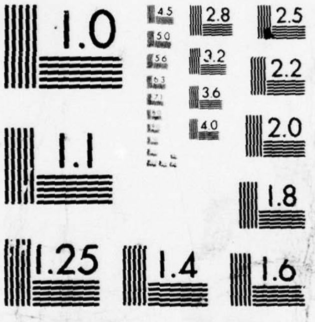
AD-A063 385 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON D C F/G 15/6
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK) PROGR--ETC(U)
NOV 78 D J SANDERS, P F MAYKRANTZ, J M HERRIN

UNCLASSIFIED CCTC-CSM-MM-9-77-V2-CHG-1 NL

1 OF 1

AD
A063385

END
DATE
FILMED
3--79
DDC


1.0
4.5
5.0
5.6
6.3
7.1
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

DEFENSE COMMUNICATIONS AGENCY
COMMAND AND CONTROL
TECHNICAL CENTER ✓
WASHINGTON, D. C. 20301

IN REPLY REFER TO: C314

TO: RECIPIENTS

-AD-A050081

SUBJECT: Change 1 to Program Maintenance Manual CSM MM 9-77, Volume II, Weapon/Target Identification 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

82 Enclosures
Change 1 pages

J. DOUGLAS POTTER
Assistant to the Director
for Administration

V3 A055 847
[illegible]
A05t [illegible]

EFFECTIVE PAGES - NOVEMBER 1978

This list is used to verify the accuracy of CSM MM 9-77 Volume II after change 1 pages have been inserted. Original pages are indicated by the letter O, and change 1 pages by the numeral 1.

Page No.	Change No.
Title Page	0
ii-iii	0
iv	1
v-vi	0
1-2	0
3	1
4	0
5-6	1
7-8	0
9-10	1
10.1-10.2	1
11-46	0
47-56	1
57-58	0
59-64	1
65-76	0
77-78	1
79-80	0
81-86	1
87-88	0
89-90	1
91-104	0
105-106	1
107-112	0
113-114	1
115-120	0
121-124	1
125-161	0
162-167	1
167.1-167.2	1
168-172	1
173-178	0
179-202	1

Section Page

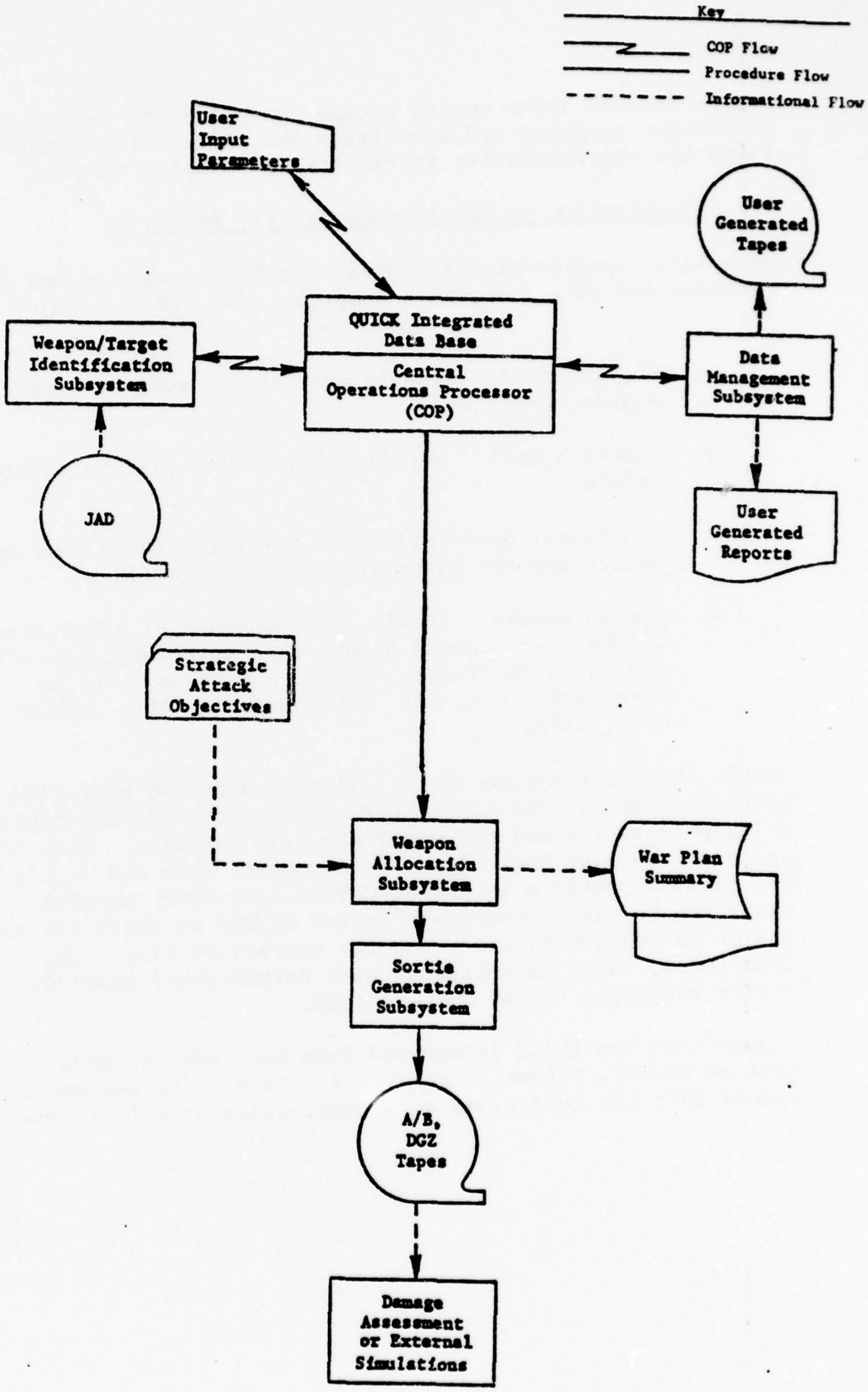

Key
COP Flow
Procedure Flow
Informational Flow
User Input Parameters
User Generated Tapes
QUICK Integrated Data Base
Central Operations Processor (COP)
Weapon/Target Identification Subsystem
Data Management Subsystem
JAD
User Generated Reports
Strategic Attack Objectives
Weapon Allocation Subsystem
War Plan Summary
Sortie Generation Subsystem
A/B, DGZ Tapes
Damage Assessment or External Simulations

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

Module PLANSET forms weapon groups, prepares the target list for the allocator, computes and normalizes the class value factors and calculates the representative attributes for complex targets.

1.3 Organization of Maintenance Manual, Volume II

Each major section of this manual details a module along with the subroutines and functions which comprise the module. Major subsections are:

a. Module input - details what chains must be created prior to module execution

b. Module output - details what chains will be updated by each module

c. Functional description - details the macro function of the module and the associated major subroutines

d. Common blocks - detail the contents of all internal common blocks. All common blocks used to communicate with the COP are given in Program Maintenance Manual, Volume I, appendix A. These are: C10, C15, C20, C30, C40, C50, ERRCOM, INS, IPQT, OOPS, STRING

Within the QUICK system the COP is viewed as the operating program. Based on user direction, the COP will execute overlay links or modules which perform the objectives of the user requests. Each overlay link is called through knowledge of the command verb and within each link the first subroutine is called ENTMOD (for entry module). That is, there are as many subroutines called ENTMOD as there are modules. Confusion is avoided by executing the correct overlay link. Subroutine discussion, then, is initiated with ENTMOD whose meaning, or function, varies according to the overlay link.

Comments on the QUICK integrated data base can be found in Program Maintenance Manual, Volume I, section 2. It will be assumed within this manual that the reader has an understanding of QUICK's data base.

SECTION 2. JAD LOADING MODULE (JLM)

2.1 Purpose

JLM and its associated subroutines assemble data from the CCTC JAD files and manipulate and reformat the data in a structure which is acceptable to the QUICK system.

2.2 Input

JLM creates target records as directed by user card image input. This module normally is the first phase in the total creation of the integrated data base. For proper execution, the organizational portion of the data base must have been finalized.

Target records are obtained from input JAD files according to the format given in figure 3. Not all JAD entries are used by JLM, only those listed under the column labeled 'USED'.

A JAD format has a maximum of 336 characters of which only the first 258 have defined inputs. In those cases when JLM generates a JAD format tape, entries are placed within characters 289 through 336 (the third column of figure 3).

If the user desires to bypass the construction of the Assignment Table by using a BTL, BTB, or DBASSES format tape, characters 289 through 336 will contain information that will be used instead of created.

2.3 Output

JLM builds that portion of the organizational structure of the data base called the Assignment Table. This table shows the kind of target to be added to the data base and how it will be included. The table includes:

- The valid country code and what region and side the country is;
- The target classes for each side;
- The selection criteria for each target type based on category code, owner, location, and capacity or name;
- The TASK that corresponds to the target types; and
- The list of DESIGs (alphabetic portion) that are to be used.

From the Assignment Table, the user selects targets from the JAD input file and creates targets (record TARGET) within the gaming portion of the integrated data base. For each created target, all linkage is properly updated.

COLS.	ITEM	USED*	CREATED BY JLM
1-5	Category Code	JCATCODE	
6-9	WAC No.	JWACNO	
10-15	BE No.	JBENO	
16-20	Blank		
21-58	Name	JNAME(21-26)	
59-64	Major number	JMAJOR	
65-88	Complex Name		
89-94	Minor number	JMINER	
95-118	Concentration Name		
119-125	Latitude DDMMSS N/S	JLAT	
126-133	Longitude DDDMMSS E/W	JLONG	
134-135	World Division		
136-137	Sub Div		
138-139	Country Location	JLOC	
140-141	Special Region	JAD14	
142-143	Region		
144-147	Blank		
148-149	Owner Country	JOWN	
150-151	Agency or Service owner		
152-153	U & S Cmd or Supreme All		
154-155	Component or All Regn CMD		
156-159	Severe VN		
160-163	Moderate VN	JVN	
164-167	Light VN		
168-171	Review Data yy mm		
172-175	ICOD yy mm		
176-190	Significance		
191-198	Capacity	JCAP	
199-200	Data Source		
201-204	Units of measure		

*Variables are named as used in common block JADREC.

Figure 3. JAD Format (Part 1 of 2)

General comments are:

- The side must be first
- The target class must be preceded by two slashes
- The target type must be preceded by a dash
- If minimum capacity or name is used, it must be preceded by a greater than symbol
- The lowest catcode must be preceded by one slash and if a range of catcodes are used the highest catcode is preceded by a dash
- TASK is preceded by an asterisk and DESIG by a comma
- Country codes are preceded by either OWNED or IN if the assignment is restricted

2.5.1.2 Subroutine PLAYERS. Similar to ALPHAS, subroutine PLAYERS reads the input clause and updates the Assignment Table. The generalized input command is:

```
ASSIGN PLAYERS side // region / country-code
     [, country-code . . .] [side // region . . .]
```

The side must be first, the region must be preceded by two slashes and the list of country codes must be preceded by a single slash.

2.5.2 Subroutine SELECT. The use of the SELECT verb (and hence the execution of overlay link (SELE)) instructs the JLM to select records from the JAD format input file according to the developed Assignment Table. The SELECT command has a maximum of six optional adverbs and are:

- WHERE - normal WHERE clause without OF or LIKE
- UNIT - used to define input unit if it is not 20
- ONPRINTS - causes the print of the output JAD format
- REPLACING or OMITTING - used to replace existing targets or to ignore duplicates
- ORDER - allows the user to specify the arrangement that the classes will be added to the integrated data base
- SETTING - used to set the value of TARDEF to allow for automatic assignment of values for attributes TARDEFHI and TARDEFLO

If the user desires to bypass the Assignment Table, the expanded functions of the adverbs UNIT and ORDER may be used and is indicated by adding 100 to the logical unit number in the UNIT clause. The ORDER clause now takes on additional meaning. The order of the target class names in the ORDER clause must correspond with the set number on the input file. Also the last two entries must be the SIDE (RED or BLUE) followed by the type of file being input, i.e., BTL, BTB, DBASSES.

If the Bypass option is used, all data on the input file is directly added to the data base using subroutine ADTOBASE. No preprocessing is done.

After user input definition, SELECT reads a JAD input record and queries the Assignment table (subroutine FNDTAR) to ascertain if the target should be added to QUICKs data base. In it is not to be added, the next JAD input record is read. Otherwise investigations are made for exclusion of the input record due to a WHERE clause. For each selected target record, data is written onto files 25 and 21.

After all target records have been selected, data files are sorted and read. For each record, subroutine ADTOBASE is called for the definition of the selected target record onto QUICKs data base.

Finally tests and code are made to quarantee the JAD records are in proper sort and additionally JAD selected records are printed if user directed.

2.5.2.1 Subroutine ADTOBASE. This subroutine adds the data from the selected JAD record onto QUICKs data base. In addition to inserting attribute values, this subroutine places the target record on the proper chains. This includes linkage under the vulnerability, class, type, and other headings as directed by the nature of the target.

2.5.3 Subroutine ASTERISK. This overlay removes targets from the integrated data base and flags all target records on the output JAD format file. If identical target records reside both within the integrated data base and the output JAD file, an asterisk (character position 294) is placed on that record within the JAD file.

The list of target record to be retained are defined within a KEEPING clause and has the form:

KEEPING lowdesig [- highdesig]

[, lowdesig [- highdesig] . . .]]

This clause consists of a list of DESIG ranges that are to be kept in the data base and flagged on the output file.

2.6 Common Block Definition

Common blocks used by JLM are outlined in table 1. Common blocks that communicate with the COP are given in appendix A of Program Maintenance Manual, Volume I.

THIS PAGE INTENTIONALLY LEFT BLANK

2.9 Subroutine SELECT*

PURPOSE: Select targets records from input JAD file

ENTRY POINTS: SELECT

FORMAL PARAMETERS: None

COMMON BLOCKS: ASNKEY, CLASSES, C10, C30, JADREC, OPTION, PRINSP, SIDES, TARDEF

SUBROUTINES CALLED: ADTOBASE, FINDCLASS, FINDSIDE, FMEDIA, FNDTAR, INSGET, KOMPCH, KRUNCH, NEXTTT, RANSIZE, SAMSET, SETDEF, SORTIT, XWHERE

CALLED BY: ENTMOD (of JLM)

Method:

Target records are selected from the input JAD files, added to QUICKs integrated data base (subroutine ADTOBASE) and a new output JAD file generated for use outside of the QUICK system (see figure 9). The selection process is user directed and outlined in subsection 2.5.

Besides the commons described in table 1 there are some special arrays used by SELECT. VADV holds the value for all legal adverbs. ADVPTR holds the adverbs and pointers from the input clauses. START holds the pointer to the beginning of the input clauses in the order described in VADV. PUNCT holds the pair of value pairs that correspond to the alphabetic prefix and the operator comma as they appear in the input. RECORD is a 336-character array which contains the JAD record before it is decoded into JADREC. LINES contain an incremented count of the items in 'subsets', based on CLASS.

Initially the input options are read. If the Bypass option is specified, input unit >100, the actual unit number is reset and processing goes directly to the 200 block of code. It is there that subroutine ADTOBASE is used to load the information onto IDS data base.

First a JAD format record is read. If the automatic generation of bomber defenses is desired and the record is a SAM site, SAMSET is called to save it on a special file (unit 19). The target then must meet the criterion defined by the Assignment table as tested by FNDTAR. The record must then pass any restrictions in the optional WHERE clause before it is decoded into JADREC. At this time the key reference code (ASNKEY) from FNDTAR is saved on the record which is put on a random file. The sortkey for this record is written onto file 21. This process continues until the input file is exhausted.

*First subroutine of overlay link SELE

The next step is to sort the random file and optionally create SAM complexes. The sort is done by subroutine SORTIT with the output defined on unit MIDLUN (21 or 23). The sort is primarily on DESIGA2 (found after FNDTAR was called by retrieving the first DESIGA2 for this type).

The DESIGA2 is retrieved so that the output tape will be in DESIG sort without necessarily doing a second sort. The file of selected JAD records is also sorted on region for the same reason. The SIDE/CLASS index (JSETNO) is used to keep the classes together within region for the output file and also to improve the efficiency of ADTOBASE. Attribute TYPE is included in the sort primarily for ADTOBASE efficiency but also for the output print. The NAME of the target is primarily there for the INDEXER module. After the sort is accomplished, KRUNCH is called to form SAM complexes if the option is on. The sorted file is then read and SETDEF is called, if the option is on, and ADTOBASE called to add the record to the integrated data base. The value for DESIG is now defined and the JAD record is output noting if the DESIG remains in sort (DESIG overflow could cause it to be out of sort). This is done for the entire sorted file. If yet out of sort, the JAD output file is resorted and put into the output file. If the optional prints are desired the JAD output file is reread and printed.

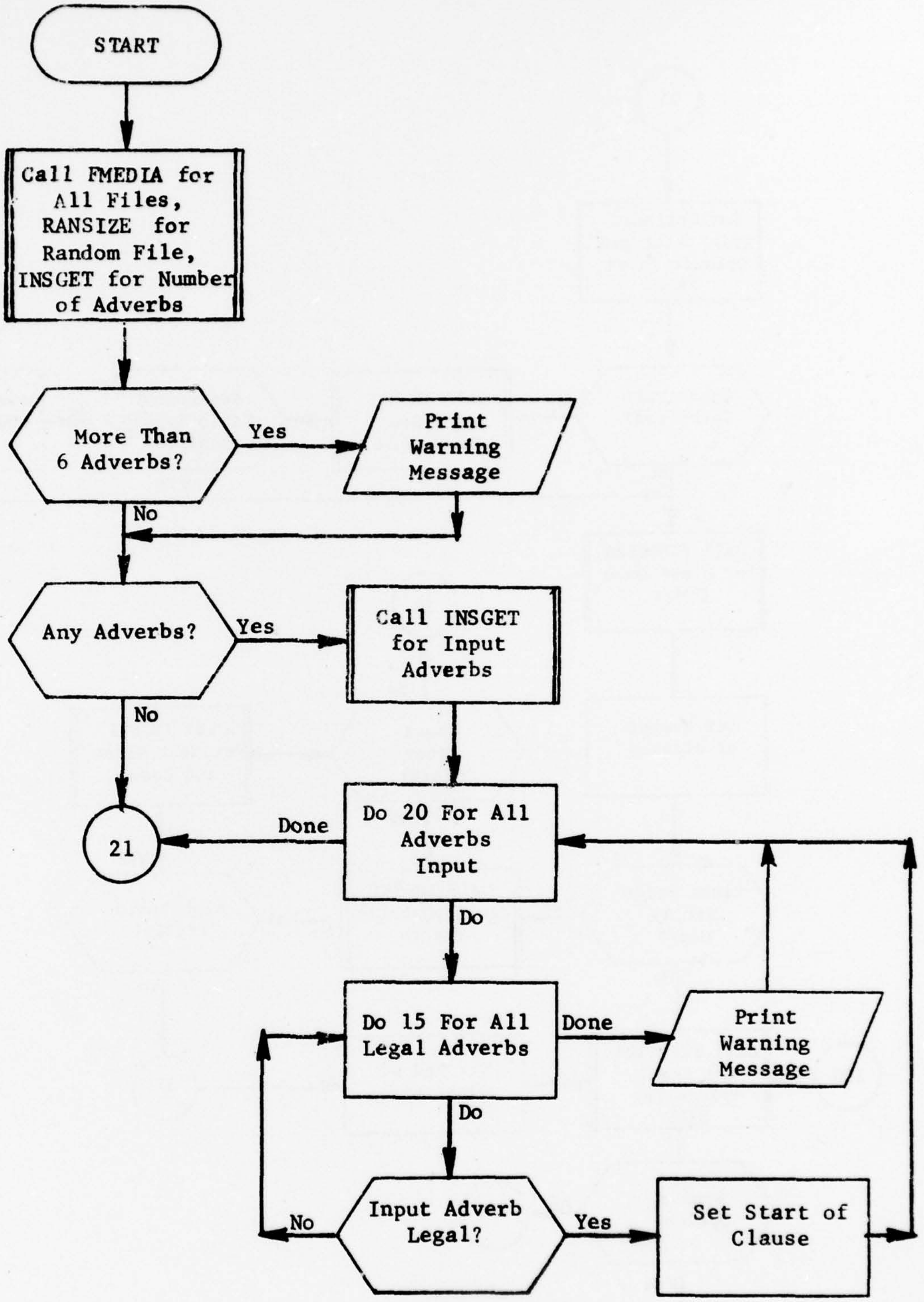

START
Call FMEDIA for All Files, RANSIZE for Random File, INSGET for Number of Adverbs
More Than 6 Adverbs?
Yes
Print Warning Message
No
Any Adverbs?
Yes
Call INSGET for Input Adverbs
No
21
Done
Do 20 For All Adverbs Input
Do
Do 15 For All Legal Adverbs
Done
Print Warning Message
Do
No
Input Adverb Legal?
Yes
Set Start of Clause

Figure 9. Subroutine SELECT (Part 1 of 6)

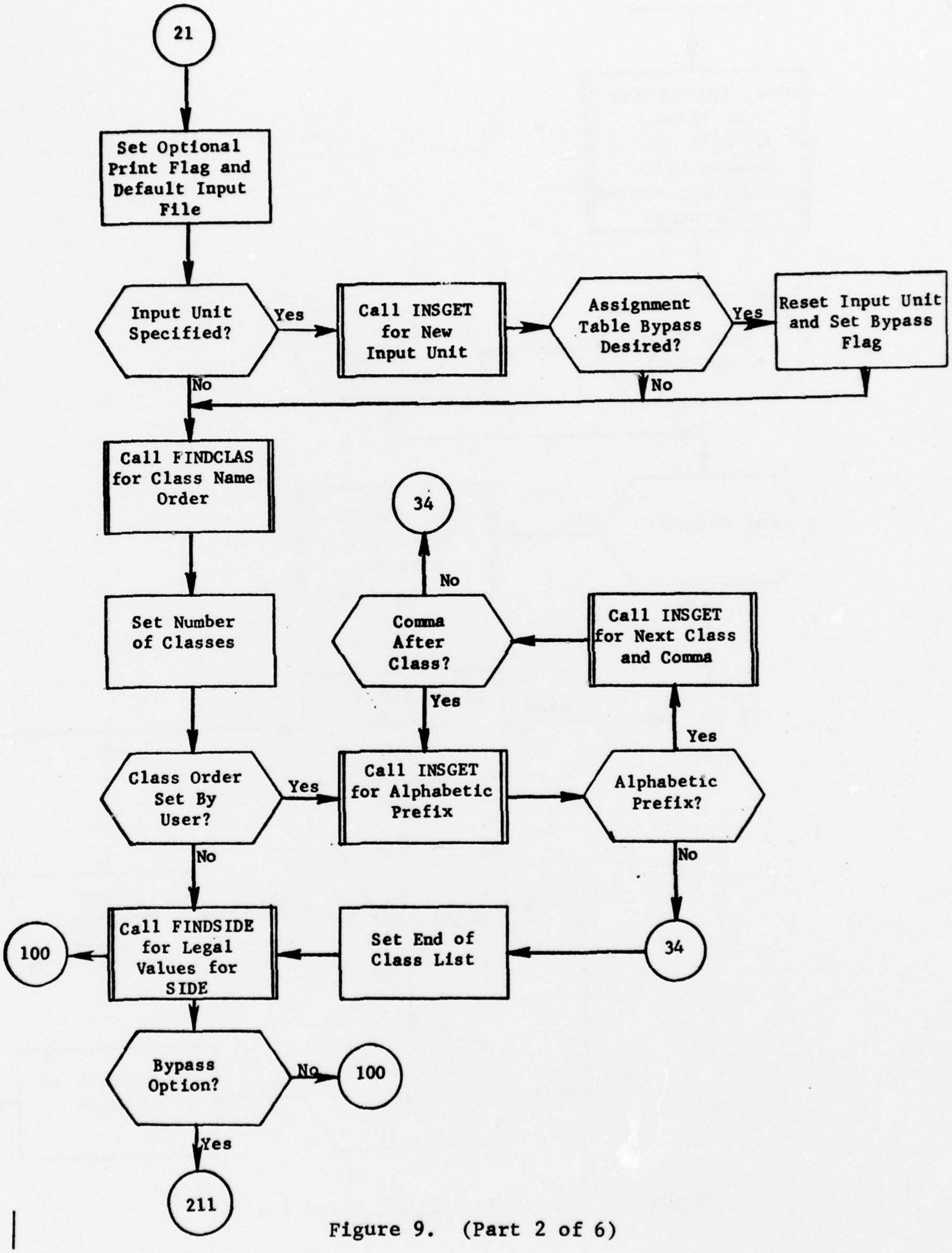

21
Set Optional Print Flag and Default Input File
Input Unit Specified?
Yes
No
Call INSGET for New Input Unit
Assignment Table Bypass Desired?
Yes
No
Reset Input Unit and Set Bypass Flag
Call FINDCLAS for Class Name Order
34
No
Comma After Class?
Yes
Call INSGET for Next Class and Comma
Yes
Set Number of Classes
Class Order Set By User?
Yes
No
Call INSGET for Alphabetic Prefix
Alphabetic Prefix?
No
100
Call FINDSIDE for Legal Values for SIDE
Set End of Class List
34
Bypass Option?
No
100
Yes
211

Figure 9. (Part 2 of 6)

