8B

BNWL-2144

UC-32

**ACORN:
A Computer Program for
Plotting Fault Trees**

**By
J. L. Carter**

November 1977

**Prepared for the
U.S. Department of Energy
under Contract EY-76-C-06-1830**

Battelle
Pacific Northwest Laboratories

BNWL-2144

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

The views, opinions and conclusions contained in this report are those of the contractor and do not necessarily represent those of the United States Government or the United States Department of Energy.

PACIFIC NORTHWEST LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
Under Contract EY-76-C-06-1830

Printed in the United States of America
Available from
National Technical Information Service
United States Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22151

Price: Printed Copy \$____*; Microfiche \$3.00

*Pages	NTIS Selling Price
001-025	\$4.50
026-050	\$5.00
051-075	\$5.50
076-100	\$6.00
101-125	\$6.50
126-150	\$7.00
151-175	\$7.75
176-200	\$8.50
201-225	\$8.75
226-250	\$9.00
251-275	\$10.00
276-300	\$10.25

ERRATA

- Page 9 In line 24 the format statement should
 read [I10,6(I7,I3)/7(I7,I3)]
- Page 14 In line 12 NOPLOT=1 should be replaced
 by NOPLOT=0

3 3679 00062 7077

ACORN: A PROGRAM FOR PLOTTING FAULT TREES

by
J. L. Carter

November 1977

BATTELLE
Pacific Northwest Laboratories
Richland, Washington 99352

CONTENTS

ACKNOWLEDGMENTS.	v
SUMMARY	vii
1.0 INTRODUCTION.	1
2.0 INFORMATION FOR THE USER	3
2.1 DESCRIPTION OF PROGRAM.	3
2.2 INPUT INSTRUCTIONS FOR ACORN.	5
2.3 DESCRIPTION OF TREE LOGIC CARDS.	11
2.4 DESCRIPTION OF OUTPUT	11
2.5 SAMPLE PROBLEM	12
3.0 PROCEDURE FOR SOLUTION	23
3.1 DETERMINING THE BASIC LOGIC OF THE TREE	23
3.2 GATE LOWERING	25
3.3 DETERMINATION OF GEOMETRICAL COORDINATES	26
3.4 PROCESSING OF DESCRIPTIVE LABELS	27
3.5 DESIGN OF GEOMETRICAL FIGURES	28
4.0 ADDITIONAL PROGRAMMING INFORMATION	33
4.1 GENERAL INFORMATION.	33
4.2 SUBROUTINES AND THEIR FUNCTION	33
4.3 CONTENTS OF BLANK COMMON	36
REFERENCES	45

FIGURES

1	Risk Assessment Computer Package	2
2	Sample Problem Illustrating Capabilities of ACORN.	13

TABLES

1	Design of Geometric Figures	30
---	---------------------------------------	----

ACKNOWLEDGMENTS

The author acknowledges the many helpful discussions with P. A. Crossetti and T. I. McSweeney during the development of the program, the assistance of the many users during the continuing debugging phase, and the valuable contribution of P. J. Pelto to the preparation of the report.

SUMMARY

A description and user instructions are presented for ACORN, a FORTRAN computer program for drawing fault trees. ACORN analyzes the input logical structure of a fault tree and provides data for a CalComp plot of the tree. AND, OR, and INHIBIT gates are permitted, and basic events are drawn as diamonds, circles, or houses. Each component (gate or basic event) can have a descriptive label within a rectangle attached to the top of its respective symbol.

Tree logic is input as a set of FORTRAN statements, each defining a gate in terms of logical operations of the components input to it. ACORN develops the logical structure of the tree from the input statements. The tree's physical structure is developed by assigning relative spatial coordinates to the logical relationships between a gate and its inputs.

ACORN provides input data checking, a printer plot of the fault tree, and plotting data for a CalComp model 763 plotter. The program is operational on a CONTROL DATA CYBER 74 computer.

1.0 INTRODUCTION

A risk-based fault tree analysis method has been developed at PNL for analysis of nuclear fuel cycle operations.⁽¹⁾ This method was developed for the Energy Research and Development Administration (ERDA)-sponsored risk analysis of systems for managing high-level waste. A computer package consisting of three programs has been developed to assist in the performance of a risk assessment: ACORN (draws fault trees), MFAULT⁽²⁾ (analyzes fault trees), and RAFT⁽³⁾ (calculates risk). Figure 1 gives a summary of the input, output, and interrelationships of these programs.

This report gives a description and user instructions for ACORN, a computer program for plotting fault trees. Section 2 gives a basic description of the program along with input/output information and a sample case. Section 3 describes the calculation procedures used in more detail. Section 4 gives additional programming information. Some familiarity with fault tree methodology is assumed. References 1, 4, and 5 will give the reader a general background in systems safety analysis and fault tree analysis.

ACORN has been used extensively for more than two years. The program is operational on a CONTROL DATA CYBER 74 and provides plotting data for a CalComp Model 763 plotter.

FIGURE 1. Risk Assessment Computer Package

2.0 INFORMATION FOR THE USER

In this section, a brief description of the program is given, including its capabilities and its limitations. This is followed by input and operating instructions and a description of the output. Input and output of a contrived sample problem illustrating the capabilities of the program complete the section.

2.1 DESCRIPTION OF PROGRAM

Fault trees drawn by ACORN have the following characteristics. The trees can contain AND, OR, and INHIBIT gates. Basic events are plotted as diamonds, circles, or houses. The gates share a common numbering system of no more than four digits with an A prefix. The basic events share a similar common numbering system with an X prefix. The prefix and number appear on the plot within the symbol for each component (gate or basic event). A rectangle attached to the top of each symbol contains a descriptive label for the component. In conformance with common practice for a multiply-appearing gate, triangles are used for transfers in and out; the gate prefix and number appear within the triangle.

The logic for a tree is input to ACORN in the form of a set of Fortran logical statements.^(a) From these statements, the gate type and the components which are input to it can be determined. The descriptive labels are read in separately. A few control parameters, some minimal additional input to help identify the types of the basic events, and special numerical instructions, if needed, complete the input.

^(a) The use of Fortran logical statements to input the logic of the tree, and of A and X arrays for the gates and basic events, came about because ACORN was originally written to obtain plots of fault trees analyzed by the computer code SAFTAC.⁽⁶⁾ This latter code required a subroutine LOGIC in this form. As ACORN has evolved, some incompatibilities with SAFTAC have entered, none of which cannot be resolved with little effort.

ACORN develops the logical structure of the tree from the input statements. The physical structure of the tree is then calculated from this logical structure. The logical structure is unaltered but the physical structure can be changed by a gate-lowering option. Judicious use of this option materially improves the compactness and readability of the tree.

Although the code does no other analysis than that needed for the plot, it provides several other services. The input data are edited and diagnostics are provided. A printer plot of the tree is provided, showing the components by number in their relative locations on the final plot. The text of the descriptive labels is broken into lines. Finally, the logical input is optionally revised and punched in a format compatible with the input requirements of the fault tree analysis program MFAULT.⁽²⁾

As ACORN uses dynamic dimensioning, there are no fixed limits to the number of components in a tree. The design of the symbols does impose a limit of four digits on the number portion of the name of a component. There are also limits of 50 on the number of levels in the tree and limits of 75 and 50 on the number of characters in the descriptive labels for components and for inhibitions respectively.

Since ACORN has an automatic central memory adjustment feature, the memory requirement is also variable. To determine the approximate central memory (CM) required for a particular problem, the following algorithm can be used:

$$CM = 20,500 + \text{Max} (C', C)$$

where

$$C' = (8A' + 7X' + 4R')/3 + 250$$

is an initial estimate of the memory increment based upon estimated numbers of gates, A' , basic events, X' , and total appearances of components R' ; and

$$\begin{aligned} C &= 18A + 15X + 6R + 200 \text{ for a CalComp plot} \\ &= 4A + X + 6R + 400 \text{ for no plot} \end{aligned}$$

is the final requirement for incremental memory for the actual numbers of gates, A , basic events, X , and total appearances, R .

The default values for A', X', and R' are 500, 1000, and 2000 respectively. An overrun of 1/3 for each of the quantities is included in the formula for C'. For problems larger than this, the default values must be overwritten to avoid overlapping of arrays at the beginning of processing. The overwriting can readily be done with the integer variables MAT, MXT, and MRHST which are in the Namelist group PARS. The default values give C' = 6600, which is sufficient final incremental memory for a plot of a tree with, for example, A, X, and R of 125, 125, and 375.

Running time for ACORN is approximately linear with the number of components appearing on the plot: a tree with 200 components requires about 30 sec on the Cyber 74, while one with 600 components requires about 100 sec.

2.2 INPUT INSTRUCTIONS FOR ACORN

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
----------------	-----------------	--------------------

CARD TYPE 1 - Title Card

1-72	TITLE	Any alphanumeric information
		This title always appears as a heading to the printed output and, if requested by LSTITL, also appears on the plot. The maximum number of characters in the title can be controlled by MLOTG, (see Card Type 2), to a maximum of 80, a default of 72.

