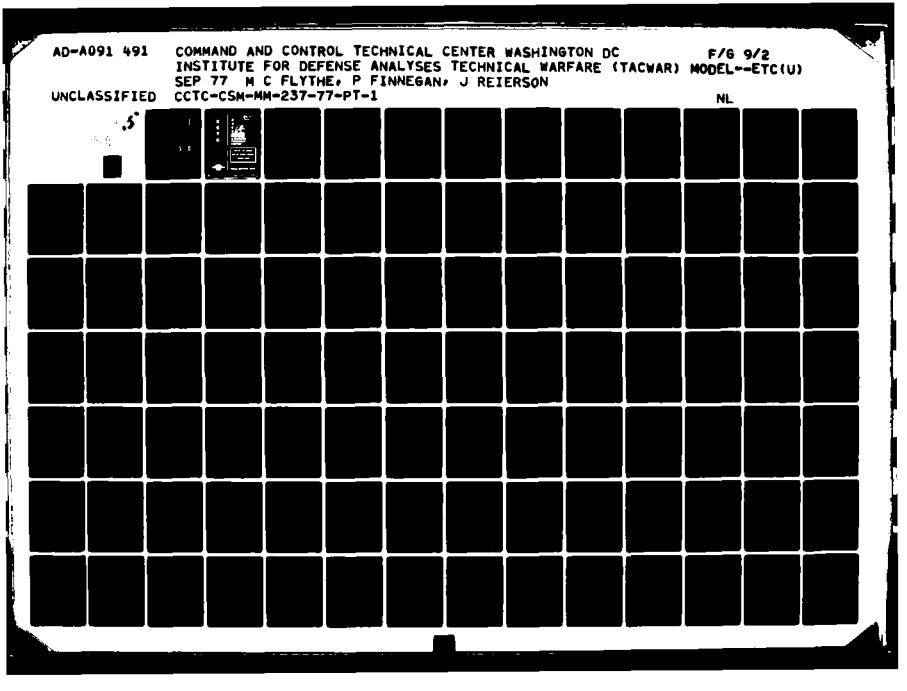
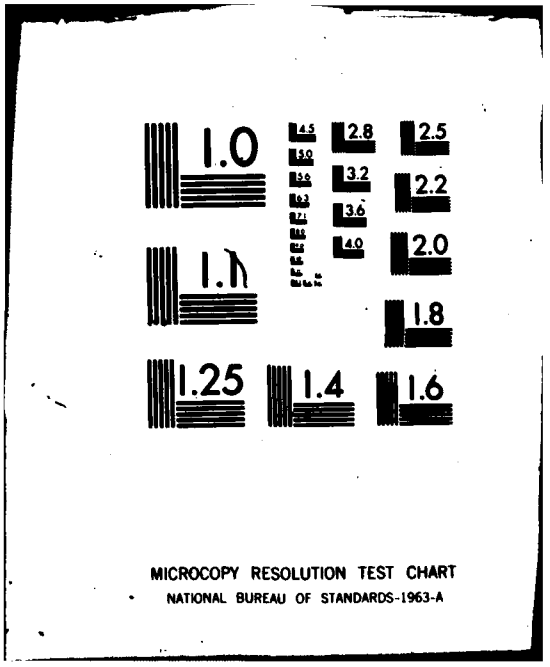
AD-A091 491 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 9/2
INSTITUTE FOR DEFENSE ANALYSES TECHNICAL WARFARE (TACWAR) MODEL--ETC(U)
SEP 77 M C FLYTHE, P FINNEGAN, J REIERSON
UNCLASSIFIED CCTC-CSM-MM-237-77-PT-1 NL




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
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

PHOTOGRAPH THIS SHEET

AD A091491

DTIC ACCESSION NUMBER

LEVEL II

INVENTORY I

Command and Control Technical Center, Wash., DC

"Institute for Defense Analysis Tactical Warfare (TACWAR) Model"

DOCUMENT IDENTIFICATION

6 Sept. 1977

Program Maintenance Manual Part I

Computer System Manual CSM MM 237-77 Part I

DISTRIBUTION STATEMENT A

Approved for public release; Distribution Unlimited

DISTRIBUTION STATEMENT

ACCESSION FOR	
NTIS	GRA&I
DTIC	TAB
UNANNOUNCED	
JUSTIFICATION	
BY	Per ltr. on file (FL-88/
DISTRIBUTION / AVAILABILITY CODES	80-2301, 17 Oct '80)

DIST	AVAIL AND/OR SPECIAL	
A		

DISTRIBUTION STAMP

DTIC ELECTE S NOV 13 1980 D

DATE ACCESSIONED

Classified reference, p.389, may remain.
Per: Dennis Konkel, CGTC (202) 695-4032,
pfcooper,
DTIC/DDA-2
13 Nov. '80

DATE RECEIVED IN DTIC

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-DDA-2

DTIC FORM OCT 79 70A

DOCUMENT PROCESSING SHEET

COMMAND AND CONTROL TECHNICAL CENTER

Computer System Manual CSM MM 237-77

6 September 1977

INSTITUTE FOR DEFENSE ANALYSIS TACTICAL WARFARE (TACWAR) MODEL

Program Maintenance Manual

Part 1

REVIEWED BY:

Randall B Saylor

CAPT RANDALL B. SAYLOR
Project Officer

APPROVED BY:

R E Harshbarger

R. E. HARSHBARGER
Acting Deputy Director
NMCS ADP

Copies of this document may be obtained from the Defense Documentation Center, Cameron Station, Alexandria, VA 22314

ACKNOWLEDGMENT

This manual was prepared for the Command and Control Technical Center (CCTC) under the direction of the Chief for Military Studies and Analysis with technical support provided by Computer Sciences Corporation under Contract Number DCA 100-74-C-0002.

CONTENTS

Section Page

Section Page

Section Page

Section Page

Section Page

Section Page

Section Page

Section Page

Section Page

Section Page

ILLUSTRATIONS

TABLES

ABSTRACT

The Institute for Defense Analyses (IDA) Tactical Warfare (TACWAR) model is a fully-automated combat simulation that can be used to assess the interaction of combat forces employing conventional, nuclear, and chemical weapons in a theater-wide campaign. This document presents the information necessary for programmer personnel to maintain the TACWAR model.

GLOSSARY

Abbreviation	Meaning
AAA	antiaircraft artillery
ABA	airbase attacker
ABAE	airbase attacker escort
ABAS	airbase attacker diverted to SAM-suppression
CAS	close air support
CASA	close-air-support attacker
CASD	close-air-support defender
CASE	close-air-support escort
CASS	close-air-support diverted to SAM-suppression
CEP	circular error probable
COMMZ	communication zone
FEBA	forward edge of battle area
INT	interdiction of division in reserve
QRA	quick reaction alert
SAM	surface-to-air missile
SSM	surface-to-surface missile
TOE	table of organization and equipment

SECTION 1. GENERAL

This section explains the purpose of the Program Maintenance Manual (PMM) and the use of the TACWAR model. Descriptions of the equipment environment and programming conventions are also included.

1.1 Purpose

The purpose of the Program Maintenance Manual is to provide programmer personnel with the information necessary to maintain TACWAR. This manual includes detailed descriptions and flowcharts of each subroutine and discusses program input and output. Also included are the procedures for maintaining and updating the files which support TACWAR.

1.2 System Application

The TACWAR model is designed to simulate nuclear and chemical warfare as well as conventional ground-air conflict of user-specified duration (in terms of 12-hour combat cycles) on a theater level. The major functions performed by the model are simulations of air, nuclear, chemical, and ground combat operations; target acquisition; theater control; and supplies transportation.

The air-combat model determines the allocation of aircraft to specific combat missions on the basis of either general or specific user-input allocations. The model calculates all air-to-air, ground-to-air, and air-to-ground attrition. The number of surviving aircraft by type on Close-Air-Support (CAS) missions calculated by this model is used as input to the ground combat model.

The target acquisition model determines the expected number of targets acquired by type and location over the length of a target acquisition cycle, given various input data on sensors, targets, and the environment. The model assumes that the general locations of major combat forces are known and that both sides operate sensors to acquire candidate targets for possible nuclear and/or chemical fires.

The nuclear model accounts for the assignment and assessment of nuclear weapons against various types of targets throughout the theater. The assignment of weapons to targets is

either user-specified or developed from user input by the internal assignment procedures. Nuclear damage assessments are made in an expected-value sense, using analytic functions to provide a rapid determination of nuclear weapon use. Firing constraints due to population centers are observed, and civilian casualties are assessed where necessary.

The chemical model accounts for the assignment and assessment of chemical weapons against various types of targets throughout the theater. This simulation is similar to that of the nuclear model.

The ground-combat model determines attrition to personnel and weapons by type within divisions as they interact in battle areas across the theater. It calculates the destruction of supplies due to enemy CAS missions. The model also calculates the capture of territory which is denoted by the movement of the Forward Edge of the Battle Area (FEBA) in each sector of the theater.

Theater control values for the first cycle of battle are computed to determine the relative value of one side's weapons against a standard force of the other side, the current combat effectiveness of all divisions, various geographical quantities, and the number and location of divisions in the first and second inactive battle areas of each combat sector. On subsequent cycles of battle, only non-battle quantities are computed, i.e., supply consumption, relocation of divisions, personnel and weapon replacement, weapon repair, and effectiveness of divisions. The theater-control model also moves divisions throughout the theater to allow each side to have the most effective front line possible with the present resources.

The supplies model determines the flow of supplies within the theater on the basis of two levels of resupply. A major resupply cycle uses an efficient transshipment algorithm to decide how supplies flow in a structured node-arc network from surplus nodes to deficit nodes. In a minor resupply cycle, supplies flow from supply nodes to demanding divisions and airbases to satisfy current needs. When supplies are low, the effectiveness of combat units is degraded and air sorties are reduced.

The model provides the user with the option of either adding new forces or changing force parameters during the game. These values are processed by subroutine TIMET during the appropriate cycles.

1.3 Equipment Environment

The TACWAR model was originally designed to operate on the CDC 6400 by the Institute for Defense Analyses and has been converted by Computer Sciences Corporation to operate on the HIS 6080. The model requires a card reader, a remote terminal (when operating via remote job entry), a disk drive, and a printer as peripheral equipment. The program requires 80K words of core for execution and 78K words of core for compilation of the largest routine. The user may update previously constructed data base files and initiate execution of TACWAR by remote terminal. Output reports generated by TACWAR are printed on the central printer and/or a remote printer.

1.4 Programming Conventions

TACWAR is written in the FORTRAN 6000 language and consists of approximately 50,000 lines of code. Approximately .22 hours of CPU time and .06 hours of I/O time are required to process a typical scenario for six cycles with no detailed output reports, only two sets of summary output reports, and no time-t inputs. Each subroutine and variable name contained in the model is mnemonically related to its definition or its use in the program.

SECTION 2. SYSTEM DESCRIPTION

This section consists of a general and a detailed description of the TACWAR model. The general description provides an overview of the processing functions of the model; the detailed description explains the program logic of each subroutine of the TACWAR model. Appendix A to this manual contains flowcharts for all routines described, with the exception of block data routines. Appendixes B and C provide information on the source code for the TACWAR model and its support programs.

2.1 General Description

The TACWAR model is a fully-automated combat simulation that can be used to assess the interactions of combat forces employing conventional, nuclear and chemical weapons in a theater-wide campaign. Duration of the war game is set by the user and is measured in fixed 12-hour cycles. The program incorporates facilities that enable the user to model a specific geographical structure for the theater. This structure is then used as the foundation for seven simulations: target acquisition, air combat, nuclear combat, chemical combat, ground combat, theater control, and supplies transportation. Output reports generated by TACWAR present two levels of statistical detail: summary game reports and detailed game reports.

Figure 1 illustrates, through a macroflowchart, the operation of the model. Table 1 gives the maximum values for the index variables that define the structure of the game. The principal features of the TACWAR model are more fully described in the following subsections.

2.1.1 Theater Structure. The TACWAR model is designed to be a theater-level combat model that can simulate the delivery of conventional, nuclear, and chemical munitions by both air and ground means anywhere in the theater. The components of the theater structure include sectors, battle areas, regions, and the communications zone (COMMZ), shown in figure 2. Although figure 2 shows only the Blue side of the theater, the Red side is structured similarly.

2.1.1.1 Sectors. The theater structure is built around a series of eight nonintersecting geographical sectors that cover the theater area of interest. These sectors are

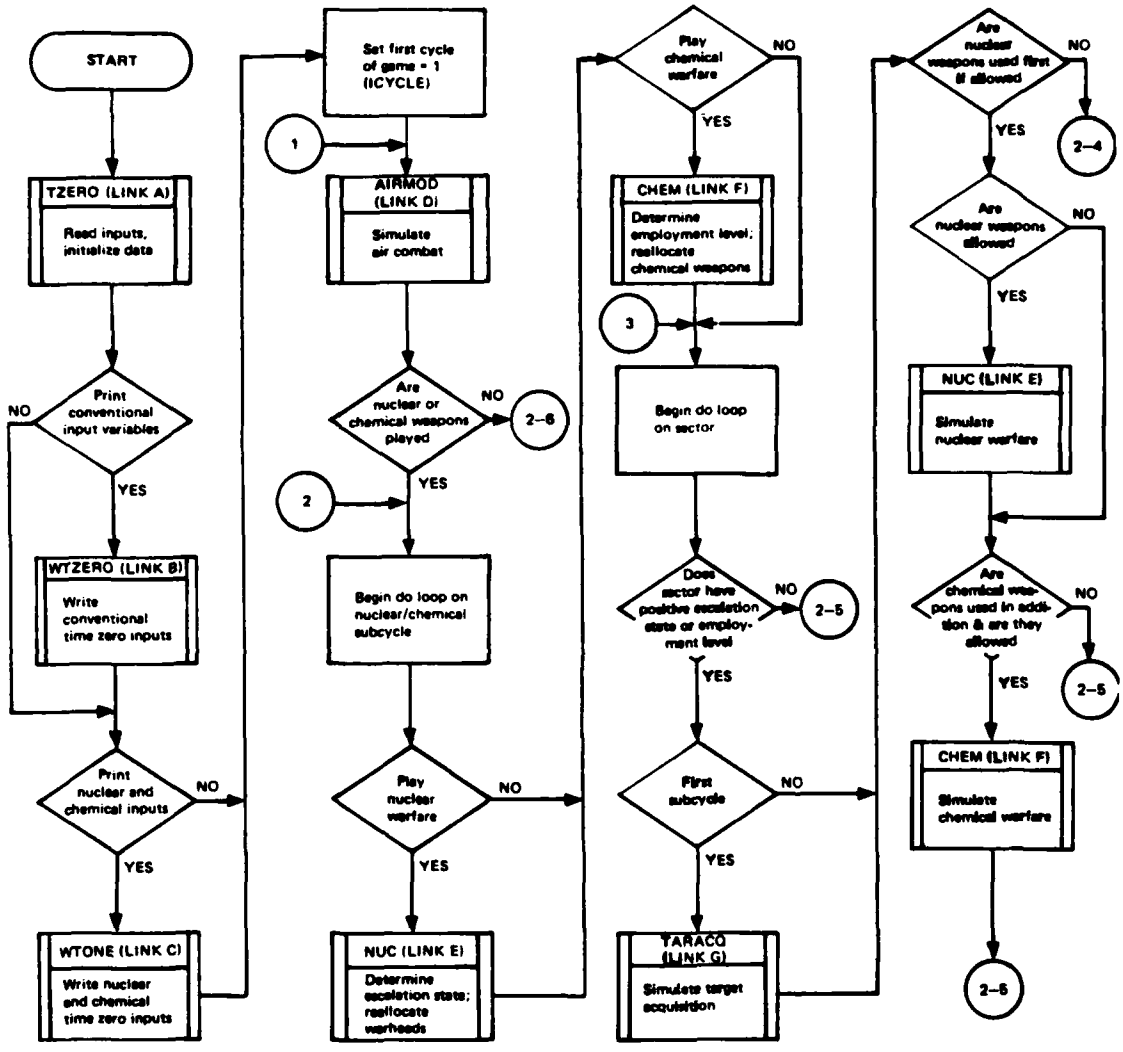


Figure 1. TACWAR Macroflowchart (Part 1 of 2)

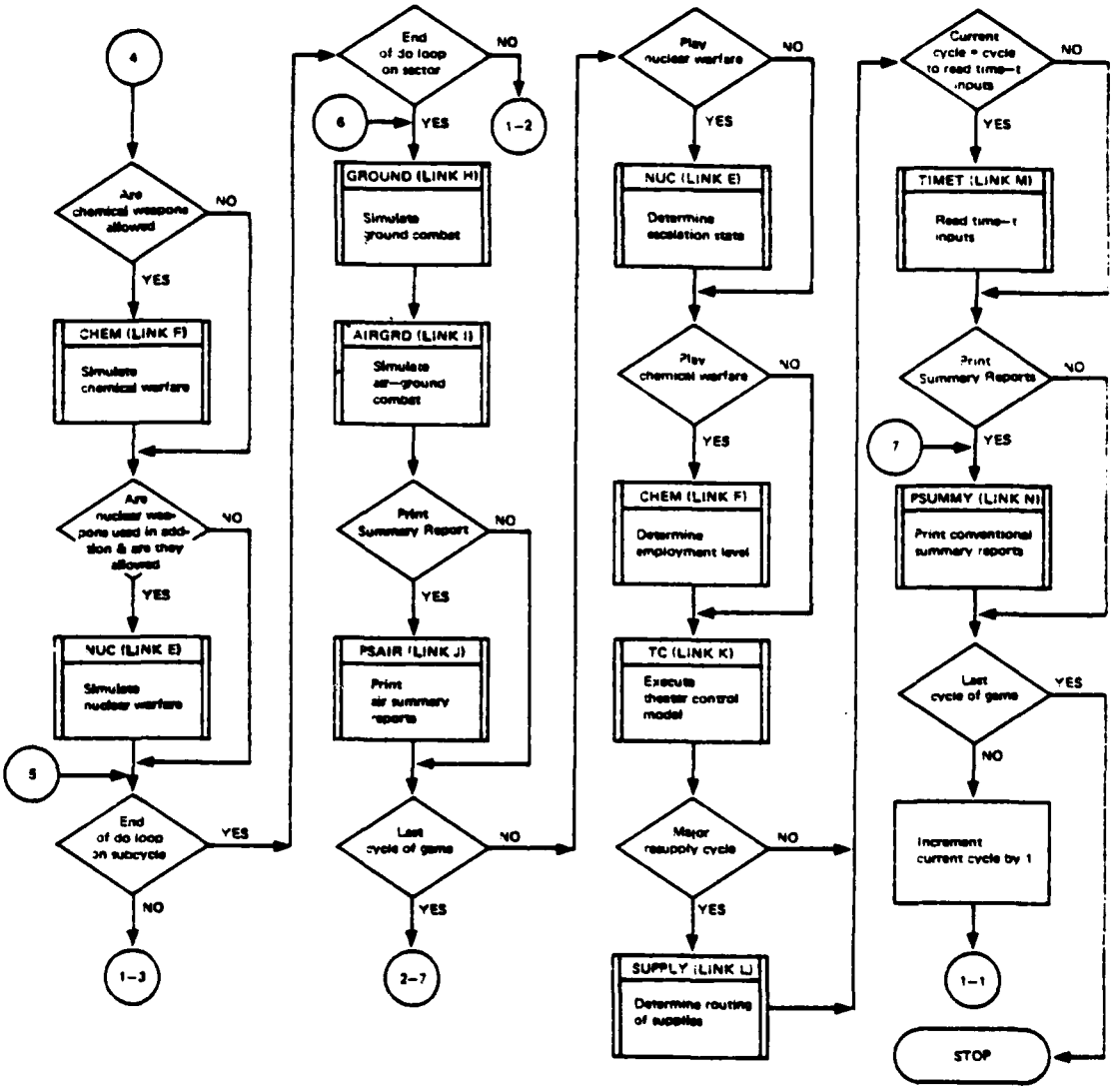


Figure 1. (Part 2 of 2)

Table 1. Maximum Values for TACWAR Index Limits (Part 1 of 2)

Index Limit	Index Definition	Maximum Value
MAD(L)	Maximum number of additional divisions for side L (i.e., number of follow-on divisions in addition to ND(L))	**
NAAC(L)	Number of army air carrier types for side L	3
NAB	Number of air bases	201*
NAC(L)	Number of aircraft types for side L	7
NAFS(L)	Number of air force sensor types for side L	4
NAM(L)	Number of air munition types for side L	6
MAS(L)		4
NBA	Number of battle areas	112*
NBNLT	Number of boundary longitude points	7
NCHDW(L)	Number of division chemical systems for side L	4
NCHSW(L)	Number of sector chemical systems for side L	4
NCHTW(L)	Number of theater chemical systems for side L	4
ND(L)	Number of divisions for side L at time-zero	**
NDVNW(L)	Number of division nuclear systems for side L	4
NEML	Number of chemical employment levels	4
NESC	Number of nuclear escalation states	4

Table 1. (Part 2 of 2)

Index Limit	Index Definition	Maximum Value
NGS(L)	Number of types of ground sensors for side L	5
NINTS	Number of intervals	18
NNSC	Number of nuclear subcycles per conventional cycle	3
NR(L)	Number of regions for side L	3
NS	Number of sectors	8*
NSCNW(L)	Number of sector nuclear systems for side L	5
NSN	Number of supply nodes	95*
NSU(L)	Number of subunit types for side L	7
NSUB	Number of target subtypes	4*
NT(L)	Number of type divisions for side L	***
NTHNW(L)	Number of theater nuclear systems for side L	5
NTR	Number of tactical roles	2*
NW(L)	Number of division weapon types for side L	10
NZ(L)	Number of zones per division for side L	4

*The maximum value must be played. The model is not programmed to accept a smaller value.

**The sum of ND(1) + MAD(1) + ND(2) + MAD(2) must be ≤ 140.

***The sum of NT(1) + NT(2) must be ≤ 10.

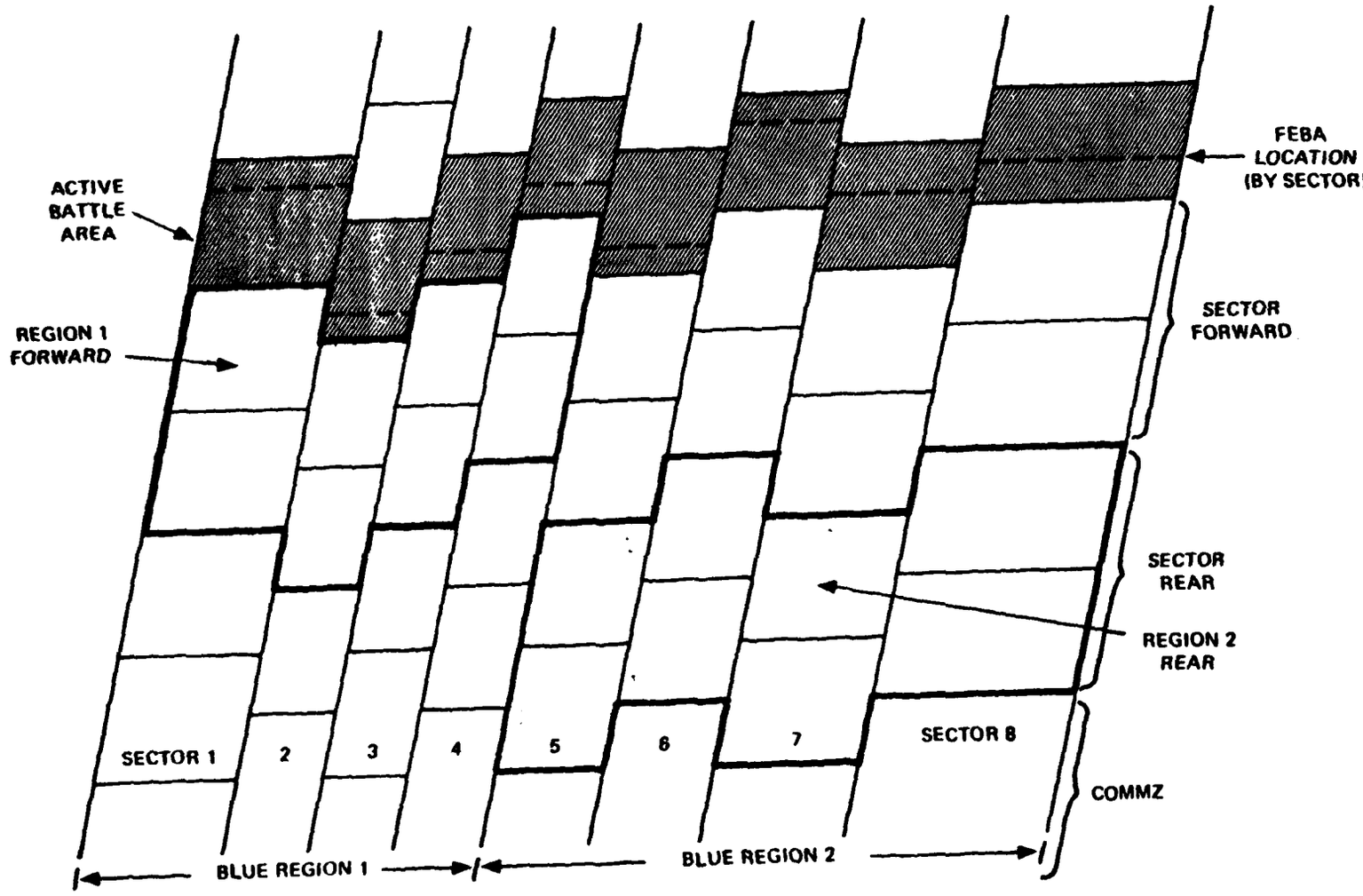


Figure 2. TACWAR Theater Structure (Blue Side)

considered to be avenues of advance that run the length of the theater and are of variable width. Each sector boundary is defined by up to seven points which serve as break points for the sector. The points are selected such that the longitudes of the points are at regular, evenly spaced intervals with the longitude of corresponding break points of all sectors being the same. The latitudes corresponding to each of the seven longitude values are specified by user input. Thus, the sectors are specified as indicated in figure 3. Note that the latitude P_{ij} marks the northern boundary of sector i and the southern boundary of section i-1 where (i > 1).

The location of the FEBA and of intervals and battle areas within a sector are indicated by the distance through the sector to the element of interest. Distances through each of the sectors are measured from the base point on the Red side to the base point on the Blue side along a sequence of line segments (joining the center of the sectors at their break points) as shown in figure 4. The distance to any element in the sector is determined as the distance through the sector to the intersection of the element with the sequence of line segments described above. Sector widths are determined at a given point by calculating the length of the line segment that joins the sector boundaries and corresponds to the longitude line which passes through the given point.

Each sector is divided into a stated number of intervals that are used to portray types of terrain and selective defensive barriers. These intervals are located by indicating the cumulative distances from the Red base point along the sequence of line segments, illustrated in figure 4, to the near edge of the interval and to the far edge of the interval.

2.1.1.2 Battle Areas. Battle areas are additional subdivisions of the battlefield (on a sector-by-sector basis) that provide for easy location of elements played in the model. Battle areas are located in the sector by specifying the ground distance from the Red base point to the leading (i.e., far) edge of the battle area. This ground distance is measured along a series of line segments, the characteristics of which are illustrated in figure 4.

In addition to its location in a sector, each battle area has an associated number to uniquely identify it. This number (N) is determined by consecutively numbering the

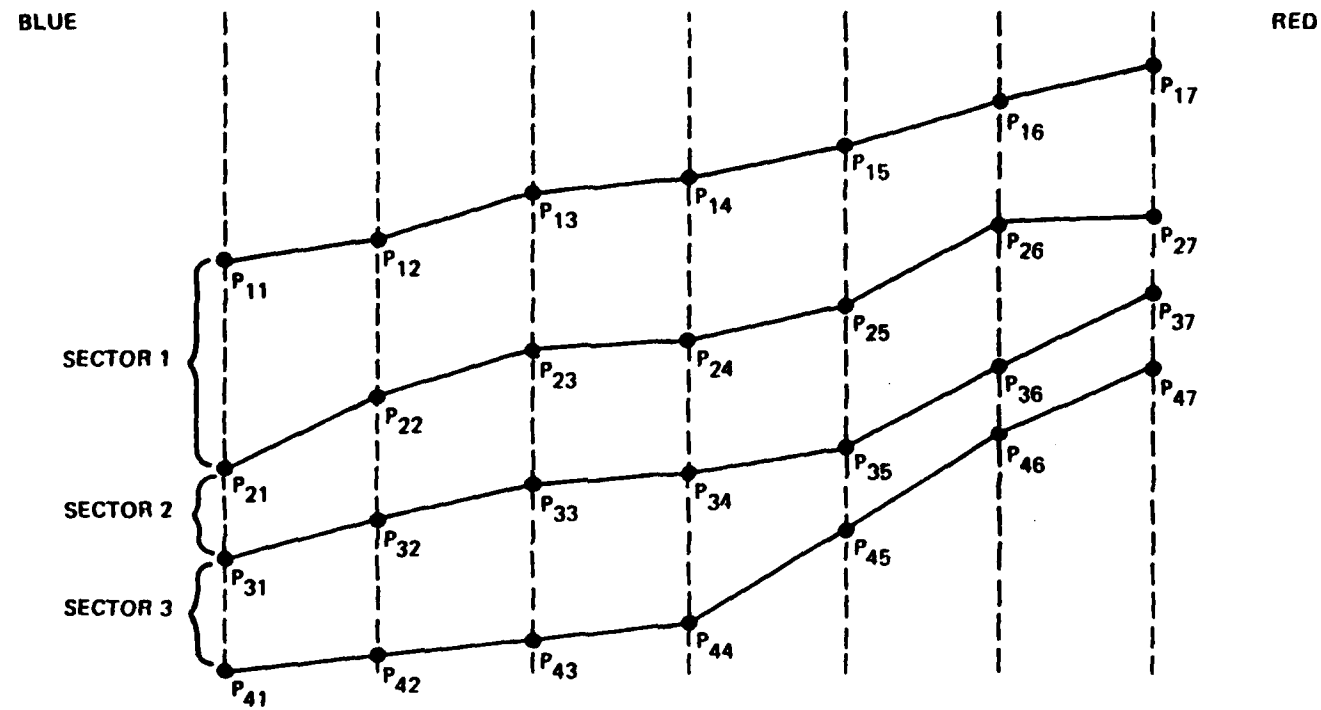


Figure 3. Sector Boundaries

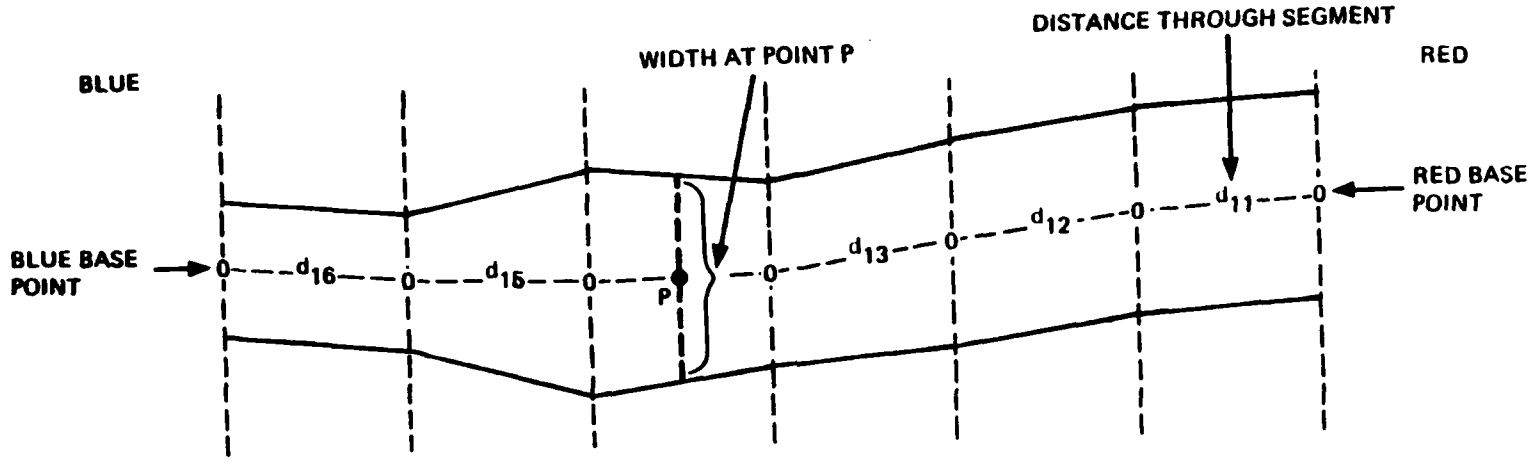


Figure 4. Distances and Widths Through a Sector

battle areas going across the theater from the first sector to the last. Thus

$$N = S + rn$$

where

S = the number of the sector in which the battle area is located

r = the rank of the battle area within the sector

n = the total number of sectors

For example, in an eight sector theater, sector 1 would contain battle areas 1, 9, 17, 25, etc., and sector 5 would contain battle areas 5, 13, 21, 29, etc.

Daily combat activity creates movement of the FEBA in a forward or rearward direction within a battle area and hence within a sector. Battle areas within which ground combat occurs are termed active battle areas, while all others are termed inactive battle areas. Units directly involved in ground combat are assumed to be located in the active battle area, while units not involved in combat are located in one of the inactive battle areas within the theater. Battle areas are also used to locate the following model elements:

a. Tactical airbases by battle area, so that airbases overrun by advancing enemy forces can be accounted for

b. Supply depots, for use by the supplies model

c. Garrison and reserve positions of combat units

d. Fixed combat elements (e.g., surface-to-surface missile (SSM) and surface-to-air missile (SAM) sites)

e. Combat divisions, as they move through the theater to the front, at the rate of a given number of battle areas per cycle.

2.1.1.3 Regions. Regions consist of the rear portions of one or more geographical sectors beginning at the rear of the active battle area and extending to a predefined depth. This grouping of sectors need not be the same for both sides. Each region is divided into two parts, i.e., a forward region and a rear region.

The depths of the forward and rear regions are user-controlled via input and are specified in terms of an integral number of battle areas. The depth of the forward region is measured from the rear boundary of the active battle area back to the depth specified by input. The depth of the rear region is measured from the rear boundary of the forward region to its proper depth as specified by input. The depth of each region stays constant (as user-provided) until one side advances to the point where existing land area precludes these defined depths. As the one side advances, the COMMZ of the opposing side shrinks to one battle area. As further advance is made, the rear region shrinks to one battle area and then the forward region to one. Next, the rear region collapses into the COMMZ. Finally, a point is reached where there is insufficient land area to consider that a COMMZ still exists, at this point, only a forward region remains. In a similar manner, each sector is divided into two parts (corresponding to the region subdivisions) and referred to as sector forward and sector rear.

2.1.1.4 COMMZ. There is one COMMZ for each opponent in the theater. The COMMZ is an area to the rear of the rear region of each sector and spans all sectors in the theater. The COMMZ is used for receiving the arriving combat units, tactical aircraft, supplies, and replacement weapons and personnel. The COMMZ serves as a holding area for combat reserves and provides airbase facilities for long-range tactical aircraft.

2.1.1.5 Summary of Structure Functions. The theater structure of the TACWAR model provides for sectors, battle areas, regions, and a COMMZ. Any element used in the model can be located by its association with a specific (active or inactive) battle area. All other components of the theater structure are used as controlling mechanisms to assign various resources to particular areas of the theater. Sectors are used as the mechanism for assigning units to combat and for subsequent movement of forces. Regions are used for assigning both tactical aircraft to missions throughout the theater and combat forces to sectors. The COMMZ is used for the central control and storage of all resources entering the theater.

2.1.2 Supplies Transportation Network. As described in the previous subsection, each combat sector is partitioned into battle areas. A supplies transportation network is superimposed on the battle areas and consists of 95 nodes with arcs joining adjacent nodes. The nodes represent entities such as ports, supply depots, and transportation centers; the arcs represent transportation links between the nodes. Associated with each arc is the length of the actual route(s) it represents.

The purpose of the supplies network is to provide a gross means of representing the transportation network that exists for shipping supplies within the theater--from sources of supplies to their users. The users of supplies are active and reserve forces in the field and at tactical airbases. The sources of supplies are ports where supplies are stockpiled for future use.

2.1.2.1 Design of the Network. Figure 5 is a schematic representation of a portion of the theater and the supplies network. The numbered dots represent the supply nodes, while the lines joining them are the arcs. For simplicity the battle areas are shown as rectangles in the figure and a particular FEBA location is indicated. (Though a similar supplies network exists for the Red side, it has been omitted from the figure.) Since there are usually more battle areas than supply nodes, some battle areas contain no supply nodes. However, others may contain two or three nodes. Nevertheless, every battle area is assigned to a supply node in such a way that forces in the field and airbases are always connected to the supply network.

If the FEBA advances in Red's favor in a sector, Blue will lose any supply node which is overrun by Red units or which falls within a user specified distance DFASN (as specified for Blue) of the new FEBA location. If a Blue supply node is overrun, it will be added to Red's network provided it is at a distance of at least DFASN (as specified for Red) from the FEBA. A supply node lost by one side but not taken over by the other side is considered to have no owner until one side can legitimately claim it. For example, in figure 5, supply node 83 belongs to neither side because it is too close to the FEBA. Whenever a supply node belongs to one side, then all arcs joining that node to adjacent nodes which also belong to that side are in the same network.

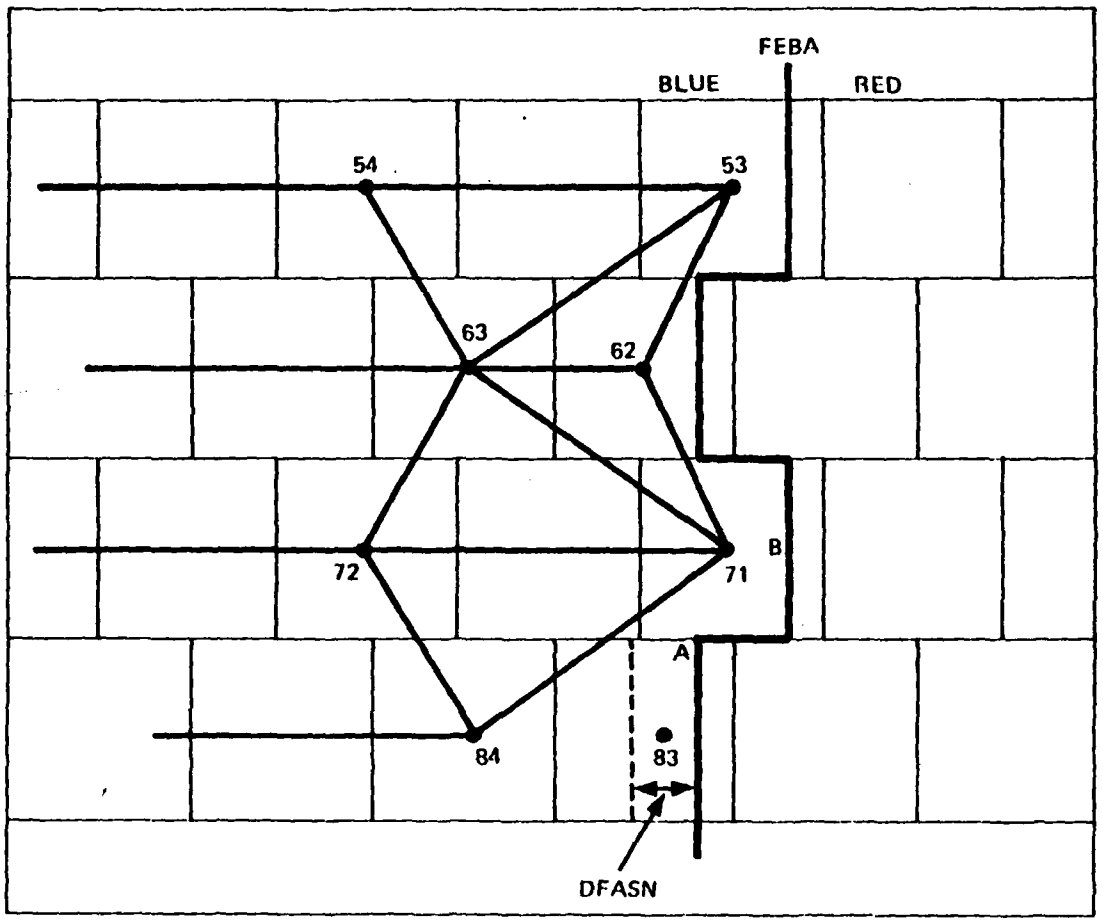


Figure 5. The Supplies Transportation Network

Each battle area owned by Blue must be associated with the Blue supplies network so that the Blue forces within the battle area can be supplied by the logistics system. A rule for assigning battle areas to supply nodes is that a battle area is assigned to the supply node in its sector, which is: (1) in the same battle area, if such a node exists; or (2) closest in a rearward direction, if no node of the first type exists. If the FEBA crosses or comes within a distance of DFASN of a supply node, all battle areas assigned to that node are assigned to the next rearward supply node in the network. Thus, all forces in a battle area are always supplied, either from a supply node within the battle area or the closest node to the rear. In figure 5, for example, battle area A would be assigned to node 84, while battle area B would be assigned to node 71.

The distribution network, which runs from supply nodes to the ultimate users of supplies (the divisions and the airbases), is not modeled explicitly. Rather, barring their destruction, it is assumed that supplies available at supply nodes will be distributed successfully to the users in the field. The reason for omitting this last stage in the distribution process is that the network over which the supplies travel is very complex and contains many arcs. In some instances the supplies could be envisioned traveling cross-country, in which case the number of arcs would be immense. Inherent in this assumption is that enough vehicles are available to carry the required supplies without undue delay.

Each division in the theater is supplied by that node which serves the battle area in which the division is located. The forward airbase in a sector is supplied by that node serving the last battle area in the sector forward. The rear airbase is similarly supplied by the node serving the last battle area in the sector rear. In addition to these direct sources of supplies, divisions and airbases are served by other supply nodes called stockage points.

For each sector there are three stockage points, each of which serves any division or airbase located in an area forward of it. The first stockage point, which is that supply node assigned to the first battle area in the sector forward, stocks supplies for divisions in the active battle area. The second stockage point, which is the supply node assigned to the first battle area in the sector rear, stocks supplies for divisions in the active battle area and the sector forward, and for aircraft at the forward airbase. The third stockage

point, which is that supply node assigned to the first battle area in the COMMZ, stocks supplies for all divisions and airbases forward of the COMMZ.

2.1.2.2 Construction of an Actual Network. Although the user may construct his own supply network, it would involve inputting via cards the location of the supply nodes and the modification of the Data statements which describe the lengths of arcs between the nodes. The actual network constructed for the TACWAR model, and partially defined by the above mentioned Data statements, is now described. The supply nodes are first located by battle area. Then, for use in resupply, the battle areas are assigned to specific supply nodes. Finally, the position of each supply node (as measured linearly along the sector) is given. The latter quantity allows the model to know when the FEBA overruns or comes within the distance DFASN of a supply node. Two notional nodes are located in the Red and Blue COMMZs and indexed 1 and 95, respectively. The remaining nodes in the theater are indexed consecutively, starting in each sector from the Red COMMZ and going toward the Blue COMMZ and advancing through the theater from sector 1 to sector 8. For example, supply nodes 2 through 13 might be located in sector 1 and nodes 14 through 27 in sector 2.

An additional feature of the two notional nodes located in the COMMZs is that they represent a last resort source of supply for any battle area or airbase when all other supply nodes have been overrun. Thus, a unit will not be unsupplied just because the supply network specified does not contain a supply node in every battle area. The number of supply nodes in battle areas near the ends of the theater has been minimized, since supply nodes increase computer running time and it is doubtful whether the model will be used to the point where the FEBA reaches these extreme edges of the theater.

In addition, the lengths of the arcs that connect the nodes of the supplies network are required to construct the network. The TACWAR model selects a 95 node network which covers an eight sector theater structure stretching from western France to eastern Poland. The distances, which are input to the model through Data statements, are taken from highway maps of the countries considered. Each represents the road distance of the most direct road link between two nodes--assuming that the road distance is a reasonable measure of the distance involved in using the different available modes of transportation. As mentioned earlier, the arcs of the network are

assumed to be unconstrained--probably a reasonable assumption, since the European transportation network is very dense and has a high capacity relative to the demands likely to be placed on it by military requirements.

2.1.2.3 Usage of the Network. When discussing the distribution of supplies from sources to users, two types of supply cycles are distinguished--major and minor. In a major cycle, supplies are shipped among the nodes of the supplies network; in a minor cycle, the divisions and airbases are resupplied from the supplies available at their assigned supply nodes.

A minor cycle is of 12-hour duration and an integral number of minor cycles make up each major cycle. In a minor cycle, the users of supplies (divisions and airbases) attempt to stock up to desired inventory levels by ordering supplies from the supply node to which they are assigned. If enough stock is available at that supply node, the order is filled; otherwise, the order is only partially filled, since available supplies are distributed to users in proportion to demand. No stock is shipped from other supply nodes.

In a major cycle, supply stocks are permitted to be shifted among supply nodes, so as to minimize the total number of ton-miles shipped. Since the algorithm used to make this shift is relatively time-consuming, its use is restricted to once each major (rather than minor) cycle.

The theater control model simulates distribution of supplies during minor cycles and the supplies model simulates the shipment of supplies between nodes in a major cycle.

2.1.3 Resources. Each side can be armed with a variety of combat resources for the purpose of simulation. These resources are of four types: ground, air, nuclear, and chemical.

2.1.3.1 Ground Resources. Only one type of personnel is played by TACWAR. The index NW(L) in table 1 gives the maximum number of divisional weapon types that may be played by each side. There are no restrictions on weapon types except that nuclear and chemical weapons are accounted for separately in TACWAR. Because of the way certain sections of the model are programmed, the highest numbered weapon type on either side must be surface-to-air missiles, and any second ground-to-air weapon played, e.g., antiaircraft artillery (AAA), must always be input as the next-highest-numbered weapon.

The maximum number of actual divisions played by side L is given by the sum of the number at time-zero, ND(L), and the maximum number of add-on divisions, MAD(L). The total number of actual Blue and Red divisions played cannot exceed 140. Even though the TACWAR model maintains bookkeeping on individual divisions, the maximum number of division types for a side is fixed by the index NT(L). The total number of Blue and Red division types played cannot exceed 10. Each division type is associated with a TOE (table of organization and equipment) which defines the level of personnel, weapons by type, and subunits considered desirable for the division type. Divisions may be located in an active battle area, in a region, or in the COMMZ. Divisions in the active battle area are considered to be engaged in combat and, thus, to be able to cause and suffer casualties. Divisions in regions or COMMZs are considered to be in reserve and so are unable to cause casualties. Divisions in the first inactive battle area, however, may suffer casualties due to attacks by enemy aircraft and all divisions in the theater may suffer casualties due to nuclear or chemical attack. Replacement pools of weapons and personnel are maintained in the COMMZ of each side. There are also weapon repair pools in each COMMZ for damaged but repairable weapons. Repaired weapons are then transferred to replacement pools for distribution to divisions.

Only one supply type is played, but that type may be characterized as general supplies to include ammunition, food, fuel for ground forces, and fuel for aircraft. (This class does not include nuclear or chemical warheads, which are handled

separately). Supplies are measured in tons, and each combat division and each aircraft by type consumes a certain amount of supplies per time period. The consumption rate for aircraft is by sortie.

2.1.3.2 Air Resources. Each side may play several types of aircraft as designated by the index NAC(L). The Quick Reaction Alert (QRA) aircraft are accounted for separately because they are not assigned to conventional air missions but are reserved for nuclear combat.

In assessing conventional air combat, the TACWAR model considers that for each side two notional airbases exist per combat sector, corresponding to forward and rear sector airbases. In addition, each side's COMMZ is assumed to contain a notional airbase. Each notional airbase is an aggregation of one or more actual airbases located in its area. The actual airbases are described in detail by the airbase data file IAD. This detailed data is also used to assess the effect of nuclear and chemical strikes on specific elements of the actual airbases.

Notional airbases within each sector automatically become targets for airbase attack and areas for airbase defense assessments. Aircraft at each notional airbase may or may not be sheltered depending on the number of shelters at the airbase and the user-specified sheltering priority scheme for aircraft by type. QRA aircraft have high priority for sheltering. When actual airbases are overrun, the resources at the airbases (e.g., shelters and other fixed installations) are lost by the side that previously owned them. All airbases that are located in the active battle area are assumed to be abandoned and are not owned by either side. However, any airbase that is located in a side's territory is assumed to be taken over by that side.

Several types of air munitions may be played by each side, as given by the index NAM(L). Air munitions are not accounted for explicitly (i.e., the air model does not keep track of a stockpile for each type of munition to determine when that type of munition is exhausted). Notional air munition loads carried by each type of aircraft are adjusted according to the distance that the aircraft must fly on a particular mission.

2.1.3.3 Target Acquisition Resources. Three types of sensor resources are used to acquire candidate targets for possible nuclear and/or chemical fire, namely air force sensors,

army-air sensors, and ground sensors. The indexes NAFS(L), NAS(L), and NGS(L) give the maximum number of these types played by each side. Some of the sensors (e.g., ground sensors) are used within a sensing division, while others (e.g., air-carried sensors) are associated with aircraft stationed at fixed geographical positions and are allocated to different target divisions by sensor allocation rules.

2.1.3.4 Nuclear Resources. Nuclear weapon systems fall into three categories: division weapon systems, sector systems, and theater systems. Division systems are located in the active battle area, sector systems in the sector forward or rear and the theater system in the COMMZ. Each system consists of a delivery vehicle and a warhead. The defining characteristics of the nuclear system are the following: (1) the number of division, sector, and theater nuclear weapon system types; (2) the set-back distance from the FEBA for deployment of each weapon system; (3) the range of each system; (4) the number of different yields available for each system; (5) the size of each yield (in kilotons) available to each system; and (6) the weapon delivery error for each system.

2.1.3.5 Chemical Resources. Chemical weapon systems also fall into three categories: division weapon systems, sector systems, and theater systems. A chemical system consists of a delivery vehicle, a dissemination mode, and an agent. The defining characteristics of chemical systems are similar to those listed above for nuclear systems. They are (1) the number of division, sector, and theater chemical weapon system types; (2) the set-back distance from the FEBA for deployment of each weapon system; (3) the range of each system; (4) the number of different dissemination modes available to each system; (5) the type and weight of each chemical agent for each chemical weapon system by specific dissemination mode; and (6) the weapon delivery error for each system.

2.1.4 Air Combat Simulation. The air combat model performs air combat calculations once each combat cycle. This model begins with the aggregation and allocation of air resources for both sides. Air resources are aggregated into notional airbases--a forward and rear notionalized airbase in each sector and a notionalized COMMZ base. The number of aircraft on each notionalized base is calculated utilizing user-specified weighting factors. These factors influence the subsequent assignment of airbase attack missions. The resources of the notional sector airbases in each region are combined into total resources for the region.

Mission assignments are made for all aircraft as a function of the aircraft resources in each region and in the COMMZ as well as the input values of fractional assignments for each mission. The number of aircraft available at each notionalized base is degraded to account for destruction while the sortie rate is degraded to account for the lack of supplies and the loss of airbase operating capability in the previous combat cycle. The number of sorties of each mission type for a cycle, by each aircraft type, is calculated using the mission assignments, the degraded numbers of aircraft at each notionalized base, and the degraded 12-hour sortie rate. In addition, air munition load factors are calculated for CAS missions and SAM defenses are initialized.

When the above calculations have been completed, attrition routines are called to perform air combat calculations. Table 2 summarizes these attrition calculations. This table lists attacking and defending variable names and the section of the subroutine making the calculations. The sections are executed sequentially. After all sections have been executed for side 1 attackers vs. side 2 defenders, they are executed again for side 2 attackers vs. side 1 defenders. The table shows, for example, that type IAC aircraft attacking enemy region forward bases ABAAFA(IAC) are engaged by defenses in the following sequence: SAM and AAA (PSRSCA) defending combat units, aircraft defending combat units (CASDA), medium-range belt SAMs in the forward area (BMRSA), aircraft defending region forward airbases (ABADFA), and SAM point defenses (PSRSFA) for region forward airbases. Those defenses are degraded by escort (ABAEFA) and suppression (ABASFA) aircraft accompanying the attack aircraft. Attrition calculations are made using a single-engagement binomial attrition function. After each interaction, the numbers of attacker and defender sorties alive and continuing on their mission, aborting damaged, killed, suppressed, and aborting undamaged is updated to reflect the results of that interaction.

Table 2. Air Model Interactions Between Attackers and Defenders

AREA DEFENDED	SIDE K DEFENDER	AREA ATTACKED / SIDE L ATTACKER	BATTLEFIELD COMBAT AIR SUPPORT			FORWARD INTERDICTION				FORWARD AIRBASES			REAR AIRBASES			COMMZ AIRBASES		
			Escort CASEA(IAC)	Suppression CASSA(IAC)	Attack CASAA(IAC)	Belt-SAM Suppression BSSUPA(IAC)	Escort INTDEA(IAC)	Suppression INTDSA(IAC)	Attack INTDAA(IAC)	Escort ABAEFA(IAC)	Suppression ABASFA(IAC)	Attack ABAFA(IAC)	Escort ABAERA(IAC)	Suppression ABASRA(IAC)	Attack ABAARA(IAC)	Escort ABAEZA(IAC)	Suppression ABASZA(IAC)	Attack ABAAZA(IAC)
BATTLEFIELD COMBAT AIR SUPPORT	defending aircraft CASDA(KAC)		AOVL1 100	ATTR5 4800	ATTR5 4800	AOVL1 300	AOVL1 500	AOVL1 600	AOVL1 600	AOVL1 500	AOVL1 600	AOVL1 600	AOVL1 500	AOVL1 600	AOVL1 600	AOVL1 500	AOVL1 600	AOVL1 600
	short-range SAMs defending units in combat in sector IST PSRSCA(ISS,IST)			ATTR5 5000	ATTR5 5200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200	AOVL1 200
REGION FORWARD	medium-range belt SAMs BMRSA(IMS)					AOVL1 400	ATTR1 800	ATTR1 700	ATTR1 800	ATTR1 800	ATTR1 700	ATTR1 800	ATTR1 800	ATTR1 700	ATTR1 800	ATTR1 800	ATTR1 700	ATTR1 800
	defending aircraft ABADPA(KAC)						ATTR1 900	ATTR1 1000	ATTR1 1000	ATTR1 900	ATTR1 1000	ATTR1 1000	ATTR1 900	ATTR1 1000	ATTR1 1000	ATTR1 900	ATTR1 1000	ATTR1 1000
	short-range SAMs defending forward airbases PSRSFA(ISS)										ATTR1 1600	ATTR1 1800						
	short-range SAMs defending interdiction targets PSRSIA(ISS,IST)							ATTR2 2100	ATTR2 2300									
REGION REAR	defending aircraft ABADRA(KAC)												ATTR3 2500	ATTR3 2600	ATTR3 2600	ATTR3 2500	ATTR3 2600	ATTR3 2600
	long-range area SAMs ALRSRA(ISS)													ATTR3 2700	ATTR3 2900	ATTR3 2900	ATTR3 2700	ATTR3 2900
	short-range SAMs defending rear airbases PSRSRA(ISS)													ATTR3 3200	ATTR3 3400			
COMMZ	defending aircraft ABADZA(KAC)															ATTR4 3600	ATTR4 3700	ATTR4 3700
	long-range area SAMs ALRSZA(ISS)																ATTR4 3800	ATTR4 4000
	short-range SAMs defending airbases PSRSZA(ISS)																ATTR4 4300	ATTR4 4500

The number of attacker sorties delivering ordnance on their targets is set equal to the number of sorties alive and continuing on their mission after engaging the point defenses protecting their targets. After the results of the engagements listed in table 2 have been calculated, the attrition of aircraft returning home is set equal to a user-input fraction of inbound attrition. Finally, the aircraft sortie attrition results are converted to numbers of aircraft undamaged, damaged, and killed in this cycle on each side from each actual airbase.

The air-ground simulation, although part of the air model logic assessments, is performed after the nuclear and chemical routines in order that attack aircraft initially set aside for either nuclear or chemical employment, but not used, can be reallocated to one of the primary conventional attack roles of CAS, ABA, or INT. In the air-ground simulation, aircraft are assigned to shelters at each notional airbase, according to user-specified priorities. Attrition to aircraft on the ground and to shelters from ABA missions, and to reserve divisions, SSM sites and supplies from INT missions is assessed. The inventories of QRA aircraft at each notional airbase are adjusted in order to maintain a user-specified minimum number of aircraft assigned to the QRA role. Repair of damaged aircraft and SAMs is included in the air-ground simulation.

2.1.5 Target Acquisition Simulation. The target acquisition simulation is performed for both sides once during each combat cycle in which nuclear or chemical weapons are to be employed. The target acquisition model covers the same time frame as a nuclear/chemical subcycle and the same results are used for each subcycle of the current combat cycle. The purpose of this simulation is to compute, by subunit type, the probability that a subunit in a particular location is detected and to compute the average sensor error and delay time associated with such a detection. The user has the option to bypass the target acquisition model by providing as input the detection probability, sensor error and delay time for each sensor type. Within the target acquisition model, targets in the active battle area are assumed to be subunits located within rectangular divisions. Divisions are subdivided into smaller rectangles, called zones, which are distinguished by their distances from the division front. Target subunits may also be located in reserve divisions in the first inactive battle area. Sensors may be ground sensors or they may be carried on army-air carriers or on air force reconnaissance aircraft. Sensors operate in one of four modes: standoff (fixed or vertical), standoff (moving), penetrating (forward area search), or penetrating (deep area search). Both glimpse and continuously-operating sensors are modeled.

The target acquisition model computes the size and location of the sensing and target divisions within a sector based on division type, combat deployment, posture and sector width. Allocation of ground sensors to target divisions depends on the relative numbers of opposing divisions online. If S/T is the ratio of sensing divisions to target divisions, then exactly S/T of the sensing divisions (and their sensors) are assigned to acquire target subunits within each target division. Army-air carriers and air force reconnaissance aircraft are apportioned into single-sensor groups which are allocated equally among all target divisions within the sector.

For targets in the active battle area, the probability that a subunit is detected is calculated, for each sensor type, as a function of target and sensor characteristics, the distance from the sensor to the target, weather factors (ceiling and visibility) and terrain factors. For targets in the rear, the detection probability depends on the effective swath width at which a sensor will detect a subunit, the velocity of the aircraft, the total search time and the average division area. The detection probabilities for individual sensor

types are combined to give the overall probability that a subunit is detected. The average sensor error and delay times are computed using the individual detection probabilities as weighting factors.

2.1.6 Nuclear Warfare Simulation. The nuclear model of TACWAR consists of subroutines which assess the damage from nuclear munitions delivered against preselected targets by enemy and friendly forces.

First, nuclear escalation states, which govern the use of nuclear weapons against different types of targets are determined. In addition, certain other functions are also performed. These are the determination of the number of weapon systems that deliver nuclear munitions, and the allocation, within each sector, of the stocks of nuclear warheads to weapons in the division, sector, and theater pools. Then nuclear warfare is simulated on a sector-by-sector basis as described below. A list of targets, in order of priority from highest to lowest, is constructed from battlefield targets (by division, subunit, and zone), and from region and COMMZ targets (airbases, missile sites, supply nodes, and divisions in the rear). Also, a single weapon priority list is produced from division, sector, and theater nuclear weapon systems allocated to a given sector. Each weapon system is classified by yield and within each yield class, weapon systems are ordered by weapon system response time, distance from the FEBA, range, and CEP. Next, the expected number of battlefield targets (subunits) detected is determined from the number of each type of subunit in a given zone that is allowed to be targeted. The total number of potential targets is then reduced by the number of targets precluded from targeting by civilian population collateral damage constraints. Weapons are assigned to allowable (battlefield, region, and COMMZ) targets by the selection of a weapon system with the preferred yield from the list of weapons. The actual weapon system that is assigned to a given target is the first one with the desired yield within range of the target. A second choice of yield is considered when a weapon system with the preferred yield is not available.

Finally, the damage inflicted by nuclear weapons is calculated. For each weapon the weapon radius for moderate and severe damage to each target type is calculated. These radii are used to calculate the fraction of targets of each type damaged outside the targeted subunit. The effects of prompt radiation are estimated by updating, for each weapon, the distribution of military personnel that exist within several defined pools of accumulated radiation. Civilian casualties and fatalities from blast and prompt radiation are also calculated for each nuclear weapon.

Except on the last cycle of the game, after the ground and air-ground combat simulations have been performed the nuclear model may be accessed again. On this call, which is made only if nuclear warfare is played, the nuclear escalation states are recomputed for input to the theater control model.

2.1.7 Chemical Warfare Simulation. The chemical model of TACWAR consists of subroutines which assess the impact of chemical munitions delivered against preselected targets by enemy and friendly forces.

First, chemical employment levels, which govern the use of chemical weapons against different types of targets, are determined. In addition, certain other functions are performed. These are the decontamination of equipment that has been contaminated in a previous cycle by a chemical agent and the determination of the number of weapon systems that deliver chemical munitions. In addition, within each sector, the supplies of chemical rounds are allocated to weapons in the division, sector, and theater pools. Then chemical warfare is simulated on a sector-by-sector basis as described below. A list of targets, in order of priority from highest to lowest, is constructed from battlefield targets (by division, subunit, and zone), and from region and COMMZ targets (airbases, missile sites, supply nodes, and divisions in the rear). Also, a single weapon priority list is produced from division, sector, and theater chemical weapon systems allocated to a given sector. Each weapon system is ordered within each category of chemical agent and dissemination mode by increasing weapon system response time, distance from the FEBA, range, and CEP. Next, the expected number of battlefield targets (subunits) detected is determined from the number of each type of subunit in a given zone that is allowed to be targeted. The total number of potential targets is then reduced by the number of targets precluded from targeting by civilian population collateral damage constraints. Weapons are assigned to allowable (battlefield, region, and COMMZ) targets by the selection of a weapon system from the list of weapons with the preferred combintion of chemical agent and dissemination mode. The actual weapon system that is assigned to a given target is the first one with the desired agent and dissemination mode within range to reach the target. Alternative combinations of agent and dissemination mode are tried when a weapon with the preferred choice is not available.

Finally, the damage inflicted by chemical munitions is calculated. Damage calculations can be made for liquid, vapor, or semivolatile agents disseminated by a point source, line source or in a uniform area coverage mode. For each weapon, the number of casualties and the number of fatalities may be calculated. The area covered by a medium incapacitating or lethal dosage of agent is calculated. This dosage is applied

to all personnel in each chemical protection category within each class of physical shielding to determine the number incapacitated and the number killed. Bonus effects outside the targeted subunit are included. The number incapacitated is subtracted from each unit and placed in a pool for some user-specified amount of time. Civilian casualties and fatalities are also calculated. Contaminated equipment is removed from each unit and placed in a pool for some user-specified time.

Except on the last cycle of the game, after the ground and air-ground combat simulations have been performed, the chemical model may be accessed again. On this call, which is made only if chemical warfare is played, the employment levels are recomputed for input to the theater control model.

2.1.8 Ground Combat Simulation. When the current cycle's air combat and, if required, the nuclear and chemical warfare simulations have been completed, the TACWAR ground combat model is processed. This model considers each combat sector in turn, repeating the computations of ground force attrition and FEBA movement until all sectors have been examined.

The model computes the percentages of opposing ground weapons killed by each side. This is accomplished by using the standard allocations previously computed to calculate the adjusted allocations of Red/Blue ground weapons when Blue/Red is on attack (defense). That value and the values of an individual ground weapon against an opponent's ground weapon are used to determine the percentages of each type weapon destroyed. The percentages of air kills to opposing ground weapons is similarly computed using the allocation and value of each side's air munitions against the ground weapons of its opponent.

Values of individual weapons and aircraft sorties are computed by the antipotential potential method. This method is a complex approach to the problem of computing the value of a weapon based on its capability to destroy the value of the enemy's weapons. The antipotential potential technique uses an iterative eigenvector procedure to place both sides' weapons on a common scale of values that is based on the value of a single Blue reference weapon. The procedure yields a constant, derived from the eigenvalue of a weapon-on-weapon kill rate matrix, that is used for ground weapons. The model determines personnel effectiveness on attack and defense, as a function of personnel strength, and supply effectiveness as a function of supplies on hand.

The model next determines which side is the attacker in the sector under consideration. (The theater attacker is designated by the user at the beginning of the game and is changed, if appropriate, by the theater control model.) To determine which side is attacking in a sector, TACWAR computes the sector force ratios, considering each side's total air and ground weapons' value on defense and on attack. Even though one side may have a superior force in a sector, that side will not be designated the sector attacker unless the force ratio equals or exceeds a user-specified threshold. A holding posture may exist if neither side is strong enough to attack.

Since the computation of personnel and weapon attrition depends upon the computation of force ratios, it is obvious that the most important operation performed by the ground model is the determination of each side's weapon values and the force ratios derived from them. The force ratios developed by the model have both air and ground components. For the purpose of computing force ratios, the ground strength of a side consists of the collective value of that side's ground weapons. The air strength of a side is the collective value of the successful CAS sorties that side flies. A standard force ratio is the ratio of the attacker's total air and ground strength to the defender's total air and ground strength.

To calculate the value lost for both sides, force ratios are recomputed as the sum of the value of ground forces and air forces (times the fraction of CAS sorties considered) divided by the value of the opponent's ground and air forces. The model computes the casualty percentages as a function of posture and force ratios. The value lost for each side may then be determined from these percentages. Another important feature of the ground combat routine is the modeling of supply consumption and destruction. The consumption of supplies by ground forces is based on division consumption rates input by the user. Supplies may be destroyed by attacking aircraft on CAS or INT missions, or by ground weapons during the course of battle. Each type of attacking weapon is assumed to destroy some user-set amount of enemy supplies in the combat sector. Each successful CAS sortie is assumed to destroy some fixed amount of supplies in addition to the weapon and personnel losses it causes. Pure supply-interdiction missions may also be modeled.

Since some weapons lost in combat are only damaged and are thus able to be repaired, the ground combat model maintains a pool of recovered and repairable weapons. The user designates the percentage of such weapons that can be repaired each cycle. The model adds such repaired weapons to the COMMZ replacement pool.

2.1.9 Theater Control Simulation. Provided the target acquisition model was used, the TACWAR theater control model first determines the assignment of reconnaissance aircraft and calculates attrition to army-air carriers and reconnaissance aircraft. The remainder of the theater control model is executed regardless of how target acquisition is determined. It determines the width of each combat sector based on the current FEBA location. Preparations for the next cycle's combat are made by the theater control model in several ways. The model computes the total tonnage of supplies consumed by combat units that are located in region areas and in the COMMZ. It then determines the sectors of main attack (if not input by the user) within each region based on opposing effective combat units. As a result of the FEBA movement calculated in the ground combat model, the model updates the location of combat divisions and supply nodes. Then, based on the new FEBA location, the region forward and region rear depths are adjusted, if necessary.

An important function of the theater control model is to compute the number of noncombat weapon losses that arise each cycle and to send the broken weapons to the repair pools. The model conducts the operation of weapon repair and sends repaired weapons to the replacement pools. A major activity of the theater control model is the relocation and upgrading of divisions for effective combat readiness on the next cycle. The model first computes the change in combat mode, if appropriate. Next, it computes a new combat effectiveness for each division based on people, weapons and supplies available in the units. Then the model withdraws from the active battle areas any ineffective divisions and replaces them with divisions of higher effectiveness, if available. For each division in the active battle area, the model computes its demand for replacements of people, weapons by type, and subunits by type. The theater control model then assigns to these divisions the required replacements, provided they are available in the replacement pools. In addition to making these adjustments, the model replaces divisions in the active battle areas with divisions of higher strength, if available, from the first inactive battle area and assigns required replacements, if available, from the replacement pools. Next the model determines which actual airbases must be abandoned because of advancing enemy forces. Then it computes the total supply demands of the tactical airbases and of the combat units in the rear areas, and ships supplies to requesting areas according to available inventories. The

model also computes the effectiveness of all divisions and orders the divisions in the first inactive battle area according to effectiveness. Next, the theater control model moves divisions from rear areas forward by the appropriate movement rate. Finally, if the present cycle is one in which the supply model is executed to distribute supplies between nodes, the theater control model updates the node assignments.

2.1.10 Supplies Transportation Simulation. The TACWAR supplies model is executed every major cycle to simulate the redistribution of supplies among supply nodes. The model utilizes a fast-running transportation algorithm to decide the most efficient way to route supplies through a simplified node-arc network of Central Europe. The transportation problem is to minimize the total cost of shipping supplies from supply nodes within the theater that have adequate supplies to supply nodes that are short of supplies. The algorithm and code were developed by Srinivasan and Thompson at Carnegie-Mellon University (see reference 1) but are now available at the National Bureau of Standards.

Subsection 2.1.2 describes the TACWAR supplies transportation network, which contains a total of 95 supply nodes serving both Blue and Red divisions and airbases. This network structure and the amount of supplies at each of the nodes are the quantities used to formulate the problem as a transshipment problem, which is then converted to a transportation problem as described in reference 2. The primal transportation algorithm known as the MODI model (or the row-column sum method described in reference 3) is used for solving the problem. Standard perturbation procedures are used to prevent the circling of the algorithm. (See reference 3).

Subroutine SUPPLY, the entry routine of the supplies model, determines the surpluses and deficits for each supply node based on the amount of supplies demanded by divisions and airbases served by the node. Then, for one side at a time, subroutine TRANPO is called to determine the optimal routing of supplies from surplus nodes to deficit nodes. The input to TRANPO is the TACWAR network of 95 nodes and their connecting arcs, and the surpluses and deficits for each node with the restriction that supply nodes not belonging to the side for which the routing is being determined have no surpluses or deficits.

Subroutine TRANPO is the driver for the series of subroutines which solve the transshipment problem using the algorithm mentioned above. Subroutine INPUT initializes the supply node data. The initial basic feasible solutuion is found in INSOL (using a variant of the row-minimum rule). LABEL1 sets up the labels corresponding to the list of initial basic cells. LABEL2 determines the values for the dual variables corresponding to the initial basis. The program then iterates, making a number of adjacent basis changes, until an optimal basis is found. MAIN finds a nonbasic cell to pivot

on using the row-minimum rule. CYCLE finds the loop created by the addition of the nonbasic cell (using a modification of the predecessor-index method described in reference 4). Subroutine FIXLIJ modifies the labels to correspond to the new basis and IJFIX modifies the dual variables accordingly. Finally, OUTPUT prints out the optimal primal solution.

When the optimal routing is returned to subroutine SUPPLY, the model then ships supplies according to that routing and determines the arrival times of the supplies. It is assumed that supplies leave surplus nodes immediately and arrive at deficit nodes during some minor cycle which occurs no later than the next major cycle. The theater control model adjusts the supplies at the nodes to reflect the arrival of goods at the deficit nodes.

2.1.11 Remote Terminal Capability. The TACWAR model can be executed from a remote terminal through a JCL file in Time-Share-System (TSS) format. Appendix D describes the procedure for executing TACWAR through such a file and subsection 4.1.3 discusses the procedures for updating the file.

THIS PAGE INTENTIONALLY LEFT BLANK

2.2 Detailed Description

This subsection contains descriptions of all TACWAR subroutines, including the processing functions performed by each routine. Descriptions of the subroutines are presented in their nominal order of execution; that is, the input routines and the time-zero write routines followed by the air, nuclear, chemical, target acquisition, ground, air-ground, air summary, theater control, supply, time-t, and summary output routines.

The TACWAR model consists of a main program, TMAIN, and 136 subroutines written in FORTRAN 6000. The main routine, plus 12 subroutines, serve to call the other 124 subroutines.

The program overlay structure*, as implemented on the HIS 6080, is organized into a root segment, 14 primary overlays, and 19 secondary overlays. The root segment includes TMAIN, blank Common, and the subroutines EIGENV, MPROD, CNTRYC, CVFW, SECWTH, GDIST, TAG, APORTN, and CLR, each of which may be called at frequent intervals during a game. As illustrated by figure 6, the remainder of the program structure is segmented according to function. Table 3 lists each routine and the routines it calls.

The majority of the TACWAR subroutines refer to variables in blank Common. However, the program does employ labeled Common blocks in certain subroutines. Table 4 lists these labeled Common blocks and the routines in which they appear. The functions of each TACWAR routine are described in the following subsections.

* modified, see CSC Letter, Page 4, TACWAR CODE INDUCTION, dtd 31 Jul 78, CCTC T.O. #6311 Subtask 1.

