

UNCLASSIFIED

AD NUMBER

ADB018527

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited. Document partially illegible.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; AUG 1976. Other requests shall be referred to Air Force Armament Lab., Eglin AFB, FL. Document partially illegible.

AUTHORITY

USADTC ltr 10 Dec 1979

THIS PAGE IS UNCLASSIFIED

THIS REPORT HAS BEEN DELIMITED  
AND CLEARED FOR PUBLIC RELEASE  
UNDER DOD DIRECTIVE 5200.20 AND  
NO RESTRICTIONS ARE IMPOSED UPON  
ITS USE AND DISCLOSURE.

DISTRIBUTION STATEMENT A

APPROVED FOR PUBLIC RELEASE;  
DISTRIBUTION UNLIMITED.

**COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION**

②

**AFATL-TR-76-84 VOLUME II**

AD B018527

**SYNERGISTIC EFFECTS OF MINEFIELDS AND COVERING FIRE (SEMAC) COMPUTER MODEL**

**VOLUME II. ANALYST'S MANUAL**

**BOOZ, ALLEN & HAMILTON INC.  
362 BEAL PARKWAY N.W.  
FORT WALTON BEACH, FLORIDA 32548**

**AUGUST 1976**

**FINAL REPORT: JUNE 1975 - APRIL 1976**



Distribution limited to U. S. Government agencies only; this report documents test and evaluation; distribution limitation applied August 1976. Other requests for this document must be referred to the Air Force Armament Laboratory (DLYW), Eglin Air Force Base, Florida 32542.

**AIR FORCE ARMAMENT LABORATORY**

**AIR FORCE SYSTEMS COMMAND • UNITED STATES AIR FORCE**

**EGLIN AIR FORCE BASE, FLORIDA**



S/C 407763

AD No. DDC FILE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

<b>19 REPORT DOCUMENTATION PAGE</b>		<b>READ INSTRUCTIONS BEFORE COMPLETING FORM</b>	
1. REPORT NUMBER AFATL-TR-76-84, Volume II	2. GOVT ACCESSION NO. 2	3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) SYNERGISTIC EFFECTS OF MINEFIELDS AND COVERING FIRE (SEMAC) COMPUTER MODEL, VOLUME II. ANALYST'S MANUAL.		5. TYPE OF REPORT & PERIOD COVERED Final Report, - June 1975 - Through April 1976	
7. AUTHOR(s) David O. Fraser Clayton Thomas		6. PERFORMING ORG. REPORT NUMBER	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Booz, Allen & Hamilton, Inc. 362 Beal Parkway, N.W. Fort Walton Beach, Florida 32548		8. CONTRACT OR GRANT NUMBER(s) F08635-75-C-0151	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Armament Laboratory Armament Development and Test Center Eglin Air Force Base, Florida 32542		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Element ESP 921H JON 9134-01-11	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 12 296p.		12. REPORT DATE August 1976	
		13. NUMBER OF PAGES	
		15. SECURITY CLASS. (of this report) Unclassified	
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report) Distribution limited to U.S. Government agencies only; this report documents test and evaluation; distribution limitation applied August 1976. Other requests for this document must be referred to the Air Force Armament Laboratory (DLYW), Eglin Air Force Base, Florida 32542.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
18. SUPPLEMENTARY NOTES Available in DDC			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Computer Simulation Model SEMAC Minefields Covering Fire Monte Carlo Techniques			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This computer simulation model, referred to as SEMAC, provides the methodology and analytical techniques required for evaluating the synergistic effects of minefields and covering fire. The model was specifically designed to consider mixed minefields, armored vehicle tactics, and the employment of combinations of different types of direct and indirect fire. Methodology is included to evaluate the effectiveness of rollers and plows, as well as line charges and fuel air explosive devices employed in a			

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

407763

UNCLASSIFIED  
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

next page  
418

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

minesweeping role. SEMAC is an event-oriented model which uses Monte Carlo techniques to simulate the passage of up to 100 intruder targets of up to five types through an engagement area. The model can evaluate the effectiveness of up to 20 direct fire defenders and up to 10 indirect fire volley aimpoints. The computer program was specifically designed for the Control Data Corporation 6600 computer system at Eglin Air Force Base, Florida.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

PREFACE

This report documents work accomplished during the period 10 June 1975 through 10 April 1976 by Booz, Allen & Hamilton, Inc., 362 Beal Parkway, N.W., Fort Walton Beach, Florida, under Contract Number F08635-75-C-0151 with the Air Force Armament Laboratory, Armament Development and Test Center, Eglin Air Force Base, Florida. The program monitor for the Armament Laboratory was Mr. Charles A. Reynolds (DLYW).

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

*J.R. Murray*  
J. R. MURRAY  
Chief, Weapon Systems Analysis Division

61	White Section	<input type="checkbox"/>
62	Blue Section	<input checked="" type="checkbox"/>
63	PERFORMED	<input type="checkbox"/>
64	IDENTIFICATION	<input type="checkbox"/>
65	OPERATION/AVAILABILITY DATA	
66	REMARKS/SPECIAL	
<b>B</b>	23	

*DFZ*

## TABLE OF CONTENTS

Section	Title	Page
I.	INTRODUCTION . . . . .	1
II.	MATHEMATICAL MODEL . . . . .	4
	Computation of Effectiveness of Mines . . . . .	4
	Computation of Effectiveness of Direct Fire Munitions . . . . .	18
	Computation of Effectiveness of Indirect Fire Munitions. . . . .	23
III.	FLOWCHARTS . . . . .	30
IV.	SOURCE LISTING . . . . .	110
V.	SIMULATION MODEL . . . . .	161
	Tables and Arrays . . . . .	161
	Discussion of Simulation . . . . .	162
	Program MAIN . . . . .	162
	Subroutine READIN . . . . .	167
	Subroutine SETUP . . . . .	175
	Subroutine ROAD . . . . .	178
	Subroutine BOOM . . . . .	183
	Subroutine SORT . . . . .	183
	Subroutine LOOPS . . . . .	184
	Subroutine EVENTC . . . . .	187
	Subroutine DIVSET . . . . .	188
	Subroutine DIVCHK . . . . .	189
	Subroutine UNIT . . . . .	190

TABLE OF CONTENTS (CONCLUDED)

Section	Title	Page
V.	SIMULATION MODEL	
	Discussion of Simulation (Concluded)	
	Subroutine TGTMIN . . . . .	192
	Subroutine SYMDET . . . . .	197
	Subroutine PRINTO . . . . .	201
	Subroutine PRINTR . . . . .	202
	Subroutine DISTR . . . . .	205
	Subroutine TABINT . . . . .	209
	Subroutine RNORM . . . . .	210
	Subroutine IPACK . . . . .	210
	Subroutine UNPACK . . . . .	211
	Subroutine NDFIRE . . . . .	211
	Subroutine DIRFIR . . . . .	216
	Subroutine CKEVTM . . . . .	217
	Subroutine STINT . . . . .	224
	Subroutine EXPSWP . . . . .	225
	Subroutine TGTMOV . . . . .	227

## LIST OF FIGURES

Figure	Title	Page
1.	Map Coordinate System . . . . .	7
2.	Transformation of Mine Locations From Map Coordinate System Into Travel Path Coordinate System . . . . .	9
3.	Geometry of Target Diversion . . . . .	13
4.	Flowchart, Program MAIN . . . . .	32
5.	Flowchart, Subroutine READIN . . . . .	39
6.	Flowchart, Subroutine SETUP . . . . .	50
7.	Flowchart, Subroutine ROAD . . . . .	52
8.	Flowchart, Subroutine BOOM . . . . .	57
9.	Flowchart, Subroutine SORT . . . . .	58
10.	Flowchart, Subroutine LOOPS . . . . .	59
11.	Flowchart, Subroutine EVENTC . . . . .	62
12.	Flowchart, Subroutine DIVSET . . . . .	64
13.	Flowchart, Subroutine DIVCHK . . . . .	65
14.	Flowchart, Subroutine UNIT . . . . .	66
15.	Flowchart, Subroutine TGTMIN . . . . .	67
16.	Flowchart, Subroutine SYMDET . . . . .	75
17.	Flowchart, Subroutine PRINTO . . . . .	77
18.	Flowchart, Subroutine PRINTR . . . . .	78
19.	Flowchart, Subroutine DISTR . . . . .	81
20.	Flowchart, Subroutine TABINT . . . . .	82
21.	Flowchart, Subroutine RNORM . . . . .	82

LIST OF FIGURES (CONCLUDED)

Figure	Title	Page
22.	Flowchart, Subroutine IPACK . . . . .	83
23.	Flowchart, Subroutine UNPACK . . . . .	84
24.	Flowchart, Subroutine NDFIRE . . . . .	85
25.	Flowchart, Subroutine DIRFIR . . . . .	89
26.	Flowchart, Subroutine CKEVTM . . . . .	92
27.	Flowchart, Subroutine STINT . . . . .	104
28.	Flowchart, Subroutine EXPSWP . . . . .	105
29.	Flowchart, Subroutine TGTMOV . . . . .	108
30.	SEMAC Source Listing . . . . .	111

LIST OF TABLES

Table	Title	Page
1.	Determination of Range and Deflection Coordinates of Mean Point of Impact . . . . .	6
2.	Bit Locations for Packed Mine Information . . . . .	11

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
a	ALWRAT	The ratio of the length of the target to the width of the target.	none
a <sub>1</sub> , a <sub>2</sub>	A1, A2	An intermediate factor used in computing the single shot probability of damage.	none
b <sub>1</sub> , b <sub>2</sub>	B1, B2	An intermediate factor used in computing the single shot probability of damage.	none
CEP	ACEP(5,2)	The circular error probable in the normal plane.	feet or mils
D(i,k)	DIS	The distance from the i <sup>th</sup> (detonating mine or exploding munition) to the k <sup>th</sup> mine.	feet
D <sub>F</sub>	DFTFAE	The distance in front of the target where the fuel air explosive sweeping device detonates.	feet
DMPI(i)		Mean point of impact in deflection.	feet
DMT	DISTPD	The distance between the mine and the target.	feet
D <sub>O</sub>	DO	The distance in deflection from a target to an indirect fire round impact point.	feet
DEP	DEP(10)	The deflection error probable in the ground plane.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
DSSP	DSSP	The single shot probability of hit in deflection.	none
EMD		The effective miss distance.	feet
H <sub>T</sub>	TARHT(5)	The height of the intruder target.	feet
I	ANGIMP(10)	The weapon impact angle.	degrees
LET	ETL	The effective target length.	feet
L <sub>P</sub>		The pattern length of the indirect fire weapon.	feet
L <sub>SH</sub>	SHADOL	The target shadow length.	feet
L <sub>T</sub>	TARL(5)	The intruder target length.	feet
MAE <sub>b</sub>		Mean area of effectiveness for blast.	square feet
MAE <sub>f</sub>		Mean area of effectiveness for fragmentation.	square feet
N <sub>S</sub>	NSUB(5)	Number of submunitions.	none
N <sub>t</sub>		The number of complete fuze cycles.	none
P	PRBITY	An interpolated value determined by Subroutine TABINT.	none
P <sub>DF</sub>	PDLCF	The probability of mine detonation within the FAE pattern.	none

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN SIMULATION MODEL	DEFINITION	UNITS
$P_{DUD}$	DUDPRB(7)	The probability that the mine type is a dud.	none
$P_{HD}$		The probability of damage given a hit.	none
$P_q, P_{q+1}$	PROBIL(840)	Probability values from tables.	none
R	AREL(5,2)	The reliability of the direct fire round.	none
$R_F$	RADFAE	The radius of the FAE effects.	feet
$R_{MPI(i)}$		The mean point of impact in range.	feet
$R_N$	RSTART	A random number from a normal distribution.	none
$R_{N1}, R_{N2}, R_{N3}, R_{N4}$	RSTART	Random numbers from a normal distribution.	none
$R_O$	RO	The distance in range from a target to an indirect fire round impact point.	feet
$R_I$	PATRAD(5)	The radius of the ICM pattern.	feet
$R_q, R_{q+1}$	RANGPR(840)	Range values from tables.	none
$R_S$	RELSUB(5)	The reliability of the submunition.	none
$R_T$	TARRAD(5)	The radius of the intruder target.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
REP	AREP(20)	The range error probable in the ground plane.	feet
RSSP	RSSP	The range single shot probability of hit.	none
SSP <sub>D</sub>	SSPD	The single shot probability of damage.	none
T <sub>H</sub>	HORIZ	The average horizontal dimension for the target.	feet
t <sub>off</sub>	SECOFF(7)	The duration of the inactive portion of the fuze timing cycle.	seconds
t <sub>on</sub>	SECON(7)	The duration of the active portion of the fuze timing cycle.	seconds
t <sub>r</sub>		The simulation time when a given event occurs.	seconds
t <sub>s</sub>	IT	The random starting point for the fuze timing cycle.	seconds
T <sub>t</sub>		A time period relating to the fuze timing cycle.	seconds
T <sub>V</sub>	VERT	The vertical dimension for the target.	feet
U <sub>RN</sub>	RN	A random number from a uniform distribution.	none
V	V	An intermediate variable used in obtaining a normal random number.	none

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
$VA_N$		The vulnerable area of the target in the normal plane.	square feet
$W_{ET}$	ETW	The effective target width.	feet
$W_P$		The width of the improved conventional munition pattern.	feet
$X_{A(n)}$	AIMPTX(50)	The X coordinate of the $n^{\text{th}}$ sortie aimpoint in the map coordinate system.	feet
$X_{D(i)}$		The X coordinate of the $i^{\text{th}}$ direct fire area entrance or exit boundary.	feet
$X_{D(k)}$	XODEF(20)	The X coordinate of the $k^{\text{th}}$ defender in the travel path coordinate system.	feet
$X_{M(i)}$		The X coordinate of the $i^{\text{th}}$ mine in the map coordinate system.	feet
$X_{M^i(i)}$		The X coordinate of the $i^{\text{th}}$ mine with respect to the nominal sortie aimpoint.	feet
$X_{MR(i)}$	OBX	The X coordinate of the $i^{\text{th}}$ mine in the travel path coordinate system.	feet
$X_{RDMP I(j)}$	XDMP(10,10)	The X coordinate of the $j^{\text{th}}$ desired mean point of impact for an indirect fire round.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
X <sub>RDPI</sub> (j)	DMPIIX(10,10)	The X coordinate of the j <sup>th</sup> desired point of impact with respect to the origin of the volley pattern.	feet
X <sub>RI</sub> (j)	ROTDMPX(10,10)	The X coordinate of the j <sup>th</sup> desired point of impact of the indirect fire round in the travel path coordinate system.	feet
X <sub>R1</sub>	XROAD(11)	The X coordinate of the travel path segment starting point in the map coordinate system.	feet
X <sub>T</sub> (j)	TGTXOL(100)	The X coordinate of the j <sup>th</sup> target in the travel path coordinate system.	feet
X <sub>TO</sub>	XFIX	The target offset distance while diverting around a damaged target.	feet
X <sub>VAP</sub> (i)		The X coordinate of the i <sup>th</sup> volley aimpoint origin in the travel path coordinate system.	feet
X <sub>O<sub>VAP</sub></sub> (i)	XOVP(10)	The X coordinate of the i <sup>th</sup> volley aimpoint origin in the map coordinate system.	feet
Y <sub>A</sub> (n)	AIMPTY(50)	The Y coordinate of the n <sup>th</sup> sortie aimpoint in the map coordinate system.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
$Y_D(i)$		The Y coordinate of the $i^{\text{th}}$ entrance or exit direct fire area boundary.	feet
$Y_D(k)$	YODEF(20)	The Y coordinate of the $k^{\text{th}}$ defender in the travel path coordinate system.	feet
$Y_{DIV}$	YDIV(100)	The Y coordinate of the damaged target which must be diverted around.	feet
$Y_{DR}(i)$		The Y coordinate of the $i^{\text{th}}$ direct fire entrance or exit boundary.	feet
$Y_M(i)$		The Y coordinate of the $i^{\text{th}}$ mine in the map coordinate system.	feet
$Y_M'(i)$		The Y coordinate of the $i^{\text{th}}$ mine with respect to the nominal sortie aim-point.	feet
$Y_{MR}(i)$	OBY	The Y coordinate of the $i^{\text{th}}$ mine in the travel path coordinate system.	feet
$Y_{PC}(i)$		The center of the pattern for the $i^{\text{th}}$ FAE detonation position.	feet
$Y_{PI}(i)$		The closest point of approach to the target of the effects of the $i^{\text{th}}$ FAE detonation.	feet
$Y_{PO}(i)$		The furthest point from the target of the $i^{\text{th}}$ FAE detonation effects.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
Y <sub>RDMP</sub> (j)	YDMP(10,10)	The Y coordinate of the j <sup>th</sup> desired mean point of impact for an indirect fire round.	feet
Y <sub>RDP</sub> (j)	DMPIIY(10,10)	The Y coordinate of the j <sup>th</sup> desired point of impact with respect to the origin of the volley pattern.	feet
Y <sub>RI</sub> (j)	ROTDMPY(10,10)	The Y coordinate of the j <sup>th</sup> desired point of impact of the indirect fire round in the travel path coordinate system.	feet
Y <sub>R1</sub>	YROAD(11)	The Y coordinate of the travel path segment starting point in the map coordinate system.	feet
Y <sub>T</sub> (j)	TGTYNW(100)	The Y coordinate of the j <sup>th</sup> target in the travel path coordinate system.	feet
Y <sub>VAP</sub> (i)		The Y coordinate of the i <sup>th</sup> volley aimpoint origin in the travel path coordinate system.	feet
Y <sub>O<sub>VAP</sub></sub> (i)	YOVP(10)	The Y coordinate of the i <sup>th</sup> volley aimpoint origin in the map coordinate system.	feet
ΔX <sub>(ij)</sub>	DISTPD	The distance between the mine and the target.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
$\Delta Y_{(ij)}$	DIRDIS(100)	Range distance that must be traveled by the $j^{\text{th}}$ target to reach the point of closest approach to the $i^{\text{th}}$ mine.	feet
$\sigma_{AD}$	SIGAD(50)	The aiming error standard deviation in deflection.	feet
$\sigma_{AR}$	SIGAR(50)	The aiming error standard deviation in range.	feet
$\sigma_{BD}$	SIGBD(50)	The ballistic error standard deviation in deflection.	feet
$\sigma_{BR}$	SIGBR(50)	The ballistic error standard deviation in range.	feet
$\theta$	THETA	The angle defining the direction of travel of the delivery aircraft (measured clockwise from the positive Y axis).	degrees
$\theta_{F(i)}$	AYYVP(10)	The angle defining the direction of attack for the indirect fire volley aimpoint, measured clockwise from the positive Y axis in the map coordinate system.	degrees
$\theta_{FR}$	DIRATK(10)	The angle defining the direction of attack for the indirect fire volley aimpoint in the travel path coordinate system.	degrees

LIST OF SYMBOLS AND ABBREVIATIONS - MATHEMATICAL MODEL  
(CONCLUDED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN SIMULATION MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
$\theta_R$	THETR	The angle between the directed travel path segment and the positive X axis in the map coordinate system.	degrees

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
ACEP(5,2)	CEP	An array containing the circular error probable in the normal plane for the defender or intruder targets by type.	feet or mils
ACEP1		The circular error probable in the normal plane for direct fire rounds.	feet or mils
ADEP(20)		An array containing the deflection error probable for the direct fire round.	feet
ADEP1,ADEP2		The deflection error probable for direct fire rounds.	feet
AEI(5,5,4)		An array containing flags indicating either a defender firing at an intruder or an intruder firing at a defender and the value of the effectiveness index.	none
AEV		The value of the effectiveness index for direct fire weapons.	various
AI(20)		An array containing the angle of fall at weapon impact for direct fire weapons.	degrees
AIMPTX(50)	$X_A(n)$	An array containing the X coordinates of the sortie aimpoints in the map coordinate system.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
AIMPTY(50)	$Y_{A(n)}$	An array containing the Y coordinates of the sortie aimpoints in the map coordinate system.	feet
AI1,AI2		The angle of fall at impact for a direct fire round.	degrees
AKIL		An internal program variable used in printing optional output.	none
ALNGLC		The length of the line charge used as a sweeping device.	feet
ALWRAT	a	The ratio of the length of the target to the width of the target.	none
AM		A variable used for temporary storage of the maximum value of the target length and target width.	feet
ANG		A variable used as temporary storage for the angle at which the indirect fire volleys are delivered.	radians
ANGIMP(10)	I	The angle of impact for the indirect fire volley delivered at each aimpoint.	degrees
APHD		The probability of damage given a hit for a weapon type employed against a target type.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
AREL(5,2)	R	The reliability of the direct fire round fired by either defender or intruder targets by type.	none
AREL1		The reliability of direct fire rounds.	none
AREP(20)	REP	An array containing the range error probable for the direct fire round.	feet
AREP1,AREP2		The range error probable for direct fire rounds.	feet
AWIDLC		The width of the line charge effects.	feet
AYYVP(10)	$\theta_{F(i)}$	The direction of attack for each indirect fire aimpoint, measured clockwise from the positive Y axis in the map coordinate system.	degrees
A1		The difference in Y coordinates of the end points of a travel path segment.	feet
A2		The difference in X coordinates of the end points of a travel path segment.	feet
BCEP		The circular error probable in the normal plane for direct or return fire munitions.	feet or mils

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
BIG		The total breach time for a target.	minutes
BIGY		A target Y coordinate used in determining the unit movement of intruder targets.	feet
BREL		The reliability of the direct fire round fired by the intruder or defender.	none
B1	$b_1$	An internal program variable.	none
B2	$b_2$	An internal program variable.	none
COSA		The cosine of the angle defining the direction of attack for an indirect fire aimpoint.	none
COSANG		The cosine of the angle defining the direction of attack for an indirect fire aimpoint.	none
COSIMP		The cosine of the angle of fall at impact for an indirect fire round.	none
COST		The cosine of the angle defining the direction of travel of the delivery aircraft.	none
COSTR		The cosine of the angle between the directed travel path segment and the positive X axis in the map coordinate system.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
DBFAE		The distance between fuel air explosive munition aimpoints.	feet
DEFHT(5)		An array containing the height of the defender targets by type.	feet
DEFL(5)		An array containing the length of the defender targets by type.	feet
DEFRAD(5)		An array containing the radius of the defender targets by type.	feet
DEFW(5)		An array containing the width of the defender targets by type.	feet
DEFX(20)		An array containing the X coordinate of the defender in the travel path coordinate system.	feet
DEFY(20)		An array containing the Y coordinate of the defender in the travel path coordinate system.	feet
DELATM(100)		An array containing the time remaining for each intruder to delay while breaching the minefield.	minutes
DELTM		The total time an intruder is delayed while traversing a travel path segment.	minutes

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
DENOM		A variable used in the calculation of single shot probability of damage from indirect fire weapons.	none
DEP(10)	DEP	An array containing the deflection error probable values for each volley aimpoint.	mils
DFTFAE	$D_F$	The intended distance in front of the intruder for deploying fuel air explosive devices.	feet
DIRATK(10)	$\theta_{FR}$	An array containing the direction of attack for each indirect fire volley aimpoint in the travel path coordinate system.	radians
DIRDIS(100)	$\Delta Y_{(ij)}$	An array containing the range distances to the next event for each target.	feet
DIS	$D_{(ik)}$	The distance between two mines (used for sympathetic detonation evaluations).	feet
DIST(52)		An array containing data used to calculate values of variables used in output of distribution data.	various
DISTPD	$D_{MT},$ $\Delta X_{(ij)}$	The distance between the mine and the target.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
DIST5(21,52)		An array containing distribution output data.	none
DLCF		The length of the line charge or the distance between fuel air explosive munition aimpoints.	feet
DMPIIX(10,10)	$X_{RDPI}(i)$	An array containing the X coordinate of the desired mean points of impact for each round delivered at each indirect fire aimpoint.	feet
DMPIIY(10,10)	$Y_{RDPI}(j)$	An array containing the Y coordinate of the desired mean points of impact for each round delivered at each indirect fire aimpoint.	feet
DO	$D_0$	The distance in deflection from a target to an indirect fire round impact point.	feet
DSQ		The sum of the squares of the distances in range and deflection from a target to an indirect fire round impact point.	feet squared
DSSP	DSSP	The single shot probability of damage in deflection.	none
DUDPRD(7)	$P_{DUD}$	An array containing the probabilities that the mine types are duds.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
DUMMY		A dummy argument used by RANF.	none
D3DEL		The time required to re-move a travel path blockage caused by a damaged target on the travel path flanked by two targets which were damaged while diverting.	minutes
EI(5,5,2)		An array containing the value and type of the effectiveness indice for each intruder target type and each indirect fire weapon type employed.	various
EIVAL		A variable used to store the value of the effectiveness index.	various
ETL	$L_{ET}$	The effective length of the target.	feet
ETW	$W_{ET}$	The effective width of the target.	feet
EV1		An internal program variable.	none
EV2		An internal program variable.	none
EV3		An internal program variable.	none
HOL(8)		An array for temporary storage of card images.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
HORIZ	$T_H$	The average horizontal dimension for the intruder or defender type.	feet
I		An internal program variable.	none
IAP		The sequential sortie aimpoint number.	none
IBIT		An internal program variable.	none
ID		The sequential number for targets which must be diverted around.	none
IDAM		A variable used to signal that a target has been damaged by direct or indirect fire.	none
IDET		The mine detonation flag as determined by Subroutine BOOM.	none
IDISOP		The output option for distribution data: 0 = distributions for output variables are not printed. 1 = distributions for output variables are calculated and printed.	none
IDP(20,15,2)		An array containing numbers specifying sets of delivery parameters used in the direct fire methodology.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
IDP1, IDP2, IDP3		A number specifying a set of delivery parameters used in the direct fire methodology.	none
IEND		The run number counter.	none
IFMT(27)		An array containing the output format for distribution data.	none
IIS		An index number indicating which target is in the next event.	none
IMINE		The sequential mine number.	none
IND		The number specifying the set of delivery parameters for a direct fire munition.	none
INDFA		A flag indicating that a target has encountered an indirect fire volley aimpoint.	none
INPUT		Local storage for the card type.	none
INT(52)		An array containing the intervals over which the distribution data are computed.	none
INTIME(7)		The fuze timing option or target counting option for each mine type: 0 = mine type is always active.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
		1 = mine type has ON/ OFF cycle or counts targets before arming.	
INTYP		The intruder type.	none
IOB(5000)		An array of mine data travel path segment boundary data and di- rect and indirect fire data in packed form (consisting of X and Y coordinates, event type, mine sequential number, and fuze timing cycle starting point or target count for arming).	none
IOUT(15)		An array containing dis- tribution output varia- bles.	none
IPO		A variable containing alpha information indi- cating that a mine was swept or detonated in the event.	none
IPOS(130)		An array used to store the position in the TMTOFR array of the events in time sequence.	none
IPRINT		The number of the itera- tion for which detailed information concerning each event is desired.	none
IPRR		An internal program vari- able used to store the priority of the intruder	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
		that the defenders may fire upon.	none
IRD		An internal program variable.	none
IRN		A variable used for temporary storage of the iteration number.	none
IRNK(7)		A flag controlling the target counting option for mines: 0 = mine will arm and detonate after sensing KOUNT(MT) number of targets. 1 = mine will arm and detonate after sensing a uniform random number of targets varying from one up to KOUNT(MT).	none
IROAD		The travel path segment that the indirect fire volleys will occur on.	none
IRTFIR		A flag indicating whether the intruders may return fire.	none
IRUNS		A replication or iteration counter which varies between 1 and NOIT.	none
ISAVE		An index number indicating which intruder target is in the next event.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
ISBL		An option describing the disposition of swept mines: 0 = swept mines are neutralized. 1 = swept mines are blown-in-place.	none
ISHOT		An internal program variable.	none
ISUB		An internal program variable.	none
ISV		An internal program variable.	none
ISVIFA(30)		An array used to determine which intruders are accessible to fire from the defender forces.	none
ISVLST		A save location for the last value of ISAVE.	none
ISYMP(3)		An array containing the sympathetic detonation options [ISYMP(1) is for mines, ISYMP(2) is for direct fire munitions, and ISYMP(3) is for indirect fire munitions]: 0 = sympathetic detonations are not evaluated. 1 = sympathetic detonations are evaluated.	none
ISYP		A flag to initiate unit movement considerations.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
IT	$t_s$	The mine random starting time, the target count required for arming, or the direct fire area.	various
ITEMP		An internal program variable.	none
ITEROP		An option for printing the iteration output: 0 = do not print iteration results. 1 = print iteration results.	none
ITEST		An internal program variable.	none
ITGT		Temporary storage for the intruder target number.	none
ITGTPR(5,5)		An array containing the list by type of the order in which the defender forces will fire at the intruders.	none
ITT		An internal program variable.	none
ITYP		Temporary storage for the intruder target type.	none
IVAP		The number of the indirect fire volley aim-point.	none
IVEL		A program switch that contains a one if all targets are in a delay condition; otherwise, it contains a zero.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
IVPO		An internal program variable.	none
IWD		An internal program variable.	none
IWEPV		The sequential number of the indirect fire round.	none
IWORD		The data for an event in packed format.	none
IWT		Temporary storage for the weapon type.	none
IWTDEF(20)		An array containing the weapon type of each defender.	none
IWTVAP(10)		An array containing the indirect fire weapon type employed at each aimpoint.	none
IX		An internal program variable.	none
IY		An internal program variable.	none
IXYM1		An internal program variable.	none
IXYM2		An internal program variable.	none
I1		An internal program variable.	none
I2		An internal program variable.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
I3		An internal program variable.	none
I4		An internal program variable.	none
I5		An internal program variable.	none
I8		An internal program variable.	none
I14		An internal program variable.	none
J		An internal program variable.	none
JAA		An internal program variable.	none
JFMT(27)		An array containing the program-derived output format for distribution data.	none
JMARK1		A control for printing the output heading.	none
JSELDS(7)		An option for specifying the type of mine pattern distribution for each mine type: 0 = mine pattern is read from cards. 1 = range and deflection values selected from random normal distribution.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
		2 = range and deflection values selected from random uniform distribution.	
		3 = range value selected from random normal distribution and deflection value selected from random uniform distribution.	
		4 = range value selected from random uniform distribution and deflection value selected from random normal distribution.	
JT		An internal program variable.	none
J1		An internal program variable.	none
J2		An internal program variable.	none
J3		An internal program variable.	none
J4		An internal program variable.	none
J5		An internal program variable.	none
K		An internal program variable.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
KABOOM(547)		An array containing 32,820 bits for maintaining mine detonation flags.	none
KB		The sequential number for the first intruder target of the same type as the damaged target and behind the damaged target.	none
KDFTTN(100)		An array containing the count of the number of direct fire areas in which each intruder target may receive direct fire.	none
KF		The sequential number for the first intruder target of the same type as the damaged target and in front of the damaged target.	none
KFMT(27)		An array containing the program-derived output format for distribution data.	none
KNOB		The number of the last event to be considered in sympathetic detonation evaluation.	none
KNOBT		A variable used as temporary storage of the type of event to occur.	none
KNRS		The sequential number of the travel path segment.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
KODEI		Temporary storage for the type of effectiveness index used.	none
KOUNT(7)		An array containing the target counts at which the mine types will arm and detonate.	none
KPOS		A variable used for temporary storage of the intruder number.	none
KREAD		A switch indicating whether Subroutine READIN has been used: 0 = READIN has not been used. 1 = READIN has been used.	none
KT		An index number which locates an event in the IOB array.	none
KTMAX		The maximum mine sequential number (used in the sympathetic detonation evaluations).	none
KTMIN		The minimum mine sequential number (used in the sympathetic detonation evaluations).	none
KTS		The location in which the sequential mine number is saved.	none
KTSAV(100)		An array of indice containing the subscript of	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
		the next event to be encountered by each of the targets.	
KTS1		An internal program variable.	none
KZ		An internal program variable.	none
K1		An internal program variable.	none
K2,K3		A flag indicating the direction of direct fire: 1 = a defender firing at an intruder. 2 = an intruder firing at a defender.	none
K5		An internal program variable.	none
K6		An internal program variable.	none
L		An internal program variable.	none
LCFOPT		The option for employing line charge or FAE devices for sweeping: 0 = neither are used. 1 = line charges are employed. 2 = FAE are employed.	none
LCFTT		The target type which deploys the line charge or fuel air explosive sweeping device.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
LDFDPR(5,5)		An array containing the order in which each intruder target type will fire at the defender target types.	none
LEFTIN		The number of targets which have not been destroyed or have not breached the engagement area.	none
LFMT(27)		An array containing the output format for distribution data.	none
LIT		An internal program variable.	none
LL		An internal program variable.	none
M		An internal program variable.	none
MADIV		An indicator which, if greater than zero, specifies that targets must be diverted around.	none
MFMT(27)		An array containing the output format for distribution data.	none
MI		An internal program variable.	none
MIFBT(7)		An array specifying the number of mines within the range of influence for each mine type.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
MINNUM		The number of sympathetic detonations.	none
MODE		The tactical mode to be employed: 1 = bull-through at normal speed with no sweeping. 3 = allow specified targets to sweep minefield.	none
MT		The mine type number.	none
MTFA(50)		An array containing the type of mine dispensed at each aimpoint.	none
MUSH		A code specifying whether the targets have started moving: 0 = the targets have not started moving. 1 = the targets have started moving.	none
N		An internal program variable.	none
NAP		The number of sortie aimpoints.	none
NAS(100)		An array containing sequential numbers of the targets associated as a group with the damaged target.	none
NB		An internal program variable.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NCTAW(100)		An array containing the number of the target leading the column with which each target is associated.	none
ND		A count of the number of direct fire areas.	none
NDEFEA(15,20)		An array containing the sequential defender numbers that may fire into the direct fire areas.	none
NDEFNR		Temporary storage of the number of the defender.	none
NDF		The defender associated with this event.	none
NDFEA		A flag indicating that an intruder target has entered a direct fire area.	none
NDFAA		The number of direct fire attack areas.	none
NDFDAM(5)		An array containing the number of defenders damaged by type.	none
NDFWD		The number of defenders.	none
NDFWT		The number of defender weapon types.	none
NDTYP		The defender weapon type.	none
NDVT		The number of targets which must be diverted around.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NE		An internal program variable.	none
NFMT(27)		An array containing the program-derived output format for distribution data.	none
NGTAW(100)		An array containing the sequential numbers of other targets with which each target is associated as a group.	none
NGTAWT		The total number of targets which are associated as a group with other targets.	none
NGTAW1(100)		An array originally containing the NGTAW array variables (changed when a target is reassociated with another target).	none
NIDBT(5)		An array containing the number of intruders damaged by type.	none
NINTT		Temporary storage of the intruder target type.	none
NITBT(5)		An array containing the number of intruders by type.	none
NKILL(5)		An array containing the number of intruder targets by type damaged by mines.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
NKILLD(5)		An array containing the number of intruders by type damaged by direct fire weapons.	none
NKILLI(5)		An array containing the number of intruders by type damaged by indirect fire weapons.	none
NKILLT		The total number of intruder targets damaged.	none
NLCFCT		A count of the number of line charge or FAE deployment positions along the travel path segment.	none
NLST		The number of sympathetic detonations.	none
NM		The mine type.	none
NMDET(7)		An array containing the number of mines detonated by mine type.	none
NMIN(7)		An array containing the total number of mines of each type emplaced.	none
NMSWPT(7)		An array containing the number of mines of each type detected and neutralized by a target capable of sweeping.	none
NMT		The total number of mine types.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NOB		The total number of mines within the range of influence of a travel path segment.	none
NOBEVT		The number of events for which optional output is printed.	none
NOBT(100)		An array containing the mine types for mines which are sympathetically detonated.	none
NOBTP2		The type of event obstacle (1 through 7 = active mine; 8 = minefield boundary; 9 through 15 = dud mine; 16 = indirect fire volley aimpoint; 17 = direct fire entry boundary; 18 = direct fire exit boundary; and 19 = point of deployment of line charge or FAE).	none
NOIT		The number of iterations desired for the case.	none
NOSTAT		The number of iterations between each statistical summary.	none
NRAD(5)		An array containing the number of rounds available for each defender weapon type.	none
NRADFD(5)		An array used for temporary storage of the number of rounds available for each defender weapon type.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NRADFI (5)		An array used for temporary storage of the number of rounds available for each intruder type.	none
NRAI (5)		An array containing the number of rounds available for each intruder type.	none
NRBA		The number of rounds the defender forces must fire before being acquired by the intruder forces.	none
NRFBDT (5)		An array containing the number of rounds fired by each defender type.	none
NRFBIT (5)		An array containing the number of rounds fired by each intruder type.	none
NRFIRD (100)		An array containing the number of rounds fired by each intruder.	none
NRS		The number of travel path segments.	none
NSLEFT		The number of targets remaining which are capable of sweeping.	none
NSTICK (50)		An array containing the number of mines dispensed at each aimpoint.	none
NSUB (5)	$N_s$	An array containing the number of submunitions for improved conventional munitions.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NT		The target type number.	none
NTABLE		The number of the probability function to be interpolated.	none
NTB		A number indicating the type of probability table.	none
NTCOL(100)		An array containing column identification for targets which must be diverted around.	none
NTGO		The total number of targets traversing the engagement area.	none
NTGTP		The total number of intruder target types.	none
NTGTYP(100)		An array containing the target type for each intruder target.	none
NTLCIF(10)		An array containing the number of the target leading the column of intruders that initiates the indirect fire at each volley aimpoint.	none
NTLOST		The number of intruder targets which are lost to the mission.	none
NTMAX		The maximum number of intruder types, mine types, and defender types.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NTN		The intruder number.	none
NTTCS (5)		An array containing flags describing the sweeping capability of each target type: 0 = the target type has no sweeping capability. 1 = the target type is capable of sweeping at least one mine type.	none
NTTTMD (5)		An array containing flags describing which target types must be diverted around by subsequent targets in the same column: 0 = the target type need not be diverted around. 1 = the target type must be diverted around.	none
NTTTRP (5)		An array containing flags describing which target types are rollers or plows: 0 = the target type is not a roller or plow. 1 = the target type is a roller. 2 = the target type is a plow.	none
NT1		An internal program variable.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
NUM		An internal program variable.	none
NUMCNT		The number of time oriented events which may occur before the distance oriented event.	none
NUMDEF		The defender number.	none
NVAP		The number of the indirect fire volley aimpoint involved in this event.	none
NVFAEA(10)		An array containing the number of volleys fired at each volley aimpoint.	none
NVLIDF(10)		An array containing the number of indirect fire volleys remaining to be fired at each volley aimpoint.	none
NVOL		The number of the indirect fire volley.	none
NW		An internal program variable.	none
NWEPV(10)		An array containing the number of rounds to be employed at each indirect fire volley aimpoint.	none
N1		An internal program variable.	none
N2		An internal program variable.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
N3		An internal program variable.	none
OBX	$X_{MR(i)}$	The X coordinate of the distance oriented event.	feet
OBY	$Y_{MR(i)}$	The Y coordinate of the distance oriented event.	feet
OBYPRT		A location for saving OBY values so that they can be printed.	feet
ORX		An intermediate X coordinate of the mine (used during the initial calculations of the mine position).	feet
ORY		An intermediate Y coordinate of the mine (used during the initial calculations of the mine position).	feet
OUT(16)		An array containing the iteration results to be printed.	none
OUT(21,15)		An array containing the values of the distribution variables to be output.	none
PATRAD(5)	$R_p$	An array containing the improved conventional munition pattern radius for each weapon type.	none
PDAM		The probability that a target has been damaged by indirect fire.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
PDLCF	$P_{DF}$	The probability of mine detonation within the line charge or fuel air explosive pattern.	none
PHD(5,5)		An array containing the probability of damage given a hit for each indirect fire weapon type employed against each intruder target type.	none
PI		A variable used to store the value of $\pi$ .	none
PK		The square of the range at which the probability of target damage is zero.	feet squared
PKK		The probability that a target has been damaged by indirect fire.	none
POWER		A variable used as the exponent in the calculation of the probability of damage given a hit by indirect fire weapons.	none
PPEAF(7)		An array containing the probability that a plowed mine type will function.	none
PRBITY	P	A probability value determined by Subroutine TABINT.	none
PRDTNO(5,7)		An array containing the minimum ranges at which the probability that a mine will detonate is	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
		zero for each mine type and target type.	
PRKLO(5,7)		An array containing the minimum ranges at which the probability that a mine will damage a target is zero for each mine type and target type.	feet
PROBIL(840)	$P_q, P_{q+1}$	An array of probability values corresponding to the range values stored in RANGPR(840), with the first eight entries specifying the probabilities of mine detection for the first target type and first mine type, the second eight entries specifying the probabilities that a mine of the first type will damage a target of the first type, and the third eight entries specifying the probabilities that a mine of the first type will detonate when encountered by a target of the first type. The next three sets of eight entries contain similar data for Target Type 1 and Mine Type 2, etc.	none
PROBTTP(8)		An array used to temporarily store probability values during input operations.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
PRSWD0(5,7)		An array containing the minimum ranges at which the probability that the target will detect a mine is zero for each mine type and target type.	feet
P2		An internal program variable.	none
P3		An internal program variable.	none
P4		An internal program variable.	none
P5		An internal program variable.	none
RADFAE	$R_F$	The radius of the fuel air explosive effects.	feet
RANGE		The width of the line charge or the square of the radius of the fuel air explosive effects.	feet
RANGPR(840)	$R_q, R_{q+1}$	An array of range values corresponding to the probability values stored in PROBIL(840), with the first eight entries specifying the ranges corresponding to the probabilities of mine detection for the first target type and first mine type, the second eight entries specifying the ranges corresponding to the probabilities that a mine of the first type	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
		will damage a target of the first type, and the third eight entries specifying the ranges corresponding to the probabilities that a mine of the first type will detonate when encountered by a target of the first type. The next three sets of eight entries contain similar data for Target Type 1 and Mine Type 2, etc.	
RANGTP (8)		An array used to temporarily store range values during input operations.	feet
RELRND (5)		An array containing the reliability of the round for each indirect fire weapon type.	none
RELSUB (5)	$R_S$	An array containing the reliability of the submunition for each improved conventional munition indirect fire weapon type.	none
REP (10)		The range error probable at each indirect fire volley aimpoint.	feet
RN	$U_{RN}$	A random number from a uniform distribution.	none
RNG		The distance between the defender and intruder.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
RNGSV(3)		An array for temporary storage of elements from the PRSWD0, PRKLO, and PRDTN0 arrays.	none
RN1, RN2 RN3, RN4		A normal random number used in computing the mine locations.	none
RO	$R_0$	The distance in range from a target to an indirect fire round impact point.	feet
ROTDMPX(10,10)	$X_{RI}(j)$	An array containing the X coordinates of the desired mean points of impact for the rounds delivered at each volley aimpoint.	feet
ROTDMPY(10,10)	$Y_{RI}(j)$	An array containing the Y coordinates of the desired mean points of impact for the rounds delivered at each volley aimpoint.	feet
RSP(5)		An array containing the ranges of influence for each target type.	feet
RSPMAX		The maximum range of influence.	feet
RSSP	RSSP	The single shot probability of damage in range.	none
RSTART	$R_N$	A random number from a normal distribution.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
RUNS		The iteration count.	none
S(100)		An array containing variables controlling the diversion direction: -1 = divert to the left. 1 = divert to the right. 0 = no diversion is required. 999 = the target path blockage has been removed; no diversion is required.	none
SAVE		An internal program variable.	none
SECON(7)	$t_{on}$	The duration of the active portion of the fuze timing cycle for each mine type.	seconds
SECOFF(7)	$t_{off}$	The duration of the inactive portion of the fuze timing cycle for each mine type.	seconds
SEED		A number used as a starting value for the random number generator.	none
SHADOL	$L_{SH}$	The target shadow length.	feet
SIGAD(50)	$\sigma_{AD}$	The aiming error standard deviation in deflection (this value multiplied by a normal random number vector is applied to each mine in a sortie).	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
SIGAR(50)	$\sigma_{AR}$	The aiming error standard deviation in range (this value multiplied by a normal random number vector is applied to each mine in a sortie).	feet
SIGBD(50)	$\sigma_{BD}$	The ballistic error standard deviation in deflection (this value multiplied by a normal random number vector is applied independently to each mine).	feet
SIGBR(50)	$\sigma_{BR}$	The ballistic error standard deviation in range (this value multiplied by a normal random number vector is applied independently to each mine).	feet
SINA		The sine of the angle defining the direction of attack for an indirect fire aimpoint.	none
SINANG		The sine of the angle defining the direction of attack for an indirect fire aimpoint.	none
SINIMP		The sine of the angle of impact of the round employed at the indirect fire aimpoint.	none
SINT		The sine of the angle defining the direction of travel of the delivery aircraft.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
SINTR		The sine of the angle between the directed travel path segment and the positive X axis in the map coordinate system.	none
SMALY		An intruder target Y coordinate used in determining the unit movement of targets.	feet
SMLDIS		The smallest distance to the next event for all targets.	feet
SMLDIS1		The smallest distance to the next event for all moving targets.	feet
SOBX		Temporary storage for the mine X coordinate.	feet
SOBY		Temporary storage for the mine Y coordinate.	feet
SRPI		The square root of $\pi$ .	none
SSPD	$SSP_D$	The single shot probability of damage.	none
STATAL(5,52)		An array of five quantities (i.e., the sum, the sum of squares, the mean, the variance, and the standard deviation) for each of the 52 parameters for which statistical output is provided.	various
SURVPR		The probability of target survival.	none

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
SWPDEL(100)		An array containing the times required to remove or neutralize mines for each target.	minutes
SYMDDF(5,7)		An array containing the maximum distances at which each direct fire weapon type can cause a mine of each type to detonate sympathetically.	feet
SYMDIF(5,7)		An array containing the maximum distances at which each indirect fire weapon type can cause a mine of each type to detonate sympathetically.	feet
SYMDIS(7,7)		An array containing the maximum distances at which each mine type can cause a mine of each type to sympathetically detonate.	feet
SYMMAX(3)		An array containing the maximum distances at which mines may be sympathetically detonated by other mines, direct fire weapons and indirect fire weapons.	feet
TADJ		An adjustment made to TIMEMV(100) to correct the target travel time at the beginning of each travel path segment.	minutes

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
TARHT (5)	$H_T$	An array containing the height of the intruder targets by type.	feet
TARL (5)	$L_T$	An array containing the length of the intruder targets by type.	feet
TARRAD (5)	$R_T$	An array containing the radius of the intruder targets by type.	feet
TARW (5)		An array containing the width of the intruder targets by type.	feet
TDBIFV (10)		An array containing the time delay between the volleys fired at each aimpoint.	minutes
TGTD		The time the target was delayed due to the removal of a travel path blockage.	minutes
TGTDEL (100)		An array containing the times each target was delayed due to the removal of a travel path blockage.	minutes
TGTOVL		The normal speed of the targets.	feet per minute
TGTSPP		The speed of the intruder targets.	feet per minute

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
TGTVEL(100)		An array containing the speed of each target.	feet per minute
TGTVL2		The sweep rate.	feet per minute
TGTVSV		Temporary storage for the target velocity.	feet per minute
TGTXOL(100)	$X_T(j)$	An array containing the original X coordinate for each target in the travel path coordinate system.	feet
TGTYNW(100)	$Y_T(j)$	An array containing the current Y coordinate of each target in the travel path coordinate system.	feet
TGTYOL(100)		An array containing the original Y coordinate of each target in the travel path coordinate system.	feet
TGTYSV		The Y coordinate of the event target at the point of closest approach to the mine.	feet
THET		THETA in radians.	radians
THETA	$\theta$	The angle defining the direction of travel of the delivery aircraft (measured clockwise from the positive Y axis).	degrees

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
THETR	$\theta_R$	The angle between the directed travel path segment and the positive X axis in the map coordinate system.	degrees
THRSIG(5,5)		An array containing distances beyond which the effects of indirect fire detonations are not considered for each indirect fire weapon type and each target type.	feet
TIMEMV(100)		An array containing the travel time for each target.	minutes
TM(130)		A temporary storage array for event times.	minutes
TMBRD(5)		An array containing the reload and aim times for each defender target type.	minutes
TMBRI(5)		An array containing the reload and aim times for each intruder target type.	minutes
TMDLCF		The amount of time the intruders will be delayed while deploying the line charge or fuel air explosive.	minutes
TMMV		The target travel time.	minutes
TMSWP(7)		An array containing the times required to remove or neutralize each mine type.	minutes

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
TMTOFR(130)		An array containing the times for events to occur for direct fire, indirect fire, and return fire.	minutes
TOTDEL		The sum of the time required to remove or neutralize a mine, the time the target was delayed due to the removal of a travel path blockage, and the time required to deploy line charges or fuel air explosives.	minutes
TOTSIM		The total breach time.	minutes
TOT2		An internal program variable.	none
TRAVTM		The travel time to the next event.	minutes
TT		A variable used as temporary storage of the travel time to the next event.	minutes
TTDI		The time between time-oriented events.	minutes
TVELPR		Temporary storage for the target velocity.	feet per minute
TYPRIT		Temporary storage for the target Y coordinate.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
T1		An intermediate variable used in determining the target offset due to diversion.	feet
T2		An intermediate variable used in determining the target offset due to diversion.	feet
T3		An intermediate variable used in determining the target offset due to diversion.	feet
V	V	An intermediate variable used in obtaining a normal random number.	none
VERT	$T_v$	The vertical dimension for the intruder or defender type.	feet
XCYCLE(7)		An array containing the total fuze cycle time (if any) for each target.	seconds
XD		The square of the distance in the X direction between the defender and intruder.	feet squared
XDIS		An intermediate target/mine distance used in determining the next event.	feet
XDMP(10,10)	$X_{RDMP I(j)}$	An array containing the X coordinate in the map coordinate system of the desired mean points of impact of each round	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
		delivered at each indi- rect fire volley aimpoint.	
XFIX	$X_{TO}$	The target offset dis- tance while diverting around a damaged target.	feet
XFIX1		Temporary storage for XFIX.	feet
XLCF(100)		An array containing the X coordinate of the in- truder employing fuel air explosives or line charges as sweeping de- vices.	feet
XLOC		Temporary storage for the X coordinate of the position of the line charge or fuel air ex- plosive.	feet
XMAX		The maximum X coordinate for mines retained for the case.	feet
XMIN		The minimum X coordinate for mines retained for the case.	feet
XODEF(20)	$X_D(k)$	An array containing the X coordinate of each defender in the map co- ordinate system.	feet
XORGVP		An internal program vari- able.	feet
XOVP(10)	$X_{OVAP}(i)$	An array containing the X coordinate of the origin of each volley aimpoint.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
XR		An internal program variable.	none
XRDFO(30)		An array containing the X coordinates of the ends of the direct fire attack areas.	feet
XRG(52)		An array containing the maximum value of each distribution output variable.	none
XRN		The reciprocal of the iteration number.	none
XROAD(11)	$X_{R1}$	An array containing the X coordinates of the ends of the travel path segments in the map coordinate system.	feet
XR1		The X coordinate of the starting point of a travel path segment.	feet
XR2		The X coordinate of the end point of a travel path segment.	feet
XSWATH(50)		An array containing mine pattern widths for uniform distributions of mines for each aimpoint.	feet
XT		The X coordinate of the intruder.	feet
XW		The X coordinate of the volley aimpoint.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
XWIDTH		The width of the area through which the targets are passing.	feet
XW2		One-half of the width of the area through which the targets are passing.	feet
XYMS(4)		An array containing the X and Y coordinates of an area of a travel path segment accessible to direct fire.	feet
YD		The square of the distance in the Y direction between the defender and intruder.	feet squared
YDIST		The distance from the Y coordinate of the position of the line charge or fuel air explosive to the outer limit of the explosive effects.	feet
YDIV(100)	Y <sub>DIV</sub>	An array containing the Y coordinates of damaged targets which must be diverted around.	feet
YDMP(10,10)	Y <sub>RDMP</sub> (j)	An array containing the Y coordinate in the map coordinate system of the desired mean points of impact of each round delivered at each indirect fire volley aimpoint.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONTINUED)

SYMBOL OR ABBREVIATION	EQUIVALENT IN MATH MODEL	DEFINITION	UNITS
YFIX		The distance that targets are moved to reassociate with a new lead target.	feet
YLENGTH		The length of the area through which the targets are passing.	feet
YLENTH		The length of the travel path segment.	feet
YL2		One-half of the length of the area through which the targets are passing.	feet
YMAX		The maximum Y coordinate of the mines which sympathetically detonate.	feet
YMIN		The minimum Y coordinate of the mines which sympathetically detonate.	feet
YODEF(20)	$Y_D(k)$	An array containing the Y coordinate of each defender in the map coordinate system.	feet
YORGVP		An internal program variable.	feet
YOVP(10)	$Y_{OVAP}(i)$	An array containing the Y coordinates in the map coordinate system of the origin of each volley aimpoint.	feet
YRDFO(30)		An array containing the Y coordinate in the map coordinate system of the ends of the direct fire attack areas.	feet

LIST OF SYMBOLS AND ABBREVIATIONS - SIMULATION MODEL  
(CONCLUDED)

<u>SYMBOL OR ABBREVIATION</u>	<u>EQUIVALENT IN MATH MODEL</u>	<u>DEFINITION</u>	<u>UNITS</u>
YROAD(11)	$Y_{R1}$	An array containing the Y coordinates of the ends of the travel path segments in the map coordinate system.	feet
YR1		The Y coordinate of the starting point of a travel path segment.	feet
YR2		The Y coordinate of the end point of a travel path segment.	feet
YST		The Y coordinate of the inner limit of the fuel air explosive effects.	feet
YSWATH(50)		An array containing the mine pattern lengths for uniform distributions of mines for each aimpoint.	feet
YT		The Y coordinate of the intruder.	feet
YW		The Y coordinate of the volley aimpoint.	feet

## SECTION I

### INTRODUCTION

This computer simulation model, referred to as SEMAC, provides the methodology and analytical techniques required for evaluating the synergistic effects of minefields and covering fire. The program was designed to allow mixed minefields, various armored vehicle tactics, and employment of combinations of different types of direct and indirect fire. SEMAC determines, on a time history basis, the damage and the breach times for the engagement area consisting of mines and covering fire. The intruder force travels along multisegmented travel paths through the engagement area, with one of two tactics employed by the formation of targets. These tactic options are:

- The targets traverse the engagement area with no sweeping and no evasive action taken.
- The engagement area is swept by selected targets traversing the engagement area.

SEMAC is an event-oriented simulation model which uses Monte Carlo techniques to obtain measures of effectiveness. The minefield is defined by randomly applying aiming errors to each sortie and ballistic errors to each mine in the sortie. The aiming range and deflection errors are independent of each sortie of mines delivered, and the ballistic range and deflection errors are independent of each mine in the pattern. The formation of targets proceeds through the minefield, and damage is assessed for every encounter of a mine by a target. A mine which detonates is evaluated against all targets. Inputs specify the locations of covering fire weapons, their time of commencing fire, and other items pertaining to weapon employment, target acquisition, system accuracies, and return fire parameters. The effect of the covering fire on the targets attempting the minefield breach is evaluated in time sequence. The simulation ends when all targets have either passed through the minefield and/or all targets have been damaged. The minefield is redefined using random aiming and ballistic errors, and another iteration is evaluated. At the end of a selected number of iterations, statistical output is computed and presented.

The targets which are engaged in the minefield breach attempt are capable of utilizing several tactics for mine-sweeping. Visual detection of mines by selected target

elements is provided. In addition, methodology for the employment of armored plows and rollers, explosive line charges, and fuel air explosives (FAE) is incorporated. Explosive line charges and FAE mine clearing provides minefield degradation within a rectangular area around line charges and within a circular area around FAE.

The "in minefield" targets receive direct and indirect covering fire from defending forces according to an input time schedule, and are capable of returning fire. The indirect fire methodology is based upon the development of Tri-Service approved methodology for the Joint Technical Coordinating Group for Munitions Effectiveness (JTTCG/ME) Methodology Working Group. These techniques are used to compute the effectiveness of single or multiple releases of unguided weapons (including cluster munitions) against various types of targets. Up to ten volley aimpoints are permitted with up to ten rounds fired per volley. The effectiveness index can be expressed in the following terms:

- Mean area of effectiveness for fragmentation ( $MAE_f$ )
- Vulnerable area ( $VA_N$ )
- Mean area of effectiveness for blast ( $MAE_b$ )
- Effective miss distance (EMD).

The direct covering fire methodology considers the effects of terrain masking on target acquisition and line of fire. The extent of terrain masking is described by inputs pertaining to location and size of natural obstacles. A direct fire area is a portion of the travel path not masked by terrain.

SEMAC will simulate the passage of up to 100 intruder targets of up to five different types passing through an engagement area. The program can consider up to 50 aimpoints for mines and up to seven different mine types. A maximum of 32,767 mines can be dispensed, and up to 4,998 mines can be located within the range of influence on either side of a travel path segment. The width of the range of influence is the maximum distance at which a mine can detonate or be detected by a target. (A mine which is outside this region cannot affect the outcome of the simulation.) The computer program can evaluate the effectiveness of up to 20 defenders of up to five direct fire defender weapon types. Each of the five defender weapon types is capable of direct covering fire against the five target types of the intruder and each of the

five target types of the intruder is capable of return fire against the target types of the defender.

This volume contains:

- A detailed description of the mathematical model which was used as a basis in developing SEMAC.
- A complete set of flowcharts depicting the logic used in the program.
- A complete FORTRAN IV source listing of the program.
- A detailed description of the simulation coding which is employed in the program.
- Listings of the symbols and abbreviations which are used in the mathematical model and the computer program.

Detailed descriptions of the input variables required to properly execute SEMAC, instructions for placing the input variables on punch cards and for arranging the punch cards in proper order, descriptions and definitions of the output available from SEMAC, and a description of a sample case are contained in Volume I, the User's Manual, of this report.

## SECTION II

### MATHEMATICAL MODEL

The mathematical model for SEMAC is separated into three principal parts, which are:

- Computation of effectiveness of mines.
- Computation of effectiveness of direct fire munitions.
- Computation of effectiveness of indirect fire munitions.

These principal parts of the mathematical model are discussed in detail in the following paragraphs.

#### COMPUTATION OF EFFECTIVENESS OF MINES

The mathematical model for SEMAC provides for up to 100 intruders of up to five different types passing through the engagement area. Up to 50 aimpoints and up to seven different mine types can be considered. A maximum of 32,767 mines can be dispensed, and up to 4,998 mines can be located within the range of influence on either side of a travel path segment. The width of the range of influence is the maximum distance at which a mine can detonate or be detected by an intruder. (A mine which is outside this region cannot affect the outcome of the simulation.)

#### Computation of Mine Locations in Map Coordinate System

At the start of each iteration, the mine impact points are computed. First, the mine locations for each sortie and sortie aimpoint, including aiming and ballistic errors, are computed using the expressions:

$$X_M^i(i) = D_{MPI}(i) + R_{N1}(\sigma_{AD}) + R_{N3}(\sigma_{BD}) \quad (1)$$

$$Y_M^i(i) = R_{MPI}(i) + R_{N2}(\sigma_{AR}) + R_{N4}(\sigma_{BR}) \quad (2)$$

where  $X'_M(i)$  and  $Y'_M(i)$  are the X and Y coordinates of the  $i$ th mine with respect to the nominal sortie aimpoint,  $D_{MPI}(i)$  and  $R_{MPI}(i)$  are the deflection and range components of the mean point of impact computed in SEMAC by one of the five methods shown in Table 1,  $R_{N1}$ ,  $R_{N2}$ ,  $R_{N3}$ , and  $R_{N4}$  are normally distributed random numbers,  $\sigma_{AR}$  and  $\sigma_{AD}$  are the standard deviations in range and deflection describing the distribution of the aiming error, and  $\sigma_{BR}$  and  $\sigma_{BD}$  are the standard deviations in range and deflection describing the distribution of the ballistic error. In determining the values of  $D_{MPI}(i)$  and  $R_{MPI}(i)$ , uniformly distributed mines are bounded by the input values of  $X_S$  (the maximum width of the pattern) and  $Y_S$  (the maximum length of the pattern). Normal distributions of mines are also determined using  $X_S$  and  $Y_S$ ; however, for this option, the input values of  $X_S$  and  $Y_S$  are six standard deviations, and  $X_S$  and  $Y_S$  are divided by the value 6 to obtain the standard deviations in deflection and range describing the pattern.

To determine  $R_{N1}$ ,  $R_{N2}$ ,  $R_{N3}$ , and  $R_{N4}$ , a uniform random number ( $U_{RN}$ ) is generated using RANF (a Control Data Corporation library function), and the following polynomial equations are evaluated to convert  $U_{RN}$  to a corresponding random number from a normal distribution.

$$V = \sqrt{(-2) \ln\{0.5[1.0 - (1.0 - 2.0(U_{RN}))^2]\}} \quad (3)$$

$$R_N = V - \frac{(2.515517 + 0.802853V + 0.010328V^2)}{(1 + 1.432788V + 0.189269V^2 + 0.0013208V^3)} \quad (4)$$

where  $R_N$  is the normal random number, and  $U_{RN}$  is a random number from a uniform distribution. The value of  $R_N$  is positive if  $0.5 < U_{RN} \leq 1.0$  and negative if  $0 \leq U_{RN} < 0.5$ .

Next, the mine coordinates [i.e., the  $X'_M(i)$  and  $Y'_M(i)$  values] are transformed into the map coordinate system using the expressions (see Figure 1):

$$X_M(i) = X_{A(n)} + X'_M(i) \cos \theta + Y'_M(i) \sin \theta \quad (5)$$

$$Y_M(i) = Y_{A(n)} - X'_M(i) \sin \theta + Y'_M(i) \cos \theta \quad (6)$$

TABLE 1. DETERMINATION OF RANGE AND DEFLECTION  
COORDINATES OF MEAN POINT OF IMPACT

OPTION	$R_{MPI}(i)$	$D_{MPI}(i)$
0	Read in from input data	Read in from input data
1	Select from random normal distribution	Select from random normal distribution
2	Select from random uniform distribution	Select from random uniform distribution
3	Select from random normal distribution	Select from random uniform distribution
4	Select from random uniform distribution	Select from random normal distribution

where  $X_M(i)$  and  $Y_M(i)$  are the X and Y coordinates of the  $i$ th mine in the map coordinate system,  $X'_M(i)$  and  $Y'_M(i)$  are as described above,  $X_A(n)$  and  $Y_A(n)$  are the X and Y coordinates of the  $n$ th sortie aimpoint with which the  $i$ th mine is associated, and  $\theta$  is the azimuth angle defining the direction of travel of the delivery aircraft (measured clockwise from the positive Y axis).

To evaluate the effects of weapon reliability, a uniform random number ( $URN$ ) is selected for each mine, and this number is compared to the input value of the probability that the mine is a dud ( $P_{DUD}$ ). If the inequality:

$$URN < P_{DUD} \quad (7)$$

is satisfied and if Tactic 1 is being employed, the mine is considered a dud and is eliminated from further consideration.

#### Transformation of Mine Locations Into Travel Path Coordinate System and Initial Placement of Targets

In order to evaluate large numbers of mines, the target formation traverses each input path segment one at a time with the following restrictions.

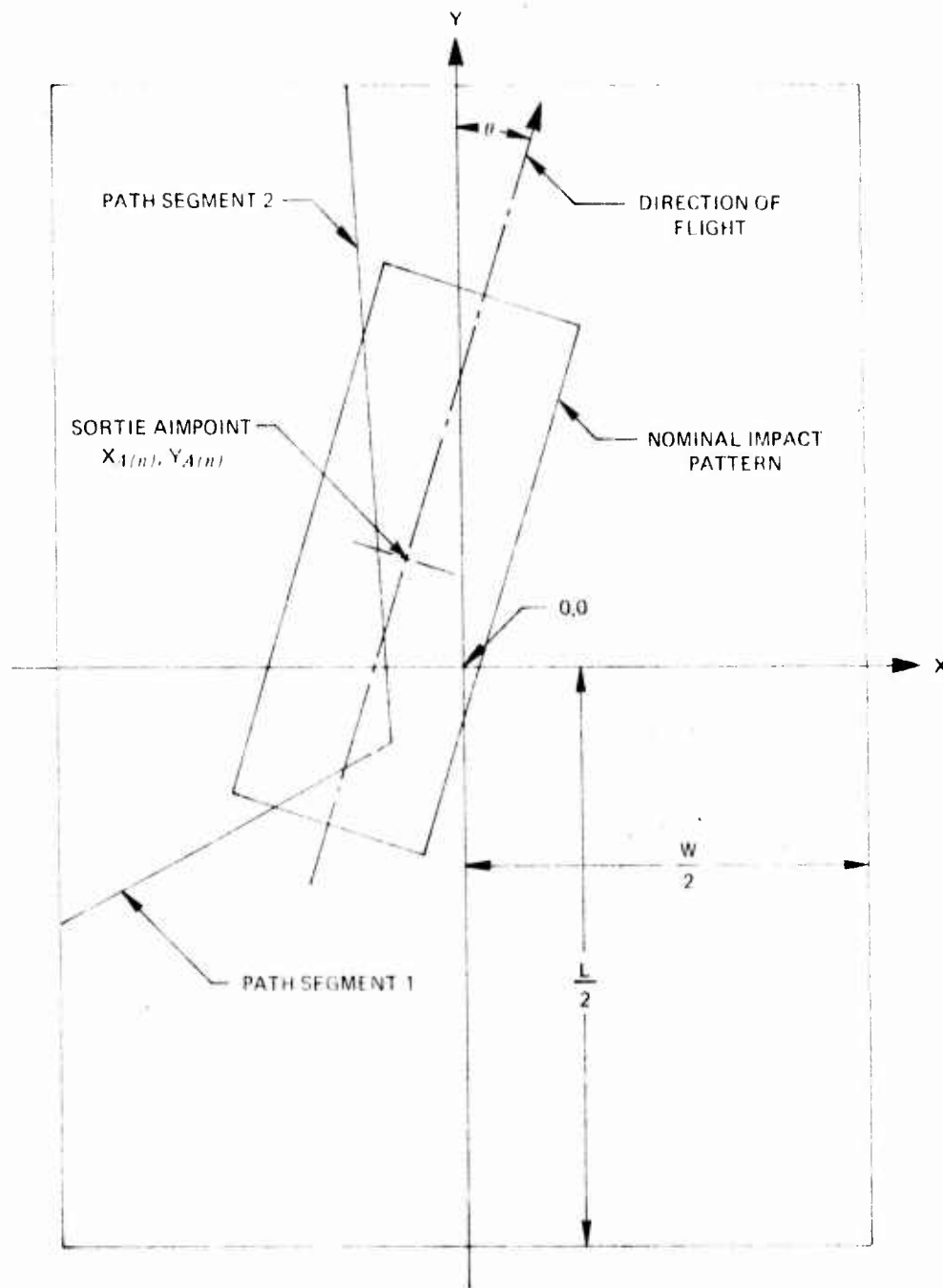


Figure 1. Map Coordinate System

- Each path segment must be less than 6,553 feet in length.
- The maximum number of mines within range of influence of any target for a path segment is limited to 4,998.
- The maximum number of mines which are permitted to fall within the width of the target area rectangle (W) and the length of the target area rectangle (L) is 32,767.
- Only those mines which are within 1,638 feet on either side of the nominal centerline of the path segment are considered. (Only those mines within the range of influence of a target are saved.)
- For mines with a timing cycle, the total of ON and OFF time for a single cycle must be less than 256 seconds.

The travel path coordinate system is defined with the origin at the path segment starting point; the positive Y axis defines the direction of travel, and the positive X axis is to the right. The mine locations are transformed into the path coordinate system using the expressions (see Figure 2):

$$X_{MR}(i) = [X_{M(i)} - X_{R1}] \cos \theta_R + [Y_{M(i)} - Y_{R1}] \sin \theta_R \quad (8)$$

$$Y_{MR}(i) = - [X_{M(i)} - X_{R1}] \sin \theta_R + [Y_{M(i)} - Y_{R1}] \cos \theta_R \quad (9)$$

where  $X_{MR}(i)$  and  $Y_{MR}(i)$  are the X and Y coordinates of the  $i^{\text{th}}$  mine in the path coordinate system,  $X_{M(i)}$  and  $Y_{M(i)}$  are as described above,  $X_{R1}$  and  $Y_{R1}$  are the X and Y coordinates of the path segment starting point in the map coordinate system, and  $\theta_R$  is the computed angle between the X and  $X_R$  axes.

After the coordinates of a mine are computed as described above, the model determines if the mine is within the range of influence. A mine is outside this range of influence and is eliminated from further consideration if its distance to the travel path is so large that both of the following conditions exist.

- The probability that the sweeping element will detect the mine is zero.

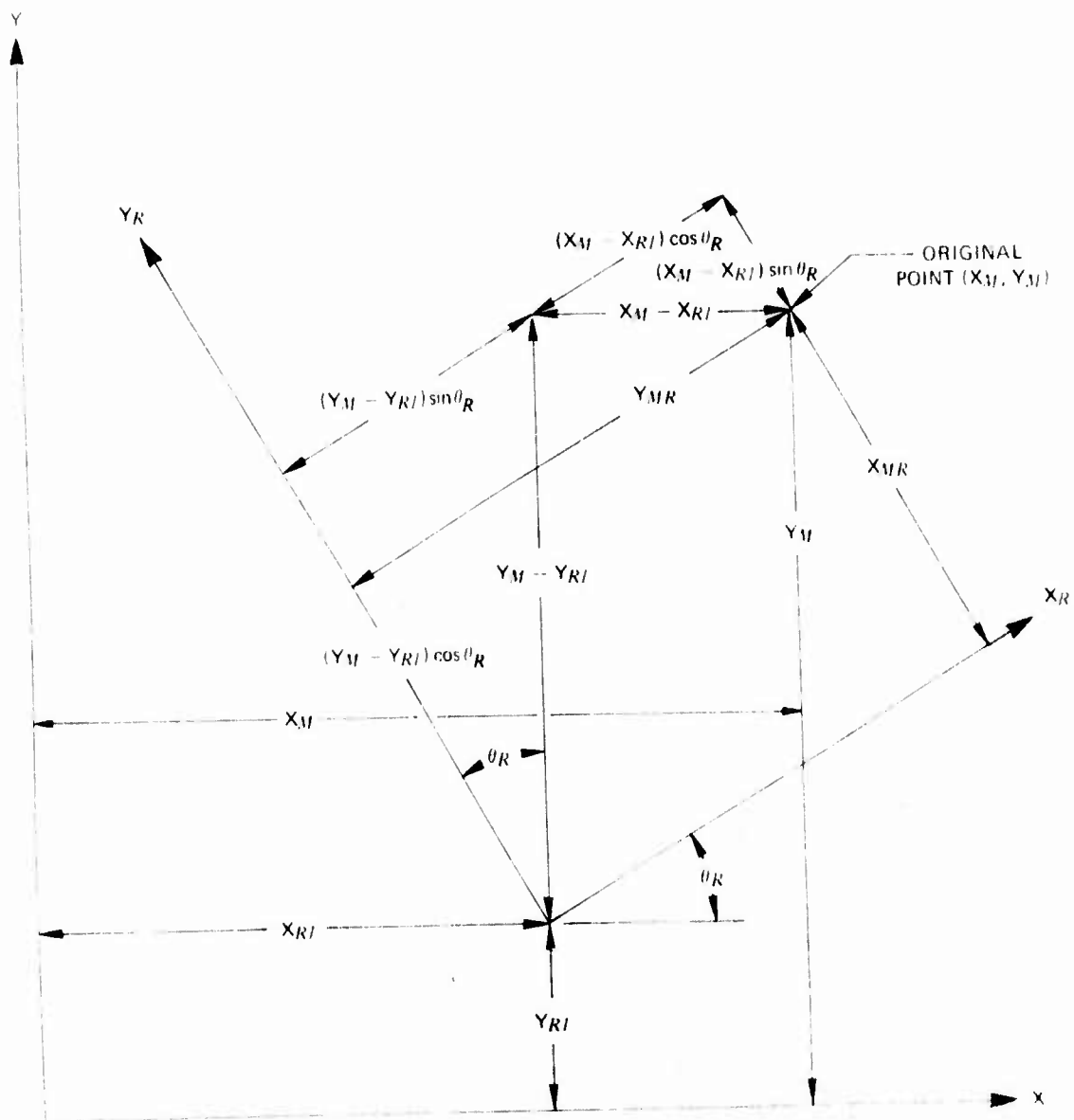


Figure 2. Transformation of Mine Locations From Map Coordinate System Into Travel Path Coordinate System

- The probability that the mine will detonate is zero.