CARD TYPE 2 - Control Card (NAMELIST input - begin first card in column 2 with \$PARS). The default value of each variable is given in parentheses. Only those values which differ from the default values need be input.

IPS	Page size (0)
	0 - 30 in. paper
	1 - 11 in. paper
NO PLOT	Plot (0)
	0 - Plot the tree
	1 - Do not plot the tree

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
	NOHOL	Description of events (0) 0 - Descriptive labels for the rectangles will be read in 1 - No descriptive labels
	NONAME	Identification of events (0) 0 - The numbers of the A's and X's will be plotted within the symbols 1 - The symbols will be left blank
	LSTITL	Title Plot (0) 0 - The title will not be plotted 1 - The title will be plotted
	MLOTC	Maximum number of letters on title card (72). Maximum permitted is 80.
	IERECT	X Rectangles (0) 0 - No rectangles for X events 1 - Rectangles for X events
	MCPL1	Maximum characters per line for A rectangles (20). Although there is no logical limit to MCPL1, there is presently a limit of 75 characters in a descriptive label and a limit of 3 lines per label, giving a prac- tical limit of 25 for MCPL1.
	MCPL2	Maximum characters per line for X rectangles (15). The comments about MCPL1 also apply to MCPL2.
	MXLIN1	Maximum lines for A rectangles (3). Maximum is 3. See comments about MCPL1.
	MXLIN2	Maximum lines for X rectangles (3). Maximum is 3. See comments about MCPL1.
	MOSTIE	Type of most X's (0) 0 - Most X's are diamonds 1 - Most X's are circles Note: Houses receive a special treatment

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
	NTRACE	Multiple trace (0) 0 - Plot without retracing 1 - Retrace the plot
	LUIN	Alternate logical unit for input (0) Tree logic and label information input by 0 - Card reader 1 - Logical Unit 2 If LUIN $\neq$ 0, a local file TAPE2 must be provided in the job stream cards. The title, control cards, and other data are always input via the card reader. Note: This option and the punch option (NPUNCH $\neq$ 0) cannot both be used, because each uses Logical Unit 2.
	LOWLV	Level lowering signal (0) 0 - No gates will be lowered 1 - Some gates will be lowered
	NPUNCH	Card punch option (0) 0 - No cards will be punched 1 - Logic cards will be punched in a form acceptable as input to the MFAULT ⁽²⁾ program. These card images are written to a local file TAPE2; therefore disposal of TAPE2 must be provided for in the job stream cards. Note: This option and the alternate input option (LUIN $\neq$ 0) cannot both be used, because each uses Logical Unit 2.
	MAT	Maximum number of A's (500). This is only an estimate for use in preliminary storage allocation before the actual number of A's is counted.
	MXT	Maximum number of X's (1000). See comment about MAT.
	MRHST	Maximum number of total appearances of A's plus X's (2000). See comment about MAT.
	MNLT	Maximum number of physical levels (50)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
	HTITL	Multiplier for letter height in title (2.0)
	SPTITL	Multiplier for spacing between title and top event (8.0)
	HTLET	Provisional letter height in inches (0.28). If the overall height of tree plus plotted title will fit on the plotter paper (10.5 in. for 11 in. paper, 28.5 in. for 30 in. paper), the reference letter height is HTLET. If the overall height is too large, HTLET is reduced in steps (0.014 in.) until the tree will fit.
	DIAGN	Switch for diagnostic printout (0.0) 0.0 - no diagnostic printout 1.0 - diagnostic printout Note: The diagnostics obtained with this option are a detailed step-by-step report of values of variables as the geometrical coordinates are determined. These should not be confused with the diagnostic statements which normally accompany error detection in the input data.
	DAPQ	Angular divisions per quadrant for plotting arcs (24.0)
	DHRGS	Multiplier for spacing between gate symbol and rectangle (0.0). At present, this value must be 0.
	ANGL	Angle subtended by side arcs of OR-gate (60.0). At present, this value must be 60. (End of NAMELIST input - terminate with \$)

CARD TYPE 3 - Tree Logic Cards. See description of Tree Logic Cards at end of Input Instructions.

CARD TYPE 4 - End-of-Logic Card

1 The letter Z

2-80 Any alphanumeric information; treated as a comment

CARD TYPE 5 - Minority Event Identification List (14I5)
A card of this type is always required. If no minority events are present, use a card with a 1 punched in Column 5.

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	ND	Number of minority events (circles if MOSTIE = 0, diamonds if MOSTIE = 1)
6-70 et seq.	ID(N), N = 1, ND	Identification numbers of the ND minority events, in any order, a maximum of 13 on the first card and 14 on each additional card, as required, for a total of ND numbers. These identification numbers should be among the i.d.s of the X's in the tree logic statements.

CARD TYPE 6 - Transfer-out Triangle Specification List (14I5)

A card of this type is always required. If no additional transfer-out triangles are requested, use a card with a 1 punched in Column 5.

1-5	ND	Number of additional gates for which transfer-out triangles are specified.
6-70 et seq.	ID(N), N = 1, ND	Identification numbers of the ND gates, in any order, a maximum of 13 on the first card and 14 on each additional card, as required, for a total of ND numbers. These identification numbers should be among the i.d.s of the A's in the tree logic statements. These triangles are in addition to those which are required by the logic of the tree, and may be desired if the tree being plotted is a subtree of a larger tree and transfers occur from this subtree to other portions of a larger tree.

CARD TYPE 7 - Gate Lowering Specification List !I10,6(I7,I3)/7(I7,I3)1

Note: This card type is required only if LOWLV ≠ 0.

1-10	NCL	Number of primary gates to be lowered.
11-17	NGATE(1)	Identification number of 1st primary gate to be lowered. This gate and all components which are input to it or to any gates under it in the tree are lowered.
18-20	LOWLVL(1)	The number of physical levels by which this gate and its inputs are to be lowered

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
21-27	NGATE(2)	Identification number and number of levels for 2nd
28-30	LOWLVL(2)	primary gate and its inputs
!	!	Continue for 6 number pairs on the first card, 7
!	!	number pair on each subsequent card, as needed, for
		a total of NCL number pairs.

Note: Card Type 7 is the end of input if NOHOL $\neq$ 0.

CARD TYPE 8 - Descriptive Labels (A1, I4, 75H)

1	CONE	A for gates, X for basic events, I for an inhibition
2-5	JD	Numerical identification
6-80	TEMP(I)	Hollerith description for this component, limited to 2 lines (50 columns maximum) for an inhibition.

Repeat Card Type 8 for each component. At present all the Hollerith description for a component must appear on a single card. The label cards can appear in any order. If a label card is missing for a component, the label for that component is blank on the plot. An inhibit gate and its inhibition are treated as separate components here, but must have the same value of JD.

CARD TYPE 9 - End-Of-Labels Card (A1, I4, 75H)

1	The number 9
2-5	Any number or blank. Do not use alphabetic information--it is read in integer format
6-80	Any alphanumeric information; treated as a comment. This is the end of input.

2.3 DESCRIPTION OF TREE LOGIC CARDS

These cards are in the form of Fortran statements defining gates as logical variables of the inputs to the gates and the logical operators .AND. or .OR. Restrictions on the statements are as follows:

- Gates must be in the A array, basic events in the X array. Subscripts on the A and X arrays must be numeric; integer variables are not allowed.
- Only the logical operators .AND. and .OR. are acceptable and mixed operations in the same gate are not allowed. The symbols .A. and .O. can be substituted for .AND. and .OR. respectively.
- The top event must be the statement $TOP = A(N)$, N a numeric value.
- A statement may be continued to another card by the Fortran "continue" rule of a non-zero punch in column 6. The maximum length of a statement is 150 non-blank characters located in columns 7-72 of as many total cards as desired.
- An inhibit gate is indicated by using =I in place of = in the Fortran statement defining the gate. This is equivalent to prefixing an I to the first component which is input to the gate, i.e., treating the component as a member of the IA or IX array.
- A house is indicated by using H instead of X for the event. This is equivalent to treating the component as a member of the H array. This substitution applies to the logic statement; the label input for the event should retain the X, and the house will be numbered with an X on the plotted tree.

Statements may be in any order. Fortran comment cards may be inserted anywhere in the logic stream; these cards are printed without further interpretation.

2.4 DESCRIPTION OF OUTPUT

The printed output from ACORN reproduces the input data in the order in which the cards are read. The form of the output may, however, be altered.

Self-explanatory diagnostic statements may be interspersed with the data, and tables of edited data will intervene. Thus, a table of all the control parameters is printed, not just the ones which were input as different from their default values. Two tables of edited data follow the Card Type 5 printout: 1) a listing by gate of the components which are input to the gate and 2) a listing by component of the gates to which the component is input. In these tables, the component numbers contain the adders which are used internally to distinguish the types of component. If punched card output is requested, these data are written to the punch and output units at this point. After the Card Type 7 printout, two more tables appear: 1) a listing by physical level of the reference y-coordinate of the level on the final plot and 2) a listing of the basic logic of the tree in the logical order in which the components appear. A detailed discussion of this latter table is inappropriate here; it should be noted, however, that the last three columns of this table, together with the Y-coordinate values of the preceding table, give each component and its reference x, y coordinates on the final plot. Following this table is a printer plot of the tree, limited to the top 22 levels. The listing of the descriptive labels completes the major printed output. The statement "End of case" formally marks the end of printed output for a case. A similar statement "End of run" terminates the output of the program.