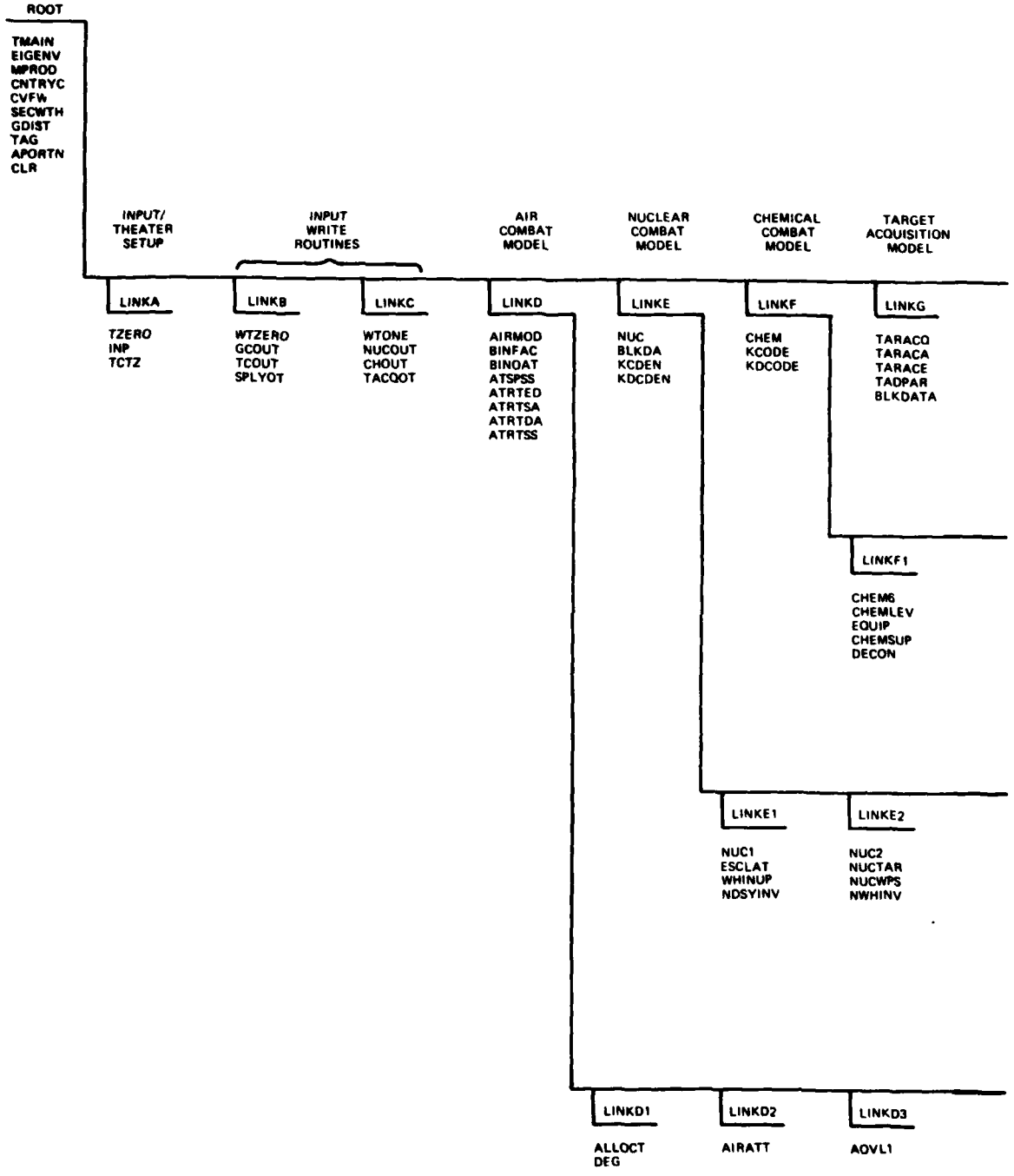


Figure 6. TACWAR Link Overlay Structure (Part 1 of 2)

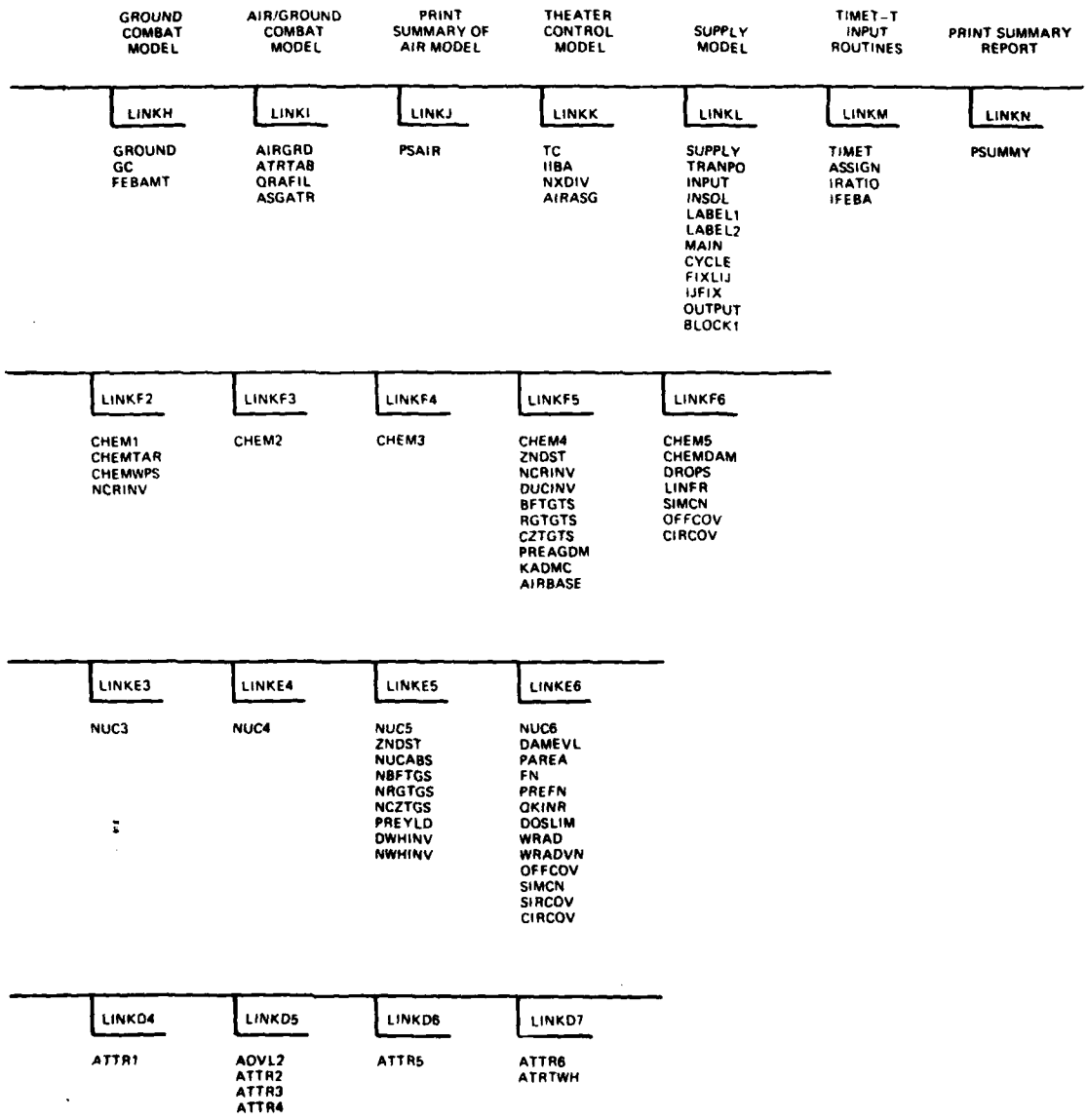


Figure 6. (Part 2 of 2)

THIS PAGE INTENTIONALLY LEFT BLANK

Table 3. TACWAR Program Calling Structure
(Part 1 of 7)

Link Name	Routines in Link	Routines Called
ROOT SEGMENT	TMAIN	TZERO, WTZERO, WTONE, AIRMOD, NUC, CHEM, TARACQ, GROUND, AIRGRD, PSAIR, TC, SUPPLY, TIMET, PSUMMY
	EIGENV	MPROD
	MPROD	
	CNTRYC	
	CVFW	
	SECWTH	GDIST
	GDIST	
	TAG	
	APORTN	CLR, TAG
	CLR	
LINK A	TZERO	INP, TCTZ
	INP	APORTN, TAG
	TCTZ	CVFW, EIGENV, MPROD, SECWTH, CLR
LINK B	WTZERO	GCOUT, TCOUT, SPLYOT
	GCOUT	
	TCOUT	
	SPLYOT	
LINK C	WTONE	CHOUT, NUCOUT, TACQOT
	NUCOUT	

Table 3. (Part 2 of 7)

Link Name	Routines in Link	Routines Called
LINK C	CHOUT	
	TACQOT	
LINK D	AIRMOD	ALLOCT, AIRATT, AOVL1, ATTR1, AOVL2, ATTR5, ATTR6, APORTN
	BINFAC	
	BINOAT	
	ATSPSS	BINOAT
	ATRTED	BINOAT
	ATRTSA	BINOAT
	ATRTDA	BINFAC
	ATRTSS	BINOAT
LINK D1	ALLOCT	DEG
	DEG	CLR, TAG, CVFW
LINK D2	AIRATT	
LINK D3	AOVL1	ATRTDA, ATRTED, ATRTSA, ATRTSS
LINK D4	ATTR1	ATRTDA, ATRTED, ATRTSA, ATRTSS, ATSPSS
LINK D5	AOVL2	ATTR2, ATTR3, ATTR4
	ATTR2	ATRTSA, ATSPSS
	ATTR3	ATRTDA, ATRTED, ATRTSA, ATRTSS, ATSPSS
	ATTR4	ATRTDA, ATRTED, ATRTSA, ATRTSS, ATSPSS

Table 3. (Part 3 of 7)

Link Name	Routines in Link	Routines Called
LINK D6	ATTR5	ATRTDA, ATRTSA, ATSPSS
LINK D7	ATTR6	ATRTWH
	ATRTWH	
LINK E	NUC	NUC1, NUC2, NUC3, NUC4, NUC5, NUC6, CLR
	BLKDA	
	KCDEN	
	KDCDEN	
LINK E1	NUC1	ESCLAT, NDSYINV, WHINUP
	ESCLAT	
	WHINUP	
	NDSYINV	
LINK E2	NUC2	NUCTAR, NUCWPS
	NUCTAR	
	NUCWPS	CLR, KCDEN, KDCDEN, NWHINV
	NWHINV	
LINK E3	NUC3	
LINK E4	NUC4	
LINK E5	NUC5	NBFTGS, NCZTGS, NRGTGS
	ZNDST	
	NUCABS	GDIST, SECWTH
	NBFTGS	PREYLD, ZNDST
	NRGTGS	NUCABS, PREYLD

Table 3. (Part 4 of 7)

Link Name	Routines in Link	Routines Called
LINK E5	NCZTGS	NUCABS, PREYLD
	PREYLD	DWHINV, KDCDEN, NUCABD (ENTRY IN NUCABS), NWHINV, ZNDST
	DWHINV	KDCDEN
	NWHINV	
LINK E6	NUC6	DAMEVL, KDCDEN
	DAMEVL	CVFW, DOSLIM, FN, OFFCOV, PAREA, PREFN, SIRCOV, WRAD, WRADVN
	PAREA	
	FN	
	PREFN	QKINR
	QKINR	
	DOSLIM	
	WRAD	
	WRADVN	
	OFFCOV	CIRCOV
	SIMCN	
	SIRCOV	CIRCOV, SIMCN
	CIRCOV	SIMCN
LINK F	CHEM	CHEM1, CHEM2, CHEM3, CHEM4, CHEM5, CHEM6, CLR
	KCODE	
	KDCODE	

Table 3. (Part 5 of 7)

Link Name	Routines in Link	Routines Called
LINK F1	CHEM6	CHEMLEV, DECON, EQUIP, CHEMSUP
	CHEMLEV	
	EQUIP	
	CHEMSUP	
	DECON	
LINK F2	CHEM1	CHEMTAR, CHEMWPS
	CHEMTAR	
	CHEMWPS	CLR, KCODE, KDCODE, NCRINV
	NCRINV	
LINK F3	CHEM2	
LINK F4	CHEM3	
LINK F5	CHEM4	BFTGTS, RGTGTS, CZTGTS
	ZNDST	
	NCRINV	
	DUCINV	KDCODE
	BFTGTS	DUCINV, KADMC, KDCODE, NCRINV, ZNDST
	RGTGTS	AIRBASE, PREAGDM
	CZTGTS	AIRBASE, PREAGDM
	PREAGDM	AIRDIST (ENTRY IN AIRBASE), CLR, DUCINV, KADMC, KDCODE, NCRINV
	KADMC	DUCINV, KDCODE, NCRINV
	AIRBASE	GDIST, SECWTH

Table 3. (Part 6 of 7)

Link Name	Routines in Link	Routines Called
LINK F6	CHEM5	CHEMDAM, KDCODE
	CHEMDAM	DROPS, LINFR, OFFCOV, SIMCN
	DROPS	
	LINFR	SIMCN
	SIMCN	
	OFFCOV	CIRCOV
	CIRCOV	SIMCN
LINK G	TARACQ	TARACA, TADPAR
	TARACA	TARACE
	TARACE	
	TADPAR	
	BLKDATA	
LINK H	GROUND	GC, FEBAMT
	GC	EIGENV, MPROD, CVFW, CNTRYC
	FEBAMT	CVFW
LINK I	AIRGRD	APORTN, ASGATR, ATRTAB, QRAFIL
	ATRTAB	
	QRAFIL	
	ASGATR	
LINK J	PSAIR	

Table 3. (Part 7 of 7)

Link Name	Routines in Link	Routines Called
LINK K	TC	AIRASG, CNTRYC, CVFW, IIBA, NXDIV, SECWTH
	IIBA	
	NXDIV	
	AIRASG	CLR
LINK L	SUPPLY	TRANPO
	TRANPO	INPUT, INSOL, LABEL1, LABEL2, MAIN, CYCLE, FIXLIJ, IJFIX, OUTPUT
	INPUT	
	INSOL	
	LABEL1	
	LABEL2	
	MAIN	
	CYCLE	
	FIXLIJ	
	IJFIX	
	OUTPUT	
	BLOCK 1	
LINK M	TIMET	ASSIGN, CVFW
	ASSIGN	IFEBA, IRATIO
	IRATIO	
	IFEBA	
LINK N	PSUMMY	

Table 4. TACWAR Labeled Common Blocks (Part 1 of 3)

Common Blocks	Routines in Which They Appear
AFSTF	BLKDA
AFSTF2	NUC, BLKDA, NUC1, ESCLAT, WHINUP, NDSYINV, NUC2, NUCTAR, NUCWPS, NWHINV, NUC3, NUC4, NUC5, NUCABS, NBFTGS, NRGTGS, PREYLD, DWHINV, NUC6, DAMEVL
AL	TACQOT
CC11	AIRMOD, ALLOCT, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
CCC1	AIRMOD, ALLOCT, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
CCC2	AIRMOD, ALLOCT, DEG, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
CCC3	AIRMOD, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
CCC3A	AIRMOD, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
CCC4	AIRMOD, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
CCCC1	AIRMOD, ALLOCT, DEG, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6

Table 4. (Part 2 of 3)

Common Blocks	Routines in Which They Appear
CHA	CHEM, CHEM6, EQUIP, CHEMSUP, DECON, CHEM1, CHEMWPS, NCRINV, CHEM2, CHEM3, CHEM4, DUCINV, BFTGTS, RGTGTS, CZTGTS, PREAGDM, KADMC, AIRBASE, CHEM5, CHEMDAM
CHB	CHEM5, CHEMDAM
CHCW	CHEM, CHEM6, CHEMLEV, EQUIP, CHEM1, CHEMTAR, CHEMWPS, CHEM2, CHEM3, CHEM4
CW	WTZERO, GCOUT, TCOUT, SPLYOT, WTONE, NUCOUT, CHOUT, PSUMMY
CZZZ	AIRMOD, ALLOCT, DEG, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
GCFM	GC, FEBAMT
LOCAA1	NUC, NUC1, ESCLAT, WHINUP, NDSYINV, NUC2, NUCTAR, NUCWPS, NWHINV, NUC3, NUC4, NUC5, NUCABS, NBFTGS, NRGTGS, PREYLD, DWHINV, NUC6, DAMEVL
LOCAA2	CHEM, CHEM6, CHEMLEV, EQUIP, CHEMSUP, DECON, CHEM1, CHEMTAR, CHEMWPS, NCRINV, CHEM2, CHEM3, CHEM4, DUCINV, BFTGTS, RGTGTS, CZTGTS, PREAGDM, KADMC, AIRBASE, CHEM5, CHEMDAM
LOCAL1	NUC, NUC1, ESCLAT, WHINUP, NDSYINV, NUC2, NUCTAR, NUCWPS, NWHINV, NUC3, NUC4, NUC5, NUCABS, NBFTGS, NRGTGS, PREYLD, DWHINV, NUC6, DAMEVL
LOCAL2	CHEM, CHEM6, CHEMLEV, EQUIP, CHEMSUP, DECON, CHEM1, CHEMTAR, CHEMWPS, NCRINV, CHEM2, CHEM3, CHEM4, DUCINV, BFTGTS, RGTGTS, CZTGTS, PREAGDM, KADMC, AIRBASE, CHEM5, CHEMDAM

Table 4. (Part 3 of 3)

Common Blocks	Routines in Which They Appear
LOCAL3	TARACQ, TARACA, TARACE, TADPAR, BLKDATA
MASS	TMAIN
MATRIX	TRANPO, BLOCK1
PR	SUPPLY, INPUT, INSOL, OUTPUT
QINRPR	FN, PREFN, QKINR
STI	SUPPLY, INPUT
TACQ	TARACQ, TARACA, TARACE, TADPAR, BLKDATA
TRANS	TRANPO INPUT, INSOL, LABEL1, LABEL2, MAIN, CYCLE, FIXLIJ, IJFIX, OUTPUT
Z2	AIRMOD, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
Z3	AIRMOD, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6
ZZ	AIRMOD, ALLOCT, DEG, AIRATT, AOVL1, ATTR1, AOVL2, ATTR2, ATTR3, ATTR4, ATTR5, ATTR6

2.2.1 Root Segment. This subsection describes the routines in the root segment of the TACWAR model.

2.2.1.1 TMAIN. Program TMAIN serves as the control executive for the TACWAR model. TMAIN sets up a cycle counter which is used in controlling the performances of certain intermittant model functions. The program activates the air, nuclear, chemical, target acquisition, ground, supplies, and theater control models. In addition, TMAIN calls routines to read input data and to write input data or summary reports.

2.2.1.1.1 Programming Specifications. The following table summarizes the principal specifications of program TMAIN:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, MASS
Subroutines called	TZERO, WTZERO, WTONE, AIRMOD, NUC, CHEM, TARACQ, GROUND, AIRGRD, PSAIR, TC, SUPPLY, TIMET, PSUMMY

2.2.1.1.2 Logic Functions. The code in this program is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Initialization. TMAIN begins by calling overlay segment LINKA and transferring control therein to subroutine TZERO, which governs the reading of all input data and the initialization of various theater control arrays. Upon return of control to TMAIN, the value of NEPD(3) is examined. If NEPD(3)=0, segment LINKB is overlaid on LINKA and control is passed to WTZERO, which directs the printing of all conventional time-zero input read under the control of TZERO. TMAIN next determines if nuclear or chemical warfare is to be played. If either one is to be played and NEPD(3)=0, segment LINKC is overlaid on LINKB and control is passed to WTONE, which directs the printing of all nuclear, chemical, and target acquisition time-zero inputs. Next, TMAIN initializes variable ICYCLE, the game cycle counter, to a value of 1, and initializes ICSM, the major supply cycle counter, to NCSM. The variable NCSM is the number of combat cycles in a major supply cycle.

b. Section 20 - Execute Air Model. The code in sections 20, 30, and 40 are executed for each cycle of the game. The program compares the current value of the cycle counter, ICYCLE, with each element of the array IPRDO. If a match is found, IPRD is set to 1. IPRD is used at many points in the simulation to direct the printing (=1) or suppression (=0) of detailed reports. The segment LINKD is called next, and control is passed to subroutine AIRMOD, which in turn calls the subroutines that comprise the air combat model.

c. Section 30 - Execute Nuclear and Chemical Models. Upon return of control to TMAIN, the value of IOMU is tested to determine if nuclear or chemical weapons are played this cycle. If they are not played, control is passed to section 40.

First, the flag KNUCH, which is the sum of the number of sectors in which nuclear weapons are played and the number in which chemical weapons are played, is set to 0. The following steps are performed for each nuclear/chemical subcycle, INCYL, with the exception that if, after the first subcycle, no nuclear or chemical weapons have been played (KNUCH=0), then control is passed to section 40. The program sets the flag IPRS to 1 if the current cycle is one on which summary reports are requested. Summary reports are printed only for the first subcycle when requested on the cycle. Then, if nuclear weapons are to be played in the model, TMAIN calls in segment LINKE and control is transferred to subroutine NUC, which will on this call (KFLAG=1) determine the escalation state and perform certain initialization functions for the nuclear model. If chemical weapons are to be played, TMAIN then calls in segment LINKF and control is transferred to subroutine CHEM, which will on this call (KFLAG=1) determine the chemical employment level and perform certain initialization functions for the chemical model.

Program TMAIN then simulates nuclear and chemical warfare on a sector by sector basis. First, it determines if nuclear weapons are played and then determines if chemical weapons are played in the sector. Then, if it is the first subcycle and either nuclear or chemical weapons are played, TMAIN calls in segment LINKG and transfers control to subroutine TARACQ, which in turn calls the routines of the target acquisition model. When control is returned to TMAIN, the program determines in which order the nuclear and chemical weapons are to be played. Then the nuclear (LINKE) and

chemical (LINKF) models are called in appropriately, provided that type weapon is to be played in this sector. The models will simulate nuclear and chemical combat on these calls (KFLAG=0).

d. Section 40 - Execute Remaining Model. Upon completion of nuclear and/or chemical warfare, TMAIN calls in LINKH and transfers control to subroutine GROUND, which governs the ground combat model. When this model has been executed, segment LINKI is overlaid on LINKH and subroutine AIRGRD is called. AIRGRD calls the routines of the air-ground combat model. Upon return of control to TMAIN, it is determined if the current cycle is one on which summary or detail reports are requested, or if it is the first or last cycle of the game. If any of these conditions exist, segment LINKJ is called and control is passed to subroutine PSAIR, which prints summary tables for the air and air-ground routines.

If the present cycle is not the last one of the game, TMAIN calls the nuclear and/or chemical routines, provided nuclear and/or chemical weapons are played by the model. On these calls, KFLAG is set to 2 indicating that only calculations to determine the nuclear escalation state or the chemical employment level are to be performed. On the last cycle, these calls and calls to TC and SUPPLY are omitted.

Next TMAIN calls in segment LINKK and transfers control to subroutine TC, the theater control model. Then if the current cycle is a major resupply cycle, TMAIN calls in segment LINKL and transfers control to subroutine SUPPLY. This link contains the supplies model. Upon return to TMAIN from SUPPLY, the resupply cycle counter ICSM is properly incremented to indicate the next resupply cycle. If the current cycle is one on which time-t variables are to be input, TMAIN calls in segment LINKM, and transfers control to subroutine TIMET, which reads the appropriate inputs from a working file. Upon return of control to TMAIN, it is determined if it is the first or last cycle of the game, or if summary or detail reports have been requested on the current cycle. If any of these conditions exist, TMAIN calls in segment LINKN and transfers control to PSUMMY, which prints summary tables for ground, theater control, and supply variables. If the present cycle is the last of the game, TMAIN terminates execution. If not, the cycle counter ICYCLE is incremented by 1 and TMAIN begins the simulation for the next cycle.

2.2.1.2 EIGENV. The purpose of this subroutine is to calculate the numerical values (eigenvectors) for Red and Blue weapons such that the magnitude of the values indicate the relative worth (in terms of combat effectiveness) of each side's weapons. One Blue weapon is chosen as a reference weapon, and the values of all other Blue weapons, as well as those of all Red weapons, are measured relative to the value of the reference weapon.

EIGENV is called to determine each side's weapon values and the resulting force ratios by means of the antipotential potential method. A crucial figure computed by EIGENV is the eigenvalue, or characteristic number, of the kill rate matrix. The square root of this figure represents an average kill rate for the weapons in the matrix. The average kill rate is used to relate the Red weapon values and the values of air munitions to the same reference weapon used to measure ground weapon values.

The methodology used by subroutine EIGENV is described in detail in appendix D of the NATO Combat Capabilities Analysis II (COMCAP II) Report (see reference 5). This report states the following two equations from which the desired eigenvalue and corresponding eigenvectors are determined:

$$\bar{V}_b = c^2 \quad BR \; \bar{V}_b \tag{1}$$

and

$$\bar{V}_r = cR \; \bar{V}_b \tag{2}$$

where

$\bar{V}_b$ = a column vector for the Blue type weapon values

$\bar{V}_r$ = a column vector for the Red type weapon values

c = the average kill rate (eigenvalue = c^2)

B = Blue's "kill rate matrix" (Blue weapons kill Red weapons)

R = Red's "kill rate matrix" (Red weapons kill Blue weapons)

The unknowns to be found are $\bar{V}_b$, $\bar{V}_r$, and c. Starting with initial values for $\bar{V}_b$ in the right side of equation (1), values for the two unknowns ($\bar{V}_b$ and c) can be determined using an iterative procedure as described in reference 5. As more iterations are performed, the value of c will converge and the calculated approximation of $\bar{V}_b$ will approach its true value. When these unknowns have been computed to within a specified degree of accuracy, equation (2) is used to calculate the $\bar{V}_r$ values. It should be noted that the vectors $\bar{V}_b$ and $\bar{V}_r$ are:

$$V_b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ 1 \\ b_5 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \text{ and } V_r = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \qquad (3,4)$$

where b_4 has been chosen (by the user) as the Blue reference weapon and maintains a combat effectiveness of 1. Thus, upon successful completion of this procedure, the combat effectiveness of any other Blue weapon or any Red weapon can be compared to this reference weapon.

2.2.1.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine EIGENV:

Characteristic	Specification
Formal parameters	AA = Blue's "kill rate matrix" (i.e., B in equation (1))
	BB = Red's "kill rate matrix" (i.e., R in equation (1))
	V = Initial $\bar{V}_b$ column vector for starting the iterative process

Characteristic	Specification
	VB = Blue weapon values, a column vector (i.e., $\overline{V}_b$ in equation (1))
	VR = Red weapon values, a column vector (i.e., $\overline{V}_r$ in equation (2))
	N = Number of Blue weapon types
	M = Number of Red weapon types
	I1 = Index for the Blue reference weapon
	MNIE = Maximum number of iterations to be attempted
	EFCE = The degree of accuracy required for the eigenvalue
	ALAM = The final eigenvalue
Common blocks	None
Subroutines called	MPROD
Called by	TCTZ, GC

2.2.1.2.2 Logic Functions. Subroutine EIGENV contains adjustable dimensions except for two working arrays (R and W) which are dimensioned for maximum weapon types. The adjustable arrays are dimensioned by the arguments N and M, which also determine loop iterations. Because SAMs have no ground value they are excluded from the computation.

EIGENV is called twice by the calling subroutine; once for the values of Blue weapons on defense and the values of Red weapons on attack, and again for the values of Blue weapons on attack and the values of Red weapons on defense. In each case, the iteration procedure must begin with initial values

for the column vector, $\overline{V}_b$ (the Blue weapon value array, VB, in the program) on the right side of equation (1).

The initial values for the Blue weapons (array V) are initialized in subroutine TCTZ prior to the first call to EIGENV. Subsequent calls use, as initial values, the values calculated for the Blue weapons value array (VB) during the previous call. EIGENV calls subroutine MPROD to perform an inner multiplication of the two "kill rate" matrices AA and BB (B and R in equation (1)) yielding as a matrix product the working array R.

An iteration loop is run, for a maximum of MNIE attempts, in order to bring the eigenvalue, ALAM, to convergence. The first step in the convergence loop is a call to MPROD for the matrix product of R and the current Blue weapons value column vector, VB. The resulting column vector, W, represents the Blue weapons' values prior to normalization by the Blue reference weapon.

The reference weapon's value, in vector W, is now compared to a minimum acceptable value. If the value of the reference weapon is less than a value of .00001 an error message is printed and the program terminates since force ratios and dependent computations based on antipotential potential calculations would be meaningless.

The current approximation of the eigenvalue, ALAM, is taken to be the inverse of the Blue reference weapon value. Each element of array W is normalized by the eigenvalue and stored in array VB. Thus, the reference weapon in VB always maintains a value of 1.

The subroutine tests the ratio of the current value of ALAM to the eigenvalue computed during the previous iteration, ALAMO. If the ratio is within the tolerance EFCE of unity, the eigenvalue is assumed to have converged satisfactorily and an exit is made from the convergence loop. If not, the iterations continue. When the loop has run MNIE times without convergence, a warning message is printed and the current eigenvalue and its corresponding eigenvectors are used for succeeding computations.

At this point, the Red weapons' values are calculated by taking the square root of the eigenvalue and using the root to solve for VR ($\overline{V}_r$ in equation (2)). Program control then returns to the calling routine.

2.2.1.3 MPROD. Subroutine MPROD performs an inner product multiplication of two matrices, one N by M and the other M by L in dimension. The resulting matrix, therefore, has the dimensions N by L.

2.2.1.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine MPROD:

Characteristic	Specification
Formal parameters	A = Input matrix, dimensioned N by M
	B = Input matrix, dimensioned M by L
	R = Output matrix containing inner products of [A] times [B], dimensioned N by L
	N = Number of rows in A and rows in R
	M = Number of columns in A and rows in B
	L = Number of columns in B and columns in R
Common blocks	None
Subroutines called	None
Called by	EIGENV, TCTZ, GC

2.2.1.3.2 Logic Functions. MPROD consists of three nested DO loops. The algorithm produces a value for each member of matrix R which has N rows and L columns. Each value of matrix R is an inner product, i.e., the sum of the products obtained by multiplying in turn each member of a given column of the first input matrix by the corresponding member of a given row of the second input matrix. In other words, the R matrix value specified by R(I,J) is found by multiplying the first member of the Ith row in matrix A by the first member of column J in matrix B and adding the resulting product to that obtained by multiplying the second member of

the Ith row in matrix A by the second member of column J in matrix B. This process is continued through the Mth member of the Ith row in matrix A and the Mth member of column J in matrix B. When all members of matrix R have been computed, control returns to the calling routine.

2.2.1.4 CNTRYC. Function CNTRYC determines the index of the country to which a division belongs. Presently this routine is a dummy routine in that it sets the country code to 1 for any division. If the user desires to play more than one country, he must change this routine to make the assignments of divisions to various countries. He must also make numerous coding changes to other routines since they do not always call CNTRYC to determine the country association for a given division. Therefore, the number of changes needed to play more than one country in the TACWAR model are not as minimal as might be expected.

2.2.1.4.1 Programming Specifications. The following table summarizes the principal specifications of CNTRYC:

Characteristic	Specification
Formal parameters	ID = Division index
Common blocks	None
Subroutines called	None
Called by	GC, TC

2.2.1.4.2 Logic Function. The function returns a value of 1 for CNTRYC for any division index ID.

2.2.1.5 CVFW. Subroutine CVFW is used to calculate an ordinate value of a piecewise linear function given the corresponding abscissa value and the coordinates of several endpoints defining the function segments. CVFW is called for the purpose of computing casualty percentages as a function of posture and force ratio; combat effectiveness as a function of division type and percent personnel strength; and supply effectiveness as a function of supplies on hand. The user must input coordinates for two sets of functions, one for Red and one for Blue. Within each set, individual function types such as casualty percentage must be defined as a family of lines, each of which corresponds to a particular type of posture, with a set of endpoint coordinates

specifying predicted casualty rates paired with graduated force ratios. The calling subroutine selects the proper function, given posture and computed force ratio, and CVFW determines casualty percentages by interpolating from the endpoint abscissas that most nearly match the force ratio figure.

2.2.1.5.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CVFW:

Characteristic	Specification
Formal parameters	NVF = Number of endpoints used in the function
	VFX = Array holding X-values at the function endpoints
	VFY = Array holding Y-values at the function endpoints
	W = Input abscissa value
	V = Computed ordinate value corresponding to W
Common blocks	None
Subroutines called	None
Called by	TCTZ, DEG, DAMEVL, GC, FEBAMT, TC, TIMET

2.2.1.5.2 Logic Functions. CVFW tests to determine if W is equal to the X-value of any endpoint of the function being used. If an equality is found, V is set equal to the Y-value of the matching endpoint before returning control to the calling routine.

If W is found to lie between two endpoints, the subroutine uses the displacement between W and the lower endpoint's X-value to compute a proportionate displacement for the Y-value.

2.2.1.6 SECWTH. Subroutine SECWTH determines the width of the sector at a given point along the path of FEBA advance.

2.2.1.6.1 Programming Specifications. The following table summarizes the principal specifications of subroutine SECWTH:

Characteristic	Specification
Formal parameters	IS = Sector index
	GD = Ground distance to the point where the width is to be determined
	NBNLT = Number of boundary longitude points along the sector
	SBNDLT = Boundary latitudes at the set longitude points along the sector
	WDTH = Width of sector at point GD
	P1LT = Latitude of midpoint of line joining the sector breakpoints at the near edge of the segment containing the point GD
	P1LN = Longitude corresponding to P1LT
	P2LT = Latitude of midpoint of line joining the sector breakpoints at the far edge of the segment containing the point GD
	P2LN = Longitude corresponding to P2LT

Characteristic	Specification
	AL = The ratio of the distance from the near edge of the segment to the point GD, and the length of the segment
	DST = Length of the segment, measured as the distance between the midpoints of the lines joining the breakpoints at the near and far edges of the segment
Common blocks	None
Subroutines called	GDIST
Called by	TCTZ, NUCABS, AIRBASE, TC

2.2.1.6.2 Logic Functions. Subroutine SECWTH first determines that segment of the sector which contains the point P at ground distance GD and then determines the width of the sector at that point. Each sector boundary is defined by up to seven points which serve as breakpoints for the sector. These points are selected such that the longitudes of the points are at regular evenly spaced intervals with the longitude of corresponding breakpoints at each sector being the same. Thus each sector is divided into a maximum of six segments whose near and far edges fall on these longitudes. See figure 3 for a schematic of a sector. For the purpose of calculations, longitudes starting at 24 degrees and decreasing in intervals of 4 degrees are used to cover the theater from the Red COMMZ to the Blue COMMZ. For each sector the latitudes corresponding to each of the longitude values, the last of which lies in the Red COMMZ, are specified by the input array SBNDLT.

Since the distance GD is measured from a point in the Red COMMZ, the routine first checks for point P in the last segment of the sector. The latitudes and longitudes of the break points which define the segment are used to determine the midpoints of the near (i.e., closer to the Red COMMZ)

and far edges of the segment. Function GDIST is called to calculate the ground distance between the two midpoints. The distance DST is the length of the segment. If DST exceeds the ground distance GD, then the last segment is the one which contains the point P. If not, the lengths of each of the remaining segments are similarly calculated one at a time from higher to lower index until the total distance (SUM) through the searched segments exceeds GD. Then, for the segment containing P, the routine calculates the ratio (AL) of: (1) the distance from the near edge of the segment to the point P, and (2) the length of the segment. This ratio is used to determine the intersections (in terms of latitude and longitude) of the longitude line, which passes through the point P, with the upper and lower boundaries of the sector. (Recall that three parallel lines always divide any intersecting line segment bounded by the two outer lines in the same proportion.) The distance between these two intersections, which is determined by GDIST, is the width of the sector at the point P.

2.2.1.7 GDIST. Function GDIST determines the ground distance between two points, given their latitudes and longitudes.

2.2.1.7.1 Programming Specifications. The following table summarizes the principal specifications of function GDIST:

Characteristic	Specification
Formal parameters	RLAT1 = Latitude of the first point
	RLN1 = Longitude of the first point
	RLAT2 = Latitude of the second point
	RLN2 = Longitude of the second point
Common blocks	None
Subroutines called	None
Called by	SECWTH, AIRBASE, NUCABS

2.2.1.7.2 Logic Functions. Function GDIST uses the properties of spherical triangles to determine the ground distance between two points. The spherical triangle under consideration is the one formed by the great circle arcs joining the two points and the closer pole. The complement of each of the two latitudes is converted from degrees to radians to yield two sides of the triangle. The polar angle, which is the difference in longitudes of the two points, is also converted to radians. The arc between the two points is determined from the following formula:

$$COS(a) = COS(b)\ COS(c) + SIN(b)\ SIN(c)\ SIN(A)$$

where a, b, and c are the sides of the spherical triangle and A is the angle opposite side a.

The arc length is converted from radians to kilometers to yield the ground distance between the two given points.

2.2.1.8 TAG. Subroutine TAG provides an identifying code for each battle area in the theater to indicate a change in status due to FEBA movement. Battle areas so identified may be in the COMMZ, region rear, or region forward for either Blue or Red or may be the active battle area of a particular sector.

2.2.1.8.1 Programming Specifications. The following table summarizes the principal specifications of TAG:

Characteristic	Specification
Formal parameters	NBA = Number of battle areas in the theater
	NS = Number of sectors
	FEBA = FEBA location by sector
	GDBA = Ground distance to leading edge of each battle area in each sector
	IABAS = Index of the active battle area by sector for the last cycle

Characteristic	Specification
	NDFAB = Number of battle areas in the region forward by side
	NDRAB = Number of battle areas in the region rear by side
	ISAT = Identifying codes assigned by TAG to battle areas
Common blocks	Blank Common
Subroutines called	None
Called by	APORTN, INP, DEG

2.2.1.8.2 Logic Functions. Subroutine TAG determines the status of a battle area last cycle and its status this cycle. Then a code is assigned to indicate the change in status.

A battle area is assigned a status code of 1 through 7 with the following relationship to location:

1 - Red COMMZ

2 - Red Rear

3 - Red Forward

4 - Active Battle Area

5 - Blue Forward

6 - Blue Rear

7 - Blue COMMZ

If I is the status of the battle area at the last cycle, and J is the current status, the matrix ITR gives the following assignments:

I	J						
	1	2	3	4	5	6	7
1	17	19					
2	10	15	8				
3		6	13	4			
4			2	11	1		
5				3	12	5	
6					7	14	9
7						18	16

where blanks indicate zeros. The meaning of the assignment code is as follows:

Code	Old Owner of B/A	New Owner of B/A
1	Active B/A	Blue Forward
2	Active B/A	Red Forward
3	Blue Forward	Active B/A
4	Red Forward	Active B/A
5	Blue Forward	Blue Rear
6	Red Forward	Red Rear
7	Blue Rear	Blue Forward
8	Red Rear	Red Forward
9	Blue Rear	Blue COMMZ
10	Red Rear	Red COMMZ
11	Active B/A	No Change
12	Blue Forward	No Change
13	Red Forward	No Change
14	Blue Rear	No Change
15	Red Rear	No Change
16	Blue COMMZ	No Change
17	Red COMMZ	No Change
18	Blue COMMZ	Blue Rear
19	Red COMMZ	Red Rear

The calculations in this routine are performed sector by sector. Subroutine TAG first determines the new active battle area INUBA in the sector, based on the new FEBA location.

Then for each of the seven locations (from Red COMMZ to Blue COMMZ), the highest numbered battle area in the location for the last cycle and the one for the current cycle are determined and stored in array IVAL. If the index for a Red location is calculated to be less than 1, then the index is set to 0. If the index determined for a Blue location is greater than the number of battle areas, then the index is set to the largest battle area in the sector or to 999. (Either will suffice for the calculations which follow.)

Finally, for each battle area, its status last cycle and this cycle are determined by comparing its index to the values of IVAL. The change in status, ISTAT, of the battle area is obtained from array ITR.

2.2.1.9 APORTN. Subroutine APORTN provides the logic for apportioning resources such as aircraft, military personnel, and shelters from each notional airbase to the actual airbases which make up the notional base.

2.2.1.9.1 Programming Specifications. The following table summarizes the principal specifications of APORTN:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	TAG
Called by	INP, AIRGRD, AIRMOD, Tmain

2.2.1.9.2 Logic Functions. Subroutine APORTN first calls TAG to determine the change in status for each of the actual airbases. Then each airbase is assigned a 3-digit tag ISAB, with the leading digit indicating sector location, the second one indicating current area location (1, 2, and 3 for region forward, region rear, and COMMZ, respectively), and the last digit indicating side. For airbases in the COMMZ, the sector location is considered to be 1; airbases in the active battle area are tagged as 0 and are not used in any of the following calculations.

For each of the locations (indicated by side, sector, and area), the actual airbases in that location are identified. (If the number of actual airbases in a location exceeds 60,

the program terminates with an error message to that effect.) Then each type of aircraft from the notional airbase for this location (e.g., ACFS for the forward region) are reallocated to the actual airbases that make up the notional. This allocation is made in proportion to a weighted number of the sum of that type aircraft that existed on the actual airbase in the initial image (IMAGE) and the level of that type aircraft at the actual airbase at the end of the last cycle (IWORD). If there were no aircraft of that type in the initial image of any of these actual airbases or at these actual airbases during the last cycle, then that type aircraft at the notional airbase will be reallocated in proportion to the weighted number of all aircraft in the initial image. Similarly, the military personnel (after being adjusted to reflect losses this cycle) and shelters at the notional airbases are reallocated to the actual airbases which make up the notional. When all reallocations have been determined, the values are rounded off and their integer values are stored appropriately in array IWORD.

When called from INP before the start of the battle, APORTN also sets the number of aircraft by type, the total number of aircraft, the number of shelters, and the number of military personnel in the original image of the actual airbases (IMAGE) equal to the values just determined for allocation to them from the notional airbases in the location.

2.2.1.10 CLR. Subroutine CLR, which contains entry point CLR2, initializes each item of an array to a value specified by the call. In general, this routine is called to clear an array by setting it to zero. To initialize a real array CLR(Z,N,V) is called; to initialize an integer array CLR2(IZ,N,IV) is called.

2.2.1.10.1 Programming Specifications. The following table summarizes the principal specifications of CLR:

Characteristic	Specification
Formal parameters	Z or IZ = Array name
	N = Size of array
	V or IV = Value to which each item of the array is to be set.

Characteristic	Specification
Common blocks	None
Subroutines called	None
Called by	APORTN, TCTZ, DEG, NUCWPS, CHEMWPS, AIRASG, CHEM, PREAGDM

2.2.1.10.2 Logic Functions. If subroutine CLR is called, each of the N items of the array Z is set to the value V. If entry is made at CLR2, each of the N items of the array IZ is set to the value IV.

2.2.2 LINKA. This subsection describes the routines contained in LINKA. These routines are input and initialization routines.

2.2.2.1 TZERO. Subroutine TZERO calls two subroutines, one to read the time-zero inputs and one to initialize the program variables and geographic quantities.

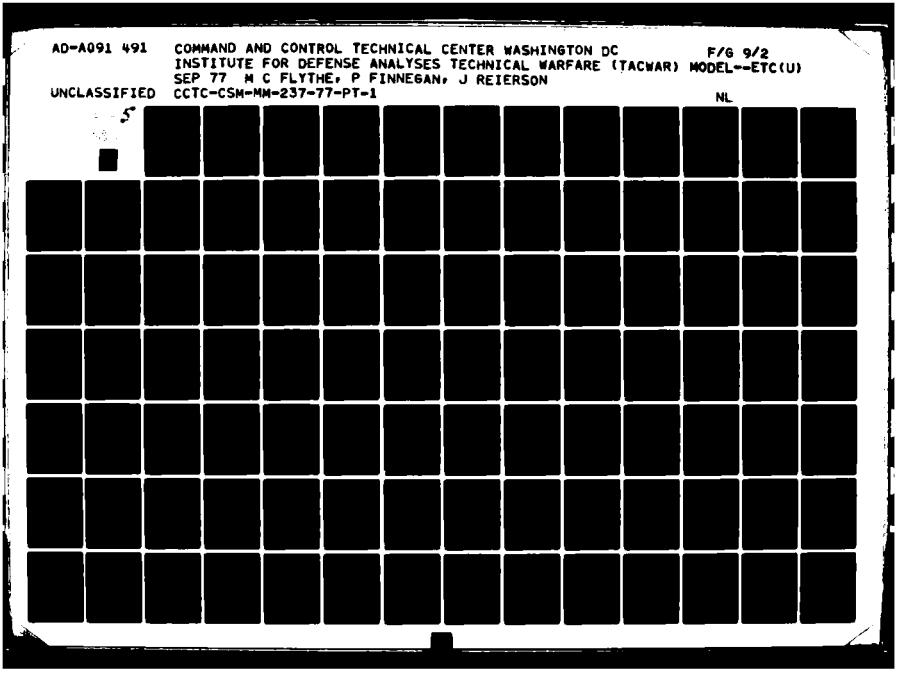
2.2.2.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TZERO:

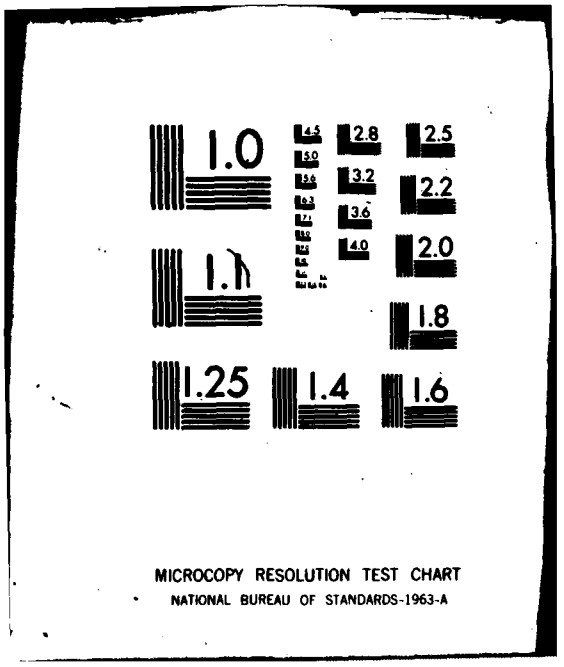
Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	INP, TCTZ
Called by	TMAIN

2.2.2.1.2 Logic Functions. Subroutine TZERO first calls subroutine INP to process all data input by the user via cards or card image files. INP screens the information for format errors and prepares certain data for efficient loading into proper data arrays at a later time. TZERO then calls subroutine TCTZ to determine certain geographic quantities and to initialize a number of working arrays. TZERO is called only once during a game.

2.2.2.2 INP. Subroutine INP processes all user-specified data input via cards or card image files. The program serves two main functions. First, it checks that the cards are formatted correctly according to one of three format types. Secondly, it prepares the data for loading into the proper data array in blank Common. The card formats may be input in any order. The only restriction is that the cards be arranged in ascending order of days. Cards that are of the first two format types contain four column integer data and six column noninteger data, respectively, that are applicable to the variable name specified on the card. Cards that are of the third format type contain data that characterize units arriving in the theater after the war has already begun. Finally, airbase image data is read and is used to determine the number of actual airbases and the resources of each airbase which compose each notional airbase.

AD-A091 491 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 9/2
INSTITUTE FOR DEFENSE ANALYSES TECHNICAL WARFARE (TACWAR) MODEL--ETC(U)
SEP 77 M C FLYTHE, P FINNEGAN, J REIERSON
UNCLASSIFIED CCTC-CSM-MM-237-77-PT-1 NL




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

2.2.2.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine INP:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	TAG, APORTN
Called by	TZERO

2.2.2.2.2 Logic Functions. The code of INP is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Read Input File MIT and Determine Format Type. A card image containing user input is read from input file MIT. If all data cards have been read, then control is transferred to section 40. Otherwise, this section performs the following functions. The last field on the card, regardless of format, is the day number field, and is decoded and stored in variable IDAY. The first field is checked to determine whether it contains an array name or the word UNIT, indicating that the card contains unit assignment data. In the former case, control is transferred to section 30 and in the latter to section 20.

b. Section 20 - Process Unit Assignment Data. The format for unit assignment data consists of fields which describe the unit ID, the unit strength (the actual number of people in the division), the unit supplies, the battle area, sector, region, nationality, and sector tag. Subsection 3.2.1.1.2 discusses how unit assignments are coded and what they mean.

If the day number (IDAY) for which the unit data apply is zero, then the unit strength, the number of unit supplies, the unit type, and the index to the battle area are decoded and stored as initial values into the proper input arrays. The sector, region, nationality, and sector tag are decoded and stored in an input buffer (IREC). The input arrays and IREC, together with IDAY and the unit index (ID) are written to output file JINP. If IDAY is greater than zero, then the data is decoded and stored into IREC, which is then written to an unformatted working file (ITTD) of time-t data. In

addition the same information is written to output file JINP. For a fuller description of the files ITTD and JINP see subsections 3.2.1.2 and 3.2.2. Control is then returned to section 10, which will read the next input card.

c. Section 30 - Process Array Name Input. First, data array IVARQ is searched for the array name. IVARQ(JV,7) contains information necessary for processing the user-coded data. Each JV corresponds to the index of the array into which the input data on the card is to be loaded. There are seven items of information associated with each numbered array name. A more complete description of the contents of IVARQ can be found in subsection 4.1.1.1. If a match is not found between the input array name on the card and the array names contained in IVARQ, then an error message is prompted indicating that the input name is not in the list and the card is therefore ignored. Control is then returned to section 10. Otherwise, once a match has been found, certain items of information associated with the array in IVARQ are retrieved. These items include the maximum index values of each dimension of the array which, when retrieved, are stored in variables MAXI, MAXJ, and MAXK. Also, the number of dimensions of the array is computed and stored in variable ICL.

Data may be input to an array in a variety of ways. A single item of data may be input into a specific location of an array by coding the index fields I, J, and K with values that indicate the desired location. When a value is coded in the I field for a one-dimensional array (J and K are left blank), the item of data on the card is input to that Ith element of the array. Similarly when values are coded in the I and J fields for a two-dimensional array or in the I, J, and K fields for a three-dimensional array, then the single item of data is input to the Ith by Jth element or the Ith by Jth by Kth element of the array, respectively. Data may also be input to an array by specifying several data values on the same card. Subsection 3.2.1.1.1 discusses coding data in this manner.

Next, a series of tests are performed comparing the maximum index values against the coded values of the I, J, and K fields on the input card to ensure that the card has been coded correctly (i.e., I, J, and K should not exceed the values of MAXI, MAXJ, and MAXK, respectively). If a card has been coded improperly, then an error message is prompted indicating that the card will be ignored. In addition, the

tests on variable ICL determine how the input data was coded and, consequently, how the data is to be entered. For example, if ICL=1 and a value greater than zero has been coded in the I field, then the array in question is one-dimensional (as indicated by ICL) and a single item of data is to be entered into the location indicated by I. If however, ICL=1 and I, J, and K have been left blank, then the data is entered into the one-dimensional array for I = 1,2,3,...MAXI.

Once the method of coding the input has been determined, a check is made to ensure that the input data lies within the starting and ending locations of the area in Common that has been reserved for it by IVARQ. This is done by first determining the dimension index (I, or J, or K) for which the data is being entered. Next, variables IST and IMAX are used to keep track of the relative array index values of the first and last items of data, respectively, on the input card.

Suppose array A(3,4) is input on three data cards, each of which contains data for a fixed I index and the four J values. Each array in Common is stored such that the first index varies more rapidly, i.e., the order is A(1,1), A(2,1), A(3,1), A(1,2), etc. Therefore, when the first data card for array A (which contains A(1,1), A(1,2), A(1,3), A(1,4)) is read, IST is set to 1 and IMAX to 10. When the second card is read, IST is set to 2, and IMAX to 11.

For each card, the relative values that IST and IMAX take on are added to the starting location of the given array in Common. If either value lies outside the interval defined by the starting and ending locations of the array in IVARQ then, again, an error message is prompted indicating that the card was coded improperly and is therefore ignored. Control is then returned to section 10 which will read the next data card.

If the values fall within the size limits of the array, the data will be processed. If the data is time-zero data, then it is loaded into the proper array in Common and control returns to section 10. If the data is for a later day, an output record is built and written to the unformatted working file ITTD for subsequent reading by subroutine TIMET. In addition, a record is built for each array name and written to output file JINP. The record contains the day number; the words "RPLACE" or "INCREM", indicating that the data is either replaced or incremented; the array name; the number

of the continuation card (blank is taken as zero); the index values coded in the I, J, and K fields, if applicable; and finally, the input data to be loaded. Control is then transferred to section 10, which will process the next data card.

d. Section 40 - Read Input File IAD and Determine Actual Airbase Resources. When all of the input cards from file MIT have been processed, subroutine TAG is called to assign a logic code to each battle area in each sector of the theater to indicate its current status. Corresponding to each logic code there is a two-digit number or a zero in data array MPD. The leftmost digit of the number gives the battle area location for a given airbase (1 designates sector forward, 2 designates sector rear, and 3 designates the COMMZ) and the rightmost digit indicates the side to which the battle area currently belongs. A zero value indicates that the location is the active battle area and that no ownership has been established.

Input file IAD (see subsection 3.2.1.3 for a description of the file), which contains airbase image data, is then read. If any index associated with the number of the airbase is out of sequence, an error message is prompted indicating the number of each division force against the standard force, and combat effectiveness based on resource levels. All quantities computed in TCTZ are written on output file JINP when requested by the user (NEPD(2)=0).

2.2.2.3 TCTZ. Subroutine TCTZ, Theater Control Time Zero, determines various quantities such as the number of people and weapons, by type, for each division, the weapon values of each division force against the standard force, and combat effectiveness based on resource levels. All quantities computed in TCTZ are written on output file JINP when requested by the user (NEPD(2)=0).

2.2.2.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TCTZ:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	SECWTH, EIGENV, MPROD, CVFW, CLR
Called by	TZERO

2.2.2.3.2 Logic Functions. The code of subroutine TCTZ is divided by comment cards which label the functions performed by the logic.

a. Section 5 - Set Division Resources not Specifically Input by User to TOE Levels. This section of TCTZ is performed for each side. The user may input specific values for the elements of four arrays - PDIV, SDIV, WDIV, and NSUTD - the actual number of people, supplies, weapons (by type), and subunits (by type) in each division. If user inputs have not been provided, this section of TCTZ sets the values of these arrays equal to the TOE resource levels for each division.

b. Section 10 - Compute NLSR(IR), NLSNS(IS), and Set FEBA(IS). This section of TCTZ begins by determining the lowest numbered sector in each region for each side (NLSR(IR)) based on the user-input highest numbered sector in each region (NHSR(IR)). The lowest numbered supply node (excluding node 1) in each sector (NLSNS(IS)) is determined similarly. The location of the Forward Edge of the Battle Area in each sector (FEBA(IS)) is set to the input value for the FEBA location at time zero (FEBATZ(IS)).

c. Section 15 - Compute Sectors of Main Attack. In this section of TCTZ, the index to the sector of main attack (ISMA(IS)) is set equal to 1 for the sector in each region that will be the sector of main attack on the first day of the battle, and equal to 0 for all other sectors. This section begins by checking the value of ICSMA, which is the index used to compute the sector of main attack. If ICSMA=2, then ISMA is set to user input values and the logic flow branches to section 20. If ICSMA=1, then the model computes ISMA. The sector of main attack in a region is assumed to be the sector of maximum FEBA advance in that region, i.e., the sector in which the FEBA is the furthest into the enemy's territory.

d. Section 20 - Calculate Geographical Quantities. The logic of this section of TCTZ is performed for each sector. Subroutine SECWTH is called to compute the width of the sector at the FEBA. The location of the FEBA is compared to interval boundaries (BNDIS) in order to determine the interval in which the FEBA is located (INTRVL). The kind of posture (KPS) and terrain (KTER) at the FEBA are then determined.

e. Section 40 - Calculate Weapon Values Against a Standard Force by Using APP. This section of TCTZ begins by computing values for the elements of the kill potential

matrices used in the antipotential potential method of computing weapon values. Kill potential matrices are constructed for Blue ground weapons and air munitions for Blue on attack and on defense against a Red standard force and, similarly, for Red ground weapons and air munitions for Red on attack and on defense against a Blue standard force. Then, individual ground weapon values on attack and defense are computed for both sides by subroutine EIGENV. Subroutine MPROD is used to compute individual air munition values.

f. Section 45 - Compute Effectiveness of All Divisions. The logic of this section of TCTZ is performed once for each side. First, the total TOE weapons values for a division on attack and on defense (WVDATS, WVDDTS) are computed for each division type. Then, the total actual weapon values for a division on attack and defense (WVDAC, WVDDC) are computed for each division. The fractional personnel strength is calculated for each division as the ratio of the actual number of people in the division to the TOE number of people for the division. Subroutine CVFW is called twice in order to evaluate the functions which give PEA and PED, which are the fractional effectiveness of a division, based on personnel strength, for attack and defense, respectively. The number of days of supplies on hand is computed and CVFW is called to evaluate the function which gives SEF, the fractional effectiveness of a division due to supply shortages. The effectiveness of each division on attack (EFFDA) is then computed based on the values of PEA and SEF, and the ratio of actual to TOE weapon values. The effectiveness of each division on defense (EFFDD) is similarly computed from PED, SEF, and the ratio of actual to TOE weapon values.

g. Section 48 - Compute IDLIBA(IDV,IS,L) and NDIBA(IS,L). The logic of this section is performed for each sector and for each side. The index to the battle area location of each division (IBALD(ID)) is compared to the index of the first inactive battle area in the sector (JBA). Based on these comparisons, the index to the division in each location of the first inactive battle area of the sector (IDLIBA) and the number of divisions in that battle area (NDIBA) are computed.

h. Section 50 - Initialize Variables. This section of TCTZ initializes, for both sides, a number of arrays which will be used to maintain cumulative results and a number of model indices for both sides. Also, weapon inventories, which were input in hundreds, are converted to units in this section.

i. Section 55 - Convert Hourly and Daily Rates to 12-Hour Rates. In this section, shipping and supply consumption rates for each side are converted to 12-hour rates.

j. Section 60 - Compute Number of Army-Air Carriers. This section of TCTZ computes the number of each type of army-air carrier in each sector for each side by summing the TOE number of army-air carriers associated with each division in the sector. Control is then returned to TZERO.

2.2.3 LINKB. This subsection describes the routines contained in LINKB. These routines produce tables of input variables for conventional warfare.

2.2.3.1 WTZERO. Subroutine WTZERO is a calling routine which controls the printing of the ground combat, theater control, and supply related time-zero inputs read by subroutine INP.

2.2.3.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine WTZERO:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	GCOUT, TCOUT, SPLYOT
Called by	TMAIN

2.2.3.1.2 Logic Functions. WTZERO calls subroutines GCOUT, TCOUT, and SPLYOT to print input tables on output file JINP.

2.2.3.2 GCOUT. Subroutine GCOUT writes input tables I1 through I22 on output file JINP. These tables describe model parameters and contain data about divisions (including location, people, weapons, munitions, and supplies), weapon and munitions values, FEBA movement, and similar data.

2.2.3.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine GCOUT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	None
Called by	WTZERO

2.2.3.2.2 Logic Functions. GCOUT uses formatted write statements and loops where appropriate to generate the following tables:

Tables	Headings
I1	Model Parameters
I2	Variables That Represent Physical Quantities
I3	Division Data
I4	Division Location in Active Battle Area (by Sector)
I5	Number of People and Weapons by Type Division (TOE Level)
I6	Ground Parameters
I7	Aircraft Munitions Load
I8	Supply Effectiveness Function (by Days of Supplies on Hand)
I9	Blue Percent Casualties
I10	Red Percent Casualties
I11	Blue Effectiveness Function (by Percent Personnel Strength)
I12	Red Effectiveness Function (by Percent Personnel Strength)
I13	Blue Standard Allocation of Air Munitions Against Weapons
I14	Red Standard Allocation of Air Munitions Against Weapons
I15	Blue Standard Allocation of Weapons Against Weapons
I16	Red Standard Allocation of Weapons Against Weapons
I17	Values for Blue, an Air Munition Against a Weapon

Tables	Headings
I18	Values for Red, an Air Munition Against a Weapon
I19	Values for Blue, Weapon Against Weapon
I20	Values for Red, Weapon Against Weapon
I21	FEBA Movement and Mobility Factors
I22	Movement Constraints Based on Flank Exposure and Security Force Ratios (Attacker to Defender)

2.2.3.3 TCOUT. Subroutine TCOUT writes input tables I23 through I32 on output file JINP. These tables describe the initial structure of the theater in terms of posture, boundaries, depth, and width parameters, etc.

2.2.3.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TCOUT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	None
Called by	WTZERO

2.2.3.3.2 Logic Functions. TCOUT uses formatted write statements and loops where appropriate to generate the following tables:

Tables	Headings
I23	Kind of Posture and Terrain in Intervals in Sectors
I24	Number of Intervals, Boundary Latitudes, and Boundaries in Sectors
I25	Cumulative Ground Distance to Leading Edge of Battle Area (KM)

Tables	Headings
I26	Theater Structure, Theater Attacker, and Method for Computing Sector of Main Attack
I27	Division and Subunit Depth and Width Parameters
I28	Parameters for Division Movement
I29	Parameters for Reinforcements, Withdrawals, and Replacement
I30	Percentages for Weapon Repair
I31	Blue Subunit Data
I32	Red Subunit Data

2.2.3.4 SPLYOT. This subroutine writes input tables I33 through I36 on output file JINP. These tables describe supply information such as supply to battle areas, days of supply on-hand, inventory, and supply consumption rates.

2.2.3.4.1 Programming Specifications. The following table summarizes the principal specifications of subroutine SPLYOT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	None
Called by	WTZERO

2.2.3.4.2 Logic Functions. SPLYOT uses formatted write statements and loops where appropriate to generate the following tables:

Tables	Headings
I33	Supply Nodes Supplying Battle Areas by Sector
I34	Days of Supply On-Hand

Tables	Headings
I35	Supply Nodes...Inventory, Ownership and Location
I36	Supply Data and Consumption Rates

2.2.4 LINKC. This subsection describes the routines contained in LINKC. These routines produce tables of input variables for nuclear and chemical warfare and target acquisition.

2.2.4.1 WTONE. Subroutine WTONE is a calling routine which controls the printing of the nuclear, chemical, and target acquisition related time-zero inputs read by subroutine INP.

2.2.4.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine WTONE:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	NUCOUT, CHOUT, TACQOT
Called by	TMAIN

2.2.4.1.2 Logic Functions. WTONE calls subroutines NUCOUT, and CHOUT to print input tables on input file JINP. If the target acquisition model is played, TACQOT is called to print additional input tables.

2.2.4.2 NUCOUT. This subroutine writes input tables I37 through I48 on output file JINP. These tables describe such items as nuclear strike characteristics, targeting information, escalation criteria, protection categories, nuclear weapon system resources, etc.

2.2.4.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUCOUT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	None
Called by	WTONE

2.2.4.2.2 Logic Functions. NUCOUT uses formatted write statements and loops where appropriate to generate the following tables:

Tables	Heading
I37	Escalation State Characteristics
I38	Characteristics of a Preemptive Nuclear Strike
I39	Priority Listings of Preferred Nuclear Targets
I40	Parameters for Nuclear Targeting
I41	Index to Allowable Target Types (Nuclear)
I42	Criteria for Nuclear Escalation - Worsening Tactical Situation
I43	Nuclear Escalation Criteria - Initial or Increased Use of Nuclear Weapons by Other Side
I44	Parameters for Surface Burst Targets
I45	Protection Categories
I46	Elements for Calculating Blast and Radiation Damage to Personnel from Airburst and Surface Burst Weapons
I47	Vulnerability Numbers
I48	Nuclear Weapon System Resources

2.2.4.3 CHOUT. This subroutine writes input tables I49 through I55 on output file JINP. These tables describe characteristics of chemical strikes, target information, employment criteria, weapon system characteristics, etc.

2.2.4.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHOUT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	None
Called by	WTONE

2.2.4.3.2 Logic Functions. CHOUT uses formatted write statements and loops where appropriate to generate the following tables:

Tables	Heading
I49	Characteristics of a Preemptive Chemical Strike
I50	Priority Listings of Preferred Chemical Targets
I51	Index to Allowable Target Types (Chemical)
I52	Chemical Employment Criteria - Initial or Increased Use of Nuclear Weapons by Other Side
I53	Chemical Employment Criteria - Initial or Increased Use of Chemical Weapons by Other Side
I54	Weapon System Characteristics (Chemical)
I55	Chemical Weapon Systems, Agents, and Dissemination Modes

2.2.4.4 TACQOT. Subroutine TACQOT writes input tables I56 through I64 on file JINP. These tables display ground sensor and air carrier data, sensor detection rates and probabilities, TOE numbers and factors for sensors and carriers, and assignments and attrition rates for army and air force reconnaissance aircraft.

2.2.4.4.1 Programming Specifications. The following table summarizes the principal specifications to subroutine TACQOT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AL
Subroutines called	None
Called by	WTONE

2.2.4.4.2 Logic Functions. Subroutine TACQOT uses formatted write statements and loops where appropriate to generate the following tables:

Tables	Heading
I56	Ground Sensor Data--Operating Altitude, Index for Target Acquisition Equation Location
I57	Air Carrier Data--Velocity, Location, Distance to Target
I58	Detection Rates of Subunits by Air and Ground Sensors
I59	Army-Air Carrier and Sensor Data
I60	TOE Number and Factors for Ground Sensors and Army-Air Carriers
I61	Mission Assignments and Attrition Rates for Reconnaissance Aircraft
I62	Reconnaissance Carriers and Aircraft in Theater
I63	Reconnaissance Aircraft and Sensor Data
I64	Probability of Subunit Detection

2.2.5 LINKD. This subsection describes the routines in LINKD. These are the air combat model routines.

2.2.5.1 AIRMOD. AIRMOD controls air attrition calculations. During each cycle AIRMOD is called once to perform the following functions: initialize air variables, call attrition subroutines, and apply the air combat attrition results to SAM and aircraft inventories.

2.2.5.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine AIRMOD:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3
Subroutines called	ALLOCT, AIRATT, AOVLI, ATTR1, AOVL2, ATTR5, ATTR6, APORTN
Called by	TMAIN

2.2.5.1.2 Logic Functions. AIRMOD performs air combat calculations for air and SAM resources aggregated into regions. The code is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Initialize Air Combat Variables for Both Sides. Subroutine ALLOCT is called to calculate the number of aircraft on each notional airbase representing region forward, region rear and COMMZ airbases. ALLOCT also calculates the sortie assignments for all aircraft in the current cycle and the air munitions load factors for combat air support aircraft.

After ALLOCT has been called, AIRMOD calculates air-to-air kill probabilities. The input probability of kill by a side L type IAC defender of an enemy type KAC escort if they are engaged is PKDE(IAC,KAC,L). The following kill probabilities are obtained by multiplying each PKDE by input factors FAPKDA (L), FAPKAD(L) and FAPKED(L), respectively: PKDAS, the probability of kill by a defender of an enemy attacker or SAM suppressor; PKAD, the probability of kill by an attacker of an

enemy defender; and PKED, the probability of kill by an escort of an enemy defender. After the kill probabilities have been calculated, arrays for air combat results are initialized. Aircraft variable names specify the mission and combat status. For example, ABAERA(IAC) specifies the number of surviving (alive) escort (E) sorties of type IAC aircraft accompanying airbase attack aircraft to region-rear (R) airbases. The last letter of the variable name specifies combat status: A = alive and continuing on mission, D = aborting damaged, K = killed, H = aborting undamaged, S = suppressed (for SAMs).

b. Section 5700 - Air Combat Attrition Calculations for Side L Attackers vs. Side K Defenders. This section, except for write statements at the end, consists of a single Do loop which is executed twice-- once for L=1, K=2, then for L=2, K=1. The first part of this Do loop initializes and aggregates SAM resources into regions. Air combat attrition calculations are then made for each defending region. The following table lists the air attrition subroutines called by AIRMOD to perform those calculations:

Subroutine	Function
AIRATT	Initialize aircraft and SAM variables
AOVL1	Calculate results of engagements in forward area
ATTR1	Calculate results of additional engagements in forward area
AOVL2	Call ATTR2, ATTR3, and ATTR4
ATTR2	Calculate results of engagements between interdiction sorties and point defenses
ATTR3	Calculate results of engagements in rear area
ATTR4	Calculate results of engagements in the COMMZ
ATTR5	Calculate results of engagements involving close air support attack sorties

Subroutine	Function
ATTR6	Calculate attrition on way home from mission; convert final sortie results to numbers of aircraft

Table 2 in subsection 2.1.4 lists the variables calculated in these subroutines. The depletion of attacker and defender resources is cumulative from each subroutine section to the next. If the detail print flag IPRD is set equal to 1, engagement results are printed out at the end of each section. The above attrition subroutines perform attrition calculations using a single-engagement binomial attrition function.

After these attrition calculations, the numbers of SAMs in each region is updated to reflect attrition during the cycle. Array WDIV, the number of weapons remaining in each combat division, is equal to the number before attrition multiplied by the fraction of short range SAMs defending that unit which are still alive.

c. Section 5800 - Apply Air Combat Kills and Damages to Aircraft Inventories. The number of aircraft on each notionalized base in each region is calculated by subtracting those killed and damaged in air combat in this cycle. Damaged aircraft are added to the maintenance pool DAMPL(IAC,L). The aircraft numbers are deaggregated from regions back to sectors. Finally subroutine APORTN is called to allocate aircraft from notionalized airbases to actual airbases.

2.2.5.2 BINFAC. BINFAC calculates FK, an attrition factor that is used in ATRTDA to calculate the number of attacking aircraft sorties killed by defending aircraft.

2.2.5.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine BINFAC:

Characteristic	Specification
Formal parameters	S = Array giving the number of type I air defense aircraft in the active battle area of a given region

Characteristic	Specification
	D = Array giving the average probability that a type I air defense aircraft located in the combat region will detect an attacking aircraft
	P = Array P(I,J) giving the probability that a type I air defense aircraft will engage and kill a type J attacking aircraft, given that the attacking aircraft was detected
	TTC = Total number of attacking aircraft
	CA = Width of the region at the current FEBA divided by the width of the normal combat area in the region utilized by the penetrating aircraft
	NI = Number of different types of air defense aircraft
	NJ = Number of different types of attacking aircraft
	FK = Array specifying the single engagement binomial attrition factor calculated by this routine for each attacking aircraft type
Common blocks	None
Subroutines called	None
Called by	ATRTDA

2.2.5.2.2 Logic Functions. BINFAC calculates the single-engagement binomial attrition factor FK(J). The calling program ATRTDA calculates AK(J), the number of attacking aircraft of each type J killed by defending aircraft from AK(J) = AT(J)* (1-FK(J)). The factor FK(J) is calculated in BINFAC for each J according to the following equation:

$$FK(J) = \prod_{I=1}^{NI} \left[1 - \frac{P(I,J)}{TTC}\left(1-(1-D(I))^{TTC}\right)\right]^{\frac{S(I)}{CA}}$$

This equation aggregates the effects of the different types of defending aircraft. For a more detailed discussion of this equation see subsection 2.2.5.3.2.

2.2.5.3 BINOAT. Subroutine BINOAT is called by the air attrition subroutines. BINOAT calculates TK, the number of targets (aircraft or SAMs) killed when engaged by searchers (SAMs, defense aircraft, or penetrating aircraft).

2.2.5.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine BINOAT:

Characteristic	Specification
Formal parameters	S = Array giving the number of searchers of each type ISE
	T = Array giving the number of targets of each type ITE
	D = Array giving the average probability that a searcher of type ISE detects a target
	P = Array P(ISE,ITE) giving the probability that a type ISE searcher will engage and kill a type ITE target given that the target was detected.

Characteristic	Specification
	C = A factor used to specify the fraction of the total number targets that can be engaged
	NSE = Number of different types of searchers
	NTE = Number of different types of targets
	TK = Array giving the number of targets of type ITE killed in this engagement
Common blocks	None
Subroutines called	None
Called by	ATRTED, ATRTSA, ATRTSS, ATSPSS

2.2.5.3.2 Logic Functions. BINOAT calculates TK(ITE), the number of targets killed of each type ITE (ITE=1,2, ..NSE). A single-engagement binomial attrition equation is used. This equation aggregates the effects of the different types of searchers. A derivation of this equation can be found in reference 6.

The equation is based on the following assumptions:

a. At a fixed time all targets become vulnerable to detection and attack.

b. The probability that a searcher of type ISE detects a target is D(ISE). D(ISE) represents a detection probability which is averaged with respect to the numbers of targets of each type ITE.

c. From the targets (of all kinds) detected by a searcher, he chooses one to fire upon according to a uniform distribution.

d. Given that he detects and fires upon a target of type ITE, a searcher of type ISE destroys that target with probability P(ISE,ITE).

e. A searcher detects different targets independently of one another.

f. No searcher may fire at more than one target.

g. The detection and firing processes of all searchers are mutually independent.

The single-engagement binomial attrition factor TOT based on these assumptions is:

$$TOT = \prod_{ITE=1}^{NTE} \left[1 - \left\lfloor \frac{P(ISE,ITE)}{TTC} \right\rfloor \left\lfloor 1 - (1-D(ISE))^{TTC} \right\rfloor \right]^{\frac{S(ISE)}{C}}$$

where

$$TTC = \frac{1}{C} \sum_{ITE=1}^{NTE} T(ITE)$$

The number of targets killed is TK(ITE) = T(ITE) * (1-TOT).

2.2.5.4 ATSPSS. Subroutine ATSPSS calculates the attrition resulting from engagements between short-range SAMs and enemy attack or suppression aircraft.

