BIT Analysis Process

Systems Engineering Master's Project

Davinia B. Chism

Submitted to the College of Engineering
Texas Tech University in
Partial Fulfillment
Of the Requirements for
The Degree of
MASTER OF ENGINEERING

October 12, 2002

Approved:

____
Dr. J. Borrelli

____
Dr. A. Ertas

____
Dr. T. Maxwell

____
Dr. M. Tanik

Abstract

Most products developed and marketed today need to be of high quality, easily maintainable and robust. As a result, product specifications have requirements for a system level built-in test (BIT). These requirements cover BIT detection percentages, false alarm rates, fault isolation percentages and other parameters to be measured at a system level.

The objective of this project is to develop a process to perform an analysis of the BIT on an existing system to determine the failure detection and fault isolation percentages and false alarm rates. An easy to follow process, adaptable to any system, and automated tools to simplify equations are a part of this process.

As an example and the test system for the process, this project includes a BIT analysis on the HTI Second Generation Forward Looking Infrared (FLIR) sensor. The HTI FLIR is an existing design that has BIT tests implemented and specific requirements for failure detection percentages and false alarm rates. An analysis was performed early in the program on a previous design and no re-analysis of the system has taken place. Prior to the conclusion of this project, the HTI program had no way of verifying these BIT requirements.

Acknowledgements

I would like to thank Dr. Ertas and Dr. Tanik for their enthusiasm and support in my goal of pursuing this degree. They both opened my eyes – not only to new topics and information – but also to the concept of pursuing topics and synthesizing the new from the traditional. I have admired their selfless devotion of time to us, the many respected educators and industry professionals they have shared with us, and their sincere interest in providing us the best education and skills to last a lifetime.

I would also like to acknowledge Greg Norby and Hector Reyes. They have supported me at Raytheon while I worked through this program and balanced the requirements of my job with school-work. They offered invaluable advice and I am grateful for the help and encouragement they gave me.

Thanks to my group members, Jean Cathcart, Kurt Himmelreich and Donna Maestas. We made a strong team and learned a great deal working together. Special thanks to Jean who helped me keep my sanity at work and in school.

My family deserves my undying thanks and love for supporting me and understanding the limitations on my time. Their belief in my goals and me means everything.

Disclaimer

This report is a generic approach to analyzing the Built-in-Test capabilities of an existing system. The opinions expressed in this paper are strictly those of the author and are not necessarily those of Raytheon, Texas Tech University, nor any U.S. Government agency.

Table of Contents

CHAPTER 1 INTRODUCTION **1**

1.1 BIT REQUIREMENTS FOR ROBUSTNESS AND MAINTAINABILITY 1
1.2 THE PROJECT GOAL IS TO SIMPLIFY THE PROCESS 1
1.3 PROCESS REQUIREMENT 1 - DATA GATHERING 2
1.4 PROCESS REQUIREMENT 2 - DATA PROCESSING 2
1.5 PROCESS REQUIREMENT 3 - DATA REPORTING 2
1.6 TRIAL RUN ON HTI 2
1.7 CONCLUSION: GENERIC PROCESS 3

CHAPTER 2 BACKGROUND **4**

2.1 PURPOSE OF BIT 4
2.2 BIT REQUIREMENTS – DEFINITIONS 5
2.3 COMMON BIT REQUIREMENT VERIFICATION METHODS 7
2.3.1 Who Performs BIT Requirements Verification? 9
2.3.2 BIT Requirements Verification in the System Development Process 10
2.4 DIFFICULTY OF VERIFYING REQUIREMENTS ON EXISTING SYSTEMS 12
2.4.1 Adverse Factors in Existing Designs 13
2.5 DESCRIPTION OF HTI SYSTEM 14
2.6 BIT REQUIREMENTS ON HTI 15
2.7 DESCRIPTION OF THE NEED FOR BIT ANALYSIS ON HTI 17
2.7.1 Original Analysis Results 17
2.7.2 Modification of Original Data 18
2.7.3 Current HTI BIT Analysis Status 19

CHAPTER 3 ANALYSIS **20**

3.1 BIT STATISTICAL METHODS 20
3.1.1 Definitions 20
3.1.2 Functional Reliability versus Component Reliability 21
3.1.3 System Mapping – Functions to Components 24
3.1.4 Failure Types in Functional Reliability Analyses 25
3.1.5 Failure Rate versus Failure Probabilities 28
3.1.6 FMEA/FMECA 32
3.2 BIT REQUIREMENT CALCULATIONS 35
3.2.1 Fault Detection Calculation 35
3.2.2 Fault Isolation Calculation 35
3.2.3 False Alarm Rate Calculation 36
3.3 TYPES OF DATA NEEDED 37
3.3.1 System Functional Data 37
3.3.2 Physical Decomposition Data 37
3.3.3 Component Data 38
3.3.4 Existing Reliability Analysis Data 38
3.4 DATA ALTERNATIVES 39
3.5 DATA PROCESSING AND ANALYSIS 40
3.5.1 Complete the FMEA 40
3.5.2 Perform the Criticality Analysis 42
3.5.3 Perform BIT Requirement Calculations 42
3.6 DOCUMENTATION OF THE ANALYSIS - REPORTS 43
3.7 TOOLS 44

3.7.1 Advanced Specialty Engineering Networked Toolkit (ASENT) 45
3.7.2 Computer Aided Reliability and Maintainability Applications (CARMA) Toolkit 46
3.7.3 RAM Commander 48

CHAPTER 4 COMPARISONS 51

4.1 FINAL PROCESS 51
4.2 APPLICATION OF PROCESS TO HTI 53
4.2.1 Assumptions 53
4.3 DATA RESULTS 56
4.4 ANALYSIS OF RESULTS 57
4.5 ASSESSMENT OF PROCESS 58
4.5.1 Ease of Use 58
4.5.2 Time Savings 59
4.5.3 Tailorability 59
4.5.4 Tool Usage 59
4.5.5 Verifies BIT Requirements 59
4.5.6 Clarity of Directions 60
4.5.7 Cost Savings 60

CHAPTER 5 CONCLUSIONS 62

5.1 BENEFITS OF THE PROCESS 62
5.2 APPLICABILITY TO ANY PROGRAM 62
5.3 DISCUSSION OF SYSTEMS ENGINEERING/TRANSDISCIPLINARY PROCESS 63
5.3.1 Application of Transdisciplinary Process to Project 63
5.3.2 Transdisciplinary Process in the BIT Analysis Process 64

CHAPTER 6 REFERENCES 65

CHAPTER 7 APPENDIX I – NOMENCLATURE 68

CHAPTER 8 APPENDIX II – HTI BIT ANALYSIS 70

CHAPTER 9 APPENDIX III – PROCESS FLOWCHARTS AND INSTRUCTION SHEETS 73

CHAPTER 10 APPENDIX IV – PROCESS TEMPLATES 111

CHAPTER 11 APPENDIX V – NEW HTI BIT ANALYSIS 116

List of Figures

FIGURE 1: BIT HAS SPECIFIC MEANINGS FOR SPECIFIC ASSEMBLY LEVELS 7
FIGURE 2: FAULT COVERAGE 11
FIGURE 3: SOURCES OF BIT FAILURES 12
FIGURE 4: HTI B-KIT COMPONENTS 15
FIGURE 5. EXAMPLE OF A RELIABILITY BLOCK DIAGRAM 22
FIGURE 6. EXAMPLE OF A FUNCTIONAL BLOCK DIAGRAM 23
FIGURE 7. FUNCTIONAL TO PHYSICAL TO COMPONENT MAPPING 25
FIGURE 8 EXAMPLE OF A BINARY MATRIX RELATIONSHIP BETWEEN FUNCTIONS AND COMPONENTS 25
FIGURE 9. SOFT AND HARD FAILURES OF ELECTRONIC COMPONENTS 27
FIGURE 10. BATHTUB CURVE DEPICTING COMPONENT FAILURE RATE AS A FUNCTION OF TIME 31
FIGURE 11. CRITICALITY MATRIX 34
FIGURE 12. DATA TYPES NEEDED FOR BIT ANALYSIS 38
FIGURE 13. CARMA ARCHITECTURE 46
FIGURE 14. FMECA SYSTEM EXECUTIVE 47
FIGURE 15. RAM COMMANDER FMECA SCREEN 49
FIGURE 16. TOP LEVEL BIT ANALYSIS PROCESS FLOWCHART 52

List of Tables

TABLE 1. COMMON BIT REQUIREMENT VERIFICATION METHODS 8
TABLE 2. HTI BIT DETECTION AND ISOLATION REQUIREMENTS AND PREDICTED PERFORMANCE 17
TABLE 3. B-KIT FAILURE RATE ADJUSTMENTS – ORIGINAL BIT ANALYSIS 18
TABLE 4. FUNCTIONAL DECOMPOSITION 24
TABLE 5. QUALITATIVE APPROACH CLASSES 29
TABLE 6. COMMON FAULT SIGNATURE GROUPINGS 36
TABLE 7. SAMPLE DATA TABLE FOR REPORT 44
TABLE 8. ASSEMBLY LEVELS ANALYZED 56
TABLE 9. HTI BIT DETECTION AND ISOLATION REQUIREMENTS FROM THE NEW ANALYSIS 58

CHAPTER 1
INTRODUCTION

1.1 BIT Requirements For Robustness and Maintainability

Most products developed and marketed today not only need to be of high quality, but also need to be well maintained and robust. As a result, product specifications have requirements for a system level built-in test (BIT). These requirements cover BIT detection percentages, false alarm rates, fault isolation percentages and other parameters to be measured at a system level. The BIT tests facilitate maintainability and robustness by correctly identifying system failures. All hardware related functionality is covered by the tests and thus BIT can provide an accurate assessment of system operability.

Successful completion of the BIT tests after power-up serves as an indicator that the system is fully functional. If the BIT tests are incomplete or ill-defined and do not test a portion of the system, then the 'pass' indicator may be a false indication of system health. Similarly, if the tests fail when the system is not broken, the indicator wrongly declares the system inoperable. Responsibility lies on the system designer to verify the BIT tests fully and correctly test system functionality. BIT analyses are performed to prove the BIT tests meet the customer requirements.

As more and more emphasis is placed on developing quality products that can be maintained and as improvements are made to existing designs, the need for robust self-tests is increased. Customers are requiring designers to detect a certain percentage of failures – accurately. As the designers develop these BIT tests, it becomes increasingly difficult to prove to the customer that they meet the requirement for false alarm rates, detection percentages and isolation percentages. In most cases the designers will implement BIT tests based on a BIT analysis performed on the initial design and incrementally improve the BIT based on trial and error. The BIT analysis for detection percentages, isolation percentages and false alarm rates is not performed at all. It is too difficult and time consuming.

1.2 The Project Goal is to Simplify the Process

The purpose of this project is to develop a process to perform an analysis of the BIT on an existing system. New designs will not be considered to reduce the scope of the project and because the more difficult, and common, task is to analyze a system that has already been

designed. The analysis will determine the failure detection percentages, fault isolation percentages and false alarm rates of the BIT implemented on the current system. The process developed in this project will include an easy to follow process, adaptable to any system, and automated tools to simplify equations and data gathering. The final product of this project is a BIT Analysis Process. The BIT Analysis Process is a standardized process to aid an engineer in determining the BIT detection percentage and false alarm rates for an existing product and create a report of the results.

1.3 Process Requirement 1 - Data Gathering

One of the most difficult tasks in verifying BIT requirements, especially on an existing design, is sorting through volumes of program data. Very specific information is needed to verify the requirements. The goal is to minimize the data gathering effort so the appropriate data can be sought and irrelevant data ignored. The BIT Analysis Process provides a standardized data gathering technique/template.

1.4 Process Requirement 2 - Data Processing

The analysis of the data is the stage where verification of the BIT requirements takes place. After the appropriate data has been gathered, system functionality and configuration is analyzed to determine the full set of operations to be tested. Using this information, the data is applied to equations to calculate the numbers for BIT detection, fault isolation and false alarm rates specified in the requirements. The BIT Analysis Process will provide a tool to process the data and calculate the numbers specified by the BIT requirements.

1.5 Process Requirement 3 - Data Reporting

Verification of the BIT requirements requires documented proof. The data gathered and analyzed in the BIT Analysis Process must be presented in a form that provides sufficient information to prove the verification/validation of BIT requirements. The BIT Analysis Process will provide a method to output the results of the calculations in such a manner to show how the results were generated and provide validity to the results.

1.6 Trial Run on HTI

As an example and the test system for the process, this project will include a BIT analysis on the HTI Second Generation Forward Looking Infrared (FLIR) sensor. The HTI FLIR is an existing design that has BIT tests implemented and specific requirements for failure detection

percentages, fault isolation percentages and false alarm rates. An analysis was performed early in the program on a previous design and no re-analysis of the system has taken place. The program currently has no way of verifying these requirements.

1.7 Conclusion: Generic Process

In conclusion, the project will create a BIT Analysis Process that is a standardized process to verify BIT requirements on an existing system. It will be used on one program, as an example, to verify its reliability but will be adaptable to other programs.

CHAPTER 2
BACKGROUND

2.1 Purpose of BIT

Built-in-Test (BIT) can generally be described as a set of evaluation and diagnostic tests that uses resources that are an integral part of the system under test. [Drees, 2002] Historically, a lack of attention and standards by professional and academic associations has led to confusion over the role of BIT. The lack of standardization is responsible for various different definitions of BIT at numerous system levels. The assorted definitions contribute to the confusion over the specific role of BIT. However, the role of BIT and its usefulness, at all levels, has been recognized in electronic equipment as early as the 1950s. [Goodman, 1967]

The role of BIT was originally designed to ensure uninterrupted availability and fault free operation of critical weapon systems (Minutemen I and II missiles) and aerospace equipment (Saturn, Apollo). [Pecht, 2001] BIT is also used for in-field maintenance by the end user, to indicate system status, and to indicate whether a system has been assembled properly. As a result, BIT has been used in diverse applications including oceanographic systems, multi-chip modules, large-scale integrated circuits, power supply systems, avionics, and even passenger entertainment systems for the Boeing 767. [Pecht, 2001] Although the general concept of BIT remains the same, each specific field has its own specialties and adaptations. [Gao, 2001]

The role of BIT in electronic systems has grown in prominence with the advances in system complexity and concern over maintenance lifecycle costs of large systems. Today, in an environment where standards drive systems designs (and provide an avenue for focused advancement in technology), standards for BIT are finally evolving. The reasons for advancing the effectiveness of BIT include reduced overhead for support, greater confidence in operation, and increased system availability. [Drees, 2002]

With the increase in the importance of the role of BIT and the development of standards for BIT comes the development of BIT requirements on programs. The cost of supporting military electronic systems (avionics, communications, and weapons systems) has driven much of the development in BIT technology and requirements development. There has also been a beneficial effect on the maintenance and availability of test and measurement instrumentation,

Automatic Test Equipment (ATE), due to the requirement for BIT in their component assemblies. BIT has found a role in all program phases including Design Verification Test, Integration Verification and Validation, Hardware Integration, Hardware-Software Integration, System Integration, Verification and Validation, System Qualification and Reliability Testing, Factory, Production and Depot Test, and Field Support. [Sallade, 1999] But as the role of BIT becomes a requirement, a new need is created: a method to verify the BIT requirements.

2.2 BIT Requirements – Definitions

Several definitions of BIT, based on the system level at which they are executed, exist. Requirements for BIT are generally in the form of specific values for BIT detection percentages, fault isolation percentages and false alarms rates. Definitions for the key BIT requirement terms are below:

Term	Definition
BIT	Built in Test – An on-board hardware/software diagnostic means to identify and locate faults. It includes error detection and correction circuits, totally self-checking circuits and self-verification circuits. [Drees, 2001]
BIT Detection	Proportion of system failures detected automatically by BIT. The higher the BIT detection percentage, the better the capability. [Giordano, 2001]
BIT False Alarm	Alarms that occur during system operation but cannot be later duplicated. [Steinmetz, 2002]
BIT Fault Isolation	Isolating a failure by BIT to a subsystem or lower level part. Generally in terms of fault isolating to one item X% of the time and N or fewer items Y% of the time. The higher the fault isolation percentage, the better the capability. [Giordano, 2001]
CBIT	Continuous BIT. Equipment is monitored continuously and automatically without affecting normal operation. [Pecht, 2001]
Fault	Erroneous state of hardware or software resulting from failures of components, physical interference from the environment, operator error or incorrect design. [Siewiorek, 1982]
Fault Detection Time	The time which elapses between the occurrence of a fault and the detection (reporting) of the fault by BIT. [MIL-STD-2165, 1989]

Fault Location	Physical points within the design which are subject to failure. Fault locations can be detectable, non-detectable, or excluded from system consideration (minimal impact to system functionality). [Giordano, 2001]
IBIT	Interruptive BIT. Normal equipment operation is suspended while test take place. IBIT normally occurs at power up or is initiated by the operator. [Pecht, 2001]
Instrument Level BIT	Module/Assembly level BIT. BIT is performed on Module level and Line Replacable Units (LRUs) that are unique electronic assemblies comprising of one or more circuit card assemblies. [Drees, 2001]
PBIT	Periodic BIT. An IBIT which interrupts normal operations periodically in order to carry out a pseudo-continuous monitoring function. [Pecht, 2001]
Self Test BIT/ BIST	Built in Self-Test. Circuit level tests – BIT techniques applied to circuit cards using digital and analog methods. [Drees, 2001]
System Level BIT	System level self-test. BIT based upon the similarity of high-level requirements and concerns such as: BIT detection percentage, execution time, false alarm rates, and fault isolation. [Drees, 2001]
Test	A routine that stimulates a portion of the circuitry, measures the response, and then compares the result to a known or desired value. [Giordano, 2001]

The term BIST is sometimes used interchangeably with BIT. However, BIST is usually associated with low level circuit card assembly tests. This level of testing is characterized by well-developed standards, technologies (IEEE 1149, boundary scan, and signature analysis) and tools. [Drees, 2001] The middle level of BIT testing, the Instrument Level, refers to programmable instrumentation in supports systems or LRUs in military systems. The highest level, System Level BIT, tests the entire system. A key requirement of this level of test that is not levied on the other BIT levels is fault isolation, the ability for a test to diagnose the failing lower level LRUs. Figure 1 shows an example of the different levels of BIT testing.

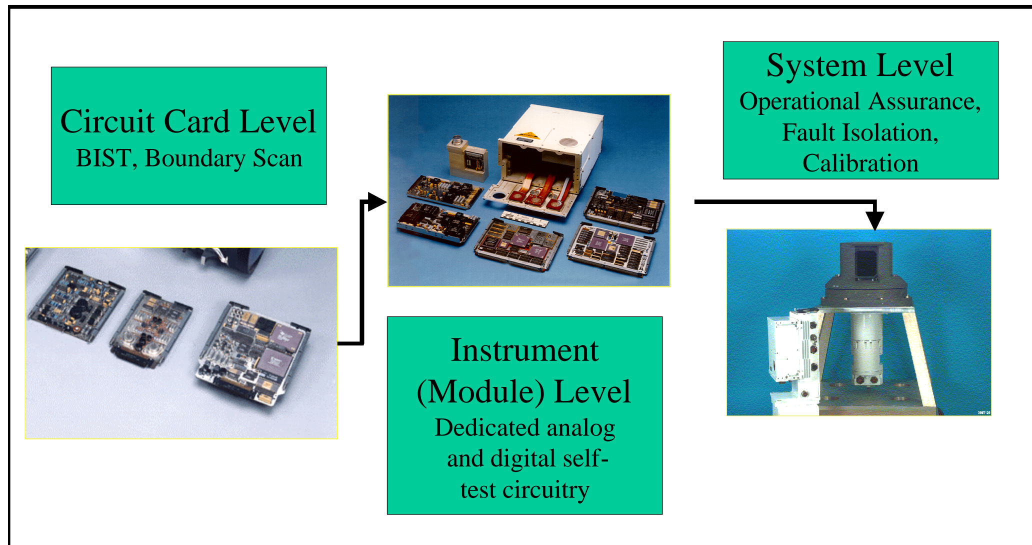


Figure 1: BIT has Specific Meanings for Specific Assembly Levels

BIT requirements represent the minimum essential levels of performance desired by the customer. BIT requirements can include the following types of requirements [MIL-STD-2165, 1989]:

- The proportion of failures to be detected automatically (versus manual troubleshooting such as swapping large items, manual probing, etc)
- Maximum acceptable failure latency
- Frequency at which periodic BIT will run
- Maximum BIT false alarm rate
- Fault isolation by BIT to a subsystem or lower level part
- Maximum fault isolation time

2.3 Common BIT Requirement Verification Methods

A number of approaches to verifying BIT requirements are used on military and commercial programs. Each approach is valid but has strengths and weaknesses in different areas. [Devlin, 1999] The verification method is not always chosen for appropriateness – at times it is directed by the customer, engrained in the company process or the only available option. Industry standards have been created for initiating BIT and for formats of error and status responses. However, no industry standard has been published for BIT fault coverage or isolation, the two most common BIT requirements. [Drees, 2001] Table 1 lists several common verification methods with a description and an explanation of the strengths and weaknesses of the method.

Table 1. Common BIT Requirement Verification Methods

Verification Method	Description	Pros/Cons
Component Based Methods	System design is analyzed and a list of all testable components generated. BIT tests are created to test all components. Components are removed from the test list if they become too time consuming or costly to test. Fault detection percentages are based on the number of components tested versus the number of testable components.	Pros: All testable hardware components are considered for BIT. There is a small chance of failures occurring undetected. Cons: Costly approach due to the number of BIT tests required. BIT circuitry may have to be added to test some components. Many BIT tests are redundant. Fault isolation becomes difficult. False alarm rates can be high. True component based analysis is difficult because it requires in depth system knowledge.
Statistical Methods	Statistical reliability data is gathered on the system components. BIT tests are designed on probability of failure based on the failure rate of the component and where it is used. If a component has a high probability of failure, a BIT test is created for it. Components with low failure rates are not tested. Fault detection percentages are based on the number of high probability of failure components tested.	Pros: Redundant tests are eliminated because every component is not tested. Cost effective because BIT development time and dollars are focused on problem areas. Cons: System functionality is not considered. Important functionality can be overlooked because it is very reliable, but the associated failure is mission critical. High-risk components that can cause loss of functionality (mission critical) or safety critical items may not be included in the BIT tests. Fault detection percentages may be low.
Functional Based Methods	The functionality of the system or subsystem is analyzed and a BIT test is created for each functional task. Complex functions may require several BIT tests. Fault detection percentages are based on the functional tasks tested.	Pros: All critical components are tested as part of their associated function. Mission and safety critical components are included as part of the functionality. Cons: All combinations of components that can lead to loss of functionality may not be considered. In this case, fault detection may not cover all functionality. Low reliability components (that are not critical) may not be tested.

Verification Method	Description	Pros/Cons
System Level Test Methods	BIT tests are created only to fill gaps that cannot be tested at a system level. All functionality that can be tested at a system level is excluded from BIT. 'As good BIT is, in our experience, it should only be used as a building block when used in a system-level test.' [Drees, 2001] Fault detection percentages are based on the number of functional tasks not tested at system level.	Pros: BIT development is kept at a minimum. Additional BIT circuitry is rarely needed. Fault detection percentages cover only a small part of the system. Cons: This method assumes system levels tests will be performed often and are readily available. If the system level test is not performed often enough, fault detection drops dramatically. An incomplete system analysis may leave some functionality not tested.
Customer Defined Methods	Customer defines a set of BIT tests to be executed on a program. Percent detection is based on the number of tests performed on the customer-defined list.	Pros: The fault detection rate is easily verified. Responsibility to verify BIT requirements can fall on the customer and not the contractor. Cons: Only as good as the data provided by the customer. Some components and functionality may not be included. A false alarm rate requirement can be difficult to meet if the customer defined BIT tests do not cover the system adequately.

2.3.1 Who Performs BIT Requirements Verification?

BIT requirement verification is a task that can be performed by several different groups. Systems Engineers or Specialty Engineering (Reliability, Maintainability, etc) engineers on the program sometimes perform the verification. Some companies have separate functional groups specializing in requirements verification that perform the analysis and requirements verification. Others may use outside contractors or consultants that are hired to perform a one-time BIT analysis and verification. In some cases, the customer may perform the BIT requirements verification.

Program Engineers – Program engineers provide continuous analysis as program time and dollars permit. Engineers have the most intimate knowledge of the system design. BIT tests tend to be thorough and well documented, but there is a chance of the final verification results and analysis being skewed to show BIT requirements are met. Engineers have a vested interest in the program

succeeding and may provide the design information in such a way to ensure the verification goes favorably.

Functional Group – Tends to be one of the more accurate analyses because, since it is an internal (to the company) group, information is more readily available. Program engineers can be interviewed and actual hardware examined. Questionable results can be reevaluated. Difficulty comes when trying to adjust the requirements definitions from the customer point of view to the traditional requirement definitions of the group.

Consultants – Consultants review the design and suggest BIT tests. Their analysis shows how the list of BIT tests satisfy the BIT requirements. The analysis is not biased and is usually very thorough based on the information the consultants gather. This requires much coordination from program engineers to provide information to the consults. However, the assumption has to be made that all the BIT test are implemented and that the system design does not change. Consultants tend to be a one-time effort so programs with changing designs are not benefited from the method of verification.

Customer – The benefit of the customer performing the verification is that the results, if satisfactory, are automatically accepted by the customer. Accurate results are hard to come by because the system may not be as well known by the customer as by the program engineers. However, the analysis can be revisited because the customer has constant access to the program.

2.3.2 BIT Requirements Verification in the System Development Process

BIT Requirements verification can start in any phase of the system development process. The final verification cannot take place until the system is fully developed and the design is stable. However, the earlier the BIT requirements verification begins, the easier it is to obtain results.

The System Development Cycle consists of four main phases: Design, Development, Test and Production. If BIT analysis and requirement verification efforts begin in the design stage, the appropriate BIT circuitry can be included in the design, design information can be collected for the required analysis and information intentionally stored to help with the verification process. One disadvantage to start BIT requirements verification in this phase is that there will be many changes to the design and some effort will be wasted will the designs change.

In the development phase, actual design data is captured and BIT tests are being implemented. Reliability tools, computer programs that take reliability data and compute testability numbers, can be used in this phase to capture design data and automatically produce verification proof. The reliability tools are extremely helpful, but care must be taken to ensure all system components are considered or realize the output of the reliability tools is not a complete assessment. There is a smaller risk of design change in this phase.

The test phase marks an essentially complete design. Only minor changes should take place in this phase. All possible failures may not be detectable by BIT circuitry and it may not be cost effective to implement logic to detect them. As the program develops, more BIT tests are developed and the failure detection percentage increases. Likewise, the risk to the program decreases. Eventually a point is reached where the risk is less of a cost to the program than implementing more fault detection measures. Figure 2 shows fault coverage takes a logarithmic curve with respect to time.

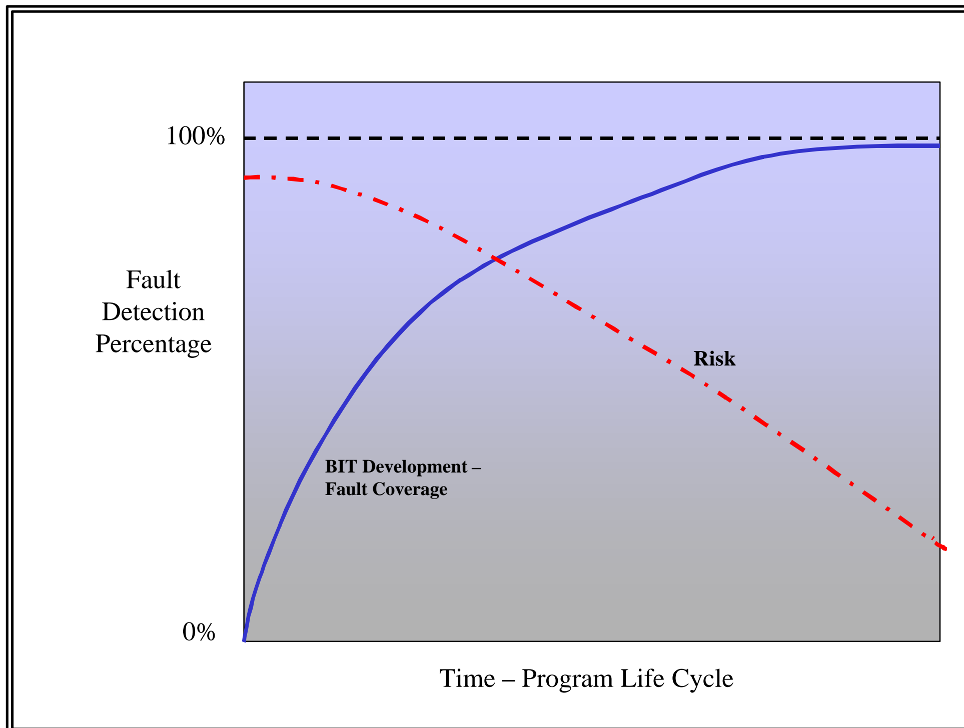


Figure 2: Fault Coverage

All sources of BIT failures can be identified the testing phase. Not all BIT problems are a result of the system design. BIT tests can fail due to certain environments and hardware or even operator mistakes. See Figure 3 for an example of the many different causes of BIT failures. The testing performed in this phase identifies many of such issues. The BIT requirements verification should be complete by the end of this phase.