2.5 SAMPLE PROBLEM

A sample problem illustrating the input and output of ACORN is given in this section. Figure 2 is a fault tree plotted by ACORN. The descriptive labels have been designed to illustrate the capabilities of the code.


```

ACORN,M11.
ACCOUNT(PW=BCC039,UN=BCC039)
REFNCE.
ATTACH(ACORN,ACORN2P,ID=ACORN)
TAPE(TAPE8,BP400,ID=PLOT)
ACORN.
SAVPLOT(TAPE8)

SAMPLE PROGRAM ILLUSTRATING CAPABILITIES OF ACORN
SPANS=10001, LSTITLE=1, IERECT=1, MCPL1=25, MCPL2=25, LOWLV=1,
MAT=15, NKT=25, MCHST=40,
HTLET=98, NOPLT=1
A(14)=A(12).OR.A(22).OR.A(32)
A(16)=A(42)
A(9)=A(10).OR.A(12)
A(155)=X(57).OR.X(55)
A(14)=H(1).AND.A(14).AND.A(16)
TOP=A(100)
A(12)=H(7).OR.X(7)
A(100)=A(7).AND.A(9).AND.A(6)
A(6)=X(11).AND.X(12).AND.A(155)
A(10)=X(15)
ZZZ      END OF LOGIC INPUT
2      13      85
1      14
:      9      3
A 100 LOGIC STATEMENT DEFINING TOP EVENT CAN APPEAR ANYWHERE IN INPUT LOGIC
A 6 THE SYMBOL BELOW IS USED FOR AN AND GATE
A 3 THE SYMBOL BELOW IS USED FOR AN OR GATE
A 14 THE SYMBOL BELOW IS USED FOR AN INHIBIT GATE
X 3 THE SYMBOL BELOW IS USED FOR A HOUSE
X 11 THE SYMBOL BELOW IS USED FOR A DIAMOND EVENT
X 12 THE SYMBOL BELOW IS USED FOR A CIRCLE EVENT
I 14 TWO-LINE LIMIT ON INHIBIT GATE INHIBITION
I 15 INHIBITION TEXT USED GATE NUMBER WITH 1, NOT A
A 12 TRANSFER-OUT TRIANGLE AT FIRST APPEARANCE OF MULTIPLY-APPEARING GATE
A 22 NORMAL IN-TRANSFER HAS NUMBER/LEVEL OF ITS OUT-TRANSFER
A 92 IN-TRANSFERS ARE ASSUMED FOR UNDEFINED GATES
A 14 A TRANSFER-OUT TRIANGLE CAN OPTIONALLY BE ADDED TO ANY GATE
X 7 INPUT TO AN INHIBIT GATE CAN BE EITHER A GATE OR A BASIC EVENT
A 16 INHIBIT GATE FLAGGED BY PREFIX I, ON (FIRST) COMPONENT INPUT
X 7 HOUSE FLAGGED BY PREFIX H FOR X IN LOGIC STATEMENT
A 9 BRANCH LOWERED BY LOWERING GATE AT TOP OF BRANCH
X 67 MAJORITY EVENTS TYPED BY MODIFIC VALUE IN INPUT PARAMETERS
X 65 MINORITY EVENTS IDENTIFIED BY NUMBER IN ADDITIONAL INPUT
X 9 EVENTS AND HOUSES NUMBERED AS IF IN SAME ARRAY, 4-DIGIT MAXIMUM
A 155 GATES NUMBERED AS IF IN SAME ARRAY, 4-DIGIT MAXIMUM
A 42 LABELS FOR INHIBIT GATE AND INHIBITION MAY BE SEPARATE IN INPUT
9999999

```

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

PAGE SIZE (0-LARGE, 1-SMALL)	IPS = 1	MULTIPLE TRACE ON PLOT (0-NO, 1-YES)	NTRACE= 0
PLOT (0-YES, 1-NO)	NOPLT= 0	TREE LOGIC AND LABEL CARDS (0-CR, 1-TAPE)	LUIN = 0
DESCRIPTION OF EVENTS (0-YES, 1-NO)	NOHOL = 0	ANY GATES LOWERED (0-NO, 1-YES)	LOWLW = 1
IDENTIFICATION OF EVENTS (0-YES, 1-NO)	NONAME= 0	PUNCH CARDS FOR "FAULT" (0-NO, 1-YES)	NPUNCH= 0
TITLE PLOT (0-NO, 1-YES)	LSTITL= 1	MULTIPLIER FOR LETTER HEIGHT IN TITLE	HTITL = 2.000
NUMBER OF COLUMNS READ ON TITLE CARD	MLOTC = 72	TOP EVENT - TITLE SPACING	SPTITL= 8.000
X-RECTANGLES DRAWN (0-NO, 1-YES)	IERECT= 1	SYMBOL - RECTANGLE SPACING	DHRGS = 0.000
MAX CHARACTERS/LINE FOR A-RECTANGLE	MCPL1 = 25	ANGLE OF SIDE ARCS OF OR-GATE, DEGREES	ANGL1 = 60.
X-RECTANGLE	MCPL2 = 25	ANGLE INCREMENTS PER QUADRANT IN ARC PLOT	DAPS = 24.
MAX LINES FOR A-RECTANGLE	MXLIN1= 3	PROVISIONAL LETTER HEIGHT, INCHES	HTLEY = .9800
X-RECTANGLE	MXLIN2= 3	DIAGNOSTIC PRINTOUT (0-NO, 1-YES)	DIAGN = 0.
MAJORITY OF X ARE (0-DIAMONDS, 1-CIRCLES)	MOSTIE= 0		

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

LISTING OF INPUT LOGIC STATEMENTS

```

1      A(14)=A(12).OR.A(22).OR.A(32)
2      A(16)=IA(42)
3      A(3)=A(10).OR.A(12)
4      A(135)=X(57).OR.X(85)
5      A(9)=M(3).AND.A(14).AND.A(16)
6      TOP=A(100)
7      A(12)=M(7).OR.X(9)
8      A(100)=A(3).AND.A(9).AND.A(6)
9      A(6)=X(11).AND.X(13).AND.A(135)
10     A(10)=IX(5)
11     ZZZ      END OF LOGIC INPUT

```

2 13 95

```

GATE 22 IS UNDEFINED AND WILL BE MARKED WITH AN INPUT TRIANGLE
GATE 32 IS UNDEFINED AND WILL BE MARKED WITH AN INPUT TRIANGLE
GATE 42 IS UNDEFINED AND WILL BE MARKED WITH AN INPUT TRIANGLE

```

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

TREE LOGIC ORDERED BY GATE NUMBER

I	GATE	COMPONENTS WHICH ARE INPUT TO GATE			
1	2	12	20010		
2	10006	135	50013	60011	
3	10009	14	20016	70003	
4	20011	50005			
5	12	50009	70007		
6	14	12	22	32	
7	20016	42			
8	22	-399			
GATE 22 HAS NO INPUT					
9	32	-399			
GATE 32 HAS NO INPUT					
10	42	-399			
GATE 42 HAS NO INPUT					
11	10160	3	10006	10009	
12	135	50085	50057		

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

TREE LOGIC ORDERED BY COMPONENT NUMBER

GATES TO WHICH COMPONENT IS INPUT

14

I	COMPONENT	GATES TO WHICH COMPONENT IS INPUT	I	GATES
1	3	10100		
2	10005	10100		
3	10009	10100		
4	20010	3		
5	12	3		
6	14	10009		
7	20015	10009		
8	22	14		
9	32	14		
10	42	20016		
11	10100	0		
12	135	10005		
13	70003	10009		
14	60005	20010		
15	70007	12		
16	60009	12		
17	40011	10006		
18	50013	10006		
19	60037	135		
20	50045	135		

TAKOFF TRIANGLES WERE SPECIFIED FOR THE FOLLOWING

14

THE FOLLOWING 1 GATES AND ALL COMPONENTS BELOW THEM ON ANY PATH HAVE BEEN LOWERED BY THE NUMBER OF LEVELS INDICATED IN THE NUMBER PAIR (GATE, LEVEL)

(0, 3) (

LISTING OF YSPACE OUTPUT ARRAYS

I Y(I,I)