2.2.5.4.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATSPSS:

Characteristic		Specification
Formal parameters	S	= Array giving the number of SAMs of each type alive and operating
	A	= Array giving the number of attacking aircraft of each type alive and continuing on mission

Characteristic	Specification
FSM	= Fraction of aircraft that use standoff munitions to avoid being shot at by SAMs
PDA	= Probability that an aircraft detects SAMs
PSA	= Probability of suppression of SAMs by aircraft
PKA	= Array giving the probability of kill of SAMs by aircraft
FKLA	= Fraction of kills that are lethal when an aircraft shoots at SAMs
ANM	= Actual number of SAM missiles in theater (including munitions for AAA)
PDS	= Probability that a SAM detects aircraft
PKS	= Probability that a SAM kills aircraft
FKLS	= Fraction of kills that are lethal when a SAM shoots at aircraft
CA	= Number of penetration corridors
NX	= Number of types of SAMs
NI	= Number of types of aircraft
SA	= Array specifying the number of SAMs alive and operating after this engagement

Characteristic	Specification
	SS = Array specifying the number of SAMs suppressed in this engagement
	SK = Array specifying the number of SAMs killed in this engagement
	SD = Array specifying the number of SAMs damaged but not killed in this engagement
	AA = Array specifying the number of aircraft alive and operating after this engagement
	AK = Array specifying the number of aircraft killed in this engagement
	AD = Array specifying the number of aircraft damaged but not killed in this engagement
Common blocks	None
Subroutines called	BINOAT
Called by	ATTR1, ATTR2, ATTR3, ATTR4, ATTR5

2.2.5.4.2 Logic Functions. Subroutine ATSPSS calculates the engagements between short-range SAMs and enemy attack or suppression aircraft. The numbers of SAMs killed, damaged, and suppressed are calculated by calling subroutine BINOAT, which uses a single-engagement binomial attrition equation. ATSPSS also calls BINOAT to calculate the number of aircraft killed and damaged by the SAMs. BINOAT calculates both aircraft and SAM kills using the number of aircraft and SAMs available at the start of the engagement. The actual number

of SAMs in the theater after this engagement, ANM, is equal to the number in the theater before this engagement minus the number fired at aircraft.

2.2.5.5 ATRTED. Subroutine ATRTED calculates the attrition resulting from engagements between defense aircraft and attack escort aircraft.

2.2.5.5.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATRTED:

Characteristic	Specification
Formal parameters	ET = Array specifying the number of attacking escort aircraft sorties of each type
	DT = Array specifying the number of defending aircraft sorties of each type
	PDE = Probability of detection by escort aircraft of defending aircraft
	PDD = Probability of detection by defending aircraft of escort aircraft
	PKE = Array specifying the probability of kill by escort of defending aircraft
	PKD = Array specifying the probability of kill by defending aircraft of escort aircraft
	FKLE = Fraction of kills that are lethal when attack escort shoots at defender

Characteristic	Specification
	FKLD = Fraction of kills that are lethal when defender shoots at escort aircraft
	CA = Fraction of the sector width occupied by the combat area
	N1 = Number of attack escort types
	N2 = Number of defending aircraft types
	AEE = Average number of additional engagements (in addition to 1) that an escort can make
	AED = Array specifying the number of additional engagements (in addition to 1) that a defender can make
	EA = Array specifying the number of escort sorties alive and continuing on their mission
	EK = Array specifying the number of escort sorties killed in this engagement
	ED = Array specifying the number of escort sorties damaged but not killed in this engagement
	EH = Array specifying the number of escort sorties aborting and returning home undamaged

Characteristic	Specification
	DA = Array specifying the number of defender sorties alive and continuing on their mission
	DK = Array specifying the number of defender sorties killed in this engagement
	DD = Array specifying the number of defender sorties damaged but not killed in this engagement
	DH = Array specifying the number of defender sorties aborting and returning home undamaged
Common block	None
Subroutines called	BINOAT
Called by	AOVL1, ATTR1, ATTR3, ATTR4

2.2.5.5.2 Logic Functions. Subroutine ATRTED calculates the results of engagements between attack escort aircraft and defending aircraft. This subroutine calculates the number of escort and defending aircraft sorties of each type that are alive, killed, damaged, and aborted after the engagement. The calculations are made using subroutine BINOAT which uses a single-engagement binomial attrition equation. The number of sorties of escort and defending aircraft aborting their mission and returning home undamaged is set equal to the number of those aircraft sorties detected by enemy aircraft minus the number killed and damaged.

2.2.5.6 ATRTSA. Subroutine ATRTSA calculates the attrition of penetrating aircraft (attack, escort, and suppression) as they fly by SAM defenses.

2.2.5.6.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATRTSA:

Characteristic		Specification
Formal parameters	S	= Array specifying the number of SAMs of each type alive and operating
	A	= Array specifying the number of penetrating aircraft sorties of each type
	PD	= Probability of detection of penetrating aircraft by SAMs
	PK	= Probability of kill by SAMs of enemy aircraft
	PARH	= Probability that an aircraft returns home when engaged by SAMs
	FKLSA	= Fraction of kills that are lethal when a SAM shoots at an aircraft
	ANM	= Actual number of missiles in the theater
	AVGSS	= Average number of possible shots per SAM fire control center
	CA	= Number of penetration corridors in region
	NTS	= Number of types of SAMs
	NTE	= Number of types of aircraft
	AA	= Array specifying the number of penetrating aircraft sorties alive and operating after this engagement

Characteristic	Specification
	AK = Array specifying the number of penetrating aircraft sorties killed in this engagement
	AH = Array specifying the number of penetrating aircraft sorties aborted and returning home undamaged before this engagement
	AD = Array specifying the number of penetrating aircraft sorties damaged but not killed in this engagement
Common blocks	None
Subroutines called	BINOAT
Called by	AOVL1, ATTR1, ATTR2, ATTR3, ATTR4, ATTR5

2.2.5.6.2 Logic Functions. Subroutine ATRTSA calculates the results of engagements between SAMs and fly-by-penetrating aircraft. Subroutine BINOAT is called to calculate the following quantities using a single-engagement binomial attrition equation: (1) the number of aircraft sorties of each type alive and continuing on their missions (AA), (2) the number of aircraft sorties aborting their missions and returning home (AH), (3) the number of aircraft sorties killed by the SAMs (AK), and (4) the number of aircraft sorties damaged but not killed (AD). The actual number of SAMs remaining in the theater after this engagement (ANM) is set equal to the number in the theater at the start of the engagement minus the SAMs fired at aircraft.

2.2.5.7 ATRTDA. Subroutine ATRTDA calculates the results of engagements between defending aircraft and penetrating attack and defense suppression aircraft.

2.2.5.7.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATRTDA:

Characteristic	Specification
Formal parameters	D = Array specifying the number of defending aircraft sorties of each type
	AT = Array specifying the number of attack aircraft sorties of each type
	SS = Array specifying the number of attacking suppression aircraft sorties of each type
	PD = Probability of detection by defending aircraft of an enemy aircraft flying by
	PKD = Array specifying the probability of kill by defender of attack aircraft if they are engaged
	PKS = Array specifying the probability of kill by suppressor of defender if they are engaged
	PAJO = Array specifying the probability that an attacker, when engaged, jettisons its ordnance and returns fire
	PSJO = Array specifying the probability that a suppressor, when engaged, jettisons its ordnance and returns fire

Characteristic	Specification
PKDNS	= Array specifying the probability of kill by defender of suppressor if they are engaged
FKLAS	= Fraction of kills that are lethal when an attacker or suppressor shoots at a defender
FKLD	= Fraction of kills that are lethal when a defender shoots at enemy aircraft
AED	= Average number of additional engagements (in addition to 1) that a defender can make
CA	= Fraction of the sector width occupied by the combat area
N1	= Number of attack and suppression aircraft types
N2	= Number of defender aircraft types
DA	= Array specifying the number of defender sorties alive and continuing on their mission
DK	= Array specifying the number of defender sorties killed in this engagement
DH	= Array specifying the number of defender sorties aborting and returning home undamaged

Characteristic	Specification
DD	= Array specifying the number of defender sorties damaged but not killed in this engagement
AA	= Array specifying the number of attack aircraft sorties alive and continuing on their mission
AK	= Array specifying the number of attack aircraft killed in this engagement
AH	= Array specifying the number of attack aircraft aborting and returning home undamaged
AD	= Array specifying the number of attack aircraft damaged but not killed in this engagement
SA	= Array specifying the number of suppression aircraft sorties alive and continuing on their mission
SK	= Array specifying the number of suppression aircraft sorties killed in this engagement
SH	= Array specifying the number of suppression aircraft sorties aborting and returning home undamaged

Characteristic	Specification
	SD = Array specifying the number of suppression aircraft sorties damaged but not killed in this engagement
Common blocks	None
Subroutines called	BINFAC
Called by	AOVL1, ATTR1, ATTR3, ATTR4, ATTR5

2.2.5.7.2 Logic Functions. Subroutine ATRTDA calculates the results of engagements between defending aircraft and penetrating enemy aircraft. The enemy penetrators are attack aircraft accompanied by defense suppression aircraft. The numbers of penetrating aircraft sorties killed, damaged, or aborted by defenders are calculated after calling subroutine BINFAC to calculate single-engagement binomial attrition factors. These factors are based on the assumption that each defender may detect several penetrators simultaneously, but can engage only one at a time. When detected, penetrators may jettison ordnance and return fire. The number of defender sorties killed in an engagement is obtained by multiplying the number of penetrators returning fire by the probability that the attacker kills the defender.

2.2.5.8 ATRTSS. Subroutine ATRTSS calculates aircraft and SAM attrition resulting from engagements between suppression aircraft and medium and long-range belt SAMs.

2.2.5.8.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATRTSS:

Characteristic	Specification
Formal parameters	S = Array specifying the number of SAMs alive and operating
	AVGSS = Average number of possible shots per SAM fire control center

Characteristic	Specification
A	= Array specifying the number of attacking suppression aircraft alive and continuing on mission
PDA	= Probability that a suppression aircraft detects SAMs
PSA	= Probability of suppression of SAMs by suppression aircraft
PKA	= Probability of kill of SAMs by suppression aircraft
FKLA	= Fraction of kills that are lethal when an aircraft shoots at SAM
ANM	= Actual number of SAM missiles in theater (including munitions for AAA)
PDS	= Probability that SAM detects SAM suppression aircraft
PKS	= Probability that SAM kills enemy aircraft
FKLS	= Fraction of kills that are lethal when SAM shoots at aircraft
CA	= Number of penetration corridors
NX	= Number of types of SAMs
NI	= Number of types of aircraft

Characteristic	Specification
	SA = Array specifying the number of SAMs alive and operating after this engagement
	SS = Array specifying the number of SAMs suppressed in this engagement
	SK = Array specifying the number of SAMs killed in this engagement
	SD = Array specifying the number of SAMs damaged but not killed in this engagement
	AA = Array specifying the number of suppression aircraft sorties alive and operating after this engagement
	AH = Array specifying the number of suppression aircraft sorties aborted and returning home undamaged after this engagement
	AK = Array specifying the number of suppression aircraft sorties killed in this engagement
	AD = Array specifying the number of suppression aircraft sorties damaged but not killed in this engagement
Common blocks	None
Subroutines called	BINOAT
Called by	AOVL1, ATTR1, ATTR3, ATTR4

2.2.5.8.2 Logic Functions. Subroutine ATRTSS calculates the results of engagements between attacking suppression aircraft and medium or long range belt SAMs. The calculations are made using subroutine BINOAT which uses a single engagement binomial attrition equation. First SK, SD and SS, which represent the number of SAMs killed, damaged and suppressed, respectively, are calculated. The number of SAMs alive after the engagement (SA) is set equal to the number killed, damaged and suppressed. The number of missiles in the theater after this engagement (ANM) is set equal to the number at the start of this engagement less those shot at the aircraft.

The number of missiles capable of firing at the start of the engagement is used to calculate the numbers of suppression aircraft sorties damaged (AD) and killed (AK). The number of aircraft sorties returning home undamaged after the engagement (AH) is set equal to the number of suppression aircraft sorties that detect SAMs minus the aircraft sorties killed or damaged. AA, the number of suppression aircraft sorties alive and continuing on their mission after this engagement, is set equal to the initial number (A) minus the sum of the number returning home undamaged, the number damaged and the number killed.

2.2.5.9 ALLOCT. Subroutine ALLOCT calculates air resource variables for use by attrition and assessment subroutines. ALLOCT first calculates the weighted number of target aircraft on each notional airbase. Assignments for aircraft based in the COMMZ are then computed. The next section computes assignments for aircraft from region rear airbases. The following section computes assignments for aircraft from region forward airbases. The number of sorties on each mission is calculated next. This is done after subroutine DEG has been called to compute the degradation of sortie capability. Finally, the last section computes air munition load factors for aircraft flying CAS missions. The aircraft assignments and sortie calculations are made to maximize the destruction of enemy resources. The equations used in ALLOCT result from the marginal allocation of aircraft to targets to obtain this maximum destruction. Marginal allocation is discussed in reference 2.

2.2.5.9.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ALLOCT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ
Subroutines called	DEG
Called by	AIRMOD

2.2.5.9.2 Logic Functions. Each section of ALLOCT is executed twice, once for Blue aircraft against Red targets and defenses and again for Red aircraft against Blue targets and defenses. The subroutine code is divided by comment cards which label the functions performed by the logic.

a. Section 100 - Compute Weighted Number of Target Aircraft on Each Notional Airbase. The air resources are represented by a notional airbase in each sector forward, a notional airbase in each sector rear and a notional airbase in the COMMZ. The weighted number of target aircraft is calculated for each of the notional bases in each sector, and then the weighted number of target aircraft on the notional COMMZ airbase is calculated. The calculations are made by adding the weighted numbers of aircraft on all of the bases in the sector forward (or rear or COMMZ) and dividing by the actual number of airbases in the sector forward (or rear or COMMZ).

The weighting procedure influences the subsequent assignment of airbase attack missions causing more firepower to be directed toward the unsheltered aircraft. Input weighting factors for side K include

WFCNSA(K) -- nonsheltered aircraft vs. sheltered aircraft

WFCQRA(K) -- quick reaction aircraft vs. nonquick reaction aircraft

WFCRA(K) -- rear based aircraft vs. forward based aircraft

WFCZA(K) -- COMMZ aircraft vs. forward based aircraft

b. Section 105 - Convert Sectors to Enemy Regions. The notional airbases in the sectors within each region are combined into a region forward and a region rear airbase. This aggregation reduces the number of allocations and sortie assignments required. The number of actual forward and rear enemy airbases ABASEF(IR) and ABASER(IR) in each region IR is obtained by summing the actual number of bases in the sectors in each region. The weighted numbers of aircraft WTAFS(IR) and WTARS(IR) in region IR are obtained by summing the weighted numbers of aircraft in the sectors of region IR.

c. Sections 110, 120, and 130 - Fractional Aircraft Assignments. These three sections are part of a single DO loop over each attacking aircraft type IAC. These sections compute the fraction of attacker aircraft from COMMZ, region rear or region forward bases assigned to each mission in each enemy region. Each section begins by initializing the fractions of aircraft allocated to each mission. Then the appropriate marginal allocation equations (selected by aircraft range) are used to calculate the fractions allocated to each mission. These fractions depend on input values of fractional assignments for various missions and the weighted number of target aircraft in each enemy region.

The fractions are stored in arrays specified by six-letter array names, e.g., FAIARF(IAC,IS,I). The third and fourth letters of the array name of each fraction specify the mission.

The following missions are specified:

AA	Air base attack (ABA)
AD	Area defense
AE	ABA escort
AS	Sam suppression for ABA missions
BS	Belt SAM suppression (BSSUP)
CA	Close air support attack (CASA)
CD	Battlefield defense
CE	CAS escort
CS	CAS SAM suppression
IA	Interdiction attack (INTDA)
IE	INTDA escort
IS	INTDA SAM suppression

The fifth letter of the array name specifies the base location of the assigned aircraft: F=region forward, R=rear, Z=COMMZ. The sixth letter specifies the enemy region to

which these aircraft are assigned. Thus FAIARF(IAC,IS,I) specifies the fraction of side I type IAC aircraft assigned from rear bases in region IS to fly INTDA missions into the enemy forward region.

Section 110 computes COMMZ-based aircraft assignments, section 120 computes rear-based aircraft assignments, and section 130 computes forward-based aircraft assignments.

d. Section 140 - Compute Number of Sorties on Each Mission. Subroutine DEG is called to compute degraded sortie rate capabilities for both Red and Blue forces by considering supply shortages or loss of other operating capabilities at airbases. The remainder of Section 140 is then executed twice, once for Red attackers and once for Blue attackers.

The number of aircraft available in each region is calculated by applying the degradations calculated by DEG to the region forward, region rear, and COMMZ airbases. Then the number of aircraft sorties of each aircraft type assigned to each mission in each enemy region is calculated. The number of sorties are calculated by summing the product of the fractional aircraft assignments, the number of aircraft available in each region, and the 12-hour sortie rate.

e. Section 150 - Compute Air Munitions Load Factors for CAS Mission. The air munition load per sortie depends on the distance flown from base to target. This section calculates AMLFD(IAC,IS,L), the average air munition load factor for each side L close-air-support attack aircraft of type IAC flying to targets in enemy sector IS.

AMLFD(IAC,IS,L)

$$= \frac{1}{NS(K)} * \left[\frac{TEMP + TEMP1*(AMLFR(IAC,L)) + TEMP2*(AMLFZ(IAC,L))}{TEMP + TEMP1 + TEMP2} \right]$$

where

TEMP = Number of side L CASA aircraft of type IAC based on region forward bases

TEMP1 = Number of side L CASA aircraft of type IAC based on region rear bases

TEMP2 = Number of side L CASA aircraft of type IAC based on COMMZ bases

AMLFR(IAC,L) = Air munition load factor (input) for rear-based aircraft

AMLFZ(IAC,L) = Air munition load factor (input) for COMMZ-based aircraft

NS(K) = Number of enemy (side K) sectors

2.2.5.10 DEG. Subroutine DEG determines the current operating capability and supply level at each notional airbase.

2.2.5.10.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DEG:

Characteristic	Specification
Formal parameters	DEGSRF = Array specifying the degraded sortie rate capability of sector forward airbases
	DEGSRR = Array specifying the degraded sortie rate capability of sector rear airbases
	DEGSRC = Array specifying the degraded sortie rate capability of COMMZ airbases
Common blocks	Blank Common, CCCC1, CCC2, CZZZ, ZZ
Subroutines called	CLR, TAG, CVFW
Called by	ALLOCT

2.2.5.10.2 Logic Functions. The sortie rate capability at an airbase can be degraded by destruction resulting from airbase attacks or by inadequate supply levels. DEG calculates the average capability (before degradation) for bases in each sector, applies the current degradations, and converts the

degraded sortie rate capabilities from sectors to regions. The following paragraphs describe the three sections of code which perform these calculations.

a. Section 100 - Operational Capability Before Current Degradation. After variables have been initialized, the current operational capability (before degradation) of each actual base is obtained by adding to the operating capability the fraction of destroyed capability which has been recovered since the previous cycle. The initial degraded sortie rate capability is calculated for the notionalized base in each sector by summing the operational capabilities of all of the actual bases in the sector and then dividing by the number of actual bases in the sector.

b. Section 200 - Apply Current Degradations. This section reduces the degraded sortie rate capability for the current cycle. Two reductions are considered: supply shortage and airbase capability destroyed by enemy air attacks. The degraded sortie rate capability for each sector is set equal to the smaller of these two degradations. The degraded sortie rate capability due to shortage of supplies is the ratio of the tons of supplies available to tons of supplies required by all of the aircraft at a base. The degraded sortie rate capability after enemy attack is the degraded sortie rate capability calculated in the first section of this subroutine multiplied by the fraction of base capability surviving the attack as determined by an exponential damage function.

c. Section 300 - Convert Degradation Factors From Sectors to Regions. This section converts the degraded sortie rate capabilities from sectors to regions in the following way: The number of aircraft on each sector forward (rear) base of the region is multiplied by the degraded sortie rate capability of the notionalized base in that sector forward (rear) to obtain a degraded number of aircraft. The degraded sortie rate capability of the region forward (rear) notionalized base is the sum of the degraded numbers of aircraft for all sectors on the region forward (rear) bases divided by the total number of undegraded aircraft based on region forward (rear) bases.

2.2.5.11 AIRATT. Subroutine AIRATT initializes the numbers of aircraft and SAMs alive, damaged and killed.

2.2.5.11.1 Programming Specifications. The following table summarizes the principal specifications of subroutine AIRATT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11 CCCC1, CCC2, CZZZ, CCC3, CCC3A, CCC4, ZZ, Z2, Z3
Subroutines called	None
Called by	AIRMOD

2.2.5.11.2 Logic Functions. AIRATT is called by AIRMOD to initialize the numbers of aircraft and SAMs alive, damaged and killed before the attrition routines are called by AIRMOD. AIRATT is called once for each region IS. Side L attacks side K. The first section of AIRATT calculates the total number of aircraft sorties in the region assigned to each attack and defense mission. The second section calculates the number of SAMs that are operating in the region.

a. Section 100 - Aircraft Calculations. In this section, the number of alive attacking aircraft sorties of each type IAC is set equal to the value calculated in subroutine ALLOCT; for example, ABAAFA(IAC) = ABAFA(IAC,IS,L). These numbers of alive attacking aircraft sorties are also added to the cumulative totals (e.g., CAAFSK(IAC,1,L)) for all regions. The numbers of aircraft killed, aborted, and damaged in the current cycle are set equal to zero.

The last half of this section initializes similar variables for the defending aircraft of each type KAC on side K. FWPC, the fraction of defenders able to engage attackers, is calculated as follows:

FWPC = (Width of penetration corridor + lateral range of defender)*(number of penetration corridors)/(width of region)

The number of alive area defense sorties capable of engaging attackers (e.g., ABADGA(KAC)) is equal to the number of defense sorties as calculated in ALLOCT multiplied by FWPC.

b. Section 200 - SAM Calculations. This section of AIRATT initializes the number of SAMs of each type alive, killed, damaged, and suppressed. Calculations are made for

short-range SAMs (or AAA) defending airbases, then for medium-range belt SAMs and finally for long-range SAMs providing area defense in front of rear and COMMZ airbases.

For short-range SAMs, the number of alive SAMs is set equal to the number of SAMs multiplied by the fraction of actual airbases under attack. The number of alive medium- and long-range SAMs capable of engaging attackers is equal to the number of those SAMs multiplied by a fraction FWPC, calculated in the same manner as for aircraft.

2.2.5.12 AOVL1. Program AOVL1 calculates the results of engagements between penetrating aircraft and defense aircraft and SAMs in the forward area of the theater. Calculations are made by calling the appropriate attrition subroutines.

2.2.5.12.1 Programming Specifications. The following table summarizes the principal specifications of AOVL1:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, CCC3, CCC3A, CCC4, Z2, Z3, ZZ
Subroutines called	ATRTED, ATRTSA, ATRTDA, ATRTSS
Called by	AIRMOD

2.2.5.12.2 Logic Functions. Subroutine AOVL1 calculates aircraft sortie and SAM attrition in the forward area of the theater. The code is divided by comment cards which label the functions performed by the logic.

a. Section 100 - Combat Air Support Escort Vs. Battlefield Defense Aircraft. The results of engagements between escort (CASE) and defense (CASD) aircraft are calculated using subroutine ATRTED.

b. Section 200 - Fly-by Penetrators Vs. Short-Range SAMs Defending Combat Units. The attrition of penetrators (ABAAF, ABAEF, ABASF, ABAAR, ARAER, ABASR, ABAAZ, ABAEZ, ABASZ, INTDA, INTDE, INTDS, BSSUP) as they fly past short-range SAMs (PSRSC) defending combat units is calculated using subroutine ATRTSA.

c. Section 300 - Belt SAM Suppression Aircraft Vs. Battlefield Defense Aircraft. The results of engagements between suppression (BSSUP) and CASD aircraft are calculated using subroutine ATRTDA.

d. Section 400 - Belt SAM Suppression Aircraft Vs. Medium-Range Belt SAMs. The results of engagements between SAM suppression aircraft (BSSUP) and medium-range belt SAMs (BMRS) are calculated using subroutine ATRTSS.

e. Section 500 - Attack Escorts Vs. Battlefield Defense Aircraft. The results of engagements between defense (CASD) and attack escort aircraft (ABAEF, ABAER, ABAEZ, INTDE) are calculated using subroutine ATRTED.

f. Section 600 - Attack and Suppression Aircraft Vs. Battlefield Defense Aircraft. ATRTDA is used to calculate results of engagements between defense (CASD) aircraft, and airbase and interdiction attack and suppression aircraft (ABAAF, ABAAR, ABAAZ, ABASF, ABASR, ABASZ, INTDA, INTDS).

At the end of each section of AOVL1, the number of attackers and defenders alive and able to continue on their missions is recomputed to reflect the attrition calculated in that section. If the detail print flag IPRD=1, engagement results are printed out at the end of each section. Additional calculations involving engagements between penetrators and forward area SAM (BMRS and PSRSF) and aircraft (ABADF) defenses are made in subroutine ATTR1. Engagement results for CASD aircraft vs. CASA and CASS penetrators are calculated in ATTR5.

2.2.5.13 ATTR1. ATTR1 calculates additional results of engagements between penetrating aircraft and defending aircraft and SAMs in the forward area of the theater. The number of successful airbase attack (ABA) sorties to forward airbases is calculated.

2.2.5.13.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATTR1:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, CCC3, CCC3A, CCC4, Z2, Z3, ZZ

Characteristic	Specification
Subroutines called	ATRTSS, ATRTED, ATRTDA, ATSPSS, ATRTSA
Called by	AIRMOD

2.2.5.13.2 Logic Functions. Subroutine ATTR1 calculates aircraft and SAM attrition in the forward area of the theater. Calculations are made as described below.

a. Section 700 - Suppression Aircraft Vs. Medium-Range Belt SAMs. The results of engagements between penetrating suppression aircraft (ABASF, ABASR, ABASZ, INTDS) and medium-range belt SAMs (BMRSA) are calculated using subroutine ATRTSS.

b. Section 800 - Fly-by Penetrators Vs. Medium-Range Belt SAMs. The attrition of attack (ABAAF, ABAAR, ABAAZ, INTDA) and escort (ABAEF, ABAER, ABAEZ, INTDE) aircraft as they fly past the medium-range belt SAMs (BMRSA) is calculated using subroutine ATRTSA.

c. Section 900 - Attack Escort Aircraft Vs. Defending Aircraft in the Forward Area. Subroutine ATRTED is called to calculate the results of engagements between defense aircraft assigned to forward regions (ABADF) and attack escort aircraft (ABAEF, ABAER, ABAEZ, INTDA).

d. Section 1000 - Attack and Suppression Aircraft Vs. Defending Aircraft in the Forward Area. Subroutine ATRTDA is called to calculate the results of engagements between ABADF defense aircraft, and attack (ABAAF, ABAAR, ABAAZ, INTDA), and suppression (ABASF, ABASR, ABASZ, INTDS) aircraft.

The remainder of ATTR1 addresses the attack of airbases in the forward region.

e. Section 1600 - Suppression Aircraft Vs. Forward Airbase Point Defenses. Subroutine ATSPSS is called to calculate the results of engagements between the short-range SAMs (or AAA) providing point defenses for forward airbases (PSRSFA) and suppression aircraft assigned to forward airbases (ABASF).

f. Section 1800 - Airbase Attack Aircraft Vs. Forward Airbase Point Defenses. Subroutine ATRTSA is called to calculate the attrition of the forward airbase attack aircraft (ABAAF) when they encounter the surviving point defenses (PSRSFA).

g. Section 1900 - Accumulate Results for Forward Area. This section sums up the results. CSABAF(IAC,L) is the cumulative number of successful side L ABA sorties of type IAC against side K forward airbases. SABAF(IAC,IS,L) equals the number of successful sorties against forward airbases in region IS. The number of ABAAF aircraft sorties alive (ABAAFA(IAC)) is set equal to the number alive at the target plus the total number that have aborted and returned home undamaged.

At the end of each section of ATTR1 the numbers of attackers and defenders alive and able to continue on their missions are recomputed to reflect the attrition calculated in that section. If the detail print flag IPRD=1, engagement results are printed out at the end of each section.

2.2.5.14 AOVL2. AOVL2 is a control program which calls attrition subroutines ATTR2, ATTR3, and ATTR4.

2.2.5.14.1 Programming Specifications. The following table summarizes the principal specifications of program AOVL2:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3
Subroutines called	ATTR2, ATTR3, ATTR4
Called by	AIRMOD

2.2.5.14.2 Logic Functions. The AOVL2 program consists of three call statements. ATTR2 is called to calculate attrition resulting from engagements between interdiction aircraft and opposing ground defenses. ATTR3 is called to calculate aircraft and SAM attrition in the rear area of the theater. ATTR4 is called to calculate aircraft and SAM attrition in the COMMZ.

2.2.5.15 ATTR2. Subroutine ATTR2 calculates the results of interactions between interdiction aircraft and opposing ground defenses prior to these aircraft delivering their ordnance on interdiction targets.

2.2.5.15.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATTR2:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3
Subroutine called	ATSPSS, ATRTSA
Called by	AOVL2

2.2.5.15.2 Logic Functions. The ATTR2 subroutine calculates the results of engagements between interdiction attack (INTDA) and suppression (INTDS) aircraft on side L, and the opposing ground point defenses (PSRSIA) in each sector IST of the enemy region. The input variable FINTRS(IST,L) specifies the fraction of side L aircraft on interdiction missions into each enemy region that are sent to sector IST of that region. The code is divided by comment cards which label the functions performed by the logic.

a. Section 2100 - Interdiction Suppression Aircraft Vs. Point Defenses. In this section, subroutine ATSPSS is called to calculate the results of engagements between interdiction suppression aircraft sorties and opposing SAM or AAA point defenses defending combat divisions in the first inactive battle area of each sector.

b. Section 2300 - Interdiction Attack Aircraft Vs. Point Defenses. Subroutine ATRTSA is called to calculate the attrition of interdiction attack aircraft sorties when engaged by the surviving point defenses PSRSIA. After this calculation, the point defenses remaining for future engagements is set equal to the number alive after this engagement plus the number suppressed. The input variable PIAIM(IAC,2,L) specifies the fraction of interdiction aircraft sorties assigned to attack combat divisions in the inactive battle area.

c. Section 2400 - Accumulate Results for Interdiction Missions. This section sums up several results. SINTDA(IAC, IST,L), the number of successful interdiction attack sorties in sector IST, is set equal to the number of interdiction attack sorties from side L of type IAC alive after engaging the point defenses. CSINDA(IAC,L) is set to the cumulative number of successful side L type IAC interdiction attack sorties. The number of interdiction attack and suppression sorties remaining alive (INTDSA(IAC) and INTDSS(IAC)) is set equal to the sum of the number of successful sorties plus the number aborting undamaged.

2.2.5.16 ATTR3. Subroutine ATTR3 calculates the results of engagements between penetrating aircraft and opposing ground and air defenses in the rear area of the theater.

2.2.5.16.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATTR3:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3
Subroutines called	ATRTED, ATRTDA, ATRTSS, ATRTSA, ATSPSS
Called by	AOVL2

2.2.5.16.2 Logic Functions. Subroutine ATTR3 calculates the results of engagements between penetrating aircraft flying to rear or COMMZ areas and aircraft and SAM defenses in the rear area of the theater.

a. Section 2500 - Escort Aircraft Vs. Rear Defending Aircraft. Subroutine ATRTED is called to calculate the results of engagements between escort aircraft (ABAER and ABAEZ) and defending aircraft (ABADR) in the rear area of the theater.

b. Section 2600 - Airbase Attack and Suppression Aircraft Vs. Rear Defending Aircraft. Subroutine ATRTDA is called to calculate the results of engagements between airbase attack (ABAAR, ABAAZ) and suppression aircraft (ABASR, ABASZ), and defending aircraft (ABADR) in the rear area of the theater.

c. Section 2700 - Suppression Aircraft Vs. Belt SAMs in the Rear Area. Subroutine ATRTSS is called to calculate aircraft and SAM attrition resulting from engagements between suppression aircraft (ABASR, ABASZ) and SAMs (ALRSR) providing area defense in the rear area of the theater.

d. Section 2900 - Attack and Escort Aircraft Vs. Belt SAMs in the Rear Area. Subroutine ATRTSA is called to calculate the attrition of attack (ABAAR, ABAAZ) and COMMZ attack escort (ABAEZ) aircraft when engaged by SAMs providing area defense in the rear area of the theater.

e. Section 3200 - Suppression Aircraft Vs. Rear Airbase Point Defenses. Subroutine ATSPSS is called to calculate the results of engagements between suppression aircraft (ABASR) accompanying rear airbase attackers and SAMs (PSRSR) providing point defense for rear airbases.

f. Section 3400 - Airbase Attack Aircraft Vs. Rear Airbase Point Defenses. Subroutine ATRTSA is called to calculate the attrition of aircraft attacking airbases in the rear area of the theater (ABAAR) when engaged by SAMs (PSRSR) providing point defense for those airbases.

g. Section 3500 - Accumulate Results for Rear Airbase Attacks. The number of successful airbase attack sorties of type IAC for side L to rear airbases in region IS (SABAR (IAC,IS,L)) is set equal to the number of attack aircraft sorties surviving engagements with the point defenses. CSABAR(IAC,L) is the cumulative number of successful airbase attack sorties to rear airbases. The number of rear airbase attack aircraft sorties alive after these engagements is set equal to the number of successful sorties plus the number of sorties aborting undamaged.

2.2.5.17 ATTR4. Subroutine ATTR4 calculates the results of engagements between penetrating aircraft and opposing ground and air defenses in the COMMZ.

2.2.5.17.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATTR4:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3

Characteristic	Specification
Subroutines called	ATRTED, ATRTDA, ATRTSS, ATRTSA, ATSPSS
Called by	AOVL2

2.2.5.17.2 Logic Functions. The ATTR4 subroutine calculates the results of engagements between penetrators flying to the COMMZ, and aircraft and SAM defenses in the COMMZ.

a. Section 3600 - Escort Aircraft Vs. COMMZ Defending Aircraft. Subroutine ATRTED is called to calculate the results of engagements between escort aircraft (ABAEZ) and defending aircraft (ABADZ) in the COMMZ.

b. Section 3700 - Airbase Attack and Suppression Aircraft Vs. COMMZ Defending Aircraft. In this section subroutine ATRTDA is called to calculate the results of engagements between airbase attack (ABAAZ) and suppression (ABASZ) aircraft, and defending aircraft (ABADZ) in the COMMZ.

c. Section 3800 - Suppression Aircraft Vs. Belt SAMs in COMMZ. Subroutine ATRTSS is called to calculate aircraft and SAM attrition resulting from engagements between suppression aircraft (ABASZ) and SAMs (ALRSZ) providing area defense in the COMMZ.

d. Section 4000 - Attack Aircraft Vs. Belt SAMs in the COMMZ. Subroutine ATRTSA is called to calculate the attrition of airbase attack (ABAAZ) aircraft when engaged by SAMs (ALRSZ) providing area defense in the COMMZ.

e. Section 4300 - Suppression Aircraft Vs. COMMZ Airbase Point Defenses. Subroutine ATSPSS is called to calculate the results of engagements between suppression aircraft (ABASZ) accompanying COMMZ airbase attackers and SAMs (PSRSZ) providing point defense for COMMZ bases.

f. Section 4500 - Airbase Attack Aircraft Vs. COMMZ Airbase Point Defenses. Subroutine ATRTSA is called to calculate the attrition of aircraft attacking airbases in the COMMZ when engaged by SAMs (PSRSZ) providing point defense for COMMZ bases.

g. Section 4600 - Accumulate Results for COMMZ Airbase Attacks. The number of successful airbase attack sorties of type IAC for side L to COMMZ airbases (SABAZ(IAC,L)) is set equal to the number of attack aircraft sorties surviving engagements with the point defenses. CSABAZ(IAC,L) is the cumulative number of successful airbase attack sorties to COMMZ airbases. The number of COMMZ airbase attack aircraft sorties alive after these engagements is set equal to the number of successful sorties plus the number of sorties aborting undamaged.

2.2.5.18 ATTR5. Subroutine ATTR5 calculates results of engagements between close-air-support (CAS) attack and suppression aircraft and opposing aircraft and point defenses.

2.2.5.18.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATTR5:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3
Subroutines called	ATRTDA, ATSPSS, ATRTSA
Called by	AIRMOD

2.2.5.18.2 Logic Functions. Subroutine ATTR5 calculates the results of engagements between CAS aircraft and battlefield defenses.

The capability of the defense was degraded earlier in the cycle by attrition calculations in program AOVL1.

a. Section 4800 - Attack and Suppression CAS Aircraft Vs. Battlefield Defense Aircraft. Subroutine ATRTDA is called to calculate the results of engagements between penetrating CAS attack (CASA) escorted by suppression aircraft (CASS) and enemy battlefield defense aircraft (CASD).

b. Section 5000 - Suppression Aircraft Vs. Battlefield Point Defenses. In this section subroutine ATSPSS is called to calculate the results of engagements between suppression aircraft (CASS) and SAMs or AAA (PSRSC) providing point defense for combat units.

c. Section 5200 - Attack Aircraft Vs. Battlefield Point Defenses. Subroutine ATRTSA is called to calculate the attrition of close air support attack (CASA) aircraft when engaged by (PSRSC) point defenses for combat units.

d. Section 5300 - Accumulate Results for CAS Attacks. The number of successful CAS attack sorties of type IAC for side L in sector IST (ACSABA(IAC,IST,L)) is set equal to the number of attack aircraft surviving engagements with the point defenses. CSCASA(IAC,L) is the cumulative number of successful CAS attack sorties. The number of CAS aircraft sorties alive after these engagements is set equal to the number of successful sorties plus the number of sorties aborting undamaged.

2.2.5.19 ATTR6. Subroutine ATTR6 calculates the attrition of aircraft on the way home (outbound) from their missions. The cumulative results of all attrition calculations for this cycle are converted from sorties to numbers of aircraft. Finally these attrition results are expressed as the number of aircraft in each basing region.

2.2.5.19.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATTR6:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CCC1, CC11, CCCC1, CCC2, CZZZ, ZZ, CCC3, CCC3A, CCC4, Z2, Z3
Subroutines called	ATRTWH
Called by	AIRMOD

2.2.5.19.2 Logic Functions. Subroutine ATTR6 is the last attrition subroutine called by subroutine AIRMOD during each cycle. The code is divided by comment cards which label the functions performed by the logic.

a. Section 5400 - Attrition on the Way Home. Subroutine ATRTWH is called for each aircraft sortie type to calculate the attrition of aircraft as they return to their home bases. Attrition is calculated for aircraft sorties returning home after aborting their missions as well as for

aircraft which have successfully completed their missions. At the conclusion of these calculations, the numbers of sorties of each aircraft type alive, damaged, and killed reflect the results of all air engagements for the current cycle.

b. Section 5500 - Convert Sorties to Numbers of Aircraft. All of the air attrition calculations have been made in terms of sorties. This section calculates the number of side L aircraft of each type IAC alive, killed, and damaged during this cycle by dividing the number of sorties alive, damaged, and killed by the 12-hour sortie rate (SRACM(IAC,J,L)) for each mission type J.

c. Section 6500 - Allocate Aircraft Killed and Damaged in Air Combat to Airbases. The six arrays calculated in this section are as follows:

Array Name	Description
ACFKC	Aircraft from forward bases killed
ACRSKC	Aircraft from rear bases killed
ACCZKC	Aircraft from COMMZ bases killed
ACFSDC	Aircraft from forward bases damaged
ACRSDC	Aircraft from rear bases damaged
ACCZDC	Aircraft from COMMZ bases damaged

For example, ACFSKC(IAC,KS,L) is the total number of side L type IAC aircraft based on forward bases in region KS that are killed in air combat during this cycle. ACFSKC is calculated from

$$ACFSKC = ACFS * TEMP$$

where

ACFS(IAC,KS,L) = Number of side L non-QRA aircraft of type IAC on forward airbase in region KS

and

$$TEMP = \sum_{\text{All mission types } J} SRACM*F*AK/A$$

where, for type IAC aircraft on side L,

SRACM = 12-hour sortie rate

F = fraction of total sorties assigned to mission J

AK = number of sorties killed on mission J

A = total number of sorties on mission J

The other five arrays are calculated using equations of this same form.

2.2.5.20 ATRTWH. Subroutine ATRTWH adds the numbers of attacking aircraft sorties killed and damaged on their way home to the numbers of aircraft sorties killed and damaged while on the way to their targets.

2.2.5.20.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATRTWH:

Characteristic	Specification
Formal parameters	AA = Array specifying the number of attacking aircraft sorties of each type alive and continuing on their mission
	AD = Array specifying the number of attacking aircraft sorties damaged in this engagement
	AK = Array specifying the number of attacking aircraft sorties killed in this engagement
	FAC = Factor for computing the number of aircraft killed on their way home
	NX = Number of types of attacking aircraft

Characteristic	Specification
Common blocks	None
Subroutine called	None
Called by	ATTR6

2.2.5.20.2 Logic Functions. Subroutine ATRTWH calculates the number of attacking aircraft sorties of each type IAC killed and damaged while returning home from their targets. The calculations assume that attrition while returning home is proportional to attrition suffered while flying to the targets. Thus PK, the probability of being killed while returning home, is:

$$PK = \left(\frac{AK(IAC)}{AA(IAC) + AD(IAC) + AK(IAC)} \right) * FAC$$

where AD(IAC) and AK(IAC) are the numbers of aircraft sorties damaged and killed, respectively, on the way to their targets and AA(IAC) is the number of aircraft sorties alive at their targets. It should be noted that in all the routines that call ATRTWH, the number of aircraft sorties aborting undamaged AH(IAC) is added to AA(IAC) before ATRTWH is called. Similarly PD, the probability of being damaged while returning home, is:

$$PD = \left(\frac{AD(IAC)}{AA(IAC) + AD(IAC) + AK(IAC)} \right) * FAC$$

The probability of being killed or damaged is: PDK=PK + PD. The number of aircraft sorties returning home undamaged AA'(IAC) is equal to the number alive at their targets minus the number killed or damaged while returning home:

$$AA'(IAC)=AA(IAC)(1-PDK)$$

The following calculations of the total aircraft sorties killed and damaged are made assuming that an aircraft damaged on the way to target is killed if it is either damaged again or killed while returning home.

The total number of aircraft sorties killed AK'(IAC) is:

$$AK'(IAC)=AK(IAC)+AA(IAC)*PK+AD(IAC)*PDK$$

where

AK(IAC) = Number of aircraft sorties killed on their way to target

AA(IAC)*PK = Number of aircraft sorties reaching their targets that are killed while returning home

AD(IAC)*PDK = Number of aircraft sorties damaged on their way to target that are killed while returning home

Similarly, the total number of aircraft sorties returning home damaged is:

$$AD'(IAC)=AD(IAC)+AA(IAC)*PD-AD(IAC)*PDK.$$

In subroutine ATRTWH, the calling variables AA, AD, and AK are returned with the values of AA'(IAC), AD'(IAC), and AK'(IAC), respectively.

2.2.6 LINKE. This subsection describes the routines in LINKE. These routines comprise the nuclear combat model.

2.2.6.1 NUC. Subroutine NUC is the main calling program for the nuclear routines.

2.2.6.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	NUC1, NUC2, NUC3, NUC4, NUC5, NUC6, CLR
Called by	TMAIN

2.2.6.1.2 Logic Functions. If KFLAG≠0, subroutine NUC1 is called to determine nuclear escalation states, the number of nuclear weapon systems, and to allocate supplies of nuclear warheads to division, sector and theater weapon systems. If KFLAG=0, the call to NUC1 is skipped. If, upon return from NUC1, KFLAG=1 or 2, the rest of the nuclear model is skipped and control is returned to TMAIN. Otherwise, the rest of NUC is executed as described below. Subroutine CLR is called three times to initialize certain arrays that are used later in the cycle to accumulate weapon usage. Subroutine NUC2 is called to construct priority lists of nuclear weapons and targets. If, upon return from NUC2, KFLAG=1, the rest of NUC is skipped. Otherwise NUC3 is called to determine the expected number of battlefield targets (subunits) detected. If variable IPOPCH≠0, subroutine NUC4 is called to determine the fraction of possible targets precluded from targeting due to civilian population centers. The weapon to target assignments are completed by subroutine NUC5, which is called next. Finally, NUC6 is called to calculate the damage inflicted on the targets.

2.2.6.2 BLKDA. Program BLKDA assigns values to the data statements described below.

2.2.6.2.1 Programming Specifications. The following table summarizes the principal specifications of program BLKDA:

Characteristic	Specification
Formal parameters	None
Common blocks	AFSTF, AFSTF2
Subroutines called	None
Called by	N/A

2.2.6.2.2 Logic Functions. This program is a block data routine and, therefore, contains no executable code. Array AFDIM is assigned specific data values which represent size dimensions for a given airbase. Array IAFBA is assigned index values representing the battle area location of each airbase. Finally, array IAFWDI is assigned values that pertain to latitude and longitude for each airbase.

2.2.6.3 KCDEN. Function KCDEN packs the weapon data listed below into a single word.

2.2.6.3.1 Programming Specifications. The following table summarizes the principal specifications of function KCDEN:

Characteristic	Specification
Formal parameters	IWC = Index to weapon category
	IWS = Index to weapon system
	IPOS = Index to weapon system position
	IYL = Index to yield
Common blocks	None
Subroutines called	None
Called by	NUCWPS

2.2.6.3.2 Logic Functions. The parameters specified above are packed into a 4-digit word from left to right as follows. First, parameter IWC is multiplied by the third power of ten (1000) in order to pack the fourth or left most digit into the word. Added to this number is the next parameter, IWS, multiplied by the second power of ten (100) in

order to pack the third digit. The remaining parameters are multiplied by the first power of ten (10) and the zeroth power of ten (1), and then added together to pack the second and first digits.

2.2.6.4 KDCDEN. Function KDCDEN unpacks the weapon data listed below from the specified value of INDEX.

2.2.6.4.1 Programming Specifications. The following table summarizes the principal specifications of function KDCDEN:

Characteristic	Specification
Formal parameters	INDEX = Word containing packed data
	IWC = Index to weapon category
	IWS = Index to weapon system
	IPOS = Index to weapon system position
	IYL = Index to yield
Common blocks	None
Subroutines called	None
Called by	PREYLD, DWHINV, NUCWPS, NUC6

2.2.6.4.2 Logic Functions. The 4-digit representation of the packed data contained in INDEX is unpacked from left to right in the following manner. First, INDEX is divided by the third power of ten (1000) in order to unpack the fourth, or leftmost digit. This value is the index to the weapon category and is stored in variable IWC. Now, IWC is multiplied by the same power of ten and the product is subtracted from INDEX. This new 3-digit value of INDEX is divided by the second power of ten (100) in order to unpack the third digit, which is the index to the weapon system, and is stored in variable IWS. The remaining parameters, located in the second and first digits, are unpacked in the same manner.

2.2.6.5 NUC1. Subroutine NUC1 is the main calling program for the routines that determine nuclear escalation states,

determine the number of nuclear weapon systems and allocate nuclear warheads to supply pools.

2.2.6.5.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC1:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAA1, LOCAL1, AFSTF2
Subroutines called	ESCLAT, NDSYINV, WHINUP
Called by	NUC

2.2.6.5.2 Logic Functions. NUC1 first calls subroutine ESCLAT to determine nuclear escalation states: If KFLAG≠2, then the remainder of NUC1 is executed. If the index to the escalation state in a given sector against a given target category for a given side is ≠ 0, a call is made to subroutine NDSYINV and to subroutine WHINUP. The former determines the number of nuclear weapon systems and the latter allocates supplies of nuclear warheads to supply pools.

2.2.6.6 ESCLAT. Subroutine ESCLAT determines the state of escalation for each side according to current conditions and the tactical nuclear escalation doctrine selected by the user.

2.2.6.6.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ESCLAT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL1, LOCAA1, AFSTF2
Subroutines called	None
Called by	NUC1

2.2.6.6.2 Logic Functions. The calculations of this routine are made for each sector IS and each side L. Each of the

following four escalatory stimuli may be a criterion for upgrading the escalation state: (1) a preplanned decision to make a preemptive strike; (2) a response to a worsening tactical environment; (3) a response to the enemy's initial or increased use of nuclear weapons; and (4) a response to the enemy's use of chemical weapons. For each criterion, the program first checks to see if the user has selected the criterion and then, if necessary, determines if conditions require an increase in the escalation state.

a. Section 10 - Preemptive Strike Considered. If the first criterion is chosen by the user (IND1=1) and the choice is not overridden because a strike is to end this cycle, the program tests to see if the present cycle is the one in which a pre-emptive strike is to take place. If it is, for each target category ITC, the escalation state IESC is set equal to the proposed state ISCL1, provided the proposed state is higher than the present state.

b. Section 20 - Has a Tactical Event Occurred Which Stimulates Firing of Nuclear Weapons. The second criterion has seven events, any of which may be selected by the user (IND2=1) for changing the escalation state. The testing of conditions for each of these events is described below.

The first event is border incursions. If the FEBA has moved into this side's territory as measured from the FEBA location at time zero, then for each target type escalation state IESC is set to the proposed state ISCL2 provided the proposed state is higher.

The second event is the advance within the sector beyond the advance in adjacent sectors. If the enemy is attacking in all adjacent sectors and if the distance between the FEBA in sector IS and the FEBA in every adjacent sector is greater than DPTH2, then for each target type the escalation state IESC may be increased.

The third event is a cumulative enemy advancement in the sector of more than a specified distance. If the total FEBA advancement by the enemy in the sector is greater than the threshold depth DPTH3 for side L, then for each target type the escalation state IESC may be increased.

The fourth event is an enemy advancement in the sector of more than a specified distance since the last cycle. If the

enemy advancement since the last cycle exceeds the value of DP2 for escalation state J, the escalation state IESC may be increased to state J for those target types selected by the user (IDEL2=1).

The fifth event is the cumulative loss of QRA aircraft beyond a specified level. If the fraction of QRA aircraft lost is greater than the threshold fraction THFRC, then for each target type the escalation state may be increased.

The sixth event is the cumulative loss of nuclear delivery systems (missiles and artillery) beyond specified levels. If the fraction of missiles (artillery) lost is greater than the threshold fraction THFR for missiles (artillery), then for each target type the escalation state may be increased.

The seventh event occurs when the rate of advancement is too slow in sectors of main attack. If side L is not the theater attacker or if the sector IS is not one of main attack, event 7 is not considered. If side L's total advancement since time zero does not exceed a specified distance DPTH7 by day NDOB7 and the side L advancement during the present cycle does not exceed a distance DP7, then for each target type the escalation state may be increased.

c. Section 30 - Is the Use of Nuclear Weapons Beyond Thresholds. If the third criterion is selected by the user (IND3=1), the program considers the number of nuclear weapons delivered by the enemy into target areas in the battlefield, region and COMMZ. First, the number of nuclear weapons delivered to the active battle area (NWABA) is compared to the threshold levels NB3 to determine the proposed escalation state. Then, for each target type against which the side wishes to escalate (IDELB3=1), the escalation state may be increased. Next, for each target subtype, the number of nuclear weapons delivered to the region (NWREG) is compared to the threshold levels NR3 to determine the proposed escalation state. Then, for each target type against which the side wishes to escalate (IDELR3=1), the escalation state may be increased. Similarly, for each target subtype, the number of nuclear weapons delivered to the COMMZ (NWCZ) is compared to the threshold levels NCZ3 to determine the proposed escalation state. Then, for each target type against which the side wishes to escalate (IDELC3=1), the escalation state may be increased.

d. Section 40 - Is the Use of Chemical Weapons Received Used as Threshold. If the fourth criterion is selected by the user (IND4=1), the program considers the number of chemical weapons delivered by the enemy into target areas in the battlefield, region and COMMZ. First, the number of chemical weapons delivered to the active battle area (NCWABA) is compared to the threshold levels NCBN4 to determine the proposed escalation state. Then, for each target type against which the side wishes to escalate (INDLB=1), the escalation state may be increased. Next for each target subtype, the number of chemical weapons delivered to the region (NCWREG) is compared to the threshold levels NCRN4 to determine the proposed escalation state. Then, for each target type against which the side wishes to escalate (INDLR=1), the escalation state may be increased. Similarly, for each target subtype, the number of chemical weapons delivered to the COMMZ (NCWCZ) is compared to the threshold levels NCZN4 to determine the proposed escalation state. Then, for each target type against which the side wishes to escalate, (INDLC=1), the escalation state may be increased.

2.2.6.7 WHINUP. Subroutine WHINUP determines the number of nuclear weapon systems for both division and sector weapon systems and reallocates the inventory of nuclear warheads to division, sector and theater pools.

2.2.6.7.1 Programming Specifications. The following table summarizes the principal specifications of subroutine WHINUP:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	None
Called by	NUC1

2.2.6.7.2 Logic Functions. The total supply of nuclear warheads in the theater are allocated to division and sector nuclear weapon systems.

a. Section 10 - Allocate Nuclear Supplies to Division Nuclear Weapon Systems. For each side, the number of division nuclear weapon systems is determined from array NDVNW.

The number of different yields available for a given division nuclear weapon system is obtained from array NYDL. However, if the system has multiple yield options, that system is said to have only one yield. The total number of warheads of a particular yield in a given division nuclear weapon system is obtained by adding the warheads of that yield which have been allocated to the theater pool to the sum of the warheads allocated to the division and sector pools in each sector. The total is placed in variable N.

Nuclear warheads are reallocated to the division and sector pools on a sector by sector basis for each division nuclear weapon system and for each type of yield. First, the number of division weapon systems in a given sector for a particular type of nuclear weapon system is divided by the total number of division weapon systems in the entire theater for that type of nuclear weapon system. The result is stored in variable FRAC. Then, the number of division nuclear warheads reallocated to the division pool is determined by obtaining the product of: (1) FRAC; (2) the fraction of division nuclear warheads in the theater allocated to the division pool, given by array FDWALD; and (3) the total number of warheads in the division nuclear weapon system, given by N. Similarly, the number of division nuclear warheads reallocated to the sector pool is determined from the product of FRAC, FDWALS and N. The total number of division nuclear warheads which have been reallocated to the division and sector pools in all of the sectors combined, given by NSUM, is subtracted from N to obtain the number of remaining warheads which are then reallocated to the theater pool.

b. Section 20 - Allocate Nuclear Supplies to Sector Nuclear Weapon Systems. The logic for determining the number of sector nuclear weapon systems is identical to the logic described above for division nuclear weapon systems, with the exception that sector nuclear warheads are reallocated to sector and theater pools only.

2.2.6.8 NDSYINV. Subroutine NDSYINV determines the inventory of division, sector and theater weapon systems that deliver nuclear munitions for each side.

2.2.6.8.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NDSYINV:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	None
Called by	NUC1

2.2.6.8.2 Logic Functions. The code in subroutine NDSYINV is divided by comment cards which label the functions of the logic.

a. Section 10 - Determine Number of Division Nuclear Weapon Systems. The number of division nuclear weapon systems is determined from array NDVNW. A particular reference weapon type (IW), used in combat divisions, is chosen from array IDSWTN (IWS,L) as being characteristic of a given division nuclear weapon system. Then, an inventory of the number of division weapon systems that employ such a typical nuclear weapon is determined in each sector for each division nuclear weapon system. This is accomplished by obtaining the product of two factors. One is the actual number of reference nuclear weapons used in a division inside the active battle area, given by array WDIV (IW,ID). The other factor is the fraction of the number of reference weapons that are representative of the given nuclear weapon system, given by array FDSWTN. The product is then summed over all divisions in the active battle area in a given sector and the result is stored by sector in array NDWSI.

b. Section 20 - Determine Weighting Factors of Aircraft for Making Air Allocations to Sector and Theater Systems. The sector and theater systems both include air weapon systems. The number of available aircraft in each region and the weighting factors of aircraft in each sector of a region need to be determined before making air allocations to the sector and theater systems. For each region, beginning with the highest numbered sector, the escalation state for nuclear weapons against a particular target category is determined. A weighting factor (WT) is determined from the fraction of aircraft assigned to a given target at the specified escalation state and is summed for each sector to yield a region weighting factor (WTT). Also, for each type of aircraft, the number of available aircraft in each region, stored in array TACT, is determined by counting the number of successful CAS, ABA and INTD sorties.

c. Section 30 - Determine Number of Sector Nuclear Weapon Systems. The logic for determining the number of types of sector nuclear weapon systems is as follows. For each sector nuclear weapon system, array ISSWTN(IWS,L) gives an index to a particular weapon type. An index value of one (1) or two (2) indicates medium- or long-range sector missile systems, respectively. An index value >2 indicates air weapon systems. The number of sector nuclear weapon systems that are medium-range (or long-range) missile systems is determined by counting the number of medium-range (or long-range) missile sites in each forward sector times the fraction of medium-range (or long-range) missiles that represent the given sector nuclear weapon system. The number of sector nuclear weapon systems that are air systems is determined for each sector by obtaining the product of three quantities. These are: (1) the number of available aircraft dedicated to nuclear missions (FADTMN), (2) the fraction of air systems that represent the given sector nuclear weapon system (FSSWTN), and (3) the ratio of the weighting factor of the assigned aircraft in the sector to the weighting factor of aircraft in the region (WT/WTT).

d. Section 40 - Determine Number of Theater Nuclear Weapon Systems. The logic for determining the number of theater nuclear weapon systems is identical to the logic for determining the number of sector nuclear weapon systems with the exception that only long-range missiles are considered for theater missile weapon systems.

2.2.6.9 NUC2. Subroutine NUC2 is the main calling program for the routines that create a list of nuclear targets, in order of priority, and a list of nuclear weapons and their characteristics.

2.2.6.9.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC2:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	NUCTAR, NUCWPS
Called by	NUC

2.2.6.9.2 Logic Functions. NUC2 begins by calling subroutine NUCTAR to create a priority list of nuclear targets. Upon return, if KFLAG=-1 subroutine NUCWPS is called to create a single priority list of nuclear weapons.

2.2.6.10 NUCTAR. Subroutine NUCTAR creates a single list of preferred nuclear targets from battlefield targets, region targets and COMMZ targets and arranges them in order of priority from highest to lowest. Battlefield targets have the highest priority, region targets have the next highest, and COMMZ targets have the lowest priority.

2.2.6.10.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUCTAR:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	None
Called by	NUC2

2.2.6.10.2 Logic Functions. The logic that follows is executed for both sides. In the active battle area, battlefield targets are arranged in priority order by target type ISU (subunit) within each zone IZ. Each target is assigned a priority from array IPRI(ISU,IZ,L). Also, a unique index value is assigned to each target, indicating its position on the priority list being created. NUCBT is set equal to the index of the last battlefield target.

Region targets are arranged in priority order by target type and target area. For each target type (ISUB), a target area (ITYP) inside the region is assigned a priority from array IPREG(ITYP,ISUB,L). A unique index value is then assigned to each target indicating its position on the priority list being created. NUCRT is set equal to the index of the last target in the region.

Targets in the COMMZ are arranged similarly by target type. Each target type is assigned a priority from array IPCMZ (ISUB,L) and each target is assigned a unique index, indicating its position on the priority list being created. NUCCT is set equal to the index of the last target in the COMMZ.

2.2.6.11 NUCWPS. The main function of subroutine NUCWPS is to create a single list of nuclear weapons and their characteristics, in order of priority, from division, sector and theater weapon systems that have been allocated to a given sector. The weapons in the list are arranged by yield, and within each yield category, weapons are arranged by weapon system response time, distance from the FEBA, range, and CEP. Also, NUCWPS uses function NWHINV to determine the current inventory of nuclear warheads available to each weapon system.

2.2.6.11.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUCWPS:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	CLR, NWHINV, KCDEN, KDCDEN
Called by	NUC2

2.2.6.11.2 Logic Functions. The logic that follows is repeated twice; the first time for creating a weapons list for the Blue side and the second time for creating a weapons list for the Red side.

a. Section 10 - Construct Weapon List From Division, Sector, and Theater Nuclear Weapon Systems. Each division nuclear weapon system has a number of different positions (locations) associated with it from which the weapons are fired. Weapon characteristics, such as the distance of each weapon system from the FEBA, the number of warheads available to each weapon system and weapon system position must be considered along with certain other parameters in order to create the weapons list. In addition, there are a certain number of yields associated with each division nuclear weapon system, given by array NYLD. In constructing the list from among division weapons, variable N is incremented once for each division nuclear weapon system, weapon system position, and yield in order to index the following local arrays when making weapon assignments: the number of the yield, given by array YLDV, is assigned to local array YL; the weapon system response time, given by array WSRTN, is assigned to local array WSRT; the average distance of the weapon system

from the FEBA, given by array RDPLDV, is assigned to local array DS; the CEP of the weapon system, given by array CEPD, is assigned to local array CEP; the range of the weapon system, given by array RNGDW is assigned to local array RNG; the number of available nuclear warheads, calculated by function NWHINV, is assigned to local array NNRAV. Finally, the weapon category, weapon system, weapon position, and yield number are packed together and assigned to array INDX1. When all weapon assignments have been made for all of the weapon systems, N is equal to the total number of division nuclear weapon systems.

The logic for determining the weapon characteristics of sector and theater weapon systems is identical except that N is not reset but is continuously incremented until it equals the total number of nuclear weapon systems in the division, sector, and theater, inclusively.

b. Section 20 - Arrange Weapons on List by Yield, Weapon System Response Time, Distance From FEBA, Range, and CEP. The next step is to arrange each weapon into a single list and classify them by type of yield, regardless of the type of weapon system. Weapons listed under a certain yield are then arranged in order of priority, from highest to lowest corresponding to the most effective weapons through the least effective.

Subroutine CLR is called consecutively to clear out arrays NSFRD, NSFRS, and NSFRT. These arrays are used to accumulate the number of nuclear warheads that have been fired from the division, sector, and theater nuclear weapon systems. The arrays are only initialized at this point but are used later in the nuclear subcycle as a firing constraint when weapon to target assignments are made.

Once the nuclear weapons list is complete, a starting and ending position is determined for all weapon systems having a certain yield. This is accomplished by comparing the yield index (IYLD) with the number of each yield (YL), working down the list. When a match is found, then the yield number becomes the starting position on the list and is assigned to array KIWYF. The ending position is found by counting backwards up the list. When a match is found, the position just above it becomes the ending position and is assigned to array KIWYL. This is repeated for each yield in the list.

2.2.6.12 NWHINV. Function NWHINV is used by several of the nuclear subroutines to determine the current inventory of nuclear warheads for division, sector, or theater weapon systems for a given yield.

2.2.6.12.1 Programming Specifications. The following table summarizes the principal specifications of function NWHINV:

Characteristic	Specification
Formal parameters	IWC = Weapon category index
	IWS = Weapon system index
	KIYD = Index to yield
	IPOS = Index to position (location)
	IS = Sector index
	L = Side index
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	None
Called by	NUCWPS, PREYLD

2.2.6.12.2 Logic Functions. The current inventory of nuclear warheads, NWHINV, for a given division weapon system, is computed by obtaining the product of two quantities. The first is the number of available warheads of a specified yield that have been allocated to the division pool in a given sector. The second is the fraction of the given division weapon system that has been assigned to the specified division position. If function NWHINV is being called from the NUC5 link, the maximum number of warheads that can be fired (NCRFIR) by a given division weapon system from the specified position is computed. Finally, the number of nuclear warheads that are available is determined from the smaller of the two values, NWHINV or NCRFIR.

The logic for determining the number of nuclear warheads available to sector and theater weapon systems is identical to the logic described above.

2.2.6.13 NUC3. Subroutine NUC3 determines the expected number of battlefield nuclear targets (subunits) detected.

2.2.6.13.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC3:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL1, LOCAA1, AFSTF2
Subroutines called	None
Called by	NUC

2.2.6.13.2 Logic Functions. NUC3 determines the number of potential nuclear targets, NPTAS, in the active battle area for sector KISS, as requested by subroutine NUC. The calculations in the routine are performed for each side L, for each subunit ISU, each zone IZ, and each division IDS in the active battle area of the sector. The program first tests to see if the target is allowable at the current escalation state. If not, no further calculations are made for this division. Otherwise, NUC3 determines the expected number of targets detected in the zone (NPT) from the product of PSZDDS(ISU,IZ,IDS), the fraction of units of a type ISU detected in zone IZ in division IDS; NSUTD(ISU,ID), the number of subunits of type ISU in the division; and FSUAZ(ISU, IZ,IT), the fraction of units of type ISU allocated to zone IZ by division type IT. Then, the maximum number of subunits allowed to be targeted (MAXNPT) is determined as the product of the number of subunits and the maximum fraction of subunits which can be targeted (FRMAXC) and under the current escalation state.

Local array NPTAS contains a running tally of the number of potential targets by type subunit and by division, i.e., NPTAS is the sum over zones in the active battle area of NPT. If NPT for any zone will cause the sum NPTAS to exceed the maximum (MAXNPT), NPT must be reduced for that zone as described below. If the maximum is reached in the first zone, then NPT and NPTAS are each set equal to the maximum value. For any other zone, NPT is the difference between the maximum value and the number of targets in lower indexed zones, and NPTAS is the maximum value.

If requested (IPRS=1) summary table C5, "Maximum Number of Nuclear Targets (Subunits) in Active Battle Area", is printed.

2.2.6.14 NUC4. Subroutine NUC4 determines the fraction of nuclear targets precluded by collateral damage constraints. It is called by program NUC for each sector KISS.

2.2.6.14.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC4:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL1, LOCAA1, AFSTF2
Subroutines called	None
Called by	NUC

2.2.6.14.2 Logic Functions. NUC4 sets index IS to the sector value KISS passed from subroutine NUC. The remaining calculations are performed for each side L.

Index JE is set to the current nuclear escalation state for the sector and target type 1 (divisions in the active battle area). Index JBA is set to the index of the active battle area in sector IS. If there are no side L divisions in the active battle area, no further calculations are performed for side L.

The distribution of population within each battle area is described by array PZPI which contains the percentage of the population held in city populations of five levels. Array POPLM defines the five levels, e.g., 5K, 10K, 25K, 50K, 100K. The input variable PDMMX indicates the minimum city size which constitutes a collateral damage constraint for the current escalation level JE. The percentage (ZOLIM) of the population of the current active battle area that resides in cities having populations of at least PDMMX is calculated, in most cases, by a linear interpolation within PZPI. To calculate ZOLIM, the subroutine compares PDMMX to the five values of POPLM to determine between which two levels the city size for constraint lies.

If the city size for constraint exceeds the largest level of POPLM, the percent of people in the zone precluded by the targeting constraint (ZOLIM) is calculated as follows. The percent of population in cities greater than the largest level (PZPI(JBA,5)) is multiplied by the ratio of the largest population level (POPLM(5)) to the city population for constraint (PDMMX). The resulting percentage is ZOLIM. In this calculation, it is assumed that the distribution of population beyond the largest level is hyperbolic.

If the city size for constraint lies between levels KJB and KJ of POPLM, the value of ZOLIM is calculated as follows. The difference in the values of PZPI for the KJB level and the KJ level is the percent population between these two levels. Since PDMMX falls between those two levels, only a portion of this percentage will be precluded from targeting. Assuming a linear distribution of population between the two levels, the percent population between the two levels which can be targeted is proportionate to the ratio of the difference between the city population for constraint and the population level KJB to the difference between the population levels KJ and KJB. ZOLIM is set to the difference between the percent population value for level KJB and the percent population, between the two levels of KJB and KJ, which can be targeted.

If the city size for constraint is smaller than the first level of POPLM, the value of ZOLIM is determined by the same formula as in the preceding paragraph, with the lower level being set to 0 people (in thousands) and the percentage to 100.

ZOLIM is used as an estimate of the fraction of military targets within the battle area that will be precluded from nuclear attack due to the proximity of a civilian population center. Therefore, the number of potential targets (NPT) of each subunit type within each zone of each division in the sector is multiplied by (1-ZOLIM) to account for civilian casualty constraints.

2.2.6.15 NUC5. Subroutine NUC5 determines the order in which nuclear weapons are assigned to battlefield, region and COMMZ targets for each side. First, the total number of possible targets is determined and then the target priority and category are determined. Finally, a call is made to the proper subroutine to complete the weapon to target assignments.

2.2.6.15.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC5:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	NBFTGS, NRGTGS, NCZTGS
Called by	NUC

2.2.6.15.2 Logic Functions. The total number of targets to be fired upon (NTAR) is determined for each side from variable NUCCT, which is the index in the priority list of the last nuclear target in the COMMZ. For each individual target, a target category (J) is determined from array INUTAR. Then the weapon to target assignments are completed by a call to the appropriate subroutine, as discussed below. If the target category falls within the range of battlefield targets on the priority list, (i.e., J≤NUCBT), a call is made to subroutine NBFTGS. If the target category lies within the range of region targets (i.e., NUCBT<J≤NUCRT), a call is made to subroutine NRGTGS. Finally, if the target category falls outside the range of region targets, (i.e., J>NUCRT) it must fall within the range of COMMZ targets and so subroutine NCZTGS is called.

2.2.6.16 ZNDST. Function ZNDST calculates the distance of the specified division location (zone) from the FEBA.

2.2.6.16.1 Programming Specifications. The following table summarizes the principal specifications of function ZNDST:

Characteristic	Specification
Formal parameters	IDS = Index to division position IZ = Index to zone IS = Index to sector
Common blocks	Blank Common

Characteristic	Specification
Subroutines called	None
Called by	BFTGTS, NBFTGS, PREYLD

2.2.6.16.2 Logic Functions. The index (ID) to the division located in the active battle area at position IDS of the given sector is determined from array IDLABA(IDS,IS). The division type (IT) is determined from array ITD, which is indexed by ID. Now index IT is used to obtain the depth of the division, given by array DVDPTH(IT), which is placed in variable D.

The fraction of the division depth which comprises a zone, given by array PZDTH, is summed for each zone in the division up to but not including the zone specified by IZ. To this sum is added one half of the fraction of the division depth represented by zone IZ, and the total is stored in variable P. The distance of the division location from the FEBA is now obtained from the product of P and D.

2.2.6.17 NUCABS. Subroutine NUCABS computes the weighted airbase value of each targeted airbase in order to establish a targeting priority. The airbase targets are ranked from highest priority to lowest corresponding to the airbases with the largest weighted airbase values to the smallest. The routine at entry point NUCABD determines the distance (along the great circle) between a given airbase target and the FEBA by plotting the latitude and longitude coordinates of the airbase and the center of the active battle area at the FEBA.

2.2.6.17.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUCABS:

Characteristic	Specification
Formal parameters	KCT = Index to a particular airbase
	NCT = Number of airbases
	L = Index to side
	IS = Index to sector
	IABT = Index to a particular airbase target

Characteristic	Specification
	DIST = Great circle distance between an airbase and the FEBA
	IPAAC = Index to an active parking area
	NPAAC = Number of active parking areas
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	SECWTH, GDIST
Called by	NRGTGS, NCZTGS, PREYLD

2.2.6.17.2 Logic Functions. The code of subroutine NUCABS is divided by comment cards which label the functions of the logic.

a. Section 10 - Compute Weighted Airbase Values and Determine Targeting Priority. For every airbase that is either a region or COMMZ target, the number of types of aircraft assigned to a particular airbase is summed and placed in variable VLA. The number of shelters assigned to an airbase is placed in variable VLS. The weighting value of the number of aircraft to the number of shelters for an airbase is given by the array WTNAST. The weighted value of the airbase, assigned to array VAL, is now computed using the following formula:

VLA*WTNAST + VLS*(1-WTNAST).

After all the values have been computed, array KCT, containing the airbases, and array VAL, containing the weighted airbase values, are reordered such that the airbase with the largest weighted airbase value (and thus having the highest priority) is listed first, continuing down to the airbase with the smallest weighted airbase value.

b. Section 20 - Find Distance Between Airbase Target and the FEBA. At entry point NUCABD, the latitude and longitude coordinates of the targeted airbase are determined from

array AF and placed in variables FLATAB and FLONAB, respectively. The cumulative ground distance between the eastern boundary of the given sector and the leading edge of the active battle area is determined from array GDBA and placed in variable GD.

At this point subroutine SECWTH is called to compute the width of the combat sector at the western boundary of the active battle area. SECWTH returns with data which describes the line of advance through the sector segment containing the western edge of the active battle area; this data includes the latitude and longitude of the end points of the line of advance through the segment and the fractional distance (AL) through the segment to the western edge of the active battle area. Projection equations and spherical geometry are used to determine the latitude and longitude of the midpoint of the active battle area. Function GDIST is now used to compute the distance between the targeted airbase, given by the coordinates (FLATAB, FLONAB), and the center of the active battle area given by the coordinates (FLATBA, FLONBA).

Finally, a count is kept in variable NPAAC of the number of active parking areas within the airbase. This is determined by searching the first six elements of array IMAGE1 for non-zero values. Array IPAAC contains the index of the active parking area.

2.2.6.18 NBFTGS. Subroutine NBFTGS assigns nuclear weapons to battlefield targets for each division in a given sector. Given that the targets are allowed under the restrictions of the current escalation state and engagement distance from the FEBA, a weapon system with the most desirable yield is selected.