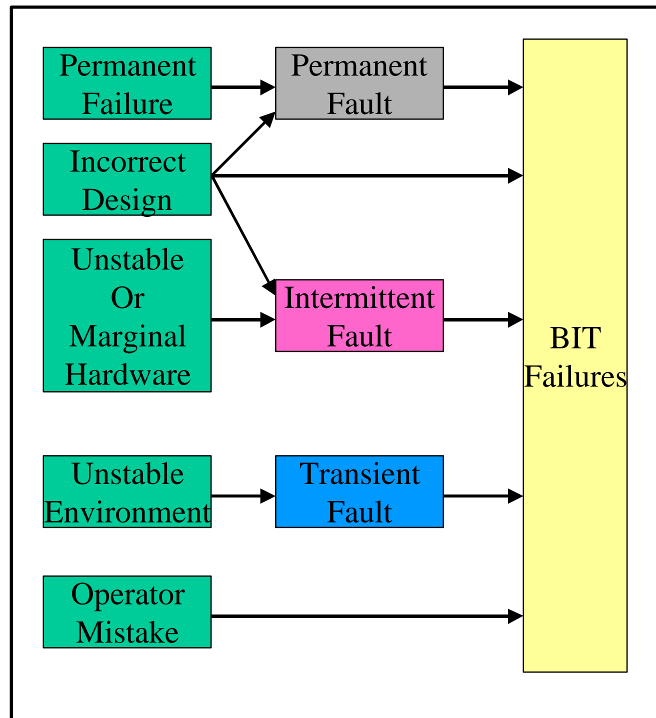


Figure 3: Sources of BIT Failures

In the production phase, the design has been finalized and all BIT requirements should have been verified. Occasionally minor changes or enhancements may take place during production and the BIT requirements verification may need to be revisited. Lack of adequate documentation of the BIT verification can make this task difficult.

2.4 Difficulty of Verifying Requirements on Existing Systems

Despite the apparent sophistication of BIT, and the fact the BIT requirements have been verified, there has been some concern that the requirement for BIT and the actual capabilities and limitations of the BIT have not been properly identified. For example, airline experience with

modern avionics systems has indicated spurious fault detection is unacceptably high. The system BIT requirements have not actually been met.

For instance, the Airbus A320, Lufthansa, had a daily average of 2000 BIT failures logged. 70 of these corresponded to failures reported by pilots, while another 70 faults not detected by BIT were reported by pilots. Of the 17 line replaceable units (LRUs) replaced every day, typically only 2 were found to have faults that corresponded with the report. Thus even for commonly observed problems, fault detection is not complete and false alarms occur. [Pecht, 2001]

The Airbus A320 is an existing design in which new functionality and capabilities have been added through the years. As this occurs, the supporting BIT was developed and verified. Why is there such a discrepancy in fault detection and reporting? It is because of the difficulty in verifying BIT requirements on an existing system.

2.4.1 Adverse Factors in Existing Designs

With a new design or a development project, information can be gathered for the BIT analysis and verification as the design proceeds. BIT is planned for and is a consideration in the design. BIT circuitry is added and the design is evaluated so the engineers know what is necessary to meet BIT requirements.

This is not the case in a situation where BIT requirements are being verified on an existing system. BIT may have been planned in the development of the program or it may have been added later. Several common problems and concerns when verifying BIT requirements are listed below:

- Incomplete or incorrect BIT analysis when starting with a baseline. This can occur when assuming the original or previous BIT analysis was correct or complete. In this case the existing numbers are used and only the new BIT tests and functionality are examined. Sections can be missed in the older analysis or performed incorrectly, causing the result to be incorrect.

- Incompatible data types when starting with a baseline. Reusing older BIT analysis data can be misleading if the data format is not understood. Percentages and rates should be clearly defined to ensure the definitions are

compatible with the way they are being used in the new analysis. Lack of attention to this detail can lead to incorrect results.

- Lack of data to perform analysis. Older programs may not have documented all program design, or it may have been lost, or what is in the current documentation may be out of date.

- True state of current system may not be known. The history of the program may not be known – decisions or changes may not be documented. The true configuration of the system could require reverse engineering to determine the current state.

- The task is overwhelming and not cost efficient. The program budget may not allow for the time necessary to reanalyze an entire system or a large system that has had multiple changes.

2.5 Description of HTI System

As an example and the test system for the BIT Analysis process, this project will include a BIT analysis on the HTI Second Generation Forward Looking Infrared (FLIR) sensor. The HTI FLIR is a scanning infrared receiver that collects radiation in the 8-12 micron spectral region across two different 2-D fields of view using a second generation IR focal plane array. The SGFLIR is composed of several modules, each having pre-defined interfaces and physical connections that allow it to function as a stand-alone unit or to be integrated into a host platform. It is designed to be versatile – with most of its image processing implemented in software and firmware to improve flexibility and decrease the complexity of the hardware. [MIL-PRF-A3207380, 1997]

The HTI FLIR, commonly called a B-KIT, is packaged so that it can be housed on multiple platforms. Thus, there is some functionality related to the particular host using it and this functionality must be tested with BIT. Figure 4 shows the components that comprise a B-KIT. The number of circuit card assemblies (CCAs) gives an indication to the complexity of BIT required to test all functionality.

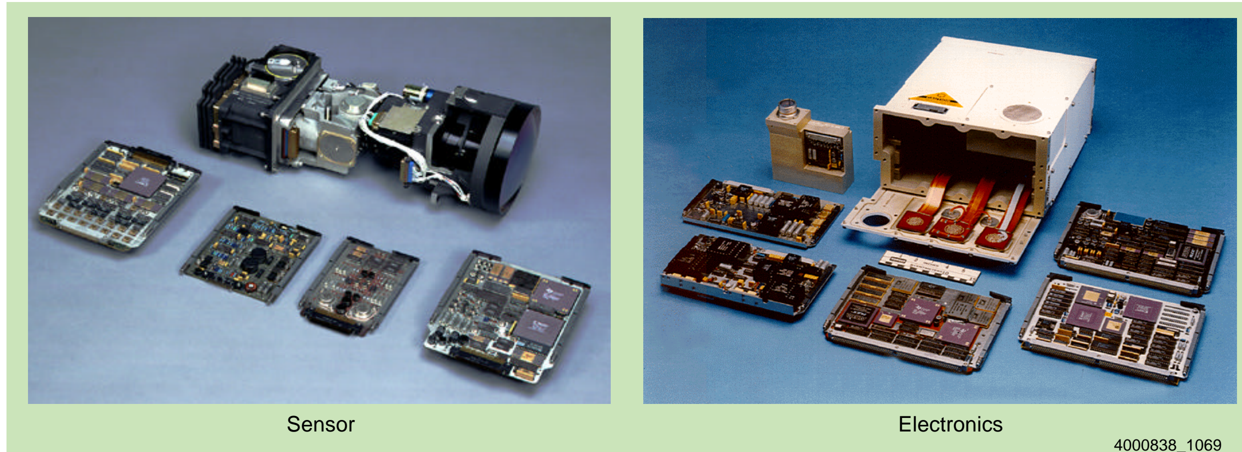


Figure 4: HTI B-Kit Components
[Raytheon4, 2002]

2.6 BIT Requirements on HTI

The HTI FLIR is an existing design that has BIT tests implemented and specific requirements for failure detection percentages and false alarm rates. The BIT requirements are defined in the HTI Performance specification and are listed below: [MIL-PRF-A3207380, 1997]

1) BIT quantitative requirements. For the purpose of the following subparagraphs, the term BIT includes IBIT, the interactive BIT of the host platform, and operator detection/isolation of failures. Operator detected/isolated failures will be such that they are obvious and require no judgment to determine that the observed artifact or condition is a failure.
2) Detection by BIT. The B-Kit shall be designed such that at least 90 percent of the B-Kit failures can be detected using BIT.
3) Isolation by BIT. BIT shall isolate to the failed LRU for at least 85 percent of the B-Kit's failures.
4) BIT false alarm rate. BIT false alarms shall not exceed a 2 percent rate when no failure is present.
5) Power-up BIT (PBIT). The B-Kit shall provide the capability for PBIT tests that are performed on the B-Kit at power up. At a minimum, PBIT shall test and determine the status of all applicable power conditioning devices, the basic processing capability, and the status of all major internal and external communications ports of the B-Kit. PBIT shall not be abortable. At the

completion of PBIT the B-Kit shall enter the operate mode regardless of BIT status.

6) Start-up BIT (SBIT). The B-Kit shall provide the capacity for BIT prior to the complete cooldown of the detector/dewar assembly. SBIT shall consist of all initial tests not conducted at power-up. SBIT shall be abortable. SBIT, when coupled with PBIT, CBIT, and the interactive BIT of the host platform, shall be capable of detecting at least 90 percent of the NV-80 B-Kit's mission critical failures. SBIT shall not be performed if the focal plane array (FPA) is already at operating temperature when power is applied to the B-Kit.

7) Continuous BIT (CBIT). The B-Kit shall provide the capability for CBIT performed while the B-Kit is operating to detect mission affecting failures. This CBIT, when coupled with the interactive BIT of the host platform, shall detect at least 85 percent of mission critical failures in the B-Kit. CBIT shall detect at least 90 percent of the mission critical failures in the B-Kit when coupled with the interactive BIT of the host platform and operator detection. These tests shall not degrade the B-Kit imagery or symbology performance.

8) Operator initiated BIT (IBIT). The B-Kit shall provide the capability for operator IBIT tests. This IBIT shall be completed within 35 seconds. The B-Kit is not required to be imaging while these tests are being performed. Operator IBIT shall be abortable via the BIT interrupt.

9) Fault isolation using BIT. The B-Kit shall make provisions to allow external control of those BIT tests where the state of a given signal must be held for external observation per the B-Kit ICD.

Table 2 shows the BIT requirements and the predicted performance for older versions of the B-Kit. The entries in the table are a result of the original B-KIT BIT analysis and requirements verification. The performance is evaluated with two different B-KIT software versions (v3.01 and v4.0) and two different hardware configurations (Engineering Manufacturing and Development (EMD) and Low Rate Initial Production (LRIP)). The false alarm rate requirement is not addressed in the analysis and is not included in this table.

	Detection % (+/- 5%)				**Fault Isolation % (+/- 5%)**
	CBIT + Host Platform Mission Critical Detection	**CBIT + Operator + Host Platform Mission Critical Detection**	**PBIT + SBIT + CBIT + Host Platform Mission Critical Detection**	**IBIT + Operator + Host Platform Detection**	**IBIT + Operator + Host Platform Isolation**
Current B-KIT Performance Spec	85%	90%	90%	90%	85%
Version 3.01 EMD Predicted Performance	86%	90.1%	91.9%	90.8%	85.9%
Version 4.0 LRIP Predicted Performance	86%	90.1%	91.9%	90.8%	89.6%

Table 2. HTI BIT Detection and Isolation Requirements and Predicted Performance [Raytheon1, 1997]

2.7 Description of the Need for BIT Analysis on HTI

An analysis was performed in 1997, early in the HTI program, on a previous design (the EMD hardware) and predictions made for the newer LRIP hardware. Pertinent data from the original analysis is contained in Appendix II.

2.7.1 Original Analysis Results

The analysis emphasized a reliability-based approach to determine the detection and isolation performance of the B-Kit BIT functions. The study examined individual components or functional groups of components and determined, via analysis, if the BIT software would detect a failure in the component. If it could detect the failure then the failure rate for that component, or a percentage of the failure rate, was added to the BIT software detection "bucket". The same process was used for determining Operator and Host Platform detection percentages.

After all components were examined a total was struck and detection percentages computed for each detection category. These in turn were combined to arrive at percentages for each combination in Table 2. Failures that could be detected by multiple categories were only counted once when computing the roll ups to avoid duplication.

The above purely analytical approach was augmented by referring to experience derived from failure injection, actual use of the system, and experience on other programs. Isolation predictions for the BIT software were determined by applying the software Version 3.01 fault isolation algorithm to the predicted BIT failures. Rules were also established for Operator and

Host Vehicle Isolation and these rules were applied to the predicted detection results. [Raytheon1, 1997]

2.7.2 Modification of Original Data

Additional modifications to the failure rates listed in the reliability reports resulted from actual experience with the system since the reliability report was last updated in December 1995. An example of such a modification is the Afocal assembly failure rate. The reliability report failure rate of 9 was judged upwards based on experience and by comparison with other programs. The failure rate from the IBAS Afocal reliability prediction was used to adjust this number upwards to a failure rate of 43. The updated failure report also was incomplete in several other sections. In these situations failure rates from the original report or from similar components were used. [Raytheon1, 1997] These adjustments to the failure rates are summarized in Table 3.

Assembly Description	Failure Rate from Reliability Report of 12/95	Adjusted Failure Rate
Sensor Unit	397.5840	433.403
Afocal Telescope Assembly	9.2392	42.7623
Cooler Control CCA	10.1472	10.143
Scanner Control CCA	22.4352	22.41
Digitizer CCA	20.3098	20.3107
Point of Load Regulator CCA	9.7960	10.1210
Detector/Cooler Bench	290.0000	292.0000
Imager Assembly	35.6558	35.656
Electronics Unit	208.0650	195.5066
Video Processor CCA	13.3285	22.0133
Interface Control CCA	33.9667	24.0201
Video Converter CCA	42.0979	56.8129
Power Converter #1 and #3	36.0812	33
Contrast Enhancement/Frame Int	25.0000	2
Power Converter #2 CCA	10.1846	10.2
Chassis	47.4060	47.4603
Total	605.6490	628.9096

Table 3. B-KIT Failure Rate Adjustments – Original BIT Analysis [Raytheon1, 1997]

2.7.3 Current HTI BIT Analysis Status

The original 1997 BIT analysis has not been revisited. The HTI B-KIT program has progressed. The EMD hardware has been discontinued and is no longer supported. The LRIP hardware, for which BIT detection percentages were predicted, has been updated by SMT (Surface Mount Technology) hardware and the software has progressed to version 7.0. The BIT analysis is no longer valid and the HTI program currently has no way of verifying the BIT requirements listed in section 2.6.

There is a need to perform the BIT Analysis on the HTI program to verify the BIT requirements on the B-KIT.

The suitability and effectiveness of BIT will be judged in terms of its ability to meet the desired objectives. The common modes of failures versus the failure types detected by BIT and the correctness of BIT results (defined as the location and type of actual failure versus the failure indicated) are analyzed to verify the BIT requirements. [Pecht, 2001]

The objective of this study is to develop a standardized process that has not been available to program engineers before. The results will include a simplified process, guidelines and outputs.

CHAPTER 3
ANALYSIS

3.1 BIT Statistical Methods

Failure modes, effects and their severity are the primary factors examined in the proven statistical methods commonly accepted by reliability engineering. These types of analyses are widely used to calculate data needed for reliability and BIT analysis efforts. Failure rates and probabilities are key to the calculations. The sections below discuss the methods employed.

3.1.1 Definitions

Key terms used in the reliability analyses are defined below:

Criticality	A relative measure of the consequences of a failure mode and its frequency of occurrence [MIL-STD-1629, 1984]
Criticality Analysis	A procedure by which each potential failure mode is ranked according to the combined influence of severity and probability of occurrence [MIL-STD-1629, 1984]
End Effect	The consequences a failure mode has on the operation, function, or status of the highest indenture level. [MIL-STD-1629, 1984]
Failure Effect	The consequence a failure mode has on the operation, function, or status of an item. Failure effects are classified as local effect, next higher level, and end effect. [MIL-STD-2165, 1985]
Fault Coverage / Fault Detection	The ratio of failures detected by BIT, expressed as a percentage. [MIL-STD-2165, 1985]
FMEA	Failure Modes and Effects Analysis. A procedure by which each potential failure mode in a system in analyzed to determine the results of effects thereof on the system and to classify each potential failure mode according to its severity. [MIL-STD-1629, 1984]

FMECA	Failure Modes, Effects, and Criticality Analysis. A FMEA with the results of the criticality analysis incorporated. [MIL-STD-2165, 1985]
Local Effect	The consequences a failure mode has on the operation, function, or status of the specific item [MIL-STD-1629, 1984]
Next Higher Level Effect	The consequences a failure mode has on the operation, functions, or status of the items in the next higher indenture level above the indenture level under consideration [MIL-STD-2165, 1985]
Severity	The consequences of a failure mode. Severity considers the worst potential consequence of a failure, determined by the degree of injury, property damage, or system damage that could ultimately occur [MIL-STD-1629, 1984]
Undetectable Failure	A postulated failure mode in the FMEA for which there is no failure detection method by which the operator is made aware of the failure. [MIL-STD-1629, 1984]

3.1.2 Functional Reliability versus Component Reliability

Two different distinct approaches exist for reliability and BIT analysis methods. The approaches focus on how the system is decomposed for reliability calculations. The traditional method is to decompose the system into components, or parts. Figure 5 shows an example of a reliability block diagram, or parts decomposition, used for this approach. The second approach focuses on the functions performed by the system and performs calculations based on components satisfying a specific function. Figure 6 shows an example of a functional block diagram.

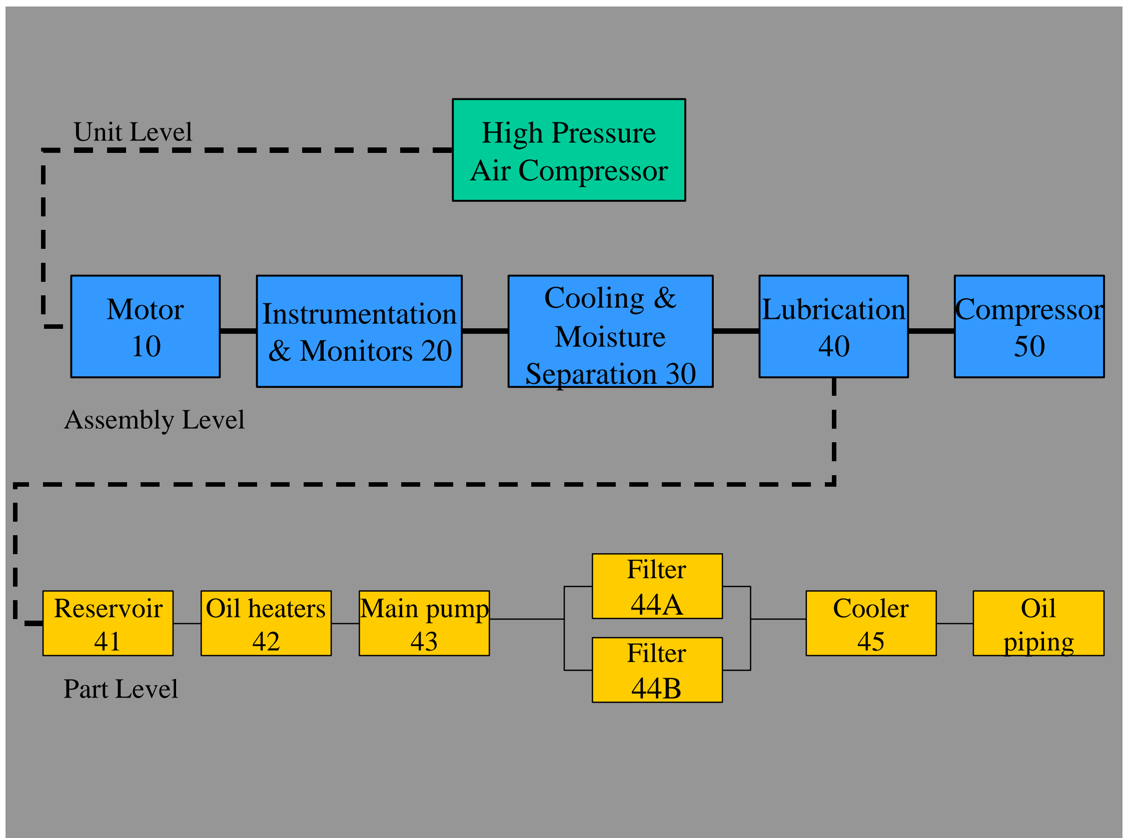


Figure 5. Example of a Reliability Block Diagram
[MIL-STD-1629, 1984]

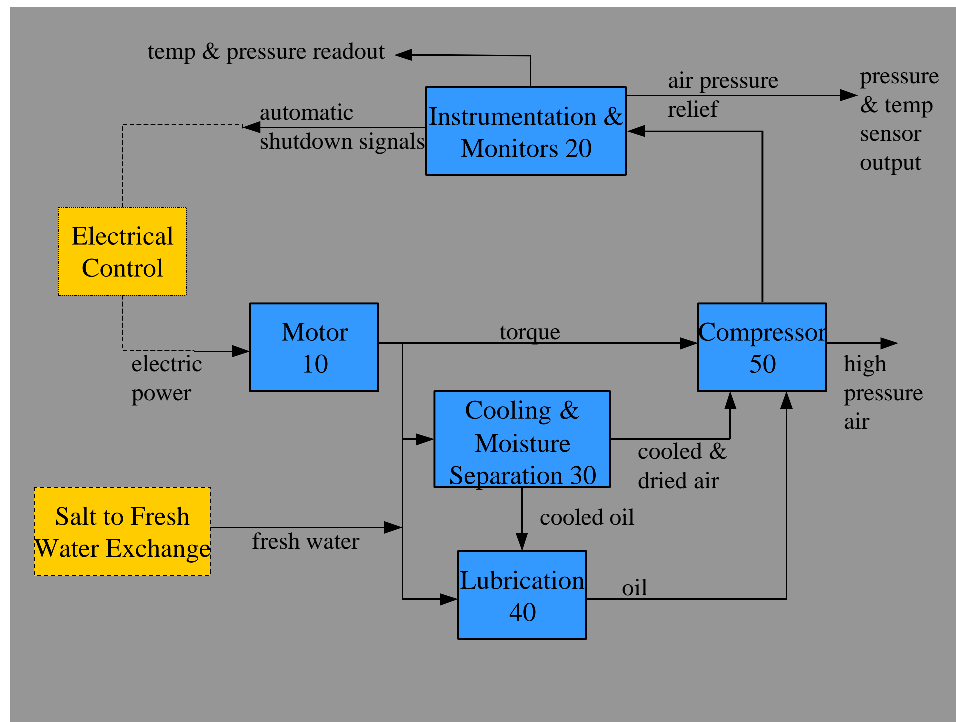


Figure 6. Example of a Functional Block Diagram
[MIL-STD-1629, 1984]

The component, or hardware approach, lists individual components or parts and analyzes their possible failure modes. The component approach is generally a 'bottoms-up' analysis. The smallest or lowest level of analysis is evaluated first and then the results are combined to form assemblies and so forth.

The functional approach recognizes that every item is designed to perform a number of functions that can be classified as outputs. The outputs are listed and their failure modes analyzed. The functional approach is usually a 'top-bottom' analysis. The highest functions are analyzed and then broken into their sub-functions. For complex systems, a combination of hardware and functional approaches may be used. [MIL-STD-1629, 1984]

Functional failure analysis is an area of growing interest because the loss of function can be correlated with failure. [Carson, 1998] This concept is corroborated by the fact that BIT tests on components that do not contribute to system functionality do not increase the BIT Fault detection percentage. This type of information is often documented in a FMECA (Failure Modes,

Effects and Criticality Analysis) process that not only defines failure modes but also defines their relationship to local and higher system functionality. [Fenton, 1996]

3.1.3 System Mapping – Functions to Components

Reliability numbers are related to components. A functional analysis must map the system function through the design layers to the components. Thus only the components actually contributed to the system performance are included in the reliability analysis. System functions can be mapped to the physical domain (design parameters) which can then be mapped to components. [Trewn, 2000] Table 5 shows the abstraction layers of the functional decomposition.

Design Layer	Abstraction	Level	Design Structure
A	Highest	Functional Requirements	Functions
B		Design Specifications	Design Parameters
C	Lowest	System Model	Components

Table 4. Functional Decomposition
[Trewn, 2000]

Through the functional decomposition, components are mapped to each function. The functional requirements (FR) are mapped to the physical domain (design parameters – DP). The DPs can then be mapped to individual components. (Reference Figure 7) Through this mapping, a binary matrix can be created showing the relationship between the functions and the components. For each function, a one is entered in the column of the component that helps to satisfy the function. A sample of this matrix, called D, is shown Figure 8 below:

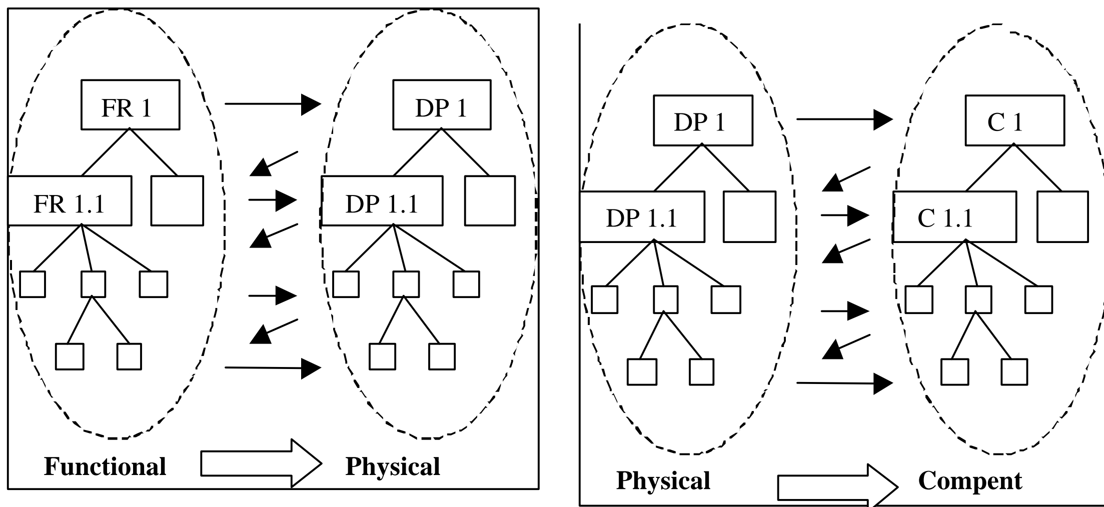


Figure 7. Functional to Physical to Component Mapping
[Chism, 2002]

	C1	C2	C3
FR1	1		
FR2	1	1	
FR3	1		1

Figure 8 Example of a Binary Matrix Relationship Between Functions and Components
[Chism, 2002]

3.1.4 Failure Types in Functional Reliability Analyses

Functional reliability is best used for BIT analysis. Functional analysis is defined as the likelihood of successfully providing necessary functions that a system or a component is intended to deliver. The components and/or parts that provide the functionality are analyzed and grouped together in one reliability number. This prevents reliability or failure rates being inflated due to testing components that would cause no detriment to the system with a failure.

There are many cases where several design parameters (DPs) can be satisfied by a single component. There are also cases where a single design parameter can only be satisfied by using several components. Therefore, the structure of relationships among components is often different than that of design parameters. However, the component/subsystem structure is often

the final form of engineering design. From a reliability perspective, it is the component structure that determines the reliability of the designed system. Thus mapping the component reliability to the function structure can help in determining functional reliability of the system.

The concept of failure in functional reliability analyses is expanded from a simple failure to multiple types of failures:

- Hard failure – complete failure of a function
- Soft failure – performance degradation in delivering functions
- Dependent failure – failure of sub-systems due to either the hard failure of the sub-system itself or the performance degradation of other subsystems. Since the dependent failure may involve failures of several subsystems, its impact on overall system performance may also be greater than the impact due to a single component failure. [Trewn, 2000]

Soft failures are exhibited when the system is stressed and hard failures occur when the system is pushed beyond an operational limit (reference Figure 9). Electronic equipment is usually designed to specifications, which include the range, or limits of environmental and operating stresses, such as temperature, humidity, and vibration. This range is called the specification limit. The stress margin, which is designed into the equipment so that the equipment will function correctly beyond the specification limit, is called the operational limit. Outside the operational limit, the equipment may show hard or soft failures.