1	9.0076
2	7.7491
3	6.4907
4	5.2323
5	3.9734
6	2.7154
7	1.4569

LETTER HEIGHT = .0260

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

BASIC TREE LOGIC OF LISC ARRAY, WITH LOCATION OF COMPONENT

I	LISC(I)	LNC(I)	COMPONENT	X LOCATION	LEVEL
1	8	1	10100	9.8085	1
2	3	4	3	2.7810	2
3	5	6	10009	9.1440	5
4	9	9	10006	14.8320	2
5	10	12	20010	0.0000	3
6	-7	13	12	5.5620	3
7	16	15	70003	3.7710	6
8	-1	15	14	7.2270	6
9	2	18	20016	14.5170	5
10	19	19	60011	11.3760	3
11	20	19	50013	14.8320	3
12	4	19	135	19.2880	3
13	21	21	60005	0.0000	4
14	17	21	70007	3.9340	4
15	18	21	60009	7.2900	4
16	100007	21	12	3.7710	7
17	100011	21	22	7.2270	7
18	100012	21	32	10.6830	7
19	100013	21	42	14.5170	7
20	14	21	60057	15.5600	4
21	15	21	50085	20.0160	4

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

PRINTER PLOT OF TREE

7	6	5	4	3	2	1
---	---	---	---	---	---	---

SAMPLE PROBLEM ILLUSTRATING CAPABILITIES OF ACORN

LISTING OF INPUT LABELS FOR RECTANGLES

```

1      A 100 LOGIC STATEMENT DEFINING TOP EVENT CAN APPEAR ANYWHERE IN INPUT LOGIC
2      A 6 THE SYMBOL BELOW IS USED FOR AN AND GATE
3      A 3 THE SYMBOL BELOW IS USED FOR AN OR GATE
4      A 10 THE SYMBOL BELOW IS USED FOR AN INHIBIT GATE
5      X 3 THE SYMBOL BELOW IS USED FOR A HOUSE
6      X 11 THE SYMBOL BELOW IS USED FOR A DIAMOND EVENT
7      X 13 THE SYMBOL BELOW IS USED FOR A CIRCLE EVENT
8      I 10 TWO-LINE LIMIT ON INHIBIT GATE INHIBITION
9      I 16 INHIBITION TEXT USES GATE NUMBER WITH I, NOT A
10     A 12 TRANSFER-OUT TRIANGLE AT FIRST APPEARANCE OF MULTIPLY-APPEARING GATE
11     A 22 NORMAL IN-TRANSFER HAS NUMBER/LABEL OF ITS OUT-TRANSFER
12     A 32 IN-TRANSFERS ARE ASSUMED FOR UNDEFINED GATES
13     A 14 A TRANSFER-OUT TRIANGLE CAN OPTIONALLY BE ADDED TO ANY GATE
14     X 5 INPUT TO AN INHIBIT GATE CAN BE EITHER A GATE OR A BASIC EVENT
15     A 16 INHIBIT GATE FLAGGED BY PREFIX I ON (FIRST) COMPONENT INPUT
16     X 7 HOUSE FLAGGED BY USING H FOR X IN LOGIC STATEMENT
17     A 9 BRANCH LOWERED BY LOWERING GATE AT TOP OF BRANCH
18     X 57 MAJORITY EVENTS TYPED BY MOSTIE VALUE IN INPUT PARAMETERS
19     X 95 MINORITY EVENTS IDENTIFIED BY NUMBER IN ADDITIONAL INPUT
20     X 9 EVENTS AND HOUSES NUMBERED AS IF IN SAME ARRAY, 4-DIGIT MAXIMUM
21     A 135 GATES NUMBERED AS IF IN SAME ARRAY, 4-DIGIT MAXIMUM
22     A 42 LABELS FOR INHIBIT GATE AND INHIBITION MAY BE SEPARATE IN INPUT
23     99999999

```

COMPONENT LABELS USE 139 WORDS OF CENTRAL MEMORY

677 WORDS ALLOCATED FOR DYNAMIC ARRAY, 600 WORDS USED

END OF CASE

END OF RUN

3.0 PROCEDURE FOR SOLUTION

In this section, brief descriptions are given of the procedures used in determining the basic logic of the tree from the input statements and in transforming this logic and the descriptive information into the data for the plotter. Reference to the descriptions of variables and subroutines in Section 4.0 may frequently be helpful. Also included here is information about the design of the geometrical figures which appear on the plot.

3.1 DETERMINING THE BASIC LOGIC OF THE TREE

The basic logic of the tree is contained in the input set of Fortran statements, each of which defines a gate in terms of logical operations on the components which are its inputs. The first task of ACORN is to put this logic in more usable form as it is read. Three arrays are used for this: 1) IDNO, containing the identification numbers (i.d.s) of the gates in the order of their definition by the logic statements, 2) INAR, containing the i.d.s of the components which are inputs to the gates in order of their appearance on the right hand side (rhs) of the logic statements, and 3) INDX, each entry containing the position of the last entry in INAR after the current gate has been defined. The statement defining the top event is interpreted as defining an extra gate with i.d. of 0.

The role of IDNO is next expanded to contain the i.d.s of all the components. This is done by scanning the INAR array to locate any i.d.s not already in IDNO and adding them to IDNO. Any gates so located are undefined; their i.d.s are added first. The i.d.s of the basic events follow in the order of their first appearance in INAR. Finally, an i.d. for a dummy event is added for each inhibit gate, to provide a reference i.d. for the additional descriptive label for the inhibition. For the undefined gates, INDX is expanded in step with IDNO, by repeating the previous entry each time an IDNO entry is made.

At the conclusion of this process, IDNO contains an entry for each distinct component (including the dummy entries for the inhibitions which are not used in further development of the tree logic), INAR contains an entry for each appearance of a component in the tree, and INDX contains an entry

for each gate. This input order is next converted into a logical order by use of the concept of logical level. The concept is simple: each appearance of a component has an associated logical level, and all components which are input to a gate in logical level L are in logical level $L + 1$. If the logical level 0 is given to the fictitious gate with i.d. of 0, then its only input, the top event, is in logical level 1. Utilizing this concept, the information in INAR is reordered by logical level into a new array, LISC, the top event being the first entry. The logical order results if, as LISC is filled, it is also scanned to determine the next data to be retrieved from INAR. Thus, immediately following the top event are the components in logical level 2 which are its inputs, followed in turn by inputs to each gate as it is encountered in the scan. As each component is added to LISC, an entry to a new companion array, LNC, is made; this entry is the current cumulative count of the total number of entries in LISC. The logical level of the component is also entered in another new array, LVL. These three arrays contain the basic logical order of the tree.

The procedure as described would redevelop the complete input logic for a multiply-appearing gate at each appearance, an undesirable result for a plot. To prevent this, each time a gate is encountered in the scan, the LISC entries preceding the gate are rescanned to determine if the gate has appeared before. If so, the first appearance is marked for a transfer-out triangle, by setting the sign of the entry negative, and all other appearances are marked for a transfer-in triangle, by adding a large number to the entry.

For further analysis, a procedure for tracing a logical path through the tree is needed. Such a path consists of a set of nested components, one from each logical level in the path, the component in level $L + 1$ being an input to the component in level L , and the terminal component being a basic event or a transfer-in triangle. A complete path starts with the top event, a partial path with any component. To trace a path, assume the entry for the starting component is located at $LISC(I)$. The entries $LNC(I) + 1$ through $LNC(I + 1)$ span the LISC locations for the components which are input to the starting component; one of these must be in the path. Assume that the desired component is at $LISC(J)$, where $LNC(I) + 1 \leq J \leq LNC(I + 1)$.

Then the entries $LNC(J) + 1$ through $LNC(J + 1)$ span the LISC locations of the inputs to this component. One of these is selected to continue the path and the process of alternating between LISC and LNC arrays continues until some $LISC(M)$ is found for which $LNC(M) = LNC(M + 1)$, marking a component with no input which therefore terminates the path.

3.2 GATE LOWERING

Converting the logical structure of a tree into a physical plot can be done straightforwardly by equating the logical level of a component, as contained in the LVL array, with the physical level of the component on the plot. For trees of several hundred components, such a procedure is apt to lead to an ill-proportioned plot which is so much wider than it is high that the logical relationships are difficult to recognize. One method for improving the plot is to relocate portions of the plot into a more compact form which increases the height and reduces the width. A relocation is readily achieved by lowering the physical position of a gate and the subtree of which the gate is the top event. An option to do this is available in ACORN.

Subtree lowering is done by altering the entries in the LVL array for all the subtree components. The procedure is as follows. Assume the subtree is to be lowered by N levels and the entry for the gate at the top of the subtree is at $LISC(I)$. Incrementing the $LVL(I)$ entry by N gives the new physical level for the gate. Next all the components of all the partial paths starting at the gate must also be lowered by N levels. It is not necessary to trace all these partial paths to effect this lowering. A simple modification of the path tracing procedure is all that is required. Thus, entries $J = LNC(I) + 1$ through $K = LNC(I + 1)$ span the LISC locations for the components from the next lower logical which input the starting component at $LISC(I)$, and entries $L = LNC(J) + 1$ through $M = LNC(K + 1)$ span the LISC locations of the components which are input to them. The process continues until it terminates when equal entries are found for the range of LNC locations encountered at some logical level.

