution time model and the logarithmic Poisson model. Briefly, the basic execution time model is used for a uniform operating profile and has

$$\lambda(\mu) = \lambda_o \left[1 - \frac{\mu}{V_o}\right]$$

where,

$\mu$  = average number of failures per CPU hour expected at a given time, and  
 $V_o$  = total number of failures in infinite time.

The logarithmic Poisson model is used for highly non-uniform profiles and has

$$\lambda(\mu) = \lambda_o \exp(-\theta \mu),$$

where,

$\theta$  = failure intensity decay parameter.

The additional parameter  $\theta$  is found by plotting two observed values: the natural log of the failure intensity against the mean failures experienced. The slope of this line is the parameter  $\theta$ .

Many more details and applications are found in Musa et al (1987, 1991). While these models require detailed information, they can provide valuable information when applied properly.

## **APPENDIX B**

**APPENDIX B Redundancy Testing Log and Summary Tables**

**Table B.1. Condition Codes**

Code Condition	
1.	High Weight (trip to cold standby) and trip reset <sup>a</sup>
2.	High-High Weight (trip to moderate hot standby) and trip reset <sup>a</sup>
3.	Safety System High Weight and trip reset <sup>b</sup>
4.	Safety System Low Weight (trip to hot standby) and trip reset <sup>b</sup>
5.	RCW/Freon Differential Pressure (DP) (trip to hot standby) and trip reset <sup>a</sup>
6.	Safety System RCW/Freon DP (trip to criticality mode) and trip reset <sup>b</sup>
7.	RCW/Freon DP (trip to criticality mode) and trip reset <sup>a</sup>
8.	Safety System Power Failure (trip to power fail mode) and trip reset <sup>b</sup>
9.	UF6 High Pressure (trip to cold standby with the vent valve open) and trip reset <sup>a</sup>
10.	Low Freon Temperature (trip to hot standby) and trip reset <sup>a</sup>
11.	Low RCW Temperature (trip to hot standby) and trip reset <sup>a</sup>
12.	State error or mode failure (trip to hot standby) and trip reset <sup>b</sup>

<sup>a</sup>Software trip

<sup>b</sup>Hardware trip

**Table B.2 Function Codes**

<b>Code Function</b>
1. Operator changes the mode from the DPCS console
2. Operator changes the mode from the F/S cabinet
5. Set the Freezer/Sublimator (F/S) to Available
6. Set the F/S to Unavailable
7. Set the F/S to Maintenance Mode
8. Weight Test
9. Weight Test Cancellation
<b>TOTAL FREQUENCY</b>

**Note for Table B.3.a Redundancy Testing Log:**

All tests for switchover mode are recorded in this table, including test 24 which had no alarm conditions present and is not included in the count for the number of successes in the failed group or the total number of tests. For this reason, the failed test is number 44 in this table, and the total number of test entries is 153.

