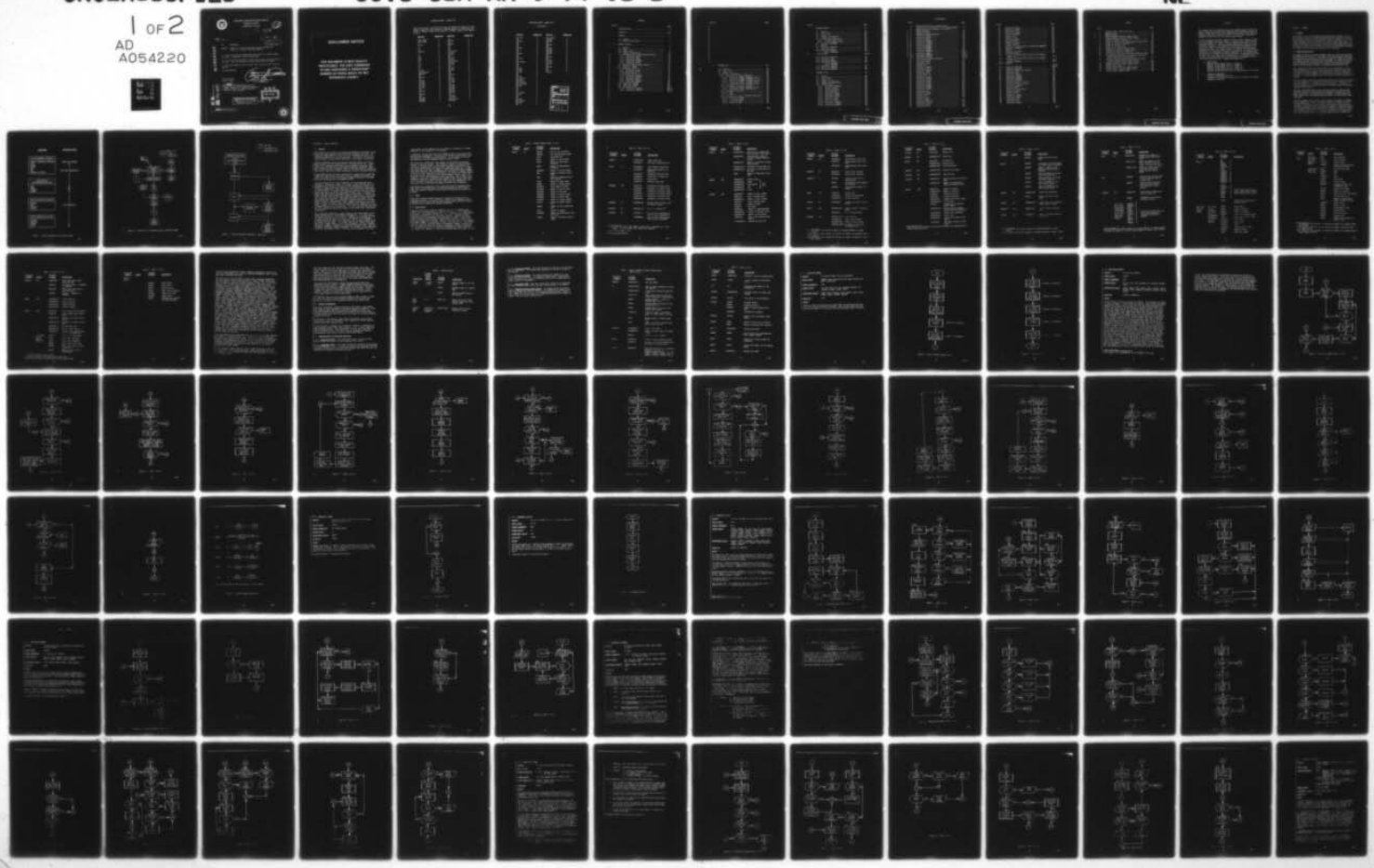
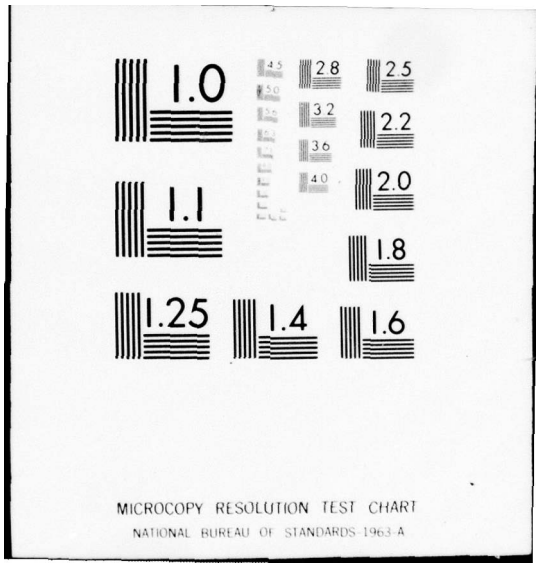
AD-A054 220 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON D C F/G 15/7
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK) PROGR--ETC(U)
APR 78

UNCLASSIFIED CCTC-CSM-MM-9-74-V3-3 NL

1 OF 2
AD
A054220




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

1 1 11
12
See AD39386
FOR FURTHER TRAN"

DEFENSE COMMUNICATIONS AGENCY

COMMAND AND CONTROL
TECHNICAL CENTER
WASHINGTON, D. C. 20301

IN REPLY
REFER TO: C314

11 APR 1978
12 172 p.

TO: RECIPIENTS

SUBJECT: Change 3 to Program Maintenance Manual CSM MM 9-74, Volume III, Weapon Allocation Subsystem

1. Insert the enclosed change pages and destroy the replaced pages according to applicable security regulations.

2. A list of Effective Pages to verify the accuracy of this manual is enclosed. This list should be inserted before the title page.

3. When this change has been posted, make an entry in the Record of Changes.

FOR THE DIRECTOR

J. DOUGLAS POTTER
Assistant to the Director
for Administration

173 Enclosures
Change 3 pages

AD A054220

AD No. ___
DDC FILE COPY

CCTC
6 The [redacted] Quick-Reacting General War Gaming System (QUICK) Program Maintenance Manual. Volume III. Weapon Allocation Subsystem. Part I. Change 3.

DDC
RECEIVED
MAY 18 1978
E

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

14 CCTC [redacted]SSC-CSM-MM-9-74-V3-3

409658

2

AMERICAN REVOLUTION BICENTENNIAL 1776-1976

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

EFFECTIVE PAGES - AUGUST 1977

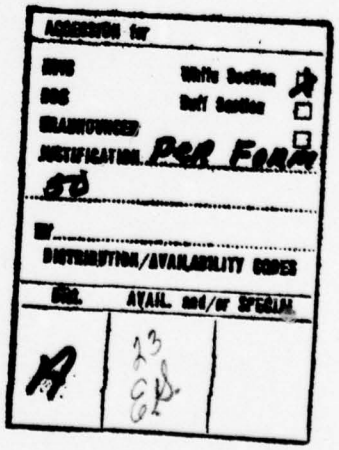
This list is used to verify the accuracy of CSM MM 9-74 Volume III after change 3 pages have been inserted. Original pages are indicated by the letter 0, change 1 pages by the numeral 1, change 2 pages by the numeral 2, etc.

Page No.	Change No.	Page No.	Change No.
Front Cover	1	202	1
Title Page	1	203-206	0
ii	1	207	1
iii-iv	3	208-212	0
v	0	213	1
vi	3	214	0
vii	1	215	1
viii-ix	3	215.1-215.2	1
x	1	216-217	1
xi	3	218-219	0
xii	0	220	1
xiii	3	221-235	0
xiv	0	236-239	1
1-4	3	239.1-239.2	1
5-6	0	240-243	1
7-8	3	244-246	0
9	2	247	1
10	0	248-255	0
11-104	3	256-257	1
104.1-104.20	3	258	0
105-106	0	259	1
107	2	260-263	0
108-110	0	264	1
111	3	265-272	0
112-132	0	273-274	1
133-136	3	275-290	0
137-138	2	291	1
139-140	0	292-300	0
141-144	3	301-302	1
145	0	303-305	0
146-150	3	306-307	1
151	1	308-309	0
152-155	0	310	1
156	1	311-316	0
157-158	0	317-318	1
188-189	1	318.1-318.2	1
190-201	0	319-320	1

EFFECTIVE PAGES - AUGUST 1977

(continued)

Page No.	Change No.	Page No.	Change No.
321-324	0	387.1-387.2	1
325	1	388-392	0
326-334	0	393-394	2
335	3	395	0
336	1	396-398	3
337	0	399	0
338-340	1	400-401	2
341	3	402-403	0
342	0	404-405	2
343-344	3	406-464	0
345	0	465	2
346	1	466-467	0
347-350	3	468	2
351	1	469-470	0
351.1-351.2	1	471-472	3
352	0	473-475	0
353	1	476	3
354	0	477-490	0
355-356	1	491	1
357-358	0	492	0
359-360	1	493-494	1
361	0		
362	3		
363-364	1		
364.1-364.2	1		
365-368	1		
368.1-368.12	1		
368.13-368.22	3		
369-370	0		
371	1		
372	3		
373	1		
373.1-373.2	1		
374-375	1		
376	3		
377	0		
378-384	1		
384.1-384.2	1		
385-386	0		
387	1		



CONTENTS

Section		Page
	ACKNOWLEDGMENT	ii
	ABSTRACT	xiii
1	GENERAL	1
	1.1 Purpose	1
	1.2 General Description	1
2	MODULE PREPALOC	7
	2.1 Purpose	7
	2.2 Input	8
	2.3 Output Files	8
	2.4 Concept of Operation	23
	2.5 Identification of Subroutine Functions	23
	2.5.1 Subroutine TARLST	23
	2.5.2 Subroutine PARTA	23
	2.5.3 Subroutine WEPPREP	25
	2.5.4 Subroutine TGTPREP	25
	2.5.5 Subroutine PARTB	25
	2.6 PREPALOC Internal Common Blocks	25
	2.7 Subroutine ENTMOD	28
	2.8 Subroutine TARLST	31
	2.8.1 Subroutine INITPG	51
	2.8.2 Subroutine INITAR	53
	2.9 Subroutine PARTA	55
	2.9.1 Subroutine DEPROUT	62
	2.9.2 Subroutine FACTORCG	68
	2.9.3 Subroutine FIXWEP	81
	2.9.4 Subroutine MAKECHG	89
	2.9.5 Subroutine PENROUT	96
	2.10 Subroutine WEPPREP	100
	2.11 Subroutine TGTPREP	104
	2.12 Subroutine PARTB	104.6
	2.12.1 Subroutine COMPWRIT	104.11
	2.12.2 Subroutine GRPWRIT	104.16

Section Page

3 PROGRAM ALOC 105

3.1 Purpose 105
3.2 Input Files 105
3.3 Output Files 106
3.4 Concept of Operation 109
3.4.1 Constraint Functions (Optional) 109
3.4.1.1 Subroutine RNGEMOD (RANGEMOD Option) ... 110
3.4.1.2 Subroutine MINRNGE (MINRANGE Option) ... 110
3.4.1.3 Subroutine MIRVRST (MIRVREST Option) ... 110
3.4.1.4 Subroutine FLAGRST (FLAGREST Option) ... 110
3.4.1.5 Subroutine LOCREST (LOCREST Option) 110
3.4.2 Convergence Functions (Optional) 110
3.4.2.1 Subroutine READMUL (READMUL Option) 110
3.4.2.2 Subroutine PUNCHM (PUNCH Option) 110
3.4.3 Termination Functions 110
3.4.3.1 STOP 111
3.4.3.2 DUMP 111
3.4.4 Allocation Function - ALLOCATE (Required) .. 111
3.4.4.1 Subroutine MULCON 117
3.4.4.2 Subroutine GETDTA 121
3.4.4.3 Subroutine STALL 121
3.4.4.4 Subroutine WAD 128

Section		Page
4	PROGRAM EVALALOC	335
	4.1 Purpose	335
	4.2 Input Files	335
	4.3 Output File	335
	4.4 Concept of Operation	335
	4.5 Common Block Definition	338
	4.5.1 External Common Blocks	338
	4.5.2 Internal Common Blocks	339
	4.6 Subroutine BOMRPAKR	352
	4.7 Deleted	355
	4.8 Subroutine EVALPLAN	357
	4.9 Subroutine EVAL2	362
	4.9.A Subroutine MAKEPRFL (Entry PRNTFILE)	368.1
	4.9.B Deleted	368.13
	4.10 Subroutine MISLPAKR	369
	4.11 Subroutine PACK	371
	4.12 Subroutine SEARCH	376
	4.13 Subroutine SSSPCALC (Entry INITPROB)	378
	4.14 Subroutine TGTMODIF	380
	4.15 Subroutine UNPACKER	385
	4.16 Subroutine WPNMODIF	388
5	PROGRAM ALOCOUT	391
	5.1 Purpose	391
	5.2 Input Files	391
	5.3 Output Files	391
	5.4 Concept of Operation	391
	5.5 Common Block Definition	395
	5.6 Overlay ALOCO1	406
	5.6.1 Subroutine COMPRESS	416
	5.6.2 Function CUMINV	418
	5.6.3 Subroutine DGZSEL	420
	5.6.4 Function ERGOT1	425
	5.6.5 Subroutine FILTGT	427
	5.6.6 Subroutine FINDMIN	429
	5.6.7 Subroutine F2BMIN	435
	5.6.8 Subroutine GRADF	437
	5.6.9 Subroutine MOVE	439
	5.6.10 Subroutine PERTBLD	441
	5.6.11 Subroutine PROCCOMP	443
	5.6.12 Subroutine PROCMULT	449

PRECEDING PAGE BLANK

ILLUSTRATIONS

Figure		Page
1	Major Subsystems of the QUICK System	2
2	Procedure and Information Flow in QUICK/HIS 6080	3
3	Weapon Allocation Subsystem - Data Flow	4
4	Module PREPALOC	29
5	Subroutine TARLST	33
6	TARFILE Packing Description	50
7	Subroutine INITPG	52
8	Subroutine INITAR	54
9	Subroutine PARTA	56
10	Subroutine DEPROUT	63
11	Subroutine FACTORCG	71
12	Subroutine FIXWEAP	83
13	Subroutine MAKECHG	91
14	Subroutine PENROUT	97
15	Subroutine WEPPREP	101
16	Subroutine TGTPREP	104.2
17	Subroutine PARTB	104.7
17.1	Subroutine COMPWRIT	104.12
17.2	Subroutine GRPWRIT	104.17
18	ALOC Calling Sequence Hierarchy	112
19	Subroutine MULCON	119
20	Subroutine STALL	123
21	Subroutine WADOUT	129
22	Program ALOC	153
23	Subroutine FORMATS	157
24	Subroutine INITALC	159
25	Subroutine PRNTNOW	161
26	Subroutine PUPDT	163
27	Subroutine SETSAL.	166
28	Overlay OVAL1	170
29	Subroutine FLAGRST	173
30	Subroutine LOCREST	176
31	Subroutine MIRVRST	179
32	Subroutine PUNCHM	181
33	Subroutine RDALCRD	187
34	Subroutine READMUL	191
35	Subroutine RNGEMOD	194
36	Subroutine TIMEPRT	197
37	Overlay MULCON	211
38	Subroutine ADDSAL	222
39	Subroutine BOMPRM	225
40	Function FMUP	228
41	Subroutine GETDTA	
42	Function LAMGET	248
43	Subroutine PREMIUMS	251
44	Subroutine PRNTALL	253
45	Subroutine PRNTCON	256
46	Subroutine RECON	258

PRECEDING PAGE BLANK

Figure		Page
47	Subroutine SALVAL	264
48	Subroutine SETABLE	270
49	Subroutine SETPAY	273
50	Subroutine SORTMIS	277
51	Function TABLEMUP	279
52	Segment STALL	283
53	Subroutine WAD	299
54	Subroutine WADOUT	317
55	Segment DEFALOC	325
56	Subroutine RESVAL	333
57	Program EVALALOC	337
58	Location of Packed Values in Subroutine BOMRPAKR	353
59	Subroutine BOMRPAKR	354
60	Deleted	356
61	Subroutine EVALPLAN	359
62	Subroutine EVAL2	364
62.1	Location of Packed Target Values for the Sample Target List	368.3
62.2	Location of Packed Weapon Values for the Sample Target List	368.5
62.3	Subroutine MAKEPRFL	368.6
62.4	Deleted	368.15
62.5	Subroutine MKTARTAB	368.16
63	Subroutine MISLPAKR	370
64	Format of JORDER Array Element	372
65	Subroutine PACK	373
66	Subroutine SEARCH	377
67	Subroutine SSSPCALC	379
68	Subroutine TGTMODIF	382
69	Subroutine UNPACKER	387
70	Subroutine WPNMODIF	389
71	Calling Hierarchy of ALOCO1	407
72	Subroutine ALOCO1; Overlay 1	410
73	Subroutine COMPRESS	417
74	Function CUMINV	419
75	DGZSEL Calling Hierarchy	421
76	Subroutine DGZSEL	422
77	Function ERGOT1	426
78	Subroutine FILTGT	428
79	Subroutine FINDMIN	431
80	Subroutine F2BMIN	436
81	Subroutine GRADF	438
82	Subroutine MOVE	440
83	Subroutine PERTBLD	443

TABLES

Table		Page
1	TGTFILE Format (Target Data Block)	9
2	BASFILE Format	11
3	TARFILE Format.....	24
4	Module PREPALOC Internal Common Blocks	26
5	Program PREPALOC Internal Common Blocks	41
6	Default Parameter Settings	78
7	Format for ALOCTAR File Logical Record Data Blocks	107
8	Format of Records on MSLTIME File	109
9	Format for WPNTGT Files	114
10	Format of the ITDAC File	115
11	Program ALOC External Common Blocks	133
12	Program ALOC Internal Common Blocks	140
13	Calculated Formats for Variables	156
14	Illustrating Calculation of Actual Payoff on Target	289
15	Illustrating Quantities Calculated for Potential Weapon Added and Deleted Payoffs	292
16	Illustrating Quantities Pre-Calculated for Each Potential Weapon Before WAD is Called	294
17	Program EVALALOC External Common Blocks	341
18	Program EVALALOC Internal Common Blocks	346
19	TMPALOC File Format	392
20	Overlay ALOC01 External Common Blocks	396
21	Overlay ALOC01 Internal Common Blocks	400
22	Overlay ALOC02 Common Blocks	404

PRECEDING PAGE BLANK

ABSTRACT

The computerized Quick-Reacting General War Gaming System (QUICK) will accept input data, automatically generate global strategic nuclear war plans, provide statistical output summaries and produce input tapes to simulator subsystems external to QUICK. QUICK has been programmed in FORTRAN for use on the CCTC HIS 6000 computer system.

The QUICK Program Maintenance Manual consists of four volumes: Volume I, Data Management Subsystem; Volume II, Weapon/Target Identification Subsystem; Volume III, Weapon Allocation Subsystem; Volume IV, Sortie Generation Subsystem. The Program Maintenance Manual complements the other QUICK Computer System Manuals to facilitate maintenance of the war gaming system. This volume, Volume III provides the programmer/analyst with a technical description of the purpose, functions, general procedures, and programming techniques applicable to the programs and subroutines of the Weapon Allocation subsystem. The associated program listings which are dynamic and voluminous are not contained herein. However, the program listings may be obtained by arrangement with the CCTC QUICK Project Officer. Companion documents are:

a. USERS MANUAL
   Computer System Manual UM 9-77, Volume I
   Computer System Manual UM 9-77, Volume II
   Computer System Manual UM 9-74, Volume III
   Computer System Manual UM 9-74, Volume IV
   Provides detailed instructions for applications of the system

b. TECHNICAL MEMORANDUM
   Technical Memorandum TM 153-77
   Provides a nontechnical description of the system for senior management personnel

PRECEDING PAGE BLANK

SECTION 1. GENERAL

1.1 Purpose

This volume of the QUICK Program Maintenance Manual describes the programs which are part of the QUICK Weapon/Allocation subsystem, detailing the programs, subroutines, and functions which it comprises. The information contained herein is presented on a program-by-program basis. The program-by-program discussions are structured so that a maintenance programmer can understand the program functions and programming techniques. The computer subjects are structured to inform the maintenance programmer of overall system programming techniques and conventions.

1.2 General Description

The Weapon Allocation subsystem operates using the integrated data base as defined by all modules up to PLANSET of the Weapon/Target Identification subsystem and produces a plan using the weapon resources specified to maximize the expected target value destroyed. The subsystem consists of programs PREPALOC, ALOC, EVALALOC, and ALOCOUT, as shown in figure 1. Figure 2 shows the relationship of the Weapon Allocation subsystem to other QUICK subsystems in terms of procedural and information flow.

The programs and supporting subroutines of this subsystem are used to define information for use in later processes and allocate given weapons to targets to optimize expected value destroyed. Figure 3 shows the subsystem data flow schematic. The spill tape from program ALOCOUT is the input to program FOOTPRNT which initiates the Sortie Generation subsystem.

The first program, PREPALOC, precomputes much of the information required by later processors. It organizes the input data for efficient use by other components of the Weapon Allocation subsystem. In addition, it provides capabilities for planning factor modification and fixed weapon assignment specification.

The basic data manipulated by this program include the distance and attrition factors for the weapons, the geographic description of the bomber penetration and depenetration corridors, the weapon characteristic tables (e.g., warhead and payload tables), and the target characteristics.

The next program, ALOC, performs the allocation of weapons to targets. Using a generalized Lagrange multiplier method, an optimal allocation is generated subject to several forms of user-input allocation constraints. These constraints include specification of minimum and maximum desired damage levels, restriction of weapons to specified subsets of the target system, and specification of weapons allocated to specific targets by the user. Within these constraints, the program generates the allocation which maximizes the expected value destroyed in

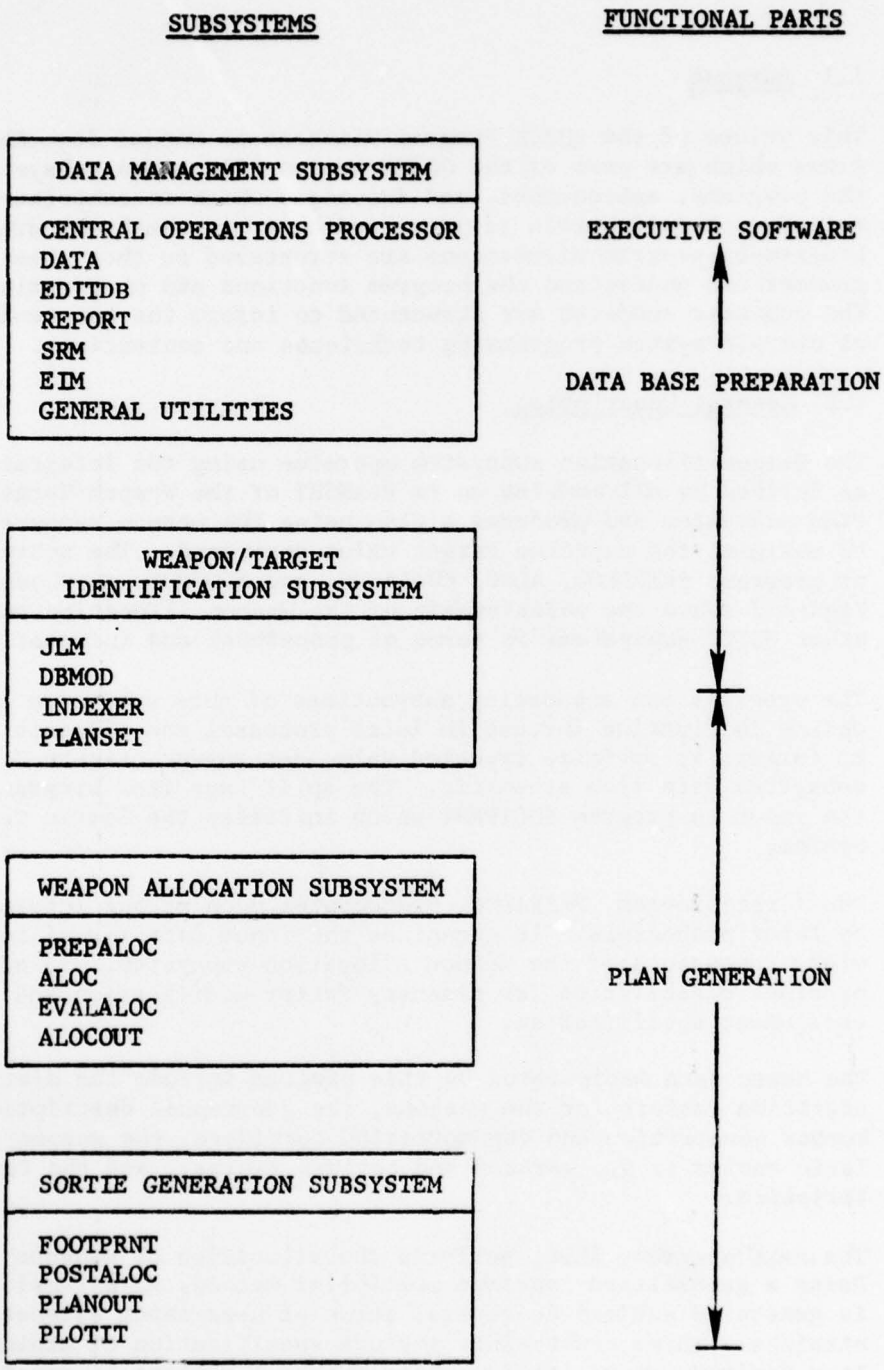


Figure 1. Major Subsystems of the QUICK System

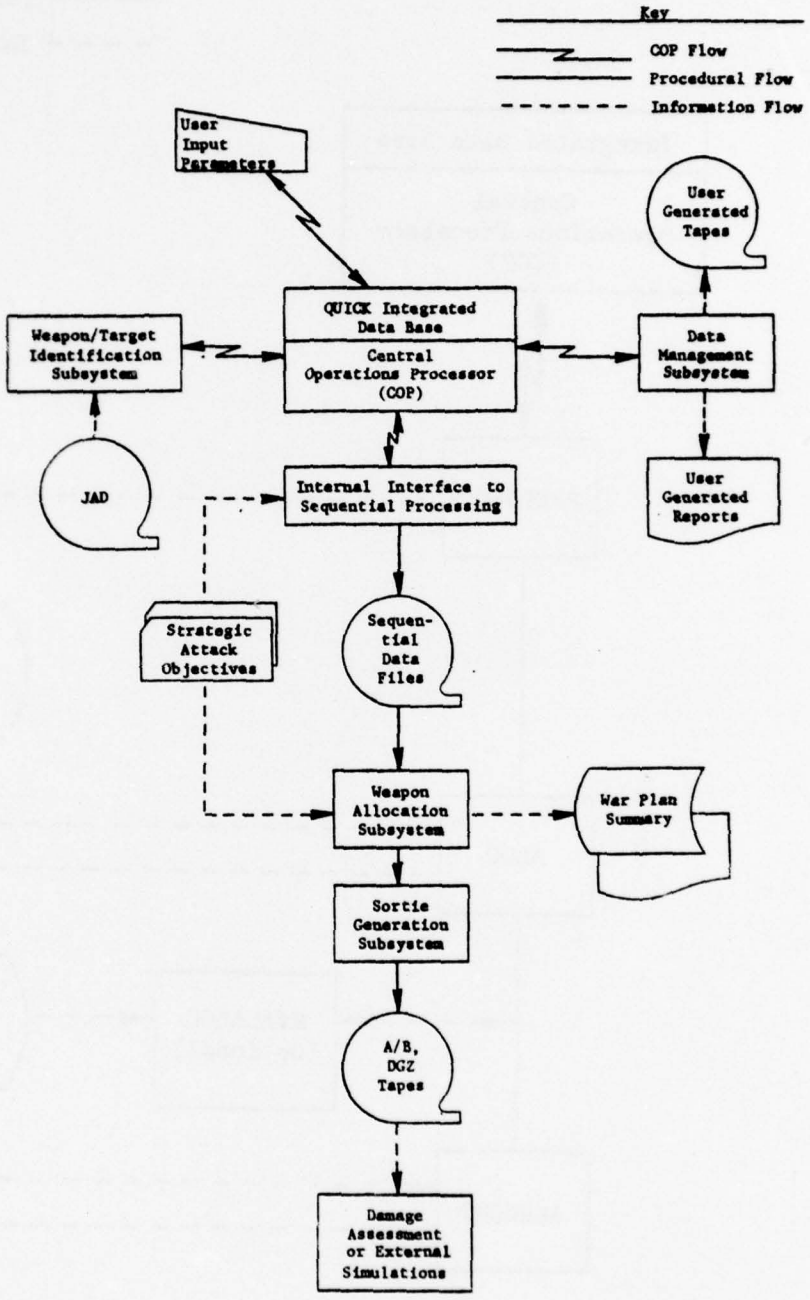


Figure 2. Procedure and Information Flow in QUICK/HIS 6000

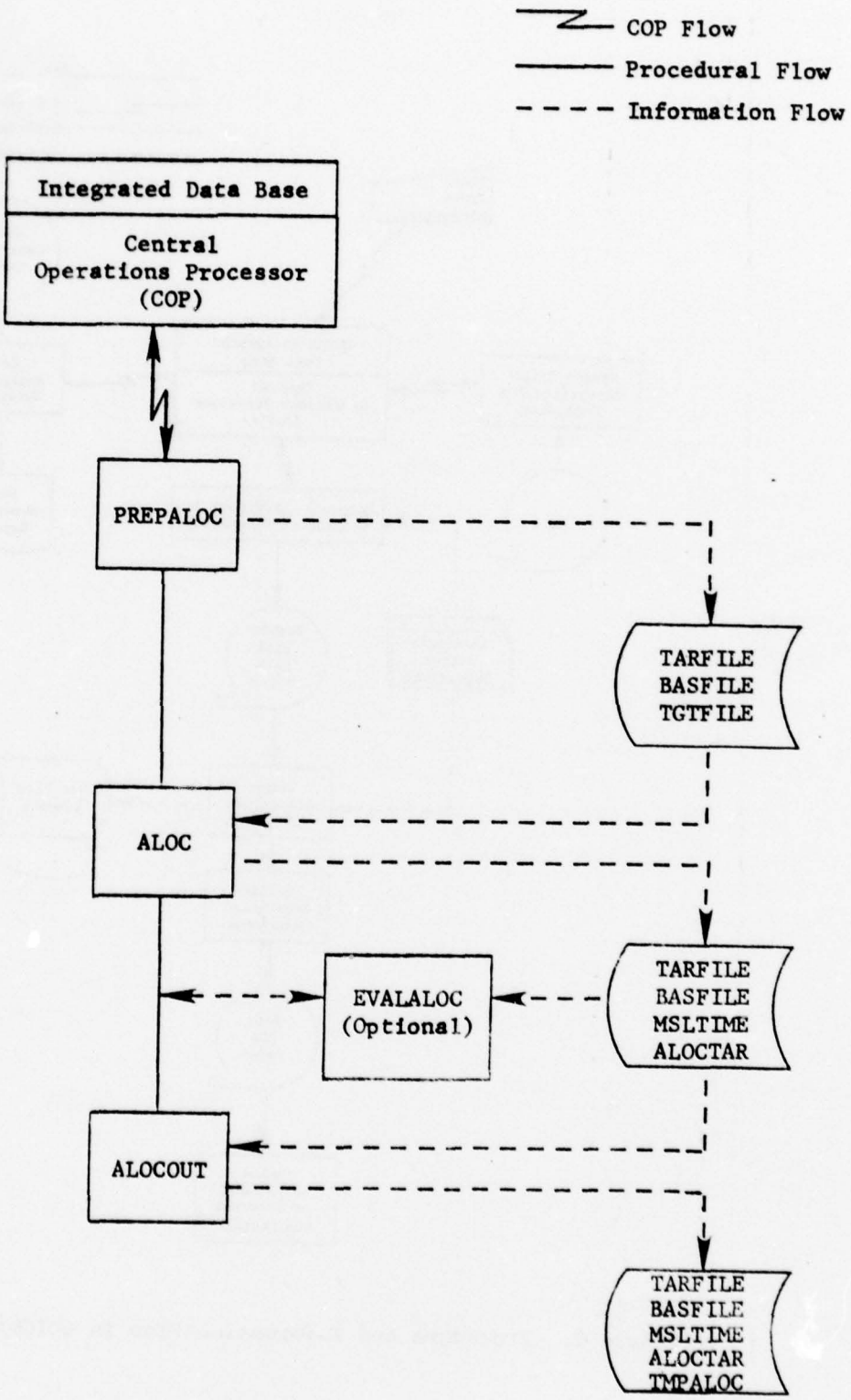


Figure 3. Weapon Allocation Subsystem - Data Flow

SECTION 2. MODULE PREPALOC

2.1 Purpose

The purpose of this module is to perform preliminary calculations on the weapon and target data as stored within the integrated data base. The output data from PREPALOC will be in a form convenient for use by the remaining processors of the plan generator. In addition, the user may select options to modify some of the data at this stage of processing.

PREPALOC (sometimes called the Internal Interface Module) is the last module within the QUICK system which is controlled by the COP. Processing for all remaining processors utilizes standards sequential data files which are controlled by the QUICK filehandler subsystem. Therefore, a major portion of PREPALOC action is the generation of sequential data files. These files obtain information from the integrated data base.

Program PREPALOC has four major capabilities: generation of weapon and target data, modification of target values and damage constraints, preparation of data for the fixed weapon assignment capability of program ALOC, and the generation of filehandler files.

The data produced by the first capability can be divided into three categories: geographic, weapon, and target. Module PREPALOC uses the geographic data to define the legs for each penetration corridor, the location of each refuel area (except for those determined automatically by program PLANOUT), the bases available for recovery from each depenetration corridor. The weapon data characteristics such as speed, range, yield, CEP, and function are aggregated by weapon type. The number of weapons per salvo and maximum salvo number are calculated. Data on payloads, warheads, and air-to-surface missiles (ASMs) are also calculated for use by later processors. The target data are the characteristics which define the target as a candidate for weapon allocation. The geographic location, vulnerability, value, damage constraints, value sensitivity to time, and other target characteristics are computed for each simple, complex, or multiple target. These data will determine the worth of weapon allocations in program ALOC. The only user-input data required for this capability identify the timing of the strike (e.g., initiative, retaliatory).

The second major capability of this program is the modification of the target characteristics, VTO, MINKILL, and MAXKILL. VTO is the value of the target relative to all the others. MINKILL is the minimum fraction of target value that must be destroyed, and MAXKILL is the maximum desired fraction of target value destroyed. Any of these parameters may be changed for any target. The change requests can change these parameters for a single target or for a set of targets. The set of targets for which a change is requested is identified by target class, type, an individual identifier (target designator code (DESIG)) or any combination of these. For complex targets, the class, type, designator code, and

index number of each component will be checked to determine if a target parameter for the complex is to be changed.

In addition, the user can specify the height of burst to be used in any weapon/target combination. The user selects either a ground burst or an air burst at the optimal air burst height. In the absence of any user specification, the most damaging height of burst is used.

The third major capability is the request for allocation of specific weapons to specific targets. This fixing of weapons to targets enables the user to determine part of the weapon allocation while leaving the allocation program free to determine the remaining allocation. In addition, the time of arrival at target or launch salvo number can be fixed for missile weapons. This information will be passed to program PLANOUT which will adjust launch time accordingly. The fixing of weapons remains in effect for the remainder of the plan generation process. Later programs will retain the assignments as best possible. (For example, it is possible to fix a set of weapons from a weapon group with multiple independently targetable reentry vehicles (MIRV) in such a manner that there are no feasible footprints that cover that target set adequately. In that case, some of the fixed assignment requests must be ignored.)

The fourth major capability is the transition from the integrated data base method of data definition to that of sequential record storage where the physical location of each record is necessary for proper processing.

2.2 Input

The entire integrated data base must be completely defined prior to PREPALOC execution. This includes the storage of all targets, related geographic data, weapon type and group characteristics and other supporting data such as warhead and payload information.

2.3 Output Files

There are three files output by PREPALOC, the TARFILE, BASFILE, and TGTFILE (all files are output by the QUICK filehandler onto a specified disk unit). The TGTFILE contains the target characteristics input on the data base as modified by the user-input requests. Appended to these characteristics are geographic data relating each target to each penetration corridor and a preferred depenetration corridor. Table 1 displays the format of the TGTFILE. There is one data block (locally stored by common /TARGET/) as described in table 1 for each target. The BASFILE contains the basic formatted source of geography, warhead, payload, weapon and target component (for complex or multiple targets) information for the remaining processors of the plan generation. Table 2 displays the format of the BASFILE. This file is divided into four major sections:

Table 2. BASFILE Format (Part 1 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
MASTER	22	IHDATE	Date of run initiation
		IDENTNO	Run identification number
		ISIDE	Attacking side
		NRTPT	Number of route points
		NCRNO	Number of penetration corridors
		NDPNO	Number of depenetration corridors
		NRECOVER	Number of recovery bases
		NREF	Number of directed refuel areas
		NREG	Number of command and control regions
		NTYPE	Number of weapon types
		NGROUP	Number of weapon groups
		NTOTBASE	Total number of bases
		NPAYLOAD	Number of payload types
		NASMTYPE	Number of ASM types
		NWHDTYPE	Number of warhead types
		NTANKBAS	Number of tanker bases
		NCOMPLEX	Number of complex targets
		NCLASS	Number of weapon classes (2)
		NUMLRT	Number of alert conditions (2)
		NTGTS	Number of targets
		NCORTYPE	Number of penetration corridor types
		NCNTRY	Number of distinct country codes

Table 2. (Part 2 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTIONS
IIMFILE	4	TGTFILE(2)*	Target data file
		BASFILE(2)	Data base information file
FILES	11	MSLTIME(2)	Fixed missile timing file
		ALOCTAR(2)	Weapon allocation by targets file
		TMPALOC(2)	Temporary allocation file
		ALOCGRP(2)	Allocation by group file
		STRKFIL(2)	Strike file
		PLANTAPE**	Detailed plans tape
CORRCHAR	180	PCLAT(30)	Latitude of corridor point
		PCLONG(30)	Longitude of corridor point
		ZPLAT(30)	Latitude of corridor origin
		ZPLONG(30)	Longitude of corridor origin
		ENTLAT(30)	Latitude of corridor entry
		ENTLONG(30)	Longitude of corridor entry
CRLENGTH	30	CRLENGTH(30)	Distance from corridor entry to corridor origin
KORSTYLE	30	KORSTYLE(30)	Power of y versus x***
PCVALUES	60	ATTRCORR(30)	High altitude attrition per nautical mile unsuppressed
		ATTRSURR(30)	High altitude attrition per nautical mile suppressed

* First word is logical unit number; second word is maximum file length in words. Single variables are logical unit numbers.

** Output on magnetic tape.

***See program POSTALOC.

Table 2. (Part 3 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
CORRTWO	271	HILOATTR(30)	Ratio of low to high altitude attrition (less than 1)
		DEFRANGE(30)	Characteristic range of corridor defense (nautical miles)
		NPRCRDEF(30)	Number of attrition sections this corridor
		DEFDIST(30,3)	Distance of precorridor leg
		ATTRPRE(30,3)	Attrition in this precorridor leg
		NDATA	Number of penetration corridors
ASMONE	100	IWHDASM(20)	Warhead index
		RANGEAAR(20)	Range
		RELASMAR(20)	Reliability
		CEPASMAR(20)	CEP
		SPEEDAAR(20)	Speed
PAYONE	440	NOBOMB1(40)	Number of type 1 bombs
		IWHD1(40)	Type 1 warhead index
		NOBOMB2(40)	Number of type 2 bombs
		IWHD2(40)	Type 2 warhead index
		NASM(40)	Number of ASMs
		IASM(40)	ASM index
		NCMSAR(40)	Number of countermeasures
		NDECOYAR(40)	Number of terminal decoys
		NAREADAR(40)	Number of area decoys
		IMIRV(40)	MIRV system identification number
		PAYALTAR(40)	Release altitude code

for ASMs

Table 2. (Part 4 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
DPENREF	150	DPLINK(50)	Depenetration point link
		DPLAT(50)	Depenetration point latitude
		DPLONG(50)	Depenetration point longitude
RFPOINTS	40	RFLAT(20)	Refuel point latitude
		RFLONG(20)	Refuel point longitude
PLANTYPE*	3	INITSTRK	Indicator for first or second strike
		CORMSL	Coordination time parameter for missiles
		CORBOMB	Coordination distance for bombers
WHONE	150	YLDAR(50)	Weapon yield
		PDUDAR(50)	Weapon dud probability
		FFRACAR(50)	Fission fraction
CCRELAR	20	CCRELAR(20)	Command and control reliability
WEPVAL	200	RANGEAR(100)	Range of vehicle (nautical miles)
		CEPAR(100)	CEP (nautical miles)
WPVALUES	400	SPEEDAR(100)**	Speed (knots)
		ALERDAR(100)	Alert delay (hours)
		NALRTDAR(100)	Nonalert delay (hours)
		RANGEDAR(100)***	Low/high altitude fuel consumption ratio

* Blocks PLANTYPE and EXCESS combined in program PREPALOC to common /GAMEVAR/.

**For missiles, this variable is replaced by TOFMIN, the minimum flight time (hours).

***For missiles, this variable is replaced by CMISS, the missile time of flight factor.

Table 2. (Part 5 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
ICLASSAR	100	ICLASS(100)	Weapon class
RANGERAR	100	RANGERAR(100)*	Reliability
WEPTWO	300	RELAR(100)	Reliability
		IRECMODE(100)	Recovery mode
		IPENMODE(100)	Penetration mode
ISIMTYPE	100	ISIMTYPE(100)	Hollerith type name
FUNCTION	100	FUNCTION(100)	Function code
LHVALUES	150	LCHINTAR(75)	Launch interval for missiles types
		SIMLUNAR(75)	Number of simultaneous launch for missiles types
WPNGRP	3250	NWPNS(250)	Number of weapons
		NVEHGRP(250)	Number of vehicles
		WLAT(250)	Centroid latitude
		WLONG(250)	Centroid longitude
		IREGAR(250)	Command and control region
		IALERTAR(250)	Alert status
		SBL(250)	Probability of survival before launch
		IREFUEAR(250)	Refuel code for bombers payload index for missiles
		YIELDAR(250)	Weapon yield (negatons)
		REFTIME(250)	Refuel time
		EXPASM(250)	Franction of ASMs in each group
		IAMSM(250)	ASM index (pointer to ASM Table)

*For missiles, this variable is replaced by RNGMIN, the minimum missile range in nautical miles.

Table 2. (Part 6 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
--	1	NEXCESS	Number of words in this block
EXCESS*	6	PEXBOMB	Percentage of weapons added to bomber groups in PREPALOC
		EXNBOMB	Number of vehicle loads added to bomber groups in PREPALOC
		PEXMIRV	Same as PEXBOMB for groups with MIRV capability
		EXBMIRV	Same as EXNBOMB for groups with MIRV capability
		PEXMISS	Same as PEXBOMB for non-MIRV missile groups
		EXNMISS	Same as EXNBOMB for non-MIRV missile groups
SBLREAL	250	SBLREAL(250)	Actual SBL probability
--	1	NNAVAL	Number of words in this block
PKNAVAR	250	PKNAVAR(250)	Single shot kill probability against NAVAL targets
CTRYCD**	150	CTRYCD(150)	List of country location codes
NFIXES	250	NFIXES(250)	Number of weapons fixed in each group

*Blocks PLANTYPE and EXCESS combined in program PREPALOC to common /GAMEVAR/.

**Blocks CTRYCD and IHVULN combined in program PREPALOC to common /CVTAB/.

Table 2. (Part 7 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
SALVO	1750	MAXSLV(250)	Maximum salvo number in salvoed group (zero otherwise)
		NSAL(3,250)	Number of weapons in each salvo (Packed 4 bits per salvo, 24 salvoes per group)
		NSFIX(3,250)	Number of weapons fixed in each salvo for salvoed groups (packed same as NSAL)
HOB	6	ISPEC(3)	Logical array set true only if user specified HOB for a weapon type
		IWHOB(3)	Logical array giving user specified HOB by weapon type (true=air; false=ground)
IHVULN*	255	IHVULN(255)	Vulnerability numbers present in target set
--	1	RRRRRR	=6HRRRRRR as end sentinel for first section
--	(34 x number of complex target components) + (8 x number of multiple target components)	NAMEZ** INDEXNOZ DESIGZ TASKZ CNTRYLZ FLAGZ TGTSTATZ LATZ LONGZ RADIUSZ VALZ NHRDCOZ	34-word record for each complex target component; variables defined as on TGTFILE

* Blocks CTRYCD and IHVULN combined in program PREPALOC to common /CVTAB/.
**In order as complex or multiple target appears on target file, TGTFILE.

Table 2. (Part 8 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
-- (cont.)		H1Z	
		H2Z	
		FVULNZ	
		NTIMCOZ	
		FVALTZ(5)	
		TZ(5)	
		IHCLASZ	
		ICLASSZ	
		IHTYPEZ	
		TARDEFZ	
		MISDEFZ	
		MINKILLZ	
		MAXKILLZ	
		MAXFRAZ	
		NAMEZ	8-word record for each multiple target element; variables defined as on TGTFILE
		INDEXNOZ	
		DESIGZ	
		TASKZ	
		CNTRYLZ	
		FLAGZ	
		LATZ	
		LONGZ	
		ZZZZZZ	=6HZZZZZZ end sentinel for secont section
(one block for each weapon group)	(10 x number of groups) + (5 x number of occurrences of bases) + (18 x number of groups)	IGROUP	Group number
		NWPNS	Number of weapons
		NVEHGRP	Number of vehicles
		IREGTM	Command and control region
		ITYPE	Weapon type index (ITYPE)
		IALERTTM	Alert status
		IREFUETM	Refuel code
		YIELDTM	Yield
		ISTARTTM	Starting weapon index
		NBASETM	Number of bases
		IBASE	Base index number

Table 2. (Part 9 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
-- (cont.)	One five-word Block for each Base (i.e., NBASE)	BLAT	Base latitude
		BLONG	Base longitude
		IPAYLOAD	Payload index
		VONBASE	Number of base and index of starting vehicle
	One 18-word Block for each Group	WPTYP	Hollerith type name
		RANGETM	Range
		CEPTM	CEP
		SPEEDTM*	Speed
		ALRTTM	Alert delay
		NLRTTM	Nonalert delay
		RNGDTM**	Low/high altitude fuel consumption ratio
		ICLSTM	Weapon class number
		NPSQTM	Number per squadron
		LCHTM	Launch interval time (hours)
		SPDTM	Low altitude speed
		SIMTM	Number of simultaneous weapon launches per site
		RNGRTM***	Refueld range
		NMPSTM	Number per site
		IREPTM	Reprogramming index
		IRECTM	Recovery mode
		IPENTM	Penetration mode
		FUNCTM	Function code
--	1	YYYYYY	=6HYYYYY end sentinel of third section

* For missiles, this variable is replaced by TOFMIN, the minimum flight time (hours)

** For missiles, this variable is replaced by CMISS, the missile time of flight factor.

***For missiles, this variable is replaced by RNGMIN, the minimum missile range (nautical miles).

Table 2. (Part 10 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
CHARTER	160	COUNT(30)*	Route pont index associated with IHAP
		IHAP(30)	Corridor pointer: /CORRCHAR/ to /HAPPEN/
		MOUNT(50)	Route point index associated with JHAP
		JHAP(50)	Pointer: /DPENREF/ to /HAPPEN/
HAPPEN	1000	JAPTYPE(250)	Go low indicator
		HAPLAT(250)	Dogleg latitude
		HAPLONG(250)	Dogleg longitude
		HAPDIST(250)	Length of dogleg
RECOVR	1100	RCBLT(50)	First recovery base latitude*
		RCBLN(50)	First recovery base longitude*
		RCBLAT(50,4)	Latitude of actual recovery base**
		RCBLONG(50,4)	Longitude of actual recovery base
		INDBAS(50,4)	Recovery base name
		INDCAP(50,4)	Recovery base capacity
		DISTR(50,4)	Distance from depenetration point to recovery base
--	(12 x number of tanker bases)***	INDEXTK	Tanker base index number
		LATTK	Tanker base latitude
		LONGTK	Tanker base longitude
		IREGTK	Tanker base refuel area
		NPSQDTK	Number of tankers per base or squadron
		NALRTK	Number of alert tankers per squadron

* Indexed by depenetration corridor.
** Indexed by depenetration corridor and base order.
***There is one of thse 12-word blocks for each tanker base.

Table 2. (Part 11 of 11)

ASSOCIATED COMMON	LENGTH	VARIABLE OR ARRAY	DESCRIPTION
RECOVR (cont.)			
--		SPEEDTK	Tanker Speed
		ALRTTK	Alert delay
		NLRTTK	Nonalert delay
		TTOSTK	Total time on station
		ITYPETK	Tanker type index
		RANGETK	Tanker range
--	1	XXXXXX	=6HXXXXXX end sentinel for fourth section

basic data base information, target component information, detailed weapon group information, and a section containing geographic data, recovery base data, and tanker base data.