3.3 DETERMINATION OF GEOMETRICAL COORDINATES

The conversion of the logic of the tree into geometrical coordinates for a plot is governed by the following general principles:

- Each physical level of the tree is characterized by a reference y-coordinate and all components in this physical level are located relative to this coordinate.
- The components which are inputs to a given gate are located from left to right in the order in which they appear in the logic statement defining that gate, regardless of their physical level.
- The horizontal line spanning the inputs to a gate is located in the interlevel spacing immediately below the symbol for the gate, the input components being connected below by vertical lines which may extend across physical levels and the gate being connected above at the midpoint of the line.
- Within a physical level, the x-coordinate of a component is as near as possible to the x-coordinate of its nearest neighbor to the left, after allowance is made for the restrictions imposed on the relative locations of a gate and its inputs.

These principles are implemented in subroutine XSPACE by use of two cumulative x-coordinates for each physical level, one containing the point of attachment of the last component to the interlevel line and the other containing the rightmost coordinate required for the geometrical figures of the last component.

The resulting x-coordinates of the components are stored in the XPOS array in the same order as the components are referenced in LISC. The x-coordinates for the beginning and end of the interlevel line for a gate are stored in the XILON and XILOFF arrays respectively, and the level at which the line is drawn is stored in the LILL array, all in the order in which the gates are referenced in LISC. Since this ordering is not convenient for drawing the tree, these arrays are rearranged for the plot. The XPOS and XILON arrays are rearranged in order of increasing x-coordinate, the XILOFF and LILL arrays rearranged in step with XILON. To retain a necessary relationship between XPOS and LISC, an additional array, JARY, is introduced to contain the position of the original entry in the XPOS array.

3.4 PROCESSING OF DESCRIPTIVE LABELS

The descriptive text for a component is divided into lines according to the following procedure:

- 1) Any blanks preceding the first nonblank character or following the last nonblank character of the text are dropped and, if the resulting number of characters exceeds the number permitted (maximum number of lines times maximum number of characters per line), the excess characters are dropped.
- 2) If the text will fit on one line, it is accepted immediately; if not, the text is divided into words by locating the blanks.
- 3) If the text can possibly fit on two lines, trial divisions at words are made until the product of the number of characters in each line is maximized. If the resulting lines are short enough, this division is accepted.
- 4) Trial divisions at the words into three lines are next made, again maximizing the product of the number of characters in each line. If the resulting lines are short enough, this division is accepted.
- 5) If the maximum product test fails, as many words as possible are put in the first two lines and, if the last line is short enough, the division is accepted.
- 6) Another try at dividing the text by words is made, this time by putting as many words as possible into the last two lines and accepting the division if the first line is short enough.
- 7) If none of the above divisions by words has been successful, the text is arbitrarily divided to put the maximum number of characters in each of the first two lines and the remainder in the third line.

The maximum number of lines and the maximum number of characters per line for gates can differ from those for basic events. The maximum number of characters per line for the inhibition of an inhibit gate is the same as for other gates, but the maximum number of lines is limited to two.

3.5 DESIGN OF GEOMETRICAL FIGURES

Two kinds of geometrical figures appear on a tree: a symbol indicating by its shape the type of component and a rectangular figure enclosing a descriptive label. The shapes of these figures are shown in the output of the sample case. The symbols are designed so that the identification of a component--the letter A or X followed by a number with a maximum of four digits--can appear inside the symbol, the letter on one line and the number below. The rectangle for a normal descriptive label is limited to three lines of text, the rounded rectangle for an inhibition limited to two, and the number of characters in a line cannot exceed 25.

The dimensions of these figures are intimately related to the dimensions of the letters used by the plotter. The letter dimensions in turn depend upon the software package for the plotter. The figures are therefore designed with the following five basic generalized dimensions:

- h = letter height
- w = letter width
- s = spacing between letters in a line
- b = spacing between lines
- c = minimum clearance between a letter and the periphery
of the symbol

The relative values of the first three of these are built into the software package and are not normally under the control of the user. For the CDC system at BNW, these values are in the ratio 7:4:3. In these designs, $b = c = 2$ has been used for the symbols and $b = c = 3$ has been used for the rectangles. These seven values appear as default values in ACORN; however, they can be overwritten via the Namelist input.

The designs are subject to the following additional conditions:

- If the maximum number in the identification of any component is three digits or less, the symbols are designed for a three-digit maximum.
- An OR gate is composed of three 60° arcs.
- An AND gate is the same height as an OR gate, provides minimum clearance for a maximum digit number in width, and is capped by a semicircle.

- A House is the same height and width as an AND gate and the roof has a 30° slope.
- A Diamond consists of two equilateral triangles base-to-base.
- The diameter of a Circle is the same as the height of a Diamond.
- An Inhibit gate is a Diamond with its side vertices truncated to provide at least minimum side clearance for a maximum digit number, and forming a regular hexagon if possible.
- The sloping sides of a Triangle have minimum letter clearance for both the letter and a maximum digit number.
- A Rounded Rectangle is capped at either end by a quarter circle.

The application of elementary geometry and algebra to these conditions leads to the design formulas which are given in Table 1.

TABLE 1. Design of Geometric Figures¹

Symbol	Dimensions	Lettering		Remarks
		Offset ^{2,3}	Origin	
Rectangle	H = (L - 1)(h + b) + h + 2c	$\Delta X = \frac{1}{2}[s - m_x (w + s)]$	center	
	W = (M - 1)(w + s) + w + 2c	$\Delta Y = \frac{1}{2}[b + (k - 2e)(h + b)]$		
Rounded Rectangle	H = 2h + b + 2c	$\Delta X = \frac{1}{2}[s - m_x (w + s)]$	center	$f = \sqrt{2} - [1 - 4(\sqrt{2} - 1)\frac{c}{H}]^2$
	W = (M - 1)(w + s) + w + fh	$\Delta Y = \frac{1}{2}[b + (k - 2e)(h + b)]$		
.OR. Gate	H = $\sqrt{3}$ $\gamma_b/2$	$\Delta X = \frac{1}{2}(w - w)$	l.l.c.	$d = 1 - \sqrt{3}/2$
	W = γ_b	$\Delta Y = d_1 b + c + h + b$		γ_b defined below ⁴
.AND. Gate	H = H _{or}	$\Delta X = \frac{1}{2}(w - w)$	l.l.c.	$\gamma_a = w/2 = \text{radius of hemispherical cap}$
	W = (N - 1)(w + s) + w + 2c	$\Delta Y = c + h + b$		
House	H = H _{or}	$\Delta X = \frac{1}{2}(w - w)$	l.l.c.	30° slope of roof
	W = (N - 1)(w + s) + w + 2c	$\Delta Y = c + h + b$		
Diamond	H = W/ $\sqrt{3}$	$\Delta X = -w/2$	center	$p_n = \frac{(n-1)(w+s)}{4\sqrt{3}}$
	W = (N - 1)(w + s) + w + 2g	$\Delta Y = \frac{1}{2}[b + 2p_n]$		g defined below ⁵
Circle ⁶	R = $\frac{1}{2}$ H _{diamond}	$\Delta X = -w/2$	center	$p_n = \frac{(n-1)(w+s)[(n-1)(w+s)+2w]}{8(2h+b)}$
		$\Delta Y = \frac{1}{2}[b + 2p_n]$		

TABLE 1. Design of Geometric Figures¹ (cont'd)

Inhibit Gate	$H = H_{\text{diamond}}$	$\Delta X = -w/2$	center	$P_n = \frac{(n-1)(ws)}{4\sqrt{3}}$ g defined below ⁷
	$W = (N-1)(w+s) + w + 2g$	$\Delta Y = \frac{1}{2}[b + 2p_n]$		
Triangle	$H = h + b + (g + wh)\tan\theta$	$\Delta X = \frac{1}{2}(w - w)$	l.l.c.	$\tan\theta = \frac{2(h+b)}{(M-1)(ws)}$
	$W = (N-1)(w+s) + w + 2g$	$\Delta Y = c + h + b$		$g = (h+c) \cot \theta + c \csc \theta$

1. Definitions

L = Maximum number of lines in a rectangle

M = Maximum number of characters per line in a rectangle

k = Number of lines in a rectangle

λ = Index of a line

m_λ = Number of characters in λ th line

N = Maximum number of digits in the i.d. for a symbol

n = Number of digits in the i.d.

2. Offset - incremental distance from indicated local origin in the symbol to the lower left corner (l.l.c.) of the first letter in a line.