A mine is also eliminated from further consideration for the path segment if its  $Y_{MR}$  coordinate is less than zero or greater than the length of the path segment. [A mine can conceivably be within the range of influence and within the path segment length for more than one path segment. Such a mine, if detonated while the targets are traversing one path segment, cannot be detonated while the targets are traversing a subsequent segment. To account for this situation, a binary switch is maintained (one bit for each mine) to ensure that a detonated mine is not included in a subsequent path segment.]

For mines controlled by an ON/OFF timing cycle, a random starting time for each mine is computed using the expression:

$$t_s = U_{RN}(t_{on} + t_{off}) \quad (10)$$

where  $t_s$  is the random starting point for the fuze timing cycle,  $U_{RN}$  is a uniform random number,  $t_{on}$  is the duration of the active portion of the fuze timing cycle, and  $t_{off}$  is the duration of the inactive portion of the fuze timing cycle.

For each mine on a path segment, the mine information described above is packed into a 60-bit CDC 6600 word in the following manner.

0yyyyyyyyyyyyyyyyysxxxxxxxxxxxxxxxxiiiiiiiiittttttmmmmmmmmmmmmmmmmmm

The meaning of each bit position is described in Table 2.

Since each component of the vector containing the packed mine information looks like a positive integer, the vector can be sorted into ascending order, thereby ordering the vector by increasing mine Y coordinate. The sorting technique utilized was developed by Mr. Richard Singleton under a Stanford Research Institute project. Details pertaining to this sorting technique can be found in Algorithm 347 of Collected Algorithms from Communications of the Association for Computing Machinery.

Initial positions of the targets are specified as inputs, with the first target input assuming a position on the path segment centerline and at the path segment starting point (i.e., the origin of the path coordinate system). All other

TABLE 2. BIT LOCATIONS FOR PACKED EVENT INFORMATION

BIT	LETTER CODE	DESCRIPTION
59	0	Always zero to create a word which looks like a positive integer.
58 through 43	y	Sixteen bits containing the event Y coordinate $[Y_{MR}(i)]$ times 10.
42	s	The sign of the event X coordinate $[X_{MR}(i)]$ .
41 through 28	x	Fourteen bits containing the magnitude of the event X coordinate $ X_{MR}(i) $ times 10.
27 through 20	i	Eight bits containing the mine random starting time ( $t_s$ ), if any, or the direct fire area number.
19 through 15	t	Five bits containing the event type (see NOBTP2 in the List of Symbols and Abbreviations - Simulation Model).
14 through 0	m	Fifteen bits containing the sequential mine number or the indirect fire volley aimpoint.

targets are placed at positions relative to the first as defined by their input [i.e.,  $X_T(j)$  and  $Y_T(j)$ ] locations. All of the targets travel along the path in formation; when all of the targets have either completed traversing the first path segment or have been damaged, the undamaged targets are placed in a similar manner at the start of the next path segment (if any). The formation of targets is maintained rather than closing up any gaps left by damaged targets.

#### Determination of Next Event

A simplified method is employed for determining the next event which will occur. The vector containing the packed event information has been sorted by ascending Y coordinate so that the next event can be determined by incrementing a counter. If the  $i^{\text{th}}$  event is the next one to be encountered by the  $j^{\text{th}}$  target, the expression used to determine the

range distance that must be traveled by the target to reach the point of closest approach to the event  $[\Delta Y_{(ij)}]$  is:

$$\Delta Y_{(ij)} = |Y_{MR(i)} - Y_{T(j)}| \quad (11)$$

where  $Y_{T(j)}$  is the Y coordinate of the  $j^{\text{th}}$  target and  $Y_{MR(i)}$  is as defined above. The smallest  $\Delta Y_{(ij)}$  is determined and the time required for the target associated with the event to travel this distance is computed. If the event target is not involved in a delay condition, the associated target/event combination constitutes the next position oriented event. Before this target is involved in the next event, the time oriented events are considered. All time ordered events that will occur in less time than the travel time to the next event are considered before the event target is moved the distance  $\Delta Y_{(ij)}$ .

#### Evaluation of Target/Mine and Target/Boundary Interactions

At this point in the execution of an event, the event target and the event mine have the same Y coordinate (i.e., the target is at its point of closest approach to the mine). The lateral distance between the target and the mine  $[\Delta X_{(ij)}]$  is computed using the expression:

$$\Delta X_{(ij)} = X_{MR(i)} - X_{T(j)} \quad (12)$$

where  $X_{T(j)}$  is the X coordinate of the  $j^{\text{th}}$  target, and the other variables are as defined above.

When some targets are damaged, subsequent targets in the same column must divert around the damaged target. If a target is within 60 feet of a damaged target and must divert around the damaged target, Equation (12) is modified to reflect the diversion geometry (see Figure 3). The target offset is computed using the expression:

$$X_{TO} = \sqrt{(75)^2 - [Y_{T(j)} - Y_{DIV}]^2} - 45 \quad (13)$$

where  $X_{TO}$  is the magnitude of the target offset while diverting around the damaged target,  $Y_{T(j)}$  is as defined above,

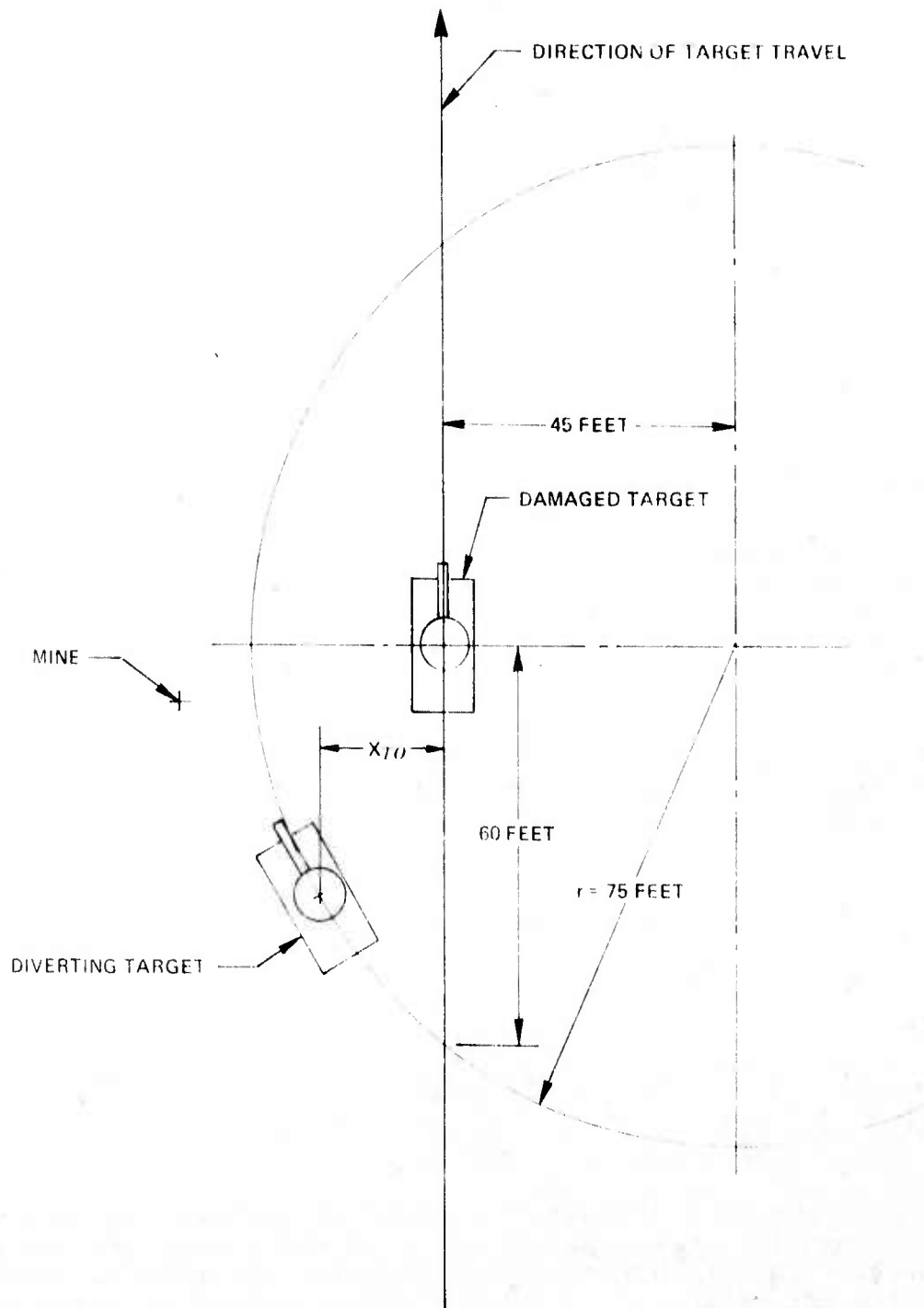


Figure 3. Geometry of Target Diversion

$Y_{DIV}$  is the Y coordinate of the damaged target, the quantity 75 is the radius of the diversion circle in feet, and the quantity 45 is the distance in feet from the center of the diversion circle to the nominal travel path.

When the event target is diverting around a damaged target while at the point of closest approach to the event mine, the lateral distance between the target and the mine  $[\Delta X(ij)]$  is computed using the expression:

$$\Delta X(ij) = |X_{MR}(i) - X_T(j) - X_{TO}| \quad (14)$$

where all variables are as defined above. If a diverting target is damaged by a mine while traveling on the diversion circle and the diverting target must itself be diverted around, subsequent targets are diverted in the opposite direction, causing a change in the sign of  $X_{TO}$ . If two targets which must be diverted around are damaged, one while diverting left and one while diverting right, a time delay is assessed for remaining targets in the column to provide for removal of the three damaged targets.

A standard linear interpolation technique is used to provide for the computation of probabilities of mine detection (sweeping), mine detonation, and target damage. The expression used to perform the linear interpolation is:

$$P = P_q + \frac{[\Delta X(ij) - R_q][P_{q+1} - P_q]}{(R_{q+1} - R_q)} \quad (15)$$

where  $P$  is the interpolated probability,  $\Delta X(ij)$  is the lateral distance between the target and the mine,  $R_q$  and  $R_{q+1}$  are ranges from the tabular input of the function being interpolated such that  $R_q \leq \Delta X(ij) < R_{q+1}$ , and  $P_q$  and  $P_{q+1}$  are probabilities corresponding to  $R_q$  and  $R_{q+1}$ .

If the event target is capable of sweeping the event mine, Equation (15) is evaluated utilizing the appropriate sweeping function to obtain the probability that the mine is detected by the target ( $P_{dt}$ ). A uniform random number is selected and compared to the value of  $P_{dt}$ . If the random number is less than the value of  $P_{dt}$ , the mine is detected and neutralized by the target, and the event is completed.

If the mine is not detected, the model next determines whether the mine detonates. If the mine employs a fuze timing device, SEMAC determines if the fuze is active at the time that the target reaches the point of closest approach. To accomplish this, the number of times that the fuze has completed its timing cycle during the simulation is determined using the expression:

$$N_t = \frac{t_r - t_s}{t_{on} + t_{off}} \quad (16)$$

where  $N_t$  is the number of complete fuze cycles (truncated to an integer number),  $t_r$  is the time period used in the simulation to the specific point in time when the event occurs,  $t_s$  is the random starting point for the fuze timing cycle (which must be smaller than the total fuze cycle time), and  $t_{on}$  and  $t_{off}$  are as defined above.

To determine whether the fuze is active at the time the event occurs, the following expression is evaluated.

$$T_t = t_r - N_t(t_{on} + t_{off}) \quad (17)$$

where  $t_r$ ,  $N_t$ ,  $t_{on}$ , and  $t_{off}$  are as defined above, and  $T_t$  is a time period relating to the fuze timing cycle. If the value of  $T_t$  is less than the time that the fuze is active, the mine is considered active and can detonate.

Equation (15) is evaluated utilizing the appropriate detonation function to obtain the probability that the mine will detonate ( $P_{dn}$ ). A uniform random number is selected and compared to the value of  $P_{dn}$ . If the random number is less than the value of  $P_{dn}$ , the mine is detonated.

The effect of a mine which detonates is evaluated against all targets which are close enough to be damaged by the mine. The distance from the mine to each target ( $DMT$ ) is determined using the expression:

$$DMT = \sqrt{[X_{MR}(i) - X_T(j) - X_{TO}]^2 + [Y_{MR}(i) - Y_T(j)]^2} \quad (18)$$

where all variables are as defined above. Substituting  $DMT$  for  $\Delta X(ij)$ , Equation (15) is evaluated utilizing the

appropriate damage function to obtain the probability that the target is damaged ( $P_d$ ). A uniform random number is selected and compared to the value of  $P_d$ . If the random number is less than the value of  $P_d$ , the target is considered damaged.

SEMAC permits targets to be associated with a lead target as a group or unit (e.g., troops on foot following a tank). If a lead target is damaged, three options are available for determining the disposition of its associated targets. The first choice is to proceed in the original direction if another target of the same type as the damaged target is in front of the associated targets in the column. If the first choice is not available, the associated targets can wait for a following lead target in the same column to reach their position and fall in behind it. If neither of the first two options is available, it is assumed that the associated targets will return back the path that they were traveling and will not attempt to breach the minefield again. These targets are assumed to be lost to the current mission and are so reported, but they are not included in the damaged target count.

One other type of interaction which is included in SEMAC and which results in degradation of the minefield is termed sympathetic detonations. Sympathetic detonations can occur if the disturbance caused by a detonating mine or exploding munition is capable of detonating nearby mines. If the sympathetic detonation options are chosen, the maximum range at which a detonating mine or exploding direct or indirect fire munition is input. When a mine detonates or a direct or indirect fire munition explodes, the mines within the maximum distance in the Y direction are identified; then, the distance to each of those mines is determined individually using the expression:

$$D_{(ik)} = \sqrt{[X_{MR(k)} - X_{MR(i)}]^2 + [Y_{MR(k)} - Y_{MR(i)}]^2} \quad (19)$$

where  $D_{(ik)}$  is the distance from the  $i^{\text{th}}$  (detonating mine or exploding munition) to the  $k^{\text{th}}$  mine, and the  $X_{MR}$  and  $Y_{MR}$  are the mine coordinates. If  $D_{(ik)}$  is less than the distance at which the  $i^{\text{th}}$  mine or exploding munition can sympathetically detonate the  $k^{\text{th}}$  mine, the  $k^{\text{th}}$  mine detonates.

## Minesweeping Techniques

The intruding forces may employ the following types of minesweeping techniques:

- Visual sweeping.
- Plows and rollers.
- Fuel-air-explosives (FAE).
- Line charges.

Visual sweeping of the mines is accomplished as the intruders traverse the minefield. As the mines are detected, the mines are either removed or detonated. Rollers and plows are modeled as separate target types located a fixed distance in front of another target. The range dependent probability functions of detonation and damage are input to the computer program. A nonunity probability of encounter for a roller is used to model the roller bouncing over a mine. For a plowed mine, the probability that the mine will still function is input. The target type employing the FAE or line charges is input in addition to the pattern effects dimensions. The pattern produced by the FAE is circular and is computed using the following expressions:

$$Y_{PC}(i) = Y_T(j) + D_F \quad (20)$$

$$Y_{PO}(i) = Y_{PC}(i) + R_F \quad (21)$$

$$Y_{PI}(i) = Y_{PC}(i) - R_F \quad (22)$$

where  $Y_{PC}(i)$  is the center of the pattern,  $Y_T(j)$  is the Y coordinate of the jth target,  $D_F$  is the distance in front of the target where the FAE detonates,  $Y_{PO}(i)$  is the furthest point from the target of the FAE effects,  $R_F$  is the radius of the FAE effects, and  $Y_{PI}(i)$  is the closest point to the target of the FAE effects. If a mine is located within the pattern of the FAE effects, a uniform random number ( $U_{RN}$ ) is selected, and this number is compared to the input value of the probability of mine detonation within the FAE pattern ( $P_{DF}$ ). If the inequality:

is satisfied, the mine is eliminated from further consideration.

For line charges, the pattern is considered to be rectangular and originates just in front of the intruder which deploys the device. The line charge effects are  $L_{LC}$  in length and  $W_{LC}$  in width. If a mine is located within the pattern of the line charge effects, a uniform random number (URN) is selected and compared to the input value of the probability of mine detonation within the line charge pattern (PDF). Inequality (23) is evaluated, and if the inequality is satisfied, the mine is eliminated from further consideration.

The locations for FAE or line charge deployment are determined at the beginning of each travel path segment and it is assumed that sufficient devices are available to sweep across the minefield. The line charge effect is continuous in that when the effective length of one line charge detonation is reached, another line charge is deployed. This is not so for FAE sweeping devices because the area of effectiveness is circular and they are deployed a specified distance in front of the intruding target. The deployment positions along the travel path segment are assigned the X coordinate of the intruder which deploys the devices and the information is packed into a computer word just like the mine information, except that the variable NOBTP2 is set to 19 for each deployment position.

#### COMPUTATION OF EFFECTIVENESS OF DIRECT FIRE MUNITIONS

SEMAC contains methodology for determining the effectiveness of direct fire munitions employed by the defending forces and return fire by the intruders. The program can consider up to 20 defenders of up to five different types firing into portions of the engagement area not protected by terrain shielding.

#### Discussion of Direct Fire Logic

For the purposes of this discussion, direct fire is defined as surface-to-surface munition deployment when the attacking element weapon system can "see" the unitary target either visually or with electronic aid.

SEMAC models the effects of terrain shielding by allowing the user to define portions of the engagement area into which direct fire can be achieved. A defender is defined to be at some particular X, Y coordinate in the engagement area and may shoot into some or all of the direct fire areas. The portion of the engagement area into which a defender cannot shoot (and from which the defender cannot receive return fire) is considered to be shielded by terrain features.

Whereas intruder/mine encounters are position (or distance) related, the logic for direct and return fire is time ordered once an intruder enters a direct fire area. Priority lists defining the order in which a defender type will shoot at the various intruder types (and vice versa) is provided for the case. In addition, the direct fire areas into which each defender can shoot (and receive return fire) are provided. When an intruder of a type on a defenders priority list enters a direct fire area into which that defender can shoot, the defender fires and the results are evaluated. That defender cannot fire again until the time to reload and aim has elapsed. Although the defender was prepared to fire into one of the direct fire areas as soon as an intruder on the priority list was observed, the intruder may not immediately be prepared to return fire. In addition, the intruder may require some time to find the defender in order to return the fire. This situation is modeled in SEMAC by requiring that an intruder (by type) observe a specified number of rounds before beginning return fire.

SEMAC handles the combination of distance related and time ordered events in the following manner. Subroutine LOOPS determines what the next distance related event is to be. This can be an intruder encountering a mine, a travel path segment boundary, a line charge or fuel air explosive deployment location, a direct fire area entrance or exit boundary, or the point at which the first volley is fired at an indirect fire volley aimpoint. Before the distance related event takes place, the possible time ordered events are checked to determine if one or more should occur first. The possible time ordered events are direct fire shots, return fire shots, or the delivery of the second or subsequent volley at an indirect fire volley aimpoint. If one or more time ordered events should occur before the intruders get to the distance related event, the intruders are moved to the location at which each time ordered event is to take place, and the event results are evaluated. Finally, when all of the required time ordered events have occurred, the intruders are moved the remaining distance and the event determined by Subroutine LOOPS is evaluated. This logic is included in Subroutine CREVTM.

### Transformation of Direct Fire Area Boundaries Into Travel Path Coordinate System

The entrance and exit boundary coordinates of the direct fire areas are input in the map coordinate system on, or adjacent to, the appropriate travel path segment. The Y coordinate of the direct fire area entrance or exit boundary  $[Y_{DR(i)}]$  in the travel path coordinate system is computed by the expression:

$$Y_{DR(i)} = - [X_{D(i)} - X_{R1}] \sin \theta_R + [Y_{D(i)} - Y_{R1}] \cos \theta_R \quad (24)$$

where  $X_{D(i)}$  and  $Y_{D(i)}$  are the X and Y coordinates of the  $i$ th entrance or exit boundary in the map coordinate system,  $X_{R1}$  and  $Y_{R1}$  are the X and Y coordinates of the travel path segment starting point in the map coordinate system, and  $\theta_R$  is the computed angle between the X and  $X_R$  axes (see Figure 2). The transformed X coordinate is not required.

The transformed direct fire area entrance and exit boundaries are packed into a computer word just like the mine information, except that the variable NOBTP2 is set to 17 for entrance boundaries and 18 for exit boundaries.

### Direct Fire Effectiveness Computations

When, during the evaluation of time ordered events, a defender is to fire on an intruder or vice versa, Subroutine DIRFIR is called to perform the computations. Two types of effectiveness indices are permitted, which are:

- Mean area of effectiveness for fragmentation ( $MAE_f$ ) in the ground plane in square feet.
- Probability of damage given a hit ( $P_{HD}$ ).

When  $MAE_f$  is specified, it is assumed that the round will arrive with a nonzero impact angle, such that the round will impact on the target or near the target on the ground. The  $P_{HD}$  option can be used to model the situation where the weapon may go past the target and detonate harmlessly some distance away.

If the effectiveness index type is  $MAE_f$ , the weapon length-to-width ratio ( $a$ ) is computed using the expression:

$$a = 1 - 0.8[\cos(I)] \quad (25)$$

where  $I$  is the impact angle. The effective target length ( $L_{ET}$ ) and the effective target width ( $W_{ET}$ ) are computed using the expressions:

$$L_{ET} = 2\sqrt{\frac{(MAE_f)(a)}{\pi}} \quad (26)$$

$$W_{ET} = \frac{L_{ET}}{a} \quad (27)$$

The range single shot probability of hit (RSSP) and the deflection single shot probability of hit (DSSP) are computed by the expressions:

$$RSSP = \frac{L_{ET}}{\sqrt{17.6(REP)^2 + (L_{ET})^2}} \quad (28)$$

$$DSSP = \frac{W_{ET}}{\sqrt{17.6(DEP)^2 + (W_{ET})^2}} \quad (29)$$

where REP and DEP are the range error probable and the deflection error probable in the ground plane. Finally, the single shot probability of damage ( $SSP_D$ ) is computed by the expression:

$$SSP_D = RSSP(DSSP)(R) \quad (30)$$

where  $R$  is the reliability of the round.

If the effectiveness index is in terms of  $P_{HD}$ , the circular error probable in the normal plane (CEP) can be input in feet or mils. When CEP is in mils, it is converted to feet by the expression:

$$CEP = \frac{CEP_{mils}}{1000} \left[ \sqrt{[X_T(j) - X_D(k)]^2 + [Y_T(j) - Y_D(k)]^2} \right] \quad (31)$$

where  $X_T(j)$  and  $Y_T(j)$  are the coordinates of the intruder, and  $X_D(k)$  and  $Y_D(k)$  are the coordinates of the defender. The target, whether intruder or defender, is attacked with consideration being given to its presented dimensions in the vertical and horizontal directions. The vertical dimension is target height, whereas the horizontal dimension is either twice the target radius (for circular targets) or the average of target length and width (for rectangular targets). Intermediate factors  $a_1$  and  $b_1$  are computed by the expressions:

$$a_1 = \frac{T_H}{1.7(CEP)} \quad (32)$$

$$b_1 = \frac{T_V}{1.7(CEP)} \quad (33)$$

where  $T_H$  and  $T_V$  are the target horizontal and vertical dimension, respectively.

The single shot probability of damage is then computed by the expression:

$$SSP_D = \frac{2}{\sqrt{2\pi}} \int_0^{a_1} e^{-\frac{x^2}{2}} dx \left[ \frac{2}{\sqrt{2\pi}} \int_0^{b_1} e^{-\frac{x^2}{2}} dx \right] (R) (P_{HD}) \quad (34)$$

where the integrals represent the cumulative normal distribution function, and the other variables are defined above.

The  $SSP_D$  value is then compared to a random number from a uniform distribution ( $U_{RN}$ ) and if the inequality:

$$U_{RN} < SSP_D \quad (35)$$

is satisfied, the target is damaged and is removed from further consideration.

## COMPUTATION OF EFFECTIVENESS OF INDIRECT FIRE MUNITIONS

The indirect attack munition effectiveness methodology is derived from methodology accepted by the Joint Technical Coordinating Group for Munitions Effectiveness (JTTCG/ME) and published in the Joint Munitions Effectiveness Manual, Air-to-Surface (JMEM/AS) Basic Manual. The methodology can assess weapon effectiveness against unitary targets. The effectiveness index can be in terms of mean area of effectiveness for fragmentation in the ground plane ( $MAE_f$ ), vulnerable area in the normal plane ( $VA_N$ ), mean area of effectiveness for blast in the ground plane ( $MAE_b$ ), or, for effective miss distance in the ground plane (EMD). The delivery accuracy is input as range error probable (REP) and deflection error probable (DEP) in the ground plane.

The methodology assumes that the indirect fire weapons are directed by a forward observer who provides an estimated time of arrival (ETA) for the lead target at a predetermined location in the minefield. When this point is reached, the weapons each fire a specified number of volleys, with a specified delay between volleys for reloading. The first volley arrives when the lead target is at the point of closest approach to the origin of the volley pattern coordinate system, thus providing the user a degree of flexibility regarding the volley pattern coverage of the target formation for the first volley.

### Transformation of Indirect Fire Volley Aimpoint Locations Into Travel Path Coordinate System

At the beginning of each travel path segment, the indirect fire volley aimpoint locations along the travel path segment are computed. First, the specified origins of the volley aimpoint coordinate systems are transformed into the travel path coordinate system using the expressions:

$$X_{VAP(i)} = X_{OVAP(i)} - X_{R1} \cos \theta_R + [Y_{OVAP(i)} - Y_{R1}] \sin \theta_R \quad (36)$$

$$Y_{VAP(i)} = - [X_{OVAP(i)} - X_{R1}] \sin \theta_R + [Y_{OVAP(i)} - Y_{R1}] \cos \theta_R \quad (37)$$

where  $X_{VAP(i)}$  and  $Y_{VAP(i)}$  are the X and Y coordinates of the  $i$ th volley aimpoint origin in the path coordinate system,  $X_{OVAP(i)}$  and  $Y_{OVAP(i)}$  are the X and Y coordinates of the  $i$ th volley aimpoint origin in the map coordinate system,  $X_{R1}$

and  $Y_{R1}$  are the X and Y coordinates of the path segment starting point in the map coordinate system, and  $\theta_R$  is the computed angle between the X and  $X_R$  axes (see Figure 2).

After the coordinates of the volley aimpoint origins are computed as described above, the model determines if the aimpoint is within the range of target travel for this path segment. An aimpoint is eliminated if its  $Y_{VAP(i)}$  coordinate is less than zero or greater than the length of the path segment, or if its  $X_{VAP(i)}$  coordinate is not within 1,638 feet on either side of the nominal centerline of the path segment.

For each volley aimpoint, the X and Y coordinates, the volley aimpoint number and the intruder leading the column of intruding targets that initiates the indirect fire at this aimpoint is packed into a 60-bit CDC 6600 word. The variable NOBTP2 is set to 16 for indirect fire volley aimpoint origins.

The first volley at each aimpoint is timed to arrive when the lead target is at the point of closest approach to the origin of the volley pattern. Subsequent volleys will arrive at the aimpoint based on the amount of time required for the indirect fire weapons to reload.

The X and Y coordinates of the desired mean points of impact for each round in the volley are input with respect to the origin of the volley pattern. The expressions:

$$X_{RDMPI}(j) = X_{OVAP}(i) + X_{RDPI}(j)\cos\theta_F(i) + Y_{RDPI}(j)\sin\theta_F(i) \quad (38)$$

$$Y_{RDMPI}(j) = Y_{OVAP}(i) + Y_{RDPI}(j)\cos\theta_F(i) - X_{RDPI}(j)\sin\theta_F(i) \quad (39)$$

transform the desired mean points of impact for each round in the volley,  $X_{RDMPI}(i)$  and  $Y_{RDMPI}(i)$ , into the map coordinate system, where  $X_{OVAP}(i)$  and  $Y_{OVAP}(i)$  are as described above,  $X_{RDPI}(j)$  and  $Y_{RDPI}(j)$  are the X and Y coordinates of the desired points of impact with respect to the origin of the volley pattern and  $\theta_F(i)$  is the direction of attack for the indirect fire volley aimpoint, measured clockwise from the positive Y axis in the map coordinate system.

The desired points of impact for each round in the volley pattern are transformed into the travel path system by the expressions:

$$X_{RI}(j) = [X_{RDMPI}(j) - X_{R1}] \cos \theta_R + [Y_{RDMPI}(j) - Y_{R1}] \sin \theta_R \quad (40)$$

$$Y_{RI}(j) = [X_{RDMPI}(j) - X_{R1}] \sin \theta_R + [Y_{RDMPI}(j) - Y_{R1}] \cos \theta_R \quad (41)$$

where  $X_{RDMPI}(j)$ ,  $Y_{RDMPI}(j)$ ,  $X_{R1}$ ,  $Y_{R1}$ ,  $\theta_R$  are as described above, and  $X_{RI}(j)$  and  $Y_{RI}(j)$  are the transformed X and Y coordinates of the desired points of impacts for each round in the volley pattern.

### Indirect Fire Effectiveness Computations

Each indirect fire volley is composed of one or more rounds detonating in the vicinity of the targets at approximately the same time. The cumulative effects of the rounds are determined for each target individually by the SEMAC methodology. Considering the  $i^{\text{th}}$  round and the  $j^{\text{th}}$  target, the components of offset,  $R_0$  and  $D_0$ , are computed by the expressions:

$$R_0 = |[X_{RI}(i) - X_T(j)] \sin \theta_{FR} + [Y_{RI}(i) - Y_T(j)] \cos \theta_{FR}| \quad (42)$$

$$D_0 = |- [X_{RI}(i) - X_T(j)] \cos \theta_{FR} + [Y_{RI}(i) - Y_T(j)] \sin \theta_{FR}| \quad (43)$$

where  $R_0$  is the target/round distance in the range direction,  $D_0$  is the target/round distance in the deflection direction,  $X_{RI}(i)$  and  $Y_{RI}(i)$  are the X and Y coordinates of the desired mean point of impact for the round,  $X_T(j)$  and  $Y_T(j)$  are the X and Y coordinates of the target, and  $\theta_{FR}$  is the direction of attack for the volley measured in the travel path coordinate system.

If the round is an improved conventional munition (ICM), the pattern length ( $L_p$ ) and width ( $W_p$ ) are computed by the expression:

$$L_p = W_p = \sqrt{\pi} (k_p) \quad (44)$$

where  $R_p$  is the radius of the ICM pattern, and the probability of damage given that the target is in the pattern ( $P_{HD}$ ) is computed by the expression:

$$P_{HD} = 1 - e^{-\frac{N_S (R_S) (MAE_f)}{L_P (W_P)}} \quad (45)$$

where  $N_S$  is the number of submunitions and  $R_S$  is the reliability of the submunition. The effective target length ( $L_{ET}$ ) is set to the value of  $L_P$ , and the effective target width ( $W_{ET}$ ) is set to the value of  $W_P$ .

If the effectiveness index type is  $MAE_f$ , and the round is high explosive (HE), the weapon length-to-width ratio ( $a$ ) is computed using the expression:

$$a = 1 = 0.8[\cos(I)] \quad (46)$$

where  $I$  is the weapon impact angle. The effective target length ( $L_{ET}$ ) and the effective target width ( $W_{ET}$ ) are computed using the expressions:

$$L_{ET} = 2 \sqrt{\frac{(MAE_f)(a)}{\pi}} \quad (47)$$

$$W_{ET} = \frac{L_{ET}}{a} \quad (48)$$

The range single shot probability of hit (RSSP) and the deflection single shot probability of hit (DSSP) are computed by the expressions:

$$RSSP = \frac{L_{ET}}{\sqrt{17.6(REP)^2 + (L_{ET})^2}} e^{-\frac{4(R_0)^2}{17.6(REP)^2 + (L_{ET})^2}} \quad (49)$$

$$DSSP = \frac{W_{ET}}{\sqrt{17.6(DEP)^2 + (W_{ET})^2}} e^{-\frac{4(D_0)^2}{17.6(DEP)^2 + (W_{ET})^2}} \quad (50)$$

where REP and DEP are the range error probable and the deflection error probable in the ground plane, respectively, and all

other variables are as defined above. The single shot probability of damage ( $SSP_D$ ) is then calculated by the expression:

$$SSP_D = RSSP(DSSP)(R)(P_{HD}) \quad (51)$$

where  $R$  is the reliability of the round.

If the effectiveness index type is  $MAE_b$ , the values of  $L_{ET}$  and  $W_{ET}$  are computed by the expression:

$$L_{ET} = W_{ET} = \sqrt{MAE_b} \quad (52)$$

and for  $VA_N$  the values are computed using the expressions:

$$L_{ET} = \frac{\sqrt{VA_N}}{\sin(I)} \quad (53)$$

$$W_{ET} = \sqrt{VA_N} \quad (54)$$

If the effectiveness index is in terms of EMD and the target is rectangular, the values of  $L_{ET}$  and  $W_{ET}$  are computed using the expressions:

$$L_{ET} = L_T + 2(EMD) \quad (55)$$

$$W_{ET} = W_T + 2(EMD) \quad (56)$$

where  $L_T$  is the target length and  $W$  is the target width. If the target is circular, the values are computed using the expression:

$$L_{ET} = W_{ET} = \sqrt{\pi(R_T + EMD)} \quad (57)$$

where  $R_T$  is the target radius. If target height is being considered, the shadow length ( $L_{SH}$ ) is computed using the expression:

$$L_{SH} = \frac{H_T}{\tan(I)} \quad (58)$$

where  $H_T$  is the target height. If the shadow length is greater than the effective miss distance, the value of  $L_{ET}$  is recomputed using the expression:

$$L_{ET} = \frac{[L_T + 2(EMD)](W_{ET}) + W_T(L_{SH} - EMD)}{W_{ET}} \quad (59)$$

for rectangular targets, or:

$$L_{ET} = \frac{\pi(R_T + EMD)^2 + 2(R_T)(L_{SH} - EMD)}{\sqrt{\pi}(R_T + EMD)} \quad (60)$$

for circular targets where all variables are defined above.

Next, four intermediate variables are computed by the expressions:

$$a_1 = \frac{L_{ET} + 2(R_0)}{2.96(REP)} \quad (61)$$

$$a_2 = \frac{|L_{ET} - 2(R_0)|}{2.96(REP)} \quad (62)$$

$$b_1 = \frac{W_{ET} + 2(R_0)}{2.96(DEP)} \quad (63)$$

$$b_2 = \frac{|W_{ET} - 2(R_0)|}{2.96(DEP)} \quad (64)$$

The range single shot probability of hit (RSSP) and the deflection single shot probability of hit (DSSP) are computed by the expressions:

$$\text{RSSP} = \frac{1}{\sqrt{2\pi}} \int_0^{a_1} e^{-\frac{x^2}{2}} dx + \left[ \frac{\text{Sign of}}{L_{ET} - 2(R_0)} \right] \frac{1}{\sqrt{2\pi}} \int_0^{a_2} e^{-\frac{x^2}{2}} dx \quad (65)$$

$$\text{DSSP} = \frac{1}{\sqrt{2\pi}} \int_0^{b_1} e^{-\frac{x^2}{2}} dx + \left[ \frac{\text{Sign of}}{W_{ET} - 2(D_0)} \right] \frac{1}{\sqrt{2\pi}} \int_0^{b_2} e^{-\frac{x^2}{2}} dx \quad (66)$$

and, finally, the single shot probability of damage ( $\text{SSP}_D$ ) is computed by the expression:

$$\text{SSP}_D = \text{RSSP}(\text{DSSP})(R)(P_{HD}) \quad (67)$$

The cumulative probability of damage ( $P_D$ ) for the  $N$  rounds in the volley is computed by the expression:

$$P_D = 1 - \prod_{i=1}^N (1 - \text{SSP}_{Di}) \quad (68)$$

The value of  $P_D$  is compared to a random number from a uniform distribution ( $U_{RN}$ ) and if the inequality:

$$U_{RN} < P_D \quad (69)$$

is satisfied, the target is damaged and is removed from further consideration.

## SECTION III

### FLOWCHARTS

This section contains flowcharts which depict the logical structure of the main program and its 25 subroutines. All flowcharts are based upon the logical intent of this model rather than on displaying the methods used to code the specific routines. Program SEMAC (Figure 4) controls the program by calling the various processing subroutines. Subroutine READIN (Figure 5) reads the data cards and performs initializations and computations which are independent of the iteration. Subroutine SETUP (Figure 6) sets up data for each Monte Carlo iteration and determines mine locations, and Subroutine ROAD (Figure 7) rotates the mines, indirect fire volley aimpoints, and defender positions into the travel path coordinate system and places the targets at the beginning of the travel path segment. Subroutine BOOM (Figure 8) maintains the detonation flag bits for each mine, and Subroutine SORT (Figure 9) performs a Singleton sort on the event information array. Subroutine LOOPS (Figure 10) determines the next event, and Subroutine EVENTC (Figure 11) executes any event involving an intruder and a travel path segment boundary. Subroutine DIVSET (Figure 12) is called to record the location of an intruder which must be diverted around, and Subroutine DIVCHK (Figure 13) determines the offset distance if an intruder is diverting around another intruder. Subroutine UNIT (Figure 14) provides for the unit movement of intruders which are associated with another intruder as a group. Subroutine TGTMIN (Figure 15) determines the results of an encounter between an intruder and a mine, and Subroutine SYMDET (Figure 16) evaluates sympathetic detonations. Subroutine PRINTO (Figure 17) prints the optional output, Subroutine PRINTR (Figure 18) prints the results for each iteration, and Subroutine DISTR (Figure 19) computes and prints distribution output. Subroutine TABINT (Figure 20) performs standard linear interpolations of the tabular probability data, and Subroutine RNORM (Figure 21) selects a random number from a normal distribution. Subroutines IPACK (Figure 22) and UNPACK (Figure 23) pack and unpack data which describes the mines, travel path segment boundaries, and indirect fire volley aimpoints. Subroutine NDFIRE (Figure 24) evaluates the indirect fire munitions employed against the intruders while Subroutine DIRFIR (Figure 25) determines the results of the direct fire munitions employed as direct fire and return fire. Subroutine CKEVTM (Figure 26) determines the time related events, if any, to occur and sequences the events in the proper order. Subroutine STINT (Figure 27) maintains the flag bits for each intruder as they enter and leave direct fire areas. Subroutine EXPSPW (Figure 28) evaluates the effects of line

charges and fuel air explosives against the mines within the range of influence. Finally, Subroutine TGTMOV (Figure 29) moves all active intruders the event distance and time and adjusts the assessed delays for all time related events that may occur.

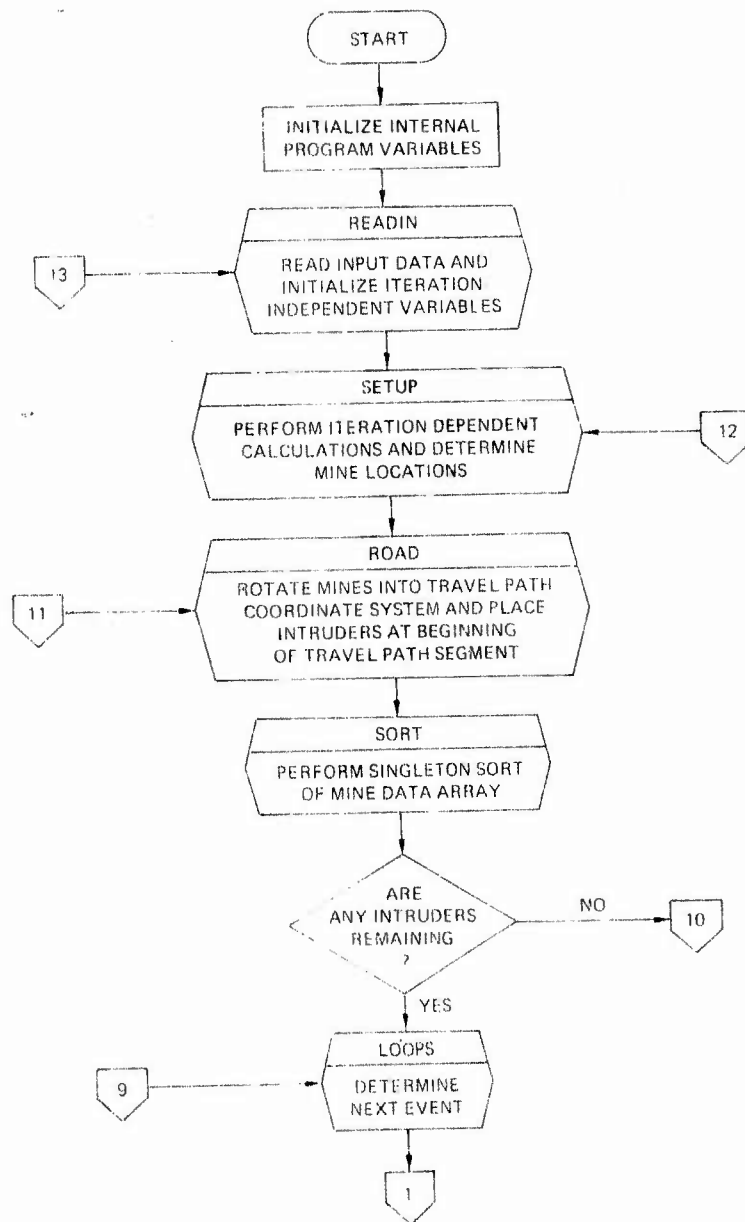


Figure 4. Flowchart, Program SEMAC

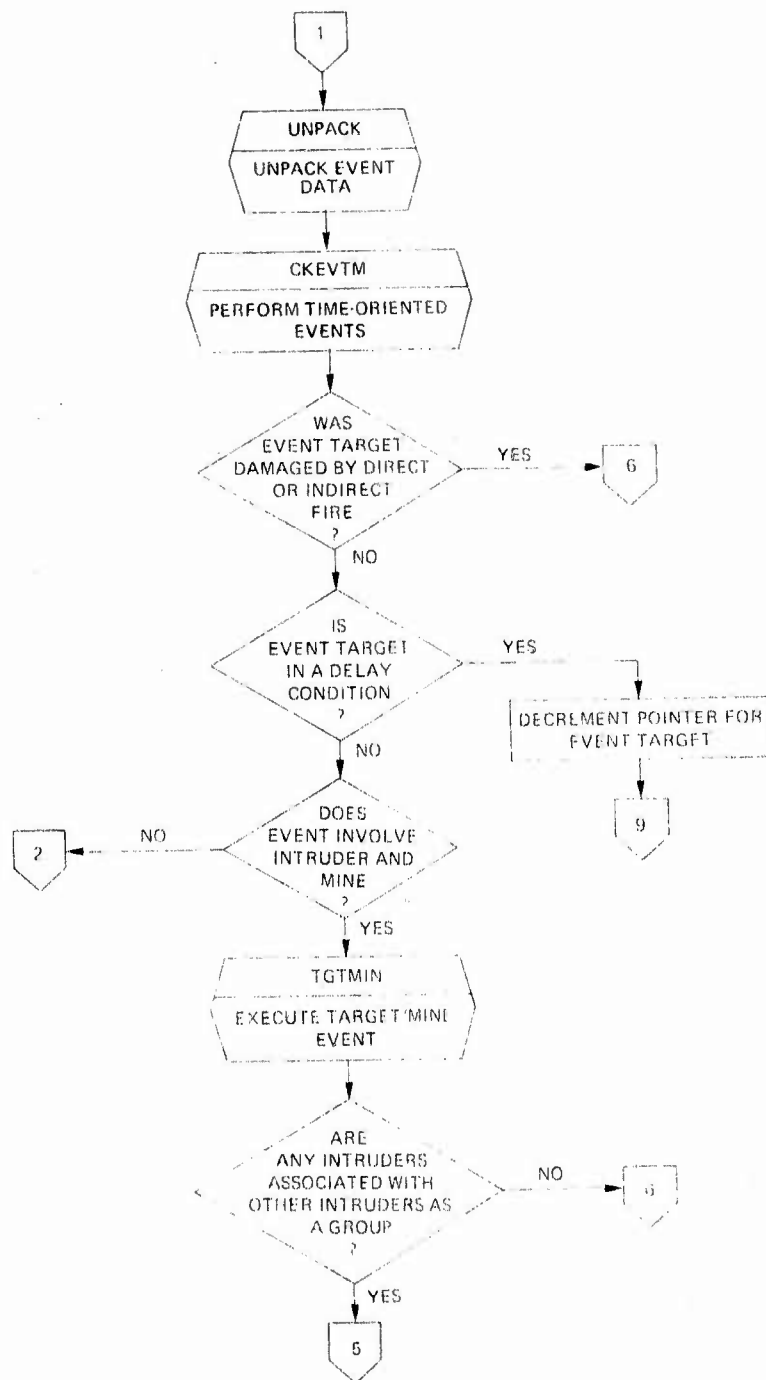


Figure 4. (Continued)

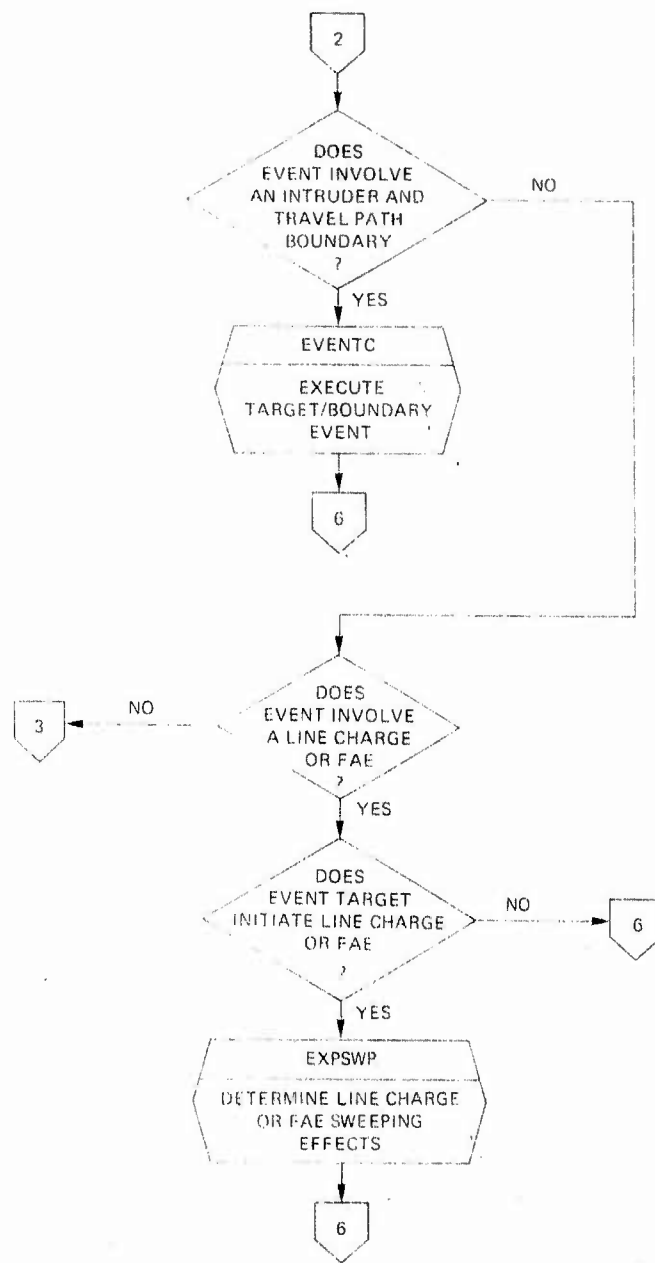


Figure 4. (Continued).

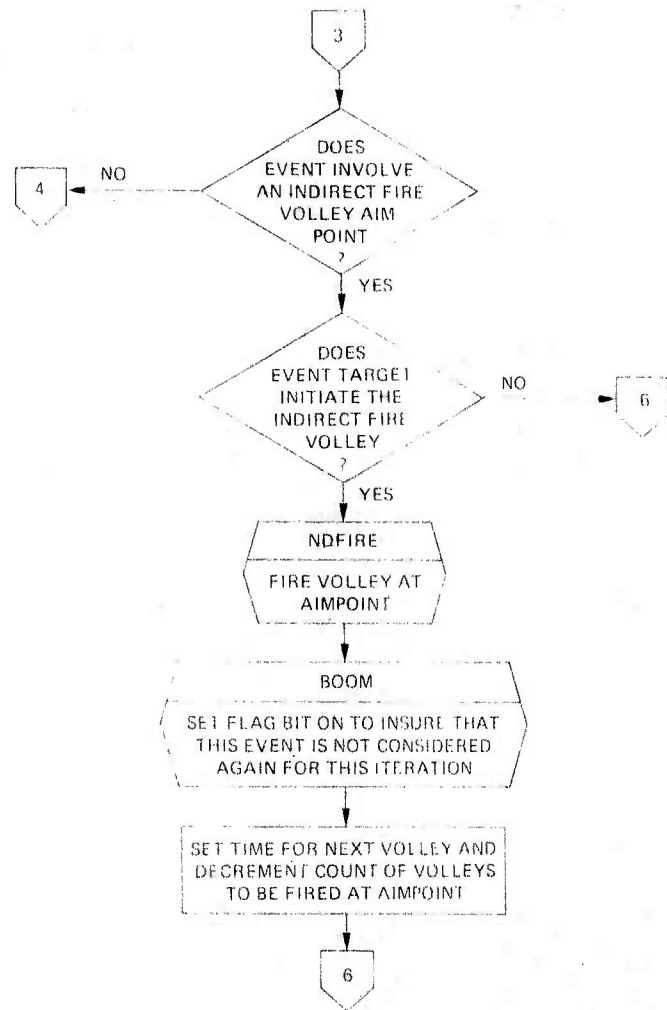


Figure 4. (Continued)

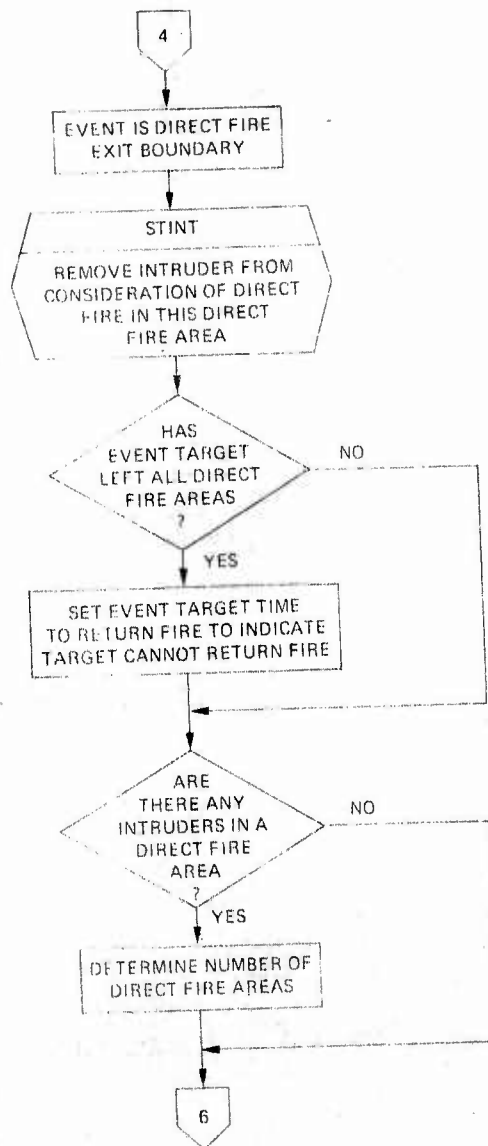


Figure 4. (Continued)

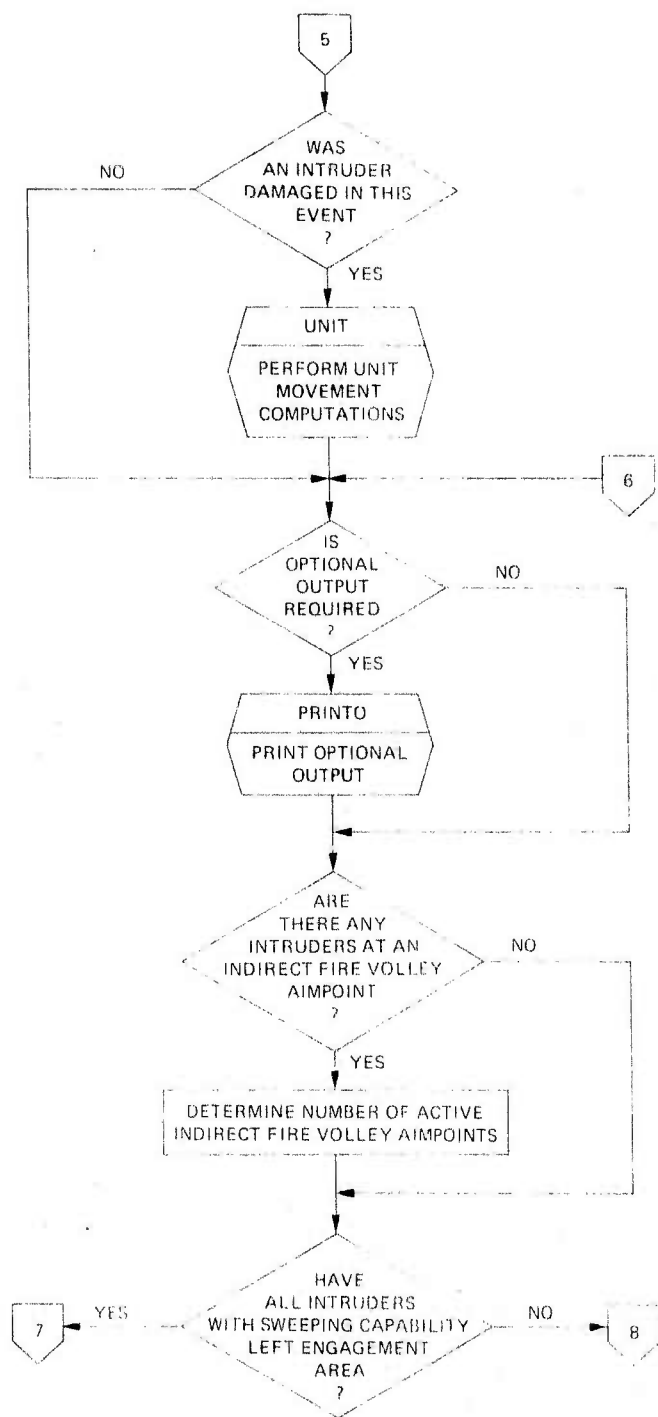


Figure 4. (Continued)

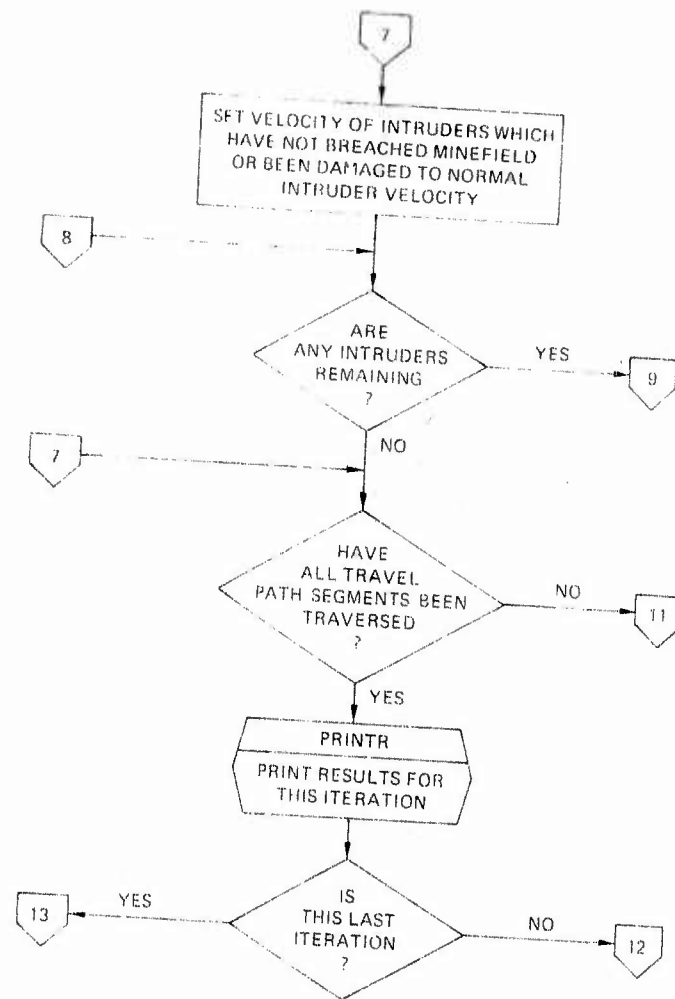


Figure 4. (Concluded)

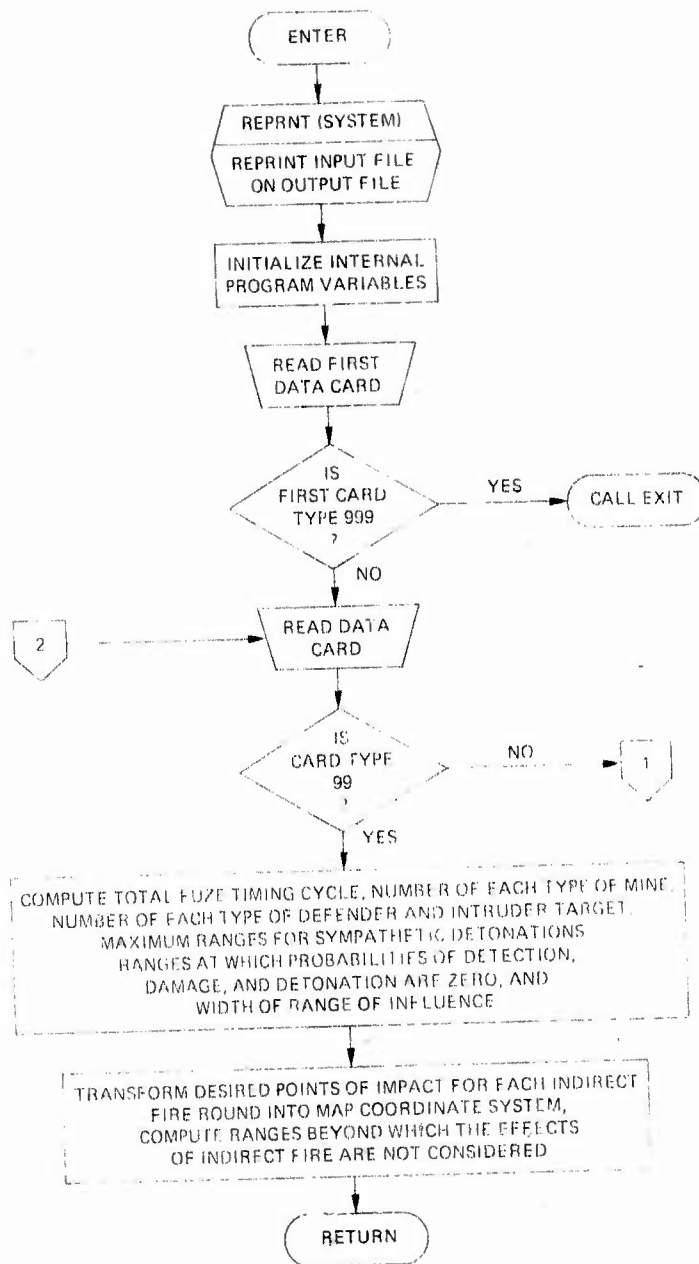


Figure 5. Flowchart, Subroutine READIN

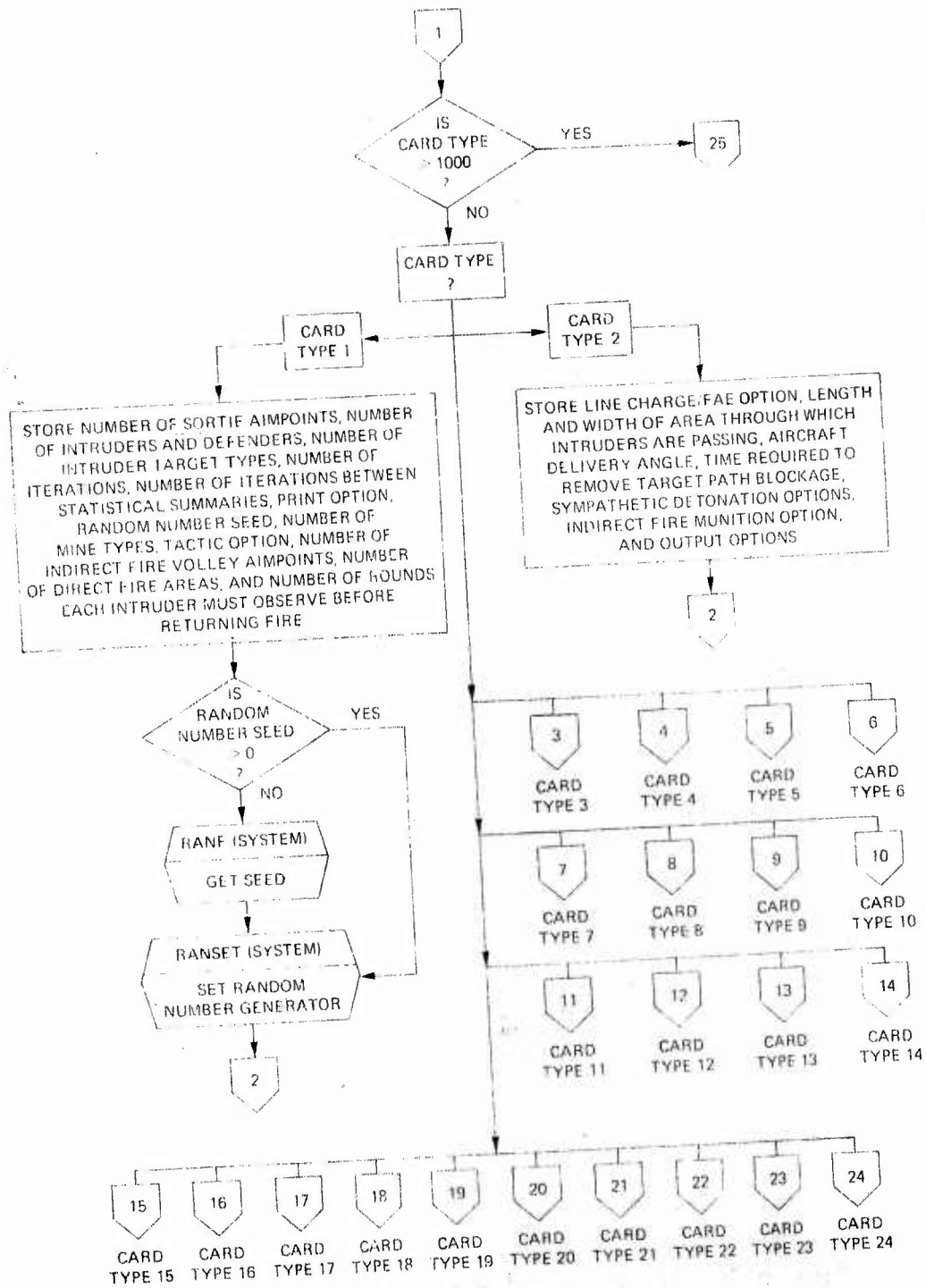


Figure 5. (Continued)

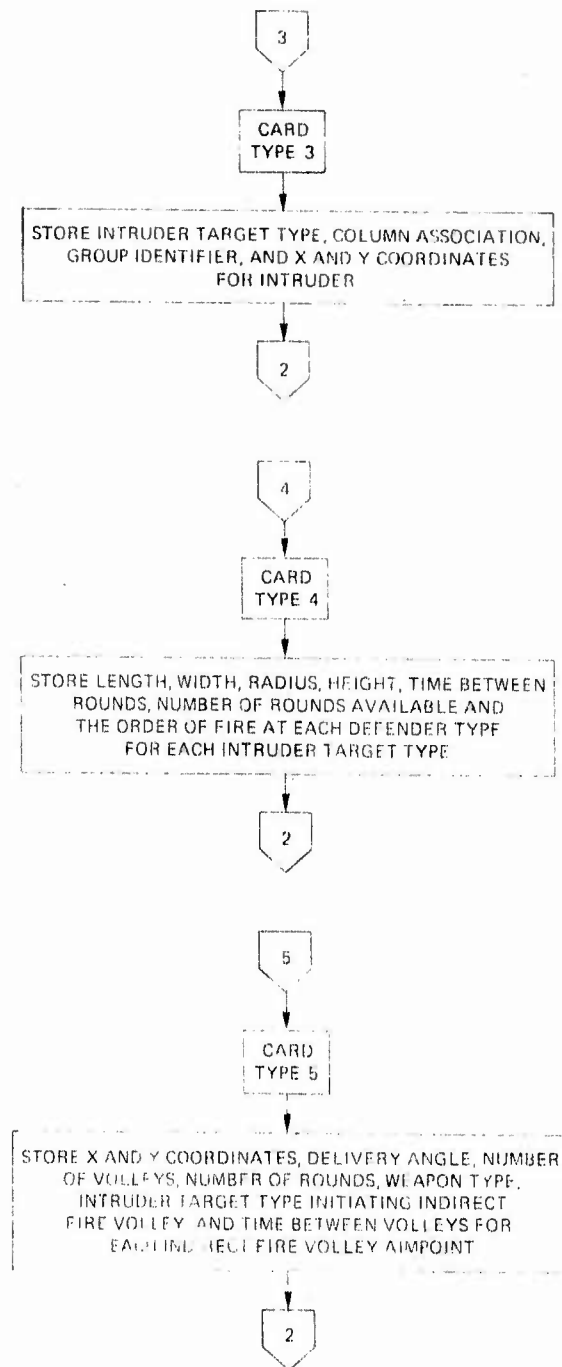


Figure 5. (Continued)

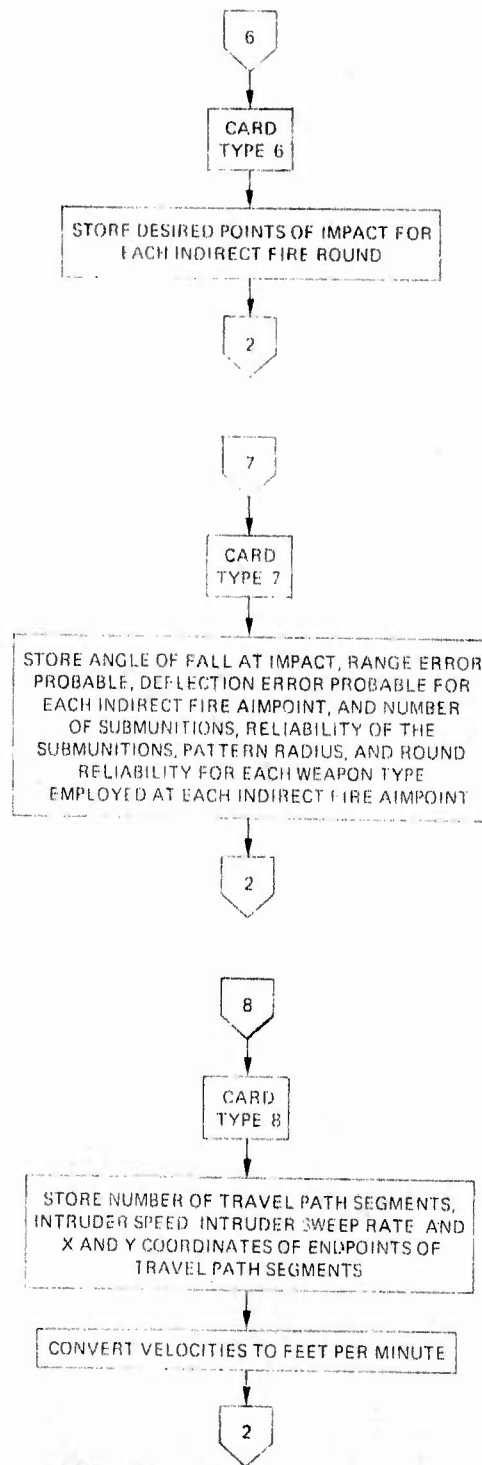


Figure 5. (Continued)

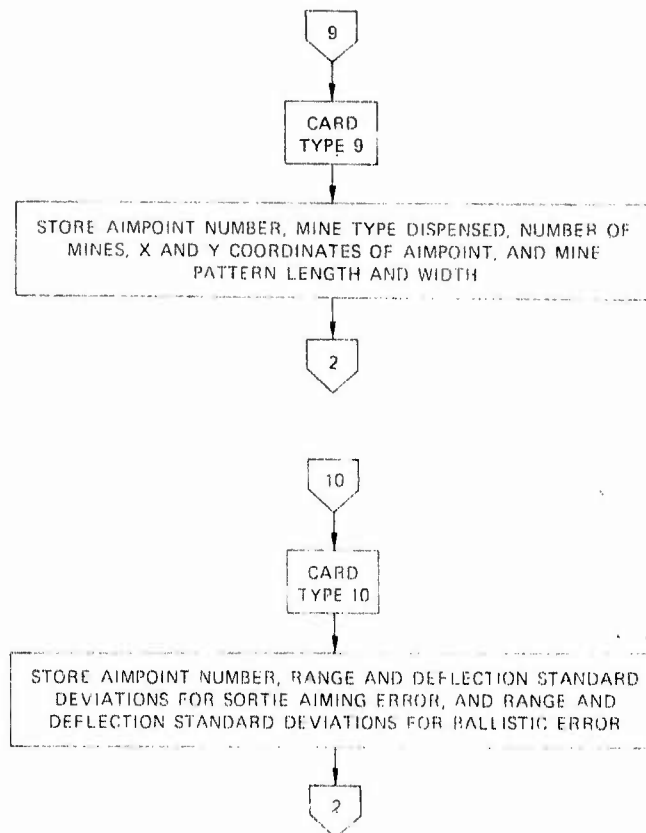


Figure 5. (Continued)

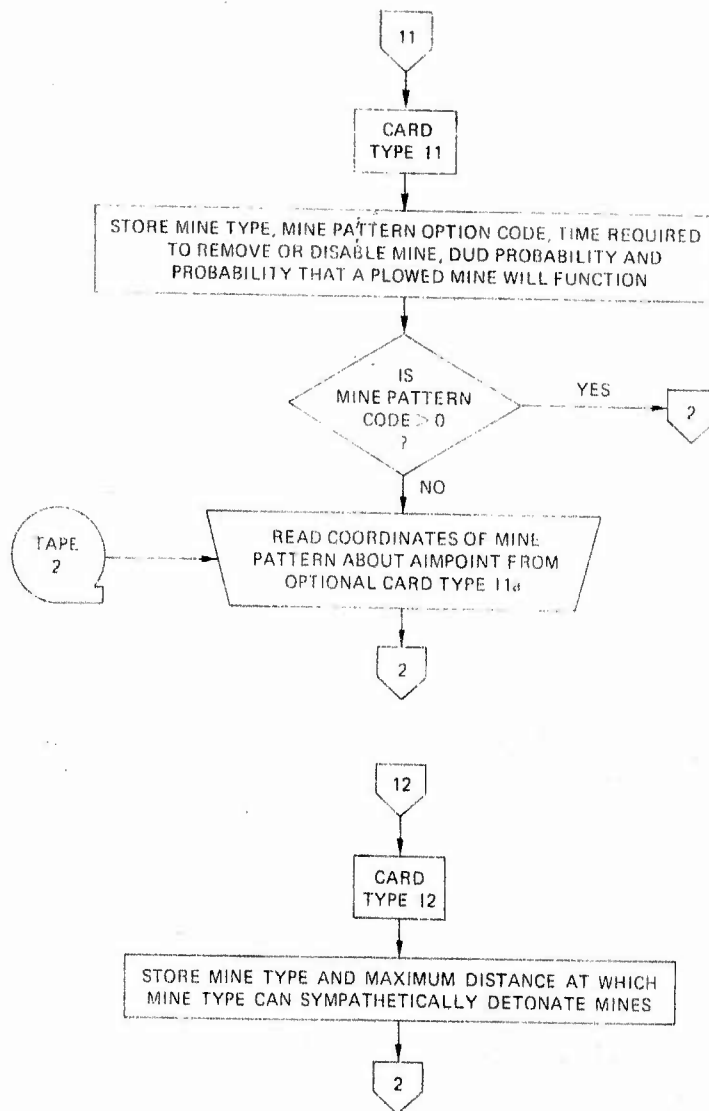


Figure 5. (Continued)