The first section on the BASFILE contains basic data base information. The first block of data in this section is common /MASTER/ which specifies the number of various entities (targets, groups, etc.) in the data base for the side whose plan is being generated. The next blocks, /IIMFILE/ and /FILES/, specify the logical unit number (for use by the filehandler) and the maximum file length for each nonscratch file used by the plan generation subsystem from programs ALOC to PLANOUT. The next blocks, commons /CORRCHAR/, /CRLENGTH/, /KORSTYLE/, /PCVALUES/, and /CORRTWO/, define the penetration corridor characteristics. Block nine, /ASMONE/, describes the characteristics of the air-to-surface missiles (ASMs). The next block, /PAYONE/, describes the contents of each weapon vehicle payload. Blocks /DPENREF/ and /RFPOINTS/ next describe the locations of the depenetration and refueling points, respectively. The next block, common /PLANTYPE/, specifies the timing considerations for the plan. Block /WHONE/ describes warhead characteristics; block /CCRELAR/ specifies the command and control reliability for each command and control region. Blocks /WEPVAL/, /WPVALUES/, /ICLASSAR/, /RANGERAR/, /WEPTWO/, /ISIMTYPE/, /FUNCTION/, and /LHVALUES/ specify the weapon type characteristics. Block /WPNGRP/ specifies the weapon group characteristics. Blocks /EXCESS/ and /SBLREAL/ contain information on the number of weapons added by program PREPALOC for allocation purposes. (These excess weapons are removed by either program POSTALOC or program FOOTPRNT, in the case of MIRV weapons.) Block /PKNAVAR/ describes the kill probability against targets in the NAVAL class. Finally, block /CTRYCD/ lists all the country location codes (attribute CNTRYLOC present in the target system for this plan). Blocks /NFIXES/ and /SALVO/ define the number of weapons fixed in each group followed by the maximum salvo number, the number of weapons in each salvo, and the number of weapons fixed in each salvo for each group. The salvo information arrays are followed by block /HOB/ which contains the user specified burst heights by weapon type, and block /IHVULN/ which contains the vulnerability numbers (attribute VULNI) for all the targets in the plan.

The second BASFILE section describes the characteristics of each component of a complex or multiple target. The data blocks in this section are ordered as they appear on the TGTFILE. That is, the data for complex and multiple target components are shuffled in the same manner as the target complexes or multiples are shuffled. For each complex, there is a set of 34-word data blocks, one block for each component of the complex, including the "lead" element.

For each multiple target, there is a set of eight-word data blocks, one block for each multiple target component. These eight words identify the values of those characteristics that vary from component to component in a multiple target.

The third BASFILE section consists of detailed weapon group data. This section consists of a set of data blocks, one block for each weapon group. Each of these blocks describes the weapon group characteristics. Each block contains a set of five-word blocks that describe the location and characteristics of each base that is used by weapons in the group. There is one five-word block for each base used by the group. Each block also contains an 18-word block for each weapon group describing the characteristics of its weapon type.

The fourth and final BASFILE section describes miscellaneous data required by later processors. Common /CHARTER/ defines various route point indices and pointers. Common /HAPPEN/ follows with a description of each leg of the penetration and depenetration corridors. Data on location and capacity of all bomber recovery bases follows in block /RECOVR/. Finally, the BASFILE is terminated by the characteristics of the tanker vehicles on each tanker base.

The TARFILE, which is used by program PLANOUT, format is shown in table 3. This file contains packed target information and pointers that define the location of the information based on the target DESIG.

2.4 Concept of Operation

The flow of execution is mainly dictated by the format of the BASFILE (see table 2) whose generation is the main objective of PREPALOC. Since the integrated data base may be accessed in numerous fashions, the writing of BASFILE simply involves the querying of data base chains at the appropriate time. There is little or no need for utilizing temporary scratch data files.

In addition to the BASFILE, files TARFILE and TGTFILE are generated. Also, prior to any file definition, user requests are honored that may redefine data base target(s) data.

In the interest of computer core efficiency, PREPALOC is subdivided in five separate overlay links (called links A, B, C, D, and E) which are executed sequentially. Upon processing an overlay link, that link is never referred to again. The first link (A) generates the TARFILE, and remaining links produce both the BASFILE and TGTFILE.

2.5 Identification of Subroutine Functions

2.5.1 Subroutine TARLST. This subroutine controls the flow of the first overlay link (called A) which generates the TARFILE.

2.5.3 Subroutine PARTA. The second overlay link (called B) is controlled by this subroutine. Weapon type data is stored, user input data read and processed through supporting subroutines FACTORCG and FIXWEP and the first section of the BASFILE is partially written.

Table 3. TARFILE Format

BLOCK TYPE	MAXIMUM LENGTH/ ACTUAL LENGTH	VARIABLE OR ARRAY	DESCRIPTION
Header /C5/	1 (each) 3 total	MXSCAT	Maximum length of the SCAT array
		MAXPPG	Maximum length of a TARLIST page
		NIOPGS	Number of TARLST pages on the TARFILE
Hash Table /C5/	7771	SCAT(7771)	Target designator "hash" ordered "page" and string pointer table
TARLIST Pages /C5/	Variable in 3516-word segments	TARLIST(3516)	TARLIST page of target designator strings

2.5.3 Subroutine WEPPREP. The third overlay link (called C) is controlled by this subroutine. It continues the generation of the first section of the BASFILE.

2.5.4 Subroutine TGTPREP. The fourth overlay link (called D) is controlled by this subroutine. It normalizes target values, adjusts salvo launch information and writes the TGTFILE file as well as completing the first section of the BASFILE.

2.5.5 Subroutine PARTB. The last overlay link (called E) is controlled by this subroutine which writes the last three sections of the BASFILE.

2.5.6 PREPALOC Internal Common Blocks. All common blocks used internally by PREPALOC are given in table 4. For definition of common blocks that communicate with the COP, see Program Maintenance Manual, Volume I. Also, common blocks used directly for BASFILE definitions are defined in table 2 and not repeated within table 4.

Table 4. Module PREPALOC Internal Common Blocks
(Part 1 of 2)

ASSOCIATED COMMON	VARIABLE OR ARRAY	DESCRIPTION
ASMTYPE	ASMTYPE(20)	ASM type names
C5	TARLIST(3516)	Page of target designator strings for TARFILE
	SRCPAD(3516)	Scratch pad of DESIG strings and items
	SCAT(7771)	Hash ordered array of page and target designator string pointers
	MAXPPG	Maximum number of words in a TARLIST page (3516)
	MXSCAT	Maximum number of entries in the SCAT array (7771)
	MAXSRC	Maximum number of words in the SRCPAD array (3516)
	LAVSPG(15)	Pointer to lists of available space for TARLST pages 1 through 15
	MXPGS	Maximum number of TARLST pages (15)
	INCR	Number of words in target designator string item (6)
CLASSCOM	CLASSNAM(15)	Target class names
	CLASSREF(15)	Header reference codes for target classes
	NTARCLS	Number of target classes
DISTEF	DISTEF(50)	Length of depenetration corridor
	DISTEG(50)	Distance from depenetration corridor entry to recovery point
GAMFLAG	GAMFLAG(9)	Flag indicating value for planning parameter was input if set. Planning parameters are: INITSTRK, CORMSL, CORBOMB, PEXBOMB, EXNBOMB, PEXMIRV, EXBMIRV, PEXMISS, EXNMISS

Table 4. (Part 2 of 2)

ASSOCIATED COMMON	VARIABLE OR ARRAY	DESCRIPTION
IGPREF	IGPREF(250)	Reference codes for weapon groups
IONPRT	IONPRT	If zero, suppress non-standard print
ITP	ITP	Requested unit number for the Filehandler
IWEPREF	IWEPREF(100)	Reference codes for weapon type records
MYIDENT	MYIDENT	File name for the Filehandler
MYLABEL	MYFORM	Outgoing format
	MYSECR	Outgoing security
	MYLNGTH	Number of words requested for file
	MYCOMM(5)	Outgoing user comments
NFIXREQ	NFIXREQ	Number of fix assignments implemented
NUMCOR	NCORR	Number of penetration corridors
	NDPEN	Number of depenetration corridors
PAYTYPE	PAYTYPE(40)	Payload type names
SUMNEW	SUMNEW	Sum of values after implementing value change requests
TWORD	ITWORD	Single word transfer medium for Filehandler
WAROUT	IWARFL	Logical unit number for war gaming output
WHTYPE	WHTYPE(50)	Warhead type names

2.7 Subroutine ENTMOD

PURPOSE: To control overall flow of processing

ENTRY POINTS: ENTMOD (first subroutine called when overlay link PREP is executed)

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C20, C25, C30, CLASSCOM, ERRCOM, ITP, MYIDENT, MYLABEL, TWORD, WAROUT

SUBROUTINES CALLED: HDFND, HEAD, INITAPE, LLINK, NEXTTT, PARTA, PARTB, RETRV, TARLST, TGTPREP, WEPPREP

CALLED BY: COP

Method:

ENTMOD retrieves and stores target class names and associated reference codes into arrays CLASSNAM and CLASSREF. Upon storage completion, each overlay (see figure 4) is executed and then processing ends for PREPALOC.

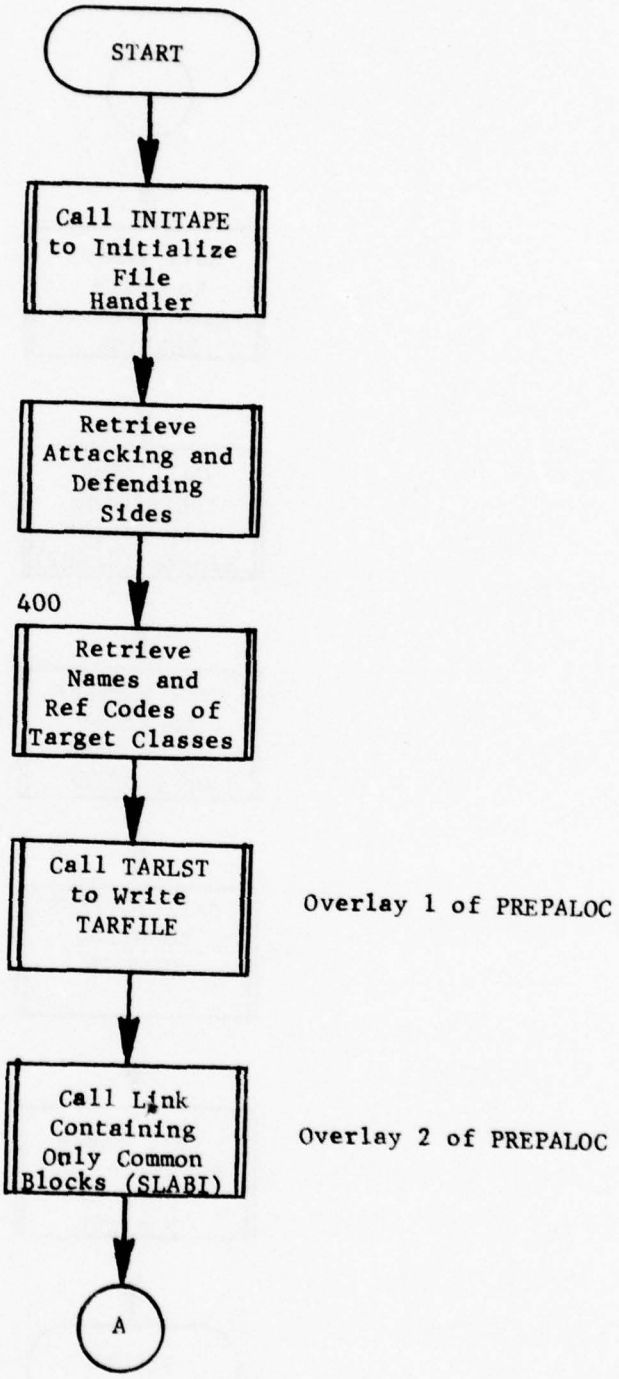


Figure 4. Module PREPALOC (Part 1 of 2)

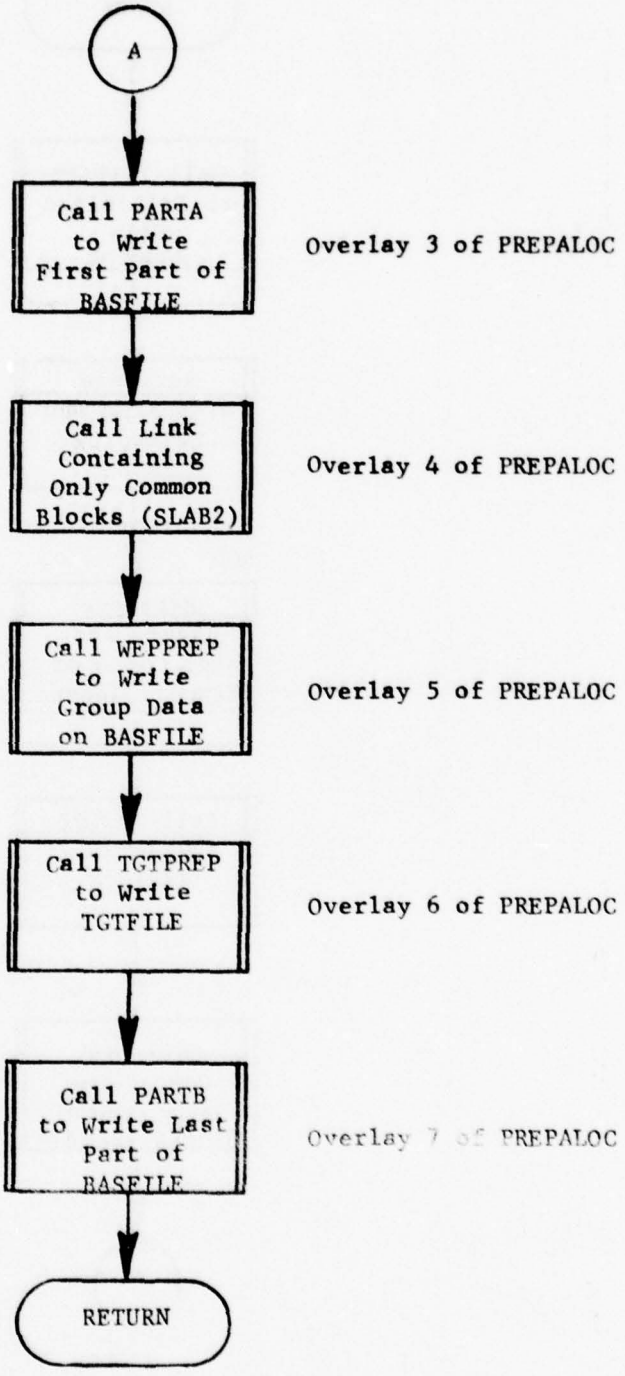


Figure 4. (Part 2 of 2)

2.8 Subroutine TARLST*

PURPOSE: To write the TARFILE

ENTRY POINTS: TARLST

FORMAL PARAMETERS: None

COMMON BLOCKS: C5, C10, C15, C30, CLASSCOM, ITP, MYIDENT, MYLABEL, TWORD

SUBROUTINES CALLED: ABORT, DIRECT, HEAD, IHASH, INITAR, INITPG, NEXTTT, RDARRAY, RETRV, SETREAD, SETWRITE, TERMTAPE, WRARRAY, WRWORD

CALLED BY: ENTMOD (of PREPALOC)

Method:

Overlay TARLST is called to create the TARFILE. The flowchart for TARLST is shown in figure 5. TARLST reads one item at a time from the individual target record chain called 'TGTTGT' and uses the target designator code (DESIG) to calculate a storage address in the hash ordered array SCAT. If the SCAT array entry at the calculated address is empty, the current TARLIST page number (KCPG) and the pointer index (INDX) to the storage area in the current page (CURPGE) are packed into the SCAT array entry. The item information LAT, LONG, DESIG, ISIX, FLAG, TARDEFHI, CNTRYLOC, TASK, CNTRYOWN, H1, and INDEXNO are packed and stored in six consecutive words of CURPGE beginning at the address INDX.** The target designator string pointer (ISIX) is set to zero to indicate that this is the first item in the DESIG string. If a subsequent item DESIG calculates the same hashed SCAT address (and the indicated page (IPG) for the previous DESIG(s) is currently in core) the pointer in the SCAT entry (INDX) is changed to point to the new item and the new item string pointer (ISIX) is set to point to the predecessor item. If the indicated page (IPG) is not currently in core the item information is stored in a "scratch pad" in the same packed format as that of the TARLIST entries. The string pointers (ISIX) for these scratch pad items are set to '9999' as a flag for later reprocessing of the TARLIST pages. As pages of the TARLIST are filled in this manner they are output on the intermediate scratch file (IOUT). If a string of DESIGs were to overflow a page as it is filled, the string is "pulled" and stored in the SRCPAD. The pointer in the SCAT entry is changed to point to the SRCPAD string and the SCAT entry is set negative to indicate that the string is in the SRCPAD. When the SRCPAD is 3/4 full, the intermediate scratch files IN and OUT are interchanged and the pages previously output in IOUT are reprocessed

* Main subroutine of overlay link A.
**See figure 5 for a description of the packing of this data.

on IN. The reprocessing is accomplished by scanning the SRCPAD for '9999' flagged item entries. If the SCAT entry IPG is the same as that of the current page (input from IN) the string is pulled and stored in the SRCPAD in the same manner as described for overflow strings (negative SCAT entry). All pages are reprocessed in this manner until the SRCPAD is full or all strings are pulled. The SCAT array is then scanned for negative entries. The strings for these negative entries are then moved from the SRCPAD to the current page. This process continues until all SCAT entries are positive. Processing of the INDEXDB tapes then continues until all items have been processed.

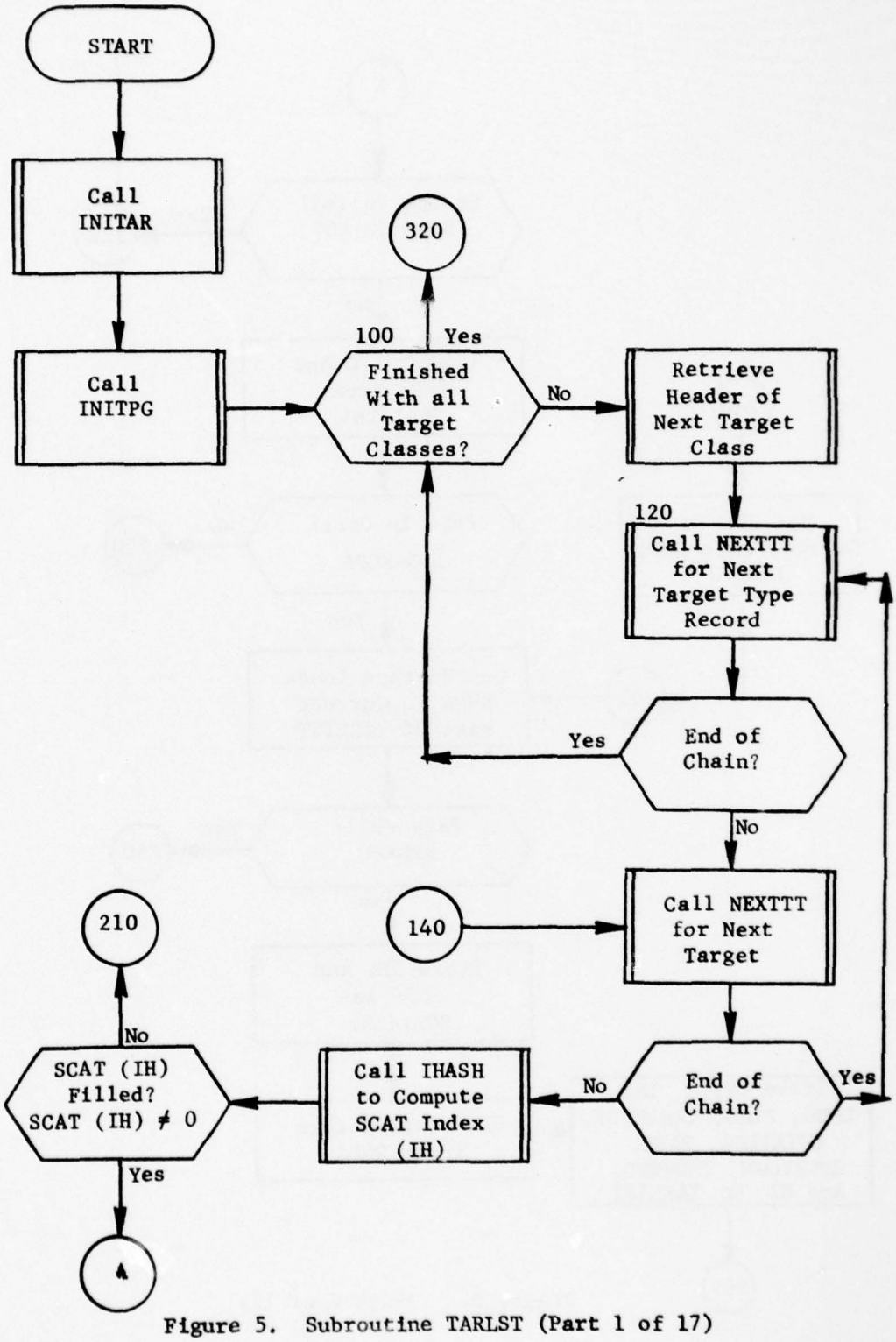


Figure 5. Subroutine TARLST (Part 1 of 17)

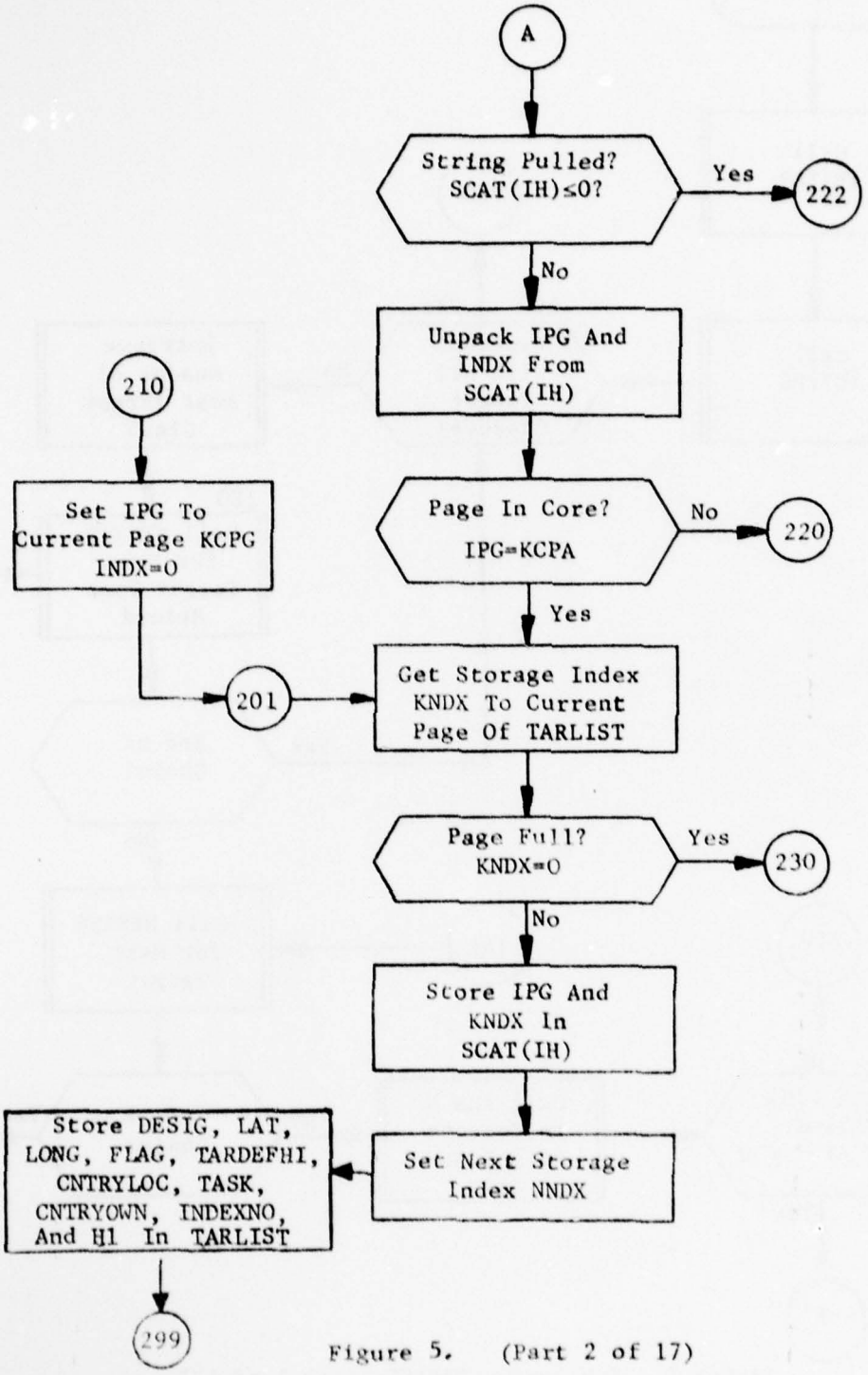


Figure 5. (Part 2 of 17)

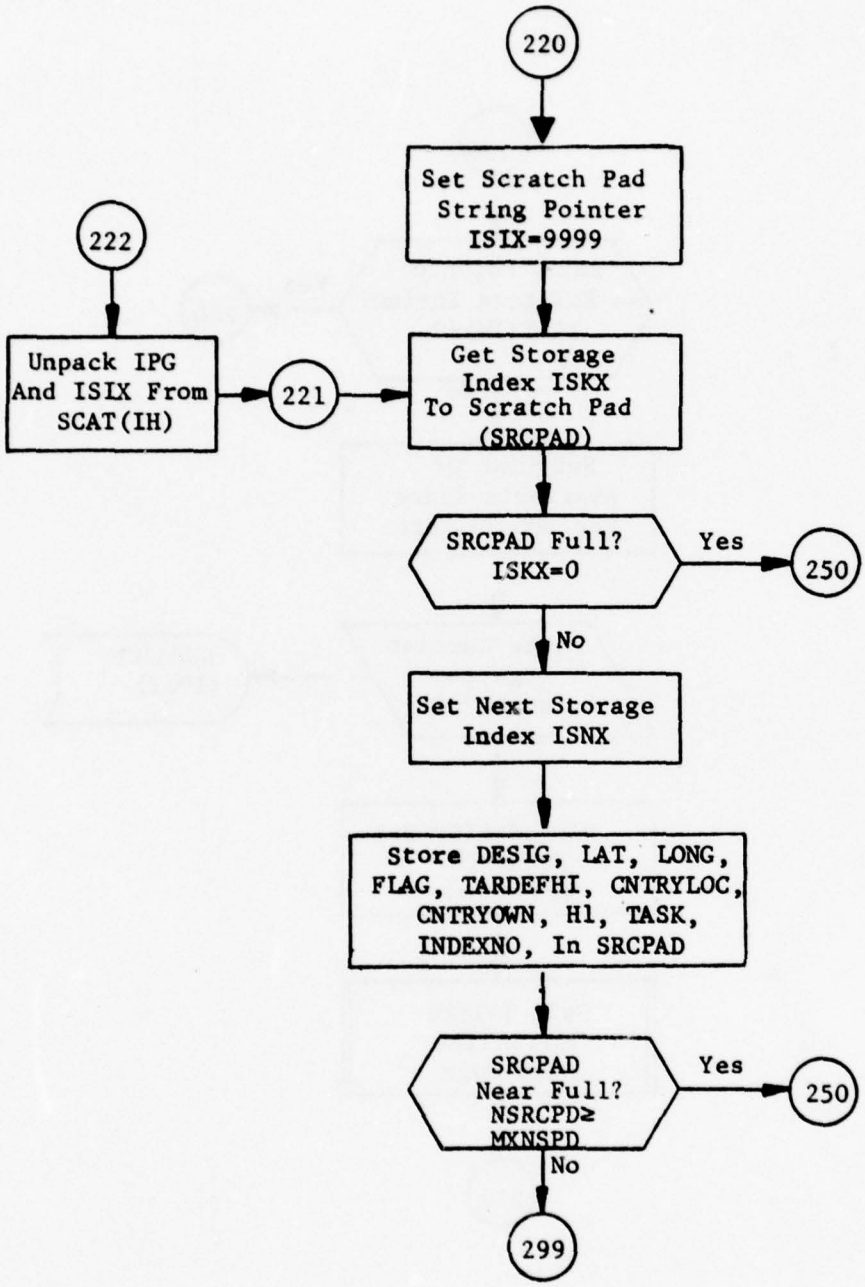


Figure 5. (Part 3 of 17)

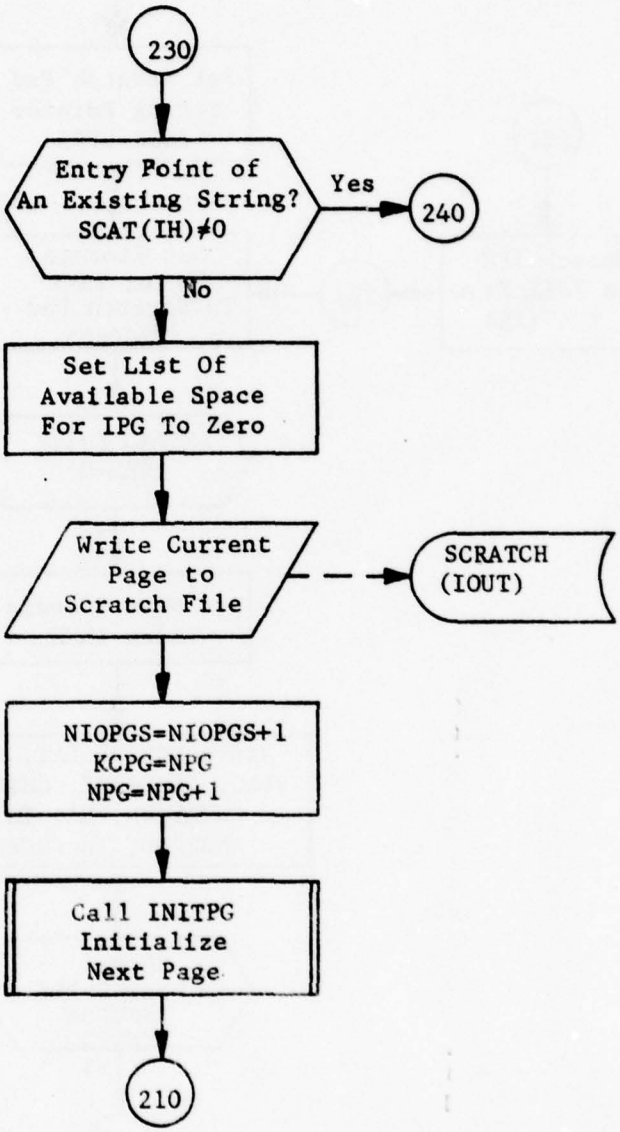


Figure 5. (Part 4 of 17)

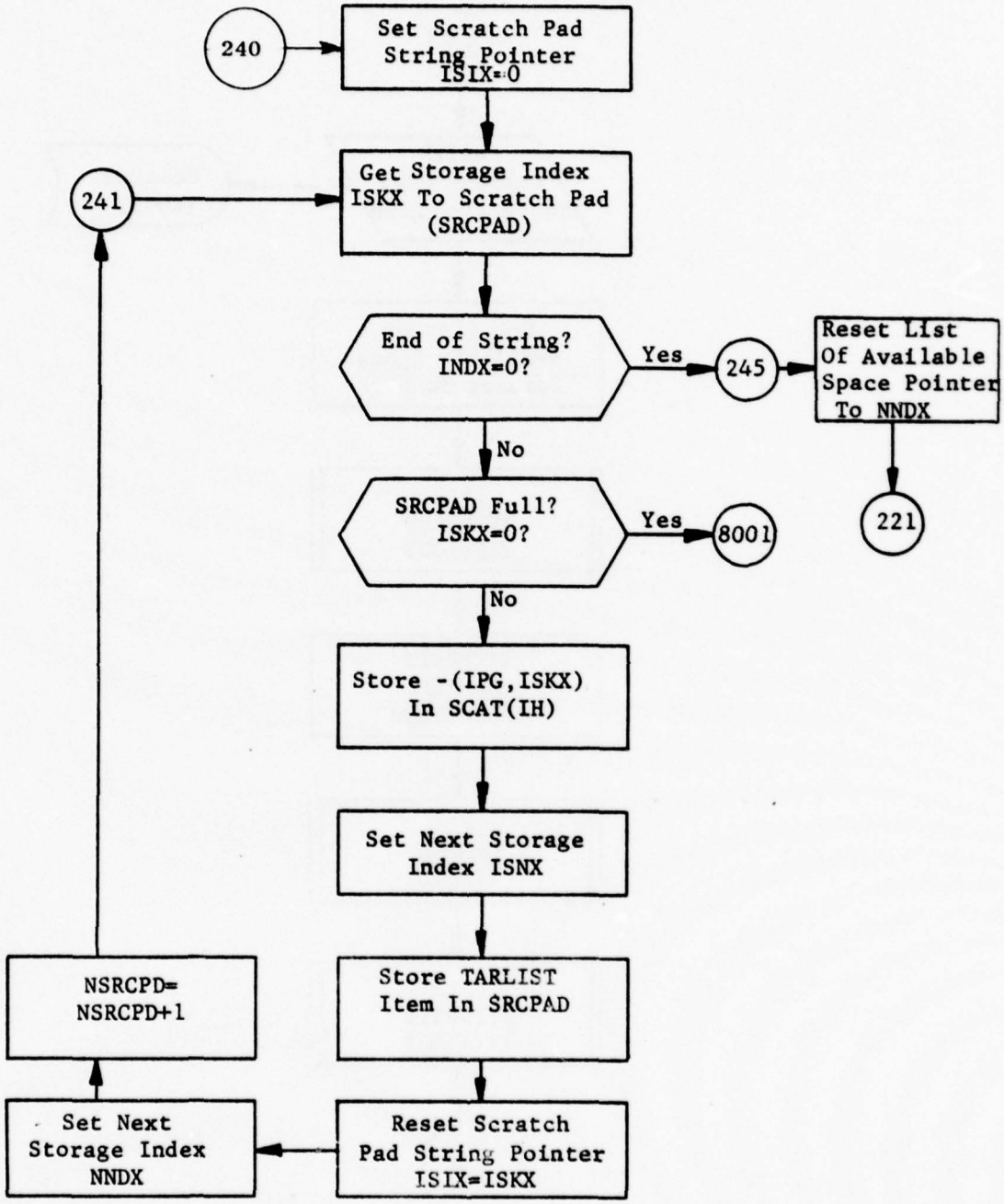


Figure 5. (Part 5 of 17)

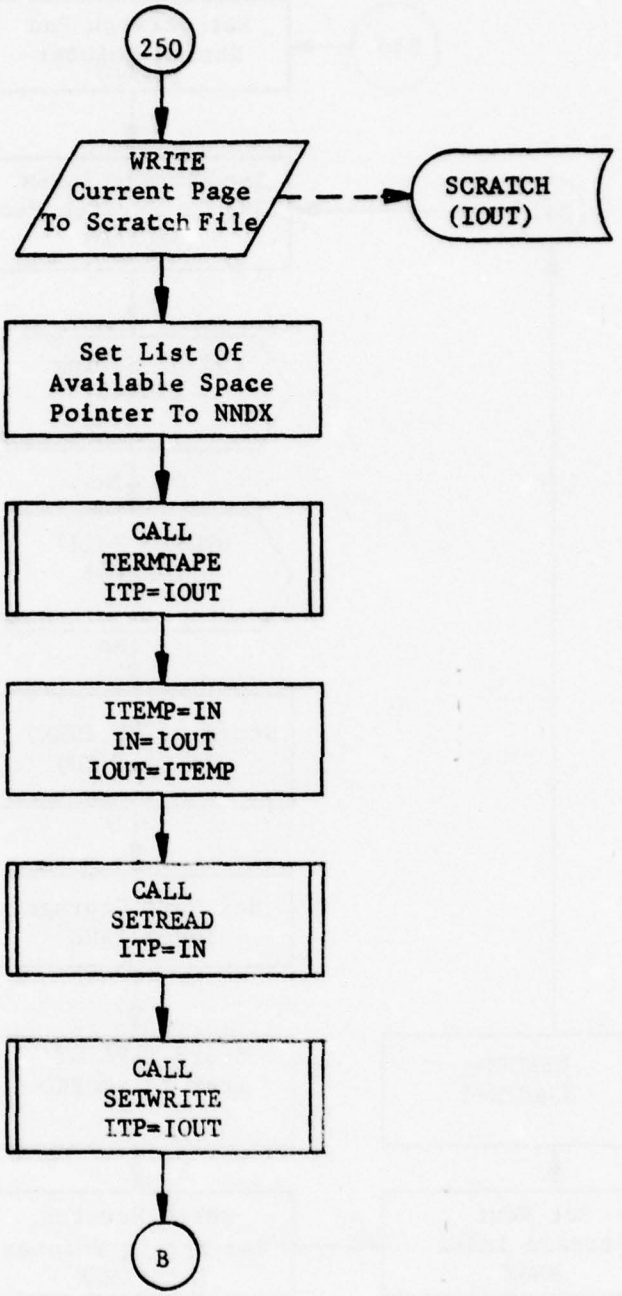


Figure 5. (Part 6 of 17)

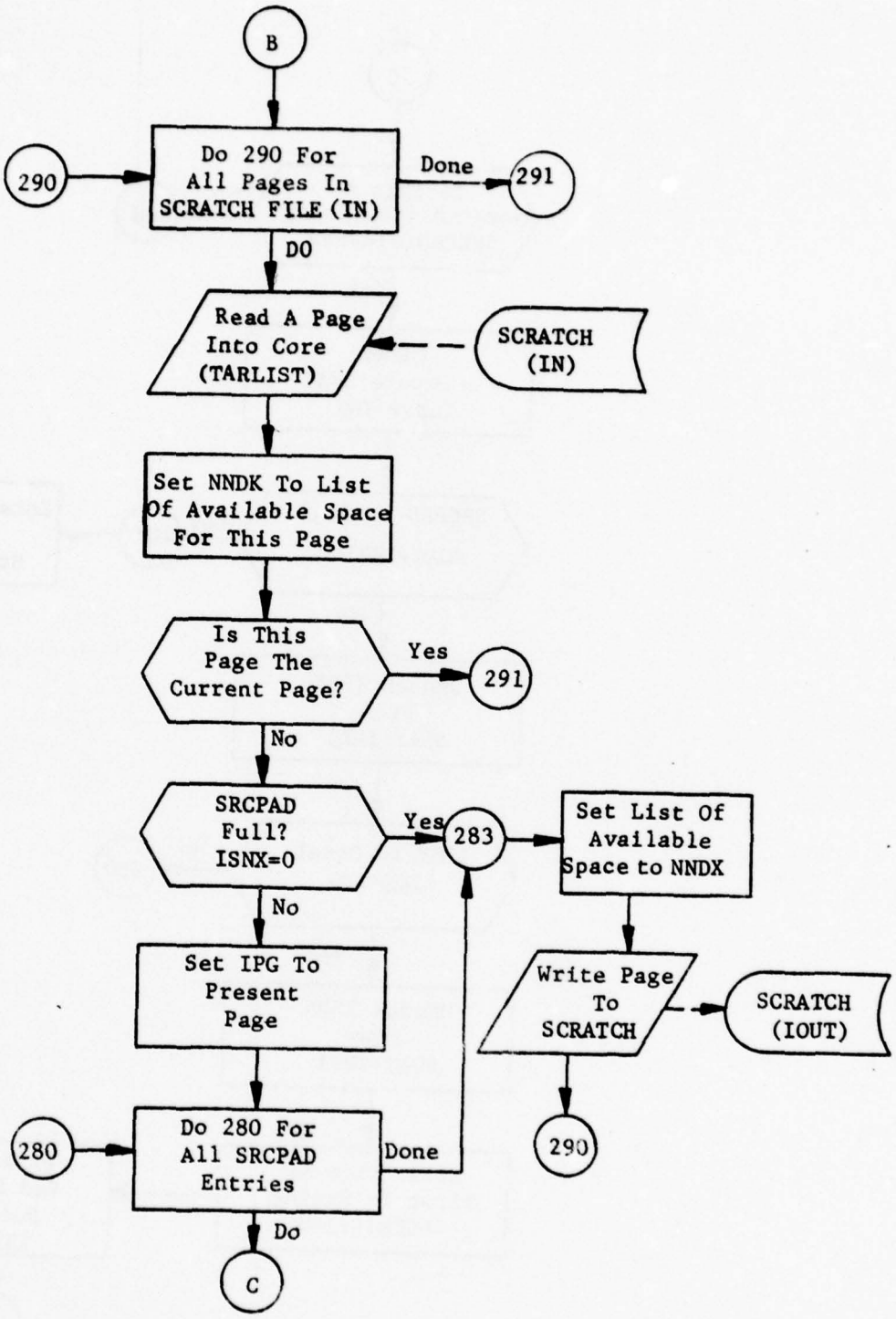


Figure 5. (Part 7 of 17)

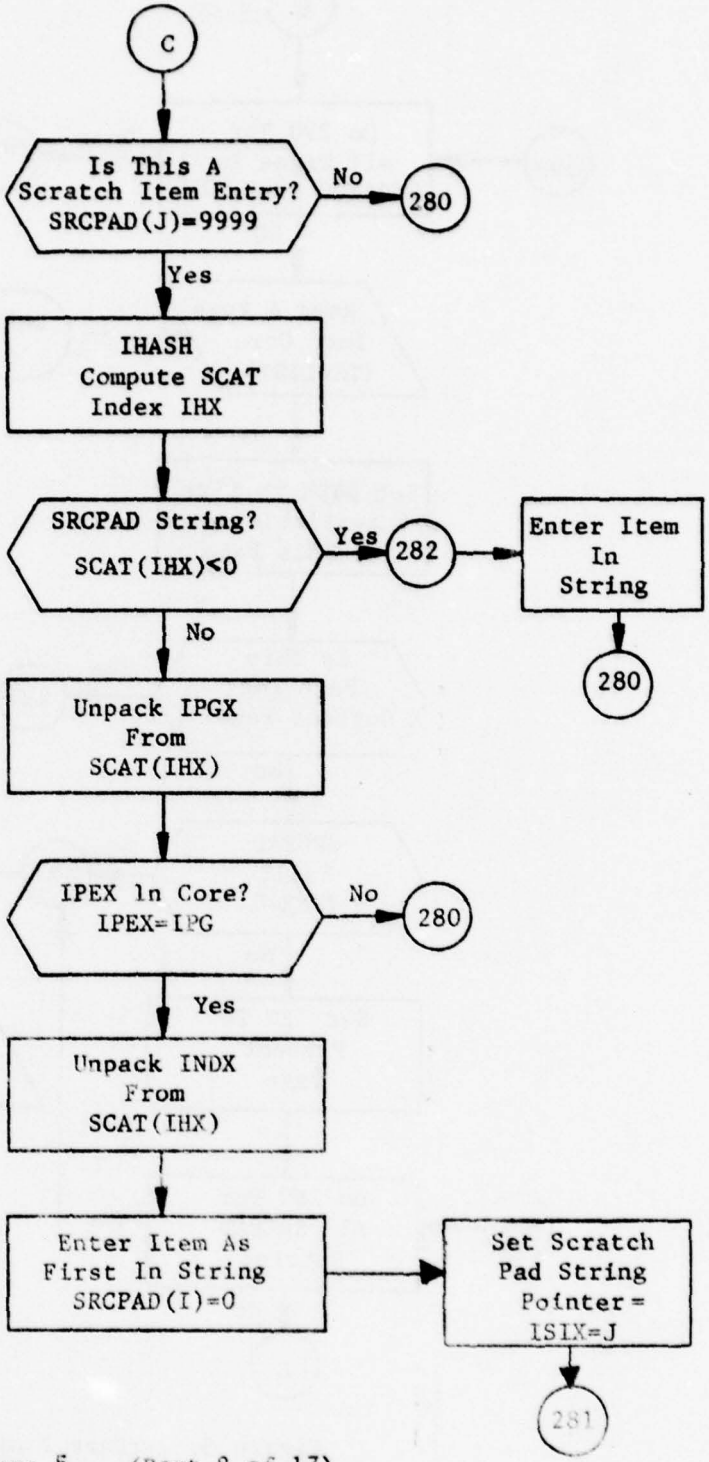


Figure 5. (Part 8 of 17)

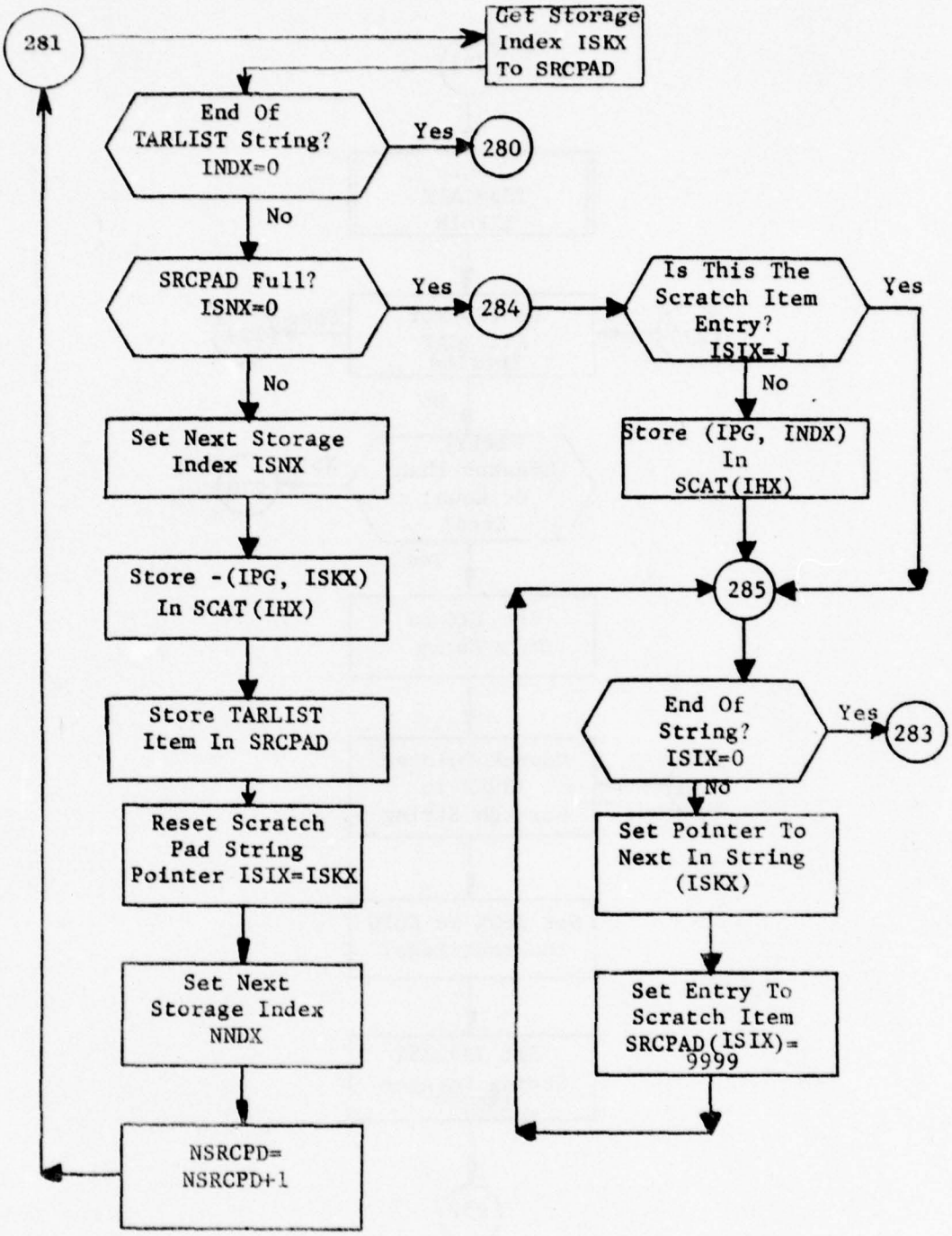


Figure 5. (Part 9 of 17)

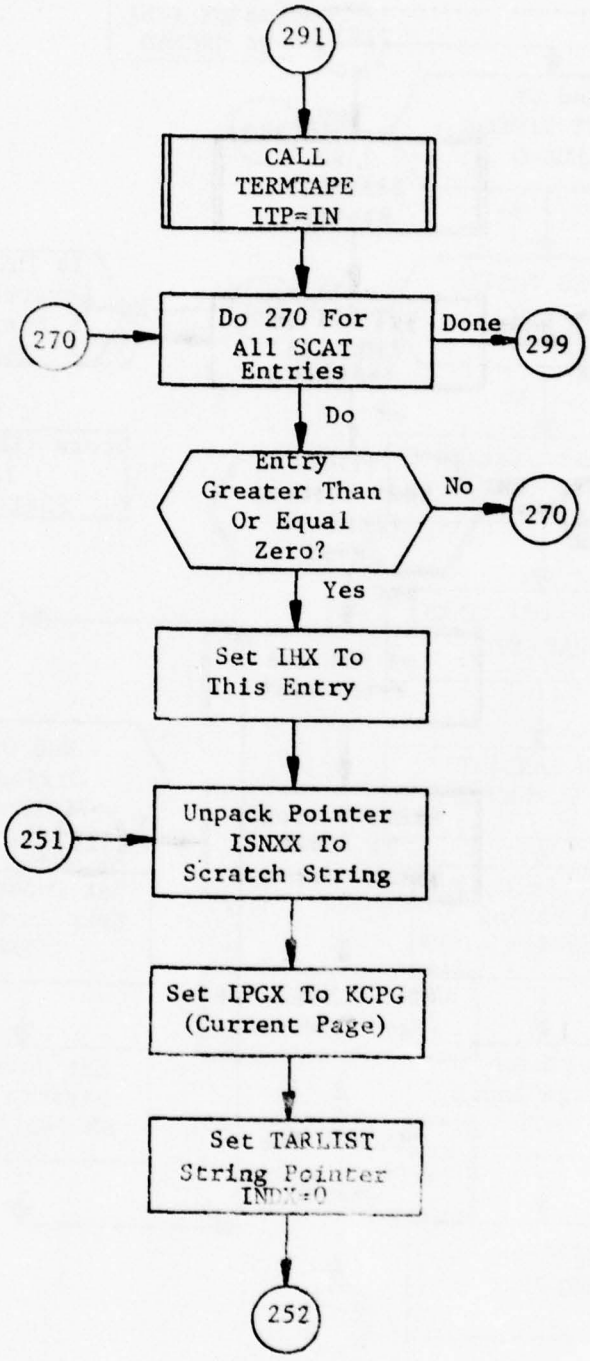


Figure 5. (Part 10 of 17)

