ornl

OAK RIDGE NATIONAL LABORATORY

UNION CARBIDE

NUREG/CR-2677
ORNL/TM-8313

ESP and NOAH—Computer Programs for Flood Risk Analysis of Nuclear Power Plants

D. P. Wagner
D. F. Montague
J. J. Rooney
J. B. Fussell
L. S. Baker

This Work Performed For
Nuclear Regulatory Commission
Under
DOE Interagency Agreement 40-550-75

OPERATED BY
UNION CARBIDE CORPORATION
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161

Available from

GPO Sales Program
Division of Technical Information and Document Control
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555

NUREG/CR-2677
ORNL/TM-8313
Distribution Category RG

Contract No. W-7405-eng-26

Engineering Physics Division

ESP AND NOAH - COMPUTER PROGRAMS FOR FLOOD RISK ANALYSIS OF NUCLEAR POWER PLANTS

D. P. Wagner*
D. F. Montague*
J. J. Rooney*
J. B. Fussell*
L. S. Baker*

NRC Monitor: P. K. Niyogi
Risk Methodology & Data Branch
Division of Risk Analysis

Manuscript Completed: December, 1981

Date Published: June 1982

This Work Performed For
Nuclear Regulatory Commission
Under
DOE Interagency Agreement 40-550-75
NRC FIN No. B0436

Work Performed Under
ORNL Subcontract No. 85B-13860C

*JBF Associates, Inc.
1630 Downtown West Boulevard
Knoxville, Tennessee 37919

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37830
operated by
UNION CARBIDE CORPORATION
for the
DEPARTMENT OF ENERGY

ACKNOWLEDGMENTS

We express sincere appreciation to S. R. Sturges (U. S. Nuclear Regulatory Commission), W. E. Vesely (Battelle Columbus National Laboratory) and G. F. Flanagan (Oak Ridge National Laboratory) for their support of this work.

ABSTRACT

This report describes a computer program package that aids in assessing the impact of floods on risk from nuclear power plants. The package consists of two distinct computer programs: ESP and NOAH.

The ESP program improves the efficiency of a flood analysis by screening accident sequences and identifying accident sequences that are potentially significant contributors to risk in the event of a flood. Input to ESP includes accident sequences from an existing risk assessment and flood screening criteria. The output from ESP includes:

- accident sequences that are potentially significant contributors to risk,
- specific plant systems contained in these accident sequences that may require detailed analysis, and
- a quantitative estimate of the flood contribution to risk.

The NOAH program provides detailed qualitative analysis of the plant systems identified by ESP. Input to NOAH includes:

- the system fault tree from the existing risk assessment,
- vulnerability elevations for each component represented in the fault tree, and
- a detailed flood level profile for the plant.

NOAH performs a qualitative flood simulation of the fault tree to determine:

- flooded minimal cut sets; that is, minimal cut sets that have all their components submerged by the flood,
- partially flooded minimal cut sets when no flooded minimal cut sets are found,
- flood protection sets; that is, groups of components that can be protected to mitigate the flood effects on the system, and
- the critical flood level; that is, the lowest flood level where at least one minimal cut set is submerged.

The output from NOAH is directly usable in quantitative evaluations of flood effects on plant risk.

This report contains detailed user's manuals for both the ESP and NOAH computer programs. Input and output of both programs is described and illustrated with example problems. Detailed programmer's guides are also provided in the appendices.

This report assumes that the reader has experience in the areas of fault tree analysis and risk assessment methodology.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

APPENDICES

LIST OF TABLES

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

ESP AND NOAH - Computer Programs for Flood Risk Analysis of Nuclear Power Plants

1. INTRODUCTION

This document is the user's manual for two computer programs developed to aid in flood risk analysis of nuclear power plants. These computer programs are an integral part of a methodology for analyzing the effects of floods on nuclear power plant systems[1].

Both the Reactor Safety Study[2] and the Lewis Committee Report[3] identify floods as external hazards that warrant further investigation in assessing the risk associated with nuclear power plants. The importance of floods results from their potential to produce multiple component failures via submersion of individual components. These multiple component failures are called common cause failures and can result in system failures which contribute to the overall risk from nuclear power plants. Thus, consideration of common cause failures due to floods is an important aspect of the overall risk assessment of nuclear power plants.

The flood risk analysis methodology is designed to identify and quantify these flood effects using existing risk assessment results as input. The ESP computer program aids in identifying accident sequences and systems that are potentially significant contributors to plant risk due to flood effects. ESP accepts as input accident sequences and system failure probabilities from an existing risk assessment, engineering criteria describing system susceptibility to floods and a potential flood probability. ESP screens the accident sequences based on the engineering criteria and determines important system failures and accident sequences along with a quantitative estimate of each sequence's contribution to risk due to floods. The important accident sequences identified by ESP provide input to the quantitative evaluation of flood effects. The important system failures identified by ESP are candidates for detailed system analysis using the NOAH computer program.

The NOAH computer program accepts system fault trees from existing risk assessments as input. Other required input includes flood level increments within the plant (discretized flood level profile) and the effective elevation (component vulnerability elevation) of each component in the fault tree. NOAH simulates flooding of the components in the fault trees based on the flood level profile and the components' vulnerability elevation. The output of the flood simulation includes the order of component submersion and the flooded system minimal cut sets, if any exist. If no flooded minimal cut sets exist, NOAH determines partially flooded system minimal cut sets. The flooded and partially flooded minimal cut sets represent the important system failure modes in the event of a flood and are essential inputs to the quantitative evaluation to determine the system failure probability as a function of flood level.

Section 2 of this document is a glossary of terms used in describing both the ESP and NOAH computer programs. Sections 3 through 7 comprise the user's manual for the ESP computer program. Sections 8 through 12 contain the user's manual for the NOAH computer program. Each user's manual is complete with examples of program input and output. References for both user's manuals are contained in Section 13.

2. CONCEPTS AND DEFINITIONS

This section describes the concepts of the flood risk analysis methodology applicable to the ESP and NOAH computer programs. A complete discussion of the methodology is found in Reference 1.

2.1 Event Sequence

Event trees are event sequence models that graphically display postulated accident scenarios (Figure 2.1). The elements of an event sequence, or accident sequence as they are often called, are an initiating event, branching operator failures and an identification of the consequence category to which the sequence leads. An initiating event is an undesirable event (component or system failure, transient or external event) that starts an accident sequence. The branching operators generally represent actions taken by plant systems or personnel which, if successful, act as barriers to the propagation of the event sequence or mitigate the effects of the initiating event. The success or failure of these branching operators determines the magnitude of the consequence of an accident. The consequence category identification defines the consequence to which the accident sequence leads.

The occurrence frequency of a particular accident sequence is the product of the initiating event occurrence frequency and the conditional probabilities of failure on demand of the branching operators. The probabilities of failure on demand of the branching operators are usually very small; therefore, the probabilities of success on demand of the branching operators are very close to one for systems encountered in nuclear power plants. In practice, the success on demand probabilities are assumed to be one and the accident sequence occurrence frequency contains only failure events.

2.2 Component Vulnerability Elevation

The "vulnerability elevation" for a component is defined as the lowest physical elevation that the flood level must surpass to affect the component. The vulnerability elevation includes the case where a component may be affected by the flood but is not yet submerged itself. For example, a pump whose function is dependent on electrical connections at an elevation that is physically lower than the pump will be assigned the lower elevation as its vulnerability elevation. However, if the pump's vital electrical connections are physically higher than the pump, the pump's vulnerability elevation may be the physical elevation of the pump. A component's vulnerability elevation will be physically higher than the component where specific barriers prevent the flood from affecting the component until it overflows the barrier. In this case, the vulnerability elevation corresponds to the physical elevation where the flood overflows the barrier.

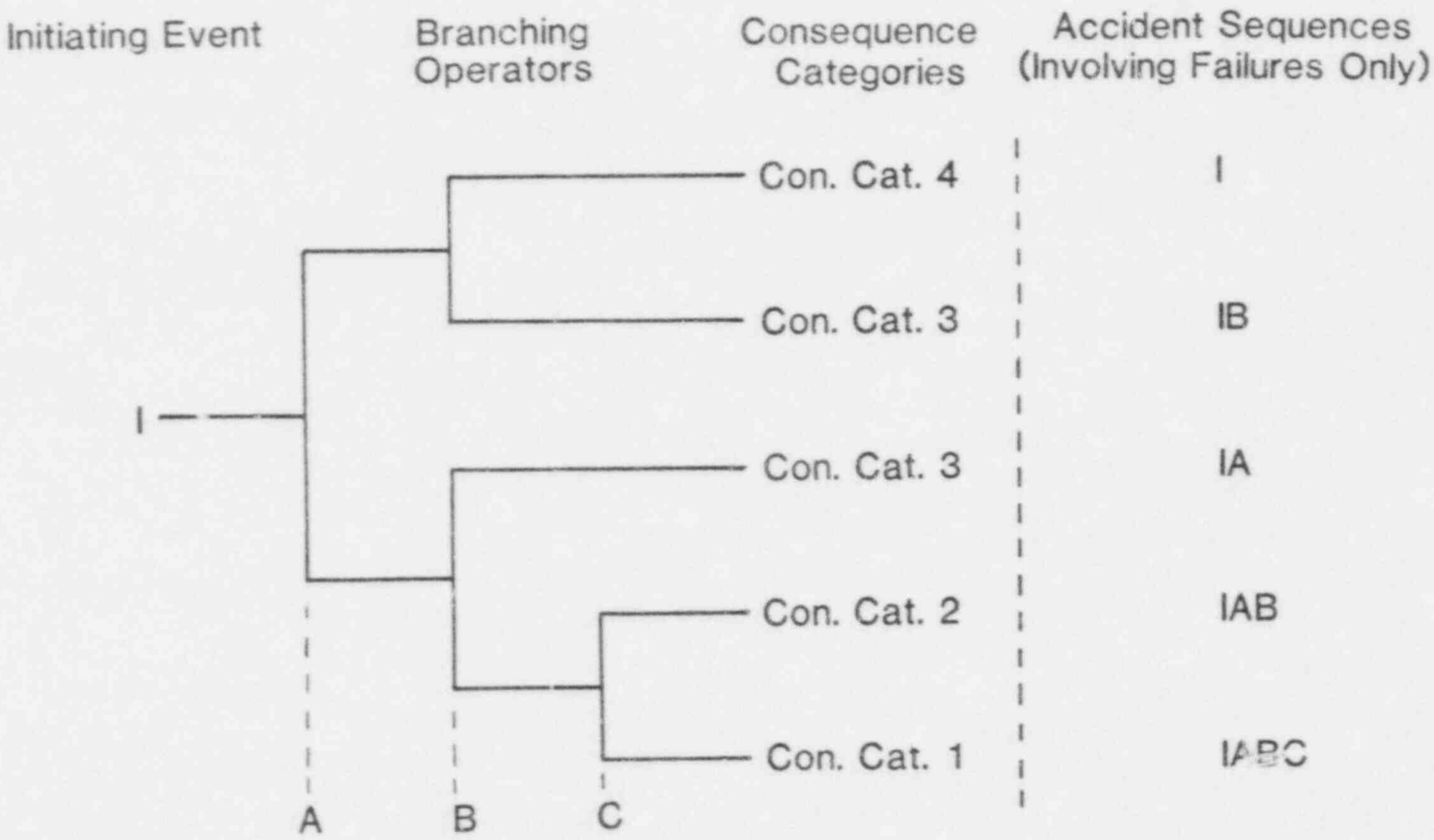

Figure 2.1 Sample Event Tree

2.3 Minimal Cut Sets

A minimal cut set is a group of basic component failures, called basic events, that are collectively sufficient to cause the system failure of interest to occur. The occurrence of each basic event in the minimal cut set is necessary if the occurrence of the system failure is the result of that minimal cut set. The failures of interest in a risk assessment are the failures of the branching operators in the event tree. Using the vulnerability elevation defined above, in conjunction with minimal cut sets, allows determination of the flood levels that affect the branching operators in the event trees.

ESP

A Computer Program for Identifying Potentially Significant Accident Sequences for Flood Analysis

3. ESP: INTRODUCTION

The ESP (Event Sequence Screening Program) computer program identifies accident sequences and elements within these sequences that are potentially significant contributors to increased risk due to flood effects ESP is written in FORTRAN IV for the IBM 360/370 computers.

ESP screens accident sequences that result in a particular consequence category and selects only those whose increased occurrence frequency due to flood effects would result in a significant increase in the total occurrence frequency of that consequence category. Within these selected accident sequences, ESP identifies those elements that are considered likely to fail in the event of a flood. These identified elements are candidates for more detailed analysis.

The screening procedure examines each accident sequence in a particular consequence category to determine if the sequence is significant in the event of a flood. ESP estimates upper bounds for the flooded occurrence frequencies of the accident sequences and compares these values to a user-specified criterion for the appropriate consequence categories. Accident sequences whose flooded occurrence frequencies are greater than this criterion are considered significant and the program identifies the flood susceptible systems in these accident sequences. ESP also calculates an upper bound for the total occurrence frequency of the consequence categories, including both unflooded and flooded effects, and ranks the significant accident sequences in order of contribution to the consequence category's flooded occurrence frequency.

Section 4 presents screening concepts used in the ESP computer program. A general description of ESP is provided in Section 5. Section 6 describes the ESP input groups in detail and the ESP program output is described in Section 7. Appendix A is a programmer's guide for the ESP computer program. Included in this guide are descriptions of ESP subroutines, major program variables, diagnostic information and subroutine calling sequences. Error messages generated by ESP are contained in Appendix B. Appendix C describes the required job control language. A condensed version of ESP input parameters and formats are given in Appendix D.

4. ESP: SCREENING CONCEPTS

Nuclear power plant probabilistic risk assessments (PRA's) usually postulate a large number (>100) of possible accident sequences. Each accident sequence contributes to one of several consequence categories. The accident sequence frequencies collectively determine the frequency at which consequences of various magnitudes occur, thereby providing a measure of risk. Usually, only a small number of sequences (termed dominant accident sequences) contribute significantly to the category frequency and these sequences are the ones analyzed in greatest detail.

The ESP program uses the structure of accident sequences and consequence categories from an existing risk assessment to identify significant accident sequences in the event of a flood. Accident sequences which were previously considered less important because of their relatively low expected frequencies may contribute significantly to risk due to flood-induced failures. ESP uses an analyst's assessment of the susceptibility to flood-induced failure for each element in an accident sequence, a user-specified criterion for identifying significant sequences, and a description of the accident sequences for a consequence category to identify accident sequences important to that consequence category.

To reduce the number of accident sequences which must be input to ESP, the user should prescreen the consequence categories to eliminate certain categories from further consideration. For example, categories with relatively minor consequences may be insignificant relative to the consequences of a flood. Also, if the flood frequency is relatively low compared to the unflooded consequence category occurrence frequency, flooding will make an insignificant contribution even if it fails all the elements of an accident sequence. In such a case, it is not necessary to further analyze the sequences which comprise such a category.

4.1 Flood Susceptible Event Sequence Element

To perform accident sequence screening using ESP, the analyst must identify accident sequence elements (initiating events or branching operators) that are considered susceptible to flood effects. An event sequence element is considered flood susceptible if it is expected to be significantly degraded or to fail in the event a flood occurs. To determine which sequence elements are flood susceptible, qualitative considerations are required. Flood susceptibility may arise from one of several considerations, such as the vulnerability elevation of equipment or the timing involved in demanding a branching operator relative to the time the flood first affects the plant. Reference 1 gives guidelines to aid in identifying flood susceptible event sequence elements. The flood susceptible initiating events and branching operators output by ESP are a subset of the flood susceptible event sequence elements initially identified by the analyst. Those sequence elements output are ones that are flood susceptible <u>and</u> are members of accident sequences that are potentially significant contributors to risk in the event of a flood.

4.2 Screening Procedure

The ESP computer program combines the accident sequence elements, the flood frequency and the screening criterion to identify accident sequences and elements within these accident sequences that are potentially significant contributors to risk. The general flow of the ESP computer program is shown in Figure 4.1. The procedure for screening accident sequences is as follows:

1. ESP selects accident sequences that contribute to a particular consequence category from the input list of accident sequences.

2. ESP calculates the unflooded occurrence frequency of an accident sequence.

3. If an accident sequence contains no flood susceptible elements, then that sequence is eliminated from further screening. If the sequence does contain flood susceptible elements, ESP assumes these elements occur with probability one and calculates the flooded occurrence frequency of the accident sequence. The sequence's flooded occurrence frequency is the product of the occurrence probabilities for the remaining sequence elements (if any) and the occurrence frequency of the flood being analyzed.* The flooded accident sequence occurrence frequency is compared to the unflooded consequence category occurrence frequency to determine if it is significant using the following relationship:

$$\frac{\text{Flooded accident sequence occurrence frequency}}{\text{Unflooded consequence category occurrence frequency}} \geq \text{CRITRA},$$

---

*In some accident sequences, the probability of occurrence of the flood should be used in place of the flood occurrence frequency. For rare floods, the occurrence frequency may be an acceptable estimate of the flood occurrence probability.

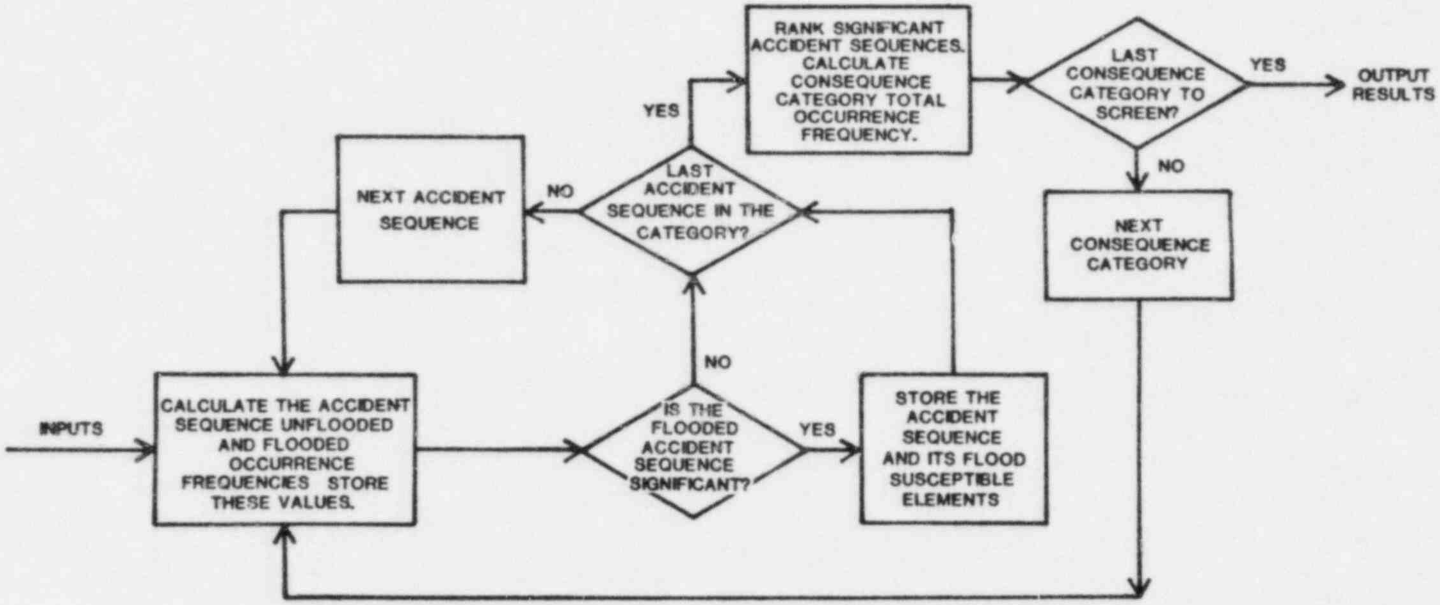

Figure 4.1 ESP Computer Program Simplified Flowchart

where CRITRA is the user-specified screening criteria (>0). If the ratio of the flooded sequence occurrence frequency to the total unflooded consequence category occurrence frequency is less than CRITRA, the accident sequence is discarded. If the ratio is greater than or equal to CRITRA, the accident sequence is considered significant and flood susceptible systems in that accident sequence are identified. The value of this ratio is used to rank the significant accident sequences in order of contribution to the consequence category's flooded occurrence frequency.

4. ESP repeats this process (Steps 2 & 3) for each accident sequence in the consequence category.

5. After all accident sequences of a particular consequence category have been analyzed, the procedure is repeated for the next consequence category.

ESP prints screening results for each consequence category immediately after analyzing each category.

4.3 Consequence Category Occurrence Frequency

The consequence category unflooded occurrence frequency is often estimated in probabilistic risk assessments by summing the unflooded occurrence frequencies of all accident sequences that result in that category. This method requires modification to include flood effects. ESP uses the following equation to estimate the total occurrence frequency, including unflooded and flooded effects, of each consequence category:

$$P(C) = \sum_{i=1}^{k} P(S_i \mid \overline{f}) + P(f)\left[\sum_{i=1}^{k} P(S_i \mid f) - \sum_{i=1}^{k} P(S_i \mid \overline{f})\right],$$

where

C ≡ the event a sequence resulting in consequence category C occurs,

$P(C)$ = the probability per unit time (frequency) of a category C occurrence,

S_i ≡ the event accident sequence S_i occurs, resulting in a consequence category C occurrence,

k = the number of accident sequences contained in consequence category C.

$\overline{f}$ ≡ the event no flood occurs, and f $\equiv$ the event a flood occurs.

This expression, for the total consequence category occurrence frequency, is a first-order overpredicting approximation and may, in some cases, greatly overestimate the total occurrence frequency. For example, when all the elements of an event sequence are susceptible to a flood, the probability of the sequence, given a flood occurs, approaches one. If this is true for several event sequences, the summation of the event sequences' probabilities will exceed one. That is,

$$\sum_{i=1}^{k} P(S_i \mid f) > 1.0,$$

given that several sequences are highly susceptible to floods. The summation of the sequence probabilities is a first order approximation of the logical union of these sets of events, and correction terms for the intersection of these sets are not included in the equation given above. These correction terms will be significant for groups of sequences that are highly susceptible to floods. Therefore, the equation described above will significantly overpredict the flood contribution to the category total for sequences that are highly susceptible to floods. In cases where

$$\left[\sum_{i=1}^{k} P(S_i \mid f) - \sum_{i=1}^{k} P(S_i \mid \bar{f}) \right] > 1.0,$$

the ESP program imposes an upper bound on the flood contribution to the consequence category occurrence frequency. The maximum contribution is the flood frequency itself. Although this is generally an overprediction of the flood's effect also, it represents a better bound than that provided by the equation for the category occurrence frequency given above.

For the reasons discussed above, the analyst should exercise caution when interpreting the flooded occurrence frequencies contained in the ESP output. These results must be viewed as upper bounds for use in the screening process. More exact determination of the quantitative flood effects is possible after detailed systems analysis using the NOAH computer program.

5. ESP: PROGRAM DESCRIPTION

ESP screens accident sequences and identifies sequences and elements within sequences that are potentially significant contributors to the total occurrence frequency of a consequence category, given the occurrence of a flood. Information input to ESP includes:

1. the title card,

2. descriptions of the initiating events and branching operators contained in the accident sequences, their respective unflooded occurrence frequencies or failure on demand probabilities, and an indication of their flood susceptibility,

3. the flood occurrence frequency,

4. the screening criterion used to identify significant accident sequences,

5. descriptions of the consequence categories and their unflooded occurrence frequencies, and

6. descriptions of the accident sequences.

The ESP input is described in detail in Section 6.

The basic output of ESP consists of:

1. the accident sequences that are potentially significant contributors to increased risk due to flood effects,

2. the initiating events or branching operators that are considered flood susceptible and appear in potentially significant accident sequences, and

3. an estimate of the total consequence category occurrence frequency, which includes both unflooded and flooded effects.

In addition to the above information, ESP ranks the potentially significant accident sequences in order of contribution to the flooded occurrence frequency of a consequence category. The flood susceptible initiating events or branching operators identified by ESP may require more detailed flood analysis before determining the specific contribution to risk of these branching operators or initiating events during flooding.

6. ESP: INPUT DESCRIPTION

This section provides a description of the inputs required by ESP. ESP input is divided into groups and each input group is described separately. The following sections give the variable or array names, the proper formats and the purpose of the input for each input group. Appendix D provides a condensed listing of the ESP input formats.

Figure 6.1 shows an example event tree and Table 6.1 gives the accident sequences obtained from this event tree. Table 6.2 gives the occurrence frequency of the initiating event for this event tree and the failure on demand probabilities of the branching operators, along with an indication of whether or not a sequence element is flood susceptible. Table 6.3 lists the flood occurrence frequency and screening criterion and Table 6.4 lists the consequence category total unflooded occurrence frequencies. This example problem illustrates the input for each input group and the ESP program output in Section 6.

6.1 ESP Input Deck Construction

Each input group begins with a special control card of the form:

Column 1	Column 2	Column 3
*	Blank	KEYWORD

with KEYWORD replaced by the appropriate input group identifying name. Table 6.5 gives a listing of all the input group keywords. The conclusion of each input group is indicated by a card with the word END beginning in the first card column. Any number of comment cards may precede the input groups. The first such card (80 characters maximum in length) appearing in the input deck is used as a title card. The title card is a required input.

The complete input deck must be assembled in the following order:

- TITLE CARD
  [COMMENT CARDS]

- * SYSTEM
  [INITIATING EVENT/BRANCHING OPERATOR NAMES, CODE NAMES AND OCCURRENCE FREQUENCY/FAILURE ON DEMAND PROBABILITIES]
  END
  [COMMENT CARDS]

- * FLOOD
  [FLOOD FREQUENCY, SCREENING CRITERIA AND FLOOD SUSCEPTIBLE ACCIDENT SEQUENCE ELEMENTS]
  END
  [COMMENT CARDS]

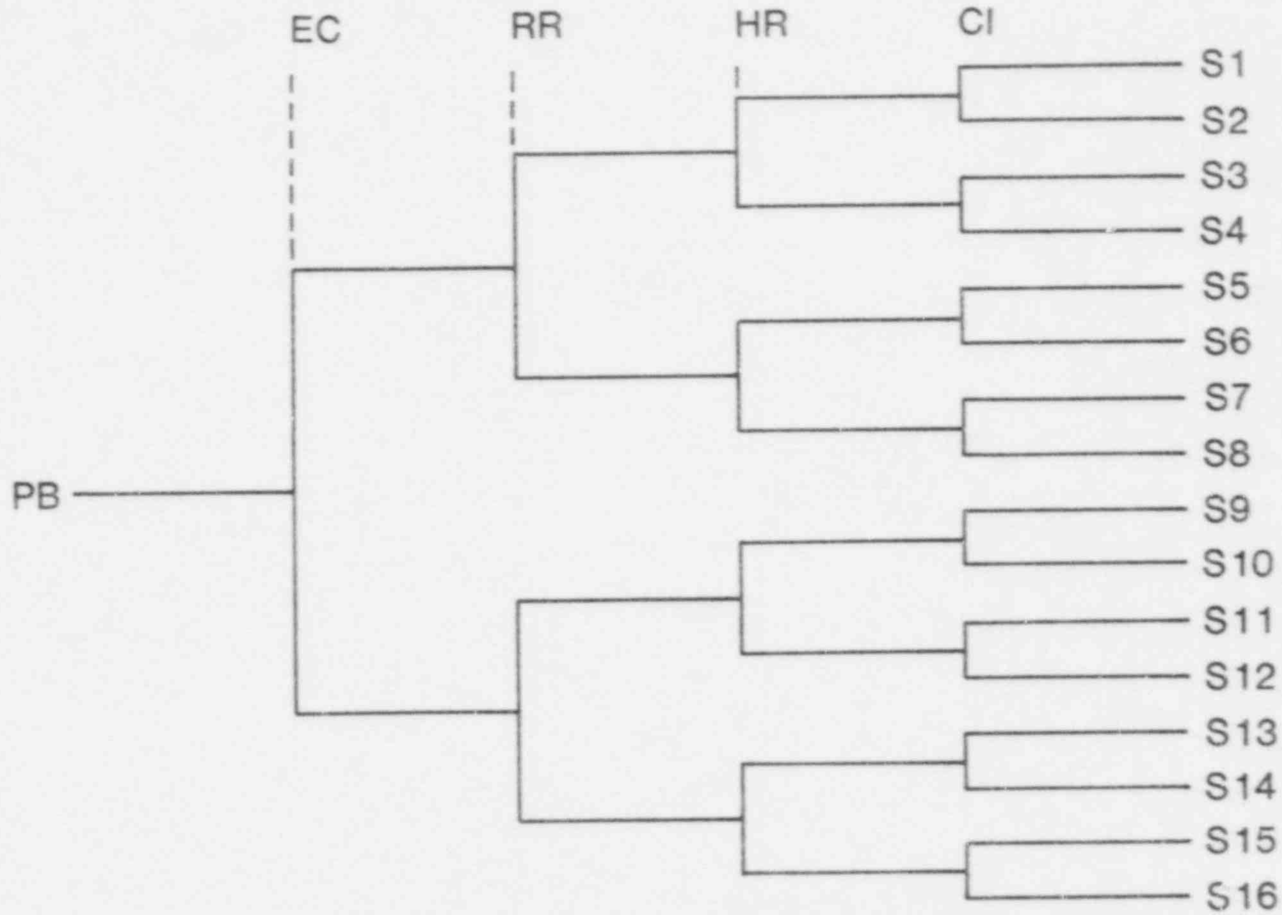

Figure 6.1 ESP Example Problem Event Tree

Table 6.1 Example Problem Accident Sequences

Number	Accident Sequence*					Consequence Category
S1	PB					4
S2	PB	CI				4
S3	PB	HR				4
S4	PB	HR	CI			3
S5	PB	RR				4
S6	PB	RR	CI			3
S7	PB	RR	HR			3
S8	PB	RR	HR	CI		2
S9	PB	EC				3
S10	PB	EC	CI			2
S11	PB	EC	HR			2
S12	PB	EC	HR	CI		2
S13	PB	EC	RR			2
S14	PB	EC	RR	CI		2
S15	PB	EC	RR	HR		2
S16	PB	EC	RR	HR	CI	1

*Syst m/component failures only.

Table 6.2 Example Problem Initiating Event/Branching Point Data

Sequence Element	Code Name	Occurrence Frequency (hr^{-1})	Failure On Demand Probability
Pipe Break	PB	3.0×10^{-4}	
Emergency Core Cooling*	EC		9.5×10^{-5}
Post Accident Rad. Removal	RR		6.0×10^{-3}
Post Accident Heat Removal*	HR		1.0×10^{-4}
Containment Isolation	CI		2.0×10^{-3}

*Sequence element is flood susceptible.

Table 6.3 Example Problem Screening Data

Flood occurrence frequency	1×10^{-5}/hr.
Screening criteria (CRITRA)	0.05

Table 6.4 Example Problem Consequence Category Unflooded Occurrence Frequency

Consequence Category*	Unflooded Occurrence Frequency
2	2.32×10^{-10}
3	3.23×10^{-8}
4	3.02×10^{-2}

*Consequence category #1 is not screened in this example.

Table 6.5 ESP Input Group Keywords

Input Group	Keyword
1	SYSTEM
2	FLOOD
3	CATEGORY
4	SEQUENCE

- * CATEGORY
  [CONSEQUENCE CATEGORY DESCRIPTIONS AND UNFLOODED OCCURRENCE FREQUENCY]
  END
  [COMMENT CARDS]

- * SEQUENCE
  [ACCIDENT SEQUENCE DESCRIPTIONS]
  END
  [COMMENT CARDS]

- STOP CARD

Figure 6.2 shows an ESP input deck in the proper order.

6.2 Title Card

The first card in the ESP input deck must be the title card. The user can use any alphanumeric information (up to 80 characters) describing the set of accident sequences to be screened. Only one title card is used and it must be the first card in the data deck. If no title is desired, a blank card must be supplied.

6.3 Input Group 1, SYSTEM (Table 6.6, Figure 6.3)

The SYSTEM input group describes the initiating events and branching operators contained in the accident sequences. Input Group 1 defines three variables.

NAME is a 24-character description of an initiating event or branching operator. CODE is a two-character code name for the initiating event or branching operator. FREQ is the occurrence frequency or failure on demand probability of the initiating event/branching operator described by NAME. One card is supplied for each unique initiating event or branching operator in the accident sequences.

6.4 Input Group 2, FLOOD (Table 6.7, Figure 6.4)

Input Group 2 provides the occurrence frequency of the flood to be analyzed and the screening criterion. This input group also identifies flood susceptible initiating events or branching operators. FFREQ is the occurrence frequency of the flood being analyzed and CRITRA is the screening criteria used to determine whether or not an accident sequence is significant (Section 4.2). FFREQ and CRITRA are input on the first input card in Input Group 2.

FCODE is the two-character code name (CODE from Input Group 1) of the initiating events or branching operators that are considered flood

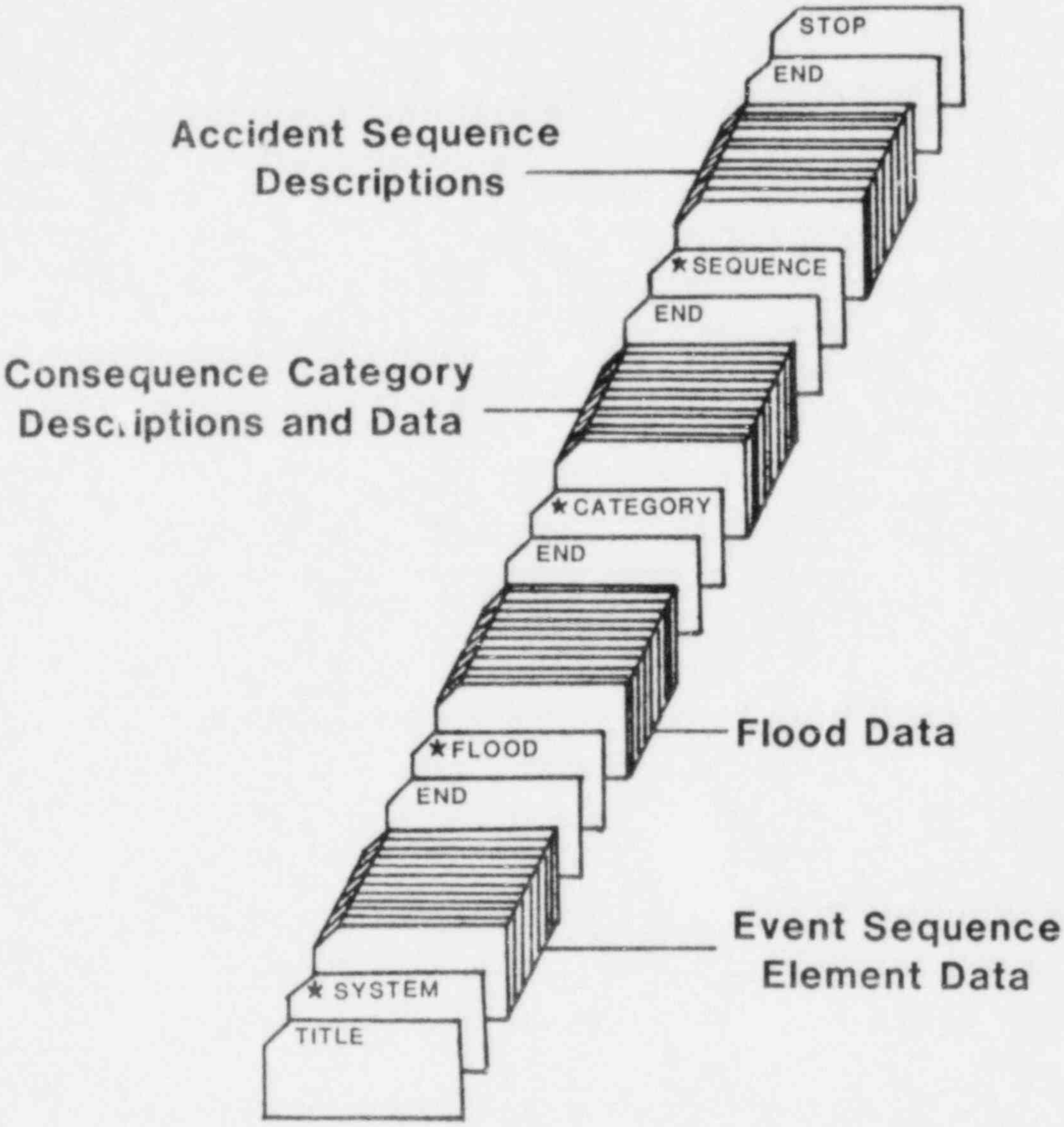

Figure 6.2 ESP Input Deck Using All Input Groups

Table 6.6 Input Format for Input Group 1, SYSTEM

Variable Name	Format	Card Columns	Adjustment
NAME	3A8	1-24	LEFT
CODE	A2	30-31	LEFT
FREQ	E10.6	35-44	RIGHT

```
         10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* SYSTEM
PIPE BREAK                   PB        3.E-4
EMERGENCY CORE COOLING       EC       9.5E-5
POST ACC RAD REMOVAL         RR       6.0E-3
POST ACC HEAT REMOVAL        HR       1.0E-4
CONTAINMENT ISOLATION        CI       2.0E-3
END
```

Figure 6.3 SYSTEM Input Example

Table 6.7 Input Format for Input Group 2, FLOOD

Variable Name	Format	Card Columns	Adjustment
(first card)			
FFREQ	F10.0	1-10	RIGHT
CRITRA	F10.0	11-20	RIGHT
(successive cards)			
FCODE	A2	1-2	LEFT

```
         10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* FLOOD
   1.0E-5       0.05
EC
HR
END
```

Figure 6.4 Flood Input Example

susceptible. One code name per card is supplied for each flood susceptible initiating event or branching operator.

6.5 Input Group 3, CATEGORY (Table 6.8, Figure 6.5)

Input Group 3 identifies the consequence categories for which accident sequences are screened. ESP screens only those accident sequences that result in a consequence category identified in Input Group 3. CAT is the identification number of the consequence category for which accident sequence screening is to be performed and FCAT is the unflooded occurrence frequency of that consequence category. One pair of consequence category number and consequence category unflooded occurrence frequency is supplied per card for each set of consequence category accident sequences to be screened.

6.6 Input Group 4, SEQUENCE (Table 6.9, Figure 6.6)

Input Group 4 describes the accident sequences to be screened. Accident sequences leading to consequence categories not listed in Input Group 3 may also be input, but they will not be screened. One accident sequence per card is described. Three variables describe each accident sequence. ICAT is the consequence category identification number to which the accident sequence belongs. NLOA is the number of elements in the accident sequence and the SEQN(I) are the code names of the initiating event and branching operators that compose the accident sequence.

6.7 STOP Card

The final card in the ESP input deck is a STOP card, with STOP beginning in card column one. The STOP card signals the computer program that the input deck is complete.

