

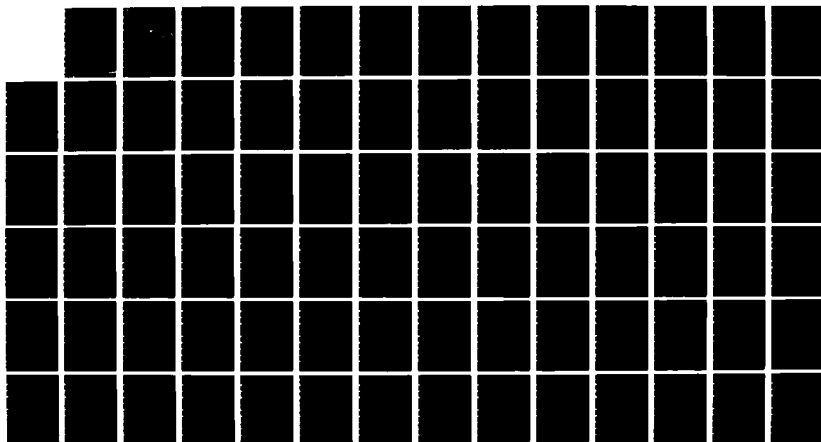
AD-A186 283

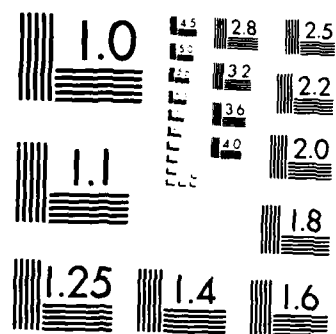
FAULT TREE RELIABILITY ANALYSIS OF THE NAVAL
POSTGRADUATE SCHOOL MINI-SATELLITE (ORION)(U) NAVAL
POSTGRADUATE SCHOOL MONTEREY CA T G KEEBLE SEP 87
F/G 12/4

1/1

UNCLASSIFIED

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

DTIC FILE COPY

NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A186 283



DTIC
ELECTE
NOV 20 1987
S H D

THESIS

FAULT TREE RELIABILITY ANALYSIS OF THE
NAVAL POSTGRADUATE SCHOOL MINI-SATELLITE
(ORION)

by

Trenton G. Keeble

September 1987

Thesis Advisor: J. D. Esary

Approved for public release; distribution is unlimited

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY		3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE		5 MONITORING ORGANIZATION REPORT NUMBER(S)	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)		7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL (if applicable) 55	7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	
8a NAME OF FUNDING SPONSORING ORGANIZATION	8b OFFICE SYMBOL (if applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS (City, State, and ZIP Code)		10 SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO	PROJECT NO
		TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) Fault Tree Reliability Analysis of the Naval Postgraduate School Mini-Satellite (ORION)			
12 PERSONAL AUTHOR(S) KEEBLE, Trenton G.			
13a TYPE OF REPORT Master's Thesis	13b TIME COVERED FROM _____ TO _____	14 DATE OF REPORT (Year, Month, Day) 1987 September	15 PAGE COUNT 82
16 SUPPLEMENTARY NOTATION			
COSAT CODES		18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB GROUP	
		Reliability, Fault Tree Analysis, ORION	
19 ABSTRACT (Continue on reverse if necessary and identify by block number)			
Fault tree analysis, which has proved to be a useful analytical tool for the reliability and safety analysis of complex systems, is applied to the Naval Postgraduate School Mini-Satellite (ORION). A general background to reliability analysis, fault tree analysis, and fault tree construction is given. Impact of a phased mission is included in the analysis. A identify minimal cut sets and minimal path sets. The cuts sets and path sets are, in turn, used to calculate an estimate of ORION's reliability to perform a three year mission. The reliability model was constructed in a Lotus 1-2-3 spreadsheet to enable the designers to do "what-if" analysis.			
20 DISTRIBUTION AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS		21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL Prof. J. D. Esary		22b TELEPHONE (Include Area Code) 408-646-2780	22c OFFICE SYMBOL 55E

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Dist Special



A-1

Approved for public release: distribution is unlimited.

Fault Tree
Reliability Analysis
of the Naval Postgraduate School
Mini-Satellite (ORION)

by

Trenton G. Keeble
Major, United States Army
B.S., United States Military Academy, 1975

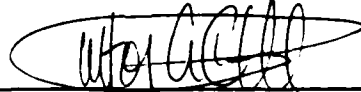
Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1987

Author:



Trenton G. Keeble

Approved by:



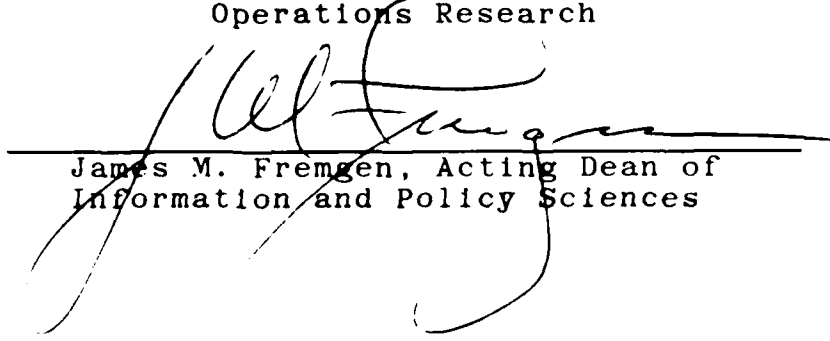
J.D. Esary, Thesis Advisor



Allen E. Fuhs, Second Reader



Peter Purdue, Chairman, Department of
Operations Research



James M. Fremgen, Acting Dean of
Information and Policy Sciences

ABSTRACT

Fault tree analysis, which has proved to be a useful analytical tool for the reliability and safety analysis of complex systems, is applied to the Naval Postgraduate School Mini-Satellite (ORION). A general background to reliability analysis, fault tree analysis, and fault tree construction is given. Impact of a phased mission is included in the analysis. A fault tree for ORION is constructed and used to identify minimal cut sets and minimal path sets. The cut sets and path sets are, in turn, used to calculate an estimate of ORION's reliability to perform a three year mission. The reliability model was constructed in a Lotus 1-2-3 spreadsheet to enable the designers to do "what-if" analysis.

TABLE OF CONTENTS

I.	INTRODUCTION	8
A.	GENERAL BACKGROUND AND PURPOSE	8
B.	SATELLITE OVERVIEW	8
	1. Objectives of ORION	8
	2. ORION Main Subsystems	9
	3. Possible Military Applications	9
C.	ORGANIZATION	10
D.	SUMMARY	10
II.	BACKGROUND TO RELIABILITY ANALYSIS	12
A.	GENERAL	12
B.	PHASED MISSIONS	14
C.	MISSION PROFILES	16
III.	FAULT TREE ANALYSIS DESCRIPTION	18
A.	BACKGROUND TO FAULT TREE ANALYSIS	18
B.	PURPOSE OF FAULT TREES	20
C.	ASSUMPTIONS	22
D.	ADVANTAGES OF FAULT TREES	23
E.	DISADVANTAGES OF FAULT TREES	24
F.	CONSTRUCTION OF A FAULT TREE	24
	1. Events	25
	2. Logic Gates	25
	3. Special Symbols	28
G.	MINIMAL CUT SETS	28
H.	MINIMAL PATH SETS	33
I.	PROBABILITY EVALUATION OF FAULT TREES	33
J.	IMPORTANCE OF BASIC EVENTS	37
IV.	SYSTEM RELIABILITY ANALYSIS	40
V.	CONCLUSION	43
	A. OVERALL FINDINGS	43
	B. RECOMMENDATIONS	43
	APPENDIX A ORION SUBSYSTEM SCHEMATICS	45

APPENDIX B	ORION FAULT TREES	55
APPENDIX C	ORION PATH AND CUT SETS	64
APPENDIX D	LOTUS SPREADSHEET LISTING	69
LIST OF REFERENCES		79
INITIAL DISTRIBUTION LIST		80

LIST OF FIGURES

3.1	Example of a Fault Tree	21
3.2	Events	26
3.3	Logic Gates	27
3.4	Special Symbols	29
3.5	Fault Tree for Example 3.1	31
A.1	ORION Subsystem Summary (Management and Design Viewpoint)	46
A.2	ORION Subsystem Summary (Reliability Viewpoint)	47
A.3	Thermal Control Subsystem	48
A.4	Propulsion Subsystem	49
A.5	Telemetry, Tracking, and Control Subsystem .	50
A.6	Computer Subsystem	51
A.7	Attitude Control Subsystem	52
A.8	Data Storage Subsystem	53
A.9	Electrical Power Subsystem	54
B.1	Top of Fault Tree	56
B.2	Control Fault Tree	57
B.3	Attitude Detection Fault Tree	58
B.4	Propulsion Fault Tree	59
B.5	Thruster Fault Tree	60
B.6	Feed Fault Tree	61
B.7	Telemetry, Tracking and Communications Fault Tree	62
B.8	Electrical Power Fault Tree	63
C.1	Path Sets	68

ACKNOWLEDGMENT

This analysis effort could not have been accomplished without the assistance and guidance from Marty Mosier, System Engineer for ORION. The additional insights provided by Ed Senasack and Tom Whitmeyer, both of the Naval Research Laboratory, were essential to the completion of this thesis. Thanks to these three gentlemen my education has been broadened and my depth of understanding of satellites has increased immeasurably.

I. INTRODUCTION

A. GENERAL BACKGROUND AND PURPOSE

The Naval Postgraduate School Mini-Satellite (subsequently referred to as ORION) is an actual engineering effort by the students and faculty of the Naval Postgraduate School to produce a low cost, multi-purpose satellite. The focus of this thesis, as a portion of that effort, is to derive a fault tree for ORION and assist in its design by identifying weak links in its system reliability. The format of the thesis is intended to make the results of this analysis readily accessible to colleagues to facilitate the design and construction of ORION.

B. SATELLITE OVERVIEW

ORION is an alternative concept for low cost military spaceflight. It is designed to be an inexpensive, reliable satellite bus that can be mission specific, yet maintain a flexible architecture. The mission payloads can vary from 50 lbs. to 130 lbs. and are designed for a mission life of three years. Due to its simplistic design, ORION includes very little redundancy.

1. Objectives of ORION

ORION is designed with eight objectives in mind. They are:

- a. to satisfy many small mission needs with a low cost, reconfigurable vehicle.
- b. to provide an affordable, boosted-free flyer to complement SPARTAN and SPAS¹.

¹SPARTAN and SPAS are existing experimental platforms used by the Shuttle. They are on station as long as the Shuttle is on station.

- c. to achieve circular orbits from 135 nm (nautical miles) to 800 nm with propellant reserve.
- d. to achieve elliptic orbits to 2200 nm with a perigee of 135 nm.
- e. to have a longer life at Shuttle altitude than SPARTAN.
- f. to provide an affordable platform for space science, space technology, and military missions.
- g. to provide a cost effective bus for constellation proliferation.
- h. to be dependable and affordable.

2. ORION Main Subsystems

For purposes of management and design, ORION can be separated into seven subsystems. The subsystems are:

- a. the propulsion subsystem.
- b. the electrical power subsystem.
- c. the data storage subsystem.
- d. the telemetry subsystem.
- e. the thermal control subsystem.
- f. the attitude control subsystem.
- g. the computer subsystem.

The reliability analysis focuses on how the subsystems interrelate. As an example, all the subsystems require the electrical power subsystem to work. These dependency relationships are developed and displayed in the fault tree.

3. Possible Military Applications

Due to ORION's objectives and simplistic design, there are several apparent military applications. Some of those applications include:

- a. proliferated platforms for communication.
- b. ultraviolet sensor platforms.

- c. high energy particle detectors.
- d. targeting laser or KE (kinetic energy) weapons, reentry vehicle simulator, or kill assessment.
- e. low cost imaging platforms.

C. ORGANIZATION

This chapter provides some background to ORION and its possible applications. Chapter II gives a short background of reliability analysis. Chapter III follows with a description of fault tree analysis. Chapter IV contains the applications of a fault tree analysis to ORION. The final chapter, Chapter V, states the conclusions, recommendations, and suggestions for further research.

D. SUMMARY

The primary benefit of this analysis has been to aid in the design of ORION. This was accomplished by identifying 82 minimal cut sets. Of these cut sets 22 are single-element sets, 29 are double-element cut sets, 27 are three-element cut sets, 2 are five-element cut sets, 1 is a six-element cut set and 1 is an eleven-element cut set.

The dual tree reveals over 33 billion distinct paths. Using modular decomposition this number is reduced to three distinct paths. The path sets were used to determine the structural importance of each component.

The structural importance analysis determined seven different levels of significance. Twenty components are structurally the most significant. A listing of them is given in Appendix C. The remaining levels and their associated components are listed in Chapter IV.

The reliability importance of components cannot be determined since the design is not completely established. A Lotus spreadsheet was developed to allow the designers to do a "what-if" analysis with component reliabilities as the subsystems are developed.

II. BACKGROUND TO RELIABILITY ANALYSIS

A salesman called on Steinway & Sons to show them a new piano-key pin. "My company believes this aluminum pin is greatly superior to the pin you have been using," he said.

Mr. Steinway deliberated for some moments. "Well, young man," he said at last, "we are an old firm, slow and cautious about making changes. But we will install your pins in one of our pianos and give them a trial."

The salesman was delighted. "That's good enough for me," he said. "How long a trial will you need?"

"Oh," said Mr. Steinway thoughtfully, "I'd say about 50 years." [Ref. 1]

A. GENERAL

Performing the mission is undoubtedly the best test of reliability. However, today's decision makers and analysts rarely have Mr. Steinway's luxury of time. Not only is time a scarce resource, but there are many cases when neither the system's working or living environment nor the money to do extensive or realistic reliability tests is available. With such constraints, other methods must be employed to estimate reliabilities or limits on reliabilities. Reliability, in the sense used here and throughout the thesis, is the probability of a device performing its function adequately for a specified length of time and operating conditions. Therefore, the purpose of reliability or system analysis is to seek out those reliabilities or limits on reliabilities. Within that pursuit, there are two important aspects to a system analysis: (1) an

inductive analysis stage and (2) a deductive analysis stage.

During the inductive analysis stage, available information on the system is gathered and organized. The system is then defined, its functional purpose described, and its critical components determined. At this stage, the question is posed "What can happen to the system as a result of component failure or human error?" Possible system failure modes are then hypothesized. A failure modes and effects analysis is conducted at the component level. Specifically, a list of all envisioned mechanical and electrical failure modes is generated. This, in turn, leads to a critical components list including assessed failure rates. Additionally, it is well known that system failures often occur at subsystem interfaces. The interfaces, therefore, become an important part of the analysis along with the components.

The deductive analysis of a system or reliability analysis answers the question "How can a system fail (or succeed) or be unavailable?" A logic tree (or fault tree) is often the best device for deducing how a major system failure event could occur. However, its construction depends on a thorough understanding of the system and the results of the system inductive analysis. A block diagram or a network graph is a useful device for representing a successfully functioning system. Since the network graph is close to a system functional representation, it cannot capture abstract system failure and human error events as well as the logic tree representation. [Ref. 2: pp. 1-2]

Also during the deductive stage a particular method of analysis must be selected and employed. Some of those methods include: fault tree analysis; state space approach; decomposition method; circuit stress

analysis; network reduction technique; block diagrams; and Monte Carlo simulation. Each has its advantages and disadvantages. The primary reason fault tree analysis was selected is that ORION is still in its design stage and fault tree analysis is particularly beneficial in developing a design.

B. PHASED MISSIONS

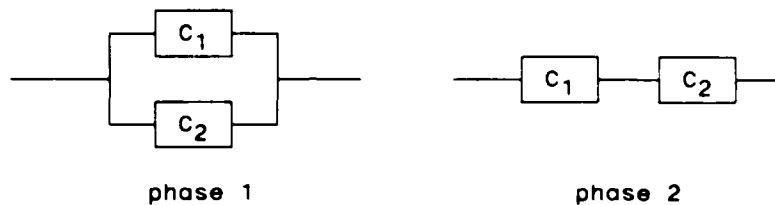
Phases of deployment affect a satellite's reliability. A phase change occurs whenever the size of the set of active components changes. Another way to look at this is to say the functional organization of the system changes with time. During each phase of the mission the system must accomplish a specified task.

A phased mission profile causes complexities not present in a single-phase system. However, it can be transformed into an equivalent synthetic single-phase system. This refined profile can then be used to derive an approximation of, or bounds on, mission or satellite reliability.

It is inappropriate to do a standard reliability analysis for each separate phase, and then multiply the resulting phase reliabilities together as if they referred to independent events. The implicit assumption, that each component is functioning at the beginning of a phase when the system has functioned throughout the previous phase, is not necessarily true. [Ref. 3: pp. 11, 12] A component must have survived the first $n-1$ phases before it can function in the n^{th} phase. Additionally, through the sequence of phases, a component or set of components may be turned on and off several times during the first $n-1$ phases before it is needed during the n^{th} phase. These are all reasons the phase reliabilities cannot be merely multiplied together to obtain an overall system reliability. A

simple example follows to illustrate phased mission analysis.

Example 2.1 A system with two independent components, C_1 and C_2 , is designed for a two-phased mission. In order for the system to perform the required tasks, at least one component has to function through phase 1 and both components have to function through phase 2. The block diagrams for this system is



For $k=1,2$, let p_{k1} denote the probability that component C_k functions through phase 1, and p_{k2} denote the conditional probability that component C_k functions through phase 2, given that it has functioned through phase 1. The system reliability for phase 1 is

$P_1 = P_{11} + P_{21} - P_{11}P_{21}$, and the system reliability for phase 2, given that both the components have functioned through phase 1, is $p_2 = P_{12}P_{22}$. Multiplying these together would lead to the mission reliability

$$P = (P_{11} + P_{21} - P_{11}P_{21})P_{12}P_{22}$$

This is greater than the correct mission reliability, which is

$$P_{11}P_{12}P_{21}P_{22}$$

since mission success is achieved only if both components function through both phases. [Ref. 3: pp. 12-13]

C. MISSION PROFILES

An additional complication to phased missions is the absence of an exact mission profile for ORION. Since ORION is designed to be a low-cost general purpose bus for an electronics package, it can be employed in an infinite variety of profiles. For purposes of this analysis, two distinct profiles are analyzed.