3. The formulas as given apply to each line for the rectangles and to the letter in the i.d. for the symbols. The additional incremental offset for the number for a symbol is in every case

$$\Delta X = -(h+b) \quad \Delta Y = -(n-1)(w+s)/2$$

TABLE 1. Design of Geometrical Figures¹ (cont'd)

4. For the .OR. Gate

$$y_b = \text{Max } \frac{1}{2i} \left[B + Pi + \left\{ (B + Pi)^2 + 4\alpha (y + qi) \right\}^{\frac{1}{2}} \right] \quad i = 1, 2$$

$$\alpha = \frac{3}{4} - d^2 = \sqrt{3} - 1$$

$$\beta = 2c + \frac{w}{2} + 2d(c + h)$$

$$y = \frac{w^2}{4} + (c + h)^2 - c^2$$

$$Pi = 2d(h + b) \quad qi = (h + b)(3h + b + 2c) \quad i = 1 \quad (1\text{-letter line})$$

$$= \frac{(N-1)(w+s)}{2} = Pi(Pi + w) \quad i = 2 \quad (N\text{-digit line})$$
5. For the Diamond

$$g = \frac{1}{2} \left[\sqrt{3}(2h+b) + 4c - \frac{(N-1)(w+s)}{2} \right]$$
6. the minimum clearance in the circle for a four digit i.d. is slightly less than c in this design.
7. for the Inhibit Gate

$$g = c + \text{Max} \left[0, \frac{1}{4} \left\{ \sqrt{3} (2h + b) - w - \frac{3(N-1)(w+s)}{2} \right\} \right]$$

4.0 ADDITIONAL PROGRAMMING INFORMATION

In this section, additional information useful for program modification or conversion is given. After some brief general information, the individual subroutines are named and their function described. This is followed by a description of blank common and identification of the integer and real variables and of the dynamically dimensioned arrays.

4.1 GENERAL INFORMATION

ACORN is written in Fortran-IV, except for a utility library routine EXFL, which is written in Compass. There are a few statements which are nonstandard Fortran. Some statements, principally in subroutines INLOG and INLBL, specify certain variables via the R format to facilitate comparison tests of Hollerith characters. A DECODE statement is used in subroutine DRAW to do a formatted read from one array to another in blank common. A function EOF(LU) is called after reads of an input unit so that program control can be maintained when the end of input data is reached.

Central memory requirements depend on the size of the problem; as presently compiled and operated on the CDC Cyber 74 under Scope 3.4.2, a minimum of about 27,000 words is required when default values of the parameters are used. The following logical units are used:

- 5 Normal input
- 6 Printed output
- 8 Data for plotter
- 2 Punched card output/alternate selective input

4.2 SUBROUTINES AND THEIR FUNCTION

The functions of the more important subroutines are described below in the order in which they are used.

ACORN - Main program. Reads NAMELIST input. Initializes parameters. Sets pointers for arrays and calls EXPL for dynamic allocation of central memory. Calls INLOG, EDIT, ORDER, PNCH, LOWER, YSPACE, XSPACE, PRPLOT, INLBL, DRAW.

INLOG - Reads and analyzes input logic statements. Fills basic arrays IDNO, INAR, and INDX. Identifies types of each gate and event. Determines NG, NIET, NIDT, IP. Sets IPROC = 1 if errors fatal to a plot are encountered. Called by ACORN.

ORDER - Develops basic tree logic from contents of IDNO, INAR and INDX arrays, filling LISC, LNC, and LVL arrays. Flags undefined gates with transfer-in triangles. Adds special transfer-out triangles if requested. Determines NC, NLL. Called by ACORN.

LOWER - Lowers by a given number the physical levels of a given gate and its subtree. Reads input to determine the gate and number. Determines NPL. Called by ACORN.

YSPACE - Finds the letter height and fills the YIL array with the Y-coordinate of the interlevel lines. Scales basic dimensions of tree symbols to inches. Calls DESIGN. Called by ACORN.

XSPACE - Determines the X-Coordinate for each appearance of each component in the tree. Fills LILL, XPOS, XILON, XILOFF arrays. Called by ACORN.

PRPLOT - Rearranges XPOS (= XARY), LILL, XILON, XILOFF arrays in plotting order. Does a line printer plot of the tree, a maximum of 22 top levels. Called by ACORN if no errors fatal to plot have been detected.

INLBL - Reads the descriptive labels for the rectangles and divides the descriptive labels into lines. Fills LABL, LBLLOC, NCPL arrays. Called by ACORN.

DRAW - Determines plotting coordinates and monitors the plotting. Calls ANGATE, ARC, DIMOND, FNAME, HEXAGN, HOUSE, INHBN, LABEL, ORGATE, PNAME, RECT, TRANGL, and CalComp PLOT, SYMBOL. Called by ACORN.

The following are auxiliary subroutines used in the design and drawing of the geometrical figures and in their labeling.

DESIGN - Determines relative dimensions of symbols in terms of basic height/width/spacing ratios of plotter symbols and maximum number of digits in component names. Called by YSPACE.

RECT - Draws rectangle. Calls CalComp PLOT. Called by DRAW.

TRANGL - Draws triangle. Calls CalComp PLOT. Called by DRAW.

DIMOND - Draws diamond. Calls CalComp PLOT. Called by DRAW.

ARC - Draws a circular arc. Calls CalComp PLOT. Called by DRAW, ANGATE, ORGATE, INHBN.

ANGATE - Draws AND-gate. Calls ARC, CalComp PLOT. Called by DRAW.

ORGATE - Draws OR-gate. Calls ARC, CalComp PLOT. Called by DRAW.

HOUSE - Draws house. Calls CalComp PLOT. Called by DRAW.

HEXAGN - Draws hexagon. Calls CalComp PLOT. Called by DRAW.

INHBN - Draws rounded rectangle for inhibition label. Calls ARC, CalComp PLOT. Called by DRAW.

LABEL - Prints the descriptive labels in the rectangles. Calls CalComp PLOT, SYMBOL. Called by DRAW.

FNAME - Determines number of digits in numerical part of component name. Called by DRAW.

PNAME - Plots the component name. Calls CalComp PLOT, SYMBOL. Called by DRAW.

The following are auxiliary subroutines used in editing and punching services.

EDIT - Checks consistency of input data and prints logic ordered by gate number and by component. Prints basic tree logic after it is determined. Calls WHICH and WHEREIN. Called by ACORN.

ARRGE - Arranges a call list array in ascending numerical order. Called by WHICH, WHEREIN, PNCH.

WHEREIN - Locates the gates to which a component is input and arranges them in ascending numerical order. Called by EDIT. Calls ARRGE.

WHICH - Locates the components which are input to a gate and arranges them in ascending numerical order. Called by EDIT. Calls ARRGE.

PNCH - Punches logic cards for input to program MFAULT⁽²⁾. Called by ACORN. Calls ARRGE.

The auxiliary subroutine EXFL, written in Compass, was obtained from a Fortran Utility Library. This subroutine extends central memory to the size needed by the case, but subject to installation or jobcard limits. At the loss of flexibility in determining central memory requirements, this subroutine can be removed, its calls from ACORN deleted, and the dimension of the IAR array increased to a sufficient size.

4.3 CONTENTS OF BLANK COMMON

Blank common is organized as follows:

1) Three small arrays

IPAR(30) - An array providing space for most of the integer parameters appearing in the Namelist group PARS. Only the first 16 positions of IPAR are now used, by equivalencing to the integer parameters.

RPAR(15) - An array providing space for many of the real parameters appearing in the Namelist group PARS. Only the first seven positions of RPAR are now used, by equivalencing to the real parameters.

TITLE(8) - An array providing space for one card of title information. This title appears in several places in the output and may optionally appear on the plot of the tree.

2) A set of integer variables, defined below.

3) A set of real variables, defined below.

4) The dynamically dimensioned array, equivalently IAR or AR, and formally dimensioned to 1. Its usable length is set by the auxiliary subroutine EXFL.

The integer variables, in the order of their occurrence, are:

IC - The total number of distinct components in the tree, that is, gates plus basic events, with TOP counted as a gate. IC is the number of entries in the IDNO array. Used in INLOG as a counter to determine this number.

IP - The total number of entries on the r.h.s. of the logic statements, that is, input events to gates. Multiple appearances are counted. IP is the number of entries in the INAR array. Used in INLOG as a counter to determine this number. After NC is defined in ORDER, IP is used in PRPLOT as a variable.

IPROC - Used as a flag to bypass a plot if INLOG analysis of logic statements detects an error which would be fatal to the plot. Normally 0, set to 1 if fatal error found.

KADDA - The increment added to the input i.d.s of .AND. gates to distinguish them from .OR. gates. I.d.s of .OR. gates are not incremented. Value set in ACORN. $KADDA < KADDC$.

KADDC - The increment added to the i.d.s of X circles to distinguish them from gates. Value set in ACORN. $KADDC < KADDD$. Thus the internal i.d.s of gates are $< KADDC$, of basic events are $> KADDC$.

KADDD - The increment added to the i.d.s of X diamonds to distinguish them from gates and X circles. Value set in ACORN. $KADDD > KADDC$.

LADD - The increment added to the LISC entry for a specific appearance of a gate to signal that the input logic for that appearance is not to be traced, i.e., that a transfer-in triangle is to be used. Value set in ACORN.

LCARD - Maximum number of columns of text of a descriptive label to be read from a card. Set to 75 in ACORN.

MAXLIN - Maximum number of lines of labeling in a rectangle. Set to MXLIN1 for gates, MXLIN2 for basic events.