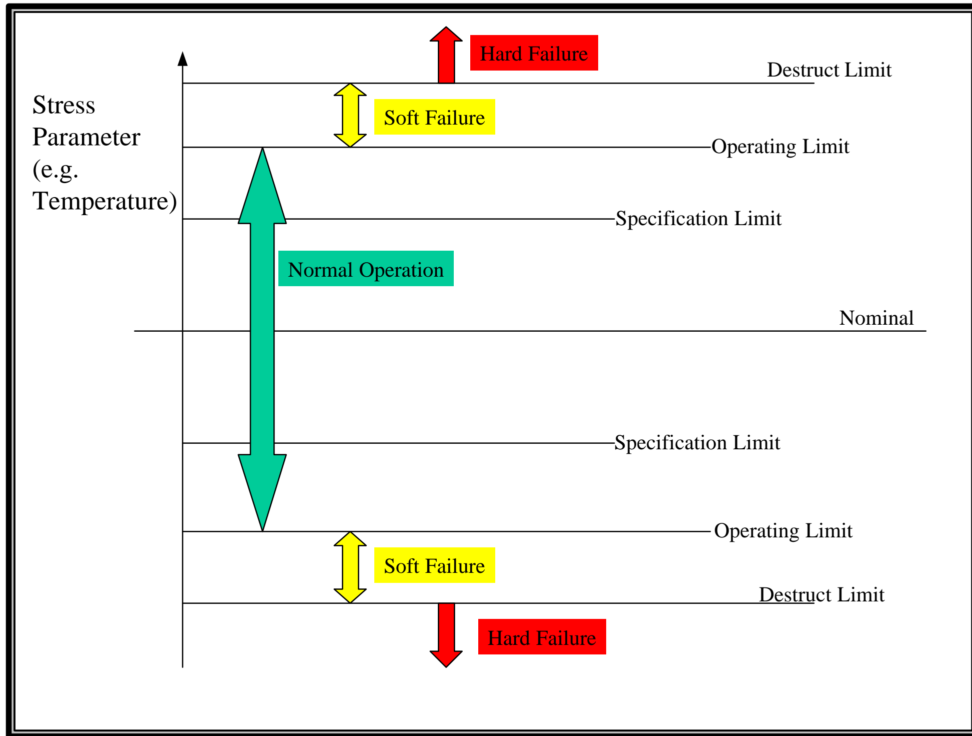


Figure 9. Soft and Hard Failures of Electronic Components [Pecht, 2001]

Functional analyses utilizing hard, soft and dependent failures require slightly modified reliability equations. The reliability of a system is defined as:

$$R_s = \prod_{i=1}^{m} P(Fn_i) \tag{1}$$

Where $P(Fn_i)$ is the probability that the function Fn_i is successfully delivered, and

$$P(Fn_i) = \prod_{k=1}^{n} (1 - p_k)^{d_{ik}} \tag{2}$$

Where d^{ik} is the entry of matrix D (as defined in 3.1.3) in the ith row and kth column and p_k is the failure probability of component k.

When a failure of a component will the cause the failure of other components, it is said the failure is a dependent failure. Trewn and Yang [2000] state that, in a binary failure model, dependent failure may be modeled by:

Let $P_{k|j}$ be the probability of failure of component k given the failure of component j, where $k \neq j$ and $p_{k|k} = 1$, then:

$$P_k = \sum_{j=1}^{n} [p_{k|j} \times p_j] \qquad (3)$$

Thus, system reliability can be expressed in terms of functions decomposed so that hard and soft failures are represented by the correct component selection. Furthermore, after the functional decomposition, a matrix can be created representing the mapping of functions to components to be used in reliability calculations. And dependent failures can be taken into account with the same equations.

3.1.5 Failure Rate versus Failure Probabilities

There are two measurements of fault detection. The first, which calculates failure probabilities, is the qualitative method. This method is more general. It usually specifies the classes of failures that are detectable, and may include failure detection percentages for different classes of failures. The second method, quantitative, is more explicit and calculates failure rates. It is the probability that a failure (any failure) is detected. The failure rate can be determined from the general failure rate specifications by using the average of the failure rates for all possible classes of failures, weighted by the probability of occurrence for each fault class. Thus the failure rate is more difficult to obtain, since the relative probabilities are implementation-dependent and may not be known. [Siewiorek, 1982]

In BIT analysis calculations, the failure rate of the component is used as a weighting factor in the fault isolation calculations. Fault isolation calculations, which are weighted by component failure rate, are more meaningful because the resulting testability analysis reflects the distribution of failure rates across the item. If parts with high failure rates can be isolated to the single part, it is more significant than parts with low failure rates that can be isolated to a single part. If many high failure rate parts are in a large ambiguity group, it is worse than if many low failure rate parts are in a large ambiguity group. [Giordono, 2001]

In many instances, simplifying assumptions are employed for the possible failure modes and probabilities. For these reasons, the qualitative method is generally used and, when possible, the quantitative method. [Siewiorek, 1982]

3.1.5.1 Qualitative Approach

The qualitative approach determines the probability of occurrence of a failure. This approach is the simplest and is used when no failure rate data is available. Failures are grouped into classes based on their probability of occurrence. Failures in a class are ranked among each other and assigned probabilities within the range of the class. Table 6 identifies the classes for the qualitative approach.

Class	Probability of Occurrence	Description
Class A	Frequent	A high probability of occurrence during the item operating time interval. High probability may be defined as a single failure mode probability greater than 0.20 of the overall probability of failure during the item operating time interval.
Class B	Reasonably Probable	A moderate probability of occurrence during the item operating time interval. Moderate probability may be defined as a single failure mode probability which is more than 0.10 but less than 0.20 of the overall probability of failure during the item operating time interval.
Class C	Occasional	An occasional probability of occurrence during the item operating time interval. Occasional probability may be defined as a single failure mode probability which is more than 0.01 but less than 0.10 of the overall probability of failure during the item operating time interval.
Class D	Remote	An unlikely probability of occurrence during the item operating time interval. Remote probability may be defined as a single failure mode probability which is more than 0.001 but less than 0.01 of the overall probability of failure during the item operating time interval.
Class E	Extremely Unlikely	A failure whose probability of occurrence is essentially zero during the item operating time interval. Extremely unlikely may be defined as a single failure mode probability which is less than 0.001 of the overall probability of failure during the item operating time interval.

Table 5. Qualitative Approach Classes
[MIL-STD-2165, 1993]

3.1.5.2 Quantitative Approach

The quantitative approach calculates failure rates in terms of failures per million hours (F x 10^{-6}). When reliability data is available, it is used. If not, failure rates for standard components can be obtained from MIL-HDBK-217, put title here. BIT detection and fault isolation percentages can be based on component, part, or functional failure rates. The quantitative approach using failure rates is preferred if data is available.

The critical goal in the quantitative approach is to obtain the appropriate failure rate for the component based on the probability of occurrence. Failure rates for electronic components vary with time as shown in Figure 10. The time dependent failure rate is called a hazard function, denoted as z(t). For electronic components on the normal-life portion of the bathtub curve, the failure rate is assumed to be constant. This means the exponential hazard function is applicable:

$$Z(t) = \lambda \tag{4}$$

For the periods of infant mortality and component wear-out, the Weibull hazard function is often used:

$$Z(t) = \alpha\lambda(\lambda t)^{\alpha-1} \tag{5}$$

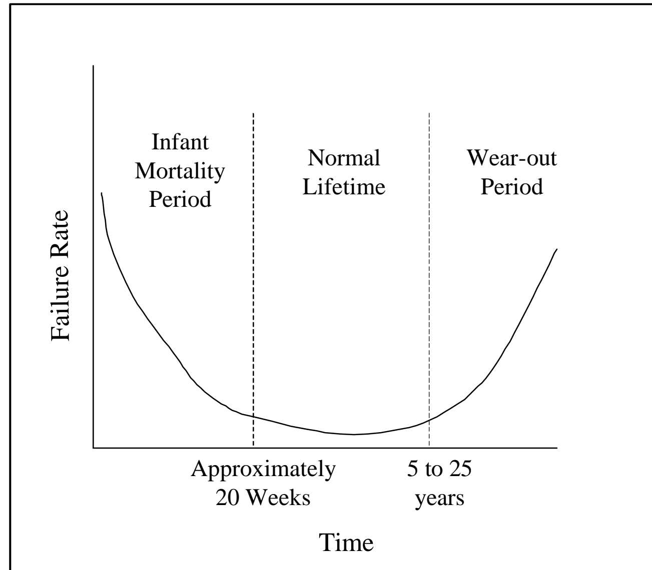


Figure 10. Bathtub Curve Depicting Component Failure Rate as a Function of Time [Siewiorek, 1982]

For BIT analysis, it is always assumed the components are on the normal life portion of the curve.

If there are no redundant components of functionality and component failures are statistically independent, the failure rate of a system or function is the sum of the failure rates of the individual components.

$$\lambda_{Fn} = \sum_{i=1}^{n} \lambda_{Ci} \qquad (6)$$

Where Fn is the function and there are n independent components, C_i, that satisfy the function. The components are said to be in series.

If all components must fail in order to cause a loss of functionality, the failure rate of the function is the product of the failure rates of the individual components.

$$\lambda_{Fn} = \prod_{i=1}^{m} \lambda_{Ci} \qquad (7)$$

Where Fn is the function and there are m parallel components, C_i, that satisfy the function.

A combination of series and parallel would result in sums of products and individual components.

$$\lambda_{Fn} = \sum_{i=1}^{n} \lambda_{Ci} + \prod_{i=1}^{m} \lambda_{Ci} + \prod_{i=1}^{x} \lambda_{Ci} + \prod_{i=1}^{y} \lambda_{Ci} + \ldots \qquad (8)$$

Where Fn is the function, there are n independent components C_i, m parallel components C_i, x parallel components C_i, y parallel components C_i, and so on.

3.1.6 FMEA/FMECA

FMEA (Failures Modes and Effects Analysis) is used to ascertain information necessary for general reliability factors which relate to fault detection and isolation. Engineering schematics, reliability and test data are used in the implementation of a FMEA.

In a FMEA, system, subsystem, LRU and part (whichever is appropriate to the BIT analysis level of concern) failure modes are established first. All significant failure modes must be identified, although those that have no effect on system operation or a very small probability of occurrence may be disregarded (ONLY if the failure will not create a hazardous or mission critical condition).

Next the failure effects are established for each failure mode. A failure effect is defined as a loss or degradation or change of a function or output due to a failure. Effects are described in terms of the manner in which item signals or outputs are displayed operationally or provided to another item. A systematic review of the functional block diagram or schematic can be used to generate a list of effects by examining the following:

- External item outputs – signals provided to other items
- Item outputs – signals output to operators
- Status and monitor panels – these often display important internal item signals
- Other performance monitoring information

Because of the many and varied skills required to determine failure modes, effects and corrective action, etc. the FMEA requires inputs from many disciplines. It is relatively unimportant which engineering group is selected by the contractor to make the analysis as long as cognizant design engineers play a major part. What is important is the critical examination of the results by all disciplines which could utilize the knowledge brought forth by the analysis.

The depth and scope of the FMEA is dependent on the BIT requirements levied on the program and the complexity of the system. A relatively simple system which has 5 or fewer LRUs will require only a small scope FMEA. A larger, more complex system with 10 or more LRUs will require a larger scope FMEA. The depth of the FMEA should be to the removable subunit of the system. [MIL-STD-470B, 1989]

A FMECA is generated by performing a criticality analysis and combining it with the FMEA. The FMECA approach can be component, functional or a combination of the two. [MIL-STD-1629, 1984] Reference Appendix IV for sample FMEA templates.

The criticality analysis is performed in two steps. The first step is to classify the severity of the failure. There are four categories of severity classifications as shown below:

Category I	Catastrophic	A failure which may cause death or weapon system loss
Category II	Critical	A failure which may cause severe injury, major property damage, or major system damage which will result in mission loss
Category III	Marginal	A failure which may cause minor injury, minor property damage, or minor system damage which will result in delay or loss of availability or mission degradation.
Category IV	Minor	A failure not serious enough to cause injury, property damage, or system damage, but which will result in unscheduled maintenance or repair

The second step is to determine the probability of occurrence. The levels, A through E, correspond to the classes identified in the qualitative approach section 3.1.5.1. The combination of the probability of occurrence and the severity classification give the criticality number, C_r. Figure 11 shows an example of a criticality matrix. Reference Appendix IV for sample Criticality Analysis templates.

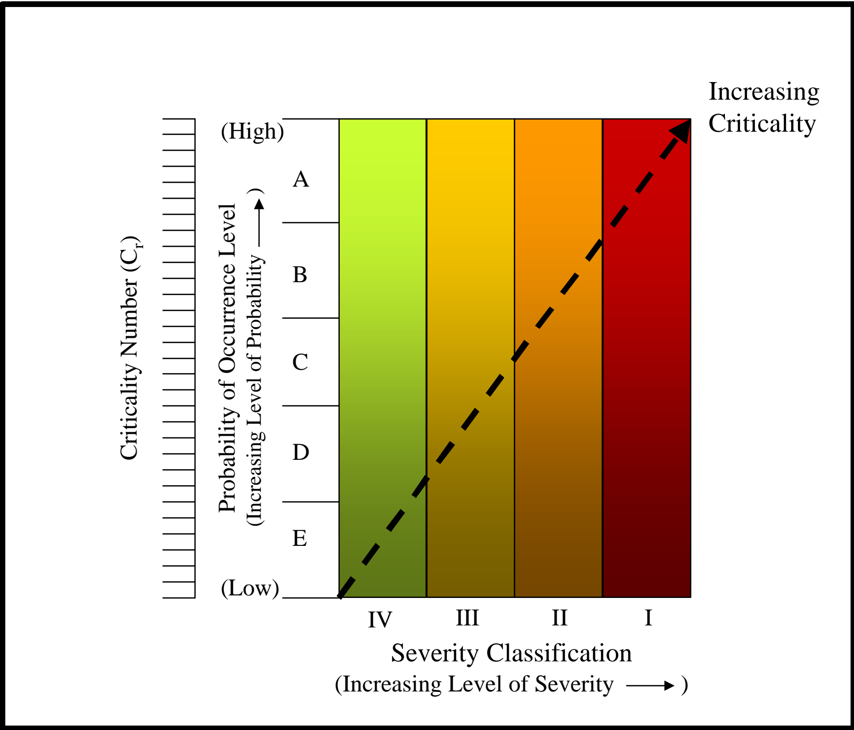


Figure 11. Criticality Matrix
[MIL-STD-1629, 1984]

3.2 BIT Requirement Calculations

The primary equations needed to verify BIT requirements are fault detection percentages, fault isolation percentages and false alarm rates. Equations for each of these requirements areas are defined in the following sections.

3.2.1 Fault Detection Calculation

The fault detection percentage is calculated as all *detected* failure mode rates divided by the total failure mode rate for the target system or functionality. If all test detected failure modes have a combined failure rate of 0.50 and the target system has a failure rate of 0.60, total detection is calculated as 0.50/0.60 = 83%.

$$\frac{\sum_{i=1}^{n} 1DFn_i}{\sum_{i=1}^{n} 1Fn_i} \qquad (9)$$

When there is a total of i functions, Fn_i, in the system and DFn_i is the failure rate of each function based on the detected failures in the function.

3.2.2 Fault Isolation Calculation

The BIT requirement for fault isolation can be verified through analysis of common fault signatures. Resultant ambiguity groups are summarized based on common fault signatures, with the probability of each fault signature calculated from the summation of failure mode rates that cause the signature. The percent isolation to maximum group sizes is then calculated from the resultant distributions.

The implemented fault isolation algorithm of the system is put in a table and mapped to the possible BIT failure combinations. The FMEA is used to determine cases where fault isolation would be incorrect. In the absence of a FMEA, failure rates can be used to determine probability for the proper isolation. The fault isolation percentage is calculated by adding failure rates of incorrect fault isolations and dividing by the total failures detected by BIT. [Fenton, 1996] Reference Table 6.

Fail Mode	**Fault Signature**	**Causing LRU**	**Failure Rate**
X	101101001	LRU A	0.25
Y	101101001	LRU B	0.33
Z	101101001	LRU C	0.77
Total			**1.35**

Table 6. Common Fault Signature Groupings
[Fenton, 1996]

In Table 6, the system fault isolation algorithm fault isolates to LRU C for the given fault signature. There are two other possible causing LRUs, but the failure rate is lower for each cause. The percent fault isolation is the sum of the failure rates isolating to all LRUs other than LRU C divided by the sum of the failures rates of all detectable causes. 0.25 + 0.33/1.35 = 43%. (Assuming these were the only tests in a system.)

$$\frac{\sum_{i=1}^{m} 1IFn_i}{\sum_{i=1}^{x} 1DFn_i} \qquad (10)$$

Where there are x detectable function failures, DFn_i, in the system and m incorrectly isolated failures, IFn_i, in the system.

3.2.3 False Alarm Rate Calculation

A critical drawback for BIT is false alarms. False alarms give an indication the system is failing when, in truth, the system is operating correctly under its defined parameters. False alarms can be caused by improper test thresholds, testing components or functionality not used by the system, or improper logic of BIT tests. Intermittent failures and CNDs (Cannot Duplicate failures) are not considered false alarms. [Gao, 2001]

False alarms are identified through examining the logic and thresholds for the implemented BIT tests and the test and field data available for the existing system. Any instance where a BIT failure is indicated without a true system failure is considered a false alarm. All false alarms should be able to be corrected by modifying a BIT test. Ideally a system's false alarm rate should be zero.

The false alarm rate is calculated by dividing the sum of the failure rates of the components that caused the incorrect BIT failure by the sum of the failure rates of all detectable failures.

$$\frac{\sum_{i=1}^{m} \lambda FFn_i}{\sum_{i=1}^{x} \lambda DFn_i} \qquad (11)$$

Where there are x detectable function failures, DFn_i, in the system and m incorrectly 'failed' failures, FFn_i, in the system.

3.3 Types of Data Needed

This section describes the type of data needed from an existing system in order to execute the BIT analysis process. Data is needed to support the system decomposition analysis from system functions, to physical decomposition to the individual components.

It is critical to base analysis on accurate and timely data, not on wishes, guesses, or experience. Dr E. Deming stated data volume has nothing to do with the accuracy of judgement. Data without context or incorrect data are not only invalid but sometimes harmful as well. It is necessary to know the nature of that data and that proper data be picked as well. [Walton, 1986]

3.3.1 System Functional Data

System functional data needed for the BIT analysis includes descriptions of the systems functions. Functional block diagrams or descriptions and functional requirements are important documents. If the system has a performance specification, system functions are often defined as performance requirements.

3.3.2 Physical Decomposition Data

Physical decomposition data needed for the BIT analysis can include physical block diagrams, traceability of physical requirements to functional requirements, parts lists, schematics, board level specifications and interface control documents.

3.3.3 Component Data

Component data critical to the BIT analysis process includes failure rate data, component specifications, test and field data and software code. The determination of possible and probable failure modes requires an analysis of reliability data on the component selected to perform each of the system internal functions. It is desirable to use reliability data resulting from reliability tests performed under the identical conditions of use. When such data are not available, reliability data from MIL-HDBK-217 or from operational experience and tests performed under similar use conditions on items similar to those in the systems should be used. [MIL-STD-1629, 1984]

3.3.4 Existing Reliability Analysis Data

Existing data from previous BIT analyses or reliability activities are useful in the BIT analysis process. Current BIT test logic, including when they tests are performed (IBIT, PBIT, CBIT), a description of the test, applicable thresholds and any false alarm data are vital to the final analysis. The system FMEA and/or FMECA and previous BIT requirement verification information are also invaluable.

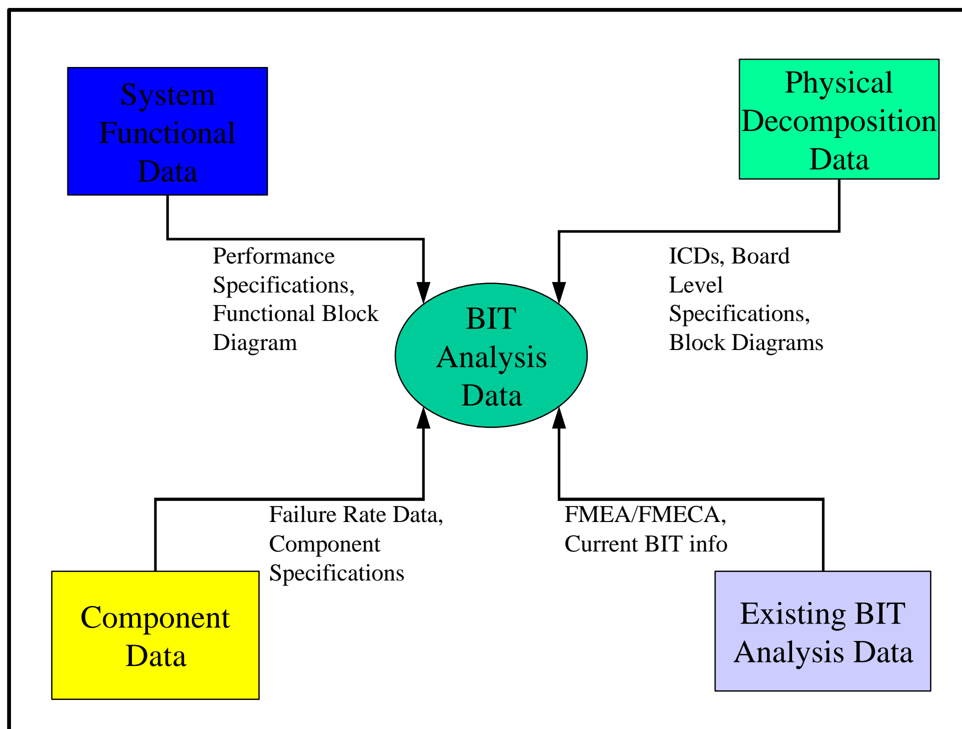


Figure 12. Data Types Needed for BIT Analysis

3.4 Data Alternatives

If the necessary data for the BIT analysis process is not readily available, there are other ways to retrieve the data from the system. [Spencer, 1985] To perform the analysis correctly, it is required to gather enough information to establish system functionality, identify the physical decomposition and the mapping to the components. It is also necessary to know the BIT tests that are currently implemented and the fault isolation algorithms, if any.

Data can be retrieved from the system through the following methods:

- Visual observation and manual recording – This includes running the system in normal environments, observing the system in an operating and non-operating state, disassembly of the system, and recording data.
- Automated test data and event recording – Automated test data can be retrieved from a system running in a normal environment or a simulated environment. In system parameters can be recorded and specific data needs researched.
- Verbal reporting – Cognizant program engineers can be interviewed for system knowledge and pertinent data. This is especially helpful in assessing failure modes.
- Written documentation - including functional diagrams, system schematics, equipment packaging, form, fit and function, examination of technical orders, content of preliminary design reviews (PDR) and critical design reviews (CDR) [MIL-STD-470B, 1989]
- Fault injection – A last resort, failures can be inserted into the system to determine the exact system behavior under the circumstances.

Some data retrieval methods can be very costly and time consuming. The order to be followed when trying to retrieve needed data is the following three steps:

Step 1	Review and Examination	Review written documentation, interview responsible engineers, examine system, use system in normal operating environment
Step 2	Test Using Simulation	Run system in a simulated environment and gather data through automated test data, observation and manual recording
Step 3	Test Case or Scenario Techniques	Perform specific tests to obtain exact results through specific test cases or fault injection

3.5 Data Processing and Analysis

Data processing and analysis is the second major sub-process in the proposed BIT Analysis process. This section covers how to analyze the data gathered in the previous section to get accurate results to the questions of BIT detection percentages, false alarm rates, and fault isolation percentages.

After system data has been gathered, the process to analyze the data begins. The first step in the process is to complete a FMEA and Criticality Analysis. The resulting FMECA contains all the information needed to perform the calculations described in Section 3.2.

3.5.1 Complete the FMEA

Several ground rules apply before starting the FMEA analysis process [Raytheon5, 2002]:

- Only one failure should be considered at a time. The effects of two or more detectable failures at once do not need to be considered.
- If a failure is undetectable, the failure must be considered in conjunction with other failures.
- Potential Category I (Catastrophic) failures should be documented in a fault tree format so the probability of occurrence can be easily determined and communicated.

The following steps should be followed to complete the analysis to create a FMEA from the gathered system data:

1. Define System to be Analyzed — Prepare a complete system definition including identification of internal and interface functions, expected performance at all indenture levels, system restraints, and failure definitions. Functional narratives of the system can be created including description of each mission in terms of functions which identify tasks to be performed for each mission, mission phase and operational mode. Narratives should describe the environmental profiles, expected mission times and equipment utilization, and the functions and outputs of each item.

2. Construct Block Diagrams	Functional and reliability block diagrams which illustrate the operation, interrelationships, and interdependencies of functional entities should be obtained or constructed for each item configuration involved in the system's use. All system interfaces should be indicated.
3. Define Failure Modes	Identify all potential item and interface failure modes and define their effect on the immediate function or item, on the system, and on the mission to be performed.
4. Define Failure Effects	Evaluate each failure in terms of the worst potential consequence that may result and assign a severity classification category. Consider the consequence of each failure mode on item operation, next higher assembly operation and total system operation when assessing severity.
5. Identify Means of Failure Detection	Identify failure detection methods and compensating provisions for each failure mode. This can be through BIT tests, operator detection through audible or visual warning signals or automatic sensing devices.
6. Identify Compensating Provisions	Identify corrective design or other actions required to eliminate the failure or control the risk. Compensating provisions are design characteristics or operator actions that negate or reduce the effects of a failure. These may include redundant systems, alternative operating modes, and safety devices. These could affect BIT fault detection percentages.
7. Identify Fault Isolation	Identify the effects of corrective actions or other system attributes, such as requirements for logistics support. BIT Fault isolation is assessed to support logistics and repair efforts. Fault isolation can include both BIT and troubleshooting flows.

8. Compile Results — Document the analysis and summarize the problems which could not be corrected by design and identify the special controls which are necessary to reduce failure risk. Failure rates, fault isolation information, BIT information and functional-component mapping will be used for BIT analysis calculations.

3.5.2 Perform the Criticality Analysis

A criticality analysis (CA) should be performed to accompany the FMEA. The results of the criticality analysis should be reviewed and items with high criticality numbers, Cr, should be highlighted. Fault detection on these relatively high risk, high probability items should be verified. Items with high Cr values are also called *mission critical* items. Some BIT requirements may require a certain fault detection percentage on mission critical parameters.

The steps for performing a CA are defined in Section 3.1. The CA combined with the FMEA creates the FMECA.

3.5.3 Perform BIT Requirement Calculations

The completed FMECA contains all the information required to perform the BIT Fault detection calculations. The BIT requirements may specify BIT fault detection at certain levels. The indenture levels or groupings provided in the FMECA can be used to separate the proper functional-component mapping to use in the calculation. Equation 9 should be used to calculate BIT fault detection

The BIT detection information in the FMECA, along with the failure rates and failure modes, are used to create the fault signature table used in calculating BIT Fault isolation percentages. The BIT fault isolation algorithm should be documented for verification purposes. Once the table has been created and failure modes with incorrect fault isolation identified, the fault isolation percentage can be calculated using equation 10.

The last calculation, BIT false alarm rate, is zero unless false alarms have been recorded or identified and data is available. If false alarms occur or are possible (through review of BIT logic and/or thresholds), the fault detection failure rates in the FMECA are used along with the

false alarm failure data to calculate the false alarm rate. The calculation is performed with equation 11.

3.6 Documentation of the Analysis - Reports

One of the most important aspects of the BIT analysis process is to communicate the results in a clear, concise manner and provide enough information to substantiate the results. The output can range from a paper specification to electronic spreadsheets, but it should contain the same information. It also covers the documentation of the analysis process for verification purposes. The following six items are the primary information requirements for the documentation of the analysis:

1) A list of assumptions, waivers or special analyses should begin the report to identify the parameters of the analysis.
2) Summary of Results – a table comparing the BIT requirement values with the calculated values from the analysis
3) Summary of BIT implementation – the current list of BIT tests (in each mode – PBIT, IBIT, CBIT, etc.) and the BIT fault isolation logic used in the analysis
4) Calculations Used in Analysis – information showing the failure rates, with the mapping from functions to components and the tests that cover them. Reference Table 7 for an example format. The fault signature table should also be included in this collection of data.
5) FMEA/FMECA – should be included in the report or referenced if it is a released document.
6) False Alarm Rate Data - Test data or BIT logic/threshold analysis supporting false alarm rate calculations should be included in the report.