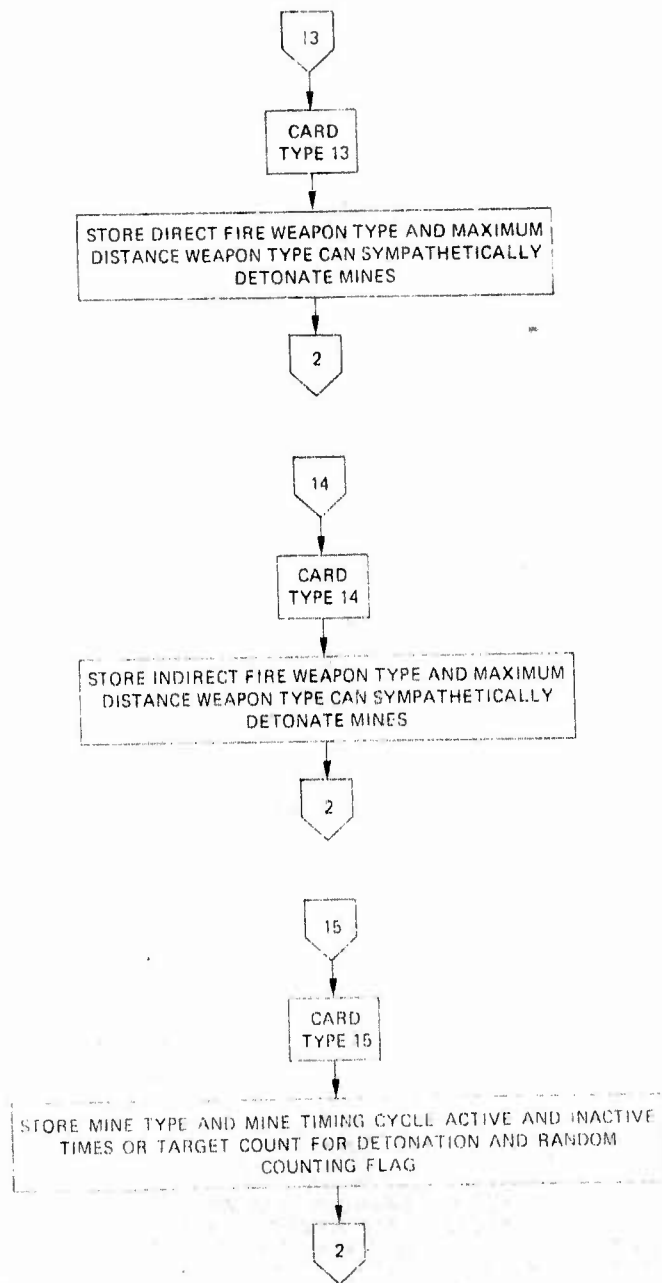


Figure 5. (Continued)

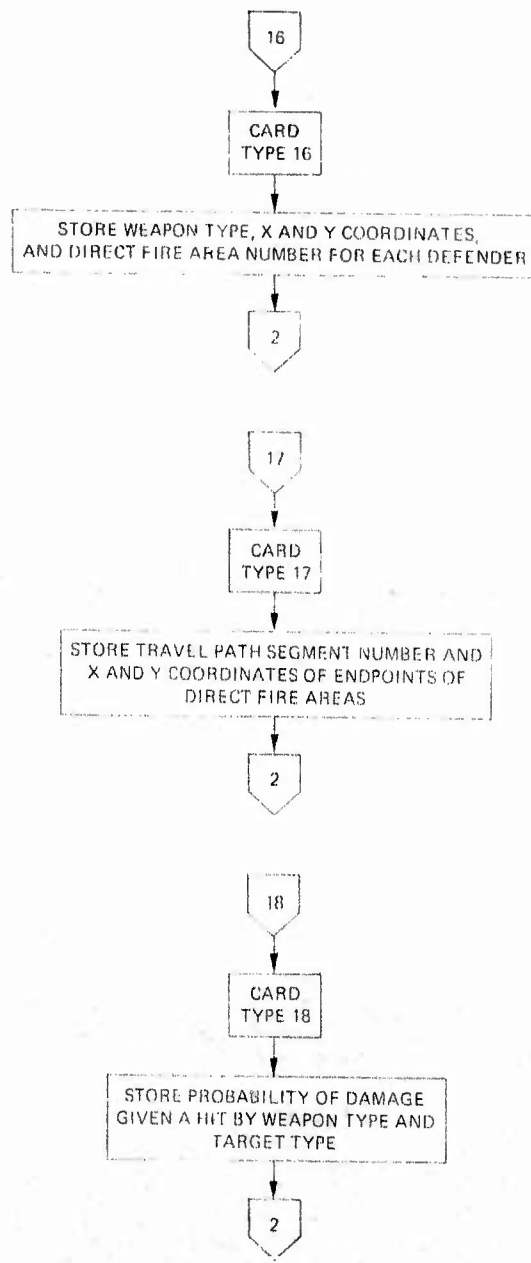


Figure 5. (Continued)

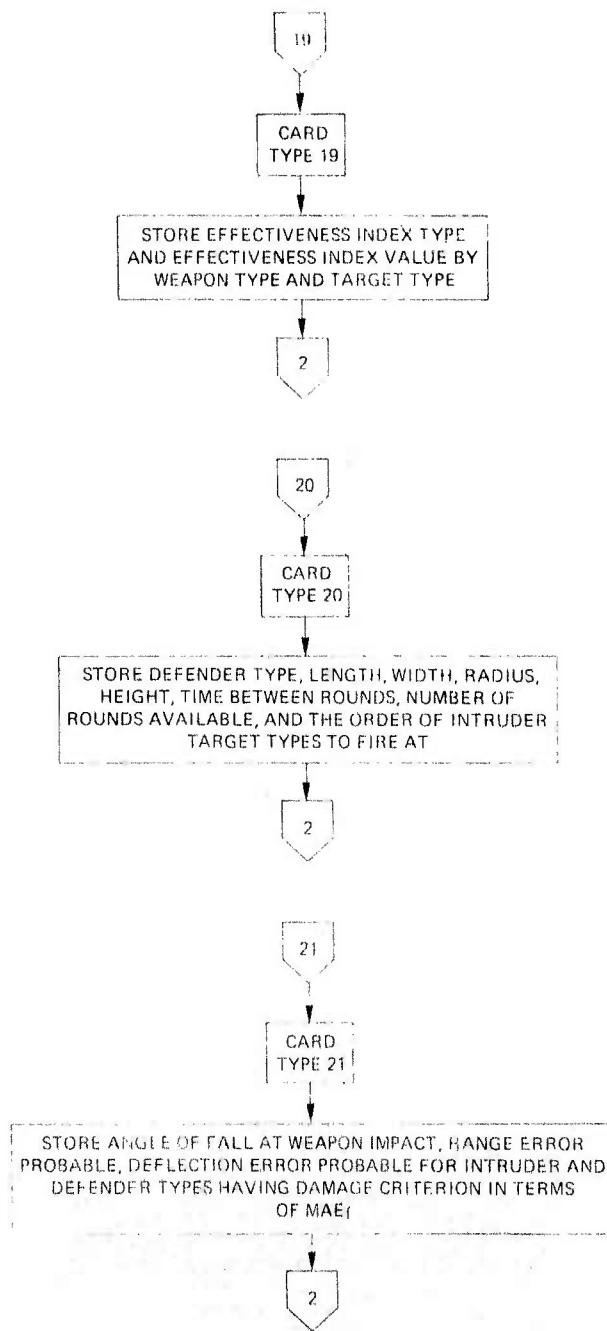


Figure 5. (Continued)

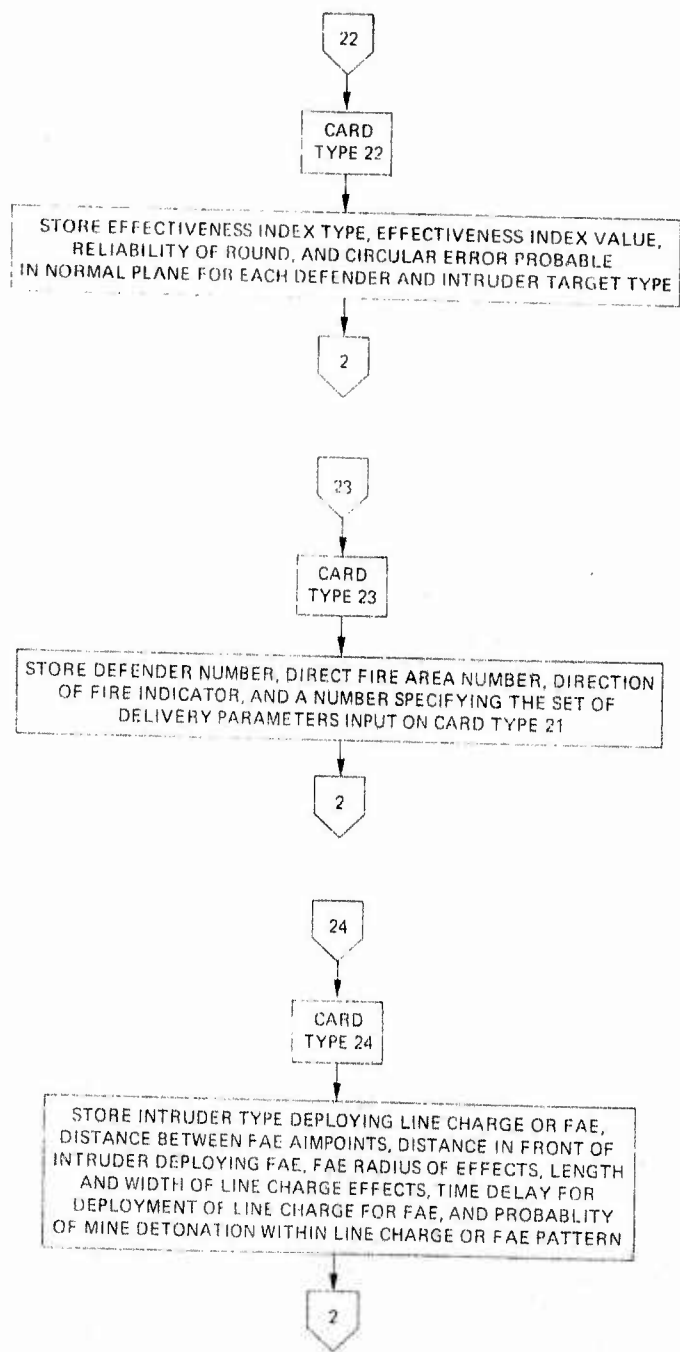


Figure 5. (Continued)

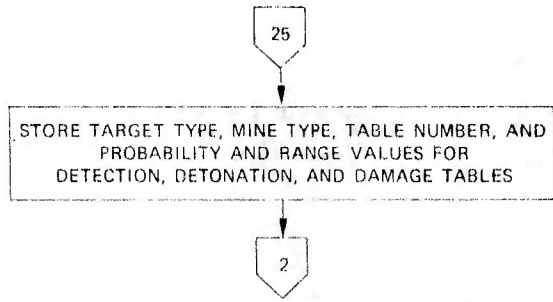


Figure 5. (Concluded)

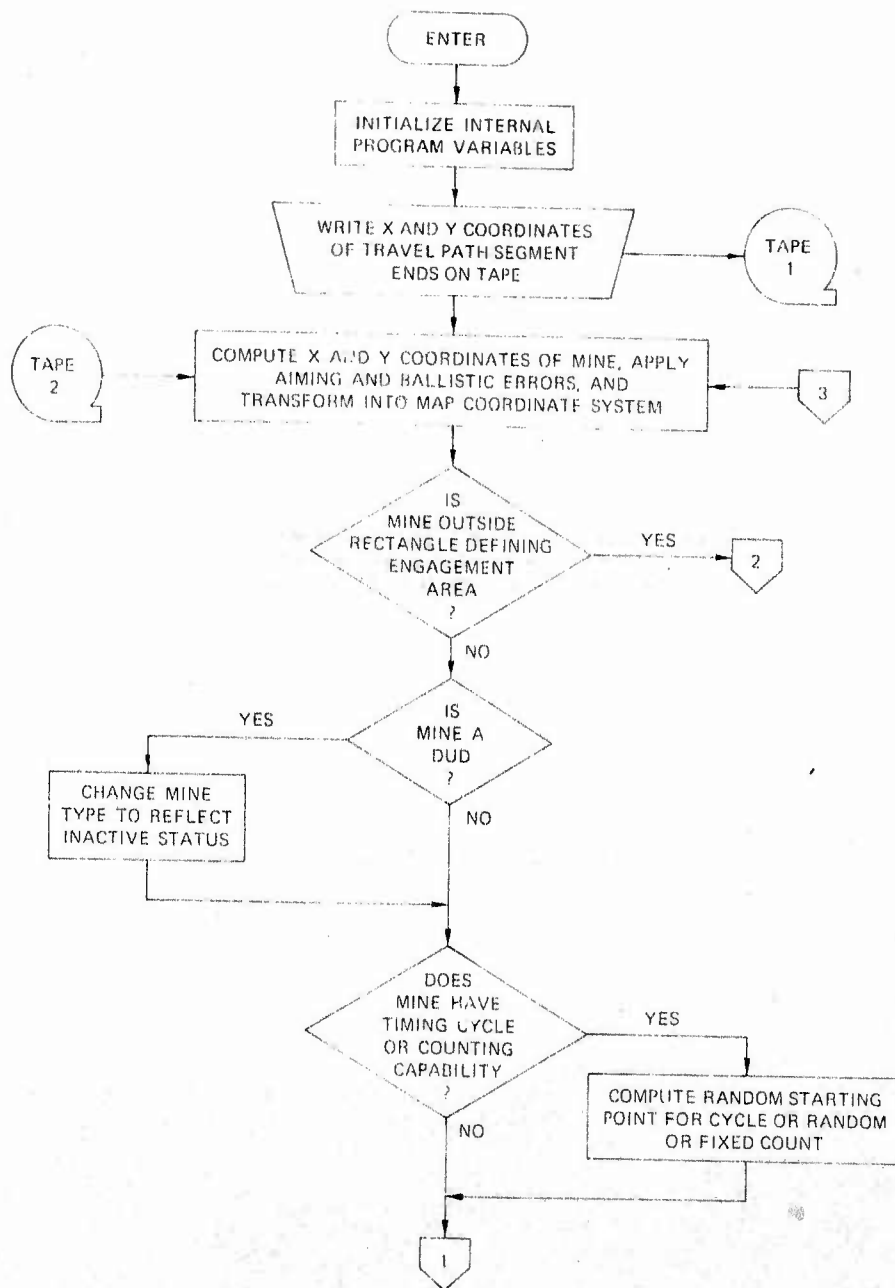


Figure 6. Flowchart, Subroutine SETUP

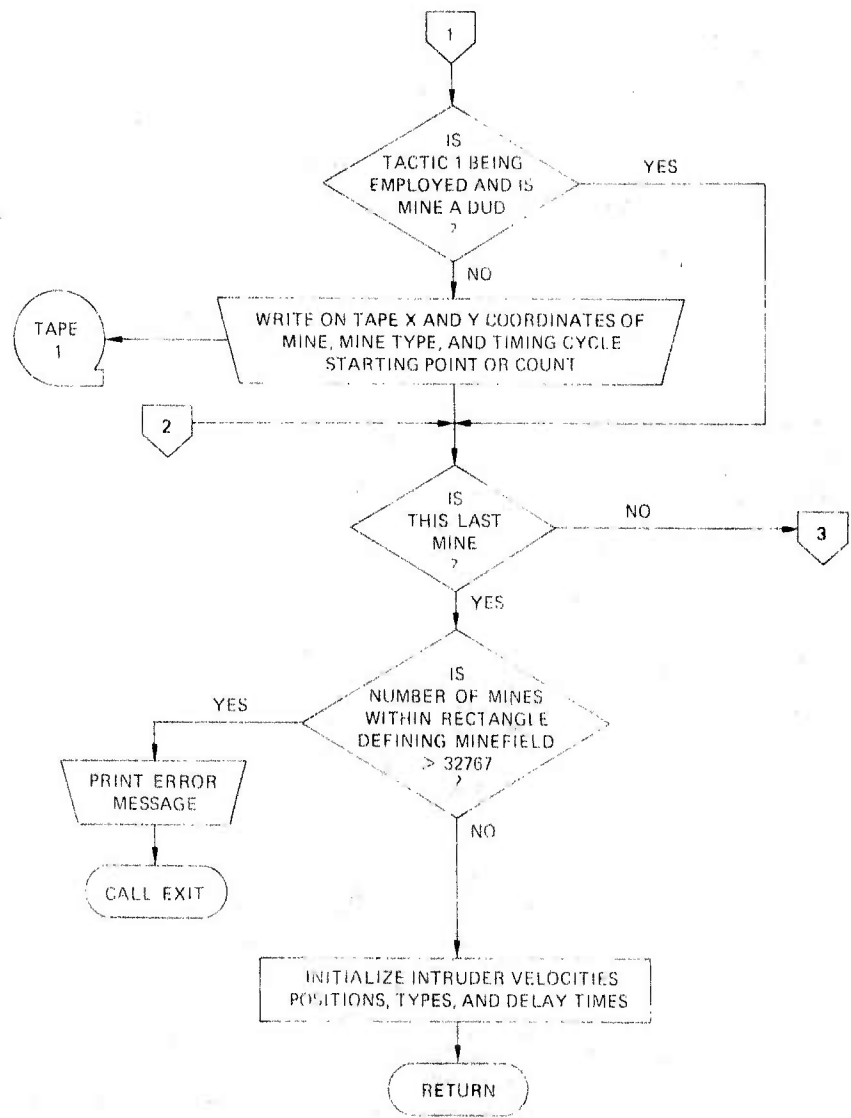


Figure 6. (Concluded)

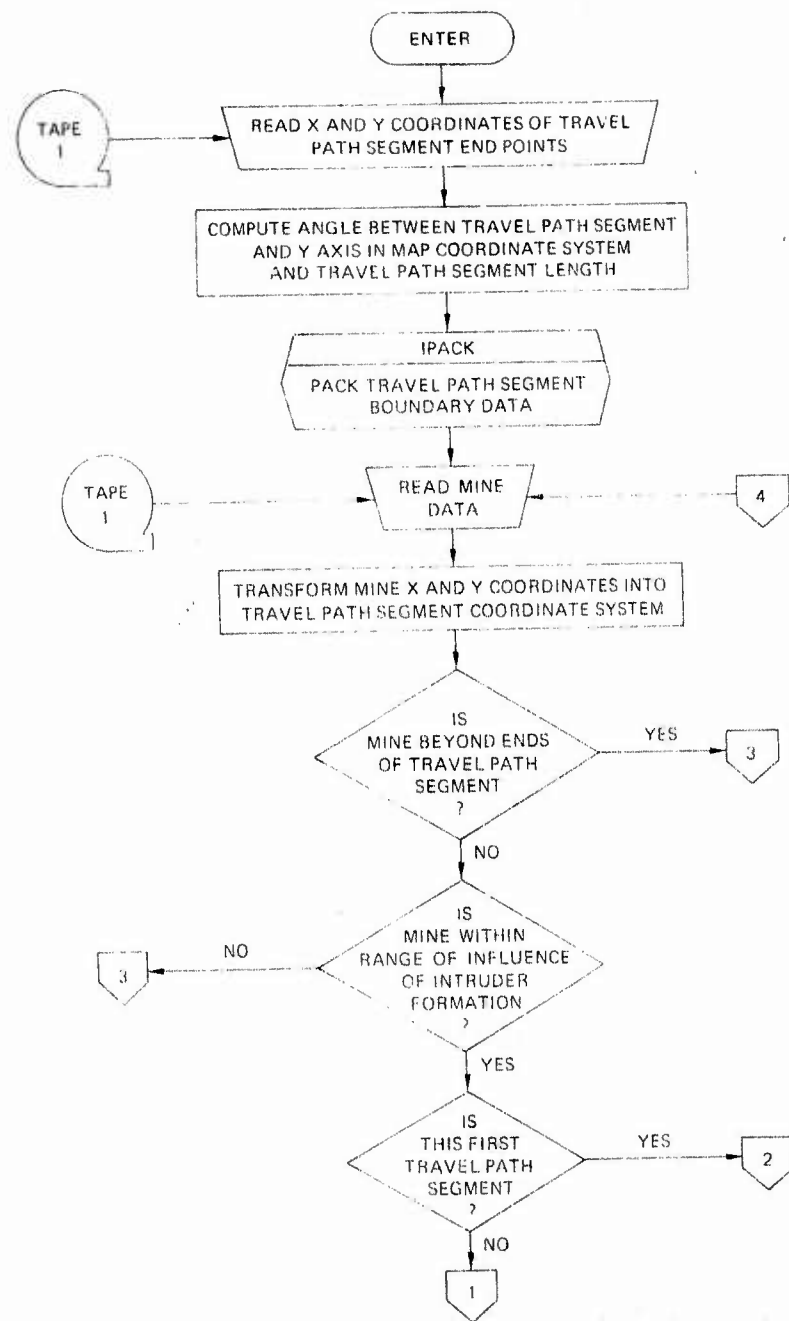


Figure 7. Flowchart, Subroutine ROAD

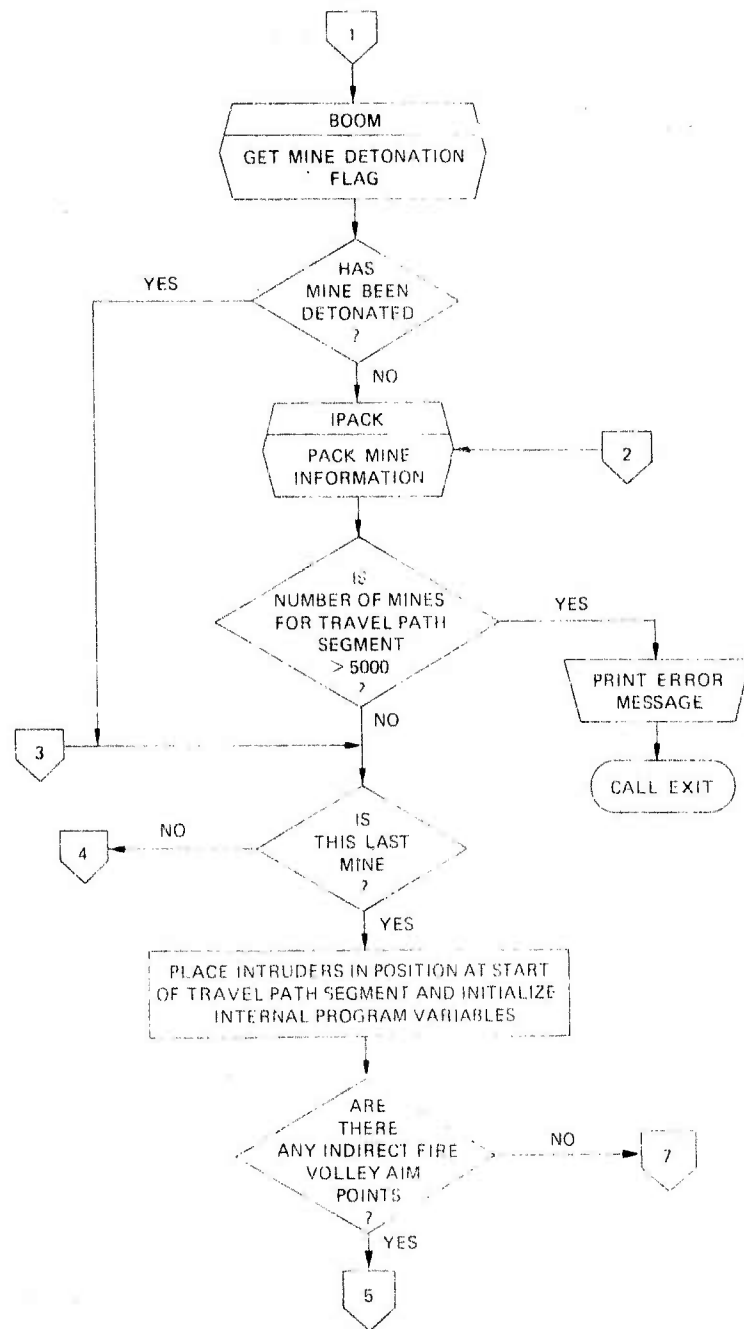


Figure 7. (Continued)

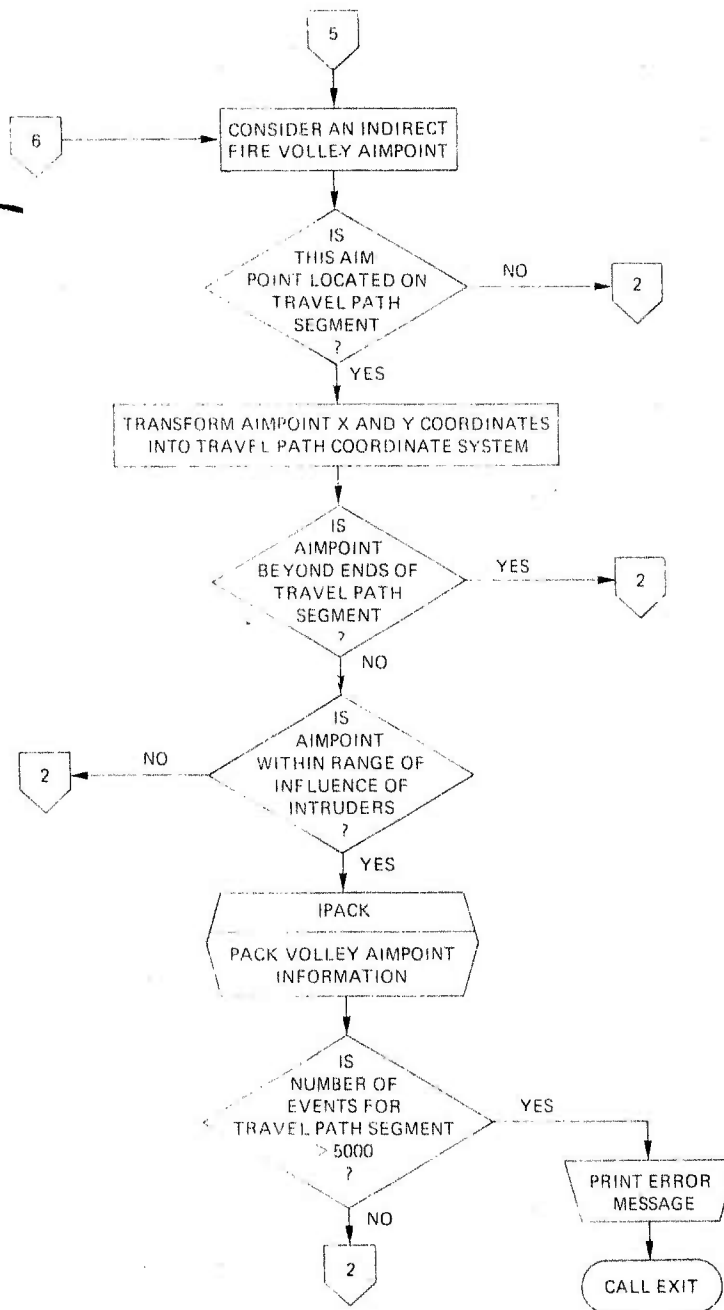


Figure 7. (Continued)

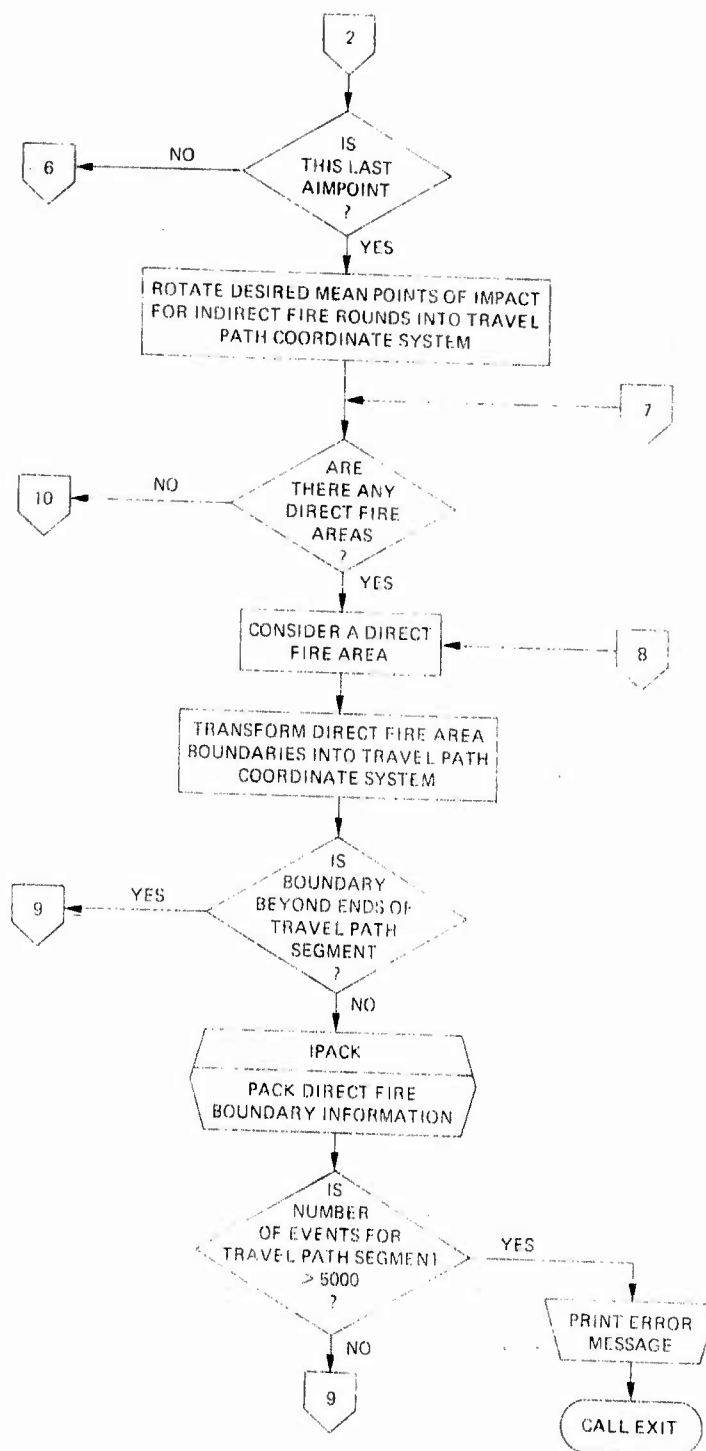


Figure 7. (Continued)

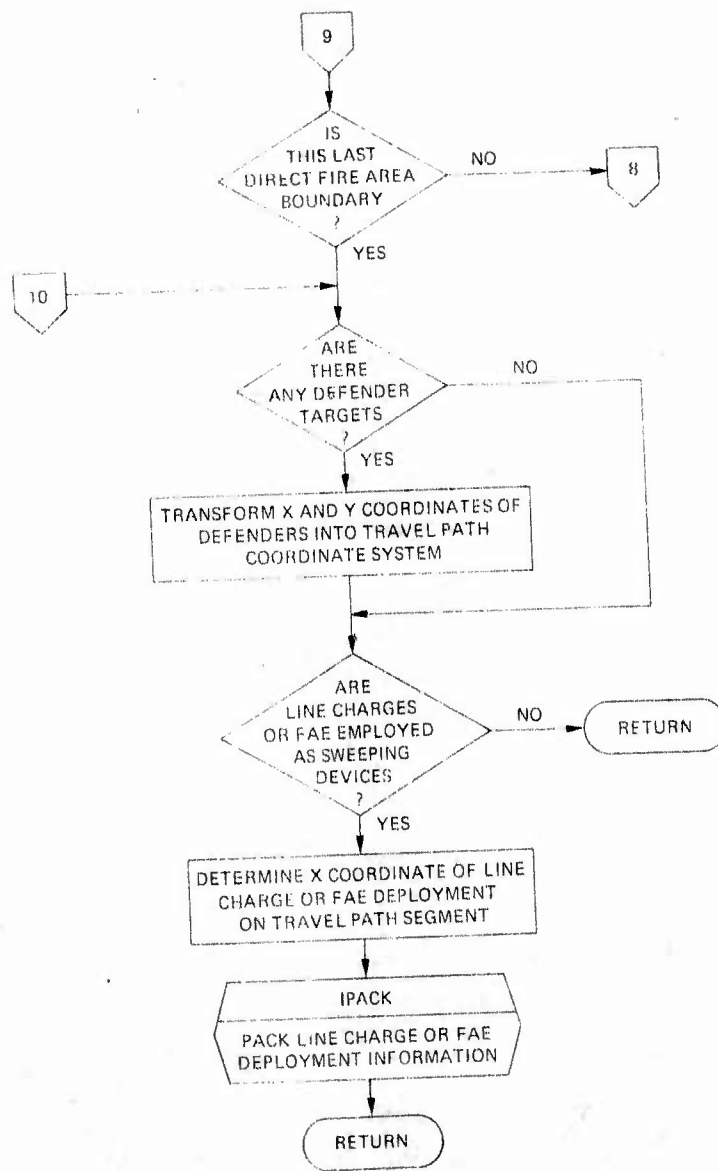


Figure 7. (Concluded)

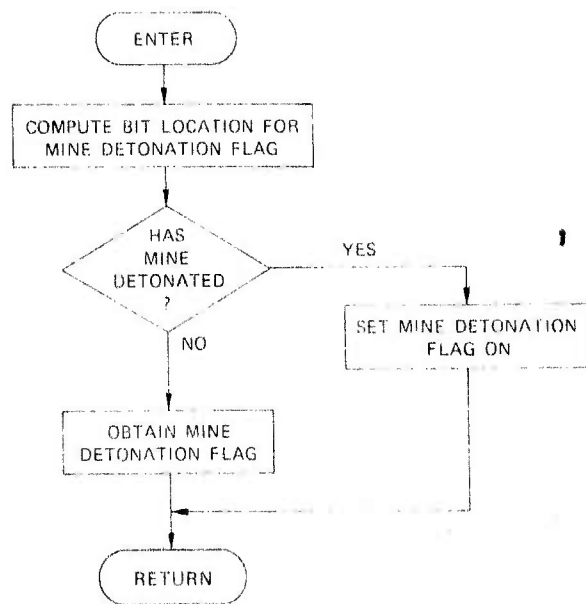


Figure 8. Flowchart, Subroutine BOOM

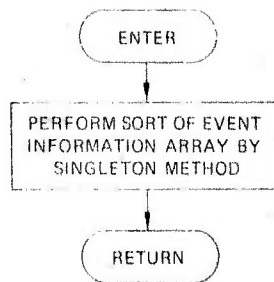


Figure 9. Flowchart, Subroutine SORT

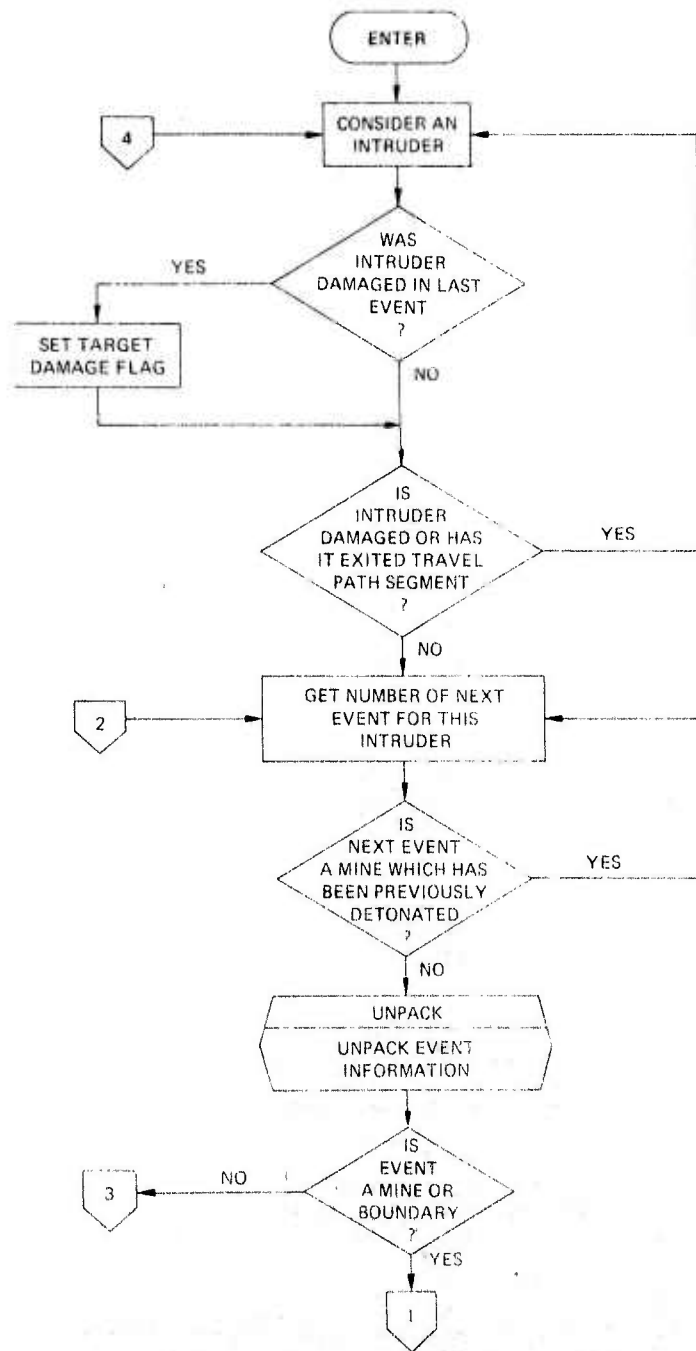


Figure 10. Flowchart, Subroutine LOOPS

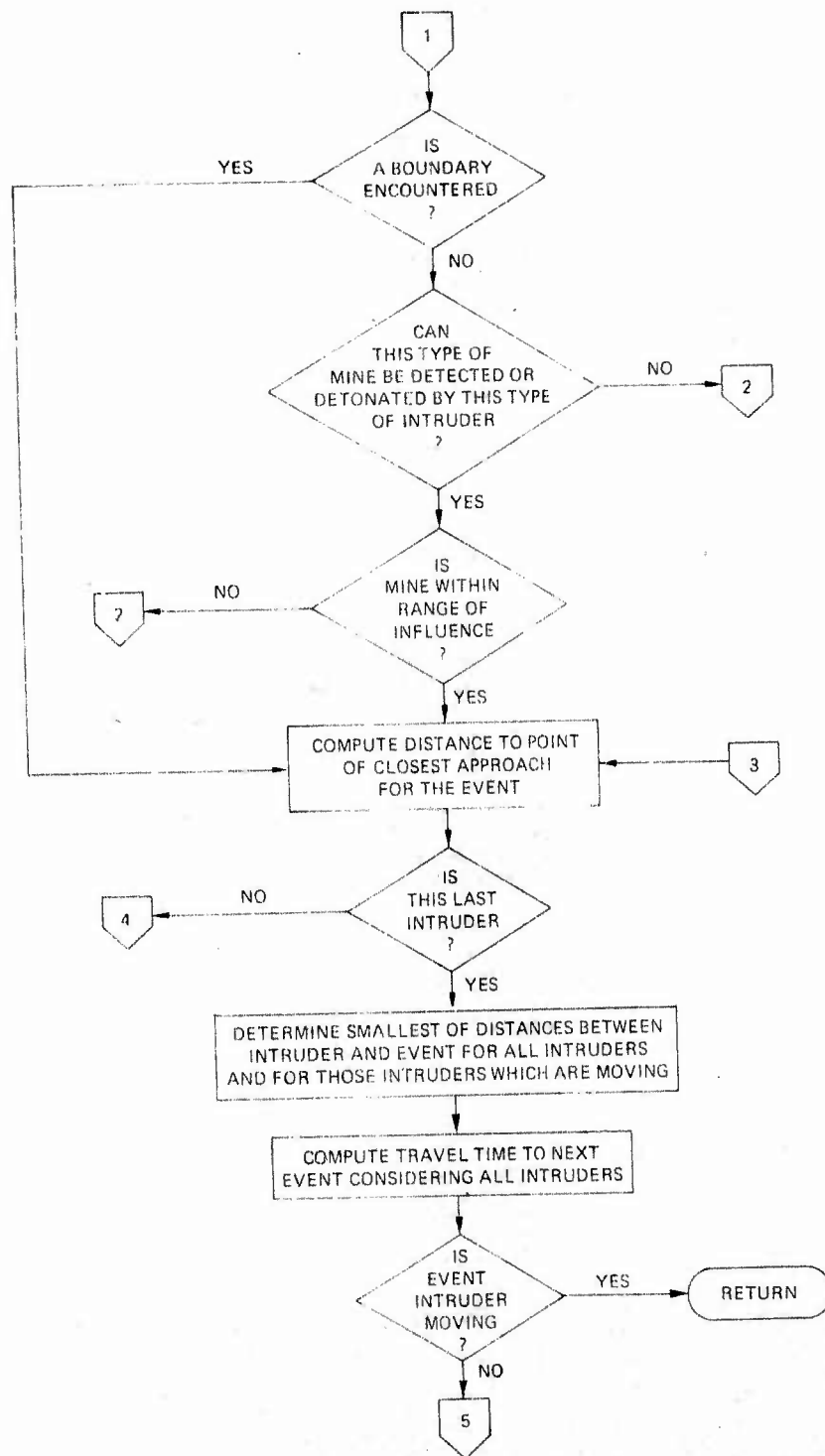


Figure 10. (Continued)

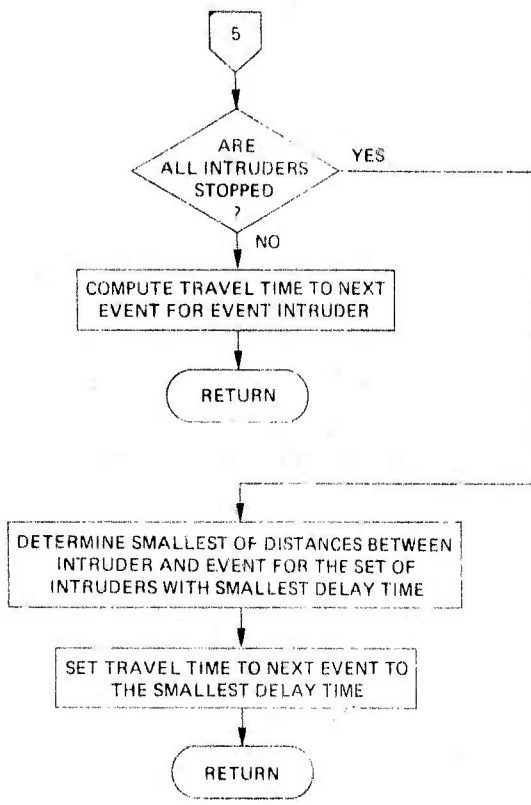


Figure 10. (Concluded)

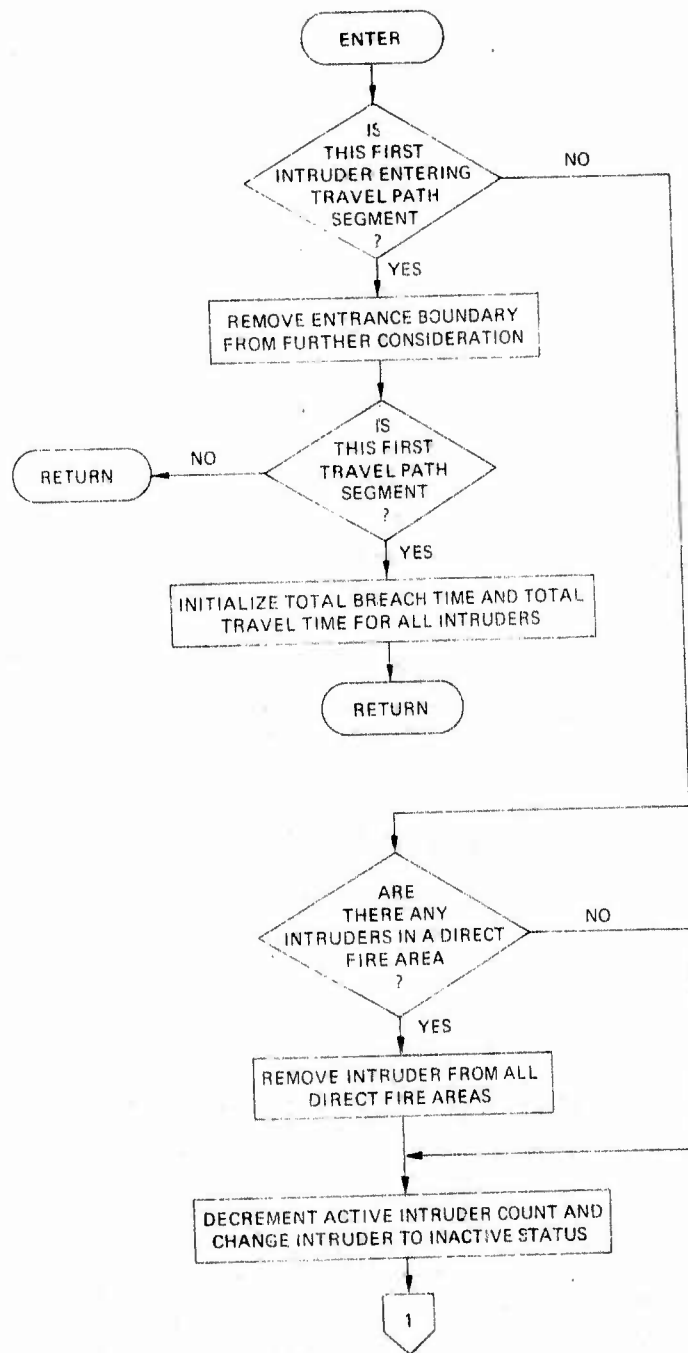


Figure 11. Flowchart, Subroutine EVENTC

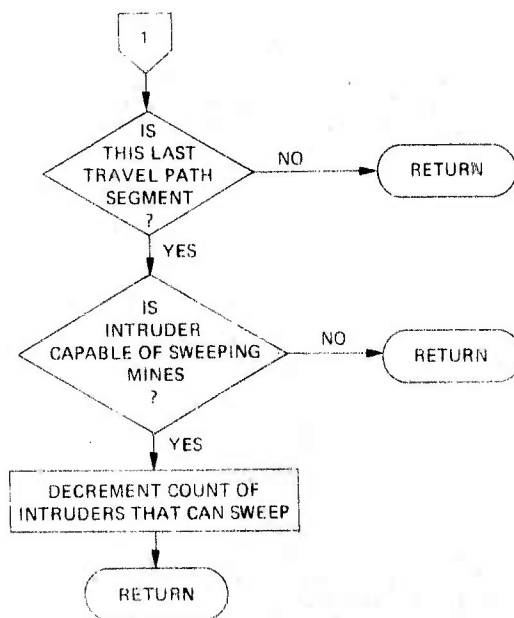


Figure 11. (Concluded)

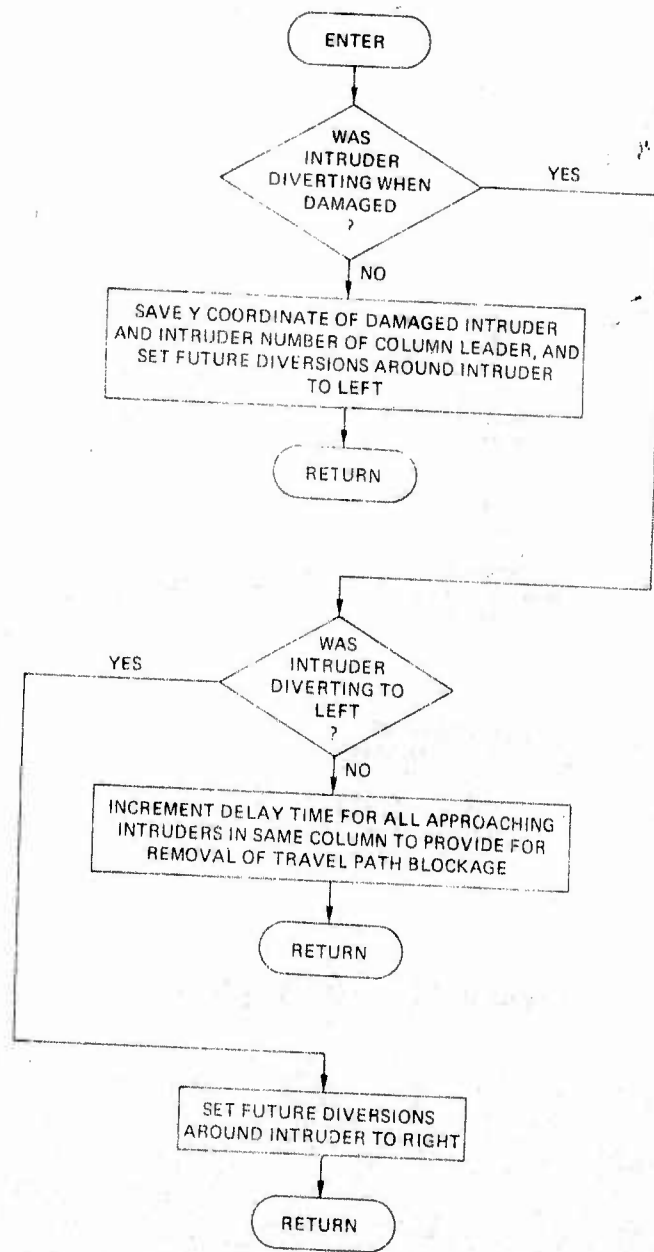


Figure 12. Flowchart, Subroutine DIVSET

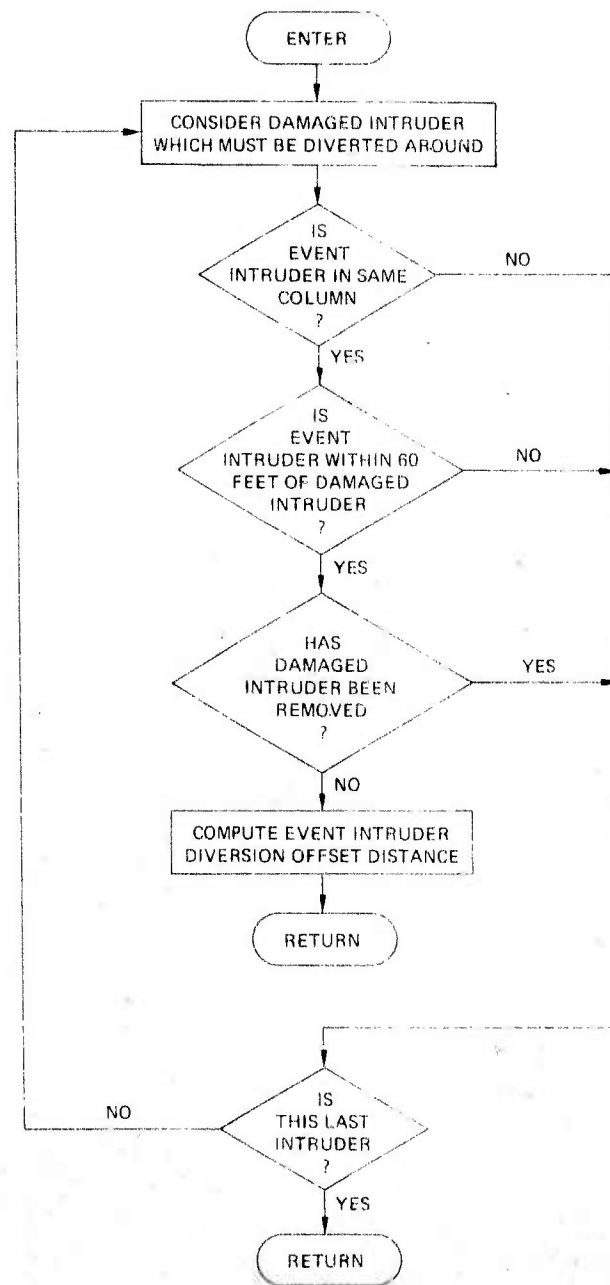


Figure 13. Flowchart, Subroutine DIVCHK

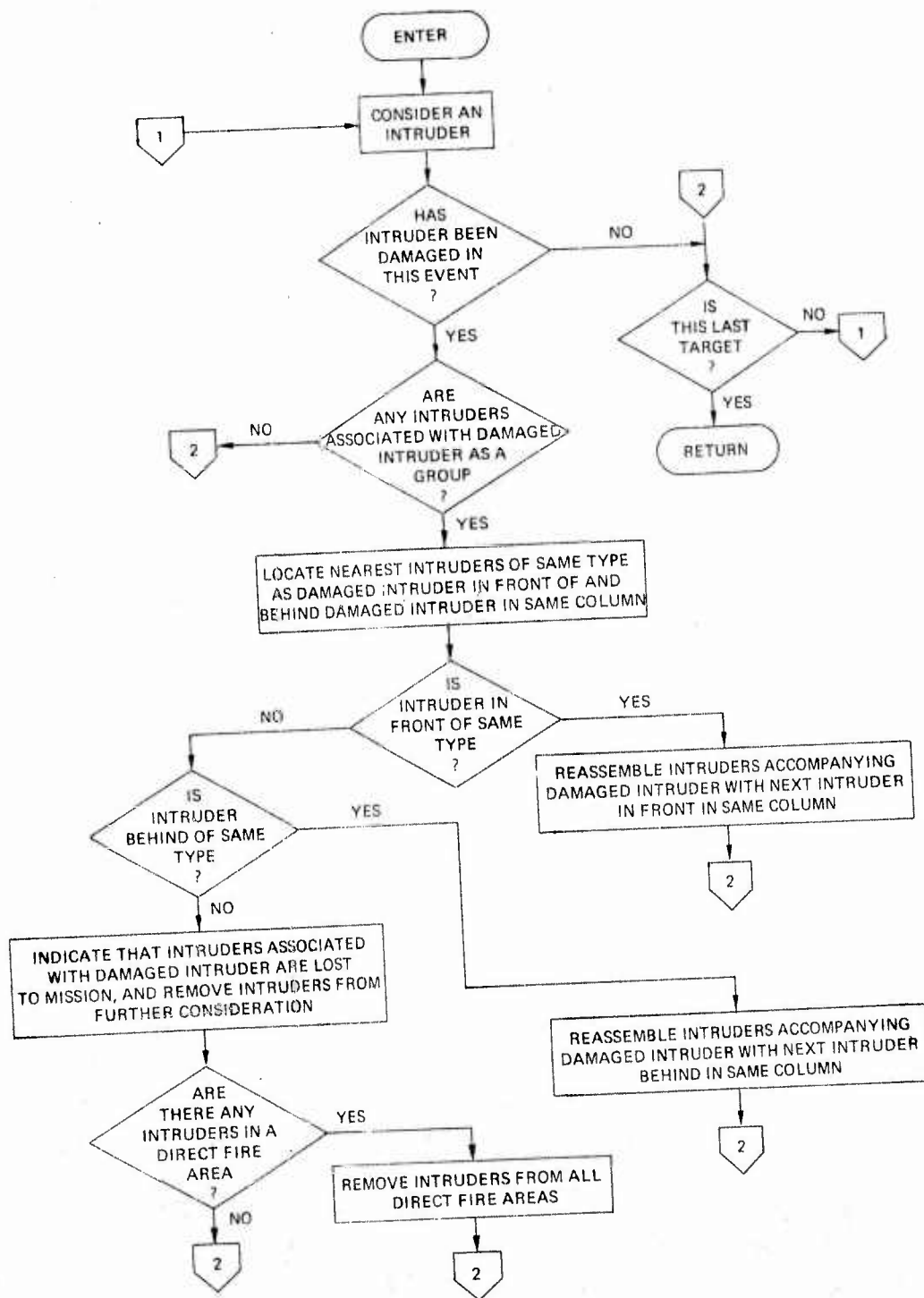


Figure 14. Flowchart, Subroutine UNIT

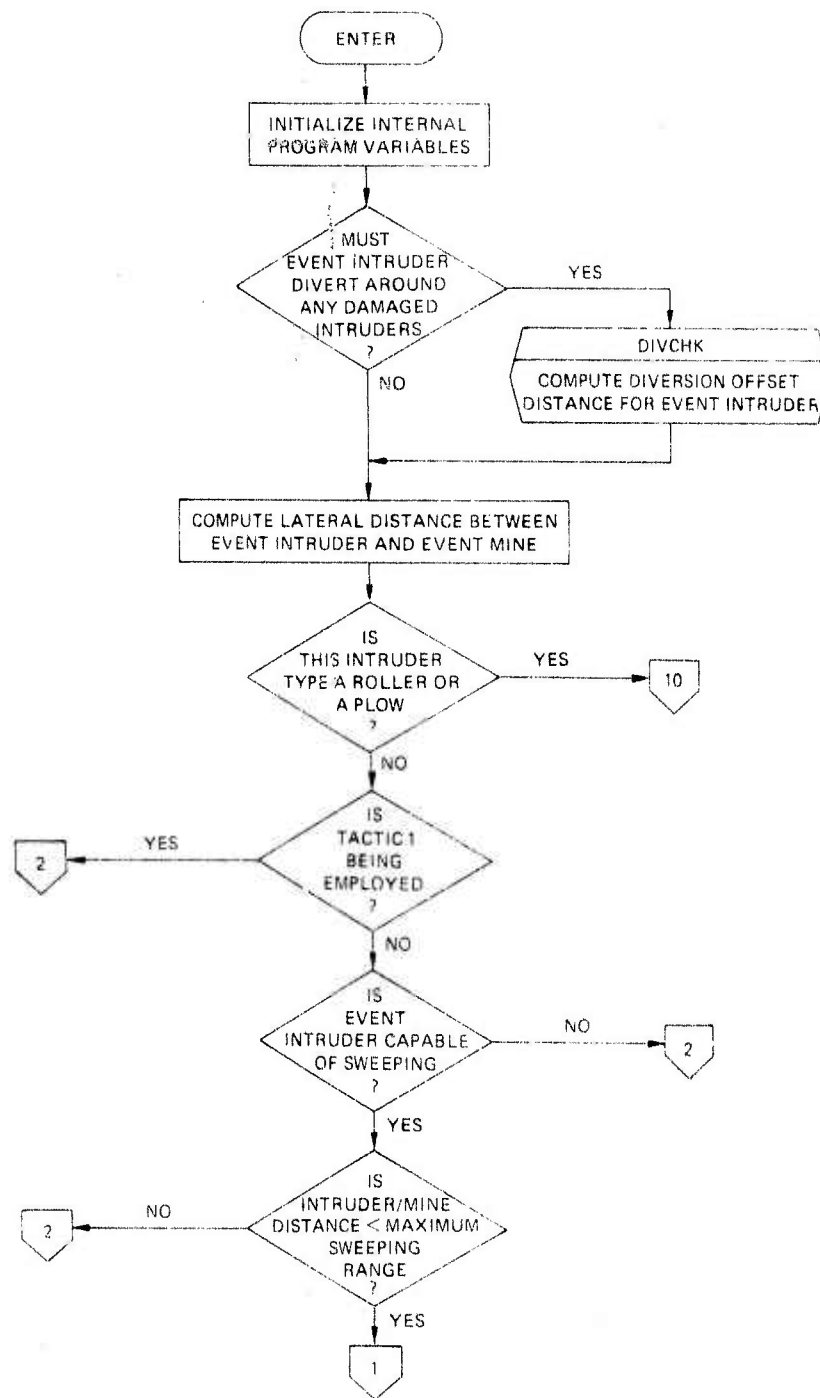


Figure 15. Flowchart, Subroutine TGTMIN

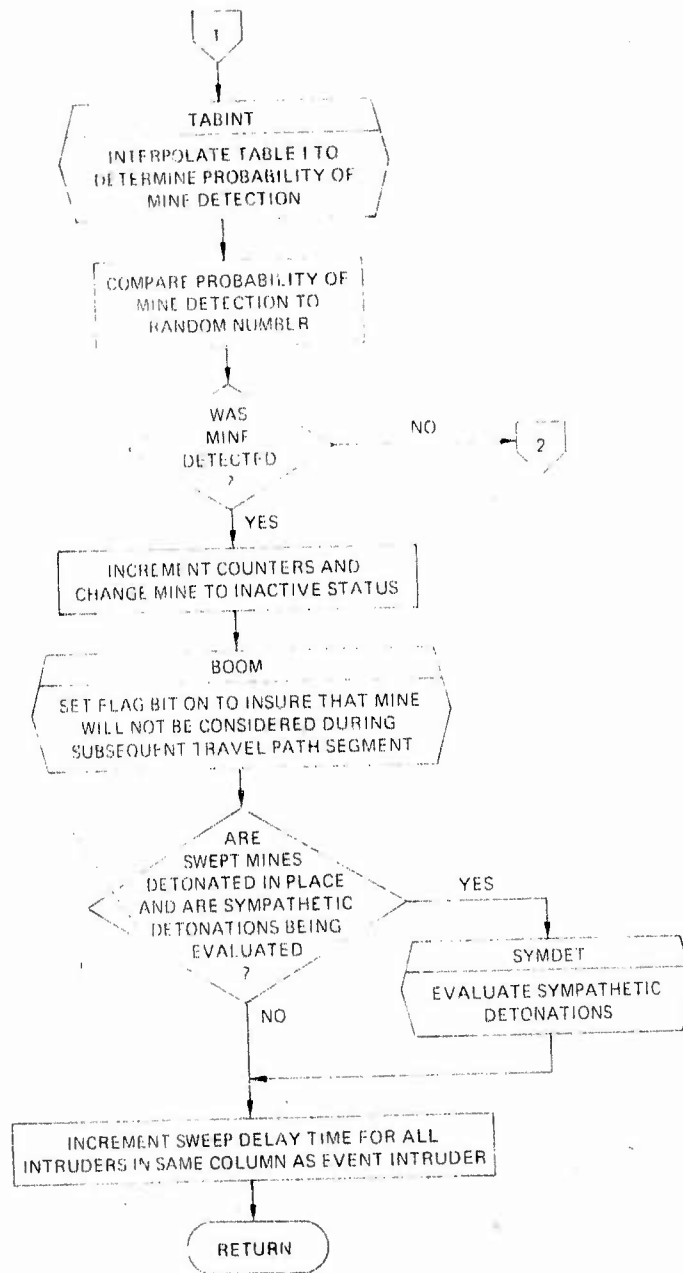


Figure 15. (Continued)

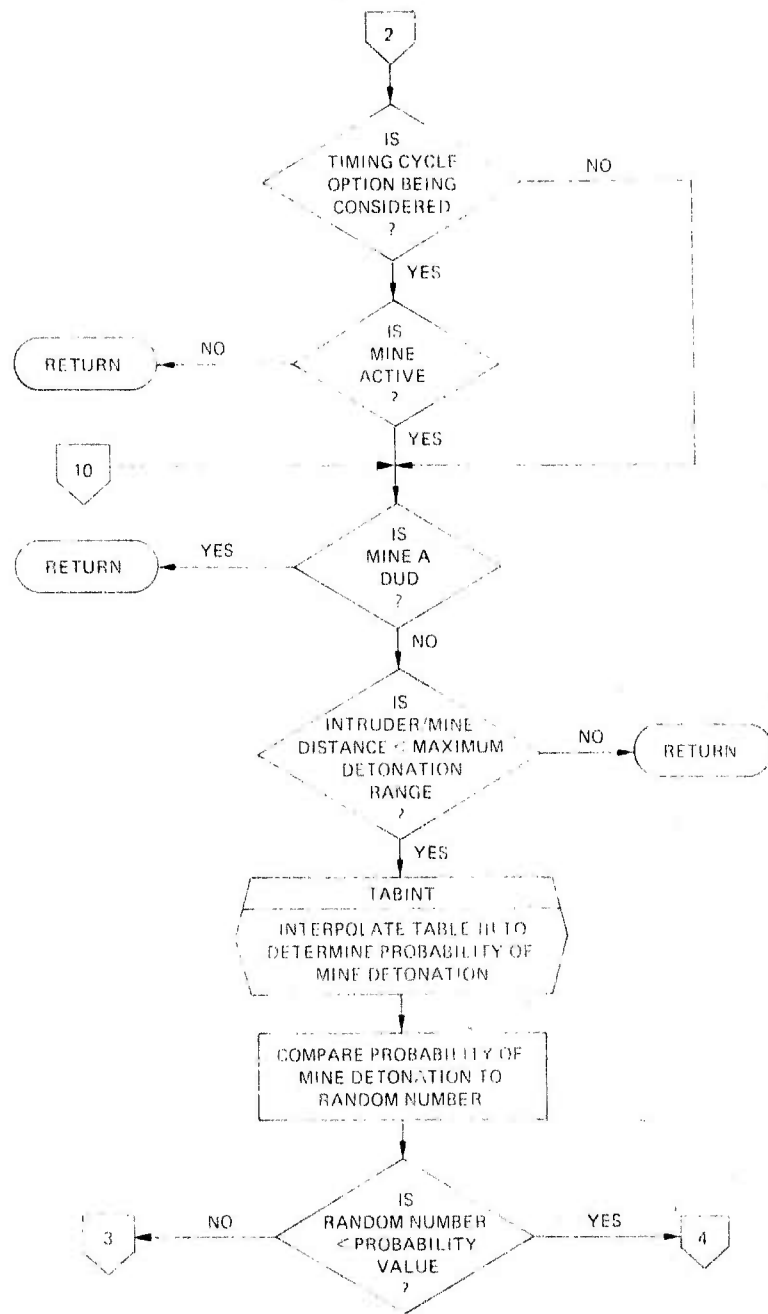


Figure 15. (Continued)

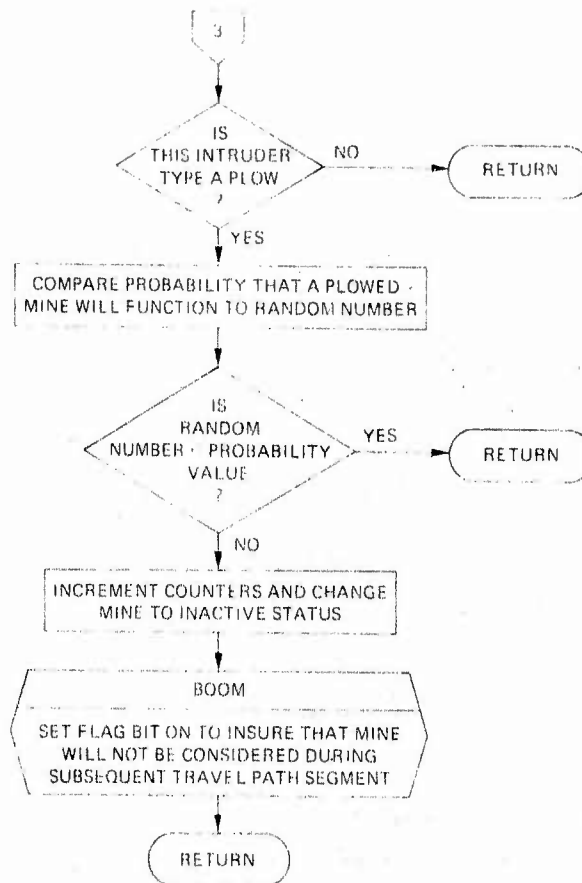


Figure 15. (Continued)

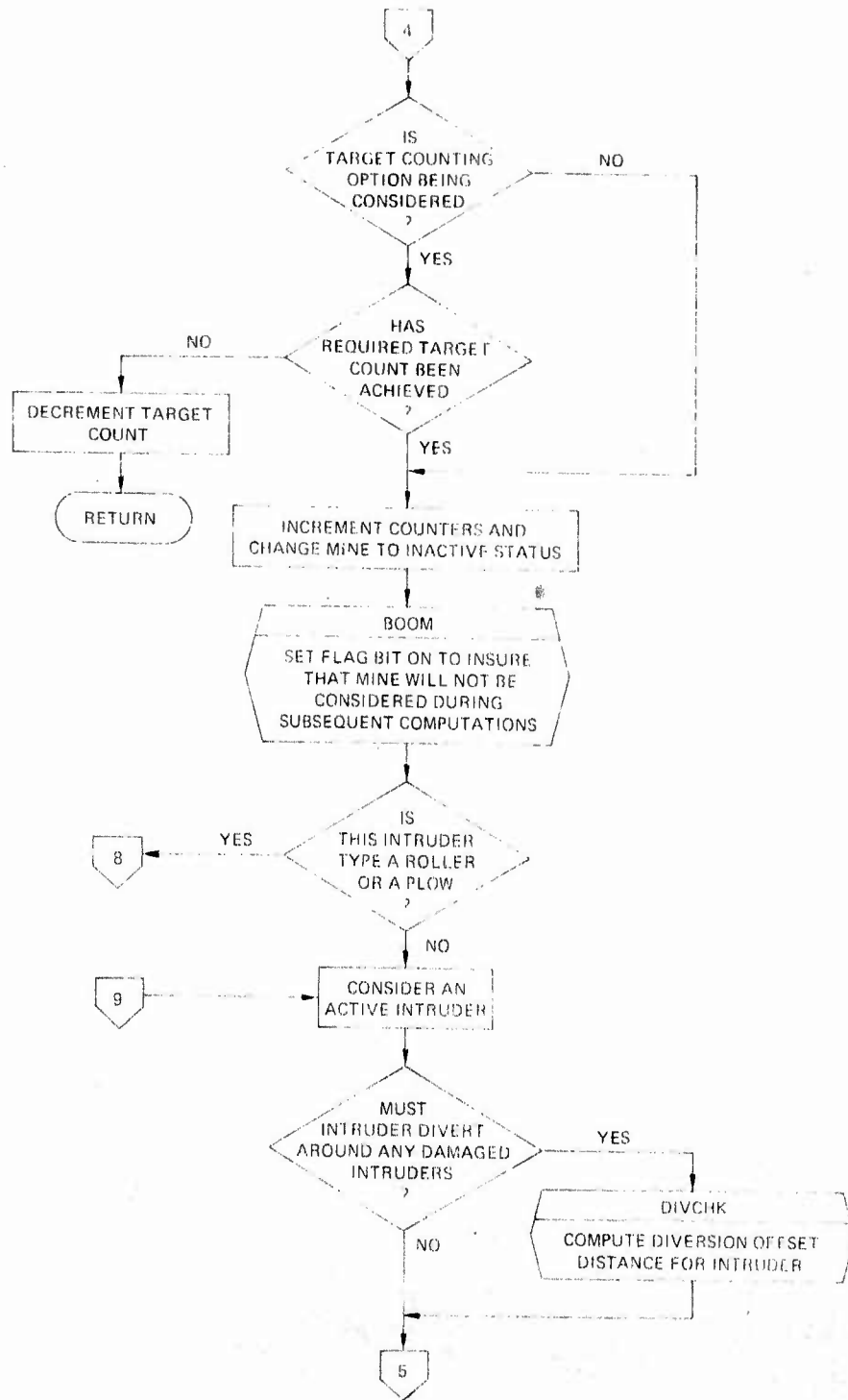


Figure 15. (Continued)

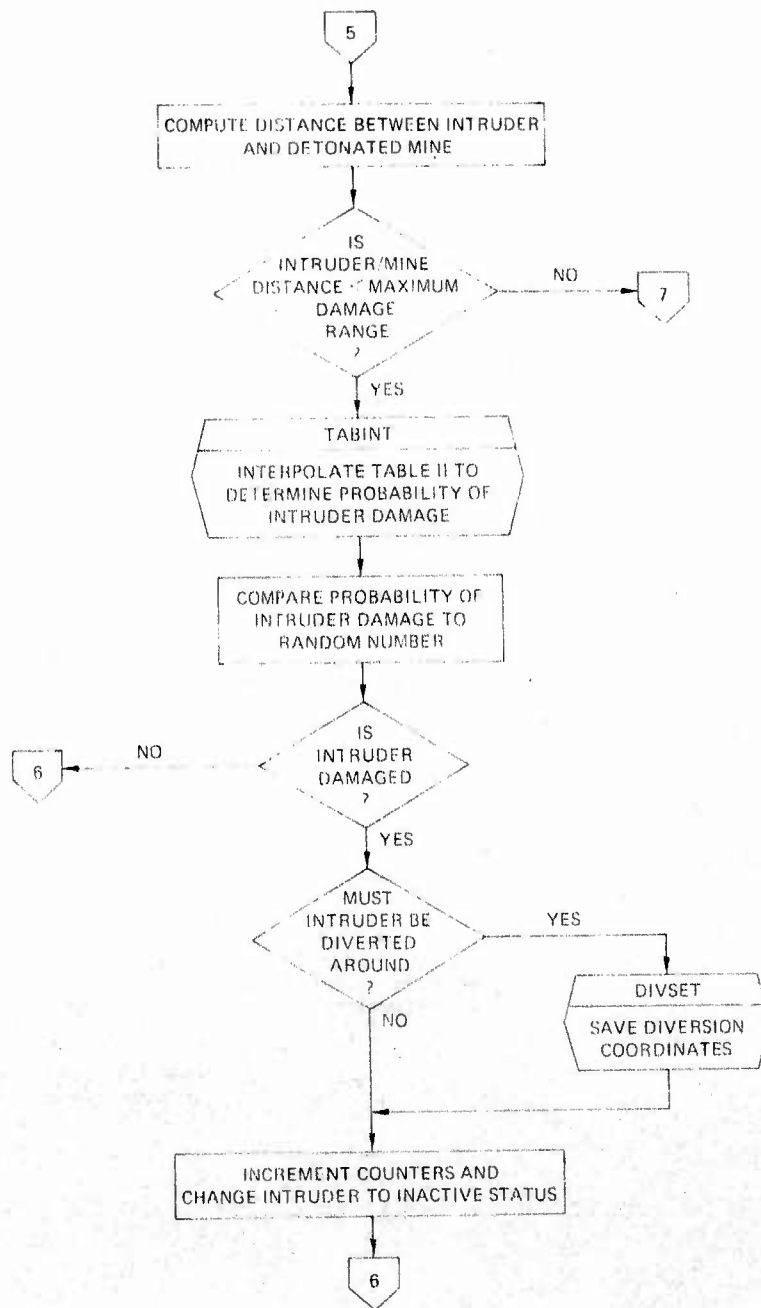


Figure 15. (Continued)