2.2.6.18.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NBFTGS:

Characteristic	Specification
Formal parameters	N = Number of targets for a given side
	JTAR = Target category of Nth priority target
	L = Side index (L=1 for Blue side; L=2 for Red side)

Characteristic	Specification
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	PREYLD, ZNDST
Called by	NUC5

2.2.6.18.2 Logic Functions. Given that the target category selected in NUC5 is allowed at the escalation state JE defined by array IESC(IS,ITC,L), the subunit (ISU) in zone (IZ) to be targeted is determined. First array IALBT is checked to determine if the targeted subunit in the zone is allowable under the given escalation state. If so, for each division in the active battle area, function ZNDST calculates the distance of the targeted subunit in the division from the FEBA. If the maximum distance allowed for the target at the given escalation state, defined by array RMXDP, is less than the distance calculated by ZNDST, then the number of potential targets (subunits) in the division is zero and another division is selected for targeting.

Once a division has been selected, subroutine PREYLD is called to select and assign a nuclear weapon system with the preferred or most desirable yield suitable for the target.

2.2.6.19 NRGTGS. Subroutine NRGTGS assigns nuclear weapons to region targets. First, the type of region target and target area is determined, and then a weapon system is chosen that has the most desirable yield.

2.2.6.19.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NRGTGS:

Characteristic	Specification
Formal parameters	N = Number of targets for a given side
	JTAR = Target category of Nth priority target
	L = Side index (L=1 for Blue side; L=2 for Red side)

Characteristic	Specification
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1
Subroutines called	NUCABS, PREYLD
Called by	NUC5

2.2.6.19.2 Logic Functions. The type of region target, ISUB, and the target area, ITYP, are determined. Given that the target, by type and area, is allowed at the current escalation state defined by array IALRT, weapon assignments are made on the basis of region target type and target area for a given sector. The four target types are airbases, supply nodes, missile sites, and divisions in the rear.

a. Section 10 - Assign Weapon System to Region Airbase Targets. For airbase targets, the target areas are either sector forward or sector rear. The logic for determining the number of airbase targets in the forward target area is similar to that for the rear target area. The depth of sector forward (or sector rear) is determined from array NDFAB (or NDRAB). The depth is measured in terms of the number of battle areas. Index NRT keeps count of the number of targeted airbases within each battle area. If NRT=0 for either sector forward or sector rear, another target must be selected. If NRT 0, subroutine NUCABS is called to order the airbase targets by the number of aircraft and shelters assigned to them. Then, for each targeted airbase, subroutine PREYLD is called to select and assign a nuclear weapon system with the most desirable yield.

b. Section 20 - Assign Weapon System to Region Supply Node Targets. For supply node targets, there are two target areas; the active battle area and the inactive battle area within the region. For supply nodes feeding the active battle area, the targeted supply node in a given sector is determined. Then subroutine PREYLD is called to select and assign a nuclear weapon system with the preferred yield. For supply nodes feeding inactive battle areas in the region, the depth of sector forward plus sector rear (measured in terms of the number of battle areas) is determined. An index, ISN, of the supply nodes that supply each inactive battle area is set. A supply node must belong to the opposite side for it to be counted as a targeted supply node. Those targeted supply nodes which supply more than one battle area in

the region are counted as one target. Then, for each targeted supply node, subroutine PREYLD is called to select and assign a nuclear weapon system with the most desirable yield.

c. Section 30 - Assign Weapon System to Region Missile Site Targets. For each targeted missile site, as determined by the product of the number of missile sites in a given sector and the probability of detection, subroutine PREYLD is called to select and assign a nuclear weapon system with the most desirable yield.

d. Section 40 - Assign Weapon System to Targeted Region Divisions in Rear. For division targets, only those divisions in the rear (i.e., in the first inactive battle area) are targeted. First, the number of divisions to be targeted (NDT) is obtained from the product of the number of divisions in the first inactive battle area of a given sector and the probability of any one being detected. For each targeted division and for each subunit type within a targeted division, there corresponds an index (ISU) to the priority targeting of subunits in divisions in the rear, and an index (ID) of a specific rear division. The actual number of subunits to be targeted is then determined from the product of the current number of subunit targets, indexed by ISU and ID, and the probability that such a subunit is detected. For each targeted subunit, subroutine PREYLD is called to assign a nuclear weapon system with the most desirable yield.

2.2.6.20 NCZTGS. Subroutine NCZTGS assigns nuclear weapons to COMMZ targets. Weapon assignments are made according to COMMZ target type. If the target type is not allowed under the current escalation state, or there are no targets in the specified target category, another target must be selected.

2.2.6.20.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NCZTGS:

Characteristic	Specification
Formal parameters	N = Number of nuclear targets for a given side (priority ordered)
	JTAR = Target category of Nth priority target
	L = Side index (L = 1 for Blue side; L = 2 for Red side)

Characteristic	Specification
Common blocks	Blank Common
Subroutines called	NUCABS, PREYLD
Called by	NUC5

2.2.6.20.2 Logic Functions. The type of COMMZ target, ISUB, is determined directly from parameter JTAR. The three types of COMMZ targets are airbases, supply nodes, and missile sites. If the target type is allowed at the escalation state defined by array IALCT, the most forward battle area in the COMMZ (KIBA) is calculated.

This calculation is based on the side being targeted. For the Blue side, the depth of sector forward and sector rear (measured in terms of the number of battle areas) is subtracted from the index to the active battle area in the sector. For the Red side, the depth is added to the active battle area index.

a. Section 10 - Assign Weapon System to COMMZ Airbase Targets. For airbase targets, an index to the number of battle areas within the bounds of the COMMZ is determined, and stored in variable IBA. For each airbase in the COMMZ, a comparison is made between the battle area location of the airbase, given by array IAFBA, and the number of the battle area, indexed by IBA. If there is a match, the COMMZ target count is incremented and is used as an index to determine the number of the targeted airbase. This process is repeated for each indexed battle area and for each airbase. Finally subroutine NUCABS is called to order the airbase targets. Then for each ordered target, subroutine PREYLD is called to complete the weapon to target assignments.

b. Section 20 - Assign Weapon System to COMMZ Supply Node Targets. The logic for determining the number of targeted supply nodes in the COMMZ is determined as follows. For each battle area in the COMMZ, the index of the supply node that supplies it is determined from ISNBA. If the supply node is owned by the opposite side, then the COMMZ target count is incremented and used to index array KCT, which represents the number of the targeted supply nodes. When the total number of targets is obtained, array KCT is searched to eliminate duplicates due to supply nodes supplying more than one

battle area in the COMMZ. Finally, subroutine PREYLD is called to assign the most desirable nuclear weapon system to each targeted supply node.

c. Section 30 - Assign Weapon System to COMMZ Missile Site Targets. The number of missile site targets is determined from the actual number of missile sites in the COMMZ and the probability of detection. Then subroutine PREYLD is called to complete the weapon to target assignments.

2.2.6.21 PREYLD. Subroutine PREYLD completes the weapon to target assignments for battlefield, region, and COMMZ targets which were initially begun in subroutines NBFTGS, NRGTGS, and NCZTGS, respectively. PREYLD selects a weapon system with the most desirable yield to be used against the specified target. If such a weapon system is not available, an alternative yield is chosen.

2.2.6.21.1 Programming Specifications. The following table summarizes the principal specifications of subroutine PREYLD:

Characteristic	Specification
Formal parameters	ITGT = Index to target type
	ITC = Index to target category
	ISUB = Index to target subtype
	ISU = Index to division subunit
	IZ = Index to division zone
	IDS = Index to division
	ITYP = Index to target area within ISUB
	KRT = Index to the number of the target within ISUB
	NRT = Number of target subtypes
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1

Characteristic	Specification
Subroutines called	NUCABD, ZNDST, KDCDEN, NWHINV, DWHINV
Called by	NBFTGS, NRGTGS, NCZTGS

2.2.6.21.2 Logic Functions. The code of subroutine PREYLD is divided by comment cards which label the functions of the logic.

a. Section 10 - Determine Preferred Yields to Use Against Battlefield, Region, and COMMZ Targets. First, the target category is determined from parameter ITC. For battlefield targets, the preferred yield to use against the targeted subunit (ISU) is given by array YLT. The number of potential targets in the division specified by parameter IDS of subunit type ISU and zone IZ is assigned to variable NWPN. Function ZNDST calculates the distance of the targeted subunit from the FEBA.

Region and COMMZ targets both consist of airbases, supply nodes, and missile sites. Region targets also include divisions in the rear. For airbase targets, subroutine NUCABD is called to determine the distance of the given airbase target from the FEBA. The preferred yield to use in this case is based on the targeting assumption, given by array IABTA, at the current escalation state. An airbase target may be assumed to be the center of the runway, the most dense parking area in the airbase or any other parking area within the airbase. In the latter case, there may be more than one parking area that is targeted. For supply node targets, the preferred yield to use is given by array YLTSN and for missile sites and divisions in the rear, the preferred yields are determined from arrays YLTSSM and YLTDR, respectively.

b. Section 20 - Find Acceptable Weapon System With Desired Yield. The starting and ending positions of those nuclear weapons on the nuclear weapon priority list that fall within range of the preferred yield are determined and assigned to arrays KIWB and KIWL, respectively. Beginning with the first weapon on the list, function KDCDEN is used to determine whether the weapon is a division, sector or theater weapon.

Once determined, the yield of the weapon, given by array YLDV (YLSW for sector weapons or YLTH for theater weapons), is compared against the preferred yield established earlier.

If they do not match, the second weapon on the list is tried. When a match is found, the range of the given weapon is compared with the combined distance of the target from the FEBA and the average distance of the weapon from the FEBA. If the range of the weapon is less than that distance, the next weapon on the list is considered and the process described thus far is repeated. If the distance of the target from the FEBA is less than the minimum distance beyond the FEBA that the weapon can be fired, the next weapon on the list is considered. At this point, if a weapon has thus far proved to be satisfactory, the number of nuclear warheads that are available to the weapon is determined by function NWHINV. If this amount is less than the amount needed for the mission, another weapon is selected and the whole process is repeated. However, if the number of available warheads is satisfactory, and does not exceed the maximum number of rounds that may be fired by the weapon, then the weapon system chosen is acceptable and subroutine DWHINV is called to reduce the inventory of warheads by the quantity used. If none of the weapons in the list meet the requirements described above, either the next higher or next lower yield is chosen, depending on the value of array IDELTA. Another list of weapons is compiled from the weapon list that fall within the range of the second choice of yield.

Finally, the weapon system that is chosen, the number of desired warheads, the preferred yield, and the target category are assigned to the various targeting arrays, to complete the weapon to target assignments.

2.2.6.22 DWHINV. Subroutine DWHINV reduces the current inventory of nuclear warheads in the supply pool for a given weapon type by the quantity used by the weapon.

2.2.6.22.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DWHINV:

Characteristic	Specification
Formal parameters	L = Index to side
	IW = Index to weapon type
	NCRDES = Desired number of nuclear rounds
Common blocks	Blank Common, AFSTF2, LOCAA1, LOCAL1

Characteristic	Specification
Subroutines called	KDCDEN
Called by	PREYLD

2.2.6.22.2 Logic Functions. First, function KDCDEN is used to obtain the weapon category (IWC), the weapon system (IWS), the position of the weapon (IPDS), and the yield (IYD) for the given weapon type. If the weapon is a division weapon, the number of nuclear warheads fired by weapon type IW is subtracted from the pool of warheads allocated to division weapons for the given yield. Also, the number of warheads fired is added to the current inventory of warheads fired by the particular division weapon system type, given by array NSFRD.

The same logic is applicable to sector and theater weapon system types. The number of warheads fired by the weapon system is subtracted from the pool of warheads allocated to the respective weapon system for a given yield and is added to the current inventory of warheads fired by the respective weapon system type. In addition, if any air missions have been flown, the number of missions for each of the CAS, ABA, and INTD sorties is reduced proportionally by subtracting from each the ratio of the number of warheads fired to the total number of missions flown.

2.2.6.23 NUC6. This subroutine is the master program for the nuclear damage assessment. It calls DAMEVL to perform nuclear damage assessment calculations in sector KISS. NUC6 sets up assessment variables before DAMEVL is called and accumulates results after the assessment calculations have been made.

2.2.6.23.1 Programming Specifications. The following table summarizes the principal specifications of subroutine NUC6:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL1, LOCAA1, AFSTF2
Subroutines called	DAMEVL, KDCDEN
Called by	NUC

2.2.6.23.2 Logic Functions. Subroutine NUC6 begins by setting up targeting variables for use by DAMEVL in assessing damage due to nuclear munitions in sector IS (IS equals common variable KISS). For each attacking side L, the number of nuclear fire missions is specified (NMF = NNIWAS(LL)). Then for each fire mission I the following are determined: target type IWLTLO(I,L)=1 (division), 2 (airbase), 3 (supply node), or 4 (surface-to-surface missile site); IWLHOB(I,L)=0 for surface burst, 1 otherwise -- as determined by target type and escalation state. The weapon CEP (WLCEP(I,L)) is specified, depending on whether the weapon is part of a division, sector, or theater system. After all nuclear fire missions in this sector have been specified, subroutine DAMEVL(KISS) is called to perform damage assessment calculations.

The last part of this program totals up the civilian casualties and fatalities to date in this sector. The number of personnel in each radiation category in the radiation pools, FPRC(ID,IRS), for each division ID is updated. The number of personnel in each division, PDIV(ID), is reduced by the number of fatalities accrued in this cycle from each radiation pool.

2.2.6.24 DAMEVL. Subroutine DAMEVL is called by program NUC6 once each cycle to perform nuclear damage assessment calculations for both sides in sector JS. Side L attacks side L2. Damage assessment is made for the following types of targets: active and inactive combat units, airbases, supply nodes, and surface-to-surface missile sites. DAMEVL calculates the number of target elements destroyed by blast and initial radiation effects. Damage calculations include the collateral effects of nuclear weapons on units near their targets.

2.2.6.24.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DAMEVL:

Characteristic	Specification
Formal parameters	JS = Sector in which damage is evaluated
Common blocks	Blank Common, LOCAL1, LOCAA1, AFSTF2

Characteristics	Specification
Subroutine called	CVFW, DOSLIM, FN, OFFCOV, PAREA, PREFN, SIRCOV, WRAD, WRADVN
Called by	NUC6

2.2.6.24.2 Logic Functions. Nearly all of subroutine DAMEVL (sections 100 to 500) consists of a Do loop over attacking side L. The first section initializes variables for the damage calculations.

a. Section 10 - Initialize Constants. This section initializes the following constants: Summary print flag IPRS is set to 1 if flag IPRD=1; the number of radiation states, NRS, is set to 5; NPR, the number of radiation protection categories also equals 5; the civilian population density, PDENSS, is currently set to a constant value of 100 per square kilometer.

b. Section 100 - Begin Nuclear Damage Assessment. A Do loop begins on L to calculate the effects of side L weapons on side L2 targets in sector JS. The following parameters are specified: NSUTT, the number of subunits for side L2; NBTP, the number of types of weapons in side L2 subunits; NNZ, the number of zones within side L2 divisions; NBTT, the number of types of side L2 division weapons. NTMP specifies the number of nuclear fire missions by side L. Variable ISA(JS) specifies the attacker in sector JS. If the attacker is side L2, tactical role parameter ITR=1. If the attacker is side L1, ITR=2. Control is transferred to section 200 to begin assessment for battlefield targets.

c. Section 200 - Begin Assessment for Battlefield Targets. A Do loop is begun on each nuclear fire mission I by side L. The loop ends in section 250. Only weapons targeted against combat divisions (active or in the rear) (IWLTLO(I,L)=1) are considered. One division at a time is considered, i.e., damage assessment calculations are made for weapons directed against divisions with location identifier IDC before IDC is advanced for another targeted division and the weapons list is searched again. When all divisions have been processed, control transfers to section 300 to begin assessment calculations for airbase targets.

The ID (identifier) of a division specified by location IDC is obtained from arrays IDLABA(IDC,JS), for an active division, or IDLIBA(IDC,JS,L) if inactive. Before the assessment is made of the first weapon assigned to that division, several assessment variables are initialized. FPRC(ID,IRS), the fraction of division ID personnel in radiation category IRS, is updated so that the sum of the fractions in pools 1,2,---NRS-1=1. Pool NRS, the immediate lethality pool, was accounted for in the previous cycle by drawing down the number of personnel in this division. TPSU(ISU) is the total number of people remaining in subunit ISU of division ID adjusted for previous attrition. For each weapon I assigned to division ID, control passes to the next section of coding.

d. Section 210 - Calculate Weapon Characteristics. This section is within the Do loop on fire mission I attacking division IDS. The weapon yield (YIELD=WLYLD(I,L)) and circular-error-probable (CEPP=WLCEP(I,L) are specified. The height of burst index IHOB (IHOB=IWLHOB(IL)) is zero for surface bursts and 1 for air bursts. The scaled height of burst SHOB is set to 1.74*IHOB for input to WRADVN for weapon radius calculations. The following target characteristics are specified: subunit index ISU (=IWLCOT(I,L)); zone index IZN (=IWLTZN(I,L)); and the number of subunits in the division which are targeted in this fire mission NMBRW (=IWLBT(I,L)).

Subroutine PREFN(YIELD,IHOB,DOSG,DOSN) is called to calculate a table of unattenuated prompt gamma and neutron dose levels (DOSG and DOSN) at 23 distances from ground zero. These dose levels will be used later by function FN for casualty calculations to invert the function of dose vs. distance. For a division in reserve posture (IZN=0), the target radius TT is specified by RDSUR(ISU,2) and the zone area is the division area in square meters. For a division in combat, the above target radius is multiplied by the factor FRDSUZ(IZN,ITR,L2) and the zone area is multiplied by the fraction PZDPTH(IZN,L2).

The weapon offset aim point distance OD is specified (temporarily set to 0). The total CEP of the weapon is

$$TCEP = \left[CEPP^2 + (TAESZD(ISU,IZN,IDS))^2 \right]^{1/2}$$

where TAESZD(ISU,IZN,IDS) is the target acquisition sensor error. Calculations are continued in the next section of coding, section 220.

e. Section 220 - Calculate Areas Covered by Damaging Effects. Section 220 is within the Do loop on fire mission I attacking division IDS. The first part of this section contains a Do loop on IW, the side L2 weapon type, nested within a Do loop on damage level IDAM (IDAM=1 for severe damage or IDAM=2 for moderate damage). For each IW and IDAM, the weapon radius WR is calculated by obtaining the value of the function WRADVN(YIELD,SHOB,IVNW(IDAM,IW,L2),WSIG,IDNT) and converting this value from feet to meters.

If the targeted subunit ISU contains weapons of type IW, subroutine SIRCOV is called to calculate PPK, the fraction of type IW weapons in the target area damaged to level IDAM. Local variable PR is the fraction of the target zone receiving damage outside the targeted subunit. PR is calculated by first calling subroutine OFFCOV to calculate COV, the expected fraction of a circle the size of the target covered by weapon effects of radius WR. PR is the ratio of the area covered by the weapon minus the target area covered to the zone area minus the target area. The probability that the target survives the effects of one weapon targeted at it is (1-PPK). The probability that the target subunit survives the effects of weapons targeted at other subunits in the zone is $(1-PR)^{NMBRW}$. The local variables PB, the cumulative probability of surviving bonus damage in the division, and EDAM, a ratio of survival probabilities, are defined as follows for later use:

$$PB(IZN,IDAM,IW)=PB(IZN,IDAM,IW)(1-PR)^{NMBRW}$$

$$EDAM(IDAM,ISBT,I) = \frac{1-PPK}{(1-PR)^{NMBRW}}$$

This completes the Do loops on IDAM and IW. For each protection category IPC, the lethal radius for a weapon in fire mission I is obtained from function WRADVN. Calculations are continued in the next section of coding.

f. Section 230 - Calculate Probabilities for Transitions Between Radiation Dose States. This section consists of a Do loop over radiation dose state IR to calculate the

probability of transition to a higher dose state IS. Calculations are performed for each state IS higher than IR and for each protection category IPC. Subroutine DOSLIM is called to calculate the radiation levels RU and RB which will produce transitions from the center of radiation state IR to the upper and lower boundaries of state IS. Local variables Y1 and Y2 are distances at which prompt radiation levels RU and RB are experienced by personnel in protection category IPC. Y1 and Y2 are calculated using function FN and then adjusted if they are less than the blast lethal radius.

Subroutine SIRCOV is called using Y1 and Y2 to calculate the fraction of the target covered by radiation doses RU and RB. The difference between these fractions is local variable RH(IPC). The probability of transition from cumulative radiation state IR to state IS for personnel in the targeted unit is P1(IR,IS), which is averaged over all protection categories:

$$P1(IR,IS) = \sum_{IPC=1}^{NPR} RH(IPC)*FRSUPC(ISU,IPC,ITR+2*(L2-1))$$

where array FRSUPC specifies the fraction of subunit ISU in protection category IPC while side L2 divisions in combat are in tactical role ITR.

The probability of transition from cumulative radiation state IR to state IS for personnel outside the targeted unit is P2(IR,IS). To calculate P2(IR,IS), a "cookie-cutter" weapon is assumed. For each protection category IPC subroutine OFFCOV is called twice to calculate F1 and F2, the fraction of personnel outside the target covered by doses RU and RB, respectively. Then PR2(IPC) is set equal to the ratio of the area covered by the weapon minum the target area covered to the zone area minus the target area. P2(IR,IS) is the average over all protection categories of the product of PR2(IPC) and the fraction of military personnel in the division in each protection category.

The probabilities that a transition from state IR does not occur are P1(IR,IR) and P2(IR,IR). These quantities are obtained by subtracting from 1.0 the sum of the probabilities that a transition does occur to another state. Calculations for fire mission I are continued in the next section of coding.

g. Section 240 - Update the Fractions of Personnel in Each Radiation Category. For division ID, the fraction of personnel in radiation state IC outside the targeted subunit after NMBRW weapons in fire mission I have been fired at targets in the division is updated by:

$$FPRC(ID,IC) = \sum_{IJ=1}^{NMBRW} \sum_{IR=1}^{IC} P2(IR,IC)*FPRC(ID,IR).$$

P2(IR,IC), the transition probability for bonus casualties, was calculated in section 230. After this fire mission the fraction of personnel in the targeted subunit in radiation state IC is TPRC(KCNT,IC). KCNT counts the fire missions against this division. Assuming the targeted subunit is initially in the same state as the division as a whole, the value of TPRC due to the weapon fired at this unit in this fire mission is:

$$TPRC(KCNT,IC) = \sum_{IR=1}^{IC} P1(IR,IC)*FPRC(ID,IR).$$

If other type I weapons are fired at other subunits in this fire mission, bonus effects are included by:

$$TPRC(KCNT,IC) = \sum_{IJ=2}^{NMBRW} \sum_{IR=1}^{IC} P2(IR,IC)*TPRC(KCNT,IR)$$

The bonus effects of this fire mission on the subunits targeted in the previous fire missions (LL=1,KCNT-1) is calculated from:

$$TPRC(LL,IC) = \sum_{IJ=1}^{NMBRW} \sum_{IR=1}^{IC} P2(IR,IC)*TPRC(LL,IR)$$

Control is passed to section 600 (KCCSW=1) to calculate civilian casualties. Control then returns to section 240 and the Do loop on fire mission I begun in section 200 ends. Control then passes to section 250.

h. Section 250 - Begin Compilation of Damage to Targeted Division. This section begins with a Do loop over all nuclear fire missions I by side L against side L2. The array EDAM stores the probability of damage to level IDAM of weapon number ISBT of type IW in division IDC:

```
EDAM(IDAM,ISBT,I) = 1-EDAM(IDAM,ISBT,I)*PB(IZN,IDAM,IW)
```

where the terms on the right-hand side were calculated in section 220. Array PB is the cumulative probability of surviving bonus damage and EDAM on the right-hand side of the equation reflects the probability of surviving damage from weapons targeted at the subunit. For moderate damage (IDAM=2), the resultant array EDAM gives the probability of moderate, but not severe damage.

The last part of this section initializes arrays for weapons damaged and for casualties.

i. Section 260 - Calculate Numbers of Damaged Unit Components and Casualties. The main body of this section consists of a Do loop on each zone IZC within division IDC. Within each zone the number of severely damaged weapons of each type IW is given by:

$$TNKL1(IW) = \sum_{I=1}^{NTMP} \sum_{ISBT=1}^{NBTP} EDAM(1,ISBT,I)*TEMP*WNMBR$$

where

EDAM(1,ISBT,I) = Probability of severe damage of weapon type ISBT in nuclear fire mission I

WNMBR = Number of nuclear weapons of index I

TEMP = An estimate of the prestrike number of weapons of type IW in the inventory of side L2.

The number of moderately damaged weapons, TNKLL(IW), is calculated by a similar equation. The fraction of fatalities in the targeted subunit after the strike is calculated by subtracting from 1 the fractions alive in each radiation

state. This fraction is compared with FPSWDU(ISU,L2) to determine if the subunit should be withdrawn. Casualties from this fire mission are added to the total casualties for the division, TDCAS, and targeted subunits, TPTGT. If the detail print flag IPRD is set to one, the number of people in each radiation pool and the number of weapons damaged in this nuclear strike are calculated and printed.

The number of undamaged division weapons WDIV(IW,ID) is reduced by the sum of TNKL1(IW) and TNKLL(IW). The number of moderately damaged weapons in the repair pool, WDRRP (IW,1,L2) is increased by TNKLL(IW). The number of subunits of each type ISU in the division NSUTD(ISU,ID) is reduced by the total number withdrawn.

After the above calculations have been made for all four zones of the division, summaries are printed if IPRS=1 one. Control passes to the next section of coding.

j. Section 270 - Include Bonus Casualties. This section concludes calculations on division ID. The fractions of personnel in radiation state IC outside the targeted subunits, FPRC(ID,IC), are used to increase TDCAS, the total casualties in the division. The fraction of division personnel not in subunits withdrawn which are casualties, FPKC(ID) is also updated. If IPRS=1, division summaries are printed. Control returns to section 200 to begin assessment of the next division.

k. Section 300 - Begin Assessment for Airbase Targets. Control passes to this section after damage to all combat divisions on side L2 has been assessed. The logic flow for damage assessment of airbase targets is similar to the logic for battlefield targets.

This section initializes variables used to calculate damage to airbase targets. The coding assumes that all weapons on a single airbase are adjacent in the weapon list. A runway is targeted if and only if none of the six possible parking areas are targeted (IWLBT(I,L)=0).

A Do loop is begun over each nuclear fire mission I. Only missions against airfields (IWLTLO(I,L)=2) are considered. INPATG is the number of parking areas targeted. INPACR is the current number of weapons on this airfield. Flag IABOL is set to zero when calculations are begun for a new airfield. When IABOL equals 1, (not a new airfield) control passes to section 310.

Before assessment of a new airfield is begun, damage variables are initialized and the following quantities are calculated. The airbase index IAB is set equal to IWLBT(I,L). If the runway is targeted, RIND is set to zero. If a parking area is targeted, RIND=1. INPATG is the number of fire missions directed at this airfield. Subroutine PAREA is called to determine the numbers of shelters and aircraft on the airbase and allocate them to parking areas. The total number of military personnel on the airbase TPMC is extracted from array IMAGE. These personnel are allocated to parking areas RNPLP(IPRB) and quadrants RNPL(IQD) of the airbase. Control passes to section 310 after initialization is completed.

1. Section 310 - Calculate Damage to Aircraft and Shelters. The targeted parking area, IPAR, is set equal to IWLBT(I,L). The height of burst, yield, and CEP are obtained as in the battlefield calculations. Function subroutine WRADVN is used to calculate the following three weapon radii in meters: WRA, the weapon radius for destruction of unsheltered aircraft; WRS, the weapon radius for destruction of aircraft in shelters; WRAM, the weapon radius for damage of unsheltered aircraft. If parking areas are targeted, the offset distance OFSTD(JPA) is calculated between each active parking area JPA and the targeted parking area using array AFDIM and local variable RIND. The target radius is calculated using array AFDIM. The fraction of unsheltered aircraft destroyed in parking area JPA is DAMA(JPA). The contribution of this weapon to DAMA(JPA) is calculated for each parking area in the following way. Subroutine OFFCOV calculates COV, the fraction of this parking area covered by weapon radius WRA offset a distance OFSTD(JPA). Then DAMA(JPA)=1-(1-DAMA(JPA))*(1-COV). Similar calculations are made for DAMS(JPA) and DAMM(JPA) corresponding to weapon radii WRS and WRAM, respectively. After these calculations have been made for all active parking areas for this weapon, control is passed to the next section of code.

m. Section 320 - Calculate Military Personnel Casualties. This section consists of a Do loop over all parking areas ICMP. The target radius and the offset distance OFSTD are calculated. Calculations then proceed as in section 230 to calculate the transition matrix P1(IR,IQ) which gives the probability that personnel initially in radiation state IR will be in state IQ after this weapon detonates. The fraction of personnel in each radiation state IR is TPRD(ICMP,IR).

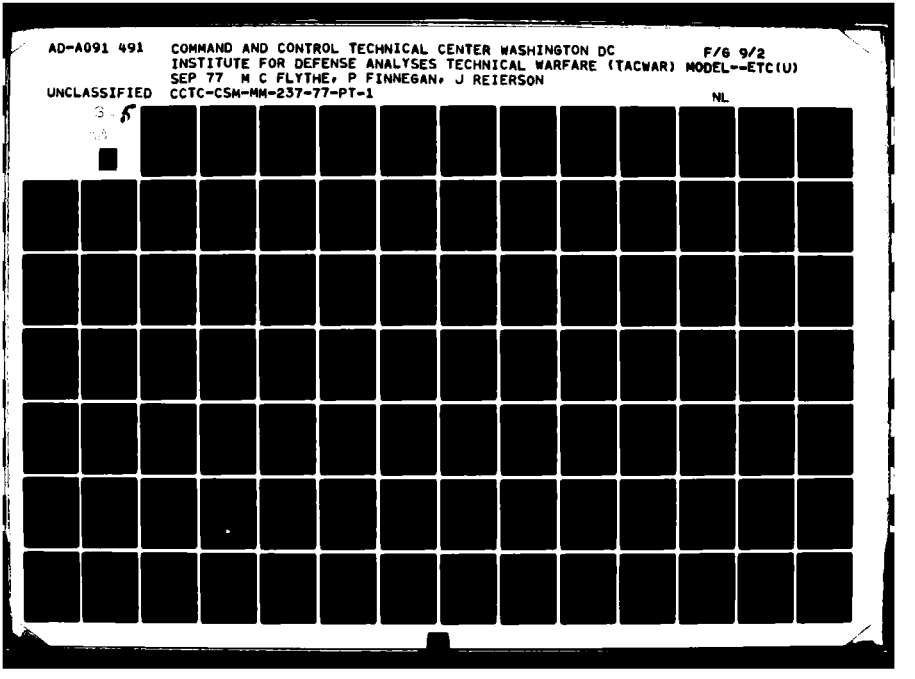
This fraction is computed by multiplying previous fractions by P1(IR,IQ), assuming that TPRD(ICMP,1)=1 before the first weapon detonates on the airbase. Control passes to the next section of coding.

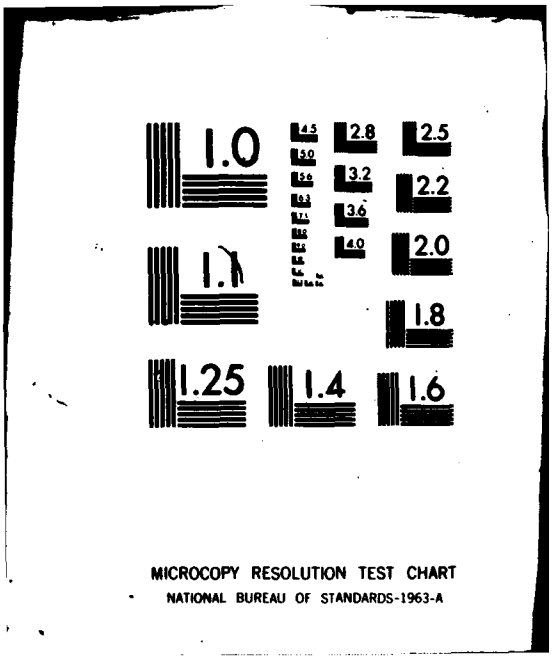
n. Section 330 - Calculate Casualties to Civilian Personnel. After target dimensions are determined using array AFDIM, control is passed to section 600 to calculate civilian casualties with flags IL=2 and KCCSW=2. Control returns to section 330 from section 600 and the fractions of civilians on the airbase killed and incapacitated are updated.

If the current weapon, INPACR, is not the last to fall on this airbase (INPACR<INPATG), control passes to the end of the Do loop on I, the nuclear fire mission. If the current weapon is the last weapon targeted against this airbase, control passes to section 340 to update the effectiveness of the airbase.

o. Section 340 - Update Airbase Arrays After Last Nuclear Weapon. Subroutine CVFW is called to calculate airbase effectiveness after INPATG weapons have impacted on it. The fraction of surviving military personnel at the airbase, FSPAB(IAB), is calculated from the sum of the casualties at each parking area. The numbers of sheltered and unsheltered aircraft of each type in each parking area is obtained and combined with the appropriate kill fractions to obtain the numbers of aircraft and shelters destroyed. The numbers of shelters, aircraft, and personnel are updated to reflect results of the nuclear damage in this cycle. Control returns to section 300 to process the next nuclear fire mission I, directed against a new airbase. If all values of I have been considered, control passes to the section 400 to access damage to supply depots.

p. Section 400 - Damage Assessment for Supply Depot Targets. Control passes to this section after damage to all combat divisions and airfields on side L2 has been assessed. This section consists of a Do loop over each nuclear fire mission I. For each nuclear fire mission against a supply depot (IWLTLO(I,L)=3), weapon characteristics are obtained and subroutine WRADVN is used to calculate weapon radius WRSP. The target radius is calculated using the supply level SUPIN(IDSP) and input variable SQMPTS(L2). Subroutine OFFCOV calculates COV, the fraction of the target area covered.


AD-A091 491
COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC
INSTITUTE FOR DEFENSE ANALYSES TECHNICAL WARFARE (TACWAR) MODEL--ETC(U)
SEP 77 M C FLYTHE, P FINNEGAN, J REIERSON
UNCLASSIFIED
CCTC-CSM-MM-237-77-PT-1
F/G 9/2
NL


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
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

The supplies destroyed (TSPK) is set to SUPIN(IDSP) multiplied by COV. The supplies destroyed are then subtracted from SUPIN(IDSP). Control is passed to section 600 with flags IL=2 and KCCSW=3 to calculate civilian casualties. When control returns to section 400 after this calculation, results are printed if IPRS=1 and the Do loop on I ends. After all nuclear fire missions have been considered, control passes to section 500 to calculate damage to surface-to-surface missile sites.

q. Section 500 - Damage Assessment for Surface-to-Surface Missile Site Targets. Control passes to this section after damage to all combat divisions, airfields, and supply depots on side L2 has been assessed. This section consists of a Do loop over each nuclear fire mission I.

For each nuclear fire mission against a surface-to-surface missile site (IWLTLO(I,L)=4), weapon characteristics are obtained and subroutine WRADVN is used to calculate weapon radii WRS (severe damage) and WRM (moderate damage). The target radius is set to RDSSMS(ISM,L2).

Subroutine OFFCOV is used to calculate COVS and COVM, the fractions of the target area suffering severe and moderate damage, respectively. The number of operating missile sites on side L2 is updated to reflect the fraction of a site incapacitated by this fire mission.

Control is passed to section 600 with IL=2 and KCCSW=4 to calculate civilian casualties. When control returns to section 500 after this calculation, results are printed if IPRS=1 and the Do loop on I ends.

After all fire missions have been considered, civilian casualties and fatalities are stored in array WYLDL. The Do loop on attacking side L, which was begun in section 100, ends. After this Do loop, control returns to subroutine NUC6.

r. Section 600 - Calculate Civilian Casualties and Fatalities. Control is passed to this section from other parts of DAMEVL to assess the effect of weapon I on CCTEM and CFTEM, the total civilian casualties and fatalities for this cycle. The lethal dose DLETH and casualty dose DCAS is specified. If it has not been done already, subroutine PREFN is called to calculate a table of unattenuated prompt gamma and neutron dose levels for weapon I.

Civilian casualties and fatalities are calculated in a Do loop on protection category IPC. Fatalities and casualties are calculated using equations of the same form, but with different arguments. Civilian fatalities are calculated as follows. The area outside the target covered by lethal weapon effects for civilians in protection category IPC is PKS. PKS is calculated by first setting the lethal weapon radius YU to the maximum of YL=FN(DLENTH,IPC,DOSG,DOSN), the radius covered by a lethal radiation dose, and WRL, the weapon lethal blast radius determined by function WRADVN. Subroutine OFFCOV calculates COVL, the fraction of the target of radius TT covered by the weapon.

Then

$$PKS = \pi(YU)^2 - \pi(TT)^2 * COVL.$$

The total civilian fatalities due to this weapon is

$$\left(\sum_{IPC=5}^{8} PKS*PCIV(IPC) \right) \frac{*PDENSS}{1,000,000}$$

where

PCIV(IPC)= FCPPC(IPC-4,IL)

= fraction of civilians in protection category IPC

PDENSS = civilian population per square kilometer.

Local variable ARL is the fraction of civilian fatalities.

$$ARL = \sum_{IPC=5}^{8} PKS*PCIV(IPC)$$

After civilian casualties and ALC, the fraction of casualties, have been calculated using similar equations, control returns to other sections of DAMEVL via index KCCSW.

2.2.6.25 PAREA. Subroutine PAREA allocates aircraft by type on an actual airbase IAF to parking areas and shelters. Sheltering is done according to the priority specified by

input array IPSHLA(IAC,L). Airfields are described on the basis of a prototype airfield as discussed in appendix D of reference 2. The parameters calculated by this subroutine are used for the calculation of nuclear damage to airfield targets.

2.2.6.25.1 Programming Specifications. The following table summarizes the principal specifications of subroutine PAREA:

Characteristic	Specification	
Formal parameters	IAF	= Airfield index
	NSHPA(J)	= Calculated number of aircraft shelters in Jth parking area
	NACPA(J)	= Calculated number of aircraft in Jth parking area
	NACTPA(I,J)	= Calculated number of aircraft of priority I in Jth parking area
	NA	= Unused index
	NSH	= Total number of shelters on airfield IAF
	NPA	= Total number of occupied parking areas on airfield IAF
	L	= Index specifying the side occupying the airbase
Common blocks	Blank Common	
Subroutines called	None	
Called by	DAMEVL	

2.2.6.25.2 Logic Functions. Information for airbase IAF is contained in arrays IWORD and IMAGE1. The number of shelters, NSH, and the total number of occupied parking areas, NPA, are determined from that information. NSHPA(J) is calculated by allocating shelters uniformly to occupied parking areas. The number of aircraft NACT(I) of each type IDTAC(I) is obtained from array IPSHLA(IDTAC(I),L).

Aircraft are allocated to shelters by aircraft priority to obtain NACTPA(I,J). If all shelters are occupied and unassigned aircraft remain, an equal number of aircraft are allocated on each parking area. Finally NACPA(J) is obtained by summing over all aircraft priorities in each parking area J.

2.2.6.26 FN. Function FN estimates the range at which an effective biological dose of radiation of level D, is felt by personnel in protection category IPC.

2.2.6.26.1 Programming Specifications. The following table summarizes the principal specifications of function FN:

Characteristic	Specification
Formal parameters	D = Dose level in roentgens
	IPC = Personnel protection category
	DOSG(L) = Unattenuated prompt gamma ray dose at range TRNG(L)
	DOSN(L) = Unattenuated prompt neutron dose at range TRNG(L)
Common blocks	Blank Common, QINRPR
Subroutines called	None
Called by	DAMEVL

2.2.6.26.2 Logic Functions. The total attenuated prompt radiation dose at range TRNG(L) is equal to

TRGM(IPC)*DOSG(L) + TRNT(IPC)*DOSN(L)

In the above formula TRGM(IPC) and TRNT(IPC) are the fractions of gamma and neutron radiation, respectively, that are transmitted through protection category IPC. The value of L is found such that dose level D is between the total attenuated dose at range TRNG(L) and the total attenuated dose at range TRNG(L-1). The value of FN is then obtained by a linear interpolation of range versus the natural logarithm of total dose.

2.2.6.27 PREFN. Subroutine PREFN calculates prompt unattenuated neutron and gamma ray doses at specified distances from a nuclear detonation. These dose values are used function subroutine FN.

2.2.6.27.1 Programming Specifications. The following table summarizes the principal specifications of subroutine PREFN:

Characteristic	Specification
Formal parameters	YIELD = Weapon yield in kilotons
	IHOB = Height of burst indicator, =0 for surface burst, =1 for blast optimized air burst
	DOSG(I) = Prompt gamma dose at range TRNG(I)
	DOSN(I) = Prompt neutron dose at range TRNG(I)
Common blocks	QINRPR
Subroutines called	QKINR
Called by	DAMEVL

2.2.6.27.2 Logic Functions. A set of 23 ranges (TRNG(I)) are defined by a data statement. Parameters required for subroutine QKINR are calculated. For each value of I, subroutine QKINR is called to calculate the prompt radiation dose at a radius of RO meters from the weapon burst.

$$RO = \sqrt{(TRNG(I))^2 + (\text{height of burst})^2}$$

DOSG(I) is set equal to the sum of the fission product gamma dose DOSIKE and the secondary gamma dose DOSGAM. DOSN(I) is set equal to the neutron dose.

2.2.6.28 QKINR. Subroutine QKINR calculates initial unattenuated neutron and gamma ray doses at a radius RO from a nuclear detonation. QKINR is a fast-running version of subroutine DOSEINR developed at the National Bureau of Standards (see reference 7).

2.2.6.28.1 Programming Specifications. The following table summarizes the principal specifications of subroutine QKINR:

Characteristic	Specification
Formal parameters	None
Common blocks	QINRPR
Subroutines called	None
Called by	PREFN

2.2.6.28.2 Logic Functions. The first part of this subroutine calculates DOSIKE, the prompt gamma ray dose from fission products. This calculation is made using a curve fitting algorithm to approximate the results of more detailed calculations which require a numerical integration to account for the rising fireball. After DOSIKE has been calculated, the radiation due to secondary gamma rays (DOSGAM) and the prompt neutron dose (DOSNEU) are calculated using equations similar to those in the NBS subroutine DOSEINR.

2.2.6.29 DOSLIM. Subroutine DOSLIM calculates the radiation levels RU and RB which will transfer personnel from radiation pool IR to upper and lower boundaries of radiation pool IS.

2.2.6.29.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DOSLIM:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common

Characteristic	Specification
Subroutines called	None
Called by	DAMEVL

2.2.6.29.2 Logic Functions. Radiation pool IS is specified by the input variable RADBPT(IS), the maximum radiation level for personnel in pool IS. The radiation level RU required to effect a transition from the center of radiation pool IR to the upper boundary of pool IS is calculated from:

$$RU = RADBPT(IS)-(1/2)*[RADBPT(IR) + RADBPT(IR-1)]$$

The radiation level RB required to effect a transition from the center of radiation pool IR to the lower boundary of pool IS is:

$$RB = RADBPT(IS-1)-(1/2)*[RADBPT(IR) + RADBPT(IR-1)]$$

2.2.6.30 WRAD. Function WRAD calculates the nuclear weapon blast radius against personnel for use in DAMEVL. This function is accessed by a dummy call and is not used for calculations in the unclassified version of DAMEVL. To make damage calculations with classified vulnerability data, function WRAD must be modified and calls to WRAD inserted in the appropriate places in DAMEVL.

2.2.6.30.1 Programming Specifications. The following table summarizes the principal specifications of function WRAD:

Characteristic	Specification
Formal parameters	YIELD = Weapon yield in kilotons
	IPC = Personnel protection category
	IHOB = Height-of-burst indicator (IHOB=1 indicates a surface burst)
	TSIG = Standard deviation associated with calculated weapon radius

Characteristic	Specification
Common blocks	Blank Common, NUCCOM
Subroutines called	None
Called by	DAMEVL

2.2.6.30.2 Logic Functions. Weapon radius WRAD is calculated from

$$WRAD = \left(\frac{C}{3.281}\right)\left(YIELD\right)^{A}$$

The parameter C is obtained from array RDCFA for an air burst (RDCFS for a surface burst). The parameter A in the above equation is obtained from array RDEXPA (RDEXPS for a surface burst). TSIG is obtained from array RDSIGA (RDSIGS for a surface burst). The value of TSIG returned to DAMEVL is the value in the RDSIGA (or RDSIGS) array divided by 100.

2.2.6.31 WRADVN. Given nuclear weapon yield (YIELD), scaled height of burst HOB, and target vulnerability IVN, function WRADVN determines the weapon radius. Information on these calculations can be found in reference 8 and in appendix C of reference 2. To run DAMEVL with classified vulnerability data, the data statements in WRADVN must be modified.

2.2.6.31.1 Programming Specifications. The following table summarizes the principal specification of function WRADVN:

Characteristic	Specification
Formal parameters	YIELD = Weapon yield (in kilotons)
	HOB = Scaled height of burst = (1/100)* (actual HOB in feet)* $(YIELD)^{1/3}$
	IVN = The VN (vulnerability number) input as BCD code in the format XXYZ where XX is the VN number, Y is P or Q and Z is the K-factor

Characteristic	Specification
	WSIG = A function value returned as 0.2 for P VN numbers or 0.3 for Q VN numbers
	IDONUT = A function value returned as 1 unless the VN is out of range in which case a value of 3 is returned
Common blocks	None
Subroutines called	None
Called by	DAMEVL

2.2.6.31.2 Logic Functions. A table of weapon radii TABWR(NVV,IPQ,IN) is specified by data statements. The arguments for table entries relate to VN number, VN type (P or Q) and HOB, respectively. The coding begins with logic to unpack the VN number IVN. Next, the VN number is adjusted by an amount DVN which is calculated using the K-value XK, YIELD and VN type (P or Q). Finally, WRADVN, the weapon radius is obtained by interpolating between entries in table TABWR. For a burst height of less than 100 feet (HOB<1), a search is made of HOBs until a value that yields a maximum weapon radius is found. This value of weapon radius, as well as the corresponding height of burst, is returned.

2.2.6.32 OFFCOV. Subroutine OFFCOV is called by the chemical and nuclear damage subroutines. OFFCOV calculates the expected coverage of a uniform value circular target by a weapon with a uniform circular damage function aimed at an offset aim point with Gaussian aiming errors.

2.2.6.32.1 Programming Specifications. The following table summarizes the principal specifications of subroutine OFFCOV:

Characteristic	Specification
Formal parameters	CAP = Input weapon circular error probable (CEP)
	TER = Target radius

Characteristic	Specification
	OEF = Offset of weapon aim point from target center
	WAN = Weapon radius
	COV = The calculated value of the fraction of the target covered by the weapon
Common blocks	None
Subroutines called	CIRCOV
Called by	CHEMDAM, DAMEVL

2.2.6.32.2 Logic Functions. OFFCOV calculates COV, the approximate expected fractional coverage of a uniform circular target by a "cookie cutter" weapon with an offset aim point and Gaussian delivery error. COV is calculated using semi-empirical equations which represent numerical fits to coverage values obtained from numerical integrations. Use of such equations results in a faster operating algorithm than could be obtained by doing the numerical integration. The coverage values obtained from OFFCOV are almost always within 10 percent and usually within 3 percent of the correct values. Reference 2, pp. C-2 through C-5 give additional details on OFFCOV.

The arguments CAP, TER, OEF, and WAN may be specified in terms of any consistent set of units. Computations in OFFCOV use the following variables normalized to the input weapon radius WAN:

CEP = CAP/WAN = normalized weapon CEP

TAR = TER/WAN = normalized target radius

OFFSET = OEF/WAN = normalized offset distance

WPN = 1 = normalized weapon radius.

If TAR < 0.2, i.e., target radius is less than 0.2 times weapon radius, the target is approximated by a point target and COV is calculated in section 10 of the coding using subroutine CIRCOV. For TAR $\geq$ 0.2, COV is calculated in

section 20 by combining PNO, the target coverage for zero offset distance, with terms which depend on OFFSET.

2.2.6.33 SIMCN. Subroutine SIMCN is called by the chemical and nuclear damage subroutines to calculate the cumulative normal distribution function.

2.2.6.33.1 Programming Specifications. The following table summarizes the principal specifications of subroutine SIMCN:

Characteristic	Specification
Formal parameters	ARG = Argument of cumulative normal distribution function
	COV = Value of cumulative normal distribution function returned by SIMCN
Common blocks	None
Subroutines called	None
Called by	LINFR, CIRCOV, CHEMDAM, SIRCOV

2.2.6.33.2 Logic Functions. SIMCN calculates the value of the cumulative normal function

$$P(ARG) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{ARG} e^{-(t^2/2)} dt$$

using a four-term approximation found in reference 9, pp. 299, 931. SIMCN returns the value of COV

where

COV = P(ARG) = (1/2)*(1 + CUP(ARG/2)) if ARG≥0

or

COV = P(ARG) = (1/2)*(1 - CUP(ARG/2)) if ARG<0

CUP, the Error Function, is calculated using the expansion

$$CUP(X) = 1 - \left(\frac{1}{(1+A_1X+A_2X^2+A_3X^3+A_4X^4)^4} \right)$$

where

$A_1 = 0.278393$

$A_2 = 0.230389$

$A_3 = 0.000972$

$A_4 = 0.078108$

The error is less than 0.00005 in this approximation.

2.2.6.34 SIRCOV. Subroutine SIRCOV calculates COV, the expected fraction of a circular Gaussian target covered by a weapon with offset aiming and aiming errors. The calling parameter SIGT specifies the sharpness of the cutoff of the probability of damage as a function of distance from target center. The algorithm is described in appendix C of reference 2, and in reference 10.

2.2.6.34.1 Programming Specifications. The following table summarizes the principal specifications of subroutine SIRCOV:

Characteristic	Specification
Formal parameters	C = Circular error probable (CEP) of the weapon
	DD = Offset distance from center of target to aim point
	WPN = Weapon radius
	T = Target radius (A-95) which contains 95 percent of the target value

Characteristic	Specification
	SIGT = Standard deviation of the probability-of-kill versus distance function, expressed as a fraction of WPN
	COV = The value returned by this subroutine, the fraction of the target value killed
Common blocks	None
Subroutines called	CIRCOV, SIMCN
Called by	DAMEVL

2.2.6.34.2 Logic Functions. If SIGT≤0.301, then COV is calculated by the circular coverage function CIRCOV(CEPM,D, WPN,COV) where CEPM, the adjusted CEP, is defined by:

$$CEPM = 1.1774\left[\left(\frac{C}{1.1774}\right)^2 + (SIGT*WPN)^2 + \left(\frac{T}{2.44775}\right)^2\right]^{1/2}.$$

If SIGT>0.301, the algorithms described in reference 2 are implemented. Different algorithms are used for each of the following cases: zero offset (D≤0.001*SIG), SIGT≤0.45 and nonzero offset, SIGT>0.45 and nonzero offset.

2.2.6.35 CIRCOV. Subroutine CIRCOV is called by nuclear and chemical damage subroutines to compute the "circular coverage function" using an approximation given in reference 9, p. 940. The value returned from CIRCOV is the approximate value of a Gaussian function integrated over a circle that is offset from the origin of the circular Gaussian distribution. This value is also the probability that a point target is covered by a circle whose center is aimed with Gaussian errors at the offset point.

2.2.6.35.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CIRCOV:

Characteristic	Specification
Formal parameters	CEP = Circular error probable of the aim point (1.1774 times the standard deviation of the circular Gaussian function)
	OFFSET = Distance between the center of the integrating circle and the aim point
	WPN = Radius of the target circle
	COV = Value returned by subroutine CIRCOV
Common blocks	None
Subroutines called	SIMCN
Called by	SIRCOV, OFFCOV

2.2.6.35.2 Logic Functions. CIRCOV calculates COV, the value of a circular Gaussian function with standard deviation $\sigma = \frac{CEP}{1.1774}$ integrated over a circle of radius WPN whose center is offset by a distance OFFSET from the center of the Gaussian. The algorithm used is as follows:

Let

$$RL = \frac{OFFSET}{\sigma}$$

and

$$R = WPN/\sigma,$$

then, for

$$R \leq 1,$$

$$COV = \frac{2R^2}{4+R^2} * \exp\left(\frac{-2(RL)^2}{4+R^2}\right)$$

for $R \geq 1$,

$$COV = SIMCN(ARG,COV)$$

where subroutine SIMCN(ARG,COV) calculates

$$COV = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{ARG} e^{-x^2/2} dx,$$

the value of the cumulative normal distribution function.

for $1<R<5$,

$$ARG = \frac{\left(\frac{R^2}{2+(RL)^2}\right)^{1/3} - \left[1 - \frac{(2/9)(2+2(RL)^2)}{(2+(RL)^2)^2}\right]}{\left[(2/9)\left(\frac{2+2(RL)^2}{(2+(RL)^2)^2}\right)\right]^{1/2}}$$

for $R>5$,

$$ARG = R - \sqrt{(RL)^2-1}$$

2.2.7 LINKF. This subsection describes the routines in LINKF. These routines comprise the chemical combat model. Subroutines ZNDST, SIMCN, OFFCOV, and CIRCOV, which belong both to the nuclear (LINKE) and chemical (LINKF) models are discussed in section 2.2.6 and the discussion is not repeated in this section.

2.2.7.1 CHEM. Subroutine CHEM is the main calling program for the chemical routines.

2.2.7.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, LOCAA2, CHCW, LOCAL2
Subroutines called	CHEM6, CHEM1, CHEM2, CHEM3, CHEM4, CHEM5, CLR
Called by	TMAIN

2.2.7.1.2 Logic Functions. If KFLAG≠0, subroutine CHEM6 is called to determine chemical employment levels and the number of chemical weapon systems, to perform any necessary decontamination, and to allocate supplies of chemical munitions to division, sector, and theater weapon systems. If KFLAG=0, the call to CHEM6 is skipped. If upon return from CHEM6, KFLAG=1 or 2, the rest of the chemical model is skipped and control is returned to TMAIN. Otherwise, the rest of CHEM is executed as described below. Subroutine CLR is called three times to initialize certain arrays that are used later on in the cycle to accumulate weapon usage. Subroutine CHEM1 is called to construct a chemical weapon priority list and a target priority list. If, upon return from CHEM1, KFLAG=-1, the rest of CHEM is skipped. Otherwise CHEM2 is called to determine the expected number of battle field targets (subunits) detected. If variable IPOPCH=0, subroutine CHEM3 is called to determine the fraction of targets precluded from targeting due to civilian population constraints. Next, CHEM4 is called to complete the assignment of weapons to targets. Finally, CHEM4 is called to calculate the damage inflicted on the targets.

2.2.7.2 KCODE. Function KCODE packs the weapon data listed below into a single word.

2.2.7.2.1 Programming Specifications. The following table summarizes the principal specifications of function KCODE:

Characteristic	Specification
Formal parameters	IWC = Index to weapon category
	IWS = Index to weapon system
	IPOS = Index to weapon system position
	IA = Index to chemical agent
	K = Index to dissemination mode
Common blocks	None
Subroutines called	None
Called by	CHEMWPS

2.2.7.2.2 Logic Functions. The parameters specified above are packed into a 5-digit word from left to right as follows. First, parameter IWC is multiplied by the fourth power of ten (1000) in order to pack the fifth or leftmost digit into the word. Added to this number is the next parameter, IWS, multiplied by the third power of ten (1000) in order to pack the fourth digit. The remaining parameters are each multiplied by consecutively smaller powers of ten and added together to form the 5-digit packed word.

2.2.7.3 KDCODE. Function KDCODE unpacks the weapon data listed below from the specified INDEX.

2.2.7.3.1 Programming Specifications. The following table summarizes the principal specifications of function KDCODE:

Characteristic	Specification
Formal parameters	INDEX = Word containing packed data
	IWC = Index to weapon category

Characteristic	Specification
	IWS = Index to weapon system
	IPOS = Index to weapon system position
	IA = Index to chemical agent
	K = Index to dissemination mode
Common blocks	None
Subroutines called	None
Called by	BFTGTS, PREAGDM, DUCINV, CHEMWPS, KADMC, CHEM5

2.2.7.3.2 Logic Functions. The 5-digit representation of the packed weapon data contained in INDEX is unpacked from left to right in the following manner. First, INDEX is divided by the fourth power of ten (10000) in order to unpack the fifth or leftmost digit. This value is the index to the weapon category and is stored in variable IWC. Now IWC is multiplied by the same power of ten and the product is subtracted from INDEX. This new 4-digit value of INDEX is then divided by the third power of ten (1000) in order to unpack the fourth digit which is the index to the weapon system, and is stored in variable IWS. The remaining parameters located in the second and first digits are unpacked in a similar manner.

2.2.7.4 CHEM6. Subroutine CHEM6 is the calling program for the routines that determine chemical employment levels, determine the number of chemical weapon systems, allocate chemical supplies and perform decontamination of equipment in contamination pools.

2.2.7.4.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM6:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, CHCW, LOCAA2, LOCAL2

Characteristic	Specification
Subroutines called	CHEMLEV, EQUIP, CHEMSUP, DECON
Called by	CHEM

2.2.7.4.2 Logic Functions. First, CHEM6 calls subroutine CHEMLEV to determine chemical employment levels. If, upon return to CHEM6, KFLAG=2, the rest of CHEM6 is skipped. Otherwise, subroutine DECON is called to decontaminate any equipment that has been contaminated by chemical agents. Finally, if the index to the employment level in each sector and for each side against a given target category is nonzero, subroutine EQUIP is called to determine the number of chemical weapon systems, followed by a call to subroutine CHEMSUP to allocate chemical supplies.

2.2.7.5 CHEMLEV. Subroutine CHEMLEV determines employment levels for chemical munitions for each side according to current conditions and the tactical chemical employment doctrine selected by the user.

2.2.7.5.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEMLEV:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL2, LOCAA2, CHCW
Subroutines called	None
Called by	CHEM6

2.2.7.5.2 Logic Functions. The calculations of this routine are made for each sector IS and each side L. Each of the following four stimuli may be a criterion for increasing the chemical employment level: (1) a preplanned decision to make a preemptive strike; (2) a response to a worsening tactical environment; (3) a response to the enemy's initial or increased use of nuclear weapons; and (4) a response to the enemy's use of chemical weapons. For each criterion, the program first checks to see if the user has selected the criterion and then, if necessary, determines if conditions require an increase in the chemical employment level.

a. Section 10 - Is Preemptive Strike Considered. If the first criterion is chosen by the user (INDC1=1) and the choice is not overridden because a strike is to end this cycle, then the program tests to see if the present cycle is the one in which a preemptive strike is to take place. If it is, then for each target category ITC the employment level IEML is set to the proposed level IEMLC1 provided the proposed level is higher than the present level.

b. Section 20 - Has a Tactical Event Occurred Which Stimulates Firing of Chemical Weapons. The second criterion has seven events, any of which may be selected by the user (INDC2=1) for changing the employment level. The testing of conditions for each of these events is described below.

The first event is border incursions. If the FEBA has moved into this side's territory as measured from the FEBA location at time zero, then, for each target type, employment level IEML is set to the proposed level IEMLC2, provided the proposed level is higher.

The second event is the advance within the sector beyond the advance in adjacent sectors. If the enemy is attacking in all adjacent sectors and, if the distance between the FEBA in sector IS and the FEBA in every adjacent sector is greater than DPTHC2, then for each target type the employment level may be increased.

The third event is a cumulative enemy advancement in the sector of more than a specified distance. If the total FEBA advancement by the enemy in the sector is greater than the threshold depth DPTHC3 for side L, then, for each target type, the employment level may be increased.

The fourth event is an enemy advancement in the sector of more than a specified distance since the last cycle. If the enemy advancement since the last cycle exceeds the value of DPC2 for employment level LE, then the employment level IEMC may be increased to level LE for those target types selected by the user (ICDLT=1).

The fifth event is the cumulative loss of QRA aircraft beyond a specified level. If the fraction of QRA aircraft lost is greater than the threshold fraction THQRAC, then for each target type the employment level may be increased.

The sixth event is the cumulative loss of nuclear delivery systems (missiles and artillery) beyond specified levels. If the fraction of missiles (artillery) lost is greater than the threshold fraction THWPC for missiles (artillery), then for each target type the employment level may be increased.

The seventh event is the rate of advancement being too slow in sectors of main attack. If side L is not the theater attacker, or if the sector IS is not one of main attack, event 7 is not considered. If side L's total advancement since time zero does not exceed a specified distance DPTHC7 by day NDOBC7 and the side L advancement during the present cycle does not exceed a distance DPC7, then for each target type the employment level may be increased.

c. Section 30 - Is the Use of Nuclear Weapons Beyond Thresholds. If the third criterion is selected by the user (INDC3=1), then the program considers the number of nuclear weapons delivered by the enemy into target areas in the battlefield, region, and COMMZ. First, the number of nuclear weapons delivered to the active battle area (NWABA) is compared to the threshold levels NBC3 to determine the proposed employment level. Then for each target type against which the side wishes to raise its employment level (IDLBC3=1), the employment level may be increased. Next, for each target subtype, the number of nuclear weapons delivered to the region (NWREG) is compared to the threshold levels NRC3 to determine the proposed employment level. Then for each target type against which the side wishes to raise its employment level (IDLRC3=1), the employment level may be increased. Similarly, for each target subtype, the number of nuclear weapons delivered to the COMMZ (NWCZ) is compared to the threshold levels NCZC3 to determine the proposed employment level. Then for each target type against which the side wishes to raise its employment level (IDLCC3=1), the employment level may be increased.

d. Section 40 - Is the Use of Chemical Weapons Received Used as Threshold. If the fourth criterion is selected by the user (INDC4=1), then the program considers the number of chemical weapons delivered by the enemy into target areas in the battlefield, region, and COMMZ. First, the number of chemical weapons delivered to the active battle area (NCWABA) is compared to the threshold levels NCBC4 to determine the proposed employment level. Then for each target type against which the side wishes to raise its employment level (ICDLB=1), the employment level may be increased. Next for each target

subtype, the number of chemical weapons delivered to the region (NCWREG) is compared to the threshold levels NCRC4 to determine the proposed employment level. Then for each target type against which the side wishes to raise its employment level (ICDLR=1), the employment level may be increased. Similarly, for each target subtype, the number of chemical weapons delivered to the COMMZ (NCWCZ) is compared to the threshold levels NCZC4 to determine the proposed employment level. Then for each target type against which the side wishes to raise its employment level (ICDLC=1), the employment level may be increased.

2.2.7.6 EQUIP. Subroutine EQUIP determines the number of types of weapons that deliver chemical munitions for division, sector, and theater weapon systems for both the Blue side and the Red side.

2.2.7.6.1 Programming Specifications. The following table summarizes the principal specifications of subroutine EQUIP:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, LOCAL2, LOCAA2, CHCW
Subroutines called	None
Called by	CHEM6

2.2.7.6.2 Logic Functions. The logic that follows is executed the first time for the Blue side and the second time for the Red side.

a. Section 10 - Determine Number of Division Weapons in Each Sector for Each Division Chemical Weapon System. For each division chemical weapon system, an index to a particular weapon type (IW) is determined from array IDSWTC(IWS,L). Next, in each sector the number of divisions (NDS) in the active battle area is determined. Then the index of each division (ID) located in the active battle area is determined. Now the number of weapon types in a given division is obtained by computing the product of the actual number of weapon types in the division location and the fraction of the weapon type that represents the division chemical weapon system type. This number is summed over all divisions in the sector to

yield the number of division chemical weapons in a given sector for a particular division chemical weapon system and is given by array NDCWSI(IWS,IS,L).

b. Section 20 - Determine Weighting Factors of Aircraft in Each Sector for Making Air Allocations to Sector and Theater Systems. The sector and theater systems both include air weapon systems. The number of available aircraft in each region and the weighting factors of aircraft in each sector of a region need to be determined before making air allocations to the sector and theater systems. For each region, beginning with the highest numbered sector, the employment level for chemical weapons against a particular target category is determined. A weighting factor (WT) is determined from the relative fraction of aircraft assigned to the given target at the specified employment level and is summed over all sectors in a region to yield a region weighting factor (WTT). Also, for each type of aircraft, the number of available aircraft in each region (TACT) is determined by counting the number of successful CAS, ABA, and INTD sorties.

c. Section 30 - Determine Number of Sector Weapons in Each Sector for Each Sector Chemical Weapon System. The logic for determining the number of types of sector chemical weapons is as follows. For each sector chemical weapon system, array ISSWTC(IWS,L) gives the index to a particular weapon type. Indexes of 1 and 2 indicate medium-range and long-range sector missile systems, respectively. An index>2 indicates aircraft systems. The number of sector chemical weapon systems that are medium-range (or long-range) missile systems is determined by counting the number of medium-range (or long-range) missile sites in each forward sector times the fraction of medium-range (or long-range) missiles that represent the weapon system under consideration. The number of sector chemical weapon systems that are air systems is determined for each sector by obtaining the product of four factors. These are: (1) the number of available aircraft in each region (TACT), (2) the fraction of aircraft dedicated to chemical missions (FADTMC), (3) the fraction of air systems that represents the given sector chemical weapon system (FSSWTC), and (4) the ratio of the weighting factor of the assigned aircraft in the sector to the weighting factor of aircraft in the region.

d. Section 40 - Determine Number of Theater Weapons in Each Sector for Each Theater Chemical Weapon System. The logic for determining the number of theater chemical weapons

is identical to the logic for determining the number of sector chemical weapons with the exception that only long-range missiles are considered for theater missile weapon systems.

2.2.7.7 CHEMSUP. Subroutine CHEMSUP determines the supply of chemical munitions in each sector of the theater at the beginning of each combat cycle. Supplies are then allocated to division, sector, and theater pools for division weapon systems and to sector and theater pools for sector weapon systems.

2.2.7.7.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEMSUP:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, LOCAL2, LOCAA2
Subroutines called	None
Called by	CHEM6

2.2.7.7.2 Logic Functions. The logic of CHEMSUP is described in the following sections.

a. Section 10 - Allocate Chemical Rounds to Division, Sector, and Theater Pools for Division Weapon Systems. For each side the number of division chemical weapon systems (NIWS) is determined and for each weapon system, the number of chemical agents (NAGT) and dissemination modes (NDM) are determined. Next, the total number of chemical rounds allocated to the theater pool is determined from array NDCRTP. This amount is added to the number of chemical rounds allocated to the division and sector pools and the total is summed over all sectors to yield the total number of chemical rounds (NCR) in the division, sector, and theater pools for a given division weapon system, agent, and dissemination mode.

Now the total number of division chemical weapon systems is summed over all sectors and is stored in variable NDWS. For each sector, the ratio of a given division weapon system to the total number of division weapon systems is computed and stored in variable FRAC. Finally, the number of chemical rounds that are reallocated to the division and sector pools

in each sector for a given division weapon system, agent, and dissemination mode, is determined by obtaining the product of three quantities: (1) FRAC; (2) the fraction of rounds allocated to the division and sector pools; and (3) the total number of rounds allocated by the theater pool, given by NCR. Any excess chemical rounds are allocated to the theater pool for division weapons.

b. Section 20 - Allocate Chemical Rounds to Sector and Theater Pools for Sector Weapon Systems. The allocation of chemical rounds to sector weapon systems is similar to the procedure described in section 10. For each side, the number of sector chemical weapon systems is determined and for each weapon system, the number of chemical agents and dissemination modes are determined. Next, the total number of chemical rounds in the theater pool for each sector weapon system is determined. This number is added to the number of chemical rounds in the sector pool, and the total is summed over all sectors. The total number of sector weapon systems is used to calculate the fraction of sector weapon system chemical rounds in each sector that are allocated to the sector pool. Again, any excess rounds are allocated to the theater pool for sector weapons.

2.2.7.8 DECON. Subroutine DECON provides for the decontamination of all types of weapons in each division that have been contaminated by a certain dosage of chemical agent. This is accomplished by determining the current decontamination capability of the division and the effort required to decontaminate all of each type of equipment.

2.2.7.8.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DECON:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, LOCAL2, LOCAA2
Subroutines called	None
Called by	CHEM6

2.2.7.8.2 Logic Functions. For each side, the number of divisions at time zero is determined and for each division, the division type (IT) is determined. The current decontamination capability of the division (DCC) is then computed by obtaining the product of the decontamination capability of the division type at TOE strength level and the effectiveness of the division on defense, given by arrays DCONTD(IT) and EFFDD(ID), respectively. The effort required to decontaminate all of each type of division equipment (DCCEQ) is summed for each type of division ID and is obtained from the product of the current number of division type weapons that are contaminated, given by array CONEQ(ID,IW), and the effort required to decontaminate them, given by array DCONEQ(IW,L). If DCC>DCCEQ, then variable FRAC, used to compute the fraction of equipment decontaminated, is set equal to one. If DDC<DCCEQ, then FRAC is set equal to the value of the fraction DCC/DCCEQ. In either case, the fraction of weapons decontaminated for a given division is computed from the product of CONEQ and FRAC for each weapon in the division. These weapons are then reallocated to the current inventory of division weapons for each type of weapon, given by array WDIV(IW,ID).

2.2.7.9 CHEM1. Subroutine CHEM1 is the main calling program for the routines that construct priority lists of targets and weapons.

2.2.7.9.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM1:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, CHCW, LOCAL2, LOCAA2
Subroutines called	CHEMTAR, CHEMWPS
Called by	CHEM

2.2.7.9.2 Logic Functions. CHEM1 begins by calling CHEMTAR to create a list of targets for a given sector. Upon return, if KFLAG≠-1, subroutine CHEMWPS is called once for each side to create a list of weapons to use. Otherwise the call to CHEMWPS is skipped.

2.2.7.10 CHEMTAR. Subroutine CHEMTAR creates a single list of preferred chemical targets from battlefield targets, region targets, and COMMZ targets and arranges them in order of priority from highest to lowest. Battlefield targets are assigned the highest priority, region targets the next highest and COMMZ targets the lowest priority.

2.2.7.10.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEMTAR:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHCW, LOCAA2, LOCAL2
Subroutines called	None
Called by	CHEM1

2.2.7.10.2 Logic Functions. In the active battle area, battlefield targets are arranged in priority order by the type of target ISU (subunit) within each zone IZ. Each target is assigned a priority from array ICPRB(ISU,IZ,L). Also a unique index value is assigned to each target, indicating its position on the priority list being created. NCHBT is set equal to the index value of the last battlefield target.

Region targets are arranged in priority order by target type and target area. For each target type (ISUB), a target area (ITYP) inside the region is assigned a priority from array ICPRR(ITYP,ISUB,L). A unique index value is then assigned to each target, indicating its position on the priority list being created. NCHRT is set equal to the index value of the last region target.

Targets in the COMMZ are arranged similarly by target type. Each target type is assigned a priority from array ICPRCZ (ISUB,L) and each target is assigned an index value indicating its position on the priority list being created. NCHCT is set equal to the index value of the last COMMZ target.

2.2.7.11 CHEMWPS. The main function of subroutine CHEMWPS is to create a single list of chemical weapons and their characteristics, in order of priority, from weapons in the

division, sector, and theater weapon systems that have been allocated to a given sector. Within each agent/dissemination mode combination, the weapon is ordered by increasing weapon system response time, distance from the FEBA, range and CEP. Also, CHEMWPS uses function NCRINV to calculate the number of rounds available to each weapon system type.

2.2.7.11.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEMWPS:

Characteristic	Specification
Formal parameters	IS = Sector index
	L = Side index
Common blocks	Blank Common, CHA, CHCW, LOCAL2, LOCAA2
Subroutines called	CLR, NCRINV, KCODE, KDCODE
Called by	CHEM1

2.2.7.11.2 Logic Functions. The logic of CHEMWPS is described below by the following sections.

a. Section 10 - Determine Weapon Characteristics of Division, Sector, and Theater Weapon Systems. For each type of division chemical weapon system, the number of types of chemical agents used by that system (NAGT) is determined. Also, for each division weapon system, CHEMWPS determines the position number (JP) that represents the distance from the FEBA of the location of the firing system. For each agent type, the number of types of dissemination modes (KDM) is determined. Variable N is incremented for each weapon type utilizing an agent/dissemination mode combination and is used to index several arrays to determine the following weapon assignments: the weapon system response time, the average distance of the weapon system (based also on position) from the FEBA, the range, and the CEP. Variable N is also used to index arrays KAT and KDSM to determine the number of the chemical agent and dissemination mode under consideration, respectively. When all of the division weapon systems have been accounted for, N is equal to the total number of types of division chemical weapon systems.

The logic for determining the weapon characteristics of weapon types in the sector and theater weapon systems is identical. However, index N is not reset but is continuously incremented for sector and theater weapons until N is equal to the total number of weapon types in the division, sector, and theater weapon systems, inclusively.

b. Section 20 - Arrange Division, Sector, and Theater Weapons by Chemical Agent and Dissemination Mode. The logic that follows arranges each weapon type into a single list based on each combination of agent and dissemination mode regardless of weapon system. The weapon characteristics of each weapon type are listed in increasing order of maximum efficiency of use of the agent/dissemination mode under consideration.

Subroutine CLR is called three times in succession to clear arrays NSFRD, NSFRS, and NSFRT. These arrays are used to accumulate the number of rounds that have been fired from division, sector, and theater system chemical weapons. The arrays are only initialized at this point but are used later in the chemical subcycle as a firing constraint when weapon to target assignments are made.