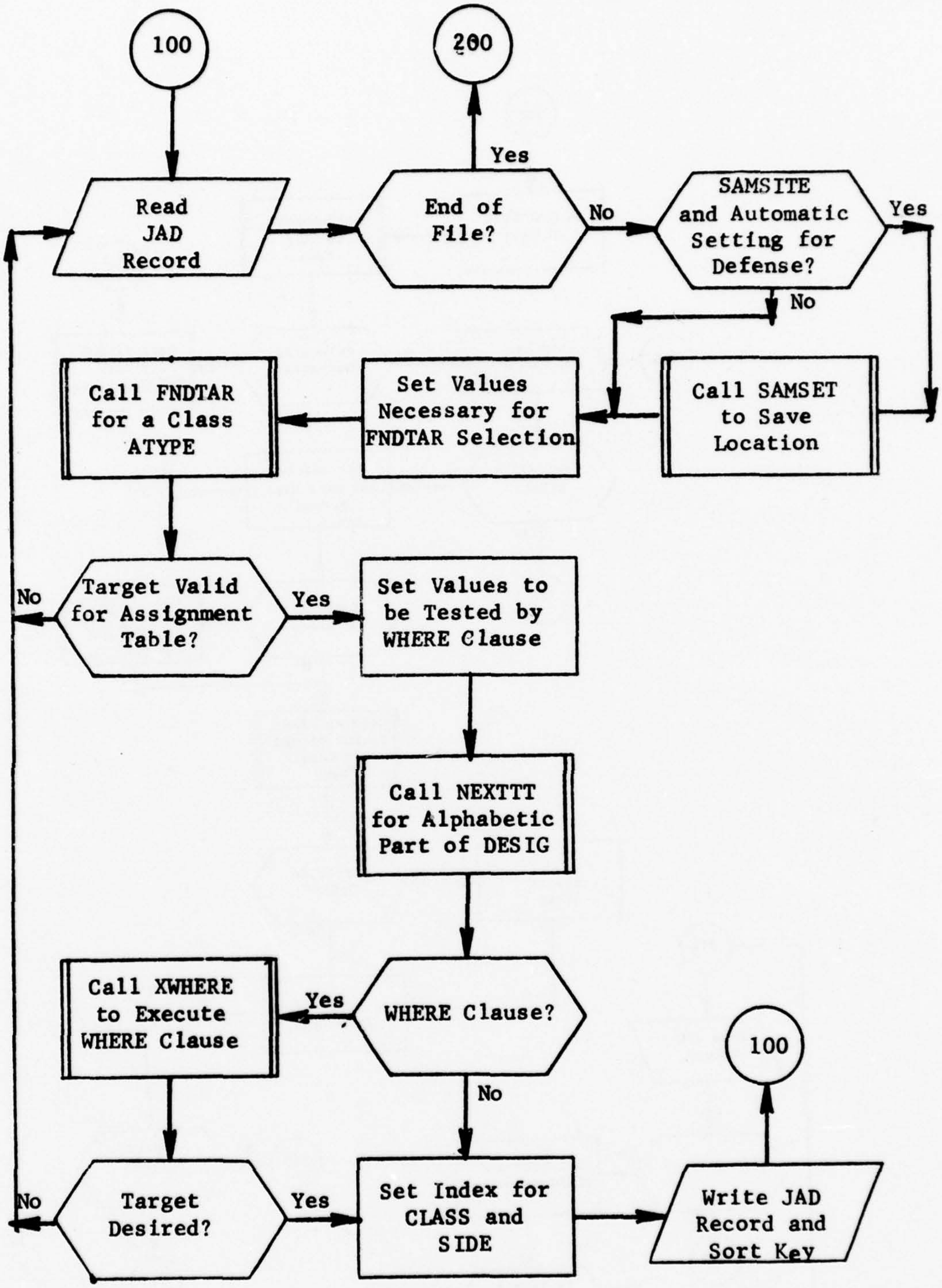

100
200
Yes
Read JAD Record
End of File?
No
SAMSITE and Automatic Setting for Defense?
Yes
No
Call FNDTAR for a Class ATYPE
Set Values Necessary for FNDTAR Selection
Call SAMSET to Save Location
No
Target Valid for Assignment Table?
Yes
Set Values to be Tested by WHERE Clause
Call NEXTTT for Alphabetic Part of DESIG
Call XWHERE to Execute WHERE Clause
Yes
WHERE Clause?
No
100
No
Target Desired?
Yes
Set Index for CLASS and SIDE
Write JAD Record and Sort Key

Figure 9. (Part 3 of 6)

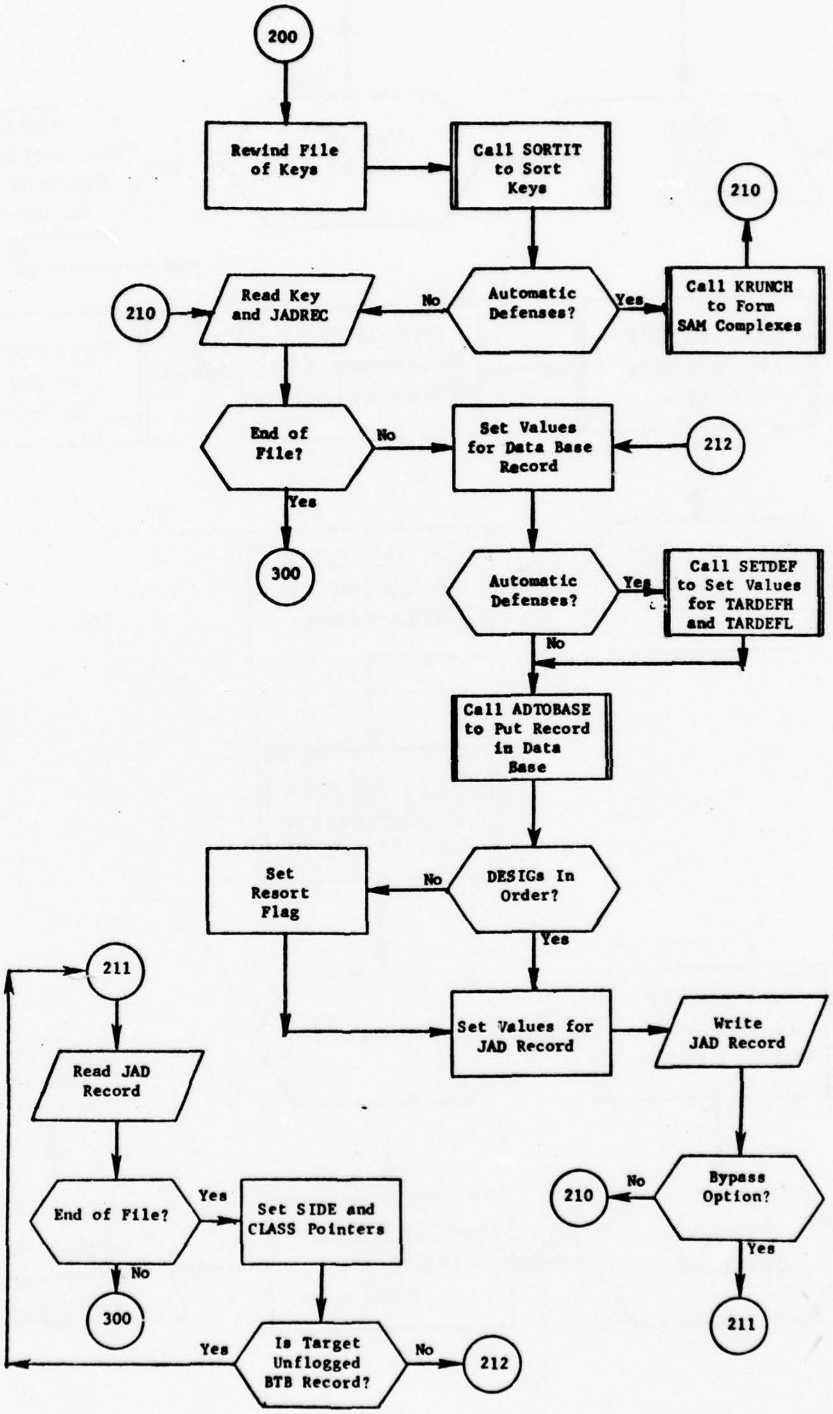


Figure 9. (Part 4 of 6)

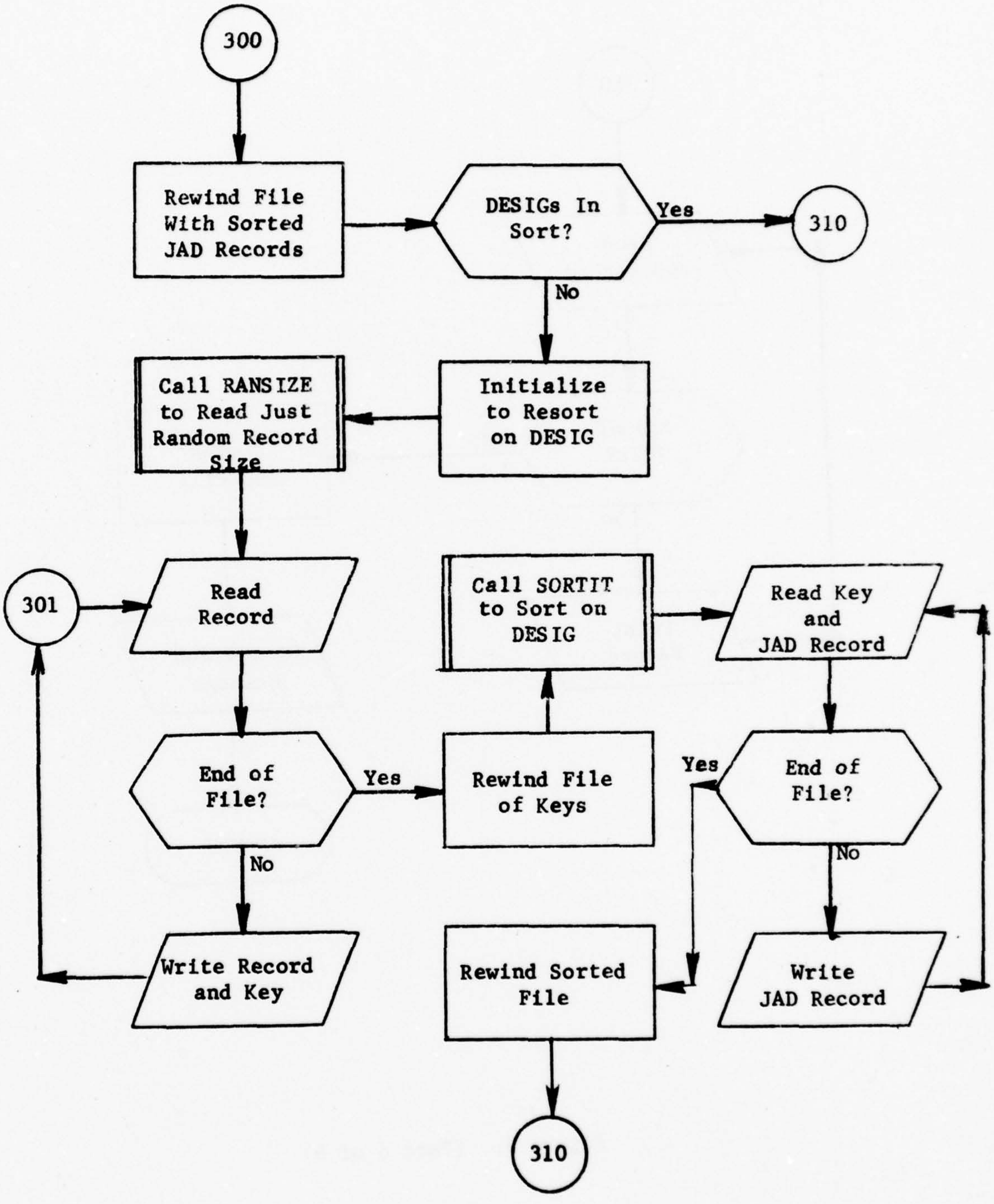


Figure 9. (Part 5 of 6)

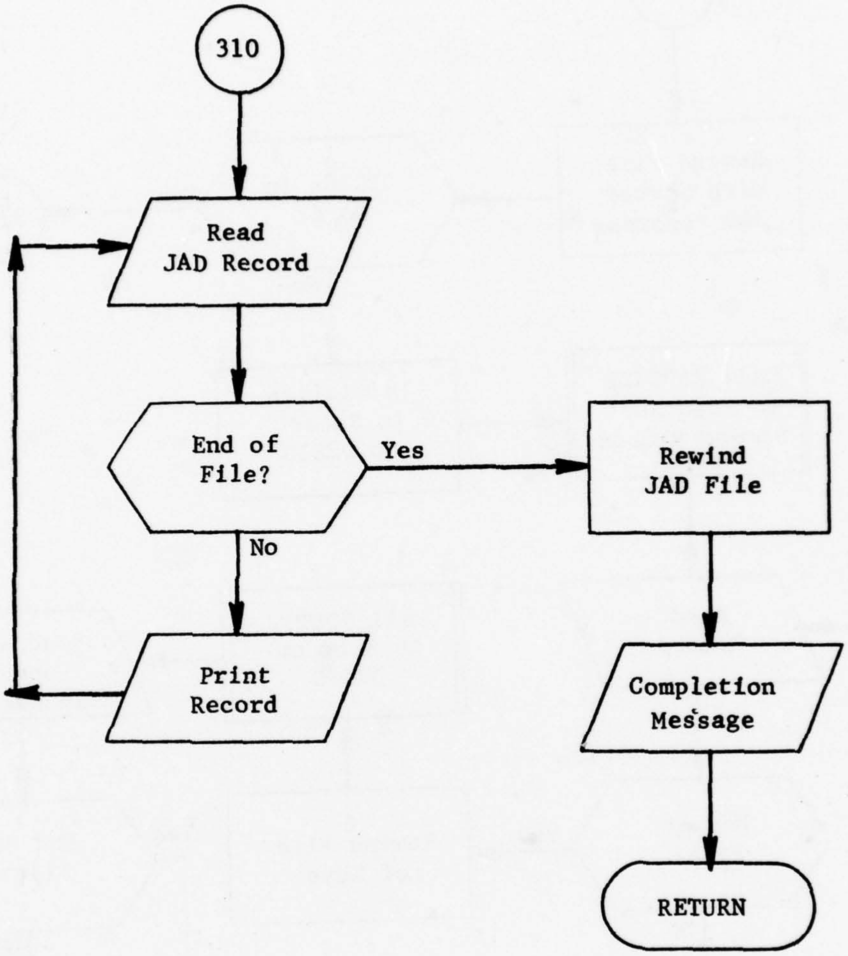


Figure 9. (Part 6 of 6)

2.9.1 Subroutine ADTOBASE

PURPOSE:	Add the selected JAD record to QUICKs integrated data base
ENTRY POINTS:	ADTOBASE
FORMAL PARAMETERS:	None
COMMON BLOCKS:	C10, C15, C30, JADREC, OPTION, TARDEF
SUBROUTINES CALLED:	DEFAULT, DLETE, GETDES, HDFND, HEAD, MODFY, NEXTTT, RETRV, STORE
CALLED BY:	SELECT

Method:

ADTOBASE is called to put the target described in common JADREC into QUICKs Integrated Data Base. In order to do this, the master of all the chains of which the TARGET Record is a detail of, must be found or in many cases created. Since the processing is very similar for most of the chains, the process has been generalized as much as possible. The general approach is to start with the headers and query the chains until the proper masters have been found or created for this TARGET record.

The first time that ADTOBASE is called the records on the ATRIB chain are checked in order to define the common C30 locations of the key attributes describing the master records. This is only done on the first call.

The first action normally performed in ADTOBASE is to define the array of values that describe the location in the data base where the TARGET record is to be linked. The linkage must be placed under the chains that contain the matching values for attributes TYPE, IREG, and CLASS. If the TARGET is a weapon, chains for weapon group, payload number and the weapon TYPE (defined under the weapon CLASS definition) must be searched for matching values.

The first step in retrieving the master records is to retrieve any headers that have not been retrieved already by the previous target.

Then the three chains (six for weapons) are searched for a record that has a key value matching that of the JAD record. If the record agrees with the one previously found, no searching is necessary. If the record does not exist, then it must be created.

Before any record is created in ADTOBASE the subroutine DEFAULT is called for that record type. DEFAULT will set all the default values for that record type. After this is done, the key value from JADREC are set and the record is created.

The TARGTY record must also be checked since it must be unique to more values than just TYPE. The values of country location and owner and vulnerability are also checked. If a match is not found on the entire chain, a new record is created. Note also that if the record had been created earlier it must now be modified to account for the additional unique values (VULN1, etc.).

For weapons there is one additional record that must be defined now, WEPSUB, the weapon subtype. As usual, if it does not already exist it is created. Note that DEFAULT is not called since the entire record consists of the Payload Table name.

Now if either the REPLACING or OMITTING options are used the TGTTGT chain is searched for targets with the same values for WACNO and BENO. Note that for some types of targets, this can result in an excessive amount of searching and that the options should be used sparingly. If a match is found OMITTING will cause a return to SELECT while REPLACING will delete the old record.

Now that the masters for the TARGET record have been defined the actual record can be added to the data base.

First, if the bypass option is not specified, subroutine GETDES completes the work that FNDTAR started using the reference code mentioned in ALPHAS, GETTAR, and SELECT.

With the DESIG now defined, and the default values defined, the values in JADREC are moved to common C30.

The target record is now created.

If the target is also a weapon then a MSBMTG record is also created. At this point ADTOBASE returns.

Subroutine ADTOBASE is illustrated in figure 10.

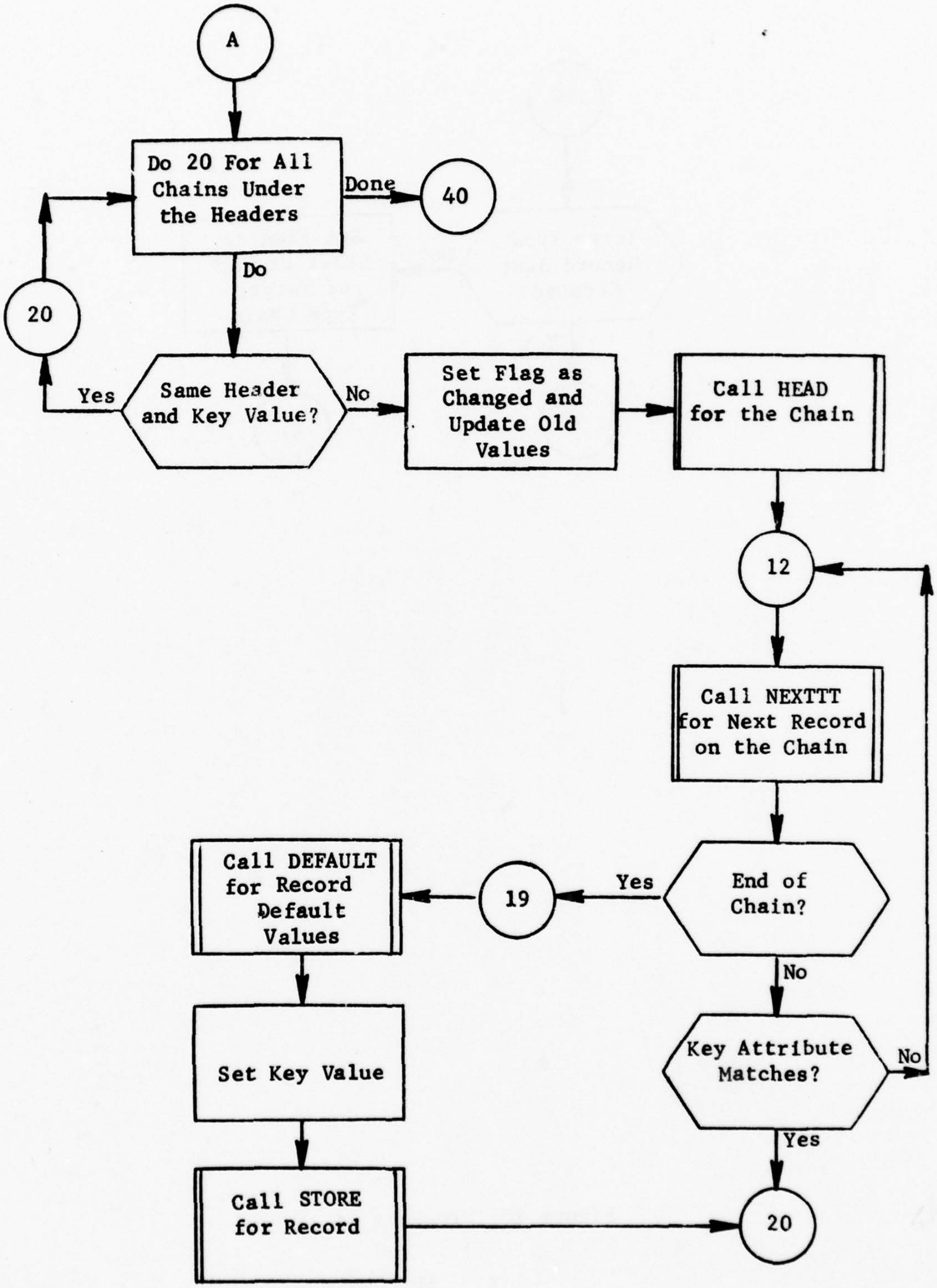

A
Do 20 For All Chains Under the Headers
Done
40
Do
20
Yes
Same Header and Key Value?
No
Set Flag as Changed and Update Old Values
Call HEAD for the Chain
12
Call NEXTTT for Next Record on the Chain
End of Chain?
Yes
19
Call DEFAULT for Record Default Values
No
Key Attribute Matches?
No
Set Key Value
Yes
Call STORE for Record
20

Figure 10. (Part 3 of 7)

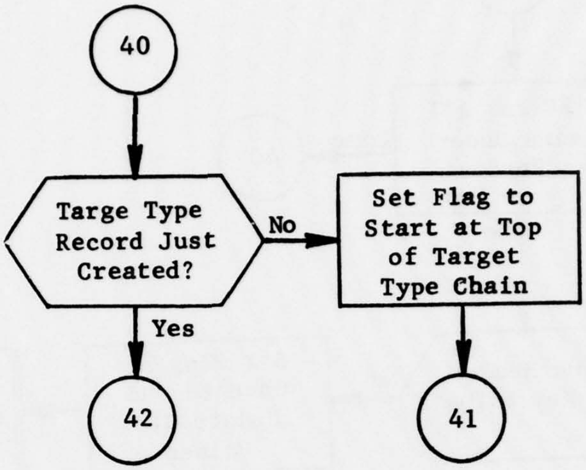

40
Targe Type Record Just Created?
No
Set Flag to Start at Top of Target Type Chain
Yes
42
41

Figure 10. (Part 4 of 7)

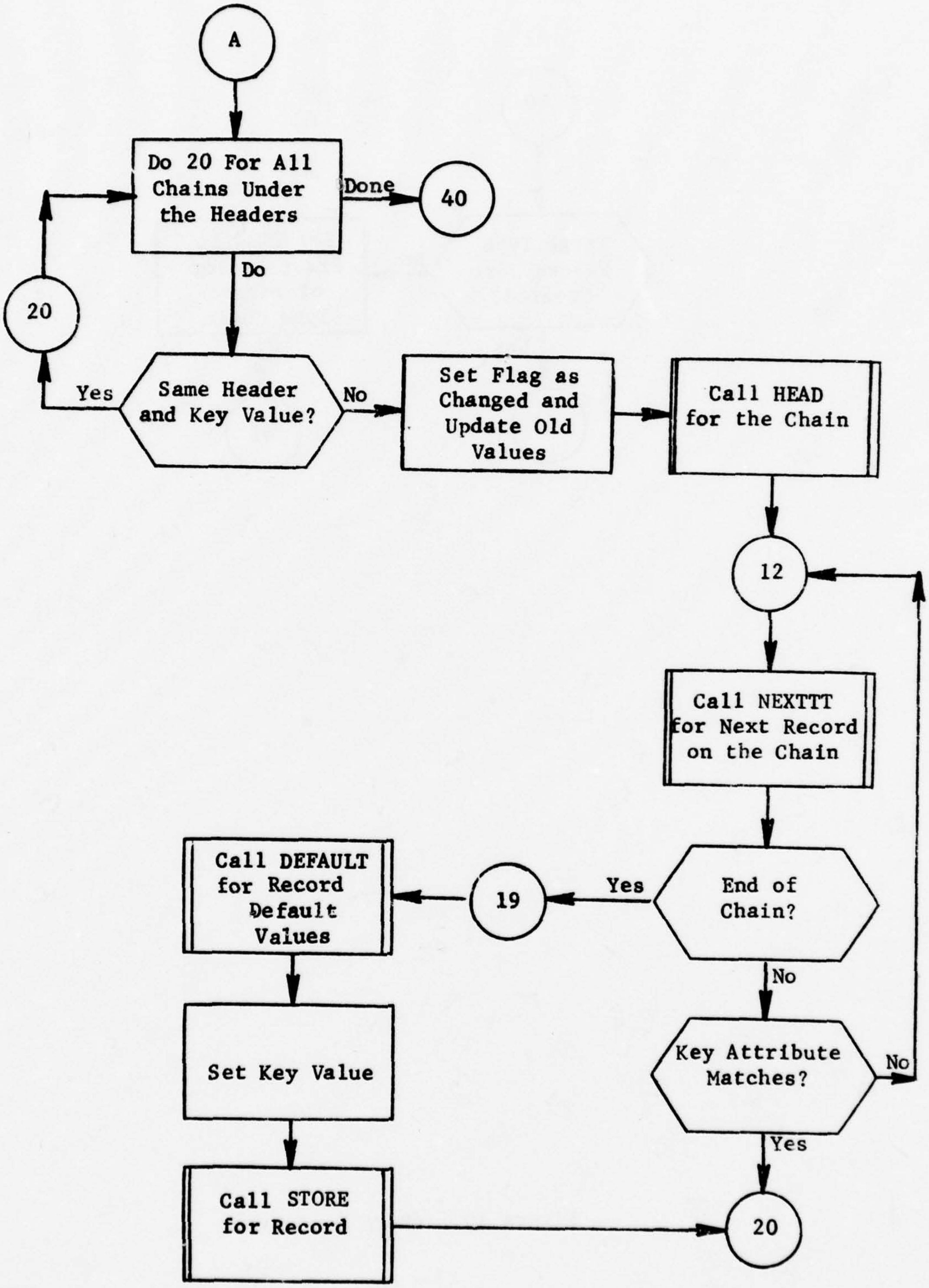

A
Do 20 For All Chains Under the Headers
Done
40
Do
20
Yes
Same Header and Key Value?
No
Set Flag as Changed and Update Old Values
Call HEAD for the Chain
12
Call NEXTTT for Next Record on the Chain
End of Chain?
Yes
19
Call DEFAULT for Record Default Values
No
Key Attribute Matches?
No
Yes
Set Key Value
Call STORE for Record
20