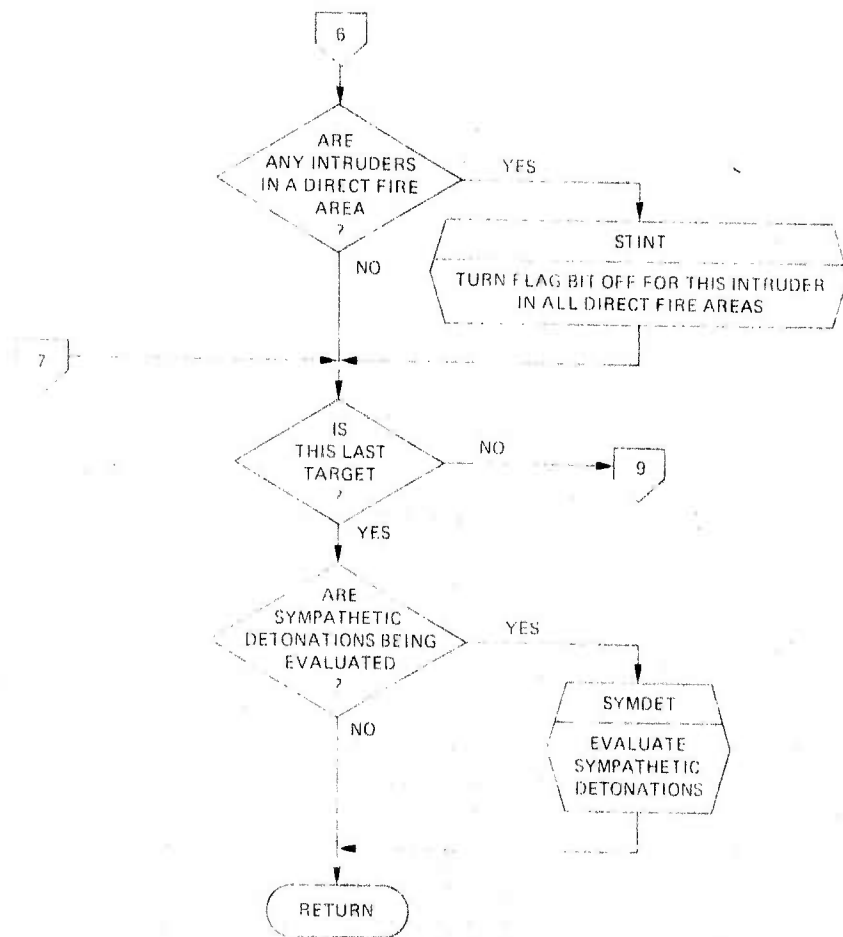


Figure 15. (Continued)

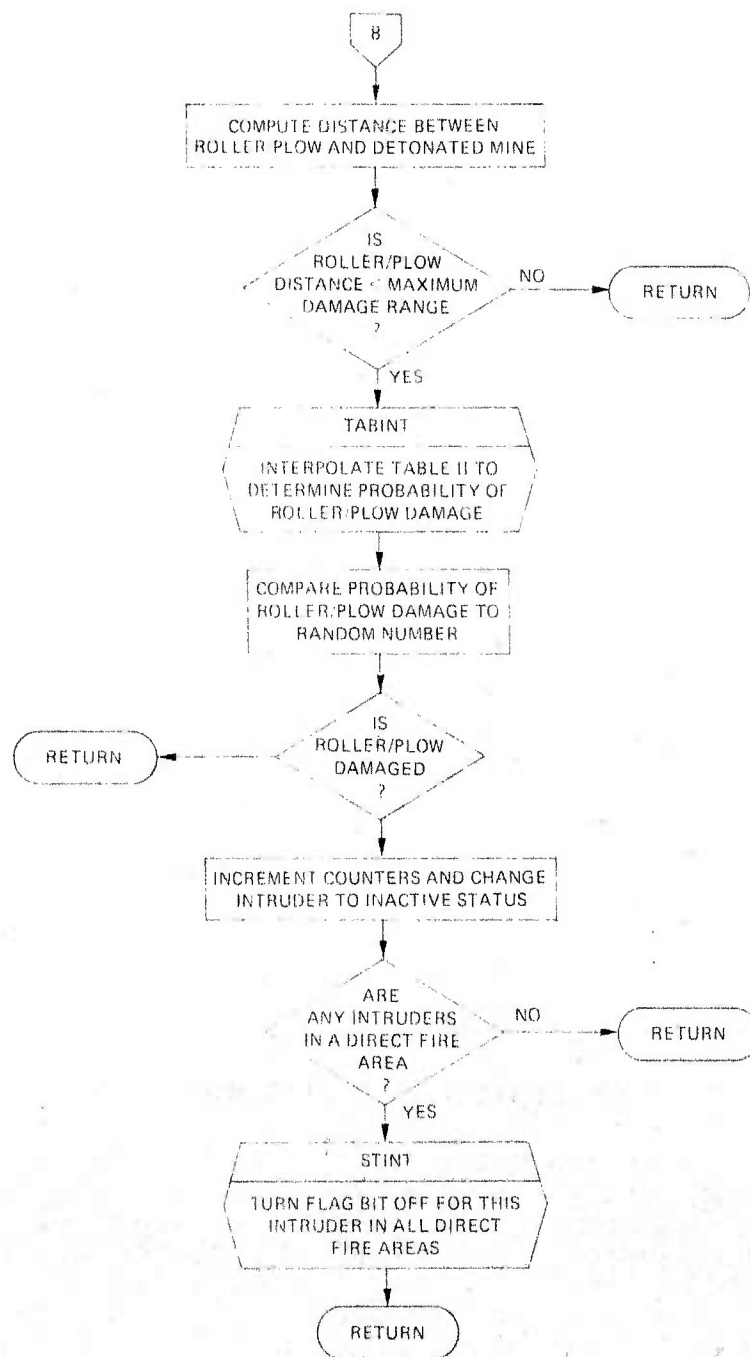


Figure 15. (Concluded)

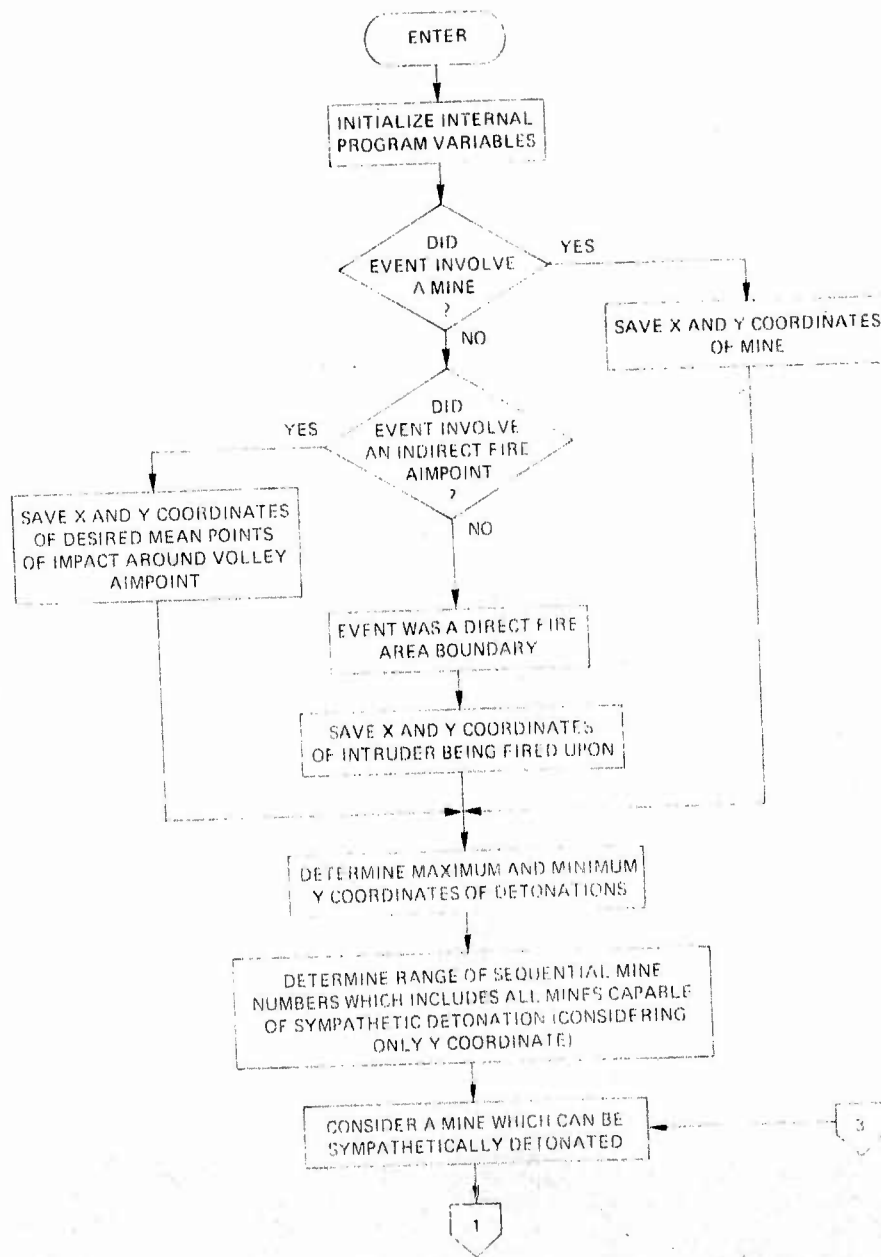


Figure 16. Flowchart, Subroutine SYMDNT

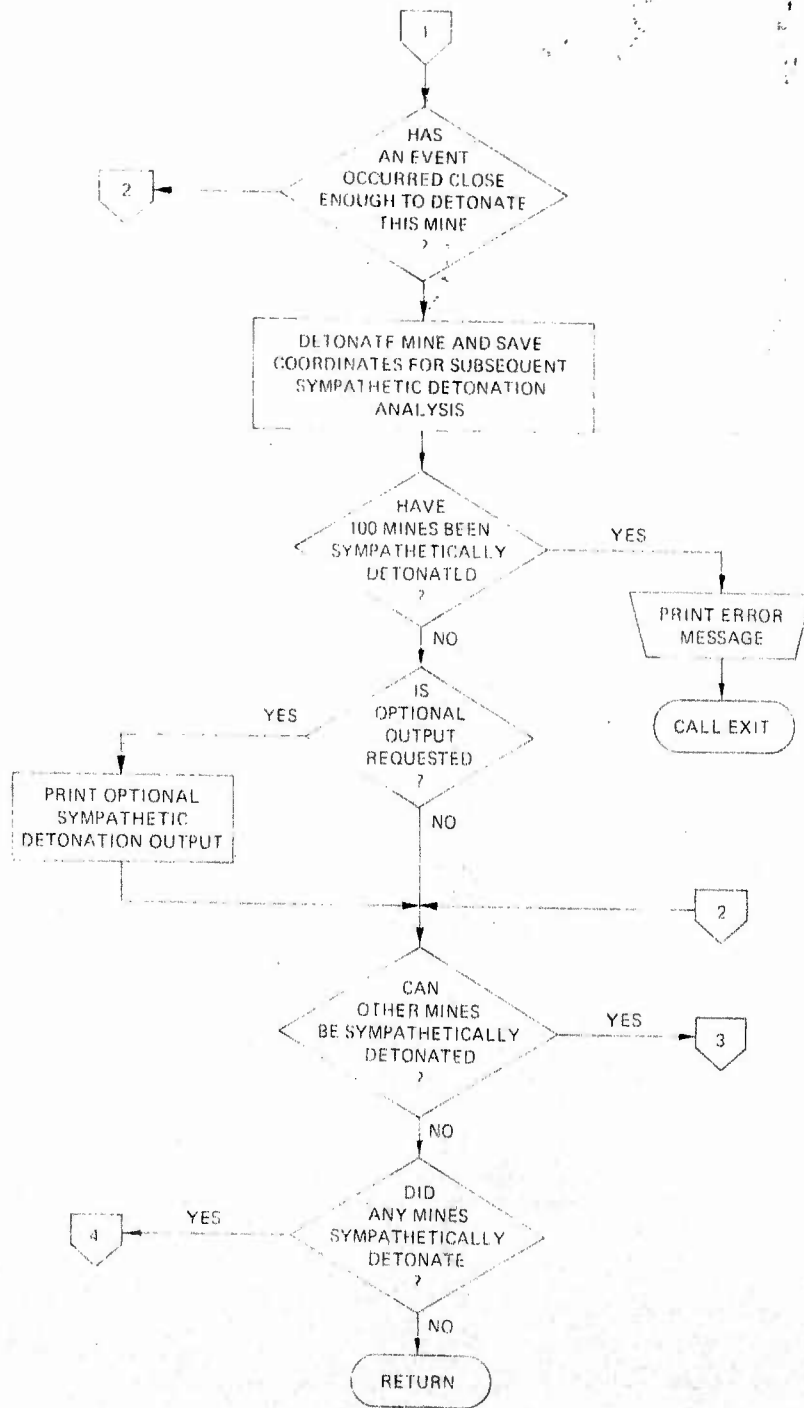


Figure 16. (Concluded)

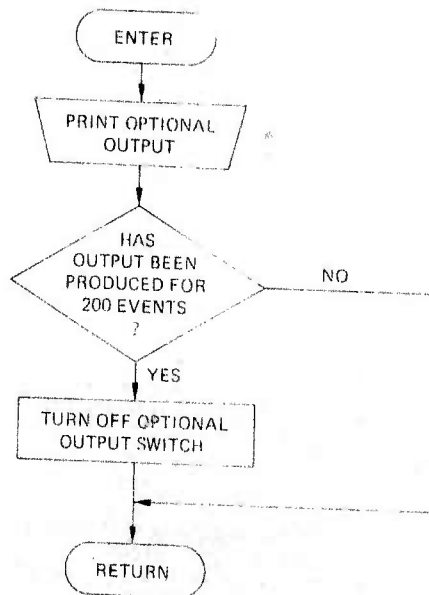


Figure 17. Flowchart, Subroutine PRINTO

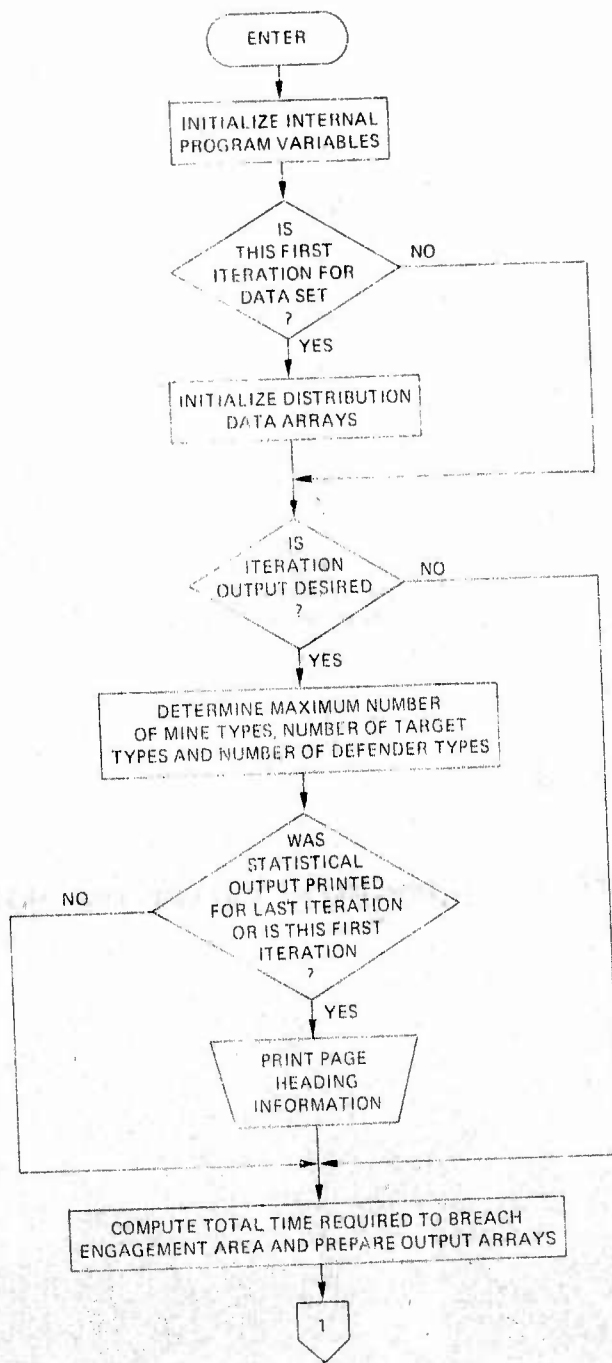


Figure 18. Flowchart, Subroutine PRINTR

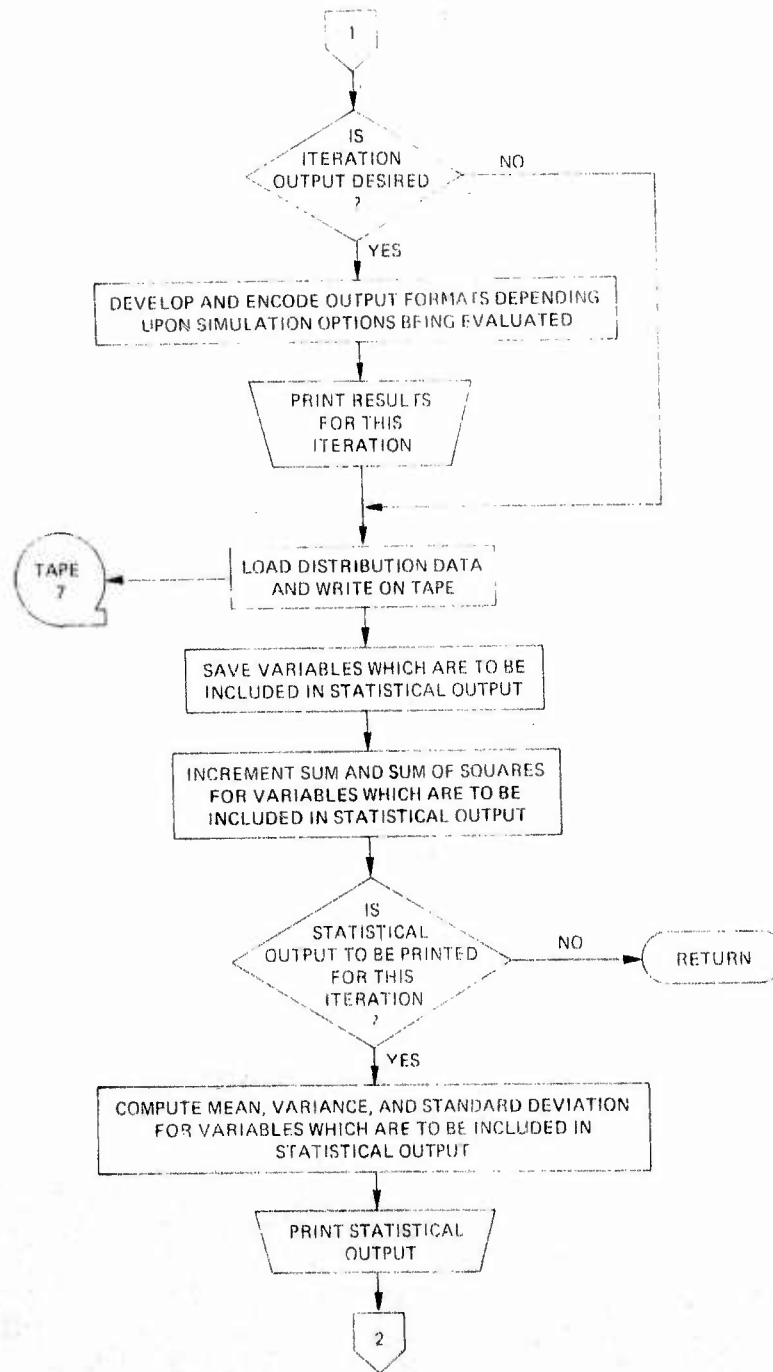


Figure 18. (Continued)

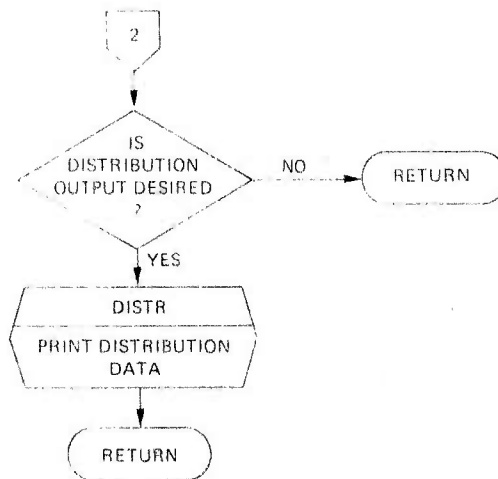


Figure 18. (Concluded)

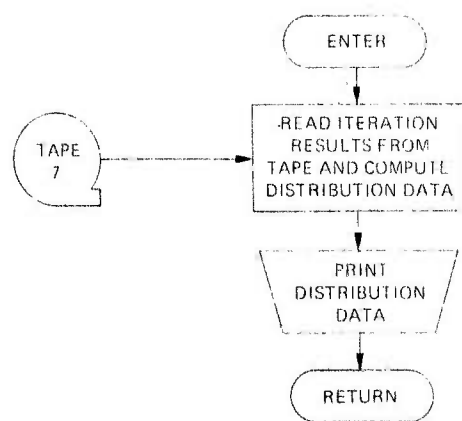


Figure 19. Flowchart, Subroutine DISTR

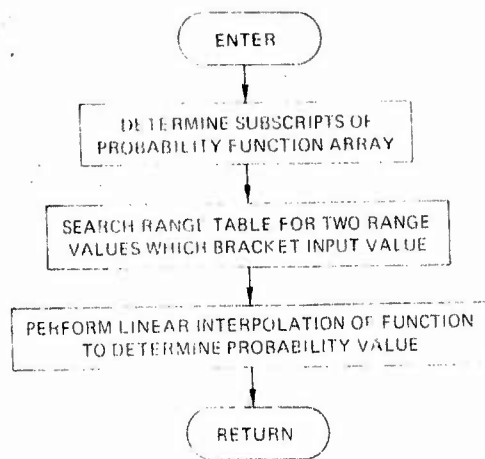


Figure 20. Flowchart, Subroutine TABINT

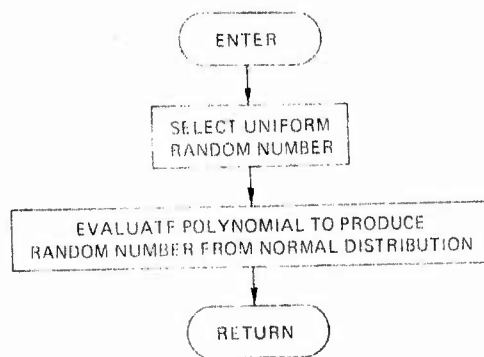


Figure 21. Flowchart, Subroutine RNORM

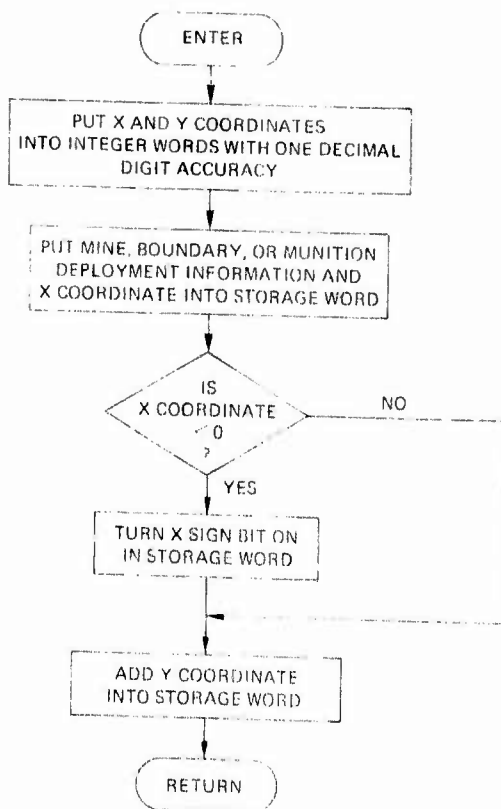


Figure 22. Flowchart, Subroutine IPACK

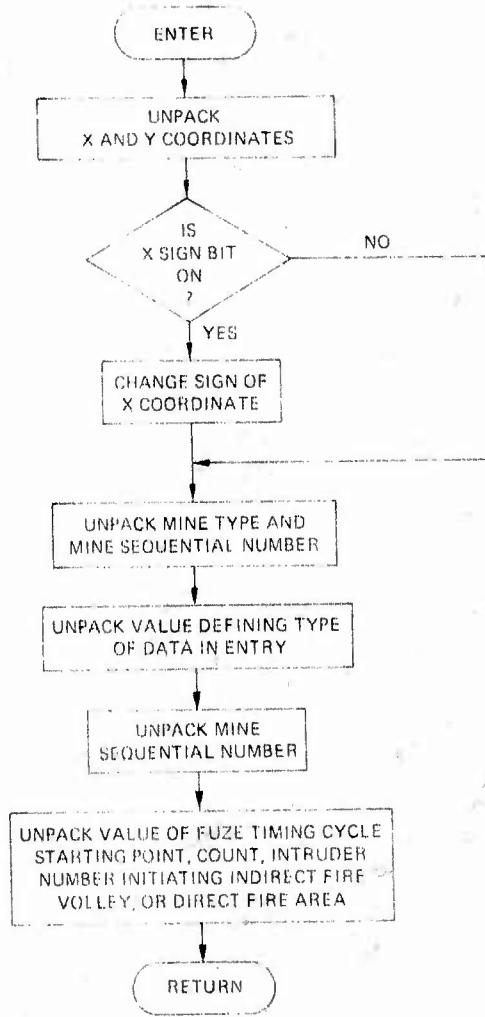


Figure 23. Flowchart, Subroutine UNPACK

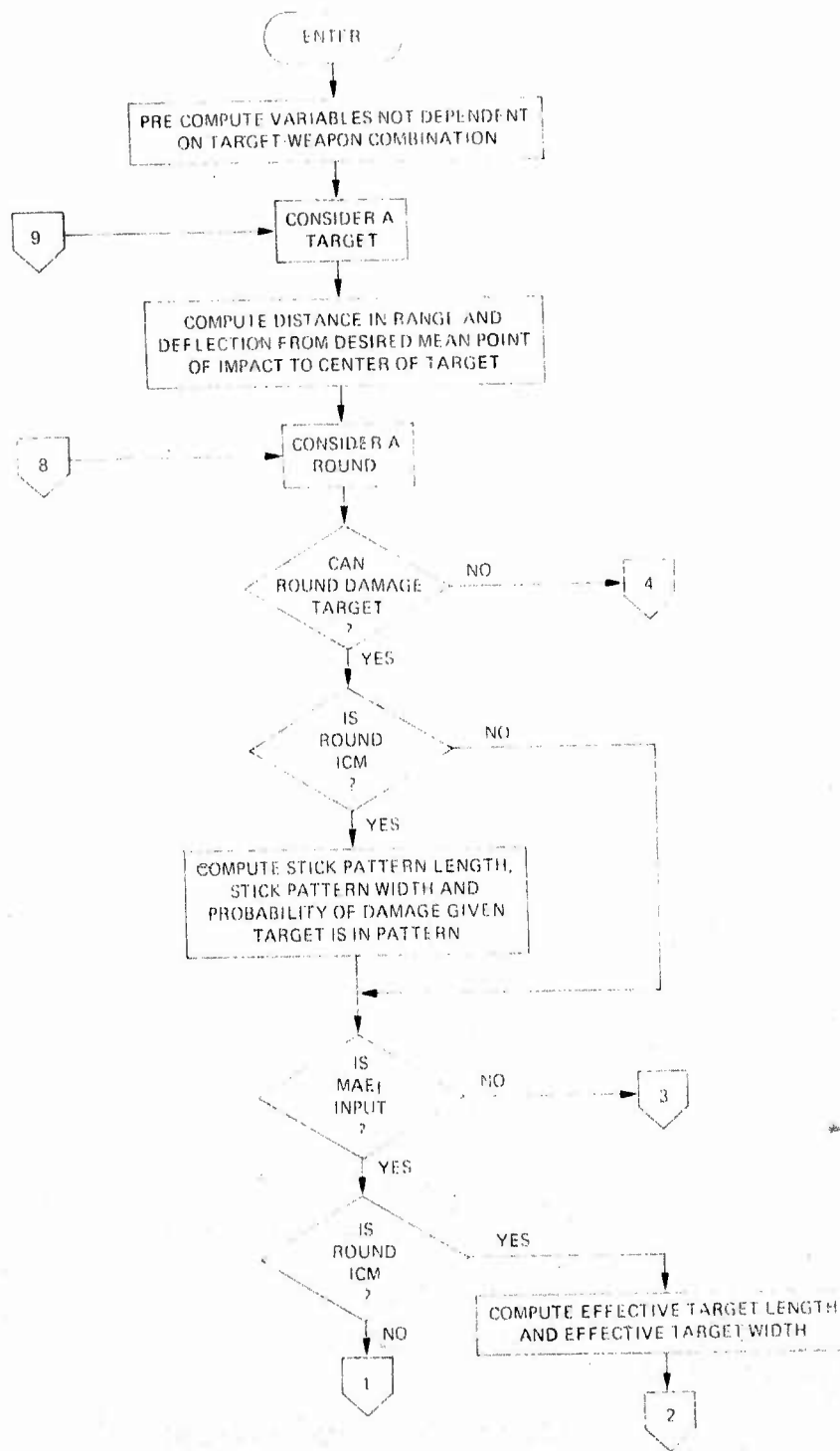


Figure 24. Flowchart, Subroutine NDFIRE

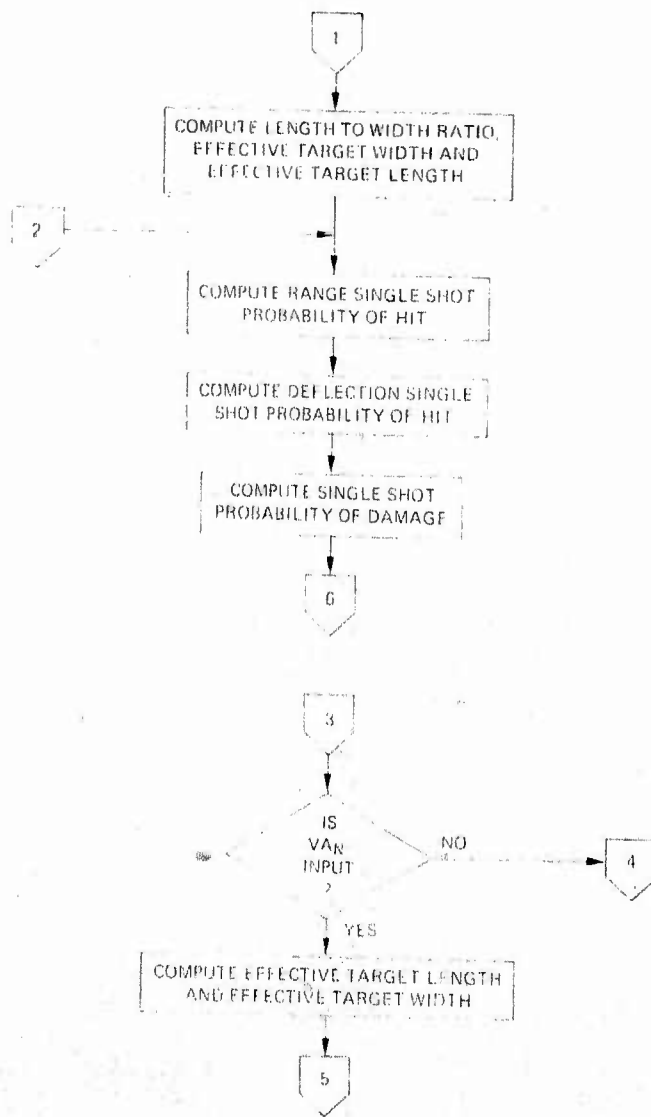


Figure 24. (Continued)

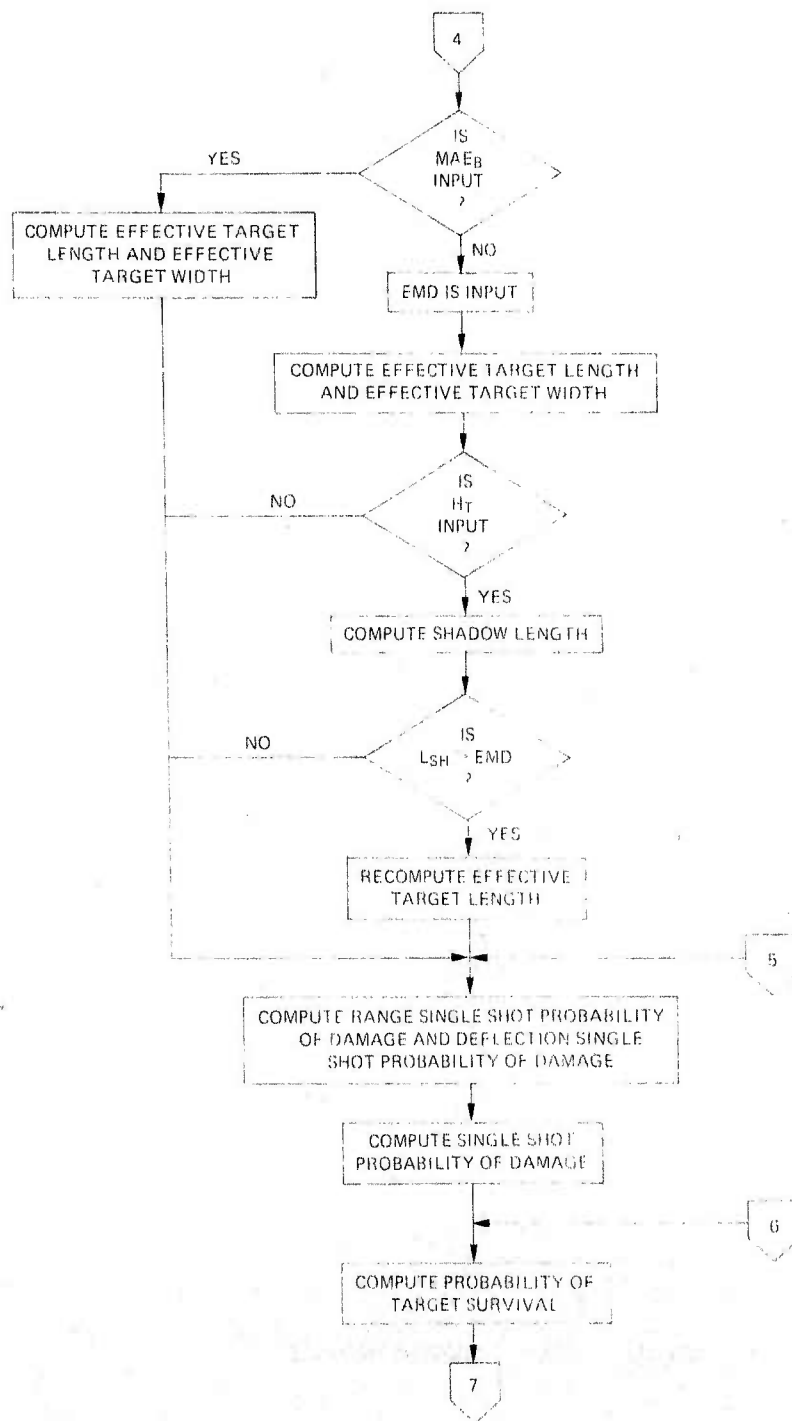


Figure 24. (Continued)

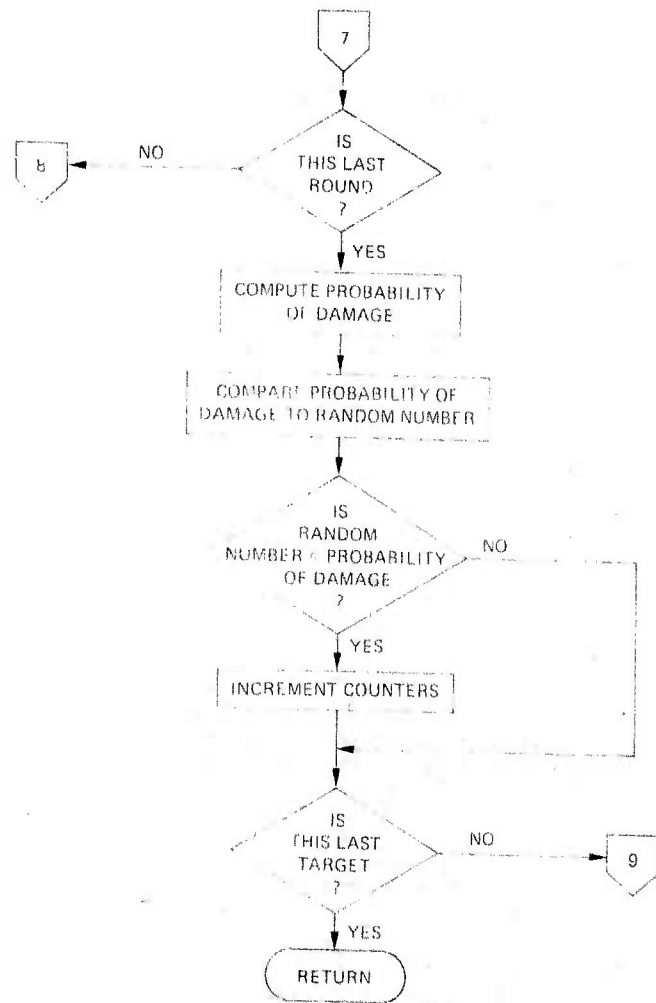


Figure 24. (Concluded)

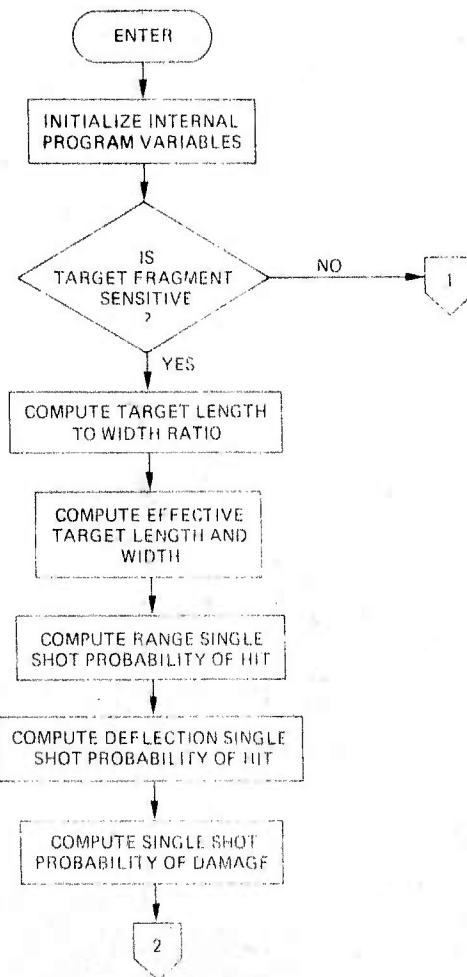


Figure 25. Flowchart, Subroutine DIRFIR

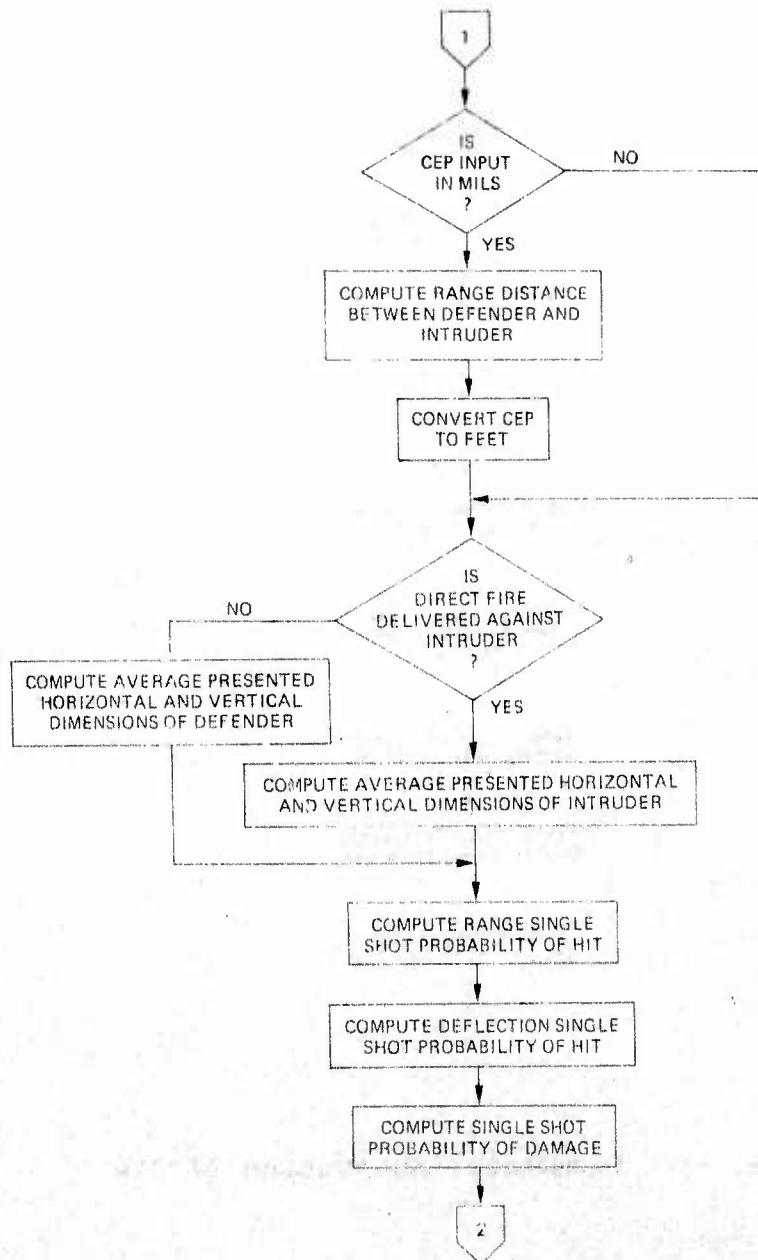


Figure 25. (Continued)

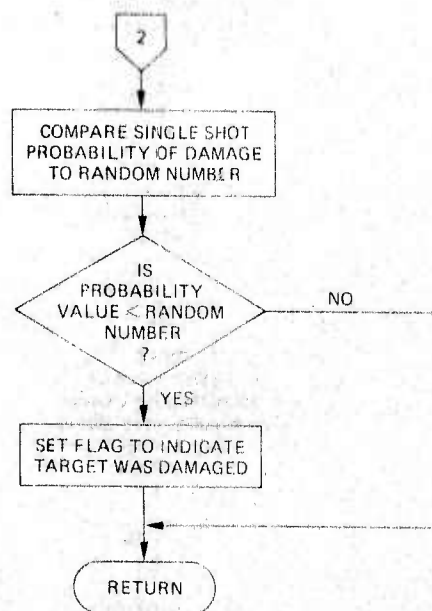


Figure 25. (Concluded)

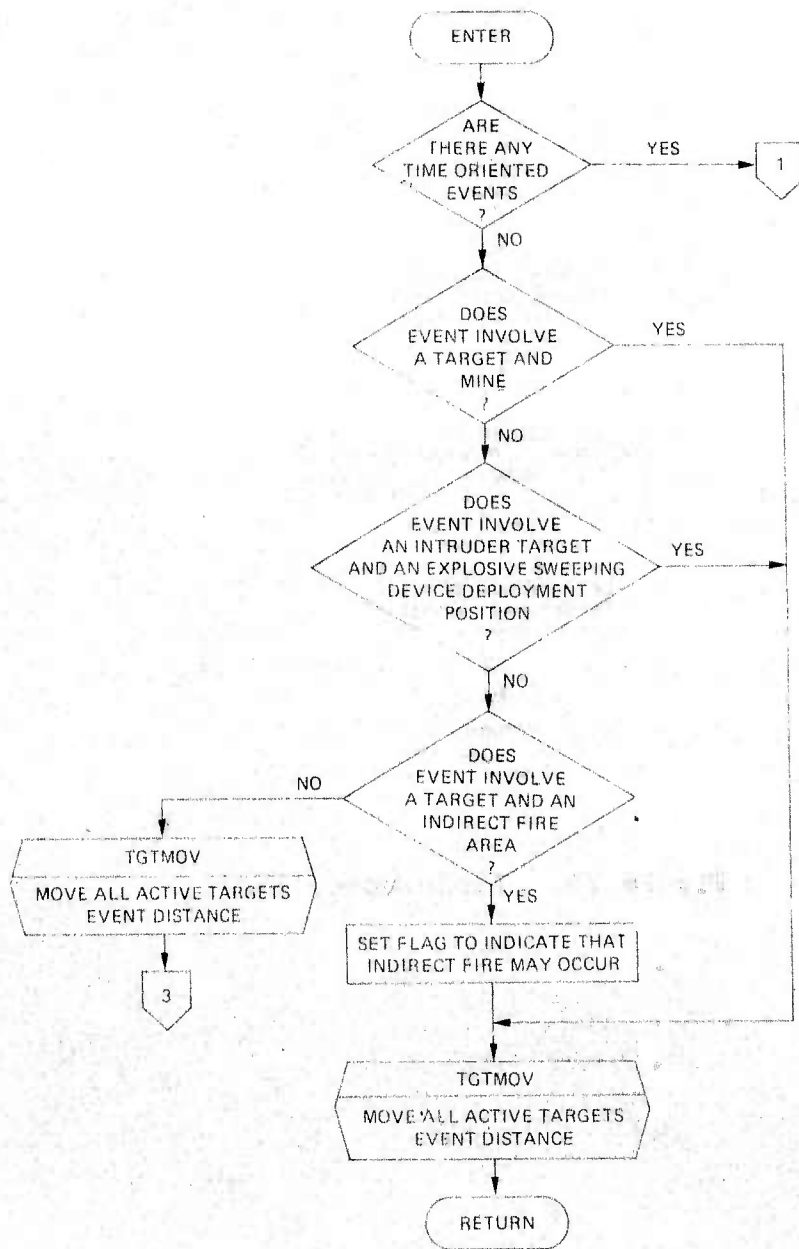


Figure 26. Flowchart, Subroutine CKEVTM

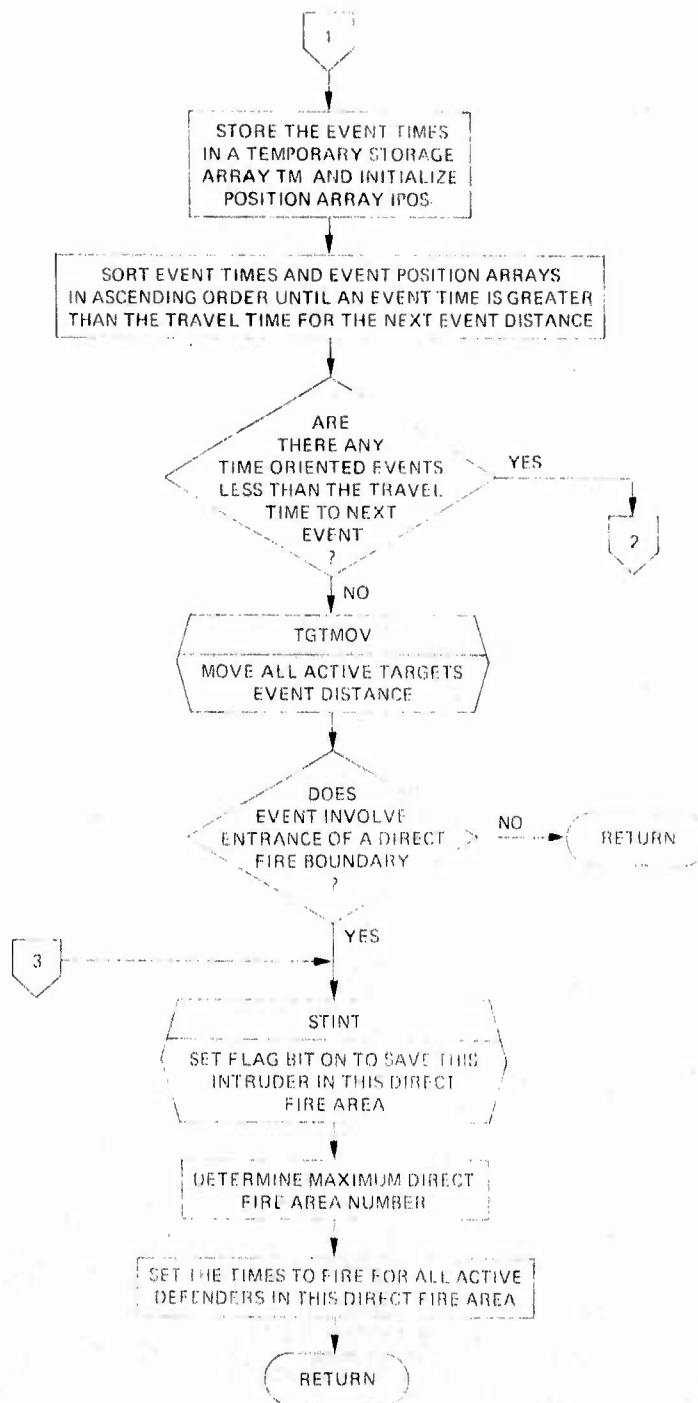


Figure 26. (Continued)

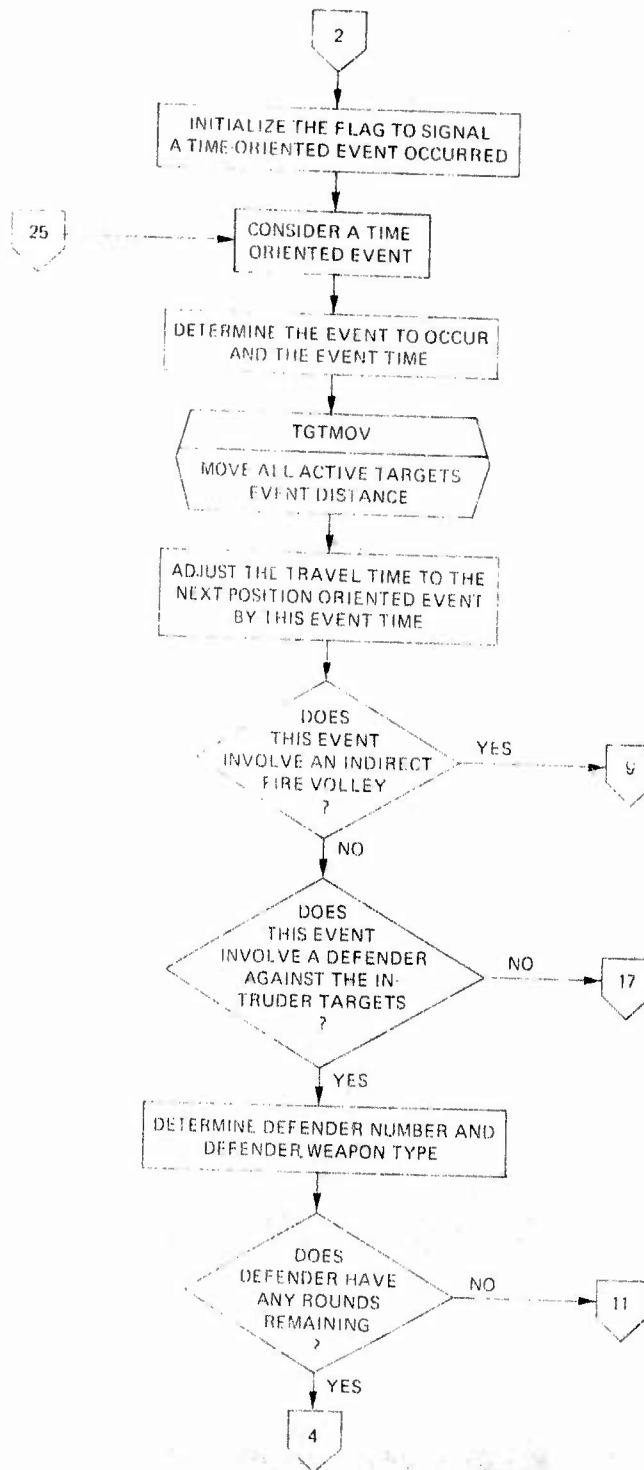


Figure 26. (Continued)

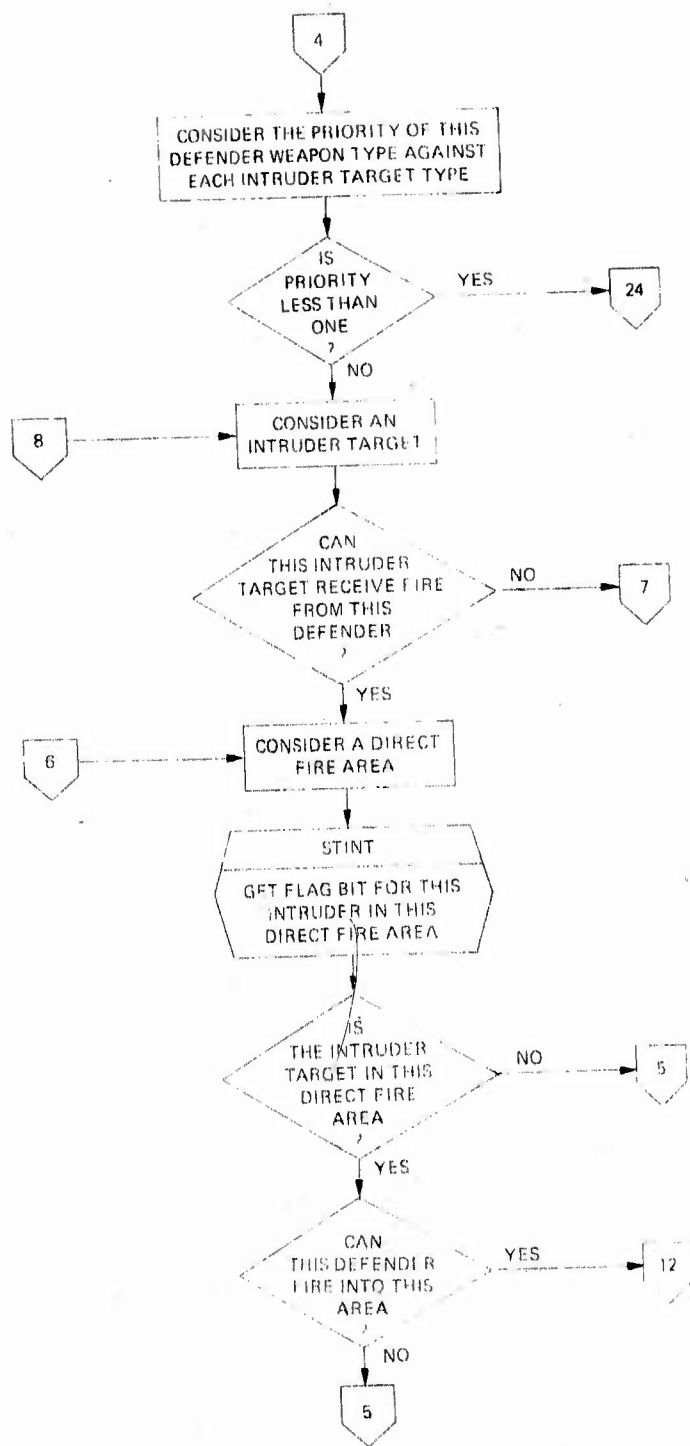


Figure 26. (Continued)

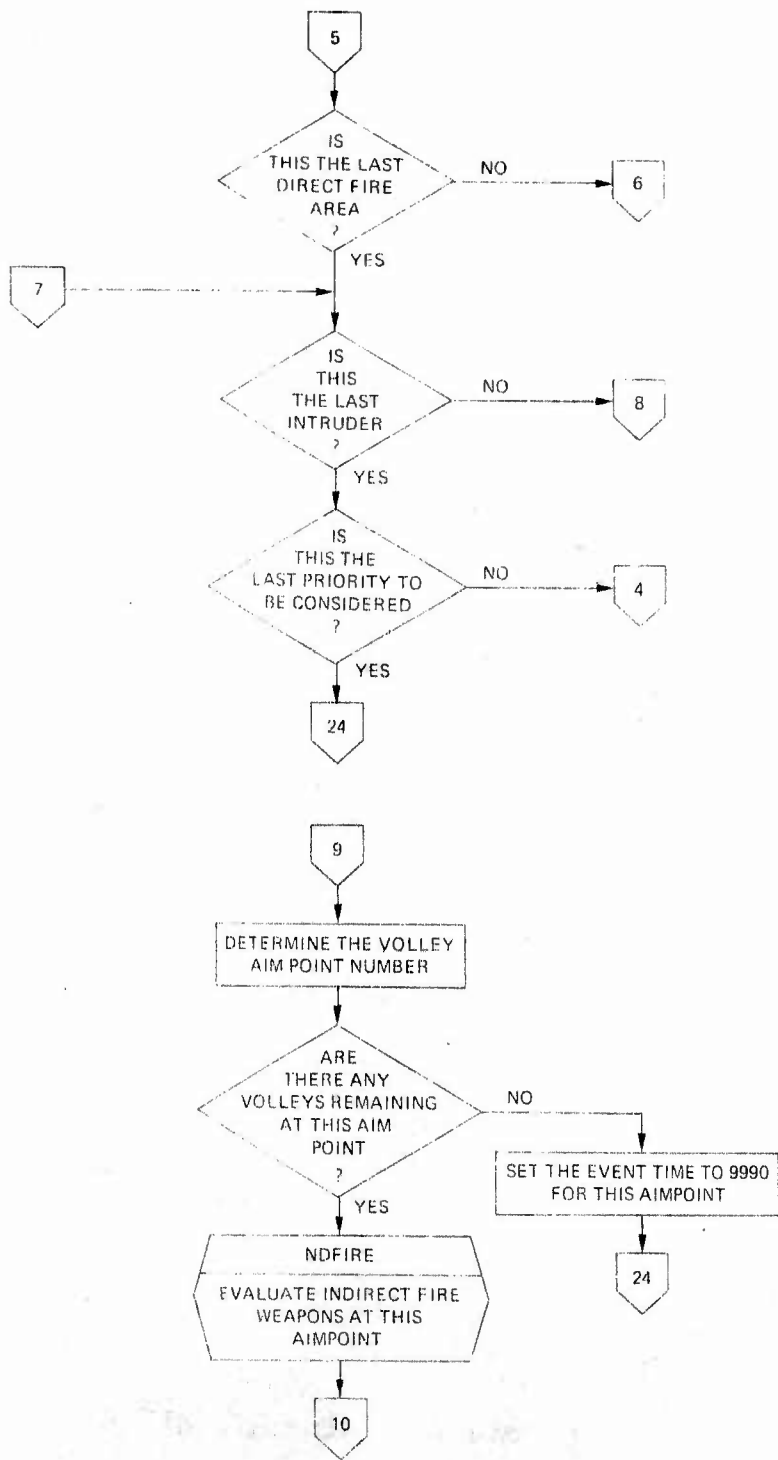


Figure 26. (Continued)

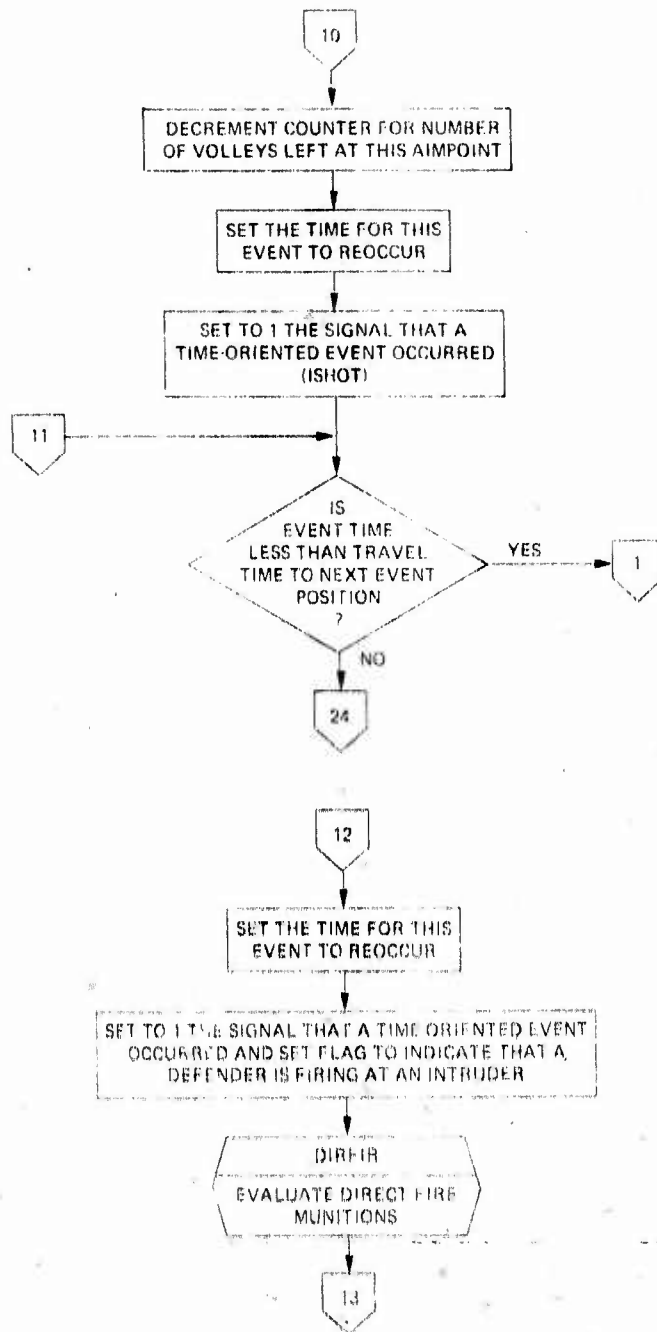


Figure 26. (Continued)

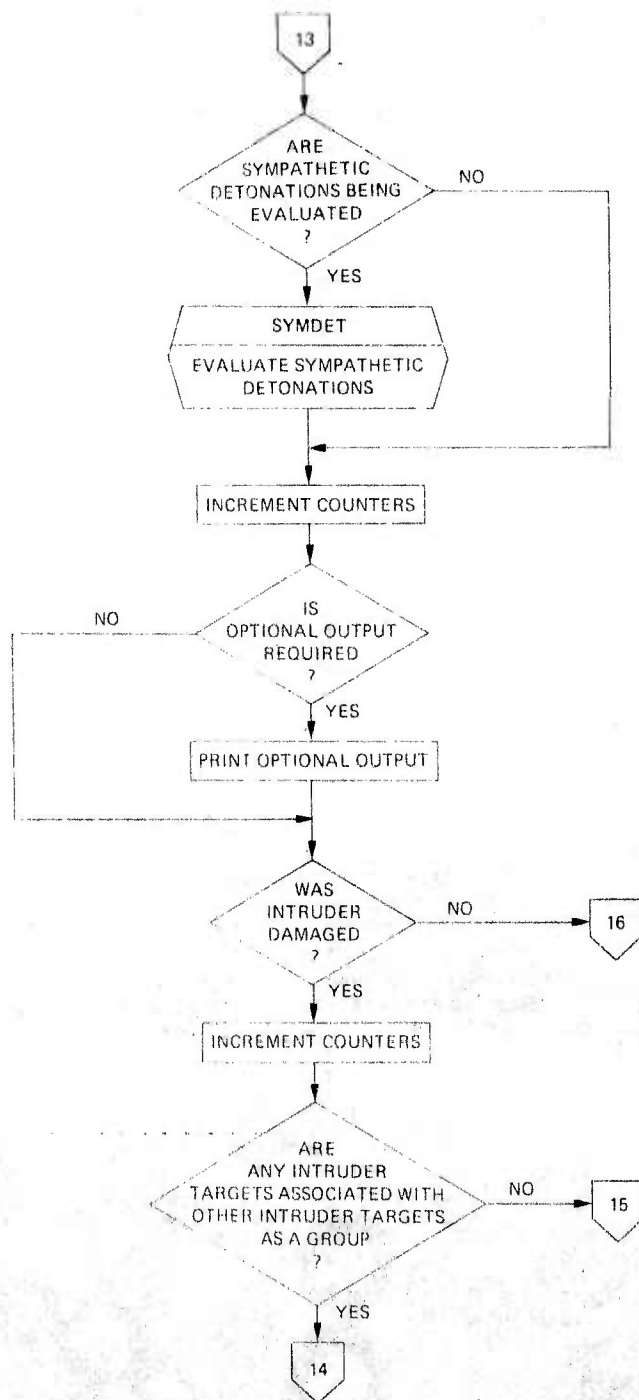


Figure 26. (Continued)

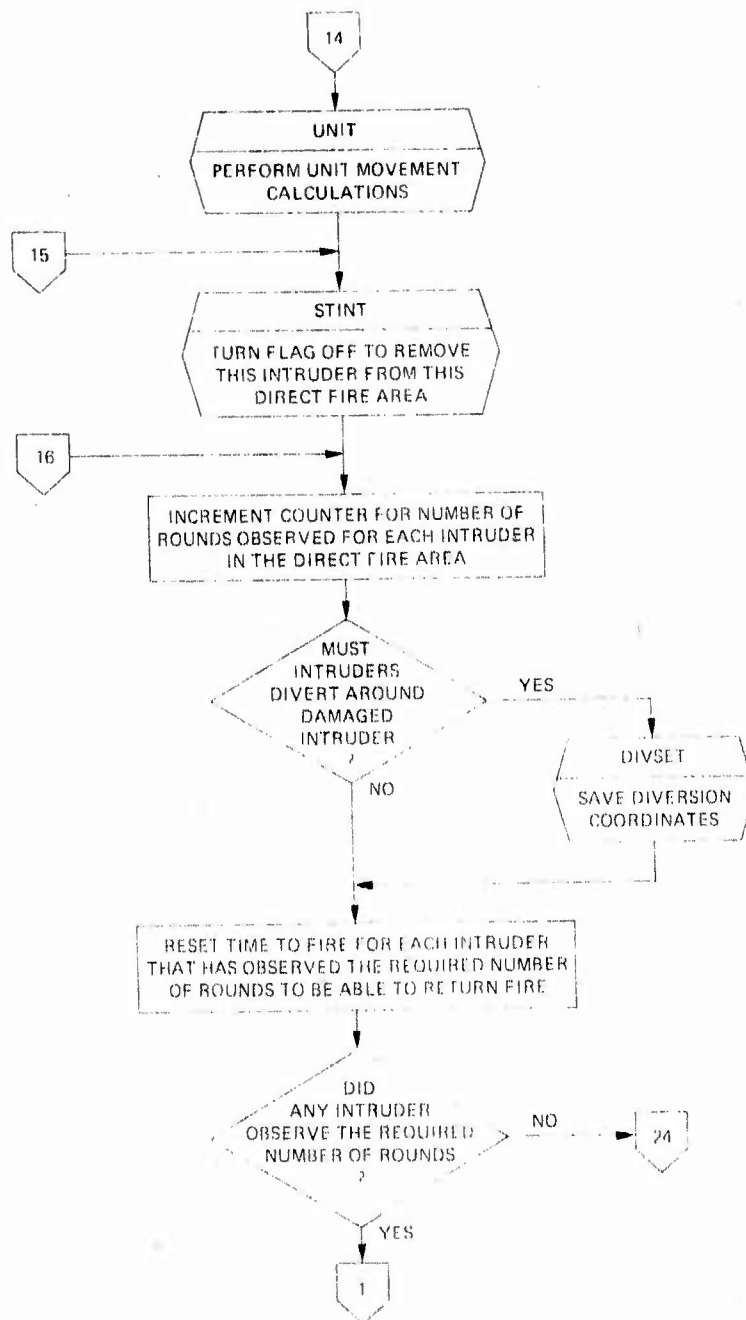


Figure 26. (Continued)

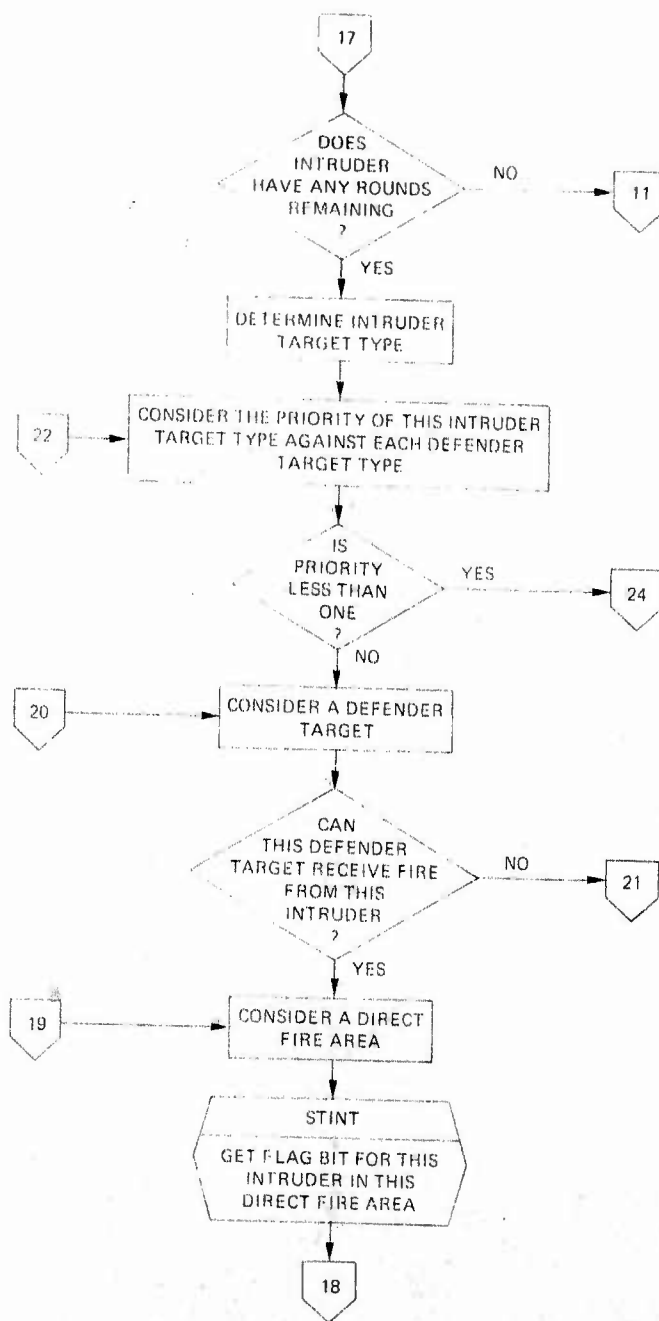


Figure 26. (Continued)