Once the chemical weapons list is complete, a starting and ending position is determined for all weapons falling within each particular agent/dissemination mode combination. This is accomplished by comparing the agent index (KAT) and dissemination mode index (KDSM) with the number of the agent (KA) and the number of the dissemination mode (KD) for every weapon in the list. When a match is found, array KIWF is set equal to the starting position of the weapon in the list. The ending position is found by counting backwards up the list until another match is found between the agent/dissemination mode index and the number. Array KIWL is set equal to the ending position.

2.2.7.12 NCRINV. Function NCRINV is used by several of the chemical subroutines to determine the current inventory of chemical rounds for division, sector, or theater weapon systems of specified agent, dissemination mode, and position.

2.2.7.12.1 Programming Specifications. The following table summarizes the principal specifications of function NCRIVN:

Characteristic	Specification
Formal parameters	IWC = Weapon category index
	IWS = Weapon system index
	IA = Agent type index
	K = Dissemination mode index
	IPOS = Position (location)
	IS = Sector index
	L = Side index
Common blocks	Blank Common, CHA, LOCAL2, LOCAA2
Subroutines called	None
Called by	CHEMWPS, KADMC, BFTGTS, PREAGDM

2.2.7.12.2 Logic Functions. The current inventory of chemical rounds (NCRINV) for a given division weapon system is computed by obtaining the product of two quantities. The first is the number of rounds which are available to the specified agent of a given dissemination mode which have been allocated to the division pool. The second is the fraction of the given division weapon system that has been assigned to the specified division position. If NCRINV has been called from CHEM4, then the maximum number of rounds that can be fired (NCRFIR) by the given division weapon system from the specified position is computed also. Finally, the number of chemical rounds that are available is determined from the smaller of the two values, NCRINV or NCRFIR. However, if NCRINV has been called from CHEM1, then the value of variable NCRINV is returned. The logic for determining the number of chemical rounds available to sector and theater weapon systems is identical to the logic described above.

2.2.7.13 CHEM2. Subroutine CHEM2 determines the expected number of battlefield chemical targets (subunits) detected.

2.2.7.13.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM2:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL2, LOCAA2, CHA, CHCW
Subroutines called	None
Called by	CHEM

2.2.7.13.2 Logic Functions. CHEM2 determines the number of potential chemical targets for sector KISS, as requested by subroutine CHEM. The calculations in the routine are performed for each side L, for each subunit ISU, each zone IZ, and each division IDS in the active battle area of the sector. The program first tests to see if the target is allowable at the current level of employment. If not, no further calculations are made for this division. Otherwise, CHEM2 determines the expected number of targets detected in the zone (NPT) from the product of: (1) PSZDDS(ISU,IZ,IDS), the fraction of units of a type ISU detected in zone IZ in division IDS; (2) NSUTD(ISU,ID), the number of subunits of type ISU in the division; and (3) FSUAZ(ISU,IZ,IT), the fraction of units of type ISU allocated to zone IZ by division type IT. Then, the maximum number of subunits allowed to be targeted (MAXNPT) is determined as the product of the number of subunits and the maximum fraction of subunits which can be targeted (FRMAXC) under the current employment level.

Local array NPTAS contains a running total of the number of potential targets by type subunit and by division, i.e., NPTAS is the sum of NPT, over all zones in the active battle area. If NPT for any zone will cause the sum NPTAS to exceed the maximum (MAXNPT), then NPT must be reduced for that zone as described below. If the maximum is reached in the first zone, then NPT and NPTAS are each set to the maximum value. For any other zones, NPT is the difference between the maximum value and the number of targets in lower indexed zones, and NPTAS is set to the maximum value.

If requested (IPRS=1), summary table C5, "Maximum Number of Chemical Targets (Subunits) in Active Battle Area," is printed.

2.2.7.14 CHEM3. Subroutine CHEM3 determines the fraction of chemical targets precluded by collateral damage constraints. It is called by subroutine CHEM for each sector KISS.

2.2.7.14.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM3:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL2, LOCAA2, CHA, CHCW
Subroutines called	None
Called by	CHEM

2.2.7.14.2 Logic Functions. CHEM3 first sets index IS to the sector value KISS passed from subroutine CHEM. The remaining calculations are performed for each side L.

Index LE is set to the current chemical employment level for the sector and target type 1 (divisions in the active battle area). Index JBA is set to the index of the active battle area in sector IS. If there are no side L divisions in the active battle area, no further calculations are performed for side L.

The distribution of population within each battle area is described by array PZPI which contains the percentage of the population held in city populations of five levels. Array POPLM defines the five levels, e.g., 5K, 19K, 25K, 50K, 100K. The input variable PMCHC indicates the city size which constitutes a collateral damage constraint for the current employment level LE. The percentage (ZOLIM) of the population of the current active battle area that resides in cities having populations of at least PMCHC is calculated, in most cases, by a linear interpolation within PZPI. To calculate ZOLIM, the subroutine compares PMCHC to the five values of POPCM to determine between which two levels the city size for constraint lies.

If the city size for constraint exceeds the largest level of POPLM, then the percent of people in the zone precluded by the targeting constraint (ZOLIM) is calculated as follows. The percent of population in cities greater than the largest population level (PZPI(JBA,5)) is multiplied by the ratio of the largest population level (POPLM(5)) to the city population for constraint (PMCHC). The resulting percentage is

ZOLIM. In this calculation, it is assumed that the distribution of population beyond the largest level is hyperbolic.

If the city size for constraint lies between levels KJB and KJ of POPLM, then the value of ZOLIM is calculated as follows. The difference in the values of PZPI for the KJB level and the KJ level is the percent population between these two levels. Since PMCHC falls between these two levels, only a portion of this percentage will be precluded from targeting. Assuming a linear distribution of population between the two levels, the percent population between the levels which can be targeted is proportional to the ratio of: (1) the difference between the city population for constraint and the population level KJB and (2) the difference between the population levels KJ and KJB. ZOLIM is set to the difference between the percent population value for level KJB and the percent population, between the two levels of KJB and KJ, which can be targeted.

If the city size for constraint is smaller than the first level of POPLM, the value of ZOLIM is determined by the same formula as in the preceding paragraph, with the lower level being set to 0 people (in thousands) and the percentage to 100.

ZOLIM is used as an estimate of the fraction of military targets within the battle area that will be precluded from chemical attack due to the proximity of a civilian population center. Therefore, the number of potential targets (NPT) of each subunit type within each zone of each division in the sector is multiplied by (1-ZOLIM) to account for civilian casualty constraints.

2.2.7.15 CHEM4. Subroutine CHEM4 is responsible for determining the order in which chemical weapons are assigned to individual targets. First the total number of possible targets is determined for each side. Then the target priority and target category are determined, and a call is made to the proper subroutine to complete the weapon-to-target assignments for battlefield, region, and COMMZ targets.

2.2.7.15.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM4:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CHA, CHCW, LOCAL2, LOCAA2

Characteristic	Specification
Subroutines called	BFTGTS, RGTGTS, CZTGTS
Called by	CHEM

2.2.7.15.2 Logic Functions. The total number of targets to be fired upon (NTAR) is determined for each side from variable NCHCT, which is the index of the last COMMZ target in the target priority list. For each individual target, a target category (J) is determined from array ICHTAR. If the target category falls within the range of battlefield targets in the priority list (i.e., J ≤ NCHBT) then a call is made to subroutine BFTGTS to determine the best weapon system to use against the battlefield target. If the target category lies within the range of region targets (i.e., NCHBT < J ≤ NCHRT), then a call is made to subroutine RGTGTS to select the best weapon system to use against the region target. Finally, if the target category falls outside the range (i.e., J > NCHRT), then it must fall within the range of COMMZ targets and subroutine CZTGTS is called to select the best weapon system to use against the COMMZ target.

2.2.7.16 DUCINV. Subroutine DUCINV reduces the current inventory of chemical rounds in the supply pool for a given weapon type by the amount of rounds used by the weapon.

2.2.7.16.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DUCINV:

Characteristic	Specification
Formal parameters	L = Index to side
	IW = Index to weapon type
	NCRDES = Desired number of chemical rounds
Common blocks	Blank Common, CHA, LOCAA2, LOCAL2
Subroutines called	KDCODE
Called by	BGTGTS, PREAGDM, KADMC

2.2.7.16.2 Logic Functions. First, function KDCODE is used to obtain the weapon category, the agent, and dissemination mode for the given weapon type. If the weapon is a division weapon, then the number of chemical rounds fired by weapon type IW is subtracted from the pool of rounds allocated to division weapons for the given agent and dissemination mode. Also, the number of rounds fired is added to the current inventory of rounds fired by the particular division weapon system type, given by array NSFRD.

The same logic is applicable to sector and theater weapon system types. The number of rounds fired by the weapon is subtracted from the pool of rounds allocated to the respective weapon system for a given agent and dissemination mode, and is added to the current inventory of rounds fired by the respective weapon system type. In addition, if any air missions have been flown, then the number of missions for each of the CAS, ABA, and INTD sorties is reduced proportionally by subtracting from each the ratio of the number of rounds fired to the total number of missions flown.

2.2.7.17 BFTGTS. Subroutine BFTGTS assigns weapons to battlefield targets for each division within a given sector. Given that the target is allowed under the restrictions of the level of chemical employment and engagement distance from the FEBA, a weapon system with the preferred agent and dissemination mode combination is determined. If such a weapon system is not available, an alternative selection is made. The first weapon system with the desired combination of agent and dissemination mode within range to reach the target is the one that is assigned.

2.2.7.17.1 Programming Specifications. The following table summarizes the principal specifications of subroutine BFTGTS:

Characteristic	Specification
Formal parameters	N = Number of targets for a given side
	JTAR = Target category of Nth priority target
	L = Side index (L = 1 for Blue side; L = 2 for Red side)

Characteristic	Specification
Common blocks	Blank Common, CHA, LOCAA2, LOCAL2
Subroutines called	ZNDST, KADMC, KDCODE, NCRINV, DUCINV
Called by	CHEM4

2.2.7.17.2 Logic Functions. The code in this subroutine is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Determine the Number of Divisions Containing the Subunit and Zone Type to be Targeted. If the current level of chemical employment against battlefield targets in a given sector is specified as zero, the target is not allowed and control is returned to CHEM4 to select another target. Additionally, if the target category is allowed but the subunit and zone to be targeted (detected in CHEM2) is not allowed under the current level of chemical employment, another target is chosen from CHEM4. Given that the target is allowed, the number of divisions which contain the subunit to be targeted within a given sector are determined. The logic for finding the preferred weapon system to assign to the target is applicable to each division in the sector.

Function ZNDST is used to calculate the distance of the targeted division from the FEBA. If the maximum distance that side L will allow for firing at the target under the given employment level, as specified by array RMXCH, is less than the distance of the division from the FEBA as calculated by ZNDST, then the number of potential targets (subunits) within the division is zero and the next division is selected for targeting.

b. Section 20 - Determine Which Weapon System has the Preferred Agent and Dissemination Mode. Once a division has been selected, a weapon system with the preferred or most desirable chemical agent and dissemination mode is determined. First, a chemical agent (either percutaneous or inhalatory) is selected from a list of agents that are ordered in terms of their strength from strongest to weakest. The agent chosen is determined by array KTARAG, which represents the preferred type of agent that would be expected to be used against the

given target. Next, the desired quantity of agent is determined by computing the product of the weight of the agent, given by array WTAGTD, and the number of potential targets within the division, given by array NPT. The preferred dissemination mode is selected from array KTARDM, which is the dissemination mode that would be expected to be used against the specified target type.

After the preferred chemical agent (KATOT) and dissemination mode (KDMOT) have been determined, function KADMC is used to compare the agent and dissemination mode of each weapon system classified on the priority list under the preferred choice. If KADMC determines that a certain weapon system is acceptable, then upon return to BFTGTS, variable NIWAS is incremented to count the number of assigned weapon systems and is used to index local arrays to make the following assignments: the chosen weapon system (IW) is assigned to array IWL; the target category (ITC) is assigned to array IWLIDS; the targeted zone (IZ) is assigned to array IWLTZN; the targeted subunit (ISU) is assigned to array IWLOCT; the chemical agent (KATOT) is assigned to array IWLKAT; the dissemination mode (KDMOT) is assigned to array IWLDSM; the number of potential targets (NPT) is assigned to array IWLBT and the number of desired rounds (NCRDES) is assigned to array IWLNCR. If a suitable weapon system has not been found, then an alternative choice of agent is determined from array KAGPTO (or KAGINO if the agent is inhalatory) and is stored in variable KATOT. Similarly, an alternative dissemination mode is determined from array KDMSC and is stored in variable KDMOT. The process for finding an acceptable weapon system is repeated as described above, with a new choice of preferred agent and dissemination mode.

c. Section 30 - Select Weapon System That Fires Maximum Amount of Agent. When all alternative choices of agent and dissemination mode have been tried and an acceptable weapon system has not been found, the weapon system that fires the maximum amount of agent on target is selected from array MAXIW. Function KDCODE is used to determine the category to which the weapon belongs (division, sector, or theater) and function NCRINV is used to determine the number of available chemical rounds for the weapon. The fraction of agent coverage is computed by dividing the number of available rounds (NCRAVL) by the number of desired chemical rounds (NCRDES) and is multiplied by the number of potential targets established previously. If the agent coverage is less than the desired level, given by array FAGTOT, no weapon assignment

is made and another division is chosen for targeting. If the agent coverage is equal to or greater than the desired level, then there is more agent than is necessary and subroutine DUCINV is called to reduce the inventory of chemical rounds for the applicable weapon type. The weapon system, target category, division, zone, subunit, number of potential targets and desired number of chemical rounds are assigned to the arrays as discussed before.

2.2.7.18 RGTGTS. Subroutine RGTGTS assigns chemical weapons to region targets. First the type of region target and target area is determined, and then a weapon system is chosen that has the most desirable combination of chemical agent and dissemination mode from the supply of available munitions.

2.2.7.18.1 Programming Specifications. The following table summarizes the principal specifications of subroutine RGTGTS:

Characteristic	Specification
Formal parameters	N = Number of chemical targets for a given side (priority ordered)
	JTAR = Target category of Nth priority target
	L = Side index (L = 1 for Blue, L = 2 for Red)
Common blocks	Blank Common, LOCAL2, CHA, LOCAA2
Subroutines called	AIRBASE, PREAGDM
Called by	CHEM4

2.2.7.18.2 Logic Functions. The code in this subroutine is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Determine Type of Region Target. The type of region target, ISUB, and the target area, ITYP, are determined from parameter JTAR. Next, the current level of chemical employment for region targets is determined. A test is performed to determine if the target, by type and area, is allowed at the specified level of employment. If

not, another target must be selected by CHEM4. If the target is allowed, weapon assignments are made on the basis of region target type and target area within the given sector. The four target types are airbases, supply nodes, missile sites, and divisions in the rear.

b. Section 20 - Assign Weapon System to Airbase Targets. For airbase targets, the target areas are either the sector forward or the sector rear. The logic for determining the number of airbase targets in either target area is similar. The depth of sector forward (rear) is determined from array NDFAB (NDRAB). The depth is measured in terms of the number of battle areas. A count is kept (NRT) of the number of targeted airbases within each battle area. If this count is zero for the sector forward (sector rear), another target must be selected. If NRT>0, subroutine AIRBASE is called to order the airbase targets by the number of aircraft and shelters. Then, for each targeted airbase, subroutine PREAGDM is called to select and assign a chemical weapon system that has the most desirable combination of chemical agent and dissemination mode.

c. Section 30 - Assign Weapon System to Supply Node Targets. For supply node targets, there are two target areas: the node feeding the active battle area and the nodes feeding all other battle areas within the region. For target areas of the first type, the logic is as follows: The number of the targeted supply node serving the active battle area in a given sector is determined. Then subroutine PREAGDM is called to select a chemical weapon system with the preferred or most desirable agent and dissemination mode.

For target areas of the second type, the logic is as follows. The depth of the sector forward and the sector rear (measured in terms of the number of battle areas) are determined. A count is kept of the supply nodes that supply each inactive battle area and which may be targeted. A supply node must belong to the opposite side for it to be counted as a targeted supply node. Those targeted supply nodes which supply more than one battle area in the region are counted as one target. Then, for each targeted supply node subroutine PREAGDM is called to select and assign a weapon system with the most desirable agent and dissemination mode.

d. Section 40 - Assign Weapon System to Missile Site Targets. The probability that a given side will detect enemy missile sites is determined from array PDSSMS and stored in

variable PCT. Now if PCT>PRTGTC, the maximum fraction of missile site targets that a given side will fire at under the specified employment level, then PRTGTC is assigned to PCT. The number of targeted missile sites is then determined by obtaining the product of PCT and the number of missile sites in the forward sector. Finally, subroutine PREAGDM is called to select and assign a weapon system with the preferred choice of chemical agent and dissemination mode.

e. Section 50 - Assign Weapon System to Targeted Divisions in Rear. Only those divisions in the rear region are targeted. First the number of divisions to be targeted (NDT) is obtained from the product of the number of divisions in the first inactive battle area of the given sector and the probability of any one being detected. For each targeted division and for each subunit type within a targeted division, there is an index (ISU) to the priority targeting of subunits in divisions in the rear, and an index (ID) of a specific rear division. The actual number of subunits to be targeted is then determined from the product of the current number of subunit targets, indexed by ISU and ID, and the probability that such a subunit is detected. For each targeted subunit, subroutine PREAGDM is called to select and assign a weapon system with the most desirable combination of chemical agent and dissemination mode.

2.2.7.19 CZTGTS. Subroutine CZTGTS assigns weapons to COMMZ targets. Weapon assignments are made according to COMMZ target type. If the target type is not allowed under the current level of employment, or there are no targets in the specified target category, then another target must be selected by CHEM4.

2.2.7.19.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CZTGTS:

Characteristic	Specification
Formal parameters	N = Number of chemical targets for a given side (priority ordered)
	JTAR = Target category of Nth priority target
	L = Side index (L = 1 for Blue; L = 2 for Red)

Characteristic	Specification
Common blocks	Blank Common, CHA, LOCAA2, LOCAL2
Subroutines called	AIRBASE, PREAGDM
Called by	CHEM4

2.2.7.19.2 Logic Functions. The type of COMMZ target, ISUB, is determined directly from parameter JTAR. The three types of COMMZ targets are airbases, supply nodes, and missile sites. If the target type is allowed under the current level of chemical employment, then the most forward battle area in the COMMZ (KIBA) is calculated. This calculation is performed differently depending on the side being targeted. For the Blue side, the depth of sector forward and sector rear (measured in terms of the number of battle areas) is subtracted from the index to the active battle area in the sector. For the Red side, the depth is added to the active battle area index.

For airbase targets, CZTGTS determines the number of airbases located in battle areas which are themselves located in the COMMZ. Subroutine AIRBASE is called to order the airbase targets. Then for each ordered target, subroutine PREADGM is called to complete the weapon to target assignments.

The number of targeted supply nodes in the COMMZ is determined by the following logic. For each battle area in the COMMZ the index of the supply node that serves it is determined from ISNBA. If the supply node is owned by the side being targeted, then the COMMZ target count is incremented and used to index array KCT, which contains a list of the targeted supply nodes. Those targeted supply nodes which serve more than one battle area in the COMMZ are counted as one target. Finally, subroutine PREAGDM is called to select and assign a weapon system with the preferred choice of chemical agent and dissemination mode.

The number of missile site targets is determined from the product of the actual number of enemy missile sites in the COMMZ, given by array SSMSRS, and the probability of their being detected, given by array PDSSMS (not to exceed the maximum probability of detection, given by array FRCMZC). Then subroutine PREAGDM is called to complete the weapon to target assignments by choosing a weapon system with the preferred choice of agent and dissemination mode.

2.2.7.20 PREAGDM. Subroutine PREAGDM completes the weapon to target assignments for region and COMMZ targets initially begun in subroutines RGTGTS and CZTGTS, respectively. PREAGDM selects a weapon system to use against the specified target from a preferred combination of chemical agent and dissemination mode. If a weapon system is not available, then alternative combinations are tried.

2.2.7.20.1 Programming Specifications. The following table summarizes the principal specifications of subroutine PREAGDM:

Characteristic	Specification
Formal parameters	ITGT = Index to target type
	ITC = Index to target category
	ISUB = Index to target subtype
	ISU = Index to division subunit
	IZ = Index to division zone
	IDS = Index to division
	ITYP = Index to target area within ISUB
	KRT = Index to the number of targets within ISUB
	NRT = Number of target subtypes
Common blocks	Blank Common, CHA, LOCAA2, LOCAL2
Subroutines called	CLR, AIRDIST, KADMC, KDCODE, NCRINV, DUCINV
Called by	RGTGTS, CZTGTS

2.2.7.20.2 Logic Functions. First PREAGDM determines the current level of chemical employment (LE) for the given target category. If LE=0, control is returned to the calling program. Otherwise, the target subtype is determined. If

the index to the employment level for the given target subtype is zero, control is returned once again to the calling program. The four target subtypes are airbases, supply nodes, surface-to-surface missile sites, and divisions in the rear.

For all targets except division targets the preferred agent and dissemination mode are determined from arrays KPRAG and KPRDM, respectively. The expected agent and dissemination mode to be used for division targets are determined from arrays KTARAG and KTARDM, respectively. Also for airbases, the following assumption is made concerning the target area. Depending on the specified employment level, the target is assumed to be a runway, the most dense parking area, or one of the active parking areas within the airbase.

The logic for selecting and assigning the most desirable weapon system for the given target is identical to the logic used in subroutine BFTGTS. To summarize, function KADMC is used to compare the agent and dissemination mode of a given weapon system with the preferred choice of agent and dissemination mode. If such a weapon system is available, then the weapon system, preferred agent and dissemination mode, division, subunit and zone (if applicable) and the number of rounds are assigned to local arrays. If such a weapon system is not available, another choice of agent and/or dissemination mode is tried and the process described above is repeated.

If, however, after all possible choices have been tried and a suitable weapon system has not been found, then the weapon system that fires the largest amount of agent on target and meets minimum requirements is chosen from array MAXIW. The number of targets that the agent can cover is determined from the number of weapons contained by the weapon system, firing constraints, and the fraction of agent coverage, which is expressed as the ratio of available chemical rounds to the desired number of chemical rounds. If the agent coverage is less than the minimum level, then no weapon assignment is made. If the agent coverage is satisfactory, then subroutine DUCINV is called to reduce the inventory of chemical rounds to be fired by the selected weapon system. The weapon assignments are made as described above.

2.2.7.21 KADMC. Function KADMC compares the preferred choice of chemical agent and dissemination mode with that of the given weapon type and checks certain other restrictions to determine if the weapon system is acceptable.

2.2.7.21.1 Programming Specifications. The following table summarizes the principal specifications of function KADMC:

Characteristic	Specification
Formal parameters	L = Side index
	IW = Index to weapon type
	KATOT = Preferred chemical agent type
	KDMOT = Preferred dissemination mode type
	ZND = Average distance of target from FEBA
	QUAN = Quantity of chemical agent needed to fire at target
	NCRDES = Desired number of chemical rounds
Common blocks	Blank Common, CHA, LOCAA2, LOCAL2
Subroutines called	DUCINV, KDCODE, NCRINV
Called by	BFTGTS, PREAGDM

2.2.7.21.2 Logic Functions. The weapon category for the given weapon type (IW) is calculated from function KDCODE. The three weapon categories consist of division weapon systems, sector weapon systems, and theater weapon systems. The logic for determining the acceptability of the given weapon type is identical for each of the three weapon systems.

First, the chemical agent and dissemination mode indexed by the given weapon is compared with the preferred agent (KATOT) and dissemination mode (KDMOT). If they do not match, the weapon is unacceptable. If they do, then the firing range of the weapon is tested. If the firing range is less than the combined distance of the weapon system from the FEBA and the average distance from the FEBA for deployment of the weapon in a given location, the weapon is again unacceptable.

Also, the weapon is unacceptable if the average distance of the target from the FEBA is less than the minimum engagement distance beyond the FEBA which the weapon can be fired. Given that the weapon is thus far acceptable, function NCRINV computes the current number of available chemical rounds for the weapon. This number is compared to the desired number of chemical rounds. If sufficient rounds are available to the weapon, then subroutine DUCINV is called to reduce the inventory of rounds by the amount used by the weapon. The value of variable KADMC is set equal to 1, indicating to the calling program that the weapon is acceptable. If insufficient rounds are available, the weapon is unacceptable (KADMC=0). However, before control is returned to the calling program, the ratio of available rounds to the desired number of rounds for the given weapon system (TMAX) is computed and compared with the percentage of preferred agent coverage, given by array RDMAX. If TMAX is greater than RDMAX, then TMAX is assigned to array RDMAX as the new maximum preferred agent coverage for the weapon. Consequently, the weapon index IW is assigned to array MAXIW as the chemical weapon that produces the maximum amount of the agent on target.

2.2.7.22 AIRBASE. Subroutine AIRBASE computes the weighted airbase value of each targeted airbase in order to establish a targeting priority. The airbase targets are ranked from highest priority to lowest corresponding to the weighted value of each airbase. The routine at entry point AIRDIST determines the distance (along the great circle) between a given airbase target and the FEBA by plotting the latitude and longitude coordinates of the airbase and the center of the active battle area at the FEBA.

2.2.7.22.1 Programming Specifications. The following table summarizes the principal specifications of subroutine AIRBASE:

Characteristic		Specification
Formal parameters	KCT	= Index to a particular airbase
	NCT	= Number of airbase targets
	L	= Index to side
	IS	= Index to sector

Characteristic	Specification
	IABT = Index to a particular airbase target
	DIST = Great circle distance between an airbase and the FEBA
	IPAAC = Index to an active parking area
	NPAAC = Number of active parking areas
Common blocks	Blank Common, CHA, LOCAA2, LOCAL2
Subroutines called	SECWTH, GDIST
Called by	RGTGTS, CZTGTS, PREAGDM

2.2.7.22.2 Logic Functions. The code of subroutine AIRBASE is divided by comment cards which label the functions of the logic.

a. Section 10 - Compute Weighted Airbase Values and Determine Targeting Priority. For every airbase that is either a region or COMMZ target, the number of types of aircraft assigned to the particular airbase is summed and placed in variable VLA. The number of shelters assigned to the airbase is placed in variable VLS. The weighting value of aircraft to shelters for the airbase is given by array WTCAST. The weighted value of the airbase, assigned to array VAL, is now computed using the formula

$$VLA*WTCAST + VLS*(1-WTCAST)$$

After the values have been computed for all airbases, array KCT, containing the airbases, and array VAL, containing the weighted airbase values, are reordered such that the airbase with the largest weighted airbase value (and thus having the highest priority) is listed first, continuing down to the airbase with the smallest weighted airbase value.

b. Section 20 - Find Distance Between Airbase Target and the FEBA. At entry point AIRDIST, the latitude and longitude coordinates of the targeted airbase are determined from array AF and placed in variables FLATAB and FLONAB, respectively. The cumulative ground distance between the eastern boundary of the given sector and the leading edge of the active battle area is determined from array GDBA and placed in variable GD.

At this point, subroutine SECWTH is called to compute the width of the sector segment at the western edge of the active battle area.

SECWTH returns with data describing the line of advance through the sector segment containing the active battle area. This data includes the latitude and longitude of the end points of the line of advance through the segment and the fractional distance (AL) through the segment to the western edge of the active battle area. Projection equations derived from spherical geometry are used to determine the latitude and longitude of the midpoint of the active battle area. Function GDIST is used to compute the distance between the targeted airbase, described by the coordinates (FLATAB, FLONAB), and the center of the active battle area given by the coordinates (FLATBA, FLONBA).

Finally, a count is kept in variable NPAAC of the number of active parking areas within the airbase. This value is determined by searching the first six elements of array IMAGE1 for nonzero values. Array IPAAC contains the index of the active parking area.

2.2.7.23 CHEM5. CHEM5 calls subroutine CHEMDAM to perform chemical damage assessments for all fire missions by each side. Before CHEMDAM is called for each fire mission, CHEM5 sets up the appropriate parameters.

2.2.7.23.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEM5:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL2, LOCAA2, CHA, CHB

Characteristic	Specification
Subroutines called	CHEMDAM, KDCODE
Called by	CHEM

2.2.7.23.2 Logic Functions. CHEM5 contains two sections. The first section calls CHEMDAM for each fire mission. The second section sums up the calculations after all the fire missions have been completed.

a. Section 1000 - Chemical Assessment for Each Fire Mission. This section consists of a Do loop for each side L which is attacking side N (N=3-1). NMF, the number of missions fired by side L, is set equal to NNIWAS(L). A Do loop is then executed over each fire mission I (I=1 to NMF).

For each fire mission I by side L, variables are set up for use in CHEMDAM which specify the chemical agent, dispersal parameters, target characteristics, and type of assessment. Function KDCODE is used to obtain IWC from the parameter INDEX. IWC specifies whether the assessment is of a division, sector, or theater weapon system. The detail print flag IPRT is also specified. Subroutine CHEMDAM is then called to do the appropriate damage assessment calculations for fire mission I by side L. After returning from CHEMDAM, the results of this fire mission are added to the total civilian casualties CIVCCH(KISS,N) and fatalities CIVFCH(KISS,N).

b. Section 2000 - Sum Up Results for This Cycle. After the damage assessment calculations have been completed for all fire missions by both sides, parameter KITC is set to 5 and CHEMDAM is called to sum up results for all of the fire missions on battlefield targets in this cycle and draw down the strength of all divisions that were attacked. Control returns to CHEM5 where a summary of civilian casualties is printed if IPRS=1. Control then returns to TMAIN.

2.2.7.24 CHEMDAM. Subroutine CHEMDAM performs the damage assessment for chemical weapons used against the following types of military targets: battlefield targets, airfields, supply nodes, and SSM sites. Military casualties, civilian casualties and equipment contamination are calculated. Damage assessment can be performed for volatile agents, liquids, or semivolatile agents spread with weapon delivery errors uniformly over an area, from a point source, or from a linear source.

The CHEMDAM coding contains simplifications of more complex models of agent dissemination. Reference 11 contains many of the equations used in CHEMDAM as well as references to those models.

2.2.7.24.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CHEMDAM:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL2, LOCAA2, CHA, CHB
Subroutines called	DROPS, LINFR, OFFCOV, SIMCN
Called by	CHEM5

2.2.7.24.2 Logic Functions. CHEMDAM performs the damage assessment for chemical weapons fired during fire mission I, by side L, against side N targets. The main thrust of the coding in CHEMDAM is the assessment of damage to battlefield targets in section 1000. Variable KITC, set in CHEM5, determines the target type assessed.

KITC	Target Type	Damage Assessment
1	battlefield (section 1000)	personnel, equipment, civilians
2	airfield (section 2000)	personnel, civilians
3	supply node (section 3000)	supplies, civilians
4	missile site (section 4000)	personnel, civilians
5	all (section 5000)	total up results and draw down divisions after all fire missions completed

The manner in which chemical damage is assessed in CHEMDAM depends on the values of input variables, on variables calculated in CHEM5 and other chemical subroutines, and on switch variables contained in the CHEMDAM coding. Some of these variables are described in the following paragraphs.

If index ICHFAT (an input variable) =1, both the number of persons incapacitated and the number of persons killed by the chemical agent is calculated. If ICHFAT=2, only the number incapacitated is calculated.

If index ICMD (an input variable) =1, calculations assume that the wind direction is known and that the agent is released so as to first impact at the upwind edge of the target. If ICMD=2, the agent is released over the target center. Local variable IMODL is obtained from IWLDM, which is set equal to IWLDSM(I,L) in CHEM5. IMODL determines the coverage model used in CHEMDAM.

IMODL	Coverage Model
1	Linear source (section 1240)
2	Uniform circular area coverage (section 1220)
3	Point source (section 1250)

The type of chemical agent IMTYPE(KA) is set equal to IWLKA which is specified in CHEM5. If IMTYPE(KA)=1, the chemical agent is a mixture of vapor and small aerosol particles. Simplified Calder-Sutton dosage and concentration equations (section 1600) are used to describe the behavior of the aerosol-vapor cloud as it travels downwind. When input variable ICMPLX=1, the regular Calder-Sutton algorithm is used (sections 1600 and 1800). If IMTYPE(KA)=3, the chemical agent is nonvolatile and falls as coarse aerosol and spray droplets. A simplified Porton model (section 1700) is used to determine the quantity of agent deposited downwind. When ICMPLX=1, the regular Porton algorithm is used (section 1700 and 1850. If IMTYPE(KA)=2, the chemical agent is a semivolatile agent that exists initially as liquid particles but evaporates quickly. The amount of semivolatile agent reaching the ground in liquid drops is calculated (section 1100) and then the Calder-Sutton algorithm is applied to the vapor fraction and the Porton algorithm is applied to the liquid fraction of the agent.

The following paragraphs describe the logic functions in each section of CHEMDAM.

a. Section 100 - Set Flag Variables. This is the beginning of CHEMDAM coding. The following variables are specified in this section.

Variable	Value	Description
L	LFIRER	Index of attacking side
N	3-L	Index of targeted side
IUN	I	Fire mission number
LRETN	1	Index for military personnel (1) or civilian (2) calculations
LLRETN	1	Index for calculating civilian casualties on airbase target
LARETN	1	Index for calculating off-base civilian casualties on airbase target
LBRETN	1	Index for calculating equipment contamination for supply node target
LCRETN	1	Index to initialize parameters for supply node target calculations
JFST	1	Index of first step in Gauss-Legendre integration over probability of casualty levels
JLST	5	Index of last step in Gauss-Legendre integration
NTPU	NTPP	Number of chemical protection levels
PDENSS	100	Factor used to calculate civilian casualties (sections 1550 and 2700)
ICOV	0	If ICOV=1, the loss of coverage because of dispersion and spreading is calculated using an input value of ASUBM, the coverage of each submissile. If ICOV=2, a value of ASUBM is estimated. If ICOV=0, the calculation of loss of coverage because of dispersion and spreading is suppressed.

After these variables are specified, control is directed to the section of CHEMDAM appropriate to the target type KITC.

b. Section 1000 - Calculate Parameters for Battlefield Targets. Indices are specified for the sector (IS), tactical role (ITR), division (ID), division type (IT), subunit (ISU), and zone (IZ).

Target dimensions in meters are calculated by different methods depending on whether or not the division is in combat or in reserve (IZ=0).

For a division in reserve the target is assumed to be square with side length

$$TLEN = \sqrt{\pi} * RDSUR(ISU,N)$$

where RDSUR(ISU,N) is the radius of a type ISU subunit in reserve posture.

The division dimensions are specified by inputs WIDDR(N) and DEPDR(N). AREAFC is the division area minus the target area. Zone dimensions are equated with division dimensions.

For a division in combat, target length, and width are defined by

$$TLEN = \frac{\sqrt{\pi}\ RDSUR(ISU,N)\ *\ FRDSUZ(IZ,ITR,N)}{\sqrt{SHAFAC(ISU,IZ,N)}}$$

and

$$TWID = TLEN * SHAFAC(ISU,IZ,N)$$

where FRDSUZ and SHAFAC are input variables.

Division dimensions are:

$$DWID = WTDCTR(IT,ICM,ITR)$$

and

$$DLEN = DWID * FDDPTH(N)$$

The zone has the same width as the division (DWID) and length

$$ZLEN = DLEN * PZDPTH(IZ,N)$$

After target dimensions have been calculated for the target and zone, AREAFC is set equal to the zone area minus the target area.

A sum over all subunits in the division is made to calculate TPZN, the total number of personnel in the zone; PZN(IP,IC), the total number of personnel in the zone in each physical protection category IP and chemical protection category IC; WZN(IW), the number of weapons of type IW in the zone; and TWZN, the total number of weapons in the zone. The numbers are obtained by multiplying input values of people or weapons by appropriate fractions. Similar quantities (TPSU, PSU(IP,IC), WSU(IW), and TWSU) are calculated for the subunit under attack. After the above calculations have been completed, section 1100 is begun.

c. Section 1100 - Initialize Damage Model Parameters. This section is called to begin chemical damage calculations for all target types (KITC=1 through 4). The following quantities are initialized: target aiming errors, coverage model type, agent constants and constants which define the initial coverage and spreading rate of the chemical cloud.

The weapon CEP is defined by

$$CEPP = \left[(TAESZD)^2 + (WLCEP)^2 \right]^{1/2}$$

where TAESZD(ISU,IZ,IDS) is the target acquisition sensor error and WLCEP is the CEP of the chemical delivery system.

A circular delivery error is assumed so the downwind and crosswind standard deviations WSIGD and WSIGC are set equal to CEPP/1.1774.

QALL, the quantity of chemical agent, is set equal to the product of WLQAT and INWFM (both defined in CHEM5) and then expressed in milligrams. QDEN, the milligrams of agent released per meter of crosswind target length, is set equal to QALL divided by WLSPR.

The coverage model to be used for damage assessment is specified by IMODL. The following coverage models are available in CHEMDAM: linear source (IMODL=1), uniform area coverage (IMODL=2), and point source (IMODL=3). The table below shows

the relationship between the coverage model and dissemination mode (IWLDM), height of burst (WLHOB), and the number of weapons delivered in a fire mission (INWFM). Variables IWLDM, WLHOB, INWFN determine IMODL and IUNF.

IMODL	IWLDM	WLHOB (meters)	INWFM
1	3		
2(IUNF=0)	2		
2(IUNF=1)	1	>250	
1	1	25-250	
1	1	<25	≥1
3	1	<25	1

If IUNF=0, the target area is covered randomly by submunitions. If IUNF=1, the target is uniformly covered.

The following parameters which affect the spreading rate of the agent are specified: UBAM, the windspeed converted to meters per second from input WINDSP(IACTWS) in km/hr; ALPHA and BETA, the expansion coefficients for the horizontal and vertical dispersion parameters, which equal ECALFA(IKASC, IACTWS) and ECBETA(IKASC,IACTWS), respectively; and dispersion parameters SIGYS (σ_Y), SIGZS (σ_Z), and SIGXS (σ_X). If non-zero dispersion parameters are not specified in CHEM5 ($\sigma_Y=\sigma_X$=WLSYS, σ_Z=WLSZS), then

$$SIGYS = 1.5\ (WLQAT)^{0.45}$$

and

$$SIGXS = SIGZS = SIGYS/3$$

The x-direction is downwind, the y-direction is crosswind, and the z-direction is vertical.

CHEMDAM contains coding for assessing the effects of three types of chemical agents. Vapor agents (IMTYPE(KA)=1) are assessed by the Calder-Sutton model. Liquid agents (IMTYPE(KA)=3) are assessed by the Porton model. The liquid fraction of semivolatile agents (IMTYPE(KA)=2) is assessed

by the Porton model and the vapor fraction is assessed by the Calder-Sutton model. Setting flags IEVV and/or IEVL to 1 signals that the agent contains vapor and/or liquid components. The remainder of section 1100 initializes parameters for vapor, liquid, and semivolatile agents.

If the agent is a vapor or a semivolatile liquid, the following parameters which are used later to calculate the crosswind and vertical standard deviations of the dosage field are defined

$$\text{SIGYI} = \left(\frac{\text{SYD100}}{100}\right)^{\text{ALPHA}}$$

$$\text{SIGZI} = \left(\frac{\text{SZDSRD}}{20}\right)^{\text{BETA}}$$

For liquid and semivolatile agents the following parameters are calculated for later use in calculating the integrated dosage: TFAC, SIGDU, BP, and ALBOT. These parameters are functions of initial particle diameter, windspeed, height of burst, the mass of agent released, and the relationship between particle diameter and the mass of agent released.

For a semivolatile agent, FRACL, the fraction of agent striking and the ground as liquid, is calculated. The calculation is made using Maxwell's evaporation equation with the ambient vapor concentration set equal to zero. A linear relationship between drop size and terminal velocity and a lognormal size distribution are also assumed. Subroutine DROPS is used to calculate DROPNU, the number of drops hitting the ground. If the rate of evaporation is high enough that the ambient vapor concentration cannot be neglected, an alternate method is used which assumes larger droplets fall fastest and therefore contribute vapor to the ambient condition for smaller droplets. A numerical integration over droplet size groups and altitude is performed to compute overall evaporation rates. If the calculated value of FRACL<0.05, the remaining calculations in CHEMDAM are done only for the vapor phase (IEVL=0). If FRACL>0.95, the vapor phase is ignored (IEVV=0).

If IEVV=0, the effective initial height of the vapor cloud (WLHOBV) and SIGZSQ (σ_z^2) are calculated for the vapor cloud by numerical integration in the vertical (z) direction.

If an unknown wind direction is assumed (ICMD=2), the CEP is increased for liquid agents and semivolatile agents with a liquid component. The downwind (WSIGD) and crosswind (WSIGC) standard deviations are

$$\text{WSIGD} = \text{WSIGC} = \sqrt{(\text{SIGCIR})^2 + (\text{DRIFT})^2}$$

where

$$\text{DRIFT} = \frac{227 * \text{wind velocity} * \text{height of burst}}{\text{initial median particle diameter}}$$

The degradation in coverage due to spacing errors and lack of precision is specified by COVS. Currently COVS is set to 1 (no degradation) if ICOV=0. The degradation COVS is computed if ICOV=1 or 2.

This initialization section normally ends by passing control to the next section of coding (section 1200). However, if supply node calculations are being made (LCRETN=2), control returns to section 3000.

d. Section 1200 - Begin Calculations of Casualties and Fatalities by Calculating Doses. This section begins the calculation of TSCU(IUN,N), TFSU, TCZN(IUN,N), and TFZN, the casualties and fatalities for the subunit and zone. The calculations are inside Do loops over each physical protection category IPP and each chemical protection category IC. These Do loops end in section 1260. Calculations are skipped if both the zone and subunit contain no personnel. The Do loops calculate the fraction of personnel incapacitated and killed in the subunit and in the zone: FSUC(IC,IPP), FSUL(IC,IPP), FZNC(IC,IPP), and FZNL(IC,IPP).

The median incapacitating dosage DCRITI of the agent is calculated from

DCRITI = PFAPOS(IPP,ITEMP)*DICT50(KA,IC)/FRACV

where

PFAPOS(IPP,ITEM) = the relative protection afforded by physical protection category IPP against a type ITEMP (=IMTYPE(KA)) agent

DICT50(KA,IC) = median incapacitating dosage for the inhalation component of type IMTYPE(KA) agent and chemical protection posture IC

FRACV = fraction of agent in vapor phase

In addition, because of the logarithmic nature of the fatalities versus dosage curve, the following local variables are calculated for use in fatality calculations in section 1230.

$$DCRLTI = \ln(DCRITI)$$

and

$$SLOI = RPSICL(KA)*\ln(10)$$

where

RPSICL(KA) = the reciprocal of the probit slope of the dosage response function for type IMTYPE(KA) agent (standard deviation of $\log_{10}$ dosage from $\log_{10}$ (median incapacitating dosage)). See reference 11.

Using other equations of the same form, variables DCRLTL and SLOL are calculated for a lethal dosage of the agent.

A Do loop is executed twice on KK. When KK=1, fatalities are calculated (if ICHFAT=1). When KK=2, casualties are calculated. This Do loop ends in section 1260. Section 1200 transfers control to section 1220, the next section of coding, if a uniform area coverage model (IMODL=2) is used. Otherwise control transfers to section 1230.

e. Section 1220 - Uniform Area Coverage Model. This section of coding is used only if IMODL=2. The uniform coverage model makes the following assumptions: the submunitions are uniformly dispersed over a circular area with

radius WRAD, the target is circular with radius RAD, and the area outside the weapon radius is unaffected. The weapon radius (WRAD) is set equal to one-third the height of burst (WLHOB). QD is the amount of agent in milligrams per square meter of weapon coverage. Dosage level DSUL is set equal to either DCRLTL (lethal) or DCRLTI (incapacitating) depending on whether KK=1 or 2. Subroutine OFFCOV is called to calculate COV, the fraction of the target area covered to level DSUL.

The remainder of this section calculates SUMINT(KK), the fraction of the target area receiving a critical dose of agent, and AOVI (or AOVL), the area covered by the weapon but outside the target which receives an incapacitating (or lethal) dosage.

For these calculations,

$$SUMINT = COV*PROB$$

and

$$AOVI = [\text{weapon area} - (\text{target area})*COV] * PROB$$

where

PROB = probability of becoming a casualty at the actual dose level covering the target

If IUNF=1 (meaning the height of burst is above 250 meters) or ICOV=0 (implying the coverage calculation is suppressed), then PROB is calculated by calling subroutine SIMCN to calculate the value of the probit integral. (See reference 11 for details.)

If IUNF=0 (height of burst below 250 meters) and ICOV=1, PROB is calculated in the following way. Each of the XNWFM submunitions covers an area ASUBM, XNWFM is assumed so large that the probability that a point is covered can be approximated by a normal distribution. PROB is calculated using a five-point Gauss-Hermite integration of VAL over dose levels. VAL, the fraction killed or incapacitated at each dosage level, is the value of the cumulative normal distribution calculated using subroutine SIMCN. Thus

$$PROB = (1/\sqrt{2\pi}) \int_{-\infty}^{\infty} e^{-t^2} VAL(t)dt$$

$$(1/\sqrt{2\pi}) \sum_{IH=1}^{5} WTHERM(IH)*VAL(XHERM(IH))$$

The abscissas XHERM(IH) and weights WTHERM(IH) are defined by Data statements. After calculating SUMINT(KK), control passes to section 1260, the end of the Do loop on KK.

f. Section 1230 - Begin Gauss-Legendre Integration. Control passes to this section when target coverage is not uniform (IMODL≠2). The fraction of the target area receiving a critical (lethal or incapacitating) dose of agent, (SUMINT(KK)), is determined from

$$SUMINT(KK) = \int_{0}^{1} coverage(P)dP$$

where coverage(P) is the fraction of the target receiving a dosage that results in a probability P of kill (KK=1) or incapacitating injury (KK=2).

To calculate SUMINT(KK), the above integral is replaced by a five-point Gauss-Legendre sum

$$SUMINT(KK) = 1/2 \sum_{J=1}^{5} COMCOV(J,KK)*PROWT(J)$$

COMCOV(J,KK) is the fraction of the target area covered to cumulate dosage level DSUL, which produces fatalities (or casualties) to the PROLVL(J) level. The PROWT(J) are the Gauss-Legendre weights, defined by Data statements. The factor 1/2 in the above equation comes from replacing the normal Gauss-Legendre integration interval of -1 to 1 by the interval 0 to 1.

This section begins the Do loop on summation index J. For fatality calculations (KK=1) the dosage DSUL is calculated from

$$DSUL = DCRLTL + PROABS(J)*SLOL$$

An equation of the same form is used for casualties. PROABS(J) is the value of the abcissa of the normal distribution curve corresponding to a cumulative normal probability of PROLVL(J). To calculate COMCOV(J,KK) control continues into section 1240 for a linear source coverage model (IMODL=2) or is transferred to section 1250 for a point source coverage model (IMODL=3). The Do loop on summation index J ends in section 1260.

g. Section 1240 - Linear Source Model. Control is transferred from section 1230 to this section to calculate COMCOV(J,KK) when IMODL=2. Section 1240 is executed within the Do loop performing the Gauss-Legendre integration of the casualty probability over dosage level DSUL.

The fraction of the target area covered to cumulative dosage level DSUL is

$$COMCOV(J,KK) = COVY*COVX*COVS$$

COVY, the fraction of the crosswind dimension of the target covered, is calculated by calling subroutine LINFR.

COVS=1 (no degradation). To calculate COVX, the fraction of the downwind dimension of the target covered, it is necessary to calculate XX and XXS, the distance downwnd on either side of the center of the distribution at which the cumulative dose is DSUL. XXS is closest to the forward edge of the target.

If IEVV≠0, the agent contains a vapor phase. KRETN is set to one and control is passed to section 1600 to calculate XX and XXS via the Calder-Sutton model. (If the agent does not have a vapor phase the Porton model is used, as will be discussed later.) After returning from section 1600, the loss of coverage due to round-to-round dispersion and spreading is calculated if ICOV=1 and the height of burst (WLHOB)<5 meters. This calculation is made in two steps. First control is transferred to section 1900 (KSZRET=4) to calculate the downwind half-length AA and crosswind half-length BB of the ellipse covered to dosage level DSUL by a single

round. Control then passes to section 1950 (KSYRET=1) to adjust XXS to reflect the loss of coverage due to round-to-round dispersion and spreading.

After these dispersion calculations have been made (if required), flag IEVL is checked to see if the agent has a liquid phase. If the agent is only vapor (IEVL≠1) control passes to the end of section 1240 for the calculation of COVX. If the agent is semivolatile (both liquid and vapor phases present) XX and XXS for the vapor phase are set equal to XXV and XXSV respectively, and control is passed to section 1900 (KSZRET=2) to calculate the dimensions AALV(=AA) and BBLV(=BB) of the vapor coverage. Then DSUL is calculated for the liquid phase and control is again passed to section 1900 (KSZRET=3) to calculate the dimensions AALL(=AA) and BBLL(=BB) of the liquid coverage. Porton model calculations are then made.

Control is passed to section 1700 (KRETN=1) to calculate XX and XXS via the Porton model for liquid agents and the liquid fraction of semivolatile agents. XX and XXS are the distances in the downwind direction at which the cumulative dose is DSUL. After returning to section 1240 the loss of coverage due to round-to-round dispersion and spreading of the liquid agent is calculated if ICOV=1 and the height of burst (WLHOB)<25 meters. This calculation is made in two steps. First control is transferred to section 1900 (KSZRET=5) to calculate the downwind half-length AA and crosswind half-length BB of the ellipse covered to dosage level DSUL by a single round. Control passes to section 1950 (KSYRET=2) to adjust XXS to reflect the loss of coverage due to round-to-round dispersion and spreading.

After the Calder-Sutton and/or Porton models have calculated XX and XXS, the coverage COVX in the downwind direction can be calculated. If only one phase (either liquid or vapor) is present, DEL, the fraction of the target length covered, is calculated from XX and XXS. Subroutine LINFR is called to use DEL to calculate COVX.

If both phases are present, a check is made to see if the areas covered by the liquid and vapor overlap. When the coverages overlap LINFR calculates COVX using a value of DEL calculated from the outside dimensions of the overlapping coverage. If the liquid and vapor coverages do not overlap, COVX equals the sum of liquid coverage and the vapor coverage, both calculated by subroutine LINFR.

COMCOV(J,KK), the fraction of the target area covered to cumulative dosage level DSUL, is obtained by multiplying COVX, COVY, COVS. Control then passes to section 1260 where the Do loop on J ends.

h. Section 1250 - Point Source Model. Control is transferred to this section from section 1230 to calculate COMCOV(J,KK) when IMODL equals 3. This section is executed within the Do loop performing the Gauss-Legendre integration of the casualty probability over dosage level DSUL.

Flag KSZRET is set equal to 1. Control is passed to section 1900 to calculate the downwind half-length AA and crosswind half-length BB of the ellipse covered to dosage level DSUL. After this calculation is performed, control is returned to section 1250.

Since the finite source size causes an apparent displacement of the coverage ellipse, XOFF, the ellipse offset distance is calculated.

The fraction of target area covered to cumulative dosage level DSUL, COMCOV(J,KK), is equal to the product of COVX and COVY. COVX, the fraction of target length covered (downwind coverage) is calculated from

$$COVX = \frac{SUM}{TAREA}$$

SUM, the expected coverage, is the integral of coverage as a function of downwind offset (aiming error and XOFF) integrated over the probability of all downwind offsets. This integration is done by the five-step Gauss-Hermite method.

The fraction of target width covered, COVY, (crosswind coverage) is calculated by assuming the intended delivery is at the center of the target. Subroutine LINFR is called to calculate COVY.

When J (the index for the Gauss-Lengendre integration)=3, the cloud area π*AA*BB is retained as AOVLV for bonus casualty calculations. After this calculation, control then passes to section 1260 where the Do loop on J ends.

i. Section 1260 - Finish Calculation of Casualties and Fatalities. The first line in this section is the end of the Do loop on J, the index specifying dosage level. After

this Do loop has been completed, SUMINT(KK) is calculated by the five-point Gauss-Legendre sum

$$\text{SUMINT(KK)} = 1/2 \sum_{J=1}^{5} \text{COMCOV(J,KK)*PROWT(J)}$$

SUMINT(KK) is the fraction of the target area receiving a lethal (KK=1) or incapacitating (KK=2) dose of chemical agent. The calculation of SUMINT(KK) ends the Do loop on lethality index KK. The remainder of section 1260 calculates subunit and zone casualties and fatalities. The casualty calculations are described below. Fatality calculations are similar except for variable names.

For chemical protection category IC and physical protection category IPP, the fraction of casualties in the subunit, FSUC(IC,IPP), equals SUMINT(2). The total casualties on side N in the subunit for fire mission IUN, TCSU(IUN,N), is increased by FSUC(IC,IPP)*PSU(IPP,IC), where PSU(IPP,IC) is the number of personnel in the subunit in physical protection category IPP chemical protection category and IC. If the agent contains a vapor component and chemical protection category IC is 2 or 3 (mask available), XM is calculated. XM is the distance that the center of the vapor cloud drifts downwind before personnel have received warning and put on their masks. AOVI, the total area covered with an incapacitating dose, is the larger of: (1) XM times the spray width (WLSPR), and (2) the percutaneous area AOVIV calculated previously by the Calder-Sutton model. If the agent has a liquid component, AOVI is modified to include any additional area covered by the vapor cloud. Finally, FZNC(IC,IPP) the fraction of casualties in the zone outside the target area is given by

$$\text{FZNC(IC,IPP)} = \frac{\text{AOVI-ACOVT}}{\text{AREAFC}}$$

where ACOVT is the area covered within the target area and AREAFC is the zone area minus the target area.

The total casualties on side N in the zone for fire mission IUN (TCZN(IUN,N)) is increased by FZNC(IC,IPP)*PZN(IPP,IC). For targets other than battlefield targets control passes to other sections of CHEMDAM depending on the values of flags LLRETN and KITC. For battlefield targets, arrays

IDSVE(IUN,N), ISUSVE(IUN,N), IZNSVE(IUN,N), and IWNSVE(IUN,N) are used to store, respectively, ID, ISU, IZ, and IWTAR, the number of subunits. These arrays will be used in section 5000 to accumulate division results. For battlefield targets, control then passes to the next section of coding, section 1400, to calculate equipment contamination.

j. Section 1400 - Calculate Equipment Contamination. This section calculates equipment contamination when battlefield or supply node targets are attacked by an agent with a liquid phase (IMTYPE(KA)=2 or 3). COVX, the fraction of downwind target coverage, is calculated by subroutine LINFR after the Porton model has been used (with J=3, KRETN=2) to calculate XX, the downwind extent of agent deposition to a cumulative level DEPAWC(KA) to assure equipment contamination.

COVY, the fraction of crosswind target coverage, is calculated using subroutine LINFR after YY, the cloud width, has been calculated for source model IMODL. COVS equals 1 (no degradation). The fraction of the target covered, COV, is set equal to the product of COVX, COVY, and COVS. ACOVT, the amount of target area containing contaminated equipment, is set equal to COV*TAREA. The fraction of the zone contaminated outside the target area, FRZNE, is also calculated.

Control is returned to section 3000 if LBRETN=2 (supply node target). For a battlefield target, for each weapon type IW in division ID the following are calculated:

CONEQ(ID,IW) = the total number of weapons contaminated in division ID

WSUCN(IW) = the number of weapons contaminated in the subunit

WZNCN(IW) = the number of weapons contaminated in the zone.

If the detail print flag IPRT=1, the results for this fire mission are printed out. Finally, TCZN(IUN,N), the casualties in the zone, is converted from a number of casualties to a fraction by the following exponential drawdown

$$TCZN(IUN,N) = 1 - \left(\frac{1-TCZN(IUN,N)}{TPZN} \right)^{IWLTAR}$$

where TPZN is the total number of personnel in the zone and ILWTAR is the number of subunits.

Control is passed to the next section (1500) to calculate civilian casualties.

k. Section 1500 - Calculate Civilian Casualties. This section concludes chemical damage calculations for all target types. The calculation of civilian casualties assumes that civilians have no chemical protection and that civilians are randomly distributed with density PKENSS among battlefield units. Civilian casualties and fatalities are calculated within a Do loop on physical protection posture IPP. For each physical protection posture the cumulative critical dosage is calculated. The appropriate source and agent models are used to find the area covered with this critical dosage. Multiplying the area covered by the population density gives the numbers of civilians casualties and fatalities. Control transfers to section 2000 if the above civilian casualties were made for an airbase attack (LARETN=2). Otherwise, the results of these calculations are printed out (if IPRT=1) and control returns to CHEM5.

l. Section 1600 - Simplified Calder-Sutton Model. This section is called to calculate the downwind dissemination of vapor agents from a linear source. The following assumptions are made: the decay in agent toxicity during dissemination can be neglected; the dosage is zero until the center of the cloud passes a downwind point, then it assumes its full value; dosage is calculated at ground level (sampling height=0); and no crosswind variation in dosage.

The equations resulting from these assumptions are contained in reference 11. This section of coding solves these equations for XX and XXS, the downwind distances between which the crosswind integrated dose is greater than or equal to DSU, where DSU = exp(DSUL).

This is accomplished by solving the dosage equation for the dissemination parameters $\sigma_z(XX)$ and $\sigma_z(XXS)$ using either polynomial fits or an iterative technique. The equation for $\sigma_z(X)$ is then solved to obtain values of XX and XXS. If flag ICMPLX=1, control is transferred to section 1800 where an iterative technique is used to calculate more accurate values of XX and XXS starting with the above values of XX and XXS.

When J=3 (J specifies dosage level DSUL), additional calculations determine the area covered (e.g., AOVL) for use in the calculation of civilian casualties (section 1500) and equipment contamination (section 1400). Finally, when personnel are initially unmasked (IC=2 or 3), XX is increased to the cloud travel distance if the distance the cloud travels before masking is greater than the percutaneous dose level distance.

m. Section 1700 - Simplified Porton Model. This section is called to calculate the downwind dissemination of liquid agents from a linear source. The following assumptions are made in this simplified Porton model: log-normal distribution of droplet mass; contamination is due to the gravitational settling of these droplets from a cloud traveling downwind from an elevated line source; equations are simplified and linearized as explained in reference 11.

This section of coding solves those equations for XX and XXS, the downwind distances between which the log of the crosswind integrated dose is greater than or equal to DSUL. In the coding, variables PHIL and ALNT represent ln ϕ and ln τ, respectively, in the referenced equations. If flag ICMPLX=1, control is transferred to section 1850 to iteratively calculate more accurate values of XX and XXS using the value calculated in this section as a starting solution.

When J=3 (J specifies dosage level DSUL in sections 1230 and 1500), additional calculations determine the area covered (e.g., AOVL) for use in the calculation of equipment contamination (section 1400) and civilian casualties (section 1500). Control then returns to the calling section.

n. Section 1800 - Complete Calder-Sutton Model. This section contains an iterative algorithm for finding more accurate values of XX and XXS, the downwind ranges at dosage DSU. This section is used only if flag ICMPLX=1. First, a simplified Calder-Sutton model (section 1600) is used to compute initial values of XX and XXS. Then CWID, the crosswind integrated dose, is calculated for XX using equations described in reference 11. If CWID and DSU differ by more 0.001*DSU, XX is adjusted, and CWID is recomputed. The process is repeated until CWID equals DSU to within .001 DSU. For the first 20 iterations, Newton's method is used to adjust XX using the difference between CWID and DSU and the derivative of dose as a function of distance evaluated from the complete equations at the initial value of XX. If in 20

iterations no solution is reached, further iterations are made adjusting XX by halving intervals. If in 200 iterations no solution is reached, the initial value of XX from the simplified equations is used and a warning message is printed. After XX has been calculated, XXS is calculated in the same way. Control returns to the simplified Calder-Sutton model (section 1600) where XX and XXS are used in additional calculations.

o. Section 1850 - Complete Porton Model. This section contains an iterative algorithm for finding more accurate values of XX and XXS, the downwind ranges corresponding to dosage DSU. This section is called from section 1700 (simplified Porton model) only if flag ICMPLX=1. Section 1700 calculates initial values of XX and XXS using the simplified Porton equations. Control then transfers to the present section. SLOPEA, the derivative of the crosswind integrated dose with respect to downwind distance is calculated. Control then transfers to section 1800 where precise values of XX and XXS are calculated using Newton's method. If Newton's method fails to converge after 20 iterations, further iterations are made adjusting XX (or XXS) by halving intervals. After each adjustment of XX (or XXS) control passes from section 1800 to section 1850 to calculate CWID corresponding to the adjusted XX (or XXS). Section 1850 then passes control back to section 1800 to test for convergence.

After precise values of XX and XXS have been obtained, control returns from section 1800 to the simplified Porton model (section 1700) where XX and XXS are used for additional calculations.

p. Section 1900 - Calculate Dimensions of Elliptical Agent Cloud. This section calculates the downwind half-length AA, the crosswind half-length BB, and XMID, the downwind location of the elliptical area covered to dosage level DSU (DSU=exp(DSUL)). This section is called when a point source model is used, when the coverage of individual rounds is needed to calculate round-to-round dispersion, or when the overlap between liquid and vapor clouds is needed.

The first part of this section uses simplified Calder-Sutton equations for a vapor cloud; if the agent is liquid, the simplified Porton equations in the last part of the section are used instead. In each case, the simplified model equations are first used to calculate AA, then the downwind midpoint of the cloud (XMID) is calculated. At this midpoint

the crosswind distance BB to dosage level DSU is calculated. Control is then transferred back to the appropriate section via flag KSZRET.

q. Section 1950 - Calculate Degradation Due to Round-to-Round Dispersion. This section is called from section 1240 (linear source model calculations) when ICOV=1 and height of burst<5 meters for vapor agents or <25 meters for liquid agents. The appropriate Calder-Sutton or Porton equations in reference 11 are used to calculate the ratio of crosswind spreading to downwind spreading. Subroutine SIMCN is used to calculate cumulative downwind and crosswind coverage probabilities. These are used to calculate the open area caused by dispersion and spreading. Variable XXS, the distance to the forward edge of the downwind coverage, is increased to reflect the decrease in area covered because of dispersion and spreading. Control is returned to section 1240 via flag KSYRET.

r. Section 2000 - Assessment for Airfield Targets. This section of CHEMDAM is used if flag KITC is set to 2 in CHEM5. Casualties and fatalities are calculated for military personnel (IL=2), on-base civilians and off-base civilians when a side N airbase is attacked.

The first part of this section calculates the military personnel ACTPAJ(I) on each of the parking areas (I=1-6). The logic for assigning personnel among parking areas is given in reference 11. The fraction of personnel in the targeted parking area, PSU(IPP,IC) in physical protection category IPP and chemical protection category IC is obtained by multiplying input fractions FPAFRC(IPP,N) and FRABCP(IC,N). The fraction of personnel in the parking areas not targeted (PZN(IPP,IC)) equals PSU(IPP,IC).

If parking area IPA is under attack, the target dimensions are those of that parking area (AFDIM, defined in data statements at the beginning of CHEMDAM, times the conversion factor from feet to meters). If IPA=0, the chemical attack is on the runway and target dimensions are set equal to 0 (no personnel on the runway).

To assess chemical damage outside the targeted item (parking area or runway), weighting factors are calculated for parking areas not under attack. The calculation of parking area weighting factors involves the estimated crosswind distance over which the spray is dispersed and the distance from the

targeted area to the parking area. Combining these weighting factors gives ZLEN of the targeted zone, and AREAFC, the area in the zone outside the target area. BONEP is set equal to the sum of the personnel in untargeted parking areas. Civilian casualty weighting factors THCVU, THCVL, and zone dimensions ZCVU, ZCVL are calculated for the upper and lower halves of the airfield, respectively. Control passes to section 1100 for chemical attack assessment. After calculation of the fraction of personnel casualties and fatalities (sections 1100-1260), control returns to this section. Subunit results TCSU and TFSU are multiplied by ACTPAJ(IPA) to give the numbers of personnel incapacitated and killed, respectively, in the target area. Zone results TCZN and TFZN are multiplied by BONEP to give the numbers of personnel incapacitated and killed, respectively, in the zone outside the target area. Detailed results are printed if IPRT equals 1.

On-base civilian casualties and fatalities are calculated for the two halves of the airbase: upper half (LLRETN=2) and lower half (LLRETN=3). For each half, the calculations are made in sections 1100-1260. The results of those sections are multiplied by the appropriate civilian casualty weighting factors. Off-base civilian casualties and fatalities are calculated in section 1500 (LARETN=2). The summary results are printed (if IPRT=2) and control returns to CHEM5.

s. Section 3000 - Assessment of Supply Node Targets. This section of CHEMDAM is used if flag KITC is set to 3 in CHEM5. The target area is assumed to be square with area

$$TAREA = SQMPTS(N)*SUPIN(ISN)$$

where SQMPTS(N) is the area in square meters occupied per ton of supplies and SUPIN(ISN) is the supply inventory at node ISN. If the above equation gives TAREA=0.0, the program sets TAREA=1,000,000 square meters.

The zone length ZLEN is set to an arbitrary nominal value of 10,000 meters and AREAFC, the zone area outside the target area, is set to 1,000,000 square meters. Control is then transferred to section 1100 (LCRETN=2) to initialize agent parameters. Supplies are not contaminated unless the agent has a liquid phase. For the liquid phase, section 1400 is called (LBRETN=2) to calculate COV, the fraction of the target covered. The tons of supplies contaminated, FSUPCT(ISN), is increased by SUPIN(ISN)*COV. The tons of supplies at the

node; SUPIN, is replaced by SUPIN*(1.0-COV). Control then passes to section 1500 to calculate civilian casualties (IL=2) and returns to CHEM5.

t. Section 4000 - Assessment of Surface-to-Surface Missile Site Targets. This section of CHEMDAM calculates the personnel (IL=2) and civilian casualties on a type ISM missile site for side N. Index ISM is specified by IWLCT and IDS obtained from CHEM5 (ISM=1 for medium-range missile sites; ISM=2 for long-missile site).

The number of personnel in the target area PSU(IPP,IC) in each physical and chemical protection category is calculated by multiplying PWSSMS(ISM,N), the number of people, by the fraction in each category. The number of personnel in zone PZN(IPP,IC) is set equal to the number in the target area. The target area (TAREA) is obtained from the site radius RDSSMS(ISM,N). TLEN and TWID, the target dimensions, are calculated by assuming the target is square. The zone length ZLEN is set to an arbitrary 10,000 meters. Control is transferred to section 1100 to calculate personnel casualties and fatalities. After returning from section 1260, results are printed if IPRT=1. Control then passes to section 1500 to calculate civilian casualties and returns to CHEM5.

u. Section 5000 - Accumulate Chemical Damage Results for Divisions. This section of CHEMDAM is used if flag KITC is set to 5 in CHEM5. This section consists of a Do loop over all fire missions by both sides.

For each fire mission, the subunit results are combined into zone results. Subunits with casualties greater than the fraction FPSWDU(ISU,N) of TOE personnel strength are withdrawn from the division. Results from all of the zones are summed up to obtain PDIV(ID), the resulting personnel strength in the division. Results are printed for each fire mission, if IPRT=1. Control then returns to CHEM5.

2.2.7.25 DROPS. Subroutine DROPS calculates the number of drops of liquid per cubic centimeter with radius larger than RL. DROPS assumes that the liquid drop size is a log-normal distribution with mean drop size XMU and standard deviation SIG.

2.2.7.25.1 Programming Specifications. The following table summarizes the principal specifications of subroutine DROP:

Characteristic	Specification
Formal parameters	RL = Input minimum radius in microns, of the number of drops calculated
	XMU = Mean drop size in microns for all drops
	SIG = Standard deviation of log-normal distribution of drop sizes
	VOL = Volume of interest in cm^3
	DROPNU = The value returned by subroutine DROPS for the number of drops per cm^3
Common blocks	None
Subroutines called	None
Called by	CHEMDAM

2.2.7.25.2 Logic Functions. DROPS calculates DROPNU, the number of drops of liquid per cm^3 with radius larger than RL microns. DROPNU is calculated from the following expression

$$DROPNU = \frac{3x10^{12}}{4\pi\sqrt{2\pi}\ (SIG)} \int_{RL}^{\infty} \exp\left[\frac{-(\ln(X) - \ln(XMU))^2}{2(SIG)^2}\right] \frac{dX}{X^4}$$

where SIG and XMU are input values.

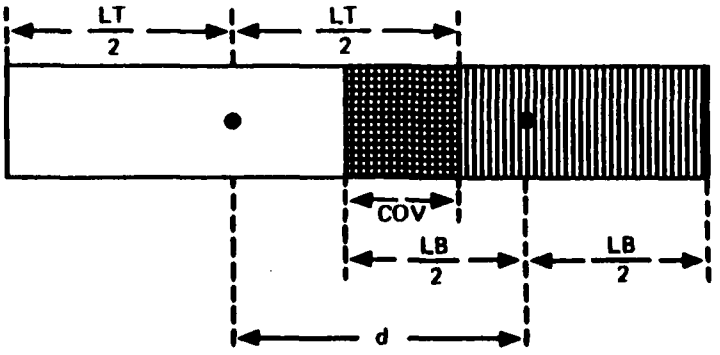
In order to numerically evaluate the above integral, the bounds of integration are replaced by XMAX and XMIN. For relatively large mean drop sizes, XMIN=RL and XMAX=4RL. Otherwise, an empirical formula is used to calculate XMIN and XMAX. The integral is evaluated using a Simpson's Rule algorithm with 20 integration steps.

2.2.7.26 LINFR. Subroutine LINFR is called by subroutine CHEMDAM to calculate the expected crosswind coverage of a target line of length LT by a spray line of length LB. The intended center of the spray line is displaced a distance D from the center of the target line with Gaussian aiming errors.

2.2.7.26.1 Programming Specifications. The following table summarizes the principal specifications of subroutine LINFR:

Characteristic	Specification
Formal parameters	RLEN = LB/LT, ratio of spray length to target length
	DEL = D/LT, ratio of displacement of spray line and target to target length
	FLAM = σ/LT, ratio of standard deviation of aiming error to target length
	ANS = Expected fraction of target line covered by spray as calculated by subroutine LINFR
Common blocks	None
Subroutines called	SIMCN
Called by	CHEMDAM

2.2.7.26.2 Logic Functions. LINFR calculates ANS, the expected fraction of a target line of length LT, covered by a displaced spray line of length LB, as shown in the following figure.



where

d = displacement of spray line

and

COV = coverage = ANS*LT

The displacement d shown in the above figure is the sum of D, the displacement of the aim point, and the random aiming error. LINFR contains two sections. Section 10 calculates ANS when the random aiming errors can be ignored. Section 20 includes those errors.

a. Section 10 - Ignore Random Errors. This section is used to calculate ANS when $\sigma \leq (.001)*LT$. By looking at the figure, one sees that if $LB \leq LT$ then

$$ANS = \frac{1}{LT}\left(\frac{LT + LB}{2} - D\right) \quad \text{for} \quad \frac{LT-LB}{2} \leq D \leq \frac{LT}{2} + \frac{LB}{2}$$

$$= \frac{LB}{LT} \quad \text{for} \quad D < \frac{LT-LB}{2}$$

$$= 0 \quad \text{for} \quad D > \frac{LT+LB}{2}$$

similarly for LB>LT

$$ANS = \frac{1}{LT}\left(\frac{LT+LB}{2} - D\right) \quad \text{for} \quad \frac{LB-LT}{2} \leq D \leq \frac{LT}{2} + \frac{LB}{2}$$

$$= 1 \quad \text{for} \quad D < \frac{LB-LT}{2}$$

$$= 0 \quad \text{for} \quad D > \frac{LT+LB}{2}$$

b. Section 20 - Include Random Errors. When $\sigma > (.001)LT$, then d in the figure is the sum of D and X, the random error. The expected coverage is obtained by summing the coverage for all values of X weighted by P(X), the probability of such an error. The resulting equations are described in reference 2. Subroutine SIMCN is used in those equations to calculate ANS.

2.2.8 LINKG. This subsection describes the routines contained in LINKG. These routines comprise the target acquisition model.

2.2.8.1 TARACQ. Subroutine TARACQ is an executive routine which controls the execution of the target acquisition portion of TACWAR.

2.2.8.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TARACQ:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL 3, TAE, TACQ
Subroutines called	TARACA, TADPAR
Called by	TMAIN

2.2.8.1.2 Logic Functions. Subroutine TARACQ begins by setting the sector index to the value passed from TMAIN. Subroutines TARACA and TADPAR are called to perform the target acquisition simulation within the specified sector. Control is then returned to TMAIN.

2.2.8.2. TARACA. Subroutine TARACA simulates the acquisition of targets in the active battle area of the specified sector by ground, army-air, and air force sensors. Sensors operate in one of three modes: standoff (fixed or vertical), standoff (moving) and penetrating (forward area search). Both continuously-operating and glimpse sensors are modeled. For each type of subunit in each division zone, the probability that a subunit of that type is detected, and the corresponding sensor error and delay time are calculated in TARACA.