Table 6.8 Input Format for Input Group 3, CATEGORY

Variable Name	Format	Card Columns	Adjustment
CAT	I10	1-10	RIGHT
FCAT	F10.0	11-20	RIGHT

```
         10        20        30        40        50        60
-- -+----\----+----\----+----\----+----\----+----\----+----\
* CATEGORY
         2  2.32E-10
         3   3.23E-8
         4   3.02E-4
END
```

Figure 6.5 CATEGORY Input Example

Table 6.9 Input Format for Input Group 4, SEQUENCE

Variable Name	Format	Card Columns	Adjustment
ICAT	I5	1-5	RIGHT
NLOA	I5	6-10	RIGHT
SEQN	21(A2,1X)	16-17	LEFT
		19-20	
		•	
		•	
		•	
		79-80	

```
        10        20        30        40        50        60
----+----\----+----\----+----\---------\----+----\----+----\
* SEQUENCE
    4    1    PB
    4    2    PB CI
    4    2    PB HR
    3    3    PB HR CI
    4    2    PB RR
    3    3    PB RR CI
    3    3    PB RR HR
    2    4    PB RR HR CI
    3    2    PB EC
    2    3    PB EC CI
    2    3    PB EC HR
    2    4    PB EC HR CI
    2    3    PB EC RR
    2    4    PB EC RR CI
    2    4    PB EC RR HR
    1    5    PB EC RR HR CI

END
STOP
```

Figure 6.6 SEQUENCE Input Example

7. ESP: OUTPUT DESCRIPTION

This section describes the output received from an ESP run. The first item output from ESP is a listing of the input data, followed by tabulation of the parameters for the run (Figure 7.1). This is followed by the results of the ESP input error checking routines for the four input groups (Figure 7.2). If an error is found, ESP gives a description of the type error. Appendix B defines these error messages.

Following the error check information are the results of the ESP accident sequence screening (Figure 7.3). ESP displays the following information for each consequence category screened:

1. the accident sequences considered significant due to flood effects, if any exist,

2. the relative rank of the significant accident sequences,

3. the flood susceptible initiating events and branching operators in the significant accident sequences, and

4. an estimate of the total occurrence frequency, including unflooded and flooded effects, of the consequence category.

```
                                                                   VERSION - 1 - DEC, 1981
                       * * * * * * * * * * * * * * * * * * * * * * * * * *
                        - - - - - - - - - - - - - - - - - - - - - - - - -
                       *                                                 *
                        -   ESP - - EVENT SEQUENCE SCREENING PROGRAM    -
                       *                                                 *
                        -      A PROGRAM FOR IDENTIFYING POTENTIALLY    -
                       *       SIGNIFICANT ACCIDENT SEQUENCES FOR        *
                        -               FLOOD ANALYSIS                  -
                       *                                                 *
                        - - - - - - - - - - - - - - - - - - - - - - - - -
                       * * * * * * * * * * * * * * * * * * * * * * * * * *


>>  LISTING OF INPUT DATA DECK  <<

ESP SAMPLE PROBLEM
* SYSTEM
PIPE BREAK                     PB         3.E-4
EMERGENCY CORE COOLING         EC        9.5E-5
POST ACC RAD REMOVAL           RR        6.0E-3
POST ACC HEAT REMOVAL          HR        1.0E-4
CONTAINMENT ISOLATION          CI        2.0E-3
END
* FLOOD
    1.0E-5       0.05
EC
HR
END
* CATEGORY
         2  2.32E-10
         3   3.23E-8
         4   3.02E-4
END
```

Figure 7.1 ESP Output Example: Input and Calculated Parameters

```
* SEQUENCE
    4     1        PB
    4     2        PB CI
    4     2        PB HR
    3     3        PB HR CI
    4     2        PB RR
    3     3        PB RR CI
    3     3        PB RR HR
    2     4        PB RR HR CI
    3     2        PB EC
    2     3        PB EC CI
    2     3        PB EC HR
    2     4        PB EC HR CI
    2     3        PB EC RR
    2     4        PB EC RR CI
    2     4        PB EC RR HR
    1     5        PB EC RR HR CI
END
STOP


 - - -    ESP PARAMETERS FOR THIS RUN    - - -

NUMBER OF SYSTEMS, NSYS ---------------------------        5

NUMBER OF FLOOD SUSCEPTIBLE SYSTEMS, NSUS ---------        2

NUMBER OF CATEGORIES, NCAT ------------------------        3

NUMBER OF EVENT SEQUENCES, NSEQ -------------------       15
```

Figure 7.1 Continued

```
INPUT CHECK: SYSTEM NAMES, CODES, AND FREQUENCIES

PIPE BREAK                  PB   3.0000E-04
EMERGENCY CORE COOLING      EC   9.5000E-05
POST ACC RAD REMOVAL        RR   6.0000E-03
POST ACC HEAT REMOVAL       HR   1.0000E-04
CONTAINMENT ISOLATION       CI   2.0000E-03


INPUT CHECK: FLOOD FREQUENCY, CRITERION,  AND FLOOD-SUSCEPTIBLE SYSTEM CODES

FLOOD FREQUENCY: 1.0000E-05    CRITERIA: 5.0000E-02
EC
HR


INPUT CHECK: CATEGORY NUMBERS AND FREQUENCIES

    2     2.3200E-10
    3     3.2300E-08
    4     3.0200E-04


INPUT CHECK: EVENT SEQUENCES

    4    1     PB
    4    2     PB  CI
    4    2     PB  HR
    3    3     PB  HR  CI
    4    2     PB  RR
    3    3     PB  RR  CI
    3    3     PB  RR  HR
    2    4     PB  RR  HR  CI
    3    2     PB  EC
    2    3     PB  EC  CI
    2    3     PB  EC  HR
    2    4     PB  EC  HR  CI
    2    3     PB  EC  RR
    2    4     PB  EC  RR  CI
    2    4     PB  EC  RR  HR
    1    5     PB  EC  RR  HR  CI
```

Figure 7.2 ESP Output Example: Input Check of Data

CONSEQUENCE CATEGORY	SIGNIFICANT ACCIDENT SEQUENCES DUE TO FLOOD EFFECTS	UNFLOODED OCCURRENCE FREQUENCY	FLOODED OCCURRENCE FREQUENCY	RANK
2	PB EC HR	2.850000E-12	3.000000E-09	1
2	PB EC RR	1.710000E-10	1.799999E-11	2
2	PB EC RR HR	1.710000E-14	1.799999E-11	2

FLOOD SUSCEPTIBLE ELEMENTS IN
SIGNIFICANT ACCIDENT SEQUENCES
FOR CATEGORY 2

CODE - SYSTEM DESCRIPTION -

EC EMERGENCY CORE COOLINC
HR POST ACC HEAT REMOVAL

ESTIMATED OCCURRENCE FREQLENCY
OF CONSEQUENCE CATEGORY 2

UNFLOODED CONTRIBUTION: 2.320000E-10
FLOOD CONTRIBUTION: 3.048060E-09
ESTIMATED TOTAL: 3.280060E-09

Figure 7.3 ESP Output Example: Accident Sequence Screening Results

CONSEQUENCE CATEGORY	SIGNIFICANT ACCIDENT SEQUENCES DUE TO FLOOD EFFECTS	UNFLOODED OCCURRENCE FREQUENCY	FLOODED OCCURRENCE FREQUENCY	RANK
3	PB EC	2.850000E-08	3.000000E-09	1

FLOOD SUSCEPTIBLE ELEMENTS IN
SIGNIFICANT ACCIDENT SEQUENCES
FOR CATEGORY 3

CODE - SYSTEM DESCRIPTION -

EC EMERGENCY CORE COOLING

ESTIMATED OCCURRENCE FREQUENCY
OF CONSEQUENCE CATEGORY 3

UNFLOODED CONTRIBUTION:	3.230000E-08
FLOOD CONTRIBUTION:	3.023709E-09
ESTIMATED TOTAL:	3.532370E-09

Figure 7.3 Continued

```
* * * * * * * * * * * * * * * *
* * * * * * * * * * * * * * * *

              NO
SIGNIFICANT ACCIDENT SEQUENCES
      FOR CATEGORY     4

* * * * * * * * * * * * * * * *
* * * * * * * * * * * * * * * *
```

```
* * * * * * * * * * * * * * * * * * * *

    ESTIMATED OCCURRENCE FREQUENCY
     OF CONSEQUENCE CATEGORY     4

UNFLOODED CONTRIBUTION:  3.020000E-04
FLOOD CONTRIBUTION:      2.999699E-09
ESTIMATED TOTAL:         3.020028E-04

* * * * * * * * * * * * * * * * * * * *
```

Figure 7.3 Continued

NOAH

A Computer Program for Qualitative
Flood Analysis

8. NOAH: INTRODUCTION

The NOAH computer program aids in assessing the impact of floods on nuclear power plant systems by identifying flooded system minimal cut sets as the flood level increases. A flooded minimal cut set is a minimal cut set that has all its components submerged by the flood. If all the components in the minimal cut set are failed upon submersion, then the flood is a single event that results in the system failure of interest.

The system failures of interest in flood risk analysis are the important branching operator failures that are identified by the ESP computer program. The NOAH computer program is intended to provide detailed flood analysis of important system (branching operator) failures using the system fault tree from the existing risk assessment and the flood profile within the plant. The primary output from NOAH is the order of component submersion and the flooded minimal cut sets for the flood levels of interest. This output can then be used in a quantitative evaluation to determine the system failure probability as a function of flood level. The NOAH computer program can also identify flood protection sets which provide information for making safety-related design changes to protect against flood events.

NOAH is written in FORTRAN IV for the IBM 360/370 computers and can be used in conjunction with the KITT-2[4] computer program to determine the quantitative effect the flood has on system reliability.

Concepts and definitions of the NOAH methodology are presented in Section 9. Section 10 provides a general description of the program and input groups are described in detail in Section 11. Section 12 describes the NOAH program output.

Appendix E provides a detailed programmer's guide for the NOAH program. Included in this guide are descriptions of NOAH subroutines, major program variables, diagnostic information and subroutine calling sequences. A sample problem is given in Appendix F. Error messages generated by NOAH are explained in Appendix G, and Appendix H lists the required job control language. A condensed version of the NOAH input parameters and their formats is given in Appendix I.

9. NOAH: CONCEPTS AND DEFINITIONS

9.1 Flooded Minimal Cut Sets

The NOAH methodology identifies flooded minimal cut sets from a system fault tree. These flooded minimal cut sets are of interest since the occurrence of a single minimal cut set guarantees the occurrence of the system failure of interest.

9.2 Critical Flood Level

The critical flood level is defined as the minimum flood level where all the components contained in at least one minimal cut set are submerged by the flood. This is the minimum flood level where the system failure of interest can be directly caused by the flood.

9.3 Partially Flooded Minimal Cut Sets

The NOAH computer program can also identify partially flooded minimal cut sets. These are minimal cut sets in which all but one or two components are submerged. They are potentially significant contributors to a system failure since they require only one or two components to fail, in addition to flood effects, to cause a system failure. The NOAH computer program only identifies partially flooded minimal cut sets for floods below the system critical flood level.

9.4 Flood Protection Sets

A flood protection set is a group of components that, if they all are unfailed, guarantee the system is not failed as a result of any flooded minimal cut set (or partially flooded minimal cut set). NOAH synthesizes the flood protection sets from the second order and higher flooded minimal cut sets (or partially flooded minimal cut sets) for each flood level. Any flooded one-event minimal cut sets are identified by NOAH but are not included in the flood protection sets. If they exist, flooded one-event cut sets should be examined as a first priority for flood protection efforts. The flood protection sets provide valuable information for determining where additional system upgrading and flood protection efforts will be most effective.

9.5 Flood Description

The methodology used in the NOAH computer program is independent of the source of the flood being considered. Regardless of the source, the level of the resulting flood can be characterized as a function of time. Figure 9.1 shows a hypothetical flood level profile. In the general case, the flood level profile will show an

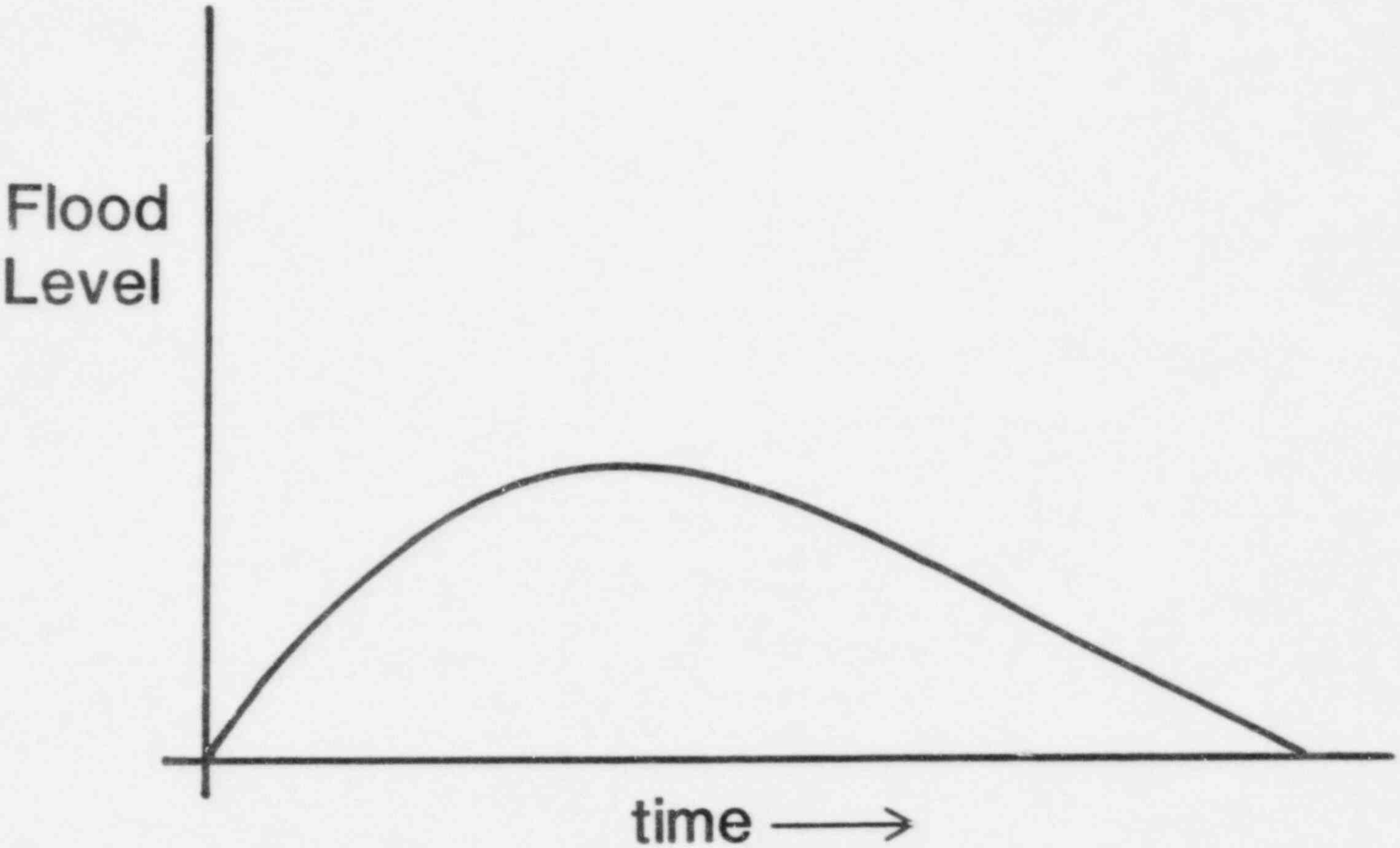

Figure 9.1 Hypothetical Flood Level Profile

increase in flood level from the onset of the event until it attains a maximum value, followed by a period of decreasing level as the flood recedes. Components that are affected by a given flood event can be identified by their vulnerability elevation and the given flood level profile.

A discretized flood level profile is one of two options that can be used as input to the NOAH computer program. Figure 9.2 shows an example of this discretized profile. The discretized profile reflects the assumption that once the flood has reached a discrete level in the plant, that entire level is flooded. The NOAH computer program will also accept a linearly increasing flood profile (Figure 9.3). NOAH works only with the increasing portion of the flood level profile.

From the flood profile, the time at which a component is submerged by the flood can be determined. This time point serves as the component's phase boundary(4) where its reliability characteristics change from those representing an unflooded component to those representing the component in the flooded state.

9.6 NOAH Flood Simulation

The NOAH computer program combines three basic input groups to perform a qualitative flood simulation. These are:

1. The system fault tree - This fault tree defines the system failure of interest and the failure logic associated with the system failure.

2. Component vulnerability elevations - This elevation is the lowest level that the flood must surpass in order to affect the component.

3. Flood levels to be analyzed - These levels are the discrete flood levels used by the NOAH computer program in the flood analysis. All basic events whose vulnerability elevations are below a particular flood level to be analyzed are considered submerged when minimal cut sets are determined.

The general flow of the NOAH computer program is shown in Figure 9.4. NOAH combines the various inputs to develop a threaded pseudo-binary tree model(5) for the portions of the system fault tree that are submerged at flood level "i". This model is tested to determine if any minimal cut sets are submerged at flood level "i". If no minimal cut sets are submerged, NOAH constructs and tests the model for the next flood increment. When the test is true, that is,

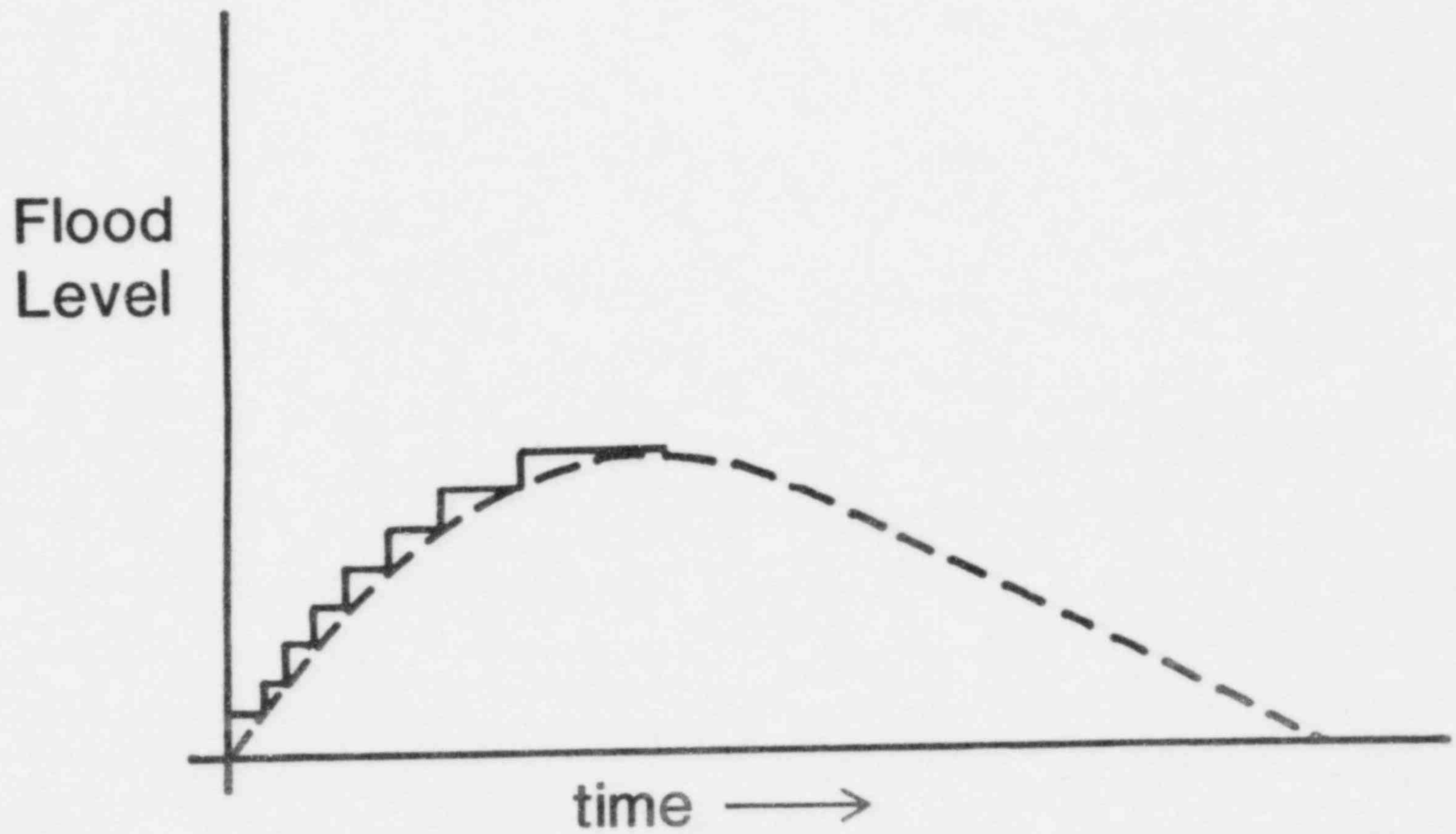

Figure 9.2 Discretized Flood Level Profile

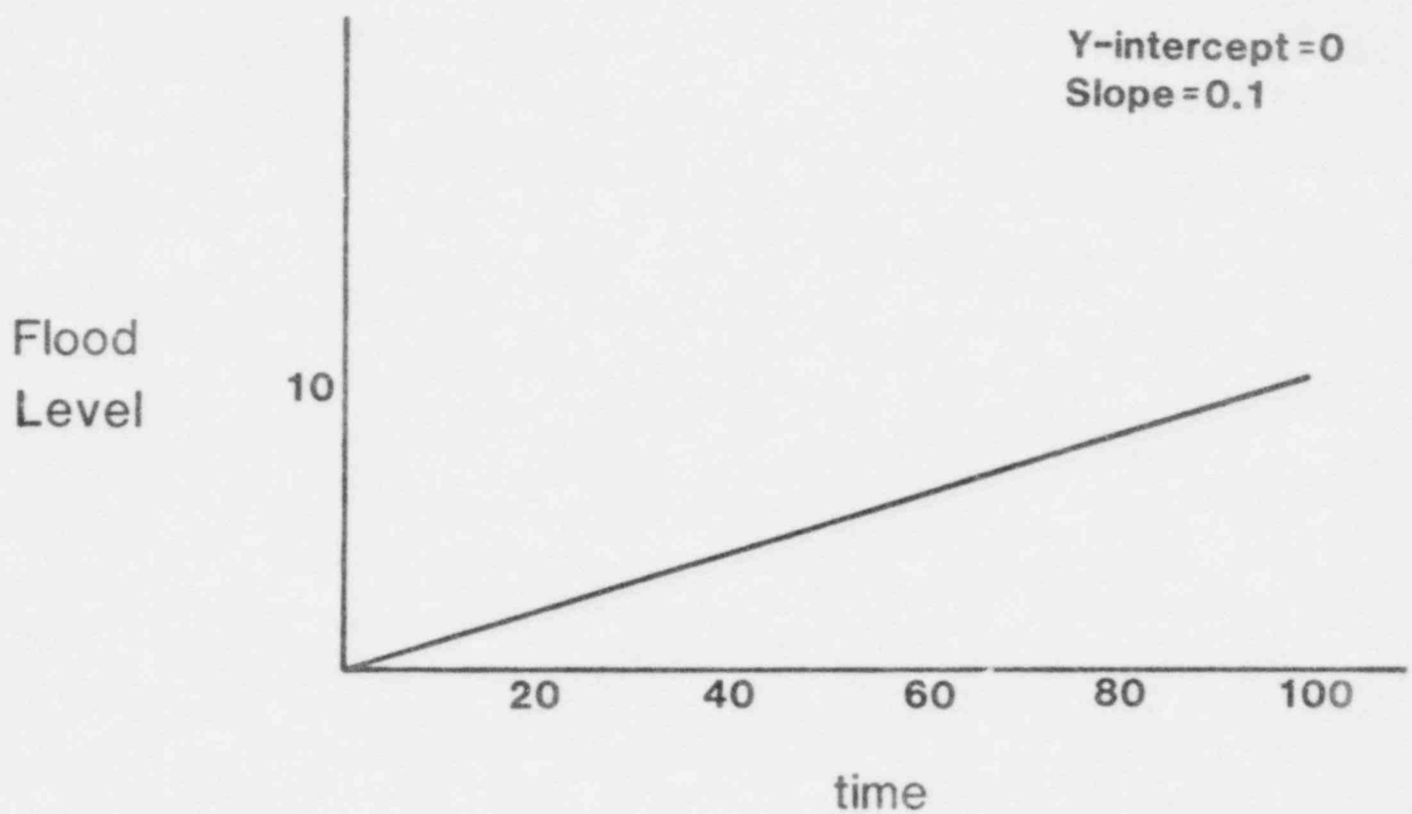

Figure 9.3 Linear Flood Level Profile

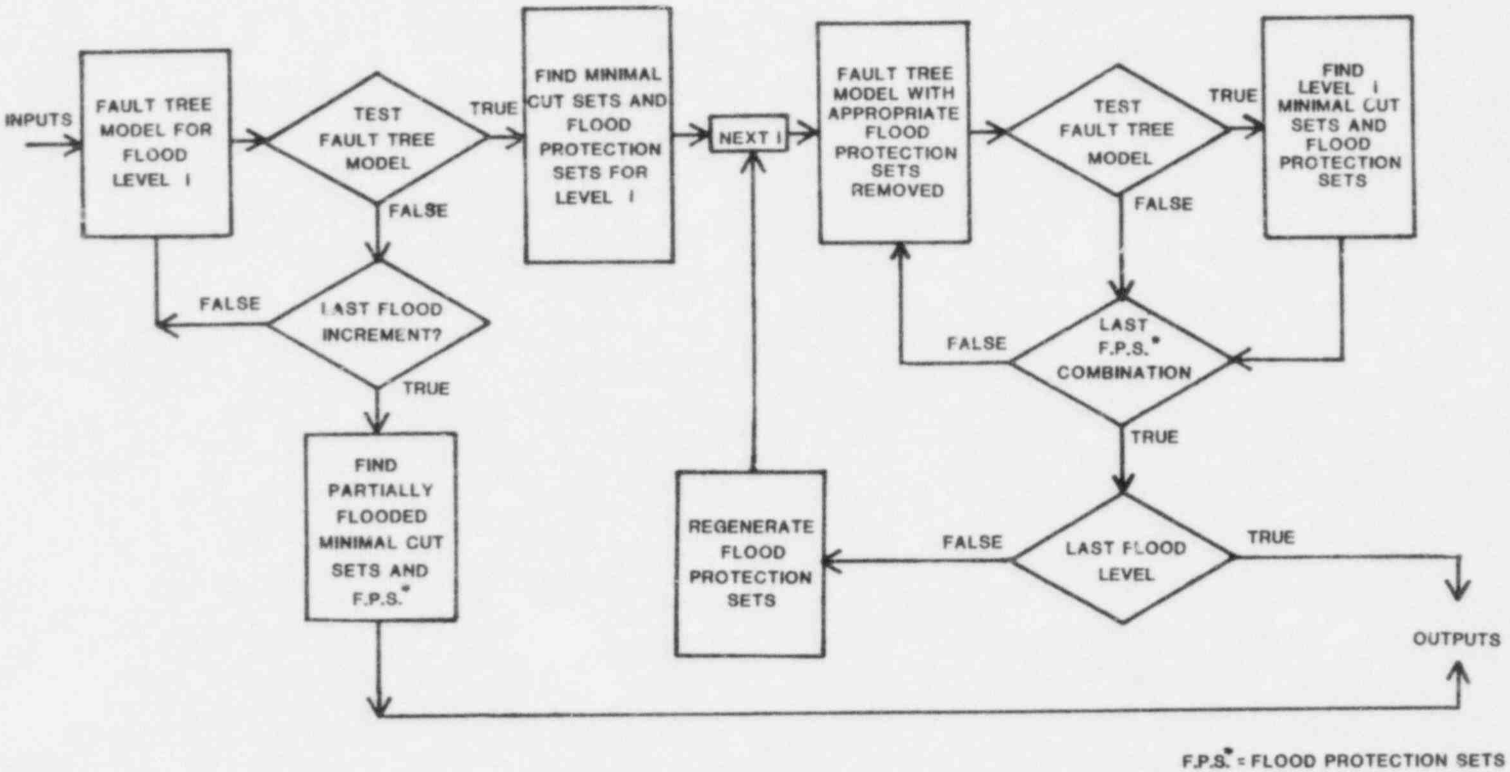

Figure 9.4 NOAH Computer Program Simplified Flowchart

one or more minimal cut sets are submerged, NOAH determines and stores the flooded minimal cut sets at that level. If additional levels are still to be analyzed, NOAH synthesizes the flood protection sets from the flooded minimal cut sets and uses these protection sets to screen out previously found flooded minimal cut sets from the fault tree before analyzing the next flood level.

10. NOAH: PROGRAM DESCRIPTION

NOAH uses a system fault tree to perform a flood simulation to determine the order of component and minimal cut set submersion. Information input to NOAH includes:

1. the job title to be printed with the output,
2. control information specifying how the flood simulation is to be performed,
3. descriptions of the flood levels to be analyzed,
4. a description of the system fault tree,
5. vulnerability elevations for the components in the system fault tree,
6. basic event failure and repair data (optional),
7. basic event search information (optional), and
8. a description of the flood level profile versus time (optional).

The fault tree description input to NOAH is identical to the fault tree description input to the MOCUS[6] and PREP[4] computer programs. Section 11 describes NOAH input in detail.

The output of the NOAH program depends on whether the analysis reaches the critical flood level. If the critical flood level is found, the basic output consists of:

1. The critical flood level - This is the flood level where the first flooded minimal cut set is found. The critical flood level can be determined without determining flooded minimal cut sets.
2. Flooded minimal cut sets for each flood level - This list identifies the minimal cut sets that have all their components submerged at each flood level increment.
3. The flood protection sets for each level - This list identifies groups of components that, if they are all made invulnerable to floods, would prevent the system as modeled

from failing as a result of a flood. (Flood protection sets are optional output and must be requested by the user.)

4. The submerged components for each flood level - This list identifies the components that are submerged within each flood level increment, that is, the order of component submersion.

If the analysis does not reach the critical flood level, the basic output consists of:

1. The partially flooded minimal cut sets - This list identifies minimal cut sets that have all but one or two of their components submerged when the highest flood level for the analysis is reached. Partially flooded minimal cut sets are not determined if flooded minimal cut sets are found during the flood analysis.

2. The flood protection sets for the maximum level analyzed - This list identifies groups of components that, if they are all made invulnerable to floods, would prevent the system as modeled from failing as a result of a flood and single or double random failures. (Flood protection sets are optional output and must be requested by the user.)

3. The submerged components for the maximum level analyzed.

In addition to the flood simulation, the NOAH computer program has the capability to identify a specified component's role in the flood analysis results. For example, if the analyst requests the role of pump A in the flood analysis results, NOAH will identify and list the following information for pump A:

1. The flooded or partially flooded minimal cut sets which contain pump A. These minimal cut sets are grouped according to the flood level where the minimal cut set is submerged.

2. The vulnerability elevation of each component in the minimal cut sets.

11. NOAH: INPUT DESCRIPTION

This section provides a description of the required and optional inputs to NOAH. NOAH input is divided into distinct groups and each input group is described separately. The following sections give the variable or array names, the proper formats and the purpose of the input for each input group. Appendix I describes NOAH input formats in brief.

Figure 11.1 gives an example problem fault tree. Table 11.1 lists vulnerability elevations for the basic events in the example fault tree and Figure 11.2 shows a discretized flood level profile for the example problem. This example problem provides the input example for each of the input group descriptions in Section 11 and the output example in Section 12.

11.1 NOAH Input Deck Construction

Each input group begins with a special control card of the form:

Column 1	Column 2	Column 3
*	Blank	KEYWORD

with KEYWORD replaced by the appropriate input group identifying name. Table 11.2 gives a listing of all input group keywords. The conclusion of each input group is indicated by a card with the word END beginning in the first card column. Any number of comment cards may precede the input groups. The first such card (80 characters maximum in length) appearing in the input deck is used as a title card and its contents are printed on each page of output. The title card is a required input.

The complete input deck must be assembled in the following order:

- TITLE CARD
  [COMMENT CARDS]

- * CONTROL
  [CONTROL PARAMETERS]
  END
  [COMMENT CARDS]

- * KEY
  [FLOOD LEVEL KEYWORDS AND KEYWORD DESCRIPTIONS]
  END
  [COMMENT CARDS]

- * TREE
  [FAULT TREE DESCRIPTION]
  END
  [COMMENT CARDS]

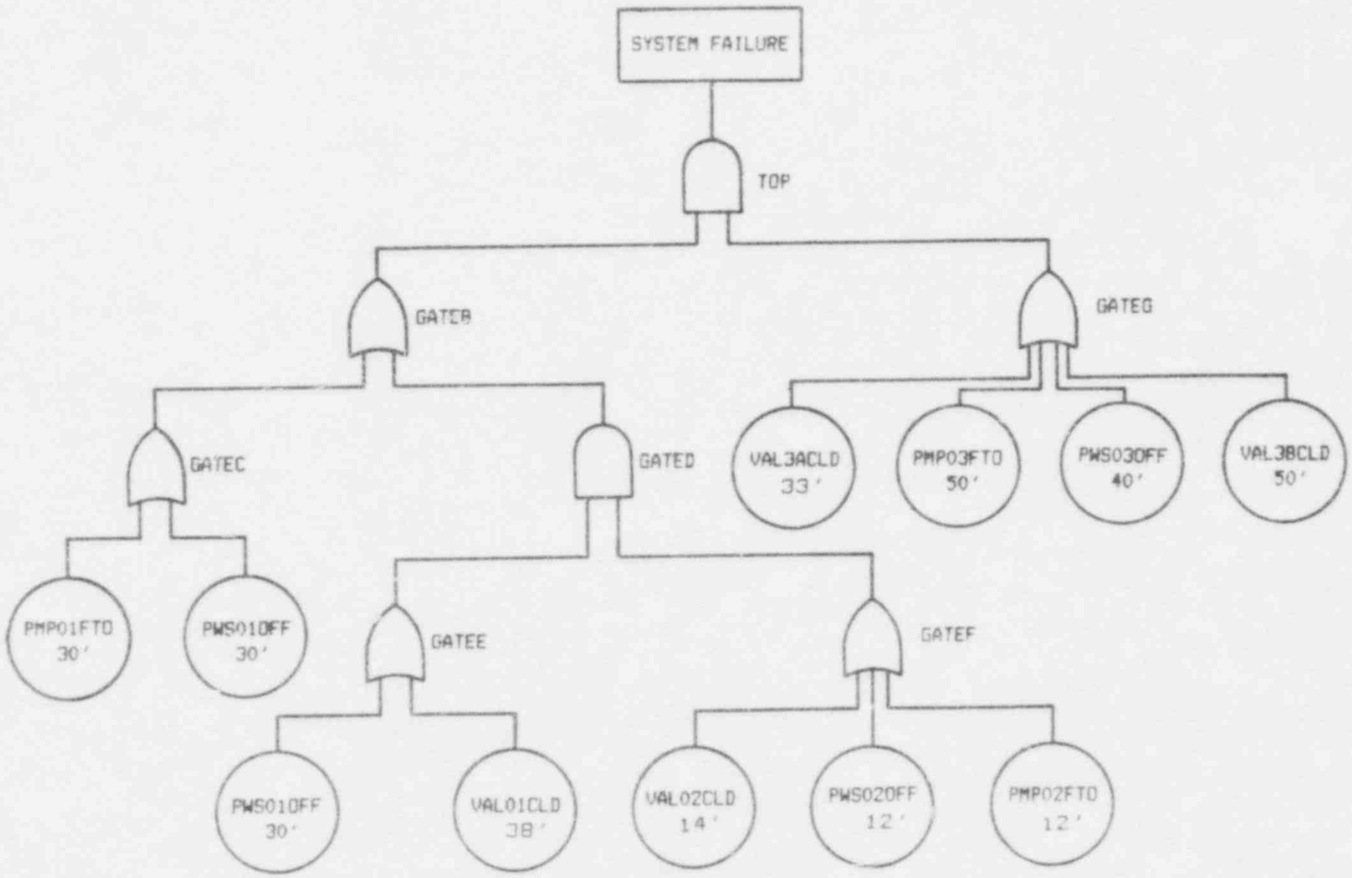

Figure 11.1 Example Problem Fault Tree

Table 11.1 Basic Event Vulnerability Elevations for the Example Problem

Basic Event	Elevation
PMP01FTO	30
PWS01OFF	30
VAL01CLD	38
VAL02CLD	14
PWS02OFF	12
PMP02FTO	12
VAL3ACLD	33
PMP03FTO	50
PWS03OFF	40
VAL3BCLD	50

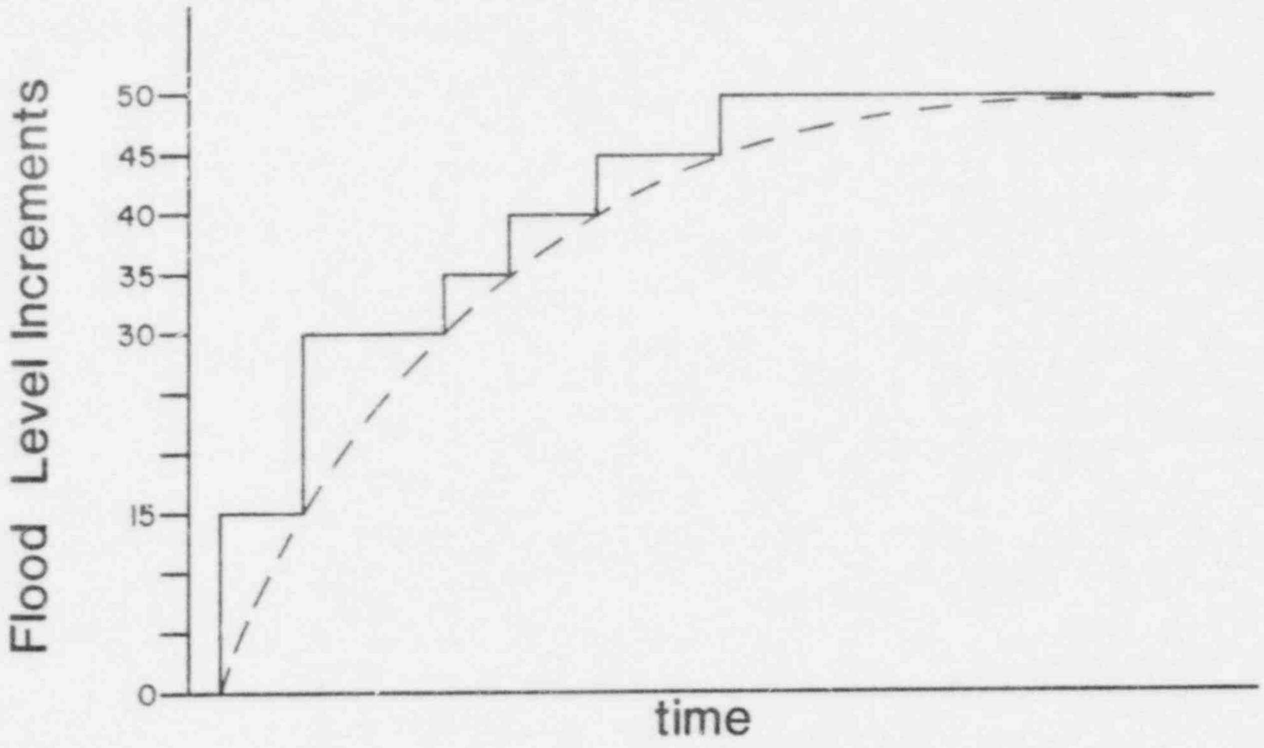

Figure 11.2 Discretized Flood Level Profile for the Example Problem

Table 11.2 NOAH Input Group Keywords

Input Group	Keyword
1	CONTROL
2	KEY
3	TREE
4	ELEVATION
5	HOUSE
6	SEARCH
7	PROFILE

- * ELEVATION
  [BASIC EVENT ELEVATIONS AND FAILURE/REPAIR INFORMATION]
  END
  [COMMENT CARDS]

- * HOUSE
  [HOUSE EVENT INFORMATION]
  END
  [COMMENT CARDS]

- * SEARCH
  [BASIC EVENT SEARCH INFORMATION]
  END
  [COMMENT CARDS]

- * PROFILE
  [FLOOD PROFILE VERSUS TIME]
  END

- STOP CARD

Figure 11.3 shows a NOAH input deck using all input groups in the proper order.