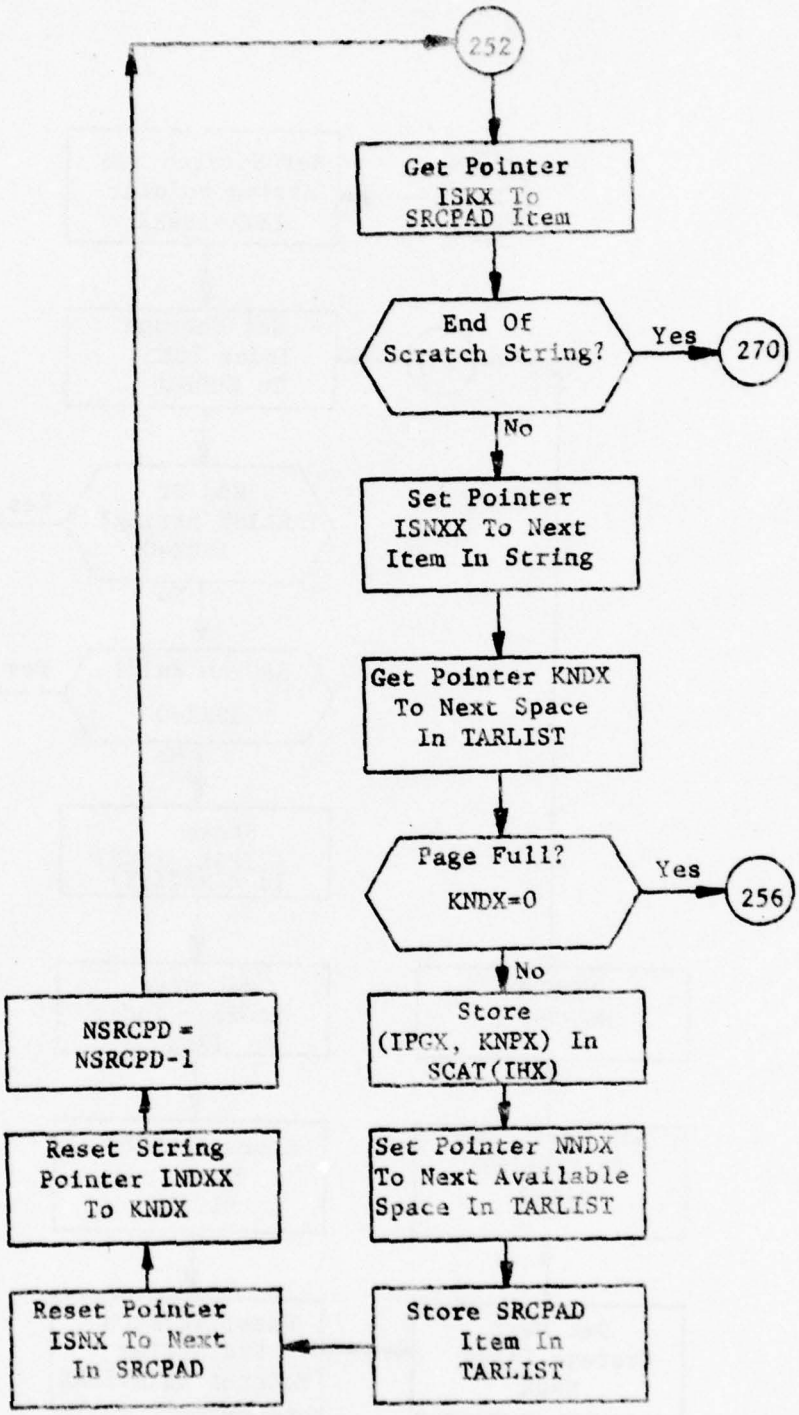


Figure 5. (Part 11 of 17)

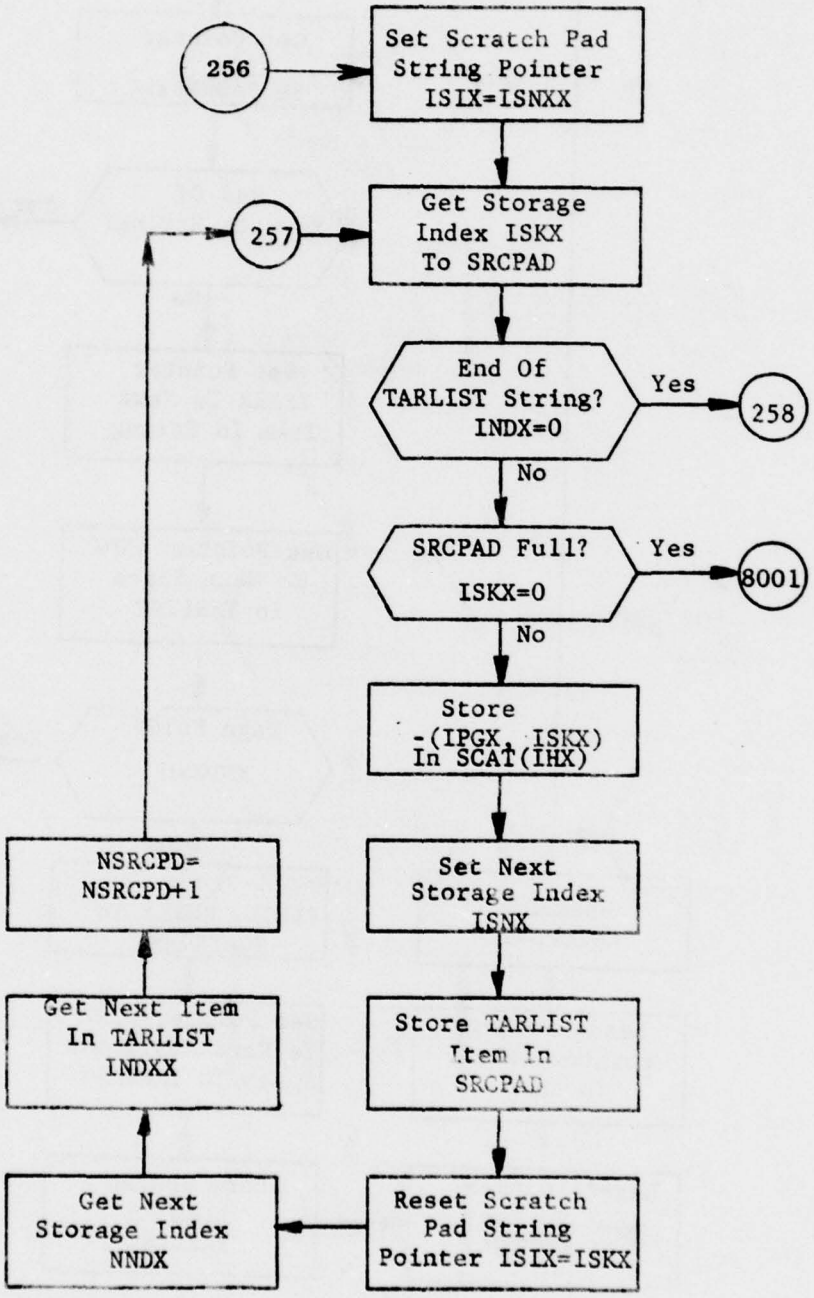


Figure 5. (Part 12 of 17)

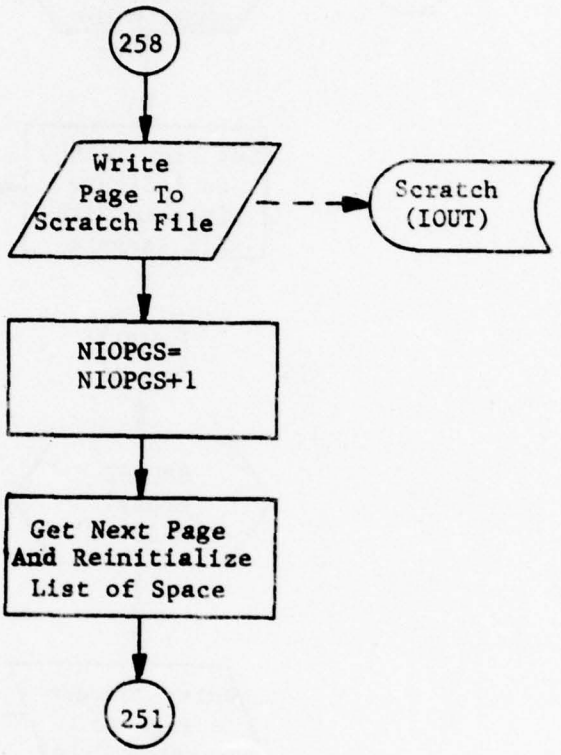


Figure 5. (Part 13 of 17)

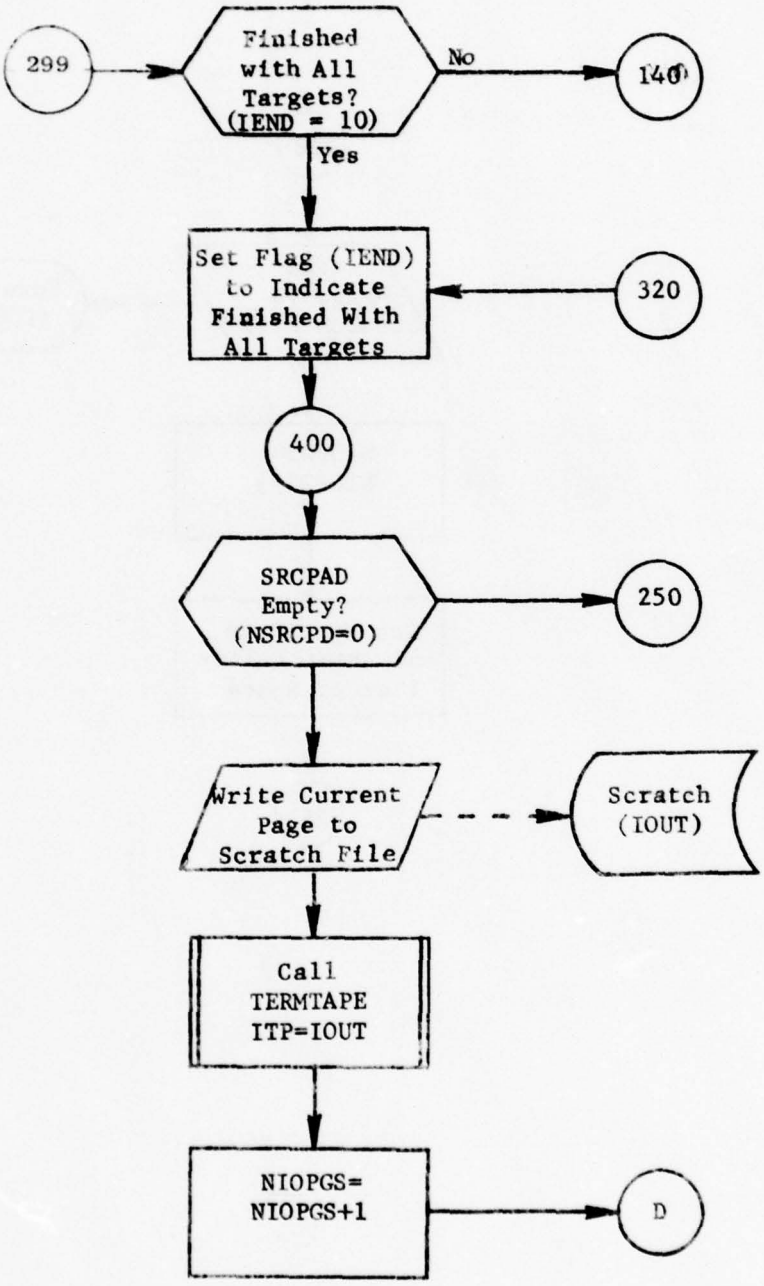


Figure 5. (Part 14 of 17)

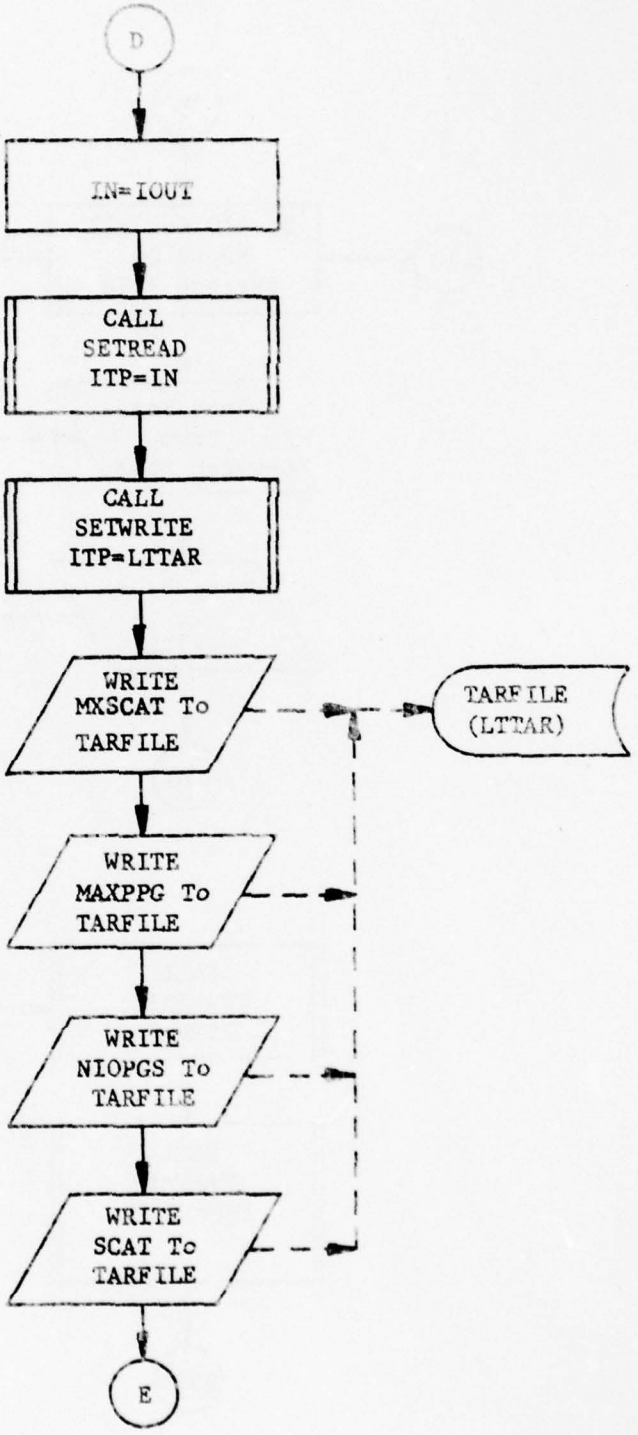


Figure 5. (Part 15 of 17)

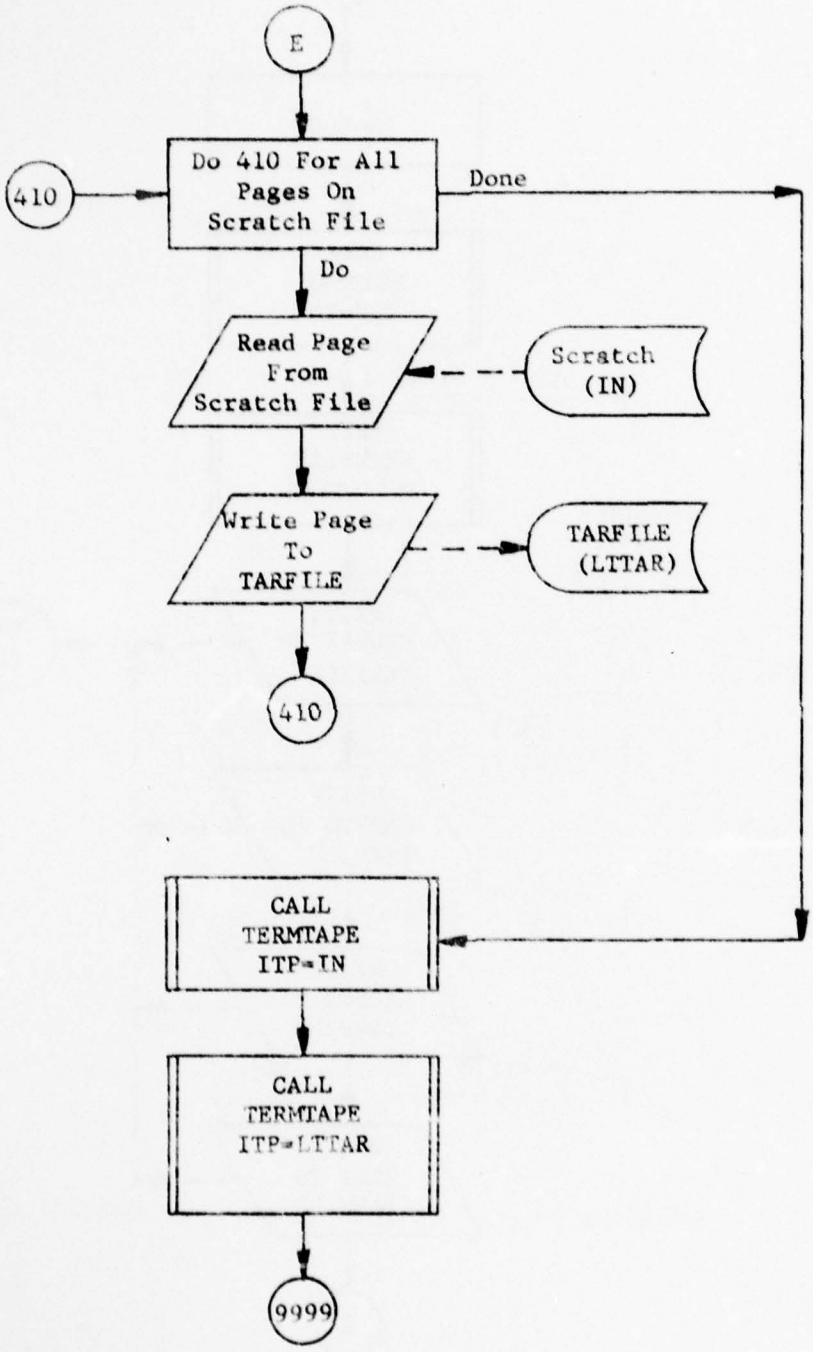


Figure 5. (Part 16 of 17)

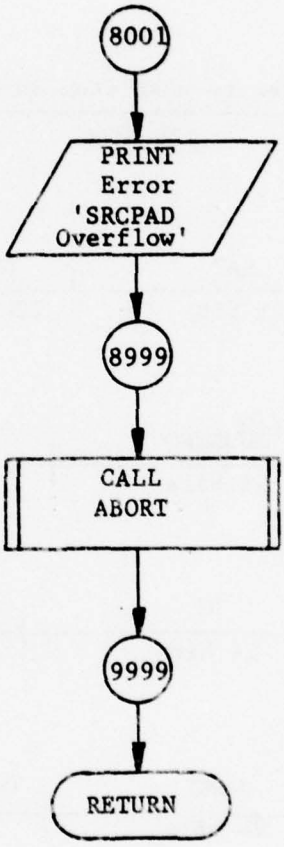
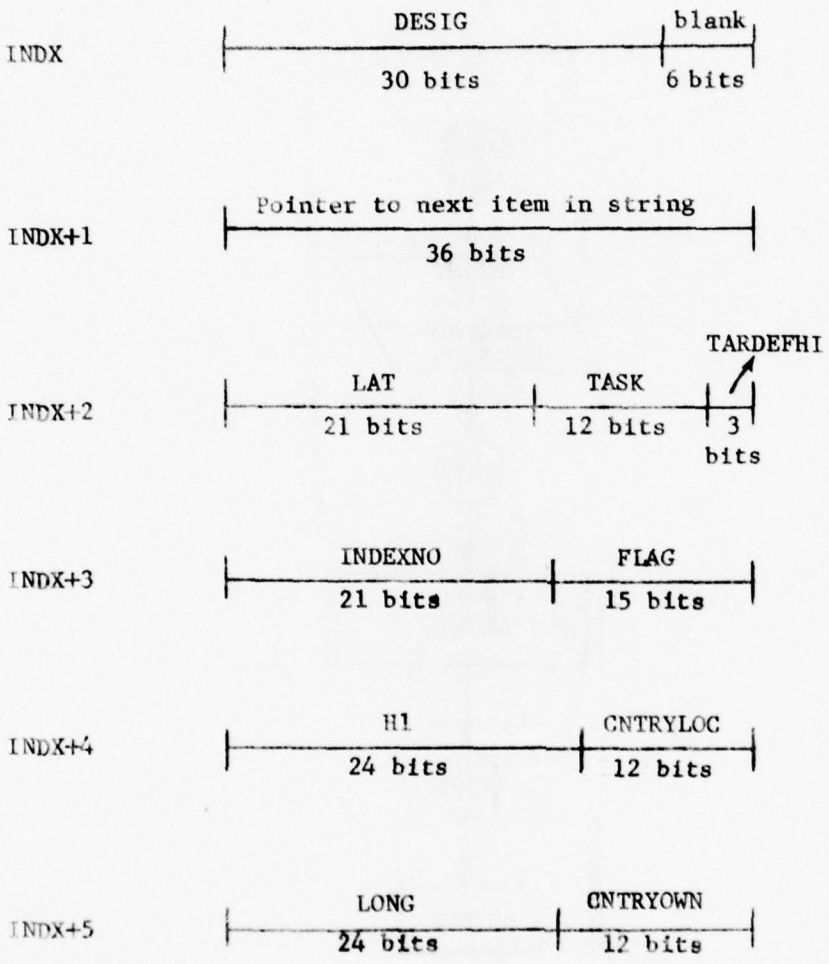


Figure 5. (Part 17 of 17)



Note: LAT and LONG are stored in units of .00001 degrees

Figure 6. TARFILE Packing Description

2.8.1 Subroutine INITPG

PURPOSE: Initialize list of available space for current TARLIST page (CURPGE).

ENTRY POINTS: INITPG

FORMAL PARAMETERS: IPG (page number)

COMMON BLOCKS: C5

SUBROUTINES CALLED: ABORT

CALLED BY: TARLST

Method:

INITPG (see figure 7) creates a list of available space for the current TARLIST page (CURPGE). The list is created by stringing increments (INCR) of words in the TARLIST array. LAVSPG (IPG) is set to one to point to the first of available space for page (IPG).

Subroutine INITPG is illustrated in figure 7.

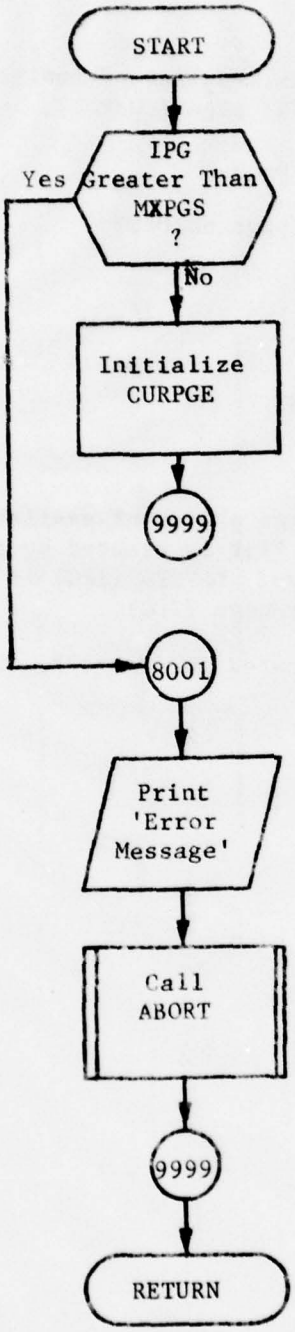


Figure 7. Subroutine INITPG

2.8.2 Subroutine INITAR

PURPOSE: Initialize common block /C5/ used by Program TARLST

ENTRY POINTS: INITAR

FORMAL PARAMETERS: None

COMMON BLOCKS: C5

SUBROUTINES CALLED: None

CALLED BY: TARLST

Method:

INITAR (see figure 8) is called at the beginning of TARLST to initialize the above common block. The TARLST program limits (i.e., MAXPPG, MXSCAT, and MXPGS) are initialized; the SCAT and LAVSPG arrays are set to zero; and the array SRCPAD is initialized to be a list of available scratch space.

Subroutine INITAR is illustrated in figure 8.

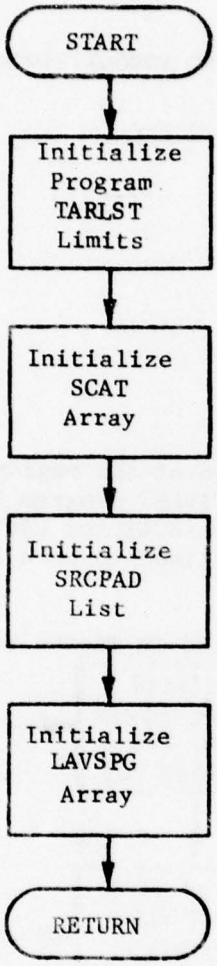


Figure 8. Subroutine INITAR

2.9 Subroutine PARTA*

PURPOSE: To write the BASFILE up to the weapon type arrays

ENTRY POINTS: PARTA

FORMAL PARAMETERS: None

COMMON BLOCKS: ASMONE, ASMTYP, C10, C15, C20, C25, C30, CHARTER, CLASSCOM, CVTAB, ERRCOM, GAMEVAR, GAMFLAG, HAPPEN, ITMFILE, IONPRT, ISIMTYPE, ITP, IWEPREF, LHVALUES, MASTER, MYIDENT, MYLABEL, NFIXES, NFIXREQ, NUMCOR, PAYONE, PAYTYPE, RANGERAR, SUMNEW, TWORD, WEPTWO, WEPVAL, WHONE, WHTYPE, WPVALUES

SUBROUTINES CALLED: DEPROUT, DIRECT, FACTORCG, FIXWEP, HDFND, HEAD, INSGET, ITLE, NEXTTT, PENROUT, RETRV, SETWRITE, WRARRAY, WRWORD

CALLED BY: ENTMOD (of PREPALOC)

Method:

The design for PARTA is nearly self-explanatory since code merely chains the proper records, stores local arrays and writes data onto the BASFILE according to the format given in table 2. No calculations are conducted, only the determination of BASFILE arrays.

In addition to BASFILE generation, input adverbs as user directed are read and local parameters LOCSET and LOCFIX are set to the starting position into INSGET's arrays. IONPRT is set if user requests non-standard prints.

Subroutine FACTORCG redefines parameters as user directed for attributes MINKILL, MAXKILL, VAL, and height-of-burst setting. Also FIXWEP updates the fix weapon assignment linkage.

Subroutines PENROUT and DEPROUT calculates penetration and depenetration corridor parameters.

Upon exiting PARTA, the BASFILE is only initially generated, other calculations to follow will complete the writing of BASFILE.

*Main subroutine of overlay link B.

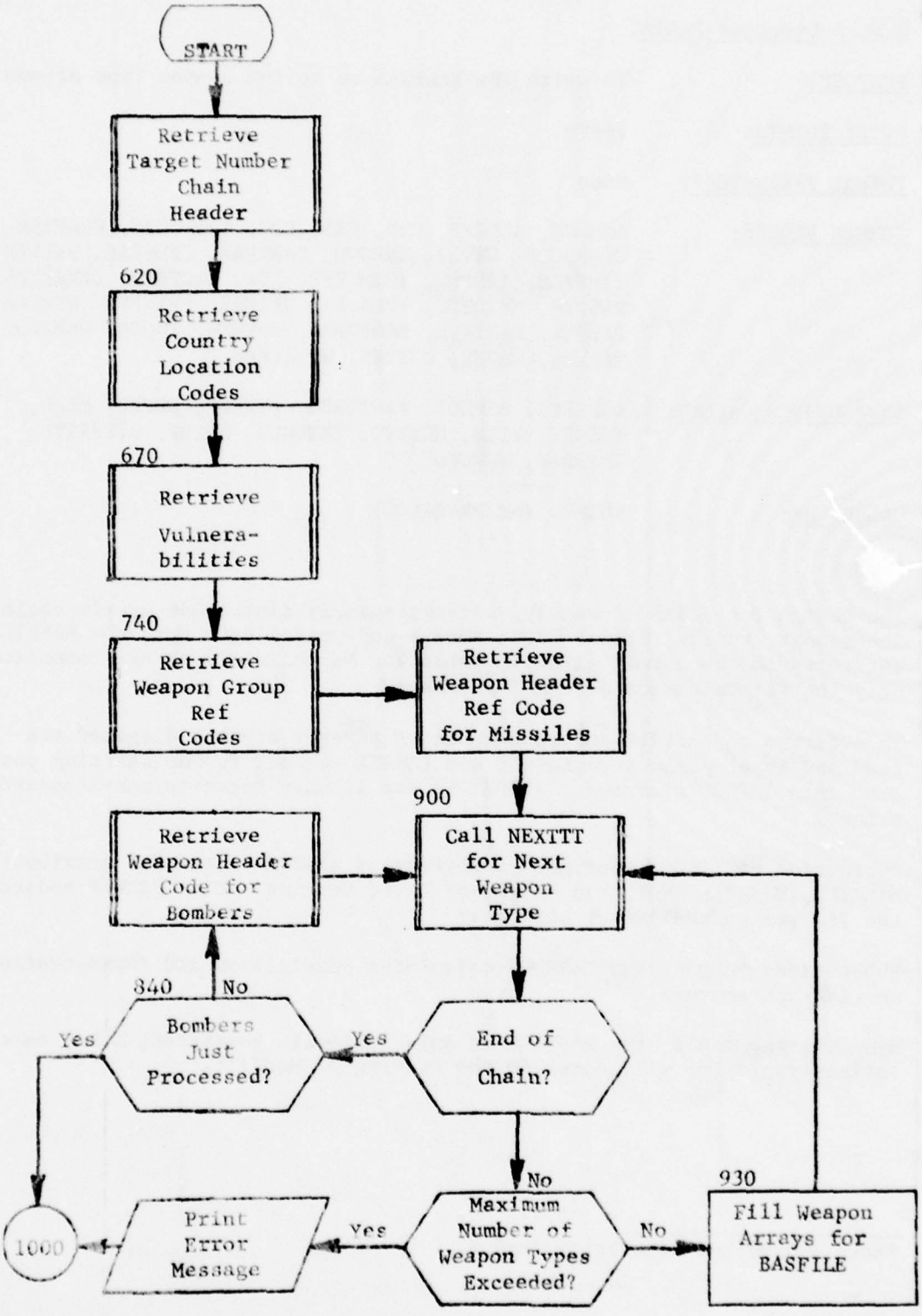


Figure 9. Subroutine PARTA (Part 1 of 6)

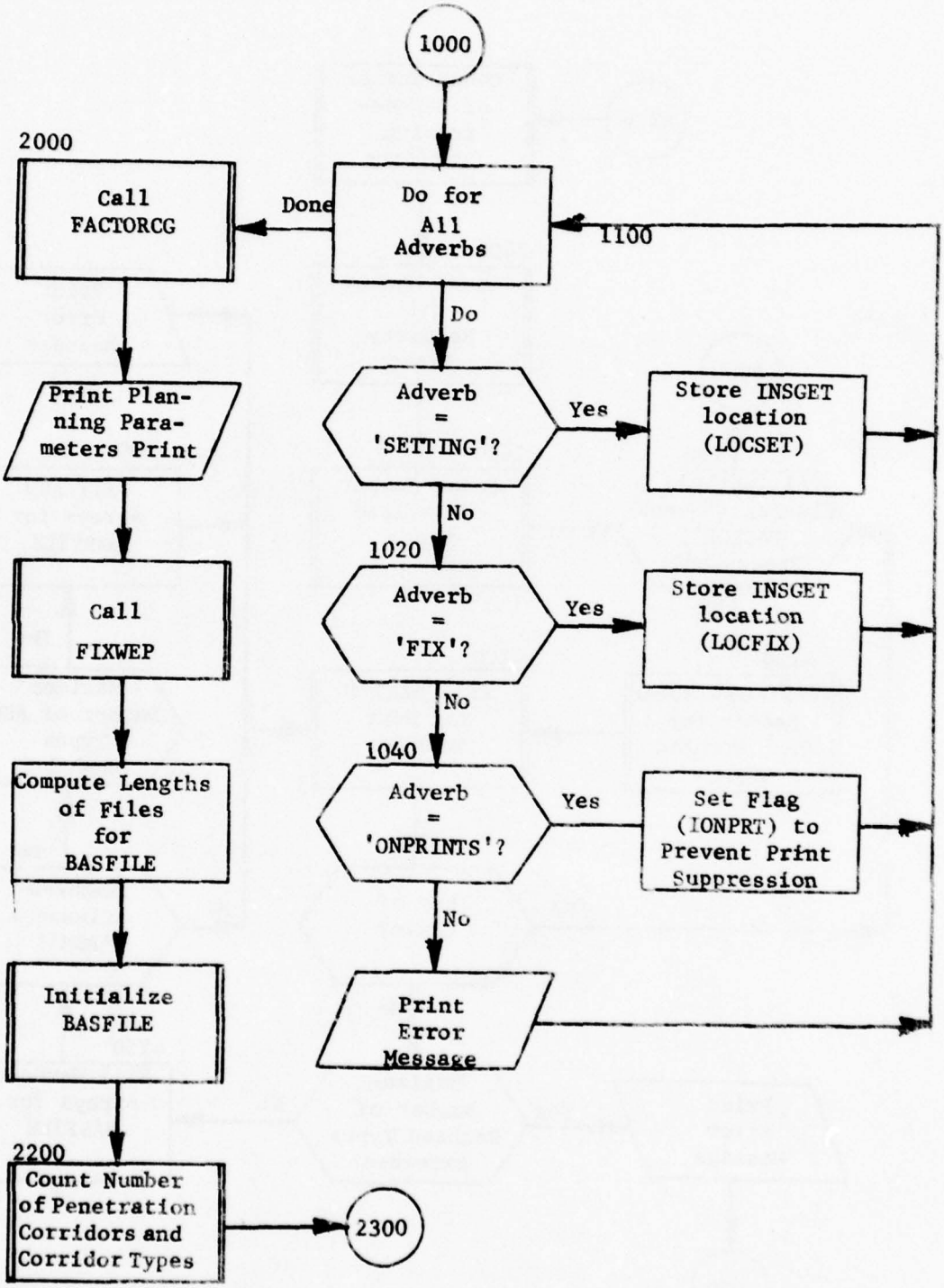


Figure 9. (Part 2 of 6)

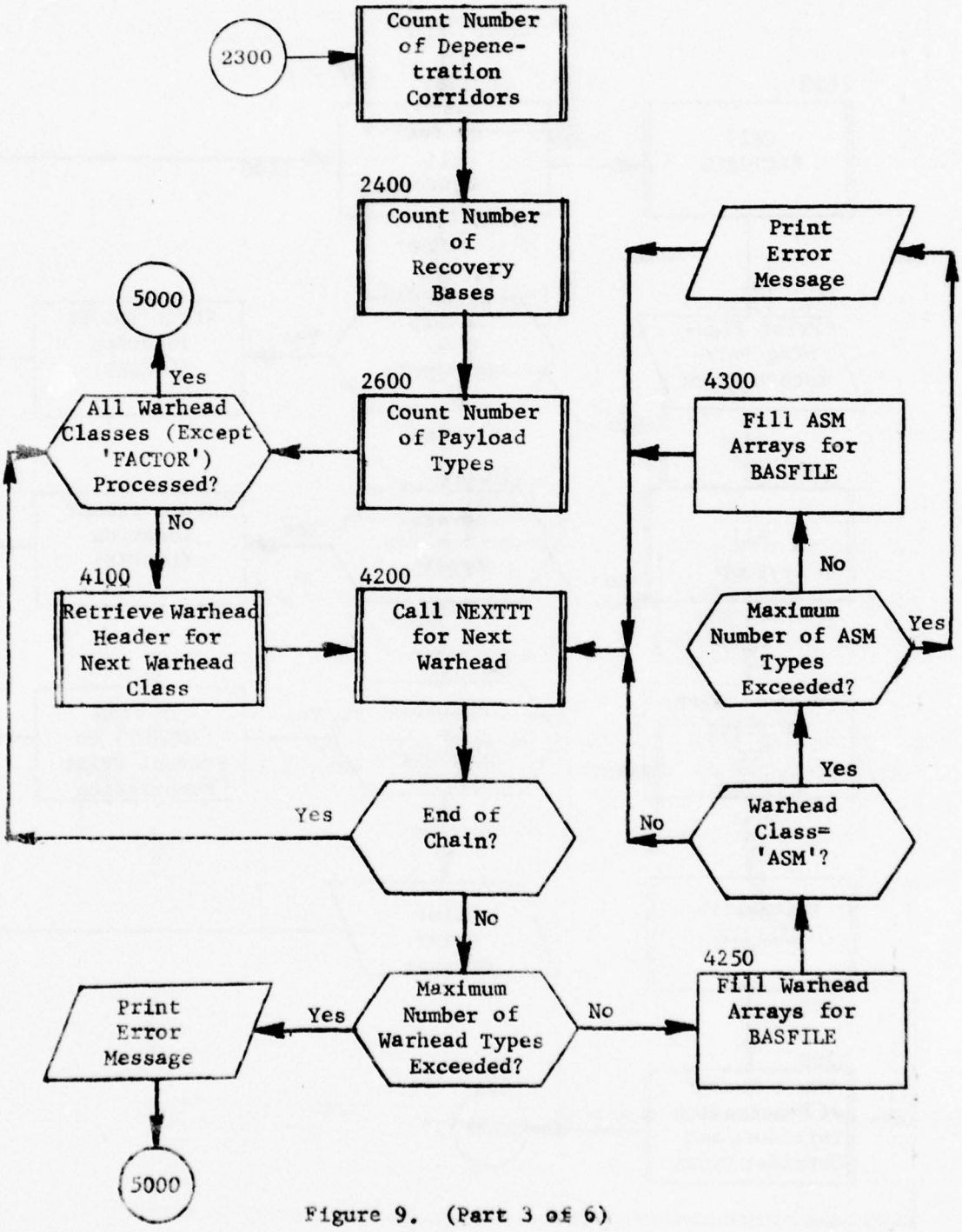


Figure 9. (Part 3 of 6)

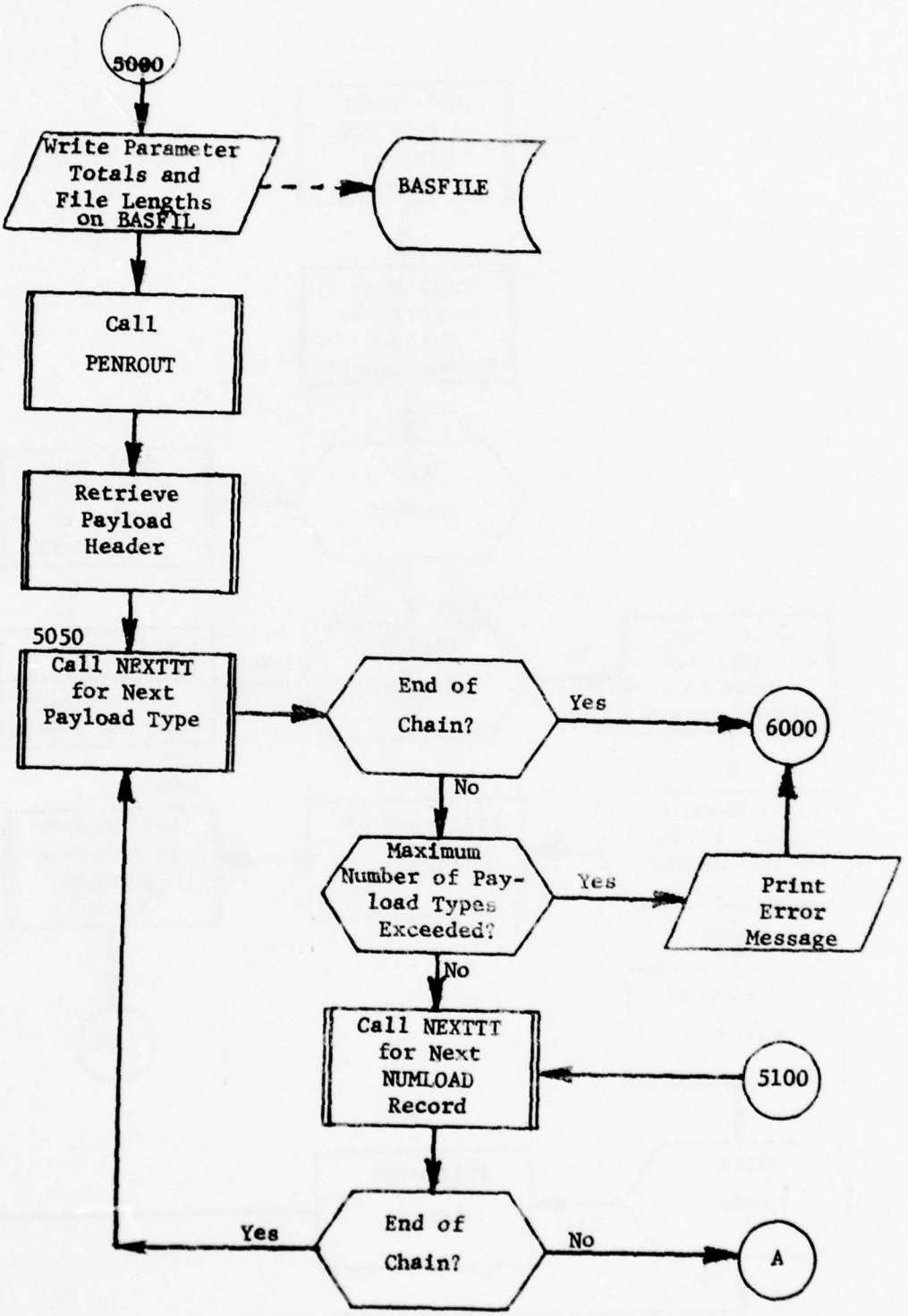


Figure 9. (Part 4 of 6)

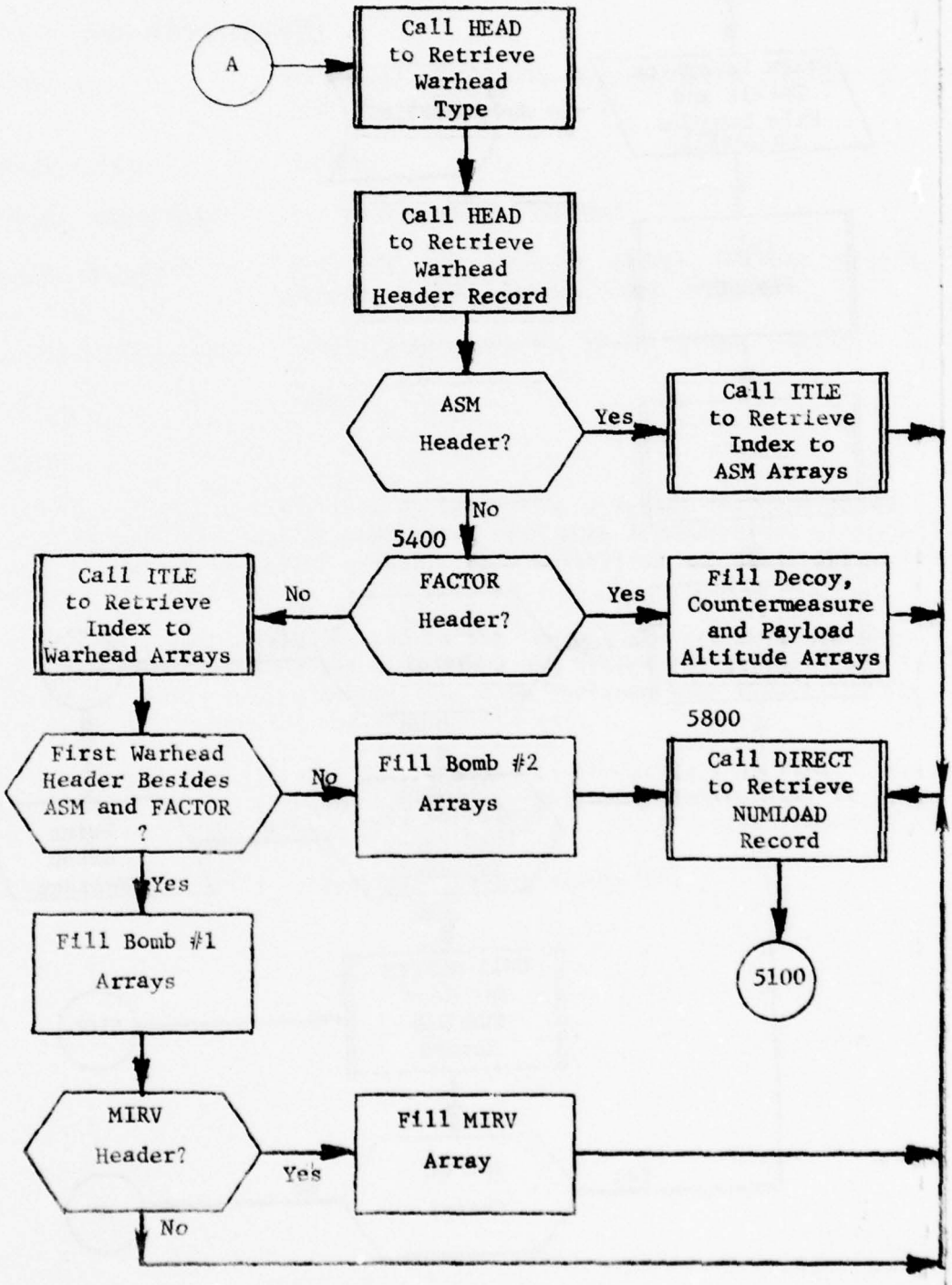


Figure 9. (Part 5 of 6)

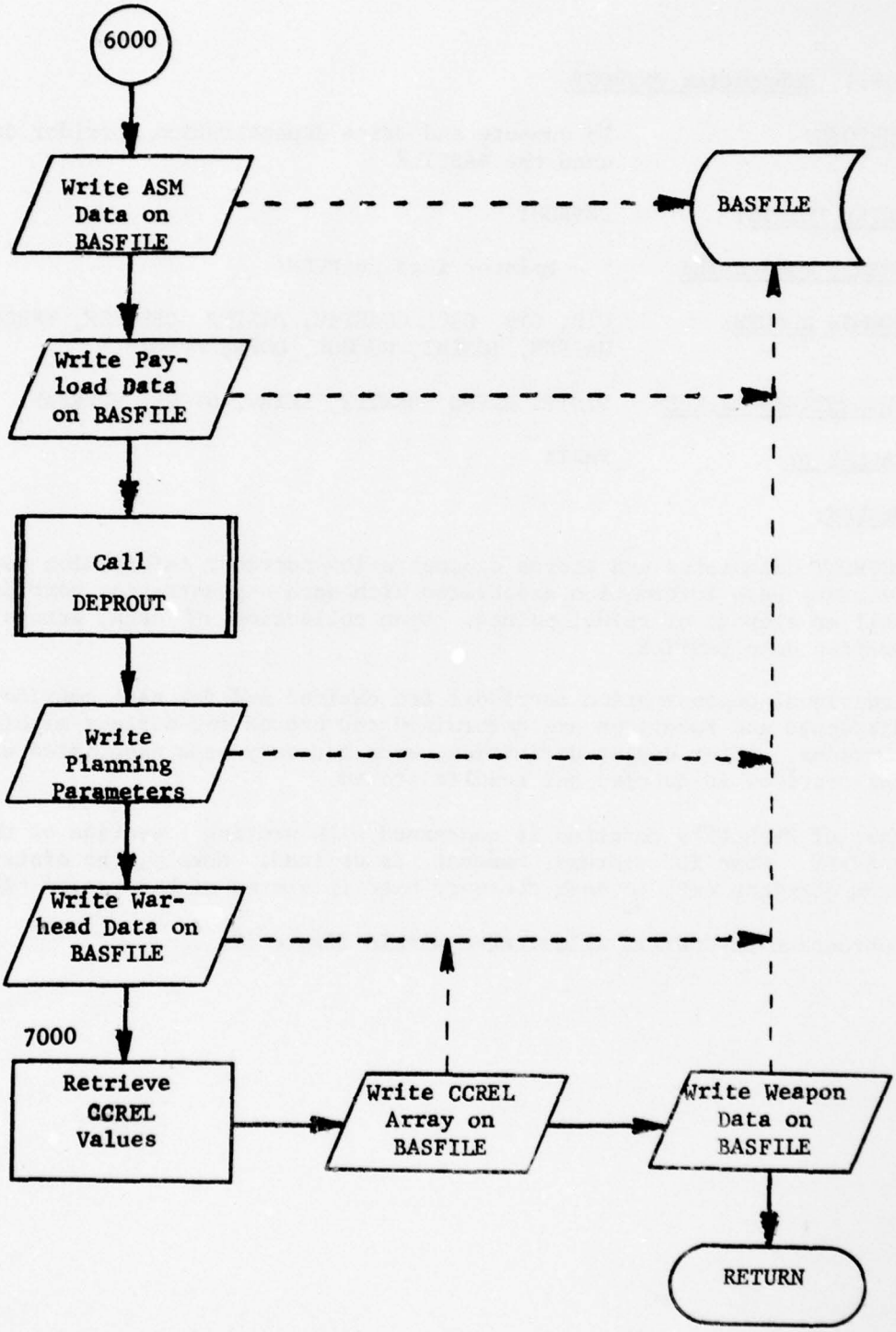


Figure 9. (Part 6 of 6)

2.9.1 Subroutine DEPROUT

PURPOSE: To compute and write depenetration corridor data onto the BASFILE

ENTRY POINTS: DEPROUT

FORMAL PARAMETERS: N - Pointer into /HAPPEN/

COMMON BLOCKS: C10, C15, C30, CHARTER, DISTEF, DPENREF, ERRCOM, HAPPEN, IONPRT, NUMCOR, OOPS, RFPOINTS

SUBROUTINES CALLED: DISTF, HDFND, NEXTTT, RETRV, STORE, WRARRAY

CALLED BY: PARTA

Method:

DEPROUT calculates and stores depenetration corridor information and recovery base information associated with each depenetration corridor as well as storage of refuel points. Upon collection of data, arrays are written onto BASFILE.