NCPW - Number of characters of label per word of storage. Set to 10 for CYBER version.

NG - Number of distinct defined gates in the tree.

NGP - $NG + 1$, thus including TOP as a separate gate.

NIET - Total number of distinct components, excluding dummy components for any inhibitions present.

NL - Total number of levels in the tree. Used as counter of logical levels in ORDER, modified to physical levels in LOWER.

NTR - Multiple trace flag, set in ACORN according to value of NTRACE.
 NTR = 1 for single trace in plotting, = 2 for retrace.

NTIN - Logical unit for reading tree logic and label information. If
 LUIN = 0, NTIN = 5; if LUIN > 0, NTIN = 2. If LUIN > 0, user is
 responsible for control card assignment of Logical Unit 2.

NC - Total number of entries in LISC array. Should equal IP. Used in
 ORDER as a counter of the LISC entries.

NGT - Number of distinct gates in the tree, including undefined gates.

NIDT - Total number of distinct components including dummy components for
 inhibitions. Should equal final value of IC.

IWDC - Total number of stored words of descriptive labels. Used in INLBL
 as a counter of the LABL entries.

KADDH - The increment added to the i.d.s of houses to distinguish them from
 other components. Value set in ACORN. $KADDH = KADDC + KADDI$.

KADDI - The increment added to the i.d.s of inhibit gates to distinguish
 them from other gates. Value set in ACORN. $KADDI = 2 * KADDA$.

MND - Maximum number of digits in the numerical part of a component name.
 Determined in INLOG. Used in DESIGN in determining size of tree symbols.

BLANK - A Hollerith integer word containing a blank character right adjusted,
 zero filled to the left.

STAR - A Hollerith integer word containing four asterisk characters right
 adjusted, zero filled to the left.

The real variables, in the order of their occurrence, are:

DXES - Minimum spacing between adjacent rectangles for events which are
 input to the same gate. Set to two times letter height.

DXGS - Minimum spacing between adjacent rectangles for events which are input
 to different gates in the same level. Set to five times letter height.

DXLTR - Minimum spacing between characters in a label. Set to two letter
 units.

DXTS - Spacing between a rectangle and its transfer-out triangle. Set to one
 letter height.

DYCIR - Incremental y spacing for positioning top of circle symbol relative to top of rectangle.

DYDMD - Incremental y spacing for positioning top of diamond symbol relative to top of rectangle.

DYILH - Minimum y spacing between an interlevel line and any rectangles or gates attached to it. Set to 0.8 letter heights.

DYRGS - Separation between top of gate symbol and bottom of rectangle. Presently set to 0.

DZ - Maximum distance from an arc to its approximating chord, expressed in fraction of the radius of the arc. Used in plotting several tree symbols. Mathematically, $DZ = \tan^2 (DA/4.)$, where $DA = \pi/DAPQ$ and DAPQ is an optional input quantity, the number of angular segments per quadrant, with a default value of 24.

The remaining real variables are in inches:

HTLTR - Letter height

XAND - Width of .AND. gate symbol

XDMD - Width of diamond symbol

XOR - Width of .OR. gate symbol

XRECT - Width of label rectangle for gates

XTRI - Width of triangle symbol

XXRECT - Width of label rectangle for basic events

YAND - Height of .AND. gate symbol

YDMD - Height of diamond symbol

YOR - Height of .OR. gate symbol

YRECT - Height of label rectangle for gates

YTRI - Height of triangle symbol

YXRECT - Height of label rectangle for basic events

ZA - Radius of semicircular cap of .AND. gate symbol

ZB - Radius of base arc of .OR. gate symbol

ZC - Radius of circle symbol

ZS - Radius of side arcs of .OR. gate symbol

DXTRS - Separation between an inhibit gate symbol and rounded rectangle.

Set to 1.5 times letter height.

XHSE - Width of house symbol
YHSE - Height of house symbol
XINH - Width of inhibit gate symbol
YINH - Height of inhibit gate symbol
XIRECT - Width of inhibit gate rounded rectangle
YIRECT - Height of inhibit gate rounded rectangle

The principal arrays are described below in the approximate order in which they are first used by ACORN. The variable dimension of each array is given in parentheses after the array name. The pointers for these arrays are all defined in the main program as local integer variables. They are readily recognized as the array names, sometimes slightly altered, prefixed by the letter M.

IDNO (NIDT) - The basic array of identification numbers (i.d.s) of the components of the tree. I.d.s are biased by adders to indicate type of component. Entries are made in INLOG and unchanged thereafter. The first NGP entries are the i.d.s of the defined gates, in the order of their definition (appearance on the l.h.s. of an input logic statement), with TOP given an i.d. of 0. Undefined gates, if any, appear in positions NGP + 1 through NGT. Positions NGT + 1 through NIET contain i.d.s for the events, in the order of their first appearance on the r.h.s. of an input logic statement. Dummy events, for the inhibitions of any inhibit gates, appear in positions NIET + 1 through NIDT.

INAR (IP) - An array of input data containing the i.d.s of the components, in their order of appearance on the r.h.s. of the input logic statements. Each appearance of a component results in an entry in INAR. I.d.s are biased by adders as in IDNO. Entries are made in subroutine INLOG. After the INAR entries are rearranged in basic logic order into array LISC in subroutine ORDER, the INAR array is no longer used. Its storage space is later used by the ITGL array.

INDX (NGT + 1) - An index array coupling the INAR entries, which are inputs to gates, to the respective gates in the IDNO array. $INDX(1) = 0$ and $INDX(I + 1)$ contains the position in the INAR array of the last component which is input to the gate which has its i.d. at $IDNO(I)$. Thus

the components of the INAR array in position $INDX(I) + 1$ through $INDX(I + 1)$ are input to the gate at $IDNO(I)$. Entries are made in subroutine INLOG. The INDX entries are used in subroutine ORDER in constructing the LISC array and are used in EDIT(1), but not thereafter. The INDX storage space is later used by the LILL array.

LISC (NC) - The array of basic logical order of the tree. LISC is a rearrangement and transformation of the INAR array. The rearrangement consists of reordering the INAR entries from order of input to that of basic logic flow, i.e., from top to bottom via logical levels and from left to right within a logical level. The transformation replaces the i.d.s of the INAR array by the position of these i.d.s in the IDNO array. A gate with a transfer-out triangle is flagged with a negative sign, a gate with a transfer-in triangle is flagged by adding a large increment, LADD. The LISC entries are made in subroutine ORDER and are used throughout the remainder of the program.

LNC (NC) - An indexing array coupling positions within the LISC array. $LNC(1) = 1$ and $LNC(I + 1)$ contains the position within the LISC array of the last component which is input to the gate at $LISC(I)$. Thus, the component at the LISC positions referenced at positions $LNC(I) + 1$ through $LNC(I + 1)$ are input to the gate at $LISC(I)$. If $LNC(I + 1) = LNC(I)$, the component at $LISC(I)$ is not a gate with input. The LNC entries are made in subroutine ORDER and are used throughout the remainder of the program.

LVL (NC) - A companion array to LISC. The $LVL(I)$ entry contains the number of the physical level in which the component referenced at $LISC(I)$ is located. Initially the LVL entries are set in subroutine ORDER to the logical levels of the components. These entries may be altered in subroutine LOWER by use of the lowering option, after the basic logic flow of the tree has been determined. The LVL array is used throughout the remainder of the program.

ITGL (NC) - A companion array to LISC. The $ITGL(I)$ entry contains the number of the physical level of the gate to which the component at $LISC(I)$ is input. If no gate lowering is involved, $ITGL(I) = LVL(I) - 1$. The ITGL entries are made in XSPACE and used in PRPLOT and DRAW. The storage space used is that previously occupied by INAR.

XREL, XPOS, XARY (NC) - Three arrays occupying the same memory space, an entry containing the x-coordinate of the center of the symbol for a component. The XREL entries are made first, in subroutine XSPACE, at which time XREL(I) contains the x-coordinate for the component at LISC(I), relative to an origin at the x-coordinate of the gate to which that component is input. When these coordinates are later converted to a single origin, in subroutine XSPACE, the array is called XPOS. The coordinates are rearranged in subroutine PRPLOT in order of increasing x and the array renamed XARY. The data thereafter are used either as XARY or XPOS. The relationship between XPOS and LISC, directly positional before the rearrangement, is contained in the JARY array after rearrangement.

XILON (NGP) - An array containing the x-coordinate for the beginning of the interlevel line connecting the inputs to a gate. As entered in subroutine XSPACE, the XILON(I) coordinate refers to the gate with i.d. at IDNO(I). The array is rearranged in subroutine PRPLOT in order of increasing x. Loss of relationship between XILON and IDNO due to the rearrangement is immaterial.

XILOFF (NGP) - A companion array to XILON, containing the x-coordinate of the end of the interlevel line connecting the inputs to a gate. Entries are made in subroutine XSPACE as in XILON, and rearranged in subroutine PRPLOT to the same ordering as XILON.