Function	Physical	Component	Failure Rate	Where Tested					Detection Percentage
				IBIT	PBIT	CBIT	Operator	Not	
A									83%
	Board 1								56%
		A	9.37	X	X				90%
		B	12.4				X		64%
		C	.56					X	0%
	Board 2								
		A	9.37	X	X				90%
		D	23.0	X	X	X			85%

Table 7. Sample Data Table for Report

3.7 Tools

The task of performing a BIT requirements analysis is very large. Engineers have been utilizing spreadsheets and other computer based tools to aid in the process for years. Many companies have implemented internal procedures and computer programs to help facilitate the process and simplify the effort. There are several commercial products available now to simplify the process of gathering and processing data, as well as reporting the results.

Specialized tools are not necessary to complete the BIT Analysis Process. They serve as an aid to help organize information, prevent operator errors and standardize procedures. It is important to remember that tools require specific inputs and the data gathered must follow the correct format. In some cases it may be more work to enter system data into the tool than to perform the analysis by hand in a spreadsheet. In other cases, the tool may prove to be invaluable. The output from the tool may or may not include the data required on the program. It is important to consider tool cost, system requirements, the size and scope of the project and the capabilities of the tool before investing.

Basic overviews of three commercially available tools are covered in the following sections.

3.7.1 Advanced Specialty Engineering Networked Toolkit (ASENT)

Produced by Raytheon, ASENT is a 32-bit application that handles large amounts of data with high speed. It is designed for large programs and has many functions other than BIT analysis. It can be installed on individual computers or hosted on a server for network use across an entire enterprise. In the network environment, ASENT allows multiple users in geographically dispersed locations to concurrently access the company's entire reliability and maintenance data.

ASENT has a modular toolkit structure so it is not necessary to purchase all the modules. ASENT is a high-end tool with a large cost, so individual modules can result in a cost savings. However, for large programs, a tool like ASENT is well worth the cost to ensure the program data is analyzed and archived properly. One of the modules includes Raytheon's newest feature, the Board Analysis Manager, which leverages the pioneering work in *physics of failure-based analysis*. The modules include:

- Reliability Manager
- FMECA Manager
- Board Analysis Manager
- Maintainability Manager
- Parts Library
- Data Interfaces Utilities

The FMECA Manager is a good tool to enter the functional and component information. The Board Analysis Manager contains the programs to compute the numbers required for verify BIT requirements. ASENT does not replace the engineering effort required to determine functional design and component reliability. It does provide a way to organize the information and automatically generate BIT requirements numbers.

ASENT is the main reliability program in use on programs at Raytheon North Texas. It is supported by the company's Information Technology (IT) department. A tool that is authorized and supported by a company should be utilized if available. For existing programs, the overhead of entering the information in the tool (if it is not already using the tool) can be very large, but may not be much larger than the effort required to enter the data in a spreadsheet or an

older, outdated tool. Assessments should be made on each program based on the tools currently in use, if any, and the tools available in the company.

3.7.2 Computer Aided Reliability and Maintainability Applications (CARMA) Toolkit

Also produced by Raytheon, and replaced by ASENT, is the Computer Aided Reliability and Maintainability Applications (CARMA) toolkit. CARMA consists of over 250 separate programs (interfaces, analysis tools, reports) that are integrated in such a way that they work off of a central set of database tables. This facilitates the ability to pass analysis results from one tool to another, and cuts down on the amount of time spent re-entering data. The diagram below shows the Architecture of the toolkit and provides a summary of the analysis functions.

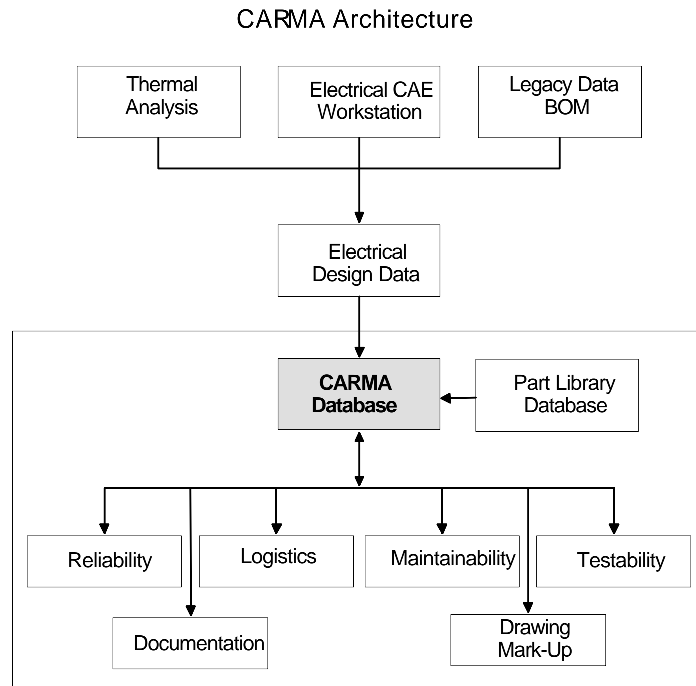


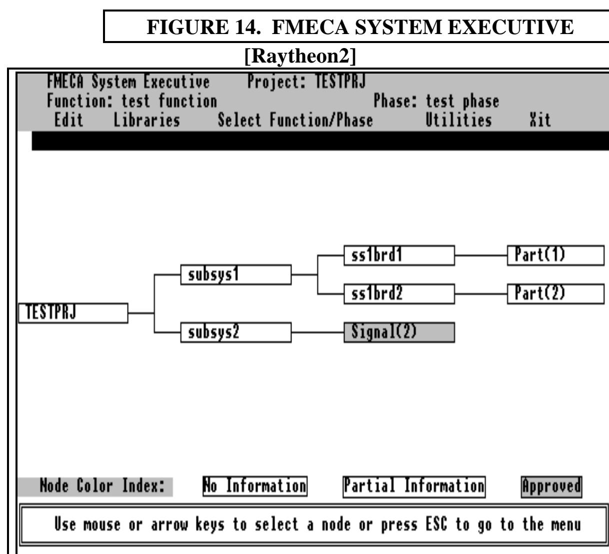
Figure 13. CARMA Architecture
[Raytheon2, 1999]

The primary CARMA modules are listed below with their associated functionality:

RELIABILITY	Allocations, Predictions, Op: 217C through FN2, Bellcore, NonOp: RAC Reliability, Toolkit: Commercial, Practices Edition, Failure Rate Calculation, Failure Rate Survey, Reliability Modeling, Thermal Analysis, Power Distribution Display, PTH Reliability, Vibration Analysis, Solder Joint Fatigue, Derating Analysis, Growth Test planning and Analysis, Program Checklist
LOGISTICS	LSAR Data Export to 1388-2A and 2B Formats
MAINTAINABILITY	Predictions, Design Checklist, Program Checklist, Reliability Centered Maintenance, Test/BIT Coverage Analysis
TESTABILITY	MIL-STD-2165 Checklist, Testability Analysis
DOCUMENTATION	Custom Reports, Customer Quality Reports
SAFETY	Hazard Analysis, Hazard Reporting
DRAWING MARK-UP	Electronic Drawing Annotation

CARMA contains a FMECA tool. The FMECA tool in CARMA uses a graphical interface to help the user remain well oriented and move easily within the project (Figure 14).

FIGURE 14. FMECA SYSTEM EXECUTIVE
[Raytheon2]



This tool supports the accomplishment of Task 101 (FMEA), Task 102 (Criticality Analysis) and Task 103 (FMECA - Maintainability Analysis) of MIL-STD-1629A. The analyses can be based on the hardware structure, the functional structure or a hybrid of the two approaches. By the use of libraries, product tree inheritance and inheritance of failure prediction information from other CARMA tools, the FMECA software reduces labor and opportunities for human error.

Failure Rates for parts on the FMECA product tree can be updated with current CARMA tree values using the Failure Rate update tool. In addition to the classical Reliability and Maintainability information, Testability, Safety and RCM analysis information may be entered as part of the FMECA Analysis.

The Maintainability tool uses the information entered in the FMECA tool to determine BIT requirements numbers. The BIT detection percentages can be based on board components, by functionality, or by groups of tests. The percentages can also be summarized on multiple system levels.

The CARMA tool has a smaller scope than ASENT. It focuses more on areas pertinent to BIT requirements verification. In that sense, it is a better tool. A person performing only BIT requirements verification would use less of the full ASENT package than of the full CARMA package. The primary disadvantage of CARMA is that it is no longer supported by Raytheon, having been replaced by ASENT.

3.7.3 RAM Commander

RAM Commander™ , a product of Reliass (Reliability and Safety Solutions), is a Reliability and Maintainability software tool for reliability professionals and design engineers. This software program covers the entire scope of engineering tasks related to reliability of electronic, electro-mechanical, and mechanical systems. [Reliass, 2000]

Modules cover:

- Reliability Prediction
- Reliability Block Diagram
- Maintainability
- Spare Parts Analysis & Optimization
- Derating Guidelines and Reports
- FMECA Analysis
- Testability Analysis
- Process & design FMEA

The Failure Mode, Effects and Criticality Analysis (FMECA) module is a natural continuation, and in many cases inseparable part, of the Reliability Analysis. Previously a separate software package, FMECA is now a fully-integrated RAM Commander module. The FMECA module uses a product tree previously created by the user for the reliability analysis purposes.

The FMECA module supports the following types of reports: FMEA- MIL-STD-1629, FMECA, Criticality Analysis - MIL-STD-1629, End Effects Criticality, Numbers, Criticality Matrix, Fault Tree, NHE Critcality, Test methods, BIT/Detection Coverage, Fault Isolation Resolution. (Reference Figure 15)

Figure 15. RAM Commander FMECA Screen

The Testability Analysis sub-module of the FMECA module is intended for in-depth Testability analysis. The main characteristics of Testability - BIT/Detection Coverage and Fault Isolation Resolution - can be calculated for each maintenance level (Organizational, Intermediate, Depot) and for specific detection methods (BIT, BITE, external test equipment, etc.). Test

method efficiency and indication are defined for each test method or a group of test methods. Testability analysis is widely used for the development of necessary supporting documentation: maintenance manuals, troubleshooting procedures and inspection requirements.

The RAM Commander tool, like CARMA, is a smaller scope than ASENT and is more appropriate for a program performing a BIT requirements verification only. All of the tools listed in this section are designed to satisfy more program requirements than BIT requirements verification. They all provide the FMECA analysis which is required for the BIT analysis and compute BIT requirements verification numbers.

The choice of which tool to use, if any, should be based on the price, the availability of resources to run the tool, the format of the data that must be entered to calculate BIT requirements numbers, the size of the program being evaluated and the types of reports generated.

CHAPTER 4
COMPARISONS

4.1 Final Process

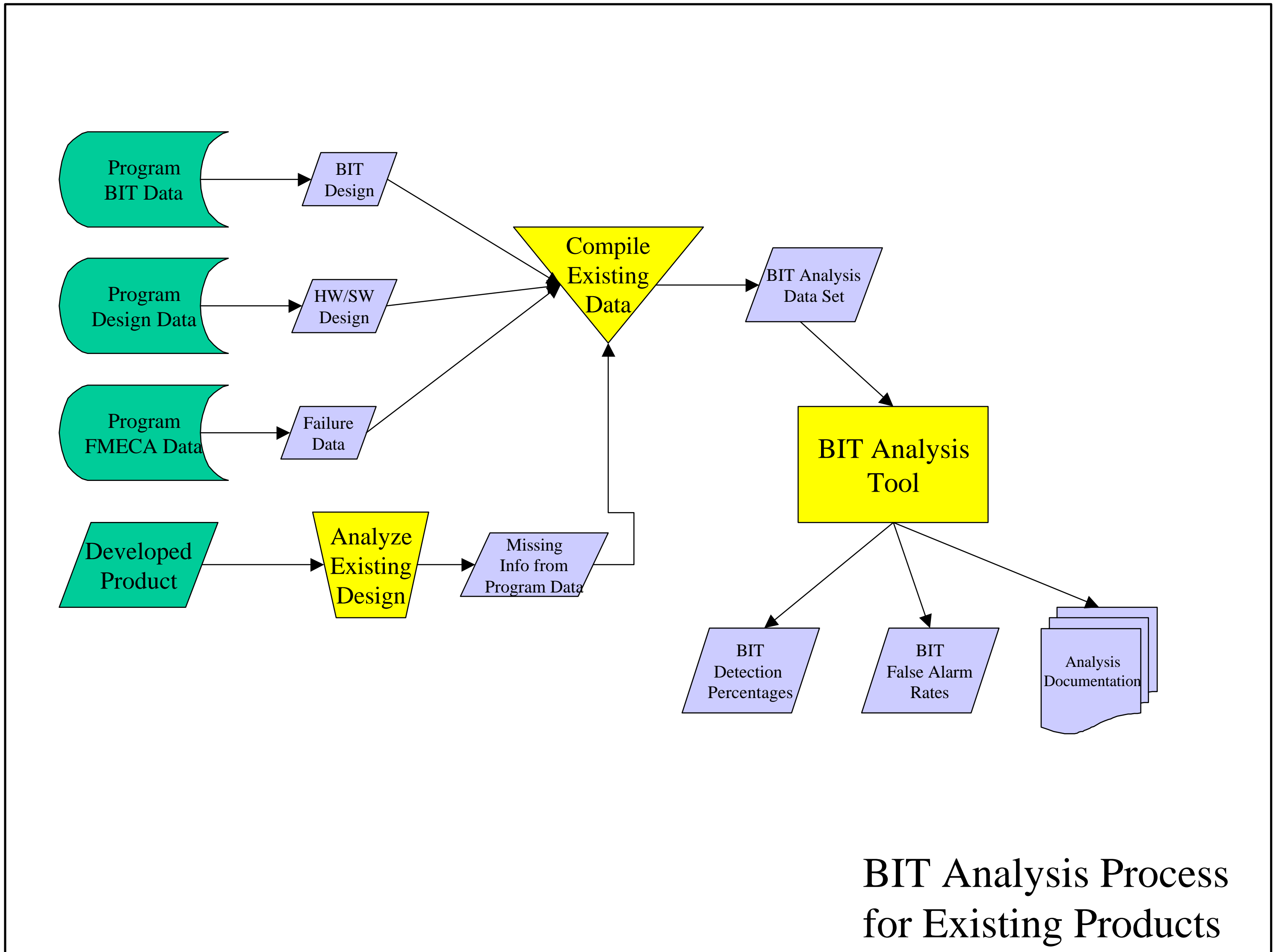
Based on the analysis in Chapter 3, the finalized BIT Analysis Process follows a functional decomposition of the system down to the appropriate level for the BIT requirements on the program (to the lowest level LRU). The process consists of three main activities related to verifying BIT requirements: data gathering, analysis, and reporting of results.

The BIT Analysis Process outlines a data gathering subprocess that contains techniques based on first examining existing documentation and analyses. If information is lacking from the program data set, then the subprocess identifies methods of retrieving information from the system through other documents, the system itself, cognizant engineers, test and simulation. This is the largest portion of the process, with the most tasks. The data gathering subprocess includes the FMEA and CA to identify the mapping from system functions to the proper level of components. The subprocess identifies each type of data to be gathered and the methods by which to gather them.

The BIT Analysis Process identifies an analysis subprocess that focuses on the calculation of the numbers needed to verify BIT requirements. The correct levels for BIT detection – based on the program requirements – are defined in this subprocess. The reliability data obtained in the data gathering subprocess is used in the calculations necessary to verify the BIT requirements. This subprocess also identifies the type of information needed in a final report verifying the requirements and provides sample formats. The tasks in this subprocess can be aided by using a tool, or specialized reliability software program. The BIT Analysis Process is designed to be used with a tool or without.

The BIT Analysis Process consists of flowcharts, instruction sheets, and templates. The flowcharts show the order of tasks to be completed. Figure 16 shows the top level flowchart for the BIT Analysis Process. The instruction sheets explain how to perform detailed tasks and templates are given as an aid. In the event a tool is used, most templates will not be necessary. The full set of flowcharts and instruction sheets are contained in Appendix III. The templates are contained in Appendix IV.

Figure 16. Top Level BIT Analysis Process Flowchart



4.2 Application of Process to HTI

A trial run of the BIT Analysis Process was performed on the HTI program. The BIT requirements for the HTI program are defined in Chapter 2. The main objectives of applying the BIT Analysis Process are:

- Assess the applicability and correctness of the BIT Analysis Process
- Verify BIT Requirements on the HTI Program
- Compare the BIT Analysis Results to the previous analysis
- Perform an analysis on the new hardware and software configurations of the HTI program

4.2.1 Assumptions

The following assumptions and definitions were defined before starting the BIT analysis on the HTI program:

Scope	The scope of the analysis was limited to assess one hardware configuration only and version 7.0 of the software. The HTI B-KIT has two current hardware configurations: LRIP and SMT. LRIP is used for this analysis. Version 7.0 is the latest release of B-KIT software.
Mission Critical Failure	Defined as any failure which causes the B-KIT to be unable to provide a FLIR image.
Catastrophic Failure	It was determined there are no B-KIT failures that could lead to a catastrophic failure.

Critical Failure	B-KIT failures that contribute to a condition where gunner boresight is incorrect or the image is frozen (leading the operator to believe there is a good image) or the loss of an image are considered critical.
Failures Present at Power Up	Detection percentage for failures present at power up should be based on failures detected by PBIT + SBIT + CBIT and the host platform.
Failures Occurring during Operation	Detection percentage for failures occurring during operation should be based on failures detected by CBIT + host platform + operator.
Indenture Level	The assembly levels analyzed for the B-KIT BIT analysis are down to the level specified in Table 8.
Failure Rates	Failure rates are based on reliability data gathered through Reliability tests and field data unless otherwise stated. Failure rates may contain data from both hardware configurations in some cases.
Fault Isolation	The B-KIT fault isolation algorithm used in the analysis will be the Generic Platform fault isolation algorithm shown in Appendix V. [U0080137(m), 2002]

The HTI B-Kit configuration has not changed since the original analysis even though some boards have been redesigned. The SMT hardware fits into the same assembly configuration as the LRIP and EMD hardware. The SMT and LRIP hardware configurations are the only ones supported. The EMD hardware configuration has been discontinued. Therefore the BIT analysis

will not consider the EMD hardware. The assembly levels analyzed for the new HTI BIT analysis are the same as the old analysis, and are shown in Table 8.

Table 8. Assembly Levels Analyzed

Assembly Description	Part Number
Sensor Unit	
Afocal Telescope Assembly	A3248010
Cooler Control CCA	A3246956
Scanner Control CCA	A3246939
Digitizer CCA	A3246943
Point of Load Regulator CCA	A3248085
Detector/Cooler Bench	A3247100
Imager Assembly	A3247020
Electronics Unit	A3248100
Video Processor CCA	A3243647
Interface Control CCA	A3248225
Video Converter CCA	A3248230
Power Converter #1 CCA	A3248170
Contrast Enhancement/Frame Int	A3246951
Power Converter #2 CCA	A3248270
Chassis	A3248210

4.3 Data Results

The data gathering subprocess of the BIT Analysis Process used on the HTI system was the largest effort in using the process. The task of obtaining information for the requirements is a multi-disciplinary activity but was manageable with the guidelines and templates. Some of the effort required finding data and entering it in the template. Some new analysis was also necessary. The functional decomposition had to be revisited. The SMT hardware provided additional functionality to the system beyond the capabilities of the LRIP hardware.

The process helped identify which information was useful and which was not. Data collected for HTI included a partially complete FMECA (very outdated), reliability numbers for the LRIP hardware configuration, and predicted reliability numbers for the SMT hardware configuration. The reliability numbers included failure rates for the assemblies listed in Table 8. Information on the BIT tests, their algorithms, functionality tested, and thresholds was available in an Algorithm Description Document (ADD). The BIT Fault Isolation algorithm for HTI was also contained in an ADD. System Failure data from the field and from production tests provided

false alarm information. Examples of key data gathered as part of this subprocess is included in Appendix V.

Analysis was necessary to complete the mapping of the system functions to the appropriate BIT tests and the hardware tested. Analysis was also necessary to determine the fault signatures for every failure to determine the fault isolation information. The analyses were easier to perform with the help of the instruction sheets and the templates, but still required an engineer familiar with the system to do the work. The task would be almost impossible for someone without system knowledge. The techniques identified to gather missing data from an existing system were also used in this phase. Additional program documents had to be reviewed, engineers questioned, and a few tests performed on the B-KIT.

At the end of the data gathering suprocess, an HTI data set was completed with enough information to proceed to the analysis subprocess.

4.4 Analysis of results

The data analysis portion of the BIT Analysis Process on the HTI program resulted in the actual numbers to verify the BIT requirements. The BIT detection percentages were obtained using the failure rate information and the functional analysis, which identified what was tested in each phase of BIT tests. There were 4 phases under consideration for BIT detection percentages and they included a combination of CBIT, PBIT, IBIT and operator or host platform detection activities. The four phases are identified in table 9. These phases match the BIT requirements defined in the HTI performance specification.

The BIT fault isolation was based on the combination of tests in IBIT as well as the operator and host platform isolation activities. The failure signature was mapped to the fault isolation implemented in software and identified in the HTI performance specification for the operator. The fault isolation percentage was calculated from the result of the mapping.

The HTI BIT false alarm rate was based entirely on field failure data and production test data. Failures flagged by BIT, but that resulted in no identifiable system failure and were not considered an intermittent failure (some based on instrumentation data) were considered false alarms.

The results of the B-KIT BIT analysis are included in Appendix V. Some failure rate data pertaining to sub assemblies made by a co-contractor was not available at the time of the analysis. A fully correct analysis will require recalculation when this data is available. Table 9 is a summary of the BIT requirements and the results of the analysis based on the data available. The results for the SMT hardware are predicted and are not part of this analysis (reference scope in Section 4.2). Appendix V also contains the data reports from the analysis process used to document the results of the analysis. All of the pertinent data is not included in the report to save space and to protect the proprietary nature of the system design.

	Detection % (+/- 5%)				**Fault Isolation % (+/- 5%)**
	CBIT + Host Platform Mission Critical Detection	**CBIT + Operator + Host Platform Mission Critical Detection**	**PBIT + SBIT + CBIT + Host Platform Mission Critical Detection**	**IBIT + Operator + Host Platform Detection**	**IBIT + Operator + Host Platform Isolation**
Current B-KIT Performance Spec	85%	90%	90%	90%	85%
Version 7.0 SMT Predicted Performance	87.6%	90.8%	92.3%	91.2%	90%
Version 7.0 LRIP Predicted Performance	87.6%	90.8%	92.3%	91.2%	89.6%

Table 9. HTI BIT Detection and Isolation Requirements from the New Analysis

4.5 Assessment of Process

The BIT Analysis Process was assessed by verifying the BIT requirements for the HTI program. Particular key areas were paid attention to when performing the process. A discussion of the assessment of the process is covered in the following paragraphs.

4.5.1 Ease of Use

The BIT Analysis process expedites a difficult task. The process itself is not overwhelming – it consists of a minimal number of flowcharts, templates and instructions sheets. BIT analysis, in general, is difficult and the instruction sheets help. However, someone other than a trained engineer would have a difficult time using the BIT analysis process. The combination of the instruction sheets and templates makes finding the correct data and using it appropriately much easier.

4.5.2 Time Savings

One of the key benefits of the BIT Analysis Process is time savings. The process provides generic, but explicit enough directions to guide an engineer through the task without wasting time trying to determine what data to use, how to perform the calculations or evaluate the system. A BIT analysis is a very time consuming task even with directions, but the process can ensure it is done properly, the first time, with adequate documentation. By specifying the exact type of data needed for the analysis, the process prevents wasted time on evaluating unnecessary information.

4.5.3 Tailorability

The BIT Analysis Process is designed to be generic. It can be tailored to work on any program. The analysis effort on the HTI program required very little tailoring of the templates and instructions for the process. More tailoring will be necessary for programs using an automated reliability tool. Tailoring will be difficult for anyone who does not understand the process. The BIT Analysis Process does not contain any information or direction that is program specific but it will naturally lend itself to some programs easier than others.

4.5.4 Tool Usage

The BIT Analysis Process is designed to be generic enough to be used with an automated reliability software tool. Some aspects of the process are unnecessary if a tool is used, such as the actual calculations of the BIT requirements numbers. The majority of the work in BIT analysis, gathering the appropriate information, is a part of the process and is needed whether a tool is used or not. No tool was used in the HTI BIT analysis for this project. Raytheon uses ASENT but the HTI program has not utilized the tool yet. Time considerations led to the use of spreadsheets rather than a tool.

4.5.5 Verifies BIT Requirements

The BIT Analysis Process produces BIT detection percentages, false alarm rates and BIT isolation percentages based on fact and calculated with proper algorithms. The results are documented. The process satisfies requirement verification through analysis. The results of the process contain enough information to comprise a well-documented analysis and full proof of the numbers calculated. Note: It is possible the results of the BIT Analysis Process will show the system does not meet the BIT requirements. If so, the reason for the failure will be known through the analysis.

4.5.6 Clarity of Directions

The BIT Analysis Process provides instruction sheets to perform many of the tasks of the process. The instruction sheets were clear in most cases. Some of the more difficult tasks (failure signature table, functionality decomposition to components, etc) are not easy to understand unless the engineer is well trained. Future revisions of the process could use instructions sheets geared towards a more junior engineer.

4.5.7 Cost Savings

The BIT Analysis process, as a side effect of proving the fault isolation and false alarm requirements, identifies gaps in fault isolation logic or BIT tests that should be investigated due to intermittent failures. These intermittent failures can be cleared and do not relate to a system failure – they are often called CNDs (cannot duplicate).

The BIT requirements verification resulting from the BIT analysis process does not affect procurement cost, but it will result in an immediate reduction in the operating and support (O&S) cost of the B-Kit. Even though the B-Kit BIT false alarm rate on the vehicles is very low, the CND rate for B-Kit items returned to the factory from all integration/maintenance activities (direct support, vehicle integration, A-Kit integration, etc.) has been running between 25 and 30 percent for the past 2 ½ years.

The impact from these CNDs is well known. They cost the Government in labor and spares and cost the contractor in fault isolation and test time. It may be safe to assume a cost to the Government of $1000 per CND and another $1000 in contractor cost for administration and test. Assuming a return rate of 400 B-Kit items per year, a potential reduction of the CND rate from the current 30 percent to 10 percent as a result of BIT enhancements will reduce the number of CND returns from 120 to 40 per year – or 80 fewer CNDs per year. At a cost of $2000 per CND, this equates to $160,000 cost avoidance per year, or $3.2 million over the anticipated 20-year life cycle of the B-Kit.

In addition to potential O&S cost savings, the BIT Analysis Process provides cost savings in the BIT analysis process itself. Verifying customer requirements is necessary on a program. Time and resources are allocated to the activity. By having a process that expedites the activity

and provides correct results the first time, the time and resources used can be reduced – resulting in a cost savings for the program.

CHAPTER 5
CONCLUSIONS

5.1 Benefits of the Process

A standardized BIT Analysis process is beneficial for any program with BIT requirements. The final BIT Analysis process can provide these benefits for most programs. Key benefits include time and cost savings as well as providing to the customer proof that requirements have been met.

Key benefits:

- Cost Savings
- Time Savings
- Specific Directions – data needed, equations to use, data to report
- Easily tailorable
- Generic
- Facilitates tool usage
- Verifies BIT Requirements
- Supports multiple data types
- Covers existing, rather than developing, systems

5.2 Applicability to any program

'*Never tell people how to do things. Tell them* <u>*what*</u> *to do and let them surprise you with their ingenuity.*' George S. Patton, <u>War as I Knew It</u>, 1947.

The BIT Analysis Process is specifically designed to be applicable to any existing program. The flowcharts, instruction sheets and templates are not program specific. Most programs lend well to functional decomposition and have program data, in some form, available. The process, while simplifying the BIT analysis task by providing direction, does not change the requirement for a well-trained reliability, systems or transdisciplinary engineer to perform the process.

Tailoring may be is required to apply the process to a program. Based on the nature of the program and what data is available, certain sub-processes, specifically the data gathering sub-process, may be modified to increase effectiveness. The BIT Analysis Process is designed to not

only provide a generic outline of how to perform the task of BIT requirements verification, but also provide enough information to educate the responsible engineer on how the requirements are verified.