Figure 10. (Part 3 of 7)

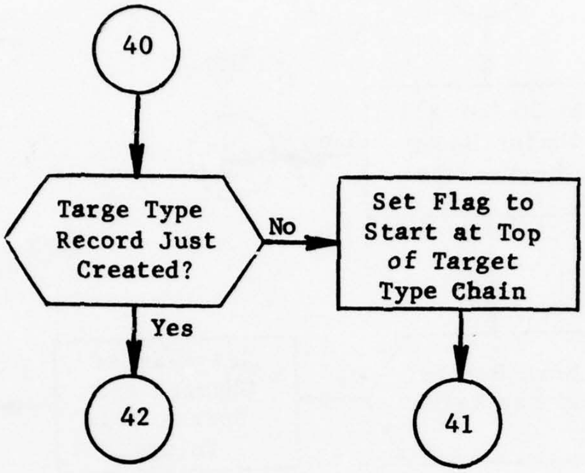

40
Targe Type Record Just Created?
No
Set Flag to Start at Top of Target Type Chain
Yes
42
41

Figure 10. (Part 4 of 7)

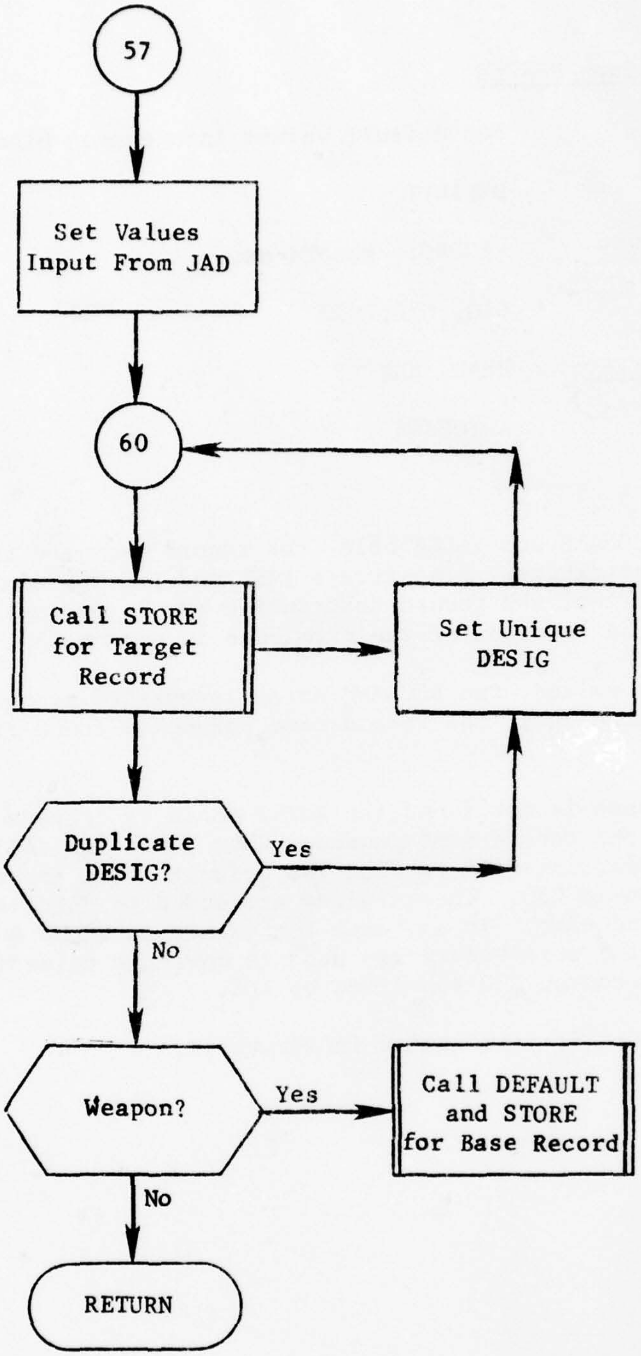

57
Set Values Input From JAD
60
Call STORE for Target Record
Set Unique DESIG
Duplicate DESIG?
Yes
No
Weapon?
Yes
Call DEFAULT and STORE for Base Record
No
RETURN

Figure 10. (Part 7 of 7)

2.9.2 Subroutine DEFAULT

PURPOSE:	Set default values into common block C30
ENTRY POINTS:	DEFAULT
FORMAL PARAMETERS:	RECORD: record name
COMMON BLOCKS:	C10, C20, C30
SUBROUTINES CALLED:	HEAD, NEXTTT
CALLED BY:	ADTOBASE

Method:

Local arrays RECNAME and VALUE hold the record name and the default value for the attribute. Also arrays LINK and LOC contain either a link to the start of the record description (0, if no more records are described) or the location of the attribute in common C30.

When DEFAULT is called, the RECNAME array is queried using the corresponding LINK value until the record name passed is found or the entire list queried.

If the record name is not found the RCTYP chain is traversed in an effort to find the record name passed. When found the IALINK and AILINK chains are traversed to find the default value for the attribute and its location in C30. These values are added to VALUE and LOC following the record name. In any case the values in VALUE and LOC between the record and the next record are used to move the value in VALUE into the location in common C30 specified by LOC.

Subroutine DEFAULT is illustrated in figure 11.

SECTION 3. DBMOD MODULE

3.1 Purpose

The purpose of DBMOD is to alter the content or characteristics of a data base in order to adapt the data base to the specific scenerio for which the plan is being developed. Because of its highly specialized nature, module DBMOD should be examined for possible revision each time a new plan is to be generated.

3.2 Input

User commands plus the integrated data base are necessary inputs to DBMOD. User inputs define the scenario, attacking and defending sides, plus optional inputs whereby nondefault scaling factors may be set.

All targets to be processed by the QUICK system must have been defined prior to DBMOD execution. This also includes a definition for each target's value (attributes VAL, IGIW or POP). For the attacking side attributes ADBLI, NADBLI, or ADBLR and NADBLR, and NPRSQ1, NPRSQ2, or NPRSQ3 must also have been defined.

3.3 Output

DBMOD generates printed reports and modifies the integrated data base for all U/I class targets for the defending side and for all missile and bomber class targets for the attacking side. U/I targets modify attribute VAL. Missile and bomber class targets modify attributes NOINCO, NALERT, NOPERSQ, ALRTDB and NLRTDB. If user requested, attributes TARDEFHI and TARDEFLO are modified.

3.4 Concept of Operation

DBMOD begins by reading input user commands and stores values that define the scenerio to be constructed, the attacking and defending sides, U/I class names, and the nondefault scaling factors used for U/I class value calculations. DBMOD, then determines the attributes for NOINCO (number in commission) and NALERT (number on alert) for bombers and missiles. The user also has the option of scaling the value (VAL) given to an U/I target based on the values for population (POP) and IGIW. The option also exists to calculate local bomber defenses (attributes TARDEFHI and TARDEFLO). For given collections of targets records, parameters are summed and properly printed.

3.5 Identification of Subroutine Function

3.5.1 Subroutine DESTAB. With the exception of utility routines, DESTAB is the only subroutine included under DBMOD. DESTAB keeps track of the number of target records within the data base, the number of target records deleted, and produces summary prints.

3.6 Common Block Definition

Common blocks used by DBMOD are outlined in table 2. Common blocks that communicate with the COP are given in appendix A of Maintenance Manual, Volume I.

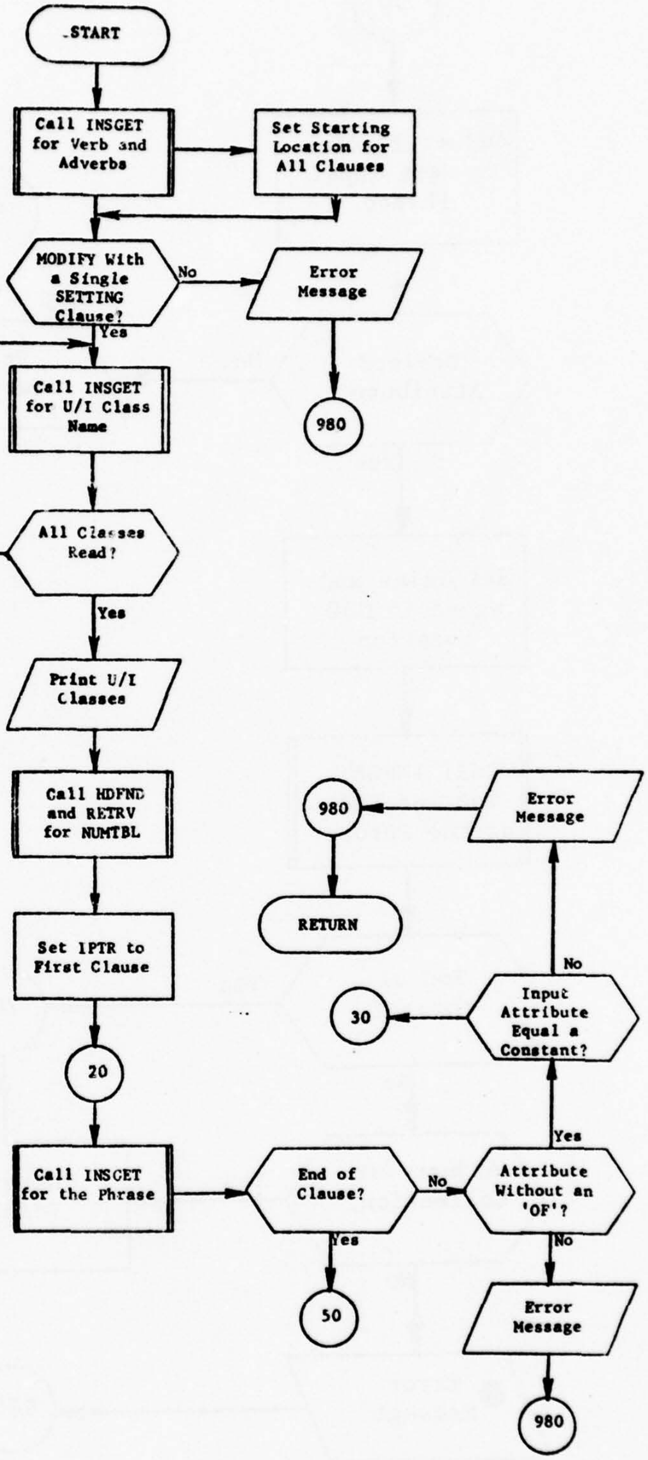

START
Call INSGET for Verb and Adverbs
Set Starting Location for All Clauses
MODIFY With a Single SETTING Clause?
No
Yes
Error Message
980
Call INSGET for U/I Class Name
All Classes Read?
No
Yes
Print U/I Classes
Call HDFND and RETRV for NUMTBL
Set IPTR to First Clause
20
Call INSGET for the Phrase
End of Clause?
Yes
50
No
Attribute Without an 'OF'?
Yes
No
Error Message
980
Input Attribute Equal a Constant?
30
No
Error Message
980
RETURN

Figure 15. DBMOD Module (Part 1 of 10)

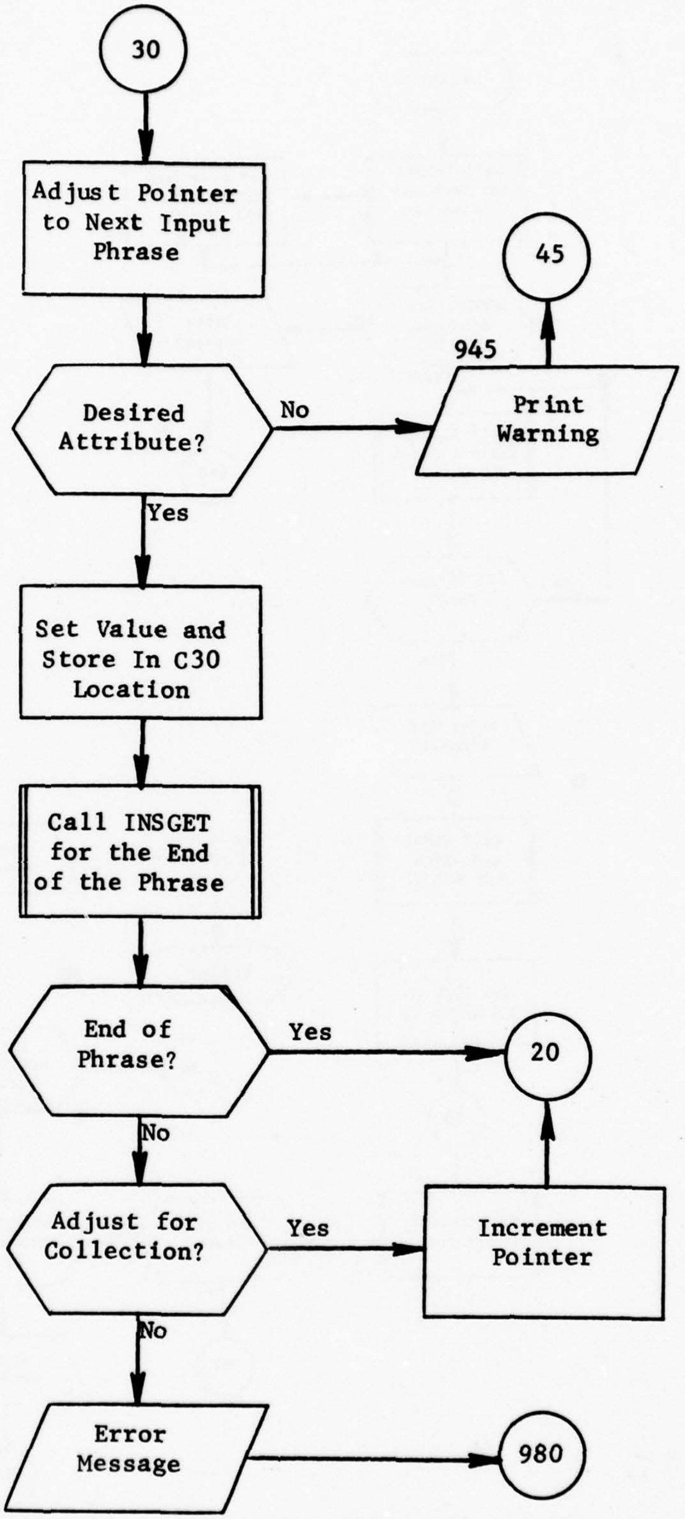


Figure 15. (Part 2 of 10)

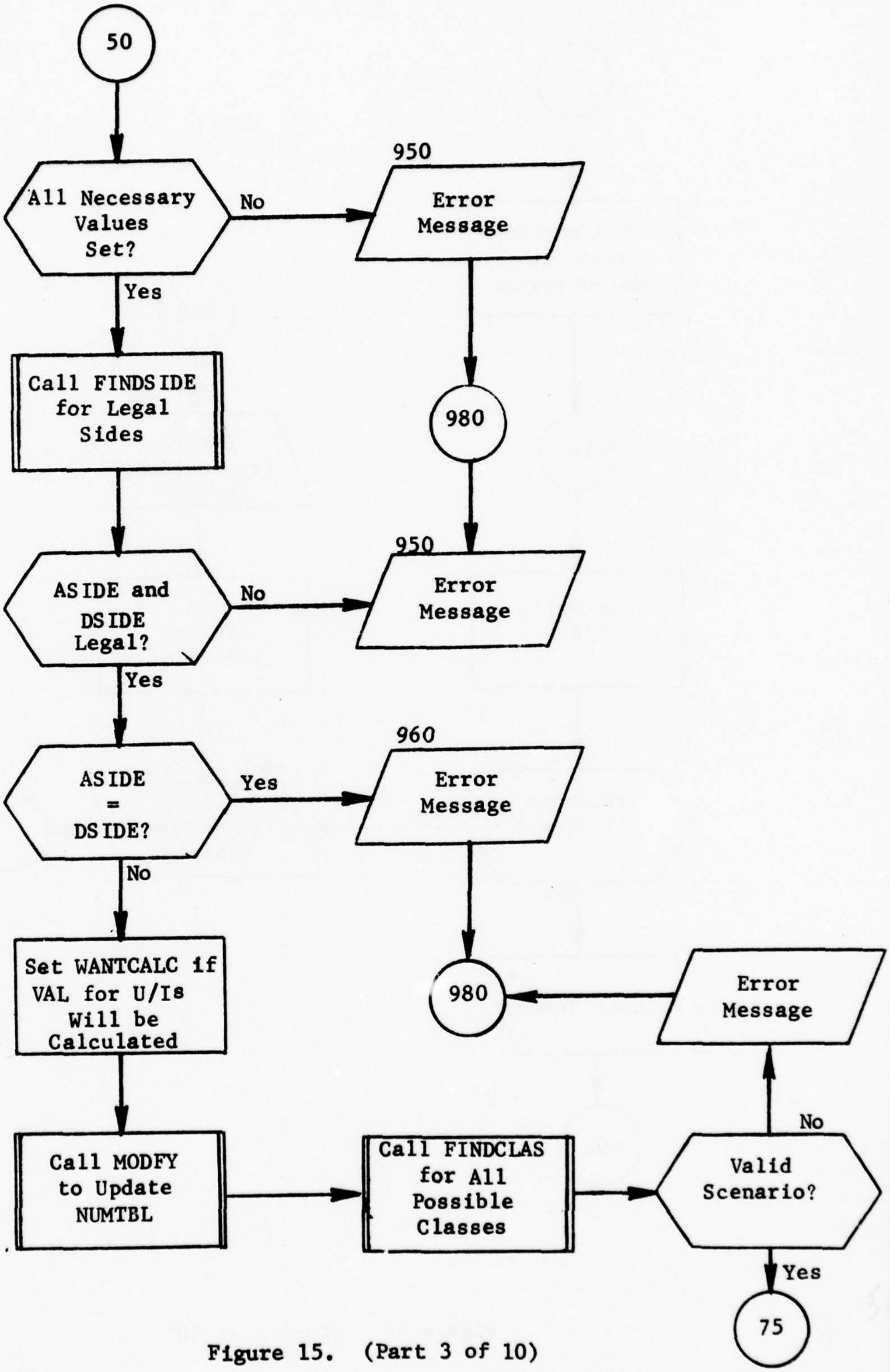

50
All Necessary Values Set?
No
950
Error Message
Yes
Call FINDSIDE for Legal Sides
980
950
ASIDE and DSIDE Legal?
No
Error Message
Yes
960
ASIDE = DSIDE?
Yes
Error Message
No
Set WANTCALC if VAL for U/Is Will be Calculated
980
Error Message
No
Call MODFY to Update NUMTBL
Call FINDCLAS for All Possible Classes
Valid Scenario?
Yes
75

Figure 15. (Part 3 of 10)

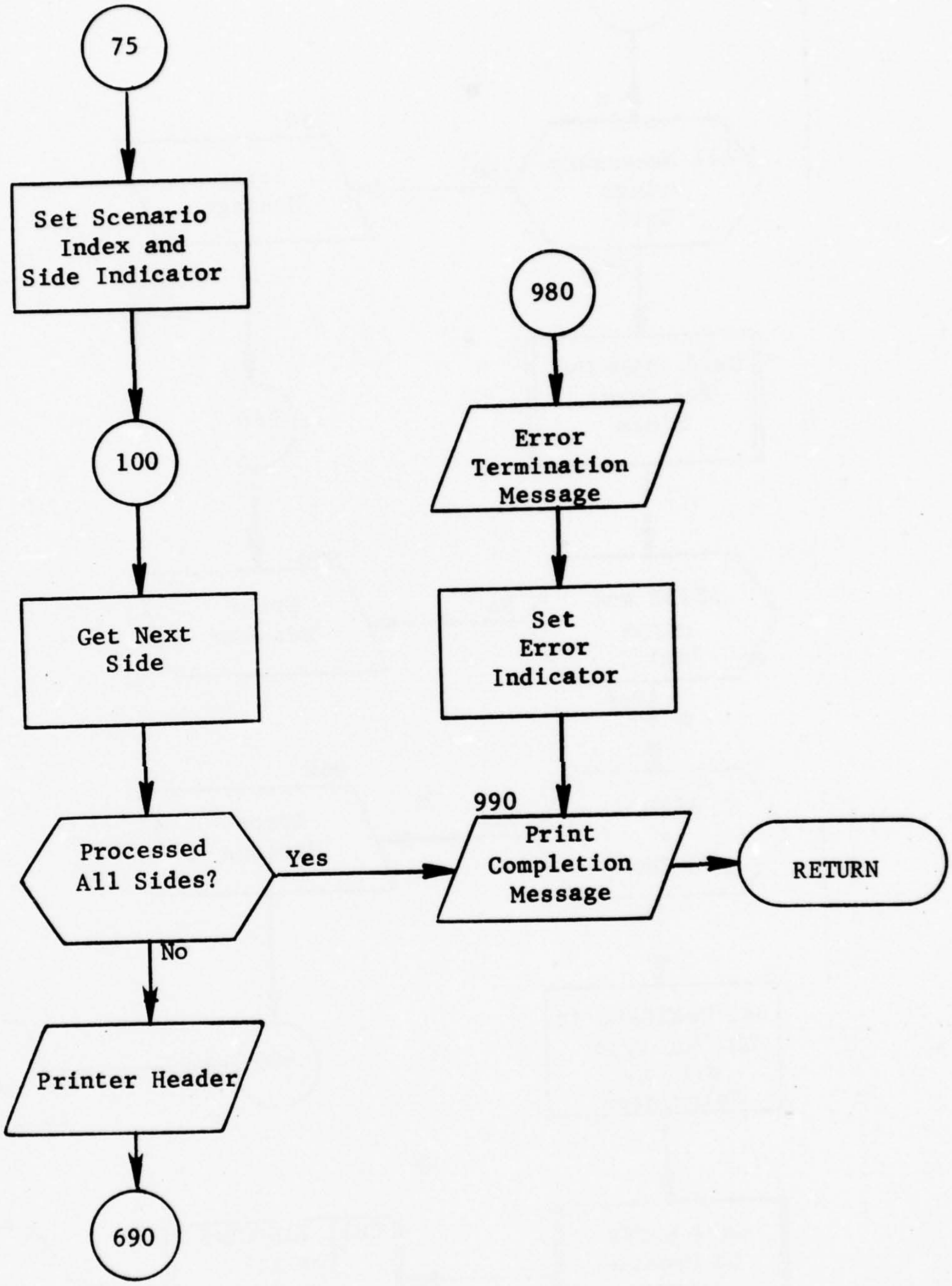


Figure 15. (Part 4 of 10)

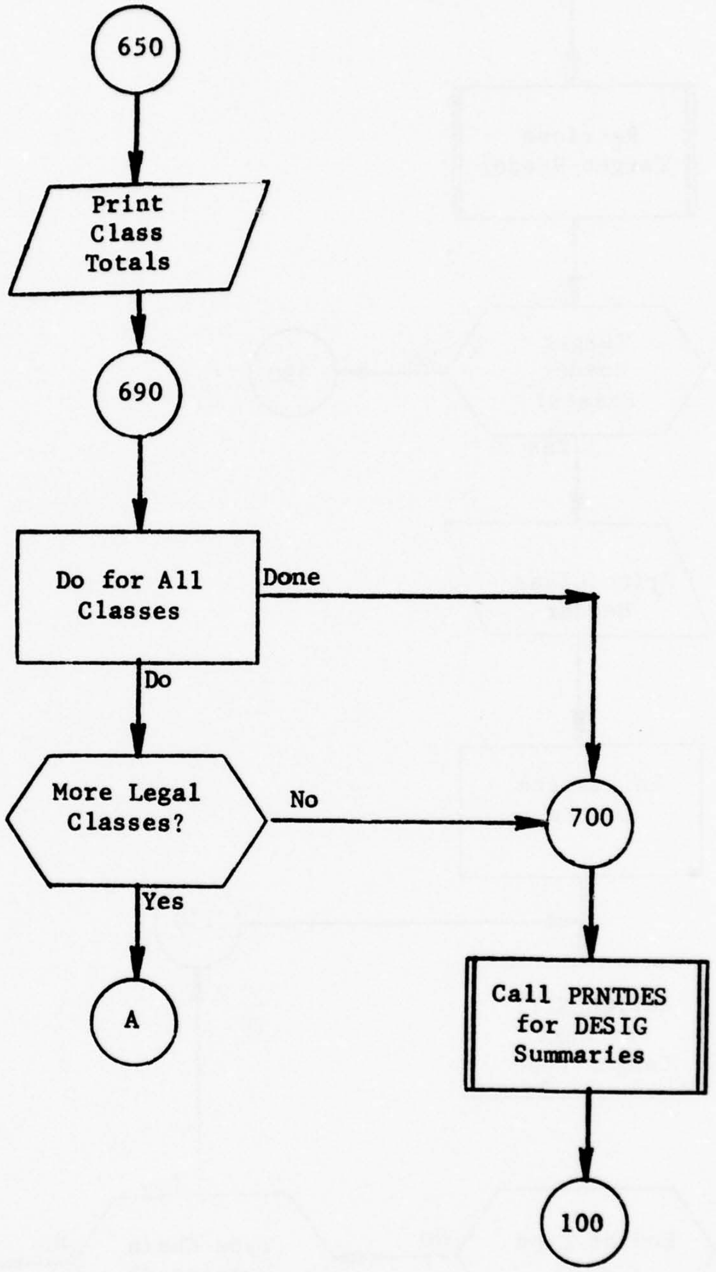


Figure 15. (Part 5 of 10)