2.2.8.2.1. Programming Specifications. The following table summarizes the principal specifications of TARACA:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL3, TAE, TACQ

Characteristic	Specification
Subroutines called	TARACE
Called by	TARACQ

2.2.8.2.2 Logic Functions. The logic of TARACA is executed for each side. For each subunit type in each division zone, values are computed for three arrays: PSZDDS, the probability that a subunit is detected; TAESZD, the target acquisition sensor error; and TADSZD, the target acquisition delay time. The code is divided by comment cards that label the functions performed by the logic.

a. Section 100 - Initialize Working Variables. In this section of TARACA, the index to the use of the target acquisition model (IUTAM) is checked. If the model is to be used, the value of each element of PSZDDS, TAESZD and TADSZD is initialized to zero. Otherwise, user-input values are assigned to these arrays. In either case, the remainder of this section is executed for both sides. Control is returned to TARACQ at this point if either side has no divisions in the sector. Otherwise, the values of the indices which represent the posture (IP), the active battle area (IBA), the weather zone (IWZ), the terrain (KT) and the current combat cycle (ITT) are determined. The length of a target acquisition cycle (TACL) is calculated.

It is assumed that divisions are located side by side, along as much of the sector front as possible, with no gaps between divisions. The total width (TW) of the divisions in the sector is determined for each side, based on combat deployment and tactical role. TW is compared to the sector width (WIDS) and the divisions in the sector are expanded or contracted so that TW=WIDS and/or (WIDS-TW) is minimized. The width (DVWDTH) and depth (DVDPTH) of each division and the width of the unoccupied area on each side of the sector (GAP=(WIDS-TW)/2) are computed. Control is returned to TARACQ at this point if the target acquisition model is not to be used. Otherwise, the remaining sections of code are executed consecutively for each side.

b. Section 200 - Target Acquisition by Ground Sensors. This section of TARACA begins by setting the probability that the weather ceiling is above the sensor altitude (PC) equal

to 1. The sensor velocity is set equal to 0. The ratio of sensing divisions to target divisions (FAC) is computed for use in the allocation algorithm. Ground sensors are allocated to target divisions according to the following assumptions:

(1) Exactly FAC of the sensing divisions (and their sensors) are assigned to each target division;

(2) Allocation of sensors to target divisions is taken in order from one edge of the sector;

(3) A division's sensors are positioned at the midpoint of the division or at the midpoint of the division fraction with which they are associated;

(4) Sensors acquire targets throughout the target division to which they are assigned.

(An example of this allocation scheme is presented in reference 2, pp. 115-116).

Prior to any acquisition assessments in TARACA, six variables are set up to implement the allocation algorithm:

- PFAC - A check which indicates whether to change the target division, the sensing division, or both for the next assessment.
- TFAC - The fraction of a sensing division considered in the current assessment.
- GFAC - The cumulative fraction of the current sensing division that has been assessed (including the current assessment).
- HFAC - The fraction of the next sensing division required to complete the assessment of the current target division.
- $ICTSD_{ing}$ - An index to the location of the current sensing division.
- ICTTD - An index to the location of the current target division.

The target acquisition assessment logic is performed once for each sensing division or division fraction; therefore, each assessment involves one target division and all or part of one sensing division. First, the location of the edge of the target division (DTX), the location of the sensor group (TSX) and the horizontal distance from the sensors to the target edge (DSX) are computed. The remainder of the assessment procedure is performed separately for each type of ground sensor against subunits of each type in each target zone. (The target acquisition equations used in TARACA are described in detail in reference 2, pp. 121-130).

The number of sensors (TEMP1) in the division or division fraction being assessed is calculated based on the division's current effectivenss in its tactical role of defense or attack (EFFDD or EFFDA), the TOE number of sensors per notional division (TGSND), the fraction of a notional division's sensors in the type division (FGSTD) and TFAC. The distance from the sensor group to targets in a particular zone (DY) is calculated, based on posture, tactical role and target division type, according to the assumption that targets are located along a line midway back in the zone. Subroutine TARACE is called to compute the expected range from sensor to target (RANGE) and the average detection potential (DINTEG).

The probability of a line of sight (PLOS) is calculated based on RANGE and terrain; the probability that the visibility exceeds the range to the target (PV) is calculated based on weather data and on the effect of visibility on the sensor. Then, the probability that a particular sensor acquires a target (PASDT) is calculated using equation 6 (reference 2, p. 123) for continuously-operating sensors or equations 19-21 (reference 2, p. 130) for glimpse sensors. Next, the probability that no sensors acquire a target (TEMP2) and the probability that at least one sensor acquires a target (TEMP3) are computed. These two probabilities are used to update the appropriate elements of two arrays: PDSZDS, which is used in the calculation of overall sensor error and delay time (see reference 11); and PGSDT, which is used in equation 16 (reference 2, p. 128) to calculate the overall probability that a subunit is detected. The appropriate elements of TAESZD and TADSZD (which at this point contain intermediate calculations for sensor error and delay time) are also updated using TEMP3 as a weighting factor.

When the target acquisition assessment logic has been completed for all sensor types in a particular sensing division, the value of PFAC is checked and the values of PFAC, TFAC, GFAC, HFAC, and ICTSD and/or ICTTD are updated appropriately. Then, the target acquisition assessment logic is performed for a new sensing and/or target division. This process continues until all sensing divisions or division fractions have been assessed.

c. Section 300 - Target Acquisition by Army-Air Sensors. In this section of TARACA, operational army-air carriers are apportioned by mode of operation (standoff or penetrating) into single-sensor groups which are allocated equally among all target divisions in the sector. The probability that the ceiling is above the sensor altitude (PC) is calculated based on the minimum operational altitude (AMOAAC) of the carrier. If necessary, the sensor altitude (ALT) is lowered to the weather ceiling. The distance from the sensor to the target edge (DSX) is calculated based on flight path location (FDWLAC). The distance from the sensor to the targets in a particular zone (DY) is calculated based on carrier type, mode of operation and target division type. The target acquisition assessment logic is similar to that of section 200.

d. Section 400 - Target Acquisition by Air Force Sensors. In this section of TARACA, reconnaissance aircraft are apportioned by mode of operation (standoff or penetrating) into single-sensor groups which are allocated equally among all target divisions in the sector. The logic of this section is similar to that of section 300.

e. Section 500 - Calculate Overall Detection Probabilities, Sensor Errors and Delay Times. The calculations in this section of TARACA are performed for each subunit type in each zone of each target division. Based on calculations in the previous sections, the probability that a subunit is detected (PSZDDS) is computed, and the computations of the corresponding sensor error (TAESZD) and delay time (TADSZD) are completed.

2.2.8.3 TARACE. Subroutine TARACE is called by TARACA to calculate the expected range from sensor to target and the average detection potential for use in the target acquisition equation. TARACE computes the values of these variables for

three modes of sensor operation - standoff (fixed or vertical), standoff (moving), and penetrating (forward area search). (The equations used in TARACE are described in reference 2, pp. 124-127).

2.2.8.3.1 Programming Specifications. The following table summarizes the principal specifications of TARACE:

Characteristic	Specification
Formal parameters	ITS = Index which specifies mode of sensor operation
	DSX = Horizontal distance from sensor to edge of target division
	DY = Distance from sensor to midline of targeted division zone
	ALT = Altitude of sensor
	ABSDT = Rate at which sensor detects target
	TIME = Length of a target acquisition cycle
	WID = Width of target division
	VEL = Sensor velocity
	DEPTH = Depth of target division
Common blocks	Blank Common, LOCAL3, TAE, TACQ
Subroutines called	None
Called by	TARACA

2.2.8.3.2 Logic Functions. The logic of subroutine TARACE is divided into three separate sections. Each section contains the logic for computing the expected range from sensor to target (RANGE) and the average detection potential (DINTEG), based on a particular mode of sensor operation. For continuously-operating sensors (ITS=1,2,3, or 4), values are computed for both RANGE and DINTEG; for glimpse sensors (ITS= 5,6,7, or 8), only RANGE is calculated.

The index ITS is checked at the beginning of TARACE and the logic flow branches to the appropriate section. The remainder of the code is divided by comment cards which indicate the functions performed by the logic.

a. Section 100 - Standoff (Fixed or Vertical) Sensors. This section of TARACE is executed if ITS=1,2,5, or 6. Equation 8, on p. 124 of reference 2, is used to calculate the value of RANGE for both the fixed (ITS=1 or 5) and the vertical (ITS=2 or 6) mode of sensor operation. The value of DINTEG is calculated according to equation 9 (reference 2, pp. 124-125) for continuously-operating sensors in the fixed mode, and according to equation 9a (reference 2, p. 125) for sensors in the vertical mode.

b. Section 200 - Standoff (Moving) Sensors. This section of TARACE is executed if ITS=3 or 7. The values of RANGE and DINTEG are computed according to equations 10 and 11 (reference 2, p. 125).

c. Section 300 - Penetrating (Forward Area Search) Sensors. This section of TARACE is executed if ITS=4 or 8. The values of RANGE and DINTEG are computed according to equations 12 and 13 (reference 2, p. 125-127)

2.2.8.4 TADPAR. Subroutine TADPAR simulates the acquisi- of targets to the rear of the active battle area of the specified sector by army-air and air force sensors in the penetrating (deep area search) mode. For each type of subunit, the probability that a subunit of that type is detected and the corresponding sensor error and delay time are calculated.

2.2.8.4.1 Programming Specifications. The following table summarizes the principal specifications of TADPAR:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, LOCAL3, TAE, TACQ
Subroutines called	None
Called by	TARACQ

2.2.8.4.2 Logic Functions. The logic of TADPAR is executed for each side. Values are computed for the elements of three arrays: PSRADS, the probability that a subunit of a particular type, located in the first inactive battle area, is detected; TAESRA, the sensor error associated with a particular subunit type and; TADSRA, the delay time associated with a particular subunit type. The code is divided by comment cards that label the functions performed by the logic.

a. Section 100 - Detection Probability, Sensor Error and Delay Time Are User Input. This section of TADPAR begins by checking IUTAM, the index to the use of the target acquisition model. If the model is to be used, the logic flow branches to section 200. Otherwise, user-input values are assigned to PSRADS, TAESRA, and TADSRA and control is returned to TARACQ.

b. Section 200 - Initialize Working Variables. Working variables used in TADPAR are initialized in this section. The expected number of divisions detected in the first inactive battle area and the average area of a reserve division are computed.

c. Section 300 - Compute Detection Probability, Sensor Error and Delay Time. The calculations in this section of TADPAR are performed for each army-air and air force sensor type and for each subunit type. First, the average number of sensors per target division (TEMP1) is computed. The probability that a particular sensor detects a subunit (PASDT) is then calculated based on the swath width at which a sensor will detect a subunit, the velocity of the sensor, the total search time and the average division area. For each subunit type, PSRADS is computed as 1 minus the product, over all

sensor types, of $(1-PASDT)^{TEMP1}$. The probability that a subunit of a given type is acquired by at least one sensor of a given type is used as a weighting factor in the calculation of TAESRA and TADSRA (the sensor error and the delay time).

2.2.8.5 BLKDATA. Subroutine BLKDATA is used to input values for the elements of five arrays used only in the target acquisition model. Values are input for line-of-sight range (RVLOST), weather zone index (IWZBA), terrain index (KTERTA), visibility (VISTWZ) and weather ceiling (CEITWZ).

2.2.8.5.1 Programming Specifications. The following table summarizes the principal specifications of BLKDATA:

Characteristic	Specification
Formal parameters	None
Common blocks	LOCAL3, TAE, TACQ
Subroutines called	None
Called by	N/A

2.2.85.2 Logic Functions. Data statements are used in BLKDATA to input array values at compilation.

2.2.9 LINKH. This subsection describes the routines in LINKH. These routines comprise the ground combat model.

2.2.9.1 GROUND. Subroutine GROUND is a calling routine for routines which account for the ground battle in terms of personnel casualties, weapons destroyed, and the resulting force movement.

2.2.9.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine GROUND:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	GC, FEBAMT
Called by	TMAIN

2.2.9.1.2 Logic Functions. Subroutine GROUND calls routines GC (ground combat) and FEBAMT (FEBA movement).

2.2.9.2 GC. Subroutine GC models the ground combat portion of TACWAR. It computes attrition to people and weapons, by type, based on the total combat value of opposing divisions in the active battle areas of each sector. Included are contributions from aircraft that are assigned to CAS missions in the sectors. The number of weapons assumed repairable after damage is also accounted for by GC, as are the total tons of supplies consumed or destroyed due to the combat action in each active battle area.

2.2.9.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine GC:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, GCFM
Subroutines called	EIGNEV, MPROD, CVFW, CNTRYC
Called by	GROUND

2.2.9.2.2 Logic Functions. Subroutine GC is modeled on a sector-by-sector basis. Therefore, most of its logic is enclosed within a large Do loop that iterates by sector number. The code is divided into sections with comments labeling the functions performed in each one.

a. Section 4 - Adjust the Standard Allocation. This section of GC is executed only on the first cycle of a game. It adjusts the standard allocations of each side's ground weapons and air munitions against its opponent's ground weapons in order that they accurately reflect the opponent's current weapons mix. The computation uses array PWSF, which represents the percentage of each type of weapon in a standard force for a side.

b. Section 5 - Check for Combat Deployment Changes This Cycle. This section of GC begins the sector Do loop which extends through section 80 of the subroutine. Flag variables (IWDRAW and IHOLD) indicating combat deployment changes are also set.

c. Section 10 - Construct Ground-to-Ground Kill Rate Matrices. The calculations in this section are performed for each side. The section begins by calculating WS(IW,L), the number of type IW weapons in side-L divisions in the given sector, for each weapon type. TWS(L), the total number of side-L weapons of all types in the sector, is also calculated. The force ratio of attacker to defender (FRAD) is set to the initial value of 999.0. The values of ground and air forces on attack and defense (VGABA, VGDBA, VAABA and VADBA) and the index to the sector attacker (ISA) are initialized to 0.

The subroutine tests to determine if both sides are weaponless on the current cycle. If so, the logic flow branches to statement 9999, the end of the sector Do loop begun in section 5, since meaningful force ratios cannot be developed. If only one side is weaponless, the other side's ground and air values on attack are set equal to 1 and the logic flow branches to statement 9999.

If both sides have weapons in the sector, the ground-to-ground kill rates matrices for attack and defense (PWAKW and PWDKW), which represent the potential number of each type of the opponent's weapons killed by a singe weapon of each

type, are constructed for both sides. First, the actual allocation of fire of each weapon type against each opposing weapon type (AAWA and AAWD) is computed for attack and defense. The kill rate for each weapon type is computed by multiplying the allocation of fire by the individual weapon value (on attack (VIWAW) or on defense (VIWDW)). Since SAMs have no ground value, they are excluded from the above calculations.

d. Section 15 - Construct Air-to-Ground Kill Rate Matrices. This section of GC computes PAAKW and PADKW, the number of each type of the opponent's ground weapons killed by a single aircraft of each type on attack and defense. The calculations in this section, which are based on notional load (AMNL and AMLFD), actual allocation of fire (AAMA and AAMD) and individual air munition value (VAMAW and VAMDW), are similar to those in section 10.

e. Section 20 - Compute the Value of an Individual Weapon and an Individual Aircraft Sortie by Anti-Potential Potential. This section first transfers the data in the three-dimensional kill-rate matrices, PWAKW and PWDKS, to one-dimensional arrays, BSUM and RSUM, so that these values may be passed to subroutine EIGENV. Individual ground weapon values on attack and defense (VIWACF and VIWDCF) are computed by subroutine EIGENV. Subroutine MPROD is used to compute the values of individual aircraft sorties (VIAACF and VIADCF).

f. Section 25 - Calculate TOE and Current Weapon Values, Effectiveness of Divisions in Active Battle Areas, and Ground Values on Attack and Defense. The calculations in this section of GC are performed for each side. The values of ground forces on attack and defense (VGABA and VGDBA) are initially set equal to 0. If a side has no divisions in the sector, the remainder of the calculations in this section are not performed for that side. First, total TOE weapon values for a division on attack and on defense (WVDAT, WVDDT) are computed for each division type. Then, the total actual weapon values for a division on attack and defense (WVDAC, WVDDC) are computed for each division. The fractional personnel strength (PPS) is calculated for each division as the ratio of the actual number of people in the division to the TOE number of people for the division. Subroutine CVFW is called twice in order to evaluate the functions which give PEA and PED, the fractional effectiveness of a division, based on personnel strength, for attack and defense respectively. The

number of days of supplies on hand is computed and CVFW is called to evaluate the function which gives SEF, the fractional effectiveness of a division due to supply shortages. The effectiveness of each division on attack (EFFDA) and on defense (EFFDD) is then computed based on PEA(PED), SEF and the ratio of actual to TOE weapon values. The ground values of each division on attack and defense (VDAC and VDDC) and the total values of side-L ground forces in the sector (VGABA and VGDBA) are then computed based on division effectiveness and current weapons values.

g. Section 30 - Calculate Air Values on Attack and Defense. GC now computes attack and defense air values for both sides. To determine the value of aircraft on attack in active battle areas (VAABA), the subroutine multiplies the number of successful CAS sorties for the type aircraft by the value of an individual type aircraft on attack and sums over all aircraft types. Air values on defense (VADBA) are computed similarly.

h. Section 35 - Compute the Sector Attacker if There is an Attack in Sector IS. This section of GC begins by setting the force ratio (FRAD) of attacker to defender to 999 and determining the posture (KP) in the sector. The index to the theater defender (II) is then set. If the total air and ground value on defense (VGDBA and VADBA) for either side is ≤ 0, the other side is considered to be the sector attacker and the index to sector attacker (ISA) is set appropriately. Otherwise, FRAD is computed as the standard force ratio for the theater attacker on attack. A maximum force ratio of 999.9 is allowed. If FRAD is $\geq$ FRATA, the minimum force ratio necessary for the theater at attacker to attack in posture KP, and if the given sector is not constrained due to an exposed flank, ISA is set equal to ITA, the index to the theater attacker. Otherwise, FRAD is computed as the standard force ratio for the theater defender on attack and compared to FRATD, the minimum force ratio necessary for the theater defender to attack in posture KP. If FRAD $\geq$ FRADT, ISA is set equal to II. If neither side has a large enough force ratio to attack, ISA is set equal to -1. If the theater attacker is constrained due to an exposed flank and the theater defender cannot attack, ISA is set equal to 0.

Once the index to sector attacker has been set, the model checks to see if there have been any changes in combat

deployment in the sector on the current cycle. If so, ISA is set equal to 0. Finally, if ISA=0 (i.e. if there is a holding situation where neither side is attacking in the sector), FRAD is also set equal to 0.

i. Section 40 - Calculate Effectiveness Reduction Due to Supply Shortage in Sector. For each side L, this section places a value of -999.0 in variable PPESE, the percent of ground troops that are affected by the enemy's firepower when supply limitations are considered. If a side has no divisions in the sector, the remainder of this section is skipped for that side. The actual number of supplies in all divisions are summed in variable TEMP. The planned supply consumption rate for all divisions is summed in variable TEMP1. Days of supply on hand (DSH) is then determined by dividing TEMP by TEMP1. Subroutine CVFW is then called to compute the supply effectiveness factor (SEF) as a function of days of supply on hand. PPESE is then set equal to its previous value or SEF, whichever is greater.

j. Section 45 - Compute Value Lost for Both Sides. GC skips this section if there is a holding situation in the sector. The force ratios used to determine personnel casualties to the attacker and to the defender (FRCA and FRCD) are initially set to 999.9. The fractions of attacker's and defender's CAS sorties that are considered when computing the force ratio for casualties to the attacker (FASFRC) and to the defender (FDSFRC) are used to adjust the standard force ratios in order to compute FRCA and FRCD. The percent casualties (PCS) to the attacker and to the defender are calculated as a function of force ratio and posture by subroutine CVFW, then the total ground value lost is computed for each side as the product of (1) that side's total ground value (VGABA or VGDBA), (2) PCS, and (3) FCVLS, the factor for scaling percent casualties to obtain (weapon) value lost.

k. Section 50 - Compute Value Lost for Holding Posture. The calculations in this section are performed for both sides. The percentage of casualties in the sector (PCS) is computed by multiplying FCHP, the fraction of casualties in a division when the division is in a holding posture, by PPESE, the percent of people affected by the enemy's firepower. The value lost is then computed as it was in section 45.

l. Section 54 - Compute Tons of Supplies Consumed by Both Sides and New Supply Inventories. The calculations in this section are performed for both sides. Section 54 begins by checking to see if there is a sector attacker; if not, the posture is set to a holding posture. If a side has no divisions in the sector, the remainder of the section is skipped for that side. The section computes tons of supplies consumed by weapons and personnel for each division in the sector. The calculations performed are described in detail in ref. 2, p. 9. The amount of supplies remaining in each division (SDIV) and the total amount of supplies consumed by each side are computed.

m. Section 55 - Calculate Casualties to Both Sides and Subtract Out Losses. The calculations in this section are performed for both sides. If a side has no divisions in the sector, the section is skipped for that side. The actual number of people in each division (PDIV) is multiplied by the percent casualties (PCS) to obtain the number of casualties. The number of personnel remaining in each division and total casualties in the sector this cycle (PLS) are computed. The cumulative casualties in the sector (CPLS) is then updated.

n. Section 60 - Calculate Weapons Lost by Type. The calculations in this section are performed on both sides. The specific calculations performed depend on which side is the sector attacker. The potential number of weapons lost (SUMM) is computed for each weapon type by summing, over all enemy weapon and aircraft types, the product of (1) the potential number of weapons killed by a single enemy weapon or CAS sortie (PWAKW, PWDKW, PAAKW or PADKW) and (2) the number of enemy weapons or successful CAS sorties (WS or ACSABA). The potential weapon value lost is then computed for each weapon type as the product of SUMM and the value of an individual weapon of the given type (VIWDCF or VIWACF). The actual ground value lost (VLS)is then used to scale the potential losses to give actual weapon losses (WLS) for each weapon type. The number of weapons remaining in the sector (WS) is computed and cumulative weapon losses (CGKGS, CAKGS and CWLS) are updated.

o. Section 65 - Compute Weapons Lost by Type for Each Division. The calculations of this section of GC are performed for both sides, but if a side has no divisions in the sector, the section is skipped for that side. The

number of weapons of a given type lost in a division is proportional to the number of weapons of that type in that division. The number of weapons remaining in each division in the sector (WDIV) is computed in this section.

p. Section 70 - Compute Tons of Supplies Lost. The calculations in this section are performed for both sides. Total tons of supplies lost in the active battle area of the sector (TSL) are computed based on the tons of supplies lost for each enemy weapon in combat or for each successful CAS sortie (SLWCB or SLSCAS) and the numbers of enemy weapons and successful CAS sorties (WS and ACSABA). Cumulative supplies destroyed by ground weapons and aircraft (CSDGW and CSDAW) are updated. Supply losses are apportioned among divisions in the section in proportion to the amount of supplies in each division. Supplies remaining in each division (SDIV) are also computed.

q. Section 75 - Compute Number of Weapons Repairable After Ground Combat. Calculations in this section of GC are skipped for a side if it has no divisions in the sector. The tactical situation (I) is established based on the sector attacker and the posture. The array WDRRP, the number of weapons of each type that have been damaged and recovered and are assumed repairable is computed based on (1), PWATS, the fraction of weapons of each type that are damaged and abandoned when the unit is in tactical situation I, (2) PWDRP, the fraction weapons of each type that are damaged but are assumed to be repairable, and (3) WLS, the number of weapons lost (i.e., damaged).

r. Section 80 - Calculate Casualties and Weapon Losses Inflicted by Air, Per Sortie. The calculations in this section of GC are performed for both sides. The particular calculations performed depend on which side is the sector attacker. The ratio of enemy air value to total side-L air and ground value is used to represent the fraction of side-L casualties and weapon losses inflicted by enemy air forces. Total side-L casualties (PLS) and weapon losses (WLS) in the sector are multiplied by this ratio to obtain CDACS and WLDAS, the number of side-L casualties and weapon losses inflicted by enemy air forces. Casualties and weapon losses inflicted per CAS sortie are then computed by dividing CDACS and WLDAS by the total number of successful enemy CAS sorties.

Statement 9999, which concludes this section of GC, ends the sector Do loop begun in section 5.

s. Section 85 - Average Casuslties and Weapon Losses Per Sortie Over Regions. In this section of GC, FPDCAS(IR), the average number of personnel casualties inflicted per CAS sortie, and FWDCAS(IW,IR) the average number of type IW weapons destroyed, are calculated in a straightforward manner for each region IR.

2.2.9.3 FEBAMT. Subroutine FEBAMT computes the movement of the forward edge of the battle area (FEBA) in each sector in the theater. The ground model assumes that FEBA movement is a function of force ratio, posture, terrain, and the mobility of the attacker's division.

2.2.9.3.1 Programming Specifications. The following table summarizes the principal programming specifications of subroutine FEBAMT:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, GCFM
Subroutines called	CVFW
Called by	GROUND

2.2.9.3.2 Logic Functions. Subroutine FEBAMT is modeled on a sector-by-sector basis. The coding is annotated with comments labeling the functions performed by the logic.

a. Section 5 - Compute Effectiveness of All Divisions. Section 5 of FEBAMT begins by testing to see if the subroutine will use the security force ratio constraint to movement. If IUSFRC=1, it will use the constraint; if IUSFRC=0, this section is skipped. For both sides, the current weapons value and the percent personnel strength of each division are calculated. A call is made to CVFW to compute the fractional effectiveness of each division based on personnel strength. A subsequent call to CVFW is made to determine the supply effectiveness factor as a function of days of supply on hand. This information is used to determine the effectiveness (EFFDD) and the combat value (VDDSF) of each division on defense.

b. Section 9 - Sort FEBA Distances in Increasing Order for Advancing. This section of FEBAMT sorts the sectors in the theater and stores the sector numbers, in order of increasing FEBA, in the array ISUM.

c. Section 10 - Compute Mobility Factor in Sector. This section begins by setting L equal to the index to the theater attacker. This section and section 15 are performed consecutively for each side. Two variables, ASIGN and ISIGN, are set to 1 if L=2 and to -1 if L=1. These variables are used in FEBA movement calculations to account for the fact that the absolute FEBA location (FEBA) increases as side 2 advances and decreases as side 1 advances. The remainder of the logic of this section and of section 15 is then performed on a sector by sector basis (in order of increasing FEBA advance from the point of view of side L) for each sector in which side L is the sector attacker. An average mobility factor (FMS) is computed based on posture, terrain and the individual mobility factors of side-L divisions in the sector.

d. Section 15 - Compute FEBA Change for Sector Attacker. Section 15 begins by calling subroutine CVFW to calculate the FEBA movement (FM), which is multiplied by the mobility factor (FMS) to determine CFEBA, the change in the FEBA in the sector since the last cycle. If the posture (KPS) is that of a breakthrough, CFEBA is adjusted by FMBP, the percent increase in FEBA movement in a breakthrough. The amount of exposure of each of the sector's flanks is computed and compared to FEAFBA, the maximum flank exposure acceptable to the attacker while advancing. If the exposure of both flanks is acceptable, the logic branches around further FEBA movement adjustments to statement 1560 in order to compute the new FEBA location. Otherwise, CFEBA is adjusted in order to avoid excess flank exposure. If the security force ratio constraint is not to be used, CFEBA is adjusted to account for withdrawal of the defender if there was a change in the defender's combat deployment and the logic flow branches to statement 1560.

If the security force ratio constraint is to be used, the model sets SUMM(2) equal to FEISF, the flank exposure increment (in addition to FEAFBA) the attacker (side L) will accept with a security force ratio of SFRFE(2,L). The location of the active battle area is checked. If it is the rearmost battle area for side L in the sector, further force ratio computations are skipped, CFEBA is adjusted to account for any withdrawal of the defender and the logic flow branches

to statement 1560. Otherwise, the total combat value of the attackers divisions in the first inactive battle area of the given sector (VALA) is computed. Then the total combat value of the defender's divisions in the flanking sector or sectors causing a flank exposure constraint (VALD) is computed. The security force ratio FR is set to the ratio of VALA to VALD. Subroutine CVFW is called to compute, based on FR, the additional FEBA movement (FM) acceptable to the attacker. CFEBA is adjusted appropriately to account for this and for any withdrawal of the defender due to a change in combat deployment.

The new FEBA location can now be determined. The FEBA is moved a distance equal to CFEBA. If in this movement the FEBA does not reach or cross any interval boundaries, the FEBA location (FEBA) is set at the computed location and sections 10 and 15 are repeated for the next sector. If the new FEBA location is at the boundary to the next interval, posture and terrain indices are updated, FEBA is set to the interval boundary and sections 10 and 15 are repeated for the next sector. If the FEBA has crossed the boundary to the next interval, the FEBA location is set back to the interval boundary, the interval, posture and terrain indices are updated and the logic flow branches back to statement 1004 in order to determine the FEBA movement in the new interval. The variable PTINI is used to indicate the fraction of total FEBA movement that will occur in the interval under consideration. If this loop is repeated 100 times, execution stops and an error message is output.

When the FEBA changes have been made for all sectors in which the theater attacker is attacking, L is set equal to the index to the theater defender and sections 10 and 15 are repeated for those sectors in which the theater defender is attacking.

e. Section 17 - Withdraw Due to Combat Deployment. This section withdraws the FEBA according to the input variable DDWDCD, the distance a defender withdraws due to a change in combat deployment, in each sector where there has been a change in combat deployment for the defender.

f. Section 18 - Determine if Theater Attacker Changes for Next Cycle. This section of FEBAMT initially sets a temporary variable, IFLAG, equal to the index to the current theater attacker. If the theater attacker is not attacking

or holding due to constrained flank in at least one sector, then IFLAG is reset to the index to the theater defender. The sector attacker index (ISA) is reset to 0 (holding) for each sector in which ISA=-1 (i.e. in each sector in which neither side is strong enough to attack).

g. Section 19 - Sort FEBA Distances in Increasing Order. This section sorts the new FEBA locations into increasing order. The logic is similar to that of section 9.

h. Section 20 - Compute FEBA Change if Defender Withdraws Due to Exposed Flank. This section begins by setting L equal to the index to the theater defender. The section is performed first for the theater defender, then for the theater attacker. The logic is performed on a sector by sector basis (in order of decreasing FEBA advance from the point of view of side L) for each sector in which side L is the sector defender. For each sector, the amount of exposure of each of the sector's flanks is computed and compared to FEDW, the maximum flank exposure the defender will accept before withdrawing. If necessary, the FEBA is withdrawn so that the flank constraint is not violated. If side L will be the theater defender on the next cycle and has withdrawn in a sector a flag (ITDWDS) indicating withdrawal by the theater defender in that sector is set equal to 1.

i. Section 21 - Determine Whether the Theater Attacker is Constrained due to Exposed Flank. Section 21 of FEBAMT begins by setting ITA, the theater attacker index for the next cycle, equal to IFLAG. The logic of this section is performed for each sector. First ISCEF, the indicator for a sector being constrained due to an exposed flank, is set to zero, indicating no constraint. The amount of exposure of each of the sector's flanks is computed and compared to FEAFBA, the maximum flank exposure the attacker will accept while advancing. If the exposure of each of a sector's flanks is less than FEAFBA, no further calculations are made for that sector. If the exposure of either of a sector's flanks exceeds FEAFBA, and the security force ratio constraint is not to be used, ISCEF is set equal to 1 for that sector. If the security force ratio constraint is to be used, the security force ratio and the additional acceptable flank exposure are computed as they were in section 15. Then, if a sector's actual flank exposure is still unacceptable, ISCEF is set equal to 1 for that sector. Control is returned to GC at the end of this section.

2.2.10 LINKI. This subsection describes the routines in LINKI. These routines comprise the air-ground combat model.

2.2.10.1 AIRGRD. Subroutine AIRGRD provides the culmination of all the assessments that have occurred in previous air subroutines. According to sheltering priorities, AIRGRD calculates the numbers of sheltered and unsheltered aircraft on each notional airbase. It calls subroutines ATRTAB and ASGATR, which assess attrition on the ground due to airbase attack and interdiction missions, respectively. Subroutine QRAFIL is called to adjust inventories of QRA aircraft. AIRGRD also contains the logic for repairing damaged aircraft and SAMs.

2.2.10.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine AIRGRD:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	ASGATR, ATRTAB, QRAFIL, APORTN
Called by	TMAIN

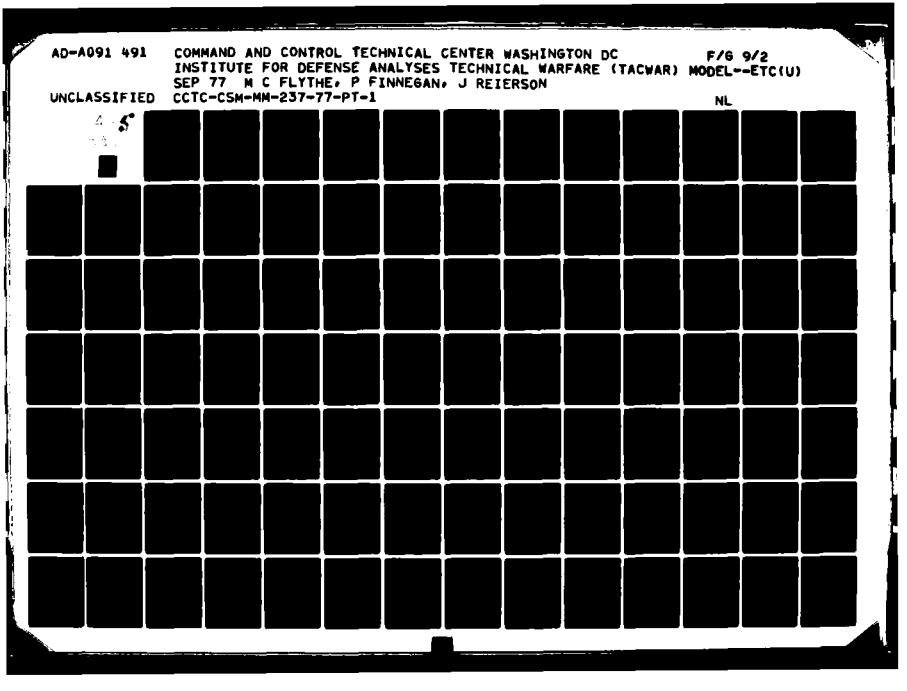
2.2.10.1.2 Logic Functions. AIRGRD begins by calling ASGATR which computes attrition due to interdiction (INT) missions. The remainder of the code of AIRGRD is divided by comment cards which label the functions performed by the logic.

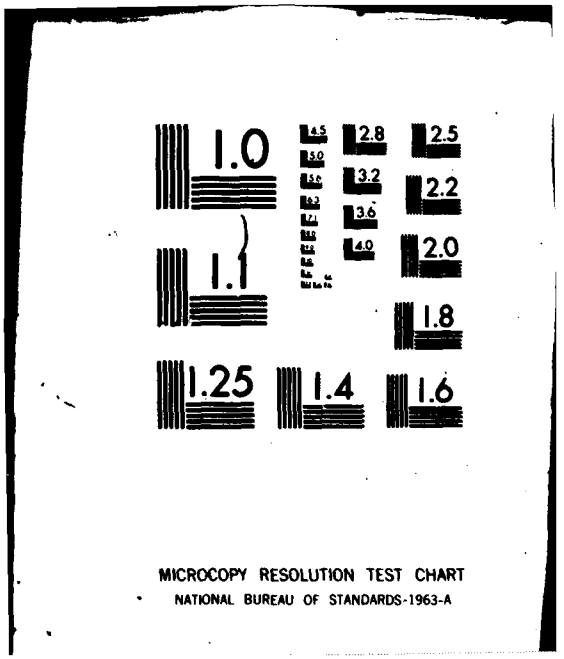
a. Section 590 - Calculate Number of Sheltered and Unsheltered Aircraft on Forward Airbases. For each side, this section and section 600 of AIRGRD are executed consecutively for each sector. The index IPSHLA(J,L) indicates the order of priority for assigning aircraft to shelters. If IPSHLA(1,L)=K, then aircraft of type K are the first to be sheltered. This index is also used to indicate aircraft types which are never sheltered, e.g. IPSHLA(J,L)=0 indicates that no type J aircraft are to be sheltered. It is assumed that if no aircraft of a particular type can be sheltered, then no aircraft of any higher numbered type can be sheltered either.

The number of aircraft of a given type assigned to shelters at the sector's forward airbases is equal to the minimum of: (1) the number of aircraft of that type that require sheltering and (2) the number of unoccupied shelters remaining. AIRGRD assigns QRA aircraft to shelters first, according to the order of priority specified by IPSHLA. Then, if all the QRA aircraft stationed at the airbase have been sheltered, non-QRA aircraft are assigned to any remaining shelters. The fraction (FAUSHL) of aircraft that can use another aircraft's shelter while that aircraft is flying its mission is used to determine the number of shelters required for non-QRA aircraft. The numbers of sheltered and unsheltered aircraft of each type are computed and stored in the arrays ASHEL and AOPEN. These computations are based on the number of shelters assigned to aircraft of a given type, the fraction (FCAIA) of a cycle that an aircraft of that type is in the air and the fraction (FAGSCN) of aircraft of that type that are on the ground and assigned to shelters but are caught unsheltered at the time of attack. The total numbers of sheltered and unsheltered aircraft of all types (TASHEL and TAOPEN) and the total number of assigned shelters (FULSHEL) are also computed. If detailed results are to be printed, AIRGRD outputs ASHEL, AOPEN, TASHEL and TAOPEN to file JCON.

b. Section 600 - Compute Attrition on Ground Due to ABA Attacks Against Forward Airbases. This section of AIRGRD begins by computing TAAB, the number of actual airbases attacked in the forward sector, based on the number of actual airbases comprising the sector's notional airbase (ABASEF), the number of successful ABA sorties against forward airbases in the sector (SABAF), and the minimum raid size for attacking airbases (RSMIN). The number of sorties per attacked airbase is then computed. The numbers of shelters, sheltered aircraft and unsheltered aircraft at the sector's notional forward airbase are each divided by ABASEF in order to obtain the numbers of shelters, sheltered aircraft and unsheltered aircraft at a typical actual airbase. For each type of attack aircraft, the probability of kill to be used for ABA assessment is calculated as a weighted average of the probabilities of kill of aircraft that attack forward airbases from forward, rear and COMMZ airbases. AIRGRD then calls ATRTAB, which calculates the numbers of shelters, sheltered aircraft and unsheltered aircraft killed at a typical actual airbase. Overall attrition at the forward

AD-A091 491 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 9/2
INSTITUTE FOR DEFENSE ANALYSES TECHNICAL WARFARE (TACWAR) MODEL--ETC(U)
SEP 77 M C FLYTHE, P FINNEGAN, J REIERSON
UNCLASSIFIED CCTC-CSM-MM-237-77-PT-1 NL




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

notional airbase is obtained by multiplying these numbers by TAAB. Aircraft kills are then apportioned among QRA and non-QRA aircraft of each type according to the sheltering scheme implemented in section 590. The numbers of aircraft and shelters surviving are computed and the cumulative numbers of QRA and non-QRA, sheltered and unsheltered aircraft killed are updated. If detailed results are to be printed, then the numbers of shelters killed and surviving and, for each aircraft type, the numbers of non-QRA, sheltered QRA, unsheltered QRA and total QRA aircraft killed are output to file JCON.

c. Section 610 - Calculate Number of Sheltered and Unsheltered Aircraft on Rear Airbases. For each side, section 610 and section 620 of AIRGRD are executed consecutively for each sector. The logic structure of this section is similar to that of section 590.

d. Section 620 - Compute Attrition on Ground Due to ABA Attacks Against Rear Airbases. The logic of this section of AIRGRD is similar to that of section 600.

e. Section 630 - Calculate Number of Sheltered and Unsheltered Aircraft on COMMZ Airbases. Section 630 and section 640 of AIRGRD are executed consecutively for each side. The logic structure of section 630 is similar to that of sections 590 and 610.

f. Section 640 - Compute Attrition on Ground Due to ABA Attacks Against COMMZ Airbases. The logic structure of this section of AIRGRD is similar to that of sections 600 and 620.

g. Section 645 - Compute Aircraft Remaining. This section of AIRGRD computes the number of aircraft surviving, by type, at each notional airbase in the theater. If detailed results are to be printed, the numbers computed in this section are output to file JCON.

h. Section 650 - Convert Aircraft to QRA Aircraft to Eliminate QRA Shortages. This section of AIRGRD calls subroutine QRAFIL to adjust the QRA inventories at notional airbases.

i. Section 700 - Repair Damaged Aircraft. This section of AIRGRD is executed once for each side. The number of aircraft of each type repaired during the current cycle (REPARA) is calculated as the product of RRAPL, the repair rate and DAMPL, the number of aircraft in the repair pool. The total number of aircraft repaired is also computed. If this total exceeds AMXRPL, the maximum number of aircraft that can be repaired in one cycle, then REPARA is proportionally reduced for each aircraft type so that the total number of aircraft repaired equals AMXRPL. DAMPL is then reduced by REPARA and the cumulative number of aircraft of each type repaired is updated. Repaired aircraft are returned to forward, rear and COMMZ notional airbases in proportion to the number of aircraft already stationed at each airbase. If detailed results are to be printed, then the number of aircraft at each notional airbase and the number of aircraft remaining in the repair pool are output to file JCON.

f. Section 710 - Repair Damaged SAMs. This section of AIRGRD calculates the number of long, medium and short range SAMs repaired in the current cycle. Repaired SAMs are then reassigned to an appropriate mission in a region or in the COMMZ in proportion to the number of SAMs already assigned to that mission in that location. The logic of this section is similar to that of section 700. Execution of AIRGRD concludes with a call to APORTN which apportions resources from notional to actual airbase.

2.2.10.2 ATRTAB. Subroutine ATRTAB computes attrition to shelters and to aircraft on the ground due to airbase attack. ATRTAB provides a choice among three sets of attrition equations which are based upon three different sets of assumptions. A user-supplied index designates which set of equation is to be used. Losses computed in ATRTAB are those which would occur at a typical actual airbase.

2.2.10.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ATRTAB:

Characteristic	Specification
Formal parameters	AA = Number of attack sorties
	SS = Number of shelters
	AS = Number of sheltered aircraft

Characteristic	Specification
	AN = Number of unsheltered aircraft
	PARK = Number of parking areas
	PDS = Probability of detecting a shelter
	PDN = Probability of detecting an unsheltered aircraft
	FSK = Fraction of shelters killed when attacked so that the aircraft inside would be killed
	PKAN = Probability of killing an unsheltered aircraft
	PKAS = Probability of killing a sheltered aircraft
	TPS = Number of passes per ABA sortie
	N1 = Number of types of attack aircraft
	IEQ = Index to method for computing attrition
	SSK = Number of shelters killed
	ASK = Number of sheltered aircraft killed
	ANK = Number of unsheltered aircraft killed
Common blocks	None
Subroutines called	None
Called by	AIRGRD

2.2.10.2.2 Logic Functions. The logic of subroutine ATRTAB is divided into three separate sections each of which is based on different assumptions; however, the general logic structure of these sections is similar. In each section, the variables representing number killed and expected fraction surviving for each of three target types (shelters, sheltered aircraft and unsheltered aircraft) are initialized. It is assumed that attackers cannot distinguish occupied shelters from empty ones. The number of targets at the airbase is checked; if no targets exist, control is returned to AIRGRD.

For each type of attack aircraft, the total number of passes made at an airbase (TEXP) is computed. Each section of ATRTAB then uses some form (or forms) of the single engagement binomial attrition equation to compute, for each type of attacker and for each target type, the expected fraction of targets that survive the attack. The particular form of the equation used depends upon the section's assumptions. For a particular target type, the expected fraction of targets killed (EFK) is given by the equation:

$$EFK = 1 - \prod_{i=1}^{N1} EFS_i$$

where EFS_i is the expected fraction of targets surviving attack by type i attackers.

The number of targets killed is then computed by multiplying EFK by the number of targets at the airbase.

The three sections of code in ATRTAB are separated by comment cards which label the functions performed by the logic.

a. Section 100 - Compute Attrition Assuming Shelters Are Attacked Only if No Unsheltered Aircraft Are Detected. The logic of this section is based on the assumption that, on any pass, an attacker attempts to kill a shelter only if he does not detect any unsheltered aircraft. For each type of attacker, the expected fraction of sheltered aircraft surviving (EFS) is computed using the following equation:

$$EFS = \left[1-PKS \left(1- \left[1-PDS\right]^{SS} \right) \left(1-PDN\right)^{AN} \right]^{TEXP}$$

where

$$PKS = \min \left\{ \frac{PKAS}{SS}, \quad PKAS \right\}$$

The term $(1 - PDN)^{AN}$ accounts for the probability that no unsheltered aircraft are detected on a particular pass. The PK term, in this and in all subsequent equations in ATRTAB, accounts for the assumption that an attacker who detects several targets randomly selects a particular target to attack. The expected fraction of shelters surviving is computed using the equation

$$EFS = \left[1-PKSS \left(1-\left[1-PDS\right]^{SS} \right) \left(1-PDN\right)^{AN} \right]^{TEXP}$$

where

$$PKSS = \min \left\{ \frac{PKAS*FSK}{SS}, \; PKAS*FSK \right\}$$

The expected fraction of unsheltered aircraft surviving is computed using the equation

$$EFS = \left[1-PKN \left(1-\left[1-PDN\right]^{AN}\right) \right]^{TEXP}$$

where

$$PKN = \min \left\{ \frac{PKAN}{DENOM}, \; PKAN \right\}$$

and

$$DENOM = \text{the number of occupied parking areas}$$

$$= \min \left\{ PARK, \; AN \right\}$$

The variable DENOM accounts for the assumption that an attacker shooting at one unsheltered aircraft may kill another unsheltered aircraft located on the same parking area. Aircraft are assumed to be distributed uniformly among parking areas. If the number of parking areas exceeds the number of unsheltered aircraft, it is assumed that there is one aircraft on each AN parking area and none on the others.

b. Section 200 - Compute Attrition Assuming Optimal Allocation of Attackers to Targets. The logic of this section of ATRTAB is based on the assumption that ABA aircraft are allocated to targets in such a way that the total number of aircraft killed on the ground is maximized. The total number of aircraft killed (TTK) is given by the equation:

$$TTK = AN\left(1-\left[1-PKN\left(1-[1-PDN]^{AN}\right)\right]^{FAAAN*TEXP}\right)$$

$$+ AS\left(1-\left[1-PKS\left(1-[1-PDS]^{SS}\right)\right]^{(1-FAAAN)*TEXP}\right)$$

where

FAAAN = the fraction of ABA aircraft that attack unsheltered aircraft

and PKN and PKS are as defined in Section 100.

The allocation algorithm used in ATRTAB was derived by differentiating this equation with respect to FAAAN and solving for the value of FAAAN ($0 \leq FAAAN \leq 1$) that maximizes TTK (see reference 12).

ATRTAB computes the optimal values of FAAAN and FAC, the fraction of ABA aircraft that attack unsheltered aircraft and the fraction that attack shelters. For each type of attacker, the expected fractions of targets surviving are then computed using equations which reflect the appropriate allocation of attack aircraft to targets. These equations are:

(1) For sheltered aircraft

$$EFS = \left[1-PKS\left(1-[1-PDS]^{SS}\right)\right]^{FAC*TEXP}$$

(2) For shelters

$$EFS = \left[1-PKSS\left(1-[1-PDS]^{SS}\right)\right]^{FAC*TEXP}$$

(3) For unsheltered aircraft

$$EFS = \left[1-PKN\left(1-[1-PDN]^{AN}\right)\right]^{FAAAN*TEXP}$$

c. Section 300 - Compute Attrition Assuming Shelters and Unsheltered Aircraft Are on the Same Parking Areas. The logic of this section of ATRTAB is based on the assumption that an attacker may kill any target on a parking area if he detects the parking area (i.e. if he detects a target located on that parking area). Targets are assumed to be uniformly distributed among parking areas. Both shelters and unsheltered aircraft are assumed to be located on the same parking areas. The probability of detecting a parking area is calculated as a weighted average of the probabilities of detecting shelters and unsheltered aircraft. Then, for each type of attacker, the expected fractions of targets surviving are computed as follows:

(1) For sheltered aircraft

$$EFS = \left[1-PKS \left(1-[1-PDP]^{TAC} \right) \right]^{TEXP}$$

where

PDP = Probability of detecting a parking area

TAC = Number of occupied parking areas

= min {PARK, SS+AN}.

(2) For shelters

$$EFS = \left[1-PKSS \left(1-[1-PDP]^{TAC} \right) \right]^{TEXP}$$

(3) For unsheltered aircraft

$$EFS = \left[1-PKN \left(1-[1-PDP]^{TAC} \right) \right]^{TEXP}$$

2.2.10.3 QRAFIL. Subroutine QRAFIL adjusts the inventories of QRA aircraft in order to maintain a user-specified minimum number of aircraft in the QRA role at each notional airbase. The subroutine converts non-QRA aircraft to the QRA role to eliminate or reduce shortages in QRS inventories, and computes cumulative numbers of aircraft converted at forward, rear, and COMMZ airbases. QRAFIL records any shortages it could not eliminate on output file JSUM.

2.2.10.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine QRAFIL:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	None
Called by	AIRGRD

2.2.10.3.2 Logic Functions. The logic of subroutine QRAFIL is performed once for each side. The code is divided by comment cards that label the functions performed by the logic.

a. Section 100 - Set Side and Sector Indices and Compute Total QRA Inventories at Forward and Rear Notional Airbases. This section of QRAFIL begins the side Do loop that extends over the entire subroutine and the sector Do loop that contains sections 100 through 160. It also computes the total number of QRA aircraft stationed at each of the two airbases in the sector.

b. Section 105 - Convert Forward Non-QRA Aircraft to Forward QRA Aircraft According to Desired Proportions of QRA Types. This section of QRAFIL is executed only if there is a shortage of QRA aircraft at the sector's forward airbase (i.e. if aircraft losses have caused the number of QRA aircraft on the airbase to fall below DQRAF, a user-specified minimum desired number of aircraft). Otherwise, the logic flow branches to section 120.

First, the individual shortage of each type of QRA aircraft is computed. Then, non-QRA aircraft of the same type already stationed at the forward airbase are converted to the QRA role in order to eliminate or reduce the shortage. The index IQRAP(J,L) indicates the aircraft type that is the J^{th} QRA type for side L. It is assumed that if no aircraft of a particular type can be converted, then no aircraft of any higher numbered type can be converted either. The number of aircraft converted is equal to the minimum of: (1) the shortage of the type of QRA aircraft being considered, (2) the number of non-QRA aircraft of that type that are stationed

at the airbase, and (3) the overall QRA shortage. As conversions are made, the aircraft inventories are adjusted appropriately and the cummulative number of aircraft converted to the forward QRA role is updated for each QRA type.

c. Section 110 - Convert Forward Non-QRA Aircraft to Forward QRA Aircraft According to User-Specified Priorities. This section of QRAFIL is executed only if the shortage of QRA aircraft at the sector's forward airbase was not eliminated in section 105. Otherwise, the logic flow branches to section 120. The index IPSQRA(J,L) indicates the order of priority for converting additional aircraft to the QRA role for side L. If IPSQRA(1,L)=K, then aircraft of QRA type K are the first to be converted. This index is also used to indicate aircraft types which can never be used in the QRA role, e.g., IPSHLA(J,L)=0 indicates that type J aircraft cannot be used as QRA aircraft. The number of aircraft converted is equal to the minimum of: (1) the remaining QRA shortage and (2) the number of non-QRA aircraft of the appropriate type that are stationed at the forward airbase. This conversion process is repeated for successively lower priority QRA types, until either the QRA shortage has been eliminated or all available aircraft of the lowest priority type have been converted to the QRA role. As conversions are made, the aircraft inventories are adjusted appropriately and the cumulative number of aircraft converted to the forward QRA role is updated for each QRA type.

d. Section 120 - Convert Rear Non-QRA Aircraft to Rear QRA Aircraft According to Desired Proportions of QRA Types. This section of QRAFIL is executed only if there is a shortage of QRA aircraft at the sector's rear airbase. Otherwise, the logic flow branches to section 140. The logic structure is similar to that of section 105.

e. Section 130 - Convert Rear Non-QRA Aircraft to Rear QRA Aircraft According to User-Specified Priorities. This section of QRAFIL is executed only if the shortage of QRA aircraft at the sector's rear airbase was not eliminated in section 120. Otherwise, the logic flow branches to section 140. The logic structure is similar to that of section 110.

f. Section 140 - Convert Rear Non-QRA Aircraft to Forward QRA Aircraft. This section of QRAFIL is executed only if the sector's rear airbase has a full QRA inventory but there remains a QRA shortage at the sector's forward airbase. If the forward airbase has a full inventory but there remains a shortage at the rear airbase, the logic flow branches to section 150. If both airbases have a full QRA inventory, or if neither one does, then the logic flow branches to section 160.

In this section, non-QRA aircraft stationed at the rear airbase are converted to the QRA role at the forward airbase according to the conversion priority index IPSQRA. The logic structure is similar to that of section 110.

g. Section 150 - Convert Forward Non-QRA Aircraft to Rear QRA Aircraft. This section of QRAFIL is executed only if the sector's forward airbase has a full QRA inventory but there remains a QRA shortage at the sector's rear airbase. The logic structure is similar to that of section 140.

h. Section 160 - Output Forward and Rear QRA Shortages. This section of QRAFIL records on file JSUM any QRA shortages at sector airbases that were not eliminated in the preceeding sections. It is the last section in the sector Do loop begun in section 100.

i. Section 200 - Compute Total QRA Inventory at the COMMZ Notional Airbase and Convert COMMZ Non-QRA Aircraft to COMMZ QRA Aircraft According to Desired Proportions of QRA Types. This section of QRAFIL begins by computing the total number of QRA aircraft stationed at the COMMZ airbase. The remainder of the section is executed only if there is a shortage of QRA aircraft at the COMMZ airbase. Otherwise, the logic flow branches to section 220. The logic structure is similar to that of section 105.

j. Section 210 - Convert COMMZ Non-QRA Aircraft to COMMZ QRA Aircraft According to User-Specified Priorities. This section of QRAFIL is executed only if the shortage of QRA aircraft at the COMMZ airbase was not eliminated in section 200. Otherwise, the logic flow branches to section 220. The logic structure is similar to that of section 110.

k. Section 220 - Convert Forward and Rear Non-QRA Aircraft to COMMZ QRA Aircraft. This section of QRAFIL is executed only if the shortage of QRA aircraft at the COMMZ airbase was not eliminated in sections 200 and 210. Otherwise, the logic flow branches to section 230.

Non-QRA aircraft stationed on forward and rear sector airbases throughout the theater are converted to the QRA role at the COMMZ airbase according to the conversion priority index IPSQRA. The number of aircraft of a given type that are converted from a particular sector airbase is proportional to the number of aircraft of that type stationed at that airbase. The logic structure of this section is similar to that of section 110.

l. Section 230 - Output COMMZ QRA Shortage. This section of QRAFIL records on file JSUM any QRA shortage at the COMMZ airbase that was not eliminated in the three preceding sections. It is the last section in the side Do loop begun in section 100 and it terminates the execution of QRAFIL.

2.2.10.4 ASGATR. Subroutine ASGATR simulates interdiction of SSM sites, reserve divisions, and supply nodes. ASGATR reapportions INT aircraft among the three INT submissions if no targets exist for some missions, or if there are insufficient SSM targets for all the aircraft assigned to SSM interdiction to attack. Attrition is computed for short and medium range SSMs in forward sectors, for personnel and weapons in reserve divisions and for supplies at active battle area supply nodes.

2.2.10.4.1 Programming Specifications. The following table summarizes the principal specifications for subroutine ASGATR:

Characteristic	Specification
Formal parameters	None
Common Blocks	Blank Common
Subroutines called	None
Called by	AIRGRD

2.2.10.4.2 Logic Functions. The logic of subroutine ASGATR is performed once for each side. The code is divided by comment cards that label the functions performed by the logic.

a. Section 50 - Initialize Variables, Set Side and Sector Indices. This section of ASGATR begins the side Do loop that extends over the entire subroutine. It initializes a number of variables and then begins the sector Do loop that extends over the remainder of the subroutine's logic.

b. Section 100 - Reapportion Aircraft Among INT Submissions. This section of ASGATR begins by determining the number of enemy divisions in reserve in the first inactive battle area of the sector (NOD), the total number of SSM sites detected in the sector (T), and the index of the region containing the sector (IR). The reassignment logic is then performed once for each type of attacking aircraft. First, the number of successful sorties (E) on each INT submission is computed. A successful sortie is one on which ordinance could have been delivered. The three INT submissions are interdiction of: (1) SSM sites, (2) reserve divisions, and (3) supply nodes. The number of enemy reserve divisions (NOD), the average number of enemy personnel casualties that were inflicted per friendly CAS sorties (FPDCAS), and the number of SSM sites detected (T) are checked. If no SSM sites have been detected, then aircraft assigned to mission 1 are divided between missions 2 and 3 in proportion to the fractions of INT aircraft already assigned to those missions. If all INT aircraft were originally assigned to mission 1, then half of them are reassigned to each of the other missions. Similarly, aircraft are reassigned to missions 1 and 3 if there are no enemy divisions in reserve or if no casualties were inflicted by CAS aircraft. In the latter case, no casualties would occur from the interdiction of reserve divisions, since the casualty rate per CAS sortie (FPDCAS) is used as the measure of effectiveness for INT sorties. All aircraft are assigned by default to supply interdiction if they cannot be used on either of the first two submissions. The number of aircraft assigned to each mission is stored by aircraft type in the array V. The total number of aircraft assigned to SSM interdiction (SI1) is also computed.

c. Section 200 - Compute SSM Attrition. This section begins by setting the number of SSM sites attacked (TNTA) equals to the minimum of: (1) the number of SSM sites detected and (2) the number of raids that are possible. The number of raids is calculated by dividing SI1 by RSIZE, the minimum raid size for attacking SSM sites. If RSIZE > SI1, then the number of raids is assumed to be 1. If the number of possible raids exceeds the number of SSM sites detected, then all excess aircraft are reassigned from SSM interdiction to the other interdiction missions in proportion to the number of aircraft already assigned to each of those missions. Atrrition to SSM sites is computed using a binomial equation. Then the number of surviving SSM sites is calculated. The section concludes by calculating SI2 and SI3, the total numbers of aircraft assigned to INT missions 2 and 3, respectively. The calculation of SI3 includes the factor ESDASI, which accounts for the effectiveness (in tons per sortie) with which supplies are destroyed by interdiction aircraft.

d. Section 300 - Compute Attrition to Reserve Divisions. If no aircraft have been assigned to INT mission 2, the logic flow branches around this section of ASGATR to section 400. INT attacks in a sector are directed against the most effective division or divisions located within the sector's first inactive battle area. The number of INT sorties allocated to attack a particular division is proportional to the number of people in that division. It is desirable that the attrition to reserve divisions from INT attacks be consistent with the attrition to combat divisions from CAS attacks. Therefore, attrition to personnel and weapons in each reserve division is computed by multiplying the number of INT sorties attacking that division by FPDCAS and FWDCAS, the average attrition to combat personnel and weapons per CAS sortie.

e. Section 400 - Compute Supply Losses. This section of ASGATR computes supply losses using an exponential equation. All supply interdiction missions in a sector are directed against the supply node, identified by ISNABA, that supplies the sector's active battle area.

2.2.11 LINKJ. This subsection describes subroutine PSAIR which is the only routine in LINKJ.

2.2.11.1 PSAIR. Subroutine PSAIR prints summary data tables A-1 through A-8 which summarize air resources (aircraft, SAMs, shelters, etc.) and their impact on other aircraft, SAMs, and shelters. PSAIR is always called on the first and last cycles. It is also called when IPRD (the index for printing detailed results)=1, or when the current combat cycle is equal to the value of IPRSO (the index for printing summary output).

2.2.11.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine PSAIR:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	None
Called by	TMAIN

2.2.11.1.2 Logic Functions. PSAIR uses simple summations, formatted write statements and loops, where appropriate, to generate the following tables:

Tables	Headings
A-1	Blue Aircraft (Undamaged), QRA and Helicopters in Theater at End of Cycle
A-2	Red Aircraft (Undamaged), QRA, and Helicopters in Theater at End of Cycle
A-3	SAMs in Repair and Replacement Pools and Total Missiles in Theater at End of Cycle
A-4	Air Munitions Expended on CAS and INTD, and Aircraft Shelters Operational and Destroyed
A-5	Blue Cumulative Aircraft Killed on Ground

Tables	Headings
A-6	Red Cumulative Aircraft Killed on Ground
A-7	Cumulative Successful Attack Sorties
A-8	SAMs Suppressed During Cycle and Cumulative SAMs Damaged and Repaired

2.2.12 LINKK. This subsection describes the routines in LINKK. These routines comprise the theater control model.

2.2.12.1 TC. Subroutine TC handles a major portion of the bookkeeping effort required by the TACWAR model. Unlike most of the other routines in the model, the theater control routine considers few attrition related calculations. TC considers the following array of items:

- Determines the assignment of reconnaissance aircraft and computes attrition to army-air carriers and reconnaissance aircraft, provided the target acquisition model is played.
- Determines the width of each combat sector (and hence active battle area) based on the current FEBA location.
- Computes the total tonnage of supplies consumed by combat units that are located in region areas and the COMMZ.
- Determines the sectors of main attack (if not input by the user) within each region based on opposing effective combat units.
- As a result of FEBA movement calculated in subroutine FEBAMT, updates the location of combat divisions and supply nodes. Then, based on the new FEBA location, the region forward and region rear depths are adjusted, if necessary. (Recall that forward and rear sectors and regions are tied directly to an integral number of battle areas defined by the user.)
- Computes the number of noncombat weapon losses that arise each cycle and sends the broken weapons to the repair pools. Conducts the operation of weapon repair and sends repaired weapons to the replacement pool.
- Computes change in combat mode, if appropriate.

- Computes a new combat effectiveness of each division based on people, weapons, and supplies available in the units. Withdraws from the active battle areas any ineffective divisions and replaces them with divisions of higher effectiveness, if available.

- Computes for each division in the active battle area its demand for replacements of people, weapons by type, and subunits by type. Assigns to these divisions the required replacements, if available in the replacement pools.

- Replaces divisions in the active battle areas with divisions of higher strength, if available, from the first inactive battle area.

- Reinforces divisions in the first inactive battle area and assigns required replacements, if available, from the replacement pools.

- Through a call to subroutine AIRASG, determines which actual airbases must be abandoned because of advancing enemy forces.

- Computes the total supply demands of the tactical airbases and of the combat units in the rear areas, and ships supplies to requesting areas according to available inventories.

- Moves divisions from rear areas forward by the appropriate movement rate.

2.2.12.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TC:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	AIRASG, CNTRYC, IIBA, NXDIV, SECWTH
Called by	TMAIN

2.2.12.1.2 Logic Functions. The coding of subroutine TC is divided into sections which are headed by comment cards labeling the functions performed by the logic. These functions are explained below.

a. Section 1 - Compute Assignment of Reconnaissance Aircraft. If the target acquisition model is not played, control is transferred to section 5. This section is performed for each side L. First, the array RACAM, the number of reconnaissance aircraft on each type air mission in each sector, is initialized to zero.

The remaining calculations are made for each region IR. Local variable TOT is set to the sum of all reconnaissance aircraft at forward and rear airbases in sectors of the region. The reconnaissance aircraft at the COMMZ airbase are prorated to the region according to the number of sectors in the region, and this proportion is added to TOT. Then TRECA, the total number of reconnaissance aircraft operable, is calculated as TOT * (1- FRACIO), where FRACIO is the fraction of inoperable reconnaissance aircraft. The operable aircraft are distributed, as described below, to each sector for each of three missions: standoff moving, forward area, and deep area.

The number of reconnaissance aircraft on the first mission type is the percent of reconnaissance aircraft on mission type 1 (PAFRAM) times the number of operable reconnaissance aircraft (TRECA). These aircraft are prorated to each sector in the region according to the ratio of the sector width to the region width, and stored in array RACAM.

The number of reconnaissance aircraft on the second mission type is the product of the percent for type 2 and total aircraft. These aircraft are prorated to each sector in the region according to the ratio of the number of enemy divisions in the active battle area in the sector to those in the region.

The number of reconnaissance aircraft on the third mission type is the product of the percent for type 3 and total aircraft. These aircraft are prorated to each sector in the region according to the ratio of the number of divisions in the enemy's first inactive battle area in the sector to those in the region.

b. Section 2 - Compute Attrition and Remaining Inventories to Army-Air Carriers. This section is executed only if the target acquisition model is played. The calculations in this section are performed for each side L, each sector IS, and each type of army-air carrier IACC. For each mission type IM, the local array SUMM is calculated as the product of PAACAM (the percent of army-air carriers on mission type IM), RAACM (the rate of attrition for army-air carriers on mission type IM), and 1 minus FRAACI (the fraction of inoperable army-air carriers). Then the number of army-air carriers of each type in each sector (AACS) is reduced by the value of SUMM for each mission type, to reflect the attrition of army-air carriers.

c. Section 3 - Compute Attrition and Remaining Inventories to Reconnaissance Aircraft. This section, which is executed only if the target acquisition model is played, is performed for each side L. First the local variable TEMP is set to the minimum of 1 and the sortie rate for side L reconnaissance aircraft. Then for each region IR, the following calculations are performed. The local variable TOT is set to the sum of reconnaissance aircraft on forward and rear airbases in the sectors of the region. The reconnaissance aircraft on the COMMZ airbase are prorated to each region according to the ratio of the number of sectors in the region to the total number of sectors in the theater. These aircraft are added to TOT. If TOT does not exceed .001 then no further calculations are made for this region. The total number of aircraft destroyed is the sum, over all mission types IM and sectors IS in the region, of the product of RAAFRM (the rate of attrition), RACAM (the number of reconnaissance aircraft) and TEMP (the sortie rate determined above). The number of reconnaissance aircraft at each forward and rear airbase in the sectors of the region (RACFS and RACRS) is multiplied by 1 minus TEMP2, the percentage of aircraft destroyed in the region, to reflect the loss of reconnaissance aircraft due to attrition. The number of reconnaissance aircraft at the COMMZ airbase which have been assigned to the region (i.e. RACCZ times the ratio of the number of sectors in the region to the number in the theater) is multiplied by TEMP2 to determine the number of COMMZ aircraft lost. This number of aircraft is subtracted from RACCZ, to reflect the number of COMMZ reconnaissance aircraft remaining after attrition in the region. The number of reconnaissance aircraft at any base is not reduced below zero.

d. Section 5 - Calculate Sector Width Based on FEBA Position. For each sector, subroutine SECWTH is called to determine the width of the sector (WIDS) at the FEBA location. Also returned from SECWTH are the longitude at the beginning of the sector segment which contains the FEBA (P2LONG), the length of the sector segment (DISTBP) and the fractional distance along the sector segment length to reach the FEBA (FRACDS). These values are used later in section 23 to update supply nodes. This section also identifies the location of the FEBA by interval within the sector and sets the sector posture and terrain to that of the interval.

e. Section 9 - Compute Supply Consumption in Regions and COMMZ. This section computes the amount of supplies consumed by each division in the regions or COMMZ. Any division that is located in an active battle area which contains no enemy division is also considered to be in the region. For each division in the region or COMMZ, the total weapons' value (WVDDC) is calculated from the number of weapons of each type and the value of that weapon type on defense. The percent personnel strength (PPS) is the ratio of the number of people in the division to the TOE people for that division type. Subroutine CVFW is called to obtain the percent combat effectiveness of the division on defense (PED) with this percent personnel strength. The effective division weapons strength (EDDWS) is the minimum of: (1) the ratio of the total weapon's value to the TOE weapons' value for that division, and (2) PED. The amount of supplies consumed by the division (TEMP) is the minimum of: (1) the effective strength (EDDWS) multiplied by the consumption rate of supplies by the division in reserve (CSDR) and (2) the amount of supplies on hand for the division (SDIV).

Finally, the division supplies (SDIV) and the cumulative supplies consumed by a side (CISCD) are adjusted to reflect the amount of supplies consumed by the division.

f. Section 10 - Compute Sectors of Main Attack. This section first tests the flag ICSMA to determine if the sectors of main attack (ISMA) are to be computed. If ICSMA=1, then ISMA is set equal to the user input values and all other calculations in this section are skipped. Otherwise, for each region, that sector in which the theater attacker has made the greatest advancement is selected as the sector of main attack. This advancement is measured, according to

user selection, from either an initial baseline location (indicated by the flag MFOPT=1) or from the time-zero FEBA location (MFOPT=2). If MFOPT=1, then the sector of main attack for the Blue (Red) theater attacker is the sector whose FEBA location has the smallest (largest) value. If MFOPT=2, then the sector of main attack for the Blue (Red) theater attacker is the sector whose FEBA location is such that the difference between the present FEBA and the time-zero FEBA is the smallest (largest) value.

g. Section 15 - Determine if a Breakthrough Should Be the Posture for the Next Cycle. The calculations in this section are performed for each sector IS. If the posture during this cycle was holding (KPSY=2) but for the next cycle is attacking (KPS=1) then the posture of the new cycle is reset to 5, to indicate a breakthrough. If the posture for this cycle was a breakthrough, then the posture for the next cycle is set equal to the posture in the interval of the sector.

h. Section 20 - Update Division Location Due to FEBA Movement. This section is executed on a sector-by-sector basis. If the distance to the FEBA (measured from a fixed point in the Red COMMZ) is greater than the cumulative ground distance to the leading edge (i.e. the one further from the Red COMMZ) of the active battle area (GDBA), then the Red side is advancing and indexes are set appropriately. If this is not the case, it is determined whether the FEBA is still located in the active battle area or if the Blue side is advancing. If the active battle area is the first or last battle area in the sector or if the FEBA is greater than the ground distance to the leading edge of the next lower battle area, then the FEBA is still in the active battle area and no divisions will have to be relocated for this sector. Otherwise the Blue side is advancing, and the indexes are appropriately set and relocation of divisions begins.

The new active battle area will be the sector next to the old active battle area in the direction of the advancement of the FEBA. The supply node for the old active battle area, which is now an inactive battle area, is that supply node which served it for the advancing side when it was active. All Blue and Red divisions in the old active battle area are relocated to the new active battle area. If there exists a battle area which can qualify as the new first inactive

battle area for the retreating side, then those divisions in the old first inactive battle area are relocated to the new first inactive. If no such battle area exists, these divisions are removed from the theater. Then, the new first and second inactive battle areas for the advancing side are set and divisions moved into these battle areas from the battle areas, if existing, which formerly had their designations.

i. Section 23 - Update Supply Nodes Due to FEBA Advance. This section updates the supply node system to reflect changes resulting from the FEBA advancement. The calculations are made on a sector-by-sector basis. For each of the retreating side's supply nodes, going from the one farthest from that side's COMMZ to that supply node which serves the active battle area, the distance from the supply node to the FEBA is determined. This distance is calculated as follows. The difference in longitude between that of the beginning of the sector segment containing the FEBA and that of the supply node is projected onto the line segment specifying the length of the sector segment. This yields the fractional distance along the sector segment length to reach the supply node. The difference between this fraction and the fractional distance to the FEBA is multiplied by the length of the sector segment to yield the distance between the FEBA and the supply node.

This distance (or its negative for side 2 nodes) is compared to DFASN, which is a user input specifying how close a supply node can be to the FEBA. If the distance is negative, the enemy has overrun the supply node. In this case the supplies at this node are destroyed by their owner, who then abandons the node (i.e. IOSN=0). If the distance is less than DFASN, then the node will also be abandoned but the supplies will be transferred to that adjacent supply node which is closer to the COMMZ of the retreating side.

If the supply node assigned to the previous active battle area had to be abandoned, then another supply node must be assigned. Each remaining supply node belonging to the retreating side, starting with the one farthest from the COMMZ, is tested to see if it is too close to the FEBA (i.e. less than DFASN). Each node too close to the FEBA is abandoned and its supplies moved closer to the COMMZ, but the first one not too close to the FEBA is assigned to the new

active battle area. If there is no supply node in the sector which qualifies to serve the active battle area, then the COMMZ node is assigned. Next, each of the supply nodes which do not belong to either side are tested to see if they are far enough into the territory of the advancing side to be taken over by the advancing units.

j. Section 25 - Change Depth of Regions if Necessary. This section will change the depths of regions if the FEBA advances far enough to cause the collapse of the regions. See subsection 2.1.1.3 for a discussion of the collapsing of regions. The following calculations are made for each sector and each side. The number of battle areas belonging to the side is determined. The active battle area is no-man's land. If the number of battle areas on a side exceeds the sum of the number of battle areas designated to the sector forward, the number to the sector rear, and one battle area for the COMMZ, then the region depths for that side will not be adjusted. If the number of battle areas on the side is insufficient, then the following steps are taken. The number of battle areas in the sector rear is reduced until the deficit is met or there is only one battle area assigned to the sector rear. Then, if the deficit has not been met, the number of battle areas in the sector forward is reduced until the deficit is met or there is only one battle area in the sector forward. If there is still a deficit, then one by one the sector rear, COMMZ and sector forward will lose its one battle area until the deficit is met.

k. Section 30 - Compute Number of Noncombat Weapon Losses and Add to the Repair Pool. The number of noncombat losses is determined for each weapon type in a division as the number of weapons of that type in the division (WDIV) times the fraction of noncombat losses when the division is in tactical situation I (PWNCTS). If the division is in an active battle area, I will be 1 indicating that the division is active, otherwise it will be 2 indicating inactive. All of these weapon losses are assumed to be repairable and are added, by weapon type, country and side, to the repair pool (WDRRP) and subtracted from the number of weapons in the division (WDIV).

l. Section 35 - Compute Number of Weapons Repaired and Sent to the Replacement Pool. The following calculations are made for each side and each weapon type. The number of

weapons repaired is first computed as the fraction of weapons which can be repaired in one cycle and sent to the replacement pool (PWRCRP) times the number of weapons in the repair pool (WDRRP). This value is then adjusted to be no larger than the maximum number of weapons which can be repaired in one cycle (OCWRP). The repaired weapons are then transferred from the repair pool to the replacement pool.

m. Section 40 - Compute Change in Combat Mode if Appropriate. This section is executed only if nuclear or chemical weapons are played. The calculations in this section, which determine appropriate combat mode changes, are performed for each side L and each sector IS. The combat modes are nonnuclear and nonchemical (=1), nuclear or chemical prepared (=2), or preemptive strike (=3).