**Table B.3.a. Redundancy Testing Log**

<b>Function or Condition Tested</b>	<b>HALM 7.0 Program</b>	<b>DALM 9.0 Program</b>	<b>MDT 3.0 Program</b>	<b>STS 5.0 Program</b>	<b>LMOD 8.0 Program</b>	<b>Date Tested</b>
1. Cond. 4	1	2	2	1	1	1/22/92
2. Cond. 4	1	2	2	1	1	1/31/92
3. Reset	1	1	2	1	1	
4. Cond. 4	1	2	2	1	1	
5. Reset	1	2	2	1	1	
6. Cond. 8	1	2	1	1	1	
7. Reset	1	1	1	1	1	
8. Cond. 8	1	2	1	1	1	
9. Reset	1	2	2	1	1	
10. Cond. 8	1	2	1	1	1	
11. Reset	1	2	1	1	1	
12. Cond. 3	1	2	2	1	1	2/3/92
13. Cond. 3	4	2	1	1	1	
14. Reset	1	2	1	1	1	
15. Cond. 3	1	2	2	1	1	
16. Reset	1	2	1	1	1	
17. Cond. 6	1	4	2	1	1	
18. Reset	1	1	1	1	1	
19. Cond. 6	1	1	1	1	1	
20. Reset	1	2	1	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
21. Cond. 6	1	2	2	1	1	
22. Reset	1	2	1	1	1	
23. Cond. 5	1	2	2	1	1	
24. No alarm conditions present	1	2	1	1	1	
25. Reset Cond. 5	1	2	1	1	1	
26. Cond. 5	1	2	2	1	1	
27. Reset	1	2	2	1	1	
28. Cond. 5	1	7	2	1	1	
29. Reset	1	2	1	1	1	
30. Reset	1	2	1	2	1	
31. Cond. 5	1	2	2	1	1	
32. Reset	1	1	1	2	1	
33. Cond. 1	1	1	1	1	1	
34. Cond. 2	1	2	1	1	1	
35. Reset	1	2	2	1	1	
36. Cond. 2	1	2	1	1	1	
37. Reset	1	2	1	2	1	
38. Cond. 2	1	2	1	1	1	
39. Reset	1	2	1	1	1	
40. Cond. 11	1	1	2	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
41. Reset	1	1	1	1	1	
42. Reset	1	1	1	1	1	
43. Reset Cond. 5 with Conds. 11 & 10	1	1	1	1	1	
44. Reset Cond. 5 with Conds. 11 & 10	1	2	1	27		FAILURE: Did not assert Conds. 10 or 11
45. Reset Cond. 5 with Conds. 10 & 11	1	1	1	1	1	2/4/92
46. Reset Cond. 5 with Conds. 10 & 11	1	1	1	27	1	
47. Reset Cond. 5 with Conds. 10 & 11	1	8	2	1	1	
48. Reset Cond. 10 with Cond. 11	1	1	1	2	1	
49. Reset Cond. 12 with Cond. 11	1	1	1	2	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
50. Reset Cond. 12 with Cond. 11	1	1	2	1	1	
51. Reset Cond. 12 with Cond. 11	1	1	1	1	1	
52. Reset Cond. 11 with Cond. 1	1	1	1	2	1	
53. Reset Cond. 11 with Cond. 1	1	1	1	2	1	2/5/92
54. Reset Cond. 5 with Conds. 11 & 1	1	9	1	1	1	
55. Reset Cond. 8 with Conds. 11 & 1	1	1	1	1	1	
56. Cond. 8 with Conds. 2, 11, & 10	1	1	1	1	1	
57. Reset	1	1	1	1	1	
58. Reset Cond. 2 with Conds. 11 & 10	1	1	1	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
59. Reset Cond. 5 with Conds. 11 & 10	1	8	2	1	1	
60. Reset Cond. 10 with Cond. 11	1	1	1	2	1	
61. Cond. 10 with Cond. 11	1	1	2	1	1	
62. Reset Cond. 12 with Conds. 10 & 11	1	1	1	1	1	
63. Cond. 6 with Conds. 10 & 11	1	1	1	1	1	
64. Reset	1	8	2	1	1	
65. Reset Cond. 6 with Cond. 11	7	9	1	1	1	
66. Reset Cond. 8 with Cond. 6	1	1	1	1	1	
67. Reset Cond. 8 with Conds. 6 & 4	1	1	1	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
68. Reset Cond. 8 with Conds. 6 & 4	6	1	1	1	1	