11.2 Title Card

Each fault tree input to NOAH must be preceded by a title card. The title can use any alphanumeric information (up to 80 characters) describing the flood analysis. Only one title card is used and must be placed as the first card in the data deck. If no title is desired, a blank card must be supplied.

11.3 Input Group 1, CONTROL (Table 11.3, Figure 11.4)

Input Group 1 is input to NOAH using the NAMELIST* option of FORTRAN. The form of the NAMELIST statement for NOAH is:

Column 1	Column 2	Columns 3-6
blank	$	NOPT

The variables for Input Group 1 are then input as assignment statements beginning in column 8. Each assignment statement is followed by a comma. The final assignment statement is followed by at least one blank space and the NAMELIST statement is concluded by $END. An END card completes the input group.

*NAMELIST is not ANSI standard; however, it is implemented in all the major dialects of FORTRAN, e.g., IBM, CDC, DEC and UNIVAC.

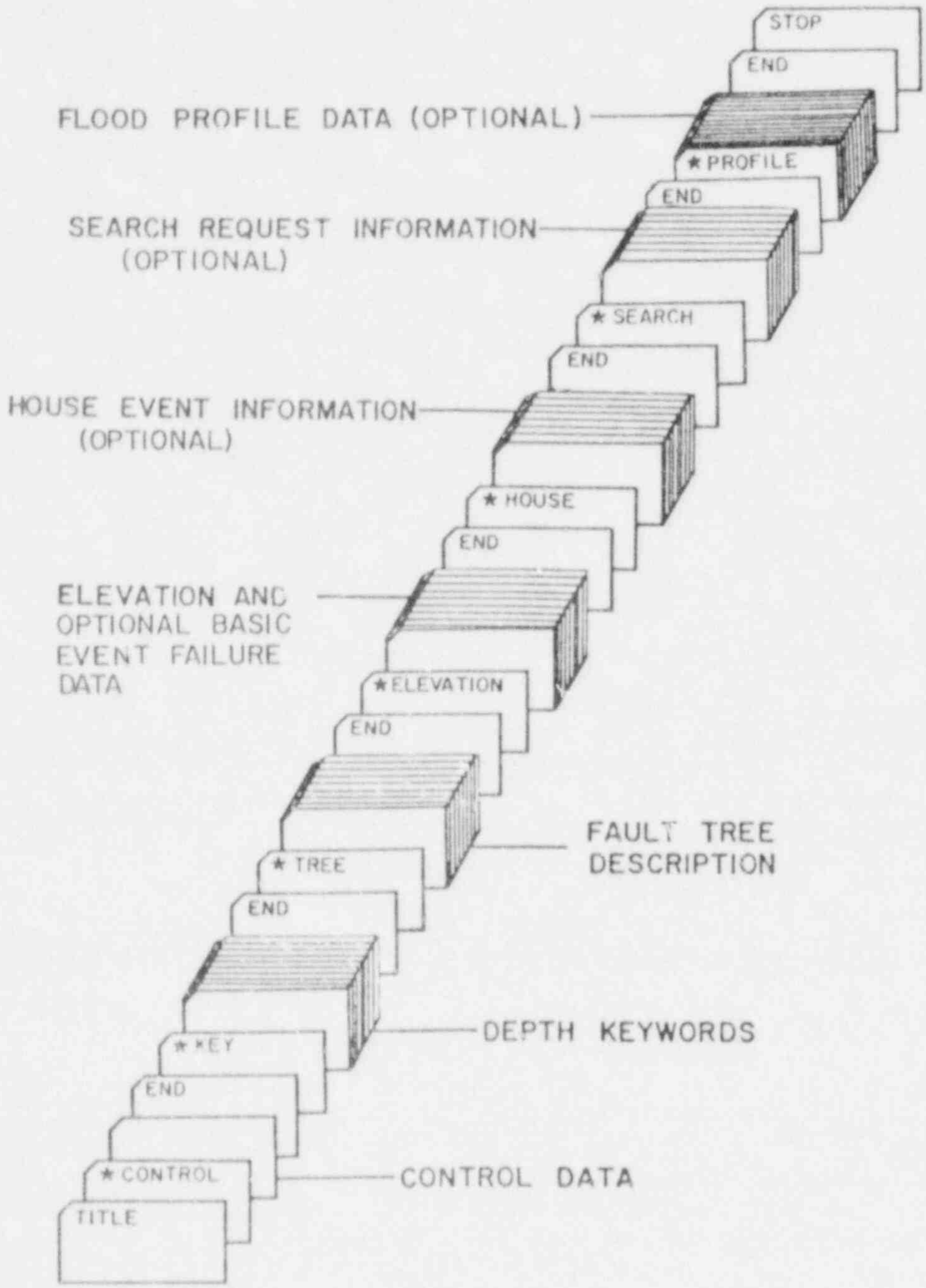

Figure 11.3 NOAH Input Deck Using All the Input Groups

Table 11.3 Input Format for Input Group 1, CONTROL

Variable Name	Variable Type	Default Value	Format
DEPTH	Integer	0	NAMELIST
NDEP	Integer	0	NAMELIST
MAXD	Integer	NDEP	NAMELIST
DOPT	Logical	T	NAMELIST
FIND	Logical	T	NAMELIST
SEEPTH	Logical	F	NAMELIST
ORDER	Integer	10	NAMELIST
MAXIN	Integer	7	NAMELIST
DEEPER	Logical	T	NAMELIST
TIMPT	Integer	0	NAMELIST
NTPT	Integer	0	NAMELIST
DSRCH	Logical	F	NAMELIST
DUMP	Logical	F	NAMELIST
TRACE	Logical	F	NAMELIST
ECHO	Logical	T	NAMELIST
CFD	Logical	F	NAMELIST

```
        10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* CONTROL
 &NOPT NDEP=6, MAXD=8, FIND=T, MAXIN=4, DEEPER=T, DSRCH=T,
       SEEPTH=T, TIMPT=1, NTPT=5, DUMP=F, TRACE=T  &END
END
```

Figure 11.4 CONTROL Input Example

Input Group 1 describes the manner in which the flood simulation will be performed and defines output options. Input Group 1 defines sixteen variables.

NOAH can simulate two types of floods which are defined by the variable DEPTH. DEPTH = 0 signifies a flood beginning at the lowest flood level (flood levels to be analyzed are defined in KEY) and more than one flood level is considered in the simulation. Assigning DEPTH a value other than zero specifies that only one specific flood level is considered submerged in the analysis, with the value of DEPTH identifying the level of interest. For example, DEPTH = 3 indicates that only the third flood level is to be considered submerged. DEPTH defaults to a value of zero.

NDEP identifies the maximum number of flood levels used in the problem description and corresponds to the number of flood levels described in the KEY input group. MAXD identifies the maximum number of flood levels considered submerged in this analysis. NDEP defaults to a value of zero and MAXD defaults to the value of NDEP.

The variable DOPT specifies how flood levels to be analyzed are input to NOAH. When DOPT = T (true), the flood levels are supplied in Input Group 2. DOPT = F (false) indicates that NOAH will divide the maximum flood height into NDEP equal intervals for analysis. When DOPT = F (false), only the maximum height of the flood is supplied in Input Group 2. The default value of DOPT is T (true).

CFD allows the analyst to determine the system's critical flood level without finding flooded minimal cut sets. When CFD = T (true), NOAH will determine the system's critical flood level and then terminate without finding flooded minimal cut sets. CFD = F (false) instructs NOAH to find flooded minimal cut sets. CFD defaults to F (false).

The variable FIND controls the method of flood analysis after MAXD is reached. When FIND = T (true) and MAXD is reached before the critical flood level, NOAH determines the partially flooded minimal cut sets and flood protection sets for the fault tree submerged to MAXD. When FIND = F (false), NOAH terminates the flood analysis after the MAXD level is reached. FIND defaults to F (false).

The variable DEEPER specifies how NOAH reaches the analysis end point. When DEEPER = T (true), the flood level is increased level by level until the flood level reaches MAXD. When DEEPER = F (false), each flood level is considered individually submerged. DEEPER defaults to T (true).

SEEPTH controls the printing of flood protection sets. When SEEPTH = T (true), flood protection sets are output in addition to flooded or partially flooded minimal cut sets. When SEEPTH = F (false), the default value, the flood protection sets are not output.

ORDER sets the maximum size of the flooded minimal cut sets. ORDER is set equal to the maximum size of flooded minimal cut set desired and defaults to 10.

MAXIN sets the maximum number of inputs to any gate in the fault tree description (Input Group 3). NOAH uses this variable internally to determine the array storage required for the analysis. MAXIN defaults to the maximum value of 7. Setting MAXIN to the maximum number of inputs to a gate in the fault tree description will eliminate excess space in the internal arrays and aid the efficiency of program execution.

DSRCH identifies whether or not a search is requested on individual basic events at the end of the analysis. DSRCH = T (true) indicates that a search is requested and that the user will provide Input Group 6. DSRCH defaults to F (false) which indicates no search.

DUMP determines whether or not detailed diagnostic output for error debugging is printed by NOAH. The output includes the starting addresses of the work arrays, a cross reference table between the indices and gates in the fault tree, a threaded pseudo-binary representation(5) of the fault tree and results of intermediate steps in determining the minimal cut sets for a flood level. DUMP = T (true) indicates that this detailed diagnostic information is to be output. The default value of DUMP is F (false), which indicates none of this diagnostic information is output. Appendix F contains sample output from NOAH when DUMP = T.

TRACE defines whether or not a detailed program flow trace is printed by NOAH. TRACE = T (true) indicates that NOAH will print a list of the subroutines as they are used in performing a flood simulation. The default value of TRACE is F (false), indicating no program flow trace is printed.

The ECHO variable is used to obtain a listing of the input arrays as they are filled. This listing is obtained by setting ECHO = T (true), the default value. Setting ECHO = F (false) indicates the input arrays are not listed in the output.

TIMPT indicates the type of time-dependent flood profile supplied in Input Group 7 and whether or not NOAH output is to be punched in a format compatible with the KITT-2 computer program. A linear flood profile must be supplied when TIMPT = -1 and a discretized flood profile must be supplied when TIMPT = 1. TIMPT = 0 indicates that Input Group 7 and the basic event failure data in Input Group 4 are not supplied and that NOAH output will not be punched. The default value of TIMPT is 0. NTPT is the number of time points used to describe a discretized flood profile. NTPT is supplied only when TIMPT = 1. The default value of NTPT is 0.

11.4 Input Group 2, KEY (Table 11.4, Figure 11.5)

Input Group 2 describes the flood levels to be analyzed and contains the flood levels used in the analysis. When DOPT = T (true), one increment must be supplied for each flood level to be analyzed, a total of NDEP levels. When DOPT = F (false), only the highest flood level increment must be supplied. Up to 16 level increments may be input per card in this input group. The increments must be in order from the lowest level to the highest level, with the first level in columns 1-5. The 80-character descriptions of the NDEP flood levels are also input. These must also be in order from lowest to highest flood level, with one level description per card. The levels are contained in the KEY array, and the level descriptions in the KEYDSC array. All NDEP level descriptions must be included regardless of the value of DOPT.

11.5 Input Group 3, TREE (Table 11.5, Figure 11.6)

Input Group 3 is identical to the TREE input group used in the MOCUS[6] and PREP[4] computer programs. Input Group 3 describes the input fault tree, identifying the name of each gate, the gate type, the number of inputs to the gate, and the names of these inputs. One card is required for each gate in the fault tree.

GATE (1,I) is the name of the gate on this input card. GATYP (I) is the gate type. NOAH accepts only "AND" and "OR" gate types. NGI(I) is the number of gate inputs to GATE(1,I), and NCI(I) is the number of basic events input to GATE(1,I). GATE(J,I) (J = 2,8) are the names of the inputs to GATE(1,I). All gates which are inputs must be listed before any basic events. The gate card describing the TOP event (the system failure of interest) must be the first gate card input in the fault tree description.

11.6 Input Group 4, ELEVATION (Table 11.6, Figure 11.7)

Input Group 4 identifies the flood vulnerability elevation assigned to each basic event in the fault tree and the basic event's unflooded and flooded failure rate and mean time to repair. The basic event's failure rates and mean times to repair are required only if TIMPT = 1 or -1 (Input Group 1). NAME is the eight-character alphanumeric used to identify the basic event in Input Group 3. ELEV is the basic event's vulnerability elevation. LMBDA1 and TAU1 are the basic event's unflooded failure rate and mean time to repair, respectively. LMBDA2 and TAU2 are the flooded failure rate and mean time to repair for the basic event. Table 11.7 provides interpretations for various combinations of component failure rates and repair data. One card is supplied for each basic event in the fault tree.

Table 11.4 Input Format for Input Group 2, KEY

Array Name	Format	Adjustment
KEY	16I5	RIGHT
KEYDSC	10A8	NONE

```
         10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* KEY
   15   30   35   40   45   50
ZERO FEET TO FIFTEEN FEET
FIFTEEN FEET TO THIRTY FEET
THIRTY FEET TO THIRTY-FIVE FEET
THIRTY-FIVE FEET TO FORTY FEET
FORTY FEET TO FORTY-FIVE FEET
FORTY-FIVE FEET TO FIFTY FEET
END
```

Figure 11.5 KEY Input Example

Table 11.5 Input Format for Input Group 3, TREE

Variable Name	Format	Card Columns	Adjustment
GATE(1,I)	A8	1-8	LEFT
GATYP(I)	A3	10-12	LEFT
NGI(I)	I2	14-15	RIGHT
NCI(I)	I2	16-17	RIGHT
GATE(2,I)	A8	19-26	LEFT
GATE(3,I)	A8	28-35	LEFT
GATE(4,I)	A8	37-44	LEFT
GATE(5,I)	A8	46-53	LEFT
GATE(6,I)	A8	55-62	LEFT
GATE(7,I)	A8	64-71	LEFT
GATE(8,I)	A8	73-80	LEFT

```
         10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* TREE
TOP      AND  2 0 GATEB    GATEG
GATEB    OR   2 0 GATEC    GATED
GATEG    OR   0 4 VAL3ACLD PMP03FTO PWS03OFF VAL3BCLD
GATEC    OR   0 2 PMP01FTO PWS01OFF
GATED    AND  2 0 GATEE    GATEF
GATEE    OR   0 2 PWS01OFF VAL01CLD
GATEF    OR   0 3 VAL02CLD PWS02OFF PMP02FTO
END
```

Figure 11.6 TREE Input Example

Table 11.6 Input Format for Input Group 4, ELEVATION

Variable Name	Format	Card Columns	Adjustment
NAME	A8	1-8	LEFT
ELEV	I5	20-24	RIGHT
LMBDA1*	E10.6	30-39	RIGHT
TAU1*	E10.6	40-49	RIGHT
LMBDA2*	E10.6	50-59	RIGHT
TAU2*	E10.6	60-69	RIGHT

*Required only when TIMPT = 1 or -1.

```
         10        20        30        40        50        60        70
----+----\----+----\----+----\----+----\----+----\----+----\----+----\
* ELEVATION
PMP01FTO              30          1.E-3       0.0     1.E-1       0.0
PWS01OFF              30          1.E-3       0.0     5.E-1       0.0
VAL01CLD              36          1.E-4       0.0     3.E-4       0.0
VAL02CLD              14          1.E-4       0.0     3.E-4       0.0
PWS02OFF              12          1.E-3       0.0     5.E-1       0.0
PMP02FTO              12          1.E-3       0.0     1.E-1       0.0
VAL3ACLD              33          1.E-4       0.0     3.E-4       0.0
PMP03FTO              50          1.E-3       0.0     1.E-1       0.0
PWS03OFF              40          1.E-3       0.0     5.E-1       0.0
VAL3BCLD              50          1.E-4       0.0     3.E-4       0.0
END
```

Figure 11.7 ELEVATION Input Example

Table 11.7 Lambda and Tau Interpretations

LAMDA	TAU	Interpretation
Positive	Positive	Component with failure rate LAMDA and constant repair time TAU
Positive	0.0 or blank	Nonrepairable component with failure rate LAMDA
0.0	Between 0.0 and 1.0	Inhibit condition with constant probability of failure TAU

11.7 Input Group 5, HOUSE (Table 11.8, Figure 11.8)

Input Group 5 is required only when house events are used in the input fault tree (Input Group 3). NAME is the house event name as identified in Input Group 3. The variable STATE specifies whether the house event is ON (exists) or OFF (does not exist). The variable FLOOD is input for house events that are expected to change states upon submersion. If the house event is expected to change states, FLOOD is coded FAILS. If the house event is not expected to change states, the variable FLOOD is not input. The example problem defined for illustrating NOAH input and output does not contain house events. An example HOUSE input is given in Figure 11.8 and is not related to the example problem.

11.8 Input Group 6, SEARCH (Table 11.9, Figure 11.9)

Input Group 6 identifies those basic events or components for which a search is requested. BENAM identifies the eight-character basic event name as used in Input Group 3. COMPNT specifies a portion of a basic event name for searches where the portion of the basic event name identifies the specific item of interest in the search. Any portion of the basic event name used must include blank spaces as necessary to maintain the eight-character format of the basic event names. For example:

basic event:	PXV0151C
acceptable COMPNT input:	PXVbbbbb
	bbb0151C
	bbb0151b
	PXVbbbbC

where b represents a blank space and is not typed on the input card. Only one search request may be input per card. Input Group 6 is optional and is supplied only when DSRCH = T in Input Group 1. Figure 11.9 provides input required to conduct two searches, one for basic event PMP01FTO and one for the specific item PWS.

11.9 Input Group 7, PROFILE (Table 11.10, Figure 11.10)

Input Group 7 describes the time-dependent flood profile. This input group is optional and is required only when TIMPT in Input Group 1 is 1 or -1. When TIMPT = 1, a discretized flood profile is specified and NTPT pairs of flood levels (LEVEL) and the times (TIME) to reach these flood levels are required. One LEVEL/TIME pair is supplied per card. When TIMPT = -1, a linear flood profile is specified. The equation for a linearly increasing flood level is:

$$y = mt + b$$

Table 11.8 Input Format for Input Group 5, HOUSE

Variable Name	Format	Card Columns	Adjustment
NAME	A8	1-8	LEFT
HOUSE	A4	11-14	LEFT
FLOOD	A8	21-28	LEFT

```
         10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* HOUSE
BASIC01   OFF       FAILS
EVENT52   ON
EVENT67   ON        FAILS
END
```

This input is not a part of the example problem.

Figure 11.8 HOUSE Input Example

Table 11.9 Input Format for Input Group 6, SEARCH

Variable Name	Format	Card Columns	Adjustment
BENAM	A8	1-8*	LEFT
COMPNT	A8	11-18*	LEFT

*Only one variable may be entered per card. If BENAM is entered in columns 1-8, columns 11-18 must be blank. If COMPNT is entered in columns 11-18, columns 1-8 must be blank.

```
         10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* SEARCH
PMP01FT0
          PWS
END
```

Figure 11.9 SEARCH Input Example

Table 11.10 Input Format for Input Group 7, PROFILE

Variable Name	Format	Card Columns	Adjustment
LEVEL*	I10	1-10	RIGHT
TIME*	F10.2	11-20	RIGHT
SLOPE**	F10.2	1-10	RIGHT
NTRCPT**	F10.2	11-20	RIGHT

*This data is supplied only when TIMPT = 1 in Input Group 1.

**This data is supplied only when TIMPT = -1 in Input Group 1.

```
        10        20        30        40        50        60
----+----\----+----\----+----\----+----\----+----\----+----\
* PROFILE
        10       5.0
        20      15.0
        30      30.0
        40      50.0
        50      80.0
END
```

Figure 11.10 PROFILE Input Example

where

- y = flood level,
- m = time-dependent rate the flood level is increasing (SLOPE),
- t = time, and
- b = initial (t = 0) flood level (NTRCPT).

The slope (SLOPE) and initial flood level (NTRCPT) are required to describe a linear flood profile.

The information supplied in Input Group 7 is used only to determine the time at which a component is submerged by the flood. This time point serves as the components' phase boundary (time point at which the components' reliability characteristics change) when NOAH punches its output in a KITT-2 compatible format.

11.10 STOP Card

The final card in the NOAH input deck is a STOP card, with STOP beginning in card column one. The STOP card signals the computer program that the input deck is complete.

12. NOAH: OUTPUT DESCRIPTION

The NOAH program provides a variety of output. This section describes the output normally received from a NOAH run. Additional output is available for diagnostic purposes.

The first item of output from NOAH is a listing of the input data (Figure 12.1). A program flow trace indicating the subroutines NOAH enters and leaves is printed next. (Not all program flow trace statements are included in the figures.) This is followed by the input and calculated parameters that control the execution of the program (Figure 12.2).

If ECHO = T (true), the contents of arrays are displayed for 6 groups of input:

1. level keys and descriptions (Figure 12.3),
2. fault tree gates and their inputs (Figure 12.4),
3. basic event vulnerability elevations (Figure 12.5),
4. house event data (not provided in the example problem),
5. search request data (if provided, Figure 12.6), and
6. time-dependent flood profile (if provided, Figure 12.7).

This information is followed by statements indicating whether or not errors were found in the data (Figure 12.8).

Following the error check information is the cross-reference table between the internal basic event code used by NOAH and the eight-character basic event names (Figure 12.9). Starting addresses of NOAH internal arrays are then displayed (Figure 12.10). These may be interpreted by referring to Appendix E, Section E.2.?.

The results of the NOAH flood simulation (Figure 12.11) are displayed for each level used in the analysis (Input Group 2). The following information is displayed for each level when flooded minimal cut sets are found:

1. the basic events flooded at that level,
2. the number of flooded minimal cut sets at that level,

3. the total number of flooded minimal cut sets through that level,

4. the flooded minimal cut sets at that level, if any exist, and

5. the vulnerability elevation of each basic event in the flooded minimal cut sets.

The critical flood level is clearly identified as such in the output. The following additional information (Figure 12.12) is provided for each flood level when flood protection sets are requested:

1. the number of flood protection sets at that level,

2. the flood protection sets at that level, and

3. the vulnerability elevation of each basic event in the flood protection sets.

When no flooded minimal cut sets are found through level MAXD and FIND = T (true), NOAH determines partially flooded minimal cut sets and displays the following information:

1. the basic events flooded to level MAXD,

2. the number of partially flooded minimal cut sets at level MAXD,

3. the partially flooded minimal cut sets at level MAXD,

4. the vulnerability elevation of each basic event in the partially flooded minimal cut sets, and

5. the flood level of the highest basic event for each partially flooded minimal cut set.

The flood protection sets synthesized from these partially flooded minimal cut sets are also displayed if requested. NOAH lists the same flood protection information as listed above. Appendix F gives examples of output for the partially flooded minimal cut sets.

Figure 12.13 presents the output of the search requested by the analyst. The output identifies all minimal cut sets that contain the basic event name or portion of the basic event name specified by the analyst in Input Group 6.

Figure 12.14 displays a listing of the input data that NOAH punches on cards for the KITT-2 computer program. The information includes basic event indices, failure rates, repair times, phase boundaries and minimal cut sets. This information can be interpreted using the input description provided in the KITT-2 user's manual[4].

```
                                                                                     VERSION - 2 - DEC, 1981
                  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                   - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                  *                                                             *
                  -   NOAH - - A PROGRAM FOR QUALITATIVE FLOOD ANALYSIS   -
                  *                                                             *
                   - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


>>  LISTING OF INPUT DATA DECK  <<

NOAH USERS MANUAL SAMPLE PROBLEM
* CONTROL
&NOPT
DEPTH=       0,NDEP=       6,MAXD=       6,DOPT=T,FIND=T,SEEPTH=T,ORDER=      10,MAXIN=       4,DEEPER=T,CFD=F,
TIMPT=       1,NTPT=       5,DSRCH=T,DUMP=F,ECHO=T,TRACE=T
&END
END
* KEY
   15   30   35   40   45   50
ZERO FEET TO FIFTEEN FEET
FIFTEEN FEET TO THIRTY FEET
THIRTY FEET TO THIRTY-FIVE FEET
THIRTY-FIVE FEET TO FORTY FEET
FORTY FEET TO FORTY-FIVE FEET
FORTY-FIVE FEET TO FIFTY FEET
END
* TREE
TOP       AND  2 0 GATEB    GATEG
GATEB     OR   2 0 GATEC    GATED
GATEG     OR   0 4 VAL3ACLD PMP03FTO PWS03OFF VAL3BCLD
GATEC     OR   0 2 PMP01FTO PWS01OFF
GATED     AND  2 0 GATEE    GATEF
GATEE     OR   0 2 PWS01OFF VAL01CLD
GATEF     OR   0 3 VAL02CLD PWS02OFF PMP02FTO
END
* ELEVATION
PMP01FTO              30          1.E-3     0.0        1.E-1        0.0
PWS01OFF              30          1.E-3     0.0        5.E-1        0.0
VAL01CLD              38          1.E-4     0.0        3.E-4        0.0
VAL02CLD              14          1.E-4     0.0        3.E-4        0.0
PWS02OFF              12          1.E-3     0.0        5.E-1        0.0
PMP02FTO              12          1.E-3     0.0        1.E-1        0.0
```

Figure 12.1 NOAH Output Example: Input Data Listing

```
>>  LISTING OF INPUT DATA DECK  <<                    NGAH USERS MANUAL SAMPLE PROBLEM

VAL3ACLD              33          1.E-4      0.0        3.E-4        0.0
PMP03FTO              50          1.E-3      0.0        1.E-1        0.0
PWS03OFF              40          1.E-3      0.0        5.E-1        0.0
VAL3BCLD              50          1.E-4      0.0        3.E-4        0.0
END
* SEARCH
PMP01FTO
          PWS
END
* PROFILE
        10         5.0
        20        15.0
        30        30.0
        40        50.0
        50        80.0
END
>>  ENTERING ROUTINE GETNG  <<    \
>>  ENTERING ROUTINE GETNBE  <<    |
 >>  LEAVING ROUTINE  <<           } PROGRAM FLOW TRACE
>>  ENTERING ROUTINE GETMAX  <<    |
 >>  LEAVING ROUTINE  <<          /
```

Figure 12.1 Continued

```
- - - - -    NOAH PARAMETERS FOR THIS RUN    - - - - -                    VERSION - 2 - DEC, 1981

NUMBER OF UNIQUE GATES, NG -------------------------------     7

NUMBER OF UNIQUE BASIC EVENTS, NBE -----------------------    10

TOTAL NUMBER OF GATES, NGATE -----------------------------     7

TOTAL NUMBER OF BASIC EVENTS, NCOMP ----------------------    11

MAXIMUM TIMES AN EVENT APPEARS, MAXREP -------------------     2
   >>  LEAVING ROUTINE  <<

LEVEL TO FLOOD, DEPTH ------------------------------------     0

LEVELS IN THE PLANT, NDEP --------------------------------     6

MAXIMUM LEVEL TO FLOOD, MAXD -----------------------------     6

LEVEL KEY OPTION, DOPT -----------------------------------     T

CONTINUOUS FLOOD, DEEPER ---------------------------------     T

DO FLOOD SCREENING ONLY, CFD -----------------------------     F

FIND SETS AFTER MAXD, FIND -------------------------------     T

PRINT FLOOD PROTECTION SETS, SEEPTH ----------------------     T

ORDER SETS TO FIND, ORDER --------------------------------    10

MAXIMUM INPUTS TO ANY GATE, MAXIN ------------------------     4

SEARCH FOR BE/COMPONENT NAME, DSRCH ----------------------     T

INTERFACE/TIME-POINT PARAMETER, TIMPT --------------------     1

NUMBER TIMEPOINTS FOR INTERFACE, NTPT --------------------     5

DUMP INTERMEDIATE INFORMATION, DUMP ----------------------     F

PRINT CONTENTS OF INPUT ARRAYS, ECHO ---------------------     T

PRINT PROGRAM FLOW TRACE, TRACE --------------------------     T
```

Figure 12.2 NOAH Output Example: Input and Calculated Parameters

```
INPUT CHECK: LEVEL KEYS AND DESCRIPTIONS                                              VERSION - 2 - DEC, 1981

KEY(  1)= '  15'    DESCRIPTION 'ZERO FEET TO FIFTEEN FEET                             '
KEY(  2)= '  30'    DESCRIPTION 'FIFTEEN FEET TO THIRTY FEET                           '
KEY(  3)= '  35'    DESCRIPTION 'THIRTY FEET TO THIRTY-FIVE FEET                       '
KEY(  4)= '  40'    DESCRIPTION 'THIRTY-FIVE FEET TO FORTY FEET                        '
KEY(  5)= '  45'    DESCRIPTION 'FORTY FEET TO FORTY-FIVE FEET                         '
KEY(  6)= '  50'    DESCRIPTION 'FORTY-FIVE FEET TO FIFTY FEET                         '
```

Figure 12.3 NOAH Output Example: Input Check of Level Keys and Descriptions


```
INPUT CHECK: GATES AND THEIR INPUTS                                                   VERSION - 2 - DEC, 1981

NAME        TYPE           INPUTS -> -> ->
TOP          AND     2 0   GATEB      GATEG
GATEB        OR      2 0   GATEC      GATED
GATEG        OR      0 4   VAL3ACLD   PMP03FTO   PWS03OFF   VAL3BCLD
GATEC        OR      0 2   PMP01FTO   PWS01OFF
GATED        AND     2 0   GATEE      GATEF
GATEE        OR      0 2   PWS01OFF   VAL01CLD
GATEF        OR      0 3   VAL02CLD   PWS02OFF   PMP02FTO
```

Figure 12.4 NOAH Output Example: Input Check of Fault Tree Gates and Inputs

INPUT CHECK: COMPONENT ELEVATIONS

VERSION - 2 - DEC, 1981

		------ UNFLOODED ------		------- FLOODED -------	
NAME	ELEVATION	LAMBDA	TAU	LAMBDA	TAU
PMP01FTO	30	0.1000E-02	0.0	0.1000E 00	0.0
PWS01OFF	30	0.1000E-02	0.0	0.5000E 00	0.0
VAL01CLD	38	0.1000E-03	0.0	0.3000E-03	0.0
VAL02CLD	14	0.1000E-03	0.0	0.3000E-03	0.0
PWS02OFF	12	0.1000E-02	0.0	0.5000E 00	0.0
PMP02FTO	12	0.1000E-02	0.0	0.1000E 00	0.0
VAL3ACLD	33	0.1000E-03	0.0	0.3000E-03	0.0
PMP03FTO	50	0.1000E-02	0.0	0.1000E 00	0.0
PWS03OFF	40	0.1000E-02	0.0	0.5000E 00	0.0
VAL3BCLD	50	0.1000E-03	0.0	0.3000E-03	0.0

Figure 12.5 NOAH Output Example: Input Check of Basic Event Vulnerability Elevations, Failure Rates and Mean Downtimes

INPUT CHECK: SEARCH DATA

VERSION - 2 - DEC, 1981

BASIC EVENT	COMPONENT
PMP01FTO	PWS

Figure 12.6 NOAH Output Example: Input Check of SEARCH Data

```
INPUT CHECK: FLOOD TIMES                                    VERSION - 2 - DEC, 1981
  LEVEL          TIME
      10            5.00
      20           15.00
      30           30.00
      40           50.00
      50           80.00
```

Figure 12.7 NOAH Output Example: Input Check of the Flood Profile

```
NO ERRORS DISCOVERED IN ROUTINE INPUT
  >>  ENTERING ROUTINE CHEKIT  <<
   >>  LEAVING ROUTINE  <<
```

PROGRAM FLOW TRACE

```
NO ERRORS DISCOVERED IN ROUTINE CHEKIT
   >>  LEAVING ROUTINE  <<
  >>  ENTERING ROUTINE BUILD  <<
```

PROGRAM FLOW TRACE

Figure 12.8 NOAH Output Example: Conclusion of Input Data Check

```
                                                                          VERSION - 2 - DEC, 1981
        CROSS REFERENCE FOR INTERNAL CODES AND EXTERNAL NAMES AND ELEVATIONS

                   INDEX            NAME             ELEVATION

                      1           VAL3ACLD              33
                      2           PMP03FTO              50
                      3           PWS03OFF              40
                      4           VAL3BCLD              50
                      5           PMP01FTO              30
                      6           PWS01OFF              30
                      7           VAL01CLD              38
                      8           VAL02CLD              14
                      9           PWS02OFF              12
                     10           PMP02FTO              12
>>  LEAVING ROUTINE  <<
```

Figure 12.9 NOAH Output Example: Cross Reference of Internal Codes, External Names and Elevations


```
                          RELATIVE STARTING ADDRESSES FOR ARRAYS              VERSION - 2 - DEC, 1981
     IW( 1)=    50      IW( 8)=   125      IW(15)=   220      IW(22)=   256      IW(29)=   350
     IW( 2)=    68      IW( 9)=   130      IW(16)=   223      IW(23)=   266      IW(30)=   355
     IW( 3)=    82      IW(10)=   132      IW(17)=   226      IW(24)=   269      IW(31)=   400
     IW( 4)=   109      IW(11)=   149      IW(18)=   236      IW(25)=   268      IW(32)=   405
     IW( 5)=   112      IW(12)=   154      IW(19)=   241      IW(26)=   276      IW(33)=   450
     IW( 6)=   115      IW(13)=   214      IW(20)=   243      IW(27)=   300      IW(34)=   458
     IW( 7)=   120      IW(14)=   217      IW(21)=   246      IW(28)=   305      IW(35)=   424
>>  ENTERING ROUTINE DOIT  <<
```

Figure 12.10 NOAH Output Example: Internal Array Starting Addresses

```
**********************************************************************************************************
**********************************************************************************************************
**********************************************************************************************************
*                                                                                                        *
*   FLOODED MINIMAL CUT SET ANALYSIS    FLOODED MINIMAL CUT SET ANALYSIS    FLOODED MINIMAL CUT SET ANALYSIS   *
*                                                                                                        *
**********************************************************************************************************
**********************************************************************************************************
**********************************************************************************************************




         NOAH USERS MANUAL SAMPLE PROBLEM                                          VERSION - 2 - DEC. 1981

FLOODED MINIMAL CUT SET ANALYSIS OF LEVEL    1
KEY IS '  15'              DESCRIPTION='ZERO FEET TO FIFTEEN FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                                 3 BASIC EVENTS ARE FLOODED AT THIS LEVEL

VAL02CLD    PWS02OFF    PMP02FTO

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                           0 NEW MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   1

                          0 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   1

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

Figure 12.11 NOAH Output Example: Flood Simulation Results

```
        NOAH USERS MANUAL SAMPLE PROBLEM                                    VERSION - 2 - DEC, 1981

FLOODED MINIMAL CUT SET ANALYSIS OF LEVEL   2
KEY IS '  30'            DESCRIPTION='FIFTEEN FEET TO THIRTY FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                                    2 BASIC EVENTS ARE FLOODED AT THIS LEVEL

PMP01FTO    PWS01OFF

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                           0 NEW MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   2

                          0 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   2

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

Figure 12.11 Continued

```
          NOAH USERS MANUAL SAMPLE PROBLEM                                   VERSION - 2 - DEC. 1981

FLOODED MINIMAL CUT SET ANALYSIS OF LEVEL    3
KEY IS '  35'             DESCRIPTION='THIRTY FEET TO THIRTY-FIVE FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                                      1 BASIC EVENT IS FLOODED AT THIS LEVEL

VAL3ACLD

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
************************                                                                      ************************
************************      LEVEL   3    KEY '  35'  IS THE CRITICAL FLOOD LEVEL            ************************
* CRITICAL FLOOD LEVEL *                                                                      * CRITICAL FLOOD LEVEL *
************************  THIRTY FEET TO THIRTY-FIVE FEET                                     ************************
************************                                                                      ************************
<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                              2 NEW MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   3

                             2 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   3

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MINIMAL CUT SETS FLOODED AT LEVEL    3

(     1)  PMP01FTO    30    VAL3ACLD   33
(     2)  PWS01OFF    30    VAL3ACLD   33
```

Figure 12.11 Continued

```
          NOAH USERS MANUAL SAMPLE PROBLEM                                  VERSION - 2 - DEC, 1981

FLOODED MINIMAL CUT SET ANALYSIS OF LEVEL   4
KEY IS '  40'             DESCRIPTION='THIRTY-FIVE FEET TO FORTY FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                             2 BASIC EVENTS ARE FLOODED AT THIS LEVEL

PWS03OFF    VAL01CLD

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                          8 NEW MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   4

                         10 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   4

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MINIMAL CUT SETS FLOODED AT LEVEL   4
(    3)  PMP01FTD   30    PWS03OFF   40
(    4)  PWS01OFF   30    PWS03OFF   40
(    5)  VAL01CLD   38    VAL3ACLD   33    VAL02CLD   14
(    6)  VAL01CLD   38    PWS03OFF   40    VAL02CLD   14
(    7)  VAL01CLD   38    VAL3ACLD   33    PWS02OFF   12
(    8)  VAL01CLD   38    VAL3ACLD   33    PMP02FTD   12
(    9)  VAL01CLD   38    PWS03OFF   40    PWS02OFF   12
(   10)  VAL01CLD   38    PWS03OFF   40    PMP02FTD   12
          NOAH USERS MANUAL SAMPLE PROBLEM                                  VERSION - 2 - DEC, 1981

FLOODED MINIMAL CUT SET ANALYSIS OF LEVEL   5
KEY IS '  45'             DESCRIPTION='FORTY FEET TO FORTY-FIVE FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                             0 BASIC EVENTS ARE FLOODED AT THIS LEVEL

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                          0 NEW MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   5

                         10 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   5

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

Figure 12.11 Continued

```
          NOAH USERS MANUAL SAMPLE PROBLEM                                    VERSION - 2 - DEC. 1981

FLOODED MINIMAL CUT SET ANALYSIS OF LEVEL   6
KEY IS '  50'            DESCRIPTION='FORTY-FIVE FEET TO FIFTY FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                                   2 BASIC EVENTS ARE FLOODED AT THIS LEVEL

PMPO3FTO    VAL3BCLD

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                          10 NEW MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   6

                          20 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   6

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MINIMAL CUT SETS FLOODED AT LEVEL   6


(   11)  PMPO1FTO   30    PMPO3FTO   50
(   12)  PMPO1FTO   30    VAL3BCLD   50
(   13)  PWSO1OFF   30    VAL3BCLD   50
(   14)  PWSO1OFF   30    PMPO3FTO   50
(   15)  VALO1CLD   38    PMPO3FTO   50    VALO2CLD   14
(   16)  VALO1CLD   38    VAL3BCLD   50    VALO2CLD   14
(   17)  VALO1CLD   38    PMPO3FTO   50    PWSO2OFF   12
(   18)  VALO1CLD   38    PMPO3FTO   50    PMPO2FTO   12
(   19)  VALO1CLD   38    VAL3BCLD   50    PWSO2OFF   12
(   20)  VALO1CLD   38    VAL3BCLD   50    PMPO2FTO   12
```

Figure 12.11 Continued