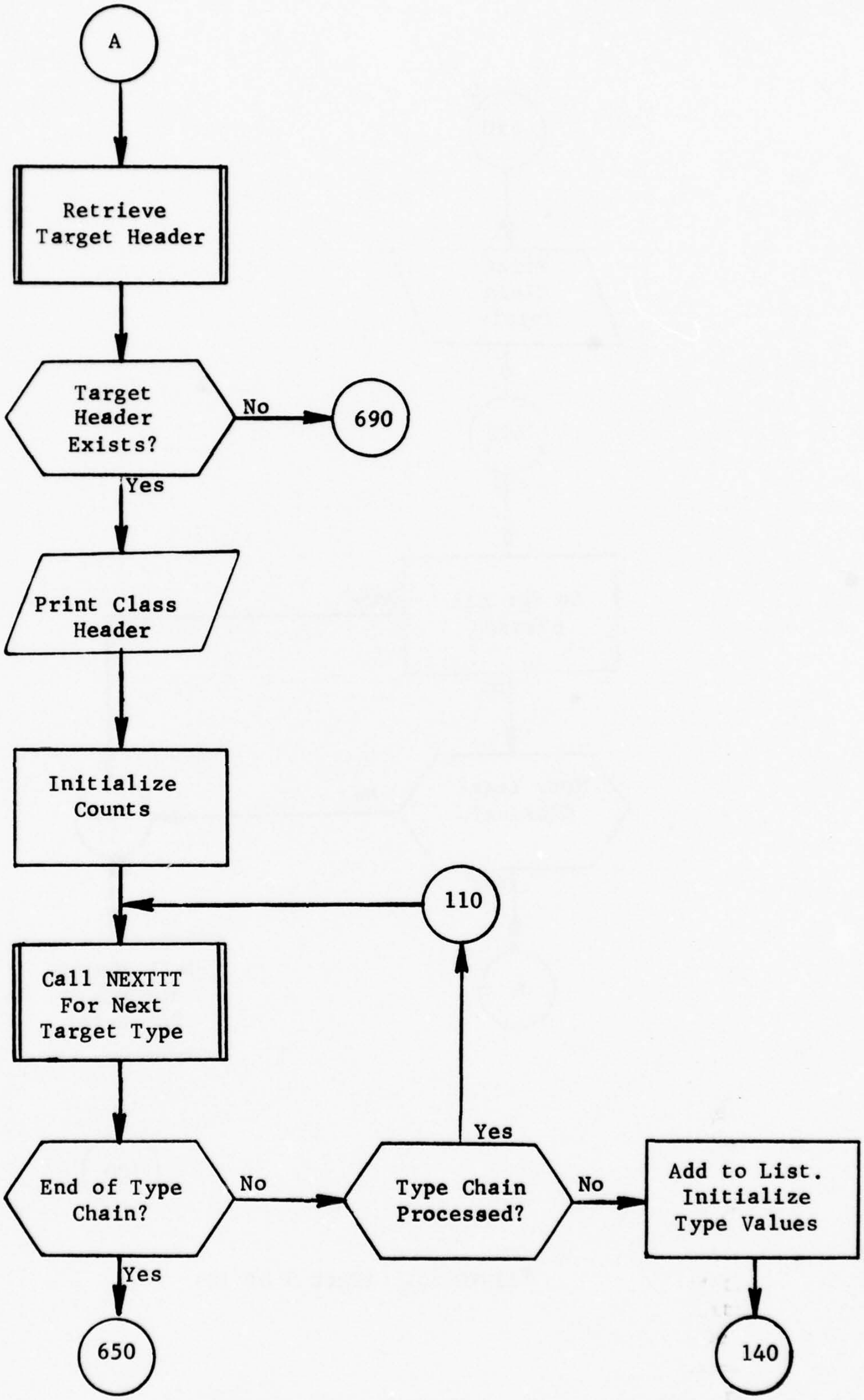


Figure 15. (Part 6 of 10)

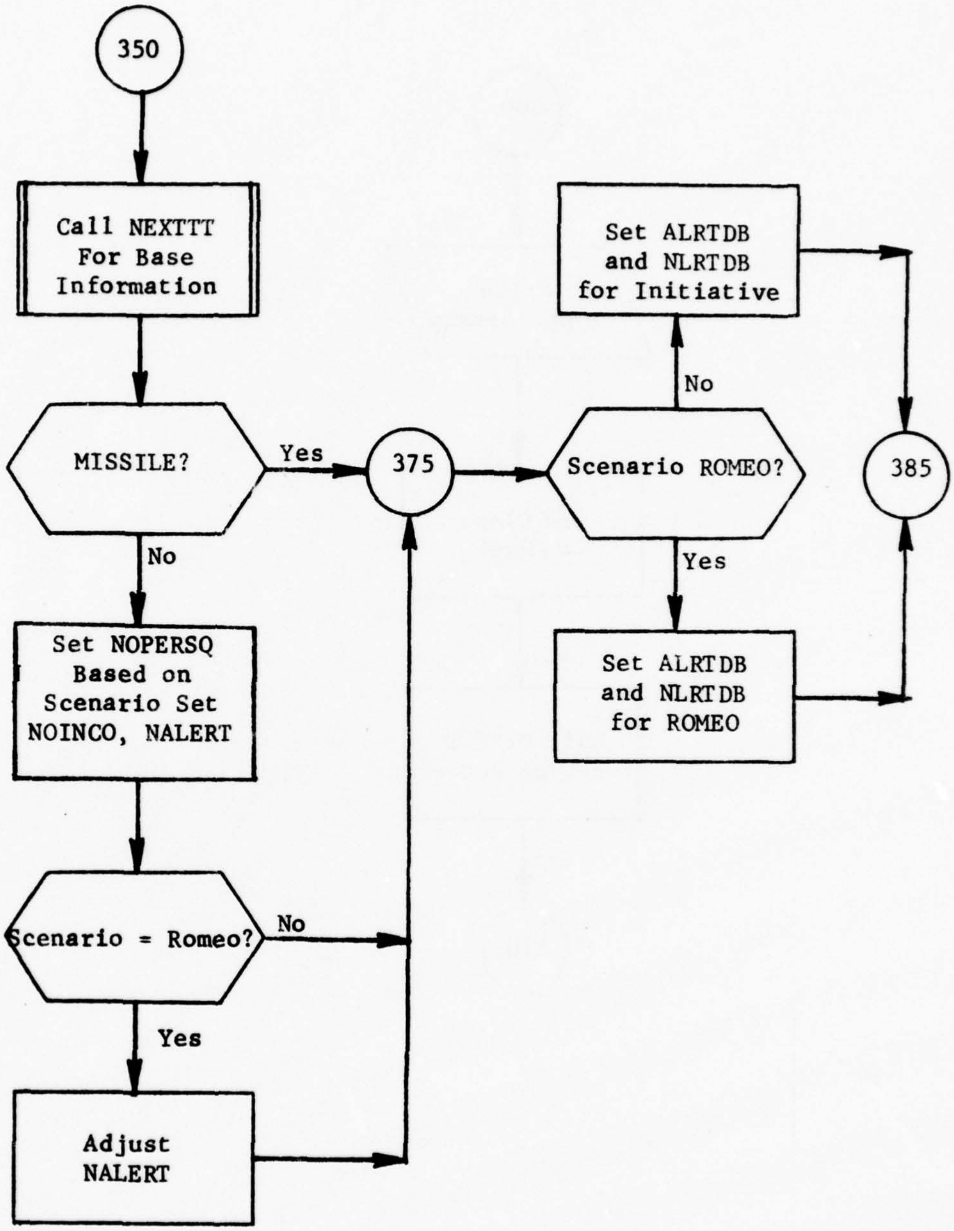

350
Call NEXTTT For Base Information
MISSILE?
Yes
No
375
Set NOPERSQ Based on Scenario Set NOINCO, NALERT
Scenario = Romeo?
No
Yes
Adjust NALERT
Scenario ROMEO?
No
Yes
Set ALRTDB and NLRTDB for Initiative
Set ALRTDB and NLRTDB for ROMEO
385

Figure 15. (Part 9 of 10)

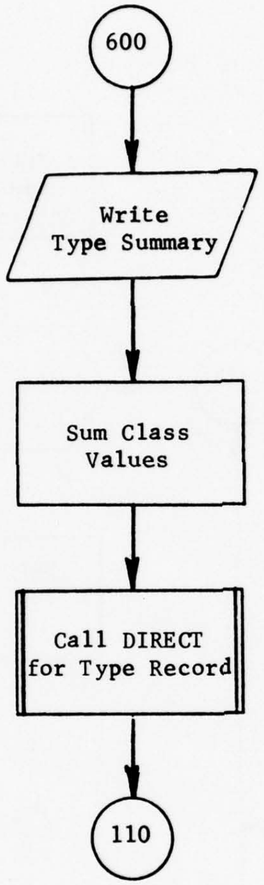


Figure 15. (Part 10 of 10)

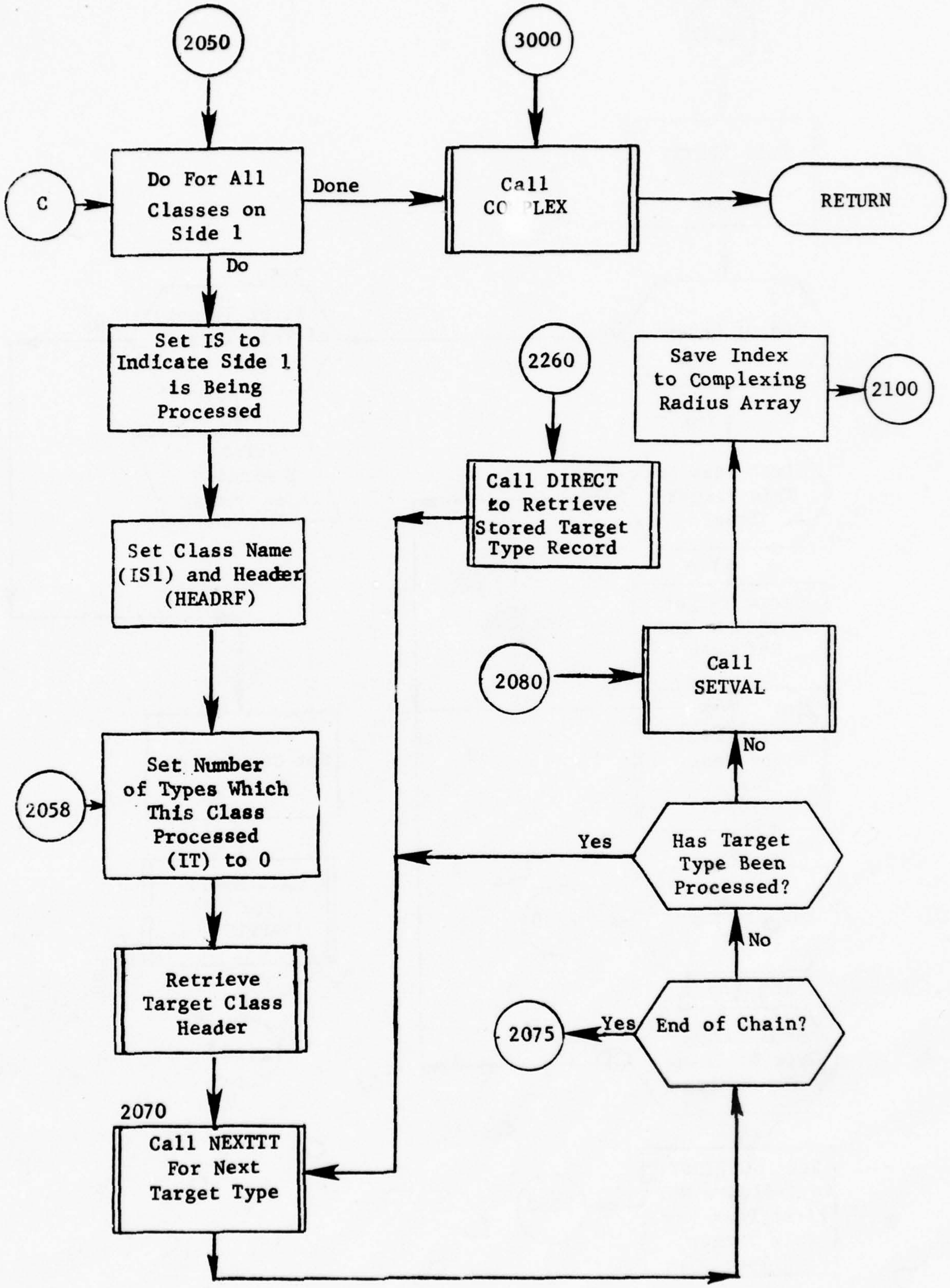

2050
3000
C
Do For All Classes on Side 1
Done
Call COMPLEX
RETURN
Do
Set IS to Indicate Side 1 is Being Processed
2260
Save Index to Complexing Radius Array
2100
Call DIRECT to Retrieve Stored Target Type Record
Set Class Name (IS1) and Header (HEADRF)
2080
Call SETVAL
No
2058
Set Number of Types Which This Class Processed (IT) to 0
Yes
Has Target Type Been Processed?
No
Retrieve Target Class Header
2075
Yes
End of Chain?
2070
Call NEXTTT For Next Target Type

Figure 17. (Part 5 of 7)

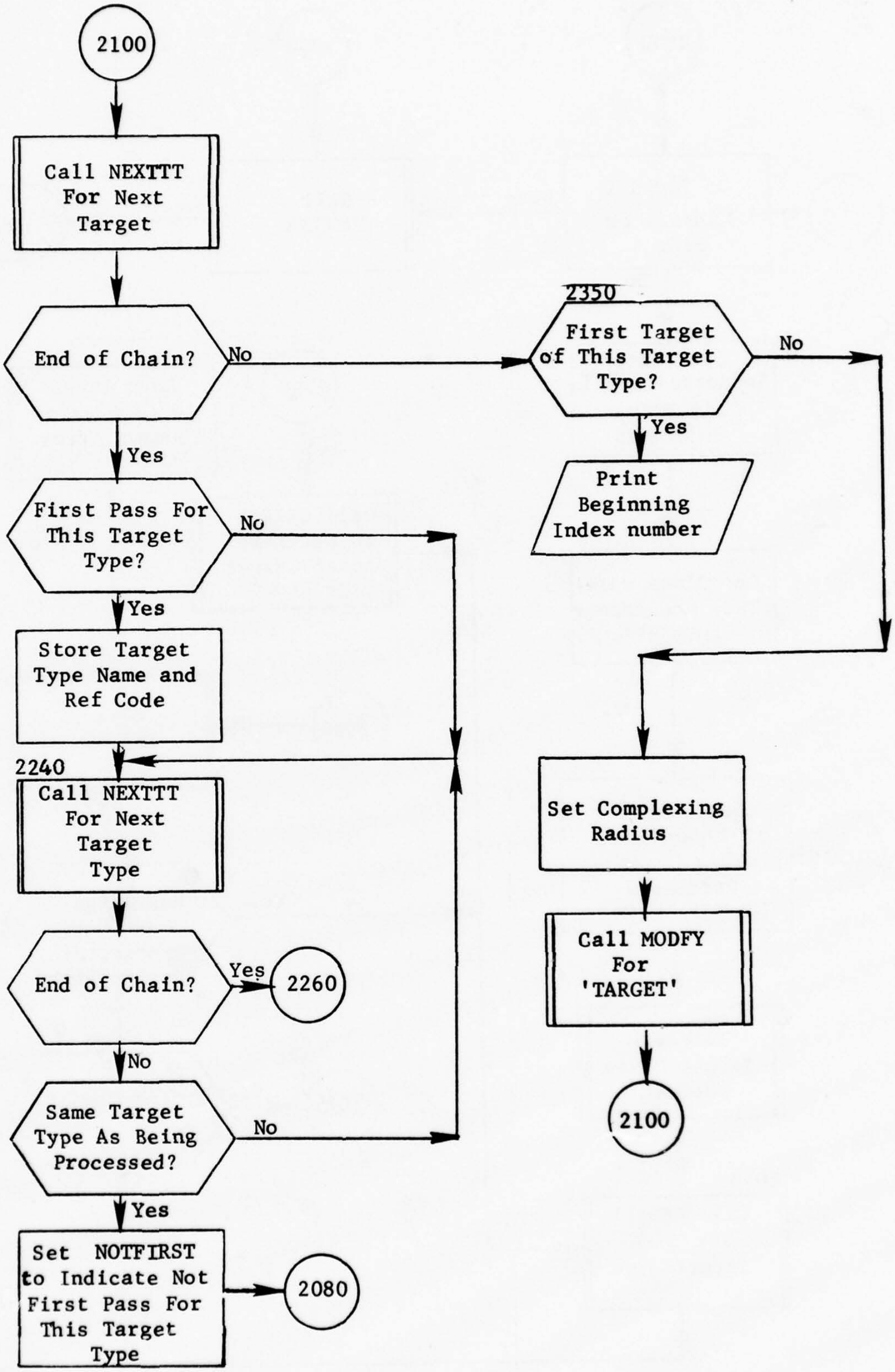

2100
Call NEXTTT For Next Target
End of Chain?
No
Yes
2350
First Target of This Target Type?
No
Yes
Print Beginning Index number
First Pass For This Target Type?
No
Yes
Store Target Type Name and Ref Code
2240
Call NEXTTT For Next Target Type
Set Complexing Radius
Call MODFY For 'TARGET'
End of Chain?
Yes
2260
No
2100
Same Target Type As Being Processed?
No
Yes
Set NOTFIRST to Indicate Not First Pass For This Target Type
2080

Figure 17. (Part 6 of 7)

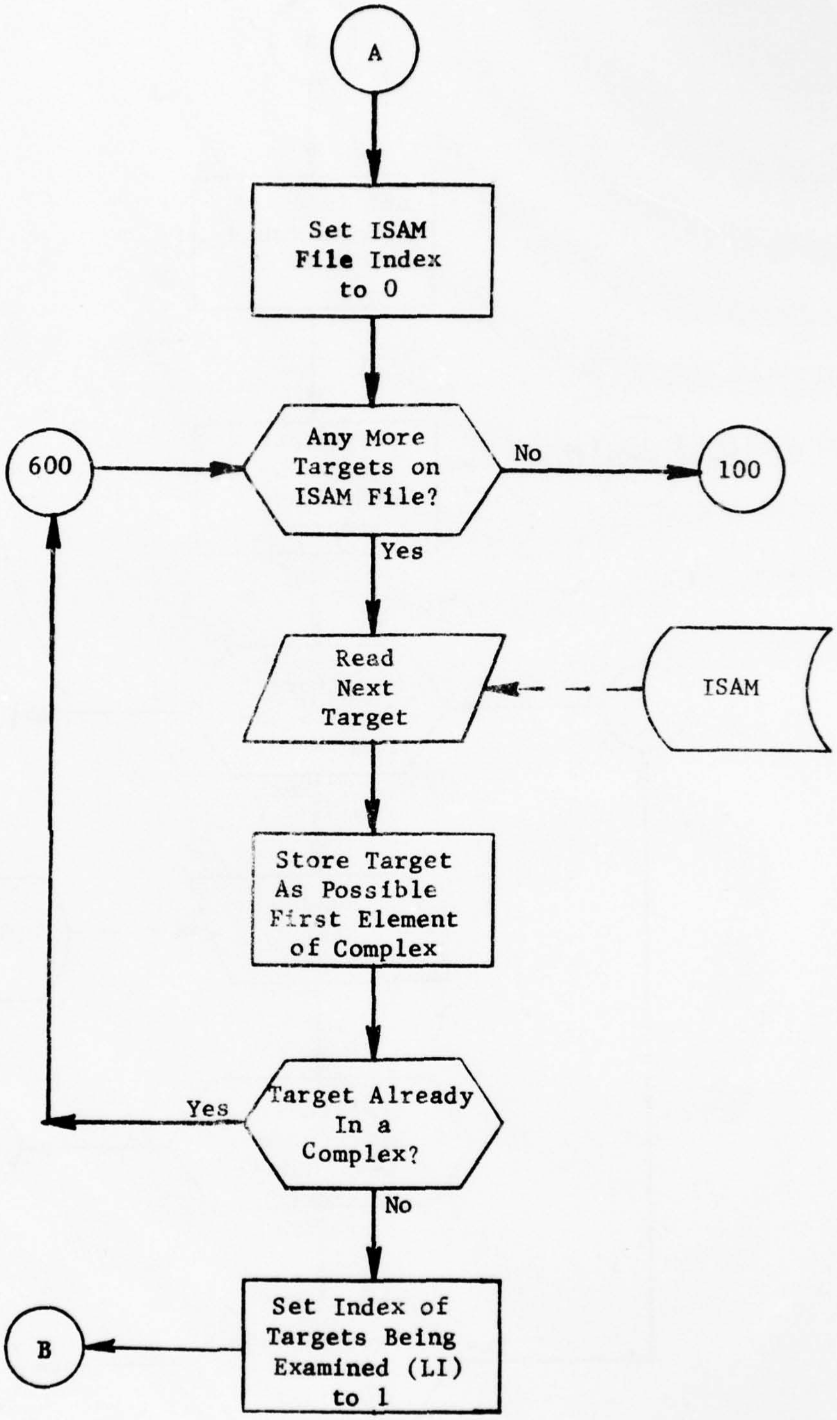

A
Set ISAM File Index to 0
Any More Targets on ISAM File?
No
100
600
Yes
Read Next Target
ISAM
Store Target As Possible First Element of Complex
Target Already In a Complex?
Yes
No
Set Index of Targets Being Examined (LI) to 1
B

Figure 18. (Part 2 of 6)

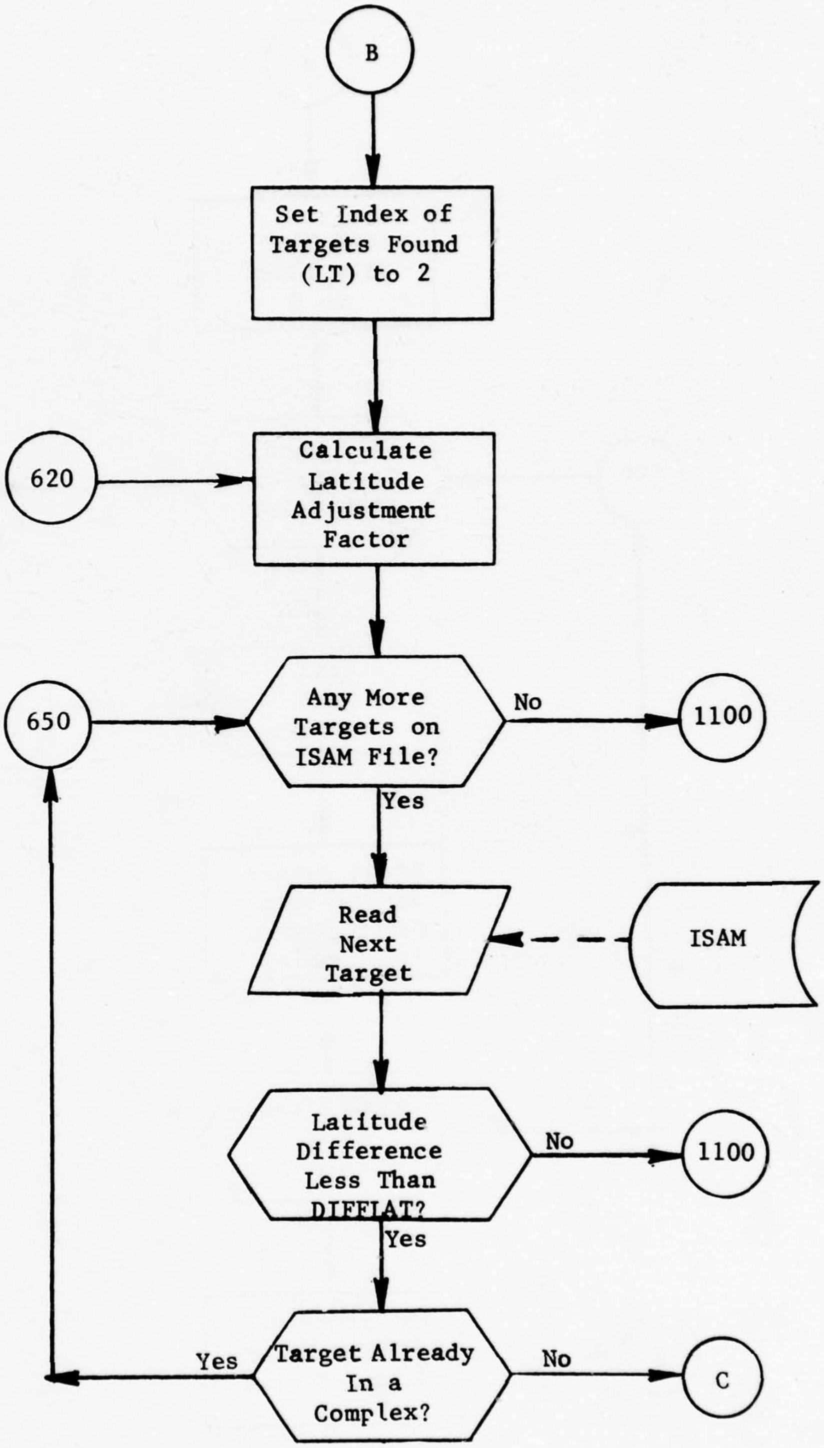

B
Set Index of Targets Found (LT) to 2
620
Calculate Latitude Adjustment Factor
650
Any More Targets on ISAM File?
No
1100
Yes
Read Next Target
ISAM
Latitude Difference Less Than DIFFLAT?
No
1100
Yes
Yes
Target Already In a Complex?
No
C