The BIT Analysis Process is designed for existing programs. It can also be used for new development, but some tailoring will be necessary because all information will not be available at the start of the analysis. There are additional factors that should be considered for a new development program so the BIT Analysis Process is not the best solution to verifying BIT requirements for new development. There are processes and tools that are designed specifically for new programs.

5.3 Discussion of Systems Engineering/Transdisciplinary Process

'*If the study of all these sciences which we have enumerated, should ever bring us to their mutual association and relationship, and teach us the nature of the ties which bind them together, I believe that the diligent treatment of them will forward the objects which we have in view, and that the labor, which otherwise would be fruitless, will be well bestowed.*' Plato [Eiermann, 2001]

Plato recognized the advantages of tying multiple sciences together. A transdisciplinary engineer has more understanding and broad capabilities than a specialist. Technology and science today requires understanding of all sciences to be fruitful. This project is a result of a transdisciplinary process. The BIT Analysis Process, the product of this project, is also a transdisciplinary process.

5.3.1 Application of Transdisciplinary Process to Project

The engineering process used to implement the BIT Analysis Process project was a transdisciplinary process. The project was designed using Raytheon's System Engineering process, Integrated Product Development System (IPDS). A master schedule and plan was developed to ensure all the necessary elements were planned for and the requirements for the project were well defined. Periodic reviews were facilitated through class assignments and through drafts of the final report. The actual analysis of the project used multiple decision making and evaluation tools. The project process was unique in the sense that the project was managed and executed by one person. All roles and phases of program development were executed in the scope of this project.

Axiomatic Design, specifically the independence axiom, played a large role in this project. Due to the large nature of the project and the duration of time necessary to complete the report, the work needed to be divided into segments for feasibility. Using axiomatic design principles, the subtasks were divided into sections that were independent. The independence ensured the sections could be worked on and completed without input from other sections or affecting the work completed in previous sections. For tasks that had some level of dependency, a schedule was established to ensure the appropriate information was available before starting a task. This prevented unnecessary changes, reworks or incorrect information from being used.

5.3.2 Transdisciplinary Process in the BIT Analysis Process

The BIT Analysis Process is a true Transdisciplinary process. The functional analyses, reliability equations and data reporting require many skills and tools to complete. The process is best performed by an engineer with experience in multiple disciplines. The mechanical, electrical and software design of a program must be considered to perform a BIT analysis. Reliability, safety, mission goals and operating scenarios must be considered in the analysis.

The independence axiom of Axiomatic Design is used in the assessment of the current system and the functional decomposition. By determined the dependency of functions (and later components) the reliability equations take into account this dependency and a more accurate answer can be reached to verify BIT requirements. Since the program is an existing program and may not have been developed with axiomatic design in process, the functions may not be independent or decoupled. Even so, knowing the dependencies helps to calculate accurate detection percentages, helps clarify fault isolation analyses such as the fault signature table and can provide clues to the reasons for false alarms.

The BIT Analysis Process is designed to be a part of an overall systems engineering or transdisciplinary process implemented on a program.

CHAPTER 6
REFERENCES

Carson, Ronald, 'BITE is not the Answer (But What is the Questions?)', IEEE Instrumentation and Measurement Magazine, 1998.

Chism, Davinia and Farmer, Frank, 'Axiomatic Design: Practical Application of the Independence Axiom', IDPT Conference Proceedings, 2002.

Clark, Major Arthur L., *Warrior's Wisdom: The Combat Guide to Corporate Life*, The Berkley Publishing Group, 1997.

Devlin, Keith, *Mathematics: The New Golden Age*, Columbia University Press, 1999.

Eiermann, Katharena, 'An Extensive Collection of Quotations by Philosophers and Psychologists', www.dividingline.com, 2001.

Ertas, Atila and Jones, Jesse, *The Engineering Design Process*, (Second Edition), John Wiley & Sons, Inc, 1996.

Fenton, Wesley, 'Testability Analysis Utilizing a Relational Database', IEEE Instrumentation and Measurement Magazine, 1996.

Gao, Robert, 'BIT for Intelligent System Design and Condition Monitoring', IEEE Instrumentation and Measurement, 2001.

Giordono Automation Corp., 'Diagnostic Profiler User's Guide: Testability Analysis', 2001. www.giordano.com

Goodman, D.M., 'History and State-of-the-Art in Automated Electronic Test and Checkout', Automation in Test Equipment, vol III, 1967.

Marick, Brian, *The Craft of Software Testing: Subsystem Testing*, Prentice Hall PTR, 1995.

MIL-HDBK-2165, *Testability Program for Systems and Equipments*, Department of Defense, 1995.

MIL-HDBK-217, *Reliability Prediction of Electronic Equipment*, Department of Defense, 1994.

MIL-PRF-A3207380, *Performance Specification for the Horizontal Technology Integration (HTI) NV-80 B-KIT*, USA Communications-Electronics Command, 1997.

MIL-STD-1629A, *Military Standard Procedures for Performing a Failure Mode, Effects and Criticality Analysis*, Department of Defense, 1984.

MIL-STD-2165, *Military Standard Testability Program for Electronic Systems and Equipments,* U.S. Department of Defense, 1993.

MIL-STD-470B, *Military Standard Maintainability Program for Systems and Equipment*, Department of the Air Force, 1989.

Nordbotten, Joan C., *The Analysis and Design of Computer-Based Information Systems*, Houghton Mifflin Company, 1985.

Pecht, Dube, Natishan, Williams, Banner, and Knowles, 'Evaluation of Built-In Test', IEEE Transactions on Aerospace and Electronic Systems, January 2001.

Petroski, Henry, *Design Paradigms: Case Histories of Error and Judgment in Engineering,* Cambridge University Press, 1998.

RASSP, 'Test Technology Overview: Module 43', RASSP Education and Facilitation Program, 1998. www.rassp.scra.org

[Raytheon1] Raytheon Systems, 'Built in Test Detection and Isolation Performance Analysis for the Second Generation FLIR Horizontal Technology Integration NV-80 B-KIT' (Contract DAAB07=94-C-J504), 1997.

[Raytheon2] Raytheon Systems, 'CARMA Core Toolkit User's Guide' (Version 4), 2002.

[Raytheon3] Raytheon Technical Services Company, 'Advanced Specialty Engineering Networked Toolkit (ASENT) User's Guide', 2001.

[Raytheon4] Raytheon Horizontal Technology Integration (HTI) Homepage, http://pub01.rsc.raytheon.com/hti/index.htm, 2002

[Raytheon5] Raytheon Integrated Product Development (IPDS) Process, Raytheon, 2002.

[Raytheon6] Raytheon Horizontal Technology Integration (HTI) BIT Analysis Failure Rate Summary, Raytheon, 1997.

Reliasis, 'Ram Commander User's Guide', 2000. www.reliability-analysis.co.uk

Sallade, Rex and Brown, Belinda, 'Consolidated Systems Engineering for Test', Raytheon Systems Company, 1999.

Siewiorek, Daniel, and Swarz, Robert, *The Theory and Practice of Reliable System Design*, Digital Press, 1982.

Spencer, Richard H., *Computer Usability Testing and Evaluation*, Prentice-Hall Inc, 1985.

Stenmetz, Michael, 'Built-in-Test Instrumentation and 21 Rules of Thumb', IEEE Instrumentation and Measurement Magazine, September 2002.

Trewn, Dr. Jayant and Yang, Dr. Kai, 'A Treatise on System Reliability and Design Complexity', Institute for Axiomatic Design, Proceedings of First International Conference on Axiomatic Design, 2000.

U0080137(M), HTI Second Generation FLIR Built In Test (BIT) Algorithm Description Document (ADD), Raytheon, 2002.

Walton, Mary, *The Deming Management Method*, The Berkley Publishing Group, 1986.

CHAPTER 7
APPENDIX I – NOMENCLATURE

2-D	2 dimensional
ADD	Algorithm Description Document
ATE	Automatic Test Equipment
BIT	Built-in-Test
BIST	Built-in-Self-Test
B-KIT	Common name for HTI SGFLIR
BOM	Bill of Material
CBIT	Continuous BIT
CCA	Circuit Card Assembly
CDR	Critical Design Review
CND	Cannot Duplicate
DP	Design Parameter
EMD	Engineering Manufacturing and Development
FMECA	Failures Modes, Effects and Criticality Analysis
FLIR	Forward Looking Infrared
FPA	Focal Plane Array
HTI	Horizontal Technology Integration
IBIT	Interruptible BIT or operator Initiated BIT
ICD	Interface Control Document
IPDS	Integrated Product Development System
IR	Infrared
IRS/DD	Interface Requirements Specification and Design Document
IT	Information Technology
LRIP	Low Rate Initial Production
LRU	Line Replaceable Unit
O&S	Operating and Support
PBIT	Periodic BIT
PDR	Preliminary Design Review

SGFLIR	Second Generation FLIR
SMT	Surface Mount Technology
SOW	Statement of Work

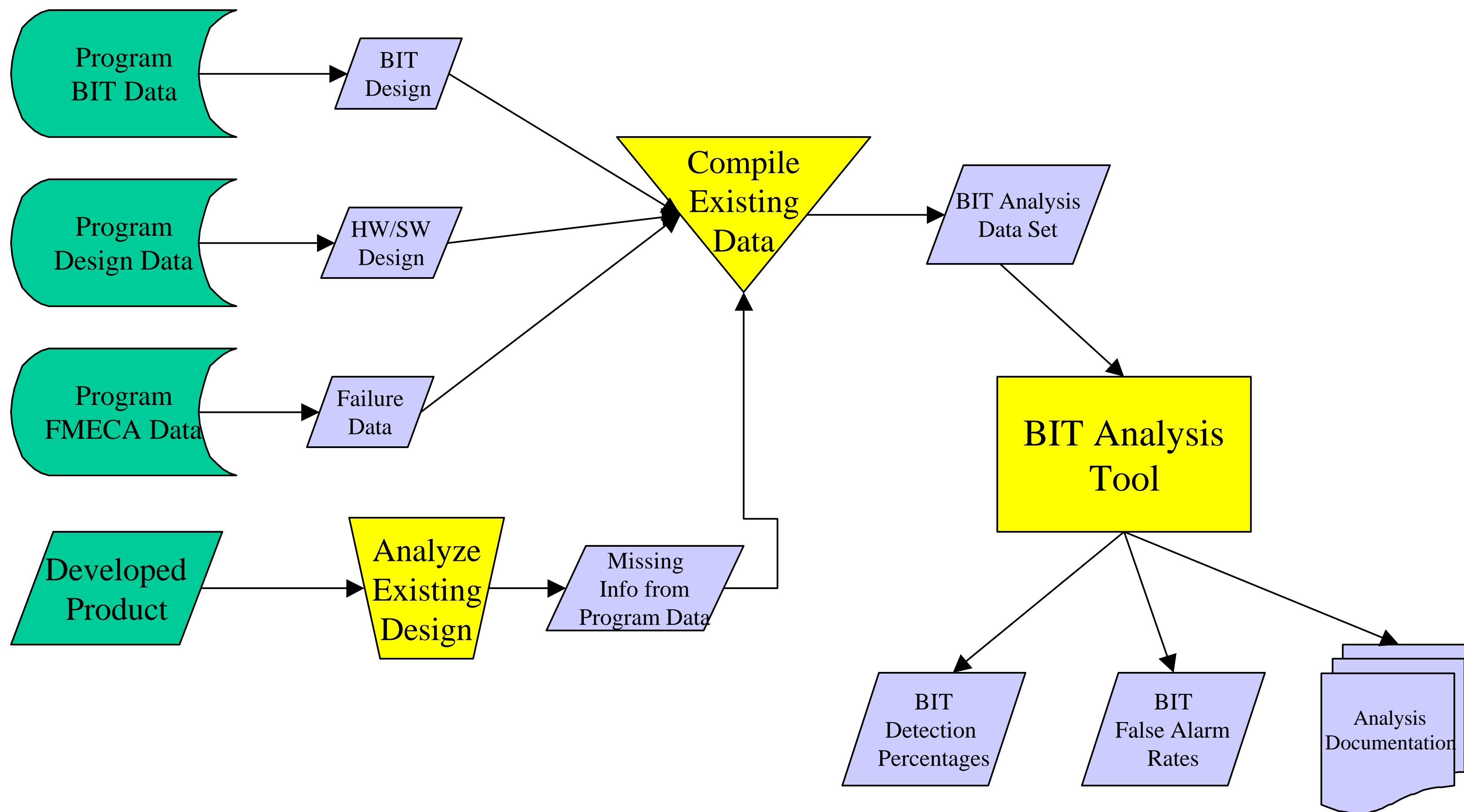
CHAPTER 8
APPENDIX II – HTI BIT ANALYSIS

HTI BIT Test List

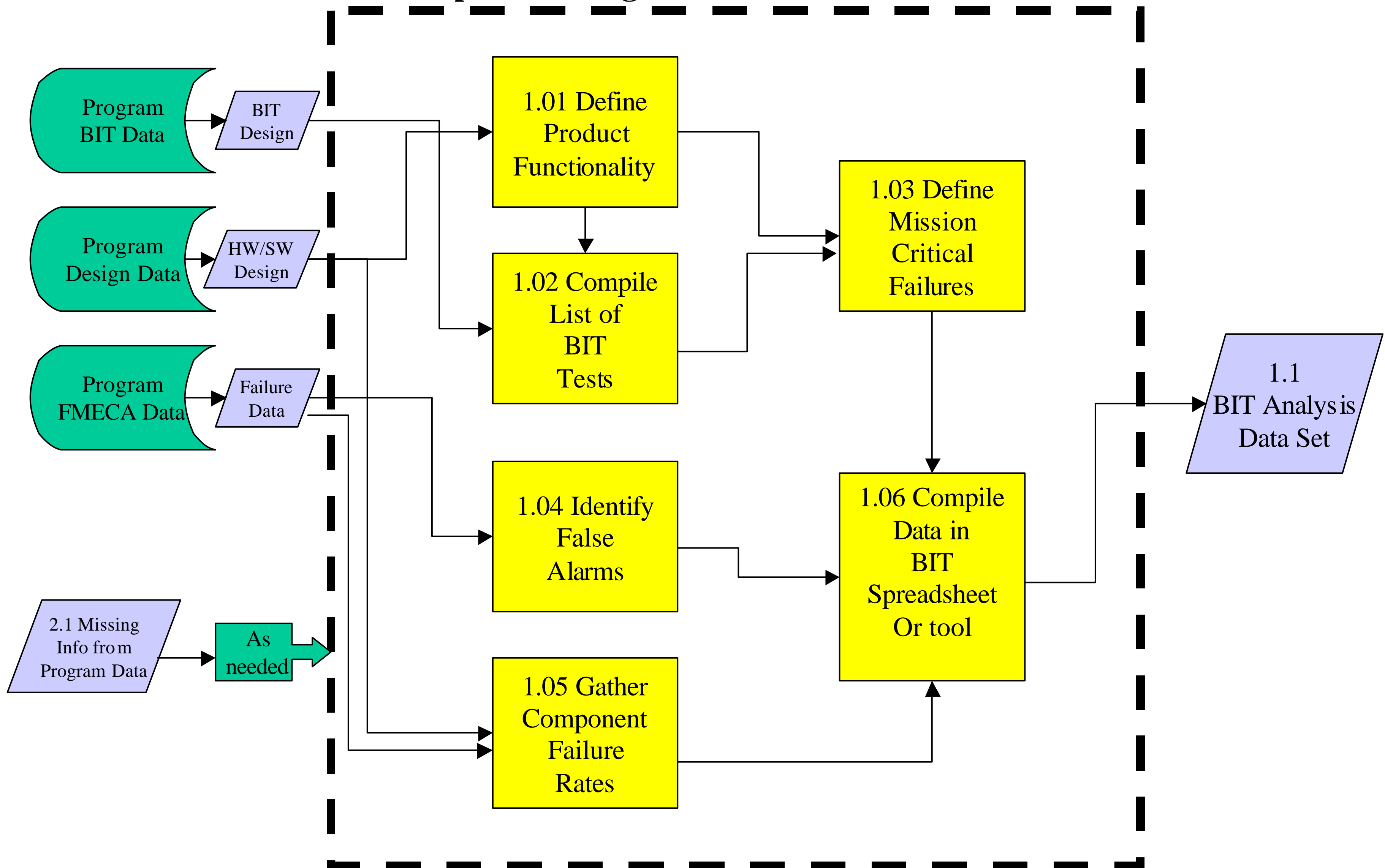
Test ID	Test	Test ID	Test
1	Cooler Compressor Temp Sensor	51	Digitizer Gain
2	Cooler Compressor Overheat	52	Digitizer Level
3	Scan Error	53	Reformatter Freeze Frame Output
4	Boresight Achieved	54	TRS 1 Response
5	Cooldown Monitor	55	TRS 2 Response
6	EU/SU Serial Link Timeout	56	TRS 1 Drive
7	FOV Position	57	TRS 2 Drive
8	TRS 1 Overheat	58	TRS 1 Temp Sensor
9	TRS 2 Overheat	59	TRS 2 Temp Sensor
10	Afocal Temp Sensor	60	SADA Bad Channels
11	VP Bad Timing	61	SU C40
12	Reformatter Freeze Frame Input	62	SU EEPROM Checksum
13	EU Box Temp Sensor	63	SU RAM
14	Platform ID	64	SADA EEPROM
15	Master C40	65	SU Fault Log EEPROM
16	Master EEPROM Checksum	66	SADA Serial I/O
17	Master RAM	67	POL N12A
18	EU/A-Kit UART	68	POL N5A
19	EU/SU UART	69	POL P12A
20	Deleted	70	POL P5A
21	EU Fault Log EEPROM	71	Cooler Input Power
22	EU/SU Serial Link Checksum	72	Digitizer Serial I/O
23	PS1 Overcurrent	73	POL P5B
24	PS1 Overvoltage	74	Digitizer Control Signature
25	PS1 Undervoltage	75	Scan Sync
26	PS2 Overvoltage	76	SADA BIST
27	PS2 Undervoltage	77	Scanner Command
28	EU Fan	78	Scanner Position Limits
29	SU Fan	79	Scanner Resolver Excitation
30	Reticle RAM	80	Scanner Resolver Feedback
31	Symbology RAM	81	Stationary Filter Wheel Voltage
32	Globalization/Polarity	82	Stationary FOV Current
33	2D Filter	83	Stationary FOV Voltage
34	Reformatter RAM	84	Aux PS Overvoltage
35	EU/A-Kit Serial Link Timeout	85	Aux PS Undervoltage
36	Slave C40	86	Focus Position Limits
37	Slave EEPROM Checksum	87	EU Overheat
38	Slave RAM	88	SU Overheat
39	Histogram/Normalization	89	Scanner Motor High
40	TRS Sums/VP Vertical Direction	90	Scanner Motor Low
41	VP 1 Control Signature	91	Scanner Current Feedback High
42	VP 2 Control Signature	92	Deleted
43	Filter Wheel Current	93	Scanner Current Feedback Zero
44	Moving Filter Wheel Voltage	94	Spare
45	Filter Wheel Position	95	Spare
46	Moving FOV Current	96	Spare
47	Moving FOV Voltage	97	Spare
48	Focus Position	98	Spare
49	Focus Position Sensor	99	Spare
50	Deleted	100	Spare

		Total Failure Rate	Mission Critical Failure Rate	Failure Rate as per 3.10.5	CBIT + Host Platform Mission Critical Detection	CBIT + Operator + Host Platform Mission Critical Detection	PBIT + SBIT + CBIT + Host Platform Mission Critical Detction	IBIT + Operator + Host Platform Detection (as per 3.10.5)	IBIT + Operator + Host Platform Isolation (as per 3.10.5)	IBIT + Operator + Host Platform MC Detection
EU										
VC CCA		56.8129	44.12168	55.4579	21.796	34.4338	27.3057	41.1001	24.3681	34.4338
%					49%	78%	62%	74%	44%	78%
Video Processor CCA		22.0133	21.1093	21.6633	3.773	7.737	17.2723	18.6883	18.0523	18.6883
%					18%	37%	82%	86%	83%	89%
Interface Control CCA		24.0201	21.0881	22.1833	4.5155	9.3553	19.4445	19.4445	19.3883	19.4445
%					21%	44%	92%	88%	87%	92%
Power Supply 1 (Converters 1 & 3)		33	25.891	33	25.891	25.891	25.891	25.891	24.5965	25.891
%					100%	100%	100%	78%	75%	100%
Power Supply 2 (Converter 2)		10.2	7.872	10.2	7.872	7.872	7.872	7.872	7.872	7.872
%					100%	100%	100%	77%	77%	100%
EU Chassis - Cooling Fan		15.72	15.72	15.72	15.72	15.72	15.72	15.72	15.72	15.72
%					100%	100%	100%	100%	100%	100%
EU Chassis - EMI PWB Assembly		7.1	4.733333	4.733333	4.7333	4.73333	4.73333	4.73333	4.73333	4.73333
%					100%	100%	100%	100%	100%	100%
EU Chassis - Fan Control Assembly		4.3403	4	4.3403	4	4	4	4	2	4
%					100%	100%	100%	92%	46%	100%
EU Chassis - Motherboard/Flex Harness, CEFI		22.3	22.3	22.3	22.3	22.3	22.3	22.3	16.725	22.3
%					100%	100%	100%	100%	75%	100%
EU Total		195.5066								
SU										
Scan Control CCA F/R		22.41	21.274	21.67325	12.761	12.7605	13.9065	14.3058	11.6803	13.9065
%					60%	60%	65%	66%	54%	65%
Digitizer CCA F/R		20.3107	20.3107	20.3107	16.751	16.7511	16.7511	16.5799	15.1318	16.5799
%					82%	82%	82%	82%	75%	82%
Cooler Control CCA F/R		10.143	7.5205	9.00025	5.2565	5.2565	5.2565	5.2565	5.2565	5.2565
%					70%	70%	70%	58%	58%	70%
POL CCA F/R		10.121	6.981	10.0515	0.5485	0.5485	0.5485	0.706	0.386	0.571
%					8%	8%	8%	7%	4%	8%
Receiver Assembly - Detector Cooler F/R		292	292	292	292	292	292	292	292	292
%					100%	100%	100%	100%	100%	100%
Receiver Assembly - Imager Assembly F/R		35.656	35.656	35.656	32.565	32.565	32.565	32.565	32.565	32.565
%					91%	91%	91%	91%	91%	91%
Afocal Assembly F/R		42.7623	42.7623	42.7623	39.762	42.7623	39.7623	42.7623	42.7623	42.7623
%					93%	100%	93%	100%	100%	100%
SU Total		433.403								
System Total		824.4162	593.3399	621.0521	510.24	534.686	545.329	563.925	533.237	556.724
					86.0%	90.1%	91.9%	90.8%	85.9%	93.8%

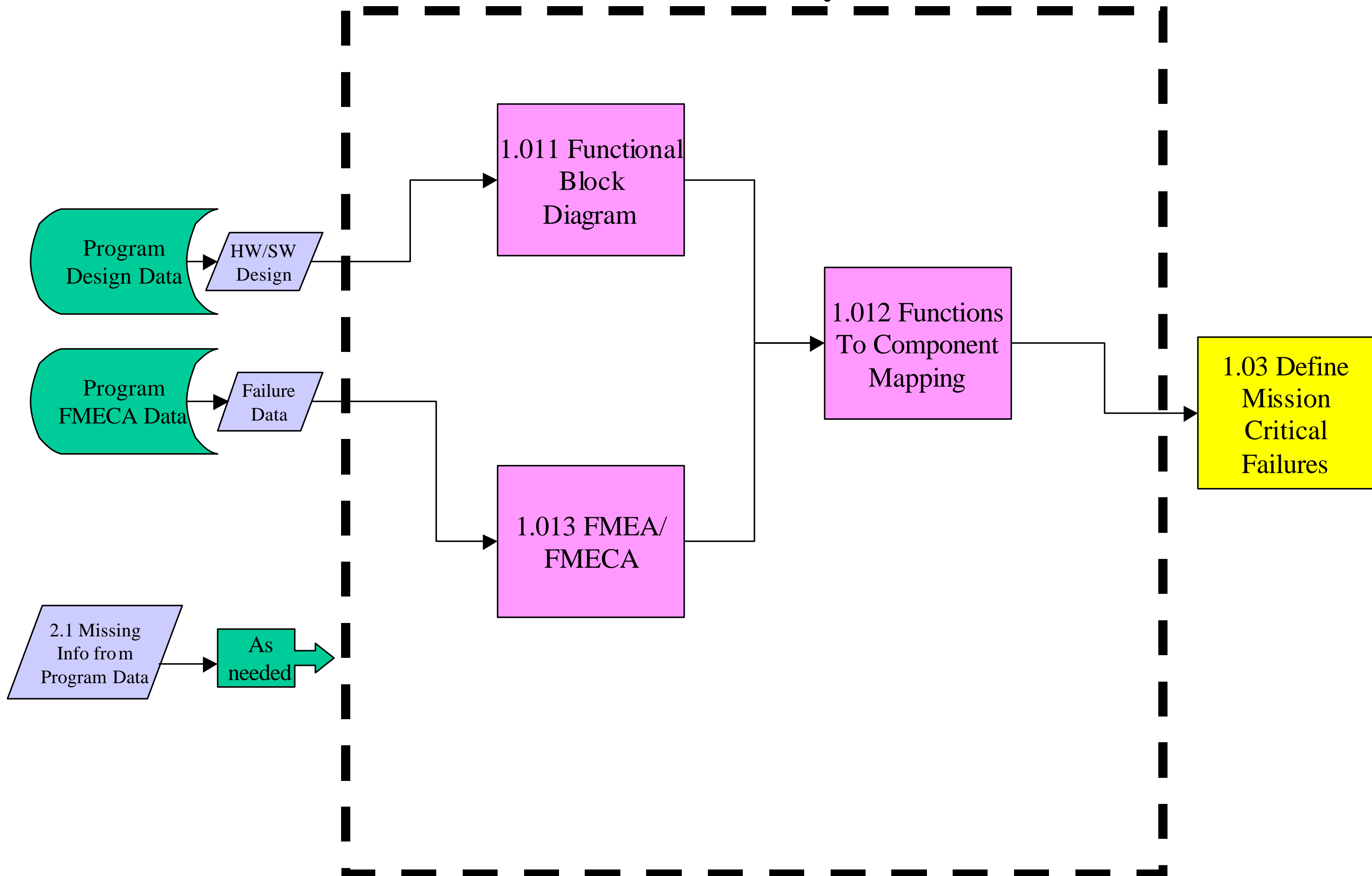
CHAPTER 9
APPENDIX III – PROCESS FLOWCHARTS AND INSTRUCTION SHEETS



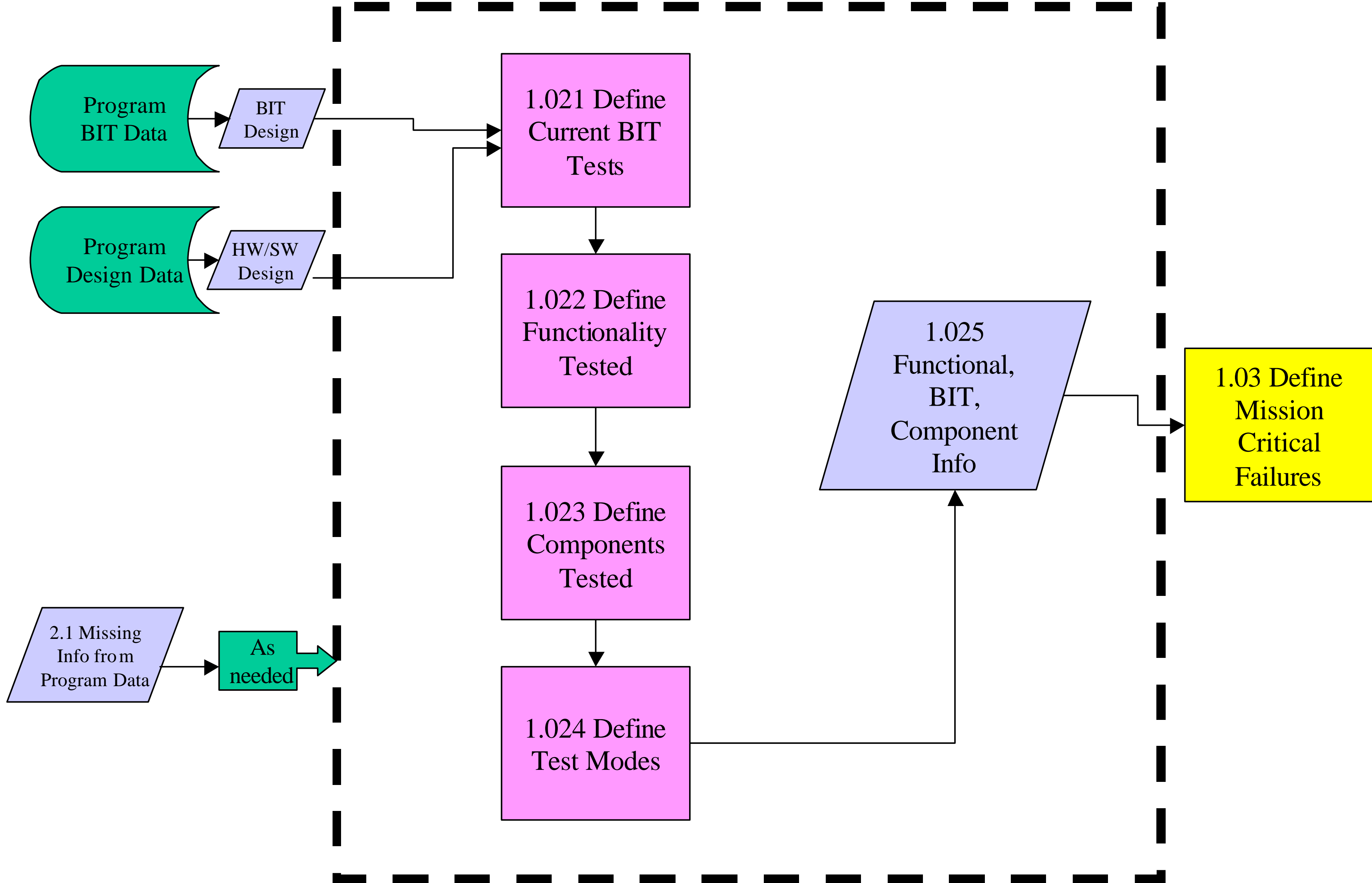
BIT Analysis Process for Existing Products