```
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
*                                                                                                             *
*    FLOOD PROTECTION SET ANALYSIS        FLOOD PROTECTION SET ANALYSIS        FLOOD PROTECTION SET ANALYSIS   *
*                                                                                                             *
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************


         NOAH USERS MANUAL SAMPLE PROBLEM                                              VERSION - 2 - DEC, 1981

FLOOD PROTECTION SET ANALYSIS OF LEVEL   3
KEY IS '  35'            DESCRIPTION='THIRTY FEET TO THIRTY-FIVE FEET                                   '

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
**********************                                                                  **********************
**********************       LEVEL   3    KEY '  35'  IS THE CRITICAL FLOOD LEVEL        **********************
* CRITICAL FLOOD LEVEL *                                                                * CRITICAL FLOOD LEVEL *
**********************  THIRTY FEET TO THIRTY-FIVE FEET                                 **********************
**********************                                                                  **********************
<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                           2 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   3

                               2 FLOOD PROTECTION SETS EXIST AT A FLOOD TO LEVEL   3

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

FLOOD PROTECTION SETS AT LEVEL   3

(    1)  VAL3ACLD   33
(    2)  PMP01FTO   30    PWS01OFF   30
```

Figure 12.12 NOAH Output Example: Flood Protection Set Results

```
        NOAH USERS MANUAL SAMPLE PROBLEM                                         VERSION - 2 - DEC, 1981

FLOOD PROTECTION SET ANALYSIS OF LEVEL   4
KEY IS '  40'              DESCRIPTION='THIRTY-FIVE FEET TO FORTY FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                      10 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   4

                           3 FLOOD PROTECTION SETS EXIST AT A FLOOD TO LEVEL   4

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

FLOOD PROTECTION SETS AT LEVEL   4

(    1)  VAL3ACLD   33    PWS03OFF   40
(    2)  PMP01FTO   30    PWS01OFF   30    VAL01CLD   38
(    3)  PMP01FTO   30    PWS01OFF   30    VAL02CLD   14    PWS02OFF   12    PMP02FTO   12


        NOAH USERS MANUAL SAMPLE PROBLEM                                         VERSION - 2 - DEC, 1981

FLOOD PROTECTION SET ANALYSIS OF LEVEL   6
KEY IS '  50'              DESCRIPTION='FORTY-FIVE FEET TO FIFTY FEET

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                      20 TOTAL MINIMAL CUT SETS ARE FLOODED BY A FLOOD TO LEVEL   6

                           3 FLOOD PROTECTION SETS EXIST AT A FLOOD TO LEVEL   6

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

FLOOD PROTECTION SETS AT LEVEL   6

(    1)  PMP01FTO   30    PWS01OFF   30    VAL01CLD   38
(    2)  VAL3ACLD   33    PWS03OFF   40    PMP03FTO   50    VAL3BCLD   50
(    3)  PMP01FTO   30    PWS01OFF   30    VAL02CLD   14    PWS02OFF   12    PMP02FTO   12
```

Figure 12.12 Continued

```
        NOAH USERS MANUAL SAMPLE PROBLEM                                   VERSION - 2 - DEC, 1981

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                              SEARCH FOR BASIC EVENT 'PMP01FTO'
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

LEVEL '  35'    PMP01FTO   30    VAL3ACLD   33

LEVEL '  40'    PMP01FTO   30    PWS03OFF   40

LEVEL '  50'    PMP01FTO   30    PMP03FTO   50

LEVEL '  50'    PMP01FTO   30    VAL3BCLD   50
        NOAH USERS MANUAL SAMPLE PROBLEM                                   VERSION - 2 - DEC, 1981

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                              SEARCH FOR COMPONENT 'PWS
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

LEVEL '  35'    PWS01OFF   30    VAL3ACLD   33

LEVEL '  40'    PMP01FTO   30    PWS03OFF   40

LEVEL '  40'    PWS01OFF   30    PWS03OFF   40

LEVEL '  40'    VAL01CLD   38    PWS03OFF   40    VAL02CLD   14

LEVEL '  40'    VAL01CLD   38    VAL3ACLD   33    PWS02OFF   12

LEVEL '  40'    VAL01CLD   38    PWS03OFF   40    PWS02OFF   12

LEVEL '  40'    VAL01CLD   38    PWS03OFF   40    PMP02FTO   12

LEVEL '  50'    PWS01OFF   30    VAL3BCLD   50

LEVEL '  50'    PWS01OFF   30    PMP03FTO   50

LEVEL '  50'    VAL01CLD   38    PMP03FTO   50    PWS02OFF   12

LEVEL '  50'    VAL01CLD   38    VAL3BCLD   50    PWS02OFF   12
   >>  LEAVING ROUTINE  <<
```

Figure 12.13 NOAH Output Example: Component SEARCH Results

```
              1          2
   30.0000   81.0000
          0
0.1000E-030.0        0.3000E-030.0
0.0
          2          2
   80.0000   81.0000
          0
0.1000E-020.0        0.1000E 000.0
0.0
          3          2
   50.0000   81.0000
          0
0.1000E-020.0        0.5000E 000.0
0.0
          4          2
  80.00000   81.0000
          0
0.1000E-030.0        0.3000E-030.0
0.0
          5          2
   30.0000   81.0000
          0
0.1000E-020.0        0.1000E 000.0
0.0
          6          2
   30.0000   81.0000
          0
0.1000E-020.0        0.5000E 000.0
0.0
          7          2
   30.0000   81.0000
          0
0.1000E-030.0        0.3000E-030.0
0.0
          8          2
    5.0000   81.0000
          0
0.1000E-030.0        0.3000E-030.0
0.0
          9          2
    5.0000   81.0000
          0
0.1000E-020.0        0.5000E 000.0
0.0
         10          2
    5.0000   81.0000
          0
0.1000E-020.0        0.1000E 000.0
0.0
```

Figure 12.14 NOAH Output Example: KITT-2 Data (This information is punched on cards if requested but is not printed with other results)

1			
20			
2	5	1	
2	6	1	
2	5	3	
2	6	3	
2	5	2	
2	5	4	
2	6	4	
2	6	2	
3	7	1	8
3	7	3	8
3	7	1	9
3	7	1	10
3	7	3	9
3	7	3	10
3	7	2	8
3	7	4	8
3	7	2	9
3	7	2	10
3	7	4	9
3	7	4	10

Figure 12.14 Continued

13. REFERENCES

1. Wagner, D. P., et al., Flood Risk Analysis Methodology Development Project Final Report, JBFA-110-81, JBF Associates, Inc., Knoxville, Tennessee, December 4, 1981.

2. Reactor Safety Study, WASH-1400 (NUREG-75/014), U. S. Nuclear Regulatory Commission, October 1975.

3. Lewis, H. W., et al., Risk Assessment Review Group Report to the U. S. Nuclear Regulatory Commission, NUREG/CR-0400, September 1978.

4. Vesely, W. E. and Narum, R. E., PREP and KITT: Computer Codes for the Automatic Evaluation of a Fault Tree, IN-1349, August 1970.

5. Knuth, D. E., The Art of Computer Programming, Vol. 1 - Fundamental Algorithms, 2nd ed., Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, February 1975.

6. Fussell, J. B., et al., MOCUS - A Computer Program to Obtain Minimal Sets from Fault Trees, ANCR-1156, August 1974.

7. Rasmuson, D. M. and Marshall, N. H., "FATRAM - A Core Efficient Cut-Set Algorithm," IEEE Transactions on Reliability, R-27, 4, October 1978.

APPENDIX A

PROGRAMMER'S GUIDE FOR THE ESP COMPUTER PROGRAM

A.1 INTRODUCTION

The ESP computer program screens large sets of accident sequences associated with a power plant to determine those sequences that are potentially significant contributors to risk in the event of a flood. ESP screens accident sequences within a consequence category by comparing their occurrence frequency in the event of a flood to a user specified fraction of the unflooded consequence category occurrence frequency. After completing the screening for a consequence category, ESP calculates an estimated consequence category occurrence frequency (including unflooded and flooded effects), lists elements of the significant accident sequences considered flood susceptible and ranks the screened accident sequences according to their relative contribution to the flooded occurrence frequency of the consequence category.

ESP is written in ANSI-66 FORTRAN for the IBM 360/370 computers using a FORTRAN H compiler.* Using a dynamic array allocation scheme (Section A.2.1), ESP can easily handle a problem of any size, providing there is sufficient computer memory available. Section A.2 describes in detail the subroutines that compose the ESP program.

*ANSI-77 FORTRAN was not used because a production compiler for large mainframes was not available.

A.2 ESP SUBROUTINE DESCRIPTIONS

This section describes the subroutines contained in the ESP computer program. Included in each description is a discussion of the subroutine function, a list of other subroutines that call or are called by the subroutine being described, and the names and purposes of the major variables in the subroutine. The section title provides the parameter list for the subroutine.

A.2.1 MAIN

Routine MAIN is the program flow controller of ESP. The purpose of MAIN is to invoke the proper subroutines for reading and processing the input data and outputting the results. MAIN calls subroutines GOOFUP, COUNT, ALOCAT, INPUT and DOIT. No subroutines call MAIN. Table A.1 lists important variables used by MAIN. The value of PLINES is set in the data statement to control the number of lines on a page of paper.

A.2.2 ALOCAT (MAX)

Subroutine ALOCAT is a dynamic array space allocation scheme. Most arrays used in ESP are stored in one large array W. ALOCAT determines the starting address of each smaller array stored in array W. These starting addresses are stored in the IW array, which occupies the first 49 spaces of the W array. Starting addresses for arrays stored in W are determined by adding the dimension of the last array stored in W to its starting address. Table A.2 lists the equations used by ALOCAT to determine the next starting address when the previous array stored in W contained smaller than doubleword values.

For example, given the following array, dimensions and storage values,

1	A1(NSYS)	R*8
2	A2(NSUS)	L*1
3	A3(NSUS)	I*2
4	A4(NSUS)	R*4
5	A5(NSUS)	I*4

and the fact that the first location in IW is at 50, the following code would be used to allocate the array space:

```
IW(1) = 50
IW(2) = IW(1) + NSYS*3
IW(3) = IW(2) + (NSUS + 7)/8
IW(4) = IW(3) + (NSUS + 3)/4
IW(5) = IW(4) + (NSUS + 1)/2
IW(6) = IW(5) + (NSUS + 1)/2
```

Table A.1 MAIN Variables

Variable	Word Type	Description
TITLE(10)	R*8	Problem title
NSYS	I*4	Number of systems input (returned from subroutine COUNT)
NSUS	I*4	Number of flood susceptible systems (returned from subroutine COUNT)
NCAT	I*4	Number of consequence categories to test (returned from subroutine NCOUNT)
NSEQ	I*4	Number of accident sequences in a consequence category (returned from subroutine COUNT)
W	R*8	ESP work array
IW	I*4	Work array starting addresses (see ALOCAT)
PLINES	I*4	Number of lines per page of paper (installation dependent)

Table A.2 Starting Addresses Calculated By ALOCAT

Array Dimension	Word Size	Dimension Value†
NDIM	Doubleword (R*8)	NDIM
NDIM	Fullword (I*4,R*4,L*4)	$\frac{NDIM + 1}{2}$
NDIM	Halfword (I*2)	$\frac{NDIM + 3}{4}$
NDIM	Byte (L*1)	$\frac{NDIM + 7}{8}$

†Addresses are determined using integer arithmetic.

IW(6) is the maximum space for the example. To pass array A3 as a subroutine, W(IW(3)) would appear in the parameter list. (See the calls to INPUT and DOIT in routine MAIN.) This method does not require array A3 to be in the calling routine before it can be passed.

MAIN is the only routine that calls subroutine ALOCAT and GOOFUP is the only subroutine called by ALOCAT. Tables A.3 and A.4 describe the important variables in this subroutine and the arrays stored in W, respectively.

A.2.3 COUNT (NUM, NLINE, PLINES, TITLE)

Subroutine COUNT determines the number of elements contained in each input group. COUNT also writes the input to logical unit 14. MAIN is the only program that calls COUNT and GOOFUP is the only subroutine called by COUNT. Table A.5 describes important variables used in this subroutine.

A.2.4 DOIT (CODE, FREQ, FLOOD, CAT, FCAT, RANK, FFREQ, CRITRA)

Subroutine DOIT performs all the quantitative calculations in ESP. DOIT calculates the unflooded and flooded occurrence frequency of the accident sequences and the consequence categories, the relative importance of each flood susceptible accident sequence and the total occurrence frequency of each consequence category. The unflooded occurrence frequency of an accident sequence is the product of the occurrence frequency/probability of failure on demand (user-supplied) of the elements composing the sequence. ESP calculates the flooded accident sequence occurrence frequency similarly; in this case, the probability of failure on demand of flood susceptible accident sequence elements (user-designated) is assumed to be one.

Subroutines called by DOIT are RANKIT and PRINT. MAIN is the only program that calls DOIT. Table A.6 describes important variables used in subroutine DOIT.

A.2.5 GOOFUP (IER, WORD1, NUM1, NUM2, CARD)

Subroutine GOOFUP prints error messages whenever errors occur in the input. The error messages printed include a reference number and a brief description of the error. Appendix B lists the error messages used by ESP, descriptions of the errors and suggested solutions.

Subroutines that call GOOFUP are MAIN, ALOCAT, COUNT and INPUT. No subroutines are called by GOOFUP. Table A.7 describes important variables used in subroutine GOOFUP.

Table A.3 ALOCAT Variables

Variable	Type	Description
MAX	I*4	Size of array W. If the space needed to store information in W exceeds MAX, an error message is printed and program execution terminates.
NSYS	I*4	Number of systems input (determined in subroutine COUNT)
NSUS	I*4	Number of flood susceptible systems (determined in subroutine COUNT)
NCAT	I*4	Number of consequence categories to test (determined in subroutine COUNT)
NSEQ	I*4	Number of accident sequences input (determined in subroutine COUNT)

Table A.4 Arrays Stored In W

Array	Array Name	Word Type	Description
IW(1)	NAME(NSYS, 3)	R*8	System names/descriptions
IW(2)	CODE(NSYS)	I*2	System code names
IW(3)	FREQ(NSYS)	R*4	Occurrence frequency/ failure on demand probability of each system
IW(4)	FLOOD(NSYS)	L*1	System susceptibility to floods
IW(5)	CAT(NCAT)	I*4	Consequence categories to test
IW(6)	FCAT(NCAT)	R*4	Consequence category occurrence frequencies
IW(7)	RANK(NCAT)	R*4	Flooded accident sequence frequency
IW(8)	IRANK(NSEQ)	I*2	Flooded accident sequence rank
IW(9)	SUS(NSYS)	I*2	System reference numbers

NSYS is the number of systems input.
NCAT is the number of consequence categories for test.
NSEQ is the number of accident sequences input.

Table A.5 COUNT Variables

Variables	Word Type	Description
CARD(10)	R*8	Input being read
NLINE	I*4	Current line number being printed out
NUM	I*4	Number of elements in the input group
TITLE(10)	R*8	Problem title
PLINES	I*4	Number of lines per page of paper

Table A.6 DOIT Variables

Variable	Word Type	Description
CODE (NSYS)	I*2	System code names
FREQ (NSYS)	R*4	Occurrence frequency/failure on demand probability of each system
FLOOD (NSYS)	L*1	System susceptibility to flood
CAT (NCAT)	I*4	Consequence categories to test
FCAT (NCAT)	R*4	Consequence category occurrence frequencies
RANK (NSEQ)	R*4	Relative rank of VALUE for an accident sequence with respect to other accident sequence VALUES in the same consequence category
FFREQ	R*4	Flood occurrence frequency
CRITRA	R*4	Criteria for accepting or rejecting a flooded accident sequence as significant
FREQA	R*4	Unflooded accident sequence occurrence frequency
FREQB	R*4	Product of the occurrence frequency failure on demand probability of all non-flood susceptible elements in a flood susceptible accident sequence
FREQC	R*4	Unflooded occurrence frequency of an accident sequence containing flood susceptible elements
SUMA	R*4	Unflooded occurrence frequency of a consequence category $SUMA = \sum_{i=1}^{m} FREQA_i$, where m is the number of accident sequences in the consequence category

Table A.6 DOIT Variables (continued)

Variable	Word Type	Description
SUMB	R*4	$SUMB = \sum_{i=1}^{n} FREQB_i$, where n is the number of flood susceptible accident sequences in a consequence category
SUMC	R*4	$SUMC = \sum_{i=1}^{n} FREQC_i$, where n is the number of flood susceptible accident sequences in a consequence category
IICAT	I*4	Category under analysis
NLOA	I*4	Number of elements in the accident sequence currently being analyzed
SEQN(21)	I*2	Reference number for the accident sequence currently being analyzed
SEQNJ	I*2	Code name of element currently being considered in the accident sequence currently being analyzed
NRANK	I*4	Number of accident sequences in a consequence category that pass the screening criteria
TOTAL	R*4	Total consequence category occurrence frequency, including both unflooded and flooded effects. TOTAL = SUMA + FFREQ*(SUMB - SUMC)
LUN1 } LUN2	I*4	Disk units which store the accident sequences. LUN1 stores the accident sequences belonging to the consequence category currently analyzed. LUN2 stores all other accident sequences.

Table A.6 DOIT Variables (continued)

Variable	Word Type	Description
VALUE	R*4	VALUE is the ratio of the flooded accident sequence occurrence frequency to its unflooded consequence category occurrence frequency. VALUE is used to screen and rank accident sequences.

NSYS is the number of systems input.
NCAT is the number of consequence categories to test.
NSEQ is the number of accident sequences input.

Table A.7 GOOFUP Variables

Variable	Word Type	Description
IER	I*4	Error reference number
WORD1	R*8	Eight byte dummy variable printed in error message
NUM1	I*4	Four byte dummy variable printed in error message
NUM2	I*4	Four byte dummy variable printed in error message
CARD(10)	R*8	Input card in error printed in error message
NERR	I*4	Counter for errors

A.2.6 INPUT (NAME, CODE, FREQ, FLOOD, CAT, FCAT, FFREQ, CRITRA)

Subroutine INPUT reads the input data and stores it in the proper arrays. INPUT also checks the input data for inconsistencies between data groups. Errors detected in the input are described by subroutine GOOFUP. Program execution halts upon completion of input if any errors are discovered.

INPUT calls only one subroutine, GOOFUP. MAIN is the only routine that calls INPUT. Table A.8 lists important variables used in subroutine INPUT.

A.2.7 PRINT (ICAT, ISEQ, CODE, FLOOD, SUS, NAME, FCAT, TOTAL, FCONT, IRANK, FIX)

Subroutine PRINT prints the results for each consequence category. Subroutine DOIT calls PRINT. PRINT does not call any subroutines. Table A.9 lists important variables used in PRINT.

A.2.8 RANKIT (RANK, IRANK, NRANK)

Subroutine RANKIT orders the screened accident sequences from the largest to the smallest contributor to the total consequence category occurrence frequency. Accident sequences that do not pass the screening criteria are not ranked. Subroutine DOIT calls RANK. Subroutine RANK does not call any other subroutines. Table A.10 lists important variables used in RANK.

Table A.8 INPUT Variables

Variable	Word Type	Description
NAME(NSYS, 3)	R*8	System names/descriptions
CODE(NSYS)	I*2	System code names
FREQ(NSYS)	R*4	Occurrence frequency/failure on demand probability of each system
FLOOD(NSYS)	I*1	System susceptibility to floods
CAT(NCAT)	I*4	Consequence categories to test
FCAT(NCAT)	R*4	Consequence category occurrence frequencies
FFREQ	R*4	Flood occurrence frequency
CRITRA	R*4	Criteria for accepting or rejecting a flooded accident sequence as significant
ICAT	I*4	Consequence category on accident sequence
NLOA	I*4	Number of elements the accident sequence contains
SEQN(21)	I*2	Accident sequence
SEQNJ	I*2	Member of accident sequence
OK	L*4	Flag to indicate if any errors found in accident sequence
STOP	R*8	End of a complete set of data for an ESP run
SYSTEM	R*8	Title of Input Group 1 (* SYSTEM)
FLUD	R*8	Title of Input Group 2 (* FLOOD)

Table A.8 INPUT Variables (continued)

Variable	Word Type	Description
CATGRY	R*8	Title of Input Group 3 (* CATEGORY)
SEQNCE	R*8	Title of Input Group 4 (* SEQUEN)

NSYS and NCAT are the number of systems input and the number of consequence categories to test, respectively.

Table A.9 PRINT Variables

Variable	Word Type	Description
ICAT	I*4	Consequence category being analyzed
ISEQ	I*4	Number of accident sequences that pass CRITRA
CODE(NSYS)	I*2	System code names
FLOOD(NSYS)	L*1	System susceptibility to floods
SUS(NSYS)	I*2	Temporary array containing the index of each system in the accident sequence in question
NAME(NSYS,3)	R*8	System names/descriptions
FCAT	R*4	Unflooded consequence category occurrence frequency
TOTAL	R*4	Total consequence category occurrence frequency, including unflooded and flooded effects
FCONT	R*4	Flood contribution to the category occurrence frequency
IRANK	I*2	Relative importance contribution rank for each accident sequence
FIX	L*4	Indicates whether or not an upper bound is used in calculating FCONT
FREQU	R*4	Accident sequence unflooded occurrence frequency
FREQF	R*4	Flood contribution to the accident sequence flooded occurrence frequency
NLOA	I*4	Number of members in accident sequence
SEQN(21)	I*2	Accident sequence
ICODE	I*2	Member of accident sequence
IHERE	I*4	Number of flood susceptible systems for this category

NSYS is the number of systems input.

Table A.10 RANK Variables

Variable	Word Type	Description
RANK(NSEQ)	R*4	Value to rank = flooded accident sequence occurrence frequency ———————————— unflooded consequence category occurrence frequency
IRANK(NSEQ)	I*2	Relative rank of RANK
NRANK	I*4	Number of values to rank
EPSLON	R*4	Used in ranking
DELTA	R*4	Used in ranking

NSEQ is the number of accident sequences in a consequence category.

APPENDIX B

ESP ERROR MESSAGES

B. ESP ERROR MESSAGES

This appendix lists the error messages written by the ESP program when errors are detected in the input data. The error number and message are stated and reasons for the error message described. Words shown in quotes are variables whose value would be printed in that location.

ERROR NUMBER 1: MISSING OR INVALID STOP/END CARDS

An END card for an input group or a STOP card for a complete data set has been omitted or mispunched. An END card must be supplied after each input group.

ERROR NUMBER 2: INVALID CONTROL CARD >> "CARD"

A card with an "*" in column 1 was encountered after an END card, but was not recognized as a correct control card for a data group.

ERROR NUMBER 3: INSUFFICIENT STORAGE. YOU HAVE "NUM1" WORDS AND YOU NEED "NUM2".

The work array is too small for the amount of data to be processed. Increase the value of variable MAX and the dimension of array W in the MAIN routine to at least NUM2.

ERROR NUMBER 4: MISSING OR INVALID INPUT GROUP "WORD1"

The WORD1 input was not recognized as being part of the input deck. This error may be accompanied by error #2, indicating a spacing problem or missing control card.

ERROR NUMBER 5: "WORD1" WAS INDICATED AS SUSCEPTIBLE TO FLOODS, BUT IT WAS NOT INCLUDED IN * SYSTEM

A flood-susceptible accident sequence element listed in * FLOOD was not included in the * SYSTEM data. Check input data for correct spelling and completeness of input data.

ERROR NUMBER 6: INVALID SYSTEM NAME "NUM2" ON THIS CARD >> "CARD"

NUM2 was identified in an accident sequence on the card following the pointer >>, but it was not described in * SYSTEM. Check for correct spelling and completeness of the input data.

ERROR NUMBER 7: BLANK SYSTEM NAME ON THIS CARD >> "CARD"

A blank was encountered in the string of accident sequence elements listed on the input card following the pointer >>. Check for the correct number of sequence elements in the accident sequence and for format errors.

APPENDIX C

ESP JOB CONTROL LANGUAGE

C. ESP JOB CONTROL LANGUAGE

This appendix is a listing of the Job Contol Language requirements for executing the ESP computer program.

```
//Jobcard

//EXEC FORTHCLG,FORTREG=320K,GOREG=320K

//FORT.SYSIN DD *

    [FORTRAN source]

//GO.FT05F001 DD DDNAME=SYSIN

//GO.FT06F001 DD SYSOUT=A

//GO.FT14F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=80,BLKSIZE=4240,BUFNO=1)

//GO.FT15F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=80,BLKSIZE=4240,BUFNO=1)

//GO.FT16F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=80,BLKSIZE=4240,BUFNO=1)

//GO.FT17F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=100,BLKSIZE=5300,BUFNO=1)

//GO.SYSIN DD *
```

APPENDIX D

ESP INPUT SUMMARY

D. ESP INPUT SUMMARY

This appendix is a quick reference guide to the input for the ESP program.

Table D.1 ESP Input Deck Layout

- TITLE CARD
  [COMMENT CARDS]

- * SYSTEM
  [INITIATING EVENT/BRANCHING OPERATOR NAMES, CODES AND OCCURRENCE FREQUENCY/FAILURE ON DEMAND PROBABILITIES]
  END
  [COMMENT CARDS]

- * FLOOD
  [FLOOD FREQUENCY, SCREENING CRITERIA AND FLOOD SUSCEPTIBLE ACCIDENT SEQUENCE ELEMENTS]
  END
  [COMMENT CARDS]

- * CATEGORY
  [CONSEQUENCE CATEGORY DESCRIPTIONS AND UNFLOODED OCCURRENCE FREQUENCY]
  END
  [COMMENT CARDS]

- * SEQUENCE
  [ACCIDENT SEQUENCE DESCRIPTIONS]
  END
  [COMMENT CARDS]

- STOP CARD

Table D.2 Input Group 1, SYSTEM

Format: 3A8, 5X, A2, 3X, E10.6

One card per initiating event/branching operator.

Variable Name	Format	Card Columns	Description
NAME	3A8	1-24	Accident sequence initiating event/branching operator description
	5X	25-29	Blank
CODE	A2	30-31	Accident sequence initiating event/branching operator code name
	3X	32-34	Blank
FREQ	E10.6	35-44	Initiating event/branching operator occurrence frequency or failure on demand probability

Table D.3 Input Group 2, FLOOD

Format: 2F10.0 First card.

A2 Additional cards, one per susceptible element.

Variable Name	Format	Card Columns	Description
FFREQ	F10.0	1-10	Flood occurrence frequency
CRITRA	F10.0	11-20	Accident sequence screening criteria
FCODE	A2	1-2	Code name of flood susceptible accident sequence elements

Table D.4 Input Group 3, CATEGORY

Format: I10,F10.0

One card per consequence category analyzed.

Variable Name	Format	Card Columns	Description
CAT	I10	1-10	Consequence category identification number
FCAT	F10.0	11-20	Consequence category unflooded occurrence frequency

Table D.5 Input Group 4, SEQUENCE

Format: 2I5, 5X, 21(A2,1X)

One card per accident sequence.

Variable Name	Format	Card Columns	Description
ICAT	I5	1-5	Consequence category the accident sequence results
NLOA	I5	6-10	Number of elements in the accident sequence
	5X	11-15	Blank
SEQN(1)	A2	16-17	Code name of first accident sequence element
	1X	18	Blank
SEQN(2)	A2	19-20	Code name of second element
	1X	21	Blank
•		•	•
•		•	•
SEQN(NLOA)		79-80	Code name of last element in sequence

APPENDIX E

PROGRAMMER'S GUIDE FOR THE

NOAH COMPUTER PROGRAM

E.1 INTRODUCTION

The NOAH computer program identifies flooded system minimal cut sets as the flood level increases. A flooded minimal cut set is a minimal cut set that has all its components submerged by the flood. In addition, NOAH is capable of identifying flood protection sets (minimal collections of components that, if they are all protected against flood-caused failure, prevent flood-caused system failure) and partially flooded minimal cut sets (all but one or two components are submerged).

To determine the flooded minimal cut sets for a particular flood level, NOAH develops a pseudo-binary image[5] of the input fault tree that contains only submerged components. NOAH tests this pseudo-binary image to determine if any minimal cut sets exist. If the test is true, NOAH determines the minimal cut sets using a modified FATRAM algorithm.[7] Otherwise, it raises the flood to the next level and develops a new pseudo-binary fault tree image for testing. After minimal cut sets are determined for a flood level, NOAH uses a cut set - path set cancellation routine in combination with FATRAM to determine minimal cut sets.

NOAH is written in ANSI-66 FORTAN for the IBM 360/370 computers*. Using a dynamic array allocation scheme (Section E.2.2), NOAH can easily handle a problem of any size providing there is sufficient computer memory available. Section E.2 describes in detail the subroutines that compose the NOAH program.

---

*ANSI-77 FORTRAN was not used because a production compiler for large mainframes was not available.

E.2 NOAH SUBROUTINE DESCRIPTIONS

This section describes the subroutines contained in the NOAH computer program. Included in each description is a discussion of the subroutine's function, the subroutine parameter statement, a list of subroutines that call or are called by the subroutine being described and the names and purposes of the major variables in the subroutine. Flowcharts are also provided for the larger, more complex subroutines. The section title provides the parameter list for the subroutine.

E.2.1 MAIN

Routine MAIN is the program flow controller of NOAH. The purpose of MAIN is to invoke the proper algorithms for reading and processing the input data and outputting the results. MAIN calls subroutines GOOFUP, GETNG, ALOCAT, INPUT, BUILD, ALOCT2, and DOIT. No subroutines call MAIN.

Table E.1 describes important variables used in MAIN. The major variables in MAIN are input parameters, which are read in using NAMELIST. NAMELIST permits input and output of variables and arrays with an identifying name instead of a format specification. (If namelist is not supported at the user's facility then the variables can be input using "standard" READ statements.) While NAMELIST is not ANSI standard, it is supported on IBM, CDC, DEC and UNIVAC machines.

The NAMELIST option includes two types of statements: (1) the NAMELIST statement and (2) associated READ/WRITE statements.

The NAMELIST statement has the following format:

```
NAMELIST /GNAM/ VBL1, VBL2, VBL3,...VBLN
```

where GNAM is the NAMELIST group name and VBL are the associated variables.

The NAMELIST name identifies which group to use in any associated I/O statement (i.e. READ (5,GNAM)). A variable or array name may belong to one or more NAMELIST groups. Data read by a single NAMELIST name READ statement must contain only names referenced in that NAMELIST statement. However, they do not all need to be included so that defaults may be taken. The variables on the data card do not need to be in the particular order specified on the NAMELIST statement. If more than one NAMELIST group appears in the input data, the groups must appear in the order and at the location specified by the READ statements. If not, an end-of-file will result, as the program will read data until it finds the specified card.

The format for the NAMELIST input card(s) is:

Table E.1 MAIN Variables

Variable	Word Type	Description
ORDER	I*4	Maximum size of minimal cut sets to find
DEPTH	I*4	Number of levels to flood
NDEP	I*4	Number of levels used in describing a plant
MAXD	I*4	Maximum level to flood
DOPT	L*4	Option for method of determining plant flood levels (user supplied vs. NOAH calculated levels)
FIND	L*4	Defines whether or not partial common cause candidates are found given MAXD is reached and the TOP event has not occurred
SEEPTH	L*4	Defines whether or not flood protection sets are printed for each flood level
MAXIN	I*4	Maximum number of inputs to any gate
DEEPER	L*4	Defines the type of flood analysis performed (analyze only on individual level at each step vs. analyzing levels 0 through i at each step)
DSRCH	L*4	Defines whether or not to identify minimal cut sets from the final minimal cut set list that contain a specified basic event or component
TIMPT	I*4	Defines the type of flood profile input (if any) and whether or not to punch KITT-2 data
DUMP	L*4	Defines whether or not intermediate results are to be printed

Table E.1 MAIN Variables (continued)

Variable	Word Type	Description
TRACE	L*4	Defines whether or not to print a program trace during program execution
ECHO	L*4	Defines whether or not to print program arrays as they are filled
NDIM	I*4	Sum of NCOMP and NGATE. NDIM is used as an array dimension
XYZ	I*4	A dummy variable used to preserve word boundaries
W(10000)	R*8	Array containing all in-core storage space used by NOAH
PLINES	I*4	Number of lines per page of paper (installation dependent)

```
b$GNAM VBL = value $END
```

A "b" represents a blank space. Column one is always blank on the NAMELIST card. An example NAMELIST input group is:

```
b$XAMPLE       INTGER = 7, LOGCAL = T,
               REAL1 = 4.3, REAL2 = 4.735E-10,
               ARRAY1 = 4, 7, 9, 11,
               ARRAY2(4,3) = 78   $END
```

If an array name with no subscripts is given, the number of values given must not exceed the dimensions. The NAMELIST data may span more than one card.

Section E.2.3 BLOCK DATA describes all the common blocks in MAIN except WORK. Common block WORK contains two scalars, NDIM and XYZ, and the storage array W (Section E.2.2 ALOCAT). The size of array W is assigned in an associated DATA statement in MAIN by the variable MAX.

E.2.2 ALOCAT (MAX)

Subroutine ALOCAT is a dynamic array space allocation program. Most subscripted variables (and their associated storage arrays) used in NOAH are stored in one large array W. ALOCAT determines the starting address of each smaller array stored in W. These starting addresses are stored in the IW array, which occupies the first 30 spaces of the W array. The dimension of the previous array stored in W, plus its starting address, determines the starting address of the next array to be stored in W. Table E.2 lists the equations used by ALOCAT to determine a new starting address when the previous array stored in W contained smaller than double word subscripted variables.

For example, given the following arrays, dimensions and storage values:

1	A1(NDIM,3)	I*2
2	A2(NBE)	L*1
3	A3(NBE)	R*8
4	A4(NBE)	R*4
5	A5(NBE)	I*4

and the fact that the first location in IW is at 50, the following code would be used to allocate the array space:

```
IW(1) = 50
IW(2) = IW(1) + (NDIM*3 + 3)/4
IW(3) = IW(2) + (NBE + 7)/8
IW(4) = IW(3) + NBE
IW(5) = IW(4) + (NBE + 1)/2
IW(6) = IW(5) + (NBE + 1)/2
```

Table E.2 Starting Addresses Calculated By ALOCAT

Array Dimension	Word Size	Dimension Value*
NDIM	Doubleword (R*8)	NDIM
NDIM	Fullword (R*4,I*4,L*4)	$\frac{NDIM + 1}{2}$
NDIM	Halfword (I*2)	$\frac{NDIM + 3}{4}$
NDIM	Byte (L*1)	$\frac{NDIM + 7}{8}$

*Addresses are determined using integer arithmetic.

IW(6) is the maximum space for this example. To pass array A3 to a subroutine, W(IW(3)) would appear in the parameter list. (See the calls to INPUT, BUILD and DOIT in routine MAIN.) This method does not require array A3 to be in the calling routine before it can be passed.

ALOCAT is divided into three major sections; ALOCAT, ALOCT2 and ALOCT3. The ALOCAT section alocates array space for input data and for work arrays used throughout the execution of NOAH. ALOCT2 overwrites the input array space with arrays used in finding minimal cut sets. ALOCT3 overwrites the input array space with arrays used in finding partially flooded minimal cut sets. ESP invokes ALOCT3 only if MAXD is reached before the critical flood depth and FIND = T (true).

MAIN and DOIT are the only programs that call subroutine ALOCAT. ALOCAT calls subroutines LAYOUT and GOOFUP. Tables E.3 and E.4 describe the important variables used in this subroutine and the arrays stored in W, respectively.

E.2.3 BLOCK DATA

Routine BLOCK DATA initializes all the common blocks used in the NOAH subroutines except for WORK (Section E.2.1). The common blocks initialized by BLOCK DATA are ERR, LEVL, OPT, OPT1, OPT2, OPT3, PARM1, PARM3, PRINT, and VRSN. Table E.5 describes important variables in the common blocks initialized by BLOCK DATA.

E.2.4 BUILD (TREE, TREEX, TRENDX, IGATYP, NAM, LEVATN, NREP, HOUSE, FLOOD, LAMBDA, TAU, NGI, NCI, IGTYP, GATE, ELVATN, INDEX, IHOUSE, IFLOOD, ILAMDA, ITAU, GNUM)

Subroutine BUILD generates a threaded pseudo-binary image of the fault tree input to NOAH (Figure E.1). NOAH manipulates this image of the fault tree in determing the flooded minimal cut sets.

Figure E.2 is a flowchart of BUILD. The development of the pseudo-binary image begins by reading the inputs to the TOP event into a circular queue. Each entry in the queue, beginning with the first, is then analyzed. If the item is a gate, BUILD adds the gate's inputs to the bottom of the queue and adds a node representing this gate to the tree. This node contains the following four items of information about the gate (Figure E.3):

1. the gate or basic event index number,

2. a pointer to the first element input to this gate (its son),

3. a pointer from this element to the next element input to this gate's father (its brother), and

Table E.3 ALOCAT Variables

Variable	Word Type	Description
MAX	I*4	Size of array W. If the space needed to store information in W exceeds MAX, an error message is printed and program execution terminates.
ORDER	I*4	Maximum size of minimal cut sets to find
DEPTH	I*4	Number of levels to flood
NDEP	I*4	Number of levels used in describing a plant
MAXD	I*4	Maximum level to flood
DOPT	L*4	Option for method of determing plant flood levels (user supplied vs. NOAT calculates levels)
FIND	L*4	Defines whether or not partial common cause candidates are found given MAXD is reached and the TOP event has not occurred
SEEPTH	L*4	Defines whether or not flood protection sets are printed for each flood level
MAXIN	I*4	Maximum number of inputs to any gate
DEEPER	L*4	Defines the type of flood anlysis performed (analyze only on individual level at each step vs. analyzing levels o through i at each step)
DSRCH	L*4	Defines whether or not to identify minimal cut sets from the final minimal cut set list that contain a specified basic event or component
TIMPT	I*4	Defines the type of flood profile input (if any) and whether or not to punch KITT-2 data
NBE	I*4	Number of unique basic events in the fault tree
NG	I*4	Number of unique gates in the fault tree

Table E.3 ALOCAT Variables (continued)

Variable	Word Type	Description
NGATE	I*4	Total number of gates in the fault tree
NCOMP	I*4	Total number of basic events in the fault tree
MAXREP	I*4	Maximum number of times a basic event is repeated in the fault tree
CROW	I*4	Estimated number of Boolean Indicated Cut Sets (BICS)
PROW	I*4	Estimated number of Boolean Indicated Path Sets (BIPS)
ROWMAX	I*4	Maximum number of rows in the in-core cut set/path set array
CCOL	I*4	Estimated order of the BICS
PCOL	I*4	Estimated order of the BIPS
DUMP	L*4	Defines whether or not intermediate results are to be printed
TRACE	L*4	Defines whether or not to print a program trace during program execution
ECHO	L*4	Defines whether or not to print program arrays as they are filled
NDIM	I*4	Sum of NCOMP and NGATE. NDIM is used as an array dimension.
XYZ	I*4	A dummy variable used to preserve word boundaries
IW(50)	I*4	Array containing all in-core storage space used by NOAH

Table E.4 Arrays Stored In W

Array	Array Name	Word Type	Description
The following array space is filled initially by ALOCAT:			
IW(1)	TREE(NDIM,4)	I*2	Threaded pseudo-binary image of the fault tree
IW(2)	TRESAV(NDIM,3)	I*2	Pseudo-binary tree image storage area
IW(3)	TRESV2(NDIM,6)	I*2	Pseudo-binary tree image storage area
IW(4)	TREEL(NDIM)	L*1	Pseudo-binary tree image state array
IW(5)	STREEL(NDIM)	L*1	Pseudo-binary tree image state work area
IW(6)	LTREE(NDIM,2)	L*1	Pseudo-binary tree image state work area
IW(7)	TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic events in the fault tree
IW(8)	TRENDX (NGATE+NCOMP)	I*2	Location in TREE where each member first appears
IW(9)	IGATYP(NG)	I*2	Type of gate (0=OR, 1=AND)
IW(10)	NAM(NBE+NG)	R*8	Gate and basic event names
IW(11)	LEVATN(NBE)	I*4	Basic event elevations
IW(12)	KEYDSC(NDEP,10)	R*8	Flood level increment descriptions
IW(13)	KEY(NDEP)	I*4	Flood level keywords
IW(14)	NREP(NBE)	I*2	Number of times a basic event is repeated in the tree
IW(15)	HOUSE(NBE)	I*2	House event identifier
IW(16)	FLOOD(NBE)	I*2	Change of state identifier for house events
IW(17)	PNAM(NBE)	R*8	Array for printing basic event names

Table E.4 Arrays Stored In W (continued)

Array	Array Name	Word Type	Description
IW(18)	PEVTN(NBE)	I*4	Array for printing basic event elevations
IW(19)	NPL(NDEP)	I*2	Number of minimal cut sets found at each level
IW(20)	ONES(NBE)	[illegible]	One-event minimal cut sets
IW(21)	LAMBDA(NBE,2)	R*4	Basic event unflooded and flooded failure rates
IW(22)	TAU(NBE,2)	R*4	Basic event unflooded and flood mean downtimes
IW(23)	TIME(NDEP)	R*4	Time points used in describing a discrete flood profile
IW(24)	TDEEP(NTPT)	I*2	Elevations used with time points
The following arrays temporarily hold input data. ALOCT2 and ALOCT3 write over these arrays.			
IW(25)	NGI(NG)	I*2	Number of gates input to gate
IW(26)	NCI(NG)	I*2	Number of basic events input to a gate
IW(27)	IGTYP(NG)	I*2	Gate type
IW(28)	GATE(NG,MAXIN)	R*8	Fault tree gate array
IW(29)	ELVATN(NG,MAXIN)	I*4	Basic event elevation array
IW(30)	INDEX(NG,MAXIN)	I*2	Gate and basic event indices for constructing the pseudo-binary tree image
IW(31)	IHOUSE(NG,MAXIN)	I*2	House event array
IW(32)	IFLOOD(NG,MAXIN	I*2	House event state change array
IW(33)	LAMBDA(NG,MAXIN,2)	R*4	Basic event unflooded and flooded failure rates

Table E.4 Arrays Stored in W (continued)

Array	Array Name	Word Type	Description
IW(34)	ITAU(NG,MAXIN,2)	R*4	Basic event unflooded and flooded mean down times
IW(35)	IGNDX(NG)	I*2	Work space
IW(36)	GNUM(NG*NBE)	I*2	Work space

During program execution, ALOCT2 writes over several of the input arrays after they are no longer needed for input data. These arrays become the following:

Array	Array Name	Word Type	Description
IW(25)	IMIC(CROW)	I*2	Number of basic events in each minimal cut set
IW(26)	MICS(CROW,CCOL)	I*2	Minimal cut sets
IW(27)	IPATH(PROW)	I*2	Number of basic events in each flood protection set
IW(28)	IIPATH(PROW,PCOL)	I*2	Flood protection sets
IW(29)	IPTH(PROW)	I*2	Number of basic events in temporary intermediate flood protection sets
IW(30)	IIPTH(PROW,PCOL)	I*2	Temporary intermediate flood protection sets synthesized from only level 1 minimal cut sets
IW(31)	IWORK(PROW)	I*2	Number of basic events in the flood protection set work area
IW(32)	IIWORK(PROW,PCOL)	I*2	Flood protection set work area
IW(33)	ISGATE(CROW)	I*2	Location of the first gate in the cut set array

Table E.4 Arrays Stored in W (continued)

Array	Array Name	Word Type	Description
During program execution, if MAXD is reached before the critical flood level and the user specifies partial common cause candidates are to be identified, ALOCT3 writes over the input arrays. These arrays become the following:			
IW(25)	IMIC(CROW)	I*2	Number of basic events in each minimal cut set
IW(26)	IIMIC(CROW,CCOL)	I*2	Minimal cut sets
IW(27)	ISGATE(CROW)	I*2	First gate in the cut set array
IW(28)			Spare array space
IW(29)	ITWO(NBE,2)	I*2	Two-event unflooded sets that cause the TOP to occur with flooded events on
IW(30)			Spare array space
IW(31)	IPATH(PROW)	I*2	Number of basic events per flood protection set
IW(32)	IIPATH(PROW,PCOL)	I*2	Flood protection sets
IW(33)	IWORK(PROW)	I*2	Number of basic events per flood protection set in the work space
IW(34)	IIWORK(PROW,PCOL)	I*2	Flood protection set work area

For definitions of dimension variables see Table E.5.

Table E.5 BLOCK DATA Common Block Variables

Common Block	Variable	Word Type	Initial Value	Description
ALL	NCUT	I*4	0	Number of cut sets
ERR	ERROR	L*4	F	Flag for the occurrence of any errors in the input data
LEVL	ILEV	I*4	0	Reference number of the flood level currently under analysis
LEVL	THSLOC	I*4	0	Elevation of flood level currently under analysis
LEVL	LSTLOC	I*4	0	Elevation of the last flood level previously analyzed
OPT1	NDEP	I*4	0	Number of levels used in describing a plant
OPT1	DEPTH	I*4	0	Number of levels to flood
OPT1	MAXD	I*4	NEP	Maximum level to flood
OPT1	DEEPER	L*4	T	Defines the type of flood aalysis performed (analyze only on individual level at each step vs. analyzing levels o → i at each step)
OPT1	DOPT	L*4	T	Option for method of determining plant flood levels (user-supplied vs. NOAH calculated levels)
OPT2	TIMPT	I*4	0	Defines the type of flood profile input, and whether or not to punch KITT-2 data
OPT2	NTPT	I*4	0	Number of time points
OPT2	DSRCH	L*4	F	Defines whether or not to identify minimal cut sets from the final cut set list that contain a specified basic event
OPT2	CFD	L*4	F	Defines whether or not to determine flooded minimal cut sets
OPT3	ORDER	I*4	0	Maximum size of cut sets to find

Table E.5 BLOCK DATA Common Block Variables (continued)

Common Block	Variable	Word Type	Initial Value	Description
OPT3	FIND	L*4	T	Defines whether or not partial common cause candidates are found given MAXD is reached and the TOP event has not occurred
OPT3	SEEPTH	L*4	F	Defines whether or not flood protection sets are printed for each flood level
OPT3	MAXIN	I*4	7	Maximum number of inputs for any gate
PARM1	NBE	I*4	0	Number of unique basic events in the fault tree
PARM1	NG	I*4	0	Number of unique gates in the fault tree
PARM1	NGATE	I*4	0	Total number of gates in the fault tree
PARM1	NCOMP	I*4	0	Total number of basic events in the fault tree
PARM1	MAXREP	I*4	0	Maximum number of times any basic event appears in the fault tree
PARM3	CCOL	I*4	0	Order of the longest possible Boolean Indicated Cut Sets (BICS)
PARM3	CROW	I*4	0	Estimated number of BICS
PARM3	ROWMAX	I*4	500	Maximum number of rows in the in-core cut set/path set array
PARM3	PCOL	I*4	0	Order of the longest possible Boolean Indicated Path Sets (BIPS)
PARM3	PROW	I*4	0	Estimated number of BIPS
PRNT	DUMP	L*4	F	Defines whether or not intermediate results are to be printed
PRNT	TRACE	L*4	F	Defines whether or not to print a program trace during program execution

Table E.5 BLOCK DATA Common Block Variables (continued)

Common Block	Variable	Word Type	Initial Value	Description
PRNT	ECHO	L*4	T	Defines whether or not to print program arrays as they are filled
TIME	SLOPE	R*4	.0	Slope of linear flood profile
TIME	NTRCPT	R*4	.0	Intercept
VRSN	VERSN	R*8	'Version-2' 'Dec., 1981'	Program version
VRSN	TITLE	R*8	Bank	Job title

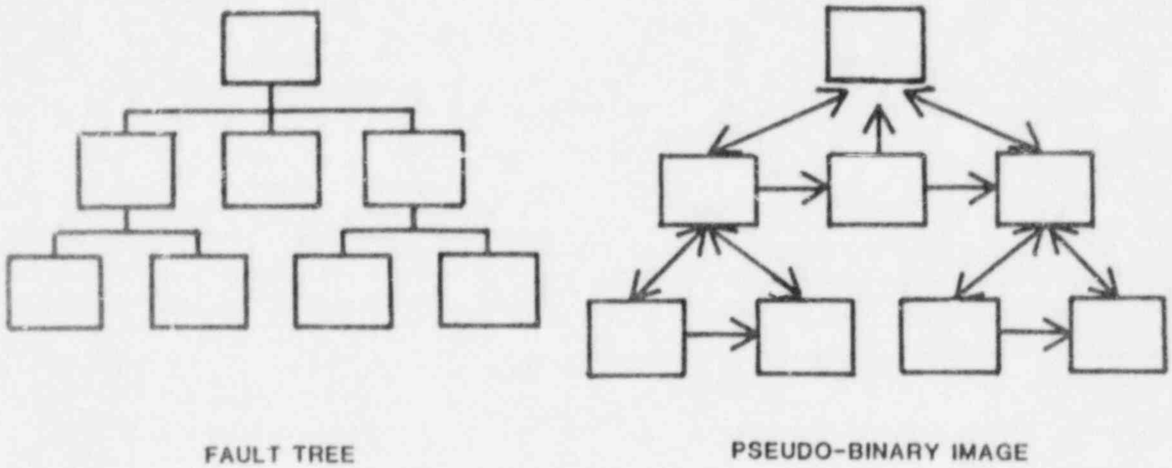

Figure E.1 Fault Tree Psuedo-binary Image Generated by BUILD

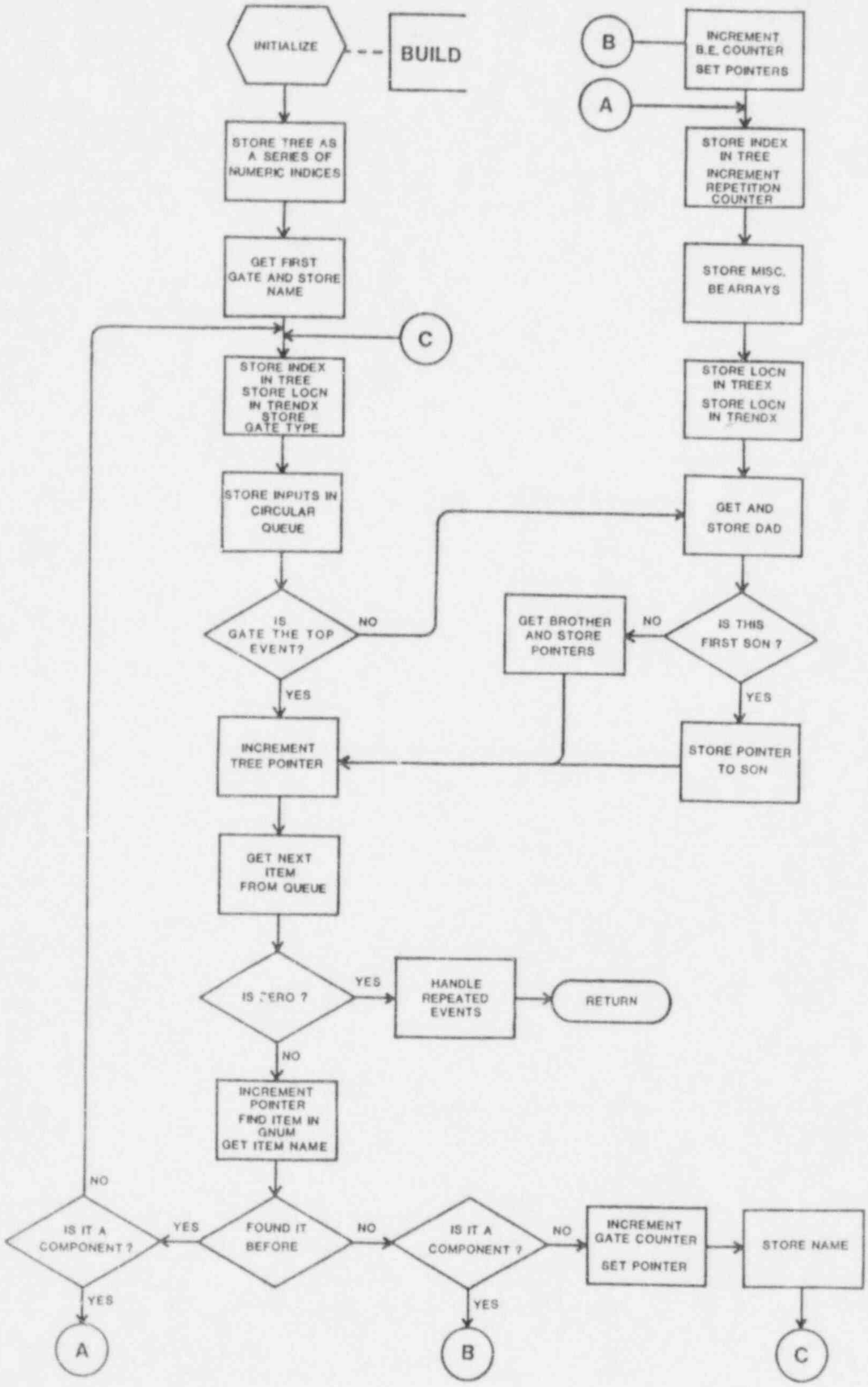

Figure E.2 Subroutine BUILD Flowchart

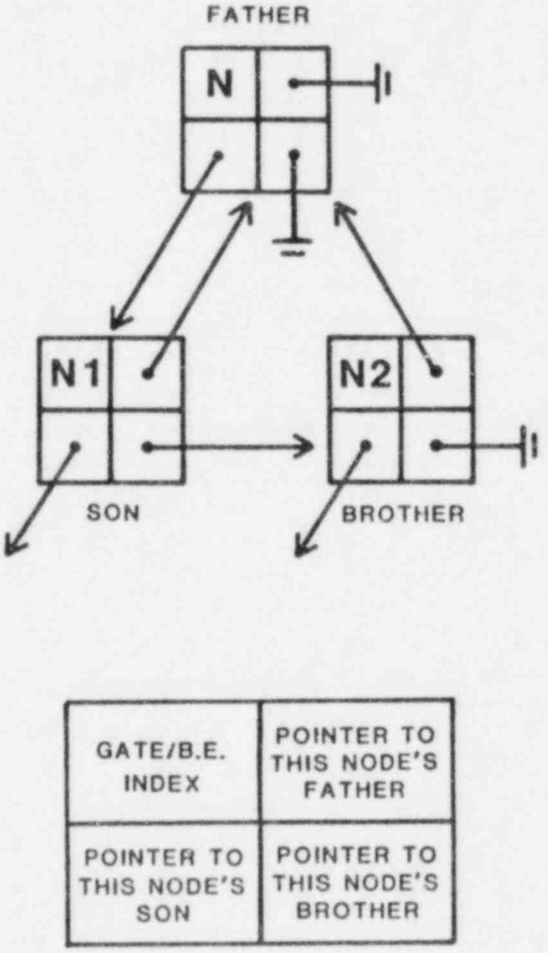

Figure E.3 Psuedo-binary Image Node of a Fault Tree Gate

4. a pointer from this element to the gate to which it is input (its father).

If the next item in the queue is a basic event, BUILD adds a node to TREE for the basic event and stores descriptive information about the basic event in the appropriate arrays. Basic event nodes contain the same information as gate nodes, except the "son" pointer is null. Upon completing the pseudo-binary image, BUILD checks to see that all repeated basic events have been included and their pointers are correctly set.

Subroutine BUILD calls the following subroutines: XREFI, GOOFUP, SEARCH, and XREFN. MAIN is the only routine that calls BUILD. Table E.6 describes important variables used in this subroutine.

E.2.5 CHEAT (COMPNT, BENAME, *, *)

Subroutine CHEAT determines if a string supplied by the user in the * SEARCH input data is contained within any basic event names. CHEAT checks the basic event names one at a time. After checking a basic event name, CHEAT returns to the calling subroutine via RETURN 1 or RETURN 2. If RETURN 1 is used, the string is contained within the given basic event name and the calling subroutine is signaled to continue execution of the search. If RETURN 2 is used, the calling subroutine is signaled that the basic event name does not contain the search string and no further searching with that basic event is necessary.

DOSRCH and INPUT are the only subroutines that call CHEAT. CHEAT does not call any subroutines. Table E.7 lists important variables used in this subroutine.

E.2.6 CHEKIT (GATE, NGI, NCI, IGNDX, ELVATN, KEY, LAMBDA, TAU, HOUSE, FLOOD, NHOUSE)

Subroutine CHEKIT checks the fault tree, basic event and house event input data for errors. CHEKIT checks the fault tree for the following possible errors:

1. a gate described in the fault tree that is not input to any other gate,

2. a gate is described more than once in the fault tree (duplicate cards),

3. the same gate has more than one set of inputs (possibly a misspelled gate name),

4. more than one gate has the same set of inputs, and

Table E.6 BUILD Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Threaded pseudo-binary image of the fault tree
TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic event in the fault tree
TRENDX(NDIM)	I*2	Location in TREE where each member first appears
IGATYP(NG)	I*2	Type of gate (0 = OR, 1 = AND)
NAM(NBE+NG)	R*8	Basic event and gate names
LEVATN(NBE)	I*4	Elevation of each basic event
NREP(NBE)	I*2	Number of times each basic event is repeated in the tree
HOUSE(NBE)	I*2	State of each house event in the tree
FLOOD(NBE)	I*2	Flood susceptibility of each house event
LAMBDA(NBE,2)	R*4	Basic event unflooded and flooded failure rates
TAU(NBE,2)	R*4	Basic event unflooded and flooded mean down times
NGI(NG)	I*2	Number of gates input to each gate
NCI(NG)	I*2	Number of basic events input to each gate
IGTYP(NG)	I*2	Gate type of each gate (0 = OR, 1 = AND)
GATE(NG,MAXIN)	R*8	Input descriptions of fault tree
ELVATN(NG,MAXIN)	I*4	Basic event elevations
INDEX(NG,MAXIN)	I*2	Gate and basic event indices. (Each gate is numbered from 1 to NG and each gate input is numbered starting with NG+1. These indices are used to build the circular queue).

Table E.6 BUILD Variables (Continued)

Variable	Word Type	Description
IHOUSE(NG,MAXIN)	I*2	State of each house event
IFLOOD(NG,MAXIN)	I*2	Flood susceptibility of each house event
ILAMDA (NG,MAXIN,2)	R*4	Basic event unflooded and flooded failure rates
ITAU(NG,MAXIN,2)	R*4	Basic event unflooded and flooded mean down times
GNUM (NGATE*NCOMP)	I*2	Circular queue for building the pseudo-binary image of the fault tree
NXTET	I*4	Next empty slot in TREE
NXTEG	I*4	Next empty slot in GNUM
NXTF	I*4	Next slot in GNUM to process
NXG	I*4	Number of gate retrieved from GNUM
NXE	I*4	Number of basic event retrieved from GNUM
NXTGAT	I*4	Gate or basic event index number (NGE or NXE) of item currently being processed

For definitions of dimension variables see Table E.5.

Table E.7 CHEAT Variables

Variable	Word Type	Description
COMPNT(8)	L*1	String requested by the user
BENAME(8)	L*1	Basic event name

5. the fault tree contains circular logic.

The basic events are checked for valid elevations and the failure rates and mean down times are checked for validity. CHEKIT checks the appropriate house event arrays to insure that the existence or nonexistence of house events is properly indicated.

Subroutine INPUT calls CHEKIT and CHEKIT calls subroutine GOOFUP. Table E.8 describes important variables used in this subroutine.

E.2.7 CONDNS (NSET, IMIC, MICS, IDIM)

Subroutine CONDNS removes supersets from the cut set array by compressing the cut set array to eliminate the holes left by supersets and adjusts the number of sets accordingly.

Subroutine CONDNS does not call any other subroutines. Subroutines DEQUAL, DSUPER, FIXIT, PARTAL and TOOBIG call CONDNS. Table E.9 lists important variables used in this subroutine.

E.2.8 DEQUAL (NSET, IMIC, IIMIC)

Subroutine DEQUAL deletes duplicate cut sets in the cut set array. DEQUAL compares each of the basic events in one cut set with the basic events in another cut set. If all the basic events in the two sets match, DEQUAL deletes one of the cut sets.

Subroutine DEQUAL calls subroutine CONDNS and is called by subroutine GATHER. Table E.10 describes important variables used in this subroutine.

E.2.9 DISK (NSET, ORDER, IIMIC, LUN1, LUN2, NUN1)

Subroutine DISK deletes supersets from the cut sets stored on disk. The array IIMIC contains only cut sets of a given order. Cut sets larger than this given order are stored on disk and are eliminated by not being transferred from LUN1 to LUN2, thereby eliminating supersets.

DISK does not call any subroutines and is called only by subroutine GATHER. Table E.11 describes important variables used in this subroutine.

E.2.10 DOIT (TREE, TRESAV, TREEL, STREEL, TREEX, LEVATN, KEY, HOUSE, FLOOD, NPL, ONES, MAX)

Subroutine DOIT controls the flood simulation. As the fault tree is flooded level by level, DOIT determines if any cut sets are

Table E.8 CHEKIT Variables

Variable	Word Type	Description
GATE(NG,MAXIN)	R*8	Input fault tree
NGI(NG)	I*2	Number of gates input to each gate
NCI(NG)	I*2	Number of basic events input to each gate
IGNDX(NG)	I*2	Internal array used in testing for circular logic
ELVATN(NG,MAXIN)	I*4	Basic event elevations
KEY(NDEP)	I*4	Flood levels used in performing a flood analysis
LAMBDA(NG,MAXIN,2)	R*4	Basic event unflooded and flooded failure rates
TAU(NG,MAXIN,2)	R*4	Basic event unflooded and flooded mean down times
HOUSE(NG,MAXIN)	I*2	State of each house event
FLOOD(NG,MAXIN)	I*2	Flood susceptibility of each house event
NHOUSE	I*4	Number of house events

For definition of dimension variables see Table E.5.

Table E.9 CONDNS Variables

Variable	Word Type	Description
NSET	I*4	Number of cut sets
IMIC(IDIM)	I*2	Size of cut set i
MICS(IDIM,1)	I*2	Cut set array
IDIM	I*4	Parameter used to dimension array MICS

Table E.10 DEQUAL Variables

Variable	Word Type	Description
NSET	I*4	Number of cut sets
IMIC(CROW)	I*2	Order of each cut set
IIMIC(CROW,CCOL)	I*2	Cut sets

CROW is the estimated number of BICS.

CCOL is the estimated order of the longest possible BICS.

Table E.11 DISK Variables

Variable	Word Type	Description
NSET	I*4	Number of cut sets stored on disk LUN1
ORDER	I*4	Order of the cut sets on disk LUN1
IIMIC(CROW,CCOL)	I*2	Cut sets
LUN1	I*4	Disk file cut sets to be tested
LUN2	I*4	Disk file where non-supersets will reside
NUN1	I*4	Number of cut sets on LUN2

CROW is the estimated number of BICS.

CCOL is the estimated order of longest possible BICS.

submerged. If cut sets are submerged, DOIT invokes the proper algorithms to determine these cut sets and shrink them to minimal form.

Figure E.4 is a flowchart of DOIT. To begin the flood simulation, all the basic events in the fault tree are turned off and the fault tree is set to false. DOIT then floods the fault tree to the appropriate level and turns on all the basic events whose elevations are at or below this level. DOIT tests this flooded fault tree. If the fault tree is true, DOIT invokes the proper algorithms to identify the minimal cut sets. If the fault tree is false, the flood is raised to the next level and the fault tree is retested.

In determining the minimal cut sets, DOIT first identifies the one-event minimal cut sets. It then eliminates these basic events from the fault tree before searching for higher-order minimal cut sets. DOIT determines the higher-order minimal cut sets using one of two algorithms (a standard top-down algorithm or a cut set/path set algorithm), depending on whether the TOP event has previously occurred or not.

After finding the minimal cut sets, the fault tree is restored and, if specified, the next level is flooded. If the fault tree never tests true during the flood simulation to level MAXD, DOIT finds the partial common cause candidates. Final output is produced and the search capability invoked after DOIT completes the flood simulation.

MAIN is the only routine that calls DOIT. Subroutines called by DOIT are SETRUE, RESET, SETIT, PRSET, FIND1S, OUTPUT, PRINTS, TRAVRS, FATRAM, DOPATH, PRINT, ALOCT3, PARTAL, DOSRCH and KITOUT. Table E.12 describes important variables used in DOIT.

E.2.11 DOPATH (IPATH, IIPATH, IPTH, IIPTH, IMIC, IIMIC, NREP, TREEX, TREEL, LTREE, TREE, TREE2, NMIC, NPATH)

Subroutine DOPATH identifies minimal cut sets for a flooded fault tree that has already tested true at a lower flood level. The DOPATH algorithm identifies new minimal cut sets without refinding minimal cut sets previously identified at lower flood levels. Figure E.5 is a flowchart of DOPATH.

DOPATH uses the following procedure to identify flooded minimal cut sets for the n^{th} flood level:

1. Path sets (PATH) are synthesized from the minimal cut sets found for the previous n-1 levels tested (if path sets do not already exist).

2. The fault tree is tested with all the basic events in the first PATH set turned off. If

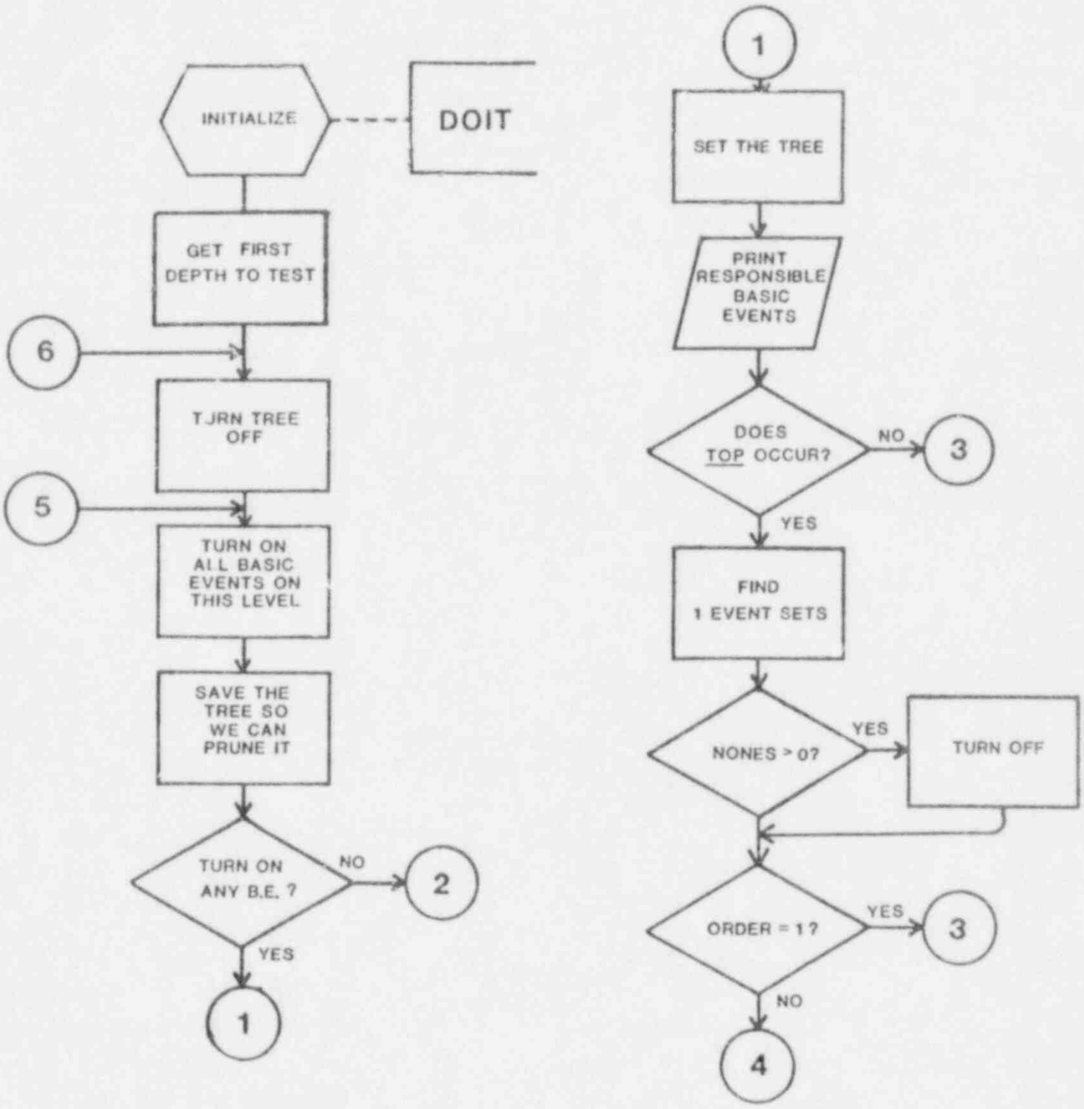

Figure E.4 Subroutine DOIT Flowchart

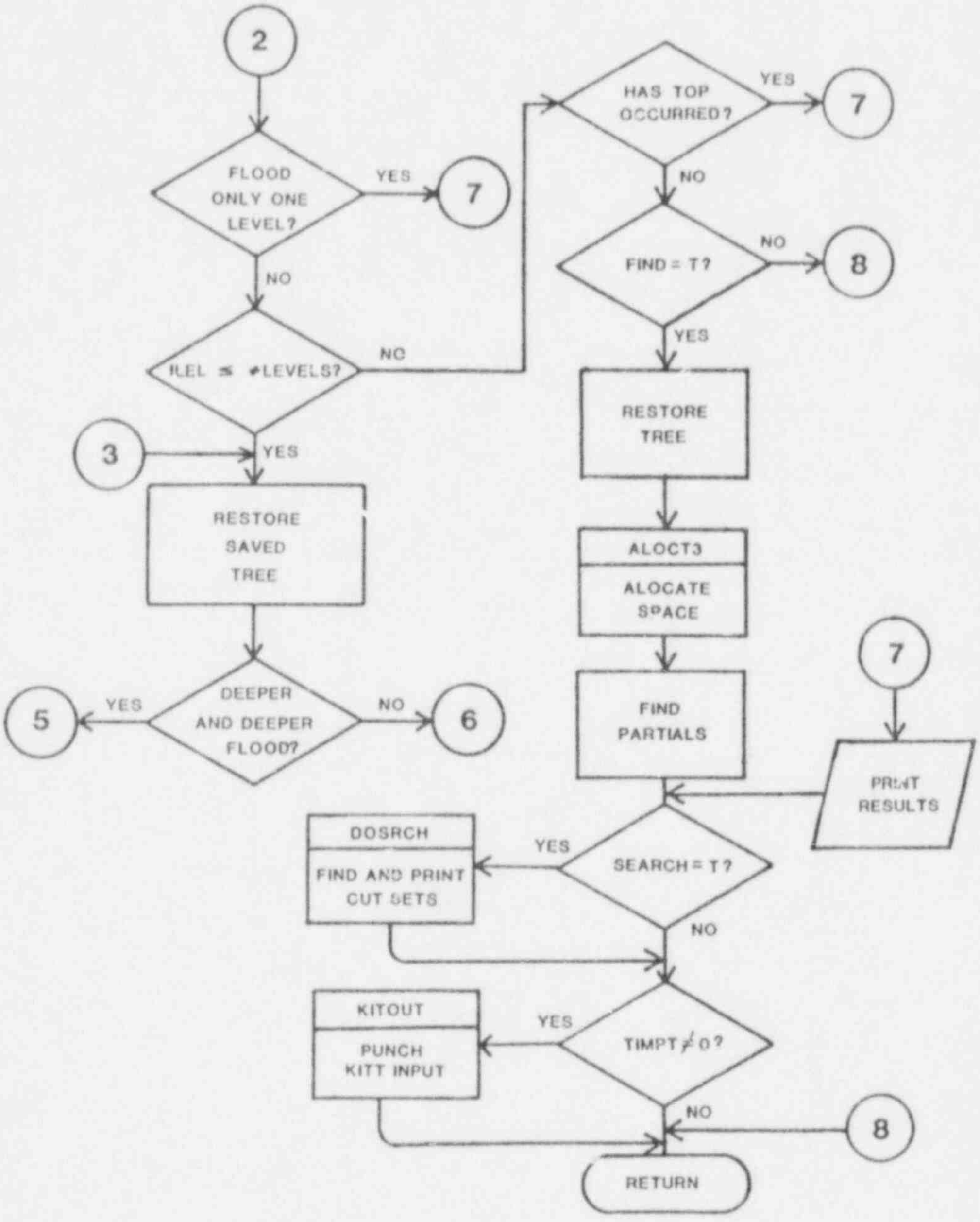

Figure E.4 Continued

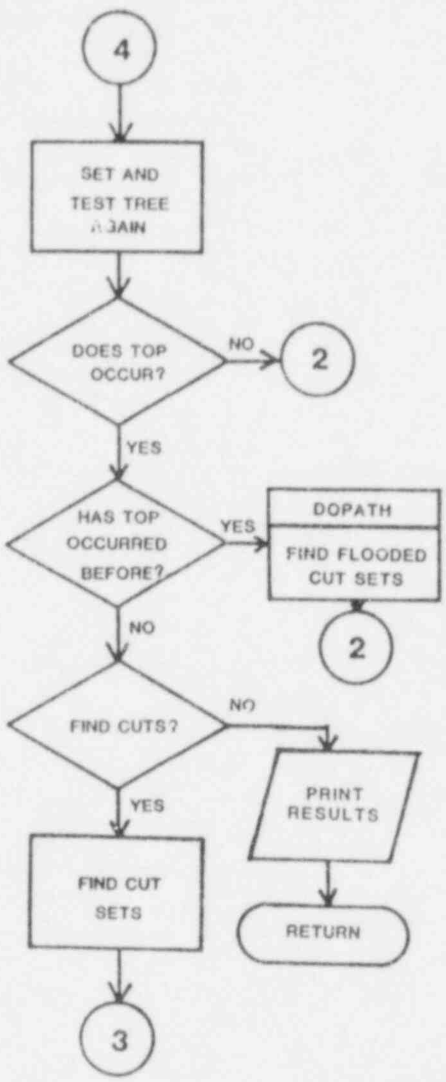

Figure E.4 Continued

Table E.12 DOIT Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Threaded pseudo-binary tree image
TRESAV(NDIM,3)	I*2	Duplicate of the pseudo-binary tree pointers (save area)
TREEL(NDIM)	L*1	Logical state of the tree
STREEL(NDIM)	L*1	Duplicate of the state of the tree (save area)
TREEX(NBE,MAXREP)	I*2	Location in TREE of each basic event
LEVATN(NBE)	I*4	Basic event elevations
KEY(NDEP)	I*4	Flood levels used in performing a flood analysis
HOUSE(NBE)	I*2	State of each house event
FLOOD(NBE)	I*2	Flood susceptibility of each house event
NPL(NLEV)	I*2	Number sets per level
ONES(NBE)	I*2	One-event cut sets
MAX	I*4	Amount of room in common block /WORK/
THSLOC	I*4	Upper bound on this depth
LSTLOC	I*4	Lower bound on this depth

For definitions of dimension variables see Table E.5.

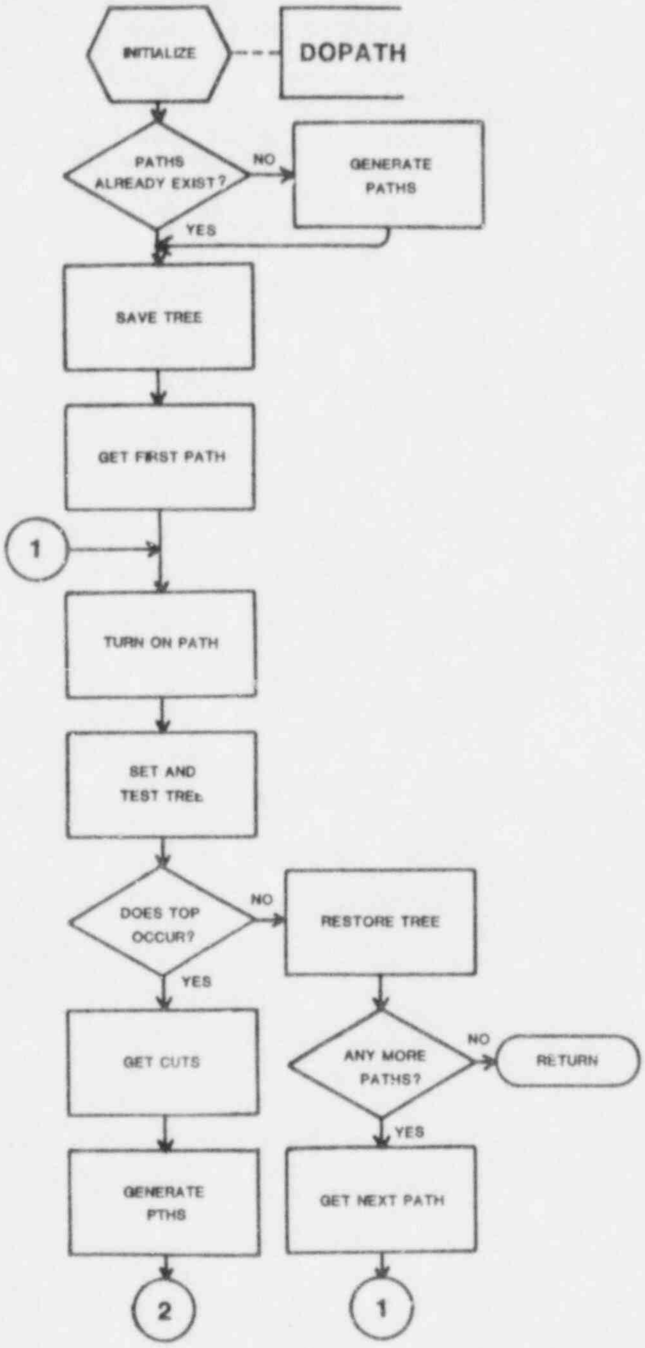

Figure E.5 Subroutine DOPATH Flowchart

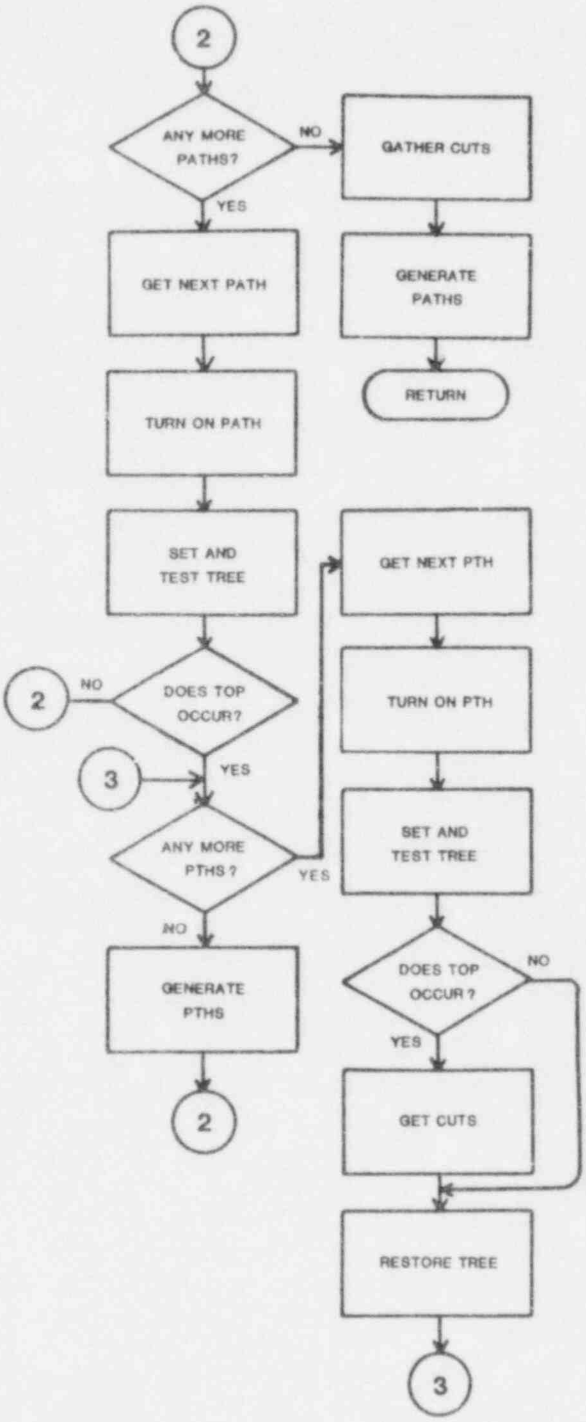

Figure E.5 Continued

the tree is false, the basic events in the first PATH set are restored and the fault tree is retested with the next PATH set off. This process is repeated until the fault tree tests true.

3. When the fault tree tests true, the new cut sets are determined and new path sets (PTH) are synthesized from only these cut sets.

4. The fault tree is then tested with the next old PATH set turned off. If the fault tree is false, the PATH set is restored and the fault tree is tested with the next old PATH set off. This continues until the fault tree tests true.

5. After the fault tree tests true, it is retested with the current old PATH set and each of the new PTH sets individually turned off. Each time the fault tree is true in this case, more new minimal cut sets are determined and added to the list of new minimal cut sets.

6. The list of new PTH sets is updated at this point and step 4 is repeated.

DOPATH repeats this procedure until the fault tree has been tested with each of the old PATH sets turned off. It then updates the list of minimal cut sets and path sets for the entire flooded portion of the fault tree and proceeds to flood the next level.

DOIT is the only subroutine that calls DOPATH. DOPATH calls subroutines GENPTH, SETPTH, TRAVRS, SETIT, FATRAM, GENP2,GATHER, GENP3, OUTPUT. Table E.13 describes important variables used in DOPATH.

E.2.12 DOSRCH (NAM, LVTN, NPL, KEY)

Subroutine DOSRCH finds and prints all minimal cut sets containing a user-specified basic event name or any other string of characters. DOSRCH searches each disk file of minimal cut sets and prints out the minimal cut sets that match as they are found. DOSRCH uses only one search string during each complete check of the minimal cut sets.

DOSRCH is called by subroutine DOIT and it calls subroutine CHEAT. Table E.14 describes important variables used in this subroutine.

Table E.13 DOPATH Variables

Variable	Word Type	Description
IPATH(PROW)	I*2	Order of each path set
IIPATH(PROW,PCOL)	I*2	Path sets
IPTH(PROW)	I*2	Order of each secondary PTH set
IIPTH(PROW,PCOL)	I*2	Secondary PTH sets
IMIC(CROW)	I*2	Order of each cut set
IIMIC(CROW,CCOL)	I*2	Cut sets
NREP(NBE)	I*2	Number of times each basic event is repeated in the fault tree
TREEX(NBE,MAXREP)	I*2	Location of each basic event in the fault tree
TREEL(NDIM)	L*1	State of the fault tree
LTREE(NDIM,2)	L*1	Duplicate of the state of the tree (save area)
TREE(NDIM,4)	I*2	Threaded pseudo-binary image of the fault tree
TREE2(NDIM,3)	I*2	Duplicate of the threaded pseudo-binary image of the fault tree (save area)
NMIC	I*4	Number of cut sets
NPATH	I*4	Number of path sets

For definitions of dimension variables see Table E.5.

Table E.14 DOSRCH Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
NPL (NLEV)	I*2	Number of MICS per level
KEY(NDEP)	I*4	Height of each flood level
MICS(20)	I*4	Minimal cut sets
BENAM	R*8	Basic event name searched for
COMPNT	R*8	Character string searched for

For definitions of dimension variables see Table E.5.

E.2.13 DSUPER (NXTEL, IMIC, MICS, IDIM)

Subroutine DSUPER deletes supersets from the list of cut sets. DSUPER compares lower-order cut sets against higher-order cut sets and deletes those higher-order cut sets in which the lower-order cut sets are completely contained. DSUPER deletes a cut set by setting the value in IMIC to zero.

Subroutine DSUPER calls CONDNS. It is called by FATRAM, FIXIT, GATHER, GENPTH, GENP2, and GENP3. Table E.15 describes important variables used in subroutine DSUPER.

E.2.14 EXIST (ISON, IGATYP, KEY)

Logical function EXIST determines the gate types input to the logic gate currently being resolved.

Function EXIST does not call any other subroutines and is invoked by subroutine FATRAM. Table E.16 describes important variables in subroutine EXIST.

E.2.15 FATRAM (TREE, TRENDX, IGATYP, ISGATE, FLAG, IMIC, MICS, NCUT, NHOLD)

Subroutine FATRAM identifies the minimal cut sets for the fault tree. The FATRAM subroutine in NOAH is a modified version of a MOCUS-type top-down replacement algorithm developed by D. M. Rasmusson and N. H. Marshall.

Figure E.6 is a flowchart of subroutine FATRAM. FATRAM begins by expanding the highest gate in the tree (TOP gate) into pseudo-cut sets composed of gates and/or basic events. FATRAM then expands all AND gates in these pseudo-cut sets until only OR gates and basic events are left. Next, FATRAM expands all OR gates in the pseudo-cut sets until only AND gates and basic events are left. FATRAM then deletes supersets from this pseudo-cut set list. AND gates are then resolved until no AND gates are left. OR gates in the pseudo-cut sets are resolved. The process of resolving AND and OR gates in the pseudo-cut sets and deleting supersets continues until the cut sets contain only basic events. These are the system minimal cut sets. Subroutine FIXIT is called any time the cut set array fills up during the expansion of OR gates.

After identifying the minimal cut sets, minimal cut sets of order larger than variable ORDER are deleted. If any cut sets are stored on disk, FATRAM calls subroutine GATHER to complete the processing. If no cut sets are stored on disk, FATRAM calls subroutine OUTPUT.

Table E.15 DSUPER Variables

Variable	Word Type	Description
NXTEL	I*4	Next empty slot in the MICS array
IMIC(IDIM)	I*2	Size of cut sets
MICS(IDIM,1)	I*2	Cut sets
IDIM	I*4	Dimension of IMICS array
NSET	I*4	Number of cut sets or flood protection sets in MICS array
LONG	I*4	Order of the longer cut set/flood protection set stored in MICS
SHORT	I*4	Order of the shorter cut set/flood protection set stored in MICS
ILONG	I*4	Index of the longer cut set/flood protection set stored in MICS
ISHORT	I*4	Index of the shorter cut set/flood protection set stored in MICS

Table E.16 EXIST Variables

Variable	Word Type	Description
ISON	I*4	Index of input gate examined
IGATYP(NG)	I*2	Gate type of each fault tree gate
KEY	I*4	Gate type of gate currently being resolved
EXIST	L*4	Flag indicating whether or not ISON and KEY gate types match

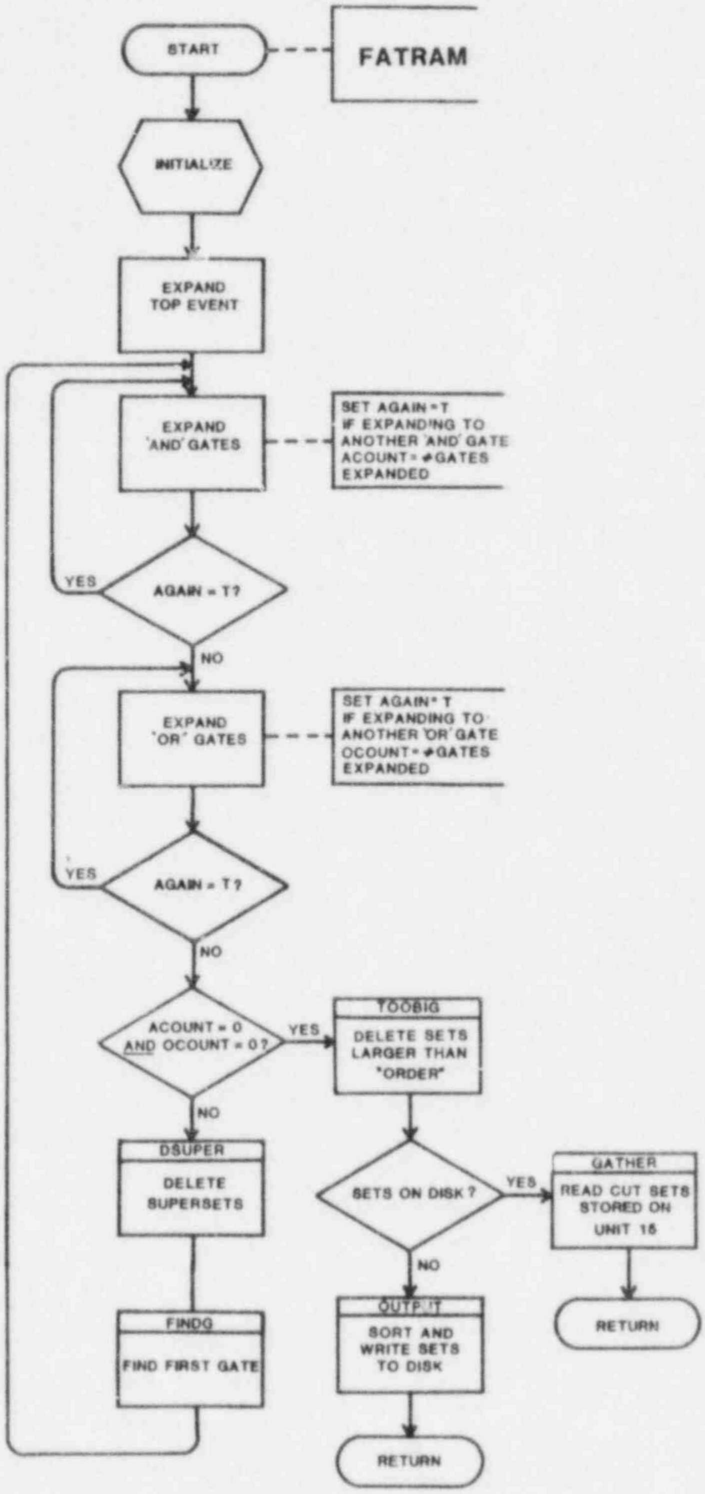

Figure E.6 Subroutine FATRAM Flowchart

Subroutine FATRAM calls subroutines FIXIT, DSUPER, FINDG, TOOBIG, GATHER, OUTPUT, EXIST, FGATE, and PREXST. Subroutines that call FATRAM are DOIT, DOPATH, PARTAL. Table E.17 describes important variables used in this subroutine.

E.2.16 FGATE (MICS, NWIDE, NSET)

Function FGATE determines which columns (if any) in the row currently being analyzed in MICS (NSET,I) contain a gate. If the element index number in any column is greater than NBE, then that element is a gate.

FGATE does not call any subroutines and is called only by FATRAM. Table E.18 describes important variables in FGATE.

E.2.17 FINDG (NMIC, IMIC, IIMIC, ISGATE, FIRST)

After supersets are deleted, FINDG sets the array ISGATE for the first column containing a gate for each cut set. It also returns the first row containing a gate. FINDG calls no routines. FINDG is called by FATRAM and FIXIT. Table E.19 describes important variables in FINDG.

E.2.18 FIND1S (TREE, TREEL, TREEX, NREP, ONES, NAM, IGTYP, KEY, LEVN)

Subroutine FIND1S identifies one-event minimal cut set for the fault tree. If the basic event was not repeated FIND1S determines if the event is a one-event minimal cut set by tracing the path of the basic event through the fault tree to the TOP event. If a basic event encounters no AND gates in its path to the TOP event, it is a one-event minimal cut set. If an event is repeated, the tree is turned off, all occurrences of the event are set true and the tree is tested for occurrence of the TOP. If the TOP occurs, the basic event is a one-event cut set. After finding the one-event minimal cut sets, FIND1S stores them on unit 21.

FIND1S calls SETRUE and SETIT and is called only by DOIT. Table E.20 describes important variables used in FIND1S.

E.2.19 FIXIT (*, MICS, IMIC, NSETP1, NHOLD, ISGATE, FIRST)

Subroutine FIXIT removes cut sets from the MICS array that contain only basic events and stores these cut sets on unit 15. Subroutine FIXIT is called whenever the MICS array becomes full. If no cut sets can be removed from MICS (each cut set still contains at least one unresolved gate), an error message is generated and program execution stops.

Table E.17 FATRAM Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Threaded pseudo-binary image of the fault tree
TRENDX(NDIM)	I*2	Index number of each gate in the tree
IGATYP(NG)	I*2	Gate type of each gate
ISGATE(CROW)	I*2	Index number of the first gate in each row of the MICS array
FLAG	L*4	Flag indicating whether or not: 1) there are too many cut sets for the cut set array (FLAG = F), or 2) FATRAM is called from DOPATH (FLAG = F)
IMIC(CROW)	I*2	Size of each row in MICS
MICS(CROW,CCOL)	I*2	Cut sets
NMIC	I*4	Number of cut sets in MICS array when leaving the FATRAM subroutine
NHOLD	I*4	Number of cut sets stored on disk
FIXUP	L*4	Flag indicating whether or not cut sets are stored on disk
NXTEL	I*4	Next empty row in MICS
IGATE	I*4	Row in array TREE currently being processed
IGTYP	I*4	Type of gate currently being expanded
SON	I*4	First input to gate currently being expanded

For definitions of dimension variables see Table E.5.

Table E.17 FATRAM Variables (continued)

Variable	Word Type	Description
ISON	I*4	Index number of the first input to the gate currently being expanded
BRO	I*4	Next input to the gate currently being expanded
IBRO	I*4	Index number of the next input to the gate currently being expanded
IWIDE	I*4	Order of pseudo-cut set containing a gate currently being expanded
NWIDE	I*4	Order of current pseudo-cut set containing a gate currently being expanded
NSET	I*4	Number of pseudo-cut sets that currently exist
MATIJ	I*4	Index number of item in the MICS array that is currently being examined
PREXST	L*4	Function to determine pre-existence of ISON in the row currently being analyzed in MICS (see section D.2.38)
EXIST	L*4	Function to determine the gate type of the item just placed in the row currently being analyzed in MICS (see D.2.14)
FGATE	I*2	Function to determine the first column in the MICS row currently being analyzed that contains a gate (see E.2.16)
AGAIN	L*4	Logical variable set to true (T) if EXIST is true
ACOUNT	I*4	Number of AND gates expanded
OCOUNT	I*4	Number of OR gates expanded

Table E.18 FGATE Variables

Variable	Word Type	Description
MICS(CROW,CCOL)	I*2	Cut sets
NWIDE	I*4	Size of MICS
NSET	I*4	Particular row of the MICS to analyze
FGATE	I*2	Column of the first gate in the MICS row currently being analyzed

CROW is the estimated number of BICS.

CCOL is the order of the longest possible BICS.

Table E.19 FINDG Variables

Variable	Word Type	Description
NMIC	I*4	Number of cut sets
IMIC(CROW)	I*2	Size of cut set
IIMIC(CROW,CCOL)	I*2	Cut set
ISGATE(CROW)	I*2	Array containing location of first gate in row
FIRST	I*4	First row that contains a gate (if FIRST = 0 then done)

CROW is the estimated number of BICS.

CCOL is the order of BICS.

Table E.20 FIND1S Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Fault tree
TREEL(NDIM)	L*1	Logical state of the fault tree
TREEX(NBE,MAXREP)	I*2	Index number of each basic event
NREP(NBE)	I*2	Number of times each basic event is repeated in the fault tree
ONES(NBE)	I*2	One-event minimal cut sets
NONES	I*4	Number of one-event sets
NAM(NBE)	R*8	Basic event names
IGTYP(NG)	I*2	Gate types
KEY(NDEP)	I*4	Level Keys
LEVN(NBE)	I*4	Elevation of each basic event

For definition of dimension variables see Table E.5.

Subroutine FIXIT calls DSUPER, FINDG, GOOFUP, and CONDNS. It is called by FATRAM. Table E.21 lists important variables used in FIXIT.

E.2.20 GATHER (MICS, IMIC, NSETP1, NHOLD)

Subroutine GATHER reads the cut sets stored on unit 15 and deletes supersets. GATHER uses one of two procedures for deleting supersets, depending on the number of cut sets that were determined. If all of the cut sets stored on unit 15 will fit in MICS, the first procedure is used. In this case, GATHER copies the unit 15 cut sets into MICS, deletes all the supersets, and calls OUTPUT.

If there are too many cut sets on unit 15 to fit in MICS, the second procedure is used. Using the second procedure, GATHER writes all cut sets currently in the MICS array to unit 15. Then all two-event sets are read in and all higher-order sets are written to unit 16. GATHER then deletes any duplicate two-event minimal cut sets in MICS and copies the two-event minimal cut sets onto unit 22. The unit used corresponds to the order of the cut sets plus twenty (2 + 20 = 22). Finally, GATHER compares the cut sets on unit 16 with the two-event minimal cut sets in MICS to identify supersets. Those unit 16 cut sets that are not supersets are rewritten to unit 15. At this point, GATHER starts the process over again with three-event cut sets stored in MICS. This procedure is repeated until all minimal cut sets are found.

Subroutine GATHER calls DSUPER, OUTPUT, DEQUAL, and DISK. Subroutines that call GATHER are DOPATH, FATRAM, and PARTAL. Table E.22 describes important variables used in GATHER.

E.2.21 GENPTH (IPATH, IIPATH, NPATH, IWORK, IIWORK, IMIC, IIMIC, NMIC)

Subroutine GENPTH synthesizes the flood protection sets from the minimal cut sets for the fault tree. Figure E.7 is a flowchart of GENPTH. The first time GENPTH is called, IWIDE one-event flood protection sets are created from the first minimal cut set where IWIDE equals the order of the minimal cut set. This set of flood protection sets then serves as the "current" flood protection sets. Next, GENPTH reads the next minimal cut for the fault tree and begins synthesizing new flood protection sets. This is accomplished by generating all possible combinations of "current" flood protection sets with single basic events from the current minimal cut set. GENPTH then deletes supersets in these combinations and makes these flood protection sets the new "current" flood protection sets. At this point, another minimal cut set is selected and the process starts over again. This continues until the last minimal cut set is analyzed.