The first mission profile envisions a 3-axis stabilized sensor platform that does not experience an orbit change. After the satellite has been ejected from the canister it becomes autonomous. A short time delay is needed before ORION begins its mission profile. The time delay is necessary to insure ORION is sufficiently away from the Shuttle before it becomes active. This profile is partitioned into five phases. They are:

- activation
- antenna boom deployment
- establish orientation
- re-orientation (if necessary)
- station keeping

The purpose of the activation phase is to "wake up" ORION and conduct internal checks to insure ORION is functioning. The antenna deployment phase is completed when the antenna booms are locked in the extended position. The specific mission of the orientation phase is to establish ORION's spatial and orbital orientation. The fourth phase may or may not occur. If it is determined that ORION is not properly oriented then re-orientation is essential. This phase includes any necessary re-orientation commands. The final phase ensures ORION maintains the orbit(s) specified by its mission profile. All of ORION's subsystems are required (i.e. must function) to perform station keeping tasks.

The second mission profile is for a spin stabilized satellite with an orbit change. Such a profile is characteristic of a communications satellite. This profile has nine phases with the same four initial phases as the first mission profile (i.e. activation, antenna boom deployment, orientation and re-orientation). The remaining five phases are:

- orbit boost
- orbit fix
- orientation
- re-orientation (if necessary)
- station keeping

The purpose of the orbit boost phase is to accelerate ORION out of its low earth orbit. The orbit fix phase establishes ORION's mission orbit. The remaining three phases are identical in purpose to the final three phases of the first mission profile. Again, all of ORION's subsystems must function to perform station keeping tasks.

In both mission profiles (or in any mission profile generated) the last phase utilizes all of the satellite's subsystems. Since all subsystems are needed during the last phase, the phased mission analysis dictates that every subsystem must survive the entire mission life. The resulting synthetic single-phase is all the subsystems operating in series during the entire length of the mission.

III. FAULT TREE ANALYSIS DESCRIPTION

A. BACKGROUND TO FAULT TREE ANALYSIS

The bulk of this chapter is a compilation of information extracted from reliability literature. It is included here only to give the reader a background to the fault tree reliability analysis performed in this thesis.

The fault tree method resulted from a contract between the Air Force Ballistics Division and Bell Telephone Laboratories for the study of an inadvertent launch of the Minuteman ICBM. The Launch Control Safety Study (1962) first described fault tree analysis in Volume I Section VII "Method of Inadvertent Launch Control Analysis." Minuteman I was in production when the study was completed, therefore no design changes resulted from the study (effecting design changes has become a primary advantage of fault tree analysis). Because the results of the analysis were so close to the observed data of Minuteman I, fault tree analysis was used during the design phase of Minuteman II. Since then, fault tree analysis has been used in combination with other techniques to predict and improve safety performance and reliability in complex aerospace and military systems.

After initial work at Bell Telephone Laboratories, development of the fault tree method continued at the Boeing Company, where the technique was applied to manned spacecraft. Boeing and AVCO published fault tree reports on the Minuteman II system in March 1963, and January 1964, respectively. In June 1965, Boeing and the University of Washington co-sponsored a System Safety Symposium in Seattle. Five of the presentations

were fault tree articles by Boeing employees. A paper by A. B. Mearns of Bell Telephone Laboratories also described fault trees. These six papers and the Launch Control Safety Study are the main references cited in articles after 1965. [Ref. 4: p. 3]

Fault tree analysis consists of six steps:

1. define the top event to be investigated,
2. gain an understanding of the system,
3. construct the tree,
4. collect quantitative data,
5. evaluate the probability of the top event,
and
6. analyze the results.

The top event of the tree should be well defined in terms of operating modes of the system, environmental conditions and time limits. However, the failure must represent a major system malfunction which threatens personnel or equipment.

Generally accepted symbols are necessary to represent differences in events and logic relationships since the fault tree is graphic as well as analytic. In addition, several people at separate locations and at different times may contribute to the analysis. The following sections describe events, logic gates and special symbols.

Instead of being hardware oriented, fault tree analysis is event or failure oriented; that is, it examines a particular system failure for all possible causes. Control of the system failure through knowledge of its causes is the analysis objective. The tree is a graphical representation of possible causes of a major failure which appears at the top of the tree (called the top event). During construction, the tree grows downward and outward as failures and causes are described in increasing detail. When the tree is

completed, probabilities are associated with the failures lowest on the tree. The bottom events concern failures of basic components which can be associated with probabilities. The assigned probabilities are combined as dictated by logic gates to give probabilities for events higher on the tree. The combination of probabilities continues until the complex top event has a probability calculated from the accurate component data at the bottom of the tree. In general, fault tree analysis involves two kinds of reasoning: the thought processes involved in construction produce a downward flow, whereas the evaluation of probability and operation of the logic gates dictate an upward flow. [Ref. 4: pp. 1,6,7] See Figure 3.1 for an example of a fault tree.

B. PURPOSE OF FAULT TREES

Generally, fault trees serve three purposes.

First, they aid in determining the possible causes of a system failure. When properly used, the fault tree often leads to discovery of failure combinations which otherwise might not have been recognized as causes of the top event.

Secondly, they serve as a display of results. If the system design is not adequate, the fault tree can be used to show what the weak points are and how they lead to undesirable events. If the design is adequate, the fault tree can be used to show that all conceivable causes have been considered.

Lastly, they provide a convenient and efficient format helpful in the computation of the probability of system failure. [Ref. 5: p. 10]

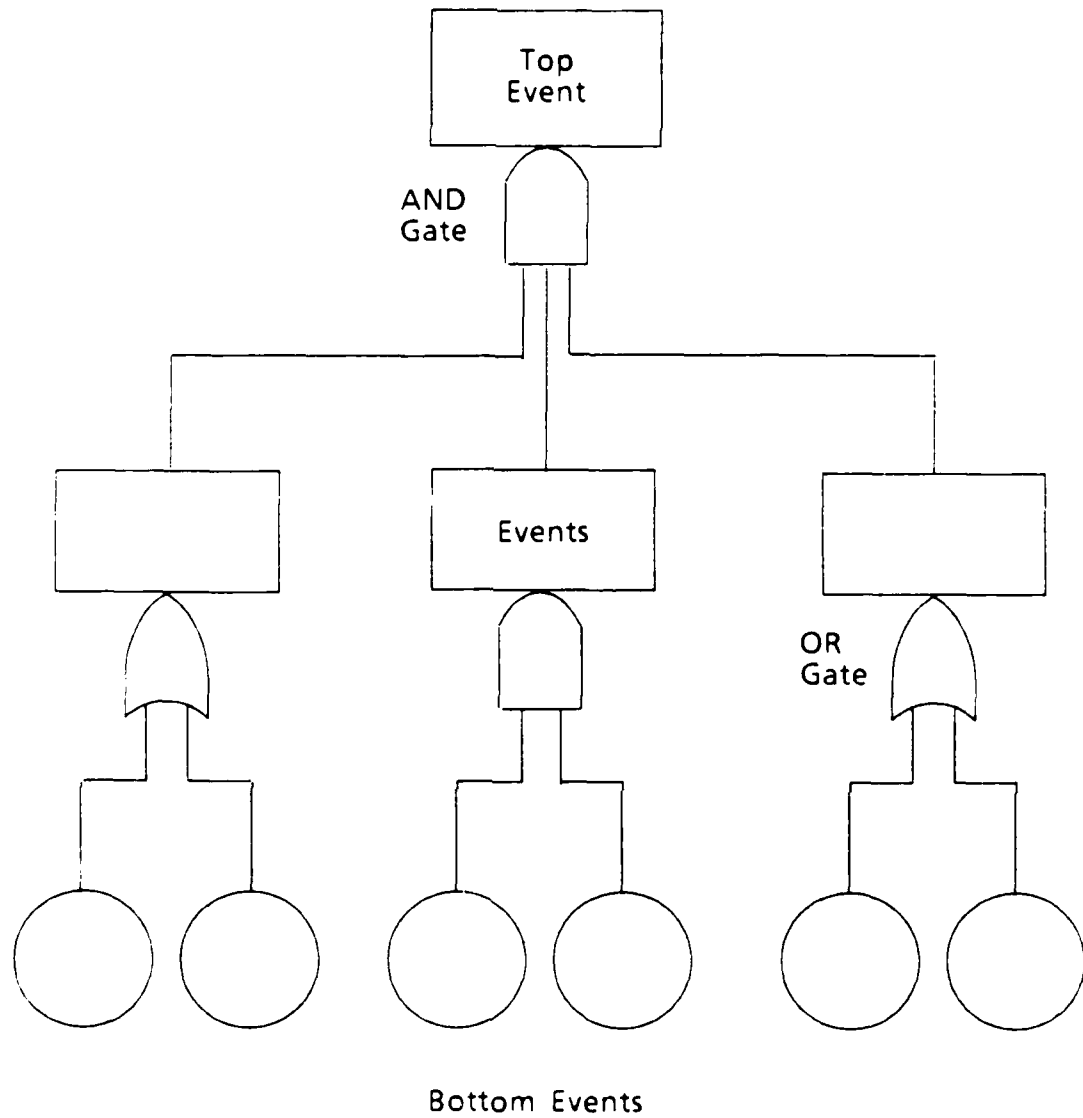


Figure 3 1 Example of a Fault Tree

C. ASSUMPTIONS

In selecting fault tree analysis as the analysis tool, some assumptions had to be made. Fault tree analysis requires each component to be either in a go or no-go status. Typically, a spacecraft has functional states which are considered as degraded. During the design of ORION, subsystems were engineered for more than just their design envelope. An example is the propulsion system. More fuel than an extreme mission profile would require is designed into ORION. As such, a true degradation will exist in the working environment (i.e. fuel is used throughout the mission and its tank is not always full), and the propulsion system is considered to either work or not work.

System components are assumed to have statistically independent lives. No component can be repaired or replaced, and each component has a finite life. [Ref. 6: p. 10] As with the components, only two states of the system are recognized, functioning or failed. It is assumed throughout this thesis that the state of the system (i.e. functioning or failed) is completely determined by the states of its components.

Each component will be tested prior to installation and again after installation to insure the system functions properly. The total test time for every component will be at least 500 hours. During these tests the components will have an opportunity to fail and be replaced. If after all the tests the component is still functioning, it is assumed it will face a constant failure rate during its mission life. This assumption means the exponential distribution will be used in determining a component's survival probability.

The physical structure of the satellite will undergo stresses and strains. Throughout the analysis it is assumed the satellite will not be stressed

outside of its design envelope. This means no component will experience loads greater than or equal to its elastic limit. Additionally, no part will experience fatigue failure due to cyclic mechanical or thermal stress loading. It is also assumed the shared stress environment creates associated components. The concept of association will be addressed later.

All basic events are assumed to be relevant to the event tree. This means each basic event appears in the union of the min cut sets. A formal definition of relevant components is presented in Section J of this chapter.

D. ADVANTAGES OF FAULT TREES

There are some distinct advantages of fault tree analysis that make it particularly suited for the reliability analysis of ORION. These advantages include:

1. the clarity of subsystem interrelation is expressed by the tree.
2. the fact that the tree can be quantified.
3. enabling the analyst to focus on one particular undesired event at a time.
4. for constructing meaningful fault trees, the analyst has to interact with the designers and operators to fully understand the system. The insight obtained during this process is of major benefit to system design, since weaknesses are spotted and corrected during this period.
5. the graphical representation of the logic structure provides a visual tool to both the engineers and management and is useful for justifying design changes and performing trade off studies.
6. the fault tree, being in essence a top-down failure mode and effect analysis, lends itself to

better organization and control than the conventional failure mode and effect analysis. Because of the top-down approach, it also offers more flexibility in terms of termination at any hardware level as well as selectively exploring certain critical faults in greater depth.

7. the fault tree can be used to obtain minimal cut sets which define the modes of system failure and identify critical components. [Ref. 7] Minimal cut sets are addressed in paragraph G of this chapter.

E. DISADVANTAGES OF FAULT TREES

Though there are some general drawbacks to fault tree analysis, these shortcomings do not adversely affect the analysis of ORION. Fault tree analysis can be time consuming, expensive to produce, and include overwhelming detail for large or complex systems. Since ORION is to be a low cost, multi-purpose bus, a fault tree analysis is not necessarily complex or time consuming. Another general drawback is it requires considerable effort to include all types of common cause failures in the fault tree. A fault tree cannot readily handle priority AND gates and elements in cold standby. A priority AND gate restricts its inputs to a specified sequence. ORION has no feature requiring a priority AND gate and has no component in cold standby.

F. CONSTRUCTION OF A FAULT TREE

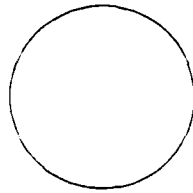
There are three groups of symbols commonly used to construct a fault tree. The three groups presented here are the events, the logic gates and some special symbols.

1. Events

Four kinds of events are represented by the four symbols in Figure 3.2. A circle represents a clearly defined failure of a basic component. In contrast to the exactness represented by the circle is the uncertainty associated with a diamond event, which is a failure not well understood because of absence of information or significance. Circles are called primary events and diamonds secondary events. Collectively they are called bottom events. As such, they are on the bottom of the tree, have reliabilities associated with them, and represent the depth of resolution. Normal, frequently occurring events are symbolized by a house-shaped figure. An example is the satellite being eclipsed by the earth. Without sunlight the solar panels will not generate a voltage. Though no voltage is considered a failure, this condition is not the result of a broken panel. Finally, several events combined together by a logic gate form a combination event represented by a rectangle. Rectangles are called gate events. Gate nodes correspond to intermediate events while the top node corresponds to a very serious system failure event.

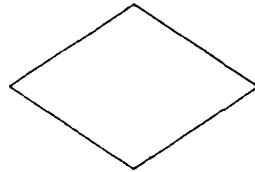
2. Logic Gates

Many different logic gates are used to combine events, but three simple ones are sufficient. These three (AND, OR, and INHIBIT) are illustrated in Figure 3.3. Note that the inputs enter from below and the output comes from the top of the gate. The AND gate produces an output if all the inputs exist simultaneously. The OR gate produces an output when at least one of the input conditions occur. These two gates are the same as ordinary usage of the words "and" and "or." The INHIBIT gate produces output when the input is present and a specified condition exists. In



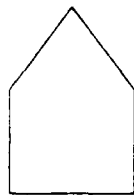
Circle

Basic component failure



Diamond

Failure undeveloped due to lack of information or lack of significance



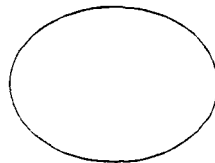
House

Normally occurring event
probability close to one



Rectangle

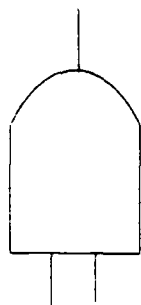
Combination of other three events
does not appear at lowest level
of tree



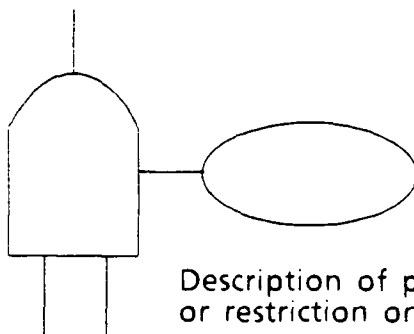
Ellipse

Priority description or restriction
placed on the gate or an
indicator of multiple components

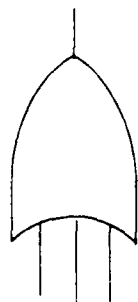
Figure 3.2 Events



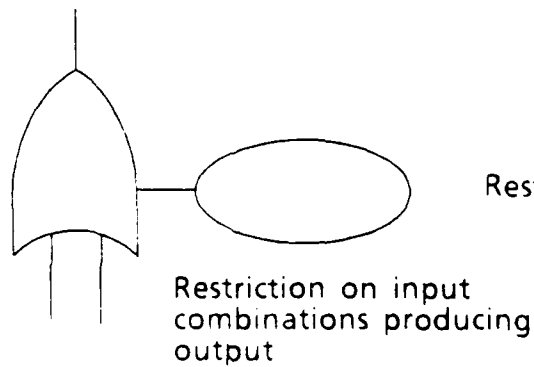
AND Gate



Priority AND Gate



OR Gate



Restricted OR Gate

Figure 3.3 Logic Gates

words, the output is "inhibited" by lack of the stated condition. The INHIBIT gate can be compared to FORTRAN's logical IF statement. The FORTRAN statement "IF (A .EQ. B) GOTO 1030" states that if the condition A equals B is satisfied, go to statement number 1030. If the condition is not satisfied, continue in normal sequence.

3. Special Symbols

Shown in Figure 3.4 are three special symbols representing parts of trees used to reduce redundancy. These comprise the last set of symbols presented for construction of a fault tree.

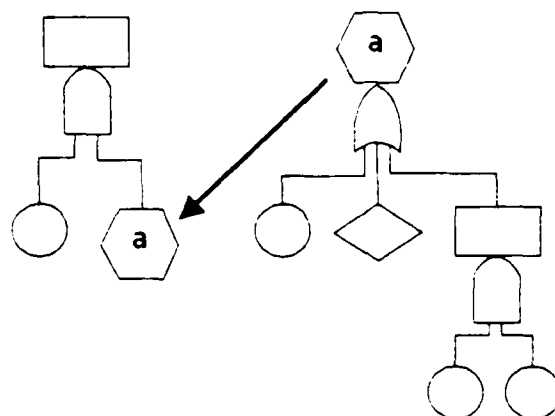
The hexagon refers to another fault tree which is substituted where the symbol appears. A good use for this symbol would be when a particular failure needs further definition. The detailed tree would be headed with another hexagon and bear the same label as the hexagon in the original tree.

To repeat another portion of the same tree, a pair of triangles is used. The portion of the tree below the triangle on the left is substituted at the point where the triangle appears on the right.

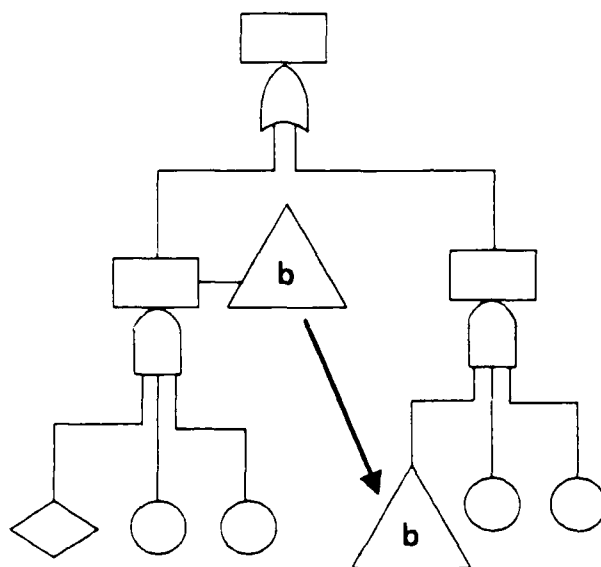
The last special symbol (an ellipse) indicates identical components either in series or parallel. In this case only one component is mentioned and the redundancy is shown by an ellipse around the input. The number of components is written beside the symbol.

