BOEING

APOLLO TIE
WASHINGTON, D. C.

DOCUMENT NO. D2-117018-1 Rev. A

TITLE APOLLO LOGIC DIAGRAM ANALYSIS GUIDELINE

MODEL NO. APOLLO CONTRACT NO. NASw-1650

D. R. PLEAS
SYSTEM SAFETY ANALYSIS

REVISION A

JUNE 17, 1968

R. S. YOSEPH
MANAGER - SYSTEM SAFETY
AND PROGRAM ASSURANCE

ISSUE NO. ISSUED TO

DISTRIBUTION LIST

R.S. Yoseph	5-2320	RS-09	(1 copy)
E.J. Boyle	5-2322	RS-09	(1 copy)
E.G. Newberg	5-2324	RS-09	(1 copy)
D.R. Pleas	5-2321	RS-09	(1 copy)
C.F. Slumpff	5-2323	RS-98	(1 copy)
D.B. Jacobs	5-2200	RS-02	(1 copy)
N. Johnson	5-8231	FT-05	(2 copies)
F.W. Bushman	5-8230	FT-68	(1 copy)
J.W. Davis	5-8232	FT-90	(2 copies)
H.D. Boring	5-2700	HR-11	(1 copy)
R.B. McMurdo	5-2660	HR-01	(2 copies)
W.B. Dalrymple	5-6700	AA-58	(2 copies)
R.E. Evans	5-6762	AA-73	(2 copies)
J.W. Griswold	2-5020	84-39	(1 copy)
N.E. Classon	2-5020	84-38	(1 copy)
H.D. Trettin	2-5020	84-38	(1 copy)
G.R. Herrold	2-5025	84-38	(1 copy)
J.M. Barker	5-2330	RS-11	(1 copy)
C.P. Martin	5-2100	RS-12	(1 copy)

APO-TIE Library File No. 1

ABSTRACT

This document provides a general set of guidelines to standardize logic diagram (fault tree) methodology for system safety analysis in the Apollo program. These guidelines cover both the qualitative and quantitative aspects.

KEY WORDS

System Safety

Logic Diagram

Event

Fault Duration Time

Dominant Path

Failure Data Development

Apollo

Logic Diagram Analysis

Qualitative Evaluation

Quantitative Evaluation

System Safety Analysis

Fault Tree Analysis

Monte Carlo Simulation

Importance Sampling

Lambda-Tau

FOREWORD

This guide has been prepared to establish uniformity in the application of logic diagram (fault tree) methodology where used in the Apollo program system safety analyses. Implementation of standardized terminology, symbolism and procedures as defined in this document is necessary to ensure compatibility in the logic diagram analyses being performed concurrently at the various Apollo geographical locations.

CONTENTS

ILLUSTRATIONS

1.0 INTRODUCTION

1.1 PURPOSE

The purpose of this document is to provide the basis for standardized Apollo system safety logic diagram analysis efforts.

1.2 SCOPE

This document establishes standards for Apollo logic diagram (fault tree) analysis methods, terminology and symbolism. Three basic analytical elements are defined:

1) System safety logic diagramming (Section 4);

2) System safety failure data development (Appendix A);

3) System safety logic diagram evaluation (Appendix B).

The document also describes the analysis flow process, logic diagram format requirements, two methods of drafting logic diagrams, and the interfaces between system safety engineering and other engineering disciplines.

2.0 SYSTEM SAFETY ANALYSIS FLOW

2.1 ANALYSIS ACTIVITY

In order to place Apollo System Safety efforts in the proper perspective, the following problem solving steps are considered essential as an underlying basis for a systems approach to safety. These steps will enable the risk of undesired (hazardous) events identified in the Apollo Program to be maintained at an acceptable level.

1) Identification of undesired events;

2) Structuring identified undesired events into a logic diagram;

3) Determination of fault inter-relationships;

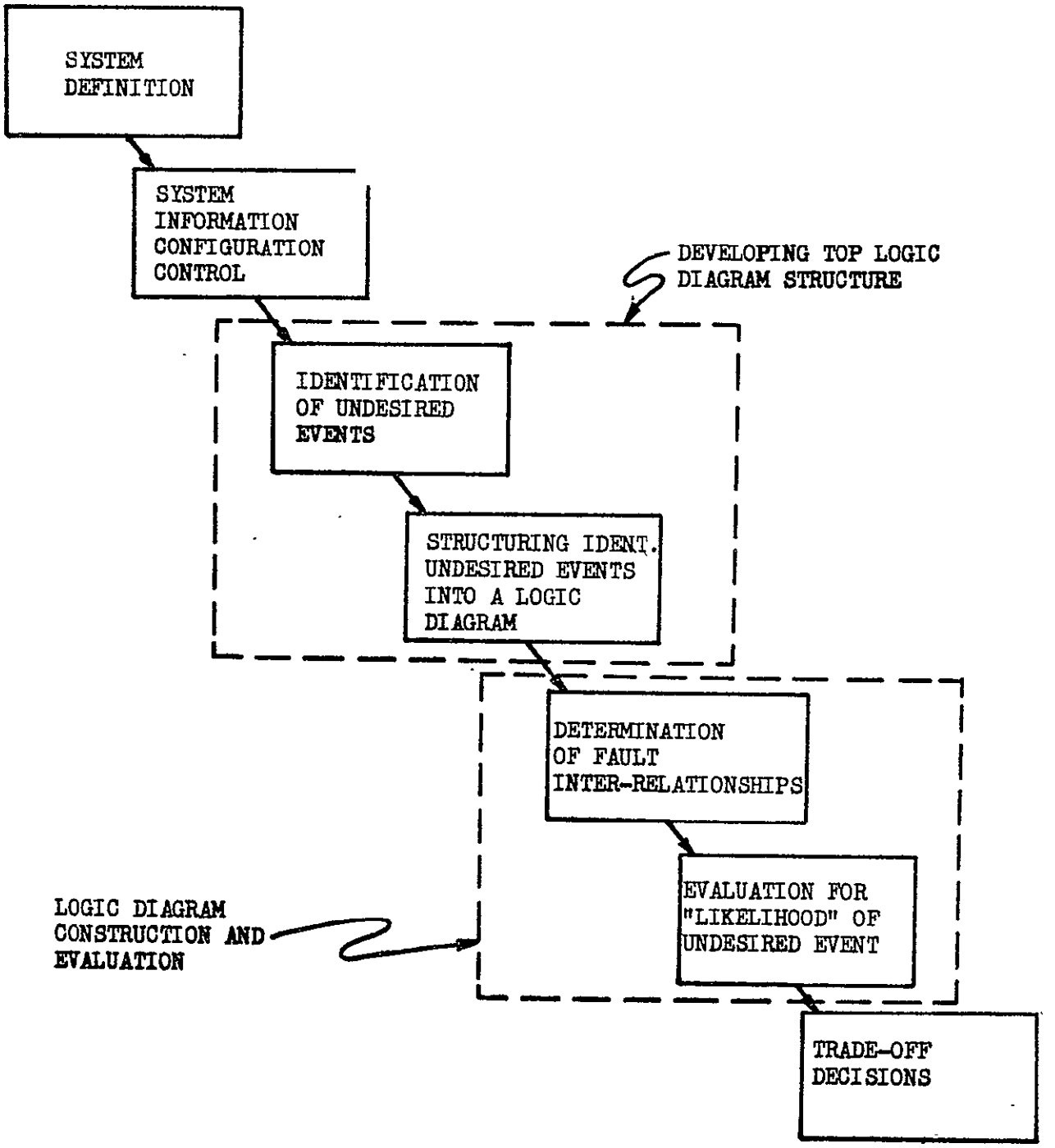
4) Evaluation for "likelihood" of identified undesired events;

5) Trade-off decisions and/or corrections.

As depicted in Figure 1, steps 1) and 2) above are necessary to develop what is commonly known as a "Top" logic diagram. The top logic diagram plays an essential part in performing a system safety logic diagram analysis. It is a starting guide which shows how and where the logic diagram is to be developed (or expanded) by further analysis activity. It organizes all of the system unique logic relationships into a pattern whereby the system hardware and software functions can be analyzed in an orderly and logical manner. This means that the top must be structured so that the end analysis is complete in satisfying what is defined by the top undesired event(s).

System unique logic relationship variables which must be carefully structured are things such as: 1) system operation modes, 2) mission phases and/or operations, 3) the degree of man/machine relationship in the system, 4) interrelationships of the Centers with the system functions, and 5) functional order of the system.

This list of relationship variables covers the top structure gross considerations, and indicates the types of activity involved. System unique logic relationship variables will vary with the different systems being analyzed, with the degree of difference depending upon the similarity between systems.



ANALYSIS FLOW

FIGURE 1

As already stated, the top logic diagram is a starting "guide" for a complete system logic diagram analysis. This means that once the top is started it is not necessarily "cast in concrete", but is subject to change as analysis activity progresses. Experience has shown that as an analysis proceeds to completion, more system information and understanding is gained. As system information and understanding develop, modification to the top logic diagram is required to reflect this current knowledge.

Step 3) is the actual development of the logic diagram. This is the point where analysis activity starts from the top logic diagram structure and proceeds through the hardware level. This step of analysis is the foundation of a logic diagram analysis. The fault mode relationships, once correctly and completely structured, will usually never change - unless hardware design changes occur.

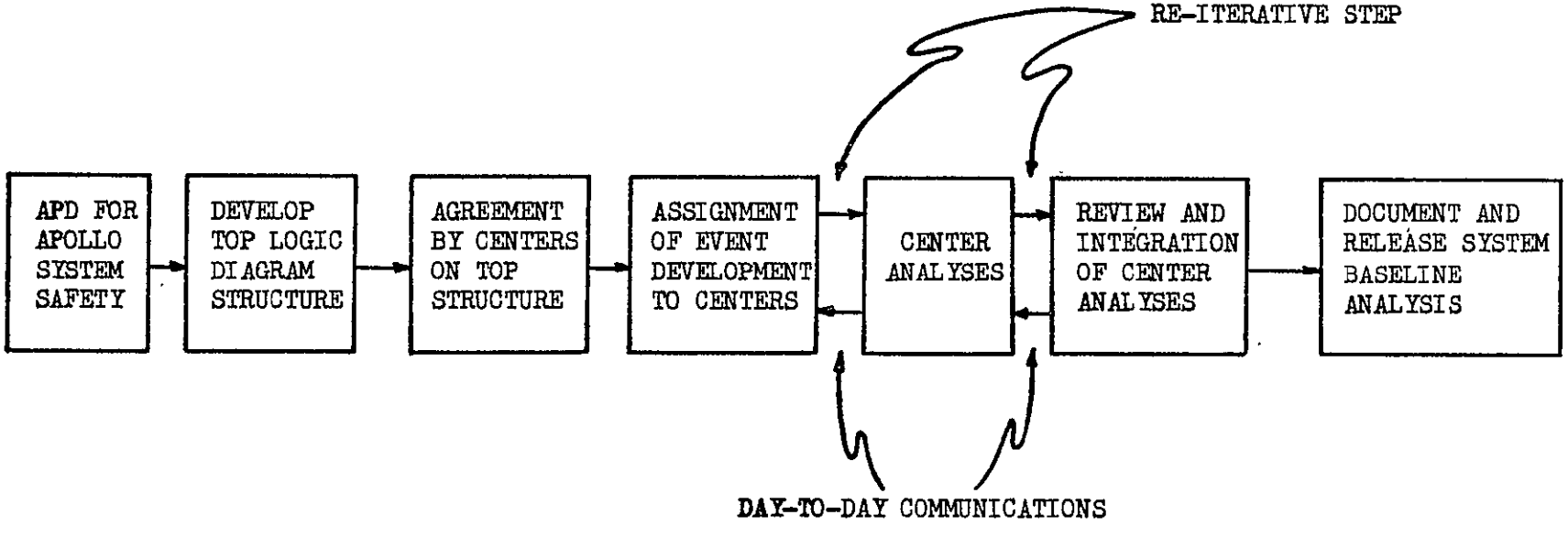
Step 4) is an evaluation of the completed logic diagram for the purpose of: 1) determining the likelihood of identified undesired events, and 2) determining the identity and ranking of "chains" of events and event relationships leading to the identified undesired event(s). Evaluation can be accomplished by rigorous mathematical processes (quantitative evaluation) or from intuitive (inductive) methods. However, the results obtained (quantitative/inductive) will only be as complete as the applied rigor. Useful results can be obtained from evaluations made during the course of development of the logic diagram analysis.

Step 5) If it is determined through the evaluation of the logic diagram (or as a result of other analyses) that corrective action is required, the logic diagram analysis itself is a valuable source of information for change decisions. Proposed corrections such as design changes, procedure changes, etc., can be evaluated in the context of the logic diagram to determine a relative measure of improvement.

In order to achieve a meaningful and useful analysis, two important points must be emphasized. First, the output of an analysis is only as valuable and reliable as the quality and quantity of effort and information going in to the analysis. Second, hardware and operating procedures configuration control must be maintained at all times to avoid erroneous conclusions being drawn from the analysis.

2.2 PROGRAM ACTIVITY

The flow of activity necessary for a complete system-integrated Apollo logic diagram analysis should follow a pattern as shown in Figure 2. This flow takes into consideration the steps required to perform an analysis, along with the



PROGRAM FLOW

FIGURE 2

difficult task of consolidating the Center analyses into one complete system/mission oriented analysis.

As shown in Figure 2, the first step in program development is the structuring of the top logic diagram. This is done primarily by the APO, with the aid of the Centers. After a suitable top has been structured and agreed upon by all involved, each of the Centers is assigned specified portions of the logic diagram for further development. While the Centers are conducting their analyses, the task of reviewing the output of each Center and combining the output into one complete systems analysis is performed by the APO. When the Apollo system analysis for system safety is complete, it will be documented, approved and released.

An important factor necessary in accomplishing a system-integrated analysis is the media of communications. Without an effective communications network, operable on a "day-to-day" basis between the Centers and the APO, loss of direction, efficiency and effectiveness will result.

2.3 APOLLO LOGIC DIAGRAM FORMAT REQUIREMENTS

To achieve a consistency of approach and to assure analysis completeness, the following requirements must be met by each MSF Center submitting logic diagram analyses.

1) Structuring must follow the rules and symbolism set forth in this document.

2) Each "diamond" event must have the following information/reason for analysis termination of the event:

   (a) Insignificant (with rationale)

   (b) Lack of system information

   (c) Identification of other analyses which satisfactorily analyze the failure modes and system effects for that event.

3) Development information sources must be identified by schematic, flow, time, mechanical, electrical, operation, maintenance drawing and/or document numbers. The revision date and/or number must be included for each source. This source information must be included as part of each submittal.

4) Each Center must utilize the logic diagram alphabetic code assignments made in the Apollo Logic Diagram Analysis Guidelines document.

5) Each submittal shall indicate the Apollo System configuration represented, i.e., AS-503, AS-504, in the "Model" block.

6) Revision codes will be included by each Center and will be based on the standard Boeing practice of assigning progressive alphabetic characters beginning with A.

7) Identify all components and subsystems by part number and part number source if part number source is different from logic diagram structure source.

3.0 SYSTEM SAFETY INTERFACES

3.1 CONFIGURATION MANAGEMENT INTERFACE

Configuration Management is the system engineering management which describes, identifies and controls the system/equipment configuration throughout the definition, development, production and change phases.

A primary function of System Safety is to develop a logic diagram analysis which reflects the system/equipment of an established and well defined baseline configuration. To perform this task System Safety must have available an established baseline configuration and detailed baseline system/equipment design and operation information. The responsibility of establishing the baseline configuration and identifying this configuration by documentation of detailed design and engineering data lies with Configuration Management.

Another function of System Safety is to analyze the effect of proposed changes on the baseline configuration with respect to the logic diagram analysis performed on the baseline configuration. Decisions as to the acceptability and cost-effectiveness of proposed changes, with regard to System Safety, can be made based on the mathematical model (logic diagram) of the system/equipment. Proposed System Safety design changes resulting from logic diagram analyses must be implemented through Configuration Management.

Demonstration that the system/equipment development is in accordance with the specification/requirements is accomplished through internal and contractually required design reviews and equipment inspections. System Safety is required to participate in the reviews and to show that the design meets the specification/requirements allocated for System Safety, and that, in general, no problem areas exist. Demonstration that baseline requirements and specifications are being met with respect to System Safety is accomplished through the medium of logic diagram analysis.

Verification and certification of all manufactured end items is the responsibility of Configuration Management, but is a direct task of Quality Assurance. The interface between System Safety and Quality Control is discussed in further detail in Section 3.3.

Data on the accumulated operating time/cycles and the number of failures of identified hardware items is essential to System Safety in order to identify potential "weak links" in a system and to quantitatively evaluate logic diagrams. Also, a record of approved changes with their scheduled

incorporation date into the hardware and the actual incorporation date is vital to System Safety. With this information, the status of the baseline configuration and the baseline logic diagram analyses can be kept current.

3.2 DESIGN ENGINEERING INTERFACE

The interface between System Safety and Design Engineering requires a "day-to-day" working relationship between members of each organization. The results of this close relationship are inherently beneficial to both organizations.

Design Engineering is a source of valuable information on the operating and functional characteristics of the system/equipment. The detailed and gross level information obtained during the initial design stage is invaluable to System Safety. In retrospect, System Safety can make known to designers the ideas and practices associated with designing an inherently safer system.

Knowledge of proposed changes to the system/equipment can be acquired during the conceptual and initial design change stage, and suggestions made to the designers to provide a safer and/or safety cost-effective change. System/equipment design changes which are needed for improved system safety can be discussed with the designers to select the most effective design alternative with respect to safety and system effectiveness.

Close work between the two organizations at an early stage will allow differences to be worked out before procedural formalities of the "organizational system" require a longer period of negotiation in order to yield a design acceptable to both Design Engineering and System Safety.

3.3 QUALITY ASSURANCE INTERFACE

The system significance of a particular event or part detail cannot be determined by study of the design alone. Therefore, predictive system safety analyses must be made from drawings, procedures and other documented instructions. Thus, the accuracy of such analyses and the conclusions derived from them are dependent on activities of quality technicians and inspectors in assuring that instructions are followed.

Quality requirements are determined and satisfied throughout all phases of contract performance, including preliminary and engineering design, development, fabrication, processing, assembly, inspection, test, checkout, packaging, shipping, storage, maintenance, field use, flight preparations, flight operations and flight analysis. The Quality Assurance program ensures that quality aspects are fully included in all

designs and are continuously maintained in the fabricated articles. Any change required to improve components, subsystem, or system performance without compromising quality, reliability or safety should be incorporated at the earliest practical point in development and fabrication. The program provides for the early and prompt detection of actual or potential deficiencies, system incompatibility, marginal quality, and trends or conditions which could result in unsatisfactory quality. Objective evidence of quality conformance, including records of inspection and test results, are made available to cognizant organizations. This information is a necessary part of system safety analyses in order to provide a high level of confidence in the logic diagram representation of system faults and confidence in the assignment of probabilities to the fault events.

3.4 TEST PLANNING AND REPORTING INTERFACE

Special tests are conducted on hardware end items for reliability data, qualification, quality assurance, and system hardware integration. From these tests considerable data is produced and made available which is necessary for logic diagram evaluation. Conversely, requirements for special tests to obtain data specifically needed by System Safety may result from logic diagram analyses.

System safety analyses conducted on proposed test plans may initiate special test procedures and corrective measures to existing test plans.

3.5 RELIABILITY INTERFACE

A function of Reliability is system/hardware analysis for failure data; such as failure modes, failure effects, mean time between failures, probabilities of failure and assessment of system failures on mission accomplishment. Much of this type of data is used by System Safety for both qualitative and quantitative system safety analysis. For example, existing and substantiated failure modes and effects data is an invaluable aid in the qualitative logic diagram analysis of a system. In a quantitative logic diagram evaluation, hardware failure rate data is a necessary item. In retrospect, the results of a system safety analysis may have a direct impact upon reliability; such as requiring further testing of certain hardware or improving the reliability of a particular system element. Therefore, it is essential that System Safety coordinate closely with Reliability for an exchange of data on a timely basis. Since reliability data is a prime source of information for system safety analyses, there is a significant degree of inherent compatibility. However, it should be noted that complete numerical parity should not be expected because reliability "numbers" normally refer to both primary and

secondary failures for particular failure modes. Thus, it is entirely possible for a system to have reliability which is the complement of one failure per 1000 operating hours and a probability of significant undesired event, i.e., accident, of one per 1,000,000 operating hours.

Conversely to System Safety use of reliability data, reliability considerations may be altered by System Safety analytical results. For example, it may be necessary to make a particular system element more reliable because of its safety significance even though the system reliability is already adequate.

3.6 MAINTAINABILITY AND HUMAN ENGINEERING INTERFACE

Maintainability is a design discipline that provides for ease, economy and safety in all maintenance functions and the use of maintenance equipment. Therefore, System Safety and Maintainability must work closely in order for System Safety to perform safety analyses on maintenance equipment and to certify the safety of maintenance equipment design and maintenance operations.

Human Engineering is involved in the designing of equipments for human use so that they can be used efficiently, accurately and safely. Human Engineering and System Safety must make use of, as a part of the safety analyses, human factor statistics (data) available as the result of human factor studies. The human/system interface is a necessary consideration in safety analyses.

3.7 HEALTH AND SAFETY INTERFACE

System Safety is concerned with test, assembly, checkout, maintenance and use of systems which provide a possibility of serious injury, loss of life, loss of equipment or significant equipment damage. In other words, concern with the above undesired events as a result of the existence of the system. Health and Safety is concerned with providing a safe working environment for employees. It is obvious that there will be some overlap between the two functions and in this case the more stringent standards of acceptability would apply.

System Safety can aid the Health and Safety activity by providing hardware oriented data and by identifying hazardous areas. Health and Safety can aid System Safety by providing information and data on human factors, toxic materials, anthropometric considerations and other specialized data related to the human working environment.

4.0 SYSTEM SAFETY LOGIC DIAGRAMMING

4.1 GENERAL

The system safety logic diagram is a logic oriented graphic representation of the parallel and series combinations of independent personnel or equipment subsystem and component failure and normal operating modes that can result in a specified undesired event. This representation can be quantified to provide a relative measure of the paths leading to these events.

The following sections discuss basic rules, definitions and methodology of the logic diagramming technique.

4.2 EVENT DESCRIPTION

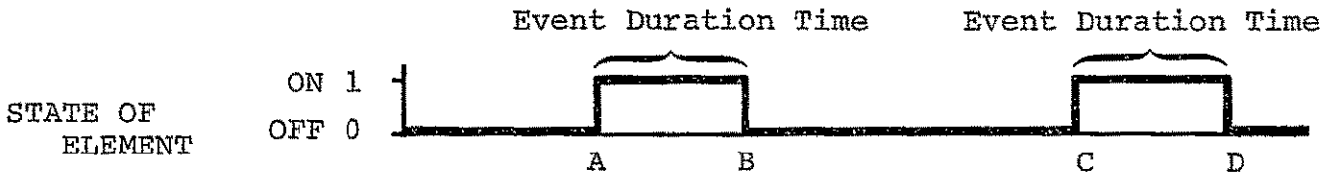
The term "event" denotes a dynamic change of state that occurs to a system element, where an element is inclusive of hardware, software, personnel and environment. If the change of state is such that the intended function of the particular element is not achieved, or an unintended function is achieved, the event is an abnormal system function or "fault event." If the change of state is such that the intended function occurs as planned (designed), the event is then a normal system function or "normal event." Thus, two types of events exist -- those which are not intended and those which are intended.

Fault events can be divided into two categories: basic events and gate events. Basic events are events whereby system elements (usually at the component level) go from an unfailed state to a failed state, and are related to a specific failure rate and fault duration time. These events are used only as inputs to a logic gate (never as outputs) and are therefore independent events. On a logic diagram, basic events are depicted by a circle and/or a diamond. A gate event is the event (or system failure) which results from the output of a logic gate. Since the gate event is dependent upon the input events and the type of logic gate function, it is therefore a dependent event. It must be noted that the gate event is not the logic gate itself, but the result of the logic gate function and the input events. The gate event is depicted by a rectangle above the logic gate. As a logic diagram development progresses, gate events on one level become inputs to gate events on the next higher level.

In the logic diagram analysis of a system the inherent modes of failure of system elements are delineated as primary, secondary and command. These failure modes are referred to as "primary events," "secondary events," and "command events" respectively, and are depicted on the logic diagram as the combination of basic events and/or gate events. In

other words, these events are generally identified at a gate event level, and depending on the level of analysis, are further developed until the event can be identified as a basic event.

In a logic diagram analysis, the dynamic change of state that occurs to a system element is defined as a binary type event. That is, a system element is always in one of two states, ON or OFF. The ON state (or 1) corresponds to a failed condition and the OFF state (or 0) corresponds to an unfailed condition. The example below illustrates the binary manner of a system element. The element operates normally (OFF state) until failure occurs (ON state). After the fault event occurs (dynamic change of state) the element remains failed (ON state) until repair of some sort has been effected. When repair is accomplished, the element returns to the unfailed state (OFF). By representing events and gates in a binary manner, logic diagrams can be analyzed by the rigorous techniques of Boolean algebra.

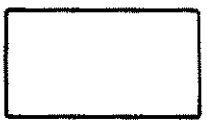


A - Time of 1st failure
B - Time 1st failure is repaired
C - Time of 2nd failure
D - Time 2nd failure is repaired

4.3 SYMBOLS

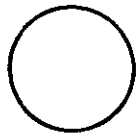
Rectangle

The rectangle identifies an event (gate event) that results from the combination of fault events through a logic gate. The rectangle is also used to describe a conditional input to a functional condition INHIBIT gate (described below).



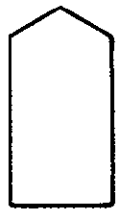
Circle

The circle describes a basic fault event that requires no further development. The frequency and mode of failure of items so identified is derived from empirical data. The rate of occurrence of such a primary event is normally the generic failure rate of the component for the particular failure mode.



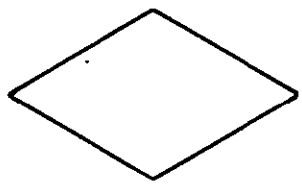
House

The house indicates an event that must occur (or is expected to occur) due to normal operating conditions in the system. The house does not indicate a fault event. An example is a phase change in a dynamic system, such as the landing, flight, and take-off phases of an aircraft.



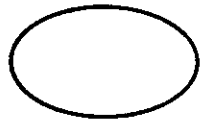
Diamond

The diamond describes a fault event that is considered basic in a given logic diagram. The possible causes of the event are not developed either because the event is of insufficient consequence or the necessary information for further development is unavailable. For Apollo T.I.E. system safety applications, it also can indicate non-development because an analysis already exists that is of satisfactory depth and breadth.



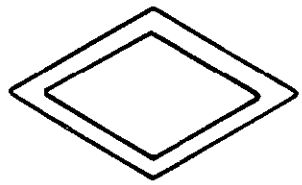
Oval

The oval is used to record the conditional input to a random condition INHIBIT gate. It defines the state of the system that permits a fault sequence to occur, and may be either normal to the system or result from failures. It is also used to indicate the necessary sequence of events required to pass through an "AND" or an "OR" gate function.



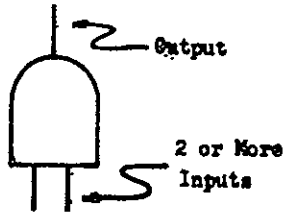
Double Diamond

The double diamond is used in the simplification of a logic diagram for numerical evaluation. The event described results from the causes that have been identified, but are not shown on a particular version of the logic diagram.

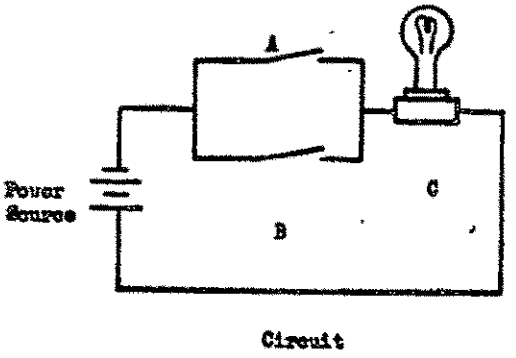


"AND" Gate

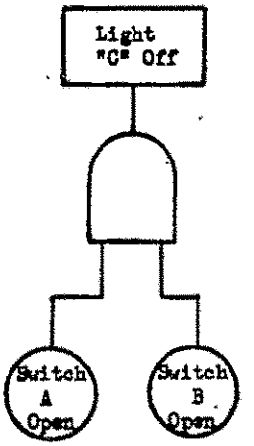
The "AND" gate describes the logical operation whereby the co-existence of all input events is required to produce the output event. The fault duration time of an "AND" gate is expressed in terms of the input fault duration times.



Example of "AND" Gate Usage:

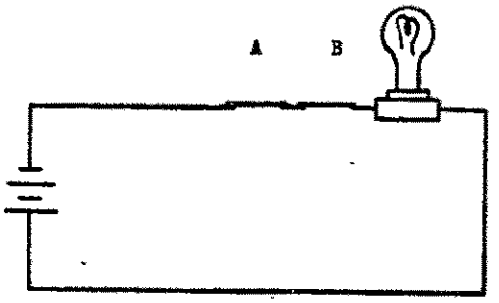


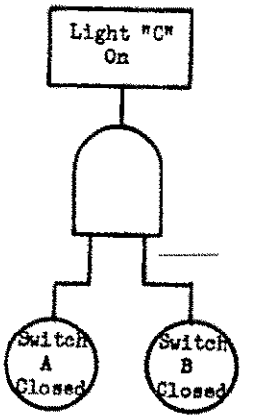
Circuit



LOGIC DIAGRAM

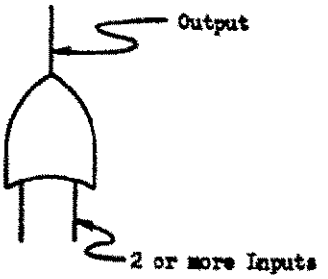
Another example of "AND" Gate Usage:



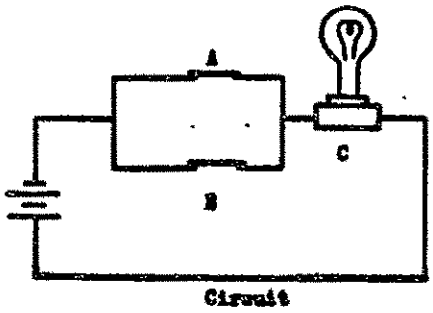


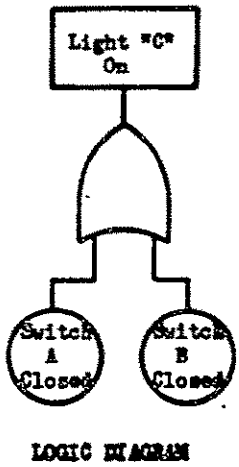
"OR" Gate

The "OR" gate defines the situation whereby the output event will exist if one or more of the input events exists. The fault duration time of an "OR" gate is expressed in terms of the input fault duration times.