69. Reset Cond. 6 with Cond. 4	1	1	1	1	1	
70. Reset Cond. 3 with Cond. 4	1	1	1	1	1	
71. Reset Cond. 3 with Cond. 4	8	1	1	1	1	
72. Reset Cond. 11 with Funct. 2 (CSB)	1	1	1	1	1	
73. Funct. 2 (DPC) with Funct. 2 (CSB)	1	1	1	1	32	
74. Funct. 2 (HSB)	1	1	1	1	21	
75. Funct. 2 (DPC)	1	1	1	1	32	
76. Funct. 2 (CSB)	1	1	2	1	4	
77. Funct. 2 (HSB)	1	1	2	1	4	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
78. Funct. 2 (DPC)	1	1	1	1	32	
79. Cond. 6 with Funct. 2 (CSB)	1	1	1	1	4	
80. Reset	7	1	1	1	1	
81. Cond. 4 with Funct. 2 (CSB)	1	1	1	1	1	
82. Reset	1	1	1	1	41	
83. Cond. 8 with Funct. 2 (CSB)	1	1	1	1	4	
84. Reset Cond. 4 with Funct. 2 (CSB) & Conds. 5, 11, & 10	9	1	1	1	1	2/6/92
85. Cond. 4 with Funct. 2 (CSB) & Conds. 5, 11, & 10	1	1	2	1	1	
86. Reset	1	7	2	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
87. Cond. 6 with Funct. 2 (CSB) & Conds. 5, 11, & 10	1	1	1	1	1	
88. Reset	1	1	1	1	1	
89. Cond. 8 with Funct. 2 (CSB) & Conds. 5, 11, & 10	2	1	1	1	1	
90. Reset	6	7	1	1	1	
91. Cond. 3 with Funct. 2 (CSB) & Conds. 5, 11, & 10	1	1	2	1	1	
92. Reset	1	1	2	1	1	
93. Funct. 2 (DPC) with Conds. 5, 11, & 10	1	1	1	1	1	
94. Reset Cond. 4 with Funct. 2 (CSB) & Conds. 9 & 5	1	1	1	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
95. Reset Cond. 4 with Funct. 2 (CSB) & Conds. 9 & 5	9	1	1	1	1	
96. Reset Cond. 4 with Funct. 2 (DPC) & Cond. 9	1	1	1	1	1	
97. Reset Cond. 4 with Funct. 2 (DPC) & Cond. 9	1	1	1	1	1	
98. Funct. 2 (DPC) with Cond. 9	1	1	2	1	32	
99. Reset Cond. 9	1	1	1	1	1	
100. Funct. 8 before first alarm reached	1	1	1	1	1	2/18/92
101. Funct. 8 before first alarm reached	1	1	1	1	1	
102. Funct. 8 after first trip reached	1	1	2	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
103. Funct. 8 after first trip reached	1	1	1	1	1	
104. Funct. 8 after first trip & reset when waiting for operator response	1	1	1	1	1	
105. Same	1	1	1	1	1	
106. Funct. 8 just prior to first trip	1	1	1	1	1	
107. Funct. 8 just prior to second trip	1	1	1	1	1	
108. Same	1	1	1	1	1	
109. Funct. 8 just after second trip	1	1	1	1	1	
110. Funct. 8 just prior to second trip	1	1	1	1	1	
111. Funct.8 just after operator response to continue message	1	1	1	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
112. Funct.8 before first trip	1	1	1	1	1	
113. Same	1	1	1	1	1	
114. Same	1	1	1	1	1	
115. Same	1	1	1	1	1	
116. Funct. 8 just after first trip	1	1	2	1	1	
117. Same	1	1	2	1	1	
118. Funct. 8 while waiting for operator response to continue message	1	1	1	1	1	
119. Funct. 8 just after operator response to continue message	1	1	1	1	1	
120. Funct. 8 while waiting for operator to respond to reset message	1	1	1	1	1	