Figure 18. (Part 3 of 6)

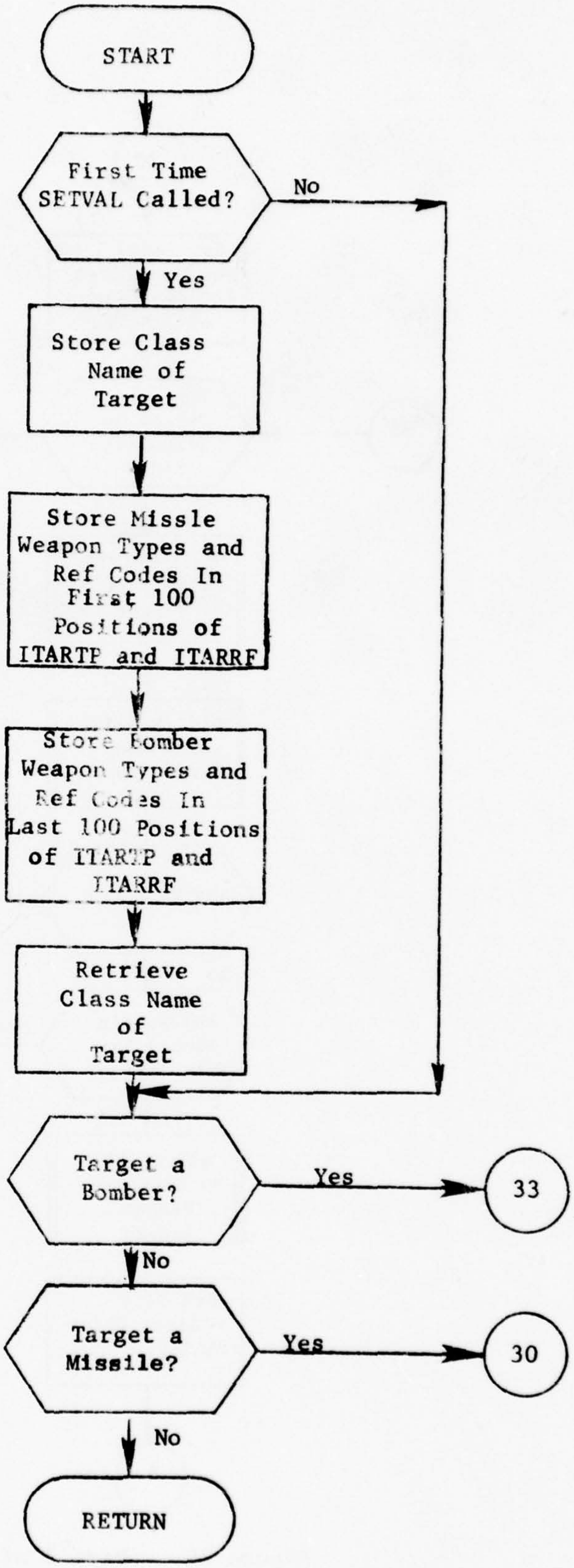

START
First Time SETVAL Called?
No
Yes
Store Class Name of Target
Store Missle Weapon Types and Ref Codes In First 100 Positions of ITARTP and ITARRF
Store Bomber Weapon Types and Ref Codes In Last 100 Positions of ITARTP and ITARRF
Retrieve Class Name of Target
Target a Bomber?
Yes
33
No
Target a Missile?
Yes
30
No
RETURN

Figure 20. Subroutine SETVAL (Part 1 of 3)

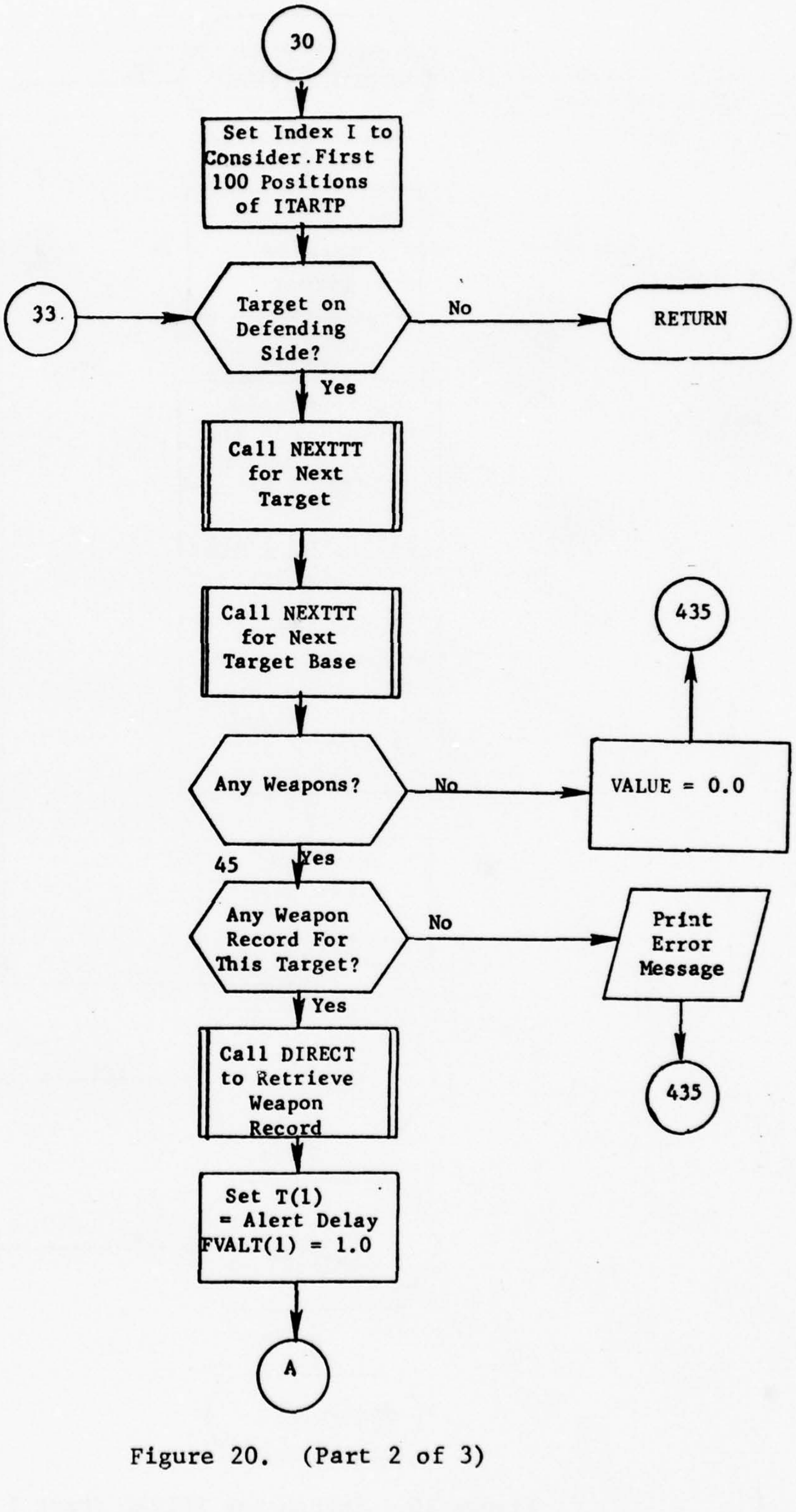

30
Set Index I to Consider First 100 Positions of ITARTP
33
Target on Defending Side?
No
RETURN
Yes
Call NEXTTT for Next Target
Call NEXTTT for Next Target Base
435
Any Weapons?
No
VALUE = 0.0
45
Yes
Any Weapon Record For This Target?
No
Print Error Message
Yes
Call DIRECT to Retrieve Weapon Record
435
Set T(1) = Alert Delay FVALT(1) = 1.0
A

Figure 20. (Part 2 of 3)

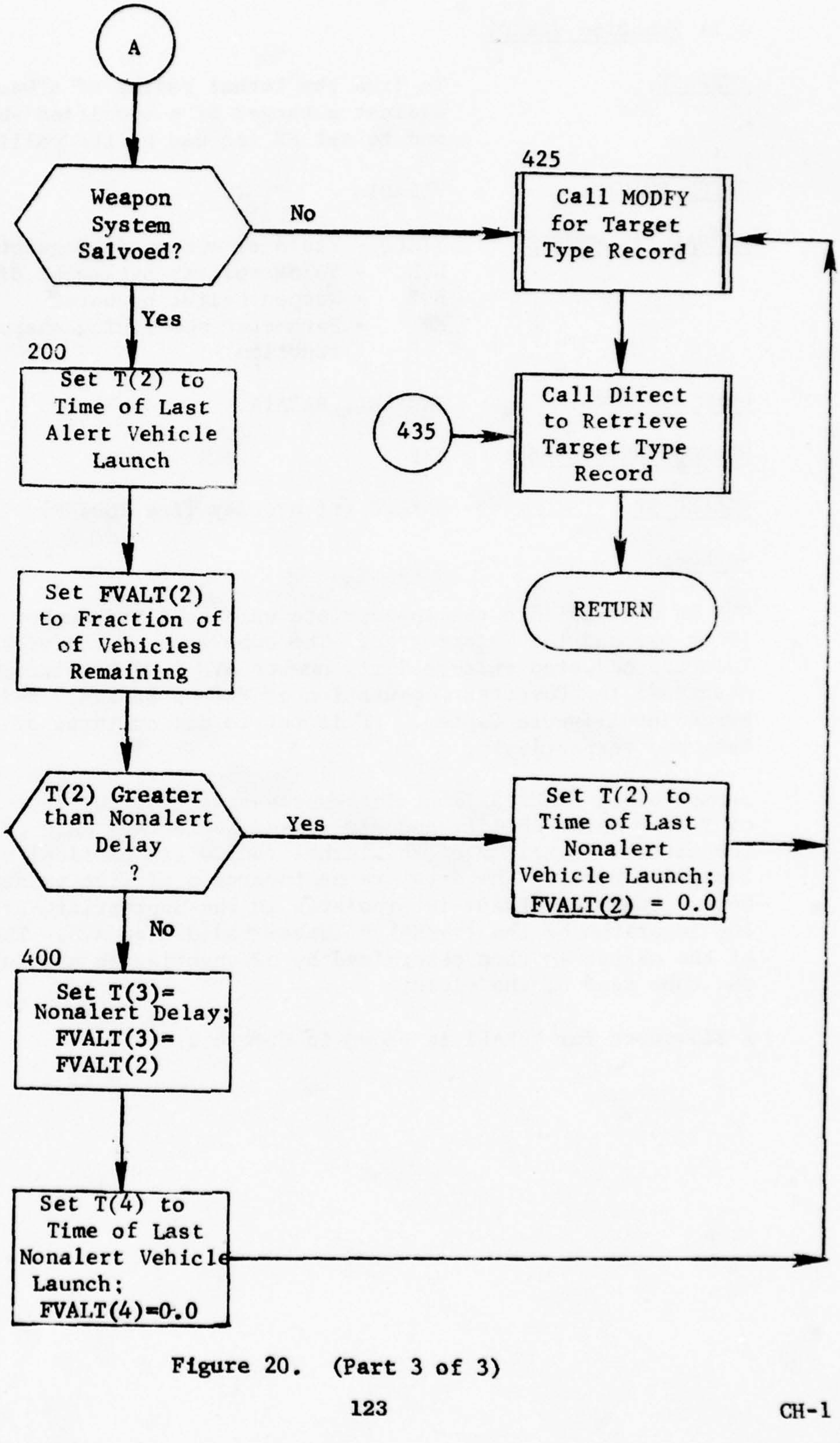

A
Weapon System Salvoed?
No
Yes
425
Call MODFY for Target Type Record
200
Set T(2) to Time of Last Alert Vehicle Launch
435
Call Direct to Retrieve Target Type Record
RETURN
Set FVALT(2) to Fraction of of Vehicles Remaining
T(2) Greater than Nonalert Delay ?
Yes
Set T(2) to Time of Last Nonalert Vehicle Launch; FVALT(2) = 0.0
No
400
Set T(3)= Nonalert Delay; FVALT(3)= FVALT(2)
Set T(4) to Time of Last Nonalert Vehicle Launch; FVALT(4)=0.0

Figure 20. (Part 3 of 3)

4.11 Function VLRADI

PURPOSE:	To find the lethal radius of a weapon delivered against a target of a specified vulnerability, and to set FN for use by the calling subroutine.
ENTRY POINTS:	VLRADI
FORMAL PARAMETERS:	YIELD - Yield of weapon in megatons NVN - Vulnerability parameter of target HOB - Weapon height of burst FN - Parameter specifying shape of damage function
COMMON BLOCKS:	INDXRHL, RADATA
SUBROUTINES CALLED:	EXP
CALLED BY:	ENTMOD (of overlay link INDXER)

Method:

NVN is decoded into the appropriate vulnerability number VN, the letter (P or Q), and the K-factor XK. The cube root of the yield is extracted. Then the adjusted vulnerability number AVN is determined by methods described in "Computer Computation of Weapon Radius," B-139-61, Air Force Intelligence Center. FN is set to six or three of P and Q type targets, respectively.

Common block /RADATA/ contains four arrays (for the four combinations of P or Q vulnerability and air-to-surface burst) each of which contains the natural logarithm of the lethal radius (in nautical miles) of a 1-megaton burst. The data are at intervals of five vulnerability numbers. Function VLRADI interpolates in the appropriate array to find the logarithm of the 1-megaton lethal radius for AVN. The lethal radius of the weapon is then determined by exponentiating and multiplying by the cube root of the yield.

A flowchart for VLRADI is shown in figure 21.

During the second pass the value of each target is normalized by multiplying its data base value by the scaling factor for its target class. Also a randomized target number is calculated for each complex target, multiple target, and simple target not in a complex or multiple target. A TARCDE record is stored for each of these targets. This record contains the target number of the target as well as the reference code of the representative target if the target is complex or multiple or if the target itself is a simple target.

Upon processing each target record, the target number can be immediately calculated and stored. The calculation is: a sorting index (LEAD), which is a function of the total number of targets (NTAR), is determined by the formula:

$$\mathrm{LEAD} = \frac{\mathrm{NTAR}\ (3 - \sqrt{5}}{2}$$

To start a cycle, a beginning index (IBEG) is designated and assigned to the target being read. Initially IBEG=LEAD. The index for the next target (IND) then is found by incrementing the previous index (LAST) by LEAD. If the result exceeds NTAR, the cycle is reset by subtracting NTAR from IND. When a cycle is completed (i.e., when IND=IBEG), the next cycle is begun by incrementing IBEG by one and proceeding as above. Thus, a unique nonsequential index is assigned to each target as it is read.

Also, during the record pass through the targets, elements of complexes not belonging to target classes chosen to be considered by the allocator are transferred from the complex chains they are on to the simple target chain. When all such targets are transfered if a complex target has just one element remaining, this element is also transfered to the simple target chain to be considered as a single target. For the remaining complex targets, subroutine CALCOMP is called.

After the second pass complex records to complex targets which do not have any elements belonging to target classes to be considered by the allocator are deleted. Also SORTIT is called and the FLAG-DESIG Listing and Target Designator-Number Directory are printed.

Subroutine SRTTGT is illustrated in figure 27.

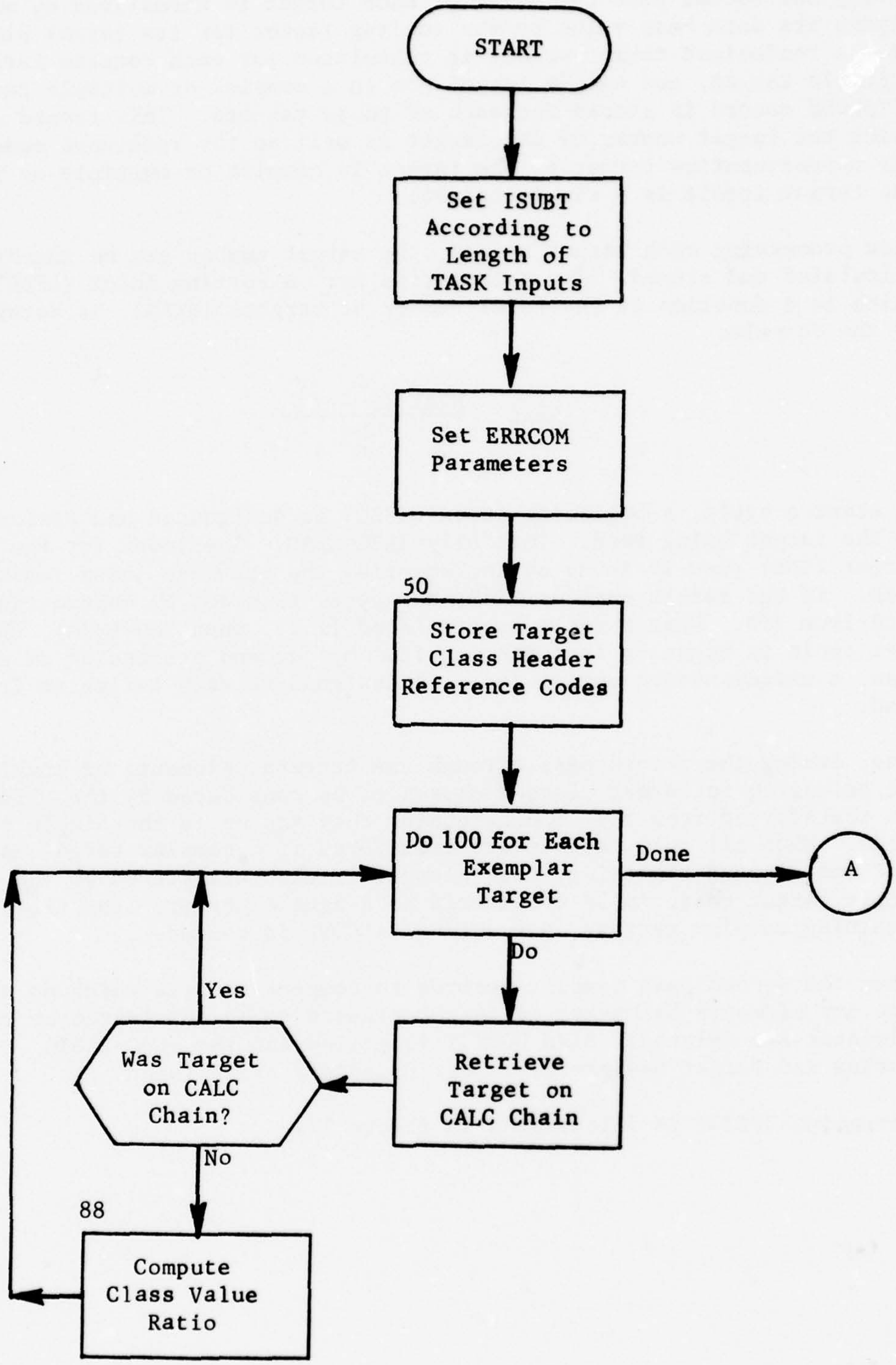

START
Set ISUBT According to Length of TASK Inputs
Set ERRCOM Parameters
50
Store Target Class Header Reference Codes
Do 100 for Each Exemplar Target
Done
A
Do
Retrieve Target on CALC Chain
Was Target on CALC Chain?
Yes
No
88
Compute Class Value Ratio

Figure 27. Subroutine SRTTGT (Part 1 of 13)

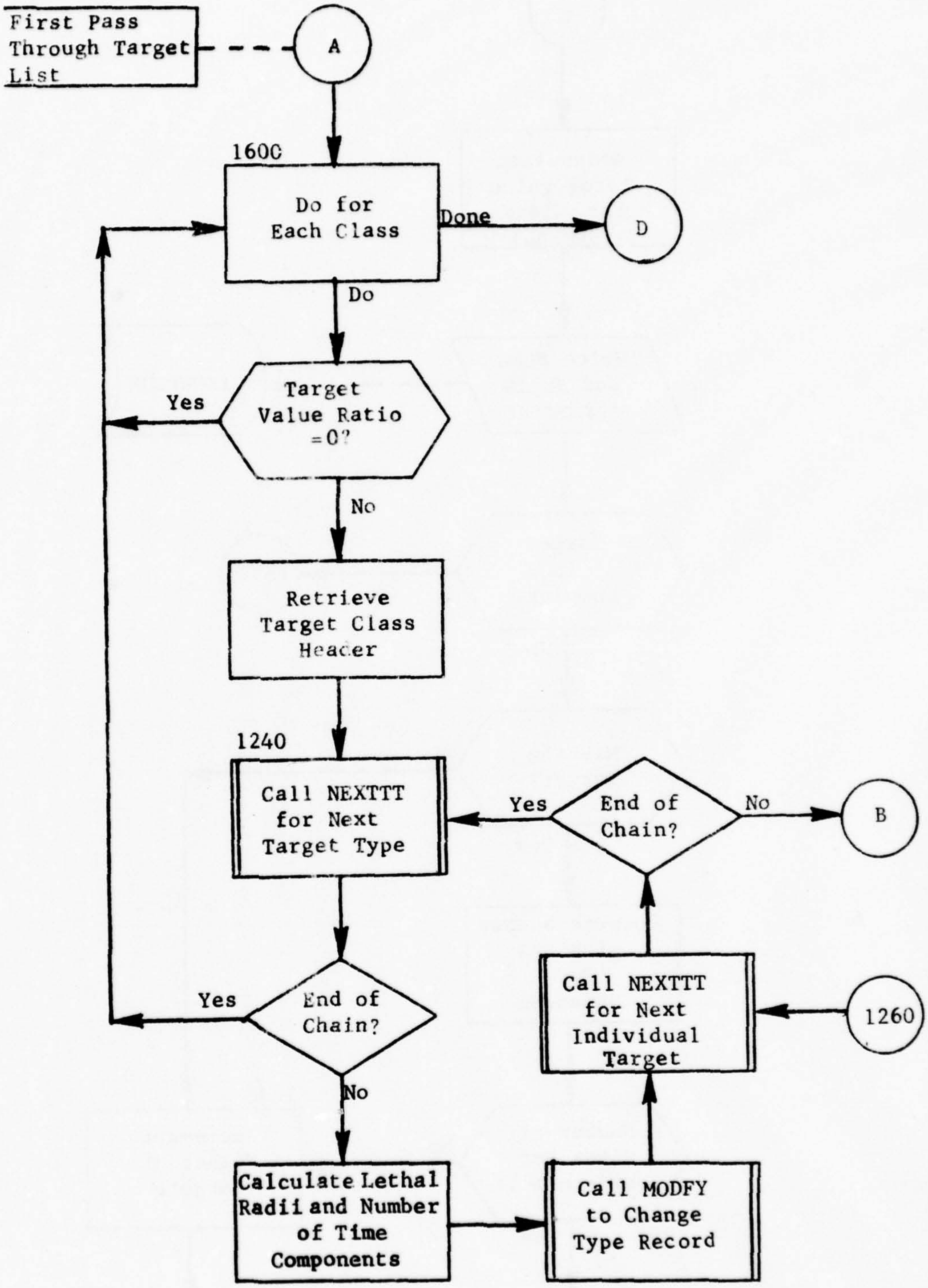

First Pass Through Target List
A
1600
Do for Each Class
Done
D
Do
Target Value Ratio =0?
Yes
No
Retrieve Target Class Header
1240
Call NEXTTT for Next Target Type
Yes
End of Chain?
No
B
End of Chain?
Yes
No
Call NEXTTT for Next Individual Target
1260
Calculate Lethal Radii and Number of Time Components
Call MODFY to Change Type Record

Figure 27. (Part 2 of 13)

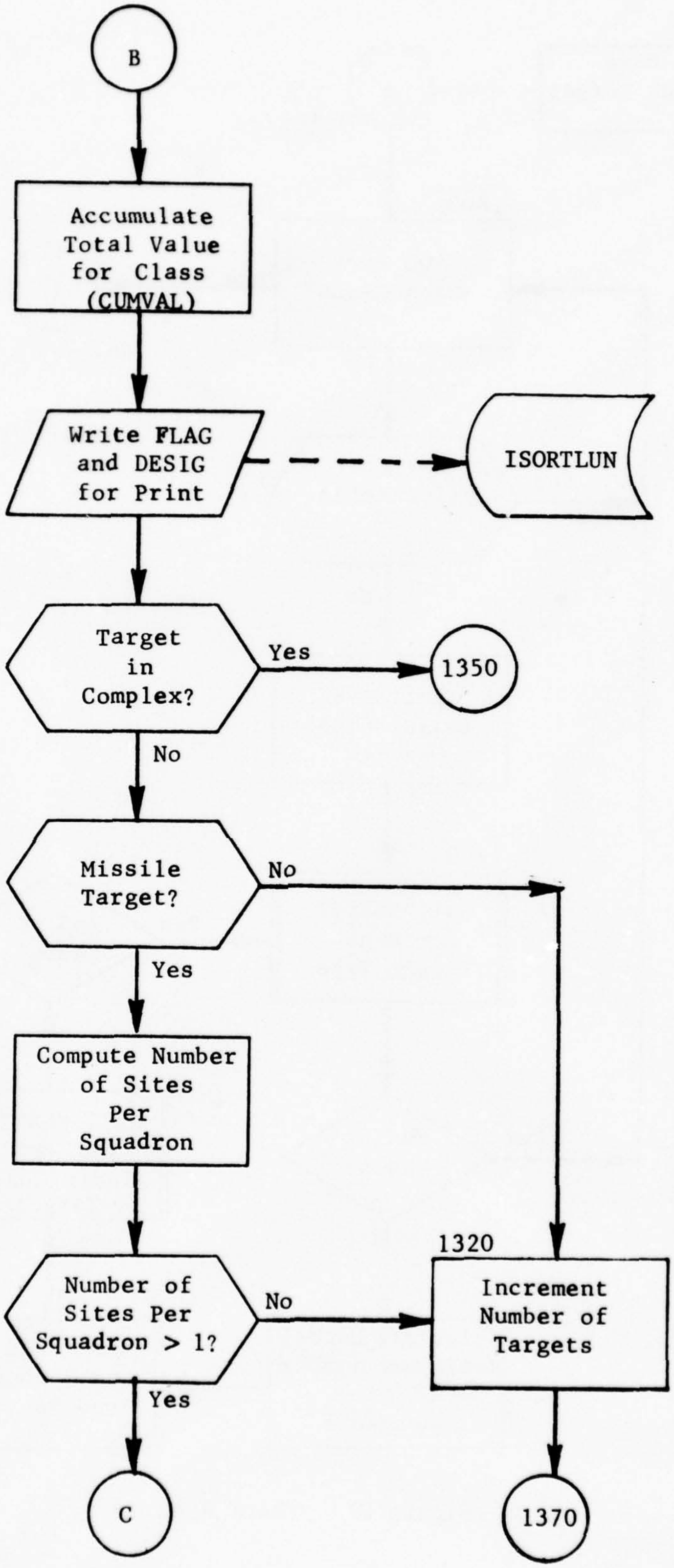

B
Accumulate Total Value for Class (CUMVAL)
Write FLAG and DESIG for Print
ISORTLUN
Target in Complex?
Yes
1350
No
Missile Target?
No
Yes
Compute Number of Sites Per Squadron
1320
Number of Sites Per Squadron > 1?
No
Increment Number of Targets
Yes
C
1370