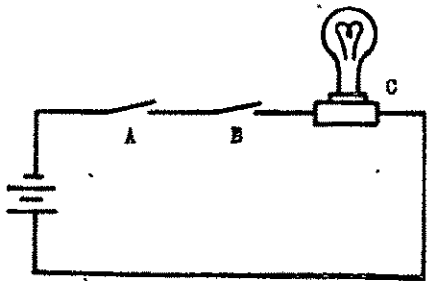
Example of "OR" Gate Usage:

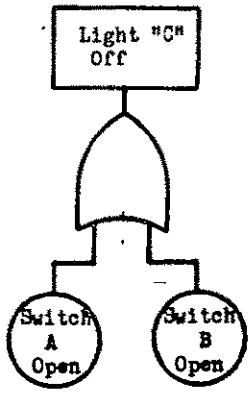




LOGIC DIAGRAM

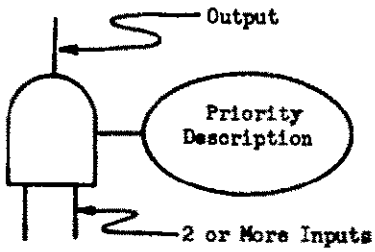
Another example of "OR" Gate Usage:





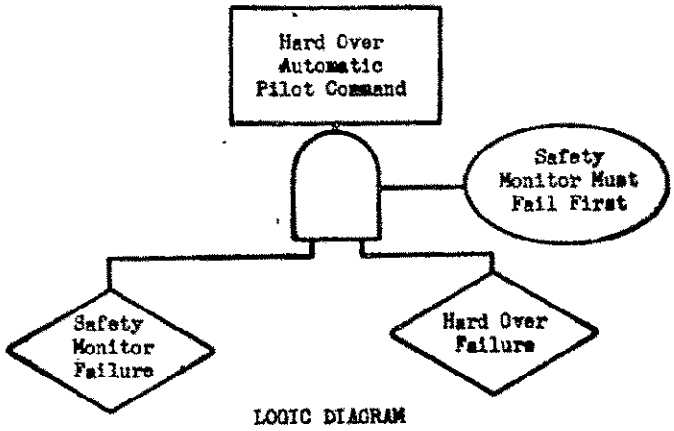
"PRIORITY AND" Gate

The "PRIORITY AND" gate performs the same logic function as the "AND" gate with the additional stipulation that sequence as well as co-existence is required.



Example of "PRIORITY AND" Gate Usage

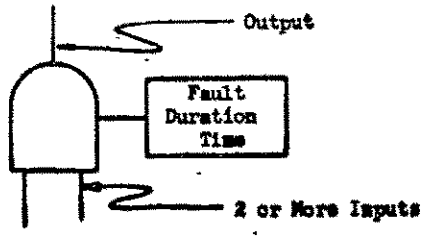
Assume: For an aircraft the safety monitor feature monitors the system response to automatically control and inhibit those control inputs which exceed system limits.



LOGIC DIAGRAM

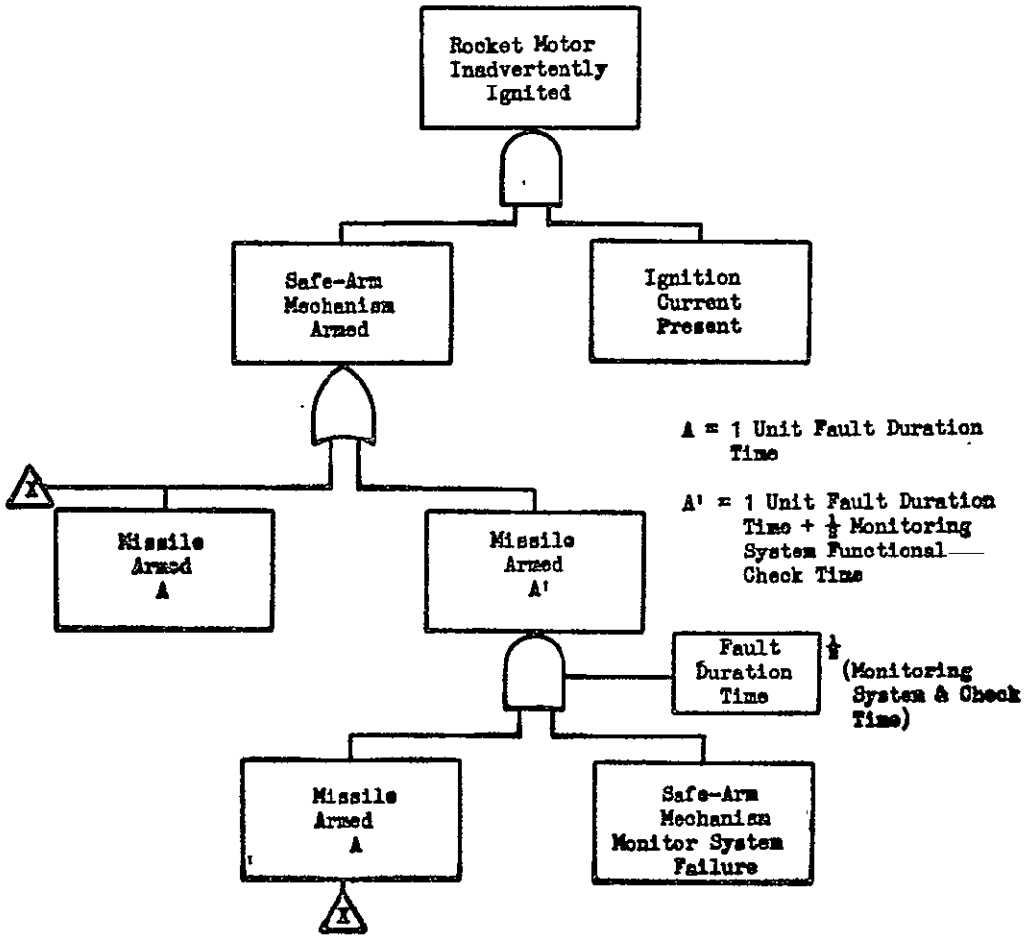
"CONSTANT FAULT DURATION AND" Gate

The "CONSTANT FAULT DURATION AND" gate symbolized describes the same logical function as the "AND" gate except that the fault duration time of the output event is not dependent upon the fault duration times of the inputs. The fault duration time of this gate is determined as a function of the system operation.



Example of "CONSTANT FAULT DURATION AND" Gate Usages

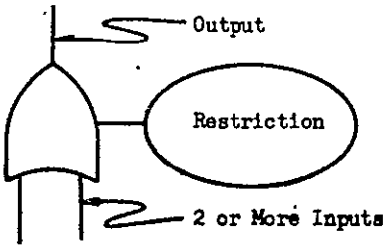
Consider the undesired event "Rocket Motor Inadvertently Ignited." Assume the "armed" results in a warning light prompting immediate repair action. If the "armed" event occurs and the warning system is working, the fault duration time is one unit. If the "armed" event occurs and the warning system has failed, the fault duration time is naturally longer, being dependent upon how often the monitoring system is functionally checked.



LOGIC DIAGRAM

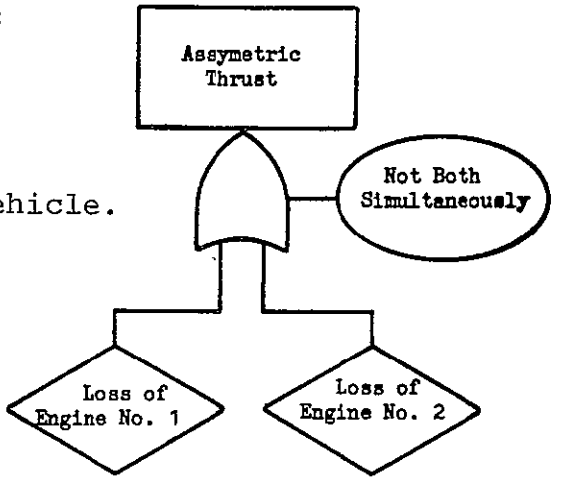
"EXCLUSIVE OR" Gate

The "EXCLUSIVE OR" gate functions as an OR gate with the restriction that specified inputs cannot co-exist. This gate will not respond to the co-existence of two or more specified input events.



Example of "EXCLUSIVE OR" Gate Usage:

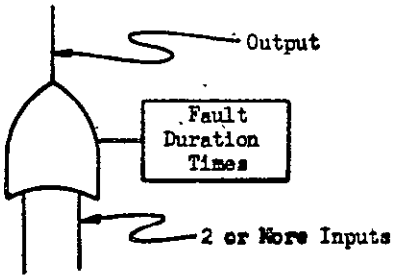
Assume: Twin, side mounted engine vehicle.



LOGIC DIAGRAM

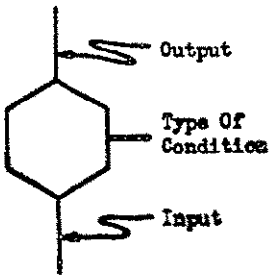
"CONSTANT FAULT DURATION OR" Gate

The "CONSTANT FAULT DURATION OR" gate performs the same function as the "OR" gate except that the fault duration time of the output event is not dependent upon the fault duration times of the inputs. The fault duration time of the output event is strictly dependent upon system operation variables, and must be determined from system information rather than in terms of the input event fault duration times.



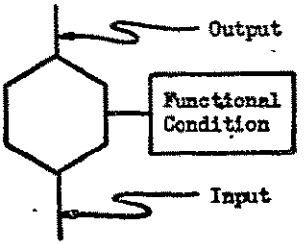
"INHIBIT" Gates

"INHIBIT" gates describe a causal relationship between one fault and another. The input event directly produces the output event if the indicated condition is satisfied. The conditional input defines a state of the system that permits the fault sequence to occur, and may be either normal to the system or result from failures. The conditional input is represented by an oval if it describes a specific failure mode and a rectangle if it describes a condition that may exist for the life of the system. The conditional input is further described on the following pages. The logical "INHIBIT" functions are symbolized in system safety logic diagrams as follows:

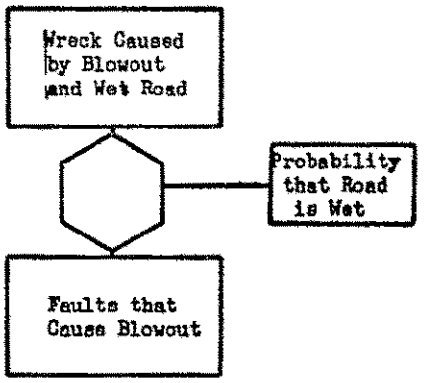


"FUNCTIONAL CONDITION INHIBIT" Gate

The "FUNCTIONAL CONDITION INHIBIT" gate provides a means for applying conditional probabilities to the fault sequences. If the input event occurs and the "condition" is satisfied, an output event will be generated. The duration time of the output event may be either the duration time of the fault input or may be separately generated.

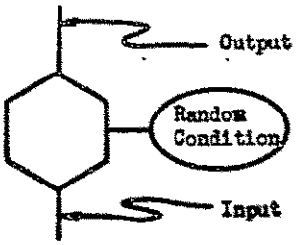


Example of "FUNCTIONAL CONDITION INHIBIT" Gate Usage:

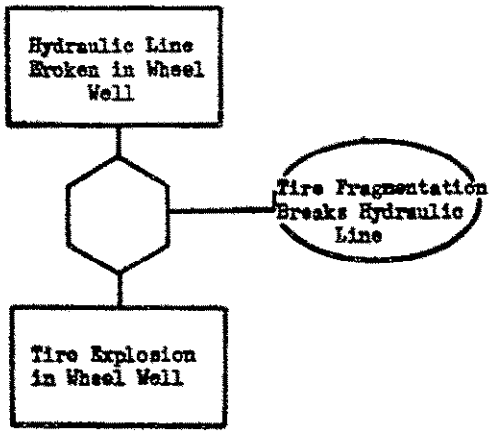


"RANDOM CONDITION INHIBIT" Gate

The "RANDOM CONDITION INHBIT" gate is the same as the "FUNCTIONAL CONDITION INHIBIT" gate except that the status of the conditional input to a "RANDOM CONDITION INHIBIT" gate is variable while it remains constant in the "FUNCTIONAL CONDITION INHIBIT" gate. The fault duration time of the output event is always generated within the gate.

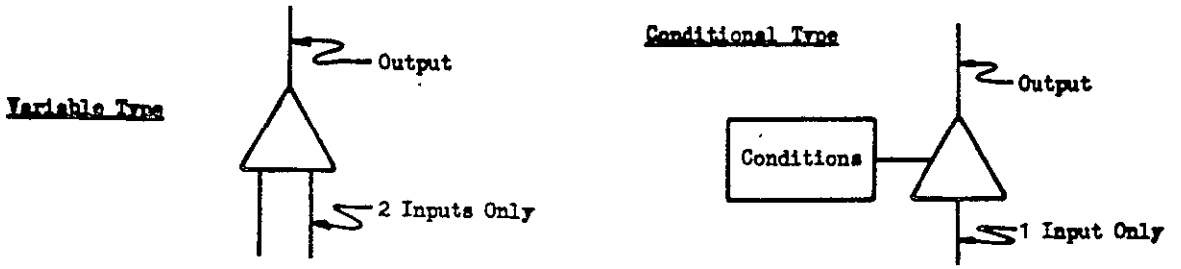


Example of "RANDOM INHIBIT" Gate Usage:



4.4 SPECIAL SYMBOLS

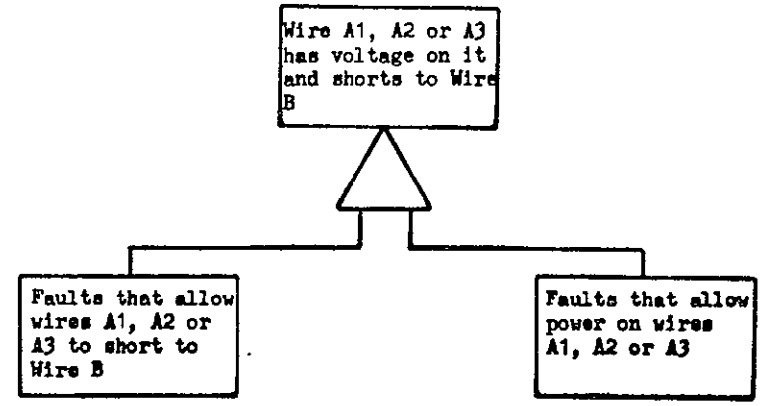
"MATRIX" Gate



The "MATRIX" gate is used to describe a situation in which an output event is produced for certain combinations of events at the inputs. A matrix showing the event combinations that produce the output event will accompany each usage of this symbol. (Refer to Appendix D for further details.)

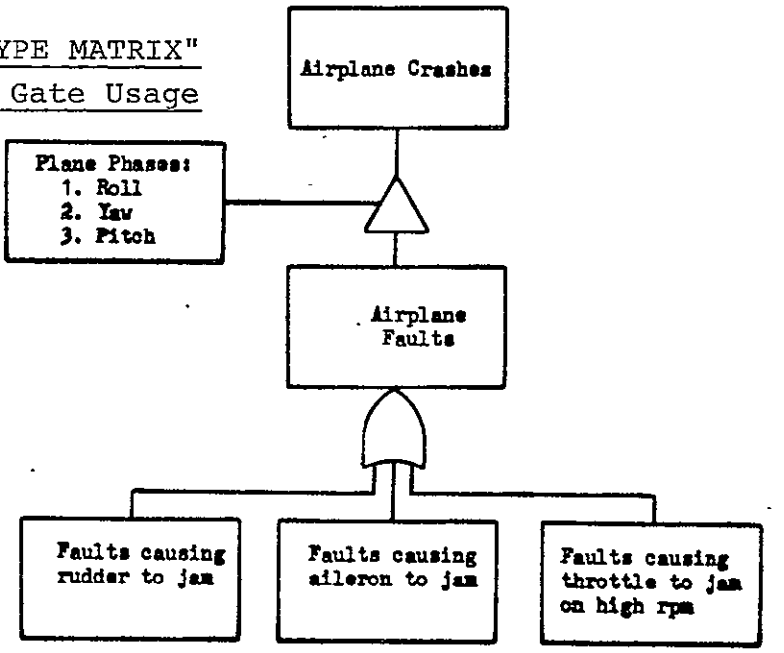
Example of "VARIABLE TYPE MATRIX" Gate Usage

	Power on A1	Power on A2	Power on A3
A1 shorts to B	1.0	0	0
A2 shorts to B	0	1.0	0
A3 shorts to B	0	0	1.0



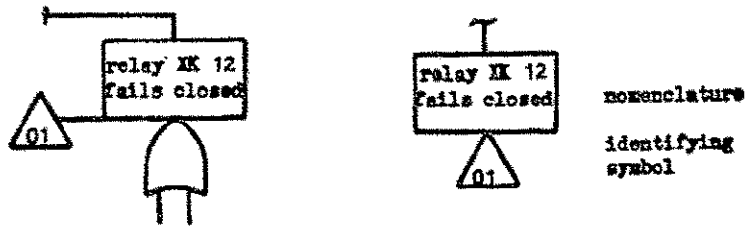
Example of "CONDITIONAL TYPE MATRIX" Gate Usage

	Rudder jams	Aileron jams	Throttle jams high rpm
Roll	.4	.5	.3
Yaw	.7	.8	.4
Pitch	.6	.7	.6



Transfer Symbols

The "transfer" symbol is used to allow continuity between two parts of a logic diagram. A line drawn into the side of a triangle transfers everything below that triangle to another location, which is identified by a triangle with a line drawn from the apex and containing matching nomenclature and identifying symbol. The methodology is illustrated below:



Two types of transfer symbols exist. The "internal" (local) transfer symbol transfers portions of a logic diagram only within a particular logic diagram. The idea behind this being that whenever the development of a certain portion of logic diagram is identical in two (or more) places, on the same logic diagram, it need only be developed in one place.

The "external" (global) transfer symbol transfers a portion of a logic diagram to another, entirely separate, logic diagram. This happens when a development is identical for one event on two separate logic diagrams. Also, when a diagram is developed until there is no longer room for further expansion on the sheet (or it is desired to end at a particular place) an external transfer is used to continue development on another sheet. This is the method by which new logic diagram developments (sub-diagrams) are started.

Figure 3 is an example of transfer symbol usage. It shows the correct use of both internal and external transfers.

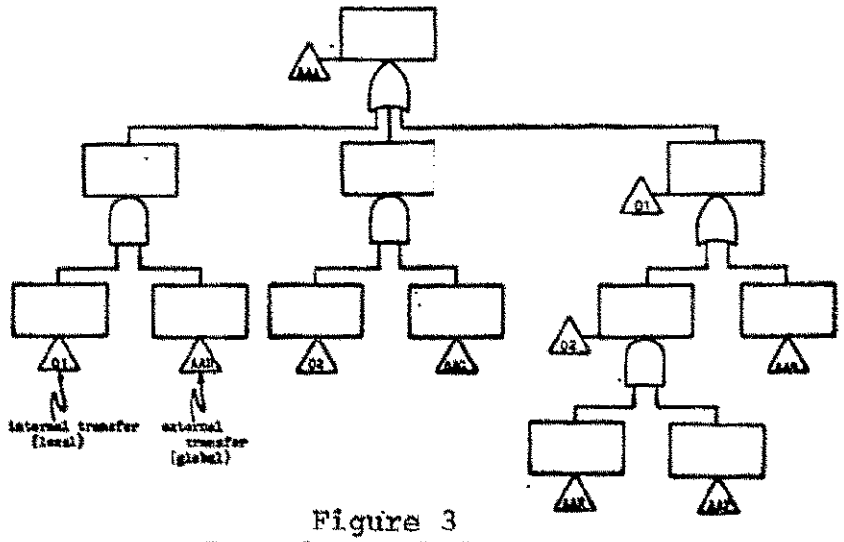
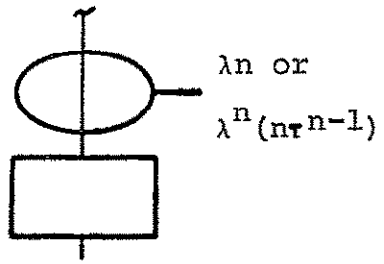


Figure 3
Transfer Symbol Usage

Output Encompassing Ellipse

An ellipse with a line extending out along the major axis is used when a component appears several times at the same place (e. g., a 10-stage counter where all 10 stages can be represented by illustrating one stage). Only one of the inputs is drawn to encompass the output. This indicates that the failure rate of that event is to be multiplied by the given factor (times 10 for the 10-stage counter) for an "OR" gate or raised to a given power and multiplies by the expression $(n\tau^{n-1})$ for an "AND" gate. This symbol is illustrated below.



4.5 · EVENT IDENTIFICATION

All events comprising a logic diagram must be identified by a code. This is necessary for four reasons: 1) easy and precise referencing, 2) for purposes of drafting, 3) in order for a log of events to be maintained, and 4) for purposes of quantitative evaluation. The means by which events are identified is generally dependent upon the requirements and objectives of the particular analysis. A standardized procedure should be set up and adhered to for an entire analysis program.

The size and complexity of the Apollo system has demanded that a unique method of event identification be utilized. The method chosen has been developed to satisfy the present requirements and objectives of the Apollo system oriented logic diagram analysis, plus allowance for future expansion or quantitative evaluation.

To begin, all events are classified into one of two categories. These two categories are referred to as "global" events and "local" events. Global events are defined as events which are used on more than one logic diagram, and local events are defined as events which are unique to one logic diagram. The notation (or code) for events allows each event to be uniquely represented, at the same time differentiating between global and local events. The standardized notation is shown in Figure 4.

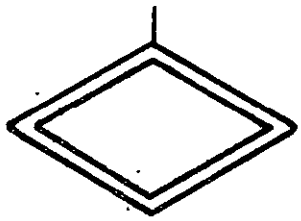
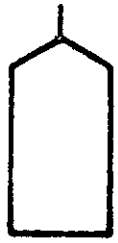
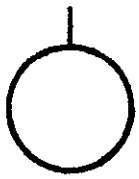
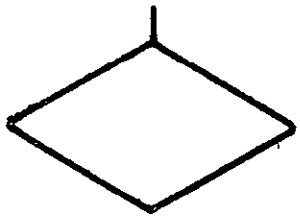
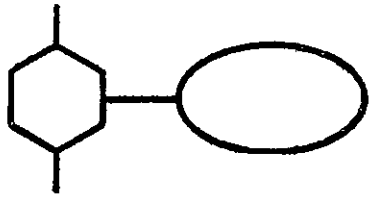
LOCAL EVENT		GLOBAL EVENT
V01 thru V99		V100 thru V999
W01 thru W99		W100 thru W999
X01 thru X99		X100 thru X999
Z01 thru Z99		Z100 thru Z999
Y01 thru Y99		Y100 thru Y999
01 thru 99		AAA thru ZZZ

Figure 4
Standardized Event Notation

From Figure 4, it can be readily discerned that the alpha character identifies the type of event. That is, "W" indicates a house, "X" indicates a circle, "Z" indicates a diamond, any "Y" indicates an oval. Local events are numbered from 01 through 99 for each and every logic diagram. For example, diamonds on the AAA logic diagram are randomly numbered as Z01, Z02, Z03, etc., and diamonds on the RAA logic diagram are also numbered as Z01, Z02, Z03, etc. The only way to differentiate between local events is by indicating the logic diagram on which they are located. Global events are numbered from 100 through 999 and an index must be used to locate logic diagrams on which these events appear.

For the identification of global transfers (sub-diagrams) a three character alpha system is utilized. Using three alpha characters allows identifying nomenclature for a possibility of 17,576 logic diagrams. In conjunction with this method, a breakdown has been established which immediately identifies the source of the logic diagram. This breakdown consists of delegating the first letter, of all three letter combinations, to a particular MSF Center. The breakdown is as follows:

AAA thru AZZ	WDC/Seattle	(676 diagrams)
BAA thru HZZ	MSFC	(4732 diagrams)
IAA thru QZZ	KSC	(6084 diagrams)
RAA thru ZZZ	MSC	(6084 diagrams)

As shown, local transfer symbols are numbered from 01 through 99 for each logic diagram. When referring to a particular local transfer, the diagram on which it appears must also be given.

4.6 BASIC DIAGRAM METHODOLOGY

The development of a logic diagram commences with the definition or identification of the top "undesired event" to be analyzed. The top undesired event can be an encompassing event, such as "mission loss," indicating a complete system analysis; it could be a limiting event, such as "crash due to engine failure," indicating analysis of an identifiable potential hazard; or it could be a specific event, such as "amplifier fails resulting in low output," indicating analysis beginning at a hardware level. Once definition of the undesired event has been accomplished, the system is analyzed using the following rules and

definitions of logic diagramming to determine and model the inter-relationships and combinations of both normal and abnormal system functions which could cause the occurrence of the top undesired event.

The next step is to divide the system operating modes into phases. A phase is that increment of a system's life which can be analyzed independently, yet recognizing that there may be commonality of analysis between any of the phases. Some phases may be further divided into sub-phases, depending upon the system structure. System phase breakdown should continue (corresponds to system engineering functional analysis) until the environment stays relatively constant through the phase element and system operational characteristics do not change the fault environment. The development of a logic diagram proceeds through the identification and combination of the system events (normal and fault) until all fault events are definable in terms of basic identifiable hardware faults, to which failure rate data can be applied.

Figure 5 shows the general relationship of logic diagram segments. Although shown as distinct elements, it should be noted that the segments will, to a certain extent, "mix" together throughout the logic diagram structure.

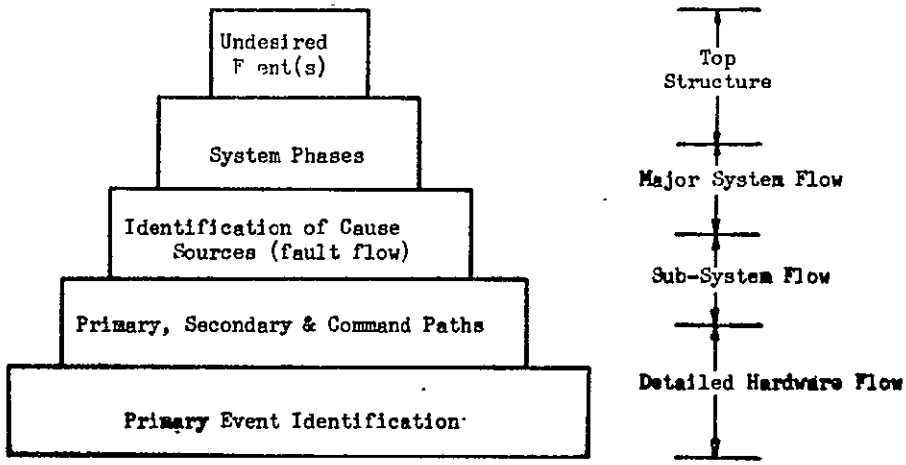


Figure 5
Logic Diagram Segments

Developing the "fault flow," or cause and effect relationship of events through a system, requires deductive reasoning at each "gate event" or level of the logic diagram. This deductive reasoning basically involves the answering of five questions: 1) necessity, 2) sufficiency, 3) primary, 4) secondary, and 5) command. These questions effectively develop the structure of the logic diagram on a progressive, or level-by-level, basis.

To answer the questions "necessity" and "sufficiency" requires an evaluation of the system for normal and abnormal functional event relationships. This evaluation determines the system unique events, and logic gates combining them, to result in the undesired event. This is accomplished by looking at the undesired event and asking, "What is necessary and sufficient to cause this undesired event?" For example, an ordnance device will be activated when two events occur: 1) the ordnance device Safe and Arm mechanism closes, "AND" 2) energizing power is available on the ordnance device ignition line. These two events are all that is "necessary" and "sufficient" to cause activation of the ordnance device.

The questions "primary" and "secondary" are questions requiring an evaluation of the system to determine what primary and/or secondary fault events can occur to result in another fault event. These questions also identify the specific primary and secondary failure modes of the fault event. For example, a primary failure mode of an ordnance device would be the mode of auto-ignition. A secondary failure mode would be that of ignition due to excessive external shock or heat.

The question "command" is really a guideline for development through the system. The question asks, "What upstream event will command the downstream event to occur?" The upstream event may be a primary and/or secondary event, or it may be an event commanded by an event further upstream. Essentially, the "command path" is a chain of events delineating the failure path of command events through a system. The command path ultimately results (at the finish of the analysis) as a primary and/or secondary fault event. Take for example, a set of relay contacts failing closed, as part of a system function. The contacts may fail closed as a primary failure, they may fail closed from a secondary cause such as foreign material bridging the contacts, or they may be commanded to close by a relay coil failure. If an upstream event causes the relay coil to be energized, the contacts are effectively "commanded" to close as a result of this upstream event.

The effective inter-relationship of the five necessary deductive questions is shown below:

Logic Diagram	necessity sufficiency	primary event secondary event command event

As indicated, a logic diagram is constructed of primary events, secondary events and command events through the medium of necessity and sufficiency.

In developing a logic diagram certain thought processes take place in the mind of the analyst. The steps of development at each level of the logic diagram delineating these thought processes are:

1) Define the undesired output event;

2) Determine what is "necessary and sufficient" to produce the undesired output;

3) List all primary events related to the undesired output;

4) List all secondary events related to the undesired output;

5) Define the undesired input event which could command the output event;

6) Repeat steps 1 - 5 for the new undesired event defined in step 5.

Figure 6 shows the relationship of the above steps to the structure of a logic diagram. The inherent simplicity and logical process is readily apparent from this example.