Function or Condition Tested	HALM 7.0 Program	DALM 9.0 Program	MDT 3.0 Program	STS 5.0 Program	LMOD 8.0 Program	Date Tested
121. Funct. 8 while resetting first trip	1	1	2	1	1	
122. Funct. 8 while awaiting operator response to disable 1A message	1	1	1	1	1	
123. Funct. 8 just prior to second trip	1	1	1	1	1	
124. Funct. 8 just after second trip	1	1	2	1	1	
125. Same	1	1	2	1	1	
126. Funct. 8 before first trip (PB F/S test)	1	1	1	1	1	
127. Funct. 8 before third trip (PB F/S test)	1	1	1	1	1	
128. Funct. 8 before fourth trip (PB F/S test)	1	1	1	1	1	



146. Funct. 1 (device DPC FREEZE)	1	1	2	1	1	1	-	
147. Funct. 1 (device CSB)	1	1	2	1	1	1	-	
148. Funct. 1 (device SUBLIME)	1	1	2	1	1	1	-	
149. Funct. 1 (device HSB)	1	1	2	1	1	1	-	
150. Funct. 1 (skid panel FREEZE)	1	1	1	1	1	1	2	
151. Funct. 1 (skid panel CSB)	1	1	2	1	1	1	2	
152. Funct. 1 (skid panel SUBLIME)	1	1	2	1	1	1	2	
153. Funct. 1 (skid panel HSB)	1	1	2	1	1	1	2	

**Table B.4. Summaries of Tests for Coverage Determination**

Conditions/Functions	Existing Conditions or Functions	Number of Tests Initiating Input	Number of Tests Resetting Input	Total Tests for the Input
Cond. 1	none	1	0	1
Cond. 2	none	3	3	
	Conds. 11 & 10	0	1	7
Cond. 3	none	3	2	
	Cond. 4	0	2	
	Funct. 2 (CSB) & Conds. 5, 10, & 11	1	1	9
Cond. 4	none	3	2	
	Funct. 2 (CSB)	1	1	
	Funct. 2 (CSB) & Conds. 5, 10, & 11	1	2	
	Funct. 2 (CSB) & Conds. 9 & 5	1	1	
	Funct. 2 (CSB) & Cond. 9	0	2	14
Cond. 5	none	4	5	
	Conds. 10 & 11	0	6	
	Conds. 11 & 1	0	1	16
Cond. 6	none	3	3	
	Conds. 11 & 10	1	1	
	Cond. 11	0	1	
	Cond. 4	0	1	

Conditions/Functions	Existing Conditions or Functions	Number of Tests Initiating Input	Number of Tests Resetting Input	Total Tests for the Input
	Funct. 2 (CSB)	1	1	
	Funct. 2 (CSB) & Conds. 5, 10, & 11	1	1	14
Cond. 7				0
Cond. 8	none	3	3	
	Conds. 11 & 1	0	1	
	Conds. 2, 11, & 10	1	1	
	Cond. 6	0	1	
	Conds. 6 & 4	0	2	
	Funct. 2 (CSB)	1	0	
	Funct. 2 (CSB) & Conds. 5, 10, & 11	1	1	15
Cond. 9	none	0	1	1
Cond. 10	Cond. 11	1	2	3
Cond. 11	none	1	2	
	Cond. 1	0	2	
	Funct. 2 (CSB)	0	1	3
Cond. 12	Cond. 11	0	3	
	Conds. 10 & 11	0	1	4
Funct. 1 (device DPC FREEZE)	none	1	N/A	
Funct. 1 (skid panel DPC FREEZE)	none	1	N/A	
Funct. 1 (device CSB)	none	1	N/A	

Conditions/Functions	Existing Conditions or Functions	Number of Tests Initiating Input	Number of Tests Resetting Input	Total Tests for the Input
Funct. 1 (skid panel CSB)	none	1	N/A	
Funct. 1 (device SUBLIME)	none	1	N/A	
Funct. 1 (skid panel SUBLIME)	none	1	N/A	
Funct. 1 (device HSB)	none	1	N/A	
Funct. 1 (skid panel HSB)	none	1	N/A	8
Funct. 2 (DPC)	none	2	N/A	
	Funct. 2 (CSB)	1	N/A	
	Conds. 5, 10, & 11	1	N/A	
	Funct. 2 (CSB) & Cond. 9	1	N/A	
(CSB)	none	1	N/A	
(HSB)	none	2	N/A	
(FREEZE)		0	N/A	
(SUBLIME)		0	N/A	8
Funct. 5	none	7	N/A	7
Funct. 6	none	4	N/A	4
Funct. 7	none	6	N/A	6
Funct. 8	none	29	N/A	29
<b>Total Tests</b>				<b>152</b>