Subroutines called by GENPTH are MYSTIC, DSUPER, and SORTP. GENPTH is called by subroutines DOPATH and PARTAL. Table E.23 lists important variables used in GENPTH.

Table E.21 FIXIT Variables

Variable	Word Type	Description
MICS(CROW,CCOL)	I*2	Cut sets
IMIC(CROW)	I*2	Order of each cut set
NSETP1	I*4	Next empty row in MICS
NHOLD	I*4	Number of cut sets on Unit 15
ISGATE(CROW)	I*2	Index number of the first gate in each row of the MICS array
FIRST	I*4	First row in MICS containing a gate
NSET	I*4	Number of cut sets in MICS
FOUND	I*4	Flag indicating if any rows are removed from MICS

CROW is the estimated number of BICS.

CCOL is the order of the longest possible BICS.

Table E.22 GATHER Variables

Variable	Word Type	Description
MICS(CROW,CCOL)	I*2	Cut sets
IMIC(CROW)	I*2	Order of each cut set
NSETP1	I*4	Next empty row in MICS
NHOLD	I*4	Number of cut sets in Unit 15
LUN1,LUN2	I*4	Disk file on Unit 15 or 16
NSET	I*4	NSETP1-1
NMAX	I*4	NSET + NHOLD
ORDER	I*4	Order of the cut set under analysis
NUN2	I*4	Number of cut sets on disk files

CROW is the estimated number of BICS.

CCOL is the order of the longest possible BICS.

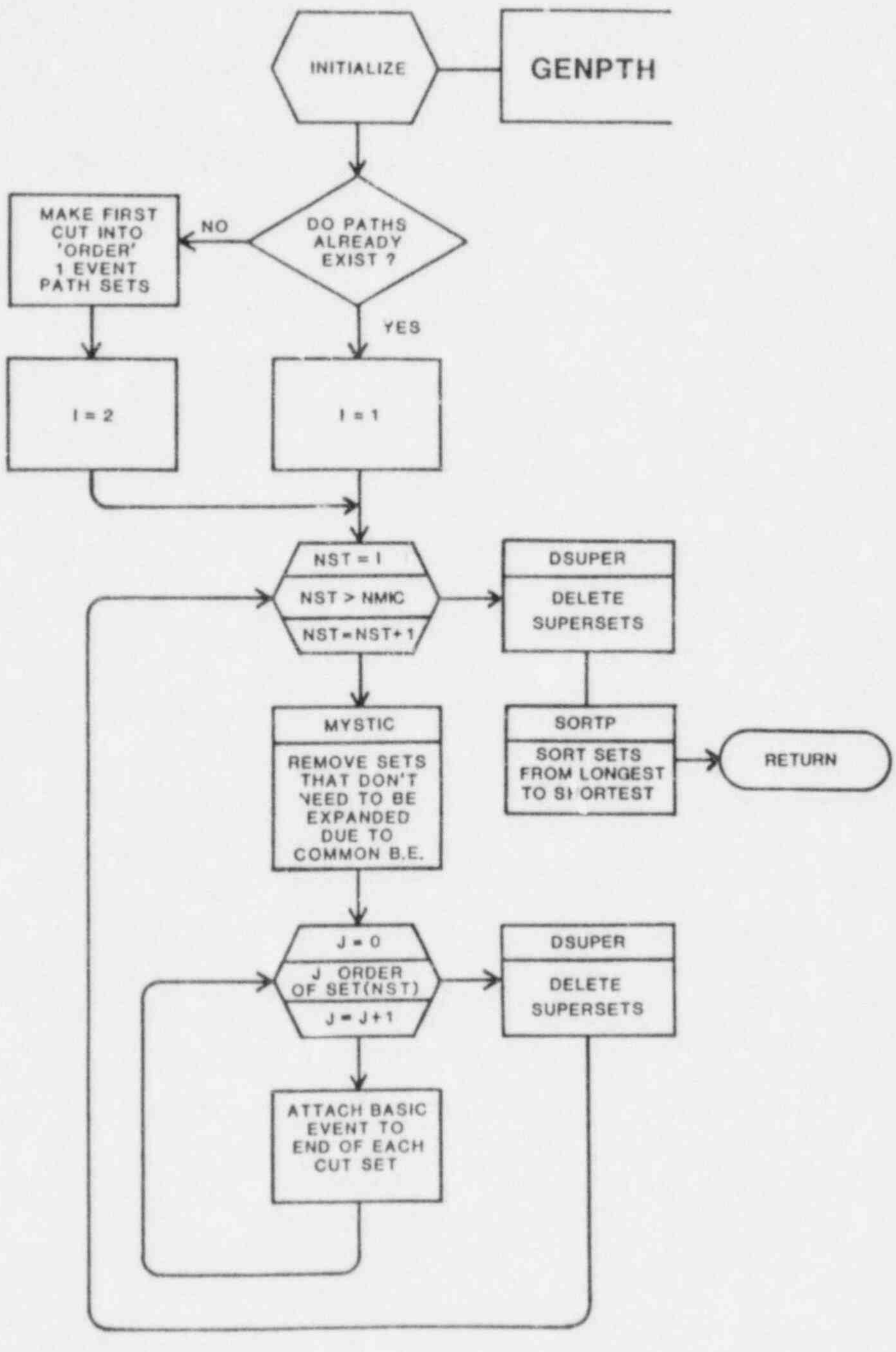

Figure E.7 Subroutine GENPTH Simplified Flowchart

Table E.23 GENPTH Variables

Variable	Word Type	Description
IPATH(PROW)	I*2	Order of each flood protection set
IIPATH (PROW,PCOL)	I*2	Flood protection set
NPATH	I*4	Number of flood protection sets
IWORK(PROW)	I*2	Order of each flood protection set in the work array
IIWORK (PROW,PCOL)	I*2	Flood protection set work space
IMIC(CROW)	I*2	Order of each cut set
IIMIC(CROW,CCOL)	I*2	Cut sets
NMIC	I*4	Number of cut sets

For definitions of dimension variables see Table E.5.

E.2.22 GENP2 (IPATH, IIPATH, NPATH, IWORK, IIWORK, NALL)

Subroutine GENP2 generates flood protection sets from minimal cut sets. The algorithms used in GENP2 are identical to those in GENPTH. GENP2 is invoked instead of GENPTH whenever the minimal cut sets are stored on disk (unit 15) instead of in memory.

Subroutine GENP2 calls MYSTIC, DSUPER, and SORTP. Subroutines DOPATH and PARTAL call GENP2. Table E.24 describes important variables in this subroutine.

E.2.23 GENP3 (NPATH, IPATH, IIPATH, NPTH, IPTH, IIPTH, IWORK, IIWORK)

Subroutine GENP3 generates the final list of flood protection sets for each level from the PATH sets and PTH sets (see E.2.11 DOPATH). GENP3 generates the final flood protection sets by finding all combinations of PATH sets and PTH sets, and then deleting all supersets.

GENP3 calls subroutines DSUPER and SORTP. GENP3 is called by subroutine DOPATH. Table E.25 describes important variables used in this subroutine.

E.2.24 GETMAX (GATE, NGI, NCI, GATYP, IWORK, ROW, COL)

Subroutine GETMAX determines the number and maximum order of the Boolean Indicated Cut Sets (BICS) and Boolean Indicated Path Sets (BIPS) for the fault tree. IWORK(1,*) and IWORK(2,*) are estimates of the number and maximum order of the BICS, respectively. The algorithm used in GETMAX is based on a similar algorithm in the MOCUS computer program.

To determine the BICS's for a fault tree, GETMAX first identifies all the gates in the fault tree that have only basic events as inputs. GETMAX then determines the BICS's and maximum order of BICS for each of these gates. If gate N being resolved is an AND gate, IWORK(1,N) is set to 1 and IWORK(2,N) is set to the number of basic events input to gate N. If gate N is an OR gate, IWORK(1,N) is set to the number of basic events input to gate N and IWORK (2,N) is set to 1.

GETMAX next resolves gates with resolved inputs. If any input gates are not yet resolved, GETMAX proceeds to analyze another gate. If all input gates are resolved, GETMAX determines the values of IWORK for this gate. First, it resolves the basic event inputs to the gate. The rules stated previously for AND and OR gates apply here. Next, the values of the input gates IWORK's are incorporated into this gate's IWORK values. If OR gate M is being resolved, IWORK(1,M)

Table E.24 GENP2 Variables

Variable	Word Type	Description
IPATH(PROW)	I*2	Order of each flood protection set
IIPATH(PROW,PCOL)	I*2	Flood protection sets
NPATH	I*4	Number of flood protection sets
IWORK(PROW)	I*2	Order of each flood protection set in the work array
IIWORK(PROW,PCOL)	I*2	Flood protection set work space
NALL	I*4	Number of cut sets on disk
IIMIC(20)	I*2	Space for one minimal cut set

PROW is the estimated number of BIPS.

PCOL is the order of the longest possible BIPS.

Table E.25 GENP3 Variables

Variable	Word Type	Description
NPATH	I*4	Number of PATH flood protection sets
IPATH(PROW)	I*2	Order of each PATH flood protection set
IIPATH(PROW,PCOL)	I*2	PATH flood protection sets
NPTH	I*4	Number of PTH flood protection sets
IPTH(PROW)	I*2	Order of each PTH flood protection set
IIPTH(PROW,PCOL)	I*2	PTH flood protection set
IWORK(PROW)	I*2	Order of each set
IIWORK(PROW,PCOL)	I*2	Work area for sets

PROW is the estimated number of BIPS.

PCOL is the order of the longest possible BIPS.

equals the sum of all the input gate IWORK(1,N)'s and IWORK(2,M) equals the maximum of all IWORK(2,N)'s for gate M. If an AND gate is being resolved, IWORK(1,M) equals the product of the input gate IWORK(1,N)'s and IWORK(2,M) equals the sum of all the input gate IWORK(2,N)'s. GETMAX repeats this process until all gates are resolved.

Subroutine GETNG calls GETMAX. Subroutine GETMAX calls no other subroutines. Table E.26 describes important variables used in this subroutine.

E.2.25 GETNBE (GATE, NGI, NCI, IGTYP, NAM, NGNC, BENAM, BENUM)

Subroutine GETNBE counts the actual number of gates and basic events in the fault tree. It also counts the number of times a basic event appears in the fault tree and it fills the IGTYP, NGI and NCI arrays.

Subroutine GETNBE calls GOOFUP and is called by GETNG. Table E.27 describes important variables used in GETNBE.

E.2.26 GETNG (GATE, MAXNG, MAX)

Subroutine GETNG counts the number of unique gates (NG) and basic events (NBE) in the fault tree input. Subroutines called by GETNG are GOOFUP, GETNBE, and GETMAX. Subroutine MAIN calls GETNG. Table E.28 describes important variables used in GETNG.

E.2.27 GOOFUP (IER, NUM1, NUM2, WORD1, WORD2, CARD)

Subroutine GOOFUP prints error messages whenever errors occur in the input. The error messages printed include a reference number and a brief description of the error. Appendix G lists the error messages used in NOAH, descriptions of the errors and suggested solutions.

Subroutines that call GOOFUP are MAIN, ALOCAT, BUILD, CHEKIT, FIXIT, GETNBE, GETNG, INPUT, and LAYOUT. Subroutine GOOFUP does not call any other subroutines. Table E.29 describes important variables used in this subroutine.

E.2.28 INPUT (KEYDSC, KEY, NGI, NCI, IGTYP, GATE, ELVATN, IHOUSE, IFLOOD, ILAMDA, ITAU)

Subroutine INPUT reads the user supplied input into the proper arrays for processing by NOAH. Arrays filled by INPUT are KEY, KEYDSC, GATE, NGI, NCI, GATYP, ELVATN, ILAMDA, ITAU, IFLOOD, and IHOUSE. INPUT checks each input group for certain errors as it is

Table E.26 GETMAX Variables

Variable	Word Type	Description
GATE(NDIM,7)	R*8	Input fault tree
NGI(NG)	I*2	Number of gates input to each gate in the fault tree
NCI(NG)	I*2	Number of basic events input to each gate in the fault tree
GATYP(NG)	I*2	Gate type of each gate
IWORK(2,NG)	I*2	Work area
ROW(2)	I*4	Estimated number of cut sets (BICS) or flood protection sets (BIPS)
COL(2)	I*4	Estimated maximum order of the cut sets or the flood protection sets
IWORK1I	I*4	Temporary value of IWORK(1,I)
IWORK2I	I*4	Temporary value of IWORK(2,I)
ROWI	I*4	Temporary value for maximum number of rows needed in MICS
COLI	I*4	Temporary value for maximum number of columns needed in MICS
IW1	I*4	Temporary value of IWORK(1,I)
IW2	I*4	Temporary value of IWORK(2,I)

NG is the number of unique gates.

NDIM is the number of unique gates plus the number of unique basic events.

Table E.27 GETNBE Variables

Variable	Word Type	Description
GATE(NDIM,7)	R*8	Input fault tree
NGI(NG)	I*2	Number of gates input
NCI(NG)	I*2	Number of basic events input
IGTYP(NG)	I*2	Gate type of each gate in the fault tree
NAM(NG*NBE)	R*8	Circular queue in counting the gates and basic events
NGNC	I*4	Dimension of NAM
BENAM(NBE)	R*8	Basic event names
BENUM(NBE)	I*4	Number of times each basic event is repeated
NGIK	I*4	Number of gates input to each gate
NGII	I*4	Total number of gates in the fault tree
NCII	I*4	Total number of basic events in the fault tree
NXTG	I*4	Next gate in array NAM to process
NXTE	I*4	Next empty slot in array NAM
GNAM	R*8	Gate in NAM currently being processed

For definitions of dimension variables see Table E.5.

Table E.28 GETNG Variables

Variable	Word Type	Description
GATE(NDIM,7)	R*8	Input fault tree
MAXNG	I*4	Maximum number of gate cards (in the input) NOAH can process
MAX	I*4	Size of the work array
MAXIN	I*4	Maximum number of inputs to any one gate
NG	I*4	Number of gates
NBE	I*4	Number of basic events

NDIM is NG plus NBE.

Table E.29 GOOFUP Variables

Variable	Word Type	Description
IER	I*4	Error reference number
NUM1	I*4	Duplicate of information in the input data printed in the error message
NUM2	I*4	Duplicate of information in the input data printed in the error message
WORD1	R*8	Duplicate of information in the input data printed in the error message
WORD2	R*8	Duplicate of information in the input data printed in the error message
CARD(10)	R*8	Input card printed in the error message

read. If errors are encountered, error messages are printed and program execution terminates after subroutine CHEKIT, which does the remainder of the error checking, is called.

INPUT calls subroutines GOOFUP, CHEAT, and CHEKIT. Routine MAIN calls subroutine INPUT. Table E.30 describes important variables in INPUT.

E.2.29 KITOUT (LAMBDA, TAU, LEVN, KEY, TPOINT, TDEEP)

Subroutine KITOUT punches a portion of the NOAH output in a format acceptable for KITT-2 computer program input.[4] KITOUT punches the basic event's failure rates (unflooded and flooded), mean down times (unflooded and flooded), phase boundary (time point when the basic event is submerged) and the system's minimal cut sets.

Subroutine KITOUT calls no other subroutines. It is called by subroutine DOIT. Table E.31 lists important variables used in KITOUT.

E.2.30 LAYOUT (IMAX, MAX, INDEX)

Subroutine LAYOUT prints the contents of IW which provides the starting addresses of each of the dynamically allocated arrays stored in W (see Table E.3). LAYOUT calls subroutine GOOFUP and is called by subroutine ALOCAT, ALOCT2 and ALOCT3. Table E.32 describes important variables used in this subroutine.

E.2.31 MYSTIC (ITEM, IIMIC, NWORK, IWORK, IIWORK, NHELP, IPATH, IIPATH)

Subroutine MYSTIC aids in generating flood protection sets by determining which combinations of all flood protection sets (levels 0 → n-1) and new minimal cut sets (level n only) need not be expanded because of common events. If an old flood protection set and new minimal cut set have a common basic event, then all new flood protection sets synthesized from the old flood protection set and the new minimal cut set will be supersets. MYSTIC checks all old flood protection sets against each one of the new minimal cut sets.

Subroutine MYSTIC calls no other subroutines. It is called from GENPTH and GENP2. Table E.33 describes important variables in MYSTIC.

E.2.32 OUTPTH (NAM, LVTN, KEYDSC, KEY, IPATH, NPL, PNAM, PLEV)

Subroutine OUTPTH prints the flood protection sets for each level of the fault tree analyzed. OUTPTH calls no other subroutines and is called by subroutine PRINT. Table E.34 describes important variables in OUTPTH.

Table E.30 INPUT Variables

Variable	Word Type	Description
KEYDSC(NDEP,10)	R*8	Description of each flood level
KEY(NDEP)	I*4	Height of each flood level
NGI(NG)	I*2	Number of gates input
NCI(NG)	I*2	Number of basic events input
IGATYP(NG,MAXIN)	I*2	Gate type
GATE(NG,MAXIN)	R*8	Input fault tree
ELVATN(NG,MAXIN)	I*4	Basic event elevations
IHOUSE(NG,MAXIN)	I*2	State of house events
IFLOOD(NG,MAXIN)	I*2	Flood susceptibility of house events
ILAMDA(NG,MAXIN,2)	R*4	Basic event unflooded and flooded failure rates
ITAU(NG,MAXIN,2)	R*4	Basic event unflooded and flooded mean down times

For definitions of dimension variables see Table E.5.

Table E.31 KITOUT Variables

Variable	Word Type	Description
LAMBDA(NBE,2)	R*4	Basic event unflooded and flooded failure rates
TAU(NBE,2)	R*4	Basic event unflooded and flooded mean down times
LEVN(NBE)	I*4	Basic event elevations
KEY(NDEP)	I*4	Height of each flood level
TPOINT(NTPT)	R*4	Time points used to describe the discretized flood profile
TDEEP(NTPT)	I*4	Elevations used to describe the discretized flood profile
TIM1	R*4	Time point the basic event is submerged
TIM2	R*4	Time the mission ends (for quantitative analysis)
NPHASE	I*4	Number of phases for each basic event
IBPHA	I*4	Basic event boundary condition flag
INIT	R*4	Basic event initial unavailability
IPATH	I*4	Flag indicating minimal cut sets are supplied as KITT-2 input
NCUT	I*4	Number of minimal cut sets
SLOPE	R*8	Slope of linear flood profile
NTRCPT	R*8	Intercept
LINE	R*8	Slope minus intercept

For definitions of dimension variables see Table E.5.

Table E.32 LAYOUT Variables

Variable	Word Type	Description
IMAX	I*2	Amount of space needed in array W
MAX	I*4	Amount of space available in array W
INDEX	I*4	Number of rows to print

Table E.33 MYSTIC Variables

Variable	Word Type	Description
ITEM	I*4	Number of basic events in the current minimal cut set
IIMIC(CROW,CCOL)	I*2	Minimal cut set array
NWORK	I*4	Number of minimal cut sets in the work array
IWORK(PROW)	I*2	Order of the flood protection sets in the work array
IIWORK(PROW,PCOL)	I*2	Work array
NHELP	I*4	Number of flood protection sets moved to the IIPATH array
IPATH(PROW)	I*2	Order of the flood protection sets
IIPATH(PROW,PCOL)	I*2	Flood protection sets

CROW is the estimated number of BICS.

CCOL is the order of the longest possible BICS.

PROW is the estimated number of BIPS.

PCOL is the order of the longest possible BIPS.

Table E.34 OUTPTH Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
KEYDSC(NDEP,10)	R*8	Description of each flood level
KEY(NDEP)	I*4	Height of each flood level
IPATH(PROW)	I*2	Order of each flood protection set
NPL(NLEV)	I*2	Number of cut sets on each level
PNAM(NBE)	R*8	Temporary array for printing basic event names in each flood protection set
PLEV(NBE)	I*4	Temporary array for printing basic event elevations in each flood protection set
ISUM	I*4	Number of basic events submerged between the last flood level and the current flood level
TKEY	I*4	Height of flood level currently being output
ICOUNT	I*4	Number of flood protection sets at this flood level

For definitions of dimension variables see Table E.5.

E.2.33 OUTP2 (NAM, LVTN, KEYDSC, KEY, NPATH, IPATH, IIPATH, PNAM, PLEV, NCUT)

Subroutine OUTP2 prints the flood protection sets for the fault tree after the partial common cause candidates have been determined. OUTP2 calls no other subroutines and is called by PARTAL. Table E.35 describes important variables in OUTP2.

E.2.34 OUTPUT (NMIC, IMIC, MICS, NAM, NPL)

Subroutine OUTPUT sorts the minimal cut sets according to size. OUTPUT writes each minimal cut set to a disk file on logical unit (20 + ITEM), where ITEM is the number of basic events in each minimal cut set. OUTPUT calls no other subroutines and is called by subroutines DOIT, DOPATH, FATRAM, GATHER and PARTAL. Table E.36 describes important variables in this subroutine.

E.2.35 PARTAL (TREE, TRE AV, TREEL, STREEL, HOUSE, FLOOD, LEVN, KEY, IMIC, IIMIC, ITWO, IPATH, IIPATH)

Subroutine PARTAL finds partial common cause candidates for the fault tree. It is invoked only if MAXD is reached and the TOP event has not occurred. Figure E.8 is a flowchart of subroutine PARTAL.

To identify partial common cause candidates, PARTAL calls subroutine PREPIT to identify all combinations of two basic events above elevation MAXD that will make the fault tree true when all basic events below MAXD are turned on. PARTAL then prunes from the fault tree any basic event above MAXD that is not contained in any of the two-event sets. Next, PARTAL identifies minimal cut sets for the pruned fault tree using FATRAM. Finally, PARTAL checks the list of minimal cut sets to see that they contain at least one flooded basic event and no more than two unflooded basic events. If this criterion is not met, the minimal cut set is discarded. The remaining minimal cut sets are output as partial common cause candidates and, if desired, flood protection sets are generated and output.

Subroutine DOIT calls PARTAL. Subroutines called by PARTAL are PREPIT, SETRUE, RESET, TRAVRS, FATRAM, GATHER,CONDNS, OUTPUT, POUTWC, GENP2, OUTP2, and GENPTH. Table E.37 describes important variables used in PARTAL.

E.2.36 POUTWC (NAM, LVTN, KEYDSC, KEY, PNAM, PLEV, NMIC, TREEL)

Subroutine POUTWC prints the partial common cause candidates. It also prints the elevation of the highest flooded basic event in the partial common cause candidate that is below MAXD.

Table E.35 OUTP2 Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
KEYDSC(NDEP,10)	R*8	Description of each flood level
KEY(NDEP)	I*4	Height of each flood level
NPATH	I*4	Number of path sets
IPATH(PROW)	I*2	Order of each flood protection set
IIPATH(PROW,PCOL)	I*2	Flood protection sets
PNAM(NBE)	R*8	Temporary array for printing basic event names in each flood protection set
PLEV(NBE)	I*4	Temporary array for printing basic event elevations in each flood protection set
NCUT	I*4	Number of partially flooded cut sets
ISUM	I*4	Number of basic events submerged between the last flood level and the current flood level
TKEY	I*4	Height of flood level currently being output
ICOUNT	I*4	Number of flood protection sets at this flood level

For definitions of dimension variables see Table E.5.

Table E.36 OUTPUT Variables

Variable	Word Type	Description
NMIC	I*4	Total number of minimal cut sets
IMIC(CROW)	I*2	Order of each minimal cut set
MICS(CROW,CCOL)	I*2	Minimal cut sets
NAM(NBE)	R*8	Basic event names
NPL(NDEP)	I*2	Number of minimal cut sets found at each flood level

For definitions of dimension variables see Table E.5.

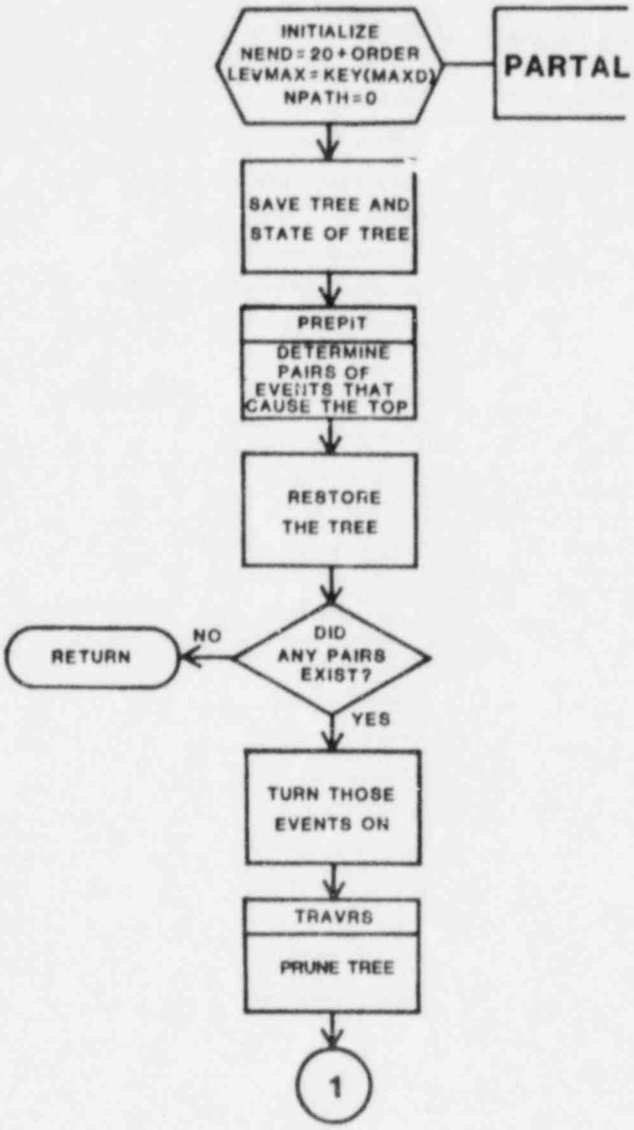

Figure E.8 Subroutine PARTAL Flowchart

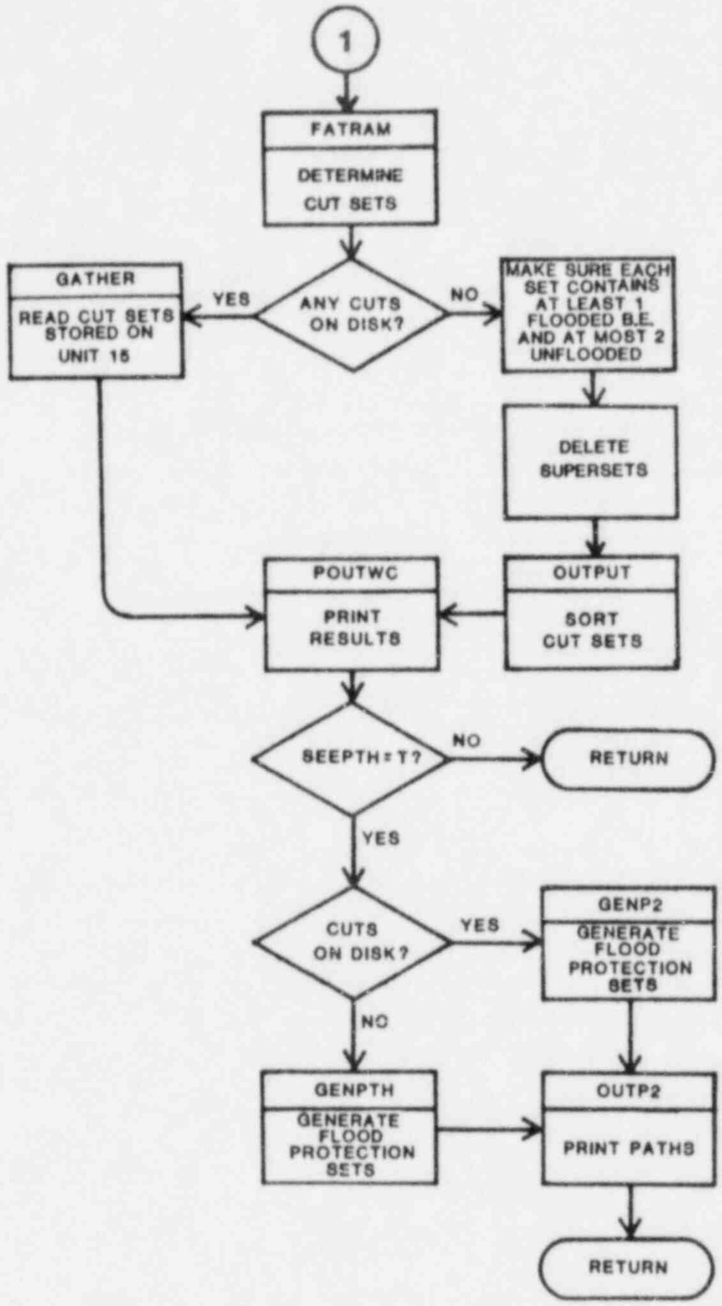

Figure E.8 Continued

Table E.37 PARTAL Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Threaded pseudo-binary image of the fault tree
TRESAV(NDIM,3)	I*2	Pseudo-binary tree image storage area
TREEL(NDIM)	L*1	Pseudo-binary image state array
STREEL(NDIM)	L*1	Pseudo-binary tree image state work area
HOUSE(NBE)	I*2	House event identifier
FLOOD(NBE)	I*2	Flood susceptibility of house events
LEVN(NBE)	I*4	Elevation of each basic event
KEY(NBE)	I*4	Height of each flood level
IMIC(CROW)	I*2	Number of basic events in each minimal cut set
IIMIC(CROW,CCOL)	I*2	Minimal cut sets
ITWO(NBE,2)	I*2	Pairs of basic events above MAXD that cause the TOP event when the flood is at level MAXD
IPATH(PROW)	I*2	Number of basic events per flood protection set
IIPATH(PROW,PCOL)	I*2	Flood protection sets
LEVMAX	I*4	Elevation of MAXD flood level
NTWOS	I*4	Number of pairs ITWO's
ISUM	I*4	Number of unflooded basic events in a partial common cause candidate

For definitions of dimension variables see Table E.5.

Subroutine POUTWC calls no other subroutines. It is called by subroutine PARTAL. Table E.38 describes important variables in this subroutine.

E.2.37 PREPIT (TREE, TREEL, TREEX, NREP, ITWO, NTWOS, NAM, LEVN, STREEL, TRESAV)

Subroutine PREPIT determines all combinations of two basic events whose elevations are above MAXD that will cause the TOP event to occur given the fault tree is submerged to MAXD and the TOP event is not on. PREPIT determines these combinations by testing to see if the fault tree is true with all the submerged basic events on and each combination of two non-submerged basic events on. If the fault tree is true, the basic events are saved. If the fault tree is false, PREPIT tests the fault tree with the next combination of two non-submerged basic events until all combinations of two non-submerged basic events have been checked.

PREPIT calls subroutines SETIT and SETRUE and is called from subroutine PARTAL. Table E.39 describes important variables in this subroutine.

E.2.38 PREXST (MICS, NSET, IWIDE, SON, HERE, KEY)

Logical function PREXST prevents a duplicate entity (gate or basic event) from being added to the cut set array in FATRAM. Subroutine PREXST calls no other subroutines and is called by FATRAM. Table E.40 describes important variables used in this subroutine.

E.2.39 PRINT (NAM, LVTN, KEYDSC, KEY, NPL, PNAM, PLEV)

Subroutine PRINT prints the minimal cut sets identified for each flood level. PRINT calls subroutine OUTPTH and is called by subroutine DOIT. Table E.41 describes important variables used in PRINT.

E.2.40 PRINTS (NAM, LVTN, KEYDSC, KEY, PNAM)

Subroutine PRINTS prints the results if screening was done. PRINTS is called from DOIT. PRINTS calls no other routines. Table E.42 describes important variables used in PRINTS.

E.2.41 PRSET (NAM, LVTN, TREEX, TREEL, PNAM, PLEV)

Subroutine PRSET prints the currently flooded basic events at each flood level if variable DUMP = T (true). PRSET calls no

Table E.38 POUTWC Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
KEYDSC(NDEP,10)	R*8	Description of each flood level
KEY(NDEP)	I*4	Height of each flood level
PNAM(NBE)	R*8	Temporary array for printing basic event names in each partial common cause candidate
PLEV(NBE)	I*4	Temporary array for printing basic event elevations in each partial common cause candidate
NMIC	I*4	Number of flood protection sets
TREEL(NDIM)	L*1	State of tree
LEVMAX	I*4	Elevation of flood level MAXD
ICOUNT	I*4	Counter for partial common cause candidates
ISUM	I*4	Number of submerged basic events in a partial common cause candidate
IDEEP	I*4	Elevation of the highest basic event in each partial common cause candidate below MAXD

NBE is the number of unique basic events.

NDEP is the number of flood levels.

NDIM is the number of unique basic events plus the number of unique gates.

Table E.39 PREPIT Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Threaded pseudo-binary image of the fault tree
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic events in the fault tree
NREP(NBE)	I*2	Number of times each basic event in the fault tree is repeated
ITWO(NBE,2)	I*2	Pairs of basic events above MAXD that cause the TOP event when the flood is at level MAXD
NTWOS	I*4	Number of pairs of two non-submerged basic events tested
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
STREEL(NDIM)	L*1	Pseudo-binary tree image state work area
TRESAV(NDIM,3)	I*2	Pseudo-binary tree image storage area
NBEP1	I*4	NBE+1
NBEM1	I*4	NBE-1

NBE is the number of unique basic events.

MAXREP is the maximum number of times any basic event appears in the fault tree.

Table E.40 PREXST Variables

Variable	Word Type	Description
MICS(CROW,CCOL)	I*2	Cut sets
NSET	I*4	Row in MICS currently being examined
IWIDE	I*4	Size of the row in MICS currently being examined
SON	I*4	Element to add to the current row
HERE	I*4	Location of the gate currently being resolved
KEY	I*4	Flag identifying gate type

CROW is the estimated number of BICS.

CCOL is the order of the longest possible BICS.

Table E.41 PRINT Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
KEYDSC(NDEP,10)	R*8	Description of each flood level
KEY(NDEP)	I*4	Level keys
NPL(NDEP)	I*2	Number of minimal cut sets identified at each level
PNAM(NBE)	R*8	Temporary array for printing basic event names
PLEV(NBE)	I*4	Temporary array for printing basic event elevations
CRITIC	I*4	Index number of the critical flood level
ICOUNT	I*4	Counter for the number of minimal cut sets
ISUM	I*4	Number of basic events submerged between flood levels
TKEY	I*4	Maximum elevation that falls within a flood level
TKEYL	I*4	Minimum elevation that falls within a flood level
ITOT	I*4	Total number of minimal cut sets identified

NBE is the number of unique basic events.

NDEP is the number of flood levels.

Table E.42 PRINTS Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
KEYDSC(NDEP,10)	R*8	Description of each flood level
KEY(NDEP)	I*4	Level keys
PNAM(NBE)	R*8	Temporary array for printing basic event elevations
PLEV(NBE)	I*4	Temporary array for printing basic event elevations
ISUM	I*4	Number of basic events submerged between flood levels
TKEY	I*4	Maximum elevation that falls within a flood level
TKEYL	I*4	Minimum elevation that falls within a flood level

NBE is the number of unique basic events.

NDEP is the number of flood levels.

subroutines and is called by subroutines DOIT and PARTAL. Table E.43 describes important variables in PRSET.

E.2.42 PRUNOF (*, *, *, TREE, TREEL, IGTYP, IBE)

Subroutine PRUNOF deletes unflooded basic events from the fault tree. In pruning the fault tree, PRUNOF deletes basic events and appropriate AND gates in the fault tree. If an unflooded basic event is input to an OR gate, PRUNOF only deletes the basic event. If it is input to an AND gate, the AND gate and all connected AND gates encountered traversing up the tree until an OR gate is reached are deleted. The appropriate portions of the fault tree are "pruned away" by altering the pointers in the threaded pseudo-binary tree image.

PRUNOF calls no other subroutines and is called by TRAVRS. The first three terms in the call sequence refer to RETURN statements. Table E.44 describes the RETURN statement alternatives and other important variables used in PRUNOF.

E.2.43 RESET (IBE, TREEL, NREP, TREEX, FLAG)

Subroutine RESET sets all occurrences of a given basic event to false. RESET calls no other subroutines and is called by DOIT. Table E.45 lists important variables used in the subroutine.

E.2.44 SEARCH (WORD, NAM, N, NXG, NXE)

Subroutine SEARCH searches the NAM array for a specific name (basic event or gate). If found, SEARCH returns the index of the element. Otherwise, a zero is returned.

SEARCH calls no subroutines and is called by BUILD. Table E.46 describes important variables used in this subroutine.

E.2.45 SETIT (TREE, TREEL, IGTYP)

Subroutine SETIT sets the value of the TOP event according to the values of the fault tree gates. The fault tree gates are set by traversing the fault tree from the TOP event in post order form. Post order form is left branch, right branch, root.[5] Figure E.9 is a flowchart of subroutine SETIT.

SETIT begins by traversing from the TOP down the left branch of the fault tree from son to son. When the lowest level son (a basic event) in a branch is reached, the state of its father (a gate) is set according to the state of the son and type of gate. The brother of this son is analyzed next. If it is a gate, SETIT finds its lowest

Table E.43 PRSET Variables

Variable	Word Type	Description
NAM(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations
TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic events in the fault tree
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
PNAM(NBE)	R*8	Temporary array for printing basic event names
PLEV(NBE)	I*4	Temporary array for printing basic event elevations

NBE is the number of unique basic events.

MAXREP is the maximum number of times any basic event appears in the fault tree.

NDIM is the number of unique basic events plus the number of unique gates.

Table E.44 PRUNOF Variables

Variable	Word Type	Description
*		Alternate Return 1: PRUNOF uses this RETURN if: 1) the basic event is flooded, 2) the basic event has already been pruned, or 3) the subroutine reaches normal completion
*		Alternate Return 2: PRUNOF uses this RETURN if the basic event pruned has no brothers
*		Alternate Return 3: PRUNOF uses this RETURN if the TOP event is "pruned away"
TREE(NDIM,4)	I*2	Threaded pseudo-binary fault tree image
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
IGTYP(NBE)	I*2	Gate type
IBE	I*4	Basic event being examined
NODE	I*4	Current event being examined in the pseudo-binary tree image
DAD	I*4	Father of current event being examined
IDAD	I*4	DAD's gate/basic event index number
NDAD	I*4	NDAD = IDAD - NBE
FSON	I*4	First input to DAD
BRO	I*4	Next input to DAD
IPOP	I*4	Father of BRO

NBE is the number of unique basic events.

NDIM is the number of unique basic events plus the number of unique gates.

Table E.45 RESET Variables

Variable	Word Type	Description
IBE	I*4	Basic event index
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
NREP(NBE)	I*2	Number of times each basic event is repeated
TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic events in the fault tree
FLAG	L*4	Flag that identifies house events that have been set true. Once set true, house events are not reset.
IREP	I*4	Number of times a basic event is repeated
HERE	I*4	Specific location of a basic event in TREEX
UNDEF	L*1	Variable that indicates if a basic event has been resolved. If the basic event is resolved, it is the value of the basic event (true or false).

NBE is the number of unique basic events.

NDIM is the number of unique basic events plus the number of unique gates.

Table E.46 SEARCH Variables

Variable	Word Type	Description
WORD	R*8	Name to find
NAM(NG+NBE)	R*8	Gate/basic event names
N	I*4	Index number of the WORD found
NXG	I*4	Number of gate names currently in NAM
NXE	I*4	Number of basic event names currently in NAM

NG is the number of unique gates.

NBE is the number of unique basic events.

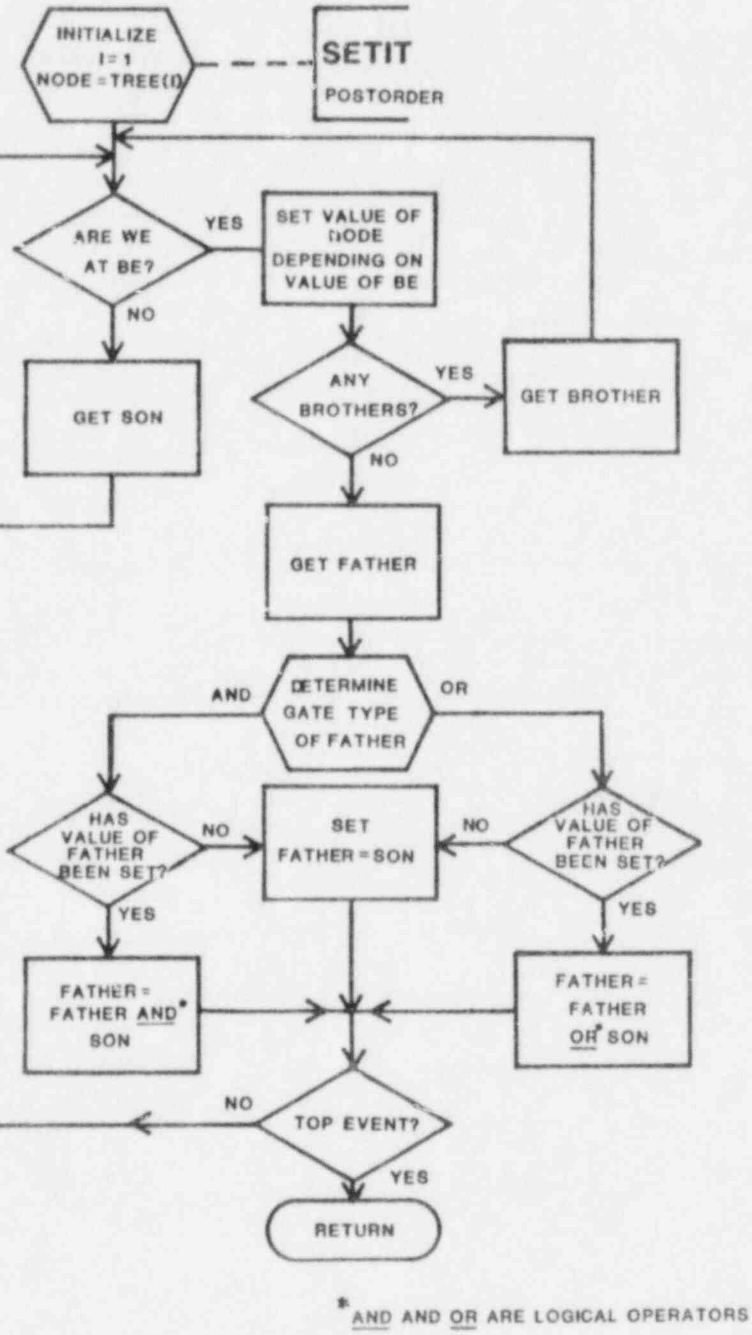

Figure E.9 Subroutine SETIT Flowchart

level son and sets the appropriate gates. If the brother is a basic event, SETIT resets the father. Brothers are analyzed until there are no more fathers and sons. At this point, the brother of the very first gate set is analyzed and the procedure repeats.

Subroutine SETIT calls no other subroutines and is called by DOIT, DOPATH, FIND1S, OK, and PREPIT. Table E.47 describes important variables in SETIT.

E.2.46 SETPTH (ISET, FLAG, IPATH, IIPATH, TREEL, NREP, TREEX)

Subroutine SETPTH sets all the members of a given flood protection set to true or false. SETPTH calls no other subroutines and is called by DOPATH. Table E.48 describes important variables used in this subroutine.

E.2.47 SETRUE (IBE, TREEL, NREP, TREEX)

Subroutine SETRUE sets all occurrences of a given basic event to true. SETRUE calls no other subroutines and it is called by DOIT, FIND1S, PARTAL, and PREPIT. Table E.49 describes important variables used in this subroutine.

E.2.48 SORTP (NPATH, IPATH, IIPATH, IWORK, IIWORK)

Subroutine SORTP sorts the flood protection sets in order of decreasing size. As SORTP sorts the flood protection sets in the work array (IIWORK), it copies them into the path set array (IPATH). SORTP calls no other subroutines and it is called by GENPTH, GENP2, and GENP3. Table E.50 lists important variables in this subroutine.

E.2.49 TOOBIG (MICS, IMIC, NSET)

Subroutine TOOBIG deletes minimal cut sets that are larger than ORDER. TOOBIG calls subroutine CONDNS and is called by subroutine FATRAM. Table E.51 describes important variables used in this subroutine.

E.2.50 TRAVRS (*, TREE, TREEL)

Subroutine TRAVRS traverses the fault tree in postorder form for the purpose of pruning the fault tree (see E.2.41 PRUNOF). TRAVRS begins tracing the fault tree with the left branch from the TOP event to the lowest son in the branch. TRAVRS then calls subroutine PRUNOF to determine if this son is to be removed. After pruning is complete for this son, the branch of the son's brother is traversed to the

Table E.47 SETIT Variables

Variable	Word Type	Description
TREE(NDIM,4)	I*2	Threaded pseudo-binary fault tree image
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
IGTYP(NG)	I*2	Gate type
NODE	I*4	Entity (gate or basic event) in question
DAD	I*4	Father of NODE
IDAD	I*4	Gate index number of DAD

NG is the number of unique gates.

NDIM is the number of unique gates plus the number of unique basic events.

Table E.48 SETPTH Variables

Variable	Word Type	Description
ISET	I*4	Index number of the flood protection set to set
FLAG	L*4	Value to set flood protection set members (true or false)
IPATH(PROW)	I*2	Number of basic events in each flood protection set
IIPATH(PROW,PCOL)	I*2	Flood protection sets
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
NREP(NBE)	I*2	Number of times each basic event is repeated in the fault tree
TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic events in the fault tree

For definitions of dimension variables see Table E.5.

Table E.49 SETRUE Variables

Variable	Word Type	Description
IBE	I*4	Index number of basic event to set
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
NREP(NBE)	I*2	Number of times each basic event is repeated in the fault tree
TREEX(NBE,MAXREP)	I*2	Basic event indices which locate the basic events in the fault tree

NBE is the number of unique basic events.

NDIM is the number of unique basic events plus the number of unique gates.

Table E.50 SORTP Variables

Variable	Word Type	Description
NPATH	I*4	Number of flood protection sets
IPATH(PROW)	I*2	Size of each flood protection set
IIPATH(PROW,PCOL)	I*2	Flood protection sets
IWORK(PROW)	I*2	Size of each flood protection set in the work array
IIWORK(PROW,PCOL)	I*2	Flood protection set work space
MIN	I*4	Smallest size flood protection set
MAX	I*4	Largest size flood protection set

PROW is the estimated number of BIPS.

PCOL is the order of the longest possible BIPS.

Table E.51 TOOBIG Variables

Variable	Word Type	Description
MICS(CROW,CCOL)	I*2	Minimal cut sets
IMIC(CROW)	I*2	Size of each minimal cut set
NSET	I*4	Number of minimal cut sets
COUNT	I*4	Number of minimal cut sets deleted

CROW is the estimated number of MICS.

CCOL is the order of the longest possible MICS.

lowest son. PRUNOF is again called. This process continues until there are no more sons and brothers to prune. At this point, TRAVRS goes to the brother of the first analyzed son's father and repeats this procedure. Figure E.10 is a simplified flowchart of TRAVRS and Figure E.11 is a graphic representation of this procedure.

Subroutine TRAVRS calls subroutine PRUNOF. It is called by subroutines DOIT, DOPATH, and PARTAL. An alternate return is used when the TOP has been pruned away. Table E.52 describes the important variables in TRAVRS.

E.2.51 XREFI (INDEX, GATE)

Subroutine XREFI prints a cross-reference table of the gate/basic event names in the gate array and the indices in the INDEX array used for building the threaded pseudo-binary image of the fault tree. XREFI calls no other subroutines and is called by subroutine BUILD. Table E.53 lists important variables used in XREFI.

E.2.52 XREFN (NAME, LVTN)

Subroutine XREFN prints a cross-reference table of the basic event indices, names and elevations. XREFN calls no other subroutines and it is called by subroutine BUILD. Table E.54 lists important variables used in this subroutine.

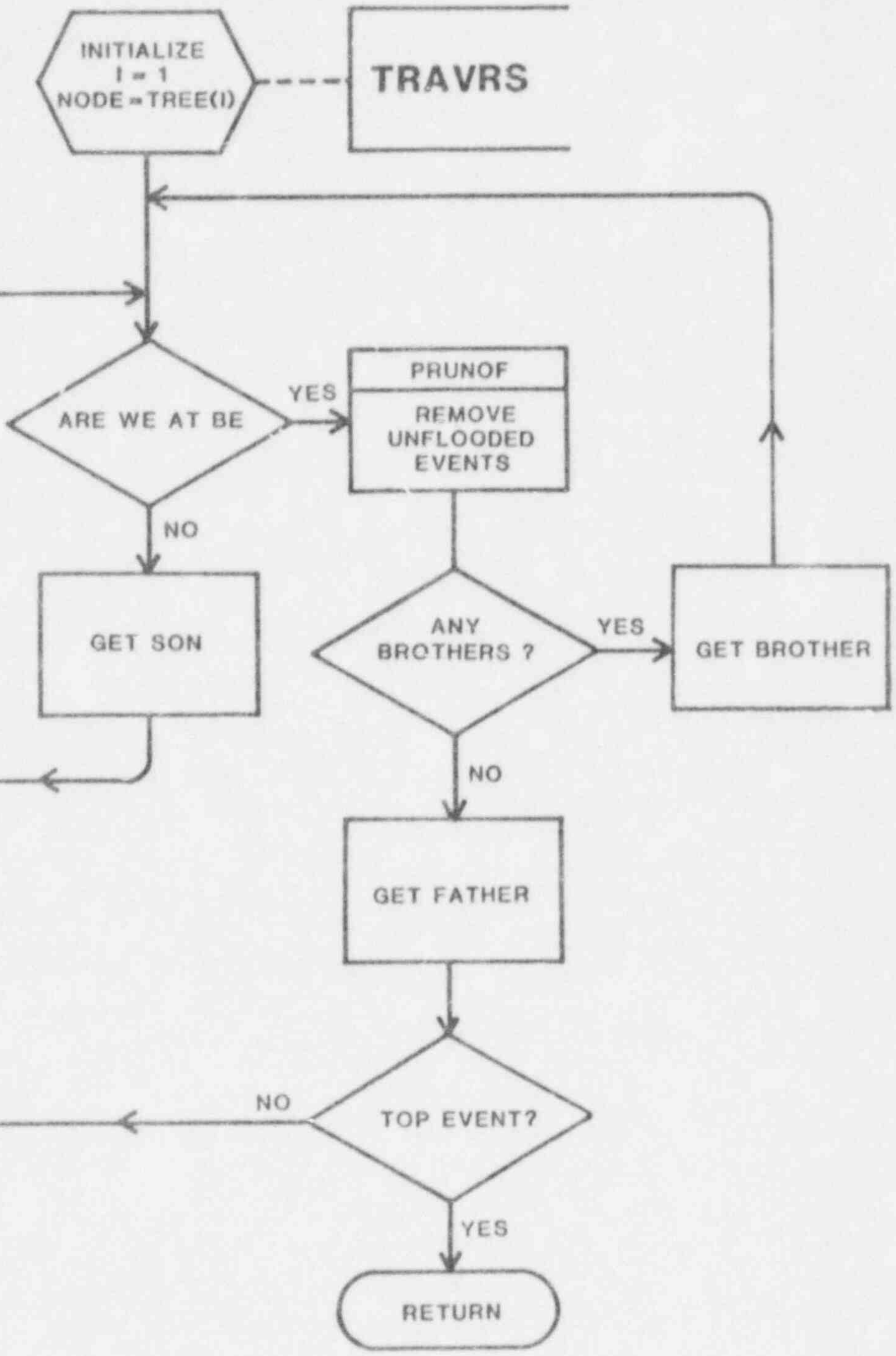

Figure E.10 Subroutine TRAVRS Flowchart

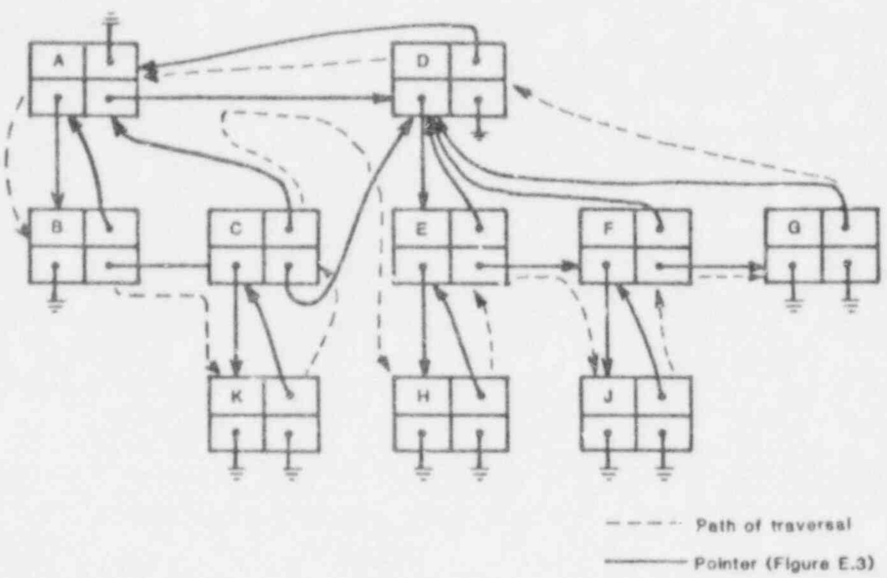

Figure E.11 Example Traversal of a Pseudo-binary Fault Tree Image

Table E.52 TRAVRS Variables

Variable	Word Type	Description
*		Alternate Return: TRAVRS uses an alternate RETURN statement if the TOP event is "pruned away".
TREE(NDIM,4)	I*2	Threaded pseudo-binary fault tree image
TREEL(NDIM)	L*1	Pseudo-binary tree image state array
NODE	I*4	Current element in the pseudo-binary tree being analyzed

NDIM is the number of unique basic events plus the number of unique gates.

Table E.53 XREFI Variables

Variable	Word Type	Description
INDEX(NG,MAXIN)	I*2	Index number of the fault tree gates or basic events
GATE(NG,MAXIN)	R*8	Gate/basic event names
MAX	I*4	Maximum number of inputs to the gate being listed

NG is the number of unique gates.

MAXIN is the maximum number of inputs for any gate.

Table E.54 XREFN Variables

Variable	Word Type	Description
NAME(NBE)	R*8	Basic event names
LVTN(NBE)	I*4	Basic event elevations

NBE is the number of unique basic events.

APPENDIX F

PARTIALLY FLOODED MINIMAL CUT SET EXAMPLE

F. PARTIALLY FLOODED MINIMAL CUT SET EXAMPLE

This appendix describes the output obtained from NOAH when partial common cause candidates are determined. Figure F.1 lists the input data supplied to NOAH for this example. Output from NOAH includes the following:

1. a listing of the input data deck (Figure F.2),

2. input and calculated parameters (Figure F.3),

3. starting addresses of NOAH internal arrays (Figure F.4),

4. results of an input data check (Figure F.5),

5. a cross-reference table of internal codes and external names and elevations used by NOAH (Figure F.6), and

6. a listing of partially flooded minimal cut sets for flood level MAXD and each flood level above MAXD (Figure F.7).

Several of the starting addresses of the NOAH internal arrays will change during the determination of the partially flooded minimal cut sets. NOAH lists these internal array starting addresses each time they change.

NOAH provides the following information for each partially flooded minimal cut set (Figure F.7):

1. a minimal cut set index number,

2. the flood level of the highest basic event in the minimal cut set that is below MAXD,

3. the names of the basic events in the minimal cut set, and

4. the elevation of each basic event in the minimal cut set.