The index JE is set to the maximum of the chemical employment levels and nuclear escalation states for this cycle and the proposed ones for the next cycle for target type 1 (divisions in the active battle area). If the value of JE is zero, no further calculations are made for this sector. Local variable ICM is set to ICMES (JE,L), the combat mode when side L is in state JE. ICM is the desired combat mode. The remainder of the coding in this section uses ICM to determine a new value for the present combat mode ICMS (IS,L) and to set variable ICMST (IS,L) for the next cycle.

The array ICMST indicates whether the combat mode is to be downgraded this cycle due to calculations made during the last cycle. If ICMST=0, no downgrading is to occur. If ICMST=1 or 2, then the combat mode is to be downgraded to that level (i.e. combat modes 1 or 2) during this cycle. The program tests the value of ICMST before changing the present combat mode ICMS, and follows the procedure described below.

If no downgrading is to occur this cycle (ICMST=0), then the present combat mode ICMS is compared to the desired combat mode ICM and one of the following adjustments is made. If ICMS=ICM, then ICMS and ICMST are ungchanged. If ICM>ICMS, then ICMS is upgraded by one level and ICMST is unchanged. If ICM<ICMS, then ICMS is unchanged this cycle but ICMST is set to one level below ICMS so that ICMS will be downgraded one level during the next cycle.

If downgrading is to occur this cycle (ICMST=1 or 2), one of the following adjustments is made. If ICMST is less than the desired mode ICM, no downgrading will occur this cycle and ICMST is reset to 0. If ICM≤ICMST, then ICM<ICMS since ICMST, in this case, is one level lower than ICMS. Under these conditions, ICMS is downgraded to ICMST and ICMST is reset to 0, indicating no downgrading during the next cycle.

n. Section 45 - Compute Effectiveness of All Divisions. This section computes the effectiveness of all divisions on attack (EFFDA) and defense (EFFDD). It first calculates the total weapons' value of a division on attack (defense) from the number of weapons of each type and the value of that weapon type on attack (defense). The percent personnel strength (PPS) is set equal to the ratio of the number of people in the division and the TOE people for that division type. Subroutine CVFW is called to obtain the percent combat effectiveness of the division on attack (defense) with this percent personnel strength. The number of days of supplies on hand (DSH) is computed from the amount of supplies for the division and the planned consumption rate for that type division. Subroutine CVFW is called to determine the supply effectiveness factor (SEF) as a function of the number of days of supplies on hand. The effectiveness of the division on attack (defense) is the minimum of: (1) the ratio of the total weapons value to the TOE weapons value for attack (defense), and (2) the percent combat effectiveness, all adjusted by the supply effectiveness factor. The weapons' value, adjusted by the supply effectiveness factor, for a division on defense (VDDSF) is the product of the effectiveness factor (EFFDD) and the total TOE weapons' value for the division on defense (WVDDTS).

o. Section 50 - Order Divisions in the First Inactive Battle Area by Effectiveness and Combat Value. This section is executed on a sector-by-sector basis for each side. The divisions in the first inactive battle area are divided into two effectiveness groups: IWORK1 for divisions whose effectiveness is greater than or equal to ERDWLE, the effectiveness at which a reinforcing division will replace a withdrawn division of lower effectiveness; and IWORK2 for divisions whose effectiveness is less than ERDWLE. Each of these groups is then ordered by ascending combat value (VDDSF). The ordered division indexes are stored in array IDLIBA with IWORK1 divisions followed by IOWRK2 divisions, which yields the correct ordering of all divisions in the first inactive battle area.

p. Section 55 - Withdraw all Ineffective Divisions and Reinforce with Divisions of Highest Effectiveness. This section is computed on a sector-by-sector basis. The initial step is to load the values of the Blue/Red array IDLABA, which identifies the divisions in the active battle area, into the local array IWORK, which is indexed by side. For each side L, the ineffective divisions in the active battle area are withdrawn and the active battle area is reinforced with divisions of highest effectiveness from the first inactive battle area, as described below.

If the active battle area is the first (last) battle area in the sector, then no Red (Blue) divisions are moved. If side L is the sector attacker, each side L division in the active battle area whose effectiveness on attack (EFFDA) is less than or equal to zero is moved to the second inactive battle area. If the second inactive battle area as usually defined, does not exist then Red (Blue) divisions are moved to the first (last) battle area in the sector. The identity of each effective division is stored in local array IWORK. If side L is not the sector attacker, the calculations are similar but the value of effectiveness on defense (EFFDD) is the criterion for withdrawing a division. Next, the total division width (TDW) for the active battle area is calculated from the width of each division by its type for the combat mode (nonnuclear and nonchemical, nuclear or chemical prepared, nuclear or chemical) and tactical role (attacker or defender) for that side and sector. Then TDW is contracted, using the factor of minimal contraction of a division width (FMNC). Divisions are then moved into the active battle area from the first inactive battle area, starting with the most effective division whose contracted division width will not increase total division width (TDW) beyond the width of the sector (WIDS). Reinforcements are sent up until all effective divisions have advanced or until there is no division whose width will fit into the remaining space in the active battle area.

As divisions are moved out of the active battle area and other divisions moved into it, the array AACDS, the number of army-air carriers in the active battle area of the sector, is updated. Army-air carriers, which are associated with individual divisions, are used only by divisions in the active battle area. Before any movement of divisions by this section of code, local array STOR1 is set to the current

value of AACDS and local array STOR2 is set to the sum, over all divisions in the active battle area, of the TOE number of carriers for the division type. If a division is removed from the active battle area, then AACDS is reduced by the percent determined by the ratio of the TOE number of army-air carriers for the division and the value STOR2. If a division is moved into the active battle area, it is assumed to have a full capability of carriers and AACDS is increased by the TOE number of army-air carriers for the type division.

After ineffective divisions have been withdrawn and reinforcements sent up for both sides, the sum of the number of Blue and Red divisions in the active battle area is compared against MDDABA, the maximum number of divisions allowed in the active battle area. If there are too many divisions, the program terminates and prints out "STOP 1111". Otherwise, the identity of the divisions in the active battle area are loaded into array IDLABA with Blue divisions followed by Red divisions.

q. Section 60 - Compute Demand for Replacement People and Weapons in the Active Battle Areas. Send Up Replacements if Sufficient Number in Pool. The following calculations are made for each side L. The demand over all sectors for replacement people in the active battle area (RPNABA) is the total number of people needed to bring each division in the active battle area of any sector up to the TOE level for that division type. The demand for replacement weapons (RWNABA) is similarly determined for each weapon type. If there are enough people in the replacement pool (RPCZ) and enough weapons of each type in the replacement pool (WRPCS) to meet the demands of all divisions, then replacements are distributed to each division according to its needs and the number of subunits in each division is upgraded to the TOE level. If any pool is insufficient, the flag IRDABA is set to 1 and no replacements are sent up for that side.

r. Section 65 - Replace With Divisions From the Inactive Battle Area if of Higher Strength. This section is computed on a sector-by-sector basis. The initial step is to load the values of the Blue/Red array IDLABA, which identifies the divisions in the active battle area, into the local array IWORK, which is indexed by side. Then for each side, it is determined which divisions, if any, in the active battle area will be replaced. Under the following conditions there will

be no replacements: (1) the flag IRDABA is zero indicating that all divisions in the active battle area are at full strength, (2) there are no divisions in the first inactive battle area, (3) no division in the first inactive battle area has an effectiveness level of at least ERDWLE, the minimum combat effectiveness of a division before it replaces a division which is withdrawn, (4) the active battle area is the first (last) in the sector and Red (Blue) divisions are being considered, i.e. there is no separate first inactive battle area.

If none of the preceding conditions exist, then the program searches for replacement candidates. First the indexes of side L divisions in the active battle area are loaded into array IWORK according to ascending combat values on defense (VDDSF). The tactical role played by side L in the sector is attacker (ITR=1) if it is the sector attacker or if there is no sector attacker and L is the theater attacker. Otherwise the tactical role is defender (ITR=2). Next, the total division width (TDW) for the active battle area is calculated from the width of each division, which depends on its type and on the combat mode and tactical role for side L and the current sector. Then TDW is contracted, using the factor of minimal contraction of a division width (FMNC). The available width of the sector (AWS) is then the difference between the sector width (WIDS) and TDW.

A division in the active battle area may be replaced if it has an effectiveness level below EWDRHE, the maximum combat effectiveness of a division before it is withdrawn from combat and replaced by a division of higher effectiveness. For each division ID which can be withdrawn, the divisions in the first inactive battle area are searched to find one which meets the following qualifications: (1) the division must have an effectiveness level above ERDWLE, (2) the fractional increase in combat value resulting from replacing division ID with this division must be greater than or equal to the required fractional increase (PICVDR), (3) the total division width resulting from making this replacement must not exceed the width of the sector (WIDS). Condition (2) above need not be met if the combat value of division ID is less than or equal to .0001. In this case, the replacement division need only meet requirements (1) and (3). If a replacement division is found, then division ID is moved to the second inactive battle area, as determined by function

IIBA, and the replacement division moved from the first inactive battle area to the active battle area. When a division is withdrawn, a zero replaces that division index in array IWORK and the index of the replacement division is entered in the first available location in IWORK, i.e. that one not previously occupied by a division index. When all replacements have been made in the active battle area, the division indexes are reset in IDLABA from the relative (i.e. zeros are ignored) location of division indexes in IWORK.

As divisions are moved out of the active battle area and other divisions moved into it, the array AACDS, the number of army-air carriers in the active battle area of the sector, is updated. Army-air carriers, which are associated with individual divisions, are used only by divisions in the active battle area. Before any movement of divisions by this section of code, local array STOR1 is set to the current value of AACDS and local array STOR2 is set to the sum, over all divisions in the active battle area, of the TOE number of carriers for the division type. If a division is removed from the active battle area, then AACDS is reduced by the percent determined by the ratio of the TOE number of army-air carriers for the division and the value STOR2. If a division is moved into the active battle area, it is assumed to have a full capability of carriers and AACDS is increased by the TOE number of army-air carriers for the type division.

s. Section 70 - Compute Demand and Replacements in the Active Battle Area. The following calculations are made for each side, provided that side's divisions in the active battle area are not all at full strength. The demand over all sectors for replacement people in the active battle area (RPNABA) is the total number of people needed to bring each division in the active battle area of any sector up to the TOE level for that type division. The demand for replacement weapons (RWNABA) is similarly determined for each weapon type. If the personnel replacement pool (RPCZ) can meet the total personnel demand, then for each division the array PNDABA will be set to the number of people needed by that division. If the pool cannot meet the demands, then PNDABA will be set to that number which represents the portion of demand which can be met by the replacement pools. The array WNDABA is similarly determined for each weapon type of a division. These two arrays, which represent the amount of replacements available for a division, will be used in the next section.

t. Section 75 - Compute Number of Subunits Currently in Divisions and Add People and Weapons to Divisions From the Replacement Pool. This section is calculated on a sector-by-sector basis with an inner Do loop on side. The following calculations are performed for each division in the active battle area. If the division does not contain the TOE level of subunits, replacements will first be used to comprise as many as possible of the needed subunits and then existing subunits will be upgraded from the remaining replacements. The process is described below.

For each subunit type, the number of subunits needed to reach the TOE level (SUNDIV) is calculated. Also, the total number of people (TPNDIV) and weapons (TWNDIV) needed in the division for all of these subunits are determined. Then for each division type, the program determines how many of the needed subunits can be constructed with the replacement people available (PNSU). If all needed subunits cannot be constructed, then PNSU is set to that proportion of subunits needed which is equal to the ratio of total replacement people available to the total people needed in the division for subunits. The value WNSU is similarly determined for the primary weapon type used in the subunit type. The minimum of PNSU and WNSU determines the number of subunits which are added to the division. The replacement pools and the personnel and weapons at the division are adjusted to reflect the shift of personnel and primary weapons to the division for the new subunits.

After all new subunits have been formed, the number of replacement people (PNDABA) and weapons (WNDABA) needed are adjusted to reflect the personnel and weapons added via the new subunits. Next, the TOE personnel and weapons which would have been needed by the division for those subunits which could not be formed are calculated. These values are used to determine the replacements which are needed by existing subunits. The replacement pools and the personnel and weapons at the division are adjusted to reflect the shift of personnel and all weapon types to upgrade the existing subunits.

If a division contains the TOE level of subunits, then all available replacement personnel and weapons, as determined by arrays PNDABA and WNDABA, are distributed to the division from the replacement pools.

u. Section 80 - Reinforce Divisions in the First Inactive Battle Area. The calculations described below are made for each side L on a sector-by-sector basis. The number of divisions which can fit into the first inactive battle area is determined from the width and length of the battle area and the width (WIDDR) and depth (DEPDR) of a side L division in reserve. The width of the battle area is the average of the widths at the near and far edges, and the length is the difference in ground distances to the near and far edges. The array IDLIBA, which identifies the divisions in the first inactive battle area, is adjusted by removing all zero values and shifting the remaining division indexes to close the gaps. Then NDLIBA is reset to the actual number of divisions in the first inactive battle area.

If the number of divisions in the first inactive battle area is the maximum which can be contained there, no further calculations are necessary. If the number of divisions exceeds the maximum, then divisions are withdrawn, starting with the least effective one, until only the maximum allowed remain. Withdrawn divisions are moved to the second inactive battle area if it exists. If not, the divisions are removed from the theater.

If the first inactive battle area is not full and there exists a nonempty second inactive battle area, divisions will be moved from the second to the first inactive battle area. The indexes of the divisions in the second inactive battle area are loaded into the local array IWORK1, whose values are then reordered according to descending effectiveness of divisions on defense and stored in local array IWORK2. Finally, divisions with effectivenesss levels higher than the minimum needed for replacement divisions (ERDWRE) are moved, starting with the most effective one, into the first inactive battle area until the area is full or there are no more reinforcement divisions.

v. Section 85 - Compute Replacements for Divisions in the First Inactive Battle Area. First, the number of replacement people (PNDABA) and the number of replacement weapons by type (WNDABA) needed in the first inactive battle area are determined for each division from the TOE and actual levels of people and weapons. If the replacement pools cannot meet the total, inventories are prorated according to the needs of the divisions. The arrays PNDABA and WNDABA are adjusted

to contain the replacements needed and available. If all replacement pools have sufficient inventories, as indicated by the flag II being zero, then all divisions are replenished from the pools to full strength for personnel, weapons and subunits.

If replacement pool inventories are insufficient, then new subunits will be created first and then replacements will be distributed to existing subunits. The total number of people (TPNDIV) and weapons (TWNDIV) needed in the decision for all of these subunits are also determined. Then, for each division type, the program determines how many of the needed subunits can be constructed with the replacement people available (PNSU). If all needed subunits cannot be constructed, then PNSU is set to that proportion of subunits needed which is equal to the ratio of total replacement people available to the total people needed in the division subunits. The value WNSU is similarly determined for the primary weapon type used in the subunit type. The minimum of PNSU and WNSU determines the number of subunits which are added to the division. The replacement pools and the personnel and weapons at the divisions are adjusted to reflect the shift of personnel and all weapon types to upgrade the existing subunits.

w. Section 86 - Update Notional Airbases Based on FEBA Change. This section calls subroutine AIRASG to update the notional airbases. The change in FEBA location requires a reassignment of aircraft to the sector and COMMZ airbases.

x. Section 87 - Calculate Supplies Demanded by Divisions and Airbases and Ship Supplies According to Inventories and Demand. The first part of this section adjusts the inventory at supply nodes to reflect the arrival of side L supplies during the present cycle. If a supply node has the same owner as it had during the last execution of the supply model, then supplies arriving for the node (SUPASN) will be added to its inventory. The value stored in SUPASN is negative and, therefore, the negative of SUPASN is added to the inventory. If a supply node S is sector IS does not have the same owner, then the supplies arriving for that node will be redirected to that supply node serving the active battle area in sector IS for the side which was the owner of supply node S during the execution of the supply model.

The routine next determines the demand (SDSN) at each supply node for supplies to be sent to divisions and airbases. The demand for supplies for a division is that amount needed, and not already at the division, to maintain DSD days of supplies on hand assuming a consumption rate of PCSD. The demand by each division is calculated and added to the demand at the supply node assigned to the battle area in which the division is located. Similarly, for each side the demand at each forward and rear airbase of each sector and at the COMMZ airbase is calculated and added to the demand at the supply node assigned to the airbase. Forward (rear) airbases are served by that supply node which serves the last battle area in the sector forward (rear). The Red (Blue) COMMZ airbase is served by the first (last) supply node in the theater.

If the inventory at a supply node is sufficient, the total demand at divisions and airbases served by the supply node will be met. If the inventory is insufficient, then the inventory will be prorated to the divisions and airbases according to demand.

y. Section 90 - Compute Effectiveness of Divisions in First Inactive Battle Area Accordingly. This section first calculates the total weapons' value of a division on attack (defense) from the number of weapons of each type on attack (defense). The percent personnel strength (PPS) is set equal to the ratio of the number of people in the division and the TOE people for that division type. Subroutine CVFW is called to obtain the percent combat effectiveness of the division on attack (defense) with this percent personnel strength. The number of days of supplies on hand (DSH) is computed from the amount of supplies for the division and the planned consumption rate for that type division. Subroutine CVFW is called to determine the supply effectiveness factor (SEF) as a function of the number of days of supplies on hand. The effectivenesss of the division on attack (defense) is the minimum of: (1) the ratio of the total weapons' value to the TOE weapons' value for attack (defense), and (2) the percent combat effectiveness, all adjusted by the supply effectiveness factor. Next, the divisions in the first inactive battle area are stored in local array IWORK1 according to descending effectiveness of the divisions on defense. This new ordering is then transferred to IDLIBA, which contains the indexes of the divisions in the first inactive battle area.

z. Section 95 - Move Divisions in the Rear According to IMUTMF(L) and NCOBAM(L). If the flag IMUTMF is 1 for side L, then side L divisions in the rear will be advanced at most NCOBAM battle areas during the present cycle. If IMUTMF is 2, side L divisions will be moved at most one battle area if the present cycle index is a multiple of NCOBAM, which in this case is the number of cycles it takes a division in the rear to move one battle area. After testing the value of the flag IMUTMF, each side L division in the rear, i.e. located behind the second inactive battle area, is moved forward the maximum number of battle areas allowed but no further forward than the second inactive battle area.

aa. Section 98 - Update Node Position for Supply Model. This section is executed only if the present cycle is one in which the supply model is executed. For each sector IS and each side L, the supply node assigned to serve the active battle area for side L is reset to that side L supply node closest to the FEBA in sector IS. If the supply node previously assigned to the first inactive battle area was also previously assigned to the active battle area for side L, then that supply node closest to the FEBA but not located in the active battle area is now assigned to the first inactive battle area.

2.2.12.2 IIBA. Function IIBA determines, from the index of an active battle area, the index of the second inactive battle area for a given side L.

2.2.12.2.1 Programming Specifications. The following table summarizes the principal specifications of function IIBA:

Characteristic	Specification
Formal parameters	I = Sign indicating direction from the active battle area to a side's territory. Value of +1 for Blue and -1 for Red.
	IABA = Index of active battle area
	NS = Number of sectors played

Characteristic	Specification
	NBA = Number of battle areas in the theater
	L = Side index
Common blocks	None
Subroutines called	None
Called by	TC

2.2.12.2.2 Logic Functions. The functional value IIBA is set to the second inactive battle area counting from the active battle area toward the side L COMMZ. If no such battle area exists, then IIBA is set to the first battle area of the sector for the Red side or the last battle area for the Blue side. However, if the active battle area is the first (last) battle area in the sector and the side is Red (Blue) then IIBA is set to -1, a flag indicating this situation.

2.2.12.3 NXDIV. Function NXDIV determines the first nonzero value which occurs after a given array index in an integer array of division indexes.

2.2.12.3.1 Programming Specifications. The following table summarizes the principal specifications of function NXDIV:

Characteristic	Specification
Formal parameters	NI = Number of items in array IDIVL
	IDIVL = Array of division indexes
	IKT = Input as index of that item of IDIVL after which the search begins, output as index of last item tested.

Characteristic	Specification
Common blocks	None
Subroutines called	None
Called by	TC

2.2.12.3.2 Logic Functions. Starting with the IKT+1 item of array IDIVL, this function searches for the first nonzero value in IDIVL. If such a value is found, it is returned as NXDIV. If the search of the array is nonproductive , a value of -1 is returned in NXDIV as a flag to indicate the situation.

2.2.12.4 AIRASG. This subroutine creates notional airbases from actual airbases and reassigns aircraft from COMMZ and rear airbases to forward airbases according to need.

2.2.12.4.1 Programming Specifications. The following table summarizes the principal specifications of subroutine AIRASG:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	None
Called by	TC

2.2.12.4.2 Logic Functions. The coding of subroutine AIRASG is divided into sections which are headed by comment cards labeling the functions performed by the logic.

a. Section 10 - Create Notionalized Airbases in Sector Forward, Sector Rear and COMMZ. This section uses actual airbases, which are located by battle area, to create notional airbases in the sector forward, sector rear and COMMZ. First subroutine TAG is called to determine the status (ISTAT) of battle areas based on the new FEBA location. Each airbase not located in the active battle area is assigned to a notional airbase covering the battle area in which the actual airbase is located. The aircraft associated with the

actual airbase are assigned to the notional base with the one exception that no aircraft are assigned for those bases which were in the active battle area last cycle. Aircraft from a side L airbase located in a battle area which is the active one this cycle but was not last cycle are placed in a pool for later distribution to side L notional airbases. The owner of an airbase is that side whose territory contained the airbase during the last cycle. If the airbase was in the active battle area during the last cycle but the active battle area has changed, the owner of the airbase is that side whose territory currently contains the airbase. Otherwise, the airbase has no owner and does not enter into the calculations.

The military personnel associated with the actual airbase are assigned to the notional airbase of which the actual base becomes a part. This section also determines the maximum number of aircraft which a notional airbase may have, based on the number of actual airbases which comprise the notional base and the maximum number of aircraft allowed on a rear or forward airbase.

b. Section 20 - Reallocate Active Battle Area Aircraft to Notional Airbases. This section first computes the total number of aircraft at each of the sector forward and sector rear notional airbases. The side L aircraft in the active battle area can be divided into three groups according to the range of that aircraft type if it is located on a forward airbase. The range IRNG is 1 if the active battle area is the destination, 2 for the enemy forward airbase and 3 for the enemy rear airbase. Then each group of aircraft, starting with the shortest range group, are distributed to side L notional airbases according to the following procedure. For each sector, those aircraft in the sector's active battle area with the appropriate range are identified and distributed to the sector forward airbase according to the number of this group of aircraft to be reassigned in this sector and the available space at the airbase. If there is enough space, all aircraft are assigned. If not, each type of aircraft is assigned in proportion to the percentage of total available aircraft for which there is space. The military personnel are also assigned based on the average number of personnel per aircraft. Aircraft not assigned are placed in region pools for later distribution.

When each sector forward airbase has been reinforced with aircraft from that sector's active battle area, aircraft will be distributed from the region pools. First aircraft from the region pool are assigned to the forward airbases in that region in a manner similar to the last distribution. Any remaining aircraft are similarly assigned to the rear airbases in the region. Then all remaining aircraft are assigned to the COMMZ notional airbase.

c. Section 30 - Assign COMMZ Aircraft to Forward Airbases. The calculations in this section are performed for each side L and for each of the following two ranges:

(1) from the forward base to the active battle area, and

(2) from the forward base to the enemy's forward base.

For each sector whose former active battle area is not in the side's forward region, the maximum number of aircraft allowed at the forward airbase (ZMXNAB) is adjusted by the factor FWDMAX, whose purpose is to allow the user to modify the number of aircraft that will be sent to forward airbases from rear or COMMZ airbases. From this adjusted maximum number of aircraft and the actual number of aircraft at the base, the space available for aircraft from the COMMZ airbase is determined. The COMMZ aircraft with the appropriate range are identified and distributed to each of the forward bases according to the total number of this group of aircraft in the COMMZ and the available space at the forward airbases. If there is enough space, all aircraft of the group will be moved forward and prorated to each sector according to the space at the base. If not, an amount of each type of aircraft will be selected in proportion to the percentage of total available aircraft for which there is space, and prorated to each sector according to the space available at its forward airbase.

d. Section 40 - Assign Aircraft From Sector Rear Airbases to Sector Forward Airbases by Region. This section is computed for each side L. If the space available for aircraft at forward bases in the sectors whose previous active battle area is now in side L territory is at least .1, then airbases by region. In computing the space available at an airbase, the maximum number of aircraft allowed at the base is multiplied by the factor FWDMAX, which allows the user to modify the number of aircraft that will be sent to forward airbases.

The following calculations are performed for the first two ranges. For each region, the aircraft in the rear airbases which have the appropriate range are identified and distributed to each of the forward bases in the region according to the total number of this group of aircraft in the rear region and the available space at the forward airbases. If there is enough space, all aircraft of the group will be moved forward and prorated to each sector in the region according to the space at its bases. If not, an amount of each type of aircraft will be selected in proportion to the percentage of total available aircraft for which there is space, and prorated to each sector according to the space available at its forward airbase. The amount of each type taken from each rear base is also in proportion to the percentage of total available aircraft for which there is space.

e. Section 50 - Assign Aircraft From Sector Rear Airbases to Sector Forward Airbases on a Theater-Wide Basis. This section is computed for each side L. Aircraft will be assigned from the sector rear bases to the sector forward bases on a theater-wide basis only if the user so chooses, as indicated by the flag LXOPT being set to 1. First the space available at forward airbases in sectors whose previous active battle area is now in side L territory is determined as in section 40. For each of the first two ranges the following calculations will be performed. If there is space for at least 1 aircraft, reassignments will occur. The aircraft in rear bases, which have the appropriate range, are identified and distributed to each forward airbase whose sector's previous active battle area is now in side L territory. This distribution is made according to the total number of aircraft in the group and the space available at the selected forward airbases. The procedure for reassignment is similar to that in section 40. The space available at these selected forward bases is then recomputed before reassigning another range group of aircraft.

2.2.13 LINKL. This subsection describes the routines in LINKL. These routines comprise the supplies model. See subsection 2.1.2 for a discussion of the supplies transportation network.

2.2.13.1 SUPPLY. Subroutine SUPPLY is called by TMAIN every major resupply cycle, which is an integral number of minor resupply cycles. A minor resupply cycle is equivalent to a combat cycle. SUPPLY calculates surpluses and deficits for each supply node. Subroutine TRANPO is called to determine optimal routing of supplies from surplus nodes to deficit nodes. Then SUPPLY ships supplies according to that routing and determines arrival times.

2.2.13.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine SUPPLY:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, STI, PR
Subroutines called	TRANPO
Called by	TMAIN

2.2.13.1.2 Logic Functions. The subroutine code is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Calculate Demand. This section calculates the demand (SDSN) at each supply node as the difference between the amount of supplies needed by divisions and aircraft served by that node and the supply inventory (SUPIN) at that node. To begin this calculation, the demand (SDSN) is set to the negative of the inventory (SUPIN).

Calculations are made by side index but are described below for only one side. For each division in the theater, the supply nodes serving that division are identified and the demand at those nodes increased by the amount of supplies needed by the division. Each battle area has assigned to it a supply node which maintains a level of supplies for each division in that battle area, and which supplies each of these divisions according to its need. In addition, for each

sector there are three supply nodes designated as stockage points for certain battle areas in the sector. There is one which serves any battle area forward of the sector forward, one which serves any battle area forward of the sector rear, and one which serves any battle area forward of the COMMZ. For the supply node serving a division and for each stockage supply node serving that division, the demand is increased by the product of the number of days of supplies to be kept on hand at the node and the planned daily consumption rate of supplies by that division. In addition, the demand at the supply node serving the division will be increased by the amount of additional supplies which should be sent to the division during the next major cycle. The demand of a division is calculated as the product of the planned consumption rate, and the number of days of supplies needed both for inventory at the division and for consumption over the next major cycle. The amount by which this demand exceeds the amount of actual supplies in the division, determines the amount of additional supplies which should be sent to the division from the supply node. If supplies at the division exceed demand, then no additional supplies will be requested.

Next, SUPPLY determines the demand for supplies by aircraft. In each sector there is one notional airbase in the sector forward and one in the sector rear. Each airbase has assigned to it a supply node which maintains a level of supplies for its aircraft and which supplies each airbase according to its needs. The supply node serving the sector forward (rear) airbase is that one which serves the last battle area in the sector forward (rear). In addition, for each sector there are two supply nodes designated as stockage points for aircraft in certain locations. There is one stockage supply node (which also serves the first battle area in the sector rear) for the sector forward airbase, and one (which also serves the first battle area of the COMMZ) for both the sector forward and sector rear airbases. The calculation of the demand by aircraft at an airbase is described below.

First the total number of aircraft on the airbase is determined. Then, the number of additional supplies which will be needed over the next major cycle is calculated as the amount by which demand by aircraft exceeds supplies already on hand at the base. Demand is the product of the planned consumption rate per aircraft, the number of aircraft on the base, and the number of days of supplies for both inventory and consumption over the next major cycle.

Next the demand at the supply node serving the airbase is increased by the additional supplies needed at the airbase and by the amount of supplies needed to maintain at the node the desired inventory level for the aircraft. In addition, the demand at each stockage point node is increased by the amount of supplies needed to maintain the desired inventory level for the aircraft at that node.

In addition to the airbases in the sectors, there is one airbase in the side's COMMZ to which a supply node is assigned. The first supply node is assigned to the Red COMMZ and the last to the Blue COMMZ. The demand at this supply node is increased by the additional supplies needed at the airbase and the amount of supplies required to maintain the desired inventory level for the aircraft at this node.

b. Section 20 - Determine Shipment of Supplies. This section determines the supply deficit and surplus at each supply node, the optimal routing of supplies to meet deficits, and the arrival times of these supplies. The procedure for determining these values, which is described below for the Blue side only, is applied to both sides. The surplus and deficit at each supply node are determined as follows. For nodes not belonging to Blue, both values are zero. For each Blue supply node, if the demand calculated in section 10 is positive then the node has a deficit equal to the demand and a zero surplus. If the demand is negative, then the node has a surplus equal to the negative of the demand and a zero deficit. Total surplus (TSUR) and total deficit (TDEF) are the sums over all supply nodes. The supply flow (SLOW), the total amount of goods which will be shipped between nodes, is set to the minimum of total deficit and total surplus.

The routing of supplies between nodes is actually a transshipment problem. However, it can be converted to a transportation problem (see reference 13 pp. 176-183) which can be solved by TRANPO and its routines. To make the conversion, the supply flow is used as the buffer stock which is added to the surplus and deficit at each Blue supply node thereby creating what will be referred to as adjusted surplus and adjusted deficit. The transportation problem then is to minimize the total ton-kilometers needed to ship supplies from surplus nodes (warehouses) to deficit nodes (markets), subject to the constraints that the total amount of supplies

shipped from any node equals the surplus at that node, and the total amount shipped to any node equals the deficit at that node (all shipments being non-negative).

Before TRANPO is called, the supplies assigned to each Blue node (SUPASN) is set to the difference between the adjusted surplus and the adjusted deficit at that node. This is equivalent to the unadjusted surplus if the node had a surplus, and to the negative of the unadjusted deficit if it had a deficit. After TRANPO is executed, SUPASN will contain the assignment of supplies to the Blue supply nodes according to the optimal routing, and the cost C will be the ton-kilometers expended in shipping these supplies. Those Blue supply nodes with surpluses, i.e., positive supply assignments, immediately relinquish those surplus supplies from their inventories. Then the cost C is used to determine the number of minor cycles it would take to ship a total of SFLOW supplies according to the optimal routing plan at the average rate of ARSS kilometers per hour. (The units of ARSS are converted from kilometers per day to kilometers per hour in subroutine TCTZ.) Supplies must always arrive after the present minor cycle but before the next major cycle.

2.2.13.2 TRANPO. Subroutine TRANPO is the driver for the set of routines which find an optimal routing of supplies from warehouses to markets. This set of routines was developed by Srinivasan and Thompson at Carnegie-Mellon University (reference 1) but was recently made available at the National Bureau of Standards. In the TACWAR model, all supply nodes are considered to be both warehouses and markets.

The transportation problem can be viewed as:

$$\text{Minimize} \quad \sum_j \sum_i C_{ij} \; X_{ij}$$

subject to constraints that

$$\sum_j X_{ij} = a_i$$

$$\sum_i X_{ij} = b_j$$

and each $X_{ij} \geq 0$

where

X_{ij} = Amount of goods shipped from warehouse i to market j

C_{ij} = Cost to ship a unit of goods from i to j

a_i = Amount of goods on hand at warehouse i

b_j = Demand at market j

Figure 7 shows a sample transportation model for three warehouses and four markets. The unit shipping costs (C_{ij}) are shown in the upper left hand corner of each box. The last column (under TOTAL) contains the amount of goods at each warehouse (a_i) and the last row (under TOTAL) contains the demand at each market (b_j). One possible basic solution of this transportation problem is shown by the values for the X_{ij}'s.

Listed below are the definitions for three major local arrays used by the subroutine TRANPO and the routines it calls.

In array ML, for each basic element (I,J) the following columns of data are retained:

1 = Row (or warehouse) index (I) of basis element

2 = Column (or market) index (J) of basis element

3 = Amount of goods shipped from warehouse I to market J

4 = Next basis element, after (I,J) in row I

5 = Next basis element, after (I,J) in column J

6 = A flag used to keep track of completed calculations.

In array MI, for each warehouse I, the following columns of data are retained:

1 = Index of the first basis element (I,J) in row I of the transportation matrix

WAREHOUSE	MARKET				
	M_1	M_2	M_3	M_4	TOTAL
W_1	25 X_{11}	10 $X_{12} = (2)$	2 $X_{13} = (8)$	30 X_{14}	10
W_2	5 $X_{21} = (5)$	5 $X_{22} = (10)$	20 X_{23}	10 X_{24}	15
W_3	100 X_{31}	65 X_{32}	0 $X_{33} = (5)$	2 $X_{34} = (15)$	20
TOTAL	5	12	13	15	45

Figure 7. Sample Transportation Matrix

2 = Associated column index J

3 = Signed shipping costs associated with stepping-stone path from row I to row 1

4 = Column index J1 such that (I,J1) is the first basis element encountered in row I along the stepping-stone path from row 1 to row I

5 = Number of stepping-stones to reach element (I,J1) from row 1

In array MJ, for each market J, the following columns of data are retained:

1 = Index of the first basis element (I,J) in column J of the transportation matrix

2 = Associated row index I

3 = Signed shipping costs associated with stepping-stone path from column J to row 1

4 = Row index I1 such that (I1,J) is the first basis element encountered in column J along the stepping-stone path from row 1 to column J

5 = Number of stepping-stones to reach element (I1,J) from row 1

2.2.13.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TRANPO:

Characteristic	Specification
Formal parameters	L1 = Number of warehouse
	L2 = Number of markets
	TDEF = Total deficit of supplies
	TSUR = Total surplus of supplies

Characteristic	Specification
	SUPASN = Supplies arriving at each supply node
	C = Cost in ton-kilometers for optimal solution
Common blocks	TRANS, MATRIX
Subroutines called	INPUT, INSOL, LABEL1, LABEL2, MAIN, CYCLE, FIXLIJ, IJFIX, OUTPUT
Called by	SUPPLY

2.2.13.2.2 Logic Functions. Subroutine TRANPO calls INPUT to initialize the supply node data. The initial basic feasible solution is found in INSOL by using a variant of the row-minimum rule. LABEL1 sets up the labels corresponding to the list of initial basic cells. LABEL2 determines the values of the dual variables corresponding to the initial basis.

Subroutine TRANPO then iterates, making a number of adjacent basis changes until an optimal basis is found. For each iteration this subroutine calls MAIN, CYCLE, FIXLIJ and IJFIX. MAIN finds a nonbasic cell to pivot on, using the row-minimum rule. Using a modification of the predecessor-index method, CYCLE finds the loop created by the introduction of the nonbasic cell. Subroutine FIXLIJ modifies the labels to correspond to the new basis and IJFIX modifies the dual variables accordingly. When an optimal routing has been found, OUTPUT adjusts the shipping assignments, and if requested, prints out the optimal primal solution.

2.2.13.3 INPUT. Subroutine INPUT initializes the supply node data which is then stored in array MA. This array will contain unit costs, surpluses, and demands in the same format in which C_{ij}'s, a_i's, and b_j's are arranged in figure 7.

2.2.13.3.1 Programming Specifications. The following table summarizes the principal specifications of subroutine INPUT:

Characteristic	Specification
Formal parameters	MA = Matrix of shipping costs between supply nodes, expanded to contain surplus and deficit at each node
Common blocks	TRANS, STI, PR
Subroutines called	None
Called by	TRANPO

2.2.13.3.2 Logic Functions. During compilation, Block Data subroutine BLOCK1 entered unit shipping costs into the lower triangular half of the matrix MA. Assuming that the cost for shipping between any two nodes is independent of direction, subroutine INPUT fills the upper half of MA. Then INPUT increases the number of supply nodes by 1 to account for a dummy node which will have that surplus or deficit needed to make total surplus equal to total deficit for the system. Other indexes are determined, for use in later calculations, from the number of supply nodes. Among these are M1, two more than the original number of warehouses, and N1, two more than the original number of markets.

Column N1 of array MA is filled with the surplus at each of the original supply nodes. An epsilon value of .01 is added to each surplus value to prevent circling of the algorithm. (See reference 3 for a discussion of this standard perturbation procedure.) The total surplus at all nodes is stored in variable S1.

The cost of shipping to the dummy market is set to zero. Row M1 of array MA is filled with the deficits at each node. Variable S2 retains the total deficit over all nodes. The cost for shipping from the dummy node is set to a large value, 99999. If the total surplus exceeds total deficit, the surplus at the dummy node is set to .01 and the deficit to the difference between the total surplus (including the dummy array) and total deficit. If the total deficit exceeds total surplus, the deficit at the dummy node is set to .01 and the surplus to the difference between the new total

deficit and total surplus. Finally, the values of the array MA are printed out, if detailed reports are requested on the current cycle.

2.2.13.4 INSOL. Subroutine INSOL uses a variant of the row-minimum role to find an initial basic feasible solution to the transportation problem.

2.2.13.4.1 Programming Specifications. The following table summarizes the principal specifications of subrcutine INSOL:

Characteristic	Specification
Formal parameters	MA = Cost, surplus and deficit matrix
	ML = List of basis elements in the initial feasible solution
Common blocks	TRANS
Subroutine called	None
Called by	TRANPO

2.2.13.4.2 Logic Functions. Subroutine INSOL first initializes arrays MM and MN to zero. Array MM is used to indicate when the surplus at a warehouse has been completely exhausted and MN to indicate when the deficit at a market had been completely met.

The routine next makes M+N-1 (one less than the number of warehouses plus the number of markets) iterations, each of which determines a basic element of the initial feasible solution. The procedure followed in each iteration is discussed below. (See figure 7 for the matrix layout of a solution to a transportation problem.)

For each warehouse U whose total surplus has not been distributed, INSOL determines that market V, from among those which still have deficits, which has the minimum unit shipping cost from the warehouse U. In case of a tie, the market with the lowest index is selected. The cell (U,V) is the basic element selected on this iteration and its value is the minimum of: (1) the surplus at the warehouse and (2) the deficit at the market. The indexes of the warehouse and

the market, and the amount of supplies shipped are stored in the first three columns of array ML. The surplus and deficit values are adjusted to reflect this assignment of surplus supplies, and the appropriate flag array (MM or MN) is adjusted. The cost of shipping this amount of supplies is determined from the unit shipping cost for this route and is accumulated in variable C, which after all iterations have occurred will contain the total cost in ton-kilometers for the initial feasible solution.

2.2.13.5 LABEL1. Subroutine LABEL1 sets up the labels corresponding to the list of the initial basic cells.

2.2.13.5.1 Programming Specifications. The following table summarizes the principal specifications of subroutine LABEL1:

Characteristic	Specification
Formal parameters	ML = List of basic elements in the solution
	MI = Warehouse labels
	MJ = Market labels
Common blocks	TRANS
Subroutine called	None
Called by	TRANPO

2.2.13.5.2 Logic Functions. The purpose of this routine is to set up a labeling system by which all the basis elements involving any market or any warehouse can be quickly identified. When this routine has been executed, for each warehouse I array MI will contain in its first column the index of that cell (I,J) in the solution whose market index J is the smallest of all the indexes for markets which are supplied by warehouse I. In other words, MI will contain the first basis elements in row I of the transportation matrix (see figure 7). The second column will contain the associated market index J. Array MJ will contain the corresponding information for each market. For the index of each basis element (I,J), the fourth column of array ML will contain the index of the basis element whose warehouse index is I and whose market index is the next highest index of all the markets supplied by warehouse I (i.e., the next basis element in row I in the matrix). If J is the highest index, then

column 4 will contain a flag N3 to indicate the end of the chain. The fifth column will contain similar information for markets. The procedure used to store this information is described below.

First LABEL1 initializes the first two columns of MI and of MJ, and the fourth and fifth columns of ML to flag N3 which is larger than the number of basis elements in the solution. For each basis element (I,J) of ML, the labelling information is determined and appropriately stored as follows. The market index J is compared to the second column of array MI for warehouse I. If the value in MI is the flag, then this is the first encounter with a basis element having a warehouse index of I. In this case, the index of this basis element is stored in MI as that of the basis element which has the lowest market index for warehouse I. If J is less than the market index J1 in MI, then the index of the basis element (I,J1) is stored in the fourth column of ML for the element (I,J) and the index of the element (I,J) is stored in MI. If J is greater than the market index J1 in MI, then the chain of markets for warehouse I is traced through the fourth column of ML until one basis element has a market index J2 less than J and the next element in the chain has a market index J3 greater than J. If J3 is equal to the flag N3, then J is the largest market index encountered and for the basic element (I,J2) the fourth column of ML is set to the index of the basis element (I,J). If J3 is not equal to the flag than for the basis element (I,J2) the fourth column of ML is set to the index of the basis element (I,J) and for the basis element (I,J) the fourth column is set to the index of the basis element (I,J3). A similar procedure is applied to obtain the ascending chain of warehouses which supply a given market.

2.2.13.6 LABEL2. Subroutine LABEL2 determines the values of the dual variables corresponding to the initial basis.

2.2.13.6.1 Programming Specifications. The following table summarizes the principal specifications of subroutine LABEL2:

Characteristic	Specification
Formal parameters	MA = Matrix of shipping costs between warehouses and markets, and of surpluses and deficits

Characteristic	Specification
	ML = List of basis elements in initial feasible solution
	MI = Warehouse labels
	MJ = Market labels
Common block	TRANS
Subroutine called	None
Called by	TRANPO

2.2.13.6.2 Logic Functions. Consider the matrix of all shipments between warehouses and markets as illustrated in figure 7. The basis elements of the initial feasible solution can be viewed as stepping-stones from which a path can be found from any basis element back to one in the first row of the matrix (see figure 8). (See references 3 and 4.) In the following discussion row I in the matrix is for warehouse I and column J for the market J.

Subroutine LABEL2 determines these paths and stores labels for the paths in arrays MI and MJ. The stepping-stones in a path are selected, starting from a stepping-stone in the first row, by moving alternately first in a column and then in a row. For a row I, the fourth column of MI will contain that column number J such that the stepping-stone (I,J) is the first one encountered in row I in tracing a path from the first row. The fifth column of MI will contain the number of stepping-stones in the path to reach (I,J), and the third column will contain the algebraic sum of the signed costs associated with those stepping-stones in the path. The signed costs are alternately positive and then negative, beginning with the (I,J) stepping-stone and proceeding backwards to the first element of the path. Array MJ contains the corresponding values for columns.

To begin the process of determining the paths of stepping-stones, LABEL2 sets columns 3, 4, and 5 of MI to zero for row 1. Also, column 6 of ML, which will be used as a flag to indicate whether a stepping-stone has been labelled, is initialized to zeros. The stepping-stones in row 1 are

WARE-HOUSE	MARKET				
	M_1	M_2	M_3	M_4	TOTAL
W_1		(2)	(8)		10
W_2	(5)	(10)			15
W_3	*		(5)	(15)	20
TOTAL	5	12	13	15	45

Figure 8. Sample Stepping Stone Path

identified by starting with MI (1,1), the lowest numbered column associated with row 1 in a basis element, and tracing through ML. For each stepping-stone (1,J) in the first row, columns 3, 4, and 5 of MJ are appropriately set for column J, and H2, the counter for row stepping-stones reached on this step, is incremented by 1. Then for each column number J associated with row 1 in a basis element, each of the stepping-stones (I,J) in that column are identified and, unless the stepping-stone has been labelled as flagged by column 6 of ML, the third, fourth, and fifth columns of MI are appropriately set for row I and H1, the counter for column stepping-stones reached in this step, is incremented by 1. This process continues, alternating between rows and columns, until each path from an element of row 1 has been traced to its end.

2.2.13.7 MAIN. Subroutine MAIN finds a nonbasic cell to pivot on, using the row-minimum rule.

2.2.13.7.1 Programming Specifications. The following table summarizes the principal specifications of subroutine MAIN:

Characteristic	Specification
Formal parameters	MA = Matrix of shipping costs between warehouses and markets, expanded to contain surpluses and deficits
	MI = Warehouse labels
	MJ = Market labels
	I = Row index for pivot cell
	J = Column index for pivot cell
	H9 = Unit cost improvement for pivot cell
	K9 = Row index for last row tested in MAIN

Characteristic	Specification
Common blocks	TRANS
Subroutines called	None
Called by	TRANPO

2.2.13.7.2 Logic Functions. Subroutine MAIN first determines a row index, depending on the last row tested (K9), on which to begin searching for a pivot cell. K9 is then reset to this index. It then finds that column J such that the unit cost improvement (H9) for introducing the nonbasic cell (K9,J) into the basis is less than -.0001, i.e., introducing this cell into the basis would yield a cheaper routing. The nonbasic cell selected must also be the one in row K9 which can yield the most reduction in costs. The unit cost improvement (H9) is determined from a stepping-stone path which starts with a stepping-stone in the same column as the cell to be evaluated and alternates between row and column moves until it ends with a stepping-stone (basis cell) in the same row as the cell to be evaluated. The unit shipping cost associated with the stepping-stones are signed, alternating between positive and negative beginning at the first stepping-stone, and their algebraic sum is subtracted from the unit shipping cost of the cell being evaluated to determine its unit cost improvement. The sum of signed costs was calculated as two components in subroutine LABEL2 and stored in MI and MJ. This unit cost improvement will be zero for any cell already in the basis, so that only nonbasic cells will have a value for H9 which is less than -.0001. If a cell is found in row K9 which has a value for less than -.0001, then control is returned to subroutine TRANPO. If no such cell is found, then each of the other rows in the transportation matrix is searched until either a pivot cell is found or all rows have been searched and no improvement can be made to the routing cost.

2.2.13.8 CYCLE. Subroutine CYCLE finds the loop created by the introduction, into the basic solution, of the nonbasic cell determined by subroutine MAIN.

2.2.13.8.1 Programming Specifications. The following table summarizes the principal specifications of subroutine CYCLE:

Characteristic	Specification
Formal parameters	ML = List of basis elements in the solution
	MI = Warehouse labels
	MJ = Market labels
	MR = Stepping-stone path for the new element which is to be introduced into the basis
	I = Warehouse index of new basis element
	J = Market index of new basis element
	I8 = Index of element in MR which is minimum giver
	I9 = Flag to indicate whether row way is best or not
Common blocks	TRANS
Subroutines called	None
Called by	TRANPO

2.2.13.8.2 Logic Functions. The subroutine code is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Determine Stepping-Stone Path. Subroutine CYCLE determines from MI and MJ, the warehouse and market labels, the stepping-stone path around the new element (I,J) to be introduced into the basis. The path will begin with a basis element in row I and end with a basis element in column J of the transportation matrix. When this subroutine has completed execution, the array MR will contain

information about the stepping-stone path. For each row of MR, the first column will contain the warehouse index I associated with a stepping-stone in the path. Column 2 will contain that market index J such that (I,J) is a stepping-stone in the path and, if I' is the warehouse index in the previous row of MR, (I',J) is the stepping-stone which immediately precedes (I,J) in the path. Columns 3 and 4 will contain the basis element indexes for (I',J) and (I,J), respectively.

Array MR is filled with I as the warehouse index for the first row and J as the market index for row N2, the last row in array MR. The path around the new element will be traced from both the Ith row and the Jth column and when all stepping-stones have been identified, MR will be compressed to give a continuous stepping-stone path around cell (I,J). From MI and MJ are determined the numbers of steps needed to trace a stepping-stone path from row I back to row 1 (S8), and from column J back to row 1 (S9). The values of S8 and S9 are compared. If more steps are needed to trace back from row I, then the stepping-stone path from row I back to row 1 is traced (using the labels stored in MI and MJ) until that column J', which is S9 steps away from row 1, is reached. The warehouse and market indexes for the stepping-stones in the first part of this path are stored appropriately in MR, beginning with the second row of MR. If J and J' are the same column, then the path from row I to column J has been identified. If the complete path has not been found, then the path from row I to row 1 and the path from column J to row 1 will each be traced one step at a time until the paths meet at a row or a column. The paths eventually will meet since each connects with row 1. At each step, the appropriate values are stored in MR.

If more steps are needed to trace back to row 1 from column J than from row I, then the path from column J is traced until row I' which is S8 steps away from row 1 is reached. If rows I and I' are the same, the complete path has been found. If not, then both the column J path and the row I path are traced until they meet.

When all stepping-stones in the path have been identified, MR is compressed to give a continuous path, i.e., data in the rows of MR which were filled by tracing from column J

are moved to those rows which immediately follow the data acquired by tracing from row I.

b. Section 20 - Find Locations of the Cells in List. Next, CYCLE determines the basis element index for each of the stepping-stones in the path just defined and stores them in columns 3 and 4 of MR. Let R2 and C2 be the warehouse index and market index, respectively, in a given row of MR, and let R1 be the warehouse index for the previous row. The basis element indexes for (R1,C2) and (R2,C2) are determined by tracing the basis elements associated with C2 from MJ(C2,1), that basis element associated with C2 which has the lowest numbered warehouse index, through the chain denoted in column 5 of ML. Since the warehouse indexes are in ascending order in the chain, the smaller of R1 and R2 is searched for first. When the basis element indexes have been determined they are properly stored in MR.

c. Section 30 - Find Minimum Giver. Next the minimum giver is determined by testing each of the stepping-stones listed in column 3 of MR to select that basis element which has the smallest value, i.e., for which the smallest amount of goods is shipped. In case of a tie, the first stepping-stone referenced in column 3 with the smallest value is designated as the minimum giver. This stepping-stone will then be replaced by the new basis element which is being introduced.

d. Section 40 - Is Row Way Best. This section determines the value of the flag I9 which will be used later by subroutine IJFIX to determine whether to start with the row index or the column index in adjusting the dual variables. This value is determined by the number of steps to trace from the row or column of the minimum giver back to the first row. If the row way requires more steps, then I9=1; otherwise I9=0.

e. Section 50 - Alter Shipments. The last section adjusts the shipments for all elements in the stepping-stone path around the new element. From the shipment of each basis element identified in column 3 of MR (the givers) is subtracted the amount shipped by the minimum giver. To each shipment of a basis element identified in column 4 of MR (the receivers) is added the amount shipped by the minimum giver. The shipment for the new basis element, which will

replace the minimum giver in the basis elements array ML, is set to the value of the shipment for the minimum giver.

2.2.13.9 FIXLIJ. Subroutine FIXLIJ modifies the labels to correspond to the new basis.

2.2.13.9.1 Programming Specifications. The following table summarizes the principal specifications of subroutine FIXLIJ.

Characteristic	Specification
Formal parameters	ML = List of basis elements in the solution
	MI = Warehouse labels
	MJ = Market labels
	I = Warehouse index of new element
	J = Market index of new element
	I8 = Index of basis element which is being replaced in the solution
Common blocks	TRANS
Subroutines called	None
Called by	TRANPO

2.2.13.9.2 Logic Functions. Let I7 and J7 be the warehouse and market indexes associated with the basis element which is being removed from the solution. First, the chain of basis elements associated with warehouse I7 are traced to find the basis element (I7,J7), which is then removed from the chain. This is accomplished by adjusting MI(I7,1) and the fourth column of ML appropriately, as described below. If J7 is the smallest indexed market, then MI(I7,1) is reset to the column index J" such that basis element (I7,J") immediately follows (I7,J7) in the chain. If J7 is greater than the smallest indexed market, the array ML is traced, using the fourth column, until the basis element (I7,J7) is

found. Then, if (I7,J7') is the basis element which immediately precedes (I7,J7) in the chain, then the fourth column of ML for the basis element (I7,J7') is set to the index of of the basis element (I7,J7"), which immediately follows (I7,J7) in the chain. This removes (I7,J7) from the chain and closes the resulting gap in the chain. Similarly, (I7,J7) is removed from the chain of basis elements associated with market J7.

The subroutine then inserts the new basis element (I,J) into the chain for warehouse I and the chain for market J. Again MI, MJ and ML are adjusted, this time to include the new basis element.

Finally the new warehouse index I, and market index J replace the old basis element values in array ML.

2.2.13.10 IJFIX. Subroutine IJFIX modifies the dual variables to correspond to the new basis.

2.2.13.10.1 Programming Specifications. The following table summarizes the principal specifications of subroutine IJFIX:

Characteristic	Specification
Formal parameters	ML = List of basis elements in solution
	MI = Warehouse labels
	MJ = Market labels
	I8 = Index of basis element which is being replaced in the solution
	I9 = Flag to indicate whether row way is best or not
	H9 = Unit cost improvement for new basis element
	I = Warehouse index of new basis element

Characteristic	Specification
	J = Market index of new basis element
Common blocks	TRANS
Subroutines called	None
Called by	TRANPO

2.2.13.10.2 Logic Functions. Subroutine IJFIX adjusts columns 3, 4, and 5 MI and MJ to reflect the introduction of the new basis element (I,J) into the solution. First the sixth column of ML, which is used as a flag to indicate whether a stepping-stone has been labelled, is initialized to zero. If the row way is best, as indicated by I9=1, then row I of array MI is adjusted and all the stepping-stone paths from row I are traced and labelled. If the column way is best (I9=0), then row J of array MJ is adjusted and all the stepping-stone paths from column J are traced and labelled. The adjustments made in the first case (I9=1) are described below.

To the sum of signed costs in MI(I,3) is added the unit cost improvement for the new basis element. The first stepping-stone reached in row I is (I,J) since it is replacing the stepping-stone which previously was the first stepping-stone reached in tracing paths from row 1. Therefore, MI(I,4) is set to J. The number of steps to reach this row is 1 more than the number to reach column J, thereby determining the value of MI(I,5).

For each column associated with row I in a basis element, not previously labelled, columns 3, 4 and 5 of MJ are adjusted to correctly label the new paths created by the introduction of the new basis element. To the sum of signed costs in column 3 of MJ is subtracted the unit cost improvement for the new basis element. The row index and the number of steps from row 1 are set to indicate the new paths. Then for each row I', associated with the columns which were associated with row I, MI is adjusted to reflect the paths from row 1 to row I'. This process continues, alternating between rows and columns, until each path has been traced to its end.

2.2.13.11 OUTPUT. Subroutine OUTPUT adjusts the shipping assignments and prints out the optimal primal solution.

2.2.13.11.1 Programming Specifications. The following table summarizes the principal specifications of subroutine OUTPUT:

Characteristic	Specification
Formal parameters	MA = Matrix of shipping costs, surpluses and deficits.
	ML = List of basis elements in solution
	MI = Warehouse labels
	MJ = Market labels
	TDEF = Total deficit
	TSUR = Total surplus
	SUPASN = Supplies arriving at each supply node
	C = Cost in ton-kilometers for the optimal solution
Common blocks	TRANS, PR
Subroutines called	None
Called by	TRANPO

2.2.13.11.2 Logic Functions. Subroutine OUTPUT adjusts the values for the amount of goods assigned to those existing supply nodes in the basis. If total surpluses are less than total deficits, the amount assigned to a supply node is increased by the amount of goods shipped from the dummy node to that node. This simulates immediate arrival of goods from outside the network. If total surpluses exceed total deficits, the amount assigned to a supply node is decreased

by the amount shipped from that supply node to the dummy node. In this case, surplus nodes are allowed to retain goods not needed by other nodes in the network.

If the user requests detailed output, then for each basis element (U,V) in the optimal solution, OUTPUT prints out the indexes U and V and the amount of goods shipped from warehouse U to market V. All costs associated with these shipments are accumulated in C which is printed out as the cost of the optimal solution.

2.2.13.12 BLOCK1. This Block Data routine loads the cost array MA.

2.2.13.12.1 Programming Specifications. The following table summarizes the principal specifications of BLOCK1:

Characteristic	Specification
Formal parameters	None
Common blocks	MATRIX
Subroutines called	None
Called by	N/A

2.2.13.12.2 Logic Functions. Through the use of Data statements, the lower triangular half of the matrix MA is loaded with shipping costs for the 95 supply nodes. All the remaining values of MA are set to 99999.

2.2.14 LINKM. This subsection describes the routines in LINKM. These routines process the TIMET inputs.

2.2.14.1 TIMET. Subroutine TIMET handles all resource changes as they occur throughout the war. It calls subroutine ASSIGN to assign arriving units to specific locations within the theater.

2.2.14.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine TIMET.

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	CVFW, ASSIGN
Called by	TMAIN

2.2.14.1.2 Logic Functions. Subroutine TIMET is called only if the user designates changes to be made to forces or to model parameters. The code is divided by comment cards which label the functions performed by the logic.

a. Section 10 - Read Time-t Inputs. This section of TIMET begins by reading a record (IREC) from file ITTD. See subsection 3.2.1.2 for a description of file ITTD. If an end of file marker is encountered, then the index to the next day for time-t inputs (IGO) is set equal to 99999 and control is returned to TMAIN. If a 9999 is encountered in the first word of IREC, it indicates that the end of the records to be read on the current day has been reached. In this case, IGO is set equal to IFN(=IREC(2)), the number of the next day to read time-t inputs, and control is returned to TMAIN. If the first word of IREC contains a 3, then the record is a unit record and the logic flow branches to section 30.

b. Section 20 - Increment or Replace Specified Parameters in Blank Common. This section of TIMET processes all changes to model parameters other than unit arrivals. The values of specified integer or real variables within blank Common are incremented or replaced according to the information contained in IREC. When all specified changes have been made, the logic flow returns to section 10 so that the next record of file ITTD can be read and processed.

c. Section 30 - Process Unit Arrivals. In this section of TIMET, the number of people (PDIV), supplies (SPIV), weapons by type (WDIV), and subunits by type (NSUTD) in the arriving division are set to TOE levels unless specific values for these variables have been input (via IREC) by the user. The fractional personnel strength (PPS) of the arriving division is computed and subroutine CVFW is called to determine the fractional effectiveness of a division on defense (PED) as a function of personnel strength. The number of days of supply on hand (DSH) is computed and subroutine CVFW is called to determine the fractional effectiveness of a division due to supply shortages (SEF). The overall effectiveness (on defense) of the arriving division is then computed. Next, subroutine ASSIGN is called to assign the division to a particular location in the theater. Then appropriate values are assigned to the variables indicating the division's location. The logic flow then returns to section 10 so that the next record of file ITTD can be read and processed.

2.2.14.2 ASSIGN. Subroutine ASSIGN is a rather complex routine that processes the assignment of arriving combat units to battle areas directly, to sectors from a region assignment, or to sectors or regions from a COMMZ assignment. For processing the unit assignments, subroutine ASSIGN has many assignment options available to it. These options are outlined below:

I. Battle area assignment options

   (a) Assign to a battle area directly (by user input)

   (b) Assign to last battle area in a sector

   (c) Assign to first inactive battle area in a sector

II. Sector assignment options

   (a) Assign to a sector directly (by user input)

   (b) Assign to the sector of main attack (SMA) in a region

   (c) Use region to sector assignment logic in table 5

III. (a) Assign to a region directly (by user input)

   (b) Use COMMZ to region assignment logic in table 5.

Table 5 summarizes the region to sector and COMMZ to region assignment logic options.

2.2.14.2.1 Programming Specifications. The following table summarizes the principal specifications of subroutine ASSIGN:

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common
Subroutines called	IFEBA, IRATIO
Called by	TIMET

2.2.14.2.2 Logic Functions. The code of subroutine ASSIGN is divided by comment cards which label the functions performed by the logic.

a. Section 5 - Initialize Working Variables. This section is executed only once on any day, namely the first time ASSIGN is called that day. First, the index MPDAY is set equal to IDAY, the index to the day currently being processed. The section then compares MPDAY to IDAY, and if they are equal (i.e. if ASSIGN has already been executed at least once on the current day), the logic flow branches to section 10. Otherwise, for each side, the total combat value of all divisions in the active and first inactive battle areas (SEFF) is computed for each sector. Also, three indices are initialized:

(1) KD is set equal to the side index of the theater defender.

(2) KFOPT is set equal to the option for computing FEBA location for the theater attacker relative to a given FEBA location.

(3) IFIRST(L), the number of divisions that have already been assigned on the current day, is set equal to 0 for each side.

b. Section 10 - Determine Side and Assignment Option. This section of ASSIGN begins by setting the variable ISIDE equal to the number of sectors (NS), if the arriving division belongs to Blue. If it belongs to Red, ISIDE is set

Table 5. Assignment Options for Arriving Units

CORPS TO REGION ASSIGNMENTS

I. THEATER ATTACKER ASSIGNMENTS (See variable ICADTA)

Tactical Situation	Unit Assignments Side 1 = CTA[a]	Unit Assignments Side 2 = CTA
One SMA[b] constrained	To region containing the constraining sector	To region containing the constraining sector
Multiple SMAs constrained	To region with sector of max FEBA[c]	To region with sector of min FEBA
No constrained SMAs (or multiple arrivals)	*Option 1.* In order of arrival to min FEBA	*Option 1.* In order of arrival to max FEBA
	Option 2. In order of arrival to region with min FR (ties broken by max FEBA)	*Option 2.* In order of arrival to region with min FR (ties broken by min FEBA)

II. THEATER DEFENDER ASSIGNMENTS (See variable ICADTD)

	Side 1 = CTD	Side 2 = CTD
Option 1	-Unit #1 to max FEBA; other units to region with sector of max FR against TD (ties broken by max FEBA)	-Unit #1 to min FEBA; other units to region with sector of max FR against TD (ties broken by min FEBA)
Option 2	-Units assigned in order to region with sector of max FR (ties broken by max FEBA)	-Units assigned in order to region with sector of max FR (ties broken by min FEBA)

REGION TO SECTOR ASSIGNMENTS

I. THEATER ATTACKER ASSIGNMENTS (See variable IRADTA)

Tactical Condition	Unit Assignments Side 1 = CTA	Unit Assignments Side 2 = CTA
Option 1		
CTD attacking in sector j	Unit assigned to sector j	Unit assigned to sector j
CTD attacking in many sectors	a. First unit assigned to sector of max FEBA	a. First unit assigned to sector of min FEBA
	b. Follow-on units to sectors of next least advanced FEBA (until all sectors where CTD is attacking have been considered)	b. Follow-on units to sectors of next least advanced FEBA (until all sectors where CTD is attacking have been handled)
CTA is attacking in and/or constrained from moving in all sectors	a. Check SMA. If constrained, assign first unit to adjacent sector causing constraint	a. Check SMA. If constrained, assign first unit to adjacent sector causing constraint
	b. Check sma. If constrained, assign first unit to adjacent sector causing constraint	b. Check sma. If constrained, assign first unit to adjacent sector causing constraint
	c. If neither the SMA nor the sma is constrained, assign the first unit to the SMA	c. If neither the SMA nor the sma is constrained, assign the first unit to the SMA
For subsequent units arriving this day	Units assigned, in order, to sector of min FR (ties broken by max FEBA)	Units assigned, in order, to sector of min FR (ties broken by min FEBA)
Option 2		
For the assignment of *all* units this day	Units assigned, in order, to sector of min FR (ties broken by max FEBA)	Units assigned, in order, to sector of min FR (ties broken by min FEBA)

II. THEATER DEFENDER ASSIGNMENTS (See variable IRADTD)

	Side 1 = CTD	Side 2 = CTD
Option 1		
a. First arriving unit	a. To sector of max FEBA	a. To sector of min FEBA
b. Subsequent unit arrivals this day	b. Units assigned, in order, to sector of max FR against defender (ties broken by max FEBA)	b. Units assigned, in order, to sector of max FR against defender (ties broken by min FEBA)
Option 2		
For the assignment of *all* units this day	Units assigned, in order, to sector of max FR against defender (ties broken by max FEBA)	Units assigned, in order, to sector of max FR against defender (ties broken by min FEBA)

[a] CTA = current theater attacker (CTD = current theater defender).

[b] SMA = sector of main attack (sma = sector of maximum advance).

[c] FEBA is determined by user to be either F_j or ΔF_j, where F_j means FEBA location in sector j as measured from original base point, and ΔF_j means FEBA advance in sector j from time zero FEBA location.

equal to -(NS). The index KAD is set equal to 1 if the arriving division belongs to the theater defender and to 2 if it belongs to the theater attacker. The values of the battle area (KBA), sector (KSA) and region (KREG) assignment variables are then used to determine ITASG, the index to the method for assigning the arriving division to a battlefield location. The value of ITASG is set according to the following rules:

(1) If no preassigned location is indicated, the unit is considered available for assignment theater-wide (ITASG=6).

(2) If a region assignment is made, the following will occur: the number 99 in the sector field (KSA) will assign the unit to the sector of main-attack of the indicated region (ITASG=4); if the sector field is left blank the unit assignment logic will hold (ITASG=5).

(3) If a sector assignment is made, the following will occur: with the number 999 in the battle area field (KBA), the unit will be assigned to the first inactive battle area of the indicated sector (thus ignoring the move through the theater to the front, or portraying an airlift of units to the front) (ITASG=3); if the battle area field is left blank, the unit is assigned to the most rearward battle area of the indicated sector (ITASG=2).

(4) If a battle area assignment is made directly, the unit will be assigned to that battle area provided the proper side owns the battle area (ITASG=1).

This section of ASSIGN ends by checking the value of ITASG and transferring control to the section of code containing the appropriate assignment logic.

c. Section 11 - Battle Area Assigned Directly. This section of ASSIGN is executed if ITASG=1. The arriving division is to be assigned to the battle area specified by the user (KBA). The sector containing the assigned battle area (IS) and the location of the assigned battle area relative to the active battle area in sector IS are determined. If the opposing side owns the assigned battle area, or if it is an active battle area, the division is reassigned to its own side's first inactive battle area in the sector.

In this case, and when the assigned battle area is the first inactive battle area, the model checks to see if the first inactive battle area is full. If it is full, the division is reassigned to the second inactive battle area in the sector. Boundary conditions are checked to be sure that the battle area that has been assigned actually exists. If the boundary conditions are violated, the specified assignment cannot be made and the division is considered available for assignment theater-wide. The logic flow branches to section 16 in this case. If the boundary conditions are satisfied, then the division is assigned to the appropriate battle area (KBA), and the logic flow branches to section 25.

d. Section 12 - Sector Assignment, Last Battle Area. This section of subroutine ASSIGN is executed if ITASG=2. The arriving division is to be assigned to the last battle area in the sector specified by the user (KSA), unless the last battle area is active or unless it is the first inactive battle area and is full. If the specified assignment cannot be made, the division is considered available for assignment theater-wide, and the logic flow branches to section 16. Otherwise, the division is assigned to the appropriate battle area (KBA) and the logic flow branches to section 25.

e. Section 13 - Sector Assignment, First Inactive Battle Area. This section of subroutine ASSIGN is executed if ITASG=3. The arriving division is to be assigned to the first inactive battle area in a specified sector (KSA) if that battle area exists and is not full. If the first inactive battle area is full, the division is reassigned to the second inactive battle area, if it exists. If the desired battle area does not exist, control is transferred to section 20 where the run is aborted. Otherwise, the division is assigned to the appropriate battle area (KBA) and the logic flow branches to section 25.

f. Section 14 - Assign to Sector of Main Attack in Given Region. This section of subroutine ASSIGN is executed if ITASG=4. The arriving division is assigned to the last battle area (KBA) in the sector of main attack in the region specified by the user (KREG). The logic flow then branches to section 25.

g. Section 15 - Assignment to a Sector Within a Region. This section of subroutine ASSIGN is executed if ITASG=5. The arriving division is to be assigned to a sector within a specified region (KREG). Section 15 contains two subsections

which provide the assignment logic for the theater defender (TD) and the theater attacker (TA), respectively. Two assignment options are also included within each subsection. Two functions, IFEBA and IRATIO, are called repeatedly in this section and in section 16. Function IFEBA is used to determine in which of the two sectors the FEBA has advanced the most (or the least) and function IRATIO is used to determine which of two sectors has the best (or worst) force ratio.

Section 15 begins by setting the index KOPT equal to the reqion assignment option for the attacker or defender, according to the value assigned to KAD in section 5. The logic flow then branches to the appropriate subsection with its corresponding options to process the assignment.

(1) Theater Defender Assignments. The following options are included in this subsection:

Defender Option 1.

a. The first arriving division for the TD in a particular region is assigned to that sector where the theater attacker has advanced the most, i.e., the sector of maximum advance as measured from the initial FEBA (or the base location).

b. Successive unit arrivals on a given day are assigned, one-at-a-time, to that sector of the region with the largest force ratio against the theater defender, with ties in appropriate sectors beign broken by choosing that sector where the theater attacker has advanced the most (i.e., sector of maximum advance as measured, according to KFOPT, from the initial time-zero FEBA or theater base location).

Defender Option 2.

a. Use directly the logic of (1)(b) above with ties being broken as indicated.

When the division has been assigned to a sector (KSA), the logic flow branches to section 13 in order to assign the division to a battle area within sector KSA.

(2) Theater Attacker Assignments. The following options are included in this subsection:

Attacker Option 1.

a. If the theater defender (TD) is attacking in any sector of this region, the TA's first arriving unit is assigned to that sector. If the TD is attacking in more than one sector, the TA's arriving unit is assigned to the sector of minimum FEBA advance (advance as measured from either initial FEBA or original base point) within those sectors where the TD is attacking.

b. If the TA is either attacking in or constrained from moving in all sectors of the region, the unit assignment is made as follows: (1) check sector of main attack; is it constrained from moving? If not, (2) check sector of maximum advance; is it constrained from moving? If either of these sectors are constrained from moving, the arriving unit is assigned to the adjacent sector causing the constraint, but in the order indicated above. If neither sector is constrained, the unit is assigned to sector of main attack.

c. For other situations (such as sectors holding because of insufficient forces to attack) and for the assignment of successive unit arrivals on a given day, the following logic is used. The arriving units, one-at-a-time, will be assigned to that sector of the region with the smallest force ratio for the TA when units in the active battle area and first inactive battle area are considered. (Ties in the appropriate sectors are broken by choosing the sector with minimum FEBA advance as measured from initial FEBA location or the base location.)

Attacker Option 2.

Use the logic in (1)(c) above to assign all units to sectors from the region level, but break ties in appropriate sectors by choosing minimum FEBA

advance as measured from initial FEBA location or theater base location.

When the division has been assigned to a sector (KSA), the logic flow branches to section 19 where the division is assigned to the first inactive battle area in sector KSA.

h. Section 16 - COMMZ Assignment of Units. This section of subroutine ASSIGN is executed if ITASG=6. The arriving division is considered to be available for assignment theater-wide. The section contains two subsections which provide the assignment logic for the theater defender (TD) and the theater attacker (TA), respectively. Two assignment options are included within each subsection. According to the value assigned to KAD in section 5, the logic flow branches to the appropriate subsection to process the assignment.

(1) Theater Defender Assignments. This subsection begins by setting KOPT equal to the COMMZ assignment option for the defender. The following options are included in this subsection:

Defender Option 1.

The first arriving unit is assigned to the sector of maximum FEBA advance (as measured from initial FEBA or the base location) considering all sectors in the theater. The second arriving unit and follow-on units are assigned, one-at-a-time, to that sector with the largest force ratio against the TD when all the forces in the active battle area plus the first inactive battle area of the sector are considered. Ties in force ratio value among sectors are broken by choosing the sector where the TA has made maximum advance from initial FEBA (or the base) location.

Defender Option 2.

The logic of largest force ratio value against the defender is used for all unit arrivals, with ties being broken as indicated.