LILL (NGP) - A companion array to XILON, containing the number of the physical level at which the interlevel line connecting the inputs to a gate is to be drawn. Entries are made in subroutine XSPACE, as in XILON, and rearranged in subroutine PRPLOT to the same ordering as XILON. The array uses the memory space previously used by INDX.

YIL (NPL) - The array of y-coordinates of the interlevel lines of the tree. Since y is measured from the bottom of the plot and the levels are counted from the top, the entries are in decreasing numerical order. Entries are made in subroutine YSPACE and unchanged thereafter.

LORDR (NIDT) - An array relating the sequential order in which labels for the components are read in to the order in which the components were

defined. The entry at LORDR(I) is the sequential order in which the label for the component with i.d. at IDNO(I) was read. The array is initialized in subroutine INLBL and unchanged thereafter.

LBLLOC (NIDT) - An array containing the locations of the labels for the components. The entry at LBLLOC(I) is the location in the LABEL array of the first processed word of the Ith label which was read in. The array is initialized in subroutine INLBL and unchanged thereafter.

NCPL (3, NIDT) - An array containing the number of characters in each of the three possible lines of a label for each component. The entry at NCPL(L,I) is the number of characters in the Lth line of the Ith label which was read in. The array is initialized in subroutine INLBL and unchanged thereafter.

LABL (9*NIDT) - An array containing the processed labels for the components in the order in which the labels were read in. The processing consists of removing any leading and trailing blanks and then dividing the remaining text into a maximum of three lines. The first nonblank character of each line starts in the leftmost position of a new word and as many words as needed are used for the line, any remaining character positions being filled with blanks. No word is used for a missing line. Memory space is allocated for 9*NIDT words. The array is initialized in subroutine INLBL.

JARY (NC) - An array relating XPOS to LISC. The JARY(I) entry contains the position in the LISC array of the component with x-coordinate of XPOS(I). The array is initialized in subroutine PRPLOT.

REFERENCES

1. T. H. Smith, P. J. Pelto, D. L. Stevens, G. D. Seybold, W. L. Purcell, and L. V. Kimmel, A Risk-Based Fault Tree Analysis Method for Identification, Preliminary Evaluation, and Screening of Potential Accidental Release Sequences in Nuclear Fuel Cycle Operations, BNWL-1959, Battelle, Pacific Northwest Laboratories, Richland, WA, January 1976.
2. P. J. Pelto and W. L. Purcell, MFAULT, A Computer Program For Analyzing Fault Trees, BNWL-2145, Battelle, Pacific Northwest Laboratories, Richland, WA, October 1977.
3. G. D. Seybold, RAFT, A Computer Program for Fault Tree Risk Calculations, BNWL-2146, Battelle, Pacific Northwest Laboratories, Richland, WA, October 1977.
4. Reactor Safety Study - An Assessment of Accident Risks in U.S. Commercial Nuclear Power Plants, Appendix II Fault Trees, WASH-1400 (NUREG-75/014), October 1975.
5. H. E. Lambert, Systems Safety Analysis and Fault Tree Analysis, UCID-16238, Lawrence Livermore Laboratory, University of CA, May 1973.
6. P. A. Crossetti, Fault Tree Simulation Computer Program, DUN-1697, June 1971.

DISTRIBUTION

No. of
Copies

No. of
Copies

OFFSITE

UNITED STATES

A. A. Churm U.S. DOE Chicago Patent Group 9800 South Cass Avenue Argonne, IL 60439	R. E. Barlow UCLA, Berkeley Berkeley, CA 94720
27 U.S. DOE Technical Information Center	G. R. Burdick Aerojet Nuclear Company 550 Second Street Idaho Falls, ID 83401
W. Brobst U.S. DOE Division of Environmental Control Technology Transportation Branch U.S. DOE Headquarters Washington, DC 20545	H. C. Claiborne Union Carbide Corporation (HNL) P. O. Box Y Oak Ridge, TN 37830
J. A. Sisler U.S. DOE Division of Environmental Control Technology Transportation Branch U.S. DOE Headquarters Washington, DC 20545	J. Curtis Department of Transportation Materials Transportation Bureau 2100 Second St. S.W. Washington, DC 20590
W. K. Eister U.S. DOE Division of Nuclear Fuel Cycle and Production Washington, DC 20545	W. S. Durant duPont Company, Aiken (U.S. DOE) E. I. duPont DeNemours and Co. Savannah River Laboratory Aiken, SC 29801
R. D. Walton U.S. DOE Division of Nuclear Fuel Cycle and Production Washington, DC 20545	D. Egan Environmental Protection Agency 5207 Longsdale Drive Springfield, VA 22151
G. E. Apostolakis UCLA 5532 Boelter Hall Los Angeles, CA 90024	R. C. Erdmann Science Applications, Inc. 2680 Hanover Street Palo Alto, CA 94303
	R. R. Fullwood Science Applications, Inc. 2680 Hanover Street Palo Alto, CA 94303

No. of
Copies

J. B. Fussell
University of Tennessee
Department of Nuclear Engineering
Knoxville, TN 37916

D. F. Haas1
Institute of System Sciences
3085 North Bar Drive
North Bend, OR 97459

J. D. Hill
Battelle Memorial Institute
505 King Avenue
Columbus, OH 43201

R. Howard
Stanford University
Stanford, CA 93045

B. R. Judd
Management Analysis Co.
Palo Alto, CA 94303

H. E. Lambert
Lawrence Livermore Laboratory
Livermore, CA 94550

G. S. Lellouche
Electric Power Research Institute
3412 Hillview Avenue
P.O. Box 10412
Palo Alto, CA 94303

S. E. Logan
University of New Mexico
Albuquerque, NM 87131

D. Okrent
UCLA
5532 Boelter Hall
Los Angeles, CA 90024

T. G. Priddy
Sandia Laboratories
P.O. Box 5800
Albuquerque, NM 87115

No. of
Copies

N. Rasmussen
Massachusetts Institute of
Technology
Cambridge, MA 02139

W. Rowe
Environmental Protection Agency
401 M. Street
Washington, DC 20460

P. Rubel
Union Carbide Corporation (HNL)
P.O. Box Y
Oak Ridge, TN 37830

T. H. Smith
EEG Company
Idaho Falls, ID 83401

D. Turner
Union Carbide Corporation
Office of Waste Isolation
P.O. Box Y
Oak Ridge, TN 37830

W. E. Vesely
NRC Office of Nuclear Regulatory
Research
Washington, DC 20545

I. B. Wall
NPC Office of Nuclear Regulatory
Research
Washington, DC 20545

R. Williams
Electric Power Research Institute
3412 Hillview Avenue
P.O. Box 10412
Palo Alto, CA 94304

FOREIGN

G. D. Bell
United Kingdom Atomic Energy
Authority
Safety and Reliability
Directorate
Warrington WA3 4NE
UNITED KINGDOM

No. of
Copies

H. W. M. Braun
C.C.R. EURATOM
21020 Ispra (Va)
C.P. 5
ITALIA

H. Dyroff
NUKEM GmbH
645 Hanau 11
Postfach 110080
WEST GERMANY

Ferruccio Gera
Laboratorio Rifiuti
Radioattivi
C.N.E.N. - C.S.N. Casaccia
Casella Postale 2400
00100 Rome, ITALY

S. Hartwig
Battelle Institute, e.v.
Am Romerhof 35
600 Frankfurt Main 90
GERMANY

International Atomic Energy
Agency
Kärtner Ring 11
P.O. Box 590
A-1011, Vienna, AUSTRIA

H. Krause
Nuclear Research Center
Waste Management Department
D75 Karlsruhe
Weberstr. 5
GERMANY

M. Stammer
Battelle-Institute, e.v.
6000 Frankfurt am Main 90
Am Romerhof 35
Frankfurt, GERMANY

No. of
Copies

H. J. Wingender
NUKEM GmbH
645 Hanau 11
Postfach 110080
WEST GERMANY

H. Witte
NUKEM GmbH
645 Hanau 11
Postfach 110080
WEST GERMANY

ONSITE

3 US DOE Richland Operations
Office

H. E. Ransom
R. B. Goranson
M. J. Zamorski

4 Rockwell Hanford Operations

R. A. Deju
R. E. Isaacson
D. D. Wodrich
File Copy

5 Exxon

J. L. Carter

1 Joint Center for Graduate Study

J. Cooper

1 United Nuclear Industries, Inc.

P. A. Crosetti

3 HEDL

J. E. Ecker
J. W. Hagen
D. E. Simpson

No. of
Copies

64 Battelle-Northwest

N. E. Carter
J. C. Fox
R. J. Hall
S. W. Heaberlin
J. W. Lichfield
T. I. McSweeney
E. S. Murphy
P. J. Pelto (10)
A. M. Platt
W. L. Purcell (10)
R. E. Rhoads
G. D. Seybold
D. L. Stevens
J. W. Voss
L. D. Williams
W. K. Winegardner (25)
Technical Information (5)
Technical Publications