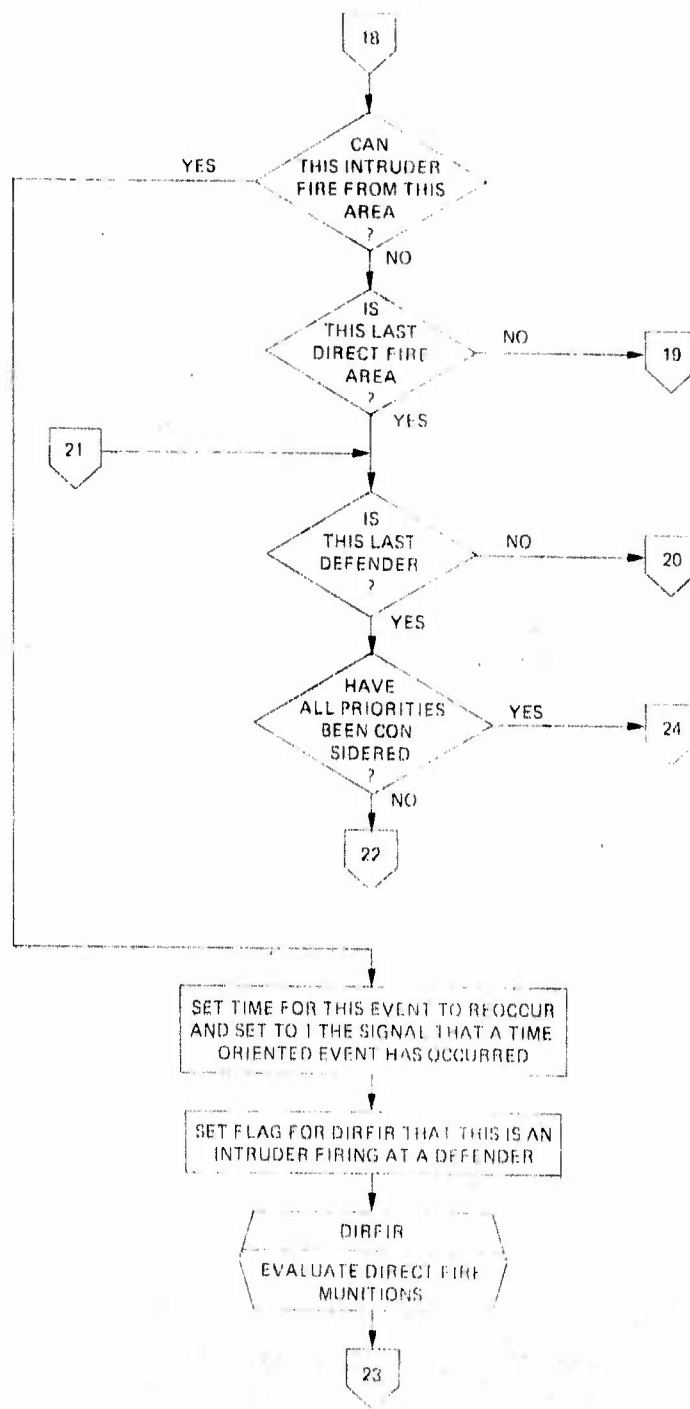


Figure 26. (Continued)

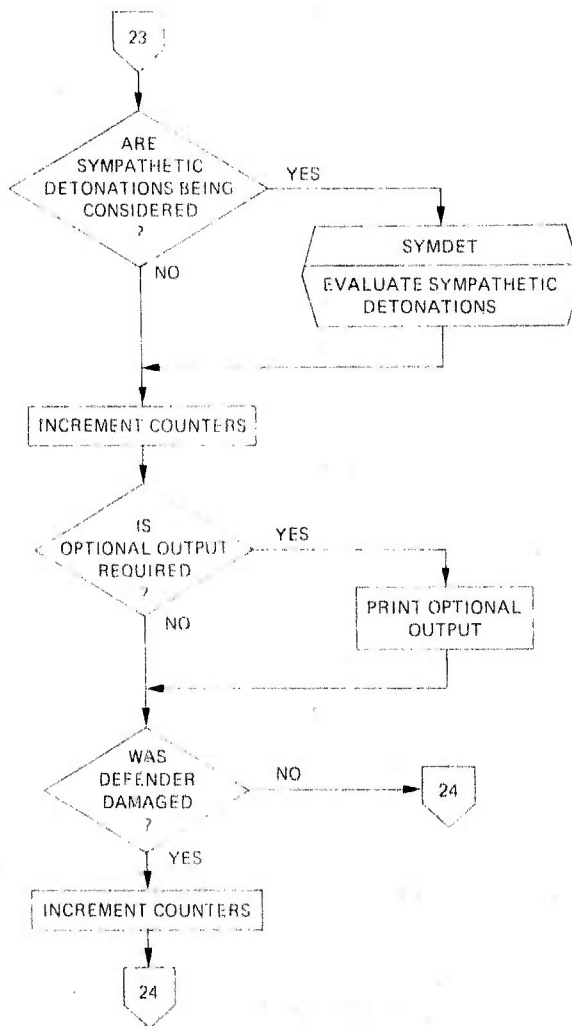


Figure 26. (Continued)

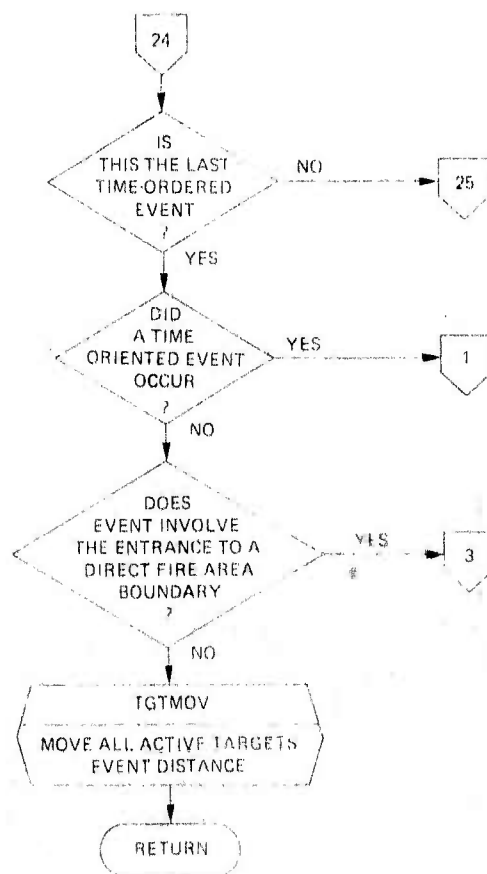


Figure 26. (Concluded)

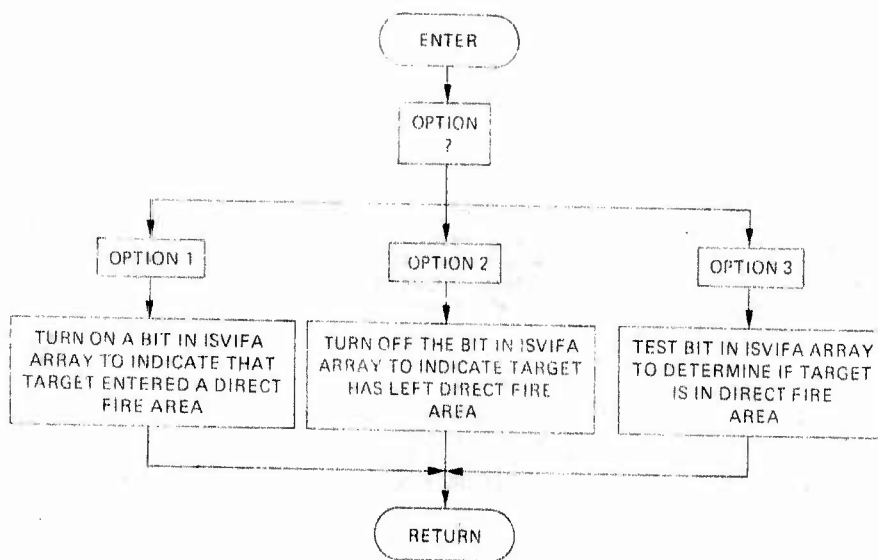


Figure 27. Flowchart, Subroutine STINT

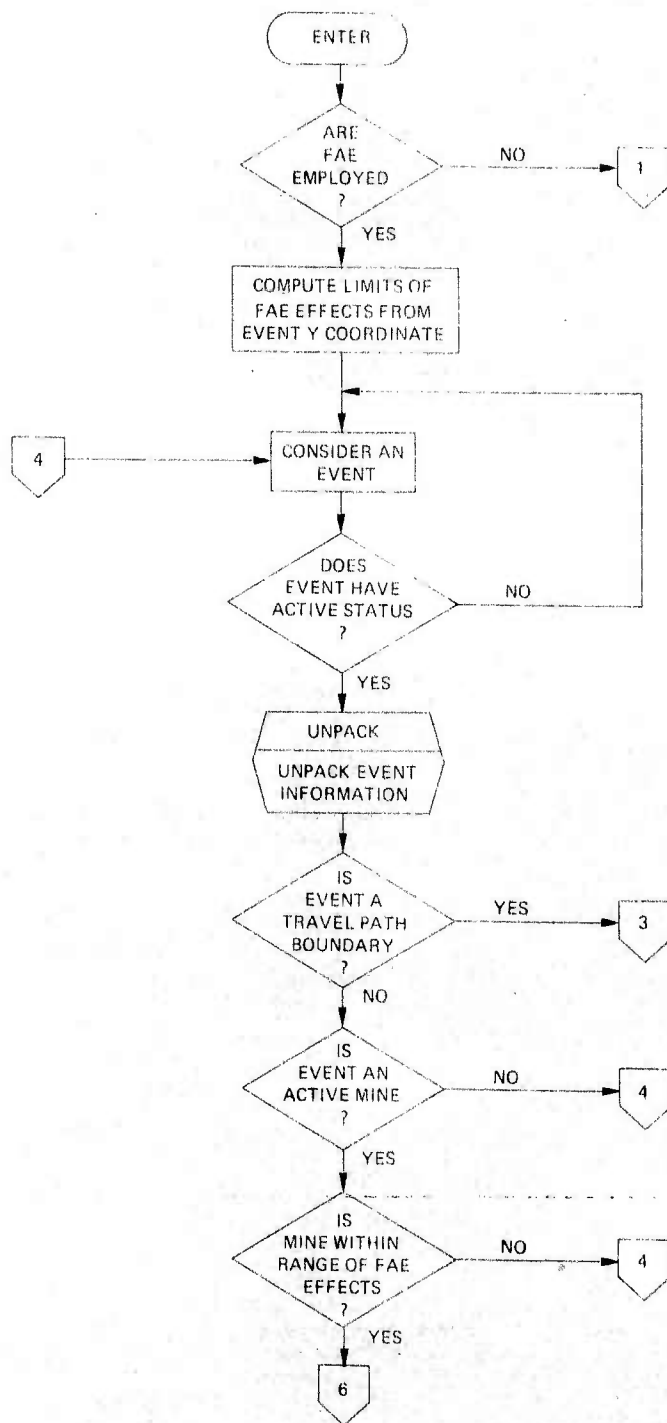


Figure 28. Flowchart, Subroutine EXPSWP

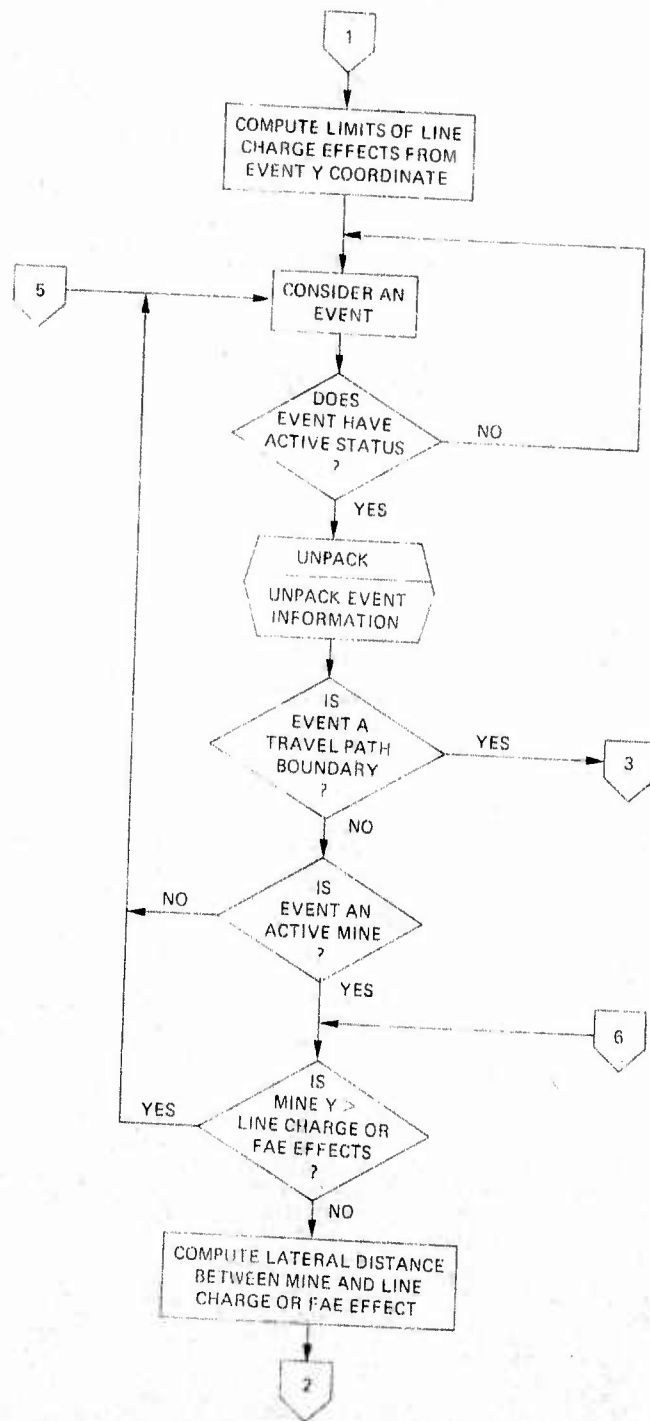


Figure 28. (Continued)

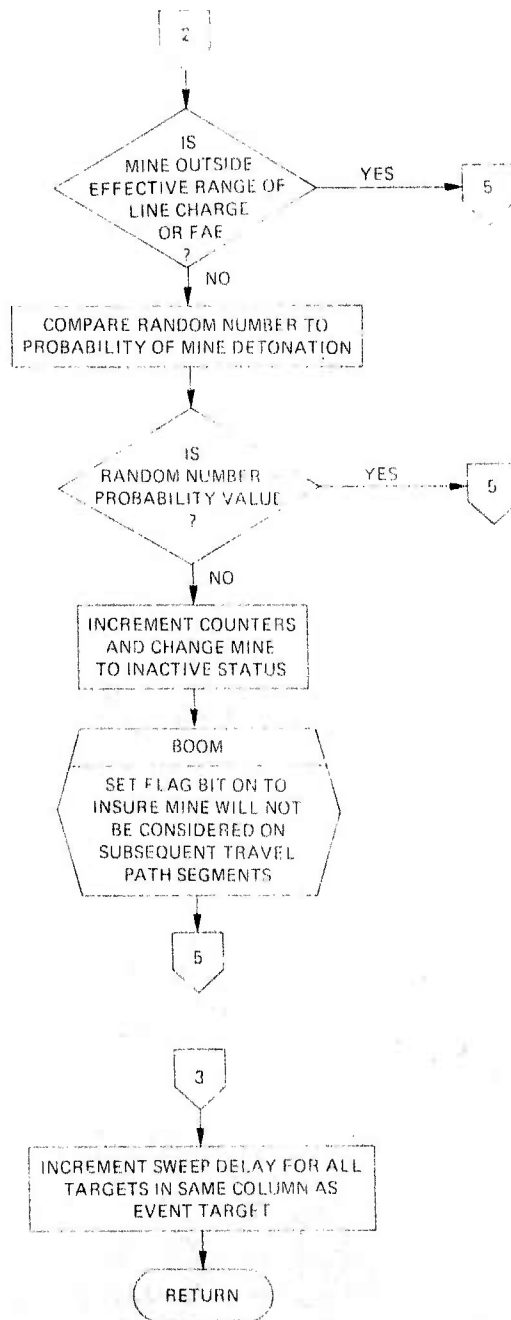


Figure 28. (Concluded)

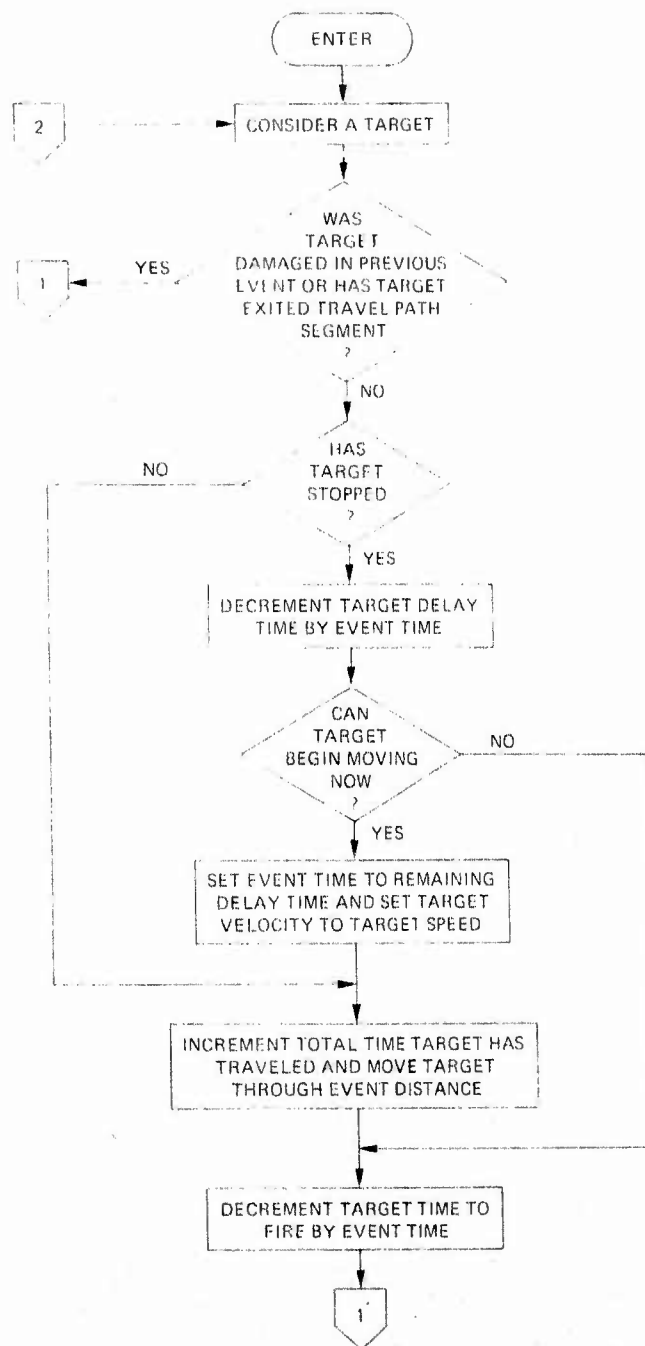


Figure 29. Flowchart, Subroutine TGTMOV

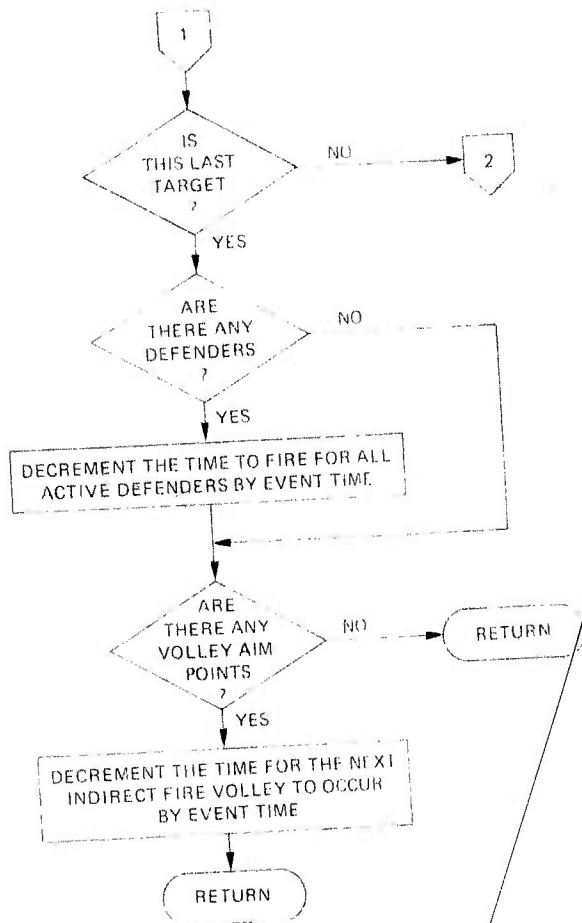


Figure 29. (Concluded)

SECTION IV  
SOURCE LISTING

Figure 30 presents a complete source listing of SEMAC which can be used to check the validity of the program.

```

*DECK SEMAC
PROGRAM SEMAC(OUTPUT=129,INPUT,TAPE5=INPUT,TAPE6=OUTPUT,
*TAPE1=513,TAPE2=513,TAPE4=129,TAPE7=513)
C
C MINEFIELD SIMULATION MODEL WITH COVERING FIRE - SEMAC
C
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDL,AYVYP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATH(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUDPRB(7),D3DEL,EI(5,5,2),ID,IDIOP,IPP(20,15,2),IEND,
*II5,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IWORD,IWDEF(20),IHTVAP(10)
COMMON JMARKI,JSELD5(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LDFOPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDFEA(15,20),NOF,NDFA,
*NDFAA,NDFOAM(5),NDFWD,NDFWT,NOVT,NOTAW(100),NOTAHT,NOTAWI(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMIN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADFD(20),NRADFI(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFIRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NHEPV(10),OBX,OBJ,OBYPRT,PATRAD(5),POLCF
COMMON PHD(5,5),PPEAF(7),PRBITY,PRDIND(5,7),PRKLO(5,7),
*PROBIL(840),PRSWD(15,7),RADFAE,RANGPR(840),RELND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDUF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARK(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTVSV,TGTXL(100),TGTYNW(100),TGTVOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEHV(100),THRO(5),THRI(5),TMDLCF,
*THSWP(7),TMTOPR(130),TOTSIM,TRAVTM,XCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XROFO(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YOMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YROFO(30),YROAD(11),YSWATH(50)
C....
KREAD = 0
5 CALL READIN (KREAD)
20 CALL SETUP
25 CALL ROAD
CALL SORT(10B,NOB)
IF(LEFTIN.EQ.0) GO TO 160
30 CALL L90PS
TGTVSV=TGTVEL(ISAVE)
IPO=1H
ISYP=0
KT = KTSAV(ISAVE)
IWORD = IOB(KT)
CALL UNPACK
OBYPRT=OBJ
CALL CKEVTH
IF(TGTYNW(ISAVE).GT.99990.) GO TO 110
IF(TGTVSV.GT.0.01)GO TO 50
KTSAV(ISVLST)=KTSAV(ISVLST)-1
GO TO 30
50 GO TO (72,72,72,72,72,72,75,72,72,72,72,72,72,
*80,110,100,77) NOBTP2
72 CALL TGTMIN
IF(NGTAWT.GE.1.AND.ISYP.EQ.2) CALL UNIT
GO TO 110
75 CALL EVENTC
GOTO 110
77 IF(TGTXL(ISAVE).NE.OBX.OR.NTGTYP(ISAVE).NE.LCFTT)GO TO 150
CALL EXPSWP
GO TO 110
80 IF(IT.NE.NCTAW(ISAVE))GO TO 150
IVAP=IMINE
CALL NOFIRE
CALL BOOM(1DET,1)
INDFA=1
IOB(KT)--1

```

Figure 30. SEMAC Source Listing

```

NVLIDF(I VAP)-NVLIDF(I VAP)-1
TMTOFR(I VAP+120)=TDBIFV(I VAP)
GO TO 110
100 CALL STINT(ISVIFA,ISAVE,IT,IBIT,KDFITN,2)
IF(KDFITN(ISAVE).GT.0) GO TO 102
NRFIRD(ISAVE)=0
TMTOFR(ISAVE)=9990.
102 JAA=0
DO 105 J=1,30
IF(ISVIFA(J))103,105
103 JAA=J
105 CONTINUE
IF(JAA.LT.1) GO TO 109
NDFA=MOD(JAA,15)
IF(NDFA.LT.1)NDFA=15
GO TO 110
109 NDFA=0
110 IF(NOBEVT.GT.0.AND.IRUNS.EQ.IPRINT)
CALL PRINTO
NOBEVT=IABS(NOBEVT)
IF(NDFA)130,150
130 JAA=0
DO 140 I=1,NVAP
IF(NVLIDF(I))140,140,135
135 JAA=1
140 CONTINUE
NDFA=JAA
150 IF(NSPLFT.GT.0.OR.KNRS.NE.NRS) GO TO 155
NSPLFT=99999
DO 152 I=1,NTGO
IF(TGTYNW(I).LT.99970.) TGTVEL(I)=TGTQVL
152 CONTINUE
TOTSPD=TGTQVL
155 IF(LEFTN.GT.0) GO TO 30
160 IF(KNRS.NE.NRS) GO TO 25
CALL PRINTR
IF (IRUNS LT. NOLT) GO TO 20
KREAD = 1
GO TO 5
END

```

Figure 30. (Continued)

```

SUBROUTINE READIN (KREAD)
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDL,AYYVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATH(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*ODDPRB(7),O3DEL,EI(5,5,2),ID,IDI5OP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYHP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IMORD,IMTDEF(20),IMTVAP(10)
COMMON JMARKI,JSELD(7),KABOOM(547),KDFITN(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LDFOPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MTFA(50),MUSH,NAP,NCTAW(100),NG,NDFEA(15,20),NDF,NDFA,
*NDFAA,NDFOAM(5),NDFHD,NDFHT,NBYT,NOTAW(100),NOTAWT,NOTAWI(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NNOET(7),
*NMIN(7),NMSWP(7),NHT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF1(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFIRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTICS(5),NTTTHD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NNEPV(10),OBX,OBJ,OBYPRT,PATRAD(5),PDLCF
COMMON PHD(5,5),PPEAF(7),PRBITY,PROTNO(5,7),PRKLO(5,7),
*PROBIL(840),PRSWD(5,7),RADFAE,RANGPR(840),RELAND(5),RELSUB(5),
*REP(10),RN,ROTHMPX(10,10),ROTHMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGDD(50),SIGOR(50),STATAL(5,52),SWDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMHAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TOBIFV(10),TGTDEL(100),TGTOLV,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTXL(100),TOTYNN(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEMV(100),TMBRD(5),TMBRI(5),TMDLCF,
*TMSWP(7),TMTOPR(130),TOTSIM,TRAVTH,XCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRDFO(30),XRROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDFO(30),YROAD(11),YSWATH(50)
C****
DIMENSION ORX(4),ORY(4),XYMS(4),RNGSV(3)
DIMENSION HOL(8),SECOFF(7),PROBTP(8),RANGTP(8)
DATA ITEST/5H 999/
REWIND 2
WRITE(6,1480)
IF(KREAD.EQ.1) GO TO 24
CALL REPRNT
8010 DO 7 I=1,840
PROBIL(I) = 0.0
RANGPR(I) = 0.0
7 CONTINUE
ND=0
OBX=OBJ=0.
DO 6 I=1,10
6 NVFAEA(I)=0
DO 8 J=1,15
DO 8 I=1,20
8 NDFEA(J,I)=0
LCFOPT=0
MADIV=0
DO 9 I=1,5
NRAI(I)=NRAD(I)=0
TARRAD(I)=TARL(I)=TARW(I)=TARHT(I)=0.
REP(I)=REP(I*2)=0.
DEP(I)=DEP(I*2)=0.
NSUB(I)=1
DO 9 J=1,7
SYMDDF(I,J)=0.
9 SYMDIF(I,J)=0.
DO 10 I=1,7
TMSWP(I)=SECON(I)+SECOFF(I)+XCYLE(I)=0.
DO 10 J=1,7
10 SYMDIS(I,J)=0.
IEND = 0
24 IEND = IEND + 1
JMARKI=1
IRUNS=0
DO 23 I=1,100
23 DELATH(I)=0
DO 25 I=1,7
25 INTIME(I)=0
READ(4,9000) HOL
DECODE(80,9000,HOL) INPUT

```

Figure 30. (Continued)

```

IF (INPUT.EQ.1TEST) CALL EXIT
DO 38 J=1,52
DO 38 I=1,5
STATAL(I,J)=0.
38 CONTINUE
50 READ(4,9000)HOL
9000 FORMAT(8A10)
DECODE(5,9001,HOL) INPUT
9001 FORMAT(15)
IF (INPUT.EQ.99) GO TO 7000
IF (INPUT.GT.1000) GO TO 6900
DO 100 I=100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400
,1500,1600,1700,1800,1900,2000,2100,2200,2300,2400) INPUT
100 DECODE(80,9002,HOL)NAP,NTGO,NTGTP,NOIT,NOSTAT,IPRINT,SEED.
*NMT,MODE,NVAP,NCFAA,NDFWD,NRBA
9002 FORMAT(5X,6I5,F10.2,6I5)
IF (SEED.LE.0.) SEED=RANF(DUMMY)
CALL RANSET(SEED)
GO TO 50
200 DECODE(80,9003,HOL)LCFOPT,YLENGTH,XWIDTH,THETA,D3DEL,
*(1SYMP(1),1=1,3),1SBL,ITEROP,10(SOP
9003 FORMAT(5X,15,2F10.2,2F5.2,4I5,12,13)
GO TO 50
300 DECODE(80,9004,HOL)INTN,NTOTYP(NTN),NCTAW(NTN),NGTAW(NTN),
*GTGXOL(NTN),GTGYOL(NTN),N1,N2,N3,N4,T1,T2
IF (N1.LT.1) GO TO 50
NTGTYP(N1)=N2
NCTAW(N1)=N3
NGTAW(N1)=N4
GTGXOL(N1)=T1
GTGYOL(N1)=T2
9004 FORMAT(2(5X,15,11,14,15,2F10.2))
GO TO 50
400 DECODE(80,9005,HOL)ITYP,IARL(ITYP),IARH(ITYP),IARRAD(ITYP),
*IARHT(ITYP),IMBR1(ITYP),NRA(ITYP),(LDFDPR(ITYP),1=1,5),
*NTTICS(ITYP),NTTTMD(ITYP),NTTTRP(ITYP)
IF (NTTTMD(ITYP).GT.0)MADIV=1
9005 FORMAT(5X,15,4F10.2,F5.2,15,5I1,3I5)
GO TO 50
500 DECODE(80,9006,HOL)IVAP,IROAD,XOVP(IVAP),YOVP(IVAP),AYYVP(IVAP),
*NVFAEA(IVAP),NWEPV(IVAP),IWTYAP(IVAP),NTLCIF(IVAP),TDBIFV(IVAP)
9006 FORMAT(10X,2I5,3F10.2,4I5,F10.2)
IVPO=IFIX(XOVP(IVAP)*100.)
XOVP(IVAP)=FLOAT(IVPO*100+IROAD*1SIGN(1,IVPO))
GO TO 50
600 DECODE(80,9007,HOL)IVAP,IWEPV,DMP1IX(IVAP,IWEPV),
*DMP1IY(IVAP,IWEPV)
9007 FORMAT(10X,2I5,2F10.2)
GO TO 50
700 DECODE(80,9008,HOL)IVAP,IWT,ANGIMP(IVAP),REP(IVAP),DEP(IVAP),
*NSUB(IWT),RELSUB(IWT),PATRAD(IWT),RELNRD(IWT)
9008 FORMAT(5X,2I5,F5.2,2F10.2,15,3F10.2)
GO TO 50
800 DECODE(80,9009,HOL)NRS,TGTUVL,TGTVL2
9009 FORMAT(10X,15,2F10.2)
TGTUVL=TGTUVL*88.
TGTVL2=TGTVL2*88.
N=NRS+1
READ(4,9007)(XROAD(I),YROAD(I),I=1,N)
9007 FORMAT(8F10.2)
GO TO 50
900 DECODE(80,9010,HOL)IAP,MFA(IAP),NSTICK(IAP),AIMPTX(IAP),
*AIMPTY(IAP),YSWATH(IAP),XSWATH(IAP)
9010 FORMAT(5X,3I5,4F10.2)
GO TO 50
1000 DECODE(80,9011,HOL)IAP,SIGAR(IAP),SIGAD(IAP),SIGBR(IAP),SIGBD(IAP)
9011 FORMAT(5X,15,4F10.2)
GO TO 50
1100 DECODE(80,9009,HOL)MT,JSELDS(MT),TMSHP(MT),DUDPRB(MT),PPEAF(MT)
9009 FORMAT(10X,2I5,3F10.2)
IF (JSELDS(MT).GT.0) GO TO 50
DO 902 M=1,NAP

```

Figure 30. (Continued)

```

      IF(MTFA(M).EQ.MI) GO TO 903
902 CONTINUE
903 J=0
905 READ(4,9007)(ORX(I),ORY(I),I=1,4)
      DO 910 I=1,4
      WRITE(2) ORX(I),ORY(I),MT
      IF(J+1.EQ.NSTICK(M)) GO TO 50
910 CONTINUE
      J=J+4
      GO TO 905
1200 DECODE(80,9010,HOL)MT,(SYMDIS(MT,J),J=1,7)
9010 FORMAT(10X,15,7F5.2)
      GO TO 50
1300 DECODE(80,9010,HOL)IWT,(SYMDDF(IWT,J),J=1,7)
      GO TO 50
1400 DECODE(80,9010,HOL)IWT,(SYMDIF(IWT,J),J=1,7)
      GO TO 50
1500 DECODE(80,9011,HOL)MT,SECON(MT),SECOFF(MT),KOUNT(MT),IRNK(MT)
9011 FORMAT(10X15,2F5.2,2I5)
      INTIME(MT)=1
      GO TO 50
1600 DECODE(80,9094,HOL)NUMDEF,IWTDEF(NUMDEF),XODEF(NUMDEF),
      *YODEF(NUMDEF),(NDEFEA(J,NUMDEF),J=1,15)
9094 FORMAT(5X,2I5,2F10.2,15I2)
      GO TO 50
1700 DECODE(80,9093,HOL)INDF,ROAD,(XYMS(I),I=1,4)
9093 FORMAT(5X,2I5,4F10.2)
      L=0
      DO 1705 I=1,2
      ND=ND+1
      IXMI=FIX(XYMS(I+L)*100.)
      XRDF(ND)=FLOAT(IXMI+1SIGN(I,IXMI)*ROAD)
      IXM2=FIX(XYMS(I+2)*100.)
      YRDF(ND)=FLOAT(IXM2+1SIGN(I,IXM2)*ND)
      L=L+1
1705 CONTINUE
      GO TO 50
1800 DECODE(80,9016,HOL)I1,J1,PHD(I1,J1),I2,J2,P2,I3,J3,P3,
      *I4,J4,P4,I5,J5,P5
9016 FORMAT(5X,5(2I5,F5.2))
      IF(I2.LT.1) GO TO 50
      PHD(I2,J2)=P2
      IF(I3.LT.1) GO TO 50
      PHD(I3,J3)=P3
      IF(I4.LT.1) GO TO 50
      PHD(I4,J4)=P4
      IF(I5.LT.1) GO TO 50
      PHD(I5,J5)=P5
      GO TO 50
1900 DECODE(80,9017,HOL)I1,J1,N1,EVI,I2,J2,N2,EV2,I3,J3,N3,EV3
9017 FORMAT(5X,3(3I5,F10.2))
      EI(I1,J1,1)=N1
      EI(I1,J1,2)=EVI
      IF(I2.LT.1) GO TO 50
      EI(I2,J2,1)=N2
      EI(I2,J2,2)=EV2
      IF(I3.LT.1) GO TO 50
      EI(I3,J3,1)=N3
      EI(I3,J3,2)=EV3
      GO TO 50
2000 DECODE(80,9050,HOL) NDFTYP,DEFL(NDFTYP),DEFW(NDFTYP),
      *DEFRAD(NDFTYP),DEFHT(NDFTYP),TMBRD(NDFTYP),NRAD(NDFTYP),
      *(ITGTPR(NDFTYP,1),I=1,5)
9050 FORMAT(5X,15,4F10.2,F5.2,15,5I1)
      GO TO 50
2100 DECODE(80,9051,HOL) I1,A11,AREP1,ADEP1,I2,A12,AREP2,ADEP2
9051 FORMAT(5X,15,3F10.2,5X,15,3F10.2)
      A1(I1)=COS(A11*.01745329)
      AREP(I1)=AREP1
      ADEP(I1)=ADEP1
      IF(I2.LT.1) GO TO 50
      A1(I2)=COS(A12*.01745329)

```

Figure 30. (Continued)

```

AREP(12)=AREP2
ADEP(12)=ADEP2
GO TO 50
2200 DECODE(80,9052,HOL1,1,J,K,N,AEV,AREL1,ACEP1
9052 FORMAT(5X,4I5,F10.2,F5.2,F10.2)
AE(1,J,2*K-1)=N
AE(1,J,2*K)=AEV
L=1
IF(K.GT.1) L=J
AREL(L,K)=AREL1
IF(N.GT.1) ACEP(L,K)=ACEP1
GO TO 50
2300 DECODE(80,9053,HOL1,1,J1,K1,1DP1,12,J2,K2,1DP2,13,J3,K3,1DP3
9053 FORMAT(5X,3(4I5,5X))
1DP(11,J1,K1)=1DP1
IF(12.LT.1) GO TO 50
1DP(12,J2,K2)=1DP2
IF(13.LT.1) GO TO 50
1DP(13,J3,K3)=1DP3
GO TO 50
2400 DECODE(80,9029,HOL1,LCFTT,UBFAE,DTFAE,RADFAE,ALNGLC,
AWIDLC,TMDLCF,POLCF
9029 FORMAT(5X,15,7F10.2)
GO TO 50
6900 DECODE(80,9008,HOL1,NT,NM,NIB,(PROBTP(1),1-1,B)
9008 FORMAT(1X,11,12,11,F5.2,7F10.2)
READ(4,9008) NT,NM,NIB,(RANGTP(1),1-1,B)
NTABLE=(NT-1)*2+(NM-1)*3+NTB
LL=B*(NTABLE-1)
DO 6910 I=1,B
PROBIL(LL+I)=PROBTP(I)
6910 RANGPR(LL+I)=RANGTP(I)
GO TO 50
C
C PERFORM INITIAL CALCULATIONS
C
7000 DO 7005 I=1,NMT
XCycle(I)=SECON(I)+SECOFF(I)
NMINI(I)=0
DO 7005 J=1,NAP
IF(MTFA(J)EQ.1) NMINI(I)=NMINI(I)+NSTICK(J)
7005 CONTINUE
IRTFIR=0
DO 7006 I=1,5
IF(NRAI(I).LT.1) GO TO 7006
IRTFIR=1
7006 NITBT(I)=0
NGTAWT=0
DO 7008 I=1,NTGO
N=NTGTYP(I)
NGTAWT=NGTAWT+NGTAW(I)
7008 NITBT(N)=NITBT(N)+1
IF(SYMP(1).LT.1) GO TO 7015
SYMMAX(1)=0
DO 7010 I=1,NMT
DO 7010 J=1,7
SYMDIS(I,J)=SYMDIS(I,J)+2
7010 SYMMAX(1)=AMAX1(SYMMAX(1),SYMDIS(I,J))
SYMMAX(1)=SORT(SYMMAX(1))
7015 RSPMAX=0
IF(SYMP(2).LT.1) GO TO 7017
SYMMAX(2)=0
DO 7016 I=1,5
DO 7016 J=1,7
SYMDDF(I,J)=SYMDDF(I,J)+2
7016 SYMMAX(2)=AMAX1(SYMMAX(2),SYMDDF(I,J))
SYMMAX(2)=SORT(SYMMAX(2))
7017 IF(SYMP(3).LT.1) GO TO 7019
SYMMAX(3)=0
DO 7018 I=1,5
DO 7018 J=1,7
SYMDIF(I,J)=SYMDIF(I,J)+2

```

Figure 30. (Continued)

```

7018 SYMMAX(3)=AMAX1(SYMMAX(3),SYMDF(I,J))
SYMMAX(3)=SORT(SYMMAX(3))
7019 IF(NMT.LT.1) GO TO 7045
DO 7040 I=1,NTGTP
RSP(I)=0.
DO 7035 J=1,NMT
NT=(I-1)*21+(J-1)*3
DO 7030 K=1,3
NTI=8*(NT+K-1)
DO 7070 L=1,8
IF(PROBIL(NTI*L).EQ.0.) GO TO 7022
7070 CONTINUE
7022 RNGSV(K)=RANGPR(NTI*L)
7030 CONTINUE
PRSHDD(I,J)=RNGSV(1)
PRKLD(I,J)=RNGSV(2)
PRDTND(I,J)=RNGSV(3)
RSP(I)=AMAX1(RSP(I),RNGSV(1),RNGSV(3))
RSPMAX=AMAX1(RSPMAX,RNGSV(1),RNGSV(3))
7035 CONTINUE
IF(MADIV.GT.0) RSP(I)=RSP(I)+30.
7040 CONTINUE
IF(MADIV.GT.0) RSPMAX=RSPMAX+30.
7045 XMIN=XMAX=0.
DO 7060 I=1,5
7060 NIDBT(I)=0
NDFWT=0
IF(NDFWD.LT.1) GO TO 7085
DO 7075 I=1,NDFWD
N=1*WDEF(I)
7075 NIDBT(N)=NIDBT(N)+1
DO 7080 I=1,5
IF(NIDBT(I).GT.0) NDFWT=NDFWT+1
7080 CONTINUE
7085 CONTINUE
DO 7100 I=1,NTGO
XMAX=AMAX1(XMAX,TOTXOL(I))
7100 XMIN=AMIN1(XMIN,TOTXOL(I))
XMAX=XMAX+RSPMAX
XMIN=XMIN-RSPMAX
IF(XMAX.GT.1638.) XMAX=1638.
IF(XMIN.LT.-1638.) XMIN=-1638.
IF(NVAP.LT.1) GO TO 9998
DO 8200 I=1,NVAP
COSA=COS(AYYVP(I)*.01745329258)
SINA=SIN(AYYVP(I)*.01745329258)
XORGVP=FLOAT(FIX(XOVP(I)/100.)/100.)
YORGVP=YOVP(I)
NHV=NHEPV(I)
IF(NHV.LT.1100 TO 8200
DO 8190 J=1,NHV
XOMP(I,J)=XORGVP+DMP11X(I,J)*COSA+DMP11Y(I,J)*SINA
YOMP(I,J)=YORGVP+DMP11Y(I,J)*COSA-DMP11X(I,J)*SINA
8190 CONTINUE
8200 CONTINUE
DO 8500 I=1,NTGTP
AM=AMAX1(TARL(I),TARH(I))
DO 8500 J=1,NVAP
K=INTVAP(I)
THRSIG(K,I)=14.44*AMAX1(REP(I),DEP(I))*AM*SORT(EI(K,1,2))**.2
8500 CONTINUE
9998 RETURN
1480 FORMAT(IH)
END

```

Figure 30. (Continued)

```

SUBROUTINE SETUP
C
C
C RETURN POINT TO BEGIN EACH MONTE-CARLO ITERATION
COMMON ACEP(5,2), ADEP(20), AEI(5,5,4), AI(20), AIMPTX(50), AIMPTI(50),
*ALNGLC, ANGLMP(10), AREL(5,2), AREP(20), AMIDLC, AYYVP(10), DBFAE,
*DEPHT(5), DEFL(5), DEFRAD(5), DEFW(5), DEFY(20), DELATH(100),
*DEP(10), DFITFAE, DIRATK(10), DIRDIS(100), DMP11X(10,10), DMP11Y(10,10),
*DUDPRB(7), DDEL, EI(5,5,2), ID, IDISOP, IDP(20,15,2), JEND,
*IS, IMINE, INDEFA, INTIME(7), IOB(5000), IPO, IPRINT, IRNK(7), IRTFIR,
*IRUNS, ISAVE, ISBL, ISVIFA(30), ISVLST, ISYMP(3), ISYP, IT, ITEROP,
*ITGTPR(5,5), IVAP, IVEL, IWORD, INTDEF(20), INTVAP(10),
COMMON JMARKI, JSELDS(7), KABOOM(547), KDFTTN(100), KNRS, KOUNT(7),
*KTSAV(100), KI, LCFOPT, LCFIT, LOFDPR(5,5), LEFTIN, MADIV, MIFBT(7),
*MODE, MFA(50), MUSH, NAP, NCTAW(100), ND, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFHD, NDFHT, NDVT, NGTAW(100), NGTAWT, NGTAWI(100),
*NIDBT(5), NITBT(5), NKILL(5), NKILLD(5), NKILLI(5), NKILLT, NMOET(7),
*NMEN(7), NMSWPT(7), NMT, NOB, NOBEVT, NOBTP2, NOI1, NOSTAT, NRAD(5),
*NRADFD(20), NRADFI(100), NRAI(5), NRBA, NREBDT(5), NREBIT(5),
*NRFIRD(100), NRS, NSPLFT, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGTYP(100), NTLCFI(10), NTLOST, NTTCS(5), NTTTHD(5), NTTTRP(5), NVAP,
*NVFAEA(10), NVLIDF(10), NHEPV(10), OBX, OBY, OBYPR, PATRAD(5), PDLCF,
COMMON PHD(5,5), PPEAF(7), PRBITY, PROTHC(5,7), FRKLU(5,7),
*PROBIL(8,0), PRSWDD(5,7), RADFAE, RANGPR(840), RELRND(5), RELSUB(5),
*REP(10), RN, ROTDMPX(10,10), ROTDMPY(10,10), RSP(5), S(100), SECON(7),
*SIGAD(50), SIGAR(50), SIGBD(50), SIBBR(50), STATA(5,5), SWPDEL(100),
*SYRDF(5,7), SYMDIF(5,7), SYMDIS(7,7), SYMMAX(3), TARHT(5), TARL(5),
*TARRAD(5), TARRW(5), TDBIFV(10), TGTDEL(100), TGTQVL, TGTSPD,
*TGTVL(100), TGTVL2, TGTVSV, TGTVOL(100), TGTYNH(100), TGTVOL(100),
*THSWP(7), THTOFR(130), TOTSIM, TRAVTM, XCYLE(7), XDMR(10,10), XFIF,
*XMAX, XMIN, XODEF(20), XOVPI(10),
*XRDF(130), XROAD(11), XSWATH(50), XWIDTH, YDIV(100), YDMP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRDF(30), YROAD(11), YSWATH(50)
C....
IT = 0
TGTSPD=TGTVL2
IFIMODE.LT.3ITGTSPD=TGTQVL
REWIND 1
REWIND 2
IRUNS=IRUNS+1
DO 300 I=1,5
NKILLI(I)=NKILLD(I)+NREBIT(I)+NREBDT(I)+NDFDAM(I)=0
300 NKILL(I)=0
DO 305 I=1, NVAP
NVLIDF(I)=NVFAEA(I)
305 CONTINUE
DO 310 I=1,7
NMSWPT(I)=NMOET(I)+MIFBT(I)=0
DO 315 J=1,20
DO 315 I=1,15
315 NDEFEA(I,J)=ABS(NDEFEA(I,J))
DO 320 J=1,15
ISVIFA(I+15)=0
320 ISVIFA(I)=0
DO 325 J=1,130
THTOFR(I,J)=9990
DO 330 I=1,100
KDFTTN(I)=0
NRFIRD(I)=0
330 DELATH(I)=0
NKILLT=NSPLFT=0
NTLOST=0
NOBEVT=1
THET=.017453295*THETA
SINT=SIN(THET)
COST=COS(THET)
KNRS=0
XW2=XWIDTH/2.0
YLP=YLENGTH/2.0
DO 210 I=1,547
210 KABOOM(I)=0

```

Figure 30. (Continued)

```

NOB = 0
NOBTP2 = 0
IT = 0
N = NRS + 1
NOB = NOB + N
DO 200 I = 1, N
200 WRITE(1) XROAD(I), YROAD(I), NOBTP2, IT
   IF(NAP.LT.1) GO TO 160
   DO 146 J = 1, NAP
147 CALL RNORM(RN1)
   CALL RNORM(RN2)
   REWIND 2
   MT = MTF(A(J))
   IT = 0
   N = NSTICK(J)
   DO 146 I = 1, N
   CALL RNORM(RN3)
   CALL RNORM(RN4)
   NOBTP2 = MT
   IF(JSELDS(MT)) 120, 130
130 READ(2) ORX, ORY, NOBTP2
   IF(NOBTP2.NE.MT) GO TO 130
   GO TO 150
120 GO TO(122, 125, 125, 122), JSELDS(MT)
122 CALL RNORM(RSTART)
   ORX = XSWATH(J)/6.*RSTART
   IF(JSELDS(MT)-4) 123, 126, 123
123 CALL RNORM(RSTART)
   ORY = YSWATH(J)/6.*RSTART
   GO TO 150
125 ORX = XSWATH(J)*(RANF(DUMMY)-.5)
   IF(JSELDS(MT)-3) 126, 123, 126
126 ORY = YSWATH(J)*(RANF(DUMMY)-.5)
150 ORX = ORX + RN1*SIGAD(J) + RN3*SIGBD(J)
   ORY = ORY + RN2*SIGAR(J) + RN4*SIGBR(J)
   OBX = AIMPTX(J) + ORX*COST + ORY*SINT
   OBY = AIMPTY(J) - ORX*SINT + ORY*COST
   IF(ABS(OBX).GT.XW2.OR.ABS(OBY).GT.YL2) GO TO 146
143 RN = RANF(DUMMY)
   IF(RN-DUDPRB(MT)) 20, 20, 25
   20 NOBTP2 = NOBTP2 + 0
   25 IF(INTIME(MT).EQ.0) GO TO 145
   RN = RANF(DUMMY)
   IT = RN*XCYLE(MT)
   IF(KOUNT(MT).GT.0) IT = KOUNT(MT)
   IF(IRNK(MT).GT.0) IT = IFIX(IT*RN)+1
145 IF(MODE.EQ.1.AND.NOBTP2.GT.8) GO TO 146
   NOB = NOB + 1
   WRITE(1) OBX, OBY, NOBTP2, IT
146 CONTINUE
   IF(NOB.LE.32767) GO TO 160
   PRINT 1000
1000 FORMAT(' MORE THAN 32767 MINES IN MINEFIELD')
   CALL EXIT
   160 NOBTP2 = 0
   WRITE(1) OBX, OBY, NOBTP2, IT
10 MUSH = 0
   TOTSIM = 0.0
   DO 100 I = 1, NTGO
   N = NTGYP(I)
   NRADF(I) = NRADIN
   IF(NTITCS(N).EQ.1) NSPLFT = NSPLFT + 1
99 TIMEHV(I) = 0.
   TOTDEL(I) = 0.
   SWPDEL(I) = 0.
   NGTAW(I) = NGTAW(I)
   TOTVEL(I) = TOTSPD
100 CONTINUE
   DO 110 J = 1, NDFWD
   N = INTDEF(J)
   NRADF(J) = NRADIN
110 CONTINUE
   RETURN
   END

```

Figure 30. (Continued)

```

SUBROUTINE ROAD
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDLC,AYYVP(10),OBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUOPRB(7),D3DEL,EI(5,5,2),ID,DISOP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYHP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IWORD,IWDEF(20),IWTVAP(10)
COMMON JMARK1,JSELD(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KTSAY(100),KI,LCFOPT,LCFTT,LOFDPR(5,5),LEFTIN,MAOIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDEFEA(15,20),NDF,NOFA,
*NDFAA,NDFOAM(5),NDFWD,NDFWT,NDVT,NGTAW(100),NGTAWT,NGTAWI(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLO(5),NKILLI(5),NKILLT,NMOET(7),
*NMIN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADFO(20),NRADFI(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFIRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NWEPV(10),ORX,ORY,OBYPRT,PATRAD(5),PDLCF
COMMON PHD(5,5),PPFAF(7),PRBITTY,PROTNO(5,7),PRKLO(5,7),
*PROBIL(840),PRSHDD(5,7),RADFAE,RANGPR(840),RELAND(5),RELSUB(5),
*REP(10),RN,ROTMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDDF(5,7),SYMDF(5,7),SYMDS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTVSV,TGTXOL(100),TGTYNW(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEHV(100),TMBRD(5),TMBRI(5),TMDLCF,
*TMSWP(7),TMTOFR(130),TOTSIM,TRAVTM,XCYCLE(7),XDMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRDFO(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDFO(30),YROAD(11),YSWATH(50)

```

```

C****
DIMENSION XLCF(100)
NDFA=INDFA+0
REWIND 1
KNRS=KNRS+1
DO 10 I=1,KNRS
10 READ(1) XRI,YRI,NOBTP2,II
   READ(1) XR2,YR2,NOBTP2,II
20 READ(1) ORX,ORY,NOBTP2,II
   IF(NAP.LT.II) GO TO 25
   IF(NOBT2.EQ.8) GO TO 20
25 N2=NOBTP2
   LIT=II
   A1=YR2-YRI
   A2=XR2-XRI
   IF(A2.EQ.0.) A2=.0000001
   THETR=ATAN2(A1,A2)-3.141592654/2
   COSTR=COS(THETR)
   SINTR=SIN(THETR)
   OBX=0.0
   OBY=0.0
   NOBTP2=8
   II=IMINE+0
   CALL IPACK
   IOB(1)=IWORD
   YLENTN=OBY*SQRT(A2**2+A1**2)
   CALL IPACK
   NOB=2
   IOB(2)=IWORD
   NOBTP2=N2
   IT=LIT
   DO 45 I=1,130
45 TMIOFR(I)=9999.0
   YMIN=99999.
   YMAX=0.
   DO 50 IMINE=1,32767
   IF(NOBT2.EQ.0) GO TO 100
   OBX=(ORX-XRI)*COSTR+(ORY-YRI)*SINTR
   OBY=- (ORX-XRI)*SINTR+(ORY-YRI)*COSTR
   IF(OBY.GT.YLENTN.OR.OBY.LT.0.) GO TO 40
   IF(OBX.LT.XMIN.OR.OBX.GT.XMAX) GO TO 40

```

Figure 30. (Continued)

```

YMIN=AMIN(IYMIN, OBY)
YMAX=AMAX(IYMAX, OBY)
IDET=0
IF(KNRS.GT.1) CALL BOOM(IDET,2)
IF(IDET.EQ.1) GO TO 40
CALL IPACK
M=MOD(NOBT2,8)
IF(NOBT2.GT.15) GO TO 29
MIFBT(M)=MIFBT(M)+1
29 NOB=NOB+1
IF(NOBT.LT.5001) GO TO 30
PRINT 1000
1000 FORMAT('MORE THAN 5000 MINES WITHIN RANGE OF INFLUENCE FOR A RDA
'D SEGMENT'/20(1H/), 'PROGRAM STOPS',20(1H/))
CALL EXIT
30 IOB(NOBT)=IWORD
40 READ(1) ORX,ORY,NOBT2,IT
50 CONTINUE
100 DO 120 I=1,NTGO
KTSAV(I)=1
IF(KNRS.EQ.1) GO TO 110
IF(TGTYNW(I).GT.99994.) GO TO 120
110 TGTYNW(I)=TGTYOL(I)
TGTVEL(I)=TGTSPD
IF(KNRS.EQ.1) GO TO 120
TADJ=ABS(TGTYNW(I)/TGTVEL(I))
DELTM=TGTDEL(I)*SWPDEL(I)
TIMEMV(I)=TIMEMV(I)-TADJ-DELTM
TGTYNW(I)=TGTYNW(I)-TGTVEL(I)*DELTM
120 CONTINUE
TGTYSV=TGTYOL(I)
LEFTIN=NTGO-NKILLT-NTLOST
MUSH=0
NOVT=0
DO 121 I=1,100
121 S(I)=0.
ISVLST=1
KTSAV(I)=0
IF(NVAP.LT.1) GO TO 135
IMINE=IT=0
NOBT2=16
DO 130 J=1,NVAP
DIRATK(J)=AYYVP(J)*.017453293-THETR
IRD=AMOD(XOVP(J),100.)
IROAD=ABS(IRD)
IF(IROAD.NE.KNRS) GO TO 130
IT=NTLCIF(J)
IMINE=J
XR=(XOVP(J)-IROAD)/10000.
YR=YOVPI(J)
OBX=(XR-XRI)*COSTR+(YR-YRI)*SINTR
OBY=-(XR-XRI)*SINTR+(YR-YRI)*COSTR
IF(OBY.GT.YLENTH.OR.OBY.LE.0.) GO TO 125
IF(OBX.LT.-1638..OR.OBX.GT.1638.) GO TO 125
CALL IPACK
NOB=NOB+1
IF(NOBT.LT.5001) GO TO 124
PRINT 1000
CALL EXIT
124 IOB(NOBT)=IWORD
125 CONTINUE
130 CONTINUE
DO 134 J=1,NVAP
NWV=NWEPV(J)
IF(NWV.LT.1) GO TO 134
DO 133 K=1,NWV
ROTDMPX(J,K)=(XDMP(J,K)-XRI)*COSTR+(YDMP(J,K)-YRI)*SINTR
ROTDMPY(J,K)=-(XDMP(J,K)-XRI)*SINTR+(YDMP(J,K)-YRI)*COSTR
133 CONTINUE
134 CONTINUE
135 IMINE=0
IT=0

```

Figure 30. (Continued)

```

L=0
IF(ND.LT.1) GO TO 150
DO 150 I=1,ND
!RD=AMOD(XRDFO(I),100.)
IRD=ABS(IRD)
IF(IRD.NE.KNRS) GO TO 150
XR=FLOAT(IFIX(XRDFO(I)/100.))
YR=FLOAT(IFIX(YRDFO(I)/100.))
OBY=-(XR-XRI)*SINTR+(YR-YRI)*COSTR
IF(OBY.GT.YLENTH OR OBY.LE.O.) GO TO 150
NOBIP2=17
IF(L)149,147
147 L=1
IT=IFIX(AMOD(YRDFO(I),100.))
IT=ABS(IT)
148 CALL IPACK
NOB=NOB+1
IF(NOB.LT.500) GO TO 140
PRINT 1000
CALL EXIT
149 NOBIP2=18
L=0
GO TO 140
140 IOB(NOB)=IWORD
150 CONTINUE
160 IF(NDFAA.LT.1) GO TO 400
DO 350 I=1,NOFHD
DEFX(I)=(XODEF(I)-XR)*COSTR+(YODEF(I)-YRI)*SINTR
DEFY(I)=-(XODEF(I)-XR)*SINTR+(YODEF(I)-YRI)*COSTR
CONTINUE
350
400 IF(LCFORT.LT.1)RETURN
NLCFCT=0
DO 410 I=1,NTGO
IF(NTGTYP(I).NE.LCFIT) GO TO 410
NLCFCT=NLCFCT+1
XLCF(NLCFCT)=TOTXO(I)
410 CONTINUE
DLCF=ALNGLC
IF(LCFORT.GT.1)DLCF=OBFAE
IT=0
IMINE=0
NOBIP2=19
OBY=YMIN-2.*DLCF
OBY=AMAX(OBY,O.)
420 OBY=OBY+DLCF
IF(OBY.GT.YMAX) GO TO 500
DO 430 I=1,NLCFCT
OBX=XLCF(I)
CALL IPACK
NOB=NOB+1
IOB(NOB)=IWORD
IF(NOB.LT.500) GO TO 430
PRINT 1000
CALL EXIT
430 CONTINUE
GO TO 420
500 RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE BOOM(IDET,N)
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNOLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDL,AYVVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATN(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUOPRB(7),D3DEL,EI(5,5,2),ID,IDISOP,IDP(20,15,2),IEND,
*IMS,IMINE,INDFA,INTINE(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,WORD,INTDEF(20),INTVAP(10)
COMMON JMARKI,USELDS(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KTSAVE(100),KI,LCFOPT,LCFTT,LDFDPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDEFEA(15,20),NDF,NDFA,
*NDFAA,NDFAM(5),NDFHD,NDFMT,NDVT,NGTAN(100),NGTAM,NGTAW(100),
*NTDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMIN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF1(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFIRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTRP(5),NYAP,
*NVFAEA(10),NVLIDF(10),NHEPV(10),OBX,OBJ,OBJPRT,PATRAD(5),POLCF
COMMON PHD(5,5),PPEAF(7),PRBITY,PRDINO(5,7),PRKLO(5,7),
*PROBIL(840),PRSHDO(5,7),RADFAE,RANGPR(840),RELRND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTXOL(100),TGTYNW(100),TGTYQL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEHV(100),TMORD(5),TMBRI(5),TMDLCF,
*TMSWP(7),TMTOPR(130),TOTSIM,TRAVTH,XCYCLE(7),XDMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVPI(10),
*XRDFO(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDFO(30),YROAD(11),YSWATH(50)
C****
IWD=(IMINE-1)/50+1
K=MOD(IMINE,50)+1
IF(K.EQ.-1) K=59
IF(IN GT 1) GO TO 200
100 I=1
KABOOM(IWD)=KABOOM(IWD).OR.SHIFT(I,K)
RETURN
200 IDET=SHIFT(KABOOM(IWD),K).AND.IB
RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE SORT(A,N)
.....
SUBROUTINE SORT
PURPOSE
  TO SORT A VECTOR INTO INCREASING ORDER FROM A(1) TO A(N).
  A MAY BE TYPE REAL OR TYPE INTEGER.
USAGE
  CALL SORT(A,N)
DESCRIPTION OF PARAMETERS
  A   - THE NAME OF THE N-VECTOR TO BE SORTED
      - IF A IS TYPE REAL THEN EACH OF ITS COMPONENTS
      - MUST BE IN NORMALIZED FORM.
  N   - THE NUMBER OF ELEMENTS IN THE VECTOR TO BE SORTED.
REMARKS
  THE PROCEDURE REQUIRES TWO ADDITIONAL ARRAYS IU(K) AND IL(K)
  WHICH PERMIT SORTING UP TO 2**((K+1)-1) ELEMENTS. THESE
  ARRAYS ARE SUPPLIED BY THE SUBROUTINE WITH K=16. THIS
  ALLOWS SORTING A MAXIMUM OF 131071 ELEMENTS.
METHOD
  AN EFFICIENT ALGORITHM FOR SORTING WITH MINIMAL STORAGE
  BY RICHARD C. SINGLETON
  PREPARED FOR
  INSTITUTE RESEARCH AND DEVELOPMENT
  STANFORD RESEARCH INSTITUTE PROJECT 387531-132
  SEPTEMBER 1968.
.....
INTEGER A(N),IU(16),IL(16),I,TT
M = 1
I = 1
J = N
5  IF(I-J) 10,70,70
10 K = I
   IU = (J+1)/2
   T = A(IJ)
   IF(A(I)-T) 20,20,15
15  A(IJ) = A(I)
   A(I) = T
   T = A(IJ)
20  L = J
   IF(A(J)-T) 23,40,40
23  A(IJ) = A(J)
   A(J) = T
   T = A(IJ)
   IF(A(I)-T) 40,40,27
27  A(IJ) = A(I)
   A(I) = T
   T = A(IJ)
   GO TO 40
30  A(L) = A(K)
   A(K) = TT
40  L = L - 1
   IF(A(L)-T) 35,35,40
35  TT = A(L)
50  K = K + 1
   IF(A(K)-T) 50,53,53
53  IF(K-L) 30,30,55
55  IF((L-1)-(J-K)) 60,60,57
57  IL(M) = I
   IU(M) = L
   I = K
   M = M + 1
   GO TO 80
60  IL(M) = K

```

Figure 30. (Continued)

```

IU(M) * J
J = L
M = M + 1
GO TO 80
70 M = M - 1
IF(M) 75,110,75
75 I = IL(M)
J = IU(M)
80 IF(J-I-1) 85,10,10
85 IF(I-1) 87,5,87
87 I = I - 1
90 I = I + 1
IF(I-J) 93,70,93
93 T = A(I+1)
IF(A(I)-T) 90,90,97
97 K = I
100 A(K+1) = A(K)
K = K - 1
IF(T-A(K)) 100,105,105
105 A(K+1) = T
GO TO 90
110 RETURN
END

```

Figure 30. (Continued)

SUBROUTINE LOOPS

```

COMMON ACEP(5,2), ADEP(20), AE(15,5,4), AI(20), AIMPTX(50), AIMPTY(50),
*ALNGLC, ANGIMP(10), AREL(5,2), AREP(20), AWIOLC, AYYVP(10), DBFAE,
*DEFHT(5), DEFL(5), DEFRA(5), DEFW(5), DEFX(20), DEFFY(20), DELATM(100),
*DEP(10), DFTFAE, DIRATK(10), DIRDIS(100), DMP1IX(10,10), DMP1IY(10,10),
*DUDPRB(7), D3DEL, E(15,5,2), ID, IDISOP, IDP(20,15,2), IEND,
*IIS, IMINE, INDA, INTIME(7), IOB(5000), IPO, IPRINT, IRNK(7), IPTFIR,
*IRUNS, ISAVE, ISBL, ISVIFA(30), ISVLST, ISYMP(3), ISYP, IT, ITEROP,
*ITGTPR(5,5), IVAP, IVEL, IHORO, IHTDEF(20), IHTVAP(10),
COMMON JMARKI, JSELDS(7), KABOOM(547), KOFFTN(100), KNRS, KOUNT(7),
*KTSAVI(100), KI, LCFOP, LCFIT, LDFOPR(5,5), LEFTIN, MADIV, MIFBT(7),
*MODE, MIFA(50), MUSH, NAP, NCTAW(100), ND, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFHD, NDFMT, NDVT, NGTAW(100), NGTAWT, NGTAWI(100),
*NIDBT(5), NIIBT(5), NKILL(5), NKILLD(5), NKILLI(5), NKILLT, NMDET(7),
*NMIN(7), NMSWPT(7), NMT, NOB, NOBEVT, NOBTP2, NOIT, NOSTAT, NRAD(5),
*NRADFD(20), NRAOFL(100), NRAI(5), NRBA, NREBTD(5), NREBIT(5),
*NRFIRD(100), NRS, NSPLFT, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGTYP(100), NTLCLF(10), NTLST, NTTTCS(5), NTTTMD(5), NTTTRP(5), NVAP,
*NVFAEA(10), NVLIDF(10), NWEPV(10), OBX, OBY, OBYPR, PATRAD(5), PDLCF,
COMMON PHD(5,5), PPEAF(7), PRBITY, PROTN(5,7), PRKLO(5,7),
*PROBIL(840), PRSWDD(5,7), RADFAE, RANGPR(840), RELRND(5), RELSUB(5),
*REP(10), RN, ROTDMPX(10,10), ROTDHPY(10,10), RSP(5), S(100), SECON(7),
*SIGAD(50), SIGAR(50), SIGBD(50), SIGBR(50), STATAL(5,52), SWPOEL(100),
*SYMDDF(5,7), SYMDF(5,7), SYMDS(7,7), SYMHAX(3), TARHT(5), TARL(5),
*TARRAD(5), TARW(5), TDBIFV(10), TGTDEL(100), TGTQVL, TGTSPD,
*TGTVEL(100), TGTVL2, TGTVSV, TGTXL(100), TGTYNW(100), TGTYOL(100),
*TGTVSV, THETA, THRSIG(5,5), TIMEMV(100), THORD(5), TMBRI(5), TMDLCF,
*TMSWP(7), TMTOPR(130), TOTSIH, TRAVTH, XCYLE(7), XDMP(10,10), XFTX,
*XMAX, XMIN, XODEF(20), XOVPR(10),
*XRDFO(30), XROAD(11), XSWATH(50), XWIDTH, YDIV(100), YDMP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRDFO(30), YROAD(11), YSWATH(50)
C****
DIMENSION IPOS(100), TM(100)
KTSAVI(15VLST)=KTSAVI(15VLST)+1
DO 250 I=1,NTGO
IF(TGTYNW(1)=10000,1110,110,100
100 TGTYNW(1)=99999.
TMTOPR(1)=9990.
DIRDIS(1)=99999.
GO TO 250
110 DIRDIS(1)=99999.
IF(TGTYNW(1) GT 99970) GO TO 250
140 KI=KTSAVI(1)
IF(10B(KI)+1) 210,205
205 KTSAVI(1)=KTSAVI(1)+1
GO TO 140
210 IHORO=10B(KI)
CALL UNPACK
N=NTGTYP(1)
IF(NOBT(2) GT 15) GO TO 240
M=MOD(NOBT(2),8)
IF(M) 212,240
212 IF(PRSWDD(N,M)) 220,215
215 IF(PROTN(N,M)) 220,205
220 XDIS=ABS(OBX-TGTXL(1))
IF(XDIS-RSP(N)) 240,205,205
240 DIRDIS(1)=OBY-TGTYNW(1)
250 CONTINUE
ISAVE=1
IVEL=0
ISV=1
SMLDIS=SMLDIS+99999.
DO 400 I=1,NTGO
IF(SMLDIS-DIRDIS(1)) 350,350,310
310 ISAVE=1
SMLDIS=DIRDIS(1)
350 IF(TGTVEL(1)) 360,400,360
360 IF(SMLDIS-DIRDIS(1)) 400,400,380
380 SMLDIS=DIRDIS(1)
ISV=1
IVEL=1

```

Figure 30. (Continued)

```

400 CONTINUE
   TRAVM=SMLDIS/TGTSPD
   ISVLST=ISAVE
   IF (TGTVEL (ISAVE) .GT. 0.) RETURN
   IF (LEVEL.LT.1) GO TO 580
   TT=SMLDIS/TGTSPD
   IF (TT.LE.DELATM (ISAVE)) GO TO 570
   TRAVM=DELATM (ISAVE)
   RETURN
570 ISAVE=ISV
   ISVLST=ISV
   TRAVM=TT
   RETURN
580 DO 590 I=1,NTGO
   TM(I)=DELATM(I)
   IPOS(I)=1
   IF (TGTYNW(I) .GT. 99990.) TM(I)=99999.
590 CONTINUE
   DO 700 I=1,NTGO
   IF (I.LT.2) GO TO 600
   IF (TM(I-1) .GT. TM(I)) GO TO 710
600 K=1
   DO 700 J=1,NTGO
   IF (TM(I) .LT. TM(J)) GO TO 700
   SAVE=TM(I)
   TM(I)=TM(J)
   TM(J)=SAVE
   KEEP=IPOS(I)
   IPOS(I)=IPOS(J)
   IPOS(J)=KEEP
700 CONTINUE
710 NUM=K-1
   DIS=99999.
   DO 750 I=1,NUM
   KPOS=IPOS(I)
   IF (DIRDIS (KPOS) .GT. DIS) GO TO 750
   DIS=DIRDIS (KPOS)
   ISAVE=KPOS
750 CONTINUE
   ISVLST=ISAVE
   TRAVM=TM(I)
   RETURN
END

```

Figure 30. (Continued)