G. MINIMAL CUT SETS

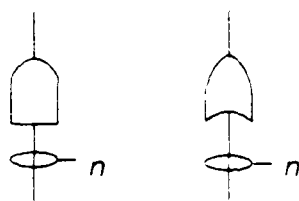
A listing of minimal cut sets (or min cut sets or MCS) is useful for design purposes by helping to determine the "weakest link(s)" in the system. A cut set is defined as any set of primary and secondary events whose occurrences cause the top event to occur.



HEXAGON
To repeat a
separate tree



TRIANGLE
To repeat a portion
of the same tree



ELLIPSE
To indicate n
identical components

Figure 3.4 Special Symbols

A cut set is minimal if it cannot be reduced and still ensure the occurrence of the top event.

The algorithm used to identify min cut sets is based on the fact that AND gates always increase the size of a cut set while an OR gate always increases the number of cut sets.

The simplest and clearest way to explain the min cut set algorithm is to illustrate its operation in an example. The event tree for Example 3.1 is Figure 3.5.

Example 3.1:

The algorithm begins with the gate immediately below the top event. If the gate is an OR gate, each input is an entry in separate rows of a list matrix. If the gate is an AND gate, each input is listed in the first row of a list matrix. Since the gate immediately below the top event in Figure 3.5 is an OR gate, the construction of the list matrix begins with inputs 1, G1, and 2 in separate rows as follows:

$$\begin{matrix} 1 \\ G1 \\ 2 \end{matrix}$$

Since any one of the inputs can cause the top event to occur, each will be a member of a separate cut set.

The idea of the algorithm is to replace each gate by its input gates and basic events until a list matrix is constructed, all of whose entries are basic events. The rows will then correspond to cut sets.

Since G1 is an OR gate, G1 is replaced by its input events in separate rows as follows:

$$\begin{matrix} 1 \\ G2 \\ 3 \\ 2 \end{matrix}$$

Likewise, G2 is replaced by its input events in separate rows.

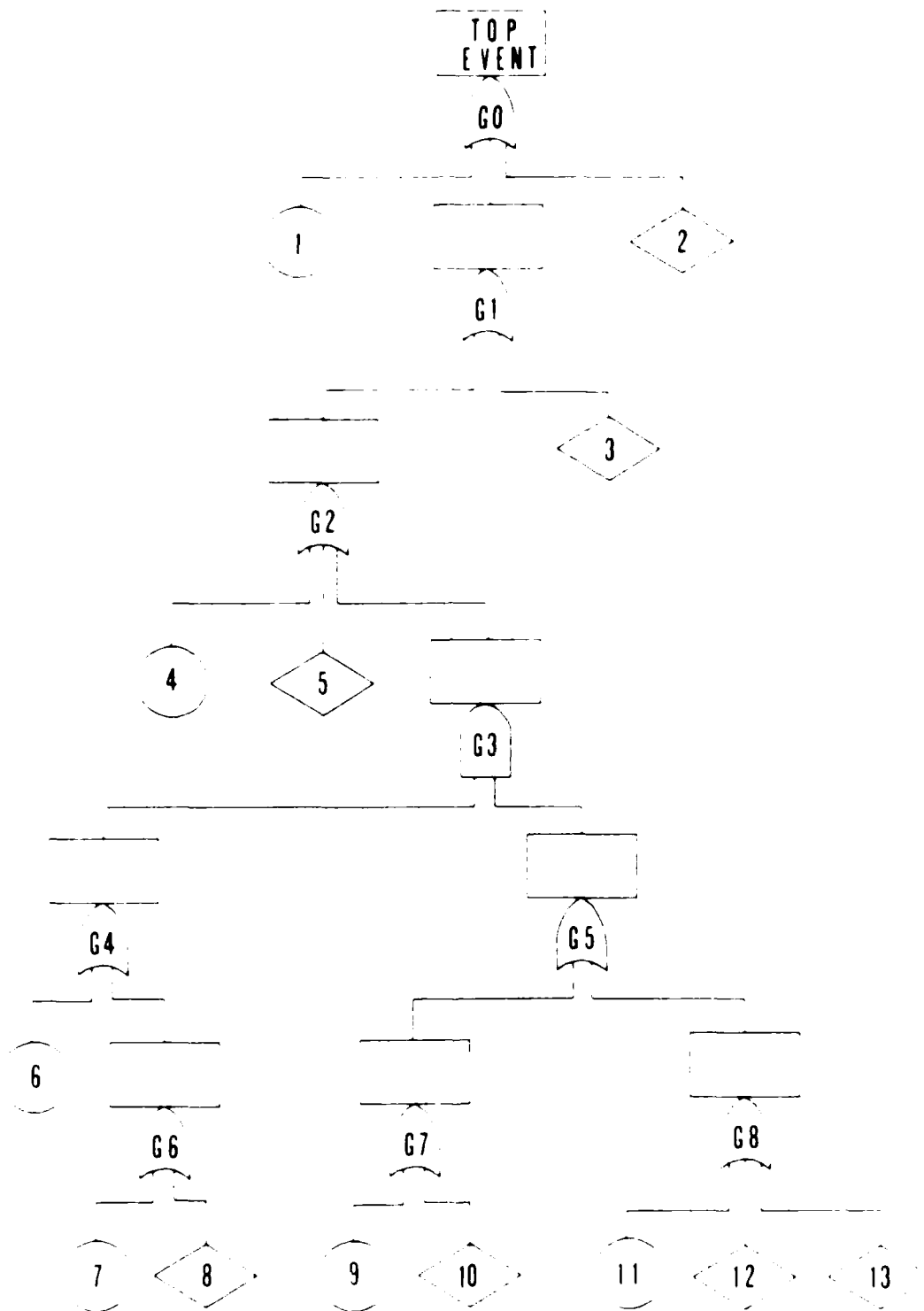


Figure 3.5 Fault Tree for Example 3.1

1
4
5
G3
3
2

Since all inputs to an AND gate must occur to cause the intermediate event above the AND gate, this shows that an AND gate increases the length of its row. An OR gate, on the other hand, increases the number of rows in the list matrix.

Replacing G3 (which is an AND gate) by its inputs, the list matrix becomes:

1
4
5
G4, G5
3
2

Replacing G4 by its inputs, the list becomes:

1
4
5
6, G5
G6, G5
3
2

Continuing until the list contains only primary or secondary events the list stops with these (rearranged cut sets):

1	6,9	7,9	8,9
2	6,10	7,10	8,10
3	6,11	7,11	8,11
4	6,12	7,12	8,12
5	6,13	7,13	8,13

In this example basic events are not repeated. If basic events are not repeated all of the cut sets are minimal cut sets. This means no one cut set is contained in any other cut set. Generally, if basic events are repeated in the tree, the algorithm does not determine only min cut sets. So, when basic events are repeated somewhere in the tree the list matrix must be searched to eliminate cut sets which contain other sets. The final list will then contain only min cut sets.

H. MINIMAL PATH SETS

The dual to a cut set is a path set. Path sets are identified through the dual event tree and consist of the events necessary to make the system function rather than fail. To draw the dual event tree, replace AND gates with OR gates and OR gates with AND gates in the original tree. Each event must also be replaced with a dual description. Failures in the original tree become successes in the dual (new) tree. In general, the dual basic events are the non-occurrence of the original basic events.

As in the cut sets, the focus is on the minimal path sets. A path set is minimal if it cannot be further reduced and still insure the top event (now a system success). Min path sets are determined by applying the same min cut algorithm to the dual (new) tree.

I. PROBABILITY EVALUATION OF FAULT TREES

To build the mathematical structure necessary to derive system reliabilities the states of a component must first be defined. To indicate the state of the i th component a binary indicator variable x_i is assigned to component i :

$$x_i = \begin{cases} 1 & \text{if component } i \text{ is functioning} \\ 0 & \text{if component } i \text{ is not functioning} \end{cases}$$

where $i = 1, \dots, n$, and n is the number of components in the system. Additionally, a binary variable indicates the functioning of the system:

$$X = \begin{cases} 1 & \text{if the system is functioning} \\ 0 & \text{if the system is not functioning} \end{cases}$$

Since it is assumed that the state of the components completely determines the state of the system the system state can be represented as

$$X = f(x)$$

where

$$x_i = \begin{cases} 1 & \text{if component } i \text{ is functioning} \\ 0 & \text{if component } i \text{ is failed} \end{cases}$$

The function $\phi(\mathbf{x})$ is called the structure function of the system. The number of components (n) in the system is called the order of the system. As an example, the structure function of a series of n components is

$$\phi(\mathbf{x}) = \prod_{i=1}^n x_i = x_1 x_2 \dots x_n$$

Consistent with above, $\phi(\mathbf{x})$ is 1 only if all the components function.

Similarly, for a parallel arrangement of n components, the structure function becomes

$$\phi(\mathbf{x}) = \prod_{i=1}^n (1 - x_i) = (1 - x_1)(1 - x_2) \dots (1 - x_n)$$

or equivalently

$$\prod_{i=1}^n (1 - x_i) = 1 - \prod_{i=1}^n x_i$$

This returns a value of 1 if there is at least one functioning component ($x_i = 1$). Both notations are consistent with their respective usages in logic.

A k -out-of- n structure functions if and only if at least k of the n components function. This structure function is shown by

$$\phi(\mathbf{x}) = \sum_{i=1}^n \sum_{j=1}^n \dots \sum_{k=1}^k x_{i_1} x_{i_2} \dots x_{i_k}$$

Fault trees with AND and OR gates create structure

functions which are coherent². Then given a coherent structure (Φ) of order n

$$\prod_{i=1}^n x_i \leq \Phi(x) \leq \prod_{i=1}^n x_i.$$

This means a system's performance is bounded below by a series representation and above by a parallel representation. [Ref. 8: pp. 6-8]

With the j^{th} ($j = 1, \dots, p$) min path set P_j , we may express a structure (called the minimal path series structure) with arguments $x_1, \dots, x_n$:

$$\Phi(x) = \prod_{j=1}^p \left(\prod_{i \in P_j} x_i \right).$$

The structure is binary and takes on the value 1 if all the components in the j^{th} min path set function. This expression depicts a path set as a series arrangement of the path set's elements. A system will function when at least one min path set functions. The structure function can then be written as

$$\Phi(x) = \prod_{j=1}^p \left(\prod_{i \in P_j} x_i \right) = \prod_{j=1}^p \left(\prod_{i \in P_j} x_i \right).$$

This means the structure function can be viewed as a parallel arrangement of the path sets. This is commonly referred to as a parallel-series arrangement.

Similarly, with minimal cut sets, the structure (called the minimal parallel cut structure) can be expressed with arguments $x_1, \dots, x_n$:

$$\Phi(x) = \prod_{j=1}^p \left(\prod_{i \in P_j} x_i \right)$$

²A coherent structure being, roughly, one whose performance does not deteriorate when failed components are replaced by functioning ones [Ref. 8: pp. 191,192].

which is binary and takes on the value 0 when all the components in the j^{th} min cut set fail, and 1 otherwise.

Since the system will fail if and only if at least one of the min cut structures fails, the structure function can be viewed as a series arrangement of the cut sets with the elements of a cut set arranged in parallel. Such an arrangement can be expressed as

$$\phi(x) = \prod_{j=1}^k K_j(x).$$

This is referred to as a series-parallel arrangement.

Initially, the components are assumed to be statistically independent. If the state of the i^{th} component is random (denoted as X_i) then

$$P[X_i = 1] = p_i = E[X_i] \text{ for } i = 1, \dots, n$$

where $E[X]$ means the expected value of X . The probability that i functions, p_i , is referred to as the reliability of component i . In similar fashion, the reliability of the system is

$$P[\phi(X) = 1] = r = E[\phi(X)].$$

The reliability of the k -out-of- n case with identical components and reliabilities becomes [Ref. 8: pp. 20-21]

$$r = \sum_{j=k}^n \binom{n}{j} p^j (1-p)^{n-j}.$$

The preceding formula holds under the assumption of component independence. In reality, this is not usually

the case. Independence will be replaced with a form of positive dependence. Components can become positively dependent in various ways. For example, if a subsystem has several like-components and one of them fails, the subsystem remains functional because the remaining functioning components share the load. Another way positive dependence is created is when all the components are subjected to the same stress environment. The components of ORION fall in this category. If the reliability of a series arrangement of independent components is calculated, when in fact they are associated³, the resultant reliability will be an underestimate of the true reliability. The opposite holds for parallel systems. [Ref. 8: pp. 29,32]

The following min-max bounds theorem is presented in Reference 8, page 37, along with the theorem's proof.

Let ϕ be a coherent structure. Let $P_1, P_2, \dots, P_p$ be the component min path sets corresponding to ϕ , and let $K_1, K_2, \dots, K_k$ be the component min cut sets corresponding to ϕ . If components are associated, then the following bounds hold:

$$\max_{1 \leq r \leq p} \prod_{i \in P_r} p_i \leq P(\phi(X) = 1) \leq \min_{1 \leq s \leq k} \prod_{i \in K_s} p_i$$

Another, equivalent relationship can be expressed in terms of $q_i = 1 - p_i$. The above bounds now become:

$$\max_{1 \leq r \leq p} \prod_{i \in P_r} q_i \leq P(\phi(X) = 1) \leq E(\phi(X)) \leq \min_{1 \leq s \leq k} \prod_{i \in P_s} q_i$$

J. IMPORTANCE OF BASIC EVENTS

There are two kinds of component importance. The first is structure importance and the second is

³Association is a particular form of positive dependence [Ref. 8: p. 150] which can be a reasonable assumption in modeling ORION.

reliability importance. Before discussing each of these, the concept and definition of relevance must be established. The following definition will be used.

The i^{th} component is irrelevant to the structure ϕ , if ϕ is constant in x_i , that is, $\phi(1_i, x) = \phi(0_i, x)$, $\forall (x_1, \dots, x_n)$.⁴ Otherwise the i^{th} component is relevant to the structure. [Ref. 8: p.4]

The structure importance of a component focuses on whether or not a component changes the structure function from 0 to 1 or from 1 to 0. In essence, the structural importance is concerned with only relevant components. If component i is relevant, then the following property holds,

$$\phi(1_i, x) - \phi(0_i, x) = 1 \quad \text{for some } (x_1, \dots, x_n).$$

When this condition exists $(1_i, x)$ is called a critical path vector for i . Let $n_p(i)$ denote the total number of critical path vectors for i . This means

$$n_p(i) = \sum_{(x_1, \dots, x_n) \in \{0,1\}^n} [\phi(1_i, x) - \phi(0_i, x)].$$

This is also the same total number of critical path sets for i . [Ref. 8:p. 13]

The following is a credible measure of the structural importance of component i :

$$I_p(i) = \frac{1}{2^n - 1} \sum_{(x_1, \dots, x_n) \in \{0,1\}^n} [\phi(1_i, x) - \phi(0_i, x)]$$

This depicts the proportional number of the $2^n - 1$ outcomes which have $x_i = 1$ in the critical path vectors for i . As a result, for any given ϕ , the components

⁴ Notation.

$(1_i, x) \equiv (x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n)$

$(0_i, x) \equiv (x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n)$

$(x_i, x) \equiv (x_1, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_n)$

may be ordered (based on structural importance) by ordering $I_p(1), \dots, I_p(n)$. [Ref. 8:p. 14]

The second type of importance is the component's reliability importance. This takes into account the component reliabilities as well as the system structure. If components can be ranked according to their importance to the system reliability, this ranking information can be helpful in determining which components should have the highest priority for research and development. This allows managers to expend effort and money more wisely. [Ref. 8:p. 26]

Intuitively, it would seem a component's reliability importance could be measured by observing the rate of change in the system's reliability as the component's reliability changes. The reliability importance $I_r(i)$ of component i is given by

$$I_r(i) = E[\phi(1_i, x) - \phi(0_i, x)] .$$

This definition holds even if the components are associated. [Ref. 8: pp. 26-27]

IV. SYSTEM RELIABILITY ANALYSIS

Using a copy of the schematics of ORION (Appendix A) and maintaining a constant interface with the designers, the ORION fault tree was developed (Appendix B). Once the fault tree was established the min cut algorithm was applied to it. This algorithm revealed 82 minimal cut sets. Of these cut sets 22 are single element sets, 29 are double element cut sets, 27 are triple element cut sets, 2 are five element cut sets, 1 is a six element cut set and 1 is an eleven element cut set. Once these cut sets were established, the dual tree was constructed and the min paths determined. There are 33,890,503,680 distinct paths, of which the vast majority is due to the large number of paths through the solar strings. In general, the paths are formed by combining the following components:

- 2 out of 3 attitude detection components
 - sun sensor
 - earth sensor
 - 1 out of 4 magnetometers
- 1 computer
- 4 out of 6 bubble memory cards each with functional heater strips and thermistors
- 1 shunt regulator
- 1 out of 2 batteries
- 14 out of 24 solar strings
- 4 solar connectors
- 3 out of 4 momentum wheels
- 1 out of 2 spin up thrusters with a functional solenoid
- 1 out of 2 spin down thrusters with a functional solenoid

- 1 out of 2 nutation thrusters with a functional solenoid
- 1 orbit insert thruster with a functional solenoid
- 2 pyrotechnic valves
- 2 fill and drain valves
- 2 pressurant tanks
- 1 hydrazine tank with functioning heaters and thermistors
- Hydrazine line intact with functional heaters and thermistors
- 1 out of 2 antennas functioning and deployed
- 1 combiner/splitter in the TT&C
- 1 TT&C transceiver
- 1 TT&C interface hardware
- Pressurant line intact
- 1 heater control hardware
- 1 bubble storage controller
- 1 attitude control interface

If the solar strings are considered as a single module, the number of paths reduces to 17,280. Similar modular reductions can take place when a subsystem consists of k out of n like-components. All but the attitude detection subsystem can be reduced to an equivalent single component. This reduces the final number of paths to three.

The three reduced paths were used to calculate the structural importance of the components. The calculations reveal seven levels of relative importance in the following hierarchy (1 being the most relevant):

1. all basic components except those listed below.
(A detailed list is given in Appendix C);
2. a momentum wheel;
3. a bubble memory card with functioning heaters and thermistors;
4. a solar string;

5. the sun sensor, the earth sensor, a nutation, spin up, and spin down thruster with their functioning solenoids;
6. a battery, an antenna, a hydrazine tank heater and a thermistor; and
7. a magnetometer.