```
- --+----1\----+----2\----+----3\----+----4\----+----5\----+----6\
SECOND SAMPLE PROBLEM - PARTIALLY FLOODED CUT SETS
* CONTROL
 $NOPT DEPTH=0, NDEP=9, MAXD=5, FIND=T, ORDER=4, MAXIN=3,
       SEEPTH=T, DOPT=T, DEEPER=T $END
END
* KEY
   10   20   30   40   50   60   70   80   90
FIRST FLOOD LEVEL
SECOND FLOOD LEVEL
THIRD FLOOD LEVEL
FOURTH FLOOD LEVEL
FIFTH FLOOD LEVEL
SIXTH FLOOD LEVEL
SEVENTH FLOOD LEVEL
EIGHTH FLOOD LEVEL
NINTH FLOOD LEVEL
END
* TREE
TOP       AND  2 0 GATE2     GATE3
GATE2     OR   1 2 GATE4     A2 OFF    B2 CLOSD
GATE4     AND  2 0 GATE6     GATE7
GATE6     OR   1 2 GATE10    E2 OFF    H2 CLOSD
GATE7     OR   1 2 GATE10    D2 OFF    G2 CLOSD
GATE10    OR   1 1 GATE12    I2 CLOSD
GATE3     OR   1 2 GATE5     A1 OFF    B1 CLOSD
GATE5     AND  2 0 GATE8     GATE9
GATE8     OR   1 2 GATE11    E1 OFF    H1 CLOSD
GATE9     OR   1 2 GATE11    D1 OFF    G1 CLOSD
GATE11    OR   1 1 GATE12    I1 CLOSD
GATE12    AND  2 0 GATE13    GATE14
GATE13    OR   0 2 K EMPTY   J CLOSED
GATE14    OR   0 2 L CLOSED  M EMPTY
END
* ELEVATION
A2 OFF              60
B2 CLOSD            60
A1 OFF              50
B1 CLOSD            10
E2 OFF              50
H2 CLOSD            30
D2 OFF              80
G2 CLOSD            70
I2 CLOSD            90
E1 OFF              60
H1 CLOSD            40
K EMPTY             20
J CLOSED            70
L CLOSED            90
M EMPTY             90
I1 CLOSD            80
D1 OFF              60
G1 CLOSD            30
END
STOP
```

Figure F.1 NOAH Example Problem 2: Input Data