SUBROUTINE EVENTC

```

COMMON ACEP(5,2), ADEP(20), AE(5,5,4), AI(20), AIMPTX(50), AIMPTY(50),
*ALNGLC, ANGLMP(10), AREL(5,2), AREP(20), AWIDLC, AYYP(10), DBFAE,
*DEFHT(5), DEFL(5), DEFRAD(5), DEFW(5), DEFIX(20), DEFY(20), DELATH(100),
*DEPI(10), OFYFAE, DIRATK(10), DIRDIS(100), DMP1IX(10,10), DMP1IY(10,10),
*DUDPRB(7), D3DEL.EI(5,5,2), ID, IDISOP, IOP(20,15,2), IEND,
* IIS, IMINE, INDFA, INTIME(7), IOB(5000), IPO, IPRINT, IRNK(7), IRTFIR,
*IRUNS, ISAVE, ISBL, ISVIFA(30), ISVLS1, ISYMP(3), ISYP, IT, ITEROP,
*ITGTPR(5,5), IVAP, IVEL, IWORD, IWTDEF(20), IWTYAP(10)
COMMON JMARK1, JSELDS(7), KABOOM(547), KOFFTN(100), KNRS, KOUNT(7),
*KTSAV(100), KI, LCFOPT, LCFIT, LDFOPR(5,5), LEFTIN, MADIV, MIFBI(7),
*MODE, MIFA(50), MUSH, NAP, NCTAW(100), ND, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFHD, NDFHT, NDVT, NGTAH(100), NGTAHT, NGTAHI(100),
*NDTBT(5), NITBT(5), NKILL(5), NKILLO(5), NKILLI(5), NKILLT, NMDET(7),
*NMINI(7), NMSWPT(7), NMT, NOB, NOBEVT, NOBTP2, NOIT, NOSTAT, NRAD(5),
*NRADFD(20), NRADFI(100), NRAI(5), NRBA, NRFBDT(5), NRFBIT(5),
*NRFRD(100), NRS, NSPLFT, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGTYP(100), NTLCIF(10), NILOST, NITTC(5), NITTD(5), NITTRP(5), NVAP,
*NVFAD(10), NVLIDF(10), NWEPY(10), OBY, OBY, OBYPR, PATRAD(5), POLCF
COMMON PHD(5,5), PPEAF(7), PRBITY, PRDTNO(5,7), PRKLO(5,7)
*PROBIL(840), PRSHD(5,7), RADFAE, RANOPR(840), RELRND(5), RELSUB(5),
*REP(10), RN, ROTOMPX(10,10), ROTOMPY(10,10), RSP(5), S(100), SECON(7),
*SIGAD(50), SIGARI(50), SIGBD(50), SIGBR(50), STATAL(5,52), SWPDEL(100),
*SYMODF(5,7), SYMOTF(5,7), SYMOTS(7,7), SYHMAX(3), TARHT(5), TARR(5),
*TARRAD(5), TARRH(5), TDBIFV(10), TGTDEL(100), TGTQVL, TGTSPD,
*TGIVEL(100), TGTVL2, TGTVSV, TGTVOL(100), TGTYNH(100), TGTVOL(100),
*TGIVSV, THEFA, THRSIG(5,5), TIMEMV(100), TMBRD(5), TMBRI(5), THDLCF,
*TMSP(7), TMTOFR(130), TOTSIM, TRAVTM, XCYCLE(7), XOMP(10,10), XFIX,
*XMAX, XMIN, XODEF(20), XOV(10),
*XROFD(30), XROAD(11), XSWATH(50), XWIDTH, YOLV(100), YDMP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRFD(30), YROAD(11), YSWATH(50)

```

C\*\*\*\*

```

TGIVSV = TGIVEL (ISAVE)
OBYPR = OBY
OBY = 99999.
IOB(1) = -1
MUSH = 1
IF (KNRS, NE, 1) GO TO 1000
TOTSIM = 0.
DO 200 I = 1, NTGO
200 TIMEMV(I) = 0.
GO TO 1000
505 CONTINUE
TGIVSV = TGIVEL (ISAVE)
LEFTIN = LEFTIN - 1
502 TGIVSV (ISAVE) = 99999.
IF (INDFA, LT, 1) GO TO 511
TMTOFR (ISAVE) = 9990.
DO 510 J = 1, NDFA
CALL STINT (ISVIFA, ISAVE, J, IBIT, KOFFTN, 2)
510 CONTINUE
511 IF (KNRS, NE, NRS) GO TO 512
N = NTGTYP (ISAVE)
IF (INTTC(S(N), EQ, 1) NSPLFI = NSPLFT - 1
512 TGIVEL (ISAVE) = 0
1000 RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE DIVSET
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AMIDL,AYVVP(10),OBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFFY(20),DELATH(100),
*DEP(10),DFFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUOPRB(7),D3DEL,EL(5,5,2),ID,IDISOP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYMP(3),ISYP,IT,ITEROP,
*IIGTPR(5,5),IVAP,IVEL,INORD,IHTDEF(20),IHTVAP(10)
COMMON JMARKI,JSELDS(7),KABOOM(547),KOFFTN(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LOFOPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDEFEA(15,20),NDF,NDFA,
*NDFAA,NDFAM(5),NDFND,NDFMT,NDVT,NGTAW(100),NGTAWT,NGTAWI(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMIN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF1(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFIRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTTND(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NNEPV(10),OBX,OBY,OBYPRT,PATRAD(5),PDLCF
COMMON PHD(5,5),PPEAF(7),PRBITY,PRDINO(5,7),PRKLO(5,7),
*PROBIL(840),PRSDO(5,7),RADFAE,RANGPR(840),RELAND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARM(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTXL(100),TGTYNM(100),TGTVOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEMV(100),TMBRD(5),TMBRI(5),TMOLCF,
*TMSWP(7),INTOFR(130),TOTSIM,TRAVIM,XCycle(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRDF(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YOMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(11),YSWATH(50)
C....
K=KI
IFIXFIX=50,10,100
NDVT=NDVT+1
YDIV=NDVT+TGTYNM(K)
NTCOL(NDVT)=NCTAW(K)
SINDVT=-1
GO TO 500
50 S(10)=1
GO TO 500
100 S(10)=999.
DO 200 I=1,NTGO
IF(TGTYNM(I).GT.99990.) GO TO 200
IF(NTCOL(I).NE.NCTAW(I)) GO TO 200
IF(TGTYNM(I).GT.YDIV(I0)+60.) GO TO 200
TGTDEL(I)=TGTDEL(I)+D3DEL
DELATH(I)=D3DEL
TGTVEL(I)=0.
200 CONTINUE
500 RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE DIVCHK
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDL,AYYVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUDPRB(7),D3DEL,EI(5,5,2),ID,DISOP,IDP(20,15,2),IEND,
*IS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,WORD,WTDEF(20),WTVAP(10)
COMMON JHARK1,JSELD(7),KABOON(547),KDFTTN(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LDFOPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MTFA(50),MUSH,NAP,NCTAW(100),ND,NDEFEA(15,20),NDF,NDFA,
*NDFAA,NDFDAM(5),NDFWD,NDFHT,NDVT,NGTAH(100),NGTAHT,NGTAHI(100),
*NIOBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,MHDET(7),
*NMIN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADFD(20),NRADFI(100),NRAT(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NWEPV(10),OBX,OBJ,OBJPRT,PATRAD(5),POLCF
COMMON PHD(5,5),PPEAF(7),PRBIT,PRDINO(5,7),PRKLO(5,7),
*PROBIL(940),PRSHDD(5,7),RADFAE,RANGPR(940),RELKND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWDEL(100),
*SYHDDF(5,7),SYHDF(5,7),SYHDS(7,7),SYHMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTYOL(100),TGTYNW(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMENV(100),TMBRD(5),TMBRI(5),TMDLCF,
*TMSWP(7),TMTOFR(130),TOTSIM,TRAVTM,XCYLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XDEF(20),XOVP(10),
*XRDF(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(11),YSWATH(50)
C****
K=KI
DO 50 I=1,NDVT
  ID=I
  IF(NCTAW(K).NE.NTCOL(I)) GO TO 50
  T1=TGTYNW(K)
  T2=YDIV(I)+60
  T3=YDIV(I)-60
  IF(T1.LT.T3.OR.T1.GT.T2) GO TO 50
  IF(S(I).EQ.999) GO TO 50
  20 XFIX=S(I)*SQRT(5625-(T1-YDIV(I))**2)-45.0
  GO TO 100
  50 CONTINUE
  100 RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE UNIT
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPIX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AMIOLC,AYYP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFM(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUDPRB(7),D3DEL,EL(5,5,2),ID,IDI5OP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),100(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISOL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IMORD,IMTDEF(20),IMTVAP(10)
COMMON JMARKI,JSELD(5,7),KABOOM(547),KOFFTN(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LOFOPR(5,5),LEFTIN,RADIV,MIFBT(7),
*NODE,MFAA(50),MUSH,NAP,NCTAM(100),ND,NDFEFA(15,20),NDF,NDFA,
*NDFAA,NDFDAM(5),NOFMD,NDFMT,NOVT,NGTAM(100),NGTAMT,NGTAMI(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMIN(7),NMSHPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF1(100),NRAI(5),NRBA,NRF0BT(5),NRF0IT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLCDF(10),NNEPV(10),OBX,OBJ,OBJPRT,PATRAD(5),POLCF
COMMON PHO(5,5),PPEAF(7),PRBITY,PROTNO(5,7),PRKLO(5,7),
*PROBIL(840),PRSHDD(5,7),RADFAE,RANGPR(840),RELKND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SMPDEL(100),
*SYHDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYRMAX(3),TARMT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSFD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTNOL(100),TGTYNM(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEHV(100),THBRD(5),THBRI(5),THDLCF,
*THSWP(7),TMTOPR(130),TOTSIN,TRAVTH,XCYLE(7),XDMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVPI(10),
*XRFO(30),XRROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRFO(30),YROAD(11),YSWATH(50)
C****
DIMENSION NAS(100)
DO 200 K=1,NTGO
IF(TGTYNM(K).LT.100000.) GO TO 200
L=0
DO 10 I=1,NTGO
IF(TGTYNM(I).GT.99999.) GO TO 10
IF(NGTAM(I).NE.K) GO TO 10
L=L+1
NAS(L)=I
10 CONTINUE
IF(L.EQ.0) GO TO 200
KF=KB=0
SMALY=99999.
BIGY=-99999.
DO 50 I=1,NTGO
IF(I.EQ.K) GO TO 50
IF(NCTAM(I).NE.NCTAM(K)) GO TO 50
IF(NTGTYP(I).NE.NTGTYP(K)) GO TO 50
IF(TGTYNM(I).GE.99999.) GO TO 50
IF(TGTYNM(I).GT.TGTYNM(K)-100000.) GO TO 20
IF(TGTYNM(I).LT.BIGY) GO TO 50
KB=I
BIGY=TGTYNM(I)
GO TO 50
20 IF(TGTYNM(I).GT.SMALY) GO TO 50
KF=I
SMALY=TGTYNM(I)
50 CONTINUE
IF(KF.EQ.0) GO TO 100
DO 60 I=1,L
NI=NAS(I)
60 NGTAM(NI)=KF
GO TO 200
100 IF(KB.EQ.0) GO TO 130
YFIX=TGTYNM(K)-BIGY-100000.
DO 110 I=1,L
NI=NAS(I)
TGTYNM(NI)=TGTYNM(NI)-YFIX
110 NGTAM(NI)=KB
GO TO 200

```

Figure 30. (Continued)

```
130 DO 150 I=1,L
    NI=NAS+11
    TGTVM(NI)=99995.
    TMTOFR(NI)=9990
    IF (IRUNS.EQ. IPRINT) PRINT 99999,NI
99999 FORMAT(* TGT LOST *,14)
    TGTVEL(NI)=0.
    LEFTIN=LEFTIN-1
    NTLOST=NTLOST+1
    IF (NDFA.LT.1) GO TO 150
    DO 140 J=1,NDFA
    CALL STINT(I,SVIFA,NI,J,IBIT,KOFTIN,2)
140 CONTINUE
150 CONTINUE
200 CONTINUE
    RETURN
    END
```

Figure 30. (Continued)

```

SUBROUTINE TGTMIN
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIHPT(50),AIHPTV(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AHIDLC,AYYP(10),DBFAE,
*DEFMT(5),DEFL(5),DEFRAD(5),DEFM(5),DEFX(20),DEFY(20),DELATH(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMPITX(10,10),DMPITV(10,10),
*DUDPR(7),D3DEL(1(5,5,2),10,DISOP,1DP(20,15,2),IEND,
*II5,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IPUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTR(5,5),IVAP,IVEL,IMORD,IMTDEF(20),IMTVAP(10)
COMMON JMARKI,JSLEDS(7),KABOOM(547),KOFITN(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LDFOPR(5,5),LEFTIN,RADIV,MIFOT(7),
*MODE,MTFA(50),MUSH,NAP,NCTAM(100),ND,NDEFA(15,20),NDF,NDFA,
*NDFAA,NDFDAM(5),NDFWD,NDFMT,NDVT,NGTAM(100),NGTAMT,NGTAMI(100),
*NIDBT(5),NIDBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF(100),NRAT(5),NRBA,NRFOT(5),NRFBIT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NHCPV(10),OBX,OBJ,OBYPRT,PATRAD(5),POLCF
COMMON PHD(5,5),PPEAF(7),PRBITY,PRDTH(5,7),PRRL(5,7),
*PROBIL(840),PRSHD(5,7),RADFAE,RANGPR(840),RELWND(5),RELSUB(5),
*REP(10),RN,ROTDHPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYHDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARM(5),TARL(5),
*TARRAD(5),TARM(5),TDBIFV(10),TGTDEL(100),TGTVEL,GTSPD,
*GTVEL(100),TGTVEL2,TGTYSV,TGTXL(100),TGTYNM(100),TGTYNM(100),
*TGYSV,THETA,THRSIG(5,5),TIMEHV(100),THORD(5),THOR(5),THOLCF,
*THSWP(7),TMTOPR(130),TOTSIM,TRAVIM,KCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRFO(30),XROAD(11),XSMATH(50),XWIDTH,YDIV(100),YOMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(11),YSMATH(50)
C....
TGTYSV=TGTYNM(ISAVE)
KT=KTSAV(ISAVE)
IPO=1H
OBYPRT=OBJ
XFIX=0
KI=ISAVE
NOBEVT=NOBEVT
IF(NDVT.GT.0) CALL DIVCHK
XFIX=XFIX
DISTPD=ABS(OBX-TGTXL(ISAVE)-XFIX)
N=NTGTYP(ISAVE)
M=MOD(NOBTP2,8)
IF(NTTTRP(N).GT.0) GO TO 649
IF(NOBTP2.GT.15) RETURN
IF(MODE.LT.3) GO TO 200
IF(NTTCS(N).LT.1) GO TO 200
IF(DISTPD.GE.PRSHD(N,M)) GO TO 200
NTABLE=(N-1)*2+(M-1)*3+1
CALL TABINT(DISTPD,NTABLE)
NOBEVT=IABS(NOBEVT)
RN=RANF(DUMMY)
IF(RN.GT.PRBITY) GO TO 200
IPO=5HSHPT
OBY=99987
NMSWPT(M)=NMSWPT(M)+1
CALL BOOM(1DET,1)
IOB(KT)=-1
IF(NOBTP2.LT.8) NMDET(M)=NMDET(M)+1
IF(ISBL.GT.0.AND.(SYMP(1).GT.0) CALL SYHDET
DO 105 I=1,NTGO
IF(TGTYNM(I).GT.99989) GO TO 105
IF(NCTAM(I) NE. NCTAM(ISAVE)) GO TO 105
SWPDEL(I)=SWPDEL(I)+THSWP(M)
DELATH(I)=THSWP(M)
TGTVEL(I)=0
105 CONTINUE
GO TO 700
200 IF(INTIME(M).LT.1) GO TO 649
C
C IF INTERMITTANT TIMER OPTION IS ON (INTIME=1), CHECK IF MINE IS

```

Figure 30. (Continued)

```

C ACTIVE AT THE TIME OF CLOSEST APPROACH
C
IF (KOUNT(M).GT.0) GO TO 649
TOTSIM=TIMEMV(ISAVE)+TOTDEL(ISAVE)+SHPDFL(ISAVE)
649 IF (AMOD(TOTSIM*60.+IT,XCYCLE(M)).GT.SECON(M)) RETURN
CONTINUE
C
C SEE IF MINE DETONATES
C
PRBITY=0.
IF (NOBTP2.GT.7) RETURN
IF (DISTPD.GE.PRDIND(N,NOBTP2)) RETURN
NTABLE=(N-1)*2+(NOBTP2-1)*3+3
CALL TABINT(DISTPD,NTABLE)
NOBEVT=IABS(INOBEVT)
RN=РАНF(DUMMY)
IF (RN.LT.PRBITY) GO TO 650
IF (NTTTRP(N).LT.2) RETURN
C
C PLOWS ONLY SECTION
C
IF (РАНF(DUMMY).LT.PPEAF(M)) RETURN
NMSWPT(M)=NMSWPT(M)+1
IOB(KT)=-1
IPO=SHSWEPT
CALL BOOM(IDET,1)
RETURN
C
C IF A DETONATION OCCURED, REMOVE MINE
C
650 IF (KOUNT(M).LE.0) GO TO 658
IF (IT.LE.1) GO TO 658
IT=IT-1
CALL IPACK
IOB(KT)=IWORD
RETURN
658 NMDET(M)=NMDET(M)+1
OBY = 99972.
IF (ISYP.EQ.0) ISYP=1
CALL BOOM(IDET,1)
IPO=9HDETONATED
IOB(KT)=-1
IF (NTTTRP(N)) 710,659
659 CONTINUE
C
C PROB. OF A KILL
C
DO 690 K=1,NTGO
IF (TGTYNW(K).GT.99969) GO TO 690
KI=K
IF (K.NE.ISAVE) GO TO 660
XFIX=XFIX+1
GO TO 670
660 XFIX=0.
IF (NDVT.GT.0) CALL DIVCHK
670 DISTPD=IOBX-TGXOL(KI)-XFIX)*.2+(OBYPR-TGTYNW(KI))*2
N=NTGTYP(KI)
PK=PRKLD(N,NOBTP2)*.2
IF (DISTPD.GT.PK) GO TO 690
DISTPD=SQRT(DISTPD)
NTABLE=(N-1)*2+(NOBTP2-1)*3+2
CALL TABINT(DISTPD,NTABLE)
RN=РАНF(DUMMY)
IF (RN.GT.PRBITY) GO TO 690
KI=K
IF (NTTTHD(N).GT.0) CALL DIVSET
TGTYNW(KI)=TGTYNW(KI)+100000.
MTOFR(KI)=9990.
TGIVSV=TGIVEL(KI)
TGIVEL(KI)=0.
IF (ISYP.EQ.1) ISYP=2
IF (NTTTC(S(N)).GT.0) NSPLFT=NSPLFT+1

```

Figure 30. (Continued)

```

LEFTIN = LEFTIN - 1
NKILLT = NKILLT + 1
NKILL(N) = NKILL(N) + 1
IF(INDFA.LT.1) GO TO 690
DO 680 J=1,NOFA
CALL STINT(1SVIFA,K,J,1BIT,KDFTN,2)
680 CONTINUE
690 CONTINUE
IF(1SYMP(1).GT.0) CALL SYMDET
700 RETURN
710 KI=ISAVE
C
C ROLLER/PLOW KILL SECTION
C
NHSWPT(M) = NHSWPT(M) + 1
PK = PRKLO(N,NOBTP2)
IF(DISTPD.GT.PK) RETURN
NTABLE = (N-1)*21 + (NOBTP2-1)*3 + 2
CALL TABINT(DISTPD,NTABLE)
RN = RANF(DUMMY)
IF(RN.GT.PRBIT) RETURN
TOTYNW(1SAVE) = TOTYNW(1SAVE) + 100000.
TMTOFR(1SAVE) = 9990.
TGTVSV = TGTVEL(1SAVE)
TGTVEL(1SAVE) = 0.
NSPLFT = NSPLFT - 1
LEFTIN = LEFTIN - 1
NKILLT = NKILLT + 1
NKILL(N) = NKILL(N) + 1
IF(INDFA.LT.1) RETURN
DO 720 J=1,NOFA
CALL STINT(1SVIFA,ISAVE,J,1BIT,KDFTN,2)
720 CONTINUE
RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE SYMDET
COMMON ACEP(5,2), ADEF(20), AEI(5,5,4), AI(20), AIMPTX(50), AIMPTY(50),
*ALNGLC, ANGIMP(10), AREL(5,2), AREP(20), ANIDLQ, AYYVP(10), DBFAE,
*DEFHT(5), DEFL(5), DEFRAD(5), DEFW(5), DEFX(20), DEFY(20), DELATH(100),
*DEP(10), DFTFAE, DIRATK(10), DIRDIS(100), DMPIIX(10,10), DMPIIY(10,10),
*DUDPRB(7), O3DEL, EI(5,5,2), IO, IOISOP, IOP(20,15,2), IEND,
*IIS, IMINE, INDOFA, INTIME(7), IOB(5000), IPO, IPRINT, IRNK(7), IRTFIR,
*IRUNS, ISAVE, ISBL, ISVIFA(30), ISVLST, ISYMP(3), ISYP, IT, ITEROP,
*ITGTPR(5,5), IVAP, IVEL, IWORD, IWTDEF(20), IWTVAP(10)
COMMON JHARKI, JSELD5(7), KABOOM(547), KDFTTN(100), KNRS, KOUNT(7),
*KTSAV(100), KI, LCFOPT, LCFIT, LDFDPR(5,5), LEFTIN, MADIV, MIFST(7),
*MODE, MTF(50), MUSH, NAP, NCTAW(100), ND, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFMD, NDFHT, NDVT, NGTAW(100), NGTAMT, NGTAMI(100),
*NIDBT(5), NITBT(5), NKILL(5), NKILLD(5), NKILLI(5), NKILLT, NMDET(7),
*NMIN(7), NMSWPT(7), NMT, NOB, NOBEVT, NOBTP2, NOIT, NOSTAT, NRAD(5),
*NRADF(20), NRADFI(100), NRAI(5), NRBA, NRFBOT(5), NRFBIT(5),
*NRFRD(100), NRS, NSPL, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGTPY(100), NTLCIF(10), NTLST, NTTCS(5), NTTMD(5), NTTTRP(5), NVAP,
*NVFAEA(10), NVLIDF(10), NWEPV(10), OOX, OBY, OBYPR, PTRAD(5), PDLCF
COMMON PHD(5,5), PPEAF(7), PRBITY, PROTNO(5,7), PRKLO(5,7),
*PROBIL(840), PRSHDD(5,7), RADFAE, RANGPR(840), RELRND(5), RELSUB(5),
*REP(10), RN, ROTDMPX(10,10), ROTDMPY(10,10), RSPI(5), S(100), SECON(7),
*SIGAD(50), SIGAR(50), SIGBD(50), SIOBR(50), STATAL(5,52), SWPDEL(100),
*SYMDDF(5,7), SYMDIF(5,7), SYMDS(7,7), SYMMAX(3), TARHT(5), TARL(5),
*TARRAD(5), TARNI(5), TDBIFV(10), TGYDEL(100), TGTQVL, TGTSPD,
*TGTVEL(100), TGTVL2, TGTYSV, TGTXL(100), TGTYNH(100), TGTYOL(100),
*TGTYSV, THETA, THRSIG(5,5), TIMEHV(100), TMBRD(5), TMBRI(5), THOLCF,
*THSWP(7), TMTOPR(130), TOTSIM, TRAVTM, XCYCLE(7), XDRP(10,10), XFIX,
*XMAX, XMIN, XODEF(20), XOVPI(10),
*XRDF(30), XROAD(11), XSWATH(50), XWIDTH, YDIV(100), YDHP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRDF(30), YROAD(11), YSWATH(50)
C....
DIMENSION XSYM(2,100), YSYM(2,100), NOBT(100)
KNOB=NOB-2
I1=1
I2=2
IF (NOBTP2-15) 300, 310, 320
300 KTS=KTSAV(IISAVE)
MINNUM=1
ISUB=1
KNOBT=NOBT(I1)-NOBTP2
SOBX=XSYM(I1,I1)-OBY
SOBY=YSYM(I1,I1)-OBYPR
GO TO 10
310 KTS=KTSAV(IISAVE)
ISUB=3
J=NWEPV(IMINE)
DO 315 K=1,J
XSYM(I1,K)=ROTDMPX(IMINE,K)
YSYM(I1,K)=ROTDMPY(IMINE,K)
315 NOBT(K)=I5
MINNUM=J
GO TO 10
320 KTS=KTSAV(IIS)
ISUB=2
MINNUM=1
NOBT(I1)=I7
XSYM(I1,I1)=TGTXL(IIS)
YSYM(I1,I1)=TGTYNH(IIS)
10 NLST=MINNUM
MINNUM=0
YMAX=0
YMIN=99999
DO 20 I=1,NLST
YMAX=AMAXI(YMAX,YSYM(I1,I1))
YMIN=AMINI(YMIN,YSYM(I1,I1))
20 KTHN=KTHMAX-KTSI-KTS
30 KTS=KTSI-1
IF (KTSI.LT.2) GO TO 40
IF (IOB(KTSI).EQ.-1) GO TO 30
IWORD=IOB(KTSI)
CALL UNPACK

```

Figure 30. (Continued)

```

      IF(NOBT2 GT.0) GO TO 30
      IF(YMIN-SYMMAX(I SUB) GT.OBY) GO TO 40
      KTMN=KTSI
      GO TO 30
40  KTSI=KTS
50  KTSI=KTSI-1
      IF(KTSI.GT.KNOB) GO TO 60
      IF(10B(KTSI).EQ.-1) GO TO 50
      IWORD=10B(KTSI)
      CALL UNPACK
      IF(NOBT2 GT.0) GO TO 50
      IF(YMAX+SYMMAX(I SUB).LT.OBY) GO TO 60
      KTRX=KTSI
      GO TO 50
60  DO 120 I=KTMN,KTRX
      IF(10B(I).EQ.-1) GO TO 120
      IWORD=10B(I)
      CALL UNPACK
      N=NOBT2
      IF(NOBT2 GT.0) GO TO 120
      DO 110 J=1,NLST
      DIS=(OBX-XSYM(1,J))**2+(OBY-YSYM(1,J))**2
      IF(NOBT(J)-16)390,400,410
390  K=MOD(NOBT(J),0)
      IF(DIS.GT.SYNDIS(K,N)) GO TO 110
      GO TO 65
400  K=IMTVAP(IMINE)
      IF(DIS.GT.SYHDIF(K,N)) GO TO 110
      GO TO 65
410  K=IMTDEF(INDF)
      IF(DIS.GT.SYMODF(K,N)) GO TO 110
65  MINNUM=MINNUM+1
      NMDET(N)=NMDET(N)+1
      IF(MINNUM LE.100) GO TO 70
      WRITE(6,1000)
1000 FORMAT(1H0,"MORE THAN 100 SYMPATHETIC DETONATIONS")
      CALL EXIT
70  XSYM(12,MINNUM)=OBX
      YSYM(12,MINNUM)=OBY
      NOBT(MINNUM)=NOBT2
      CALL BOOM(1DET,1)
      10B(1)=-1
      IF(1PRINT NE.1RUMS) GO TO 120
      IF(MINNUM.EQ.1) WRITE(6,1010)
1010 FORMAT(* SYMPATHETIC DETONATIONS      OB-NUM  OBS-TYPE  OB X    OB Y*)
      WRITE(6,1020) 1,NOBT2,OBX,OBY
1020 FORMAT(26X,15,110,F8.1,F7.1)
      GO TO 120
110  CONTINUE
120  CONTINUE
      IF(MINNUM.EQ.0) GO TO 200
      I3=11
      I1=12
      I2=13
      I SUB=1
      GO TO 10
200  NOBT2=KNOBT
      OBX=SOBX
      OBY=SOBY
      RETURN
      END

```

Figure 30. (Continued)

```

SUBROUTINE PRINTO
COMMON ACEP(5,2),ADFP(20),ACI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDL,AYVYP(10),DBFAE,
*DEFT(5),DEFL(5),DEFRA(5),DEFW(5),DEFX(20),DEFY(20),DELATH(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMPPIX(10,10),DMPPIY(10,10),
*DUOPRB(7),D3DEL,ET(5,5,2),ID,DISOP,IDP(20,15,2),IEND,
*IS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IVORD,IVTDEF(20),IVTVAP(10)
COMMON JMARKI,JSELDS(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KISAV(100),KILCFOPT,LCFTT,LDFDPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDEFEA(15,20),NDF,NDFA,
*NOFAA,NOFOAM(5),NOFHD,NDFMT,NOYT,NGTAH(100),NGTAWT,NGTAMI(100),
*NOBT(5),NITBT(5),NKILL(5),NKILLO(5),NKILLI(5),NKILLT,NNDET(7),
*NMINT(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF1(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYPI(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVEFAA(10),NVLIDF(10),NWEPV(10),OBX,OBJ,OBYPRT,PATRAD(5),PDLCF,
COMMON PHD(5,5),PEAF(7),PRBITY,PRDNO(5,7),PRKLO(5,7),
*PROBIL(100),PRSHDD(5,7),RADFAE,RANGPR(1040),RELNRD(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTXOL(100),TGTYNW(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEHV(100),THBR(5),THBR1(5),THDLCF,
*THSWP(7),TMTOPR(130),TOTSIM,TRAVTM,XCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XDEF(20),XOVP(10),
*XRDF(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YOMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(11),YSWATH(50)
C****
TOTSIM=TIMEHV(ISAVE)+TGTDEL(ISAVE)+SWPDEL(ISAVE)
IF(NOBTP2 EQ.17) GO TO 1520
IF(NOBTP2 EQ.18) GO TO 1530
IF(NOBTP2 EQ.16) RETURN
IF(NOBTP2=8)10,1500
10 DO 280 K1=1,11
K6=K1*10
K5=K6/9
IF(K6.GT.NTGO) K6=NTGO
PRINT 1001,(1,1=K5,K6)
PRINT 1002,(KISAV(1),1=K5,K6)
PRINT 1003,(DIRDIS(1),1=K5,K6)
IF(K6 EQ.NTGO) GO TO 40
280 CONTINUE
1001 FORMAT(*TARGET*,2X,1019)
1002 FORMAT(* OBSTACLE*,1019)
1003 FORMAT(* DISTANCE*,10F9.2)
40 TYPRIT = TGTYNW(ISAVE)
TVLPR=TGTVEL(ISAVE)
IF(TGTYNW(ISAVE).LT.99970.) GO TO 1004
TYPRI=TGTYSV
TVLPR=TGTYSV
1004 WRITE(6,1400) ISAVE,NTGTYPI(ISAVE),TYPRIT,TGTXOL(ISAVE),
*TVLPR,TIMEHV(ISAVE),TOTSIM
WRITE(6,1410)KISAV(ISAVE),NOBTP2,OBX,OBYPRT,PRBITY,RN,IPO
WRITE(6,1420)
NOBEVT=NOBEVT+1
IF(NOBEVT.GT.200) IPRINT=0
N2=NTGO/2
DO 1019 K=1,N2
M=K*N2
TOT2=DELATH(M)
TOTDEL=DELATH(M)
1019 WRITE(6,1120) K,NTGTYPI(K),TGTYNW(K),TGTXOL(K),TGTVEL(K),TIMEHV(K),
*TOTDEL,M,NTGTYPI(M),TGTYNW(M),TGTXOL(M),TGTVEL(M),TIMEHV(M),
*TOT2
IF(M=NTGO)1440,1012
1440 M=NTGO
TOTDEL=DELATH(M)
WRITE(6,1120)M,NTGTYPI(M),TGTYNW(M),TGTXOL(M),TGTVEL(M),TIMEHV(M),

```

Figure 30. (Continued)

```

*TOTDEL
1012 RETURN
1500 WRITE(6,1510)ISAVE,OBYPRT,TOTSIM
1510 FORMAT(1X,9(1H*)) * TRAVEL PATH BOUNDARY EVENT TOT = *
      @14 * OB Y=*,F8.1 * BREACH TIME=*,F8.3,9(1H*)
      RETURN
1520 IPO=9HENTRANCE
      GO TO 1540
1530 IPO=9HEXIT
1540 WRITE(6,1545)IPO,IT,OBYPRT,ISAVE
1545 FORMAT(1X,9(1H*)) * DIRECT FIRE AREA * ,A10.
      @ * NUMBER * ,13 * OB Y=*,F8.1 * EVENT TOT * ,15,9(1H*)
      RETURN
1120 FORMAT(1X,2(13,14,1X,F10.2,F9.2,F9.2,F11.2,F10.2,6X))
1400 FORMAT(1X/1X,10EVENT TOT=,14,2X,9HTGT TYPE=,13,2X,9HTGT Y=,
      *F8.1,9H TGT X=,F7.1/9H TGT VEL=,F6.1,2X,16HTOTAL TRAV TIME=,
      *F8.3,2X,12HBREACH TIME=,F8.3)
1410 FORMAT(1X/1X,10EVENT OBS=,14,2X,9HOB5 TYPE=,13,2X,9HOB X=,F8.1,
      *2X,9HOB Y=,F8.1,2X,9HPROB=,F9.2,2X,9HRN=,F9.2,5X,A10)
1420 FORMAT(1M,2(9HTGT TYPE,9X,9HTGT Y,4X,9HTGT X,2X,7HTGT VEL,2X,
      *9HTRAV TIME,2X,9HDEL TIME,6X)/)
      END

```

Figure 30. (Continued)

```

SUBROUTINE PRINTR
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AMIDL,AYYVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFM(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUDPRB(7),D3DEL,EI(5,5,2),ID,IDI5OP,IDI(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IMORD,IMTDEF(20),IMTVAP(10)
COMMON JMARKI, JSELOS(7), KABOOM(547), KDFTTN(100), KNRS, KOUNT(7),
*KYSAV(100), KI, LCFOPT, LCFIT, LDFDPR(5,5), LEFTIN, MADIV, WIFBT(7),
*MODE, NTFA(50), MUSH, NAP, NCTAM(100), NO, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFHO, NDFMT, NDVT, NGTAM(100), NGTAMT, NGTAMI(100),
*NIDBT(5), NITBT(5), NKILL(5), NKILLD(5), NKILLI(5), NKILLT, NMDET(7),
*NMNI(7), NMSWPT(7), NMT, NOB, NOBEVT, NOBTIP2, NOIT, NOSTAT, NRAD(5),
*NRADFD(20), NRADFI(100), NRAT(5), NRBA, NREBTD(5), NREBIT(5),
*NRFIRD(100), NRS, NSPLFT, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGTPR(100), NTLCIF(10), NTLOST, NTTICS(5), NTTMD(5), NTTTRP(5), NVAP,
*NVFAEA(10), NVLIDF(10), NNEPV(10), OOX, OBY, OBYPR, PATRAD(5), POLCF
COMMON PHD(5,5), PPEAF(7), PRBITY, PRDINO(5,7), PRKLO(5,7),
*PROBIL(840), PRSWDD(5,7), RADFAE, RANGPRIB(40), RELRND(5), RELSUB(5),
*REP(10), RN, ROTDMPX(10,10), ROTDMPY(10,10), RSP(5), S(100), SECON(7),
*SIGAD(50), SIGAR(50), SIGBD(50), SIGBR(50), STATAL(5,52), SWPDEL(100),
*SYMDDF(5,7), SYMDIF(5,7), SYMDIS(7,7), SYMMAX(3), TARHT(5), TARL(5),
*TARRAD(5), TARM(5), TDBIFV(10), TGTDEL(100), TGTQVL, TGTSPD,
*TGTVL(100), TGTVL2, TGTVSV, TGTXOL(100), TGTYNM(100), TGTYOL(100),
*TGTVSV, THETA, THRSIG(5,5), TIMEHV(100), INBRD(5), INBRI(5), INDLCF,
*THSWP(7), TMTOFRI(30), TOTSIM, TRAVTM, XCYCLE(7), XDMF(10,10), XFIX
*XMAX, XMIN, XODEF(20), XOVPI(10),
*XRDF(10), XROAD(10), XSWATH(50), XWIDTH, YDIV(100), YDMP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRDF(10), YROAD(10), YSWATH(50)
C....
COMMON /PD/ XRG(52), IRN, DIST(52)
DIMENSION OUT(16)
DO 60 I=1,52
50 DIST(I)=0.00
IF(JMARKI.LT.1) GO TO 80
DO 70 J=1,52
70 XRG(I,J)=0.
REWIND 7
JMARKI=0
IF(ITEROP.LT.1) GO TO 100
NTMAX=MAXD(NMT,NTGTP,NDFMT)
IF(MOD(IRUNS,NOSTAT).NE.1) GO TO 100
WRITE(6,5000) IEND,MODE,(NITBT(I),I=1,NTGTP)
IF(NDFMT.GT.0) WRITE(6,5001)(NIDBT(I),I=1,NDFMT)
IF(NMT.GT.0) WRITE(6,5002)(NMNI(I),I=1,NMT)
IF(ITEROP.LT.1) GO TO 100
WRITE(6,5003)
WRITE(6,5004)
100 BIG=0.
DO 110 I=1,NTGO
SUM=TIMEHV(I)+TGTDEL(I)+SWPDEL(I)
IF(SUM.LT.BIG) GO TO 110
BIG=SUM
TGTD=TGTDEL(I)
SWPD=SWPDEL(I)
TMNV=TIMEHV(I)
110 CONTINUE
TOTSIM=BIG
IF(ITEROP.LT.1) GO TO 310
WRITE(6,2010)
120 DO 300 I=1,NTMAX
DO 150 J=1,16
150 OUT(I,J)=1W
IF(I.LT.2) ENCODE(5,1001,OUT(1)) IRUNS
ENCODE(5,1001,OUT(2))
IF(NTGTP.GE.1.AND.NMT.GT.0) ENCODE(6,1002,OUT(3)) NKILL(I)
IF(NTGTP.GE.1.AND.NDFMT.GT.0) ENCODE(7,1004,OUT(4)) NKILLD(I)
IF(NTGTP.GE.1.AND.NVAP.GT.0) ENCODE(7,1004,OUT(5)) NKILLI(I)
IF(NDFMT.GE.1.AND.IRTFIR.GT.0) ENCODE(9,1006,OUT(6)) NDFDAMI(I)
IF(NTGTP.GE.1.AND.IRTFIR.GT.0) ENCODE(10,1007,OUT(7)) NREBIT(I)

```

Figure 30. (Continued)

```

IF (NDFWT.GT.1) ENCODE(8,1008,OUT(8)) NRFBDT(1)
IF (1.GT.NMT) GO TO 180
ENCODE(10,1007,OUT(9)) NMDET(1)
ENCODE(10,1007,OUT(10)) MIFBT(1)
ENCODE(8,1008,OUT(11)) NMSWPT(1)
180 IF (1.GT.1) GO TO 200
IF (INGTAMT.GT.0) ENCODE(6,1002,OUT(12)) NTLOST
IF (MADIV.GT.0) ENCODE(8,1013,OUT(13)) TGTD
IF (MODE.GT.1) ENCODE(7,1014,OUT(14)) SWPD
ENCODE(8,1013,OUT(15)) THMV
ENCODE(8,1013,OUT(16)) TOTSIW
200 N=11
IF (1.LT.2) N=16
WRITE(6,2000) (OUT(K),K=1,N)
300 CONTINUE
310 DO 320 I=1,NTGTP
DIST(1)=NKILL(1)
DIST(1+5)=NKILLD(1)
DIST(1+10)=NKILLI(1)
320 DIST(1+20)=NRFBT(1)
DO 330 I=1,NDFWT
DIST(1+15)=NDFDAM(1)
330 DIST(1+25)=NRFBDT(1)
DO 340 I=1,NMT
DIST(1+30)=NMDET(1)
DIST(1+37)=MIFBT(1)
340 DIST(1+44)=NMSWPT(1)
DIST(1+52)=TOTSIW
IF (DISOP.LT.1) GO TO 360
DO 350 I=1,52
350 XRG(I)=AMAX(XRG(I),DIST(I))
WRITE(7) DIST
360 DO 370 I=1,52
STATAL(1,I)=STATAL(1,I)+DIST(I)
370 STATAL(2,I)=STATAL(2,I)+DIST(I)+2
IF (MOD(IRUNS,NOSTAT) NE.0) RETURN
RUNS=IRUNS
IRN=IRUNS
DO 380 I=1,52
STATAL(3,I)=STATAL(1,I)/RUNS
STATAL(4,I)=(STATAL(2,I)-(STATAL(1,I)**2)/RUNS)/(RUNS-1.0)
IF (STATAL(4,I).GT.0) STATAL(5,I)=SQRT(STATAL(4,I))
380 CONTINUE
WRITE(6,6000)
IF (NMT.GT.0) WRITE(6,6010) ((STATAL(1,J),I=3,5),J=1,NTGTP)
K=NTGTP+5
IF (NDFWT.GT.0) WRITE(6,6020) ((STATAL(1,J),I=3,5),J=6,K)
K=NTGTP+10
IF (NVAP.GT.0) WRITE(6,6030) ((STATAL(1,J),I=3,5),J=11,K)
K=NDFWT+15
IF (NDFWT.GT.0) WRITE(6,6040) ((STATAL(1,J),I=3,5),J=16,K)
K=NTGTP+20
IF (NDFWT.GT.0) WRITE(6,6050)
IF (IRTFIR.GT.0) WRITE(6,6060) ((STATAL(1,J),I=3,5),J=21,K)
K=NDFWT+25
IF (NDFWT.GT.0) WRITE(6,6070) ((STATAL(1,J),I=3,5),J=26,K)
K=NMT+30
IF (NMT.LT.1) GO TO 400
WRITE(6,6080) ((STATAL(1,J),I=3,5),J=31,K)
K=NMT+37
WRITE(6,6090) ((STATAL(1,J),I=3,5),J=38,K)
K=NMT+44
IF (MODE.EQ.3) WRITE(6,6100) ((STATAL(1,J),I=3,5),J=45,K)
400 WRITE(6,6110) (STATAL(1,52),I=3,5)
IF (DISOP.GT.0) CALL DISTR(NMT,NTGTP,NDFWT)
RETURN
5000 FORMAT(1H1,40X,'RUN NUMBER ',13,' TACTIC NUMBER ',13/
'35X,'INITIAL NUMBER OF INTRUDERS BY TYPE ',5I6)
5001 FORMAT(35X,'INITIAL NUMBER OF DEFENDERS BY TYPE ',5I6)
5002 FORMAT(35X,'NUMBER OF MINES DISPENSED BY TYPE ',7I6)
5003 FORMAT(1H0,12X,'INTRUDERS DAMAGED BY',2X,'ITER',16X,
' DIR IND. DEFENDERS ROUNDS FIRED BY MINES ')

```

Figure 30. (Continued)

```

**MINES MINES TGT5 RMVL SWEEP TRAVEL BREACH**
*2X.*NUM. TYPE MINES FIRE FIRE DAMAGED INT.*
** DETONATED IN FIELD SWEEP LOST TIME TIME**
** TIME TIME**
5004 FORMAT(2(2X.4H*****),3(2X.5H*****),2X.3(3H***),1X,
*2(2X.6H*****),2X.3(3H***),2X.4(2H**),2X.5H*****),2X.4H*****
*2(2X.5H*****),2(2X.6H*****))
1001 FORMAT(15)
1002 FORMAT(16)
1004 FORMAT(17)
1006 FORMAT(19)
1007 FORMAT(110)
1008 FORMAT(10)
1013 FORMAT(F0.1)
1014 FORMAT(F7.1)
2000 FORMAT(1X,2A5,A6,2A7,1X,A9,A10,AB,2A10,1X,AB,A6,AB,A7,2AB)
2010 FORMAT(1X)
6000 FORMAT(11H1,*,STATISTICAL SUMMARY*,21X,*,MEAN*,6X,*,VARIANCE*,
*5X,*,STD. DEV.*/** INTRUDERS DAMAGED*)
6010 FORMAT(1H0,14X,*,BY MINES*,0X,3F14.3/(31X,3F14.3))
6020 FORMAT(1H0,14X,*,BY DIRECT FIRE *,3F14.3/(31X,3F14.3))
6030 FORMAT(1H0,14X,*,BY INDIRECT FIRE*,3F14.3/(31X,3F14.3))
6040 FORMAT(1H0,*,DEFENDERS DAMAGED*,13X,3F14.3/(31X,3F14.3))
6050 FORMAT(1H0,*,ROUNDS FIRED*)
6060 FORMAT(1H0,9X,*,BY INTRUDERS*,9X,3F14.3/(31X,3F14.3))
6070 FORMAT(1H0,9X,*,BY DEFENDERS*,9X,3F14.3/(31X,3F14.3))
6080 FORMAT(1H0,*,MINES DETONATED*,15X,3F14.3/(31X,3F14.3))
6090 FORMAT(1H0,*,MINES IN FIELD*,16X,3F14.3/(31X,3F14.3))
6100 FORMAT(1H0,*,MINES SWEEP*,19X,3F14.3/(31X,3F14.3))
6110 FORMAT(1H0,*,BREACH TIME (MINUTES)*,9X,3F14.3)
END

```

Figure 30. (Continued)



```

150 CONTINUE
180 IF(NDFMT.LT.1) GO TO 200
DO 195 I=1,NTGTP
  I14=I*14
  IOUT(I+10)=INT(I+10)
  LFMT(I14)=3H17.
  MFMT(I14)=5HF7.3.
189 DO 190 J=1,21
  OUT(J,I+10)=DISTS(J,I+10)
190 CONTINUE
195 CONTINUE
200 WRITE(6,LFMT)IOUT
  WRITE(6,MFMT)(I,(OUT(I+1,J),J=1,15),I=KZ,20)
  IF(NDFMT.LT.1) GO TO 500

C
DO 210 I=1,15
  IOUT(I)=1H
DO 210 J=1,21
  OUT(J,I)=1H
210 CONTINUE
DO 215 I=1,27
  LFMT(I)=1FMT(I)
  MFMT(I)=1JFMT(I)
215 CONTINUE
WRITE(6,1002)
1002 FORMAT(////,9X,112(1H),/,
*9X,1H*.34X,4H* *.22X*.R O U N D S F I R E D*.
1* 8 Y*.23X,1H*.7.9X,39H* D E F E N D E R S D A M A G E D *.
28X,17H I N T R U D E R S.9X,4H* *.9X*.D E F E N D E R S*
3.8X,1H*)
WRITE(6,1007)
DO 250 I=1,NDFMT
  IOUT(I)=INT(I+15)
  LFMT(I+2)=3H17.
  MFMT(I+2)=5HF7.3.
DO 240 J=1,21
  OUT(J,I)=DISTS(J,I+15)
240 CONTINUE
250 CONTINUE
DO 285 I=1,NTGTP
  IOUT(I+5)=INT(I+20)
  LFMT(I+8)=3H17.
  MFMT(I+8)=5HF7.3.
DO 280 J=1,21
  OUT(J,I+5)=DISTS(J,I+20)
280 CONTINUE
285 CONTINUE
DO 300 I=1,NDFMT
  IOUT(I+10)=INT(I+25)
  LFMT(I+14)=3H17.
  MFMT(I+14)=5HF7.3.
DO 295 J=1,21
  OUT(J,I+10)=DISTS(J,I+25)
295 CONTINUE
300 CONTINUE
WRITE(6,LFMT)IOUT
WRITE(6,MFMT)(I,(OUT(I+1,J),J=1,15),I=KZ,20)
WRITE(6,1003)
1003 FORMAT(////)
500 WRITE(6,1000)
  IF(NMT.LT.1) GO TO 700
1005 FORMAT(9X,1H*.10X,28H I N E S D E T O N A T E D.9X
1.4H* *.10X,25H I N E S I N F I E L D.12X,1H*)
1006 FORMAT(9X,1H*.13X,20H I N E S S W E P T.14X,4H* *.
113X,20H R E A C H T I M E.14X,1H*)
DO 510 I=1,27
  LFMT(I)=MFMT(I)
  510 MFMT(I)=KFMT(I)
DO 520 I=1,15
  IOUT(I)=1H
DO 520 J=1,21

```

Figure 30. (Continued)

```

      OUT(J,1)=1M
520  CONTINUE
      WRITE(6,1000)
      WRITE(6,1005)
      WRITE(6,1008)
C
      DO 550 I=1,NMT
      IOUT(I)=INT(I*30)
      LFMT(I*2)=3M17.
      MFMT(I*2)=5MF7.3.
      DO 540 J=1,21
      OUT(J,1)=DISTS(J,I*30)
540  CONTINUE
550  CONTINUE
      DO 570 I=1,NMT
      IOUT(I*7)=INT(I*37)
      LFMT(I*10)=2M17
      MFMT(I*10)=4MF7.3
      DO 560 J=1,21
      OUT(J,I*7)=DISTS(J,I*37)
560  CONTINUE
570  CONTINUE
      WRITE(6,LFMT) (IOUT(I),I=1,14)
      WRITE(6,MFMT) (IOUT(I*1,J),J=1,14),I=KZ,20)
1008 FORMAT(9X,49(1M*),2X,49(1M*))
      WRITE(6,1003)
700  DO 710 I=1,27
      LFMT(I)=KFMT(I)
      MFMT(I)=KFMT(I)
710  CONTINUE
      LFMT(13)=3M17.
      MFMT(13)=5MF7.3.
      DO 720 I=1,15
      IOUT(I)=1M
      DO 720 J=1,21
      OUT(J,I)=1M
720  CONTINUE
      DO 760 I=1,NMT
      IOUT(I)=INT(I*44)
      LFMT(I*2)=3M17.
      MFMT(I*2)=5MF7.3.
      DO 770 J=1,21
      OUT(J,I)=DISTS(J,I*44)
770  CONTINUE
780  CONTINUE
      IOUT(10)=INT(52)
      DO 790 J=1,21
      OUT(J,10)=DISTS(J,52)
790  CONTINUE
      WRITE(6,1008)
      WRITE(6,1006)
      WRITE(6,1008)
      WRITE(6,LFMT) (IOUT(I),I=1,14)
      WRITE(6,MFMT) (IOUT(I*1,J),J=1,14),I=KZ,20)
      RETURN
      END

```

Figure 30. (Continued)

```

SUBROUTINE TABINT(DISTPD,NTABLE)
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AMIDL,AVYVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP(1,10,10),DMP(1,10,10),
*DUDPRB(7),D3DEL,EI(5,5,2),ID,IDISOP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IVORD,IVTDEF(20),IVTVAP(10)
COMMON JMARK,USELDS(7),KABOOM(5*7),KDFTTN(100),KNRS,KOUNT(7),
*KTSAY(100),KI,LCFOPT,LCFTT,LDFDPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,RTFA(50),MUSH,NAP,NCTAM(100),ND,NDEFEA(5,20),NOF,NOFA,
*NOFAA,NOFAM(5),NOFMD,NOFMT,NOVT,NGTAM(100),NGTANT,NGTAMI(100),
*NIOBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMOET(7),
*NMIN(7),NHSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF(100),NRAL(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NNEPV(10),OBX,OBJ,OBJPRT,PATRAD(5),PDLCF
COMMON PHD(5,5),PPEAF(7),PROBIL,PROTND(5,7),PRKLO(5,7),
*PROBIL(8*0),PRSWDB(5,7),RADFAE,RANGPR(8*0),RELRND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPOEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYHMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARN(5),TOBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTVSV,TGTXOL(100),TGTYNH(100),TGTYOL(100),
*TGTVSV,THETA,THRSIG(5,5),TIMEHV(100),THORD(5),THBRI(5),TMDLCF,
*THSWP(7),TMTOPR(10),TOTSIM,TRAVTM,XCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XDEFF(20),XOVP(10),
*XRDF(30),XROAD(11),XSWATH(50),XWIDTH,YDIVER(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(11),YSWATH(50)
.....
10  N0=NTABLE*8-7
    NE=N0*7
    DO 50 K5=N0,NE
      IF (DISTPD GE RANGPR(K5) AND DISTPD LE RANGPR(K5+1)) GO TO 55
50  CONTINUE
    GO TO 60
55  PROBIL=PROBIL(K5)+ (DISTPD-RANGPR(K5))/(RANGPR(K5+1) - RANGPR(K5)
2    )*(PROBIL(K5+1)-PROBIL(K5))
    GO TO 70
60  PROBIL=0
70  RETURN
    END

```

Figure 30. (Continued)

```

SUBROUTINE RNORM1START
RN19 = RANF(DUMMY)
Q = ABS(1.-2.0*RN19)
Q = .5*(1.0-Q)
Q = -2.*ALOG(Q)
V = SQRT(Q)
SIGN = 1.
IF(RN19-.5) 10,10,20
10 SIGN=-1.
20 Q=2.519517+.002093*V+.010320*V*V
QQ = 1.+1.432700*V+.100259*V*V+.0013200*V*V*V
RSTART=(V-Q/QQ)*SIGN
RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE IPACK
C
C THIS SUBROUTINE PACKS THE MINE INFORMATION INTO IWORD
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDLC,AYYVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMPFIX(10,10),DMPFIY(10,10),
*DUDPRR(7),D3DEL,EI(5,5,2),ID,IDI SOP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTEIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,IWORD,IMTDEF(20),IMTVAP(10)
COMMON JMARKI,JSELD(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KTSP(100),KIL,LCFOPT,LCFTT,LCFDR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MTEFA(50),MUSH,NAP,NCTAW(100),ND,NOEFAE(15,20),NDF,NOFA,
*NDFAA,NDFDAM(5),NDFND,NDFWT,NOVT,NGTAW(100),NGTAWT,NGTANI(100),
*NIDBT(5),NIBT(5),NKILL(5),NKILLD(5),NKILL(5),NKILLT,NHDET(7),
*NMIN(7),NMSMPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADFO(20),NRADFI(100),NRAT(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFRDI(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGP,
*NTGTYPI(100),NTLCIF(10),NTLOST,NTTICS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NHEPV(10),OBX,OBJ,OBJPRT,PATRAD(5),POLCF
COMMON PHD(5,5),PPEAF(7),PRBITY,PRDTNO(5,7),PRKLD(5,7),
*PRUBIL(840),PRSHD(5,7),RADFAE,RANGPR(840),RELNRD(5),RELSUB(5),
*REPI(10),RN,ROTHPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYNDDF(5,7),SYNDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVL(100),TGTVL2,TGTYSV,TGTXL(100),TGTYNW(100),TGTYOL(100),
*TGTVSV,THETA,THRSIG(5,5),TINEMV(100),TMBRD(5),TMBRI(5),TMDLCP,
*TMSWP(7),TMTOPR(130),TOTSIM,TRAYTH,XCYCLE(7),XDMPI(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOPP(10),
*XRDF(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVPI(10),YRDF(30),YROAD(11),YSWATH(50)
C...
IX = OBJ * 10 0
IY = OBJ * 10 0
IWORD = 0
IWORD=IWORD.OR.IMINE
IWORD=IWORD.OR.SHIFT(INOBT2,15)
IWORD=IWORD.OR.SHIFT(IT,20)
IWORD=IWORD.OR.SHIFT(LABS(IX),28)
IF(IX.LT.0) IWORD=IWORD.OR.1000000000000000
IWORD=IWORD.OR.SHIFT(IY,43)
RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE UNPACK
C
C THIS SUBROUTINE UNPACKS THE MINE INFORMATION FROM IWORD
C
COMMON ACEP(5,2), ADEP(20), AEI(5,5,4), AI(20), AIMPTX(50), AIMPTY(50),
*ALNGLC, ANGLMP(10), AREL(5,2), AREP(20), AMIDL, AYYVP(10), DBFAE,
*DEFHT(5), DEFL(5), DEFRAD(5), DEFW(5), DEFY(20), DELATN(100),
*DEP(10), DFTFAE, DIRATK(10), DIRDIS(100), DMPIIX(10,10), DMPIIY(10,10),
*DU DPRB(7), D3DEL, EI(5,5,2), ID, IDISOP, IDP(20,15,2), IEND,
*IIS, IMINE, INDFA, INTIME(7), IOB(5000), IPO, IPRINT, IRNK(7), IRTFIR,
*IRUNS, ISAVE, ISBL, ISVIFA(30), ISVLST, ISYMP(3), ISYP, IT, ITEROP,
*ITGTPR(5,5), IVAP, IVEL, IWORD, INTDEF(20), INTVAP(10)
COMMON JMARKI, JSELDS(7), KABOOM(547), KOFTTN(100), KNRS, KOUNT(7),
*KTSAV(100), KI, LCFOP, LCFIT, LDFDPR(5,5), LEFTIN, MADTV, MIFBT(7),
*MODE, MTEA(50), MUSH, NAP, NCTAW(100), ND, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFWD, NDFWT, NDTV, NGTAW(100), NGTAWT, NGTAWI(100),
*NIOBT(5), NITBT(5), NKILL(5), NKILLD(5), NKILLI(5), NKILLT, NMOET(7),
*NMIN(7), NMSWPT(7), NMT, NOB, NOBEVT, NOBTP2, NOIT, NOSTAT, NRAD(5),
*NRADFD(20), NRADFI(100), NRAI(5), NRBA, NRFBUT(5), NRFBIT(5),
*NRFIRD(100), NRS, NSPLFT, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGTYP(100), NTLCIF(10), NTLOST, NITTC(5), NITTHD(5), NITTRP(5), NVAP,
*NVFAEA(10), NVLIDF(10), NWEPV(10), OBX, OBY, OBYPR, PATRAD(5), PDLCF
COMMON PHD(5,5), PPEAF(7), PROBITY, PRDTNO(5,7), PRKLO(5,7),
*PROBIL(840), PRSWDO(5,7), RADFAE, RANGPR(840), RELRND(5), RELSUB(5),
*REP(10), RN, ROTDMPX(10,10), ROTOMPY(10,10), RSP(5), S(100), SECON(7),
*SIGAD(50), SIGAR(50), SIGBD(50), SIGBR(50), STAL(5,52), SWDEL(100),
*SYNDDF(5,7), SYMDIF(5,7), SYNOIS(7,7), SYMMAX(3), TARHT(5), TARL(5),
*TARRAD(5), TARM(5), TDBIFV(10), TGTDEL(100), TGTQVL, TGTSPD,
*TGTVEL(100), TGTVL2, TGTYSV, TGTXL(100), TGTYNW(100), TGTYOL(100),
*TGTYSV, THETA, THRSIO(5,5), TINEMV(100), THRD(5), THRI(5), THOLCF,
*THSWP(7), TMTOPR(130), TOTSIM, TRAVTH, XCYLE(7), XDMP(10,10), XFIX,
*XMAX, XMIN, XODEF(20), XOVP(10),
*XRDFQ(30), XROAD(11), XSWATH(50), XWIDTH, YDIV(100), YDMP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRDFQ(30), YROAD(11), YSWATH(50)
C....
OBY=FLOAT(SHIFT(IWORD,-43).AND.177777B)/10.
OBX=FLOAT(SHIFT(IWORD,-28).AND.37777B)/10.
ITEMP=IWORD.AND.1000000000000000B
IF(ITEMP.NE.0) OBX=-OBX
NOBTP2=SHIFT(IWORD,-15).AND.378
IMINE=IWORD.AND.77777B
10 IT=SHIFT(IWORD,-20).AND.3778
20 RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE NOFIRE
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AINPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDLC,AYYVP(10),DBFAL,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFF(20),DELATH(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP(10),DMP(10),DMP(10),
*DUUPRB(7),D3DEL,ET(5,5,2),ID,DISOP,DP(20),IS,2),TEND,
*IS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,INORD,INTDEF(20),INTVAP(10)
COMMON JMARKI,USELDS(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KTSAV(100),KILCFOPT,LCFTT,LDOPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDFEA(15,20),NDF,NDFA,
*NOFAA,NDFDAH(5),NDFWD,NDFMT,NDVT,NGTAW(100),NGTAW,NGTAW(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMINE(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADFO(20),NRADFI(100),NRAL(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLST,NTTCS(5),NTTMD(5),NTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NWPV(10),OBX,OBJ,OBYPRT,PATRAD(5),POLCF
COMMON PHD(5,5),PPEAF(7),RADFAE,RANGPR(840),RELNRD(5),RELSUB(5),
*PROBIL(840),PRSWD(5,7),PRBITY,PROTNO(5,7),PRKLD(5,7),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SICAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SHPDEL(100),
*SYHDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYHMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTXL(100),TGTYNW(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMEHV(100),TMBRD(5),TMBRI(5),TMDLCF,
*MSWP(7),TMTOFR(130),TOTSIM,TRAVTH,XCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRDF(30),XROAD(1),XSWATH(50),XWIDTH,YDIV(100),YDWP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(1),YSWATH(50)
C....
BIGG(X)= 5*SQR(1-EXP(-63*X**2))
PI=3.141592654
SRPI=SQR(PI)
IWT=INTVAP(IVAP)
ICMHE=1
IF(NSUB(IWT),LT,1) ICMHE=2
COSIMP=COS(ANGIMP(IVAP)*.01745329258)
SINIMP=SIN(ANGIMP(IVAP)*.01745329258)
ANG=DIRATK(IVAP)
COSANG=COS(ANG)
SINANG=SIN(ANG)
NW=NWEPV(IVAP)
IF(IRUNS.NE.IPRINT)GO TO 90
NVOL=NVFAEA(IVAP)-NVLIDF(IVAP)+1
WRITE(6,1000)IVAP,NVOL,IWT
1000 FORMAT(IX,' AIM POINT NO.',I4,' VOLLEY NO.',I4
*,' INDIRECT FIRE WEAPON TYPE ',I3)
90 CONTINUE
DO 500 I=1,NTGO
IF(TGTYNW(I).GT.99990) GO TO 500
XT=TGTKOL(I)
YT=TGTYNW(I)
ITT=NTGTYP(I)
SURVPR=1
DO 480 J=1,NW
EIVAL=ET(IWT,ITT,2)
KODEI=ET(IWT,ITT,1)
XW=ROTDMPX(IVAP,J)
YW=ROTDMPY(IVAP,J)
RO=ABS((XW-XT)*SINANG+(YW-YT)*COSANG)
DO=ABS(-(XW-XT)*COSANG+(YW-YT)*SINANG)
DSQ=RO**2+DO**2
IF(DSQ.GT.THRSIG(IWT,ITT)) GO TO 480
APHD=PHD(IWT,ITT)
IF(ICMHE.GT.1) GO TO 110
POWER=(NSUB(IWT)*RELSUB(IWT)*EIVAL)/PI*PATRAD(IWT)**2
IF(POWER.GT.-227) GO TO 100
APHD=1
GO TO 105
100 APHD=1-EXP(POWER)

```

Figure 30. (Continued)

```

105 ETL=ETH*SRPI*PATRAD(IWT)
110 GO TO (120,140,150,160),KODEI
120 IF(ICMHE.LT.2) GO TO 130
    ALWRAT=1.-.8*COSIMP
    ETL=2.*SQRT(EIVAL*ALWRAT/PI)
    ETH=ETL/ALWRAT
130 DENOM=17.6*REP(IVAP)**2*ETL**2
    RSSP=ETL/SQRT(DENOM)*EXP((-4.*RO**2)/DENOM)
    DENOM=17.6*DEP(IVAP)**2*ETH**2
    DSSP=ETH/SQRT(DENOM)*EXP((-4.*DO**2)/DENOM)
135 PKK=RSSP*DSSP*RELND(IWT)*APHD
    GO TO 450
140 ETH=SQRT(EIVAL)
    ETL=ETH/SINIMP
    GO TO 220
150 ETL=ETH*SQRT(EIVAL)
    GO TO 220
160 IF(TARRAD(ITT))170,180
170 ETL=ETH*SRPI*(TARRAD(ITT)*EIVAL)
    GO TO 190
180 ETL=TARL(ITT)*2.*EIVAL
    ETH=TARH(ITT)*2.*EIVAL
190 IF(TARH(ITT))200,220
200 SHADOL=TARHT(ITT)*SINIMP/COSIMP
    IF(SHADOL.LT.EIVAL) GO TO 220
    IF(TARRAD(ITT))205,210
205 ETL=(ETL**2*2.*TARRAD(ITT)*(SHADOL-EIVAL))/ETL
    GO TO 220
210 ETL=(ETL*ETH*TARH(ITT)*(SHADOL-EIVAL))/ETH
220 A1=(ETL*2.*RO)/(2.96*REP(IVAP))
    A2=ABS(ETL-2.*RO)/(2.96*REP(IVAP))
    B1=(ETH*2.*DO)/(2.96*DEP(IVAP))
    B2=ABS(ETH-2.*DO)/(2.96*DEP(IVAP))
    RSSP=BIGG(A1)*SIGN(1.,ETL-2.*RO)*BIGG(A2)
    DSSP=BIGG(B1)*SIGN(1.,ETH-2.*DO)*BIGG(B2)
    GO TO 135
450 SURVPR=SRVPR*(1-PKK)
480 CONTINUE
    IF(SURVPR.EQ.1.) GO TO 500
    POAM=1.-SURVPR
    RN=RN*(DUMMY)
    IF(RN.GT.POAM) GO TO 490
    IF(IRUNS.NE.IPRINT) GO TO 485
    WRITE(6,1001),XT,YT,POAM
1001 FORMAT(' INTRUDER NO. ',13,' TGT X ',F8.1,' TGT Y ',F8.1,
    * ' POAM ',F14.3,' HIT HIM ')
485 TGTYNW(1)=TGTYNW(1)+100000.
    TGTYSV=TGTVEL(1)
    TMTOPR(1)=9990.
    TGTVEL(1)=0.
    LEFTIN=LEFTIN-1
    IF(INTTCS(1).GT.0)NSPLFT=NSPLFT-1
    NKILLT=NKILLT+1
    NKILL(ITT)=NKILL(ITT)+1
    IF(NDFA.LT.1)GO TO 500
    DO 487 L=1,NDFA
487 CALL STINT(SVIFA,L,L,IBIT,KDFTIN,2)
    GO TO 500
490 IF(IRUNS.EQ.IPRINT)WRITE(6,1002),XT,YT,POAM
1002 FORMAT(' INTRUDER NO. ',13,' TGT X ',F8.1,' TGT Y ',F8.1,
    * ' POAM ',F14.3,' MISSED ')
500 CONTINUE
    RETURN
    END

```

Figure 30. (Continued)

```

SUBROUTINE DIRFIR(IDAM)
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50)
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDLC,AYYVP(10),DBFAL,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFM(5),DEFX(20),DEFY(20),DELATH(100),
*DEP(10),DETFAE,DIRATK(10),DIRDIS(100),DHPIIX(10,10),DHPIIY(10,10),
*DUDPRP(7),D3DEL,ET(5,5,2),ID,IDIOP,IPD(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*ITRNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYMP(3),ISYP,IT,ITEROP,
*ITGTP(5,5),IVAP,IVEL,IWORD,IWDEF(20),IMTVAP(10)
COMMON JMARK1,JSELD(7),KABOOM(547),KDFITN(100),KNRS,KOUNT(7),
*KTSAV(100),KILCOFT,LCFTT,LDOPPR(5,5),LEFTIN,RADIV,MIFBT(7),
*MODE,MIFA(50),MUSH,NAP,NCTAM(100),ND,NDFEAT(5,20),NDF,NDFA,
*NDFAA,NDFAM(5),NDFHD,NDFMT,NDVT,NGTAM(100),NGTAMT,NGTAMI(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDE(7),
*NM(17),NMSHPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADFD(20),NRADF(100),NRAL(5),NRBA,NRFBOT(5),NRFBIT(5),
*NRFRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTP(100),NTLCIF(10),NTLOST,NTTTC(5),NTTTHD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NNEPV(10),OBX,OBJ,OBYPRT,PATRAD(5),PDLCF,
COMMON PHD(5,5),PPEAF(7),PRBITY,PROTNO(5,7),PRKLO(5,7),
*PROBIL(840),PRSHDD(5,7),RADFAE,RANGPR(840),RELRND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGADI(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYHMAX(3),TARMT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIFV(10),TGTDEL(100),TGTVL, TGTSPD,
*TGTVL(100),TGTVL2,TGTYSV,TGTXL(100),TGTYNH(100),TGYOL(100),
*TGTVSV,THETA,THRSIG(5,5),TIMEHV(100),TMBRD(5),TMBRI(5),TMDLCF,
*TMWHP(7),TMTOFRI(30),TOTSIM,TRAVTN,XCYCLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRDF(10),XRADD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(10,30),YRADD(11),YSWATH(50)
C....
BIGG(X)=.5*SQRT(1-EXP(-.63*X**2))
ID=IDAM
IDAM=0
ITT=NTGTP(115)
IWT=IWTDEF(INDF)
L=IWT
IF(ID.GT.1) L=ITT
BREL=AREL(L,10)
BCEP=ACEP(L,10)
KODEI=AEI(IWT,ITT,2*ID-1)
EIVAL=AEI(IWT,ITT,2*ID)
IF(KODEI.GT.1) GO TO 50
IND=IDP(INDF,KI,10)
ALHRT=1-.8*AI(IND)
ETL=2.*SQRT(EIVAL*ALHRT/3.141592654)
ETW=ETL/ALHRT
DENOM=17.6*AREP(IND)**2*ETL**2
RSSP=ETL/SQRT(DENOM)
DENOM=17.6*ADEP(IND)**2*ETW**2
DSSP=ETW/SQRT(DENOM)
SSPD=RSSP*DSSP*BREL
GO TO 100
50 IF(BCEP.GT.0) GO TO 60
XD=(TGTXL(115)-DEFX(INDF))**2
YD=(TGTYNH(115)-DEFY(INDF))**2
RNG=SQRT(XD+YD)
BCEP=.001*RNG*ABS(BCEP)
DENOM=1.7*BCEP
IF(ID.GT.1) GO TO 70
VERT=IARMT(ITT)
HORIZ=2.*TARRAD(ITT)
IF(HORIZ.LT.0) HORIZ=(TARL(ITT)+TARW(ITT))/2.
GO TO 80
70 VERT=DEFHT(IWT)
HORIZ=2.*DEFRAD(IWT)
IF(HORIZ.LT.0) HORIZ=(DEFL(IWT)+DEFM(IWT))/2
80 AI=VERT/DENOM
BI=VERT/DENOM
RSSP=2.*BIGG(AI)
DSSP=2.*BIGG(BI)

```

Figure 30. (Continued)

```
100  SSPD=RSSP*DSSP*BREL*EIVAL  
      RN=RAN(DUMMY)  
      IF(RN.LT.SSPD) 10AM=1  
      PRBITY=SSPD  
      RETURN  
      END
```

Figure 30. (Continued)

```

SUBROUTINE CKEYTM
COMMON ACEP(5,2), ADEP(20), AEI(5,5,4), AI(20), AIMPTX(50), AIMPTY(50),
*ALNGLC, ANGIMP(10), AREL(5,2), AREP(20), AWIDLK, AYYVP(10), DBFAE,
*DEFHT(5), DEFL(5), DEFRAD(5), DEFW(5), DEFY(20), DELATM(100),
*DEP(10), DFTFAE, DIRATK(10), DIRDIS(100), DMP1IX(10,10), DMP1IY(10,10),
*DUDPRB(7), D3DEL, E1(5,5,2), ID, IDISOP, IDP(20,15,2), IEND,
*IIS, IMINE, INDA, INTIME(7), IOB(5000), IPO, IPRINT, IRNK(7), IRTEIR,
*IRUNS, ISAVE, ISBL, ISVIFA(30), ISVLSL, ISYMP(3), ISYP, IT, ITEROP,
*ITGTPR(5,5), IVAP, IVEL, IWORD, IWTDEF(20), IWTVAP(10)
COMMON JMARK1, JSELD5(7), KABOOM(547), KDFTTN(100), KNRS, KOUNT(7),
*KTSAV(100), KI, LCFOPT, LCFTT, LDFOPR(5,5), LEFTIN, MADIV, MIFBT(7),
*MODE, MTF(50), MUSH, NAP, NCTAW(100), ND, NDEFEA(15,20), NDF, NDFA,
*NDFAA, NDFDAM(5), NDFWD, NDFWT, NOV, NGTAW(100), NGTAWT, NGTAWI(100),
*NIDBT(5), NITBT(5), NKILL(5), NKILLD(5), NKILLI(5), NKILLT, NHDET(7),
*NMIN(7), NMSWPT(7), NMI, NOB, NOBEVT, NOBTP2, NOIT, NOSTAT, NRAD(5),
*NRADF(20), NRADF1(100), NRAI(5), NRBA, NREBTD(5), NREBIT(5),
*NRFIRD(100), NRS, NSPLFT, NSTICK(50), NSUB(5), NTCOL(100), NTGO, NTGTP,
*NTGYPI(100), NTLCIF(10), NTLOST, NTTICS(5), NTTTMD(5), NTTTRP(5), NVAP,
*NVFAEA(10), NVLIDF(10), NNEPV(10), OBX, OBY, OBYPRT, PATRAD(5), POLCF
COMMON PHD(5,5), PPEAF(7), PRBIT, PROTNO(5,7), PRKLD(5,7),
*PROBIL(840), PRSDD(5,7), RADFAE, RANOPR(840), RELRND(5), RELSUB(5),
*REPI(10), RN, ROTDMPX(10,10), ROTDMPY(10,10), RSP(5), S(100), SECON(7),
*SIGAD(50), SIGAR(50), SIGBD(50), SIGBR(50), STATAL(5,52), SWPDEL(100),
*SYMDDF(5,7), SYMDIF(5,7), SYMDIS(7,7), SYMHAX(3), TARHT(5), TARL(5),
*TARRAD(5), TARW(5), TDBIFV(10), TGTDEL(100), TGTQVL, TGTSPD,
*TGTVEL(100), TGTVL2, TGTYSV, TGTXL(100), TGTYNW(100), TGTYOL(100),
*TGYSV, THETA, THRSIG(5,5), TIMEMV(100), TMBRD(5), TMBRI(5), TMDLCF,
*THSWP(7), THTOFR(130), TOTSIM, TRAVTM, XCYCLE(7), XDMP(10,10), XFIX,
*XMAX, XMIN, XODEF(20), XQVP(10),
*XRDF(130), XROAD(11), XSWATH(50), XWIDTH, YDIV(100), YDMP(10,10),
*YLENGTH, YODEF(20), YOVP(10), YRDF(30), YROAD(11), YSWATH(50)
C****
DIMENSION TM(130), IPOS(130)
IF(INDFA.GT.0) GOTO 50
IF(INDFA.GT.0) GO TO 50
IF(NOBT2.LT.16.OR.NOBT2.GT.18) GO TO 20
IF(NOBT2.GT.16) GO TO 15
INDFA=1
GO TO 20
C
C SAVE THIS INTRUDER FOR THIS AREA BY TURNING ON A BIT IN ISVIFA
C
15 CALL STINT(ISVIFA, ISAVE, IT, IBIT, KDFTTN, I)
NDFA=MAX0(IT, NDFA)
CALL TGTMOV
16 DO 18 J=1, NDFWD
IF(NDEFEA(IT, J), LT, I) GO TO 18
NDF=NDEFEA(IT, J)+100
IF(THTOFR(NDF).GT.0.O.AND.THTOFR(NDF).LT.9000.) GO TO 18
THTOFR(NDF)=0.
18 CONTINUE
RETURN
20 CALL TGTMOV
RETURN
50 DO 60 I=1, 130
IPOS(I)=1
TM(I)=THTOFR(I)
60 CONTINUE
DO 90 J=1, 130
IF(TM(I)-TM(J))90,90,85
85 KEEP=IPOS(I)
IPOS(I)=IPOS(J)
IPOS(J)=KEEP
SAVE=TM(I)
TM(I)=TM(J)
TM(J)=SAVE
90 CONTINUE
IF(TM(I).GT. TRAVTM) GO TO 120
100 CONTINUE
120 NUMCNT=MIND(I, 130)
IF(NUMCNT.GT.1) GO TO 150