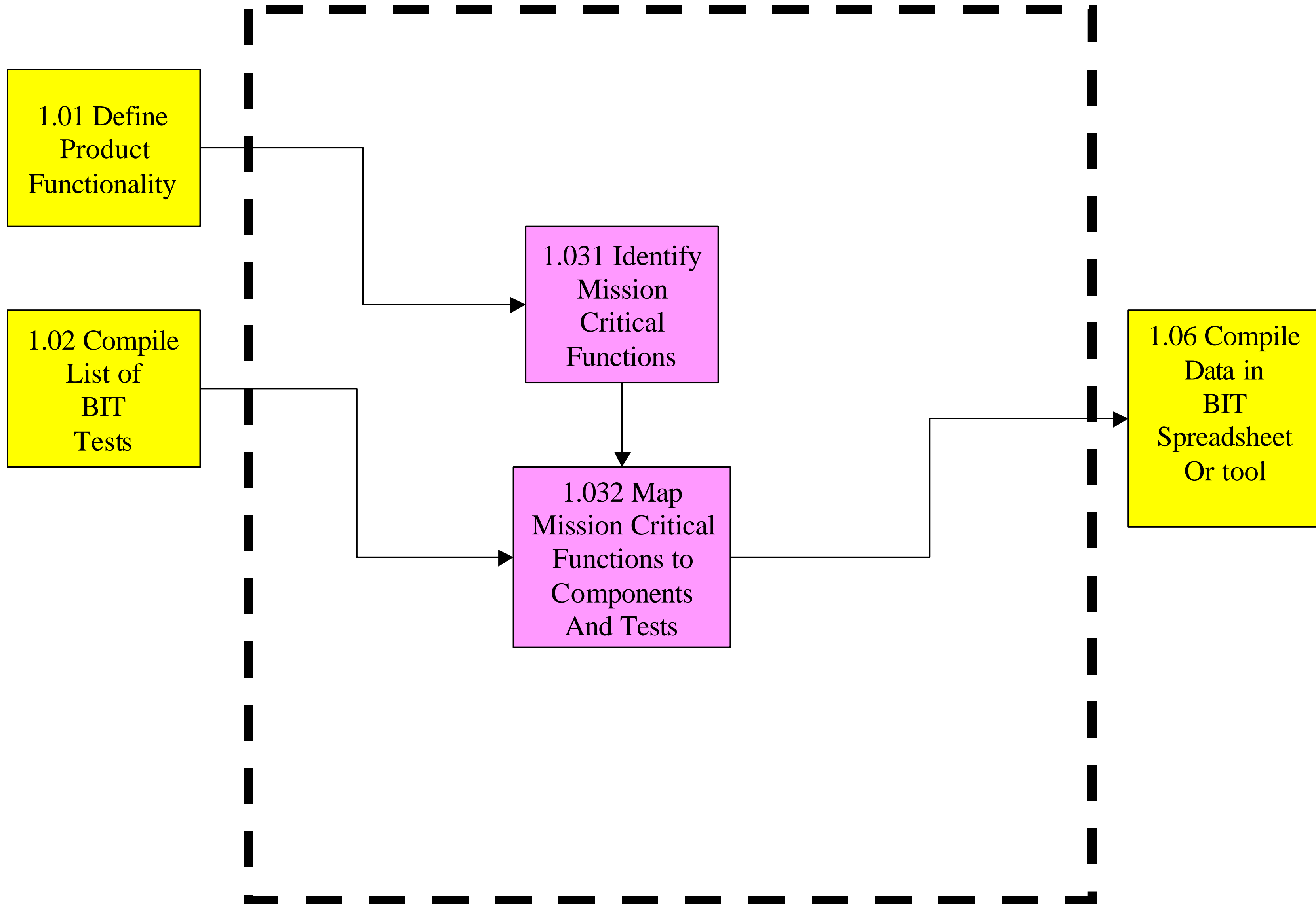
BIT Analysis Process for Existing Products



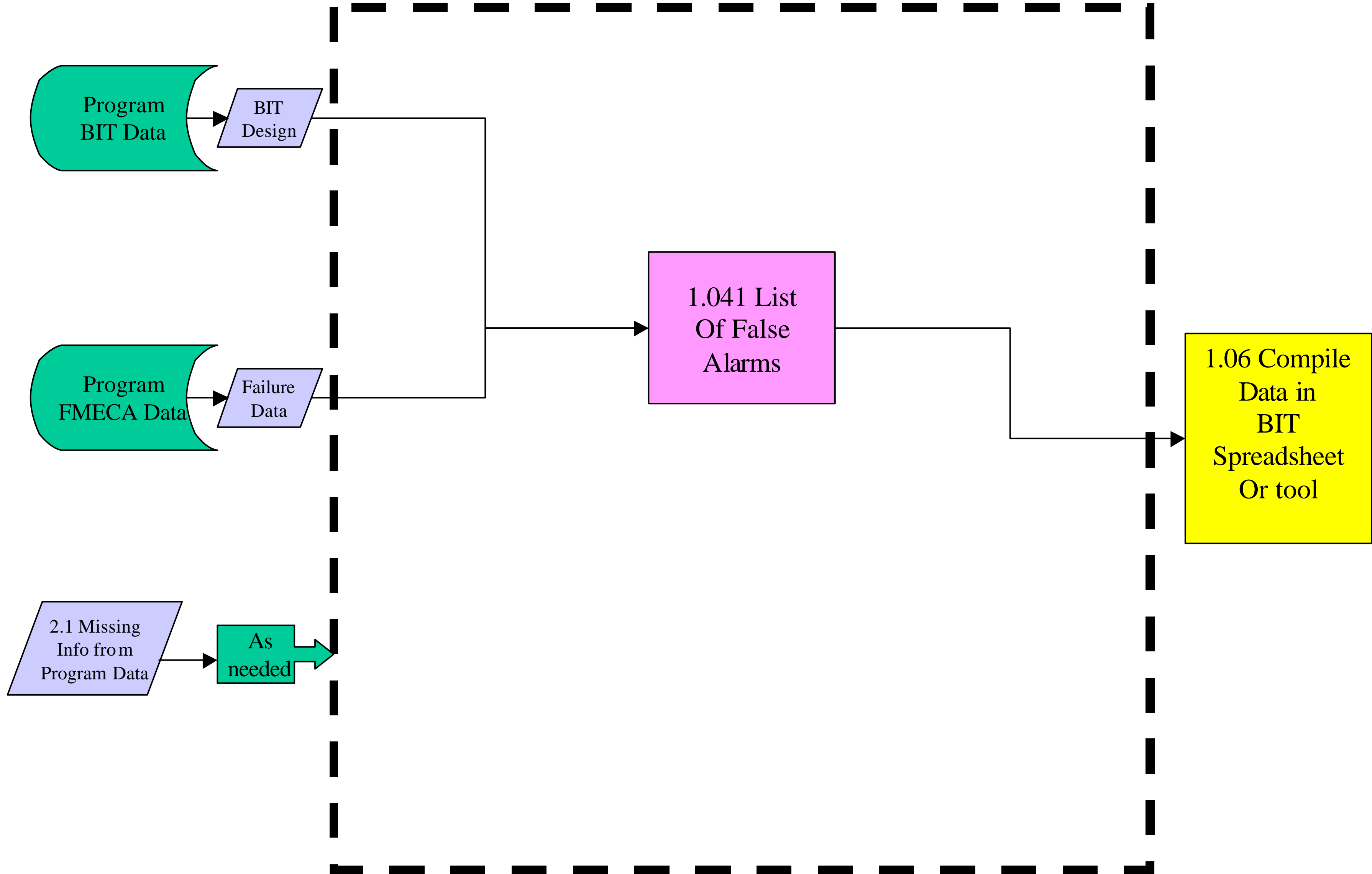
BIT Analysis Process for Existing Products



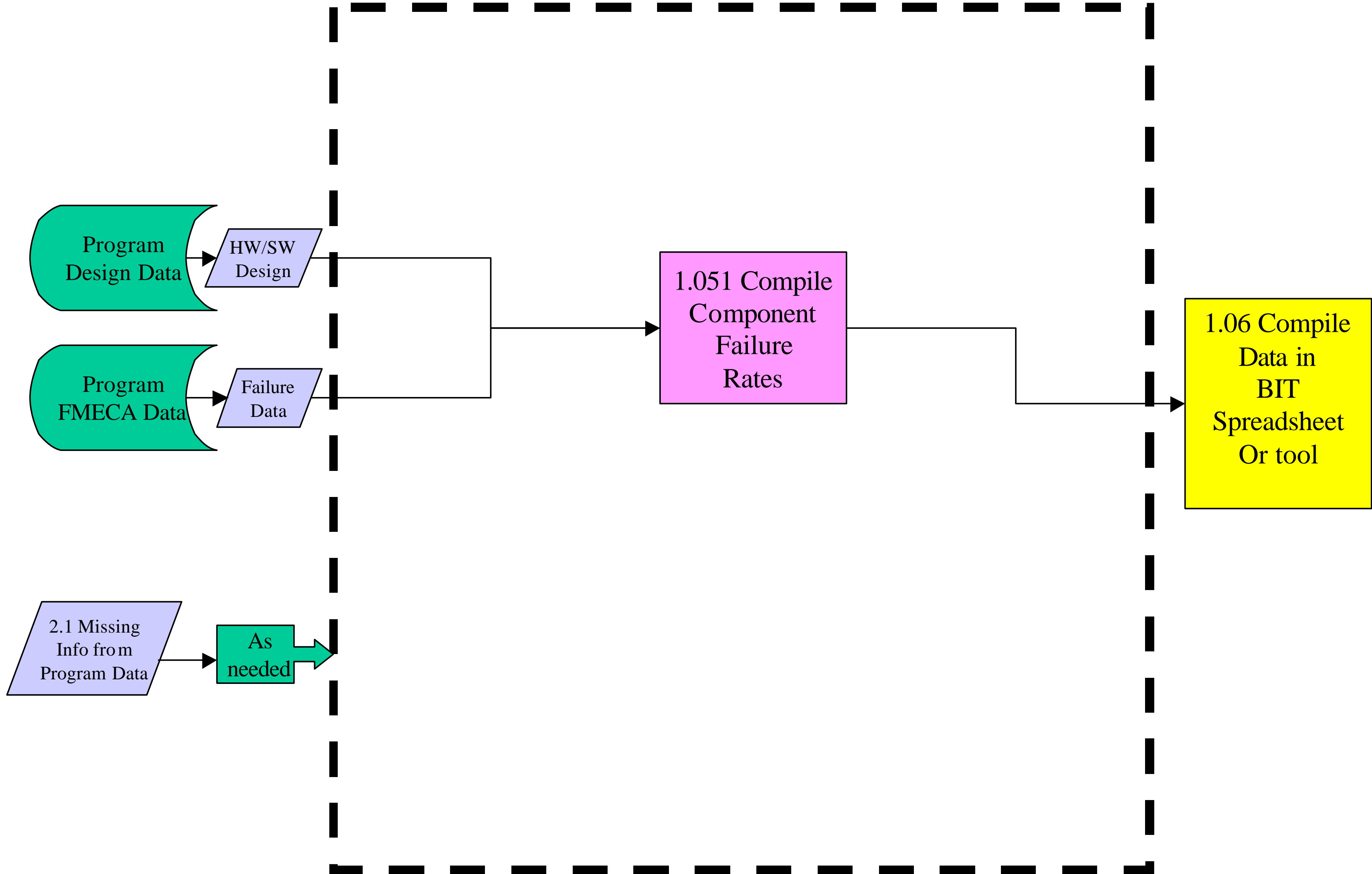
BIT Analysis Process for Existing Products



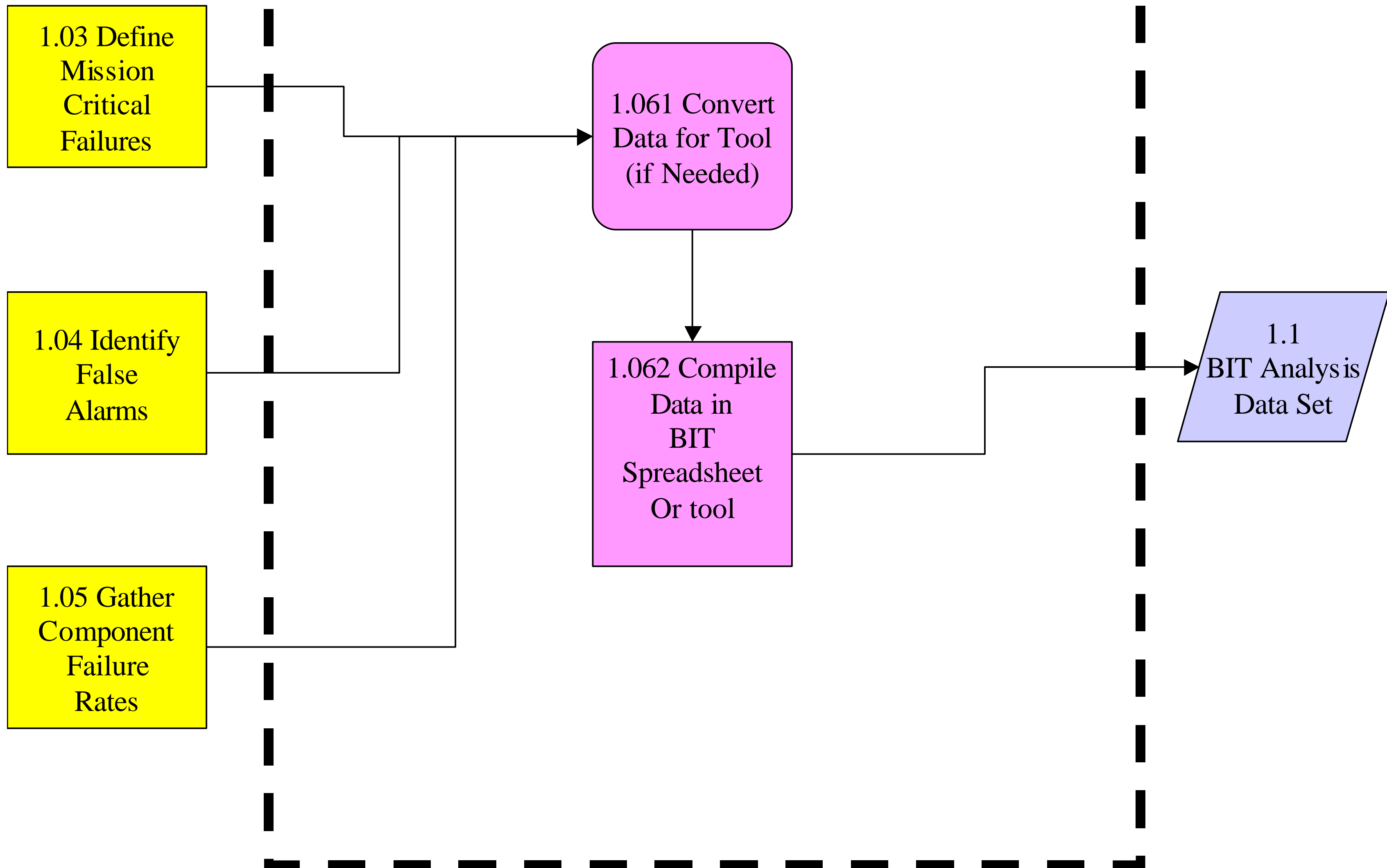
BIT Analysis Process for Existing Products



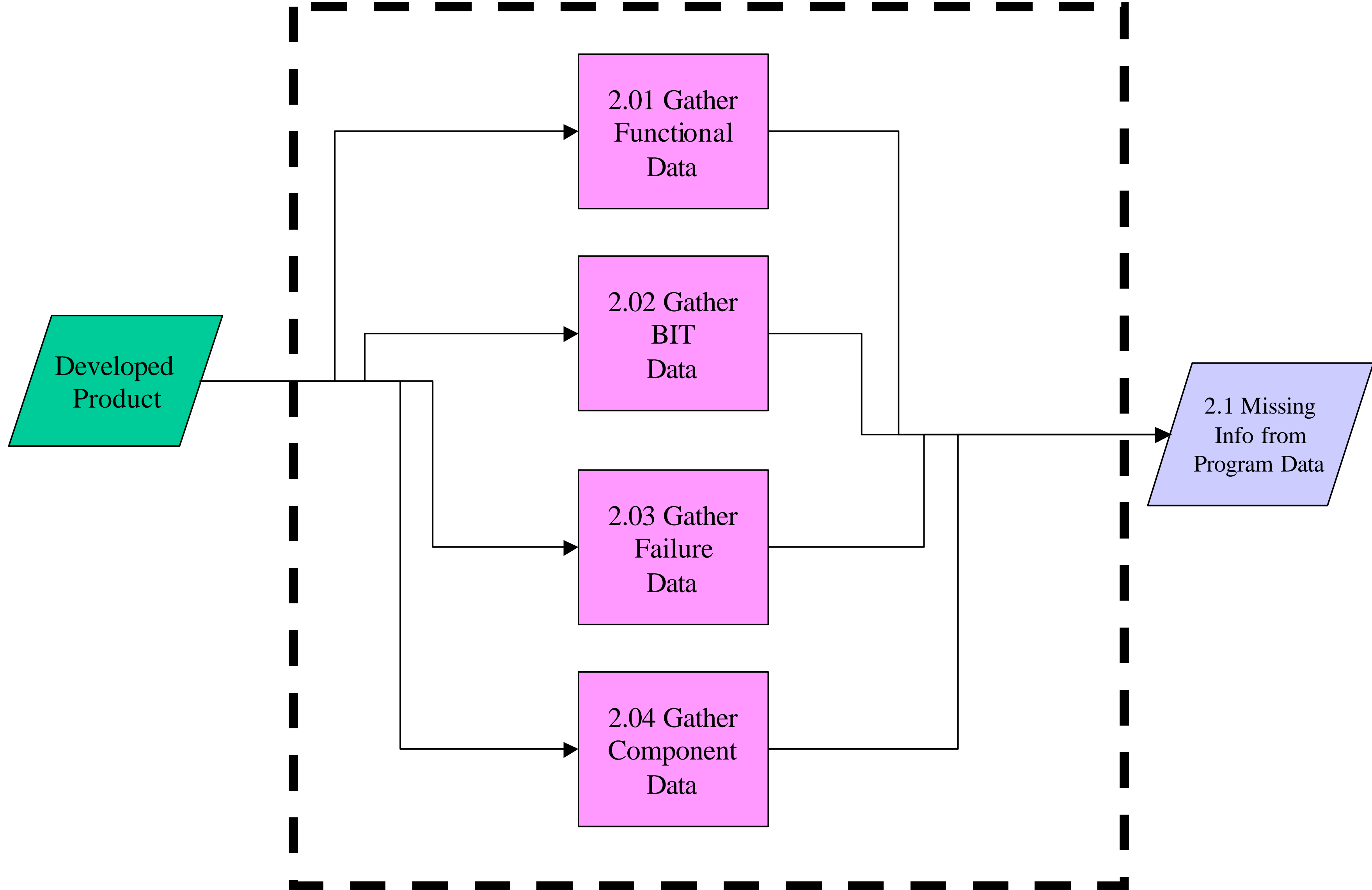
BIT Analysis Process for Existing Products



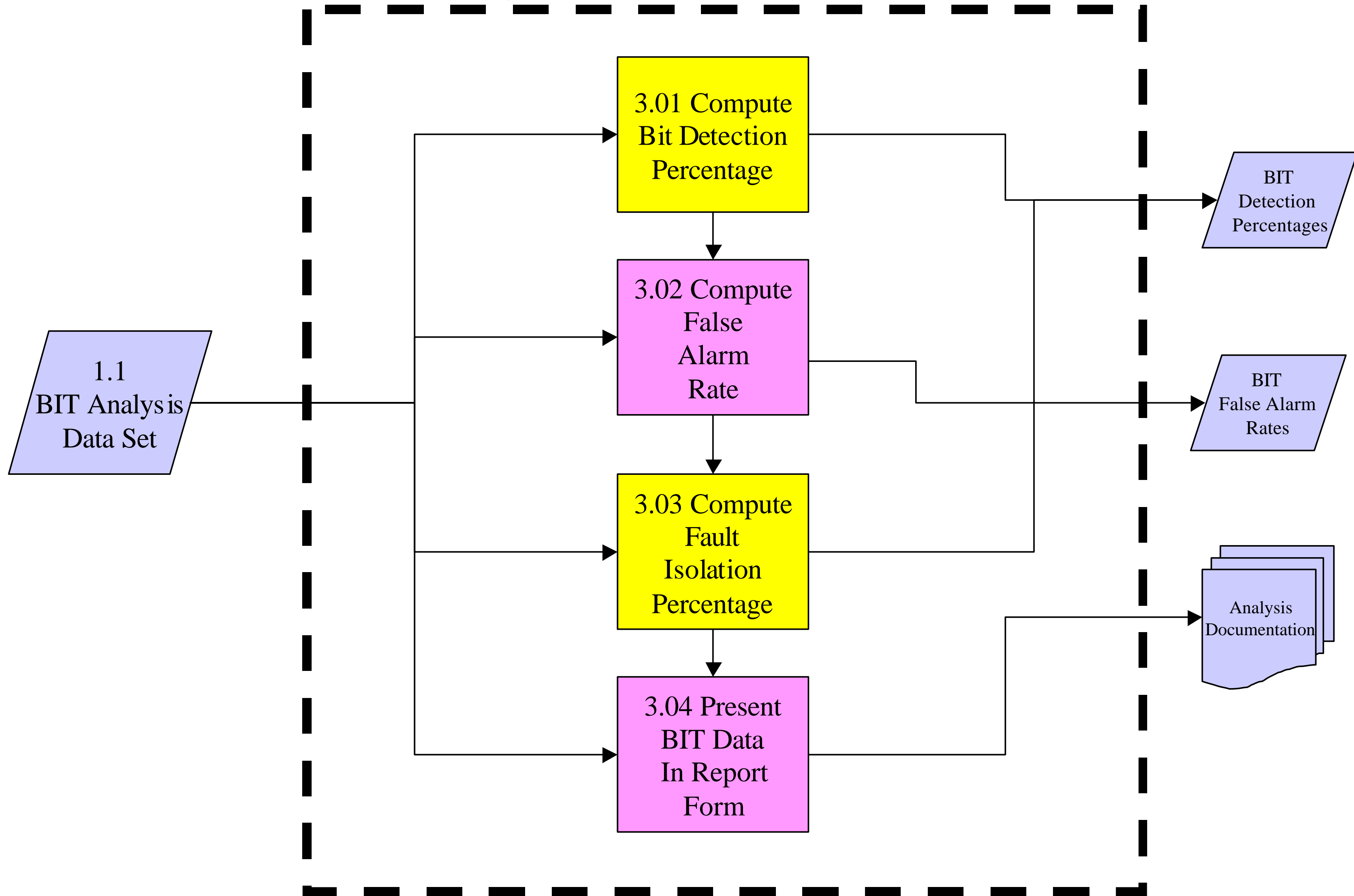
BIT Analysis Process for Existing Products



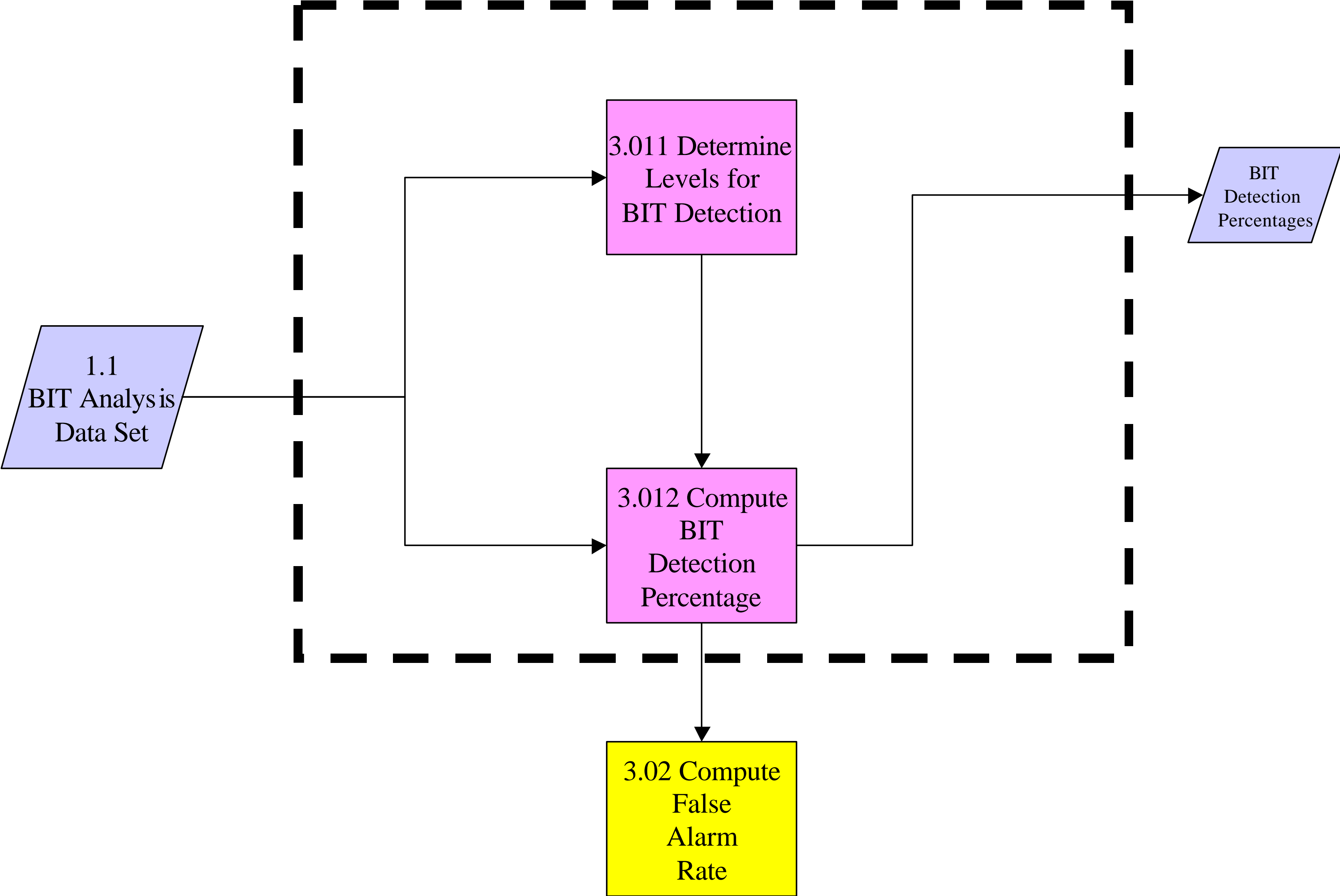
BIT Analysis Process for Existing Products