A schematic of the path sets is at Appendix C.

The reliability importance cannot be specifically calculated since the actual hardware for several subsystems has not been defined. A Lotus 1-2-3 spreadsheet was developed so the designers can input component reliabilities as the subsystems are defined. The spreadsheet can then calculate the system's reliability boundaries and components' reliability importance. The data (i.e. component failure rates) for inclusion in the spreadsheet come from two major sources, JPL TR 32-1505 and MILSTD 217D. The spreadsheet identifies the lower boundary as the most reliable path and the upper boundary as the least reliable cut. The number of paths to compare is significantly reduced by using a modular approach (i.e. using the binomial distribution to calculate the reliability of a k out of n subsystem). Such a reduction allows the problem to be handled by a spreadsheet. Even in a reduced form, the model maintains the ability to discern an impact on the system reliability when changing, for example, only a solar string's reliability. The spreadsheet is then singularly important because it can readily do this "what-if" analysis.

V. CONCLUSION

A. OVERALL FINDINGS

Throughout the analysis, it became apparent that the fault tree is a "living" document. It must be maintained to reflect the existing design if it is to aid in the design process. The fault tree can help explain the cause of a failure after design is complete and the system is on station, but only if the fault tree reflects the current design. Aiding in the design, and determination of a failure after system employment are strong motives to maintain the fault tree. This thesis includes sufficient background so maintenance can be done to insure the longevity of the fault tree.

A total of 82 cut sets were determined and the components' structural importance derived. The information can be used to help focus research and budget efforts.

Lastly, a spreadsheet was developed to model the system's reliability boundaries as well as component reliability importance.

B. RECOMMENDATIONS

There are five recommendations based upon the fault tree analysis. They are:

1. as each subsystem is developed, conduct a detailed fault tree analysis of that subsystem.
2. after a subsystem is constructed, conduct a circuit stress analysis of each component and the subsystem.
3. as the design may change, maintain the fault tree.
4. for electrical components, use the designing engineer's reliability based diagram to help

construct the fault tree. If a diagram is not available, request one be made.

5. focus research and budget attention on those components listed with the highest structural and reliability importance.

Due to ORION's design to be low cost and reconfigurable, ORION is an excellent candidate for constellation proliferation. A logical follow-on study to this one would be a study of a constellation's reliability.

APPENDIX A
ORION SUBSYSTEM SCHEMATICS

The enclosed schematics were used to develop the fault tree for ORION.

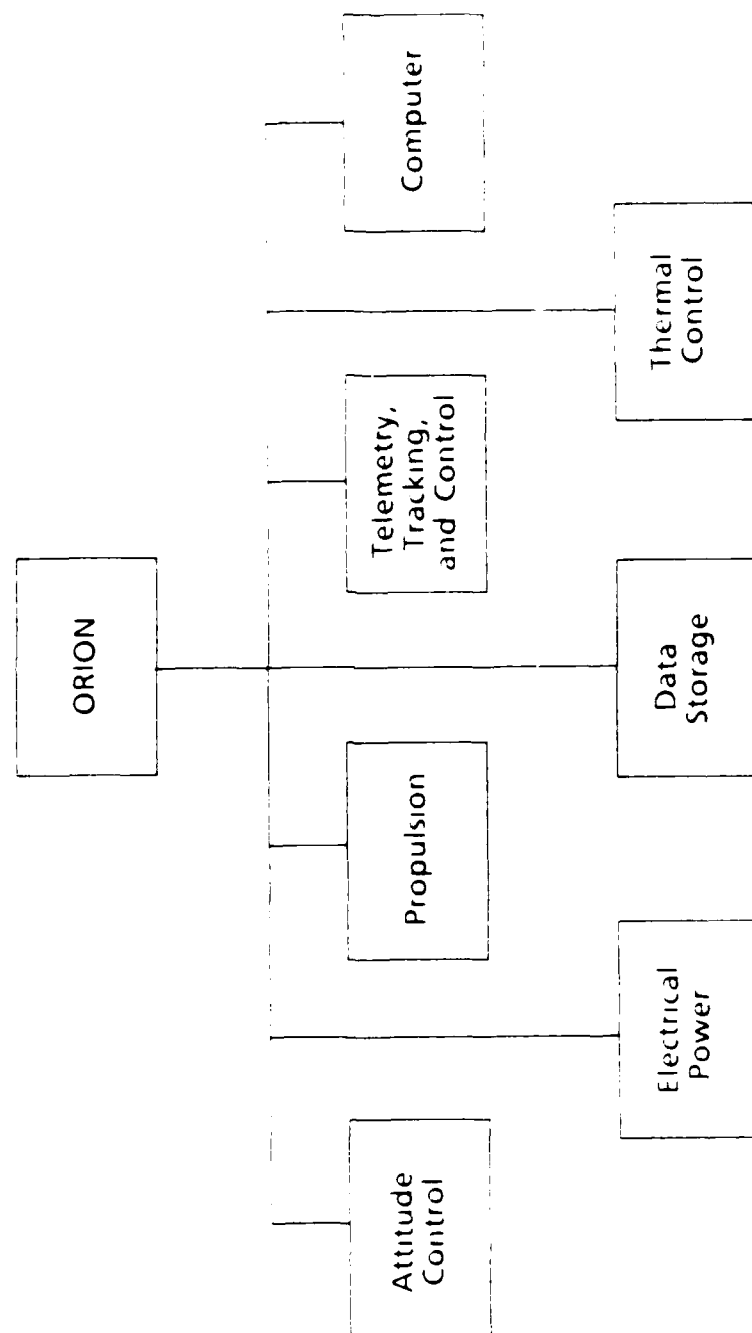


Figure A 1 ORION Subsystem Summary (Management and Design Viewpoint)

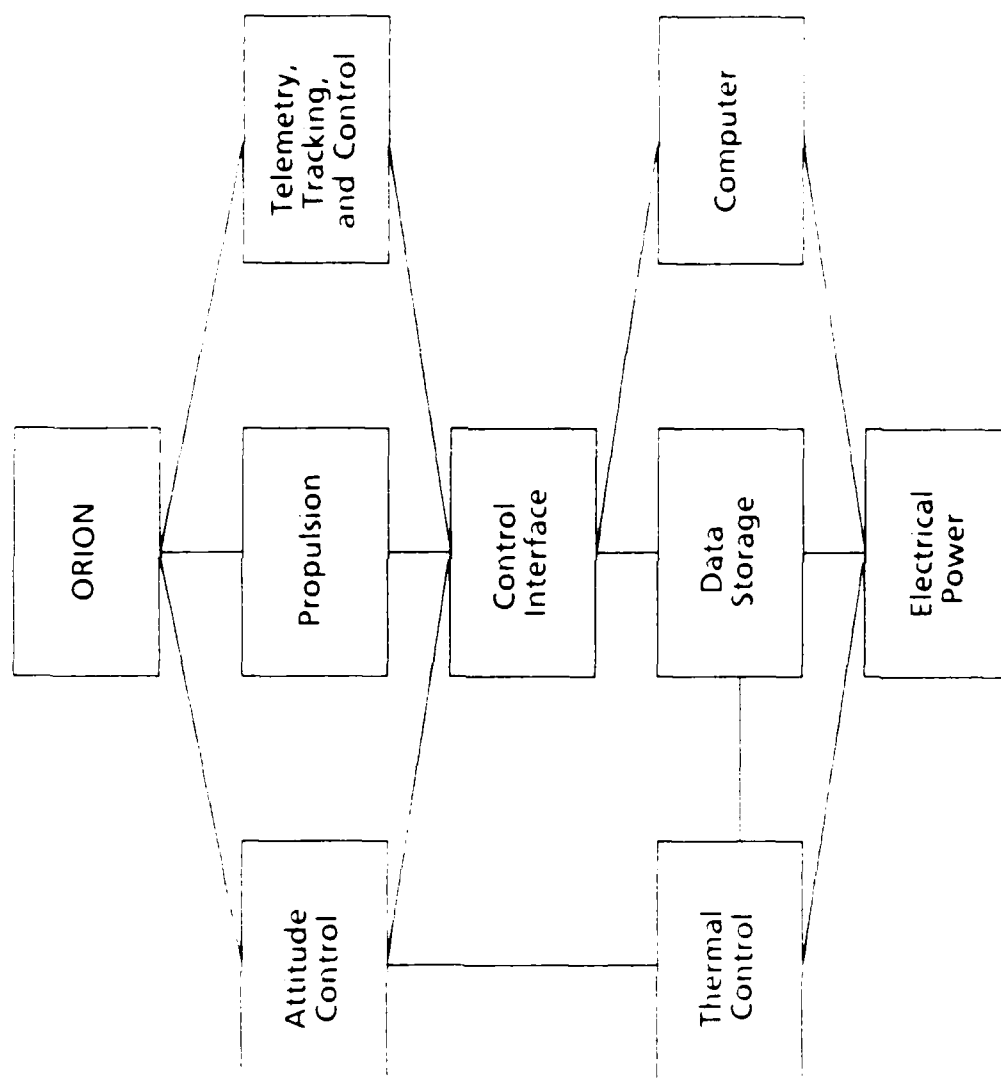


Figure A 2 ORION Subsystem Summary (Reliability Viewpoint)