Individual depenetration corridors are chained and for each corridor, distances and locations are determined and stored for doglegs within the corridor. After dogleg definition, each recovery base associated with the corridor is queried and results stored.

Most of DEPROUT's function is concerned with writing a section of the BASFILE. Some IDS storage, however, is defined. Namely, the distances from corridor exit to each recovery base is stored within record 'RDDIST.'

Subroutine DEPROUT is illustrated within figure 10.

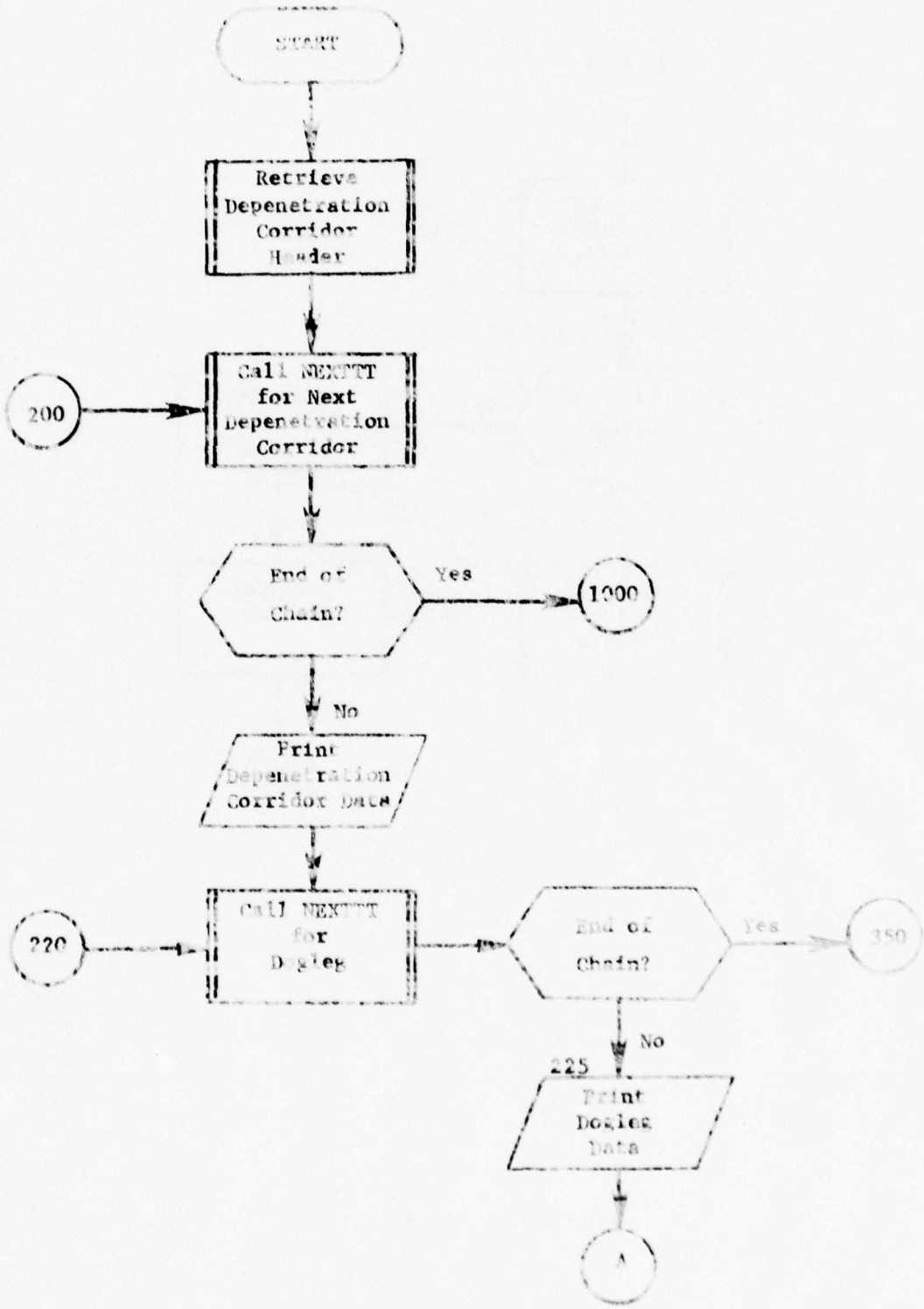


Figure 10. Subroutine DEPROUT (Part 1 of 5)

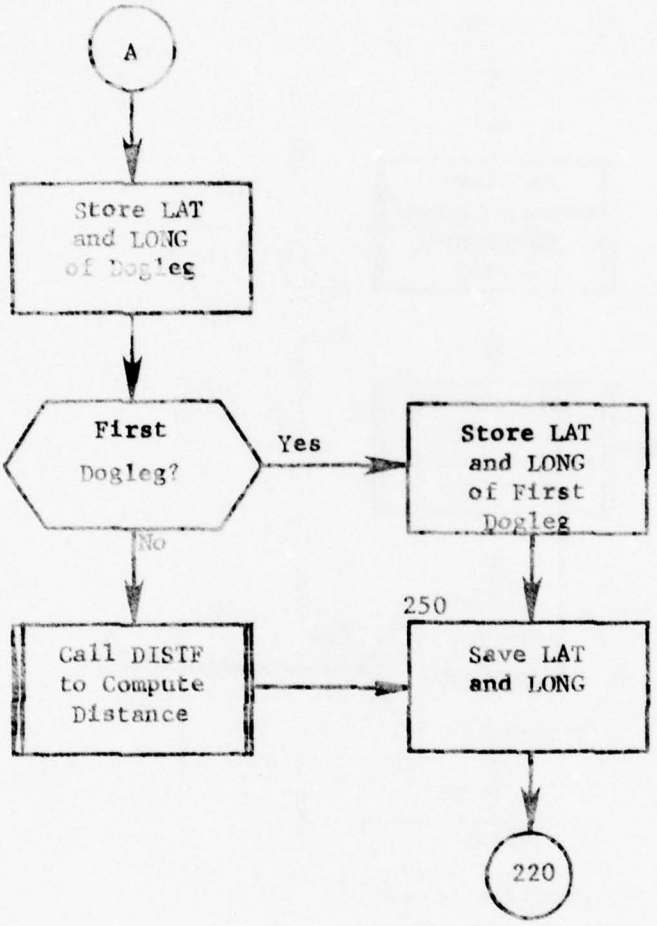


Figure 10. (Part 2 of 5)

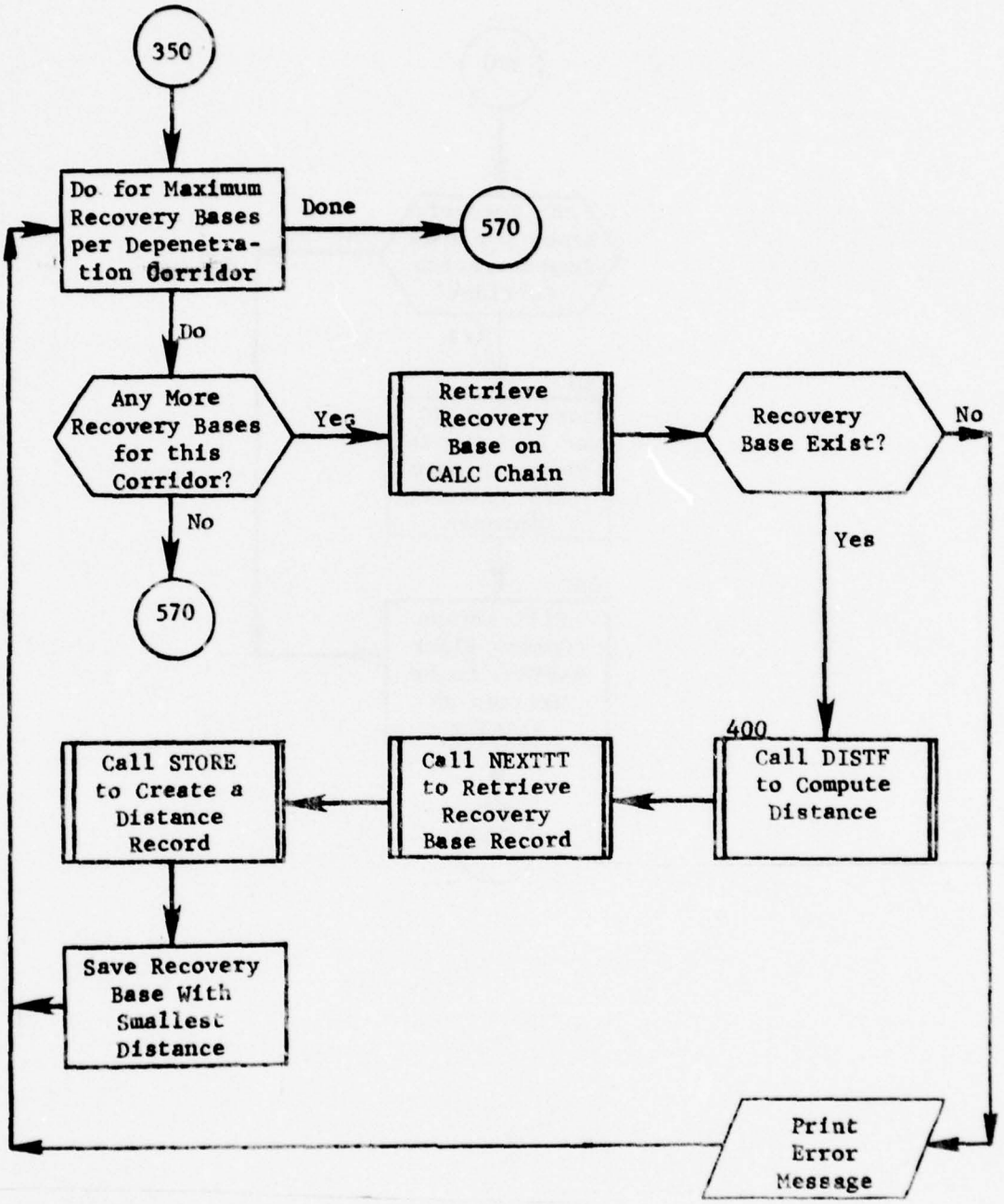


Figure 10. (Part 3 of 5)

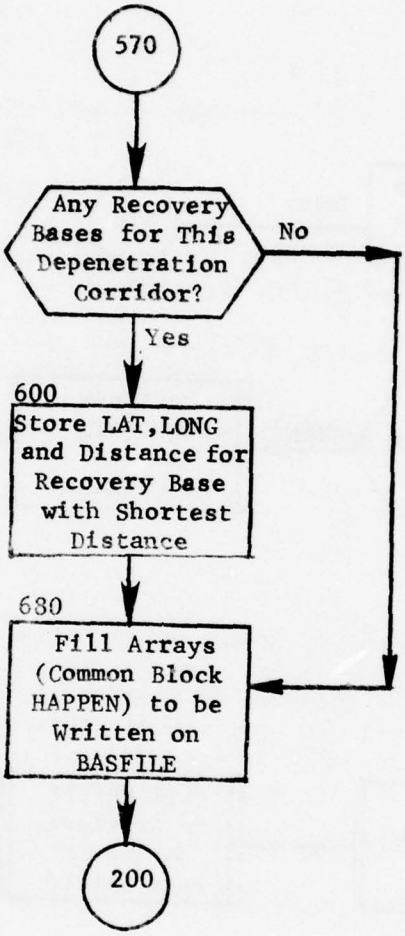


Figure 10. (Part 4 of 5)

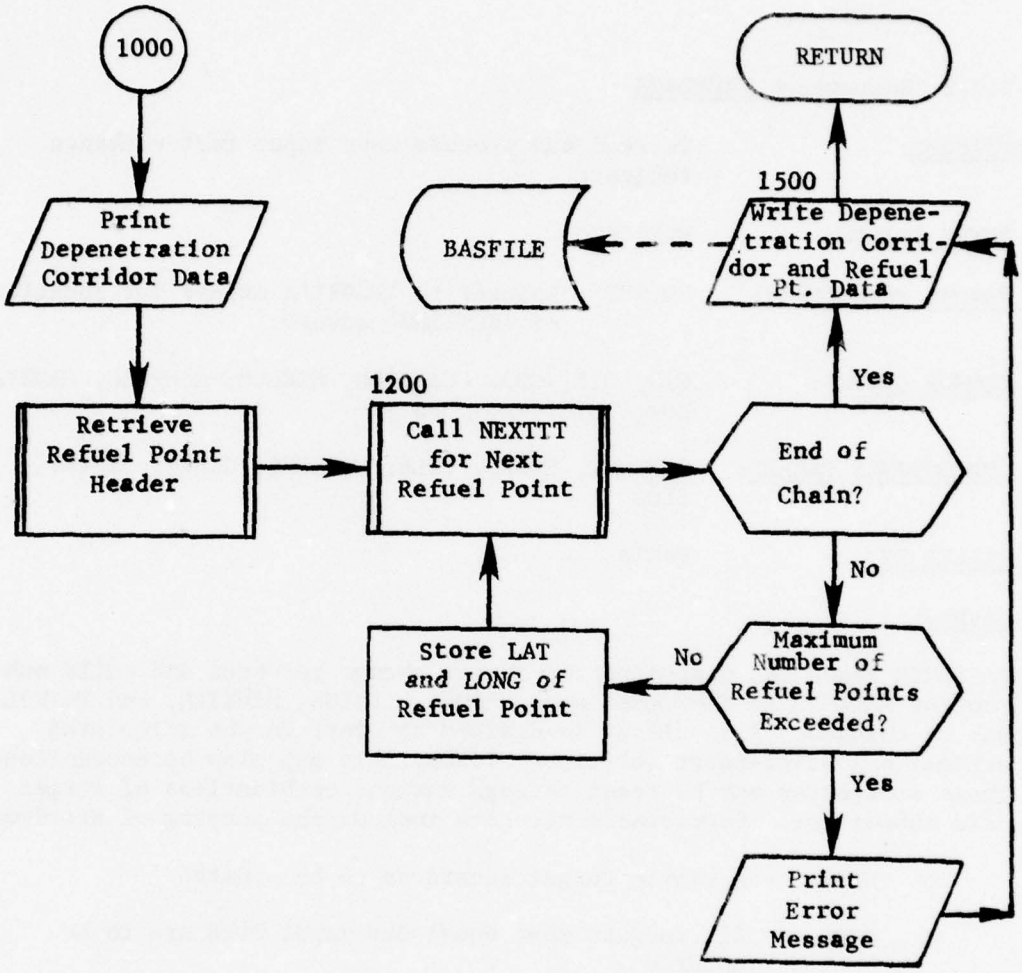


Figure 10. (Part 5 of 5)

2.9.2 Subroutine FACTORCG

PURPOSE: To read and process user input factor change requests

ENTRY POINTS: FACTORCG

FORMAL PARAMETERS: LOCSET - Pointer to INSGET's arrays for location of 'SETTING' adverb

COMMON BLOCKS: C10, C15, C30, CLASSCOM, ERRCOM, GAMFLAG, GAMEVAR, HOB, ISIMTYPE, OOPS

SUBROUTINES CALLED: CINSGET, HDFND, ITLE, MAKECHG, NEXTTT, RETRV, SLOG

CALLED BY: PARTA

Method:

FACTORCG reads the user directed factor change requests and calls subroutine MAKECHG to implement each change. VALUE, MINKILL, and MAXKILL may be changed. Also, if it is desired to override the calculated optimal height-of-burst (attribute IDHOB) this may also be accomplished. These attributes may be reset through various combinations of target data subsetting. Permissable requests include the setting of attributes:

- o DESIG - A single target record is to be updated
- o TYPE - All targets that equal the input TYPE are to be updated
- o CLASS - All targets that equal the input target class name are to be updated
- o CNTRYL - The height-of-burst of all targets located within the input country location are to be updated
- o IREG - The height-of-burst of all targets located within the input region are to be updated

The last two requests recognize only height-of-burst requests. Otherwise, any combination of target subsetting is premissable but there is a ranking order in the final storage of input values. The order of priority is: DESIG, TYPE, CLASS, CNTRYL, IREG. That is, if a given target is to have an attribute updated by more than one input target sets, the cited order applies. For instance, consider inputs that request updating attribute VALUE to 40 for TYPE=TITAN and to 50 for CLASS=MISSILE.

Since TITAN's are in fact missiles, the ranking order resolves any conflicts and, accordingly, all target records where CLASS=MISSILE will have VALUE=50 except for those records where TYPE=TITAN in which case VALUE is set to 40.

All code implemented within FACTORCC up to statement number 1500 reads the input requests (by calling subroutine INSGET) and upon proper definition, the changes are made by calling subroutine MAKECHG.

Changes are not made as read. This is because of the rule that once a complex target has been changed, its components may not be separately changed for the same factor. (All of its components are, however, changed by a similar ratio as the complex target as part of the change to the complex target.) Changes are made first for individual targets, then for targets based on TYPE, then CLASS and so on. Local parameter ICRIT equals 1, 2, 3, 4, or 5 defining if the latest input defines targets to be updated on a region, country, class, type, or DESIG, respectively basis. A second parameter called NOWCRIT (and assuming the same values as ICRIT) is initially set to 5 to show that only targets specified by DESIG are to be updated. If ICRIT does not equal NOWCRIT, processing is delayed and if this condition exists array ICRFIRST (ICRIT) is set to the first location into INSGET's array that defines a change request at ICRIT level.

Upon processing all inputs, NOWCRIT is decremented by one and processing reinitiated for that level of priority.

The generalized nature of inputs does not demand any one order of input definition. That is, for say a DESIG, VALUE intersection setting, the user may within the command sentence define either DESIG or VALUE initially; neither order of input is more powerful than the other. Therefore, the design must recognize that changes may not be honored until necessary data has been read. Accordingly, the following parameters are used to control processing:

- IFACTOR - =1, updating attribute VALUE
  =2, updating attribute MINKILL
  =3, updating attribute MAXKILL
  =4, updating attribute IDHOS

- ICRTFAC - =0, a subset of targets pertaining to an input factor has not been read
  =1, a subset of targets has been read and was of the correct processing priority
  =2, a subset of targets has been read but it was not the correct processing priority (or a typographical error was detected). Request change is delayed

- IFACFLAG - =0, an attribute to be changed has not been read
  =1, an attribute to be changed has been read and may be processed
  =2, an attribute to be changed has been read but a typographical error exists

Upon proper conditions, a change request is honored. If an individual target record is to be updated (ICRIT=1), the record is retrieved and MAKECHG is called for target modifications. Otherwise the correct target record level must be retrieved, for subroutine MAKECHG will chain the individual target record chain with the assumption the next highest IDS record level has been retrieved.

Subroutine FACTORCG is illustrated in figure 11.

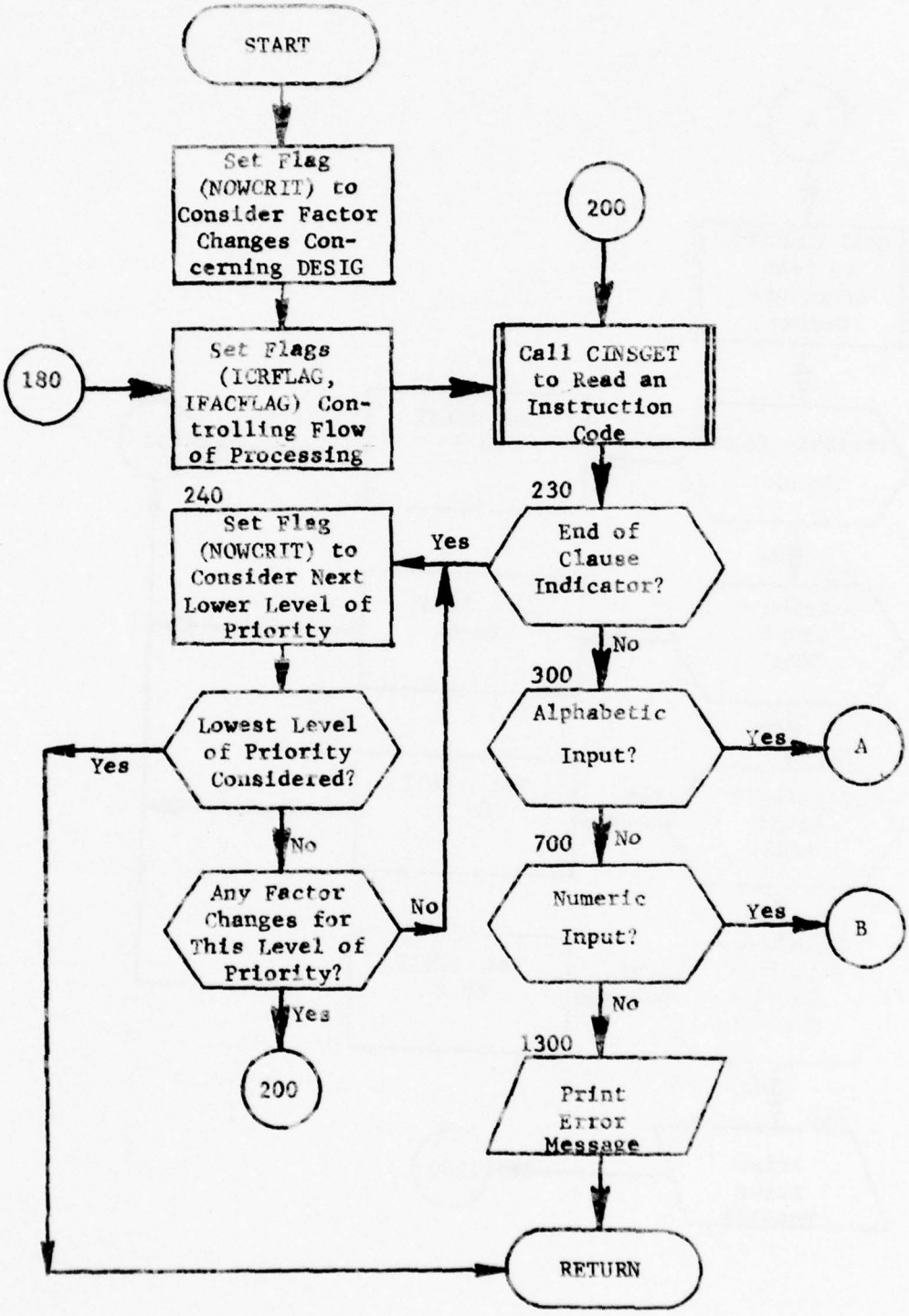


Figure 11. Subroutine FACTORCG (Part 1 of 10)

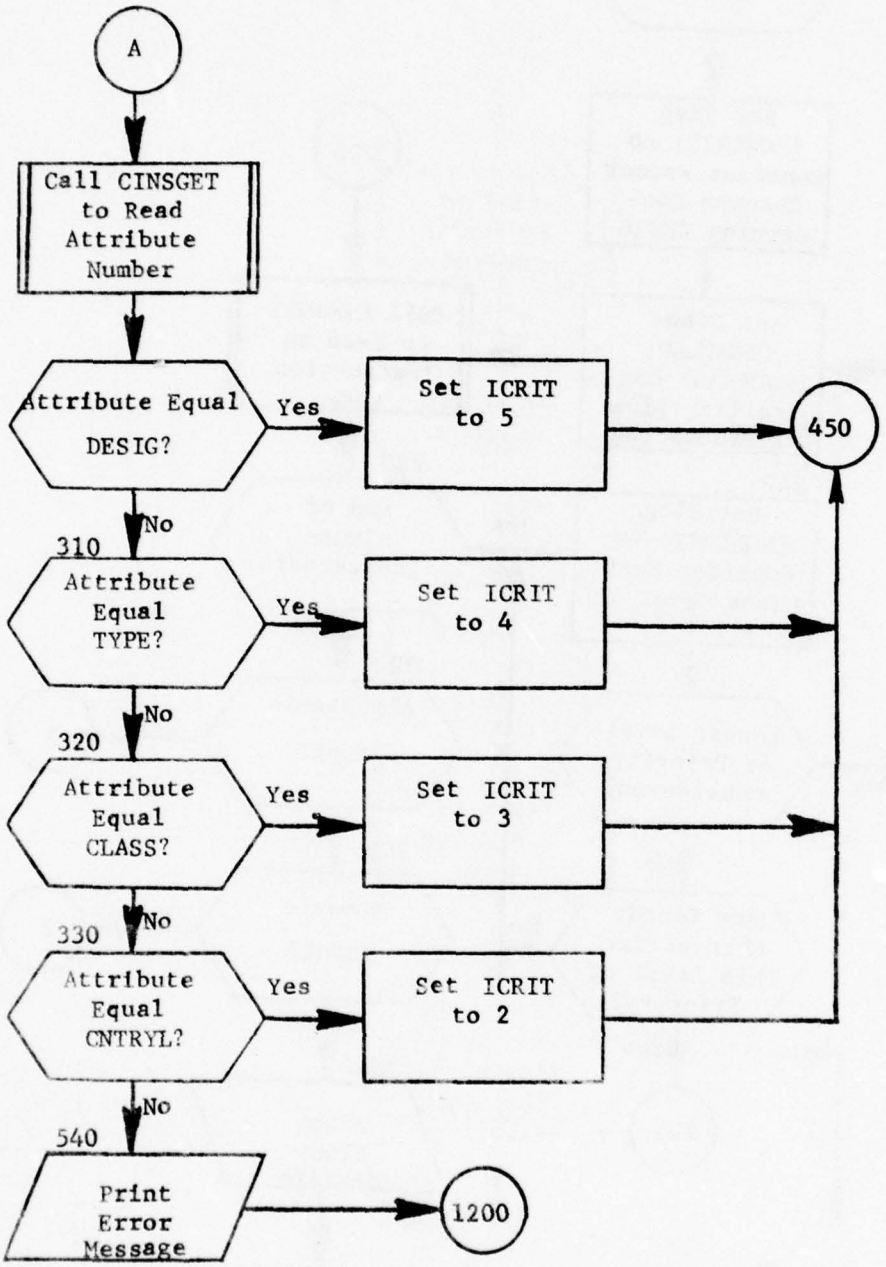


Figure 11. (Part 2 of 10)

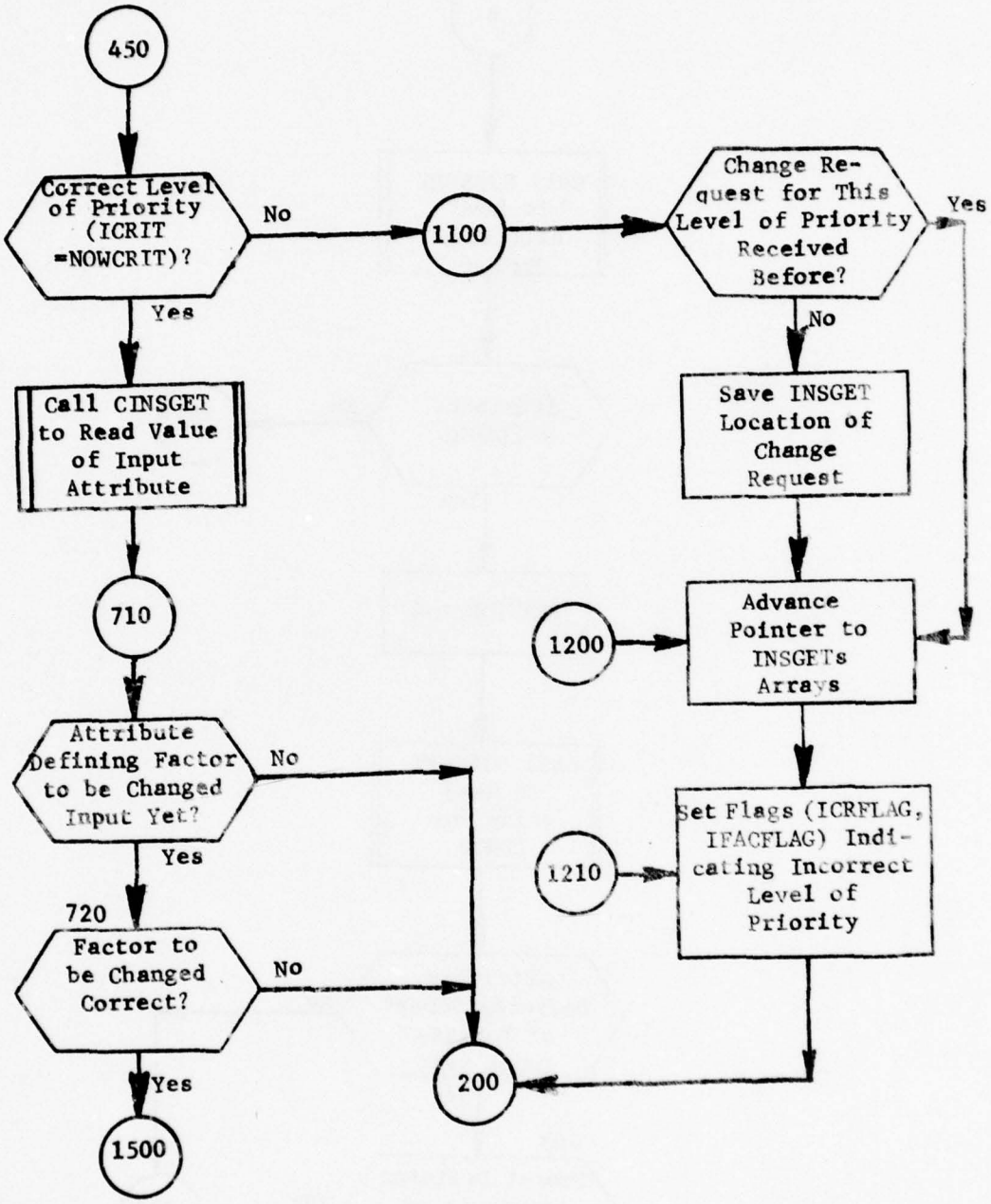


Figure 11. (Part 3 of 10)

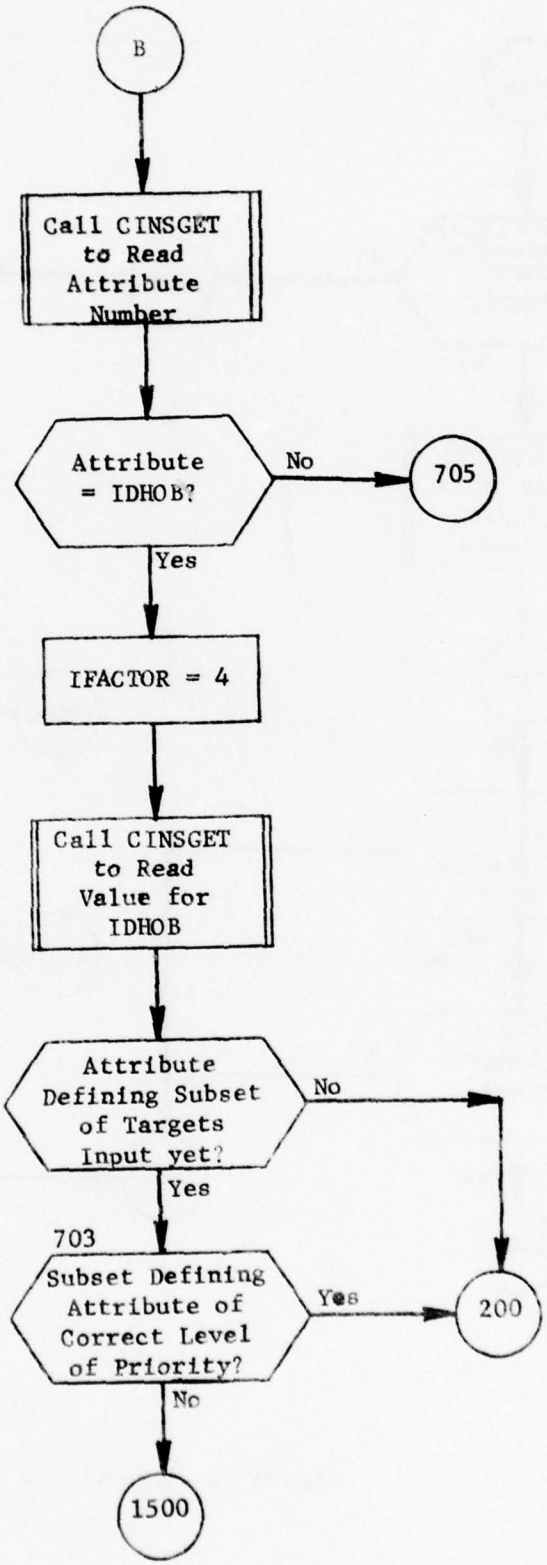


Figure 11. (Part 4 of 10)

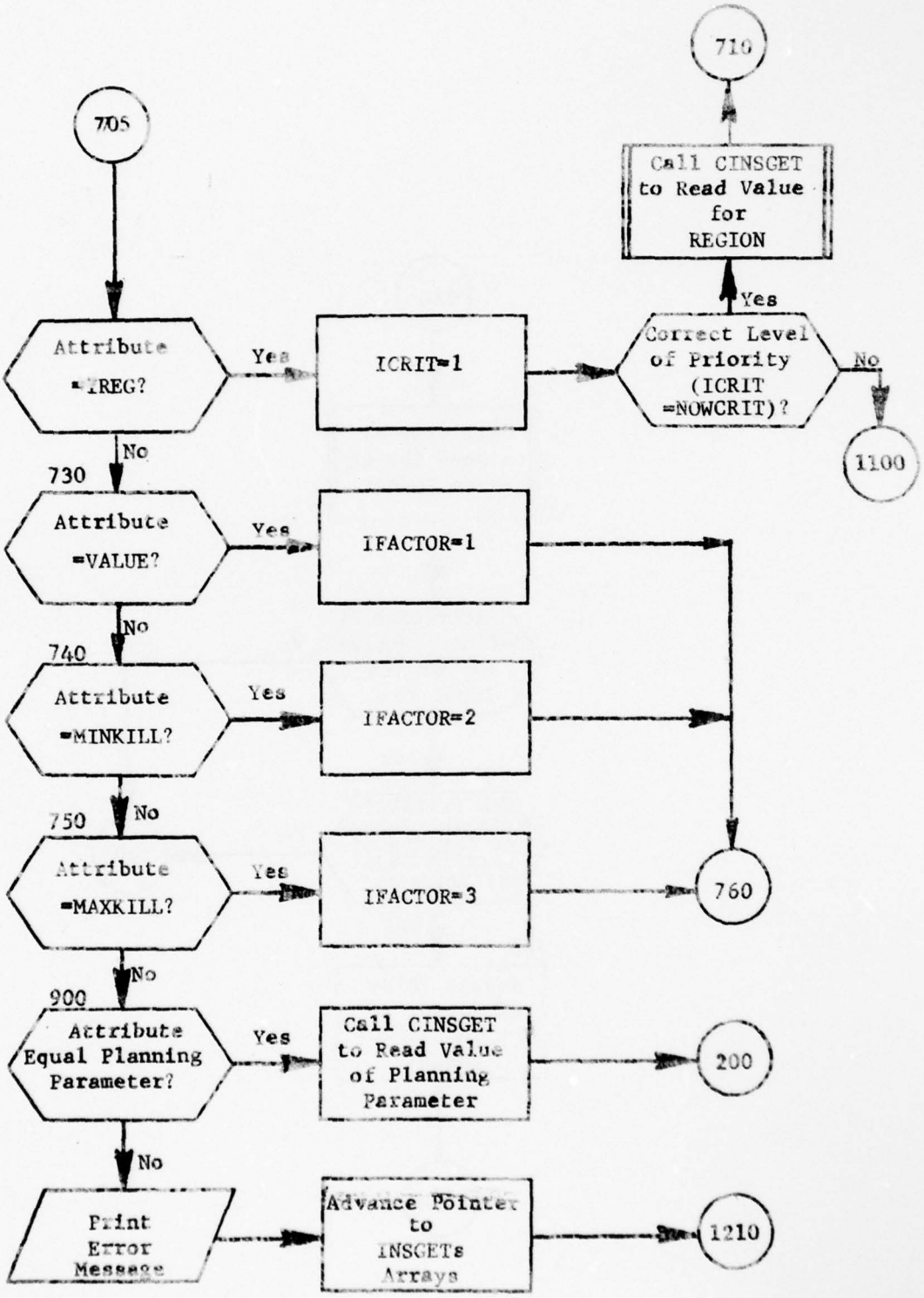
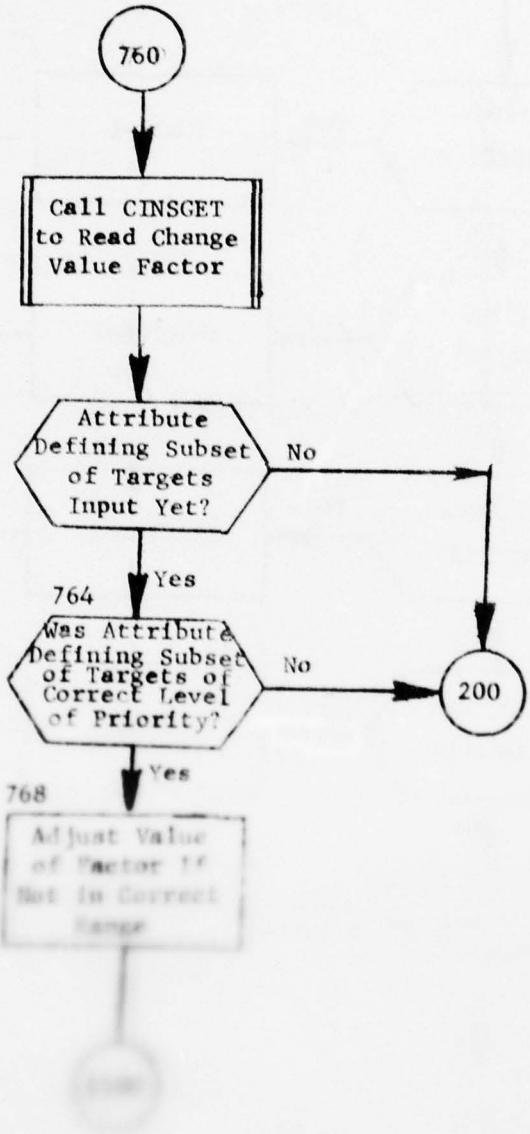
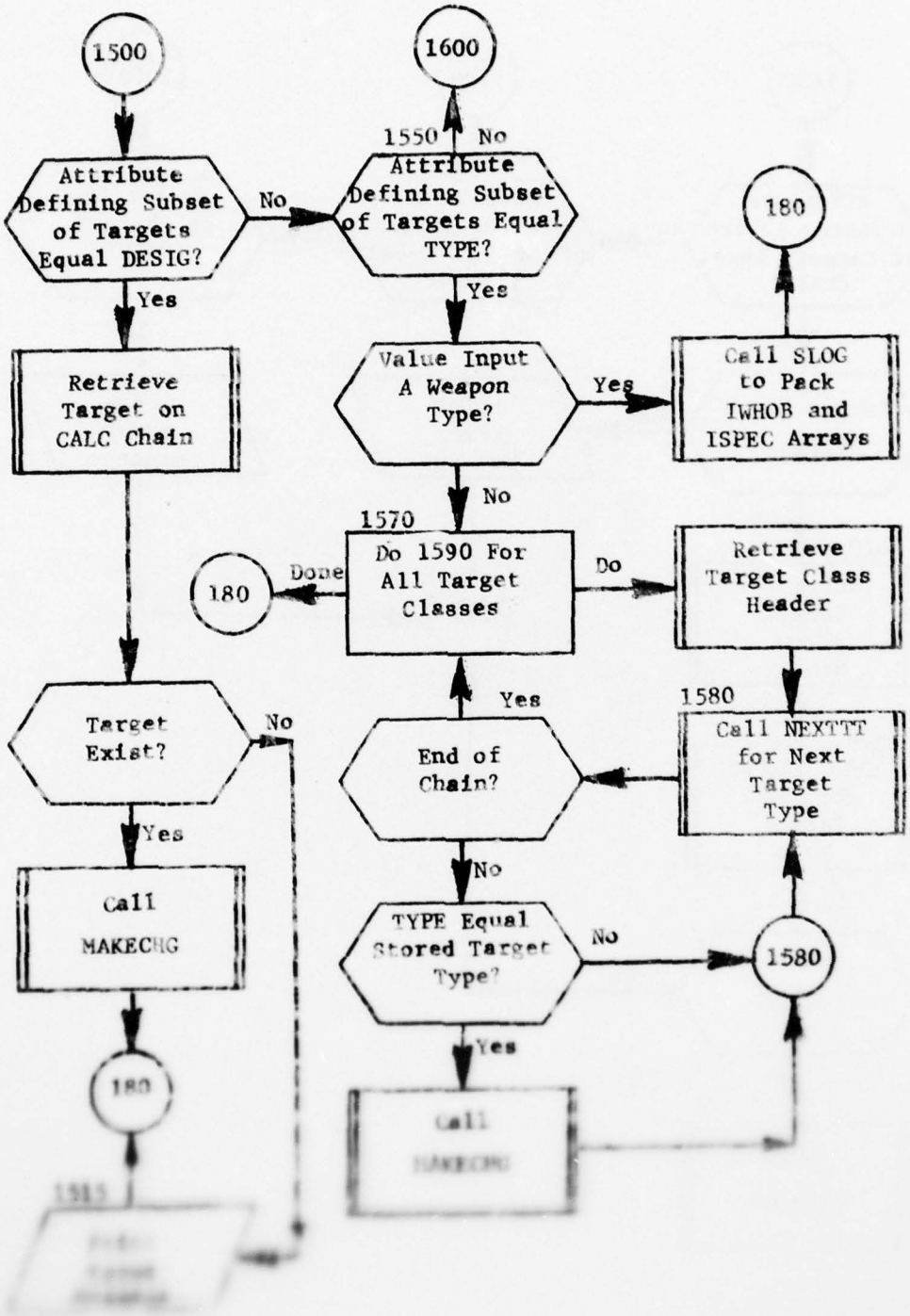
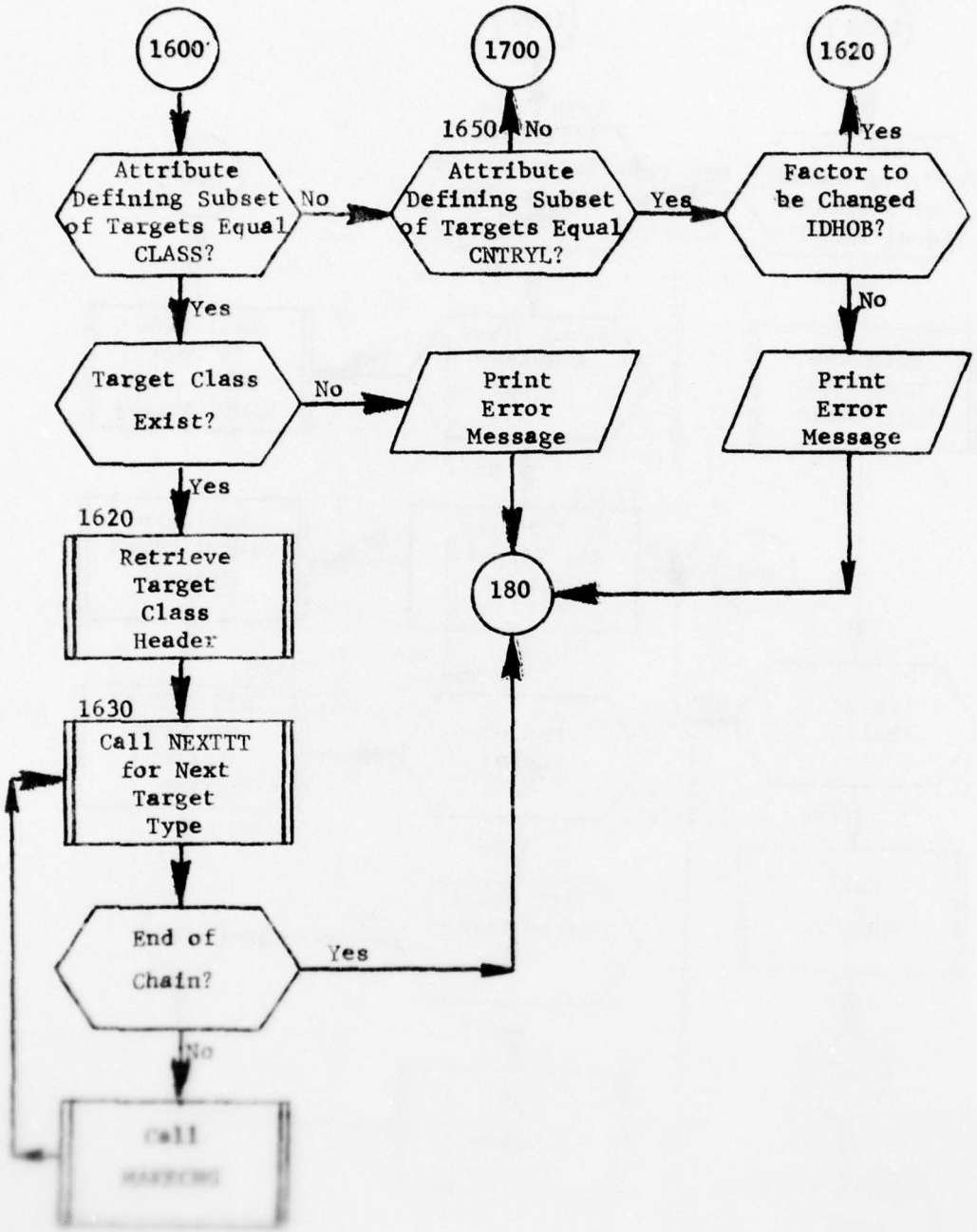
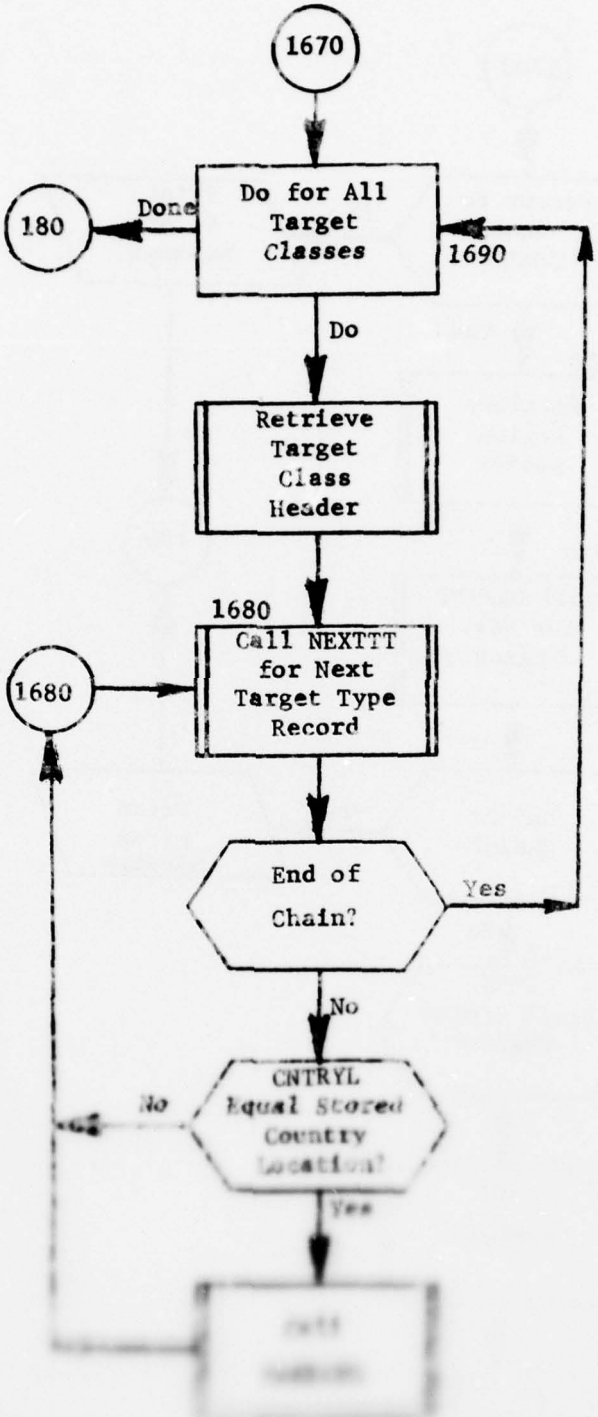


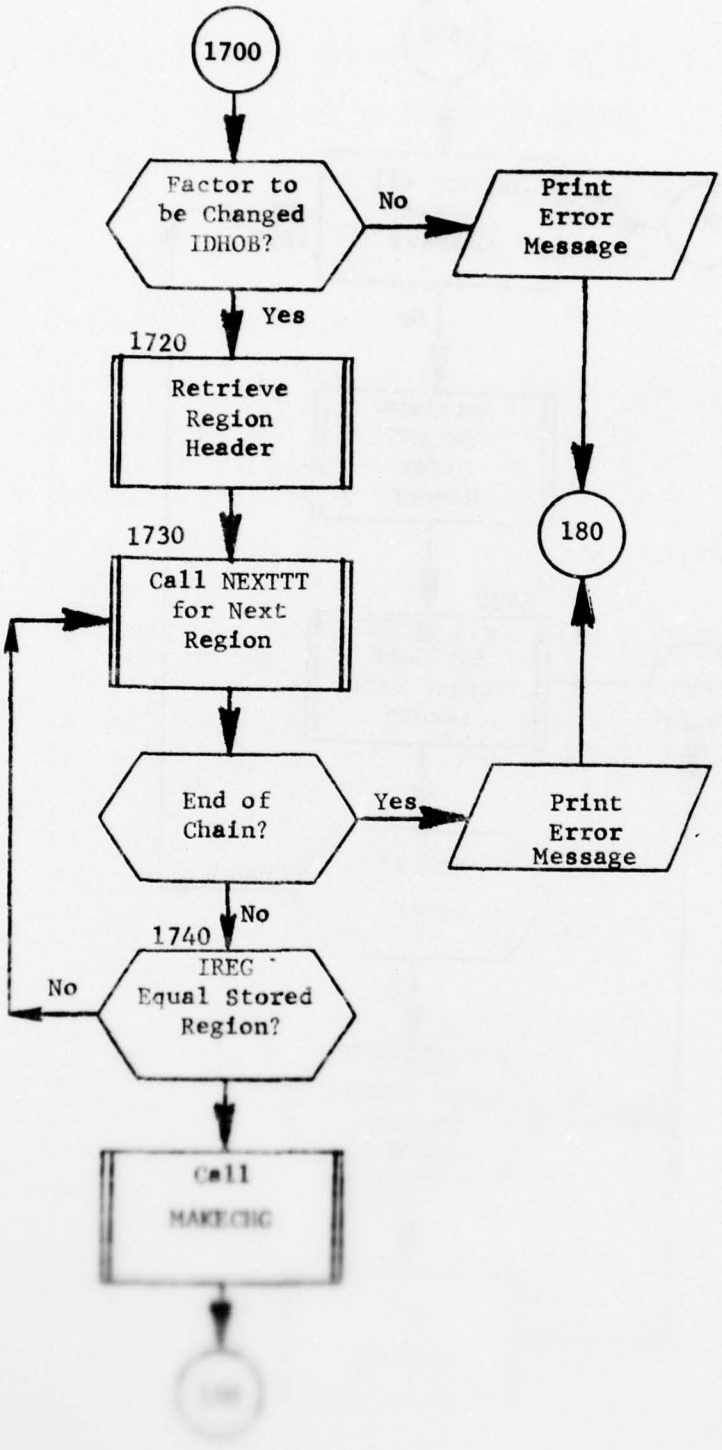
Figure 11. (Part 5 of 10)


760
Call CINSGET to Read Change Value Factor
Attribute Defining Subset of Targets Input Yet?
No
Yes
764
Was Attribute Defining Subset of Targets of Correct Level of Priority?
No
200
Yes
768
Adjust Value of Factor if Not in Correct Range


1500
1600
1550
No
Attribute Defining Subset of Targets Equal DESIG?
No
Attribute Defining Subset of Targets Equal TYPE?
180
Yes
Yes
Retrieve Target on CALC Chain
Value Input A Weapon Type?
Yes
Call SLOG to Pack IWHOB and ISPEC Arrays
No
1570
180
Done
Do 1590 For All Target Classes
Do
Retrieve Target Class Header
Yes
1580
Target Exist?
No
End of Chain?
Call NEXTTT for Next Target Type
Yes
No
Call MAKECHG
TYPE Equal Stored Target Type?
No
1580
180
Yes
Call
1585


1600
1700
1620
1650
No
Yes
Attribute Defining Subset of Targets Equal CLASS?
No
Attribute Defining Subset of Targets Equal CNTRYL?
Yes
Factor to be Changed IDHOB?
Yes
No
Target Class Exist?
No
Print Error Message
Print Error Message
Yes
1620
Retrieve Target Class Header
180
1630
Call NEXTTT for Next Target Type
End of Chain?
Yes
No
Call


1670
Do for All Target Classes
Done
180
1690
Do
Retrieve Target Class Header
1680
Call NEXTTT for Next Target Type Record
1680
End of Chain?
Yes
No
CNTRYL Equal Stored Country Location?
No
Yes


1700
Factor to be Changed IDHOB?
No
Print Error Message
Yes
1720
Retrieve Region Header
1730
Call NEXTTT for Next Region
180
End of Chain?
Yes
Print Error Message
No
1740
IREG Equal Stored Region?
No
Call
MAKECHC

2.9.3 Subroutine FIXWEP

PURPOSE: To read and process fix assignment requests

ENTRY POINTS: FIXWEP

FORMAL PARAMETERS: LOCFIX - Pointer to INSGET's arrays for location of 'FIX' adverb

COMMON BLOCKS: C10, C30, ERRCOM, NFIXES, NFIXREQ, OOPS

SUBROUTINES CALLED: CINSGET, DIRECT, HEAD, KEYMAKE, IPUT, MODFY, NFXITT, RETRV, STORE

CALLED BY: PARTA

Method:

Subroutine FIXWEP creates record type FIXASG records for all user requested fix assignments as defined by a clause introduced by the adverb FIX. Each created record stores the weapon group number and, if defined, the down time.