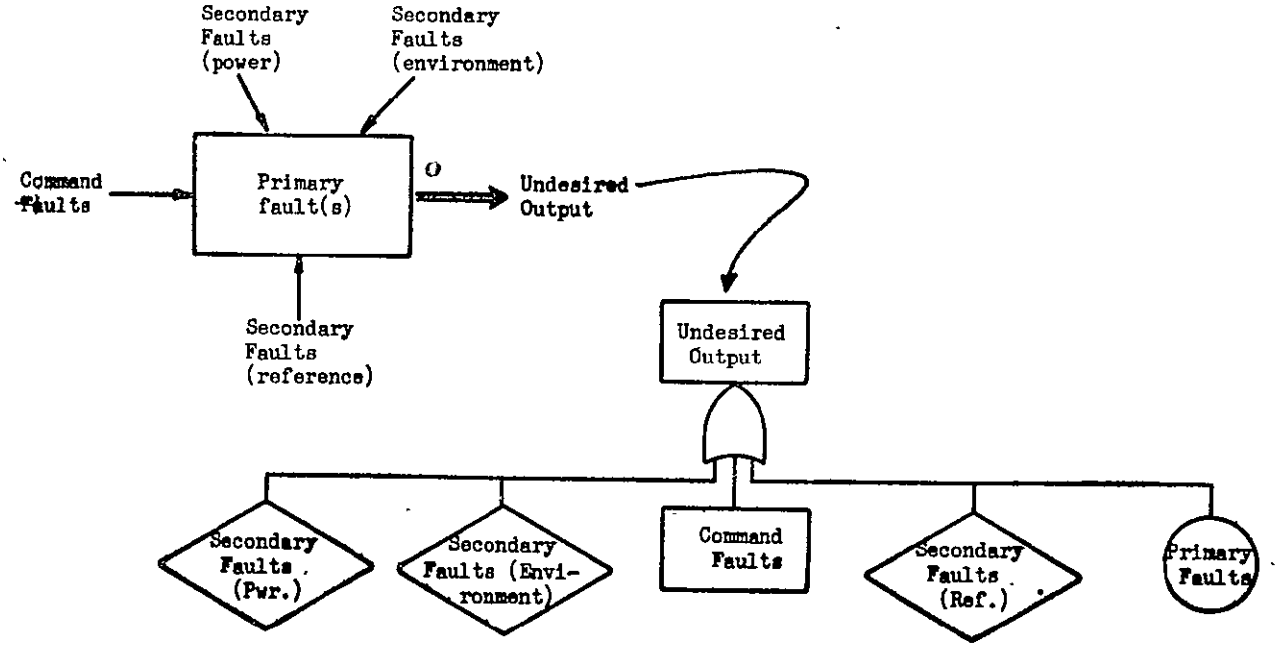


Figure 6

Logic Diagram Relationships

Figure 7 shows a logic diagram structure which portrays the relationship of the command event to the primary and secondary events, and also how command events lead to a "command path." It must be remembered that the command path, as such, is only a guideline for analysis of event development through a system. Command events create an orderly and logical manner of analysis at each level of the logic diagram. Once an analysis is completed, comparison between the logic diagram and signal flow diagram will show that the logic diagram "command path" of a branch will represent the steps of signal flow along a single thread.

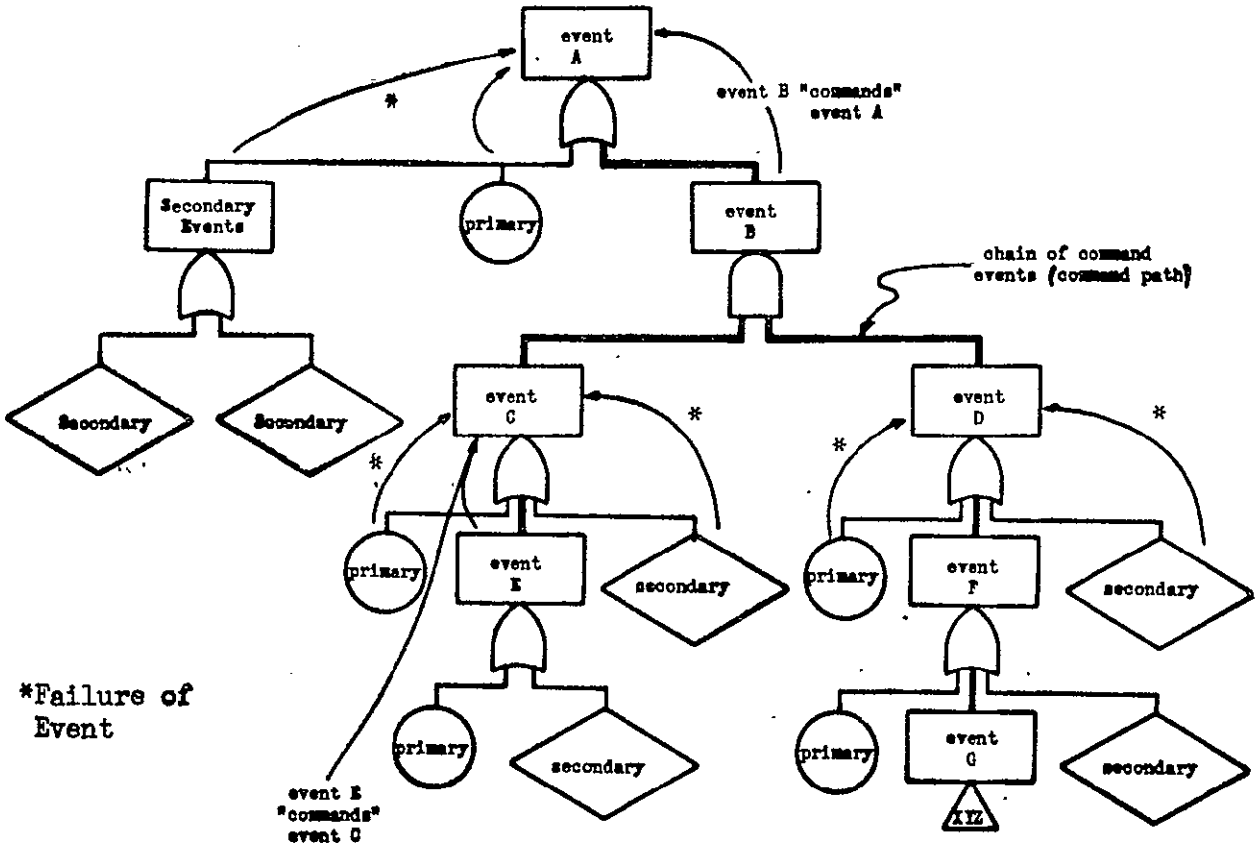


Figure 7
Example of Command Path

4.7 THE HUMAN ELEMENT

Logic diagram analysis has been applied to hardware and software elements of a system extensively. The human element has been included only tc the extent of "lumping" all cause and effect relationships into a single consideration of human failure.

Any system which requires the human element in order to perform its intended function must have an analytical development that includes the human as part of the system. The human element is as much a subsystem as are the hardware and software subsystems which make up the rest of the total system. The human element is much more complex and the least understood perhaps, but still a subsystem of the total system.

Since the human element is a major subsystem of Apollo, analyses must include human aspects in a natural manner. In other words, human cause and effect relationships must be an integral part of the system hardware and software logic diagram structure.

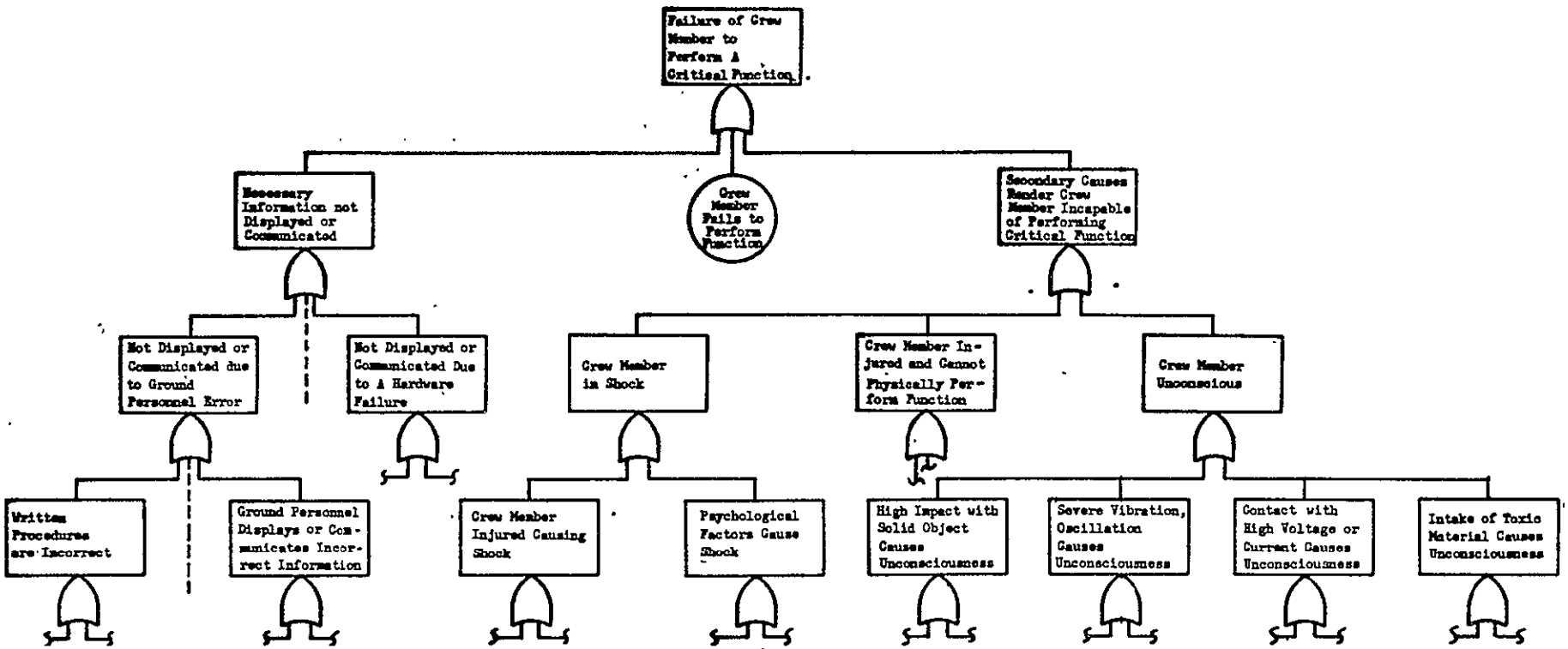
An example of how the human element can be portrayed in a logic diagram is shown in Figure 8. The top event defines any arbitrary human operation and is used merely to illustrate the development below the event. The circle shown as "Crew Member Fails to Perform Function" (the identified critical function) represents the possibility of inadvertent error, usually highly improbable. The other two inputs to the top "OR" gate represent the "command" (no input information) development and the "secondary" cause development. Either of these two branches will most likely contain the dominant factors associated with failure of a crew member to perform a critical function. The events shown in this logic diagram, Figure 8, are examples of the types of causes which could result in no action taken by a crew member. There are others which for simplicity are not shown in this illustration (indicated by dotted lines).

During the development of system safety logic diagram analyses on the Apollo mission, the human element must be considered an integral part of the total system.

4.8 DOMINANT PATH

A dominant path is the chain of events which is most "likely" to result in the undesired event (potential accident). In a typical case, there may be several paths of various degrees of dominance which can result in a given undesired event. These chains and their associated degrees of dominance are most clearly identified by the system safety model (logic diagram). Dominant path(s) and their relative degrees of dominance are determined by event weighting (inspection) or rigorous mathematical solution of the model.

Since the dominant path is the most likely avenue along which the undesired event(s) can occur, the most cost effective approach is to concentrate the initial prevention effort in this area. It may be necessary to consider other paths within the model, in a descending order of dominance,



JMAN ELEMENT EXAMPLE

Figure 8

in order to achieve an acceptable level of risk for the occurrence of a particular undesired event.

Preparing to locate dominant paths requires that the system safety model for a given undesired event (potential accident) has been developed to the extent necessary to identify dominant paths. As a minimum, the logic diagram development must encompass all those safety features and devices which have been designed into the system. This assures that adequate consideration has been given to those areas of the system which are of the greatest "risk," since safety devices are normally placed where the greatest risk of an undesired event occurring exists.

Logical inspection or mathematical processes determine the degree of dominance for those paths of the model which contribute the most to the likelihood of the undesired event. The term "logical inspection" is defined to mean the logical thought processes of a trained and experienced analyst being applied through examination of the model, these processes associated with whatever mental weighting factors he may consider during the examination to determine which events "look to be" more probable than others. This would lead to the resulting statement by the analyst of "these events (identified) and path(s) look to be the most probable."

The term "mathematical process" can be a solution of the model by any of the methods discussed in Appendix B. Normal practice is to solve a diagram with 250 events or less by the Lambda-Tau (hand calculated) method and a diagram with greater than 250 events on a digital computer using Monte Carlo simulation with importance sampling. An event in this case is defined to be any element of the diagram other than a logic gate. Since the purpose of the quantitative evaluation of a diagram is to identify dominant paths and their relative significance, the diagram is usually simplified by inspection to minimize the structure to be simulated. This inspection is the elimination of those events and branches which are obviously insignificant compared to others which are inputs to the same gate.

Control of the dominant path(s) is accomplished by the following:

1) Establish a predetermined limit within which the initial path selection is bounded. This involves the identification of those paths which are computed to be above any established limit for the system.

   If the paths are near or below the limit, then they are selected by picking those which are within an "order of magnitude" or so of the limit, or are of the same type.

2) The initial selection must be divided into groups for which a set of predetermined limits has been established for each grouping. The grouping of paths is accomplished by selecting those within an order of magnitude of each other or those which have an apparent commonality within the system.

3) Determine if a common point of departure exists among the paths of each group. This evaluation involves determining if there are common faults among the paths. Recommended changes to the system at these common points provides the most effective way to eliminate paths, or at least reduce them to an acceptable level.

4) Convert the logic diagram dominant paths by grouping events at logical summary points. Conversion of the logic diagram dominant paths involves making a listing of these events which, when "OR"-ed, result in an interim event. The method is to convert each path to a simplified alternating "AND," "OR," "AND," "OR," etc., relationship.

5) Simplify the logic diagram of the dominant path by logically rediagramming. Simplification involves rediagramming the relationships summarized in step 4. This results in a simplified diagram of each path which can be readily correlated with a functional flow diagram of the system. The paths can now be verified as to accuracy and the actual fault points introduced into a functional flow diagram to show where and how the fault combinations affect system operation.

6) Determine those events for which a design change or the development of a procedure will best and most cost effectively reduce the probability of occurrence of an undesired event to an acceptable level of risk.

7) Insert alternative solutions as derived by steps 1 through 6 and repeat the process until an acceptable level of risk is obtained. This step involves working with designers and selecting several alternative system changes to reduce the probability of occurence of each path. For each alternative to be evaluated, the logic diagram is changed to reflect the change and the diagram is recomputed to determine the change impact. Care must be exercised to assure that other paths or branches of the diagram which have the same event or fault sequence are also changed to reflect the change being evaluated.

8) Advise appropriate level of management of findings and recommendations.

5.0 DRAFTING OF LOGIC DIAGRAMS

At the present time two methods exist for drafting logic diagrams in a document form. The first, or traditional method, is hand drafting which requires draftsmen or illustrators. The second method is mechanized drafting, which uses a computer and electro-mechanical plotter.

The mechanized drafting method, which has recently been developed for the drafting of logic diagrams, uses a Univac 1108 computer in conjunction with a Cal-Comp (California Computer Products) incremental X-Y Plotter. A preliminary cost analysis shows that the mechanized method can cost approximately one-third of the hand drawn method.

For the Apollo System Safety effort, the most efficient and cost-effective drafting of logic diagrams can be accomplished using the mechanized drafting method. This is true for four reasons: 1) The total Apollo Program integrated logic diagram analysis will be comprised of a large quantity of diagrams, 2) document quality vellums can be produced in approximately two days for use as working drawings or for support of meetings, 3) revisions and updates can be made with little expense or effort, and 4) the cost savings involved.

Figure 9 shows the steps required for mechanized drafting of logic diagrams. This process begins with the engineer's rough draft (logic diagram) which is coded, in a specified format, on a keypunch form. The information on the keypunch form is punched in standard computer data cards, which are combined with a standardized computer program and submitted to the computer. The computer calculates the spacing between blocks and instructs the plotter in the proper manner to draw (plot) a particular logic diagram. When the computer completes all calculations it outputs the necessary information on a magnetic tape. The magnetic tape is then placed in the Cal-Comp tape reader, which translates, the tape into mechanical movement instructions for the plotter. Once the information is placed on magnetic tape, it is semi-permanent. That is to say, it remains on the tape and can be used as many times as desired to produce plots. To make a change in a drafted logic diagram, it is only necessary to make the appropriate computer data card change(s) and generate a new magnetic tape.

In order for the computer to process a logic diagram, the "deck" of computer data cards must be arranged in a certain order, with the information on these cards in a specific format. Transferring information from the rough draft logic diagram onto computer data cards, via keypunch forms, is referred to as "coding". The required arrangement of cards in the logic diagram plotting deck is shown in Figure 10.

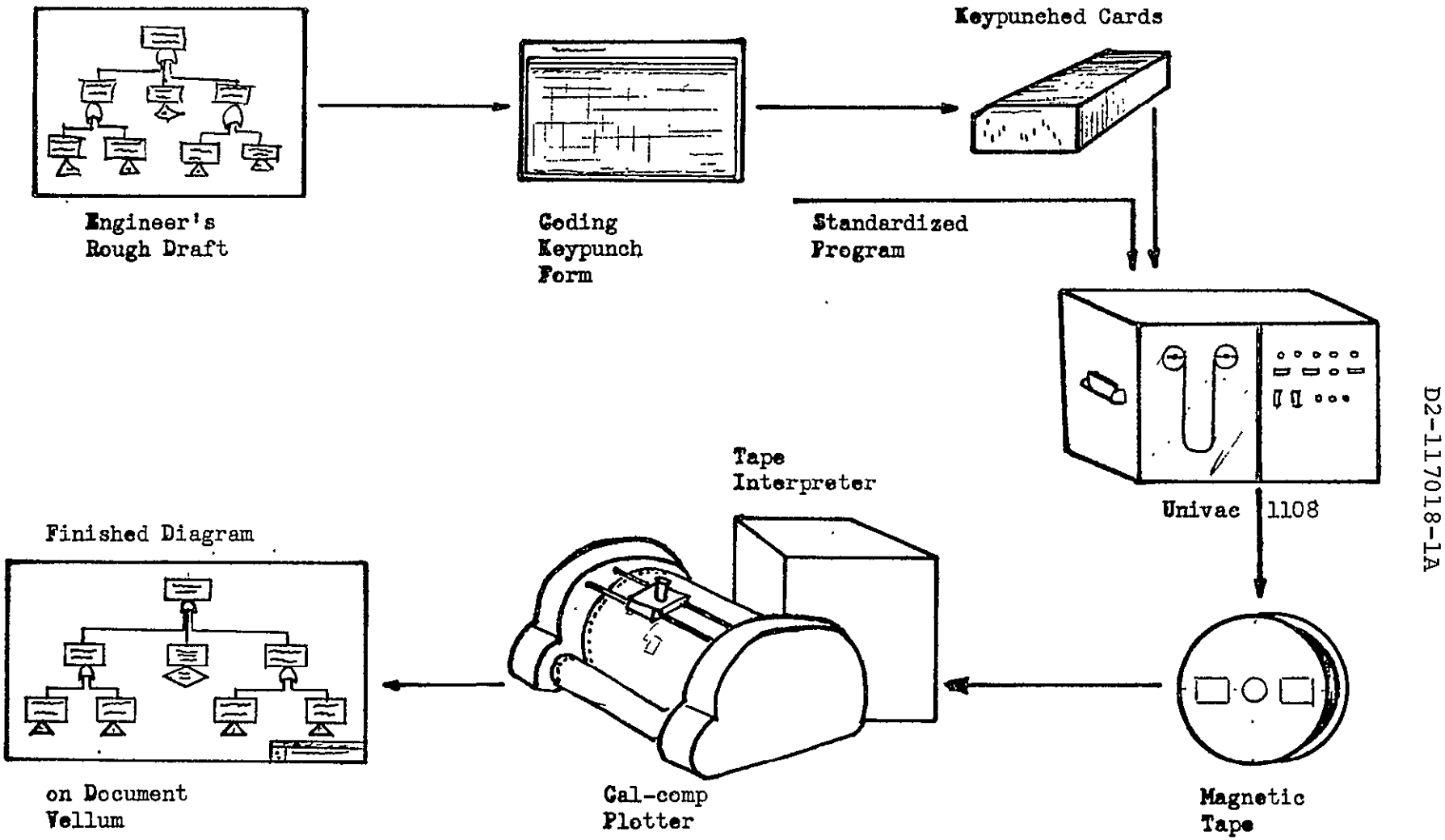


Figure 9

STEPS OF MECHANIZED DRAFTING

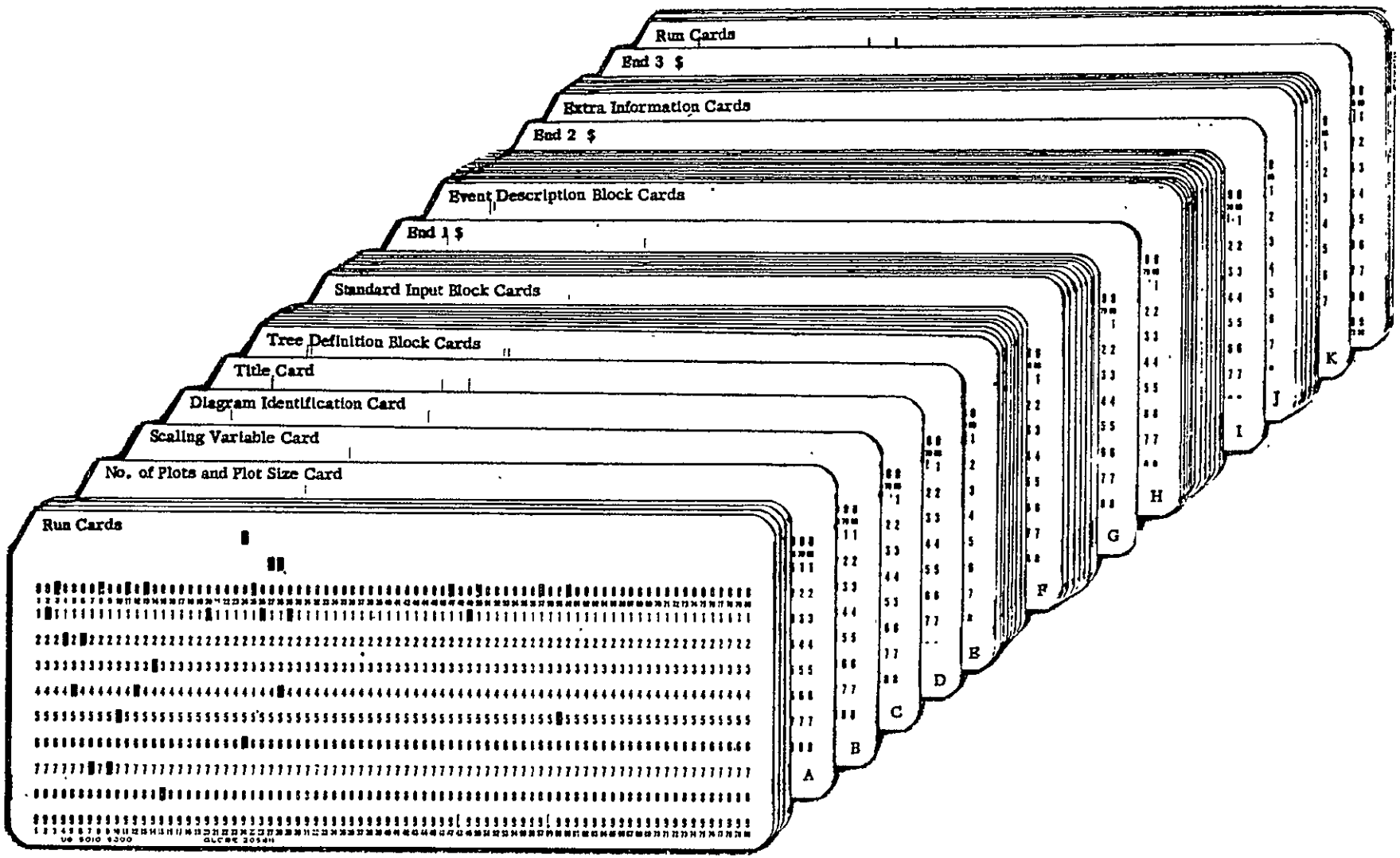


Figure 10
ARRANGEMENT OF COMPUTER CARD DECK

"Run cards" are special cards necessary for a program to enter and exit from the computer. These cards are generally unique to a particular computer system. As shown, these cards are at the beginning and end of the deck.

To begin the coding, all "events" on the rough draft logic diagram must be identified. Gates are identified using an alphanumeric system. The top gate (first gate) is identified as the "A1" gate. Gates on the second level are identifies as B1 through B14, moving from left to right on the logic diagram. Gates on the third level are identified as C1 throu C40, and so on, until the ninth level is reached where the gates are identified as I1 through I40. Circles, diamonds, houses, ovals, and triangles are identified according to the procedures defined for a particular program. For the Apollo logic diagram analysis, these events will be identified as defined in Paragraph 4.3.

The information that must be transferred from the rough draft to computer data cards is delineated as follows:

A- Number of Plots and Plot Size Card

States the number of plots (logic diagrams) in the particular run and the size of the plots. The size of the drawing can be varied from an 11 x 260 inch vellum to a 29 x 260 inch vellum.

B- Scaling Variable Card

States the desired distance between rectangles on a drawing.

C- Diagram Identification Card

States the revision letter, sheet number, document number and model which will appear on the drawing.

D- Title Card

States the title which will appear on the drawing.

E- Tree Definition Block Cards

Defines the structure of the logic diagram. Effectively, they position the location of each "event", state the type of gates used, the inputs to each gate, and identify all ovals.

F- Standard Input Block Cards

Defines the type of input for all input events called out in Section D. The types are: triangle, circle, diamond and house.

G- End 1 $ Card

This is essentially a special program run card. It denotes to the computer the end of a certain section of information.

H- Event Description Block Cards

These cards state the words which will appear in each event on the drawing.

I- End 2 $ Card

Same as card G.

J- Extra Information Cards

These cards locate and state words in transfer symbols that are connected to the side of a block, if any. Also, state information under any particular transfer symbol, if desired. It is possible for this section to be void of any cards.

K- End 3 $ Card

Same as card G.

Certain limitations must be observed for each mechanized logic diagram drawing. These limitations are written into the standardized program and could be changed if necessary. They are:

1) 10 levels of rectangles, or nine levels of gates;

2) 14 inputs per gate;

3) 40 events per level;

4) 250 events per logic diagram;

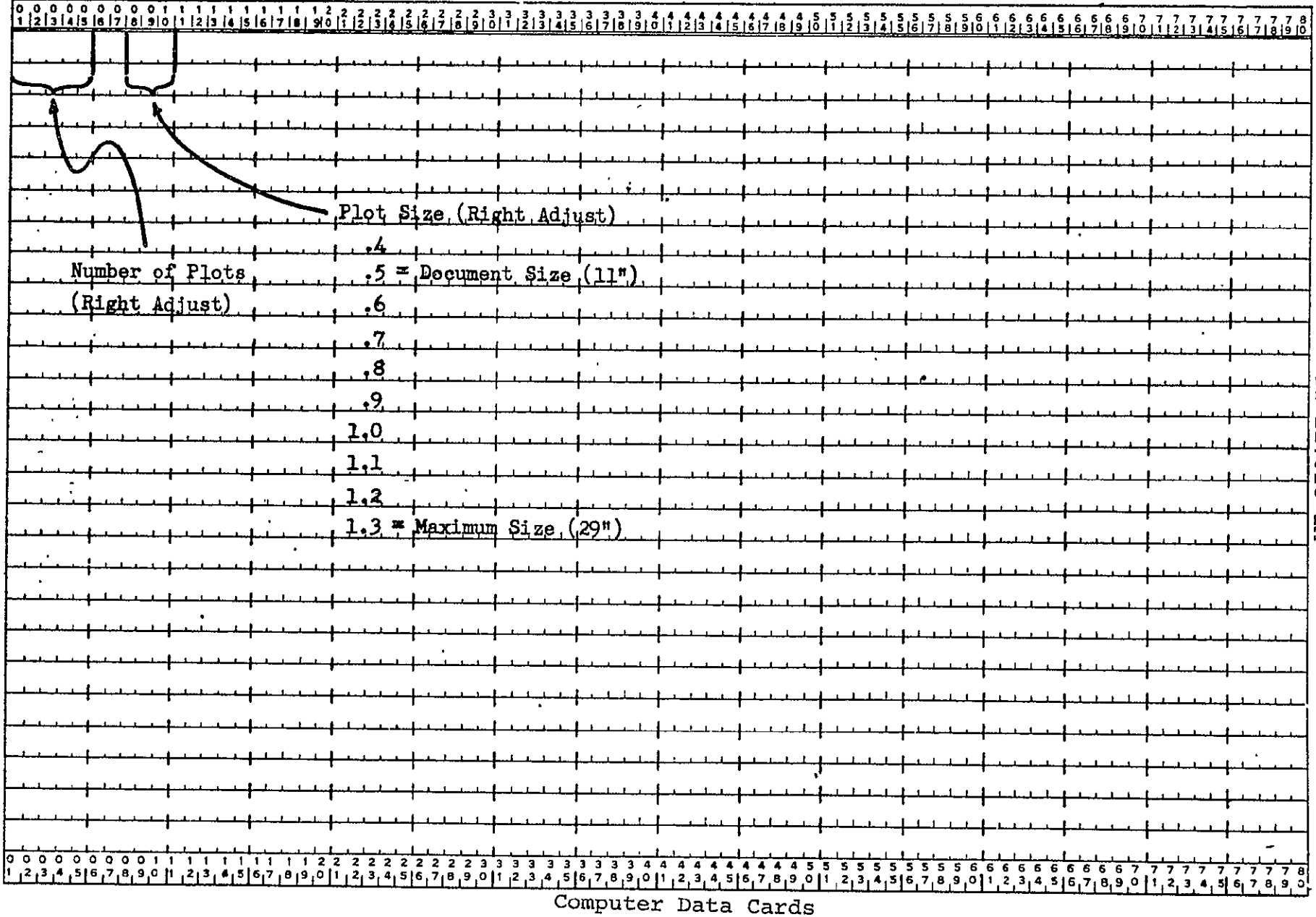
5) 260 inches by 29 inches vellum size.

The positioning of information on the computer data cards (Figure 11) is shown on pages 5-7 through 5-17 as it would be done on keypunch forms. It must be remembered that each line on a keypunch form represents one computer data card,

and that the numbered columns on the keypunch form correspond to the same columns on the data cards.

Figure 12 (page 5-18) is an example "listing" (printout) of the computer data card deck required to produce a drafted logic diagram. The coded information shown was used to produce the example logic diagram in Figure 13 (page 5-19). Column numbers have been purposely placed at the top of the listing in order to denote the proper perspective of the coded information. This example demonstrates the relationship between the coded information and the finished logic diagram.