When the division has been assigned to a sector (KSA), it is assigned to the first inactive battle area (KBA) and the logic flow branches to section 25.

(2) Theater Attacker Assignments. This subsection begins by setting KOPT equal to the COMMZ assignment option for the theater attacker. The first arriving unit (on a given day) is assigned to that region containing the sector that is causing a sector of main attack (SMA) (because of the extent of exposed flank) to be constrained from moving. If more than one SMA is constrained, the first arriving unit is assigned to that region containing the sector (adjacent to a constrained SMA) that has minimum FEBA advance (as measured from initial FEBA or from original base point). The second arriving unit, on a given day, is assigned to the region containing the sector adjacent to a constrained SMA that has the next least-advanced FEBA. This procedure is contained until all constrained SMAs are accounted for; i.e., the appropriate region has been assigned one division per SMA.

If none of the main attack sectors are constrained from moving, or if additional arriving units are still available, two assignment options are available.

Attacker Option 1.

The first arriving unit is assigned to the sector that has made maximum advance from the initial FEBA location (or maximum advance from theater base location) considering all sectors in the theater. The second unit, and all follow-on units are assigned according to the logic described under option 2 below.

Attacker Option 2.

Multiple unit arrivals on a given day are assigned, one-at-a-time, to that region with the smallest force ratio for the TA when all the forces in the active battle area plus the first inactive battle

area of all the sectors of the region are considered. (Note: assignment of units already made this day have been included.) Ties are broken in appropriate regions by choosing that region containing the sector with minimum advance from the initial FEBA location (or the base location).

When the division has been assigned to a region (KREG), the logic flow branches to section 15 in order to assign it to a sector within region KREG. When the division is assigned to a sector (KSA) in this section (option 1), the logic flow branches to section 19 where the division is assigned to the first inactive battle area in sector KSA.

i. Section 19 - Unconditional Assignment to First Inactive Battle Area. In this section of ASSIGN, the arriving division is to be assigned to the first inactive battle area of a specified sector. If boundary conditions are not violated by this assignment, the logic flow branches around section 20 to section 25.

j. Section 20 - Boundary Conditions Violated-Abort Run. This section of ASSIGN is executed only if the logic of one of the preceeding sections causes a division to be assigned to a "nonexistent" battle area (i.e., a battle area with an identification number less than 1 or greater than the number of battle areas in the theater (NBA)). This situation would occur if the first (last) battle area in a sector were an active battle area or if it were a full inactive battle area. If either case, an arriving division could not be assigned to the first (last) battle area and the program logic would assign it to the next battle area in the sector, even though no other battle areas exist. When this occurs, a check on boundary conditions transfers the logic flow to section 20 which aborts with the message "STOP 133"

k. Section 25 - Assign Division and Update Accounting. This section begins by incrementing IFIRST, the number of divisions already assigned on the current day, by 1 for the appropriate side (L). The combat value of the arriving division is added to the total combat value of the divisions in the appropriate sector (IS). The maximum number of additional divisions (MAD) for side L is decreased by 1 and the number of side L divisions (ND) is increased by 1. If the

division has been assigned to the first inactive battle area in sector IS, then the number of divisions in that battle area (NDIBA) is increased by 1. The new force ratio (X) in sector IS is also computed. Control is returned to TIMET from this section.

2.2.14.3 IRATIO. Function IRATIO selects from two sectors, K1 and K2, the one having the best (or worst) force ratio.

2.2.14.3.1 Programming Specifications. The following table summarizes the principal specifications of function IRATIO:

Characteristic	Specification
Formal parameters	L = Side from whose viewpoint selection is to be made
	K1,K2 = Sector numbers of sectors being compared
	SEFF = Combat value for each side in each sector
	MBEST = Index to best/worst option
	JAT = Index to attacker
Common blocks	None
Subroutines called	None
Called by	ASSIGN

2.2.14.3.2 Logic Functions. Given two sectors (K1 and K2), the combat value for each side (SEFF), the side (L) under consideration, and the attacker (JTA), function IRATIO returns to the calling program the sector number of the sector having the best or worst (from the point of view of side L) force ratio. If the index MBEST=1, the sector with the best ratio is returned; if MBEST=2, the sector with the worst ratio is returned.

2.2.14.4 IFEBA. Function IFEBA selects from two sectors, K1 and K2, the one having the minimum or maximum FEBA location. The FEBA location can be measured in two ways: (1) as the absolute FEBA location (measured from the theater base line) or, (2) as delta FEBA, the change in the FEBA location within a sector since time zero.

2.2.14.4.1 Programming Specifications. The following table summarizes the principal specifications of IFEBA:

Characteristic		Specification
Formal parameters	L	= Side from whose viewpoint measurement is to be made
	K1,K2	= Sector numbers of sectors being compared
	FEBA	= Current FEBA location
	FEBATZ	= FEBA location at time zero
	KFOTP	= Index to FEBA location option
	MINMAX	= Index to minimum/maximum FEBA option
Common blocks	None	
Subroutines called	None	
Called from	ASSIGN	

2.2.14.4.2 Logic Functions. Given two sectors (K1 and K2), the current and time-zero FEBA locations of each of the two sectors (FEBA and FEBATZ) and the side (L) under consideration, function IFEBA returns to the calling program the sector number of the sector having the minimum or maximum (from the point of view of side L) FEBA location. If the index MINMAX=1, the sector with the minimum FEBA is

returned; if MINMAX=2, the sector with the maximum FEBA is returned. If the index KFOPT=1, the absolute FEBA locations are compared between sectors; if KFOPT=2, the changes in the FEBA locations are compared.

2.2.15 LINKN. This subsection describes subroutine PSUMMY which is the only routine in LINKN.

2.2.15.1 PSUMMY. Subroutine PSUMMY prints summary data tables S-1 through S-10 displaying the ground combat and logistic data for both Red and Blue forces at the end of each designated combat cycle. PSUMMY is always called on the first and last cycles. It is also called when the index for printing detailed results (IPRD)=1 or when the current combat cycle is equal to the value of the index for printing summary output IPRSO.

2.2.15.1.1 Programming Specifications. The following table summarizes the principal specifications of subroutine PSUMMY.

Characteristic	Specification
Formal parameters	None
Common blocks	Blank Common, CW
Subroutines called	None
Called by	TMAIN

2.2.15.1.2 Logic Functions. PSUMMY is the last routine to be executed in any cycle and reflects all changes due to combat attrition or the increase in forces resulting from arrivals that cycle. It uses formatted write statements and loops, where appropriate, to generate the following tables:

Tables	Headings
S-1	Attacker Indices and Geographical Quantities
S-2	FEBA and Force Ratios
S-3	Divisions in Theater
S-4	Personnel and Weapons in Active Battle Area of Each Sector
S-5	Division Statistics
S-6	People and Weapon Replacement Pool Inventories
S-7	Cumulative Supply Consumption and Losses

Tables	Headings
S-8	Cumulative Casualties and Weapon Losses Due to Ground Combat
S-9	Cumulative Killer - Target Scoreboard (Ground Combat)
S-10	Supply Nodes--Owner, Inventory, Arrivals, and Cumulative Supplies Destroyed

THIS PAGE INTENTIONALLY LEFT BLANK

SECTION 3. INPUT/OUTPUT DESCRIPTION

This section provides both a general and detailed description of the inputs and outputs of the TACWAR model. The general description is an overview of the nature and contents of the model inputs and outputs. The detailed description includes discussions with examples of the format and content of the various TACWAR input files and output reports. This section also discusses TACWAR program variables.

3.1 General Description

The TACWAR model uses two input data files, one working file, and five output files as illustrated in figure 9. The input files contain data which serve to structure the model theater of battle, to allocate and maneuver personnel and materiel, and to define the rules of the game. The one working file is used to store time-t data from the user selected input data file in a format which can be accepted by subroutine TIMET on appropriate days. Outputs produced by TACWAR consist of five kinds of printed listings, each of which is written to one or more output files. The outputs include: (1) an alphabetic listing of the blank Common variables with their initial data values (2) theater control initialized data, (3) records of selected inputs in tabular form, (4) detailed game reports, and (5) summary game reports.

Throughout the TACWAR model, variable names are used to indicate files in input/output commands such as READ and WRITE. Table 6 shows the file codes which subroutine INP assigns to each of the files used in TACWAR. Since file codes are not hard coded, the user can easily change them to meet his own requirements.

3.2 Characteristics, Organization, and Detailed Description

The TACWAR model uses two input data files, one working file for time-t data, and four groups of output files. These files are described in the following subsections.

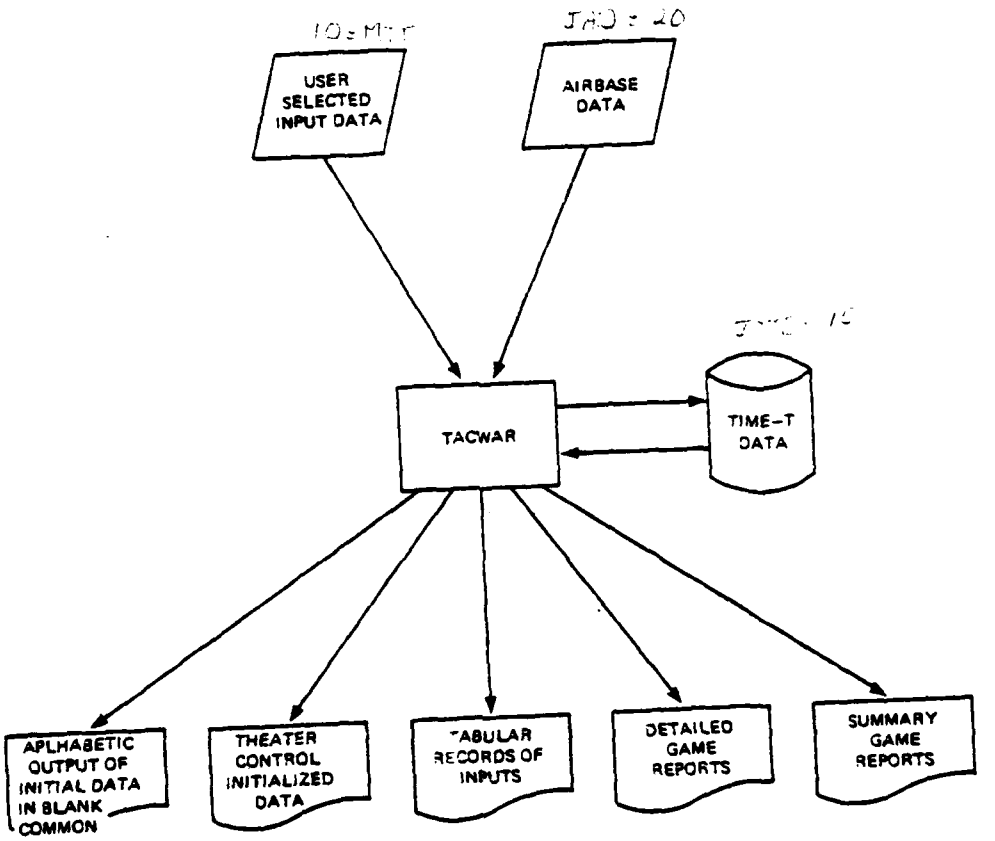


Figure 9. TACWAR Information Flow

Table 6. File Codes Assigned to the TACWAR Input/Output Files

File Name	Description	File Code
IAD	Airbase input data	20
ITTD	Time-t data on working file	15
JCHEM	Chemical detailed report	9
JCON	Conventional detailed report	7
JINP	Display of inputs	4
JNUC	Nuclear detailed report	8
JSUM	Summary output tables	6
MIT	User-selected input data	10

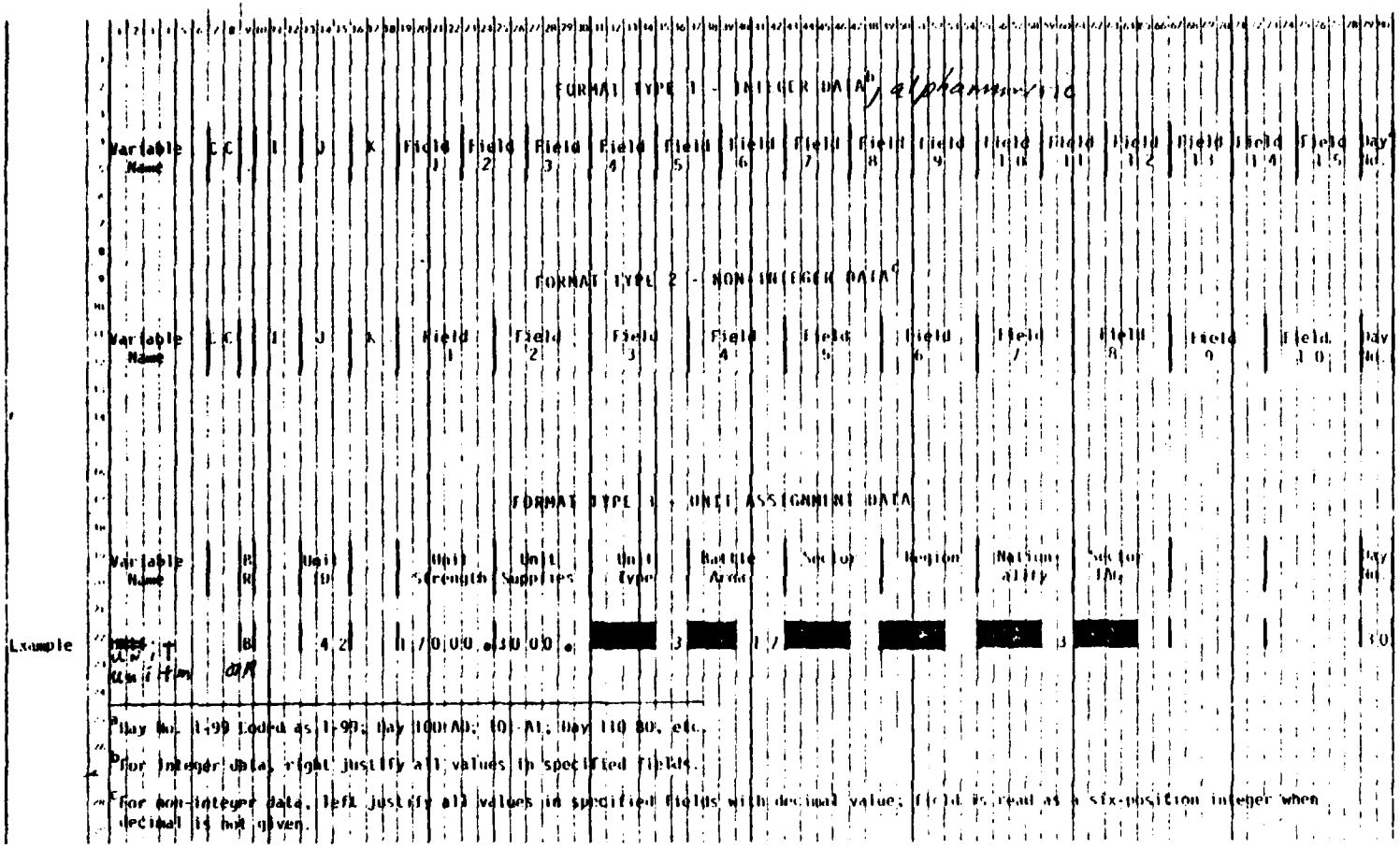
3.2.1 Input and Working Files. Initial user-selected input data for the TACWAR model are read from one file, MIT, whose contents define the theater structure, game parameters, and resources to be played. If this file contains data for time-t updates, then the input routine will reformat these data and write them to a working file, ITTD, which will be read and processed during the course of the simulation. A second input file, IAD, contains airbase data which are not user-selected but may be updated using programs NOTION and AFLDS described in subsection 4.1.1.2.

3.2.1.1 Input File MIT (User-Selected Data). File MIT, which is read in subroutine INP, consists of cards or card images which contain a user-selected group of time-zero and time-t model inputs. The user is permitted flexibility in the coding of all data items. The only restriction placed on the user is that cards must be in ascending order of days. Within the data for any given day, the cards need not be in any order. A variable name of "ZZZZZZ" denotes the end of file MIT. Figure 10 depicts the three possible formats by which all data are entered. They are (1) a four-column (integer or alphanumeric) data field format; (2) a six-column (noninteger) data field format; (3) a specially designed format for characterizing units that arrive in the theater after the war begins. Figure 11 contains a portion of a sample data file. Blanks may be used for zero values on the data files.

3.2.1.1.1 Types 1 and 2 Data. For types 1 and 2, the information in columns 1-18 and in columns 79-80 is identical and is described below.

a. Variable Name. (Columns 1-6.) The alphanumeric name of the blank Common variable to which the data on the card apply.

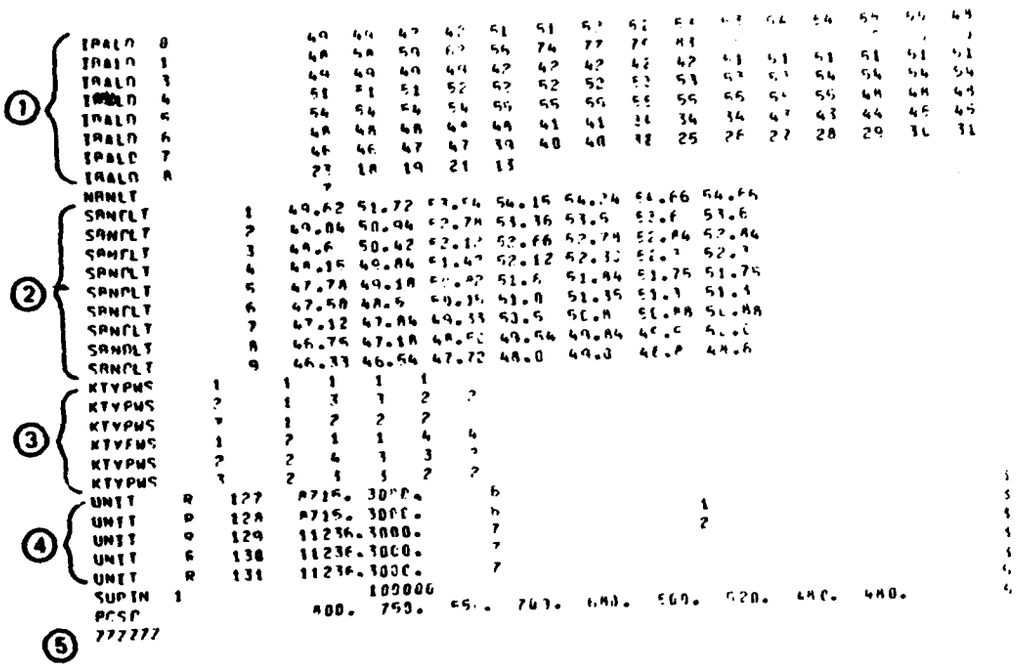
b. Continuation. (Columns 7-8.) When more than 10 values are required in the case of noninteger data or more than 15 values in the case of integer data, the continuation number indicates the index values that are assumed.



unit = arriving unit

unitm = unit in a battle area to move

Figure 10. Formats for TACWAR Input Variables



① One-dimensional integer array with continuation cards.

② Two-dimensional real array.

③ Three-dimensional integer array.

④ Time-t unit data.

⑤ End-of-file flag.

Figure 11. Excerpt From Sample Input Data

c. I, J, K. (Columns 10-18.) These variables, taken together, give the user the flexibility to code the input in an order that is meaningful to him. In the case of a one-dimensional array, I, J, and K are left blank. The values on continuation card 0 are assumed to be 1-15 if the data are integral and 1-10 if the data are non-integral. Continuation card 1 provides the integral values 16-30 or non-integral values 11-20.

In the case of a two-dimensional array, the user may wish to input the data by rows or columns. To specify values for row I of an array, I is given in columns 10-12, and J and K are left blank. Similarly, in the case of a three-dimensional array, two indexes are specified on the card, thereby implying that (for this and every case) the values on the card correspond to the index which has been left blank.

d. Day Number. (Columns 79-80.) The day number is the day at which values are to be input to the model. Days 0 through 99 are coded as required. If a simulation is longer than 99 days, day 100 is coded as A0, 110 as B0, etc. Thus, day 107 would be coded as A7, day 115 as B5.

The following examples illustrate the procedure for inputting data of types 1 and 2.

Assume input is to be coded for array BB(16). Two cards are required. The first contains values 1-10 of BB, and the second, having a 1 in column 8, contains values 11-16 of BB.

Assume input is required for array NN(2,16). The coding can be accomplished in two ways. The first way is to specify I=1 and enter 15 values on the card, followed by I=1, continuation =1, and a single value. Then repeating the above for I=2. The second way of coding is to specify J=1 on a card and provide two values on the card, and repeat the process for J=2 through 16. Obviously, the first method requires fewer cards but the user may wish to use the second method because the data could be more easily verified.

Assume input is required for MMM(4,15,4). Specifying I and J on each card, or J and K, as many as 60 cards may be required, whereas by specifying I and K, a maximum of 16 cards would be required.

3.2.1.1.2 Unit Assignment Data. The special input format for arriving units as shown in figure 10 contains the following assignments:

a. If no preassigned location is indicated, the unit is considered available for assignment theater-wide.

b. If a region assignment is made, the following will occur: the number 99 in the sector field will assign the unit to the sector of main-attack of the indicated region; if the sector field is left blank, the unit assignment logic (section 2.2.14.2) will hold.

c. If a sector assignment is made, the following will occur: with the number 999 in the battle area field, the unit will be assigned to the first inactive battle area of the sector indicated (thus ignoring the move through the theater to the front, or portraying an airlift of units to the front); if the battle area field is left blank, the unit is assigned to the most rearward battle area of the sector indicated.

d. If a battle area assignment is made directly, the unit will be assigned to that battle area provided the proper side owns the battle area.

3.2.1.2 Working File ITTD (Time-T Data). When the input routine INP processes the input file MIT, it extracts all time-t data, and writes them to an unformatted file, ITTD, for later reading by subroutine TIMET during the course of the game. The records output by INP are of three types: (1) integer, real, or alphanumeric data inputs (2) unit information and (3) flag cards. File ITTD consists of records of types 1 and 2 arranged in ascending day numbers interspersed with flag cards which indicate the end of the group of cards for a specific day.

The contents of the records for integer, real or alphanumeric data are described below:

Word Number	Contents
1	Day number for which the following data is to be processed.
2	Variable type - 1 means integer 2 means real 3 not used 4 means alphanumeric
3	Code indicating processing procedure for information in record: 1 - means increment data value 0 - means replace data value
4	I1. The relative location of the first word of blank Common into which information in the record is to be loaded.
5	I2. The relative location of the last word in blank Common into which information is to be loaded.
6	I3. The increment by which I1 is increased to arrive at I2.
7-21	Information to be loaded into blank Common as indicated by I1, I2, and I3. (Note: The following limits apply to variables Integer - words 7 to 21 Real - words 7 to 16 Alphanumeric - words 7 to 21).

The contents of the records for unit information is slightly different than for the types shown above. The UNIT data is found in words 3-12 as shown below. On the right are the comparable card columns for UNIT data input as formatted on file MIT.

Word	Contents	Card Columns
1	Day Number	79-80
2	Code indicating variable type 3 = unit ~~data~~ arrival 4 = move existing unit	
3	Side 1 or 2 (for Blue or Red)	9
4	Unit ID number (integer)	13-15
5	Unit personnel strength (real)	19-24
6	Unit supplies on hand (real)	25-30
7	Unit type (integer)	35-36
8	Battle area assignment (integer)	40-42
9	Sector assignment (integer)	47-48
10	Region assignment (integer)	53-54
11	Nationality index (integer)	59-60
12	Sector tag (integer)	65-66

The flag cards contain two words of information as shown below:

Word Number	Contents
1	A "9999" in word 1 indicates the end of records for the day number currently being processed
2	Day number for the next day for which data are to be read. If no more data are to be read, then word 2 must contain a value higher than the highest day played in the game.

3.2.1.3 Input File IAD (Airbase Data). File IAD contains card image data describing the initial images of airbases. These data are preset values extracted from a TANDEM F airbase data tape and notionalized for seven aircraft types.

(See appendix D of reference 2.) These data are read by subroutine INP and stored in arrays IMAGE (15,201), IMAGE1 (11,201) and AF (2,201). Each card on this file begins with the index of the airbase associated with the data which follow on that card. Within each group, the indexes must be in sequence from 1 to 201 or an error message will be printed and the program terminated. The formats of the three sets of data are as follows:

Array	Format
IMAGE (15,201)	1X, I3, 1X, 15I5
IMAGE1 (11,201)	1X, I3, 1X, 11I5, 20X
AF (2,201)	I10, 2F20.6

The contents of each of these arrays are described in appendix E under the appropriate variable name.

3.2.2 Output Files. TACWAR writes output reports presenting various levels of statistical details on five output files. These reports include: (1) alphabetic listing of initial data in blank Common; (2) theater control initialized data; (3) tabular records of inputs read by the program; (4) detailed game reports; and (5) summary game reports. In addition, all diagnostics produced by TACWAR are written to the same file as summary game reports. Table 7 shows the file names used for each of these types of output and the routines or modules which utilizes each file. The five output files are described in the following subsections.

3.2.2.1 Output File JINP. The following sets of input are written to file JINP.

3.2.2.1.1 Alphabetic Listing of Initial Data. The input routine INP writes an alphabetic listing of the blank Common variables with their dimensions and initial data values on the JINP file. Figure 12 is a sample of this output. The user can suppress this output by setting variable NEPD(1) to 1 in the input data.

Table 7. Output Files Used in the TACWAR Model

Type of Output	File Names	Print Switch	User by Model or Routine Name
Alphabetic Listing of Initial Data in Blank Common	JINP	NEPD(1)	INP
Theater Control Initialized Data	JINP	NEPD(2)	TCTZ
Tabular Records of Inputs	JINP	NEPD(3)	GCOUT, SPLYOT, TCOUT, CHOUT, NUCOUT, TACQOT
Detailed Game Reports	JCON	IPRDO	Conventional Combat Models
	JCHEM		Chemical Model
	JNUC		Nuclear Model
Summary Game Reports	JSUM	IPRSO	PSAIR, PSUMMY, Chemical Model, Nuclear Model
Diagnostics	JSUM	N/A	INP, EIGENV, QRAFIL, TAG, APORTN, CHEMDAM, TC, ASSIGN

```
VARIABLE ----NLSHS (  8,  0,  0)
          0    0    0    0    0    0    0    0

VARIABLE ----NLSR  (  5,  0,  0)
          0    0    0    0    0

VARIABLE ----NNSC  (  1,  0,  0)
          1

VARIABLE ----NOCDAB(  2,  0,  0)
          7    7

VARIABLE ----NOCDPC(  2,  0,  0)
          0    0

VARIABLE ----NODIST(  2,  0,  0)
          0    0

VARIABLE ----NPCAF (  2,  0,  0)
          8    8

VARIABLE ----NPCDF (  2,  0,  0)
          8    8

VARIABLE ----NPCHD (  4,  2,  0)
          1    2
          1    1
          1    1
          0    2

VARIABLE ----NPCHS (  4,  2,  0)
          2    2
          2    2
          2    2
          2    2

VARIABLE ----NPCHT (  4,  2,  0)
          2    2
          2    2
          2    2
          2    2

VARIABLE ----NPERAG(  1,  0,  0)
          5

VARIABLE ----NPOSD (  4,  2,  0)
          1    1
          1    1
          0    0
          0    0

VARIABLE ----NPOSS (  5,  2,  0)
          2    2
          2    2
          2    2
          2    2
          0    0
```

Figure 12. Sample Alphabetic Listing of Input Variables

3.2.2.1.2 Theater Control Initialized Data. Subroutine TCTZ initializes numerous variables before the start of the game. These data are written to file JINP in the same format as that used for detailed reports. The user can suppress the writing of this data by setting NEPD(2) to 1 in the input data.

3.2.2.1.3 Tabular Records of Inputs. The record of model inputs written to file JINP consists of 64 tables listing selected time-zero data read from input file MIT. The user can suppress the printing of these tables by setting NEPD(3) to 1 in the input data. Figure 13 illustrates a sample input table. Table 8 lists the 64 input record tables.

3.2.2.2 Output Files JCON, JCHEM, JNUC (Detailed Reports). The detailed reports record changes made to input and working variables during the course of a game. Data for conventional, chemical, and nuclear warfare are written to files JCON, JCHEM, and JNUC, respectively. Detailed reports may be generated on as many as five different game cycles, as specified by the values of input array IPRDO. Use of the detailed reports requires a familiarity with the structure of the model and the definitions of the variables used, since the reports themselves consist of variable names and values, without reference to the meanings of the variables listed.

Simulation routines in all models produce detailed reports. Each routine which produces detailed reports tests the print switch IPRD during execution to determine if a detailed report has been requested on the current cycle. The value of IPRD is determined in TMAIN by a comparison of the current cycle and the values of IPRDO. When IPRD=1, each routine writes to the appropriate file the values it has computed. If IPRD=0 on the current game cycle, nothing is written on the file. As an aid to the model user, detailed reports are subdivided by headings that identify the particular values computed by each subroutine. Figure 14 illustrates a sample page from a detailed output report.

3.2.2.3 Output File JSUM (Summary Report). A summary report consisting of 34 tables is written on file JSUM. A report is always printed on the first cycle of a game and on the

TABLE 14 DIVISION LOCATION IN ACTIVE BATTLE AREA (BY SECTOR)--(DLABA(IDS,IS)

SIDE BLUE

SECTOR	DIVISION POSITIONS IN SECTOR (FROM LEFT TO RIGHT FACING THE FEBA)	
1	1	2
2	3	4
3	5	6
4	7	8
5	9	10
6	11	12
7	13	14
8	15	16

SIDE RED

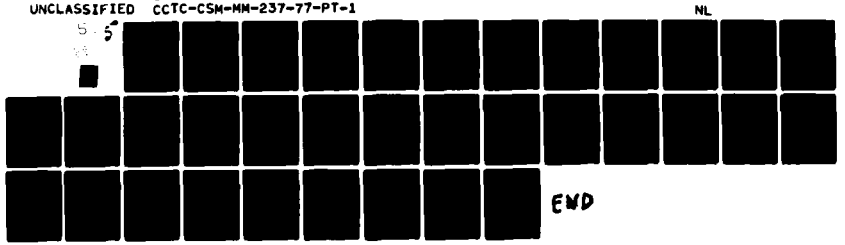
SECTOR	DIVISION POSITIONS IN SECTOR (FROM RIGHT TO LEFT FACING THE FEBA)								
1	46	47	48	49					
2	50	51	52	53	54				
3	55	56	57	58	59	60	61	62	63
4	64	65	66	67					
5	68	69	70	71					
6	72	73	74	75	76	77	78	79	
7	80	81	82	83	84	85	86	87	
8	88	89	90	91	92	93	94	95	

Figure 13. Sample Input Record Table

Table 8. Listing of Input Table Headings
(Part 1 of 5)

Table Number	Title of Input Report	Side
I1	Model Parameters	both
I2	Variables That Represent Physical Quantities	both
I3	Division Data	both
I4	Division Location in Active Battle Area (By Sector)	both
I5	Number of People and Weapons By Type Division (TOE Level)	both
I6	Ground Parameters	both
I7	Aircraft Munitions Load	both
I8	Supply Effectiveness Function (By Days of Supply On-Hand)	both
I9	Blue Percent Casualties	Blue
I10	Red Percent Casualties	Red
I11	Blue Effectiveness Functions (By Percent Personnel Strength)	Blue
I12	Red Effectiveness Functions (By Percent Personnel Strength)	Red
I13	Blue Standard Allocation of Air Munitions Against Weapons	Blue
I14	Red Standard Allocation of Air Munitions Against Weapons	Red
I15	Blue Standard Allocation of Weapons Against Weapons	Blue
I16	Red Standard Allocation of Weapons Against Weapons	Red

AD-A091 491 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 9/2
INSTITUTE FOR DEFENSE ANALYSES TECHNICAL WARFARE (TACWAR) MODEL--ETC(U)
SEP 77 M C FLYTHE, P FINNEGAN, J REIERSON
UNCLASSIFIED CCTC-CSM-MM-237-77-PT-1 NL



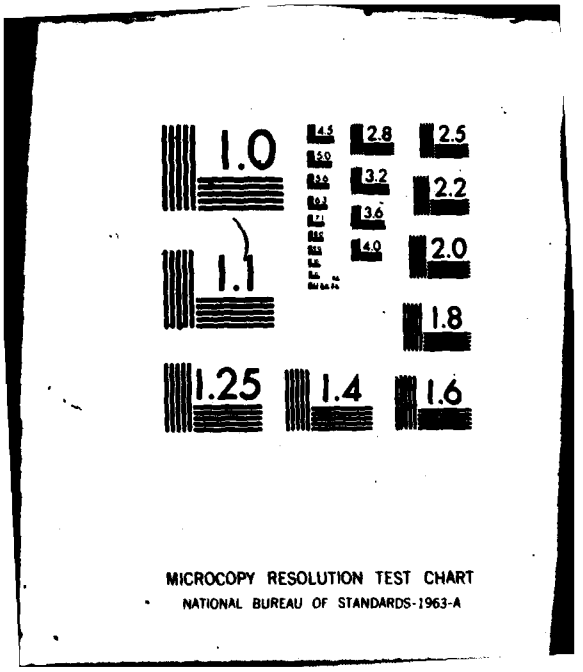

1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Table 8. (Part 2 of 5)

Table Number	Title of Input Report	Side
I17	Values for Blue, an Air Munition Against a Weapon	Blue
I18	Values for Red, an Air Munition Against a Weapon	Red
I19	Values for Blue, Weapon Against Weapon	Blue
I20	Values for Red, Weapon Against Weapon	Red
I21	FEBA Movement and Mobility Factors	both
I22	Movement Constraints Based on Flank Exposure and Security Force Ratio (Attacker to Defender)	both
I23	Kind of Posture and Terrain in Intervals in Sectors	both
I24	Number of Intervals, Boundary Latitudes, and Boundaries in Sectors	both
I25	Cumulative Ground Distance to Leading Edge of Battle Area (KM)	both
I26	Theater Structure, Theater Attacker, and Method for Computing Sectors of Main Attack	both
I27	Division and Subunit Width and Depth Parameters	both
I28	Parameters for Division Movement	both
I29	Parameters for Reinforcements, Withdrawals, and Replacements	both
I30	Percentages for Weapon Repair	both

Table 8. (Part 3 of 5)

Table Number	Title of Input Report	Side
I31	Blue Subunit Data	Blue
I32	Red Subunit Data	Red
I33	Supply Nodes Supplying Battle Areas by Sector	both
I34	Days of Supply On-Hand	both
I35	Supply Nodes - Inventory, Ownership and Location	both
I36	Supply Data and Consumption Rates	both
I37	Escalation State Characteristics	both
I38	Characteristics of a Preemptive Nuclear Strike	both
I39	Priority Listings of Preferred Nuclear Targets	both
I40	Parameters for Nuclear Targeting	both
I41	Index to Allowable Target Types (Nuclear)	both
I42	Criteria for Nuclear Escalation--Worsening Tactical Situation	both
I43	Nuclear Escalation Criteria - Initial or Increased Use of Nuclear Weapons by Other Side	both
I44	Parameters for Surface Burst Targets	both
I45	Protection Categories	both
I46	Elements for Calculating Blast and Radiation Damage to Personnel from Airburst and Surface Burst Weapons	both

Table 8. (Part 4 of 5)

Table Number	Title of Input Report	Side
I47	Vulnerability Numbers	both
I48	Nuclear Weapon System Resources	both
I49	Characteristics of a Preemptive Chemical Strike	both
I50	Priority Listings of Preferred Chemical Targets	both
I51	Index to Allowable Target Types (Chemical)	both
I52	Chemical Employment Criteria - Initial or Increased Use of Nuclear Weapons by Other Side	both
I53	Chemical Employment Criteria - Initial or Increased Use of Chemical Weapons by Other Side	both
I54	Weapon System Characteristics (Chemical)	both
I55	Chemical Weapon Systems, Agents, and Dissemination Modes	both
I56	Ground Sensor Data - Operating Altitude, Index for Target Acquisition Equation, Location	both
I57	Air Carrier Data - Velocity, Location, Distance to Target	both
I58	Detection Rates of Subunits by Air and Ground Sensors	both
I59	Army-Air Carrier and Sensor Data	both
I60	TOE Number and Factors for Ground Sensors and Army-Air Carriers	both

Table 8. (Part 5 of 5)

Table Number	Title of Input Report	Side
I61	Mission Assignments and Attrition Rates for Reconnaissance Aircraft	both
I62	Reconnaissance Carriers and Aircraft in Theater	both
I63	Reconnaissance Aircraft and Sensor Data	both
I64	Probability of Subunit Detection print message, not always printed	both

```
KPS(IS)
     1         1         2         1         1         3         1         1

IABAS(IS)
    49    42    51    52    53    54    55    48

IBALD(ID)
    49    49    42    42    51    51    52    52    53    53    54    54    55    55    48    48    58    59    62    56
    74    77    79    85     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0    49    49    49    49    42    42    42    42    42    42    51    51    51    51    51
    51    51    51    52    52    52    52    53    53    53    53    54    54    54    54    54    54    54    54    55
    55    55    55    55    55    55    55    48    48    48    48    48    48    48    48    41    41    34    34    34
    43    43    44    45    45    46    46    47    47    39    40    40    32    25    26    27    28    29    30    31
    23    18    19    21    13

ISHABA(IS,L)
     4    15    27    38    50    60    72    88
     3    14    26    37    49    59    70    85

IOSN(ISN)
     2     2     2     1     1     1     1     1     1     1     1     2     2     2     1     1     1     1     1     1
     1     2     2     2     2     2     1     1     1     1     1     1     1     1     1     2     2     2     2     1
     1     1     1     1     1     2     2     2     2     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     2     2     2     1     1     1     1     1
     2     2     2     2     2     2     0     1     1     1     1     1     1     1     1

NDFAB(L),NDRAB(L)
     2         2         2         1

NDRRP(IW,1,L)
   226.19      0.    138.42     94.03     24.33      7.01      5.83      8.49      2.77      5.37
   538.71    225.28  545.70    321.70     65.15     53.33      5.97      0.       12.56     22.77

NDRRP(IW,1,L)
   158.33      0.     96.90     47.01     13.16      2.80      2.33      7.64      1.59      5.37
   484.84    202.76  491.13    225.19     45.60     26.67      2.98      0.        6.28     15.94

NRPCZ(IW,1,L)
   917.86    200.00  941.53    347.01    313.16     64.21     65.50     50.85     31.39     50.00
   453.87    172.53  354.57    196.51    169.54     86.67     62.98     20.00     36.28     56.83

ICMS(IS,L)
     2     2     2     2     2     2     2     2
     1     2     2     2     2     2     2     2

ICMST(IS,L)
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0

 ID NVDAC(ID),NVDATS(IT),NVDDC(ID),NVDDTS(IT),EFFDA(ID),EFFDB(ID),VDDSF(ID)
  1   359.123   370.426   391.588   403.382     0.893     0.894   360.729
  2   359.123   370.426   391.588   403.382     0.893     0.894   360.729
  3   508.014   527.199   541.566   562.304     0.889     0.888   499.497
  4   359.126   370.426   391.555   403.382     0.894     0.895   361.075
  5   352.370   370.426   383.639   403.382     0.        0.        0.
  6   514.422   536.050   546.809   569.513     0.        0.435   247.546
  7   359.311   370.426   391.775   403.382     0.894     0.895   360.954
  8   359.311   370.426   391.775   403.382     0.894     0.895   360.954
```

Figure 14. Sample Page From Detailed Game Report

last cycle of a game. A report may be generated for as many as 30 additional game cycles as specified by the user when entering values for array IPRSO. A summary report is also printed automatically on any cycle chosen by the user for the writing of a detailed game report.

The summary report provides the user with a comprehensive overview of game status on the current cycle. Among the statistics given by the report are the current quantities of personnel, weapons, and supplies in the theater by location and by type, cumulative casualties in the form of a killer-target scoreboard, and force ratios by sector. The heading of each of the 34 tables gives the table number and the game cycle on which the table was generated. Figure 15 illustrates a sample summary table, and table 9 lists the headings of the summary tables. Tables N1 through N8 are produced by the nuclear model, C1 through C8 by the chemical model, A1 through A8 by subroutine PSAIR and S1 through S10 by subroutine PSUMMY. The nuclear and chemical tables will be printed only if nuclear or chemical weapons are played.

3.3 Program Variables

Appendix E contains an alphabetic listing of TACWAR program variables with their definitions. Only variables which are passed from one subroutine to another are included, i.e. local variables have been omitted.

Appendix F contains a list of the input and summary working variables grouped according to function within a given submodel. These functional groupings approximate types of operations, strategies, resource levels, parameters of specific kinds, index values, and basic mission assignments. Appendix G contains cross-reference tables of all Common variables and the subroutines that use or modify them.

TABLE S4 CYCLE 6 PERSONNEL AND WEAPONS IN ACTIVE BATTLE AREA OF EACH SECTOR

BLUE	SECTOR 1	2	3	4	5	6	7	8	TOTAL
PERSONNEL	21857.	47511.	26606.	27084.	57551.	29503.	52825.	0.	262737.
WEAPONS									
BW1	1136.	1192.	955.	1231.	1214.	705.	1190.	0.	7622.
BW2	0.	0.	0.	0.	0.	0.	0.	0.	0.
BW3	300.	812.	457.	325.	860.	597.	962.	0.	4312.
BW4	722.	934.	654.	785.	1014.	448.	689.	0.	5246.
BW5	42.	310.	113.	47.	327.	172.	282.	0.	1294.
BW6	69.	82.	62.	70.	82.	27.	27.	0.	418.
BW7	48.	151.	66.	48.	152.	125.	163.	0.	754.
BW8	20.	36.	21.	19.	34.	35.	58.	0.	224.
BW9	0.	0.	28.	0.	73.	55.	86.	0.	242.
BW10	0.	0.	23.	0.	70.	54.	83.	0.	230.

RED	SECTOR 1	2	3	4	5	6	7	8	TOTAL
PERSONNEL	47788.	67880.	55952.	44192.	60031.	71687.	94256.	98012.	539798.
WEAPONS									
RW1	622.	1207.	1566.	1484.	1937.	1412.	1810.	2846.	12883.
RW2	708.	856.	724.	647.	745.	804.	1082.	1134.	6701.
RW3	1579.	1889.	1556.	1254.	1546.	1874.	2481.	2400.	14579.
RW4	529.	1015.	1298.	1384.	1749.	1135.	1494.	2528.	11133.
RW5	97.	189.	245.	244.	302.	210.	275.	433.	1995.
RW6	376.	593.	1200.	1904.	2658.	712.	968.	1058.	11470.
RW7	11.	19.	23.	8.	27.	30.	40.	44.	204.
RW8	0.	0.	0.	0.	0.	0.	0.	0.	0.
RW9	0.	0.	192.	251.	424.	207.	282.	544.	1899.
RW10	0.	0.	188.	243.	413.	201.	276.	528.	1850.

Figure 15. Sample Summary Game Report

Table 9. Listing of Summary Report Headings
(Part 1 of 3)

Table Number	Title of Summary Report	Side
N1	Escalation States for Nuclear	both
N2	Nuclear Delivery System Availability	both
N3	Nuclear Warhead Availability	both
N4	Target Priorities	both
N5	Nuclear Weapon Systems and Number of Available Warheads in Priority Order (by Side and Sector)	both
N6	Maximum Number of Nuclear Targets (subunits) in Active Battle Area (by Sector)	both
N7	Assignment of Nuclear Weapons to Targets (by Sector)	both
N8	Nuclear Damage Assessment	both
C1	Employment Levels for Chemical	both
C2	Chemical Delivery System Availability	both
C3	Chemical Warhead Availability	both
C4	Target Priorities	both
C5	Weapon System (with Number of Rounds) in Priority Order for Each Agent and Dissemination Mode (by Side and Sector)	both
C6	Maximum Number of Chemical Targets (Subunits) in Active Battle Area	both
C7	Assignment of Chemical Weapons to Targets	both
C8	Chemical Damage Assessment	both

Table 9. (Part 2 of 3)

Table Number	Title of Summary Report	Side
A1	Blue Aircraft (Undamaged), QRA and Helicopters in Theater at End of Cycle	Blue
A2	Red Aircraft (Undamaged), QRA, and Helicopters in Theater at End of Cycle	Red
A3	SAMs in Repair and Replacement Pools and Total Missiles in Theater at End of Cycle	both
A4	Air Munitions Expended on CAS and INTD, and Aircraft Shelters Operational and Destroyed	both
A5	Blue Cumulative Aircraft Killed on Ground	Blue
A6	Red Cumulative Aircraft Killed on Ground	Red
A7	Cumulative Successful Attack Sorties	both
A8	SAMs Suppressed During Cycle and Cumulative SAMs Damaged and Repaired	both
S1	Attacker Indices and Geographical Quantities	both
S2	FEBA and Force Ratios	both
S3	Divisions in Theater	both
S4	Personnel and Weapons in Active Battle Area of Each Sector	both
S5	Division Statistics	both
S6	People and Weapon Replacement Pool Inventories	both

Table 9. (Part 3 of 3)

Table Number	Title of Summary Report	Side
S7	Cumulative Supply Consumption and Losses	both
S8	Cumulative Casualties and Weapon Losses Due to Ground Combat	both
S9	Cumulative Killer-Target Score-Board (Ground Combat)	both
S10	Supply nodes -- Owner, Inventory, Arrivals, and Cumulative Supplies Destroyed	both

SECTION 4. PROGRAM ASSEMBLY, LOADING, AND MAINTENANCE PROCEDURES

This section provides information concerning the procedures used to maintain the TACWAR system files. Also included is a discussion of the warning messages and error messages flagged by TACWAR.

4.1 Procedures

This subsection describes the procedures for maintaining the TACWAR model. Topics included in the discussion are: (1) executing offline routines to produce new input data; (2) creating a new H* file for the TACWAR model; (3) and modifying the TSS JCL file for remote terminal execution. Appendix D contains procedures for executing TACWAR in both the batch and remote terminal modes and for updating the input files.

4.1.1 Offline Routines. There are three offline routines that have been developed for use by various elements of the TACWAR model. One routine, COMM, is a preprocessor which produces Data statements for the input routine INP. The other two routines, NOTION and AFLDS, process airbase data for input to the model. These routines are described in the following subsections.

4.1.1.1 Routine for Changing Blank Common. Subroutine INP, as originally designed, required an input data set that described the order of variables in blank Common, their respective dimensions and type, and an updating procedure for input at time-t. Then when Common was modified, a corresponding change had to be made to this set of data. This type of operation became the source of a great number of errors. The procedure to be described here is a method which reduces the chance of making these errors. The ingredients of this procedure are listed below.

a. INP contains a set of Data statements providing the set of specifications described above.

b. INP must be recompiled with a changed set of Data statements whenever Common is modified in any way.

c. The set of Data statements is generated by an independent program, COMM*, that accepts as input the Common deck itself as well as statements noting exceptions.

d. The exceptions to be noted are with respect to the typing of variables and the time-t processing procedure. The conventions for typing of variables are those accepted as the standard for FORTRAN, i.e., names beginning with the letters I through N are integer and all others are assumed real. If a variable contains alphanumeric information, it must be noted as ALPHA. If a variable is to be excepted as real with an initial letter of I through N, it must be noted as REAL. Similarly, a variable that does not begin with letters I through N (but is of integer type) must be noted as INTGER. The time-t processing procedure of a variable is always assumed to be a reaplacement of the current value. If the value of the variable being processed is to be incremented, it must be noted as INCREM.

e. The output Data statements are written to a file for subsequent recompilation of INP.

Figure 16 shows the procedure for updating the TACWAR model to reflect changes to blank Common. First, the original Common block is updated either through time-sharing or batch editing procedures and the updated Common (not including the type cards) is written to file 10. This data, followed by the exceptions which reside on file 25, are input to Program COMM. The Data statements generated by COMM are written to file 15 which is then used to replace the previous Data statements in subroutine INP. The updated Common block is also used to replace the previous Common and all changed routines are recompiled.

The exceptions data on file 25 are in free format in columns 6-72. Program COMM processes this file as if it were a continuation of Common file 10. Therefore, column 6 of each file 25 record must contain a non-blank character denoting continuation. The first string of characters to be input on file 25 is the 5-character string ,END, which terminates the reading of file 10. The exceptions on file 25 then follow in the format OPER, V_1, V_2,..., V_n,. The exceptions operator

* see C3C letter Report dtd 21 July 78, CCTC TO 634, Subtask 1 TACWAR CODE REDUCTION

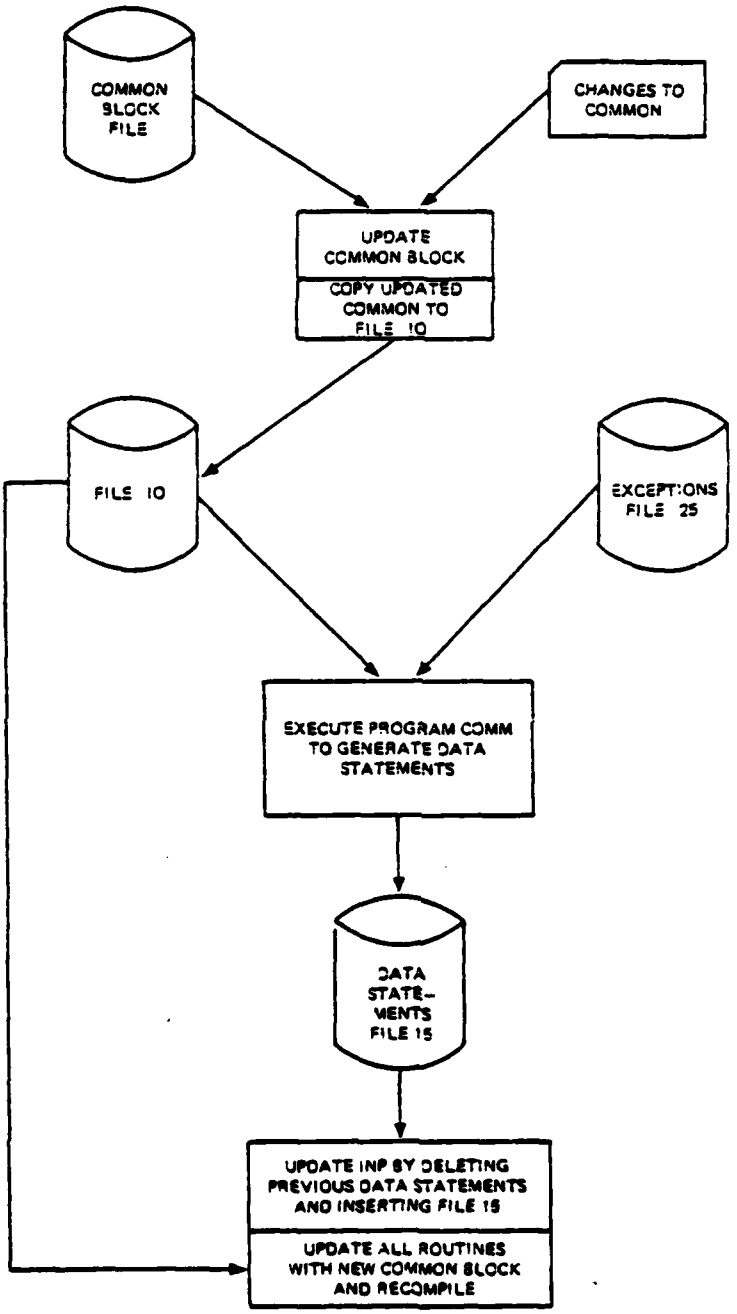


Figure 16. Procedures for Updating TACWAR Routines to Reflect Changes to Blank Common

OPER can take on the values ALPHA, INTGER, REAL, or INCREM, as defined previously in this subsection. The V_i's that follow are the names of the exception variables of the type specified by OPER. Operator names ALPHA, INTGER, REAL and INCREM cannot be used as variable names. The exceptions are terminated by the string END.

EXAMPLE OF EXCEPTION INPUT

```
*  , END, ALPHA, A, KKK, REAL,NSUTD,
*         INTGER,  CCZZ,XYZ,
*  INCREM,P,Q,       L
*  ,END
```

Program COMM is comprised of a main routine and two subroutines, IGF and RD. Appendix C contains a listing of these routines. Subroutine RD is called by IGF to read, one line at a time, the input data on files 10 and 25. IGF identifies variable names and dimensions from the Common block, and operators and variable names from the exceptions by testing the data line, character by character. COMM uses the groups of characters identified by IGF to load array IV with the information which will be contained in Data statements for array IVARQ in INP. IVARQ is a two dimensional array which contains 7 items of information for each variable in blank Common. The contents of IVARQ, which are the same as those in IV, are described in table 10 and are listed by the value of its second index.

Program COMM contains two error messages. If a variable name encountered on the exception file is not found in the list of blank Common variables, the variable name will be considered invalid. A message will be printed to indicate the invalid variable name and its operator; the program then ignores this exception and continues to process the exception file. In the event that the exception file does not terminate appropriately, as described earlier in this subsection, the program stops with the message, "PREMATURE EOF ON FILE 25. PROGRAM TERMINATED". In this case, no Data statements are generated since COMM processes all of the input data before producing any output.

Table 10. Definition of Array IVARQ

Second Index Value	Description
1	Variable name
2	Size of first dimension
3	Size of second dimension
4	Size of third dimension
5	The relative ending location of the preceding variable in the Common block
6	Two digit flag indicating variable type and whether it is to be replaced or incremented on later cycles. The value is determined as follows: The units digit is 0 replaced 1 incremented The tens digit is 1 integer 2 real 4 alphanumeric
7	Flag indicating ending location for this variable in the Common block and the number of index arguments. The value is 10 times the location plus the number of arguments.

4.1.1.2 Routines for Reading Airbase Data Tapes. Programs NOTION and AFLDS are two CDC 6400 FORTRAN routines written to retrieve airbase data from a TANDEM F data type, which is maintained by the Defense Nuclear Agency. (See appendix D of reference 2.) These two routines are described in the following paragraphs.

4.1.1.2.1 Program NOTION. This program is designed to read an airbase data tape that contains for specific airbases various information items such as number of aircraft by type, number of shelters, number of parking areas, etc. Program NOTION then creates two new tapes, based on the original data, with data depicting either seven or 10 notional classes of aircraft rather than the many actual types existing on the original tape.

4.1.1.2.2 Program AFLDS. This program reads the data tape created by program NOTION and punches out Data statements as required by subroutines UNPAFO and UNPAF3. The program can also punch out data statements as required by function AF. The data in these statements are packed, using the CDC 6400 60 bit word size (although only 48 bits are utilized). For operation on the CDC 6400, it was assumed that the number of aircraft on any airbase would not exceed 1023 (i.e., $2^{10}-1$). The converted TACWAR model does not use subroutines UNPAFO and UNPAF3, and function AF which unpack the airbase data. The HIS version reads the data from card images and stores the information in arrays IMAGE, IMAGE1 and AF. Program AFLDS was altered on the CDC 6400 to punch the airbase data on cards in the correct formats for reading by subroutines INP on the HIS computer. These cards are contained on input file IAD.

4.1.2 TACWAR H* File. The TACWAR model is executed from system loadable files. Table 11 describes the files used by TACWAR and those used to maintain the model.

The TACWAR model is executed from the H* file, TWHSTAR. When modifications are made to the TACWAR program logic, a new H* file must be created. First, the necessary changes must be made to an existing TACWAR K* source tape and R* object tape through the FILEDIT system. (See reference 13.)

Table 11. TACWAR System Files

File-name	Size (LLINKS)	Mode	Purpose
TWDB	120	Sequential	User selected input data for TACWAR
ABD	120	Sequential	Airbase data
TWHSTAR	1200	Random	H* file of TACWAR model
TWTSS	10	Sequential	JCL file for executing TACWAR by remote terminal

Next, the contents of the changed tape must be loaded for execution and the resulting H* load module saved on file TWHSTAR. Figure 17 illustrates the deck structure used to create this file.

4.1.3 TSS JCL File. Execution of the TACWAR model can also be initiated from a remote terminal through the use of a JCL file in TSS format. An example of such a file is shown in figure 18. The use of a parameter card allows for quick alternation of five options on the JCL file, each of which is listed below.

Parameter Number	Option
#1	file name of TACWAR H* file
#2	file name of TACWAR user-selected input data file
#3	file name of TACWAR airbase data file
#4	maximum processing time for execution
#5	maximum number of lines to be printed during execution.

To alter the TSS JCL file, the maintenance programmer must log on to a terminal and proceed through the steps listed below. User responses are underlined.

SYSTEM?

CARDIN

OLD OR NEW

OLD catalog-file string for TSS JCL FILE

Example: OLD 674IDP00/SAYLOR/TACWAR/TWTSS.

```
$       IDENT   installation dependent
$       USERID  installation dependent
$       FILEDIT SOURCE,OBJECT,UPDATE
$       TAPE    M*,X1S,,xxxxx,,OLDSOURCE
$       TAPE    *R,X2S,,yyyyy,,OLDOBJECT
$       FILE    R*,X3S,150L
$       FILE    K*,X4S,150L
$       DATA    *C,,COPY
        .
        .
        .
        FILEDIT DIRECTIVES
        .
        .
        .
$       ENDEDIT
$       ENDCOPY
$       CONVER
$       FILE    IN,X3S
$       TAPE    OT,X2D,,yyyyy
$       CONVER
$       FILE    IN,X4R
$       TAPE    OT,X1D,,xxxxx
$       EXECUTE
$       LIMITS  30,80K,-4K,30K
$       FILE    R*,X3R
$       PRMFL   10,R,S,catalog-file string for user input data
$       PRMFL   20,R,S,catalog-file string for airbase data
$       FILE    15,X5S,100L
$       PRMFL   H*,R/W,R,catalog-file string for TACWAR H*
$       SYSOUT  04
$       SYSOUT  07
$       SYSOUT  08
$       SYSOUT  09
$       ENDJOB
***EOF
```

Figure 17. Deck Structure for Creating TACWAR H* File

```
010$##NORM
020$:IDENT:4981,TWTSS,315,CAPTSAYLOR,674,10
030$:PARAM:TWHSTAR,TWDB,ABD,30,30K
040$:PROGRAM:RLHS
050$:PRMFL:H*,R,R,674IDP00/CSC/SAYLOR/TACWAR/#1
060$:LIMITS:#4,80K,-4K,#5
070$:PRMFL:10,R,S,674IDP00/CSC/SAYLOR/TACWAR/#2
080$:PRMFL:20,R,S,674IDP00/CSC/SAYLOR/TACWAR/#3
090$:FILE:15,X1S,100L
100$:SYSOUT:04
110$:SYSOUT:07
120$:SYSOUT:08
130$:SYSOUT:09
140$:ENDJOB
```

Figure 18. Example of JCL File for Executing TACWAR From the Terminal

LIST mmmm-nnnn

The user may elect to display portions of the JCL file on his terminal. In this case, lines mmmm through nnnn will be displayed. (The initial alter number precedes the hyphen and the final alter number follows it.)

mmmm line of JCL

.

.

.

nnnn line of JCL

*

mmmm line of JCL

The user inputs an alter number (mmmm) followed by a line of JCL. If the alter number does not exist, the number and its line of JCL are entered in proper numerical sequence into the file. If the alter number does exist, the new line of JCL replaces the old line of JCL.

DELETE mmmm-nnnn

To delete lines of JCL, the user enters the word DELETE, followed by the alter numbers of those lines to be deleted.

RESAVE file name for TSS JCL file

The updated version of the JCL file will now replace the old version.

Example: RESAVE TWTSS

TWTSS SAVED

*

BYE

The user will be disconnected from the terminal.

4.2 Warning and Error Messages

This subsection describes the various abnormal conditions detected by the TACWAR model. The following subsections describe the warning and error messages printed by the model when these conditions are encountered. All warning and error messages appear on output file JSUM.

4.2.1 Warning Messages. Warning messages are written by several subroutines in the TACWAR Model.

4.2.1.1 Subroutine EIGENV Messages. Subroutine EIGENV writes the following warning message:

> "NOTE--METHOD FOR COMPUTING EIGENVALUE DID NOT CONVERGE WITHIN MAXIMUM ALLOWED NUMBER OF ITERATIONS."

This message is produced when the eigenvalue developed by the EIGENV routine has failed to converge within user-specified limits. Before attempting another execution of the model, the user should adjust the iteration limit MNIE and/or the convergence tolerance EFCE.

4.2.1.2 Subroutine INP Messages. Subroutine INP writes two warning messages during processing of user-specified data from file MIT. The messages are as follows:

a. INPUT NAME NOT IN LIST--CARD IGNORED (followed by the card image)

b. IMPROPER CODING OF DIMENSIONAL VARIABLE-CARD BELOW IGNORED (followed by the card image).

These messages warn the user that certain cards of input data could not be processed by the input routine. If failure to process these cards will affect the results of the run, the user should check the format on these cards, correct them or change blank Common and/or the IVARQ Data statements as appropriate, and resubmit the job.

4.2.1.3 Subroutine CHEMDAM Messages. Subroutine CHEMDAM prints the following warning message:

```
WARNING--FOR MISSION NO. i OF SIDE k ITERATION LIMIT
EXCEEDED IN CHEMICAL DAMAGE ASSESSMENT.
CHEM PROT = ic, PHYS PROT - ipp, DOSE LEVEL = j,
CAS INDEX - kk.
DISTANCE SET TO SIMPLIFIED CALCULATION VALUE.
```

The purpose of this message is to inform the user that 20 iterations using Newton's method and 180 iterations using the interval halving method did not converge on a solution for the crosswind integrated dose. The program has selected to use the solution found with the simplified Calder-Sutton or Porton equations. The user should note the possibility of a lesser degree of accuracy in the solution.

4.2.1.4 Subroutine QRAFIL Messages. Subroutine QRAFIL writes the following three warning messages:

a. "SIDE k HAS A QRA SHORTFALL OF x AIRCRAFT ON NOTIONAL FORWARD AIRBASE IN SECTOR i ON CYCLE j."

b. "SIDE k HAS A QRA SHORTFALL OF x AIRCRAFT ON NOTIONAL REAR AIRBASE IN SECTOR i ON CYCLE j."

c. "SIDE k HAS A QRA SHORTFALL OF x AIRCRAFT IN COMMZ ON CYCLE j."

These messages warn the user that there were not enough aircraft on a particular airbase to fill the user-specified level of QRA aircraft. The user should remember this when reviewing the results of the model. The program continues processing after printing any of these messages.

4.2.2 Error Messages. There are several error conditions which, when encountered, will cause the program to terminate. Each encounter is flagged by a "STOP i" which serves as an error message to the user. The following subsections describe the error conditions and their associated messages.

4.2.2.1 STOP 201 (in INP). The input routine INP reads from the airbase data file, IAD, data for three arrays: IMAGE, IMAGE1, and AF. The first item on each data card is the index of the airbase with which the remaining items on the card are associated. For each array the data cards must be arranged in ascending order of airbase index, from 1 to 201. If any one of the three groups of data is not in the proper order, the program will terminate with "STOP 201" preceded by the following message:

> "I/O ERROR DETECTED IN AIRBASE DATA. ERROR IN SET i INDEX VALUE j. PROGRAM TERMINATED."

where i is 1, 2 or 3 to indicate the data for IMAGE, IMAGE1, or AF, respectively; and j is the airbase index which was the first card out of order. This message will allow the user to locate the first card on file IAD which is out of order. The user can then correct the data set and resubmit the job.

4.2.2.2 STOP 1 (in EIGENV). The subroutine EIGENV prints the following message before terminating the program with "STOP 1".

> "VALUE OF BLUE REFERENCE WEAPON EQUALS 0. RERUN SELECTING A DIFFERENT WEAPON FOR IWUCE."

This message is generated when the model is computing weapon values by means of antipotential potential methodology, but the user has selected a reference weapon with a value too small for practical computation. Before attempting another execution of the model, the user must adjust the value of index IWUCE to reflect the choice of another weapon as the reference weapon.

4.2.2.3 STOP 2 (in TAG). This STOP occurs whenever the FEBA in any sector has moved the entire distance through the sector. No additional message is printed.

4.2.2.4 STOP 60 (in APORTN). This STOP occurs when more than 60 actual airbases are to be aggregated into one notional airbase.

4.2.2.5 STOP 11111 (in TC). This STOP occurs when the user tries to assign a new unit to a sector that has already reached its defined limit. The current limit is set at 30 divisions per sector (total of both sides). No additional message is printed.

4.2.2.6 STOP 133 (in ASSIGN). This STOP occurs when the user tries to assign a new unit to a "non-existent" battle area (i.e., a battle area with an identification number less than or greater than the number of battle areas in the theater (NBA)). The situation may occur when time-t data is incorrect or when the game situation is such that the FEBA location precludes the existence of first or second inactive battle areas for one side.

THIS PAGE INTENTIONALLY LEFT BLANK

REFERENCES

1. V. Srinivasan and G.L. Thompson, A FORTRAN V Code for the Primal Transportation Algorithm, Carnegie-Mellon University, Pittsburgh, Pa., June 1971.

2. Institute for Defense Analyses, IDA TACNUC Model: Theater-Level Assessment of Conventional and Nuclear Combat Volume II: Detailed Description, WSEG Report 275 (including Report R-211), October 1975.

3. A. Charnes and W.W. Cooper, Management Models and Industrial Applications of Linear Programming, Volumes I and II, John Wiley & Sons, Inc., New York, N.Y., 1961.

4. F. Glover and D. Klingman, "Locating Stepping Stone Paths in Distribution Problems via the Predecessor Index Method," Transportation Science, No. 4, pp. 220-225, 1970.

5. William H. Holter, NATO Combat Capabilities Analysis II (COMCAP II), Appendix D, Secret GCR Report OAD-CR-8, McLean, Va., August 1973.

6. Alan F. Karr, On a Class of Binomial Attrition Processes, IDA Paper P-1031, June 1974.

7. Leo A. Schmidt, Jr., A Numerical Fit to an Algorithm Which Computes Prompt Fission Product Gamma Radiation Doses, Working Paper WP-41, TACNUC Model Study, Institute for Defense Analyses, Arlington, Va., December 1975.

8. Defense Intelligence Agency, Mathematical Background and Programming Aids for the Physical Vulnerability System for Nuclear Weapons, DI-550-27-74, 1 November 1974.

9. U.S. Department of Commerce, National Bureau of Standards, Handbook of Mathematical Functions, Applied Mathematics Series No. 55, June 1964.

10. Ralph Mason, Computer Approximations of the Circular Log Normal Damage Function, National Military Command System Support Center (NMCSSC), Technical Memorandum TM-78-72, 7 August 1972.

11. Institute for Defense Analyses, IDA Tactical Warfare Model: A Theater-Level Model of Conventional, Nuclear, and Chemical Warfare (formerly the IDA TACNUC Model), "Documentation Part I Modifications," Report R-211, Draft.

12. Lowel Bruce Anderson, Jerome Bracken, and Eleanor L. Schwartz, Revised OPTSA Model, "Methodology," Volume I, Paper, P-1111, Institute for Defense Analyses, Arlington, Va., September 1975.

13. Harvey M. Wagner, Principles of Operation Research, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1969 and 1975.

14. Honeywell Information Systems, Inc., Source and Object Library Editor, Order No. BJ71, Rev. 0, Wellesley Hills, Mass., October 1971.

15. National Military Command System Support Center, Institute for Defense Analysis Ground-Air Model (IDAGAM II), Computer System Manual CSM UM 201-75, Washington, D.C., October 1976.

16. Honeywell Information Systems, Inc., Time-Sharing Text Editor, Order No. BR40, Rev. 1, Wellesley Hills, Mass., June 1973.

BIBLIOGRAPHY

Honeywell Information Systems, Inc., Control Cards Reference Manual, Order No. BS19, Rev. 2, Wellesley Hills, Mass., February 1973

Honeywell Information Systems, Inc., FORTRAN, Order No. BJ67, Rev. 1, Wellesley Hills, Mass., March 1973

Honeywell Information Systems, Inc., GCOS Time-Sharing System General Information, Order No. BS01, Rev. 1, Wellesley Hills, Mass., July 1973

Honeywell Information Systems, Inc., GCOS Time-Sharing System Programmer's Reference Manual, Order No. BR39, Rev. 1, Wellesley Hills, Mass., November 1971

Honeywell Information Systems, Inc., GCOS Time-Sharing System Terminal/Batch Interface Facility, Order No. BR99, Rev. 1, Wellesley Hills, Mass., January 1972

Honeywell Information Systems, Inc., General Loader, Order No. BN90, Rev. 0, Wellesley Hills, Mass., March 1972

Institute for Defense Analyses, IDA TACNUC Model: Theater Level Assessment of Conventional and Nuclear Combat Volume 1: Executive Summary, WSEG Report 275, Arlington, Va., October 1975

Institute for Defense Analyses, IDA Tactical Warfare Model: A Theater-Level Model of Conventional, Nuclear and Chemical Warfare (formerly the IDA TACNUC Model) Volume III: Documentation Part II: Program Description, Report R-211, Arlington, Va., Draft

THIS PAGE INTENTIONALLY LEFT BLANK

DISTRIBUTION

ADDRESSEE	NO. OF COPIES
CCTC Codes	
C124 (Reference and Record Set)	3
C124 (Stock)	6
C315	15
DCA Code 205	1
Documentation Center C126 ATTN: Ms. Palmer 11440 Isaac Newton Square Reston, VA 22090	1
WWMCCS ADP Management Division, J-3 ATTN: Mr. Goertzel The Pentagon Washington, DC 20301	1
Defense Documentation Center Cameron Station Building 5 Alexandria, VA 22314	2
Assistant to the Secretary of Defense for Atomic Energy Room 3C128, The Pentagon Washington, DC 20301	1
Defense Advanced Research Projects Agency Director, Tactical Technology 1400 Wilson Boulevard Arlington, VA 22209	1
Defense Nuclear Agency ATTN: Col. M. Johnsrud Director, Net Assessment Studies Office 6801 Telegraph Road Alexandria, VA 20305	1
Studies, Analysis and Gaming Agency, GPFD The Pentagon Washington, DC 20301	15
Deputy Under Secretary of the Army (OR) Room 2E621, The Pentagon Washington, DC 20301	1

DISTRIBUTION

ADDRESSEE	NO. OF COPIES
Department of the Army Office of the Chief of Research, Development and Acquisition ATTN: DAMA-RAZ-A Room 3E412, The Pentagon Washington, DC 20301	1
Department of the Army Office of the Deputy Chief of Staff for Operations and Plans ATTN: DAMO-ZD, Mr. Louer The Pentagon Washington, DC 20301	1
U.S. Army Concepts Analysis Agency (CAA) ATTN: MOCA-MR 8120 Woodmont Avenue Bethesda, MD 20014	2
Director, TRADOC Systems Analysis Activity ATTN: LTC John Hesse White Sands Missile Range New Mexico 88002	15
Commandant, U.S. Army War College Carlisle Barracks Pennsylvania 17013	1
Office of the Chief of Naval Operations Systems Analysis Division (NOP96C) Room 4A526, The Pentagon Washington, DC 20301	
Commanding General Marine Corps Development & Education Command ATTN: Director, Development Center Quantico, VA 22134	1
Office of the Assistant Secretary of the Air Force (Research and Development) Room 4E968, The Pentagon Washington, DC 20301	1

DISTRIBUTION

ADDRESSEE	NO. OF COPIES
Office of the Assistant Chief of Staff, USAF (Studies and Analysis) Room 1E388, The Pentagon Washington, DC 20301	1
U.S. Arms Control and Disarmament Agency 21st Street and Virginia Avenue, N.W. Washington, DC 20451	1
Institute for Defense Analyses ATTN: Mr. Kerlin 400 Army Navy Drive Arlington, VA 22202	5
SHAPE Technical Center ATTN: Mr. Rex Goad APO New York 09159	2
Computer Science Corporation ATTN: Ms. Flythe 400 Army Navy Drive Arlington, VA 22202	5
	84 TOTAL

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (when data entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT. ACCESSION NO.	3. RECEIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Institute for Defense Analyses Tactical Warfare (TACWAR) Model Program Maintenance Manual		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR (s) Flythe, Mary Catherine; Finnegan, Pat; Reierson, Jim; Truszcynski, Peter; Tsang, Theresa; and Lee, John		8. CONTRACT OR GRANT NUMBER (s) DCA 100-74-C-0002
9. PERFORMING ORGANIZATION NAME & ADDRESS Computer Sciences Corporation 400 Army Navy Drive Arlington, VA 22202		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME & ADDRESS Command and Control Technical Center Support Center (C315) The Pentagon Washington, D.C. 20301		12. REPORT DATE 6 September 1977 13. NUMBER OF PAGES 947
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASS/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in block 20, if different from report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (continue on reverse side if necessary and identify by block number)

ground-air warfare, nuclear warfare, chemical warfare, theater-level model, military operations research, defense planning, ground forces, tactical air forces

20. ABSTRACT (continue on reverse side if necessary and identify by block number) The Institute for Defense Analyses Tactical Warfare (TACWAR) model is a fully-automated combat simulation that can be used to assess the interaction of combat forces employing conventional, nuclear, and chemical weapons in a theater-wide campaign. This document presents the information necessary for programmer personnel to maintain the TACWAR model.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE 926 UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (when data entered)