One word of information is defined for each weapon assigned. This word contains the group number of the weapon to be fixed in the first three characters. The lower 18 bits can be used to specify the arrival time (in minutes) of a missile weapon. If the lower 18 bits are left blank, the launch timing for the weapon will be calculated in the normal manner. If an arrival time is specified, it takes precedence over all other launch timing considerations.

The FIX clause recognizes attributes DESIG, GROUP, and the optional downtime attribute ARRIVE. Insertion of values for these two attributes (and optionally three) is sufficient for record creation. However, to provide ease of input values, the user may specify two DESIG's for one GROUP input. This mode of operation implies that a range of fixed assignments will be made and that range defined as all the targets that fall within the interval of the two DESIGs. The subroutine first reads INSGET's arrays and checks for values of necessary attributes and then, if error free, a fixed assignment record is created.

Since attributes may be defined in any order (that is GROUP appears before DESIG or vice-versa) the implementing design must delay processing until all attributes necessary are defined. The following local parameter control input processing:

- [illegible] - caused a [illegible] at the start of the input search for each [illegible] and reset to the value of the first input [illegible]

- o TERDESIG - Set to the value of the second DESIG, if it exists

- o KGROUPNO - Requested weapon group number

- o IOPTION - =1, at start of each phrase;
  =2, if a range of DESIG's was input
  =3, if ARRIVE defined
  =4, if range of DESIG's and ARRIVE defined

Fixed assignments are now stored within the data unless:

a. If an attempt is made to fix a bomber weapon on a target with more than 30 fix requests, the request will be ignored, since only targets with terminal ballistic missile defenses undergo a saturation m ssile attack. The allocation procedure will not allow a bomber to participate in such an attack

b. If an attempt is made to fix more weapons than are present in a group, the excess requests are ignored

c. If two DESIGs were input and the alpha portions did not match, the requests are ignored

d. If non-lead (either for complexes or multiplier) targets were to be fixed and the representative target could not be found, the request is ignored.

e. If the target record or if the group number is invalid, the request is ignored.

Subroutine FIXWEAP is illustrated in figure 12.

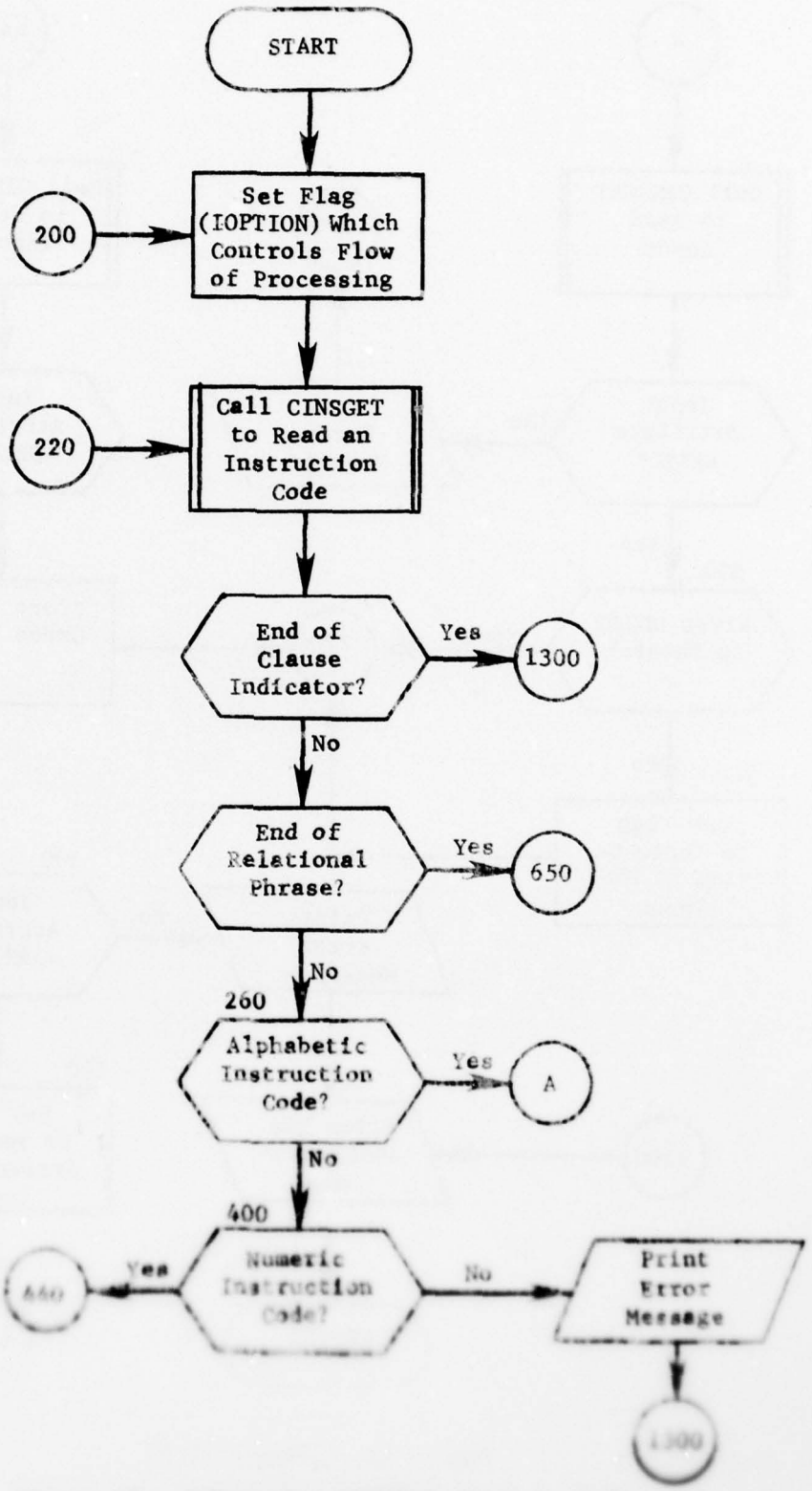


Figure 12. Subroutine FINMAP (Part 1 of 6)

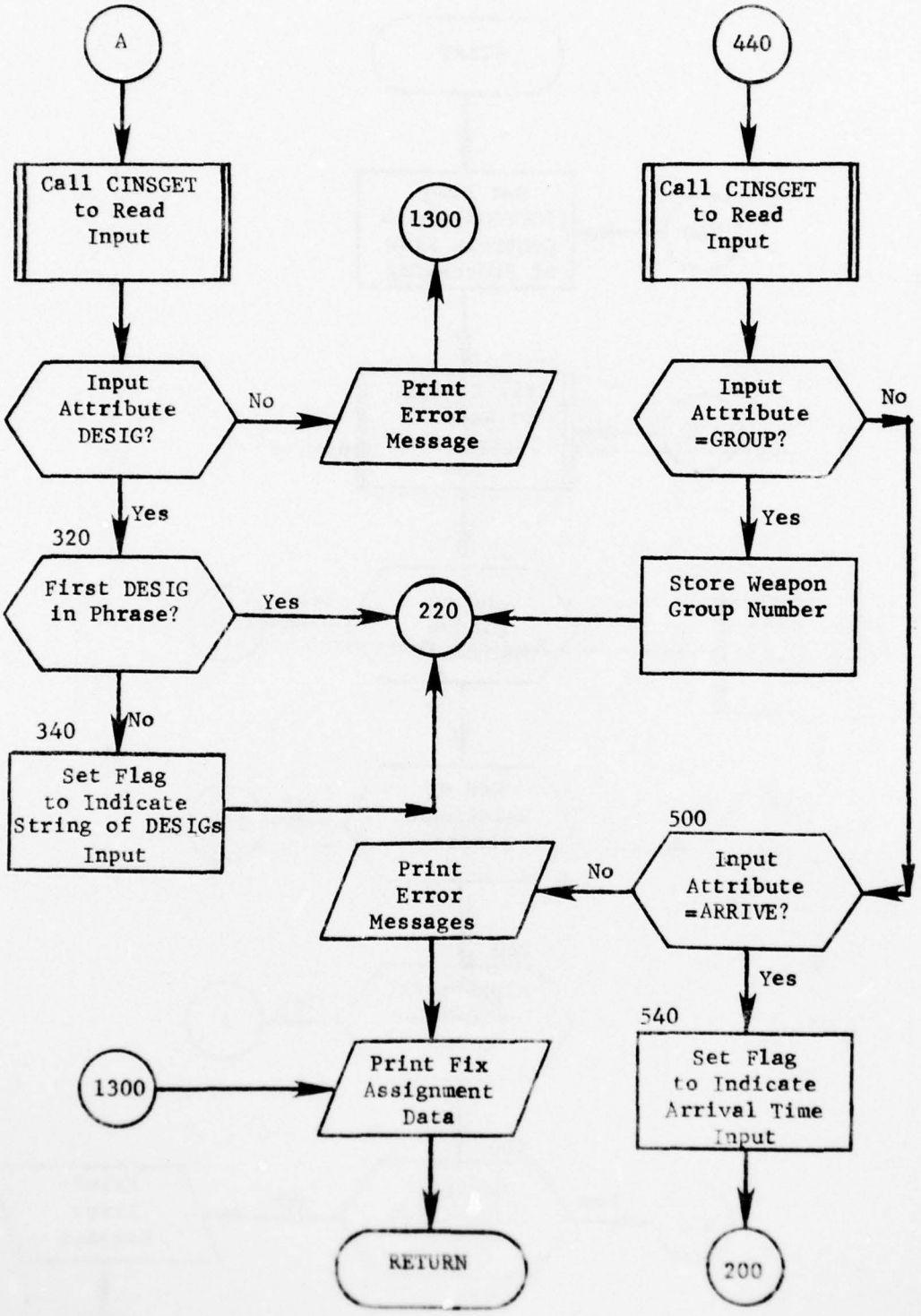


Figure 12. (Part 2 of 6)

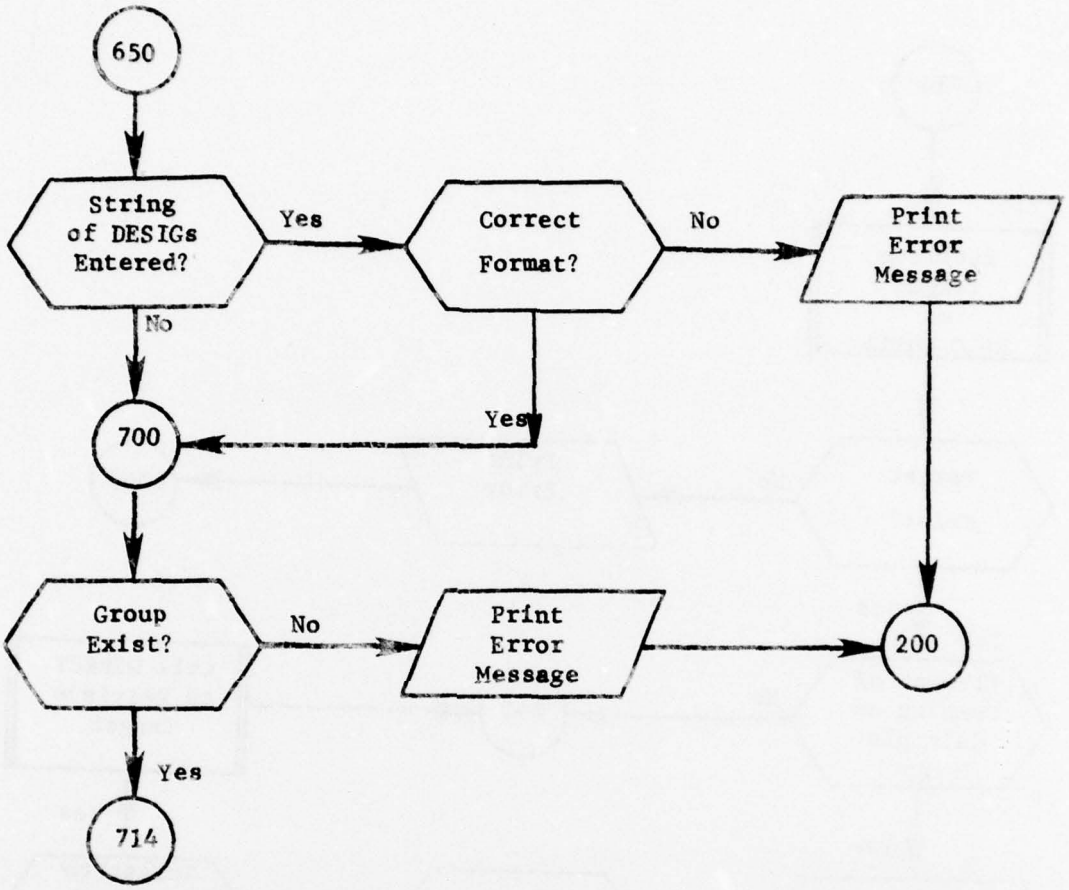


Figure 12. (Part 3 of 6)

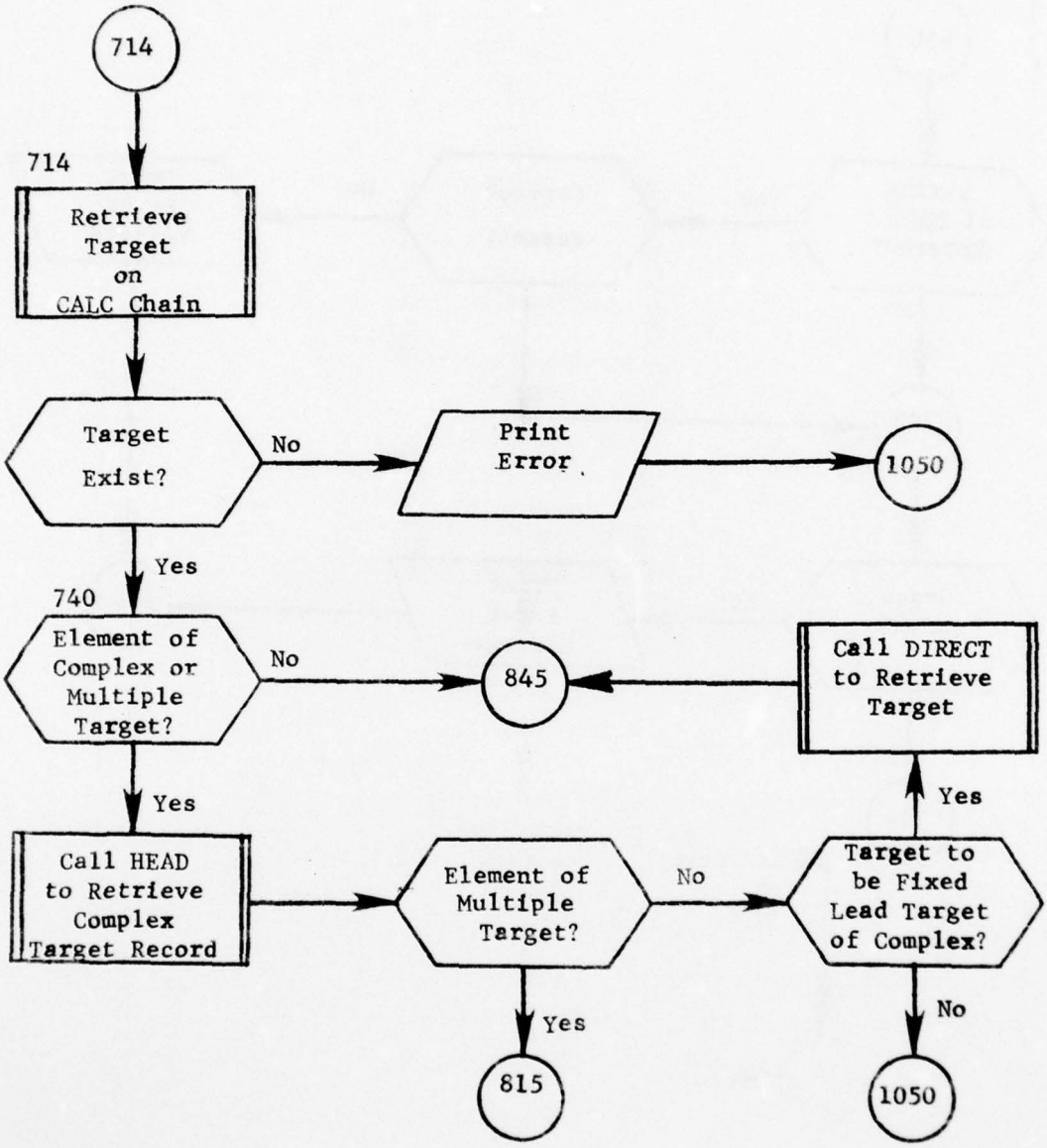


Figure 12. (Part 4 of 6)

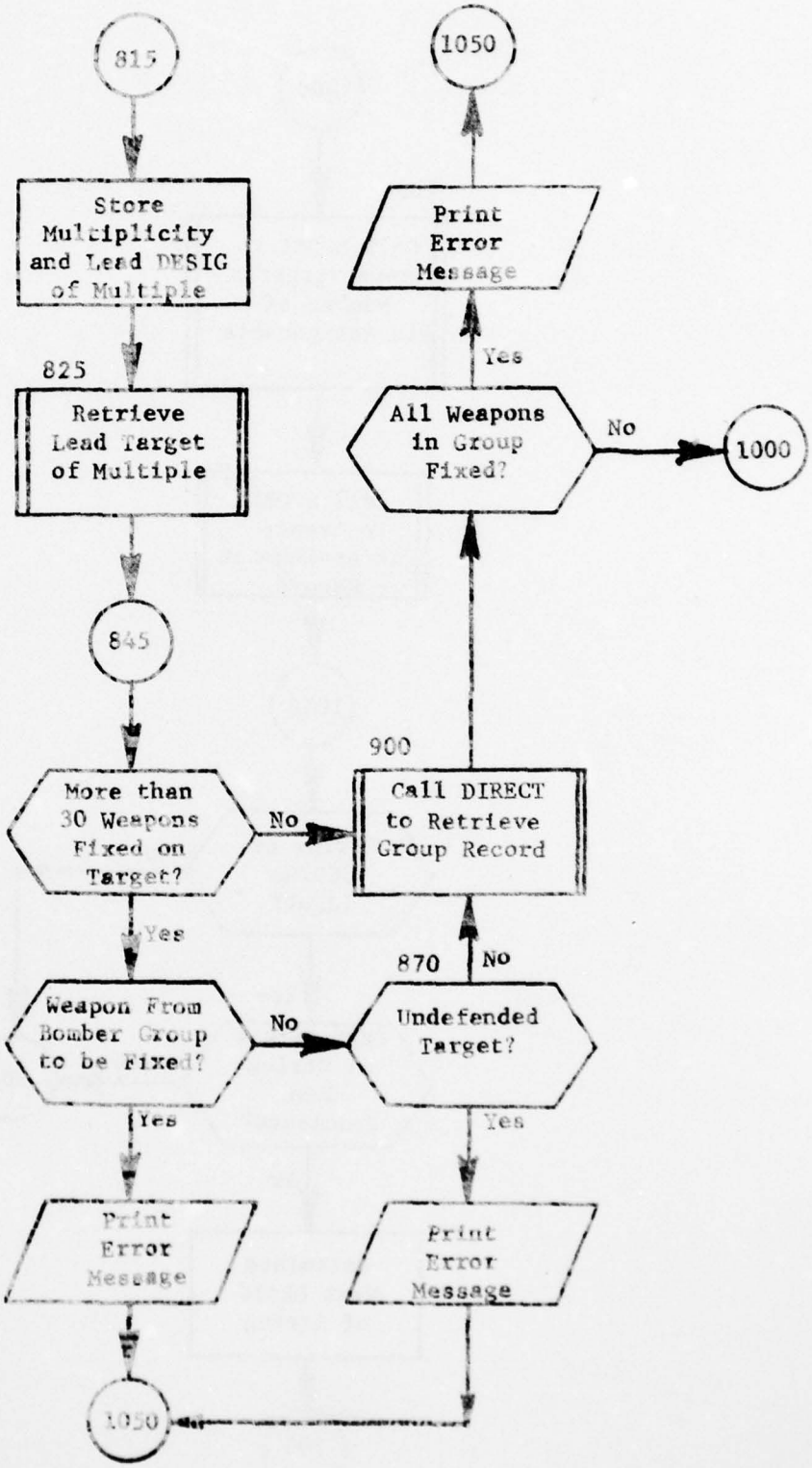


Figure 12. (Part 5 of 6)

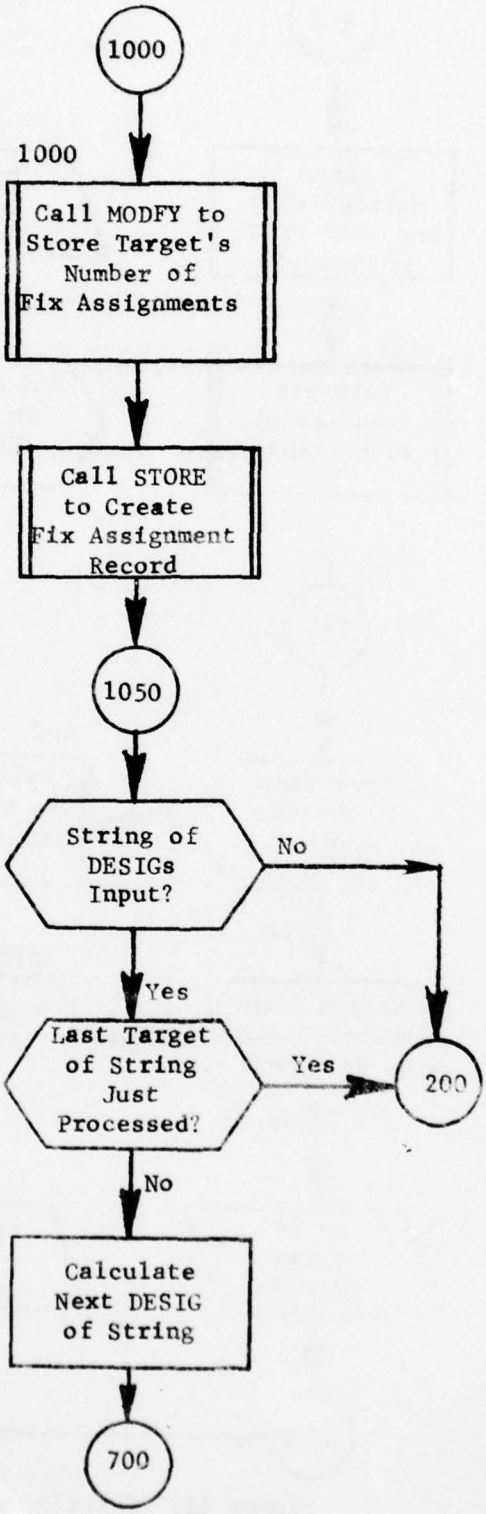


Figure 12. (Part 6 of 6)

2.3.4 Subroutine MAKECHG

PURPOSE: To make changes to the data base for valid factor change requests

ENTRY POINTS: MAKECHG

FORMAT PARAMETERS:
ICRIT - Indicates type of subset of targets to be changed (1, 2, 3, 4, 5 for region, country location, class, type, single target, respectively)
CRWORD - Value of subset
IFACTOR - Indicates attribute to be changed (1, 2, 3, 4 for VALUE, MINKILL, MAXKILL, height of burst, respectively)
FACWORD - New value of attribute

COMMON BLOCKS: C10, C30, SUMNEW

SUBROUTINES CALLED: DIRECT, HEAD, MODFY, NEXTTT

CALLED BY: FACTORCG

Method:

Subroutine MAKECHG modifies target record(s) as directed by subroutine FACTORCG. Depending upon the level of processing (parameter ICRIT) either the TGTTGT or TGTREC chain is queried. When FACTORCG calls MAKECHG the correct next highest IDS record has been defined. By stepping through the targets, modification is accomplished.

If a factor is changed on the main target of a complex, the factors for each component are changed appropriately. For example, if the value of the complex is doubled, the value of each component is doubled. This procedure is followed to allow the user to change one factor for the entire complex and other factors by component. If a factor is changed for a complex component individually (not through the main target) that factor is checked on all the remaining components. The method used in determining VALUE, MINKILL, and MAXKILL for a complex target is identical to that used in subroutine CALCOMP of module PLANSET. For height-of-burst specifications, only the main target of a complex is checked.

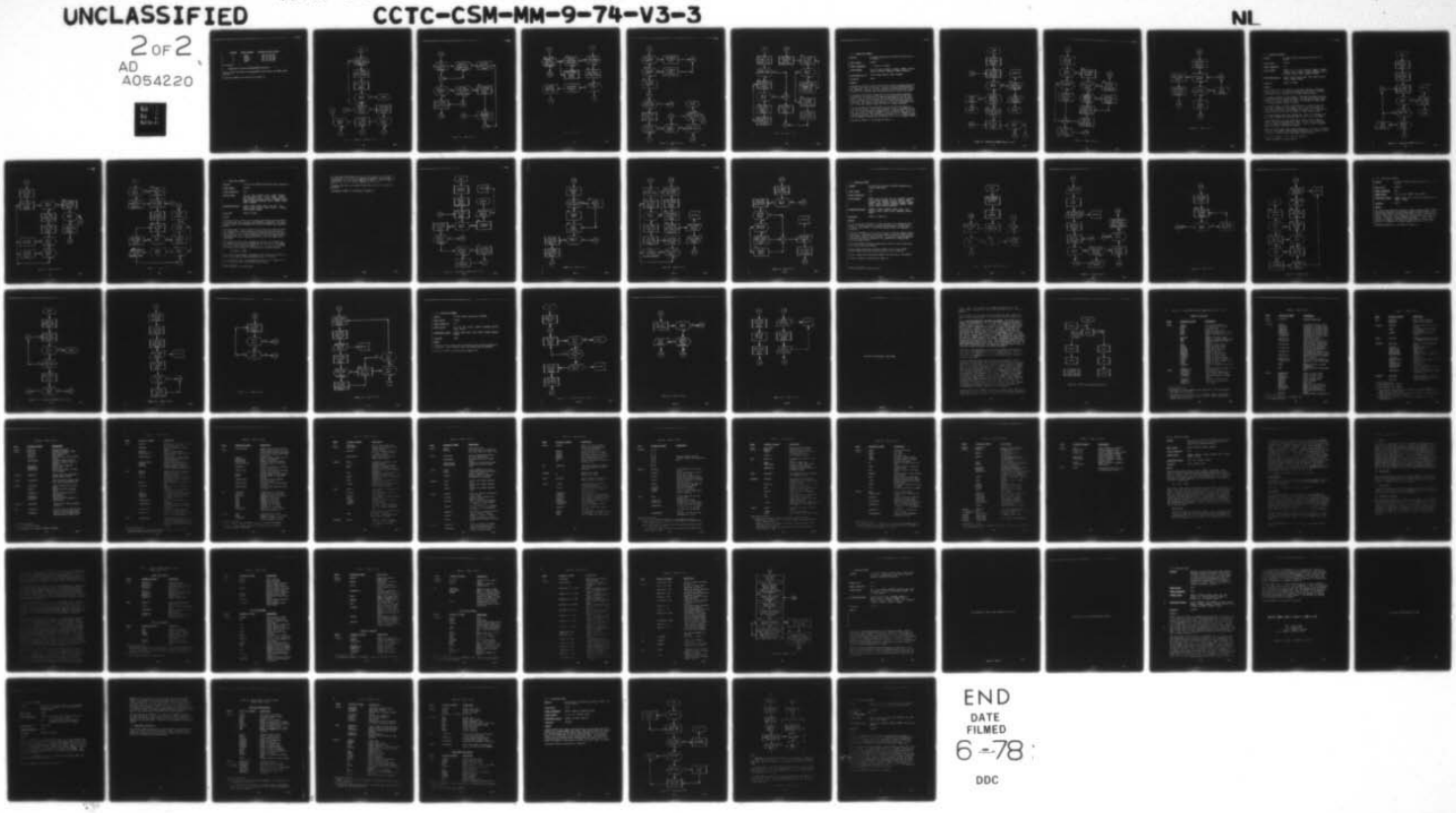
If attribute VALUE is the factor to be updated, parameter SUMNEW is redefined for further normalizing calculations.

Whenever a factor is changed for a target, the attribute IDHOB is 'marked' so that the same factor is not changed again by a change request of lower priority. IDHOB is 'marked' by packing a 1 in an octal digit as shown below:

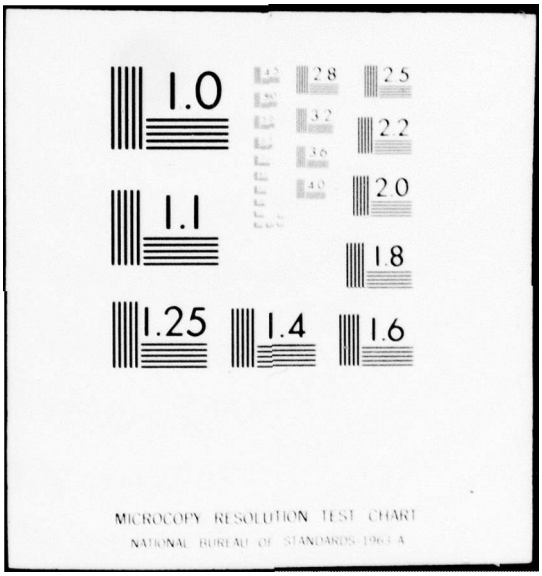
AD-A054 220 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON D C F/G 15/7
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK) PROGR--ETC(U)
APR 78

UNCLASSIFIED CCTC-CSM-MM-9-74-V3-3 NL

2 OF 2
AD
A054220



END
DATE
FILMED
6-78
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

IFACTOR	Factor Changed	Position of Octal Digit
1	VALUE	000 001 000 000
2	MINKILL	000 010 000 000
3	MAXKILL	000 100 000 000
4	IDHOB	001 000 000 000

The formula

```
IDHOB/MSHIFT(IFACTOR)-8*IDHOB/MSHIFT(IFACTOR+1)
```

merely yields the value to the appropriate octal digit (of IDHOB) given IFACTOR.

Subroutine MAKECHG is illustrated in figure 13.

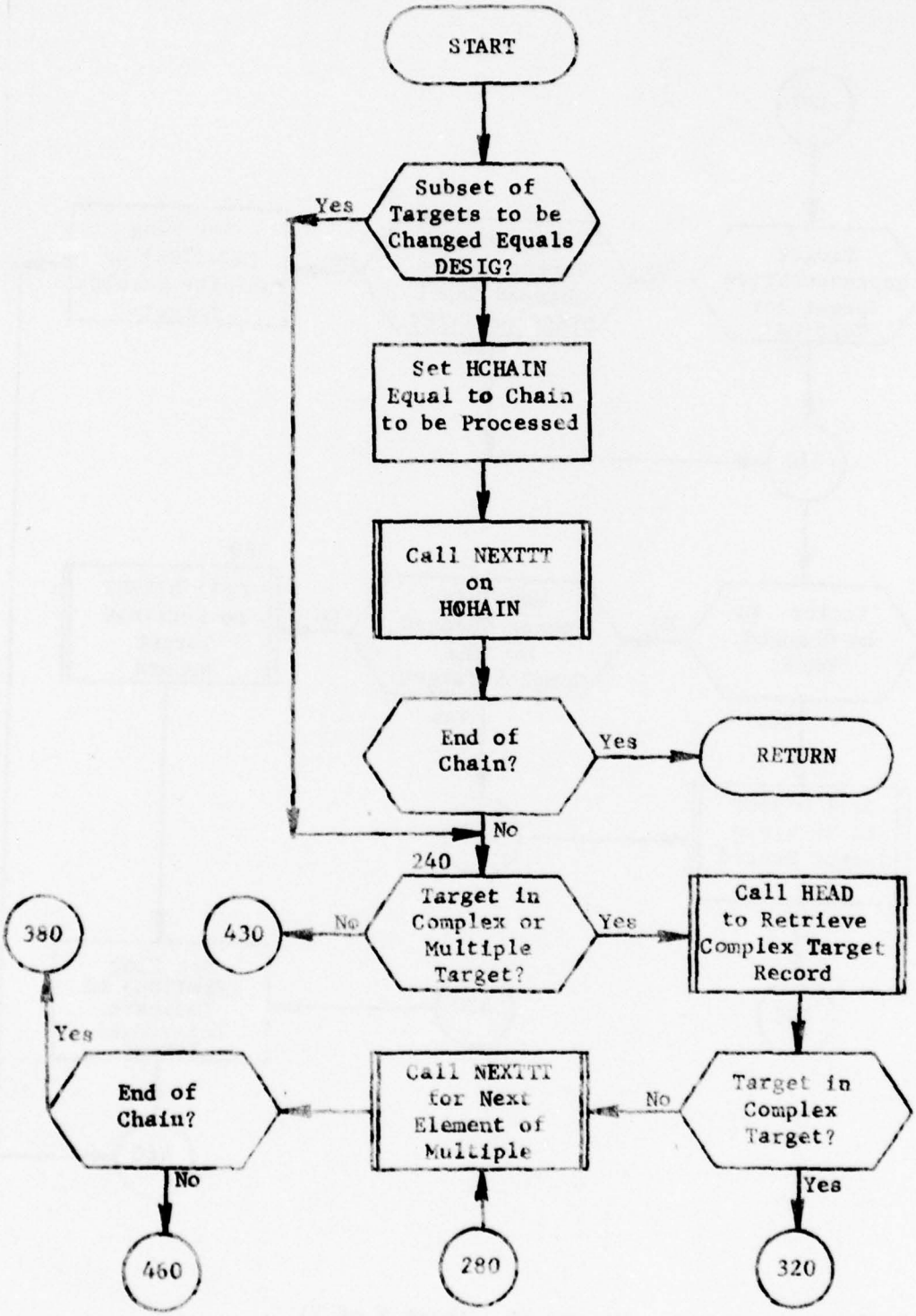


Figure 13. Subroutine MAKECHG (Part 1 of 5)

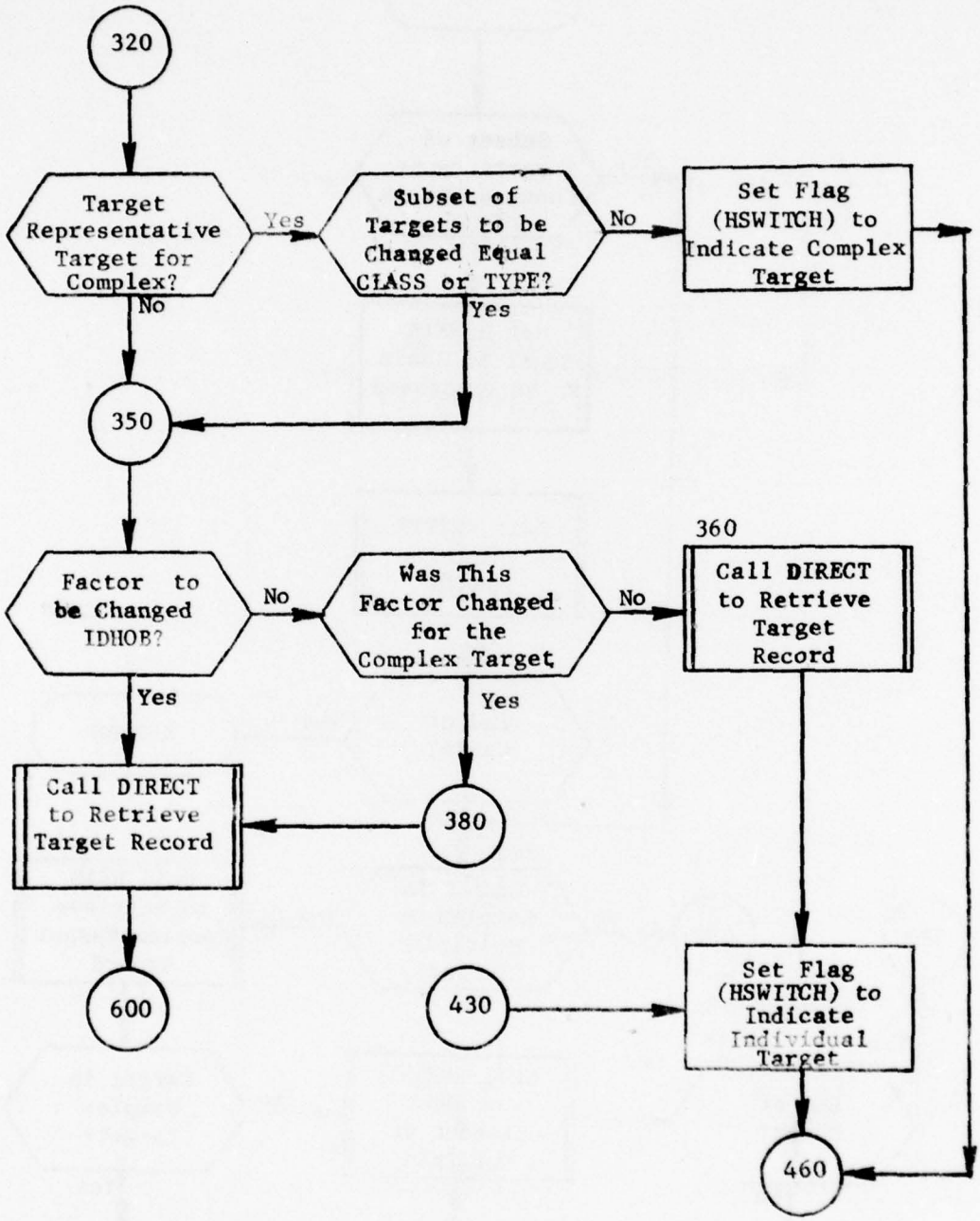


Figure 13. (Part 2 of 5)

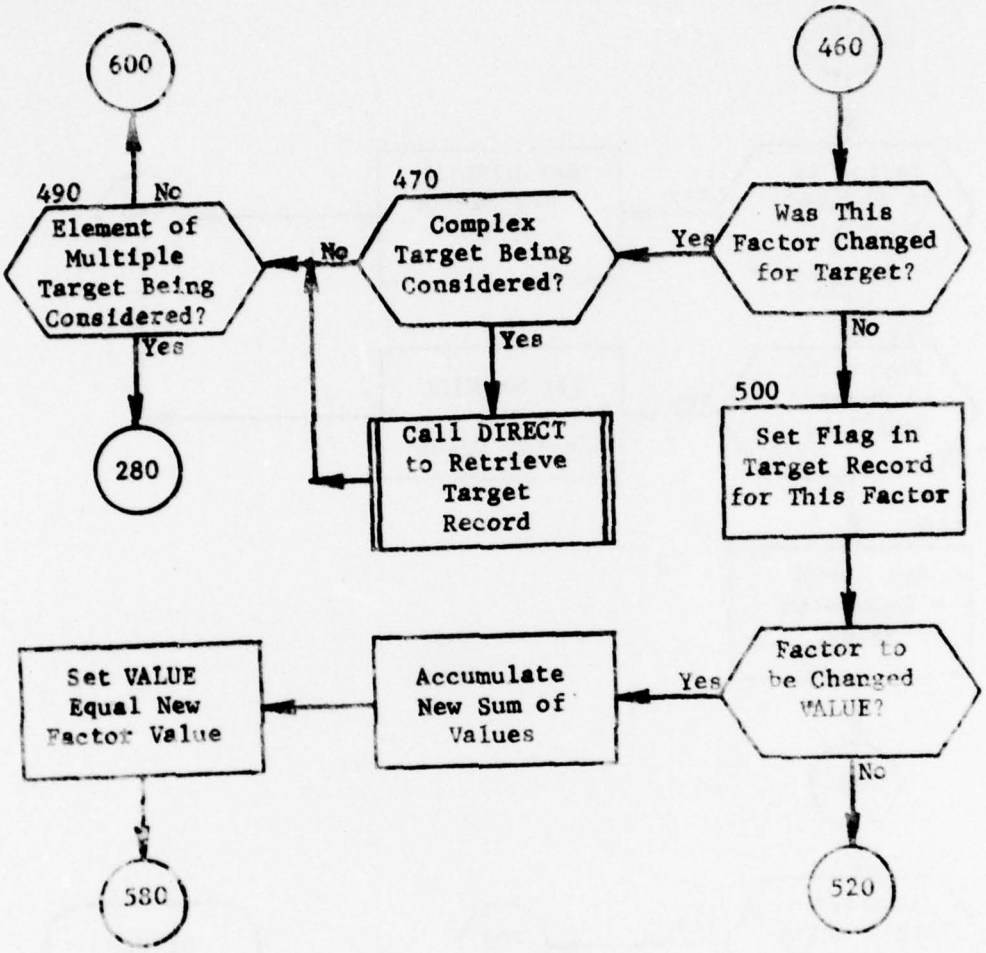


Figure 13. (Part 3 of 5)

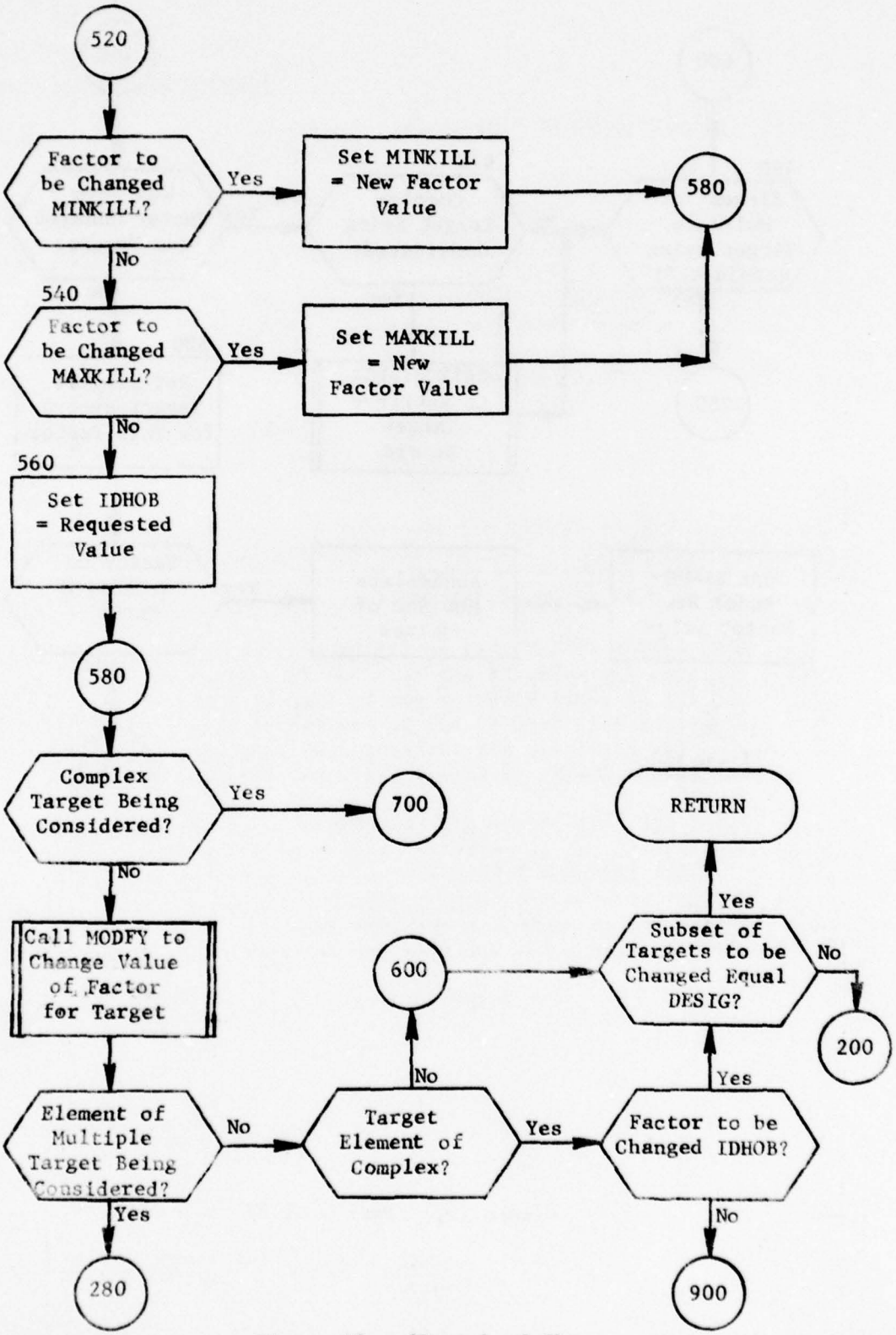


Figure 13. (Part 4 of 5)

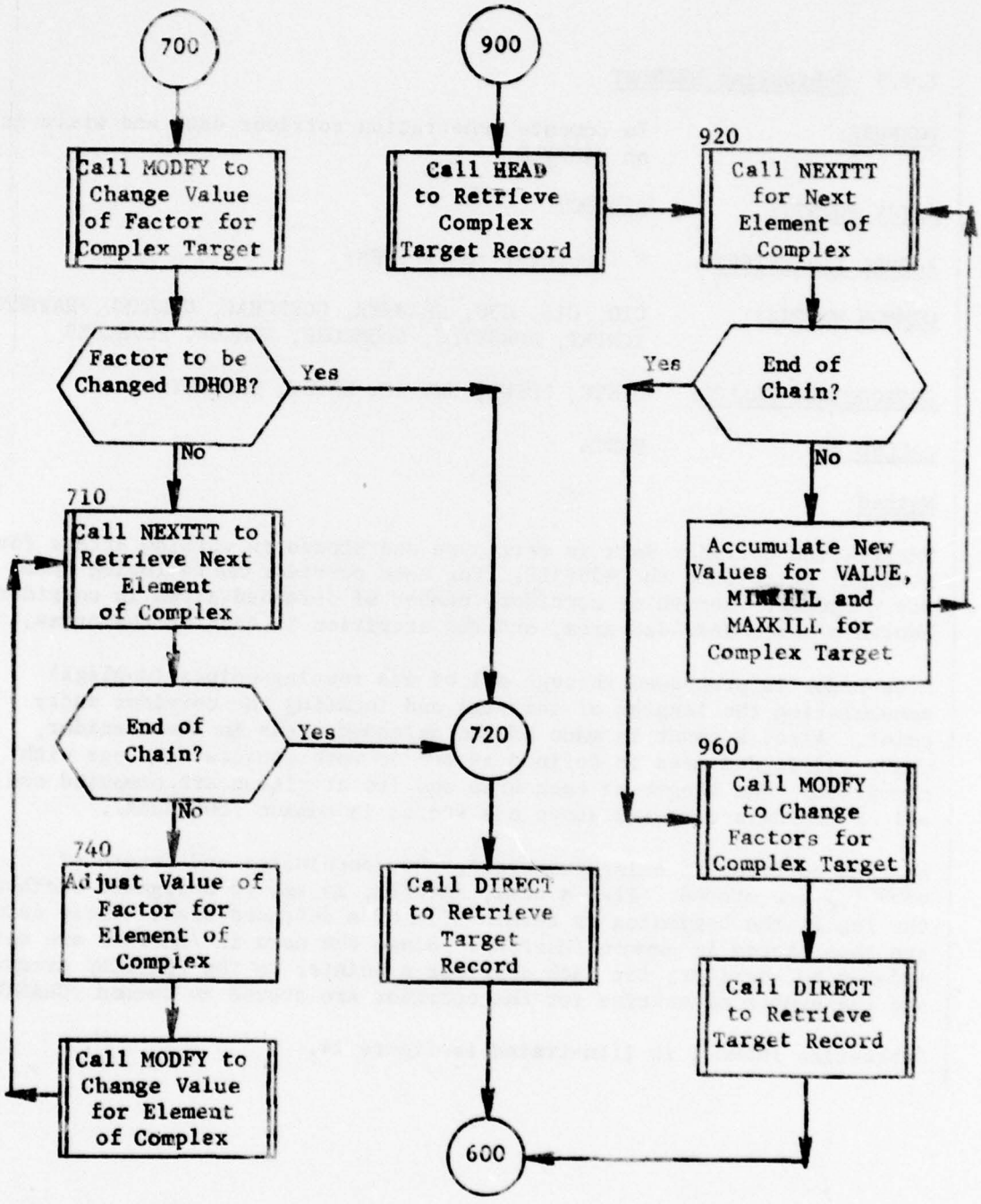


Figure 13. (Part 5 of 5)

2.9.5 Subroutine PENROUT

PURPOSE: To compute penetration corridor data and write it on BASFILE

ENTRY POINTS: PENROUT

FORMAL PARAMETERS: N - Pointer to /HAPPEN/

COMMON BLOCKS: C10, C15, C30, CHARTER, CORRCHAR, CORRTWO, HAPPEN, IONPRT, KORSTYLE, NDUMCORR, NUMCOR, PCVALUES

SUBROUTINES CALLED: DISTF, HDFND, NEXTTT, RETRV, WRARRAY

CALLED BY: PARTA

Method:

Penetration corridor data is retrieved and stored in working arrays for eventual write onto the BASFILE. For each corridor the following data are computed: length of corridor, number of defended areas in corridor, length of each defended area, and the attrition in each of the areas.