**INTERNAL DISTRIBUTION**

- |       |                            |        |                                 |
|-------|----------------------------|--------|---------------------------------|
| 1.    | B. R. Appleton             | 17-21. | S. Raby                         |
| 2.    | T. Darland                 | 22-26. | R. F. Sincovec                  |
|       | Mathematical Sciences Lib. | 27.    | J. N. Sumner                    |
| 3-7.  | J. N. Davis                | 28-32. | R. C. Ward                      |
| 8.    | D. J. Downing              | 33.    | Central Research Library        |
| 9-13. | D. M. Flanagan             | 34.    | K-25 Applied Technology Library |
| 14.   | A. Geist                   | 35.    | ORNL Patent Office              |
| 15.   | L. J. Gray                 | 36.    | Y-12 Technical Library          |
| 16.   | C. E. Oliver               | 37-41. | Laboratory Records Department   |
|       |                            | 42.    | Laboratory Records Depart. - RC |

**EXTERNAL DISTRIBUTION**

43. Professor Roger W. Brockett (EPMD Advisory Committee), Wang Professor of Electrical Engineering and Computer Science, Division of Applied Sciences, Harvard University, Cambridge, MA 02138.
44. Siddhartha Chatterjee, RIACS, Mail Stop T045-1, NASA Ames Research Center, Moffett Field, California 94035-1000.
45. Dr. Donald J. Dudziak (EPMD Advisory Committee)  
Department of Nuclear Engineering  
110B Burlington Engineering Labs  
North Carolina State University  
Raleigh, NC 27695-7909
46. Dr. Jerome Friedman, Department of Statistics, Sequoia Hall, Stanford University, Stanford, CA 94305.
47. Dr. Dan Hitchcock, Office of Scientific Computing, ER-7, Applied Mathematical Sciences, Office of Energy Research, U.S. Department of Energy, Washington, DC 20585.
48. Dr. Fred Howes, Office of Scientific Computing, ER-7, Applied Mathematical Sciences, Office of Energy Research, U.S. Department of Energy, Washington, DC 20585.
49. Dr. James E. Leiss (EPMD Advisory Committee), Rt. 2, Box 142C, Broadway, VA 22815.

50. Professor Neville Moray (EPMD Advisory Committee), Department of Mechanical and Industrial Engineering, University of Illinois, 1206 West Green Street, Urbana, IL 61801.
51. Dr. Vijayan Nair, Statistics Research, AT&T Bell Labs, Murray Hill, New Jersey 07974.
52. Dr. David Nelson, Scientific Computing Staff, Applied Mathematical Sciences, Office of Energy Research, U.S. Department of Energy, Washington, D.C. 20585.
53. Dr. Jerome Sacks, NISS, P. O. Box 14162, Research Triangle Park, North Carolina, 27709-4162
54. Dr. L. R. Shenton, Office of Computing and Information Services, Boyd Graduate Studies Building, University of Georgia, Athens, Georgia 30602.
55. Dr. Daniel L. Solomon, Department of Statistics, North Carolina State University, P. O. Box 5457, Raleigh, North Carolina 27650.
56. Dr. Werner Stuetzle, Department of Statistics, GN-22, University of Washington, Seattle, Washington 98195.
57. Dr. Ray A. Waller, S-1, Statistics, Los Alamos National Laboratory, P. O. Box 1663, Los Alamos, NM 87545.
58. Professor Mary F. Wheeler (EPMD Advisory Committee), Rice University, Department of Mathematical Sciences, P.O. Box 1892, Houston, TX 77251.
59. Office of Assistant Manager for Energy Research and Development, U.S. Department of Energy, Oak Ridge Operations Office, P. O. Box 2001, Oak Ridge, Tennessee 37831-8600.
- 60-61. Office of Scientific and Technical Information, P. O. Box 62, Oak Ridge, Tennessee 37831-0062.