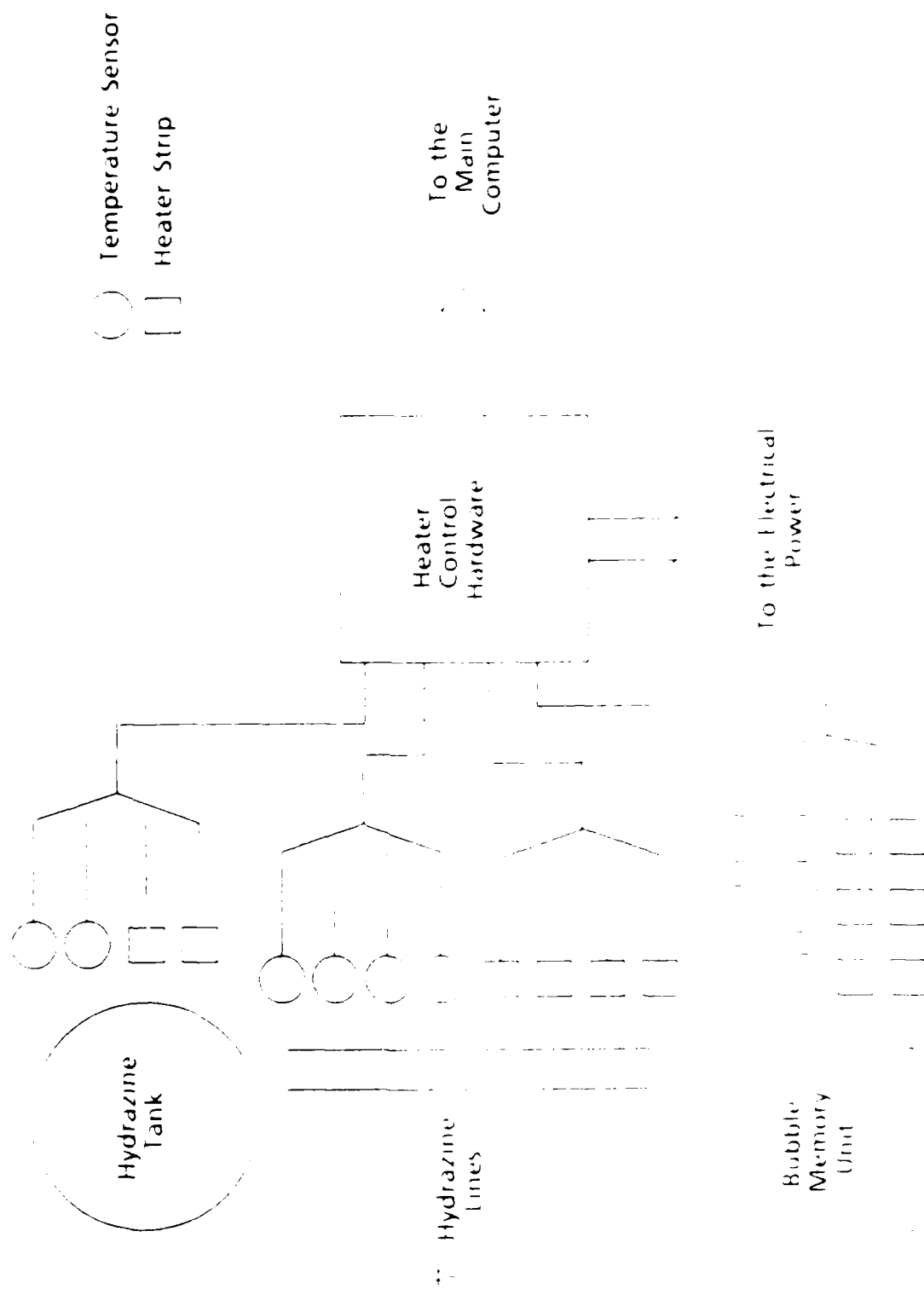


Figure A 3 Thermal Control Subsystem

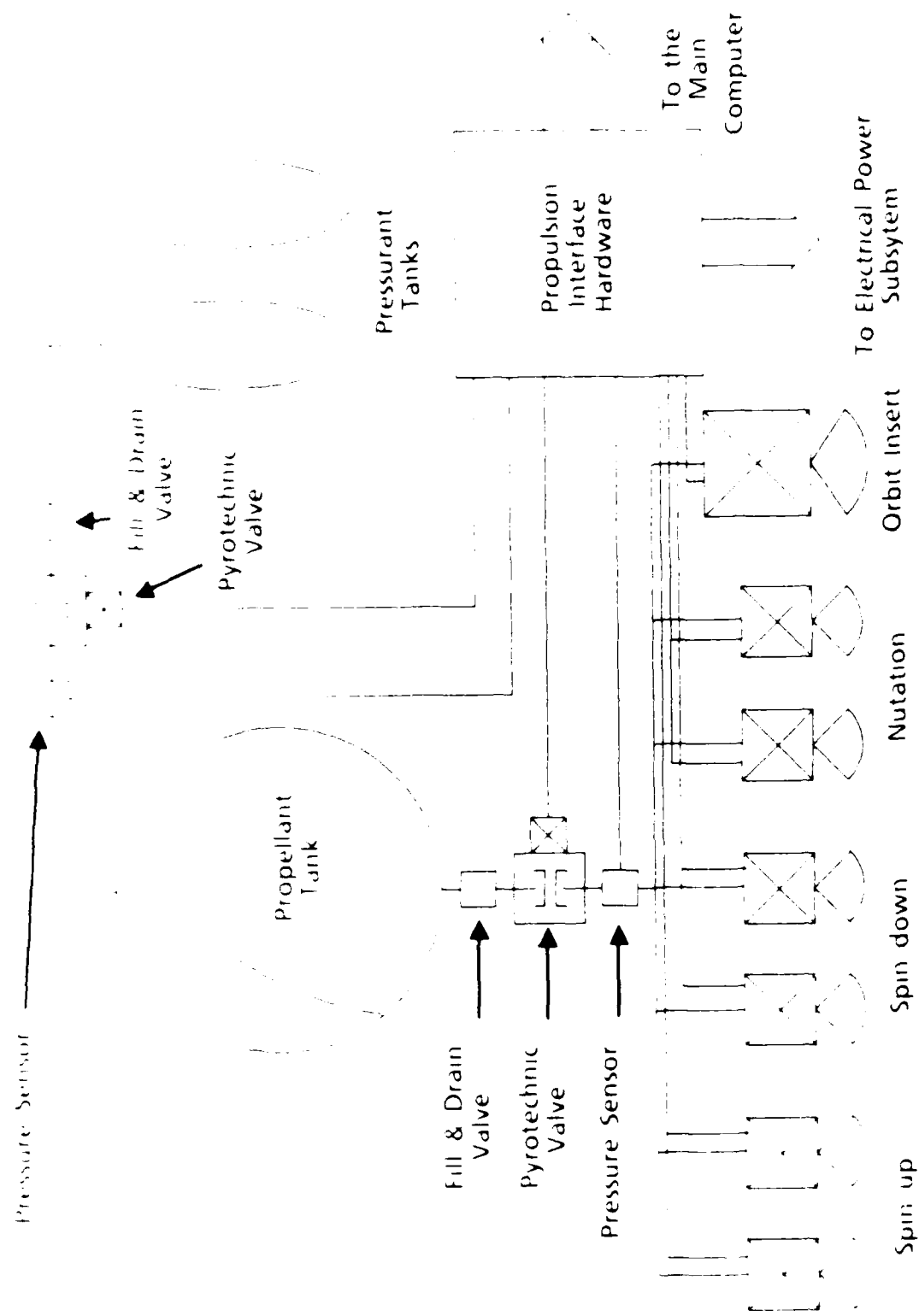


Figure A 4 Propulsion Subsystem

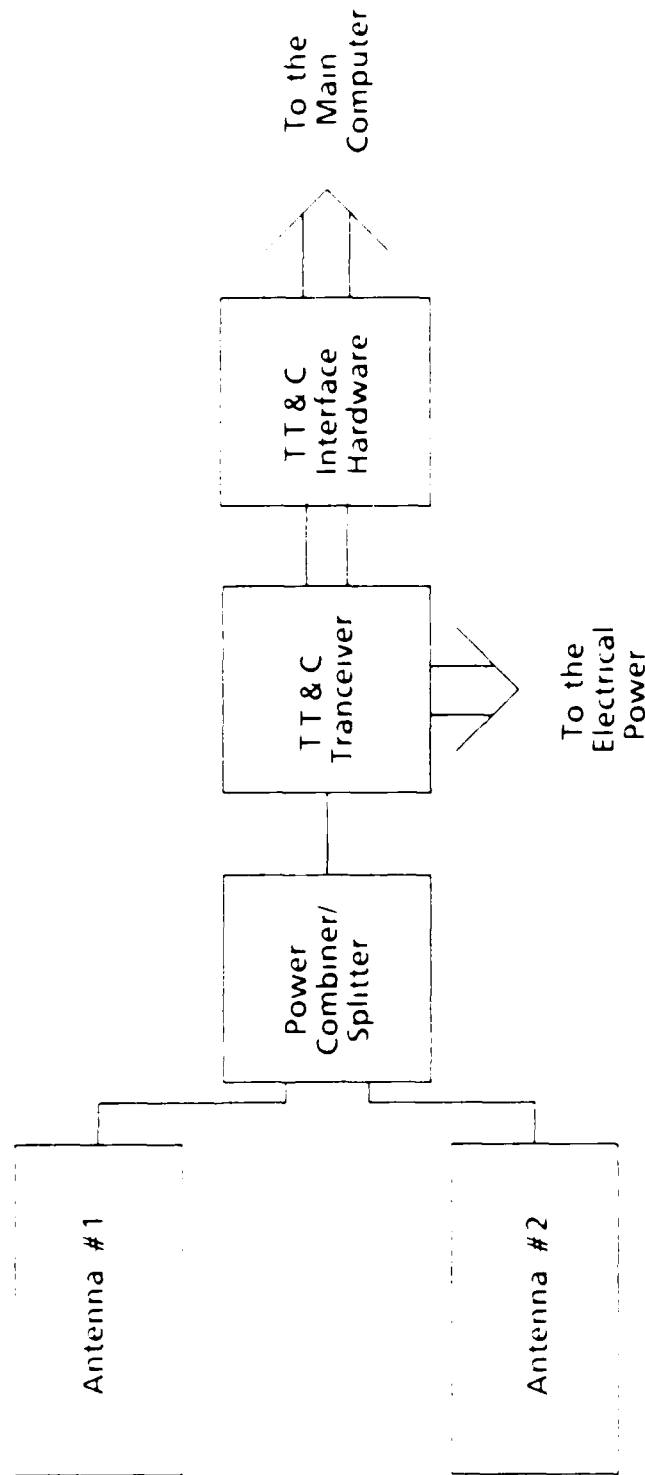


Figure A 5 Telemetry, Tracking and Control Subsystem

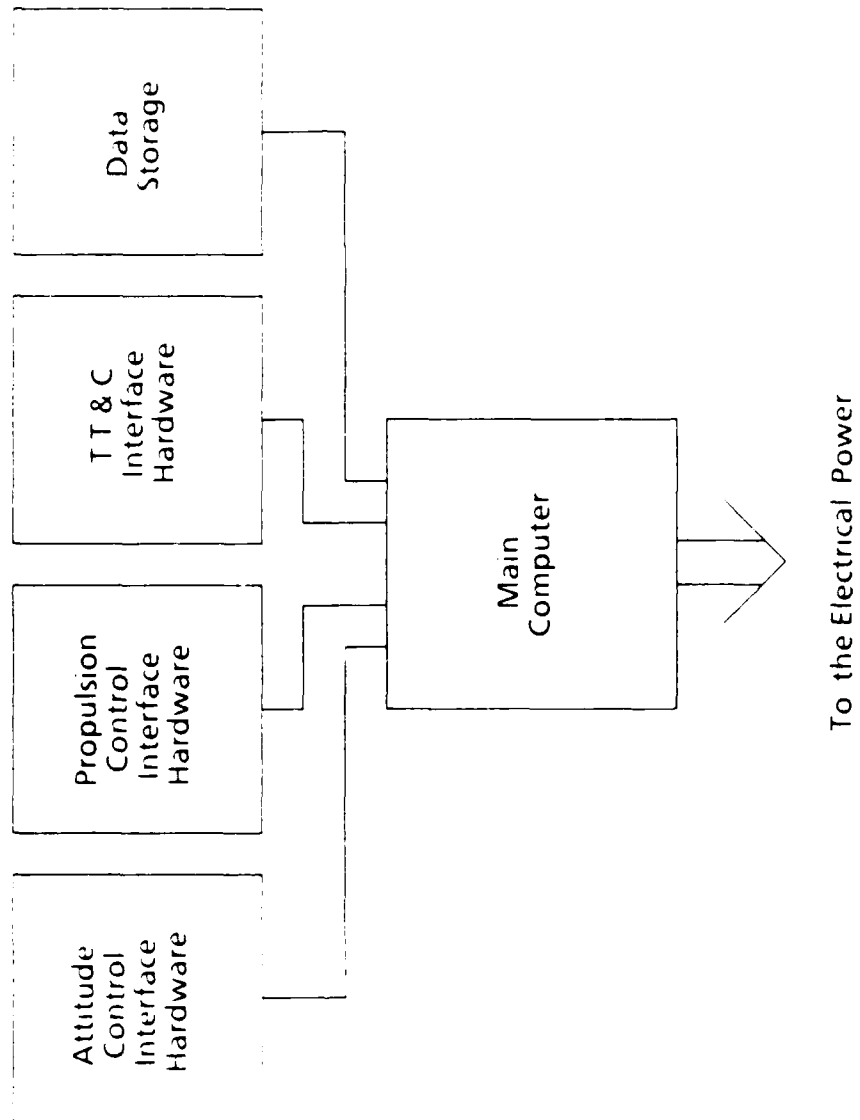


Figure A 6 Computer Subsystem

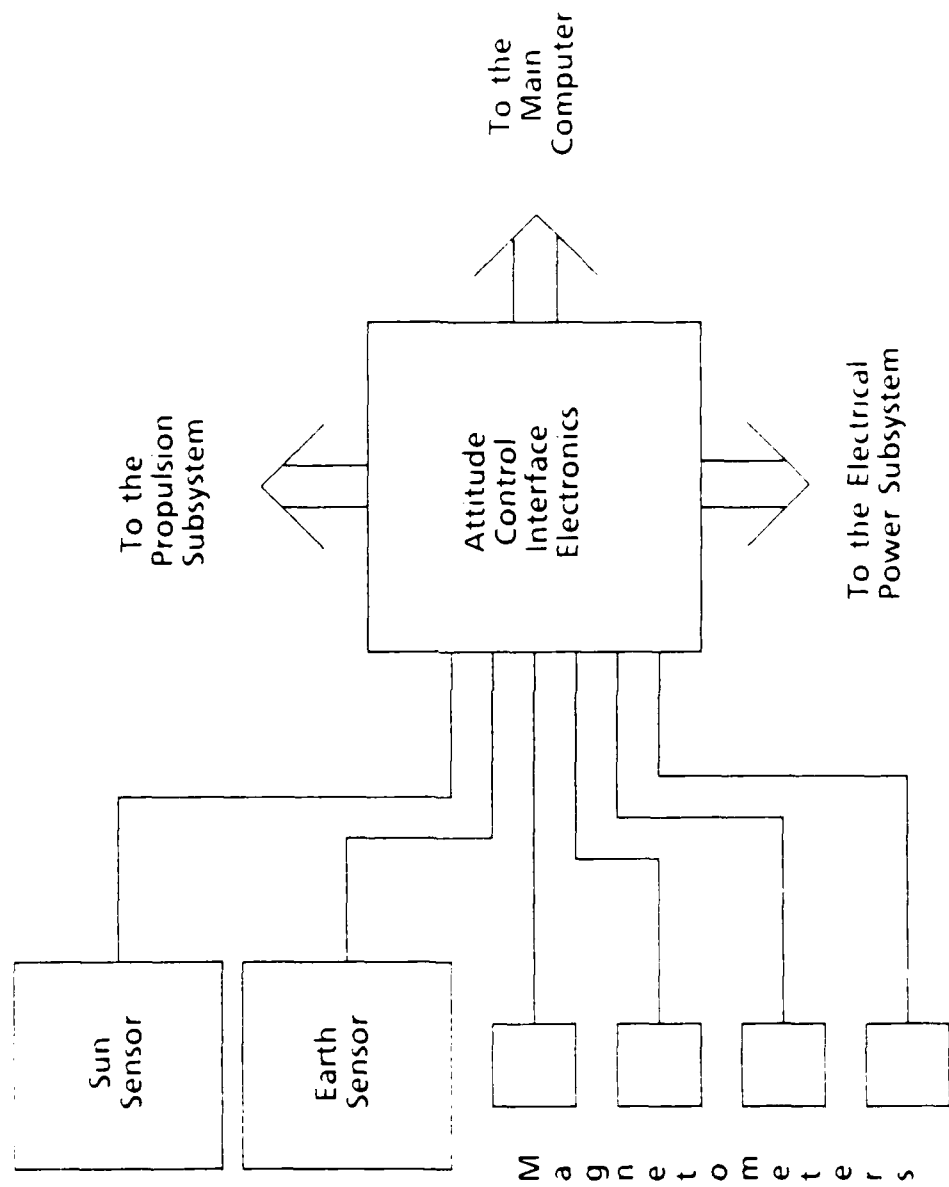


Figure A.7 Attitude Control Subsystem

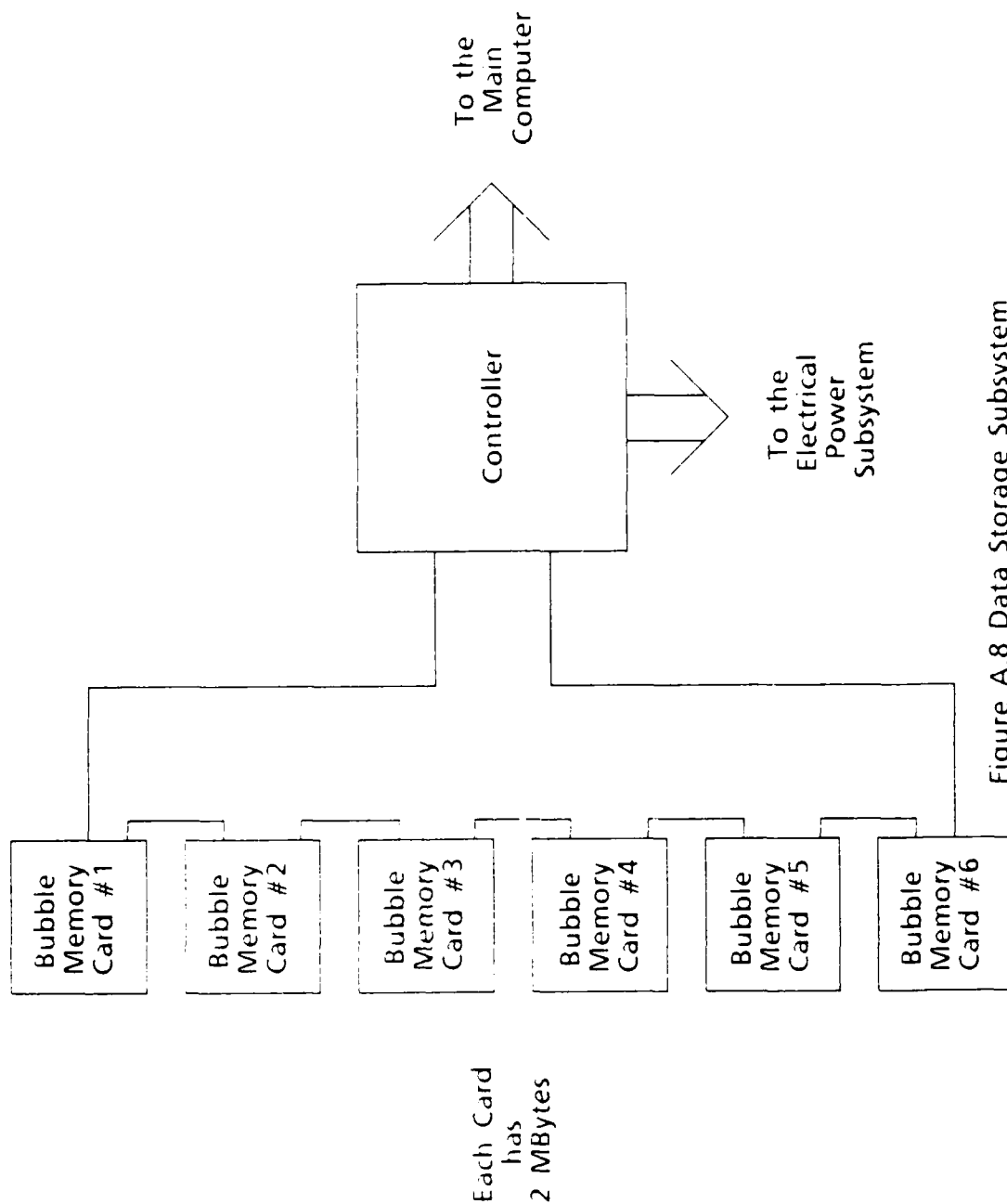


Figure A.8 Data Storage Subsystem

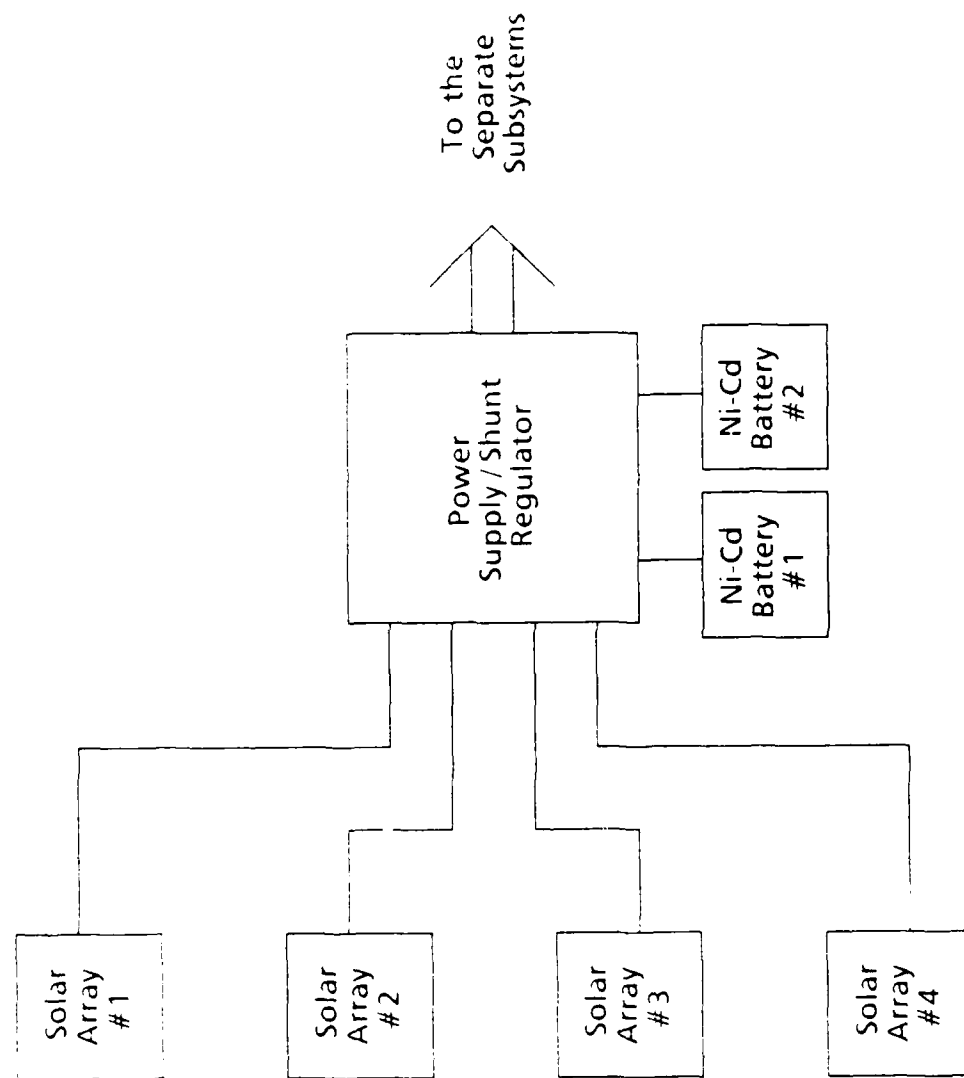


Figure A.9 Electrical Power Subsystem

APPENDIX B
ORION FAULT TREES

The large fault tree developed is broken into small sections and is included in this Appendix.