A corridor is processed through all of its routing points (doglegs) accumulating the lengths of the legs and locating the corridor entry point. Also, a count is made of the defended areas in the corridor, where a defended area is defined as one or more consecutive legs with attrition. The length of each area and its attrition are computed and all of the data computed above are stored in common /CORRCHAR/.

As each corridor is being processed, the coordinates and length of each leg are stored. Also a cell, JAPTYPE, is set to designate whether the leg is the beginning or termination of a defended area. These data are then stored in common /HAPPEN/. Since the data in /HAPPEN/ are not indexed by corridor, for each corridor a pointer to the /HAPPEN/ arrays and the number of entries for the corridor are stored in common /CHARTER/.

Subroutine PENROUT is illustrated in figure 14.

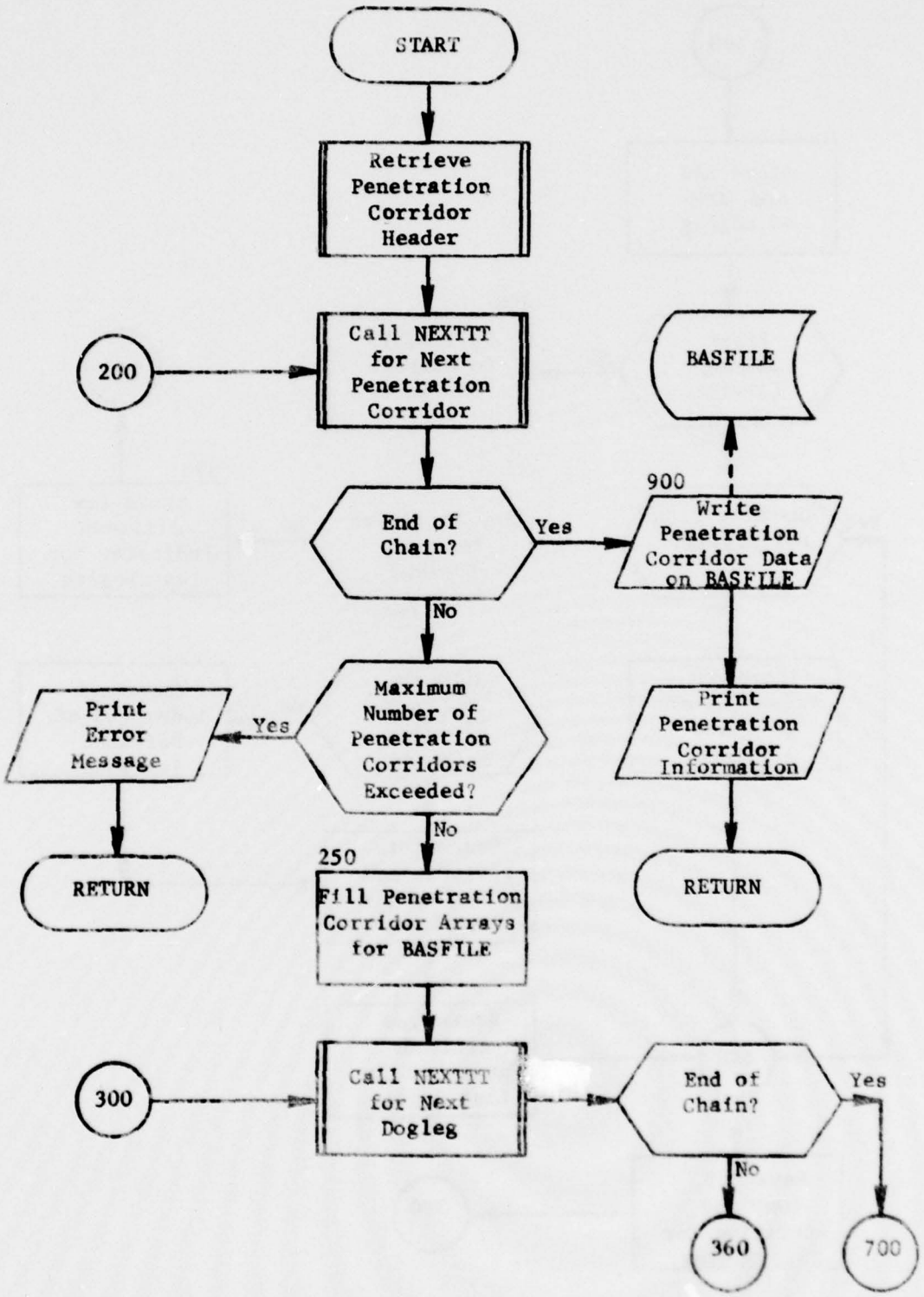


Figure 14. Subroutine PENROUT (Part 1 of 3)

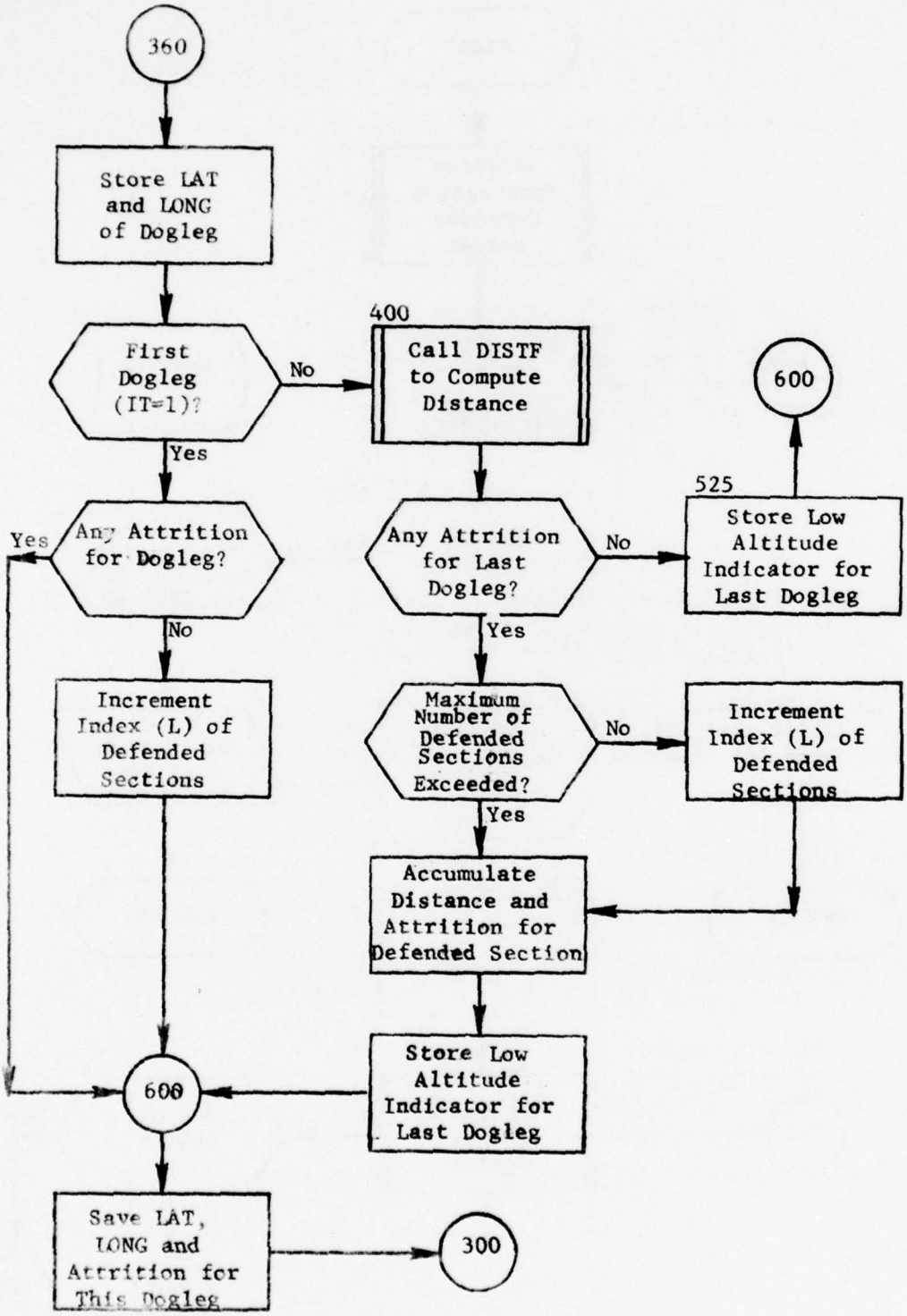


Figure 14. (Part 2 of 3)

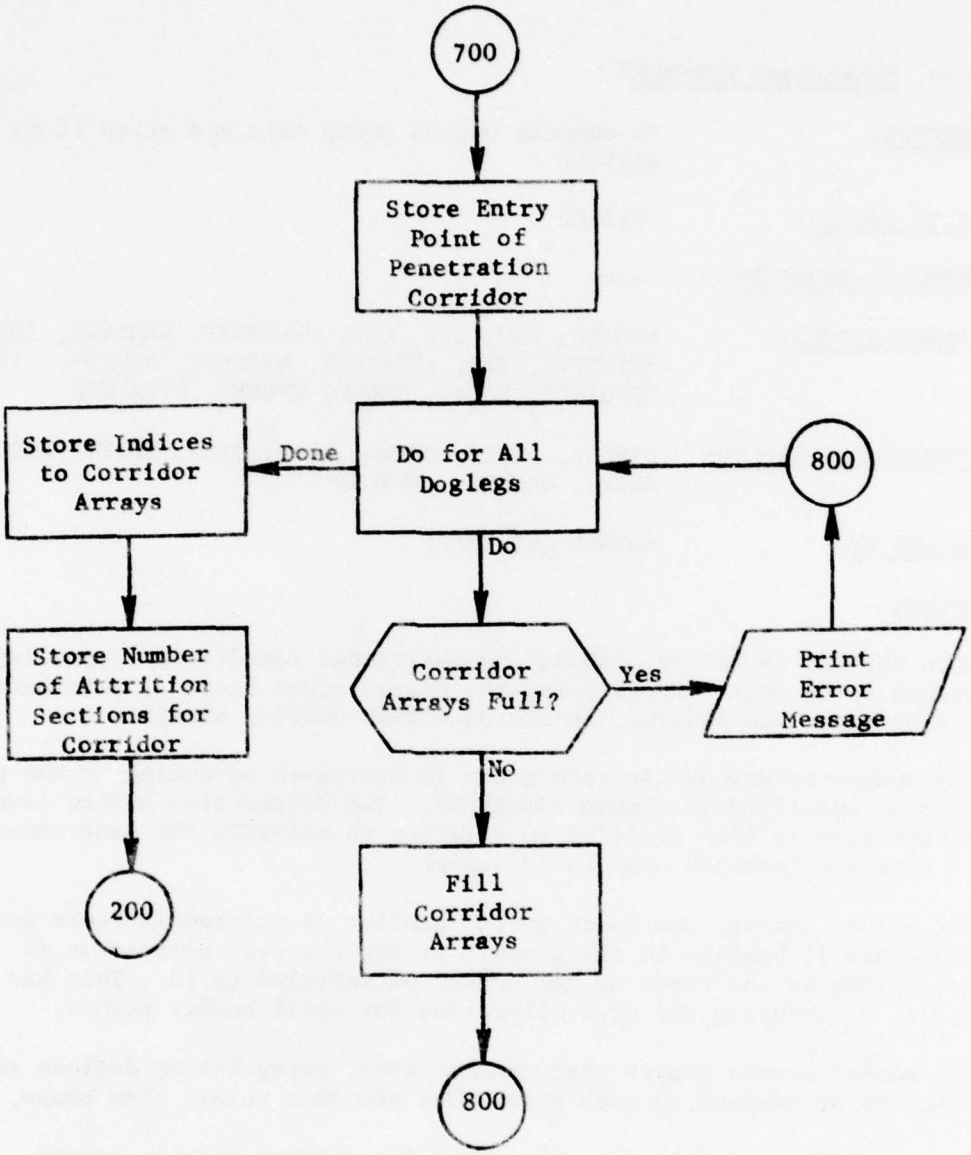


Figure 14. (Part 3 of 3)

2.10 Subroutine WEPPREP*

PURPOSE: To compute weapon group data and write it on BASFILE

ENTRY POINTS: WEPPREP

FORMAL PARAMETERS: None

COMMON BLOCKS: ASMTYP, C10, C15, C30, CRLENGTH, GAMEVAR, IONPRT, ISIMTYPE, ITP, LHVALUES, MYIDENT, NUMCOR, PAYTYPE, RFPOINTS, SALVO, TWORD, WPNGRP, WPVALUES

SUBROUTINES CALLED: DIRECT, DISTF, HDFND, HEAD, ITLE, MODFY, NEXTTT, RETRV, WRARRAY, WRWORD

CALLED BY: ENTMOD (of PREP)

Method:

When WEPPREP is called, BASFILE is positioned immediately preceding weapon group data. By chaining the weapon group records, attributes are obtained and information stored within working arrays.

The number of weapons in each group is increased according to the propostion specified in common /GAMEVAR/. The destruction before launch probability is then modified by a factor to maintain the same number of expected launched weapons as before.

For bomber groups, the basic overallocation is reduced if there are less than 15 bombers in the group. If so, the over-allocation is multiplied by the ratio of the number of vehicles to 15. This has the effect of reducing the over-allocation for small bomber groups.

For bomber weapon groups that contain ASMs, array EXPASM defines the fraction of weapons in each group that are ASMs rather than bombs.

Arrays MAXSLV and NSAL are calculated for salvoed missile groups. MAXSLV contains the maximum salvo number for each salvoed group and NSAL the number of weapons in each salvo. The NSAL array is packed into three words, four bits per salvo.

Upon storing all data, weapon group information is written onto BASFILE followed by parameters that described the percentage of weapon overallocation and, finally, naval weapon group data is written.

Subroutine WEPPREP is illustrated in figure 15.

*Main subroutine of overlay link C.

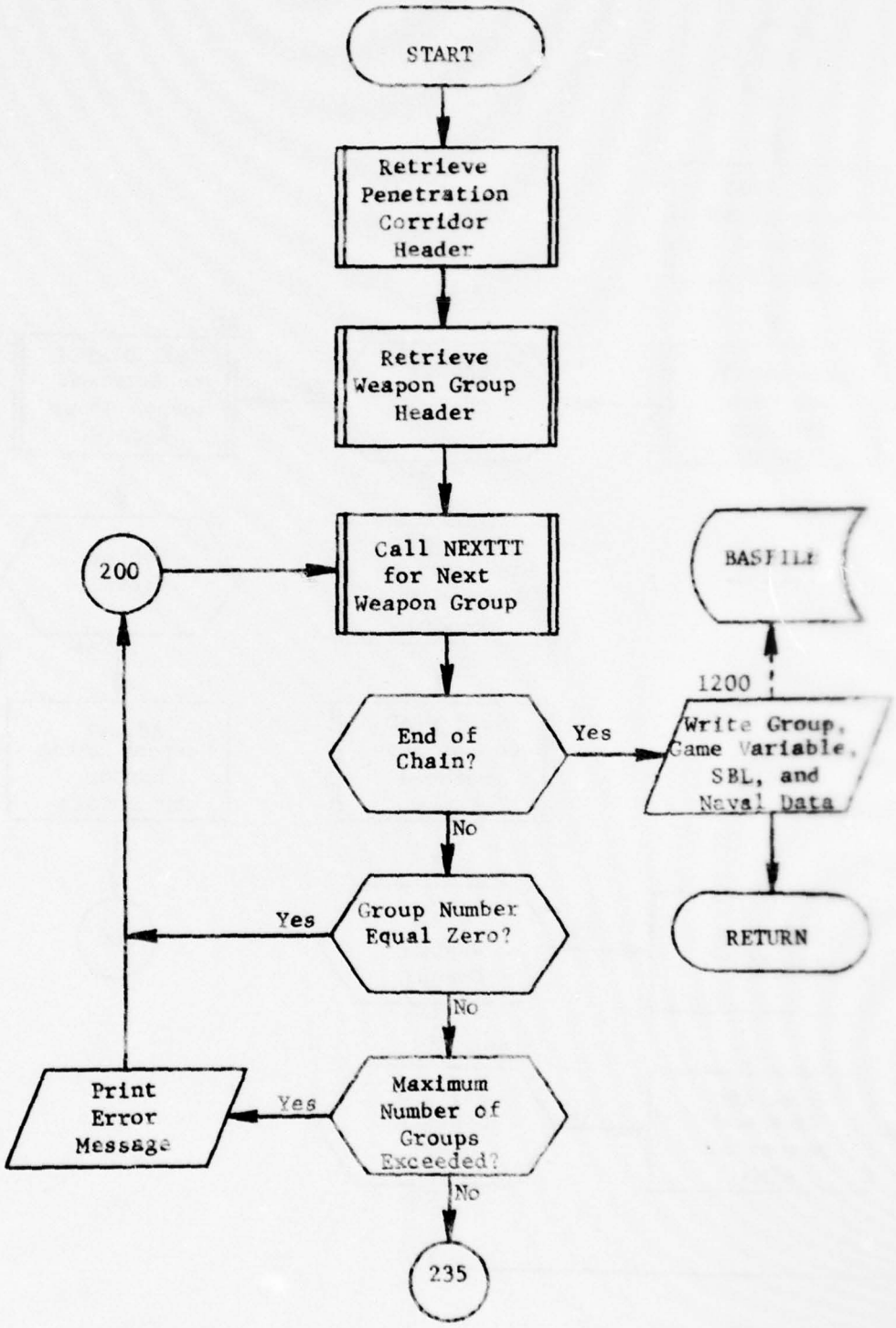


Figure 15. Subroutine WEPPREP (Part 1 of 3)

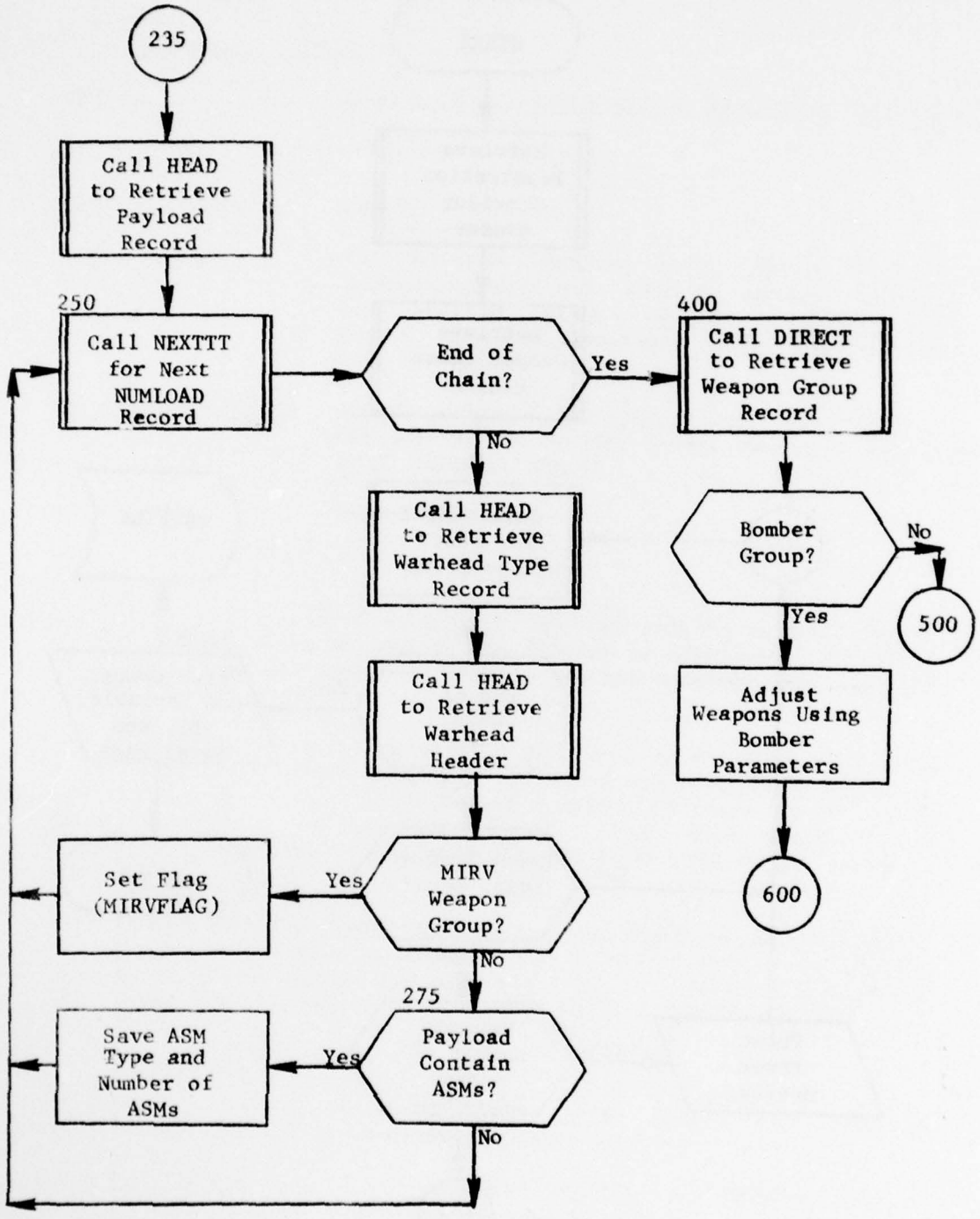


Figure 15. (Part 2 of 3)

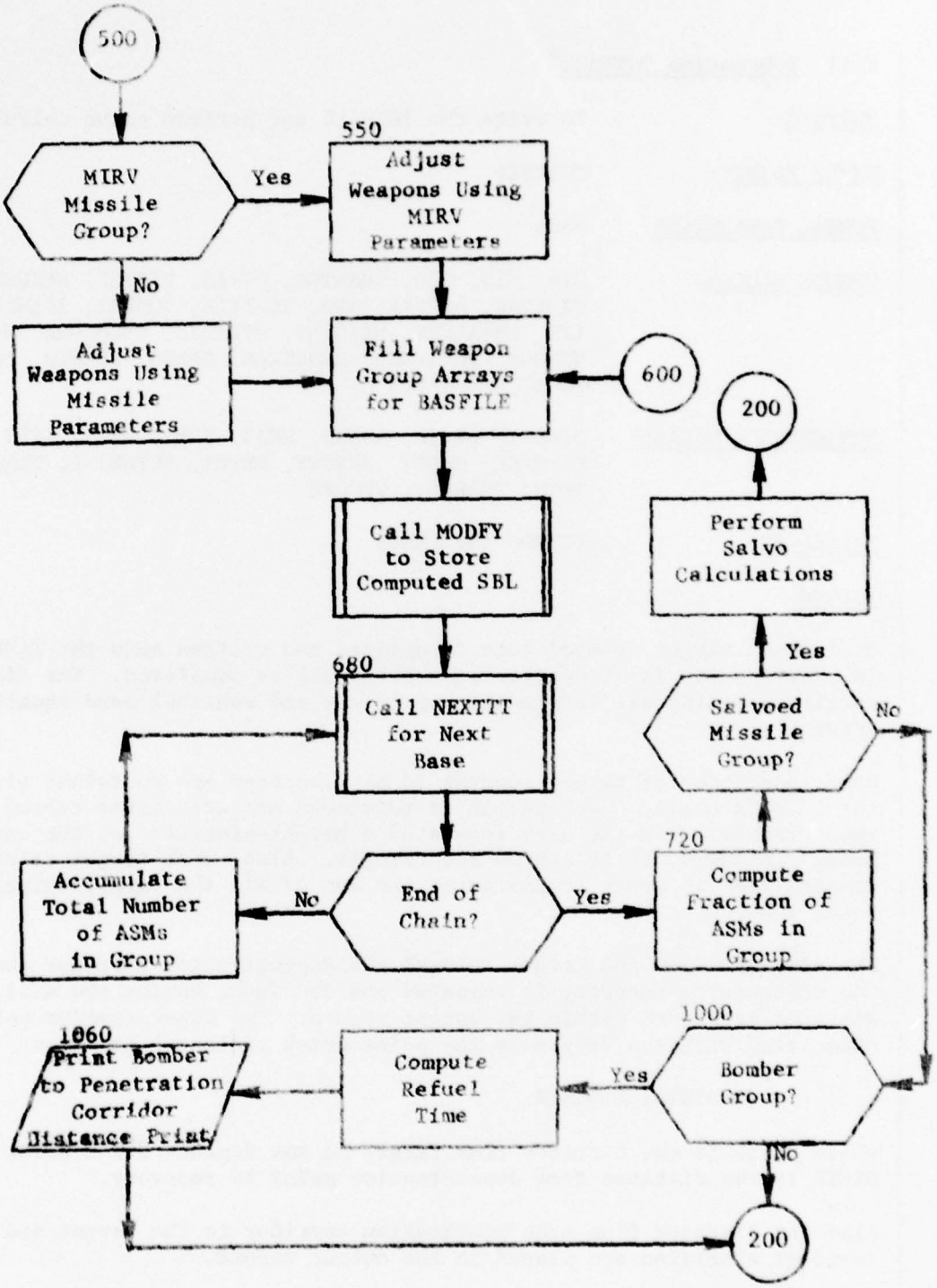


Figure 15. (Part 3 of 3)

2.11 Subroutine TGTPREP*

PURPOSE: To write the TGTFILE and perform salvo calculations

ENTRY POINTS: TGTPREP

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, CHARTER, CVTAB, DISTEF, DPENREF, GAMEVAR, HAPPEN, HOB, IIMFILE, IONPRT, ISIMTYPE, ITP, LHVALUES, MYIDENT, MYLABEL, NDUMCORR, NFIXES, NUMCOR, PCVALUES, RANGERAR, SALVO, SUMNEW, TARGET, TWORD, WPVALUES

SUBROUTINES CALLED: DIRECT, DISTF, HDFND, HEAD, IGET, IPUT, ITLE, KAYMAKE, MODFY, NEXTTT, RETRV, SETWRTS, TERMTAPE, TOFM, WRARRAY, WRWORD

CALLED BY: ENTMOD (of PREP)

Method:

Individual target related data is defined and written onto the TARFILE. In addition the first section of the BASFILE is completed. The first section contains all information up to the end sentinel word equalling RRRRRR.

Reference Codes of target records to be processed are contained within the LIXSTXX chain. Each target is retrieved and attributes stored into /TARGET/. If the user requested a height-of-burst for the target being processed, it is stored accordingly. Also, each target value is renormalized in order to guarantee the sum of all the target values equal 1000.

The distance from the target through the depenetration corridor and the distance to recovery is computed and the index having the minimum distance is stored within the target record. The depenetration point associated with the target is the point which minimizes the sum:

$$(2*DISTD) + DISTR$$

where DISTD is the distance from target to the depenetration point and DISTR is the distance from depenetration point to recovery.

Also the distance from each penetration corridor to the target and the corridor attrition are placed in the output record.

*Main subroutine of overlay link D.

If there are fixed weapon requests for the target being processed, the MYASGN chain is queried for definition of specified downtimes and, if applicable, the salvo launch number is determined. Fixed assignment data is also written onto the TARFILE for each target.

Finally, the remaining portion of the first section of the BASFILE is written.

Subroutine TGTPREP is illustrated in figure 16.

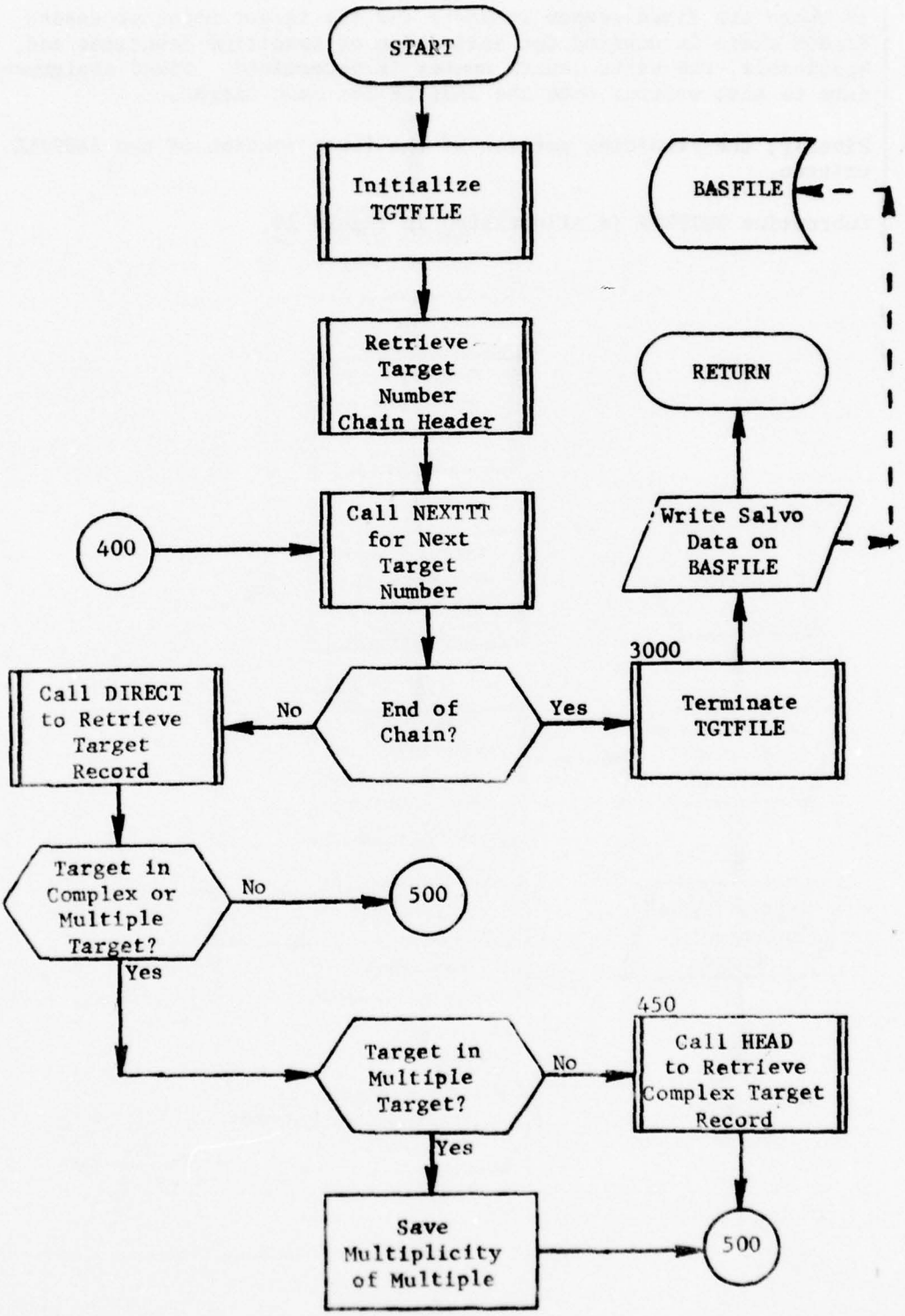


Figure 16. Subroutine TGTPREP (Part 1 of 4)

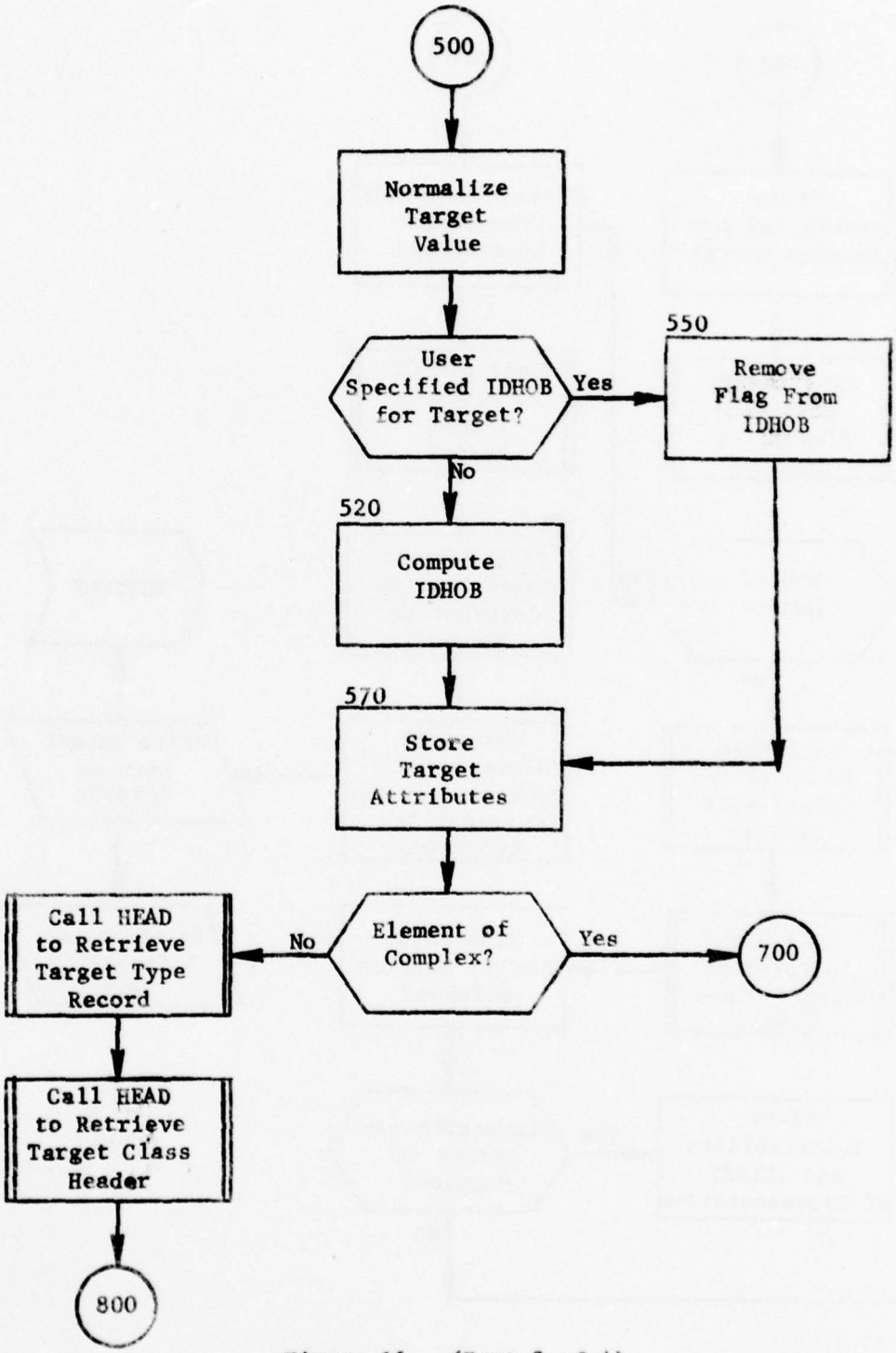


Figure 16. (Part 2 of 4)

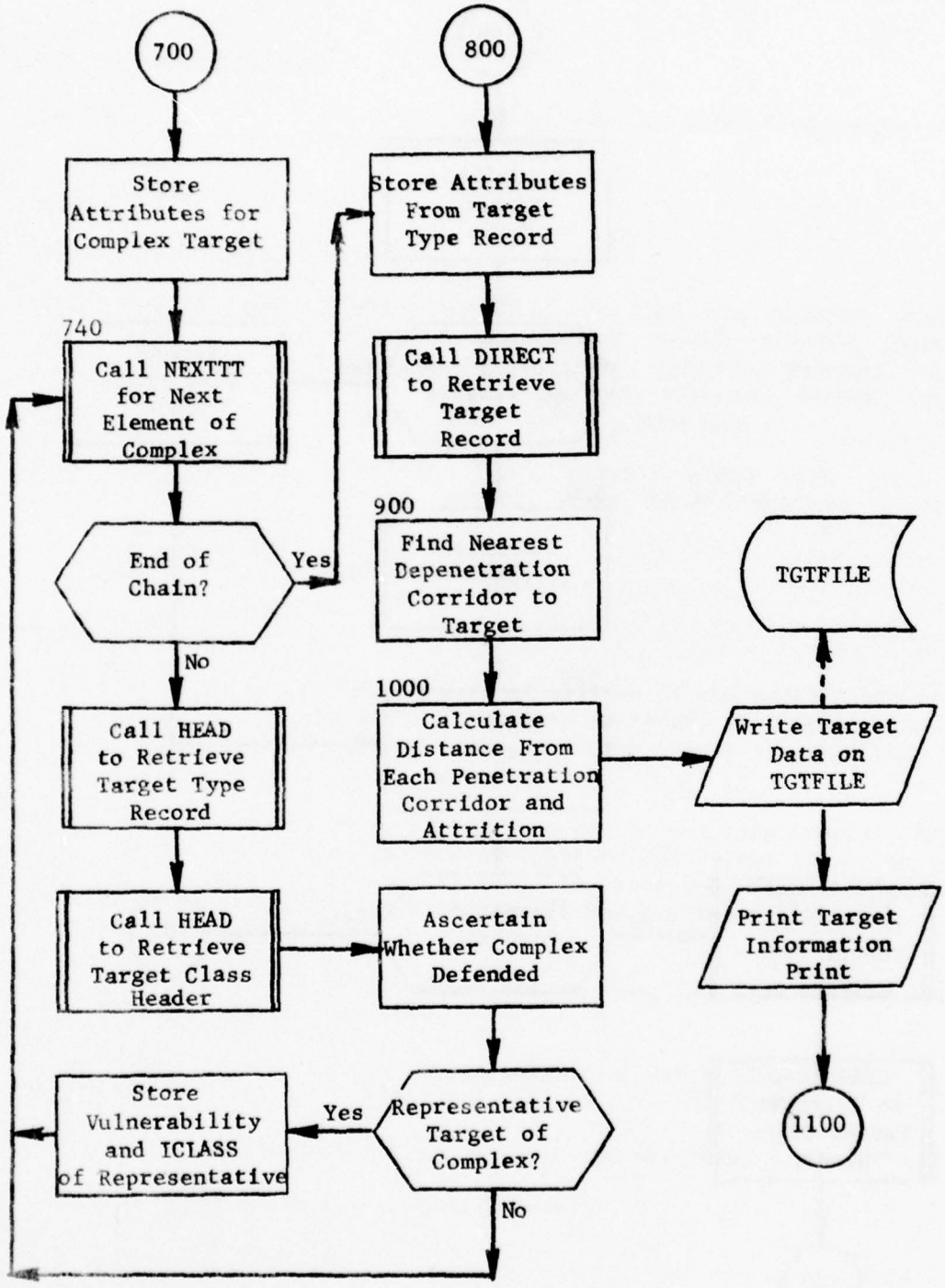


Figure 16. (Part 3 of 4)

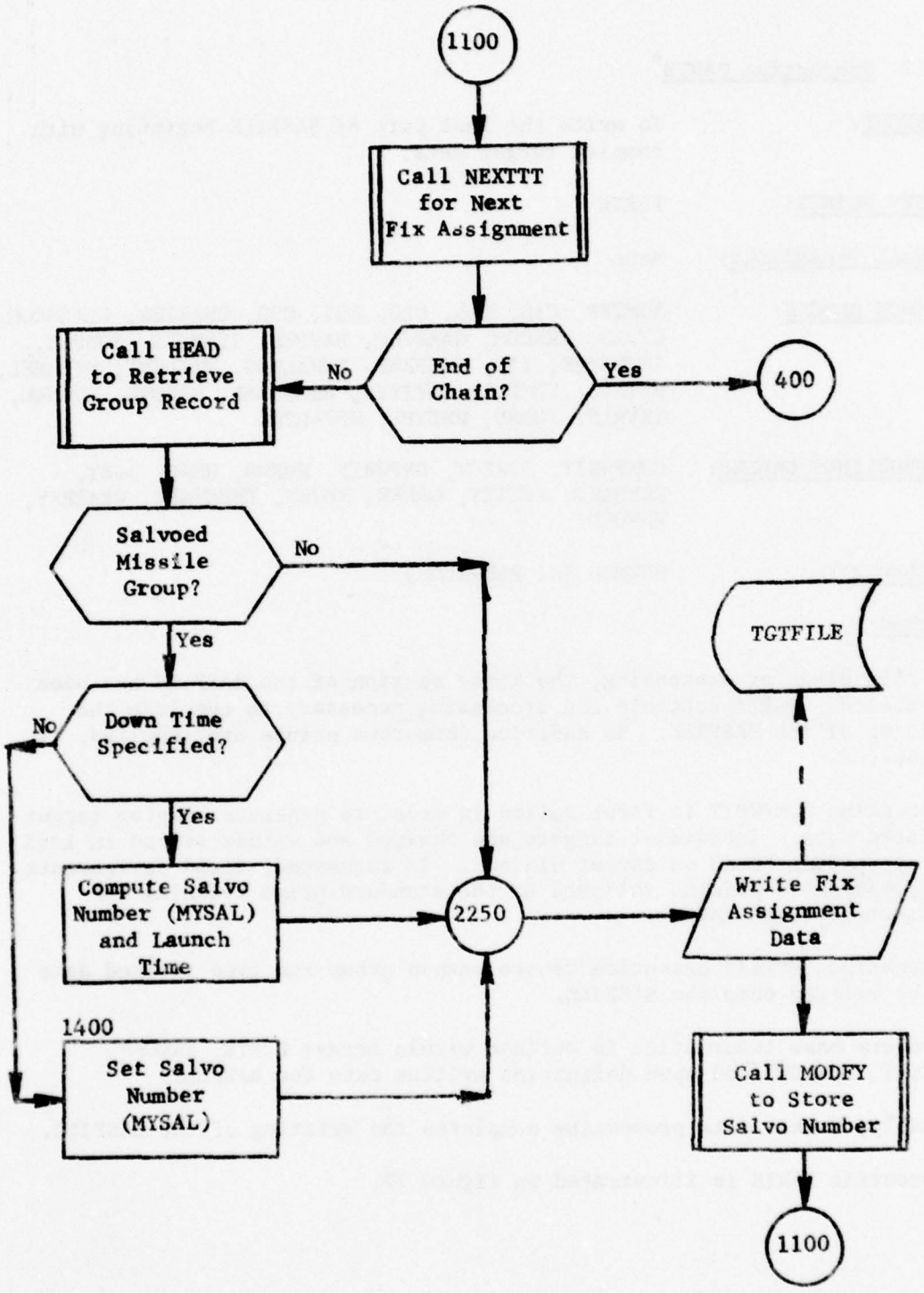


Figure 16. (Part 4 of 4)

2.12 Subroutine PARTB*

PURPOSE: To write the last part of BASFILE beginning with complex target data.

ENTRY POINTS: PARTB

FORMAL PARAMETERS: None

COMMON BLOCKS: ASMTYP, C10, C15, C20, C25, C30, CHARTER, CLASSCOM, CVTAB, ERRCOM, GAMEVAR, HAPPEN, IIMFILE, IONPRT, ISIMTYPE, ITP, IWEPREF, LHVALUES, MYIDENT, MYLABEL, NFIXES, NUMCOR, PAYTYPE, RANGERAR, RECOVR, SUMNEW, TKVALS, TWORD, WHTYPE, WPVALUES

SUBROUTINES CALLED: COMPWRIT, DIRECT, GRPWRIT, HDFND, HEAD, IGET, KEYMAKE, NEXTTT, ORDER, RETRV, TERMTAPE, WRARRAY, WRWORD

CALLED BY: ENTMOD (of PREPALOC)

Method:

At this stage of processing, the first section of the BASFILE has been generated. PARTB controls the processing necessary to complete the writing of the BASFILE. In addition, numerous prints are supplied, if requested.

Subroutine COMPWRIT is first called in order to generate complex target related data. Individual targets are chained and values stored in load array CLASVAL based on target classes. If requested, fixed assignments information is printed followed by the standard print defining the collected target values.

Subroutine GRPWRIT execution causes weapon group and type related data to be written onto the BASFILE.

Recovery base information is defined within arrays DISTR, INDCAP, RCBLAT, RCBLONG and upon definition written onto the BASFILE.

Finally, tanker data processing completes the writting of the BASFILE.

Subroutine PARTB is illustrated in figure 17.

*Main subroutine of overlay link E.