A - Number of Plots and Plot Size (1 Card)



Computer Data Cards

Figure 11

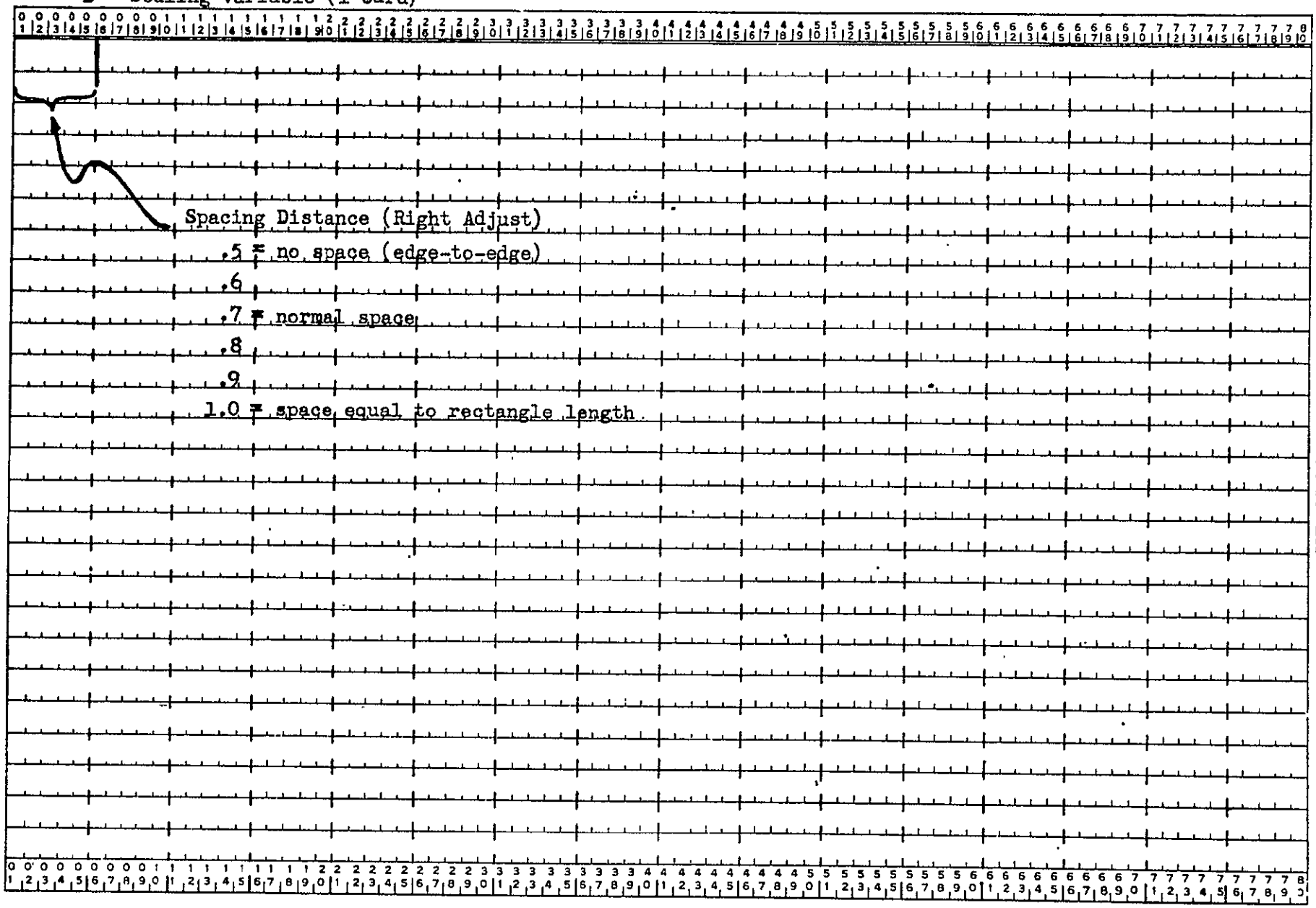


Figure 11 (Cont'd)

C - Diagram Identification (1 Card)

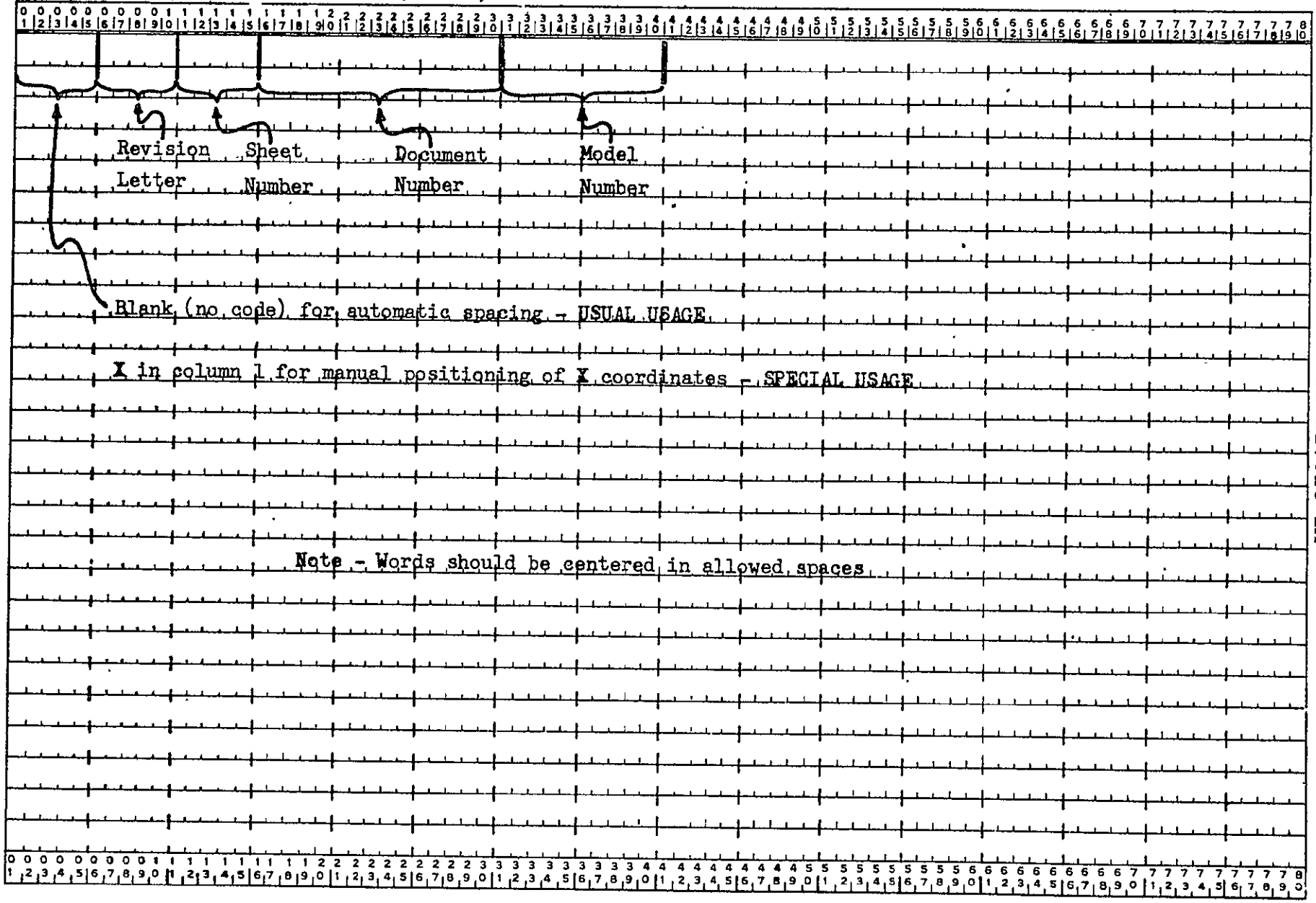


Figure 11 (Cont'd)

D - Title (1 card)

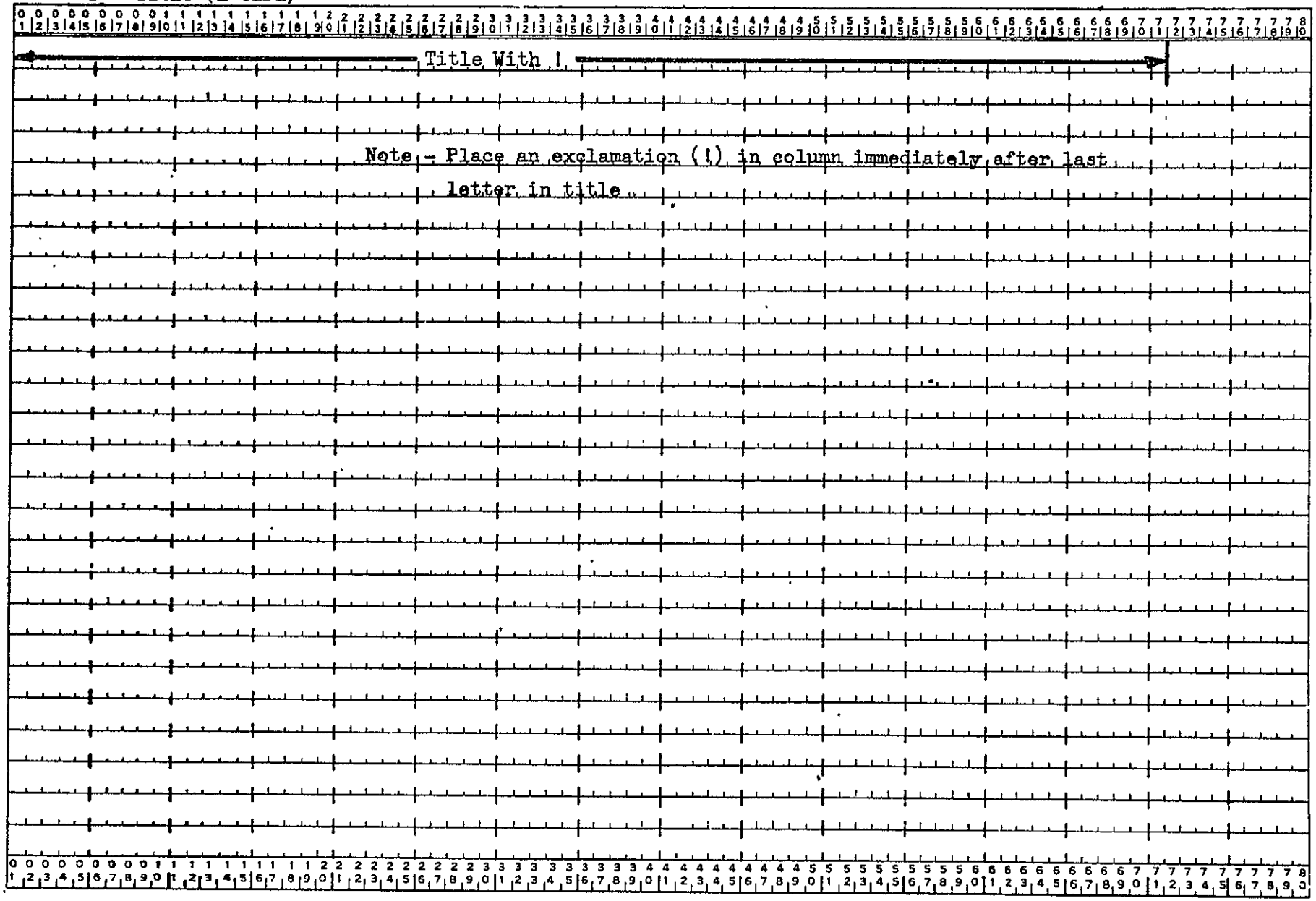


Figure 11 (Cont'd)

E - Tree Definition Block (more than 1 card)

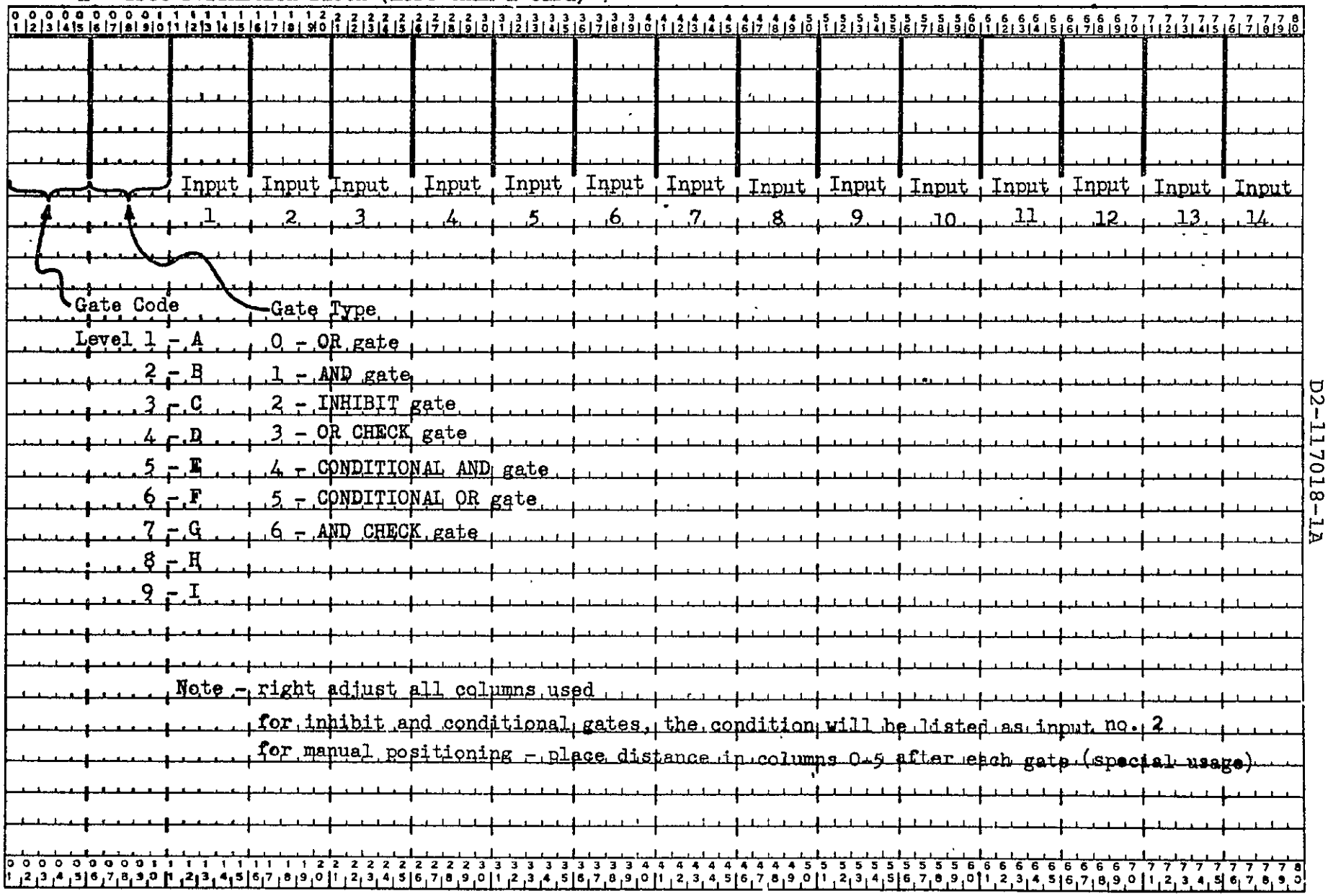


Figure 11 (Cont'd)

F - Standard Input Block (more than 1 card)

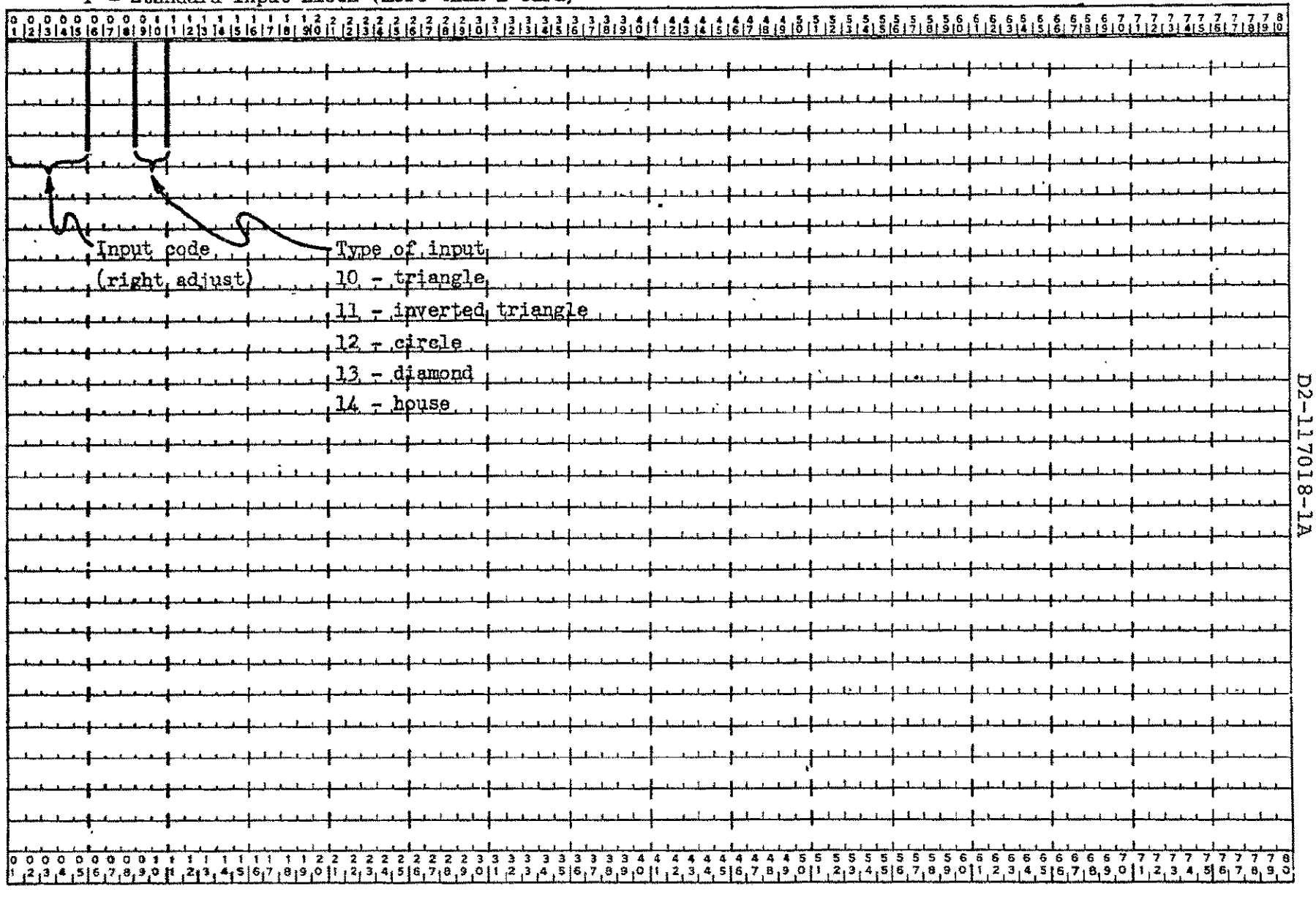


Figure 11 (Cont'd)

G - End 1 $ (1 Card)

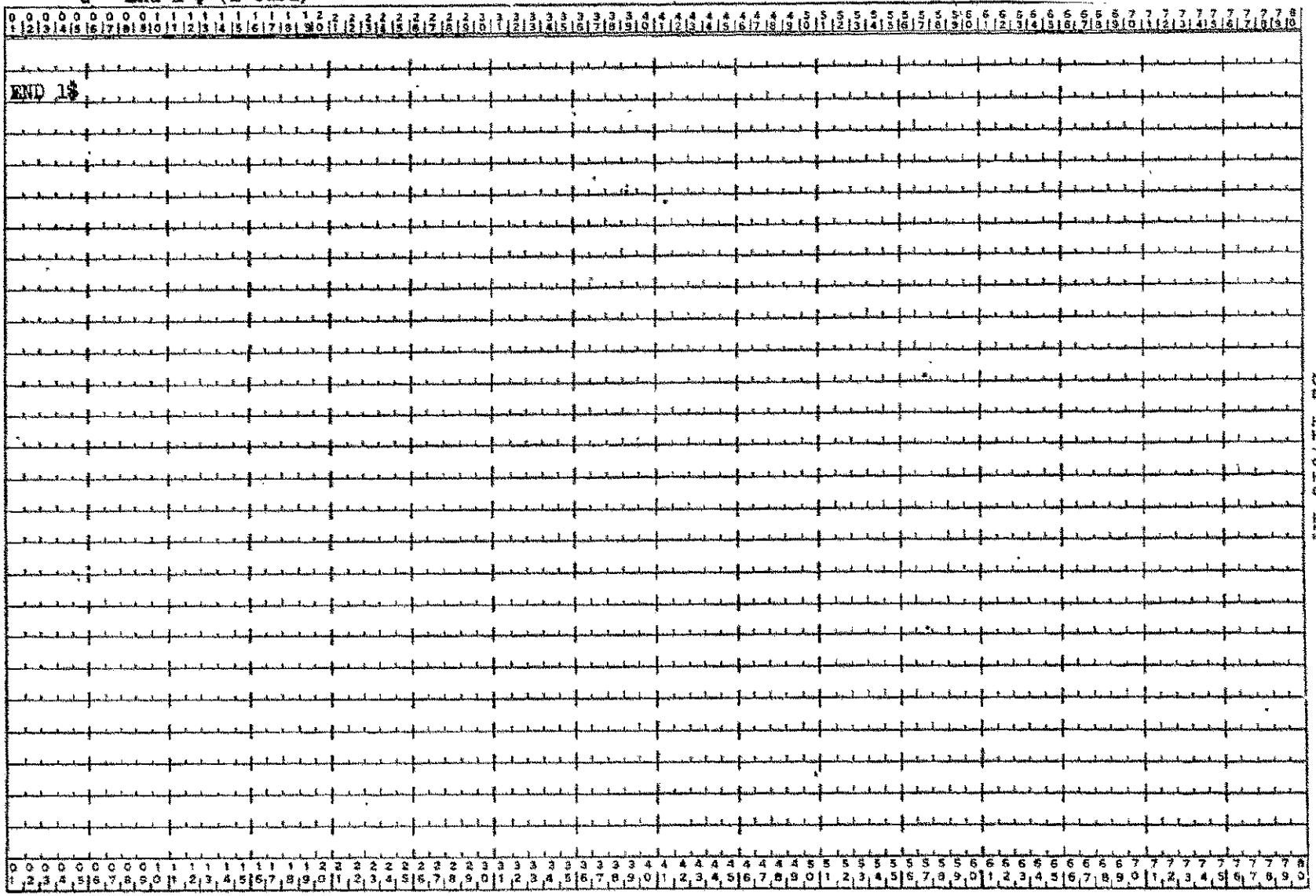


Figure 11 (Cont'd)

H - Event Description Block (more than 1 card)

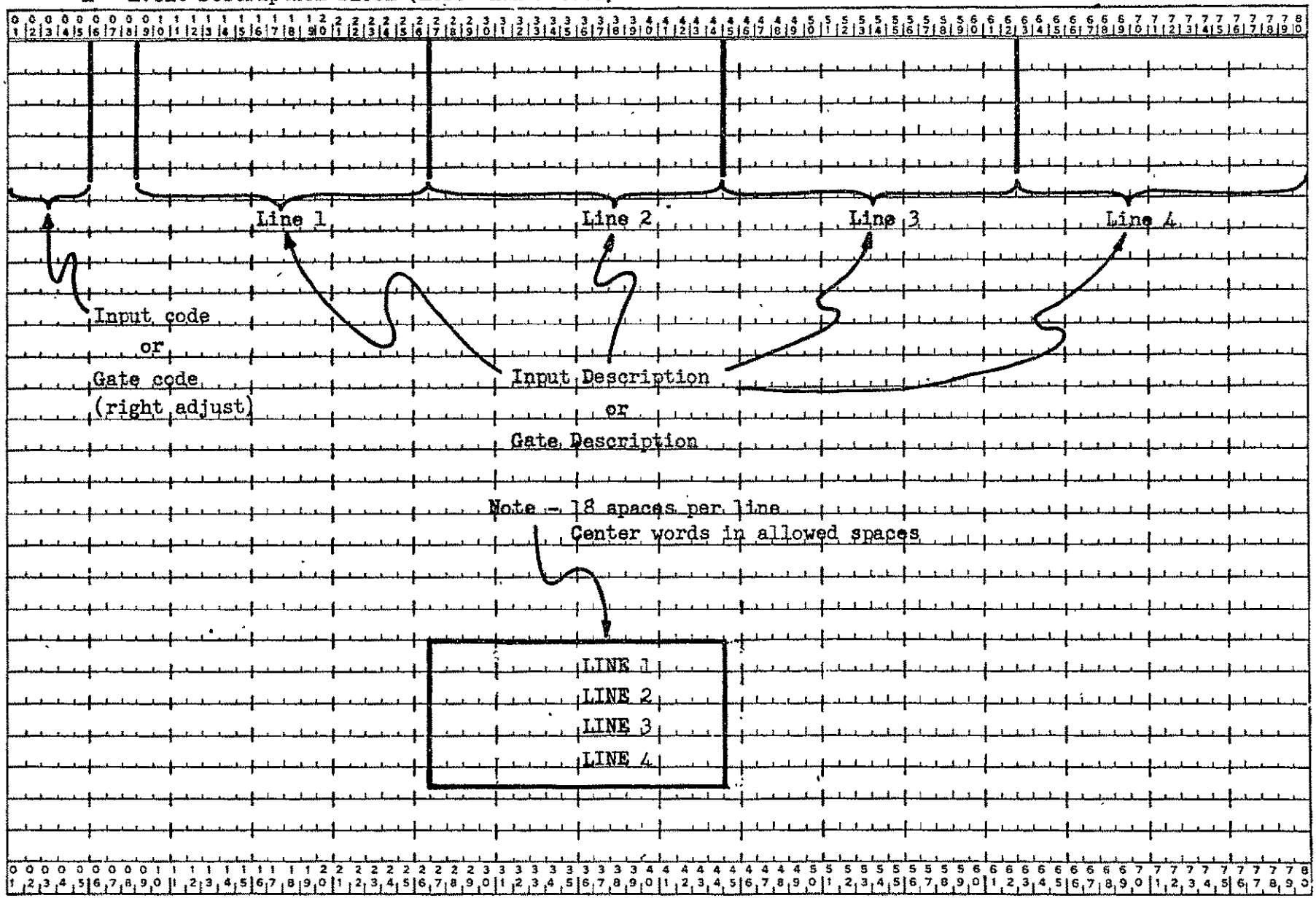


Figure 11 (Cont'd)

I - End 2 $ (1 Card)

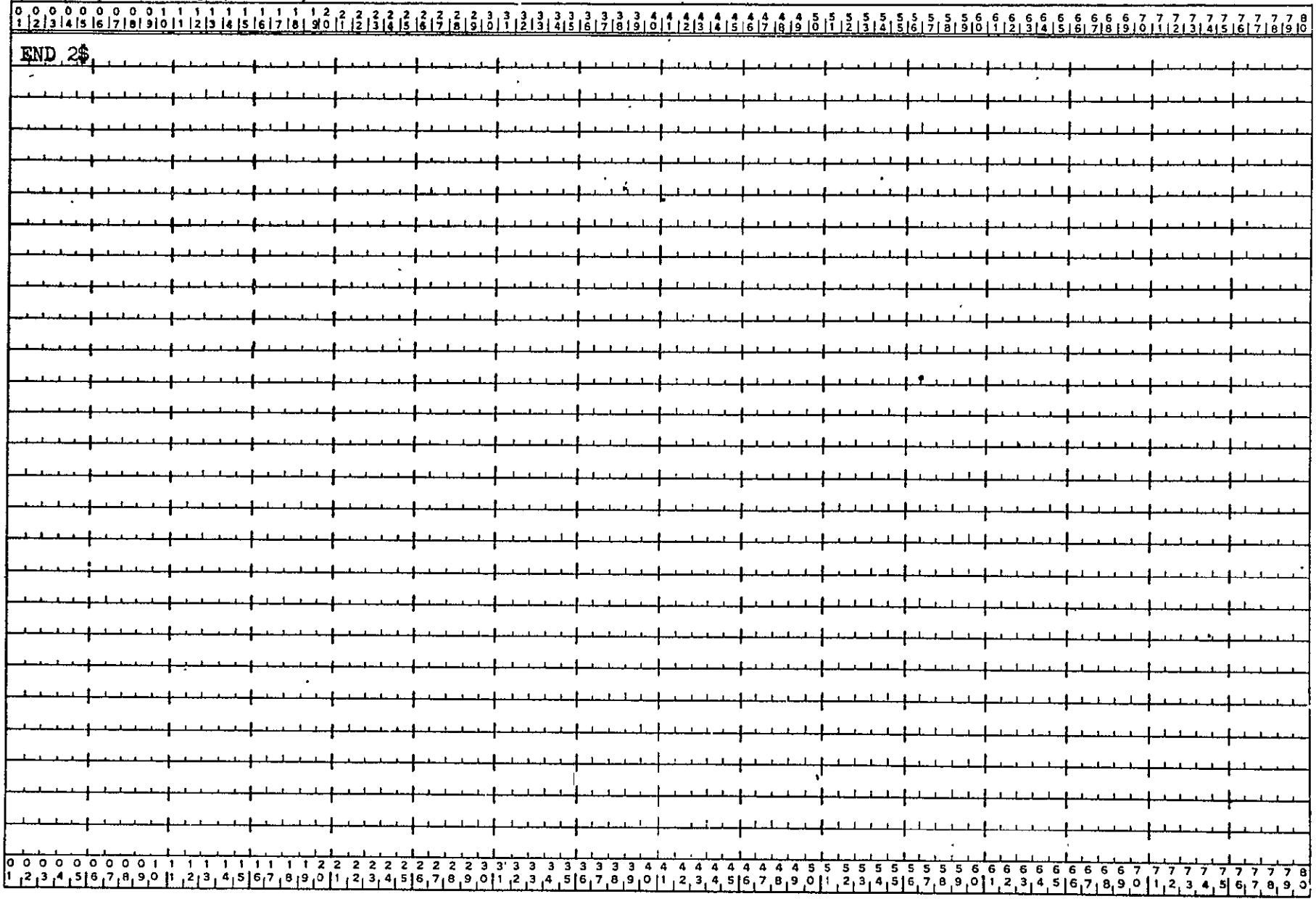


Figure 11 (Cont'd)