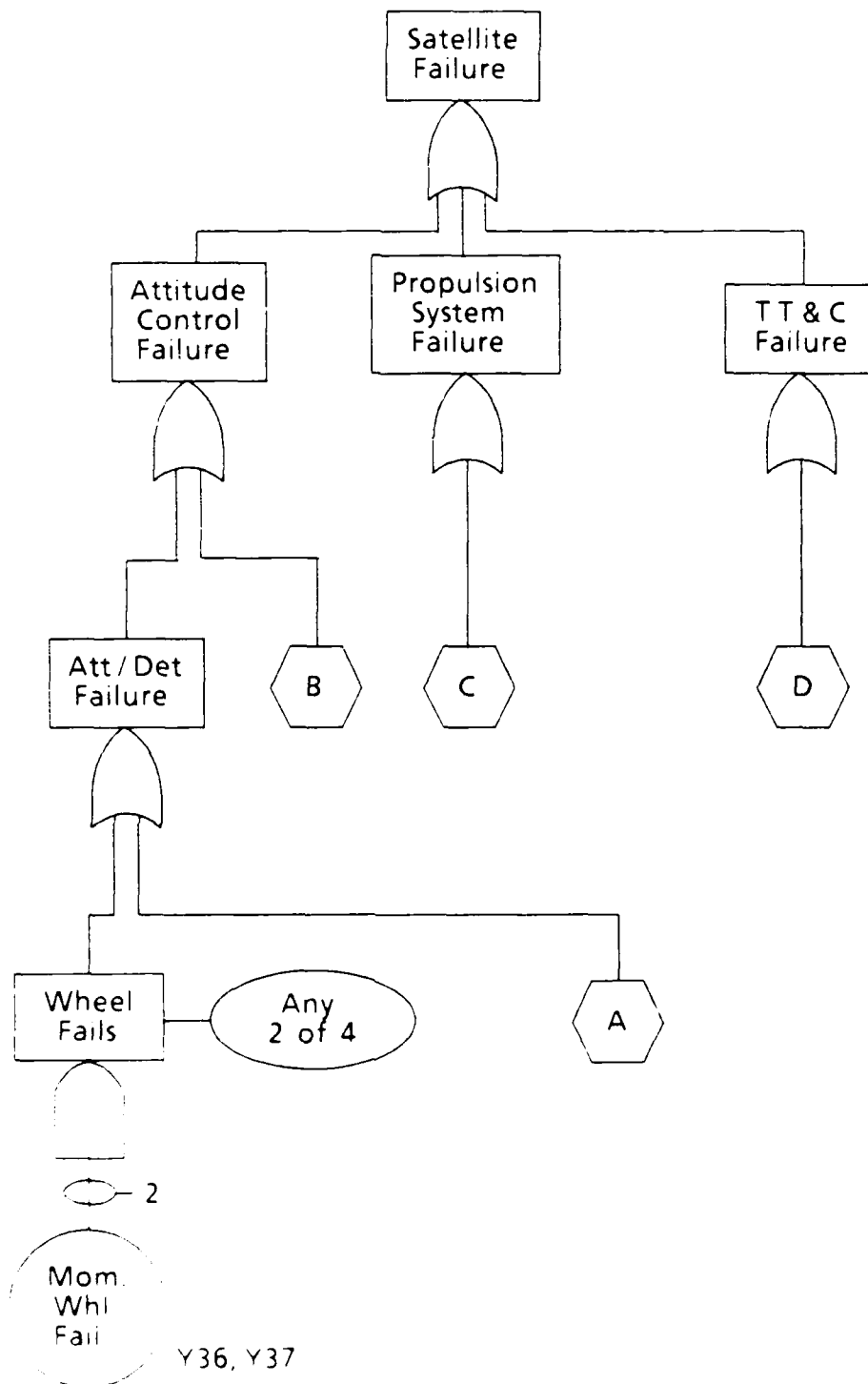


Figure B 1 Top of Fault Tree

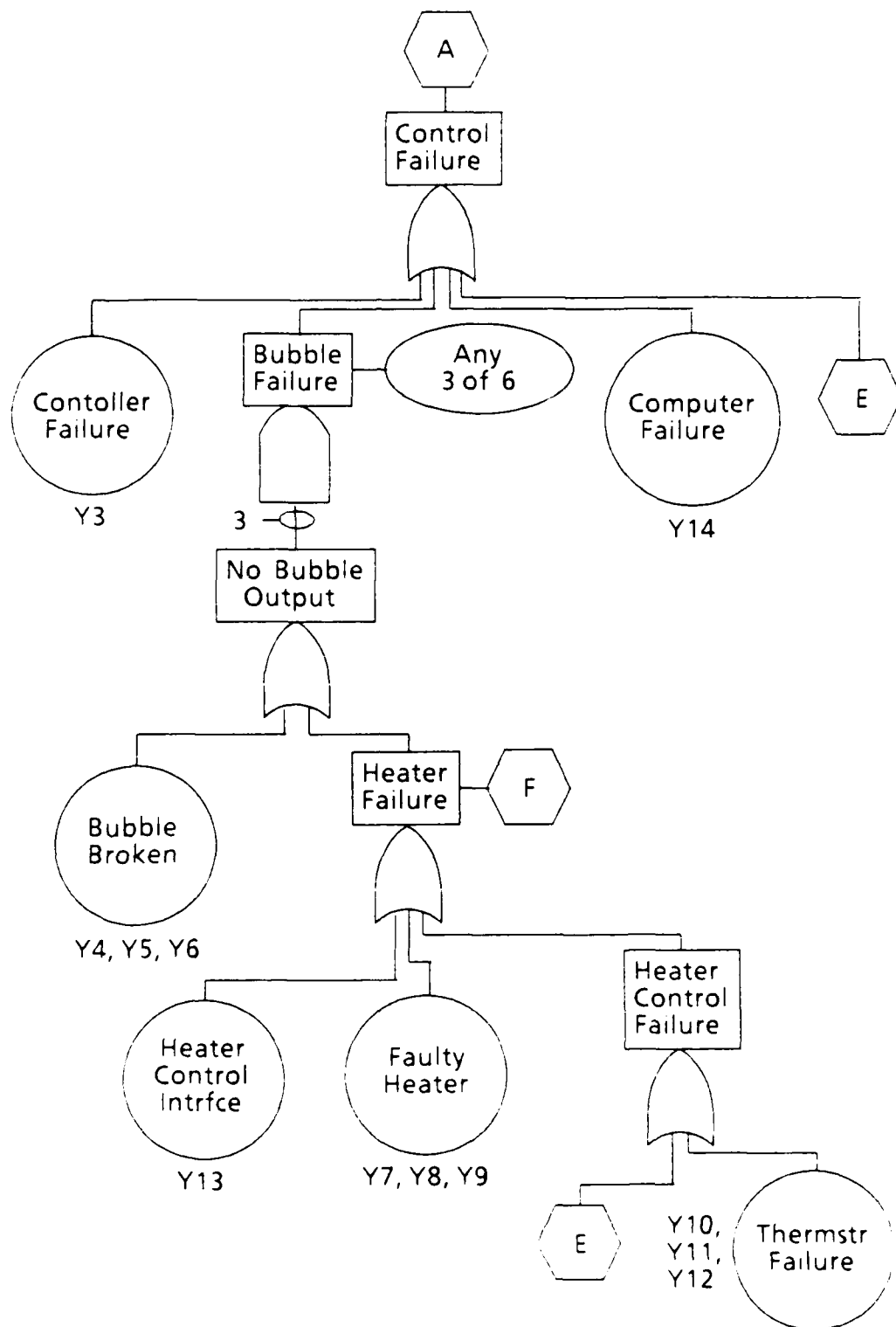


Figure B.2 Control Fault Tree

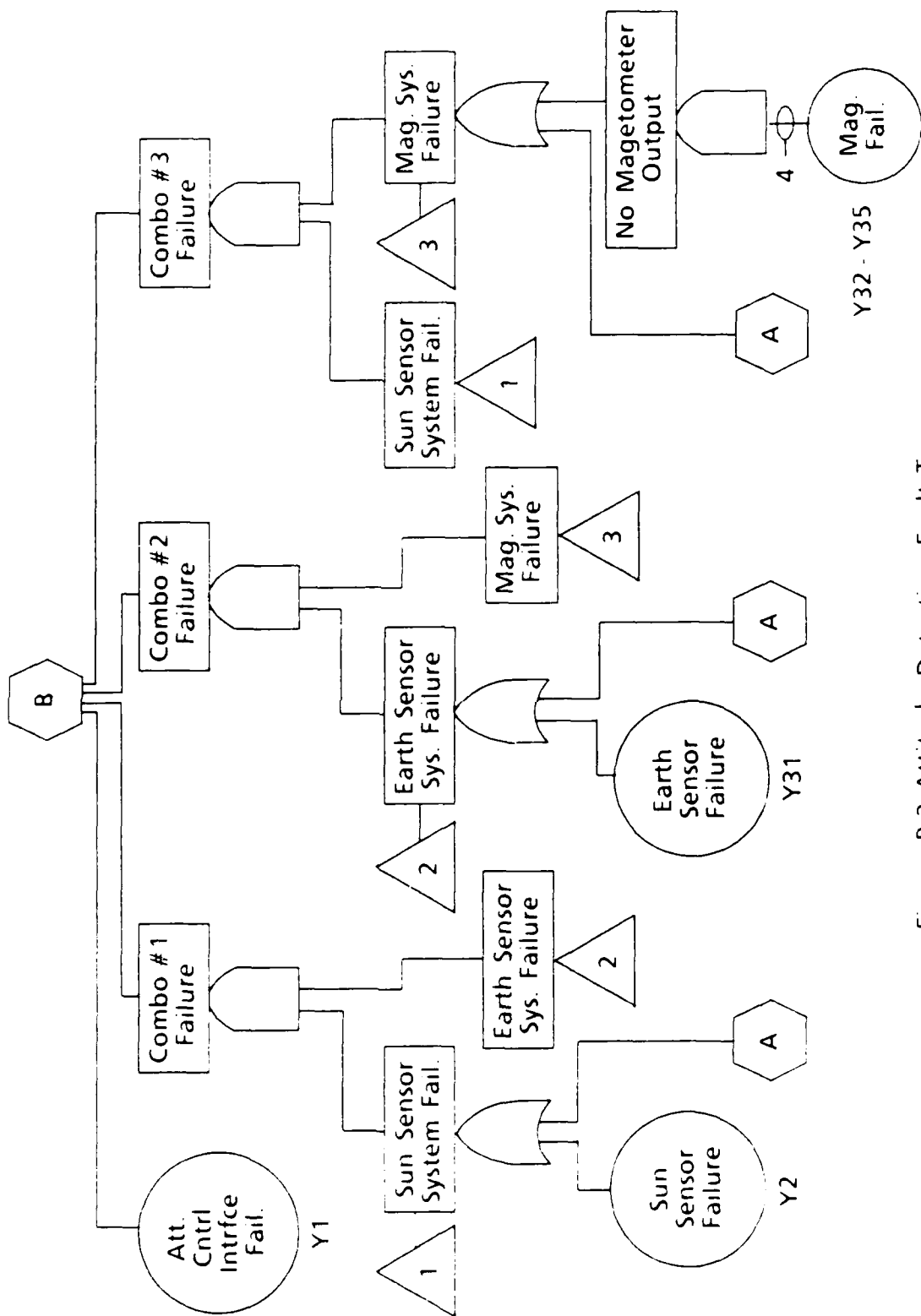


Figure B 3 Attitude Detection Fault Tree

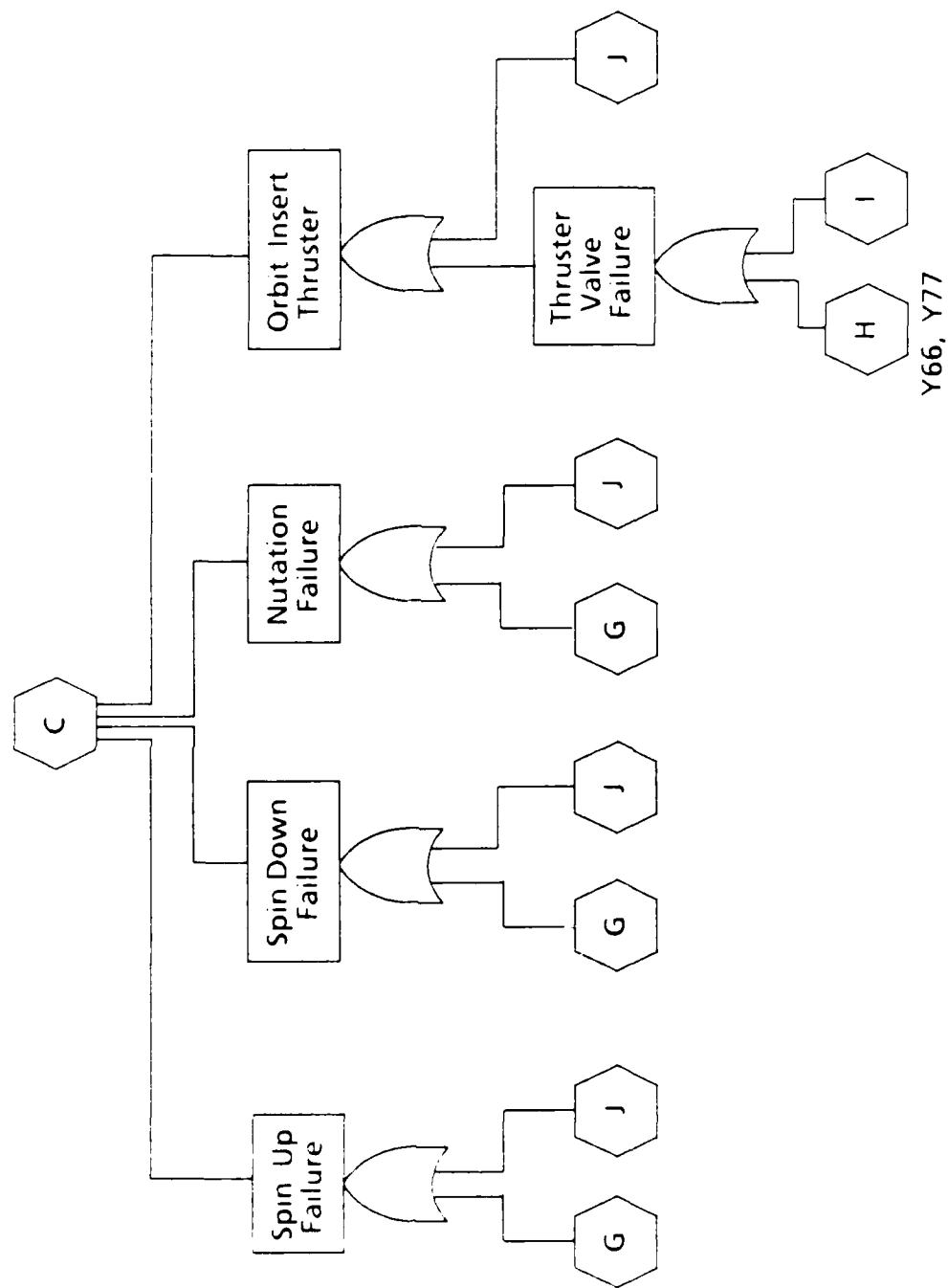


Figure B.4 Propulsion Fault Tree

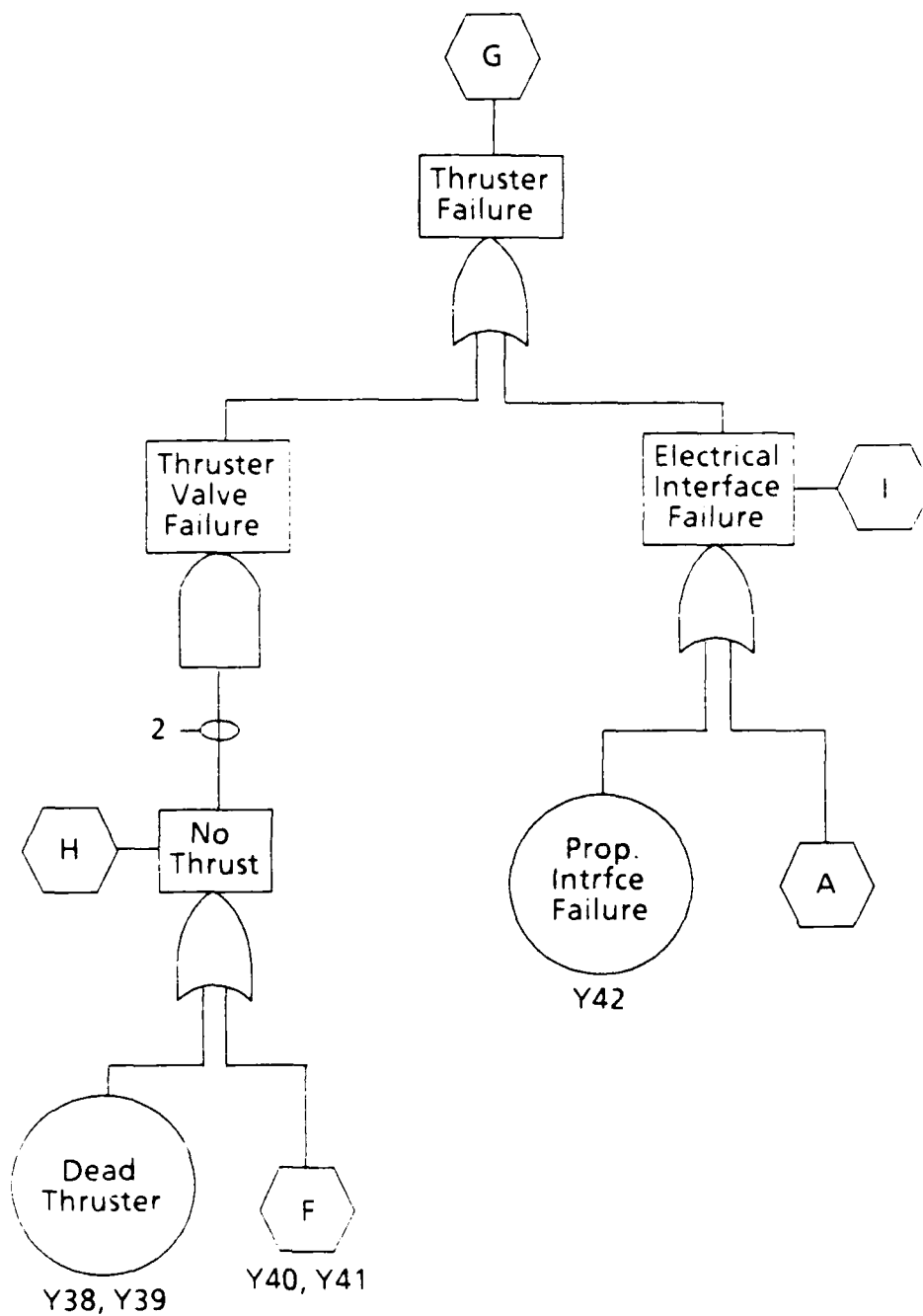


Figure B.5 Thruster Fault Tree

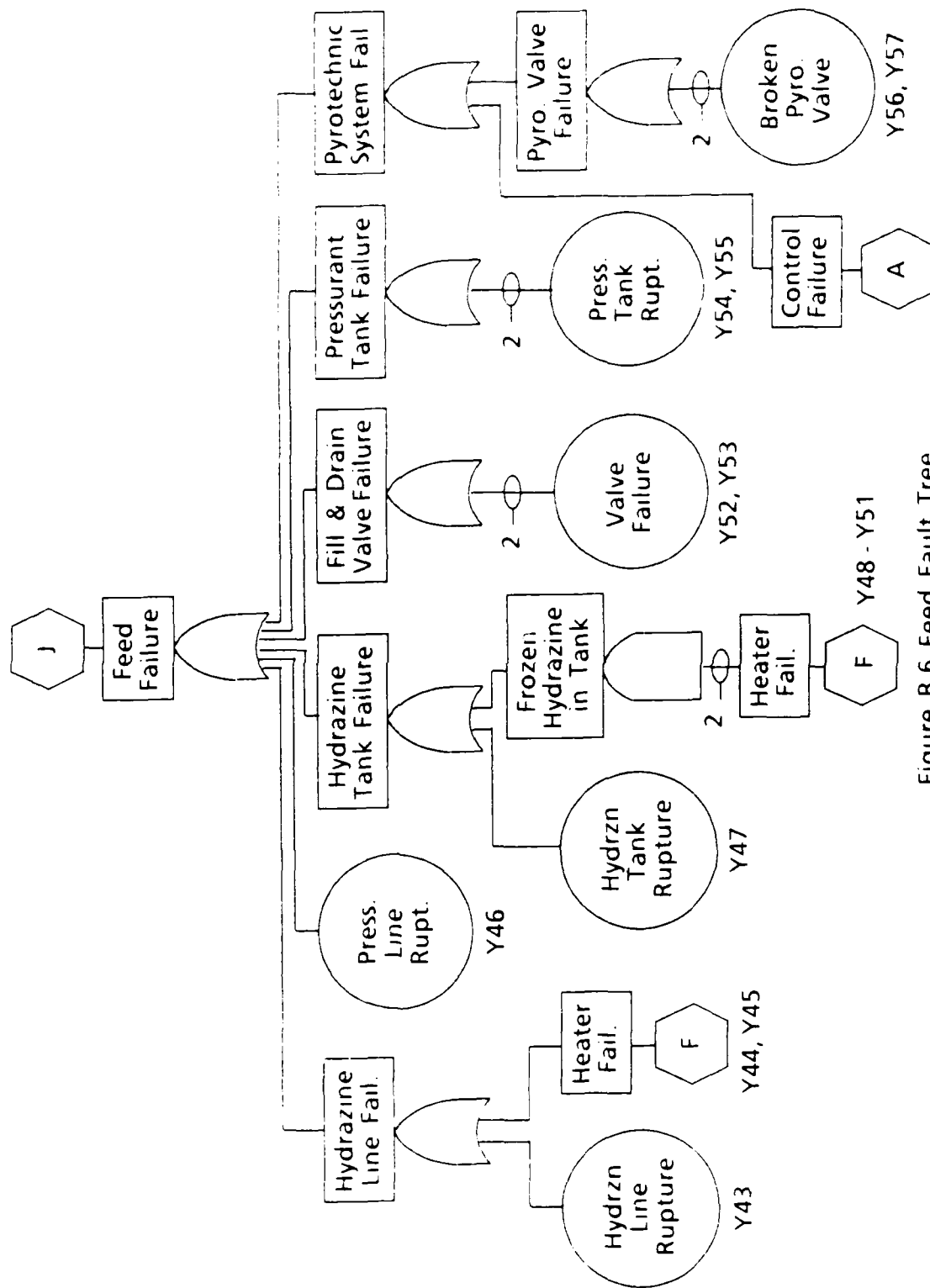


Figure B.6 Feed Fault Tree

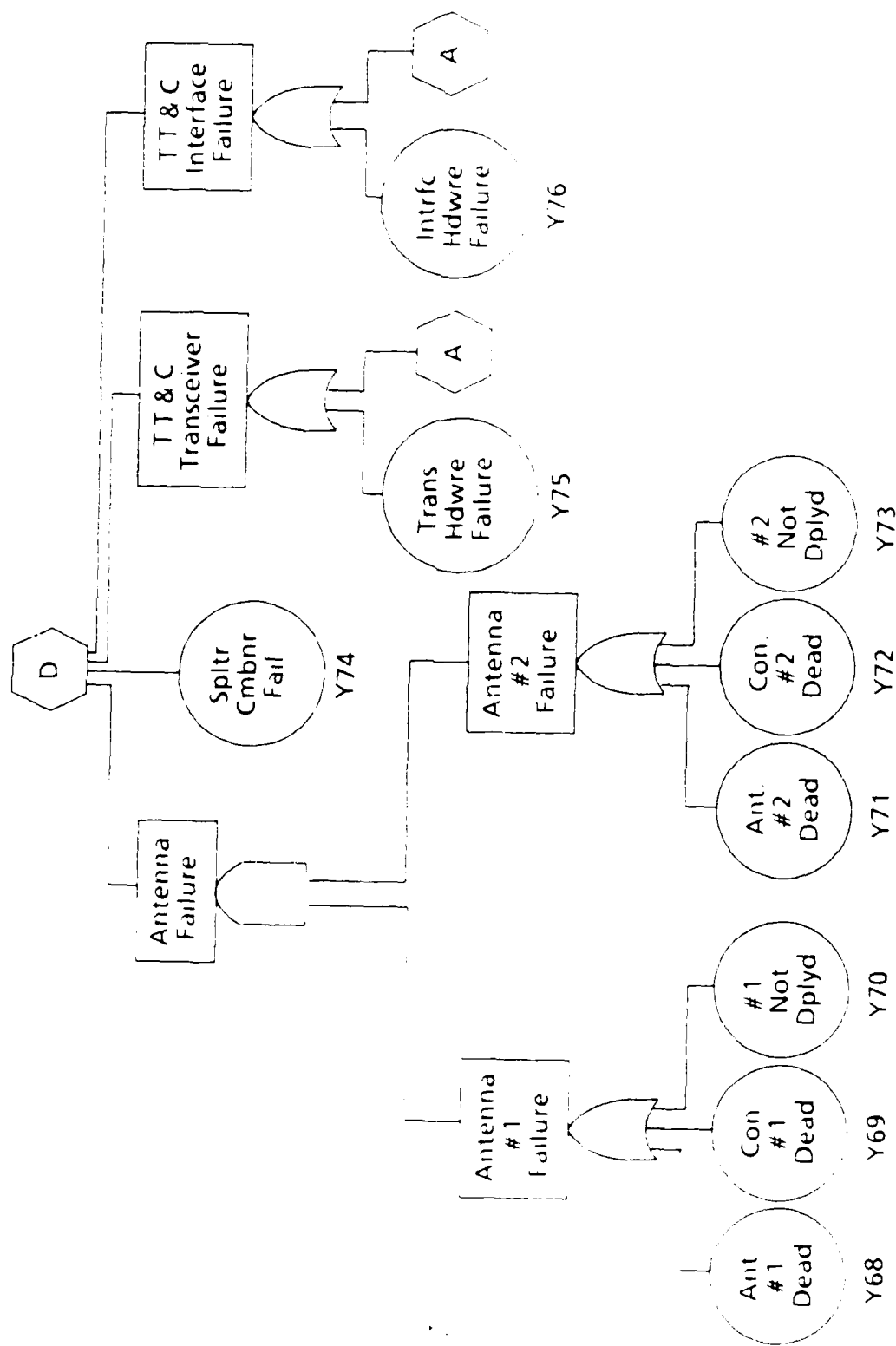


Figure B 7 Telemetry, Tracking and Communication Fault Tree

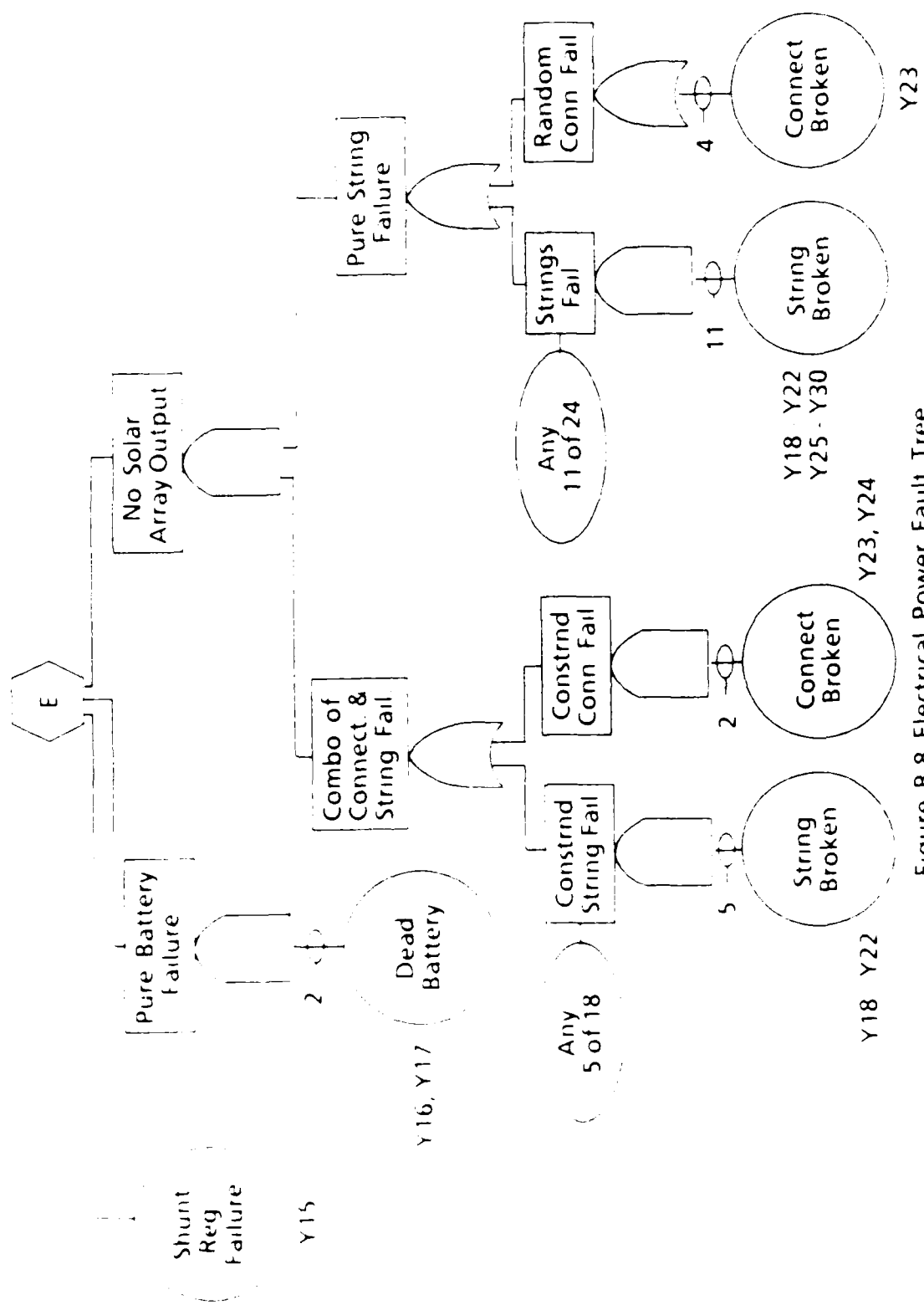


Figure B 8 Electrical Power Fault Tree

APPENDIX C

ORION PATH AND CUT SETS

Use of the min cut algorithm produced 82 minimal cut sets. Their basic component designation and description are listed below:

Single Element Cut Sets

Y1 Attitude control interface electronics
Y3 Data storage controller
Y13 Heater control hardware
Y14 Computer
Y15 Shunt regulator
Y42 Propulsion interface electronics
Y43 Hydrazine line
Y44 Hydrazine line heater
Y45 Hydrazine line thermistor
Y46 Pressurant line
Y47 Hydrazine tank
Y52 and Y53 Fill and drain valve
Y54 and Y55 Pressurant tank
Y56 and Y57 Pyrotechnic valve
Y66 Orbit thruster
Y67 Orbit thruster heater
Y74 TT&C combiner splitter
Y75 TT&C transceiver hardware
Y76 TT&C interface hardware

Double Element Cut Sets

Y2, Y31 Sun sensor and earth sensor
Y16, Y17 Both batteries
Y23, Y24 Two solar array connectors
Y36, Y37 Two momentum wheels
Y38, Y39 Y38, Y41 Any pair of thrusters

Y39, Y40	Y40, Y41	(spin up, spin down or
Y58, Y59	Y58, Y61	nutatation) disabled by a
Y59, Y60	Y60, Y61	combination of the
.62, Y63	Y62, Y65	thruster or its heater
Y63, Y64	Y64, Y65	failing and a similar
		failure on the coupled
		thruster.
Y48, Y50	Y48, Y51	Any combination of the
Y49, Y50	Y49, Y51	heaters and thermistors
		on the hydrazine tank.
Y68, Y71	Y68, Y72	Any combination of an
Y68, Y73	Y69, Y71	antenna, an antenna
Y69, Y72	Y69, Y73	connector, or antenna