```
                                                                              VERSION - 2 - DEC, 1981
                         * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                          - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                         *                                                          *
                          -   NOAH - - A PROGRAM FOR QUALITATIVE FLOOD ANALYSIS   -
                         *                                                          *
                          - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                         * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

>>  LISTING OF INPUT DATA DECK  <<

SECOND SAMPLE PROBLEM - PARTIALLY FLOODED CUT SETS
* CONTROL
&NOPT
DEPTH=      0,NDEP=      9,MAXD=      5,DOPT=T,FIND=T,SEEPTH=T,ORDER=      4,MAXIN=      3,DEEPER=T,CFD=F,
TIMPT=      0,NTPT=      0,DSRCH=F,DUMP=F,ECHO=T,TRACE=F
&END
END
* KEY
   10   20   30   40   50   60   70   80   90
FIRST FLOOD LEVEL
SECOND FLOOD LEVEL
THIRD FLOOD LEVEL
FOURTH FLOOD LEVEL
FIFTH FLOOD LEVEL
SIXTH FLOOD LEVEL
SEVENTH FLOOD LEVEL
EIGHTH FLOOD LEVEL
NINTH FLOOD LEVEL
END
* TREE
TOP       AND  2 0 GATE2    GATE3
GATE2     OR   1 2 GATE4    A2 OFF   B2 CLOSD
GATE4     AND  2 0 GATE6    GATE7
GATE6     OR   1 2 GATE10   E2 OFF   H2 CLOSD
GATE7     OR   1 2 GATE10   D2 OFF   G2 CLOSD
GATE10    OR   1 1 GATE12   I2 CLOSD
GATE3     OR   1 2 GATE5    A1 OFF   B1 CLOSD
GATE5     AND  2 0 GATE8    GATE9
GATE8     OR   1 2 GATE11   E1 OFF   H1 CLOSD
GATE9     OR   1 2 GATE11   D1 OFF   G1 CLOSD
GATE11    OR   1 1 GATE12   I1 CLOSD
GATE12    AND  2 0 GATE13   GATE14
GATE13    OR   0 2 K EMPTY  J CLOSED
GATE14    OR   0 2 L CLOSED M EMPTY
END
```

Figure F.2 NOAH Example Problem 2: Input Data Listing

```
>>  LISTING OF INPUT DATA DECK  <<               SECOND SAMPLE PROBLEM - PARTIALLY FLOODED CUT SETS

* ELEVATION
A2 OFF                 60
B2 CLOSD               60
A1 OFF                 50
B1 CLOSD               10
E2 OFF                 50
H2 CLOSD               30
D2 OFF                 80
G2 CLOSD               70
I2 CLOSD               90
E1 OFF                 60
H1 CLOSD               40
K EMPTY                20
J CLOSED               70
L CLOSED               90
M EMPTY                90
I1 CLOSD               80
D1 OFF                 60
G1 CLOSD               30
END
```

Figure F.2 Continued

----- NOAH PARAMETERS FOR THIS RUN ----- VERSION - 2 - DEC, 1981

NUMBER OF UNIQUE GATES, NG ----------------------	14
NUMBER OF UNIQUE BASIC EVENTS, NBE ---------------	18
TOTAL NUMBER OF GATES, NGATE --------------------	25
TOTAL NUMBER OF BASIC EVENTS, NCOMP --------------	32
MAXIMUM TIMES AN EVENT APPEARS, MAXREP -----------	4
IT MAY BE POSSIBLE TO FIND CUT SETS OF UP TO ORDER	8
LEVEL TO FLOOD, DEPTH ---------------------------	0
LEVELS IN THE PLANT, NDEP -----------------------	9
MAXIMUM LEVEL TO FLOOD, MAXD --------------------	5
LEVEL KEY OPTION, DOPT --------------------------	T
CONTINUOUS FLOOD, DEEPER ------------------------	T
DO FLOOD SCREENING ONLY, CFD --------------------	F
FIND SETS AFTER MAXD, FIND ----------------------	T
PRINT FLOOD PROTECTION SETS, SEEPTH --------------	T
ORDER SETS TO FIND, ORDER -----------------------	4
MAXIMUM INPUTS TO ANY GATE, MAXIN ----------------	3
SEARCH FOR BE/COMPONENT NAME, DSRCH --------------	F
INTERFACE/TIME-POINT PARAMETER, TIMPT ------------	0
NUMBER TIMEPOINTS FOR INTERFACE, NTPT ------------	0
DUMP INTERMEDIATE INFORMATION, DUMP --------------	F
PRINT CONTENTS OF INPUT ARRAYS, ECHO -------------	T
PRINT PROGRAM FLOW TRACE, TRACE ------------------	F

Figure F.3 NOAH Example Problem 2: Input and Calculated Parameters

```
INPUT CHECK: LEVEL KEYS AND DESCRIPTIONS                                    VERSION - 2 - DEC, 1981

KEY(  1)= '   10'     DESCRIPTION 'FIRST FLOOD LEVEL                              '
KEY(  2)= '   20'     DESCRIPTION 'SECOND FLOOD LEVEL                             '
KEY(  3)= '   30'     DESCRIPTION 'THIRD FLOOD LEVEL                              '
KEY(  4)= '   40'     DESCRIPTION 'FOURTH FLOOD LEVEL                             '
KEY(  5)= '   50'     DESCRIPTION 'FIFTH FLOOD LEVEL                              '
KEY(  6)= '   60'     DESCRIPTION 'SIXTH FLOOD LEVEL                              '
KEY(  7)= '   70'     DESCRIPTION 'SEVENTH FLOOD LEVEL                            '
KEY(  8)= '   80'     DESCRIPTION 'EIGHTH FLOOD LEVEL                             '
KEY(  9)= '   90'     DESCRIPTION 'NINTH FLOOD LEVEL                              '


INPUT CHECK: GATES AND THEIR INPUTS                                         VERSION - 2 - DEC, 1981

NAME       TYPE          INPUTS -> -> ->
TOP         AND     2 0  GATE2     GATE3
GATE2       OR      1 2  GATE4     A2 OFF    B2 CLOSD
GATE4       AND     2 0  GATE6     GATE7
GATE6       OR      1 2  GATE10    E2 OFF    H2 CLOSD
GATE7       OR      1 2  GATE10    D2 OFF    G2 CLOSD
GATE10      OR      1 1  GATE12    I2 CLOSD
GATE3       OR      1 2  GATE5     A1 OFF    B1 CLOSD
GATE5       AND     2 0  GATE8     GATE9
GATE8       OR      1 2  GATE11    E1 OFF    H1 CLOSD
GATE9       OR      1 2  GATE11    D1 OFF    G1 CLOSD
GATE11      OR      1 1  GATE12    I1 CLOSD
GATE12      AND     2 0  GATE13    GATE14
GATE13      OR      0 2  K EMPTY   J CLOSED
GATE14      OR      0 2  L CLOSED  M EMPTY
```

Figure F.4 NOAH Example Problem 2: Input Check of Data

INPUT CHECK: COMPONENT ELEVATIONS VERSION - 2 - DEC, 1981

NAME	ELEVATION
A2 OFF	60
B2 CLOSD	60
A1 OFF	50
B1 CLOSD	10
E2 OFF	50
H2 CLOSD	30
D2 OFF	80
G2 CLOSD	70
I2 CLOSD	90
E1 OFF	60
H1 CLOSD	40
K EMPTY	20
J CLOSED	70
L CLOSED	90
M EMPTY	90
I1 CLOSD	80
D1 OFF	60
G1 CLOSD	30

NO ERRORS DISCOVERED IN ROUTINE INPUT

NO ERRORS DISCOVERED IN ROUTINE CHEKIT

Figure F.4 Continued

VERSION - 2 - DEC, 1981

CROSS REFERENCE FOR INTERNAL CODES AND EXTERNAL NAMES AND ELEVATIONS

INDEX	NAME	ELEVATION
1	A2 OFF	60
2	B2 CLOSD	60
3	A1 OFF	50
4	B1 CLOSD	10
5	E2 OFF	50
6	H2 CLOSD	30
7	D2 OFF	80
8	G2 CLOSD	70
9	E1 OFF	60
10	H1 CLOSD	40
11	D1 OFF	60
12	G1 CLOSD	30
13	I2 CLOSD	90
14	I1 CLOSD	80
15	K EMPTY	20
16	J CLOSED	70
17	L CLOSED	90
18	M EMPTY	90

Figure F.5 NOAH Example Problem 2: Cross Reference of Internal Codes, External Names and Elevation

RELATIVE STARTING ADDRESSES FOR ARRAYS VERSION - 2 - DEC, 1981

IW(1)=	50	IW(8)=	285	IW(15)=	445	IW(22)=	508	IW(29)=	6859
IW(2)=	107	IW(9)=	300	IW(16)=	450	IW(23)=	526	IW(30)=	6891
IW(3)=	150	IW(10)=	304	IW(17)=	455	IW(24)=	526	IW(31)=	7339
IW(4)=	236	IW(11)=	336	IW(18)=	473	IW(25)=	526	IW(32)=	7371
IW(5)=	244	IW(12)=	345	IW(19)=	482	IW(26)=	1177	IW(33)=	7819
IW(6)=	252	IW(13)=	435	IW(20)=	485	IW(27)=	6379	IW(34)=	8470
IW(7)=	267	IW(14)=	440	IW(21)=	490	IW(28)=	6411	IW(35)=	776

RELATIVE STARTING ADDRESSES FOR ARRAYS VERSION - 2 - DEC, 1981

IW(1)=	50	IW(8)=	285	IW(15)=	445	IW(22)=	508	IW(29)=	7030
IW(2)=	107	IW(9)=	300	IW(16)=	450	IW(23)=	526	IW(30)=	6891
IW(3)=	150	IW(10)=	304	IW(17)=	455	IW(24)=	526	IW(31)=	6388
IW(4)=	236	IW(11)=	336	IW(18)=	473	IW(25)=	526	IW(32)=	6420
IW(5)=	244	IW(12)=	345	IW(19)=	482	IW(26)=	1177	IW(33)=	6868
IW(6)=	252	IW(13)=	435	IW(20)=	485	IW(27)=	6379	IW(34)=	6900
IW(7)=	267	IW(14)=	440	IW(21)=	490	IW(28)=	6411	IW(35)=	7348

Figure F.6 NOAH Example Problem 2: Internal Array Starting Addresses

```
*******************************************************************************************
*******************************************************************************************
*******************************************************************************************
*                                                                                         *
*       PARTIALLY FLOODED MINIMAL CUT SET ANALYSIS      PARTIALLY FLOODED MINIMAL CUT SET ANALYSIS      *
*                                                                                         *
*******************************************************************************************
*******************************************************************************************
*******************************************************************************************


       SECOND SAMPLE PROBLEM - PARTIALLY FLOODED CUT SETS                  VERSION - 2 - DEC, 1981

PARTIALLY FLOODED MINIMAL CUT SET ANALYSIS AT LEVEL    5
KEY IS '  50'            DESCRIPTION='FIFTH FLOOD LEVEL

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                              7 BASIC EVENTS ARE FLOODED AT THIS LEVEL

A1 OFF      B1 CLOSD    E2 OFF      H2 CLOSD    H1 CLOSD    G1 CLOSD    K EMPTY

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                    41 MINIMAL CUT SETS ARE PARTIALLY FLOODED BY A FLOOD TO LEVEL   5

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

MINIMAL CUT SETS PARTIALLY FLOODED AT LEVEL   5


(    1) - 2- K EMPTY    20    L CLOSED   90
(    2) - 5- A2 OFF     60    A1 OFF     50
(    3) - 1- A2 OFF     60    B1 CLOSD   10
(    4) - 5- B2 CLOSD   60    A1 OFF     50
(    5) - 1- B2 CLOSD   60    B1 CLOSD   10
(    6) - 5- I2 CLOSD   90    A1 OFF     50
(    7) - 1- I2 CLOSD   90    B1 CLOSD   10
(    8) - 2- K EMPTY    20    M EMPTY    90
```

Figure F.7 NOAH Example Problem 2: Partially Flooded Minimal Cut Set Results

(	9)	-	3-	A2 OFF	60	E1 OFF	60	G1 CLOSD	30		
(	10)	-	4-	A2 OFF	60	H1 CLOSD	40	D1 OFF	60		
(	11)	-	4-	A2 OFF	60	H1 CLOSD	40	G1 CLOSD	30		
(	12)	-	3-	B2 CLOSD	60	E1 OFF	60	G1 CLOSD	30		
(	13)	-	4-	B2 CLOSD	60	H1 CLOSD	40	D1 OFF	60		
(	14)	-	4-	B2 CLOSD	60	H1 CLOSD	40	G1 CLOSD	30		
(	15)	-	5-	E2 OFF	50	A1 OFF	50	D2 OFF	80		
(	16)	-	5-	E2 OFF	50	A1 OFF	50	G2 CLOSD	70		
(	17)	-	5-	H2 CLOSD	30	A1 OFF	50	D2 OFF	80		
(	18)	-	5-	H2 CLOSD	30	A1 OFF	50	G2 CLOSD	70		
(	19)	-	5-	E2 OFF	50	B1 CLOSD	10	D2 OFF	80		
(	20)	-	5-	E2 OFF	50	B1 CLOSD	10	G2 CLOSD	70		
(	21)	-	3-	H2 CLOSD	30	B1 CLOSD	10	D2 OFF	80		
(	22)	-	3-	H2 CLOSD	30	B1 CLOSD	10	G2 CLOSD	70		
(	23)	-	3-	I2 CLOSD	90	E1 OFF	60	G1 CLOSD	30		
(	24)	-	4-	I2 CLOSD	90	H1 CLOSD	40	D1 OFF	60		
(	25)	-	4-	I2 CLOSD	90	H1 CLOSD	40	G1 CLOSD	30		
(	26)	-	5-	E2 OFF	50	I1 CLOSD	80	D2 OFF	80		
(	27)	-	5-	E2 OFF	50	I1 CLOSD	80	G2 CLOSD	70		
(	28)	-	3-	H2 CLOSD	30	I1 CLOSD	80	D2 OFF	80		
(	29)	-	3-	H2 CLOSD	30	I1 CLOSD	80	G2 CLOSD	70		
(	30)	-	5-	E2 OFF	50	E1 OFF	60	D2 OFF	80	G1 CLOSD	30
(	31)	-	5-	E2 OFF	50	E1 OFF	60	G2 CLOSD	70	G1 CLOSD	30
(	32)	-	5-	E2 OFF	50	H1 CLOSD	40	D2 OFF	80	D1 OFF	60
(	33)	-	5-	E2 OFF	50	H1 CLOSD	40	D2 OFF	80	G1 CLOSD	30
(	34)	-	5-	E2 OFF	50	H1 CLOSD	40	G2 CLOSD	70	D1 OFF	60
(	35)	-	5-	E2 OFF	50	H1 CLOSD	40	G2 CLOSD	70	G1 CLOSD	30
(	36)	-	3-	H2 CLOSD	30	E1 OFF	60	D2 OFF	80	G1 CLOSD	30
(	37)	-	3-	H2 CLOSD	30	E1 OFF	60	G2 CLOSD	70	G1 CLOSD	30
(	38)	-	4-	H2 CLOSD	30	H1 CLOSD	40	D2 OFF	80	D1 OFF	60
(	39)	-	4-	H2 CLOSD	30	H1 CLOSD	40	D2 OFF	80	G1 CLOSD	30
(	40)	-	4-	H2 CLOSD	30	H1 CLOSD	40	G2 CLOSD	70	D1 OFF	60
(	41)	-	4-	H2 CLOSD	30	H1 CLOSD	40	G2 CLOSD	70	G1 CLOSD	30

Figure F.7 Continued

```
*****************************************************************************************
*****************************************************************************************
*****************************************************************************************
*                                                                                       *
*   FLOOD PROTECTION SET ANALYSIS      FLOOD PROTECTION SET ANALYSIS      FLOOD PROTECTION SET ANALYSIS   *
*                                                                                       *
*****************************************************************************************
*****************************************************************************************
*****************************************************************************************
```

```
        SECOND SAMPLE PROBLEM - PARTIALLY FLOODED CUT SETS                 VERSION - 2 - DEC, 1981

FLOOD PROTECTION SET ANALYSIS OF LEVEL   5
KEY IS '  50'            DESCRIPTION='FIFTH FLOOD LEVEL

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

              41 MINIMAL CUT SETS ARE PARTIALLY FLOODED BY A FLOOD TO LEVEL   5

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                 22 FLOOD PROTECTION SETS EXIST AT A FLOOD TO LEVEL   5

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

FLOOD PROTECTION SETS AT LEVEL   5
```

(1)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	D1 OFF	60	I1 CLOSD	80
(2)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	H1 CLOSD	40	I1 CLOSD	80
(3)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	E1 OFF	60	H1 CLOSD	40	I1 CLOSD	80
(4)	K EMPTY	20	A2 OFF	60	B2 CLOSD	60	E2 OFF	50	H2 CLOSD	30	I2 CLOSD	90
(5)	K EMPTY	20	A2 OFF	60	B2 CLOSD	60	D2 OFF	80	G2 CLOSD	70	I2 CLOSD	90

Figure F.7 Continued

(6)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	D1 OFF	60	I1 CLOSD	80
	M EMPTY	90										
(7)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	H1 CLOSD	40	I1 CLOSD	80
	M EMPTY	90										
(8)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	E1 OFF	60	H1 CLOSD	40	I1 CLOSD	80
	M EMPTY	90										
(9)	L CLOSED	90	A2 OFF	60	B2 CLOSD	60	E2 OFF	50	H2 CLOSD	30	I2 CLOSD	90
	M EMPTY	90										
(10)	L CLOSED	90	A2 OFF	60	B2 CLOSD	60	D2 OFF	80	G2 CLOSD	70	I2 CLOSD	90
	M EMPTY	90										
(11)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	D1 OFF	60	E2 OFF	50
	H2 CLOSD	30										
(12)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	H1 CLOSD	40	E2 OFF	50
	H2 CLOSD	30										
(13)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	E1 OFF	60	H1 CLOSD	40	E2 OFF	50
	H2 CLOSD	30										
(14)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	D1 OFF	60	D2 OFF	80
	G2 CLOSD	70										
(15)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	H1 CLOSD	40	D2 OFF	80
	G2 CLOSD	70										
(16)	K EMPTY	20	A1 OFF	50	B1 CLOSD	10	E1 OFF	60	H1 CLOSD	40	D2 OFF	80
	G2 CLOSD	70										
(17)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	D1 OFF	60	E2 OFF	50
	H2 CLOSD	30	M EMPTY	90								
(18)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	H1 CLOSD	40	E2 OFF	50
	H2 CLOSD	30	M EMPTY	90								
(19)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	E1 OFF	60	H1 CLOSD	40	E2 OFF	50
	H2 CLOSD	30	M EMPTY	90								
(20)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	D1 OFF	60	D2 OFF	80
	G2 CLOSD	70	M EMPTY	90								
(21)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	G1 CLOSD	30	H1 CLOSD	40	D2 OFF	80
	G2 CLOSD	70	M EMPTY	90								
(22)	L CLOSED	90	A1 OFF	50	B1 CLOSD	10	E1 OFF	60	H1 CLOSD	40	D2 OFF	80
	G2 CLOSD	70	M EMPTY	90								

- - - - - CONCLUSION OF OUTPUT FROM NOAH - - - - -

Figure F.7 Continued

APPENDIX G

NOAH ERROR MESSAGES

G. NOAH ERROR MESSAGES

This appendix lists the error messages written by the NOAH program when errors are detected in the input data. The error number and message are stated and reasons for the error message are described. Words shown in quotes are variables whose value would be printed in that location.

ERROR NUMBER 1: MISSING OR INVALID END CARD

An END card for an input group has been omitted or mispunched. An END card must be supplied after each input group.

ERROR NUMBER 2: MISSING OR INVALID * CONTROL

The * CONTROL input group is either missing or unrecognizable to the program. Supply a corrected * CONTROL input group.

ERROR NUMBER 3: MISSING OR INVALID STOP CARD

The STOP card at the end of the input deck is either missing or mispunched.

ERROR NUMBER 4: WORK ARRAY IS TOO SMALL. YOU HAVE "NUM1" WORDS AND YOU NEED "NUM2".

The existing array space (NUM1) for the program run is too small. The array space should be increased to NUM2 by changing the dimensions of the W array in COMMON/WORK/ and the value of MAX in the data statement in the MAIN routine.

ERROR NUMBER 5: THIS GATE HAS MORE THAN THE "NUM1" INPUTS SPECIFIED. NINPUT = "NUM2"

The gate has more inputs (NUM2) than allowed by the value of MAXIN (NUM1). Increase MAXIN (maximum value of 7) or restructure the inputs to more than one gate.

ERROR NUMBER 6: WORK ARRAY IS TOO SMALL. YOU HAVE "NUM1" WORDS AND YOU NEED AT LEAST "NUM2".

Refer to Error Number 4.

ERROR NUMBER 7: GATE "WORD1" HAS AN INVALID TYPE "WORD2".

The gate type (WORD2) specified for gate WORD1 is invalid. Only AND and OR gate types are permitted.

ERROR NUMBER 8: GATE "WORD1" IDENTIFIED BUT NOT INPUT ANYWHERE.

Gate WORD1 has been identified on a gate card (Input Group 3) but is not input to any other gate. Check the fault tree description in Input Group 3.

ERROR NUMBER 9: GATE "WORD1" WAS USED MORE THAN ONCE WITH DIFFERENT INPUTS.

Gate WORD1 has been input with more than one set of inputs in Input Group 3. A gate may have only one set of inputs. A card may be mispunched.

ERROR NUMBER 10: GATE CARD "WORD1" HAS BEEN INCLUDED TWICE.

More than one card describing gate WORD1 is included in the fault tree description.

ERROR NUMBER 11: CIRCULAR LOGIC IN THE TREE IS NOT ALLOWED.

Circular logic has been identified in the fault tree description.

ERROR NUMBER 12: GATE "WORD1" USED AS INPUT, BUT NOT IDENTIFIED ANYWHERE.

Gate WORD1 has been used as an input to another gate, but no gate card is supplied for gate WORD1. Each gate must be identified on a gate card.

ERROR NUMBER 13: BASIC EVENT "WORD1" ON GATE CARD "NUM1" HAS AN INVALID ELEVATION "NUM2".

Basic event WORD1 has been assigned an elevation that lies outside the allowable elevations specified in Input Group 2 (* KEY).

ERROR NUMBER 14: MISPUNCHED CARD >> "CARD"

The identified card has a character punched in a space that is required to be blank.

ERROR NUMBER 15: GATE "WORD1" HAS A BLANK INPUT.

Fewer gate and basic event names are supplied than indicated on the card for gate WORD1.

ERROR NUMBER 16: THE "CARD1", "CARD2", "WORD1" DOES NOT APPEAR IN * TREE.

A search has been requested (Input Group 6) for a basic event or a component that does not appear in the fault tree description (* TREE, Input Group 3).

ERROR NUMBER 17: BASIC EVENT "WORD1" ON GATE CARD "WORD2" NOT GIVEN AN ELEVATION.

No elevation is given in Input Group 4 for basic event WORD1. An elevation must be specified for each basic event.

ERROR NUMBER 18: BASIC EVENT "WORD1" NOT IN * TREE.

Basic event WORD1 has been identified in * ELEVATION (Input Group 4) but does not appear in * TREE (Input Group 3).

ERROR NUMBER 19: TREE BUILDING SPACE IS TOO SMALL!! INCREASE VALUE OF COUNTER NGNC AND DIMENSIONS IN "WORD1".

The work space is too small for this problem.
If WORD1=GETNBE,
Change NGNC=NG+NBE
to NGNC=NG*NBE in Subroutine GETNG.

If WORD1=BUILD,
Change IW(37)=IW(36)+(NG*NBE+3)/4
to IW(37)=IW(36)+(NG*NBE+1)/2 in Subroutine ALOCAT.

ERROR NUMBER 20: GATE "WORD1" AND GATE "WORD2" HAVE THE SAME INPUT.

Gates WORD1 and WORD2 have identical inputs. Check the inputs to the gates. If the inputs are correct, then WORD1 and WORD2 are identical, and one of the two gate cards must be removed. Also, substitute the gate which is retained for all other times the deleted gate appeared (i.e. as input to gates).

ERROR NUMBER 21: SEARCH REQUESTED, BUT SEARCH DATA NOT INCLUDED.

DSRCH=T has been coded in Input Group 1 (* CONTROL) and Input Group 6 (* SEARCH) has not been supplied. Supply * SEARCH or code DSRCH=F.

ERROR NUMBER 22: REQUESTED ANALYSIS FOR DEPTH "NUM1" BUT THERE ARE ONLY "NUM2" DEPTHS IN THE PLANT.

The input value of DEPTH (NUM1) is greater than NDEP (NUM2). DEPTH cannot exceed NDEP (Input Group 1).

ERROR NUMBER 23: INVALID CUT SET SIZE >> ORDER="NUM1"

Minimal cut sets of order NUM1 have been requested in Input Group 1 (variable ORDER). The maximum value of ORDER is 10.

ERROR NUMBER 24: INVALID CONTROL CARD >> "CARD"

This card has an asterisk in column 1 and a missing or unrecognizable Keyword.

ERROR NUMBER 25: MISSING OR INVALID * KEY

The * KEY input group is missing or unrecognizable to the program. Supply a corrected * KEY input group.

ERROR NUMBER 26: GATE "WORD1" ON GATE CARD "WORD2" HAS BEEN ASSIGNED AN ELEVATION.

Gate WORD1 has been assigned an elevation in Input Group 4. Gates do not require an elevation input.

ERROR NUMBER 27: MISSING, MISPLACED, OR INVALID * ELEVATION

The * ELEVATION input group is missing or unrecognizable to the program. Supply a corrected * ELEVATION input group.

ERROR NUMBER 28: EVENT "WORD1" HAS BEEN ASSIGNED 2 ELEVATIONS OLD="NUM1", NEW="NUM2".

Only one elevation is permitted for each event. NUM1 is the first elevation input, NUM2 is the second elevation input.

ERROR NUMBER 29: THERE IS NOT ENOUGH ROOM IN THE CUT SET ARRAY. ROWMAX="NUM1". PLEASE INCREASE IT.

More space is required in the cut set array. The value of ROWMAX in Subroutine BLOCK DATA should be increased.

ERROR NUMBER 30: BLANK INPUT CARD IN * ELEVATION.

A blank input card has been found in the * ELEVATION input group. Blank cards are not permitted in * ELEVATION.

ERROR NUMBER 31: * PROFILE INPUT, BUT TIMPT IS 0

The * PROFILE input group has been included, but the user specified TIMPT = 0 in Input Group 1. The * PROFILE input group is not required.

ERROR NUMBER 32: BASIC EVENT "WORD1" ON GATE CARD "NUM1" HAS BEEN GIVEN AN INVALID HOUSE DESIGNATION.

One of the two parameters on the input card for the house event is missing.

ERROR NUMBER 33: INVALID ELEVATION "NUM1" GIVEN IN * PROFILE

An elevation has been given in * PROFILE which lies outside the range specified in * KEY.

ERROR NUMBER 34: BASIC EVENT "WORD1" ON GATE CARD "WORD2" NOT GIVEN A LAMBDA

The basic event WORD1 has been given a negative LAMBDA (If a basic event has not been given a lambda, it is perceived as a lambda of 0.0).

ERROR NUMBER 35: BASIC EVENT "WORD1" ON GATE CARD "WORD2" NOT GIVEN A TAU

The basic event WORD1 has been given a negative TAU.

APPENDIX H

NOAH JOB CONTROL LANGUAGE

H. NOAH JOB CONTROL LANGUAGE

This appendix is a listing of the minimum Job Control Language requirements for executing the NOAH computer program.

```
//Jobcard

// EXEC FORTHCLG,FORTREG=320K,GOREG=320K

//FORT.SYSIN DD *

  [FORTRAN source]

//GO.FT05F001 DD DDNAME=SYSIN

//GO.FT06F001 DD SYSOUT=A

//GO.FT07F001 DD SYSOUT=B

//GO.FT14F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=80,BLKSIZE=4240,BUFNO=1)

//GO.FT15F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=110,BLKSIZE=4180,BUFNO=1)

//GO.FT16F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=110,BLKSIZE=4180,BUFNO=1)

//GO.FT17F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=110,BLKSIZE=4180,BUFNO=1)

//GO.FT21F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DISP=(NEW,DELETE),

// DCB=(RECFM=FB,LRECL=100,BLKSIZE=4200,BUFNO=1)

//*

//*      One copy of the DD statement is required for each order of
         cut set desired.  The maximum order of cut set found is
//*      specified by the variable ORDER in Input Group 1.  Order has
         a maximum value of 10.  FTnnF001 is used on each DD statement
//*      required, where, nn equals twenty plus the order of the cut
         set for which that statement is supplied (i.e., 21, 22...
//*      20 + ORDER).

//*

//GO.SYSIN DD *
```

APPENDIX I

NOAH INPUT SUMMARY

I. NOAH INPUT SUMMARY

This appendix is a quick reference guide to the input for the NOAH program.

Table I.1 NOAH Input Deck Layout

- TITLE CARD
  [COMMENT CARDS]

- * CONTROL
  [CONTROL PARAMETERS]
  END
  [COMMENT CARDS]

- * KEY
  [ANALYSIS LEVEL KEYWORDS AND KEYWORD DESCRIPTIONS]
  END
  [COMMENT CARDS]

- * TREE
  [FAULT TREE DESCRIPTION]
  END
  [COMMENT CARDS]

- * ELEVATION
  [BASIC EVENT ELEVATIONS AND FAILURE/REPAIR INFORMATION]
  END
  [COMMENT CARDS]

- * HOUSE
  [HOUSE EVENT INFORMATION]
  END
  [COMMENT CARDS]

- * SEARCH
  [BASIC EVENT SEARCH INFORMATION]
  END
  [COMMENT CARDS]

- * PROFILE
  [FLOOD PROFILE VERSUS TIME]
  END

- STOP CARD

Table I.2 Input Group 1, CONTROL

Format: NAMELIST

Variable Name	Default Value	Description
DEPTH	0	DEPTH = 0 signifies that more than one flood level increment will be analyzed.
		DEPTH = n, n≠0, signifies that only level increment n will be analyzed.
NDEP	0	Number of flood level increments in the flood description.
MAXD	NDEP	Highest flood level increment considered in the flood simulation.
DOPT	T	DOPT = T (true) signifies that the user will supply all the flood levels to be analyzed and their descriptions in Input Group 2.
		DOPT = F (false) signifies that the user will provide only the maximum flood height and NOAH will divide it into NDEP equal intervals.
FIND	T	FIND = T (true) signifies that partially flooded minimal cut sets will be found at MAXD if the critical flood level has not been found.
		FIND = F (false) signifies that the analysis will terminate at MAXD without finding partially flooded minimal cut sets.

Table I.2 Input Group 1, CONTROL (continued)

Format: NAMELIST

Variable Name	Default Value	Description
SEEPTH	F	SEEPTH = T (true) indicates that the flood protection sets are to be output in addition to the minimal cut sets. SEEPTH = F (false) indicates that the flood protection sets are not output.
ORDER	10	Maximum size of the minimal cut sets to be found.
MAXIN	7	Maximum number of inputs to any gate in the fault tree description.
DEEPER	T	DEEPER = T (true) signifies a progressive flood from the lowest increment to MAXD. DEEPER = F (false) signifies that each depth increment will be considered individually.
TIMPT	0	TIMPT = -1 indicates a linear flood profile must be supplied and NOAH output will be punched (Input Groups 4 and 7 required). TIMPT = 1 indicates a discretized flood profile must be supplied and NOAH output will be punched (Input Groups 4 and 7 required). TIMPT = 0 indicates Input Group 7 and the basic event failure/repair information is not required in Input Group 4. NOAH output is not punched.

Table I.2 Input Group 1, CONTROL (continued)

Format: NAMELIST

Variable Name	Default Value	Description
NTPT	0	Number of time points in the discretized flood profile description. Supplied only if TIMPT = 1.
DSRCH	F	DSRCH = F (false) signifies that no search is requested. DSRCH = T (true) signifies that a search is requested and Input Group 6 will be supplied.
DUMP	F	DUMP = T (true) indicates detailed diagnostic output for error debugging is printed by NOAH. DUMP = F (false) indicates detailed diagnostic output is not printed.
TRACE	F	TRACE = T (true) signifies NOAH will list the subroutines as they are called in performing a flood simulation. TRACE = F (false) signifies a program flow trace is not printed.
ECHO	T	ECHO = T (true) signifies the input arrays are printed as they are filled. ECHO = F (false) signifies the input arrays are not listed in the output.

Table I.2 Input Group 1, CONTROL (continued)

Format: NAMELIST

Variable Name	Default Value	Description
CFD	F	CFD = T (true) signifies that the critical flood level will be found without determining flooded minimal cut sets.

Table I.3 Input Group 2, KEY

Variable	Format	Card Columns	Description
KEY	16I5	1-5 6-10 • • • 76-80	Flood level increment keywords, ordered from lowest to highest, 16 per card permitted
KEYDSC	10A8	1-80	Flood level increment descriptions, ordered from lowest to highest, one description per card

Table I.4 Input Group 3, TREE

Format: A8, 1X, A3, 1X, I2, I2, 7(1X,A8)

Variable	Format	Card Column	Description
GATE (1,I)	A8	1-8	Gate name (any eight alphanumeric characters)
	1X	9	Blank
GATYP(I)	A3	10-12	Gate type ("AND" OR "OR")
	1X	13	Blank
NGI(I)	I2	14-15	Number of gate inputs
NCI(I)	I2	16-17	Number of basic event inputs
	1X	18	Blank
GATE(2,I)	A8	19-26	First input name
	1X	27	Blank
GATE(3,I)	A8	28-35	Second input name
•		36	Blank
•		•	•
•		•	•
•		•	•
GATE(8,I)	A8	73-80	Seventh input name

One card per fault tree gate.

Table I.5 Input Group 4, ELEVATION

Format: A8, 11X, I5, 5X, 4(F10.0)

Variable	Format	Card Column	Description
NAME	A8	1-8	Basic event name (8 character alphanumeric)
	11X	9-19	Blank
ELEV	I5	20-24	Basic event vulnerability elevation keyword
	5X	25-29	Blank
LMBDA1*	F10.0	30-39	Basic event unflooded failure rate
TAU1*	F10.0	40-49	Basic event unflooded mean down time
LMBDA2*	F10.0	50-59	Basic event flooded failure rate
TAU2*	F10.0	60-69	Basic event flooded mean down time

One card per basic event.

*Required only when TIMTT = 1 or -1.

Table I.6 Input Group 5, HOUSE

Format: A8, 2X, A4, 6X, A8

Variable	Format	Card Columns	Description
NAME	A8	1-8	House event name (8 character alphanumeric)
	2X	9-10	Blank
HOUSE	A4	11-14	House event state, ON or OFF
	6X	15-20	Blank
FLOOD	A8	21-28	FLOOD = FAILS if the house event changes state upon submersion FLOOD = blank otherwise

One card per house event.

Table I.7 Input Group 6, SEARCH

Format: A8, 2X, A8

Variable	Format	Card Columns	Description
BENAM	A8	1-8	Basic event name (8 character alphanumeric)
	2X	9-10	Blank
COMPNT	A8	11-18	Component name, i.e., a portion of a basic event name with blanks for missing characters

Only one search request, BENAM or COMPNT is permitted per card. One card per search request.

Table I.8 Input Group 7, PROFILE

Format: I10, F10.0*
or 2F10.0**

Variable	Format	Card Columns	Description
LEVEL*	I5	1-10	Flood level
TIME*	F10.0	11-20	Time point at which LEVEL is reached
SLOPE**	F10.0	1-10	Rate at which the flood level is rising
NTCEPT**	F10.0	11-20	Initial flood level

*Supplied when TIMPT = 1 in Input Group 1.
**Supplied when TIMPT = -1 in Input Group 1.

NUREG/CR-2677
ORNL/TM-8313
Category Distribution RG

INTERNAL DISTRIBUTION

1- 2. L. S. Abbott
3. D. E. Bartine
4. W. B. Cottrell
5. T. E. Cole
6. J. Drago
7-10. G. F. Flanagan
11. P. M. Haas
12. H. Inhaber
13. H. E. Knee
14. F. C. Maienschein
15. A. L. Lotts
16. C. R. Richmond
17. D. B. Trauger
18. V. R. R. Uppuluri
19. A. Zucker
20. P. W. Dickson, Jr. (Consultant)
21. H. J. C. Kouts (Consultant)
22. W. B. Loewenstein (Consultant)
23. R. Wilson (Consultant)
24. Central Research Library
25. Y-12 Document Reference Section
26. Lab Records - ORNL RC
27. NSIC

EXTERNAL DISTRIBUTION

28-29. D. P. Wagner, JBF Associates, Inc., 1630 Downtown West Boulevard, Knoxville, TN. 37919
30. D. F. Montague, JBF Associates, Inc., 1630 Downtown West Boulevard, Knoxville, TN. 37919
31. J. J. Rooney, JBF Associates, Inc., 1630 Downtown West Boulevard, Knoxville, TN. 37919
32. J. B. Fussell, JBF Associates, Inc., 1630 Downtown West Boulevard, Knoxville, TN. 37919
33. L. S. Baker, JBF Associates, Inc., 1630 Downtown West Boulevard, Knoxville, TN. 37919
34. D. M. Rasmuson, EG&G-Idaho, Inc., P. O. Box 1625, Idaho Falls, ID. 83415
35. K. N. Fleming, Pickard, Lowe & Garrick, Inc., 17840 Skypark Boulevard, Irvine, CA. 92714
36. W. E. Vesely, Battelle Columbus Laboratories, 505 King Avenue, Columbus, OH. 43201
37. A. R. Buhl, Technology for Energy Corporation, 10770 Dutchtown Road, Knoxville, TN. 37922
38. N. D. Cox, EG&G-Idaho, Inc., P. O. Box 1625, Idaho Falls, ID. 83415
39. Boyer B. Chu, Electric Power Research Institute, 3412 Hillview Avenue, Palo Alto, CA. 94303
40. Bob Christie, Tennessee Valley Authority, Nuclear Engineering Branch, W70C726, 400 Commerce Avenue, Knoxville, TN. 37902
41. Joe Johnson, Tennessee Valley Authority, Power Operations Training Center, P. O. Box 2000, Daisy, TN. 37319
42. Zack Pate, Institute of Nuclear Power Operations, 1820 Water Place, Atlanta, GA. 30339

NUREG/CR-2677
ORNL/TM-8313
Category Distribution RG

43. K. A. Solomon, Rand Corporation, 1700 Main Street, Santa Monica, Ca. 90406
44. A. K. Bhattacharyya, Kuljian Corporation, 3700 Science Center, Philadelphia, PA. 19104
45. G. Apostolakis, Chemical, Nuclear & Thermal Engineering, 5532 Boelter Hall, School of Engineering & Applied Science, University of California-Los Angeles, Los Angeles, CA. 90024
46. M. Stamatelatos, Torrey Pines Technology, P. O. Box 81608, San Diego, CA. 92138
47. Office of Assistant Manager of Reactor Research & Technology, U. S. DOE, ORO, Oak Ridge, TN. 37830

48- 49. U. S. DOE Technical Information Center, Oak Ridge, TN. 37830

50. P. K. Niyogi, Methodology Development Section, Risk Methodology & Data Branch, Division of Risk Analysis, Office of Nuclear Regulatory Research, U. S. NRC, Washington, D. C. 20555
51. S. R. Sturges, Methodology Development Section, Risk Methodology & Data Branch, Division of Risk Analysis, Office of Nuclear Regulatory Research, U. S. NRC, Washington, D. C. 20555

52-321. Given distribution as shown in Category RG, Systems & Reliability Reports.

*U.S. GOVERNMENT PRINTING OFFICE 1982-546-067/66

120555078877   1 ANRG
US NRC
ADM DIV OF TIDC
POLICY & PUBLICATIONS MGT BR
PDR NUREG COPY
LA 212
WASHINGTON   DC 20555