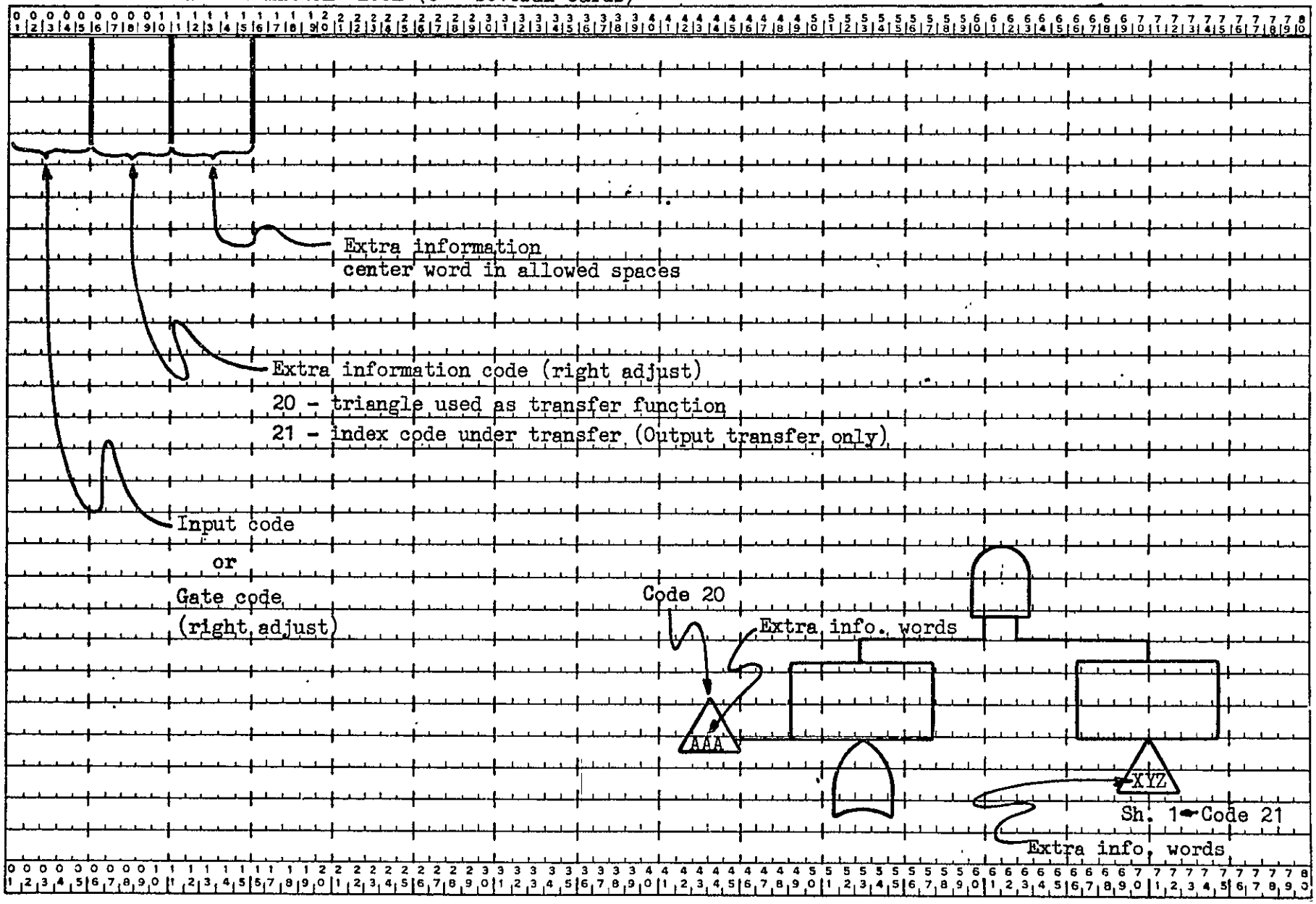


Figure 11 (Cont'd)

K - End 3 $ (1 card)

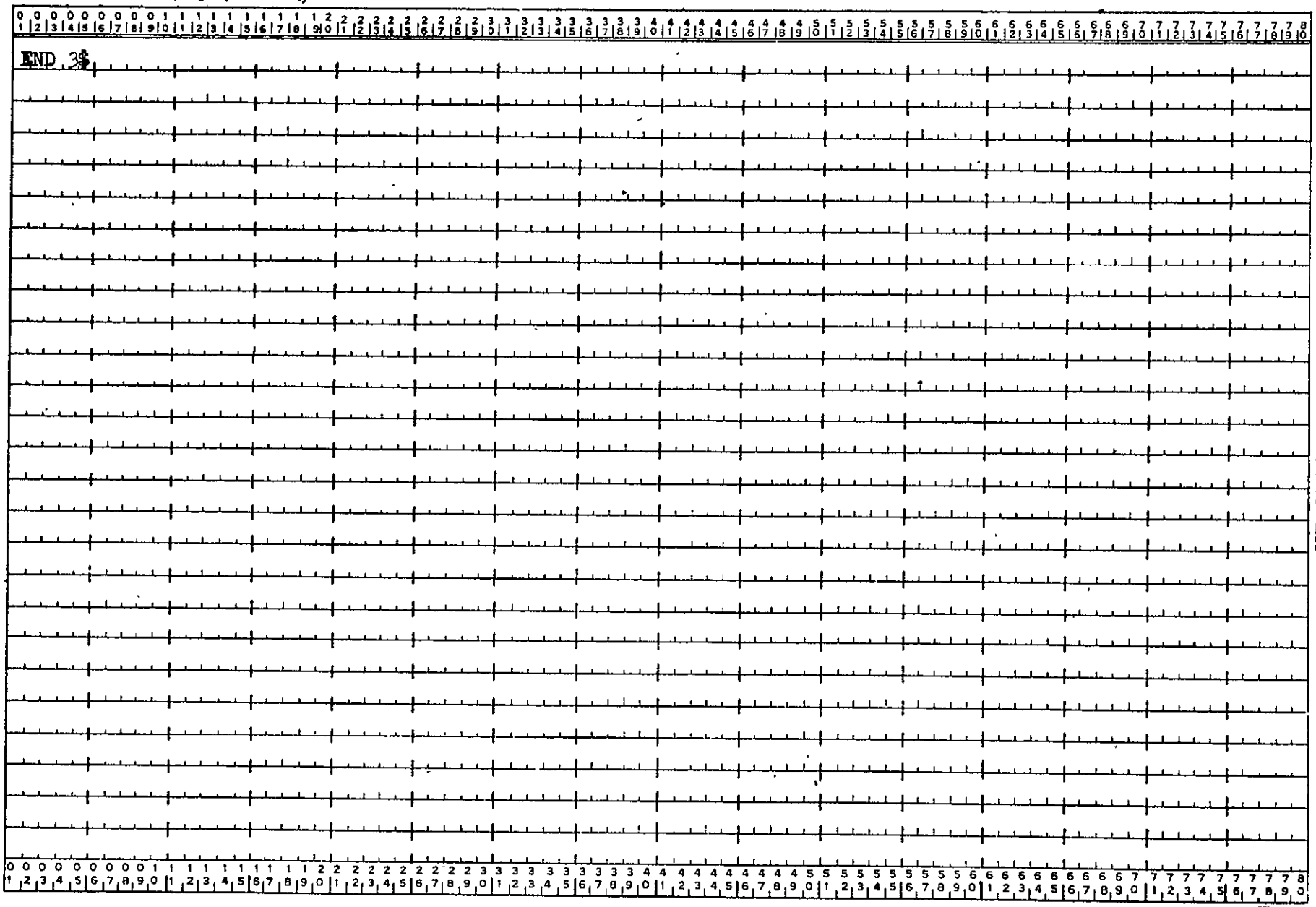


Figure 11 (Cont'd)

```
)0000000011111111112222222222333333333344444444455555555556666666667
12345678901234567890123456789012345678901234567890123456789012345678901234567890.
**************************************************************************
```

```
    1  0.6
  0.7
                                        APOLLO
     CODING EXAMPLE
    A1     0    B1    B2
    B1     1    C1    C2
    B2     2   Y01    C3
    C1     4   Y02   AAB    D1
    C2     5   Y03    02    01
    C3     1   X01   W01
    D1     0   X02   Z01
   W01    14
   X01    12
   X02    12
   Z01    13
    01    10
    02    10
   AAB    10
END1$
    A1                          UNDESIRED          EVENT
    B1                            EVENT              A
    B2                            EVENT              B
    C1                            EVENT              C
    C2                            EVENT              D
    C3                            EVENT              E
    D1                            EVENT              G
   W01                         NORMALLY          EXPECTED          EVENT
   X01                         PRIMARY           FAILURE
   X02                         PRIMARY           FAILURE
   Y01                         CONDITIONAL       EVENT
   Y02                         PRIORITY          DESCRIPTION
   Y03                         RESTRICTION       DESCRIPTION
   Z01                         SECONDARY         FAILURE
    01                            EVENT              I
    02                            EVENT              H
   AAB                            EVENT              F
END2$
    A1    20  AAA
    C3    20   01
    D1    20   02
END3$
```

Figure 12

Data Card Listing

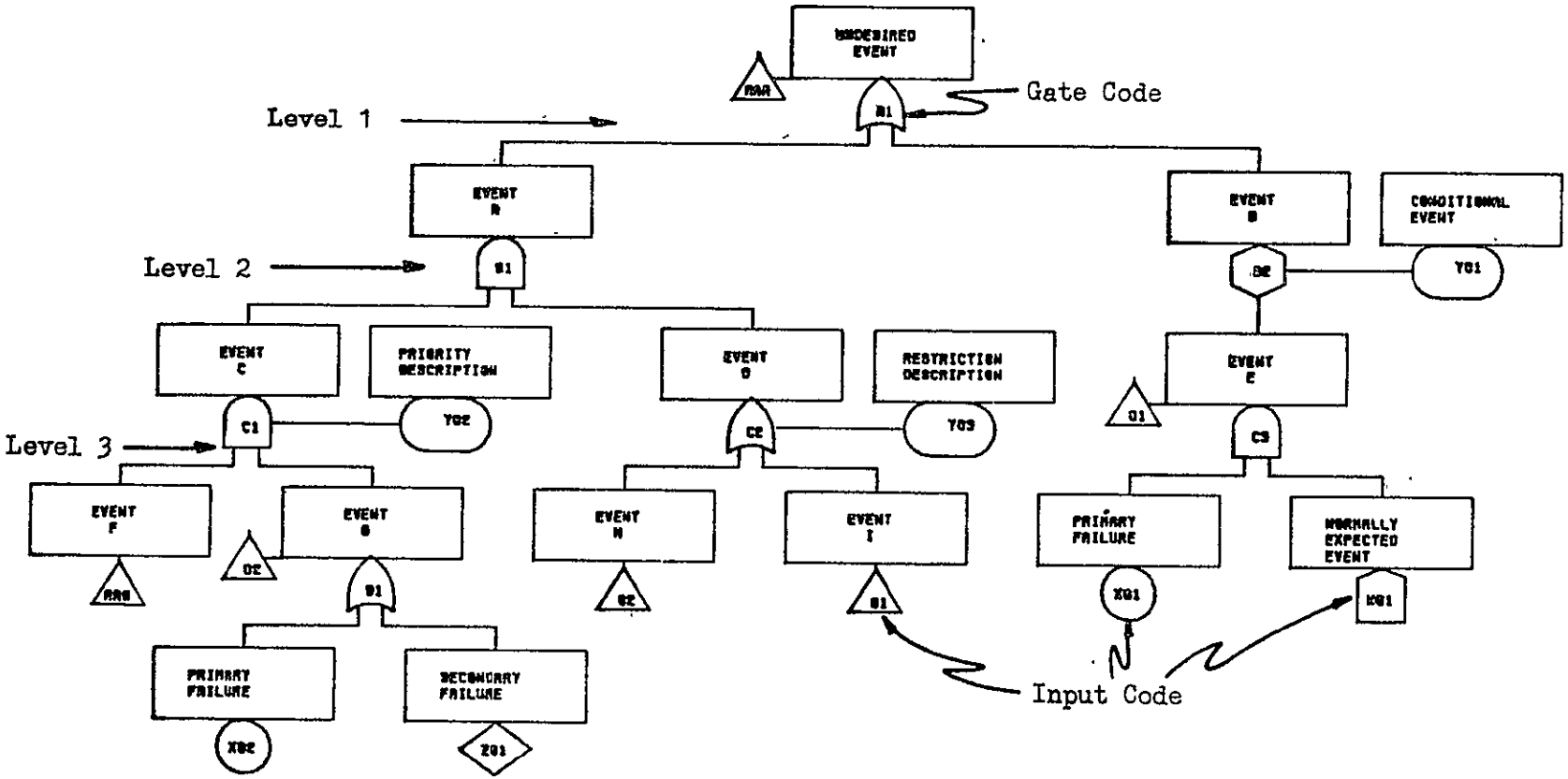


Figure 13

Example Logic Diagram

	INITIALS	DATE	REV BY INITIALS	DATE	TITLE	MODEL
CALC					CODING EXAMPLE	APOLLO
CHECK						
APPD						
APPD						
APPD						

REV LTR ____

BOEING | NO. | SH

APPENDIX A

SYSTEM SAFETY FAILURE DATA DEVELOPMENT

Failure data is developed as a tool to define the effects of various component failure modes and classify these effects on system equipment or personnel. The format in Figure A-1 is provided for assistance and guidance in developing system safety failure data. This format can be changed accord- to various requirements and should be considered as an example only.

The various columns are explained as follows:

COLUMN I - COMPONENT

Components are defined, at the discretion of the analyst, by their physical or functional significance. The following guide will help in defining the major components or facilitate understanding of the types of natural separations to consider. It is not intended to be exhaustive.

1) Electronic Logic Circuits

Many systems or subsystems are made up of a number of basic circuit designs which perform an identifiable purpose. These are used as building blocks for larger circuits designed to perform the required logic functions of the system or subsystem. To minimize the analysis required, the basic circuits can be defined as major components, and an analysis made of each logic function.

2) Mechanical Devices

Mechanical devices can be either a single part or an assembly of parts which perform one function. The use in the circuit will dictate to what level of detail mechanical parts should be considered. Single parts which can be considered major components are: solid driveshafts, engine blocks, primary structure, etc. The majority of mechanical devices will be assemblies of many parts and it is more reasonable to treat the assemblies as major components, for example: relays, pumps, motors, mechanical safety devices, etc. This permits the majority of vendor-supplied mechanical devices to be analyzed as major components.

3) Electrical Systems

Major components can be basic components of a circuit or combinations of components used to perform one

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS

FIGURE A1

System Safety Failure Data Format

single function such as amplifiers, rectifiers, or regulators. The level of data development should be based on the importance of the part as a functional element in the design.

4) Chemical Systems

In systems containing chemical compounds, the chemicals should be considered as major components if these compounds can cause failures of other components through chemical reaction or release of chemical energy. Examples of chemical components are: fuels, pressurants, coolants, and preservatives.

5) Safety Devices

Safety devices will normally be considered major components since they are used primarily to protect against undesired events.

6) Wiring

Interconnecting wiring of major components will be considered a major component. Internal wiring will be considered as a part of a major component. Physical characteristics of cables which circumvent failures between wires should be stated in the cable analysis.

COLUMN II - COMPONENT FAILURE MODE

Failures of major components consisting of one part require a listing of the modes in which that part may fail. Failures of major components consisting of more than one part will require a failure mode and effects analysis to determine how the failure modes of each part affect the components' output. These effects will be the failure modes of the major component listed in the system safety failure data. All failure modes of the component should be listed.

COLUMN III - COMPONENT FAILURE RATE

The predicted reliability or the failure rate computed from actual field data of primary failures should be tabulated in this column for each major component in each of its modes of failure. This data can be used in evaluating the probability of the fault event or in selecting which critical or catastrophic events should be analyzed if the decision is made not to analyze an event so classified. It also serves as a data bank for future reference when the need arises to analyze other undesired events as a result of system changes.

COLUMN IV - SOURCE OF DATA

This column states the source of the failure rate data. It shows the differentiation between field data, test data, calculated data, etc.

COLUMN V - FAILURE STATE

Many major components are only recurrently activated during the system's operational life. The level of stress on these components will change from one system mode to another. The effect of a failure in each mode can be different; for example, components supplied with power only during a test can create a fault hazard only while a test is performed. Failures existing in one mode of system operations can also adversely affect the system when the mode is changed. This column therefore should reflect the environmental state of the component when it failed.

COLUMN VI - EFFECT OF COMPONENT FAILURE

This column states the effect on related system equipment and/or personnel due to the component failure.

COLUMN VII - REMARKS

This column may be used to include additional information needed to clarify or verify information in other columns as well as other information currently pertinent to system safety efforts.

Appendix E shows the application of this data format to a specific example.

APPENDIX B

SYSTEM SAFETY LOGIC DIAGRAM EVALUATION

After the logic diagram has been constructed and input data acquired, the diagram can be evaluated. The object is to establish the likelihood of occurrence of the "undesired event" and to evaluate the relative contribution of each indicated failure mode. With this information the safety analyst can identify the dominant system failure modes (dominant paths) and management can make the decision as to whether or not corrective action is warranted.

There are two basic approaches used to quantify system safety logic diagrams. The two approaches are 1) calculation, and 2) simulation. The calculation or deterministic approach will be considered first. For logic diagrams where every basic input is non-repairable, classical probability can be used. In this case, each gate merely represents the operation to be performed(i.e., union for "OR" gates and intersection for "AND" gates). The classical probability approach, while simple and efficient, is not adequate for logic diagrams where the effect (s) of a basic failure can be eliminated before the defined mission length is revealed (e.g., a failed component is located and repaired). A basic failure whose effect can be removed is called repairable; however, the usage of the word "repairable" is irregular because the effect may be terminated without actually repairing or replacing the failed item. A more definitive time is "fault duration time" which will be used in the application of any quantitative solution to Apollo logic diagrams. The analysis of repairable systems requires special statistical techniques.

COMPUTATION

One technique in the calculation or deterministic approach is the "Lambda-Tau" method to evaluate logic diagrams, failure rates must be small, fault duration times must be small with regard to mission length, and redundant inputs must be removed. Redundancies that are not removed may lead to serious unbounded errors in the answer. The logic diagram is usually expressed algebraically and operated on by theorems of Boolean algebra to remove redundancies. The "Lambda-Tau" method can be applied by hand or by digital computer. However, as the logic diagrams get larger in size, the task of hand calculation becomes time consuming, laborious and error prone. A computer program can write the algebraic expression and can use Boolean algebra to remove the redundancies. However, computer core storage on most computers limits the size of the logic diagram solvable by this method.

Nevertheless, smaller logic diagrams can be calculated accurately by hand or computer using "Lambda-Tau" methods. (See Appendix C for further details.)

SIMULATION

In the simulation approach, a logic diagram is represented on a computer and failures are simulated over a given mission length. The computer prints out the failure which leads to the undesired event, and the probability is calculated. The simulation approach has all the advantages of the calculation approach except for the greater amount of computer time needed to simulate system safety logic diagrams with small probabilities. Simulation offers several advantages: namely, the dominant paths are listed and the computer can solve larger diagrams (10 times larger than "Lanbda-Tau". Simulation has gone through many stages of development. In its early stages, the amount of computer time required became prohibitive; however, special Monte Carlo variance reducing techniques (importance sampling) have reduced greatly the computer time required. The importance sampling technique distorts the true failure distribution to make events occur more rapidly. Thus, the number of trials (a trial represents the predefined mission length of the system) required for an acceptable statistical confidence is reduced. With fewer trials required, computer time is reduced. The distortion of the distribution, when using importance sampling, is compensated for by calculation weight factors. See Nagel, P.M., and Schroder, R. J., "The Efficient Simulation of Rare Events in Complex Systems", D2-114072-1. Overall, simulation offers more potential and had proven to be more effective in calculation accurate answers than the calculation method.

APPENDIX C

CONSTANT REPAIR OR "LAMBDA TAU" METHOD FOR LOGIC DIAGRAM EVALUATION

Coexistence of Failures

Suppose there is given a group of n repairable items, and these items may or may not fail in a given time period, T. Let event A_1 represent the failure of item 1, event A_2 the failure of item 2, and in general event A_i the failure of item i, i = 1, 2, . . . , n. These failures are chance failures, occurring at random and independent of each other. It is these chance failures which have an exponential distribution of their time to failure. Hence the probability that an item in that group will not fail may be expressed in

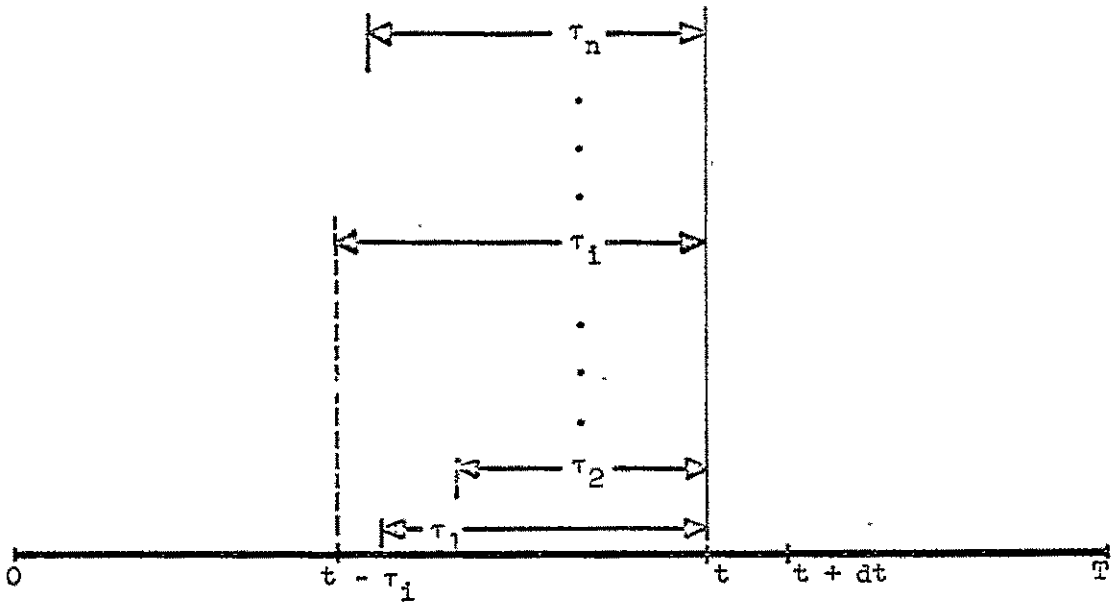
$$R_i(t) = e^{-\lambda_i t_i} \tag{1}$$

where t_i is the given time period, and λ_i is the number of failures per unit time. This may also be called the reliability of an item in that group. The unreliability or chance of failure is

$$Q_i(t) = 1 - R_i(t) = 1 - e^{-\lambda_i t_i} \tag{2}$$

This unreliability may also be called the probability that item i will fail. Hence, it expresses the probability that event A_i will happen. For each item i assume a constant failure rate λ_i and repair time τ_i. Further assume that τ_i/T is small, λ_i is small, and that $\lambda_i\tau_i$ is small.

Consider an interval of time from 0 to T as shown in Figure 1.



In order for a failure to exist in the small time interval, dt, the failure must occur either in the small interval, dt, or in some time interval previous to t, from $t - \tau_i$ to t. If the failure occurs before $t - \tau_i$, it will be repaired before it can exist in the dt interval; and if it occurs after t + dt, it cannot possibly exist in the dt interval. The probability of the event A_i happening some time in the τ_i time period is $(1 - e^{-\lambda_i \tau_i})$. The probability of event A_i happening in the dt time interval is $\lambda_i dt$. These are the only two ways in which the event A_i can happen. The probability for all events, A_1, A_2, ..., A_n to coexist in the dt interval is given by the formula

$$\lambda_1 dt(1-e^{-\lambda_2\tau_2})(1-e^{-\lambda_3\tau_3}) \ldots (1-e^{-\lambda_n\tau_n})$$

$$+ \lambda_2 dt(1-e^{-\lambda_1\tau_1})(1-e^{-\lambda_3\tau_3}) \ldots (1-e^{-\lambda_n\tau_n})$$

$$\vdots$$

$$+ \lambda_n dt(1-e^{-\lambda_1\tau_1})(1-e^{-\lambda_2\tau_2}) \ldots (1-e^{-\lambda_{n-1}\tau_{n-1}})$$

$$= H dt$$

Consider the first term in this formula, which expresses the probability that event A_1 occurs in the dt interval and coexists with the other failure having occurred previous to t. The probability of event A_1 occurring in dt is $\lambda_1 dt$, and the probability of the occurrence of A_2 in some time period τ_2 previous to t is $(1-e^{-\lambda_2 \tau_2})$. The product of these probabilities gives the probability of the coexistence of $A_1, A_2, \ldots, A_n$ where A_1 occurs during dt. The second term gives the probability of the coexistence of $A_1, A_2, \ldots A_n$ where event A_2 occurs during the time interval dt. The sum of these terms gives the total probability of n events coexisting in the dt interval.

Now let $f(t)$ be the probability that $A_1, A_2, \ldots A_n$ have not coexisted up to time t. Likewise $f(t + dt)$ expresses the probability that $A_1, A_2, \ldots A_n$ have not coexisted from time 0 to $t + dt$. This can be expressed as

$$f(t + dt) = f(t)\ (1 - Hdt)$$

Where $f(t + dt)$ is the product of the probability of no coexistence of the items $A_1, \ldots A_n$ from 0 to t, $f(t)$, and the probability of no coexistence of the items $A_1, \ldots A_n$ from time t to $t + dt$, $(1 - Hdt)$.

Now by definition, the differential of $f(t)$ is $f(t + dt) - f(t)$; therefore:

$$df(t) = f(t)\ (1 - Hdt) - f(t)$$

$$df(t) = - f(t)\ Hdt$$

and $$\frac{df(t)}{f(t)} = - Hdt$$

Solving this differential equation by integration,

$$\ln f(t) = - Ht + C$$

Substituting the condition that when $t = 0$, $f(t) = 1$, that is, at time zero, the probability that $A_1, A_2, \ldots A_n$ have not coexisted is equal to 1. Since $\ln(1) = 0$,

$$f(t) = e^{-HT} \tag{4}$$

The probability that events $A_1, A_2, \ldots A_n$ have coexisted at some time t is

$$P(A) = 1 - f(t) = 1 - e^{-HT}$$

For sufficiently small HT,

$$P(A) \sim HT \tag{5}$$

Hence $P(A) \sim HT = (\lambda_1\lambda_2\tau_2\lambda_3\tau_3 \cdots \lambda_n\tau_n$

$$+ \lambda_2\lambda_1\tau_1\lambda_3\tau_3 \cdots \lambda_n\tau_n$$

$$\vdots$$

$$+ \lambda_n\lambda_1\tau_1\lambda_2\tau_2 \cdots \lambda_{n-1}\tau_{n-1})T \tag{6}$$

$$= \lambda_1\lambda_2 \cdots \lambda_n(\tau_2\tau_3 \cdots \tau_n + \tau_1\tau_3 \cdots \tau_n + \cdots + \tau_1\tau_2 \cdots \tau_{n-1})T$$

"AND" GATE λ

The form of the probability figure for the coexistence of failures A_1, A_2, . . . A_n, suggests that the failure rate for all these events is

$$\lambda_n = \lambda_1\lambda_2 \cdots \lambda_n(\tau_2\tau_3 \cdots \tau_n + \tau_1\tau_3 \cdots \tau_n + \cdots \tau_1\tau_2 \cdots \tau_{n-1})$$

"AND" GATE τ

Let us consider a situation in which events A_1, A_2, . . . A_{n+1} coexist and find the duration time τ_n for the coexistence of events A_1, A_2, ..., A_n. Let λ_n be the failure rate and τ_n the effective duration time of the coexistence of failures A_1, A_2, . . . A_n.

$$\lambda_n\lambda_{n+1}(\tau_n + \tau_{n+1}) = \lambda_1\lambda_2 \cdots \lambda_{n+1}(\tau_2\tau_3 \cdots \tau_{n+1} + \cdots \tau_1\tau_2 \cdots \tau_n).$$

Since $\lambda_n = \lambda_1\lambda_2 \cdots \lambda_n(\tau_2\tau_3 \cdots \tau_n + \tau_1\tau_3 \cdots \tau_n + \cdots \tau_1\tau_2 \cdots \tau_{n-1})$

Then

$$\lambda_1\lambda_2 \cdots \lambda_n(\tau_2\tau_3 \cdots \tau_n + \tau_1\tau_3 \cdots \tau_n + \cdots \tau_1\tau_2 \cdots \tau_{n-1})\ \lambda_{n+1}\ (\tau_n + \tau_{n+1}) =$$

$$\lambda_1\lambda_2 \cdots \lambda_{n+1}(\tau_2\tau_3 \cdots \tau_{n+1} + \tau_1\tau_3 \cdots \tau_{n+1} + \cdots + \tau_1\tau_2 \cdots \tau_n)$$

Therefore

$$\tau_n = \frac{\tau_1\tau_2 \cdots \tau_n}{\tau_2\tau_3 \cdots \tau_n + \tau_1\tau_3 \cdots \tau_n + \tau_1\tau_2 \cdots \tau_{n-1}} = \frac{1}{\frac{1}{\tau_1} + \frac{1}{\tau_2} + \cdots + \frac{1}{\tau_n}}$$

by mathematical induction.

"OR" GATE λ

Considering the same group of n items i = 1, 2, ... n in a given time period T, the probability that none of the events occurs during this time period is given by

$$R_i(t) = e^{-\lambda_1 T} e^{-\lambda_2 T} e^{-\lambda_3 T} \cdots e^{-\lambda_n T}$$

$$R_i(t) = e^{-(\lambda_1 + \lambda_2 + \cdots + \lambda_n) T}$$

Hence the probability that any one of the events occurs is

$$Q_i(t) = 1 - R_i(t) = 1 - e^{-(\lambda_1 + \lambda_2 + \cdots + \lambda_n)T}$$

Hence the failure rate for the occurrence of any event in the time interval is $\lambda_U = \lambda_1 + \lambda_2 + \cdots + \lambda_n$ from the general form of the reliability equation.

"OR" GATE τ

To find effective duration time for the condition that any one of the group of items may fail in the time period, consider the following example. Let any one of the events $A_1, A_2, \ldots A_n$ coexist with an event A_{n+1}. Let λ_U and τ_U represent respectively the failure rate and effectivity time obtained from the union of events A_1 to A_n, when event A_1 or A_2 or $A_3, \ldots A_n$ occur in the given time interval. If these events $A_1, A_2, \ldots A_n$ occur with event A_{n+1}, the result is $\lambda_U \lambda_{n+1} (\tau_U + \tau_{n+1})$ from the coexistence of failures discussion, and

$$\lambda_u\lambda_{n+1}(\tau_u + \tau_{n+1}) = \lambda_1\lambda_{n+1}(\tau_1 + \tau_{n+1}) + \lambda_2\lambda_{n+1}(\tau_2 + \tau_{n+1}) + \cdots$$

$$+ \lambda_n\lambda_{n+1}(\tau_n + \tau_{n+1})$$

Since $\lambda_u = \lambda_1 + \lambda_2 + \cdots + \lambda_n$

Then

$$(\lambda_1 + \lambda_2 + \cdots + \lambda_n)\lambda_{n+1}(\tau_u + \tau_{n+1}) = \lambda_1\lambda_{n+1}(\tau_1 + \tau_{n+1}) + \lambda_2\lambda_{n+1}(\tau_2 + \tau_{n+1}) +$$

$$\cdots + \lambda_n\lambda_{n+1}(\tau_n + \tau_{n+1})$$

Hence

$$\tau_u = \frac{\lambda_1\tau_1 + \lambda_2\tau_2 + \cdots + \lambda_n\tau_n}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}$$

The outputs of the AND and OR gates are given in tabular form at the end of this paper.