Y70, Y71	Y70, Y72	deployment with the
Y70, Y73		similar events of the
		other antenna.

Triple Element Cut Sets

All of these cut sets are any combination of a bubble memory card, its heater or its thermistor with the similar events on any other two bubble memory cards.

Y4, Y5, Y6	Y4, Y5, Y9	Y4, Y5, Y12
Y4, Y8, Y6	Y4, Y8, Y9	Y4, Y8, Y12
Y4, Y11, Y6	Y4, Y11, Y9	Y4, Y11, Y12
Y7, Y5, Y6	Y7, Y5, Y9	Y7, Y5, Y12
Y7, Y8, Y6	Y7, Y8, Y9	Y7, Y8, Y12
Y7, Y11, Y6	Y7, Y11, Y9	Y7, Y11, Y12
Y10, Y5, Y6	Y10, Y5, Y9	Y10, Y5, Y12
Y10, Y8, Y6	Y10, Y8, Y9	Y10, Y8, Y12
Y10, Y11, Y6	Y10, Y11, Y9	Y10, Y11, Y12

Five Element Cut Sets

Y2, Y32, Y33, Y34, Y35 The sun sensor and all
four magnetometers

Y31, Y32, Y33, Y34, Y35 The earth sensor and all
four magnetometers

Six Element Cut Set

Y18, Y19, Y20, Y21, Y22, Y23 One solar array
and any five solar strings from the
remaining 18

Eleven Element Cut Set

Y18, Y19, Y20, Y21, Y22, Y25, Y26, Y27, Y28, Y29, Y30
Any combination of 11 solar strings from the 24

The following components were determined to have
the highest structural importance.

- Computer
- Shunt regulator
- Solar array connectors
- Heater control hardware
- Hydrazine tank
- Hydrazine line
- Hydrazine line heater
- Hydrazine line thermistor
- Pressurant tanks
- Pressurant line
- Fill and drain valves
- Propulsion interface electronics
- Orbit thruster
- Orbit thruster heater
- Attitude control interface
- Data storage controller

- TT&C combiner splitter
- TT&C transceiver hardware
- TT&C interface hardware

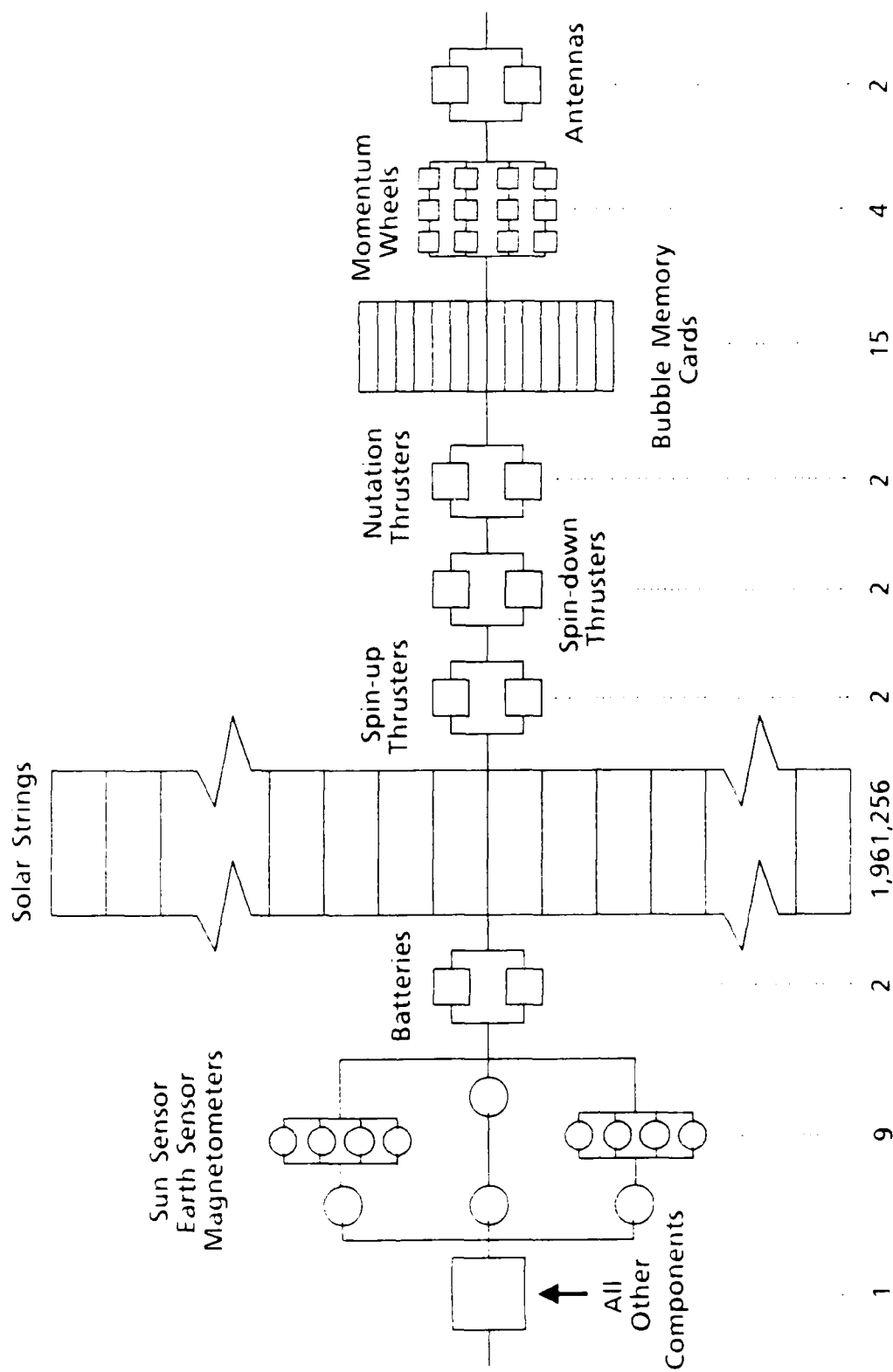


Figure C.1 Path Sets

APPENDIX D

LOTUS SPREADSHEET LISTING

The enclosed listing of a Lotus 1-2-3 spreadsheet was converted to a MathPlan 3.0 format for inclusion in this Appendix. It contains the elements necessary to do a "what-if" analysis. As the subsystems are designed and constructed, their reliabilities can be placed in the spreadsheet to observe the subsystem's impact on the system's reliability.