Figure 27. (Part 3 of 13)

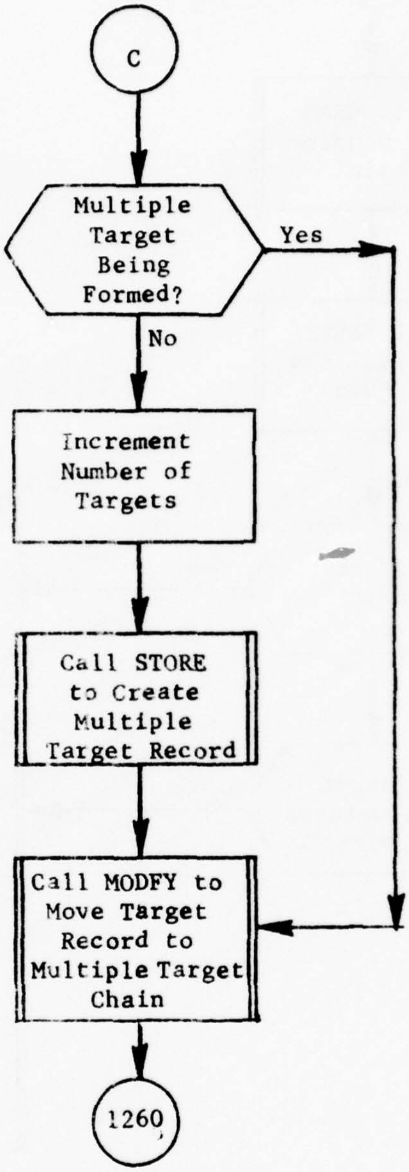

C
Multiple Target Being Formed?
Yes
No
Increment Number of Targets
Call STORE to Create Multiple Target Record
Call MODFY to Move Target Record to Multiple Target Chain
1260

Figure 27. (Part 4 of 13)

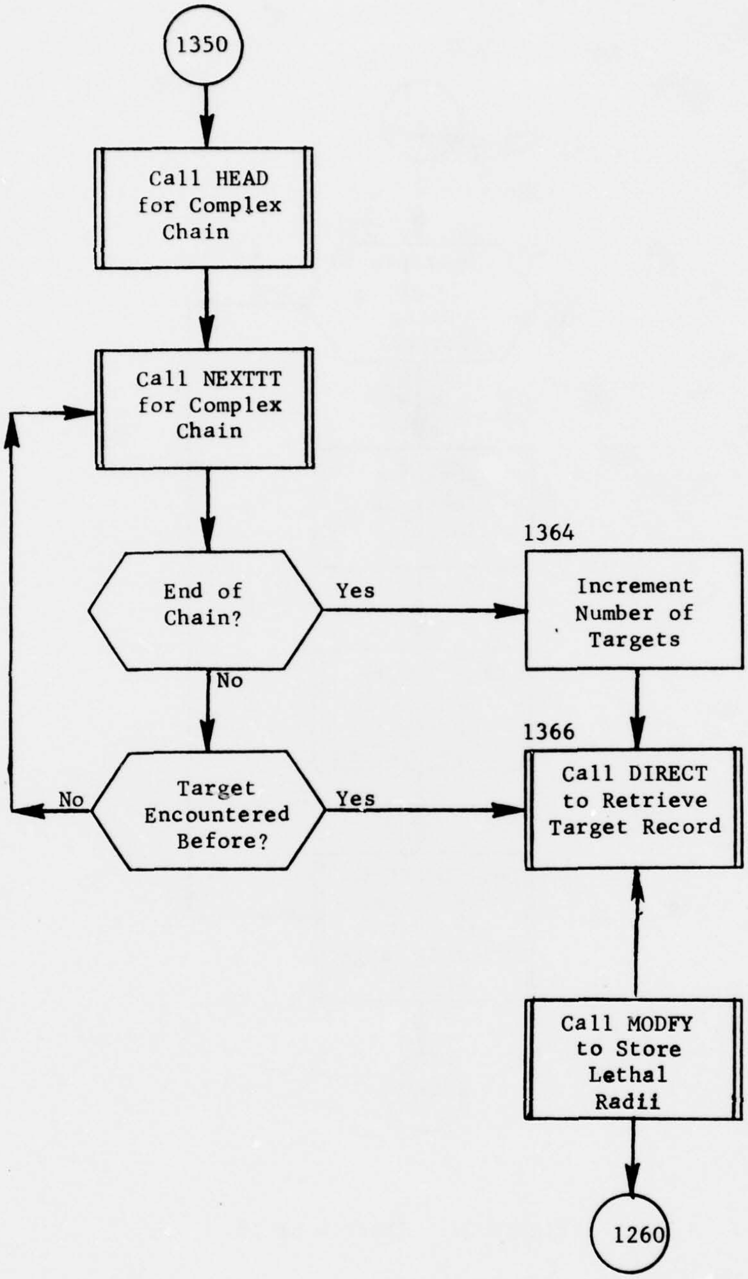

1350
Call HEAD for Complex Chain
Call NEXTTT for Complex Chain
End of Chain?
Yes
No
1364
Increment Number of Targets
Target Encountered Before?
No
Yes
1366
Call DIRECT to Retrieve Target Record
Call MODFY to Store Lethal Radii
1260

Figure 27. (Part 5 of 13)

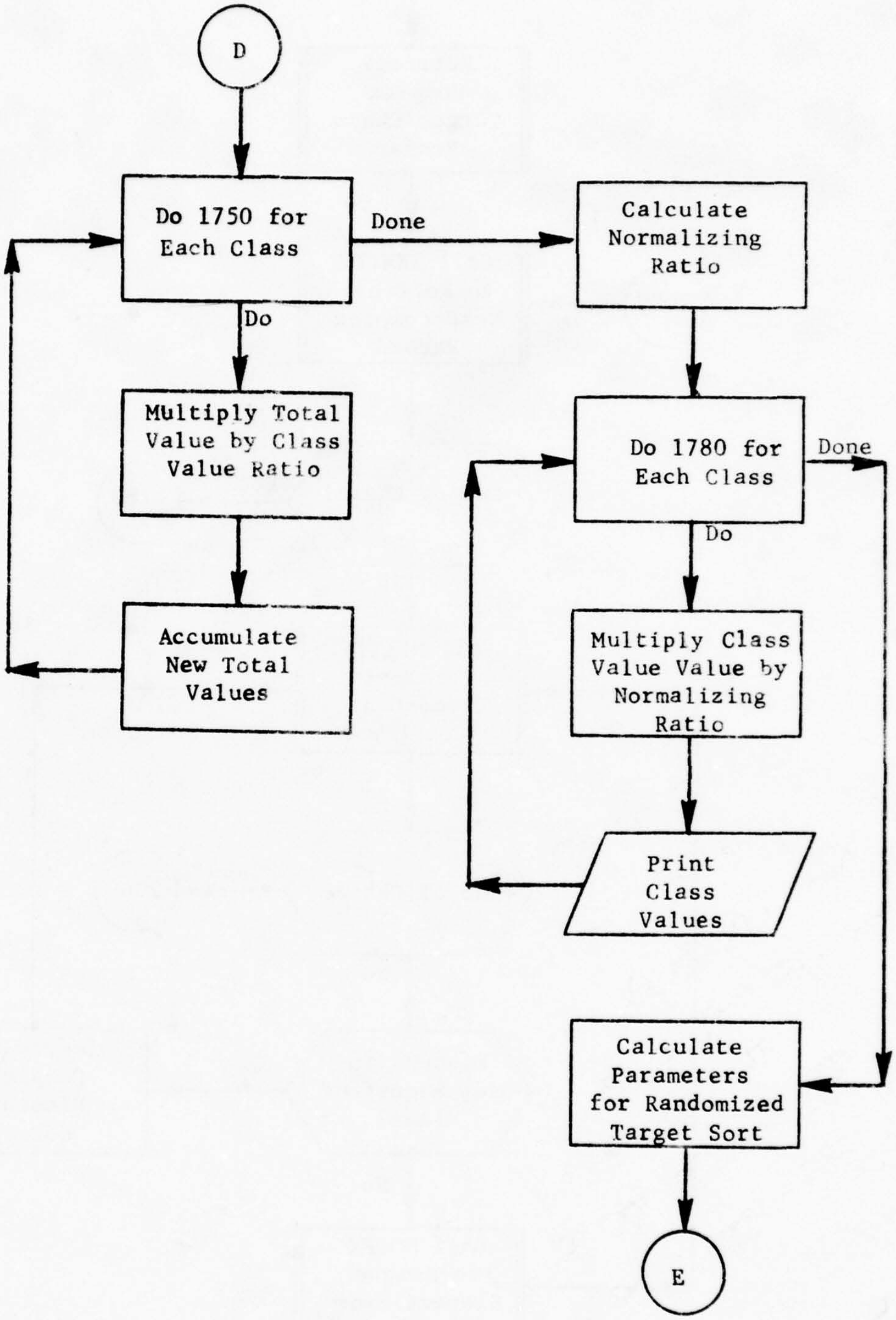

D
Do 1750 for Each Class
Done
Do
Multiply Total Value by Class Value Ratio
Accumulate New Total Values
Calculate Normalizing Ratio
Do 1780 for Each Class
Done
Do
Multiply Class Value Value by Normalizing Ratio
Print Class Values
Calculate Parameters for Randomized Target Sort
E

Figure 27. (Part 6 of 13)

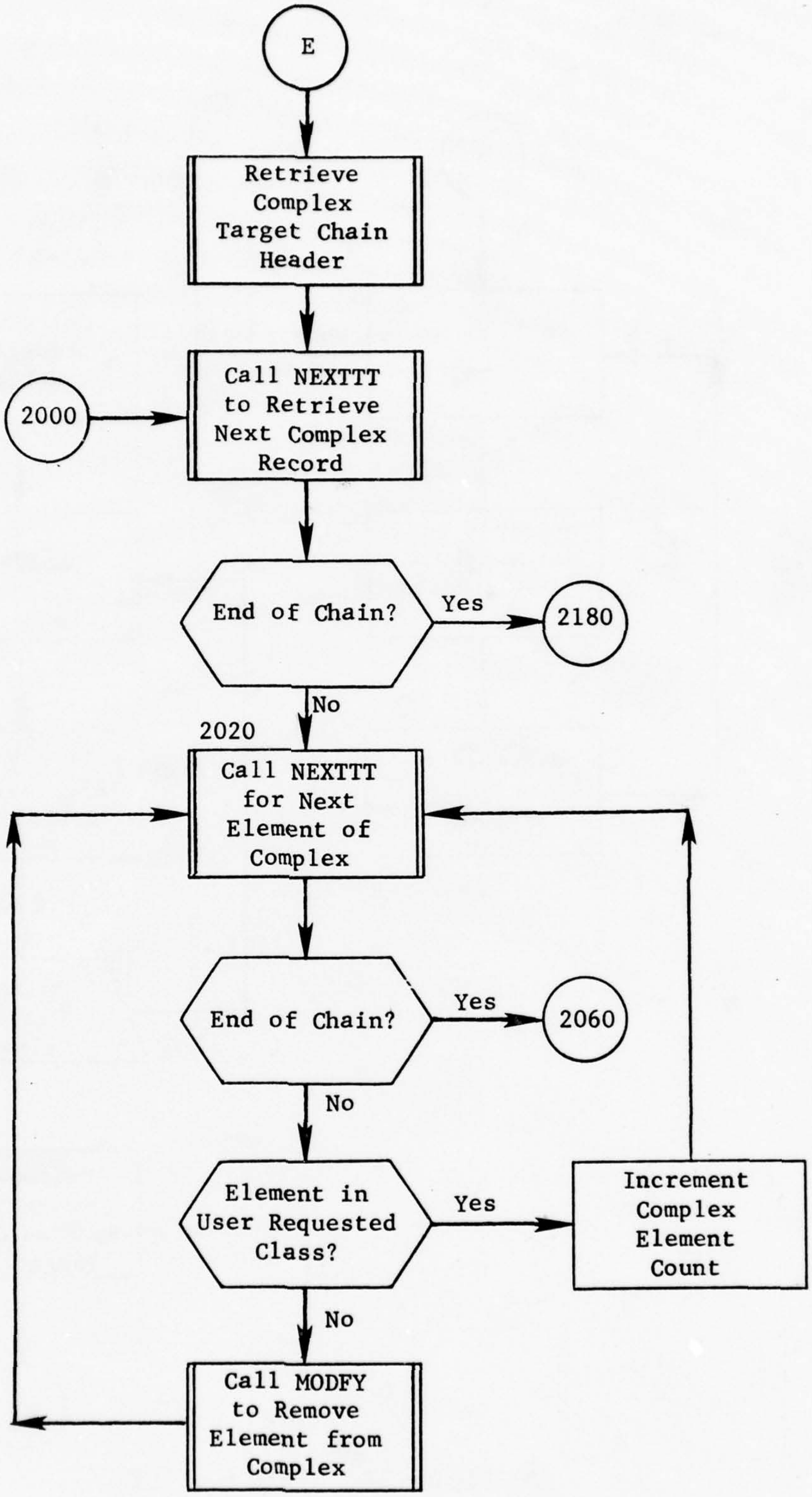

E
Retrieve Complex Target Chain Header
2000
Call NEXTTT to Retrieve Next Complex Record
End of Chain?
Yes
2180
No
2020
Call NEXTTT for Next Element of Complex
End of Chain?
Yes
2060
No
Element in User Requested Class?
Yes
Increment Complex Element Count
No
Call MODFY to Remove Element from Complex

Figure 27. (Part 7 of 13)

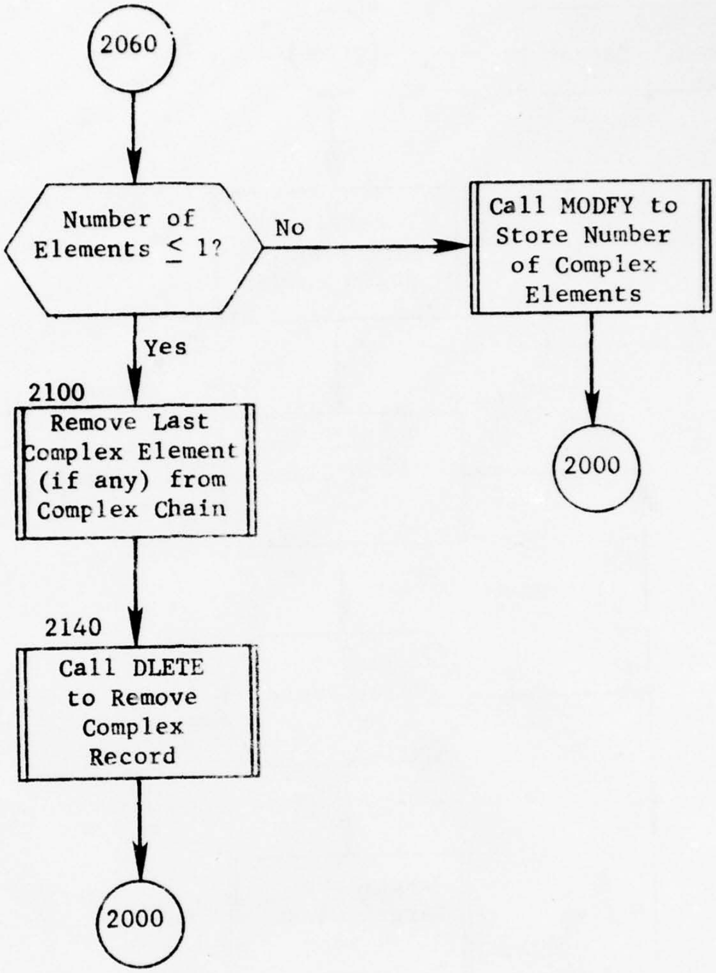

2060
Number of Elements ≤ 1?
No
Call MODFY to Store Number of Complex Elements
2000
Yes
2100
Remove Last Complex Element (if any) from Complex Chain
2140
Call DLETE to Remove Complex Record
2000

Figure 27. (Part 8 of 13)

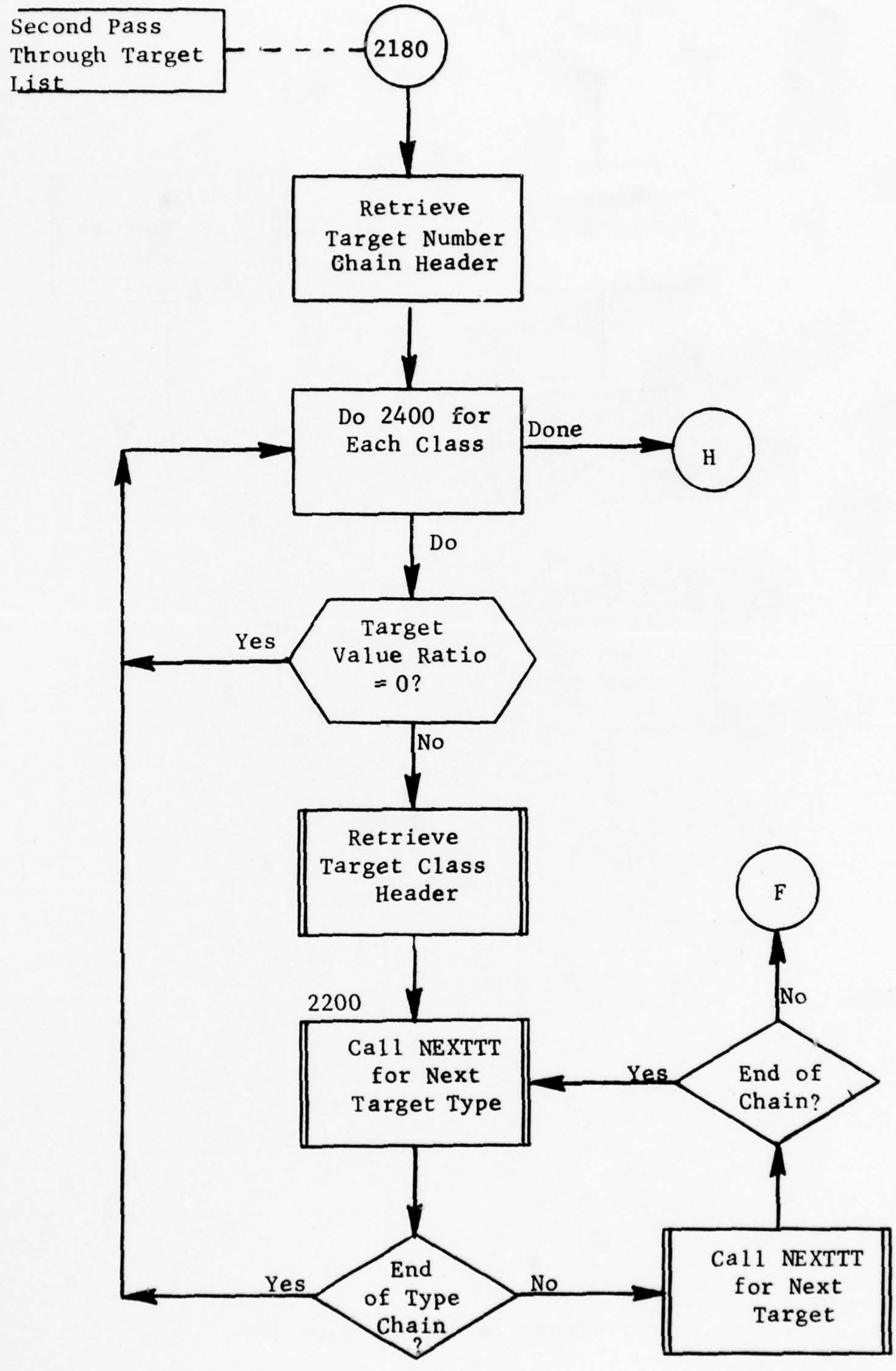

Second Pass Through Target List
2180
Retrieve Target Number Chain Header
Do 2400 for Each Class
Done
H
Do
Yes
Target Value Ratio = 0?
No
Retrieve Target Class Header
F
No
2200
Call NEXTTT for Next Target Type
Yes
End of Chain?
Yes
End of Type Chain ?
No
Call NEXTTT for Next Target

Figure 27. (Part 9 of 13)

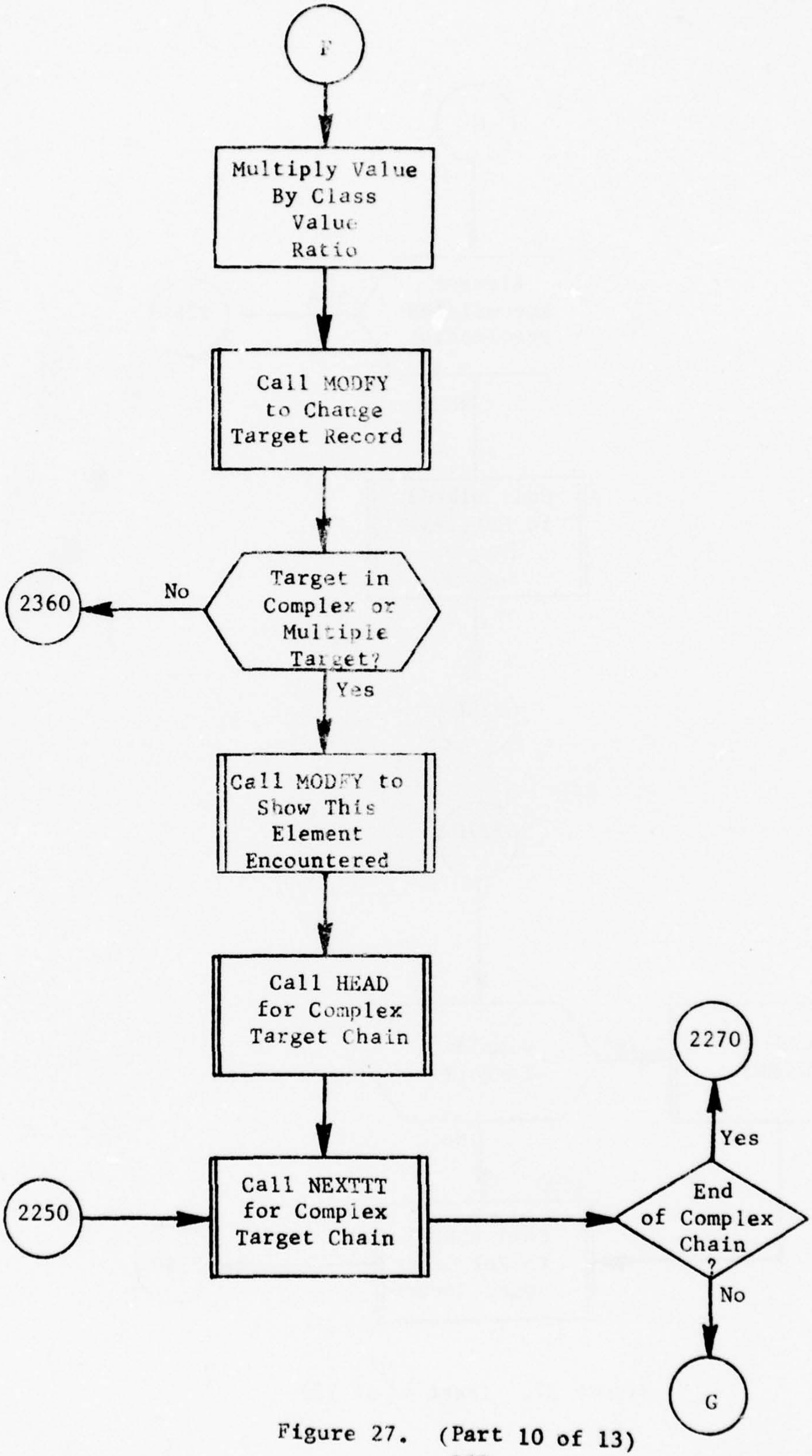

F
Multiply Value By Class Value Ratio
Call MODFY to Change Target Record
Target in Complex or Multiple Target?
No
2360
Yes
Call MODFY to Show This Element Encountered
Call HEAD for Complex Target Chain
2250
Call NEXTTT for Complex Target Chain
End of Complex Chain ?
Yes
2270
No
G

Figure 27. (Part 10 of 13)