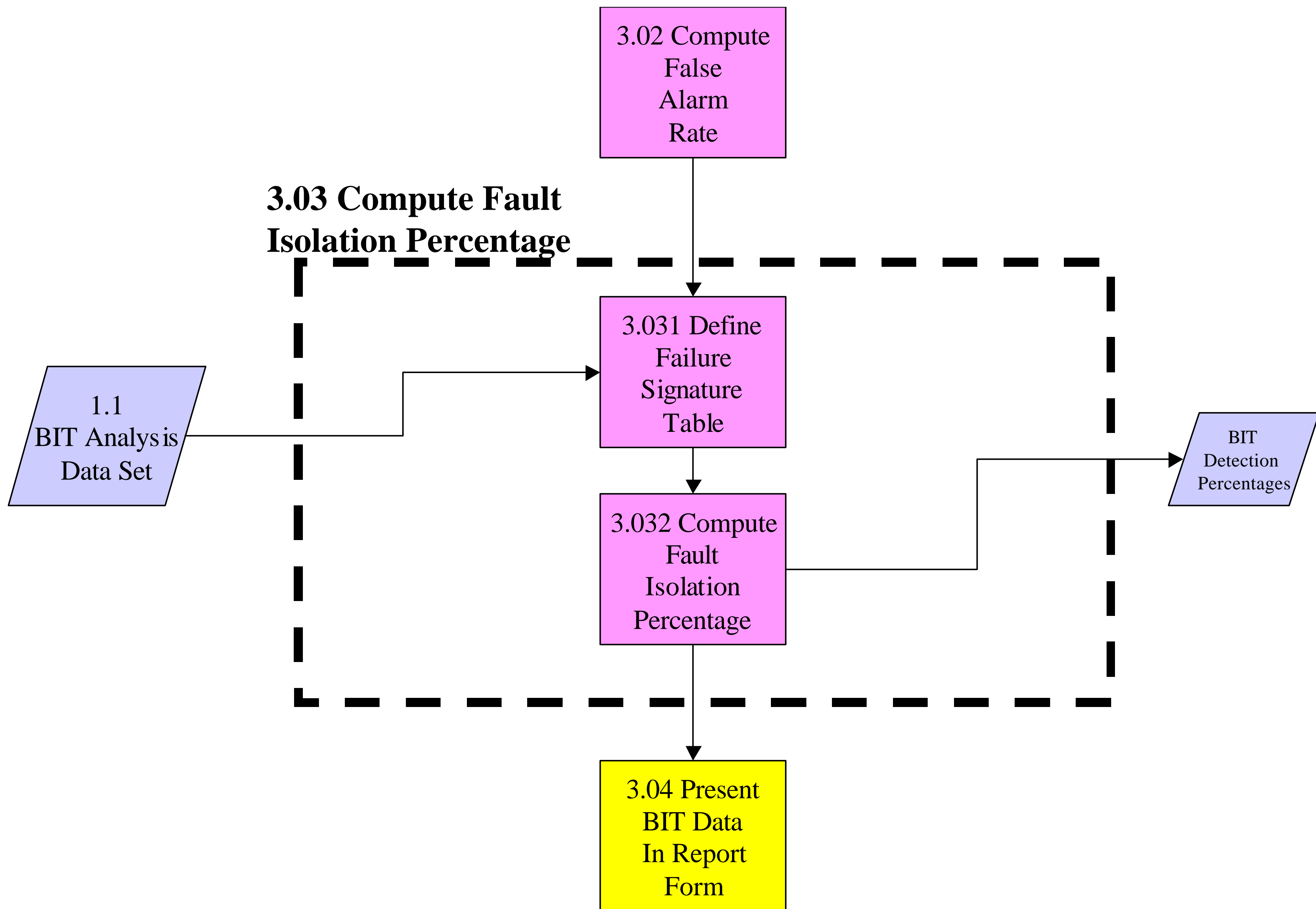
BIT Analysis Process for Existing Products



BIT Analysis Process for Existing Products



BIT Analysis Process for Existing Products



BIT Analysis Process for Existing Products

1.011 Functional Block Diagram

A functional block diagram identifies the major functions of a system and the interactions between these functions.

Steps:

1) Determine the major functions of the system. This information can be obtained from system performance specifications, statements of work (SOW), program overviews, etc. Prepare a complete system definition including identification of expected performance at all indenture levels and system restraints. Functional narratives of the system can be created including description of each mission in terms of functions which identify tasks to be performed for each mission, mission phase and operational mode. Narratives should describe the environmental profiles, expected mission times and equipment utilization, and the functions and outputs of each item. List the functions.

2) Determine the interactions between the functions of the system. This information can be determined from interface control documents (ICDs), system specifications, and design documents.

3) Determine the modes where interactions and functions are valid.

4) Draw a block diagram for each mode or specify mode by surrounding applicable functions/interactions by a box, or specify mode(s) by listing it in the function box and on interaction line.

5) The functional block diagram is created by listing functions in box and interactions on lines in between boxes.

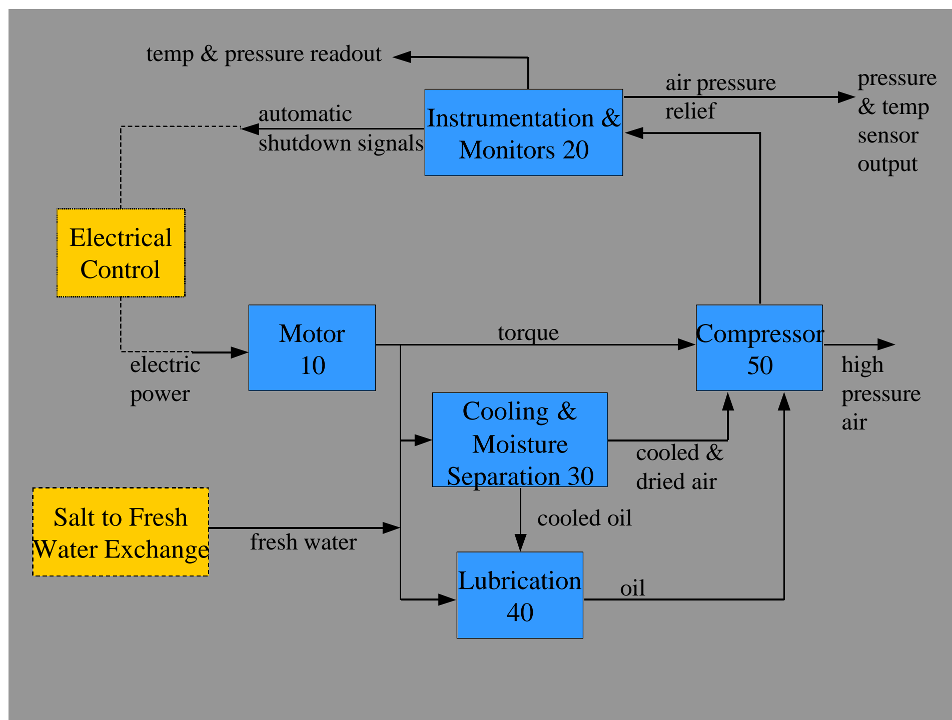
1.011 Functional Block Diagram - Continued

1.011 Continued

Steps:

6) A functional block diagram should be created for the system at system level. If necessary for your analysis, create functional block diagrams for each LRU in the system as well.

7) Reference the example below:



1.012 Functions To Component Mapping

A functions to component map identifies the components that are exercised to satisfy a function.

Steps:

1) Use the functional block diagrams created in 1.011. The lowest level block diagrams should be used.

2) Determine the components in each LRU. These can be obtained from a bill of material (BOM), parts lists, or item drawings.

3) Determine the components used for each function. This can be determined from the LRU schematics, software code, interface requirements specifications and/or design documents (IRS/DD), and algorithm design documents (ADD).

4) Determine if the components are in series, parallel, dependent on each other or redundant. This can be observed with the schematics.

5) Create a list of functions and components and their flow. Use the data sheet template. Reference the example below:

Function	**LRU**	**Component**	**Component Flow**
Switch FOV	Afocal		
		Hall effect sensor A	series
		FOV motor	series
		Hall effect sensor B	series
	Video Converter		
		FOV drive +	parallel
		FOV drive -	parallel

1.013 FMEA/FMECA

A Failure Modes and Effects Analysis (FMEA) identifies possible system failures and their effects. A Failure Modes, Effects and Criticality Analysis (FMECA) adds the criticality of the failure effects to the FMEA.

Steps:

1) Use the functional block diagrams created in 1.011 and the functional to components mapping defined in 1.012.

2) Define failure modes. Identify all potential item and interface failure modes and define their effect on the immediate function or item, on the system, and on the mission to be performed.

3) Define Failure Effects. Evaluate each failure in terms of the worst potential consequence that may result and assign a severity classification category. Consider the consequence of each failure mode on item operation, next higher assembly operation and total system operation when assessing severity.

4) Identify Means of Failure Detection. Identify failure detection methods and compensating provisions for each failure mode. This can be through BIT tests, operator detection through audible or visual warning signals or automatic sensing devices.

5) Identify Compensating Provisions. Identify corrective design or other actions required to eliminate the failure or control the risk. Compensating provisions are design characteristics or operator actions that negate or reduce the effects of a failure. These may include redundant systems, alternative operating modes, and safety devices. These could affect BIT fault detection percentages. NOTE: This step is not necessary if the FMEA is being used strictly for BIT requirements verification.

6) Identify Fault Isolation. Identify the effects of corrective actions or other system attributes, such as requirements for logistics support. BIT Fault isolation is assessed to support logistics and repair efforts. Fault isolation can include both BIT and troubleshooting flows.

7) Compile Results. Failure rates, fault isolation information, BIT information and functional-component mapping will be used for BIT analysis calculations. Use the FMEA template.

1.013 FMEA/FMECA - Continued

1.013 Continued

A criticality analysis (CA) should be performed to accompany the FMEA. Items with high Cr values are also called *mission critical* items. Some BIT requirements may require a certain fault detection percentage on mission critical parameters.

Steps:

8) Use the failure modes in the FMEA. Classify the severity of the failure. There are four categories of severity classifications as shown below:

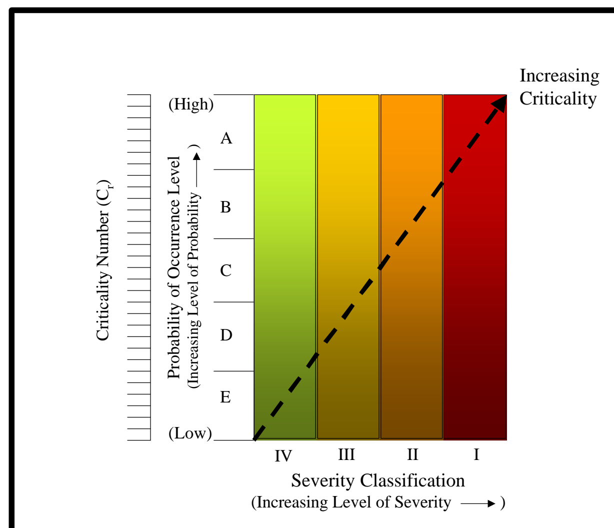
Category I – Catastrophic - A failure which may cause death or weapon system loss

Category II – Critical - A failure which may cause severe injury, major property damage, or major system damage which will result in mission loss

Category III – Marginal - A failure which may cause minor injury, minor property damage, or minor system damage which will result in delay or loss of availability or mission degradation.

Category IV – Minor - A failure not serious enough to cause injury, property damage, or system damage, but which will result in unscheduled maintenance or repair

9) Determine the probability of occurrence. The levels, A through E, correspond to the classes identified on the next page. The combination of the probability of occurrence and the severity classification give the criticality number, C_r. An example of a criticality matrix is below. Use the Criticality Analysis template.



1.013 FMEA/FMECA - Continued

1.013 Continued

9) Determine the probability of occurrence. The levels, A through E, correspond to the classes identified below.

Class	Probability of Occurrence	Description
Class A	Frequent	A high probability of occurrence during the item operating time interval. High probability may be defined as a single failure mode probability greater than 0.20 of the overall probability of failure during the item operating time interval.
Class B	Reasonably Probable	A moderate probability of occurrence during the item operating time interval. Moderate probability may be defined as a single failure mode probability which is more than 0.10 but less than 0.20 of the overall probability of failure during the item operating time interval.
Class C	Occasional	An occasional probability of occurrence during the item operating time interval. Occasional probability may be defined as a single failure mode probability which is more than 0.01 but less than 0.10 of the overall probability of failure during the item operating time interval.
Class D	Remote	An unlikely probability of occurrence during the item operating time interval. Remote probability may be defined as a single failure mode probability which is more than 0.001 but less than 0.01 of the overall probability of failure during the item operating time interval.
Class E	Extremely Unlikely	A failure whose probability of occurrence is essentially zero during the item operating time interval. Extremely unlikely may be defined as a single failure mode probability which is less than 0.001 of the overall probability of failure during the item operating time interval.

1.021 Define Current BIT Tests

Current BIT tests are the tests executed in software and/or firmware to test system components/functionality.

Steps:

1) Create a list of BIT tests executed in software. This can be found in the software design documentation or an algorithm description document (ADD).

2) Create a list of BIT tests executed in firmware. This can be found in firmware code, schematics and/or interface requirements specifications.design documents.

3) Identify any system level tests that are a requirement and may affect the BIT fault detection percentage and/or fault isolation percentage. For example, if a system has a requirement for the operator to detect loss of video, and the requirement is part of the list of tests used to comprise the total DIT detection percentage, include it in this list.

4) Identify when the tests are executed or the BIT Mode. Modes can include power up, continuously, operator initiated, special condition test or a combination of these. Compile the list of BIT tests, their modes and a short description.

1.022 Define Functionality Tested

BIT tests are designed to test system functionality and identify complete or partial loss of functionality.

Steps:

1) Start with the list of BIT tests from 1.021 and the data sheet containing the list of functions and components (1.012).
2) From BIT description, determine the functionality (or partial functionality) tested by each BIT test. The list of function to component mapping will help with BIT tests that test specific components.
3) Compile the list of BIT tests and their functions.

1.023 Define Components Tested

BIT tests are designed to test system functionality and identify complete or partial loss of functionality. The components that satisfy each function have been identified in 1.012, so the BIT test to component mapping can be derived.

Steps:

1) Start with the list of BIT tests and their functions from 1.022 and the data sheet containing the list of functions and components (1.012).

2) Derive the mapping of the BIT test to component based on the function and the BIT test description. Reference schematics if necessary.

3) Add the list of BIT tests to the data sheet template. Map the Bit test to the appropriate components under each function. Some BIT tests may be listed more than once. Some components may have more than one BIT test listed. Some components and/or functions may have no BIT tests listed. Reference the example below:

Function	LRU	Component	Component Flow	BIT Test
Switch FOV	Afocal			
		Hall effect sensor A	series	
		FOV motor	series	
		Hall effect sensor B	series	
	Video Converter			
		FOV drive +	parallel	FOV Drive
				FOV Overvoltage
				FOV OverCurrent
		FOV drive -	parallel	FOV Drive
				FOV Overvoltage
				FOV OverCurrent

1.024 Define Test Modes

BIT tests are executed at predetermined times, or modes. The modes were defined in 1.021.

Steps:

1) Start with the list of BIT tests and their modes from 1.021 and the data sheet containing the list of functions, components, and BIT tests (1.023).

2) Add the list of BIT modes to the data sheet template. Some BIT tests may be executed in more than one mode. Be sure to include operator tests, platform tests, etc. if appropriate. Reference the example below:

Function	LRU	Component	Component Flow	BIT Test	BIT Mode
Switch FOV	Afocal				
		Hall effect sensor A	series		
		FOV motor	series		
		Hall effect sensor B	series		
	Video Converter				
		FOV drive +	parallel	FOV Drive	IBIT
				FOV Overvoltage	CBIT, IBIT
				FOV OverCurrent	CBIT, IBIT
		FOV drive -	parallel	FOV Drive	IBIT
				FOV Overvoltage	CBIT, IBIT
				FOV OverCurrent	CBIT, IBIT

1.031 Identify Mission Critical Functions

Mission Critical Functions are functions that, per the system performance requirements, cause a catastrophic or critical failure in the system if the function does not perform.

Steps:

1) Start with the FMECA from 1.013 and the data sheet containing the list of functions, components, and BIT tests (1.024).

2) Using the customer requirements (performance specification, statement of work, etc), determine the safety critical and mission critical functions. Use the criticality numbers, Cr, assigned in the FMECA to help in the assessment. Create a list of mission critical functions.

1.032 Map Mission Critical Functions to Components And Tests

Mission Critical Functions are functions that, per the system performance requirements, cause a catastrophic or critical failure in the system if the function does not perform.

Steps:

1) Start with the mission critical functions from 1.031 and the data sheet containing the list of functions, components, and BIT tests (1.024).
2) For functions with multiple components, identify which components can lead to the mission critical failure.
3) For each BIT test on a component, determine if it tests a mission critical failure – or partial failure of the component. Partial failure of a component (I.e. degradation) may not mean a mission critical failure.
4) Add the mission critical identifiers to the functions in the data sheet. Mark the appropriate components and tests as mission critical that effect that function. Reference the example below:

Function	LRU	Component	Component Flow	BIT Test	BIT Mode	Mission Critical
Switch FOV	Afocal					Yes
		Hall effect sensor A	series			No
		FOV motor	series			Yes
		Hall effect sensor B	series			No
	Video Converter					
		FOV drive +	parallel	FOV Drive	IBIT	Yes
				FOV Overvoltage	CBIT, IBIT	No
				FOV OverCurrent	CBIT, IBIT	No
		FOV drive -	parallel	FOV Drive	IBIT	Yes
				FOV Overvoltage	CBIT, IBIT	No
				FOV OverCurrent	CBIT, IBIT	No

1.041 List Of False Alarms

False alarms are BIT failures that are flagged by the system when there is no loss of functionality, degradation of performance or failure present.

Steps:

1) Start with the list of BIT tests, their functions, components and modes from 1.024.

2) Review system failure data from production, the field, environmental tests, and any other sources. Ignore true failures (document parts being replaced), ignore intermittent failures on a system. Only consider failures that are repeatable under the same conditions with no failure found.

3) False alarms can usually be contributed to incorrect BIT thresholds, BIT algorithms or test logic.

4) Compile a list of false alarms.

1.051 Compile Component Failure Rates

Component failure rates are critical to the BIT analysis calculations.

Steps:

1) Start with the FMECA from 1.013 and the data sheet containing the list of functions, components, and BIT tests (1.024).

2) The FMECA will contain many components and their failure rates. For components not list on the FMECA, determine the failure rate from MIL-HANDBK-217 or from collected reliability data on the program.

3) All failure rates should be converted to a common factor (i.e. failure rate per 1 million hours, etc).

4) Add the component failure rates to the data sheet. Reference the example below:

Function	LRU	Component	Component Flow	BIT Test	BIT Mode	Mission Critical	Failure Rate
Switch FOV	Afocal					Yes	
		Hall effect sensor A	series			No	2.12
		FOV motor	series			Yes	1.67
		Hall effect sensor B	series			No	2.12
	Video Converter						
		FOV drive +	parallel	FOV Drive	IBIT	Yes	0.023
				FOV Overvoltage	CBIT, IBIT	No	
				FOV OverCurrent	CBIT, IBIT	No	
		FOV drive -	parallel	FOV Drive	IBIT	Yes	0.023
				FOV Overvoltage	CBIT, IBIT	No	
				FOV OverCurrent	CBIT, IBIT	No	

1.061 Convert Data for Tool (if Needed)

Automated reliability tools such as ASENT, CARMA, and Ram Commander may require data in a certain format.

Steps:

1) Start with the the BIT data sheet from 1.051.

2) For each data column in the data sheet, determine the format of the data required for the tool.

3) Convert if necessary. This will most likely affect failure rates.

1.062 Compile Data in BIT Spreadsheet Or tool

Ensure all data is entered appropriately in the BIT data sheet or in the tool. Perform calculations as necessary on the BIT data sheet.

Steps:

1) Start with the the BIT data sheet from 1.051 or the converted data from 1.061.

2) If using a tool, enter the data in the tool per the directions.

3) If using the BIT data sheet, ensure no mistakes have been made on the data entry.

4) Calculate the failure rate for the functions. To do so, use the component, component flow and failure rate to sum the failure rates for each LRU in the function and the function itself. Use the following equations to sum the failure rates:

a) If there are no redundant components of functionality and component failures are statistically independent, the failure rate of a system or function is the sum of the failure rates of the individual components.

$$\lambda_{Fn} = \sum_{i=1}^{n} \lambda_{Ci}$$

Where Fn is the function and there are n independent components, $C_{i,}$ that satisfy the function. The components are said to be in series.

b) If all components must fail in order to cause a loss of functionality, the failure rate of the function is the product of the failure rates of the individual components.

$$\lambda_{Fn} = \prod_{i=1}^{m} \lambda_{Ci}$$

Where Fn is the function and there are m parallel components, C_i, that satisfy the function.

c) A combination of series and parallel would result in sums of products and individual components.

$$\lambda_{Fn} = \sum_{i=1}^{n} \lambda_{Ci} + \prod_{i=1}^{m} \lambda_{Ci} + \prod_{i=1}^{x} \lambda_{Ci} + \prod_{i=1}^{y} \lambda_{Ci} + \ldots$$

Where Fn is the function, there are n independent components C_i, m parallel components $C_{i,}$ x parallel components C_i, y parallel components C_i, and so on.

1.062 Compile Data in BIT Spreadsheet Or tool - Continued

1.062 Continued

Steps:

5) Reference the example below:

Function	LRU	Component	Component Flow	BIT Test	BIT Mode	Mission Critical	Failure Rate
Switch FOV							5.910529
	Afocal					Yes	5.91
		Hall effect sensor A	series			No	2.12
		FOV motor	series			Yes	1.67
		Hall effect sensor B	series			No	2.12
	Video Converter						0.000529
		FOV drive +	parallel	FOV Drive	IBIT	Yes	0.023
				FOV Overvoltage	CBIT, IBIT	No	
				FOV OverCurrent	CBIT, IBIT	No	
		FOV drive -	parallel	FOV Drive	IBIT	Yes	0.023
				FOV Overvoltage	CBIT, IBIT	No	
				FOV OverCurrent	CBIT, IBIT	No	

3.011 Determine Levels for BIT Detection

BIT requirements may require BIT detection percentages to be calculated at different levels.

Steps:

1) Start with the customer BIT detection percentage requirements that need to be verified.

2) For each BIT detection percentage requirement, determine the BIT mode, or combination of BIT modes that are included in the requirement.

3) Prepare a list of applicable BIT tests for each BIT requirement.

3.012 Compute BIT Detection Percentage

Compute the BIT detection percentages to verify the BIT requirements.

Steps:

1) Start with the BIT detection levels from 3.011 and the BIT data sheet from 1.0.
2) For each BIT detection level, compute the BIT detection percentage.
3) The BIT detection percentage is calculated as all *detected* failure mode rates divided by the total failure mode rate for the target system or functionality.
4) If all test detected failure modes have a combined failure rate of 0.50 and the target system has a failure rate of 0.60, total detection is calculated as 0.50/0.60 = 83%.
5) All failure rates for the BIT detection level divided by the total failure rate for all functions listed on the BIT data sheet:

$$\frac{\sum_{i=1}^{n} \boldsymbol{l} DFn_i}{\sum_{i=1}^{n} \boldsymbol{l} Fn_i}$$

When there is a total of i functions, Fn_i, in the system and DFn_i is the failure rate of each function based on the detected failures in the function.

3.02 Compute False Alarm Rate

The False Alarm Rate is calculated from the list of false alarms and the list of BIT tests.

Steps:

1) Start with the BIT data sheet from 1.0 and the false alarm list from 1.041.

2) The false alarm rate is calculated by dividing the sum of the failure rates of the components that caused the incorrect BIT failure by the sum of the failure rates of all detectable failures.

3) The sum of the failure rates of the components that caused the incorrect BIT failure is the components associated with the particular BIT test listed on the false alarm sheet. If the exact nature of the cause of the false alarm is known, only consider the failure rates of the components associated with the failure (not all components affected by BIT test). If not, consider all components related to the BIT test.

4) The sum of the failure rate of all detectable failures is the sum of the component failures detected by BIT. This may include operator test, etc, based on system requirements. Enter this data in the BIT data sheet under Failure Rate detected. For each function, the detected failure rate is only the sum of the failure rates f the components tested by BIT. Reference the example below:

Function	LRU	Component	Component Flow	BIT Test	BIT Mode	Mission Critical	Failure Rate	Failure Rate detected
Switch FOV							5.910529	1.670529
	Afocal					Yes	5.91	1.67
		Hall effect sensor A	series			No	2.12	
		FOV motor	series	FOV motor	IBIT	Yes	1.67	1.67
		Hall effect sensor B	series			No	2.12	
	Video Converter						0.000529	0.000529
		FOV drive +	parallel	FOV Drive	IBIT	Yes	0.023	0.023
				FOV Overvoltage	CBIT, IBIT	No		
				FOV OverCurrent	CBIT, IBIT	No		
		FOV drive -	parallel	FOV Drive	IBIT	Yes	0.023	0.023
				FOV Overvoltage	CBIT, IBIT	No		
				FOV OverCurrent	CBIT, IBIT	No		

3.02 Compute False Alarm Rate - Continued

3.02 Continued

Steps:

5) The false alarm rate is calculated by dividing the sum of the failure rates on the false alarm list by the sum of the failure rates of all detectable failure rates of the functions on the BIT data sheet:

$$\frac{\sum_{i=1}^{m} \mathbf{1} FFn_i}{\sum_{i=1}^{x} \mathbf{1} DFn_i}$$

Where there are x detectable function failures, Dfn_i, in the system and m incorrectly ‘failed’ failures, Ffn_i, in the system.

3.031 Define Failure Signature Table

The Failure Signature Table defines all the possible LRUs that contribute to a failure.

Steps:

1) Start with the list of BIT tests and their modes and the BIT data sheet from 1.0.

2) Use the Fault Signature Table template, and create a table identifying each possible combination of BIT failures.

3) For each BIT failure, list the possible LRUs that contribute to the BIT test from the BIT data sheet.

4) Enter the LRU failure rate for each LRU in the fault signature table.

5) Identify the fault isolation algorithm for the system from system software and maintenance flows.

6) In the Fault Signature table, for each failure signature, identify the LRU the fault isolation algorithm selects.

7) The percent fault isolation is the sum of the failure rates isolating to all LRUs other than the LRU selected by the fault isolation algorithm divided by the sum of the failures rates of all detectable causes.

8) In the example below, the system fault isolation algorithm fault isolates to LRU C for the given fault signature. There are two other possible causing LRUs, but the failure rate is lower for each cause. The percent fault isolation is the sum of the failure rates isolating to all LRUs other than LRU C divided by the sum of the failures rates of all detectable causes. 0.25 + 0.33/1.35 = 43%. (Assuming these were the only tests in a system.)

Fail Mode	Fault Signature	Causing LRU	Failure Rate
X	101101001	LRU A	0.25
Y	101101001	LRU B	0.33
Z	101101001	LRU C	0.77
Total			**1.35**

3.032 Compute Fault Isolation Percentage

The Fault Isolation Percentage is calculated from the Fault Signature Table and the BIT data sheet.

Steps:

1) Start with the Fault Signature Table from 3.031 and the BIT data sheet from 1.0.

2) The percent fault isolation is the sum of the failure rates isolating to all LRUs other than the LRU selected by the fault isolation algorithm divided by the sum of the failures rates of all detectable causes of the failures.

$$\frac{\sum_{i=1}^{m} \mathbf{1} IFn_i}{\sum_{i=1}^{x} \mathbf{1} DFn_i}$$

Where there are x detectable function failures, Dfn_i, in the system and m incorrectly isolated failures, Ifn_i, in the system.

3.04 Present BIT Data In Report Form

The BIT data should be provided in report form to verify the BIT requirements.