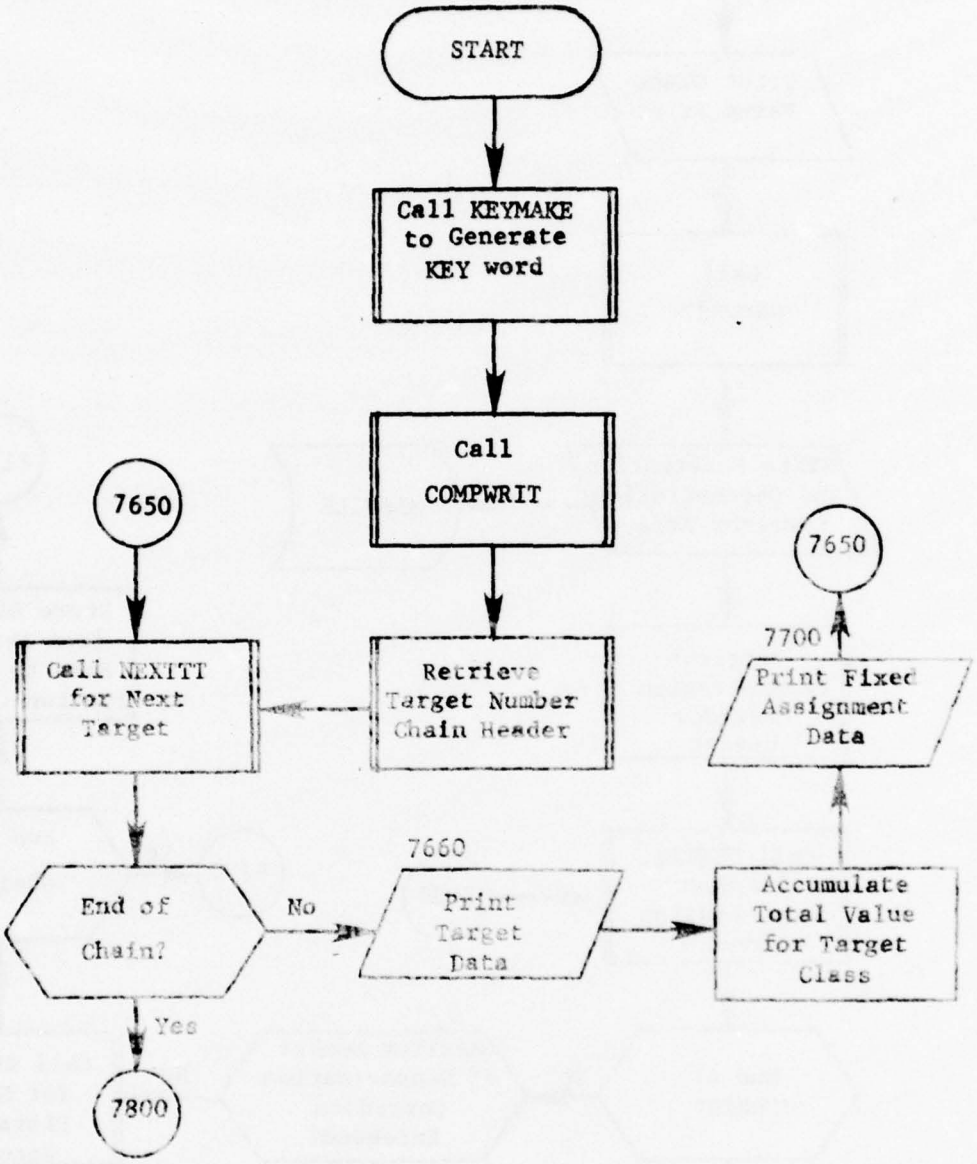


Figure 17. Subroutine PARTB (Part 1 of 4)

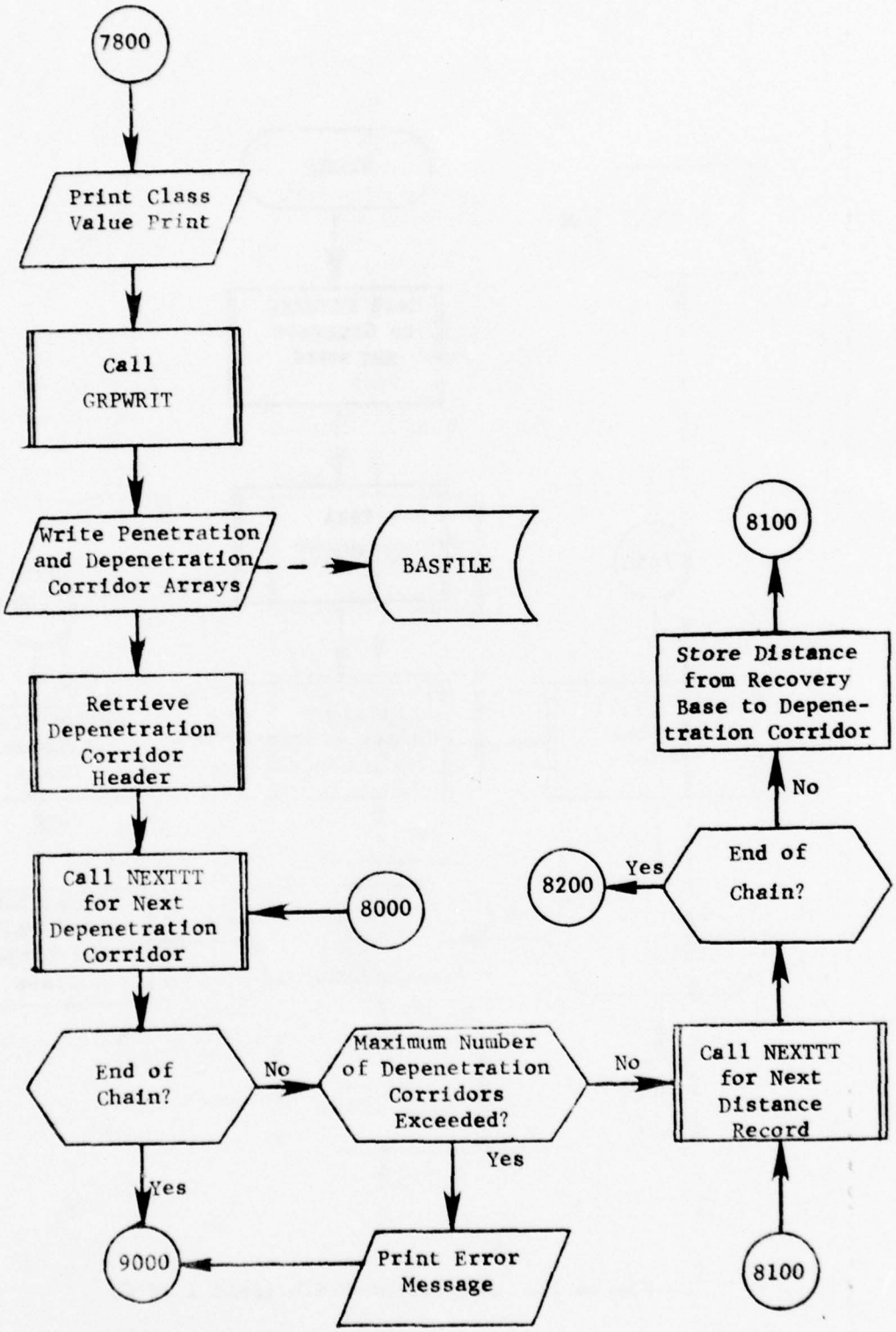


Figure 17. (Part 2 of 4)

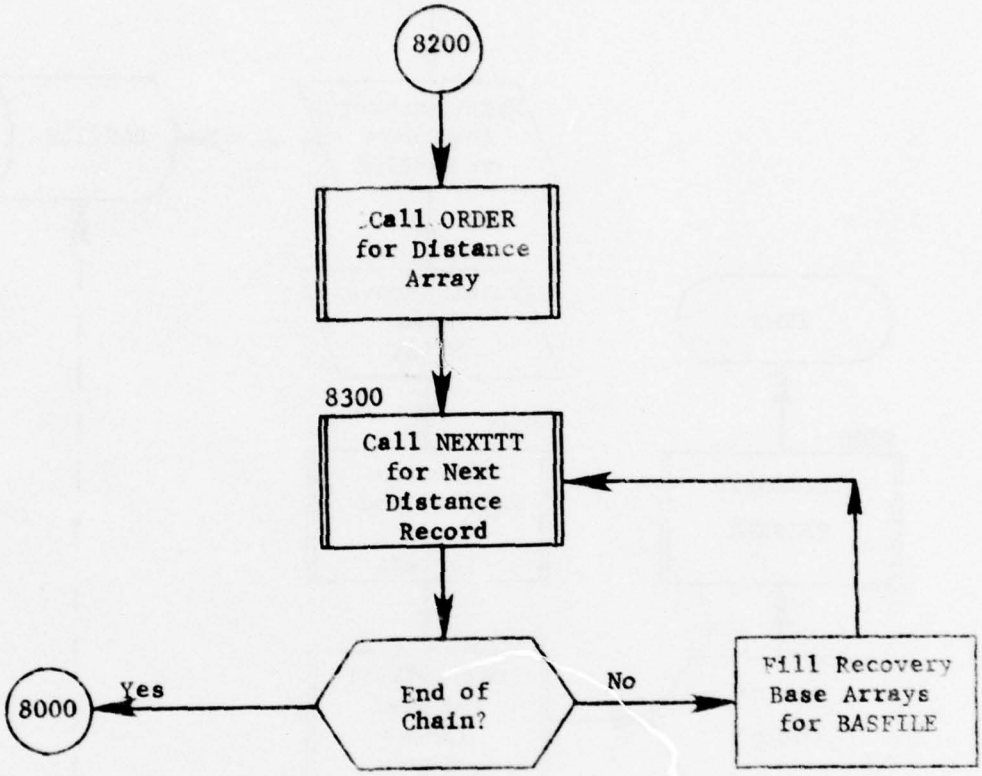


Figure 17. (Part 3 of 4)

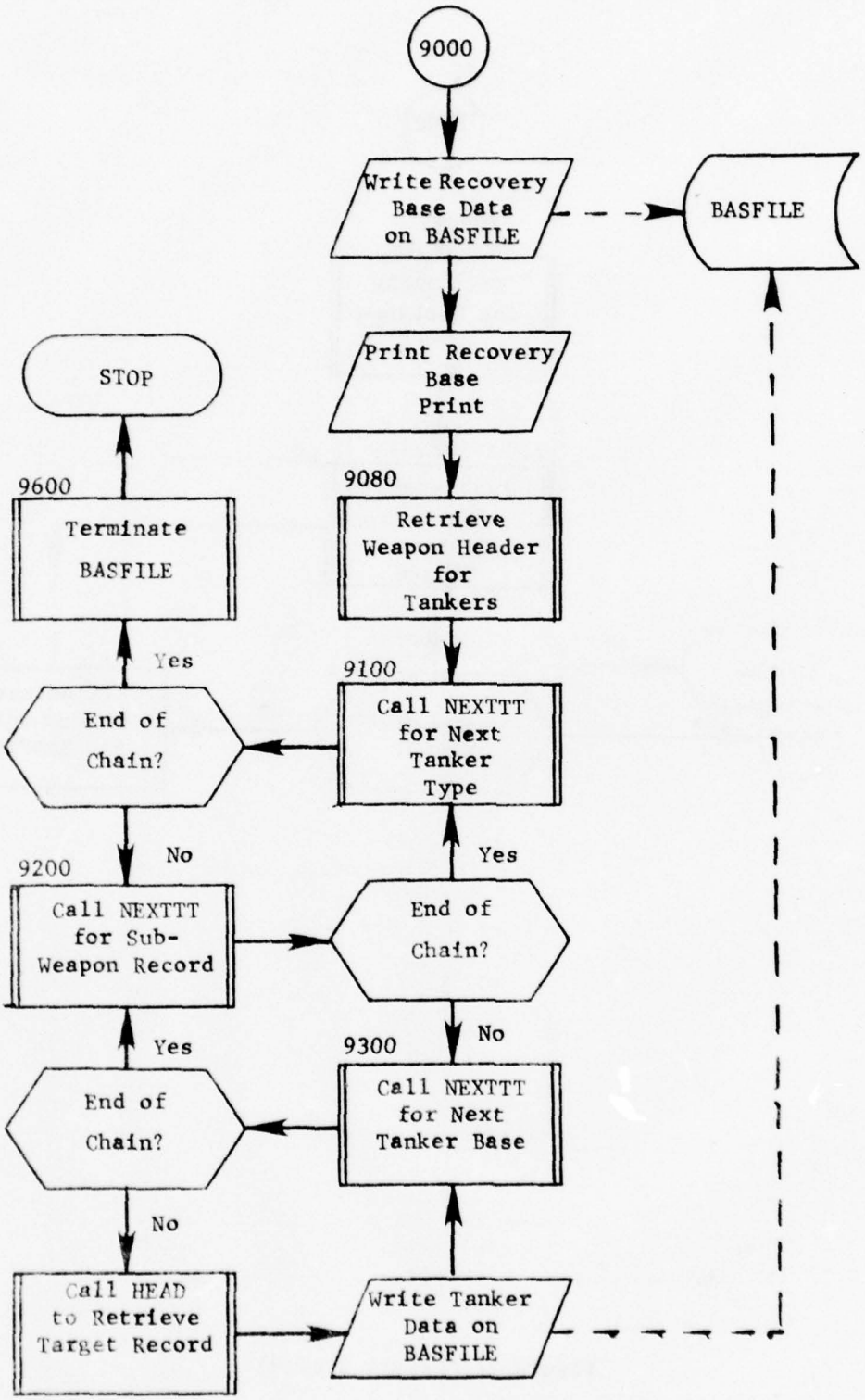


Figure 17. (Part 4 of 4)

2.12.1 Subroutine COMPWRIT

PURPOSE: To compute complex target data and write it on BASFILE

ENTRY POINTS: COMPWRIT

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, SUMNEW, TGTAR, TWORD

SUBROUTINES CALLED: DIRECT, HDFND, HEAD, ITLE, MODFY, NEXTTT, RETRV, WRARRAY, WRWORD

CALLED BY: PARTB

Method:

Data describing individual elements of target complexes and/or elements of multiple target are collected and written onto the BASFILE. The record format differs depending on the type of target. The information must be written such that it parallels the order as written onto the TGTFILE. Therefore, as with TARFILE processing, the LISTXX chain is queried and targets investigated for being defined as being either a multiple or a complex. For simple targets no action is required.

Subroutine COMPWRIT is illustrated in figure 17.1.

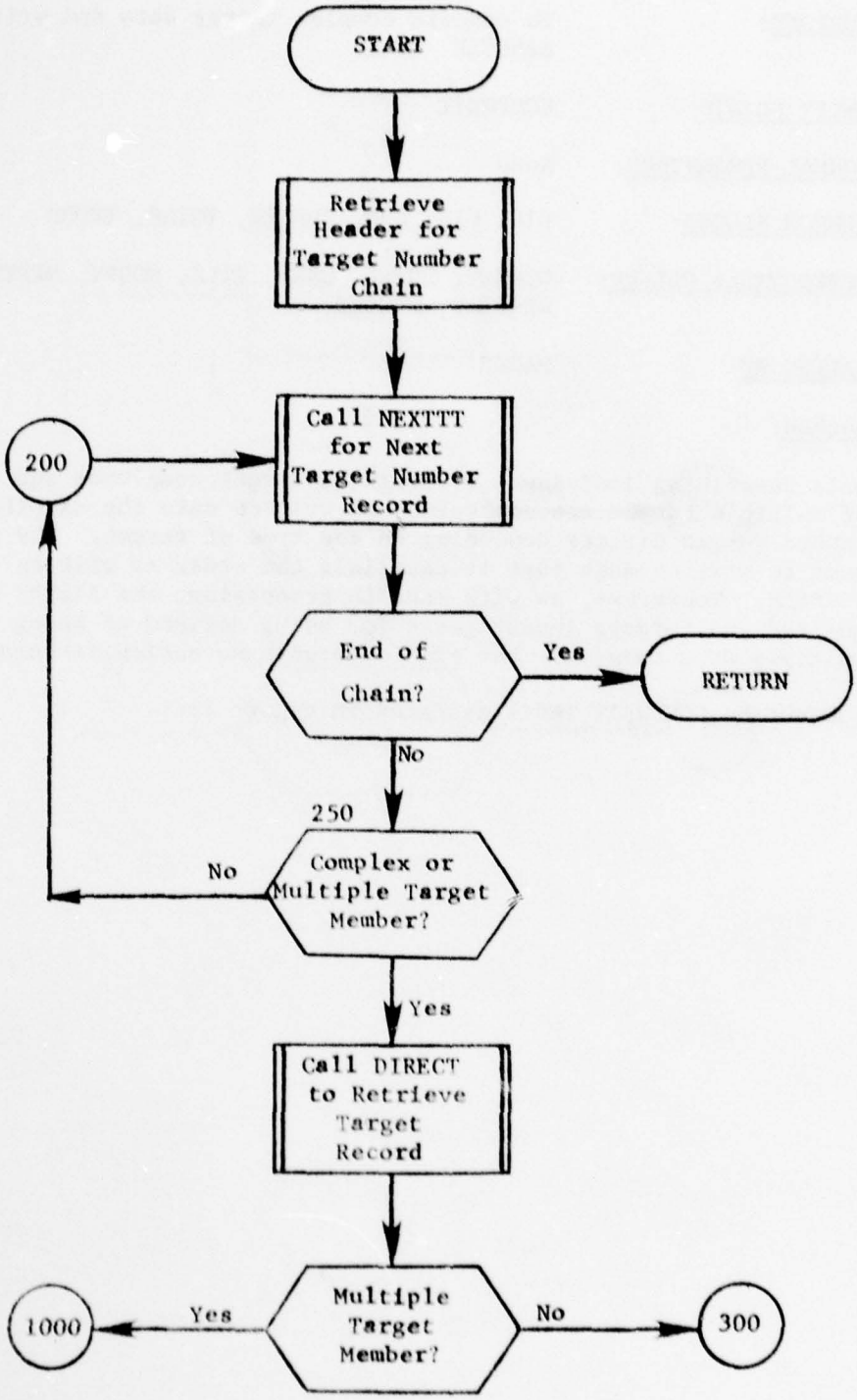


Figure 17.1. Subroutine COMPWRIT (Part 1 of 4)

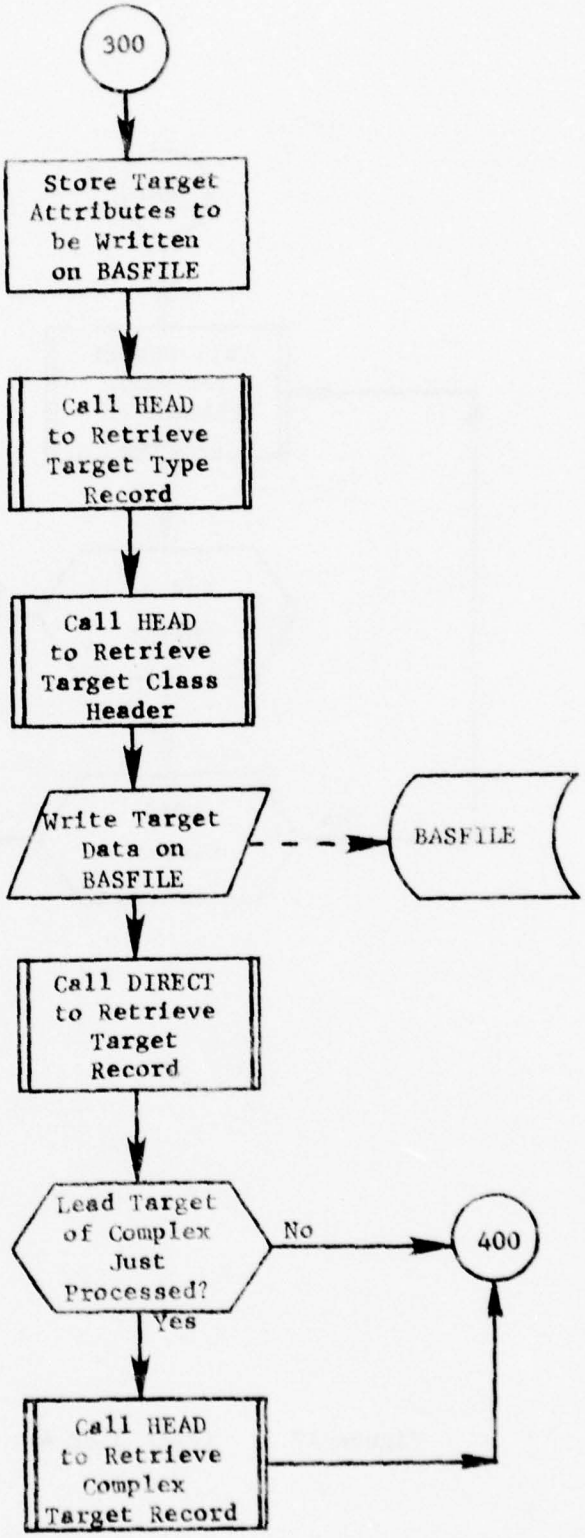


Figure 17.1. (Part 2 of 4)

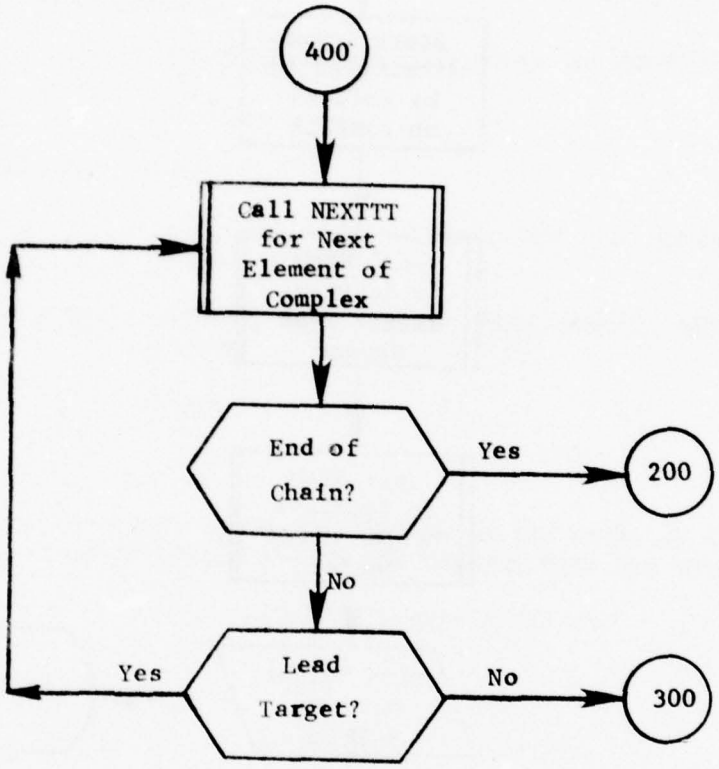


Figure 17.1. (Part 3 of 4)

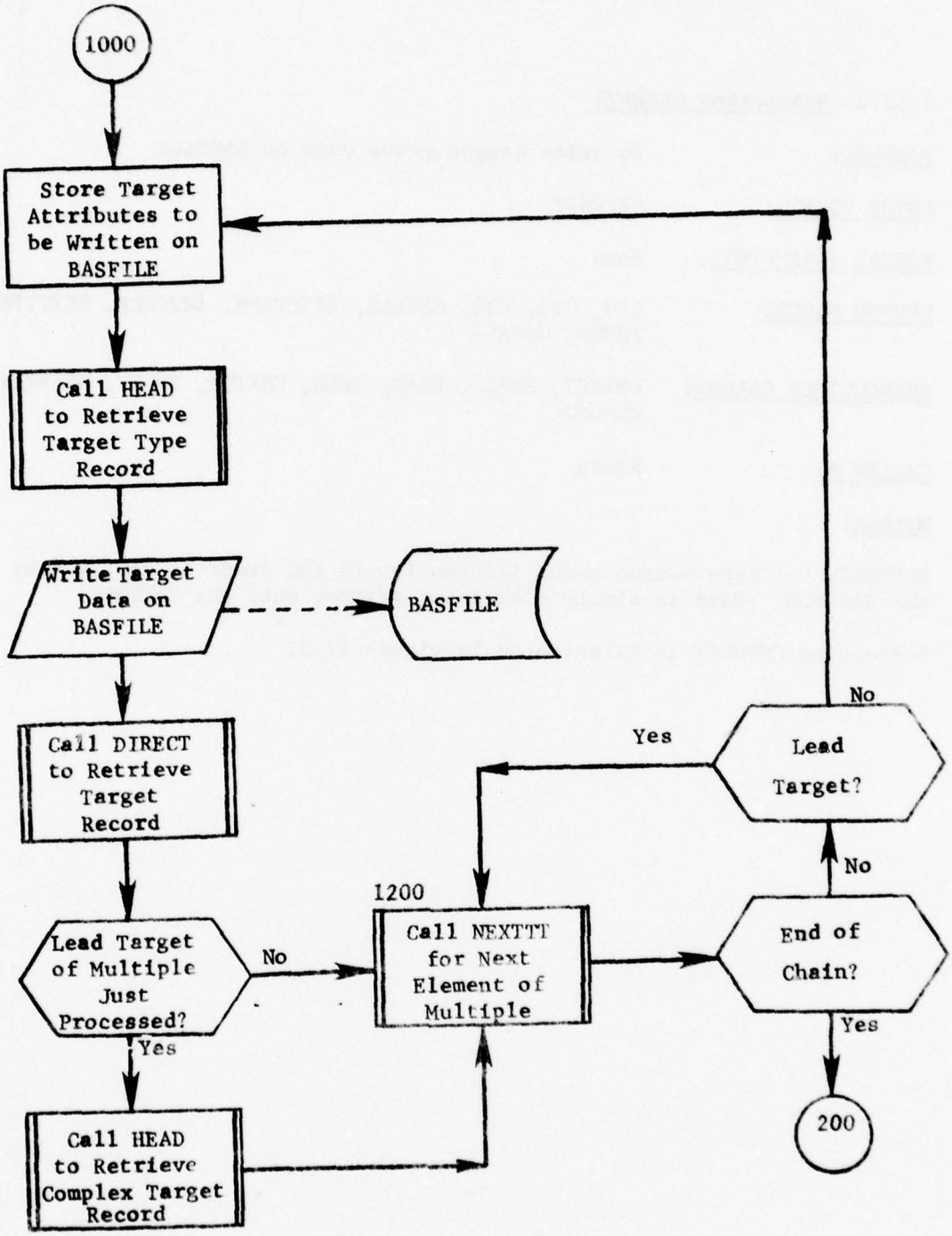


Figure 17.1. (Part 4 of 4)

2.12.2 Subroutine GRPWRIT

PURPOSE: **To write weapon group data on BASFILE**

ENTRY POINTS: GRPWRIT

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, GPVALS, ISIMTYPE, IWEPREF, PAYTYPE, TWORD, WPVALS

SUBROUTINES CALLED: DIRECT, HDFND, HEAD, ITLE, NEXTTT, RETRV, WRARRAY, WRWORD

CALLED BY: PARTB

Method:

GRPWRITE collects weapon group information in the order as dictated by the BASFILE. Data is simply stored and written onto the BASFILE.

Subroutine GRPWRIT is illustrated in figure 17.2.

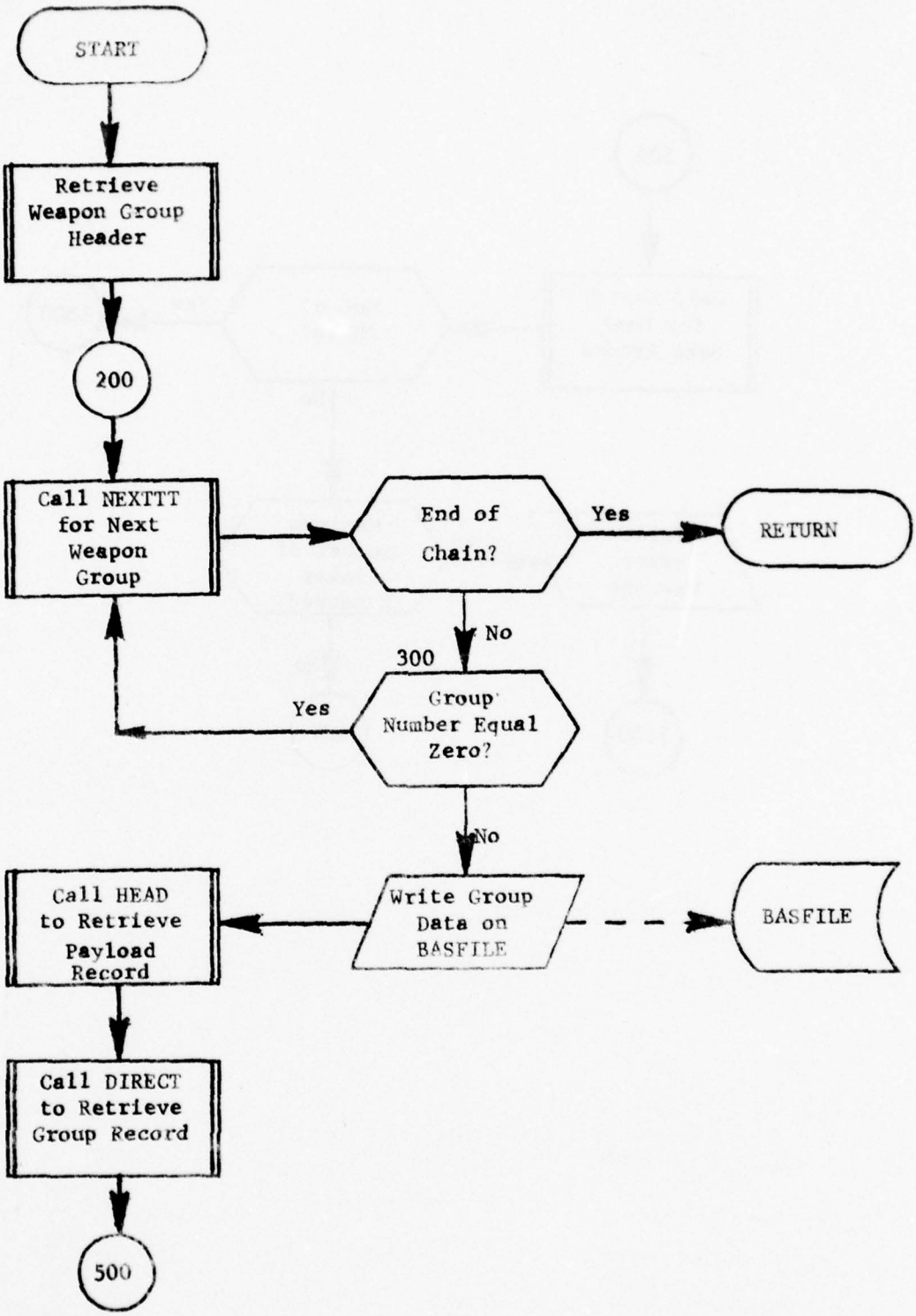


Figure 17.2. Subroutine GRPWRIT (Part 1 of 3)

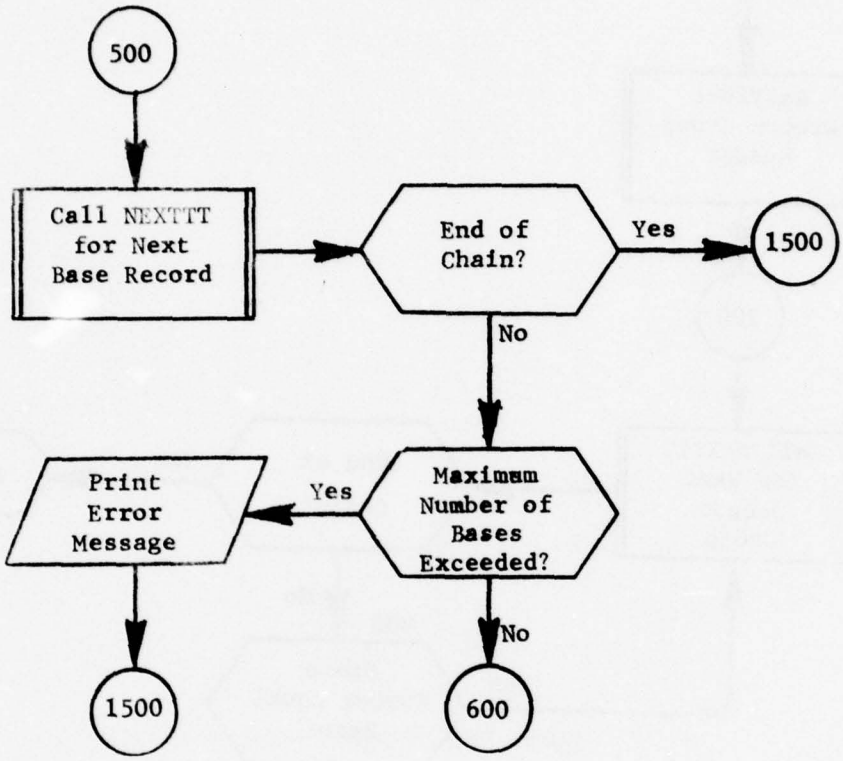


Figure 17.2 (Part 2 of 3)

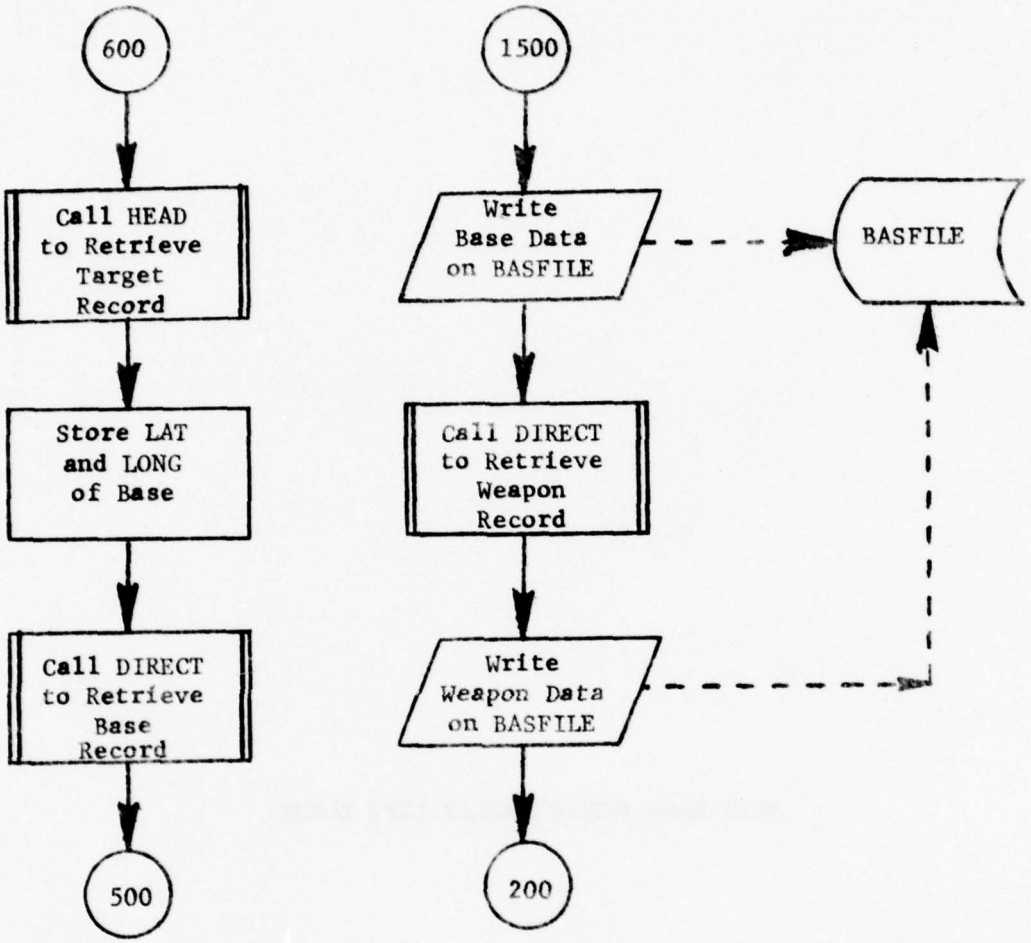


Figure 17.2. (Part 3 of 3)

THIS PAGE INTENTIONALLY LEFT BLANK

3.4.3.1 STOP: This function is performed in subprogram ALOC and provides normal termination. This function may be replaced by the DUMP function.

3.4.3.2 DUMP: This optional function replaces the STOP function and uses utility subroutine ABORT to terminate the job with a memory dump.

3.4.4 Allocation Function - ALLOCATE (Required). The QUICK system is capable of allocating up to 250 distinguishable weapon groups against a target system of any size. It is clearly not feasible to keep information in memory concerning the capability of all weapons with respect to all targets. Moreover, it would be inefficient to recompute such information each time it was required. Therefore, during the first pass through the targets, subroutine GETDTA prepares files which for each target list the capability of all relevant weapon types. In this way, on later passes, it is possible to examine efficiently any desired allocation of weapons against each target. To minimize the time required for the allocation, as much of the required information as possible is recomputed and stored on these files. The allocator generates a complete weapon assignment against each target before proceeding to the next. It thus passes through the files one target at a time and then recycles the files as required to obtain a satisfactory allocation.

The design of the weapon-to-target allocator utilizes a hierarchy of eight subroutines operating at different levels of detail. Figure 18 illustrates this hierarchy. The major functions associated with these subroutines are summarized below and related to the overall concept in subsequent paragraphs.

Subroutine MULCON is the first subroutine in the hierarchy and is responsible for the control and adjustment of the Lagrange multipliers. MULCON monitors the rate at which various classes and types of weapons are being allocated to the target system and makes appropriate adjustments in the values of the Lagrange multipliers. In this role, MULCON does not need any detailed information concerning actual allocation. It is concerned only with the actual rate of allocation of the available inventory as the targets are processed. To obtain the assignment of weapons to each successive target, MULCON simply calls subroutine STALL (Single Target Allocator) for targets without missile defenses, or subroutines STALL and DEFALOC if the target is defended. STALL and DEFALOC utilize the current values of the multipliers to make an allocation to the next target, then return control to MULCON. However, since MULCON is the main controlling routine it is convenient for it to handle much of the data input and output operations.

In this second role, it reads and writes the dynamic target data files, ALOCT1 and ALOCTAR (or ALOCT2), which contain the allocation information. Since this information changes from pass to pass, these files are called

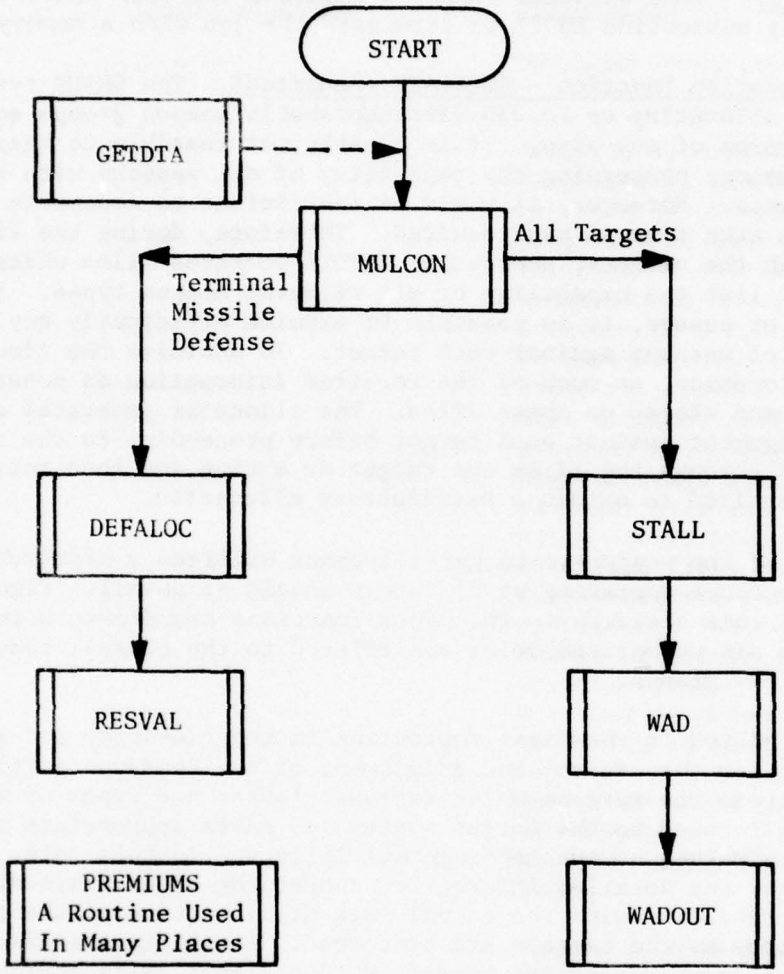


Figure 18. ALOC Calling Sequence Hierarchy

Table 11. Program ALOC External Common Blocks (Part 1 of 7)

INPUT FROM BASFILE

BLOCK	VARIABLE OR ARRAY*	DESCRIPTION
MASTER	IHDATE	Date of run initiation
	IDENTNO	Run identification number
	ISIDE	Attacking side
	NRTPT	Number of route points
	NCORR	Number of penetration corridors
	NDPEN	Number of depenetration corridors
	NRECOVER	Number of recovery bases
	NREF	Number of directed refuel areas
	NREG	Number of command and control regions
	NTYPE	Number of weapon types
	NGROUP	Number of weapon groups
	NTOTBASE	Total number of bases
	NPAYLOAD	Number of payload types
	NASMTYPE	Number of ASM types
	NWHDTYPE	Number of warhead types
	NTANKBAS	Number of tanker bases
	NCOMPLEX	Number of complex targets
	NCLASS	Number of weapon classes (2)
	NALERT	Number of alert conditions (2)
	NTGTS	Number of targets
	NCORTYPE	Number of penetration corridor types
	NCNTRY	Number of distinct country codes
FILES	TGTFILE(2)**	Target data file
	BASFILE(2)	Data base information file
	MSLTIME(2)	Fixed missile timing file
	ALOCTAR(2)	Weapon allocation by targets file
	TMPALOC(2)	Temporary allocation file
	ALOCGRP(2)	Allocation by group file
	STRKFIL(2)	Strike file

* Parenthetical values indicate array dimensions. All other elements are single word variables.

** In two-word arrays, first word is filehandler buffer usage number; second word is maximum file length in words. Single variables are logical tape unit numbers.

Table 11. (Part 2 of 7)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
FILES (cont.)	PLANTAPE*	Detailed plans tape
CORRCHAR	PCLAT(30)	Latitude of corridor point
	PCLONG(30)	Longitude of corridor point
	ZPLAT(30)	Latitude of corridor origin
	ZPLONG(30)	Longitude of corridor origin
	ENTLAT(30)	Latitude of corridor entry
	ENTLONG(30)	Longitude of corridor entry
	CRLENGTH(30)	Distance from corridor entry to corridor origin
	KORSTYLE(30)	Power of y versus x **
	ATTCORR(30)	High altitude attrition per nautical mile unsuppressed
	ATTRSUPF(30)	High altitude attrition per nautical mile suppressed
	HILOATTR(30)	Ratio low to high altitude attrition (less than 1)
	DEFRANGE(30)	Characteristic range of corridor defense (nautical miles)
	NPRCRDEF(30)	Number of attrition sections this corridor
	DEFDIST(30,3)	Distance of precorridor leg
	ATTRPRE(30,3)	Attrition in this precorridor leg
	NDATA	Number of words in common /CORRCHAR/
	LMAX	Maximum number of precorridor legs
PAYLOAD	NOBOMB1(40)	Number of type-1 bombs
	IWHD1(40)	Type-1 warhead index
	NOBOMB2(40)	Number of type-2 bombs
	IWHD2(40)	Type-2 warhead index
	NASM(40)	Number of ASMs
	IASM(40)	ASM index
	NCM(40)	Number of countermeasures
	NDECOYS(40)	Number of terminal decoys
	NADECOYS(40)	Number of area decoys
	IMIRV(40)	MIRV system identification number
	MPAYLOAD	Maximum number of payload types

* These files are output on magnetic tape.

** See Program POSTALOC.

Table 11. (Part 3 of 7)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
REF*	RFLAT(20)	Refuel point latitude
	RFLONG(20)	Refuel point longitude
PLANTYPE	INITSTRK	Indicator for first or second strike
	CORMSL	Coordination time parameter for missiles
	CORBOMB	Coordination distance for bombers
WPNREG	CCREL(20)	Command and control reliability by command and control region
WPNTYPE**	RANGE(100)	Range of vehicle (nautical miles)
	CEP(100) ***	CEP (nautical miles)
	SPEED(100) ***	Speed (knots)
	ALERTDLY(100)	Alert delay (hours)
	NALRTDLY(100)	Nonalert delay (hours)
	RANGEDEC(100)≠	Low/high altitude fuel consumption ratio
	ICLASS(100)	Weapon class
	RANGEREF(100)≠≠	Refueled range (nautical miles)
	REL(100)	Reliability
	IRECMODE(100)	Recovery mode
	IPENMODE(100)	Penetration mode
	LCHINTVL(75)	Launch time interval (missiles only) (hours)
	SIMLUNCH(75)	Number of simultaneous launches (missiles only)
WPNGRP**	NWPNS(250)	Number of weapons in group after removal of fixed weapons
	WLAT(250)	Centroid latitude

* From BASFILE block DPENREF.

** From BASFILE block WPNDATA.

*** For missiles, this variable is replaced by attribute TOFMIN, minimum time for flight (hours).

≠ For missiles, this variable is replaced by CMISS, a missile flight time variable computed by PLANSET.

≠≠ For missiles, this variable is replaced by RNGMIN, the minimum range (miles).

Table 11. (Part 4 of 7)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
WPNGRP (cont.)	WLONG(250)	Centroid longitude
	IREG(250)	Command and control region
	ITYPE(250)	Type index (LTYPE)
	IALERT(250)	Alert status
	SBL(250)	Probability of survival before launch
	IREFUEL(250)	Indicates refuel code for bombers or payload index for missiles
	YIELD(250)	Weapon yield (megatons)
	REFTIME(250)	Refuel time
	EXPASM(250)	Fraction of weapons in group that are ASMs
PKNAVAL*	PKNAV(250)	Single shot kill probability against naval targets
CTRYCD	CTRYCD(150)	List of distinct country location codes on defending side
ASMT**	IAMSM(250)	Index to ASM table for each group
	YLD(50)	Warhead yield from warhead table
	IWHDASM(20)	Index to warhead table for ASM warhead
	RELASM(20)	ASM reliability
	CEPASM(20)	Circular error probable for ASM delivery
PAYDATA	PAYALT(40)	Bomber weapon release altitude indicator
HOB	LXISPEC(3)	Logical array set true for user specified HOB by weapon type
	LXIWHOB(3)	Logical array designating user specified HOB by weapon type

* From BASFILE block NAVAL.

** From BASFILE blocks WPNGRP, WARHEAD, ASMTABLE.

Table 12. (Part 2 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
C222	WTFAC(3)	Divide into old target weights for commensurability
	WTRATE(3)	Rate of increase of target weights
	WTSUM(3)	Sum of target weights
	JATTRIB(6,250)	Index to local multipliers for weapon group
	RUNSUM(380,3)	Running sum of target weight times weapons allocated for local multiplier
	ALLEREST(380,3)	Estimate of allocation error
	LA(380)	Value of local multiplier
	MXATTRIB	Maximum number of local attributes (Lagrange multipliers)
C333	NWP(10)	Number of weapons in TOA set
	VAL(10)	Unattrited target value at TOA for each set
	V(11,2)	Unattrited component at TOA for each set N and each hardness component
	MU(10,2)	Sum of means through each TOA set
	SIG(10,2)	Sum of variants through each TOA set
	S(10,2)	Component survival probability through each TOA set
	VS(10,2)	= (V(N,JH)-V(N+1, JH))*S(N,JH)
	VSN(11,2)	= VSN(N-1, JH)+VS(N-1, JH)
	ITOA(250)	TOA index for weapons in group if added
	IADDTOA(250)	Set to 1 if new TOA set is required
	SIGP(250,10,2)	Increase in variance for each TOA set N if weapon is added from group
	SIGP(30,10,2)	Change in variance for each TOA set if weapon deleted
	DSIG(250,2)	Variance contribution for weapon vs. specified weapon
C333*	LXFLAGOK(63)	Logical array set true only if weapons from group can be allocated to targets with designated flag code (nine flag codes per group)

* This version of C333 is used during the input phase of ALOC, and by subroutine GETDTA during the first pass.

Table 12. (Part 3 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
C333 (cont.)	LXCTRYOK(1042)	Logical array set true only if weapons from group can be allocated against targets designated country code (150 country codes per group)
	MYNMIRV	Number of MIRV systems in list
	MYMIRV(100)	IMIRV system number
	NPERMIT(100)	Number of permitted classes
	MYPERMIT(100,16)	Hollerith names of permitted classes
	LXTBMDEF(3)	Logical array--true if defence targets are permitted
	LXMULTARG(3)	Logical array--true if multiple targets are permitted
	MAXMIRV	Maximum number of MIRV systems
	MAXNP	Maximum number of permitted classes
	RANGEMUL(250)	Multiplier for unrefueled range for group
	RANRFMUL(250)	Multiplier for refueled range for group
	RANGEMIN(250)	Minimum range (nautical miles) for group
C333*	LXFLAGR(1)	FLAGOK array for one group
	LXCTRYR(5)	CTRYOK array for one group
	MPMIT(16)	MYPERMIT array for one group
	NPMIT	NPERMIT for one group
	TBMD	TBMDEF for one group
	MULTG	MULTARG for one group
	RMUL	RANGEMUL for one group
	RFMUL	RANRFMUL for one group
	RXMIN	RANGEMIN for one group
	FILL3(3052)	Dummy array used to retain common block length
C333**	VTDX	Surviving target value
	RATM	Maximum missile rate of return
	NOWEP(250)	Number of weapons (from a group) assigned to target

* These variables (LXFLAGR-RXMIN) used only in subroutine GETDTA.