FAILURES OCCURING IN A GIVEN ORDER

The probability expression for n items failing in an interval of time in a given order, will be derived in the following discussion, and the $\lambda\tau$ approximation will be proved valid for small $\lambda\tau$.

Suppose there is given a group of n items, $A_1, A_2, \ldots A_n$, each working at the beginning of an arbitrary interval of time, τ. Let $\lambda_1, \lambda_2, \ldots \lambda_n$ be the respective failure rates of the n items, and suppose that $\lambda_1\tau, \ldots \lambda_n\tau$ are very small. Let E be the event which occurs when $A_1, A_2, \ldots A_n$ all fail in some specified order, e.g., A_1 occurs, then A_2, then A_3, etc., then

$$P(E) \sim \frac{\lambda_1\lambda_2 \cdots \lambda_n\tau^n}{n!}$$

PROOF:

The above approximation will be proved by induction. After proving it is valid for n = 1, it will be valid for n if it is valid for n - 1. If $\lambda\tau$ is small then for all $0 < t \leq \tau$, $\lambda > 0$ and $\varepsilon > 0$

$$\left| 1 - e^{-\lambda\tau} - \lambda t \right| < \varepsilon\lambda t$$

where ε is very small. In other words, $1 - e^{-\lambda\tau}$ is closely approximated by $\lambda\tau$ when the difference of the two expressions is very small. The following discussion shows that ε is very small. Since by a Taylor series expansion, $e^{-\lambda\tau} = 1 - \lambda\tau + \frac{\lambda^2 t^2}{2} - \frac{\lambda^3 t^3}{3!} + \cdots$

$$\left| 1 - e^{-\lambda\tau} - \lambda\tau \right| = \left| \lambda\tau - \frac{\lambda^2 t^2}{2} + \frac{\lambda^3 t^3}{3!} - \cdots - \lambda t \right|$$

$$= \left| -\frac{\lambda^2 t^2}{2} + \frac{\lambda^3 t^3}{3!} - \frac{\lambda^4 t^4}{4!} + \cdots \right|$$

$$= \lambda t \left| \frac{\lambda t}{2} - \frac{\lambda^2 t^2}{3!} + \frac{\lambda^3 t^3}{4!} + \cdots \right|$$

$$\leq \lambda t \left| \frac{\lambda t}{2} + \frac{\lambda^2 t^2}{3!} + \frac{\lambda^3 t^3}{4!} + \cdots \right| \leq \lambda\tau \left| \frac{\lambda\tau}{2} + \frac{\lambda^2 \tau^2}{3!} + \cdots \right|$$

Let $\varepsilon = \frac{\lambda\tau}{2} + \frac{\lambda^2 \tau^2}{3!} + \frac{\lambda^3 \tau^3}{4!} + \cdots$

An upper bound for ε is

$$\frac{\lambda\tau}{2} + \left(\frac{\lambda\tau}{2}\right)^2 + \left(\frac{\lambda\tau}{2}\right)^3 + \cdots = \frac{\lambda\tau}{2} \left[\frac{1}{1 - \frac{\lambda\tau}{2}} \right]$$

which is in itself a very small number since $\lambda\tau$ was chosen to be very small.

Now let $g_i(t)$ be the true probability that i events have occurred up to time t in a specified order, e.g., A_1 occurs, then A_2, then A_3, etc.

Then let $f_i(t) = 1 - g_i(t)$ be the true probability that i events have not occurred up to time t in order. Assume the approximation is very good for n-1. This may be expressed in the following form:

$$\left| g_{n-1}(t) - \frac{\lambda_1 \cdot \cdot \cdot \lambda_{n-1}}{(n-1)!} t^{n-1} \right| < \frac{\epsilon\lambda_1 \cdot \cdot \cdot \lambda_{n-1}}{(n-1)!} t^{n-1} \qquad (8)$$

where t is a very small number.

Herafter $\overline{\lambda}_k = \lambda_1 \lambda_2 \cdot \cdot \cdot \lambda_k$

Now let $f_n(t + dt)$ express the probability that n events have not occurred up to time $t + dt$ in some specified order. That is

$$f_n(t + dt) = f_n(t) \left[1 - g_{n-1}(t) \lambda_n dt \right]$$

where $f_n(t + dt)$ is the product of the two following probabilities; the probability that n-1 events have not occurred up to time t in order and the probability that the n^{th} event occurred during dt.

The expression $g_{n-1}(t)$ is the probability that n-1 events have occurred up to time t in order, and $(\lambda_n dt)$ is the probability that event A_n occurs during dt. The product of these probabilities gives the probability that the n^{th} event occurs during dt. The probability that the n^{th} event does not occur during dt is given by

$$1 - g_{n-1}(t)\,\lambda_n dt$$

By definition of a differential, $df_n(t) = f_n(t + dt) - f_n(t)$

$$df_n(t) = f_n(t) \quad - g_{n-1}(t)\lambda_n dt$$

$$\frac{df_n(t)}{f_n(t)} = - g_{n-1}(t)\,\lambda_n dt$$

$$\ln f_n(t) = - \lambda_n \int_o^t g_{n-1}(s)ds$$

$$\ln f_n(t) = - \lambda_n \int_o^t g_{n-1}(s) - \frac{\bar{\lambda}_{n-1} s^{n-1}}{(n-1)!} + \frac{\bar{\lambda}_{n-1} s^{n-1}}{(n-1)!}\, ds$$

$$\ln f_n(t) = - \lambda_n \left[\int_o^t g_{n-1}(s) - \frac{\bar{\lambda}_{n-1} s^{n-1}}{(n-1)!}\, ds + \frac{\bar{\lambda}_{n-1} t^n}{n!} \right] \quad (9)$$

Using equation 8 and the notation $\bar{\lambda}_{n-1} = \lambda_1 \cdots \lambda_{n-1}$ yields

$$- \frac{\epsilon\bar{\lambda}_{n-1} s^{n-1}}{(n-1)!} < g_{n-1}(s) - \frac{\bar{\lambda}_{n-1} s^{n-1}}{(n-1)!} < \frac{\epsilon\bar{\lambda}_{n-1}}{(n-1)!}\, s^{n-1}$$

Integrating this inequality yields

$$-\frac{e\bar{\lambda}_{n-1}t^n}{n!} < \int_0^t g_{n-1}(s) - \frac{\bar{\lambda}_{n-1}s^{n-1}}{(n-1)!} \, ds < \frac{c\bar{\lambda}_{n-1}t^n}{n!}$$

Postulate a variable $r(t)$ such that $-\epsilon < r(t) < \epsilon$

$$\int_0^t g_{n-1}(s) - \frac{\bar{\lambda}_{n-1}s^{n-1}}{(n-1)!} \, ds \sim \frac{r(t)\bar{\lambda}_{n-1}t^n}{n!}$$

Substituting this value into equation (9) yields

$$\ln f_n(t) = -\lambda_n \left[\frac{r(t)\bar{\lambda}_{n-1}t^n}{n!} + \frac{\bar{\lambda}_{n-1}t^n}{n!} \right]$$

and

$$f_n(t) = e^{-\frac{\bar{\lambda}_n t^n}{n!}\left[r(t)+1\right]}$$

Since $g_i(t) = 1 - f_i(t)$

$$g_n(t) = 1 - f_n(t) = 1 - e^{\frac{-\bar{\lambda}_n t^n}{n!}\left[r(t)+1\right]}$$

For sufficiently small λt, so that the exponent of the above expression will be small

$$g_n(t) \sim \frac{\bar{\lambda}_n t^n}{n!}\left[r(t)+1\right]$$

This may also be expressed as follows

$$\left| 1-e^{-\frac{\bar{\lambda}_n t^n}{n!}\left[r(t) + 1 \right]} - \frac{\bar{\lambda}_n t^n}{n!} \left[r(t) + 1 \right] \right| < \delta \frac{\bar{\lambda}_n t^n}{n!} \left[r(t) + 1 \right]$$

or

$$\left| g_n(t) - \frac{\bar{\lambda}_n t^n}{n!} \left[r(t) + 1 \right] \right| < \frac{\delta \bar{\lambda}_n t^n}{n!} \left[r(t) + 1 \right] < \frac{\delta \bar{\lambda}_n t^n (c + 1)}{n!}.$$

where $\delta > 0$ is very small. Hence

$$- \frac{\delta \bar{\lambda}_n t^n (\epsilon + 1)}{n!} < g_n(t) - \frac{\bar{\lambda}_n t^n}{n!} \left[r(t) + 1 \right] < \frac{\delta \bar{\lambda}_n t^n}{n!} (c + 1).$$

Expanding the above expression yields

$$- \frac{\delta \bar{\lambda}_n t^n (\epsilon + 1)}{n!} < g_n(t) - \frac{\bar{\lambda}_n t^n r(t)}{n!} - \frac{\bar{\lambda}_n t^n}{n!} < \frac{\delta \bar{\lambda}_n t^n}{n!} (\epsilon + 1)$$

$$- \frac{\delta \bar{\lambda}_n t^n}{n!} (\epsilon + 1) - \frac{\epsilon \bar{\lambda}_n t^n}{n!} < g_n(t) - \frac{\bar{\lambda}_n t^n}{n!} < \frac{\delta \bar{\lambda}_n t^n}{n!} (c + 1) + \frac{\epsilon \bar{\lambda}_n t^n}{n!}$$

$$\left| g_n(t) - \frac{\bar{\lambda}_n t^n}{n!} \right| < \frac{\delta \bar{\lambda}_n t^n}{n!} (\epsilon + 1) + \frac{\bar{\lambda}_n t^n \epsilon}{n!}$$

Factoring the right hand side of the inequality yields

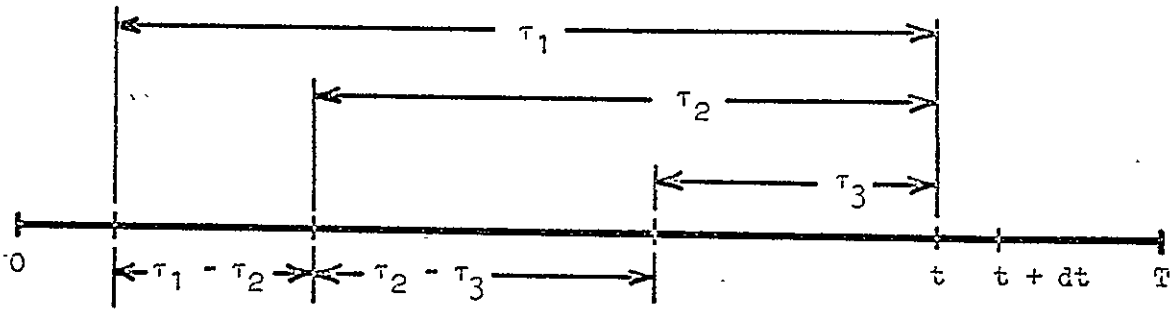
$$\left| g_n(t) - \frac{\bar{\lambda}_n t^n}{n!} \right| < \frac{\bar{\lambda}_n t^n}{n!} \left[\delta(1 + \epsilon) + \epsilon \right]$$

Since δ and ϵ were chosen to be very small, the right half of the inequality is very small, and hence

$$\varepsilon_n(t) \sim \frac{\overline{\lambda}_n t^n}{n!} \qquad \text{Q. E. D.}$$

In previous discussion, the expression $\frac{\lambda_1 \cdot \lambda_2 \cdot \lambda_3 \cdot \ldots \lambda_n \tau^n}{n!}$ was obtained for the probability of occurrence of n events $A_1, A_2, \ldots A_n$ in a particular order over a time period τ. Using these results, the probability will now be obtained for the occurrence of four events in order over a time period T when repair times are unequal.

Let four events A_1, A_2, A_3, A_4 have respective repair times $\tau_1, \tau_2, \tau_3, \tau_4$ and failure rates $\lambda_1, \lambda_2, \lambda_3, \lambda_4$. Let the magnitudes of the repair times have the relationship, $\tau_1 > \tau_2 > \tau_3 > \tau_4$, as shown below



For this particular example event A_1 shall occur first, then A_2, then A_3, then A_4. Events A_1, A_2, and A_3 shall occur prior to t and event A_4 shall occur in the dt interval. The probability of A_4 occurring in the dt interval is $\lambda_4 dt$. To coexist with A_4 in the dt interval, A_1, A_2, and A_3 can occur in the five following ways:

a. A_1 occurs in interval $\tau_1 - \tau_2$, A_2 occurs in interval $\tau_2 - \tau_3$, and A_3 occurs in interval τ_3

$P(a) = \lambda_1(\tau_1 - \tau_2)\lambda_2(\tau_2 - \tau_3)\lambda_3\tau_3$

b. A_1 occurs in interval $\tau_1 - \tau_2$ and A_2 and A_3 both occur in order in interval τ_3

$$P(b) = \lambda_1(\tau_1 - \tau_2) \frac{\lambda_2\lambda_3\tau_3^2}{2}$$

c. A_1 and A_2 both occur in order in the interval $\tau_2 - \tau_3$ and A_3 occurs in the interval τ_3

$$P(c) = \frac{\lambda_1\lambda_2(\tau_2 - \tau_3)^2}{2} \lambda_3\tau_3$$

d. A_1 occurs in interval $(\tau_2 - \tau_3)$ and A_2 and A_3 occur in order in the interval τ_3

$$P(d) = \lambda_1(\tau_2 - \tau_3) \frac{\lambda_2\lambda_3\tau_3^2}{2}$$

e. A_1, A_2 and A_3 all occur in order in the interval τ_3

$$P(e) = \frac{\lambda_1\lambda_2\lambda_3\tau_3^3}{6}$$

The total probability, $P(t)$, for the occurrence of A_1, A_2, A_3 in order is the sum of these probabilities

$$P(t) = P(a) + P(b) + P(c) + P(d)$$

$$= \lambda_1\lambda_2\lambda_3 \left(\tau_1\tau_2\tau_3 - \frac{\tau_1\tau_3^2}{2} - \frac{\tau_2^2\tau_3}{2} + \frac{\tau_3^3}{6}\right)$$

The product of $P(t)$ and $\lambda_4 dt$ therfore, gives the probability that A_1, A_2, A_3, A_4 occur in the given order and coexist for the first time in the dt interval. If $f(t)$ is defined as the probability that A_1, A_2, A_3, A_4 have not occurred up to time t in a given order, and $f(t + dt)$ is the probability of A_1, A_2, A_3, and A_4 have not occurred up to time $t + dt$ in a given order, then

$$f(t + dt) = f(t) \quad (1 - P(t)\ \lambda_4 dt)$$

Since $P(t)\ \lambda_4 dt$ gives the probability that A_1, A_2, A_3, A_4 occur in the given order, $1 - P(t)\lambda_4 dt$ gives the probability that they do not occur as specified.

$$f(t + dt) - ft = - f(t)P(t)\lambda_4 dt$$

$$df(t) = - f(t)P(t)\lambda_4 dt$$

$$\frac{df(t)}{f(t)} = - P(t)\lambda_4 dt$$

$$\ln f(t) = - P(t)\lambda_4 \int_o^t dt = - P(t)\lambda_4 T$$

$$f(t) = e^{- P(t)\lambda_4 T}$$

$$1-f(t) = 1-e^{- P(t)\lambda_4 T}$$

If $P(T)\lambda_4 T$ is small, then the probability of the occurrence of this chain of events over time T, $P(1234)$ is

$$P(1234) = \lambda_1\lambda_2\lambda_3\lambda_4 \left(\tau_1\tau_2\tau_3 - \frac{\tau_1\tau_3^2}{2} - \frac{\tau_2^2\tau_3}{2} + \frac{\tau_3^3}{6}\right)T$$

By similar manipulations, the probability for the occurrence of A_1, A_2, A_3, A_4 in that order is

$$P_{(2134)} = \lambda_1\lambda_2\lambda_3\lambda_4 \left[\frac{\tau_2^2\tau_3}{2} - \frac{\tau_2\tau_3^2}{2} + \frac{\tau_3^3}{6} \right] T$$

Similarly

$$P_{(2314)} = \lambda_1\lambda_2\lambda_3\lambda_4 \left[\frac{\tau_2\tau_3^2}{2} - \frac{\tau_3^3}{2} + \frac{\tau_3^3}{6} \right] T$$

$$P_{(3214)} = \lambda_1\lambda_2\lambda_3\lambda_4 \frac{\tau_3^3}{6} T$$

$$P_{(3124)} = \lambda_1\lambda_2\lambda_3\lambda_4 \frac{\tau_3^3}{6} T$$

$$P_{(1324)} = \lambda_1\lambda_2\lambda_3\lambda_4 \left[\frac{\tau_1\tau_3^2}{2} - \frac{\tau_3^3}{2} + \frac{\tau_3^3}{6} \right] T$$

The sum of these probabilities is

$$P(4) = \lambda_1\lambda_2\lambda_3\lambda_4 (\tau_1\tau_2\tau_3) T$$

If A_3 is the last event, τ_4 takes the place of τ_3 on the figure and the resulting probability is

$$P(3) = \lambda_1\lambda_2\lambda_3\lambda_4 (\tau_1\tau_2\tau_4) T$$

Similarly if A_2 and A_1 are respectively the last events, the associated probabilities are

$$P(2) = \lambda_1\lambda_2\lambda_3\lambda_4(\tau_1\tau_3\tau_4)\ T$$

$$P(1) = \lambda_1\lambda_2\lambda_3\lambda_4(\tau_2\tau_3\tau_4)\ T$$

These probabilities may be added (P(1), P(2), P(3), and P(4) mutually exclusive) giving the total probability of the coexistence of A_1, A_2, A_3, and A_4.

$$P = \lambda_1\lambda_2\lambda_3\lambda_4\ (\tau_1\tau_2\tau_3 + \tau_1\tau_2\tau_4 + \tau_1\tau_3\tau_4 + \tau_2\tau_3\tau_4)\ T$$

It is to be noted that this is equivalent to the coexistence formula. Thus, the probability for the coexistence of events can be obtained as the sum of the probabilities of each ordered chain of events.

"λ τ' OMBINATION

			2 INPUTS	3 INPUTS	n INPUTS
AND	τ's UNEQUAL	λ_n	$\lambda_1\lambda_2(\tau_1+\tau_2)$	$\lambda_1\lambda_2\lambda_3(\tau_2\tau_3+\tau_1\tau_3+\tau_1\tau_2)$	$\lambda_1\lambda_2\cdots\lambda_n(\tau_2\tau_3\cdots\tau_n+\tau_1\tau_3\cdots\tau_n+\cdots+\tau_1\tau_2\cdots\tau_{n-1})$
		τ_n	$\frac{\tau_1\tau_2}{\tau_1+\tau_2}$	$\frac{\tau_1\tau_2\tau_3}{\tau_2\tau_3+\tau_1\tau_3+\tau_1\tau_2}$	$\frac{1}{\frac{1}{\tau_1}+\frac{1}{\tau_2}+\cdots+\frac{1}{\tau_n}}$
	τ's EQUAL	λ_n	$2\lambda_1\lambda_2\tau$	$3\lambda_1\lambda_2\lambda_3\tau^2$	$n\lambda_1\lambda_2\cdots\lambda_n\tau^{n-1}$
		τ_n	$\frac{\tau}{2}$	$\frac{\tau}{3}$	$\frac{\tau}{n}$
OR	τ's UNEQUAL	λ_u	$\lambda_1+\lambda_2$	$\lambda_1+\lambda_2+\lambda_3$	$\lambda_1+\lambda_2+\cdots+\lambda_n$
		τ_u	$\frac{\lambda_1\tau_1+\lambda_2\tau_2}{\lambda_1+\lambda_2}$	$\frac{\lambda_1\tau_1+\lambda_2\tau_2+\lambda_3\tau_3}{\lambda_1+\lambda_2+\lambda_3}$	$\frac{\lambda_1\tau_1+\lambda_2\tau_2+\cdots+\lambda_n\tau_n}{\lambda_1+\lambda_2+\cdots+\lambda_n}$
	τ's EQUAL	λ_u	$\lambda_1+\lambda_2$	$\lambda_1+\lambda_2+\lambda_3$	$\lambda_1+\lambda_2+\cdots+\lambda_n$
		τ_u	τ	τ	τ

APPENDIX D

ADVANCED CONCEPTS IN THE USAGE OF THE MATRIX GATE

INTRODUCTION

In logic diagram analysis of systems and subsystems many fault events are used repeatedly in order to denote the proper sequence of logic leading to an undesired event. Frequently the redundant fault events are related to one another by a second fault event, resulting in a unique combination of events. When these combinations are expressed by conventional fault tree techniques, the result is usually long and repetitive. The Matrix Gate is a method by which logic diagram construction is simplified with reference to permutations of redundant (or similar) fault events.

It must be emphasized that the Matrix Gate is not a unique logic operator in logic diagram analysis techniques. The Matrix Gate is merely a simplified or abbreviated representation of an already existing portion of logic diagram; the existing portion of a logic diagram being a series of two-input AND gates (with related inputs) summed together by an OR gate.

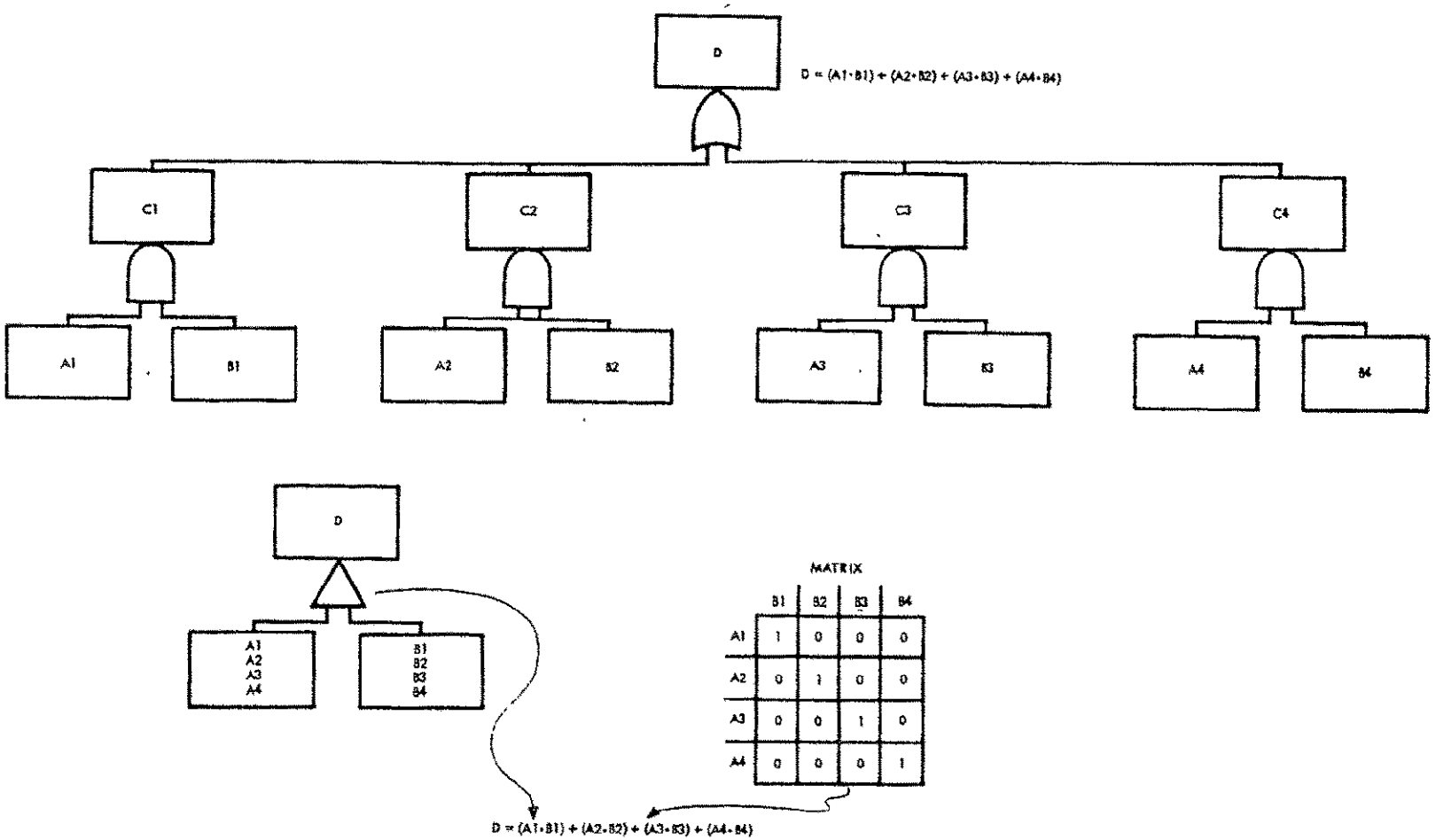
Whenever the Matrix Gate is used it is accompanied by a matrix, whose elements are the redundant (or similar) fault events. This matrix is necessary in order to denote which combination of events are applicable to the analysis, the total number of combinations, and the probability of a particular combination resulting in the undesired event.

In order for the Matrix Gate to meet all possible situations it is necessary for two types of gate to exist; the variable type Matrix Gate and the conditional type Matrix Gate. The variable type gate handles situations where both of the inputs to the gate consist of fault events (fault events being referred to as variables). The conditional type gate handles situations where one input consists of fault events (variable) and the other input consists of conditional events.

Example 1 (Figure D1) is a generalized case using the variable type Matrix Gate. Fault events A1, A2, A3 and A4 are unique but similar and fault events B1, B2, B3 and B4 are unique but similar. The Boolean Expression derived from the sample fault tree agrees with the Boolean expression extracted from the Matrix Gate and its concomitant matrix.

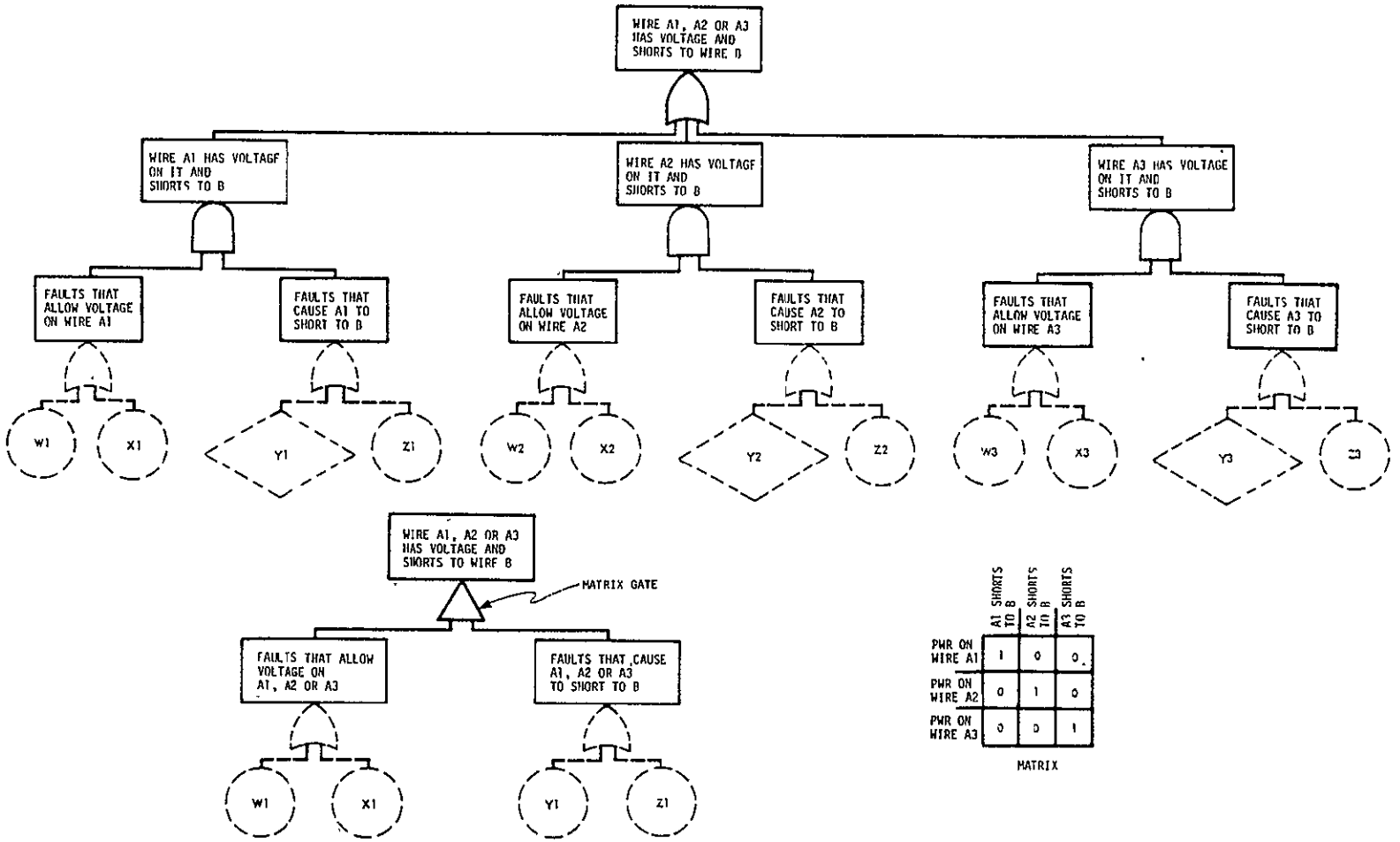
VARIABLE TYPE MATRIX GATE

Example 2 (Figure D2) is a typical problem in which a four-wire cable is to be analyzed. The wires are identified as A1, A2, A3 and B. Under standby operating conditions, assume that none of these wires carry voltage, and furthermore, that wire B is an ordnance line and wires A1, A2 and A3 carry



Example 1

Figure D1
Generalized Matrix Gate



Example 2

Figure D2
Variable Matrix Gate

voltage at certain discrete time intervals. The undesired event is wire A1, A2, or A3 shorting to wire B and at the same time having voltage on it from a fault condition at the voltage source.

In this example the events which cause wire A1 to short to wire B will be similar to the events which cause wire A2 to short to wire B and wire A3 to short to wire B. For example, they could be shorts caused by an insulation failure or a primary wire failure. Therefore, the fault conditions of these three wires are unique, yet similar. Since they are similar, they are drawn only once with the Matrix Gate, instead of three times under conventional techniques.