Steps:

1) Summarize the BIT detection percentages calculated in 3.012.
2) Provide the false alarm rate calculated in 3.021.
3) Summarize the fault isolation percentage calculated in 3.032.
4) Provide the BIT data sheet from 3.0.
5) Provide the Fault Signature Table from 3.031.
6) Provide the list of BIT tests and modes from 1.021.
7) Provide the FMECA 1.013.
8) Provide BIT test for each level from 3.011.
9) Provide the functional block diagrams from 1.011.

CHAPTER 10
APPENDIX IV – PROCESS TEMPLATES

Failure Mode and Effects Analysis

System__________ Date__________

Indenture Level____ Sheet ___ of ____

Reference Drawing__________ Compiled By_____

ID Number	Functional Nomenclature	Function	Failure Modes And Causes	Operational Mode	Failure Effects			Failure Detection Method	Compensating Provisions	Severity Class	Remarks
					Local Effects	Next Higher Level	End Effects				

Criticality Analysis

System__________ Date__________

Indenture Level____ Sheet ___ of ____

Reference Drawing__________ Compiled By_____

ID Number	Functional Nomenclature	Function	Failure Modes And Causes	Operational Mode	Severity Class	Failure Prob ? data source	Failure Effect Prob (ß)	Failure Mode Ratio (a)	Failure Rate (?)	Operating Time (t)	Failure Mode Crit #	Item Crit #	Remarks

	BIT DATA SHEET									
Function	LRU	Component	Component Flow	BIT Test	BIT Mode	Mission Critical	Failure Rate	Failure Rate detected	Detection % at Detection LevelA	Detection % at Detection LevelB

FAULT SIGNATURE TABLE

Fault Signature	Loss of Functionality	LRU	Failure Rate	Isolated LRU?	Fault Sum	% Fault Isolation

CHAPTER 11
APPENDIX V – NEW HTI BIT ANALYSIS

Operating failure rates

Engineer: Dave Martin

Using: MIL-HDBK - 217FN1

Prediction Date: 4 Oct 1999

Project: LRIP_HTI

Project Desc.: HORIZONTAL TECHNOLOGY INTEGRATION

Indenture	Part #	Ref Des	Description	Temp	Env	QTY	Failure Rate	MTBF	
0	0-0000	1	HTI NV-80 B-KIT	60	GM	1	563.87021	1,773.46	
1	0-0000	1	ELECTRONICS UNIT	60	GM	1	170.49163	5,865.39	
2	0-0000	20	EU CHASSIS +	60	GM	1	47.40603	21,094.36	
3	0-0000	8	COOLING FAN	60	GM	1	15.72	63,613.23	*
3	0-0000	9	MOTHERBOARD/FLEX HARNESS	60	GM	1	20.26	49,358.34	*
3	A3248240-0000	A95	EMI FILTER MODULE	60	GM	1	11.42603	87,519.46	
4	A3248243-0000	1	FAN CONTROL ASSEMBLY	60	GM	1	4.34034	230,396.70	*
4	A3248247-0000	2	EMI PWB ASSEMBLY	60	GM	1	7.08569	141,129.52	*
2	1916948-0001	A88	VIDEO PROCESSOR CCA	60	GM	1	11.4672	87,205.25	
2	A3248225-0000	A89	I/F CONTROL CCA	60	GM	1	33.9667	29,440.60	*
2	A3248230-0000	A90	VIDEO CONVERTER CCA	60	GM	1	42.0979	23,754.15	*
2	A3248270-0000	A91	B-KIT POWER CONVERTER #1	60	GM	1	20.3692	49,093.73	*
2	A3246913-0000	A93	CE/FI Jumper Board	60	GM	1	5	200,000.00	*
2	A3248270-0000	A94	B-KIT POWER CONVERTER #2	60	GM	1	10.1846	98,187.46	*
1	0-0000	1	SENSOR UNIT	60	GM	1	393.37858	2,542.08	
2	A3246915-0000	1	RECEIVER ASSEMBLY	60	GM	1	325.6558	3,070.73	
3	A3246915-0000	1	IMAGER ASSEMBLY	60	GM	1	35.6558	28,045.93	
4	A3246902-0000	3	FILTER WHEEL ASSEMBLY	60	GM	1	12.1794	82,105.85	
4	0-0000	A83	IMAGER OPTICS	60	GM	1	0.688	1,453,488.37	
4	A3246901-0000	A84	SCANNER ASSEMBLY	60	GM	1	22.7884	43,881.98	
3	A3246903-0000	A96	DETECTOR/COOLER BENCH	60	GM	1	290	3,448.28	
4	0-0000	2	Non-OI SADA DETECTOR	60	GM	1	35	28,571.43	*
4	A3190640-000A	3	ONE WATT LINEAR COOLER	60	GM	1	250	4,000.00	*
4	0-0000	4	FLEX CABLE	60	GM	1	5	200,000.00	*
2	A3248010-0000	A81	AFOCAL ASSEMBLY	60	GM	1	9.2392	108,234.48	*
2	1916956-0001	A85	COOLER CONTROL CCA	60	GM	1	8.31428	120,275.00	
2	1916939-0001	A86	SCANNER CONTROL CCA	60	GM	1	21.179	47,216.58	
2	1916943-0001	A87	DIGITIZER CCA	60	GM	1	19.1939	52,099.89	
2	A3248085-0000	A92	POINT OF LOAD REGULATOR	60	GM	1	9.7964	102,078.31	*

Generic Fault Isolation Algorithm

The purpose of the Generic Fault Isolation algorithm is to compute the values for the EU Status and SU Status (i.e., to isolate detected failures to the correct LRU).

If the Generic Fault Isolation algorithm is executing upon transition to the IBIT state or as a result of receiving a Reset Failures Control set to RESET, the EU and SU Statuses will be reset as follows:

EU Status = PASS
SU Status = PASS

where,

- The above values will also be used as the initial values following system power-up.

If the Generic Fault Isolation algorithm is not executing upon transition to the IBIT state or as a result of receiving a Reset Failures Control set to RESET, the EU and SU Statuses will be computed as follows:

```
if (B-Kit Test Results(20) = FAIL ) then                    //SW Compatibility
  if ( EU SW Compatibility = FAIL ) then
    EU Status = FAIL
  else
    SU Status = FAIL
  end if
else if [ ( B-Kit Test Results(23) = FAIL ) or              // PS1 Overcurrent
( B-Kit Test Results(24) = FAIL ) ] then                    // PS1 Overvoltage
  SU Status = FAIL
else if { ( B-Kit Test Results(13) = FAIL ) or              // EU Box Temp Sensor
( B-Kit Test Results(15) = FAIL ) or                        // Master C40
( B-Kit Test Results(16) = FAIL ) or                        // Master EEPROM Checksum
( B-Kit Test Results(17) = FAIL ) or                        // Master RAM
( B-Kit Test Results(18) = FAIL ) or                        // EU/A-Kit UART
( B-Kit Test Results(19) = FAIL ) or                        // EU/SU UART

( B-Kit Test Results(21) = FAIL ) or                        // EU Fault Log EEPROM
( B-Kit Test Results(25) = FAIL ) or                        // PS1 Undervoltage
( B-Kit Test Results(26) = FAIL ) or                        // PS2 Overvoltage
( B-Kit Test Results(27) = FAIL ) or                        // PS2 Undervoltage
( B-Kit Test Results(28) = FAIL ) or                        // EU Fan
[ ( B-Kit Test Results(29) = FAIL ) and                     // SU Fan
( B-Kit Test Results(71) = FAIL ) ] or                      // Cooler Input Power
( B-Kit Test Results(30) = FAIL ) or                        // Reticle RAM
( B-Kit Test Results(31) = FAIL ) or                        // Symbology RAM
( B-Kit Test Results(34) = FAIL ) or                        // Reformatter RAM
( B-Kit Test Results(36) = FAIL ) or                        // Slave C40
( B-Kit Test Results(37) = FAIL ) or                        // Slave EEPROM Checksum
( B-Kit Test Results(38) = FAIL ) } then                    // Slave RAM
  EU Status = FAIL
else if [ ( B-Kit Test Results(11) = FAIL ) or              // VP Bad Timing
( B-Kit Test Results(12) = FAIL ) or                        // Reformatter Freeze Frame Input
( B-Kit Test Results(41) = FAIL ) or                        // VP 1 Control Signature
( B-Kit Test Results(42) = FAIL ) or                        // VP 2 Control Signature
( B-Kit Test Results(75) = FAIL ) ] then                    // Scan Sync
  SU Status = FAIL
else if { ( B-Kit Test Results(32) = FAIL ) or              // Globalization/Polarity
```

```
( B-Kit Test Results(33) = FAIL ) or                    // 2D Filter
( B-Kit Test Results(39) = FAIL ) or                    // Histogram/Normalization
( B-Kit Test Results(40) = FAIL ) or                    // TRS Sums/VP Vertical Direction
[ ( B-Kit Test Results(43) = PASS ) and                 // Filter Wheel Current
( B-Kit Test Results(44) = FAIL ) and                   // Moving Filter Wheel Voltage
( B-Kit Test Results(81) = PASS ) ] or                  // Stationary Filter Wheel Voltage
[ ( B-Kit Test Results(46) = FAIL ) and                 // Moving FOV Current
( B-Kit Test Results(47) = FAIL ) ] or                  // Moving FOV Voltage
( B-Kit Test Results(53) = FAIL ) or                    // Reformatter Freeze Frame Output
[ ( B-Kit Test Results(47) = FAIL ) and                 // Moving FOV Voltage
( B-Kit Test Results(56) = FAIL ) and                   // TRS 1 Drive
( B-Kit Test Results(57) = FAIL ) ] or                  // TRS 2 Drive
[ ( B-Kit Test Results(82) = FAIL ) and                 // Stationary FOV Current
( B-Kit Test Results(83) = FAIL ) ] } then              // Stationary FOV Voltage
  EU Status = FAIL
else if [ ( B-Kit Test Results(1) = FAIL ) or           // Cooler Compressor Temp Sensor
( B-Kit Test Results(2) = FAIL ) or                     // Cooler Compressor Overheat
( B-Kit Test Results(3) = FAIL ) or                     // Scan Error
( B-Kit Test Results(4) = FAIL ) or                     // Boresight Achieved
( B-Kit Test Results(5) = FAIL ) or                     // Cooldown Monitor
( B-Kit Test Results(6) = FAIL ) or                     // EU/SU Serial Link Timeout
( B-Kit Test Results(7) = FAIL ) or                     // FOV Position
( B-Kit Test Results(8) = FAIL ) or                     // TRS 1 Overheat
( B-Kit Test Results(9) = FAIL ) or                     // TRS 2 Overheat
( B-Kit Test Results(10) = FAIL ) or                    // Afocal Temp Sensor
( B-Kit Test Results(14) = FAIL ) or                    // Platform ID
( B-Kit Test Results(22) = FAIL ) or                    // EU/SU Serial Link Checksum
( B-Kit Test Results(29) = FAIL ) or                    // SU Fan
( B-Kit Test Results(43) = FAIL) or                     // Filter Wheel Current
( B-Kit Test Results(44) = FAIL ) or                    // Moving Filter Wheel Voltage
( B-Kit Test Results(45) = FAIL ) or                    // Filter Wheel Position
( B-Kit Test Results(46) = FAIL ) or                    // Moving FOV Current
( B-Kit Test Results(47) = FAIL ) or                    // Moving FOV Voltage
( B-Kit Test Results(48) = FAIL ) or                    // Focus Position
( B-Kit Test Results(49) = FAIL ) or                    // Focus Position Sensor
 ( B-Kit Test Results(51) = FAIL ) or                   // Digitizer Gain
( B-Kit Test Results(52) = FAIL ) or                    // Digitizer Level
( B-Kit Test Results(54) = FAIL ) or                    // TRS 1 Response
( B-Kit Test Results(55) = FAIL ) or                    // TRS 2 Response
( B-Kit Test Results(56) = FAIL ) or                    // TRS 1 Drive
( B-Kit Test Results(57) = FAIL ) or                    // TRS 2 Drive
( B-Kit Test Results(58) = FAIL ) or                    // TRS 1 Temp Sensor
( B-Kit Test Results(59) = FAIL ) or                    // TRS 2 Temp Sensor
( B-Kit Test Results(60) = FAIL ) or                    // SADA Bad Channels
( B-Kit Test Results(61) = FAIL ) or                    // SU C40
( B-Kit Test Results(62) = FAIL ) or                    // SU EEPROM Checksum
( B-Kit Test Results(63) = FAIL ) or                    // SU RAM
( B-Kit Test Results(64) = FAIL ) or                    // SADA EEPROM
( B-Kit Test Results(65) = FAIL ) or                    // SU Fault Log EEPROM
( B-Kit Test Results(66) = FAIL ) or                    // SADA Serial I/O
( B-Kit Test Results(67) = FAIL ) or                    // POL N12A
( B-Kit Test Results(68) = FAIL ) or                    // POL N5A
( B-Kit Test Results(69) = FAIL ) or                    // POL P12A
( B-Kit Test Results(70) = FAIL ) or                    // POL P5A
( B-Kit Test Results(71) = FAIL ) or                    // Cooler Input Power
( B-Kit Test Results(72) = FAIL ) or                    // Digitizer Serial I/O
```

```
( B-Kit Test Results(73) = FAIL ) or                    // POL P5B
( B-Kit Test Results(74) = FAIL ) or                    // Digitizer Control Signature
( B-Kit Test Results(76) = FAIL ) or                    // SADA BIST
( B-Kit Test Results(77) = FAIL ) or                    // Scanner Command
( B-Kit Test Results(78) = FAIL ) or                    // Scanner Position Limits
( B-Kit Test Results(79) = FAIL ) or                    // Scanner Resolver Excitation
( B-Kit Test Results(80) = FAIL ) or                    // Scanner Resolver Feedback
( B-Kit Test Results(81) = FAIL ) or                    // Stationary Filter Wheel Voltage
( B-Kit Test Results(82) = FAIL ) or                    // Stationary FOV Current
( B-Kit Test Results(83) = FAIL ) or                    // Stationary FOV Voltage
( B-Kit Test Results(86) = FAIL ) or                    // Focus Position Limits
( B-Kit Test Results(89) = FAIL ) or                    // Scanner Motor High
( B-Kit Test Results(90) = FAIL ) or                    // Scanner Motor Low
( B-Kit Test Results(91) = FAIL ) or                    // Scanner Current Feedback High
 ( B-Kit Test Results(93) = FAIL ) ] then          // Scanner Current Feedback Zero
   SU Status = FAIL
end if
```

Test ID	Test Name	Instrumentation Parameter	Units	Max Expected	Min Expected	Failure Threshold	When Executed	Clear Failure Reqmts	Comments
1	Cooler Compressor Temp Sensor	filtered cooler compressor temp	degrees C	76	-45	>110 or < -65	FW BIT, FOV BIT, TRS BIT, CBIT	Temp returns to normal range and next BIT manager fault detection cycle executes	Not instrumented
2	Cooler Compressor Overheat		degrees C			> 90 deg C	CBIT, FW BIT, FOV BIT, TRS BIT	If the temp overheat indicator has been set, it will be set to 'temp OK' when temp is less than 80 deg C	Test fails if the temp - after it is filtered - exceed the threshold. The filtered temp is output to the Cooler Compressor sensor test.
3	Scan Error	scanner command	amps			> 3.56 amps	whenever scanner is running	Start scanner - 30 Hz init, 60 Hz init, TRS BIT, transition to thermal	test fails with 5 scanner command readings over threshold. Scanner will shut down with this failure
4	Boresight Achieved	Boresight Data Valid	Boolean	N/A	N/A	TRUE = Fail	whenever scanner is running	Cycle power or command IBIT	Not instrumented. This test fails if the Boresight Data Valid is false more than 29 times
5	Cooldown Monitor						Upon power being applied to cooler		Test fails if the current exceeds the threshold or the time count exceeds the threshold and/or the FPA cooled indicator is not set.
		Cooler Motor Current Min and Max							
		CC Temp	degrees C						
		Cooldown Indicator	Boolean	N/A	N/A	TRUE = Fail			
		Cooldown Timer	minutes			> 14.9			
		FPA Temp	deg K						
6	EU/SU Serial Link Timeout	EU/SU New Communication Indicator	Boolean	N/A	N/A	TRUE = Fail	All segments except C40 Boot	Cycle power or command IBIT	
7	FOV Position						FW BIT, FOV BIT, TRS BIT, CBIT	Cycle power or command IBIT	If the FOV Position is not In-Transit and the FOV Position indicator is not equal to the HW FOV Position Indicator, the test fails
		FOV Position Indicator	Enumera	N/A	N/A	FOV Ind not equal to HW FOV Ind			

		HW FOV Position Indicator	Enumera	N/A	N/A	FOV Ind not equal to HW FOV Ind			
		FOV Position BIT	Boolean	N/A	N/A	TRUE = Fail			
8	TRS 1 Overheat								
9	TRS 2 Overheat								
10	Afocal Temp Sensor						FW BIT, FOV BIT, TRS BIT, CBIT	Temp returns to normal range and next BIT manager fault detection cycle executes	
		Afocal Filtered Temp	degrees	71.9	-44	>110 or < -65			
		Afocal Housing Temp Sensor BIT	Boolean	N/A	N/A	TRUE = Fail			
11	VP Bad Timing						All BIT segments except for C40 Boot and SADA Init and Standby	Timing and sequence returns to normal and next BIT manager fault detection cycle executes	This test fails if the VP interrupt is not received within the timing or sequence specified in the Norm ADD. If the test fails 14 times, the BIT flag is set.
		IDO/ID1 State	Enumera	N/A	N/A				
		Scan Sync State	Enumera	N/A	N/A				
		VP Interrupt	Boolean	N/A	N/A				Not instrumented
		VP Bad Timing	Boolean	N/A	N/A	TRUE = Fail			
12	Reformatter Freeze Frame Input					7 True readings on the Detect Write and/or Detect Read Indicators	All BIT segments except for C40 Boot and SADA Init	Cycle power or command IBIT	This test verifies the presence of the ID1 and ID0 inputs (detect write) and the TV Field Sync and Filed ID outputs (detect read) sync signals to the reformatter.

		Detect Write	Boolean	N/A	N/A	TRUE			
		Detect Read	Boolean	N/A	N/A	TRUE			
13	EU Box Temp Sensor	EU Temp	degrees	85	-49	>110 or < -65	FW BIT, FOV BIT, TRS BIT, CBIT	Temp returns to normal range and next BIT manager fault detection cycle executes	
14	Platform ID						SADA Init	Cycle power or command IBIT	
		Platform ID Voltage	Volts			Invalid code			Platform ID codes in ICD
15	Master C40		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
16	Master EEPROM Checksum		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
17	Master RAM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
18	EU/A-Kit UART		Boolean	N/A	N/A	TRUE = Fail	SADA Init	Cycle power or command IBIT	Not instrumented
19	EU/SU UART		Boolean	N/A	N/A	TRUE = Fail	SADA Init	Cycle power or command IBIT	Not instrumented
20	Deleted		N/A						
21	EU Fault Log EEPROM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot	Cycle power or command IBIT	Not instrumented
22	EU/SU Serial Link Checksum		Boolean	N/A	N/A	TRUE = Fail	From completion of C40 Boot to end of power cycle	Checksum passes and next BIT manager fault detection cycle executes	Not instrumented
23	PS1 Overcurrent	PS1 Overcurrent	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	
24	PS1 Overvoltage	PS1 Overvoltage	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	

25	PS1 Undervoltage	PS1 Undervoltage	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	
26	PS2 Overvoltage	PS2 Overvoltage	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	
27	PS2 Undervoltage	PS2 Undervoltage	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	
28	EU Fan	EU Fan	Boolean	N/A	N/A	FALSE = fail	CBIT	Test passes and next BIT manager fault detection cycle executes	Instrumentation values are invalid for 1st 90 seconds after power up
29	SU Fan	SU Fan	Boolean	N/A	N/A	FALSE = fail	CBIT	Test passes and next BIT manager fault detection cycle executes	Instrumentation values are invalid for 1st 90 seconds after power up
30	Reticle RAM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
31	Symbology RAM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
32	Globalization/Polarity		Boolean	N/A	N/A		30 Hz Init, 60 Hz Init	Cycle power or command IBIT	Not instrumented
33	2D Filter		Boolean	N/A	N/A		30 Hz Init, 60 Hz Init	Cycle power or command IBIT	Not instrumented
34	Reformatter RAM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
35	EU/A-Kit Serial Link Timeout	EU/A-Kit New Communication Indicator	Boolean	N/A	N/A	TRUE = Fail	All segments except C40 Boot	Cycle power or command IBIT	This test will be set to fail if the Indicator is FALSE 8 or more times

36	Slave C40		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
37	Slave EEPROM Checksum		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
38	Slave RAM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
39	Histogram/Normalization	Histogram files					30 Hz Init, 60 Hz Init	Cycle power or command IBIT	Executes on SMT only
40	TRS Sums/VP Vertical Direction						30 Hz Init, 60 Hz Init	Cycle power or command IBIT	
41	VP 1 Control Signature					7 Failures	All segments except C40 Boot, SADA Init	Cycle power or command IBIT	
		Clock Fail	Boolean	N/A	N/A	FAIL			
		Column Sync Fail	Boolean	N/A	N/A	FAIL			
		ID0 Fail	Boolean	N/A	N/A	FAIL			
		ID1 Fail	Boolean	N/A	N/A	FAIL			
		IDO VDATA	Enumera	N/A	N/A	12=PASS			
		Data BIT	Boolean	N/A	N/A	FAIL			
		Data Noise	Boolean	N/A	N/A	FAIL			
42	VP 2 Control Signature	Scan Sync Fail	Boolean	N/A	N/A	7 Failures	All segments except C40 Boot, SADA Init	Cycle power or command IBIT	
43	Filter Wheel Current					Test fails with 2 readings over threshold	FW BIT, FOV BIT, TRS BIT, CBIT	Cycle power or command IBIT	
		SW Position	Enumera	N/A	N/A				
		HW Position	Enumera	N/A	N/A				
		FW Current	amps			> 0.45 amps			
		Current BIT	Boolean	N/A	N/A	FAIL			

44	Moving Filter Wheel Voltage								
45	Filter Wheel Position								
46	Moving FOV Current								
47	Moving FOV Voltage								
48	Focus Position								
49	Focus Position Sensor	Filtered Focus Position	steps	807	269	>1012 or < 12	FOV BIT, TRS BIT, CBIT	Temp returns to normal range and next BIT manager fault detection cycle executes	
		Focus Position Sensor BIT	Boolean	N/A	N/A	TRUE = Fail			
50	Deleted								
51	Digitizer Gain								
52	Digitizer Level								
53	Reformatter Freeze Frame Output								
54	TRS 1 Response								
55	TRS 2 Response								
56	TRS 1 Drive								
57	TRS 2 Drive								
58	TRS 1 Temp Sensor	TRS1 Filtered Temp	degrees	77.5	-55.5	>110 or < -65	FW BIT, FOV BIT, TRS BIT, CBIT	Temp returns to normal range and next BIT manager fault detection cycle executes	
		TRS1 Temp Sensor BIT	Boolean	N/A	N/A	TRUE = Fail			
59	TRS 2 Temp Sensor	TRS2 Filtered Temp	degrees	77.5	-55.5	>110 or < -65	FW BIT, FOV BIT, TRS BIT, CBIT	Temp returns to normal range and next BIT manager fault detection cycle executes	
		TRS2 Temp Sensor BIT	Boolean	N/A	N/A	TRUE = Fail			
60	SADA Bad Channels								

61	SU C40		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
62	SU EEPROM Checksum		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
63	SU RAM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
64	SADA EEPROM								
65	SU Fault Log EEPROM		Boolean	N/A	N/A	TRUE = Fail	C40 Boot		Not instrumented
66	SADA Serial I/O								
67	POL N12A								
68	POL N5A								
69	POL P12A								
70	POL P5A								
71	Cooler Input Power	Cooler Power Data	Boolean	N/A	N/A		From completion of C40 Boot to end of power cycle	Cycle Power or run IBIT	
72	Digitizer Serial I/O								
73	POL P5B								
74	Digitizer Control Signature					7 Failures	All segments except C40 Boot, SADA Init	Cycle Power or run IBIT	
75	Scan Sync						Whenever scanner is running	Cycle Power or run IBIT	Not instrumented.
76	SADA BIST		Boolean	N/A	N/A	TRUE = Fail			Not instrumented. This test is not executed.
77	Scanner Command								
78	Scanner Position Limits								
79	Scanner Resolver Excitation								
80	Scanner Resolver Feedback								

81	Stationary Filter Wheel Voltage								
82	Stationary FOV Current								
83	Stationary FOV Voltage								
84	Aux PS Overvoltage	Aux PS Overvoltage	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	
85	Aux PS Undervoltage	Aux PS Undervoltage	Boolean	N/A	N/A	3 failure indications (TRUE = fail)	From completion of C40 Boot to end of power cycle	Test passes and next BIT manager fault detection cycle executes	
86	Focus Position Limits								
87	EU Overheat	EU Temp	degrees C			> 96 deg C	CBIT, FW BIT, FOV BIT, TRS BIT	filtered temp falls below 86 deg C	based on EU box temp. The test fails if the filtered EU box temp is greater than threshold. The filtered temp is output to the EU Box sensor test.
88	SU Overheat		degrees C			> 94 deg C	CBIT, FW BIT, FOV BIT, TRS BIT	filtered temp falls below 84 deg C	based on Afocal temp. The test fails if the filtered afocal temp is greater than threshold. The filtered temp is output to the afocal sensor test.
		Afocal Raw Temperature	degrees C						
		Afocal Filtered Temp	degrees C			> 94 deg C			
		SU Overheat BIT	Boolean	N/A	N/A	TRUE = Fail			
89	Scanner Motor High								
90	Scanner Motor Low								
91	Scanner Current Feedback High								
92	Deleted								
93	Scanner Current Feedback Zero								

				HTI DATA SHEET							
				BIT ANALYSIS - LRIP							
Using: MIL-HDBK-217FN1								Engineer: Davinia Chism			
Temperature: 60.00					Environment: GM			Prediction Date: Sept 2002			
Project: LRIP_HTI					Project Desc.: HORIZONTAL TECHNOLOGY INTEGRATION						
Function	**LRU**	**Component Part #**	**Ref Des**	**Component**	**Component Flow**	**BIT Test**	**BIT Mode**	**Mission Critical?**	**F/R**	**F/R Detected**	**CBIT + Host Platform**
Entire System		0-0000	1	HTI NV-80 B-KIT				Y	565.3229	565.32	
1		0-0000	1	SENSOR UNIT				1	393.9897	393.99	
2		A3246915-0000	1	RECEIVER ASSEMBLY				1	325.6558	325.66	
3		A3246915-0000	1	IMAGER ASSEMBLY				1	35.6558	35.66	
Switch Filters	Filter Wheel Assy	A3246902-0000	3	FILTER WHEEL ASSEMBLY				1	12.1794	12.18	
		1917047-0001	X0001	Filter Number 1	series			1	0.0470		*
		1917048-0001	X0002	Filter Number 2	series			1	0.0470		*
		1917049-0001	X0003	Filter Number 3	series			1	0.0470		*
						FW Pos	SBIT, IBIT, CBIT	Y		12.04	
		1917057-0001	X0004	Filter Wheel filter CCA	series				1.9328	1.93	*
		1917053-0001	X0005	Bearing	series				0.0940	0.09	*
		1917051-0001	X0006	Potentiometer	series				0.5000	0.50	*
		1917052-0001	X0007	Motor, filter wheel	series				9.5116	9.51	*
						FW Current	SBIT, IBIT, CBIT	N		11.44	
		1917052-0001	X0007	Motor, filter wheel	series				9.5116	9.51	*
		1917057-0001	X0004	Filter Wheel filter CCA	series				1.9328	1.93	*
						Stat FW Volt	SBIT, IBIT, CBIT	N		11.44	
		1917052-0001	X0007	Motor, filter wheel	series				9.5116	9.51	*
		1917057-0001	X0004	Filter Wheel filter CCA	series				1.9328	1.93	*
						Mov FW Volt	SBIT, IBIT, CBIT	N		11.44	
		1917052-0001	X0007	Motor, filter wheel	series				9.5116	9.51	*
		1917057-0001	X0004	Filter Wheel filter CCA	series				1.9328	1.93	*
4	Imager Optics	0-0000	A83	IMAGER OPTICS				1	0.6880	0.69	
5		1917023-0001	X0001	Lens 1, Imager	NA			1	0.0470	0.05	*
5		1917024-0001	X0002	Lens 2, Imager	NA			1	0.0470	0.05	*