** A redefinition of common block /C333/ used only in subroutines PRNTNOW, RESVAL, and DEFALOC.

Table 12. (Part 4 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
C333 (cont.)	RATE(250)	Rate of return for group
	ISALFX(250)	Average salvo number of fixed salvoed missiles
	NSL(250)	Number of weapons available in salvo. (Set to 1000 for nonsalvoed groups)
	FILL3D(5742)	Dummy array used to retain common block length
DEFENSE	NTX(3)	Estimates of terminal interceptors present
	PX(3)	Probability that NTX occurs
	PKTX	Terminal interceptor kill probability against unhardened warhead
	RX(2)	NTX/MISDEF for upper and lower deviations from MISDEF
	PRX(2)	Probability there are RX*MISDEF interceptors
	RADPX	Probability of warhead kill by random area defense
	TINTFAC	Multiplier for terminal interceptor defense levels
FIXED	IFW(31)	Group numbers of weapons fixed this target
	TIME(30)	Specified arrival time of fixed missiles
	NFIXWPS	Total number of fixed missiles
	SX(5)	Scratch array
	IFIXBEG, IFIXEND	Pointers to input data for fixed weapons
	SAVFIX	Logical variable set true only if fixed missile assignments are to be output on MSLTIME file
	IADD	If negative -- no fixed assignments from TGTFILE If zero -- fixed assignments according to TGTFILE
FIXEDASS	NLFTAR	Number of fixed assignments for this target on TGTFILE that have not yet been processed

Table 12. (Part 5 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
FORMTT	INWORD	Input word value
	INFORMAT	Hollerith code for appropriate 10-column format for input word
LAMBDA	LAMEF(250)	Value of Lagrange multiplier for each weapon group
	SURPWP(250)	Estimated surplus for weapon group
	PREMIUM(250)	Premium for using weapon group
	DPREMIUM(250)	Premium for deleting weapon group
LOCFIL	WPNTGT	Current WPNTGT file in use
	ALOCT1	Filehandler buffer utilization number of ALOCT1 file
	ITMPMSL	Buffer number for temporary MSLTIME file (unordered)
MACHINE	IREAD	Logical unit number-standard input
	IWRIT	Logical unit number-standard output
	IPUNCH	Logical unit number-standard punch
MULADJ	NINTPRD	Number of separate integration periods used
	RINTPRD	Controls ratio between integration periods
	RATIOINT	Ratio of controlling integration period to that implied in allocation rates
	SNSTVTY	Rate of multiplier convergence relative to the significance of errors
	FSNSTVTY	Final rate of convergence relative to number of targets
	SETTLE	Number of cycles for multipliers to settle to closing
	BPENFAC	Multiplier for bomber attrition rates
NALLY	NALL(250)	Number of weapons allocated this pass from each group
	RNALL(250)	Number of weapons now on all targets from each group

Table 12. (Part 6 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
PAYOFF	OPROFIT	Profit for hold allocation (last pass) evaluated with present values LAMEF
	SPAYOFF	Cumulative payoff all targets
	SUMCOST	Cumulative cost all targets
	SPROFIT	Cumulative profit all targets
	SUMPREM	Cumulative premiums all targets
	TBENEFIT	Total benefit this target
	NAT	Number of attribute categories (currently 6)
PEN	PENALT(30)	Temporary storage for penetration probabilities for each corridor
PRINEED	N	Weapon group index
	G	Weapon group index
PRIN1	STALPRIN	Print indicator showing location in STALL processing
PRNTCN	IFRSTPR	Initialization trigger for subroutine PRNTCON
	IDO(40)	Contains optional print flags (set to 3 if print active, 1 for inactive)
	INDEXPR(40)	Requested option
	JPASS(40)	First pass to activate request
	JTGTP(40)	First target to activate request
	LPASS(40)	Last pass to activate request
	LTGT(40)	Last target to activate request
	KTGTFREQ(40)	Print frequency
	ICOUNT(40)	Number of targets processed since last print
	MYPRT(40)	Request setting mode (DEFAULT or INPUT)
	MAXREQ	Maximum number of print requests
	MPRNT	Maximum number of print options
	NREQ	Actual number of print requests

Table 12. (Part 7 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
PRNTWADD	IG1	
	J1	
	N1	
	XMU	
	S1	Internal indexes used in optional prints from subroutine WAD
	VSN1	
	IG2	
	J2	
	N2	
	NW2	
PRTMULL	NTATTRIB	Total number of local multipliers
	PROCMULT	Target multiplicity now processed this target
	ERRCLOS	Factor which controls the termination of the allocation
	CLOSER	Controls increase in closing force per pass
	DELTEFF	Increase in profit from last pass divided by VALWPNS
	SDELTEFF	Cumulative DELTEFF
	VALWPNS	Value of all weapons
	VALERR	Value of surplus and deficit weapons combined
SALVO	MGSAL	Maximum number of salvoed groups (75)
	MAXSAL(75)	Maximum salvo number
	NSALAL(450)	Running sums of salvo allocation (six words per group)
	LXIHAVE(50)	Logical array set true only if salvo has some weapons available (24 salvoes per group)
	SAVLAM(250)*	Temporary storage array for Lagrange multipliers

* For bomber groups these variables are equivalenced as follows:
SAVLAM equivalenced to AVDE, the average damage difference between use of ASMs and bombs
MYSAL equivalenced to ISETPAY, the bomber payload indicator (zero for bombs, one for ASMs)
P equivalenced to FASM, the fraction of weapons allocated from a group which are ASMs

Table 12. (Part 8 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
SALVO (cont.)	MYSAL(250)*	Preferred salvo indicator
	KEYSAL(4)	Packing keys for salvoes
	KSALWD	Number of salvoes in one word of NSALAL (set to 4)
	P(250)*	Salvo balance variable
SMAT	SMAT(6,5)	Interweapon correlation information
	LSMAT	Length of SMAT array
	SMATMIRV(3)	Storage of SMAT values which change for MIRV systems
	SMNOMIRV(3)	Storage of original values for non-MIRV systems
TABLE	TABLE(101)	Kill factors for use in square root damage function
WADFINAL	VTP(250)	Total expected undestroyed target value with weapon added from each group
	DELVT(30)	Change in residual target value by weapon allocation
	NUMO	Number of weapons assigned this target on old target
	IGO(30)	Index of weapons assigned on old pass
	IOP	Total number of add and delete operations this target
	IOPS	Grand total number of add and delete operations
	CTSPILL	Number of multiple target elements postponed for later processing
WADOTX	PVRMX	Maximum efficiency for weapon IPVRMX
	IPVRMX	Weapon group index
	PPMX	Maximum profit for weapon IPPMX

* For bomber groups these variables are equivalenced as follows:
SAVLAM equivalenced to AVDE, the average damage difference between use of ASMs and bombs
MYSAL equivalenced to ISETPAY, the bomber payload indicator (zero for bombs, one for ASMs)
P equivalenced to FASM, the fraction of weapons allocated from a group which are ASMs

Table 12. (Part 9 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
WADOTX (cont.)	IPPMX	Weapon group index
	DVRMN	Not used
	IDVRMN	Not used
	DPMN	Minimum profit
	IDPMN	Estimated minimum profit
	NUMMAX	Maximum number of weapons allowed per target
	NW	Number of weapons now on target
	TPMX	Largest profit or potential profit so far
	NTOA	Number of time arrival bins used
	NTOAMAX	Maximum number of time arrival bins allowed
	VTMIN	Destruction of target below VTMIN considered of no value
	VTMAX	Maximum acceptable surviving target value
	ALPHA	Value factor required to justify VTMAX
	VTEF	Maximum of either VT or VTMIN
WADWPN*	JTGT	Static target number
	INACTIVE(250)	Nonzero value flags inactive groups
	TOA(250)	Time of arrival at target of weapons from each group
	TVALTOA(250)	Unattrited target value at TOA for weapons in each group
	VTOA(250,2)	Unattrited value for each component at TOA for weapons in group
	MUP(250,2)	Contribution of weapon to mean if added
	RISK(6,250,2)	SQRTF(2*SUMMODE(GAMAMODE* LN(MODEK)/LN(TK)))

* This block is used as an input buffer in subroutines RDALCRD, LOCREST, FLAGRST, MIRVRST, and RNGEMOD. The alternate definition of items in this block follows the original definition.

Table 12. (Part 10 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
WADWPN (cont.)	SSIG(250,2)	Square root of -log of s
	MINKILL	Minimum acceptable kill probability
	MAXKILL	Maximum desired kill probability
	MAXCOST	Maximum acceptable cost to achieve MINKILL (weapon cost /target value)
	ILAW	Kill probability law indicator
	MISDEF	Number of terminal ballistic missile interceptors
	MORR(250)	Preferred penetration corridor
	PEX(250)	Penetration probability
	XMUP(250,2)	Single shot survival probability -- used only for targets with ballistic missile defenses
	JTGTX*	Static target number for next target
	LNEXT	Number of words in WPNTGT record
	MINKILX	MINKILL for next target
	MAXKILX	MAXKILL for next target
	MAXCOSX	MAXCOST for next target
	NACTV	Number of active groups on next target
	IGX(250)	Group numbers of active groups
	TOAX(250)	TOA for active groups
	MORRX(250)	MORR for active groups
	PEXX(250)	PEX for active groups
	STKX(250,2) } STK2X(250,2) }	Intermediate interaction calculation variables for active groups
	LSTMAX	Length of WPNTGT I/O buffer
WADWPN (as redefined in RDALCRD, LOCREST, FLAGRST, MIRVRST, RNGEMOD.)	INPUT(14)	Input card storage array
	NVARS	Number of parameters on input card
	NAMES(100)	Parameter names
	INVALU(2,100)	Parameter input values
	INDEX1(100) INDEX2(100) INDEX3(100)	Array indexes for parameters

* This area, from JTGTX to the end of the block, is used as an I/O buffer for the WPNTGT files.

Table 12. (Part 11 of 11)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
WADWPN (cont.)	MORE	Input termination indicator
	NYNAME(100)	Default parameter names
	MYFORM(100)	Default parameter output format
	MYTYPE(100)	Default parameter setting mode (equivalent to FVAL)
	MYVAL(100)	Default parameter - value
	MYGOTO(100)	Default parameter - index to variable type
	NDEFLT	Number of default parameters
	FILLER(7147)	Dummy array
WAROUT	IWARFL	Logical unit numbers for the war gaming print output.

3.8.10 Subroutine SALVAL

PURPOSE: This routine selects the preferred salvo for each salvoed missile group, saves, and restores the appropriate Lagrange multipliers.

ENTRY POINTS: SALVAL, INITSAL, NEWSAL, RESTORE

FORMAL PARAMETERS: None

COMMON BLOCKS: CONTROL, DYNAMIC, LAMBDA, PRINEED, SALVO, WADWPN, WPNGRP, WPNTYPE

SUBROUTINES CALLED: GLOG, IGET, LAMGET, VALTAR

CALLED BY: STALL, DEFALOC, WAD

Method:

There are three entry points, INITSAL, RESTORE, and NEWSAL. Entry INITSAL is called at the beginning of automatic weapon allocation in STALL or DEFALOC. First, it saves the multipliers LAM for each salvoed group in array SAVLAM. Then, for each group and each salvo, INITSAL determines the salvo number and cost of the best salvo. The profit of each salvo is defined as

$$PR_i = VTAR_i - TLAM_i$$

where $VTAR_i$ is the value of the target at arrival time of salvo i and $TLAM_i$ is the value of the multiplier for salvo i as determined by function LAMGET. The salvo with the highest PR_i is selected as the best salvo. The salvo number is entered in array MYSAL in common block /SALVO/ and the cost L_i replaces the multiplier for the group LAM. Entry RESTORE merely restores the original values of the multipliers (SAVLAM) back to the multiplier array LAM. Entry NEWSAL checks the current allocation and running sum for the weapon just added or deleted. If the salvo has been completely allocated, NEWSAL flags the salvo as unavailable. Entry SALVAL is never used.

Entry INITSAL

This entry is the most complex of SALVAL. The local variable IAM is set to "INITSL" to flag the exit points after statement 420 and at statement 500. The first processing (DO loop to statement 10) saves the Lagrange multipliers (LAM) in the SAVLAM array in /SALVO/.

The major processing in INITSAL occurs in the DO loop to statement 100 over all the salvoed groups. Within this loop, the DO loop to statement 300 investigates each salvo to determine its worth. Array IHAVE in /SALVO/ is used to exclude salvoes from consideration. If IHAVE (I, J) is false, then salvo I in group J does not have any available weapons and is ignored. If weapons were available, a jump is made to statement 420 in entry NEWSAL. That section of code, described later, determines if the salvo to be considered is overallocated beyond its limit. If so, the salvo is not further considered. If the salvo is available, function VALTAR is used to obtain VTAR the target value for the salvo. Function LAMGET is used to calculate TLAM, the multiplier for the salvo. The best salvo is the one with the highest positive difference between VTAR and TLAM. When this maximum is selected, MYSAL in /SALVO/ is set to the best salvo number. LAM in /LAMBDA/ is set to the appropriate multiplier. The arrays TOA, TVALTOA, and VTOA (in /WADWPN/) are set to correspond to the arrival time of the best salvo.

Entry INITSAL is illustrated in part 1 of figure 47.

Entry RESTORE

This simple entry point uses a DO loop to statement 1000 to restore the original values of the Lagrange multipliers (SAVLAM) to the multiplier array (LAM).

This entry is illustrated in part 3 of figure 47.

Entry NEWSAL

This entry is used after each allocation of salvoed weapons to determine if the salvo is still available. Each salvo has a maximum limit of overallocation. Before PROGRESS equals one, this limit is +225 which is the largest number which can be stored in the NSALAL array. (A description of the structure of the NSALAL array is contained in the Method section of Subroutine ADDSAL, in this chapter.) When PROGRESS equals one, a more severe limit is imposed in order to accelerate closure to the stockpile. In this case the limit is zero. That is, a salvo is available only if it is underallocated.

To exit from this entry, local variable IAM is checked. If it is "INITSL," then the original call was to INITSAL and control passes to statement 220 in entry INITSAL. If it is "NEWSAL" the routine exits.

*In line function IHAVE extracts information from logical array LXIHAVE.

SECTION 4. PROGRAM EVALALOC

4.1 Purpose

The purpose of program EVALALOC is to summarize the planned allocation of weapons to targets and provide an expected-value estimate of the results. Provision is also included to evaluate the allocation for variations in the values assigned selected parameters (planning factors) associated with the weapons and targets. The evaluation can be made for either the whole plan or for only targets in selected countries. EVALALOC may be run at two stages of plan development, before program ALOCOUT or after program PLANOUT. If run prior to the selection of desired ground zeros (DGZ) for complex targets (accomplished in ALOCOUT), the analysis of aim point offsets is not included. In this case, the results produced by EVALALOC represent an upper limit estimate which assumes that each target element in a complex is directly targeted. When EVALALOC is run after program PLANOUT, the weapon aim points offsets are available and are included in the expected value computations.

4.2 Input Files

When program EVALALOC is run before program ALOCOUT, the input files are the BASFILE prepared by program PREPALOC and the ALOCTAR file prepared by program ALOC. When run after program PLANOUT, the PLANTAPE produced by PLANOUT is also required as input.

4.3 Output File

Program EVALALOC does not produce an output file for use by later processors; its sole output is a set of summaries which present the expected value results of the planned weapon allocation.

4.4 Concept of Operation

Program EVALALOC processes the targets one at a time. For each target (or target element of a complex target), the assigned weapons are read in and ordered by time of arrival. Surviving target values are calculated, utilizing the same damage functions used in program ALOC (subroutine WAD), except that correlations are ignored. After the survival probability of each target is computed, the target and the assigned weapons are classified for summarization purposes. When all targets have been processed, the expected-value results are summarized and printed.

the allocation by target). When EVALALOC is run in the post PLANOUT mode, the weapon allocation data are obtained from the PLANTAPE but cannot be used directly. The PLANTAPE reflects the allocation by sortie (i.e., each block of data describes a delivery vehicle which transports warheads to one or more targets). Consequently, in this mode of operation, EVALALOC must first process the PLANTAPE and construct from it a file which reflects the allocation by target.

The program consists of the main executive program EVALALOC, a summarizing, data handling, and print subroutine EVAL2, and a computing subroutine EVALPLAN. EVALALOC includes provisions for exploring the sensitivity of the results to the assumed or calculated values of some of the weapon or target planning parameters. It also has the ability to use all targets or any subset of targets, based on country codes, in the evaluation. The program can be recycled and these parameter values can be varied using subroutines WPNMODIF and TGTMODIF.

As indicated above, when EVALALOC is run after PLANOUT, it is necessary to prepare a new file on which the weapon-to-target allocation is target-oriented. Because a large amount of data must be stored to describe the allocation, several items of information must be packed in each word. This is done by the four packing subroutines, PACK, BOMRPAKR, MISLPAKR, and UNPACKER. In the event that the pre-PLANOUT operation is prescribed, the packing routines are never called and the allocations are read from ALOCTAR.

Program EVALALOC (figure 57) reads the user's general control data card and stores the input parameters ITGTMAX, JOPT, and PREFABRT in common block /OPT/; the parameter PKTX in common block /MIS/; and the parameter LAW(1), LAW(2) in common block /LAW/. If ITGTMAX is negative, it stops the run. Otherwise, it initializes the filehandler and reads the /FILES/ and /MASTER/ common blocks from the BASFILE. If the run is post-PLANOUT it also reads the REL, ITYPE, and DBL arrays into common block /CWPNMOD/ so they can be used by subroutine UNPACKER to modify the values of the weapon penetration probability obtained from the PLANTAPE. In this mode, EVALPLAN next calls subroutine PACK to reorder the weapon allocation data from the PLANTAPE. It then calls EVAL2 to evaluate the allocation and summarize the results. In the pre-ALOCOUT mode of processing, subroutine PACK and its associated subroutines are not used, and EVALALOC proceeds directly to the EVAL2 call. After EVAL2 completes its processing, EVALALOC reads the next user general control data card and repeats the process described here until it reads a card where ITGTMAX is negative.

Subroutine PACK is used only in the post-PLANOUT operation of EVALALOC. PACK determines the order of the targets on the ALOCTAR file and reorders the weapon-to-target allocation data on the PLANTAPE in that target order. To do this, it must pack the weapon allocation data using subroutines MISLPAKR or BOMRPAKR (depending on the weapon type). At the time of packing, BOMRPAKR and MISLPAKR use subroutine SEARCH to locate the ALOCTAR target number associated with the target index number, INDEXNO, contained on the PLANTAPE. This number is also packed with the weapon

Table 17. Program EVALALOC External Common Blocks (Part 1 of 5)

INPUT FROM BASFILE

BLOCK	VARIABLE OR ARRAY*	DESCRIPTION
FILES	TGTFILE(2)*	Target data file
	BASFILE(2)*	Data base information file
	MASLTIME(2)*	Fixed missile timing file
	ALOCTAR(2)*	Weapon allocation by targets file
	TMPALOC(2)	Temporary allocation file
	ALOCGRP(2)	Allocation by group file
	STRKFIL(2)	Strike file
	PLANTAPE**	Detailed plans tape
ASMT	IAMSM(200)	Index to ASM table for each group
	YLD(50)	Warhead yield from warhead table
	IWHDASM(20)	Index to warhead table for ASM warhead
	RELASM(20)	Asm reliability
	CEPASM(20)	Circular error probable for ASM delivery

INPUT FROM PLANTAPE

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
/N/	LGRP	Weapon group index
	KORD	Corridor index number
	TMLAUN	Missile time of launch
	HDT(90)	Missile flight time or bomber/tanker time since last event
	KPL(90)	Place or site index
	JTP(90)	Missile index or event tape

* Parenthetical values indicate array dimensions. All other elements are single word variables. In two-word arrays, first word is logical unit number; second word is maximum file length in words. Single variables are logical unit numbers.

** These files are output on magnetic tape.

Table 17. (Part 2 of 5)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
/N/ (cont.)	HLA(90)	Event latitude
	HLO(90)	Event longitude
	TZT(90)	Missile weapon site latitude or bomber weapon delivery offset latitude
	TZN(90)	Missile weapon site longitude or bomber weapon delivery offset longitude
	IWH(90)	Warhead type or index
	PA(90)	Reliability/damage expectancy
	ICMT(90)	Bomber/tanker cumulative time to event
	ITGTX(90)	Missile/bomber target index number

INPUT FROM ALOCTAR

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
CTGTMOD	INDEXNO	Index number of target
	VTO	Original target value
	M	Number of hardness components ($\leq$2)
	H(2)	Lethal radius of each component (ground burst)
	HA(2)	Lethal radius of each component (air burst)
	VO(2)	Original value of each component
	IDHOB	Desired height of burst
	NK	Number of time periods ($\leq$5)
	FVAL(5)	Fraction value escaping in each period
	TAU(5)	Time ending each period
	NUM	Number of weapons assigned
	LDN	Number of words read from ALOCTAR file from /DYNAMIC/
	LDNO	Number of words for target records on scratch file
	IG(30)	Group number of assigned weapons
	KORR(30)	Weapon penetration corridor
	RVAL(30)	Relative value of weapon allocation

Table 17. (Part 3 of 5)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
CTGTMOD (cont.)	PEN(30)	Weapon penetration probability
	TOA(30)	Weapon time of arrival on target
*	DE(30)	Survival probability of target after arrival of weapon I
	TIMEVAL(30)	Survival probability of time-dependent target value after arrival of weapon I
	NTYPE	Target type name (same as IHTYPE on ALOCIAR file)
	INDTYPE	Index of target type used for summarization
	VTOC	Total value of all components of complex target
	ITGTINDX	Number of targets being processed; in UNPACKER, number of target for which PLANTAPE weapons must be found
	IBGINDX	Index number of target for which PLANTAPE weapons must be found by UNPACKER
	ISAL(30)	Salvo number of weapon (zero for bombers and non-salvoed missiles)

INPUT FROM BASFILE

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
CWPNMOD	NTYPES	Number of weapon types
	CEP(100)	Weapon CEP (nautical miles)
	ICLASS(100)	Weapon class index (1 for missile, 2 for bombers)
	REL(100)	Weapon reliability
	NAMEWPN(100)	Weapon type name
	FUNC(100)	Weapon function code
	ITYPE(250)	Weapon type index
	DBL(250)	Probability that the

* The remaining segment of /CTGTMOD/ is used for internal processing.

Table 17. (Part 4 of 5)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
CWPNMOD (cont.)	DBL(250) (cont.)	weapon survives before launch
	NPASS	Number of times the program runs
MIS	NWHDS(40)	Number of type 1 bombs
	NDECOYS(40)	Number of terminal decoys
	MISDEF	Number of terminal ballistic missile interceptors (minus the number of interceptors if a DEFALOC allocation) (ALOCTAR file)
	PKTX	Probability of kill of a terminal ballistic missile interceptor

INPUT FROM PLANTAPE

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
PLANREC	JSIDE	Side*
	LGROUP	Group index
	LPEN	Penetration corridor number
	JSORTIE	Missile record counter or bomber/tanker sortie number
	JUNIT	Zero or base index number
	JVEHIC	Zero or vehicle index number
	LCIAS	Weapon class (1, 2, or 3)
	JWPNTYPE	Weapon type index
	JREG	Zero or launch region
	JALERT	Alert status
	JPAYLOAD	Zero or payload index
	JDEPEN	Zero or depenetration
	LTOT	Number of missiles or total
	LPLAN	Number of targets or total number of planned events
	G(1)	Missile time of launch
	G(2), G(3)	Not used
	MLO(2)	Lower plot markers for sortie

* This is the second word on a PLANTAPE record. The first word, ISORTN, is read into common ISORTN.

Table 18. (Part 4 of 7)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
P (cont.)	YIELD(250)	Weapon yield in megatons (from BASFILE)
	CLASTYPE(250)	Summarization index for aggregating types into classes
	NALLTYPE(I,J) (7,270)	Number of weapons of category I scheduled against targets of type J
	SKDWPTYP(I,J) (7,270)	Number of weapons of category I scheduled against targets of type J
	DELWPTYP(I,J) (7,270)	Yield from weapon of category I against targets of type J
	ALLTYPE(I,J) (7,270)	Number of weapons of category I delivered to targets of type J
	NALLCLAS(I,J) (7,50)	Number of weapons of category I scheduled against targets of class J
	SKDWPCL(I,J) (7,50)	Yield from weapon of category I scheduled against targets of class J
	DELWPCL(I,J) (7,50)	Yield from weapon of category I delivered against targets of class J
	ALLCLAS(I,J) (7,50)	Number of weapons of category I delivered against targets of class J
	NAMECLAS(I) (50)	Name of class I
	NOFCLAS(I) (50)	Number of targets in class I
	VOCLAS(I) (50)	Total target value for class I
	VDESCLAS(I) (50)	Time-dependent value of target destroyed in class I
	VESCCLAS(I) (50)	Time-dependent value of target remaining in class I
	VREMCLAS(I) (50)	Time-dependent value of target remaining in class I
	SURVCLAS(I) (50)	Percent of target value surviving in class I
	SKEDCLAS(I) (50)	Megatons scheduled for target class I
	DELCLAS(I) (50)	Megatons delivered to class I targets

Table 18. (Part 5 of 7)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
P (cont.)	DESTNOCL(I) (50)	Value of target destroyed in class I
	PCDESTCL(I) (50)	Percent of target value destroyed in class I
	NSUMCLAS(I) (50)	Number of weapons allocated to targets in class I
	SUMCLAS(I) (50)	Number of weapons delivered to targets in class I
	KCLSNGL(I) (50)	Number of targets in class I attacked alone
	KCLSCOMP(I) (50)	Number of targets in class I attacked as part of a complex
	SKEDALL(J) (7)	Total megatonnage scheduled for weapon category J
	DELALL(J) (7)	Total megatonnage delivered from weapon category J
	NAMETYPE(K) (250)	The definitions of these arrays exactly parallel those of arrays NAMECLAS through KCLSCOMP, except that they are for target type K (instead of class I)
	.	
	.	
	.	
	KTYPCOMP(K) (250)	
	NWPNTYP(J) (7)	Total number of weapons scheduled from category J
	XWPNTYP(J) (7)	Total number of weapons delivered from category J
P		As used in MAKEPRFL and MKTARTAB
	PL(14038)	Work area for sorts
HOB	IHOB(90)	Storage for height of burst code
	LXMYHOB	Logical array giving HOB (packed into one word)
KEY	KEYJA	Packing key used for most significant part of target number in (JLINK)
	KEYJB	Packing key used for least significant part of target number (JLINK)

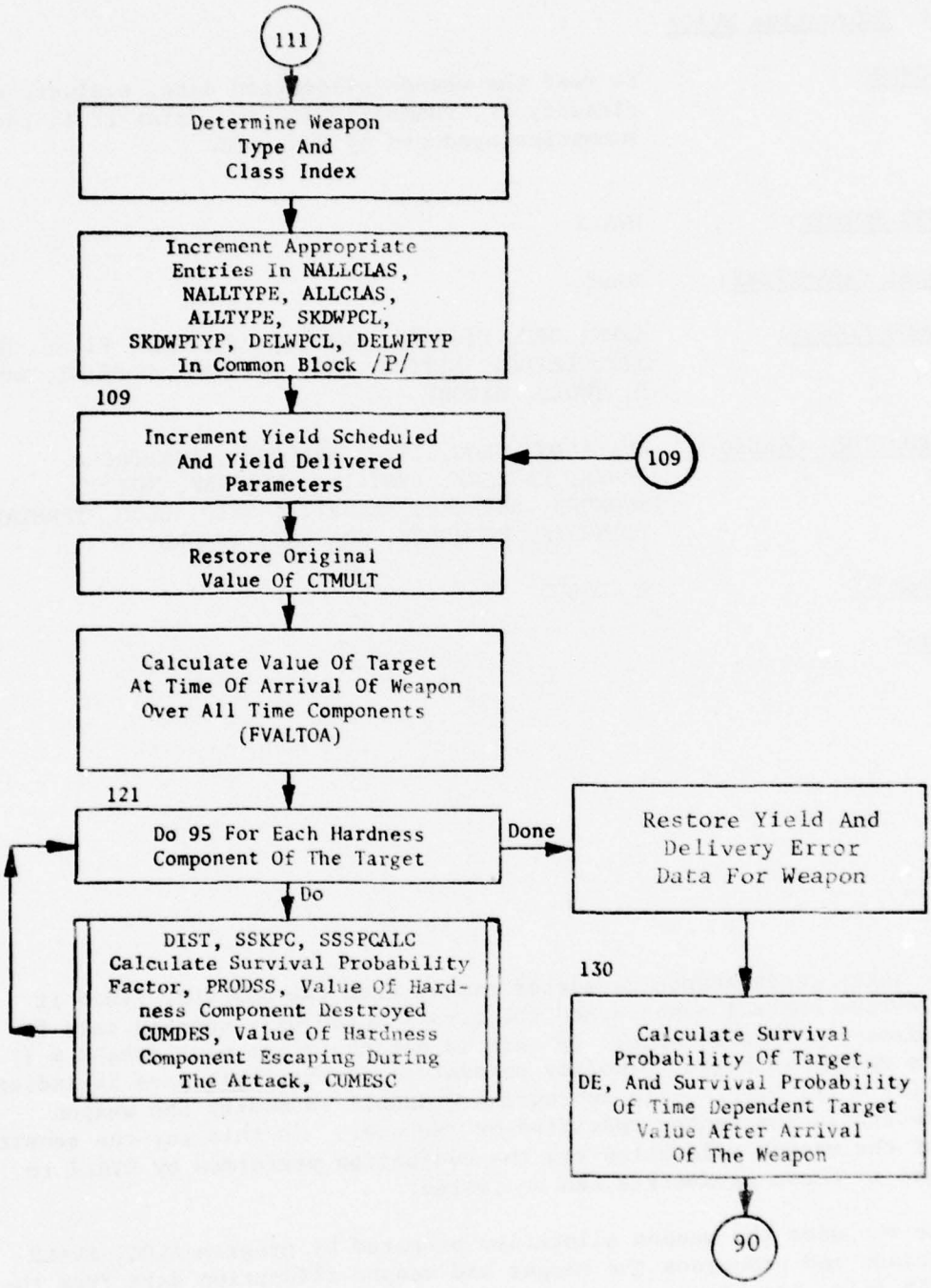


Figure 61. (Part 3 of 3)

4.9 Subroutine EVAL2

PURPOSE: To read the weapon allocation data, evaluate and classify it, summarize it, and print it in the summaries produced by EVALALOC.

ENTRY POINTS: EVAL2

FORMAL PARAMETERS: None

COMMON BLOCKS: ASMT, CPF, CTGTMOD, CWPNMOD, FILABEL, FILES, HOB, ITP, LATLON, LITTLE, MIS, MYIDENT, NOPRINT, OPT, P, TWORD, WAROUT

SUBROUTINES CALLED: EVALPLAN, GLOG, ITLE, LREORDER, MAKEPRFL, ORDER, PAGESKP, PRNTFILE, RDARRAY, RDWORD, REORDER, SETREAD, SETWRITE, SKIP, SLOG, TERMTAPE, TGTMODIF, WPNMODIF, WRARRAY, WRWORD

CALLED BY: EVALALOC

Method:

EVAL2 first reads weapon parameter arrays from the BASFILE. Then it initializes control numbers and the arrays which will contain data for the summary prints to zero. If this is not the first pass (NPASS $\neq$ 1) or the option to begin to modify parameters on the first pass is indicated (JOPT(2) = 1), EVAL2 calls subroutine WPNMODIF to modify the weapon parameters in the manner specified by the user. In this way the sensitivity of the weapon allocation and the evaluation performed by EVAL2 to changes in these parameters can be tested.

If the run uses the weapon allocation prepared by program ALOC, EVAL2 then reads and processes the target and weapon allocation data from the ALOCTAR file, one block at a time. Otherwise, it reads these data from the SCRATCH file prepared by subroutine PACK using the ALOCTAR file and the PLANTAPE.

THE CONTENTS OF THESE PAGES INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

4.11 Subroutine PACK

PURPOSE: PACK with its associated subroutines BOMRPAKR, MISLPAKR, SEARCH, and UNPACKER converts the sortie-order weapon allocation on the PLANTAPE to target-ordered allocation and writes this new allocation onto a SCRATCH file which is treated like the ALOCTAR file in subsequent processing by EVAL2. PACK is used only in post-PLANOUT operation of EVALALOC.

ENTRY POINTS: PACK

FORMAL PARAMETERS: None

COMMON BLOCKS: BINSCH, CTGTMOD, FILABEL, FILES, HOB, KEY, ISORTN, ITP, LATLON, MYIDENT, MYLABEL, N, NOPRINT, OPT, P, PLANREC, TWORD

SUBROUTINES CALLED: ABORT, BOMRPAKR, IPUT, KEYMAKE, MISLPAKR, ORDER, RDWORD, RDARRAY, REORDER, SETREAD, SETWRITE, SKIP, TERMTAPE, UNPACKER, WRARRAY, WRWORD

CALLED BY: EVALALOC

Method:

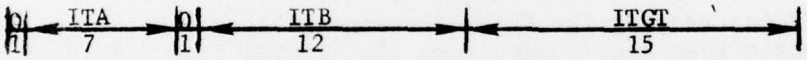
Before PACK can reorder the PLANTAPE weapon allocation into target order, it must obtain the target order from the ALOCTAR file. It does this by reading the target index number and target number one target at a time from the ALOCTAR file and by packing these into the next unfilled word of the JORDER array until the whole file has been read (see figure 64). Then it uses ORDER and REORDER to reorder the array in target index number, INDEXNO order (since the PLANTAPE does not contain target number data). PACK is then ready to process the PLANTAPE, which it does one record at a time. Tanker records on the PLANTAPE are ignored by PACK since they contain no weapon allocation data.

PACK packs data for each event on the PLANTAPE which is a target event. When processing a bomber record, it calls subroutine BOMRPAKR once for each event involving a target burst to pack the weapon allocation data into the next unused words in the JLINK, KLINK, LLINK, and SLINK arrays and to determine the target number on the ALOCTAR file which correspond to this target index number. Subroutine MISLPAKR is called by PACK to perform the analagous function for missile target events. This process is repeated for every missile and bomber record on the PLANTAPE until the JLINK, KLINK, LLINK, and SLINK arrays are filled. Then PACK reorders these arrays in target number order. If the arrays are filled for the first time, they are simply written onto a SCRATCH file in the new order. If not, they are merged with the packed data which must be read

from the previously written SCRATCH file; the merged data, which is in target number order, is written onto a new SCRATCH file. Then PACK continues reading and processing PLANTAPE records as before until the packed arrays are again filled or until the PLANTAPE has been completely read and all the weapon allocation data are contained in target number order on one scratch file. (If the total number of warheads is no greater than 3,500, no SCRATCH file is needed or written.)

The final phase of processing by PACK consists of reading the ALOCTAR file one target block at a time, calling UNPACKER to unpack the associated weapon allocation data for the target from the previously written SCRATCH file, and writing the ALOCTAR target data and PLANTAPE weapon allocation data together on a new SCRATCH file. When all targets on the ALOCTAR file and weapon data from the PLANTAPE have been processes in this way, PACK writes the ALOCTAR end-of-file record on the ALOCTAR-like SCRATCH file and returns control to EVALALOC.

Subroutine PACK is illustrated in figure 65.



ITGT = Target number
ITA = INDEXNO/2048
ITB = INDEXNO - ITA*2048
(i.e., INDEXNO = ITA*2048 + ITB)

Figure 64. Format of JORDER Array Element

THIS PAGE INTENTIONALLY LEFT BLANK

4.12 Subroutine SEARCH

PURPOSE: To determine the target number from the ALOCTAR file which corresponds to the index number of a PLANTAPE target.

ENTRY POINTS: SEARCH

FORMAL PARAMETERS: INDEXNO - The target index number for which the target number is sought
ITGT - The target number corresponding to INDEXNO and determined by SEARCH

COMMON BLOCKS: BINSCH, KEY

SUBROUTINES CALLED: IGET

CALLED BY: BOMRPAKR, MISLPAKR

Method:

Subroutine PACK sets up the JORDER array, which is a list of packed words. Each word in the array contains a target index number (INDEX) and a target number (ITGT) corresponding to a target on the ALOCTAR file, and the array is in increasing order by INDEX. SEARCH uses a binary search to find the word in the JORDER array which has target index number INDEXNO or which is the index number closest to INDEXNO but not larger than INDEXNO. This word has index NNOW. SEARCH calls subroutine IGET which unpacks JORDER (NNOW) to obtain ITGT.

The first time SEARCH is called it determines the parameters NLOG, INT, and NOW used in the binary search.

Subroutine SEARCH is illustrated in figure 66.

program ALOC, one record at a time, calling subroutines PROCSIMP, PROCMULT, or PROCCOMP to process each simple, multiple, or complex target, respectively. These routines extract the essential data from the input record and, for each weapon strike, cause a record to be written by PROCSIMP on the INTERMED file in a standard form that can be sorted later in ALOCO2. All of this processing, including that done by subroutine DGZSEL for complex targets, takes place during the ALOCO1 phase of the program while the ALOCTAR file is being read in.

The main function of PROCSIMP is to write the INTERMED file which is the input for ALOCO2. Overlay ALOCO2 sorts this file by penetration corridor within weapon group, does the final processing, and writes the TMPALOC file (see figure 71). Figure 72 is the detailed flowchart for ALOCO1.

5.5 Common Block Definition

The external common blocks used by overlay ALOCO1 in processing input/output (I/O) files are shown in table 20. Table 21 describes the additional common blocks used internally by overlay ALOCO1. Table 22 describes the common blocks used by ALOCO2.

Table 20. Overlay ALOC01 External Common Blocks (Part 1 of 4)

INPUT DATA FROM BASFILE

BLOCK	VARIABLE OR ARRAY*	DESCRIPTION
MASTER	IHDATE	Date of run initiation
	IDENTNO	Run identification number
	ISIDE	Attacking side
	NRTPT	Number of route points
	NCORR	Number of penetration corridors
	NDPEN	Number of depenetration corridors
	NRECOVER	Number of recovery bases
	NREF	Number of directed refuel areas
	NREG	Number of command and control regions
	NTYPE	Number of weapon types
	NGROUP	Number of weapon groups
	NTOTBASE	Total number of bases
	NPAYLOAD	Number of payload types
	NASMTYPE	Number of ASM types
	NWHDTYPE	Number of warhead types
	NTANKBAS	Number of tanker bases
	NCOMPLEX	Number of complex targets
	NCLASS	Number of weapon classes(2)
	NALERT	Number of alert conditions (2)
	NTGTS	Number of targets
	NCORTYPE	Number of penetration corridor types
	NCNTRY	Number of distinct country codes
FILES (also used by ALOC02)	TGTFILE(2)**	Target data file
	BASFILE(2)	Data base information file
	MSLTIME(2)	Fixed missile timing file
	ALOCTAR(2)	Weapon allocation by targets file
	TMPALOC(2)	Temporary allocation file
	ALOCGRP(2)	Allocation by group file
	STRKFIL(2)	Strike file

* Parenthetical values indicate array dimensions. All other elements are single-word variables.

** In two-word arrays, first word is filehandler buffer usage number; second word is maximum file length in words. Single variables are logical tape unit numbers.

Table 20. (Part 2 of 4)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
WGROUP	IREG(250)	Command and control region
	ITYPE(250)	Type index (LTYPE)
	SBL(250)	Probability of survival before launch
	YIELD(250)	Weapon yield (megatons)
	CEP(250)	CEP (nautical miles)
	REL(100)	Reliability
	CCREL(20)	Command and control reliability by command and control region
ASMT	IAMSM(250)	Index to ASM table for each group
	YLD(50)	Warhead yield from warhead table
	IWHDASM(20)	Index to warhead table for ASM warhead
	RELASM(20)	ASM reliability
	CEPASM(20)	Circular error probable for ASM delivery
TARGET*	TGTNAME	Target name
	INDEXNO	Target index number
	DESIG	Target designator code
	TASK	Target task and country owner codes
	CNTRYLOC	Target country location code and vulnerability
	FLAG	Target flag code
	TGTMULT	Target multiplicity
	TGTLAT	Target latitude
	TGTLONG	Target longitude
	TGTRAD	Target radius (nautical miles)
	VTO	Original target value
	M	Number of hardness components
	H(2)	Lethal radii by hardness component**
	FVALH1	Fraction of value of first hardness component
	NK	Number of time components
	FVAL(5)	Fraction of value escaping in each time period
	TAU(5)	Time ending each time component

* TARGET is the 34-word record contained on the BASFILE for each complex target component.

** Ground burst radius in lower 18 bits; air burst radius in upper 18 bits in units of .0001 nautical miles.

Table 20. (Part 3 of 4)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
TARGET (cont.)	IHCLASS	Target class name
	ICLASS	Target class number
	IHTYPE	Target type name
	TARGET(30)-TARGET(34)	Not used
MULTTGT*	NAME	Target name
	INDEX	Target index number
	DSIG	Target designator code
	TSK	Target task and country owner codes
	CNTRLC	Target country location code
	FLG	Target flag code
	TLAT	Target latitude
	TLONG	Target longitude
HOB*	LXISPEC(3)	Logical array set true for weapon type with specified HOB
	LXIWHOB(3)	Logical array containing user specified HOB by weapon type
	IHVULN(255)	Vulnerability numbers in target set
WTYPE**	IWTYPE(250)	Weapon type index for each group (equal to ITYPE in block /WGROUP/)

INPUT DATA FROM ALOCTAR

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
DYNAMIC	TGTNAME	Hollerith target name
	INDEXNO	Index number of target
	DESIG	Target designator code
	TASK	Target task and country owner codes
	CNTRYLOC	Target country location code and vulnerability
	FLAG	Target flag code
	TGTMULT	Target multiplicity (original)
	TGTLAT	Target latitude
	TGTLONG	Target longitude
	TGTRAD	Target radius
	VTO	Original target value
	M	Number of hardness components (≤ 2)

* MULTTGT is the 8-word record contained on the BASFILE for each multiple target element.

** These blocks are also used in ALOCO2.

5.7.1 Subroutine SETPG

PURPOSE: This subroutine controls the paging of target data in core memory.

ENTRY POINT: SETPG

FORMAL PARAMETERS: IBETA - number of targets written

COMMON BLOCKS: C1, C3, ITP, MYIDENT, TWORD

SUBROUTINES CALLED: RDARRAY, SETREAD, TERMTAPE

CALLED BY: WRRDSTRK

Method:

Overlay ALOCO2 wrote target related data onto a scratch disk file (unit number 24) in target number sort order. When called, SETPG will quarantee desired target data are stored within working arrays ITD1 to ITD6. Up to 3,000 entries may be held in working storage. If the desired target is not contained within working arrays, SETPG calculates where the desired data resides within the disk file and simply reads (a rewind is necessary in some cases), or skips the proper number of records.

Subroutine SETPG is illustrated in figure 94.

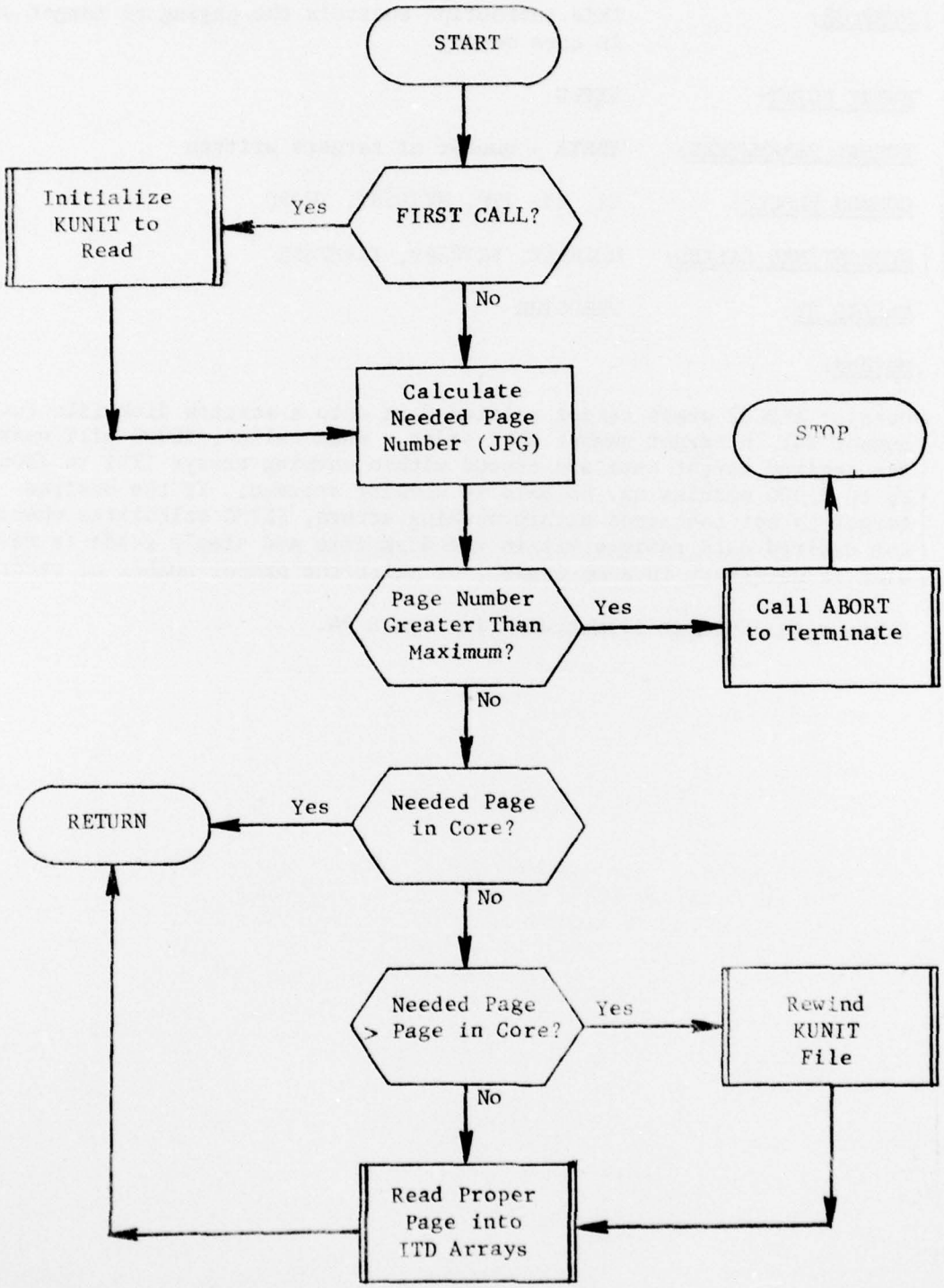
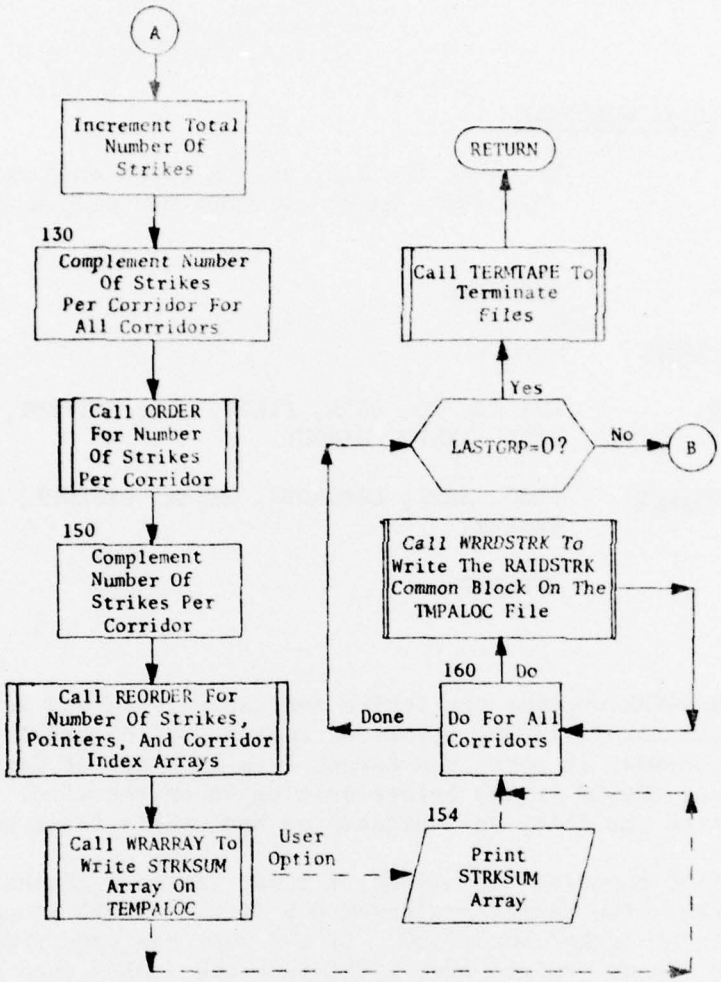


Figure 94. Subroutine SETPG



Notes:

1. ICORIN(NC) is an array of the corridor index numbers. IPT(NC) contains the I-index in the IWD array of the first item for the corridor. Both arrays are later reordered in the sequence determined by ordering NSTRK.

2. When all items in the IWD array have been processed, those for the current group (i.e., those that have not yet been written on the output tape) are moved to the beginning of the array, and the rest of the array is filled from the sort tape.

3. As the items are processed and when a new weapon group number is encountered, the tallying, sorting, and tape writing for the computed group is done. Unless ILASTGRP=1, the program then returns to process the remaining items.

Figure 95. (Part 2 of 2)

5.7.3 Subroutine WRRDSTRK

PURPOSE: To write the /C2/ common block onto the TMPALOC file for a given corridor and weapon group.

ENTRY POINTS: WRRDSTRK

FORMAL PARAMETERS: None

COMMON BLOCKS: C1, C2, C3, DATA, FILES, HOB, IFTPRNT, ITP, KEYS, RAID, WTYPE, WAROUT

SUBROUTINES CALLED: GLOG, IGET, LREORDER, ORDER, REORDER, SETPG, SLOG, WRARRAY

CALLED BY: STRKOUT

Method:

Subroutine WRRDSTRK unpacks the strike and target data and writes a variable length record in the format of common /C2/ on the TMPALOC file. For missile records, it sorts the target data in order of decreasing relative values (RVAIM array) before writing it on the file. It can optionally print the data, as indicated on the user's input card.

Local array IHOB contains the height of burst (false = ground; true = air) for each strike. The target preferred HOB (i.e., IDHOB) is packed into the fourth target packed word ITD3. If the user has specified a HOB for a weapon type (as determined from ISPEC in block /HOB/) then the preferred HOB is overridden by the user specified HOB (IWHOB in block /HOB/). The vulnerability codes are not packed with the other target information because each code would require 24 bits of storage. Rather, the index of the code in the IHVULN table in block/HOB/ is packed into the fifth packed target data word ITD5. The index ranges in value from one to 63; therefore it requires only six bits of storage. After unpacking the index, WRRDSTRK adds the vulnerability number as the last four characters of variable CNTRYLOC for use by program PLANOUT.

Subroutine WRRDSTRK is illustrated in figure 96.