```

Figure 30. (Continued)

```

CALL TGTMOV
IF( NOBT2.NE.17) RETURN
CALL STINT(1SVIFA,1SAVE,1T,1BIT,KDFTN,1)
NDFA=MAX(11,NDFA1)
GO TO 16

```

C  
C

```

150 NUM=NUMCNT-1
ISHT=0
DO 1000 MI=1,NUM
M=IP0S(M1)
TTOI=TTOFR(M)
IF(TTOI.GT.9000.) GO TO 1000
IT=TRAVM
TRAVM=TTOI
CALL TGTMOV
TRAVM=IT-TTOI
IF(M.GT.120) GO TO 750
IF(M.LT.101) GO TO 500

```

C  
C  
C

THIS IS A DEFENDER FIRING AT AN INTRUDER

```

NDF=M-100
IF(NRADFD(NDF).LT.1) GO TO 900
NDTYP=1WTOEF(NDF)
DO 300 I=1,5
IPRR=1TGTPR(NDTYP,1)
IF(IPRR.LT.1) GO TO 1000
DO 280 IIS=1,NTGO
IF(NTGTYP(IIS).NE.IPRR) GO TO 280
DO 270 K=1,NDFA
CALL STINT(1SVIFA,IIS,K,1BIT,KDFTN,3)
IF(1BIT.LT.1) GOTO 270
IF(INDEFEATK(NDF).LT.1) GO TO 270
GO TO 320

```

270 CONTINUE  
280 CONTINUE  
300 CONTINUE  
GO TO 1000

320 TMTOFR(M)=TMBRD(NDTYP)

```

K1=K
IDAM=1
ISHT=1
CALL DIRF(R(IDAM)
IF(1SYMP(2).GT.0) CALL SYMDET
NRADFD(NDF)=NRADFD(NDF)-1
IF(NRADFD(NDF).LT.1) TMTOFR(M)=9991
NRFBDT(NDTYP)=NRFBDT(NDTYP)+1
IF(IRUNS.NE.1) PRINT) GOTO 340
AKIL=0H MISSED
IF(IDAM.GT.0) AKIL=0H HIT HIM
PRINT 9003,NDF,NDTYP,IIS,NTGTYP(IIS),AKIL,PRBITY

```

9003 FORMAT(' DEF NO. ',15,' TYPE ',12,' FIRED AT INT NO. ',  
113,' TYPE ',11,' AND ',AB,' PROB= ',F7.3)

340 IF(IDAM.LT.1) GO TO 475  
TGTYNH(IIS)=TGTYNH(IIS)+100000.

```

IF(INGTAWT.GT.0) CALL UNIT
LEFTIN=LEFTIN-1
TGTVEL(IIS)=0.
TMTOFR(IIS)=9999.
CALL STINT(1SVIFA,IIS,K,1BIT,KDFTN,2)
INTYP=NTGTYP(IIS)
NKILLD(INTYP)=NKILLD(INTYP)+1
NKILLT=NKILLT+1
IF(NTTICS(INTYP).GT.0) NSPLFT=NSPLFT-1
K1=IIS
XFIX=0.0
IF(NTTMD(INTYP).GT.0) CALL DIVSET

```

475 J=0  
DO 485 I=1,NTGO  
CALL STINT(1SVIFA,I,K,1BIT,KDFTN,3)  
IF(1BIT.LT.1) GO TO 485

Figure 30. (Continued)

```

      NRFIRD(1)=NRFIRD(1)+1
      IF(NRFIRD(1)-NRBA)485,480,485
480  TMTOFR(1)=0.
      J=1
485  CONTINUE
      IF(J)50,990
C
C   THIS IS AN INTRUDER FIRING AT A DEFENDER
C
500  NINTT=NTGTYP(M)
      IF(NRADF(M).LT.1)GO TO 900
      DO 500 I=1,5
      IPRR=LDFDPR(NINTT,I)
      IF(IPRR.LT.1) GO TO 1000
      DO 590 J=1,NDFWD
      IF(IHTDEF(J).NE.IPRR) GO TO 590
      DO 580 K=1,NDFA
      IF(NDEFEA(K,J).LT.1) GO TO 580
      CALL STINT(SVIFA,M,K,IBIT,KDFTN,3)
      IF(1BIT.LT.1) GO TO 580
      GO TO 620
580  CONTINUE
590  CONTINUE
600  CONTINUE
      GO TO 1000
620  TMTOFR(M)=TMBRI(NINTT)
      KI=K
      IDAM=2
      NDF=NDEFEA(K,J)
      ISHOT=1
      IIS=M
      CALL DIRFIR(IDAM)
      NRADF(IIS)=NRADF(IIS)-1
      IF(NRADF(IIS).LT.1)TMTOFR(M)=9990.
      NRFBIT(NINTT)=NRFBIT(NINTT)+1
      IF(IRUNS.NE.IPRINT) GO TO 640
      AKIL=8H MISSED
      IF(IDAM.GT.0) AKIL=8H HIT HIM
      PRINT 9005,M,NINTT,NDF,IHTDEF(INDF),AKIL,PRBLTY
9005  FORMAT(' INT NO. ',14,' TYPE',12,' FIRED AT DEF NO. ',
13,' TYPE ',11,' AND',A8,' PROB' ,F7.3)
640  IF(IDAM.LT.1) GO TO 990
      NDEFEA(K,J)=-NDF
      NDTP=IHTDEF(INDF)
      NDFDAM(NDTP)=NDFDAM(NDTP)+1
      TMTOFR(INDF+100)=9999.
      GO TO 990
750  IVAP=M-120
      IF(NVLIDF(IVAP).LT.1)GO TO 900
      CALL NDFIRE
      TMTOFR(M)=TDBIFV(IVAP)
      NVLIDF(IVAP)=NVLIDF(IVAP)-1
      ISHOT=1
      GO TO 990
900  TMTOFR(M)=9990.
      GO TO 1000
990  IF(TMTOFR(M).GT.TRAVTM)GO TO 1000
      GO TO 50
1000 CONTINUE
C
C
      IF(ISHOT)50,1050
1050  IF(NOBTPE-17)20,15,20
      END

```

Figure 30. (Continued)

```

SUBROUTINE STINT(I,SV,INT,IT,IBIT,KDFTTN,N)
DIMENSION SV(30),KDFTTN(100)
K=INT
KT=IT
I=1
IF(K.LT.59) GO TO 10
KT=IT+15
K=K-59
10 GO TO (20,50,100),N
20 ISV(KT)=ISV(KT).OR.SHIFT(I,K)
KDFTTN(INT)=KDFTTN(INT)+1
RETURN
50 ISW=0
J=15W
KK=J.OR.SHIFT(I,K)
KK=.NOT.KK
ISV(KT)=ISV(KT).AND.KK
KDFTTN(INT)=KDFTTN(INT)-1
RETURN
100 IBIT=SHIFT(ISV(KT),-K).AND.16
RETURN
END

```

Figure 30. (Continued)

```

SUBROUTINE EXPSWP
COMMON ACEP(5,2),ADEP(20),AET(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDLC,AYYP(10),OBFAE,
*DEFNT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DOOPR(7),D3DEL,FI(5,5,2),ID,IOISOP,IDP(20,15,2),IEND,
*IS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLST,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,LEVEL,IWORD,IMTDEF(20),IMTVAP(10)
COMMON JMARK1,USELDS(7),KABOOM(547),KDFTTN(100),KNRS,KOUNT(7),
*KISAV(100),K1,LCFOPT,LCFTT,LDFOPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MTFA(50),MUSH,NAP,NCTAW(100),ND,NDFEFA(15,20),NDF,NDFA,
*NDFAA,NDFDM(5),NDFWD,NDFWT,NDVT,NGTAW(100),NGTAMT,NGTAM(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMIN(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRADF(20),NRADF(100),NRAI(5),NRBA,NRFBDT(5),NRFBIT(5),
*NRFIRD(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGC,NTGTP,
*NTOTYP(100),NTLCTF(10),NTLST,NTTTC(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAE(10),NVLDF(10),NWEPY(10),OBX,OBJ,OBJPRT,PATRAD(5),PDLCF
COMMON PHD(5,5),PPEAF(7),PRBIT,PRDIND(5,7),PRKLQ(5,7),
*PROBIL(840),PRSHDD(5,7),RADFAE,RANGPR(840),RELRND(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGOD(50),SIGOR(50),STAL(5,52),SWDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYHMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARH(5),TOBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVL(100),TGTVL2,TGTYSV,TGTXOL(100),TGTYNH(100),TGTYGL(100),
*TGTVSV,THETA,THRSIG(5,5),TIMEHV(100),THORD(5),THORI(5),TMDLFC,
*THSWP(7),TMTOR(10),TOTSIM,TRAVIM,XCYCLE(7),XDMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XRDF(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(11),YSWATH(50)
C....
XLOC=OBX
KT=KISAV(ISAVE)
IOB(KT)=1
NH=NOBTP2
IF(IRUNS.EQ.IPRINT) WRITE(6,1000)
1000 FORMAT(' THE EXPLOSIVE SWEEP ROUTINE HAS REMOVED THE *
* FOLLOWING MINES*')
IF(LCFOPT.GT.1)GO TO 20
YDIST=OBJ*ALNGLC
RANGE=AWIDLC
GO TO 50
20 YLOC=OBJ*DFTFAE
YDIST=YLOC+RADFAE
YST=YLOC-RADFAE
RANGE=RADFAE**2
30 KT=KT+1
IF(IOB(KT).LT.0) GO TO 30
IWORD=IOB(KT)
CALL UNPACK
IF(NOBTP2-8)40,200,30
40 IF(OBY.LT.YST) GO TO 30
GO TO 60
50 KT=KT+1
IF(IOB(KT).LT.0) GO TO 50
IWORD=IOB(KT)
CALL UNPACK
IF(NOBTP2-8)60,200,50
60 IF(OBY.GT.YDIST) GO TO 200
DIST=ABS(XLOC-OBX)
IF(LCFOPT.GT.1)DIST=DIST**2+(YLOC-OBY)**2
IF(DIST.GT.RANGE) GO TO 50
RN=RANF(DUMMY)
IF(RN.GT.PDLCF)GO TO 50
IF(IRUNS.EQ.IPRINT) WRITE(6,1010) NOBTP2,OBX,OBJ
1010 FORMAT(' OBS TYPE= *.15.* OB X= *.F8.2.* OB Y= *.F8.2)
NMSWPT(NOBTP2)=NMSWPT(NOBTP2)+1
NMDET(NOBTP2)=NMDET(NOBTP2)+1
CALL BOOM(1DET,1)
IOB(KT)=-1
GO TO 50

```

Figure 30. (Continued)

```
240 DO 250 I=1,NTGO
      IF (TGTYNM(I) .GT. 99969) GO TO 250
      IF (NCTAM(I).NE.NCTAM(I)SAVE) GO TO 250
      SMPDEL(I)=SMPDEL(I)+TMDLCF
      DELATM(I)=TMDLCF
      TGTVEL(I)=0.
250 CONTINUE
      OBN=XLOC
      NOSTP2=NB
      RETURN
      END
```

Figure 30. (Continued)

```

SUBROUTINE TGTMOV
COMMON ACEP(5,2),ADEP(20),AEI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGTMP(10),AREL(5,2),AREP(20),AWIDLC,AYYVP(10),DBFAE,
*DEFHT(5),DEFL(5),DEFRAD(5),DEFW(5),DEFX(20),DEFF(20),DELATH(100),
*DFP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP(10,10),DNPI(10,10),
*DUOPRB(7),D3DEL,EI(5,5,2),ID,DISOP,IDP(20,15,2),IEND,
*IIS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSL,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,INORD,INTDEF(20),INTVAP(10)
COMMON JMARKI,JSELOS(7),KABOOM(5,7),KOFFTN(100),KNRS,KOUNT(7),
*KSAVI(100),K1,LCFOPT,LCFIT,LDFDPR(5,5),LEFTIN,MADIV,MIFBT(7),
*MODE,MFA(50),MUSH,NAP,NCTAW(100),ND,NDEFEA(15,20),NDF,NDFA,
*NDFAA,NDFDAM(5),NDFWD,NDFWT,NDVT,NGTAW(100),NGTAMT,NGTAWI(100),
*NIDRT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NMDET(7),
*NMNI(7),NMSWPT(7),NMT,NOB,NOBEVT,NOBTP2,NOIT,NOSTAT,NRAD(5),
*NRAOFD(20),NRAOFI(100),NRAI(5),NRBA,NRFBOT(5),NRFBIT(5),
*NRI(10,100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGO,NTGTP,
*NTGTP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTMD(5),NTTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NWEPV(10),OBX,OBJ,OBYPRT,PATRAD(5),POLCF,
COMMON PHD(5,5),PPEAF(7),PRBITY,PRDINO(5,7),PRKLO(5,7),
*PROBIL(8,8),PRSHDD(5,7),RADFAE,RANGPRI(8,8),RELNRD(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGBR(50),STATAL(5,52),SWPDEL(100),
*SYMDDF(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TOBIFV(10),TGTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTVSV,TGTXL(100),TGTYNW(100),TGTYOL(100),
*TGTVSV,THETA,THRSIG(5,5),TIMENV(100),TMBRD(5),TMBRI(5),TMDLCF,
*TMSWP(7),TMTOFR(130),TOTSIM,TRAVTM,XCYLE(7),XOMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XROFO(30),XROAD(11),XSWATH(50),XWIDTH,YDIV(100),YDNPI(10,10),
*YLENGTH,YODEF(20),YOVP(10),YROFO(30),YROAD(11),YSWATH(50)
.....
DO 600 I=1,NTGO
IF(TGTYNW(I).GT.99990.)GO TO 600
IF(TGTVEL(I).GT.0.)GO TO 500
DELATH(I)=DELATH(I)-TRAVTM
IF(DELATH(I).GT.0.00001)GO TO 550
TGTVEL(I)=TGTSPD
DTM=ABS(DELATH(I))
DELATH(I)=0.
TGTYNW(I)=TGTYNW(I)+TGTSPD*DTM
TIMENV(I)=TIMENV(I)+DTM
IF(TMTOFR(I).GT.9000.)GO TO 600
TMTOFR(I)=TMTOFR(I)-DTM
IF(TMTOFR(I).LT.0.)TMTOFR(I)=0.
GO TO 600
500 TGTYNW(I)=TGTYNW(I)+TGTSPD*TRAVTM
TIMENV(I)=TIMENV(I)+TRAVTM
950 IF(TMTOFR(I).GT.9000.)GO TO 600
TMTOFR(I)=TMTOFR(I)-TRAVTM
IF(TMTOFR(I).LT.0.)TMTOFR(I)=0.
600 CONTINUE
IF(NDFWD.LT.1)GO TO 750
NEND=NDFWD+100
DO 700 I=101,NEND
IF(TMTOFR(I).GT.9000.)GO TO 700
TMTOFR(I)=TMTOFR(I)-TRAVTM
IF(TMTOFR(I).LT.0.)TMTOFR(I)=0.
700 CONTINUE
750 IF(NVAP.LT.1)RETURN
NEND=120+NVAP
DO 900 I=121,NEND
IF(TMTOFR(I).GT.9000.)GO TO 900
TMTOFR(I)=TMTOFR(I)-TRAVTM
IF(TMTOFR(I).LT.0.)TMTOFR(I)=0.
900 CONTINUE
RETURN
END

```

Figure 30. (Concluded)

## SECTION V

### SIMULATION MODEL

SEMAC is capable of simulating the passage of up to 100 intruder targets of five different types through an engagement area defended by direct and indirect fire. The program can consider up to 50 sortie aimpoints, 32,767 mines of seven different types, ten indirect fire volley aimpoints with five different types of indirect fire munitions, and 20 defenders with five different direct fire weapon types. The intruders can be in any formation desired, and are masked from the defenders in selected portions of the engagement area by terrain shielding. The intruders are capable of return fire and may employ a variety of sweeping techniques.

#### TABLES AND ARRAYS

The three probability functions employed in SEMAC are entered as tables which have been set at a maximum of three sets of eight pairs of values for each target/mine combination being evaluated. All probability inputs are stored in PROBIL(840), and all corresponding range values are stored in RANGPR(840). The first eight entries in each table contain the probabilities of mine detection and corresponding ranges for the first target type and the first mine type. The second eight locations contain the probabilities that a detonated mine of the first type will damage a target of the first type and corresponding ranges. The third eight entries contain the probabilities that a mine of the first type will detonate when encountered by a target of the first type. The next three sets of eight entries contain similar data for the first target type and the second mine type, etc.

In addition, allocation is made in COMMON locations for a number of other arrays. The COMMON statements which are used in SEMAC are shown below.

```
COMMON ACEP(5,2),ADEP(20),ACI(5,5,4),AI(20),AIMPTX(50),AIMPTY(50),
*ALNGLC,ANGIMP(10),AREL(5,2),AREP(20),AWIDLC,AYVVP(10),DQFAE,
*DEPHT(5),DEFL(5),DEFRAD(5),DEFM(5),DEFX(20),DEFY(20),DELATM(100),
*DEP(10),DFTFAE,DIRATK(10),DIRDIS(100),DMP1IX(10,10),DMP1IY(10,10),
*DUDPR(7),D3DEL,EI(5,5,2),ID,IOISOP,IPD(20,15,2),IEND,
*IS,IMINE,INDFA,INTIME(7),IOB(5000),IPO,IPRINT,IRNK(7),IRTFIR,
*IRUNS,ISAVE,ISBL,ISVIFA(30),ISVLSI,ISYMP(3),ISYP,IT,ITEROP,
*ITGTPR(5,5),IVAP,IVEL,INORD,IMTDEF(20),IMTVAP(10)
COMMON JHARKI,USELDS(7),KABOOM(547),KDFTTM(100),KNRS,KOUNT(7),
*KTSAV(100),KI,LCFOPT,LCFTT,LOFOPR(5,5),LEFTIN,MADIV,WIFBT(7),
*MODE,MTEFA(50),MUSH,NAP,NCTAM(100),ND,NOFEA(15,20),NDF,NDFA,
*NDFAA,NDFDAM(5),NDFMD,NDFMT,NDVT,NGTAM(100),NGTAMT,NGTAMT(100),
*NIDBT(5),NITBT(5),NKILL(5),NKILLD(5),NKILLI(5),NKILLT,NHDEF(7),
*NM(7),NMSWPT(7),NMT,NOB,NOBEVT,NORTP2,NOIT,NOSTAT,NRAD(5),
*NRADFD(20),NRADFI(100),NRAI(5),NRRA,NRFBDT(5),NRFBIT(5),
```

```

*NRH(100),NRS,NSPLFT,NSTICK(50),NSUB(5),NTCOL(100),NTGC,NTGTP,
*NTGTYP(100),NTLCIF(10),NTLOST,NTTCS(5),NTTND(5),NTTRP(5),NVAP,
*NVFAEA(10),NVLIDF(10),NNEPV(10),OBY,OBYPRT,PATRAD(5),POLCF,
*COMMON,PHD(5,5),PPEAF(7),PRBITY,PRDINO(5,7),PRKLD(5,7),
*PROBIL(840),PRSWD(5,7),RADFAE,RANGPR(840),RELNRD(5),RELSUB(5),
*REP(10),RN,ROTDMPX(10,10),ROTDMPY(10,10),RSP(5),S(100),SECON(7),
*SIGAD(50),SIGAR(50),SIGBD(50),SIGOR(50),STATAL(5,52),SWPDEL(100),
*SYMDU(5,7),SYMDIF(5,7),SYMDIS(7,7),SYMMAX(3),TARHT(5),TARL(5),
*TARRAD(5),TARW(5),TDBIF(10),TOTDEL(100),TGTQVL,TGTSPD,
*TGTVEL(100),TGTVL2,TGTYSV,TGTXL(100),TGTYNM(100),TGTYOL(100),
*TGTYSV,THETA,THRSIG(5,5),TIMENV(100),TMORD(5),TMORI(5),TMDLCF,
*THSWP(7),THTOFR(130),TOTSIM,TRAVTM,XCYCLE(7),XDMP(10,10),XFIX,
*XMAX,XMIN,XODEF(20),XOVP(10),
*XPDF(30),XROAD(1),XSWATH(50),XWIDTH,YDIV(100),YDMP(10,10),
*YLENGTH,YODEF(20),YOVP(10),YRDF(30),YROAD(1),YSWATH(50)

```

All variables which appear in these statements are individually defined in the List of Symbols and Abbreviations (Simulation Model).

#### DISCUSSION OF SIMULATION

The remainder of this section discusses in detail the coding of the various subroutines.

#### Program MAIN

Program MAIN exercises control over the simulation by calling the various processing subroutines in a manner defined by the program logic. The first executable statement:

```
KREAD = 0
```

initializes KREAD at the beginning of the job. The next statement:

```
5 CALL READIN (KREAD)
```

transfers control to Subroutine READIN. This subroutine reads the data for the first case (KREAD=0) and performs several iteration independent calculations. The statement:

```
20 CALL SETUP
```

is the return point for each iteration. Subroutine SETUP performs iteration-dependent calculations. The statements:

```
25 CALL ROAD  
CALL SORT(10R,NOB)  
IF(LEFTIN.EQ 0) GO TO 160
```

transfer control to Subroutine ROAD to initialize target travel on a travel path segment. Subroutine SORT performs a Singleton sort on the event information array [IOB(5000)] to order the array by increasing event Y coordinate. If all targets have been damaged or have breached the engagement area, transfer is made to determine if the last travel path segment has been traversed. The next statements:

```
30 CALL LOOPS  
TGTYSV=TGTVEL(IISAVE)  
IPO=IM  
ISYP=0  
KT = KTSAV(IISAVE)  
IWORD = ICHIKT  
CALL UNPACK  
OBYPR=OBY
```

call Subroutine LOOPS to determine the next event and unpack the information for the event. The statement:

```
CALL CKEVTM
```

provides a transfer to Subroutine CKEVTM to process all time oriented events involving direct and indirect fire. If the event target was damaged by direct or indirect fire, the next statement:

```
IF(TGTYNW(IISAVE).GT.99990) GO TO 110
```

removes this target from further consideration. Otherwise, the following statements:

```
IF(TGTYSV.GT.0.0) GO TO 50  
KTSAV(IISVLST)=KTSAV(IISVLST)-1  
GO TO 30
```

determine if the event target is in a delay condition and provides transfer to Subroutine LOOPS to determine the next event. The following statement:

```
50 GO TO (72,72,72,72,72,72,72,75,72,72,72,72,72,72,
*80,110,100,77) NOBTP2
```

provides transfers to the proper subroutine based on the event. If the event is one for a target/mine encounter, the next statement:

```
72 CALL TGTMIN
```

calls Subroutine TGTMIN to evaluate the target/mine interactions. If any targets are associated with other targets as a group and if a target is damaged in this event, the next statements:

```
IF (NGTAWT .GE. 1 .AND. ISVP .EQ. 2) CALL UNIT
GO TO 110
```

provide transfer to Subroutine UNIT to determine whether any unit movement of targets is required. If the event is one for a target/travel path boundary, the statements:

```
75 CALL EVENTC
GOTO 110
```

calls Subroutine EVENTC to evaluate the target/boundary event. The next statements:

```
77 IF (TGTXOLE (SAVE) .NE. 0 .OR. NTGTYP (SAVE) .NE. LCFTT) GO TO 150
CALL EXPSWP
GO TO 110
```

provide a call to Subroutine EXPSWP if the event target has reached a point along a travel path segment where line charge or fuel air explosive sweeping devices will be employed.

If the event target is leading the column that initiates an indirect fire volley, the following statements:

```
90 IF (IT.NE.NCTAM(IISAVE)) GO TO 150
   IVAP=IMINE
   CALL NDFIRE
   CALL BOOM(IDET,1)
   NDFA=1
   TOR(KT)=1
   NVLIDF(IVAP)=NVLIDF(IVAP)-1
   TMTOFR(IVAP*120)=TDBIFV(IVAP)
   GO TO 110
```

call Subroutine NDFIRE to evaluate the indirect fire volley at the aimpoint. Subroutine BOOM is called to set the flag bit on for this event to insure that the remaining volleys for this aimpoint arrive on a time basis. When a target has encountered a direct fire area exit boundary, the following statements:

```
100 CALL STINT(IISVIFA,IISAVE,IT,IBIT,KDFTN,2)
   IF (KDFTN(IISAVE).GT.0) GO TO 102
   NRFIRD(IISAVE)=0
   TMTOFR(IISAVE)=9990
102 JAA=0
   DO 105 J=1,30
   IF (ISVIFA(J)) 103,105
103 JAA=J
105 CONTINUE
   IF (JAA.LT.1) GO TO 109
   NDFA=MOD(JAA,15)
   IF (NDFA.LT.1) NDFA=15
   GO TO 110
109 NDFA=0
```

provide a call to Subroutine STINT to set the flag bit off for the event target in the direct fire area. This eliminates the target from any consideration by direct fire. This section of coding also determines the largest direct fire area number that contains at least one intruder. The next statements:

```
110 IF (NOBEVT.GT.0.AND.(IRUNS.EQ.IPRINT))
   * CALL PRINTO
   NOBEVT=IABS(NOBEVT)
```

transfer control to Subroutine PRINTO to print the optional output if required and if a mine is swept or detonated in the event. The next statements:

```
IF (NDFA) 130,150
130 JAA=0
```

```
DO 140 I=1,NVAP
IF INVLIDF(1111140,140,135)
135 JAA=1
140 CONTINUE
INDFA=JAA
```

determine if all volleys have been fired at every indirect fire aimpoint and sets the indirect fire flag. If all targets which are capable of sweeping have been damaged or have breached the minefield, the statements:

```
150 IF (NSPLFT.GT.0.OR.KNRS.NE.NRS) GO TO 155
NSPLFT=99999
DO 152 I=1,N100
IF (TGTYNWH(1) LT 99970) TSTVEL(I)=TGTVEL
152 CONTINUE
TGTSPD=TGTVEL
```

set the undamaged target velocities to normal speed. The next statement:

```
155 IF (LEFTIN.GT.0) GO TO 30
```

branches to Subroutine LOOPS if any targets have not completed traversing the travel path segment and have not been damaged. If all targets are damaged or have traversed the travel path segment, the statement:

```
160 IF (KNRS.NE.NRS) GO TO 25
```

branches to Subroutine ROAD if more travel path segments are to be traversed. Otherwise, the statements:

```
CALL PRINTR
IF (IRONS LT NOITI) GO TO 20
```

call Subroutine PRINTR to print the results for the iteration and, if more iterations are required, branch to Subroutine SETUP to prepare for the next iteration. If all iterations have been completed, the statements:

```
KREAD = 1
GO TO 5
END
```

transfer control to Subroutine READIN to read data cards for the next case, or the program is terminated if all cases have been processed.

### Subroutine READIN

This subroutine reads in the data cards and performs initial calculations which are independent of the iteration. The statements:

```
DIMENSION ORY(4),ORY(4),XYMS(4),RNGSV(3)
DIMENSION HOL(8),SECOFF(7),PROBTP(8),RANGTP(8)
DATA ITEST/5H 999/
```

provide local storage for input variables and initialize the value of ITEST. If this is the first call to Subroutine READIN, the statements:

```
REWIND 2
WRITE(6,1480)
IF IKREAD.EQ.11 GO TO 24
CALL REPRNT
8010 DO 7 I=1,840
PROB1(I) = 0.0
RANGPR(I) = 0.0
7 CONTINUE
ND=0
DO 8 J=1,15
DO 8 I=1,20
9 NDEFEA(I,J)=0
LCFOPT=0
MADIV=0
DO 9 I=1,5
NRAD(I)=NRAD(I)+0
TARRAD(I)=TARL(I)+TARH(I)+0.
REP(I)=REP(I+2)+0.
DEP(I)=DEP(I+2)+0.
NSUB(I)=1
DO 9 J=1,7
SYMDDF(I,J)=0.
9 SYMDIF(I,J)=0.
DO 10 I=1,7
TMSMP(I)=SECON(I)+SECOFF(I)+XCYLE(I)+0.
DO 10 J=1,7
10 SYMDIS(I,J)=0.
IEND= 0
```

call System Subroutine REPRNT to list the input data on the output device, and initialize several variables at the beginning of the job. The statements:

```
24 IEND= IEND+ 1
JMARK1=1
IRUNS=0
DO 23 I=1,100
23 DELATM(I)=0
```

```
00 25 I=1,7  
25 INTIME(1)=0
```

initialize variables at the beginning of each data set. The first card of each subdeck, which must be a title card, is read by the statements:

```
READ(4,9000) HOL  
DECODE(80,9000,HOL) INPUT  
IF (INPUT EQ 115) CALL EXIT
```

DECODE(N,M,HOL) is a Control Data Corporation system function which provides for the decoding of N characters of Hollerith information by Format M. A check is made on the first card of each subdeck to determine if it is a Card Type 999, which signals the end of the job. The statements:

```
00 38 J=1,52  
00 38 I=1,5  
STATAL(I,J)=0  
38 CONTINUE  
50 READ(4,9000) HOL  
9000 FORMAT(8A10)  
DECODE(5,9001,HOL) INPUT  
9001 FORMAT(15)
```

are executed next to initialize variables, read a data card, and decode the card type. Branches to the appropriate section of coding to obtain the data values on the card are provided by the statements:

```
IF (INPUT EQ 99) GO TO 7000  
IF (INPUT GT 1000) GO TO 5900  
GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,  
1500,1600,1700,1800,1900,2000,2100,2200,2300,2400) INPUT
```

The following sections are similar in that they decode input data on a particular card. For Card Type 1, the decoding statements are:

```
100 DECODE(80,9002,HOL) NAP,NTGO,NTGTP,NOIT,NOSTAT,IPRINT,SEED,  
*NMT,MODE,MVAP,NDFAA,NDFHD,NRBA  
9002 FORMAT(5X,6I5,F10.2,6I5)  
IF (SEED LE 0) SEED=RANF(DUMMY)  
CALL RANSET(SEED)  
GO TO 90
```

If the value of SEED is zero, a call to RANF (the Control Data Corporation system uniform random number generator) is performed to determine a starting value. The call to RANSET causes RANF to start with the value of SEED. For Card Types 2, 3, and 4, the decoding statements are:

```

200 DECODE (80,9003,HOL,ILCFOPT,YLENGTH,XWIDTH,THETA,D3DEL,
* (ISYMP(1),1-1,3),ISBL,ITEROP,IDISOP
9003 FORMAT (5X,15,2F10.2,2F5.2,415,12,13)
GO TO 50
300 DECODE (80,9004,HOL,INTN,NTGTYP(INTN),NCTAW(INTN),NGTAW(INTN),
* TGTVOL(INTN),TGTVOL(INTN),N1,N2,N3,N4,T1,T2
IF (N1.LT.1) GO TO 50
NTGTYP(N1)=N2
NCTAW(N1)=N3
NGTAW(N1)=N4
TGTVOL(N1)=T1
TGTVOL(N1)=T2
9004 FORMAT (215X,15,11,14,15,2F10.21)
GO TO 50
400 DECODE (80,9005,HOL,ITYP,TARL(ITYP),TARW(ITYP),TARRAD(ITYP),
* TARH(ITYP),TMBRI(ITYP),NRAI(ITYP),LDFDPR(ITYP),1-1,5),
* N:ITCS(ITYP),NTTMO(ITYP),NTTIRP(ITYP)
IF (NTTMO(ITYP).GT.0) MADIV=1
9005 FORMAT (5X,15,4F10.2,F5.2,15,511,315)
GO TO 50

```

For Card Types 5, 6, 7, 8, and 9, the decoding statements are:

```

500 DECODE (80,9091,HOL,IVAP,IROAD,XOVP(IVAP),YOVP(IVAP),AYVVP(IVAP),
* NVFAEA(IVAP),NMEPV(IVAP),IMTVAP(IVAP),NTLCIF(IVAP),TDBIFV(IVAP)
9091 FORMAT (10X,215,3F10.2,415,F10.2)
IVPO=IFIX(XOVP(IVAP)+100.)
XOVP(IVAP)=FLOAT(IVPO+100)*IROAD*ISIGN(1,IVPO)
GO TO 50
600 DECODE (80,9090,HOL,IVAP,IMEPV,DMP1IX(IVAP,IMEPV),
* DMP1IY(IVAP,IMEPV)
9090 FORMAT (10X,215,2F10.21)
GO TO 50
700 DECODE (80,9015,HOL,IVAP,IMT,ANGIMP(IVAP),REP(IVAP),DEP(IVAP),
* NSUB(IMT),RELSUB(IMT),PATRAD(IMT),RELRND(IMT)
9015 FORMAT (5X,215,F5.2,2F10.2,15,3F10.21)
GO TO 50
800 DECODE (80,9089,HOL,NRS,TGTOVL,TGTVL2
9089 FORMAT (10X,15,2F10.2)
TGTOVL=TGTOVL*88.
TGTVL2=TGTVL2*88.
N=NRS+1
READ(4,9007)(XROAD(1),YROAD(1),1-1,N)
9007 FORMAT (8F10.2)
GO TO 50
900 DECODE (80,9013,HOL,IAP,MFA(IAP),NSTICK(IAP),AIMPTX(IAP),
* AIMPTY(IAP),YSWATH(IAP),XSWATH(IAP)
9013 FORMAT (5X,315,4F10.2)
GO TO 50

```

The value 88 is used to convert miles per hour to feet per minute. For Card Types 10 and 11, the decoding statements are:

```

1000 DECODE (80,9012,HOLI IAP,SIGAR:IAP),SIGAD:IAP),SIGBR:IAP),SIGDD:IAP)
9012 FORMAT(5X,15,4F10,2)
GO TO 50
1100 DECODE(80,9009,HOLI MT,JSELDS(MT),TMSWP(MT),DUDPRB(MT),PPEAF(MT)
9009 FORMAT(10X,215,3F10,2)
IF(JSELDS(MT).GT.0) GO TO 50
DO 902 M=1,NAP
IF(MTFA(M),EQ,MT) GO TO 903
902 CONTINUE
903 J=0
905 READ(4,9007) (ORX(I),ORY(I),I=1,4)
DO 910 I=1,4
WRITE(2) ORX(I),ORY(I),MT
IF(IJ=1,EQ,NSTICK(M)) GO TO 50
910 CONTINUE
J=J+4
GO TO 905

```

If the value of JSELDS(MT) is zero, the nominal X and Y coordinates of each mine (measured relative to the sortie aimpoint) are read in and written on Tape 2. For Card Types 12, 13, and 14, the decoding statements are:

```

1200 DECODE(80,9010,HOLI MT),(SYMDIS(MT),J),J=1,7)
9010 FORMAT(10X,15,7F5,2)
GO TO 50
1300 DECODE(80,9010,HOLI MT),(SYMDDF(MT),J),J=1,7)
GO TO 50
1400 DECODE(80,9010,HOLI MT),(SYMDIF(MT),J),J=1,7)
GO TO 50

```

For Card Types 15, 16, and 17, the decoding statements are:

```

1500 DECODE (80,9011,HOLI MT,SECON(MT),SECOFF(MT),KOUNT(MT),IRNK(MT)
9011 FORMAT (10X15,2F5,2,215)
INTIME(MT)=1
GO TO 50
1600 DECODE(80,9094,HOLI NUMDEF,INTDEF(NUMDEF),XODEF(NUMDEF),
*YDEF(NUMDEF),INDEF(A(J,NUMDEF)),J=1,15)
9094 FORMAT(5X,215,2F10,2,15I2)
GO TO 50
1700 DECODE(80,9093,HOLI NDF,IRROAD,(XYMS(I),I=1,4)
9093 FORMAT(5X,215,4F10,2)
L=0
DO 1705 I=1,2
ND=ND+1
IXY1=IFIX(XYMS(I+L)*100.)
XRDF0(ND)=FLOAT(IXY1*ISIGN(1,IXY1))*IRROAD
IXY2=IFIX(XYMS(I+2)*100.)
YRDF0(ND)=FLOAT(IXY2*ISIGN(1,IXY2))*NDF
L=L+1
1705 CONTINUE
GO TO 50

```

For Card Types 18, 19, and 20, the decoding statements are:

```

1800 DECODE(80,9016,MOL) 11,J1,PHD(11,J1),12,J2,P2,13,J3,P3,
*14,J4,P4,15,J5,P5
9016 FORMAT(5X,3(215,F5.2))
IF(12,LT,1) GO TO 90
PHD(12,J2)=P2
IF(13,LT,1) GO TO 50
PHD(13,J3)=P3
IF(14,LT,1) GO TO 50
PHD(14,J4)=P4
IF(15,LT,1) GO TO 50
PHD(15,J5)=P5
GO TO 50
1900 DECODE(80,9017,MOL) 11,J1,N1,EV1,12,J2,N2,EV2,13,J3,N3,EV3
9017 FORMAT(5X,3(315,F10.2))
EI(11,J1,1)=N1
EI(11,J1,2)=EV1
IF(12,LT,1) GO TO 50
EI(12,J2,1)=N2
EI(12,J2,2)=EV2
IF(13,LT,1) GO TO 50
EI(13,J3,1)=N3
EI(13,J3,2)=EV3
GO TO 50
2000 DECODE(80,9050,MOL) NDFTYP,DEFL(NDFTYP),DEFM(NDFTYP),
*DEFRAD(NDFTYP),DEFHT(NDFTYP),THBRD(NDFTYP),NRAD(NDFTYP),
*(1TGTPR(NDFTYP,1),1-1,5)
9050 FORMAT(5X,15,4F10.2,F5.2,15,511)
GO TO 50

```

The following statements decode Card Types 21, 22, 23, and 24.

```

2100 DECODE(80,9091,MOL) 11,A11,AREP1,ADEP1,12,A12,AREP2,ADEP2
9091 FORMAT(5X,15,3F10.2,5X,15,3F10.2)
A1(11)=COS(A11*.01745329)
AREP(11)=AREP1
ADEP(11)=ADEP1
IF(12,LT,1) GO TO 50
A1(12)=COS(A12*.01745329)
AREP(12)=AREP2
ADEP(12)=ADEP2
GO TO 50
2200 DECODE(80,9092,MOL) 1,J,K,N,AEV,AREL1,ACEP1
9092 FORMAT(5X,415,F10.2,F5.2,F10.2)
AEI(1,J,2*K-1)=N
AEI(1,J,2*K)=AEV
L=1
IF(K,GT,1) L=J
AREL(L,K)=AREL1
IF(N,GT,1) ACEP(L,K)=ACEP1
GO TO 50
2300 DECODE(80,9093,MOL) 11,J1,K1,IDP1,12,J2,K2,IDP2,13,J3,K3,IDP3
9093 FORMAT(5X,3(415,5X1))
IDP(11,J1,K1)=IDP1
IF(12,LT,1) GO TO 50
IDP(12,J2,K2)=IDP2
IF(13,LT,1) GO TO 50
IDP(13,J3,K3)=IDP3
GO TO 50
2400 DECODE(80,9029,MOL) LCFTT,DBFAE,DFTFAE,RADFAE,ALNGLC,
*AMIDL,THDLCF,PDLCF
9029 FORMAT(5X,15,7F10.2)
GO TO 50

```

For card types greater than 1000, the decoding statements are:

```

6900 DECODE(89,9008,MOL) NT,NM,NTB,(PROBTP(1),1-.8)
9008 FORMAT(IX,11,12,11,F5.2,7F10.2)
READ(4,9008) NT,NM,NTB,(RANGTP(1),1-.8)
NTABLE=(NT-1)*2+(NM-1)*3*NTB
LL=8*(NTABLE-1)
DO 6910 I=1,8
PROBIL(LL+1)=PROBTP(I)
6910 RANGPR(LL+1)=RANGTP(I)
GO TO 50

```

Transfer is made to statement 6900 for input of each probability function. The target type, mine type, and table number are decoded, and the probability and corresponding range values are stored in appropriate array locations. When a Card Type 99 signals the end of the data for the subdeck, the following section performs various calculations required for the case. The statements:

```

7000 DO 7005 I=1,NMT
NVCYCLE(I)=SECON(I)+SECOFF(I)
NMN(I)=0
DO 7005 J=1,NAP
IF(MTFA(J).EQ.1) NMN(I)=NMN(I)+NSTICK(J)
7005 CONTINUE
IRTFIR=0
DO 7006 I=1,5
IF(NRAI(I).LT.1) GO TO 7006
IRTFIR=1
7006 NITBT(I)=0
NGTAWT=0
DO 7008 I=1,NTGO
N=NTGTYPI(I)
NGTAWT=NGTAWT+NCTAN(I)
7008 NITBT(N)=NITBT(N)+1

```

compute the duration of the fuze timing cycle (if any), the total number of mines emplaced by type, and the total number of targets by type. If sympathetic detonations for mines are being evaluated, the statements:

```

IF(SYMP(I).LT.1) GO TO 7015
SYHMAX(I)=0
DO 7010 I=1,NMT
DO 7010 J=1,7
SYMDIS(I,J)=SYMDIS(I,J)**2
7010 SYHMAX(I)=AMAX1(SYHMAX(I),SYMDIS(I,J))
SYHMAX(I)=SQRT(SYHMAX(I))
7015 RSPHAX=0

```

compute the maximum distance at which sympathetic detonations can occur. If sympathetic detonations are being evaluated for direct fire rounds, the following statements:

```

IF (SYMP(2).LT.1) GO TO 7017
SYMMAX(2)=0
DO 7016 I=1.5
DO 7016 J=1.7
SYMDDF(I,J)=SYMDDF(I,J)**2
7016 SYMMAX(2)=AMAX1(SYMMAX(2),SYMDDF(I,J))
SYMMAX(2)=SQRT(SYMMAX(2))

```

compute the maximum distance at which sympathetic detonations can occur. If sympathetic detonations for indirect fire rounds are being evaluated, the following statements:

```

7017 IF (SYMP(3).LT.1) GO TO 7019
SYMMAX(3)=0
DO 7018 I=1.5
DO 7018 J=1.7
SYMDIF(I,J)=SYMDIF(I,J)**2
7018 SYMMAX(3)=AMAX1(SYMMAX(3),SYMDIF(I,J))
SYMMAX(3)=SQRT(SYMMAX(3))

```

compute the maximum distance at which sympathetic detonations can occur. The next section of coding:

```

7019 IF (NMT.LT.1) GO TO 7045
DO 7040 I=1,NTGTP
RSP(I)=0
DO 7035 J=1,NMT
NT=(I-1)*21+(J-1)*3
DO 7030 K=1.3
NTI=8*(NT+K-1)
DO 7070 L=1.8
IF (PROBIL(NTI*L).EQ.0.) GO TO 7022
7070 CONTINUE
7022 RNGSV(K)=RANGP(NTI*L)
7030 CONTINUE
PRSHD0(I,J)=RNGSV(1)
PRK0(I,J)=RNGSV(2)
PRDINO(I,J)=RNGSV(3)
RSP(I)=AMAX1(RSP(I),RNGSV(1),RNGSV(3))
RSPMAX=AMAX1(RSPMAX,RNGSV(1),RNGSV(3))
7035 CONTINUE
IF (MADIV.GT.0) RSP(I)=RSP(I)+30.
7040 CONTINUE
IF (MADIV.GT.0) RSPMAX=RSPMAX+30.

```

determines the minimum ranges at which each of the input probability functions are zero and computes values for the range of influence for each target type and the maximum range of influence. The largest possible diversion offset distance (30 feet) is added to the range of influence if target diversion is being considered. The next group of statements:

```

7045 XMIN=XMAX+0.
      DO 7060 I=1,5
7060 NIDBT(I)=0
      NDFWT=0
      IF (NDFWD.LT.1) GO TO 7085
      DO 7075 I=1,NDFWD
      N=INTDEF(I)
7075 NIDBT(N)=NIDBT(N)+1
      DO 7080 I=1,5
      IF (NIDBT(I).GT.0) NDFWT=NDFWT+1
7080 CONTINUE
7085 CONTINUE
      DO 7100 I=1,NTGD
      XMAX=AMAXI(XMAX,IGTXOL(I))
7100 XMIN=AMINI(XMIN,IGTXOL(I))
      XMAX=XMAX+RSPMAX
      XMIN=XMIN-RSPMAX
      IF (XMAX.GT.1638.) XMAX=1638.
      IF (XMIN.LT.-1638.) XMIN=-1638.

```

determine the number of defender targets by type, the total number of defender weapon types, and the maximum and minimum X coordinates within which mines will be retained for the case. The following statements:

```

      IF (NVAP.LT.1) GO TO 9998
      DO 8200 I=1,NVAP
      COSA=COS(AYYP(I))*0.1745329258
      SINA=SIN(AYYP(I))*0.1745329258
      XORGVP=FLOAT(IFIX(XOVP(I)/100.)) / 100.
      YORGVP=YQVP(I)
      NHV=NHVP(I)
      IF (NHV.LT.1) GO TO 8200
      DO 8190 J=1,NHV
      XDMP(I,J)=XORGVP+DMP1IX(I,J)*COSA+DMP1IY(I,J)*SINA
      YDMP(I,J)=YORGVP+DMP1IY(I,J)*COSA-DMP1IX(I,J)*SINA
8190 CONTINUE
8200 CONTINUE

```

transform the desired mean points of impact for each round in each indirect fire volley into the map coordinate system [Equations (38) and (39)]. The last group of statements in the subroutine:

```

      DO 8500 I=1,NTGTP
      AM=AMAXI(TARL(I),TARW(I))
      DO 8500 J=1,NVAP
      K=INTVAP(J)
      THRSIG(K,I)=14.44*AMAXI(REP(J),DEP(J))+AM*SQRT(EI(K,I,2))**2
8500 CONTINUE
9998 RETURN
1480 FORMAT(IH1)
      END

```

compute the distances beyond which the effects of an indirect fire round are not considered.

## Subroutine SETUP

This subroutine performs iteration-dependent calculations and determines the mine X and Y coordinates. The first executable statements:

```

      II = 0
      TGTSPD = TGTVL2
      IF (MODE .LT. 3) TGTSPD = TGT0VL
      REWIND 1
      REWIND 2
      IRUNS = IRUNS + 1
      DO 300 I = 1, 5
      NKILL(I) = NKILLD(I) - NREBIT(I) - NREBDT(I) - NDEFAM(I) + 0
300  NKILL(I) = 0
      DO 305 I = 1, NVAP
      NVLIDF(I) = NVFAEA(I)
305  CONTINUE
      DO 310 I = 1, 7
      NMSHPT(I) = NMDET(I) - NIFBT(I) + 0
      DO 315 J = 1, 20
      DO 315 I = 1, 15
315  NDEFEA(I, J) = IABS(NDEFEA(I, J))
      DO 320 J = 1, 15
      ISVIFA(J + 15) = 0
320  ISVIFA(J) = 0
      DO 325 J = 1, 130
325  TMTQFR(J) = 9990.
      DO 330 I = 1, 100
      KDFTN(I) = 0
      NRFIRD(I) = 0
330  DELATM(I) = 0.
      NKILLT = NSPLFT = 0
      NTLOST = 0
      NOBEVT = 1
      THET = .017453295 * THETA
      SINT = SIN(THET)
      COST = COS(THET)
      KNRS = 0
      XM2 = XWIDTH / 2.0
      YL2 = YLENGTH / 2.0
      DO 210 I = 1, 547
210  XABOOM(I) = 0
      NOB = 0
      NOBTP2 = R
      IT = 0
      N = NRS + 1
      NOB = NOB + N

```

initialize variables and set to zero the 547 words containing the mine detonation flag bits. The next statements:

```

      DO 200 I = 1, N
200  WRITE(I) XROAD(I), YROAD(I), NOBTP2, IT

```

write the X and Y coordinates of the travel path end points on Tape 1. Next, a double-nested DO LOOP is entered to determine the mine locations for the iteration. The statements:

```

IF(NAP.LT.1) GO TO 160
DO 146 J=1,NAP
147 CALL RNORM(RN1)
CALL RNORM(RN2)
REWIND 2
MT=MTFA(J)
IT=0
N=NSTICK(J)
DO 146 I=1,N
CALL RNORM(RN3)
CALL RNORM(RN4)
NOBTP2=MT

```

select two normal random numbers for use with the aiming error standard deviations. The mine type for the sortie aimpoint is determined, and, inside the inner DO LOOP, two normal random numbers are selected for use with the ballistic error standard deviations. If the nominal mine locations about the sortie aimpoint are read in from cards, the statements:

```

IF(JSELDS(MT)) 120,130
130 READ(2) ORX,ORY,NOBTP2
IF(NOBTP2.NE.MT) GO TO 130
GO TO 150

```

read the nominal mine X and Y coordinates from Tape 2. Otherwise, the statements:

```

120 GO TO(122,125,125,122) JSELDS(MT)
122 CALL RNORM(RSTART)
ORX=XSWATH(J)/6.*RSTART
IF(JSELDS(MT)=4) 123,126,123
123 CALL RNORM(RSTART)
ORY=YSWATH(J)/6.*RSTART
GO TO 150
125 ORX=XSWATH(J)*(RANF(DUMMY)-.5)
IF(JSELDS(MT)=3) 126,123,126
126 ORY=YSWATH(J)*(RANF(DUMMY)-.5)

```

determine the mean points of impact in range and deflection from either normal or uniform distributions of mines depending upon the value of JSELDS(MT). For normal distributions, the values of XSWATH(J) and YSWATH(J) contain six standard deviations; therefore, these values are divided by the value 6 to obtain the one-sigma values in the statements:

```

150 ORX=ORX*RN1*SIGAD(J)+RN3*SIGBD(J)
ORY=ORY*RN2*SIGAR(J)+RN4*SIGBR(J)
OBX=AIMPTX(J)+ORX*COST+ORY*SINT
OBY=AIMPTY(J)-ORX*SINT+ORY*COST

```

The mine X and Y coordinates with respect to the sortie aim-point are computed [Equations (1) and (2)] and are then transformed into the map coordinate system [Equations (5) and (6)]. If the mine location is within the rectangle defining the area through which the targets are to travel, the statements:

```

IF (ABS(OBX) .GT. XM2 .OR. ABS(OBY) .GT. YL2) GO TO 146
143 RN = RANF ( DUMMY )
IF (RN-DUDPRB(MT)) 20,20,25
20 NOBTP2=NOBTP2+B

```

compare a uniform random number to the probability of mine malfunction [Equation (7)]. If the mine is a dud, the quantity 8 is added to the mine type to reflect this condition. If the mine has a fuze timing cycle or target counting capability, the statements:

```

25 IF (INTIME(MT).EQ.0) GO TO 145
RN = RANF ( DUMMY )
IT=RN*XCYLE(MT)
IF (KOUNT(MT).GT.0) IT=KOUNT(MT)
IF (IRNK(MT).GT.0) IT=IFIX(IT*RN)+1

```

compute the random starting point for the mine [Equation (10)] or set the target count at which the mine will arm. This can be a random integer number of target counts varying from 1 up to a preset maximum. If Tactic 1 is being evaluated and if the mine is a dud, the mine is discarded. Otherwise, the statements:

```

145 IF (MODE.EQ.1 .AND. NOBTP2 .GT. 8) GO TO 146
NOB = NOB + 1
WRITE (1) OBY, OBY, NOBTP2, IT
146 CONTINUE
IF (NOB .LE. 32767) GO TO 160
PRINT 1000
1000 FORMAT ( ' MORE THAN 32767 MINES IN MINEFIELD.' )
CALL EXIT

```

write the mine coordinates, mine type, and fuze timing cycle starting point or target count for arming (if any) on Tape 1. The final statements:

```

160 NOBTP2=0
WRITE(1) OBX, OBY, NOBTP2, 11
100 MUSH = 0
TOTSIM = 0.0
DO 100 I=1, NTGO
N=NTGTYP(I)
NRADF(I)=NRAI(N)
IF(NTTICS(N).EQ.1) NSPLIT=NSPLFI+1
99 TIMENY(I)=0.
TGTDEL(I)=0.
SNPDEL(I)=0.
NGTAW(I)=NGTAW(I)
TGTVEL(I)=TGTSPD
100 CONTINUE
IF(NDFWD.LT.1) RETURN
DO 110 J=1, NDFWD
N=INTDEF(J)
NRADF(J)=NRADIN
110 CONTINUE
RETURN
END

```

write a dummy record on Tape 1, initialize certain variables, and determine the number of targets which are capable of sweeping.

#### Subroutine ROAD

This subroutine transforms the mine locations into the travel path coordinate system and places the targets at the beginning of the travel path segment. The first executable statements:

```

DIMENSION XLCF(100)
NDFA=INDFA=0
REWIND 1
KNRS=KNRS+1
DO 10 I=1, KNRS
10 READ(1) XRI, YRI, NOBTP2, 11
READ(1) XR2, YR2, NOBTP2, 11
20 READ(1) ORX, ORY, NOBTP2, 11
IF(INAP.LT.1) GO TO 25
IF(NOBTP2.EQ.8) GO TO 20

```

read the X and Y coordinates of the end points of a travel path segment from Tape 1 and read the tape until the first mine coordinate record is found. The next statements:

```

25 N2=NOBTP2
LIT=11
A1=YR2-YRI
A2=XR2-XRI
IF(A2.EQ.0.) A2=.000001
THETR=ATAN2(A1, A2)-3.141592654/2.
COSTR=COS(THETR)
SINTR=SIN(THETR)

```

compute the sine and cosine of the angle that the directed travel path segment makes with the X axis of the map coordinate system. The statements:

```

OBX=0.0
OBY=0.0
NOBTP2=0
IT=IMINE*0
CALL IPACK
IOB(1)=IWORD
YLENTH=OBY*SQRT(A2**2+A1**2)
CALL IPACK
NOB=2
IOB(2)=IWORD
NOBTP2=N2
IT=LIT
DO 45 I=1,130
45 TMTOFR(I)=9999.0
YMIN=99999
YMAX=0.

```

pack the travel path segment boundary information into the IOB(5000) array and initializes the TMTOFR(130) array which is used to control the time at which direct or indirect fire rounds may be fired. A DO LOOP is next entered to consider each mine on Tape 1. The statements:

```

DO 50 IMINE=1,32767
IF (NOBTP2.EQ.0) GO TO 100
OBX=(ORX-XR1)*COSTR+(ORY-YR1)*SINTR
OBY=-(ORX-XR1)*SINTR+(ORY-YR1)*COSTR

```

transform the mine X and Y coordinates into the travel path coordinate system [Equations (8) and (9)]. If the mine lies within the travel path segment boundaries and within the range of influence of at least one target in the formation, the statements:

```

IF(OBY.GT.YLENTH.OR.OBY.LE.0) GO TO 40
IF(OBX.LT.XMIN.OR.OBX.GT.XMAX) GO TO 40
YMIN=AMIN(YMIN,OBY)
YMAX=AMAX(YMAX,OBY)
IDET=0
IF(KNRS.GT.1) CALL BOOM(IDET,2)

```

call Subroutine BOOM to determine if the mine was detonated while the targets were traversing a previous travel path segment (if any). If the mine was not previously detonated, the statements:

```

IF (IDET EQ 1) GO TO 40
CALL IPACK
M=MOD(NOBTP2,B)
IF (NOBTP2 GT 15100 TO 29
MFBT(M)=MFBT(M)+1
29 NOB=NOB+1

```

pack the mine information into the variable IWORD. If more than 5,000 mines have been retained for the travel path segment, the statements:

```

IF (NOB LT 5001) GO TO 30
PRINT 1000
1000 FORMAT('MORE THAN 5000 MINES WITHIN RANGE OF INFLUENCE FOR A ROAD
SEGMENT',/20(1H/), 'PROGRAM STOPS',/20(1H/))
CALL EXIT

```

print an error message, and the program stops. Otherwise, the statements:

```

30 IOB(NOB)=IWORD
40 READ(1) DRX,DRY,NOBTP2,11
50 CONTINUE

```

place the mine information into the IOB(5000) array and read another record from Tape 1. When all mines have been processed, the statements:

```

100 DO 120 I=1,NTGO
KTSAV(I)=1
IF (KNRS EQ 1) GO TO 110
IF (TGTYNW(I) GT .99994) GO TO 120
110 TGTYNW(I)=TGTYOL(I)
TGTVEL(I)=TGTSPD
IF (KNRS EQ 1) GO TO 120
TADJ=ABS(TGTYNW(I)/TGTVEL(I))
DELTM=TGTDEL(I)*SMPDEL(I)
TIMEHV(I)=TIMEHV(I)-TADJ*DELTM
TGTYNW(I)=TGTYNW(I)-TGTVEL(I)*DELTM
120 CONTINUE

```

initialize the target positions and velocities and adjust the target travel times at the beginning of the second and subsequent travel path segments. The following statements:

```

TGTYSV=TGTYOL(I)
LEFTIN=NTGO-MKILL-NTLOST
MUSH=0
NDVT=0
DO 121 I=1,100
121 S(I)=0
ISVLS(I)=1
KTSAV(I)=0

```

initialize several variables. If indirect fire is employed on this travel path segment, the following statements:

```

IF (NVAP.LT.1) GO TO 135
IMINE=IT+0
NOBTP2=16
DO 130 J=1,NVAP
DIRATK(J)=AYVVP(J)*.017453293-TMETR
IRD=AMOD(XOVP(J),100.)
IROAD=IABS(IRD)
IF (IROAD.NE.KNRS) GO TO 130
IT=NTLCIF(J)
IMINE=J
XR=(XOVP(J)-IROAD)/10000.
YR=YDVP(J)
OBX=-(XR-XR1)*COSTR+(YR-YR1)*SINTR
OBY=-(XR-XR1)*SINTR+(YR-YR1)*COSTR
IF (OBY.GT.YLENTH.OR.OBY.LE.0.) GO TO 125
IF (OBX.LT.-1638 .OR.OBX.GT.1638.) GO TO 125
CALL IPACK
NOB=NOB+1
IF (NOB.LT.5001) GO TO 124
PRINT 1000
CALL EXIT
124 IOB(NOB)=IWORD
125 CONTINUE
130 CONTINUE
DO 134 J=1,NVAP
NHV=NHVP(J)
IF (NHV.LT.1) GO TO 134
DO 133 K=1,NHV
ROTDMPX(J,K)=(XDMPI(J,K)-XR1)*COSTR+(YDMP(J,K)-YR1)*SINTR
ROTDMPY(J,K)=(XDMPI(J,K)-XR1)*SINTR+(YDMP(J,K)-YR1)*COSTR
133 CONTINUE
134 CONTINUE

```

transform the origin of the volley aimpoints into the travel path coordinate system [Equations (36) and (37)] and pack the information into the variable IWORD. The desired mean points of impact are also transformed into the travel path coordinate system [Equations (40) and (41)] and stored in the arrays ROTDMPX(10,10) and ROTDMPY(10,10). The next group of statements:

```

135 IMINE=0
IT=0
L=0
IF (ND.LT.1) GO TO 160
OBX=0
DO 150 I=1,ND
IRD=AMOD(XRDFO(I),100.)
IRD=IABS(IRD)
IF (IRD.NE.KNRS) GO TO 150
XR=FLOAT(IFIX(XRDFO(I)/100.))
YR=FLOAT(IFIX(YRDFO(I)/100.))
OBY=-(XR-XR1)*SINTR+(YR-YR1)*COSTR
IF (OBY.GT.YLENTH.OR.OBY.LE.0.) GO TO 150
NOBTP2=17
IF (L=149,147)
147 L=1
IT=IFIX(AMOD(YRDFO(I),100.))
IT=IABS(IT)
148 CALL IPACK
NOB=NOB+1
IF (NOB.LT.5001) GO TO 140
PRINT 1000
CALL EXIT

```

```

149 NOBTP2=18
    L=0
    GO TO 148
140 IOB(NOB)=IWORD
150 CONTINUE

```

transform the boundaries of the direct fire areas into the travel path coordinate system [Equation (24)] and pack the information into the variable IWORD. The statements:

```

350 IF(INDFAA.LT.1) GO TO 400
    DO 350 I=1,NDPWO
    DEFY(I)=(XODEF(I)-XRI)*COSTR+(YODEF(I)-YRI)*SINR
    DEFY(I)=(XODEF(I)-XRI)*SINR+(YODEF(I)-YRI)*COSTR
350 CONTINUE

```

transform the coordinates of the position of the defenders into the travel path coordinate system. The last statements in the subroutine:

```

400 IF(LCFOPT.LT.1) RETURN
    NLCFCT=0
    DO 410 I=1,N1CO
    IF(NIGTYP(I).NE.NLCFCT) GO TO 410
    NLCFCT=NLCFCT+1
    XLCF(NLCFCT)=TGTXYL(I)
410 CONTINUE
    DLCF=ALNGLC
    IF(LCFOPT.GT.1) DLCF=DBPAE
    IT=0
    IMINE=0
    NOBTP2=19
    OBY=YMIN-2.*DLCF
    OBY=AMAXI(OBY,0.)
420 OBY=OBY+DLCF
    IF(OBY.GT.YMAX) GO TO 500
    DO 430 I=1,NLCFCT
    OBY=XLCF(I)
    CALL IPACK
    NOB=NOB+1
    IOB(NOB)=IWORD
    IF(NOB.LT.500) GO TO 430
    PRINT 1000
    CALL EXIT
430 CONTINUE
    GO TO 420
500 RETURN
    END

```

determine the position along the travel path segment that the line charges or fuel air explosives will be placed and pack this information into the variable IWORD.

### Subroutine BOOM

This subroutine sets and checks the mine detonation flag bits contained in the KABOOM(547) array. The first executable statements:

```
IWD=(IMINE-1)/60+1
K=MOD(IMINE,60)+1
IF(K.EQ.-1) K=59
```

compute the word and bit position within the word for the mine being considered. If the call to Subroutine BOOM was made for the purpose of recording that a mine has detonated, the statements:

```
IF(N.GT.1) GO TO 200
100 I=1
KABOOM(IWD)=KABOOM(IWD).OR.SHIFT(1,K)
RETURN
```

shift a 1 into the appropriate bit position. If the status of a mine is to be checked, the statements:

```
200 IDET=SHIFT(KABOOM(IWD),K).AND.1B
RETURN
END
```

place the bit, whether ON or OFF, into the variable IDET for interrogation in the calling routine.

### Subroutine SORT

The purpose of this subroutine is to sort the IOB(5000) vector into increasing order by a method developed by Richard C. Singleton. Details pertaining to the sorting technique can be found in Algorithm 347 of the Collected Algorithms From the Communications of the Association for Computing Machinery. The listing is presented here for completeness.

```
INTEGER A(N),IU(16),IL(16),T,IT
M = 1
I = 1
J = N
5 IF(I-J) 10,70,70
10 K = I
IU = (J+1)/2
IT = A(IJ)
```

```

      IF(A(I)-T) 20,20,15
15  A(I) = A(J)
      A(I) = T
      T = A(J)
20  L = J
      IF(A(J)-T) 23,40,40
23  A(I) = A(J)
      A(J) = T
      T = A(J)
      IF(A(I)-T) 40,40,27
27  A(I) = A(J)
      A(I) = T
      T = A(J)
      GO TO 40
30  A(L) = A(K)
      A(K) = T
40  L = L - 1
      IF(A(L)-T) 35,35,40
35  T = A(L)
50  K = K + 1
      IF(A(K)-T) 50,53,53
53  IF(K-L) 30,30,55
55  IF((L-1)-(J-K)) 60,60,57
57  IL(M) = I
      IU(M) = L
      I = K
      M = M + 1
      GO TO 80
60  IL(M) = K
      IU(M) = J
      J = L
      M = M + 1
      GO TO 80
70  M = M - 1
      IF(M) 75,110,75
75  I = IL(M)
      J = IU(M)
80  IF(J-I-1) 85,10,10
85  IF(I-1) 87,5,87
87  I = I - 1
90  I = I + 1
      IF(I-J) 93,70,93
93  T = A(I+1)
      IF(A(I)-T) 90,90,97
97  K = I
100 A(K+1) = A(K)
      K = K - 1
      IF(T-A(K)) 100,105,105
105 A(K+1) = T
      GO TO 90
110 RETURN
      END

```

### Subroutine LOOPS

The purpose of this subroutine is to determine which target is involved in the next event. The event information in the IOB(5000) array has already been sorted so that the next event that a target will be involved in can be determined by incrementing the appropriate location in the KTSAV(100) array. The first executable statement:

```
KTSAV(I:SVLST) = KTSAV(I:SVLST) + 1
```

increments the counter for the target that was considered in the last event. A DO LOOP is entered to consider each target, and the statements:

```
DO 250 I=1,NTGO
IF(TGTYNW(I)-100000.1110.110.100
100 TGTYNW(I)=99999.
TMTOPR(I)=9990.
DIRDIS(I)=99999.
GO TO 250
110 DIRDIS(I)=99999.
```

remove the target from further consideration if it was damaged in the last event. If the target has not been damaged, the statements:

```
IF(TGTYNW(I).GT.99970.1 GO TO 250
140 KT=KTSAV(I)
IF(LOB(KT)+1) 210,205
205 KTSAV(I)=KTSAV(I)+1
GO TO 140
```

determine whether the next event that the target will be involved in has previously occurred. If it has been, the counter is incremented, and the determination is made again. Otherwise, the statements:

```
210 IWORD=LOB(KT)
CALL UNPACK
N=NTGTYP(I)
IF(NOBTP2.GT.151 GO TO 240.
M=MOD(NOBTP2,8)
```

unpack the event information. If a target/mine encounter is being evaluated, the statements:

```
IF(N) 212,240
212 IF(PRSWDO(N,M))220,215
215 IF(PRDTNO(N,M))220,205
220 XDIS=ABS(OBX-TGTXL(I))
IF(XDIS-RSP(N))240,205,205
```

compute the distance between the target and the mine when the target is at the point of closest approach to the mine [Equation (12)]. This computation is performed only if the mine type can be detected or detonated by the target. If the

mine is outside of the range of influence of the target, a transfer is made to consider the next mine. Otherwise, the statements:

```
240 DIRDIS(1)=OBY-TGTYNW(1)
250 CONTINUE
```

compute the range distance which must be traveled by the target to reach the point of closest approach to the mine [Equation (11)]. After all active targets have been considered, the statements:

```
ISAVE=1
LEVEL=0
ISV=1
SMLDIS=SMLDIS1-99999.
DO 400 I=1,NTGO
IF(SMLDIS-DIRDIS(I))350,350,310
310 ISAVE=1
SMLDIS=DIRDIS(I)
350 IF(TGTVEL(I))360,400,360
360 IF(SMLDIS-DIRDIS(I))400,400,380
380 SMLDIS=DIRDIS(I)
ISV=1
LEVEL=1
400 CONTINUE
```

determine the smallest range distance for all targets, and the smallest range distance for only the moving targets. The next statements:

```
TRAVM=SMLDIS/TGTSPD
ISVLSI=ISAVE
IF(TGTVEL(ISAVE))GT 0,RETURN
IF(LEVEL.LT.1) GO TO 580
```

compute the travel time to the next event for all targets. If the event target is stopped but at least one target is moving, the following statements:

```
TT=SMLDIS/TGTSPD
IF(TT.LE.DELATM(ISAVE))GO TO 570
TRAVM=DELATM(ISAVE)
RETURN
```

compute the travel time for the moving target that is closest to the event and if this travel time is greater than the delay that the event target is experiencing, the travel time to the

next event is set to the delay time of the event target.  
Otherwise:

```
570 ISAVE=ISV
    ISVLSI=ISV
    TRAVM=IT
    RETURN
```

the event target is set to the moving target that is closest  
to the event. The last statements in the subroutine:

```
580 DO 590 I=1,NTGO
    TM(I)=DELATM(I)
    IPOS(I)=I
    IF (TGTVM(I) GT 99990) TM(I)=99999
590 CONTINUE
    DO 700 J=1,NTGO
    IF (I.LT.2) GO TO 600
    IF (TM(I)-J).GT.TM(J) GO TO 710
600 K=I
    DO 700 J=1,NTGO
    IF (TM(I).LT.TM(J)) GO TO 700
    SAVE=TM(I)
    TM(I)=TM(J)
    TM(J)=SAVE
    KEEP=IPOS(I)
    IPOS(I)=IPOS(J)
    IPOS(J)=KEEP
700 CONTINUE
710 NUM=K-1
    DIS=99999
    DO 750 I=1,NUM
    KPOS=IPOS(I)
    IF (DIRDIS(KPOS) GT DIS) GO TO 750
    DIS=DIRDIS(KPOS)
    ISAVE=KPOS
750 CONTINUE
    ISVLSI=ISAVE
    TRAVM=TM(I)
    RETURN
    END
```

determine the event target by selecting the smallest delay  
time and the smallest event distance.

### Subroutine EVENTC

This subroutine evaluates an event involving a target and  
a travel path segment boundary. The first executable state-  
ments:

```
TGTVSV=TGTVEL (ISAVE)
ORYPRT=ORBY
```

save the target velocity and the boundary Y coordinate, as these variables may be changed. If the targets have not started moving, the event involves the entrance boundary, and the statements:

```

IF MUSH EQ 11 GO TO 505
OBY=99999
10B=11-1
MUSH=1
IF IKNRS.NE.11 GO TO 1000
TOTSIM=0
DO 200 I=1,NTGO
200 TIMEV(I)=0
GO TO 1000

```

remove the boundary from further consideration and initialize the total breach time and total target travel time if the first travel path segment has just been entered. If an exit boundary is involved, the statements:

```

505 CONTINUE
TGTYSV = TGTYNM(I)SAVE
LEFTIN = LEFTIN - 1
502 TGTYNM(I)SAVE=99991
IF (NDFA.LT.11 GO TO 511
TMTDFR(I)SAVE=9990
DO 510 J=1,NDFA
CALL STINT(I)SVIFA,ISAVE,J,IBIT,KDFTIN,2)
510 CONTINUE
511 IF (IKNRS.NE.NRS) GO TO 512
N=NTGTYP(I)SAVE
IF (INTTCS.NE.0) NSPLFT=NSPLFT-1
512 TGTVEL(I)SAVE=0
1000 RETURN
END

```

remove the target from further consideration. If the target is exiting the last travel path segment and has sweeping capability, the counter for the number of targets remaining with sweeping capability is decremented. If there are any targets in a direct fire area, the event target is removed from further consideration by direct fire by a call to Subroutine STINT.

#### Subroutine DIVSET

The purpose of this subroutine is to control the direction of target diversion. When a target which must be diverted around is damaged, Subroutine DIVSET is called. The first executable statements:

```

K=K1
IF (XFIX) 50,10,100
10 NDVT=NDVT+1
YDIV(NDVT)=TGTYNW(K)
NTCOL(NDVT)=NCTAW(K)
S(NDVT)=-1
GO TO 500

```

save the Y coordinate and column identification of the target if the target is not in the process of diverting around another damaged target. A flag is set so that future diversions at that location will be to the left. If the target which is damaged in this event is diverting to the left when damaged, the statements:

```

50 S(1D)=1
GO TO 500

```

change the diversion direction for subsequent targets in the column. If the target is diverting to the right when damaged, a target path blockage is formed, and the statements:

```

100 S(1D)=999
DO 200 I=1,NTGO
IF (TGTYNW(I) .GT. 99990) GO TO 200
IF (NTCOL(I) .NE. NCTAW(I)) GO TO 200
IF (TGTYNW(I) .GT. YDIV(I)-60) GO TO 200
TGTDEL(I)=TGTDEL(I)+D3DEL
DELATM(I)=D3DEL
TGTVEL(I)=0.
200 CONTINUE
500 RETURN
END

```

remove the three damaged targets from the travel path and assess a time delay to all targets behind the blockage in the same column.