The fault events which allow power onto wire A1 may or may not be similar to the events which allow power onto wire A2 or A3, depending upon the circuitry involved. If the fault events are similar (or the same) the Matrix Gate can be utilized easily, with the fault event drawn only once. However, if the fault events are completely different for each wire, the Matrix Gate becomes more complex, and each distinct fault event must be drawn (with little saving over conventional techniques). Since the circuitry at the voltage source is not developed in this example, an assumption will be made that the faults are similar for each wire.

The 3 x 3 matrix drawn in Example 2 points out the combinations of interest in this particular analysis. The boxes which contain a "one" are the combinations of concern. These boxes, figuratively speaking, say that "the faults allowing power on wire A1" are ANDED with "the faults causing wire A1 to short to wire B", and "the faults allowing power on wire A2" are ANDED with "the faults causing wire A2 to short to wire B", and "the faults allowing power on wire A3" are ANDED with "the faults causing wire A3 to short to wire B" which are all summed together by an OR gate.

The significance of using a Matrix Gate in Example 2 may not be readily apparent, but suppose the four-wire cable had been a 50 wire cable. Instead of drawing 50 iterations of wire shorts combined with faults allowing power on the wire, the Matrix Gate requires only one iteration of the combination. The tediousness of drawing and reading superfluous information has been eliminated, yet the necessary information is not lost.

CONDITION TYPE MATRIX GATE

Example 2 demonstrated the Matrix Gate with both of the inputs as variables. That is, both of the inputs to the gate consisted of fault conditions. A second, and slightly different, way of using the Matrix Gate is with one input as a variable and the other input as a condition. This type of usage is

fitted for situations whereby the Matrix Gate is employed to replace Inhibit Gates which have similar or redundant inputs. Example 3 depicts this type of usage.

Example 3 (Figure D3) deals with a car and highway situation. In this example a car is analyzed for the undesired event "car wreck" and the only failure modes being considered are: 1) blowout, 2) loss of steering, and 3) brakes locking. In addition to analyzing the car to determine the causes of these failure modes, certain road conditions are placed on each failure mode. These conditions are: 1) the road being wet, 2) the road being dry, and 3) the road being icy.

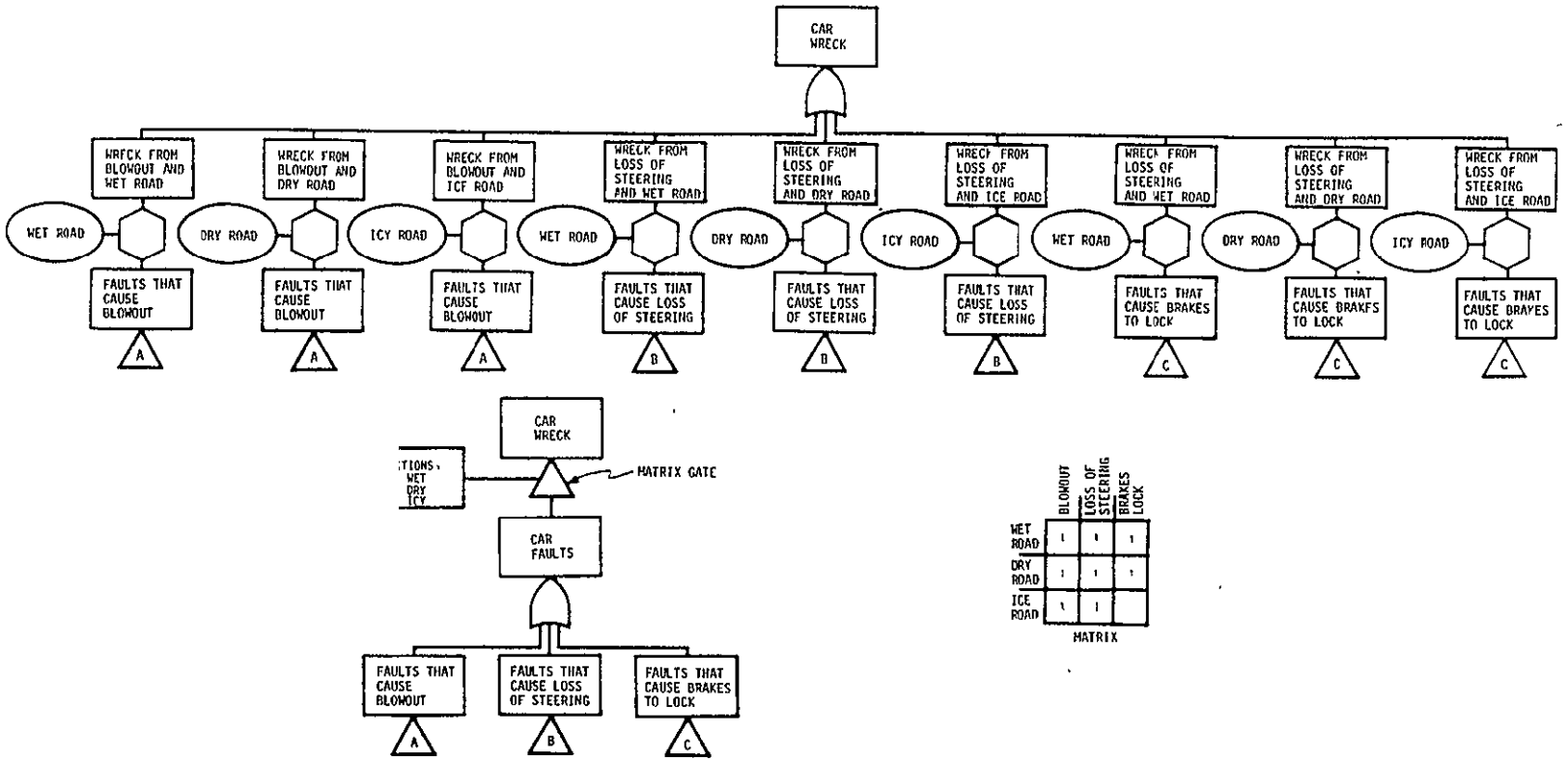
As is apparent from the logic diagram shown in Example 3, the variable inputs to the Inhibit Gates are redundant, and result in a unique set of combinations. This unique set of combinations results in a long and repetitious fault tree, which can be effectively reduced in size and complexity as shown.

The 3x3 matrix shown in Example 3 demonstrates that nine unique, but related combinations result from this particular example. Furthermore, it shows which fault event is combined with which conditional event, and the number of times each event is combined.

THE MATRIX

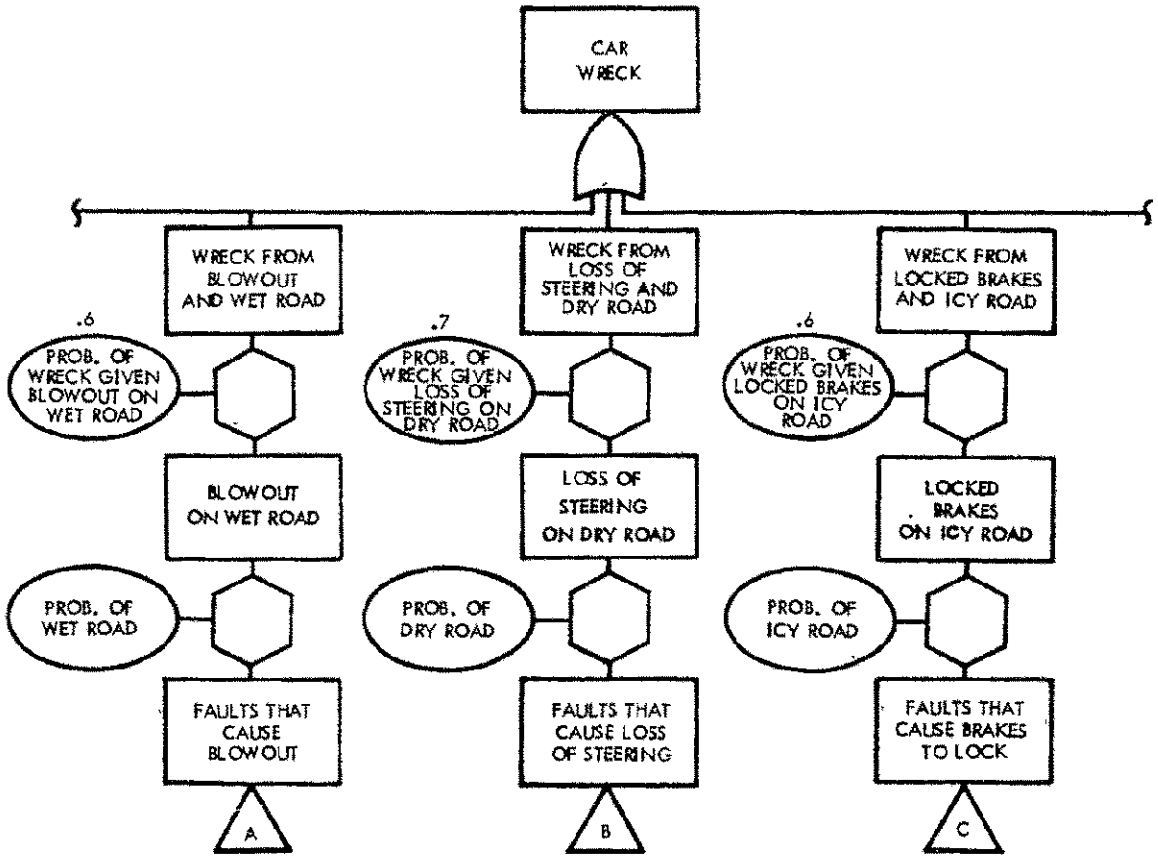
Now that the Matrix Gate has been exemplified in a simple a and concise manner, a small adjustment factor must be introduced. This adjustment factor involves the "one" and "zero" placed inside the boxes of the matrices. These numbers are, in actuality probability numbers which represent the probability of an Inhibit Gate allowing each combination (of fault events) to result in the undesired event. To be specific, an Inhibit Gate is located between each AND gate combination and the summing OR Gate. This "hidden" Inhibit Gate does not appear in the fault trees of Examples 1, 2, and 3 because the probability of a particular combination resulting in the undesired event has been assumed as one or zero. When the probability was zero for a certain combination this meant that the combination was either impossible or not desired for analysis. When the probability was one for a certain combination this meant that when the two events occurred, the undesired event was immediately realized. The probability of the combination resulting in the end event is not always one or zero, but frequently some value in-between.

Example 4 (Figure D4) is a continuation of Example 3, except the "hidden" Inhibit Gate is shown in the logic diagram. This example demonstrates the probability involved for realizing a car wreck given that a car fault occurs and the



Example 3

Figure D3
Condition Matrix Gate



	BLOWOUT	LOSS OF STEERING	BRAKES LOCK
WET ROAD	.6	.8	.5
DRY ROAD	.4	.7	.4
ICY ROAD	.8	.9	.6

MATRIX

Figure D4
Condition Matrix Gate

Example 4

appropriate road condition is fulfilled. Take for example the fault tree path "blowout on a wet road". When a blowout occurs and the road is wet it does not necessarily follow that there will be a car wreck. There is a certain probability involved for a blowout on a wet road to result in a wreck, and this probability is represented by an Inhibit Gate condition. The probability of this condition is placed inside the matrix which accompanies the Matrix Gate.

The probability numbers in the matrix should not be taken as the probability of two fault events being combined together. These numbers indicate the probability that two combined fault events will result in the undesired event after they have statistically been combined. Example 5 (Figure D5) shows the generalized case and the mathematical equations involved.

CONCLUSION

The preceeding discussion provides evidence that the Matrix Gate and its concomitant matrix successfully represent a condition of similar or redundant fault event combinations in a simple and concise form while at the same time yielding all of the qualitative information involved.

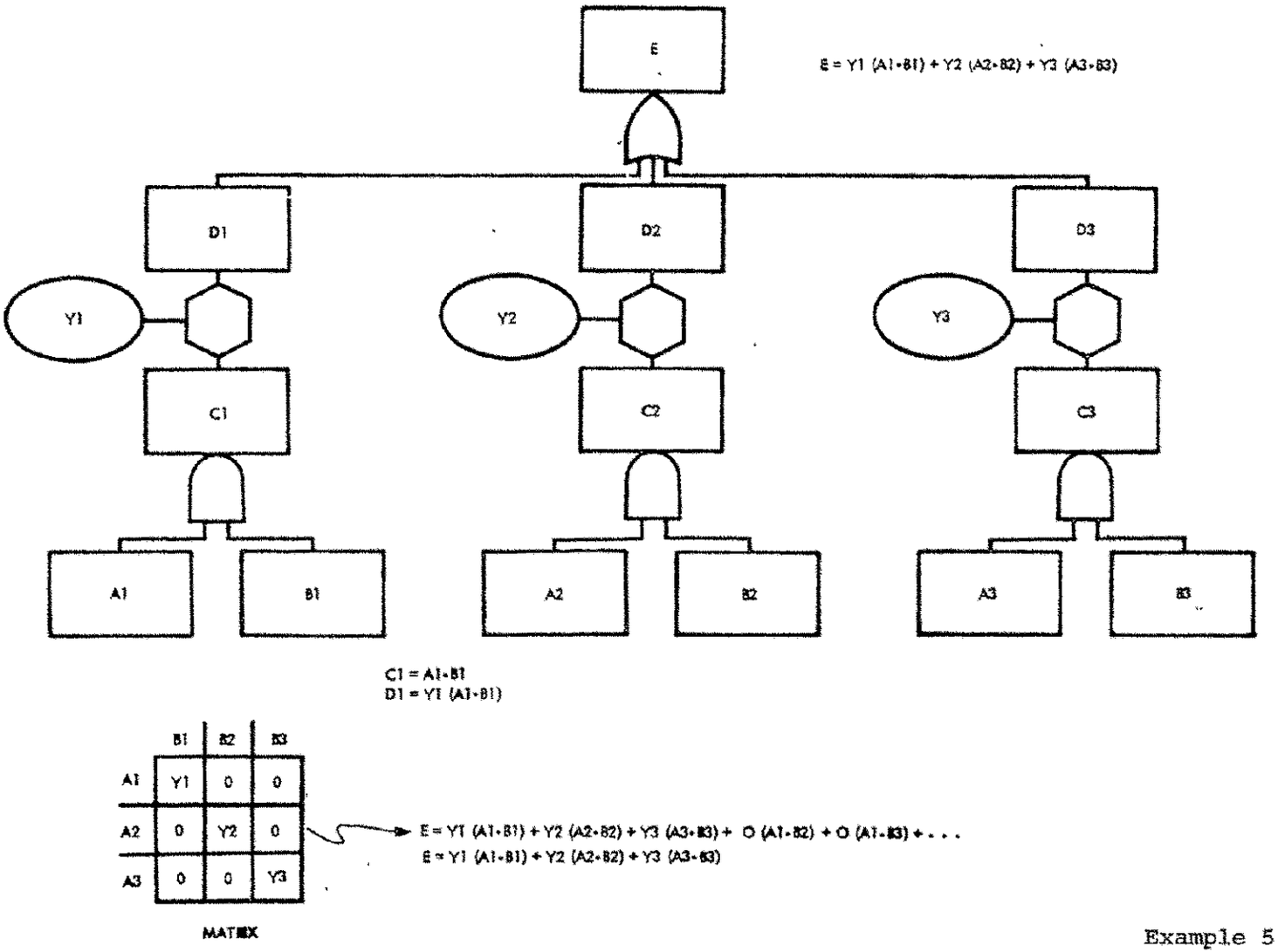


Figure D5

Generalized Matrix Gate - Mathematical Equations

APPENDIX E

LOGIC DIAGRAM EXAMPLE

The following analysis is an example of logic diagramming, notation and the use of logic diagram symbols. This example analysis is made on a rocket launching system, composed of the following manufactured subsystems: control panel, power and signal distribution box, and rocket. The system has been analyzed by a logic diagram for the undesired event of "Inadvertent Rocket Ignition."

Figure E1 is essentially a block diagram which describes the components in the subsystem, and their relationship to each other. Figure E2 is an electrical schematic of the system and from this the method of system operation can be derived.

To launch the rocket in a normal operating manner, switches S1 and S2 on the control panel are operated simultaneously. Switch S1 is moved to the "arm" position and switch S2 is moved to the "launch" position. Then, in exactly five minutes the rocket is ignited (launched). By moving switch S2 to the "inhibit" position within five minutes after switch S1 has been moved to "arm" the launch can be inhibited, but not after five minutes. Also, switch S1 can be moved from "arm" to "safe" (within five minutes) in order to inhibit the launch. When positive voltage is applied to terminal 1 and ground to terminal 2 of the "arm & monitor switch" motor, the motor drives the arm switch closed and the monitor switch open. In order to open the arm switch and close the monitor switch, positive voltage must be applied to terminal 2 of the motor and ground to terminal 1.

When positive voltage is applied to the coil in the "five minute sequencer" a timing device is initiated in the stepper switch (position 1) five minutes after voltage has been applied to the coil, as long as the voltage is continuous. If, in the five-minute period, voltage is discontinued, the timer will stop and automatically readjust to zero time (position 2).

When voltage is applied to the coil in the "delay gate" the switch is instantaneously closed. If voltage to the coil is terminated the switch will open.

The light on the control panel is the monitoring device. This light is on continuously and extinguishes when the arm switch is closed. When the light goes out the control panel operator knows that the arm switch is closed and that the five-minute sequencer stepper switch will be closed in five minutes. The only other alternative when the monitor

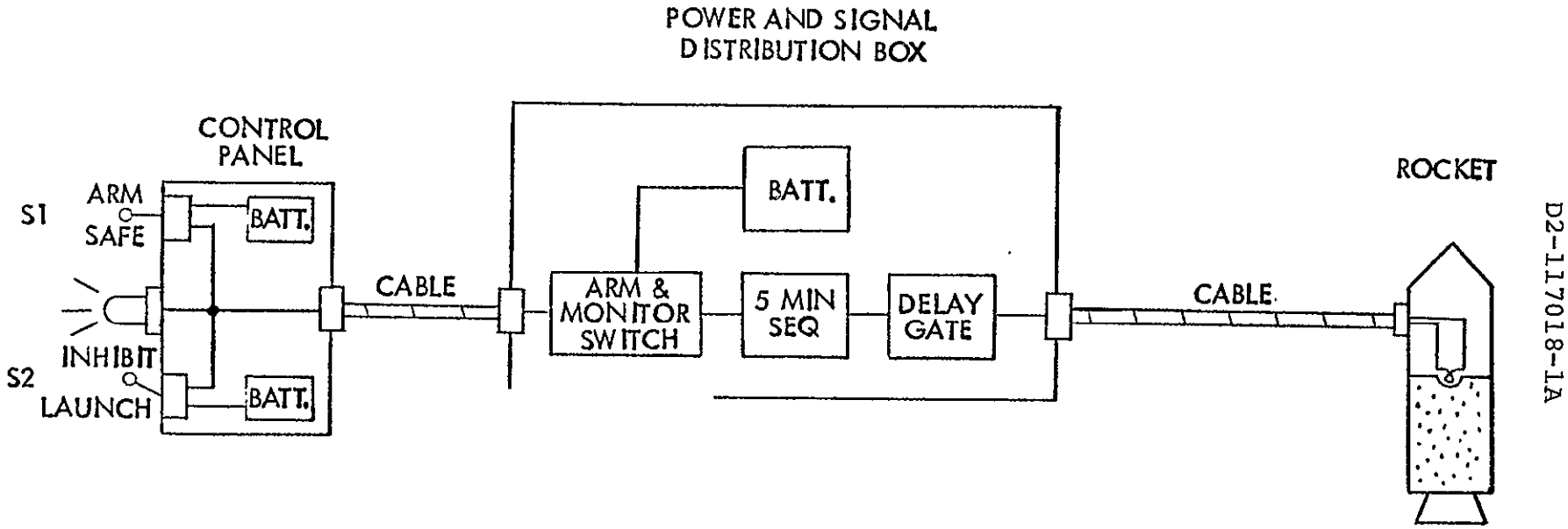


Figure E1
System Block Diagram

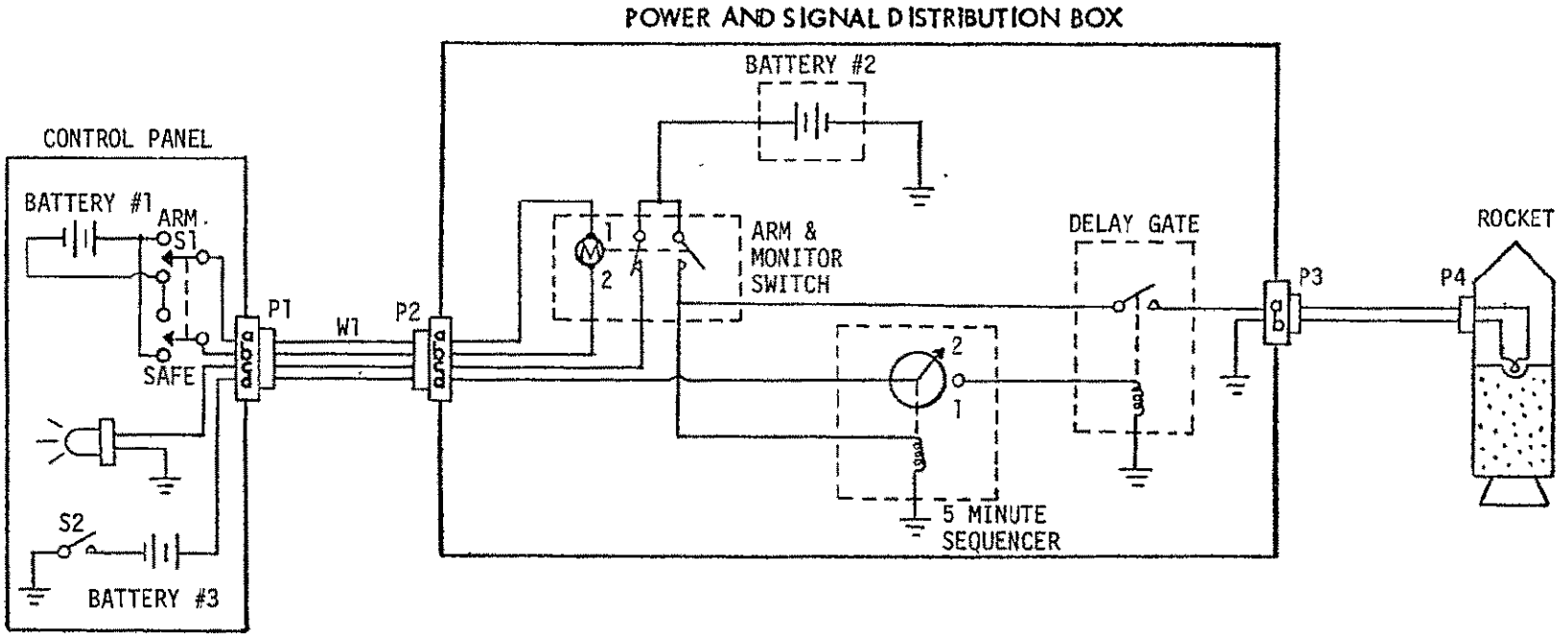


Figure E2
System Schematic

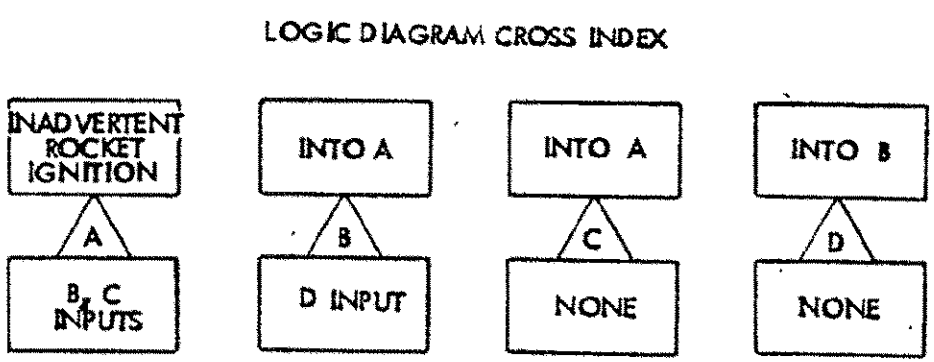
light goes out is that battery #2 on the battery connecting circuitry has failed.

Figure E3 shows two methods for denoting the relationship of the logic diagrams constructed for the undesired event "Inadvertent Rocket Ignition." These two methods merely denote which logic diagrams are an input into another logic diagram and is a cross index of the entire set of diagrams.

Figures E4 through E11 are the logic diagrams for the rocket launching system. These diagrams use most of the symbols of logic diagramming and the alpha-numeric system of identifying input events. Figures E4 through E7 are hand drawn logic diagrams using a basic system of event identification. Figures E8 through E11 are the same logic diagrams as E4 through E7 respectively except they are mechanically drafted and use the event identification method described in section 4.5.

The data charts (pages E-14 through E-20) shown following the logic diagrams represent an application of these charts to an example problem. The failure rate (MTBF) and "source of data" information columns contain information which was arbitrarily selected to show the ranges of data and types of sources which could be expected. All other columns, however, represent exact information related to the system of this example.

Column definitions and general discussion of the data chart can be found in Appendix A.