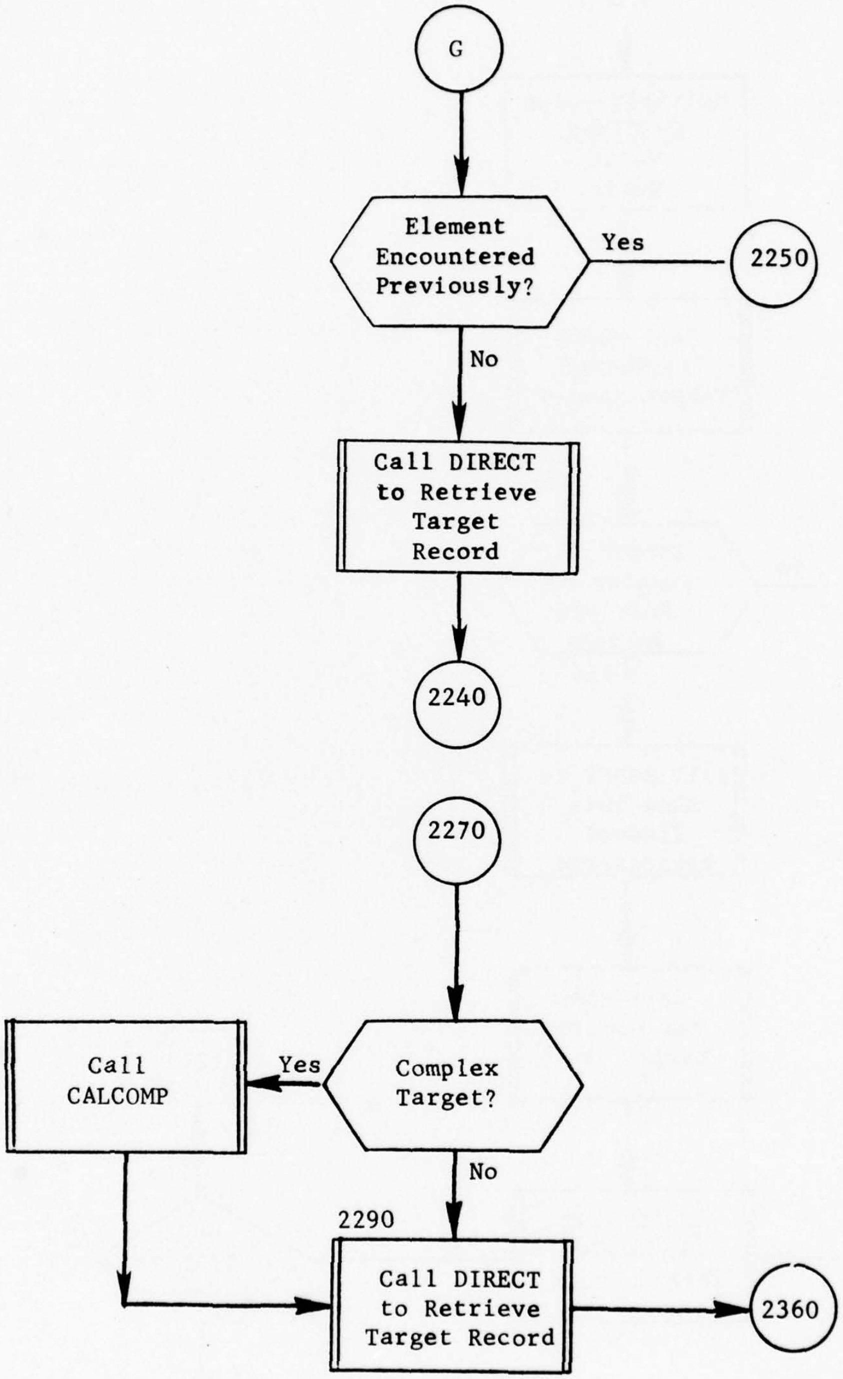

G
Element Encountered Previously?
Yes
2250
No
Call DIRECT to Retrieve Target Record
2240
2270
Complex Target?
Yes
Call CALCOMP
No
2290
Call DIRECT to Retrieve Target Record
2360

Figure 27. (Part 11 of 13)

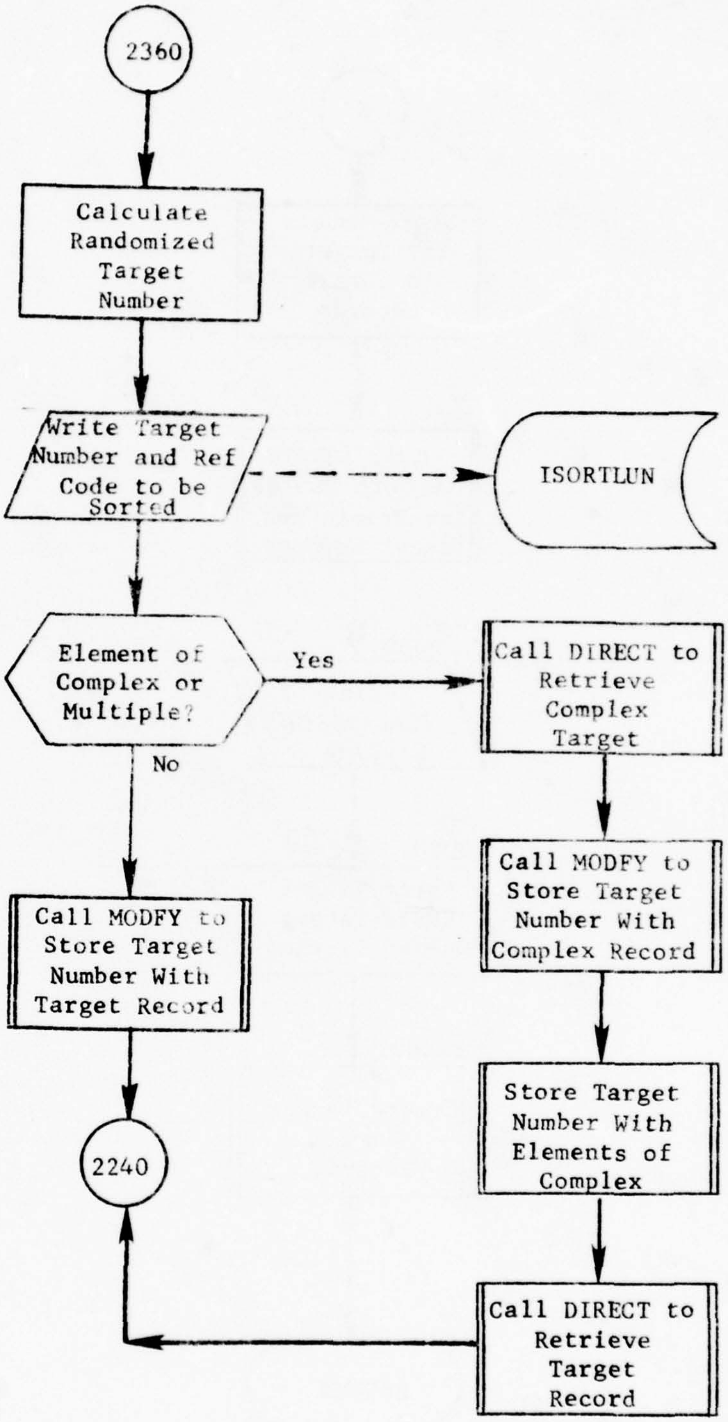

2360
Calculate Randomized Target Number
Write Target Number and Ref Code to be Sorted
ISORTLUN
Element of Complex or Multiple?
Yes
No
Call DIRECT to Retrieve Complex Target
Call MODFY to Store Target Number With Complex Record
Store Target Number With Elements of Complex
Call DIRECT to Retrieve Target Record
Call MODFY to Store Target Number With Target Record
2240

Figure 27. (Part 12 of 13)

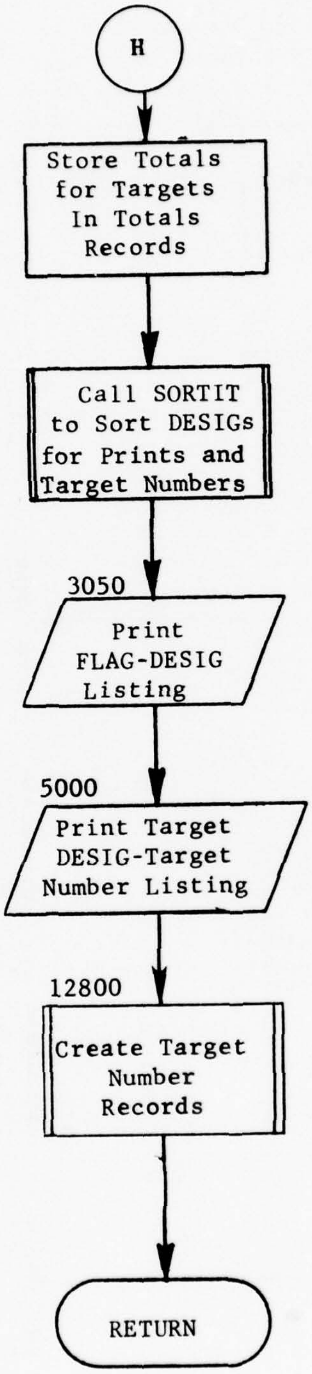

H
Store Totals for Targets In Totals Records
Call SORTIT to Sort DESIGs for Prints and Target Numbers
3050
Print FLAG-DESIG Listing
5000
Print Target DESIG-Target Number Listing
12800
Create Target Number Records
RETURN

Figure 27. (Part 13 of 13)

APPENDIX A

ANALYTICAL CONCEPTS AND TECHNIQUES

This appendix describes the major analytical concepts, techniques, and algorithms employed within modules JLM, INDEXER, and PLANSET. Topics of discussion consists of explanations of weapon grouping, missile reprogramming, target list preparation, and missile time of flight calculations.

A.1 Weapon Grouping

The initial phase of plan development provides an allocation of weapons* to targets. To reduce the amount of processing required during this phase, the offensive weapons are aggregated into "weapon groups" which for the purpose of the allocation, all weapons within a group are treated identically. This phase of processing is then followed by the sortie generation phase during which the specific missile and bomber plans are developed.

Grouping Criteria: On the basis of user input, which specifies the type (TYPE) weapons to be considered, module PLANSET processes the indexed data base and assembles the individual missile and bomber units (items in classes MISWEP and BMBWEP) into weapon groups.

A weapon group is defined as a set of weapons which are assigned to delivery vehicles that are located in the same geographic area and have like characteristics. Specifically, to be in the same group, these weapons must be of the same type; i.e., the attribute TYPE** must have the same value; they must have the same alert status (alert or non-alert), originate in the same geographic region, and have the same payload. Bombers must have the same refueling index (IREFUEL). In order for missiles or nonrefueling bombers to be grouped, they must lie within a geographic area which, for alert weapons, has a radius equal to a certain percentage of the range of the weapon. This percentage is a parameter RANGEMOD specified by the user for input to program PLANSET. If RANGEMOD is not specified, it is assumed to be 15 percent. The RANGEMOD value

* A weapon is defined here as a warhead plus the characteristics of its delivery vehicle.

** A single set of delivery vehicle characteristics is associated with all weapons of a given type. These characteristics include attributes (for all weapons) ALRTDL, CEP, FUNCTION, LCHINTVL, NLRTDL, PLABT, RANGE, REL, SIMLUNCH, SPEED (for missiles only), ALRTDB (if IREP greater than zero), IREP, NMPSITE, PDES, PFPF, PINC, RNGMIN, TOFMIN, IRECMODE, PRABT, RANGEDEC, RANGEREF, and SPDLO.

used for alert weapons is automatically doubled for nonalert weapons (to reduce the proliferation of groups). Under this criterion, it is appropriate to think of the weapons of a given group as being capable of attacking the same set of targets.

If the weapons are to be used exclusively against naval targets (a player option), all the weapons in the group must have the same value for the attribute PKNAV (the single shot kill probability for these weapons against targets of class NAVAL).

Group Centroid: As a new base is added to a group, the latitude and longitude of the group centroid are adjusted so that the final values reflect the true group centroid. That adjustment is effected as follows.

Let	NG	= The number of bases included in the group prior to this addition
	LSTLAT	= Latitude of centroid before addition
	LSTLONG	= Longitude of centroid before addition
	LAT	= Latitude of the weapon being added
	LONG	= Longitude of the weapon being added

Then for the new centroid latitude (NEWLAT),

$$\text{NEWLAT} = \frac{(\text{NG}*\text{LSTLAT})+\text{LAT}}{\text{NG}+1}$$

To determine the new centroid longitude (NEWLONG) an intermediate quantity (GLONG) is calculated. If GLONG $<$ 0, NEWLONG = GLONG + 360; otherwise NEWLONG = GLONG. GLONG is calculated as follows:

1. If $-180 \leq (\text{LSTLONG} - \text{LONG}) \leq 180$ then $\text{GLONG} = [(\text{NG} * \text{LSTLONG}) + \text{LONG}]/[\text{NG} + 1]$

2. If $(\text{LSTLONG} - \text{LONG}) > 180$ then $\text{GLONG} = [(\text{NG} * [\text{LSTLONG} - 360]) + \text{LONG}]/[\text{NG} + 1]$

3. If $(\text{LSTLONG} - \text{LONG}) < -180$ then $\text{GLONG} = [(\text{NG} * \text{LSTLONG}) + (\text{LONG} - 360)]/[\text{NG} + 1]$

Basic Yield (Bombers): One of the composite characteristics calculated for a bomber group is its basic yield per gravity bomb. That value is obtained as follows. Define the following variables:

NOBOMB1	Number of bombs of type 1
NOBOMB2	Number of bombs of type 2
YIELD1	Yield of type 1 bomb
YIELD2	Yield of type 2 bomb

Then, the basic yield, YIELD(G), for a group is calculated as

$$\text{YIELD(G)} = \left[\frac{\text{NOBOMB1} * \text{YIELD1}^{(2/3)} + \text{NOBOMB2} * \text{YIELD2}^{(2/3)}}{\text{NOBOMB1} + \text{NOBOMB2}}\right]^{3/2}$$

It is this basic yield which is used for all bombs of the group during the allocation phase (module ALOC). The actual yield of the ASMs is used for these weapons.

MRV/MIRV Payloads: In QUICK, those missiles equipped with a multiple reentry vehicle (MRV) capability are allocated to a single target. For allocation purposes, the component RVs (reentry vehicles) are considered to be a single warhead; however, the added effect of the MRV's pattern is reflected in the formula used to determine its expected yield:

$$\text{MRV yield} = (\text{yield for one warhead of the given type}) * (\text{the number of warheads, or RVs})^{3/2}$$

The number of warheads (reentry vehicles) is raised to the 3/2 power in order to accommodate the "2/3 rule" for comparing the yield of N MRV warheads delivering X megatons each against the yield of one warhead of NX megatons striking the target center.

Multiple independently targetable reentry vehicles (MIRVs), on the other hand, are allocated as separate weapons, subject to footprinting constraints. Hence, for the case in which the independently targetable reentry vehicles (IRVs) of a missile with MIRV capability are in turn equipped with MRVs, the expected yield calculated is:

Yield for missile with MIRV capability =

$$(\text{yield for one warhead of the given type}) * (\text{the number of IRV's}) * (\text{the number of warheads, or RV's, per IRV})^{3/2}$$

Command and Control Reliability: Each weapon item in the data base is assigned to a command and control region (IREG) by the user. This command and control region is an arbitrary designation for the extent of command and control functions and has no geographic meaning. The reliability for command and control (CC) is a function of this region IREG. Thus, the user must divide the offensive weapons systems into these "regions" according to the command and control which is appropriate for the plan being developed. The maximum number of command and control regions is 20.

Overallocation: The QUICK weapon allocator is designed to assign the individual weapon of a group to specific targets. In developing this allocation, program ALOC does not consider serial bombing constraints or MIRV footprint constraints. These constraints reflect the physical limitations on a delivery vehicle's ability to deliver warheads to geographically separated targets. In addition, in allocating bomber weapons, the number of weapons associated with a given penetration corridor may not correspond to an integral number of delivery vehicles.

The above constraints are considered in the sortie generation phase of plan development. To provide some flexibility in developing feasible weapon assignments for each delivery vehicle, a few extra weapons are added to each MIRV and bomber weapon group for allocation by module ALOC. Subsequent processing by modules POSTALOC (for bombers) and FOOTPRNT (for MIRVs) removes this overallocation in creating the sortie specifications.

For this reason, a number of weapons are artificially added to each weapon group. The formula used to add these weapons is as follows:

$$NEX = NWOLD * (PEX + EXB/NVOLD)$$

where NEX = number of weapons added to group
NWOLD = original number of weapons in group
NVOLD = original number of vehicles in group
PEX = percentage extra factor
EXB = extra vehicle factor

There is one set of increase factors (PEX and EXB) each for bombers, non-MIRV missiles, and MIRV missiles. These increase factors are user-input parameters.

This excess of weapons appears as an over-allocation of weapons from the weapon allocation phase. The sortie generation phase removes this over-allocation in creating the sorties. Thus the final number of weapons for which plans are generated closely approximates the number requested in the data base. (In some extreme cases, some weapons may be omitted.)

In order that the Plan Generator will perceive the correct number of expected weapons, the survival before launch probability (SBL) is modified to reflect this change.

If: NACTUAL = actual number of weapons in a group
NEXCESS = number of weapons added to the group

then: $$SBL = SBLREAL * \left[\frac{NACTUAL}{NACTUAL + NEXCESS}\right]$$

The actual survival before launch probability (SBLREAL) is used after the excess weapons have been removed in the sortie generation phase of plan development.

A.2 Missile Reprogramming

Each missile type in the data base has an associated attribute IREP which indicates its reprogramming capability. Missiles may be retargetable, for instance, if other weapons in the squadron have been destroyed before launch, during launch, or in powered flight. The reprogramming capabilities considered within the Plan Generator are:

No reprogramming capability (IREP=1)
Reprogramming for not in commission (IREP=2)
Reprogramming for destruction before launch (IREP=3)
Reprogramming for failure through launch (IREP=4)
Reprogramming for failure through powered flight (IREP=5)

During QUICK plan generation, this reprogramming capability is exercised only if the user specifies a RETARGET option in program PLANSET. The effects of missile reprogramming during plan generation are to: 1) decrease the number of vehicles per squadron; 2) reduce the DBL probability for alert vehicles to zero for those missiles which reprogram for this failure mode; and 3) increase the reliability factor for reprogrammable missiles. In computing replacement values for these parameters, the data base value associated with the following attributes is considered.

PINC: Probability that the missile is in commission
ALRTDB: Probability of DBL for alert vehicles
PLABT: Probability of a launch abort
PFPF: Probability of failure during powered flight

Table 5 shows the method of calculating replacement values for each level of reprogramming capability. To illustrate the reprogramming calculations, let N be the original number per squadron, R the original reliability for any missile squadron, and S be the probability of survival before launch. If N' is the reduced number of weapons, R' the increased squadron reliability resulting from reprogramming calculations, and S' the modified survival probability, N'*R'*S' will still equal N*R*S. The new values, however, reflect the probability, with retargeting, of striking the N' highest priority targets to be assigned to the squadron. For example: for a non-SLBM (submarine-launched ballistic missile) missile squadron with attributes IREP=3, PINC=.8, ALRTDB=.1, PLABT=.2, and PFPF=.3, and a number per squadron of 30, the new attribute values assigned (see table 5) are:

Number per squadron = PINC(1-ALRTDB) (N)
= (.8)(1-.1)(30) = 21.6, truncated to 21
New ALRTDB = 0
= (1-PLABT)(1-PFPF)
Reliability = (1-.2)(1-.3) = 56

Had reprogramming not been considered, the values would have been:

Number per squadron = 30
ALRTDB = .1
Reliability = .8(1-.2)(1-.3) = .448

Table 5. Computations for Reprogrammable Missiles

Reprogramming Capability Index		New number per squadron (N = original number)	New ALRTDB for this type	Reliability for this type
IREP = 1		N	ALRTDB	PINC *(1-PLABT)*(1-PFPF)
IREP = 2		PINC * N	ALRTDB	(1-PLABT)*(1-PFPF)
IREP = 3	Non SLBM	PINC *(1-ALRTDB) *N	0	(1-PLABT)*(1-PFPF)
	SLBM	N	#	#
IREP = 4	Non SLBM	PINC *(1-PLABT) *(1-ALRTDB)	0	1-PFPF
	SLBM	PINC *(1-PLABT)*N	ALRTDB	1-PFPF
IREP = 5	Non SLBM	PINC *(1-ALRTDB) *(1-PLABT)*(1-PFPF)*N	0	1
	SLBM	PINC *(1-PLABT) *(1-PFPF)*N	ALRTDB	1

Reprogramming for destruction before launch is not applicable to submarine-launched weapons since the destruction of one launch site detroys all remaining missiles in the squadron.

A.3 Target List Preparation

The information provided to the Plan Generator consists of information on the target system which is to be attacked and on the available weapon systems which have been provided to deal with the target system. The weapon allocator (module ALOC) receives its targets as a shuffled target list; that is, a list of targets that are arranged in a random order. To be discussed are: target categories, time dependent target, target value, methods of complexing, and target shuffling (or randomizing).

Target Categories: From a computational point of view, QUICK considers three categories of targets: simple targets, multiple targets, and complex targets. Target numbers are assigned to all simple targets, multiple targets, and complex targets in classes 1-15 for both sides, one side at a time. A simple target is a single data base item with a single unique geographical location.

The concept of a multiple target was added to the system to increase its speed in dealing with missile squadrons. For example, a Minuteman squadron may have as many as 50 separately targetable points. From the targeting point of view, all these points have essentially the same geographic location, the same value, and the same characteristics. For efficiency in processing, therefore, QUICK allows multiple targets. A multiple target is defined as several independent, identical missile targets (such as separate missile silos in a Minuteman squadron) that are close together relative to the range of the weapon systems, but far enough apart that each target element must be treated as an independent aim point. For such targets, the right targeting for one of them is undoubtedly the right targeting for them all. Thus, the Plan Generator determines the targeting of all elements of a multiple target through a single calculation of targeting for a representative target (of the appropriate multiplicity).

The third category of target, the complex target, allows the Plan Generator to deal with targets consisting of several elements and to treat them as a single simple target during the weapon allocation phase. Complex targets are formed by the Weapon/Target Identification subsystem (INDEXER) and consist of target elements (up to 40 data base items) in which each element is separated from some other element in the complex by a distance not greater than one-half the sum of the lethal radii of the two elements from a given weapon yield, considering the vulnerabilities for each of the elements. The complexing yield assumes a default value of one-megaton or any user input value. Also, at user discretion complexing yield may be obtained through query of VN and associated table look ups (see subsection under target complexes). Under either criterion, the complex target is input to program ALOC as a single element target with characteristics which are representative of the complete complex. The procedures used in identifying and describing this representative target element are discussed later in this chapter (see subsection Complex Targets).

With the above simplifications, the method of allocation used by program ALOC can be essentially the same for all three types of targets.

Time-Dependent Target Value: The relative value of the targets considered during plan generation is established on the basis of two sets of input data supplied by the user. In the data base each potential target is assigned a value (VAL) which establishes its relative worth within its assigned class. Then, the user provides data, for input to module PLANSET, which establish the target's value relative to all other potential targets in the game base (see Target Value, this appendix).

Since the relative strategic worth of a target may degrade over time (e.g., the value of a missile launch site before and after launch), the time dependence of target value must be considered in developing the attack plan. In QUICK, this relationship is established on the basis of data supplied by the user and included in the data base. The user can specify up to five separate time components which represent specified fractions of the total target value. For each of the five components, the user specifies the time (in hours) at which the value changes, T(I), and the fraction of the target value that is removed at that time, FVALT(I).

In using these data, the system (module PLANSET) automatically assumes a standard uncertainty in times specified. Figure 30 illustrates the time dependence of target value. As a result of the "value escaped" method of target damage calculation and the launch interval timing considerations, target value cannot increase at any time.

Target Value: The Plan Generator allocates weapons so as to maximize the target value destroyed. To accomplish this, the relative importance or value of the targets to be considered must be established. These target values reflect the major strategic objectives of the war plan which is to be generated. They must, therefore, be established by the user within the context of a specific game scenario.

The QUICK system uses a two-step procedure to input the user judgmental data required for target value calculations.

1. In the data base, each potential target is assigned a value calculated to reflect its relative worth within its assigned class.

2. To generate a specific plan, the user must also provide data to the Plan Generator (module PLANSET) which determine the relative value of the target classes, and hence all targets, for the current plan.