#### Subroutine DIVCHK

This subroutine determines whether a target is diverting around another target and, if so, computes the offset distance due to diversion. A DO LOOP considers each target which must be diverted around. The first executable statements:

```

K=K1
DO 50 I=1,NDVT
10=1
IF (NCTAW(K) .NE. NTCOL(I)) GO TO 50

```

```

T1=GTYNWIK)
T2=YDIVI(1)*60.
T3=YDIVI(1)*60.
IF(T1.LT.T3.OR.T1.GT.T2) GO TO 50
IF(S(1) EQ.999.) GO TO 50

```

determine whether the event target is in the same column and within 60 feet of the target which must be diverted around. If so, the statements:

```

20 XFIX=5(1)*150RT(5625)-(1)*YDIVI(1)*21-45.
GO TO 100
50 CONTINUE
100 RETURN
END

```

compute the target offset due to diversion around the damaged target [Equation (13)].

#### Subroutine UNIT

The purpose of this subroutine is to determine if the lead target of a group has been damaged and, if so, to move as a unit the targets associated with the lead target. A DO LOOP is entered to consider each target that is damaged in the event. The first executable statements:

```

00 200 K=1,N100
IF(TGTYNWIK).LT.100000 ) GO TO 200
L=0
DO 10 I=1,N100
IF(TGTYNWIK).GT.99990.) GO TO 10
IF(NCTAWIK).NE.K) GO TO 10
L=L+1
NASIL=I
10 CONTINUE
IF (L.EQ.0) GO TO 200

```

determine whether any targets are associated with the damaged target as a group. If so, the next statements:

```

KF=KB=0
SMALY=99999.
810Y=99999.
DO 50 I=1,N100
IF(I EQ.K) GO TO 50
IF(NCTAW(I).NE.NCTAW(K)) GO TO 50
IF(NTGTYP(I).NE.NTGTYP(K)) GO TO 50
IF(TGTYNW(I).GE.99999.) GO TO 50

```

determine whether a target of the same type as the damaged target is in the same column. For targets which meet this qualification, the statements:

```
IF(TGTYNM(I).GT.TGTYNM(K)-100000.) GO TO 20
IF(TGTYNM(I).LT.BIGY) GO TO 50
KB=I
BIGY=TGTYNM(I)
GO TO 50
20 IF(TGTYNM(I).GT.SMALY) GO TO 50
KF=I
SMALY=TGTYNM(I)
50 CONTINUE
```

determine the closest ones in front of and behind the damaged target. If a target of the same type is ahead of the damaged target, the statements:

```
IF(KF.EQ.0) GO TO 100
DO 60 I=1,L
NI=NAS(I)
60 NGTAM(NI)=KF
GO TO 200
```

reassociate the remaining targets with the new lead target. If no such target is in front of the damaged target but one is behind, the statements:

```
100 IF(KB.EQ.0) GO TO 130
YFIX=TGTYNM(K)-BIGY-100000
DO 110 I=1,L
NI=NAS(I)
TGTYNM(NI)=TGTYNM(NI)+YFIX
110 NGTAM(NI)=KB
GO TO 200
```

move the remaining targets to reassociate them with the new lead target. If no targets which could be lead targets are in the column, the statements:

```
130 DO 150 I=1,L
NI=NAS(I)
TGTYNM(NI)=99999.
INTOFR(NI)=9990.
IF(I.RUNS.EQ.I.PRINT)PRINT 99999.NI
99999 FORMAT(' TGT LOST ',I4)
TGTVEL(NI)=0.
```

```

LEFTIN=LEFTIN-1
NTLOST=NTLOST+1
IF(NDFA.LT.1) GO TO 150
DO 140 J=1,NDFA
CALL STINT((SVIFA.NI.J,IBIT,KDFTN,2)
140 CONTINUE
150 CONTINUE
200 CONTINUE
RETURN
END

```

remove the remaining targets from the simulation and from all direct fire areas and define them as "lost" to the mission. (Such targets are not included in the count of targets damaged.)

#### Subroutine TGTMIN

The purpose of this subroutine is to evaluate the interactions between a target and a mine. The first executable statements:

```

TGTYSV=TGTYNH((SAVE)
KT=KTSV((SAVE)
IPD=1H
OBYPRT=OBY
XFIX=0
KI=(SAVE
NOBEVT=NOBEVT
IF(NDVT.GT.0) CALL DIVCHK

```

initialize several variables. If any targets which must be diverted around have been damaged, Subroutine DIVCHK is called to determine whether the event target must divert. The next statements:

```

XFIXI=XFIX
DISTPD=ABS(ODX-TGTKOL((SAVE)-XFIX)

```

compute the lateral distance between the event target and the event mine [Equations (12) and (14)]. If Tactic 1 is being evaluated or if the event target is not capable of sweeping mines, the statements:

```

N=NTGTYP(IISAVE)
M=MOD(INOBT2,8)
IF(NTTTRP(N).GT.0) GO TO 649
IF(INOBT2.GT.15) RETURN
IF(MODE.LT.3) GO TO 200
IF(NTTTC5(N).LT.1) GO TO 200

```

provide a branch to the section of the subroutine that determines whether the mine detonates. Similar branches are provided by the statement:

```

IF(DISTPD.GE.PRSWOO(N,M)) GO TO 200

```

if the target/mine distance is greater than the maximum range at which the target can sweep the mine. The statements:

```

NTABLE=(N-1)*2+(M-1)*3+1
CALL TABINT(DISTPD,NTABLE)

```

determine which mine detection table is to be interpolated, and Subroutine TABINT performs a standard linear interpolation. A uniform random number is compared to the probability of mine detection by the statements:

```

NOBEVT=IABS(NOBEVT)
RN=RNRF(DUMMY)
IF(RN.GT.PRBITY) GO TO 200

```

and if the mine is detected, the statements:

```

IPO=5HSWPT
OBY=99987.
NHSWPT(M)=NHSWPT(M)+1
CALL BOOM(IDEI,1)
JOB(KT)=-1
IF(NOBT2.LT.8) NMDET(M)=NMDET(M)+1
IF(ISBL.GT.0.AND.ISYMP(1).GT.0) CALL SYMDET

```

remove the mine from further consideration. Subroutine BOOM turns on the detonation flag bit for the mine; if swept mines are blown-in-place, Subroutine SYMDET evaluates sympathetic detonations (if required). The statements:

```

DO 105 I=1,NTGO
IF(TGTYNM(I).GT.99969) GO TO 105
IF(NCTAM(I).NE.NCTAM(IISAVE)) GO TO 105
SWPDEL(I)=SWPDEL(I)+TMSWP(M)

```

```

DELATM(1)=TMSWP(M)
TGTVEL(1)=0.
105 CONTINUE
GO TO 700

```

increment the sweep delay time and sets the velocity to zero for all active targets in the column. If the mine is not detected and if the mine employs a fuze timing cycle, the statements:

```

200 IF (INTIME(M) LT 1) GO TO 649
IF (KOUNT(M) GT 0) GO TO 649
TOTSIM=TIMEMV(IISAVE)+TGTDEL(IISAVE)+SWPDEL(IISAVE)
IF (AMOD(TOTSIM*60+IT.XCYCLE(M)) GT SECON(M)) RETURN
649 CONTINUE

```

determine if the fuze is active [Equations (16) and (17)]. If the fuze is inactive, no further evaluation is required. If the fuze is active or if the mine has no timing cycle, the next step is to determine if the mine detonates. The statements:

```

PRBITY=0
IF (NOBTP2 GT 7) RETURN
IF (DISTPD GE PRDIND(N,NOBTP2)) RETURN
NTABLE=(N-1)*2+(NOBTP2-1)*3+3
CALL TABINT(DISTPD,NTABLE)

```

call Subroutine TABINT to perform a linear interpolation of the probability of mine detonation function if the mine is not a dud and if the target/mine distance is less than the maximum range at which the mine can be detonated. A uniform random number is compared to the probability of mine detonation by the statements:

```

NOBEVT=1ABS(NOBEVT)
RN=RANF(DUMMY)
IF (RN LT PRBITY) GO TO 650
IF (INTTRP(N) LT 2) RETURN

```

The evaluation ends if the mine does not detonate and the sweeping device is not a plow. If the random number is greater than the probability of mine detonation, the statements:

```

IF (RANF(0UMHY).LT.PPEAF(M)) RETURN
NMSWPT(M)=NMSWPT(M)+1
IOB(KT)=I
IPO=5HSWEPT
CALL BOOM(IDEI,I)
RETURN

```

determine if the plowed mine can still function by comparing the probability value to a uniform random number. If the mine was swept, the mine is removed from further consideration. Subroutine BOOM is called to turn on the detonation flag for the mine. The next group of statements:

```

550 IF (KOUNT(M).LE.0) GO TO 650
IF (IT.LE.I) GO TO 650
IF IT=I
CALL IPACK
IOB(KT)=IWORD
RETURN

```

determine whether the mine has counted the required number of targets if the target counting feature is being evaluated. If the required target count has not been achieved, the variable is decremented, the mine information is packed again, and the evaluation is completed. If the required target count has been achieved or if the mine does not arm itself after counting a predetermined number of targets, the mine is detonated. The statements:

```

658 NMDET(M)=NMDET(M)+1
OBY = 99972
IF (ISYP.EQ.0) ISYP=1
CALL BOOM(IDEI,I)
IPO=9MDETONATED
IOB(KT)=I
IF (INTTRP(N)) 710,659
659 CONTINUE

```

remove the mine from further consideration. Subroutine BOOM is called to turn on the detonation flag for the mine; if this target type does not employ a roller or a plow as a sweeping device, a DO LOOP is entered to determine if any targets are damaged by the detonated mine. The statements:

```

DO 690 K=1,NTGO
IF (ITGYNM(K).GT.99969) GO TO 690
KI=K
IF (K.NE.ISAVE) GO TO 660
XFIX=XFIX+1
GO TO 670
660 XFIX=0
IF (INDVT.GT.0) CALL DIVCHK

```

consider an active target; if any targets which must be diverted around have been damaged, Subroutine DIVCHK is called to determine whether the target being considered must divert. The statement:

```
570 DISTPD=(OBK-TGTXOLIK)-XFIX)**2+(OBYPR-TGTYNWIK))**2
```

computes the target/mine distance [Equation (18)]. If the target/mine distance is less than the maximum range at which the target can be damaged, the statements:

```
N=NTGTYPK)
PK=PRKLDIN.NOBTP2)**2
IF(DISTPD.GT.PK) GO TO 690
DISTPD=SQRT(DISTPD)
NTABLE=(N-1)*21+(NOBTP2-1)*3*2
CALL TABINT(DISTPD,NTABLE)
```

call Subroutine TABINT to perform a linear interpolation of the probability of target damage function. Next, a uniform random number is compared to the interpolated probability value. If the target is damaged, the statements:

```
LEFTIN = LEFTIN - 1
NKILLI = NKILLI + 1
NKILL(N) = NKILL(N) + 1
IF(NDFA.LT.1) GO TO 690
DO 680 J=1,NDFA
CALL STINT(SVIFA,K,J,IBIT,KDFTN,2)
580 CONTINUE
590 CONTINUE
RN=RANF(DUMMY)
IF(RN.GT.PRBIT) GO TO 690
K1=K
IF(NTITMD(N).GT.0) CALL DIVSET
TGTYNWIK1=TGTYNWIK+100000.
TNTOFRIK)=9990.
TGTVSV=TGTVELIK)
TGTVELIK)=0
IF(1SY.PEQ.1) 1SY=2
IF(NTITCSIN).GT.0) NSPLFT=NSPLFT + 1
```

remove the target from further consideration, set the target velocity to zero, remove the target from all direct fire areas, and increment several counters. If the target must be diverted around by subsequent targets in the column, Subroutine DIVSET is called to save the target Y coordinate and set the diversion direction. The statements:

```

IF (ISYMP1) .GT. 0: CALL SYMDET
700 RETURN

```

provide a call to Subroutine SYMDET if sympathetic detonations for mines are being evaluated. The statements:

```

710 K1=ISAVE
NMSWPT(M1)-NMSWPT(M1)+1
PK=PRKLDIN,NOBTP2)
IF (DISTPD .GT. PK) RETURN
NTABLE=(N-1)*2+(NOBTP2-1)*3+2
CALL TABINT(DISTPD,NTABLE)

```

evaluate a roller or plow interaction with a mine. If the target/mine distance is greater than the maximum range at which the target can be damaged, the evaluation ends. Otherwise, Subroutine TABINT is called to perform a linear interpolation of the probability of damage function. Next, a uniform random number is compared to the interpolated probability value. If the target is damaged, the statements:

```

RN=RANF(DUMMY)
IF (RN .GT. PROBITY) RETURN
TGTYNW(ISAVE)=TGTYNW(ISAVE)+100000.
TMTDFR(ISAVE)=9990.
TGTVSV=TGTVEL(ISAVE)
TGTVEL(ISAVE)=0.
NSPLFT=NSPLFT-1
LEFTIN=LEFTIN-1
NKILLT=NKILLT+1
NKILL(N)=NKILL(N)+1
IF (NDFA .LT. 1) RETURN
DO 720 J=1,NDFA
CALL STINT(ISVIFA,ISAVE,J,IBIT,KDFTN,2)
720 CONTINUE
RETURN
END

```

remove the target from further consideration, set the target velocity to zero, remove the target from all direct fire areas, and increment several counters.

### Subroutine SYMDET

The purpose of this subroutine is to evaluate sympathetic detonations. Sympathetic detonations occur when a mine is sufficiently close to a detonating mine or an exploding direct or indirect fire round to itself detonate due to the disturbance. The first statements:

```

DIMENSION XSYM(2,100),YSYM(2,100),NOBT(100)
*NOB=NOB-2
I1=1
I2=2
IF (NOBTP2-16) 300,310,320
300 KTS=KTSAV(IISAVE)
MINNUM=I
ISUB=I
KNOBT=NOBT(I1)=NOBTP2
SOBX=XSYM(I,1)=OBX
SOBY=YSYM(I,1)=OBYPR
GO TO 10
310 KTS=KTSAV(IISAVE)
ISUB=3
J=NHEPV(IMIN)
DO 315 K=1,J
XSYM(I1,K)=ROTDMPX(IMINE,K)
YSYM(I1,K)=ROTDMPY(IMINE,K)
315 NOBT(K)=16
MINNUM=J
GO TO 10
320 KTS=KTSAV(IIS)
ISUB=2
MINNUM=I
NOBT(I1)=17
XSYM(I1,1)=TGT XOL(IIS)
YSYM(I1,1)=TGT YNW(IIS)

```

provide local storage for the event X and Y coordinates and the mine or weapon type and initialize several variables prior to the evaluation. The X and Y coordinates of the impacting round or of the mine which detonated in the event are placed into the XSYM(2,100) and YSYM(2,100) arrays. This mine or round is evaluated against other mines; if any sympathetic detonations occur, the X and Y coordinates of the detonated mines are also saved for evaluation against other mines. The statements:

```

10 NLST=MINNUM
MINNUM=0
YMAX=0
YMIN=99999
DO 20 I=1,NLST
YMAX=AMAXI(YMAX,YSYM(I1,I))
20 YMIN=AMINI(YMIN,YSYM(I1,I))

```

determine the maximum and minimum Y coordinates of the mines or rounds. Next, all mines within the maximum sympathetic detonation range of the detonated mines are identified (comparing Y coordinate only). The statements:

```

KMIN=KIMAX-KTSI-KTS
30 KIS=KTSI
IF (KTSI.LT.2) GO TO 40
IF (IOBKTSI).EQ.-1) GO TO 30
IWORD=IOBKTSI
CALL UNPACK
IF (NOBTP2.GT.8) GO TO 30
IF (YMIN-SYMMAX(IISUB).GT.OBY) GO TO 40
KIMIN=KIS
GO TO 30

```

perform this function for mines with smaller Y coordinates than the value of YMIN, and the statements:

```
40 KTSI=KTS
50 KTSI=KTSI+1
   IF(KTSI.GT.KNOB) GO TO 60
   IF(10B(KTSI).EQ.-1) GO TO 50
   IWORD=10B(KTSI)
   CALL UNPACK
   IF(NOBTP2.GT.8) GO TO 50
   IF(YMAX+SYMMAX(I SUB).LT.OBY) GO TO 60
   KIMAX=KTSI
   GO TO 50
```

perform this function for mines with greater Y coordinates than the value of YMAX. Next, a DO LOOP is entered to determine which, if any, of the mines detonate sympathetically. The statements:

```
60 DO 120 I=KIMIN,KIMAX
   IF(10B(I).EQ.-1) GO TO 120
   IWORD=10B(I)
   CALL UNPACK
```

unpack the event information if the event has not previously occurred. Next, the mine is evaluated against the mines which have detonated in this event. The statements:

```
N=NOBTP2
IF(NOBTP2.GT.8) GO TO 120
DO 110 J=1,NLST
DIS=10BX-XSYM(I,J)**2+(OBY-YSYM(I,J))**2
```

compute the square of the distance from the detonated mine to the mine being considered [Equation (19)], and the statements:

```
IF(NOBT(J)-16)390,400,410
390 K=MOD(NOBT(J),8)
   IF(DIS.GT.SYMDIS(K,N)) GO TO 110
   GO TO 65
400 K=INTVAP(MINE)
   IF(DIS.GT.SYMDIF(K,N)) GO TO 110
   GO TO 65
410 K=INTDEF(NDF)
   IF(DIS.GT.SYMOOF(K,N)) GO TO 110
```

compare this value to the square of the maximum range at which sympathetic detonations can occur for the mines involved with a detonating mine or round. If the mine detonates, the statements:

```

65 MINNUM=MINNUM+1
   NMDET(NI)=NMDET(NI)+1
   IF MINNUM.LE 1000 GO TO 70
   WRITE(6,1000)
1000 FORMAT(1H0,'MORE THAN 100 SYMPATHETIC DETONATIONS')
   CALL EXIT
   70 XSYN(12,MINNUM)=OBX
   YSYN(12,MINNUM)=OBY
   NOBT(MINNUM)=NOBTP2
   CALL BOOM(IDET,1)
   IOBTT=-1

```

remove the mine from further consideration, save the mine coordinates, and call Subroutine BOOM (which turns on the mine detonation flag bit). If more than 100 mines sympathetically detonate, an error message is printed, and the program stops. If the optional output is required, the statements:

```

IF (PRINT NE (RUNS) GO TO 120
IF (MINNUM EQ 1) WRITE(6,1010)
1010 FORMAT(' SYMPATHETIC DETONATIONS  OB-NUM  OBS-TYPE  OB X  OB Y')
WRITE(6,1020) 1,NOBTP2,OBX,OBY
1020 FORMAT(26X,15,110,15,110,15,110,15,110)
GO TO 120
110 CONTINUE
120 CONTINUE

```

print the mine sequential number, the mine type, and the mine X and Y coordinates for the sympathetically detonated mine. If any mines were detonated in this analysis, the statements:

```

IF (MINNUM EQ 0) GO TO 200
13=11
11=12
12=13
15=0
GO TO 10

```

provide for further evaluation of sympathetic detonations. Otherwise, the statements:

```

200 NOBTP2=KNOBT
   OBX=SOBX
   OBYPRI=SOBY
   RETURN
   END

```

reset several variables which may have changed in the subroutine, and the evaluation is completed.

### Subroutine PRINTO

The purpose of this subroutine is to print the optional output at the end of the event (if required). The first executable statements:

```
TOTSIM=TIMENV(I SAVE)+TGTDEL(I SAVE)+SMPDEL(I SAVE)
IF(NOBTP2.EQ.17) GO TO 1520
IF(NOBTP2.EQ.18) GO TO 1530
IF(NOBTP2.EQ.16) RETURN
IF(NOBTP2=8)10,1500
```

compute the total breach time for the event target and if the event was a travel path boundary or direct fire area boundary, branches are provided to other sections of the subroutine. The next statements:

```
10 DO 200 K1=1,11
   K6=K1*10
   K5=K6-9
   IF(K6 GT. NTGO) K6=NTGO
   PRINT 1001,(1,1=K5,K6)
   PRINT 1002,(KTSAV(1),1=K5,K6)
   PRINT 1003,(DIRDIS(1),1=K5,K6)
   IF(K6.EQ. NTGO) GO TO 40
200 CONTINUE
```

print the sequential number of the next event to be encountered by each target and the distance to the event. The statements:

```
40 TYPRI = TOTYNM(I SAVE)
   TVELPR=TGYVEL(I SAVE)
   IF(TGTYNM(I SAVE) LT. 99970.) GO TO 1004
   TYPRI = TGTYSV
   TVELPR=TGTYSV
```

obtain the Y coordinate and velocity of the event target. The statements:

```
1004 WRITE(6,1400) I SAVE,NTGTYP(I SAVE),TYPRI,TOTKOL(I SAVE),
   TVELPR,TIMENV(I SAVE),TOTSIM
   WRITE(6,1410)KTSAV(I SAVE),NOBTP2,OBX,OBYPRI,PROBITY,RN,1PO
```

print the sequential target number of the event target, the target type, the coordinates and velocity of the event target, the total time that the event target has been moving, the total breach time, the sequential number of the event, the

event type and coordinates of the event, and the last interpolated probability and random number with a variable indicating that a mine was either swept or detonated in the event. If optional output has been printed for 200 events, the statements:

```
WRITE(6,1420)
NOBEVT=NOBEVT+1
IF(NOBEVT.GT.200) IPRINT=0
```

turn off the optional output request variable to limit excessive output. Finally, the statements:

```

N2=NTGO/2
DO 1019 K=1,N2
M=K+N2
TOT2=DELATM(K)
TOTDEL=DELATM(K)
1019 WRITE(6,1120) K,NTGTYP(K),TGTYNW(K),TGTXL(K),TGTVEL(K),TIMERV(K),
  *TOTDEL,M,NTGTYP(M),TGTYNW(M),TGTXL(M),TGTVEL(M),TIMERV(M),
  *TOT2
  IF(M-NTGO)1440,1012
1440 M=NTGO
  TOTDEL=DELATM(M)
  WRITE(6,1120)(M,NTGTYP(M),TGTYNW(M),TGTXL(M),TGTVEL(M),TIMERV(M),
  *TOTDEL
1012 RETURN
1500 WRITE(6,1510) ISAVE,OBYPRT,TOTSIM
1510 FORMAT(1X,9(1H*),* TRAVEL PATH BOUNDARY EVENT TGT *
  @14,* OB Y*,FB,1,* BREACH TIME*,FB,3,9(1H*)
  RETURN
1520 IPO=8HENTRANCE
  GO TO 1540
1530 IPO=4HEXIT
1540 WRITE(6,1545) IPO,IT,OBYPRT,ISAVE
1545 FORMAT(1X,9(1H*),* DIRECT FIRE AREA *,A10,
  @* NUMBER *,13,* OB Y*,FB,1,* EVENT TGT*,15,9(1H*)
  RETURN
```

print, for each target, the sequential target number, the target type, the target coordinates and velocity, the total time the target has been moving, and the delay time experienced by the target. The travel path segment boundary and direct fire area boundary information is also printed.

### Subroutine PRINTR

The purpose of this subroutine is to print the results for each iteration and a periodic statistical summary. The formats for the output are determined at execution time and are a function of the various options being evaluated. The statements:

initialize an array used to store the variables used in the distribution output. For the first call to Subroutine PRINTR, the statements:

```
IF(JMARK1.LT.1) GO TO 80
DO 70 J=1,52
70 XRG(J)=0.
REWIND 7
JMARK1=0
```

initialize an array that creates the variables used in the distribution output and rewinds an intermediate disk file on which the distribution data is stored. If iteration output is desired and the results of the first iteration is being printed or if the statistical summary was printed for the previous iteration, the statements:

```
IF(ITEROP.LT.1) GO TO 100
NTMAX=MAX0(NMT,NTGTP,NDFMT)
IF(MOD(IRUNS,NOSTAT).NE.1) GO TO 100
WRITE(6,5000) IEND,MODE,(NITBT(I),I=1,NTGTP)
IF(NDFMT.GT.0) WRITE(6,5001)((NIDBT(I),I=1,NDFMT)
IF(NMT.GT.0) WRITE(6,5002)((NMIN(I),I=1,NMT)
IF(ITEROP.LT.1) GO TO 100
WRITE(6,5003)
WRITE(6,5004)
```

print page heading and column heading information. The page heading includes the run number, the tactic option being employed, the number of intruders and defenders by type, and the number of mines by type. Next, the statements:

```
100 BIG=0.
DO 110 I=1,NTGO
SUM=TIMENV(I)+TGTDEL(I)+SMPDEL(I)
IF(SUM.LT.BIG) GO TO 110
BIG=SUM
TGTD=TGTDEL(I)
SMPD=SMPDEL(I)
TMNV=TIMERV(I)
110 CONTINUE
TOTSIM=BIG
```

compute the total breach time. If iteration output is desired, a DO LOOP is entered to create the output lines by encoding the data into the OUT(16) array. ENCODE(N,M,HOL) is a Control Data Corporation system function which encodes N characters from Format M into the variable HOL.

```

IF(ITEROP.LT.1) GO TO 310
WRITE(6,2010)
120 DO 300 I=1,NIMAX
    DO 150 J=1,16
150   OUT(J)=1H
      IF(I.LT.2) ENCODE(5,1001,OUT(1)) IRUNS
      ENCODE(5,1001,OUT(2)) I
      IF(NTGTP.GE.1.AND.NMT.GT.0) ENCODE(6,1002,OUT(3)) NKILL(1)
      IF(NTGTP.GE.1.AND.NDFWT.GT.0) ENCODE(7,1004,OUT(4)) NKILLD(1)
      IF(NTGTP.GE.1.AND.NVAP.GT.0) ENCODE(7,1004,OUT(5)) NKILLI(1)
      IF(NDFWT.GE.1.AND.IRTFIR.GT.0) ENCODE(9,1006,OUT(6)) NDFDAM(1)
      IF(NTGTP.GE.1.AND.IRTFIR.GT.0) ENCODE(10,1007,OUT(7)) NRFBIT(1)
      IF(NDFWT.GE.1) ENCODE(9,1008,OUT(8)) NRFBDT(1)
      IF(I.GT.NMT) GO TO 180
      ENCODE(10,1007,OUT(9)) NMDET(1)
      ENCODE(10,1007,OUT(10)) MIFBT(1)
      ENCODE(9,1008,OUT(11)) NMSWPT(1)
100   IF(I.GT.1) GO TO 200
      IF(NGTAWT.GT.0) ENCODE(6,1002,OUT(12)) NTLOST
      IF(MADIV.GT.0) ENCODE(8,1013,OUT(13)) TGTD
      IF(MODE.GT.1) ENCODE(7,1014,OUT(14)) SWPD
      ENCODE(8,1013,OUT(15)) TMMV
      ENCODE(8,1013,OUT(16)) TOTSIM
200   N=11
      IF(I.LT.2) N=16
      WRITE(6,2000) (OUT(K),K=1,N)
300   CONTINUE

```

The next statements:

```

310   DO 320 I=1,NTGTP
      DIST(1)=NKILL(1)
      DIST(1+5)=NKILLD(1)
      DIST(1+10)=NKILLI(1)
320   DIST(1+20)=NRFBIT(1)
      DO 330 I=1,NDFWT
      DIST(1+15)=NDFDAM(1)
330   DIST(1+25)=NRFBDT(1)
      DO 340 I=1,NMT
      DIST(1+30)=NMDET(1)
      DIST(1+37)=MIFBT(1)
340   DIST(1+44)=NMSWPT(1)
      DIST(52)=TOTSIM

```

store the data used in the distribution output into the DIST(52) array. If distribution data is desired, the statements:

```

IF(ITEROP.LT.1) GO TO 360
DO 350 I=1,52
350   XRG(I)=AMAXI(XRG(I),DIST(I))
      WRITE(7) DIST

```

determine the maximum value of each distribution output variable and store the distribution data on Tape 7. In the next statements:

```

350   DO 370 I=1,52
      STATAL(1,I)=STATAL(1,I)+DIST(I)
370   STATAL(2,I)=STATAL(2,I)+DIST(I)**2

```

the variables to be included in the statistical summary are saved. The sums and the sums of the squares for each of the variables are also computed. If statistical results are to be printed for this iteration, the statements:

```

IF (MOD(IRUNS,NOSTAT).NE.0) RETURN
RUNS=IRUNS
IRN=IRUNS
DO 380 I=1,52
STATAL(3,I)=STATAL(1,I)/RUNS
STATAL(4,I)=(STATAL(2,I)-(STATAL(1,I)**2/RUNS)/(RUNS-1.0)
IF (STATAL(4,I).GT.0.1 STATAL(5,I)=SQRT(STATAL(4,I))
380 CONTINUE
WRITE(6,6000)
IF (NMT.GT.0) WRITE(6,6010)((STATAL(I,J),I=3,5),J=1,NTGTP)
K=NTGTP+5
IF (NDFWT.GT.0) WRITE(6,6020)((STATAL(I,J),I=3,5),J=6,K)
K=NTGTP+10
IF (INVAP.GT.0) WRITE(6,6030)((STATAL(I,J),I=3,5),J=11,K)
K=NDFWT+15
IF (NDFWT.GT.0) WRITE(6,6040)((STATAL(I,J),I=3,5),J=16,K)
K=NTGTP+20
IF (NDFWT.GT.0) WRITE(6,6050)
IF (IRTFIR.GT.0) WRITE(6,6060)((STATAL(I,J),I=3,5),J=21,K)
K=NDFWT+25
IF (NDFWT.GT.0) WRITE(6,6070)((STATAL(I,J),I=3,5),J=26,K)
K=NMT+30
IF (NMT.LT.1) GO TO 400
WRITE(6,6080)((STATAL(I,J),I=3,5),J=31,K)
K=NMT+37
WRITE(6,6090)((STATAL(I,J),I=3,5),J=38,K)
K=NMT+44
400 IF (MODE.EQ.3) WRITE(6,6100)((STATAL(I,J),I=3,5),J=45,K)
WRITE(6,6110)(STATAL(I,52),I=3,5)

```

compute and print the means, variances, and standard deviations for each variable. If distribution output is desired, the last statements in the subroutine:

```

IF (DISOP.GT.0) CALL DISTR(NMT,NTGTP,NDFWT)
RETURN

```

provide a call to Subroutine DISTR to print the distribution data.

#### Subroutine DISTR

The purpose of this subroutine is to print the distribution data. The formats for the output are determined at execution time and are a function of the various options being evaluated. The statements:

```

DATA IFMT/10H1X,8HINTE,5HRVAL,,3HA7,,3HA7,,3HA7,,3HA7,,
*3HA7,,3H3X,,3HA7,,3HA7,,3HA7,,3HA7,,3HA7,,3H3X,,
*3HA7,,3HA7,,3HA7,,3HA7,,3HA7,,3H3X,1H)6*(1H) /
DATA JFMT/10H12X,5HCLAS,10HS/(3X12,6X,3HR7,,3HR7,,3HR7,,
*3HR7,,3HR7,,3H3X,,3HR7,,3HR7,,3HR7,,3HR7,,3HR7,,3H3X,,
*3HR7,,3HR7,,3HR7,,3HR7,,3HR7,,3H3X)1H)6*(1H) /
DATA NFMT/10H1X,8HINTE,5HRVAL,,3HA7,,3HA7,,3HA7,,3HA7,,
*3HA7,,3HA7,,3HA7,,3H3X,,3HA7,,3HA7,,3HA7,,3HA7,,
*3HA7,,3HA7,,3H3X,1H)8*(1H) /
DATA KFMT/10H12X,5HCLAS,10HS/(3X12,6X,3HR7,,3HR7,,3HR7,,
*3HR7,,3HR7,,3HR7,,3HR7,,3H3X,,3HR7,,3HR7,,3HR7,,
*3HR7,,3HR7,,3HR7,,3H3X)1H)8*(1H) /

```

provide Hollerith data from which the output formats can be created. The statements:

```

XRN=1 0/FLOAT(1RN)
DO 20 I=1,52
INT(I)=FIX(XRG(I))/20+1
IF (INT(I).LT.1) GO TO 10
L=ALOG10(FLOAT(INT(I)))
LI=10*L
INT(I)=(INT(I)/LI+1)*LI
10 DO 20 K=1,21
20 DIST5(K,I)=0.0
REWIND 7

```

computed the interval over which the distributions are defined. The statements:

```

DO 50 I=1,1RN
READ(7)DIST
DO 50 J=1,52
KI=IDIST(J)-.01/FLOAT(INT(J))+2.0
IF (DIST(J).EQ.0.0)KI=1
50 DIST5(KI,J)=DIST5(KI,J)+XRN

```

compute the distribution data for each variable to be output. The next statements:

```

KZ=0
WRITE(6,1000)
WRITE(6,1001)
WRITE(6,1007)
DO 85 I=1,27
LFMT(I)=IFMT(I)
MFMT(I)=JFMT(I)
85 CONTINUE
DO 90 I=1,15
100T(I)=1H
DO 90 J=1,21
OUT(J,I)=1H
90 CONTINUE

```

print the first page headings and initialize the output arrays. The next statements:

```

      IF (NMT.LT.1) GO TO 110
      DO 100 I=1,NTGTP
      IOUT(I)=INT(I)
      LFMT(I*2)=3H17.
      MFMT(I*2)=5HF7.3.
94   DO 95 J=1,21
      OUT(J,1)=DIST5(J,1)
95   CONTINUE
100  CONTINUE

```

store the intervals and the distribution data for the intruders damaged by mines into the output arrays and create the output formats for the number of intruder target types. The next statements:

```

110  IF (NDFMT.LT.1) GO TO 180
      DO 150 I=1,NTGTP
      IB=I*8
      IOUT(I*5)=INT(I*5)
      LFMT(I*5)=3H17.
      MFMT(I*5)=5HF7.3.
144  DO 145 J=1,21
      OUT(J,1*5)=DIST5(J,1*5)
145  CONTINUE
150  CONTINUE

```

store the intervals and distribution data for the intruders damaged by direct fire into the output arrays and create the output formats for the number of intruder target types. The next statements:

```

180  IF (NDFMT.LT.1) GO TO 200
      DO 195 I=1,NTGTP
      I14=I*14
      IOUT(I*10)=INT(I*10)
      LFMT(I*10)=3H17.
      MFMT(I*10)=5HF7.3.
189  DO 190 J=1,21
      OUT(J,1*10)=DIST5(J,1*10)
190  CONTINUE
195  CONTINUE
200  WRITE(6,LFMT)IOUT
      WRITE(6,MFMT)(I,1OUT(I*1,J),J=1,15),1*#2,20)

```

store the intervals and distribution data for the intruders damaged by indirect fire and print the output data. The next statements:

```

      IF (NDFMT.LT.1) GO TO 500
      DO 210 I=1,15
      IOUT(I)=1H
      DO 210 J=1,21
      OUT(J,1)=1H
210  CONTINUE
      DO 215 I=1,27
      LFMT(I)=1FMT(I)
      MFMT(I)=1MFMT(I)
215  CONTINUE
      WRITE(6,1002)

```

```

WRITE(6,1007)
DO 250 I=1,NDFWT
  IOUT(I)=INT(I*.15)
  LFMT(I+2)=3HF7.3.
  MFMT(I+2)=5HF7.3.
  DO 240 J=1.21
    OUT(J,I)=DIST5(J,I*.15)
240 CONTINUE
250 CONTINUE
DO 285 I=1,NTGTP
  IOUT(I+5)=INT(I*.20)
  LFMT(I+8)=3HF7.3.
  MFMT(I+8)=5HF7.3.
  DO 280 J=1.21
    OUT(J,I+5)=DIST5(J,I*.20)
280 CONTINUE
285 CONTINUE
DO 300 I=1,NDFWT
  IOUT(I+10)=INT(I*.25)
  LFMT(I+14)=3HF7.3.
  MFMT(I+14)=5HF7.3.
  DO 295 J=1.21
    OUT(J,I+10)=DIST5(J,I*.25)
295 CONTINUE
300 CONTINUE
WRITE(6,LFMT)IOUT
WRITE(6,MFMT)I,IOUT(I+1,J),J=1.15),I=KZ.20)

```

initialize the storage arrays, write new headings, store the intervals and distribution data, and create the output formats. The output is printed for defenders damaged, the rounds fired by the intruders, and the rounds fired by the defenders. The statements:

```

WRITE(6,1003)
500 WRITE(6,1000)
  IF(NMT.LT.1) GO TO 700
  DO 510 I=1.27
    LFMT(I)=NFMT(I)
510 MFMT(I)=KFMT(I)
  DO 520 I=1.15
    IOUT(I)=IH
  DO 520 J=1.21
    OUT(J,I)=IH
520 CONTINUE
  WRITE(6,1008)
  WRITE(6,1005)
  WRITE(6,1008)
  DO 550 I=1,NMT
    IOUT(I)=INT(I*.30)
    LFMT(I+2)=3HF7.3.
    MFMT(I+2)=5HF7.3.
    DO 540 J=1.21
      OUT(J,I)=DIST5(J,I*.30)
540 CONTINUE
550 CONTINUE
  DO 570 I=1,NMT
    IOUT(I+7)=INT(I*.37)
    LFMT(I+10)=2HF7.3.
    MFMT(I+10)=4HF7.3.
    DO 560 J=1.21
      OUT(J,I+7)=DIST5(J,I*.37)
560 CONTINUE
570 CONTINUE
  WRITE(6,LFMT)IOUT(I+1,J),J=1.14),I=KZ.20)
  WRITE(6,MFMT)I,IOUT(I+1,J),J=1.14),I=KZ.20)

```

initialize the storage arrays, write new headings, and provide for storage of distribution data into the output arrays while creating the output formats. The output is printed for mines detonated and mines in field. The statements:

```

WRITE(6,1003)
700 DO 710 I=1,27
    LFMT(1)=MFMT(1)
    MFMT(1)=KFMT(1)
710 CONTINUE
    LFMT(13)=3H17.
    MFMT(13)=5HF7.3.
    DO 720 I=1,15
        IOUT(1)=1H
        DO 720 J=1,21
            OUT(J,1)=1H
720 CONTINUE
    DO 740 I=1,NM1
        IOUT(1)=INT(I*44)
        LFMT(1+2I)=3H17.
        MFMT(1+2I)=5HF7.3.
        DO 770 J=1,21
            OUT(J,I)=DIST5(J,I*44)
770 CONTINUE
780 CONTINUE
        IOUT(10I)=INT(52)
        DO 790 J=1,21
            OUT(J,10I)=DIST5(J,52)
790 CONTINUE
    WRITE(6,1004)
    WRITE(6,1005)
    WRITE(6,1006)
    WRITE(6,LFMT)(IOUT(1),I=1,14)
    WRITE(6,MFMT)(I,OUT(I=1,J),J=1,14),I=KZ,20)
    RETURN
    END

```

provide for storage of the distribution data into the output arrays and create the output formats. The output is printed for mines swept and breach time.

#### Subroutine TABINT

The purpose of \*this subroutine is to perform a standard linear interpolation [Equation (15)] for any of the probability functions. The first executable statements:

```

10 NB=NTABLE*8-7
   NE=NB*7
   DO 50 K5=NB,NE
   IF (DISTPD.GE.RANGPR(K5) AND.DISTPD.LE. RANGPR(K5+1)) GO TO 55
50 CONTINUE
   GO TO 60

```

determine the table location in the function arrays and the two range values which bracket the distance to be interpolated. The statements:

```

55 PRBITY=PROBIL(K5) * (DISTPD-RANGPR(K5)) / (RANGPR(K5) - RANGPR(K5))
2 1 * (PROBIL(K5+1) - PROBIL(K5))
GO TO 70
60 PRBITY=C.
70 RETURN
END

```

perform the interpolation.

### Subroutine RNORM

The purpose of this subroutine is to select a random variable from a normal distribution [Equations (3) and (4)]. This is accomplished by first selecting a uniform random number and then evaluating a polynomial using the uniform random number. The statements:

```

RN19 = RANF(DUMMY)
Q = ABS(1 - 2.0 * RN19)
Q = .5 * (1.0 - Q)
Q = -2. * ALOG(Q)
V = SQRT(Q)
SIGN = 1
IF (RN19 - .5) 10, 10, 20
10 SIGN = -1.
20 Q = 2.515517 * .802853 * V + .010328 * V * V
GG = 1. + 1.432788 * V + .189259 * V * V + .0013208 * V * V * V
RSTART = (V - Q / GG) * SIGN
RETURN
END

```

perform this function, returning random variables with a mean of zero and a standard deviation of 1.

### Subroutine IPACK

The purpose of this subroutine is to pack the event information into the variable IWORD. The word contains the event X and Y coordinates, the starting point for the mine timing cycle or the required target count for arming (if applicable), the event type, and the sequential mine number. The first executable statements:

```

IX = OBY * 10.0
IY = OBY * 10.0
IWORD = 0

```

compute integer values of the mine or boundary coordinates with one decimal-digit accuracy and set the word to zero. The statements:

```

IWORD=IWORD.OR.IMINE
IWORD=IWORD.OR.SHIFT(IWORD,15)
IWORD=IWORD.OR.SHIFT(IWORD,20)
IWORD=IWORD.OR.SHIFT(ABS(X),20)
IF(X.LT.0) IWORD=IWORD.OR.1000000000000000
RETURN
END

```

pack the information as described in Table 2 of the Mathematical Model. The SHIFT function is a Control Data Corporation library function which shifts the bits right or left.

#### Subroutine UNPACK

The purpose of this subroutine is to unpack the event information which was packed by Subroutine IPACK; the process is simply the reverse of that used in Subroutine IPACK. The first executable statements:

```

OBY=FLOAT(SHIFT(IWORD,-3) AND.1777791710)
OBX=FLOAT(SHIFT(IWORD,-20) AND.377778710)
TEMP=IWORD.AND.1000000000000000
IF(TEMP.NE.0) OBY=OBX

```

unpack the event X and Y coordinates and set the sign of the X coordinate. The next statements:

```

NOBIT2=SHIFT(IWORD,-15) AND.378
IMINE=IWORD.AND.77778
10 IT=SHIFT(IWORD,-20) AND.3778
20 RETURN
END

```

unpack the event type and the event sequential number.

#### Subroutine NDFIRE

The purpose of this subroutine is to evaluate the effects of the indirect fire rounds employed against the intruding targets. The first executable statements:

```

BIGG(X)=.5*SQRTE(1.-EXPI(-.63*X**2))
PI=3.141592654
SRPI=SQRTE(PI)
IWT=INT(VAP(IVAP))
ICMHE=1
IF(INSUB(IWT).LT.1) ICMHE=2

```

```

COSIMP=COS(ANGIMP(IIVAP))* 017453292581
SINIMP=SIN(ANGIMP(IIVAP))* 017453292581
ANG=DIRATK(IIVAP)
COSANG=COS(ANG)
SINANG=SIN(ANG)
NH=NMEMPV(IIVAP)
IF (IRUNS.NE.IPRINT)GO TO 90
NVOL=NVFAEA(IIVAP)-NVLIDF(IIVAP)*I
WRITE(6,1000)IIVAP,NVOL,IWT
90 CONTINUE

```

compute several variables which are independent of the target and round and print the optional output, if required. The statements:

```

DO 500 I=1,NTGO
IF (IGTYNW(I) GT .99990) GO TO 500
XT=IGTXOL(I)
YT=IGTYNW(I)
ITT=NTGTYP(I)
SURVPR=I

```

consider each active target and determine the target X and Y coordinate and target type. The following statements:

```

DO 400 J=1,NH
EIVAL=E(I,IWT,ITT,2)
KODEI=E(I,IWT,ITT,1)
XW=ROTOPX(IIVAP,J)
YW=ROTOPY(IIVAP,J)

```

consider each round employed at the volley aimpoint, and determine the value of the effectiveness index, the code for the effectiveness index used, and the X and Y coordinates of the desired mean points of impact. The statements:

```

RO=ABS((XW-XT)*SINANG+(YW-YT)*COSANG)
DO=ABS(-(XW-XT)*COSANG+(YW-YT)*SINANG)
DSQ=RO**2+DO**2
IF (DSQ.GT. THRSIG(IWT,ITT)) GO TO 400

```

compute the distance in range and deflection from the center of the intruder target to the desired mean point of impact [Equations (42) and (43)] and if the round impacted near enough to the target to cause damage, the statements:

```

APHD=PHD(IWT,ITT)
IF (ICMHE GT 1) GO TO 110
POWER=-INSUB(IWT)*RELSUB(IWT)*EIVAL/(PI*PATRAD(IWT)**2)
IF (POWER GT -.227) GO TO 100
APHD=1

```

```
GO TO 105
100 APND=1 EXPPOWER
```

compute the probability of damage given the target is in the pattern [Equation (45)] for the improved conventional munition (ICM). The next statements:

```
105 ETL=ETM*SRPI*PATRAD(IWT)
110 GO TO (120,140,150,160),KODEI
```

compute the effective target length and width based on the radius of the pattern for the ICM weapon type [Equation (44)] and branch to the appropriate section of the subroutine depending on the code for the effectiveness index. For high explosive (HE) munitions described in terms of MAE<sub>f</sub> the following statements:

```
120 IF (ICMHE LT 2) GO TO 130
ALWRAT=1.-.8*COSIMP
ETL=2.*SQRT(EIVAL*ALWRAT/PI)
ETM=ETL/ALWRAT
```

compute the effective target length and width [Equations (47) and (48)] based on the ratio of the target length to the target width [Equation (46)]. The statements:

```
130 DENOM=17.6*REP(IIVAP)**2*ETL**2
RSSP=ETL/SQRT(DENOM)*EXP((-4.*RO**2)/DENOM)
DENOM=17.6*DEP(IIVAP)**2*ETM**2
DSSP=ETM/SQRT(DENOM)*EXP((-4.*DO**2)/DENOM)
135 PCK=RSSP*DSSP*RELNRND(IWT)*APHD
GO TO 450
```

compute the single shot probability of hit in range and deflection [Equations (49) and (50)] and the single shot probability of damage [Equations (51) and (67)]. The next statements:

```
140 ETM=SQRT(EIVAL)
ETL=ETM/SINIMP
GO TO 220
```

compute the effective target length and width based on the square root of the vulnerable area (VA<sub>N</sub>) and the sine of the

angle of fall at impact for this munition [Equations (53) and (54)]. The statements:

```
150 ETL=ETW*SQR1(EIVAL)
GO TO 220
```

compute the effective target length and width based on the square root of the mean area of effectiveness ( $MAE_b$ ) for blast [Equation (52)]. If the effectiveness index is in terms of effective miss distance (EMD), the statements:

```
160 IF(TARRAD(IIT))170,180
170 ETL=ETW*SQR1(TARRAD(IIT)*EIVAL)
GO TO 190
180 ETL=TARL(IIT)*2.*EIVAL
ETH=TARW(IIT)*2.*EIVAL
```

compute the effective target length and width [Equations (55), (56), and (57)]. If the height of the target is input, the statements:

```
190 IF(TARHT(IIT))200,220
200 SHADOL=TARHT(IIT)*SINIMP/COSIMP
```

compute the target shadow length [Equation (58)]. If the shadow length is greater than the effective miss distance, the statements:

```
IF(SHADOL.LT.EIVAL) GO TO 220
IF(TARRAD(IIT))205,210
205 ETL=(ETL**2+2.*TARRAD(IIT)*SHADOL-EIVAL)/ETL
GO TO 220
210 ETL=(ETL*ETH+TARW(IIT)*(SHADOL-EIVAL))/ETH
```

recompute the effective target length [Equations (59) and (60)]. The statements:

```
220 A1=(ETL+2.*RO)/(2.96*REP(IVAP))
A2=ABS(ETL-2.*RO)/(2.96*REP(IVAP))
B1=(ETW+2.*DO)/(2.96*DEP(IVAP))
B2=ABS(ETW-2.*DO)/(2.96*DEP(IVAP))
RSSP=BIGG(A1)*SIGN(1.,ETL-2.*RO)*BIGG(A2)
DSSP=BIGG(B1)*SIGN(1.,ETW-2.*DO)*BIGG(B2)
GO TO 135
```

compute the range single shot probability of hit and deflection single shot probability of hit [Equations (61) through (66)]. The probability of target survival for the volley is computed by the statement:

```
450 SURVPR=SURVPR*(1.-PKK)
490 CONTINUE
```

If the probability of target survival has been computed, the statements:

```
IF(SURVPR.EQ.1.) GO TO 500
PDAM=1.-SURVPR
RN=RANF(DUMMY)
```

compute the probability of target damage [Equation (68)] and a uniform random number is obtained. The random number is compared to the probability of target damage [Equation (69)] and if the target is damaged, the statements:

```
IF(RN.GT.PDAM) GO TO 490
IF(IRUNS.NE.IPRINT) GO TO 485
WRITE(6,1001)I,XT,YT,PDAM
485 TGTYNM(1)=TGTYNM(1)+100000.
TGTYSV=TGTVEL(1)
TMTOFR(1)=9990.
TGTVEL(1)=0.
LEFTIN=LEFTIN-1
IF(NTTCS(1).GT.0)NSPLFT=NSPLFT-1
NKILLT=NKILLT+1
NKILL(1:ITT)=NKILL(1:ITT)+1
```

print the optional output, if required, and increment several counters. The last statements in the subroutine:

```
IF(NDFA.LT.1)GO TO 500
DO 487 L=1,NDFA
487 CALL STINT(1SVIFA,I,L,IBIT,KOFTN,2)
GO TO 500
490 IF(IRUNS.EQ.IPRINT)WRITE(6,1002)I,XT,YT,PDAM
500 CONTINUE
RETURN
END
```

remove the intruder from all direct fire areas and print additional output, if required.

### Subroutine DIRFIR

The purpose of this subroutine is to evaluate the effects of the direct fire munitions against either the intruder or defender targets. The variable IDAM controls whether direct fire or return fire is being delivered. The first executable statements:

```
BIGG(X) = 5 * SQRT(1 - EXP(-.63 * X ** 2))
ID = IDAM
IDAM = 0
ITT = INTGTYP(IIS)
IWT = INTDEF(INDF)
L = IWT
IF (ID .GT. 1) L = ITT
BREL = AREL(L, ID)
BCEP = ACEP(L, ID)
KODEI = AE(IWT, ITT, 2 * ID - 1)
EIVAL = AE(IWT, ITT, 2 * ID)
IF (KODEI .GT. 1) GO TO 50
IND = IDP(INDF, KI, ID)
```

initialize several variables. If the effectiveness index is in terms of the mean area of effectiveness for fragmentation in the ground plane ( $MAE_f$ ), the statements:

```
ALWRAT = 1 - B * AI(IND)
ETL = 2 * SQRT(EIVAL * ALWRAT / 3.141592654)
ETW = ETL / ALWRAT
DENOM = 17.6 * AREP(IND) ** 2 * ETL ** 2
RSSP = ETL / SQRT(DENOM)
DENOM = 17.6 * ADEP(IND) ** 2 * ETW ** 2
DSSP = ETW / SQRT(DENOM)
SSPD = RSSP * DSSP * BREL
GO TO 100
```

compute the target length-to-width ratio [Equation (25)], the effective target length and width [Equations (26) and (27)], and the single shot probability of damaged based on the range and deflection single shot probability of hit [Equations (28), (29), and (30)]. The statements:

```
50 IF (BCEP .GT. 0) GO TO 60
XD = (TGTXOL(IIS) - DEFY(INDF)) ** 2
YD = (TGTYNW(IIS) - DEFY(INDF)) ** 2
RNG = SQRT(XD + YD)
BCEP = .001 * RNG * ABS(BCEP)
```

compute the distance between the intruder and defender, and convert the circular error probable from mils to feet [Equation (31)]. The next statements:

```

60 DENOM=1.7*BCEP
   IF (10.GT.1) GO TO 70
   VERT=TARHT(IIT)
   HORIZ=2.*TARRAD(IIT)
   IF (HORIZ.LT..01) HORIZ=(TARL(IIT)+TARH(IIT))/2
   GO TO 80

```

compute the vertical and horizontal dimensions of the intruder. If direct fire munitions are employed against the defender, the statements:

```

70 VERT=DEFHT(IWT)
   HORIZ=2.*DEFRAD(IWT)
   IF (HORIZ.LT..01) HORIZ=(DEFL(IWT)+DEFH(IWT))/2

```

compute the vertical and horizontal dimensions of the defender. In the next statements:

```

80 A1=HORIZ/DENOM
   B1=VERT/DENOM
   R5SP=2.*BIGG(A1)
   D5SP=2.*BIGG(B1)
   S5PD=R5SP*D5SP*BREL*EIVAL

```

the single shot probability of damage [Equations (32), (33), and (34)] is computed. The last statements in the subroutine:

```

100 RN=RANF(DUMMY)
    IF (RN.LT.S5PD) IDAM=1
    PR=ITY-S5PD
    RETURN
    END

```

obtain a uniform random number. The random number is compared to the single shot probability of damage [Inequality (35)], and, if the target is damaged, the variable IDAM is set to one to so indicate to the calling routine.

#### Subroutine CKEVTM

The purpose of this subroutine is to determine if any time ordered events should occur before the distance related event takes place and to sequence the events in the proper order. The possible time ordered events are direct fire shots, return fire shots, or the delivery of the second or subsequent volley at an indirect fire aimpoint. The first statements:

```

DIMENSION TM(130), IPOS(130)
IF (NDF.A.GT.0) GO TO 50
IF (INDFA.GT.0) GO TO 50

```

provide for local storage and determine if any time related events have previously occurred. If time ordered events have not occurred previously and the event is not an encounter of a direct fire area entrance boundary, the statements:

```

IF (NOBTP2.LT.16.OR.NOBTP2.GT.16) GO TO 20
IF (NOBTP2.GT.16) GO TO 15
INDFA=1
GO TO 20
15 CALL STINT( ISVIFA, ISAVE, IT, IBIT, KDFTN, I)
NDF=MAX0( IT, NDF)
CALL TGTMOV
16 DO 18 J=1, NDFWD
IF (NDFEFA( IT, J), LT. 1) GO TO 18
NDF=NDFEFA( IT, J)+100
IF (TMTOFR( NDF), GT. 0.0.AND.TMTOFR( NDF), LT. 9000.) GO TO 18
TMTOFR( NDF)=0
18 CONTINUE
RETURN
20 CALL TGTMOV
RETURN

```

provide for a call to Subroutine TGTMOV. If the intruder has encountered a direct fire area entrance boundary, Subroutine STINT is called to save this intruder number in the ISVIFA array for subsequent evaluation. Then, Subroutine TGTMOV is called, and all defenders that may fire into the direct fire area are determined. Control is then transferred to the calling routine. If there are time ordered events to occur, the next statements:

```

50 DO 60 I=1, 130
IPOS( I)=1
TM( I)=TMTOFR( I)
60 CONTINUE
DO 100 I=1, 130
DO 90 J=1, 130
IF (TM( I)-TM( J)) 90, 90, 85
85 KEEP=IPOS( I)
IPOS( I)=IPOS( J)
IPOS( J)=KEEP
SAVE=TM( I)
TM( I)=TM( J)
TM( J)=SAVE
90 CONTINUE
IF (TM( I).GT. TRAVTM) GO TO 120
100 CONTINUE
120 NUMCNT=MIN0( I, 130)

```

sort the event times for all time ordered events until an event time is found which is greater than the travel time to the next position related event. If the position related event in the next event to occur, the statements:

```

IF (NUMCNT.GT.1) GO TO 150
CALL TGTMOV
IF (NOBTP2.NE.17) RETURN
CALL STINT (ISVIFA,ISAVE,IT,IBIT,KOFFTN,1)
NDFA=MAXD (IT,NDFA)
GO TO 16

```

provide for a call to Subroutine TGTMOV, and, if the intruder has encountered an entrance to a direct fire area, Subroutine STINT is called to save the intruder number in the ISVIFA array for subsequent evaluation. Next, a DO LOOP is entered to consider each time ordered event. The statements:

```

150 NUM=NUMCNT-1
   ISHOT=0
   DO 1000 NI=1,NUM
     N=IPOS (NI)
     ITDI=TIMTOE (N)

```

initialize the variable that signals that a time related event may occur and determine the event to occur and the event time. If the time ordered event can occur, the statements:

```

IF (ITDI.GT.9000) GO TO 1000
TT=TRAVT
TRAVT=ITDI
CALL TGTMOV
TRAVT=TT-ITDI

```

provide a call to Subroutine TGTMOV and decrement the travel time to the position related event by this event time. The next statements:

```

IF (M.GT.120) GO TO 750
IF (M.LT.101) GO TO 500

```

provide branching to sections of the subroutine that evaluate indirect fire volleys and intruder targets returning direct fire. If the event is one in which the defender employs direct fire munitions, the statements:

```

NDF=M-100
IF (NRAFDF (NDF) LT.1) GO TO 900
NDTYP=INTDEF (NDF)

```

determine the defender number and, if the defender has any rounds remaining, the defender type is obtained. Next, three nested DO LOOPS are evaluated to determine if this defender may fire at an intruder. The statements:

```
DO 300 I=1,5
  IPRR=1101PRINDTYP,11
  IF(IIPRR.LT.11) GO TO 1000
```

consider each priority value from the defenders ordered list of intruder types to fire upon. If the priority value is zero, the defender cannot fire, and the next time related event is considered. If there are intruder types that the defender may shoot at, the statements:

```
DO 280 IIS=1,NTGO
  IF(NTGTYP(IIS).NE.IPRR) GO TO 280
```

consider each intruder. If the intruder type is of the type the defender may fire upon, the statements:

```
DO 270 K=1,NQFA
  CALL STINT(IISVIFA,IIS,K,IBIT,KDFITN,3)
  IF(1BIT.LT.11) GOTO 270
  IF(INDEFEA(K,NDF).LT.11) GO TO 270
  GO TO 320
270 CONTINUE
280 CONTINUE
300 CONTINUE
GO TO 1000
```

consider each direct fire area. Subroutine STINT is called to determine if this intruder is in a direct fire area. If the intruder may be fired upon by the defender, the statements:

```
320 TMTOFR(M)=IMBRD(INDTYP)
  KI=K
  IDAM=1
  ISHOT=1
  CALL DIRFIR(IDAM)
  IF(1SYMP(2).GT.0) CALL SYMDET
  NRADFD(NDF)=NRADFD(NDF)-1
  IF(NRADFD(NDF).LT.1) TMTOFR(M)=9991
  NRFBDT(INDTYP)=NRFBDT(INDTYP)+1
```

increment several counters and provide a call to Subroutine DIRFIR. If sympathetic detonations are evaluated for direct fire munitions, Subroutine SYMDET is called. If optional output is required, the statements:

```

IF(IRUNS.NE (PRINT) GOTO 340
AKIL=0H MISSED
IF(DAM.GT.0)AKIL=0H HIT HIM
PRINT 9003,NDF,NDTYP,IIS,NIGTYP(IIS),AKIL,PROBITY

```

print the defender number and type, the intruder number and type, a message indicating whether the intruder was damaged by the direct fire munition, and the single shot probability of damage. If the intruder was damaged, the statements:

```

340 IF(DAM.LT.1) GO TO 475
TGTYNM(IIS)=TGTYNM(IIS)+100000.
IF(NGTAWT.GT.0)CALL UNIT
LEFTIN=LEFTIN-1
TGTVEL(IIS)=0.
TMTOFR(IIS)=9999.
CALL STINT(ISVIFA,IIS,K,IBIT,KOFTTN,2)
INTYP=INTYP(IIS)
NKILLD(INTYP)=NKILLD(INTYP)+1
NKILLT=NKILLT+1
IF(INTTCS(INTYP).GT.0) NSPLFT=NSPLFT-1
KI=IIS
XFIX=0.0

```

increment several counters, provide calls to Subroutine UNIT if any intruders are associated as a group, and Subroutine STINT to turn the flag bit off for this intruder in the direct fire area. If the intruder must be diverted around by subsequent intruders in the column, the statement:

```

IF(INTTMD(INTYP).GT.0)CALL DIVSET

```

provides a call to Subroutine DIVSET to save the intruder Y coordinate and set the dimension direction. The next statements:

```

475 J=0
DO 485 I=1,NTGO
CALL STINT(ISVIFA,I,K,IBIT,KOFTTN,3)
IF(IBIT.LT.1) GO TO 485
NRFIRD(I)=NRFIRD(I)+1
IF(NRFIRD(I) NRBA)485,480,485
480 TMTOFR(I)=0.
J=J+1
485 CONTINUE
IF(J)50,990

```

consider each intruder. If an intruder has observed the required number of rounds to be able to return fire, the array containing the times for the time ordered events is updated and a sort of the updated array is performed. If the time

ordered event is one in which the intruder is returning fire, the statements:

```
500 NINTT=NIGTYPIM)
   IF (NRADF(EM) LT 1000 TO 900
```

determine the intruder type. If the intruder has direct fire rounds remaining, three nested DO LOOPS are evaluated to determine if this intruder may fire at a defender. The statements:

```
JO 600 I=1,5
IPRR=DFOPR(NINTT,II)
IF (IPRR LT 1) GO TO 1000
```

consider each priority value from the intruders ordered list of defender types to fire upon. If the priority value is zero, the intruder cannot fire and the next time ordered event is considered. If there are defender types the intruder may shoot at, the statements:

```
DO 590 J=1,NDFWD
IF (LWDEF(J).NE. IPRR) GO TO 590
```

consider each defender. If the defender type is of the type the intruder may shoot at, the statements:

```
DO 580 K=1,NDFA
IF (NDEFEAK(J).LT 1) GO TO 580
CALL STINT(SVIFA,M,K,IBIT,KOFTTN,3)
IF (IBIT LT 1) GO TO 580
GO TO 620
580 CONTINUE
590 CONTINUE
600 CONTINUE
GO TO 1000
```

consider each direct fire area. Subroutine STINT is called to determine if the intruder may fire from the direct fire area at a defender. If a defender may receive return fire, the statements:

```

520 TMTDFR(MI)+TMBP(I)NINTI)
K1=K
IDAM=2
NDF=NDFEAK(J)
ISHOT=1
IIS=M
CALL DIRFIR(IDAM)
NRADF(IIS)=NRADF(IIS)+1
IF(NRADF(IIS).LT.1)TMTDFR(MI)=9999
NRFBI(I)NINTI)+NRFBI(I)NINTI)+1

```

increment several counters and provide a call to Subroutine DIRFIR. If optional output is required, the statements:

```

IF(IIRUNS NE.1)PRINTI GO TO 640
AKIL=8H MISSED
IF(IDAM GT 0) AKIL= 8H HIT HIM
PRINT 9005,M,NINTI,NDF,(TMTDFR(MI),AKIL,PRBITY

```

print the intruder number and type, the defender number and type, a message indicating whether the defender was damaged, and the single shot probability of damage. If the defender was damaged, the statements:

```

640 IF(IDAM.LT.1) GO TO 990
NDFEAK(J)=-NDF
NDTYP=(TMTDEF(NDF)
NDFDAM(NDTYP)=NDFDAM(NDTYP)+1
TMTDFR(NDF+100)=9999.
GO TO 990

```

increment several counters. If the time ordered event is one in which the second or subsequent indirect fire volley is involved, the statements:

```

750 IVAP=M-120
IF(NVLIDF(IVAP) LT.1)GO TO 900
CALL NDFIRE
TMTDFR(MI)+TDBIFV(IVAP)
NVLIDF(IVAP)=NVLIDF(IVAP)-1
ISHOT=1
GO TO 990

```

determine the volley aimpoint number. If all volleys have not been delivered at this aimpoint, a call to Subroutine NDFIRE is provided and several counters incremented. The next statements:

```

900 TMTDFR(MI)+9999.
GO TO 1000

```

set the element of the array containing the times for the time ordered events to 9990 to indicate that an event will no longer be considered on a time basis. The next statements:

```
990 IF(TIMEORDER(I) GT TRAVTIME) GO TO 1000
    GO TO 50
1000 CONTINUE
```

determine if the time for a time ordered event to occur is less than the travel time to the next position related event, and if so, provides branching to the section of the routine that will sort the array containing the times for the time ordered events to occur. Otherwise, the next time ordered event is considered. The last statements in the subroutine:

```
      IF(I5H01)50,1050
1050 IF(NOBIP2-17)20,15,20
      END
```

evaluate the signal that a time related event occurred, and, if one did occur, appropriate sections of the routine are branched to.

#### Subroutine STINT

The purposes of this subroutine are to turn on a bit to indicate that an intruder has entered a direct fire area, to turn off a bit to indicate that an intruder has exited a direct fire area, and to test a bit to determine if an intruder is located within a direct fire area. The first executable statements:

```
K=INT
KT=IT
I=I
IF(K.LT.59) GO TO 10
KT=IT+15
K=K-59
```

save the intruder number and the direct fire area number. For the intruder target numbers greater than 58, the direct fire area number is incremented by 15 and the intruder target number is decremented by 59. The variable KT is used for the subscript of the ISV(30) array and the variable K is used for the number of the bit in the word. The next statement:

transfers to the proper section of the routine for the option desired. If the call to Subroutine STINT was made for the purpose of recording an intruder in a direct fire area, the statements:

```
20 ISV(KT)=ISV(KT).OR SHIFT(I,K)
   KDF(TN(INI)+KDF(TN(INI)+1)
   RETURN
```

shift a 1 into the appropriate bit position and increment the counter for the number of direct fire areas this intruder target is located in. If Subroutine STINT was called when an intruder left a direct fire area, the statements:

```
50 ISW=0
   J=ISW
   KK=J.OR SHIFT(I,K)
   KK=.NOT KK
   ISV(KT)=ISV(KT) AND KK
   KDF(TN(INI)-KDF(TN(INI)-1)
   RETURN
```

turn the flag bit off to so indicate and decrement the count of direct fire areas this intruder is located in. The last statements in the subroutine:

```
100 IBIT=SHIFT(ISV(KT),-K) AND 10
   RETURN
   END
```

test the status of an intruder and place the bit whether ON or OFF into the variable IBIT for interrogation in the calling routine.

#### Subroutine EXPSWP

The purpose of this subroutine is to evaluate the effects of line charges and fuel air explosives as sweeping devices against the mines within the range of influence. The first statements:

```
XLOC=OBX
K1=KTSAV(IGAVE)
IOB(KT)=-1
NB=NOBTP2
```

save the event X coordinate and remove the event from further consideration. If line charges are employed, the following statements:

```
IF (IRUNS EQ IPRINT) WRITE(6,1000)
IF (LCFOPT GT 1) GO TO 20
YDIST=OBY*ALNCLC
RANGE=AWIDLCL
GO TO 50
```

set the forward and lateral limits of the line charge pattern effects. The following statements:

```
20 YLOC=OBY*DFIAE
YDIST=YLOC*RADEAE
YST=YLOC*RADEAE
RANGE=RADEAE**2
```

define the limits of the FAE pattern effects [Equations (20), (21), and (22)]. The next statements:

```
30 KI=KI+1
IF (IOBKTI) LT 01 GO TO 30
INORD=IOBKTI
CALL UNPACK
IF (INORD TP2-B) 40,200,30
40 IF (IOBY LT YST) GO TO 30
GO TO 60
```

determine whether a mine is within range of the FAE effects. For line charges, the next statements:

```
50 KI=KI+1
IF (IOBKTI) LT 01 GO TO 50
INORD=IOBKTI
CALL UNPACK
IF (INORD TP2-B) 60,200,50
60 IF (IOBY GT YDIST) GO TO 200
```

determine whether a mine is within the forward limits of the line charge effects. The statements:

```
DIST=ABS(XLOC-OBX)
IF (LCFOPT GT 1) DIST=DIST**2*(YLOC-OBY)**2
IF (DIST GT RANGE) GO TO 50
```

determine the lateral distance from the mine to the center of the exploding line charge or FAE. If the distance is greater than the effective range of the sweeping device, another mine is chosen for consideration. Next, a uniform random number is compared to the probability of detonation from line charge or FAE effects [Inequality (23)]. If the mine was detonated, the statements:

```

RN=RANF(DUMMY)
IF(RN.GT.PDLCF)GO TO 50
IF(IRUNS.EQ.IPRINT)WRITE(6,1010) NOBTP2,OBX,OBV
NMSWPT(NOBT2)=NMSWPT(NOBT2)+1
NMDET(NOBT2)=NMDET(NOBT2)+1
CALL BOOM(IDET,1)
JOB(KT)=-1
GO TO 50

```

remove the mine from further consideration and increment several counters. Subroutine BOOM is called to turn on the detonation flag for the mine. After all the mines for this travel path segment have been considered, or when a mine is located which is outside the forward effects of the line charge or FAE, a DO LOOP is next entered to increment the sweep delay for all active targets in the column. The statements:

```

200 DO 250 I=1,NTGO
IF(TGTYNM(I).GT.99969.)GO TO 250
IF(INCTAM(I).NE.NCTAM(1)SAVE)GO TO 250
SWPDEL(I)=SWPDEL(I)+TMDLCF
DELATM(I)=TMDLCF
TGTVEL(I)=0.
250 CONTINUE
OBX=XLOC
NOBT2=NB
RETURN
END

```

consider an active target in the same column as the event target and increment the sweep delay time by the time delay assessed for employment of line charges or FAE.

#### Subroutine TGTMOV

The purpose of this subroutine is to move all active intruders the event distance and time and to adjust the assessed delays and event times by the time for the next event. The first statements:

```

DO 500 I=1,NTGO
IF(TGTYNM(I).GT.99990.)GO TO 500
IF(TGTVEL(I).GT.0.)GO TO 500
DELATM(I)=DELATM(I)-TRAVTM

```

```
IF (DELATM(I)) GT 0.00001 GO TO 550
```

decrement the delay time for the active intruders that are stopped. If the time for the next event is greater than the remaining delay time, the following statements:

```
TGTYN(I) = TGTSPO  
DTM = ABS(DELATM(I))  
DELATM(I) = 0.  
TGTYN(I) = TGTYN(I) + TGTSPO * DTM  
TIMEMV(I) = TIMEMV(I) + DTM  
IF (THTOFR(I) GT 9000.) GO TO 600  
THTOFR(I) = THTOFR(I) - DTM  
IF (THTOFR(I) LT 0.) THTOFR(I) = 0.  
GO TO 600
```

reset the intruder velocity, initialize the delay time for the intruders, and adjust the time to fire by the remaining delay time. The next statements:

```
500 TGTYN(I) = TGTYN(I) + TGTSPO * TRAVTM  
TIMEMV(I) = TIMEMV(I) + TRAVTM  
550 IF (THTOFR(I) GT 9000.) GO TO 600  
THTOFR(I) = THTOFR(I) - TRAVTM  
IF (THTOFR(I) LT 0.) THTOFR(I) = 0.  
600 CONTINUE
```

move the active intruders the event distance and time, and adjust the time to fire for all active direct fire intruders by the time for the next event. If there are any defender direct fire weapons, the statements:

```
IF (NDFWD LT 1) GO TO 750  
NEND = NDFWD + 100  
DO 700 I = 101, NEND  
IF (THTOFR(I) GT 9000.) GO TO 700  
THTOFR(I) = THTOFR(I) - TRAVTM  
IF (THTOFR(I) LT 0.) THTOFR(I) = 0.  
700 CONTINUE
```

adjust the time to fire for all active defenders. The last statements in the subroutine:

```
750 IF (NVAP LT 1) RETURN  
NEND = 120 + NVAP  
DO 900 I = 121, NEND  
IF (THTOFR(I) GT 9000.) GO TO 900  
THTOFR(I) = THTOFR(I) - TRAVTM  
IF (THTOFR(I) LT 0.) THTOFR(I) = 0.  
900 CONTINUE
```

RETURN  
END

adjust the time to fire for all indirect fire volley aim-  
points.

## INITIAL DISTRIBUTION

DDC	2
DRXSY/GB	3
SMJPA/DW	2
DRC PM-SA-CP	1
USN Weapons Cntr/Code 317	1
USAMERDC/DRXFB-O	1
Nav Coastal Sys Lab/Code 722	1
TFWC/SA	1
AFAITC/TINAT	1
USACAA/MOCA-SA-1	1
Nav Surface Wpn Cntr/DL	1
Air Force Sys Comd/SDW	1
USA Training and Doctrine Comd/ ATCD-CM	1
TAC/DRA	1
AFATL/DLOSL	9
AFATL/DLYW	10
ADTC/XRO	1
AFATL/DL	1
AUL (AUL-LSE-70-239)	1
ASD/ENFEA	1
TAWC/TRADOCLO	1
Hq USAF/SAMI	1
Ogden ALC/MMM	2
AFIS/INTA	1
Hq USAFE/DOQ	1
Hq PACAF/DOO	1