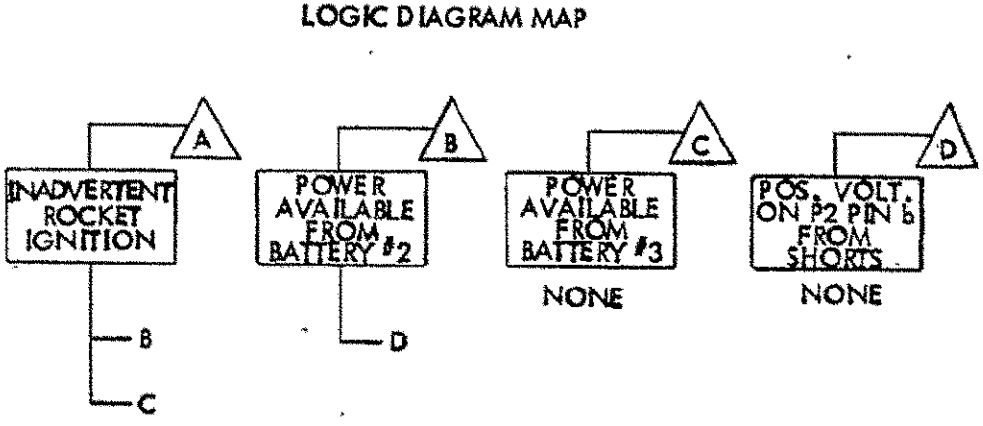


Figure E3
Logic Diagram Cross Reference

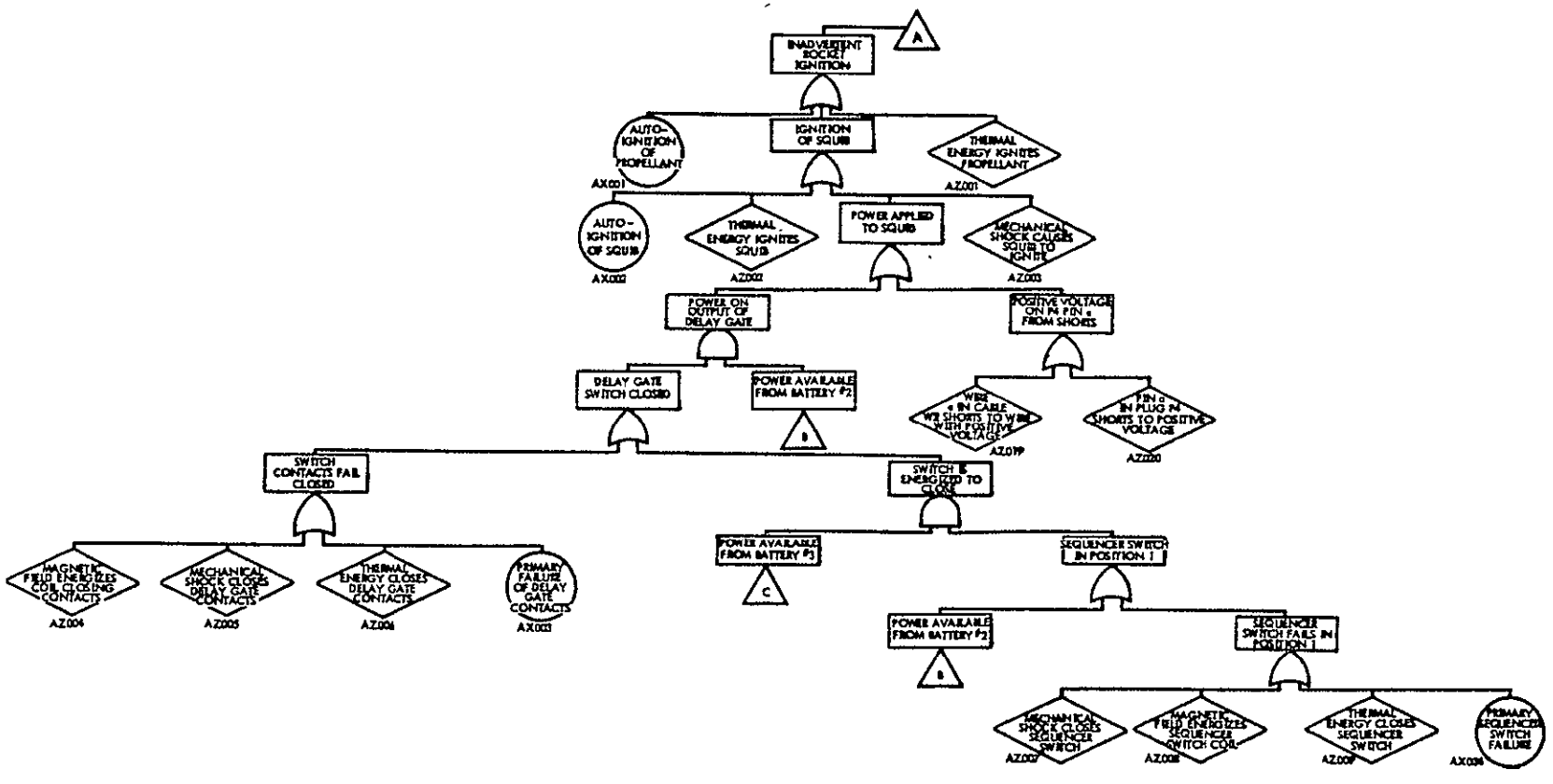


Figure E4
Hand Drawn Logic Diagram

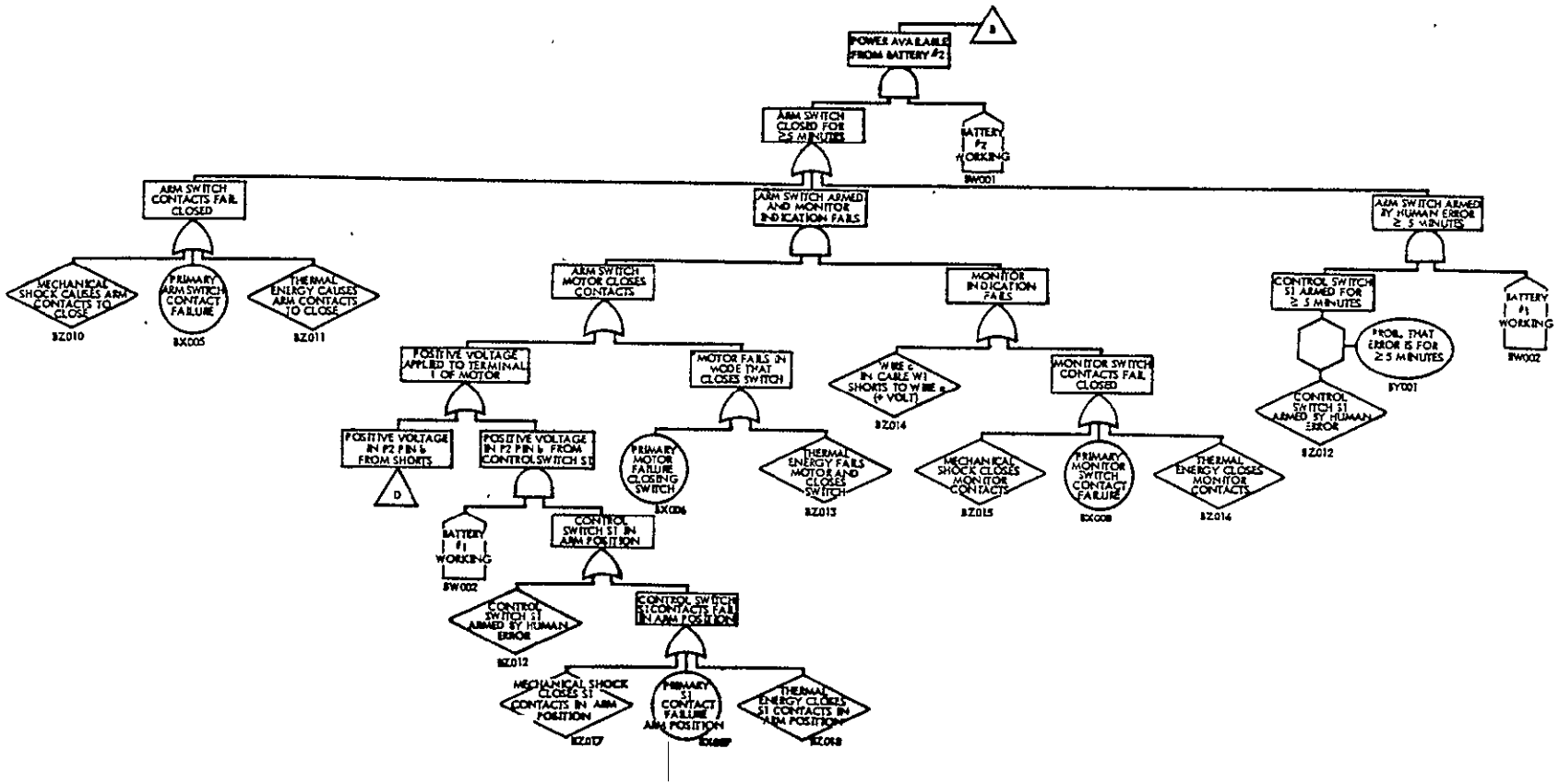


Figure E5
Hand Drawn Logic Diagram

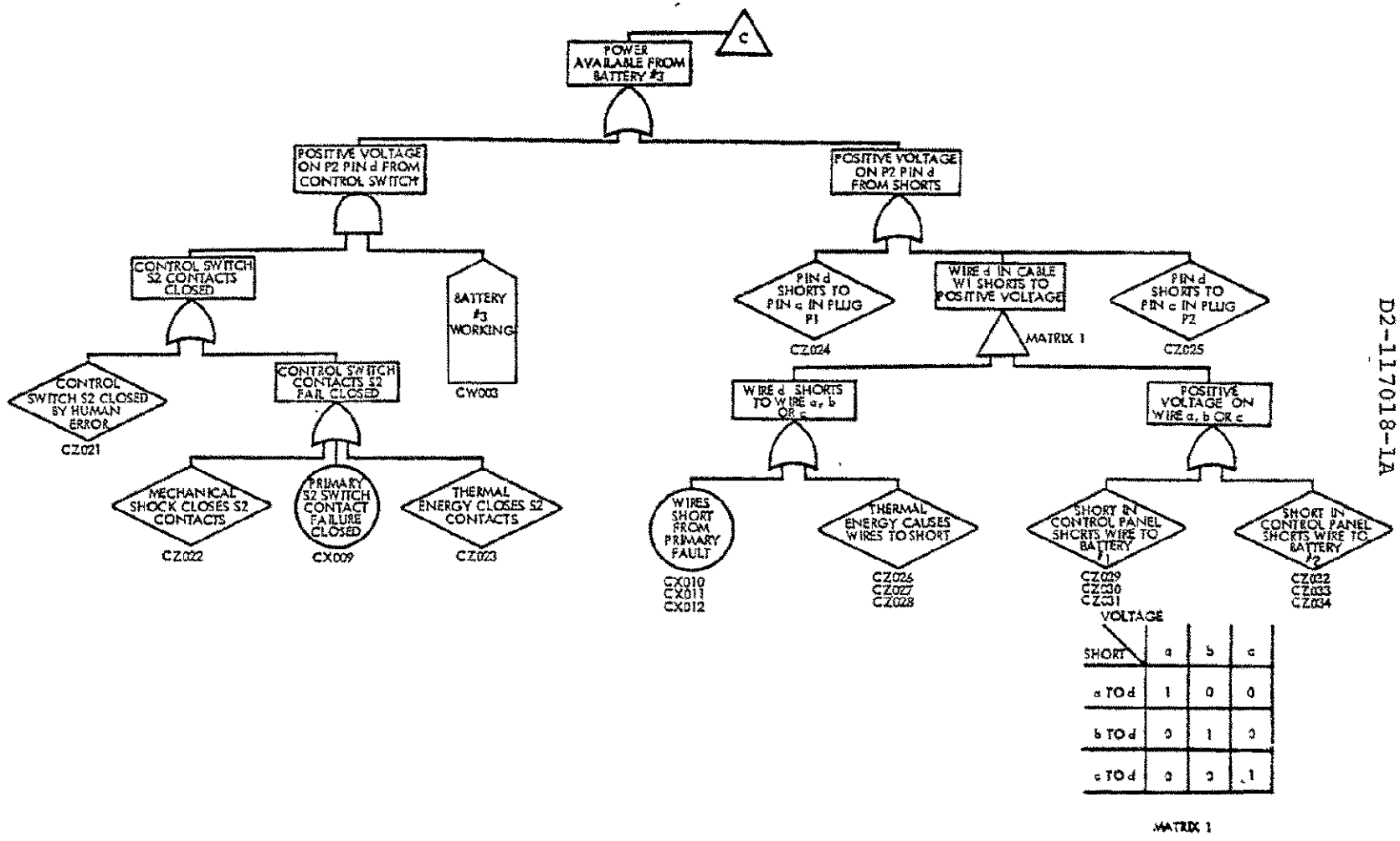
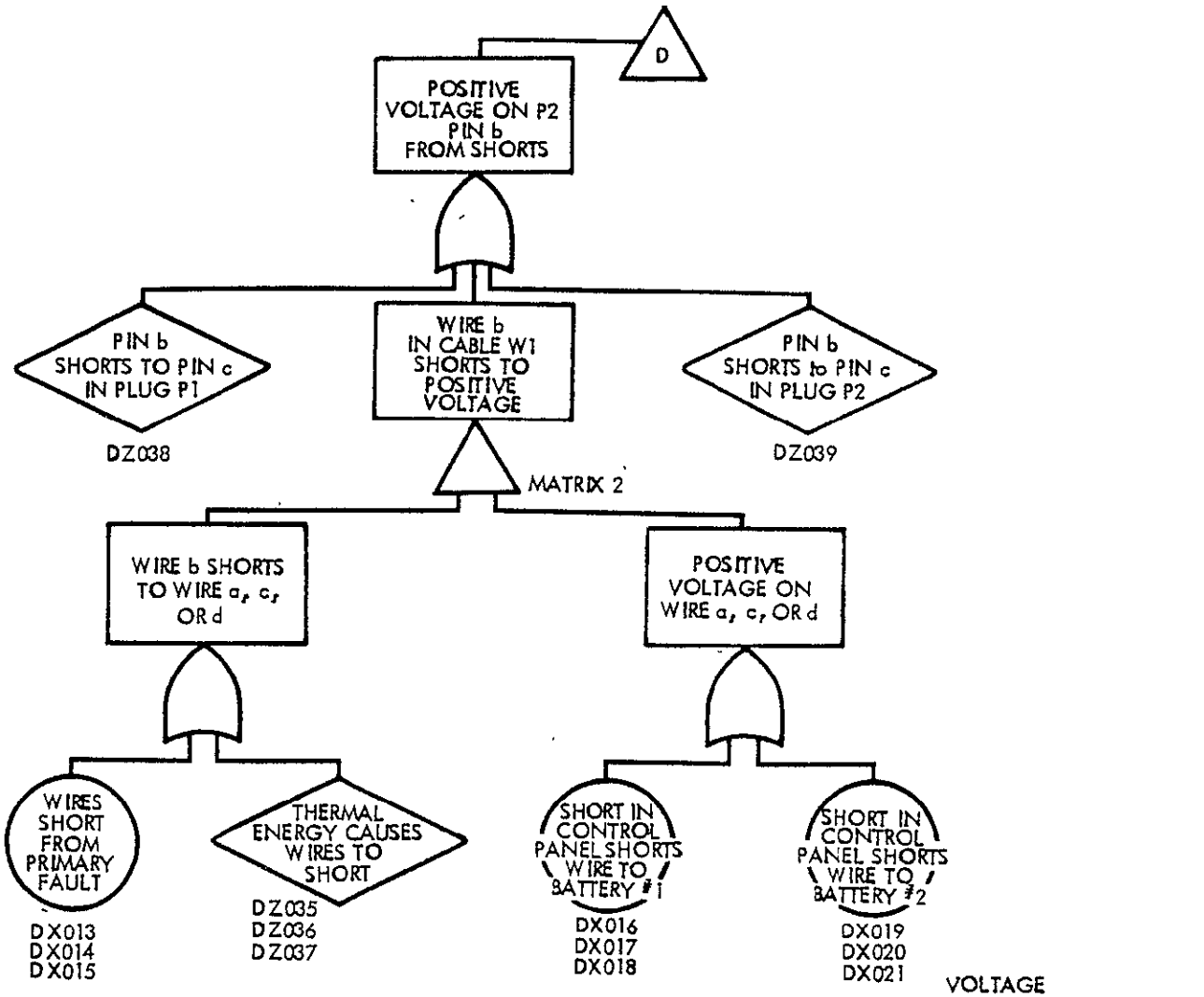


Figure E6
Hand Drawn Logic Diagram



VOLTAGE

SHORT	a	b	c
a TO b	1	0	0
c TO b	0	1	0
d TO b	0	0	1

MATRIX 2

Figure E7
Hand Drawn Logic Diagram

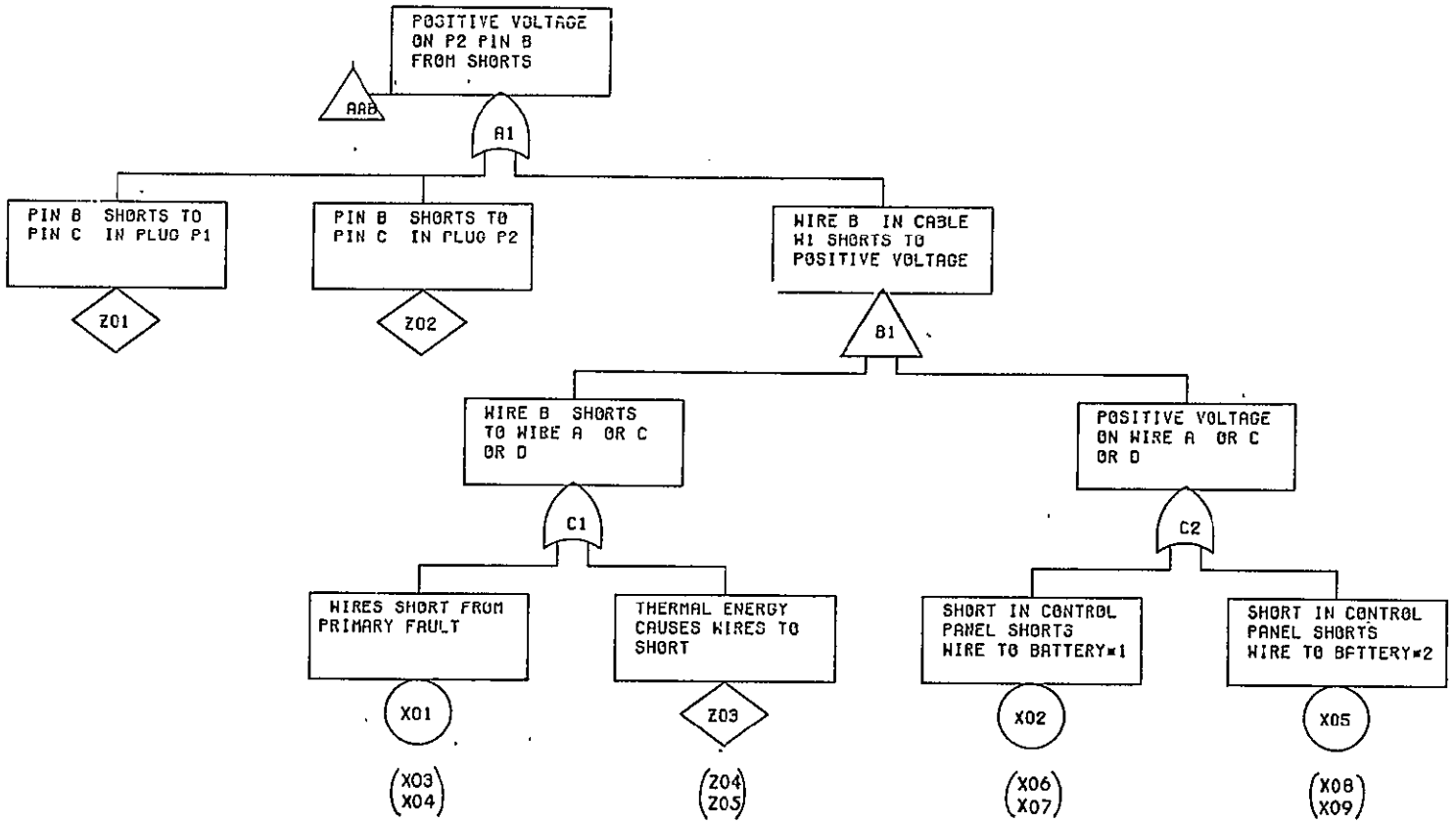


Figure E10

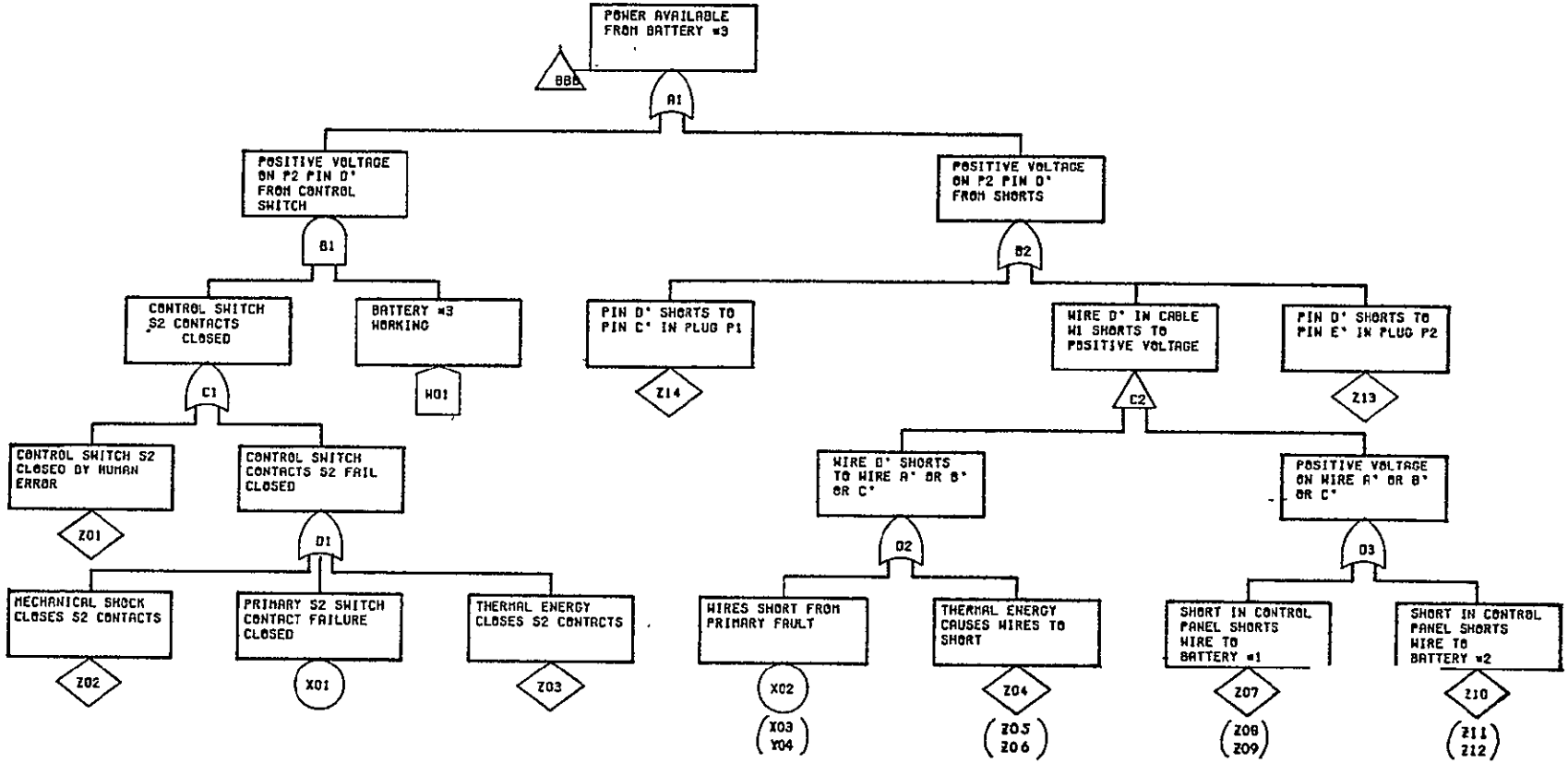
Machine Drafted Diagram

	INITIALS	DATE	REV BY INITIALS	DATE	TITLE	MODEL
CALC					AAB-POSITIVE VOLTAGE ON P2 PIN B FROM SHORTS	APOLLO
CHECK						
APPD						
APPD						
APPD					AAB	

4/9/6

REV LTR _____

BOEING | NO. D2-170181 SH



	INITIALS	DATE	REV BY INITIALS	DATE	TITLE	MODEL
CALC					BBB--POWER AVAILABLE FROM BATTERY =3	APOLLO
CHECK						
APPD						
APPD						
APPD					BBB	

REV LTR

BOEING WA. D2-117018-1 SH

Figure E11

Machine Drafted Diagram

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
Switch S1	Contacts fail open	MTBF 7×10^9 Hr	Field Experience	Standby	Arm & Monitor (A&M) Switch Motor cannot be activated.	System is disabled
	Contacts fail closed - arm position	MTBF 4×10^7 Hr	Field Experience	Standby	Positive voltage applied to terminal 1 of Arm Switch Motor	
	Contacts fail closed - safe position	MTBF 4×10^7 Hr	Field Experience	Standby	Positive voltage applied to terminal 2 of Arm Switch Motor	
	Contacts fail to open - arm position	MTBF 5×10^{11} Hr	Safety Test	Standby	Positive voltage remains on terminal 1 of Arm Switch Motor	
	Contacts fail to open - safe position	MTBF 5×10^{11} Hr	Safety Test	Standby	Positive voltage remains on terminal 2 of Arm Switch Motor	
	Contacts fail to close	MTBF 5×10^{12} Hr	Safety Test	Standby	Positive voltage cannot be applied to either terminal 1 or 2 of Arm Switch Motor.	System is disabled

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
Switch S2	Contacts fail closed	MTBF 4×10^7 Hr	Field Experience	Standby	Positive voltage applied to sequencer stepping switch	
	Contacts fail open	MTBF 7×10^9 Hr	Field Experience	Standby	Positive voltage cannot be applied to sequencer stepping switch	
	Contacts fail to open	MTBF 5×10^{11} Hr	Safety Test	Standby	Positive voltage cannot be removed from sequencer stepping switch	
	Contacts fail to close	MTBF 5×10^{11} Hr	Safety Test	Standby	Positive voltage cannot be applied to sequencer stepping switch	

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
Arm & Monitor (A&M) Switch	Arm contacts fail open	MTBF 1×10^7 Hr	Safety Test	Standby	Positive voltage cannot be applied to coil of sequencer	
	Arm contacts fail closed	MTBF 6.3×10^6 Hr	Safety Test	Standby	Positive voltage is applied to coil of sequencer	
	Monitor contacts fail open	MTBF 1×10^7 Hr	Safety Test	Standby	Monitor light goes out	
	Monitor contacts fail closed	MTBF 6.3×10^6 Hr	Safety Test	Standby	Monitor light stays on continually	
	Motor fails to operate	MTBF 5.1×10^4 Hr	Safety Test	Standby	Positive voltage cannot be applied to delay gate or sequencer	
	Motor operates inadvertently	MTBF 8×10^{11} Hr	Safety Test	Standby	Positive voltage applied to delay gate and sequencer, monitor light goes off	

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
5 Minute Sequencer	Stepping switch contacts fail open (position 2)	MTBF 9×10^{11} Hr	Field experience	Standby	Positive voltage cannot be applied to delay gate coil	
	Stepping switch contacts fail closed (position 1)	MTBF 1.3×10^{8} Hr	Field experience	Standby	Circuit to delay gate coil is completed	
	Coil fails open	MTBF 9×10^{8} Hr	Field experience	Standby	Stepping switch will not step to position 2	
	Coil fails shorted	MTBF 1.1×10^{8} Hr	Field experience	Standby	Battery #2 output will be shorted to ground - switch will not step to position 1	
Monitor Lamp	Burns out	MTBF 4×10^{5} Hr	Field experience	Standby	Indicates A&M switch is armed	

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
Delay Gate	Contacts fail open	MTBF $.7 \times 10^7$ Hr	Field experience	Standby	Positive voltage cannot be applied to rocket ignition squib	
	Contacts fail closed	MTBF 1.5×10^9 hr	Field experience	Standby	Cirquit to rocket ignition squib is completed	
	Coil fails open	MTBF 1.1×10^8 Hr	Field experience	Standby	Switch cannot be closed	
	Coil fails shorted	MTBF $.7 \times 10^7$ Hr	Field experience	Standby	Switch cannot be closed - Battery #2 output will be shorted to ground	

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
Battery #1	No power output	MTBF $.5 \times 10^7$ Hr	Field experience	Standby	A&M motor cannot be operated	
Battery #2	No power output	MTBF $.5 \times 10^7$ Hr	Field experience	Standby	Power not available to ignition squib or sequencer coil	
Battery #3	No power output	MTBF $.5 \times 10^7$ Hr	Field experience	Standby	Power not availabe to delay gate coil	

I COMPONENT	II COMPONENT FAILURE MODE	III FAILURE RATE	IV SOURCE OF DATA	V COMPONENT FAILURE STATE	VI EFFECT OF COMPONENT FAILURE	VII REMARKS
Rocket Ignition Squib	Fails to ignite	MTBF 3×10^{7} Hr	Safety Test	Standby	Rocket propellant will not be ignited	
	Ignites inadvertently (auto-ignition)	MTBF 7×10^{11} Hr	Safety Test	Standby	Rocket propellant will be ignited prematurely	
Rocket Propellant (Solid)	Fails to ignite	MTBF 1×10^{6} Hr	Safety Test	Standby	Rocket will not be launched	
	Ignites inadvertently (auto-ignition)	MTBF $.9 \times 10^{12}$ Hr	Safety Test	Standby	Rocket will be launched prematurely	

LIMITATIONS

This document is controlled by Apollo/ TIE System Safety and Program Assurance.

All revisions to this document shall be approved by the above noted organization prior to release.

ACTIVE SHEET RECORD

SHEET NUMBER	REV LTR	ADDED SHEETS			
		SHEET NUMBER	REV LTR	SHEET NUMBER	REV LTR
ii	A				
iii					
iv					
v					
vi					
1-1					
2-1					
2-2					
2-3					
2-4					
2-5					
2-6					
3-1					
3-2					
3-3					
3-4					
4-1					
4-2					
4-3					
4-4					
4-5					
4-6					
4-7					
4-8					
4-9					
4-10					
4-11					
4-12					
4-13					
4-14					
4-15					
4-16					
4-17					
4-18					
4-19					
4-20					
4-21	A				
4-22	A				
4-23					
4-24					
5-1					
5-2					
5-3					
5-4					
5-5					
5-6					
5-7					
5-8					
5-9					
5-10					
5-11					
5-12					
5-13					
5-14					
5-15					
5-16					
5-17					
5-18					
5-19					
A-1					
A-2					
A-3					
A-4					
B-1					
B-2					
C-1					
C-2					
C-3					
C-4					
C-5					
C-6					
C-7					
C-8					
C-9	A				

3

ACTIVE SHEET RECORD

SHEET NUMBER	REV LTR	ADDED SHEETS				SHEET NUMBER	REV LTR	ADDED SHEETS			
		SHEET NUMBER	REV LTR	SHEET NUMBER	REV LTR			SHEET NUMBER	REV LTR	SHEET NUMBER	REV LTR
C-10	A					F-1	A				
C-11						F-2					
C-12						F-3					
C-13						F-4	A				
C-14											
C-15											
C-16											
C-17											
C-18;											
D-1											
D-2											
D-3											
D-4											
D-5											
D-6											
D-7											
D-8											
D-9,											
E-1											
E-2											
E-3											
E-4											
E-5											
E-6											
E-7											
E-8											
E-9											
E-10											
E-11											
E-12											
E-13											
E-14											
E-15											
E-16											
E-17											
E-18											
E-19											
E-20	A										

REVISIONS

REV. SYM	DESCRIPTION	DATE	APPROVED
A	COMPLETE REVISION	6/17/8	D.R. PLEAS D R Pleas

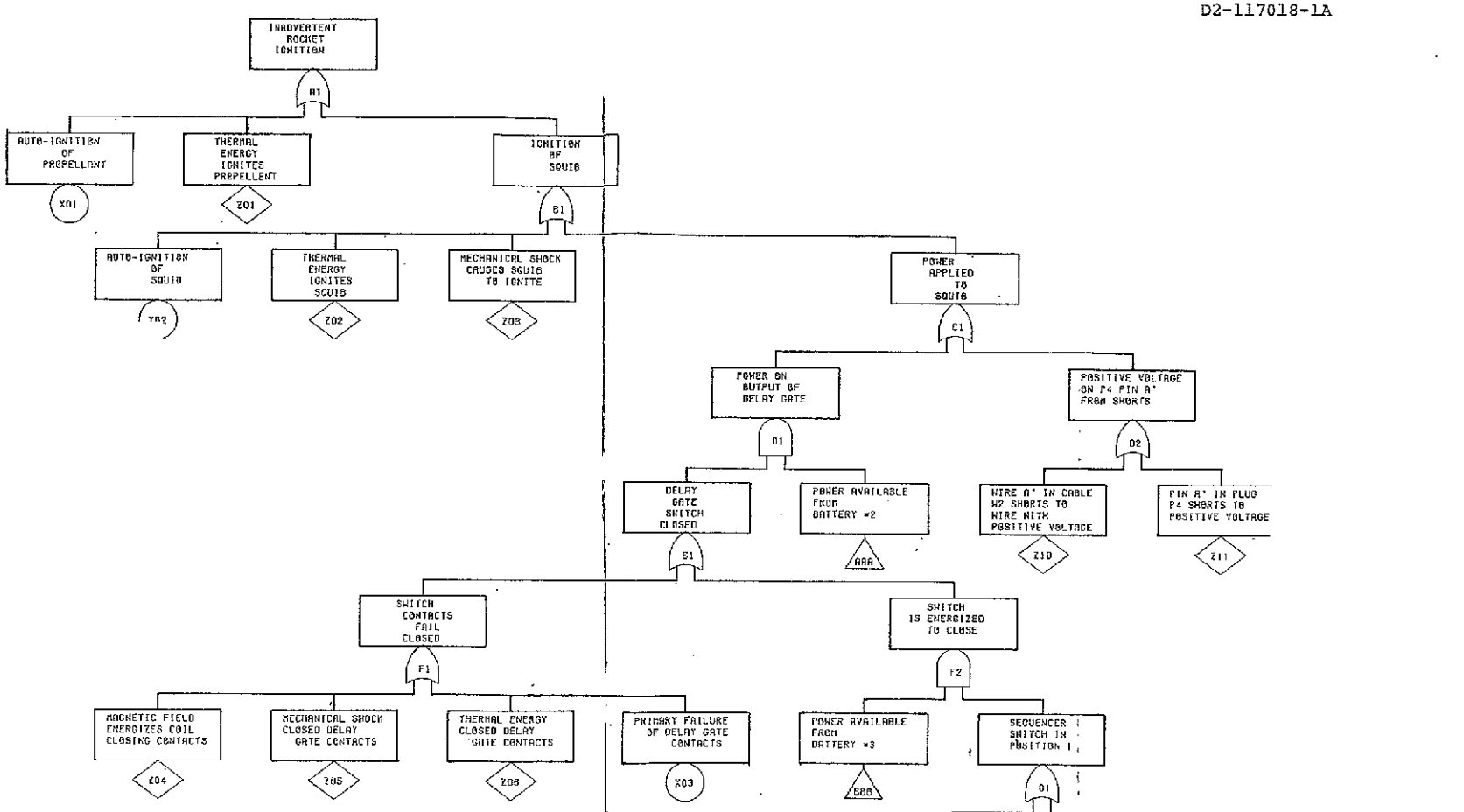


Figure E8

Machine Drafted Diagram

	INITIALS	DATE	REV BY INITIALS	DATE	TITLE	MODEL
CALC					TOP-- INADVERTANT ROCKET IGNITION	APOLLO
CHECK						
APPD						
APPD						
APPD						

REV LTR ____

BOEING NO D2-120181 SH

FOLDOUT FF

FOLDOUT FRAME 2