AB1 = (1-[I6])/[I6]
 X2 = [D29]*[D30]*[D31]
 AC2 = AC3*[AB1]
 AG2 = [I31]
 AI2 = [I31]
 AK2 = [I31]
 AL2 = 1-((1-[AG2])*(1-[AI2])*(1-[AK2]))
 L3 = [I38]
 O3 = [I20]
 Q3 = [I20]
 R3 = 1-((1-[O3])*(1-[Q3]))
 V3 = (1-[I9])/[I9]
 X3 = (1-X[2])/X[2]
 AC3 = AC4*[AB1]
 AD3 = AD4+(AB3*AC3)
 AG3 = [I31]
 AI3 = [I38]
 AK3 = [I39]
 AL3 = 1-((1-[AG3])*(1-[AI3])*(1-[AK3]))
 L4 = [I8]
 O4 = [I9]
 Q4 = [I9]
 R4 = 1-((1-[O4])*(1-[Q4]))
 V4 = V5*[V3]
 W4 = W5+U4*V4
 X4 = X[5]*X[3]
 Y4 = Y5+[U]4*X4
 AC4 = AC5*[AB1]
 AD4 = AD5+(AB4*AC4)
 AG4 = [I31]
 AI4 = [I39]
 AK4 = [I39]
 AL4 = 1-((1-[AG4])*(1-[AI4])*(1-[AK4]))
 L5 = [D7]
 O5 = [I16]
 Q5 = [I17]
 R5 = 1-((1-[O5])*(1-[Q5]))
 V5 = V6*[V3]
 W5 = W6+U5*V5
 X5 = X[6]*X[3]
 Y5 = Y6+[U]5*X5
 AC5 = AC6*[AB1]
 AD5 = AD6+(AB5*AC5)
 AG5 = [I31]
 AI5 = [I31]
 AK5 = [I39]
 AL5 = 1-((1-[AG5])*(1-[AI5])*(1-[AK5]))
 D6 = EXP(-[C]6*26280)
 I6 = EXP(-[H]6*26280)

```

L6      = [ I32 ]
O6      = [ I7 ]
Q6      = [ I7 ]
R6      = 1-((1-[O6]))*(1-[Q6]))
V6      = [ I9 ]^T6
W6      = V6
X6      = X[2]^2
Y6      = X6
AC6     = AC7*[AB1]
AD6     = AD7+(AB6*AC6)
AG6     = [ I31 ]
AI6     = [ I38 ]
AK6     = [ I38 ]
AL6     = 1-((1-[AG6]))*(1-[AI6]))*(1-[AK6]))
D7      = EXP(-[C]7*26280)
I7      = EXP(-[H]7*26280)
L7      = [ D6 ]
O7      = [ D9 ]
Q7      = [ D9 ]
R7      = 1-((1-[O7]))*(1-[Q7]))
AC7     = AC8*[AB1]
AD7     = AD8+(AB7*AC7)
AG7     = [ I31 ]
AI7     = [ I31 ]
AK7     = [ I38 ]
AL7     = 1-((1-[AG7]))*(1-[AI7]))*(1-[AK7]))
I8      = EXP(-[H]8*26280)
L8      = [ D18 ]
O8      = [ D9 ]
Q8      = [ D13 ]
R8      = 1-((1-[O8]))*(1-[Q8]))
AC8     = AC9*[AB1]
AD8     = AD9+(AB8*AC8)
AG8     = [ I39 ]
AI8     = [ I39 ]
AK8     = [ I39 ]
AL8     = 1-((1-[AG8]))*(1-[AI8]))*(1-[AK8]))
D9      = EXP(-C9*26280)
I9      = EXP(-[H]9*26280)
L9      = [ D33 ]
O9      = [ D31 ]
Q9      = [ D31 ]
R9      = 1-((1-[O9]))*(1-[Q9]))
V9      = (1-[I20])/[I20]
X9      = (1-[I18])/[I18]
AC9     = AC10*[AB1]
AD9     = AD10+(AB9*AC9)
AG9     = [ I38 ]
AI9     = [ I38 ]

```

```

AK9      = [ I38 ]
AL9      = 1-((1-[AG9]))*(1-[AI9]))*(1-[AK9]))
D10      = EXP(-[C]10*26280)
L10      = [ D34 ]
O10      = [ I38 ]
Q10      = [ I39 ]
R10      = 1-((1-[O10]))*(1-[Q10]))
V10      = V11*V[9]
W10      = W11+(V10*[U]10)
X10      = X11*X[9]
Y10      = Y11+(X10*[U]10)
Z10      = Z11+(Y10*[U]10)
AC10     = AC11*[AB1]
AD10     = AD11+(AB10*AC10)
AG10     = [ I38 ]
AI10     = [ I39 ]
AK10     = [ I39 ]
AL10     = 1-((1-[AG10]))*(1-[AI10]))*(1-[AK10]))
D11      = EXP(-[C]11*26280)
L11      = [ D17 ]
O11      = [ D13 ]
Q11      = [ D13 ]
R11      = 1-((1-[O11]))*(1-[Q11]))
V11      = V12*V[9]
W11      = W12+(V11*[U]11)
X11      = X12*X[9]
Y11      = Y12+(X11*[U]11)
Z11      = Z12+(Y11*[U]11)
AC11     = AC12*[AB1]
AD11     = AD12+(AB11*AC11)
AG11     = [ I38 ]
AI11     = [ I38 ]
AK11     = [ I39 ]
AL11     = 1-((1-[AG11]))*(1-[AI11]))*(1-[AK11]))
L12      = [ I39 ]
O12      = [ D30 ]
Q12      = [ D31 ]
R12      = 1-((1-[O12]))*(1-[Q12]))
V12      = V13*V[9]
W12      = W13+(V12*[U]12)
X12      = X13*X[9]
Y12      = Y13+(X12*[U]12)
Z12      = Z13+(Y12*[U]12)
AC12     = AC13*[AB1]
AD12     = AD13+(AB12*AC12)
D13      = EXP(-[C]13*26280)
L13      = [ I40 ]
O13      = [ D29 ]
Q13      = [ D31 ]

```

```

R13      = 1-((1-[O13]))*(1-[Q13]))
V13      = V14*V[9]
W13      = W14+(V13*[U]13)
X13      = X14*X[9]
Y13      = Y14+(X13*[U]13)
Z13      = Z14+(Y21*[U]13)
AC13     = AC14*[AB1]
AD13     = AD14+(AB13*AC13)
D14      = EXP(-[C]14*26280)
L14      = [I19]
O14      = [D30]
Q14      = [D30]
R14      = 1-((1-[O14]))*(1-[Q14]))
V14      = [I20]^4
W14      = V14*U14
X14      = [I18]^4
Y14      = [I18]^4
Z14      = [I7]^4
AC14     = AC15*[AB1]
AD14     = AD15+(AB14*AC14)
D15      = EXP(-[C]15*26280)
L15      = [D21]
O15      = [D29]
Q15      = [D30]
R15      = 1-((1-[O15]))*(1-[Q15]))
AC15     = AC16*[AB1]
AD15     = AD16+(AB15*AC15)
AG15     = [I16]
AI15     = LOOKUP(( [G18]-[J18]+1 ),[Y10]:[T14])
AL15     = 1-((1-[AG15]))*(1-[AI15]))
I16      = EXP(-[H]16*26280)
L16      = [I25]
O16      = [D29]
Q16      = [D29]
R16      = 1-((1-[O16]))*(1-[Q16]))
V16      = (1-[X16])/[X16]
X16      = [I38]*[I39]*[I31]
AC16     = AC17*[AB1]
AD16     = AD17+(AB16*AC16)
AG16     = [I17]
AI16     = LOOKUP(( [G18]-[J18]+1 ),[Y10]:[T14])
AL16     = 1-((1-[AI16]))*(1-[AG16]))
D17      = EXP(-[C]17*26280)
I17      = EXP(-[H]17*26280)
L17      = [D19]
V17      = V18*V[16]
W17      = W18+(U17*V17)
Y17      = (1-[I7])/[I7]
AC17     = AC18*[AB1]

```

```

AD17 = AD18+(AB17*AC17)
D18 = EXP(-[C]18*26280)
I18 = EXP(-[H]18*26280)
L18 = [D23]
V18 = V19*V[16]
W18 = W19+(U18*V18)
Y18 = Y19*Y[17]
AC18 = AC19*[AB1]
AD18 = AD19+(AB18*AC18)
D19 = EXP(-[C]19*26280)
I19 = EXP(-[H]19*26280)
L19 = [D32]
V19 = V20*V[16]
W19 = W20+(U19*V19)
Y19 = Y20*Y[17]
AC19 = AC20*[AB1]
AD19 = AD20+(AB19*AC19)
D20 = EXP(-[C]20*26280)
I20 = EXP(-[H]20*26280)
L20 = [D20]
V20 = V21*V[16]
W20 = W21+(U20*V20)
Y20 = Y21*Y[17]
AC20 = AC21*[AB1]
AD20 = AD21+(AB20*AC20)
AG20 = LOOKUP([J6],[AA33]:[AD51])
AI20 = LOOKUP([J7],[T10]:[Z14])
AL20 = 1-((1-[AI20])*(1-[AG20]))
AN20 = [B84]
AP20 = MIN([L30]:[L75])
D21 = EXP(-[C]21*26280)
V21 = V22*V[16]
W21 = W22+(U21*V21)
Y21 = Y22*Y[17]
AC21 = AC22*[AB1]
AD21 = AD22+(AB21*AC21)
AG21 = LOOKUP([J6],[AA2]:[AD26])
AL21 = [AG21]
AN21 = [A55]
D22 = EXP(-[C]22*26280)
V22 = V23*V[16]
W22 = W23+(U22*V22)
Y22 = [I7]^4
AC22 = AC23*[AB1]
AD22 = AD23+(AB22*AC22)
AN22 = [C55]
AP22 = [K30]
D23 = EXP(-[C]23*26280)
V23 = [X16]^6

```

$W23 = V23$
 $AC23 = AC24 * [AB1]$
 $AD23 = AD24 + (AB23 * AC23)$
 $AC24 = AC25 * [A31]$
 $AD24 = AD25 + (AB24 * AC24)$
 $I25 = EXP(-[H]25 * 26280)$
 $AC25 = AC26 * [AB1]$
 $AD25 = AD26 + (AB25 * AC25)$
 $V26 = [D9] * [D13]$
 $X26 = [D10] * [D14]$
 $Z26 = D11 * D15$
 $AC26 = [I6]^{24}$
 $AD26 = AB26 * AC26$
 $V27 = (1 - V26) / V26$
 $X27 = (1 - X26) / X26$
 $Z27 = (1 - Z26) / Z26$
 $V28 = V29 * V[27]$
 $W28 = W29 + (U28 * V28)$
 $X28 = X29 * X[27]$
 $Y28 = Y29 + ([U28] * X28)$
 $Z28 = Z29 * Z[27]$
 $AA28 = AA29 + ([U28] * Z28)$
 $D29 = EXP(-[C]29 * 26280)$
 $V29 = V30 * V[27]$
 $W29 = W30 + (U29 * V29)$
 $X29 = X30 * X[27]$
 $Y29 = Y30 + ([U29] * X29)$
 $Z29 = Z30 * Z[27]$
 $AA29 = AA30 + ([U29] * Z29)$
 $D30 = EXP(-[C]30 * 26280)$
 $L30 = [I8]$
 $V30 = V26^2$
 $W30 = V30$
 $X30 = X26^2$
 $Y30 = X30$
 $Z30 = Z26^2$
 $AA30 = Z30$
 $I31 = EXP(-[H]31 * 26280)$
 $L31 = [I38]$
 $D32 = EXP(-[C]32 * 26280)$
 $I32 = EXP(-[H]32 * 26280)$
 $L32 = [D7]$
 $D33 = EXP(-[C]33 * 26280)$
 $L33 = [I32]$
 $AC33 = AC34 * [AB1]$
 $D34 = EXP(-[C]34 * 26280)$
 $L34 = [D6]$
 $AC34 = AC35 * [AB1]$
 $AD34 = AD35 + (AB34 * AC34)$

L35 = [D18]
 AC35 = AC36*[AB1]
 AD35 = AD36+(AB35*AC35)
 L36 = [D33]
 AC36 = AC37*[AB1]
 AD36 = AD37+(AB36*AC36)
 L37 = [D34]
 AC37 = AC38*[AB1]
 AD37 = AD38+(AB37*AC37)
 L38 = EXP(-[H]38*26280)
 L38 = [D17]
 AC38 = AC39*[AB1]
 AD38 = AD39+(AB38*AC38)
 L39 = EXP(-[H]39*26280)
 L39 = 1-((1-[AG15])*1-[A115])
 AC39 = AC40*[AB1]
 AD39 = AD40+(AB39*AC39)
 L40 = EXP(-[H]40*26280)
 L40 = [I39]
 AC40 = AC41*[AB1]
 AD40 = AD41+(AB40*AC40)
 L41 = [I40]
 AC41 = AC42*[AB1]
 AD41 = AD42+(AB41*AC41)
 L42 = [I19]
 AC42 = AC43*[AB1]
 AD42 = AD43+(AB42*AC42)
 L43 = 1-((1-[A116])*1-[AG16])
 AC43 = AC44*[AB1]
 AD43 = AD44+(AB43*AC43)
 L44 = [D21]
 AC44 = AC45*[AB1]
 AD44 = AD45+(AB44*AC44)
 B45 = [I16]
 L45 = 1-((1-[O5])*1-[Q5])
 AC45 = AC46*[AB1]
 AD45 = AD46+(AB45*AC45)
 B46 = LOOKUP([J18],[T10],[Y14])
 L46 = [I25]
 AC46 = AC47*[AB1]
 AD46 = AD47+(AB46*AC46)
 C47 = [B45]*[B46]
 L47 = [D19]
 AC47 = AC48*[AB1]
 AD47 = AD48+(AB47*AC47)
 B48 = [I17]
 L48 = 1-((1-[O3])*1-[Q3])
 AC48 = AC49*[AB1]
 AD48 = AD49+(AB48*AC48)

```

B49 = LOOKUP([E18],[T10],[Y14]
149 = [D23]
AC49 = AC50*[AB1]
AD49 = AD50+[AB49]*AC49
C50 = [B48]*[B49]
150 = [D32]
AC50 = AC51*[AB1]
AD50 = AD51+[AB50]*AC50
B51 = [16]
151 = 1-[01]*[1-[Q1]]
AC51 = [16]*18
AD51 = AB51*AC51
B52 = [11]
152 = 1-[04]*[1-[Q4]]
C53 = [B51]*[B52]
153 = [D20]
154 = 1-[08]*[1-[Q8]]
B55 = [138]*12
155 = [B51]*[B52]
155 = [G81]*[G82]
155 = [AG21]
B56 = [D6]*[D1]
156 = [B45]*[B46]
156 = [G76]*[G77]
156 = 1-[09]*[1-[Q9]]
B57 = [139]*12
157 = [B48]*[B49]
157 = 1-[011]*[1-[Q11]]
B58 = [18]
158 = 1-[010]*[1-[Q10]]
B59 = LOOKUP([E10],[T28],[AA30])
159 = 1-[012]*[1-[Q12]]
B60 = MAX([C47],[C53])
160 = 1-[11-[AG9]]*[1-[AI9]]*[1-[AK9]]
B61 = LOOKUP([E11],[T28],[AA30])
161 = 1-[06]*[1-[Q6]]
B62 = [132]
162 = 1-[014]*[1-[Q14]]
B63 = [D18]
163 = 1-[A120]*[1-[AG20]]
B64 = [D33]
164 = 1-[013]*[1-[Q13]]
B65 = [D34]
165 = 1-[AG11]*[1-[AI11]]*[1-[AK11]]
B66 = [D17]
166 = 1-[015]*[1-[Q15]]
B67 = MAX([G78],[G83])
167 = 1-[AG10]*[1-[AI10]]*[1-[AK10]]
B68 = LOOKUP([J20],[T10],[W14])

```

L68 = 1-((1-[AG6])*(1-[AI6])*(1-[AK6]))
 B69 = [D19]^2
 L69 = 1-((1-[O16])*(1-[Q16]))
 B70 = [I40]
 L70 = 1-((1-[AG8])*(1-[AIS])*(1-[AKS]))
 B71 = [I19]
 L71 = 1-((1-[AG3])*(1-[AI3])*(1-[AK3]))
 B72 = [D21]
 L72 = 1-((1-[AG4])*(1-[AI4])*(1-[AK4]))
 B73 = [I25]
 L73 = 1-((1-[AG7])*(1-[AI7])*(1-[AK7]))
 B74 = LOOKUP([E9],[T28]:[AA30])
 L74 = 1-((1-[AG5])*(1-[AI5])*(1-[AK5]))
 B75 = [D32]
 L75 = 1-((1-[AG2])*(1-[AI2])*(1-[AK2]))
 B76 = [D23]
 G76 = LOOKUP([J6],[AA2]:[AD26])
 B77 = [D22]^2
 G77 = [I7]^4
 B78 = LOOKUP([J9],[T4]:[W6])
 G78 = [G76]*[G77]
 B79 = [D20]^2
 B80 = LOOKUP([J31],[T17]:[W23])
 B81 = LOOKUP([E29],[T4]:[Y6])
 G81 = LOOKUP([J6],[AA33]:[AD51])
 G82 = LOOKUP([J7],[T10]:[Z14])
 G83 = [G81]*[G82]
 B84 = B55*B56*B57*B58*B59*B60*B61*B62*B63*B64*B65*
 B66*B67*B68*B69*B70*B71*B72*B73*B74*B75*B76*
 B77*B78*B79*B80*B81

LIST OF REFERENCES

1. BIT's 'n Pieces, v. 11, no. 5, p. 18, May 1978.
2. Operations Research Center University of California, Berkeley, System Reliability Analysis: Foundations, by R. E. Barlow, July 1982.
3. Ziehms, H., Reliability Analysis of Phased Missions, Ph.D. Thesis, Naval Postgraduate School, Monterey, California, December 1974.
4. USAMC Intern Training Center, A Fault Tree Manual, by T. W. Long, December 1970.
5. Office of Naval Research, Arlington, Virginia, Introduction to Fault Tree Analysis, by R. E. Barlow and P. Chatterjee, December 1973.
6. Esary, J.D., and Marshall, A.W., "System Structure and the Existence of a System Life," Technometrics v. 6, pp. 459-462, 1964.
7. Dhillon, B. S., and Singh, G., "On Fault Trees and Other Reliability Evaluation Methods," Microelectronics and Reliability, v. 19, pp. 57-63, 1979.
8. Barlow, R. E., and Proschan, F. Statistical Theory of Reliability and Life Testing Probability Models, 1st ed., To Begin With, 1975.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3. Superintendent Attn: Prof. J. D. Esary, Code 55Ey Naval Postgraduate School Monterey, California 93943-5000	1
4. Prof. Allen E. Fuhs (Emeritus) Box 222040 Carmel, California 93922	1
5. MAJ Trenton G. Keeble 3491 Earl Dr. Santa Clara, California 95051	2
6. Superintendent Attn: Marty Mosier, Code 72 Naval Postgraduate School Monterey, California 93943-5000	2
7. Rand Corporation Attn: Herb Shuklar 1700 Main St. Santa Monica, California 90406-2138	1
8. Ed Senasack Naval Research Laboratory, Code 8220 4555 Overlook Ave., S.W. Washington, D.C. 20375	1

9. Tom Whitmeyer
Naval Research Laboratory, Code 8223
4555 Overlook Ave., S.W.
Washington, D.C. 20375

1

END

DATE
FILMED

DEC.

1987