For the data base, a reasonably good judgment can be made of the relative values of the targets within each target class (such as missile, bomber, urban/industrial, or navel classes). The values may be based, for instance, on relative population or industrial importance for urban/industrial targets. For missile and bomber classes, the user will probably select target values which take into account each weapon's effective

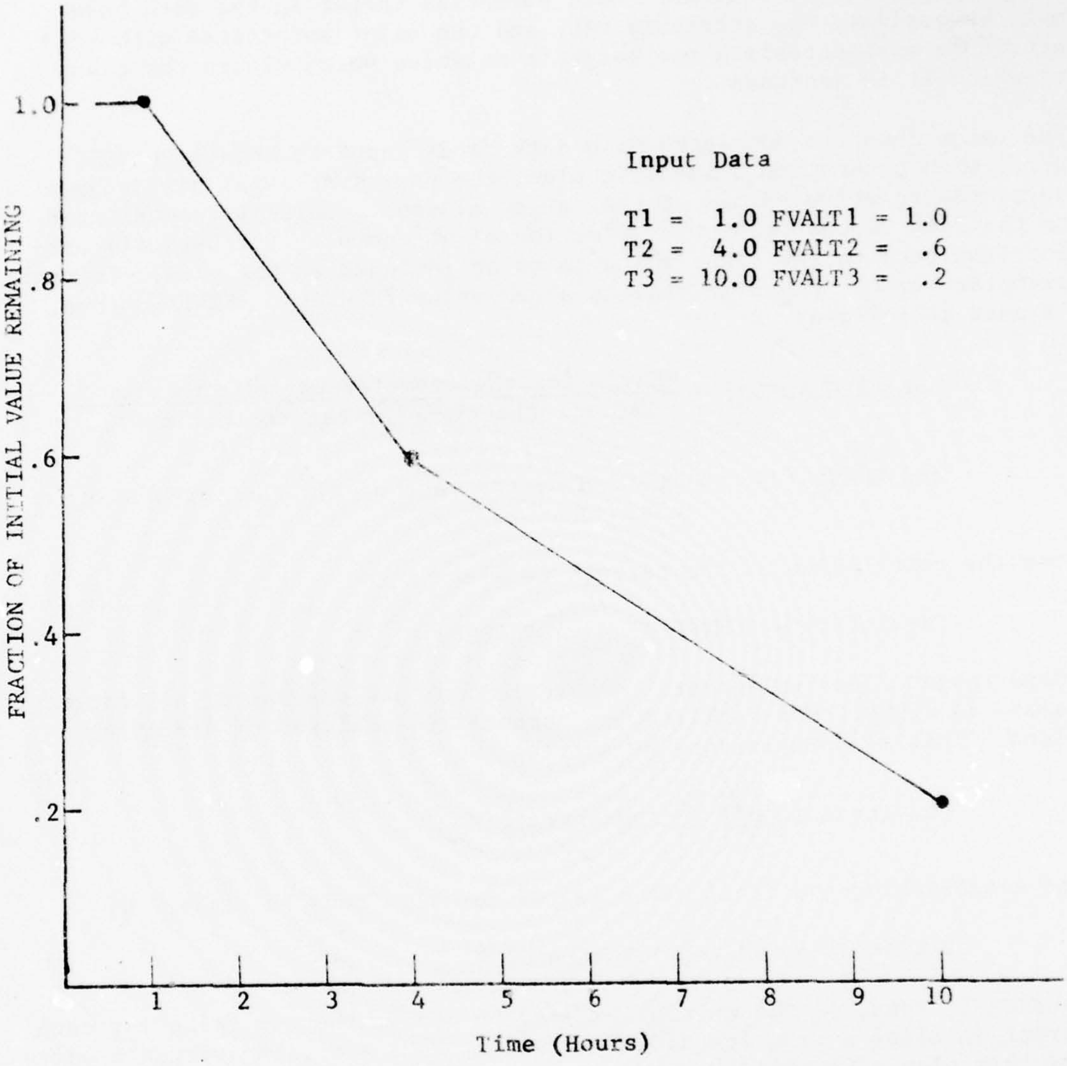


Figure 30. Time Dependent Target Value Curve

megatonnage, range, and CEP. Each potential target in the data base must be assigned the attribute VAL, and the value associated with this attribute must establish the target's relative worth within the class to which it is assigned.

The value input is completed with data cards input to module PLANSET. Here, when generating a specific plan, the user must input his judgment as to the relative values of the target classes. This is communicated to the Plan Generator by the selection of an exemplar (or typical) target from each target class which is to be included in the plan. To that exemplar target, the user assigns a new value (NEWVAL). NEWVAL, then, is used as follows:

$$\text{Let VALCLASS(J)} = \frac{\text{NEWVAL for the exemplar target in class J}}{\text{VAL for the exemplar target in class J}}$$

and CUMVAL(J) = the sum of the VALs of all the targets in class J

Then the total value of the targets in class J is

$$\text{CUMVALF(J)} = \text{CUMVAL(J)} * \text{VALCLASS(J)}$$

These target class values are then scaled so that the sum of all target values is 1,000, thus facilitating comparative analyses of differing plans. This scaling is done by setting

$$\text{SUMVALX} = 1000\Bigg/\left(\sum_{j=1}^{15} \text{CUMVALF(J)}\right)$$

and establishing the final value factor for all items in class J by

$$\text{VALFAC(J)} = \text{SUMVALX} * \text{VALCLASS(J)}$$

VALFAC(J), then, is the multiplier used to derive the new value for each target in class J from its data base value, VAL; i.e., the target's value for this plan = VALFAC(J) * VAL.

The QUICK value scheme allows the user to reflect a relative judgment between the worth of two specific targets in different classes, rather than to decide the total distribution of VALUE which is to be apportioned between those two classes. This judgment is much more analogous to the usual strategic decisions. It is generally easier to specify the relative worth of Moscow vs. an SS-9 missile site than it is to specify the fraction of value that will be associated with urban/industrial targets vs. missile sites. In order to better illustrate this exemplar value scheme, a simple set of four targets is shown in table 6. In this table, one exemplar target from each class is assigned a value. The final calculated values used in the allocator sum to 1,000 and maintain the original data base ratios within each class. Also, the ratio of values between the exemplar targets is the same as the ratio between the user inputs.

Table 6. Sample Exemplar Target Value Calculation

TARGET CLASS	TARGET NAME	DATA BASE VALUE (VAL)	USER INPUT EXEMPLAR VALUE (NEWVAL)	FINAL CALCULATED VALUE
U/I	Moscow	80	16	400
U/I	Kiev	60		300
Missile	Ipich	5	10	250
Missile	Aag	1		50
			Total	1,000

The intermediate calculations used to derive the final calculated values above are:

	U/I CLASS	MISSILE CLASS
VALCLASS	16/80 = .2	10/5 = 2
CUMVAL	80 + 60 = 140	5 + 1 = 6
CUMVALF	.2(140) = 28	2(6) = 12
VALFAC*	.2(25) = 5	2(25) = 50

*Where SUMVALX = 1000/(28 + 12) = 25

Complex Target: A complex target consists of target elements in which each element is separated from some other element in the complex by a distance not greater than one-half the sum of the complexing lethal radii of the two elements. A complexing weapon yield plus the vulnerabilities of each target element determines the lethal radii. A single global user input weapon yield (a default of one-megaton is assumed) may be used for complex determination. That is, for each target element, the same weapon yield will be used for lethal radius determination. A second optional method of lethal radii calculation is permissible which employs table lookups of hard coded values. Through observation of the VN of each target element a yield is obtained from the table, then an adjusted VN is calculated and, finally, from the yield and adjusted VN tables are entered and lethal radii obtained. After lethal radii determination the collection of target elements and setting of target attributes (see below) is identical.

The potential target list input to module ALOC reflects the complex target as a single element. The target attributes for this representative target, calculated in module PLANSET, are derived from the target data associated with the individual elements of the complex. The largest target radius associated with any element of the complex is assigned as the radius* (TGTRAD) of the representative target. Similarly, the maximum value of TARDEF (local bomber defense potential) is assigned to represent the complex. The target value (VAL) and the number of terminal defense missile interceptors (NTINT) assigned each element are accumulated and their totals assigned to the representative element. MINKILL (the minimum kill probability required) and MAXKILL (the maximum kill probability desired) are weighted (by VAL) averages of the element attributes. The time dependence of the value of the complex, which is due to the time components of its elements, is approximated by at most five time components. That approximation is accomplished as follows.

First, the list of time components is checked for equal values. If any are found, the corresponding values are added together, and all but one of the equal components are removed from consideration, along with any zero components. If the number of remaining entries does not exceed five, the time dependence of the complex is approximated by these time components. Otherwise, an elimination procedure to reduce the number of entries to five is performed. For this, the slopes (change in value per change in time) are calculated for all remaining value points and the value point that produces the smallest slope is grouped together with its neighboring value point. Hence the number of entries is reduced by one. If more than five entries still remain, the above accumulation process is repeated, until no more than five time components remain.

*This technique represents an oversimplification. However, computing an adjusted radius, based on the geographic locations and dimensions of each target element, would not necessarily be an improvement. In order to provide a significantly more accurate treatment, a much more detailed analysis would be required of each complex target, which should take into account the yield and accuracy of the available weapons as well as the number, hardness, and geographic distribution of target elements.

Air lethal radius for each target in the complex must be determined for a uniform height of burst over the complex. Air lethal radius associated with simple targets are calculated assuming an optimal height of burst based on the target's vulnerability. Since a complex may be comprised of targets with differing vulnerabilities, it's necessary to define one height of burst over the entire complex. Accordingly, the height of burst for the complex is defined as the optimal scaled height of burst associated with the hardest target in the complex. Using this defined air height of burst, the air lethal radius for each target in the complex is computed.

Similarly, the hardness components (VULN1, VULN2) and the corresponding fractional value (FVALH1) which represent the complex are determined by first taking, for each target of the complex, its VAL, FVALH1, the number of hardness components (1 or 2), and the lethal radius corresponding to each hardness. The complement of FVALH1 is found to represent the second hardness component. If either fractional value is nonzero, it is multiplied by VAL to obtain the actual value at that hardness. After all targets have been considered, the lethal radii are separated into radii belonging to hard targets (radii less than 1.5 miles) and radii belonging to soft targets. The average lethal radius, weighted by the actual value at the corresponding hardness, is calculated from both hard and soft targets for those radii, and the result (HHARD or HSOFT) is assigned to the complex. Similarly, the actual value at each hardness (VHARD or VSOFT) is accumulated. If there are no hard targets (i.e., VHARD = 0), FVALH1 for the complex is set to 1; otherwise the fraction of actual value for hard targets (VHARD/VTOT) is assigned to FVALH1. This FVALH1, then, and the corresponding number of hardness components are assigned to the complex.

The index number (INDEXNO) and the target designator code (DESIG) associated with the representative target will normally be the INDEXNO and DESIG assigned to the first member of the complex (i.e., the first element of the complex encountered when processing the game data base). The user may, however, establish criteria for selecting the representative INDEXNO and DESIG (a control feature used in RISOP development). The procedure for exercising this option are outlined in the Users Manual (module PLANSET).

Target Shuffling: During the allocation phase of plan generation, the rate of allocation for each weapon group is monitored as the targets are processed. To prevent these rates from being biased by a large number of similar targets considered consecutively, the basic target list is shuffled.

Since similar targets appear together in the data base (by class and type), target shuffling randomizes the order in which various types of targets are encountered. Thus the rate of allocation provides a good estimate of whether a group is being over-allocated or under-allocated. The algorithm used to achieve the required shuffling is accomplished in the following manner.

Consider the target indices (I) as equally spaced points on a circle, with targets in a partiuclar class occurring consecutively. If the Ith point is displaced along the circle to the index

$$\frac{1}{2}(3 - \sqrt{5})\ C^{*}\ (I) \quad \text{mod } C$$

where C is the number of points on the circumference of the circle, the result will be the desired distribution. To accomplish the corresponding reordering of the discrete list of targets, each index must be multiplied by

$$\frac{1}{2}(3 - \sqrt{5})\ N$$

where N is the number of elements in the list, and then reduced by modulo N. A direct application of this procedure, however, will result in some cases with the same final index being assigned to more than one element.

Therefore, the following algorithm is used by program PLANSET to assign n new indices to the elements of the list in such a way that the above criterion is satisfied and each index from 1 to N is assigned exactly once.

Let L be the greatest integer such that $L \leq \frac{1}{2}(3 - \sqrt{5})\ N$

where N is the total number of targets.

Let P = J = L ,

where P = the beginning index of the current cycle, and
J = the index number currently being assigned.

As each list element is processed, J is replaced by J + L to obtain the next index number. If J becomes >N, J is replaced by J - N. If J becomes = P, a new cycle is to be started; 1 is added to P and to J, and the procedure continues as before.

A.4 Missile Time of Flight Calculations

This subsection outlines a set of equations to generate the time of flight (TOF) for single and multiple warhead missile systems.

The data needed for operation is as follows:

- o Launch location for booster (PX: Longitude; PY: Latitude)
- o Warhead data for each reentry object as follows:
  - Reentry angle (REANG); if no reentry angle is given, a minimum energy path is assumed

- Ballistic coefficient (BALC) of reentry vehicle

- Location of the target point (QX: Longitude; QY: Latitude)

The process results are a set of path segments that describe the movement of a launch vehicle and its subsidiaries from launch to target hit as follows:

o Launch: Launch to orbit insertion as a linear path with constant acceleration. Orbit insertion is assumed at the atmosphere limit (ATML). The time to complete this path is called TB.

o Elliptical Orbit: Elliptical orbit commensurate with a minimum energy trajectory or a specified angle to reentry into sensible atmosphere. The time to complete this path is called TF.

o Reentry to deceleration band: The first reentry segment is a linear path with constant velocity to the point where deceleration becomes significant. The time to complete this path is called TR1.

o Deceleration: The second reentry segment is a linear path with deceleration to the point where terminal velocity approached. The time to complete this path is called TR2.

o Terminal phase: The third reentry segment is a linear path with constant velocity to the point of detonation. The time to complete this path is called TR3.

The elliptical orbit path generation consumes the vast amount of the calculation. This path is calculated initially followed by the various paths defined within the earth's atmosphere. Kepler's equations are used for the elliptical orbit determination along with a heuristic cycling approach. The main problem is the earth's rotation for as the reentry vehicle spends time (the unknown parameter to be determined) above the atmosphere, the longitude of the aim point is moving and the amount of movement is dependent on the TOF. Iterative approaches resolves the dilemma. The aim point longitude is calculated using a TOF from a previous set of calculations (initially set to zero) and a new TOF is determined based on the given longitude. If the new TOF compares within tolerances (say 5 seconds) of the previously determinated TOF the solution is final. If the two TOF are outside tolerances, the new TOF is used to calculate the aim point longitude and equations are solved anew. This iterative approach converges for all possible trajectories.

In addition to defined parameters the following parameters are necessary for calculations:

TTO: Total TOF on a previous interaction (initiated to zero)

ERAT: Earth's rotational rate

ACON: Earth's radius

G: One over the square root of the gravitational constant

<u>Elliptical Orbit Determination</u>: Begin calculations of elliptical parameters by computing great circle range angle 'ALPHO2' between launch and target points as:

SY = SINE (PY)
QX = QX + ERAT (TF -TTO)
TTO = TF
WQ = SY * SINE (QY)
COSA = WQ * COSINE (PX -QX) + COSINE (PY) * COSINE (QY)
ALPHAX = COS^{-1} (COSA)
ALPHO2 = ALPHAX * .5
SA = SIN (ALPHO2)

Compute distance to unoccupied focal point (DTFP) from target point in plane of ellipse, and distance between focal points (FF). Calculations assume a synetrical ellipse about apogee of target and launch, see figure 31.

For a specified reentry angle calculate:

DTFP = (AGON + RR) * SA/SINE (2.*REANG + ALPHO2)
FF = (DTFP * SINE (2. * REANG)/SA

where RR is the altitude of the aim point.

For no specified reentry angle, assume minimum energy trajectory and calculate:

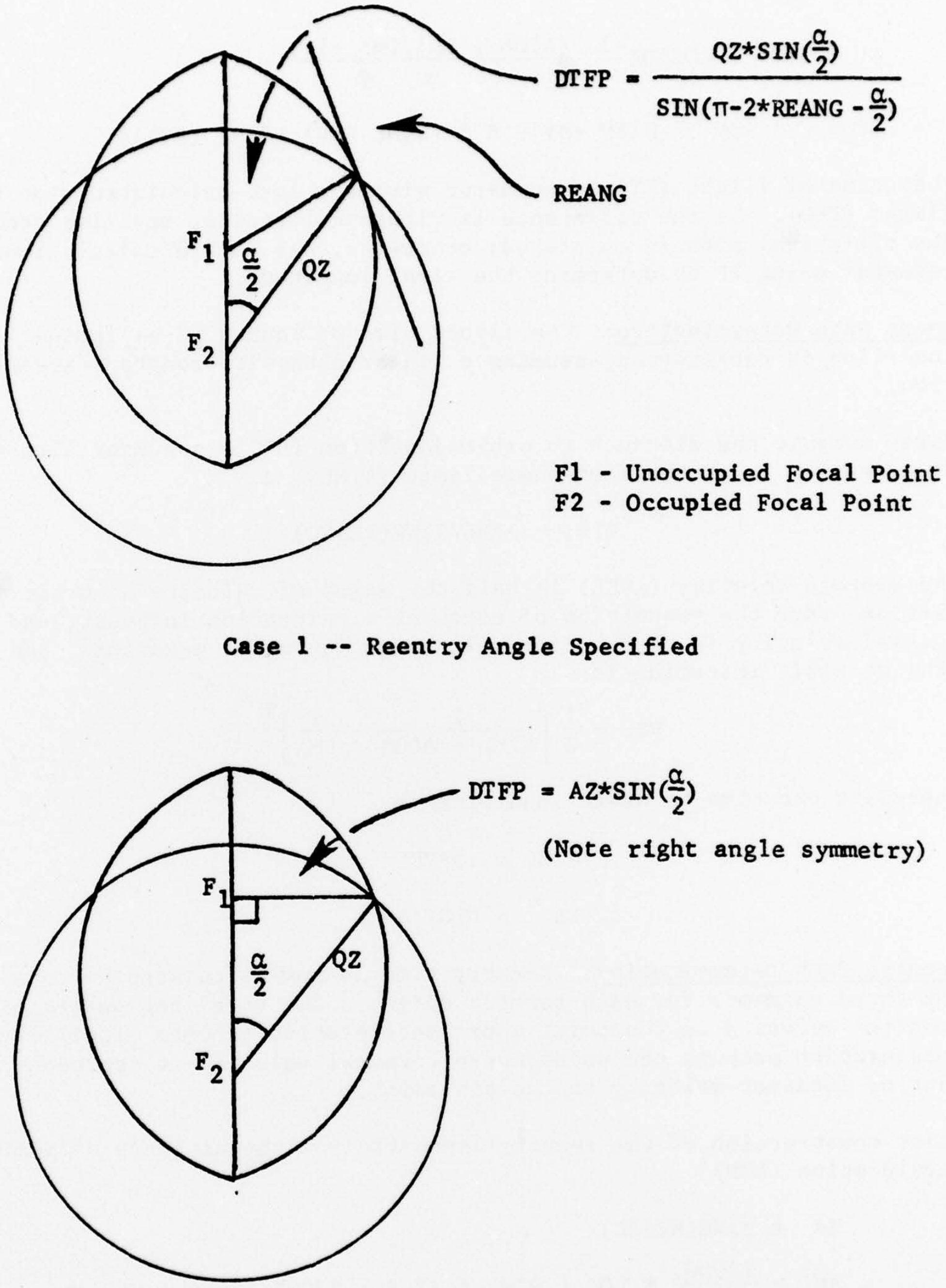
DTFP = (DTFP + RR) * SA
FF = (ACON + RR) * COSINE (ALPHAO2)
REANG = .5 (90^{o} - ALPHO2)

No calculate the semimajor axis (SMA) and eccentricity (E), of the ellipse.

SMA = (ACON + RR + DTFP) * .5
E = FF/(2. * SMA)

Elliptical flight time is computed as:

$$\mathrm{BV} = 360^{o} - \mathrm{COSINE}^{-1}\left[\frac{\mathrm{(ACON + RR)/SMA\ -1.)}}{\mathrm{E}}\right]$$


DTFP = $\frac{QZ*SIN(\frac{\alpha}{2})}{SIN(\pi-2*REANG - \frac{\alpha}{2})}$
REANG
F_1
$\frac{\alpha}{2}$
QZ
F_2
F1 - Unoccupied Focal Point
F2 - Occupied Focal Point

Case 1 -- Reentry Angle Specified

DTFP = AZ*SIN($\frac{\alpha}{2}$)
(Note right angle symmetry)
F_1
$\frac{\alpha}{2}$
QZ
F_2

Case 2 -- Minimum Energy Specified

Figure 1. Focal Point Calculations

$$FV = 360^{o} + COSINE^{-1} \frac{(ACON + RR)/SMA -1.)}{E}$$

$$TF = G * SMA^{3/2} [(FV -BV + E * (SINE (FV) -SINE (BV))]$$

This time of flight (TF) is compared with the last calculated time of flight (TTO). If the difference is within tolerances, say five seconds, the elliptical path is completed; otherwise, the entire calculations are repeated using TF to determine the final longitude.

Boost Path Determination: The flyout time of launch to elliptical orbit insertion is constructed assuming a linear path with constant acceleration.

First compute the distance to orbit insertion (DTOI) assuming altitude of insertion is at the atmosphere limit (ATML) as:

$$DTOI = ATML/SINE(REANG)$$

The average velocity (AVEL) is half the value of velocity at orbit insertion under the assumption of constant acceleration in boost, and that initial velocity is zero. From well known Keplerian equations, the velocity at orbit insertion is:

$$VEL = \frac{1}{G}\left[\frac{2}{ATML + ACON} - \frac{1}{SMA}\right]^{\frac{1}{2}}$$

Therefore the time to orbit insertion is

$$AVEL = .5*VEL$$

$$TB = DTOI/AVEL$$

Reentry Path Determination: Reentry time is now calculated by constructing three segments for each reentry object. The three segments are: constant velocity to the point where deceleration becomes significant; deceleration path to the point where terminal velocity is approach; segment of constant velocity to the aim point.

Start construction of the reentry segments from the altitude of maximum deceleration (AMD)

$$SA = SINE(REANG)$$

$$AMD = \frac{22000.}{6080.} * LOG [.0034 * 32.2 * 22000/(BALC * SA)]$$

Where BALC is the ballistic coefficient.

22000. scales the atmosphere density.

.0034 is the sea level density in slugs/FT^{3}.

Define half the altitude width of the deceleration segment as:

$$\text{HWAB} = 50{,}000/6080.$$

A constant is used since this segment has little variance for different weapons systems. Time for first segment is now

$$\text{TR1} = (\text{ATML} - \text{AMD} - \text{HWAB})/(\text{SA}*\text{VEL})$$

The velocity at the end of the second segment is computed as:

$$\text{CON} = \frac{-22000}{2} *32.2*.0034/(\text{BALC}*\text{SA})$$

$$\text{VELT} = \text{VEL}*\text{EXP} \quad \text{CON}*\text{EXP} \quad \frac{\text{HWAB} - \text{AMD}}{22000.}$$

Second segment time interval then is:

$$\text{TR2} = \frac{4.\ *\text{HWAB}}{\text{SA}(\text{VELT} - \text{VEL})}$$

The last segment time interval is:

$$\text{TR3} = \frac{(\text{AMD} - \text{HWAB})}{\text{SA}*\text{VELT}}$$

Total trajectory time, then, is the summation of the defined time segments.

A.5 User Inputs

Two new data base attributes are required as input for each missile system type. These are: the ballistic coefficient (in units of pounds per square inch) and, if desired, a reentry angle (in degrees). The omittion of a reentry angle for a given missile system type implies the generation of a minimum energy trajectory.

An altitude of orbit insertion (ATML) is also required as input. However, since total flight time variation is rather insensitive to this parameter, it is suggested that ATML be hard coded rather than be a user input requirement. Also, the definition of orbit insertion is atmospheric related, not missile related. A value of 200,000 feet includes 99% of the atmosphere.

THIS PAGE INTENTIONALLY LEFT BLANK

DISTRIBUTION

Addressee	Copies
CCTC Codes	
Technical Library (C124)	3
C124 (Stock)	6
C313	1
C314	17
C600	1
DCA Code	
205	1
EXTERNAL	
Chief, Studies, Analysis and Gaming Agency, OJCS ATTN: SFD, Room 1D957, Pentagon, Washington, DC 20301	2
Chief of Naval Operations, ATTN: OP-96C4, Room 4A478, Pentagon, Washington, DC 20350	2
Commander-in-Chief, North American Air Defense Command ATTN: NPXYA, Ent Air Force Base, CO 80912	2
U. S. Air Force Weapons Laboratory (AFSC) ATTN: AFWL/SUL (Technical Library), Kirtland Air Force Base, NM 87117	1
Director, Strategic Target Planning, ATTN: (JPS), Offutt Air Force Base, NE 68113	2
Defense Documentation Center, Cameron Station, Alexandria, VA 22314	12
	50

THIS PAGE INTENTIONALLY LEFT BLANK

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
CSM MM 9-77, Volume II

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle) The CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK), Program Maintenance Manual, Weapon/Target Identification Subsystem

5. TYPE OF REPORT & PERIOD COVERED

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
10
Dale J. Sanders,
Paul F. M. Maykrantz,
Jim M. Herrin
Edward F. Bersson

8. CONTRACT OR GRANT NUMBER(s)
DCA 100-75-C-0019

9. PERFORMING ~~ORGANIZATION NAME AND ADDRESS~~
System Sciences, Incorporated
4720 Montgomery Lane
Bethesda, Maryland 20014

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Command and Control Technical Center
Room BE-685, The Pentagon,
Washington, DC 20301

12. REPORT DATE
1 June 1977

13. NUMBER OF PAGES
188

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)
9 Computer system manuals

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
11 Nov 78   12 77 p.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
Approved for public release; distribution unlimited.

6 The CCTC Quick-Reacting General War Gaming System (QUICK) Program Maintenance Manual. Volume II. Weapon/Target Identification Subsystem. Change 1.

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
War Gaming, Resource Allocation

14 CCTC-CSM-MM-9-77-V2-CHG-1

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

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.

The Program Maintenance Manual consists of four volumes which facilitate maintenance of the war gaming system. This volume, Volume II, provides the programmer/analyst with a technical description of the purpose, functions, general procedures, and programming techniques applicable to the modules and

409 658

20. ABSTRACT (CONT'D)

subroutines of the Weapon/Target Identification Subsystem.

The Program Maintenance Manual complements the other QUICK Computer Manuals to facilitate application of the war gaming system. These manuals (Series 9-77 for Volumes I & II, Series 9-74 for Volumes III & IV) are published by the Command and Control Technical Center (CCTC), Defense Communications Agency (DCA), The Pentagon, Washington, DC 20301.